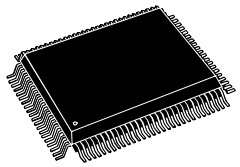
ST92F120

8/16-BIT FLASH MCU FAMILY WITH RAM, EEPROM AND J1850 BLPD

DATASHEET

- **Memories**
  - Internal Memory: up to 128 Kbytes Single Voltage FLASH, 2 to 4 Kbytes RAM, 1K byte emulated EEPROM (E3PROM™)
  - 224 general purpose registers (register file) available as RAM, accumulators or index pointers
- **Clock, Reset and Supply Management**
  - Register-oriented 8/16 bit CORE with RUN, WFI, SLOW, HALT and STOP modes
  - 0-24 MHz Operation (internal Clock), 4.5 - 5.5 Volt voltage range
  - PLL Clock Generator (3-5 MHz crystal)
  - Min. instruction time: 83 ns (24 MHz int. clock)
- **Interrupt Management**
  - 77 I/O pins
  - 4 external fast interrupts + 1 NMI
  - Up to 16 pins programmable as wake-up or additional external interrupt with multi-level interrupt handler
- **Timers**
  - 16-bit Timer with 8 bit Prescaler, able to be used as a Watchdog Timer with a large range of service time (HW/SW enabling through dedicated pin)
  - 16-bit Standard Timer that can be used to generate a time base independent of PLL Clock Generator
  - Two 16-bit independent Extended Function Timers (EFTs) with Prescaler, 2 Input Captures and two Output Compares
  - Two 16-bit Multifunction Timers, with Prescaler, 2 Input Captures and 2 Output Compares



PQFP100

- **Communication Interfaces**
  - 2 Serial Communication Interfaces with asynchronous and synchronous capabilities. Software Management and synchronous mode supported
  - Serial Peripheral Interface (SPI) with Selectable Master/Slave mode
  - J1850 Byte Level Protocol Decoder (JBLPD) (on J versions only)
  - Full I²C multiple Master/Slave Interface supporting Access Bus
- **8-bit Analog to Digital Converter** allowing up to 16 input channels
- **DMA Controller** for reduced processor overhead
- **Instruction Set**
  - Rich Instruction Set with 14 Addressing Modes
  - Division-by-zero trap generation
- **Development Tools**
  - Versatile Development Tools, including Assembler, Linker, C-Compiler, Archiver, Source Level Debugger, Hardware Emulators and Real Time Operating System

DEVICE SUMMARY

Features	ST92F120V9	ST92F120JV9	ST92F120V1	ST92F120JV1
FLASH - bytes	60K		128K	
RAM - bytes	2K		4K	
E3PROM - bytes	1K			
Network Interface	--	J1850	--	J1850
Temp. Range	-40° C to 105° C or -40°C to 85° C			
Packages	PQFP100			

Rev. 2.6

Table of Contents

Table of Contents

1 GENERAL DESCRIPTION

1.1 INTRODUCTION

The ST92F120 microcontroller is developed and manufactured by STMicroelectronics using a proprietary n-well HCMOS process. Its performance derives from the use of a flexible 256-register programming model for ultra-fast context switching and real-time event response. The intelligent on-chip peripherals offload the ST9 core from I/O and data management processing tasks allowing critical application tasks to get the maximum use of core resources. The new-generation ST9 MCU devices now also support low power consumption and low voltage operation for power-efficient and low-cost embedded systems.

1.1.1 ST9+ Core

The advanced Core consists of the Central Processing Unit (CPU), the Register File, the Interrupt and DMA controller, and the Memory Management Unit. The MMU allows a single linear address space of up to 4 Mbytes.

Four independent buses are controlled by the Core: a 16-bit memory bus, an 8-bit register data bus, an 8-bit register address bus and a 6-bit interrupt/DMA bus which connects the interrupt and DMA controllers in the on-chip peripherals with the core.

This multiple bus architecture makes the ST9 family devices highly efficient for accessing on and off-chip memory and fast exchange of data with the on-chip peripherals.

The general-purpose registers can be used as accumulators, index registers, or address pointers. Adjacent register pairs make up 16-bit registers for addressing or 16-bit processing. Although the ST9 has an 8-bit ALU, the chip handles 16-bit operations, including arithmetic, loads/stores, and memory/register and memory/memory exchanges.

The powerful I/O capabilities demanded by microcontroller applications are fulfilled by the ST92F120 with 77 I/O lines dedicated to digital Input/Output. These lines are grouped into up to ten 8-bit I/O Ports and can be configured on a bit basis under software control to provide timing, status signals, an address/data bus for interfacing to the external memory, timer inputs and outputs, analog inputs, external interrupts and serial or parallel I/O. Two memory spaces are available to support this wide range of configurations: a combined Program/Data Memory Space and the internal Register File, which includes the control and status registers of the on-chip peripherals.

1.1.2 External Memory Interface

100-pin devices have a 16-bit external address bus allowing them to address up to 64K bytes of external memory.

1.1.3 Flash and E3PROM Memories

In the ST92F120, the embedded Flash memory cell is used to implement emulated EEPROM capability for applications that require regular updates of single-byte parameters.

1.1.4 On-chip Peripherals

Two 16-bit MultiFunction Timers, each with an 8 bit Prescaler and 12 operating modes allow simple use for complex waveform generation and measurement, PWM functions and many other system timing functions by the usage of the two associated DMA channels for each timer.

Two Extended Function Timers provide further timing and signal generation capabilities.

A Standard Timer can be used to generate a stable time base independent from the PLL.

An I^2C interface provides fast I^2C and Access Bus support.

The SPI is a synchronous serial interface for Master and Slave device communication. It supports single master and multimaster systems.

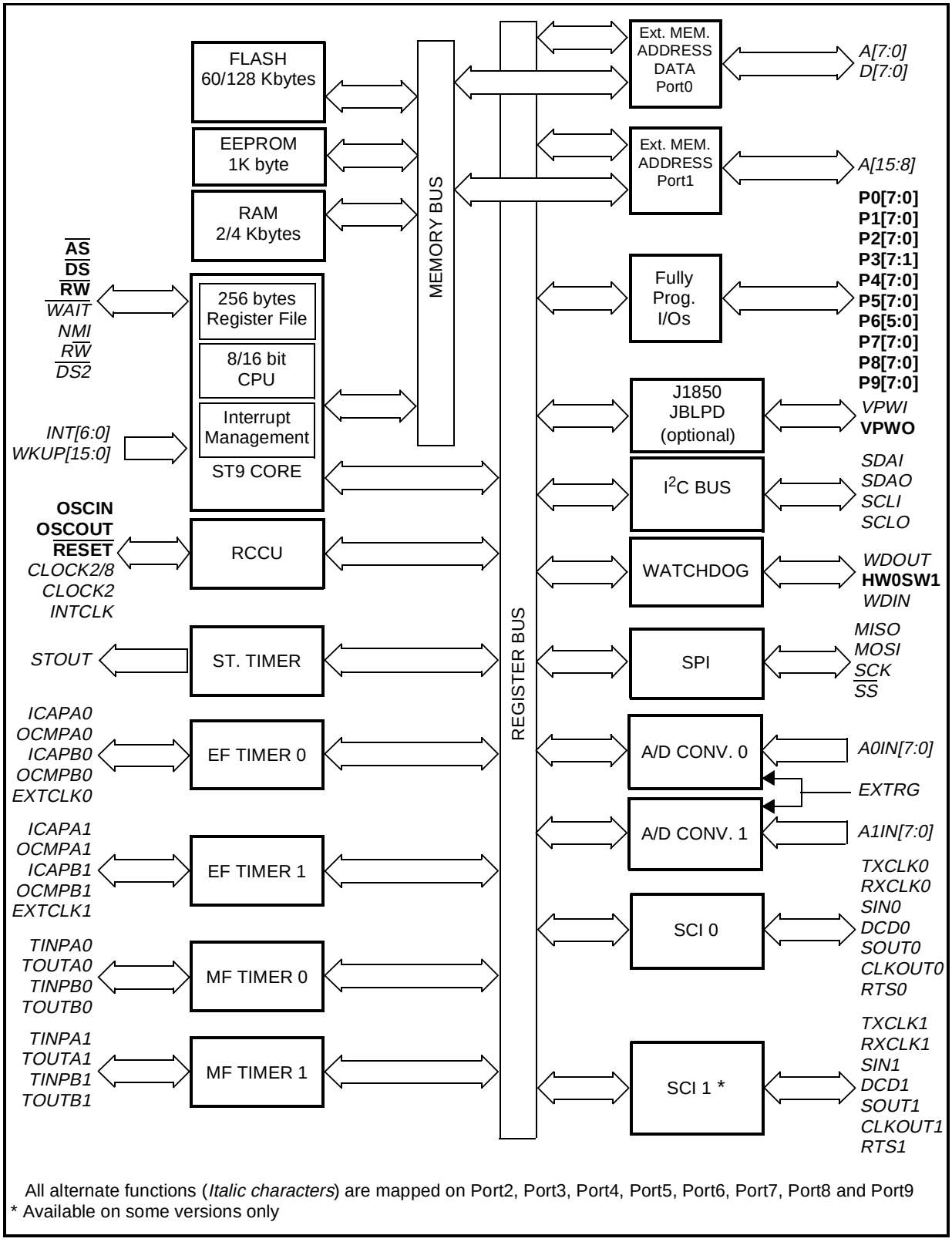
A J1850 Byte Level Protocol Decoder is available (on some devices only) for communicating with a J1850 network.

In addition, there is an 16 channel Analog to Digital Converters with integral sample and hold, fast conversion time and 8-bit resolution.

Completing the device are two or one full duplex Serial Communications Interfaces with an integral generator, asynchronous and synchronous capability (fully programmable format) and associated address/wake-up option, plus two DMA channels.

Finally, a programmable PLL Clock Generator allows the usage of standard 3 to 5 MHz crystals to obtain a large range of internal frequencies up to 24MHz. Low power Run (SLOW), Wait For Interrupt, low power Wait For Interrupt, STOP and HALT modes are also available.

Figure 1. ST92F120JV: Architectural Block Diagram



All alternate functions (*Italic characters*) are mapped on Port2, Port3, Port4, Port5, Port6, Port7, Port8 and Port9
* Available on some versions only

1.2 PIN DESCRIPTION

$\overline{AS}$. Address Strobe (output, active low, 3-state). Address Strobe is pulsed low once at the beginning of each memory cycle. The rising edge of $\overline{AS}$ indicates that address, Read/Write ($\overline{RW}$), and Data signals are valid for memory transfers.

$\overline{DS}$. Data Strobe (output, active low, 3-state). Data Strobe provides the timing for data movement to or from Port 0 for each memory transfer. During a write cycle, data out is valid at the leading edge of $\overline{DS}$. During a read cycle, Data In must be valid prior to the trailing edge of $\overline{DS}$. When the ST9 accesses on-chip memory, $\overline{DS}$ is held high during the whole memory cycle.

$\overline{RESET}$. Reset (input, active low). The ST9 is initialised by the Reset signal. With the deactivation of $\overline{RESET}$, program execution begins from the Program memory location pointed to by the vector contained in program memory locations 00h and 01h.

R$\overline{W}$. Read/Write (output, 3-state). Read/Write determines the direction of data transfer for external memory transactions. $\overline{RW}$ is low when writing to external memory, and high for all other transactions.

OSCIN, OSCOUT. Oscillator (input and output). These pins connect a parallel-resonant crystal, or an external source to the on-chip clock oscillator and buffer. OSCIN is the input of the oscillator inverter and internal clock generator; OSCOUT is the output of the oscillator inverter.

HW0SW1. When connected to V_{DD} through a 1K pull-up resistor, the software watchdog option is selected. When connected to V_{SS} through a 1K pull-down resistor, the hardware watchdog option is selected.

VPWO. This pin is the output line of the J1850 peripheral (JBLPD). It is available only on some devices. On devices without JBLPD peripheral, this pin must not be connected.

P0[7:0], P1[2:0] or P1[7:0]*(Input/Output, TTL or CMOS compatible)*. 16 lines providing the external memory interface for addressing 2K or 64 K bytes of external memory.

P0[7:0], P1[2:0], P2[7:0], P3[7:4], P4[7:4], P5[7:0], P6[5:2,0], P7[7:0] *I/O Port Lines (Input/Output, TTL or CMOS compatible)*. I/O lines grouped into I/O ports of 8 bits, bit programmable under software control as general purpose I/O or as alternate functions.

P1[7:3], P3[3:1], P4[3:0], P6.1, P8[7:0], P9[7:0] *Additional I/O Port Lines available on PQFP100 versions only.*

AV_{DD} Analog V_{DD} of the Analog to Digital Converter (common for A/D 0 and A/D 1).

AV_{SS} Analog V_{SS} of the Analog to Digital Converter (common for A/D 0 and A/D 1).

V_{DD} Main Power Supply Voltage. Four pins are available. The pins are internally connected.

V_{SS} Digital Circuit Ground. Four pins are available. The pins are internally connected.

V_{TEST} Power Supply Voltage for Flash test purposes. This pin is bonded and must be kept to 0 in User mode.

V_{REG} 3V regulator output (on future versions, i.e. ST92F124 and ST92F150).

1.2.1 Electromagnetic Compatibility (EMC)

To reduce the electromagnetic interference the following features have been implemented:

– A low power oscillator is included with a controlled gain to reduce EMI and the power consumption in Halt mode.

– Four pairs of digital power supply pins (V_{DD}, V_{SS}) are located on each side of the package.

– Digital and analog power supplies are completely separated.

– Digital power supplies for internal logic and I/O ports are separated internally.

– Internal decoupling capacitance is located between V_{DD} and V_{SS}.

Note: Each pair of digital V_{DD}/V_{SS} pins should be externally connected by a 10 µF chemical pulling capacitor and a 100 nF ceramic chip capacitor.

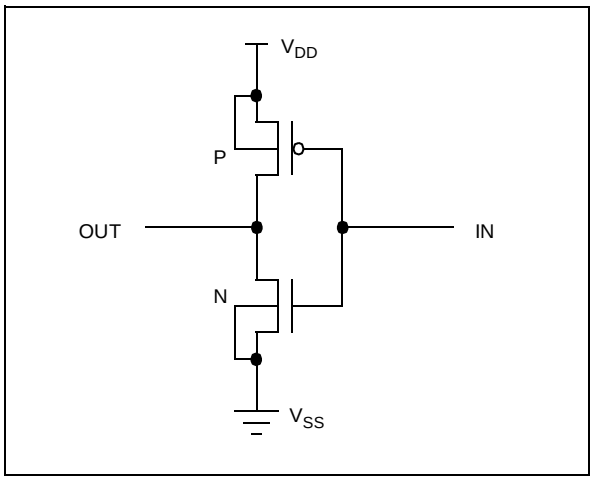
1.2.2 I/O Port Alternate Functions

Each pin of the I/O ports of the ST92F120 may assume software programmable Alternate Functions as shown in Section 1.3.

1.2.3 Termination of Unused Pins

The ST9 device is implemented using CMOS technology; therefore unused pins must be properly terminated in order to avoid application reliability problems. In fact, as shown in Figure 2, the standard input circuitry is based on the CMOS inverter structure.

Figure 2. CMOS Basic Inverter



When an input is kept at logic zero, the N-channel transistor is off, while the P-channel is on and can conduct. The opposite occurs when an input is kept at logic one. CMOS transistors are essentially linear devices with relatively broad switching points. During commutation, the input passes through midsupply, and there is a region of input voltage values where both P and N-channel transistors are on. Since normally the transitions are fast, there is a very short time in which a current can flow: once the switching is completed there is no longer current. This phenomenon explains why the overall current depends on the switching rate: the consumption is directly proportional to the number of transistors inside the device which are in the linear region during transitions, charging and discharging internal capacitances.

In order to avoid extra power supply current, it is important to bias input pins properly when not used. In fact, if the input impedance is very high, pins can float, when not connected, either to a midsupply level or can oscillate (injecting noise in the device).

Depending on the specific configuration of each I/O pin on different ST9 devices, it can be more or less critical to leave unused pins floating. For this reason, on most pins, the configuration after RESET enables an internal weak pull-up transistor in order to avoid floating conditions. For other pins this is intrinsically forbidden, like for the true open-drain pins. In any case, the application software must program the right state for unused pins to avoid conflicts with external circuitry (whichever it is: pull-up, pull-down, floating, etc.).

The suggested method of terminating unused I/O is to connect an external individual pull-up or pull-down for each pin, even though initialization software can force outputs to a specified and defined value, during a particular phase of the RESET routine there could be an undetermined status at the input section.

Usage of pull-ups and/or pull-downs is preferable in place of direct connection to V_{DD} or V_{SS}. If pull-up or pull-down resistors are used, inputs can be forced for test purposes to a different value, and outputs can be programmed to both digital levels without generating high current drain due to the conflict.

Anyway, during system verification flow, attention must be paid to reviewing the connection of each pin, in order to avoid potential problems.

1.2.4 Avoidance of Pin Damage

Although integrated circuit data sheets provide the user with conservative limits and conditions in order to prevent damage, sometimes it is useful for the hardware system designer to know the internal failure mechanisms: the risk of exposure to illegal voltages and conditions can be reduced by smart protection design.

It is not possible to classify and to predict all the possible damage resulting from violating maximum ratings and conditions, due to the large number of variables that come into play in defining the failures: in fact, when an overvoltage condition is applied, the effects on the device can vary significantly depending on lot-to-lot process variations, operating temperature, external interfacing of the ST9 with other devices, etc.

In the following sections, background technical information is given in order to help system designers to reduce risk of damage to the ST9 device.

1.2.4.1 Electrostatic Discharge and Latchup

CMOS integrated circuits are generally sensitive to exposure to high voltage static electricity, which can induce permanent damage to the device: a typical failure is the breakdown of thin oxides, which causes high leakage current and sometimes shorts.

Latchup is another typical phenomenon occurring in integrated circuits: unwanted turning on of parasitic bipolar structures, or silicon-controlled rectifiers (SCR), may overheat and rapidly destroy the device. These unintentional structures are composed of P and N regions which work as emitters, bases and collectors of parasitic bipolar transistors: the bulk resistance of the silicon in the wells and substrate act as resistors on the SCR structure. Applying voltages below V_{SS} or above V_{DD}, and when the level of current is able to generate a

voltage drop across the SCR parasitic resistor, the SCR may be turned on; to turn off the SCR it is necessary to remove the power supply from the device.

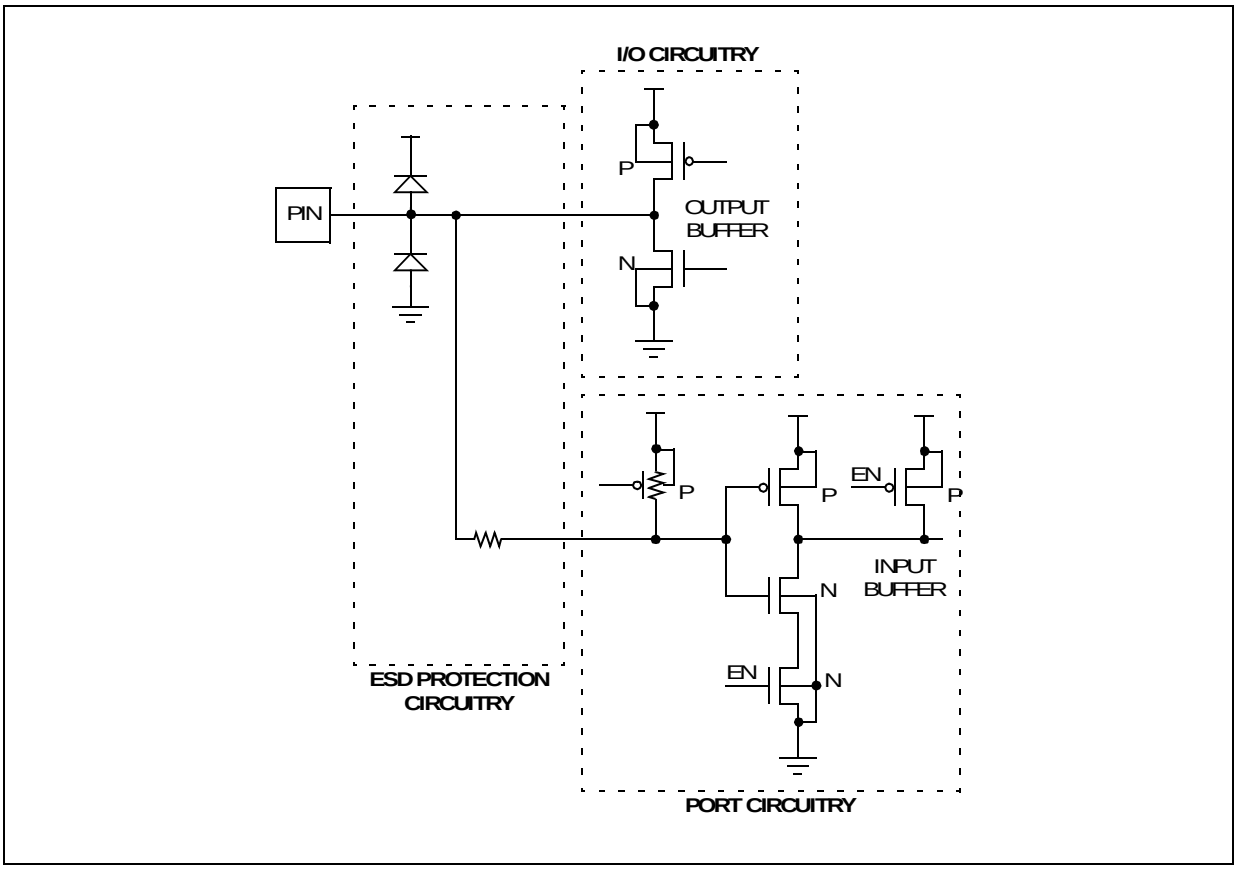
The present ST9 design implements layout and process solutions to decrease the effects of electrostatic discharges (ESD) and latchup. Of course it is not possible to test all devices, due to the destructive nature of the mechanism; in order to guarantee product reliability, destructive tests are carried out on groups of devices, according to STMicroelectronics internal Quality Assurance standards and recommendations.

1.2.4.2 Protective Interface

Although ST9 input/output circuitry has been designed taking ESD and Latchup problems into account, for those applications and systems where ST9 pins are exposed to illegal voltages and high current injections, the user is strongly recommended to implement hardware solutions which reduce the risk of damage to the microcontroller: low-pass filters and clamp diodes are usually sufficient in preventing stress conditions.

The risk of having out-of-range voltages and currents is greater for those signals coming from outside the system, where noise effect or uncontrolled spikes could occur with higher probability than for the internal signals; it must be underlined that in some cases, adoption of filters or other dedicated interface circuitries might affect global microcontroller performance, inducing undesired timing delays, and impacting the global system speed.

Figure 3. Digital Input/Output - Push-Pull



1.2.4.3 Internal Circuitry: Digital I/O pin

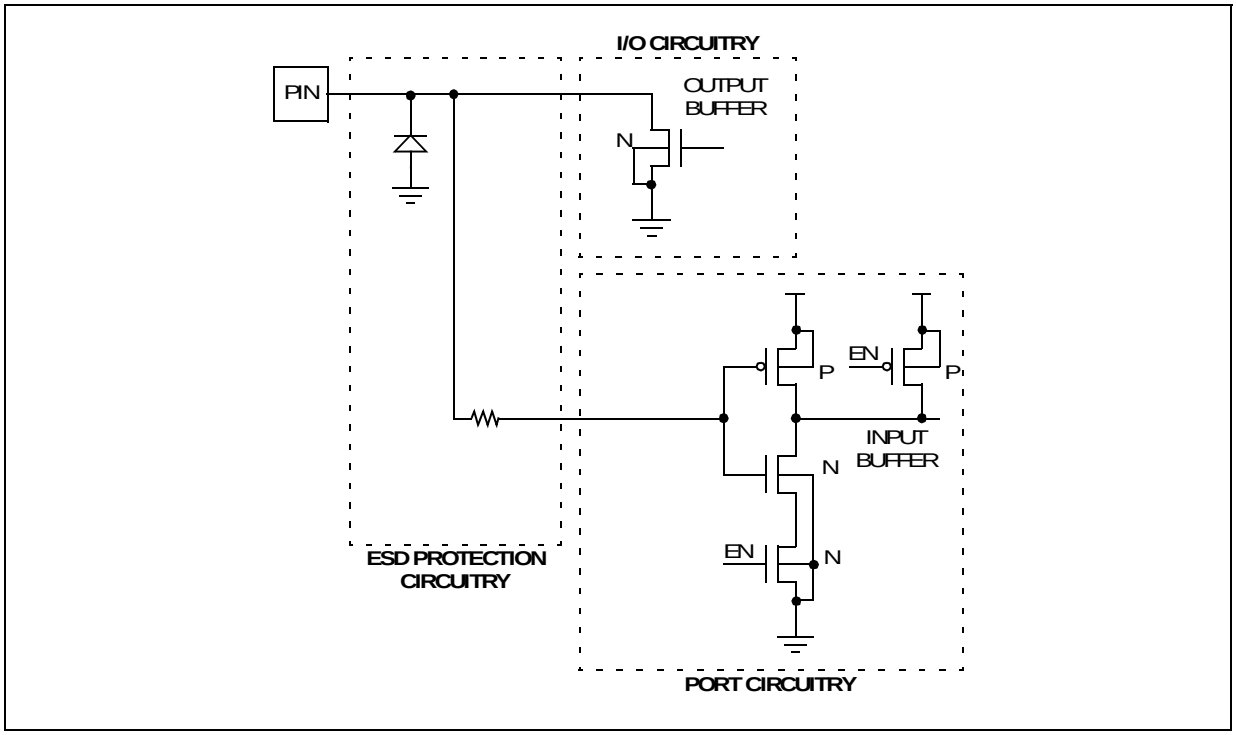
In Figure 3 a schematic representation of an ST9 pin able to operate either as an input or as an output is shown. The circuitry implements a standard input buffer and a push-pull configuration for the output buffer. It is evident that although it is possible to disable the output buffer when the input section is used, the MOS transistors of the buffer itself can still affect the behaviour of the pin when exposed to illegal conditions. In fact, the P-channel transistor of the output buffer implements a direct diode to V_{DD} (P-diffusion of the drain connected to the pin and N-well connected to V_{DD}), while the N-channel of the output buffer implements a diode to V_{SS} (P-substrate connected to VSS and N-diffusion of the drain connected to the pin). In parallel to these diodes, dedicated circuitry is implemented to protect the logic from ESD events (MOS, diodes and input series resistor).

The most important characteristic of these extra devices is that they must not disturb normal operating modes, while acting during exposure to over limit conditions, avoiding permanent damage to the logic circuitry.

All I/O pins can generally be programmed to work also as open-drain outputs, by simply writing in the corresponding register of the I/O Port. The gate of the P-channel of the output buffer is disabled: it is important to highlight that physically the P-channel transistor is still present, so the diode to V_{DD} works. In some applications it can occur that the voltage applied to the pin is higher than the V_{DD} value (supposing the external line is kept high, while the ST9 power supply is turned off): this condition will inject current through the diode, risking permanent damages to the device.

In any case, programming I/O pins as open-drain can help when several pins in the system are tied to the same point: of course software must pay attention to program only one of them as output at any time, to avoid output driver contentions; it is advisable to configure these pins as output open-drain in order to reduce the risk of current contentions.

Figure 4. Digital Input/Output - True Open Drain Output



In Figure 5 a true open-drain pin schematic is shown. In this case all paths to V_{DD} are removed (P-channel driver, ESD protection diode, internal weak pull-up) in order to allow the system to turn off the power supply of the microcontroller and keep the voltage level at the pin high without injecting current in the device. This is a typical condition which can occur when several devices interface a serial bus: if one device is not involved in the communication, it can be disabled by turning off its power supply to reduce the system current consumption.

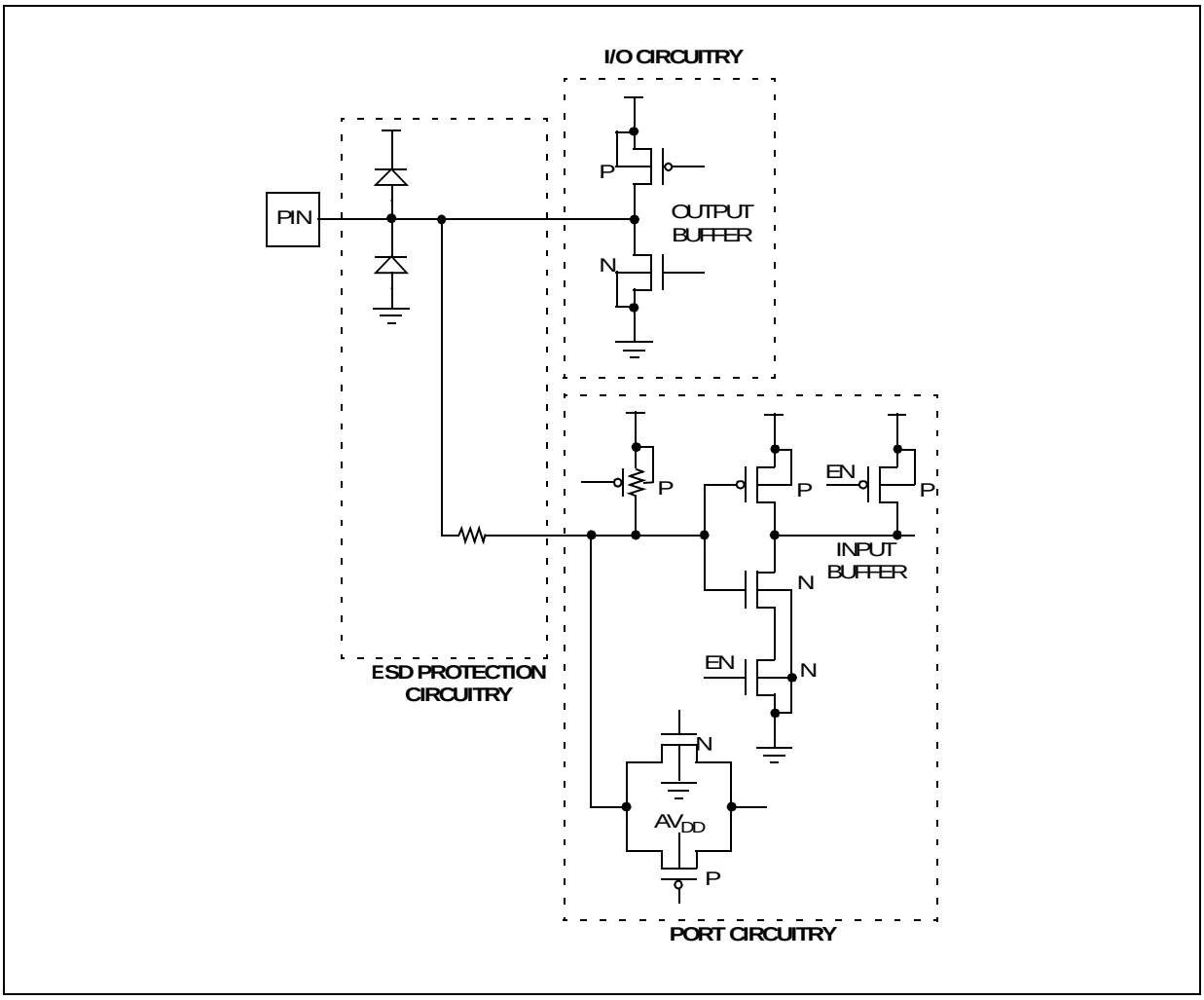
When an illegal negative voltage level is applied to the ST9 I/O pins (both versions, push-pull and true open-drain output) the clamp diode is always present and active (see ESD protection circuitry and N-channel driver).

1.2.4.4 Internal Circuitry: Analog Input pin

Figure 5 shows the internal circuitry used for analog input. It is substantially a digital I/O with an added analog multiplexer for the A/D Converter input signal selection.

The presence of the multiplexer P-channel and N-channel can affect the behaviour of the pin when exposed to illegal voltage conditions. These transistors are controlled by a low noise logic, biased through AV_{DD} and AV_{SS} including P-channel N-well: it is important to always verify the input voltage value with respect to both analog power supply and digital power supply, in order to avoid unintended current injections which (if not limited) could destroy the device.

Figure 5. Digital Input/Output - Push-Pull Output - Analog Multiplexer Input



1.2.4.5 Power Supply and Ground

As already said for the I/O pins, in order to guarantee ST9 compliancy with respect to Quality Assurance recommendations concerning ESD and Latchup, dedicated circuits are added to the different power supply and ground pins (digital and analog). These structures create preferred paths for the high current injected during discharges, avoiding damage to active logic and circuitry. It is important for the system designer to take this added circuitry into account, which is not always transparent with respect to the relative level of voltages applied to the different power supply and ground pins. Figure 6 shows schematically the protection net implemented on ST9 devices, composed of diodes and other special structures.

The clamp structure between the V_{DD} and V_{SS} pins is designed to be active during very fast transitions (typical of electrostatic discharges). Other paths are implemented through diodes: they limit the possibility of positively differentiating AV_{DD} and V_{DD} (i.e. $AV_{DD} > V_{DD}$); similar considerations are valid for AV_{SS} and V_{SS} due to the back-to-back diode structure implemented between the two pins. Anyway, it must be highlighted that, because V_{SS} and AV_{SS} are connected to the substrate of the silicon die (even though in different areas of the die itself), they represent the reference point from which all other voltages are measured, and it is recommended to never differentiate AV_{SS} from V_{SS}.

Note: If more than one pair of pins for V_{SS} and V_{DD} is available on the device, they are connected internally and the protection net diagram remains the same as shown in Figure 6.

Figure 6. Power Supply and Ground Configuration

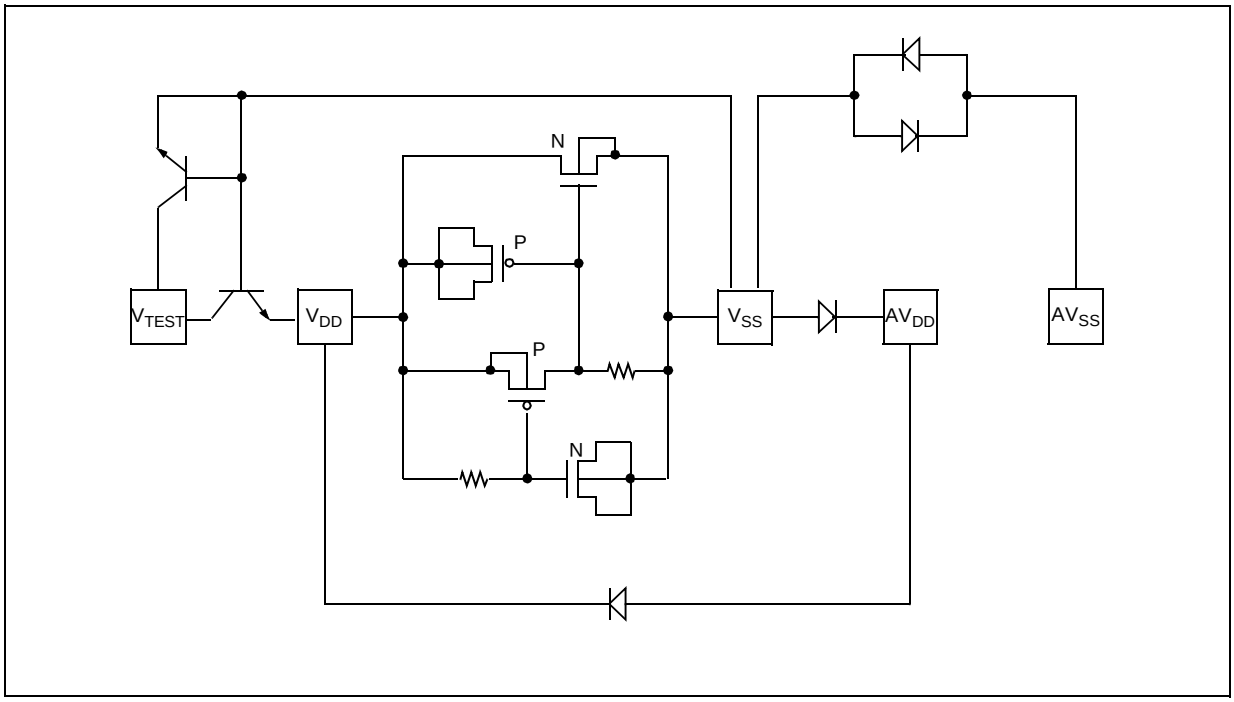
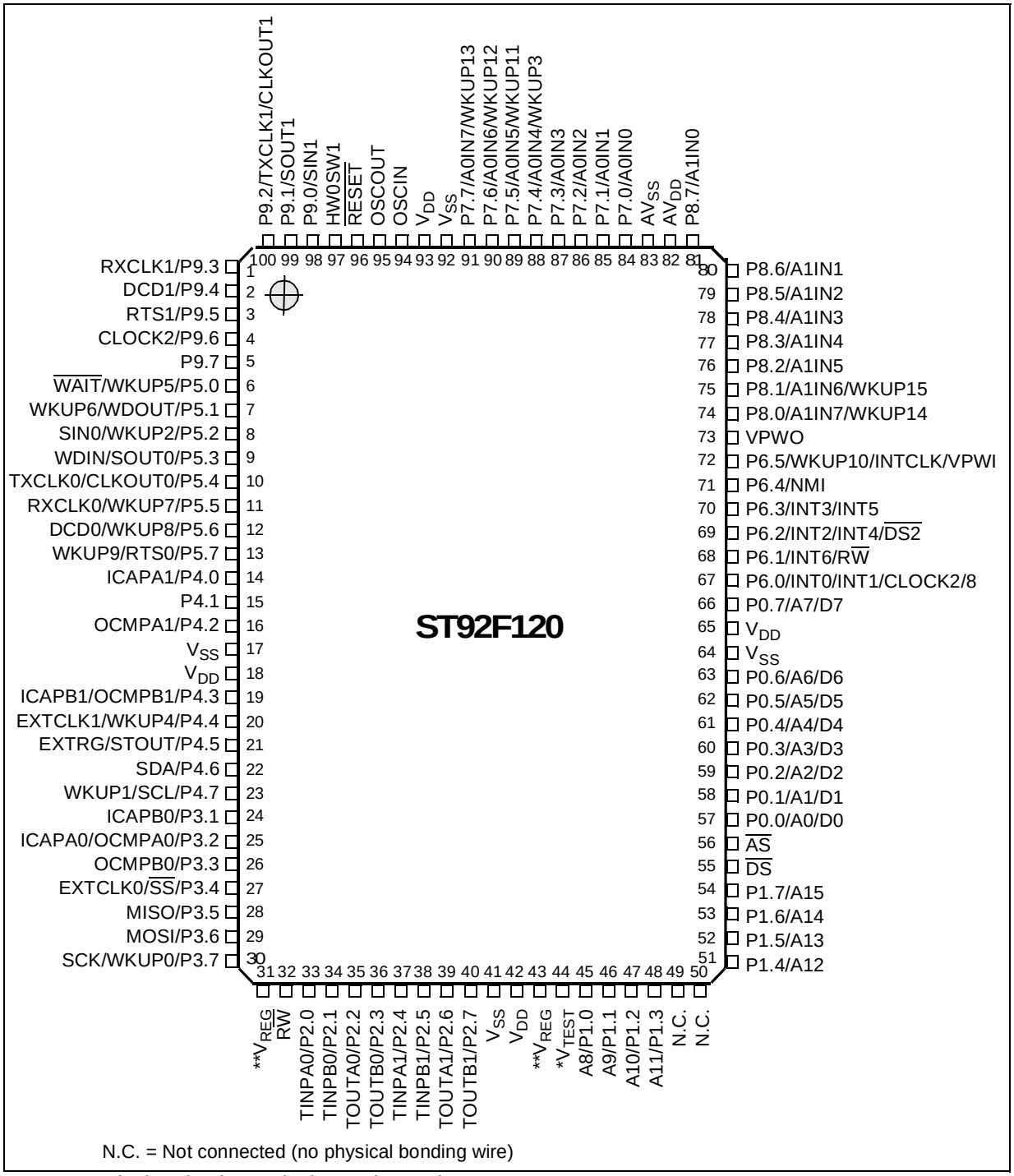


Figure 7. ST92F120: Pin Configuration (Top-view PQFP100)



* V_{TEST} must be kept low in standard operating mode.

**On future versions.

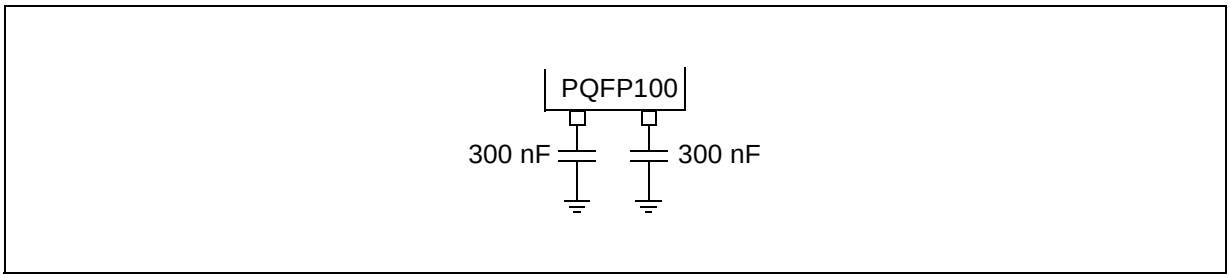
Table 1. ST92F120 Power Supply Pins

Name	Function	PQFP100
V_{DD}	Main Power Supply Voltage (pins internally connected)	18
		42
		65
		93
V_{SS}	Digital Circuit Ground (pins internally connected)	17
		41
		64
		92
AV_{DD}	Analog Circuit Supply Voltage	82
AV_{SS}	Analog Circuit Ground	83
V_{TEST}	Must be kept low in standard operating mode	44
V_{REG}	3V Regulator output (on future versions, i.e. ST92F124 and ST92F150)	31 43

Table 2. ST92F120 Primary Function Pins

Name	Function	PQFP100
$\overline{AS}$	Address Strobe	56
$\overline{DS}$	Data Strobe	55
$R\overline{W}$	Read/Write	32
OSCIN	Oscillator Input	94
OSCOUT	Oscillator Output	95
RESET	Reset to initialize the Microcontroller	96
HW0SW1	Watchdog HW/SW enabling selection	97
VPWO	J1850 JBLPD Output. On devices without JBPLD peripheral, this pin must not be connected.	73

Figure 8. Recommended Connections for V_{REG}



Note : For future compatibility with shrinked versions, the V_{REG} pins should be connected to a minimum of 600 nF (total). Special care should be taken to minimize the distance between the ST9 microcontroller and the capacitors.

1.3 I/O PORTS

Port 0 and Port 1 provide the external memory interface. All the ports of the device can be programmed as Input/Output or in Input mode, compatible with TTL or CMOS levels (except where Schmitt Trigger is present). Each bit can be programmed individually (Refer to the I/O ports chapter).

Internal Weak Pull-up

As shown in Table 3, not all input sections implement a Weak Pull-up. This means that the pull-up must be connected externally when the pin is not used or programmed as bidirectional.

TTL/CMOS Input

For all those port bits where no input schmitt trigger is implemented, it is always possible to program the input level as TTL or CMOS compatible by programming the relevant PxC2.n control bit. Refer I/O Ports Chapter to the section titled “Input/ Output Bit Configuration”.

Schmitt Trigger Input

Two different kind of Schmitt Trigger circuitries are implemented: Standard and High Hysteresis. Standard Schmitt Trigger is widely used (see Table 3), while the High Hysteresis one is present on the NMI and VPWI input function pins mapped on Port 6 [5:4] (see Table 4).

All inputs which can be used for detecting interrupt events have been configured with a “standard” Schmitt Trigger, apart from, as already said, the NMI pin which implements the “High Hysteresis” version. In this way, all interrupt lines are guaranteed as “level sensitive”.

Push-Pull/OD Output

The output buffer can be programmed as push-pull or open-drain: attention must be paid to the fact that the open-drain option corresponds only to a disabling of P-channel MOS transistor of the buffer itself: it is still present and physically connected to the pin. Consequently it is not possible to increase the output voltage on the pin over V_{DD}+0.3 Volt, to avoid direct junction biasing.

Pure Open-Drain Output

The user can increase the voltage on an I/O pin over V_{DD}+0.3 Volt where the P-channel MOS transistor is physically absent: this is allowed on all “Pure Open Drain” pins. Of course, in this case the push-pull option is not available and any weak pull-up must implemented externally.

Table 3. I/O Port Characteristics

	Input	Output	Weak Pull-Up	Reset State
Port 0[7:0]	TTL/CMOS	Push-Pull/OD	No	Bidirectional
Port 1[7:0]	TTL/CMOS	Push-Pull/OD	No	Bidirectional
Port 2[1:0]	Schmitt trigger	Push-Pull/OD	Yes	Input
Port 2[3:2]	TTL/CMOS	Pure OD	No	Input CMOS
Port 2[5:4]	Schmitt trigger	Push-Pull/OD	Yes	Input
Port 2[7:6]	TTL/CMOS	Push-Pull/OD	Yes	Input CMOS
Port 3[2:1]	Schmitt trigger	Push-Pull/OD	Yes	Input
Port 3.3	TTL/CMOS	Push-Pull/OD	Yes	Input CMOS
Port 3[7:4]	Schmitt trigger	Push-Pull/OD	Yes	Input
Port 4.0, Port 4.4	Schmitt trigger	Push-Pull/OD	No	Input
Port 4.1	Schmitt trigger	Push-Pull/OD	Yes	Bidirectional WPU
Port 4.2, Port 4.5	TTL/CMOS	Push-Pull/OD	Yes	Input CMOS
Port 4.3	Schmitt trigger	Push-Pull/OD	Yes	Input
Port 4[7:6]	Schmitt trigger inside I/O cell	Pure OD	No	Input
Port 5[2:0], Port 5[7:4]	Schmitt trigger	Push-Pull/OD	No	Input
Port 5.3	TTL/CMOS	Push-Pull/OD	Yes	Input CMOS
Port 6[3:0]	Schmitt trigger	Push-Pull/OD	Yes	Input
Port 6[5:4]	High hysteresis Schmitt trigger inside I/O cell	Push-Pull/OD	Yes (inside I/O cell)	Input
Port 7[7:0]	Schmitt trigger	Push-Pull/OD	Yes	Input
Port 8[1:0]	Schmitt trigger	Push-Pull/OD	Yes	Input
Port 8[7:2]	Schmitt trigger	Push-Pull/OD	Yes	Bidirectional WPU
Port 9[7:0]	Schmitt trigger	Push-Pull/OD	Yes	Bidirectional WPU

Legend: WPU = Weak Pull-Up, OD = Open Drain

How to Configure the I/O Ports

To configure the I/O ports, use the information in Table 3, Table 4 and the Port Bit Configuration Table in the I/O Ports Chapter (See page 120).

Input Note = the hardware characteristics fixed for each port line in Table 3.

– If Input note = TTL/CMOS, either TTL or CMOS input level can be selected by software.

– If Input note = Schmitt trigger, selecting CMOS or TTL input by software has no effect, the input will always be Schmitt Trigger.

Alternate Functions (AF) = More than one AF cannot be assigned to an I/O pin at the same time:

An alternate function can be selected as follows.

AF Inputs:

– AF is selected implicitly by enabling the corresponding peripheral. Exception to this are A/D inputs which must be explicitly selected as AF by software.

AF Outputs or Bidirectional Lines:

– In the case of Outputs or I/Os, AF is selected explicitly by software.

Example 1: SCI input

AF: SIN0, Port: P5.2, Input note: Schmitt Trigger.

Write the port configuration bits:

P5C2.2=1
P5C1.2=0
P5C0.2 =1

Enable the SCI peripheral by software as described in the SCI chapter.

Example 2: SCI output

AF: SOUT0, Port: P5.3, Output note:
Push-Pull/OD.

Write the port configuration bits (for AF OUT PP):

P5C2.3=0
P5C1.3=1
P5C0.3 =1

Example 3: External Memory I/O

AF: A0/D0, Port : P0.0, Input Note: TTL/CMOS

Write the port configuration bits:

P0C2.0=1
P0C1.0=1
P0C0.0 =1

Example 4: Analog input

AF: A0IN0, Port : 7.0, Input Note: does not apply to analog input

Write the port configuration bits:

P7C2.0=1
P7C1.0=1
P7C0.0 =1

Table 4. I/O Port Alternate Functions

Port Name	General Purpose I/O	Pin No.	Alternate Functions		
P0.0	All ports useable for general purpose I/O (input, output or bidirectional)	57	A0/D0	I/O	Address/Data bit 0
P0.1		58	A1/D1	I/O	Address/Data bit 1
P0.2		59	A2/D2	I/O	Address/Data bit 2
P0.3		60	A3/D3	I/O	Address/Data bit 3
P0.4		61	A4/D4	I/O	Address/Data bit 4
P0.5		62	A5/D5	I/O	Address/Data bit 5
P0.6		63	A6/D6	I/O	Address/Data bit 6
P0.7		66	A7/D7	I/O	Address/Data bit 7
P1.0		45	A8	I/O	Address bit 8
P1.1		46	A9	I/O	Address bit 9
P1.2		47	A10	I/O	Address bit 10
P1.3		48	A11	I/O	Address bit 11
P1.4		51	A12	I/O	Address bit 12
P1.5		52	A13	I/O	Address bit 13
P1.6		53	A14	I/O	Address bit 14
P1.7		54	A15	I/O	Address bit 15
P2.0		33	TINPA0	I	Multifunction Timer 0 - Input A
P2.1		34	TINPB0	I	Multifunction Timer 0 - Input B
P2.2		35	TOUTA0	O	Multifunction Timer 0 - Output A
P2.3		36	TOUTB0	O	Multifunction Timer 0 - Output B
P2.4		37	TINPA1	I	Multifunction Timer 1 - Input A
P2.5		38	TINPB1	I	Multifunction Timer 1 - Input B
P2.6		39	TOUTA1	O	Multifunction Timer 1 - Output A
P2.7		40	TOUTB1	O	Multifunction Timer 1 - Output B
P3.1		24	ICAPB0	I	Ext. Timer 0 - Input Capture B
P3.2		25	ICAPA0	I	Ext. Timer 0 - Input Capture A
			OCMPA0	O	Ext. Timer 0 - Output Compare A
P3.3		26	OCMPB0	O	Ext. Timer 0 - Output Compare B
P3.4		27	EXTCLK0	I	Ext. Timer 0 - Input Clock
			SS	I	SPI - Slave Select
P3.5		28	MISO	I/O	SPI - Master Input/Slave Output Data
P3.6		29	MOSI	I/O	SPI - Master Output/Slave Input Data
P3.7		30	SCK	I	SPI - Serial Input Clock
			WKUP0	I	Wake-up Line 0
			SCK	O	SPI - Serial Output Clock
P4.0		14	ICAPA1	I	Ext. Timer 1 - Input Capture A

Port Name	General Purpose I/O	Pin No.	Alternate Functions		
P4.1	All ports useable for general purpose I/O (input, output or bidirectional)	15		I/O	
P4.2		16	OCMPA1	O	Ext. Timer 1 - Output Compare A
P4.3		19	ICAPB1	I	Ext. Timer 1 - Input Capture B
			OCMPB1	O	Ext. Timer 1 - Output Compare B
P4.4		20	EXTCLK1	I	Ext. Timer 1 - Input Clock
			WKUP4	I	Wake-up Line 4
P4.5		21	EXTRG	I	A/D 0 and A/D 1 - Ext. Trigger
			STOUT	O	Standard Timer Output
P4.6		22	SDA	I/O	I^2C Data
P4.7		23	WKUP1	I	Wake-up Line 1
			SCL	I/O	I^2C Clock
P5.0		6	WAIT	I	External Wait Request
			WKUP5	I	Wake-up Line 5
P5.1		7	WKUP6	I	Wake-up Line 6
			WDOUT	O	Watchdog Timer Output
P5.2		8	SIN0	I	SCI0 - Serial Data Input
			WKUP2	I	Wake-up Line 2
P5.3		9	SOUT0	O	SCI0 - Serial Data Output
			WDIN	I	Watchdog Timer Input
P5.4		10	TXCLK0	I	SCI0 - Transmit Clock Input
			CLKOUT0	O	SCI0 - Clock Output
P5.5		11	RXCLK0	I	SCI0 - Receive Clock Input
			WKUP7	I	Wake-up Line 7
P5.6		12	DCD0	I	SCI0 - Data Carrier Detect
			WKUP8	I	Wake-up Line 8
P5.7		13	WKUP9	I	Wake-up Line 9
			RTS0	O	SCI0 - Request To Send
P6.0		67	INT0	I	External Interrupt 0
			INT1	I	External Interrupt 1
			CLOCK2/8	O	CLOCK2 divided by 8
P6.1		68	INT6	I	External Interrupt 6
			$R\overline{W}$	O	Read/Write
P6.2		69	INT2	I	External Interrupt 2
			INT4	I	External Interrupt 4
			DS2	O	Data Strobe 2

Port Name	General Purpose I/O	Pin No.	Alternate Functions		
P6.3	All ports useable for general purpose I/O (input, output or bidirectional)	70	INT3	I	External Interrupt 3
			INT5	I	External Interrupt 5
P6.4		71	NMI	I	Non Maskable Interrupt
P6.5		72	WKUP10	I	Wake-up Line 10
			VPWI	I	JBLPD input
			INTCLK	O	Internal Main Clock
P7.0		84	A0IN0	I	A/D 0 - Analog Data Input 0
P7.1		85	A0IN1	I	A/D 0 - Analog Data Input 1
P7.2		86	A0IN2	I	A/D 0 - Analog Data Input 2
P7.3		87	A0IN3	I	A/D 0 - Analog Data Input 3
P7.4		88	WKUP3	I	Wake-up Line 3
			A0IN4	I	A/D 0 - Analog Data Input 4
P7.5		89	A0IN5	I	A/D 0 - Analog Data Input 5
			WKUP11	I	Wake-up Line 11
P7.6		90	A0IN6	I	A/D 0 - Analog Data Input 6
			WKUP12	I	Wake-up Line 12
P7.7		91	A0IN7	I	A/D 0 - Analog Data Input 7
			WKUP13	I	Wake-up Line 13
P8.0		74	A1IN7	I	A/D 1 - Analog Data Input 7
			WKUP14	I	Wake-up Line 14
P8.1		75	A1IN6	I	A/D 1 - Analog Data Input 6
			WKUP15	I	Wake-up Line 15
P8.2		76	A1IN5	I	A/D 1 - Analog Data Input 5
P8.3		77	A1IN4	I	A/D 1 - Analog Data Input 4
P8.4		78	A1IN3	I	A/D 1 - Analog Data Input 3
P8.5		79	A1IN2	I	A/D 1 - Analog Data Input 2
P8.6		80	A1IN1	I	A/D 1 - Analog Data Input 1
P8.7		81	A1IN0	I	A/D 1 - Analog Data Input 0
P9.0		98	SIN1	I	SCI1 - Serial Data Input
P9.1		99	SOUT1	O	SCI1 - Serial Data Output
P9.2		100	TXCLK1	I	SCI1 - Transmit Clock input
			CLKOUT1	O	SCI1 - Clock Input
P9.3		1	RXCLK1	I	SCI1 - Receive Clock Input
P9.4		2	DCD1	I	SCI1 - Data Carrier Detect
P9.5		3	RTS1	O	SCI1 - Request To Send
P9.6		4	CLOCK2	O	CLOCK2 internal signal
P9.7		5		I/O	

1.4 OPERATING MODES

To optimize the performance versus the power consumption of the device, the ST92F120 supports different operating modes that can be dynamically selected depending on the performance and functionality requirements of the application at a given moment.

RUN MODE: This is the full speed execution mode with CPU and peripherals running at the maximum clock speed delivered by the Phase Locked Loop (PLL) of the Clock Control Unit (CCU).

SLOW MODE: Power consumption can be significantly reduced by running the CPU and the peripherals at reduced clock speed using the CPU Prescaler and CCU Clock Divider.

WAIT FOR INTERRUPT MODE: The Wait For Interrupt (WFI) instruction suspends program execution until an interrupt request is acknowledged. During WFI, the CPU clock is halted while the peripheral and interrupt controller keep running at a frequency depending on the CCU programming.

LOW POWER WAIT FOR INTERRUPT MODE: Combining SLOW mode and Wait For Interrupt mode it is possible to reduce the power consumption by more than 80%.

STOP MODE: When the STOP is requested by executing the STOP bit writing sequence (see dedicated section on Wake-up Management Unit paragraph), and if NMI is kept low, the CPU and the peripherals stop operating. Operations resume after a wake-up line is activated (16 wake-up lines plus NMI pin). See the RCCU and Wake-up Management Unit paragraphs in the following for the details. The difference with the HALT mode consists in the way the CPU exits this state: when the STOP is executed, the status of the registers is recorded, and when the system exits from the STOP mode the CPU continues the execution with the same status, without a system reset.

When the MCU enters STOP mode the Watchdog stops counting. After the MCU exits from STOP mode, the Watchdog resumes counting from where it left off.

When the MCU exits from STOP mode, the oscillator, which was sleeping too, requires about 5 ms to restart working properly (at a 4 MHz oscillator frequency). An internal counter is present to guarantee that all operations after exiting STOP Mode, take place with the clock stabilised.

The counter is active only when the oscillation has already taken place. This means that 1-2 ms must be added to take into account the first phase of the oscillator restart.

HALT MODE: When executing the HALT instruction, and if the Watchdog is not enabled, the CPU and its peripherals stop operating and the status of the machine remains frozen (the clock is also stopped). A reset is necessary to exit from Halt mode.

2 DEVICE ARCHITECTURE

2.1 CORE ARCHITECTURE

The ST9 Core or Central Processing Unit (CPU) features a highly optimised instruction set, capable of handling bit, byte (8-bit) and word (16-bit) data, as well as BCD and Boolean formats; 14 addressing modes are available.

Four independent buses are controlled by the Core: a 16-bit Memory bus, an 8-bit Register data bus, an 8-bit Register address bus and a 6-bit Interrupt/DMA bus which connects the interrupt and DMA controllers in the on-chip peripherals with the Core.

This multiple bus architecture affords a high degree of pipelining and parallel operation, thus making the ST9 family devices highly efficient, both for numerical calculation, data handling and with regard to communication with on-chip peripheral resources.

2.2 MEMORY SPACES

There are two separate memory spaces:

– The Register File, which comprises 240 8-bit registers, arranged as 15 groups (Group 0 to E), each containing sixteen 8-bit registers plus up to 64 pages of 16 registers mapped in Group F, which hold data and control bits for the on-chip peripherals and I/Os.

– A single linear memory space accommodating both program and data. All of the physically separate memory areas, including the internal ROM, internal RAM and external memory are mapped in this common address space. The total addressable memory space of 4 Mbytes (limited by the size of on-chip memory and the number of external address pins) is arranged as 64 segments of 64 Kbytes. Each segment is further subdivided into four pages of 16 Kbytes, as illustrated in Figure 1. A Memory Management Unit uses a set of pointer registers to address a 22-bit memory field using 16-bit address-based instructions.

2.2.1 Register File

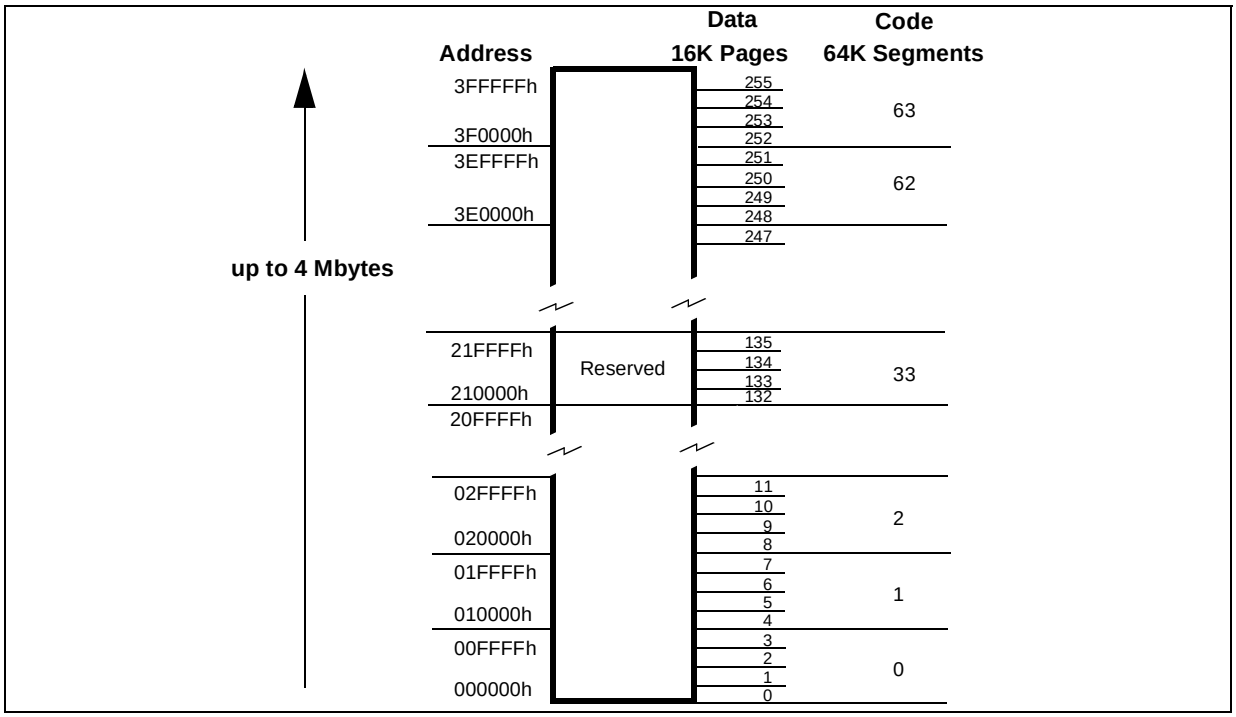
The Register File consists of (see Figure 2):

– 224 general purpose registers (Group 0 to D, registers R0 to R223)

– 6 system registers in the System Group (Group E, registers R224 to R239)

– Up to 64 pages, depending on device configuration, each containing up to 16 registers, mapped to Group F (R240 to R255), see Figure 3.

Figure 9. Single Program and Data Memory Address Space



MEMORY SPACES (Cont'd)

Figure 10. Register Groups

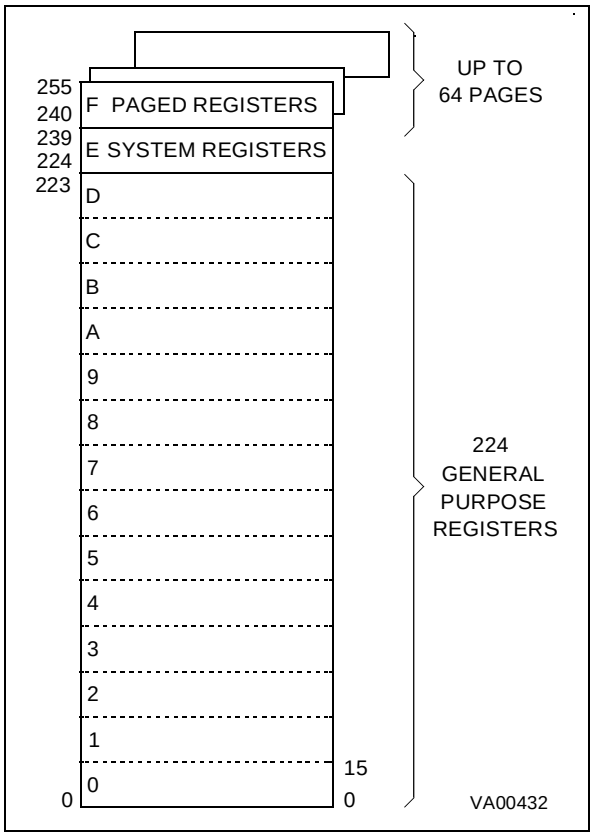


Figure 11. Page Pointer for Group F mapping

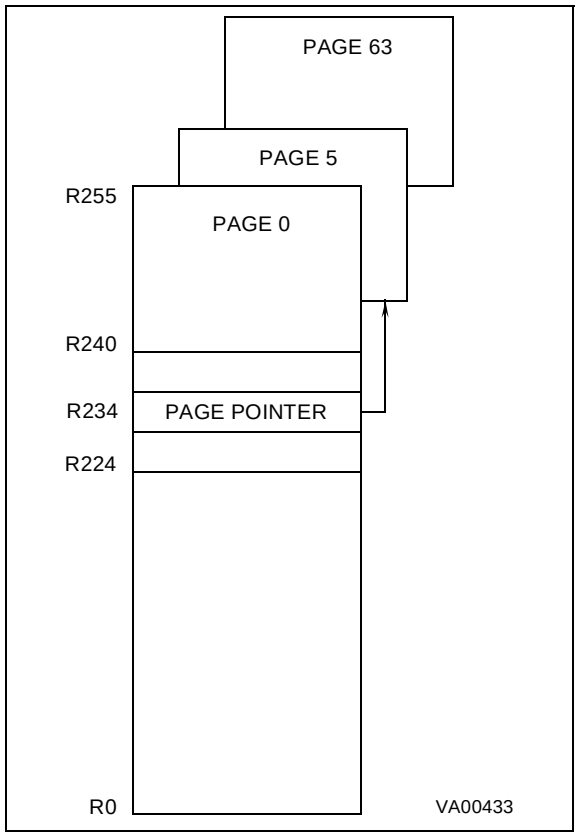
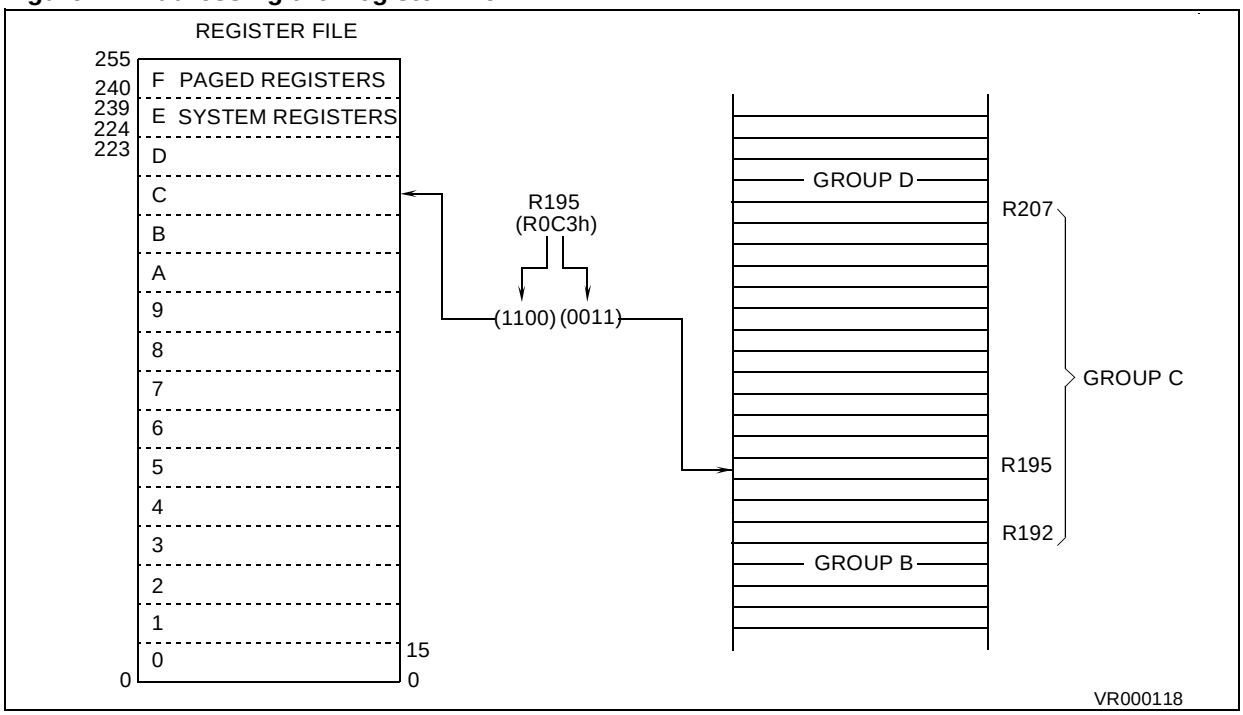


Figure 12. Addressing the Register File



MEMORY SPACES (Cont'd)

2.2.2 Register Addressing

Register File registers, including Group F paged registers (but excluding Group D), may be addressed explicitly by means of a decimal, hexadecimal or binary address; thus `R231, RE7h` and `R11100111b` represent the same register (see Figure 4). Group D registers can only be addressed in Working Register mode.

Note that an upper case "R" is used to denote this direct addressing mode.

Working Registers

Certain types of instruction require that registers be specified in the form "`rx`", where `x` is in the range 0 to 15: these are known as Working Registers.

Note that a lower case "`r`" is used to denote this indirect addressing mode.

Two addressing schemes are available: a single group of 16 working registers, or two separately mapped groups, each consisting of 8 working registers. These groups may be mapped starting at any 8 or 16 byte boundary in the register file by means of dedicated pointer registers. This technique is described in more detail in Section 1.3.3, and illustrated in Figure 5 and in Figure 6.

System Registers

The 16 registers in Group E (R224 to R239) are System registers and may be addressed using any of the register addressing modes. These registers are described in greater detail in Section 1.3.

Paged Registers

Up to 64 pages, each containing 16 registers, may be mapped to Group F. These are addressed using any register addressing mode, in conjunction with the Page Pointer register, R234, which is one of the System registers. This register selects the page to be mapped to Group F and, once set, does not need to be changed if two or more registers on the same page are to be addressed in succession.

Therefore if the Page Pointer, R234, is set to 5, the instructions:

```
spp #5
ld R242, r4
```

will load the contents of working register r4 into the third register of page 5 (R242).

These paged registers hold data and control information relating to the on-chip peripherals, each peripheral always being associated with the same pages and registers to ensure code compatibility between ST9 devices. The number of these registers therefore depends on the peripherals which are present in the specific ST9 family device. In other words, pages only exist if the relevant peripheral is present.

Table 5. Register File Organization

Hex. Address	Decimal Address	Function	Register File Group
F0-FF	240-255	Paged Registers	Group F
E0-EF	224-239	System Registers	Group E
D0-DF	208-223	General Purpose Registers	Group D
C0-CF	192-207		Group C
B0-BF	176-191		Group B
A0-AF	160-175		Group A
90-9F	144-159		Group 9
80-8F	128-143		Group 8
70-7F	112-127		Group 7
60-6F	96-111		Group 6
50-5F	80-95		Group 5
40-4F	64-79		Group 4
30-3F	48-63		Group 3
20-2F	32-47		Group 2
10-1F	16-31		Group 1
00-0F	00-15		Group 0

2.3 SYSTEM REGISTERS

The System registers are listed in Table 2 System Registers (Group E). They are used to perform all the important system settings. Their purpose is described in the following pages. Refer to the chapter dealing with I/O for a description of the PORT[5:0] Data registers.

Table 6. System Registers (Group E)

R239 (EFh)	SSPLR
R238 (EEh)	SSPHR
R237 (EDh)	USPLR
R236 (ECh)	USPHR
R235 (EBh)	MODE REGISTER
R234 (EAh)	PAGE POINTER REGISTER
R233 (E9h)	REGISTER POINTER 1
R232 (E8h)	REGISTER POINTER 0
R231 (E7h)	FLAG REGISTER
R230 (E6h)	CENTRAL INT. CNTL REG
R229 (E5h)	PORT5 DATA REG.
R228 (E4h)	PORT4 DATA REG.
R227 (E3h)	PORT3 DATA REG.
R226 (E2h)	PORT2 DATA REG.
R225 (E1h)	PORT1 DATA REG.
R224 (E0h)	PORT0 DATA REG.

2.3.1 Central Interrupt Control Register

Please refer to the "INTERRUPT" chapter for a detailed description of the ST9 interrupt philosophy.

CENTRAL INTERRUPT CONTROL REGISTER (CICR)
R230 - Read/Write
Register Group: E (System)
Reset Value: 1000 0111 (87h)

7							0
GCEN	TLIP	TLI	IEN	IAM	CPL2	CPL1	CPL0

Bit 7 = **GCEN**: *Global Counter Enable*.
This bit is the Global Counter Enable of the Multifunction Timers. The GCEN bit is ANDed with the CE bit in the TCR Register (only in devices featuring the MFT Multifunction Timer) in order to enable the Timers when both bits are set. This bit is set after the Reset cycle.

Note: If an MFT is not included in the ST9 device, then this bit has no effect.

Bit 6 = **TLIP**: *Top Level Interrupt Pending*.
This bit is set by hardware when a Top Level Interrupt Request is recognized. This bit can also be set by software to simulate a Top Level Interrupt Request.
0: No Top Level Interrupt pending
1: Top Level Interrupt pending

Bit 5 = **TLI**: *Top Level Interrupt bit*.
0: Top Level Interrupt is acknowledged depending on the TLNM bit in the NICR Register.
1: Top Level Interrupt is acknowledged depending on the IEN and TLNM bits in the NICR Register (described in the Interrupt chapter).

Bit 4 = **IEN**: *Interrupt Enable .*
This bit is cleared by interrupt acknowledgement, and set by interrupt return (`iret`). IEN is modified implicitly by `iret`, `ei` and `di` instructions or by an interrupt acknowledge cycle. It can also be explicitly written by the user, but only when no interrupt is pending. Therefore, the user should execute a `di` instruction (or guarantee by other means that no interrupt request can arrive) before any write operation to the CICR register.
0: Disable all interrupts except Top Level Interrupt.
1: Enable Interrupts

Bit 3 = **IAM**: *Interrupt Arbitration Mode*.
This bit is set and cleared by software to select the arbitration mode.
0: Concurrent Mode
1: Nested Mode.

Bits 2:0 = **CPL[2:0]**: *Current Priority Level*.
These three bits record the priority level of the routine currently running (i.e. the Current Priority Level, CPL). The highest priority level is represented by 000, and the lowest by 111. The CPL bits can be set by hardware or software and provide the reference according to which subsequent interrupts are either left pending or are allowed to interrupt the current interrupt service routine. When the current interrupt is replaced by one of a higher priority, the current priority value is automatically stored until required in the NICR register.

SYSTEM REGISTERS (Cont'd)

2.3.2 Flag Register

The Flag Register contains 8 flags which indicate the CPU status. During an interrupt, the flag register is automatically stored in the system stack area and recalled at the end of the interrupt service routine, thus returning the CPU to its original status.

This occurs for all interrupts and, when operating in nested mode, up to seven versions of the flag register may be stored.

FLAG REGISTER (FLAGR)
R231- Read/Write
Register Group: E (System)
Reset value: 0000 0000 (00h**)**

7							0
C	Z	S	V	DA	H	-	DP

Bit 7 = **C**: *Carry Flag*.
The carry flag is affected by:
Addition (`add, addw, adc, adcw`),
Subtraction (`sub, subw, sbc, sbcw`),
Compare (`cp, cpw`),
Shift Right Arithmetic (`sra, sraw`),
Shift Left Arithmetic (`sla, slaw`),
Swap Nibbles (`swap`),
Rotate (`rrc, rrcw, rlc, rlcw, ror, rol`),
Decimal Adjust (`da`),
Multiply and Divide (`mul, div, divws`).

When set, it generally indicates a carry out of the most significant bit position of the register being used as an accumulator (bit 7 for byte operations and bit 15 for word operations).

The carry flag can be set by the Set Carry Flag (`scf`) instruction, cleared by the Reset Carry Flag (`rcf`) instruction, and complemented by the Complement Carry Flag (`ccf`) instruction.

Bit 6 = **Z:** *Zero Flag*. The Zero flag is affected by:
Addition (`add, addw, adc, adcw`),
Subtraction (`sub, subw, sbc, sbcw`),
Compare (`cp, cpw`),
Shift Right Arithmetic (`sra, sraw`),
Shift Left Arithmetic (`sla, slaw`),
Swap Nibbles (`swap`),
Rotate (`rrc, rrcw, rlc, rlcw, ror, rol`),
Decimal Adjust (`da`),
Multiply and Divide (`mul, div, divws`),
Logical (`and, andw, or, orw, xor, xorw, cpl`),
Increment and Decrement (`inc, incw, dec, decw`),
Test (`tm, tmw, tcm, tcmw, btset`).

In most cases, the Zero flag is set when the contents of the register being used as an accumulator become zero, following one of the above operations.

Bit 5 = **S**: *Sign Flag*.
The Sign flag is affected by the same instructions as the Zero flag.

The Sign flag is set when bit 7 (for a byte operation) or bit 15 (for a word operation) of the register used as an accumulator is one.

Bit 4 = **V**: *Overflow Flag*.
The Overflow flag is affected by the same instructions as the Zero and Sign flags.

When set, the Overflow flag indicates that a two's-complement number, in a result register, is in error, since it has exceeded the largest (or is less than the smallest), number that can be represented in two's-complement notation.

Bit 3 = **DA**: *Decimal Adjust Flag*.
The DA flag is used for BCD arithmetic. Since the algorithm for correcting BCD operations is different for addition and subtraction, this flag is used to specify which type of instruction was executed last, so that the subsequent Decimal Adjust (`da`) operation can perform its function correctly. The DA flag cannot normally be used as a test condition by the programmer.

Bit 2 = **H**: *Half Carry Flag.*
The H flag indicates a carry out of (or a borrow into) bit 3, as the result of adding or subtracting two 8-bit bytes, each representing two BCD digits. The H flag is used by the Decimal Adjust (`da`) instruction to convert the binary result of a previous addition or subtraction into the correct BCD result. Like the DA flag, this flag is not normally accessed by the user.

Bit 1 = Reserved bit (must be 0).

Bit 0 = **DP**: *Data/Program Memory Flag*.
This bit indicates the memory area addressed. Its value is affected by the Set Data Memory (`sdm`) and Set Program Memory (`spm`) instructions. Refer to the Memory Management Unit for further details.

SYSTEM REGISTERS (Cont'd)

If the bit is set, data is accessed using the Data Pointers (DPRs registers), otherwise it is pointed to by the Code Pointer (CSR register); therefore, the user initialization routine must include a Sdm instruction. Note that code is always pointed to by the Code Pointer (CSR).

Note: In the current ST9 devices, the DP flag is only for compatibility with software developed for the first generation of ST9 devices. With the single memory addressing space, its use is now redundant. It must be kept to 1 with a Sdm instruction at the beginning of the program to ensure a normal use of the different memory pointers.

2.3.3 Register Pointing Techniques

Two registers within the System register group, are used as pointers to the working registers. Register Pointer 0 (R232) may be used on its own as a single pointer to a 16-register working space, or in conjunction with Register Pointer 1 (R233), to point to two separate 8-register spaces.

For the purpose of register pointing, the 16 register groups of the register file are subdivided into 32 8-register blocks. The values specified with the Set Register Pointer instructions refer to the blocks to be pointed to in twin 8-register mode, or to the lower 8-register block location in single 16-register mode.

The Set Register Pointer instructions srp, srp0 and srp1 automatically inform the CPU whether the Register File is to operate in single 16-register mode or in twin 8-register mode. The srp instruction selects the single 16-register group mode and specifies the location of the lower 8-register block, while the srp0 and srp1 instructions automatically select the twin 8-register group mode and specify the locations of each 8-register block.

There is no limitation on the order or position of these register groups, other than that they must start on an 8-register boundary in twin 8-register mode, or on a 16-register boundary in single 16-register mode.

The block number should always be an even number in single 16-register mode. The 16-register group will always start at the block whose number is the nearest even number equal to or lower than the block number specified in the srp instruction. Avoid using odd block numbers, since this can be confusing if twin mode is subsequently selected.

Thus:

srp #3 will be interpreted as srp #2 and will allow using R16 ..R31 as r0 .. r15.

In single 16-register mode, the working registers are referred to as r0 to r15. In twin 8-register mode, registers r0 to r7 are in the block pointed to by RP0 (by means of the srp0 instruction), while registers r8 to r15 are in the block pointed to by RP1 (by means of the srp1 instruction).

Caution: *Group D registers can only be accessed as working registers using the Register Pointers, or by means of the Stack Pointers. They cannot be addressed explicitly in the form "Rxxx".*

SYSTEM REGISTERS (Cont'd)

POINTER 0 REGISTER (RP0)
R232 - Read/Write
Register Group: E (System)
Reset Value: xxxx xx00 (xxh)

7							0
RG4	RG3	RG2	RG1	RG0	RPS	0	0

Bits 7:3 = **RG[4:0]**: *Register Group number.*
These bits contain the number (in the range 0 to 31) of the register block specified in the `srp0` or `srp` instructions. In single 16-register mode the number indicates the lower of the two 8-register blocks to which the 16 working registers are to be mapped, while in twin 8-register mode it indicates the 8-register block to which `r0` to `r7` are to be mapped.

Bit 2 = **RPS**: *Register Pointer Selector.*
This bit is set by the instructions `srp0` and `srp1` to indicate that the twin register pointing mode is selected. The bit is reset by the `srp` instruction to indicate that the single register pointing mode is selected.
0: Single register pointing mode
1: Twin register pointing mode

Bits 1:0: Reserved. Forced by hardware to zero.

POINTER 1 REGISTER (RP1)
R233 - Read/Write
Register Group: E (System)
Reset Value: xxxx xx00 (xxh)

7							0
RG4	RG3	RG2	RG1	RG0	RPS	0	0

This register is only used in the twin register pointing mode. When using the single register pointing mode, or when using only one of the twin register groups, the RP1 register must be considered as RESERVED and may NOT be used as a general purpose register.

Bits 7:3 = **RG[4:0]:** *Register Group number.*
These bits contain the number (in the range 0 to 31) of the 8-register block specified in the `srp1` instruction, to which `r8` to `r15` are to be mapped.

Bit 2 = **RPS**: *Register Pointer Selector.*
This bit is set by the `srp0` and `srp1` instructions to indicate that the twin register pointing mode is selected. The bit is reset by the `srp` instruction to indicate that the single register pointing mode is selected.
0: Single register pointing mode
1: Twin register pointing mode

Bits 1:0: Reserved. Forced by hardware to zero.

SYSTEM REGISTERS (Cont'd)

Figure 13. Pointing to a single group of 16 registers

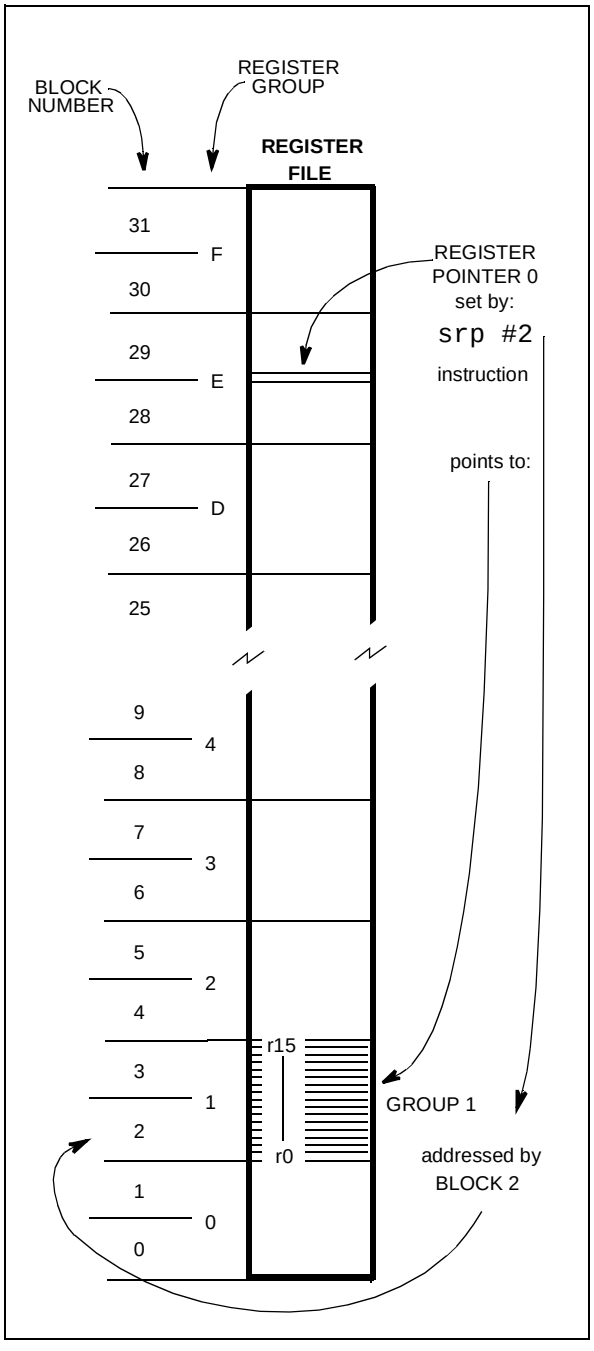
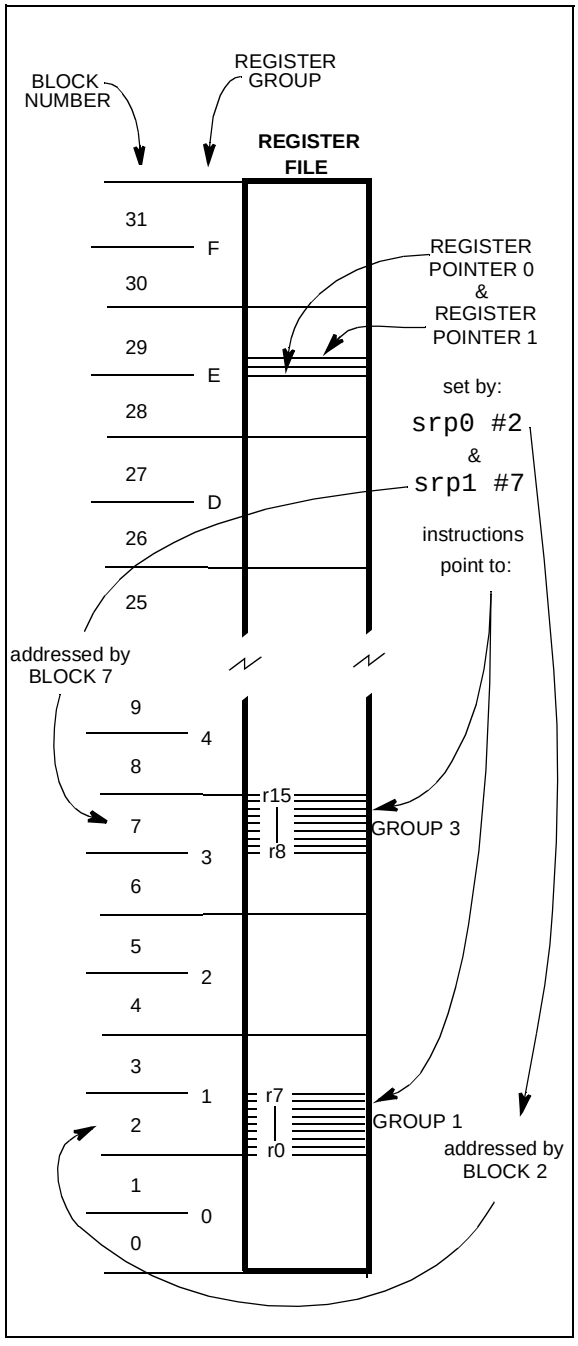


Figure 14. Pointing to two groups of 8 registers



SYSTEM REGISTERS (Cont'd)

2.3.4 Paged Registers

Up to 64 pages, each containing 16 registers, may be mapped to Group F. These paged registers hold data and control information relating to the on-chip peripherals, each peripheral always being associated with the same pages and registers to ensure code compatibility between ST9 devices. The number of these registers depends on the peripherals present in the specific ST9 device. In other words, pages only exist if the relevant peripheral is present.

The paged registers are addressed using the normal register addressing modes, in conjunction with the Page Pointer register, R234, which is one of the System registers. This register selects the page to be mapped to Group F and, once set, does not need to be changed if two or more registers on the same page are to be addressed in succession.

Thus the instructions:

```
spp #5
ld R242, r4
```

will load the contents of working register r4 into the third register of page 5 (R242).

Warning: During an interrupt, the PPR register is not saved automatically in the stack. If needed, it should be saved/restored by the user within the interrupt routine.

PAGE POINTER REGISTER (PPR)
R234 - Read/Write
Register Group: E (System)
Reset value: xxxx xx00 (xxh**)**

7							0
PP5	PP4	PP3	PP2	PP1	PP0	0	0

Bits 7:2 = **PP[5:0]**: *Page Pointer*.
These bits contain the number (in the range 0 to 63) of the page specified in the spp instruction. Once the page pointer has been set, there is no need to refresh it unless a different page is required.

Bits 1:0: Reserved. Forced by hardware to 0.

2.3.5 Mode Register

The Mode Register allows control of the following operating parameters:

– Selection of internal or external System and User Stack areas,
– Management of the clock frequency,
– Enabling of Bus request and Wait signals when interfacing to external memory.

MODE REGISTER (MODER)
R235 - Read/Write
Register Group: E (System)
Reset value: 1110 0000 (E0h)

7							0
SSP	USP	DIV2	PRS2	PRS1	PRS0	BRQEN	HIMP

Bit 7 = **SSP**: *System Stack Pointer*.
This bit selects an internal or external System Stack area.
0: External system stack area, in memory space.
1: Internal system stack area, in the Register File (reset state).

Bit 6 = **USP**: *User Stack Pointer*.
This bit selects an internal or external User Stack area.
0: External user stack area, in memory space.
1: Internal user stack area, in the Register File (reset state).

Bit 5 = **DIV2**: *Crystal Oscillator Clock Divided by 2*.
This bit controls the divide-by-2 circuit operating on the crystal oscillator clock (CLOCK1).
0: Clock divided by 1
1: Clock divided by 2

Bits 4:2 = **PRS[2:0]**: *CPUCLK Prescaler*.
These bits load the prescaler division factor for the internal clock (INTCLK). The prescaler factor selects the internal clock frequency, which can be divided by a factor from 1 to 8. Refer to the Reset and Clock Control chapter for further information.

Bit 1 = **BRQEN**: *Bus Request Enable*.
0: External Memory Bus Request disabled
1: External Memory Bus Request enabled on $\overline{\text{BREQ}}$ pin (where available).

Note: Disregard this bit if $\overline{\text{BREQ}}$ pin is not available.

Bit 0 = **HIMP**: *High Impedance Enable*.
When a port is programmed as Address and Data lines to interface external Memory, these lines and the Memory interface control lines (AS, DS, R/W) can be forced into the High Impedance state.
0: External memory interface lines in normal state
1: High Impedance state.

Note: Setting the HIMP bit is recommended for noise reduction when only internal Memory is used.

If the memory access ports are declared as an address AND as an I/O port (for example: P10... P14 = Address, and P15... P17 = I/O), the HIMP bit has no effect on the I/O lines.

2.3.6 Stack Pointers

Two separate, double-register stack pointers are available: the System Stack Pointer and the User Stack Pointer, both of which can address registers or memory.

The stack pointers point to the "bottom" of the stacks which are filled using the push commands and emptied using the pop commands. The stack pointer is automatically pre-decremented when data is "pushed" in and post-incremented when data is "popped" out.

The push and pop commands used to manage the System Stack may be addressed to the User Stack by adding the suffix "u". To use a stack instruction for a word, the suffix "w" is added. These suffixes may be combined.

When bytes (or words) are "popped" out from a stack, the contents of the stack locations are unchanged until fresh data is loaded. Thus, when data is "popped" from a stack area, the stack contents remain unchanged.

Note: Instructions such as: `pushuw RR236` or `pushw RR238`, as well as the corresponding `pop` instructions (where R236 & R237, and R238 & R239 are themselves the user and system stack pointers respectively), must not be used, since the pointer values are themselves automatically changed by the `push` or `pop` instruction, thus corrupting their value.

System Stack

The System Stack is used for the temporary storage of system and/or control data, such as the Flag register and the Program counter.

The following automatically push data onto the System Stack:

– **Interrupts**

When entering an interrupt, the PC and the Flag Register are pushed onto the System Stack. If the ENCSR bit in the EMR2 register is set, then the Code Segment Register is also pushed onto the System Stack.

– **Subroutine Calls**

When a `call` instruction is executed, only the PC is pushed onto stack, whereas when a `calls` instruction (call segment) is executed, both the PC and the Code Segment Register are pushed onto the System Stack.

– **Link Instruction**

The `link` or `linku` instructions create a C language stack frame of user-defined length in the System or User Stack.

All of the above conditions are associated with their counterparts, such as return instructions, which pop the stored data items off the stack.

User Stack

The User Stack provides a totally user-controlled stacking area.

The User Stack Pointer consists of two registers, R236 and R237, which are both used for addressing a stack in memory. When stacking in the Register File, the User Stack Pointer High Register, R236, becomes redundant but must be considered as reserved.

Stack Pointers

Both System and User stacks are pointed to by double-byte stack pointers. Stacks may be set up in RAM or in the Register File. Only the lower byte will be required if the stack is in the Register File. The upper byte must then be considered as reserved and must not be used as a general purpose register.

The stack pointer registers are located in the System Group of the Register File, this is illustrated in Table 2 System Registers (Group E).

Stack Location

Care is necessary when managing stacks as there is no limit to stack sizes apart from the bottom of any address space in which the stack is placed. Consequently programmers are advised to use a stack pointer value as high as possible, particularly when using the Register File as a stacking area.

Group D is a good location for a stack in the Register File, since it is the highest available area. The stacks may be located anywhere in the first 14 groups of the Register File (internal stacks) or in RAM (external stacks).

Note. Stacks must not be located in the Paged Register Group or in the System Register Group.

SYSTEM REGISTERS (Cont'd)

USER STACK POINTER HIGH REGISTER (USPHR)
R236 - Read/Write
Register Group: E (System)
Reset value: undefined

7							0
USP15	USP14	USP13	USP12	USP11	USP10	USP9	USP8

USER STACK POINTER LOW REGISTER (USPLR)
R237 - Read/Write
Register Group: E (System)
Reset value: undefined

7							0
USP7	USP6	USP5	USP4	USP3	USP2	USP1	USP0

SYSTEM STACK POINTER HIGH REGISTER (SSPHR)
R238 - Read/Write
Register Group: E (System)
Reset value: undefined

7							0
SSP15	SSP14	SSP13	SSP12	SSP11	SSP10	SSP9	SSP8

SYSTEM STACK POINTER LOW REGISTER (SSPLR)
R239 - Read/Write
Register Group: E (System)
Reset value: undefined

7							0
SSP7	SSP6	SSP5	SSP4	SSP3	SSP2	SSP1	SSP0

Figure 15. Internal Stack Mode

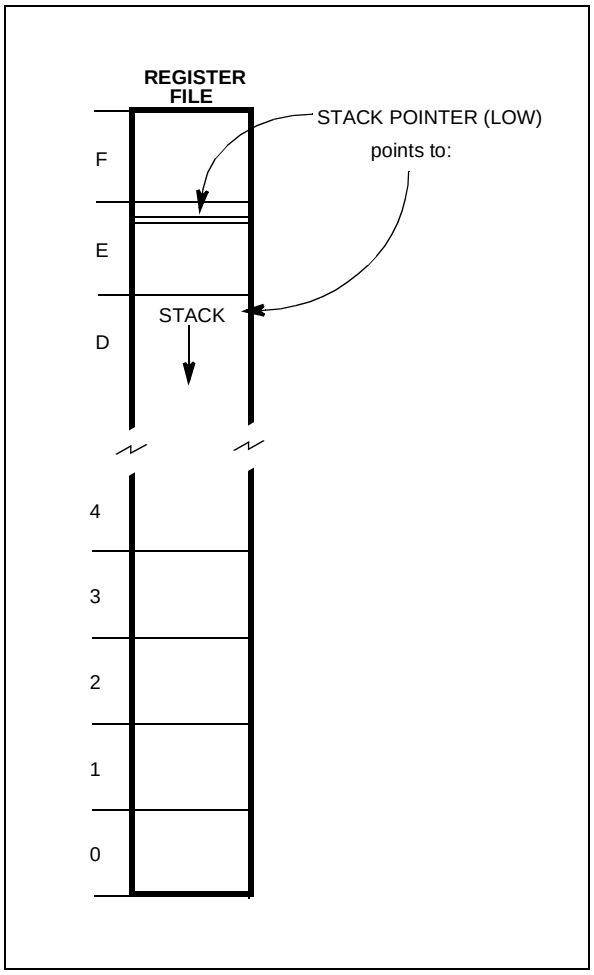
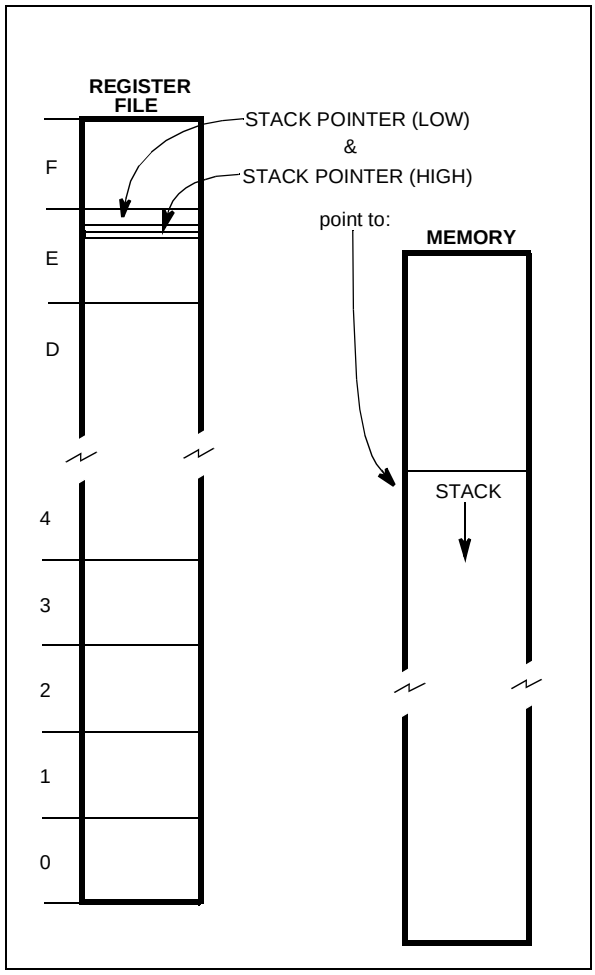


Figure 16. External Stack Mode



2.4 MEMORY ORGANIZATION

Code and data are accessed within the same linear address space. All of the physically separate memory areas, including the internal ROM, internal RAM and external memory are mapped in a common address space.

The ST9 provides a total addressable memory space of 4 Mbytes. This address space is arranged as 64 segments of 64 Kbytes; each segment is again subdivided into four 16 Kbyte pages.

The mapping of the various memory areas (internal RAM or ROM, external memory) differs from device to device. Each 64-Kbyte physical memory segment is mapped either internally or externally; if the memory is internal and smaller than 64 Kbytes, the remaining locations in the 64-Kbyte segment are not used (reserved).

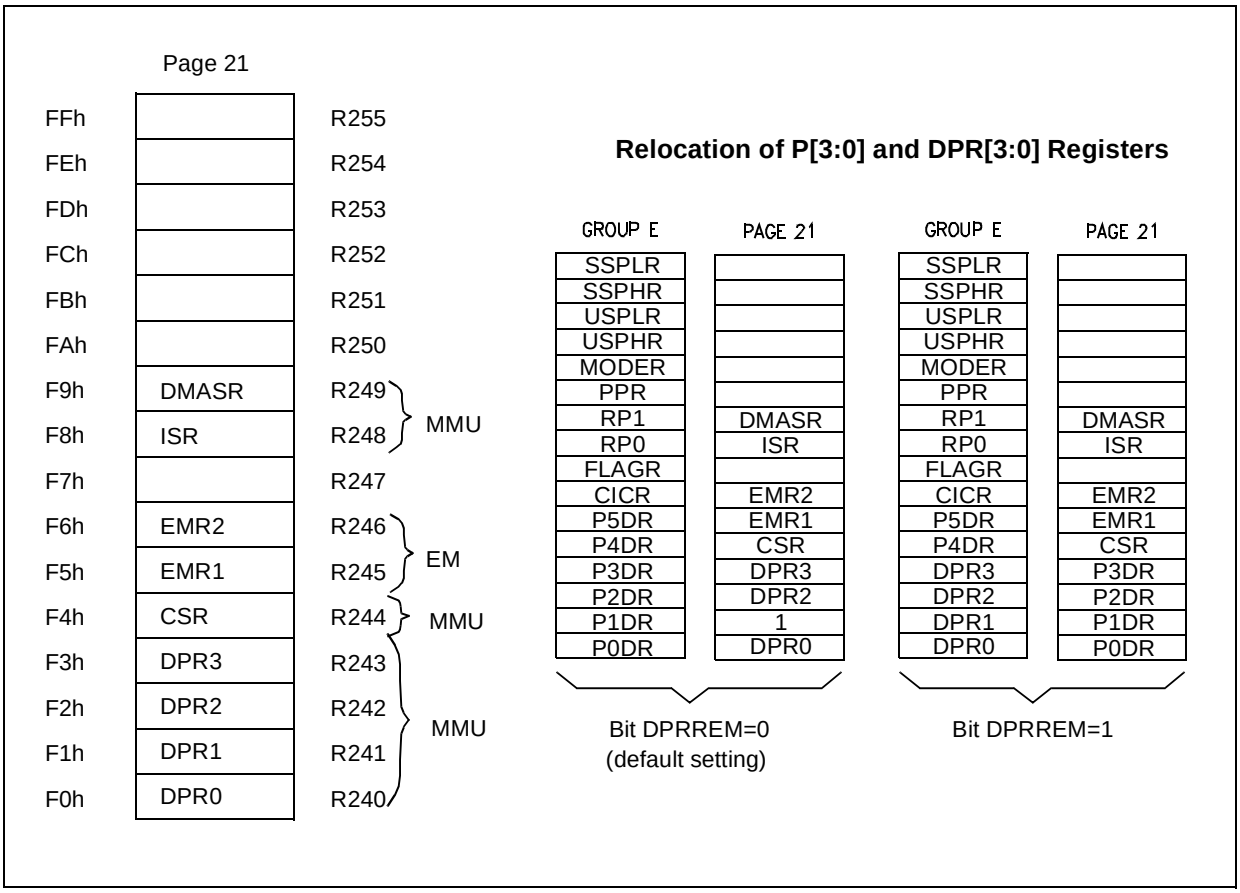
Refer to the Register and Memory Map Chapter for more details on the memory map.

2.5 MEMORY MANAGEMENT UNIT

The CPU Core includes a Memory Management Unit (MMU) which must be programmed to perform memory accesses (even if external memory is not used).

The MMU is controlled by 7 registers and 2 bits (ENCSR and DPRREM) present in EMR2, which may be written and read by the user program. These registers are mapped within group F, Page 21 of the Register File. The 7 registers may be sub-divided into 2 main groups: a first group of four 8-bit registers (DPR[3:0]), and a second group of three 6-bit registers (CSR, ISR, and DMASR). The first group is used to extend the address during Data Memory access (DPR[3:0]). The second is used to manage Program and Data Memory accesses during Code execution (CSR), Interrupts Service Routines (ISR or CSR), and DMA transfers (DMASR or ISR).

Figure 17. Page 21 Registers



2.6 ADDRESS SPACE EXTENSION

To manage 4 Mbytes of addressing space, it is necessary to have 22 address bits. The MMU adds 6 bits to the usual 16-bit address, thus translating a 16-bit virtual address into a 22-bit physical address. There are 2 different ways to do this depending on the memory involved and on the operation being performed.

2.6.1 Addressing 16-Kbyte Pages

This extension mode is implicitly used to address Data memory space if no DMA is being performed.

The Data memory space is divided into 4 pages of 16 Kbytes. Each one of the four 8-bit registers (DPR[3:0], Data Page Registers) selects a different 16-Kbyte page. The DPR registers allow access to the entire memory space which contains 256 pages of 16 Kbytes.

Data paging is performed by extending the 14 LSB of the 16-bit address with the contents of a DPR register. The two MSBs of the 16-bit address are interpreted as the identification number of the DPR register to be used. Therefore, the DPR registers are involved in the following virtual address ranges:

DPR0: from 0000h to 3FFFh;

DPR1: from 4000h to 7FFFh;

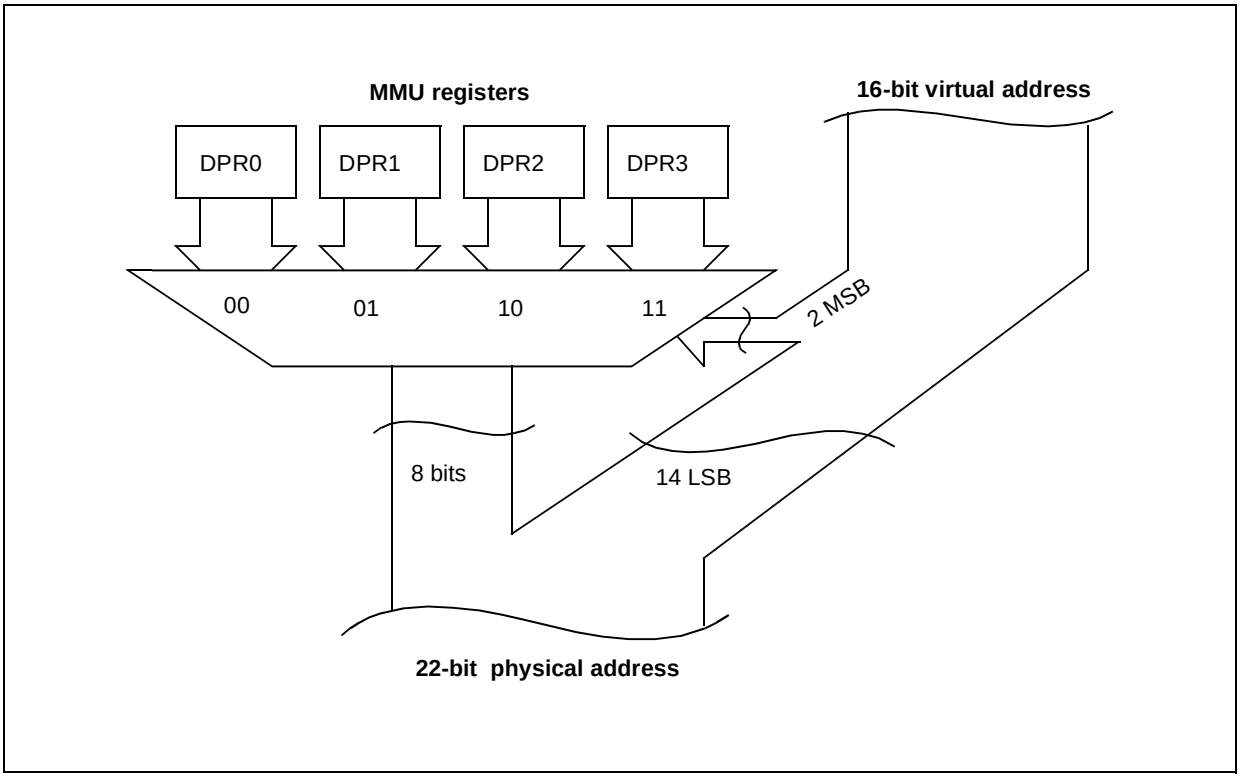
DPR2: from 8000h to BFFFh;

DPR3: from C000h to FFFFh.

The contents of the selected DPR register specify one of the 256 possible data memory pages. This 8-bit data page number, in addition to the remaining 14-bit page offset address forms the physical 22-bit address (see Figure 10).

A DPR register cannot be modified via an addressing mode that uses the same DPR register. For instance, the instruction “POPW DPR0” is legal only if the stack is kept either in the register file or in a memory location above 8000h, where DPR2 and DPR3 are used. Otherwise, since DPR0 and DPR1 are modified by the instruction, unpredictable behaviour could result.

Figure 18. Addressing via DPR[3:0]



ADDRESS SPACE EXTENSION (Cont'd)

2.6.2 Addressing 64-Kbyte Segments

This extension mode is used to address Data memory space during a DMA and Program memory space during any code execution (normal code and interrupt routines).

Three registers are used: CSR, ISR, and DMASR. The 6-bit contents of one of the registers CSR, ISR, or DMASR define one out of 64 Memory segments of 64 Kbytes within the 4 Mbytes address space. The register contents represent the 6 MSBs of the memory address, whereas the 16 LSBs of the address (intra-segment address) are given by the virtual 16-bit address (see Figure 11).

2.7 MMU REGISTERS

The MMU uses 7 registers mapped into Group F, Page 21 of the Register File and 2 bits of the EMR2 register.

Most of these registers do not have a default value after reset.

2.7.1 DPR[3:0]: Data Page Registers

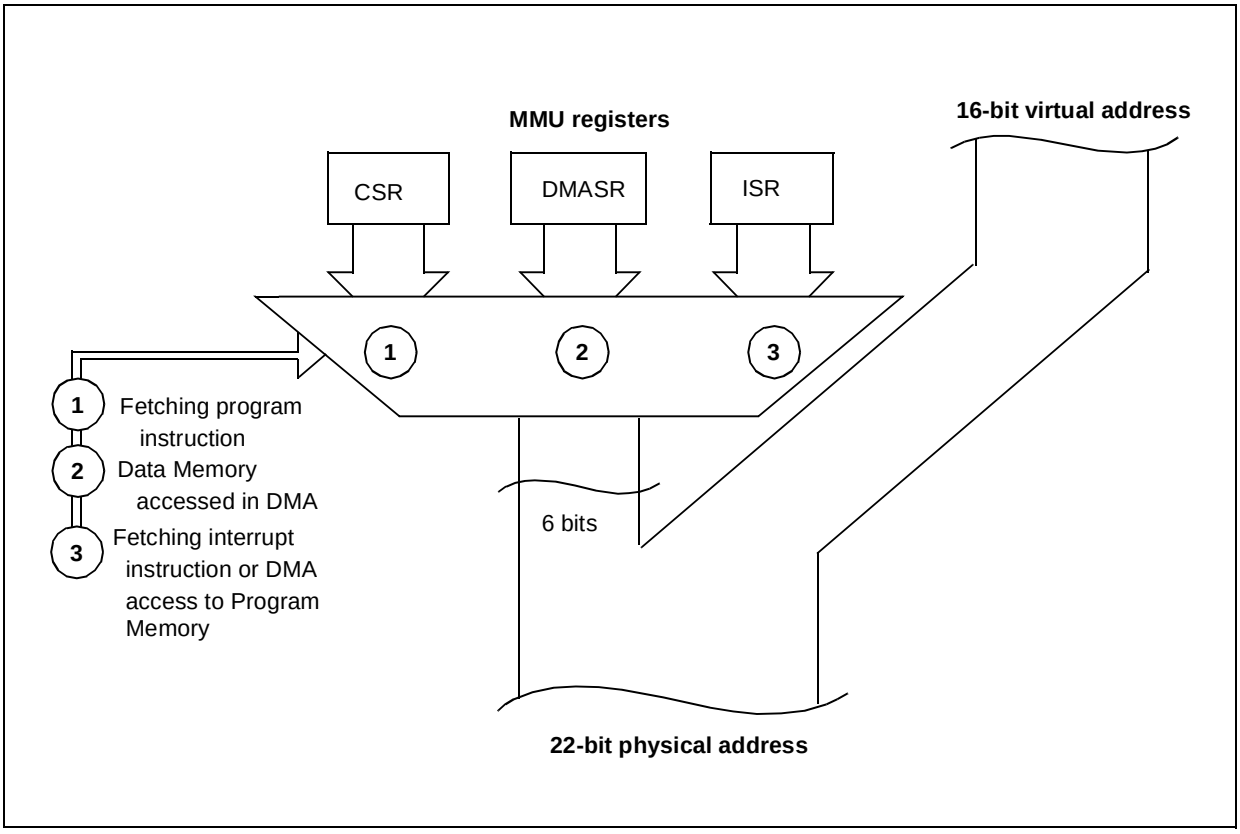
The DPR[3:0] registers allow access to the entire 4 Mbyte memory space composed of 256 pages of 16 Kbytes.

2.7.1.1 Data Page Register Relocation

If these registers are to be used frequently, they may be relocated in register group E, by programming bit 5 of the EMR2-R246 register in page 21. If this bit is set, the DPR[3:0] registers are located at R224-227 in place of the Port 0-3 Data Registers, which are re-mapped to the default DPR's locations: R240-243 page 21.

Data Page Register relocation is illustrated in Figure 9.

Figure 19. Addressing via CSR, ISR, and DMASR



MMU REGISTERS (Cont'd)

DATA PAGE REGISTER 0 (DPR0)
R240 - Read/Write
Register Page: 21
Reset value: undefined

This register is relocated to R224 if EMR2.5 is set.

7							0
DPR0_7	DPR0_6	DPR0_5	DPR0_4	DPR0_3	DPR0_2	DPR0_1	DPR0_0

Bits 7:0 = **DPR0_[7:0]**: These bits define the 16-Kbyte Data Memory page number. They are used as the most significant address bits (A21-14) to extend the address during a Data Memory access. The DPR0 register is used when addressing the virtual address range 0000h-3FFFh.

DATA PAGE REGISTER 1 (DPR1)
R241 - Read/Write
Register Page: 21
Reset value: undefined

This register is relocated to R225 if EMR2.5 is set.

7							0
DPR1_7	DPR1_6	DPR1_5	DPR1_4	DPR1_3	DPR1_2	DPR1_1	DPR1_0

Bits 7:0 = **DPR1_[7:0]**: These bits define the 16-Kbyte Data Memory page number. They are used as the most significant address bits (A21-14) to extend the address during a Data Memory access. The DPR1 register is used when addressing the virtual address range 4000h-7FFFh.

DATA PAGE REGISTER 2 (DPR2)
R242 - Read/Write
Register Page: 21
Reset value: undefined

This register is relocated to R226 if EMR2.5 is set.

7							0
DPR2_7	DPR2_6	DPR2_5	DPR2_4	DPR2_3	DPR2_2	DPR2_1	DPR2_0

Bits 7:0 = **DPR2_[7:0]**: These bits define the 16-Kbyte Data memory page. They are used as the most significant address bits (A21-14) to extend the address during a Data memory access. The DPR2 register is involved when the virtual address is in the range 8000h-BFFFh.

DATA PAGE REGISTER 3 (DPR3)
R243 - Read/Write
Register Page: 21
Reset value: undefined

This register is relocated to R227 if EMR2.5 is set.

7							0
DPR3_7	DPR3_6	DPR3_5	DPR3_4	DPR3_3	DPR3_2	DPR3_1	DPR3_0

Bits 7:0 = **DPR3_[7:0]**: These bits define the 16-Kbyte Data memory page. They are used as the most significant address bits (A21-14) to extend the address during a Data memory access. The DPR3 register is involved when the virtual address is in the range C000h-FFFFh.

MMU REGISTERS (Cont'd)

2.7.2 CSR: Code Segment Register

This register selects the 64-Kbyte code segment being used at run-time to access instructions. It can also be used to access data if the `spm` instruction has been executed (or `ldpp, ldpd, lddp`). Only the 6 LSBs of the CSR register are implemented, and bits 6 and 7 are reserved. The CSR register allows access to the entire memory space, divided into 64 segments of 64 Kbytes.

To generate the 22-bit Program memory address, the contents of the CSR register is directly used as the 6 MSBs, and the 16-bit virtual address as the 16 LSBs.

Note: The CSR register should only be read and not written for data operations (there are some exceptions which are documented in the following paragraph). It is, however, modified either directly by means of the `jps` and `calls` instructions, or indirectly via the stack, by means of the `rets` instruction.

CODE SEGMENT REGISTER (CSR)
R244 **-** Read/Write
Register Page: 21
Reset value: 0000 0000 (00h)

7							0
0	0	CSR_5	CSR_4	CSR_3	CSR_2	CSR_1	CSR_0

Bits 7:6 = Reserved, keep in reset state.

Bits 5:0 = **CSR_[5:0]**: These bits define the 64-Kbyte memory segment (among 64) which contains the code being executed. These bits are used as the most significant address bits (A21-16).

2.7.3 ISR: Interrupt Segment Register

INTERRUPT SEGMENT REGISTER (ISR)
R248 - Read/Write
Register Page: 21
Reset value: undefined

7							0
0	0	ISR_5	ISR_4	ISR_3	ISR_2	ISR_1	ISR_0

ISR and ENCSR bit (EMR2 register) are also described in the chapter relating to Interrupts, please refer to this description for further details.

Bits 7:6 = Reserved, keep in reset state.

Bits 5:0 = **ISR_[5:0]**: These bits define the 64-Kbyte memory segment (among 64) which contains the interrupt vector table and the code for interrupt service routines and DMA transfers (when the PS bit of the DAPR register is reset). These bits are used as the most significant address bits (A21-16). The ISR is used to extend the address space in two cases:

– Whenever an interrupt occurs: ISR points to the 64-Kbyte memory segment containing the interrupt vector table and the interrupt service routine code. See also the Interrupts chapter.
– During DMA transactions between the peripheral and memory when the PS bit of the DAPR register is reset : ISR points to the 64 K-byte Memory segment that will be involved in the DMA transaction.

2.7.4 DMASR: DMA Segment Register

DMA SEGMENT REGISTER (DMASR)
R249 - Read/Write
Register Page: 21
Reset value: undefined

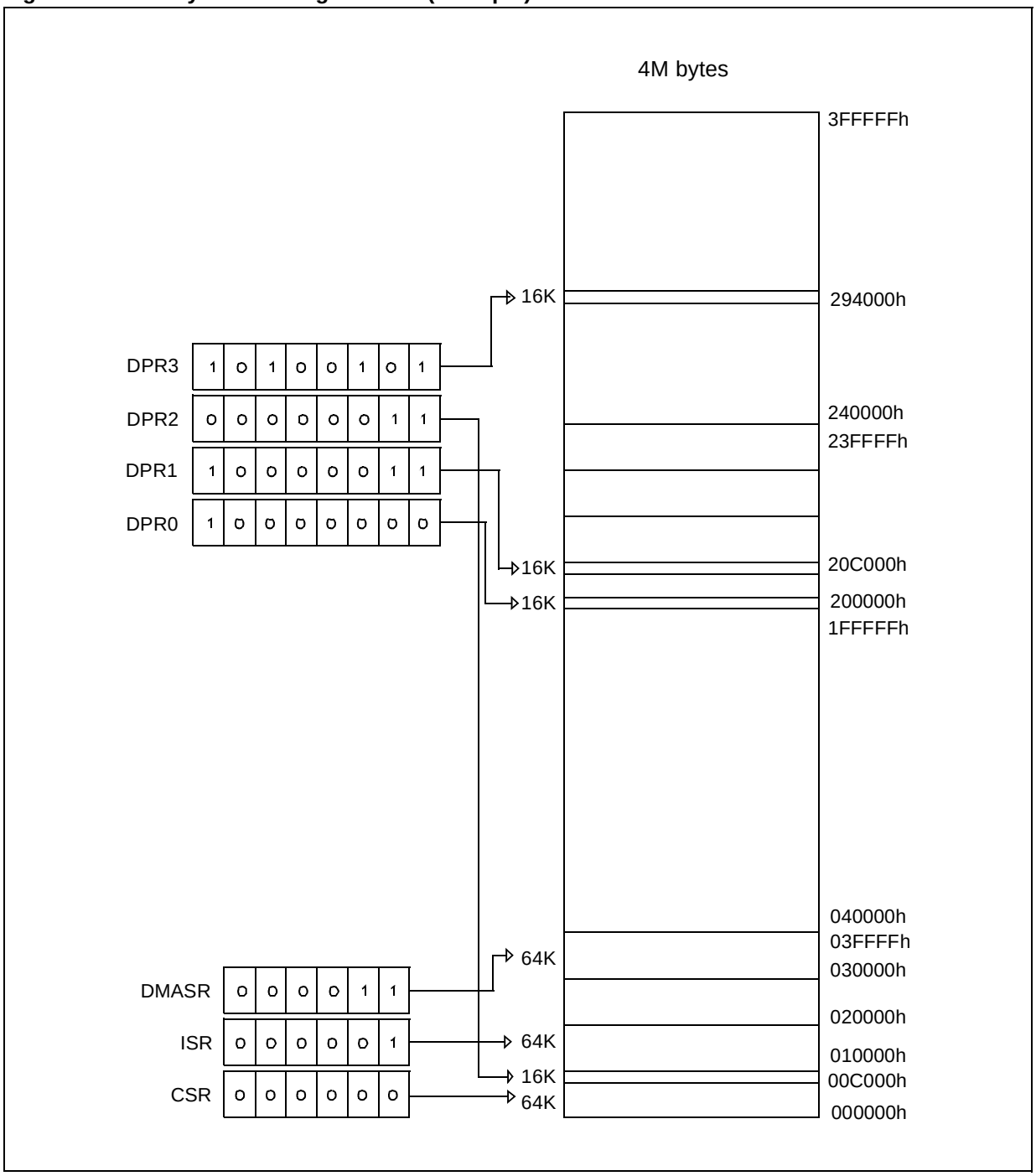
7							0
0	0	DMA SR_5	DMA SR_4	DMA SR_3	DMA SR_2	DMA SR_1	DMA SR_0

Bits 7:6 = Reserved, keep in reset state.

Bits 5:0 = **DMASR_[5:0]**: These bits define the 64-Kbyte Memory segment (among 64) used when a DMA transaction is performed between the peripheral's data register and Memory, with the PS bit of the DAPR register set. These bits are used as the most significant address bits (A21-16). If the PS bit is reset, the ISR register is used to extend the address.

MMU REGISTERS (Cont'd)

Figure 20. Memory Addressing Scheme (example)



2.8 MMU USAGE

2.8.1 Normal Program Execution

Program memory is organized as a set of 64-Kbyte segments. The program can span as many segments as needed, but a procedure cannot stretch across segment boundaries. `jps`, `calls` and `rets` instructions, which automatically modify the CSR, must be used to jump across segment boundaries. Writing to the CSR is forbidden during normal program execution because it is not synchronized with the opcode fetch. This could result in fetching the first byte of an instruction from one memory segment and the second byte from another. Writing to the CSR is allowed when it is not being used, i.e during an interrupt service routine if ENCSR is reset.

Note that a routine must always be called in the same way, i.e. either always with `call` or always with `calls`, depending on whether the routine ends with `ret` or `rets`. This means that if the routine is written without prior knowledge of the location of other routines which call it, and all the program code does not fit into a single 64-Kbyte segment, then `calls`/`rets` should be used.

In typical microcontroller applications, less than 64 Kbytes of RAM are used, so the four Data space pages are normally sufficient, and no change of DPR[3:0] is needed during Program execution. It may be useful however to map part of the ROM into the data space if it contains strings, tables, bit maps, etc.

If there is to be frequent use of paging, the user can set bit 5 (DPRREM) in register R246 (EMR2) of Page 21. This swaps the location of registers DPR[3:0] with that of the data registers of Ports 0-3. In this way, DPR registers can be accessed without the need to save/set/restore the Page Pointer Register. Port registers are therefore moved to page 21. Applications that require a lot of paging typically use more than 64 Kbytes of external memory, and as ports 0, 1 and 2 are required to address it, their data registers are unused.

2.8.2 Interrupts

The ISR register has been created so that the interrupt routines may be found by means of the same vector table even after a segment jump/call.

When an interrupt occurs, the CPU behaves in one of 2 ways, depending on the value of the ENCSR bit in the EMR2 register (R246 on Page 21).

If this bit is reset (default condition), the CPU works in original ST9 compatibility mode. For the duration of the interrupt service routine, the ISR is used instead of the CSR, and the interrupt stack frame is kept exactly as in the original ST9 (only the PC and flags are pushed). This avoids the need to save the CSR on the stack in the case of an interrupt, ensuring a fast interrupt response time. The drawback is that it is not possible for an interrupt service routine to perform segment `calls`/`jps`: these instructions would update the CSR, which, in this case, is not used (ISR is used instead). The code size of all interrupt service routines is thus limited to 64 Kbytes.

If, instead, bit 6 of the EMR2 register is set, the ISR is used only to point to the interrupt vector table and to initialize the CSR at the beginning of the interrupt service routine: the old CSR is pushed onto the stack together with the PC and the flags, and then the CSR is loaded with the ISR. In this case, an `iret` will also restore the CSR from the stack. This approach lets interrupt service routines access the whole 4-Mbyte address space. The drawback is that the interrupt response time is slightly increased, because of the need to also save the CSR on the stack. Compatibility with the original ST9 is also lost in this case, because the interrupt stack frame is different; this difference, however, would not be noticeable for a vast majority of programs.

Data memory mapping is independent of the value of bit 6 of the EMR2 register, and remains the same as for normal code execution: the stack is the same as that used by the main program, as in the ST9. If the interrupt service routine needs to access additional Data memory, it must save one (or more) of the DPRs, load it with the needed memory page and restore it before completion.

2.8.3 DMA

Depending on the PS bit in the DAPR register (see DMA chapter) DMA uses either the ISR or the DMASR for memory accesses: this guarantees that a DMA will always find its memory segment(s), no matter what segment changes the application has performed. Unlike interrupts, DMA transactions cannot save/restore paging registers, so a dedicated segment register (DMASR) has been created. Having only one register of this kind means that all DMA accesses should be programmed in one of the two following segments: the one pointed to by the ISR (when the PS bit of the DAPR register is reset), and the one referenced by the DMASR (when the PS bit is set).

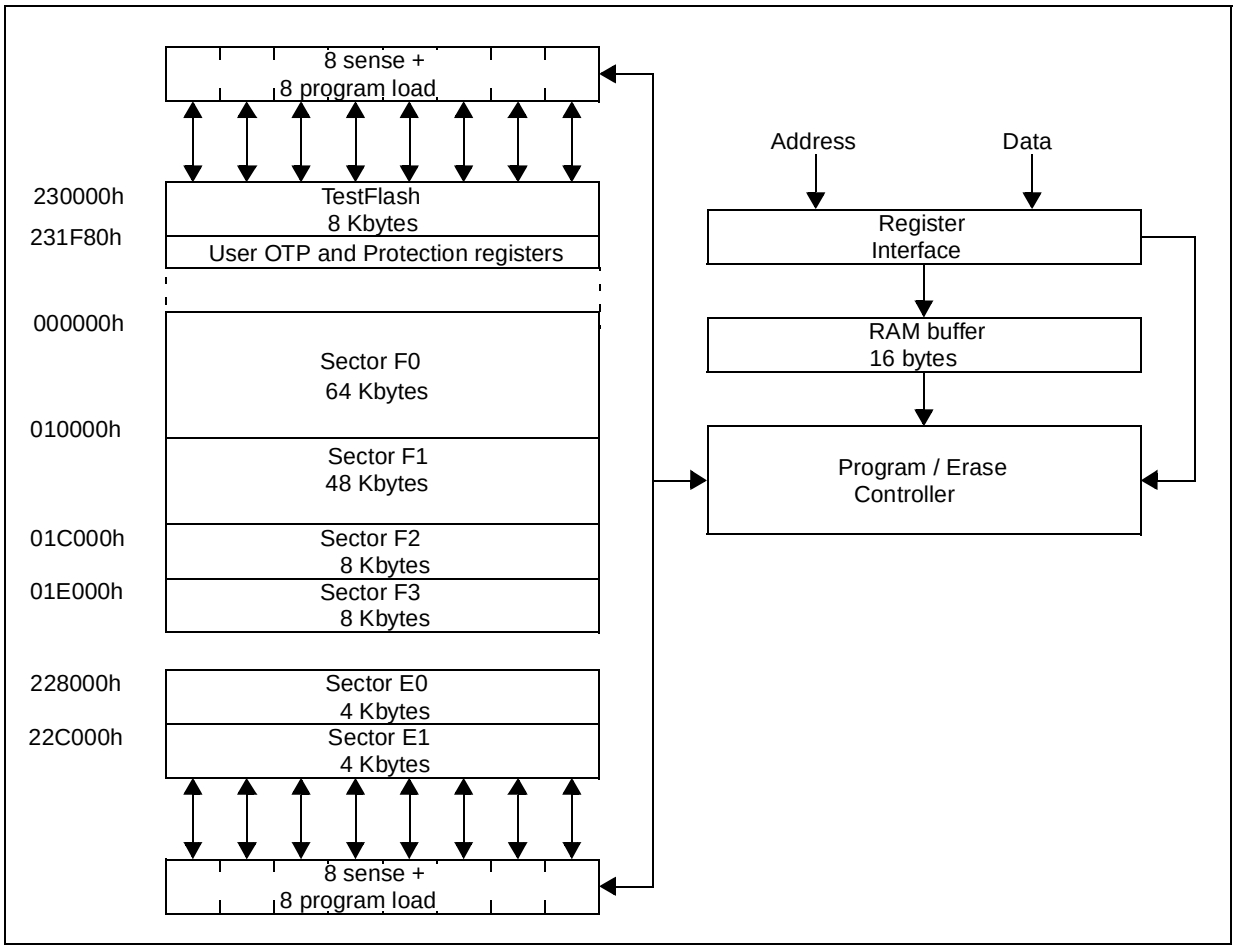
3 SINGLE VOLTAGE FLASH & EEPROM

3.1 INTRODUCTION

The Flash circuitry contains one array divided in two main parts that can each be read independently. The first part contains the main Flash array for code storage, a reserved array (TestFlash) for system routines and a 128-byte area available as one time programmable memory (OTP). The second part contains the two dedicated Flash sectors used for EEPROM Hardware Emulation.

The write operations of the two parts are managed by an embedded Program/Erase Controller, that uses a dedicated ROM. Through a dedicated RAM buffer the Flash and the EEPROM can be written in blocks of 16 bytes.

Figure 21. Flash Memory Structure (Example for 128K Flash device)



3.2 FUNCTIONAL DESCRIPTION

3.2.1 Structure

The Flash memory is composed of three parts (see following table):

– 1 reserved sector for system routines (TestFlash including user OTP area)

– 4 main sectors for code

– 2 sectors of the same size for EEPROM emulation

The last 128 bytes of the TestFlash are available to the user as an OTP area. The user can program these bytes, but cannot erase them. The last 4 bytes of this OTP area (231FFCh to 231FFFh) are reserved for the Non-Volatile Protection registers and cannot be used as a storage area (see Section 3.6 PROTECTION STRATEGY for more details).

3.2.2 EEPROM Emulation

A hardware EEPROM emulation is implemented using special flash sectors E0 and E1 to emulate an EEPROM memory whose size is 1/4 of a sector (1 Kbyte). This EEPROM can be directly addressed from 220000h to 2203FFh.

(see Section 3.5.1 Hardware EEPROM Emulation for more details).

Table 7. Memory Structure for 128K Flash Device

Sector	Addresses	Max Size
TestFlash (TF) (Reserved)	230000h to 231F7Fh	8064 bytes
OTP Area Protection Registers (reserved)	231F80h to 231FFBh 231FFCh to 231FFFh	124 bytes 4 bytes
Flash 0 (F0)	000000h to 00FFFFh	64 Kbytes
Flash 1 (F1)	010000h to 01BFFFh	48 Kbytes
Flash 2 (F2)	01C000h to 01DFFFh	8 Kbytes
Flash 3 (F3)	01E000h to 01FFFFh	8 Kbytes
EEPROM 0 (E0)	228000h to 228FFFh	4 Kbytes
EEPROM 1 (E1)	22C000h to 22CFFFh	4 Kbytes
Emulated EEPROM	220000h to 2203FFh	1 Kbyte

Table 8. Memory Structure for 60K Flash Device

Sector	Addresses	Max Size
TestFlash (TF) (Reserved)	230000h to 231F7Fh	8064 bytes
OTP Area Protection Registers (reserved)	231F80h to 231FFBh 231FFCh to 231FFFh	124 bytes 4 bytes
Flash 0 (F0)	000000h to 000FFFh	4 Kbytes
Reserved	001000h to 00FFFFh	60 Kbytes
Flash 1 (F1)	010000h to 01BFFFh	48 Kbytes
Flash 2 (F2)	01C000h to 01DFFFh	8 Kbytes
EEPROM 0 (E0)	228000h to 228FFFh	4 Kbytes
EEPROM 1 (E1)	22C000h to 22CFFFh	4 Kbytes
Emulated EEPROM	220000h to 2203FFh	1Kbyte

FUNCTIONAL DESCRIPTION (Cont'd)

3.2.3 Operation

The memory has a register interface mapped in memory space (segment 22h). All operations are enabled through the FCR (Flash Control Register) ECR (EEPROM Control Register).

All operations on the Flash must be executed from another memory (internal RAM, EEPROM, external memory).

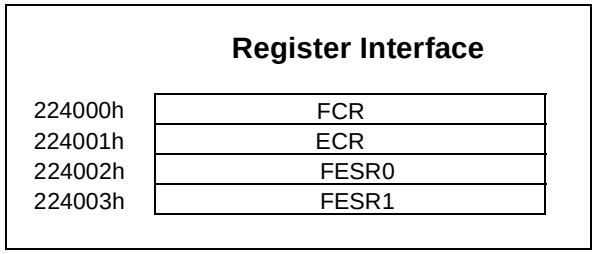
Flash (including TestFlash) and EEPROM have duplicated sense amplifiers, so that one can be read while the other is written. However simultaneous Flash and EEPROM write operations are forbidden.

An interrupt can be generated at the end of a Flash or an EEPROM write operation: this interrupt is multiplexed with an external interrupt EXTINTx (device dependent) to generate an interrupt INTx.

The status of a write operation inside the Flash and the EEPROM memories can be monitored through the FESR[1:0] registers.

Control and Status registers are mapped in memory (segment 22h), as shown in the following figure.

Figure 22. Control and Status Register Map



During a write operation, if the power supply drops or the RESET pin is activated, the write operation is immediately interrupted. In this case the user must repeat the last write operation following power on or reset.

3.3 REGISTER DESCRIPTION

3.3.1 Control Registers

FLASH CONTROL REGISTER (FCR)

Address: 224000h - Read/Write

Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
FWMS	FPAGE	FCHIP	FBYTE	FSECT	FSUSP	PROT	FBUSY

The Flash Control Register is used to enable all the operations for the Flash and the TestFlash memories. The write access to the TestFlash is possible only in Test mode, except the OTP area of the TestFlash that can be programmed in user mode (but not erased).

Bit 7 = **FWMS:** *Flash Write Mode Start (Read/ Write).*
This bit must be set to start every write/erase operation in Flash memory. At the end of the write/ erase operation or during a Sector Erase Suspend this bit is automatically reset. To resume a suspended Sector Erase operation, this bit must be set again. Resetting this bit by software does not stop the current write operation.
0: No effect
1: Start Flash write

Bit 6 = **FPAGE**: *Flash Page program (Read/Write).*
This bit must be set to select the Page Program operation in Flash memory. The Page Program operation allows to program “0”s in place of “1”s. From 1 to 16 bytes can be entered (in any order, no need for an ordered address sequence) before starting the execution by setting the FWMS bit . All the addresses must belong to the same page (only the 4 LSBs of address can change). Data to be programmed and addresses in which to program must be provided (through an LD instruction, for example). Data contained in page addresses that are not entered are left unchanged. This bit is automatically reset at the end of the Page Program operation.
0: Deselect page program
1: Select page program

Bit 5 = **FCHIP:** *Flash CHIP erase (Read/Write).*
This bit must be set to select the Chip Erase operation in Flash memory. The Chip Erase operation allows to erase all the Flash locations to FFh. The operation is limited to Flash code (sectors F0-F3; TestFlash and EEPROM sectors excluded). The execution starts by setting the FWMS bit. It is not necessary to pre-program the sectors to 00h, because this is done automatically. This bit is automatically reset at the end of the Chip Erase operation.
0: Deselect chip erase
1: Select chip erase

Bit 4 = **FBYTE**: *Flash byte program (Read/Write).*
This bit must be set to select the Byte Program operation in Flash memory. The Byte Program operation allows “0”s to be programmedin place of “1”s. Data to be programmed and an address in which to program must be provided (through an LD instruction, for example) before starting execution by setting bit FWMS. This bit is automatically reset at the end of the Byte Program operation.
0: Deselect byte program
1: Select byte program

Bit 3 = **FSECT:** *Flash sector erase (Read/Write).*
This bit must be set to select the Sector Erase operation in Flash memory. The Sector Erase operation erases all the Flash locations to FFh. From 1 to 4 sectors (F0, ..,F3) can be simultaneously erased, while TF must be individually erased. Sectors to be simultaneously erased can be entered before starting the execution by setting the FWMS bit. An address located in the sector to erase must be provided (through an LD instruction, for example), while the data to be provided is don’t care. It is not necessary to pre-program the sectors to 00h, because this is done automatically. This bit is automatically reset at the end of the Sector Erase operation.
0: Deselect sector erase
1: Select sector erase

Bit 2 = **FSUSP**: *Flash sector erase suspend (Read/Write).*
This bit must be set to suspend the current Sector Erase operation in Flash memory in order to read data to or from program data to a sector not being erased. The Erase Suspend operation resets the Flash memory to normal read mode (automatically resetting bit FBUSY) in a maximum time of 15µs.

REGISTER DESCRIPTION (Cont'd)

When in Erase Suspend the memory accepts only the following operations: Read, Erase Resume and Byte Program. Updating the EEPROM memory is not possible during a Flash Erase Suspend.

The FSUSP bit must be reset (and FWMS must be set again) to resume a suspended Sector Erase operation.
0: Resume sector erase when FWMS is set again.
1: Suspend Sector erase

Bit 1 = **PROT**: *Set Protection (Read/Write).*
This bit must be set to select the Set Protection operation. The Set Protection operation allows "0"s in place of "1"s to be programmed in the four Non Volatile Protection registers. From 1 to 4 bytes can be entered (in any order, no need for an ordered address sequence) before starting the execution by setting the FWMS bit . Data to be programmed and addresses in which to program must be provided (through an LD instruction, for example). Protection contained in addresses that are not entered are left unchanged. This bit is automatically reset at the end of the Set Protection operation.
0: Deselect protection
1: Select protection

Bit 0 = **FBUSY**: *Flash Busy (Read Only).*
This bit is automatically set during Page Program, Byte Program, Sector Erase or Set Protection operations when the first address to be modified is latched in Flash memory, or during Chip Erase operation when bit FWMS is set. When this bit is set every read access to the Flash memory will output invalid data (FFh equivalent to a NOP instruction), while every write access to the Flash memory will be ignored. At the end of the write operations or during a Sector Erase Suspend, this bit is automatically reset and the memory returns to read mode. After an Erase Resume this bit is automatically set again. The FBUSY bit remains high for a maximum of 10 µs after Power-Up and when exiting Power-Down mode, meaning that the Flash memory is not yet ready to be accessed.
0: Flash not busy
1: Flash busy

EEPROM CONTROL REGISTER (ECR)

Address: 224001h - Read/Write

Reset value: 000x x000 (xxh)

7	6	5	4	3	2	1	0
EWMS	EPAGE	ECHIP			WFIS	FEIEN	EBUSY

The EEPROM Control Register is used to enable all the operations for the EEPROM memory in devices with EEPROM hardware emulation.

The ECR also contains two bits (WFIS and FEIEN) that are related to both Flash and EEPROM memories.

Bit 7 = **EWMS**: *EEPROM Write Mode Start*.
This bit must be set to start every write/erase operation in the EEPROM memory. At the end of the write/erase operation this bit is automatically reset. Resetting by software this bit does not stop the current write operation.
0: No effect
1: Start EEPROM write

Bit 6 = **EPAGE**: *EEPROM page update.*
This bit must be set to select the Page Update operation in EEPROM memory. The Page Update operation allows to write a new content: both "0"s in place of "1"s and "1"s in place of "0"s. From 1 to 16 bytes can be entered (in any order, no need for an ordered address sequence) before starting the execution by setting bit EWMS. All the addresses must belong to the same page (only the 4 LSBs of address can change). Data to be programmed and addresses in which to program must be provided (through an LD instruction, for example). Data contained in page addresses that are not entered are left unchanged. This bit is automatically reset at the end of the Page Update operation.
0: Deselect page update
1: Select page update

Bit 5 = **ECHIP**: *EEPROM chip erase.*
This bit must be set to select the Chip Erase operation in the EEPROM memory. The Chip Erase operation allows to erase all the EEPROM locations to (E0 and E1 sectors) FFh. The execution starts by setting bit EWMS. This bit is automatically reset at the end of the Chip Erase operation.
0: Deselect chip erase
1: Select chip erase

Bit 4:3 = Reserved.

REGISTER DESCRIPTION (Cont'd)

Bit 2 = **WFIS**: *Wait For Interrupt Status.*
If this bit is reset, the WFI instruction puts the Flash macrocell in Stand-by mode (immediate read possible, but higher consumption: 100 µA); if it is set, the WFI instruction puts the Flash macrocell in Power-Down mode (recovery time of 10µs needed before reading, but lower consumption: 10µA). The Stand-by mode or the Power-Down mode will be entered only at the end of any current Flash or EEPROM write operation.

In the same way following an HALT or a STOP instruction, the Memory enters Power-Down mode only after the completion of any current write operation.
0: Flash in Standby mode on WFI
1: Flash in Power-Down mode on WFI

Note: HALT or STOP mode can be exited without problems, but the user should take care when exiting WFI Power Down mode. If WFIS is set, the user code must reset the XT_DIV16 bit in the R242 register (page 55) before executing the WFI instruction. When exiting WFI mode, this gives the Flash enough time to wake up before the interrupt vector fetch.

Bit 1 = **FEIEN**: *Flash & EEPROM Interrupt enable*.
This bit selects the source of interrupt channel INTx between the external interrupt pin and the Flash/EEPROM End of Write interrupt. Refer to the Interrupt chapter for the channel number.
0: External interrupt enabled
1: Flash & EEPROM Interrupt enabled

Bit 0 = **EBUSY:** *EEPROM Busy (Read Only).*
This bit is automatically set during a Page Update operation when the first address to be modified is latched in the EEPROM memory, or during Chip Erase operation when bit EWMS is set. At the end of the write operation or during a Sector Erase Suspend this bit is automatically reset and the memory returns to read mode. When this bit is set every read access to the EEPROM memory will output invalid data (FFh equivalent to a NOP instruction), while every write access to the EEPROM memory will be ignored. At the end of the write operation this bit is automatically reset and the memory returns to read mode. Bit EBUSY remains high for a maximum of 10ms after Power-Up and when exiting Power-Down mode, meaning that the EEPROM memory is not yet ready to be accessed.
0: EEPROM not busy
1: EEPROM busy

REGISTER DESCRIPTION (Cont'd)

3.3.2 Status Registers

During a Flash or an EEPROM write operation any attempt to read the memory under modification will output invalid data (FFh equivalent to a NOP instruction). This means that the Flash memory is not fetchable when a write operation is active: the write operation commands must be given from another memory (EEPROM, internal RAM, or external memory).

Two Status Registers (FESR[1:0] are available to check the status of the current write operation in Flash and EEPROM memories.

FLASH & EEPROM STATUS REGISTER 0 (FESR0)

Address: 224002h -Read/Write

Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
FEERR	FESS6	FESS5	FESS4	FESS3	FESS2	FESS1	FESS0

Bit 7 = **FEERR**: *Flash or EEPROM write ERRor (Read/Write).*
This bit is set by hardware when an error occurs during a Flash or an EEPROM write operation. It must be cleared by software.
0: Write OK
1: Flash or EEPROM write error

Bits 6:0 = **FESS[6:0]**. *Flash and EEPROM Status Sector 6-0 (Read Only).*
These bits are set by hardware and give the status of the 7 Flash and EEPROM sectors (TF, E1, E0, F3, F2, F1, F0). The meaning of FESSx bit for sector x is given by the following table:

Table 9. FESSx bit Values

FEERR	FBUSY EBUSY	FSUSP	FESSx=1 meaning
1	-	-	Write Error in Sector x
0	1	-	Write operation on-going in sector x
0	0	1	Sector Erase Suspended in sector x
0	0	0	Don't care

FLASH & EEPROM STATUS REGISTER 1 (FESR1)

Address: 224003h -Read Only

Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
ERER	PGER	SWER					

Bit 7 = **ERER**. *Erase error (Read Only).*
This bit is set by hardware when an Erase error occurs during a Flash or an EEPROM write operation. This error is due to a real failure of a Flash cell, that can not be erased anymore. This kind of error is fatal and the sector where it occurred must be discarded (if it was in one of the EEPROM sectors, the hardware emulation can not be used anymore). This bit is automatically cleared when bit FEERR of the FESR0 register is cleared by software.
0: Erase OK
1: Erase error

Bit 6 = **PGER**. *Program error (Read Only).*
This bit is automatically set when a Program error occurs during a Flash or an EEPROM write operation. This error is due to a real failure of a Flash cell, that can not be programmed anymore. The byte where this error occurred must be discarded (if it was in the EEPROM memory, the byte must be reprogrammed to FFh and then discarded, to avoid the error occurring again when that byte is internally moved). This bit is automatically cleared when bit FEERR of the FESR0 register is cleared by software.
0: Program OK
1: Flash or EEPROM Programming error

REGISTER DESCRIPTION (Cont'd)

Bit 5 = **SWER**. *Swap or 1 over 0 Error (Read Only).*
This bit has two different meanings, depending on whether the current write operation is to Flash or EEPROM memory.

In Flash memory, this bit is automatically set when trying to program at 1 bits previously set at 0 (this does not happen when programming the Protection bits). This error is not due to a failure of the Flash cell, but only flags that the desired data has not been written.

In the EEPROM memory, this bit is automatically set when a Program error occurs during the swapping of the unselected pages to the new sector when the old sector is full (see Section 3.5.1 Hardware EEPROM Emulation for more details).

This error is due to a real failure of a Flash cell, that can not be programmed anymore. When this error is detected, the embedded algorithm automatically exits the Page Update operation at the end of the Swap phase, without performing the Erase Phase 0 on the full sector. In this way the old data are kept, and through predefined routines in TestFlash (Find Wrong Pages = 230029h and Find Wrong Bytes = 23002Ch), the user can compare the old and the new data to find where the error occurred.

Once the error has been discovered the user must take to end the stopped Erase Phase 0 on the old sector (through another predefined routine in TestFlash: Complete Swap = 23002Fh). The byte where the error occurred must be reprogrammed to FFh and then discarded, to avoid the error occurring again when that byte is internally moved.

This bit is automatically cleared when bit FEERR of the FESR0 register is cleared by software.

Bits 4:0 = Reserved.

3.4 WRITE OPERATION EXAMPLE

Each operation (both Flash and EEPROM) is activated by a sequence of instructions like the following:

```
OR    CR, #OPMASK  ;Operation selection
LD    ADD1, #DATA1 ;1st Add and Data
LD    ADD2, #DATA2 ;2nd Add and Data
..    ...., ......
LD    ADDn, #DATAn ;nth Add and Data
                   ;n range = (1 to 16)
OR    CR, #80h     ;Operation start
```

The first instruction is used to select the desired operation by setting its corresponding selection bit in the Control Register (FCR for Flash operations, ECR for EEPROM operations).

The load instructions are used to set the addresses (in the Flash or in the EEPROM memory space) and the data to be modified.

The last instruction is used to start the write operation, by setting the start bit (FWMS for Flash operations, EWMS for EEPROM operation) in the Control register.

Once selected, but not yet started, one operation can be cancelled by resetting the operation selection bit. Any latched address and data will be reset.

Caution: during the Flash Page Program or the EEPROM Page Update operation it is forbidden to change the page address: only the last page address is effectively kept and all programming will effect only that page.

A summary of the available Flash and EEPROM write operations are shown in the following tables:

Table 10. Flash Write Operations

Operation	Selection bit	Addresses and Data	Start bit	Typical Duration
Byte Program	FBYTE	1 byte	FWMS	10 µs
Page Program	FPAGE	From 1 to 16 bytes	FWMS	160 µs (16 bytes)
Sector Erase	FSECT	From 1 to 4 sectors	FWMS	1.5 s (1 sector)
Sector Erase Suspend	FSUSP	None	None	15 µs
Chip Erase	FCHIP	None	FWMS	3 s
Set Protection	PROT	From 1 to 4 bytes	FWMS	40 µs (4 bytes)

Table 11. EEPROM Write Operations

Operation	Selection bit	Addresses and Data	Start bit	Typical Duration
Page Update	EPAGE	From 1 to 16 bytes	EWMS	30 ms
Chip Erase	ECHIP	None	EWMS	70 ms

3.5 EEPROM

3.5.1 Hardware EEPROM Emulation

Note: This section provides general information only. Users do not have to be concerned with the hardware EEPROM emulation.

The last 256 bytes of the two EEPROM dedicated sectors (229000h to 2290FFh for sector E0 and 22D000h to 22D0FFh for sector E1) are reserved for the Non Volatile pointers used for the hardware Emulation.

When the EEPROM is directly addressed through the addresses 220000h to 2203FFh, a Hardware Emulation mechanism is automatically activated, so avoiding the user having to manage the Non Volatile pointers that are used to map the EEPROM inside the two dedicated Flash sectors E0 and E1.

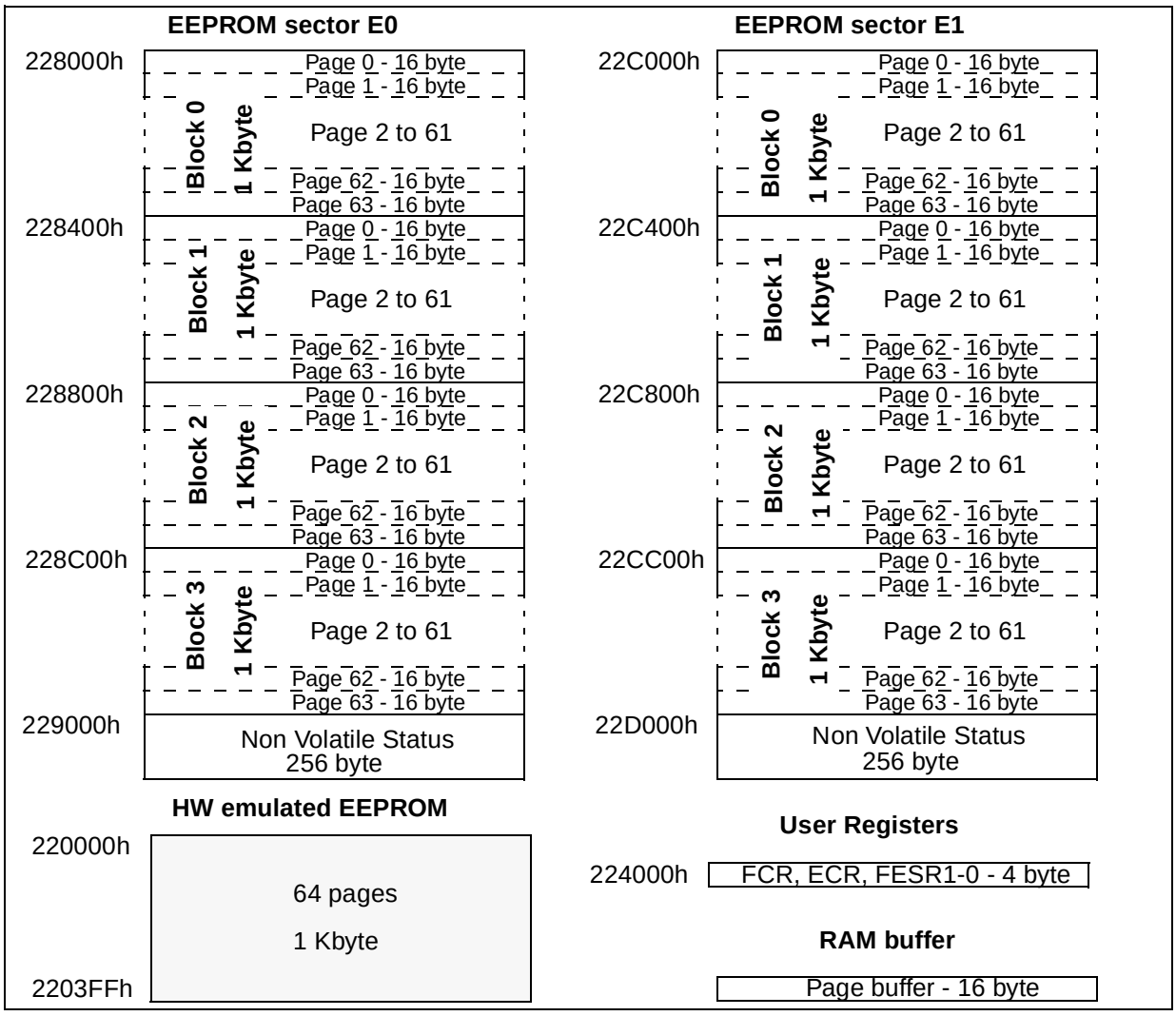
The structure of the hardware emulation is shown in Figure 23.

Each one of the two EEPROM dedicated Flash sectors E0 and E1 is divided in 4 blocks of the same size of the EEPROM to emulate (1Kbyte max).

Each one of the 4 blocks is then divided in up to 64 pages of 16 bytes, the size of the available RAM buffer.

The RAM buffer is used internally to temporarily store the new content of the page to update, during the Page Program operation (both in Flash and in EEPROM).

Figure 23. Segment 22h Structure (Example for 128K Flash device)



EEPROM (Cont'd)

3.5.2 EEPROM Update Operation

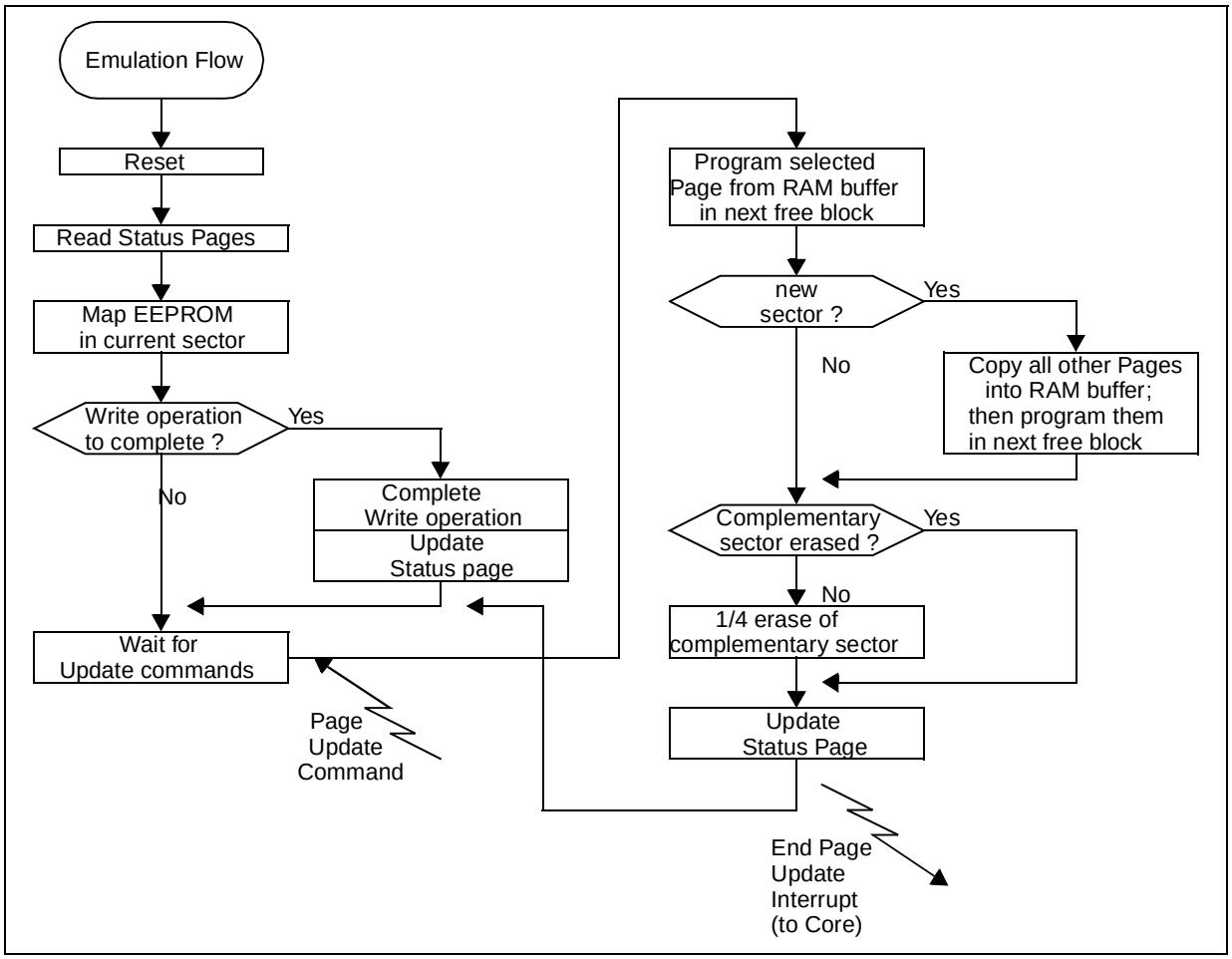
The update of the EEPROM content can be made by pages of 16 consecutive bytes. The Page Update operation allows up to 16 bytes to be loaded into the RAM buffer that replace the ones already contained in the specified address.

Each time a Page Update operation is executed in the EEPROM, the RAM buffer content is programmed in the next free block relative to the specified page (the RAM buffer is previously automatically filled with old data for all the page addresses not selected for updating). If all the 4 blocks of the specified page in the current EEPROM sector are full, the page content is copied to the complementary sector, that becomes the new current one.

After that the specified page has been copied to the next free block, one erase phase is executed on the complementary sector, if the 4 erase phases have not yet been executed. When the selected page is copied to the complementary sector, the remaining 63 pages are also copied to the first block of the new sector; then the first erase phase is executed on the previous full sector. All this is executed in a hidden manner, and the End Page Update Interrupt is generated only after the end of the complete operation.

At Reset the two status pages are read in order to detect which is the sector that is currently mapping the EEPROM, and in which block each page is mapped. A system defined routine written in TestFlash is executed at reset, so that any previously aborted write operation is restarted and completed.

Figure 24. Hardware Emulation Flow



3.6 PROTECTION STRATEGY

The protection bits are stored in the last 4 locations of the TestFlash (from 231FFCh) (see Figure 25).

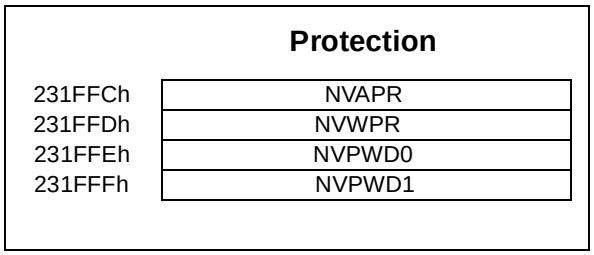
All the available protections are forced active during reset, then in the initialisation phase they are read from the TestFlash.

The protections are stored in 2 Non Volatile Registers. Other 2 Non Volatile Registers can be used as a password to re-enable test modes once they have been disabled.

The protections can be programmed using the Set Protection operation (see Control Registers paragraph), that can be executed from all the internal or external memories except the Flash or TestFlash itself.

The TestFlash area (230000h to 231F7Fh) is always protected against write access.

Figure 25. Protection Map



3.6.1 Non Volatile Registers

The 4 Non Volatile Registers used to store the protection bits for the different protection features are one time programmable by the user, but they are erasable in Test mode (if not disabled).

Access to these registers is controlled by the protections related to the TestFlash where they are mapped. Since the code to program the Protection Registers cannot be fetched by the Flash or the TestFlash memories, this means that, once the APRO or APBR bits in the NVAPR register are programmed, it is no longer possible to modify any of the protection bits. For this reason the NV Password, if needed, must be set with the same Set Protection operation used to program these bits. For the same reason it is strongly advised to never program the WPBR bit in the NVWPR register, as this will prevent any further write access to the TestFlash, and consequently to the Protection Registers.

NON VOLATILE ACCESS PROTECTION REGISTER (NVAPR)

Address: 231FFCh - Read/Write

Delivery value: 1111 1111 (FFh)

7	6	5	4	3	2	1	0
1	APRO	APBR	APEE	APEX	PWT2	PWT1	PWT0

Bit 7 = Reserved**.**

Bit 6 = **APRO**: *Flash Access Protection.*
This bit, if programmed at 0, disables any access (read/write) to operands mapped inside the Flash address space (EEPROM excluded), unless the current instruction is fetched from the TestFlash or from the Flash itself.
0: Flash protection on
1: Flash protection off

Bit 5 = **APBR**: *TestFlash Access Protection.*
This bit, if programmed at 0, disables any access (read/write) to operands mapped inside the TestFlash address space, unless the current instruction is fetched from the TestFlash itself.
0: TestFlash protection on
1: TestFlash protection off

Bit 4 = **APEE**: *EEPROM Access Protection.*

This bit, if programmed at 0, disables any access (read/write) to operands mapped inside the EEPROM address space, unless the current instruction is fetched from the TestFlash or from the Flash, or from the EEPROM itself.
0: EEPROM protection on
1: EEPROM protection off

Bit 3 = **APEX**: *Access Protection from External Memory.*
This bit, if programmed at 0, disables any access (read/write) to operands mapped inside the address space of one of the internal memories (TestFlash, Flash, EEPROM, RAM), if the current instruction is fetched from an external memory.

PROTECTION STRATEGY (Cont'd)

Bits 2:0 = **PWT[2:0]:** *Password Attempt 2-0.*

If the TMDIS bit in the NVWPR register (231FFDh) is programmed to 0, every time a Set Protection operation is executed with Program Addresses equal to NVPWD1-0 (231FFE-Fh), the two provided Program Data are compared with the NVPWD1-0 content; if there is not a match one of PWT2-0 bits is automatically programmed to 0: when these three bits are all programmed to 0 the test modes are disabled forever. In order to intentionally disable test modes forever, it is sufficient to set a random Password and then to make 3 wrong attempts to enter it.

NON VOLATILE WRITE PROTECTION REGISTER (NVWPR)

Address: 231FFDh - Read/Write

Delivery value: 1111 1111 (FFh)

7	6	5	4	3	2	1	0
TMDIS	PWOK	WPBR	WPEE	WPRS3	WPRS2	WPRS1	WPRS0

Bit 7 = **TMDIS**: *Test mode disable (Read Only).*
This bit, if set to 1, allows to bypass all the protections in test and EPB modes. If programmed to 0, on the contrary, all the protections remain active also in test mode. The only way to enable the test modes if this bit is programmed to 0, is to execute the Set Protection operation with Program Addresses equal to NVPWD1-0 (231FFF-Eh) and Program Data matching with the content of NVPWD1-0. This bit is read only: it is automatically programmed to 0 when NVPWD1-0 are written for the first time.
0: Test mode disabled
1: Test mode enabled

Bit 6 = **PWOK**: *Password OK (Read Only).*
If the TMDIS bit is programmed to 0, when the Set Protection operation is executed with Program Addresses equal to NVPWD[1:0] and Program Data matching with NVPWD[1:0] content, the PWOK bit is automatically programmed to 0. When this bit is programmed to 0 TMDIS protection is bypassed and the test and EPB modes are enabled.
0: Password OK
1: Password not OK

Bit 5 = **WPBR:** *TestFlash Write Protection.*
This bit, if programmed at 0, disables any write access to the TestFlash address space. This protection cannot be temporarily disabled.
0: TestFlash write protection on
1: TestFlash write protection off

Note: it is strongly advised to never program the WPBR bit in the NVWPR register, as this will prevent any further write access to the protection registers.

Bit 4 = **WPEE**: *EEPROM Write Protection.*
This bit, if programmed to 0, disables any write access to the EEPROM address space. This protection can be temporary disabled by executing the Set Protection operation and writing 1 into this bit. To restore the protection it needs to reset the micro or to execute another Set Protection operation and write 0 to this bit.
0: EEPROM write protection on
1: EEPROM write protection off

Bits 3:0 = **WPRS[3:0]:** *ROM Segments 3-0 Write Protection.*

These bits, if programmed to 0, disable any write access to the 4 Flash sectors address spaces. These protections can be temporary disabled by executing the Set Protection operation and writing 1 into these bits. To restore the protection it needs to reset the micro or to execute another Set Protection operation and write 0 into these bits.
0: ROM Segments 3-0 write protection on
1: ROM Segments 3-0 write protection off

NON VOLATILE PASSWORD (NVPWD1-0)

Address: 231FFF-231FFEh - Write Only

Delivery value: 1111 1111 (FFh)

7	6	5	4	3	2	1	0
PWD7	PWD6	PWD5	PWD4	PWD3	PWD2	PWD1	PWD0

Bits 7:0 = **PWD[7:0]:** *Password bits 7:0 (Write Only).*
These bits must be programmed with the Non Volatile Password that must be provided with the Set Protection operation to disable (first write access) or to reenable (second write access) the test and EPB modes. The first write access fixes the password value and resets the TMDIS bit of NVWPR

(231FFDh). The second write access, with Program Data matching with NVPWD[1:0] content, resets the PWOK bit of NVWPR.

These two registers can be accessed only in write mode (a read access returns FFh).

3.6.2 Temporary Unprotection

On user request the memory can be configured so as to allow the temporary unprotection also of all access protections bits of NVAPR (write protection bits of NVWPR are always temporarily unprotectable).

Bit APEX can be temporarily disabled by executing the Set Protection operation and writing 1 into this bit, but only if this write instruction is executed from an internal memory (Flash and Test Flash excluded).

Bit APEE can be temporarily disabled by executing the Set Protection operation and writing 1 into this bit, but only if this write instruction is executed from the memory itself to unprotect (EEPROM).

Bits APRO and APBR can be temporarily disabled through a direct write at NVAPR location, by overwriting at 1 these bits, but only if this write instruction is executed from the memory itself to unprotect.

To restore the access protection bits it needs to reset the micro or to execute a Set Protection operation and write 0 into the desired bits.

When an internal memory (Flash, TestFlash or EEPROM) is protected in access, also the data access through a DMA of a peripheral is forbidden (it returns FFh). To read data in DMA mode from a protected memory, first it is necessary to temporarily unprotect that memory.

The temporary unprotection allows also to update a protected code.

3.7 FLASH IN-SYSTEM PROGRAMMING

The Flash memory can be programmed in-system through a serial interface (SCI0).

Exiting from reset, the ST9 executes the initialization from the TestFlash code (written in TestFlash), where it checks the value of the SOUT0 pin. If it is at 0, this means that the user wishes to update the Flash code, otherwise normal execution continues. In this second case, the TestFlash code reads the reset vector.

If the Flash is virgin (read content is always FFh), its reset vector contains FFFFh. This is interpreted by the TestFlash code as a flag indicating that the Flash memory is virgin and needs to be programmed. If the value 1 is detected on the SOUT0 pin and the Flash is virgin, a HALT instruction is executed, waiting for a hardware Reset.

3.7.1 Code Update Routine

The TestFlash Code Update routine is called automatically if the SOUT0 pin is held low during power-on.

The Code Update routine performs the following operations:

- Enables the SCI0 peripheral in synchronous mode
- Transmits a synchronization datum (25h);
- Waits for an address match (23h) with a timeout of 10ms (@ f_{OSC} 4 MHz)
- If the match is not received before the timeout, the execution returns to the Power-On routine
- If the match is received, the SCI0 transmits a new datum (21h) to tell the external device that it is ready to receive the data to be loaded in RAM (that represents the code of the in-system programming routine).
- Receives two data representing the number of bytes to be loaded (max. 4 Kbytes)
- Receive the specified number of bytes (each one preceded by the transmission of a Ready to Receive character: (21h) and writes them in internal RAM starting from address 200010h. The first 4 words should be the interrupt vectors of the 4 possible SCI interrupts, to be used by the in-system programming routine.
- Transmit a last datum (21h) as a request for end of communications.
- Receives the end of communication confirmation datum (any byte other than 25h).
- Resets all the unused RAM locations to FFh;
- Calls address 200018h in internal RAM.
- After completion of the in-system programming routine, an HALT instruction is executed and an Hardware Reset is needed.

The Code Update routine initializes the SCI0 peripheral as shown in the following table:

Table 12. SCI0 Registers (page 24) initialization

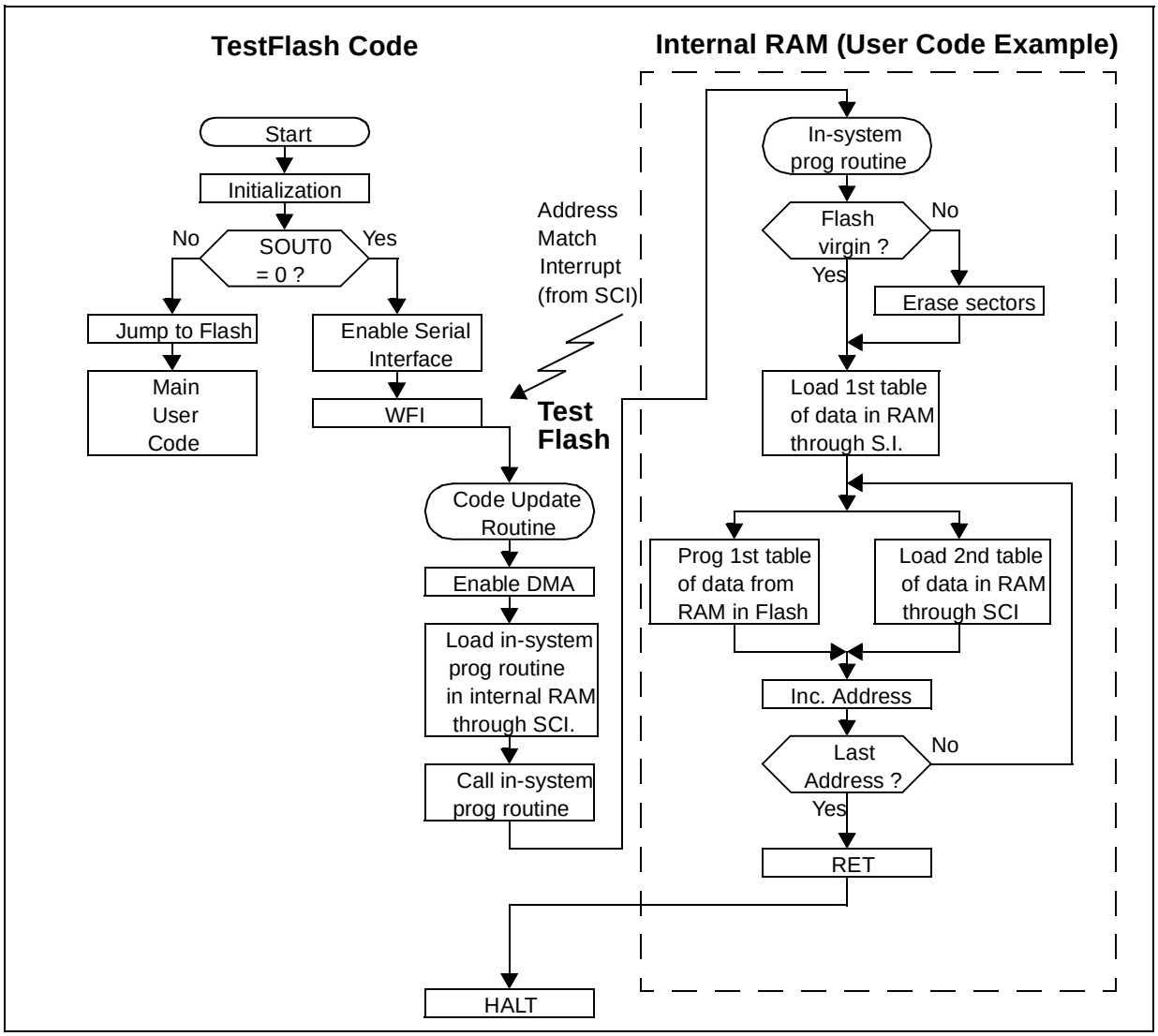
Register	Value	Notes
IVR - R244	10h	Vector Table in 0010h
ACR - R245	23h	Address Match is 23h
IDPR - R249	00h	SCI interrupt priority is 0
CHCR - R250	83h	8 Data Bits
CCR - R251	E8h	rec. clock: ext RXCLK0 trx clock: int CLKOUT0
BRGHR - R252	00h	

Register	Value	Notes
BRGLR - R253	04h	Baud Rate Divider is 4
SICR - R254	83h	Synchronous Mode
SOCR - R255	01h	

In addition, the Code Update routine remaps the interrupts in the TestFlash (ISR = 23h), and configures I/O Ports P5.3 (SOUT0) and and P5.4 (CLKOUT0) as Alternate Functions.

Note: Four interrupt routines are used by the code update routine: SCI Receiver Error Interrupt routine (vector in 0010h), SCI address Match Interrupt routine (vector in 0012h), SCI Receiver Data Ready Interrupt routine (vector in 0014h) and SCI Transmitter Buffer Empty Interrupt routine (vector in 0016h).

Figure 26. Flash in-system Programming



4 REGISTER AND MEMORY MAP

4.1 INTRODUCTION

The ST92F120 register map, memory map and peripheral options are documented in this section. Use this reference information to supplement the functional descriptions given elsewhere in this document.

4.2 MEMORY CONFIGURATION

The Program memory space of the ST92F120 up to 128K bytes of directly addressable on-chip memory, is fully available to the user.

4.2.1 Reset Vector Location

In 128k devices, the reset vector is located in 01E000h.

In 60k devices, the reset vector is located in 000000h.

4.2.2 Location of Vector for External Watchdog Refresh

If an external watchdog is used, it must be refreshed during TestFlash execution by a user written routine. This routine has to be located in Flash memory, the address where the routine starts has to be written in 000006h (one word) while the segement where the routine is located has to be written in 000009h (one byte).

This routine is called at least once every time that the TestFlash executes an EEPROM write operation. If the write operation has a long duration, the user routine is called with a rate fixed by location 000008h with an internal clock frequency of 2 MHz, location 000008h fixes the number of millseconds to wait between two calls of the user routine.

Table 13. User Routine Parameters

Location	Size	Description
000006h to 000007h	2 bytes	User routine address
000008h	1 byte	ms rate at 2 MHz.
000009h	1 byte	User routine segment

If location 000006h to 000007h is virgin (FFFFh), the user routine is not called.

Figure 27. ST92F120JV1Q7/ST92F120V1Q7 User Memory Map (part 1)

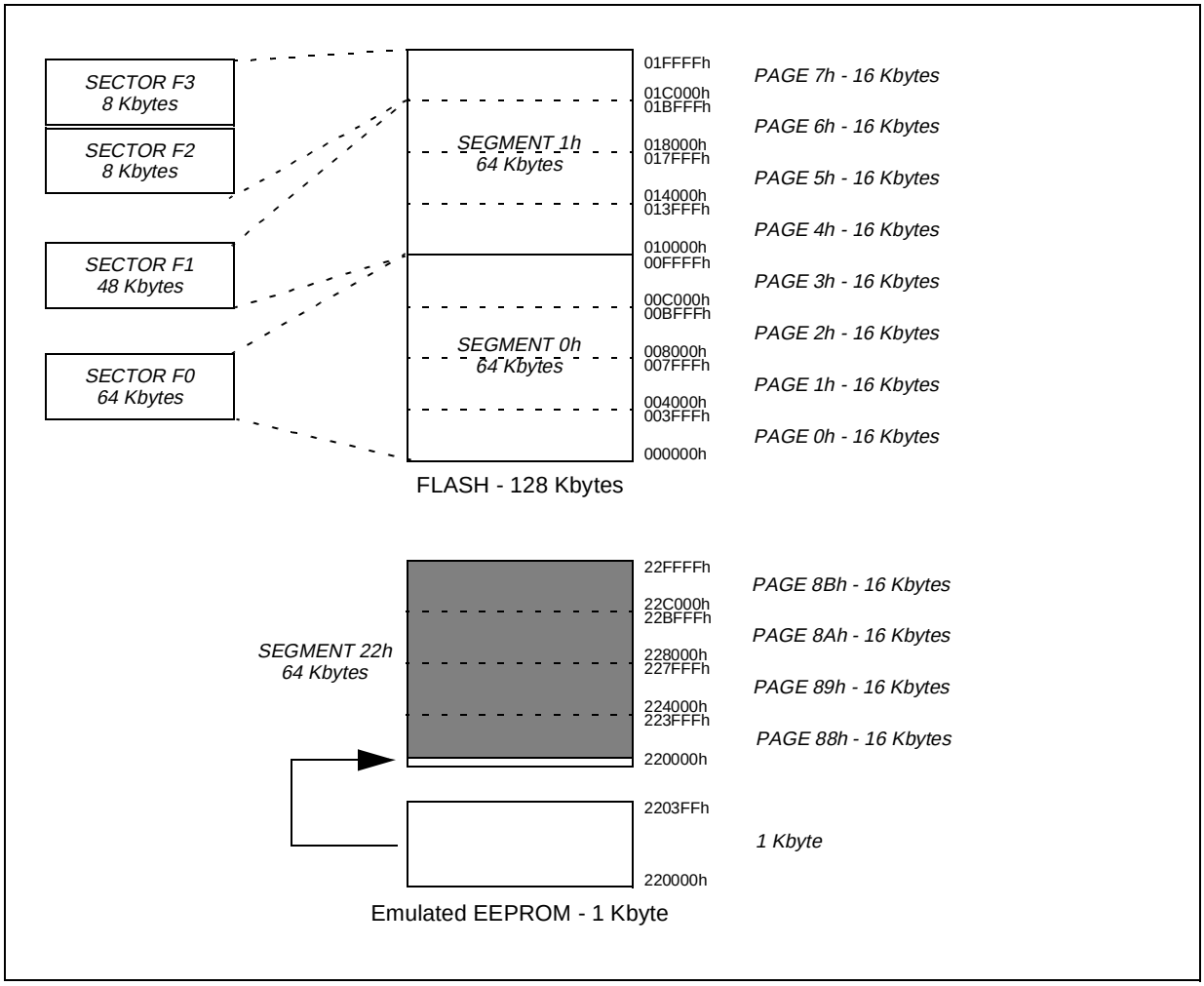


Figure 28. ST92F120JV1Q7/ST92F120V1Q7 User Memory Map (part 2)

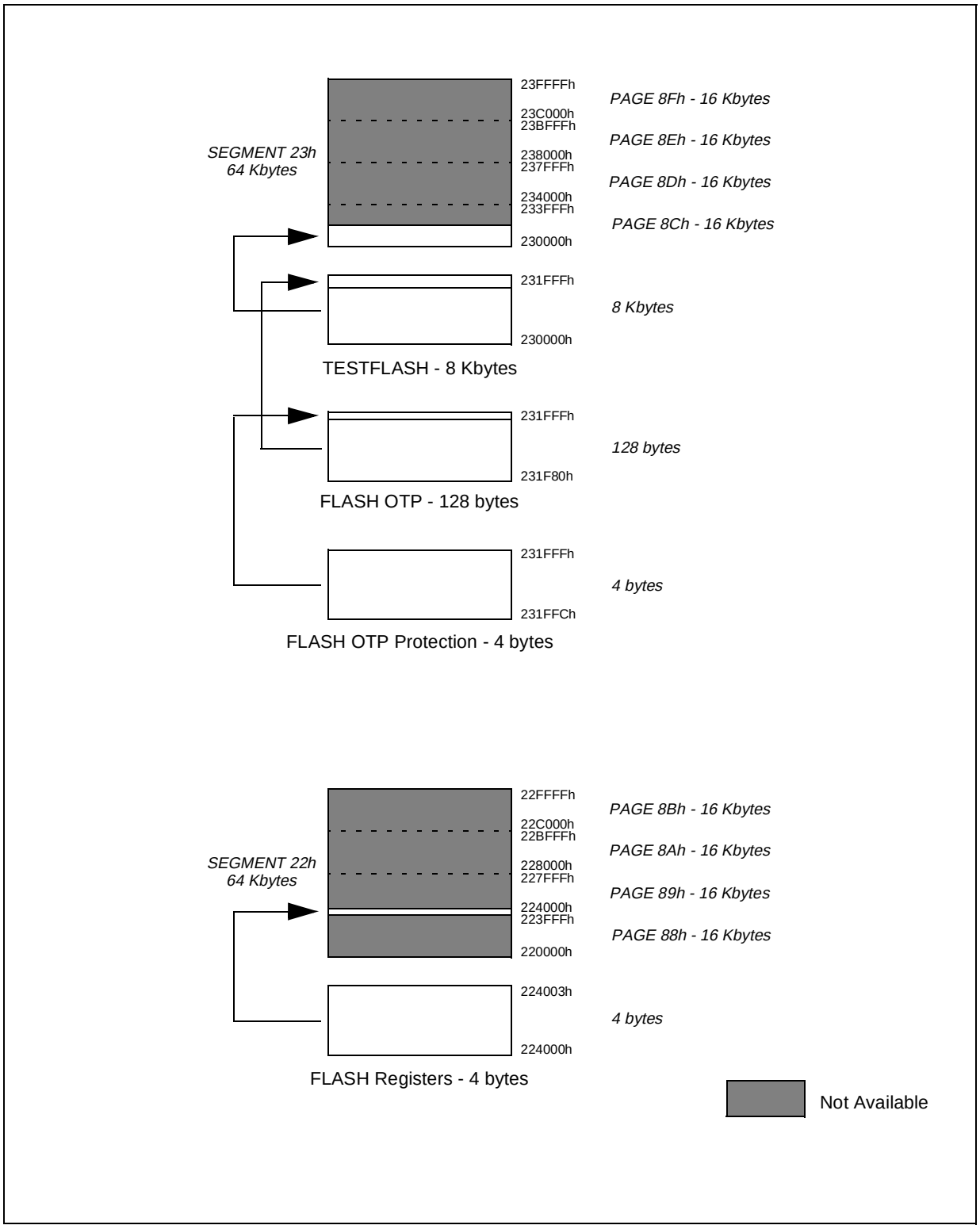
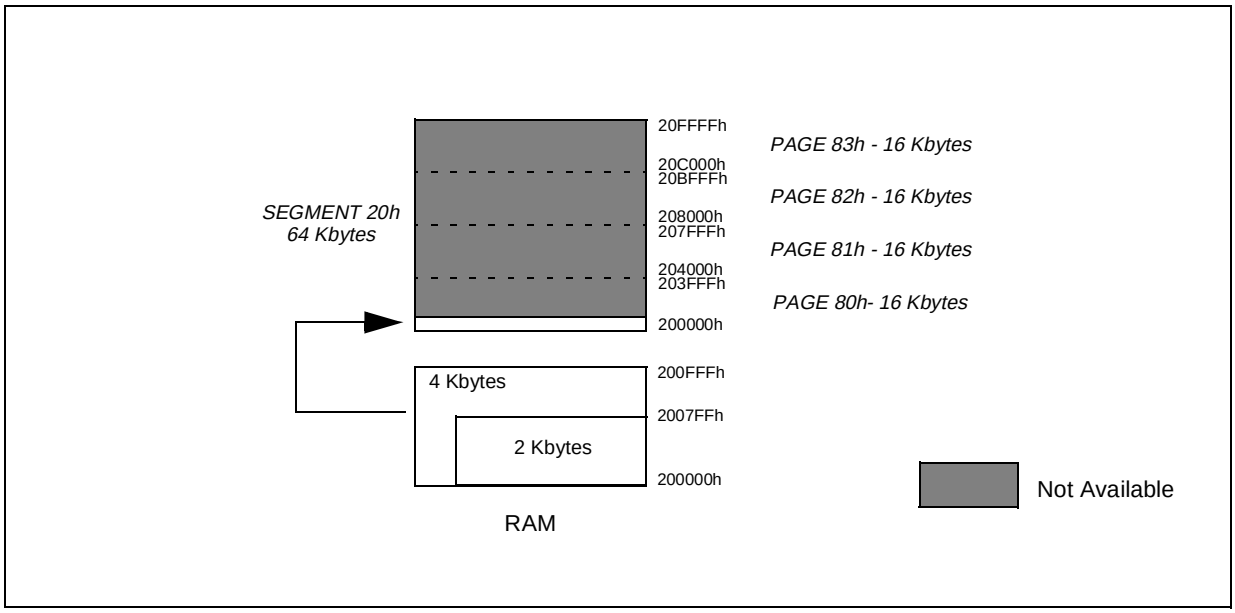


Figure 29. ST92F120 User Memory Map (part 3)



4.3 ST92F120 REGISTER MAP

Table 15 contains the map of the group F peripheral pages.

The common registers used by each peripheral are listed in Table 14.

Be very careful to correctly program both:

– The set of registers dedicated to a particular function or peripheral.

– Registers common to other functions.

– In particular, double-check that any registers with “undefined” reset values have been correctly initialised.

Caution: Note that in the **EIVR** and each **IVR** register, all bits are significant. Take care when defining base vector addresses that entries in the Interrupt Vector table do not overlap.

Table 14. Common Registers

Function or Peripheral	Common Registers
SCI, MFT	CICR + NICR + DMA REGISTERS + I/O PORT REGISTERS
A/D	CICR + NICR + I/O PORT REGISTERS
SPI, WDT, STIM	CICR + NICR + EXTERNAL INTERRUPT REGISTERS + I/O PORT REGISTERS
I/O PORTS	I/O PORT REGISTERS + MODER
EXTERNAL INTERRUPT	INTERRUPT REGISTERS + I/O PORT REGISTERS
RCCU	INTERRUPT REGISTERS + MODER

Table 15. Group F Pages Register Map

Resources available on the ST92F120 device:

Reg.	Page																			
	0	2	3	7	8	9	10	11	20	21	23	24	25	28	29	43	55	57	61	63
R255	Res.	Res	Port 7	Res.	MFT1	Res.	MFT0	Res.	I2C	MMU	JBLPD	SCI0	SCI1	EFT0	EFT1	Port 9	Res.	WUIMU	A/D 1	A/D 0
R254		Port 3																		
R253																				
R252	WCR																			
R251	WDT	Res	Port 6													Port 8				
R250		Port 2																		
R249																				
R248																		Res.		
R247	EXT INT	Res.	Res.			MFT1										Res.				
R246		Port 1	Port 5														RCCU			
R245																				
R244																				
R243		Res.	Res.	SPI		MFT0		STIM								Res.				
R242		Port 0	Port 4																	
R241	Res.																			
R240																				

Table 16. Detailed Register Map

Page (Dec)	Block	Reg. No.	Register Name	Description	Reset Value Hex.	Doc. Page
N/A	Core	R230	CICR	Central Interrupt Control Register	87	23
		R231	FLAGR	Flag Register	00	24
		R232	RP0	Pointer 0 Register	xx	26
		R233	RP1	Pointer 1 Register	xx	26
		R234	PPR	Page Pointer Register	xx	28
		R235	MODER	Mode Register	E0	28
		R236	USPHR	User Stack Pointer High Register	xx	30
		R237	USPLR	User Stack Pointer Low Register	xx	30
		R238	SSPHR	System Stack Pointer High Reg.	xx	30
		R239	SSPLR	System Stack Pointer Low Reg.	xx	30
	I/O Port 0:5	R224	P0DR	Port 0 Data Register	FF	118
		R225	P1DR	Port 1 Data Register	FF	
		R226	P2DR	Port 2 Data Register	FF	
		R227	P3DR	Port 3 Data Register	FF	
		R228	P4DR	Port 4 Data Register	FF	
		R229	P5DR	Port 5 Data Register	FF	
0	INT	R242	EITR	External Interrupt Trigger Register	00	78
		R243	EIPR	External Interrupt Pending Reg.	00	79
		R244	EIMR	External Interrupt Mask-bit Reg.	00	79
		R245	EIPLR	External Interrupt Priority Level Reg.	FF	79
		R246	EIVR	External Interrupt Vector Register	x6	130
		R247	NICR	Nested Interrupt Control	00	80
	WDT	R248	WDTHR	Watchdog Timer High Register	FF	129
		R249	WDTLR	Watchdog Timer Low Register	FF	129
		R250	WDTPR	Watchdog Timer Prescaler Reg.	FF	129
		R251	WDTCR	Watchdog Timer Control Register	12	129
		R252	WCR	Wait Control Register	7F	130
2	I/O Port 0	R240	P0C0	Port 0 Configuration Register 0	00	118
		R241	P0C1	Port 0 Configuration Register 1	00	
		R242	P0C2	Port 0 Configuration Register 2	00	
	I/O Port 1	R244	P1C0	Port 1 Configuration Register 0	00	
		R245	P1C1	Port 1 Configuration Register 1	00	
		R246	P1C2	Port 1 Configuration Register 2	00	
	I/O Port 2	R248	P2C0	Port 2 Configuration Register 0	FF	
		R249	P2C1	Port 2 Configuration Register 1	00	
		R250	P2C2	Port 2 Configuration Register 2	00	
	I/O Port 3	R252	P3C0	Port 3 Configuration Register 0	FE	
		R253	P3C1	Port 3 Configuration Register 1	00	
		R254	P3C2	Port 3 Configuration Register 2	00	

Page (Dec)	Block	Reg. No.	Register Name	Description	Reset Value Hex.	Doc. Page
3	I/O Port 4	R240	P4C0	Port 4 Configuration Register 0	FD	118
		R241	P4C1	Port 4 Configuration Register 1	00	
		R242	P4C2	Port 4 Configuration Register 2	00	
	I/O Port 5	R244	P5C0	Port 5 Configuration Register 0	FF	
		R245	P5C1	Port 5 Configuration Register 1	00	
		R246	P5C2	Port 5 Configuration Register 2	00	
	I/O Port 6	R248	P6C0	Port 6 Configuration Register 0	3F	
		R249	P6C1	Port 6 Configuration Register 1	00	
		R250	P6C2	Port 6 Configuration Register 2	00	
		R251	P6DR	Port 6 Data Register	FF	
	I/O Port 7	R252	P7C0	Port 7 Configuration Register 0	FF	
		R253	P7C1	Port 7 Configuration Register 1	00	
		R254	P7C2	Port 7 Configuration Register 2	00	
		R255	P7DR	Port 7 Data Register	FF	
7	SPI	R240	SPDR0	SPI0 Data Register	00	214
		R241	SPCR0	SPI0 Control Register	00	214
		R242	SPSR0	SPI0 Status Register	00	215
		R243	SPPR0	SPI0 Prescaler Register	00	215

Page (Dec)	Block	Reg. No.	Register Name	Description	Reset Value Hex.	Doc. Page
8	MFT1	R240	REG0HR1	Capture Load Register 0 High	xx	169
		R241	REG0LR1	Capture Load Register 0 Low	xx	169
		R242	REG1HR1	Capture Load Register 1 High	xx	169
		R243	REG1LR1	Capture Load Register 1 Low	xx	169
		R244	CMP0HR1	Compare 0 Register High	00	169
		R245	CMP0LR1	Compare 0 Register Low	00	169
		R246	CMP1HR1	Compare 1 Register High	00	169
		R247	CMP1LR1	Compare 1 Register Low	00	169
		R248	TCR1	Timer Control Register	00	170
		R249	TMR1	Timer Mode Register	00	171
		R250	T_ICR1	External Input Control Register	00	172
		R251	PRSR1	Prescaler Register	00	172
		R252	OACR1	Output A Control Register	00	173
		R253	OBCR1	Output B Control Register	00	174
		R254	T_FLAGR1	Flags Register	00	174
		R255	IDMR1	Interrupt/DMA Mask Register	00	176
9		R244	DCPR1	DMA Counter Pointer Register	xx	169
		R245	DAPR1	DMA Address Pointer Register	xx	169
		R246	T_IVR1	Interrupt Vector Register	xx	169
		R247	IDCR1	Interrupt/DMA Control Register	C7	169
	MFT0,1	R248	IOCR	I/O Connection Register	FC	178
	MFT0	R240	DCPR0	DMA Counter Pointer Register	xx	176
		R241	DAPR0	DMA Address Pointer Register	xx	177
		R242	T_IVR0	Interrupt Vector Register	xx	177
		R243	IDCR0	Interrupt/DMA Control Register	C7	178
10		R240	REG0HR0	Capture Load Register 0 High	xx	169
		R241	REG0LR0	Capture Load Register 0 Low	xx	169
		R242	REG1HR0	Capture Load Register 1 High	xx	169
		R243	REG1LR0	Capture Load Register 1 Low	xx	169
		R244	CMP0HR0	Compare 0 Register High	00	169
		R245	CMP0LR0	Compare 0 Register Low	00	169
		R246	CMP1HR0	Compare 1 Register High	00	169
		R247	CMP1LR0	Compare 1 Register Low	00	169
		R248	TCR0	Timer Control Register	00	170
		R249	TMR0	Timer Mode Register	00	171
		R250	T_ICR0	External Input Control Register	00	172
		R251	PRSR0	Prescaler Register	00	172
		R252	OACR0	Output A Control Register	00	173
		R253	OBCR0	Output B Control Register	00	174
		R254	T_FLAGR0	Flags Register	00	174
		R255	IDMR0	Interrupt/DMA Mask Register	00	176

Page (Dec)	Block	Reg. No.	Register Name	Description	Reset Value Hex.	Doc. Page
11	STIM	R240	STH	Counter High Byte Register	FF	134
		R241	STL	Counter Low Byte Register	FF	134
		R242	STP	Standard Timer Prescaler Register	FF	134
		R243	STC	Standard Timer Control Register	14	134
20	I2C	R240	I2DCCR	I^2C Control Register	00	227
		R241	I2CSR1	I^2C Status Register 1	00	228
		R242	I2CSR2	I^2C Status Register 2	00	230
		R243	I2CCCR	I^2C Clock Control Register	00	231
		R244	I2COAR1	I^2C Own Address Register 1	00	231
		R245	I2COAR2	I^2C Own Address Register 2	00	232
		R246	I2CDR	I^2C Data Register	00	232
		R247	I2CADR	I^2C General Call Address	A0	232
		R248	I2CISR	I^2C Interrupt Status Register	xx	233
		R249	I2CIVR	I^2C Interrupt Vector Register	xx	234
		R250	I2CRDAP	Receiver DMA Source Addr. Pointer	xx	234
		R251	I2CRDC	Receiver DMA Transaction Counter	xx	234
		R252	I2CTDAP	Transmitter DMA Source Addr. Pointer	xx	235
		R253	I2CTDC	Transmitter DMA Transaction Counter	xx	235
		R254	I2CECCR	Extended Clock Control Register	00	235
		R255	I2CIMR	I^2C Interrupt Mask Register	x0	236
21	MMU	R240	DPR0	Data Page Register 0	xx	35
		R241	DPR1	Data Page Register 1	xx	35
		R242	DPR2	Data Page Register 2	xx	35
		R243	DPR3	Data Page Register 3	xx	35
		R244	CSR	Code Segment Register	00	36
		R248	ISR	Interrupt Segment Register	xx	36
		R249	DMASR	DMA Segment Register	xx	36
	EXTMI	R245	EMR1	External Memory Register 1	80	115
		R246	EMR2	External Memory Register 2	1F	116

Page (Dec)	Block	Reg. No.	Register Name	Description	Reset Value Hex.	Doc. Page
23	JBLPD	R240	STATUS	Status Register	40	259
		R241	TXDATA	Transmit Data Register	xx	260
		R242	RXDATA	Receive Data Register	xx	261
		R243	TXOP	Transmit Opcode Register	00	261
		R244	CLKSEL	System Frequency Selection Register	00	266
		R245	CONTROL	Control Register	40	266
		R246	PADDR	Physiscal Address Register	xx	267
		R247	ERROR	Error Register	00	268
		R248	IVR	Interrupt Vector Register	xx	270
		R249	PRLR	Priority Level Register	10	270
		R250	IMR	Interrupt Mask Register	00	270
		R251	OPTIONS	Options and Register Group Selection	00	272
		R252	CREG0	Current Register 0	xx	274
		R253	CREG1	Current Register 1	xx	274
		R254	CREG2	Current Register 2	xx	274
		R255	CREG3	Current Register 4	xx	274
24	SCI0	R240	RDCPR0	Receiver DMA Transaction Counter Pointer	xx	194
		R241	RDAPR0	Receiver DMA Source Address Pointer	xx	194
		R242	TDCPR0	Transmitter DMA Transaction Counter Pointer	xx	194
		R243	TDAPR0	Transmitter DMA Destination Address Pointer	xx	194
		R244	S_IVR0	Interrupt Vector Register	xx	196
		R245	ACR0	Address/Data Compare Register	xx	196
		R246	IMR0	Interrupt Mask Register	x0	196
		R247	S_ISR0	Interrupt Status Register	xx	196
		R248	RXBR0	Receive Buffer Register	xx	198
		R248	TXBR0	Transmitter Buffer Register	xx	198
		R249	IDPR0	Interrupt/DMA Priority Register	xx	199
		R250	CHCR0	Character Configuration Register	xx	200
		R251	CCR0	Clock Configuration Register	00	201
		R252	BRGHR0	Baud Rate Generator High Reg.	xx	202
		R253	BRGLR0	Baud Rate Generator Low Register	xx	202
		R254	SICR0	Synchronous Input Control	03	202
		R255	SOCR0	Synchronous Output Control	01	203

Page (Dec)	Block	Reg. No.	Register Name	Description	Reset Value Hex.	Doc. Page
25	SCI1	R240	RDCPR1	Receiver DMA Transaction Counter Pointer	xx	194
		R241	RDAPR1	Receiver DMA Source Address Pointer	xx	194
		R242	TDCPR1	Transmitter DMA Transaction Counter Pointer	xx	194
		R243	TDAPR1	Transmitter DMA Destination Address Pointer	xx	194
		R244	S_IVR1	Interrupt Vector Register	xx	196
		R245	ACR1	Address/Data Compare Register	xx	196
		R246	IMR1	Interrupt Mask Register	x0	196
		R247	S_ISR1	Interrupt Status Register	xx	196
		R248	RXBR1	Receive Buffer Register	xx	198
		R248	TXBR1	Transmitter Buffer Register	xx	198
		R249	IDPR1	Interrupt/DMA Priority Register	xx	199
		R250	CHCR1	Character Configuration Register	xx	200
		R251	CCR1	Clock Configuration Register	00	201
		R252	BRGHR1	Baud Rate Generator High Reg.	xx	202
		R253	BRGLR1	Baud Rate Generator Low Register	xx	202
		R254	SICR1	Synchronous Input Control	03	202
		R255	SOCR1	Synchronous Output Control	01	203
28	EFT0	R240	IC1HR0	Input Capture 1 High Register	xx	147
		R241	IC1LR0	Input Capture 1 Low Register	xx	147
		R242	IC2HR0	Input Capture 2 High Register	xx	147
		R243	IC2LR0	Input Capture 2 Low Register	xx	147
		R244	CHR0	Counter High Register	FF	148
		R245	CLR0	Counter Low Register	FC	148
		R246	ACHR0	Alternate Counter High Register	FF	148
		R247	ACLR0	Alternate Counter Low Register	FC	148
		R248	OC1HR0	Output Compare 1 High Register	80	149
		R249	OC1LR0	Output Compare 1 Low Register	00	149
		R250	OC2HR0	Output Compare 2 High Register	80	149
		R251	OC2LR0	Output Compare 2 Low Register	00	149
		R252	CR1_0	Control Register 1	00	151
		R253	CR2_0	Control Register 2	00	151
		R254	SR0	Status Register	00	151
		R255	CR3_0	Control Register 3	00	151

Page (Dec)	Block	Reg. No.	Register Name	Description	Reset Value Hex.	Doc. Page
29	EFT1	R240	IC1HR1	Input Capture 1 High Register	xx	147
		R241	IC1LR1	Input Capture 1 Low Register	xx	147
		R242	IC2HR1	Input Capture 2 High Register	xx	147
		R243	IC2LR1	Input Capture 2 Low Register	xx	147
		R244	CHR1	Counter High Register	FF	148
		R245	CLR1	Counter Low Register	FC	148
		R246	ACHR1	Alternate Counter High Register	FF	148
		R247	ACLR1	Alternate Counter Low Register	FC	148
		R248	OC1HR1	Output Compare 1 High Register	80	149
		R249	OC1LR1	Output Compare 1 Low Register	00	149
		R250	OC2HR1	Output Compare 2 High Register	80	149
		R251	OC2LR1	Output Compare 2 Low Register	00	149
		R252	CR1_1	Control Register 1	00	151
		R253	CR2_1	Control Register 2	00	151
		R254	SR1	Status Register	00	151
		R255	CR3_1	Control Register 3	00	151
43	I/O Port 8	R248	P8C0	Port 8 Configuration Register 0	03	118
		R249	P8C1	Port 8 Configuration Register 1	00	
		R250	P8C2	Port 8 Configuration Register 2	00	
		R251	P8DR	Port 8 Data Register	FF	
	I/O Port 9	R252	P9C0	Port 9 Configuration Register 0	00	
		R253	P9C1	Port 9 Configuration Register 1	00	
		R254	P9C2	Port 9 Configuration Register 2	00	
		R255	P9DR	Port 9 Data Register	FF	
55	RCCU	R240	CLKCTL	Clock Control Register	00	101
		R242	CLK_FLAG	Clock Flag Register	48, 28 or 08	102
		R246	PLLCONF	PLL Configuration Register	xx	102
57	WUIMU	R249	WUCTRL	Wake-Up Control Register	00	86
		R250	WUMRH	Wake-Up Mask Register High	00	87
		R251	WUMRL	Wake-Up Mask Register Low	00	87
		R252	WUTRH	Wake-Up Trigger Register High	00	88
		R253	WUTRL	Wake-Up Trigger Register Low	00	88
		R254	WUPRH	Wake-Up Pending Register High	00	88
		R255	WUPRL	Wake-Up Pending Register Low	00	88

Page (Dec)	Block	Reg. No.	Register Name	Description	Reset Value Hex.	Doc. Page
61	A/D 1	R240	D0R1	Channel 0 Data Register	xx	283
		R241	D1R1	Channel 1 Data Register	xx	283
		R242	D2R1	Channel 2 Data Register	xx	283
		R243	D3R1	Channel 3 Data Register	xx	283
		R244	D4R1	Channel 4 Data Register	xx	283
		R245	D5R1	Channel 5 Data Register	xx	283
		R246	D6R1	Channel 6 Data Register	xx	283
		R247	D7R1	Channel 7 Data Register	xx	283
		R248	LT6R1	Channel 6 Lower Threshold Reg.	xx	284
		R249	LT7R1	Channel 7 Lower Threshold Reg.	xx	284
		R250	UT6R1	Channel 6 Upper Threshold Reg.	xx	284
		R251	UT7R1	Channel 7 Upper Threshold Reg.	xx	284
		R252	CRR1	Compare Result Register	0F	284
		R253	CLR1	Control Logic Register	00	285
		R254	AD_ICR1	Interrupt Control Register	0F	286
		R255	AD_IVR1	Interrupt Vector Register	x2	286
63	A/D 0	R240	D0R0	Channel 0 Data Register	xx	283
		R241	D1R0	Channel 1 Data Register	xx	283
		R242	D2R0	Channel 2 Data Register	xx	283
		R243	D3R0	Channel 3 Data Register	xx	283
		R244	D4R0	Channel 4 Data Register	xx	283
		R245	D5R0	Channel 5 Data Register	xx	283
		R246	D6R0	Channel 6 Data Register	xx	283
		R247	D7R0	Channel 7 Data Register	xx	283
		R248	LT6R0	Channel 6 Lower Threshold Reg.	xx	284
		R249	LT7R0	Channel 7 Lower Threshold Reg.	xx	284
		R250	UT6R0	Channel 6 Upper Threshold Reg.	xx	284
		R251	UT7R0	Channel 7 Upper Threshold Reg.	xx	284
		R252	CRR0	Compare Result Register	0F	284
		R253	CLR0	Control Logic Register	00	285
		R254	AD_ICR0	Interrupt Control Register	0F	286
		R255	AD_IVR0	Interrupt Vector Register	x2	286

Note: xx denotes a byte with an undefined value, however some of the bits may have defined values. Refer to register description for details.

5 INTERRUPTS

5.1 INTRODUCTION

The ST9 responds to peripheral and external events through its interrupt channels. Current program execution can be suspended to allow the ST9 to execute a specific response routine when such an event occurs, providing that interrupts have been enabled, and according to a priority mechanism. If an event generates a valid interrupt request, the current program status is saved and control passes to the appropriate Interrupt Service Routine.

The ST9 CPU can receive requests from the following sources:

– On-chip peripherals

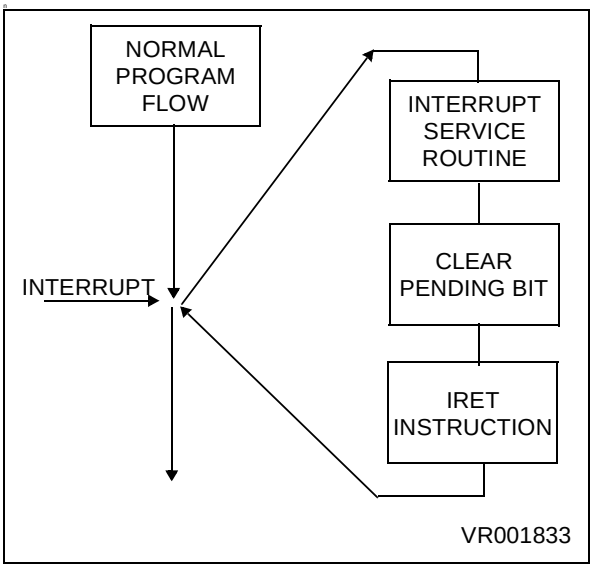
– External pins

– Top-Level Pseudo-non-maskable interrupt

According to the on-chip peripheral features, an event occurrence can generate an Interrupt request which depends on the selected mode.

Up to eight external interrupt channels, with programmable input trigger edge, are available. In addition, a dedicated interrupt channel, set to the Top-level priority, can be devoted either to the external NMI pin (where available) to provide a Non-Maskable Interrupt, or to the Timer/Watchdog. Interrupt service routines are addressed through a vector table mapped in Memory.

Figure 30. Interrupt Response



5.2 INTERRUPT VECTORING

The ST9 implements an interrupt vectoring structure which allows the on-chip peripheral to identify the location of the first instruction of the Interrupt Service Routine automatically.

When an interrupt request is acknowledged, the peripheral interrupt module provides, through its Interrupt Vector Register (IVR), a vector to point into the vector table of locations containing the start addresses of the Interrupt Service Routines (defined by the programmer).

Each peripheral has a specific IVR mapped within its Register File pages.

The Interrupt Vector table, containing the addresses of the Interrupt Service Routines, is located in the first 256 locations of Memory pointed to by the ISR register, thus allowing 8-bit vector addressing. For a description of the ISR register refer to the chapter describing the MMU.

The user Power on Reset vector address is specified in Section 4.2.1. If an external watchdog is used, refer to Section 4.2.2. If an external watchdog is not used, locations 000006h to 000007h must contain FFFFh for correct operation.

The Top Level Interrupt vector is located at addresses 0004h and 0005h in the segment pointed to by the Interrupt Segment Register (ISR).

With one Interrupt Vector register, it is possible to address several interrupt service routines; in fact, peripherals can share the same interrupt vector register among several interrupt channels. The most significant bits of the vector are user programmable to define the base vector address within the vector table, the least significant bits are controlled by the interrupt module, in hardware, to select the appropriate vector.

Note: The first 256 locations of the memory segment pointed to by ISR can contain program code.

5.2.1 Divide by Zero Trap

The Divide by Zero trap vector is located at addresses 0002h and 0003h of each code segment; it should be noted that for each code segment a Divide by Zero service routine is required.

***Caution**. Although the Divide by Zero Trap operates as an interrupt, the FLAG Register is not pushed onto the system Stack automatically. As a result it must be regarded as a subroutine, and the service routine must end with the `RET` instruction (not `IRET`).*

5.2.2 Segment Paging During Interrupt Routines

The ENCSR bit in the EMR2 register can be used to select between original ST9 backward compatibility mode and ST9+ interrupt management mode.

ST9 Backward Compatibility Mode (ENCSR= 0)

If ENCSR is reset, the CPU works in original ST9 compatibility mode. For the duration of the interrupt service routine, ISR is used instead of CSR, and the interrupt stack frame is identical to that of the original ST9: only the PC and Flags are pushed.

This avoids saving the CSR on the stack in the event of an interrupt, thus ensuring a faster interrupt response time.

It is not possible for an interrupt service routine to perform inter-segment calls or jumps: these instructions would update the CSR, which, in this case, is not used (ISR is used instead). The code segment size for all interrupt service routines is thus limited to 64K bytes.

ST9+ mode (ENCSR = 1)

If ENCSR is set, ISR is only used to point to the interrupt vector table and to initialize the CSR at the beginning of the interrupt service routine: the old CSR is pushed onto the stack together with the PC and flags, and CSR is then loaded with the contents of ISR.

In this case, `iret` will also restore CSR from the stack. This approach allows interrupt service routines to access the entire 4 Mbytes of address space. The drawback is that the interrupt response time is slightly increased, because of the need to also save CSR on the stack.

Full compatibility with the original ST9 is lost in this case, because the interrupt stack frame is different.

ENCSR Bit	0	1
Mode	ST9 Compatible	ST9+
Pushed/Popped Registers	PC, FLAGR	PC, FLAGR, CSR
Max. Code Size for interrupt service routine	64KB Within 1 segment	No limit Across segments

5.3 INTERRUPT PRIORITY LEVELS

The ST9 supports a fully programmable interrupt priority structure. Nine priority levels are available to define the channel priority relationships:

– The on-chip peripheral channels and the eight external interrupt sources can be programmed within eight priority levels. Each channel has a 3-bit field, PRL (Priority Level), that defines its priority level in the range from 0 (highest priority) to 7 (lowest priority).

– The 9th level (Top Level Priority) is reserved for the Timer/Watchdog or the External Pseudo Non-Maskable Interrupt. An Interrupt service routine at this level cannot be interrupted in any arbitration mode. Its mask can be both maskable (TLI) or non-maskable (TLNM).

5.4 PRIORITY LEVEL ARBITRATION

The 3 bits of CPL (Current Priority Level) in the Central Interrupt Control Register contain the priority of the currently running program (CPU priority). CPL is set to 7 (lowest priority) upon reset and can be modified during program execution either by software or automatically by hardware according to the selected Arbitration Mode.

During every instruction, an arbitration phase takes place, during which, for every channel capable of generating an Interrupt, each priority level is compared to all the other requests (interrupts or DMA).

If the highest priority request is an interrupt, its PRL value must be strictly lower (that is, higher priority) than the CPL value stored in the CICR register (R230) in order to be acknowledged. The Top Level Interrupt overrides every other priority.

5.4.1 Priority level 7 (Lowest)

Interrupt requests at PRL level 7 cannot be acknowledged, as this PRL value (the lowest possible priority) cannot be strictly lower than the CPL value. This can be of use in a fully polled interrupt environment.

5.4.2 Maximum depth of nesting

No more than 8 routines can be nested. If an interrupt routine at level N is being serviced, no other Interrupts located at level N can interrupt it. This guarantees a maximum number of 8 nested levels including the Top Level Interrupt request.

5.4.3 Simultaneous Interrupts

If two or more requests occur at the same time and at the same priority level, an on-chip daisy chain, specific to every ST9 version, selects the channel

with the highest position in the chain, as shown in Table 17

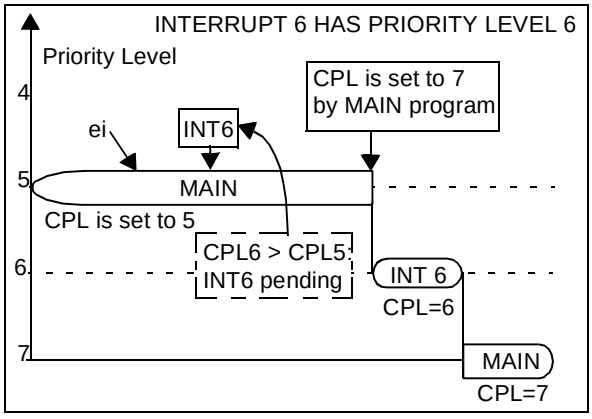
Table 17. Daisy Chain Priority

Highest Position	INTA0 / Watchdog Timer
	INTA1 / Standard Timer
	INTB0 / Extended Function Timer 0
	INTB1 / Extended Function Timer 1
	INTC0 / EEPROM/Flash
	INTC1 / SPI
	INTD0 / RCCU
	INTD1 / WKUP MGT
	Multifunction Timer 0
	JBLPD
	I^2C bus Interface
	A/D Converter 0
	A/D Converter 1
	Multifunction Timer 1
	Serial Communication Interface 0
Lowest Position	Serial Communication Interface 1

5.4.4 Dynamic Priority Level Modification

The main program and routines can be specifically prioritized. Since the CPL is represented by 3 bits in a read/write register, it is possible to modify dynamically the current priority value during program execution. This means that a critical section can have a higher priority with respect to other interrupt requests. Furthermore it is possible to prioritize even the Main Program execution by modifying the CPL during its execution. See Figure 31.

Figure 31. Example of Dynamic priority level modification in Nested Mode



5.5 ARBITRATION MODES

The ST9 provides two interrupt arbitration modes: Concurrent mode and Nested mode. Concurrent mode is the standard interrupt arbitration mode. Nested mode improves the effective interrupt response time when service routine nesting is required, depending on the request priority levels.

The IAM control bit in the CICR Register selects Concurrent Arbitration mode or Nested Arbitration Mode.

5.5.1 Concurrent Mode

This mode is selected when the IAM bit is cleared (reset condition). The arbitration phase, performed during every instruction, selects the request with the highest priority level. The CPL value is not modified in this mode.

Start of Interrupt Routine

The interrupt cycle performs the following steps:

– All maskable interrupt requests are disabled by clearing CICR.IEN.

– The PC low byte is pushed onto system stack.

– The PC high byte is pushed onto system stack.

– If ENCSR is set, CSR is pushed onto system stack.

– The Flag register is pushed onto system stack.

– The PC is loaded with the 16-bit vector stored in the Vector Table, pointed to by the IVR.

– If ENCSR is set, CSR is loaded with ISR contents; otherwise ISR is used in place of CSR until `iret` instruction.

End of Interrupt Routine

The Interrupt Service Routine must be ended with the `iret` instruction. The `iret` instruction executes the following operations:

– The Flag register is popped from system stack.

– If ENCSR is set, CSR is popped from system stack.

– The PC high byte is popped from system stack.

– The PC low byte is popped from system stack.

– All unmasked Interrupts are enabled by setting the CICR.IEN bit.

– If ENCSR is reset, CSR is used instead of ISR.

Normal program execution thus resumes at the interrupted instruction. All pending interrupts remain pending until the next `ei` instruction (even if it is executed during the interrupt service routine).

Note: In Concurrent mode, the source priority level is only useful during the arbitration phase, where it is compared with all other priority levels and with the CPL. No trace is kept of its value during the ISR. If other requests are issued during the interrupt service routine, once the global CICR.IEN is

ARBITRATION MODES (Cont'd)

re-enabled, they will be acknowledged regardless of the interrupt service routine's priority. This may cause undesirable interrupt response sequences.

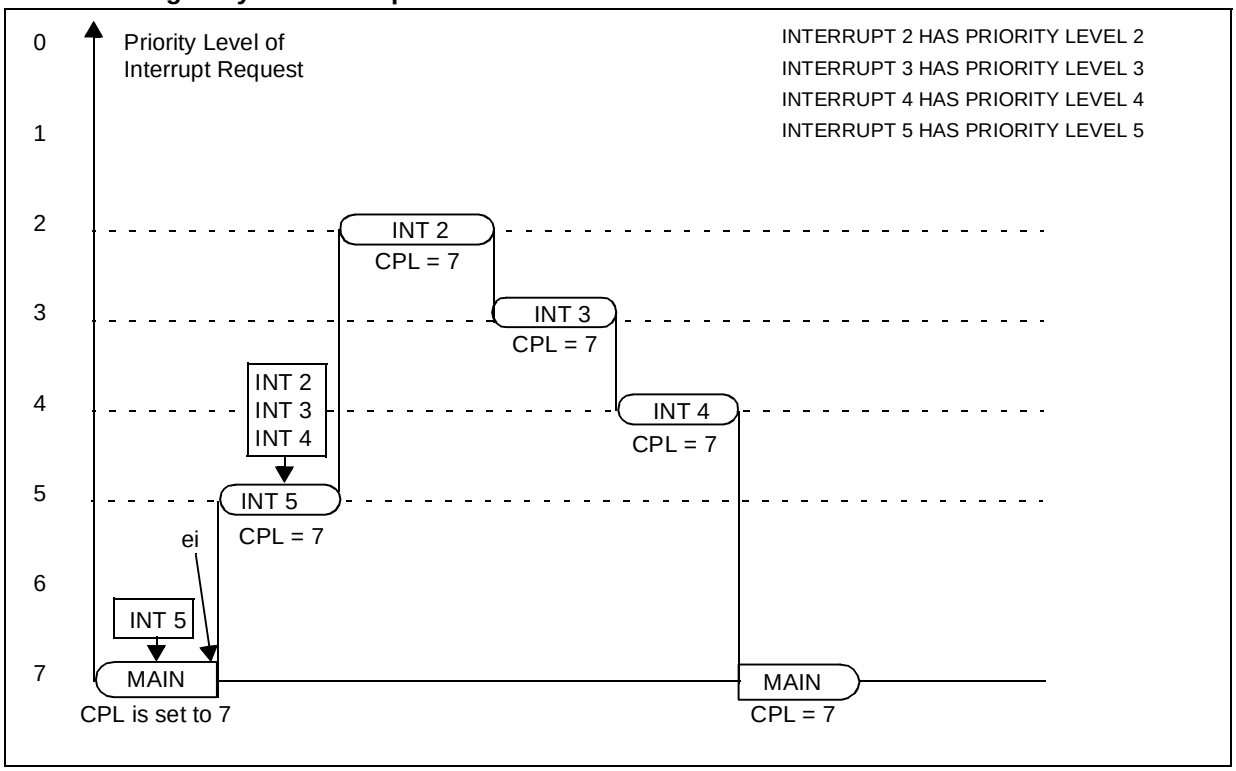
Examples

In the following two examples, three interrupt requests with different priority levels (2, 3 & 4) occur simultaneously during the interrupt 5 service routine.

Example 1

In the first example, (simplest case, Figure 32) the `ei` instruction is not used within the interrupt service routines. This means that no new interrupt can be serviced in the middle of the current one. The interrupt routines will thus be serviced one after another, in the order of their priority, until the main program eventually resumes.

Figure 32. Simple Example of a Sequence of Interrupt Requests with:
- Concurrent mode selected and
- IEN unchanged by the interrupt routines



ARBITRATION MODES (Cont'd)

Example 2

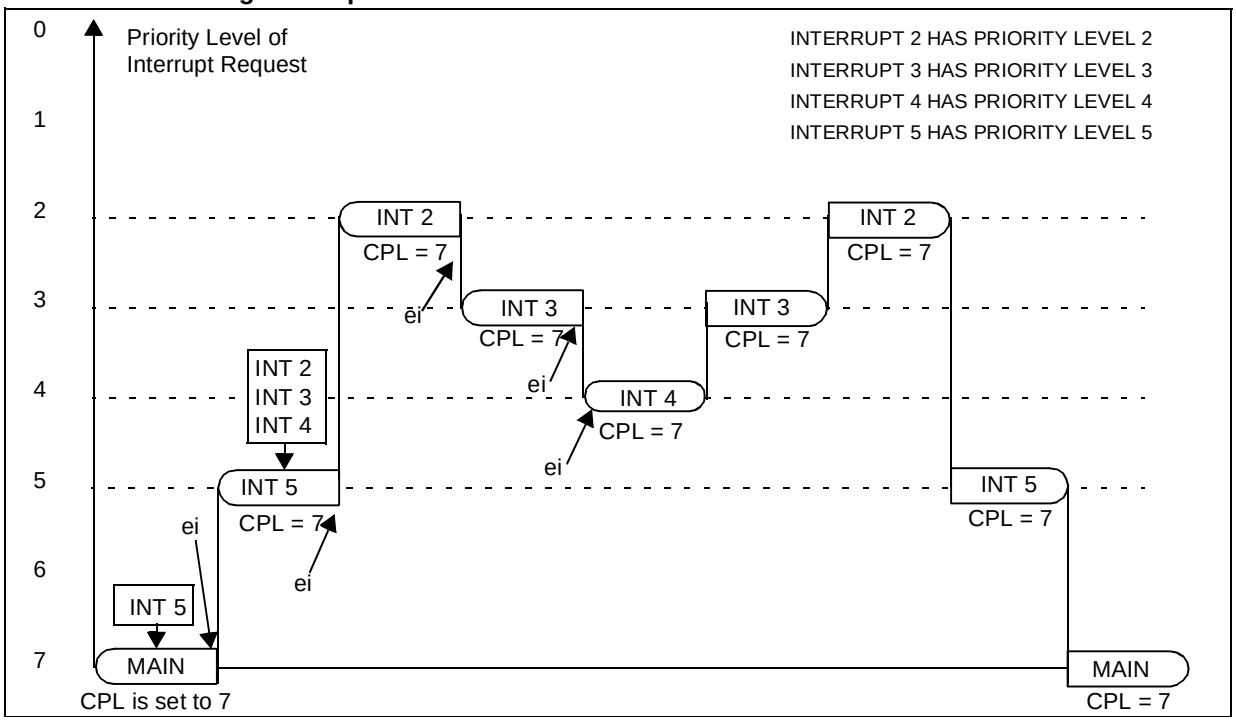
In the second example, (more complex, Figure 33), each interrupt service routine sets Interrupt Enable with the `ei` instruction at the beginning of the routine. Placed here, it minimizes response time for requests with a higher priority than the one being serviced.

The level 2 interrupt routine (with the highest priority) will be acknowledged first, then, when the `ei` instruction is executed, it will be interrupted by the level 3 interrupt routine, which itself will be interrupted by the level 4 interrupt routine. When the level 4 interrupt routine is completed, the level 3 interrupt routine resumes and finally the level 2 interrupt routine. This results in the three interrupt service routines being executed in the opposite order of their priority.

It is therefore recommended to avoid inserting the `ei` instruction in the interrupt service routine in Concurrent mode. **Use the `ei` instruction only in nested mode.**

CAUTION: If, in Concurrent Mode, interrupts are nested (by executing `ei` in an interrupt service routine), make sure that either ENCSR is set or CSR=ISR, otherwise the `iret` of the innermost interrupt will make the CPU use CSR instead of ISR before the outermost interrupt service routine is terminated, thus making the outermost routine fail.

Figure 33. Complex Example of a Sequence of Interrupt Requests with:
- Concurrent mode selected
- IEN set to 1 during interrupt service routine execution



ARBITRATION MODES (Cont'd)

5.5.2 Nested Mode

The difference between Nested mode and Concurrent mode, lies in the modification of the Current Priority Level (CPL) during interrupt processing.

The arbitration phase is basically identical to Concurrent mode, however, once the request is acknowledged, the CPL is saved in the Nested Interrupt Control Register (NICR) by setting the NICR bit corresponding to the CPL value (i.e. if the CPL is 3, the bit 3 will be set).

The CPL is then loaded with the priority of the request just acknowledged; the next arbitration cycle is thus performed with reference to the priority of the interrupt service routine currently being executed.

Start of Interrupt Routine

The interrupt cycle performs the following steps:

– All maskable interrupt requests are disabled by clearing CICR.IEN.

– CPL is saved in the special NICR stack to hold the priority level of the suspended routine.

– Priority level of the acknowledged routine is stored in CPL, so that the next request priority will be compared with the one of the routine currently being serviced.

– The PC low byte is pushed onto system stack.

– The PC high byte is pushed onto system stack.

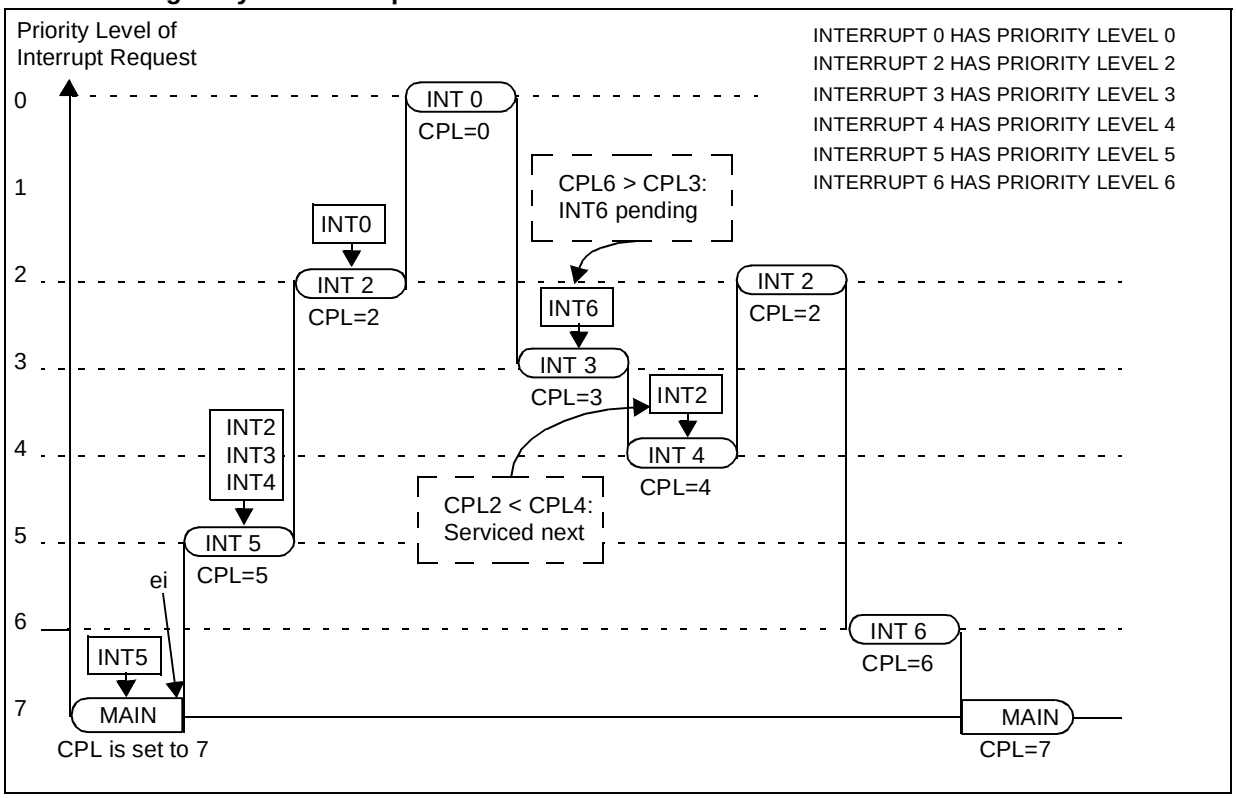
– If ENCSR is set, CSR is pushed onto system stack.

– The Flag register is pushed onto system stack.

– The PC is loaded with the 16-bit vector stored in the Vector Table, pointed to by the IVR.

– If ENCSR is set, CSR is loaded with ISR contents; otherwise ISR is used in place of CSR until `iret` instruction.

Figure 34. Simple Example of a Sequence of Interrupt Requests with:
- Nested mode
- IEN unchanged by the interrupt routines



ARBITRATION MODES (Cont'd)

End of Interrupt Routine

The `iret` Interrupt Return instruction executes the following steps:

– The Flag register is popped from system stack.

– If ENCSR is set, CSR is popped from system stack.

– The PC high byte is popped from system stack.

– The PC low byte is popped from system stack.

– All unmasked Interrupts are enabled by setting the CICR.IEN bit.

– The priority level of the interrupted routine is popped from the special register (NICR) and copied into CPL.

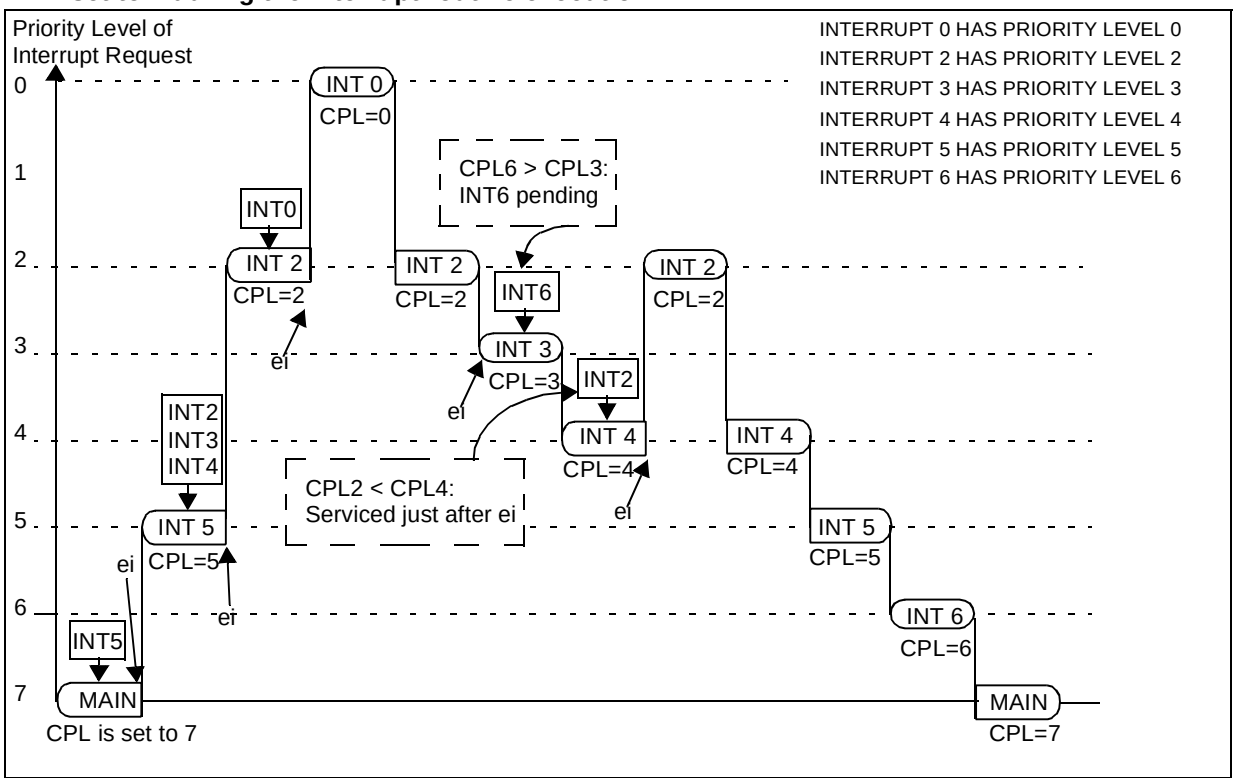
– If ENCSR is reset, CSR is used instead of ISR, unless the program returns to another nested routine.

The suspended routine thus resumes at the interrupted instruction.

Figure 34 contains a simple example, showing that if the `ei` instruction is not used in the interrupt service routines, nested and concurrent modes are equivalent.

Figure 35 contains a more complex example showing how nested mode allows nested interrupt processing (enabled inside the interrupt service routinesi using the `ei` instruction) according to their priority level.

Figure 35. Complex Example of a Sequence of Interrupt Requests with:
- Nested mode
- IEN set to 1 during the interrupt routine execution



5.6 EXTERNAL INTERRUPTS

5.6.1 Standard External Interrupts

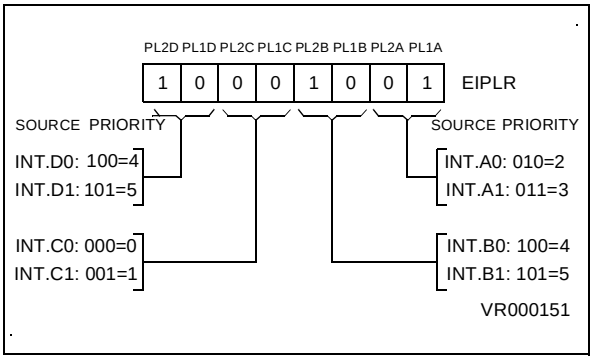
The standard ST9 core contains 8 external interrupts sources grouped into four pairs.

Table 18. External Interrupt Channel Grouping

External Interrupt	Channel	I/O Port Pin
WKUP[0:15]	INTD1	P8[1:0] P7[7:5] P6[7,5] P5[7:5, 2:0] P4[7,4]
INT6	INTD0	P6.1
INT5	INTC1	P6.3
INT4	INTC0	P6.2
INT3	INTB1	P6.3
INT2	INTB0	P6.2
INT1	INTA1	P6.0
INT0	INTA0	P6.0

Each source has a trigger control bit TEA0,..TED1 (R242,EITR.0,..,7 Page 0) to select triggering on the rising or falling edge of the external pin. If the Trigger control bit is set to “1”, the corresponding pending bit IPA0,..,IPD1 (R243,EIPR.0,..,7 Page 0) is set on the input pin rising edge, if it is cleared, the pending bit is set on the falling edge of the input pin. Each source can be individually masked through the corresponding control bit IMA0,..,IMD1 (EIMR.7,..,0). See Figure 37.

Figure 36. Priority Level Examples



The priority level of the external interrupt sources can be programmed among the eight priority levels with the control register EIPLR (R245). The priority level of each pair is software defined using the bits PRL2,PRL1. For each pair, the even channel (A0,B0,C0,D0) of the group has the even priority level and the odd channel (A1,B1,C1,D1) has the odd (lower) priority level.

Figure 36 shows an example of priority levels.

Figure 37 gives an overview of the external interrupts and vectors.

Channel	Internal Interrupt Source	External Interrupt Source
INTA0	Timer/Watchdog	INT0
INTA1	Standard Timer	INT1
INTB0	Extended Function Timer 0	INT2
INTB1	Extended Function Timer 1	INT3
INTC0	EEPROM/Flash	INT4
INTC1	SPI Interrupt	INT5
INTD0	RCCU	INT6
INTD1	Wake-up Management Unit	

– The source of interrupt channel A0 can be selected between the external pin INT0 or the Timer/Watchdog peripheral using the IA0S bit in the EIVR register (R246 Page 0).

– The source of interrupt channel A1 can be selected between the external pin INT1 or the Standard Timer using the INTS bit in the STC register (R232 Page 11).

– The source of the interrupt channel B0 can be selected between the external pin INT2 or the on-chip Extended Function Timer 0 using the EFTIS bit in the CR3 register (R255 Page 28).

– The source of interrupt channel B1 can be selected between external pin INT3 or the on-chip Extended Function Timer 1 using the EFTIS bit in the CR3 register (R255 Page 29).

– The source of the interrupt channel C0 can be selected between external pin INT4 or the On-chip EEPROM/Flash Memory using bit FEIEN in the ECR register (Address 224001h).

– The source of interrupt channel C1 can be selected between external pin INT5 or the on-chip SPI using the SPIS bit in the SPCR0 register (R241 Page 7).

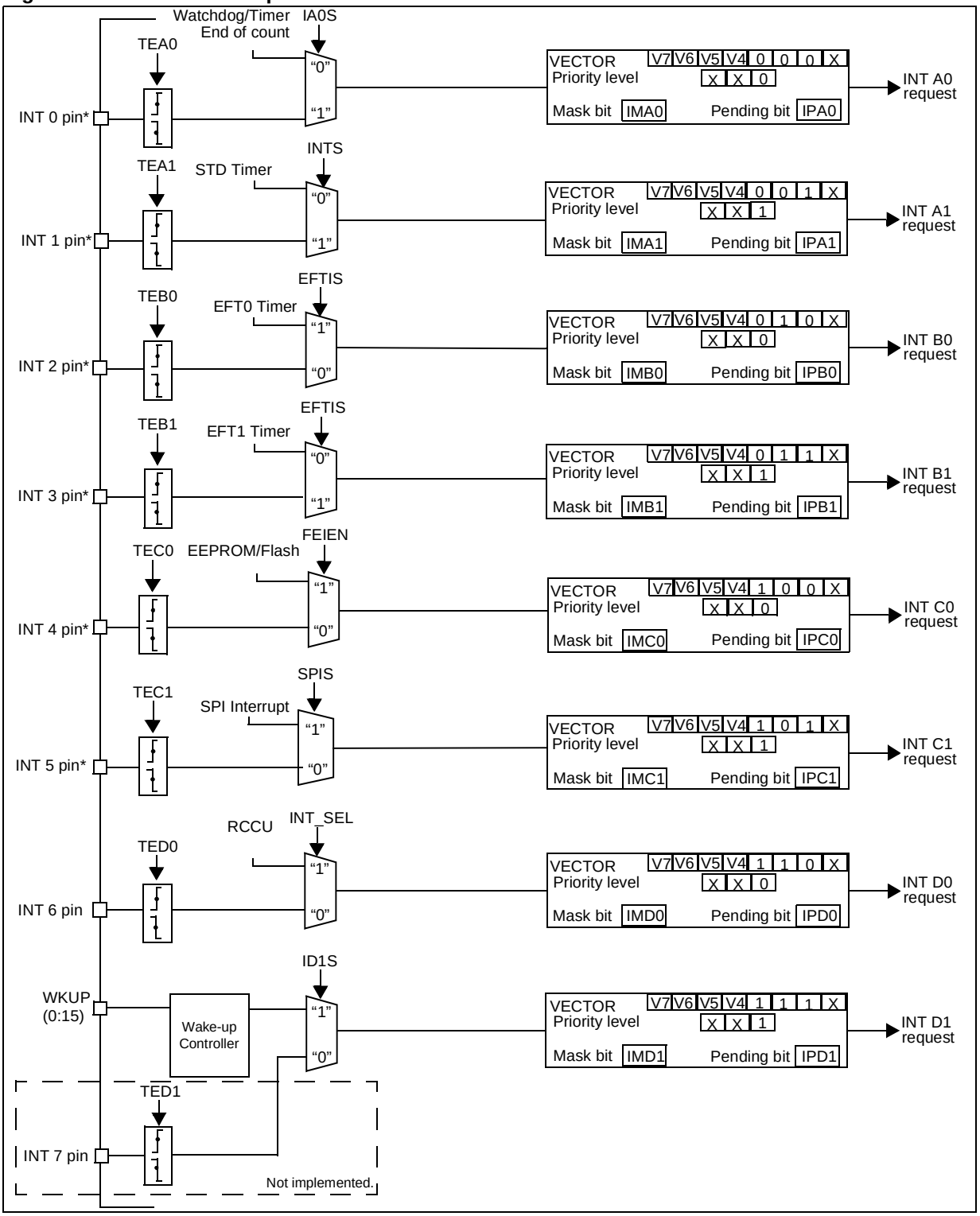
– The source of interrupt channel D0 can be selected between external pin INT6 or the Reset and Clock Unit RCCU using the INT_SEL bit in the CLKCTL register (R240 Page 55).

– The source of interrupt channel D1 selected between the NMI pin and the WUIMU Wakeup/Interrupt Lines using the ID1S bit in the WUCRTL register (R248 Page 9).

Caution: When using external interrupt channels shared by both external interrupts and peripherals, special care must be taken to configure control registers both for peripheral and interrupts.

EXTERNAL INTERRUPTS (Cont'd)

Figure 37. External Interrupts Control Bits and Vectors



* Only four interrupt pins are available at the same time. Refer to Table 18 for I/O pin mapping.

5.7 TOP LEVEL INTERRUPT

The Top Level Interrupt channel can be assigned either to the external pin NMI or to the Timer/ Watchdog according to the status of the control bit EIVR.TLIS (R246.2, Page 0). If this bit is high (the reset condition) the source is the external pin NMI. If it is low, the source is the Timer/ Watchdog End Of Count. When the source is the NMI external pin, the control bit EIVR.TLTEV (R246.3; Page 0) selects between the rising (if set) or falling (if reset) edge generating the interrupt request. When the selected event occurs, the CICR.TLIP bit (R230.6) is set. Depending on the mask situation, a Top Level Interrupt request may be generated. Two kinds of masks are available, a Maskable mask and a Non-Maskable mask. The first mask is the CICR.TLI bit (R230.5): it can be set or cleared to enable or disable respectively the Top Level Interrupt request. If it is enabled, the global Enable Interrupt bit, CICR.IEN (R230.4) must also be enabled in order to allow a Top Level Request.

The second mask NICR.TLNM (R247.7) is a set-only mask. Once set, it enables the Top Level Interrupt request independently of the value of CICR.IEN and it cannot be cleared by the program. Only the processor RESET cycle can clear this bit. This does not prevent the user from ignoring some sources due to a change in TLIS.

The Top Level Interrupt Service Routine cannot be interrupted by any other interrupt or DMA request, in any arbitration mode, not even by a subsequent Top Level Interrupt request.

Caution. The interrupt machine cycle of the Top Level Interrupt does not clear the CICR.IEN bit, and the corresponding `iret` does not set it. Furthermore the TLI never modifies the CPL bits and the NICR register.

5.8 ON-CHIP PERIPHERAL INTERRUPTS

The general structure of the peripheral interrupt unit is described here, however each on-chip peripheral has its own specific interrupt unit containing one or more interrupt channels, or DMA channels. Please refer to the specific peripheral chapter for the description of its interrupt features and control registers.

The on-chip peripheral interrupt channels provide the following control bits:

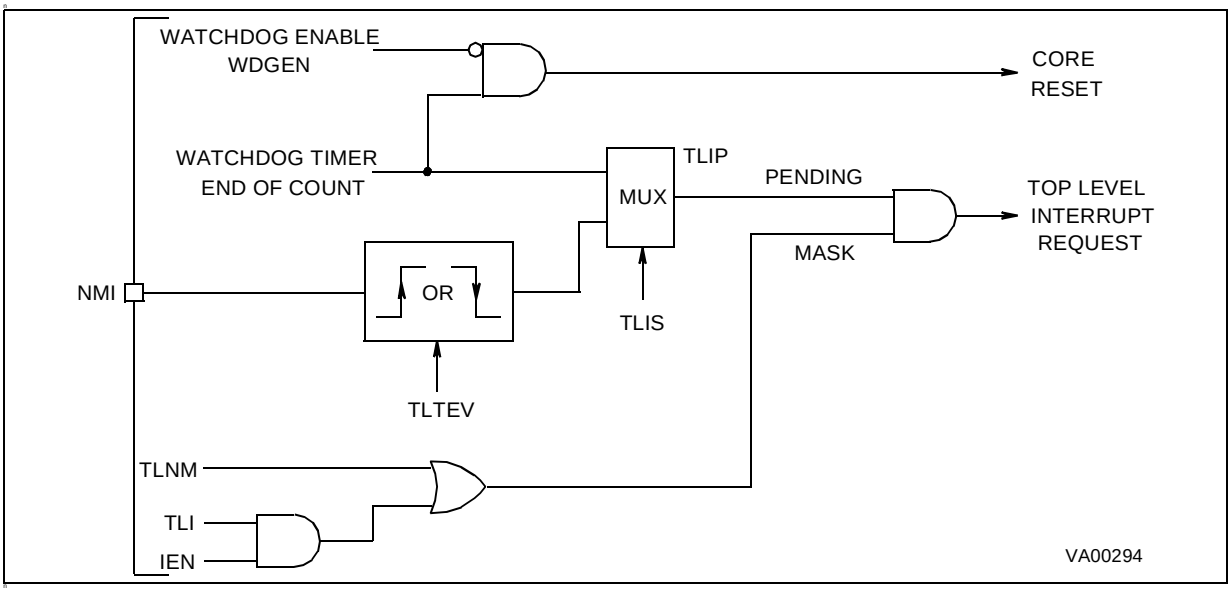
– **Interrupt Pending bit** (IP). Set by hardware when the Trigger Event occurs. Can be set/ cleared by software to generate/cancel pending interrupts and give the status for Interrupt polling.

– **Interrupt Mask bit** (IM). If IM = “0”, no interrupt request is generated. If IM =“1” an interrupt request is generated whenever IP = “1” and CICR.IEN = “1”.

– **Priority Level** (PRL, 3 bits). These bits define the current priority level, PRL=0: the highest priority, PRL=7: the lowest priority (the interrupt cannot be acknowledged)

– **Interrupt Vector Register** (IVR, up to 7 bits). The IVR points to the vector table which itself contains the interrupt routine start address.

Figure 38. Top Level Interrupt Structure



5.9 INTERRUPT RESPONSE TIME

The interrupt arbitration protocol functions completely asynchronously from instruction flow and requires 5 clock cycles. One more CPUCLK cycle is required when an interrupt is acknowledged. Requests are sampled every 5 CPUCLK cycles.

If the interrupt request comes from an external pin, the trigger event must occur a minimum of one INTCLK cycle before the sampling time.

When an arbitration results in an interrupt request being generated, the interrupt logic checks if the current instruction (which could be at any stage of execution) can be safely aborted; if this is the case, instruction execution is terminated immediately and the interrupt request is serviced; if not, the CPU waits until the current instruction is terminated and then services the request. Instruction execution can normally be aborted provided no write operation has been performed.

For an interrupt deriving from an external interrupt channel, the response time between a user event and the start of the interrupt service routine can range from a minimum of 26 clock cycles to a maximum of 55 clock cycles (DIV instruction), 53 clock cycles (DIVWS and MUL instructions) or 49 for other instructions.

For a non-maskable Top Level interrupt, the response time between a user event and the start of the interrupt service routine can range from a minimum of 22 clock cycles to a maximum of 51 clock cycles (DIV instruction), 49 clock cycles (DIVWS and MUL instructions) or 45 for other instructions.

In order to guarantee edge detection, input signals must be kept low/high for a minimum of one INTCLK cycle.

An interrupt machine cycle requires a basic 18 internal clock cycles (CPUCLK), to which must be added a further 2 clock cycles if the stack is in the Register File. 2 more clock cycles must further be added if the CSR is pushed (ENCSR =1).

The interrupt machine cycle duration forms part of the two examples of interrupt response time previously quoted; it includes the time required to push values on the stack, as well as interrupt vector handling.

In Wait for Interrupt mode, a further cycle is required as wake-up delay.

5.10 INTERRUPT REGISTERS

CENTRAL INTERRUPT CONTROL REGISTER (CICR)
R230 - Read/Write
Register Group: System
Reset value: 1000 0111 (87h)

7							0
GCEN	TLIP	TLI	IEN	IAM	CPL2	CPL1	CPL0

Bit 7 = **GCEN**: *Global Counter Enable.*
This bit enables the 16-bit Multifunction Timer peripheral.
0: MFT disabled
1: MFT enabled

Bit 6 = **TLIP**: *Top Level Interrupt Pending.*
This bit is set by hardware when Top Level Interrupt (TLI) trigger event occurs. It is cleared by hardware when a TLI is acknowledged. It can also be set by software to implement a software TLI.
0: No TLI pending
1: TLI pending

Bit 5 = **TLI**: *Top Level Interrupt.*
This bit is set and cleared by software.
0: A Top Level Interrupt is generared when TLIP is set, only if TLNM=1 in the NICR register (independently of the value of the IEN bit).
1: A Top Level Interrupt request is generated when IEN=1 and the TLIP bit are set.

Bit 4 = **IEN**: *Interrupt Enable.*
This bit is cleared by the interrupt machine cycle (except for a TLI).
It is set by the `iret` instruction (except for a return from TLI).
It is set by the `EI` instruction.
It is cleared by the `DI` instruction.
0: Maskable interrupts disabled
1: Maskable Interrupts enabled

Note: The IEN bit can also be changed by software using any instruction that operates on register CICR, however in this case, take care to avoid spurious interrupts, since IEN cannot be cleared in the middle of an interrupt arbitration. Only modify the IEN bit when interrupts are disabled or when no peripheral can generate interrupts. For example, if the state of IEN is not known in advance, and its value must be restored from a previous push of CICR on the stack, use the sequence `DI; POP CICR` to make sure that no interrupts are being arbitrated when CICR is modified.

Bit 3 = **IAM**: *Interrupt Arbitration Mode.*
This bit is set and cleared by software.
0: Concurrent Mode
1: Nested Mode

Bits 2:0 = **CPL[2:0]:** *Current Priority Level.*
These bits define the Current Priority Level. CPL=0 is the highest priority. CPL=7 is the lowest priority. These bits may be modified directly by the interrupt hardware when Nested Interrupt Mode is used.

EXTERNAL INTERRUPT TRIGGER REGISTER (EITR)
R242 - Read/Write
Register Page: 0
Reset value: 0000 0000 (00h)

7							0
TED1	TED0	TEC1	TEC0	TEB1	TEB0	TEA1	TEA0

Bit 7 = **TED1**: *INTD1 Trigger Event*
Bit 6 = **TED0**: *INTD0 Trigger Event*
Bit 5 = **TEC1**: *INTC1 Trigger Event*
Bit 4 = **TEC0**: *INTC0 Trigger Event*
Bit 3 = **TEB1**: *INTB1 Trigger Event*
Bit 2 = **TEB0**: *INTB0 Trigger Event*
Bit 1 = **TEA1**: *INTA1 Trigger Event*
Bit 0 = **TEA0**: *INTA0 Trigger Event*

These bits are set and cleared by software.
0: Select falling edge as interrupt trigger event
1: Select rising edge as interrupt trigger event

INTERRUPT REGISTERS (Cont'd)

EXTERNAL INTERRUPT PENDING REGISTER (EIPR)
R243 - Read/Write
Register Page: 0
Reset value: 0000 0000 (00h)

7							0
IPD1	IPD0	IPC1	IPC0	IPB1	IPB0	IPA1	IPA0

Bit 7 = **IPD1**: *INTD1 Interrupt Pending bit*

Bit 6 = **IPD0**: *INTD0 Interrupt Pending bit*

Bit 5 = **IPC1**: *INTC1 Interrupt Pending bit*

Bit 4 = **IPC0**: *INTC0 Interrupt Pending bit*

Bit 3 = **IPB1**: *INTB1 Interrupt Pending bit*

Bit 2 = **IPB0**: *INTB0 Interrupt Pending bit*

Bit 1 = **IPA1**: *INTA1 Interrupt Pending bit*

Bit 0 = **IPA0**: *INTA0 Interrupt Pending bit*

These bits are set by hardware on occurrence of a trigger event (as specified in the EITR register) and are cleared by hardware on interrupt acknowledge. They can also be set by software to implement a software interrupt.
0: No interrupt pending
1: Interrupt pending

EXTERNAL INTERRUPT MASK-BIT REGISTER (EIMR)
R244 - Read/Write
Register Page: 0
Reset value: 0000 0000 (00h**)**

7							0
IMD1	IMD0	IMC1	IMC0	IMB1	IMB0	IMA1	IMA0

Bit 7 = **IMD1**: *INTD1 Interrupt Mask*

Bit 6 = **IMD0**: *INTD0 Interrupt Mask*

Bit 5 = **IMC1**: *INTC1 Interrupt Mask*

Bit 4 = **IMC0**: *INTC0 Interrupt Mask*

Bit 3 = **IMB1**: *INTB1 Interrupt Mask*

Bit 2 = **IMB0**: *INTB0 Interrupt Mask*

Bit 1 = **IMA1**: *INTA1 Interrupt Mask*

Bit 0 = **IMA0**: *INTA0 Interrupt Mask*

These bits are set and cleared by software.
0: Interrupt masked
1: Interrupt not masked (an interrupt is generated if the IPxx and IEN bits = 1)

EXTERNAL INTERRUPT PRIORITY LEVEL REGISTER (EIPLR)
R245 - Read/Write
Register Page: 0
Reset value: 1111 1111 (FFh**)**

7							0
PL2D	PL1D	PL2C	PL1C	PL2B	PL1B	PL2A	PL1A

Bits 7:6 = **PL2D, PL1D:** *INTD0, D1 Priority Level.*

Bits 5:4 = **PL2C, PL1C**: *INTC0, C1 Priority Level.*

Bits 3:2 = **PL2B, PL1B**: *INTB0, B1 Priority Level.*

Bits 1:0 = **PL2A, PL1A**: *INTA0, A1 Priority Level.*

These bits are set and cleared by software.

The priority is a three-bit value. The LSB is fixed by hardware at 0 for Channels A0, B0, C0 and D0 and at 1 for Channels A1, B1, C1 and D1.

PL2x	PL1x	Hardware bit	Priority
0	0	0 1	0 (Highest) 1
0	1	0 1	2 3
1	0	0 1	4 5
1	1	0 1	6 7 (Lowest)

INTERRUPT REGISTERS (Cont'd)

EXTERNAL INTERRUPT VECTOR REGISTER (EIVR)

R246 - Read/Write
Register Page: 0
Reset value: xxxx 0110b (x6h)

7							0
V7	V6	V5	V4	TLTEV	TLIS	IAOS	EWEN

Bits 7:4 = **V[7:4]**: *Most significant nibble of External Interrupt Vector*.
These bits are not initialized by reset. For a representation of how the full vector is generated from V[7:4] and the selected external interrupt channel, refer to Figure 37.

Bit 3 = **TLTEV**: *Top Level Trigger Event bit.*
This bit is set and cleared by software.
0: Select falling edge as NMI trigger event
1: Select rising edge as NMI trigger event

Bit 2 = **TLIS**: *Top Level Input Selection*.
This bit is set and cleared by software.
0: Watchdog End of Count is TL interrupt source (the IA0S bit must be set in this case)
1: NMI is TL interrupt source

Bit 1 = **IA0S**: *Interrupt Channel A0 Selection.*
This bit is set and cleared by software.
0: Watchdog End of Count is INTA0 source (the TLIS bit must be set in this case)
1: External Interrupt pin is INTA0 source

Bit 0 = **EWEN**: *External Wait Enable.*
This bit is set and cleared by software.
0: WAITN pin disabled
1: WAITN pin enabled (to stretch the external memory access cycle).

Note: For more details on Wait mode refer to the section describing the WAITN pin in the External Memory Chapter.

NESTED INTERRUPT CONTROL (NICR)

R247 - Read/Write
Register Page: 0
Reset value: 0000 0000 (00h)

7							0
TLNM	HL6	HL5	HL4	HL3	HL2	HL1	HL0

Bit 7 = **TLNM**: *Top Level Not Maskable*.
This bit is set by software and cleared only by a hardware reset.
0: Top Level Interrupt Maskable. A top level request is generated if the IEN, TLI and TLIP bits =1
1: Top Level Interrupt Not Maskable. A top level request is generated if the TLIP bit =1

Bits 6:0 = **HL[6:0]**: *Hold Level* x
These bits are set by hardware when, in Nested Mode, an interrupt service routine at level x is interrupted from a request with higher priority (other than the Top Level interrupt request). They are cleared by hardware at the `iret` execution when the routine at level x is recovered.

5.11 WAKE-UP / INTERRUPT LINES MANAGEMENT UNIT (WUIMU)

5.11.1 Introduction

The Wake-up/Interrupt Management Unit extends the number of external interrupt lines from 8 to 23 (depending on the number of external interrupt lines mapped on external pins of the device). It allows the source of the INTD1 external interrupt channel to be selected between the INT7 pin (when available) and up to 16 additional external Wake-up/interrupt pins.

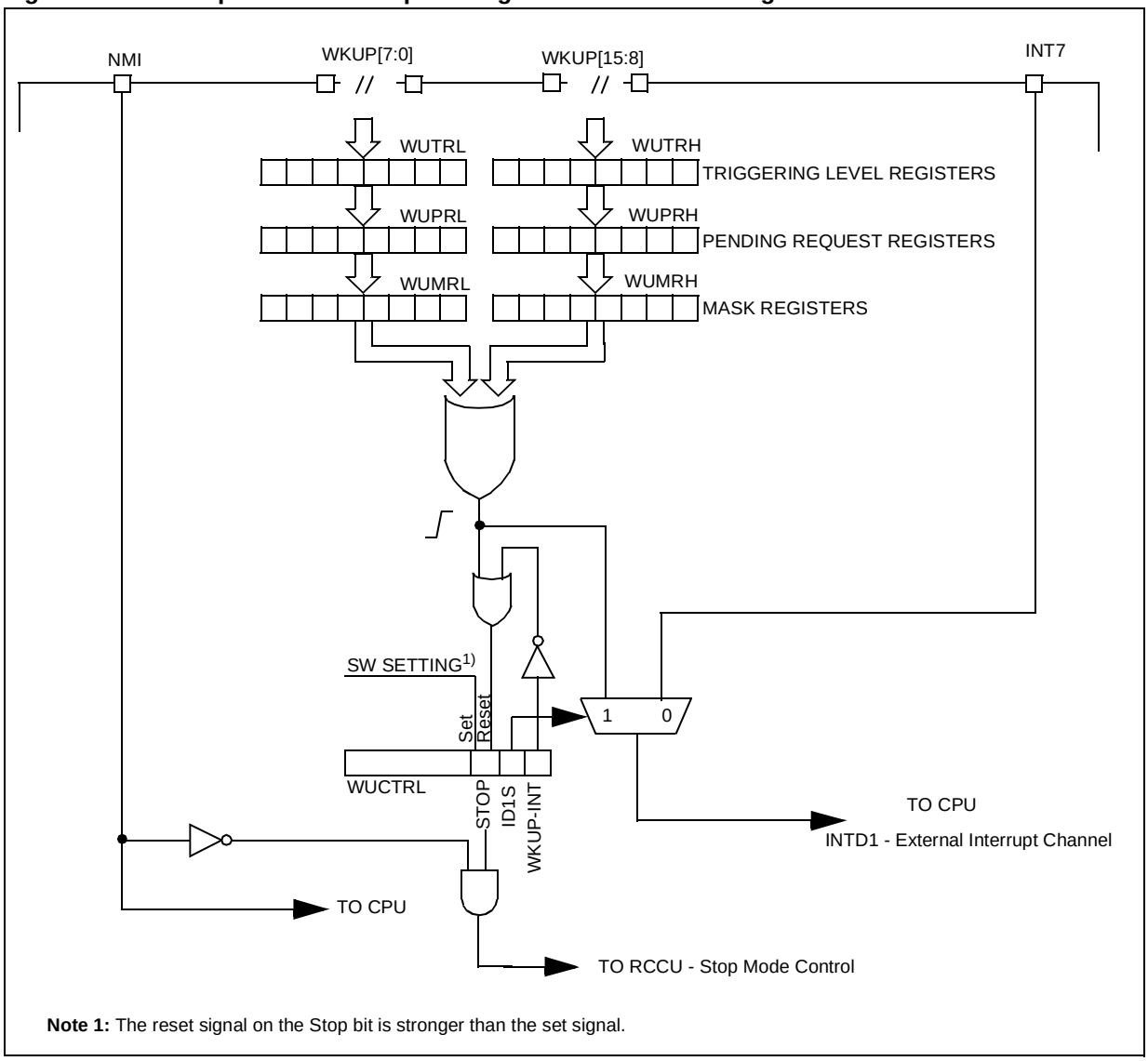
These 16 WKUP pins can be programmed as external interrupt lines or as wake-up lines, able to exit the microcontroller from low power mode (STOP mode) (see Figure 1).

5.11.2 Main Features

- Supports up to 16 additional external wake-up or interrupt lines
- Wake-Up lines can be used to wake-up the ST9 from STOP mode.
- Programmable selection of wake-up or interrupt
- Programmable wake-up trigger edge polarity
- All Wake-Up Lines maskable

Note: The number of available pins is device dependent. Refer to the device pinout description.

Figure 39. Wake-Up Lines / Interrupt Management Unit Block Diagram



Note 1: The reset signal on the Stop bit is stronger than the set signal.

WAKE-UP / INTERRUPT LINES MANAGEMENT UNIT (Cont'd)

5.11.3 Functional Description

5.11.3.1 Interrupt Mode

To configure the 16 wake-up lines as interrupt sources, use the following procedure:

1. Configure the mask bits of the 16 wake-up lines (WUMRL, WUMRH)
2. Configure the triggering edge registers of the wake-up lines (WUTRL, WUTRH)
3. Set bit 7 of EIMR (R244 Page 0) and EITR (R242 Page 0) registers of the CPU: so an interrupt coming from one of the 16 lines can be correctly acknowledged
4. Reset the WKUP-INT bit in the WUCTRL register to disable Wake-up Mode
5. Set the ID1S bit in the WUCTRL register to disable the INT7 external interrupt source and enable the 16 wake-up lines as external interrupt source lines.

To return to standard mode (INT7 external interrupt source enabled and 16 wake-up lines disabled) it is sufficient to reset the ID1S bit.

5.11.3.2 Wake-up Mode Selection

To configure the 16 lines as wake-up sources, use the following procedure:

1. Configure the mask bits of the 16 wake-up lines (WUMRL, WUMRH).
2. Configure the triggering edge registers of the wake-up lines (WUTRL, WUTRH).
3. Set, as for Interrupt Mode selection, bit 7 of EIMR and EITR registers only if an interrupt routine is to be executed after a wake-up event. Otherwise, if the wake-up event only restarts the execution of the code from where it was stopped, the INTD1 interrupt channel must be masked or the external source must be selected by resetting the ID1S bit.
4. Since the RCCU can generate an interrupt request when exiting from STOP mode, take care to mask it even if the wake-up event is only to restart code execution.
5. Set the WKUP-INT bit in the WUCTRL register to select Wake-up Mode
6. Set the ID1S bit in the WUCTRL register to disable the INT7 external interrupt source and enable the 16 wake-up lines as external interrupt source lines. This is not mandatory if the wake-up event does not require an interrupt response.
7. Write the sequence 1,0,1 to the STOP bit of the WUCTRL register with three consecutive write operations. This is the STOP bit setting sequence.

To detect if STOP Mode was entered or not, immediately after the STOP bit setting sequence, poll the RCCU EX_STP bit (R242.7, Page 55) and the STOP bit itself.

5.11.3.3 STOP Mode Entry Conditions

Assuming the ST9 is in Run mode: during the STOP bit setting sequence the following cases may occur:

Case 1: NMI = 0, wrong STOP bit setting sequence

This can happen if an Interrupt/DMA request is acknowledged during the STOP bit setting sequence. In this case polling the STOP and EX_STP bits will give:

STOP = 0, EX_STP = 0

This means that the ST9 did not enter STOP mode due to a bad STOP bit setting sequence: the user must retry the sequence.

Case 2: NMI = 0, correct STOP bit setting sequence

In this case the ST9 enters STOP mode. There are two ways to exit STOP mode:

1. A wake-up interrupt (not an NMI interrupt) is acknowledged. That implies:

   STOP = 0, EX_STP = 1

This means that the ST9 entered and exited STOP mode due to an external wake-up line event.

2. A NMI rising edge woke up the ST9. This implies:

   STOP = 1, EX_STP = 1

This means that the ST9 entered and exited STOP mode due to an NMI (rising edge) event. The user should clear the STOP bit via software.

WAKE-UP / INTERRUPT LINES MANAGEMENT UNIT (Cont'd)

Case 3: NMI = 1 (NMI kept high during the 3rd write instruction of the sequence), bad STOP bit setting sequence

The result is the same as Case 1:

STOP = 0, EX_STP = 0

This means that the ST9 did not enter STOP mode due to a bad STOP bit setting sequence: the user must retry the sequence.

Case 4: NMI = 1 (NMI kept high during the 3rd write instruction of the sequence), correct STOP bit setting sequence

In this case:

STOP = 1, EX_STP = 0

This means that the ST9 did not enter STOP mode due to NMI being kept high. The user should clear the STOP bit via software.

Note: If NMI goes to 0 before resetting the STOP bit, the ST9 will not enter STOP mode.

Case 5: A rising edge on the NMI pin occurs during the STOP bit setting sequence.

The NMI interrupt will be acknowledged and the ST9 will not enter STOP mode. This implies:

STOP = 0, EX_STP = 0

This means that the ST9 did not enter STOP mode due to an NMI interrupt serviced during the STOP bit setting sequence. At the end of NMI routine, the user must re-enter the sequence: if NMI is still high at the end of the sequence, the ST9 can not enter STOP mode (See "NMI Pin Management" on page 4).

Case 6: A wake-up event on the external wake-up lines occurs during the STOP bit setting sequence

There are two possible cases:

1. Interrupt requests to the CPU are disabled: in this case the ST9 will not enter STOP mode, no interrupt service routine will be executed and the program execution continues from the instruction following the STOP bit setting sequence. The status of STOP and EX_STP bits will be again:

   STOP = 0, EX_STP = 0

   The application can determine why the ST9 did not enter STOP mode by polling the pending bits of the external lines (at least one must be at 1).

2. Interrupt requests to CPU are enabled: in this case the ST9 will not enter STOP mode and the interrupt service routine will be executed. The status of STOP and EX_STP bits will be again:

   STOP = 0, EX_STP = 0

   The interrupt service routine can determine why the ST9 did not enter STOP mode by polling the pending bits of the external lines (at least one must be at 1).

If the MCU really exits from STOP Mode, the RCCU EX_STP bit is still set and must be reset by software. Otherwise, if NMI was high or an Interrupt/DMA request was acknowledged during the STOP bit setting sequence, the RCCU EX_STP bit is reset. This means that the MCU has filtered the STOP Mode entry request.

The WKUP-INT bit can be used by an interrupt routine to detect and to distinguish events coming from Interrupt Mode or from Wake-up Mode, allowing the code to execute different procedures.

To exit STOP mode, it is sufficient that one of the 16 wake-up lines (not masked) generates an event: the clock restarts after the delay needed for the oscillator to restart.

The same effect is obtained when a rising edge is detected on the NMI pin, which works as a 17th wake-up line.

Note: After exiting from STOP Mode, the software can successfully reset the pending bits (edge sensitive), even though the corresponding wake-up line is still active (high or low, depending on the Trigger Event register programming); the user must poll the external pin status to detect and distinguish a short event from a long one (for example keyboard input with keystrokes of varying length).

WAKE-UP / INTERRUPT LINES MANAGEMENT UNIT (Cont'd)

5.11.3.4 NMI Pin Management

On the CPU side, if TLTEV=1 (Top Level Trigger Event, bit 3 of register R246, page 0) then a rising edge on the NMI pin will set the TLIP bit (Top Level Interrupt Pending bit, R230.6). At this point an interrupt request to the CPU is given either if TLNM=1 (Top Level Not Maskable bit, R247.7 - once set it can only be cleared by RESET) or if TLI=1 and IEN=1 (bits R230.5, R230.4).

Assuming that the application uses a non-maskable Top Level Interrupt (TLNM=1): in this case, whenever a rising edge occurs on the NMI pin, the related service routine will be executed. To service further Top Level Interrupt Requests, it is necessary to generate a new rising edge on the external NMI pin.

The following summarizes some typical cases:

– If the ST9 is in STOP mode and a rising edge on the NMI pin occurs, the ST9 will exit STOP mode and the NMI service routine will be executed.

– If the ST9 is in Run mode and a rising edge occurs on the NMI pin: the NMI service routine is executed and then the ST9 restarts the execution of the main program. **Now, suppose that the user wants to enter STOP mode with NMI still at 1. The ST9 will not enter STOP mode and it will not execute an NMI routine because there were no transitions on the external NMI line**.

– If the ST9 is in run mode and a rising edge on NMI pin occurs during the STOP bit setting sequence: the NMI interrupt will be acknowledged and the ST9 will not enter STOP mode. At the end of the NMI routine, the user must re-enter the sequence: if NMI is still high at the end of the sequence, the ST9 can not enter STOP mode (see previous case).

– If the ST9 is in run mode and the NMI pin is high: if NMI is forced low just before the third write instruction of the STOP bit setting sequence then the ST9 will enter STOP mode.

WAKE-UP / INTERRUPT LINES MANAGEMENT UNIT (Cont'd)

5.11.4 Programming Considerations

The following paragraphs give some guidelines for designing an application program.

5.11.4.1 Procedure for Entering/Exiting STOP mode

1. Program the polarity of the trigger event of external wake-up lines by writing registers WUTRH and WUTRL.
2. Check that at least one mask bit (registers WUMRH, WUMRL) is equal to 1 (so at least one external wake-up line is not masked).
3. Reset at least the unmasked pending bits: this allows a rising edge to be generated on the INTD1 channel when the trigger event occurs (an interrupt on channel INTD1 is recognized when a rising edge occurs).
4. Select the interrupt source of the INTD1 channel (see description of ID1S bit in the WUCTRL register) and set the WKUP-INT bit.
5. To generate an interrupt on channel INTD1, bits EITR.1 (R242.7, Page 0) and EIMR.1 (R244.7, Page 0) must be set and bit EIPR.7 must be reset. Bits 7 and 6 of register R245, Page 0 must be written with the desired priority level for interrupt channel INTD1.
6. Reset the STOP bit in register WUCTRL and the EX_STP bit in the CLK_FLAG register (R242.7, Page 55). Refer to the RCCU chapter.
7. To enter STOP mode, write the sequence 1, 0, 1 to the STOP bit in the WUCTRL register with three consecutive write operations.
8. The code to be executed just after the STOP sequence must check the status of the STOP and RCCU EX_STP bits to determine if the ST9 entered STOP mode or not (See "Wake-up Mode Selection" on page 2 for details). If the ST9 did not enter in STOP mode it is necessary to reloop the procedure from the beginning, otherwise the procedure continues from next point.
9. Poll the wake-up pending bits to determine which wake-up line caused the exit from STOP mode.
10. Clear the wake-up pending bit that was set.

5.11.4.2 Simultaneous Setting of Pending Bits

It is possible that several simultaneous events set different pending bits. In order to accept subsequent events on external wake-up/interrupt lines, it is necessary to clear at least one pending bit: this operation allows a rising edge to be generated on the INTD1 line (if there is at least one more pending bit set and not masked) and so to set EIPR.7 bit again. A further interrupt on channel INTD1 will be serviced depending on the status of bit EIMR.7. Two possible situations may arise:

1. The user chooses to reset all pending bits: no further interrupt requests will be generated on channel INTD1. In this case the user has to:
   - Reset EIMR.7 bit (to avoid generating a spurious interrupt request during the next reset operation on the WUPRH register)
   - Reset WUPRH register using a read-modify-write instruction (AND, BRES, BAND)
   - Clear the EIPR.7 bit
   - Reset the WUPRL register using a read-modify-write instruction (AND, BRES, BAND)
2. The user chooses to keep at least one pending bit active: at least one additional interrupt request will be generated on the INTD1 channel. In this case the user has to reset the desired pending bits with a read-modify-write instruction (AND, BRES, BAND). This operation will generate a rising edge on the INTD1 channel and the EIPR.7 bit will be set again. An interrupt on the INTD1 channel will be serviced depending on the status of EIMR.7 bit.

WAKE-UP / INTERRUPT LINES MANAGEMENT UNIT (Cont'd)

5.11.5 Register Description

WAKE-UP CONTROL REGISTER (**WUCTRL)**
R249 - Read/Write
Register Page: 57
Reset Value: 0000 0000 (00h)

7							0
-	-	-	-	-	STOP	ID1S	WKUP-INT

Bit 2 = **STOP:** *Stop bit.*
To enter STOP Mode, write the sequence 1,0,1 to this bit with **three consecutive write operations**. When a correct sequence is recognized, the STOP bit is set and the RCCU puts the MCU in STOP Mode. The software sequence succeeds only if the following conditions are true:

– The NMI pin is kept low,

– The WKUP-INT bit is 1,

– All unmasked pending bits are reset

– At least one mask bit is equal to 1 (at least one external wake-up line is not masked).

Otherwise the MCU cannot enter STOP mode, the program code continues executing and the STOP bit remains cleared.

The bit is reset by hardware if, while the MCU is in STOP mode, a wake-up interrupt comes from any of the unmasked wake-up lines. The bit is kept high if, during STOP mode, a rising edge on NMI pin wakes up the ST9. In this case the user should reset it by software. The STOP bit is at 1 in the four following cases (See "Wake-up Mode Selection" on page 2 for details):

– After the first write instruction of the sequence (a 1 is written to the STOP bit)

– At the end of a successful sequence (i.e. after the third write instruction of the sequence)

– The ST9 entered and exited STOP mode due to a rising edge on the NMI pin. In this case the EX_STP bit in the CLK_FLAG is at 1 (see RCCU chapter).

– The ST9 did not enter STOP mode due to the NMI pin being kept high. In this case RCCU bit EX_STP is at 0

Note: The STOP request generated by the WUIMU (that allows the ST9 to enter STOP mode) is ORed with the external $\overline{\text{STOP}}$ pin (active low). This means that if the external $\overline{\text{STOP}}$ pin is forced low, the ST9 will enter STOP mode independently of the status of the STOP bit.

WARNINGS:

– Writing the sequence 1,0,1 to the STOP bit will enter STOP mode only if no other register write instructions are executed during the sequence. If Interrupt or DMA requests (which always perform register write operations) are acknowledged during the sequence, the ST9 will not enter STOP mode: the user must re-enter the sequence to set the STOP bit.

– Whenever a STOP request is issued to the MCU, a few clock cycles are needed to enter STOP mode (see RCCU chapter for further details). Hence the execution of the instruction following the STOP bit setting sequence might start before entering STOP mode: if such instruction performs a register write operation, the ST9 will not enter in STOP mode. In order to avoid to execute register write instructions after a correct STOP bit setting sequence and before entering the STOP mode, it is mandatory to execute 3 NOP instructions after the STOP bit setting sequence.

Bit 1 = **ID1S:** *Interrupt Channel INTD1 Source.*
This bit is set and cleared by software.
0: INT7 external interrupt source selected, excluding wake-up line interrupt requests
1: The 16 external wake-up lines enabled as interrupt sources, replacing the INT7 external pin function

WARNING: To avoid spurious interrupt requests on the INTD1 channel due to changing the interrupt source, use this procedure to modify the ID1S bit:

1. Mask the INTD1 interrupt channel (bit 7 of register EIMR - R244, Page 0 - reset to 0).
2. Program the ID1S bit as needed.
3. Clear the IPD1 interrupt pending bit (bit 7 of register EIPR - R243, Page 0)
4. Remove the mask on INTD1 (bit EIMR.7=1).

Bit 0 = **WKUP-INT:** *Wakeup Interrupt.*
This bit is set and cleared by software.
0: The 16 external wakeup lines can be used to generate interrupt requests
1: The 16 external wake-up lines to work as wake-up sources for exiting from STOP mode

WAKE-UP / INTERRUPT LINES MANAGEMENT UNIT (Cont'd)

WAKE-UP MASK REGISTER HIGH (WUMRH)
R250 - Read/Write
Register Page: 57
Reset Value: 0000 0000 (00h)

7							0
WUM15	WUM14	WUM13	WUM12	WUM11	WUM10	WUM9	WUM8

Bit 7:0 = **WUM[15:8]:** *Wake-Up Mask bits.*
If WUMx is set, an interrupt on channel INTD1 and/or a wake-up event (depending on ID1S and WKUP-INT bits) are generated if the corresponding WUPx pending bit is set. More precisely, if WUMx=1 and WUPx=1 then:

– If ID1S=1 and WKUP-INT=1 then an interrupt on channel INTD1 and a wake-up event are generated.

– If ID1S=1 and WKUP-INT=0 only an interrupt on channel INTD1 is generated.

– If ID1S=0 and WKUP-INT=1 only a wake-up event is generated.

– If ID1S=0 and WKUP-INT=0 neither interrupts on channel INTD1 nor wake-up events are generated. Interrupt requests on channel INTD1 may be generated only from external interrupt source INT7.

If WUMx is reset, no wake-up events can be generated. Interrupt requests on channel INTD1 may be generated only from external interrupt source INT7 (resetting ID1S bit to 0).

WAKE-UP MASK REGISTER LOW (WUMRL)
R251 **-** Read/Write
Register Page: 57
Reset Value: 0000 0000 (00h)

7							0
WUM7	WUM6	WUM5	WUM4	WUM3	WUM2	WUM1	WUM0

Bit 7:0 = **WUM[7:0]:** *Wake-Up Mask bits.*
If WUMx is set, an interrupt on channel INTD1 and/or a wake-up event (depending on ID1S and WKUP-INT bits) are generated if the corresponding WUPx pending bit is set. More precisely, if WUMx=1 and WUPx=1 then:

– If ID1S=1 and WKUP-INT=1 then an interrupt on channel INTD1 and a wake-up event are generated.

– If ID1S=1 and WKUP-INT=0 only an interrupt on channel INTD1 is generated.

– If ID1S=0 and WKUP-INT=1 only a wake-up event is generated.

– If ID1S=0 and WKUP-INT=0 neither interrupts on channel INTD1 nor wake-up events are generated. Interrupt requests on channel INTD1 may be generated only from external interrupt source INT7.

If WUMx is reset, no wake-up events can be generated. Interrupt requests on channel INTD1 may be generated only from external interrupt source INT7 (resetting ID1S bit to 0).

WAKE-UP / INTERRUPT LINES MANAGEMENT UNIT (Cont'd)

WAKE-UP TRIGGER REGISTER HIGH (WUTRH)
R252 - Read/Write
Register Page: 57
Reset Value: 0000 0000 (00h)

7							0
WUT15	WUT14	WUT13	WUT12	WUT11	WUT10	WUT9	WUT8

Bit 7:0 = **WUT[15:8]:** *Wake-Up Trigger Polarity Bits*
These bits are set and cleared by software.
0: The corresponding WUPx pending bit will be set on the falling edge of the input wake-up line .
1: The corresponding WUPx pending bit will be set on the rising edge of the input wake-up line.

WAKE-UP TRIGGER REGISTER LOW (WUTRL)
R253 - Read/Write
Register Page: 57
Reset Value: 0000 0000 (00h)

7							0
WUT7	WUT6	WUT5	WUT4	WUT3	WUT2	WUT1	WUT0

Bit 7:0 = **WUT[7:0]:** *Wake-Up Trigger Polarity Bits*
These bits are set and cleared by software.
0: The corresponding WUPx pending bit will be set on the falling edge of the input wake-up line.
1: The corresponding WUPx pending bit will be set on the rising edge of the input wake-up line.

WARNING

1. As the external wake-up lines are edge triggered, no glitches must be generated on these lines.
2. If either a rising or a falling edge on the external wake-up lines occurs while writing the WUTRLH or WUTRL registers, the pending bit will not be set.

WAKE-UP PENDING REGISTER HIGH (WUPRH)
R254 - Read/Write
Register Page: 57
Reset Value: 0000 0000 (00h)

7							0
WUP15	WUP14	WUP13	WUP12	WUP11	WUP10	WUP9	WUP8

Bit 7:0 = **WUP[15:8]:** *Wake-Up Pending Bits*
These bits are set by hardware on occurrence of the trigger event on the corresponding wake-up line. They must be cleared by software. They can be set by software to implement a software interrupt.
0: No Wake-up Trigger event occurred
1: Wake-up Trigger event occured

WAKE-UP PENDING REGISTER LOW (WUPRL)
R255 - Read/Write
Register Page: 57
Reset Value: 0000 0000 (00h)

7							0
WUP7	WUP6	WUP5	WUP4	WUP3	WUP2	WUP1	WUP0

Bit 7:0 = **WUP[7:0]:** *Wake-Up Pending Bits*
These bits are set by hardware on occurrence of the trigger event on the corresponding wake-up line. They must be cleared by software. They can be set by software to implement a software interrupt.
0: No Wake-up Trigger event occurred
1: Wake-up Trigger event occured

Note: To avoid losing a trigger event while clearing the pending bits, **it is recommended** to use read-modify-write instructions (AND, BRES, BAND) to clear them.

6 ON-CHIP DIRECT MEMORY ACCESS (DMA)

6.1 INTRODUCTION

The ST9 includes on-chip Direct Memory Access (DMA) in order to provide high-speed data transfer between peripherals and memory or Register File. Multi-channel DMA is fully supported by peripherals having their own controller and DMA channel(s). Each DMA channel transfers data to or from contiguous locations in the Register File, or in Memory. The maximum number of bytes that can be transferred per transaction by each DMA channel is 222 with the Register File, or 65536 with Memory.

The DMA controller in the Peripheral uses an indirect addressing mechanism to DMA Pointers and Counter Registers stored in the Register File. This is the reason why the maximum number of transactions for the Register File is 222, since two Registers are allocated for the Pointer and Counter. Register pairs are used for memory pointers and counters in order to offer the full 65536 byte and count capability.

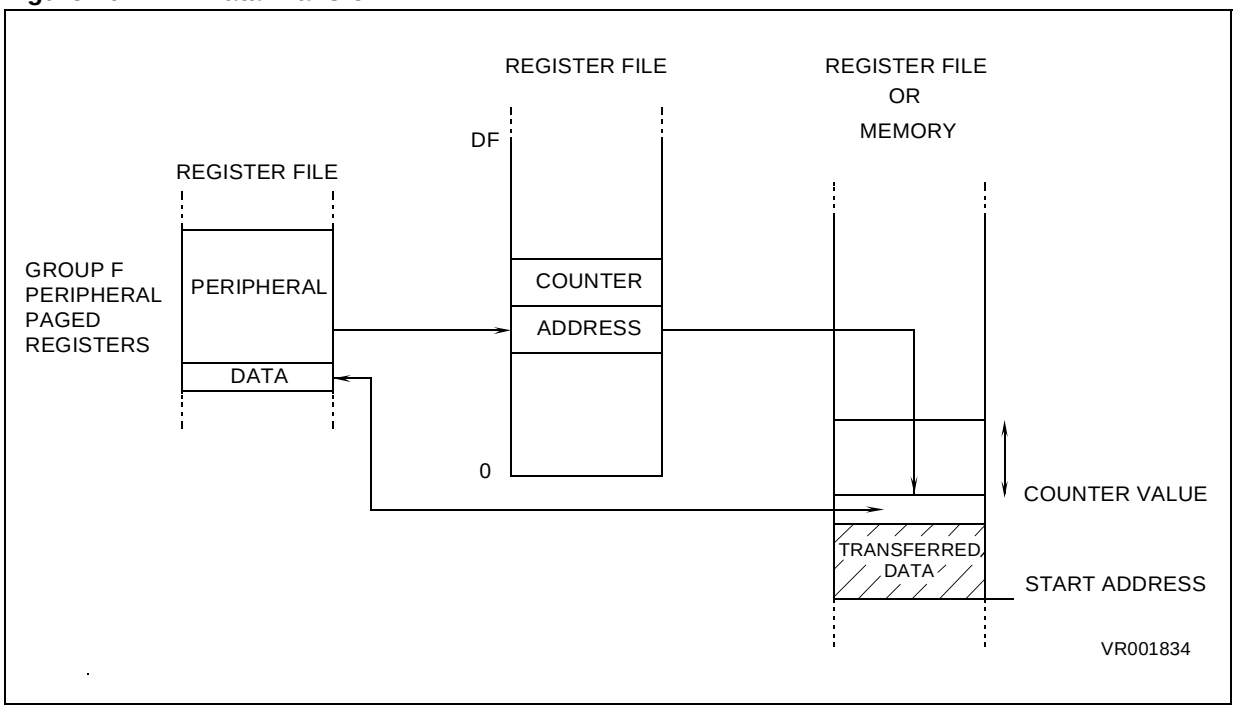
6.2 DMA PRIORITY LEVELS

The 8 priority levels used for interrupts are also used to prioritize the DMA requests, which are arbitrated in the same arbitration phase as interrupt requests. If the event occurrence requires a DMA transaction, this will take place at the end of the current instruction execution. When an interrupt and a DMA request occur simultaneously, on the same priority level, the DMA request is serviced before the interrupt.

An interrupt priority request must be strictly higher than the CPL value in order to be acknowledged, whereas, for a DMA transaction request, it must be equal to or higher than the CPL value in order to be executed. Thus only DMA transaction requests can be acknowledged when the CPL=0.

DMA requests do not modify the CPL value, since the DMA transaction is not interruptable.

Figure 40. DMA Data Transfer



6.3 DMA TRANSACTIONS

The purpose of an on-chip DMA channel is to transfer a block of data between a peripheral and the Register File, or Memory. Each DMA transfer consists of three operations:

– A load from/to the peripheral data register to/ from a location of Register File (or Memory) addressed through the DMA Address Register (or Register pair)

– A post-increment of the DMA Address Register (or Register pair)

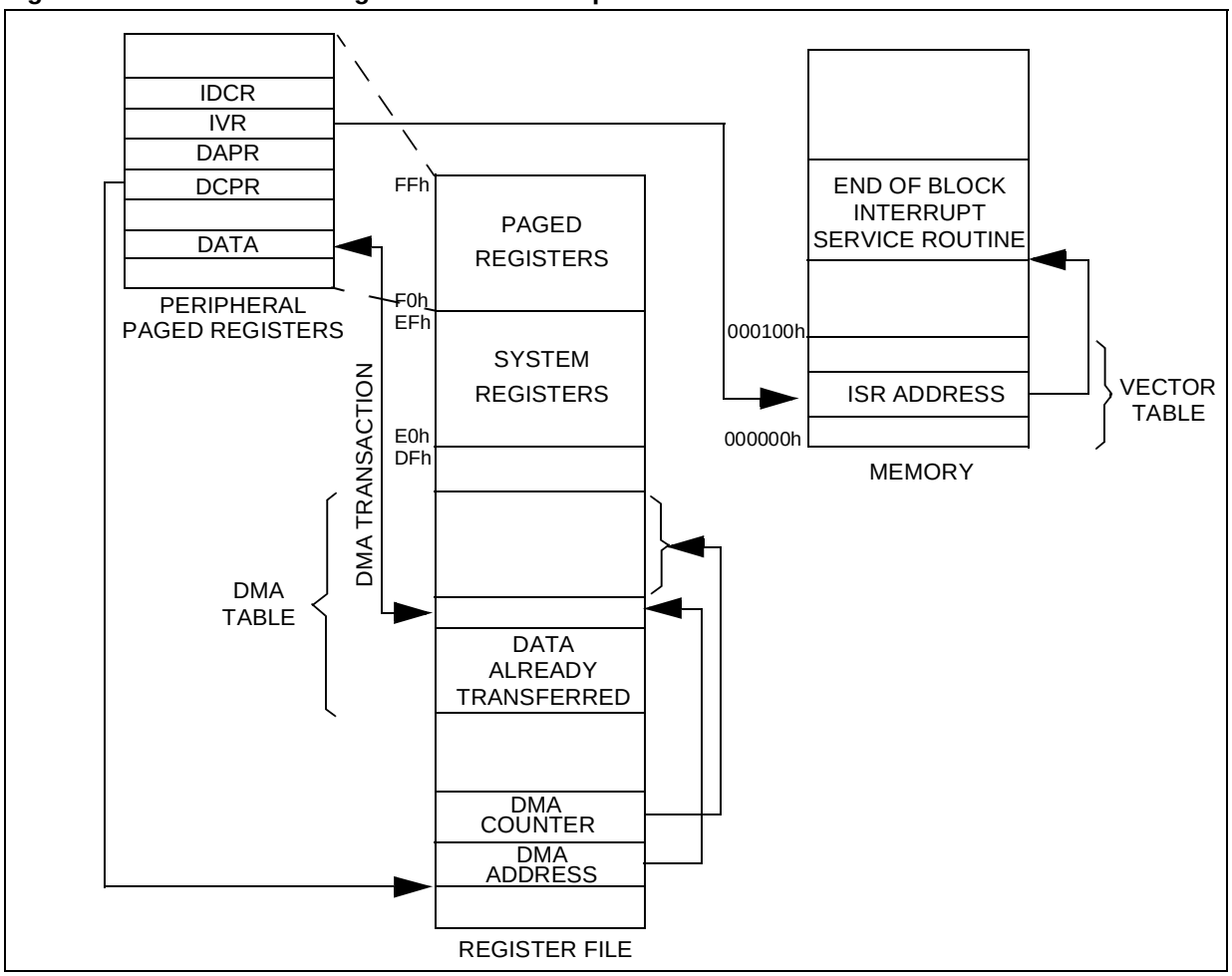
– A post-decrement of the DMA transaction counter, which contains the number of transactions that have still to be performed.

If the DMA transaction is carried out between **the peripheral and the Register File** (Figure 41), one register is required to hold the DMA Address, and one to hold the DMA transaction counter. These two registers must be located in the Register File: the DMA Address Register in the even address register, and the DMA Transaction Counter in the next register (odd address). They are pointed to by the DMA Transaction Counter Pointer Register (DCPR), located in the peripheral's paged registers. In order to select a DMA transaction with the Register File, the control bit DCPR.RM (bit 0 of DCPR) must be set.

If the transaction is made between **the peripheral and Memory**, a register pair (16 bits) is required for the DMA Address and the DMA Transaction Counter (Figure 42). Thus, two register pairs must be located in the Register File.

The DMA Transaction Counter is pointed to by the DMA Transaction Counter Pointer Register (DCPR), the DMA Address is pointed to by the DMA Address Pointer Register (DAPR),both DCPR and DAPR are located in the paged registers of the peripheral.

Figure 41. DMA Between Register File and Peripheral



DMA TRANSACTIONS (Cont'd)

When selecting the DMA transaction with memory, bit DCPR.RM (bit 0 of DCPR) must be cleared.

To select between using the ISR or the DMASR register to extend the address, (see Memory Management Unit chapter), the control bit DAPR.PS (bit 0 of DAPR) must be cleared or set respectively.

The DMA transaction Counter must be initialized with the number of transactions to perform and will be decremented after each transaction. The DMA Address must be initialized with the starting address of the DMA table and is increased after each transaction. These two registers must be located between addresses 00h and DFh of the Register File.

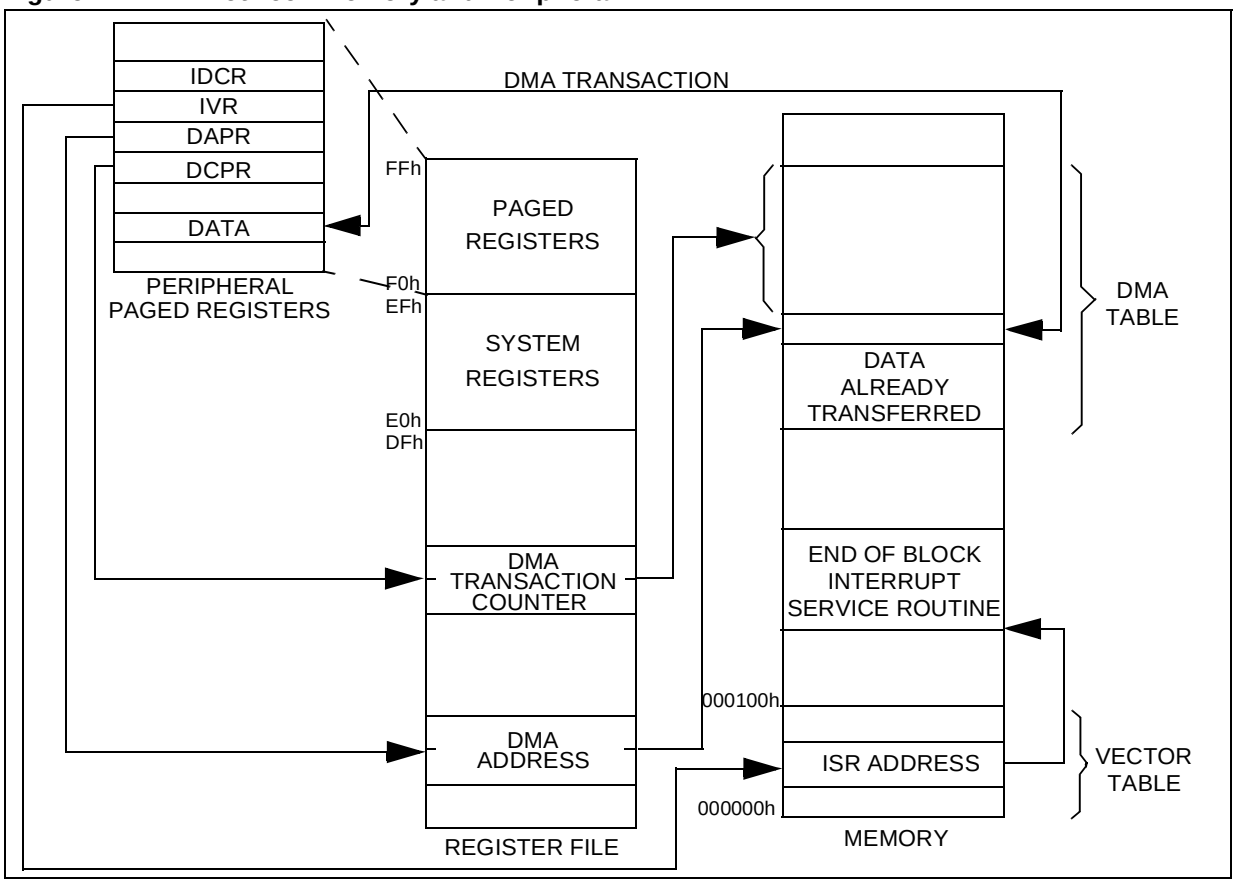
Once a DMA channel is initialized, a transfer can start. The direction of the transfer is automatically defined by the type of peripheral and programming mode.

Once the DMA table is completed (the transaction counter reaches 0 value), an Interrupt request to the CPU is generated.

When the Interrupt Pending (IDCR.IP) bit is set by a hardware event (or by software), and the DMA Mask bit (IDCR.DM) is set, a DMA request is generated. If the Priority Level of the DMA source is higher than, or equal to, the Current Priority Level (CPL), the DMA transfer is executed at the end of the current instruction. DMA transfers read/write data from/to the location pointed to by the DMA Address Register, the DMA Address register is incremented and the Transaction Counter Register is decremented. When the contents of the Transaction Counter are decremented to zero, the DMA Mask bit (DM) is cleared and an interrupt request is generated, according to the Interrupt Mask bit (End of Block interrupt). This End-of-Block interrupt request is taken into account, depending on the PRL value.

***WARNING**. DMA requests are not acknowledged if the top level interrupt service is in progress.*

Figure 42. DMA Between Memory and Peripheral



DMA TRANSACTIONS (Cont'd)

6.4 DMA CYCLE TIME

The interrupt and DMA arbitration protocol functions completely asynchronously from instruction flow.

Requests are sampled every 5 CPUCLK cycles.

DMA transactions are executed if their priority allows it.

A DMA transfer with the Register file requires 8 CPUCLK cycles.

A DMA transfer with memory requires 16 CPUCLK cycles, plus any required wait states.

6.5 SWAP MODE

An extra feature which may be found on the DMA channels of some peripherals (e.g. the MultiFunction Timer) is the Swap mode. This feature allows transfer from two DMA tables alternatively. All the DMA descriptors in the Register File are thus doubled. Two DMA transaction counters and two DMA address pointers allow the definition of two fully independent tables (they only have to belong to the same space, Register File or Memory). The DMA transaction is programmed to start on one of the two tables (say table 0) and, at the end of the block, the DMA controller automatically swaps to the other table (table 1) by pointing to the other DMA descriptors. In this case, the DMA mask (DM bit) control bit is not cleared, but the End Of Block interrupt request is generated to allow the optional updating of the first data table (table 0).

Until the swap mode is disabled, the DMA controller will continue to swap between DMA Table 0 and DMA Table 1.

6.6 DMA REGISTERS

As each peripheral DMA channel has its own specific control registers, the following register list should be considered as a general example. The names and register bit allocations shown here may be different from those found in the peripheral chapters.

DMA COUNTER POINTER REGISTER (DCPR)
Read/Write
Address set by Peripheral
Reset value: undefined

7							0
C7	C6	C5	C4	C3	C2	C1	RM

Bit 7:1 = **C[7:1]**: *DMA Transaction Counter Pointer.*

Software should write the pointer to the DMA Transaction Counter in these bits.

Bit 0 = **RM**: *Register File/Memory Selector.*
This bit is set and cleared by software.
0: DMA transactions are with memory (see also DAPR.DP)
1: DMA transactions are with the Register File

GENERIC EXTERNAL PERIPHERAL INTERRUPT AND DMA CONTROL (IDCR)
Read/Write
Address set by Peripheral
Reset value: undefined

7							0
		IP	DM	IM	PRL2	PRL1	PRL0

Bit 5 = **IP**: *Interrupt Pending*.

This bit is set by hardware when the Trigger Event occurs. It is cleared by hardware when the request is acknowledged. It can be set/cleared by software in order to generate/cancel a pending request.
0: No interrupt pending
1: Interrupt pending

Bit 4 = **DM**: *DMA Request Mask*.

This bit is set and cleared by software. It is also cleared when the transaction counter reaches zero (unless SWAP mode is active).
0: No DMA request is generated when IP is set.
1: DMA request is generated when IP is set

Bit 3 = **IM**: *End of block Interrupt Mask*.
This bit is set and cleared by software.
0: No End of block interrupt request is generated when IP is set
1: End of Block interrupt is generated when IP is set. DMA requests depend on the DM bit value as shown in the table below.

DM	IM	Meaning
1	0	A DMA request generated without End of Block interrupt when IP=1
1	1	A DMA request generated with End of Block interrupt when IP=1
0	0	No End of block interrupt or DMA request is generated when IP=1
0	1	An End of block Interrupt is generated without associated DMA request (not used)

Bit 2:0 = **PRL[2:0]**: *Source Priority Level*.
These bits are set and cleared by software. Refer to Section 6.2 DMA PRIORITY LEVELS for a description of priority levels.

PRL2	PRL1	PRL0	Source Priority Level
0	0	0	0 Highest
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7 Lowest

DMA ADDRESS POINTER REGISTER (DAPR)
Read/Write
Address set by Peripheral
Reset value: undefined

7							0
A7	A6	A5	A4	A3	A2	A1	PS

Bit 7:1 = **A[7:1]**: *DMA Address Register(s) Pointer*

Software should write the pointer to the DMA Address Register(s) in these bits.

Bit 0 = **PS**: *Memory Segment Pointer Selector*:
This bit is set and cleared by software. It is only meaningful if DCPR.RM=0.
0: The ISR register is used to extend the address of data transferred by DMA (see MMU chapter).
1: The DMASR register is used to extend the address of data transferred by DMA (see MMU chapter).

7 RESET AND CLOCK CONTROL UNIT (RCCU)

7.1 INTRODUCTION

The Reset and Clock Control Unit (RCCU) comprises two distinct sections:

– The Clock Control Unit, which generates and manages the internal clock signals.

– The Reset/Stop Manager, which detects and flags Hardware, Software and Watchdog generated resets.

On ST9 devices where the external Stop pin is available, this circuit also detects and manages the externally triggered Stop mode, during which all oscillators are frozen in order to achieve the lowest possible power consumption.

7.2 CLOCK CONTROL UNIT

The Clock Control Unit generates the internal clocks for the CPU core (CPUCLK) and for the on-chip peripherals (INTCLK). The Clock Control Unit may be driven by an external crystal circuit, connected to the OSCIN and OSCOUT pins, or by an external pulse generator, connected to OSCIN (see Figure 49 and Figure 51). A low frequency external clock may be connected to the CK_AF pin, and this clock source may be selected when low power operation is required.

7.2.1 Clock Control Unit Overview

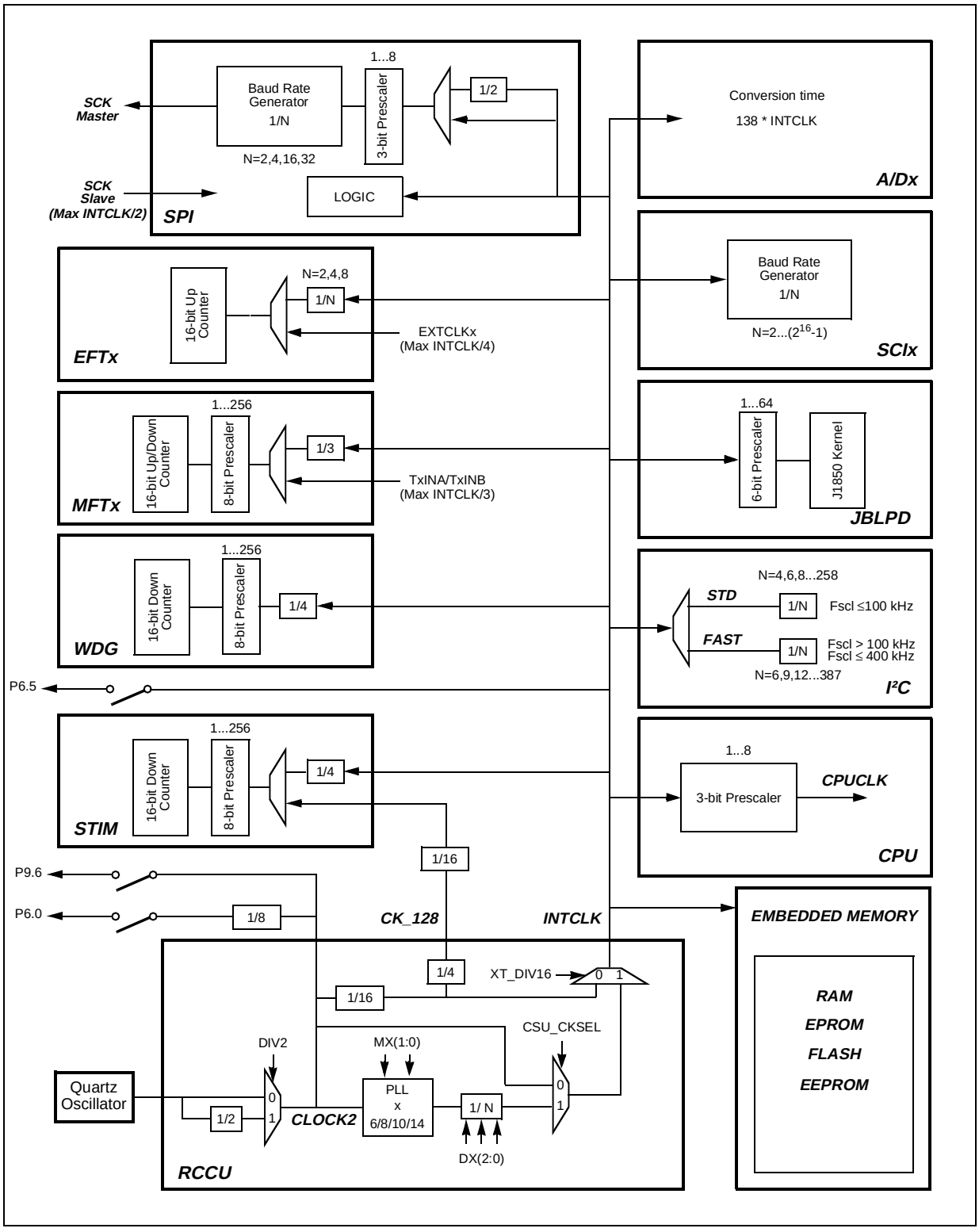
As shown in Figure 43 a programmable divider can divide the CLOCK1 input clock signal by two. In practice, the divide-by-two is virtually always used in order to ensure a 50% duty cycle signal to the PLL multiplier circuit. The resulting signal, CLOCK2, is the reference input clock to the programmable Phase Locked Loop frequency multiplier, which is capable of multiplying the clock frequency by a factor of 6, 8, 10 or 14; the multiplied clock is then divided by a programmable divider, by a factor of 1 to 7. By this means, the ST9 can operate with cheaper, medium frequency (3-5 MHz) crystals, while still providing a high frequency internal clock for maximum system performance; the range of available multiplication and division factors allow a great number of operating clock frequencies to be derived from a single crystal frequency.

For low power operation, especially in Wait for Interrupt mode, the Clock Multiplier unit may be turned off, whereupon the output clock signal may be programmed as CLOCK2 divided by 16. For further power reduction, a low frequency external clock connected to the CK_AF pin may be selected, whereupon the crystal controlled main oscillator may be turned off.

The internal system clock, INTCLK, is routed to all on-chip peripherals, as well as to the programmable Clock Prescaler Unit which generates the clock for the CPU core (CPUCLK). (See Figure 43)

The Clock Prescaler is programmable and can slow the CPU clock by a factor of up to 8, allowing the programmer to reduce CPU processing speed, and thus power consumption, while maintaining a high speed clock to the peripherals. This is particularly useful when little actual processing is being done by the CPU and the peripherals are doing most of the work.

Figure 43. ST92F120 Clock Distribution Diagram



7.3 CLOCK MANAGEMENT

The various programmable features and operating modes of the CCU are handled by four registers:

– **MODER** (Mode Register)
This is a System Register (R235, Group E).

The input clock divide-by-two and the CPU clock prescaler factors are handled by this register.

– **CLKCTL** (Clock Control Register)
This is a Paged Register (R240, Page 55).

The low power modes and the interpretation of the HALT instruction are handled by this register.

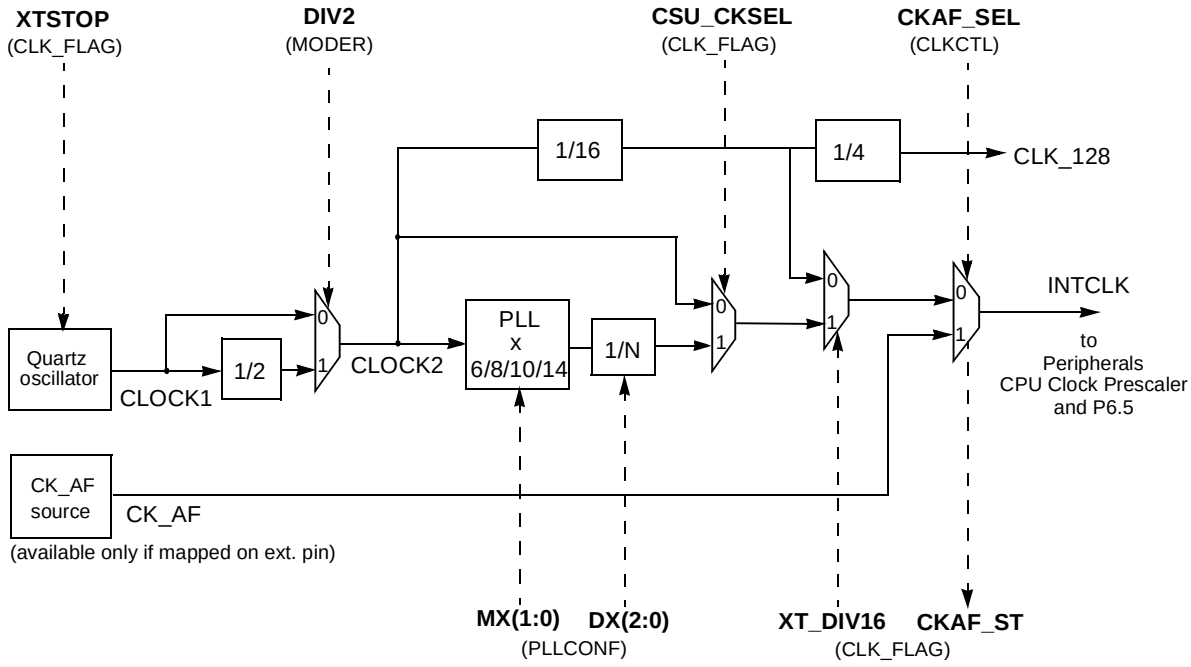
– **CLK_FLAG** (Clock Flag Register)
This is a Paged Register (R242, Page 55).

This register contains various status flags, as well as control bits for clock selection.

– **PLLCONF** (PLL Configuration Register)
This is a Paged Register (R246, Page 55).

The PLL multiplication and division factors are programmed in this register.

Figure 44. Clock Control Unit Programming



Wait for Interrupt and Low Power Modes:

LPOWFI (CLKCTL) selects Low Power operation automatically on entering WFI mode.
WFI_CKSEL (CLKCTL) selects the CK_AF clock automatically, if present, on entering WFI mode.
XTSTOP (CLK_FLAG) automatically stops the Xtal oscillator when the CK_AF clock is present and selected.

CLOCK MANAGEMENT (Cont'd)

7.3.1 PLL Clock Multiplier Programming

The CLOCK1 signal generated by the oscillator drives a programmable divide-by-two circuit. If the DIV2 control bit in MODER is set (Reset Condition), CLOCK2, is equal to CLOCK1 divided by two; if DIV2 is reset, CLOCK2 is identical to CLOCK1. Since the input clock to the Clock Multiplier circuit requires a 50% duty cycle for correct PLL operation, the divide by two circuit should be enabled when a crystal oscillator is used, or when the external clock generator does not provide a 50% duty cycle. In practice, the divide-by-two is virtually always used in order to ensure a 50% duty cycle signal to the PLL multiplier circuit.

When the PLL is active, it multiplies CLOCK2 by 6, 8, 10 or 14, depending on the status of the MX0 -1 bits in PLLCONF. The multiplied clock is then divided by a factor in the range 1 to 7, determined by the status of the DX0-2 bits; when these bits are programmed to 111, the PLL is switched off.

Following a RESET phase, programming bits DX0-2 to a value different from 111 will turn the PLL on. To select the multiplier clock, set the CSU_CKSEL bit in the CLK_FLAG Register after allowing a stabilisation period for the PLL.

Care is required, when programming the PLL multiplier and divider factors, not to exceed the maximum permissible operating frequency for INTCLK, according to supply voltage, as reported in Electrical Characteristics section.

The ST9 being a static machine, there is no lower limit for INTCLK. However, some peripherals have their own minimum internal clock frequency limit below which the functionality is not guaranteed.

7.3.2 CPU Clock Prescaling

The system clock, INTCLK, which may be the output of the PLL clock multiplier, CLOCK2, CLOCK2/16 or CK_AF, drives a programmable prescaler which generates the basic time base, CPUCLK, for the instruction executer of the ST9 CPU core. This allows the user to slow down program execution during non processor intensive routines, thus reducing power dissipation.

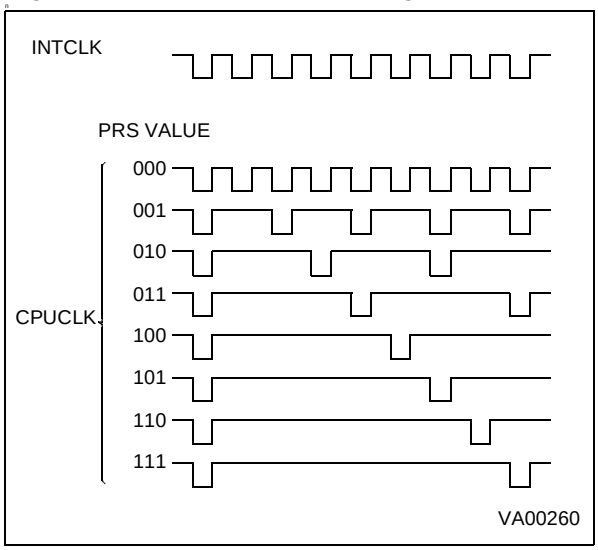
The internal peripherals are not affected by the CPUCLK prescaler and continue to operate at the full INTCLK frequency. This is particularly useful when little processing is being done and the peripherals are doing most of the work.

The prescaler divides the input clock by the value programmed in the control bits PRS2,1,0 in the MODER register. If the prescaler value is zero, no prescaling takes place, thus CPUCLK has the same period and phase as INTCLK. If the value is different from 0, the prescaling is equal to the value plus one, ranging thus from two (PRS2,1,0 = 1) to eight (PRS2,1,0 = 7).

The clock generated is shown in Figure 45, and it will be noted that the prescaling of the clock does not preserve the 50% duty cycle, since the high level is stretched to replace the missing cycles.

This is analogous to the introduction of wait cycles for access to external memory. When External Memory Wait or Bus Request events occur, CPUCLK is stretched at the high level for the whole period required by the function.

Figure 45. CPU Clock Prescaling



CLOCK MANAGEMENT (Cont'd)

7.3.3 Peripheral Clock

The system clock, INTCLK, which may be the output of the PLL clock multiplier, CLOCK2, CLOCK2/16 or CK_AF, is also routed to all ST9 on-chip peripherals and acts as the central timebase for all timing functions.

7.3.4 Low Power Modes

The user can select an automatic slowdown of clock frequency during Wait for Interrupt operation, thus idling in low power mode while waiting for an interrupt. In WFI operation the clock to the CPU core is stopped, thus suspending program execution, while the clock to the peripherals may be programmed as described in the following paragraphs. Two examples of Low Power operation in WFI are illustrated in Figure 46 and Figure 47.

Providing that low power operation during Wait for Interrupt is enabled (by setting the LPOWFI bit in the CLKCTL Register), as soon as the CPU executes the WFI instruction, the PLL is turned off and the system clock will be forced to CLOCK2 divided by 16, or to the external low frequency clock, CK_AF, if this has been selected by setting WFI_CKSEL, and providing CKAF_ST is set, thus indicating that the external clock is selected and actually present on the CK_AF pin.

If the external clock source is used, the crystal oscillator may be stopped by setting the XTSTOP bit, providing that the CK_AK clock is present and selected, indicated by CKAF_ST being set. The crystal oscillator will be stopped automatically on entering WFI if the WFI_CKSEL bit has been set. It should be noted that selecting a non-existent CK_AF clock source is impossible, since such a selection requires that the auxiliary clock source be actually present and selected. In no event can a non-existent clock source be selected inadvertently.

It is up to the user program to switch back to a faster clock on the occurrence of an interrupt, taking care to respect the oscillator and PLL stabilisation delays, as appropriate.

It should be noted that any of the low power modes may also be selected explicitly by the user program even when not in Wait for Interrupt mode, by setting the appropriate bits.

7.3.5 Interrupt Generation

System clock selection modifies the CLKCTL and CLK_FLAG registers.

The clock control unit generates an external interrupt request when CK_AF and CLOCK2/16 are selected or deselected as system clock source, as well as when the system clock restarts after a hardware stop (when the STOP MODE feature is available on the specific device). This interrupt can be masked by resetting the INT_SEL bit in the CLKCTL register. Note that this is the only case in the ST9 where an interrupt is generated with a high to low transition.

Table 20. Summary of Operating Modes using main Crystal Controlled Oscillator

MODE	INTCLK	CPUCLK	DIV2	PRS0-2	CSU_CKSEL	MX0-1	DX2-0	LPOWFI	XT_DIV16
PLL x BY 14	XTAL/2 x (14/D)	INTCLK/N	1	N-1	1	1 0	D-1	X	1
PLL x BY 10	XTAL/2 x (10/D)	INTCLK/N	1	N-1	1	0 0	D-1	X	1
PLL x BY 8	XTAL/2 x (8/D)	INTCLK/N	1	N-1	1	1 1	D-1	X	1
PLL x BY 6	XTAL/2 x (6/D)	INTCLK/N	1	N-1	1	0 1	D-1	X	1
SLOW 1	XTAL/2	INTCLK/N	1	N-1	X	X	111	X	1
SLOW 2	XTAL/32	INTCLK/N	1	N-1	X	X	X	X	0
WAIT FOR INTERRUPT	If LPOWFI=0, no changes occur on INTCLK ,but CPUCLK is stopped anyway.								
LOW POWER WAIT FOR INTERRUPT	XTAL/32	STOP	1	X	X	X	X	1	1
RESET	XTAL/2	INTCLK	1	0	0	00	111	0	1
EXAMPLE XTAL=4.4 MHz	2.2*10/2 = 11MHz	11MHz	1	0	1	00	001	X	1

Figure 46. Example of Low Power mode programming in WFI using CK_AF external clock

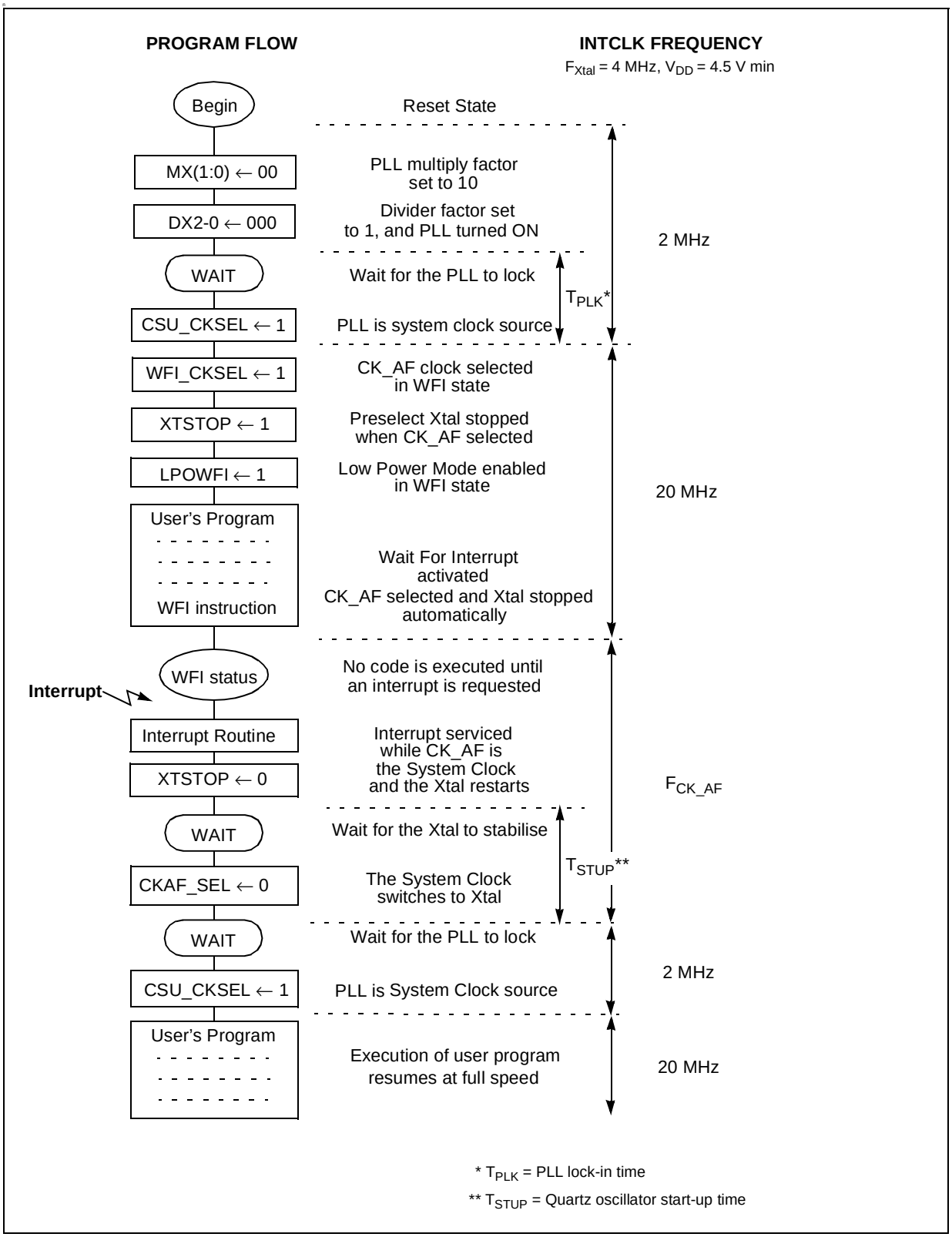
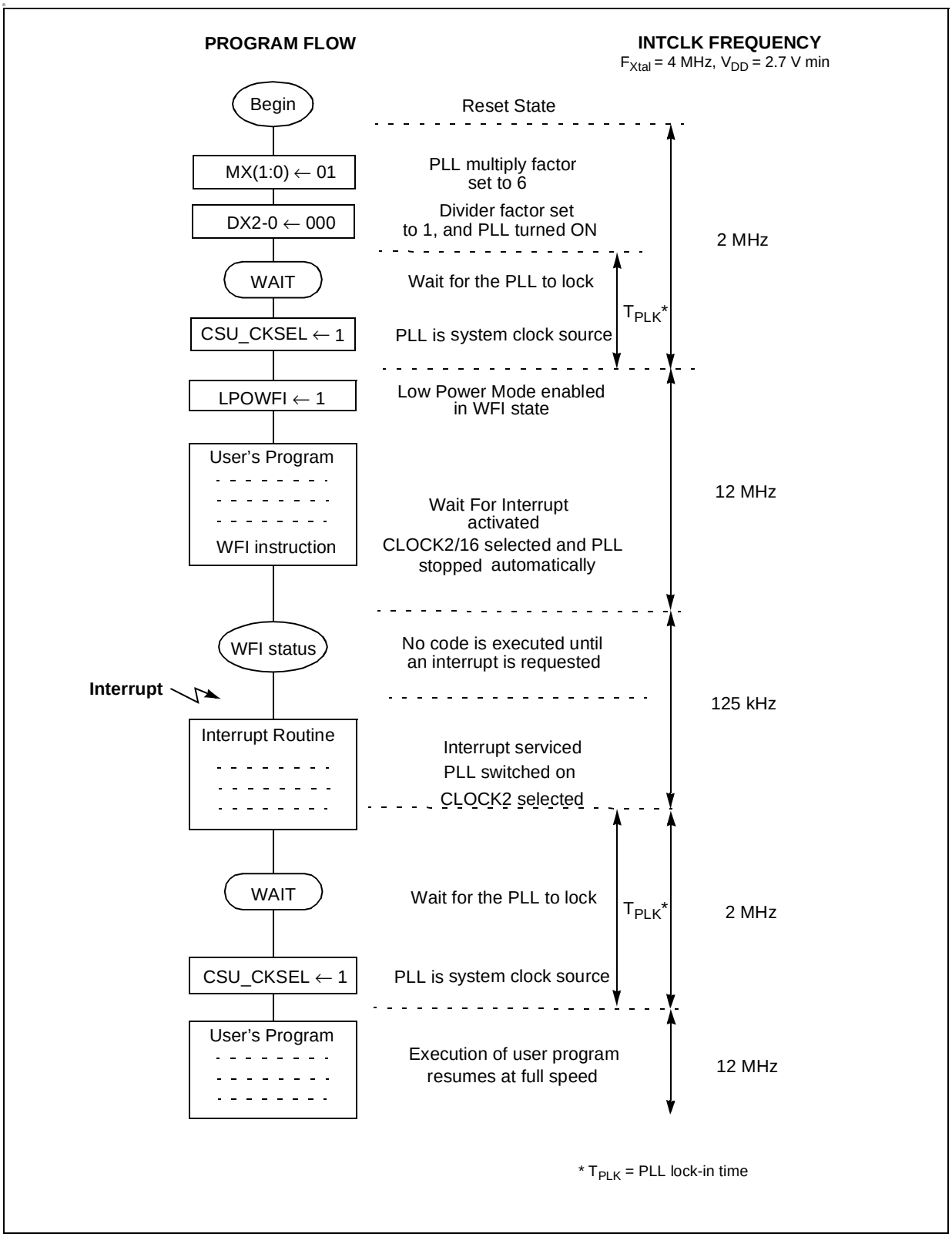


Figure 47. Example of Low Power mode programming in WFI using CLOCK2/16



7.4 CLOCK CONTROL REGISTERS

MODE REGISTER (MODER)
R235 - Read/Write
System Register
Reset Value: 1110 0000 (E0h)

7							0
-	-	DIV2	PRS2	PRS1	PRS0	-	-

***Note**: *This register contains bits which relate to other functions; these are described in the chapter dealing with Device Architecture. Only those bits relating to Clock functions are described here.*

Bit 5 = **DIV2**: *OSCIN Divided by 2*.
This bit controls the divide by 2 circuit which operates on the OSCIN Clock.
0: No division of the OSCIN Clock
1: OSCIN clock is internally divided by 2

Bits 4:2 = **PRS[2:0]**: *Clock Prescaling*.
These bits define the prescaler value used to prescale CPUCLK from INTCLK. When these three bits are reset, the CPUCLK is not prescaled, and is equal to INTCLK; in all other cases, the internal clock is prescaled by the value of these three bits plus one.

CLOCK CONTROL REGISTER (CLKCTL)
R240 - Read Write
Register Page: 55

Reset Value: 0000 0000 (00h)

7							0
INT_SEL	0	0	0	SRESEN	CKAF_SEL	WFI_CKSEL	LPOWFI

Bit 7 = **INT_SEL**: *Interrupt Selection*.
0: The external interrupt channel input signal is selected (Reset state)
1: Select the internal RCCU interrupt as the source of the interrupt request

Bits 4:6 = **Reserved for test purposes**
Must be kept reset for normal operation.

Bit 3 = **SRESEN**: *Software Reset Enable.*
0: The HALT instruction turns off the quartz, the PLL and the CCU
1: A Reset is generated when HALT is executed

Bit 2 = **CKAF_SEL**: *Alternate Function Clock Select*.
0: CK_AF clock not selected
1: Select CK_AF clock

Note: To check if the selection has actually occurred, check that CKAF_ST is set. If no clock is present on the CK_AF pin, the selection will not occur.

Bit 1 = **WFI_CKSEL**: *WFI Clock Select*.
This bit selects the clock used in Low power WFI mode if LPOWFI = 1.
0: INTCLK during WFI is CLOCK2/16
1: INTCLK during WFI is CK_AF, providing it is present. In effect this bit sets CKAF_SEL in WFI mode

Caution: When the CK_AF is selected as Low Power WFI clock but the XTAL is not turned off (R242.4 = 0), after exiting from the WFI, CK_AF will be still selected as system clock. In this case, reset the R240.2 bit to switch back to the XT.

Bit 0 = **LPOWFI**: *Low Power mode during Wait For Interrupt*.
0: Low Power mode during WFI disabled. When WFI is executed, the CPUCLK is stopped and INTCLK is unchanged
1: The ST9 enters Low Power mode when the WFI instruction is executed. The clock during this state depends on WFI_CKSEL

CLOCK CONTROL REGISTERS (Cont'd)

CLOCK FLAG REGISTER (CLK_FLAG)
R242 -Read/Write
Register Page: 55
Reset Value: 01001000 after a Watchdog Reset
Reset Value: 00101000 after a Software Reset
Reset Value: 00001000 after a Power-On Reset

7							0
EX_ STP	WDGRES	SOFTRES	XTSTOP	XT_ DIV16	CKAF_ST	LOCK	CSU_ CKSEL

CAUTION: If this register is accessed with a logical instruction, such as AND or OR, some bits may not be set as expected.

CAUTION: If you select the CK_AF as system clock and turn off the oscillator (bits R240.2 and R242.4 at 1), and then switch back to the XT clock by resetting the R240.2 bit, you must wait for the oscillator to restart correctly (T_{STUP} refer to Electrical Characteristics section).

Bit 7 = **EX_STP**: *External Stop flag.*
This bit is set by hardware and cleared by software.
0: No External Stop condition occurred
1: External Stop condition occurred

Bit 6 = **WDGRES**: *Watchdog reset flag.*
This bit is read only.
0: No Watchdog reset occurred
1: Watchdog reset occurred

Bit 5 = **SOFTRES**: *Software Reset Flag.*
This bit is read only.
0: No software reset occurred
1: Software reset occurred (HALT instruction)

Bit 4 = **XTSTOP**: *External Stop Enable.*
0: External stop disabled
1: The Xtal oscillator will be stopped as soon as the CK_AF clock is present and selected, whether this is done explicitly by the user program, or as a result of WFI, if WFI_CKSEL has previously been set to select the CK_AF clock during WFI.

CAUTION: When the program writes '1' to the XTSTOP bit, it will still be read as 0 and is only set when the CK_AF clock is running (CKAF_ST=1). Take care, as any operation such as a subsequent AND with `1' or an OR with `0' to the XTSTOP bit will reset it and the oscillator will not be stopped even if CKAF_ST is subsequently set.

Bit 3 = **XT_DIV16**: *CLOCK/16 Selection.*

This bit is set and cleared by software. An interrupt is generated when the bit is toggled.
0: CLOCK2/16 is selected and the PLL is off
1: The input is CLOCK2 (or the PLL output depending on the value of CSU_CKSEL)

CAUTION: After this bit is modified from 0 to 1, take care that the PLL lock-in time has elapsed before setting the CSU_CKSEL bit.

Bit 2 = **CKAF_ST**: (Read Only)

If set, indicates that the alternate function clock has been selected. If no clock signal is present on the CK_AF pin, the selection will not occur. If reset, the PLL clock, CLOCK2 or CLOCK2/16 is selected (depending on bit 0).

Bit 1= **LOCK**: *PLL locked-in*
This bit is read only.
0: The PLL is turned off or not locked and cannot be selected as system clock source.
1: The PLL is locked

Bit 0 = **CSU_CKSEL**: *CSU Clock Select.*
This bit is set and cleared by software. It is also cleared by hardware when:

– bits DX[2:0] (PLLCONF) are set to 111;

– the quartz is stopped (by hardware or software);

– WFI is executed while the LPOWFI bit is set;

– the XT_DIV16 bit (CLK_FLAG) is forced to '0'.

This prevents the PLL, when not yet locked, from providing an irregular clock. Furthermore, a '0' stored in this bit speeds up the PLL's locking.

0: CLOCK2 provides the system clock
1: The PLL Multiplier provides the system clock.

NOTE: Setting the CKAF_SEL bit overrides any other clock selection. Resetting the XT_DIV16 bit overrides the CSU_CKSEL selection (see Figure 44).

CLOCK CONTROL REGISTERS (Cont'd)

PLL CONFIGURATION REGISTER (PLLCONF)
R246 - Read/Write
Register Page: 55
Reset Value: xx00x111 (xxh)

7							0
-	-	MX1	MX0	-	DX2	DX1	DX0

Bits 5:4 = **MX[1:0]**: *PLL Multiplication Factor.*
Refer to Table 21 for multiplier settings.

CAUTION: After these bits are modified, take care that the PLL lock-in time has elapsed before setting the CSU_CKSEL bit in the CLK_FLAG register.

Bits 2:0 = **DX[2:0]**: *PLL output clock divider factor.*
Refer to Table 22 for divider settings.

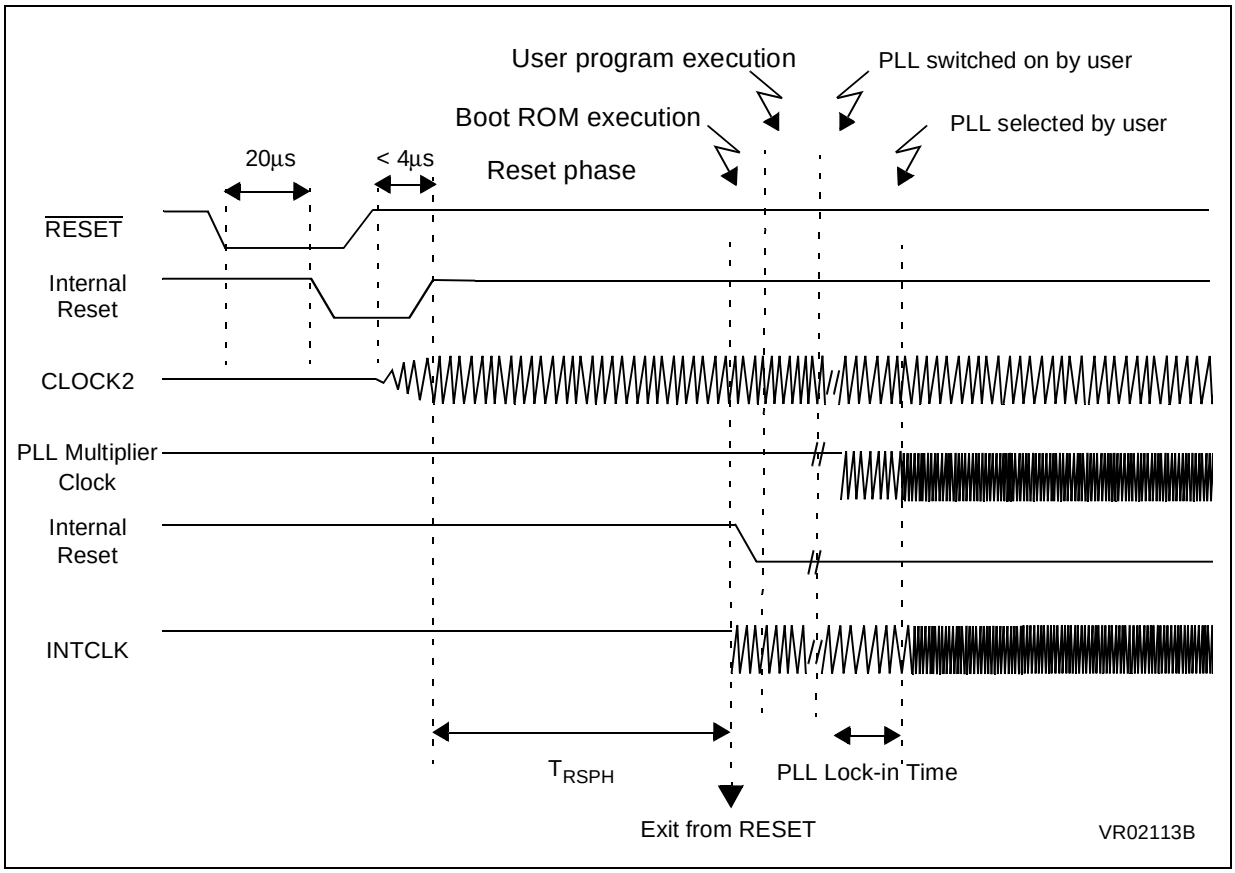
Table 21. PLL Multiplication Factors

MX1	MX0	CLOCK2 x
1	0	14
0	0	10
1	1	8
0	1	6

Table 22. PLL Divider Factors

DX2	DX1	DX0	CK
0	0	0	PLL CLOCK/1
0	0	1	PLL CLOCK/2
0	1	0	PLL CLOCK/3
0	1	1	PLL CLOCK/4
1	0	0	PLL CLOCK/5
1	0	1	PLL CLOCK/6
1	1	0	PLL CLOCK/7
1	1	1	CLOCK2 (PLL OFF, Reset State)

Figure 48. RCCU General Timing



7.5 OSCILLATOR CHARACTERISTICS

The on-chip oscillator circuit uses an inverting gate circuit with tri-state output.

OSCOUT must not be used to drive external circuits.

When the oscillator is stopped, OSCOUT goes high impedance.

In Halt mode, set by means of the HALT instruction, the parallel resistor, R, is disconnected and the oscillator is disabled, forcing the internal clock, CLOCK1, to a high level, and OSCOUT to a high impedance state.

To exit the HALT condition and restart the oscillator, an external RESET pulse is required, having a a minimum duration of 20µs, as illustrated in Figure 54.

It should be noted that, if the Watchdog function is enabled, a HALT instruction will not disable the oscillator. This to avoid stopping the Watchdog if a HALT code is executed in error. When this occurs, the CPU will be reset when the Watchdog times out or when an external reset is applied.

Figure 49. Crystal Oscillator

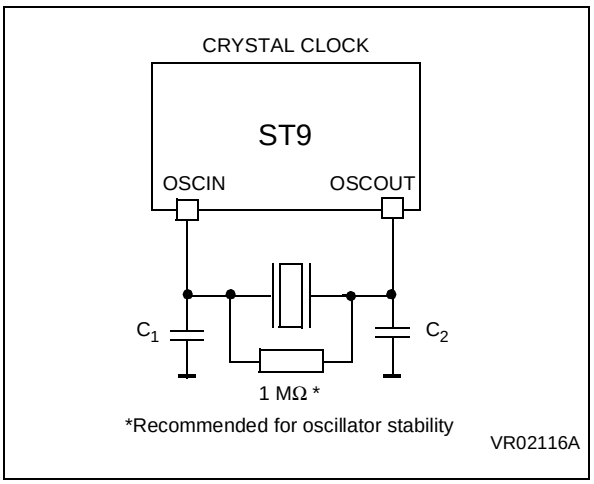


Table 23. R_S Crystal Specification

$C_1=C_2$ Freq.	33pF	22pF
5 MHz	80 ohm	130 ohm
4 MHz	130 ohm	200 ohm
3 MHz	250 ohm	-

Legend:
C_1, C_2: Maximum Total Capacitances on pins OSCIN and OSCOUT (the value includes the external capacitance tied to the pin plus the parasitic capacitance of the board and of the device)

Note: The tables are relative to the fundamental quartz crystal only (not ceramic resonator).

Figure 50. Internal Oscillator Schematic

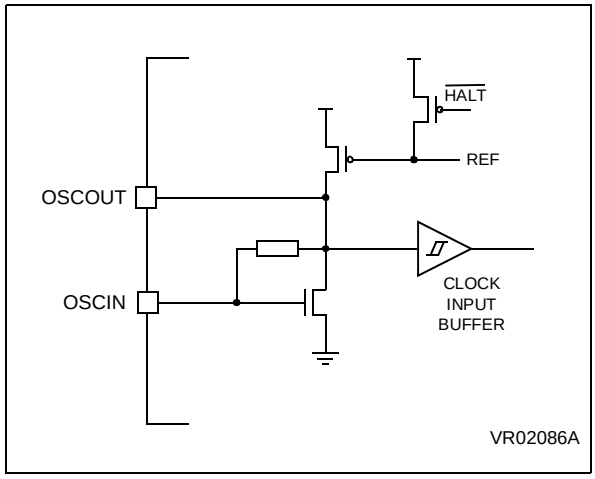
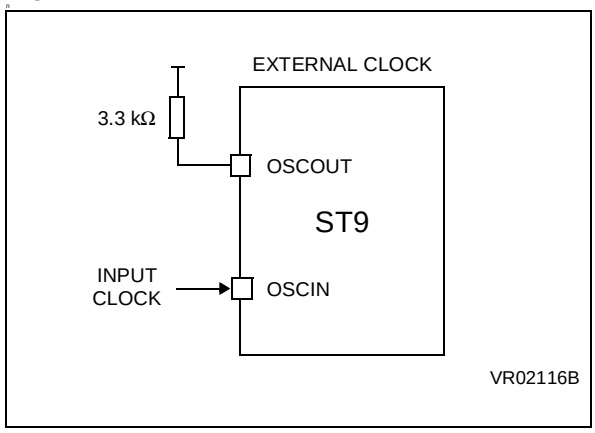


Figure 51. External Clock



CERAMIC RESONATORS

Murata Electronics CERALOCK resonators have been tested with the ST92F120 at 3, 3.68, 4 and 5 MHz. Some resonators have built-in capacitors (see Table 24).

The test circuit is shown in Figure .

Figure 52. Test Circuit

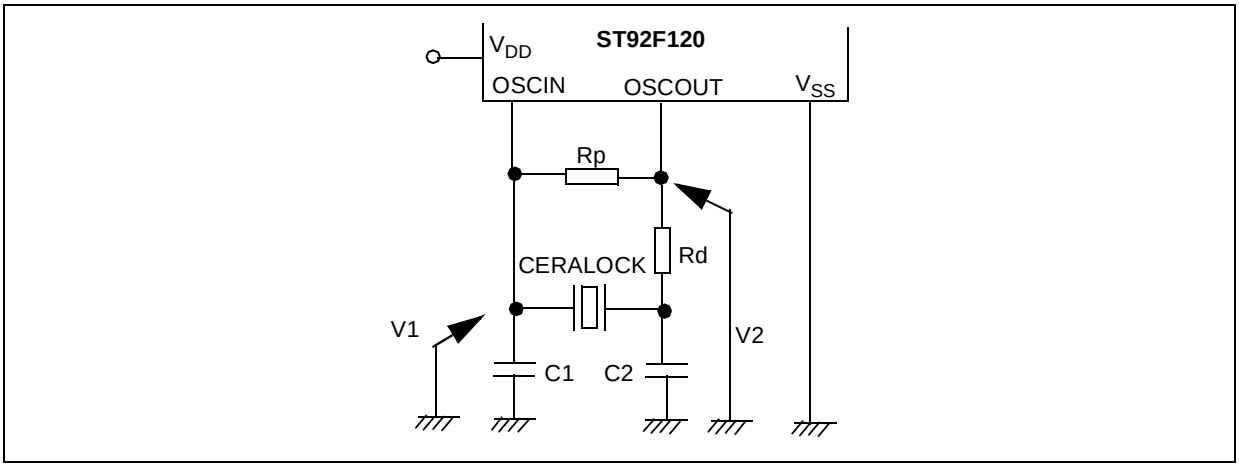


Table 24 shows the recommended conditions at different frequencies.

Table 24. Obtained Results

Freq. (MHz)	Parts Number	C1 (PF)	C2 (PF)	Rp (Ohm)	Rd (Ohm)
3	CSA3.00MG	30	30	Open	0
	CST3.00MGW	(30)	(30)	Open	0
3.68	CSA3.68MG	30	30	Open	0
	CST3.68MGW	(30)	(30)	Open	0
	CSTCC3.68MG	(15)	(15)	Open	0
4	CSA4.00MG	30	30	Open	0
	CST4.00MGW	(30)	(30)	Open	0
	CSTCC4.00MG	(15)	(15)	Open	0
5	CSA5.00MG	30	30	Open	0
	CST5.00MGW	(30)	(30)	Open	0
	CSTCC5.00MG	(15)	(15)	Open	0

Advantages of using ceramic resonators:

CST and CSTCC types have built-in loading capacitors (those with values shown in parentheses ()).

Rp is always open in the previous table because there is no need for a parallel resistor with a resonator (it is needed only with a crystal).

Test conditions:

The evaluation conditions are 4.5 to 5.5 V for the supply voltage and -40° to 105° C for the temperature range.

Caution:
The above circuit condition is for design reference only.
Recommended C1, C2 value depends on the circuit board used.

7.6 RESET/STOP MANAGER

The Reset/Stop Manager resets the MCU when one of the three following events occurs:

– A Hardware reset, initiated by a low level on the Reset pin.

– A Software reset, initiated by a HALT instruction (when enabled).

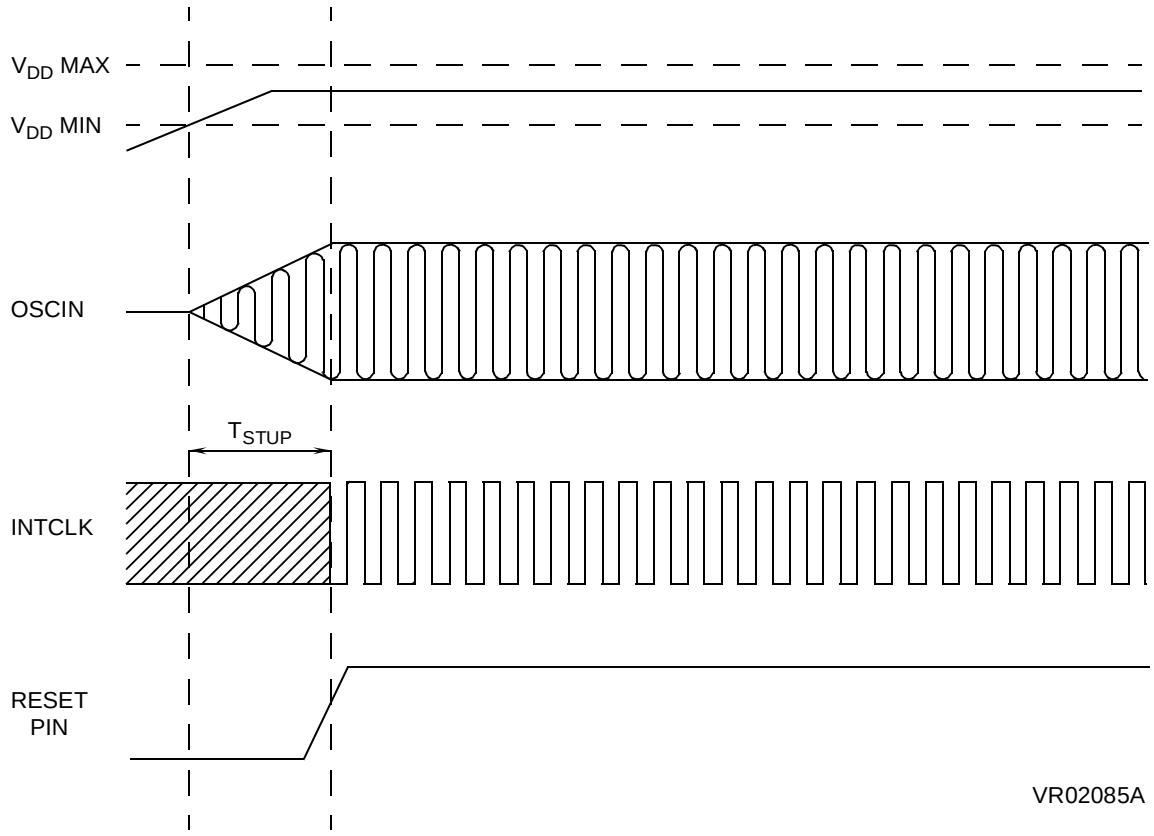
– A Watchdog end of count condition.

The event which caused the last Reset is flagged in the CLK_FLAG register, by setting the SOFTRES or the WDGRES bits respectively; a hardware initiated reset will leave both these bits reset.

The hardware reset overrides all other conditions and forces the ST9 to the reset state. During Reset, the internal registers are set to their reset values (when these reset values are defined, otherwise the register content will remain unchanged), and the I/O pins are set to Bidirectional Weak-Pull-Up or High impedance input. See Section 1.3.

Reset is asynchronous: as soon as the reset pin is driven low, a Reset cycle is initiated.

Figure 53. Oscillator Start-up Sequence and Reset Timing



Note: For T_{STUP} value refer to Oscillator Electrical Characteristics

RESET/STOP MANAGER (Cont'd)

The on-chip Timer/Watchdog generates a reset condition if the Watchdog mode is enabled (WCR.WDGEN cleared, R252 page 0), and if the programmed period elapses without the specific code (AAh, 55h) written to the appropriate register. The input pin RESET is not driven low by the on-chip reset generated by the Timer/Watchdog.

When the Reset pin goes high again, a deterministic number of oscillator clock cycles (CLOCK1) is counted (refer to T_{RSPH}) before exiting the Reset state (±1 CLOCK1 period, depending on the delay between the rising edge of the Reset pin and the first rising edge of CLOCK1). Subsequently a short Boot routine is executed from the device internal Boot memory, and control then passes to the user program.

The Boot routine sets the device characteristics and loads the correct values in the Memory Management Unit's pointer registers, so that these point to the physical memory areas as mapped in the specific device. The precise duration of this short Boot routine varies from device to device, depending on the Boot memory contents.

At the end of the Boot routine the Program Counter will be set to the location specified in the Reset Vector located in the lowest two bytes of memory.

7.6.1 Reset Pin Timing

To improve the noise immunity of the device, the Reset pin has a Schmitt trigger input circuit with hysteresis. In addition, a filter will prevent an unwanted reset in case of a single glitch of less than 50 ns on the Reset pin. The device is certain to reset if a negative pulse of more than 20µs is applied. When the reset pin goes high again, a delay of up to 4µs will elapse before the RCCU detects this rising front. From this event on, a defined number of CLOCK1 cycles (refer to T_{RSPH}) is counted before exiting the Reset state (±1CLOCK1 period depending on the delay between the positive edge the RCCU detects and the first rising edge of CLOCK1)

If the ST9 is a ROMLESS version, without on-chip program memory, the memory interface ports are set to external memory mode (i.e Alternate Function) and the memory accesses are made to external Program memory with wait cycles insertion.

Figure 54. Recommended Signal to be Applied on Reset Pin

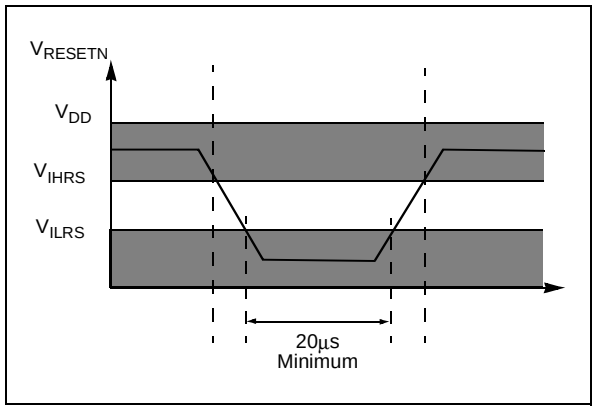
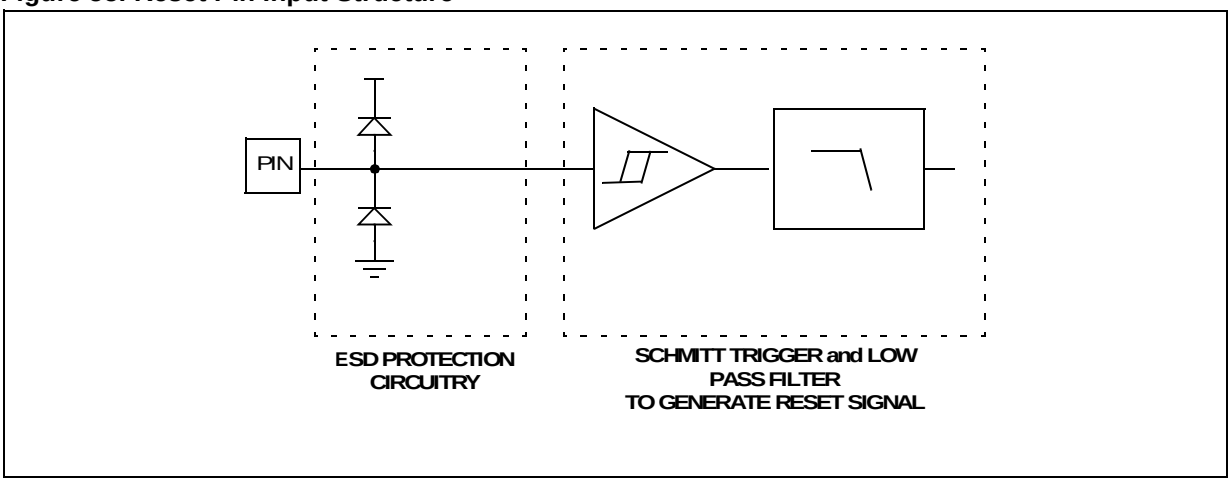


Figure 55. Reset Pin Input Structure



7.7 STOP MODE

On ST9 devices provided with an external $\overline{\text{STOP}}$ pin, the Reset/Stop Manager can also stop all oscillators without resetting the device.

To enter STOP Mode, the $\overline{\text{STOP}}$ pin must be tied low. When the $\overline{\text{STOP}}$ pin is tied high again, the MCU resumes execution of the program after a set number of CLOCK2 cycles (refer to T_{STR} in the Electrical Characteristics section), without losing the status.

Note: If STOP Mode is entered, the clock is stopped: hence, also the watchdog counter is stopped. When the ST9 exits from STOP Mode, the watchdog counter restarts from where it was before STOP Mode was entered.

8 EXTERNAL MEMORY INTERFACE (EXTMI)

8.1 INTRODUCTION

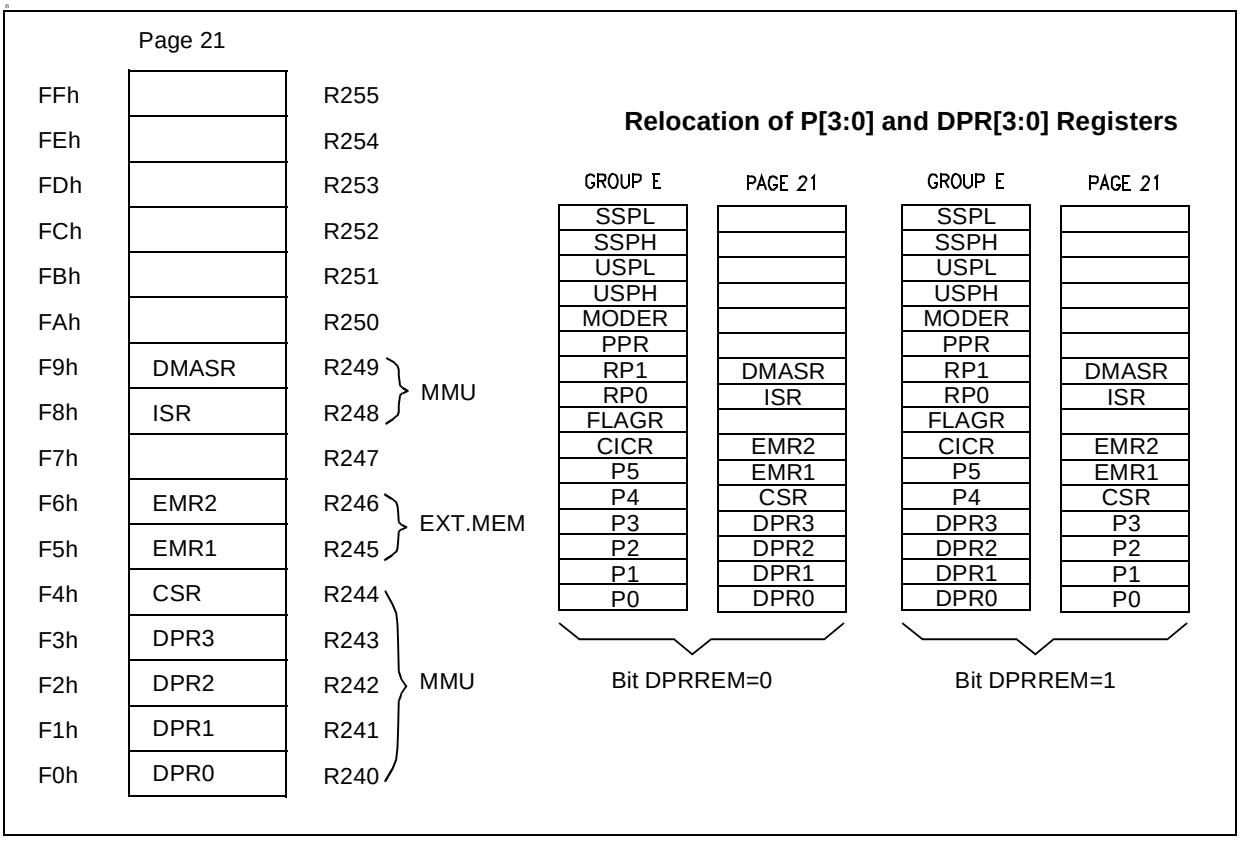
The ST9 External Memory Interface uses two registers (EMR1 and EMR2) to configure external memory accesses. Some interface signals are also affected by WCR - R252 Page 0.

If the two registers EMR1 and EMR2 are set to the proper values, the ST9+ memory access cycle is similar to that of the original ST9, with the only exception that it is composed of just two system clock phases, named T1 and T2.

During phase T1, the memory address is output on the AS falling edge and is valid on the rising edge of AS. Port0 and Port 1 maintain the address stable until the following T1 phase.

During phase T2, two forms of behavior are possible. If the memory access is a Read cycle, Port 0 pins are released in high-impedance until the next T1 phase and the data signals are sampled by the ST9 on the rising edge of DS. If the memory access is a Write cycle, on the falling edge of DS, Port 0 outputs data to be written in the external memory. Those data signals are valid on the rising edge of DS and are maintained stable until the next address is output. Note that DS is pulled low at the beginning of phase T2 only during an external memory access.

Figure 56. Page 21 Registers



8.2 EXTERNAL MEMORY SIGNALS

The access to external memory is made using the $\overline{AS}$, $\overline{DS}$, $\overline{DS2}$, $R\overline{W}$, Port 0, Port1, and $\overline{WAIT}$ signals described below.

Refer to Figure 58

8.2.1 $\overline{AS}$: Address Strobe

$\overline{AS}$ (Output, Active low, Tristate) is active during the System Clock high-level phase of each T1 memory cycle: an $\overline{AS}$ rising edge indicates that Memory Address and Read/Write Memory control signals are valid. $\overline{AS}$ is released in high-impedance during the bus acknowledge cycle or under the processor control by setting the HIMP bit (MODER.0, R235). Depending on the device $\overline{AS}$ is available as Alternate Function or as a dedicated pin.

Under Reset, $\overline{AS}$ is held high with an internal weak pull-up.

The behavior of this signal is affected by the MC, ASAF, ETO, BSZ, LAS[1:0] and UAS[1:0] bits in the EMR1 or EMR2 registers. Refer to the Register description.

8.2.2 $\overline{DS}$: Data Strobe

$\overline{DS}$ (Output, Active low, Tristate) is active during the internal clock high-level phase of each T2 memory cycle. During an external memory read cycle, the data on Port 0 must be valid before the $\overline{DS}$ rising edge. During an external memory write cycle, the data on Port 0 are output on the falling edge of $\overline{DS}$ and they are valid on the rising edge of $\overline{DS}$. When the internal memory is accessed $\overline{DS}$ is kept high during the whole memory cycle. $\overline{DS}$ is released in high-impedance during bus acknowledge cycle or under processor control by setting the HIMP bit (MODER.0, R235). Under Reset status, $\overline{DS}$ is held high with an internal weak pull-up.

The behavior of this signal is affected by the MC, DS2EN, and BSZ bits in the EMR1 register. Refer to the Register description.

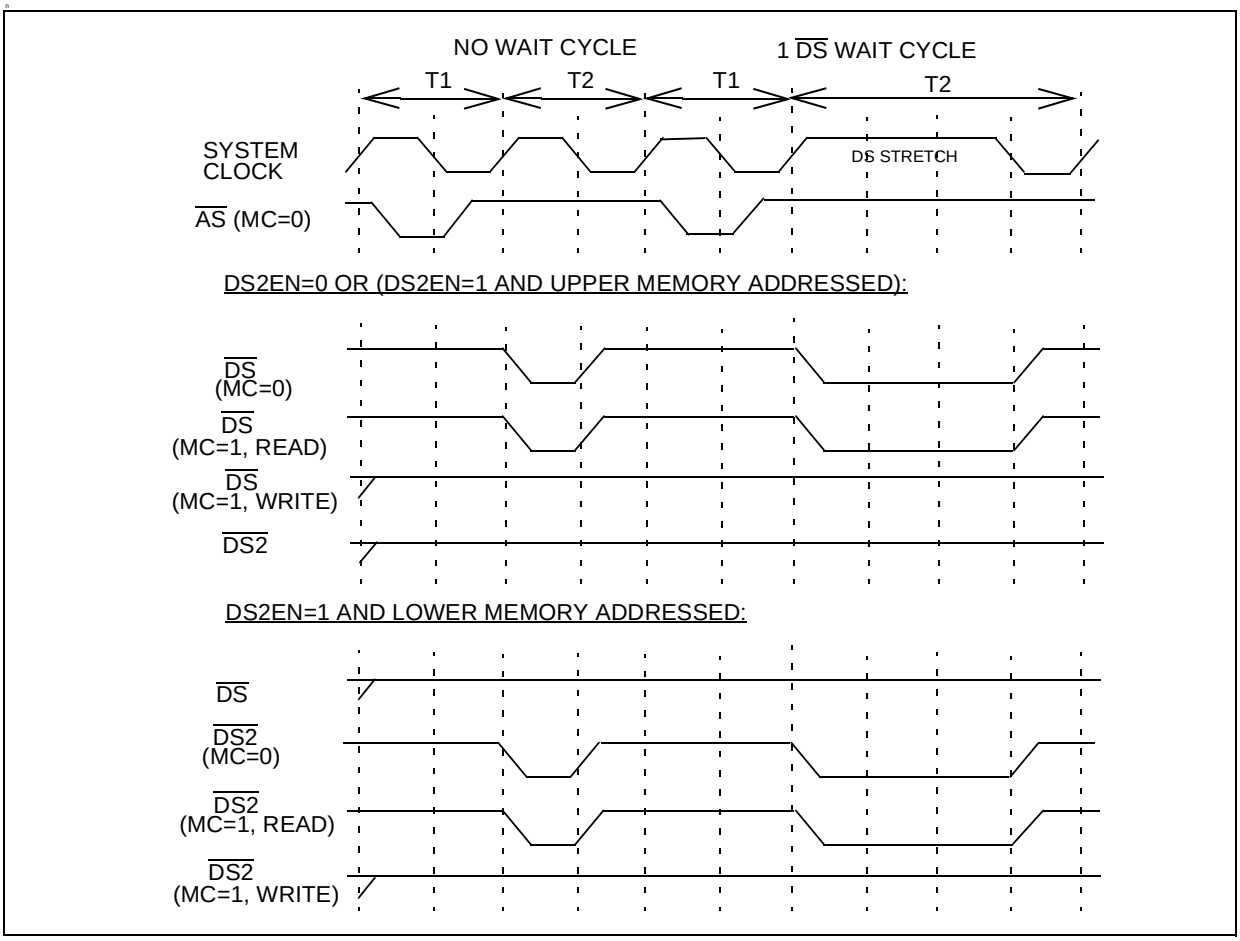
8.2.3 $\overline{DS2}$: Data Strobe 2

This additional Data Strobe pin (Alternate Function Output, Active low, Tristate) is available on some ST9 devices only. It allows two external memories to be connected to the ST9, the upper memory block (A21=1 typically RAM) and the lower memory block (A21=0 typically ROM) without any external logic. The selection between the upper and lower memory blocks depends on the A21 address pin value.

The upper memory block is controlled by the $\overline{DS}$ pin while the lower memory block is controlled by the $\overline{DS2}$ pin. When the internal memory is addressed, $\overline{DS2}$ is kept high during the whole memory cycle. $\overline{DS2}$ is released in high-impedance during bus acknowledge cycle or under processor control by setting the HIMP bit (MODER.0, R235). $\overline{DS2}$ is enabled via software as the Alternate Function output of the associated I/O port bit (refer to specific ST9 version to identify the specific port and pin).

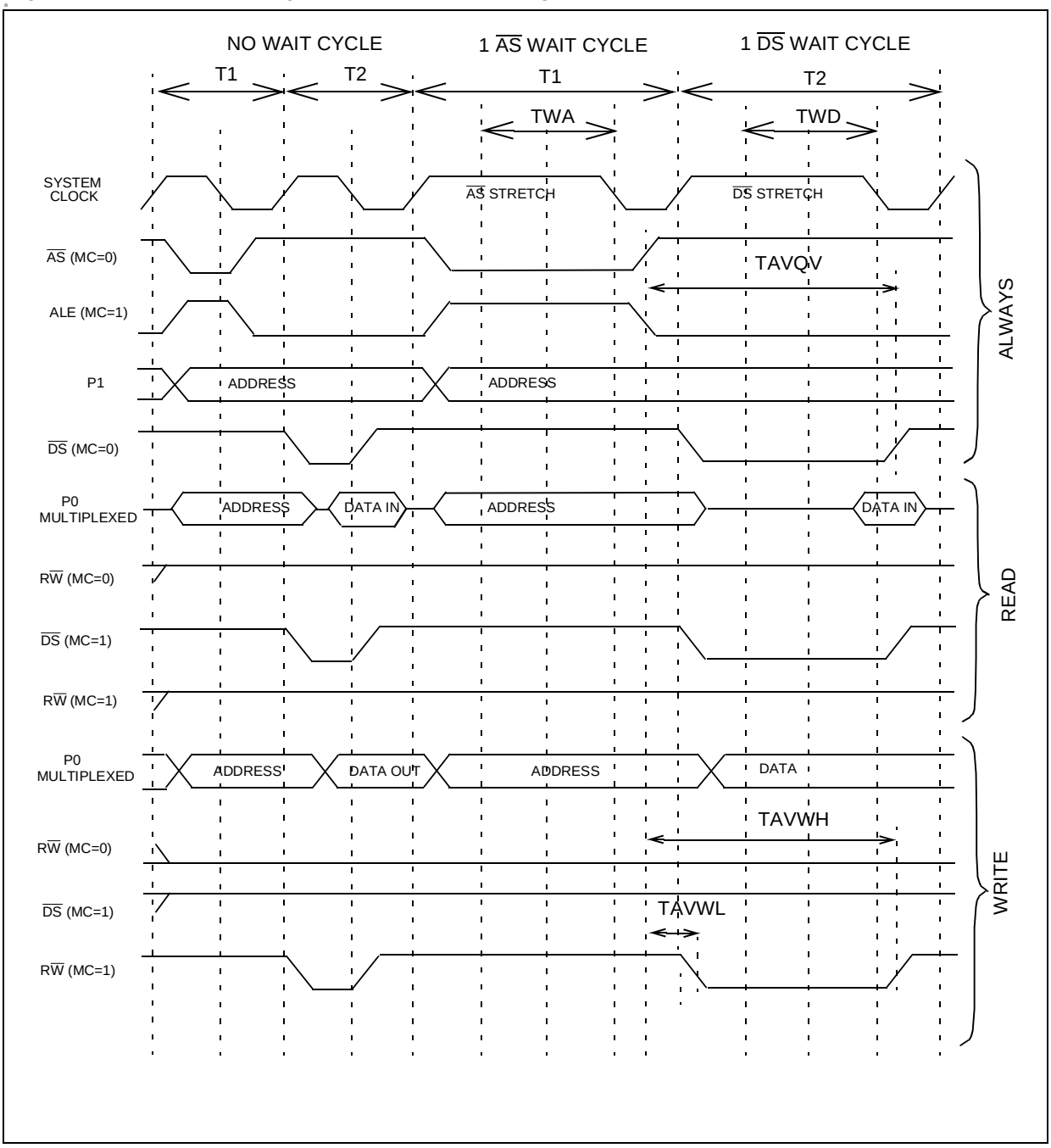
The behavior of this signal is affected by the DS2EN, and BSZ bits in the EMR1 register. Refer to the Register description.

Figure 57. Effects of DS2EN on the behavior of $\overline{DS}$ and $\overline{DS2}$



EXTERNAL MEMORY SIGNALS (Cont'd)

Figure 58. External Memory Read/Write with a Programmable Wait



EXTERNAL MEMORY SIGNALS (Cont'd)

8.2.4 $\overline{RW}$: Read/Write

$\overline{RW}$ (Alternate Function Output, Active low, Tristate) identifies the type of memory cycle: $\overline{RW}$="1" identifies a memory read cycle, $\overline{RW}$="0" identifies a memory write cycle. It is defined at the beginning of each memory cycle and it remains stable until the following memory cycle. $\overline{RW}$ is released in high-impedance during bus acknowledge cycle or under processor control by setting the HIMP bit (MODER). $\overline{RW}$ is enabled via software as the Alternate Function output of the associated I/O port bit (refer to specific ST9 device to identify the port and pin). Under Reset status, the associated bit of the port is set into bidirectional weak pull-up mode.

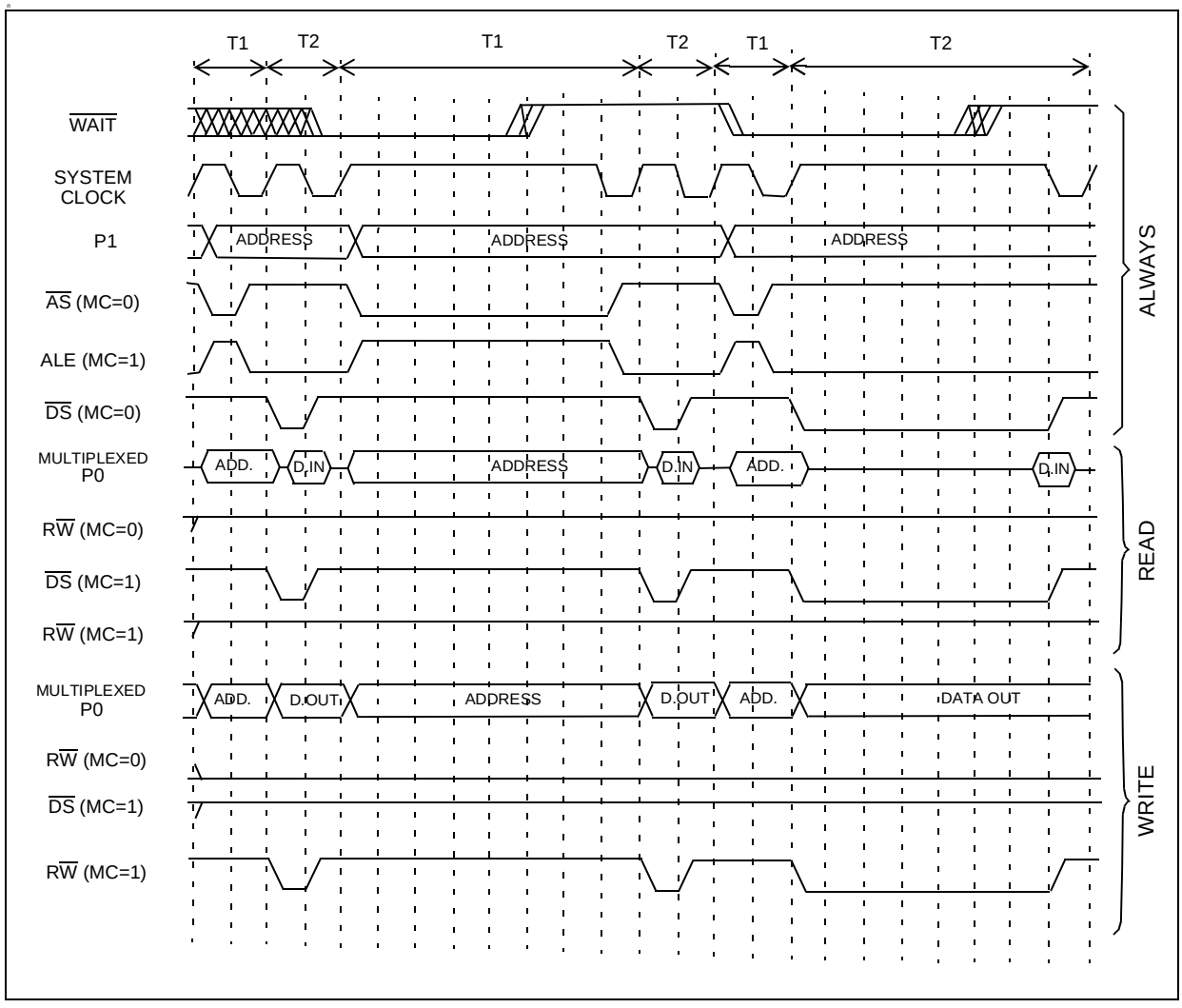
The behavior of this signal is affected by the MC, ETO and BSZ bits in the EMR1 register. Refer to the Register description.

8.2.5 $\overline{BREQ}$, $\overline{BACK}$: Bus Request, Bus Acknowledge

Note: These pins are available only on some ST9 devices (see Pin description).

$\overline{BREQ}$ (Alternate Function Input, Active low) indicates to the ST9 that a bus request has tried or is trying to gain control of the memory bus. Once enabled by setting the BRQEN bit (MODER.1, R235), $\overline{BREQ}$ is sampled with the falling edge of the processor internal clock during phase T2.

Figure 59. External Memory Read/Write Sequence with External Wait ($\overline{WAIT}$ pin)



EXTERNAL MEMORY SIGNALS (Cont'd)

Whenever it is sampled low, the System Clock is stretched and the external memory signals (AS, DS, DS2, RW, P0 and P1) are released in high-impedance. The external memory interface pins are driven again by the ST9 as soon as BREQ is sampled high.

BACK (Alternate Function Output, Active low) indicates that the ST9 has relinquished control of the memory bus in response to a bus request. BREQ is driven low when the external memory interface signals are released in high-impedance.

At MCU reset, the bus request function is disabled. To enable it, configure the I/O port pins assigned to BREQ and BACK as Alternate Function and set the BRQEN bit in the MODER register.

8.2.6 PORT 0

If Port 0 (Input/Output, Push-Pull/Open-Drain/ Weak Pull-up) is used as a bit programmable parallel I/O port, it has the same features as a regular port. When set as an Alternate Function, it is used as the External Memory interface: it outputs the multiplexed Address 8 LSB: A[7:0] /Data bus D[7:0].

8.2.7 PORT 1

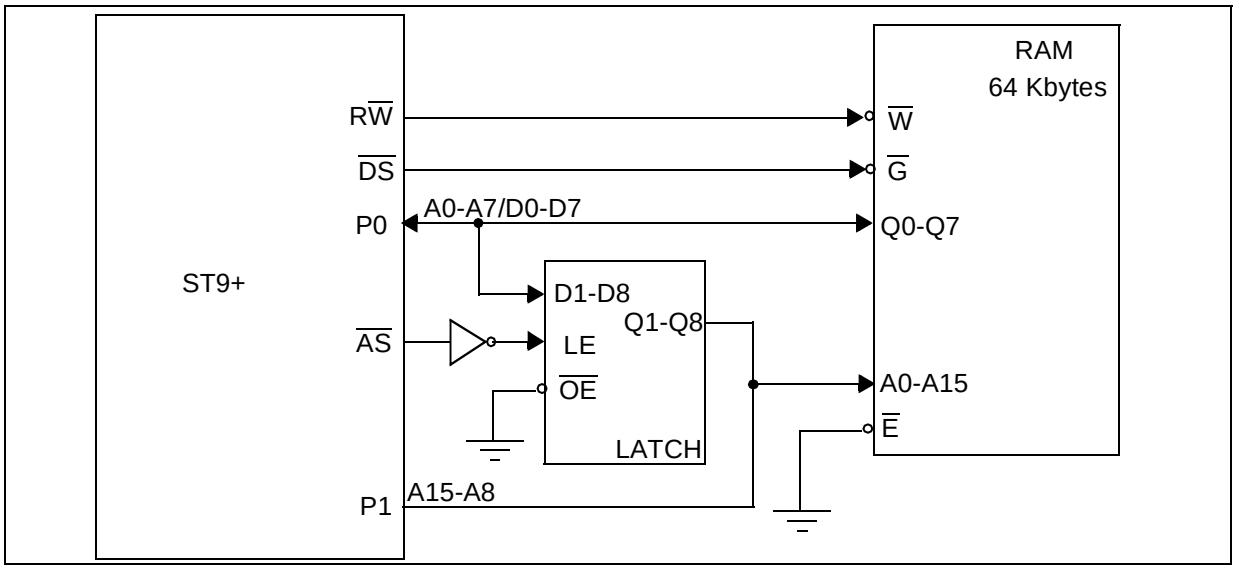
If Port 1 (Input/Output, Push-Pull/Open-Drain/ Weak Pull-up) is used as a bit programmable parallel I/O port, it has the same features as a regular port. When set as an Alternate Function, it is used as the external memory interface to provide the 8 MSB of the address A[15:8].

The behavior of the Port 0 and 1 pins is affected by the BSZ and ETO bits in the EMR1 register. Refer to the Register description.

8.2.8 WAIT: External Memory Wait

WAIT (Alternate Function Input, Active low) indicates to the ST9 that the external memory requires more time to complete the memory access cycle. If bit EWEN (EIVR) is set, the WAIT signal is sampled with the rising edge of the processor internal clock during phase T1 or T2 of every memory cycle. If the signal was sampled active, one more internal clock cycle is added to the memory cycle. On the rising edge of the added internal clock cycle, WAIT is sampled again to continue or finish the memory cycle stretching. Note that if WAIT is sampled active during phase T1 then AS is stretched, while if WAIT is sampled active during phase T2 then DS is stretched. WAIT is enabled via software as the Alternate Function input of the associated I/O port bit (refer to specific ST9 version to identify the specific port and pin). Under Reset status, the associated bit of the port is set to the bidirectional weak pull-up mode. Refer to Figure 59.

Figure 60. Application Example



8.3 REGISTER DESCRIPTION

EXTERNAL MEMORY REGISTER 1 (EMR1)
R245 - Read/Write
Register Page: 21
Reset value: 1000 0000 (80h)

7							0
x	MC	DS2EN	ASAF	x	ETO	BSZ	X

Bit 7 = Reserved.

Bit 6 = **MC**: *Mode Control*.
0: $\overline{AS}$, $\overline{DS}$ and R$\overline{W}$ pins have the standard ST9 meaning.
1: $\overline{AS}$ pin becomes ALE, Address Load Enable ($\overline{AS}$ inverted); Thus Memory Adress, Read/ Write signals are valid whenever a falling edge of ALE occurs.
$\overline{DS}$ becomes OEN, Output ENable: it has the standard ST9 meaning during external read operations, but is forced to “1” during external write operations.
R$\overline{W}$ pin becomes WEN, Write ENable: it follows the standard ST9 $\overline{DS}$ meaning during external write operations, but is forced to “1” during external read operations.

Bit 5 = **DS2EN**: *Data Strobe 2 enable*.
0: The $\overline{DS2}$ pin is forced to “1” during the whole memory cycle. In this case, access to upper or lower memory is controlled by the $\overline{DS}$ pin.
1: If the lower memory block is addressed, the $\overline{DS2}$ pin follows the standard ST9 $\overline{DS}$ meaning (if MC=0) or it becomes OEN (if MC=1). The $\overline{DS}$ pin is forced to 1 during the whole memory cycle.
If the upper memory block is used, $\overline{DS2}$ is forced to “1” during the whole memory cycle. The $\overline{DS}$ pin behaviour is not modified.
Refer to Figure 57

Bit 4 = **ASAF**: *Address Strobe as Alternate Function.*
Depending on the device, $\overline{AS}$ can be either a dedicated pin or a port Alternate Function. This bit is used only in the second case.
0: $\overline{AS}$ Alternate function disabled.
1: $\overline{AS}$ Alternate Function enabled.

Bit 2 = **ETO**: *External toggle.*
0: The external memory interface pins ($\overline{AS}$, $\overline{DS}$, $\overline{DS2}$, R$\overline{W}$, Port0, Port1) toggle only if an access to external memory is performed.
1: When the internal memory protection is disabled (mask option available on some devices only), the above pins (except $\overline{DS}$ and $\overline{DS2}$ which never toggle during internal memory accesses) toggle during both internal and external memory accesses.

Bit 1 = **BSZ**: *Bus size*.
0: All the I/O ports including the external memory interface pins use smaller, less noisy output buffers. This may limit the operation frequency of the device, unless the clock is slow enough or sufficient wait states are inserted.
1: All the I/O ports including the external memory interface pins ($\overline{AS}$, $\overline{DS}$, $\overline{DS2}$, R/$\overline{W}$, Port 0, 1) use larger, more noisy output buffers .

Bit 0 = Reserved.

***CAUTION**: External memory must be correctly addressed before and after a write operation on the EMR1 register. For example, if code is fetched from external memory using the standard ST9 external memory interface configuration (MC=0), setting the MC bit will cause the device to behave unpredictably.*

EXTERNAL MEMORY INTERFACE REGISTERS (Cont'd)

EXTERNAL MEMORY REGISTER 2 (EMR2)
R246 - Read/Write
Register Page: 21
Reset value: 0001 1111 (1Fh)

7							0
-	ENCSR	DPRREM	MEM SEL	LAS1	LAS0	UAS1	UAS0

Bit 7 = Reserved.

Bit 6 = **ENCSR**: *Enable Code Segment Register.*

This bit affects the ST9 CPU behavior whenever an interrupt request is issued.
0: The CPU works in original ST9 compatibility mode concerning stack frame during interrupts. For the duration of the interrupt service routine, ISR is used instead of CSR, and the interrupt stack frame is identical to that of the original ST9: only the PC and Flags are pushed. This avoids saving the CSR on the stack in the event of an interrupt, thus ensuring a faster interrupt response time. The drawback is that it is not possible for an interrupt service routine to perform inter-segment calls or jumps: these instructions would update the CSR, which, in this case, is not used (ISR is used instead). The code segment size for all interrupt service routines is thus limited to 64K bytes.
1: If ENCSR is set, ISR is only used to point to the interrupt vector table and to initialize the CSR at the beginning of the interrupt service routine: the old CSR is pushed onto the stack together with the PC and flags, and CSR is then loaded with the contents of ISR. In this case, iret will also restore CSR from the stack. This approach allows interrupt service routines to access the entire 4Mbytes of address space; the drawback is that the interrupt response time is slightly increased, because of the need to also save CSR on the stack. Full compatibility with the original ST9 is lost in this case, because the interrupt stack frame is different; this difference, however, should not affect the vast majority of programs.

Bit 5 = **DPRREM**: *Data Page Registers remapping*
0: The locations of the four MMU (Memory Management Unit) Data Page Registers (DPR0, DPR1, DPR2 and DPR3) are in page 21.
1: The four MMU Data Page Registers are swapped with that of the Data Registers of ports 0-3.
Refer to Figure 56

Bit 4 = **MEMSEL**: Memory Selection.

Caution: Must be kept as it is set in the TestFlash Boot Code (Reset value is 1).

Bits 3:2 = **LAS[1:0]**: *Lower memory address strobe stretch*.
These two bits contain the number of wait cycles (from 0 to 3) to add to the System Clock to stretch $\overline{AS}$ during external lower memory block accesses (MSB of 22-bit internal address=0). The reset value is 3.

EXTERNAL MEMORY INTERFACE REGISTERS (Cont'd)

Bits 1:0 = **UAS[1:0]**: *Upper memory address strobe stretch*.
These two bits contain the number of wait cycles (from 0 to 3) to add to the System Clock to stretch $\overline{AS}$ during external upper memory block accesses (MSB of 22-bit internal address=1). The reset value is 3.

CAUTION: The EMR2 register cannot be written during an interrupt service routine.

WAIT CONTROL REGISTER (WCR)
R252 - Read/Write
Register Page: 0
Reset Value: 0111 1111 (7Fh)

7							0
0	WDGEN	UDS2	UDS1	UDS0	LDS2	LDS1	LDS0

Bit 7 = Reserved, forced by hardware to 0.

Bit 6 = **WDGEN**: *Watchdog Enable.*

For a description of this bit, refer to the Timer/ Watchdog chapter.

CAUTION: Clearing this bit has the effect of setting the Timer/Watchdog to Watchdog mode. Unless this is desired, it must be set to "1".

Bits 5:3 = **UDS[2:0]**: *Upper memory data strobe stretch.*
These bits contain the number of INTCLK cycles to be added automatically to $\overline{DS}$ for external upper memory block accesses. UDS = 0 adds no additional wait cycles. UDS = 7 adds the maximum 7 INTCLK cycles (reset condition).

Bits 2:0 = **LDS[2:0]**: *Lower memory data strobe stretch.*
These bits contain the number of INTCLK cycles to be added automatically to $\overline{DS}$ or $\overline{DS2}$ (depending on the DS2EN bit of the EMR1 register) for external lower memory block accesses. LDS = 0 adds no additional wait cycles, LDS = 7 adds the maximum 7 INTCLK cycles (reset condition).

Note 1: The number of clock cycles added refers to INTCLK and NOT to CPUCLK.

Note 2: The distinction between the Upper memory block and the Lower memory block allows different wait cycles between the first 2 Mbytes and the second 2 Mbytes, and allows 2 different data strobe signals to be used to access 2 different memories.

Typically, the RAM will be located above address 0x200000 and the ROM below address 0x1FFFFF, with different access times. No extra hardware is required as $\overline{DS}$ is used to access the upper memory block and $\overline{DS2}$ is used to access the lower memory block.

CAUTION*:* The reset value of the Wait Control Register gives the maximum number of Wait cycles for external memory. To get optimum performance from the ST9, the user should write the UDS[2:0] and LDS[2:0] bits to 0, if the external addressed memories are fast enough.

9 I/O PORTS

9.1 INTRODUCTION

ST9 devices feature flexible individually programmable multifunctional input/output lines. Refer to the Pin Description Chapter for specific pin allocations. These lines, which are logically grouped as 8-bit ports, can be individually programmed to provide digital input/output and analog input, or to connect input/output signals to the on-chip peripherals as alternate pin functions. All ports can be individually configured as an input, bi-directional, output or alternate function. In addition, pull-ups can be turned off for open-drain operation, and weak pull-ups can be turned on in their place, to avoid the need for off-chip resistive pull-ups. Ports configured as open drain must never have voltage on the port pin exceeding V_{DD} (refer to the Electrical Characteristics section). Depending on the specific port, input buffers are software selectable to be TTL or CMOS compatible, however on Schmitt trigger ports, no selection is possible.

9.2 SPECIFIC PORT CONFIGURATIONS

Refer to the Pin Description chapter for a list of the specific port styles and reset values.

9.3 PORT CONTROL REGISTERS

Each port is associated with a Data register (PxDR) and three Control registers (PxC0, PxC1, PxC2). These define the port configuration and allow dynamic configuration changes during program execution. Port Data and Control registers are mapped into the Register File as shown in Figure 1. Port Data and Control registers are treated just like any other general purpose register. There are no special instructions for port manipulation: any instruction that can address a register, can address the ports. Data can be directly accessed in the port register, without passing through other memory or “accumulator” locations.

Figure 61. I/O Register Map

	GROUP E			GROUP F PAGE 2	GROUP F PAGE 3	GROUP F PAGE 43	
			FFh	Reserved	P7DR	P9DR	R255
			FEh	P3C2	P7C2	P9C2	R254
			FDh	P3C1	P7C1	P9C1	R253
			FCh	P3C0	P7C0	P9C0	R252
			FBh	Reserved	P6DR	P8DR	R251
			FAh	P2C2	P6C2	P8C2	R250
			F9h	P2C1	P6C1	P8C1	R249
			F8h	P2C0	P6C0	P8C0	R248
			F7h	Reserved	Reserved		R247
			F6h	P1C2	P5C2		R246
E5h	P5DR	R229	F5h	P1C1	P5C1		R245
E4h	P4DR	R228	F4h	P1C0	P5C0	Reserved	R244
E3h	P3DR	R227	F3h	Reserved	Reserved		R243
E2h	P2DR	R226	F2h	P0C2	P4C2		R242
E1h	P1DR	R225	F1h	P0C1	P4C1		R241
E0h	P0DR	R224	F0h	P0C0	P4C0		R240

System Registers

PORT CONTROL REGISTERS (Cont'd)

During Reset, ports with weak pull-ups are set in bidirectional/weak pull-up mode and the output Data Register is set to FFh. This condition is also held after Reset, except for Ports 0 and 1 in ROM-less devices, and can be redefined under software control.

Bidirectional ports without weak pull-ups are set in high impedance during reset. To ensure proper levels during reset, these ports must be externally connected to either V_{DD} or V_{SS} through external pull-up or pull-down resistors.

Other reset conditions may apply in specific ST9 devices.

9.4 INPUT/OUTPUT BIT CONFIGURATION

By programming the control bits PxC0.n and PxC1.n (see Figure 2) it is possible to configure bit Px.n as Input, Output, Bidirectional or Alternate Function Output, where X is the number of the I/O port, and n the bit within the port (n = 0 to 7).

When programmed as input, it is possible to select the input level as TTL or CMOS compatible by programming the relevant PxC2.n control bit.
This option is not available on Schmitt trigger ports.

The output buffer can be programmed as push-pull or open-drain.

A weak pull-up configuration can be used to avoid external pull-ups when programmed as bidirectional (except where the weak pull-up option has been permanently disabled in the pin hardware assignment).

Each pin of an I/O port may assume software programmable Alternate Functions (refer to the device Pin Description and to Section 1.5). To output signals from the ST9 peripherals, the port must be configured as AF OUT. On ST9 devices with A/D Converter(s), configure the ports used for analog inputs as AF IN.

The basic structure of the bit Px.n of a general purpose port Px is shown in Figure 3.

Independently of the chosen configuration, when the user addresses the port as the destination register of an instruction, the port is written to and the data is transferred from the internal Data Bus to the Output Master Latches. When the port is addressed as the source register of an instruction, the port is read and the data (stored in the Input Latch) is transferred to the internal Data Bus.

When Px.n is programmed as an Input:
(See Figure 4).

– The Output Buffer is forced tristate.

– The data present on the I/O pin is sampled into the Input Latch at the beginning of each instruction execution.

– The data stored in the Output Master Latch is copied into the Output Slave Latch at the end of the execution of each instruction. Thus, if bit Px.n is reconfigured as an Output or Bidirectional, the data stored in the Output Slave Latch will be reflected on the I/O pin.

INPUT/OUTPUT BIT CONFIGURATION (Cont'd)

Figure 62. Control Bits

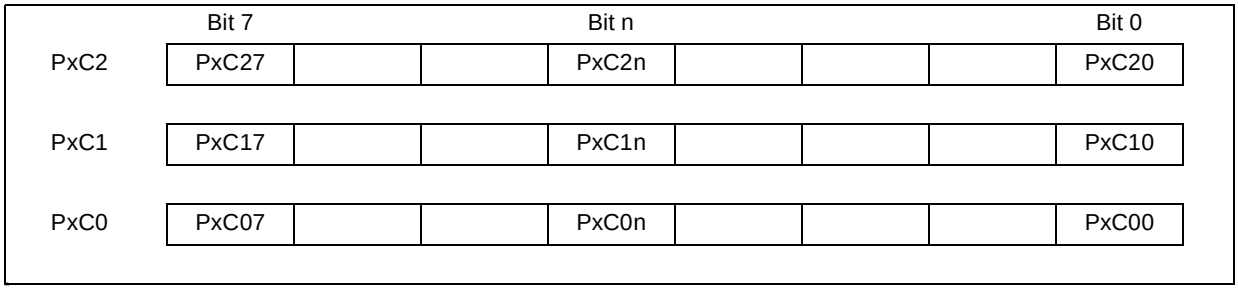


Table 25. Port Bit Configuration Table (n = 0, 1... 7; X = port number)

	General Purpose I/O Pins								A/D Pins
PXC2n PXC1n PXC0n	0 0 0	1 0 0	0 1 0	1 1 0	0 0 1	1 0 1	0 1 1	1 1 1	1 1 1
PXn Configuration	BID	BID	OUT	OUT	IN	IN	AF OUT	AF OUT	AF IN
PXn Output Type	WP OD	OD	PP	OD	HI-Z	HI-Z	PP	OD	HI-Z(1)
PXn Input Type	TTL (or Schmitt Trigger)	TTL (or Schmitt Trigger)	TTL (or Schmitt Trigger)	TTL (or Schmitt Trigger)	CMOS (or Schmitt Trigger)	TTL (or Schmitt Trigger)	TTL (or Schmitt Trigger)	TTL (or Schmitt Trigger)	Analog Input

(1) For A/D Converter inputs.

Legend:

X = Port
n = Bit
AF = Alternate Function
BID = Bidirectional
CMOS= CMOS Standard Input Levels
HI-Z = High Impedance
IN = Input
OD = Open Drain
OUT = Output
PP = Push-Pull
TTL = TTL Standard Input Levels
WP = Weak Pull-up

INPUT/OUTPUT BIT CONFIGURATION (Cont'd)

Figure 63. Basic Structure of an I/O Port Pin

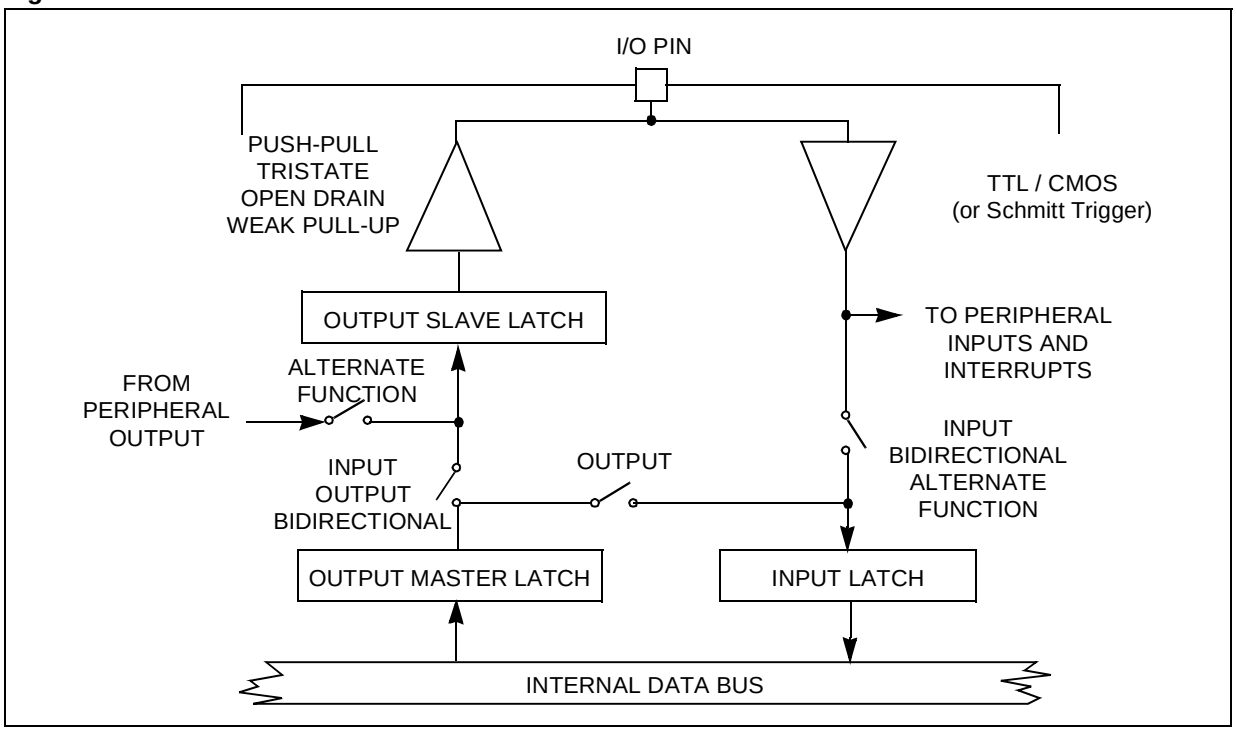


Figure 64. Input Configuration

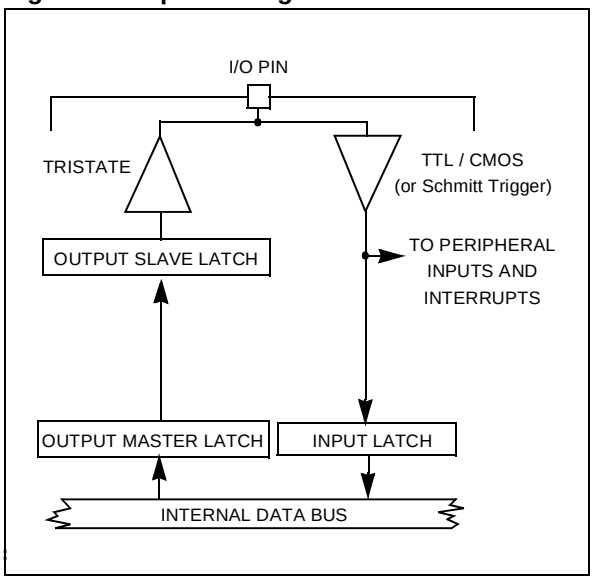
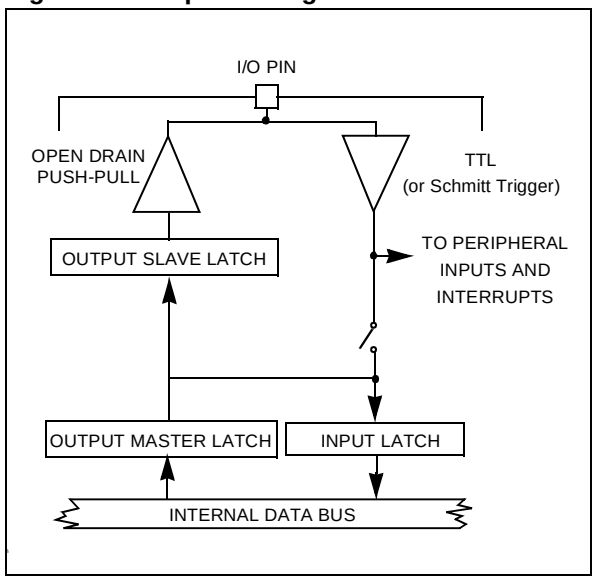


Figure 65. Output Configuration



INPUT/OUTPUT BIT CONFIGURATION (Cont'd)

When Px.n is programmed as an Output:
(Figure 5)

– The Output Buffer is turned on in an Open-drain or Push-pull configuration.

– The data stored in the Output Master Latch is copied both into the Input Latch and into the Output Slave Latch, driving the I/O pin, at the end of the execution of the instruction.

When Px.n is programmed as Bidirectional:
(Figure 6)

– The Output Buffer is turned on in an Open-Drain or Weak Pull-up configuration (except when disabled in hardware).

– The data present on the I/O pin is sampled into the Input Latch at the beginning of the execution of the instruction.

– The data stored in the Output Master Latch is copied into the Output Slave Latch, driving the I/O pin, at the end of the execution of the instruction.

WARNING: Due to the fact that in bidirectional mode the external pin is read instead of the output latch, particular care must be taken with arithmetic/logic and Boolean instructions performed on a bidirectional port pin.

These instructions use a read-modify-write sequence, and the result written in the port register depends on the logical level present on the external pin.

This may bring unwanted modifications to the port output register content.

For example:

Port register content, 0Fh
external port value, 03h
(Bits 3 and 2 are externally forced to 0)

A `bset` instruction on bit 7 will return:

Port register content, 83h
external port value, 83h
(Bits 3 and 2 have been cleared).

To avoid this situation, it is suggested that all operations on a port, using at least one bit in bidirectional mode, are performed on a copy of the port register, then transferring the result with a load instruction to the I/O port.

When Px.n is programmed as a digital Alternate Function Output:
(Figure 7)

– The Output Buffer is turned on in an Open-Drain or Push-Pull configuration.

– The data present on the I/O pin is sampled into the Input Latch at the beginning of the execution of the instruction.

– The signal from an on-chip function is allowed to load the Output Slave Latch driving the I/O pin. Signal timing is under control of the alternate function. If no alternate function is connected to Px.n, the I/O pin is driven to a high level when in Push-Pull configuration, and to a high impedance state when in open drain configuration.

Figure 66. Bidirectional Configuration

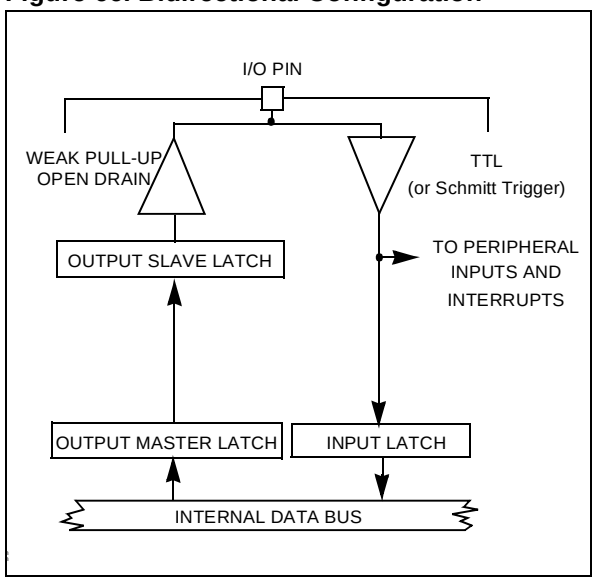
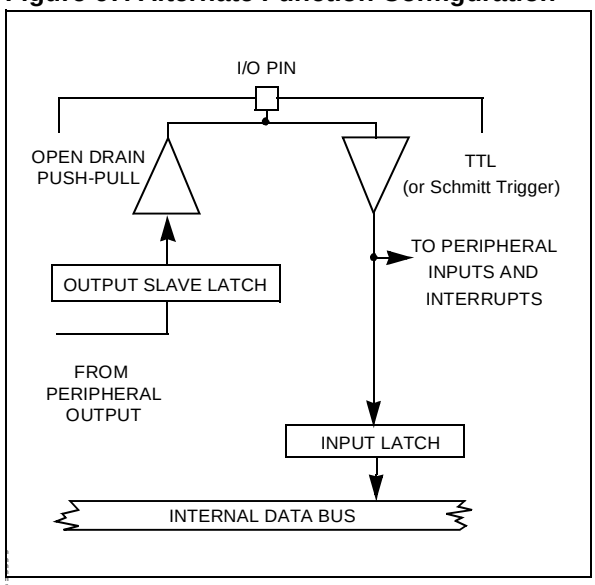


Figure 67. Alternate Function Configuration



9.5 ALTERNATE FUNCTION ARCHITECTURE

Each I/O pin may be connected to three different types of internal signal:

– Data bus Input/Output

– Alternate Function Input

– Alternate Function Output

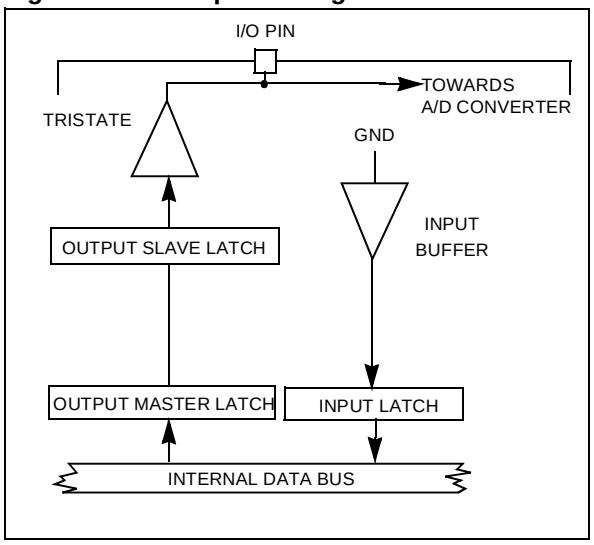
9.5.1 Pin Declared as I/O

A pin declared as I/O, is connected to the I/O buffer. This pin may be an Input, an Output, or a bidirectional I/O, depending on the value stored in (PxC2, PxC1 and PxC0).

9.5.2 Pin Declared as an Alternate Function Input

A single pin may be directly connected to several Alternate Function inputs. In this case, the user must select the required input mode (with the PxC2, PxC1, PxC0 bits) and enable the selected Alternate Function in the Control Register of the peripheral. No specific port configuration is required to enable an Alternate Function input, since the input buffer is directly connected to each alternate function module on the shared pin. As more than one module can use the same input, it is up to the user software to enable the required module as necessary. Parallel I/Os remain operational even when using an Alternate Function input. The exception to this is when an I/O port bit is permanently assigned by hardware as an A/D bit. In this case , after software programming of the bit in AF-OD-TTL, the Alternate function output is forced to logic level 1. The analog voltage level on the corresponding pin is directly input to the A/D (See Figure 8).

Figure 68. A/D Input Configuration



9.5.3 Pin Declared as an Alternate Function Output

The user must select the AF OUT configuration using the PxC2, PxC1, PxC0 bits. Several Alternate Function outputs may drive a common pin. In such case, the Alternate Function output signals are logically ANDed before driving the common pin. The user must therefore enable the required Alternate Function Output by software.

***WARNING**: When a pin is connected both to an alternate function output and to an alternate function input, it should be noted that the output signal will always be present on the alternate function input.*

9.6 I/O STATUS AFTER WFI, HALT AND RESET

The status of the I/O ports during the Wait For Interrupt, Halt and Reset operational modes is shown in the following table. The External Memory Interface ports are shown separately. If only the internal memory is being used and the ports are acting as I/O, the status is the same as shown for the other I/O ports.

Mode	Ext. Mem - I/O Ports		I/O Ports
	P0	P1, P2, P6, P9[7:2] *	
WFI	High Impedance or next address (depending on the last memory operation performed on Port)	Next Address	Not Affected (clock outputs running)
HALT	High Impedance	Next Address	Not Affected (clock outputs stopped)
RESET	Alternate function push-pull (ROMless device)		Bidirectional Weak Pull-up (High impedance when disabled in hardware).

* Depending on device

10 ON-CHIP PERIPHERALS

10.1 TIMER/WATCHDOG (WDT)

Important Note: This chapter is a generic description of the WDT peripheral. However depending on the ST9 device, some or all of WDT interface signals described may not be connected to external pins. For the list of WDT pins present on the ST9 device, refer to the device pinout description in the first section of the data sheet.

10.1.1 Introduction

The Timer/Watchdog (WDT) peripheral consists of a programmable 16-bit timer and an 8-bit prescaler. It can be used, for example, to:

– Generate periodic interrupts

– Measure input signal pulse widths

– Request an interrupt after a set number of events

– Generate an output signal waveform

– Act as a Watchdog timer to monitor system integrity

The main WDT registers are:

– Control register for the input, output and interrupt logic blocks (WDTCR)

– 16-bit counter register pair (WDTHR, WDTLR)

– Prescaler register (WDTPR)

The hardware interface consists of up to five signals:

– WDIN External clock input

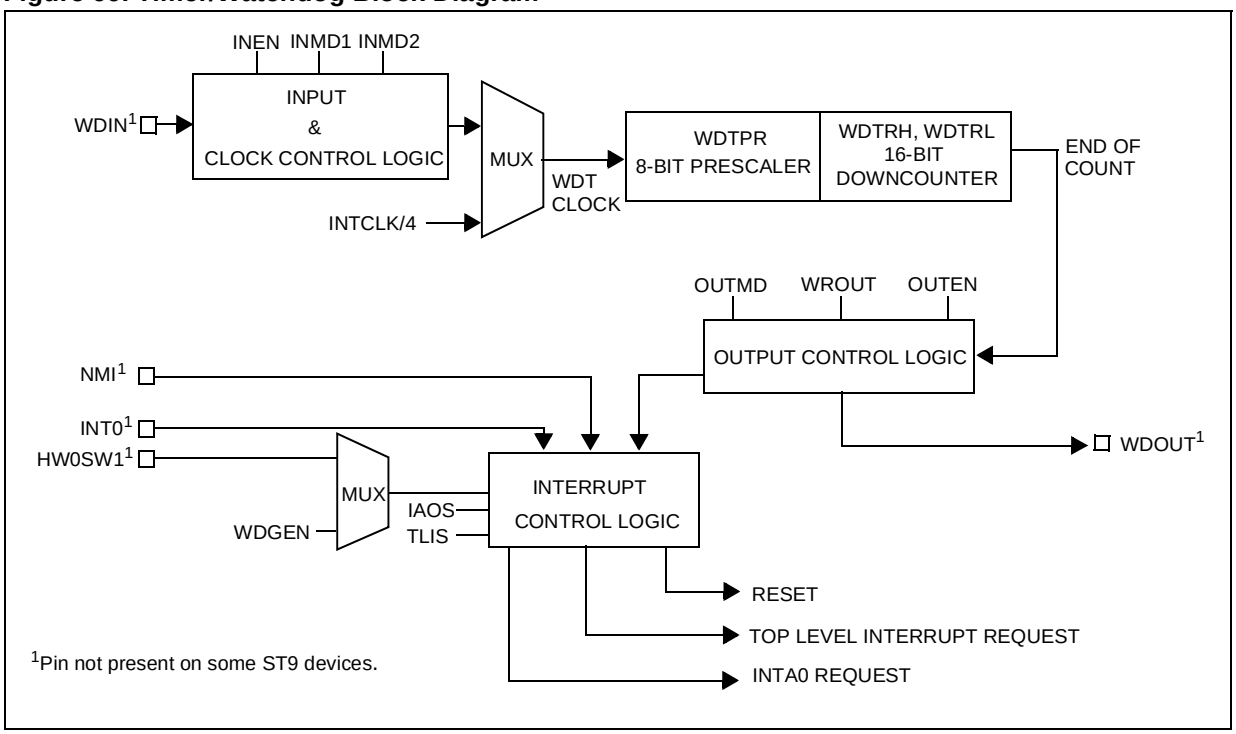
– WDOUT Square wave or PWM signal output

– INT0 External interrupt input

– NMI Non-Maskable Interrupt input

– HW0SW1 Hardware/Software Watchdog enable.

Figure 69. Timer/Watchdog Block Diagram



TIMER/WATCHDOG (Cont'd)

10.1.2 Functional Description

10.1.2.1 External Signals

The HW0SW1 pin can be used to permanently enable Watchdog mode. Refer to section 10.1.3.1 on page 126.

The WDIN Input pin can be used in one of four modes:

– Event Counter Mode

– Gated External Input Mode

– Triggerable Input Mode

– Retriggerable Input Mode

The WDOUT output pin can be used to generate a square wave or a Pulse Width Modulated signal.

An interrupt, generated when the WDT is running as the 16-bit Timer/Counter, can be used as a Top Level Interrupt or as an interrupt source connected to channel A0 of the external interrupt structure (replacing the INT0 interrupt input).

The counter can be driven either by an external clock, or internally by INTCLK divided by 4.

10.1.2.2 Initialisation

The prescaler (WDTPR) and counter (WDTRL, WDTRH) registers must be loaded with initial values before starting the Timer/Counter. If this is not done, counting will start with reset values.

10.1.2.3 Start/Stop

The ST_SP bit enables downcounting. When this bit is set, the Timer will start at the beginning of the following instruction. Resetting this bit stops the counter.

If the counter is stopped and restarted, counting will resume from the last value unless a new constant has been entered in the Timer registers (WDTRL, WDTRH).

A new constant can be written in the WDTRH, WDTRL, WDTPR registers while the counter is running. The new value of the WDTRH, WDTRL registers will be loaded at the next End of Count (EOC) condition while the new value of the WDTPR register will be effective immediately.

End of Count is when the counter is 0.

When Watchdog mode is enabled the state of the ST_SP bit is irrelevant.

10.1.2.4 Single/Continuous Mode

The S_C bit allows selection of single or continuous mode.This Mode bit can be written with the Timer stopped or running. It is possible to toggle the S_C bit and start the counter with the same instruction.

Single Mode

On reaching the End Of Count condition, the Timer stops, reloads the constant, and resets the Start/Stop bit. Software can check the current status by reading this bit. To restart the Timer, set the Start/Stop bit.

Note: If the Timer constant has been modified during the stop period, it is reloaded at start time.

Continuous Mode

On reaching the End Of Count condition, the counter automatically reloads the constant and restarts. It is stopped only if the Start/Stop bit is reset.

10.1.2.5 Input Section

If the Timer/Counter input is enabled (INEN bit) it can count pulses input on the WDIN pin. Otherwise it counts the internal clock/4.

For instance, when INTCLK = 24MHz, the End Of Count rate is:

2.79 seconds for Maximum Count
(Timer Const. = FFFFh, Prescaler Const. = FFh)

166 ns for Minimum Count
(Timer Const. = 0000h, Prescaler Const. = 00h)

The Input pin can be used in one of four modes:

– Event Counter Mode

– Gated External Input Mode

– Triggerable Input Mode

– Retriggerable Input Mode

The mode is configurable in the WDTCR.

10.1.2.6 Event Counter Mode

In this mode the Timer is driven by the external clock applied to the input pin, thus operating as an event counter. The event is defined as a high to low transition of the input signal. Spacing between trailing edges should be at least 8 INTCLK periods (or 333ns with INTCLK = 24MHz).

Counting starts at the next input event after the ST_SP bit is set and stops when the ST_SP bit is reset.

TIMER/WATCHDOG (Cont'd)

10.1.2.7 Gated Input Mode

This mode can be used for pulse width measurement. The Timer is clocked by INTCLK/4, and is started and stopped by means of the input pin and the ST_SP bit. When the input pin is high, the Timer counts. When it is low, counting stops. The maximum input pin frequency is equivalent to INTCLK/8.

10.1.2.8 Triggerable Input Mode

The Timer (clocked internally by INTCLK/4) is started by the following sequence:

– setting the Start-Stop bit, followed by

– a High to Low transition on the input pin.

To stop the Timer, reset the ST_SP bit.

10.1.2.9 Retriggerable Input Mode

In this mode, the Timer (clocked internally by INTCLK/4) is started by setting the ST_SP bit. A High to Low transition on the input pin causes counting to restart from the initial value. When the Timer is stopped (ST_SP bit reset), a High to Low transition of the input pin has no effect.

10.1.2.10 Timer/Counter Output Modes

Output modes are selected by means of the OUTEN (Output Enable) and OUTMD (Output Mode) bits of the WDTCR register.

No Output Mode
(OUTEN = "0")

The output is disabled and the corresponding pin is set high, in order to allow other alternate functions to use the I/O pin.

Square Wave Output Mode
(OUTEN = "1", OUTMD = "0")

The Timer outputs a signal with a frequency equal to half the End of Count repetition rate on the WDOUT pin. With an INTCLK frequency of 20MHz, this allows a square wave signal to be generated whose period can range from 400ns to 6.7 seconds.

Pulse Width Modulated Output Mode
(OUTEN = "1", OUTMD = "1")

The state of the WROUT bit is transferred to the output pin (WDOUT) at the End of Count, and is held until the next End of Count condition. The user can thus generate PWM signals by modifying the status of the WROUT pin between End of Count events, based on software counters decremented by the Timer Watchdog interrupt.

10.1.3 Watchdog Timer Operation

This mode is used to detect the occurrence of a software fault, usually generated by external interference or by unforeseen logical conditions, which causes the application program to abandon its normal sequence of operation. The Watchdog, when enabled, resets the MCU, unless the program executes the correct write sequence before expiry of the programmed time period. The application program must be designed so as to correctly write to the WDTLR Watchdog register at regular intervals during all phases of normal operation.

10.1.3.1 Hardware Watchdog/Software Watchdog

The HW0SW1 pin (when available) selects Hardware Watchdog or Software Watchdog.

If HW0SW1 is held low:

– The Watchdog is enabled by hardware immediately after an external reset. (Note: Software reset or Watchdog reset have no effect on the Watchdog enable status).

– The initial counter value (FFFFh) cannot be modified, however software can change the prescaler value on the fly.

– The WDGEN bit has no effect. (Note: it is not forced low).

If HW0SW1 is held high, or is not present:

– The Watchdog can be enabled by resetting the WDGEN bit.

10.1.3.2 Starting the Watchdog

In Watchdog mode the Timer is clocked by INTCLK/4.

If the Watchdog is software enabled, the time base must be written in the timer registers before entering Watchdog mode by resetting the WDGEN bit. Once reset, this bit cannot be changed by software.

If the Watchdog is hardware enabled, the time base is fixed by the reset value of the registers.

Resetting WDGEN causes the counter to start, regardless of the value of the Start-Stop bit.

In Watchdog mode, only the Prescaler Constant may be modified.

If the End of Count condition is reached a System Reset is generated.

TIMER/WATCHDOG (Cont'd)

10.1.3.3 Preventing Watchdog System Reset

In order to prevent a system reset, the sequence AAh, 55h must be written to WDTLR (Watchdog Timer Low Register). Once 55h has been written, the Timer reloads the constant and counting restarts from the preset value.

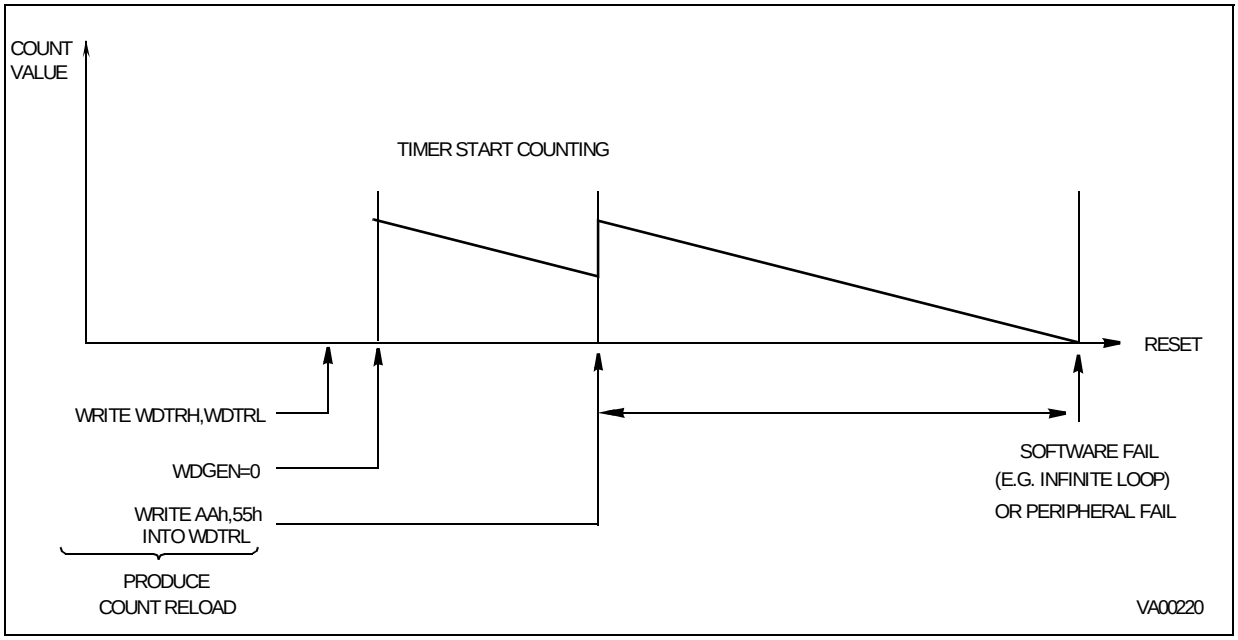
To reload the counter, the two writing operations must be performed sequentially without inserting other instructions that modify the value of the WDTLR register between the writing operations. The maximum allowed time between two reloads of the counter depends on the Watchdog timeout period.

10.1.3.4 Non-Stop Operation

In Watchdog Mode, a `Halt` instruction is regarded as illegal. Execution of the `Halt` instruction stops further execution by the CPU and interrupt acknowledgment, but does not stop INTCLK, CPUCLK or the Watchdog Timer, which will cause a System Reset when the End of Count condition is reached. Furthermore, ST_SP, S_C and the Input Mode selection bits are ignored. Hence, regardless of their status, the counter always runs in Continuous Mode, driven by the internal clock.

The Output mode should not be enabled, since in this context it is meaningless.

Figure 70. Watchdog Timer Mode



TIMER/WATCHDOG (Cont'd)

10.1.4 WDT Interrupts

The Timer/Watchdog issues an interrupt request at every End of Count, when this feature is enabled.

A pair of control bits, IA0S (EIVR.1, Interrupt A0 selection bit) and TLIS (EIVR.2, Top Level Input Selection bit) allow the selection of 2 interrupt sources (Timer/Watchdog End of Count, or External Pin) handled in two different ways, as a Top Level Non Maskable Interrupt (Software Reset), or as a source for channel A0 of the external interrupt logic.

A block diagram of the interrupt logic is given in Figure 71.

Note: Software traps can be generated by setting the appropriate interrupt pending bit.

Table 26 below, shows all the possible configurations of interrupt/reset sources which relate to the Timer/Watchdog.

A reset caused by the watchdog will set bit 6, WDGRES of R242 - Page 55 (Clock Flag Register). See section CLOCK CONTROL REGISTERS.

Figure 71. Interrupt Sources

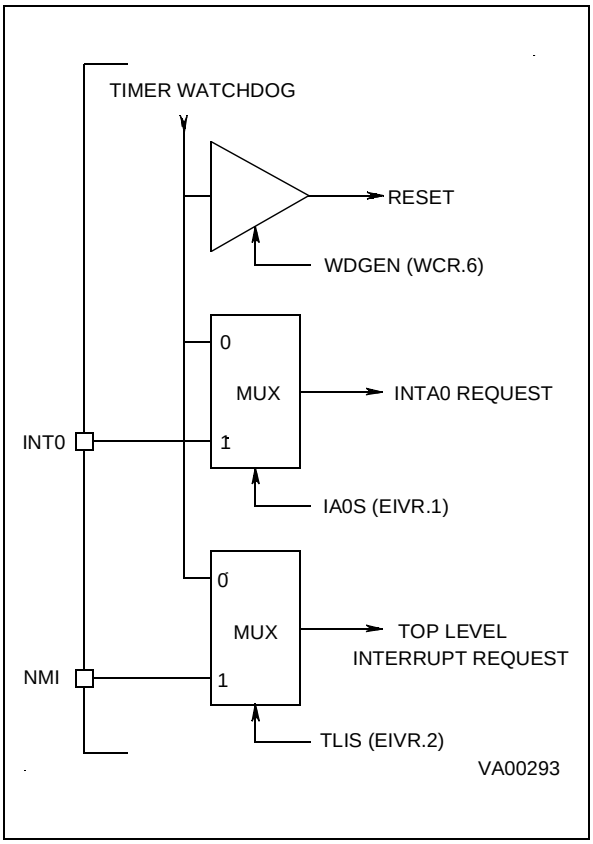


Table 26. Interrupt Configuration

Control Bits			Enabled Sources			Operating Mode
WDGEN	IA0S	TLIS	Reset	INTA0	Top Level	
0	0	0	WDG/Ext Reset	SW TRAP	SW TRAP	Watchdog
0	0	1	WDG/Ext Reset	SW TRAP	Ext Pin	Watchdog
0	1	0	WDG/Ext Reset	Ext Pin	SW TRAP	Watchdog
0	1	1	WDG/Ext Reset	Ext Pin	Ext Pin	Watchdog
1	0	0	Ext Reset	Timer	Timer	Timer
1	0	1	Ext Reset	Timer	Ext Pin	Timer
1	1	0	Ext Reset	Ext Pin	Timer	Timer
1	1	1	Ext Reset	Ext Pin	Ext Pin	Timer

Legend:
WDG = Watchdog function
SW TRAP = Software Trap

Note: If IA0S and TLIS = 0 (enabling the Watchdog EOC as interrupt source for both Top Level and INTA0 interrupts), only the INTA0 interrupt is taken into account.

TIMER/WATCHDOG (Cont'd)

10.1.5 Register Description

The Timer/Watchdog is associated with 4 registers mapped into Group F, Page 0 of the Register File.

WDTHR: Timer/Watchdog High Register

WDTLR: Timer/Watchdog Low Register

WDTPR: Timer/Watchdog Prescaler Register

WDTCR: Timer/Watchdog Control Register

Three additional control bits are mapped in the following registers on Page 0:

Watchdog Mode Enable, (WCR.6)

Top Level Interrupt Selection, (EIVR.2)

Interrupt A0 Channel Selection, (EIVR.1)

Note: The registers containing these bits also contain other functions. Only the bits relevant to the operation of the Timer/Watchdog are shown here.

Counter Register

This 16-bit register (WDTLR, WDTHR) is used to load the 16-bit counter value. The registers can be read or written "on the fly".

TIMER/WATCHDOG HIGH REGISTER (WDTHR)
R248 - Read/Write
Register Page: 0
Reset value: 1111 1111 (FFh)

7							0
R15	R14	R13	R12	R11	R10	R9	R8

Bits 7:0 = **R[15:8]** *Counter Most Significant Bits*.

TIMER/WATCHDOG LOW REGISTER (WDTLR)
R249 - Read/Write
Register Page: 0
Reset value: 1111 1111b (FFh)

7							0
R7	R6	R5	R4	R3	R2	R1	R0

Bits 7:0 = **R[7:0]** *Counter Least Significant Bits.*

TIMER/WATCHDOG PRESCALER REGISTER (WDTPR)
R250 - Read/Write
Register Page: 0
Reset value: 1111 1111 (FFh)

7							0
PR7	PR6	PR5	PR4	PR3	PR2	PR1	PR0

Bits 7:0 = **PR[7:0]** *Prescaler value.*
A programmable value from 1 (00h) to 256 (FFh).

Warning: *In order to prevent incorrect operation of the Timer/Watchdog, the prescaler (WDTPR) and counter (WDTRL, WDTRH) registers must be initialised before starting the Timer/Watchdog. If this is not done, counting will start with the reset (un-initialised) values.*

WATCHDOG TIMER CONTROL REGISTER (WDTCR)
R251- Read/Write
Register Page: 0
Reset value: 0001 0010 (12h)

7							0
ST_SP	S_C	INMD1	INMD2	INEN	OUTMD	WROUT	OUTEN

Bit 7 = **ST_SP**: *Start/Stop Bit*.
This bit is set and cleared by software.
0: Stop counting
1: Start counting (see Warning above)

Bit 6 = **S_C**: *Single/Continuous*.
This bit is set and cleared by software.
0: Continuous Mode
1: Single Mode

Bits 5:4 = **INMD[1:2]**: *Input mode selection bits*.

These bits select the input mode:

INMD1	INMD2	INPUT MODE
0	0	Event Counter
0	1	Gated Input (Reset value)
1	0	Triggerable Input
1	1	Retriggerable Input

TIMER/WATCHDOG (Cont'd)

Bit 3 = **INEN**: *Input Enable*.
This bit is set and cleared by software.
0: Disable input section
1: Enable input section

Bit 2 = **OUTMD**: *Output Mode.*
This bit is set and cleared by software.
0: The output is toggled at every End of Count
1: The value of the WROUT bit is transferred to the output pin on every End Of Count if OUTEN=1.

Bit 1 = **WROUT**: *Write Out*.
The status of this bit is transferred to the Output pin when OUTMD is set; it is user definable to allow PWM output (on Reset WROUT is set).

Bit 0 = **OUTEN**: *Output Enable bit*.
This bit is set and cleared by software.
0: Disable output
1: Enable output

WAIT CONTROL REGISTER (WCR)
R252 - Read/Write
Register Page: 0
Reset value: 0111 1111 (7Fh)

7							0
x	WDGEN	x	x	x	x	x	x

Bit 6 = **WDGEN**: *Watchdog Enable* (active low).
Resetting this bit via software enters the Watchdog mode. Once reset, it cannot be set anymore by the user program. At System Reset, the Watchdog mode is disabled.

Note: This bit is ignored if the Hardware Watchdog option is enabled by pin HW0SW1 (if available).

EXTERNAL INTERRUPT VECTOR REGISTER (EIVR)
R246 - Read/Write
Register Page: 0
Reset value: xxxx 0110 (x6h)

7							0
x	x	x	x	x	TLIS	IA0S	x

Bit 2 = **TLIS**: *Top Level Input Selection*.
This bit is set and cleared by software.
0: Watchdog End of Count is TL interrupt source
1: NMI is TL interrupt source

Bit 1 = **IA0S**: *Interrupt Channel A0 Selection.*
This bit is set and cleared by software.
0: Watchdog End of Count is INTA0 source
1: External Interrupt pin is INTA0 source

Warning: *To avoid spurious interrupt requests, the IA0S bit should be accessed only when the interrupt logic is disabled (i.e. after the DI instruction). It is also necessary to clear any possible interrupt pending requests on channel A0 before enabling this interrupt channel. A delay instruction (e.g. a NOP instruction) must be inserted between the reset of the interrupt pending bit and the IA0S write instruction.*

Other bits are described in the Interrupt section.

10.2 STANDARD TIMER (STIM)

Important Note: This chapter is a generic description of the STIM peripheral. Depending on the ST9 device, some or all of the interface signals described may not be connected to external pins. For the list of STIM pins present on the particular ST9 device, refer to the pinout description in the first section of the data sheet.

10.2.1 Introduction

The Standard Timer includes a programmable 16-bit down counter and an associated 8-bit prescaler with Single and Continuous counting modes capability. The Standard Timer uses an input pin (STIN) and an output (STOUT) pin. These pins, when available, may be independent pins or connected as Alternate Functions of an I/O port bit.

STIN can be used in one of four programmable input modes:

– event counter,

– gated external input mode,

– triggerable input mode,

– retriggerable input mode.

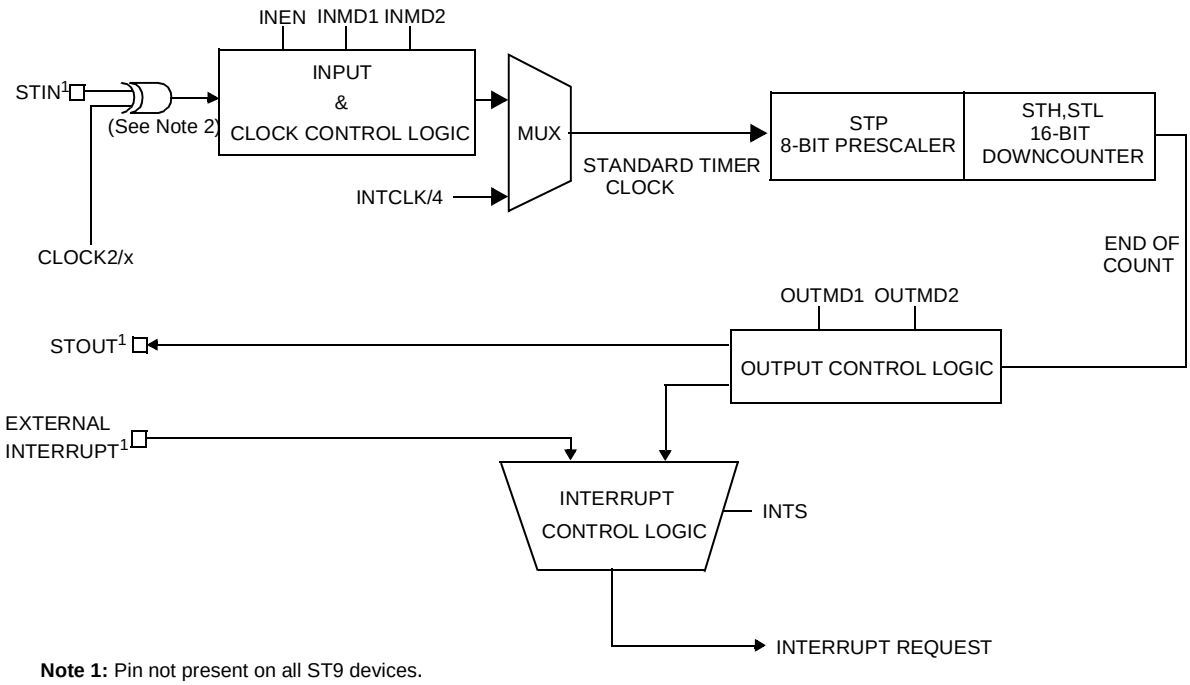
STOUT can be used to generate a Square Wave or Pulse Width Modulated signal.

The Standard Timer is composed of a 16-bit down counter with an 8-bit prescaler. The input clock to the prescaler can be driven either by an internal clock equal to INTCLK divided by 4, or by CLOCK2 derived directly from the external oscillator, divided by device dependent prescaler value, thus providing a stable time reference independent from the PLL programming or by an external clock connected to the STIN pin.

The Standard Timer End Of Count condition is able to generate an interrupt which is connected to one of the external interrupt channels.

The End of Count condition is defined as the Counter Underflow, whenever 00h is reached.

Figure 72. Standard Timer Block Diagram



Note 1: Pin not present on all ST9 devices.

Note 2: Depending on device, the source of the INPUT & CLOCK CONTROL LOGIC block may be permanently connected either to STIN or the RCCU signal CLOCK2/x. In devices without STIN and CLOCK2, the INEN bit must be held at 0.

STANDARD TIMER (Cont'd)

10.2.2 Functional Description

10.2.2.1 Timer/Counter control

Start-stop Count. The ST-SP bit (STC.7) is used in order to start and stop counting. An instruction which sets this bit will cause the Standard Timer to start counting at the beginning of the next instruction. Resetting this bit will stop the counter.

If the counter is stopped and restarted, counting will resume from the value held at the stop condition, unless a new constant has been entered in the Standard Timer registers during the stop period. In this case, the new constant will be loaded as soon as counting is restarted.

A new constant can be written in STH, STL, STP registers while the counter is running. The new value of the STH and STL registers will be loaded at the next End of Count condition, while the new value of the STP register will be loaded immediately.

WARNING: In order to prevent incorrect counting of the Standard Timer, the prescaler (STP) and counter (STL, STH) registers must be initialised before the starting of the timer. If this is not done, counting will start with the reset values (STH=FFh, STL=FFh, STP=FFh).

Single/Continuous Mode.
The S-C bit (STC.6) selects between the Single or Continuous mode.

SINGLE MODE: at the End of Count, the Standard Timer stops, reloads the constant and resets the Start/Stop bit (the user programmer can inspect the timer current status by reading this bit). Setting the Start/Stop bit will restart the counter.

CONTINUOUS MODE: At the End of the Count, the counter automatically reloads the constant and restarts. It is only stopped by resetting the Start/Stop bit.

The S-C bit can be written either with the timer stopped or running. It is possible to toggle the S-C bit and start the Standard Timer with the same instruction.

10.2.2.2 Standard Timer Input Modes (ST9 devices with Standard Timer Input STIN)

Bits INMD2, INMD1 and INEN are used to select the input modes. The Input Enable (INEN) bit enables the input mode selected by the INMD2 and INMD1 bits. If the input is disabled (INEN="0"), the values of INMD2 and INMD1 are not taken into account. In this case, this unit acts as a 16-bit timer (plus prescaler) directly driven by INTCLK/4 and transitions on the input pin have no effect.

Event Counter Mode (INMD1 = "0", INMD2 = "0")

The Standard Timer is driven by the signal applied to the input pin (STIN) which acts as an external clock. The unit works therefore as an event counter. The event is a high to low transition on STIN. Spacing between trailing edges should be at least the period of INTCLK multiplied by 8 (i.e. the maximum Standard Timer input frequency is 3 MHz with INTCLK = 24MHz).

Gated Input Mode (INMD1 = "0", INMD2 = "1")

The Timer uses the internal clock (INTCLK divided by 4) and starts and stops the Timer according to the state of STIN pin. When the status of the STIN is High the Standard Timer count operation proceeds, and when Low, counting is stopped.

Triggerable Input Mode (INMD1 = "1", INMD2 = "0")

The Standard Timer is started by:

a) setting the Start-Stop bit, AND

b) a High to Low (low trigger) transition on STIN.

In order to stop the Standard Timer in this mode, it is only necessary to reset the Start-Stop bit.

Retriggerable Input Mode (INMD1 = "1", INMD2 = "1")

In this mode, when the Standard Timer is running (with internal clock), a High to Low transition on STIN causes the counting to start from the last constant loaded into the STL/STH and STP registers. When the Standard Timer is stopped (ST-SP bit equal to zero), a High to Low transition on STIN has no effect.

10.2.2.3 Time Base Generator (ST9 devices without Standard Timer Input STIN)

For devices where STIN is replaced by a connection to CLOCK2, the condition (INMD1 = "0", INMD2 = "0") will allow the Standard Timer to generate a stable time base independent from the PLL programming.

STANDARD TIMER (Cont'd)

10.2.2.4 Standard Timer Output Modes

OUTPUT modes are selected using 2 bits of the STC register: OUTMD1 and OUTMD2.

No Output Mode (OUTMD1 = "0", OUTMD2 = "0")

The output is disabled and the corresponding pin is set high, in order to allow other alternate functions to use the I/O pin.

Square Wave Output Mode (OUTMD1 = "0", OUTMD2 = "1")

The Standard Timer toggles the state of the STOUT pin on every End Of Count condition. With INTCLK = 24MHz, this allows generation of a square wave with a period ranging from 333ns to 5.59 seconds.

PWM Output Mode (OUTMD1 = "1")

The value of the OUTMD2 bit is transferred to the STOUT output pin at the End Of Count. This allows the user to generate PWM signals, by modifying the status of OUTMD2 between End of Count events, based on software counters decremented on the Standard Timer interrupt.

10.2.3 Interrupt Selection

The Standard Timer may generate an interrupt request at every End of Count.

Bit 2 of the STC register (INTS) selects the interrupt source between the Standard Timer interrupt and the external interrupt pin. Thus the Standard Timer Interrupt uses the interrupt channel and takes the priority and vector of the external interrupt channel.

If INTS is set to "1", the Standard Timer interrupt is disabled; otherwise, an interrupt request is generated at every End of Count.

Note: When enabling or disabling the Standard Timer Interrupt (writing INTS in the STC register) an edge may be generated on the interrupt channel, causing an unwanted interrupt.

To avoid this spurious interrupt request, the INTS bit should be accessed only when the interrupt logic is disabled (i.e. after the DI instruction). It is also necessary to clear any possible interrupt pending requests on the corresponding external interrupt channel before enabling it. A delay instruction (i.e. a NOP instruction) must be inserted between the reset of the interrupt pending bit and the INTS write instruction.

10.2.4 Register Mapping

Depending on the ST9 device there may be up to 4 Standard Timers (refer to the block diagram in the first section of the data sheet).

Each Standard Timer has 4 registers mapped into Page 11 in Group F of the Register File

In the register description on the following page, register addresses refer to STIM0 only.

STD Timer	Register	Register Address
STIM0	STH0	R240 (F0h)
	STL0	R241 (F1h)
	STP0	R242 (F2h)
	STC0	R243 (F3h)
STIM1	STH1	R244 (F4h)
	STL1	R245 (F5h)
	STP1	R246 (F6h)
	STC1	R247 (F7h)
STIM2	STH2	R248 (F8h)
	STL2	R249 (F9h)
	STP2	R250 (FAh)
	STC2	R251 (FBh)
STIM3	STH3	R252 (FCh)
	STL3	R253 (FDh)
	STP3	R254 (FEh)
	STC3	R255 (FFh)

Note: The four standard timers are not implemented on all ST9 devices. Refer to the block diagram of the device for the number of timers.

STANDARD TIMER (Cont'd)

10.2.5 Register Description

COUNTER HIGH BYTE REGISTER (STH)
R240 - Read/Write
Register Page: 11
Reset value: 1111 1111 (FFh)

7							0
ST.15	ST.14	ST.13	ST.12	ST.11	ST.10	ST.9	ST.8

Bits 7:0 = **ST.[15:8]**: *Counter High-Byte.*

COUNTER LOW BYTE REGISTER (STL)
R241 - Read/Write
Register Page: 11
Reset value: 1111 1111 (FFh)

7							0
ST.7	ST.6	ST.5	ST.4	ST.3	ST.2	ST.1	ST.0

Bits 7:0 = **ST.[7:0]**: *Counter Low Byte.*

Writing to the STH and STL registers allows the user to enter the Standard Timer constant, while reading it provides the counter's current value. Thus it is possible to read the counter on-the-fly.

STANDARD TIMER PRESCALER REGISTER (STP)
R242 - Read/Write
Register Page: 11
Reset value: 1111 1111 (FFh)

7							0
STP.7	STP.6	STP.5	STP.4	STP.3	STP.2	STP.1	STP.0

Bits 7:0 = **STP.[7:0]**: *Prescaler.*
The Prescaler value for the Standard Timer is programmed into this register. When reading the STP register, the returned value corresponds to the programmed data instead of the current data.
00h: No prescaler
01h: Divide by 2
FFh: Divide by 256

STANDARD TIMER CONTROL REGISTER (STC)
R243 - Read/Write
Register Page: 11
Reset value: 0001 0100 (14h)

7							0
ST-SP	S-C	INMD1	INMD2	INEN	INTS	OUTMD1	OUTMD2

Bit 7 = **ST-SP**: *Start-Stop Bit.*
This bit is set and cleared by software.
0: Stop counting
1: Start counting

Bit 6 = **S-C**: *Single-Continuous Mode Select.*
This bit is set and cleared by software.
0: Continuous Mode
1: Single Mode

Bits 5:4 = **INMD[1:2]**: *Input Mode Selection.*
These bits select the Input functions as shown in Section 0.1.2.2, when enabled by INEN.

INMD1	INMD2	Mode
0	**0**	Event Counter mode
0	**1**	Gated input mode
1	**0**	Triggerable mode
1	**1**	Retriggerable mode

Bit 3 = **INEN**: *Input Enable.*
This bit is set and cleared by software. If neither the STIN pin nor the CLOCK2 line are present, INEN must be 0.
0: Input section disabled
1: Input section enabled

Bit 2 = **INTS**: *Interrupt Selection.*
0: Standard Timer interrupt enabled
1: Standard Timer interrupt is disabled and the external interrupt pin is enabled.

Bits 1:0 = **OUTMD[1:2]**: Output Mode Selection.
These bits select the output functions as described in Section 0.1.2.4.

OUTMD1	OUTMD2	Mode
0	**0**	No output mode
0	**1**	Square wave output mode
1	**x**	PWM output mode

10.3 EXTENDED FUNCTION TIMER (EFT)

10.3.1 Introduction

The timer consists of a 16-bit free-running counter driven by a programmable prescaler.

It may be used for a variety of purposes, including pulse length measurement of up to two input signals (*input capture*) or generation of up to two output waveforms (*output compare* and *PWM*).

Pulse lengths and waveform periods can be modulated from a few microseconds to several milliseconds using the timer prescaler and the INTCLK prescaler.

10.3.2 Main Features

- **Programmable prescaler: INTCLK divided by 2, 4 or 8.**
- **Overflow status flag and maskable interrupts**
- **External clock input (must be at least 4 times slower than the INTCLK clock speed) with the choice of active edge**
- **Output compare functions with:**
  - 2 dedicated 16-bit registers
  - 2 dedicated programmable signals
  - 2 dedicated status flags
  - 1 dedicated maskable interrupt
- **Input capture functions with**
  - 2 dedicated 16-bit registers
  - 2 dedicated active edge selection signals
  - 2 dedicated status flags
  - 1 dedicated maskable interrupt
- **Pulse width modulation mode (PWM)**
- **One pulse mode**
- **5 alternate functions on I/O ports**
- **Global timer interrupt (EFTI) mapped on external interrupt channel**

The Block Diagram is shown in Figure 73.

Table 27. EFT Pin Naming Conventions

Function	EFT0	EFT1
Input Capture 1 - ICAP1	ICAPA0	ICAPA1
Input Capture 2 - ICAP2	ICAPB0	ICAPB1
Output Compare 1 - OCMP1	OCMPA0	OCMPA1
Output Compare 2 - OCMP2	OCMPB0	OCMPB1

10.3.3 Functional Description

10.3.3.1 Counter

The principal block of the Programmable Timer is a 16-bit free running counter and its associated 16-bit registers:

Counter Registers

- Counter High Register (CHR) is the most significant byte (MSB).
- Counter Low Register (CLR) is the least significant byte (LSB).

Alternate Counter Registers

- Alternate Counter High Register (ACHR) is the most significant byte (MSB).
- Alternate Counter Low Register (ACLR) is the least significant byte (LSB).

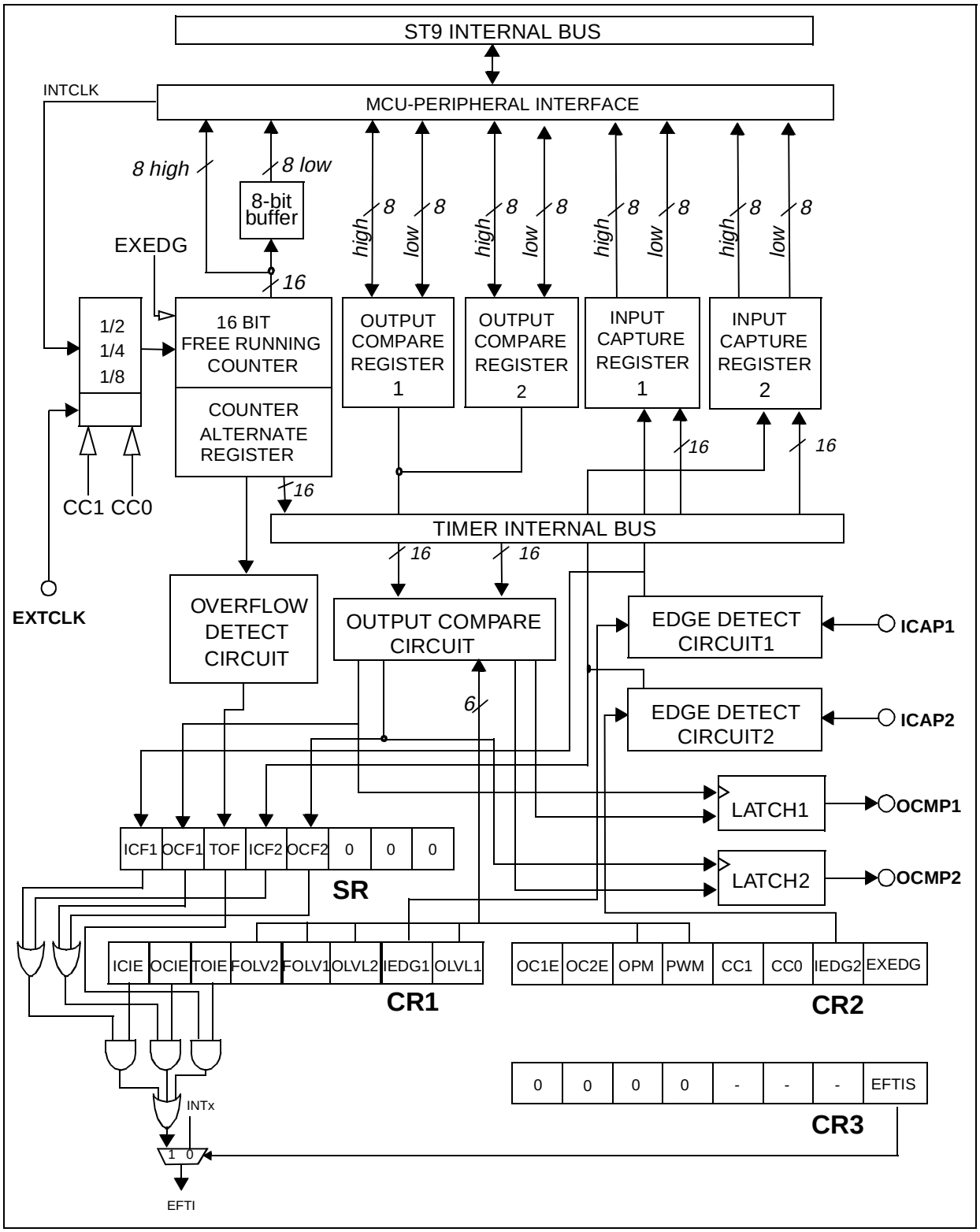
These two read-only 16-bit registers contain the same value but with the difference that reading the ACLR register does not clear the TOF bit (overflow flag), (see note page 137).

Writing in the CLR register or ACLR register resets the free running counter to the FFFCh value.

The timer clock depends on the clock control bits of the CR2 register, as illustrated in Table 28. The value in the counter register repeats every 131.072, 262.144 or 524.288 INTCLK cycles depending on the CC1 and CC0 bits.

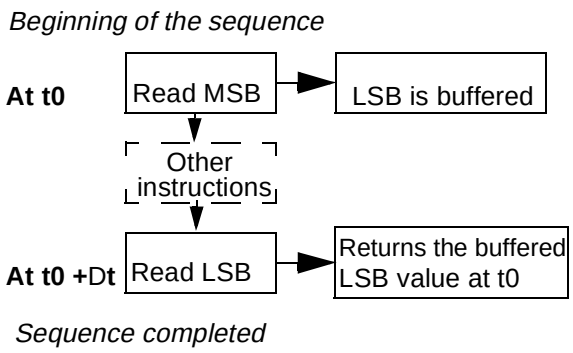
EXTENDED FUNCTION TIMER (Cont'd)

Figure 73. Timer Block Diagram



EXTENDED FUNCTION TIMER (Cont'd)

16-bit read sequence: (from either the Counter Register or the Alternate Counter Register).



The user must read the MSB first, then the LSB value is buffered automatically.

This buffered value remains unchanged until the 16-bit read sequence is completed, even if the user reads the MSB several times.

After a complete reading sequence, if only the CLR register or ACLR register are read, they return the LSB of the count value at the time of the read.

An overflow occurs when the counter rolls over from FFFFh to 0000h then:

– The TOF bit of the SR register is set.

– A timer interrupt is generated if the TOIE bit of the CR1 register and the EFTIS bit of the CR3 register are set.

If one of these conditions is false, the interrupt remains pending to be issued as soon as they are both true.

Clearing the overflow interrupt request is done by:

1. Reading the SR register while the TOF bit is set.
2. An access (read or write) to the CLR register.

Notes: The TOF bit is not cleared by accesses to ACLR register. This feature allows simultaneous use of the overflow function and reads of the free running counter at random times (for example, to measure elapsed time) without the risk of clearing the TOF bit erroneously.

The timer is not affected by WAIT mode.

In HALT mode, the counter stops counting until the mode is exited. Counting then resumes from the reset count (MCU awakened by a Reset).

10.3.3.2 External Clock

The external clock (where available) is selected if CC0=1 and CC1=1 in CR2 register.

The status of the EXEDG bit determines the type of level transition on the external clock pin EXTCLK that will trigger the free running counter.

The counter is synchronised with the falling edge of INTCLK.

At least four falling edges of the INTCLK must occur between two consecutive active edges of the external clock; thus the external clock frequency must be less than a quarter of the INTCLK frequency.

EXTENDED FUNCTION TIMER (Cont'd)

Figure 74. Counter Timing Diagram, INTCLK divided by 2

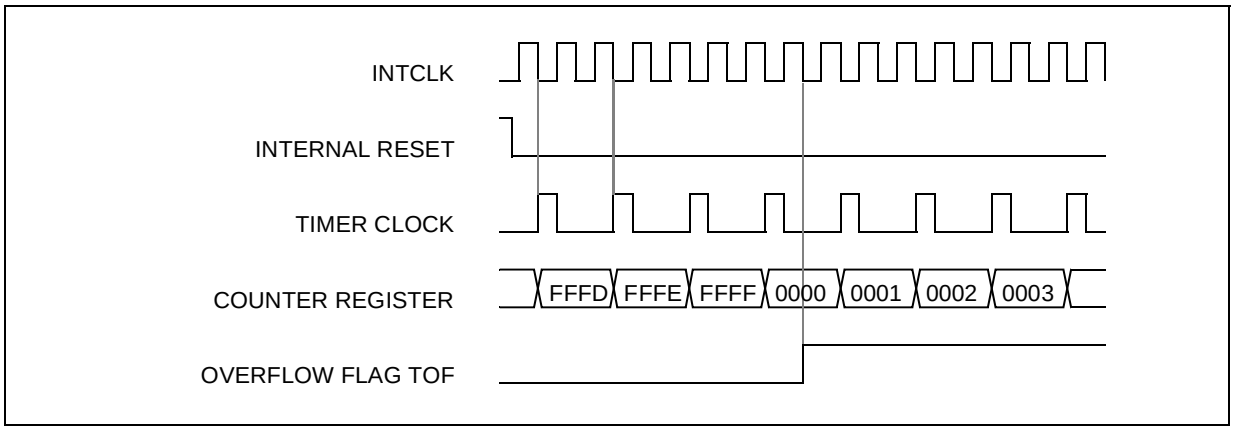


Figure 75. Counter Timing Diagram, INTCLK divided by 4

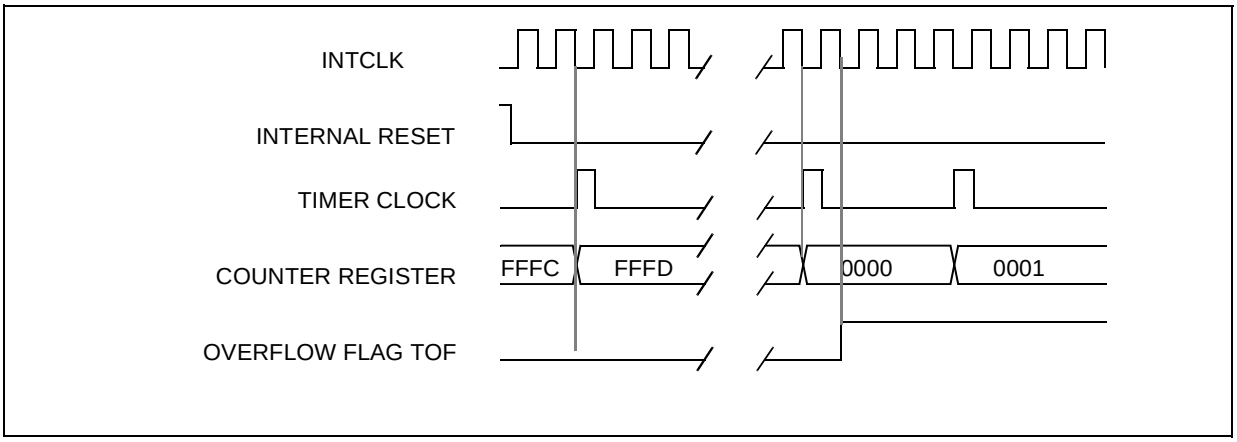
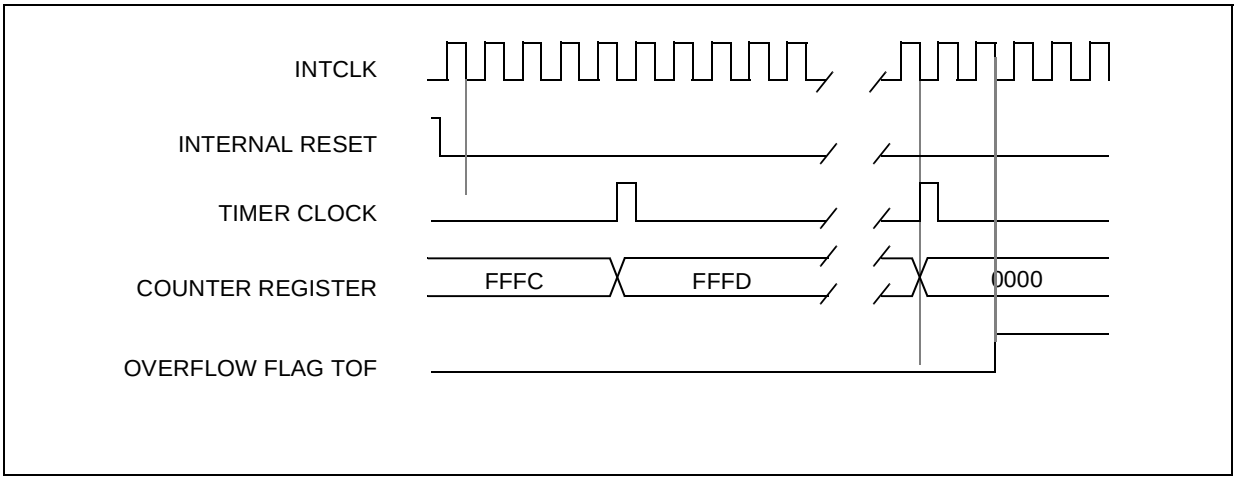


Figure 76. Counter Timing Diagram, INTCLK divided by 8



EXTENDED FUNCTION TIMER (Cont'd)

10.3.3.3 Input Capture

In this section, the index, *i*, may be 1 or 2.

The two input capture 16-bit registers (IC1R and IC2R) are used to latch the value of the free running counter after a transition detected by the ICAP*i* pin (see figure 5).

	MS Byte	LS Byte
IC*i*R	**IC*i*HR**	**IC*i*LR**

IC*i* Rregister is a read-only register.

The active transition is software programmable through the IEDG*i* bit of the Control Register (CR*i*).

Timing resolution is one count of the free running counter: ($INTCLK_{/CC[1:0]}$).

Procedure

To use the input capture function select the following in the CR2 register:

– Select the timer clock (CC[1:0] (see Table 28).

– Select the edge of the active transition on the ICAP2 pin with the IEDG2 bit.

And select the following in the CR1 register:

– Set the ICIE bit and the EFTIS bit to generate an interrupt after an input capture.

– Select the edge of the active transition on the ICAP1 pin with the IEDG1 bit.

When an input capture occurs:

– ICF*i* bit is set.

– The IC*i*R register contains the value of the free running counter on the active transition on the ICAP*i* pin (see Figure 78).

– A timer interrupt is generated if the ICIE bit and the EFTIS bit are set. Otherwise, the interrupt remains pending until both conditions become true.

Clearing the Input Capture interrupt request is done by:

1. Reading the SR register while the ICF*i* bit is set.

2. An access (read or write) to the IC*i*LR register.

Note: After reading the IC*i*HR register, transfer of input capture data is inhibited until the IC*i*LR register is also read.

The IC*i*R register always contains the free running counter value which corresponds to the most recent input capture.

EXTENDED FUNCTION TIMER (Cont'd)

Figure 77. Input Capture Block Diagram

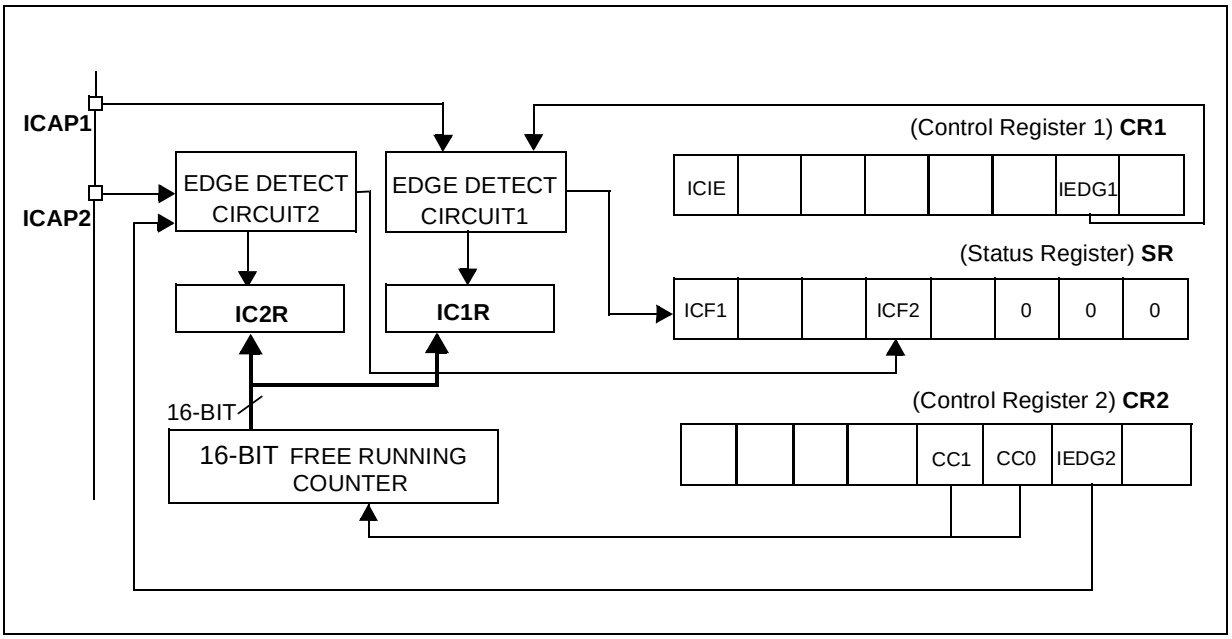
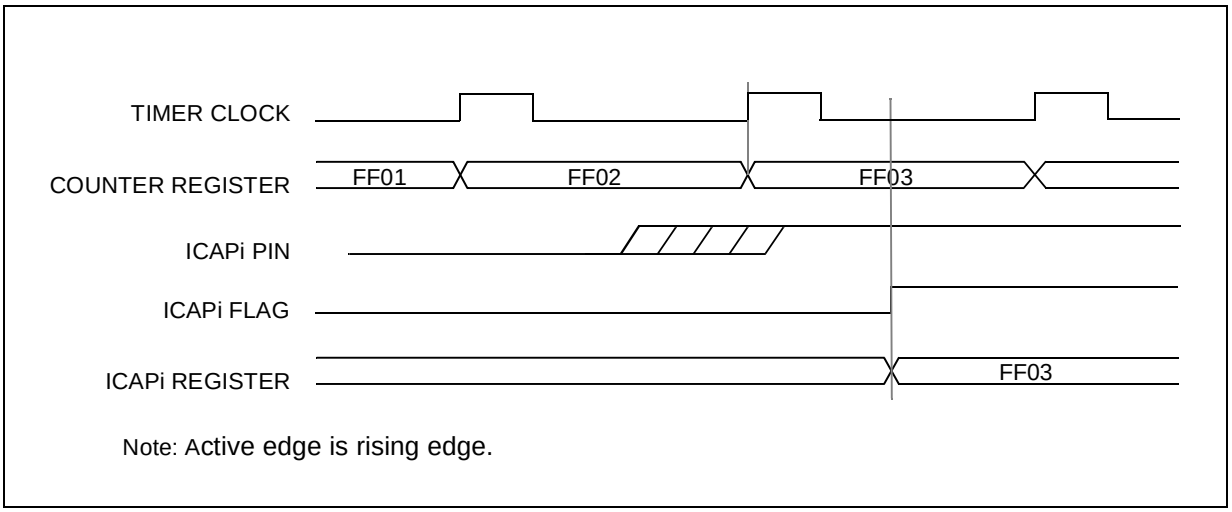


Figure 78. Input Capture Timing Diagram



Note: Active edge is rising edge.

EXTENDED FUNCTION TIMER (Cont'd)

10.3.3.4 Output Compare

In this section, the index, *i*, may be 1 or 2.

This function can be used to control an output waveform or indicating when a period of time has elapsed.

When a match is found between the Output Compare register and the free running counter, the output compare function:

- Assigns pins with a programmable value if the OC*i*E bit is set
- Sets a flag in the status register
- Generates an interrupt if enabled

Two 16-bit registers Output Compare Register 1 (OC1R) and Output Compare Register 2 (OC2R) contain the value to be compared to the free running counter each timer clock cycle.

	MS Byte	LS Byte
OC*i*R	OC*i*HR	OC*i*LR

These registers are readable and writable and are not affected by the timer hardware. A reset event changes the OC*i*R value to 8000h.

Timing resolution is one count of the free running counter: ($INTCLK_{/CC[1:0]}$).

Procedure

To use the output compare function, select the following in the CR2 register:

- Set the OC*i*E bit if an output is needed then the OCMP*i* pin is dedicated to the output compare *i* function.
- Select the timer clock CC[1:0] (see Table 28).

And select the following in the CR1 register:

- Select the OLVL*i* bit to applied to the OCMP*i* pins after the match occurs.
- Set the OCIE and EFTIS bit to generate an interrupt if it is needed.

When match is found:

- OCF*i* bit is set.
- The OCMP*i* pin takes OLVL*i* bit value (OCMP*i* pin latch is forced low during reset and stays low until valid compares change it to OLVL*i* level).
- A timer interrupt is generated if the OCIE bit in the CR2 register and the EFTIS bit in the CR3 register are set.

Clearing the output compare interrupt request is done by:

3. Reading the SR register while the OCF*i* bit is set.
4. An access (read or write) to the OC*i*LR register.

Note: After a processor write cycle to the OC*i*HR register, the output compare function is inhibited until the OC*i*LR register is also written.

If the OC*i*E bit is not set, the OCMP*i* pin is a general I/O port and the OLVL*i* bit will not appear when match is found but an interrupt could be generated if the OCIE bit is set.

The value in the 16-bit OC*i*R register and the OLVL*i* bit should be changed after each successful comparison in order to control an output waveform or establish a new elapsed timeout.

The OC*i*R register value required for a specific timing application can be calculated using the following formula:

$$\Delta \text{ OC}i\text{R} = \frac{\Delta t * \text{INTCLK}}{(\text{CC1.CC0})}$$

Where:

Δt = Desired output compare period (in seconds)

INTCLK = Internal clock frequency

CC1-CC0 = Timer clock prescaler

The following procedure is recommended to prevent the OCF*i* bit from being set between the time it is read and the write to the OC*i*R register:

- Write to the OC*i*HR register (further compares are inhibited).
- Read the SR register (first step of the clearance of the OCF*i* bit, which may be already set).
- Write to the OC*i*LR register (enables the output compare function and clears the OCF*i* bit).

EXTENDED FUNCTION TIMER (Cont'd)

Figure 79. Output Compare Block Diagram

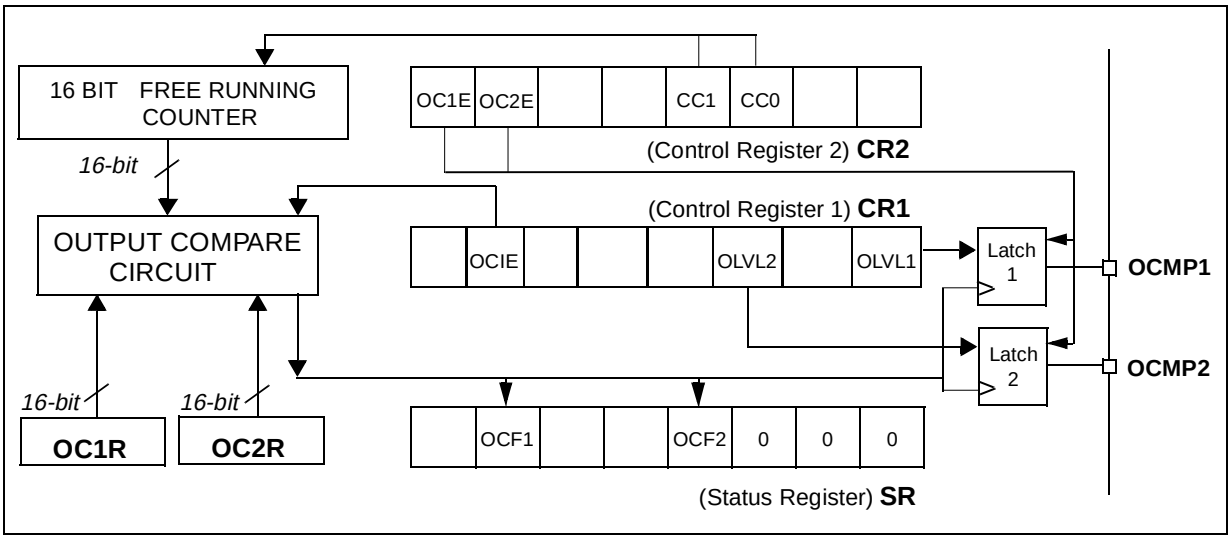
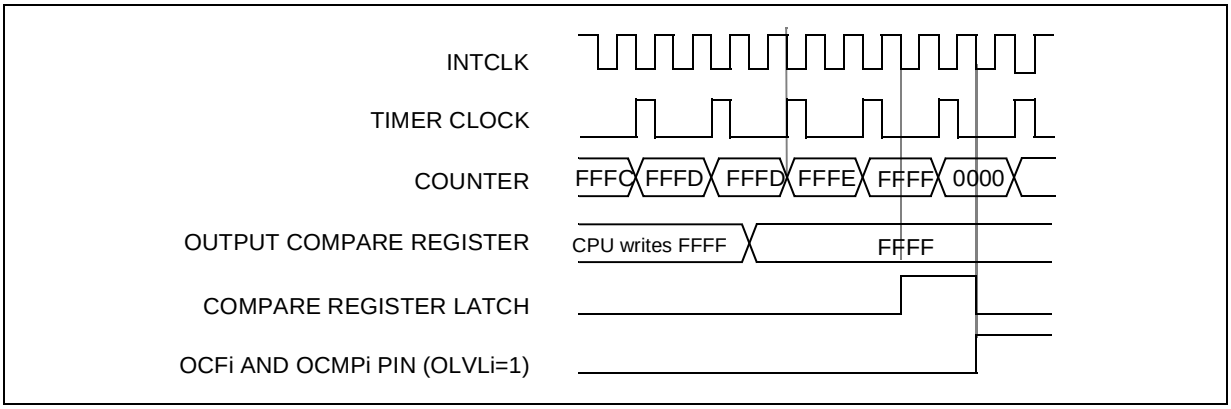


Figure 80. Output Compare Timing Diagram, Internal Clock Divided by 2



EXTENDED FUNCTION TIMER (Cont'd)

10.3.3.5 Forced Compare Mode

In this section *i* may represent 1 or 2.

The following bits of the CR1 register are used:

			FOLV2	FOLV1	OLVL2		OLVL1

When the FOLV*i* bit is set, the OLVL*i* bit is copied to the OCMP*i* pin. The OLVL*i* bit has to be toggled in order to toggle the OCMP*i* pin when it is enabled (OC*i*E bit=1).

The OCF*i* bit is not set, and thus no interrupt request is generated.

10.3.3.6 One Pulse Mode

One Pulse mode enables the generation of a pulse when an external event occurs. This mode is selected via the OPM bit in the CR2 register.

The one pulse mode uses the Input Capture1 function and the Output Compare1 function.

Procedure

To use one pulse mode, select the following in the the CR1 register:

– Using the OLVL1 bit, select the level to be applied to the OCMP1 pin after the pulse.

– Using the OLVL2 bit, select the level to be applied to the OCMP1 pin during the pulse.

– Select the edge of the active transition on the ICAP1 pin with the IEDG1 bit.

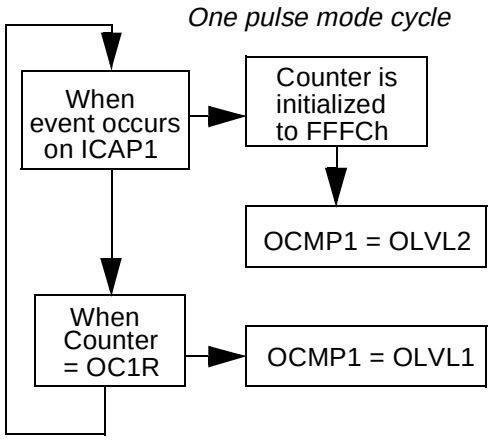
And select the following in the CR2 register:

– Set the OC1E bit, the OCMP1 pin is then dedicated to the Output Compare 1 function.

– Set the OPM bit.

– Select the timer clock CC[1:0] (see Table 28).

Load the OC1R register with the value corresponding to the length of the pulse (see the formula in Section 10.3.3.7).



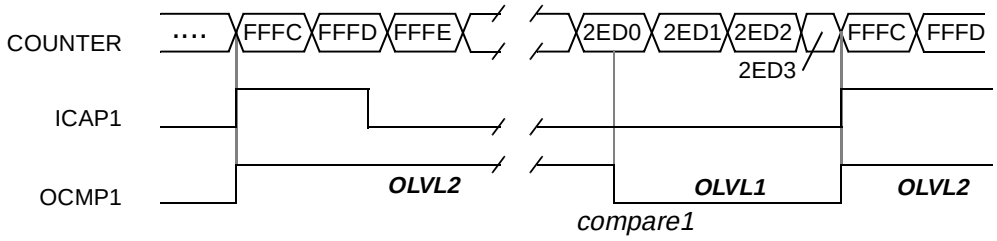
Then, on a valid event on the ICAP1 pin, the counter is initialized to FFFCh and OLVL2 bit is loaded on the OCMP1 pin. When the value of the counter is equal to the value of the contents of the OC1R register, the OLVL1 bit is output on the OCMP1 pin, (See Figure 81).

Note: The OCF1 bit cannot be set by hardware in one pulse mode but the OCF2 bit can generate an Output Compare interrupt.

The ICF1 bit is set when an active edge occurs and can generate an interrupt if the ICIE bit is set.

When the Pulse Width Modulation (PWM) and One Pulse Mode (OPM) bits are both set, the PWM mode is the only active one.

Figure 81. One Pulse Mode Timing



Note: IEDG1=1, OC1R=2ED0h, OLVL1=0, OLVL2=1

EXTENDED FUNCTION TIMER (Cont'd)

10.3.3.7 Pulse Width Modulation Mode

Pulse Width Modulation mode enables the generation of a signal with a frequency and pulse length determined by the value of the OC1R and OC2R registers.

The pulse width modulation mode uses the complete Output Compare 1 function plus the OC2R register.

Procedure

To use pulse width modulation mode select the following in the CR1 register:

– Using the OLVL1 bit, select the level to be applied to the OCMP1 pin after a successful comparison with OC1R register.

– Using the OLVL2 bit, select the level to be applied to the OCMP1 pin after a successful comparison with OC2R register.

And select the following in the CR2 register:

– Set OC1E bit: the OCMP1 pin is then dedicated to the output compare 1 function.

– Set the PWM bit.

– Select the timer clock CC[1:0] bits (see Table 28).

Load the OC2R register with the value corresponding to the period of the signal.

Load the OC1R register with the value corresponding to the length of the pulse if (OLVL1=0 and OLVL2=1).

If OLVL1=1 and OLVL2=0 the length of the pulse is the difference between the OC2R and OC1R registers.

The OC*i*R register value required for a specific timing application can be calculated using the following formula:

$$\text{OC}i\text{R Value} = \frac{t * \text{INTCLK}}{\text{CC[1:0]}} - 5$$

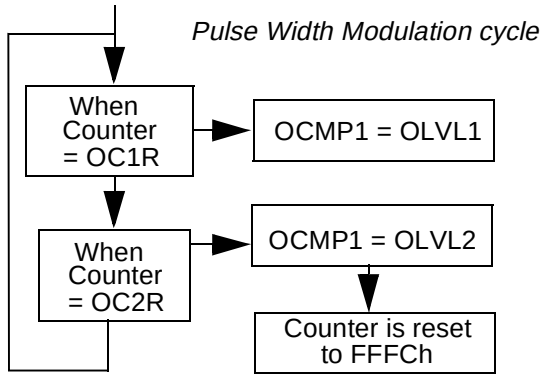
Where:

– t = Desired output compare period (seconds)

– INTCLK = Internal clock frequency

– CC1-CC0 = Timer clock prescaler

The Output Compare 2 event causes the counter to be initialized to FFFCh (See Figure 82).

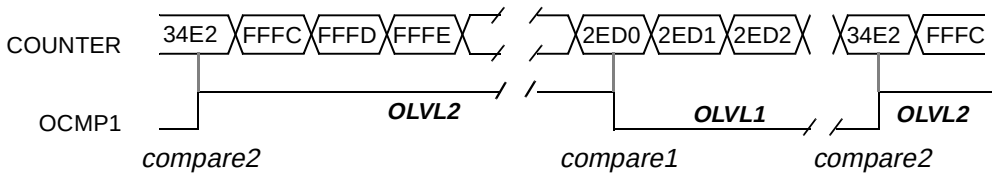


Note: After a write instruction to the OC*i*HR register, the output compare function is inhibited until the OC*i*LR register is also written.

The OCF1 and OCF2 bits cannot be set by hardware in PWM mode therefore the Output Compare interrupt is inhibited. The Input Capture interrupts are available.

When the Pulse Width Modulation (PWM) and One Pulse Mode (OPM) bits are both set, the PWM mode is the only active one.

Figure 82. Pulse Width Modulation Mode Timing



Note: OC1R=2ED0h, OC2R=34E2, OLVL1=0, OLVL2= 1

EXTENDED FUNCTION TIMER (Cont'd)

10.3.4 Interrupt Management

The interrupts of the Extended Function Timers are mapped on external interrupt channels of the microcontroller (refer to the "Interrupts" chapter).

The five interrupt sources (2 input captures, 2 output compares and overflow) are mapped on the same interrupt channel.

Each External Interrupt Channel has:

– A trigger control bit in the EITR register (R242 - Page 0)

– A pending bit in the EIPR register (R243 - Page 0)

– A mask bit in the EIMR register (R244 - Page 0)

Program the interrupt priority level using the EIPLR register (R245 - Page 0). For a description of these registers refer to the "Interrupts" and "DMA" chapters.

To use the interrupt features, perform the following sequence:

– Set the priority level of the interrupt channel used (EIPLR register)

– Select the interrupt trigger edge as rising edge (set the corresponding bit in the EITR register)

– Set the EFTIS bit of the CR3 register to select the peripheral interrupt sources

– Set the OCIE and/or ICIE and/or TOIE bit(s) of the CR1 register to enable the peripheral to perform interrupt requests on the wanted events

– In the EIPR register, reset the pending bit of the interrupt channel used by the peripheral interrupts to avoid any spurious interrupt requests being performed when the mask bits is set

– Set the mask bits of the interrupt channels used to enable the MCU to acknowledge the interrupt requests of the peripheral.

Caution: Care should be taken when using only one of the input capture pins, as both capture interrupts are enabled by the ICIE bit in the CR1 register. If only ICAP1 is used (for example), an interrupt can still be generated by the ICAP2 pin when this pin toggles, even if it is configured as a standard output. If this case, the interrupt capture status bits in the SR register should handled in polling mode.

Caution:

1. It is mandatory to clear all EFT interrupt flags simultaneously at least once before exiting an EFT timer interrupt routine (the SR register must = 00h at some point during the interrupt routine), otherwise no interrupts can be issued on that channel anymore.
   Refer to the following assembly code for an interrupt sequence example.

2. Since a loop statement is needed inside the IT routine, the user must avoid situations where an interrupt event period is narrower than the duration of the interrupt treatment. Otherwise nested interrupt mode must be used to serve higher priority requests.

EXTENDED FUNCTION TIMER (Cont'd)

Note: A single access (read/write) to the SR register at the beginning of the interrupt routine is the first step needed to clear all the EFT interrupt flags. In a second step, the lower bytes of the data registers must be accessed if the corresponding flag is set. It is not necessary to access the SR register between these instructions, but it can done.

```
; INTERRUPT ROUTINE EXAMPLE
            push R234          ; Save current page
            spp #28            ; Set EFT page
L6:
            cp R254,#0         ; while E0_SR is not cleared
            jxz L7
            tm R254,#128       ; Check Input Capture 1 flag
            jxz L2             ; else go to next test
            ld r1,R241         ; Dummy read to clear IC1LR
            ; Insert your code here
L2:
            tm R254,#16       ; Check Input Capture 2 flag
            jxz L3             ; else go to next test
            ld r1,R243         ; Dummy read to clear IC2LR
            ; Insert your code here
L3:
            tm R254,#64       ; Check Input Compare 1 flag
            jxz L4             ; else go to next test
            ld r1,R249         ; Dummy read to clear OC1LR
            ; Insert your code here
L4:
            tm R254,#8        ; Check Input Compare 2 flag
            jxz L5             ; else go to next test
            ld r1,R251         ; Dummy read to clear OC1LR
            ; Insert your code here
L5:
            tm R254,#32        ; Check Input Overflow flag
            jxz L6             ; else go to next test
            ld r1,R245         ; Dummy read to clear Overflow flag
            ; Insert your code here
            jx L6
L7:
            pop R234           ; Restore current page
iret
```

EXTENDED FUNCTION TIMER (Cont'd)

10.3.5 Register Description

Each Timer is associated with three control and one status registers, and with six pairs of data registers (16-bit values) relating to the two input captures, the two output compares, the counter and the alternate counter.

Notes:

1. In the register description on the following pages, register and page numbers are given using the example of Timer 0. On devices with more than one timer, refer to the device register map for the adresses and page numbers.

2. To work correctly with register pairs, it is strongly recommended to use single byte instructions. Do not use word instructions to access any of the 16-bit registers.

INPUT CAPTURE 1 HIGH REGISTER (IC1HR)

R240 - Read Only
Register Page: 28
Reset Value: Undefined

This is an 8-bit read only register that contains the high part of the counter value (transferred by the input capture 1 event).

7							0
MSB							LSB

INPUT CAPTURE 1 LOW REGISTER (IC1LR)

R241 - Read Only
Register Page: 28
Reset Value: Undefined

This is an 8-bit read only register that contains the low part of the counter value (transferred by the input capture 1 event).

7							0
MSB							LSB

INPUT CAPTURE 2 HIGH REGISTER (IC2HR)

R242 - Read Only
Register Page: 28
Reset Value: Undefined

This is an 8-bit read only register that contains the high part of the counter value (transferred by the Input Capture 2 event).

7							0
MSB							LSB

INPUT CAPTURE 2 LOW REGISTER (IC2LR)

R243 - Read Only
Register Page: 28
Reset Value: Undefined

This is an 8-bit read only register that contains the low part of the counter value (transferred by the Input Capture 2 event).

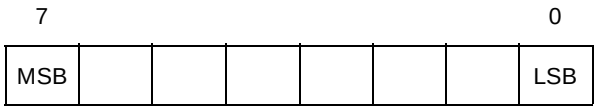
7							0
MSB							LSB

EXTENDED FUNCTION TIMER (Cont'd)

COUNTER HIGH REGISTER (CHR)

R244 - Read Only
Register Page: 28
Reset Value: 1111 1111 (FFh)

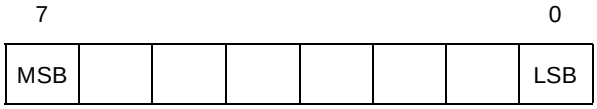
This is an 8-bit register that contains the high part of the counter value.



COUNTER LOW REGISTER (CLR)

R245 - Read/Write
Register Page: 28
Reset Value: 1111 1100 (FCh)

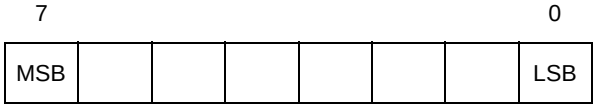
This is an 8-bit register that contains the low part of the counter value. A write to this register resets the counter. An access to this register after accessing the SR register clears the TOF bit.



ALTERNATE COUNTER HIGH REGISTER (ACHR)

R246 - Read Only
Register Page: 28
Reset Value: 1111 1111 (FFh)

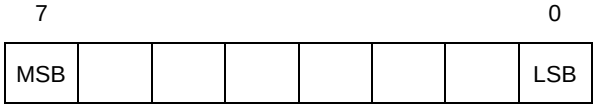
This is an 8-bit register that contains the high part of the counter value.



ALTERNATE COUNTER LOW REGISTER (ACLR)

R247 - Read/Write
Register Page: 28
Reset Value: 1111 1100 (FCh)

This is an 8-bit register that contains the low part of the counter value. A write to this register resets the counter. An access to this register after an access to SR register does not clear the TOF bit in SR register.



EXTENDED FUNCTION TIMER (Cont'd)

OUTPUT COMPARE 1 HIGH REGISTER (OC1HR)

R248 - Read/Write
Register Page: 28
Reset Value: 1000 0000 (80h)

This is an 8-bit register that contains the high part of the value to be compared to the CHR register.

7							0
MSB							LSB

OUTPUT COMPARE 2 HIGH REGISTER (OC2HR)

R250 - Read/Write
Register Page: 28
Reset Value: 1000 0000 (80h)

This is an 8-bit register that contains the high part of the value to be compared to the CHR register.

7							0
MSB							LSB

OUTPUT COMPARE 1 LOW REGISTER (OC1LR)

R249 - Read/Write
Register Page: 28
Reset Value: 0000 0000 (00h)

This is an 8-bit register that contains the low part of the value to be compared to the CLR register.

7							0
MSB							LSB

OUTPUT COMPARE 2 LOW REGISTER (OC2LR)

R251 - Read/Write
Register Page: 28
Reset Value: 0000 0000 (00h)

This is an 8-bit register that contains the low part of the value to be compared to the CLR register.

7							0
MSB							LSB

EXTENDED FUNCTION TIMER (Cont'd)

CONTROL REGISTER 1 (CR1)

R252 - Read/Write
Register Page: 28
Reset Value: 0000 0000 (00h)

7							0
ICIE	OCIE	TOIE	FOLV2	FOLV1	OLVL2	IEDG1	OLVL1

Bit 7 = **ICIE** *Input Capture Interrupt Enable.*
0: Interrupt is inhibited.
1: A timer interrupt is generated whenever the ICF1 or ICF2 bit of the SR register is set.

Bit 6 = **OCIE** *Output Compare Interrupt Enable.*
0: Interrupt is inhibited.
1: A timer interrupt is generated whenever the OCF1 or OCF2 bit of the SR register is set.

Bit 5 = **TOIE** *Timer Overflow Interrupt Enable.*
0: Interrupt is inhibited.
1: A timer interrupt is enabled whenever the TOF bit of the SR register is set.

Bit 4 = **FOLV2** *Forced Output Compare 2.*
0: No effect.
1: Forces the OLVL2 bit to be copied to the OCMP2 pin.

Bit 3 = **FOLV1** *Forced Output Compare 1.*
0: No effect.
1: Forces OLVL1 to be copied to the OCMP1 pin.

Bit 2 = **OLVL2** *Output Level 2.*
This bit is copied to the OCMP2 pin whenever a successful comparison occurs with the OC2R register and OC2E is set in the CR2 register. This value is copied to the OCMP1 pin in One Pulse Mode and Pulse Width Modulation mode.

Bit 1 = **IEDG1** *Input Edge 1.*
This bit determines which type of level transition on the ICAP1 pin will trigger the capture.
0: A falling edge triggers the capture.
1: A rising edge triggers the capture.

Bit 0 = **OLVL1** *Output Level 1.*
The OLVL1 bit is copied to the OCMP1 pin whenever a successful comparison occurs with the OC1R register and the OC1E bit is set in the CR2 register.

EXTENDED FUNCTION TIMER (Cont'd)

CONTROL REGISTER 2 (CR2)

R253 - Read/Write
Register Page: 28
Reset Value: 0000 0000 (00h)

7							0
OC1E	OC2E	OPM	PWM	CC1	CC0	IEDG2	EXEDG

Bit 7 = **OC1E** *Output Compare 1 Enable.*
0: Output Compare 1 function is enabled, but the OCMP1 pin is a general I/O.
1: Output Compare 1 function is enabled, the OCMP1 pin is dedicated to the Output Compare 1 capability of the timer.

Bit 6 = **OC2E** *Output Compare 2 Enable.*
0: Output Compare 2 function is enabled, but the OCMP2 pin is a general I/O.
1: Output Compare 2 function is enabled, the OCMP2 pin is dedicated to the Output Compare 2 capability of the timer.

Bit 5 = **OPM** *One Pulse Mode.*
0: One Pulse Mode is not active.
1: One Pulse Mode is active, the ICAP1 pin can be used to trigger one pulse on the OCMP1 pin; the active transition is given by the IEDG1 bit. The length of the generated pulse depends on the contents of the OC1R register.

Bit 4 = **PWM** *Pulse Width Modulation.*
0: PWM mode is not active.
1: PWM mode is active, the OCMP1 pin outputs a programmable cyclic signal; the length of the pulse depends on the value of OC1R register; the period depends on the value of OC2R register.

Bit 3, 2 = **CC[1:0]** *Clock Control.*

The value of the timer clock depends on these bits:

Table 28. Clock Control Bits

CC1	CC0	Timer Clock
0	0	INTCLK / 4
0	1	INTCLK / 2
1	0	INTCLK / 8
1	1	External Clock

Bit 1 = **IEDG2** *Input Edge 2.*
This bit determines which type of level transition on the ICAP2 pin will trigger the capture.
0: A falling edge triggers the capture.
1: A rising edge triggers the capture.

Bit 0 = **EXEDG** *External Clock Edge.*
This bit determines which type of level transition on the external clock pin EXTCLK will trigger the free running counter.
0: A falling edge triggers the free running counter.
1: A rising edge triggers the free running counter.

EXTENDED FUNCTION TIMER (Cont'd)

STATUS REGISTER (SR)

R254 - Read Only
Register Page: 28
Reset Value: 0000 0000 (00h)

The three least significant bits are not used.

7							0
ICF1	OCF1	TOF	ICF2	OCF2	0	0	0

Bit 7 = **ICF1** *Input Capture Flag 1.*
0: No input capture (reset value).
1: An input capture has occurred. To clear this bit, first read the SR register, then read or write the low byte of the IC1R (IC1LR) register.

Bit 6 = **OCF1** *Output Compare Flag 1.*
0: No match (reset value).
1: The content of the free running counter has matched the content of the OC1R register. To clear this bit, first read the SR register, then read or write the low byte of the OC1R (OC1LR) register.

Bit 5 = **TOF** *Timer Overflow.*
0: No timer overflow (reset value).
1: The free running counter rolled over from FFFFh to 0000h. To clear this bit, first read the SR register, then read or write the low byte of the CR (CLR) register.

Note: Reading or writing the ACLR register does not clear TOF.

Bit 4 = **ICF2** *Input Capture Flag 2.*
0: No input capture (reset value).
1: An input capture has occurred. To clear this bit, first read the SR register, then read or write the low byte of the IC2R (IC2LR) register.

Bit 3 = **OCF2** *Output Compare Flag 2.*
0: No match (reset value).
1: The content of the free running counter has matched the content of the OC2R register. To clear this bit, first read the SR register, then read or write the low byte of the OC2R (OC2LR) register.

Bit 2-0 = Reserved, forced by hardware to 0.

CONTROL REGISTER 3 (CR3)

R255 - Read/Write
Register Page: 28
Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	0	0	0	EFTIS

Bits 7:1 = **Unused** *Read as 0.*

Bit 0 = **EFTIS** *Global Timer Interrupt Selection.*
0: Select External interrupt.
1: Select Global Timer Interrupt

EXTENDED FUNCTION TIMER (Cont'd)

Table 29. Extended Function Timer Register Map

Address (Dec.)	Register Name	7	6	5	4	3	2	1	0
R240	**IC1HR**	MSB							LSB
	Reset Value	x	x	x	x	x	x	x	x
R241	**IC1LR**	MSB							LSB
	Reset Value	x	x	x	x	x	x	x	x
R242	**IC2HR**	MSB							LSB
	Reset Value	x	x	x	x	x	x	x	x
R243	**IC2LR**	MSB							LSB
	Reset Value	x	x	x	x	x	x	x	x
R244	**CHR**	MSB							LSB
	Reset Value	1	1	1	1	1	1	1	1
R245	**CLR**	MSB							LSB
	Reset Value	1	1	1	1	1	1	0	0
R246	**ACHR**	MSB							LSB
	Reset Value	1	1	1	1	1	1	1	1
R247	**ACLR**	MSB							LSB
	Reset Value	1	1	1	1	1	1	0	0
R248	**OC1HR**	MSB							LSB
	Reset Value	1	0	0	0	0	0	0	0
R249	**OC1LR**	MSB							LSB
	Reset Value	0	0	0	0	0	0	0	0
R250	**OC2HR**	MSB							LSB
	Reset Value	1	0	0	0	0	0	0	0
R251	**OC2LR**	MSB							LSB
	Reset Value	0	0	0	0	0	0	0	0
R252	**CR1**	OC1E	OC2E	OPM	PWM	CC1	CC0	IEDG2	EXEDG
	Reset Value	0	0	0	0	0	0	0	0
R253	**CR2**	ICIE	OCIE	TOIE	FOLV2	FOLV1	OLVL2	IEDG1	OLVL1
	Reset Value	0	0	0	0	0	0	0	0
R254	**SR**	ICF1	OCF1	TOF	ICF2	OCF2			
	Reset Value	0	0	0	0	0	0	0	0
R255	**CR3**								EFTIS
	Reset Value	0	0	0	0	0	0	0	0

EXTENDED FUNCTION TIMER (Cont'd)

Table 30. Extended Function Timer Page Map

Timer number	Page (hex)
EFT0	**1C**
EFT1	**1D**

10.4 MULTIFUNCTION TIMER (MFT)

10.4.1 Introduction

The Multifunction Timer (MFT) peripheral offers powerful timing capabilities and features 12 operating modes, including automatic PWM generation and frequency measurement.

The MFT comprises a 16-bit Up/Down counter driven by an 8-bit programmable prescaler. The input clock may be INTCLK/3 or an external source. The timer features two 16-bit Comparison Registers, and two 16-bit Capture/Load/Reload Registers. Two input pins and two alternate function output pins are available.

Several functional configurations are possible, for instance:

– 2 input captures on separate external lines, and 2 independent output compare functions with the counter in free-running mode, or 1 output compare at a fixed repetition rate.

– 1 input capture, 1 counter reload and 2 independent output compares.

– 2 alternate autoreloads and 2 independent output compares.

– 2 alternate captures on the same external line and 2 independent output compares at a fixed repetition rate.

When two MFTs are present in an ST9 device, a combined operating mode is available.

An internal On-Chip Event signal can be used on some devices to control other on-chip peripherals.

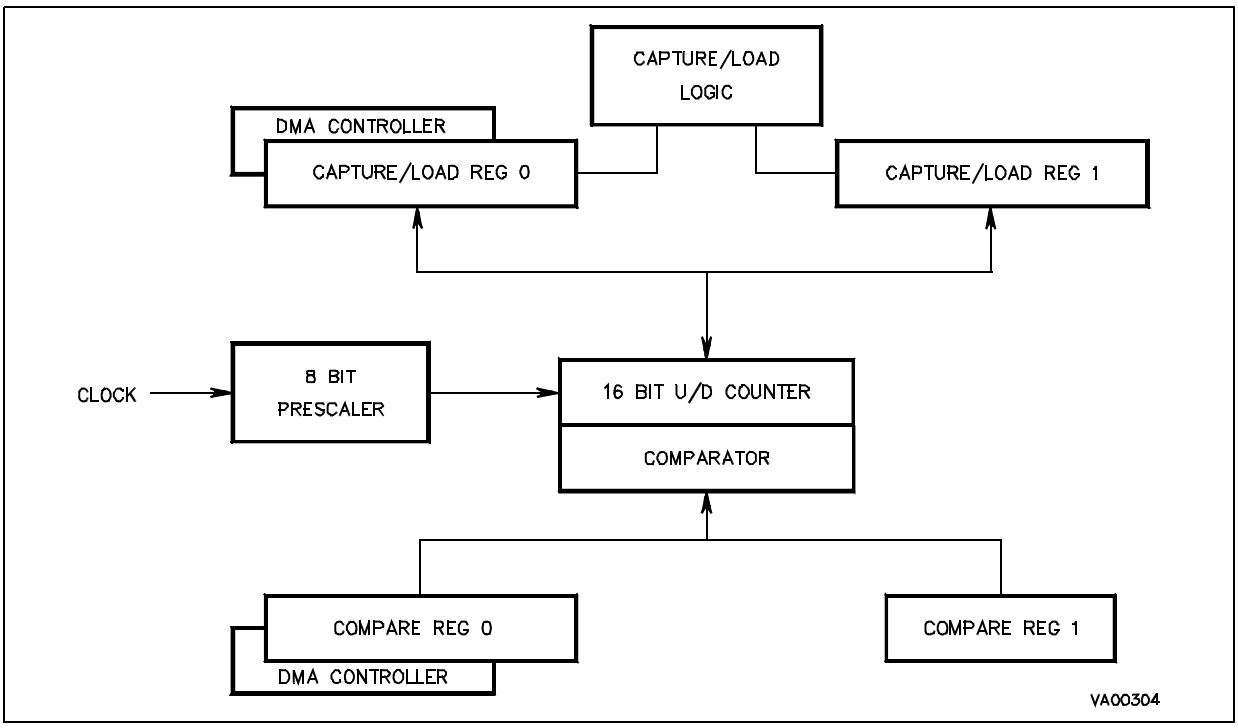
The two external inputs may be individually programmed to detect any of the following:

– rising edges

– falling edges

– both rising and falling edges

Figure 83. MFT Simplified Block Diagram



MULTIFUNCTION TIMER (Cont'd)

The configuration of each input is programmed in the Input Control Register.

Each of the two output pins can be driven from any of three possible sources:

– Compare Register 0 logic

– Compare Register 1 logic

– Overflow/Underflow logic

Each of these three sources can cause one of the following four actions, independently, on each of the two outputs:

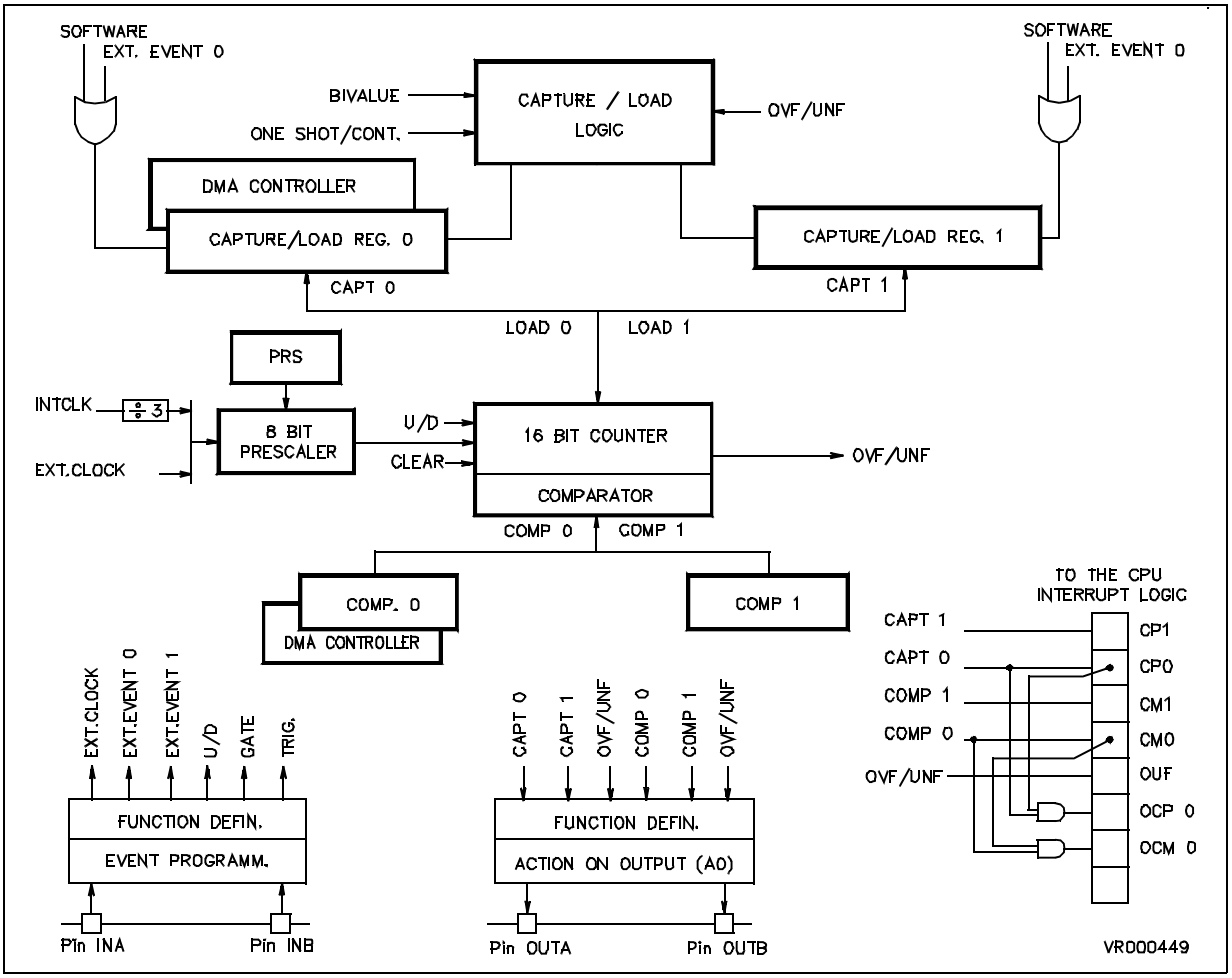
– Nop, Set, Reset, Toggle

In addition, an additional On-Chip Event signal can be generated by two of the three sources mentioned above, i.e. Over/Underflow event and Compare 0 event. This signal can be used internally to synchronise another on-chip peripheral. Five maskable interrupt sources referring to an End Of Count condition, 2 input captures and 2 output compares, can generate 3 different interrupt requests (with hardware fixed priority), pointing to 3 interrupt routine vectors.

Two independent DMA channels are available for rapid data transfer operations. Each DMA request (associated with a capture on the REG0R register, or with a compare on the CMP0R register) has priority over an interrupt request generated by the same source.

A SWAP mode is also available to allow high speed continuous transfers (see Interrupt and DMA chapter).

Figure 84. Detailed Block Diagram



MULTIFUNCTION TIMER (Cont'd)

10.4.2 Functional Description

The MFT operating modes are selected by programming the Timer Control Register (TCR) and the Timer Mode Register (TMR).

10.4.2.1 Trigger Events

A trigger event may be generated by software (by setting either the CP0 or the CP1 bits in the T_FLAGR register) or by an external source which may be programmed to respond to the rising edge, the falling edge or both by programming bits A0-A1 and B0-B1 in the T_ICR register. This trigger event can be used to perform a capture or a load, depending on the Timer mode (configured using the bits in Table 4).

An event on the TxINA input or setting the CP0 bit triggers a capture to, or a load from the REG0R register (except in Bicapture mode, see Section 0.1.2.11).

An event on the TxINB input or setting the CP1 bit triggers a capture to, or a load from the REG1R register.

In addition, in the special case of "Load from REG0R and monitor on REG1R", it is possible to use the TxINB input as a trigger for REG0R."

10.4.2.2 One Shot Mode

When the counter generates an overflow (in up-count mode), or an underflow (in down-count mode), that is to say when an End Of Count condition is reached, the counter stops and no counter reload occurs. The counter may only be restarted by an external trigger on TxINA or B or a by software trigger on CP0 only. One Shot Mode is entered by setting the CO bit in TMR.

10.4.2.3 Continuous Mode

Whenever the counter reaches an End Of Count condition, the counting sequence is automatically restarted and the counter is reloaded from REG0R (or from REG1R, when selected in Biload Mode). Continuous Mode is entered by resetting the C0 bit in TMR.

10.4.2.4 Triggered And Retriggered Modes

A triggered event may be generated by software (by setting either the CP0 or the CP1 bit in the T_FLAGR register), or by an external source which may be programmed to respond to the rising edge, the falling edge or both, by programming bits A0-A1 and B0-B1 in T_ICR.

In One Shot and Triggered Mode, every trigger event arriving before an End Of Count, is masked. In One Shot and Retriggered Mode, every trigger received while the counter is running, automatically reloads the counter from REG0R. Triggered/Retriggered Mode is set by the REN bit in TMR.

The TxINA input refers to REG0R and the TxINB input refers to REG1R.

WARNING. If the Triggered Mode is selected when the counter is in Continuous Mode, every trigger is disabled, it is not therefore possible to synchronise the counting cycle by hardware or software.

10.4.2.5 Gated Mode

In this mode, counting takes place only when the external gate input is at a logic low level. The selection of TxINA or TxINB as the gate input is made by programming the IN0-IN3 bits in T_ICR.

10.4.2.6 Capture Mode

The REG0R and REG1R registers may be independently set in Capture Mode by setting RM0 or RM1 in TMR, so that a capture of the current count value can be performed either on REG0R or on REG1R, initiated by software (by setting CP0 or CP1 in the T_FLAGR register) or by an event on the external input pins.

WARNING. Care should be taken when two software captures are to be performed on the same register. In this case, at least one instruction must be present between the first CP0/CP1 bit set and the subsequent CP0/CP1 bit reset instructions.

10.4.2.7 Up/Down Mode

The counter can count up or down depending on the state of the UDC bit (Up/Down Count) in TCR, or on the configuration of the external input pins, which have priority over UDC (see Input pin assignment in T_ICR). The UDCS bit returns the counter up/down current status (see also the Up/Down Autodiscrimination mode in the Input Pin Assignment Section).

MULTIFUNCTION TIMER (Cont'd)

10.4.2.8 Free Running Mode

The timer counts continuously (in Up or Down mode) and the counter value simply overflows or underflows through FFFFh or zero; there is no End Of Count condition as such, and no reloading takes place. This mode is automatically selected either in Bi-capture mode or by setting register REG0R for a Capture function (Continuous mode must also be set). In Autoclear mode, free running operation can be selected, with the possibility of choosing a maximum count value less than 2^{16} before overflow or underflow (see Autoclear mode).

10.4.2.9 Monitor Mode

When the RM1 bit in TMR is reset, and the timer is not in Bi-value mode, REG1R acts as a monitor, duplicating the current up or down counter contents, thus allowing the counter to be read "on the fly".

10.4.2.10 Autoclear Mode

A clear command forces the counter either to 0000h or to FFFFh, depending on whether up-counting or downcounting is selected. The counter reset may be obtained either directly, through the CCL bit in TCR, or by entering the Autoclear Mode, through the CCP0 and CCMP0 bits in TCR.

Every capture performed on REG0R (if CCP0 is set), or every successful compare performed by CMP0R (if CCMP0 is set), clears the counter and reloads the prescaler.

The Clear On Capture mode allows direct measurement of delta time between successive captures on REG0R, while the Clear On Compare mode allows free running with the possibility of choosing a maximum count value before overflow or underflow which is less than 2^{16} (see Free Running Mode).

10.4.2.11 Bi-value Mode

Depending on the value of the RM0 bit in TMR, the Bi-load Mode (RM0 reset) or the Bi-capture Mode (RM0 set) can be selected as illustrated in Figure 1 below:

Table 31. Bi-value Modes

TMR bits			Timer Operating Modes
RM0	RM1	BM	
0	X	1	Bi-Load mode
1	X	1	Bi-Capture Mode

A) Biload Mode

The Bi-load Mode is entered by selecting the Bi-value Mode (BM set in TMR) and programming REG0R as a reload register (RM0 reset in TMR).

At any End Of Count, counter reloading is performed alternately from REG0R and REG1R, (a low level for BM bit always sets REG0R as the current register, so that, after a Low to High transition of BM bit, the first reload is always from REG0R).

MULTIFUNCTION TIMER (Cont'd)

Every software or external trigger event on REG0R performs a reload from REG0R resetting the Biload cycle. In One Shot mode (reload initiated by software or by an external trigger), reloading is always from REG0R.

B) Bicapture Mode

The Bicapture Mode is entered by selecting the Bivalue Mode (the BM bit in TMR is set) and by programming REG0R as a capture register (the RM0 bit in TMR is set).

Interrupt generation can be configured as an AND or OR function of the two Capture events. This is configured by the A0 bit in the T_FLAGR register.

Every capture event, software simulated (by setting the CP0 flag) or coming directly from the TxINA input line, captures the current counter value alternately into REG0R and REG1R. When the BM bit is reset, REG0R is the current register, so that the first capture, after resetting the BM bit, is always into REG0R.

10.4.2.12 Parallel Mode

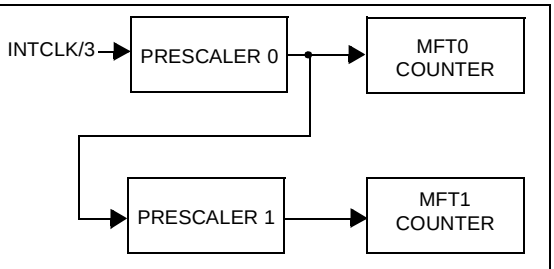
When two MFTs are present on an ST9 device, the parallel mode is entered when the ECK bit in the TMR register of Timer 1 is set. The Timer 1 prescaler input is internally connected to the Timer 0 prescaler output. Timer 0 prescaler input is connected to the system clock line.

By loading the Prescaler Register of Timer 1 with the value 00h the two timers (Timer 0 and Timer 1) are driven by the same frequency in parallel mode. In this mode the clock frequency may be divided by a factor in the range from 1 to 2^{16}.

10.4.2.13 Autodiscriminator Mode

The phase difference sign of two overlapping pulses (respectively on TxINB and TxINA) generates a one step up/down count, so that the up/down control and the counter clock are both external. The setting of the UDC bit in the TCR register has no effect in this configuration.

Figure 85. Parallel Mode Description



Note: MFT 1 is not available on all devices. Refer to the device block diagram and register map.

MULTIFUNCTION TIMER (Cont'd)

10.4.3 Input Pin Assignment

The two external inputs (TxINA and TxINB) of the timer can be individually configured to catch a particular external event (i.e. rising edge, falling edge, or both rising and falling edges) by programming the two relevant bits (A0, A1 and B0, B1) for each input in the external Input Control Register (T_ICR).

The 16 different functional modes of the two external inputs can be selected by programming bits IN0 - IN3 of the T_ICR, as illustrated in Figure 2

Table 32. Input Pin Function

I C Reg. IN3-IN0 bits	TxINA Input Function	TxINB Input Function
0000	not used	not used
0001	not used	Trigger
0010	Gate	not used
0011	Gate	Trigger
0100	not used	Ext. Clock
0101	Trigger	not used
0110	Gate	Ext. Clock
0111	Trigger	Trigger
1000	Clock Up	Clock Down
1001	Up/Down	Ext. Clock
1010	Trigger Up	Trigger Down
1011	Up/Down	not used
1100	Autodiscr.	Autodiscr.
1101	Trigger	Ext. Clock
1110	Ext. Clock	Trigger
1111	Trigger	Gate

Some choices relating to the external input pin assignment are defined in conjunction with the RM0 and RM1 bits in TMR.

For input pin assignment codes which use the input pins as Trigger Inputs (except for code 1010, Trigger Up:Trigger Down), the following conditions apply:

– a trigger signal on the TxINA input pin performs an U/D counter load if RM0 is reset, or an external capture if RM0 is set.

– a trigger signal on the TxINB input pin always performs an external capture on REG1R. The TxINB input pin is disabled when the Bivalue Mode is set.

Note: For proper operation of the External Input pins, the following must be observed:

– the minimum external clock/trigger pulse width must not be less than the system clock (INTCLK) period if the input pin is programmed as rising or falling edge sensitive.

– the minimum external clock/trigger pulse width must not be less than the prescaler clock period (INTCLK/3) if the input pin is programmed as rising and falling edge sensitive (valid also in Auto discrimination mode).

– the minimum delay between two clock/trigger pulse active edges must be greater than the prescaler clock period (INTCLK/3), while the minimum delay between two consecutive clock/trigger pulses must be greater than the system clock (INTCLK) period.

– the minimum gate pulse width must be at least twice the prescaler clock period (INTCLK/3).

– in Autodiscrimination mode, the minimum delay between the input pin A pulse edge and the edge of the input pin B pulse, must be at least equal to the system clock (INTCLK) period.

– if a number, N, of external pulses must be counted using a Compare Register in External Clock mode, then the Compare Register must be loaded with the value [X +/- (N-1)], where X is the starting counter value and the sign is chosen depending on whether Up or Down count mode is selected.

MULTIFUNCTION TIMER (Cont'd)

10.4.3.1 TxINA = I/O - TxINB = I/O

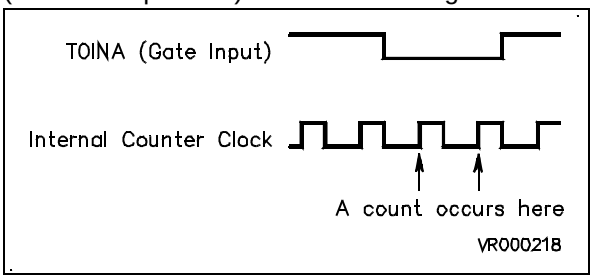
Input pins A and B are not used by the Timer. The counter clock is internally generated and the up/down selection may be made only by software via the UDC (Software Up/Down) bit in the TCR register.

10.4.3.2 TxINA = I/O - TxINB = Trigger

The signal applied to input pin B acts as a trigger signal on REG1R register. The prescaler clock is internally generated and the up/down selection may be made only by software via the UDC (Software Up/Down) bit in the TCR register.

10.4.3.3 TxINA = Gate - TxINB = I/O

The signal applied to input pin A acts as a gate signal for the internal clock (i.e. the counter runs only when the gate signal is at a low level). The counter clock is internally generated and the up/down control may be made only by software via the UDC (Software Up/Down) bit in the TCR register.



10.4.3.4 TxINA = Gate - TxINB = Trigger

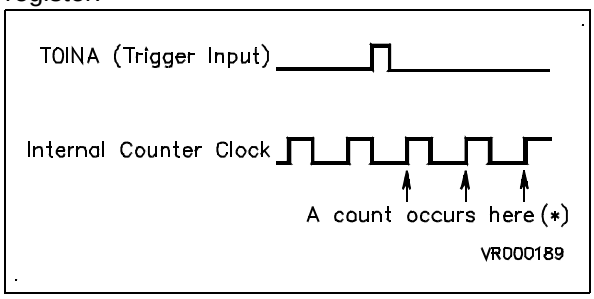
Both input pins A and B are connected to the timer, with the resulting effect of combining the actions relating to the previously described configurations.

10.4.3.5 TxINA = I/O - TxINB = Ext. Clock

The signal applied to input pin B is used as the external clock for the prescaler. The up/down selection may be made only by software via the UDC (Software Up/Down) bit in the TCR register.

10.4.3.6 TxINA = Trigger - TxINB = I/O

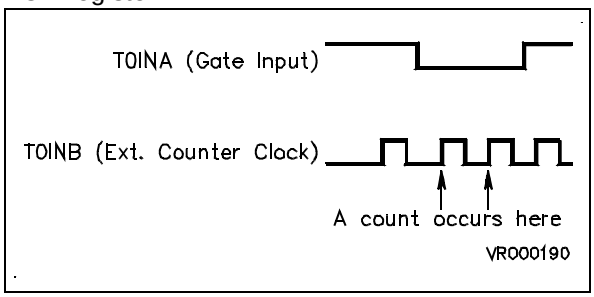
The signal applied to input pin A acts as a trigger for REG0R, initiating the action for which the register was programmed (i.e. a reload or capture). The prescaler clock is internally generated and the up/down selection may be made only by software via the UDC (Software Up/Down) bit in the TCR register.



(*) The timer is in One shot mode and REGOR in Reload mode

10.4.3.7 TxINA = Gate - TxINB = Ext. Clock

The signal applied to input pin B, gated by the signal applied to input pin A, acts as external clock for the prescaler. The up/down control may be made only by software action through the UDC bit in the TCR register.



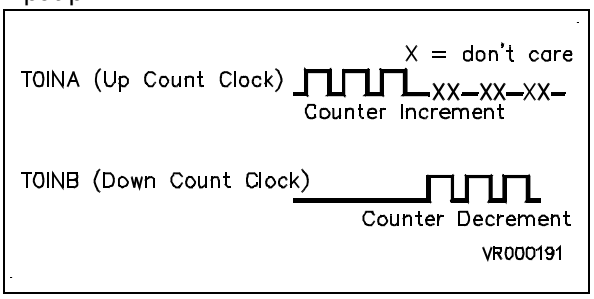
10.4.3.8 TxINA = Trigger - TxINB = Trigger

The signal applied to input pin A (or B) acts as trigger signal for REG0R (or REG1R), initiating the action for which the register has been programmed. The counter clock is internally generated and the up/down selection may be made only by software via the UDC (Software Up/Down) bit in the TCR register.

MULTIFUNCTION TIMER (Cont'd)

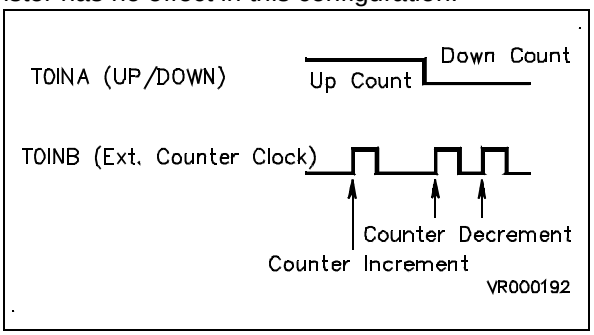
10.4.3.9 TxINA = Clock Up - TxINB = Clock Down

The edge received on input pin A (or B) performs a one step up (or down) count, so that the counter clock and the up/down control are external. Setting the UDC bit in the TCR register has no effect in this configuration, and input pin B has priority on input pin A.



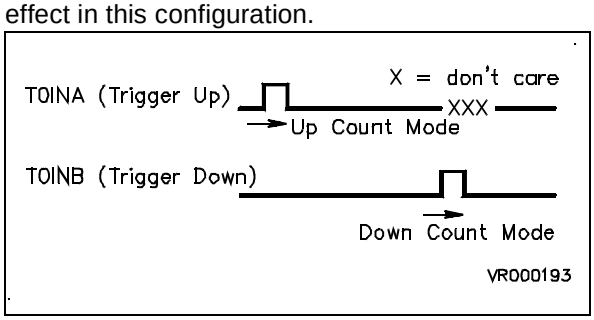
10.4.3.10 TxINA = Up/Down - TxINB = Ext Clock

An High (or Low) level applied to input pin A sets the counter in the up (or down) count mode, while the signal applied to input pin B is used as clock for the prescaler. Setting the UDC bit in the TCR register has no effect in this configuration.



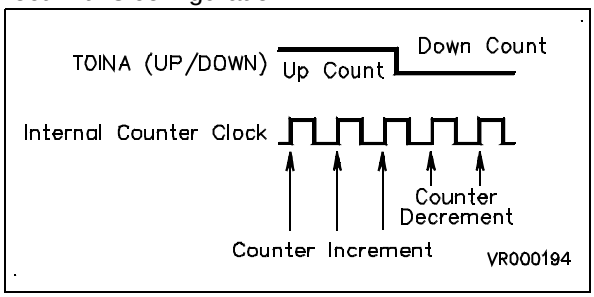
10.4.3.11 TxINA = Trigger Up - TxINB = Trigger Down

Up/down control is performed through both input pins A and B. A edge on input pin A sets the up count mode, while a edge on input pin B (which has priority on input pin A) sets the down count mode. The counter clock is internally generated, and setting the UDC bit in the TCR register has no effect in this configuration.



10.4.3.12 TxINA = Up/Down - TxINB = I/O

An High (or Low) level of the signal applied on input pin A sets the counter in the up (or down) count mode. The counter clock is internally generated. Setting the UDC bit in the TCR register has no effect in this configuration.

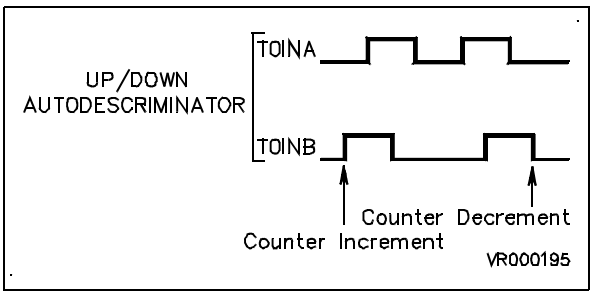


MULTIFUNCTION TIMER (Cont'd)

10.4.3.13 Autodiscrimination Mode

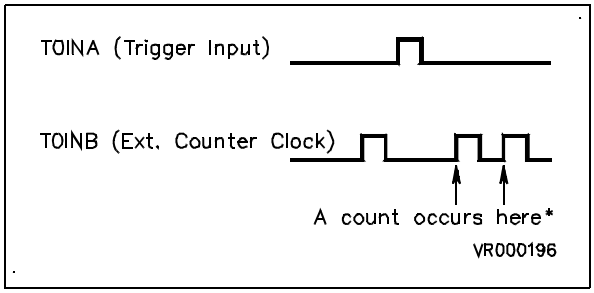
The phase between two pulses (respectively on input pin B and input pin A) generates a one step up (or down) count, so that the up/down control and the counter clock are both external. Thus, if the rising edge of TxINB arrives when TxINA is at a low level, the timer is incremented (no action if the rising edge of TxINB arrives when TxINA is at a high level). If the falling edge of TxINB arrives when TxINA is at a low level, the timer is decremented (no action if the falling edge of TxINB arrives when TxINA is at a high level).

Setting the UDC bit in the TCR register has no effect in this configuration.



10.4.3.14 TxINA = Trigger - TxINB = Ext. Clock

The signal applied to input pin A acts as a trigger signal on REG0R, initiating the action for which the register was programmed (i.e. a reload or capture), while the signal applied to input pin B is used as the clock for the prescaler.



(*) The timer is in One shot mode and REG0R in reload mode

10.4.3.15 TxINA = Ext. Clock - TxINB = Trigger

The signal applied to input pin B acts as a trigger, performing a capture on REG1R, while the signal applied to input pin A is used as the clock for the prescaler.

10.4.3.16 TxINA = Trigger - TxINB = Gate

The signal applied to input pin A acts as a trigger signal on REG0R, initiating the action for which the register was programmed (i.e. a reload or capture), while the signal applied to input pin B acts as a gate signal for the internal clock (i.e. the counter runs only when the gate signal is at a low level).

MULTIFUNCTION TIMER (Cont'd)

10.4.4 Output Pin Assignment

Two external outputs are available when programmed as Alternate Function Outputs of the I/O pins.

Two registers Output A Control Register (OACR) and Output B Control Register (OBCR) define the driver for the outputs and the actions to be performed.

Each of the two output pins can be driven from any of the three possible sources:

– Compare Register 0 event logic

– Compare Register 1 event logic

– Overflow/Underflow event logic.

Each of these three sources can cause one of the following four actions on any of the two outputs:

– Nop

– Set

– Reset

– Toggle

Furthermore an On Chip Event signal can be driven by two of the three sources: the Over/Underflow event and Compare 0 event by programming the CEV bit of the OACR register and the OEV bit of OBCR register respectively. This signal can be used internally to synchronise another on-chip peripheral.

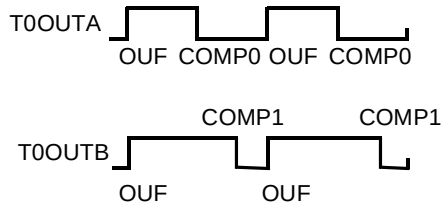
Output Waveforms

Depending on the programming of OACR and OBCR, the following example waveforms can be generated on TxOUTA and TxOUTB pins.

For a configuration where TxOUTA is driven by the Over/Underflow (OUF) and the Compare 0 event (CM0), and TxOUTB is driven by the Over/Underflow and Compare 1 event (CM1):

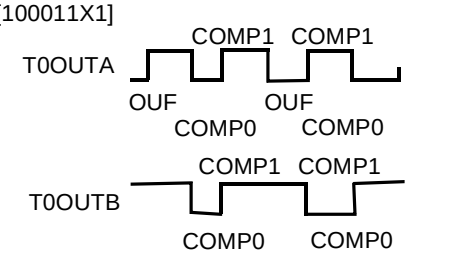
OACR is programmed with TxOUTA preset to "0", OUF sets TxOUTA, CM0 resets TxOUTA and CM1 does not affect the output.
OBCR is programmed with TxOUTB preset to "0", OUF sets TxOUTB, CM1 resets TxOUTB while CM0 does not affect the output.

OACR = [101100X0]
OBCR = [111000X0]



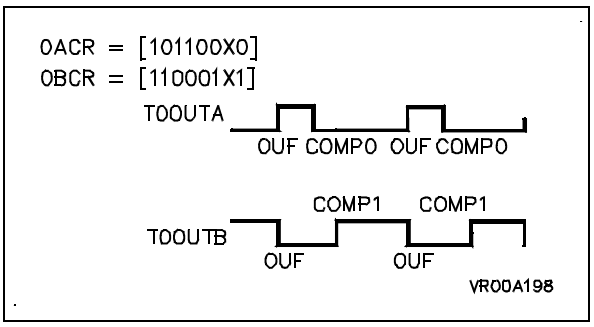
For a configuration where TxOUTA is driven by the Over/Underflow, by Compare 0 and by Compare 1; TxOUTB is driven by both Compare 0 and Compare 1. OACR is programmed with TxOUTA preset to "0". OUF toggles Output 0, as do CM0 and CM1. OBCR is programmed with TxOUTB preset to "1". OUF does not affect the output; CM0 resets TxOUTB and CM1 sets it.

OACR = [010101X0]
OBCR = [100011X1]

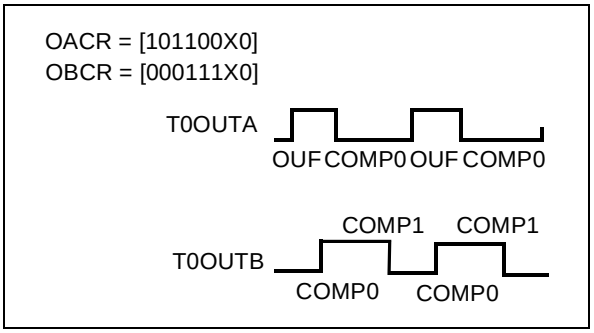


MULTIFUNCTION TIMER (Cont'd)

For a configuration where TxOUTA is driven by the Over/Underflow and by Compare 0, and TxOUTB is driven by the Over/Underflow and by Compare 1. OACR is programmed with TxOUTA preset to "0". OUF sets TxOUTA while CM0 resets it, and CM1 has no effect. OBCR is programmed with TxOUTB preset to "1". OUF toggles TxOUTB, CM1 sets it and CM0 has no effect.



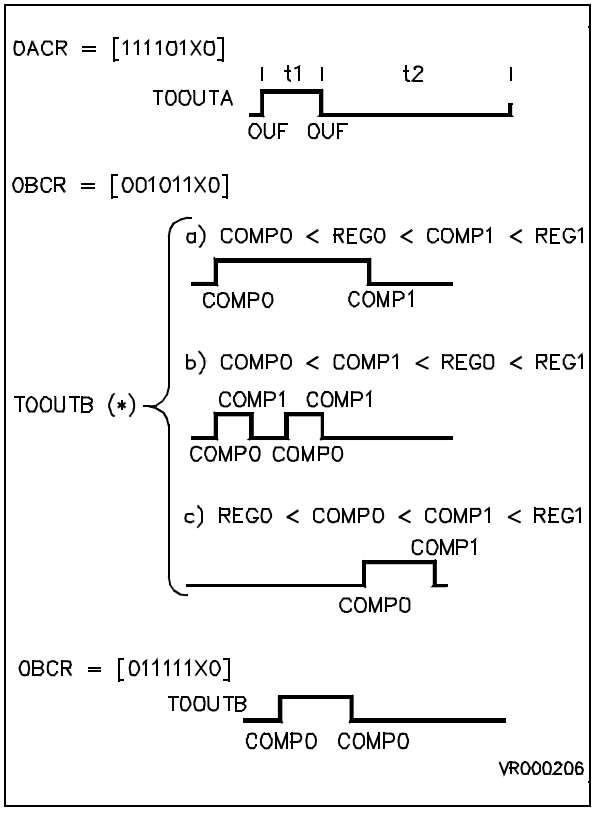
For a configuration where TxOUTA is driven by the Over/Underflow and by Compare 0, and TxOUTB is driven by Compare 0 and 1. OACR is programmed with TxOUTA preset to "0". OUF sets TxOUTA, CM0 resets it and CM1 has no effect. OBCR is programmed with TxOUTB preset to "0". OUF has no effect, CM0 sets TxOUTB and CM1 toggles it.



Output Waveform Samples In Biload Mode

TxOUTA is programmed to monitor the two time intervals, t1 and t2, of the Biload Mode, while TxOUTB is independent of the Over/Underflow and is driven by the different values of Compare 0 and Compare 1. OACR is programmed with TxOUTA preset to "0". OUF toggles the output and CM0 and CM1 do not affect TxOUTA. OBCR is programmed with TxOUTB preset to "0". OUF has no effect, while CM1 resets TxOUTB and CM0 sets it.

Depending on the CM1/CM0 values, three different sample waveforms have been drawn based on the above mentioned configuration of OBCR. In the last case, with a different programmed value of OBCR, only Compare 0 drives TxOUTB, toggling the output.



Note (*) Depending on the CMP1R/CMP0R values

MULTIFUNCTION TIMER (Cont'd)

10.4.5 Interrupt and DMA

10.4.5.1 Timer Interrupt

The timer has 5 different Interrupt sources, belonging to 3 independent groups, which are assigned to the following Interrupt vectors:

Table 33. Timer Interrupt Structure

Interrupt Source	Vector Address
COMP 0 COMP 1	xxxx x110
CAPT 0 CAPT 1	xxxx x100
Overflow/Underflow	xxxx x000

The three least significant bits of the vector pointer address represent the relative priority assigned to each group, where 000 represents the highest priority level. These relative priorities are fixed by hardware, according to the source which generates the interrupt request. The 5 most significant bits represent the general priority and are programmed by the user in the Interrupt Vector Register (T_IVR).

Each source can be masked by a dedicated bit in the Interrupt/DMA Mask Register (IDMR) of each timer, as well as by a global mask enable bit (IDMR.7) which masks all interrupts.

If an interrupt request (CM0 or CP0) is present before the corresponding pending bit is reset, an overrun condition occurs. This condition is flagged in two dedicated overrun bits, relating to the Comp0 and Capt0 sources, in the Timer Flag Register (T_FLAGR).

10.4.5.2 Timer DMA

Two Independent DMA channels, associated with Comp0 and Capt0 respectively, allow DMA transfers from Register File or Memory to the Comp0 Register, and from the Capt0 Register to Register File or Memory). If DMA is enabled, the Capt0 and Comp0 interrupts are generated by the corresponding DMA End of Block event. Their priority is set by hardware as follows:

– Compare 0 Destination — Lower Priority

– Capture 0 Source — Higher Priority

The two DMA request sources are independently maskable by the CP0D and CM0D DMA Mask bits in the IDMR register.

The two DMA End of Block interrupts are independently enabled by the CP0I and CM0I Interrupt mask bits in the IDMR register.

10.4.5.3 DMA Pointers

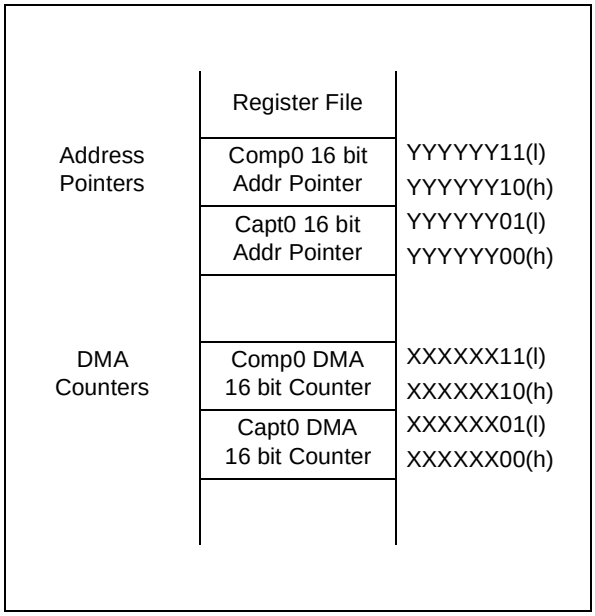
The 6 programmable most significant bits of the DMA Counter Pointer Register (DCPR) and of the DMA Address Pointer Register (DAPR) are common to both channels (Comp0 and Capt0). The Comp0 and Capt0 Address Pointers are mapped as a pair in the Register File, as are the Comp0 and Capt0 DMA Counter pair.

In order to specify either the Capt0 or the Comp0 pointers, according to the channel being serviced, the Timer resets address bit 1 for CAPT0 and sets it for COMP0, when the D0 bit in the DCPR register is equal to zero (Word address in Register File). In this case (transfers between peripheral registers and memory), the pointers are split into two groups of adjacent Address and Counter pairs respectively.

For peripheral register to register transfers (selected by programming "1" into bit 0 of the DCPR register), only one pair of pointers is required, and the pointers are mapped into one group of adjacent positions.

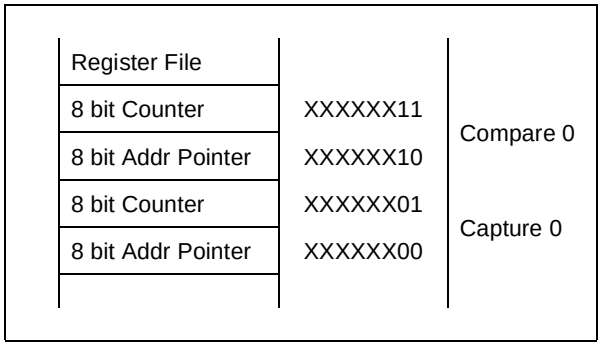
The DMA Address Pointer Register (DAPR) is not used in this case, but must be considered reserved.

Figure 86. Pointer Mapping for Transfers between Registers and Memory



MULTIFUNCTION TIMER (Cont'd)

Figure 87. Pointer Mapping for Register to Register Transfers



10.4.5.4 DMA Transaction Priorities

Each Timer DMA transaction is a 16-bit operation, therefore two bytes must be transferred sequentially, by means of two DMA transfers. In order to speed up each word transfer, the second byte transfer is executed by automatically forcing the peripheral priority to the highest level (000), regardless of the previously set level. It is then restored to its original value after executing the transfer. Thus, once a request is being serviced, its hardware priority is kept at the highest level regardless of the other Timer internal sources, i.e. once a Comp0 request is being serviced, it maintains a higher priority, even if a Capt0 request occurs between the two byte transfers.

10.4.5.5 DMA Swap Mode

After a complete data table transfer, the transaction counter is reset and an End Of Block (EOB) condition occurs, the block transfer is completed.

The End Of Block Interrupt routine must at this point reload both address and counter pointers of the channel referred to by the End Of Block interrupt source, if the application requires a continuous high speed data flow. This procedure causes speed limitations because of the time required for the reload routine.

The SWAP feature overcomes this drawback, allowing high speed continuous transfers. Bit 2 of the DMA Counter Pointer Register (DCPR) and of the DMA Address Pointer Register (DAPR), toggles after every End Of Block condition, alternately providing odd and even address (D2-D7) for the pair of pointers, thus pointing to an updated pair, after a block has been completely transferred. This allows the User to update or read the first block and to update the pointer values while the second is being transferred. These two toggle bits are software writable and readable, mapped in DCPR bit 2 for the CM0 channel, and in DAPR bit 2 for the CP0 channel (though a DMA event on a channel, in Swap mode, modifies a field in DAPR and DCPR common to both channels, the DAPR/DCPR content used in the transfer is always the bit related to the correct channel).

SWAP mode can be enabled by the SWEN bit in the IDCR Register.

WARNING: *Enabling SWAP mode affects both channels (CM0 and CP0).*

MULTIFUNCTION TIMER (Cont'd)

10.4.5.6 DMA End Of Block Interrupt Routine

An interrupt request is generated after each block transfer (EOB) and its priority is the same as that assigned in the usual interrupt request, for the two channels. As a consequence, they will be serviced only when no DMA request occurs, and will be subject to a possible OUF Interrupt request, which has higher priority.

The following is a typical EOB procedure (with swap mode enabled):

– Test Toggle bit and Jump.

– Reload Pointers (odd or even depending on toggle bit status).

– Reset EOB bit: this bit must be reset only after the old pair of pointers has been restored, so that, if a new EOB condition occurs, the next pair of pointers is ready for swapping.

– Verify the software protection condition (see Section 0.1.5.7).

– Read the corresponding Overrun bit: this confirms that no DMA request has been lost in the meantime.

– Reset the corresponding pending bit.

– Reenable DMA with the corresponding DMA mask bit **(must always be done after resetting the pending bit)**

– Return.

WARNING: The EOB bits are read/write **only** for test purposes. Writing a logical "1" by software (when the SWEN bit is set) will cause a spurious interrupt request. These bits are normally only reset by software.

10.4.5.7 DMA Software Protection

A second EOB condition may occur before the first EOB routine is completed, this would cause a not yet updated pointer pair to be addressed, with consequent overwriting of memory. To prevent these errors, a protection mechanism is provided, such that the attempted setting of the EOB bit before it has been reset by software will cause the DMA mask on that channel to be reset (DMA disabled), thus blocking any further DMA operation. As shown above, this mask bit should always be checked in each EOB routine, to ensure that all DMA transfers are properly served.

10.4.6 Register Description

Note: In the register description on the following pages, register and page numbers are given using the example of Timer 0. On devices with more than one timer, refer to the device register map for the adresses and page numbers.

MULTIFUNCTION TIMER (Cont'd)

CAPTURE LOAD 0 HIGH REGISTER (REG0HR)

R240 - Read/Write
Register Page: 10
Reset value: undefined

7							0
R15	R14	R13	R12	R11	R10	R9	R8

This register is used to capture values from the Up/Down counter or load preset values (MSB).

CAPTURE LOAD 0 LOW REGISTER (REG0LR)

R241 - Read/Write
Register Page: 10
Reset value: undefined

7							0
R7	R6	R5	R4	R3	R2	R1	R0

This register is used to capture values from the Up/Down counter or load preset values (LSB).

CAPTURE LOAD 1 HIGH REGISTER (REG1HR)

R242 - Read/Write
Register Page: 10
Reset value: undefined

7							0
R15	R14	R13	R12	R11	R10	R9	R8

This register is used to capture values from the Up/Down counter or load preset values (MSB).

CAPTURE LOAD 1 LOW REGISTER (REG1LR)

R243 - Read/Write
Register Page: 10
Reset value: undefined

7							0
R7	R6	R5	R4	R3	R2	R1	R0

This register is used to capture values from the Up/Down counter or load preset values (LSB).

COMPARE 0 HIGH REGISTER (CMP0HR)

R244 - Read/Write
Register Page: 10
Reset value: 0000 0000 (00h)

7							0
R15	R14	R13	R12	R11	R10	R9	R8

This register is used to store the MSB of the 16-bit value to be compared to the Up/Down counter content.

COMPARE 0 LOW REGISTER (CMP0LR)

R245 - Read/Write
Register Page: 10
Reset value: 0000 0000 (00h)

7							0
R7	R6	R5	R4	R3	R2	R1	R0

This register is used to store the LSB of the 16-bit value to be compared to the Up/Down counter content.

COMPARE 1 HIGH REGISTER (CMP1HR)

R246 - Read/Write
Register Page: 10
Reset value: 0000 0000 (00h)

7							0
R15	R14	R13	R12	R11	R10	R9	R8

This register is used to store the MSB of the 16-bit value to be compared to the Up/Down counter content.

COMPARE 1 LOW REGISTER (CMP1LR)

R247 - Read/Write
Register Page: 10
Reset value: 0000 0000 (00h)

7							0
R7	R6	R5	R4	R3	R2	R1	R0

This register is used to store the LSB of the 16-bit value to be compared to the Up/Down counter content.

MULTIFUNCTION TIMER (Cont'd)

TIMER CONTROL REGISTER (TCR)
R248 - Read/Write
Register Page: 10
Reset value: 0000 0000 (00h)

7							0
CEN	CCP0	CCMP0	CCL	UDC	UDCS	OF0	CS

Bit 7 = **CEN**: *Counter enable*.
This bit is ANDed with the Global Counter Enable bit (GCEN) in the CICR register (R230). The GCEN bit is set after the Reset cycle.
0: Stop the counter and prescaler
1: Start the counter and prescaler (without reload).

Note: Even if CEN=0, capture and loading will take place on a trigger event.

Bit 6 = **CCP0**: *Clear on capture*.
0: No effect
1: Clear the counter and reload the prescaler on a REG0R or REG1R capture event

Bit 5 = **CCMP0**: *Clear on Compare*.
0: No effect
1: Clear the counter and reload the prescaler on a CMP0R compare event

Bit 4 = **CCL**: *Counter clear*.

This bit is reset by hardware after being set by software (this bit always returns "0" when read).
0: No effect
1: Clear the counter without generating an interrupt request

Bit 3 = **UDC**: *Up/Down software selection*.
If the direction of the counter is not fixed by hardware (TxINA and/or TxINB pins, see par. 10.3) it can be controlled by software using the UDC bit.
0: Down counting
1: Up counting

Bit 2 = **UDCS**: *Up/Down count status*.
This bit is read only and indicates the direction of the counter.
0: Down counting
1: Up counting

Bit 1 = **OF0**: *OVF/UNF state*.
This bit is read only.
0: No overflow or underflow occurred
1: Overflow or underflow occurred during a Capture on Register 0

Bit 0 = **CS** *Counter Status*.
This bit is read only and indicates the status of the counter.
0: Counter halted
1: Counter running

MULTIFUNCTION TIMER (Cont'd)

TIMER MODE REGISTER (TMR)
R249 - Read/Write
Register Page: 10
Reset value: 0000 0000 (00h)

7							0
OE1	OE0	BM	RM1	RM0	ECK	REN	C0

Bit 7 = **OE1**: *Output 1 enable.*
0: Disable the Output 1 (TxOUTB pin) and force it high.
1: Enable the Output 1 (TxOUTB pin)
The relevant I/O bit must also be set to Alternate Function

Bit 6 = **OE0**: *Output 0 enable.*
0: Disable the Output 0 (TxOUTA pin) and force it high
1: Enable the Output 0 (TxOUTA pin).
The relevant I/O bit must also be set to Alternate Function

Bit 5 = **BM**: *Bivalue mode*.
This bit works together with the RM1 and RM0 bits to select the timer operating mode (see Table 4).
0: Disable bivalue mode
1: Enable bivalue mode

Bit 4 = **RM1**: *REG1R mode*.
This bit works together with the BM and RM0 bits to select the timer operating mode. Refer to Table 4.

Note: This bit has no effect when the Bivalue Mode is enabled (BM=1).

Bit 3 = **RM0**: *REG0R mode*.
This bit works together with the BM and RM1 bits to select the timer operating mode. Refer to Table 4.

Table 34. Timer Operating Modes

TMR Bits			Timer Operating Modes
BM	**RM1**	**RM0**	
1	x	0	Biload mode
1	**x**	1	Bicapture mode
0	0	0	Load from REG0R and Monitor on REG1R
0	1	0	Load from REG0R and Capture on REG1R
0	0	1	Capture on REG0R and Monitor on REG1R
0	1	1	Capture on REG0R and REG1R

Bit 2 = **ECK** *Timer clock control*.
0: The prescaler clock source is selected depending on the IN0 - IN3 bits in the T_ICR register
1: Enter Parallel mode (for Timer 1 and Timer 3 only, no effect for Timer 0 and 2). See Section 0.1.2.12.

Bit 1 = **REN**: *Retrigger mode*.
0: Enable retriggerable mode
1: Disable retriggerable mode

Bit 0 = **CO**: *Continous/One shot mode*.
0: Continuous mode (with autoreload on End of Count condition)
1: One shot mode

MULTIFUNCTION TIMER (Cont'd)

EXTERNAL INPUT CONTROL REGISTER (T_ICR)
R250 - Read/Write
Register Page: 10
Reset value: 0000 0000 (00h)

7							0
IN3	IN2	IN1	IN0	A0	A1	B0	B1

Bits 7:4 = **IN[3:0]**: *Input pin function.*
These bits are set and cleared by software.

IN[3:0] bits	TxINA Pin Function	TxINB Input Pin Function
0000	not used	not used
0001	not used	Trigger
0010	Gate	not used
0011	Gate	Trigger
0100	not used	Ext. Clock
0101	Trigger	not used
0110	Gate	Ext. Clock
0111	Trigger	Trigger
1000	Clock Up	Clock Down
1001	Up/Down	Ext. Clock
1010	Trigger Up	Trigger Down
1011	Up/Down	not used
1100	Autodiscr.	Autodiscr.
1101	Trigger	Ext. Clock
1110	Ext. Clock	Trigger
1111	Trigger	Gate

Bits 3:2 = **A[0:1]**: *TxINA Pin event.*

These bits are set and cleared by software.

A0	A1	TxINA Pin Event
0	0	No operation
0	1	Falling edge sensitive
1	0	Rising edge sensitive
1	1	Rising and falling edges

Bits 1:0 = **B[0:1]:** *TxINB Pin event.*
These bits are set and cleared by software.

B0	B1	TxINB Pin Event
0	0	No operation
0	1	Falling edge sensitive
1	0	Rising edge sensitive
1	1	Rising and falling edges

PRESCALER REGISTER (PRSR)
R251 - Read/Write
Register Page: 10
Reset value: 0000 0000 (00h)

7							0
P7	P6	P5	P4	P3	P2	P1	P0

This register holds the preset value for the 8-bit prescaler. The PRSR content may be modified at any time, but it will be loaded into the prescaler at the following prescaler underflow, or as a consequence of a counter reload (either by software or upon external request).

Following a RESET condition, the prescaler is automatically loaded with 00h, so that the prescaler divides by 1 and the maximum counter clock is generated (Crystal oscillator clock frequency divided by 6 when MODER.5 = DIV2 bit is set).

The binary value programmed in the PRSR register is equal to the divider value minus one. For example, loading PRSR with 24 causes the prescaler to divide by 25.

MULTIFUNCTION TIMER (Cont'd)

OUTPUT A CONTROL REGISTER (OACR)
R252 - Read/Write
Register Page: 10
Reset value: 0000 0000

7							0
C0E0	C0E1	C1E0	C1E1	OUE0	OUE1	CEV	0P

Bits 7:6 = **C0E[0:1]**: *COMP0 action bits*.
These bits are set and cleared by software. They configure the action to be performed on the TxOUTA pin when a successful compare of the CMP0R register occurs. Refer to Table 5 for the list of actions that can be configured.

Bits 5:4 = **C1E[0:1]:** *COMP1 action bits*.
These bits are set and cleared by software. They configure the action to be performed on the TxOUTA pin when a successful compare of the CMP1R register occurs. Refer to Table 5 for the list of actions that can be configured.

Bits 3:2 = **OUE[0:1]**: *OVF/UNF action bits*.
These bits are set and cleared by software. They configure the action to be performed on the TxOUTA pin when an Overflow or Underflow of the U/D counter occurs. Refer to Table 5 for the list of actions that can be configured.

Table 35. Output A Action Bits

xxE0	xxE1	Action on TxOUTA pin when an xx event occurs
0	0	Set
0	1	Toggle
1	0	Reset
1	1	NOP

Notes:

– xx stands for C0, C1 or OU.

– Whenever more than one event occurs simultaneously, Action bit 0 will be the result of ANDing Action bit 0 of all simultaneous events and Action bit 1 will be the result of ANDing Action bit 1 of all simultaneous events.

Bit 1 = **CEV**: *On-Chip event on CMP0R*.
This bit is set and cleared by software.
0: No action
1: A successful compare on CMP0R activates the on-chip event signal (a single pulse is generated)

Bit 0 = **OP**: *TxOUTA preset value*.
This bit is set and cleared by software and by hardware. The value of this bit is the preset value of the TxOUTA pin. Reading this bit returns the current state of the TxOUTA pin (useful when it is selected in toggle mode).

MULTIFUNCTION TIMER (Cont'd)

OUTPUT B CONTROL REGISTER (OBCR)
R253 - Read/Write
Register Page: 10
Reset value: 0000 0000 (00h)

7							0
C0E0	C0E1	C1E0	C1E1	OUE0	OUE1	OEV	0P

Bits 7:6 = **C0E[0:1]**: *COMP0 Action Bits*.
These bits are set and cleared by software. They configure the type of action to be performed on the TxOUTB output pin when successful compare of the CMP0R register occurs. Refer to Table 6 for the list of actions that can be configured.

Bits 5:4 = **C0E[0:1]:** *COMP1 Action Bits*.
These bits are set and cleared by software. They configure the type of action to be performed on the TxOUTB output pin when a successful compare of the CMP1R register occurs. Refer to Table 6 for the list of actions that can be configured.

Bits 3:2 = **OUE[0:1]**: *OVF/UNF Action Bits*.
These bits are set and cleared by software.They configure the type of action to be performed on the TxOUTB output pin when an Overflow or Underflow on the U/D counter occurs. Refer to Table 6 for the list of actions that can be configured.

Table 36. Output B Action Bits

xxE0	xxE1	Action on the TxOUTB pin when an xx event occurs
0	0	Set
0	1	Toggle
1	0	Reset
1	1	NOP

Notes:

– xx stands for C0, C1 or OU.

– Whenever more than one event occurs simultaneously, Action Bit 0 will be the result of ANDing Action Bit 0 of all simultaneous events and Action Bit 1 will be the result of ANDing Action Bit 1 of all simultaneous events.

Bit 1 = **OEV**: *On-Chip event on OVF/UNF*.
This bit is set and cleared by software.
0: No action
1: An underflow/overflow activates the on-chip event signal (a single pulse is generated)

Bit 0 = **OP**: *TxOUTB preset value*.
This bit is set and cleared by software and by hardware. The value of this bit is the preset value of the TxOUTB pin. Reading this bit returns the current state of the TxOUTB pin (useful when it is selected in toggle mode).

MULTIFUNCTION TIMER (Cont'd)

FLAG REGISTER (T_FLAGR)
R254 - Read/Write
Register Page: 10
Reset value: 0000 0000 (00h)

7							0
CP0	CP1	CM0	CM1	OUF	OCP0	OCM0	A0

Bit 7 = **CP0**: *Capture 0 flag.*
This bit is set by hardware after a capture on REG0R register. An interrupt is generated depending on the value of the GTIEN, CP0I bits in the IDMR register and the A0 bit in the T_FLAGR register. The CP0 bit must be cleared by software. Setting by software acts as a software load/capture to/from the REG0R register.
0: No Capture 0 event
1: Capture 0 event occurred

Bit 6 = **CP1**: *Capture 1 flag*.
This bit is set by hardware after a capture on REG1R register. An interrupt is generated depending on the value of the GTIEN, CP0I bits in the IDMR register and the A0 bit in the T_FLAGR register. The CP1 bit must be cleared by software. Setting by software acts as a capture event on the REG1R register, except when in Bicapture mode.
0: No Capture 1 event
1: Capture 1 event occurred

Bit 5 = **CM0**: *Compare 0 flag*.
This bit is set by hardware after a successful compare on the CMP0R register. An interrupt is generated if the GTIEN and CM0I bits in the IDMR register are set. The CM0 bit is cleared by software.
0: No Compare 0 event
1: Compare 0 event occurred

Bit 4 = **CM1**: *Compare 1 flag.*
This bit is set after a successful compare on CMP1R register. An interrupt is generated if the GTIEN and CM1I bits in the IDMR register are set. The CM1 bit is cleared by software.
0: No Compare 1 event
1: Compare 1 event occurred

Bit 3 = **OUF**: *Overflow/Underflow*.
This bit is set by hardware after a counter Over/Underflow condition. An interrupt is generated if GTIEN and OUI=1 in the IDMR register. The OUF bit is cleared by software.
0: No counter overflow/underflow
1: Counter overflow/underflow

Bit 2 = **OCP0**: *Overrun on Capture 0.*
This bit is set by hardware when more than one INT/DMA requests occur before the CP0 flag is cleared by software or whenever a capture is simulated by setting the CP0 flag by software. The OCP0 flag is cleared by software.
0: No capture 0 overrun
1: Capture 0 overrun

Bit 1 = **OCM0**: *Overrun on compare 0.*
This bit is set by hardware when more than one INT/DMA requests occur before the CM0 flag is cleared by software.The OCM0 flag is cleared by software.
0: No compare 0 overrun
1: Compare 0 overrun

Bit 0 = **A0**: *Capture interrupt function*.
This bit is set and cleared by software.
0: Configure the capture interrupt as an OR function of REG0R/REG1R captures
1: Configure the capture interrupt as an AND function of REG0R/REG1R captures

Note: When A0 is set, both CP0I and CP1I in the IDMR register must be set to enable both capture interrupts.

MULTIFUNCTION TIMER (Cont'd)

INTERRUPT/DMA MASK REGISTER (IDMR)
R255 - Read/Write
Register Page: 10
Reset value: 0000 0000 (00h)

7							0
GTIEN	CP0D	CP0I	CP1I	CM0D	CM0I	CM1I	OUI

Bit 7 = **GTIEN**: *Global timer interrupt enable*.
This bit is set and cleared by software.
0: Disable all Timer interrupts
1: Enable all timer Timer Interrupts from enabled sources

Bit 6 = **CP0D**: *Capture 0 DMA mask.*
This bit is set by software to enable a Capt0 DMA transfer and cleared by hardware at the end of the block transfer.
0: Disable capture on REG0R DMA
1: Enable capture on REG0R DMA

Bit 5 = **CP0I**: *Capture 0 interrupt mask*.
0: Disable capture on REG0R interrupt
1: Enable capture on REG0R interrupt (or Capt0 DMA End of Block interrupt if CP0D=1)

Bit 4 = **CP1I**: *Capture 1 interrupt mask*.
This bit is set and cleared by software.
0: Disable capture on REG1R interrupt
1: Enable capture on REG1R interrupt

Bit 3 = **CM0D**: *Compare 0 DMA mask*.
This bit is set by software to enable a Comp0 DMA transfer and cleared by hardware at the end of the block transfer.
0: Disable compare on CMP0R DMA
1: Enable compare on CMP0R DMA

Bit 2 = **CM0I**: *Compare 0 Interrupt mask*.
This bit is set and cleared by software.
0: Disable compare on CMP0R interrupt
1: Enable compare on CMP0R interrupt (or Comp0 DMA End of Block interrupt if CM0D=1)

Bit 1 = **CM1I**: *Compare 1 Interrupt mask*.
This bit is set and cleared by software.
0: Disable compare on CMP1R interrupt
1: Enable compare on CMP1R interrupt

Bit 0 = **OUI**:
Overflow/Underflow interrupt mask.
This bit is set and cleared by software.
0: Disable Overflow/Underflow interrupt
1: Enable Overflow/Underflow interrupt

DMA COUNTER POINTER REGISTER (DCPR)
R240 - Read/Write
Register Page: 9
Reset value: undefined

7							0
DCP7	DCP6	DCP5	DCP4	DCP3	DCP2	DMA SRCE	REG/ MEM

Bits 7:2 = **DCP[7:2]**: *MSBs of DMA counter register address.*
These are the most significant bits of the DMA counter register address programmable by software. The DCP2 bit may also be toggled by hardware if the Timer DMA section for the Compare 0 channel is configured in Swap mode.

Bit 1 = **DMA-SRCE**: *DMA source selection.*
This bit is set and cleared by hardware.
0: DMA source is a Capture on REG0R register
1: DMA destination is a Compare on CMP0R register

Bit 0 = **REG/MEM**: *DMA area selection*.

This bit is set and cleared by software. It selects the source and destination of the DMA area
0: DMA from/to memory
1: DMA from/to Register File

MULTIFUNCTION TIMER (Cont'd)

DMA ADDRESS POINTER REGISTER (DAPR)
R241 - Read/Write
Register Page: 9
Reset value: undefined

7							0
DAP7	DAP6	DAP5	DAP4	DAP3	DAP2	DMA SRCE	PRG /DAT

Bits 7:2 = **DAP[7:2]**: *MSB of DMA address register location.*
These are the most significant bits of the DMA address register location programmable by software. The DAP2 bit may also be toggled by hardware if the Timer DMA section for the Compare 0 channel is configured in Swap mode.

Note: During a DMA transfer with the Register File, the DAPR is not used; however, in Swap mode, DAP2 is used to point to the correct table.

Bit 1 = **DMA-SRCE**: *DMA source selection.*
This bit is fixed by hardware.
0: DMA source is a Capture on REG0R register
1: DMA destination is a Compare on the CMP0R register

Bit 0 = **PRG/DAT:** *DMA memory selection*.
This bit is set and cleared by software. It is only meaningful if DCPR.REG/MEM=0.
0: The ISR register is used to extend the address of data transferred by DMA (see MMU chapter).
1: The DMASR register is used to extend the address of data transferred by DMA (see MMU chapter).

REG/MEM	PRG/DAT	DMA Source/Destination
0	0	ISR register used to address memory
0	1	DMASR register used to address memory
1	0	Register file
1	1	Register file

INTERRUPT VECTOR REGISTER (T_IVR)

R242 - Read/Write
Register Page: 9
Reset value: xxxx xxx0

7							0
V4	V3	V2	V1	V0	W1	W0	0

This register is used as a vector, pointing to the 16-bit interrupt vectors in memory which contain the starting addresses of the three interrupt subroutines managed by each timer.

Only one Interrupt Vector Register is available for each timer, and it is able to manage three interrupt groups, because the 3 least significant bits are fixed by hardware depending on the group which generated the interrupt request.

In order to determine which request generated the interrupt within a group, the T_FLAGR register can be used to check the relevant interrupt source.

Bits 7:3 = **V[4:0]:** *MSB of the vector address.*
These bits are user programmable and contain the five most significant bits of the Timer interrupt vector addresses in memory. In any case, an 8-bit address can be used to indicate the Timer interrupt vector locations, because they are within the first 256 memory locations (see Interrupt and DMA chapters).

Bits 2:1 = **W[1:0]:** *Vector address bits.*
These bits are equivalent to bit 1 and bit 2 of the Timer interrupt vector addresses in memory. They are fixed by hardware, depending on the group of sources which generated the interrupt request as follows:.

W1	W0	Interrupt Source
0	0	Overflow/Underflow even interrupt
0	1	Not available
1	0	Capture event interrupt
1	1	Compare event interrupt

Bit 0 = This bit is forced by hardware to 0.

MULTIFUNCTION TIMER (Cont'd)

INTERRUPT/DMA CONTROL REGISTER (IDCR)

R243 - Read/Write
Register Page: 9
Reset value: 1100 0111 (C7h)

7							0
CPE	CME	DCTS	DCTD	SWEN	PL2	PL1	PL0

Bit 7 = **CPE**: *Capture 0 EOB*.
This bit is set by hardware when the End Of Block condition is reached during a Capture 0 DMA operation with the Swap mode enabled. When Swap mode is disabled (SWEN bit = "0"), the CPE bit is forced to 1 by hardware.
0: No end of block condition
1: Capture 0 End of block

Bit 6 = **CME**: *Compare 0 EOB*.
This bit is set by hardware when the End Of Block condition is reached during a Compare 0 DMA operation with the Swap mode enabled. When the Swap mode is disabled (SWEN bit = "0"), the CME bit is forced to 1 by hardware.
0: No end of block condition
1: Compare 0 End of block

Bit 5 = **DCTS**: *DMA capture transfer source*.
This bit is set and cleared by software. It selects the source of the DMA operation related to the channel associated with the Capture 0.
Note: The I/O port source is available only on specific devices.
0: REG0R register
1: I/O port.

Bit 4 = **DCTD**: *DMA compare transfer destination*.
This bit is set and cleared by software. It selects the destination of the DMA operation related to the channel associated with Compare 0.
Note: The I/O port destination is available only on specific devices.
0: CMP0R register
1: I/O port

Bit 3 = **SWEN**: *Swap function enable*.
This bit is set and cleared by software.
0: Disable Swap mode
1: Enable Swap mode for both DMA channels.

Bits 2:0 = **PL[2:0]:** *Interrupt/DMA priority level*.
With these three bits it is possible to select the Interrupt and DMA priority level of each timer, as one of eight levels (see Interrupt/DMA chapter).

I/O CONNECTION REGISTER (IOCR)

R248 - Read/Write
Register Page: 9
Reset value: 1111 1100 (FCh)

7							0
						SC1	SC0

Bits 7:2 = not used.

Bit 1 = **SC1**: *Select connection odd.*
This bit is set and cleared by software. It selects if the TxOUTA and TxINA pins for Timer 1 and Timer 3 are connected on-chip or not.
0: T1OUTA / T1INA and T3OUTA/ T3INA unconnected
1: T1OUTA connected internally to T1INA and T3OUTA connected internally to T3INA

Bit 0 = **SC0**: *Select connection even.*
This bit is set and cleared by software. It selects if the TxOUTA and TxINA pins for Timer 0 and Timer 2 are connected on-chip or not.
0: T0OUTA / T0INA and T2OUTA/ T2INA unconnected
1: T0OUTA connected internally to T0INA and T2OUTA connected internally to T2INA

Note: Timer 1 and 2 are available only on some devices. Refer to the device block diagram and register map.

10.5 MULTIPROTOCOL SERIAL COMMUNICATIONS INTERFACE (SCI-M)

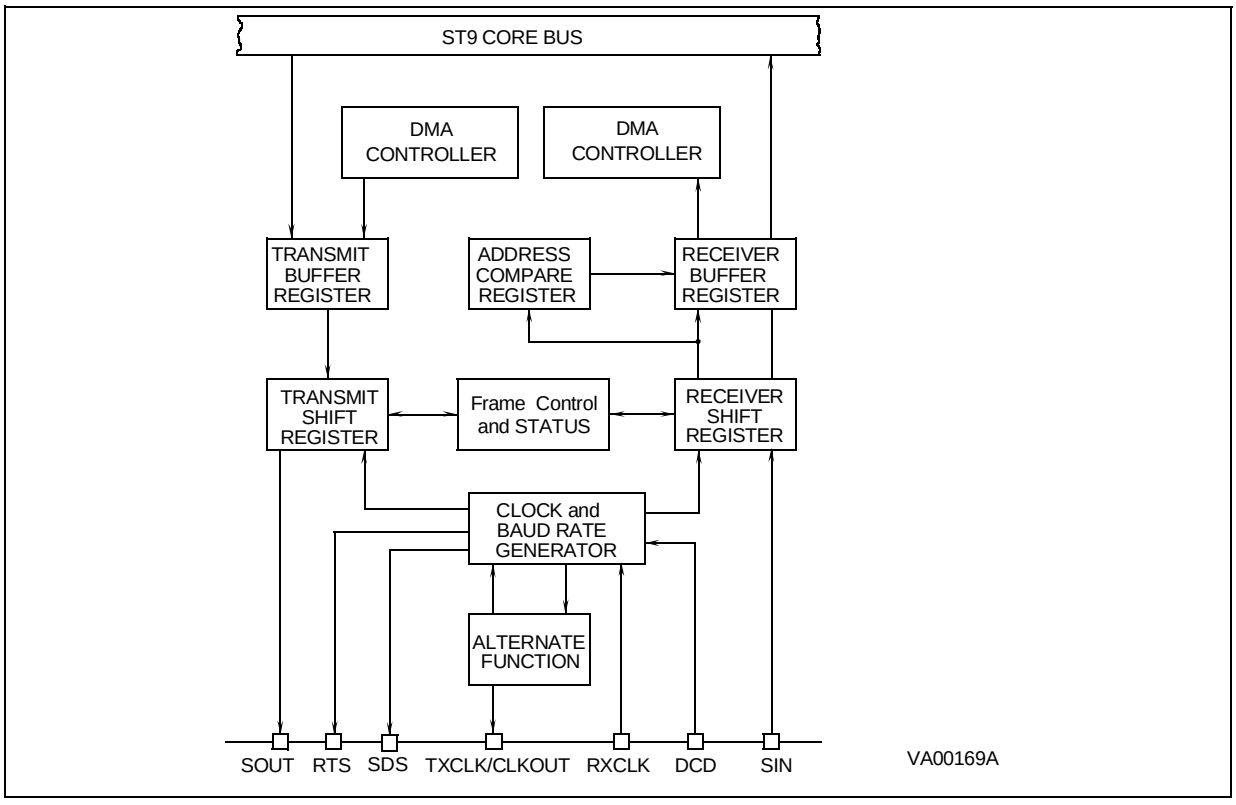
10.5.1 Introduction

The Multiprotocol Serial Communications Interface (SCI-M) offers full-duplex serial data exchange with a wide range of external equipment. The SCI-M offers four operating modes: Asynchronous, Asynchronous with synchronous clock, Serial expansion and Synchronous.

10.5.2 Main Features

- Full duplex synchronous and asynchronous operation.
- Transmit, receive, line status, and device address interrupt generation.
- Integral Baud Rate Generator capable of dividing the input clock by any value from 2 to 2^{16}-1 (16 bit word) and generating the internal 16X data sampling clock for asynchronous operation or the 1X clock for synchronous operation.
- Fully programmable serial interface:
  - 5, 6, 7, or 8 bit word length.
  - Even, odd, or no parity generation and detection.
  - 0, 1, 1.5, 2, 2.5, 3 stop bit generation.
  - Complete status reporting capabilities.
  - Line break generation and detection.
- Programmable address indication bit (wake-up bit) and user invisible compare logic to support multiple microcomputer networking. Optional character search function.
- Internal diagnostic capabilities:
  - Local loopback for communications link fault isolation.
  - Auto-echo for communications link fault isolation.
- Separate interrupt/DMA channels for transmit and receive.
- In addition, a Synchronous mode supports:
  - High speed communication
  - Possibility of hardware synchronization (RTS/DCD signals).
  - Programmable polarity and stand-by level for data SIN/SOUT.
  - Programmable active edge and stand-by level for clocks CLKOUT/RXCL.
  - Programmable active levels of RTS/DCD signals.
  - Full Loop-Back and Auto-Echo modes for DATA, CLOCKs and CONTROLs.

Figure 88. SCI-M Block Diagram



MULTIPROTOCOL SERIAL COMMUNICATIONS INTERFACE (Cont'd)

10.5.3 Functional Description

The SCI-M has four operating modes:

– Asynchronous mode

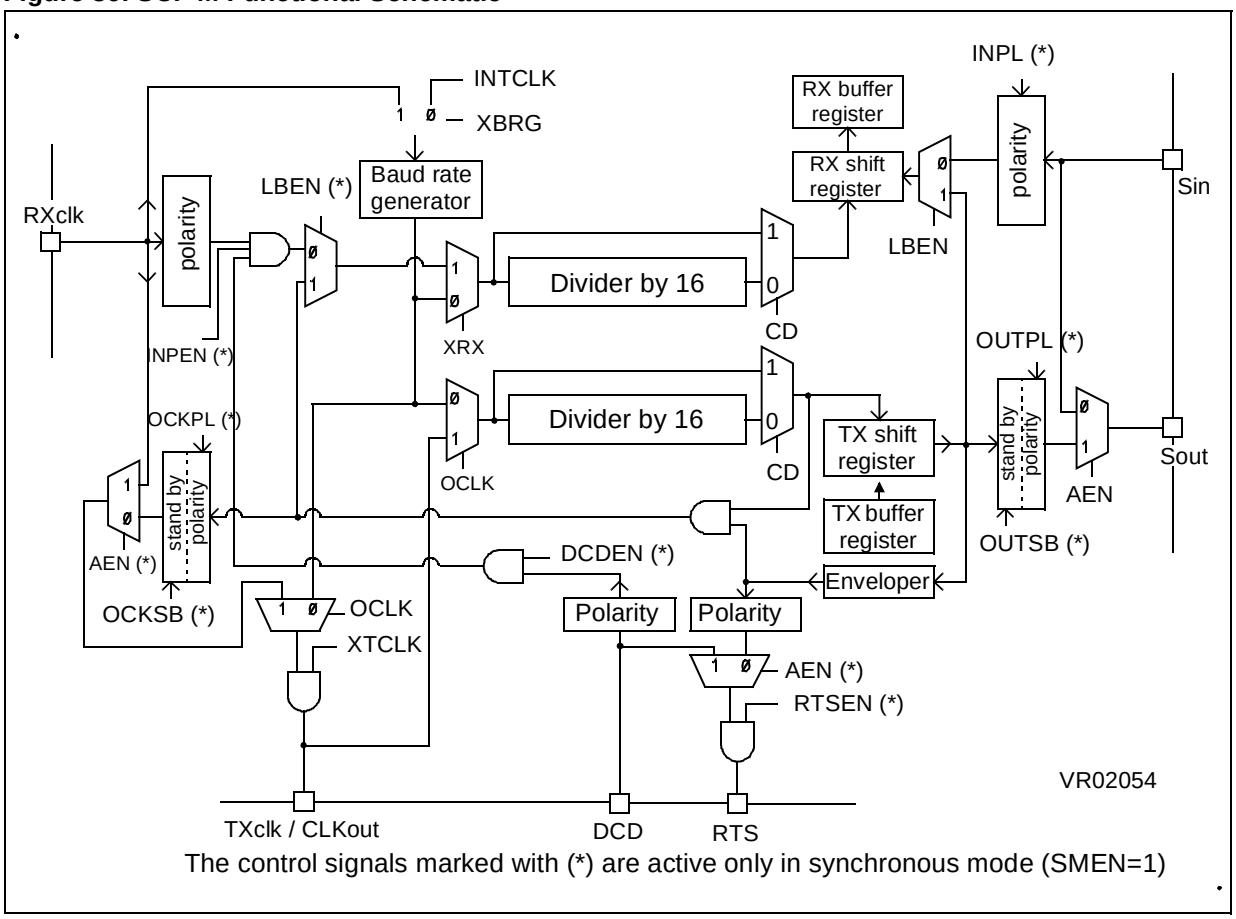
– Asynchronous mode with synchronous clock

– Serial expansion mode

– Synchronous mode

Asynchronous mode, Asynchronous mode with synchronous clock and Serial expansion mode output data with the same serial frame format. The differences lie in the data sampling clock rates (1X, 16X) and in the protocol used.

Figure 89. SCI -M Functional Schematic



Note: Some pins may not be available on some devices. Refer to the device Pinout Description.

MULTIPROTOCOL SERIAL COMMUNICATIONS INTERFACE (Cont'd)

10.5.4 SCI-M Operating Modes

10.5.4.1 Asynchronous Mode

In this mode, data and clock can be asynchronous (the transmitter and receiver can use their own clocks to sample received data), each data bit is sampled 16 times per clock period.

The baud rate clock should be set to the ÷16 Mode and the frequency of the input clock (from an external source or from the internal baud-rate generator output) is set to suit.

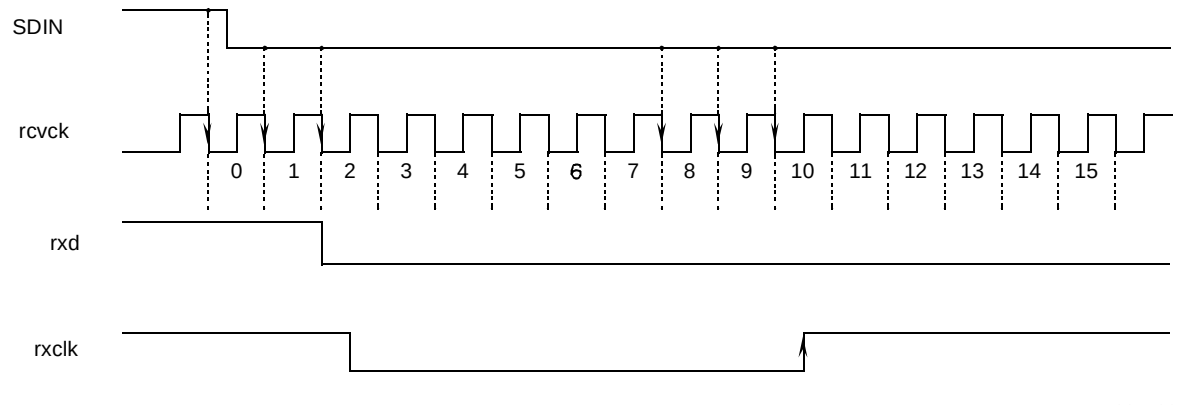
10.5.4.2 Asynchronous Mode with Synchronous Clock

In this mode, data and clock are synchronous, each data bit is sampled once per clock period.

For transmit operation, a general purpose I/O port pin can be programmed to output the CLKOUT signal from the baud rate generator. If the SCI is provided with an external transmission clock source, there will be a skew equivalent to two INTCLK periods between clock and data.

Data will be transmitted on the falling edge of the transmit clock. Received data will be latched into the SCI on the rising edge of the receive clock.

Figure 90. Sampling Times in Asynchronous Format



LEGEND:
SIN: Serial Data Input line
rcvck: Internal X16 Receiver Clock
rxd: Internal Serial Data Input Line
rxclk: Internal Receiver Shift Register Sampling Clock

MULTIPROTOCOL SERIAL COMMUNICATIONS INTERFACE (Cont'd)

10.5.4.3 Serial Expansion Mode

This mode is used to communicate with an external synchronous peripheral.

The transmitter only provides the clock waveform during the period that data is being transmitted on the CLKOUT pin (the Data Envelope). Data is latched on the rising edge of this clock.

Whenever the SCI is to receive data in serial port expansion mode, the clock must be supplied externally, and be synchronous with the transmitted data. The SCI latches the incoming data on the rising edge of the received clock, which is input on the RXCLK pin.

10.5.4.4 Synchronous Mode

This mode is used to access an external synchronous peripheral, dummy start/stop bits are not included in the data frame. Polarity, stand-by level and active edges of I/O signals are fully and separately programmable for both inputs and outputs.

It's necessary to set the SMEN bit of the Synchronous Input Control Register (SICR) to enable this mode and all the related extra features (otherwise disabled).

The transmitter will provide the clock waveform only during the period when the data is being transmitted via the CLKOUT pin, which can be enabled by setting both the XTCLK and OCLK bits of the Clock Configuration Register. Whenever the SCI is to receive data in synchronous mode, the clock waveform must be supplied externally via the RXCLK pin and be synchronous with the data. For correct receiver operation, the XRX bit of the Clock Configuration Register must be set.

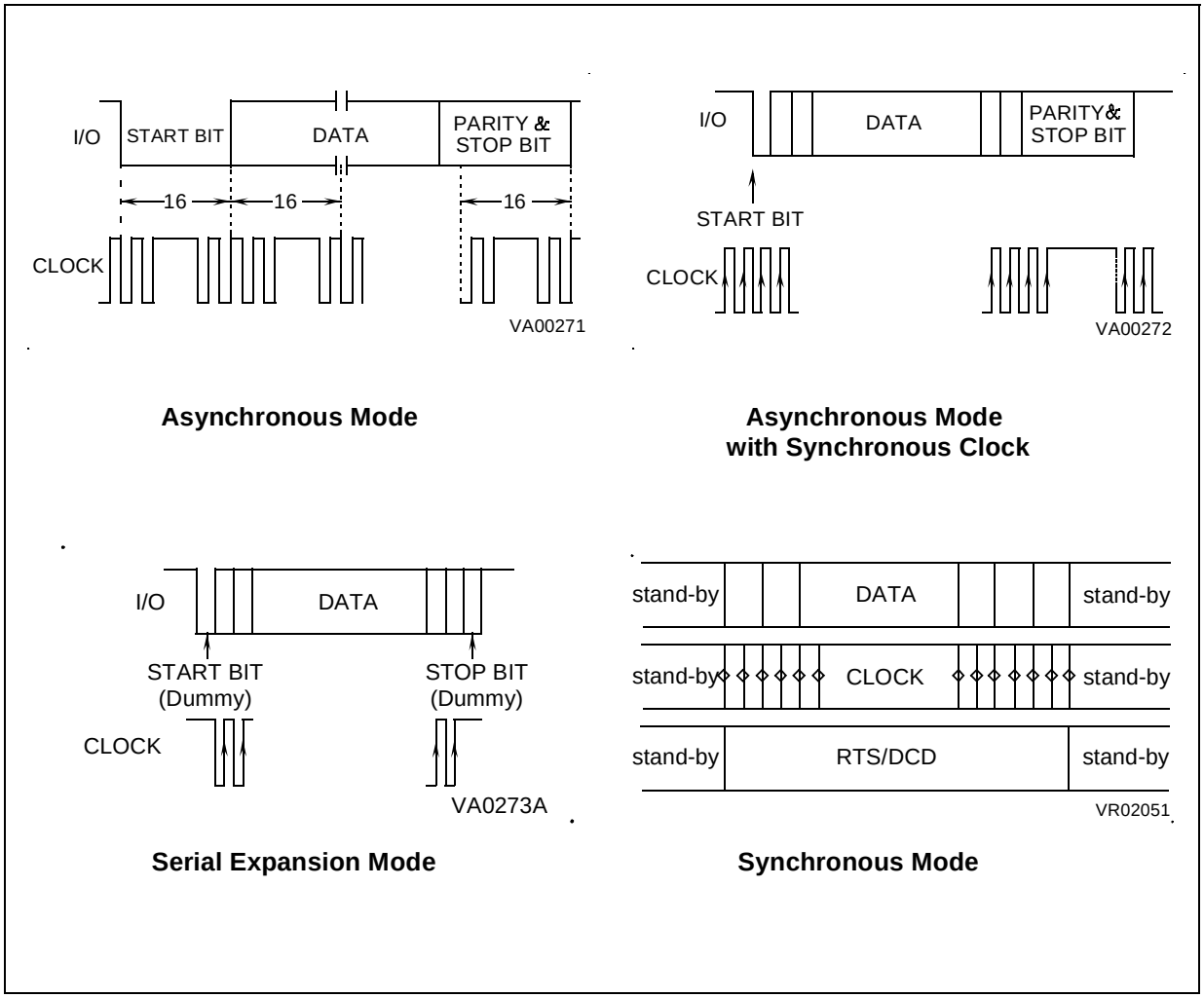
Two external signals, Request-To-Send and Data-Carrier-Detect (RTS/DCD), can be enabled to synchronise the data exchange between two serial units. The RTS output becomes active just before the first active edge of CLKOUT and indicates to the target device that the MCU is about to send a synchronous frame; it returns to its stand-by state following the last active edge of CLKOUT (MSB transmitted).

The DCD input can be considered as a gate that filters RXCLK and informs the MCU that a transmitting device is transmitting a data frame. Polarity of RTS/DCD is individually programmable, as for clocks and data.

The data word is programmable from 5 to 8 bits, as for the other modes; parity, address/9th, stop bits and break cannot be inserted into the transmitted frame. Programming of the related bits of the SCI control registers is irrelevant in Synchronous Mode: all the corresponding interrupt requests must, in any case, be masked in order to avoid incorrect operation during data reception.

MULTIPROTOCOL SERIAL COMMUNICATIONS INTERFACE (Cont'd)

Figure 91. SCI -M Operating Modes



Note: In all operating modes, the Least Significant Bit is transmitted/received first.

MULTIPROTOCOL SERIAL COMMUNICATIONS INTERFACE (Cont'd)

10.5.5 Serial Frame Format

Characters sent or received by the SCI can have some or all of the features in the following format, depending on the operating mode:

START: the START bit indicates the beginning of a data frame in Asynchronous modes. The START condition is detected as a high to low transition.
A dummy START bit is generated in Serial Expansion mode. The START bit is not generated in Synchronous mode.

DATA: the DATA word length is programmable from 5 to 8 bits, for both Synchronous and Asynchronous modes. LSB are transmitted first.

PARITY: The Parity Bit (not available in Serial Expansion mode and Synchronous mode) is optional, and can be used with any word length. It is used for error checking and is set so as to make the total number of high bits in DATA plus PARITY odd or even, depending on the number of "1"s in the DATA field.

ADDRESS/9TH: The Address/9th Bit is optional and may be added to any word format. It is used in both Serial Expansion and Asynchronous modes to indicate that the data is an address (bit set).

The ADDRESS/9TH bit is useful when several microcontrollers are exchanging data on the same serial bus. Individual microcontrollers can stay idle on the serial bus, waiting for a transmitted address. When a microcontroller recognizes its own address, it can begin Data Reception, likewise, on the transmit side, the microcontroller can transmit another address to begin communication with a different microcontroller.

The ADDRESS/9TH bit can be used as an additional data bit or to mark control words (9th bit).

STOP: Indicates the end of a data frame in Asynchronous modes. A dummy STOP bit is generated in Serial Expansion mode. The STOP bit can be programmed to be 1, 1.5, 2, 2.5 or 3 bits long, depending on the mode. It returns the SCI to the quiescent marking state (i.e., a constant high-state condition) which lasts until a new start bit indicates an incoming word. The STOP bit is not generated in Synchronous mode.

Figure 92. SCI Character Formats

	START(2)	DATA(1)	PARITY(3)	ADDRESS(2)	STOP(2)	
# bits	1	5, 6, 7, 8	0, 1	0, 1	1, 1.5, 2, 2.5, 1, 2, 3	16X 1X
states			NONE ODD EVEN	ON OFF		

(1) LSB First

(2) Not available in Synchronous mode

(3) Not available in Serial Expansion mode and Synchronous mode

MULTIPROTOCOL SERIAL COMMUNICATIONS INTERFACE (Cont'd)

10.5.5.1 Data transfer

Data to be transmitted by the SCI is first loaded by the program into the Transmitter Buffer Register. The SCI will transfer the data into the Transmitter Shift Register when the Shift Register becomes available (empty). The Transmitter Shift Register converts the parallel data into serial format for transmission via the SCI Alternate Function output, Serial Data Out. On completion of the transfer, the transmitter buffer register interrupt pending bit will be updated. If the selected word length is less than 8 bits, the unused most significant bits do not need to be defined.

Incoming serial data from the Serial Data Input pin is converted into parallel format by the Receiver Shift Register. At the end of the input data frame, the valid data portion of the received word is transferred from the Receiver Shift Register into the Receiver Buffer Register. All Receiver interrupt conditions are updated at the time of transfer. If the selected character format is less than 8 bits, the unused most significant bits will be set.

The Frame Control and Status block creates and checks the character configuration (Data length and number of Stop bits), as well as the source of the transmitter/receiver clock.

The internal Baud Rate Generator contains a programmable divide by "N" counter which can be used to generate the clocks for the transmitter and/or receiver. The baud rate generator can use INTCLK or the Receiver clock input via RXCLK.

The Address bit/D9 is optional and may be added to any word in Asynchronous and Serial Expansion modes. It is commonly used in network or machine control applications. When enabled (AB set), an address or ninth data bit can be added to a transmitted word by setting the Set Address bit (SA). This is then appended to the next word entered into the (empty) Transmitter Buffer Register and then cleared by hardware. On character input, a set Address Bit can indicate that the data preceding the bit is an address which may be compared in hardware with the value in the Address Compare Register (ACR) to generate an Address Match interrupt when equal.

The Address bit and Address Comparison Register can also be combined to generate four different types of Address Interrupt to suit different protocols, based on the status of the Address Mode Enable bit (AMEN) and the Address Mode bit (AM) in the CHCR register.

The character match Address Interrupt mode may be used as a powerful character search mode, generating an interrupt on reception of a predetermined character e.g. Carriage Return or End of Block codes (Character Match Interrupt). This is the only Address Interrupt Mode available in Synchronous mode.

The Line Break condition is fully supported for both transmission and reception. Line Break is sent by setting the SB bit (IDPR). This causes the transmitter output to be held low (after all buffered data has been transmitted) for a minimum of one complete word length and until the SB bit is Reset. Break cannot be inserted into the transmitted frame for the Synchronous mode.

Testing of the communications channel may be performed using the built-in facilities of the SCI peripheral. Auto-Echo mode and Loop-Back mode may be used individually or together. In Asynchronous, Asynchronous with Synchronous Clock and Serial Expansion modes they are available only on SIN/SOUT pins through the programming of AEN/LBEN bits in CCR. In Synchronous mode (SMEN set) the above configurations are available on SIN/SOUT, RXCLK/CLKOUT and DCD/RTS pins by programming the AEN/LBEN bits and independently of the programmed polarity. In the Synchronous mode case, when AEN is set, the transmitter outputs (data, clock and control) are disconnected from the I/O pins, which are driven directly by the receiver input pins (Auto-Echo mode: SOUT=SIN, CLKOUT=RXCLK and RTS=DCD, even if they act on the internal receiver with the programmed polarity/edge). When LBEN is set, the receiver inputs (data, clock and controls) are disconnected and the transmitter outputs are looped-back into the receiver section (Loop-Back mode: SIN=SOUT, RXCLK=CLKOUT, DCD=RTS. The output pins are locked to their programmed stand-by level and the status of the INPL, XCKPL, DCDPL, OUTPL, OCKPL and RTSPL bits in the SICR register are irrelevant). Refer to Figure 6, Figure 7, and Figure 8 for these different configurations.

Table 37. Address Interrupt Modes

If 9th Data Bit is set (1)
If Character Match
If Character Match and 9th Data Bit is set(1)
If Character Match Immediately Follows BREAK (1)

(1) Not available in Synchronous mode

MULTIPROTOCOL SERIAL COMMUNICATIONS INTERFACE (Cont'd)

Figure 93. Auto Echo Configuration

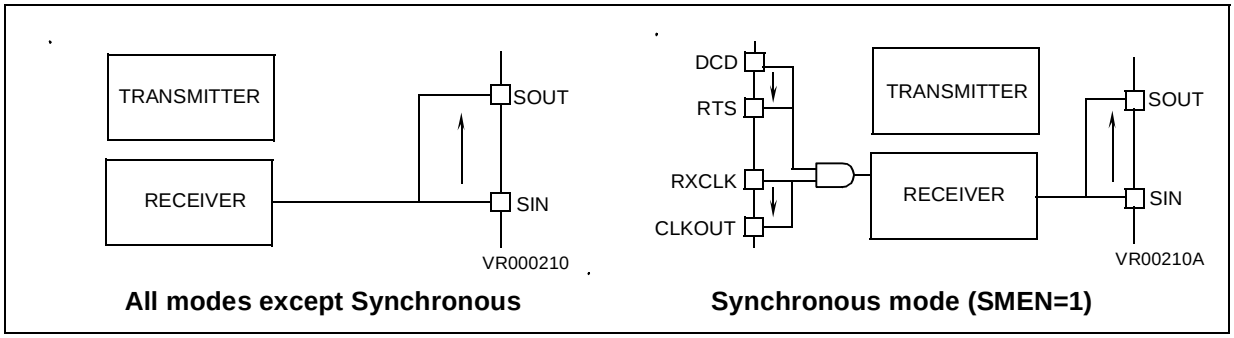


Figure 94. Loop Back Configuration

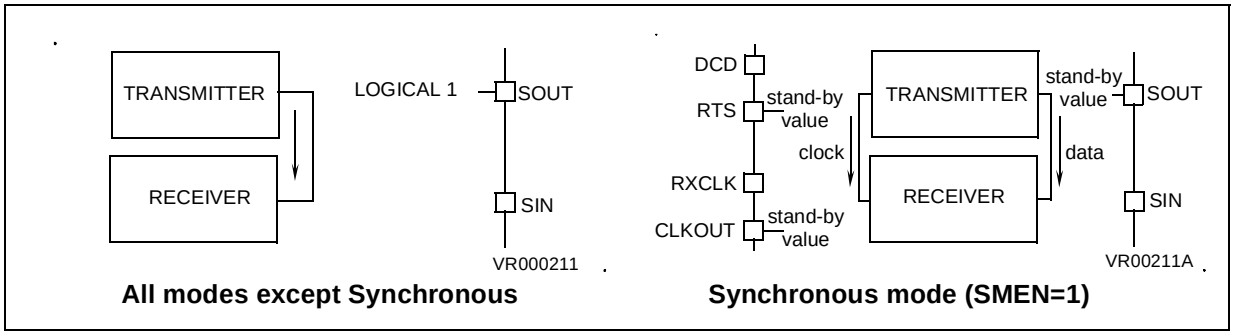
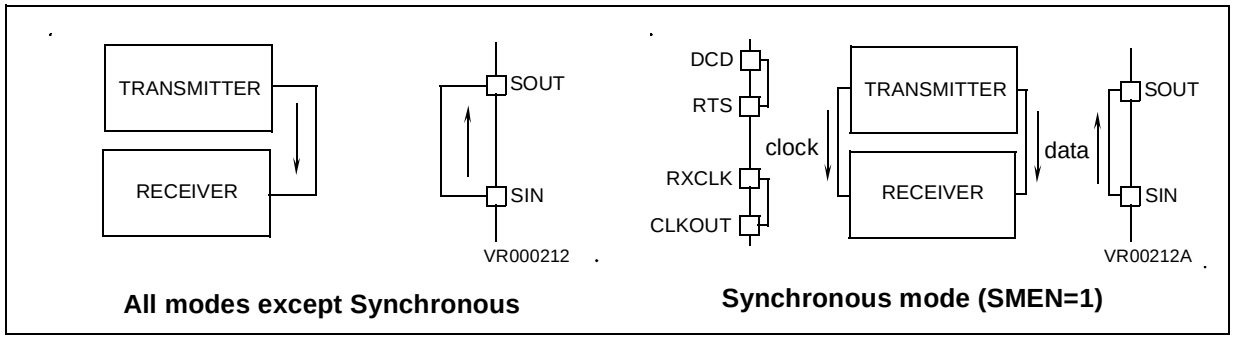


Figure 95. Auto Echo and Loop-Back Configuration



MULTIPROTOCOL SERIAL COMMUNICATIONS INTERFACE (Cont'd)

10.5.6 Clocks And Serial Transmission Rates

The communication bit rate of the SCI transmitter and receiver sections can be provided from the internal Baud Rate Generator or from external sources. The bit rate clock is divided by 16 in Asynchronous mode (CD in CCR reset), or undivided in the 3 other modes (CD set).

With INTCLK running at 24MHz and no external Clock provided, a maximum bit rate of 3MBaud and 750KBaud is available in undivided and divide by-16-mode respectively.

With INTCLK running at 24MHz and an external Clock provided through the RXCLK/TXCLK lines, a maximum bit rate of 3MBaud and 375KBaud is available in undivided and divided by 16 mode respectively (see Figure 10).

External Clock Sources. The External Clock input pin TXCLK may be programmed by the XTCLK and OCLK bits in the CCR register as: the transmit clock input, Baud Rate Generator output (allowing an external divider circuit to provide the receive clock for split rate transmit and receive), or as CLKOUT output in Synchronous and Serial Expansion modes. The RXCLK Receive clock input is enabled by the XRX bit, this input should be set in accordance with the setting of the CD bit.

Baud Rate Generator. The internal Baud Rate Generator consists of a 16-bit programmable divide by "N" counter which can be used to generate the transmitter and/or receiver clocks. The minimum baud rate divisor is 2 and the maximum divisor is 2^{16}-1. After initialising the baud rate generator, the divisor value is immediately loaded into the counter. This prevents potentially long random counts on the initial load.
The Baud Rate generator frequency is equal to the Input Clock frequency divided by the Divisor value.

WARNING: *Programming the baud rate divider to 0 or 1 will stop the divider.*

The output of the Baud Rate generator has a precise 50% duty cycle. The Baud Rate generator can use INTCLK for the input clock source. In this case, INTCLK (and therefore the MCU Xtal) should be chosen to provide a suitable frequency for division by the Baud Rate Generator to give the required transmit and receive bit rates. Suitable INTCLK frequencies and the respective divider values for standard Baud rates are shown in Table 2.

10.5.7 SCI -M Initialization Procedure

Writing to either of the two Baud Rate Generator Registers immediately disables and resets the SCI baud rate generator, as well as the transmitter and receiver circuitry.

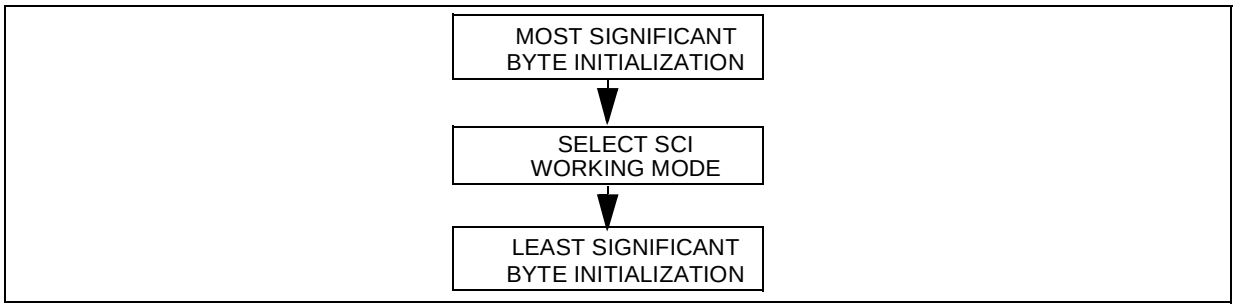
After writing to the second Baud Rate Generator Register, the transmitter and receiver circuits are enabled. The Baud Rate Generator will load the new value and start counting.

To initialize the SCI, the user should first initialize the most significant byte of the Baud Rate Generator Register; this will reset all SCI circuitry. The user should then initialize all other SCI registers (SICR/SOCR included) for the desired operating mode and then, to enable the SCI, he should initialize the least significant byte Baud Rate Generator Register.

'On-the-Fly' modifications of the control registers' content during transmitter/receiver operations, although possible, can corrupt data and produce undesirable spikes on the I/O lines (data, clock and control). Furthermore, modifying the control registers' content without reinitialising the SCI circuitry (during stand-by cycles, waiting to transmit or receive data) must be kept carefully under control by software to avoid spurious data being transmitted or received.

Note: For synchronous receive operation, the data and receive clock must not exhibit significant skew between clock and data. The received data and clock are internally synchronized to INTCLK.

Figure 96. SCI-M Baud Rate Generator Initialization Sequence



MULTIPROTOCOL SERIAL COMMUNICATIONS INTERFACE (Cont'd)

Table 38. SCI-M Baud Rate Generator Divider Values Example 1

INTCLK: 19660.800 KHz							
Baud Rate	Clock Factor	Desired Freq (kHz)	Divisor		Actual Baud Rate	Actual Freq (kHz)	Deviation
			Dec	Hex			
50.00	16 X	0.80000	24576	6000	50.00	0.80000	0.0000%
75.00	16 X	1.20000	16384	4000	75.00	1.20000	0.0000%
110.00	16 X	1.76000	11170	2BA2	110.01	1.76014	-0.00081%
300.00	16 X	4.80000	4096	1000	300.00	4.80000	0.0000%
600.00	16 X	9.60000	2048	800	600.00	9.60000	0.0000%
1200.00	16 X	19.20000	1024	400	1200.00	19.20000	0.0000%
2400.00	16 X	38.40000	512	200	2400.00	38.40000	0.0000%
4800.00	16 X	76.80000	256	100	4800.00	76.80000	0.0000%
9600.00	16 X	153.60000	128	80	9600.00	153.60000	0.0000%
19200.00	16 X	307.20000	64	40	19200.00	307.20000	0.0000%
38400.00	16 X	614.40000	32	20	38400.00	614.40000	0.0000%
76800.00	16 X	1228.80000	16	10	76800.00	1228.80000	0.0000%

Table 39. SCI-M Baud Rate Generator Divider Values Example 2

INTCLK: 24576 KHz							
Baud Rate	Clock Factor	Desired Freq (kHz)	Divisor		Actual Baud Rate	Actual Freq (kHz)	Deviation
			Dec	Hex			
50.00	16 X	0.80000	30720	7800	50.00	0.80000	0.0000%
75.00	16 X	1.20000	20480	5000	75.00	1.20000	0.0000%
110.00	16 X	1.76000	13963	383B	110.01	1.76014	-0.00046%
300.00	16 X	4.80000	5120	1400	300.00	4.80000	0.0000%
600.00	16 X	9.60000	2560	A00	600.00	9.60000	0.0000%
1200.00	16 X	19.20000	1280	500	1200.00	19.20000	0.0000%
2400.00	16 X	38.40000	640	280	2400.00	38.40000	0.0000%
4800.00	16 X	76.80000	320	140	4800.00	76.80000	0.0000%
9600.00	16 X	153.60000	160	A0	9600.00	153.60000	0.0000%
19200.00	16 X	307.20000	80	50	19200.00	307.20000	0.0000%
38400.00	16 X	614.40000	40	28	38400.00	614.40000	0.0000%
76800.00	16 X	1228.80000	20	14	76800.00	1228.80000	0.0000%

MULTIPROTOCOL SERIAL COMMUNICATIONS INTERFACE (Cont'd)

10.5.8 Input Signals

SIN: Serial Data Input. This pin is the serial data input to the SCI receiver shift register.

TXCLK: External Transmitter Clock Input. This pin is the external input clock driving the SCI transmitter. The TXCLK frequency must be greater than or equal to 16 times the transmitter data rate (depending whether the X16 or the X1 clock have been selected). A 50% duty cycle is required for this input and must have a period of at least twice INTCLK. The use of the TXCLK pin is optional.

RXCLK: External Receiver Clock Input. This input is the clock to the SCI receiver when using an external clock source connected to the baud rate generator. INTCLK is normally the clock source. A 50% duty cycle is required for this input and must have a period of at least twice INTCLK. Use of RXCLK is optional.

DCD: Data Carrier Detect. This input is enabled only in Synchronous mode; it works as a gate for the RXCLK clock and informs the MCU that an emitting device is transmitting a synchronous frame. The active level can be programmed as 1 or 0 and must be provided at least one INTCLK period before the first active edge of the input clock.

10.5.9 Output Signals

SOUT: Serial Data Output. This Alternate Function output signal is the serial data output for the SCI transmitter in all operating modes.

CLKOUT: Clock Output. The alternate Function of this pin outputs either the data clock from the transmitter in Serial Expansion or Synchronous modes, or the clock output from the Baud Rate Generator. In Serial expansion mode it will clock only the data portion of the frame and its stand-by state is high: data is valid on the rising edge of the clock. Even in Synchronous mode CLKOUT will only clock the data portion of the frame, but the stand-by level and active edge polarity are programmable by the user.

When Synchronous mode is disabled (SMEN in SICR is reset), the state of the XTCLK and OCLK bits in CCR determine the source of CLKOUT; '11' enables the Serial Expansion Mode.

When the Synchronous mode is enabled (SMEN in SICR is set), the state of the XTCLK and OCLK bits in CCR determine the source of CLKOUT; '00' disables it for PLM applications.

RTS: Request To Send. This output Alternate Function is only enabled in Synchronous mode; it becomes active when the Least Significant Bit of the data frame is sent to the Serial Output Pin (SOUT) and indicates to the target device that the MCU is about to send a synchronous frame; it returns to its stand-by value just after the last active edge of CLKOUT (MSB transmitted). The active level can be programmed high or low.

SDS: Synchronous Data Strobe. This output Alternate function is only enabled in Synchronous mode; it becomes active high when the Least Significant Bit is sent to the Serial Output Pins (SOUT) and indicates to the target device that the MCU is about to send the first bit for each synchronous frame. It is active high on the first bit and it is low for all the rest of the frame. The active level can not be programmed.

Figure 97. Receiver and Transmitter Clock Frequencies

		Min	Max	Conditions
Receiver Clock Frequency	External RXCLK	0	INTCLK/8	1x mode
		0	INTCLK/4	16x mode
	Internal Receiver Clock	0	INTCLK/8	1x mode
		0	INTCLK/2	16x mode
Transmitter Clock Frequency	External TXCLK	0	INTCLK/8	1x mode
		0	INTCLK/4	16x mode
	Internal Transmitter Clock	0	INTCLK/8	1x mode
		0	INTCLK/2	16x mode

Note: The internal receiver and transmitter clocks are the ones applied to the Tx and Rx shift registers (see Figure 1).

MULTIPROTOCOL SERIAL COMMUNICATIONS INTERFACE (Cont'd)

10.5.10 Interrupts and DMA

10.5.10.1 Interrupts

The SCI can generate interrupts as a result of several conditions. Receiver interrupts include data pending, receive errors (overrun, framing and parity), as well as address or break pending. Transmitter interrupts are software selectable for either Transmit Buffer Register Empty (BSN set) or for Transmit Shift Register Empty (BSN reset) conditions.

Typical usage of the Interrupts generated by the SCI peripheral are illustrated in Figure 11.

The SCI peripheral is able to generate interrupt requests as a result of a number of events, several of which share the same interrupt vector. It is therefore necessary to poll S_ISR, the Interrupt Status Register, in order to determine the active trigger. These bits should be reset by the programmer during the Interrupt Service routine.

The four major levels of interrupt are encoded in hardware to provide two bits of the interrupt vector register, allowing the position of the block of pointer vectors to be resolved to an 8 byte block size.

The SCI interrupts have an internal priority structure in order to resolve simultaneous events. Refer also to Section 0.1.4 for more details relating to Synchronous mode.

Table 40. SCI Interrupt Internal Priority

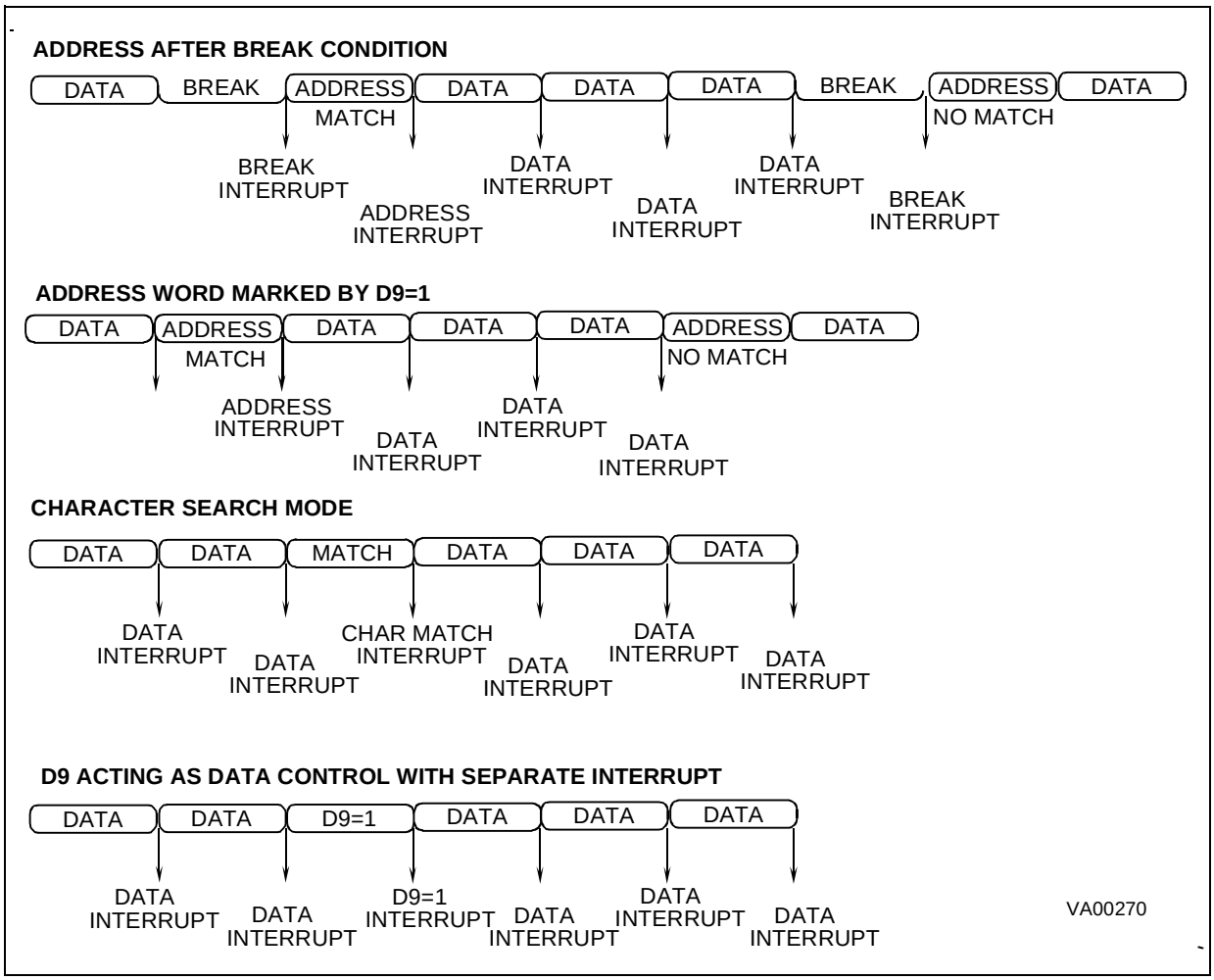
Receive DMA Request	Highest Priority
Transmit DMA Request	
Receive Interrupt	
Transmit Interrupt	Lowest Priority

MULTIPROTOCOL SERIAL COMMUNICATIONS INTERFACE (Cont'd)

Table 41. SCI-M Interrupt Vectors

Interrupt Source	Vector Address
Transmitter Buffer or Shift Register Empty Transmit DMA end of Block	xxx x110
Received Data Pending Receive DMA end of Block	xxxx x100
Break Detector Address Word Match	xxxx x010
Receiver Error	xxxx x000

Figure 98. SCI-M Interrupts: Example of Typical Usage



MULTIPROTOCOL SERIAL COMMUNICATIONS INTERFACE (Cont'd)

10.5.10.2 DMA

Two DMA channels are associated with the SCI, for transmit and for receive. These follow the register scheme as described in the DMA chapter.

DMA Reception

To perform a DMA transfer in reception mode:

1. Initialize the DMA counter (RDCPR) and DMA address (RDAPR) registers
2. Enable DMA by setting the RXD bit in the IDPR register.
3. DMA transfer is started when data is received by the SCI.

DMA Transmission

To perform a DMA transfer in transmission mode:

1. Initialize the DMA counter (TDCPR) and DMA address (TDAPR) registers.
2. Enable DMA by setting the TXD bit in the IDPR register.
3. DMA transfer is started by writing a byte in the Transmitter Buffer register (TXBR).

If this byte is the first data byte to be transmitted, the DMA counter and address registers must be initialized to begin DMA transmission at the second byte. Alternatively, DMA transfer can be started by writing a dummy byte in the TXBR register.

DMA Interrupts

When DMA is active, the Received Data Pending and the Transmitter Shift Register Empty interrupt sources are replaced by the DMA End Of Block receive and transmit interrupt sources.

Note: To handle DMA transfer correctly in transmission, the BSN bit in the IMR register must be cleared. This selects the Transmitter Shift Register Empty event as the DMA interrupt source.

The transfer of the last byte of a DMA data block will be followed by a DMA End Of Block transmit or receive interrupt, setting the TXEOB or RXEOB bit.

A typical Transmission End Of Block interrupt routine will perform the following actions:

1. Restore the DMA counter register (TDCPR).
2. Restore the DMA address register (TDAPR).
3. Clear the Transmitter Shift Register Empty bit TXSEM in the S_ISR register to avoid spurious interrupts.
4. Clear the Transmitter End Of Block (TXEOB) pending bit in the IMR register.
5. Set the TXD bit in the IDPR register to enable DMA.
6. Load the Transmitter Buffer Register (TXBR) with the next byte to transmit.

The above procedure handles the case where a further DMA transfer is to be performed.

Error Interrupt Handling

If an error interrupt occurs while DMA is enabled in reception mode, DMA transfer is stopped.

To resume DMA transfer, the error interrupt handling routine must clear the corresponding error flag. In the case of an Overrun error, the routine must also read the RXBR register.

Character Search Mode with DMA

In Character Search Mode with DMA, when a character match occurs, this character is not transferred. DMA continues with the next received character. To avoid an Overrun error occurring, the Character Match interrupt service routine must read the RXBR register.

MULTIPROTOCOL SERIAL COMMUNICATIONS INTERFACE (Cont'd)

10.5.11 Register Description

The SCI-M registers are located in the following pages in the ST9:

SCI-M number 0: page 24 (18h)

SCI-M number 1: page 25 (19h) (when present)

The SCI is controlled by the following registers:

Address	Register
R240 (F0h)	Receiver DMA Transaction Counter Pointer Register
R241 (F1h)	Receiver DMA Source Address Pointer Register
R242 (F2h)	Transmitter DMA Transaction Counter Pointer Register
R243 (F3h)	Transmitter DMA Destination Address Pointer Register
R244 (F4h)	Interrupt Vector Register
R245 (F5h)	Address Compare Register
R246 (F6h)	Interrupt Mask Register
R247 (F7h)	Interrupt Status Register
R248 (F8h)	Receive Buffer Register same Address as Transmitter Buffer Register (Read Only)
R248 (F8h)	Transmitter Buffer Register same Address as Receive Buffer Register (Write only)
R249 (F9h)	Interrupt/DMA Priority Register
R250 (FAh)	Character Configuration Register
R251 (FBh)	Clock Configuration Register
R252 (FCh)	Baud Rate Generator High Register
R253 (FDh)	Baud Rate Generator Low Register
R254 (FEh)	Synchronous Input Control Register
R255 (FFh)	Synchronous Output Control Register

MULTIPROTOCOL SERIAL COMMUNICATIONS INTERFACE (Cont'd)

RECEIVER DMA COUNTER POINTER (RDCPR)

R240 - Read/Write

Reset value: undefined

7							0
RC7	RC6	RC5	RC4	RC3	RC2	RC1	RR/M

Bit 7:1 = **RC[7:1]**: *Receiver DMA Counter Pointer.*
These bits contain the address of the receiver DMA transaction counter in the Register File.

Bit 0 = **RR/M**: *Receiver Register File/Memory Selector*.
0: Select Memory space as destination.
1: Select the Register File as destination.

RECEIVER DMA ADDRESS POINTER (RDAPR)

R241 - Read/Write

Reset value: undefined

7							0
RA7	RA6	RA5	RA4	RA3	RA2	RA1	RPS

Bit 7:1 = **RA[7:1]**: *Receiver DMA Address Pointer.*
These bits contain the address of the pointer (in the Register File) of the receiver DMA data source.

Bit 0 = **RPS**: *Receiver DMA Memory Pointer Selector.*
This bit is only significant if memory has been selected for DMA transfers (RR/M = 0 in the RDCPR register).
0: Select ISR register for receiver DMA transfers address extension.
1: Select DMASR register for receiver DMA transfers address extension.

TRANSMITTER DMA COUNTER POINTER (TDCPR)

R242 - Read/Write

Reset value: undefined

7							0
TC7	TC6	TC5	TC4	TC3	TC2	TC1	TR/M

Bit 7:1 = **TC[7:1]**: *Transmitter DMA Counter Pointer*.
These bits contain the address of the transmitter DMA transaction counter in the Register File.

Bit 0 = **TR/M**: *Transmitter Register File/Memory Selector*.
0: Select Memory space as source.
1: Select the Register File as source.

TRANSMITTER DMA ADDRESS POINTER (TDAPR)

R243 - Read/Write

Reset value: undefined

7							0
TA7	TA6	TA5	TA4	TA3	TA2	TA1	TPS

Bit 7:1 = **TA[7:1]**: *Transmitter DMA Address Pointer.*
These bits contain the address of the pointer (in the Register File) of the transmitter DMA data source.

Bit 0 = **TPS**: *Transmitter DMA Memory Pointer Selector.*
This bit is only significant if memory has been selected for DMA transfers (TR/M = 0 in the TDCPR register).
0: Select ISR register for transmitter DMA transfers address extension.
1: Select DMASR register for transmitter DMA transfers address extension.

MULTIPROTOCOL SERIAL COMMUNICATIONS INTERFACE (Cont'd)

INTERRUPT VECTOR REGISTER (S_IVR)

R244 - Read/Write

Reset value: undefined

7							0
V7	V6	V5	V4	V3	EV2	EV1	0

Bit 7:3 = **V[7:3]**: *SCI Interrupt Vector Base Address.*
User programmable interrupt vector bits for transmitter and receiver.

Bit 2:1 = **EV[2:1]**: *Encoded Interrupt Source.*
Both bits EV2 and EV1 are read only and set by hardware according to the interrupt source.

EV2	EV1	Interrupt source
0	0	Receiver Error (Overrun, Framing, Parity)
0	1	Break Detect or Address Match
1	0	Received Data Pending/Receiver DMA End of Block
1	1	Transmitter buffer or shift register empty transmitter DMA End of Block

Bit 0 = **D0**: This bit is forced by hardware to 0.

ADDRESS/DATA COMPARE REGISTER (ACR)

R245 - Read/Write

Reset value: undefined

7							0
AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0

Bit 7:0 = **AC[7:0]**: *Address/Compare Character*. With either 9th bit address mode, address after break mode, or character search, the received address will be compared to the value stored in this register. When a valid address matches this register content, the Receiver Address Pending bit (RXAP in the S_ISR register) is set. After the RXAP bit is set in an addressed mode, all received data words will be transferred to the Receiver Buffer Register.

MULTIPROTOCOL SERIAL COMMUNICATIONS INTERFACE (Cont'd)

INTERRUPT MASK REGISTER (IMR)

R246 - Read/Write

Reset value: 0xx00000

7							0
BSN	RXEOB	TXEOB	RXE	RXA	RXB	RXDI	TXDI

Bit 7 = **BSN**: *Buffer or shift register empty interrupt.*
This bit selects the source of the transmitter register empty interrupt.
0: Select a Shift Register Empty as source of a Transmitter Register Empty interrupt.
1: Select a Buffer Register Empty as source of a Transmitter Register Empty interrupt.

Bit 6 = **RXEOB**: *Received End of Block.*
This bit is set by hardware only and must be reset by software. RXEOB is set after a receiver DMA cycle to mark the end of a data block.
0: Clear the interrupt request.
1: Mark the end of a received block of data.

Bit 5 = **TXEOB**: *Transmitter End of Block.*
This bit is set by hardware only and must be reset by software. TXEOB is set after a transmitter DMA cycle to mark the end of a data block.
0: Clear the interrupt request.
1: Mark the end of a transmitted block of data.

Bit 4 = **RXE**: *Receiver Error Mask.*
0: Disable Receiver error interrupts (OE, PE, and FE pending bits in the S_ISR register).
1: Enable Receiver error interrupts.

Bit 3 = **RXA**: *Receiver Address Mask.*
0: Disable Receiver Address interrupt (RXAP pending bit in the S_ISR register).
1: Enable Receiver Address interrupt.

Bit 2 = **RXB**: *Receiver Break Mask.*
0: Disable Receiver Break interrupt (RXBP pending bit in the S_ISR register).
1: Enable Receiver Break interrupt.

Bit 1 = **RXDI**: *Receiver Data Interrupt Mask.*
0: Disable Receiver Data Pending and Receiver End of Block interrupts (RXDP and RXEOB pending bits in the S_ISR register).
1: Enable Receiver Data Pending and Receiver End of Block interrupts.

Note: RXDI has no effect on DMA transfers.

Bit 0 = **TXDI**: *Transmitter Data Interrupt Mask.*
0: Disable Transmitter Buffer Register Empty, Transmitter Shift Register Empty, or Transmitter End of Block interrupts (TXBEM, TXSEM, and TXEOB bits in the S_ISR register).
1: Enable Transmitter Buffer Register Empty, Transmitter Shift Register Empty, or Transmitter End of Block interrupts.

Note: TXDI has no effect on DMA transfers.

MULTIPROTOCOL SERIAL COMMUNICATIONS INTERFACE (Cont'd)

INTERRUPT STATUS REGISTER (S_ISR)

R247 - Read/Write

Reset value: undefined

7							0
OE	FE	PE	RXAP	RXBP	RXDP	TXBEM	TXSEM

Bit 7 = **OE**: *Overrun Error Pending*.
This bit is set by hardware if the data in the Receiver Buffer Register was not read by the CPU before the next character was transferred into the Receiver Buffer Register (the previous data is lost).
0: No Overrun Error.
1: Overrun Error occurred.

Bit 6 = **FE**: *Framing Error Pending bit*.
This bit is set by hardware if the received data word did not have a valid stop bit.
0: No Framing Error.
1: Framing Error occurred.

Note: In the case where a framing error occurs when the SCI is programmed in address mode and is monitoring an address, the interrupt is asserted and the corrupted data element is transferred to the Receiver Buffer Register.

Bit 5 = **PE**: *Parity Error Pending*.
This bit is set by hardware if the received word did not have the correct even or odd parity bit.
0: No Parity Error.
1: Parity Error occurred.

Bit 4 = **RXAP**: *Receiver Address Pending*.
RXAP is set by hardware after an interrupt acknowledged in the address mode.
0: No interrupt in address mode.
1: Interrupt in address mode occurred.

Note: The source of this interrupt is given by the couple of bits (AMEN, AM) as detailed in the IDPR register description.

Bit 3 = **RXBP**: *Receiver Break Pending bit*.
This bit is set by hardware if the received data input is held low for the full word transmission time (start bit, data bits, parity bit, stop bit).
0: No break received.
1: Break event occurred.

Bit 2 = **RXDP**: *Receiver Data Pending bit.*
This bit is set by hardware when data is loaded into the Receiver Buffer Register.
0: No data received.
1: Data received in Receiver Buffer Register.

Bit 1 = **TXBEM**: *Transmitter Buffer Register Empty*.
This bit is set by hardware if the Buffer Register is empty.
0: No Buffer Register Empty event.
1: Buffer Register Empty.

Bit 0 = **TXSEM**: *Transmitter Shift Register Empty*.
This bit is set by hardware if the Shift Register has completed the transmission of the available data.
0: No Shift Register Empty event.
1: Shift Register Empty.

Note: The Interrupt Status Register bits can be reset but cannot be set by the user. The interrupt source must be cleared by resetting the related bit when executing the interrupt service routine (naturally the other pending bits should not be reset).

MULTIPROTOCOL SERIAL COMMUNICATIONS INTERFACE (Cont'd)

RECEIVER BUFFER REGISTER (RXBR)

R248 - Read only

Reset value: undefined

7							0
RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0

Bit 7:0 = **RD[7:0]**: *Received Data.*
This register stores the data portion of the received word. The data will be transferred from the Receiver Shift Register into the Receiver Buffer Register at the end of the word. All receiver interrupt conditions will be updated at the time of transfer. If the selected character format is less than 8 bits, unused most significant bits will forced to "1".

Note: RXBR and TXBR are two physically different registers located at the same address.

TRANSMITTER BUFFER REGISTER (TXBR)

R248 - Write only

Reset value: undefined

7							0
TD7	TD6	TD5	TD4	TD3	TD2	TD1	TD0

Bit 7:0 = **TD[7:0]**: *Transmit Data.*
The ST9 core will load the data for transmission into this register. The SCI will transfer the data from the buffer into the Shift Register when available. At the transfer, the Transmitter Buffer Register interrupt is updated. If the selected word format is less than 8 bits, the unused most significant bits are not significant.

Note: TXBR and RXBR are two physically different registers located at the same address.

MULTIPROTOCOL SERIAL COMMUNICATIONS INTERFACE (Cont'd)

INTERRUPT/DMA PRIORITY REGISTER (IDPR)

R249 - Read/Write

Reset value: undefined

7							0
AMEN	SB	SA	RXD	TXD	PRL2	PRL1	PRL0

Bit 7 = **AMEN**: *Address Mode Enable.*
This bit, together with the AM bit (in the CHCR register), decodes the desired addressing/9th data bit/character match operation.

In Address mode the SCI monitors the input serial data until its address is detected

AMEN	AM	
0	0	Address interrupt if 9th data bit = 1
0	1	Address interrupt if character match
1	0	Address interrupt if character match and 9th data bit =1
1	1	Address interrupt if character match with word immediately following Break

Note: Upon reception of address, the RXAP bit (in the Interrupt Status Register) is set and an interrupt cycle can begin. The address character will not be transferred into the Receiver Buffer Register but all data following the matched SCI address and preceding the next address word will be transferred to the Receiver Buffer Register and the proper interrupts updated. If the address does not match, all data following this unmatched address will not be transferred to the Receiver Buffer Register.

In any of the cases the RXAP bit must be reset by software before the next word is transferred into the Buffer Register.

When AMEN is reset and AM is set, a useful character search function is performed. This allows the SCI to generate an interrupt whenever a specific character is encountered (e.g. Carriage Return).

Bit 6 = **SB**: *Set Break.*
0: Stop the break transmission after minimum break length.
1: Transmit a break following the transmission of all data in the Transmitter Shift Register and the Buffer Register.

Note: The break will be a low level on the transmitter data output for at least one complete word format. If software does not reset SB before the minimum break length has finished, the break condition will continue until software resets SB. The SCI terminates the break condition with a high level on the transmitter data output for one transmission clock period.

Bit 5 = **SA**: *Set Address.*
If an address/9th data bit mode is selected, SA value will be loaded for transmission into the Shift Register. This bit is cleared by hardware after its load.
0: Indicate it is not an address word.
1: Indicate an address word.

Note: Proper procedure would be, when the Transmitter Buffer Register is empty, to load the value of SA and then load the data into the Transmitter Buffer Register.

Bit 4 = **RXD**: *Receiver DMA Mask.*
This bit is reset by hardware when the transaction counter value decrements to zero. At that time a receiver End of Block interrupt can occur.
0: Disable Receiver DMA request (the RXDP bit in the S_ISR register can request an interrupt).
1: Enable Receiver DMA request (the RXDP bit in the S_ISR register can request a DMA transfer).

Bit 3 = **TXD**: *Transmitter DMA Mask.*
This bit is reset by hardware when the transaction counter value decrements to zero. At that time a transmitter End Of Block interrupt can occur.
0: Disable Transmitter DMA request (TXBEM or TXSEM bits in S_ISR can request an interrupt).
1: Enable Transmitter DMA request (TXBEM or TXSEM bits in S_ISR can request a DMA transfer).

Bit 2:0 = **PRL[2:0]**: *SCI Interrupt/DMA Priority bits.*
The priority for the SCI is encoded with (PRL2,PRL1,PRL0). Priority level 0 is the highest, while level 7 represents no priority.

When the user has defined a priority level for the SCI, priorities within the SCI are hardware defined. These SCI internal priorities are:

Receiver DMA request	highest priority
Transmitter DMA request	
Receiver interrupt	
Transmitter interrupt	lowest priority

MULTIPROTOCOL SERIAL COMMUNICATIONS INTERFACE (Cont'd)

CHARACTER CONFIGURATION REGISTER (CHCR)

R250 - Read/Write

Reset value: undefined

7							0
AM	EP	PEN	AB	SB1	SB0	WL1	WL0

Bit 7 = **AM**: *Address Mode*.
This bit, together with the AMEN bit (in the IDPR register), decodes the desired addressing/9th data bit/character match operation. Please refer to the table in the IDPR register description.

Bit 6 = **EP**: *Even Parity*.
0: Select odd parity (when parity is enabled).
1: Select even parity (when parity is enabled).

Bit 5 = **PEN**: *Parity Enable*.
0: No parity bit.
1: Parity bit generated (transmit data) or checked (received data).

Note: If the address/9th bit is enabled, the parity bit will precede the address/9th bit (the 9th bit is never included in the parity calculation).

Bit 4 = **AB**: *Address/9th Bit*.
0: No Address/9th bit.
1: Address/9th bit included in the character format between the parity bit and the first stop bit. This bit can be used to address the SCI or as a ninth data bit.

Bit 3:2 = **SB[1:0]**: *Number of Stop Bits.*.

SB1	SB0	Number of stop bits	
		in 16X mode	in 1X mode
0	0	1	1
0	1	1.5	2
1	0	2	2
1	1	2.5	3

Bit 1:0 = **WL[1:0]**: *Number of Data Bits*

WL1	WL0	Data Length
0	0	5 bits
0	1	6 bits
1	0	7 bits
1	1	8 bits

MULTIPROTOCOL SERIAL COMMUNICATIONS INTERFACE (Cont'd)

CLOCK CONFIGURATION REGISTER (CCR)

R251 - Read/Write

Reset value: 0000 0000 (00h)

7							0
XTCLK	OCLK	XRX	XBRG	CD	AEN	LBEN	STPEN

Bit 7 = **XTCLK**
This bit, together with the OCLK bit, selects the source for the transmitter clock. The following table shows the coding of XTCLK and OCLK.

Bit 6 = **OCLK**
This bit, together with the XTCLK bit, selects the source for the transmitter clock. The following table shows the coding of XTCLK and OCLK.

XTCLK	OCLK	Pin Function
0	0	Pin is used as a general I/O
0	1	Pin = TXCLK (used as an input)
1	0	Pin = CLKOUT (outputs the Baud Rate Generator clock)
1	1	Pin = CLKOUT (outputs the Serial expansion and synchronous mode clock)

Bit 5 = **XRX**: *External Receiver Clock Source*.
0: External receiver clock source not used.
1: Select the external receiver clock source.

Note: The external receiver clock frequency must be 16 times the data rate, or equal to the data rate, depending on the status of the CD bit.

Bit 4 = **XBRG**: *Baud Rate Generator Clock Source*.
0: Select INTCLK for the baud rate generator.
1: Select the external receiver clock for the baud rate generator.

Bit 3 = **CD**: *Clock Divisor*.
The status of CD will determine the SCI configuration (synchronous/asynchronous).
0: Select 16X clock mode for both receiver and transmitter.
1: Select 1X clock mode for both receiver and transmitter.

Note: In 1X clock mode, the transmitter will transmit data at one data bit per clock period. In 16X mode each data bit period will be 16 clock periods long.

Bit 2 = **AEN**: *Auto Echo Enable*.
0: No auto echo mode.
1: Put the SCI in auto echo mode.

Note: Auto Echo mode has the following effect: the SCI transmitter is disconnected from the data-out pin SOUT, which is driven directly by the receiver data-in pin, SIN. The receiver remains connected to SIN and is operational, unless loopback mode is also selected.

Bit 1 = **LBEN**: *Loopback Enable*.
0: No loopback mode.
1: Put the SCI in loopback mode.

Note: In this mode, the transmitter output is set to a high level, the receiver input is disconnected, and the output of the Transmitter Shift Register is looped back into the Receiver Shift Register input. All interrupt sources (transmitter and receiver) are operational.

Bit 0 = **STPEN**: *Stick Parity Enable*.
0: The transmitter and the receiver will follow the parity of even parity bit EP in the CHCR register.
1: The transmitter and the receiver will use the opposite parity type selected by the even parity bit EP in the CHCR register.

EP	SPEN	Parity (Transmitter & Receiver)
0 (odd)	0	Odd
1 (even)	0	Even
0 (odd)	1	Even
1 (even)	1	Odd

MULTIPROTOCOL SERIAL COMMUNICATIONS INTERFACE (Cont'd)

BAUD RATE GENERATOR HIGH REGISTER (BRGHR)

R252 - Read/Write

Reset value: undefined

15							8
BG15	BG14	BG13	BG12	BG11	BG10	BG9	BG8

BAUD RATE GENERATOR LOW REGISTER (BRGLR)

R253 **-** Read/Write

Reset value: undefined

7							0
BG7	BG6	BG5	BG4	BG3	BG2	BG1	BG0

Bit 15:0 = *Baud Rate Generator MSB and LSB.*

The Baud Rate generator is a programmable divide by "N" counter which can be used to generate the clocks for the transmitter and/or receiver. This counter divides the clock input by the value in the Baud Rate Generator Register. The minimum baud rate divisor is 2 and the maximum divisor is 2^{16}-1. After initialization of the baud rate generator, the divisor value is immediately loaded into the counter. This prevents potentially long random counts on the initial load. If set to 0 or 1, the Baud Rate Generator is stopped.

SYNCHRONOUS INPUT CONTROL (SICR)

R254 - Read/Write

Reset value: 0000 0011 (03h)

7							0
SMEN	INPL	XCKPL	DCDEN	DCDPL	INPEN	X	X

Bit 7 = **SMEN**: *Synchronous Mode Enable*.
0: Disable all features relating to Synchronous mode (the contents of SICR and SOCR are ignored).
1: Select Synchronous mode with its programmed I/O configuration.

Bit 6 = **INPL**: *SIN Input Polarity*.
0: Polarity not inverted.
1: Polarity inverted.

Note: INPL only affects received data. In Auto-Echo mode SOUT = SIN even if INPL is set. In Loop-Back mode the state of the INPL bit is irrelevant.

Bit 5 = **XCKPL**: *Receiver Clock Polarity*.
0: RXCLK is active on the rising edge.
1: RXCLK is active on the falling edge.

Note: XCKPL only affects the receiver clock. In Auto-Echo mode CLKOUT = RXCLK independently of the XCKPL status. In Loop-Back the state of the XCKPL bit is irrelevant.

Bit 4 = **DCDEN**: *DCD Input Enable*.
0: Disable hardware synchronization.
1: Enable hardware synchronization.

Note: When DCDEN is set, RXCLK drives the receiver section only during the active level of the DCD input (DCD works as a gate on RXCLK, informing the MCU that a transmitting device is sending a synchronous frame to it).

Bit 3 = **DCDPL**: *DCD Input Polarity*.
0: The DCD input is active when LOW.
1: The DCD input is active when HIGH.

Note: DCDPL only affects the gating activity of the receiver clock. In Auto-Echo mode RTS = DCD independently of DCDPL. In Loop-Back mode, the state of DCDPL is irrelevant.

Bit 2 = **INPEN**: *All Input Disable*.
0: Enable SIN/RXCLK/DCD inputs.
1: Disable SIN/RXCLK/DCD inputs.

Bit 1:0 = "Don't Care"

MULTIPROTOCOL SERIAL COMMUNICATIONS INTERFACE (Cont'd)

SYNCHRONOUS OUTPUT CONTROL (SOCR)

R255 - Read/Write

Reset value: 0000 0001 (01h)

7							0
OUTPL	OUTSB	OCKPL	OCKSB	RTSEN	RTSPL	OUTDIS	X

Bit 7 = **OUTPL**: *SOUT Output Polarity.*
0: Polarity not inverted.
1: Polarity inverted.

Note: OUTPL only affects the data sent by the transmitter section. In Auto-Echo mode SOUT = SIN even if OUTPL=1. In Loop-Back mode, the state of OUTPL is irrelevant.

Bit 6 = **OUTSB**: *SOUT Output Stand-By Level.*
0: SOUT stand-by level is HIGH.
1: SOUT stand-by level is LOW.

Bit 5 = **OCKPL**: *Transmitter Clock Polarity.*
0: CLKOUT is active on the rising edge.
1: CLKOUT is active on the falling edge.

Note: OCKPL only affects the transmitter clock. In Auto-Echo mode CLKOUT = RXCLK independently of the state of OCKPL. In Loop-Back mode the state of OCKPL is irrelevant.

Bit 4 = **OCKSB**: *Transmitter Clock Stand-By Level.*
0: The CLKOUT stand-by level is HIGH.
1: The CLKOUT stand-by level is LOW.

Bit 3 = **RTSEN**: *RTS and SDS Output Enable.*
0: Disable the RTS and SDS hardware synchronisation.
1: Enable the RTS and SDS hardware synchronisation.

Notes:

– When RTSEN is set, the RTS output becomes active just before the first active edge of CLKOUT and indicates to target device that the MCU is about to send a synchronous frame; it returns to its stand-by value just after the last active edge of CLKOUT (MSB transmitted).

– When RTSEN is set, the SDS output becomes active high and indicates to the target device that the MCU is about to send the first bit of a synchronous frame on the Serial Output Pin (SOUT); it returns to low level as soon as the second bit is sent on the Serial Output Pin (SOUT). In this way a positive pulse is generated each time that the first bit of a synchronous frame is present on the Serial Output Pin (SOUT).

Bit 2 = **RTSPL**: *RTS Output Polarity.*
0: The RTS output is active when LOW.
1: The RTS output is active when HIGH.

Note: RTSPL only affects the RTS activity on the output pin. In Auto-Echo mode RTS = DCD independently from the RTSPL value. In Loop-Back mode RTSPL value is 'Don't Care'.

Bit 1 = **OUTDIS**: *Disable all outputs.*
This feature is available on specific devices only (see device pin-out description).
When OUTDIS=1, all output pins (if configured in Alternate Function mode) will be put in High Impedance for networking.
0: SOUT/CLKOUT/enabled
1: SOUT/CLKOUT/RTS put in high impedance

Bit 0 = "Don't Care"

10.6 SERIAL PERIPHERAL INTERFACE (SPI)

10.6.1 Introduction

The Serial Peripheral Interface (SPI) allows full-duplex, synchronous, serial communication with external devices. An SPI system may consist of a master and one or more slaves or a system in which devices may be either masters or slaves.

The SPI is normally used for communication between the microcontroller and external peripherals or another Microcontroller.

Refer to the Pin Description chapter for the device-specific pin-out.

10.6.2 Main Features

- Full duplex, three-wire synchronous transfers
- Master or slave operation
- Maximum slave mode frequency = INTCLK/2.
- Programmable prescalers for a wide range of baud rates
- Programmable clock polarity and phase
- End of transfer interrupt flag
- Write collision flag protection
- Master mode fault protection capability.

10.6.3 General Description

The SPI is connected to external devices through 4 alternate function pins:

– MISO: Master In Slave Out pin

– MOSI: Master Out Slave In pin

– SCK: Serial Clock pin

– $\overline{SS}$: Slave select pin

To use any of these alternate functions (input or output), the corresponding I/O port must be programmed as alternate function output.

A basic example of interconnections between a single master and a single slave is illustrated on Figure 1.

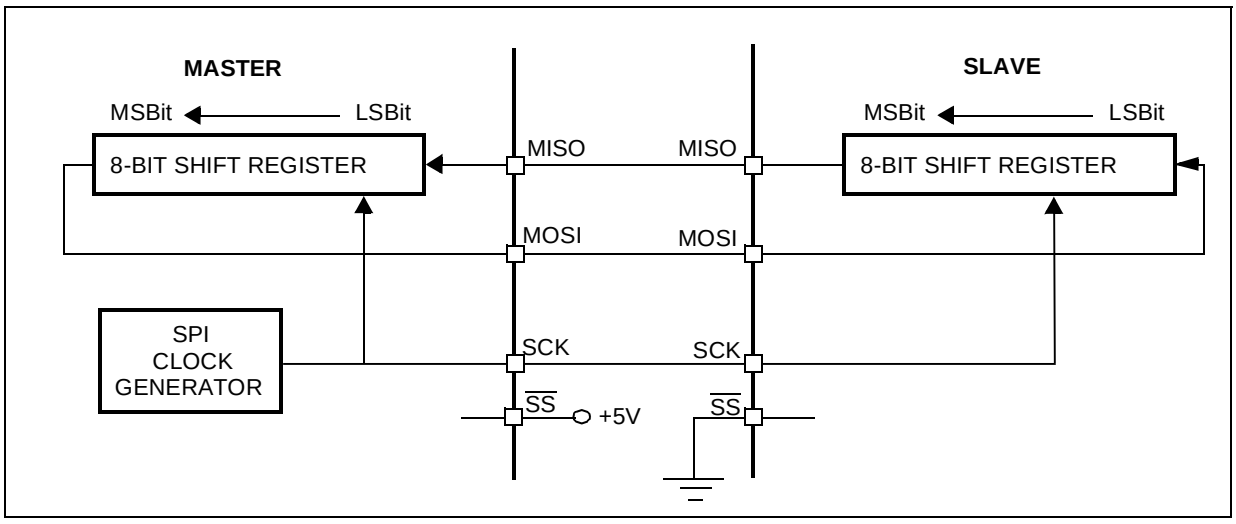
The MOSI pins are connected together as are MISO pins. In this way data is transferred serially between master and slave.

When the master device transmits data to a slave device via MOSI pin, the slave device responds by sending data to the master device via the MISO pin. This implies full duplex transmission with both data out and data in synchronized with the same clock signal (which is provided by the master device via the SCK pin).

Thus, the byte transmitted is replaced by the byte received and eliminates the need for separate transmit-empty and receiver-full bits. A status flag is used to indicate that the I/O operation is complete.

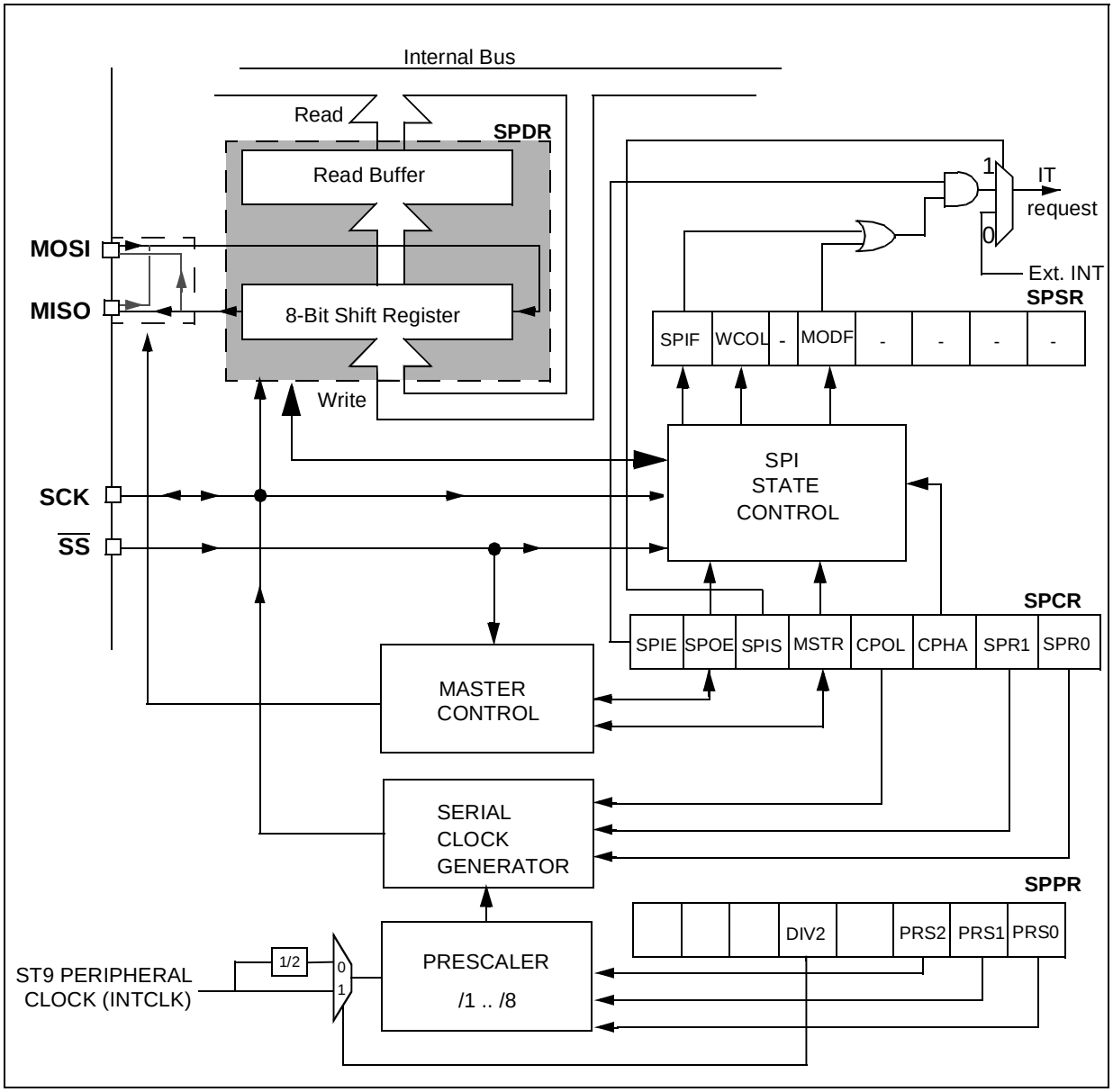
Various data/clock timing relationships may be chosen (see Figure 4) but master and slave must be programmed with the same timing mode.

Figure 99. Serial Peripheral Interface Master/Slave



SERIAL PERIPHERAL INTERFACE (Cont'd)

Figure 100. Serial Peripheral Interface Block Diagram



SERIAL PERIPHERAL INTERFACE (Cont'd)

10.6.4 Functional Description

Figure 2 shows the serial peripheral interface (SPI) block diagram.

This interface contains 4 dedicated registers:

- A Control Register (SPCR)
- A Prescaler Register (SPPR)
- A Status Register (SPSR)
- A Data Register (SPDR)

Refer to the SPCR, SPPR, SPSR and SPDR registers in Section 0.1.6 for the bit definitions.

10.6.4.1 Master Configuration

In a master configuration, the serial clock is generated on the SCK pin.

Procedure

- Define the serial clock baud rate by setting/resetting the DIV2 bit of SPPR register, by writing a prescaler value in the SPPR register and programming the SPR0 & SPR1 bits in the SPCR register.
- Select the CPOL and CPHA bits to define one of the four relationships between the data transfer and the serial clock (see Figure 4).
- The $\overline{SS}$ pin must be connected to a high level signal during the complete byte transmit sequence.
- The MSTR and SPOE bits must be set (they remain set only if the $\overline{SS}$ pin is connected to a high level signal).

In this configuration the MOSI pin is a data output and the MISO pin is a data input.

Transmit Sequence

The transmit sequence begins when a byte is written the SPDR register.

The data byte is parallel loaded into the 8-bit shift register (from the internal bus) during a write cycle and then shifted out serially to the MOSI pin most significant bit first.

When data transfer is complete:

- The SPIF bit is set by hardware
- An interrupt is generated if the SPIS and SPIE bits are set.

During the last clock cycle the SPIF bit is set, a copy of the data byte received in the shift register is moved to a buffer. When the SPDR register is read, the SPI peripheral returns this buffered value.

Clearing the SPIF bit is performed by the following software sequence:

1. An access to the SPSR register while the SPIF bit is set
2. A read of the SPDR register.

Note: While the SPIF bit is set, all writes to the SPDR register are inhibited until the SPSR register is read.

SERIAL PERIPHERAL INTERFACE (Cont'd)

10.6.4.2 Slave Configuration

In slave configuration, the serial clock is received on the SCK pin from the master device.

The value of the SPPR register and SPR0 & SPR1 bits in the SPCR is not used for the data transfer.

Procedure

- For correct data transfer, the slave device must be in the same timing mode as the master device (CPOL and CPHA bits). See Figure 4.
- The $\overline{SS}$ pin must be connected to a low level signal during the complete byte transmit sequence.
- Clear the MSTR bit and set the SPOE bit to assign the pins to alternate function.

In this configuration the MOSI pin is a data input and the MISO pin is a data output.

Transmit Sequence

The data byte is parallel loaded into the 8-bit shift register (from the internal bus) during a write cycle and then shifted out serially to the MISO pin most significant bit first.

The transmit sequence begins when the slave device receives the clock signal and the most significant bit of the data on its MOSI pin.

When data transfer is complete:

- The SPIF bit is set by hardware
- An interrupt is generated if the SPIS and SPIE bits are set.

During the last clock cycle the SPIF bit is set, a copy of the data byte received in the shift register is moved to a buffer. When the SPDR register is read, the SPI peripheral returns this buffered value.

Clearing the SPIF bit is performed by the following software sequence:

1. An access to the SPSR register while the SPIF bit is set.
2. A read of the SPDR register.

Notes: While the SPIF bit is set, all writes to the SPDR register are inhibited until the SPSR register is read.

The SPIF bit can be cleared during a second transmission; however, it must be cleared before the second SPIF bit in order to prevent an overrun condition (see Section 0.1.4.6).

Depending on the CPHA bit, the $\overline{SS}$ pin has to be set to write to the SPDR register between each data byte transfer to avoid a write collision (see Section 0.1.4.4).

SERIAL PERIPHERAL INTERFACE (Cont'd)

10.6.4.3 Data Transfer Format

During an SPI transfer, data is simultaneously transmitted (shifted out serially) and received (shifted in serially). The serial clock is used to synchronize the data transfer during a sequence of eight clock pulses.

The $\overline{SS}$ pin allows individual selection of a slave device; the other slave devices that are not selected do not interfere with the SPI transfer.

Clock Phase and Clock Polarity

Four possible timing relationships may be chosen by software, using the CPOL and CPHA bits.

The CPOL (clock polarity) bit controls the steady state value of the clock when no data is being transferred. This bit affects both master and slave modes.

The combination between the CPOL and CPHA (clock phase) bits selects the data capture clock edge.

Figure 4 shows an SPI transfer with the four combinations of the CPHA and CPOL bits. The diagram may be interpreted as a master or slave timing diagram where the SCK pin, the MISO pin, the MOSI pin are directly connected between the master and the slave device.

The $\overline{SS}$ pin is the slave device select input and can be driven by the master device.

The master device applies data to its MOSI pin-clock edge before the capture clock edge.

CPHA Bit is Set

The second edge on the SCK pin (falling edge if the CPOL bit is reset, rising edge if the CPOL bit is set) is the MSBit capture strobe. Data is latched on the occurrence of the first clock transition.

No write collision should occur even if the $\overline{SS}$ pin stays low during a transfer of several bytes (see Figure 3).

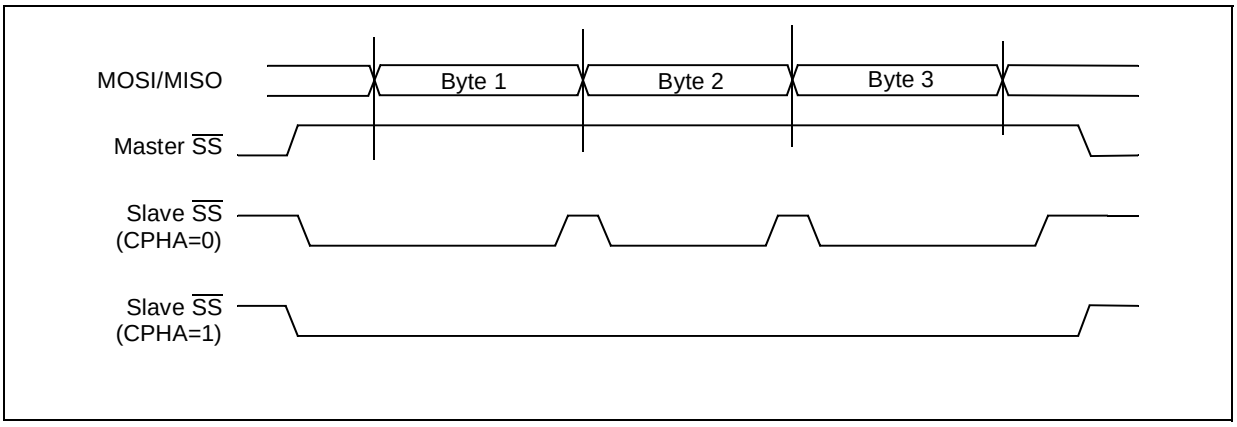
CPHA Bit is Reset

The first edge on the SCK pin (falling edge if CPOL bit is set, rising edge if CPOL bit is reset) is the MSBit capture strobe. Data is latched on the occurrence of the second clock transition.

This pin must be toggled high and low between each byte transmitted (see Figure 3).

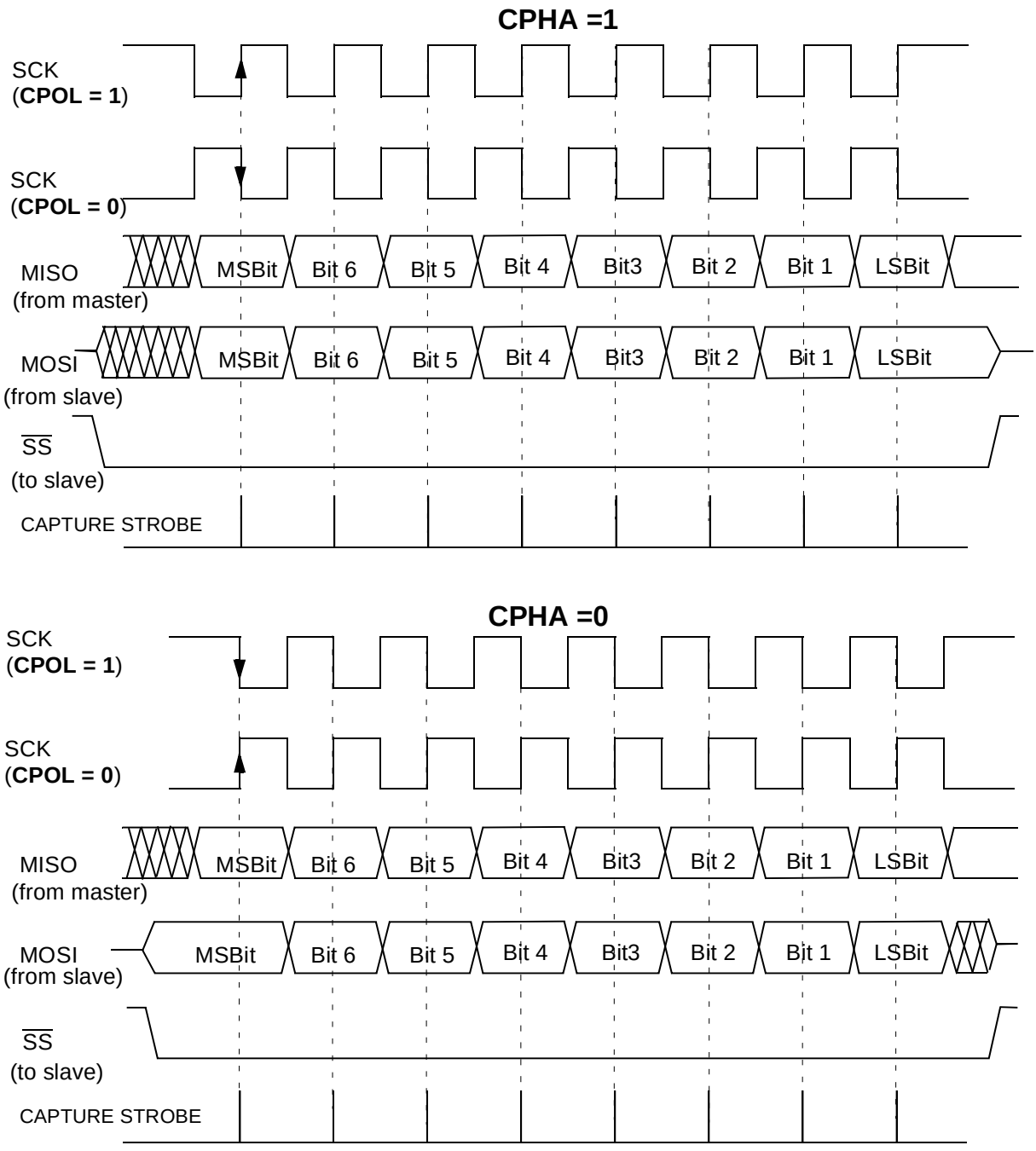
To protect the transmission from a write collision a low value on the $\overline{SS}$ pin of a slave device freezes the data in its SPDR register and does not allow it to be altered. Therefore the $\overline{SS}$ pin must be high to write a new data byte in the SPDR without producing a write collision.

Figure 101. CPHA / $\overline{SS}$ Timing Diagram



SERIAL PERIPHERAL INTERFACE (Cont'd)

Figure 102. Data Clock Timing Diagram



Note: This figure should not be used as a replacement for parametric information. Refer to the SPI Timing table in the Electrical Characteristics Section.

SERIAL PERIPHERAL INTERFACE (Cont'd)

10.6.4.4 Write Collision Error

A write collision occurs when the software tries to write to the SPDR register while a data transfer is taking place with an external device. When this happens, the transfer continues uninterrupted and the software write will be unsuccessful.

Write collisions can occur both in master and slave mode.

Note: a "read collision" will never occur since the received data byte is placed in a buffer in which access is always synchronous with the MCU operation.

In Slave mode

When the CPHA bit is set:

The slave device will receive a clock (SCK) edge prior to the latch of the first data transfer. This first clock edge will freeze the data in the slave device SPDR register and output the MSBit on to the external MISO pin of the slave device.

The $\overline{SS}$ pin low state enables the slave device but the output of the MSBit onto the MISO pin does not take place until the first data transfer clock edge.

When the CPHA bit is reset:

Data is latched on the occurrence of the first clock transition. The slave device does not have any way of knowing when that transition will occur; therefore, the slave device collision occurs when software attempts to write the SPDR register after its $\overline{SS}$ pin has been pulled low.

For this reason, the $\overline{SS}$ pin must be high, between each data byte transfer, to allow the CPU to write in the SPDR register without generating a write collision.

In Master mode

Collision in the master device is defined as a write of the SPDR register while the internal serial clock (SCK) is in the process of transfer.

The $\overline{SS}$ pin signal must be always high on the master device.

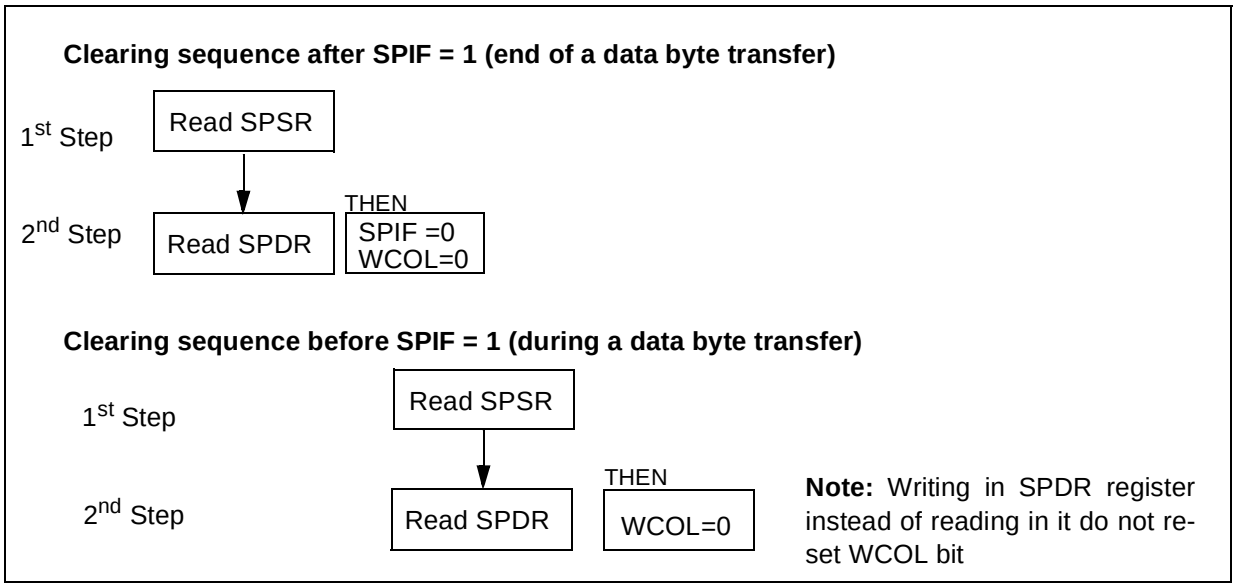
WCOL Bit

The WCOL bit in the SPSR register is set if a write collision occurs.

No SPI interrupt is generated when the WCOL bit is set (the WCOL bit is a status flag only).

Clearing the WCOL bit is done through a software sequence (see Figure 5).

Figure 103. Clearing the WCOL bit (Write Collision Flag) Software Sequence



SERIAL PERIPHERAL INTERFACE (Cont'd)

10.6.4.5 Master Mode Fault

Master mode fault occurs when the master device has its $\overline{SS}$ pin pulled low, then the MODF bit is set.

Master mode fault affects the SPI peripheral in the following ways:

- The MODF bit is set and an SPI interrupt is generated if the SPIE bit is set.
- The SPOE bit is reset. This blocks all output from the device and disables the SPI peripheral.
- The MSTR bit is reset, thus forcing the device into slave mode.

Clearing the MODF bit is done through a software sequence:

1. A read access to the SPSR register while the MODF bit is set.
2. A write to the SPCR register.

Notes: To avoid any multiple slave conflicts in the case of a system comprising several MCUs, the $\overline{SS}$ pin must be pulled high during the clearing sequence of the MODF bit. The SPOE and MSTR bits may be restored to their original state during or after this clearing sequence.

Hardware does not allow the user to set the SPOE and MSTR bits while the MODF bit is set except in the MODF bit clearing sequence.

In a slave device the MODF bit can not be set, but in a multi master configuration the device can be in slave mode with this MODF bit set.

The MODF bit indicates that there might have been a multi-master conflict for system control and allows a proper exit from system operation to a reset or default system state using an interrupt routine.

10.6.4.6 Overrun Condition

An overrun condition occurs, when the master device has sent several data bytes and the slave device has not cleared the SPIF bit issuing from the previous data byte transmitted.

In this case, the receiver buffer contains the byte sent after the SPIF bit was last cleared. A read to the SPDR register returns this byte. All other bytes are lost.

This condition is not detected by the SPI peripheral.

SERIAL PERIPHERAL INTERFACE (Cont'd)

10.6.4.7 Single Master and Multimaster Configurations

There are two types of SPI systems:

– Single Master System

– Multimaster System

Single Master System

A typical single master system may be configured, using an MCU as the master and four MCUs as slaves (see Figure 6).

The master device selects the individual slave devices by using four pins of a parallel port to control the four $\overline{SS}$ pins of the slave devices.

The $\overline{SS}$ pins are pulled high during reset since the master device ports will be forced to be inputs at that time, thus disabling the slave devices.

Note: To prevent a bus conflict on the MISO line the master allows only one slave device during a transmission.

For more security, the slave device may respond to the master with the received data byte. Then the master will receive the previous byte back from the slave device if all MISO and MOSI pins are connected and the slave has not written its SPDR register.

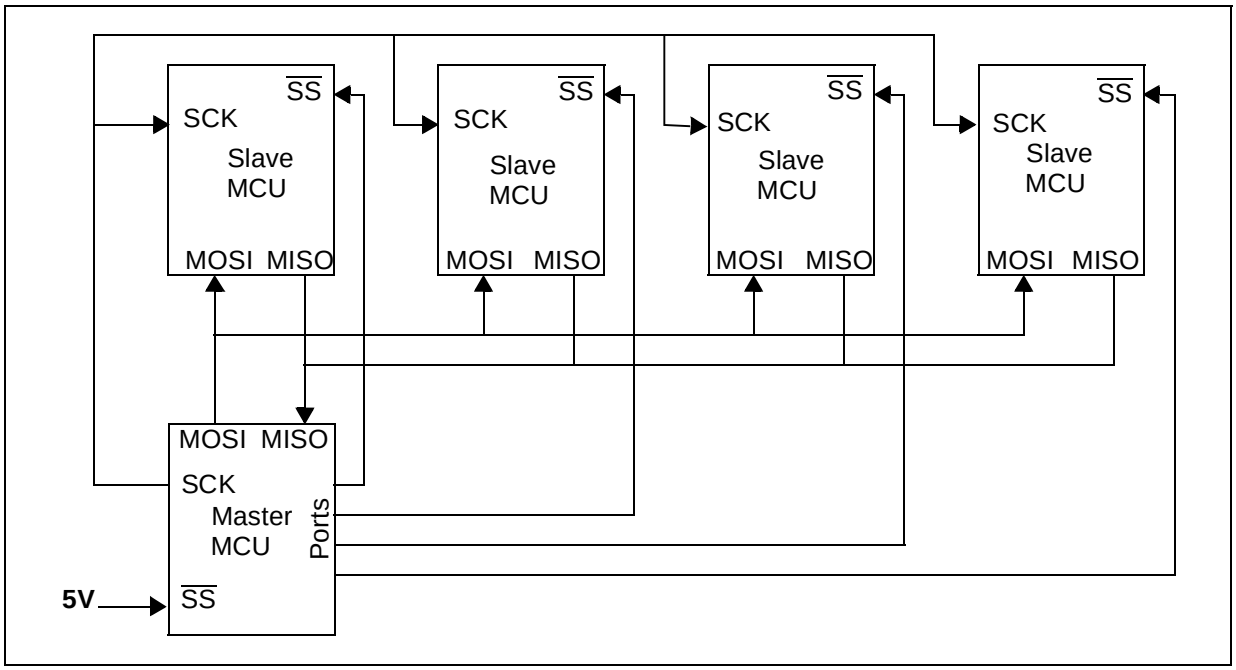
Other transmission security methods can use ports for handshake lines or data bytes with command fields.

Multi-Master System

A multi-master system may also be configured by the user. Transfer of master control could be implemented using a handshake method through the I/O ports or by an exchange of code messages through the serial peripheral interface system.

The multi-master system is principally handled by the MSTR bit in the SPCR register and the MODF bit in the SPSR register.

Figure 104. Single Master Configuration



SERIAL PERIPHERAL INTERFACE (Cont'd)

10.6.5 Interrupt Management

The interrupt of the Serial Peripheral Interface is mapped on one of the eight External Interrupt Channels of the microcontroller (refer to the "Interrupts" chapter).

Each External Interrupt Channel has:

– A trigger control bit in the EITR register (R242 - Page 0),

– A pending bit in the EIPR register (R243 - Page0),

– A mask bit in the EIMR register (R244 - Page 0).

Program the interrupt priority level using the EIPLR register (R245 - Page 0). For a description of these registers refer to the "Interrupts" and "DMA" chapters.

To use the interrupt feature, perform the following sequence:

– Set the priority level of the interrupt channel used for the SPI (EIPRL register)

– Select the interrupt trigger edge as rising edge (set the corresponding bit in the EITR register)

– Set the SPIS bit of the SPCR register to select the peripheral interrupt source

– Set the SPIE bit of the SPCR register to enable the peripheral to perform interrupt requests

– In the EIPR register, reset the pending bit of the interrupt channel used by the SPI interrupt to avoid any spurious interrupt requests being performed when the mask bit is set

– Set the mask bit of the interrupt channel used to enable the MCU to acknowledge the interrupt requests of the peripheral.

Note: In the interrupt routine, reset the related pending bit to avoid the interrupt request that was just acknowledged being proposed again.
Then, after resetting the pending bit and before the IRET instruction, check if the SPIF and MODF interrupt flags in the SPSR register) are reset; otherwise jump to the beginning of the routine. If, on return from an interrupt routine, the pending bit is reset while one of the interrupt flags is set, no interrupt is performed on that channel until the flags are set. A new interrupt request is performed only when a flag is set with the other not set.

10.6.5.1 Register Map

Depending on the device, one or two Serial Peripheral interfaces can be present. The previous table summarizes the position of the registers of the two peripherals in the register map of the microcontroller.

	Address	Page	Name
SPI0	R240 (F0h)	7	DR0
	R241 (F1h)	7	CR0
	R242 (F2h)	7	SR0
	R243 (F3h)	7	PR0
SPI1	R248 (F8h)	7	DR1
	R249 (F9h)	7	CR1
	R250 (FAh)	7	SR1
	R251 (FBh)	7	PR1

SERIAL PERIPHERAL INTERFACE (Cont'd)

10.6.6 Register Description

DATA REGISTER (SPDR)
R240 - Read/Write
Register Page: 7
Reset Value: 0000 0000 (00h)

7							0
D7	D6	D5	D4	D3	D2	D1	D0

The SPDR register is used to transmit and receive data on the serial bus. In the master device only a write to this register will initiate transmission/reception of another byte.

Notes: During the last clock cycle the SPIF bit is set, a copy of the received data byte in the shift register is moved to a buffer. When the user reads the serial peripheral data register, the buffer is actually being read.

Warning: A write to the SPDR register places data directly into the shift register for transmission.

A read to the SPDR register returns the value located in the buffer and not the content of the shift register (see Figure 2).

CONTROL REGISTER (SPCR)
R241 - Read/Write
Register Page: 7
Reset Value: 0000 0000 (00h)

7							0
SPIE	SPOE	SPIS	MSTR	CPOL	CPHA	SPR1	SPR0

Bit 7 = **SPIE** *Serial peripheral interrupt enable.*
This bit is set and cleared by software.
0: Interrupt is inhibited
1: An SPI interrupt is generated whenever either SPIF or MODF are set in the SPSR register while the other flag is 0.

Bit 6 = **SPOE** *Serial peripheral output enable.*
This bit is set and cleared by software. It is also cleared by hardware when, in master mode, $\overline{SS}$=0 (see Section 0.1.4.5 Master Mode Fault).
0: SPI alternate functions disabled (MISO, MOSI and SCK can only work as input)
1: SPI alternate functions enabled (MISO, MOSI and SCK can work as input or output depending on the value of MSTR)

Note: To use the MISO, MOSI and SCK alternate functions (input or output), the corresponding I/O port must be programmed as alternate function output.

Bit 5 = **SPIS** *Interrupt Selection.*
This bit is set and cleared by software.
0: Interrupt source is external interrupt
1: Interrupt source is SPI

Bit 4 = **MSTR** *Master.*
This bit is set and cleared by software. It is also cleared by hardware when, in master mode, $\overline{SS}$=0 (see Section 0.1.4.5 Master Mode Fault).
0: Slave mode is selected
1: Master mode is selected, the function of the SCK pin changes from an input to an output and the functions of the MISO and MOSI pins are reversed.

Bit 3 = **CPOL** *Clock polarity.*
This bit is set and cleared by software. This bit determines the steady state of the serial Clock. The CPOL bit affects both the master and slave modes.
0: The steady state is a low value at the SCK pin.
1: The steady state is a high value at the SCK pin.

Bit 2 = **CPHA** *Clock phase.*
This bit is set and cleared by software.
0: The first clock transition is the first data capture edge.
1: The second clock transition is the first capture edge.

Bit 1:0 = **SPR[1:0]** *Serial peripheral rate.*
These bits are set and cleared by software. They select one of four baud rates to be used as the serial clock when the device is a master.

These 2 bits have no effect in slave mode.

Table 42. Serial Peripheral Baud Rate

INTCLK Clock Divide	SPR1	SPR0
2	0	0
4	0	1
16	1	0
32	1	1

SERIAL PERIPHERAL INTERFACE (Cont'd)

STATUS REGISTER (SPSR)
R242 - Read Only
Register Page: 7
Reset Value: 0000 0000 (00h)

7							0
SPIF	WCOL	-	MODF	-	-	-	-

Bit 7 = **SPIF** *Serial Peripheral data transfer flag.*
This bit is set by hardware when a transfer has been completed. An interrupt is generated if SPIE=1 in the SPCR register. It is cleared by a software sequence (an access to the SPSR register followed by a read or write to the SPDR register).
0: Data transfer is in progress or has been approved by a clearing sequence.
1: Data transfer between the device and an external device has been completed.

Note: While the SPIF bit is set, all writes to the SPDR register are inhibited.

Bit 6 = **WCOL** *Write Collision status.*

This bit is set by hardware when a write to the SPDR register is done during a transmit sequence. It is cleared by a software sequence (see Figure 5).
0: No write collision occurred
1: A write collision has been detected

Bit 5 = Unused.

Bit 4 = **MODF** *Mode Fault flag.*
This bit is set by hardware when the $\overline{SS}$ pin is pulled low in master mode (see Section 0.1.4.5 Master Mode Fault). An SPI interrupt can be generated if SPIE=1 in the SPCR register. This bit is cleared by a software sequence (An access to the SPSR register while MODF=1 followed by a write to the SPCR register).
0: No master mode fault detected
1: A fault in master mode has been detected

Bits 3:0 = Unused.

PRESCALER REGISTER (SPPR)
R243 - Read/Write
Register Page: 7
Reset Value: 0000 0000 (00h)

7							0
0	0	0	DIV2	0	PRS2	PRS1	PRS0

Bits 7:5 = Reserved, forced by hardware to 0**.**

Bit 4 = **DIV2** *Divider enable.*
This bit is set and cleared by software.
0: Divider by 2 enabled.
1: Divider by 2 disabled.

Bit 3 = Reserved. forced by hardware to 0.

Bits 2:0 = **PRS[2:0]** *Prescaler Value.*
These bits are set and cleared by software. The baud rate generator is driven by INTCLK/(n1*n2*n3) where n1= PRS[2:0]+1, n2 is the value defined by the SPR[1:0] bits (refer to Table 1 and Table 2), n3 = 1 if DIV2=1 and n3= 2 if DIV2=0. Refer to Figure 2.

These bits have no effect in slave mode.

Table 43. Prescaler Baud Rate

Prescaler Division Factor	PRS2	PRS1	PRS0
1 (no division)	0	0	0
2	0	0	1
...			
8	1	1	1

10.7 I²C BUS INTERFACE

10.7.1 Introduction

The I²C bus Interface serves as an interface between the microcontroller and the serial I²C bus. It provides both multimaster and slave functions with both 7-bit and 10-bit address modes; it controls all I²C bus-specific sequencing, protocol, arbitration, timing and supports both standard (100KHz) and fast I²C modes (400KHz).
Using DMA, data can be transferred with minimum use of CPU time.

The peripheral uses two external lines to perform the protocols: SDA, SCL.

10.7.2 Main Features

- Parallel-bus/I²C protocol converter
- Multi-master capability
- 7-bit/10-bit Addressing
- Standard I²C mode/Fast I²C mode
- Transmitter/Receiver flag
- End-of-byte transmission flag
- Transfer problem detection
- Interrupt generation on error conditions
- Interrupt generation on transfer request and on data received

I²C Master Features:

- Start bit detection flag
- Clock generation
- I²C bus busy flag
- Arbitration Lost flag
- End of byte transmission flag
- Transmitter/Receiver flag
- Stop/Start generation

I²C Slave Features:

- Stop bit detection
- I²C bus busy flag
- Detection of misplaced start or stop condition
- Programmable I²C Address detection (both 7-bit and 10-bit mode)
- General Call address programmable
- Transfer problem detection
- End of byte transmission flag
- Transmitter/Receiver flag.

Interrupt Features:

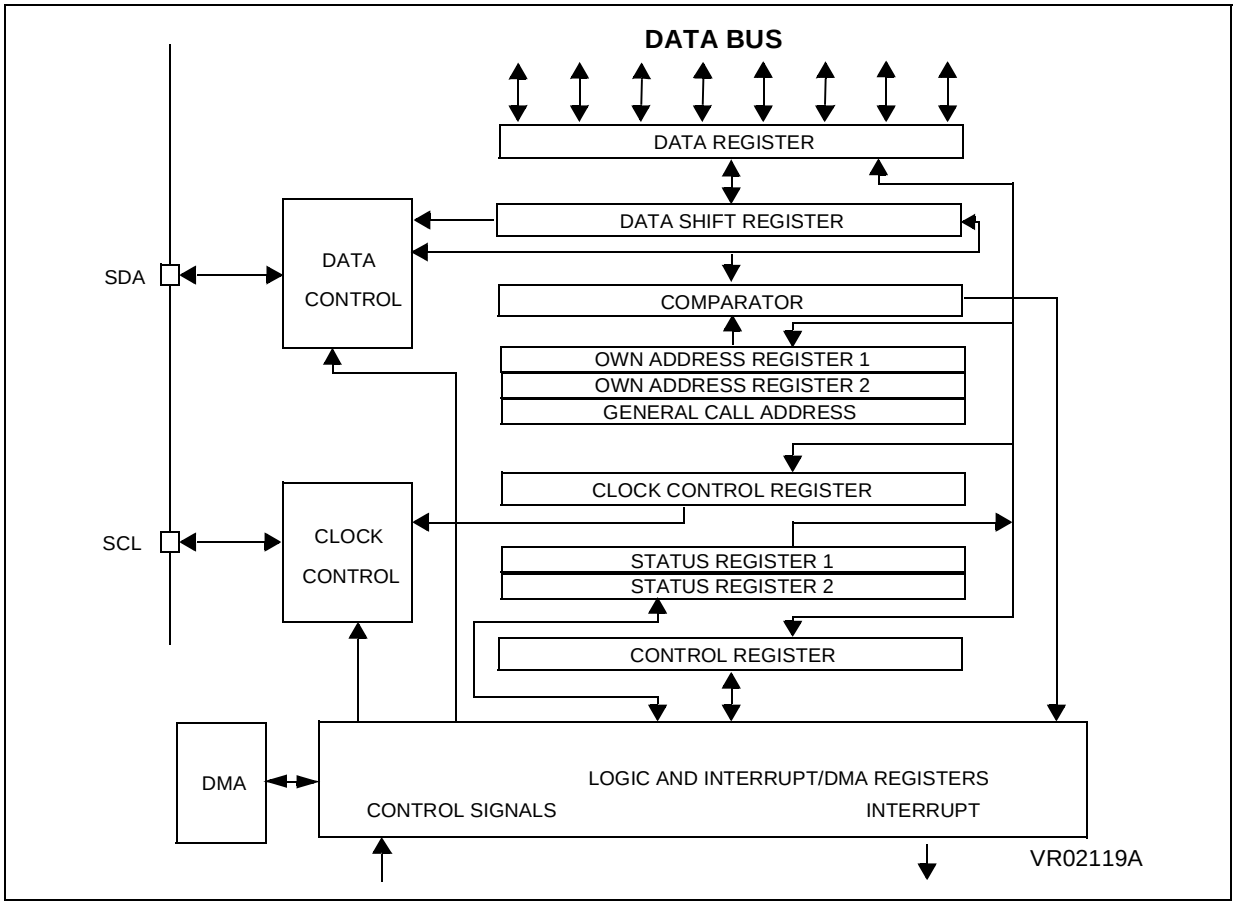
- Interrupt generation on error condition, on transmission request and on data received
- Interrupt address vector for each interrupt source
- Pending bit and mask bit for each interrupt source
- Programmable interrupt priority respects the other peripherals of the microcontroller
- Interrupt address vector programmable

DMA Features:

- DMA both in transmission and in reception with enabling bits
- DMA from/toward both Register File and Memory
- End Of Block interrupt sources with the related pending bits
- Selection between DMA Suspended and DMA Not-Suspended mode if error condition occurs.

I²C BUS INTERFACE (Cont'd)

Figure 105. I²C Interface Block Diagram



10.7.3 Functional Description

Refer to the I2CCR, I2CSR1 and I2CSR2 registers in Section 0.1.7. for the bit definitions.

The I²C interface works as an I/O interface between the ST9 microcontroller and the I²C bus protocol. In addition to receiving and transmitting data, the interface converts data from serial to parallel format and vice versa using an interrupt or polled handshake.

It operates in Multimaster/slave I²C mode. The selection of the operating mode is made by software.

The I²C interface is connected to the I²C bus by a data pin (SDA) and a clock pin (SCL) which must be configured as open drain when the I²C cell is enabled by programming the I/O port bits and the PE bit in the I2CCR register. In this case, the value of the external pull-up resistance used depends on the application.

When the I²C cell is disabled, the SDA and SCL ports revert to being standard I/O port pins.

The I²C interface has sixteen internal registers.

Six of them are used for initialization:

– Own Address Registers I2COAR1, I2COAR2

– General Call Address Register I2CADR

– Clock Control Registers I2CCCR, I2CECCR

– Control register I2CCR

The following four registers are used during data transmission/reception:

– Data Register I2CDR

– Control Register I2CCR

– Status Register 1 I2CSR1

– Status Register 2 I2CSR2

I²C BUS INTERFACE (Cont'd)

The following seven registers are used to handle the interrupt and the DMA features:

– Interrupt Status Register I2CISR

– Interrupt Mask Register I2CIMR

– Interrupt Vector Register I2CIVR

– Receiver DMA Address Pointer Register I2CRDAP

– Receiver DMA Transaction Counter Register I2CRDC

– Transmitter DMA Address Pointer Register I2CTDAP

– Transmitter DMA transaction Counter Register I2CTDC

The interface can decode both addresses:

– Software programmable 7-bit General Call address

– I²C address stored by software in the I2COAR1 register in 7-bit address mode or stored in I2COAR1 and I2COAR2 registers in 10-bit address mode.

After a reset, the interface is disabled.

***IMPORTANT*:**
1. To guarantee correct operation, before enabling the peripheral (while I2CCR.PE=0), configure bit7 and bit6 of the I2COAR2 register according to the internal clock INTCLK (for example 11xxxxxxb in the range 14 - 30 MHz).

2. Bit7 of the I2CCR register must be cleared.

10.7.3.1 Mode Selection

In I²C mode, the interface can operate in the four following modes:

– Master transmitter/receiver

– Slave transmitter/receiver

By default, it operates in slave mode.

This interface automatically switches from slave to master after a start condition is generated on the bus and from master to slave in case of arbitration loss or stop condition generation.

In Master mode, it initiates a data transfer and generates the clock signal. A serial data transfer always begins with a start condition and ends with a stop condition. Both start and stop conditions are generated in master mode by software.

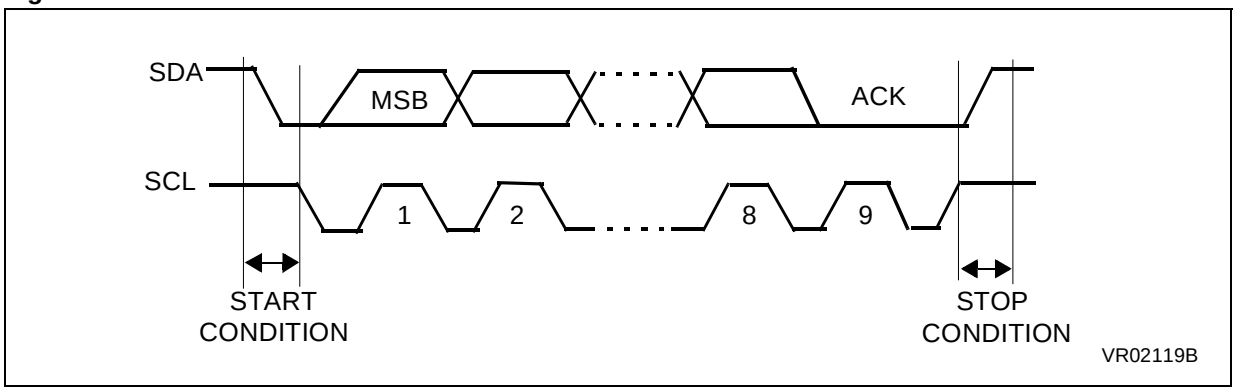
In Slave mode, it is able to recognize its own address (7 or 10-bit), as stored in the I2COAR1 and I2COAR2 registers and (when the I2CCR.ENGC bit is set) the General Call address (stored in I2CADR register). It never recognizes the Start Byte (address byte 01h) whatever its own address is.

Data and addresses are transferred in 8 bits, MSB first. The first byte(s) following the start condition contain the address (one byte in 7-bit mode, two bytes in 10-bit mode). The address is always transmitted in master mode.

A 9th clock pulse follows the 8 clock cycles of a byte transfer, during which the receiver must send an acknowledge bit to the transmitter. Acknowledge is enabled and disabled by software. Refer to Figure 2.

I²C BUS INTERFACE (Cont'd)

Figure 106. I²C BUS Protocol



Any transfer can be done using either the I²C registers directly or via the DMA.

If the transfer is to be done directly by accessing the I2CDR, the interface waits (by holding the SCL line low) for software to write in the Data Register before transmission of a data byte, or to read the Data Register after a data byte is received.

If the transfer is to be done via DMA, the interface sends a request for a DMA transfer. Then it waits for the DMA to complete. The transfer between the interface and the I²C bus will begin on the next rising edge of the SCL clock.

The SCL frequency (F_{scl}) generated in master mode is controlled by a programmable clock divider. The speed of the I²C interface may be selected between Standard (0-100KHz) and Fast (100-400KHz) I²C modes.

10.7.4 I²C State Machine

To enable the interface in I²C mode the I2CCR.PE bit must be set **twice** as the first write only activates the interface (only the PE bit is set); and the bit7 of I2CCR register must be cleared.

The I²C interface always operates in slave mode (the M/SL bit is cleared) except when it initiates a transmission or a receipt sequencing (master mode).

The multimaster function is enabled with an automatic switch from master mode to slave mode when the interface loses the arbitration of the I²C bus.

10.7.4.1 I²C Slave Mode

As soon as a start condition is detected, the address word is received from the SDA line and sent to the shift register; then it is compared with the address of the interface or the General Call address (if selected by software).

Note: In 10-bit addressing mode, the comparison includes the header sequence (11110xx0) and the two most significant bits of the address.

- **Header (10-bit mode) or Address (both 10-bit and 7-bit modes) not matched**: the state machine is reset and waits for another Start condition.
- **Header matched** (10-bit mode only): the interface generates an acknowledge pulse if the ACK bit of the control register (I2CCR) is set.
- **Address matched**: the I2CSR1.ADSL bit is set and an acknowledge bit is sent to the master if the I2CCR.ACK bit is set. An interrupt request occurs if the I2CCR.ITE bit is set. Then the SCL line is held low until the microcontroller reads the I2CSR1 register (see Figure 3 Transfer sequencing EV1).

I²C BUS INTERFACE (Cont'd)

Next, depending on the data direction bit (least significant bit of the address byte), and after the generation of an acknowledge, the slave must go in sending or receiving mode.

In 10-bit mode, after receiving the address sequence the slave is always in receive mode. It will enter transmit mode on receiving a repeated Start condition followed by the header sequence with matching address bits and the least significant bit set (11110xx1).

Slave Receiver

Following the address reception and after I2CSR1 register has been read, the slave receives bytes from the SDA line into the Shift Register and sends them to the I2CDR register. After each byte it generates an acknowledge bit if the I2CCR.ACK bit is set.

When the acknowledge bit is sent, the I2CSR1.BTF flag is set and an interrupt is generated if the I2CCR.ITE bit is set (see Figure 3 Transfer sequencing EV2).
Then the interface waits for a read of the I2CSR1 register followed by a read of the I2CDR register, or waits for the DMA to complete.

Slave Transmitter

Following the address reception and after I2CSR1 register has been read, the slave sends bytes from the I2CDR register to the SDA line via the internal shift register.

When the acknowledge bit is received, the I2CCR.BTF flag is set and an interrupt is generated if the I2CCR.ITE bit is set (see Figure 3 Transfer sequencing EV3).
The slave waits for a read of the I2CSR1 register followed by a write in the I2CDR register or waits for the DMA to complete, **both holding the SCL line low** (except on EV3-1).

Error Cases

– **BERR**: Detection of a Stop or a Start condition during a byte transfer.
The I2CSR2.BERR flag is set and an interrupt is generated if I2CCR.ITE bit is set.
If it is a stop then the state machine is reset.
If it is a start then the state machine is reset and it waits for the new slave address on the bus.

– **AF**: Detection of a no-acknowledge bit.
The I2CSR2.AF flag is set and an interrupt is generated if the I2CCR.ITE bit is set.

Note: In both cases, SCL line is not stretched low; however, the SDA line, due to possible «0» bits transmitted last, can remain low. It is then necessary to release both lines by software.

Other Events

– **ADSL**: Detection of a Start condition after an acknowledge time-slot.
The state machine is reset and starts a new process. The I2CSR1.ADSL flag bit is set and an interrupt is generated if the I2CCR.ITE bit is set.
The SCL line is stretched low.

– **STOPF**: Detection of a Stop condition after an acknowledge time-slot.
The state machine is reset. Then the I2CSR2.STOPF flag is set and an interrupt is generated if the I2CCR.ITE bit is set.

How to release the SDA / SCL lines

Check that the I2CSR1.BUSY bit is reset. Set and subsequently clear the I2CCR.STOP bit while the I2CSR1.BTF bit is set; then the SDA/SCL lines are released immediately after the transfer of the current byte.

This will also reset the state machine; any subsequent STOP bit (EV4) will **not** be detected.

10.7.4.2 I²C Master Mode

To switch from default Slave mode to Master mode a Start condition generation is needed.

Setting the I2CCR.START bit while the I2CSR1.BUSY bit is cleared causes the interface to generate a Start condition.
Once the Start condition is generated, the peripheral is in master mode (I2CSR1.M/SL=1) and I2CSR1.SB (Start bit) flag is set and an interrupt is generated if the I2CCR.ITE bit is set (see Figure 3 Transfer sequencing EV5 event).

The interface waits for a read of the I2CSR1 register followed by a write in the I2CDR register with the Slave address, **holding the SCL line low**.

I²C BUS INTERFACE (Cont'd)

Then the slave address is sent to the SDA line.
In 7-bit addressing mode, one address byte is sent.
In 10-bit addressing mode, sending the first byte including the header sequence causes the I2CSR1.EVF and I2CSR1.ADD10 bits to be set by hardware with interrupt generation if the I2CCR.ITE bit is set.
Then the master waits for a read of the I2CSR1 register followed by a write in the I2CDR register, **holding the SCL line low** (see Figure 3 Transfer sequencing EV9). Then the second address byte is sent by the interface.

After each address byte, an acknowledge clock pulse is sent to the SCL line if the I2CSR1.EVF and

– I2CSR1.ADD10 bit (if first header)

– I2CSR2.ADDTX bit (if address or second header)

are set, and an interrupt is generated if the I2CCR.ITE bit is set.

The peripheral waits for a read of the I2CSR1 register followed by a write into the Control Register (I2CCR) by holding the SCL line low (see Figure 3 Transfer sequencing EV6 event).

If there was no acknowledge (I2CSR2.AF=1), the master must stop or restart the communication (set the I2CCR.START or I2CCR.STOP bits).
If there was an acknowledge, the state machine enters a sending or receiving process according to the data direction bit (least significant bit of the address), the I2CSR1.BTF flag is set and an interrupt is generated if I2CCR.ITE bit is set (see Transfer sequencing EV7, EV8 events).

If the master loses the arbitration of the bus there is no acknowledge, the I2CSR2.AF flag is set and the master must set the START or STOP bit in the control register (I2CCR).The I2CSR2.ARLO flag is set, the I2CSR1.M/SL flag is cleared and the process is reset. An interrupt is generated if I2CCR.ITE is set.

Master Transmitter:
The master waits for the microcontroller to write in the Data Register (I2CDR) or it waits for the DMA to complete **both holding the SCL line low** (see Transfer sequencing EV8).
Then the byte is received into the shift register and sent to the SDA line. When the acknowledge bit is received, the I2CSR1.BTF flag is set and an interrupt is generated if the I2CCR.ITE bit is set or the DMA is requested.

Note: In 10-bit addressing mode, to switch the master to Receiver mode, software must generate a repeated Start condition and resend the header sequence with the least significant bit set (11110xx1).

Master Receiver:
The master receives a byte from the SDA line into the shift register and sends it to the I2CDR register. It generates an acknowledge bit if the I2CCR.ACK bit is set and an interrupt if the I2CCR.ITE bit is set or a DMA is requested (see Transfer sequencing EV7 event).
Then it waits for the microcontroller to read the Data Register (I2CDR) or waits for the DMA to complete **both holding SCL line low**.

Error Cases

- **BERR**: Detection of a Stop or a Start condition during a byte transfer.
  The I2CSR2.BERR flag is set and an interrupt is generated if I2CCR.ITE is set.
- **AF**: Detection of a no acknowledge bit
  The I2CSR2.AF flag is set and an interrupt is generated if I2CCR.ITE is set.
- **ARLO**: Arbitration Lost
  The I2CSR2.ARLO flag is set, the I2CSR1.M/SL flag is cleared and the process is reset. An interrupt is generated if the I2CCR.ITE bit is set.

Note: In all cases, to resume communications, set the I2CCR.START or I2CCR.STOP bits.

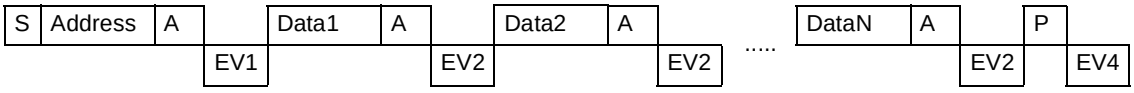
Events generated by the I²C interface

- **STOP condition**
  When the I2CCR.STOP bit is set, a Stop condition is generated **after the transfer** of the current byte, the I2CSR1.M/SL flag is cleared and the state machine is reset. No interrupt is generated in master mode at the detection of the stop condition.
- **START condition**
  When the I2CCR.START bit is set, a start condition is generated as soon as the I²C bus is free. The I2CSR1.SB flag is set and an interrupt is generated if the I2CCR.ITE bit is set.

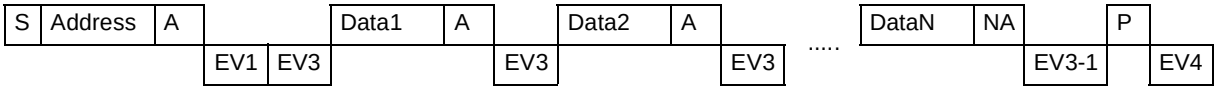
I²C BUS INTERFACE (Cont'd)

Figure 107. Transfer Sequencing

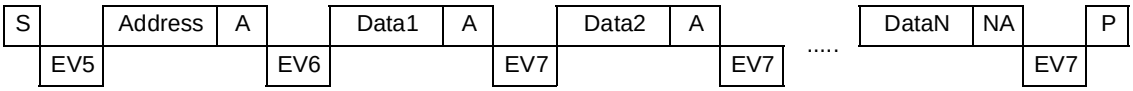
7-bit Slave receiver:



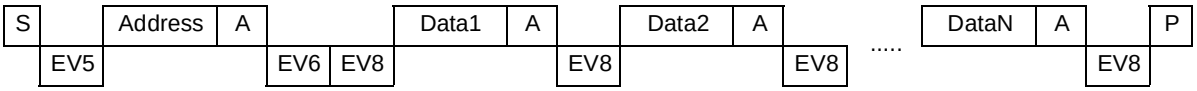
7-bit Slave transmitter:



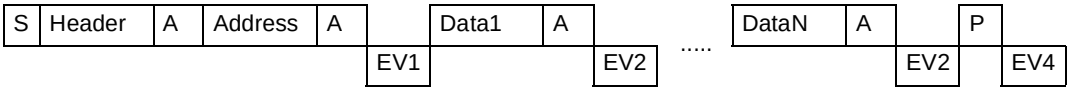
7-bit Master receiver:



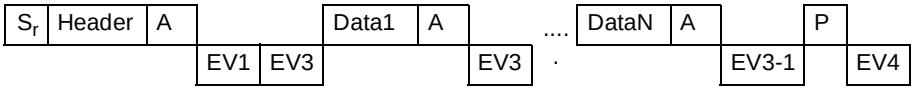
7-bit Master transmitter:



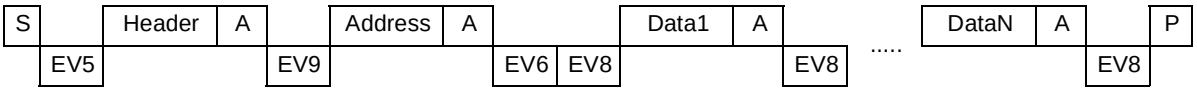
10-bit Slave receiver:



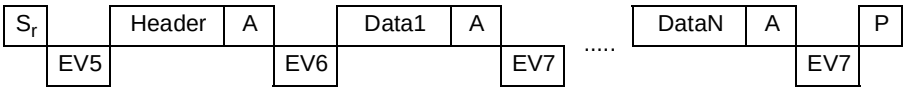
10-bit Slave transmitter:



10-bit Master transmitter



10-bit Master receiver:



Legend:

S=Start, Sr = Repeated Start, P=Stop, A=Acknowledge, NA=Non-acknowledge,
EVx=Event (with interrupt if ITE=1)

EV1: EVF=1, ADSL=1, cleared by reading SR1 register.

EV2: EVF=1, BTF=1, cleared by reading SR1 register followed by reading DR register or when DMA is complete.

EV3: EVF=1, BTF=1, cleared by reading SR1 register followed by writing DR register or when DMA is complete.

EV3-1: EVF=1, AF=1, BTF=1; AF is cleared by reading SR1 register, BTF is cleared by releasing the lines (STOP=1, STOP=0) or writing DR register (for example DR=FFh). **Note**: If lines are released by STOP=1, STOP=0 the subsequent EV4 is not seen.

EV4: EVF=1, STOPF=1, cleared by reading SR2 register.

EV5: EVF=1, SB=1, cleared by reading SR1 register followed by writing DR register.

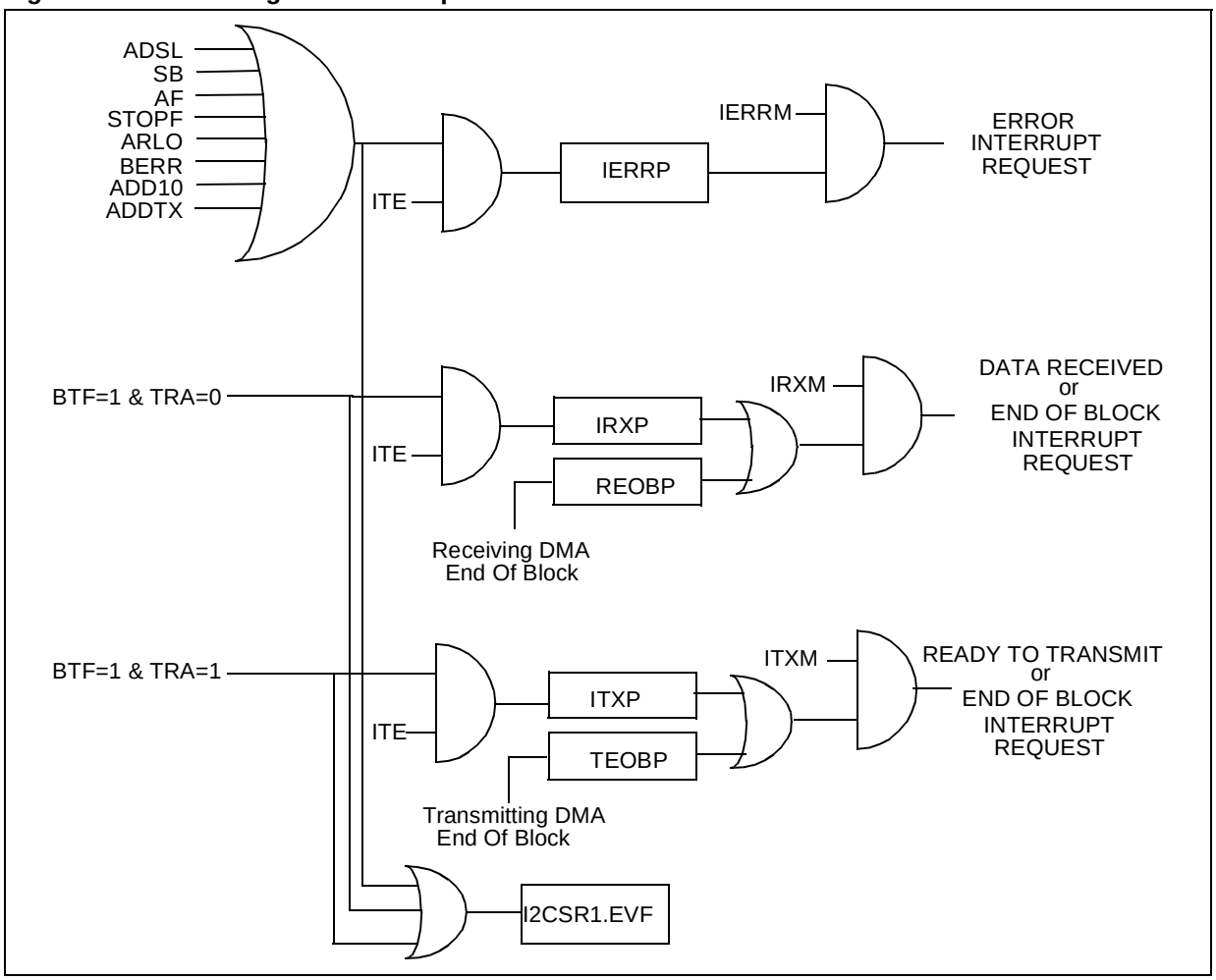
EV6: EVF=1, ADDTX=1, cleared by reading SR1 register followed by writing CR register (for example PE=1).

EV7: EVF=1, BTF=1, cleared by reading SR1 register followed by reading DR register or when DMA is complete.

EV8: EVF=1, BTF=1, cleared by reading SR1 register followed by writing DR register or when DMA is complete.

EV9: EVF=1, ADD10=1, cleared by reading SR1 register followed by writing DR register.

Figure 108. Event Flags and Interrupt Generation



I²C BUS INTERFACE (Cont'd)

10.7.5 Interrupt Features

The I²Cbus interface has three interrupt sources related to "Error Condition", "Peripheral Ready to Transmit" and "Data Received".

The peripheral uses the ST9+ interrupt internal protocol without requiring the use of the external interrupt channel. Dedicated registers of the peripheral should be loaded with appropriate values to set the interrupt vector (see the description of the I2CIVR register), the interrupt mask bits (see the description of the I2CIMR register) and the interrupt priority and pending bits (see the description of the I2CISR register).
The peripheral also has a global interrupt enable (the I2CCR.ITE bit) that must be set to enable the interrupt features. Moreover there is a global interrupt flag (I2CSR1.EVF bit) which is set when one of the interrupt events occurs (except the End Of Block interrupts - see the DMA Features section).

The "Data Received" interrupt source occurs after the acknowledge of a received data byte is performed. It is generated when the I2CSR1.BTF flag is set and the I2CSR1.TRA flag is zero.
If the DMA feature is enabled in receiver mode, this interrupt is not generated and the same interrupt vector is used to send a Receiving End Of Block interrupt (See the DMA feature section).

The "Peripheral Ready To Transmit" interrupt source occurs as soon as a data byte can be transmitted by the peripheral. It is generated when the I2CSR1.BTF and the I2CSR1.TRA flags are set.
If the DMA feature is enabled in transmitter mode, this interrupt is not generated and the same interrupt vector is used to send a Transmitting End Of Block interrupt (See the DMA feature section).

The "Error condition" interrupt source occurs when one of the following condition occurs:

- Address matched in Slave mode while I2CCR.ACK=1
  (I2CSR1.ADSL and I2CSR1.EVF flags = 1)
- Start condition generated
  (I2CSR1.SB and I2CSR1.EVF flags = 1)
- No acknowledge received after byte transmission
  (I2CSR2.AF and I2CSR1.EVF flags = 1)
- Stop detected in Slave mode
  (I2CSR2.STOPF and I2CSR1.EVF flags = 1)
- Arbitration lost in Master mode
  (I2CSR2.ARLO and I2CSR1.EVF flags = 1)
- Bus error, Start or Stop condition detected during data transfer
  (I2CSR2.BERR and I2CSR1.EVF flags = 1)
- Master has sent the header byte
  (I2CSR1.ADD10 and I2CSR1.EVF flags = 1)
- Address byte successfully transmitted in Master mode.
  (I2CSR1.EVF = 1 and I2CSR2.ADDTX=1)

Note: Depending on the value of I2CISR.DMASTOP bit, the pending bit related to the "error condition" interrupt source is able to suspend or not suspend DMA transfers.

Each interrupt source has a dedicated interrupt address pointer vector stored in the I2CIVR register. The five more significant bits of the vector address are programmable by the customer, whereas the three less significant bits are set by hardware depending on the interrupt source:

- 010: error condition detected
- 100: data received
- 110: peripheral ready to transmit

The priority with respect to the other peripherals is programmable by setting the PRL[2:0] bits in the I2CISR register. The lowest interrupt priority is obtained by setting all the bits (this priority level is never acknowledged by the CPU and is equivalent to disabling the interrupts of the peripheral); the highest interrupt priority is programmed by resetting all the bits. See the Interrupt and DMA chapters for more details.

The internal priority of the interrupt sources of the peripheral is fixed by hardware with the following order: "Error Condition" (highest priority), "Data Received", "Peripheral Ready to Transmit".
Note: The DMA has the highest priority over the interrupts; moreover the "Transmitting End Of Block" interrupt has the same priority as the "Peripheral Ready to Transmit" interrupt and the "Receiving End Of Block" interrupt has the same priority as the "Data received" interrupt.

Each of these three interrupt sources has a pending bit (IERRP, IRXP, ITXP) in the I2CISR register that is set by hardware when the corresponding interrupt event occurs. An interrupt request is performed only if the corresponding mask bit is set (IERRM, IRXM, ITXM) in the I2CIMR register and the peripheral has a proper priority level.
The pending bit has to be reset by software.

I²C BUS INTERFACE (Cont'd)

Note: Until the pending bit is reset (while the corresponding mask bit is set), the peripheral processes an interrupt request. So, if at the end of an interrupt routine the pending bit is not reset, another interrupt request is performed.

Note: Before the end of the transmission and reception interrupt routines, the I2CSR1.BTF flag bit should be checked, to acknowledge any interrupt requests that occurred during the interrupt routine and to avoid masking subsequent interrupt requests.

Note: The "Error" event interrupt pending bit (I2CISR.IERRP) is forced high while the error event flags are set (ADD10, ADSL and SB flags of the I2CSR1 register; SCLF, ADDTX, AF, STOPF, ARLO and BERR flags of the I2CSR2 register).

Note: If the I2CISR.DMASTOP bit is reset, then the DMA has the highest priority with respect to the interrupts; if the bit is set (as after the MCU reset) and the "Error event" pending bit is set (I2CISR.IERRP), then the DMA is suspended until the pending bit is reset by software. In the second case, the "Error" interrupt sources have higher priority, followed by DMA, "Data received" and "Receiving End Of Block" interrupts, "Peripheral Ready to Transmit" and "Transmitting End Of Block".
Moreover the Transmitting End Of Block interrupt has the same priority as the "Peripheral Ready to Transmit" interrupt and the Receiving End Of Block interrupt has the same priority as the "Data received" interrupt.

10.7.6 DMA Features

The peripheral can use the ST9+ on-chip Direct Memory Access (DMA) channels to provide high-speed data transaction between the peripheral and contiguous locations of Register File, and Memory. The transactions can occur from and toward the peripheral. The maximum number of transactions that each DMA channel can perform is 222 if the register file is selected or 65536 if memory is selected. The control of the DMA features is performed using registers placed in the peripheral register page (I2CISR, I2CIMR, I2CRDAP, I2CRDC, I2CTDAP, I2CTDC).

Each DMA transfer consists of three operations:

– A load from/to the peripheral data register (I2CDR) to/from a location of Register File/Memory addressed through the DMA Address Register (or Register pair)
– A post-increment of the DMA Address Register (or Register pair)
– A post-decrement of the DMA transaction counter, which contains the number of transactions that have still to be performed.

Depending on the value of the DDCISR.DMASTOP bit the DMA feature can be suspended or not (both in transmission and in reception) until the pending bit related to the "Error event" interrupt request is set.

The priority level of the DMA features of the I²C interface with respect to the other peripherals and the CPU is the same as programmed in the I2CISR register for the interrupt sources. In the internal priority level order of the peripheral, if DDCISR.DMASTOP=0, DMA has a higher priority with respect to the interrupt sources. Otherwise (if DDCISR.DMASTOP=1), the DMA has a priority lower than "error" event interrupt sources but greater than reception and transmission interrupt sources.
Refer to the Interrupt and DMA chapters for details on the priority levels.

The DMA features are enabled by setting the corresponding enabling bits (RXDM, TXDM) in the I2CIMR register. It is possible to select also the direction of the DMA transactions.

Once the DMA transfer is completed (the transaction counter reaches 0 value), an interrupt request to the CPU is generated. This kind of interrupt is called "End Of Block". The peripheral sends two different "End Of Block" interrupts depending on the direction of the DMA (Receiving End Of Block - Transmitting End Of Block). These interrupt sources have dedicated interrupt pending bits in the I2CIMR register (REOBP, TEOBP) and they are mapped on the same interrupt vectors as respectively "Data Received" and "Peripheral Ready to Transmit" interrupt sources. The same correspondence exists about the internal priority between interrupts.

Note: The I2CCR.ITE bit has no effect on the End Of Block interrupts.
Moreover, the I2CSR1.EVF flag is not set by the End Of Block interrupts.

I²C BUS INTERFACE (Cont'd)

10.7.6.1 DMA between Peripheral and Register File

If the DMA transaction is made between the peripheral and the Register File, one register is required to hold the DMA Address and one to hold the DMA transaction counter.
These two registers must be located in the Register File:

– the DMA Address Register in the even addressed register,

– the DMA Transaction Counter in the following register (odd address).

They are pointed to by the DMA Transaction Counter Pointer Register (I2CRDC register in receiving, I2CTDC register in transmitting) located in the peripheral register page.

In order to select the DMA transaction with the Register File, the control bit I2CRDC.RF/MEM in receiving mode or I2CTDC.RF/MEM in transmitting mode must be set.

The transaction Counter Register must be initialized with the number of DMA transfers to perform and will be decremented after each transaction.
The DMA Address Register must be initialized with the starting address of the DMA table in the Register File, and it is increased after each transaction. These two registers must be located between addresses 00h and DFh of the Register File.

When the DMA occurs between Peripheral and Register File, the I2CTDAP register (in transmission) and the I2CRDAP one (in reception) are not used.

10.7.6.2 DMA between Peripheral and Memory Space

If the DMA transaction is made between the peripheral and Memory, a register pair is required to hold the DMA Address and another register pair to hold the DMA Transaction counter. These two pairs of registers must be located in the Register File. The DMA Address pair is pointed to by the DMA Address Pointer Register (I2CRDAP register in reception, I2CTDAP register in transmission) located in the peripheral register page; the DMA Transaction Counter pair is pointed to by the DMA Transaction Counter Pointer Register (I2CRDC register in reception, I2CTDC register in transmission) located in the peripheral register page.

In order to select the DMA transaction with the Memory Space, the control bit I2CRDC.RF/MEM in receiving mode or I2CTDC.RF/MEM in transmitting mode must be reset.

The Transaction Counter registers pair must be initialized with the number of DMA transfers to perform and will be decremented after each transaction. The DMA Address register pair must be initialized with the starting address of the DMA table in the Memory Space, and it is increased after each transaction. These two register pairs must be located between addresses 00h and DFh of the Register File.

10.7.6.3 DMA in Master Receive

To correctly manage the reception of the last byte when the DMA in Master Receive mode is used, the following sequence of operations must be performed:

1. The number of data bytes to be received must be set to the effective number of bytes minus one byte.
2. When the Receiving End Of Block condition occurs, the I2CCR.STOP bit must be set and the I2CCR.ACK bit must be reset.

The last byte of the reception sequence can be received either using interrupts/polling or using DMA. If the user wants to receive the last byte using DMA, the number of bytes to be received must be set to 1, and the DMA in reception must be reenabled (IMR.RXDM bit set) to receive the last byte. Moreover the Receiving End Of Block interrupt service routine must be designed to recognize and manage the two different End Of Block situations (after the first sequence of data bytes and after the last data byte).

I²C BUS INTERFACE (Cont'd)

10.7.7 Register Description

***IMPORTANT*:**
1. To guarantee correct operation, before enabling the peripheral (while I2CCR.PE=0), configure bit7 and bit6 of the I2COAR2 register according to the internal clock INTCLK (for example 11xxxxxxb in the range 14 - 30 MHz).

2. Bit7 of the I2CCR register must be cleared.

I²C CONTROL REGISTER (I2CCR)
R240 - Read / Write
Register Page: 20
Reset Value: 0000 0000 (00h)

7							0
0	0	PE	ENGC	START	ACK	STOP	ITE

Bit 7:6 = **Reserved**
Must be cleared

Bit 5 = **PE** *Peripheral Enable.*
This bit is set and cleared by software.
0: Peripheral disabled (reset value)
1: Master/Slave capability
Notes:
– When I2CCR.PE=0, all the bits of the I2CCR register and the I2CSR1-I2CSR2 registers except the STOP bit are reset. All outputs will be released while I2CCR.PE=0
– When I2CCR.PE=1, the corresponding I/O pins are selected by hardware as alternate functions (open drain).
– To enable the I²C interface, write the I2CCR register **TWICE** with I2CCR.PE=1 as the first write only activates the interface (only I2CCR.PE is set).
– When PE=1, the FREQ[2:0] and EN10BIT bits in the I2COAR2 and I2CADR registers cannot be written. The value of these bits can be changed only when PE=0.

Bit 4 = **ENGC** *General Call address enable.*
Setting this bit the peripheral works as a slave and the value stored in the I2CADR register is recognized as device address.

This bit is set and cleared by software. It is also cleared by hardware when the interface is disabled (I2CCR.PE=0).
0: The address stored in the I2CADR register is ignored (reset value)
1: The General Call address stored in the I2CADR register will be acknowledged
Note: The correct value (usually 00h) must be written in the I2CADR register before enabling the General Call feature.

Bit 3 = **START** *Generation of a Start condition.*
This bit is set and cleared by software. It is also cleared by hardware when the interface is disabled (I2CCR.PE=0) or when the Start condition is sent (with interrupt generation if ITE=1).
– In master mode:
0: No start generation
1: Repeated start generation
– In slave mode:
0: No start generation (reset value)
1: Start generation when the bus is free

Bit 2 = **ACK** *Acknowledge enable.*
This bit is set and cleared by software. It is also cleared by hardware when the interface is disabled (I2CCR.PE=0).
0: No acknowledge returned (reset value)
1: Acknowledge returned after an address byte or a data byte is received

Bit 1 = **STOP** *Generation of a Stop condition.*
This bit is set and cleared by software. It is also cleared by hardware in master mode. It is not cleared when the interface is disabled (I2CCR.PE=0). In slave mode, this bit must be set only when I2CSR1.BTF=1.

– In master mode:
0: No stop generation
1: Stop generation after the current byte transfer or after the current Start condition is sent. The STOP bit is cleared by hardware when the Stop condition is sent.
– In slave mode:
0: No stop generation (reset value)
1: Release SCL and SDA lines after the current byte transfer (I2CSR1.BTF=1). In this mode the STOP bit has to be cleared by software.

I²C BUS INTERFACE (Cont'd)
Bit 0 = **ITE** *Interrupt Enable.*
The ITE bit enables the generation of interrupts. This bit is set and cleared by software and cleared by hardware when the interface is disabled (I2CCR.PE=0).
0: Interrupts disabled (reset value)
1: Interrupts enabled after any of the following conditions:

- Byte received or to be transmitted (I2CSR1.BTF and I2CSR1.EVF flags = 1)
- Address matched in Slave mode while I2CCR.ACK=1 (I2CSR1.ADSL and I2CSR1.EVF flags = 1)
- Start condition generated (I2CSR1.SB and I2CSR1.EVF flags = 1)
- No acknowledge received after byte transmission (I2CSR2.AF and I2CSR1.EVF flags = 1)
- Stop detected in Slave mode (I2CSR2.STOPF and I2CSR1.EVF flags = 1)
- Arbitration lost in Master mode (I2CSR2.ARLO and I2CSR1.EVF flags = 1)
- Bus error, Start or Stop condition detected during data transfer (I2CSR2.BERR and I2CSR1.EVF flags = 1)
- Master has sent header byte (I2CSR1.ADD10 and I2CSR1.EVF flags = 1)
- Address byte successfully transmitted in Master mode. (I2CSR1.EVF = 1 and I2CSR2.ADDTX = 1)

SCL is held low when the ADDTX flag of the I2CSR2 register or the ADD10, SB, BTF or ADSL flags of I2CSR1 register are set (See Figure 3) or when the DMA is not complete.
The transfer is suspended in all cases except when the BTF bit is set and the DMA is enabled. In this case the event routine must suspend the DMA transfer if it is required.

I²C STATUS REGISTER 1 (I2CSR1)
R241 - Read Only
Register Page: 20
Reset Value: 0000 0000 (00h)

7							0
EVF	ADD10	TRA	BUSY	BTF	ADSL	M/SL	SB

Note: Some bits of this register are reset by a read operation of the register. Care must be taken when using instructions that work on single bit. Some of them perform a read of all the bits of the register before modifying or testing the wanted bit. So other bits of the register could be affected by the operation.
In the same way, the test/compare operations perform a read operation.
Moreover, if some interrupt events occur while the register is read, the corresponding flags are set, and correctly read, but if the read operation resets the flags, no interrupt request occurs.

Bit 7 = **EVF** *Event Flag.*
This bit is set by hardware as soon as an event (listed below or described in Figure 3) occurs. It is cleared by software when all event conditions that set the flag are cleared. It is also cleared by hardware when the interface is disabled (I2CCR.PE=0).

0: No event
1: One of the following events has occurred:

- Byte received or to be transmitted (I2CSR1.BTF and I2CSR1.EVF flags = 1)
- Address matched in Slave mode while I2CCR.ACK=1 (I2CSR1.ADSL and I2CSR1.EVF flags = 1)
- Start condition generated (I2CSR1.SB and I2CSR1.EVF flags = 1)
- No acknowledge received after byte transmission (I2CSR2.AF and I2CSR1.EVF flags = 1)
- Stop detected in Slave mode (I2CSR2.STOPF and I2CSR1.EVF flags = 1)
- Arbitration lost in Master mode (I2CSR2.ARLO and I2CSR1.EVF flags = 1)
- Bus error, Start or Stop condition detected during data transfer (I2CSR2.BERR and I2CSR1.EVF flags = 1)
- Master has sent header byte (I2CSR1.ADD10 and I2CSR1.EVF flags = 1)

I²C BUS INTERFACE (Cont'd)

– Address byte successfully transmitted in Master mode.
(I2CSR1.EVF = 1 and I2CSR2.ADDTX=1)

Bit 6 = **ADD10** *10-bit addressing in Master mode.*
This bit is set when the master has sent the first byte in 10-bit address mode. An interrupt is generated if ITE=1.
It is cleared by software reading I2CSR1 register followed by a write in the I2CDR register of the second address byte. It is also cleared by hardware when peripheral is disabled (I2CCR.PE=0) or when the STOPF bit is set.

0: No ADD10 event occurred.
1: Master has sent first address byte (header).

Bit 5 = **TRA** *Transmitter/ Receiver.*
When BTF flag of this register is set and also TRA=1, then a data byte has to be transmitted. It is cleared automatically when BTF is cleared. It is also cleared by hardware after the STOPF flag of I2CSR2 register is set, loss of bus arbitration (ARLO flag of I2CSR2 register is set) or when the interface is disabled (I2CCR.PE=0).
0: A data byte is received (if I2CSR1.BTF=1)
1: A data byte can be transmitted (if I2CSR1.BTF=1)

Bit 4 = **BUSY** *Bus Busy*.
It indicates a communication in progress on the bus. The detection of the communications is always active (even if the peripheral is disabled).
This bit is set by hardware on detection of a Start condition and cleared by hardware on detection of a Stop condition. This information is still updated when the interface is disabled (I2CCR.PE=0).
0: No communication on the bus
1: Communication ongoing on the bus

Bit 3 = **BTF** *Byte Transfer Finished.*
This bit is set by hardware as soon as a byte is correctly received or before the transmission of a data byte with interrupt generation if ITE=1. It is cleared by software reading I2CSR1 register followed by a read or write of I2CDR register or when DMA is complete. It is also cleared by hardware when the interface is disabled (I2CCR.PE=0).

– Following a byte transmission, this bit is set after reception of the acknowledge clock pulse. BTF is cleared by reading I2CSR1 register followed by writing the next byte in I2CDR register or when DMA is complete.
– Following a byte reception, this bit is set after transmission of the acknowledge clock pulse if ACK=1. BTF is cleared by reading I2CSR1 register followed by reading the byte from I2CDR register or when DMA is complete.

The SCL line is held low while I2CSR1.BTF=1.

0: Byte transfer not done
1: Byte transfer succeeded

Bit 2 = **ADSL** *Address matched (Slave mode).*
This bit is set by hardware if the received slave address matches the I2COAR1/I2COAR2 register content or a General Call address. An interrupt is generated if ITE=1. It is cleared by software reading I2CSR1 register or by hardware when the interface is disabled (I2CCR.PE=0). The SCL line is held low while ADSL=1.
0: Address mismatched or not received
1: Received address matched

Bit 1 = **M/SL** *Master/Slave.*
This bit is set by hardware as soon as the interface is in Master mode (Start condition generated on the lines after the I2CCR.START bit is set). It is cleared by hardware after detecting a Stop condition on the bus or a loss of arbitration (ARLO=1). It is also cleared when the interface is disabled (I2CCR.PE=0).
0: Slave mode
1: Master mode

Bit 0 = **SB** *Start Bit (Master mode).*
This bit is set by hardware as soon as the Start condition is generated (following a write of START=1 if the bus is free). An interrupt is generated if ITE=1. It is cleared by software reading I2CSR1 register followed by writing the address byte in I2CDR register. It is also cleared by hardware when the interface is disabled (I2CCR.PE=0).
The SCL line is held low while SB=1.
0: No Start condition
1: Start condition generated

I²C BUS INTERFACE (Cont'd)

I²C STATUS REGISTER 2 (I2CSR2)

R242 - Read Only
Register Page: 20
Reset Value: 0000 0000 (00h)

7							0
0	0	ADDTX	AF	STOPF	ARLO	BERR	GCAL

Note: Some bits of this register are reset by a read operation of the register. Care must be taken when using instructions that work on single bit. Some of them perform a read of all the bits of the register before modifying or testing the wanted bit. So other bits of the register could be affected by the operation.
In the same way, the test/compare operations perform a read operation.
Moreover, if some interrupt events occur while the register is read, the corresponding flags are set, and correctly read, but if the read operation resets the flags, no interrupt request occurs.

Bits 7:6 = Reserved. Forced to 0 by hardware.

Bit 5 = **ADDTX** *Address or 2nd header transmitted in Master mode.*
This bit is set by hardware when the peripheral, enabled in Master mode, has received the acknowledge relative to:

– Address byte in 7-bit mode

– Address or 2nd header byte in 10-bit mode.

0: No address or 2nd header byte transmitted
1: Address or 2nd header byte transmitted.

Bit 4 = **AF** *Acknowledge Failure*.
This bit is set by hardware when no acknowledge is returned. An interrupt is generated if ITE=1.
It is cleared by software reading I2CSR2 register **after the falling edge of the acknowledge SCL pulse**, or by hardware when the interface is disabled (I2CCR.PE=0).
The SCL line is not held low while AF=1.

0: No acknowledge failure detected
1: A data or address byte was not acknowledged

Bit 3 = **STOPF** *Stop Detection (Slave mode).*
This bit is set by hardware when a Stop condition is detected on the bus after an acknowledge. An interrupt is generated if ITE=1.
It is cleared by software reading I2CSR2 register or by hardware when the interface is disabled (I2CCR.PE=0).
The SCL line is not held low while STOPF=1.

0: No Stop condition detected
1: Stop condition detected (**while slave receiver**)

Bit 2 = **ARLO** *Arbitration Lost*.
This bit is set by hardware when the interface (in master mode) loses the arbitration of the bus to another master. An interrupt is generated if ITE=1.
It is cleared by software reading I2CSR2 register or by hardware when the interface is disabled (I2CCR.PE=0).
After an ARLO event the interface switches back automatically to Slave mode (M/SL=0).
The SCL line is not held low while ARLO=1.

0: No arbitration lost detected
1: Arbitration lost detected

Bit 1 = **BERR** *Bus Error.*
This bit is set by hardware when the interface detects a Start or Stop condition during a byte transfer. An interrupt is generated if ITE=1.
It is cleared by software reading I2CSR2 register or by hardware when the interface is disabled (I2CCR.PE=0).
The SCL line is not held low while BERR=1.
Note: If a misplaced start condition is detected, also the **ARLO** flag is set; moreover, if a misplaced stop condition is placed on the acknowledge SCL pulse, also the **AF** flag is set.

0: No Start or Stop condition detected during byte transfer
1: Start or Stop condition detected during byte transfer

Bit 0 = **GCAL** *General Call address matched.*
This bit is set by hardware after an address matches with the value stored in the I2CADR register while ENGC=1. In the I2CADR the General Call address must be placed before enabling the peripheral.
It is cleared by hardware after the detection of a Stop condition, or when the peripheral is disabled (I2CCR.PE=0).
0: No match
1: General Call address matched.

I²C BUS INTERFACE (Cont'd)

I²C CLOCK CONTROL REGISTER (I2CCCR)

R243 - Read / Write
Register Page: 20
Reset Value: 0000 0000 (00h)

7							0
FM/SM	CC6	CC5	CC4	CC3	CC2	CC1	CC0

Bit 7 = **FM/SM** *Fast/Standard I²C mode.*
This bit is used to select between fast and standard mode. See the description of the following bits. It is set and cleared by software. It is not cleared when the peripheral is disabled (I2CCR.PE=0)

Bits 6:0 = **CC[6:0]** *9-bit divider programming*
Implementation of a programmable clock divider. These bits and the CC[8:7] bits of the I2CECCR register select the speed of the bus (F_{SCL}) depending on the I²C mode.
They are not cleared when the interface is disabled (I2CCR.PE=0).

– Standard mode (FM/SM=0): F_{SCL} <= 100kHz

F_{SCL} = INTCLK/(2x([CC8..CC0]+2))

– Fast mode (FM/SM=1): F_{SCL} > 100kHz

F_{SCL} = INTCLK/(3x([CC8..CC0]+2))

Note: The programmed frequency is available with no load on SCL and SDA pins.

I²C OWN ADDRESS REGISTER 1 (I2COAR1)

R244 - Read / Write
Register Page: 20
Reset Value: 0000 0000 (00h)

7							0
ADD7	ADD6	ADD5	ADD4	ADD3	ADD2	ADD1	ADD0

7-bit Addressing Mode

Bits 7:1 = **ADD[7:1]** *Interface address*.
These bits define the I²C bus address of the interface.
They are not cleared when the interface is disabled (I2CCR.PE=0).

Bit 0 = **ADD0** *Address direction bit.*
This bit is don't care; the interface acknowledges either 0 or 1.
It is not cleared when the interface is disabled (I2CCR.PE=0).

Note: Address 01h is always ignored.

10-bit Addressing Mode

Bits 7:0 = **ADD[7:0]** *Interface address*.
These are the least significant bits of the I²Cbus address of the interface.
They are not cleared when the interface is disabled (I2CCR.PE=0).

I²C BUS INTERFACE (Cont'd)

I²C OWN ADDRESS REGISTER 2 (I2COAR2)

R245 - Read / Write
Register Page: 20
Reset Value: 0000 0000 (00h)

7							0
FREQ1	FREQ0	EN10BIT	FREQ2	0	ADD9	ADD8	0

Bits 7:6,4 = **FREQ[2:0]** *Frequency bits.*

IMPORTANT: To guarantee correct operation, set these bits before enabling the interface (while I2CCR.PE=0).

These bits can be set only when the interface is disabled (I2CCR.PE=0). To configure the interface to I²C specified delays, select the value corresponding to the microcontroller internal frequency INTCLK.

INTCLK Range (MHz)	FREQ2	FREQ1	FREQ0
2.5 - 6	0	0	0
6- 10	0	0	1
10- 14	0	1	0
14 - 30	0	1	1
30 - 50	1	0	0

Note: If an incorrect value, with respect to the MCU internal frequency, is written in these bits, the timings of the peripheral will not meet the I²C bus standard requirements.

Note: The FREQ[2:0] = 101, 110, 111 configurations must not be used.

Bit 5 = **EN10BIT** *Enable 10-bit I²Cbus mode.*
When this bit is set, the 10-bit I²Cbus mode is enabled.
This bit can be written only when the peripheral is disabled (I2CCR.PE=0).
0: 7-bit mode selected
1: 10-bit mode selected

Bits 4:3 = Reserved.

Bits 2:1 = **ADD[9:8]** *Interface address.*
These are the most significant bits of the I²Cbus address of the interface (10-bit mode only). They are not cleared when the interface is disabled (I2CCR.PE=0).

Bit 0 = Reserved.

I²C DATA REGISTER (I2CDR)

R246 - Read / Write
Register Page: 20
Reset Value: 0000 0000 (00h)

7							**0**
DR7	DR6	DR5	DR4	DR3	DR2	DR1	DR0

Bits 7:0 = **DR[7:0]** *I2C Data.*

– In transmitter mode:
I2CDR contains the next byte of data to be transmitted. The byte transmission begins after the microcontroller has written in I2CDR or on the next rising edge of the clock if DMA is complete.

– In receiver mode:
I2CDR contains the last byte of data received. The next byte receipt begins after the I2CDR read by the microcontroller or on the next rising edge of the clock if DMA is complete.

GENERAL CALL ADDRESS (I2CADR)

R247 - Read / Write
Register Page: 20
Reset Value: 1010 0000 (A0h)

7							0
ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	ADR0

Bits 7:0 = **ADR[7:0]** *Interface address.*
These bits define the I²Cbus General Call address of the interface. It must be written with the correct value depending on the use of the peripheral.If the peripheral is used in I²C bus mode, the 00h value must be loaded as General Call address.
The customer could load the register with other values.
The bits can be written only when the peripheral is disabled (I2CCR.PE=0)
The ADR0 bit is don't care; the interface acknowledges either 0 or 1.
Note: Address 01h is always ignored.

I²C BUS INTERFACE (Cont'd)

INTERRUPT STATUS REGISTER (I2CISR)
R248 - Read / Write
Register Page: 20
Reset Value: 1xxx xxxx (xxh)

7							0
DMASTOP	PRL2	PRL1	PRL0	0	IERRP	IRXP	ITXP

Bit 7 = **DMASTOP** *DMA suspended mode*.
This bit selects between DMA suspended mode and DMA not suspended mode.
In DMA Suspended mode, if the error interrupt pending bit (I2CISR.IERRP) is set, no DMA request is performed. DMA requests are performed only when IERRP=0. Moreover the "Error Condition" interrupt source has a higher priority than the DMA.
In DMA Not-Suspended mode, the status of IERRP bit has no effect on DMA requests. Moreover the DMA has higher priority with respect to other interrupt sources.
0: DMA Suspended mode
1: DMA Not-Suspended mode

Bits 6:4 = **PRL[2:0]** *Interrupt/DMA Priority Bits*.
The priority is encoded with these three bits. The value of "0" has the highest priority, the value "7" has no priority. After the setting of this priority level, the priorities between the different Interrupt/ DMA sources is hardware defined according with the following scheme:

– Error condition Interrupt (If DMASTOP=1) (Highest priority)
– Receiver DMA request
– Transmitter DMA request
– Error Condition Interrupt (If DMASTOP=0
– Data Received/Receiver End Of Block
– Peripheral Ready To Transmit/Transmitter End Of Block (Lowest priority)

Bit 3 = Reserved.
Must be cleared.

Bit 2 = **IERRP** *Error Condition pending bit*
0: No error
1: Error event detected (if ITE=1)

Note: Depending on the status of the I2CISR.DMASTOP bit, this flag can suspend or not suspend the DMA requests.

Note: The Interrupt pending bits can be reset by writing a "0" but is not possible to write a "1". It is mandatory to clear the interrupt source by writing a "0" in the pending bit when executing the interrupt service routine. When serving an interrupt routine, the user should reset ONLY the pending bit related to the served interrupt routine (and not reset the other pending bits).
To detect the specific error condition that occurred, the flag bits of the I2CSR1 and I2CSR2 register should be checked.

Note: The IERRP pending bit is forced high while the error event flags are set (ADSL and SB flags in the I2CSR1 register, SCLF, ADDTX, AF, STOPF, ARLO and BERR flags in the I2CSR2 register). If at least one flag is set, it is not possible to reset the IERRP bit.

Bit 1 = **IRXP** *Data Received pending bit*
0: No data received
1: data received (if ITE=1).

Bit 0 = **ITXP** *Peripheral Ready To Transmit pending bit*
0: Peripheral not ready to transmit
1: Peripheral ready to transmit a data byte (if ITE=1).

I²C BUS INTERFACE (Cont'd)

INTERRUPT VECTOR REGISTER (I2CIVR)
R249 - Read / Write
Register Page: 20
Reset Value: Undefined

7							0
V7	V6	V5	V4	V3	EV2	EV1	0

Bits 7:3 = **V[7:3]** *Interrupt Vector Base Address*.
User programmable interrupt vector bits. These are the five more significant bits of the interrupt vector base address. They must be set before enabling the interrupt features.

Bits 2:1 = **EV[2:1]** *Encoded Interrupt Source*.
These Read-Only bits are set by hardware according to the interrupt source:

– 01: error condition detected

– 10: data received

– 11: peripheral ready to transmit

Bit 0 = Reserved.
Forced by hardware to 0.

RECEIVER DMA SOURCE ADDRESS POINTER REGISTER (I2CRDAP)
R250 - Read / Write
Register Page: 20
Reset Value: Undefined

7							0
RA7	RA6	RA5	RA4	RA3	RA2	RA1	RPS

Bits 7:1 = **RA[7:1]** *Receiver DMA Address Pointer*.
I2CRDAP contains the address of the pointer (in the Register File) of the Receiver DMA data source when the DMA is selected between the peripheral and the Memory Space. Otherwise, (DMA between peripheral and Register file), this register has no meaning.
See Section 0.1.6.1 for more details on the use of this register.

Bit 0 = **RPS** *Receiver DMA Memory Pointer Selector.*
If memory has been selected for DMA transfer (DDCRDC.RF/MEM = 0) then:
0: Select ISR register for Receiver DMA transfer address extension.
1: Select DMASR register for Receiver DMA transfer address extension.

RECEIVER DMA TRANSACTION COUNTER REGISTER (I2CRDC)
R251 - Read / Write
Register Page: 20
Reset Value: Undefined

7							0
RC7	RC6	RC5	RC4	RC3	RC2	RC1	RF/MEM

Bits 7:1 = **RC[7:1]** *Receiver DMA Counter Pointer.*
I2CRDC contains the address of the pointer (in the Register File) of the DMA receiver transaction counter when the DMA between Peripheral and Memory Space is selected. Otherwise (DMA between Peripheral and Register File), this register points to a pair of registers that are used as DMA Address register and DMA Transaction Counter.
See Section 0.1.6.1 and Section 0.1.6.2 for more details on the use of this register.

Bit 0 = **RF/MEM** *Receiver Register File/ Memory Selector.*
0: DMA towards Memory
1: DMA towards Register file

I²C BUS INTERFACE (Cont'd)

TRANSMITTER DMA SOURCE ADDRESS POINTER REGISTER (I2CTDAP)
R252 - Read / Write
Register Page: 20
Reset Value: Undefined

7							0
TA7	TA6	TA5	TA4	TA3	TA2	TA1	TPS

Bits 7:1= **TA[7:1]** *Transmit DMA Address Pointer.* I2CTDAP contains the address of the pointer (in the Register File) of the Transmitter DMA data source when the DMA between the peripheral and the Memory Space is selected. Otherwise (DMA between the peripheral and Register file), this register has no meaning.
See Section 0.1.6.2 for more details on the use of this register.

Bit 0 = **TPS** *Transmitter DMA Memory Pointer Selector*.
If memory has been selected for DMA transfer (DDCTDC.RF/MEM = 0) then:
0: Select ISR register for transmitter DMA transfer address extension.
1: Select DMASR register for transmitter DMA transfer address extension.

TRANSMITTER DMA TRANSACTION COUNTER REGISTER (I2CTDC)
R253 - Read / Write
Register Page: 20
Reset Value: Undefined

7							0
TC7	TC6	TC5	TC4	TC3	TC2	TC1	RF/MEM

Bits 7:1 = **TC[7:1]** *Transmit DMA Counter Pointer*. I2CTDC contains the address of the pointer (in the Register File) of the DMA transmitter transaction counter when the DMA between Peripheral and Memory Space is selected. Otherwise, if the DMA between Peripheral and Register File is selected, this register points to a pair of registers that are used as DMA Address register and DMA Transaction Counter.
See Section 0.1.6.1 and Section 0.1.6.2 for more details on the use of this register.

Bit 0 = **RF/MEM** *Transmitter Register File/ Memory Selector.*
0: DMA from Memory
1: DMA from Register file

EXTENDED CLOCK CONTROL REGISTER (I2CECCR)
R254 - Read / Write
Register Page: 20
Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	0	0	CC8	CC7

Bits 7:2 = Reserved. Must always be cleared.

Bits 1:0 = **CC[8:7]** *9-bit divider programming*
Implementation of a programmable clock divider. These bits and the CC[6:0] bits of the I2CCCR register select the speed of the bus (F_{SCL}).
For a description of the use of these bits, see the I2CCCR register.
They are not cleared when the interface is disabled (I2CCCR.PE=0).

I²C BUS INTERFACE (Cont'd)

INTERRUPT MASK REGISTER (I2CIMR)
R255 - Read / Write
Register Page: 20
Reset Value: 00xx 0000 (x0h)

7							0
RXDM	TXDM	REOBP	TEOBP	0	IERRM	IRXM	ITXM

Bit 7 = **RXDM** *Receiver DMA Mask*.
0: DMA reception disable.
1: DMA reception enable
RXDM is reset by hardware when the transaction counter value decrements to zero, that is when a Receiver End Of Block interrupt is issued.

Bit 6 = **TXDM** *Transmitter DMA Mask*.
0: DMA transmission disable.
1: DMA transmission enable.
TXDM is reset by hardware when the transaction counter value decrements to zero, that is when a Transmitter End Of Block interrupt is issued.

Bit 5 = **REOBP** *Receiver DMA End Of Block Flag*.
REOBP should be reset by software in order to avoid undesired interrupt routines, especially in initialization routine (after reset) and after entering the End Of Block interrupt routine.Writing "0" in this bit will cancel the interrupt request
Note: REOBP can only be written to "0".
0: End of block not reached.
1: End of data block in DMA receiver detected

Bit 4 = **TEOBP** *Transmitter DMA End Of Block* TEOBP should be reset by software in order to avoid undesired interrupt routines, especially in initialization routine (after reset) and after entering the End Of Block interrupt routine.Writing "0" will cancel the interrupt request.
Note: TEOBP can only be written to "0".
0: End of block not reached
1: End of data block in DMA transmitter detected.

Bit 3 = Reserved. This bit **must** be cleared.

Bit 2 = **IERRM** *Error Condition interrupt mask bit.*
This bit enables/ disables the Error interrupt.
0: Error interrupt disabled.
1: Error Interrupt enabled.

Bit 1 = **IRXM** *Data Received interrupt mask bit.*
This bit enables/ disables the Data Received and Receive DMA End of Block interrupts.
0: Interrupts disabled
1: Interrupts enabled

Note: This bit has no effect on DMA transfer

Bit 0 = **ITXM** *Peripheral Ready To Transmit interrupt mask bit.*
This bit enables/ disables the Peripheral Ready To Transmit and Transmit DMA End of Block interrupts.
0: Interrupts disabled
1: Interrupts enabled

Note: This bit has no effect on DMA transfer.

I²C BUS INTERFACE (Cont'd)

Table 44. I²C BUS Register Map and Reset Values

Address (Hex.)	Register Name	7	6	5	4	3	2	1	0
F0h	**I2CCR**	-	-	PE	ENGC	START	ACK	STOP	ITE
	Reset Value	0	0	0	0	0	0	0	0
F1h	**I2CSR1**	EVF	ADD10	TRA	BUSY	BTF	ADSL	M/SL	SB
	Reset Value	0	0	0	0	0	0	0	0
F2h	**I2CSR2**	-	0	ADDTX	AF	STOPF	ARLO	BERR	GCAL
	Reset Value	0	0	0	0	0	0	0	0
F3h	**I2CCCR**	FM/SM	CC6	CC5	CC4	CC3	CC2	CC1	CC0
	Reset Value	0	0	0	0	0	0	0	0
F4h	**I2COAR1**	ADD7	ADD6	ADD5	ADD4	ADD3	ADD2	ADD1	ADD0
	Reset Value	0	0	0	0	0	0	0	0
F5h	**I2COAR2**	FREQ1	FREQ0	EN10BIT	FREQ2	0	ADD9	ADD8	0
	Reset Value	0	0	0	0	0	0	0	0
F6h	**I2CDR**	DR7	DR6	DR5	DR4	DR3	DR2	DR1	DR0
	Reset Value	0	0	0	0	0	0	0	0
F7h	**I2CADR**	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	ADR0
	Reset Value	1	0	1	0	0	0	0	0
F8h	**I2CISR**	DMASTOP	PRL2	PRL1	PRL0		IERRP	IRXP	ITXP
	Reset Value	1	X	X	X	X	X	X	X
F9h	**I2CIVR**	V7	V6	V5	V4	V3	EV2	EV1	0
	Reset Value	X	X	X	X	X	X	X	0
FAh	**I2CRDAP**	RA7	RA6	RA5	RA4	RA3	RA2	RA1	RPS
	Reset Value	X	X	X	X	X	X	X	X
FBh	**I2CRDC**	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RF/MEM
	Reset Value	X	X	X	X	X	X	X	X
FCh	**I2CTDAP**	TA7	TA6	TA5	TA4	TA3	TA2	TA1	TPS
	Reset Value	X	X	X	X	X	X	X	X
FDh	**I2CTDC**	TC7	TC6	TC5	TC4	TC3	TC2	TC1	RF/MEM
	Reset Value	X	X	X	X	X	X	X	X
FEh	**I2CECCR**	0	0	0	0	0	0	CC8	CC7
		0	0	0	0	0	0	0	0
FFh	**I2CIMR**	RXDM	TXDM	REOBP	TEOBP		IERRM	IRXM	ITXM
	Reset Value	0	0	X	X	0	0	0	0

10.8 J1850 Byte Level Protocol Decoder (JBLPD)

10.8.1 Introduction

The JBLPD is used to exchange data between the ST9 microcontroller and an external J1850 transceiver I.C.

The JBLPD transmits a string of variable pulse width (VPW) symbols to the transceiver. It also receives VPW encoded symbols from the transceiver, decodes them and places the data in a register.

In-frame responses of type 0, 1, 2 and 3 are supported and the appropriate normalization bit is generated automatically. The JBLPD filters out any incoming messages which it does not care to receive. It also includes a programmable external loop delay.

The JBLPD uses two signals to communicate with the transceiver:

– VPWI (input)

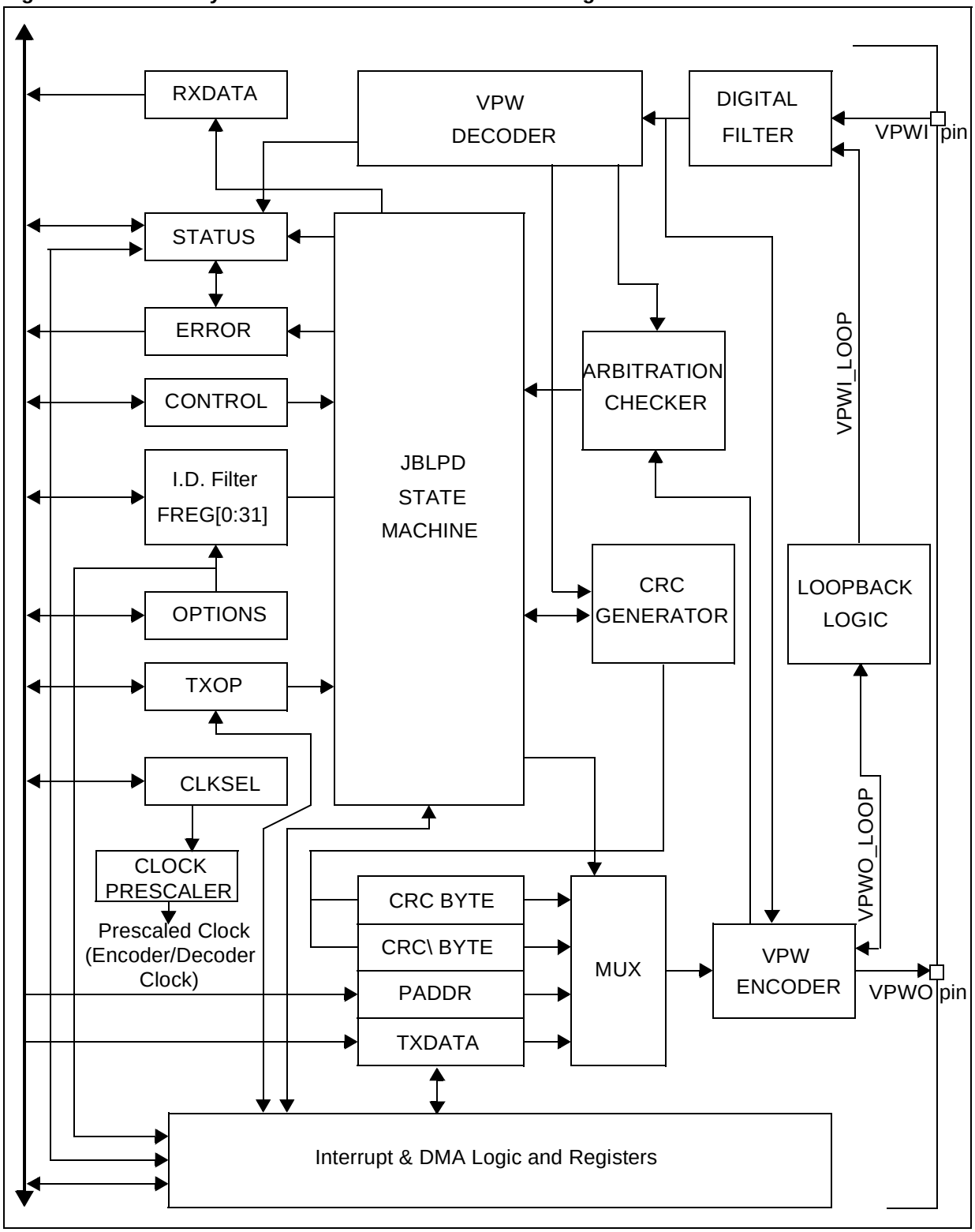
– VPWO (output)

10.8.2 Main Features

- SAE J1850 compatible
- Digital filter
- In-Frame Responses of type 0, 1, 2, 3 supported with automatic normalization bit
- Programmable External Loop Delay
- Diagnostic 4x time mode
- Diagnostic Local Loopback mode
- Wide range of MCU internal frequencies allowed
- Low power consumption mode (JBLPD suspended)
- Very low power consumption mode (JBLPD disabled)
- Don’t care message filter
- Selectable VPWI input polarity
- Selectable Normalization Bit symbol form
- 6 maskable interrupts
- DMA transmission and reception with End Of Block interrupts

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

Figure 109. JBLPD Byte Level Protocol Decoder Block Diagram



J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

10.8.3 Functional Description

10.8.3.1 J1850 protocol symbols

J1850 symbols are defined as a duration (in microseconds or clock cycles) and a state which can be either an active state (logic high level on VPWO) or a passive state (logic low level on VPWO).
An idle J1850 bus is in a passive state.
Any symbol begins by changing the state of the VPW line. The line is in this state for a specific duration depending on the symbol being transmitted.

Durations, and hence symbols, are measured as time between successive state transitions. Each symbol has only one level transition of a specific duration.
Symbols for logic zero and one data bits can be either a high or a low level, but all other symbols are defined at only one level.

Each symbol is placed directly next to another. Therefore, every level transition means that another symbol has begun.

Data bits of a logic zero are either a short duration if in a passive state or a long duration if in an active state. Data bits of a logic one are either a long duration if in a passive state or a short duration if in an active state. This ensures that data logic zeros predominate during bus arbitration.

An eight bit data byte transmission will always have eight transitions. For all data byte and CRC byte transfers, the first bit is a passive state and the last bit is an active state.

For the duration of the VPW, symbols are expressed in terms of Tv's (or VPW mode timing values). J1850 symbols and Tv values are described in the SAE J1850 specification, in Table 1 and in Table 2.

An ignored Tv I.D. occurs for level transitions which occur in less than the minimum time required for an invalid bit detect. The VPW encoder does not recognize these characters as they are filtered out by the digital filter. The VPW decoder does not resynchronize its counter with either edge of "ignored" pulses. Therefore, the counter which times symbols continues to time from the last transition which occurred after a valid symbol (including the invalid bit symbol) was recognized.

A symbol recognized as an invalid bit will resynchronize the VPW decoder to the invalid bit edges. In the case of the reception of an invalid bit, the JBLPD peripheral will set the IBD bit in the ERROR register. The JBLPD peripheral shall terminate any transmissions in progress, and disable receive transfers and RDRF flags until the VPW decoder recognizes a valid EOF symbol from the bus.

The JBLPD's state machine handles all the Tv I.D.s in accordance with the SAE J1850 specification.

Note: Depending on the value of a control bit, the polarity of the VPWI input can be the same as the J1850 bus or inverted with respect to it.

Table 45. J1850 Symbol definitions

Symbol	Definition
Data Bit Zero	Passive for Tv1 or Active for Tv2
Data Bit One	Passive for Tv2 or Active for Tv1
Start of Frame (SOF)	Active for Tv3
End of Data (EOD)	Passive for Tv3
End of Frame (EOF)	Passive for Tv4
Inter Frame Separation (IFS)	Passive for Tv6
IDLE Bus Condition (IDLE)	Passive for > Tv6
Normalization Bit (NB)	Active for Tv1 or Tv2
Break (BRK)	Active for Tv5

Table 46. J1850 VPW Mode Timing Value (Tv) definitions (in clock cycles)

Pulse Width or Tv I.D.	Minimum Duration	Nominal Duration	Maximum Duration
Ignored	0	N/A	<=7
Invalid Bit	>7	N/A	<=34
Tv1	>34	64	<=96
Tv2	>96	128	<=163
Tv3	>163	200	<=239
Tv4	>239	280	N/A
Tv5	>239	300	N/A
Tv6	>280	300	N/A

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

10.8.3.2 Transmitting Messages

This section describes the general procedures used by the JBLPD to successfully transmit J1850 frames of data out the VPWO pin. The first five sub-sections describe the procedures used for transmitting the specific transmit data types. The last section goes into the details of the transmitted symbol timing, synchronizing of symbols received from the external J1850 bus, and how data bit arbitration works.

The important concept to note for transmitting data is: the activity sent over the VPWO line should be timed with respect to the levels and transitions seen on the filtered VPWI line.

The J1850 bus is a multiplexed bus, and the VPWO & VPWI pins interface to this bus through a transceiver I.C. Therefore, the propagation delay through the transceiver I.C. and external bus filtering must be taken into account when looking for transmitted edges to appear back at the receiver.
The external propagation delay for an edge sent out on the VPWO line, to be detected on the VPWI line is denoted as $T_{p\text{-}ext}$ and is programmable between 0 and 31 µs nominal via the JDLY[4:0] bits in CONTROL register.

The transmitter VPW encoder sets the proper level to be sent out the VPWO line. It then waits for the corresponding level transition to be reflected back at the VPW decoder input.
Taking into account the external loop delay ($T_{p\text{-}ext}$) and the digital filter delay, the encoder will time its output to remain at this level so that the received symbol is at the correct nominal symbol time (refer to "Transmit Opcode Queuing" section). If arbitration is lost at any time during bit 0 or bit 1 transmission, then the VPWO line goes passive. At the end of the symbol time on VPWO, the encoder changes the state of VPWO if any more information is to be transmitted. It then times the new state change from the receiver decoder output.
Note that depending on the symbol (especially the SOF, NB0, NB1 symbols) the decoder output may actually change to the desired state before the transmit is attempted. It is important to still synchronize off the decoder output to time the VPWO symbol time.

A detailed description of the JBLPD opcodes can be find in the description of the OP[2:0] bits in the TXOP register.

Message Byte String Transmission (Type 0 IFR)

Message byte transmitting is the outputting of data bytes on the VPWO pin that occurs subsequent to a received bus idle condition. All message byte strings start with a SOF symbol transmission, then one or more data bytes are transmitted. A CRC byte is then transmitted followed by an EOD symbol (see Figure 2) to complete the transmission. If transmission is queued while another frame is being received, then the JBLPD will time an Inter-Frame Separation (IFS) time (Tv6) before commencing with the SOF character.

The user program will decide at some point that it wants to initiate a message byte string. The user program writes the TXDATA register with the first message data byte to be transmitted. Next, the TXOP register is written with the MSG opcode if more than one data byte is contained within the message, or with MSG+CRC opcode if one data byte is to be transmitted. The action of writing the TXOP register causes the TRDY bit to be cleared signifying that the TXDATA register is full and a corresponding opcode has been queued. The JBLPD must wait for an EOF nominal time period at which time data is transferred from the TXDATA register to the transmit shift register. The TRDY bit is again set since the TXDATA register is empty.

The JBLPD should also begin transmission if another device begins transmitting early. As long as an EOF minimum time period elapses, the JBLPD should begin timing and asserting the SOF symbol with the intention of arbitrating for the bus during the transmission of the first data byte. If a transmit is requested during an incoming SOF symbol, the JBLPD should be able to synchronize itself to the incoming SOF up to a time of Tv1 max. (96 µs) into the SOF symbol before declaring that it was too late to arbitrate for this frame.

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

If the J1850 bus was IDLE at the time the first data byte and opcode are written, the transmitter will immediately transfer data from the TXDATA register to the transmit shift register. The TRDY bit will once again be set signifying the readiness to accept a new data byte. The second data byte can then be written followed by the respective opcode. In the case of the last data byte, the TXOP register should be written with the MSG+CRC opcode. The transmitter will transmit the internally generated CRC after the last bit of the data byte. Once the TRDY bit is set signifying the acceptance of the last data byte, the first byte of the next message can be queued by writing the TXDATA register followed by a TXOP register write. The block will wait until the current data and the CRC data byte are sent out and a new IFS has expired before transmitting the new data. This is the case even if IFR data reception takes place in the interim.

Lost arbitration any time during the transfer of type 0 data will be honoured by immediately relinquishing control to the higher priority message. The TLA bit in the STATUS register is set accordingly and an interrupt will be generated assuming the TLA_M bit in the IMR register is set. It is responsibility of the user program to re-send the message beginning with the first byte if desired. This may be done at any time by rewriting only the TXOP register **if** the TXDATA contents have not changed.

Any transmitted data and CRC bytes during the transmit frame will also be received and transferred to the RXDATA register if the corresponding message filter bit is set in the FREG[0:31] registers. If the corresponding bit is not set in FREG[0:31], then the transmitted data is also not transferred to RXDATA. Also, the RDRF will not get set during frame and receive events such as RDOF & EODM.

NOTE: The correct procedure for transmitting is to write first the TXDATA register and then the TXOP register except during DMA transfers (see Section 0.1.6.4 DMA Management in Transmission Mode).

Transmitting a Type 1 IFR

The user program will decide to transmit an IFR type 1 byte in response to a message which is currently being received (See Figure 3). It does so by writing the IFR1 opcode to the TXOP register. Transmitting IFR data type 1 requires only a single write of the TXOP register with the IFR1 opcode set. The MLC[3:0] bits should be set to the proper "byte-received-count-required-before-IFR'ing" value. If no error conditions (IBD, IFD, TRA, RBRK or CRCE) exist to prevent transmission, the JBLPD peripheral will then transmit out the contents of the PADDR register at the next EOD nominal time period or at a time greater than the EOD minimum time period if a falling edge is detected on filtered J1850 bus line signifying another transmitter is beginning early. The NB1 symbol precedes the PADDR register value and is followed with an EOF delimiter. The TRDY flag is cleared on the write of the TXOP register. The TRDY bit is set once the NB1 begins transmitting.

Although the JBLPD should never lose arbitration for data in the IFR portion of a type 1 frame, higher priority messages are always honoured under the rules of arbitration. If arbitration is lost then the VPWO line is set to the passive state. The TLA bit in the STATUS register is set accordingly and an interrupt will be generated if enabled. The IFR1 is not retried. It is lost if the JBLPD peripheral loses arbitration. Also, the data that made it out on the bus will be received in the RXDATA register if not put into sleep mode. Note that for the transmitter to synchronize to the incoming signals of a frame, an IFR should be queued before an EODM is received for the present frame.

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

Transmitting a Type 2 IFR

The user program will decide to transmit an IFR type 2 byte in response to a message which is currently being received (See Figure 4). It does so by writing the IFR2 opcode to the TXOP register. Transmitting IFR data type 2 requires only a single write of the TXOP register with the IFR2 opcode set. The MLC[3:0] bits can also be set to check for message length errors. If no error conditions (IBD, IFD, TRA, RBRK or CRCE) exist to prevent transmission, the JBLPD will transmit out the contents of the PADDR register at the next EOD nominal time period or after an EOD minimum time period if a rising edge is detected on the filtered VPWI line signifying another transmitter beginning early. The NB1 symbol precedes the PADDR register value and is followed with an EOF delimiter. The TRDY flag will be cleared on the write of the TXOP register. The TRDY bit is set once the NB1 begins transmitting.

Lost arbitration for this case is a normal occurrence since type 2 IFR data is made up of single bytes from multiple responders. If arbitration is lost the VPWO line is released and the JBLPD waits until the byte on the VPWI line is completed. Note that the IFR that did make it out on the bus will be received in the RXDATA register if it is not put into sleep mode. Then, the JBLPD re-attempts to send its physical address immediately after the end of the last byte. The TLA bit is not set if arbitration is lost and the user program does not need to re-queue data or an opcode. The JBLPD will re-attempt to send its PADDR register contents until it successfully does so or the 12-byte frame maximum is reached if NFL=0. If NFL=1, then re-attempts to send an lFR2 are executed until cancelled by the CANCEL opcode or a JBLPD disable. Note that for the transmitter to synchronize to the incoming signals of a frame, an IFR should be queued before an EODM is received for the present frame.

Transmitting a Type 3 IFR Data String

The user program will decide to transmit an IFR type 3 byte string in response to a message which is currently being received (See Figure 5). It does so by writing the IFR3 or IFR3+CRC opcode to the TXOP register. Transmitting IFR data type 3 is similar to transmitting a message, in that the TXDATA register is written with the first data byte followed by a TXOP register write. For a single data byte IFR3 transmission, the TXOP register would be written with IFR3+CRC opcode set. The MLC[3:0] bits can also be set to a proper value to check for message length errors before enabling the IFR transmit.

If no error conditions (IBD, IFD, TRA, RBRK or CRCE) exist to prevent transmission, the JBLPD will wait for an EOD nominal time period on the filtered VPWI line (or for at least an EOD minimum time followed by a rising edge signifying another transmitter beginning early) at which time data is transferred from the TXDATA register to the transmit shift register. The TRDY bit is set since the TXDATA register is empty. A NB0 symbol is output on the VPWO line followed by the data byte and possibly the CRC byte if a IFR3+CRC opcode was set. Once the first IFR3 byte has been successfully transmitted, successive IFR3 bytes are sent with TXDATA/TXOP write sequences where the MLC[3:O] bits are don't cares. The final byte in the IFR3 string must be transmitted with the IFR3+CRC opcode to trigger the JBLPD to append the CRC byte to the string. The user program may queue up the next message opcode sequence once the TRDY bit has been set.

Although arbitration should never be lost for data in the IFR portion of a type 3 frame, higher priority messages are always honoured under the rules of arbitration. If arbitration is lost then the block should relinquish the bus by taking the VPWO line to the passive state. In this case the TLA bit in the STATUS register is set, and an interrupt will be generated if enabled. Note also, that the IFR data that did make it out on the bus will be received in the RXDATA register if not in sleep mode. Note that for the transmitter to synchronize to the incoming signals of a frame, an IFR should be queued before an EODM is received for the current frame.

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

Figure 110. J1850 String Transmission Type 0

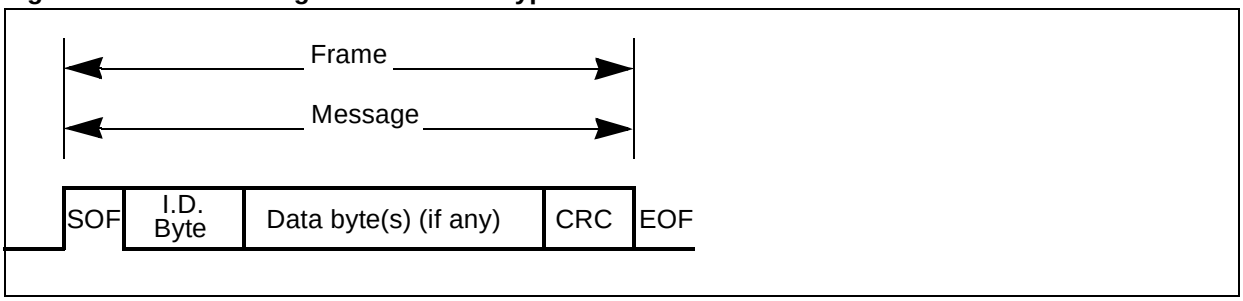


Figure 111. J1850 String Transmission Type 1

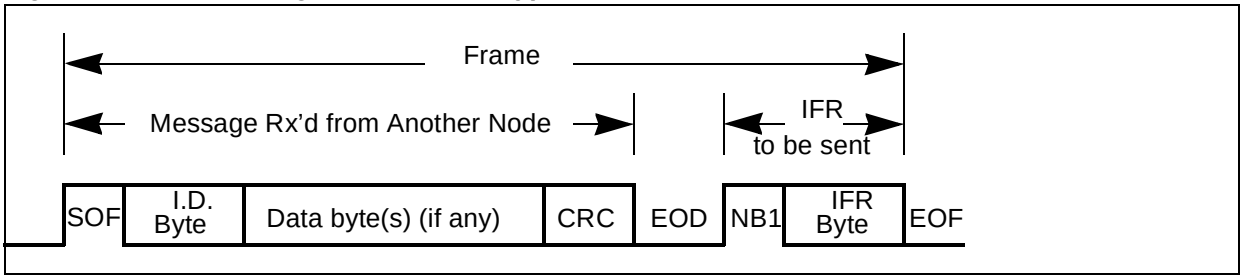


Figure 112. J1850 String Transmission Type 2

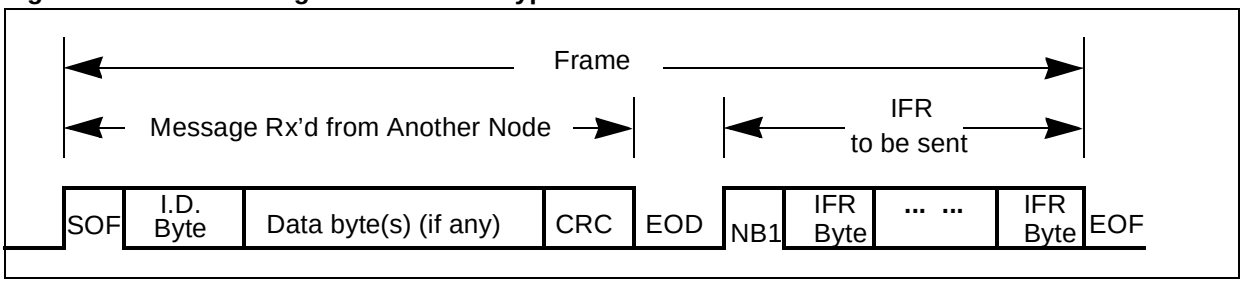
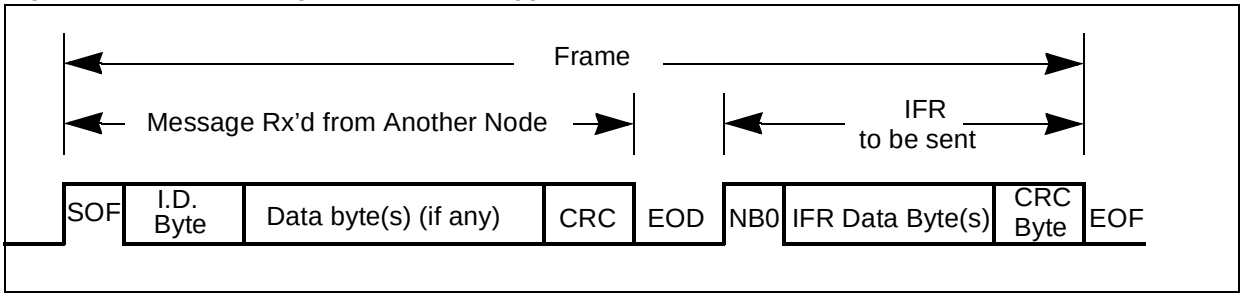


Figure 113. J1850 String Transmission Type 3



J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

Transmit Opcode Queuing

The JBLPD has the capability of queuing opcode transmits written to the TXOP register until J1850 bus conditions are in a correct state for the transmit to occur. For example, a MSGx opcode can be queued when the JBLPD is presently receiving a frame (or transmitting a MSG+CRC opcode) or an IFRx opcode can be queued when currently receiving or transmitting the message portion of a frame.

Queuing a MSG or MSG+CRC opcode for the next frame can occur while another frame is in progress. A MSGx opcode is written to the TXOP register when the present frame is past the point where arbitration for control of the bus for this frame can occur. The JBLPD will wait for a nominal IFS symbol (or EOFmin if another node begins early) to appear on the VPWI line before commencing to transmit this queued opcode. The TRDY bit for the queued opcode will remain clear until the EOFmin is detected on the VPWI line where it will then get set. Queued MSGx transmits for the next frame do not get cancelled for TLA, IBD, IFD or CRCE errors that occur in the present frame. An RBRK error will cancel a queued opcode for the next frame.

Queuing an IFRx opcode for the present frame can occur at any time after the detection of the beginning of an SOF character from the VPWI line. The queued IFR will wait for a nominal EOD symbol (or EODmin if another node begins early) before commencing to transmit the IFR. A queued IFR transmit will be cancelled on IBD, IFD, CRCE, RBRK errors as well as on a correct message length check error or frame length limit violation if these checks are enabled.

Transmit Bus Timing, Arbitration, and Synchronization

The external J1850 bus on the other side of the transceiver I.C. is a single wire multiplex bus with multiple nodes transmitting a number of different types of message frames. Each node can transmit at any time and synchronization and arbitration is used to determine who wins control of the transmit. It is the obligation of the JBLPD transmitter section to synchronize off of symbols on the bus, and to place only nominal symbol times onto the bus within the accuracy of the peripheral (+/- 1 µs).

To transmit proper symbols the JBLPD must know what is going on out on the bus. Fortunately, the JBLPD has a receiver pin which tells the transmitter about bus activity. Due to characteristics of the J1850 bus and the eight-clock digital filter, the signals presented to the VPW symbol decoder are delayed a certain amount of time behind the actual J1850 bus. Also, due to wave shaping and other signal conditioning of the transceiver I.C. the actions of the VPWO pin on the transmitter take time to appear on the bus itself. The total external J1850 bus delays are defined in the SAE J1850 standard as nominally 16 µs. The nominal 16 µs loop delay will actually vary between different transceiver I.C's. The JBLPD peripheral thus includes a programmability of the external loop delay in the bit positions JDLY[4:0]. This assures only nominal transmit symbols are placed on the bus by the JBLPD.

The method of transmitting for the JBLPD includes interaction between the transmitter and the receiver. The transmitter starts a symbol by placing the proper level (active or passive) on its VPWO pin. The transmitter then waits for the corresponding pin transition (inverted, of course) at the VPW decoder input. Note that the level may actually appear at the input before the transmitter places the value on the VPWO pin. Timing of the remainder of the symbol starts when the transition is detected. Refer to Figure 7, Case 1. The symbol timeout value is defined as:

SymbolTimeout = NominalSymbolTime - ExternalLoopDelay - 8 µs

NominalSymbolTime = Tv Symbol time
ExternalLoopDelay = defined via JDLY[4:0]
8 µs = Digital Filter

Bit-by-bit arbitration must be used to settle the conflicts that occur when multiple nodes attempt to transmit frames simultaneously. Arbitration is applied to each data bit symbol transmitted starting after the SOF or NBx symbol and continuing until the EOD symbol. During simultaneous transmissions of active and passive states on the bus, the resultant state on the bus is the active state. If the JBLPD detects a received symbol from the bus that is different from the symbol being transmitted, then the JBLPD will discontinue its transmit operation prior to the start of the next bit. Once arbitration has been lost, the VPWO pin must go passive within one period of the prescaled clock of the peripheral. Figure 6 shows 3 nodes attempting to arbitrate for the bus with Node B eventually winning with the highest priority data.

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

Figure 114. J1850 Arbitration Example

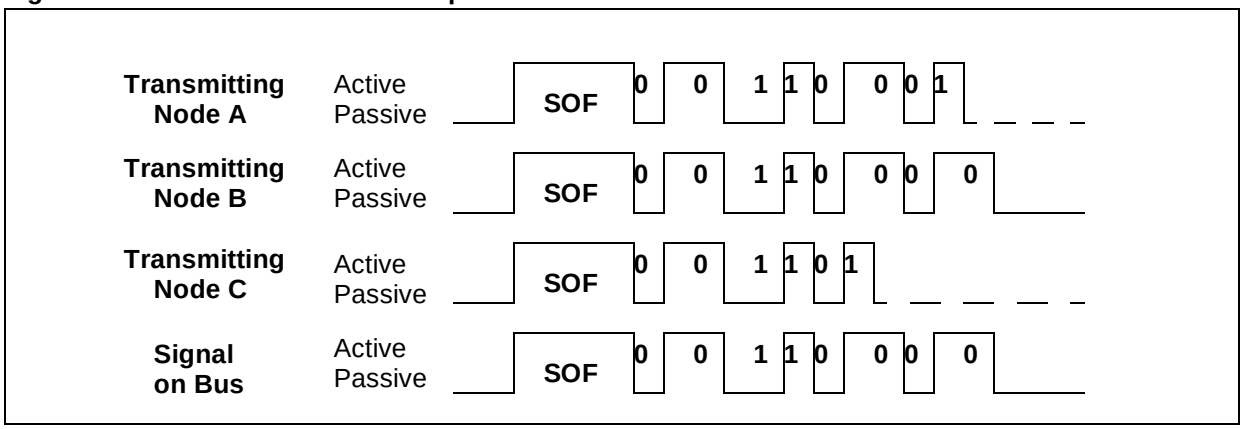
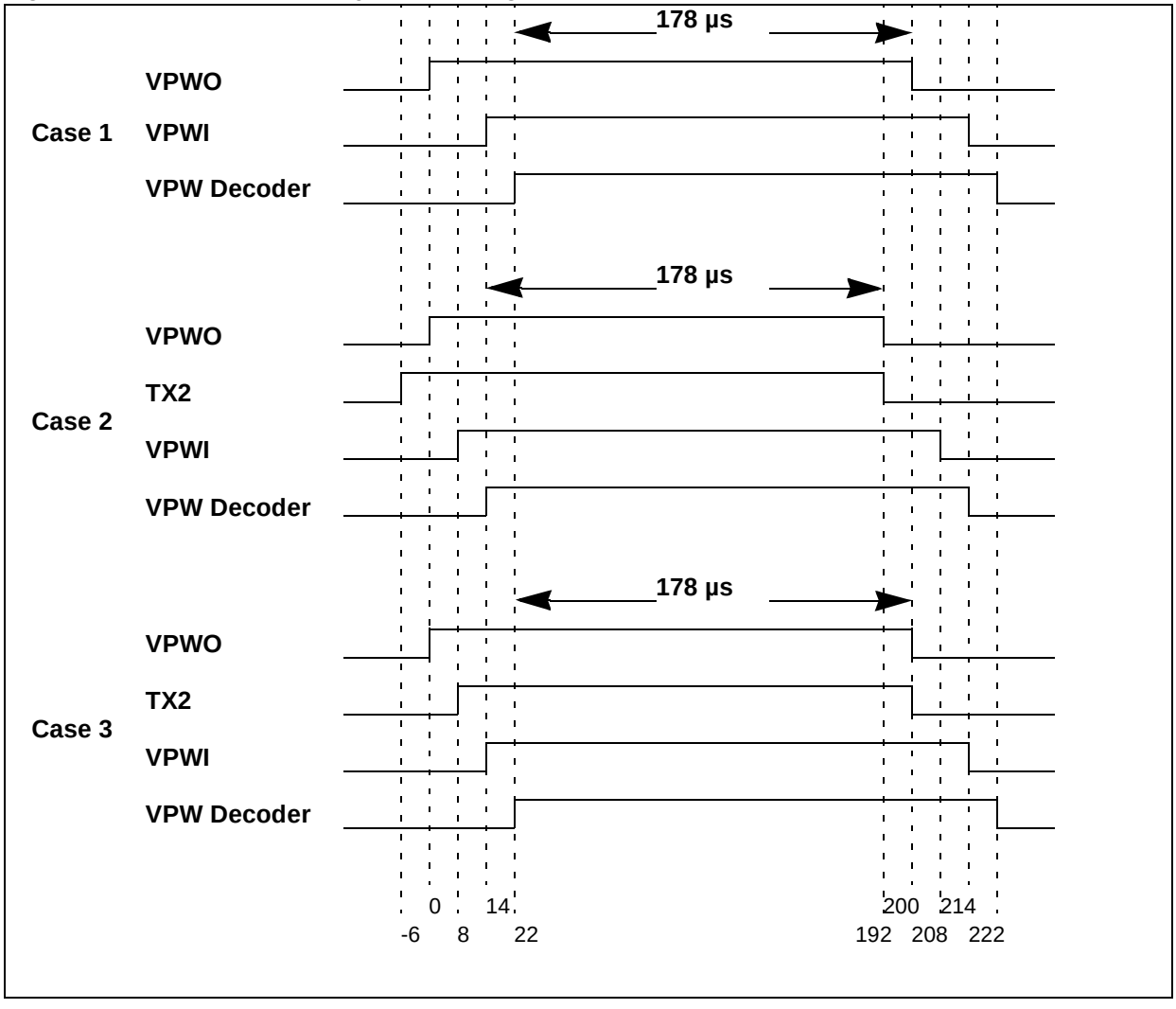


Figure 115. J1850 Received Symbol Timing



J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

Use of symbol and bit synchronization is an integral part of the J1850 bus scheme. Therefore, tight coupling of the encoder and decoder functions is required to maintain synchronization during transmits. Transmitted symbols and bits are initiated by the encoder and are timed through the decoder to realize synchronization. Figure 7 exemplifies synchronization with 3 examples for an SOF symbol and JDLY[4:0] = 01110b.

Case 1 shows a single transmitter arbitrating for the bus. The VPWO pin is asserted, and 14µs later the bus transitions to an active state. The 14µs delay is due to the nominal delay through the external transceiver chip. The signal is echoed back to the transceiver through the VPWI pin, and proceeds through the digital filter. The digital filter has a loop delay of 8 clock cycles with the signal finally presented to the decoder 22 µs after the VPWO pin was asserted. The decoder waits 178 µs before issuing a signal to the encoder signifying the end of the symbol. The VPWO pin is de-asserted producing the nominal SOF bit timing (22 µs + 178µs = 200 µs).

Case 2 shows a condition where 2 transmitters attempt to arbitrate for the bus at nearly the same time with a second transmitter, TX2, beginning slightly earlier than the VPWO pin. Since the JBLPD always times symbols from its receiver perspective, 178µs after the decoder sees the rising edge it issues a signal to the encoder to signify the end of the SOF. Nominal SOF timings are maintained and the JBLPD re-synchronizes to TX2.

Case 3 again shows an example of 2 transmitters attempting to arbitrate for the bus at nearly the same time with the VPWO pin starting earlier than TX2. In this case TX2 is required to re-synchronize to VPWO.

All 3 examples exemplify how bus timings are driven from the receiver perspective. Once the receiver detects an active bus, the transmitter symbol timings are timed minus the transceiver and digital filter delays (i.e. SOF = 200 µs - 14µs - 8µs = 178µs). This synchronization and timing off of the VPWI pin occurs for every symbol while transmitting. This ensures true arbitration during data byte transmissions.

10.8.3.3 Receiving Messages

Data is received from the external analog transceiver on the VPWI pin. VPWI data is immediately passed through a digital filter that ignores all pulses that are less than 7µs. Pulses greater than or equal to 7µs and less than 34µs are flagged as invalid bits (IBD) in the ERROR register.

Once data passes through the filter, all delimiters are stripped from the data stream and data bits are shifted into the receive shift register by the decoder logic. The first byte received after a valid SOF character is compared with the flags contained in FREG[0:31]. If the compare indicates that this message should be received, then the receive shift register contents are moved to the receive data register (RXDATA) for the user program to access. The Receive Data Register Full bit (RDRF) is set to indicate that a complete byte has been received. For each byte that is to be received in a frame, once an entire byte has been received, the receive shift register contents are moved to the receive data register (RXDATA). All data bits received, including CRC bits, are transferred to the RXDATA register. The Receive Data Register Full bit (RDRF) is set to indicate that a complete byte has been received.

If the first byte after a valid SOF indicates non-reception of this frame, then the current byte in the receive shift register is inhibited from being transferred to the RXDATA register and the RDRF flag remains clear (see the "Received Message Filtering" section). Also, no flags associated with receiving a message (RDOF, CRCE, IFD, IBD) are set.

A CRC check is kept on all bytes that are transferred to the RXDATA register during message byte reception (succeeding an SOF symbol) and IFR3 reception (succeeding an NB0 symbol). The CRC is initialized on receipt of the first byte that follows an SOF symbol or an NB0 symbol. The CRC check concludes on receipt of an EODM symbol. The CRC error bit (CRCE), therefore, gets set after the EODM symbol has been recognized. Refer to the "SAE Recommended Practice - J1850" manual for more information on CRCs.

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

Received Message Filtering

The FREG[0:31] registers can be considered an array of 256 bits (the FREG[0].0 bit is bit 0 of the array and the FREG[31].7 bit is bit 255). The I.D. byte of a message frame is used as a pointer to the array (See Figure 8).

Upon the start of a frame, the first data byte received after the SOF symbol determines the I.D. of the message frame. This I.D. byte addresses the I.D. byte flags stored in registers FREG[0:31]. This operation is accomplished before the transfer of the I.D. byte into the RXDATA register and before the RDRF bit is set.

If the corresponding bit in the message filter array, FREG[0:31], is set to zero (0), then the I.D. byte is not transferred to the RXDATA register and the RDRF bit is not set. Also, the remainder of the message frame is ignored until reception of an EOFmin symbol. A received EOFmin symbol terminates the operation of the message filter and enables the receiver for the next message. None of the flags related to the receiver, other than IDLE, are set. The EODM flag does not get set during a filtered frame. No error flags other than RBRK can get set.

If the corresponding bit in the message filter array, FREG[0:31], is set to a one (1), then the I.D. byte is transferred to the RXDATA register and the RDRF is set. Also, the remainder of the message is received unless sleep mode is invoked by the user program. All receiver flags and interrupts function normally.

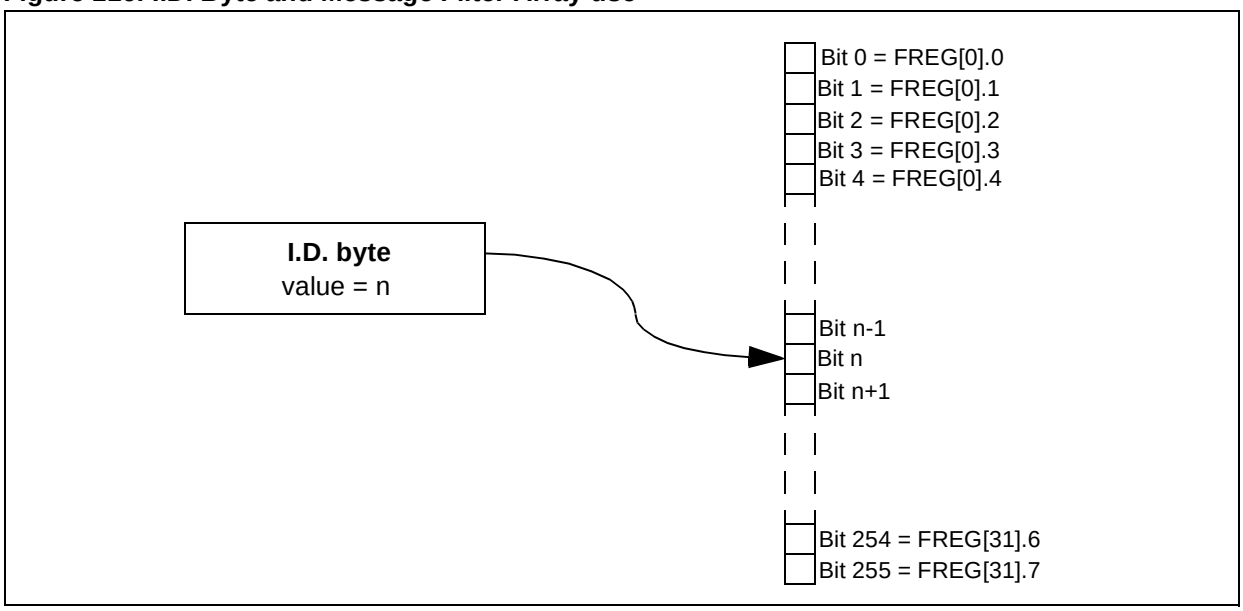
Note that a break symbol received during a filtered out message will still be received. Note also that the filter comparison occurs after reception of the first byte. So, any receive errors that occur before the message filter comparison (i.e. IBD, IFD) will be active at least until the filter comparison.

Transmitted Message Filtering

When transmitting a message, the corresponding FREG[0:31] I.D. filter bit may be set or cleared. If set, then the JBLPD will receive all data information transferred during the frame, unless sleep mode is invoked. Everything the JBLPD transmits will be reflected in the RXDATA register.

Because the JBLPD has invalid bit detect (IBD), invalid frame detect (IFD), transmitter lost arbitration (TRA), and Cyclic Redundancy Check Error (CRCE) it is not necessary for the transmitter to listen to the bytes that it is transmitting. The user may wish to filter out the transmitted messages from the receiver. This can reduce interrupt burden. When a transmitted I.D. byte is filtered by the receiver section of the block, then RDRF, RDOF, EODM flags are inhibited and no RXDATA transfers occur. The other flags associated normally with receiving - RBRK, CRCE, IFD, and IBD - are not inhibited, and they can be used to ascertain the condition of the message transmit.

Figure 116. I.D. Byte and Message Filter Array use



J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

10.8.3.4 Sleep Mode

Sleep mode allows the user program to ignore the remainder of a message. Normally, the user program can recognise if the message is of interest from the header bytes at the beginning of the message. If the user program is not interested in the message it simply writes the SLP bit in the PRLR register. This causes all additional data on the bus to be ignored until an EOF minimum occurs. No additional flags (but not the EOFM flag) and, therefore, interrupts are generated for the remainder of the message. The single exception to this is a received break symbol while in sleep mode. Break symbols always take precedence and will set the RBRK bit in the ERROR register and generate an interrupt if the ERR_M bit in IMR is set. Sleep mode and the SLP bit gets cleared on reception of an EOF or Break symbol.

Writes to the SLP bit will be ignored if:

1) A valid EOFM symbol was the last valid symbol detected,

AND

2) The J1850 bus line (after the filter) is passive.

Therefore, sleep mode can only be invoked after the SOF symbol and subsequent data has been received, but before a valid EOF is detected. If sleep mode is invoked within this time window, then any queued IFR transmit is aborted. If a MSG type is queued and sleep mode is invoked, then the MSG type will remain queued and an attempt to transmit will occur after the EOF period has elapsed as usual.

If SLP mode is invoked while the JBLPD is currently transmitting, then the JBLPD effectively inhibits the RDRF, RDT, EODM, & RDOF flags from being set, and disallows RXDATA transfers. But, it otherwise functions normally as a transmitter, still allowing the TRDY, TLA, TTO, TDUF, TRA, IBD, IFD, and CRCE bits to be set if required. This mode allows the user to not have to listen while talking.

10.8.3.5 Normalization Bit symbol selection

The form of the NB0/NB1 symbol changes depending on the industry standard followed. A bit (NBSYMS) in the OPTIONS register selects the symbol timings used. Refer to Table 3.

10.8.3.6 VPWI input line management

The JBLPD is able to work with J1850 transceiver chips that have both inverted and not inverted RX signal. A dedicated bit (INPOL) of the OPTIONS register must be programmed with the correct value depending on the polarity of the VPWI input with respect to the J1850 bus line. Refer to the INPOL bit description for more details.

10.8.3.7 Loopback mode

The JBLPD is able to work in loopback mode. This mode, enabled setting the LOOPB bit of the OPTIONS register, internally connects the output signal (VPWO) of the JBLPD to the input (VPWI) without polarity inversion. The external VPWO pin of the MCU is forced in its passive state and the external VPWI pin is ignored (Refer to Figure 9).

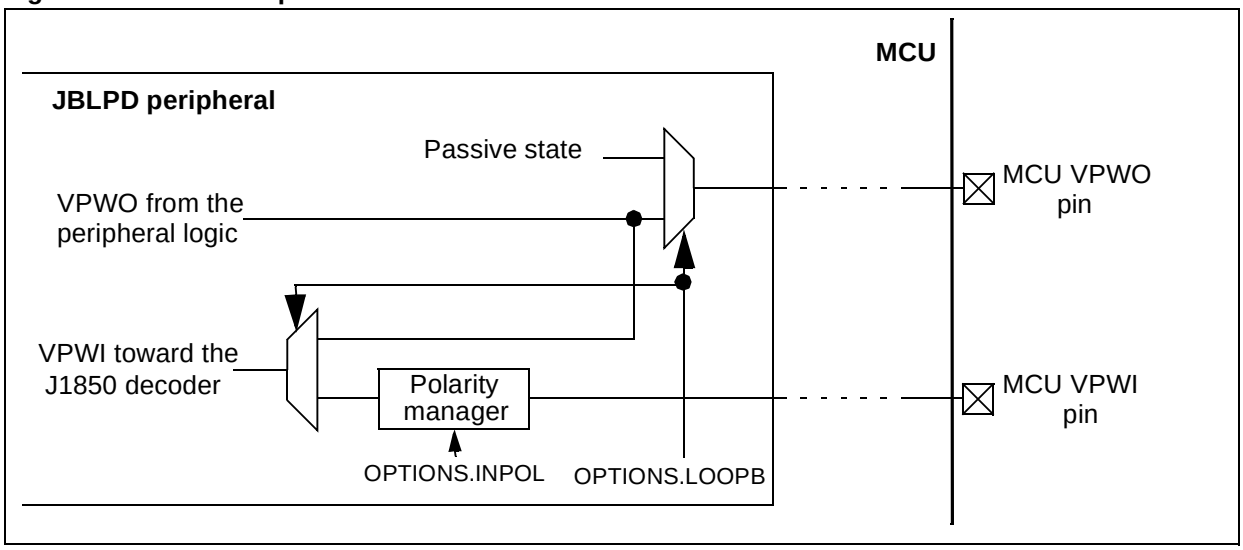
Note: When the LOOPB bit is set or reset, edges could be detected by the J1850 decoder on the internal VPWI line. These edges could be managed by the JBLPD as J1850 protocol errors. It is suggested to enable/disable LOOPB when the JBLPD is suspended (CONTROL.JE=0, CONTROL.JDIS=0) or when the JBLPD is disabled (CONTROL.JDIS=1).

Table 47. Normalization Bit configurations

	Symbol	NBSYMS=0	NBSYMS=1
IFR with CRC	**NB0**	active Tv2 (active long)	active Tv1 (active short)
IFR without CRC	**NB1**	active Tv1 (active short)	active Tv2 (active long)

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

Figure 117. Local Loopback structure



10.8.3.8 Peripheral clock management

To work correctly, the encoder and decoder sections of the peripheral need an internal clock at 1MHz. This clock is used to evaluate the protocol symbols timings in transmission and in reception.

The prescaled clock is obtained by dividing the MCU internal clock frequency. The CLKSEL register allows the selection of the right prescaling factor. The six least significant bits of the register (FREQ[5:0]) must be programmed with a value using the following formula:

MCU Internal Freq. = 1MHz * (FREQ[5:0] + 1).

Note: If the MCU internal clock frequency is lower than 1MHz, the JBLPD is not able to work correctly. If a frequency lower than 1MHz is used, the user program must disable the JBLPD.

Note: When the MCU internal clock frequency or the clock prescaler factor are changed, the JBLPD could lose synchronization with the J1850 bus.

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

10.8.4 Peripheral Functional Modes

The JBLPD can be programmed in 3 modes, depending on the value of the JE and JDIS bits in the CONTROL register, as shown in Table 4.

Table 48. JBLPD functional modes

JE	JDIS	mode
0	1	JBLPD Disabled
0	0	JBLPD Suspended
1	0	JBLPD Enabled

Depending on the mode selected, the JBLPD is able or unable to transmit or receive messages. Moreover the power consumption of the peripheral is affected.

Note: The configuration with both JE and JDIS set is forbidden.

10.8.4.1 JBLPD Enabled

When the JBLPD is enabled (CONTROL.JE=1), it is able to transmit and receive messages. Every feature is available and every register can be written.

10.8.4.2 JBLPD Suspended (Low Power Mode)

When the JBLPD is suspended (CONTROL.JE=0 and CONTROL.JDIS=0), all the logic of the JBLPD is stopped except the decoder logic.
This feature allows a reduction of power consumption when the JBLPD is not used, even if the decoder is able to follow the bus traffic. So, at any time the JBLPD is enabled, it is immediately synchronized with the J1850 bus.

Note: While the JBLPD is suspended, the STATUS register, the ERROR register and the SLP bit of the PRLR register are forced into their reset value.

10.8.4.3 JBLPD Disabled (Very Low Power Mode)

Setting the JDIS bit in the CONTROL register, the JBLPD is stopped until the bit is reset by software. Also the J1850 decoder is stopped, so the JBLPD is no longer synchronized with the bus. When the bit is reset, the JBLPD will wait for a new idle state on the J1850 bus. This mode can be used to minimize power consumption when the JBLPD is not used.

Note: While the JDIS bit is set, the STATUS register, the ERROR register, the IMR register and the SLP, TEOBP and REOBP bits of the PRLR register are forced to their reset value.

Note: In order that the JDIS bit is able to reset the IMR register and the TEOBP and REOBP bits, the JDIS bit must be left at 1 at least for 6 MCU clock cycles (3 NOPs).

Note: The JE bit of CONTROL register cannot be set with the same instruction that reset the JDIS bit. It can be set only after the JDIS bit is reset.

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

10.8.5 Interrupt Features

The JBLPD has six interrupt sources that it handles using the internal interrupts protocol. Other two interrupt sources (REOB and TEOB) are related to the DMA feature (See Section 0.1.6 DMA Features).
No external interrupt channel is used by the JBLPD.

The dedicated registers of the JBLPD should be loaded with appropriate values to set the interrupt vector (see the description of the IVR register), the interrupt mask bits (see the description of the IMR register) and the interrupt pending bits (see the description of the STATUS and PRLR registers).

The interrupt sources are as follows:

– The ERROR interrupt is generated when the ERROR bit of the STATUS register is set. This bit is set when the following events occur: Transmitter Timeout, Transmitter Data Underflow, Receiver Data Overflow, Transmit Request Aborted, Received Break Symbol, Cyclic Redundancy Check Error, Invalid Frame Detect, Invalid Bit Detect (a more detailed description of these events is given in the description of the ERROR register).

– The TLA interrupt is generated when the transmitter loses the arbitration (a more detailed description of this condition is given in the TLA bit description of the STATUS register).

– The EODM interrupt is generated when the JBLPD detects a passive level on the VPWI line longer than the minimum time accepted by the standard for the End Of Data symbol (a more detailed description of this condition is given in the EODM bit description of the STATUS register).

– The EOFM interrupt is generated when the JBLPD detects a passive level on the VPWI line longer than the minimum time accepted by the standard for the End Of Frame symbol (a more detailed description of this condition is given in the EOFM bit description of the STATUS register).

– The RDRF interrupt is generated when a complete data byte has been received and placed in the RXDATA register (see also the RDRF bit description of the STATUS register).

– The REOB (Receive End Of Block) interrupt is generated when receiving using DMA and the last byte of a sequence of data is read from the JBLPD.

– The TRDY interrupt is generated by two conditions: when the TXOP register is ready to accept a new opcode for transmission; when the transmit state machine accepts the opcode for transmission (a more detailed description of this condition is given in the TRDY bit description of the STATUS register).

– The TEOB (Transmit End Of Block) interrupt is generated when transmitting using DMA and the last byte of a sequence of data is written to the JBLPD.

10.8.5.1 Interrupt Management

To use the interrupt features the user has to follow these steps:

– Set the correct priority level of the JBLPD

– Set the correct interrupt vector

– Reset the Pending bits

– Enable the required interrupt source

Note: It is strongly recommended to reset the pending bits before un-masking the related interrupt sources to avoid spurious interrupt requests.

The priority with respect the other ST9 peripherals is programmable by the user setting the three most significant bits of the Interrupt Priority Level register (PRLR). The lowest interrupt priority is obtained by setting all the bits (this priority level is never acknowledged by the CPU and is equivalent to disabling the interrupts of the JBLPD); the highest interrupt priority is programmed resetting the bits. See the Interrupt and DMA chapters of the datasheet for more details.
When the JBLPD interrupt priority is set, the priority between the internal interrupt sources is fixed by hardware as shown in Table 5.

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

Note: After an MCU reset, the DMA requests of the JBLPD have a higher priority than the interrupt requests.
If the DMASUSP bit of the OPTIONS register is set, while the ERROR and TLA flags are set, no DMA transfer will be performed, allowing the relavent interrupt routines to manage each condition and, if necessary, disable the DMA transfer (Refer to Section 0.1.6 DMA Features).

Table 49. JBLPD internal priority levels

Priority Level	Interrupt Source
Higher	ERROR, TLA
	EODM, EOFM
	RDRF, REOB
Lower	TRDY, TEOB

The user can program the most significant bits of the interrupt vectors by writing the V[7:3] bits of the IVR register. Starting from the value stored by the user, the JBLPD sets the three least significant bits of the IVR register to produce four interrupt vectors that are associated with interrupt sources as shown in Table 6.

Table 50. JBLPD interrupt vectors

Interrupt Vector	Interrupt Source
V[7:3] 000b	ERROR, TLA
V[7:3] 010b	EODM, EOFM
V[7:3] 100b	RDRF, REOB
V[7:3] 110b	TRDY, TEOB

Each interrupt source has a pending bit in the STATUS register, except the DMA interrupt sources that have the interrupt pending bits located in the PRLR register.
These bits are set by hardware when the corresponding interrupt event occurs. An interrupt request is performed only if the related mask bits are set in the IMR register and the JBLPD has priority. The pending bits have to be reset by the user software. Note that until the pending bits are set (while the corresponding mask bits are set), the JBLPD processes interrupt requests. So, if at the end of an interrupt routine the related pending bit is not reset, another interrupt request is performed.
To reset the pending bits, different actions have to be done, depending on each bit: see the description of the STATUS and PRLR registers.

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

10.8.6 DMA Features

The JBLPD can use the ST9 on-chip Direct Memory Access (DMA) channels to provide high-speed data transactions between the JBLPD and contiguous locations of Register File and Memory. The transactions can occur from and toward the JBLPD. The maximum number of transactions that each DMA channel can perform is 222 with Register File or 65536 with Memory. Control of the DMA features is performed using registers located in the JBLPD register page (IVR, PRLR, IMR, RDAPR, RDCPR, TDAPR, TDCPR).

The priority level of the DMA features of the JBLPD with respect to the other ST9 peripherals and the CPU is the same as programmed in the PRLR register for the interrupt sources. In the internal priority level order of the JBLPD, depending on the value of the DMASUSP bit in the OPTIONS register, the DMA may or may not have a higher priority than the interrupt sources.
Refer to the Interrupt and DMA chapters of the datasheet for details on priority levels.

The DMA features are enabled by setting the appropriate enabling bits (RXD_M, TXD_M) in the IMR register. It is also possible to select the direction of the DMA transactions.

Once the DMA table is completed (the transaction counter reaches 0 value), an interrupt request to the CPU is generated if the related mask bit is set (RDRF_M bit in reception, TRDY_M bit in transmission). This kind of interrupt is called "End Of Block". The peripheral sends two different "End Of Block" interrupts depending on the direction of the DMA (Receiving End Of Block (REOB) - Transmitting End Of Block (TEOB)). These interrupt sources have dedicated interrupt pending bits in the PRLR register (REOBP, TEOBP) and they are mapped to the same interrupt vectors: "Receive Data Register Full (RDRF)" and "Transmit Ready (TRDY)" respectively. The same correspondence exists for the internal priority between interrupts and interrupt vectors.

10.8.6.1 DMA between JBLPD and Register File

If the DMA transaction is made between the JBLPD and the Register File, one register is required to hold the DMA Address and one to hold the DMA transaction counter. These two registers must be located in the Register File: the DMA Address Register in an even addressed register, the DMA Transaction Counter in the following register (odd address). They are pointed to by the DMA Transaction Counter Pointer Register (RDCPR register in receiving, TDCPR register in transmitting) located in the JBLPD register page.

To select DMA transactions with the Register File, the control bits RDCPR.RF/MEM in receiving mode or TDCPR.RF/MEM in transmitting mode must be set.

The transaction Counter Register must be initialized with the number of DMA transfers to perform and it will be decremented after each transaction. The DMA Address Register must be initialized with the starting address of the DMA table in the Register File, and it is incremented after each transaction. These two registers must be located between addresses 00h and DFh of the Register File.

When the DMA occurs between JBLPD and Register File, the TDAPR register (in transmission) and the RDAPR register (in reception) are not used.

10.8.6.2 DMA between JBLPD and Memory Space

If the DMA transaction is made between the JBLPD and Memory, a register pair is required to hold the DMA Address and another register pair to hold the DMA Transaction counter. These two pairs of registers must be located in the Register File. The DMA Address pair is pointed to by the DMA Address Pointer Registers (RDAPR register in reception, TDAPR register in transmission) located in the JBLPD register page; the DMA Transaction Counter pair is pointed to by the DMA Transaction Counter Pointer Registers (RDCPR register in reception, TDCPR register in transmission) located in the JBLPD register page.

To select DMA transactions with Memory Space, the control bits RDCPR.RF/MEM in receiving mode or TDCPR.RF/MEM in transmitting mode must be reset.

The Transaction Counter register pair must be initialized with the number of DMA transfers to perform and it will be decremented after each transaction. The DMA Address register pair must be initialized with the starting address of the DMA table in Memory Space, and it is incremented after each transaction. These two register pairs must be located between addresses 00h and DFh of the Register File.

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

10.8.6.3 DMA Management in Reception Mode

The DMA in reception is performed when the RDRF bit of the STATUS register is set (by hardware). The RDRF bit is reset as soon as the DMA cycle is finished.
To enable the DMA feature, the RXD_M bit of the IMR register must be set (by software).

Each DMA request performs the transfer of a single byte from the RXDATA register of the peripheral toward Register File or Memory Space (Figure 10).

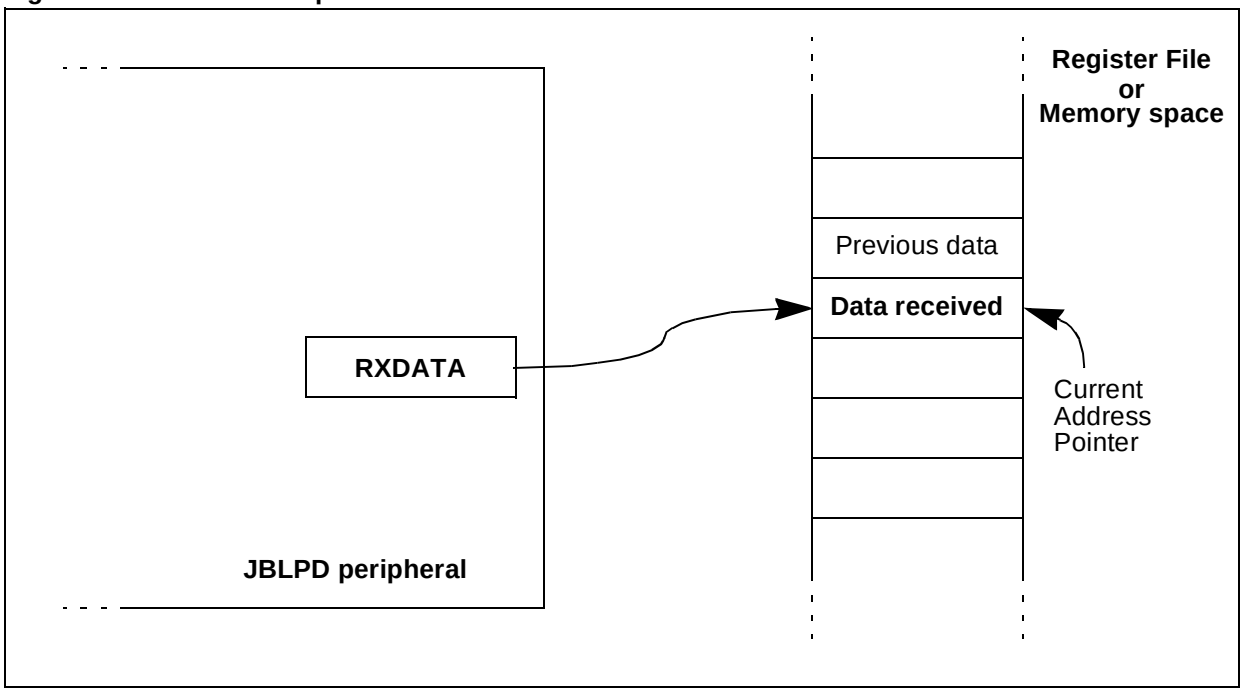
Each DMA transfer consists of three operations that are performed with minimum use of CPU time:

– A load from the JBLPD data register (RXDATA) to a location of Register File/Memory addressed through the DMA Address Register (or Register pair);
– A post-increment of the DMA Address Register (or Register pair);
– A post-decrement of the DMA transaction counter, which contains the number of transactions that have still to be performed.

Note: When the REOBP pending bit is set (at the end of the last DMA transfer), the reception DMA enable bit (RXD_M) is automatically reset by hardware. However, the DMA can be disabled by software resetting the RXD_M bit.

Note: The DMA request acknowledge could depend on the priority level stored in the PRLR register.

Figure 118. DMA in Reception Mode



J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

10.8.6.4 DMA Management in Transmission Mode

DMA in transmission is performed when the TRDY bit of the STATUS register is set (by hardware). The TRDY bit is reset as soon as the DMA cycle is finished.
To enable the DMA feature, the TXD_M bit in the IMR register must be set (by software).

Compared to reception, in transmission each DMA request performs the transfer of either a single byte or a couple of bytes depending on the value of the Transmit Opcode bits (TXOP.OP[2:0]) written during the DMA transfer.
The table of values managed by the DMA must be a sequence of opcode bytes (that will be written in the TXOP register by the DMA) each one followed by a data byte (that will be written in the TXDATA register by the DMA) if the opcode needs it (see Figure 11).

Each DMA cycle consists of the following transfers for a total of three/six operations that are performed with minimum use of CPU time:

– A load to the JBLPD Transmit Opcode register (TXOP) from a location of Register File/Memory addressed through the DMA Address Register (or Register pair);
– A post-increment of the DMA Address Register (or Register pair);
– A post-decrement of the DMA transaction counter, which contains the number of transactions that have still to be performed;

and if the Transmit Opcode placed in TXOP requires a datum:

– A load to the peripheral data register (TXDATA) from a location of Register File/Memory addressed through the DMA Address Register (or Register pair); it is the next location in the TXDATA transfer cycle;
– A post-increment of the DMA Address Register (or Register pair);
– A post-decrement of the DMA transaction counter, which contains the number of transactions that have still to be performed.

Note: When the TEOBP pending bit is set (at the end of the last DMA transfer), the transmission DMA enable bit (TXD_M) is automatically reset by hardware. However, the DMA can be disabled by software resetting the TXD_M bit.

Note: When using DMA, the TXOP byte is written before the TXDATA register. This order is accepted by the JBLPD **only** when the DMA in transmission is enabled.

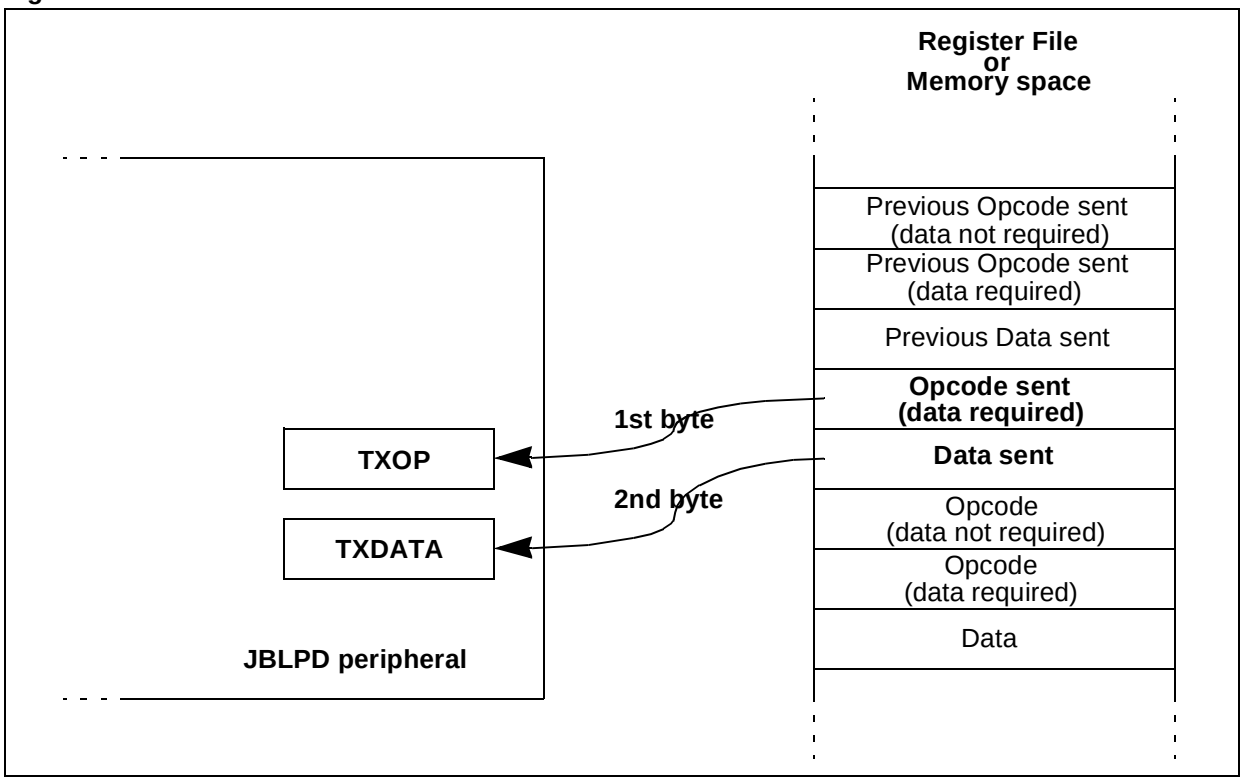
Note: The DMA request acknowledge could depend on the priority level stored in the PRLR register. In the same way, some time can occur between the transfer of the first byte and the transfer of the second one if another interrupt or DMA request with higher priority occurs.

10.8.6.5 DMA Suspend mode

In the JBLPD it is possible to suspend or not to suspend the DMA transfer while some J1850 protocol events occur. The selection between the two modes is done by programming the DMASUSP bit of the OPTIONS register.
If the DMASUSP bit is set (DMA suspended mode), while the ERROR or TLA flag is set, the DMA transfers are suspended, to allow the user program to handle the event condition.
If the DMASUSP bit is reset (DMA not suspended mode), the previous flags have no effect on the DMA transfers.

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

Figure 119. DMA in Transmission Mode



J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

10.8.7 Register Description

The JBLPD peripheral uses 48 registers that are mapped in a single page of the ST9 register file.

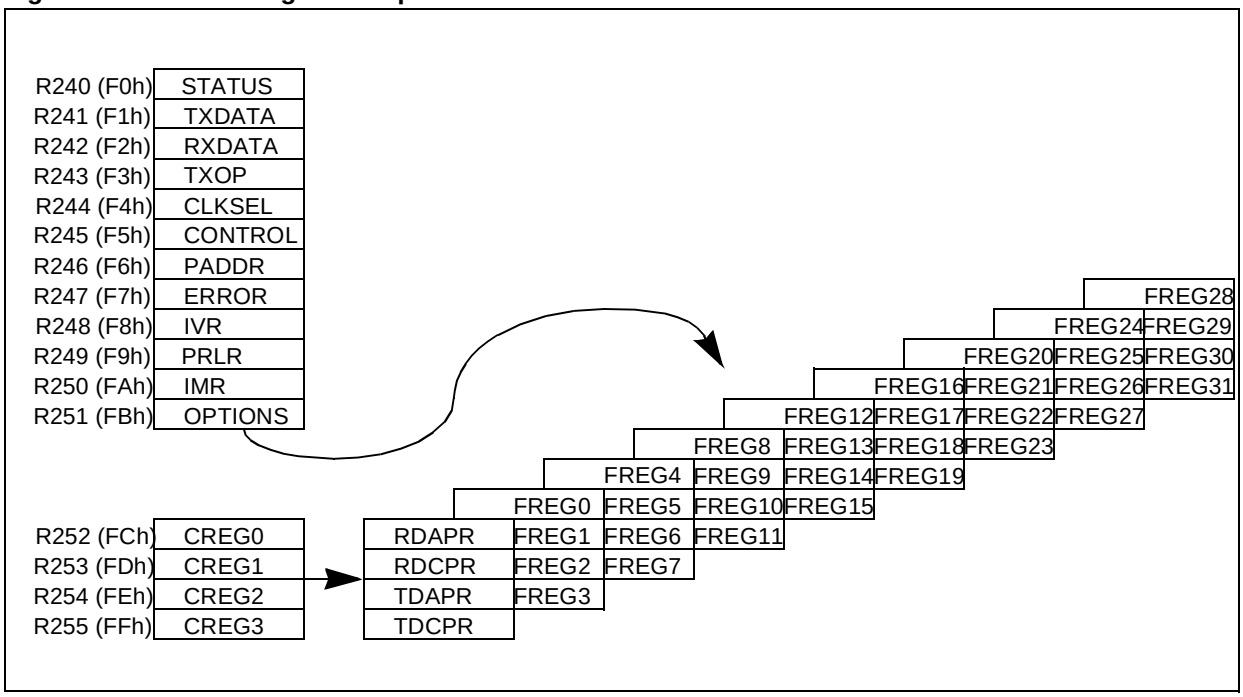
Twelve registers are mapped from R240 (F0h) to R251 (FBh): these registers are usually used to control the JBLPD. See Section 0.1.7.1 Un-Stacked Registers for a detailed description of these registers.

Thirty-six registers are mapped from R252 (FCh) to R255 (FFh). This is obtained by creating 9 sub-pages, each containing 4 registers, mapped in the same register addresses; 4 bits (RSEL[3:0]) of a register (OPTIONS) are used to select the current sub-page. See Section 0.1.7.2 Stacked Registers section for a detailed description of these registers.

The ST9 Register File page used is 23 (17h).

NOTE: Bits marked as "Reserved" should be left at their reset value to guarantee software compatibility with future versions of the JBLPD.

Figure 120. JBLPD Register Map



J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

10.8.7.1 Un-Stacked Registers

STATUS REGISTER (STATUS)
R240 - Read/Write
Register Page: 23
Reset Value: 0100 0000 (40h)

7							0
ERR	TRDY	RDRF	TLA	RDT	EODM	EOFM	IDLE

The bits of this register indicate the status of the JBLPD peripheral.
This register is forced to its reset value after the MCU reset and while the CONTROL.JDIS bit is set. While the CONTROL.JE bit is reset, all bits except IDLE are forced to their reset values.

Bit 7 = **ERR** *Error Flag.*
The ERR bit indicates that one or more bits in the ERROR register have been set. As long as any bit in the ERROR register remains set, the ERR bit remains set. When all the bits in the ERROR register are cleared, then the ERR bit is reset by hardware. The ERR bit is also cleared on reset or while the CONTROL.JE bit is reset, or while the CONTROL.JDIS bit is set.
If the ERR_M bit of the IMR register is set, when this bit is set an interrupt request occurs.
0: No error
1: One or more errors have occurred

Bit 6 = **TRDY** *Transmit Ready Flag.*
The TRDY bit indicates that the TXOP register is ready to accept another opcode for transmission. The TRDY bit is set when the TXOP register is empty and it is cleared whenever the TXOP register is written (by software or by DMA). TRDY will be set again when the transmit state machine accepts the opcode for transmission.
When attempting to transmit a data byte without using DMA, two writes are required: first a write to TXDATA, then a write to the TXOP.

– If a byte is written into the TXOP which results in TRA getting set, then the TRDY bit will immediately be set.
– If a TLA occurs and the opcode for which TRDY is low is scheduled for this frame, then TRDY will go high, if the opcode is scheduled for the next frame, then TRDY will stay low.
– If an IBD, IFD or CRCE error condition occurs, then TRDY will be set and any queued transmit opcode scheduled to transmit in the present frame will be cancelled by the JBLPD peripheral. A MSGx opcode scheduled to be sent in the next frame will not be cancelled for these errors, so TRDY would not get set.
– An RBRK error condition cancels all transmits for this frame or any successive frames, so the TRDY bit will always be immediately set on an RBRK condition.

TRDY is set on reset or while CONTROL.JE is reset, or while the CONTROL.JDIS bit is set.
If the TRDY_M bit of the IMR register is set, when this bit is set an interrupt request occurs.
0: TXOP register not ready to receive a new opcode
1: TXOP register ready to receive a new opcode

Bit 5 = **RDRF** *Receive Data Register Full Flag.*
RDRF is set when a complete data byte has been received and transferred from the serial shift register to the RXDATA register.
RDRF is cleared when the RXDATA register is read (by software or by DMA). RDRF is also cleared on reset or while CONTROL.JE is reset, or while CONTROL.JDIS bit is set.
If the RDRF_M bit of the IMR register is set, when this bit is set an interrupt request occurs.
0: RXDATA register doesn't contain a new data
1: RXDATA register contains a new data

Bit 4 = **TLA** *Transmitter Lost Arbitration.*
The TLA bit gets set when the transmitter loses arbitration while transmitting messages or type 1 and 3 IFRs. Lost arbitration for a type 2 IFR does not set the TLA bit. (Type 2 messages require retries of the physical address if the arbitration is lost until the frame length is reached (if NFL=0)). The TLA bit gets set when, while transmitting a MSG, MSG+CRC, IFR1, IFR3, or IFR3+CRC, the decoded VPWI data bit symbol received does not match the VPWO data bit symbol that the JBLPD is attempting to send out. If arbitration is lost, the VPWO line is switched to its passive state and nothing further is transmitted until an end-of-data (EOD) symbol is detected on the VPWI line. Also, any queued transmit opcode scheduled for transmission during this frame is cancelled (but the TRA bit is not set).
The TLA bit can be cleared by software writing a logic "zero" in the TLA position. TLA is also cleared on reset or while CONTROL.JE is reset, or while CONTROL.JDIS bit is set.
If the TLA_M bit of the IMR register is set, when this bit is set an interrupt request occurs.
0: The JBLPD doesn't lose arbitration
1: The JBLPD loses arbitration

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

Bit 3 = **RDT** *Receive Data Type.*
The RDT bit indicates the type of data which is in the RXDATA register: message byte or IFR byte. Any byte received after an SOF but before an EODM is considered a message byte type. Any byte received after an SOF, EODM and NBx is an IFR type.
RDT gets set or cleared at the same time that RDRF gets set.
RDT is cleared on reset or while CONTROL.JE is reset, or while CONTROL.JDIS bit is set.
0: Last RXDATA byte was a message type byte
1: Last RXDATA byte was a IRF type byte

Bit 2 = **EODM** *End of Data Minimum Flag.*
The EODM flag is set when the JBLPD decoded VPWI pin has been in a passive state for longer that the minimum Tv3 symbol time unless the EODM is inhibited by a sleep, filter or CRCE, IBD, IFD or RBRK error condition during a frame. EODM bit does not get set when in the sleep mode or when a message is filtered.
The EODM bit can be cleared by software writing a logic "zero" in the EODM position. EODM is cleared on reset, while CONTROL.JE is reset or while CONTROL.JDIS bit is set.
If the EODM_M bit of the IMR register is set, when this bit is set an interrupt request occurs.
0: No EOD symbol detected
1: EOD symbol detected

Note: The EODM bit is not an error flag. It means that the minimum time related to the passive Tv3 symbol is passed.

Bit 1 = **EOFM** *End of Frame Minimum Flag.*
The EOFM flag is set when the JBLPD decoded VPWI pin has been in a passive state for longer that the minimum Tv4 symbol time. EOFM will still get set at the end of filtered frames or frames where sleep mode was invoked. Consequently, multiple EOFM flags may be encountered between frames of interest.
The EOFM bit can be cleared by software writing a logic "zero" in the EOFM position. EOFM is cleared on reset, while CONTROL.JE is reset or while CONTROL.JDIS bit is set.
If the EOFM_M bit of the IMR register is set, when this bit is set an interrupt request occurs.
0: No EOF symbol detected
1: EOF symbol detected

Note: The EOFM bit is not an error flag. It means that the minimum time related to the passive Tv4 symbol is passed.

Bit 0 = **IDLE** *Idle Bus Flag*
IDLE is set when the JBLPD decoded VPWI pin recognized an IFS symbol. That is, an idle bus is when the bus has been in a passive state for longer that the Tv6 symbol time. The IDLE flag will remain set as long as the decoded VPWI pin is passive. IDLE is cleared when the decoded VPWI pin transitions to an active state.
Note that if the VPWI pin remains in a passive state after JE is set, then the IDLE bit may go high sometime before a Tv6 symbol is timed on VPWI (since VPWI timers may be active when JE is clear).
IDLE is cleared on reset or while the CONTROL.JDIS bit is set.
0: J1850 bus not in idle state
1: J1850 bus in idle state

JBLPD TRANSMIT DATA REGISTER (TXDATA)
R241- Read/Write
Register Page: 23
Reset Value: xxxx xxxx (xxh)

7							0
TXD7	TXD6	TXD5	TXD4	TXD3	TXD2	TXD1	TXD0

The TXDATA register is an eight bits read/write register in which the data to be transmitted must be placed. A write to TXDATA merely enters a byte into the register. To initiate an attempt to transmit the data, the TXOP register must also be written. When the TXOP write occurs, the TRDY flag is cleared. While the TRDY bit is clear, the data is still in the TXDATA register, so writes to the TXDATA register with TRDY clear will overwrite existing TXDATA. When the TXDATA is transferred to the shift register, the TRDY bit is set again.
Reads of the TXDATA register will always return the last byte written.
TXDATA contents are undefined after a reset.

Note: The correct sequence to transmit is to write first the TXDATA register (if datum is needed) and then the TXOP one.
Only using the DMA, the correct sequence of writing operations is first the TXOP register and then the TXDATA one (if needed).

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

JBLPD RECEIVE DATA REGISTER (RXDATA)
R242- Read only
Register Page: 23
Reset Value: xxxx xxxx (xxh)

7							0
RXD7	RXD6	RXD5	RXD4	RXD3	RXD2	RXD1	RXD0

The RXDATA register is an 8-bit read only register in which the data received from VPWI is stored. VPWI data is transferred from the input VPW decoder to a serial shift register unless it is inhibited by sleep mode, filter mode or an error condition (IBD, IFD, CRCE, RBRK) during a frame. When the shift register is full, this data is transferred to the RXDATA register, and the RDRF flag gets set. All received data bytes are transferred to RXDATA including CRC bytes. A read of the RXDATA register will clear the RDRF flag.
Note that care must be taken when reading RXDATA subsequent to an RDRF flag. Multiple reads of RXDATA after an RDRF should only be attempted if the user can be sure that another RDRF will not occur by the time the read takes place.
RXDATA content is undefined after a reset.

JBLPD TRANSMIT OPCODE REGISTER (TXOP)
R243 - Read/Write
Register Page: 23
Reset Value: 0000 0000 (00h)

7							0
MLC3	MLC2	MLC1	MLC0	-	OP2	OP1	OP0

TXOP is an 8-bit read/write register which contains the instructions required by the JBLPD to transmit a byte. A write to the TXOP triggers the state machine to initialize an attempt to serially transmit a byte out on the VPWO pin. An opcode which triggers a message byte or IFR type 3 to be sent will transfer the TXDATA register contents to the transmit serial shift register. An opcode which triggers a message byte or IFR type 3 to be sent with a CRC appended will transfer the TXDATA register contents to the transmit serial shift register and subsequently the computed CRC byte. An opcode which triggers an IFR type 1 or 2 to be sent will transfer the PADDR register contents to the transmit serial shift register. If a TXOP opcode is written which is invalid for the bus conditions at the time (e.g. 12 byte frame or IFR3ing an IFR2), then no transmit attempt is tried and the TRA bit in the ERROR register is set.
Transmission of a string of data bytes requires multiple TXDATA/TXOP write sequences. Each write combination should be accomplished while the TRDY flag is set. However, writes to the TXOP when TRDY is not set will be accepted by the state machine, but it may override the previous data and opcode.
Under normal message transmission conditions the MSG opcode is written. If the last data byte of a string is to be sent, then the MSG+CRC opcode will be written. An IFRx opcode is written if a response byte or bytes to a received message (i.e. bytes received in RXDATA with RDT=0) is wanted to transmit. The Message Length Count bits (MLC[3:0]) may be used to require that the IFR be enabled only if the correct number of message bytes has been received.

NOTE: The correct sequence to transmit is to write first the TXDATA register and then the TXOP one. Only using the DMA, the correct sequence of writing operations is first the TXOP register and then the TXDATA one (if needed).

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

Bit 7:4 = **MLC[3:0]** *Message Length Count.*
Message Length Count bits 3 to 0 are written when the program writes one of the IFR opcodes. Upon detection of the EOD symbol which delineates the body of a frame from the IFR portion of the frame, the received byte counter is compared against the count contained in MLC[3:0]. If they match, then the IFR will be transmitted. If they do not match, then the TRA bit in the ERROR register is set and no transmit attempt occurs.

– While NFL=0, an MCL[3:0] decimal value between 1 and 11 is considered valid. MCL[3:0] values of 12, 13, 14, 15 are considered invalid and will set the Transmit Request Aborted (TRA) bit in the ERROR register.

– While NFL=1, an MCL[3:0] value between 1 and 15 is considered valid.

– For NFL=1 or 0, MCL[3:0] bits are don't care during a MSG or MSG+CRC opcode write.

– If writing an IFR opcode and MCL[3:0]=0000, then the message length count check is ignored (i.e. MLC=Count is disabled), and the IFR is enabled only on a correct CRC and a valid EOD symbol assuming no other error conditions (IFD, IBD, RBRK) appear.

Bit 3 = Reserved.

Bit 2:0 = **OP[2:0]** *Transmit Opcode Select Bits.*
The bits OP[2:0] form the code that the transmitter uses to perform a transmit sequence. The codes are listed in Table 7.

Table 51. Opcode definitions

OP[2:0]	Transmit opcode	Abbreviation
000	No operation or Cancel	CANCEL
001	Send Break Symbol	SBRK
010	Message Byte	MSG
011	Message Byte then append CRC	MSG+CRC
100	In-Frame Response Type 1	IFR1
101	In-Frame Response Type 2	IFR2
110	In-Frame Response Type 3	IFR3
111	IFR Type 3 then append CRC	IFR3+CRC

MSG, Message Byte Opcode.
The Message byte opcode is set when the user program wants to initiate or continue transmitting the body of a message out the VPWO pin.
The body of a message is the string of data bytes following an SOF symbol, but before the first EOD symbol in a frame. If the J1850 bus is in an idle condition when the opcode is written, an SOF symbol is transmitted out the VPWO pin immediately before it transmits the data contained in TXDATA. If the JBLPD is not in idle and the J1850 transmitter has not been locked out by loss of arbitration, then the TXDATA byte is transferred to the serial output shift register for transmission immediately on completion of any previously transmitted data. The final byte of a message string is not transmitted using the MSG opcode (use the MSG+CRC opcode).

Special Conditions for MSG Transmit:

– 1) A MSG cannot be queued on top of an executing IFR3 opcode. If so, then TRA is set, and TDUF will get set because the transmit state machine will be expecting more data, then the inverted CRC is appended to this frame. Also, no message byte will be sent on the next frame.

– 2) If NFL = 0 and an MSG queued without CRC on Received Byte Count for this frame=10 will trigger the TRA to get set, and TDUF will get set because the state machine will be expecting more data and the transmit machine will send the inverted CRC after the byte which is presently transmitting. Also, no message byte will be sent on the next frame.

Caution should be taken when TRA gets set in these cases because the TDUF error sequence may engage before the user program has a chance to rewrite the TXOP register with the correct opcode. If a TDUF error occurs, a subsequent MSG write to the TXOP register will be used as the first byte of the next frame.

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

MSG+CRC, Message byte then append CRC opcode.
The 'Message byte with CRC' opcode is set when the user program wants to transmit a single byte message followed by a CRC byte, or transmit the final byte of a message string followed by a CRC byte.
A single byte message is basically an SOF symbol followed by a single data byte retrieved from TXDATA register followed by the computed CRC byte followed by an EOD symbol. If the J1850 bus is in idle condition when the opcode is written, an SOF symbol is immediately transmitted out the VPWO pin. It then transmits the byte contained in the TXDATA register, then the computed CRC byte is transmitted. VPWO is then set to a passive state. If the J1850 bus is not idle and the J1850 transmitter has not been locked out by loss of arbitration, then the TXDATA byte is transferred to the serial output shift register for transmission immediately on completion of any previously transmitted data. After completion of the TXDATA byte the computed CRC byte is transferred out the VPWO pin and then the VPWO pin is set passive to time an EOD symbol.
Special Conditions for MSG+CRC Transmit:

– 1) A MSG+CRC opcode cannot be queued on top of an executing IFR3 opcode. If so, then TRA is set, and TDUF will get set because the transmit state machine will be expecting more data, then the inverted CRC is appended to this frame. Also, no message byte will be sent on the next frame.

– 2) If NFL=0, a MSG+CRC can only be queued if Received Byte Count for this frame <=10 otherwise the TRA will get set, and TDUF will get set because the state machine will be expecting more data, so the transmit machine will send the inverted CRC after the byte which is presently transmitting. Also, no message byte will be sent on the next frame.

Caution should be taken when TRA gets set in these cases because the TDUF error sequence may engage before the user program has a chance to rewrite the TXOP register with the correct opcode. If a TDUF error occurs, a subsequent MSG+CRC write to the TXOP register will be used as the first byte of the next frame.

IFR1, In-Frame Response Type 1 opcode.
The In-frame Response Type 1 (IFR 1) opcode is written if the user program wants to transmit a physical address byte (contained in the PADDR register) in response to a message that is currently being received.
The user program decides to set up an IFR1 upon receiving a certain portion of the data byte string of an incoming message. No write of the TXDATA register is required. The IFR1 gets its data byte from the PADDR register.
The JBLPD block will enable the transmission of the IFR1 on these conditions:

– 1) The CRC check is valid (otherwise the CRCE is set)

– 2) The received message length is valid if enabled (otherwise the TRA is set)

– 3) A valid EOD minimum symbol is received (otherwise the IFD may eventually get set due to byte synchronization errors)

– 4) If NFL = 0 & Received Byte Count for this frame <=11 (otherwise TRA is set)

– 5) If not presently executing an MSG, IFR3, opcode (otherwise TRA is set, and TDUF will get set because the transmit state machine will be expecting more data, so the inverted CRC will be appended to this frame)

– 6) If not presently executing an IFR1, IFR2, or IFR3+CRC opcode otherwise TRA is set (but no TDUF)

– 7) If not presently receiving an IFR portion of a frame, otherwise TRA is set.

The IFR1 byte is then attempted according to the procedure described in section "Transmitting a type 1 IFR". Note that if an IFR1 opcode is written, a queued MSG or MSG+CRC is overridden by the IFR1.

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

IFR2, In-Frame Response Type 2 opcode.
The In-frame Response Type 2 (IFR2) opcode is set if the user program wants to transmit a physical address byte (contained in the PADDR register) in response to a message that is currently being received.
The user program decides to set up an IFR2 upon receiving a certain portion of the data byte string of an incoming message. No write of the TXDATA register is required. The IFR gets its data byte from the PADDR register.
The JBLPD block will enable the transmission of the IFR2 on these conditions:

– 1) The CRC check is valid (otherwise the CRCE is set)

– 2) The received message length is valid if enabled (otherwise the TRA is set)

– 3) A valid EOD minimum symbol is received (otherwise the IFD may eventually get set due to byte synchronization errors)

– 4) If NFL = 0 & Received Byte Count for this frame <=11 (otherwise TRA is set)

– 5) If not presently executing an MSG, IFR3, opcode (otherwise TRA is set, and TDUF will get set because the transmit state machine will be expecting more data, so the inverted CRC will be appended to this frame)

– 6) If not presently executing an IFR1, IFR2, or IFR3+CRC opcodes, otherwise TRA is set (but no TDUF)

– 7) If not presently receiving an IFR portion of a frame, otherwise TRA is set.

The IFR byte is then attempted according to the procedure described in section "Transmitting a type 2 IFR". Note that if an IFR opcode is written, a queued MSG or MSG+CRC is overridden by the IFR2.

IFR3, In-Frame Response Type 3 opcode.
The In-Frame Response Type 3 (IFR3) opcode is set if the user program wants to initiate to transmit or continue to transmit a string of data bytes in response to a message that is currently being received.
The IFR3 uses the contents of the TXDATA register for data. The user program decides to set up an IFR3 upon receiving a certain portion of the data byte string of an incoming message. A previous write of the TXDATA register should have occurred.
The JBLPD block will enable the transmission of the first byte of an IFR3 string on these conditions:

– 1) The CRC check is valid (otherwise the CRCE is set)

– 2) The received message length is valid if enabled (otherwise the TRA is set)

– 3) A valid EOD minimum symbol is received (otherwise the IFD may eventually get set due to byte synchronization errors)

– 4) If NFL = 0 & Received Byte Count for this frame <=9 (otherwise TRA is set and inverted CRC is transmitted due to TDUF)

– 5) If not presently executing an MSG opcode (otherwise TRA is set, and TDUF will get set because the transmit state machine will be expecting more data and the inverted CRC will be appended to this frame)

– 6) If not presently executing an IFR1, IFR2, or IFR3+CRC opcode, otherwise TRA is set (but no TDUF)

– 7) If not presently receiving an IFR portion of a frame, otherwise TRA is set.

The IFR3 byte string is then attempted according to the procedure described in section "Transmitting a type 3 IFR". Note that if an IFR3 opcode is written, a queued MSG or MSG+CRC is overridden by the IFR3.

The next byte(s) in the IFR3 data string shall also be written with the IFR3 opcode except for the last byte in the string which shall be written with the IFR3+CRC opcode. Each IFR3 data byte transmission is accomplished with a TXDATA/TXOP write sequence. The succeeding IFR3 transmit requests will be enabled on conditions 4 and 5 listed above.

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

IFR3+CRC, In-Frame Response Type 3 then append CRC opcode.
The In-frame Response Type 3 then append CRC opcode (IFR3+CRC) is set if the user program wants to either initiate to transmit a single data byte IFR3 followed by a CRC, or transmit the last data byte of an IFR3 string followed by the CRC byte in response to a message that is currently being received.
The IFR3+CRC opcode transmits the contents of the TXDATA register followed by the computed CRC byte. The user program decides to set up an IFR3 upon receiving a certain portion of the data byte string of an incoming message. A previous write of the TXDATA register should have occurred.

The J1850 block will enable the transmission of the first byte of an IFR3 string on these conditions:

– 1) The CRC check is valid (otherwise the CRCE is set)

– 2) The received message length is valid if enabled (otherwise the TRA is set)

– 3) A valid EOD minimum symbol is received (otherwise the IFD may eventually get set due to byte synchronization errors)

– 4) If NFL = 0 & Received Byte Count for this frame <=10 (otherwise TRA is set and inverted CRC is transmitted)

– 5) If not presently executing an MSG opcode (otherwise TRA is set, and TDUF will get set because the transmit state machine will be expecting more data and the inverted CRC will be appended to this frame)

– 6) If not presently executing an IFR1, IFR2 or IFR3+CRC opcodes, otherwise TRA is set (but no TDUF)

– 7) If not presently receiving an IFR portion of a frame, otherwise TRA is set.

The IFR3 byte is attempted according to the procedure described in section "Transmitting a type 3 IFR". The CRC byte is transmitted out on completion of the transmit of the IFR3 byte.
If this opcode sets up the last byte in an IFR3 data string, then the TXDATA register contents shall be transmitted out immediately upon completion of the previous IFR3 data byte followed by the transmit of the CRC byte. In this case the IFR3+CRC is enabled on conditions 4 and 5 listed above. Note that if an IFR3+CRC opcode is written, a queued MSG or MSG+CRC is overridden by the IFR3+CRC.

SBRK, Send Break Symbol.
The SBRK opcode is written to transmit a nominal break (BRK) symbol out the VPWO pin. A Break symbol can be initiated at any time. Once the SBRK opcode is written a BRK symbol of the nominal Tv5 duration will be transmitted out the VPWO pin immediately. To terminate the transmission of an in-progress break symbol the JE bit should be set to a logic zero. An SBRK command is non-maskable, it will override any present transmit operation, and it does not wait for the present transmit to complete. Note that in the 4X mode a SBRK will send a break character for the nominal Tv5 time times four (4 x Tv5) so that all nodes on the bus will recognize the break. A CANCEL opcode does not override a SBRK command.

CANCEL, No Operation or Cancel Pending Transmit.
The Cancel opcode is used by the user program to tell the J1850 transmitter that a previously queued opcode should not be transmitted. The Cancel opcode will set the TRDY bit. If the JBLPD peripheral is presently not transmitting, the Cancel command effectively cancels a pending MSGx or IFRx opcode if one was queued, or it does nothing if no opcode was queued. If the JBLPD peripheral is presently transmitting, then a queued MSGx or IFRx opcode is aborted and the TDUF circuit may take affect.

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

JBLPD SYSTEM FREQUENCY SELECTION REGISTER (CLKSEL)
R244- Read/Write
Register Page: 23
Reset Value: 0000 0000 (00h)

7							0
4X	-	FREQ5	FREQ4	FREQ3	FREQ2	FREQ1	FREQ0

Bit 7 = **4X** *Diagnostic Four Times Mode.*
This bit is set when the J1850 clock rate is chosen four times faster than the standard requests, to force the BREAK symbol (nominally 300 µs long) and the Transmitter Timeout Time (nominally 1 ms) at their nominal durations.

When the user want to use a 4 times faster J1850 clock rate, the new prescaler factor should be stored in the FREQ[5:0] bits and the 4X bit must be set with the same instruction. In the same way, to exit from the mode, FREQ[5:0] and 4X bits must be placed at the previous value with the same instruction.
0: Diagnostic Four Times Mode disabled
1: Diagnostic Four Times Mode enabled

Note: Setting this bit, the prescaler factor is not automatically divided by four. The user must adapt the value stored in FREQ[5:0] bits by software.

Note: The customer should take care using this mode when the MCU internal frequency is less than 4MHz.

Bit 6 = Reserved.

Bit 5:0 = **FREQ[5:0]** *Internal Frequency Selectors.*
These 6 bits must be programmed depending on the internal frequency of the device. The formula that must be used is the following one:
MCU Int. Freq.= 1MHz * (FREQ[5:0] + 1).

Note: To obtain a correct operation of the peripheral, the internal frequency of the MCU (INTCLK) must be an integer multiple of 1MHz and the correct value must be written in the register. So an internal frequency less than 1MHz is not allowed.

Note: If the MCU internal clock frequency is lower than 1MHz, the peripheral is not able to work correctly. If a frequency lower than 1MHz is used, the user program must disable the peripheral.

Note: When the clock prescaler factor or the MCU internal frequency is changed, the peripheral could lose the synchronization with the J1850 bus.

JBLPD CONTROL REGISTER (CONTROL)
R245- Read/Write
Register Page: 23
Reset Value: 0100 0000 (40h)

7							0
JE	JDIS	NFL	JDLY4	JDLY3	JDLY2	JDLY1	JDLY0

The CONTROL register is an eight bit read/write register which contains JBLPD control information. Reads of this register return the last written data.

Bit 7 = **JE** *JBLPD Enable.*
The JBLPD block enable bit (JE) enables and disables the transmitter and receiver to the VPWO and VPWI pins respectively. When the JBLPD peripheral is disabled the VPWO pin is in its passive state and information coming in the VPWI pin is ignored. When the JBLPD block is enabled, the transmitter and receiver function normally. Note that queued transmits are aborted when JE is cleared. JE is cleared on reset, by software and setting the JDIS bit.
0: The peripheral is disabled
1: The peripheral is enabled

Note: It is not possible to reset the JDIS bit and to set the JE bit with the same instruction. The correct sequence is to first reset the JDIS bit and then set the JE bit with another instruction.

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

Bit 6 = **JDIS** *Peripheral clock frozen*.
When this bit is set by software, the peripheral is stopped and the bus is not decoded anymore. A reset of the bit restarts the internal state machines as after a MCU reset. The JDIS bit is set on MCU reset.
0: The peripheral clock is running
1: The peripheral clock is stopped

Note: When the JDIS bit is set, the STATUS register, the ERROR register, the IMR register and the TEOBP and REOBP bits of the PRLR register are forced into their reset value.

Note: It is not possible to reset the JDIS bit and to set the JE bit with the same instruction. The correct sequence is to first reset the JDIS bit and then set the JE bit with another instruction.

Bit 5 = **NFL** *No Frame Length Check*
The NFL bit is used to enable/disable the J1850 requirement of 12 bytes maximum per frame limit. The SAE J1850 standard states that a maximum of 12 bytes (including CRCs and IFRs) can be on the J1850 between a start of frame symbol (SOF) and an end of frame symbol (EOF). If this condition is violated, then the JBLPD peripheral gets an Invalid Frame Detect (IFD) and the sleep mode ensues until a valid EOFM is detected. If the valid frame check is disabled (NFL=1), then no limits are imposed on the number of data bytes which can be sent or received on the bus between an SOF and an EOF. The default upon reset is for the frame checking to be enabled.
The NFL bit is cleared on reset
0: Twelve bytes frame length check enabled
1: Twelve bytes frame length check disabled

Bit 4:0 = **JDLY[4:0]** *JBLPD Transceiver External Loop Delay Selector.*
These five bits are used to select the nominal external loop time delay which normally occurs when the peripheral is connected and transmitting in a J1850 bus system. The external loop delay is defined as the time between when the VPWO is set to a certain level to when the VPWI recognizes the corresponding (inverted) edge on its input. Refer to "Transmit Opcode Queuing" section and the SAE-J1850 standard for information on how the external loop delay is used in timing transmitted symbols.
The allowed values are integer values between 0 µs and 31 µs.

JBLPD PHYSICAL ADDRESS REGISTER (PADDR)
R246- Read/Write
Register Page: 23
Reset Value: xxxx xxxx (xxh)

7							0
ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	ADR0

The PADDR is an eight bit read/write register which contains the physical address of the JBLPD peripheral. During initialization the user program will write the PADDR register with its physical address. The Physical Address is used during in-frame response types 1 and 2 to acknowledge the receipt of a message. The JBLPD peripheral will transmit the contents of the PADDR register for type 1 or 2 IFRs as defined by the TXOP register.
This register is undefined on reset.

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

JBLPD ERROR REGISTER (ERROR)
R247- Read only
Register Page: 23
Reset Value: 0000 0000 (00h)

7							0
TTO	TDUF	RDOF	TRA	RBRK	CRCE	IFD	IBD

ERROR is an eight bit read only register indicating error conditions that may arise on the VPWO and VPWI pins. A read of the ERROR register clears all bits (except for TTO and possibly the RBRK bit) which were set at the time of the read. The register is cleared after the MCU reset, while the CONTROL.JE bit is reset, or while the CONTROL.JDIS bit is set.

All error conditions that can be read in the ERROR register need to have redundant ERROR indicator flags because:

– With JE set, the TDUF, RDOF, TRA, CRCE, IFD, & IBD bits in the ERROR register can only be cleared by reading the register.

– The TTO bit can only be cleared by clearing the JE bit.

– The RBRK bit can only be cleared by reading the ERROR register after the break condition has disappeared.

Error condition indicator flags associated with the error condition are cleared when the error condition ends. Since error conditions may alter the actions of the transmitter and receiver, the error condition indicators must remain set throughout the error condition. All error conditions, including the RBRK condition, are events that get set during a particular clock cycle of the prescaled clock of the peripheral. The IFD, IBD, RBRK, and CRCE error conditions are then cleared when a valid EOF symbol is detected from the VPWI pin. The TRA error condition is a singular event that sets the corresponding ERROR register bit, but this error itself causes no other actions.

Bit 7 = **TTO** *Transmitter Timeout Flag*
The TTO bit is set when the VPWO pin has been in a logic one (or active) state for longer than 1 ms. This flag is the output of a diagnostic circuit based on the prescaled system clock input. If the 4X bit is not set, the TTO will trip if the VPWO is constantly active for 1000 prescaled clock cycles. If the 4X bit is set, then the TTO will timeout at 4000 prescaled clock cycles. When the TTO flag is set then the diagnostic circuit will disable the VPWO signal, and disable the JBLPD peripheral. The user program must then clear the JE bit to remove the TTO error. It can then retry the block by setting the JE bit again.
The TTO bit can be used to determine if the external J1850 bus is shorted low. Since the transmitter looks for proper edges returned at the VPWI pin for its timing, a lack of edges seen at VPWI when trying to transmit (assuming the RBRK does not get set) would indicate a constant low condition. The user program can take appropriate actions to test the J1850 bus circuit when a TTO occurs. Note that a transmit attempt must occur to detect a bus shorted low condition.
The TTO bit is cleared while the CONTROL.JE bit is reset or while the CONTROL.JDIS bit is set. TTO is cleared on reset.
0: VPWO line at 1 for less than 1 ms
1: VPWO line at 1 for longer than 1 ms

Bit 6 = **TDUF** *Transmitter Data Underflow.*
The TDUF will be set to a logic one if the transmitter expects more information to be transmitted, but a TXOP write has not occurred in time (by the end of transmission of the last bit).
The transmitter knows to expect more information from the user program when transmitting messages or type 3 IFRs only. If an opcode is written to TXOP that does not include appending a CRC byte, then the JBLPD peripheral assumes more data is to be written. When the JBLPD peripheral has shifted out the data byte it must have the next data byte in time to place it directly next to it. If the user program does not place new data in the TXDATA register and write the TXOP register with a proper opcode, then the CRC byte which is being kept tabulated by the transmitter is logically inverted and transmitted out the VPWO pin. This will ensure that listeners will detect this message as an error. In this case the TDUF bit is set to a logic one.
TDUF is cleared by reading the ERROR register with TDUF set. TDUF is also cleared on reset, while the CONTROL.JE bit is reset or while the CONTROL.JDIS bit is set.
0: No transmitter data underflow condition occurred
1: Transmitter data underflow condition occurred

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

Bit 5 = **RDOF** *Receiver Data Overflow*
The RDOF gets set to a logic one if the data in the RXDATA register has not been read and new data is ready to be transferred to the RXDATA register. The old RXDATA information is lost since it is overwritten with new data.
RDOF is cleared by reading the ERROR register with RDOF set, while the CONTROL.JE bit is reset or while the CONTROL.JDIS bit is set, or on reset.
0: No receiver data overflow condition occurred
1: Receiver data overflow condition occurred

Bit 4 = **TRA** *Transmit Request Aborted*
The TRA gets set to a logic one if a transmit opcode is aborted by the JBLPD state machine. Many conditions may cause a TRA. They are explained in the transmit opcode section. If the TRA bit gets set after a TXOP write, then a transmit is not attempted, and the TRDY bit is not cleared.
If a TRA error condition occurs, then the requested transmit is aborted, and the JBLPD peripheral takes appropriate measures as described under the TXOP register section.
TRA is cleared on reset, while the CONTROL.JE bit is reset or while the CONTROL.JDIS bit is set.
0: No transmission request aborted
1: Transmission request aborted

Bit 3 = **RBRK** *Received Break Symbol Flag*
The RBRK gets set to a logic one if a valid break (BRK) symbol is detected from the filtered VPWI pin. A Break received from the J1850 bus will cancel queued transmits of all types. The RBRK bit remains set as long as the break character is detected from the VPWI. Reads of the ERROR register will not clear the RBRK bit as long as a break character is being received. Once the break character is gone, a final read of the ERROR register clears this bit.
An RBRK error occurs once for a frame if it is received during a frame. Afterwards, the receiver is disabled from receiving information (other than the break) until an EOFM symbol is received.
RBRK bit is cleared on reset, while the CONTROL.JE bit is reset or while the CONTROL.JDIS bit is set.
The RBRK bit can be used to detect J1850 bus shorted high conditions. If RBRK is read as a logic high multiple times before an EOFM occurs, then a possible bus shorted high condition exists. The user program can take appropriate measures to test the bus if this condition occurs. Note that this bit does not necessarily clear when ERROR is read.
0: No valid Break symbol received
1: Valid Break symbol received

Bit 2 = **CRCE** *Cyclic Redundancy Check Error*
The receiver section always keeps a running tab of the CRC of all data bytes received from the VPWI since the last EOD symbol. The CRC check is performed when a valid EOD symbol is received both after a message string (subsequent to an SOF symbol) and after an IFR3 string (subsequent to an NB0 symbol). If the received CRC check fails, then the CRCE bit is set to a logic one. CRC errors are inhibited if the JBLPD peripheral is in the "sleep or filter and NOT presently transmitting" mode. A CRC error occurs once for a frame. Afterwards, the receiver is disabled until an EOFM symbol is received and queued transmits for the present frame are cancelled (but the TRA bit is not set). CRCE is cleared when ERROR is read. It is also cleared while the CONTROL.JE bit is reset or while the CONTROL.JDIS bit is set, or on reset.
0: No CRC error detected
1: CRC error detected

Bit 1 = **IFD** *Invalid Frame Detect*
The IFD bit gets set when the following conditions are detected from the filtered VPWI pin:

– An SOF symbol is received after an EOD minimum, but before an EOF minimum.
– An SOF symbol is received when expecting data bits.
– If NFL = 0 and a message frame greater than 12 bytes (i.e. 12 bytes plus one bit) has been received in one frame.
– An EOD minimum time has elapsed when data bits are expected.
– A logic 0 or 1 symbol is received (active for Tv1 or Tv2) when an SOF was expected.
– The second EODM symbol received in a frame is NOT followed directly by an EOFM symbol.

IFD errors are inhibited if the JBLPD peripheral is in the "sleep or filter and NOT presently transmitting" mode. An IFD error occurs once for a frame. Afterwards, the receiver is disabled until an EOFM symbol is received, and queued transmits for the present frame are cancelled (but the TRA bit is not set). IFD is cleared when ERROR is read. It is also cleared while the CONTROL.JE bit is reset or while the CONTROL.JDIS bit is set or on reset.
0: No invalid frame detected
1: Invalid frame detected

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

Bit 0 = **IBD** *Invalid Bit Detect.*
The IBD bit gets set whenever the receiver detects that the filtered VPWI pin was not fixed in a state long enough to reach the minimum valid symbol time of Tv1 (or 35 µs). Any timing event less than 35 µs (and, of course, > 7 µs since the VPWI digital filter will not allow pulses less than this through its filter) is considered as noise and sets the IBD accordingly. At this point the JBLPD peripheral will cease transmitting and receiving any information until a valid EOF symbol is received.
IBD errors are inhibited if the JBLPD peripheral is in the "sleep or filter and NOT presently transmitting" mode. An IBD error occurs once for a frame. Afterwards, the receiver is disabled until an EOFM symbol is received, and queued transmits for the present frame are cancelled (but the TRA bit is not set).
IBD is cleared when ERROR is read. Note that if an invalid bit is detected during a bus idle condition, the IBD flag gets set and a new EOFmin must be seen after the invalid bit before commencing to receive again. IBD is also cleared while the CONTROL.JE bit is reset or while the CONTROL.JDIS bit is set and on reset.
0: No invalid bit detected
1: Invalid bit detected

JBLPD INTERRUPT VECTOR REGISTER (IVR)

R248- Read/Write (except bits 2:1)
Register Page: 23
Reset Value: xxxx xxx0 (xxh)

7							0
V7	V6	V5	V4	V3	EV2	EV1	-

Bit 7:3 = **V[7:3]** *Interrupt Vector Base Address.*
User programmable interrupt vector bits.

Bit 2:1 = **EV[2:1]** *Encoded Interrupt Source (Read Only).*
EV2 and EV1 are set by hardware according to the interrupt source, given in Table 8 (refer to the Status register bits description about the explanation of the meaning of the interrupt sources)

Table 52. Interrupt Sources

EV2	EV1	Interrupt Sources
0	0	ERROR, TLA
0	1	EODM, EOFM
1	0	RDRF, REOB
1	1	TRDY, TEOB

Bit 0 = *Reserved.*

JBLPD PRIORITY LEVEL REGISTER (PRLR)

R249- Read/Write
Register Page: 23
Reset Value: 0001 0000 (10h)

7							0
PRL2	PRL1	PRL0	SLP	-	-	REOBP	TEOBP

Bit 7:5 = **PRL[2:0]** *Priority level bits*
The priority with respect to the other peripherals and the CPU is encoded with these three bits. The value of "0" has the highest priority, the value "7" has no priority. After the setting of this priority level, the priorities between the different Interrupt sources and DMA of the JBLPD peripheral is hardware defined (refer to the "Status register" bits description, the "Interrupts Management" and the section about the explanation of the meaning of the interrupt sources).
Depending on the value of the OPTIONS.DMASUSP bit, the DMA transfers can or cannot be suspended by an ERROR or TLA event. Refer to the description of DMASUSP bit.

Table 53. Internal Interrupt and DMA Priorities without DMA suspend mode

Priority Level	Event Sources
Higher Priority	TX-DMA
	RX-DMA
	ERROR, TLA
	EODM, EOFM
	RDRF, REOB
Lower Priority	TRDY, TEOB

Table 54. Internal Interrupt and DMA Priorities with DMA suspend mode

Priority Level	Event Sources
Higher Priority	ERROR, TLA
	TX-DMA
	RX-DMA
	EODM, EOFM
	RDRF, REOB
Lower Priority	TRDY, TEOB

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

Bit 4 = **SLP** *Receiver Sleep Mode.*
The SLP bit is written to one when the user program does not want to receive any data from the JBLPD VPWI pin until an EOFM symbol occurs. This mode is usually set when a message is received that the user does not require - including messages that the JBLPD is transmitting.
If the JBLPD is not transmitting and is in Sleep mode, no data is transferred to the RXDATA register, the RDRF flag does not get set, and errors associated with received data (RDOF, CRCE, IFD, IBD) do not get set. Also, the EODM flag will not get set.
If the JBLPD peripheral is transmitting and is in sleep mode, no data is transferred to the RXDATA register, the RDRF flag does not get set and the RDOF error flag is inhibited. The CRCE, IFD, and IBD flags, however, will NOT be inhibited while transmitting in sleep mode.
The SLP bit cannot be written to zero by the user program. The SLP bit is set on reset or TTO getting set, and it will stay set upon JE getting set until an EOFM symbol is received.
The SLP gets cleared on reception of an EOF or a Break symbol. SLP is set while CONTROL.JE is reset and while CONTROL.JDIS is set.
0: The JBLPD is not in Sleep Mode
1: The JBLPD is in Sleep Mode

Bit 3:2 = *Reserved.*

Bit 1 = **REOP** *Receiver DMA End Of Block Pending*.
This bit is set after a receiver DMA cycle to mark the end of a block of data. An interrupt request is performed if the RDRF_M bit of the IMR register is set. REOBP should be reset by software in order to avoid undesired interrupt routines, especially in initialisation routine (after reset) and after entering the End Of Block interrupt routine.
Writing "0" in this bit will cancel the interrupt request.
This bit is reset when the CONTROL.JDIS bit is set at least for 6 MCU clock cycles (3 NOPs).

Note: When the REOBP flag is set, the RXD_M bit is reset by hardware.

Note: REOBP can only be written to "0".

Bit 0 = **TEOP** *Transmitter DMA End Of Block Pending*.
This bit is set after a transmitter DMA cycle to mark the end of a block of data. An interrupt request is performed if the TRDY_M bit of the IMR register is set. TEOBP should be reset by software in order to avoid undesired interrupt routines, especially in initialisation routine (after reset) and after entering the End Of Block interrupt routine.
Writing "0" in this bit will cancel the interrupt request.
This bit is reset when the CONTROL.JDIS bit is set at least for 6 MCU clock cycles (3 NOPs).

Note: When the TEOBP flag is set, the TXD_M bit is reset by hardware.

Note: TEOBP can only be written to "0".

JBLPD INTERRUPT MASK REGISTER (IMR)
R250 - Read/Write
Register Page: 23
Reset Value: 0000 0000 (00h)

7							0
ERR_M	TRDY_M	RDRF_M	TLA_M	RXD_M	EODM_M	EOFM_M	TXD_M

To enable an interrupt source to produce an interrupt request, the related mask bit must be set. When these bits are reset, the related Interrupt Pending bit can not generate an interrupt.

Note: This register is forced to its reset value if the CONTROL.JDIS bit is set at least for 6 clock cycles (3 NOPs). If the JDIS bit is set for a shorter time, the bits could be reset or not reset.

Bit 7 = **ERR_M** *Error Interrupt Mask bit.*
This bit enables the "error" interrupt source to generate an interrupt request.
This bit is reset if the CONTROL.JDIS bit is set at least for 6 clock cycles (3 NOPs).
0: Error interrupt source masked
1: Error interrupt source un-masked

Bit 6 = **TRDY_M** *Transmit Ready Interrupt Mask bit.*
This bit enables the "transmit ready" interrupt source to generate an interrupt request.
This bit is reset if the CONTROL.JDIS bit is set at least for 6 clock cycles (3 NOPs).
0: TRDY interrupt source masked
1: TRDY interrupt source un-masked

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

Bit 5 = **RDRF_M** *Receive Data Register Full Interrupt Mask bit.*
This bit enables the "receive data register full" interrupt source to generate an interrupt request.
This bit is reset if the CONTROL.JDIS bit is set at least for 6 clock cycles (3 NOPs).
0: RDRF interrupt source masked
1: RDRF interrupt source un-masked

Bit 4 = **TLA_M** *Transmitter Lost Arbitration Interrupt Mask bit.*
This bit enables the "transmitter lost arbitration" interrupt source to generate an interrupt request.
This bit is reset if the CONTROL.JDIS bit is set at least for 6 clock cycles (3 NOPs).
0: TLA interrupt source masked
1: TLA interrupt source un-masked

Bit 3 = **RXD_M** *Receiver DMA Mask bit.*
If this bit is "0" no receiver DMA request will be generated, and the RDRF bit, in the Status Register (STATUS), can request an interrupt. If RXD_M bit is set to "1" then the RDRF bit can request a DMA transfer. RXD_M is reset by hardware when the transaction counter value decrements to zero, that is when a Receiver End Of Block condition occurs (REOBP flag set).
This bit is reset if the CONTROL.JDIS bit is set at least for 6 clock cycles (3 NOPs).
0: Receiver DMA disabled
1: Receiver DMA enabled

Bit 2 = **EODM_M** *End of Data Minimum Interrupt Mask bit.*
This bit enables the "end of data minimum" interrupt source to generate an interrupt request.
This bit is reset if the CONTROL.JDIS bit is set at least for 6 clock cycles (3 NOPs).
0: EODM interrupt source mask
1: EODM interrupt source un-masked

Bit 1 = **EOFM_M** *End of Frame Minimum Interrupt Mask bit.*
This bit enables the "end of frame minimum" interrupt source to generate an interrupt request.
This bit is reset if the CONTROL.JDIS bit is set at least for 6 clock cycles (3 NOPs).
0: EOFM interrupt source masked
1: EOFM interrupt source un-masked

Bit 0 = **TXD_M** *Transmitter DMA Mask bit.*
If this bit is "0" no transmitter DMA request will be generated, and the TRDY bit, in the Status Register (STATUS), can request an interrupt. If TXD_M bit is set to "1" then the TRDY bit can request a DMA transfer. TXD_M is reset by hardware when the transaction counter value decrements to zero, that is when a Transmitter End Of Block condition occurs (TEOBP flag set).
This bit is reset if the CONTROL.JDIS bit is set at least for 6 clock cycles (3 NOPs).
0: Transmitter DMA disabled
1: Transmitter DMA enabled

JBLPD OPTIONS AND REGISTER GROUPS SELECTION REGISTER (OPTIONS)

R251- Read/Write
Register Page: 23
Reset Value: 0000 0000 (00h)

7							0
INPOL	NBSYMS	DMASUSP	LOOPB	RSEL3	RSEL2	RSEL1	RSEL0

Bit 7 = **INPOL** *VPWI Input Polarity Selector.*
This bit allows the selection of the polarity of the RX signal coming from the transceivers. Depending on the specific transceiver, the RX signal is inverted or not inverted respect the VPWO and the J1850 bus line.
0: VPWI input is inverted by the transceiver with respect to the J1850 line.
1: VPWI input is not inverted by the transceiver with respect to the J1850 line.

Bit 6 = **NBSYMS** *NB Symbol Form Selector.*
This bit allows the selection of the form of the Normalization Bits (NB0/NB1).
0: NB0 active long symbol (Tv2), NB1 active short symbol (Tv1)
1: NB0 active short symbol (Tv1), NB1 active long symbol (Tv2)

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

Bit 5 = **DMASUSP** *DMA Suspended Selector.*
If this bit is "0", JBLPD DMA has higher priority with respect to the Interrupts of the peripheral. DMA is performed even if an interrupt request is already scheduled or if the relative interrupt routine is in execution.
If the bit is "1", while the ERROR or TLA flag of the STATUS register are set, the DMA transfers are suspended. As soon as the flags are reset, the DMA transfers can be performed.
0: DMA not suspended
1: DMA suspended

Note: This bit has effect only on the priorities of the JBLPD peripheral.

Bit 4 = **LOOPB** *Local Loopback Selector.*
This bit allows the Local Loopback mode. When this mode is enabled (LOOPB=1), the VPWO output of the peripheral is sent to the VPWI input without inversions whereas the VPWO output line of the MCU is placed in the passive state. Moreover the VPWI input of the MCU is ignored by the peripheral. (Refer to Figure 9).
0: Local Loopback disabled
1: Local Loopback enabled

Note: When the LOOPB bit is set, **also the INPOL bit must be set** to obtain the correct management of the polarity.

Bit 3:0 = **RSEL[3:0]** *Registers Group Selection bits.*
These four bits are used to select one of the 9 groups of registers, each one composed of four registers that are stacked at the addresses from R252 (FCh) to R255 (FFh) of this register page (23). Unless the wanted registers group is already selected, to address a specific registers group, these bits must be correctly written.
This feature allows that 36 registers (4 DMA registers - RDADR, RDCPR, TDAPR, TDCPR - and 32 Message Filtering Registers - FREG[0:31]) are mapped using only 4 registers (here called Current Registers - CREG[3:0]).

Since the Message Filtering Registers (FREG[0:31]) are seldom read or written, it is suggested to always reset the RSEL[3:0] bits after accessing the FREG[0:31] registers. In this way the DMA registers are the current registers.

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

JBLPD CURRENT REGISTER 0 (CREG0)
R252- Read/Write
Register Page: 23
Reset Value: xxxx xxxx (xxh)

7							0
b7	b6	b5	b4	b3	b2	b1	b0

Depending on the RSEL[3:0] value of the OPTIONS register, this register is one of the following stacked registers: RDAPR, FREG0, FREG4, FREG8, FREG12, FREG16, FREG20, FREG24, FREG28.

JBLPD CURRENT REGISTER 1 (CREG1)
R253 - Read/Write
Register Page: 23
Reset Value: xxxx xxxx (xxh)

7							0
b7	b6	b5	b4	b3	b2	b1	b0

Depending on the RSEL[3:0] value of the OPTIONS register, this register is one of the following stacked registers: RDCPR, FREG1, FREG5, FREG9, FREG13, FREG17, FREG21, FREG25, FREG29.

JBLPD CURRENT REGISTER 2 (CREG2)
R254- Read/Write
Register Page: 23
Reset Value: xxxx xxxx (xxh)

7							0
b7	b6	b5	b4	b3	b2	b1	b0

Depending on the RSEL[3:0] value of the OPTIONS register, this register is one of the following stacked registers: TDAPR, FREG2, FREG6, FREG10, FREG14, FREG18, FREG22, FREG26, FREG30.

JBLPD CURRENT REGISTER 3 (CREG3)
R255- Read/Write
Register Page: 23
Reset Value: xxxx xxxx (xxh)

7							0
b7	b6	b5	b4	b3	b2	b1	b0

Depending on the RSEL[3:0] value of the OPTIONS register, this register is one of the following stacked registers: TDCPR, FREG3, FREG7, FREG11, FREG15, FREG19, FREG23, FREG27, FREG31.

Table 55. Stacked registers map

RSEL[3:0] Current Registers	0000b	1000b	1001b	1010b	1011b	1100b	1101b	1110b	1111b
CREG0	RDAPR	FREG0	FREG4	FREG8	FREG12	FREG16	FREG20	FREG24	FREG28
CREG1	RDCPR	FREG1	FREG5	FREG9	FREG13	FREG17	FREG21	FREG25	FREG29
CREG2	TDAPR	FREG2	FREG6	FREG10	FREG14	FREG18	FREG22	FREG26	FREG30
CREG3	TDCPR	FREG3	FREG7	FREG11	FREG15	FREG19	FREG23	FREG27	FREG31

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

10.8.7.2 Stacked Registers

See the description of the OPTIONS register to obtain more information on the map of the registers of this section.

JBLPD RECEIVER DMA ADDRESS POINTER REGISTER (RDAPR)
R252 - RSEL[3:0]=0000b
Register Page: 23
Reset Value: xxxx xxxx (xxh)

7							0
RA7	RA6	RA5	RA4	RA3	RA2	RA1	PS

To select this register, the RSEL[3:0] bits of the OPTIONS register must be reset

Bit 7:1 = **RA[7:1]** *Receiver DMA Address Pointer.*
RDAPR contains the address of the pointer (in the Register File) of the Receiver DMA data source when the DMA between the peripheral and the Memory Space is selected. Otherwise, when the DMA between the peripheral and Register File is selected, this register has no meaning.
See Section 0.1.6.2 for more details on the use of this register.

Bit 0 = **PS** *Memory Segment Pointer Selector.*
This bit is set and cleared by software. It is only meaningful if RDCPR.RF/MEM = 1.
0: The ISR register is used to extend the address of data received by DMA (see MMU chapter)
1: The DMASR register is used to extend the address of data received by DMA (see MMU chapter)

JBLPD RECEIVER DMA TRANSACTION COUNTER REGISTER (RDCPR)
R253 - RSEL[3:0]=0000b
Register Page: 23
Reset Value: xxxx xxxx (xxh)

7							0
RC7	RC6	RC5	RC4	RC3	RC2	RC1	RF/MEM

To select this register, the RSEL[3:0] bits of the OPTIONS register must be reset

Bit 7:1 = **RC[7:1]** *Receiver DMA Counter Pointer.*
RDCPR contains the address of the pointer (in the Register File) of the DMA receiver transaction counter when the DMA between Peripheral and Memory Space is selected. Otherwise, if the DMA between Peripheral and Register File is selected, this register points to a pair of registers that are used as DMA Address register and DMA Transaction Counter.
See Section 0.1.6.1 and Section 0.1.6.2 for more details on the use of this register.

Bit 0 = **RF/MEM** *Receiver Register File/Memory Selector.*
If this bit is set to "1", then the Register File will be selected as Destination, otherwise the Memory space will be used.
0: Receiver DMA with Memory space
1: Receiver DMA with Register File

JBLPD TRANSMITTER DMA ADDRESS POINTER REGISTER (TDAPR)
R254 - RSEL[3:0]=0000b
Register Page: 23
Reset Value: xxxx xxxx (xxh)

7							0
TA7	TA6	TA5	TA4	TA3	TA2	TA1	PS

To select this register, the RSEL[3:0] bits of the OPTIONS register must be reset

Bit 7:1 = **TA[7:1]** *Transmitter DMA Address Pointer.*
TDAPR contains the address of the pointer (in the Register File) of the Transmitter DMA data source when the DMA between the Memory Space and the peripheral is selected. Otherwise, when the DMA between Register File and the peripheral is selected, this register has no meaning.
See Section 0.1.6.2 for more details on the use of this register.

Bit 0 = **PS** *Memory Segment Pointer Selector.*
This bit is set and cleared by software. It is only meaningful if TDCPR.RF/MEM = 1.
0: The ISR register is used to extend the address of data transmitted by DMA (see MMU chapter)
1: The DMASR register is used to extend the address of data transmitted by DMA (see MMU chapter)

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

JBLPD TRANSMITTER DMA TRANSACTION COUNTER REGISTER (TDCPR)

R255 - RSEL[3:0]=0000b
Register Page: 23
Reset Value: xxxx xxxx (xxh)

7							0
TC7	TC6	TC5	TC4	TC3	TC2	TC1	RF/MEM

To select this register, the RSEL[3:0] bits of the OPTIONS register must be reset

Bit 7:1 = **TC[7:1]** *Transmitter DMA Counter Pointer.*
RDCPR contains the address of the pointer (in the Register File) of the DMA transmitter transaction counter when the DMA between Memory Space and peripheral is selected. Otherwise, if the DMA between Register File and peripheral is selected, this register points to a pair of registers that are used as DMA Address register and DMA Transaction Counter.
See Section 0.1.6.1 and Section 0.1.6.2 for more details on the use of this register.

Bit 0 = **RF/MEM** *Transmitter Register File/Memory Selector.*
If this bit is set to "1", then the Register File will be selected as Destination, otherwise the Memory space will be used.
0: Transmitter DMA with Memory space
1: Transmitter DMA with Register File

JBLPD MESSAGE FILTERING REGISTERS (FREG[0:31])

R252/R253/R254/R255 - RSEL[3]=1
Register Page: 23
Reset Value: xxxx xxxx (xxh)

Register	7							0
FREG0	F_07	F_06	F_05	F_04	F_03	F_02	F_01	F_00
FREG1	F_0F	F_0E	F_0D	F_0C	F_0B	F_0A	F_09	F_08
FREG2	F_17	F_16	F_15	F_14	F_13	F_12	F_11	F_10
FREG3	F_1F	F_1E	F_1D	F_1C	F_1B	F_1A	F_19	F_18
FREG4	F_27	F_26	F_25	F_24	F_23	F_22	F_21	F_20
FREG5	F_2F	F_2E	F_2D	F_2C	F_2B	F_2A	F_29	F_28
FREG6	F_37	F_36	F_35	F_34	F_33	F_32	F_31	F_30
FREG7	F_3F	F_3E	F_3D	F_3C	F_3B	F_3A	F_39	F_38
FREG8	F_47	F_46	F_45	F_44	F_43	F_42	F_41	F_40
FREG9	F_4F	F_4E	F_4D	F_4C	F_4B	F_4A	F_49	F_48
FREG10	F_57	F_56	F_55	F_54	F_53	F_52	F_51	F_50
FREG11	F_5F	F_5E	F_5D	F_5C	F_5B	F_5A	F_59	F_58
FREG12	F_67	F_66	F_65	F_64	F_63	F_62	F_61	F_60
FREG13	F_6F	F_6E	F_6D	F_6C	F_6B	F_6A	F_69	F_68
FREG14	F_77	F_76	F_75	F_74	F_73	F_72	F_71	F_70
FREG15	F_7F	F_7E	F_7D	F_7C	F_7B	F_7A	F_79	F_78
FREG16	F_87	F_86	F_85	F_84	F_83	F_82	F_81	F_80
FREG17	F_8F	F_8E	F_8D	F_8C	F_8B	F_8A	F_89	F_88
FREG18	F_97	F_96	F_95	F_94	F_93	F_92	F_91	F_90
FREG19	F_9F	F_9E	F_9D	F_9C	F_9B	F_9A	F_99	F_98
FREG20	F_A7	F_A6	F_A5	F_A4	F_A3	F_A2	F_A1	F_A0
FREG21	F_AF	F_AE	F_AD	F_AC	F_AB	F_AA	F_A9	F_A8
FREG22	F_B7	F_B6	F_B5	F_B4	F_B3	F_B2	F_B1	F_B0
FREG23	F_BF	F_BE	F_BD	F_BC	F_BB	F_BA	F_B9	F_B8
FREG24	F_C7	F_C6	F_C5	F_C4	F_C3	F_C2	F_C1	F_C0
FREG25	F_CF	F_CE	F_CD	F_CC	F_CB	F_CA	F_C9	F_C8
FREG26	F_D7	F_D6	F_D5	F_D4	F_D3	F_D2	F_D1	F_D0
FREG27	F_DF	F_DE	F_DD	F_DC	F_DB	F_DA	F_D9	F_D8
FREG28	F_E7	F_E6	F_E5	F_E4	F_E3	F_E2	F_E1	F_E0
FREG29	F_EF	F_EE	F_ED	F_EC	F_EB	F_EA	F_E9	F_E8
FREG30	F_F7	F_F6	F_F5	F_F4	F_F3	F_F2	F_F1	F_F0
FREG31	F_FF	F_FE	F_FD	F_FC	F_FB	F_FA	F_F9	F_F8

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

These registers are structured in eight groups of four registers. The user can gain access to these registers programming the RSEL[2:0] bits of the OPTIONS register while the RSEL[3] bit of the same register must be placed at 1. In this way the user can select the group where the registers that he/she wants to use are placed. See the description of OPTIONS register for the correspondence between registers and the values of RSEL[2:0] bits (See Table 11).

From the functional point of view, the FREG[0]-FREG[31] registers can be seen as an array of 256 bits involved in the J1850 received message filtering system.

The first byte received in a frame (following a valid received SOF character) is an Identifier (I.D.) byte. It is used by the JBLPD peripheral as the address of the 256 bits array.

If the bit of the array correspondent to the I.D. byte is set, then the byte is transferred to the RXDATA register and the RDRF flag is set. Also, every other data byte received in this frame is transferred to the RXDATA register unless the JBLPD peripheral is put into sleep mode setting the SLP bit.

If the bit of the array correspondent to the I.D. byte is clear, then the transfer of this byte as well as any byte for the balance of this frame is inhibited, and the RDRF bit remains cleared.

The bit 0 of the FREG[0] register (FREG[0].0 - marked as F_00 in the previous table) corresponds to the I.D. byte equal to 00h while the bit 7 of the FREG[31] register (FREG[31].7 - marked as F_FF in the previous table) corresponds to the I.D. byte equal to FFh.

Note: The FREG registers are undefined upon reset. Because of this, it is strongly recommended that the contents of these registers has to be defined before JE is set for the first time after reset. Otherwise, unpredictable results may occur.

J1850 BYTE LEVEL PROTOCOL DECODER (Cont'd)

Register	Address	7							0
STATUS reset value	F0h	ERR 0	TRDY 1	RDRF 0	TLA 0	RDT 0	EODM 0	EOFM 0	IDLE 0
TXDATA reset value	F1h	TXD7 x	TXD6 x	TXD5 x	TXD4 x	TXD3 x	TXD2 x	TXD1 x	TXD0 x
RXDATA reset value	F2h	RXD7 x	RXD6 x	RXD5 x	RXD4 x	RXD3 x	RXD2 x	RXD1 x	RXD0 x
TXOP reset value	F3h	MLC3 0	MLC2 0	MLC1 0	MLC0 0	- 0	OP2 0	OP1 0	OP0 0
CLKSEL reset value	F4h	4X 0	- 0	FREQ5 0	FREQ4 0	FREQ3 0	FREQ2 0	FREQ1 0	FREQ0 0
CONTROL reset value	F5h	JE 0	JDIS 1	NFL 0	JDLY4 0	JDLY3 0	JDLY2 0	JDLY1 0	JDLY0 0
PADDR reset value	F6h	ADR7 x	ADR6 x	ADR5 x	ADR4 x	ADR3 x	ADR2 x	ADR1 x	ADR0 x
ERROR reset value	F7h	TTO 0	TDUF 0	RDOF 0	TRA 0	RBRK 0	CRCE 0	IFD 0	IBD 0
IVR reset value	F8h	V7 x	V6 x	V5 x	V4 x	V3 x	EV2 x	EV1 x	- 0
PRLR reset value	F9h	PRL2 0	PRL1 0	PRL0 0	SLP 1	- 0	- 0	REOBP 0	TEOBP 0
IMR reset value	FAh	ERR_M 0	TRDY_M 0	RDRF_M 0	TLA_M 0	RXD_M 0	EODM_M 0	EOFM_M 0	TXD_M 0
OPTIONS reset value	FBh	INPOL 0	NBSYMS 0	DMASUSP 0	LOOPB 0	RSEL3 0	RSEL2 0	RSEL1 0	RSEL0 0
CREG0 reset value	FCh	b7 x	b6 x	b5 x	b4 x	b3 x	b2 x	b1 x	b0 x
CREG1 reset value	FDh	b7 x	b6 x	b5 x	b4 x	b3 x	b2 x	b1 x	b0 x
CREG2 reset value	FEh	b7 x	b6 x	b5 x	b4 x	b3 x	b2 x	b1 x	b0 x
CREG3 reset value	FFh	b7 x	b6 x	b5 x	b4 x	b3 x	b2 x	b1 x	b0 x

10.9 EIGHT-CHANNEL ANALOG TO DIGITAL CONVERTER (A/D)

10.9.1 Introduction

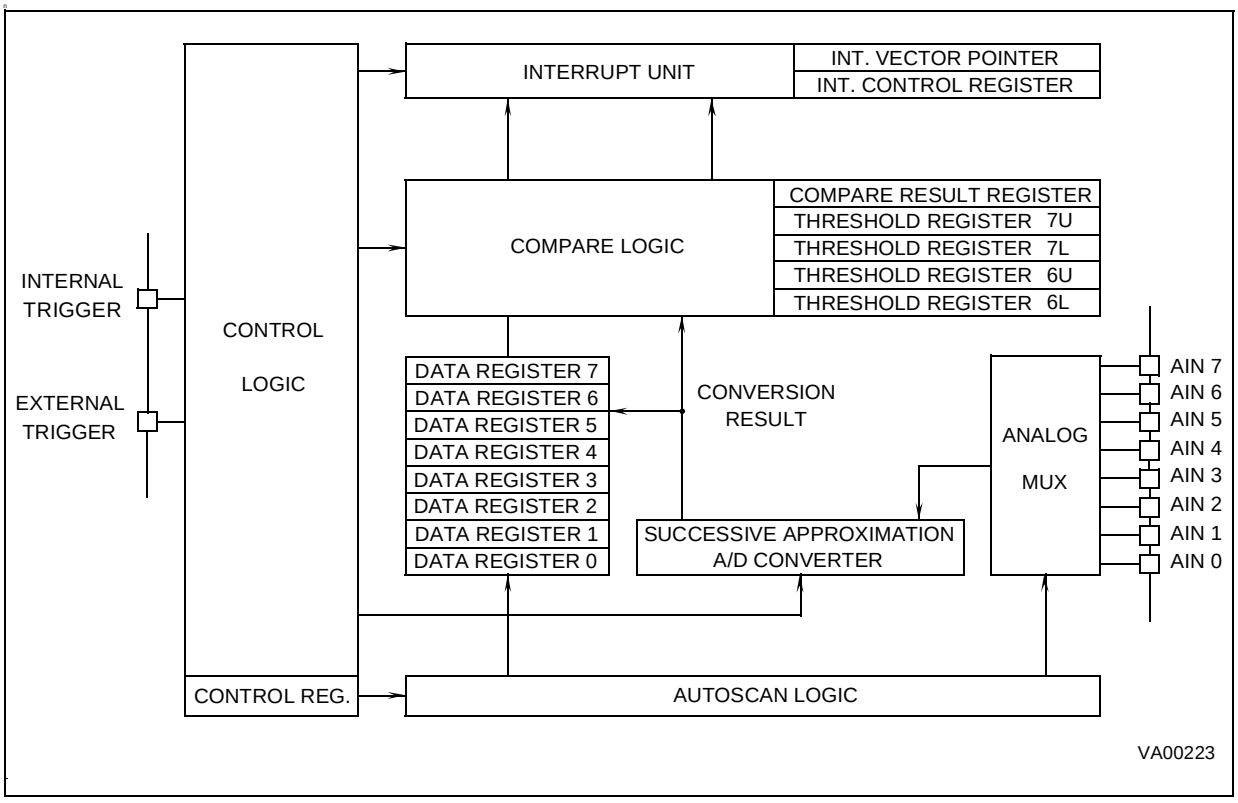
The 8-Channel Analog to Digital Converter (A/D) comprises an input multiplex channel selector feeding a successive approximation converter. Conversion requires 138 INTCLK cycles (of which 84 are required for sampling), conversion time is thus a function of the INTCLK frequency; for instance, for a 20MHz clock rate, conversion of the selected channel requires 6.9µs. This time includes the 4.2µs required by the built-in Sample and Hold circuitry, which minimizes the need for external components and allows quick sampling of the signal to minimise warping and conversion error. Conversion resolution is 8 bits, with ±1 LSB maximum error in the input range between V_{SS} and the analog V_{DD} reference.

The converter uses a fully differential analog input configuration for the best noise immunity and precision performance. Two separate supply references are provided to ensure the best possible supply noise rejection. In fact, the converted digital value, is referred to the analog reference voltage which determines the full scale converted value. Naturally**,** Analog and Digital V_{SS} MUST be common. If analog supplies are not present, input reference voltages are referred to the digital ground and supply.

Up to 8 multiplexed Analog Inputs are available, depending on the specific device type. A group of signals can be converted sequentially by simply programming the starting address of the first analog channel to be converted and with the AUTOSCAN feature.

Two Analog Watchdogs are provided, allowing continuous hardware monitoring of two input channels. An Interrupt request is generated whenever the converted value of either of these two analog inputs is outside the upper or lower programmed threshold values. The comparison result is stored in a dedicated register.

Figure 121. Block Diagram



ANALOG TO DIGITAL CONVERTER (Cont'd)

Single and continuous conversion modes are available. Conversion may be triggered by an external signal or, internally, by the Multifunction Timer.

A Power-Down programmable bit allows the A/D to be set in low-power idle mode.

The A/D's Interrupt Unit provides two maskable channels (Analog Watchdog and End of Conversion) with hardware fixed priority, and up to 7 programmable priority levels.

CAUTION: A/D INPUT PIN CONFIGURATION

The input Analog channel is selected by using the I/O pin Alternate Function setting (PXC2, PXC1, PXC0 = 1,1,1) as described in the I/O ports section. The I/O pin configuration of the port connected to the A/D converter is modified in order to prevent the analog voltage present on the I/O pin from causing high power dissipation across the input buffer. Deselected analog channels should also be maintained in Alternate function configuration for the same reason.

10.9.2 Functional Description

10.9.2.1 Operating Modes

Two operating modes are available: Continuous Mode and Single Mode. To enter one of these modes it is necessary to program the CONT bit of the Control Logic Register. Continuous Mode is selected when CONT is set, while Single Mode is selected when CONT is reset.

Both modes operate in AUTOSCAN configuration, allowing sequential conversion of the input channels. The number of analog inputs to be converted may be set by software, by setting the number of the first channel to be converted into the Control Register (SC2, SC1, SC0 bits). As each conversion is completed, the channel number is automatically incremented, up to channel 7. For example, if SC2, SC1, SC0 are set to 0,1,1, conversion will proceed from channel 3 to channel 7, whereas, if SC2, SC1, SC0 are set to 1,1,1, only channel 7 will be converted.

When the ST bit of the Control Logic Register is set, either by software or by hardware (by an internal or external synchronisation trigger signal), the analog inputs are sequentially converted (from the first selected channel up to channel 7) and the results are stored in the relevant Data Registers.

In **Single Mode** (CONT = "0"), the ST bit is reset by hardware following conversion of channel 7; an End of Conversion (ECV) interrupt request is issued and the A/D waits for a new start event.

In **Continuous Mode** (CONT = "1"), a continuous conversion flow is initiated by the start event. When conversion of channel 7 is complete, conversion of channel 's' is initiated (where 's' is specified by the setting of the SC2, SC1 and SC0 bits); this will continue until the ST bit is reset by software. In all cases, an ECV interrupt is issued each time channel 7 conversion ends.

When channel 'i' is converted ('s' <'i' <7), the related Data Register is reloaded with the new conversion result and the previous value is lost. The End of Conversion (ECV) interrupt service routine can be used to save the current values before a new conversion sequence (so as to create signal sample tables in the Register File or in Memory).

10.9.2.2 Triggering and Synchronisation

In both modes, conversion may be triggered by internal or external conditions; externally this may be tied to EXTRG, as an Alternate Function input on an I/O port pin, and internally, it may be tied to INTRG, generated by a Multifunction Timer peripheral. Both external and internal events can be separately masked by programming the EXTG/ INTG bits of the Control Logic Register (CLR). The events are internally ORed, thus avoiding potential hardware conflicts. However, the correct procedure is to enable only one alternate synchronisation condition at any time.

The effect either of these synchronisation modes is to set the ST bit by hardware. This bit is reset, in Single Mode only, at the end of each group of conversions. In Continuous Mode, all trigger pulses after the first are ignored.

The synchronisation sources must be at a logic low level for at least the duration of one INTCLK cycle and, in Single Mode, the period between trigger pulses must be greater than the total time required for a group of conversions. If a trigger occurs when the ST bit is still set, i.e. when conversion is still in progress, it will be ignored.

On devices where two A/D Converters are present they can be triggered from the same source.

Converter	External Trigger	On Chip Event (Internal trigger)
A/D 0	EXTRG pin	MFT 0
A/D 1		

10.9.2.3 Analog Watchdogs

Two internal Analog Watchdogs are available for highly flexible automatic threshold monitoring of external analog signal levels.

ANALOG TO DIGITAL CONVERTER (Cont'd)

Analog channels 6 and 7 monitor an acceptable voltage level window for the converted analog inputs. The external voltages applied to inputs 6 and 7 are considered normal while they remain below their respective Upper thresholds, and above or at their respective Lower thresholds.

When the external signal voltage level is greater than, or equal to, the upper programmed voltage limit, or when it is less than the lower programmed voltage limit, a maskable interrupt request is generated and the Compare Results Register is updated in order to flag the threshold (Upper or Lower) and channel (6 or 7) responsible for the interrupt. The four threshold voltages are user programmable in dedicated registers (08h to 0Bh) of the A/D register page. Only the 4 MSBs of the Compare Results Register are used as flags (the 4 LSBs always return "1" if read), each of the four MSBs being associated with a threshold condition.

Following a hardware reset, these flags are reset. During normal A/D operation, the CRR bits are set, in order to flag an out of range condition and are automatically reset by hardware after a software reset of the Analog Watchdog Request flag in the AD_ICR Register.

10.9.2.4 Power Down Mode

Before enabling an A/D conversion, the POW bit of the Control Logic Register must be set; this must be done at least 60µs before the first conversion start, in order to correctly bias the analog section of the converter circuitry.

When the A/D is not required, the POW bit may be reset in order to reduce the total power consumption. This is the reset configuration, and this state is also selected automatically when the ST9 is placed in Halt Mode (following the execution of the `halt` instruction).

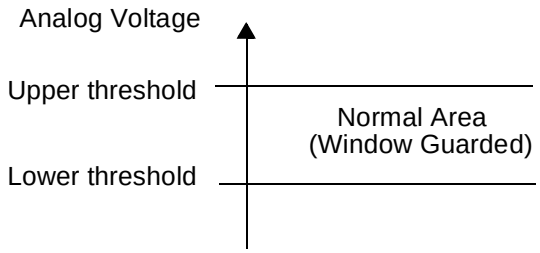
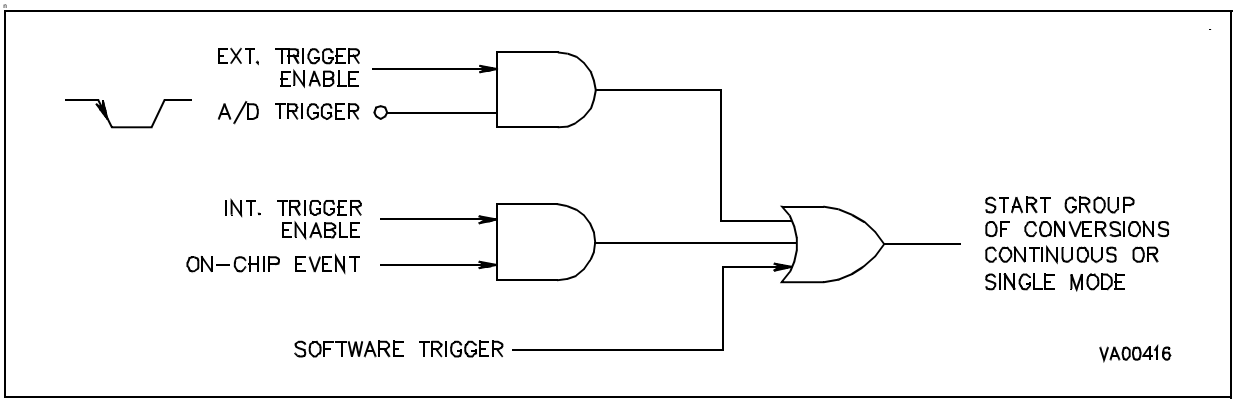
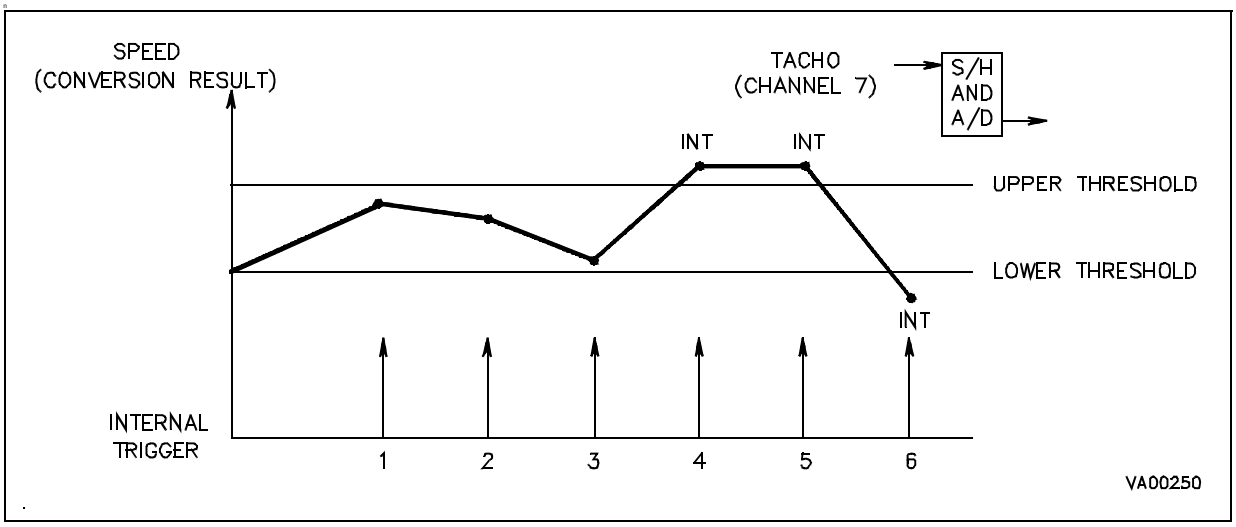


Figure 122. A/D Trigger Source



ANALOG TO DIGITAL CONVERTER (Cont'd)

Figure 123. Application Example: Analog Watchdog used in Motorspeed Control



10.9.3 Interrupts

The A/D provides two interrupt sources:

– End of Conversion

– Analog Watchdog Request

The A/D Interrupt Vector Register (AD_IVR) provides hardware generated flags which indicate the interrupt source, thus allowing automatic selection of the correct interrupt service routine.

	7							0	
Analog Watchdog Request	X	X	X	X	X	X	0	0	Lower Word Address

	7							0	
End of Conv. Request	X	X	X	X	X	X	1	0	Upper Word Address

The A/D Interrupt vector should be programmed by the User to point to the first memory location in the Interrupt Vector table containing the base address of the four byte area of the interrupt vector table in which the address of the A/D interrupt service routines are stored.

The Analog Watchdog Interrupt Pending bit (AWD, AD_ICR.6), is automatically set by hardware whenever any of the two guarded analog inputs go out of range. The Compare Result Register (CRR) tracks the analog inputs which exceed their programmed thresholds.

When two requests occur simultaneously, the Analog Watchdog Request has priority over the End of Conversion request, which is held pending.

The Analog Watchdog Request requires the user to poll the Compare Result Register (CRR) to determine which of the four thresholds has been exceeded. The threshold status bits are set to flag an out of range condition, and are automatically reset by hardware after a software reset of the Analog Watchdog Request flag in the AD_ICR Register. The interrupt pending flags, ECV and AWD, should be reset by the user within the interrupt service routine. Setting either of these two bits by software will cause an interrupt request to be generated.

10.9.3.1 Register Mapping

It is possible to have two independent A/D converters in the same device. In this case they are named A/D 0 and A/D 1. If the device has one A/D converter it uses the register addresses of A/D 0. The register pages are the following:

A/Dn	Register Page
A/D 0	63
A/D 1	61

ANALOG TO DIGITAL CONVERTER (Cont'd)

10.9.4 Register Description

DATA REGISTERS (DiR)

The conversion results for the 8 available channels are loaded into the 8 Data registers following conversion of the corresponding analog input.

CHANNEL 0 DATA REGISTER (D0R)
R240 - Read/Write
Register Page: 63
Reset Value: undefined

7							0
D0.7	D0.6	D0.5	D0.4	D0.3	D0.2	D0.1	D0.0

Bit 7:0 = **D0.[7:0]**: *Channel 0 Data.*

CHANNEL 1 DATA REGISTER (D1R)
R241 - Read/Write
Register Page: 63
Reset Value: undefined

7							0
D1.7	D1.6	D1.5	D1.4	D1.3	D1.2	D1.1	D1.0

Bit 7:0 = **D1.[7:0]**: *Channel 1 Data.*

CHANNEL 2 DATA REGISTER (D2R)
R242 - Read/Write
Register Page: 63
Reset Value: undefined

7							0
D2.7	D2.6	D2.5	D2.4	D2.3	D2.2	D2.1	D2.0

Bit 7:0 = **D2.[7:0]**: *Channel 2 Data.*

CHANNEL 3 DATA REGISTER (D3R)
R243 - Read/Write
Register Page: 63
Reset Value: undefined

Bit 7:0 = **D3.[7:0]**: *Channel 3 Data.*

7							0
D3.7	D3.6	D3.5	D3.4	D3.3	D3.2	D3.1	D3.0

CHANNEL 4 DATA REGISTER (D4R)
R244 - Read/Write
Register Page: 63
Reset Value: undefined

7							0
D4.7	D4.6	D4.5	D4.4	D4.3	D4.2	D4.1	D4.0

Bit 7:0 = **D4.[7:0]**: *Channel 4 Data*

CHANNEL 5 DATA REGISTER (D5R)
R245 - Read/Write
Register Page: 63
Reset Value: undefined

7							0
D5.7	D5.6	D5.5	D5.4	D5.3	D5.2	D5.1	D5.0

Bit 7:0 = **D5.[7:0]**: *Channel 5 Data.*

CHANNEL 6 DATA REGISTER (D6R)
R246 - Read/Write
Register Page: 63
Reset Value: undefined

7							0
D6.7	D6.6	D6.5	D6.4	D6.3	D6.2	D6.1	D6.0

Bit 7:0 = **D6.[7:0]**: *Channel 6 Data*

CHANNEL 7 DATA REGISTER (D7R)
R247 - Read/Write
Register Page: 63
Reset Value: undefined

7							0
D7.7	D7.6	D7.5	D7.4	D7.3	D7.2	D7.1	D7.0

ANALOG TO DIGITAL CONVERTER (Cont'd)

CHANNEL 6 LOWER THRESHOLD REGISTER (LT6R)
R248 - Read/Write
Register Page: 63
Reset Value: undefined

7							0
LT6.7	LT6.6	LT6.5	LT6.4	LT6.3	LT6.2	LT6.1	LT6.0

Bit 7:0 = **LT6.[7:0]:** *Channel 6 Lower Threshold*
User-defined lower threshold value for Channel 6, to be compared with the conversion results.

CHANNEL 7 LOWER THRESHOLD REGISTER (LT7R)
R249 - Read/Write
Register Page: 63
Reset Value: undefined

7							0
LT7.7	LT7.6	LT7.5	LT7.4	LT7.3	LT7.2	LT7.1	LT7.0

Bit 7:0 = **LT7.[7:0]:** *Channel 7 Lower Threshold.*
User-defined lower threshold value for Channel 7, to be compared with the conversion results.

CHANNEL 6 UPPER THRESHOLD REGISTER (UT6R)
R250 - Read/Write
Register Page: 63
Reset Value: undefined

7							0
UT6.7	UT6.6	UT6.5	UT6.4	UT6.3	UT6.2	UT6.1	UT6.0

Bit 7:0 = **UT6.[7:0]**: *Channel 6 Upper Threshold value.*
User-defined upper threshold value for Channel 6, to be compared with the conversion results.

CHANNEL 7 UPPER THRESHOLD REGISTER (UT7R)
R251 - Read/Write
Register Page: 63
Reset Value: undefined

7							0
UT7.7	UT7.6	UT7.5	UT7.4	UT7.3	UT7.2	UT7.1	UT7.0

Bit 7:0 = **UT7.[7:0]**: *Channel 7 Upper Threshold value*
User-defined upper threshold value for Channel 7, to be compared with the conversion results.

COMPARE RESULT REGISTER (CRR)
R252 - Read/Write
Register Page: 63
Reset Value: 0000 1111 (0Fh)

7							0
C7U	C6U	C7L	C6L	1	1	1	1

These bits are set by hardware and cleared by software.

Bit 7 = **C7U**: *Compare Reg 7 Upper threshold*
0: Threshold not reached
1: Channel 7 converted data is greater than or equal to UT7R threshold register value.

Bit 6 = **C6U**: *Compare Reg 6Upper threshold*
0: Threshold not reached
1: Channel 6 converted data is greater than or equal to UT6R threshold register value.

Bit 5 = **C7L**: *Compare Reg 7 Lower threshold*
0: Threshold not reached
1: Channel 7 converted data is less than the LT7R threshold register value.

Bit 4 = **C6L**: *Compare Reg 6 Lower threshold*
0: Threshold not reached
1: Channel 6 converted data is less than the LT6R threshold register value.

Bit 3:0 = Reserved, returns "1" when read.

Note: Any software reset request generated by writing to the AD_ICR, will also cause all the compare status bits to be cleared.

ANALOG TO DIGITAL CONVERTER (Cont'd)

CONTROL LOGIC REGISTER (CLR)

The Control Logic Register (CLR) manages the A/D converter logic. Writing to this register will cause the current conversion to be aborted and the autoscan logic to be re-initialized.

CONTROL LOGIC REGISTER (CLR)
R253 - Read/Write
Register Page: 63
Reset Value: 0000 0000 (00h)

7							0
SC2	SC1	SC0	EXTG	INTG	POW	CONT	ST

Bit 7:5 = **SC[2:0]:** *Start Conversion Address*.
These 3 bits define the starting analog input channel (Autoscan mode). The first channel addressed by SC[2:0] is converted, then the channel number is incremented for the successive conversion, until channel 7 (111) is converted. When SC2, SC1 and SC0 are all set, only channel 7 will be converted.

Bit 4 = **EXTG**: *External Trigger Enable*.
This bit is set and cleared by software.
0: External trigger disabled.
1: External trigger enabled. Allows a conversion sequence to be started on the subsequent edge of the external signal applied to the EXTRG pin (when enabled as an Alternate Function).

Bit 3 = **INTG**: *Internal Trigger Enable*.
This bit is set and cleared by software.
0: Internal trigger disabled.
1: Internal trigger enabled. Allows a conversion sequence to be started, synchronized by an internal signal (On-chip Event signal) from a Multifunction Timer peripheral.

Both External and Internal Trigger inputs are internally ORed, thus avoiding Hardware conflicts; however, the correct procedure is to enable only one alternate synchronization input at a time.

Note: The effect of either synchronization mode is to set the START/STOP bit, which is reset by hardware when in SINGLE mode, at the end of each sequence of conversions.

Requirements: The External Synchronisation Input must receive a low level pulse longer than an INTCLK period and, for both External and On-Chip Event synchronisation, the repetition period must be greater than the time required for the selected sequence of conversions.

Bit 2 = **POW**: *Power Up/Power Down.*
This bit is set and cleared by software.
0: Power down mode: all power-consuming logic is disabled, thus selecting a low power idle mode.
1: Power up mode: the A/D converter logic and analog circuitry is enabled.

Bit 1 = **CONT**: *Continuous/Single*.
0: Single Mode: a single sequence of conversions is initiated whenever an external (or internal) trigger occurs, or when the ST bit is set by software.
1: Continuous Mode: the first sequence of conversions is started, either by software (by setting the ST bit), or by hardware (on an internal or external trigger, depending on the setting of the INTG and EXTG bits); a continuous conversion sequence is then initiated.

Bit 0 = **ST**: *Start/Stop.*
0: Stop conversion. When the A/D converter is running in Single Mode, this bit is hardware reset at the end of a sequence of conversions.
1: Start a sequence of conversions.

ANALOG TO DIGITAL CONVERTER (Cont'd)

INTERRUPT CONTROL REGISTER (AD_ICR)
R254 - Read/Write
Register Page: 63
Reset Value: 0000 1111 (0Fh)

7							0
ECV	AWD	ECI	AWDI	X	PL2	PL1	PL0

Bit 7 = **ECV**: *End of Conversion.*
This bit is set by hardware after a group of conversions is completed. It must be reset by the user, before returning from the Interrupt Service Routine. Setting this bit by software will cause a software interrupt request to be generated.
0: No End of Conversion event occurred
1: An End of Conversion event occurred

Bit 6 = **AWD**: *Analog Watchdog.*
This is automatically set by hardware whenever either of the two monitored analog inputs goes out of bounds. The threshold values are stored in registers F8h and FAh for channel 6, and in registers F9h and FBh for channel 7 respectively. The Compare Result Register (CRR) keeps track of the analog inputs exceeding the thresholds.

The AWD bit must be reset by the user, before returning from the Interrupt Service Routine. Setting this bit by software will cause a software interrupt request to be generated.
0: No Analog Watchdog event occurred
1: An Analog Watchdog event occurred

Bit 5 = **ECI**: *End of Conversion Interrupt Enable.*

This bit masks the End of Conversion interrupt request.
0: Mask End of Conversion interrupts
1: Enable End of Conversion interrupts

Bit 4 = **AWDI**: *Analog Watchdog Interrupt Enable*.
This bit masks or enables the Analog Watchdog interrupt request.
0: Mask Analog Watchdog interrupts
1: Enable Analog Watchdog interrupts

Bit 3 = Reserved.

Bit 2:0 = **PL[2:0]:** *A/D Interrupt Priority Level*.
These three bits allow selection of the Interrupt priority level for the A/D.

INTERRUPT VECTOR REGISTER (AD_IVR)
R255 - Read/Write
Register Page: 63
Reset Value: xxxx xx10 (x2h**)**

7							0
V7	V6	V5	V4	V3	V2	W1	0

Bit 7:2 = **V[7:2]:** *A/D Interrupt Vector.*
This vector should be programmed by the User to point to the first memory location in the Interrupt Vector table containing the starting addresses of the A/D interrupt service routines.

Bit 1 = **W1**: *Word Select.*
This bit is set and cleared by hardware, according to the A/D interrupt source.
0: Interrupt source is the Analog Watchdog, pointing to the lower word of the A/D interrupt service block (defined by V[7:2]).
1:Interrupt source is the End of Conversion interrupt, thus pointing to the upper word.

Note: When two requests occur simultaneously, the Analog Watchdog Request has priority over the End of Conversion request, which is held pending.

Bit 0 = Reserved. Forced by hardware to 0.

11 ELECTRICAL CHARACTERISTICS

This product contains devices to protect the inputs against damage due to high static voltages, however it is advisable to take normal precautions to avoid application of any voltage higher than the specified maximum rated voltages.

For proper operation it is recommended that V_{IN} and V_O be higher than V_{SS} and lower than V_{DD}. Reliability is enhanced if unused inputs are connected to an appropriate logic voltage level (V_{DD} or V_{SS}).

Power Considerations. The average chip-junction temperature, T_J, in Celsius can be obtained from:

$T_J = T_A + P_D$ x RthJA

Where: T_A = Ambient Temperature.

RthJA = Package thermal resistance (junction-to ambient).

$P_D = P_{INT} + P_{PORT}$.

$P_{INT} = I_{DD}$ x V_{DD} (chip internal power).

P_{PORT} = Port power dissipation (determined by the user)

ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
V_{DD}	Supply Voltage	– 0.3 to 6.5	V
AV_{DD}	A/D Converter Analog Reference	V_{DD} – 0.3 to V_{DD} + 0.3	V
AV_{SS}	A/D Converter V_{SS}	V_{SS}	
V_{IN}	Input Voltage (standard I/O pins)	– 0.3 to V_{DD} + 0.3	V
V_{INOD}	Input Voltage (open drain I/O pins)	– 0.3 to 6.5	V
V_{AIN}	Analog Input Voltage (A/D Converter)	AV_{SS} to AV_{DD}	V
T_{STG}	Storage Temperature	– 55 to +150	°C
I_{INJ}	Pin Injection Current - Digital and Analog Input	±10	mA
	Maximum Accumulated Pin injection Current in the device	±100	mA
ESD	ESD Susceptibility	2000	V

Note:
Stresses above those listed as “absolute maximum ratings“ may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these conditions is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability. All voltages are referenced to V_{SS}=0.

THERMAL CHARACTERISTICS

Symbol	Package	Value	Unit
RthJA	PQFP100	28	°C/W

RECOMMENDED OPERATING CONDITIONS

Symbol	Parameter	Value		Unit
		Min	Max	
T_A	Operating Temperature	–40	105	°C
V_{DD}	Operating Supply Voltage	4.5	5.5	V
AV_{DD}	Analog Supply Voltage	V_{DD} - 0.3	V_{DD} + 0.3	V
f_{INTCLK}	Internal Clock Frequency @ 4.5V - 5.5V	0(1)	24	MHz

Note:
(1) 1MHz when A/D or JBLPD is used, 2.6MHz when I²C is used.

DC ELECTRICAL CHARACTERISTICS

(V_{DD} = 5V ± 10%, T_A = –40°C to +105°C, unless otherwise specified)

Symbol	Parameter	Comment	Value			Unit
			Min	Typ(1)	Max	
V_{IH}	Input High Level P0[7:0]-P1[7.0]-P2[7:6] P3.3 P4.2-P4.5-P5.3	TTL	2.0			V
		CMOS	0.7 x V_{DD}			V
	Input High Level Pure open-drain I/O P2[3:2]	TTL	2.0			V
		CMOS	0.7 x V_{DD}			V
	Input High Level Standard Schmitt Trigger P2[5:4]-P2[1:0]-P3[7:4] P3[2:1]-P4[4:3]-P4[1:0] P5[7:4]-P5[2:0]-P6[3:0] P7[7:0]-P8[7:0]-P9[7:0]		0.7 x V_{DD}			V
	Input High Level Pure open-drain I/O Special Schmitt Trigger P4[7:6]		0.7 x V_{DD}			V
	Input High Level High Hyst. Schmitt Trigger P6[5:4]		0.7 x V_{DD}			V
V_{IL}	Input Low Level P0[7:0]-P1[7:0]-P2[7:6] P2[3:2]-P3.3-P4.2-P4.5-P5.3	TTL			0.8	V
		CMOS			0.3 x V_{DD}	V
	Input Low Level Standard Schmitt Trigger P2[5:4]-P2[1:0]-P3[7:4] P3[2:1]-P4[4:3]-P4[1:0] P5[7:4]-P5[2:0]-P6[3:0] P7[7:0]-P8[7:0]-P9[7:0]				0.8	V
	Input Low Level Special Schmitt Trigger P4[7:6]				0.3 x V_{DD}	
	Input Low Level High Hyst. Schmitt Trigger P6[5:4]				0.3 x V_{DD}	
V_I	Input Voltage Range Pure Open Drain P2[3:2]-P4[7:6]		-0.3		6.0	V
	Input Voltage Range All other pins		-0.3		V_{DD} + 0.3	V

Symbol	Parameter	Comment	Value			Unit
			Min	Typ(1)	Max	
V_{HYS}	Input Hysteresis (2) Standard Schmitt Trigger P2[5:4]-P2[1:0]-P3[7:4] P3[2:1]-P4[4:3]-P4[1:0] P5[7:4]-P5[2:0]-P6[3:0] P7[7:0]-P8[7:0]-P9[7:0]			600		mV
	Input Hysteresis (2) Special Schmitt Trigger P4[7:6]			800		mV
	Input Hysteresis (2) High Hyst. Schmitt Trigger P6[5:4]			900		mV
V_{OH}	Output High Level P0[7:0]-P6[5:4] $\overline{AS}$-$\overline{DS}$-R$\overline{W}$	Push Pull, I_{OH}= – 2mA EMR1 Register - BSZ bit = 0 (3)	V_{DD} – 0.8			V
		Push Pull, I_{OH}= – 8mA EMR1 Register - BSZ bit = 1 (3)	V_{DD} – 0.8			V
	Output High Level P1[7:0]-P2[7:4]-P2[1:0] P3[7:1]-P4[5:0]-P5[7:0]-P6[3:0]-P7[7:0]-P8[7:0] P9[7:0]-VPWO	Push Pull, I_{OH}= – 2mA EMR1 Register - BSZ bit = 0 (3)	V_{DD} – 0.8			V
		Push Pull, I_{OH}= – 4mA EMR1 Register - BSZ bit = 1 (3)	V_{DD} – 0.8			V
V_{OL}	Output Low Level P0[7:0]-P2[3:2]-P4[7:6] P6[5:4]-$\overline{AS}$-$\overline{DS}$-R$\overline{W}$	Push Pull / Open Drain, I_{OL}=2mA EMR1 Register - BSZ bit = 0 (3)			0.4	V
		Push Pull / Open Drain, I_{OL}=8mA, EMR1 Register - BSZ bit = 1 (3)			0.4	V
	Output Low Level P1[7:0]-P2[7:4]-P2[1:0] P3[7:1]-P4[5:0]-P5[7:0] P6[3:0]-P7[7:0]-P8[7:0] P9[7:0]-VPWO	Push Pull / Open Drain, I_{OL}=2mA, EMR1 Register - BSZ bit = 0 (3)			0.4	V
		Push Pull / Open Drain, I_{OL}=4mA, EMR1 Register - BSZ bit = 1 (3)			0.4	V
R_{WPU}	Weak Pull-up Current P2[7:4]-P2[1:0]-P3[7:1] P4.5-P4[3:1]-P5.3-P6[3:0] P7[7:0] P8[7:0]-P9[7:0]	Bidirectional Weak Pull-up V_{OL} = 0V	30	100	400	µA
	Weak Pull-up Current P6[5:4]-$\overline{AS}$-$\overline{DS}$	Bidirectional Weak Pull-up V_{OL} = 0V	100	200	450	µA
I_{LKIO}	I/O Pin Input Leakage	Input/Tri-State, $0V < V_{IN} < V_{DD}$	– 1		+ 1	µA
I_{LKIOD}	I/O Pin Open Drain Input Leakage	Input/Tri-State, $0V < V_{IN} < V_{DD}$	– 1		+ 1	µA
$I_{LKA/D}$	A/D Conv. Input Leakage		– 1		+ 1	µA
I_{OV}	Overload Current	(4)			5	mA
SR_R	Slew Rate Rise	(5)			70	mA/ns
SR_F	Slew Rate Fall	(5)			70	mA/ns

Note:
(1) Unless otherwise stated, typical data are based on T_A= 25°C and V_{DD}= 5V. They are only reported for design guide lines not tested in production. Hysteresis voltage between switching levels: characterization results - not tested.
(2) Hysteresis voltage between switching levels: characterization results - not tested.
(3) For a description of the EMR1 Register - BSZ bit refer to the External Memory Interface Chapter.
(4) Not 100% tested, guaranteed by design characterisation. The absolute sum of input overload currents on all port pins may not exceed 100 mA.
(5) Indicative values extracted from Design simulation.

AC ELECTRICAL CHARACTERISTICS

(V_{DD} = 5V ± 10%, T_A = –40°C to +105°C, unless otherwise specified)

Symbol	Parameter	INTCLK	Typ (1)	Max	Unit
I_{DDRUN}	Run Mode Current (2)	24 MHz	45	60 2.5	mA mA/MHz
I_{DDWFI}	WFI Mode Current	24 MHz	14	22 0.9	mA mA/MHz
$I_{DDLPWFI}$	Low Power WFI Mode Current (3)	4MHz / 32	400	600	µA
I_{DDHALT}	HALT Mode Current	-		10	µA
I_{DDTR}	Input Transient I_{DD} Current (4)	-	500		µA

Note:

All I/O Ports are configured in bidirectional weak pull-up mode with no DC load, external clock pin (OSCIN) is driven by square wave external clock.

(1) Unless otherwise stated, typical data are based on T_A= 25°C and V_{DD}= 5V. They are only reported for design guide lines not tested in production.

(2) CPU running with memory access, all peripherals switched off.

(3) FLASH/EEPROM in Power-Down Mode.

(4) Measured in HALT Mode, forcing a slow ramp voltage on one I/O pin, configured in input.

FLASH / EEPROM SPECIFICATIONS

(V_{DD} = 5V ± 10%, T_A = –40°C to +105°C, unless otherwise specified) (1)

Parameter		Min	Typ	Max	Unit
MAIN FLASH	Byte Program		10	1200	µs
	128 kbytes Flash Program		1.3		s
	64 kbytes Flash Sector Erase	1	1.5	30	s
	128 kbytes Flash Chip Erase		3		s
	Erase Suspend Latency			15	µs
EEPROM	16 bytes Page Update (1k EEPROM)	0.16	30	200	ms
	EEPROM Chip Erase		70		ms
RELIABILITY	Flash Endurance 25°C	10000			cycles
	Flash Endurance -40°C +105°C	3000			
	EEPROM Endurance	100000 (2)			cycles / sector
	Data Retention	15			Years

Note:
(1) The full range of characteristics will be available after final product characterisation.
(2) Relationship computation between EEPROM sector cycling and single byte cycling is provided in a dedicated STMicroelectronics Application Note (ref. AN1152).

FLASH / EEPROM DC & AC CHARACTERISTICS

(V_{DD} = 5V ± 10%, T_A = –40°C to +105°C, unless otherwise specified)

Symbol	Parameter	Test Conditions	Min	Max	Unit
V_{CWL}	Write Lock Supply Voltage (1)		3	4	V
V_{CRL}	Read Lock Supply Voltage (2)		1.5	2.5	V
IDD1	Supply Current (Read)	V_{DD} = 5.5 V, T_A = –40°C, f_{INTCLK} = 24 MHz		60	mA
IDD2	Supply Current (Write)	V_{DD} = 5.5 V, T_A = –40°C, f_{INTCLK} = 24 MHz		60	mA
IDD3	Supply Current (Stand-by)	V_{DD} = 5.5 V, T_A = –40°C		100	µA
IDD4	Supply Current (Power-Down)	V_{DD} = 5.5 V, T_A = 105°C		10	µA
TPD	Recovery from Power-Down	V_{DD} = 4.5 V, T_A = 105°C		10	µs

Note:
(1) Below the min value the FLASH / EEPROM can never be written; above the max value FLASH/EEPROM can always be written.
(2) Below the min value the FLASH / EEPROM can never be read; above the max value FLASH/EEPROM can always be read.

EXTERNAL INTERRUPT TIMING TABLE

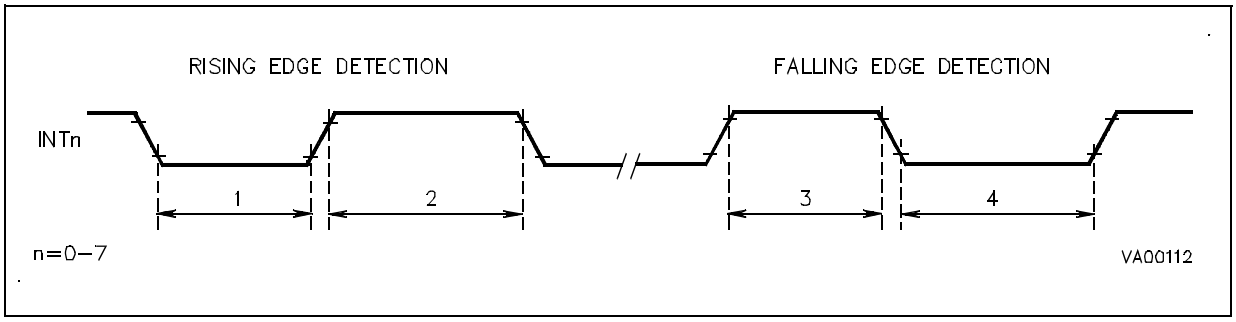
(V_{DD} = 5V ± 10%, T_A = –40°C to +105°C, C_{Load} = 50pF, f_{INTCLK} = 24MHz, unless otherwise specified)

N°	Symbol	Parameter	Value		Unit
			Formula (1)	Min	
1	TwINTLR	Low Level Minimum Pulse Width in Rising Edge Mode	≥Tck+10	50	ns
2	TwINTHR	High Level Minimum Pulse Width in Rising Edge Mode	≥Tck+10	50	ns
3	TwINTHF	High Level Minimum Pulse Width in Falling Edge Mode	≥Tck+10	50	ns
4	TwINTLF	Low Level Minimum Pulse Width in Falling Edge Mode	≥Tck+10	50	ns

Note:
The value in the left hand column shows the formula used to calculate the timing minimum or maximum from the oscillator clock period. The value in the right hand two columns shows the timing minimum and maximum for an internal clock at 24MHz (INTCLK).
Measurement points are V_{IH} for positive pulses and V_{IL} for negative pulses.
(1) Formula guaranteed by design.

Legend:
Tck = INTCLK period = OSCIN period when OSCIN is not divided by 2;
2 x OSCIN period when OSCIN is divided by 2;
OSCIN period x PLL factor when the PLL is enabled.

EXTERNAL INTERRUPT TIMING



WAKE-UP MANAGEMENT TIMING TABLE

(V_{DD} = 5V ± 10%, T_A = –40°C to +105°C, C_{Load} = 50pF, f_{INTCLK} = 24MHz, , unless otherwise specified)

N°	Symbol	Parameter	Value		Unit
			Formula(1)	Min	
1	TwWKPLR	Low Level Minimum Pulse Width in Rising Edge Mode	≥Tck+10	≥ 50	ns
2	TwWKPHR	High Level Minimum Pulse Width in Rising Edge Mode	≥Tck+10	≥ 50	ns
3	TwWKPHF	High Level Minimum Pulse Width in Falling Edge Mode	≥Tck+10	50	ns
4	TwWKPLF	Low Level Minimum Pulse Width in Falling Edge Mode	≥Tck+10	50	ns

Note: The value in the left hand column shows the formula used to calculate the timing minimum or maximum from the oscillator clock period. The value in the right hand two columns show the timing minimum and maximum for an internal clock at 24MHz (INTCLK).

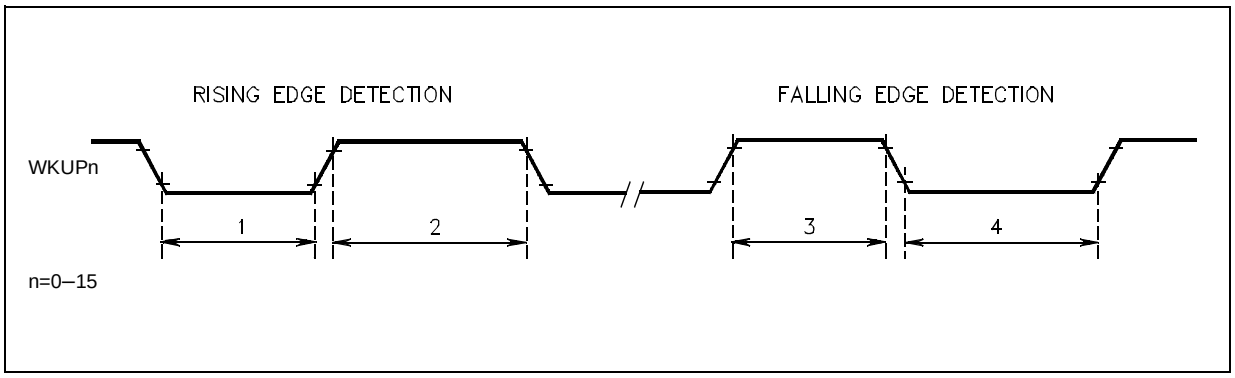
The given data are related to Wake-up Management Unit used in External Interrupt mode.

Measurement points are V_{IH} for positive pulses and V_{IL} for negative pulses.

(1) Formula guaranteed by design.

Legend:
Tck = INTCLK period = OSCIN period when OSCIN is not divided by 2;
2 x OSCIN period when OSCIN is divided by 2;
OSCIN period x PLL factor when the PLL is enabled.

WAKE-UP MANAGEMENT TIMING



RCCU CHARACTERISTICS

(V_{DD} = 5V ± 10%, T_A = –40°C to +105°C, C_{Load} = 50pF, f_{INTCLK} = 24MHz, unless otherwise specified)

Symbol	Parameter	Comment	Value			Unit
			Min	Typ (1)	Max	
V_{IHRS}	$\overline{RESET}$ Input High Level	Input Threshold		3.2		V
		Input Voltage Range			V_{DD} + 0.3	V
V_{ILRS}	$\overline{RESET}$ Input Low Level	Input Threshold		2.4		V
		Input Voltage Range	– 0.3			
V_{HYRS}	$\overline{RESET}$ Input Hysteresis			800		mV
I_{LKRS}	$\overline{RESET}$ Pin Input Leakage	0V < V_{IN} < V_{DD}	– 1		+ 1	µA

Note:
(1) Unless otherwise stated, typical data are based on T_A= 25°C and V_{DD}= 5V. They are only reported for design guide lines not tested in production.

RCCU TIMING TABLE

(V_{DD} = 5V ± 10%, T_A = –40°C to +105°C, C_{Load} = 50pF, f_{INTCLK} = 24MHz, unless otherwise specified)

Symbol	Parameter	Comment	Value			Unit
			Min	Typ (1)	Max	
T_{FRS}	$\overline{RESET}$ Input Filtered Pulse				50	ns
T_{NFR}	RESET Input Non Filtered Pulse		20			µs
T_{RSPH}(2)	RESET Phase duration			20400 x T_{osc}		µs
T_{STR}	STOP Restart duration	DIV2 = 0 DIV2 = 1		10200 x T_{osc} 20400 x T_{osc}		µs

Note:
(1) Unless otherwise stated, typical data are based on T_A= 25°C and V_{DD}= 5V. They are only reported for design guide lines not tested in production.
(2) Depending on the delay between rising edge of $\overline{RESET}$ pin and the first rising edge of CLOCK1, the value can differ from the typical value for +/- 1 CLOCK1 cycle.
Legend: T_{osc} = OSCIN clock periods.

PLL CHARACTERISTICS

(V_{DD} = 5V ± 10%, T_A = –40°C to +105°C, C_{Load} = 50pF, f_{INTCLK} = 24MHz, unless otherwise specified)

Symbol	Parameter	Comment	Value			Unit
			Min	Typ (1)	Max	
F_{XTL}	Crystal Reference Frequency		2		5	MHz
F_{VCO}	VCO Operating Frequency		6		24	MHz
T_{PLK}	Lock-in Time			350 x Tosc	1000 x Tosc	µs
	PLL Jitter		0		1200 (2)	ps

Note:
(1) Unless otherwise stated, typical data are based on T_A= 25°C and V_{DD}= 5V. They are only reported for design guide lines not tested in production.
(2) Measured at 24MHz (INTCLK). Guaranteed by Design Characterisation (not tested).
Legend: Tosc = OSCIN clock periods.

OSCILLATOR CHARACTERISTICS

(V_{DD} = 5V ± 10%, T_A = –40°C to +105°C, C_{Load} = 50pF, f_{INTCLK} = 24MHz, unless otherwise specified)

Symbol	Parameter	Comment	Value			Unit
			Min	Typ (1)	Max	
F_{OSC}	Crystal Frequency	Fundamental mode crystal only	2		5	MHz
g_m	Oscillator		1.77	2.0	3.8	mA/V
V_{IHCK}	Clock Input High Level	External Clock	1.2		V_{DD} + 0.3	V
V_{ILCK}	Clock Input Low Level	External Clock	0.4		0.5	V
I_{LKOS}	OSCIN/OSCOUT Pins Input Leakage	$0V < V_{IN} < V_{DD}$ (HALT/STOP)	– 1		+ 1	µA
T_{STUP}	Oscillator Start-up Time				5	ms

Note:
(1) Unless otherwise stated, typical data are based on T_A= 25°C and V_{DD}= 5V. They are only reported for design guide lines not tested in production.

EXTERNAL BUS TIMING TABLE

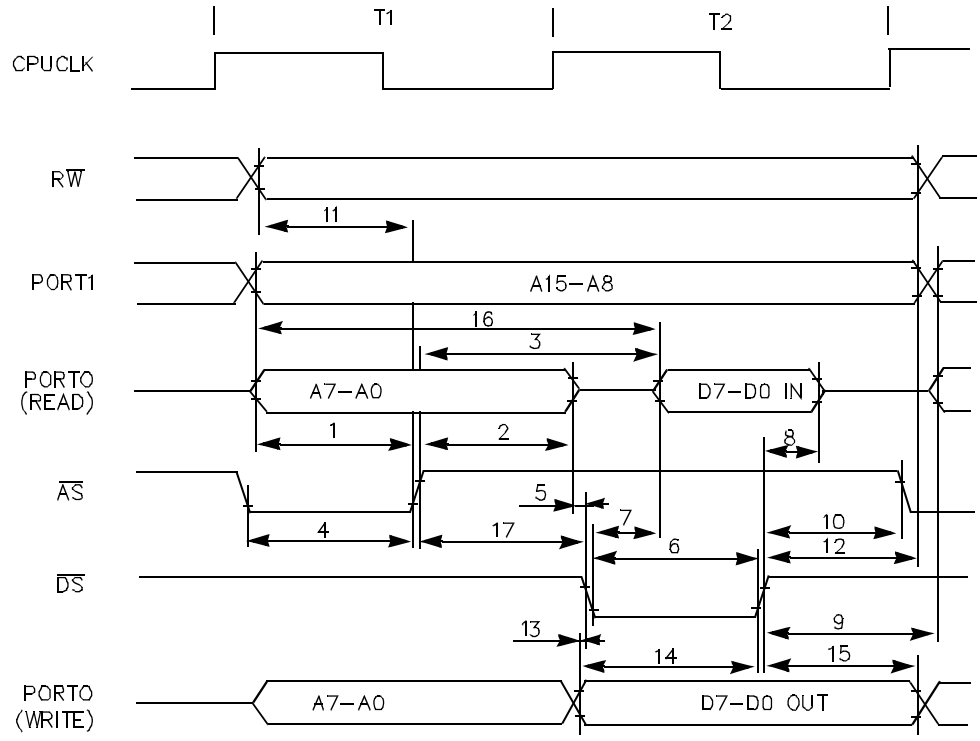
(V_{DD} = 5V ± 10%, T_A = –40°C to +105°C, C_{Load} = 50pF, f_{INTCLK} = 24MHz, unless otherwise specified)

N°	Symbol	Parameter	Value (Note)			Unit
			Formula	Min	Max	
1	TsA (AS)	Address Set-up Time before $\overline{AS}$ ↑	Tck x Wa+TckH-9	12		ns
2	ThAS (A)	Address Hold Time after $\overline{AS}$ ↑	TckL-4	17		ns
3	TdAS (DR)	$\overline{AS}$ ↑ to Data Available (read)	Tck x (Wd+1)+3		45	ns
4	TwAS	$\overline{AS}$ Low Pulse Width	Tck x Wa+TckH-5	16		ns
5	TdAz (DS)	Address Float to $\overline{DS}$ ↓	0	0		ns
6	TwDS	$\overline{DS}$ Low Pulse Width	Tck x Wd+TckH-5	16		ns
7	TdDSR (DR)	$\overline{DS}$ ↓ to Data Valid Delay (read)	Tck x Wd+TckH+4		25	ns
8	ThDR (DS)	Data to $\overline{DS}$ ↑ Hold Time (read)	7	7		ns
9	TdDS (A)	$\overline{DS}$ ↑ to Address Active Delay	TckL+11	32		ns
10	TdDS (AS)	$\overline{DS}$ ↑ to $\overline{AS}$ ↓ Delay	TckL-4	17		ns
11	TsR/W (AS)	R$\overline{W}$ Set-up Time before AS ↑	Tck x Wa+TckH-17	4		ns
12	TdDSR (R/W)	$\overline{DS}$ ↑ to R$\overline{W}$ and Address Not Valid Delay	TckL-1	20		ns
13	TdDW (DSW)	Write Data Valid to $\overline{DS}$ ↓ Delay	-16	-16		ns
14	TsD(DSW)	Write Data Set-up before $\overline{DS}$ ↑	Tck x Wd+TckH-16	5		ns
15	ThDS (DW)	Data Hold Time after $\overline{DS}$ ↑ (write)	TckL-3	18		ns
16	TdA (DR)	Address Valid to Data Valid Delay (read)	Tck x (Wa+Wd+1)+TckH-7		55	ns
17	TdAs (DS)	$\overline{AS}$ ↑ to $\overline{DS}$ ↓ Delay	TckL-6	15		ns

Note: The value in the left hand column shows the formula used to calculate the timing minimum or maximum from the oscillator clock period, prescaler value and number of wait cycles inserted.
The values in the right hand two columns show the timing minimum and maximum for an external clock at 24MHz, prescaler value of zero and zero wait states.

Legend:
Tck = INTCLK period = OSCIN period when OSCIN is not divided by 2;
2 x OSCIN period when OSCIN is divided by 2;
OSCIN period x PLL factor when the PLL is enabled.
TckH = INTCLK high pulse width (normally = Tck/2, except when INTCLK = OSCIN, in which case it is OSCIN high pulse width)
TckL = INTCLK low pulse width (normally = Tck/2, except when INTCLK = OSCIN, in which case it is OSCIN low pulse width)
P = clock prescaling value (=PRS; division factor = 1+P)
Wa = wait cycles on $\overline{AS}$; = max (P, programmed wait cycles in EMR2, requested wait cycles with $\overline{WAIT}$)
Wd = wait cycles on $\overline{DS}$; = max (P, programmed wait cycles in WCR, requested wait cycles with $\overline{WAIT}$)

EXTERNAL BUS TIMING



WATCHDOG TIMING TABLE

(V_{DD} = 5V ± 10%, T_A = –40°C to +105°C, C_{Load} = 50pF, f_{INTCLK} = 24MHz, Push-pull output configuration, unless otherwise specified)

N°	Symbol	Parameter	Value			Unit
			Formula (1)	Min	Max	
1	TwWDOL	WDOUT Low Pulse Width	4 x (Psc+1) x (Cnt+1) x Tck	167	2.8	ns s
			(Psc+1) x (Cnt+1) x T_{WDIN} with $T_{WDIN} \geq 8$ x Tck	333		ns
2	TwWDOH	WDOUT High Pulse Width	4 x (Psc+1) x (Cnt+1) x Tck	167	2.8	ns s
			(Psc+1) x (Cnt+1) x T_{WDIN} with $T_{WDIN} \geq 8$ x Tck	333		ns
3	TwWDIL	WDIN High Pulse Width	≥ 4 x Tck	167		ns
4	TwWDIH	WDIN Low Pulse Width	≥ 4 x Tck	167		ns

Note: The value in the left hand column shows the formula used to calculate the timing minimum or maximum from the oscillator clock period, watchdog prescaler and counter programmed values.
The value in the right hand two columns show the timing minimum and maximum for an internal clock (INTCLK) at 24MHz, with minimum and maximum prescaler value and minimum and maximum counter value.

Measurement points are V_{OH} or V_{IH} for positive pulses and V_{OL} or V_{IL} for negative pulses.

(1) Formula guaranteed by design.

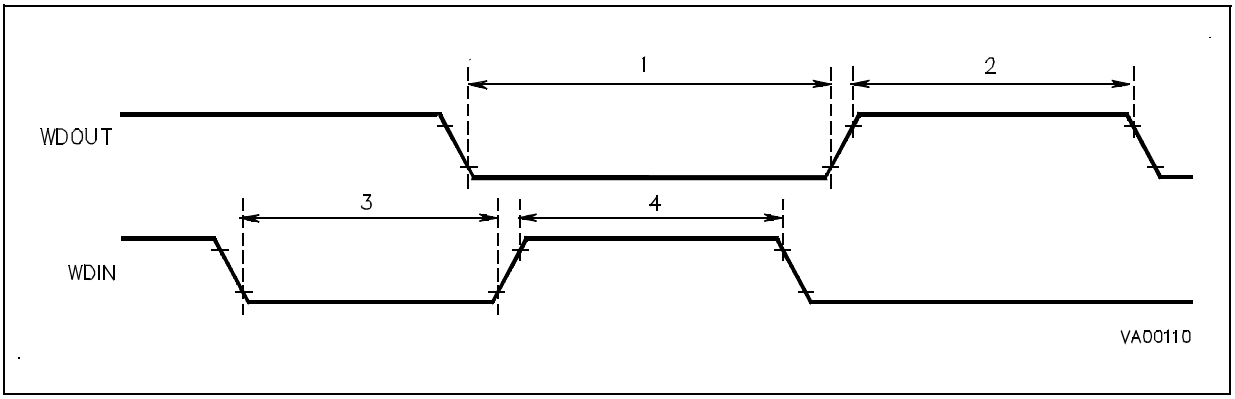
Legend:
Tck = INTCLK period = OSCIN period when OSCIN is not divided by 2;
2 x OSCIN period when OSCIN is divided by 2;
OSCIN period x PLL factor when the PLL is enabled.

Psc = Watchdog Prescaler Register content (WDTPR): from 0 to 255

Cnt = Watchdog Couter Registers content (WDTRH,WDTRL): from 0 to 65535

T_{WDIN} = Watchdog Input signal period (WDIN)

WATCHDOG TIMING



STANDARD TIMER TIMING TABLE

(V_{DD} = 5V ± 10%, T_A = –40°C to +105°C, C_{Load} = 50pF, f_{INTCLK} = 24MHz, Push-pull output configuration, unless otherwise specified)

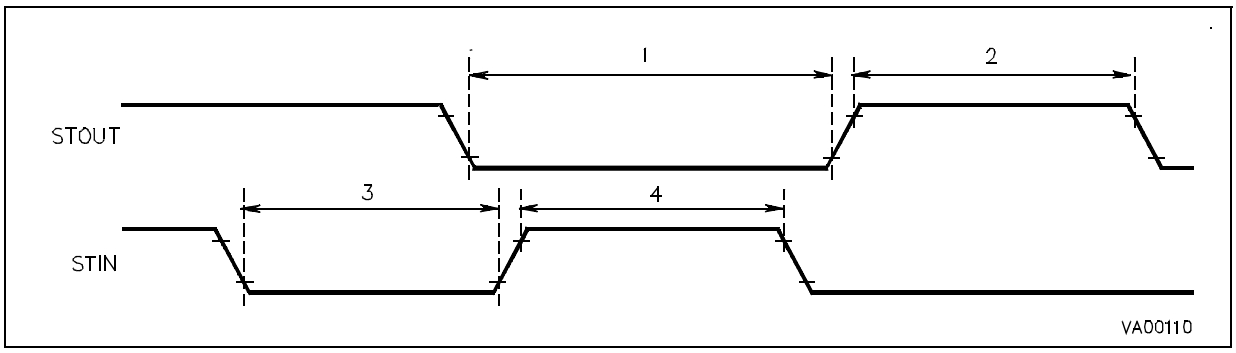
N°	Symbol	Parameter	Value			Unit
			Formula (1)	Min	Max	
1	TwSTOL	STOUT Low Pulse Width	4 x (Psc+1) x (Cnt+1) x Tck	167	2.8	ns s
			(Psc+1) x (Cnt+1) x T_{STIN} with $T_{STIN} \geq 8$ x Tck	(2)	(2)	ns
2	TwSTOH	STOUT High Pulse Width	4 x (Psc+1) x (Cnt+1) x Tck	167	2.8	ns s
			(Psc+1) x (Cnt+1) x T_{STIN} with $T_{STIN} \geq 8$ x Tck	(2)	(2)	ns
3	TwSTIL	STIN High Pulse Width	≥ 4 x Tck	(2)	(2)	ns
4	TwSTIH	STIN Low Pulse Width	≥ 4 x Tck	(2)	(2)	ns

Note: The value in the left hand column shows the formula used to calculate the timing minimum or maximum from the oscillator clock period, standard timer prescaler and counter programmed values.
The value in the right hand two columns show the timing minimum and maximum for an internal clock (INTCLK) at 24MHz, with minimum and maximum prescaler value and minimum and maximum counter value.
Measurement points are V_{OH} or V_{IH} for positive pulses and V_{OL} or V_{IL} for negative pulses.
(1) Formula guaranteed by design.

(2) On this product STIN is not available as Alternate Function but it is internally connected to a precise clock source directly derived from OSCIN. Refer to RCCU chapter for details about clock distribution.

Legend:
Tck = INTCLK period = OSCIN period when OSCIN is not divided by 2;
2 x OSCIN period when OSCIN is divided by 2;
OSCIN period x PLL factor when the PLL is enabled.
Psc = Standard Timer Prescaler Register content (STP): from 0 to 255
Cnt = Standard Timer Couter Registers content (STH,STL): from 0 to 65535
T_{STIN} = Standard Timer Input signal period (STIN).

STANDARD TIMER TIMING



EXTENDED FUNCTION TIMER EXTERNAL TIMING TABLE

(V_{DD} = 5V ± 10%, T_A = –40°C to +105°C, C_{Load} = 50pF, f_{INTCLK} = 24MHz, unless otherwise specified)

N°	Symbol	Parameter	Value		Unit
			Formula (1)	Min	
1	Tw_{PEWL}	External Clock low pulse width (EXTCLK)	-	2 x Tck + 10	ns
2	Tw_{PEWH}	External Clock high pulse width (EXTCLK)	-	2 x Tck + 10	ns
3	Tw_{PIWL}	Input Capture low pulse width (ICAPx)	-	2 x Tck + 10	ns
4	Tw_{PIWH}	Input Capture high pulse width (ICAPx)	-	2 x Tck + 10	ns
5	Tw_{ECKD}	Distance between two active edges on EXTCLK	≥ 4 x Tck + 10	177	ns
6	Tw_{EICD}	Distance between two active edges on ICAPx	2 x Tck x Prsc +10	177	ns

Note: The value in the left hand column shows the formula used to calculate the timing minimum or maximum from the oscillator clock period, standard timer prescaler and counter programmed values.•
The value in the right hand two columns show the timing minimum and maximum for an internal clock (INTCLK) at 24MHz, and minimum prescaler factor (=2).

Measurement points are V_{IH} for positive pulses and V_{IL} for negative pulses.

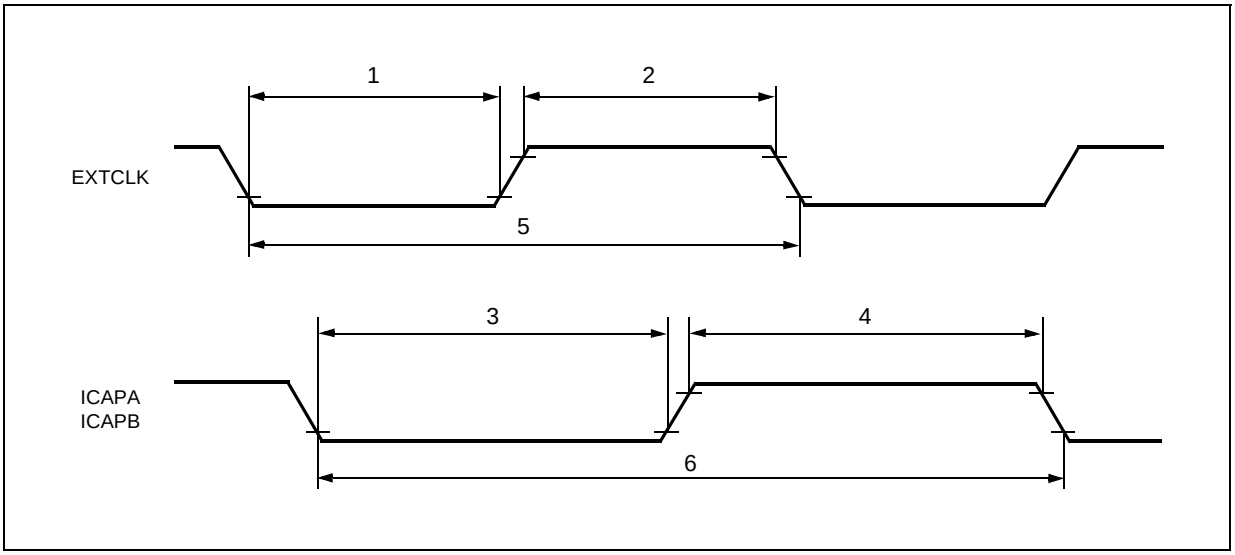
(1) Formula guaranteed by design.

Legend:

Tck = INTCLK period = OSCIN period when OSCIN is not divided by 2;
2 x OSCIN period when OSCIN is divided by 2;
OSCIN period x PLL factor when the PLL is enabled.

Prsc = Precsaler factor defined by Extended Function Timer Clock Control bits (CC1,CC0) on control register CR2 (values: 2,4,8).

EXTENDED FUNCTION TIMER EXTERNAL TIMING



MULTIFUNCTION TIMER EXTERNAL TIMING TABLE

(V_{DD} = 5V ± 10%, T_A = –40°C to +105°C, C_{Load} = 50pF, f_{INTCLK} = 24MHz, unless otherwise specified)

N°	Symbol	Parameter	Value			Unit	Note
			Formula	Min	Max		
1	Tw_{CTW}	External clock/trigger pulse width	n x Tck	n x 42	-	ns	(1)
2	Tw_{CTD}	External clock/trigger pulse distance	n x Tck	n x 42	-	ns	(1)
3	Tw_{AED}	Distance between two active edges	3 x Tck	125	-	ns	
4	Tw_{GW}	Gate pulse width	6 x Tck	250	-	ns	
5	Tw_{LBA}	Distance between TINB pulse edge and the following TINA pulse edge	Tck	42	-	ns	(2)
6	Tw_{LAB}	Distance between TINA pulse edge and the following TINB pulse edge		0	-	ns	(2)
7	Tw_{AD}	Distance between two TxINA pulses		0	-	ns	(2)
8	Tw_{OWD}	Minimum output pulse width/distance	3 x Tck	125	-	ns	

Note: The value in the left hand column shows the formula used to calculate the timing minimum or maximum from the oscillator clock period, standard timer prescaler and counter programmed values.
The value in the right hand two columns show the timing minimum and maximum for an internal clock (INTCLK) at 24MHz.

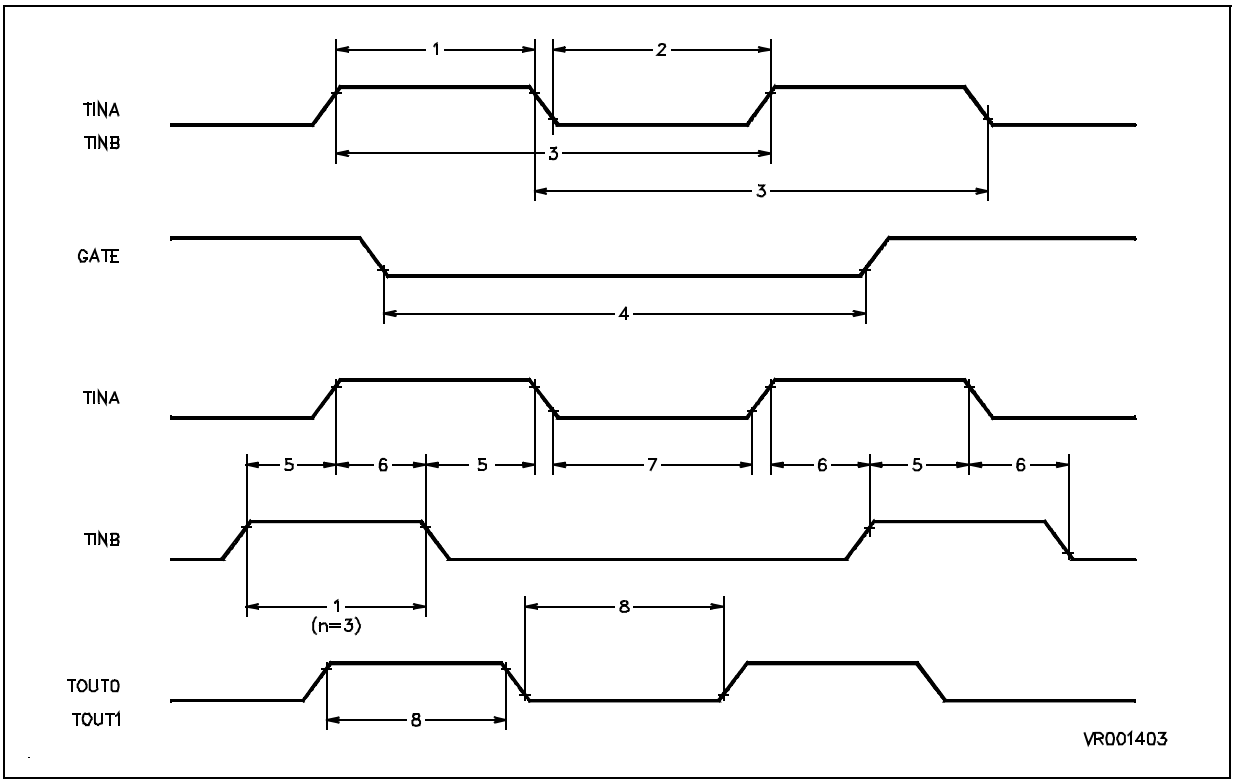
(1) n = 1 if the input is rising OR falling edge sensitive
n = 3 if the input is rising AND falling edge sensitive

(2) In Autodiscrimination mode

Legend:

Tck = INTCLK period = OSCIN period when OSCIN is not divided by 2;
2 x OSCIN period when OSCIN is divided by 2;
OSCIN period x PLL factor when the PLL is enabled.

MULTIFUNCTION TIMER EXTERNAL TIMING



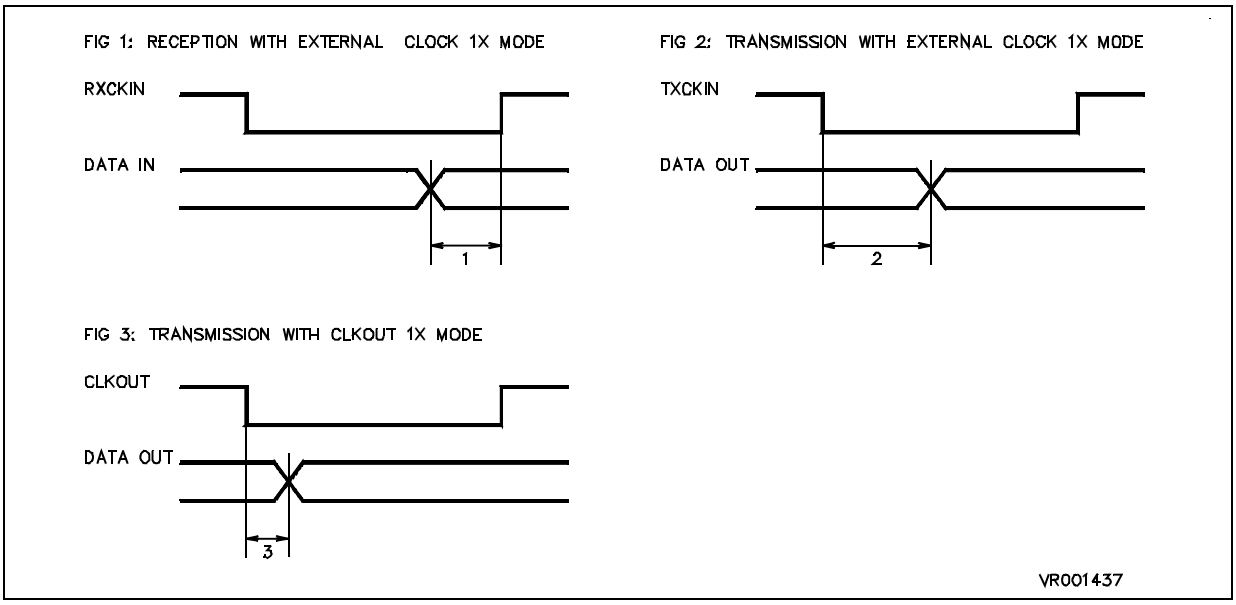
SCI TIMING TABLE

(V_{DD} = 5V ± 10%, T_A = –40°C to +105°C, C_{Load} = 50pF, f_{INTCLK} = 24MHz, unless otherwise specified)

N°	Symbol	Parameter	Condition	Value		Unit
				Min	Max	
	F_{RxCKIN}	Frequency of RxCKIN	1x mode		f_{INTCLK} / 8	MHz
			16x mode		f_{INTCLK} / 4	MHz
	Tw_{RxCKIN}	RxCKIN shortest pulse	1x mode	4 x Tck		s
			16x mode	2 x Tck		s
	F_{TxCKIN}	Frequency of TxCKIN	1x mode		f_{INTCLK} / 8	MHz
			16x mode		f_{INTCLK} / 4	MHz
	Tw_{TxCKIN}	TxCKIN shortest pulse	1x mode	4 x Tck		s
			16x mode	2 x Tck		s
1	Ts_{DS}	DS (Data Stable) before rising edge of RxCKIN	1x mode reception with RxCKIN	Tck / 2		ns
2	Td_{D1}	TxCKIN to Data out delay Time	1x mode transmission with external clock C_{Load} < 50pF		2.5 x Tck	ns
3	Td_{D2}	CLKOUT to Data out delay Time	1x mode transmission with CLKOUT	350		ns

Legend:
Tck = INTCLK period = OSCIN period when OSCIN is not divided by 2;
2 x OSCIN period when OSCIN is divided by 2;
OSCIN period x PLL factor when the PLL is enabled.

SCI TIMING



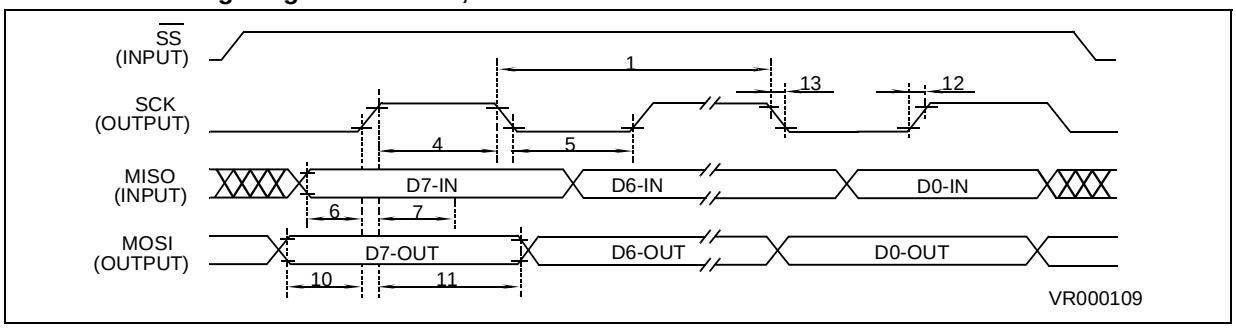
SPI TIMING TABLE

(V_{DD} = 5V ± 10%, T_A = –40°C to +105°C, C_{Load} = 50pF, f_{INTCLK} = 24MHz, unless otherwise specified)

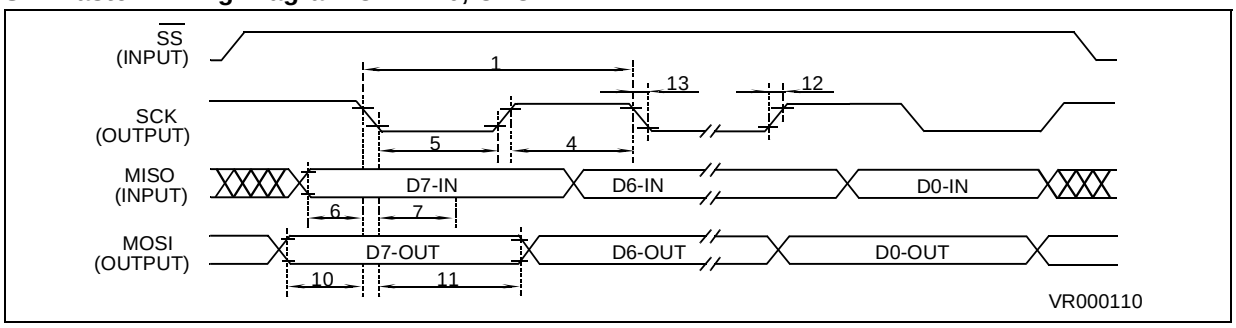
N°	Symbol	Parameter	Condition	Value (1)		Unit
				Min	Max	
	f_{SPI}	SPI frequency	Master Slave	f_{INTCLK} / 128 0	f_{INTCLK} / 4 f_{INTCLK} / 2	MHz
1	t_{SPI}	SPI clock period	Master Slave	4 x Tck 2 x Tck		ns
2	t_{Lead}	Enable lead time	Slave	40		ns
3	t_{Lag}	Enable lag time	Slave	40		ns
4	t_{SPI_H}	Clock (SCK) high time	Master Slave	80 90		ns
5	t_{SPI_L}	Clock (SCK) low time	Master Slave	80 90		ns
6	t_{SU}	Data set-up time	Master Slave	40 40		ns
7	t_H	Data hold time (inputs)	Master Slave	40 40		ns
8	t_A	Access time (time to data active from high impedance state)	Slave	0	120	ns
9	t_{Dis}	Disable time (hold time to high impedance state)			240	ns
10	t_V	Data valid	Master (before capture edge) Slave (after enable edge)	Tck / 4	 120	ns ns
11	t_{Hold}	Data hold time (outputs)	Master (before capture edge) Slave (after enable edge)	Tck / 4 0		ns ns
12	t_{Rise}	Rise time (20% V_{DD} to 70% V_{DD}, C_L = 200pF)	Outputs: SCK,MOSI,MISO Inputs: SCK,MOSI,MISO,$\overline{SS}$		100 100	ns µs
13	t_{Fall}	Fall time (70% V_{DD} to 20% V_{DD}, C_L = 200pF)	Outputs: SCK,MOSI,MISO Inputs: SCK,MOSI,MISO,$\overline{SS}$		100 100	ns µs

Note:
Measurement points are V_{OL}, V_{OH}, V_{IL} and V_{IH} in the SPI Timing Diagram.
(1) Values guaranteed by design.
Legend:
Tck = INTCLK period = OSCIN period when OSCIN is not divided by 2;
2 x OSCIN period when OSCIN is divided by 2;
OSCIN period x PLL factor when the PLL is enabled.

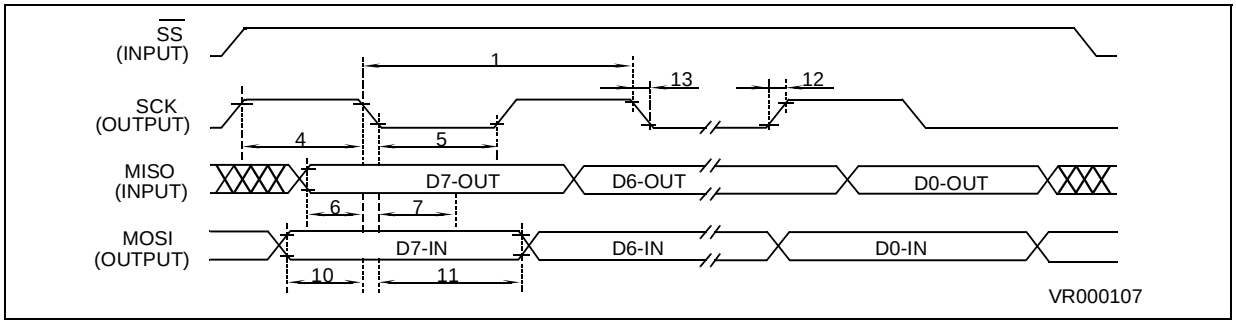
SPI Master Timing Diagram CPHA=0, CPOL=0



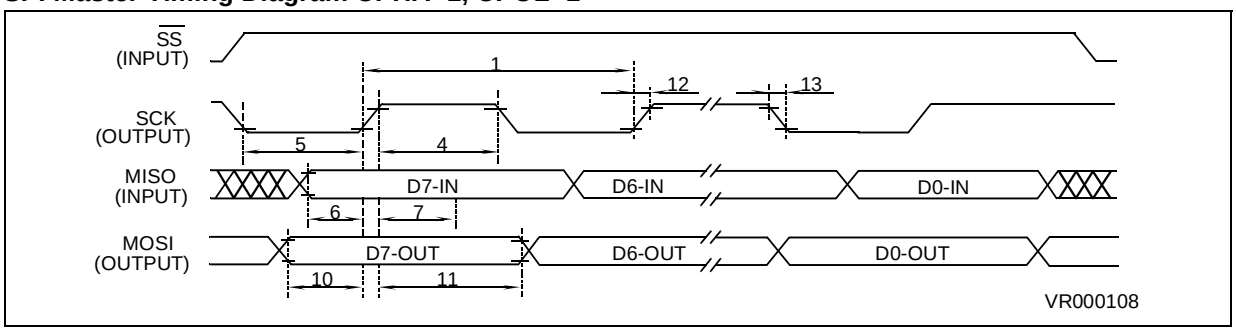
SPI Master Timing Diagram CPHA=0, CPOL=1



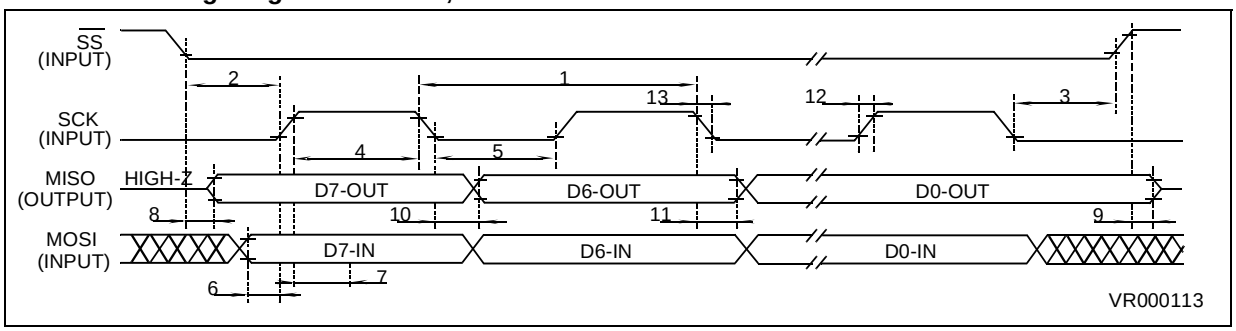
SPI Master Timing Diagram CPHA=1, CPOL=0



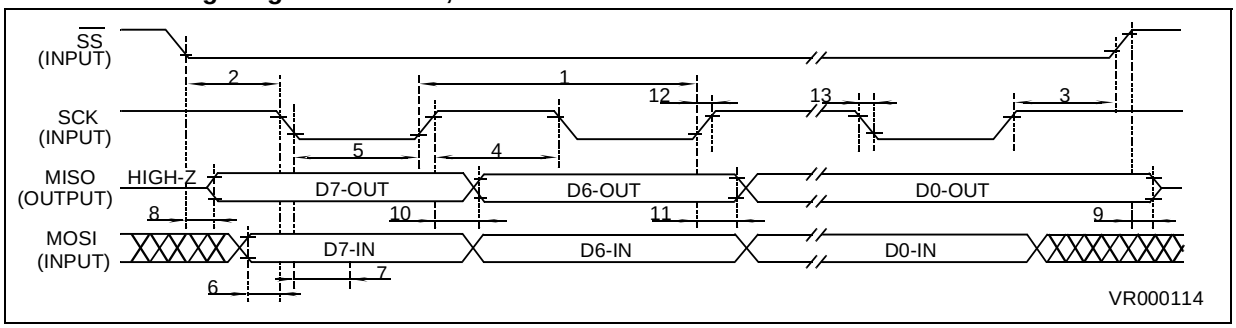
SPI Master Timing Diagram CPHA=1, CPOL=1



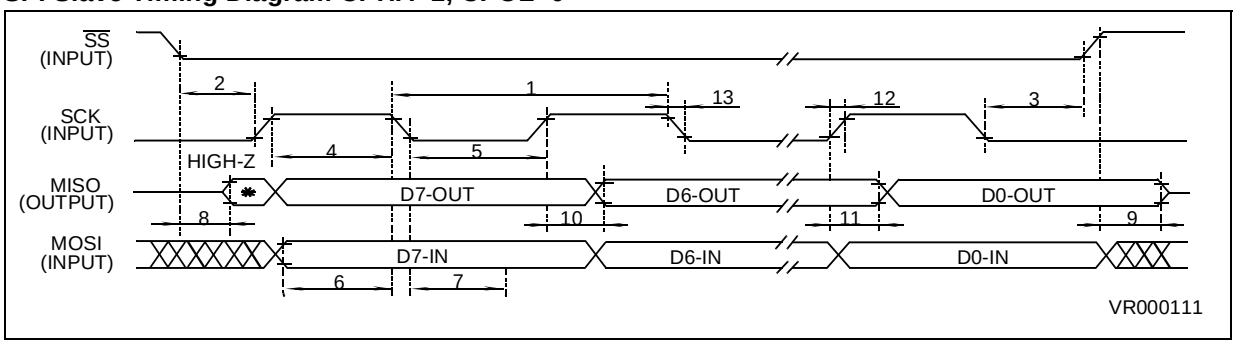
SPI Slave Timing Diagram CPHA=0, CPOL=0



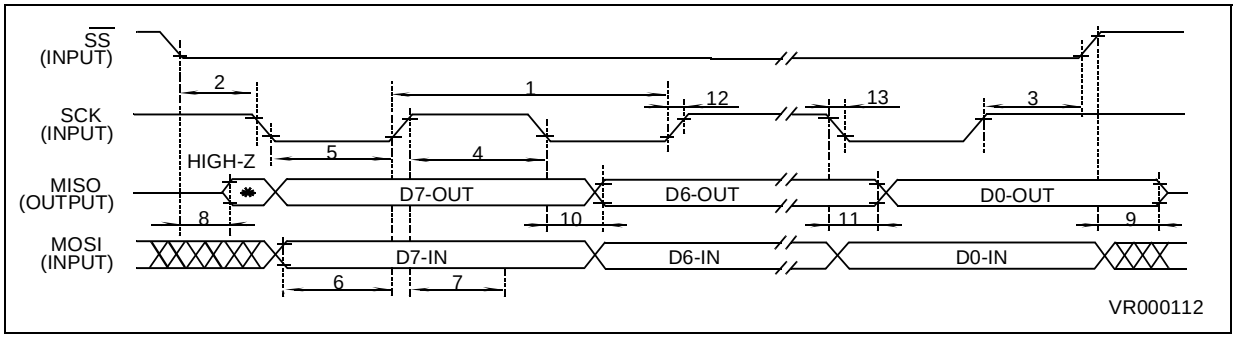
SPI Slave Timing Diagram CPHA=0, CPOL=1



SPI Slave Timing Diagram CPHA=1, CPOL=0



SPI Slave Timing Diagram CPHA=1, CPOL=1



I²C/DDC-BUS TIMING TABLE

(V_{DD} = 5V ± 10%, T_A = –40°C to +105°C, C_{Load} = 50pF, f_{INTCLK} = 24MHz, unless otherwise specified)

Symbol	Parameter		Formula	Protocol Specifications				Unit
				Standard I²C		Fast I²C		
				Min	Max	Min	Max	
f_{INTCLK}	Internal Frequency (Slave Mode)			2.5		2.5		MHz
f_{SCL}	SCL clock frequency			0	100	0	400	kHz
T_{BUF}	Bus free time between a STOP and START condition			4.7		1.3		µs
T_{HIGH}	SCL clock high period			4.0		0.6		µs
T_{LOW}	SCL clock low period	Standard Mode Fast Mode	T_{HIGH} – 3 x Tck 2 x (T_{HIGH} – 3 x Tck)	4.7		 1.3		µs
$T_{HD:STA}$	Hold time START condition. After this period, the first clock pulse is generated		T_{LOW} + Tck	4.0		0.6		µs
$T_{SU:STA}$	Set-up time for a repeated START condition		T_{LOW} + T_{HIGH} – $T_{HD:STA}$	4.7		0.6		µs
$T_{HD:DAT}$	Data hold time	FREQ[2:0] = 000 FREQ[2:0] = 001 FREQ[2:0] = 010 FREQ[2:0] = 011	3 x Tck 4 x Tck 4 x Tck 10 x Tck	0 (1;2)		0 (1;2)	0.9 (1;3)	ns
$T_{SU:DAT}$	Data set-up time (Without SCL stretching)		T_{LOW} – $T_{HD:DAT}$	250 (1)		100 (1)		ns
	Data set-up time (With SCL stretching)	FREQ[2:0] = 000 FREQ[2:0] = 001 FREQ[2:0] = 010 FREQ[2:0] = 011	7 x Tck 15 x Tck 15 x Tck 31 x Tck					
T_R	Rise time of both SDA and SCL signals				1000 (1)	20+0.1Cb (1)		ns
T_F	Fall time of both SDA and SCL signals				300 (1)	20+0.1Cb(1)		ns
$T_{SU:STO}$	Set-up time for STOP condition		T_{LOW} + T_{HIGH} – $T_{HD:STA}$	4.0		0.6		ns
Cb	Capacitive load for each bus line				400		400	pF

Note:

(1) Value guaranteed by design

(2) The device must internally provide a hold time of at least 300 ns for the SDA signal in order to bridge the undefined region of the falling edge of SCL

(3) The maximum hold time of the START condition has only to be met if the interface does not stretch the low period of SCL signal

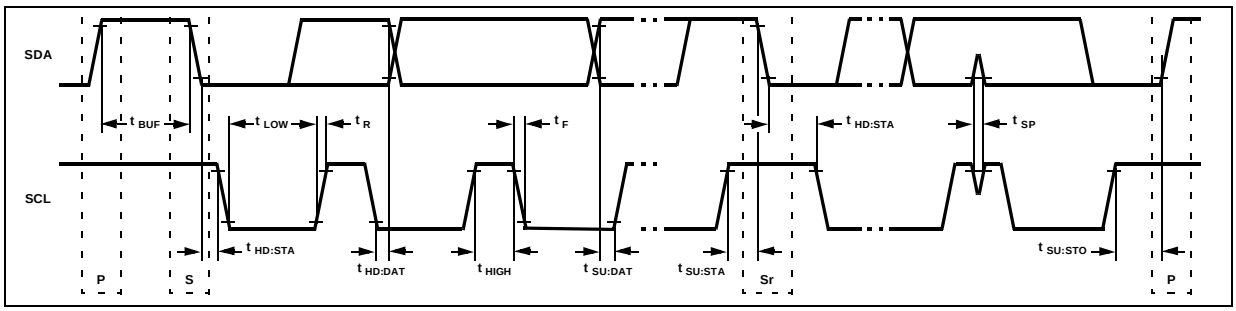
Legend:

Tck = INTCLK period = OSCIN period when OSCIN is not divided by 2;
2 x OSCIN period when OSCIN is divided by 2;
OSCIN period x PLL factor when the PLL is enabled.

Cb = total capacitance of one bus line in pF

FREQ[2:0] = Frequency bits value of I²C Own Address Register 2 (I2COAR2)

I²C TIMING


SDA
SCL
t BUF
t LOW
t R
t F
t HD:STA
t SP
t HD:STA
t HD:DAT
t HIGH
t SU:DAT
t SU:STA
t SU:STO
P
S
Sr
P

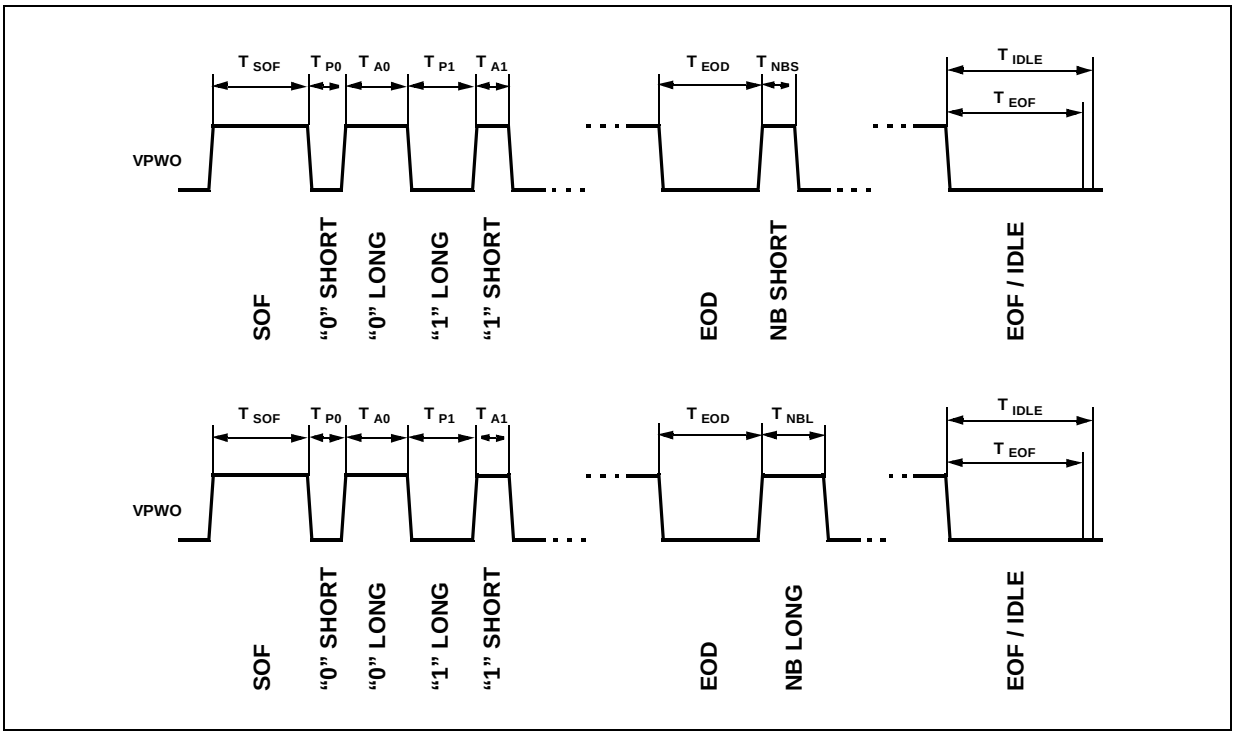
J1850 BYTE LEVEL PROTOCOL DECODER TIMING TABLE

(V_{DD} = 5V ± 10%, T_A = –40°C to +105°C, C_{Load} = 50pF, f_{INTCLK} = 24MHz, unless otherwise specified)

Symbol	Parameter	Value			Unit	Note
		Receive Mode		Transmission Mode		
		Min	Max	Nominal		
T_F	Symbols Filtered	0	≤ 7	-	µs	(1)(2)
T_{IB}	Invalid Bit Detected	> 7	≤ 34	-	µs	(1)(2)
T_{P0}	Passive Data Bit “0”	> 34	≤ 96	64	µs	(1)(2)(3)
T_{A0}	Active Data Bit “0”	> 96	≤ 163	128	µs	(1)(2)(3)
T_{P1}	Passive Data Bit “1”	> 96	≤ 163	128	µs	(1)(2)(3)
T_{A1}	Active Data Bit “1”	> 34	≤ 96	64	µs	(1)(2)(3)
T_{NBS}	Short Normalization Bit	> 34	≤ 96	64	µs	(1)(2)(3)
T_{NBL}	Long Normalization Bit	> 96	≤ 163	128	µs	(1)(2)(3)
T_{SOF}	Start Of Frame Symbol	> 163	≤ 239	200	µs	(1)(2)(3)
T_{EOD}	End Of Data Symbol	> 163	≤ 239	200	µs	(1)(2)(3)
T_{EOF}	End Of Frame Symbol	> 239	-	280	µs	(1)(2)(3)
T_{BRK}	Break Symbol	> 239	-	300	µs	(1)(2)(3)
T_{IDLE}	Idle Symbol	> 280	-	300	µs	(1)(2)(3)

Note:
(1) Values obtained with internal frequency at 24 MHz (INTCLK), with CLKSEL Register set to 23.
(2) In Transmission Mode, symbol durations are compliant to nominal values defined by the J1850 Protocol Specifications.
(3) All values are reported with a precision of ±1 µs.

J1850 PROTOCOL TIMING



A/D EXTERNAL TRIGGER TIMING TABLE

(V_{DD} = 5V ± 10%, T_A = –40°C to +105°C, C_{Load} = 50pF, f_{INTCLK} = 24MHz, unless otherwise specified)

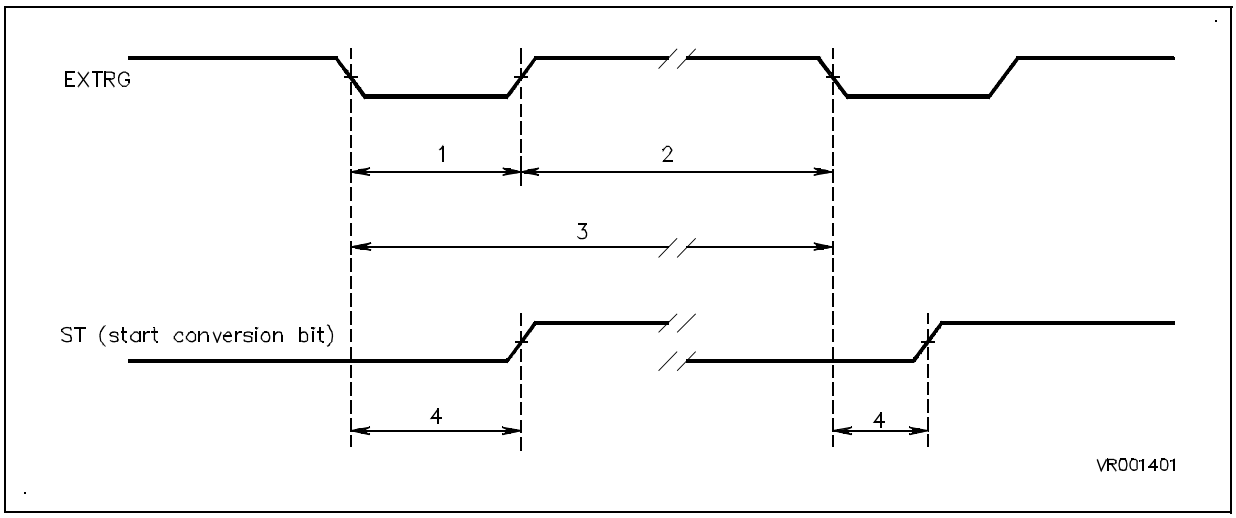
N°	Symbol	Parameter	Value (Note)			Unit
			Formula	Min.	Max.	
1	Tw_{LOW}	External trigger pulse width	1.5 x Tck	62.5	-	ns
2	Tw_{HIGH}	External trigger pulse distance	1.5 x Tck	62.5	-	ns
3	Tw_{EXT}	External trigger active edges distance	138 x n x Tck	n x 5.75	-	µs
4	Td_{STR}	EXTRG falling edge and first conversion start	0.5 x Tck 1.5 x Tck	20.8	62.5	ns

Note: The value in the left hand column shows the formula used to calculate the timing minimum or maximum from the oscillator clock period, standard timer prescaler and counter programmed values.
The value in the right hand two columns show the timing minimum and maximum for an internal clock (INTCLK) at 24MHz.

Legend:
Tck = INTCLK period = OSCIN period when OSCIN is not divided by 2;
2*OSCIN period when OSCIN is divided by 2;
OSCIN period / PLL factor when the PLL is enabled.

n = number of autoscanned channels ($1 \le n \le 8$)

A/D EXTERNAL TRIGGER TIMING



A/D CHANNEL ENABLE TIMING TABLE

(V_{DD} = 5V ± 10%, T_A = –40°C to +105°C, C_{Load} = 50pF, f_{INTCLK} = 24MHz, unless otherwise specified)

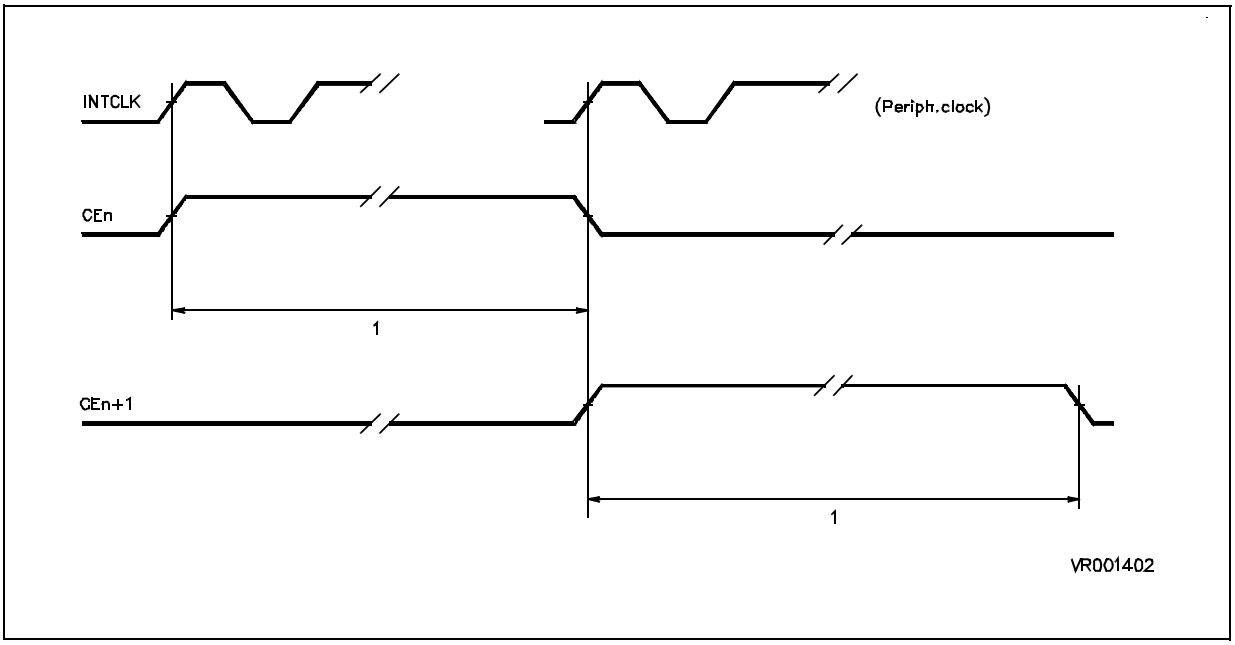
N°	Symbol	Parameter	Value (Note)			Unit
			Formula	Min.	Max.	
1	Tw_{EXT}	CEn Pulse width	138 x n x Tck	n x 5.75	-	µs

Note: The value in the left hand column shows the formula used to calculate the timing minimum or maximum from the oscillator clock period, standard timer prescaler and counter programmed values.
The value in the right hand two columns show the timing minimum and maximum for an internal clock (INTCLK) at 24MHz.

Legend:
Tck = INTCLK period = OSCIN period when OSCIN is not divided by 2;
2*OSCIN period when OSCIN is divided by 2;
OSCIN period / PLL factor when the PLL is enabled.

n = number of autoscanned channels (1 ≤ n ≤ 8)

A/D CHANNEL ENABLE TIMING



A/D ANALOG SPECIFICATIONS

(V_{DD} = 5V ± 10%, T_A = –40°C to +105°C, f_{INTCLK} = 24MHz, unless otherwise specified)

Parameter	Typical	Minimum	Maximum	Units (1)	Notes
Conversion time		138		INTCLK	(2)(6)
Sample time		85		INTCLK	(6)
Power-up time		60		µs	(6)
Resolution	8	8		bits	
Monotonicity	GUARANTEED				
No missing codes	GUARANTEED				
Zero input reading		00		Hex	(6)
Full scale reading			FF	Hex	(6)
Offset error	0.3		0.5	LSBs	(1)(4)(6)
Gain error			0.6	LSBs	(4)(6)
DLE (Diff. Non Linearity error)			0.6	LSBs	(4)(6)
ILE (Int. Non Linearity error)			1.0	LSBs	(4)(6)
TUE (Absolute Accuracy)		–1.0	1.0	LSBs	(4)(6)
Input Resistance	1.3	0.8	2.7	kΩ	(3)(5)(6)
Hold Capacitance	1.4			pF	(5)(6)
Input Leakage			±1	µA	(6)

Note:

(1) "1LSBideal" has a value of $AV_{DD}/256$

(2) Including sample time

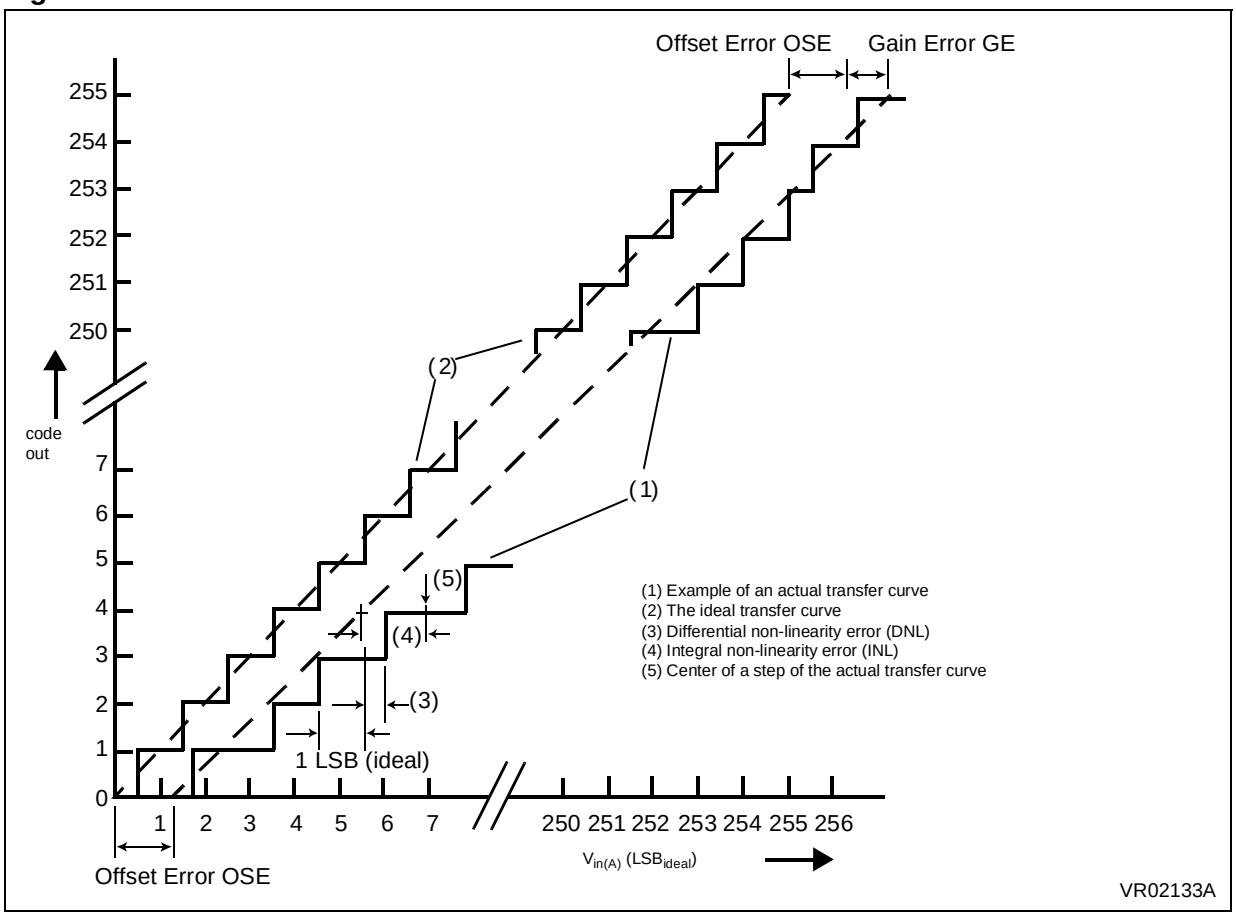
(3) This is the internal series resistance before the sampling capacitor

(4) This is a typical expected value, but not a tested production parameter.
If V(i) is the value of the i-th transition level ($0 \le i \le 254$), the performance of the A/D converter has been evaluated as follows:
OFFSET ERROR= deviation between the actual V(0) and the ideal V(0) (=1/2 LSB)
GAIN ERROR= deviation between the actual V(254) and the ideal V(254) - V(0) (ideal V(254)=AV_{DD}-3/2 LSB)
DNL ERROR= max {[V(i) - V(i-1)]/LSB - 1}
INL ERROR= max {[V(i) - V(0)]/LSB - i}
ABS. ACCURACY= overall max conversion error

(5) Simulated value, to be confirmed by characterisation.

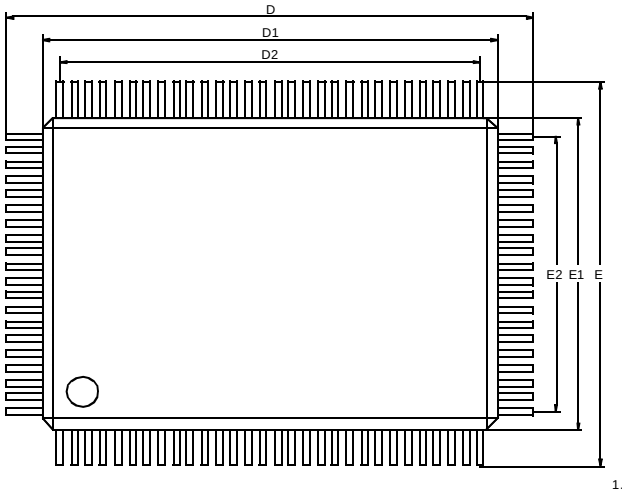
(6) The specified values are guaranteed only if an overload condition occurs on a maximum of 2 non-selected analog input pins and the absolute sum of input overload currents on all analog input pins does not exceed ±10 mA.

Figure 124. A/D Conversion Characteristics



12 PACKAGE MECHANICAL DATA

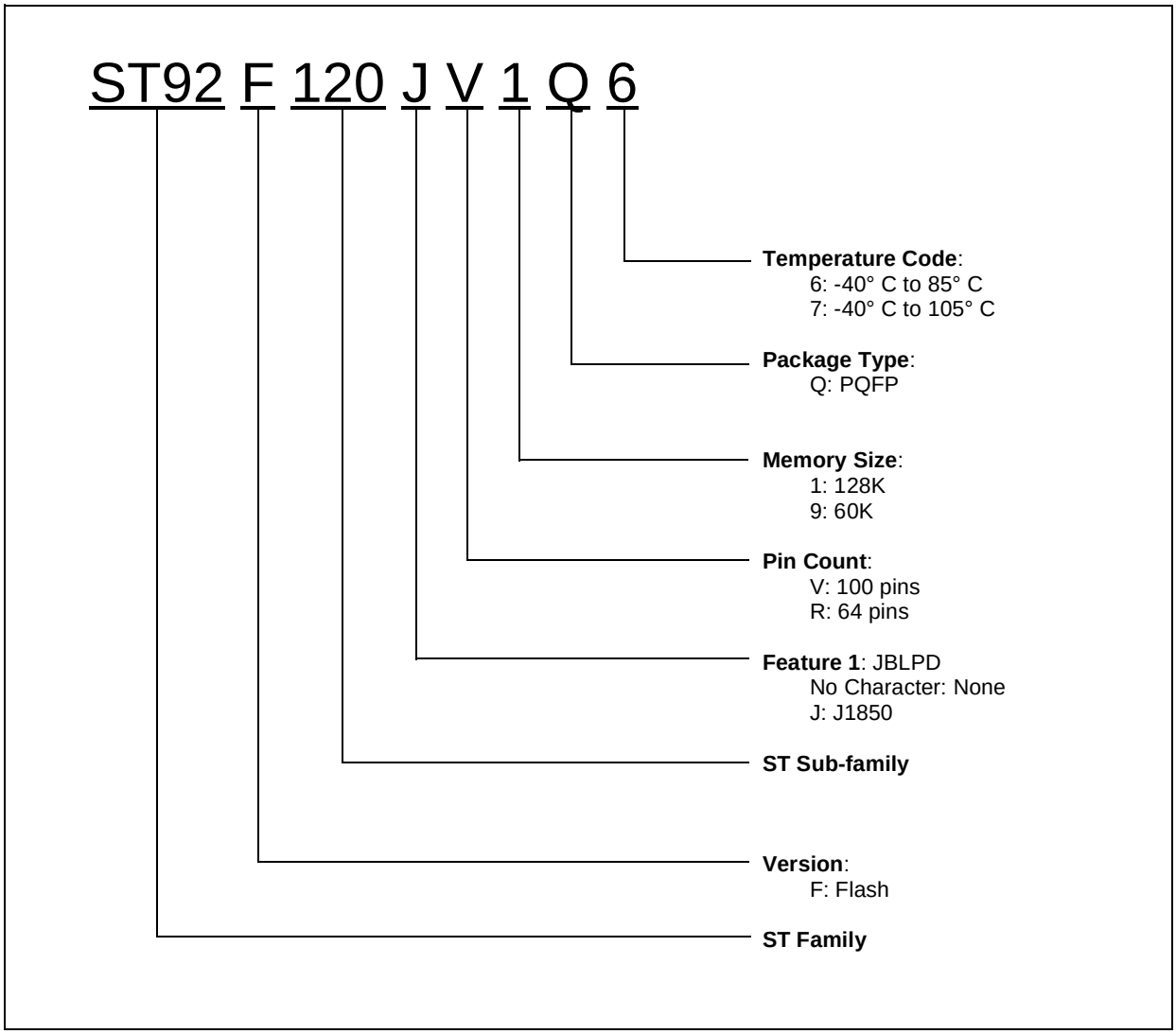
Figure 125. 100-Pin Plastic Quad Flat Package



Dim.	mm			inches		
	Min	Typ	Max	Min	Typ	Max
A			3.40			0.134
A1	0.25		0.50	0.010		0.020
A2	2.50	2.70	2.90	0.098	0.106	0.114
b	0.22		0.40	0.009		0.016
c	0.11		0.23	0.004		0.009
D		23.20			0.913	
D1		20.00			0.787	
D2		18.85			0.742	
E		17.20			0.677	
E1		14.00			0.551	
E2		12.35			0.486	
e		0.65			0.026	
L	0.73	0.88	1.03	0.029	0.035	0.041
	Number of Pins					
N	100					

13 DEVICE ORDERING INFORMATION

Figure 126. Device Types



14 SUMMARY OF CHANGES

Rev.	Page	Section	Main changes	Date
1.2	Page 1.1	1.2	Added History pages. Updated ARC, STIM, WUIMU, SPI7, DDC, FEE.SCI replaced ZAD by ADC8 Changed order of on-chip peripheral chapters.	12/01/97
	1		In phrase "66 (80 ...) I/O bits"; replaced "bits" with "pins". In phrase "8-bit Analog to Digital..." replaced with "Two 8-bit Analog to Digital..." Second col. In phrase "In addition, there is an 8 channel..." replaced with "In addition, there are two 8 channel..."	
	4		At the end of the sentence, added: "On QFP80 version, only 10 input channels are available." End of second col. In phrase "Low power Run, Wait for interrupt, and STOP modes are also av." replaced with: "Low power Run (SLOW), Wait for Interrupt, low power Wait for interrupt, HALT and STOP modes are also available."	
	5		In picture: . Added in RCCU output signals the new: CLOCK2/8 . Removed from A/D conv. blocks the labels "zad_0" and "zad_1" . Added in A/D CONV. 1 signals the EXTRG item replaced INT3:0 by INT 6:0	
	6		Same as Page 5. . Added RWN in the CPU signals (bold/no italic) leaving the italic one Changed the picture as follow:	
	7		. Added in RCCU output signals the new: CLOCK2/8 . Removed from A/D conv. block the labels "zad" . Duplicated A/D block like at page 5 (same input signals) replaced INT3:0 by INT 6:0	

Rev.	Page	Section	Main changes	Date
	Page 8 9		- Changed the picture as follows: . Added in RCCU output signals the new: CLOCK2/8 . Removed from A/D conv. block the labels "zad" . Duplicated A/D block like at page 6 (same input signals) . Added RWN in the CPU signals (bold/no italic) leaving the italic one replaced INT3:0 by INT 6:0 - For consistency with block diagram: ASN instead of AS+overbar DSN instead of DS+overbar RWN instead of R/W+overbar on W RESETN instead of RESET+overbar HW0SW1 instead of HW0_SW1 ST92E120 twice replaced with ST92E120/F120 Inside ASN item (at the end) removed the last sentence "Under program.." Inside DSN item (at the end) removed the last sentence "It can be..." Inside RWN item (at the end) removed the last sentence "It can be..." Added: "On QFP100 version, RWN is also available as true pin." VDD item: Added "Two internally connected pins are available" VSS item: Added description "Two internally connected pins are available" Added a new item for VPP: VPP. High voltage Power Supply for Eprom memory (only on ST92E120; on ST92F120 the pin is not connected). P2.0...P9.7 item: highlighted that on QFP80, only P8.0-P8.1 is available and P9 is not available at all	

Rev.	Page	Section	Main changes	Date
			Added alternate functions to pin labels. Swapped OSCIN OSCOUT	
	10		Put a note saying:" NC = Not Connected (no physical bonding wire)"	
			Same as Page 10. Shifted labelling on pins 1-8, 100 and 73-81.	
	11		Two extra tables inserted before the AF table. The first table is the list of Power Supply pins with reference to the package.	
	12		The second is the dedicated pins table (ASN/DSN/RWN/RESET/OSCIN/OSCOUT /HW0SW1...).	
			In AF table: WPU in the header substituted with "Weak Pull-up".	
			AF table duplicated: the first for ST92E120 and the second	
			for ST92F120. The same done for Power Supply and dedicated pins tables.	
			P21: TIMPB0: replaced with TINPB0	
			P23:"Output A": replaced with "Output B"	
			P25: TIMPB1: replaced with TINPB1	
			P46: Cleared the cells where "SDAI" and "DDC - I2C Input Data"	
			Replaced "SDAO" with "SDAI/SDAO	
			Replaced "O" with "I/O"	
			Replaced "DDC - I2C Data Output" with "DDC - I2C Data"	
			P64: Replaced "SCHMITT TRIGGER" with "HIGH HYST. S.T."	
			In the table of F120 only:	
			P22: Replaced YES with NO	
			P23: Replaced YES with NO	
			In AF table, added new column giving the conf. after reset.	
			For E120 table, filled up all cells (one for each Pxy item), with BID-WPU.	
			For F120 table, filled up P8.2-P8.7 and P9 cells with BID-WPU; all the others are INPUT.	
			P6.0: added CLOCK2/8	
			P6.1: replaced INT1 by INT6	
			P7.-P7.7 and P8.0-P8.7 Removed names in brackets and added ADC0 and ADC1 to descriptions	
			4.5 Added A/D0 and A/D 1 ext trigger	
			Removed Note about Port0 and Port1	
		23	- "Internal Weak Pull-up" section.	
			Removed the first sentence: "The internal ... 57 kohm."	
	16		"TTL/CMOS Input" section.	
	17		Added a cross ref to the current paragraph 9.4 (INPUT/OUTPUT BIT CONFI-GURATION) at the end of the paragraph:	

Rev.	Page	Section	Main changes	Date
	18	24	Replaced ST92E120 with ST92E120/F120 In WAIT FOR INTERRUPT MODE item, removed the sentence: "Under this mode,... (LP WFI)." Created a new item: "LOW POWER WAIT FOR INTERRUPT" with the following description: "Combining SLOW Mode and Wait For Interrupt mode it is possible to reduce the power consumption by more than 80%." Second column: "Watchdog counter ..." sentence rewritten. At the end of STOP MODE item, added: "The counter is active only when the oscillation has already taken place: this means that 1-2 ms must be added to take into account of the first phase of the oscillator restarting."	
	20	26	- Fig. 10: PAGE REGISTERS: replaced with PAGED REGISTERS	
	21	27	- Second col. Last sentence "A table of available...": removed.	
	29	35	End of first col."...into four16 Kbytes pages.". Removed space.	
	38	44	End of second col.Chapter 3: cross ref updated	
	50	59	Replaced the Chapter 3 new doc specs Added a register and Section on protection strategy).	
	51	60	Moved para. on register map to page 63	
	53	62	- A note in the picture added: "RAM addresses are repeated each 4 Kbytes inside the segment 20h" - FLASH OTP - 128 bytes: the starting address 210000h	
	54	63	Replaced the address with 211F80h	
	55	64	Same as Page 51	
			ZAD0 and ZAD1 removed and replaced with "Res." Added registers for AD0 and AD1: . AD0 - Page 63(3F) All registers from R240 to R255 . AD1 - Page 61(3D) All registers from R240 to R255	
		65	Note removed.	
	57	73	Added detailed register map (7 pages)	
	63	79	Changed table	
	64	80	Changed 2 tables	
	65	86	Changed figure	
	86	103	WUIMU section revised (edited text throughout)	
	87	104	Changed figure: added 16 divider to CK128 input to STIM and P6.0 output with 8 div.	
	95	112	Changed figure: like previous page	
	226	248	Changed values in two tables	
	227	249	Added rows in second table. Changed RESET overbar to RESETN.	
1.3	18 112		Changed P2.2 and P2.3 to Pure OD output, no WPU In Column 2: “I/O pins are set to Bidirectional Weak-Pull-Up or High impedance input. See Table of I/O port alternate functions.	12/16/97

Rev.	Page	Section	Main changes	Date
1.4	10		Replaced ST9_DDC by ST9 I2C macrocell. Updated ST9_FEE, ST9_DMA, ST9MFT, ST9_WUIMU macrocells to new rev levels. Removed references to DDC throughout document	02/06/98
	10		Removed WKU3, ICAPB1, ICAPA0	
	11		Removed ICAPA0	
	15		Added footnote **ST92F120 only. Removed P6.6, 4.1 and 3.1. Added WKUP"**, ICAPB1** and ICAPA0**. Inserted VDD VSS on pins 11,12 and 54,55 Swapped VDD/VSS on pins 76,75	
	16		Added footnote **ST92F120 only. Removed P6.6, 4.1 and 3.1. Added *to WKUP3/ P6.7, ICAPB1/P4.1 . Added WKUP"**, ICAPB1** and ICAPA0** Inserted 2x VDD VSS Swapped VDD/VSS on pins 93,92	
	17		Added 2x VDD/VSS to table 1	
	19		Removed P3.0	
	20		Removed P6.6. Moved WKUP3 from P6.7 to P7.4	
	22		Added 2x VDD/VSS to table 3	
	24		Added ICAPA0 to P3.2. Removed P3.0. Moved ICAPB1 from P4.1 to P4.3	
	25		Moved WKUP 3 from P6.7 to 7.4. Removed P6.6	
	66		Changed DDC to I2C	
	70		Changed DDC to I2C. Changed I2C register names	
1.5	5,7		Removed P3.0, P4.1, P6.7, P6.6	2/18/98
	6,8		Removed P3.0, P6.6	
	9		Updated list of I/O lines	
	19		Changed P3.2 to Input	
			Changed P3.7 to YES Weak Pull-up	
			Changed P4.4 to NO Weak Pull-up	
			Changed P4.5 to YES Weak Pull-up	
	20		Changed P5.2 to NO Weak Pull-up	
			Changed P5.3 to YES Weak Pull-up	
			Changed P5.7 to NO Weak Pull-up	
	21-22		Table format problem corrected	
	247		In VOL item, replaced IOH by IOL.	
1.6	1,4		Changed total I/O pins to 62 and 78. Changed 30MHz to 25MHz	
	9		Added paragraph on EMC features	
	10		Removed comment "ST92F120 only"	
	11		Removed comment "ST92F120 only" and removed ICAPB1 from P4.1 and WKUP3 from P6.7	
	14		Added ICAPA0 to P3.2 and ICAPB1 to P4.3 removed ICAPB1 from P4.1	
	15		Change P6.4 removed hi hys note from Schmitt Trigger. Removed WKUP3 from P6.7	
	16		Added WKUP3 To P7.4	
	19		SDAI/SDAO renamed SDA, SCLI/SCLO renamed SCL. Added note 1 to Schmitt Trigger	
	20		Added note 2 to Schmitt Trigger P6.4	
	65,67		Changed Wakeup register mapping to page 59	

Rev.	Page	Section	Main changes	Date
1.7	**Page 1.6**	**Page 1.7**	Removed ST92E120 information. Added JBLPD, updated RCCU, INT, FEE, WDG, STIM, EFT, I2C, Ext Mem I/F, and Electrical characteristics	
	1	1	Changed 25 MHz to 24 MHz, added J1850 feature. Removed EPROM version. Added device summary.	
	2-3	2-6	Added 1 level to table of contents	
	4	7	Changed 25 MHz to 24 MHz, added J1850 feature.	
	5-6	-	Removed E120 diagrams	
	8	9	Add JBLPD, removed P6.7 added CLOCK2	
	9	10-15	Expanded Pin description detail. Added VPWO	
	10	-	Removed E120 diagrams	
	11	16	Removed note on E120, duplicate SCK removed on P3.7 and WKUP1 corrected to WKUP5 on P5.0	
	12-17	-	Removed E120 table.	
	17	18	Added VWO to table 2 and section 1.3.5 using I/O AF. Added P0 and P1 to gen purpose I/O para at bottom of first col.	
	20	21	Added VPWI to P6.5	
	22	22	Added CLOCK2 to P9.6	
	61	-	Removed E120 memory map	
	62-64	61-63	Changed Flash reg from 3 to 4 bytes, and Testflash sector from 21 to 23. Added Flash sector labels.	
	65	64	Changed WU reg. page from 59 to 57, added JBLPD to page 23	
	66-73	65-72	Added page ref in right column. Changed WU reg. page from 59 to 57, added JBLPD to page 23	
			renamed MFT registers FLAGR, ICR and IVR to T_FLAGR, T_ICR, T_IVR	
			renamed SCI registers IVR, ISR to S_IVR, S_ISR	
			renamed ADC registers ICR, IVR to AD_ICR, AD_IVR	
	126	121	MEMSEL bit, bit 4 of EMR2, change to 1 (Reset value=1Fh).	
1.8	**Page 1.7**	**Page 1.8**	Updated description of DMA in MFT and SCI chapters	04/30/99
		1-11	Removed PFQ80	
			Added TQFP64	
			Overbar format applied to active low pin names	
		16-23	CTS changed to RTS	
			I/O table format changed.	
		112	Stop mode description changed. Removed table of register reset values.	
		287-289	Electrical characteristics added for IINJ RTHJA, VIH/VIL VHYS for Schmitt trigger	

Rev.	Page	Section	Main changes	Date
1.9	**Page**	**Page**		06/04/99
	1.8	**1.9**		
	1	1	Changed V to R on TQFP64 devices in device summary;	
	8	8	Changed figure caption ”JR instead of V9T”. Added extra memory sizes and optional J1850 block	
	9	9	Same changes as page 8 plus added note to SCI1 “on some versions only”	
	16	16	Added TXCAN0 and RXCAN0 with note. Changed note to VPP and VPWO. Added VPWI to P6.5. Added VREG to pin 28	
	17	17	Added TXCAN0, RXCAN0, TXCAN1 and RXCAN1 with note. Added note to VPP. Added VREG to pin 31 and 43	
	18	18	Added figure for Vreg. Added Vreg in table 1. Removed “not used” from VPP	
	109	109	Removed Notes on Ceramic Resonator from line three	
	-	110	Added 1 new page on Ceramic resonators	
	287	288	Added ESD susceptibility 2000V	
	289	290	Added note to describe typical values. Changed TBD to 10 mA in note for IOV. Added values to RWPU.	
	290	291	Changed typ and max values. Added max values per Mhz. Added note about run mode.	
	292	293	Replaced x by 10 in formula. Added ≥ sign. Added min values. Added note “formula guaranteed by design” Added note “measurement points are ....”	
	293	294	Same changes as previous table	
	294	295	Added TNFR . Changed value of TPLK typ. and PLL jitter.	
	297	299	Added note “formula guaranteed by design” Added note “measurement points are ....”	
	298	300	Added note “formula guaranteed by design” Added note “measurement points are ....”	
	299	301	Added note “formula guaranteed by design” Added note “measurement points are ....” Removed TEXTCLK Added 10 in formula to TWECKD and TWEICD. Changed min values for these parameters.	
	302	304	Added note “values guaranteed by design”	
	305	307	Removed VIL and VIH. Replaced TBD by specified values.	
	306	308	Added max values for TR and TF	
	310	312	Changed values of offset error Moved values for DLE ILE and TUE from typ to max.	

Rev.	Page	Section	Main changes	Date
2.0	**Page**	**Page**		09/08/99
	1.9	**2.0**		
			Updated ST9_SCI and ST9_MFT macrocells	
	1	1	Changed minimum instruction time (83 ns)	
	63	63	Table 16; R248/Page 9: changed “Reserved” to “MFT0 and MFT1”	
	66	66	Table 17: Changed Register Names for R244, R245, R246, R247 (MFT1 block) and R240, R241, R242 and R243 (MFT0 block) Changed reset values for TCR1, T_ICR1, OACR1, OBCR1, TCR0, T_ICR0, OACR0, OBCR0	
	67	67	I2C block: Changed R254 to R255 for I2CIMR and added I2CECCR register (R254)	
	68	68	Changed reset value for PRLR (JBLPD block)	
	107	107	Added description of bit 1	
	121	121	Changed reset value and comment on Bit 4 of EMR2	
	151	151	10.3.5 chapter (Register description): added note	
	158	158	Removed EFT2 and EFT3	
	180	180	Changed R254 to R255 for IDMR	
	289	290	Added “Input Threshold” typical values and removed TBD for V_{IH} and V_{IL} For V_{HYS}: Removed min and max values. Changed TBD to typical values	
	290	291	In note 4: 100mA instead of 10mA	
	291	292	For I_{DDTR}: Removed TBD (max value) and added typical value	
	295	296	“RCCU characteristics” table: added “input threshold” typical values for V_{IHRS} and V_{ILRS}; Removed TBD for min and max values. Changed TBD to 800 for V_{HYRS} typical value	
	296	297	Changed TBD to values. Changed -0.3 to 0.4 for V_{ILCK} Min value	
	300	301	(2) instead of (1) for note 2	
	301	302	Removed max values; Changed TBD to 2Tck + 10	
	303	304	Changed TBD to 350	
	307	308	Changed TBD to values	
2.1	Page	Page		
	2.0	2.1		
			Changed document status (“preliminary data” instead of “product preview”)	
	10	10	Added V_{REG} description	
	17	17	Figure 9, added pin numbers	
	24	24	Changed Halt mode description	01/31/00
	119	119	8.2.7 section, removed Warning	
	273	276	EOFM_M instead of EOF_M	
	290	293	Removed min values (0.7 x V_{DD}) for V_{IH} ; Removed max values (0.8) for V_{IL} Changed 900 to 600 for V_{HYS} (Standard Schmitt Trigger)	
	293	296	Flash/EEPROM Specifications Table: Added one row (flash endurance -40°C +105°C)	
2.2			<u>Removal of TQFP64 package and 36K Flash Memory</u>	
	8	1.1	TQFP64 Version deleted	
	45	3.2	Modification of Flash Memory Structure	
	59	4.3	Location of Reset Vector	4 Sept 00
	72	5.2	Ext. Watchdog/Reset Vectors	
	120	8.3	Bit 5: Upper/lower Memory Access	
	296	11	Max. EEPROM Parameter values modified	

Rev.	Page	Section	Main changes	Date
2.3			Addition of TQFP100 Package V_{PP} changed to V_{TEST} V_{33} changed to V_{REG} "On future versions" added for V_{REG} Removed "ST9OLD" references (in EMR1 register description on page 115)	8 Dec 00
	7	1.1.3	Addition of "Flash and E3PROM Memories" section	
2.4	11	1.3	Updated Figure 3, Figure 4 and Figure 5.	06 Mar 01
	56	3.7.1	Replaced Section 3.7.1 on page 56 by "Code Update Routine" on page 52	
	111	7.6	Added Figure 55	
	163	10.4	Added Note on Bi Capture mode on using A0 bit in MFT Section 10.4.2.11	
	295	11	Reference to AN1102 changed to AN1152 on page 295.	
	292	11	Added V_{IH} min and V_{IL} max values	
	318	12	Updated package mechanical data	
	319	13	Updated Figure 126 (temperature code).	
2.5			Deletion of TQFP100 Package. Added WDIN on P5.3 Changed EEPROM size for 60K devices. Changed section 3 on page 39. Added one note in section 3.7.1 on page 52. Changed section 10.3 on page 135: removed references to the 3 separate interrupt channels (ICI, OCI and TOI), not present on the ST92F120. In section 10.7.4.1 on page 219: changed "address matched" section, changed "slave revceiver" section: removed "both holding the SCL line low) changed "slave transmitter" section: added "except on EV3-1) changed "how to release the SDA/SCL lines" section: added one sentence ("check that...") Changed note in section 10.7.5 on page 224: "The error event interrupt pending bit...while the error event flags are set" instead of "... until the error event flags are set". Swapped 0 and 1 for I2CSR.DMASTOP bit. Changed V_{INOD} in Absolute Maximum Ratings table on page 287. Changed Thermal Characteristics value on page 287. Changed AV_{DD} min value in recommended operating conditions table on page 287. Changed V_{IH} in DC electrical characteristics value	02 Aug 01
2.6			Changed status of the document: "datasheet" instead of "preliminary data"	12 Sept 02
	39	3	Replaced Bootrom by TestFlash in descriptions	
	40	3.2.1	Changed Table 7 and Table 8	
	51	3.6.1	Added note on NVWPR register description	
	55	4.2	Changed Figure 27	
	56	4.2	Changed Figure 28	
	57	4.2	Changed Figure 29	
	204	10.6	SPI section: changed register names: SPCR instead of CR, etc	
	206	10.6.4.1	Changed SPIF clearing sequence description	
	210	10.6.4.4	Changed Figure 103	
	211	10.6.4.5	Changed MODF clearing sequence description	
	288	11	Changed DC electrical characteristics table (V_{IH}, V_{IL}, V_I)	
	306	11	Changed I²C/DDC-bus Timing Table	

Notes: