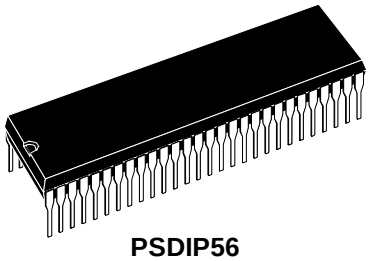
ST92196A ST92T196 ST92E196

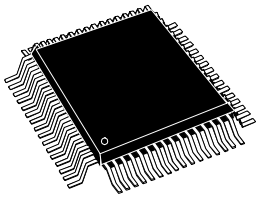
8/16-BIT MCU FOR TV APPLICATIONS WITH UP TO 96K ROM, ON-SCREEN-DISPLAY AND 1 OR 2 DATA SLICERS

- Register file based 8/16 bit Core Architecture with RUN, WFI, and HALT modes
- -10 to 75°C Operating Temperature Range
- 24 MHz Operation @5 V ±10%
- Min. instruction cycle time: 165 ns at 24 MHz
- 32 - 96 Kbytes ROM, 1 - 3 Kbytes static RAM
- 256 bytes of Register file
- 384 bytes of display RAM (OSDRAM)
- 56-pin Shrink DIP and TQFP64 packages
- 37 fully programmable I/O pins
- Flexible Clock controller for OSD, Data slicer and Core clocks, running from one single low frequency external crystal
- Enhanced Display Controller with rows of up to 63 characters per row
  - 50/60Hz and 100/120 Hz operation
  - 525/625 lines operation, 4/3 or 16/9 format
  - interlaced and progressive scanning
  - 18x26 or 9x13 character matrix
  - 384 (18x26) characters, or 1536 (9x13) characters definable in ROM by user
  - 512 possible colors, in 4x16-entry palettes
  - 2 x 16-entry palettes for Foreground, and 2 x 16-entry palettes for Background
  - 8 levels of translucency on Fast Blanking
  - Serial, Parallel and Extended Parallel Attribute modes
  - Mouse pointers user-definable in ROM
  - 7 character sizes in 18x26 mode, 4 in 9x13
  - Rounding, Fringe, Scrolling, Flashing, Shadowing, Italics, Semi-transparent
- I^2C Multi-Master / Slave with 4 channels
- Serial Communications Interface (SCI)*
- Serial Peripheral Interface (SPI)
- 8-channel A/D converter with 6-bit accuracy
- 16-bit Watchdog timer with 8-bit prescaler
- 14-bit Voltage Synthesis for tuning reference voltage with 2 outputs for 2 tuners
- 16-bit standard timer with 8-bit prescaler
- 16-bit Multi-Function timer*
- Eight 8-bit programmable PWM outputs

* On some devices



PSDIP56



TQFP64

- NMI and 6 external interrupts
- 1 or 2 data slicers for Closed Captioning and Extended Data Service data extraction, on 2 independent video sources. Support for FCC V-Chip and Gemstar bitstream decoding
- Infra-Red signal digital pre-processor
- 2-channel Sync error detection with integrated Sync extractor
- Rich instruction set and 14 addressing modes
- Versatile Development Tools, including C-Compiler, Assembler, Linker, Source Level Debugger, Emulator and Real-Time Operating Systems from third-parties
- Windows Based OSD Font and Screen Editor

DEVICE SUMMARY

Device	ROM	RAM	Slicers	SCI	MFT
ST92196A7	96K	3K	2	1	1
ST92196A6		2K	2	1	1
ST92196A4	64K		2	-	1
ST92196A3			1	-	-
ST92196A2	48K	1K	1	-	-
ST92196A1	32K		1	-	-

1 GENERAL DESCRIPTION

1.1 INTRODUCTION

The ST92196A family brings the enhanced ST9 register-based architecture to a new range of high-performance microcontrollers specifically designed for TV applications. Their 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 ST9 MCU devices support low power consumption and low voltage operation for power-efficient and low-cost embedded systems.

1.1.1 Core Architecture

The nucleus of the ST92196A is the enhanced ST9 Core that includes the Central Processing Unit (CPU), the register file, the interrupt and DMA controller.

Three independent buses are controlled by the Core: a 16-bit memory bus, an 8-bit register addressing 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. Many opcodes specify byte or word operations, the hardware automatically handles 16-bit operations and accesses.

For interrupts or subroutine calls, the CPU uses a system stack in conjunction with the stack pointer (SP). A separate user stack has its own SP. The separate stacks, without size limitations, can be in on-chip RAM (or in Register File) or off-chip memory.

1.1.2 Instruction Set

The ST9 instruction set consists of 94 instruction types, including instructions for bit handling, byte (8-bit) and word (16-bit) data, as well as BCD and Boolean formats. Instructions have been added to facilitate large program and data handling through the MMU, as well as to improve the performance and code density of C Function calls. 14 addressing modes are available, including powerful indirect addressing capabilities.

The ST9's bit-manipulation instructions are set, clear, complement, test and set, load, and various logic instructions (AND, OR, and XOR). Math functions include add, subtract, increment, decrement, decimal adjust, multiply, and divide.

1.1.3 Operating Modes

To optimize performance versus the power consumption of the device, ST9 devices now support a range of 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).

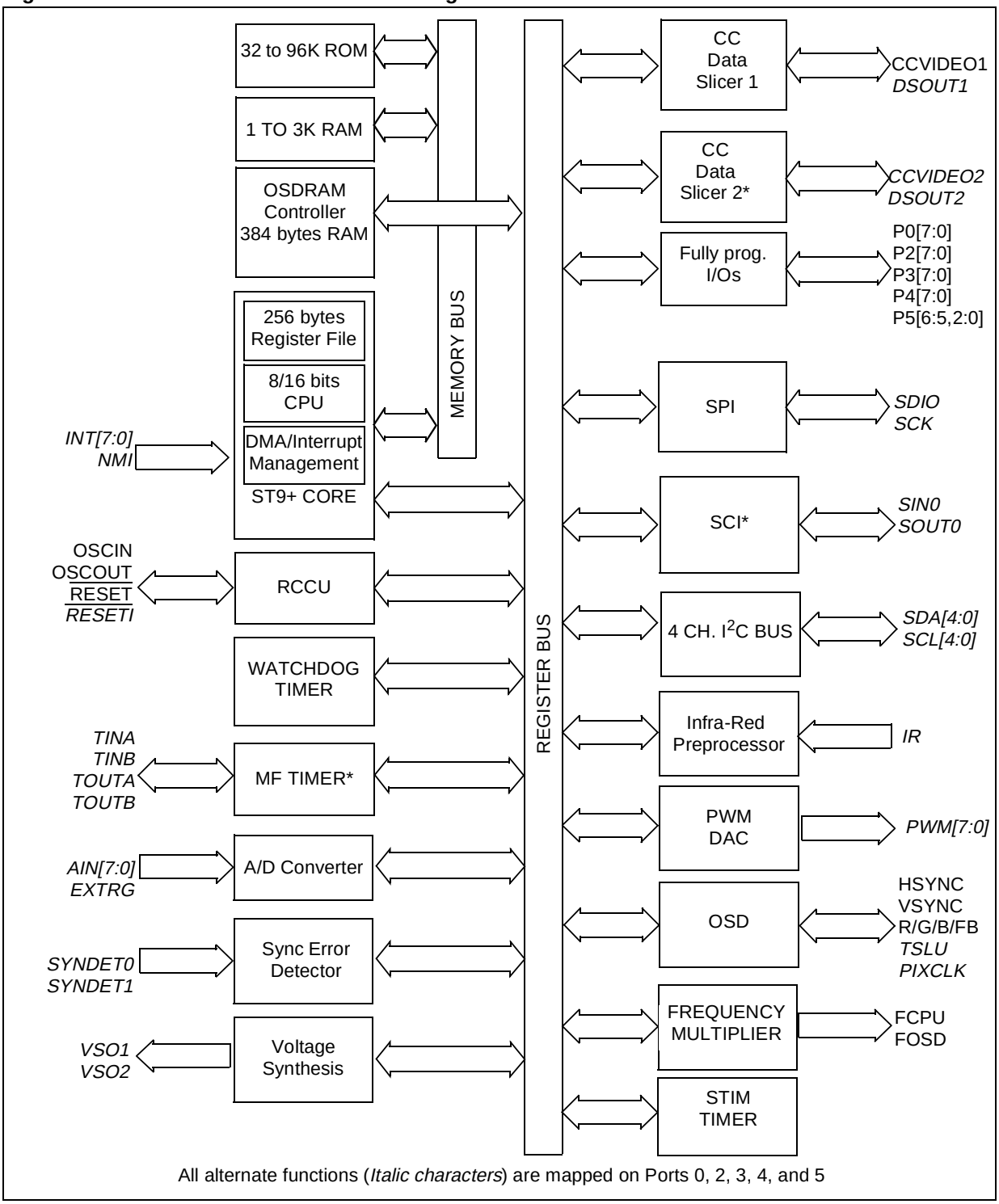
Slow Mode. Power consumption can be significantly reduced by running the CPU and the peripherals at reduced clock speed using the CPU Prescaler and CCU Clock Divider.

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 programmable via the CCU. In this mode, the power consumption of the device can be reduced by more than 95% (Low Power WFI).

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.

INTRODUCTION (Cont'd)

Figure 1. ST92196A Architectural Block Diagram



*On some devices only

Note: Not all peripherals are available on all device versions. Please check the Device Summary on page 1.

INTRODUCTION (Cont'd)

1.1.4 On-chip Peripherals

OSD Controller

The On Screen Display displays closed caption or extended service format data received from the on-chip data slicers or any text or menu data generated by the application. Rows of up to 63 characters can be displayed with two user-definable fonts. Colors, character shape and other attributes are software programmable. Support is provided for mouse or other pointing devices.

Parallel I/O Ports

The ST9 is provided with dedicated lines for input/output. These lines, grouped into 8-bit ports, can be independently programmed to provide parallel input/output or to carry input/output signals to or from the on-chip peripherals and core e.g. SCI and Multifunction Timer. All ports have active pull-ups and pull-down resistors compatible with TTL loads. In addition pull-ups can be turned off for open drain operation and weak pull-ups can be turned on to save chip resistive pull-ups. Input buffers can be either TTL or CMOS compatible.

Multifunction Timer

The multifunction timer has a 16-bit Up/Down counter supported by two 16-bit Compare registers and two 16-bit input capture registers. Timing resolution can be programmed using an 8-bit prescaler.

Serial Communications Controller

The SCI provides an asynchronous serial I/O port using two DMA channels. Baud rates and data formats are programmable. Controller applications can further benefit from the self test and address wake-up facility offered by the character search mode.

I^2C Bus Interface

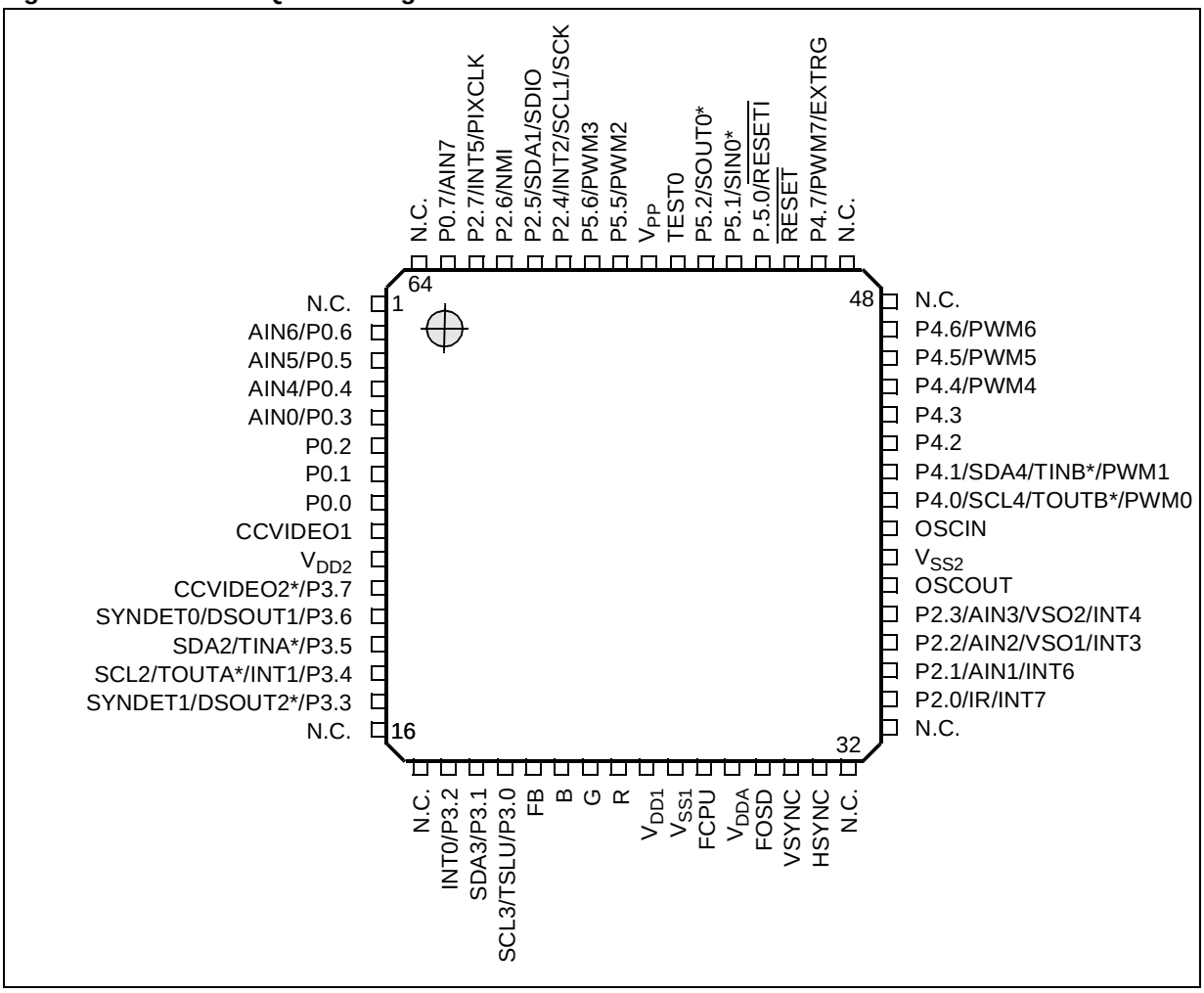
The I^2C bus is a synchronous serial bus for connecting multiple devices using a data line and a clock line. Multimaster and slave modes are supported. Up to four channels are supported. The I^2C interface supports 7-bit addressing. It operates in multimaster or slave mode and supports speeds of up to 666.67 kHz. Bus events (Bus busy, slave address recognised) and error conditions are automatically flagged in peripheral registers and interrupts are optionally generated.

Analog/Digital Converter

The ADC provides up to 8 analog inputs with on-chip sample and hold. Conversion can be triggered by a signal from the MFT.

1.2 PIN DESCRIPTION

Figure 2. 64-Pin Thin QFP Package Pin-Out

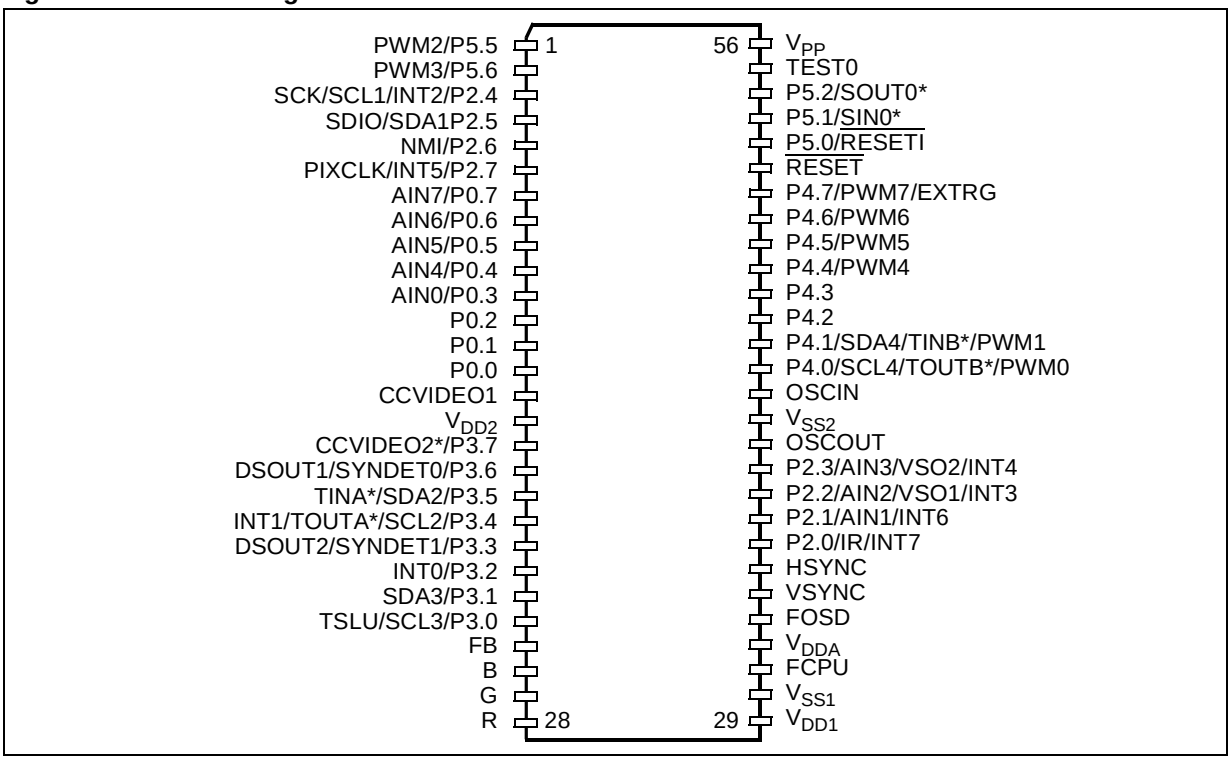


N.C. = Not connected

* Not available on some devices.

PIN DESCRIPTION (Cont'd)

Figure 3. 56-Pin Package Pin-Out



* Not available on some devices.

Table 1. Power Supply Pins

Name	Function	SDIP56	QFP64
V_{DD1}	Main Power Supply Voltage (2 pins internally connected)	29	25
V_{DD2}		16	10
V_{SS1}	Analog and Digital Circuit Ground (2 pins internally connected)	30	26
V_{SS2}		41	39
V_{DDA}	Analog Circuit Supply Voltage	32	28
V_{PP}	EPROM Programming Voltage. Must be connected to V_{DD} in normal operating mode.	56	56

Table 2. Primary Function pins

Name	Function	SDIP56	QFP64
OSCIN	Oscillator input	42	40
OSCOUT	Oscillator output	40	38
$\overline{RESET}$	Reset to initialize the ST9	51	51
HSYNC	Video Horizontal Sync Input (Schmitt trigger)	35	31
VSYNC	Video Vertical Sync input (Schmitt trigger)	34	30
R	Red video analog DAC output	28	24
G	Green video analog DAC output	27	23
B	Blue video analog DAC output	26	22
FB	Fast Blanking analog DAC output	25	21
CCVIDEO1	Closed Caption Composite Video input 1 (2V +/- 3 dB)	15	9
FCPU	CPU frequency multiplier filter output	31	27
FOSD	OSD frequency multiplier filter output	33	29
TEST0	Test input (must be tied to V_{DD})	55	55

PIN DESCRIPTION (Cont'd)

1.2.1 I/O Port Configuration

All ports can be individually configured as input, bidirectional, output, or alternate function. Refer to the Port Bit Configuration Table in the I/O Port Chapter.

No I/O pins have any physical weak pull-up capability (they will show no pull-up if they are programmed in the "weak pull-up" software mode).

Input levels can be selected on a bit basis by choosing between TTL or CMOS input levels for I/O port pin except for P2.(5:4,0), P3.(6:3,1:0), P4.(1:0) which are implemented with a Schmitt trigger function.

All port output configurations can be software selected on a bit basis to provide push-pull or open drain driving capabilities. For all ports, when configured as open-drain, the voltage on the pin must never exceed the V_{DD} power line value (refer to Electrical characteristics section).

1.2.2 I/O Port Reset State

I/Os are reset asynchronously as soon as the $\overline{\text{RESET}}$ pin is asserted low.

All I/O are forced by the Reset in bidirectional, high impedance output due to the lack of physical pull-up except P5.0 (refer to the Reset section) which is forced into the "Push-Pull Alternate Function" mode until being reconfigured by software.

Warning

When a common pin is declared to be connected to an alternate function input and to an alternate function output, the user must be aware of the fact that the alternate function output signal always inputs to the alternate function module declared as input.

When any given pin is declared to be connected to a digital alternate function input, the user must be aware of the fact that the alternate function input is always connected to the pin. When a given pin is declared to be connected to an analog alternate function input (ADC input for example) and if this pin is programmed in the "AF-OD" mode, the digital input path is disconnected from the pin to prevent any DC consumption.

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 2.0	Schmitt trigger	Push-Pull/OD	No	Bidirectional
Port 2[3:1]	TTL/CMOS	Push-Pull/OD	No	Bidirectional
Port 2[5:4]	Schmitt trigger	Push-Pull/OD	No	Bidirectional
Port 2[7:6]	TTL/CMOS	Push-Pull/OD	No	Bidirectional
Port 3.0	Schmitt trigger	Push-Pull/OD	No	Bidirectional
Port 3.1	Schmitt trigger	Push-Pull/OD	No	Bidirectional
Port 3.2	TTL/CMOS	Push-Pull/OD	No	Bidirectional
Port 3[6:3]	Schmitt trigger	Push-Pull/OD	No	Bidirectional
Port 3.7	TTL/CMOS	Push-Pull/OD	No	Bidirectional
Port 4.[1:0]	Schmitt trigger	Push-Pull/OD	No	Bidirectional
Port 4.[7:2]	TTL/CMOS	Push-Pull/OD	No	Bidirectional
Port 5.0	TTL/CMOS	Push-Pull/OD	No	Push-Pull AF Out
Port 5[6:1]	TTL/CMOS	Push-Pull/OD	No	Bidirectional

Legend: OD = Open Drain, AF = Alternate Function

Table 4. I/O Port Alternate Functions

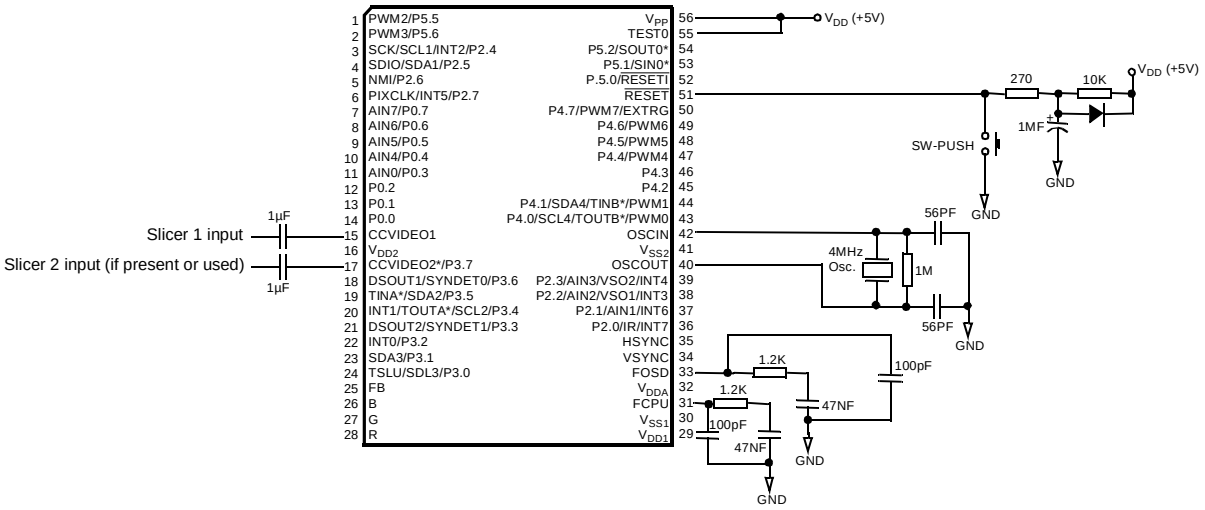
Port Name	General Purpose I/O	Pin No.		Alternate Functions		
		SDIP56	TQFP64			
P0.0	All ports useable for general purpose I/O (input, output or bidirectional)	14	8		I/O	
P0.1		13	7		I/O	
P0.2		12	6		I/O	
P0.3		11	5	AIN0	I	A/D Analog Data Input 0
P0.4		10	4	AIN4	I	A/D Analog Data Input 4
P0.5		9	3	AIN5	I	A/D Analog Data Input 5
P0.6		8	2	AIN6	I	A/D Analog Data Input 6
P0.7		7	63	AIN7	I	A/D Analog Data Input 7
P2.0		36	34	IR	I	IFR Infrared Input
				INT7	I	External Interrupt 7
P2.1		37	35	AIN1	I	A/D Analog Data Input 1
				INT6	I	External Interrupt 6
P2.2		38	36	INT3	I	External Interrupt 3
				AIN2		A/D Analog Data Input 2
				VSO1	O	Voltage Synthesis Converter Output 1
P2.3		39	37	INT4	I	External Interrupt 4
				AIN3	I	A/D Analog Data Input 3
				VSO2	O	Voltage Synthesis Converter Output 2
P2.4		3	59	INT2	I	External Interrupt 2
				SCL1	I/O	I^2C Channel 1 Serial Clock
				SCK	O	SPI Serial Clock Output
P2.5		4	60	SDIO	I/O	SPI Serial Data
				SDA1	I/O	I^2C Channel 1 Serial Data
P2.6		5	61	NMI	I	Non Maskable Interrupt Input
P2.7		6	62	INT5	I	External Interrupt 5
				PIXCLK	O	Pixel Clock (after divide-by-2) Output
P3.0		24	20	SCL3	I/O	I^2C Channel 3 Serial Clock
				TSLU	O	Translucency Digital Video Output
P3.1		23	19	SDA3	I/O	I^2C Channel 3 Serial Data
P3.2		22	18	INT0	I	External Interrupt 0
P3.3		21	15	SYNDET1	I	Sync Error Detector Input 1
				DSOUT2*	O	Data Slicer Comparator Output 2
P3.4		20	14	INT1	I	External Interrupt 1
				SCL2	I/O	I^2C Channel 2 Serial Clock
				TOUTA*	O	MFT Timer output A
P3.5		19	13	TINA*	I	MFT Timer input A
				SDA2	I/O	I^2C Channel 2 Serial Data

GENERAL DESCRIPTION

Port Name	General Purpose I/O	Pin No.		Alternate Functions		
		SDIP56	TQFP64			
P3.6	All ports useable for general purpose I/O (input, output or bidirectional)	18	12	SYNDET0	I	Sync Error Detector Input 0
				DSOUT1	O	Data Slicer Comparator Output 1
P3.7		17	11	CCVIDEO2*	I	Closed Caption Composite Video input 1 (2V +/- 3 dB)
P4.0		43	41	SCL4	I/O	I^2C Channel 4 Serial Clock
				TOUTB*	O	MFT Timer output B
				PWM0	O	PWM D/A Converter Output 0
P4.1		44	42	TINB*	I	MFT Timer input B
				SDA4	I/O	I^2C Channel 4 Serial Data
				PWM1	O	PWM D/A Converter Output 1
P4.2		45	43		I/O	
P4.3		46	44		I/O	
P4.4		47	45	PWM4	O	PWM D/A Converter Output 4
P4.5		48	46	PWM5	O	PWM D/A Converter Output 5
P4.6		49	47	PWM6	O	PWM D/A Converter Output 6
P4.7		50	50	EXTRG	I	A/D Converter External Trigger Input
				PWM7	O	PWM D/A Converter Output 7
P5.0		52	52	RESETI	O	Internal Delayed Reset Output
P5.1		53	53	SIN0*	I	SCI Serial Comm. Interface Input
P5.2		54	54	SOUT0*	O	SCI Serial Comm. Interface Output
P5.5		1	57	PWM2	O	PWM D/A Converter Output 2
P5.6		2	58	PWM3	O	PWM D/A Converter Output 3

* Not available on some devices.

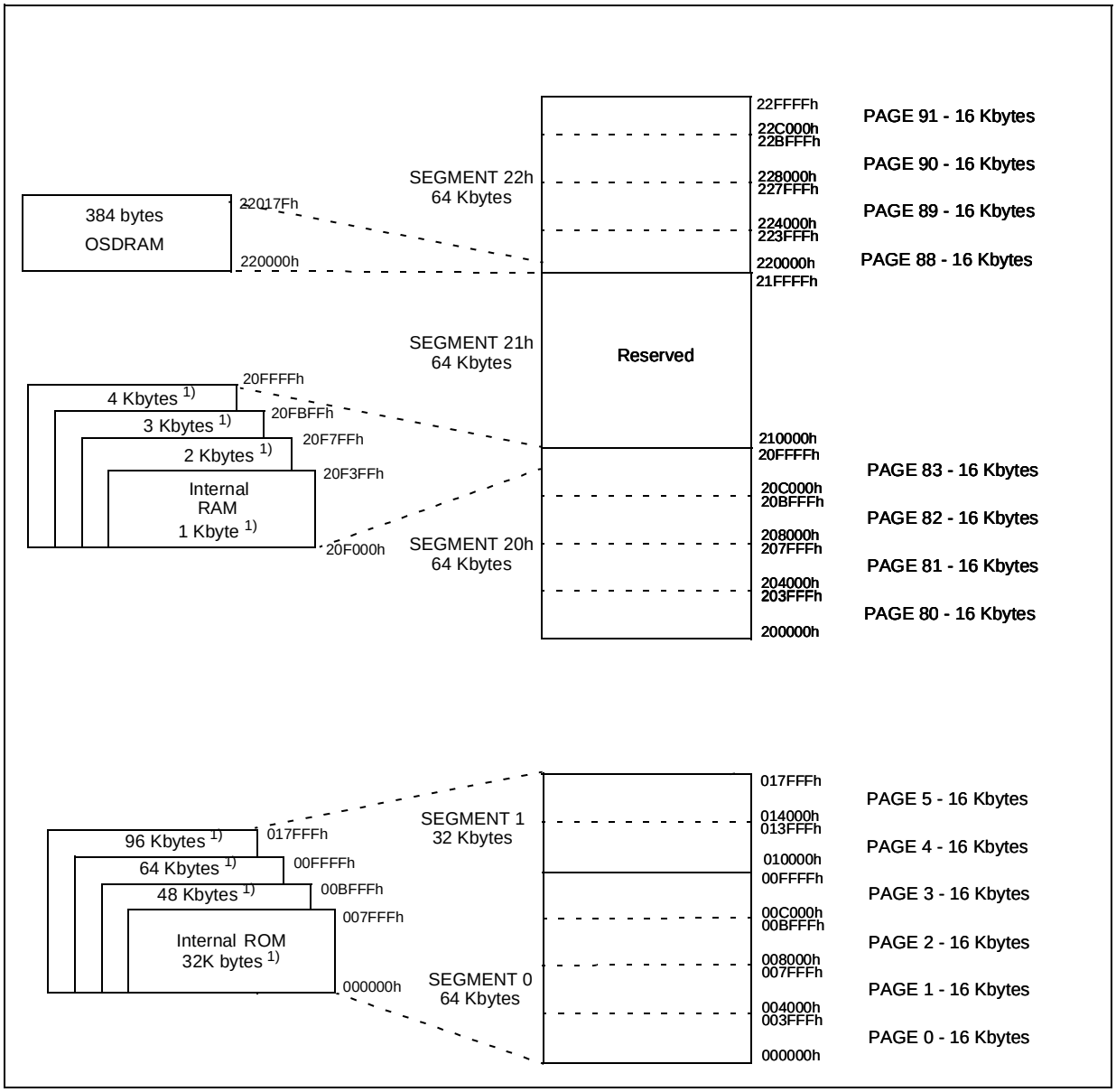
1.3 REQUIRED EXTERNAL COMPONENTS



Warning : The decoupling capacitors between analog and digital +5V (V_{DDA}, V_{DD1}, V_{DD2}), and ground (V_{SS1}, V_{SS2}) are not shown. Add a 100NF and a 4.7µF capacitor close to the corresponding pins if needed.

1.4 MEMORY MAP

Figure 4. ST92196A Memory Map



Note 1: ROM and RAM sizes are product dependent, refer to the Ordering Information section on page 234.

1.5 ST92196A REGISTER MAP

Table 6 contains the map of the group F peripheral pages.

The common registers used by each peripheral are listed in Table 5.

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.

Warning: 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 5. Common Registers

Function or Peripheral	Common Registers
SCI, MFT	CICR + NICR + DMA REGISTERS + I/O PORT REGISTERS
ADC	CICR + NICR + I/O PORT REGISTERS
WDT	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

ST92196A REGISTER MAP (Cont'd)

Table 6. Group F Pages Register Map

Resources available on the ST92196A device:

Register	Page															
	0	2	3	9	10	11	21	24	42	43	44	45	46	55	59	62
R255	Res.	Res.	Res.	Res.	MFT	Res.	MMU	SCI0	Res.	Res.	Res.	Res.	Res.	Res.	VS	Res.
R254	SPI	Port 3								TCC						
R253															Res.	
R252	WCR								OSD	Res.					PWM	
R251	WDT	Res.								IR/ SYNC ERR						
R250		Port 2														
R249																
R248				MFT												
R247	EXT INT	Res.	Port 5	Res.					Res.	Res.	I2C	DS0	DS1	RCCU		ADC
R246																
R245																
R244																
R243			Res.	MFT		STIM										
R242		Port 0	Port 4													
R241	Res.															
R240																

ST92196A REGISTER MAP (Cont'd)

Table 7. Detailed Register Map

Group F Page Dec.	Block	Reg. No.	Register Name	Description	Reset Value Hex.	Doc. Page
N/A	I/O Port 0:5	R224	P0DR	Port 0 Data Register	FF	69
		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	
	Core	R230	CICR	Central Interrupt Control Register	87	27
		R231	FLAGR	Flag Register	00	28
		R232	RP0	Pointer 0 Register	00	30
		R233	RP1	Pointer 1 Register	00	30
		R234	PPR	Page Pointer Register	54	32
		R235	MODER	Mode Register	E0	32
		R236	USPHR	User Stack Pointer High Register	xx	34
		R237	USPLR	User Stack Pointer Low Register	xx	34
		R238	SSPHR	System Stack Pointer High Reg.	xx	34
		R239	SSPLR	System Stack Pointer Low Reg.	xx	34
0	INT	R242	EITR	External Interrupt Trigger Register	00	56
		R243	EIPR	External Interrupt Pending Reg.	00	56
		R244	EIMR	External Interrupt Mask-bit Reg.	00	56
		R245	EIPLR	External Interrupt Priority Level Reg.	FF	57
		R246	EIVR	External Interrupt Vector Register	x6	57
		R247	NICR	Nested Interrupt Control	00	57
	WDT	R248	WDTHR	Watchdog Timer High Register	FF	81
		R249	WDTLR	Watchdog Timer Low Register	FF	81
		R250	WDTPR	Watchdog Timer Prescaler Reg.	FF	81
		R251	WDTCR	Watchdog Timer Control Register	12	81
		R252	WCR	Wait Control Register	7F	82
	SPI	R253	SPIDR	SPI Data Register	xx	190
		R254	SPICR	SPI Control Register	00	190
2	I/O Port 0	R240	P0C0	Port 0 Configuration Register 0	00	69
		R241	P0C1	Port 0 Configuration Register 1	00	
		R242	P0C2	Port 0 Configuration Register 2	00	
	I/O Port 2	R248	P2C0	Port 2 Configuration Register 0	00	
		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	00	
		R253	P3C1	Port 3 Configuration Register 1	00	
		R254	P3C2	Port 3 Configuration Register 2	00	

Group F 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	00	69
		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	00	
		R245	P5C1	Port 5 Configuration Register 1	00	
		R246	P5C2	Port 5 Configuration Register 2	00	
9	MFT	R240	DCPR	DMA Counter Pointer Register	xx	108
		R241	DAPR	DMA Address Pointer Register	xx	109
		R242	T_IVR	Interrupt Vector Register	xx	109
		R243	IDCR	Interrupt/DMA Control Register	C7	110
		R248	IOCR	I/O Connection Register	FC	110
10		R240	REG0HR	Capture Load Register 0 High	xx	101
		R241	REG0LR	Capture Load Register 0 Low	xx	101
		R242	REG1HR	Capture Load Register 1 High	xx	101
		R243	REG1LR	Capture Load Register 1 Low	xx	101
		R244	CMP0HR	Compare 0 Register High	00	101
		R245	CMP0LR	Compare 0 Register Low	00	101
		R246	CMP1HR	Compare 1 Register High	00	101
		R247	CMP1LR	Compare 1 Register Low	00	101
		R248	TCR	Timer Control Register	0x	102
		R249	TMR	Timer Mode Register	00	103
		R250	T_ICR	External Input Control Register	0x	104
		R251	PRSR	Prescaler Register	00	104
		R252	OACR	Output A Control Register	xx	105
		R253	OBCR	Output B Control Register	xx	106
		R254	T_FLAGR	Flags Register	00	107
		R255	IDMR	Interrupt/DMA Mask Register	00	108
11	STIM	R240	STH	Counter High Byte Register	FF	86
		R241	STL	Counter Low Byte Register	FF	86
		R242	STP	Standard Timer Prescaler Register	FF	86
		R243	STC	Standard Timer Control Register	14	86
21	MMU	R240	DPR0	Data Page Register 0	00	39
		R241	DPR1	Data Page Register 1	01	39
		R242	DPR2	Data Page Register 2	02	39
		R243	DPR3	Data Page Register 3	83	39
		R244	CSR	Code Segment Register	00	40
		R248	ISR	Interrupt Segment Register	x0	40
		R249	DMASR	DMA Segment Register	x0	40
	EXTMI	R246	EMR2	External Memory Register 2	0F	58

Group F Page Dec.	Block	Reg. No.	Register Name	Description	Reset Value Hex.	Doc. Page
24	SCI0	R240	RDCPR	Receiver DMA Transaction Counter Pointer	xx	201
		R241	RDAPR	Receiver DMA Source Address Pointer	xx	201
		R242	TDCPR	Transmitter DMA Transaction Counter Pointer	xx	201
		R243	TDAPR	Transmitter DMA Destination Address Pointer	xx	201
		R244	S_IVR	Interrupt Vector Register	xx	202
		R245	ACR	Address/Data Compare Register	xx	203
		R246	IMR	Interrupt Mask Register	x0	203
		R247	S_ISR	Interrupt Status Register	xx	204
		R248	RXBR	Receive Buffer Register	xx	205
		R248	TXBR	Transmitter Buffer Register	xx	205
		R249	IDPR	Interrupt/DMA Priority Register	xx	206
		R250	CHCR	Character Configuration Register	xx	207
		R251	CCR	Clock Configuration Register	00	207
		R252	BRGHR	Baud Rate Generator High Reg.	xx	208
		R253	BRGLR	Baud Rate Generator Low Register	xx	208
		R254	SICR	Input Control	03	208
		R255	SOCR	Output Control	01	208
42	OSD	R246	OSDBCR2	Border Color Register 2	x0	152
		R247	OSDBCR1	Border Color Register 1	x0	152
		R248	OSDER	Enable Register	00	153
		R249	OSDDR	Delay Register	xx	156
		R250	OSDFBR	Flag Bit Register	xx	157
		R251	OSDSLR	Scan Line Register	xx	158
		R252	OSDMR	Mute Register	xx	158
43	IR/SYNC ERR	R248	IRPR	Infrared Pulse Register	00	169
		R249	SYNCER	Sync Error Register	00	168
		R250	IRSCR	Infrared / Sync Control Register	00	168
	TCC	R253	MCCR	Main Clock Control Register	00	69
		R254	SKCCR	Skew Clock Control Register	00	69
44	I2C	R240	I2COAR	Own Address Register	00	175
		R241	I2CFQR	Frequency Register	00	176
		R242	I2CCTR	Control Register	01	177
		R243	I2CDR	Data Register	00	178
		R244	I2CSTR2	Status Register 2	00	178
		R245	I2CSTR1	Status Register 1	00	179

Group F Page Dec.	Block	Reg. No.	Register Name	Description	Reset Value Hex.	Doc. Page
45	DS0	R240	DS0DR1	Data Register 1	00	164
		R241	DS0DR2	Data Register 2	00	164
		R242	DS0DR3	Data Register 3	00	164
		R243	DS0DR4	Data Register 4	00	165
		R244	DS0CR1	Control Register 1	00	165
		R245	DS0CR2	Control Register 2	00	165
		R246	DS0MR	Monitor Register	00	166
46	DS1	R240	DS1DR1	Data Register 1	00	164
		R241	DS1DR2	Data Register 2	00	164
		R242	DS1DR3	Data Register 3	00	164
		R243	DS1DR4	Data Register 4	00	165
		R244	DS1CR1	Control Register 1	00	165
		R245	DS1CR2	Control Register 2	00	165
		R246	DS1MR	Monitor Register	00	166
55	RCCU	R240	CLKCTL	Clock Control Register	00	64
		R242	CLK_FLAG	Clock Flag Register	48, 28 or 08	64
59	PWM	R240	CM0	Compare Register 0	00	216
		R241	CM1	Compare Register 1	00	216
		R242	CM2	Compare Register 2	00	216
		R243	CM3	Compare Register 3	00	216
		R244	CM4	Compare Register 4	00	216
		R245	CM5	Compare Register 5	00	216
		R246	CM6	Compare Register 6	00	216
		R247	CM7	Compare Register 7	00	216
		R248	ACR	Autoclear Register	FF	217
		R249	CCR	Counter Register	00	217
		R250	PCTL	Prescaler and Control Register	0C	217
		R251	OCPL	Output Complement Register	00	218
		R252	OER	Output Enable Register	00	218
	VS	R254	VSDR1	Data and Control Register 1	00	213
		R255	VSDR2	Data Register 2	00	213
62	ADC	R240	ADDTR	Channel i Data Register	xx	221
		R241	ADCLR	Control Logic Register	00	221
		R242	ADINT	AD Interrupt Register	01	222

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

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 5. 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

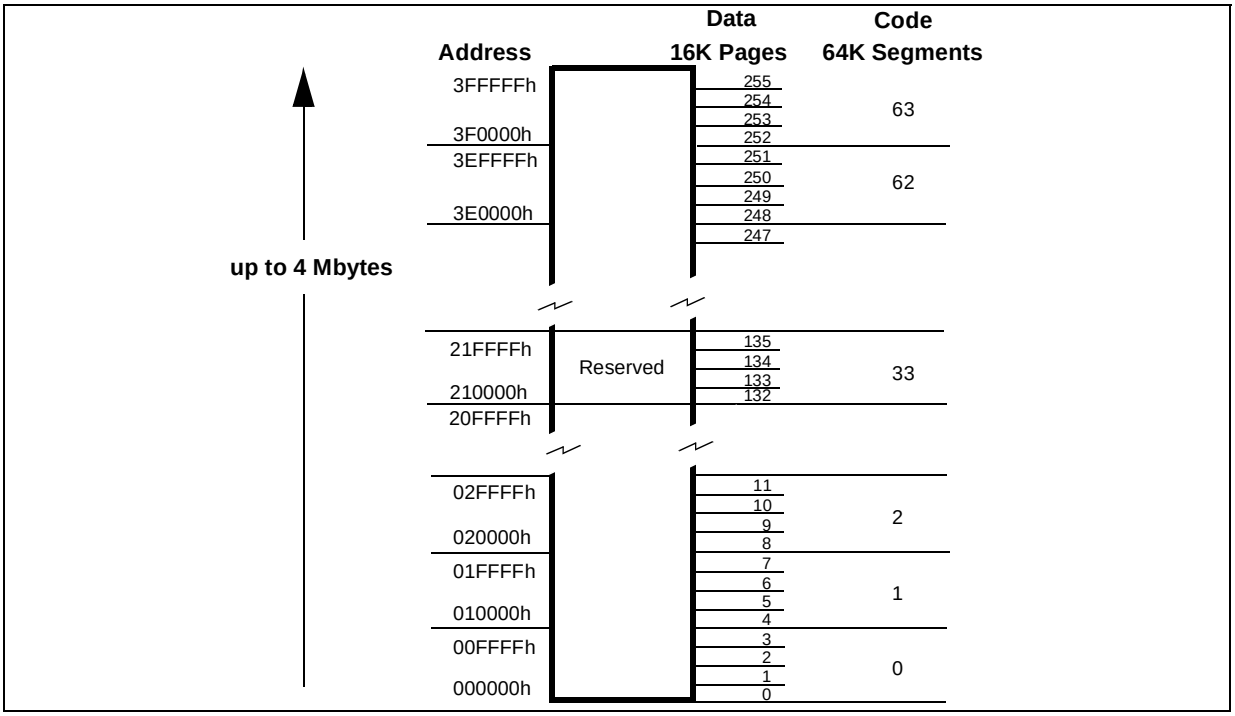
The Register File consists of (see Figure 6):

– 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 7.

Figure 5. Single Program and Data Memory Address Space



MEMORY SPACES (Cont'd)

Figure 6. Register Groups

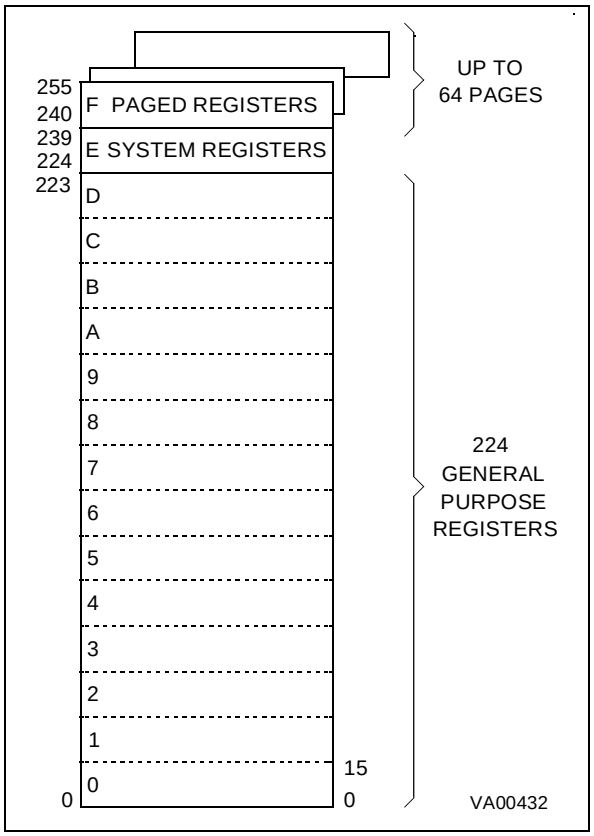


Figure 7. Page Pointer for Group F mapping

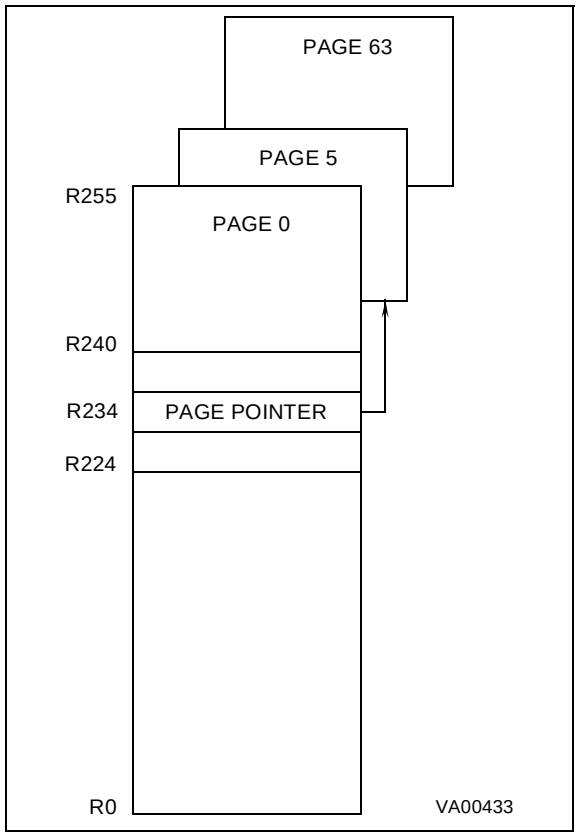
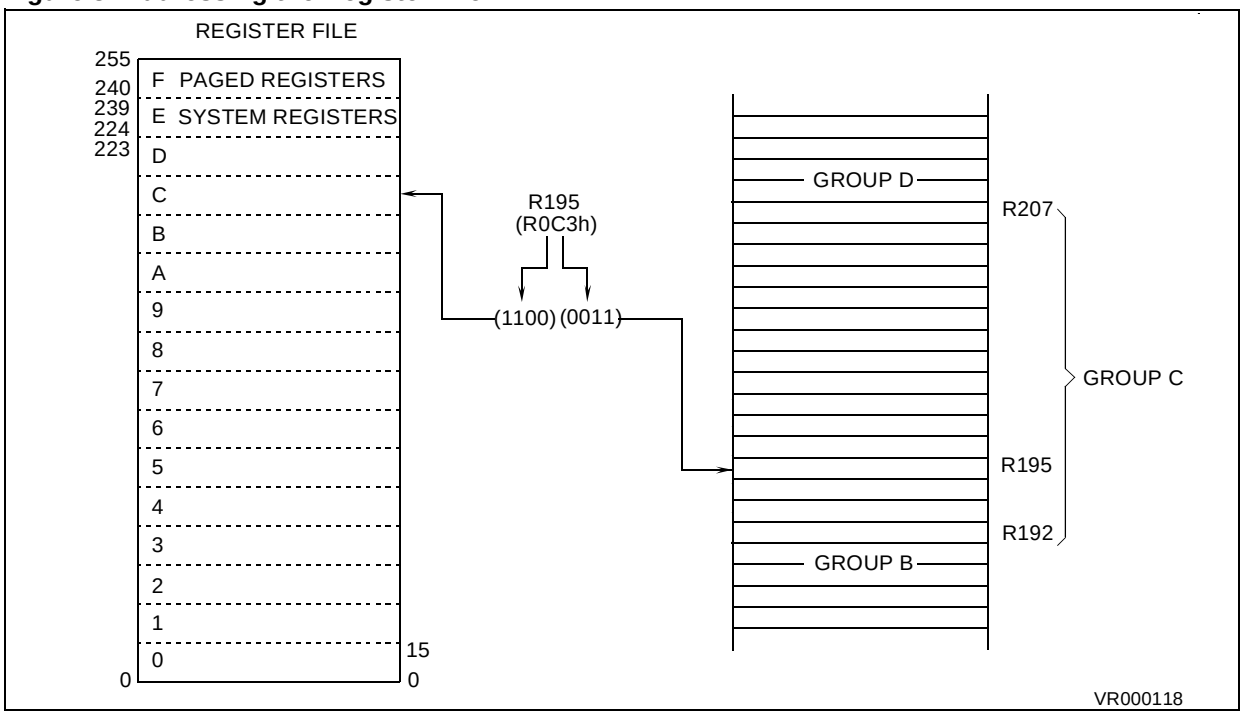


Figure 8. Addressing the Register File



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 8). 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 2.3.3, and illustrated in Figure 9 and in Figure 10.

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 2.3.

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 8. 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 9. 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 9. 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 9. Pointing to a single group of 16 registers

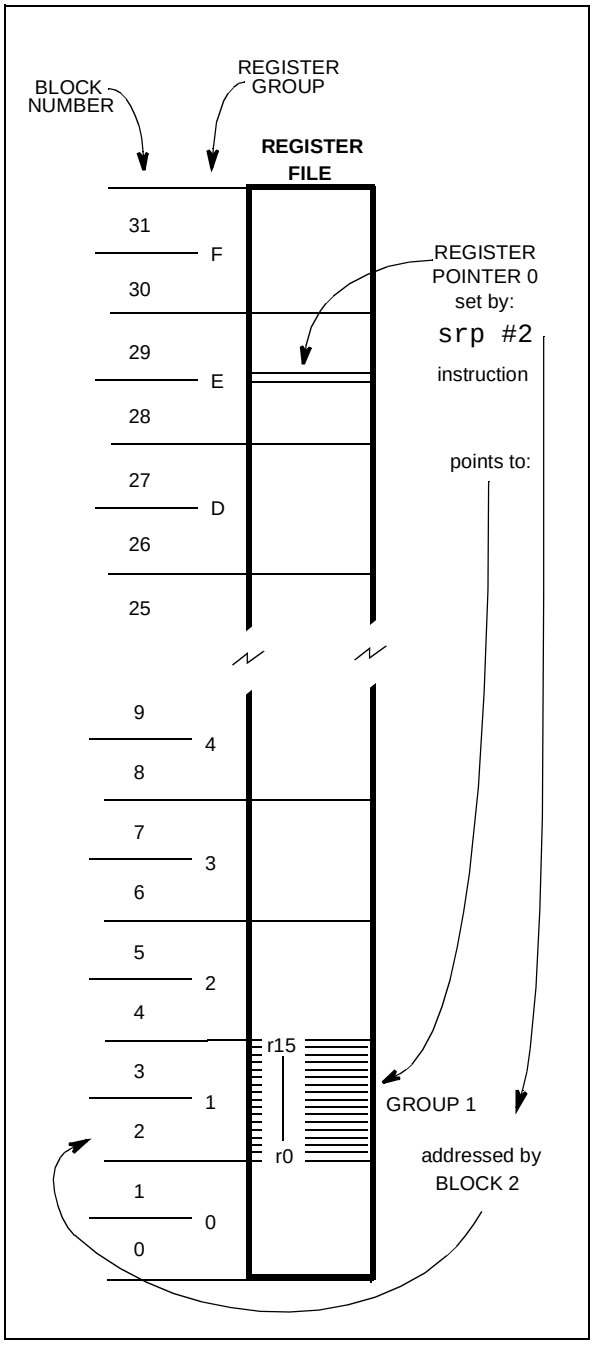
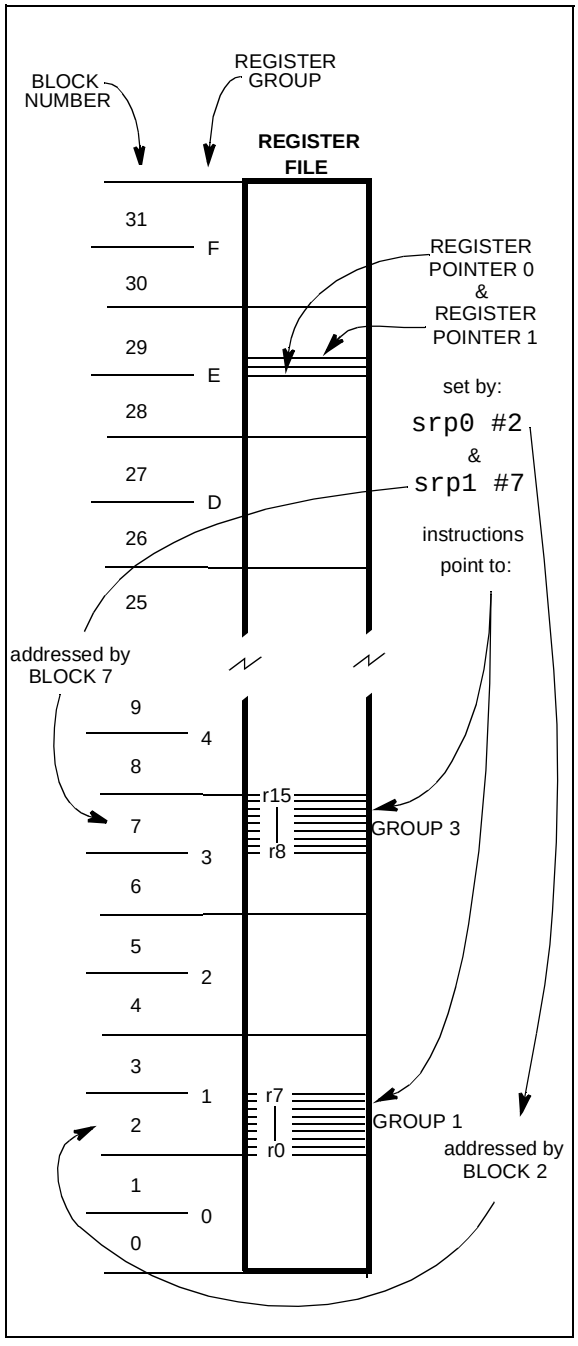


Figure 10. Pointing to two groups of 8 registers



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 any of Ports 0, 1, 2 or 6 depending on device configuration, are 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

SYSTEM REGISTERS (Cont'd)

state by setting the HIMP bit. When this bit is reset, it has no effect.

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

If Port 1 and/or 2 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 9.

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 11. Internal Stack Mode

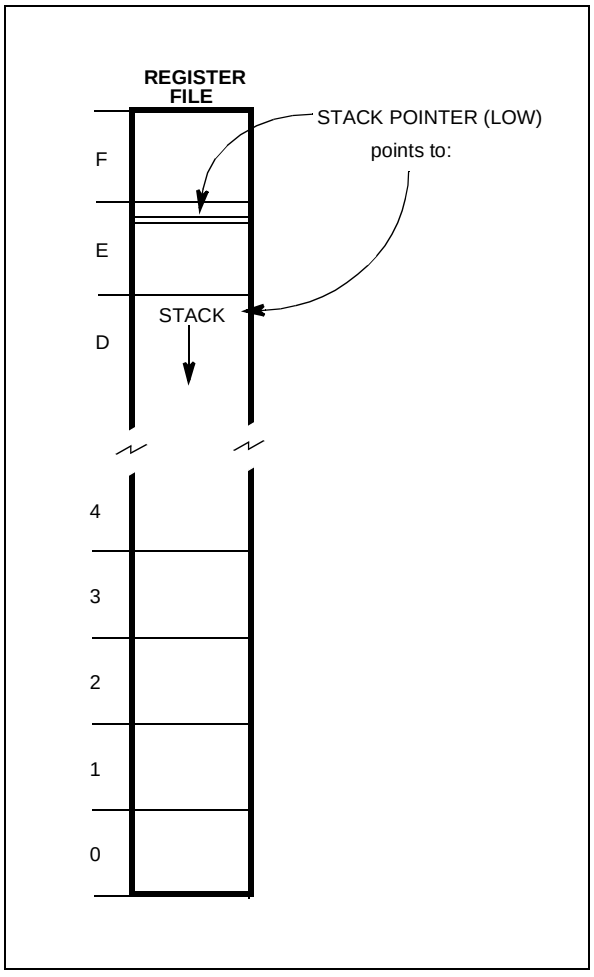
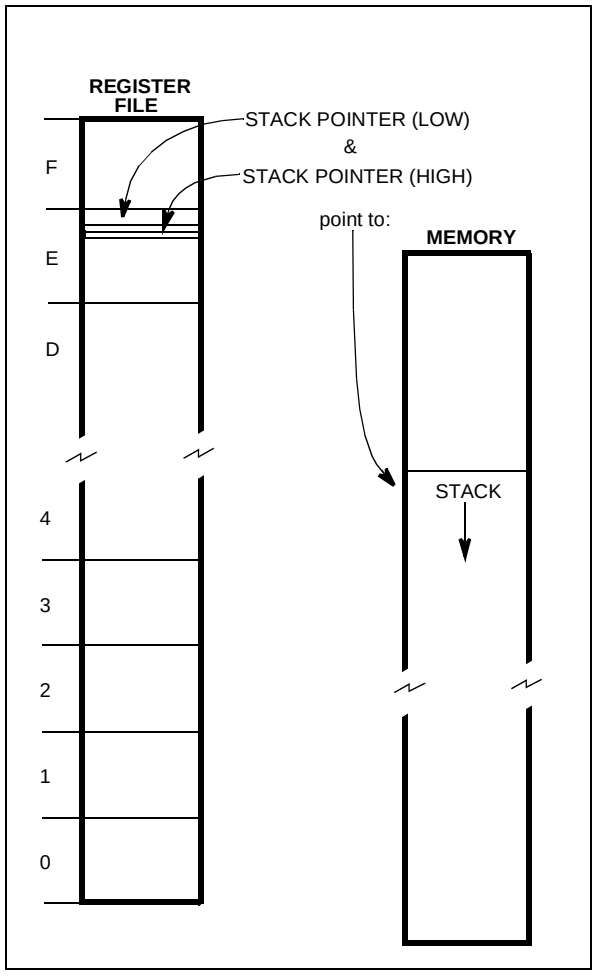


Figure 12. External Stack Mode



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).

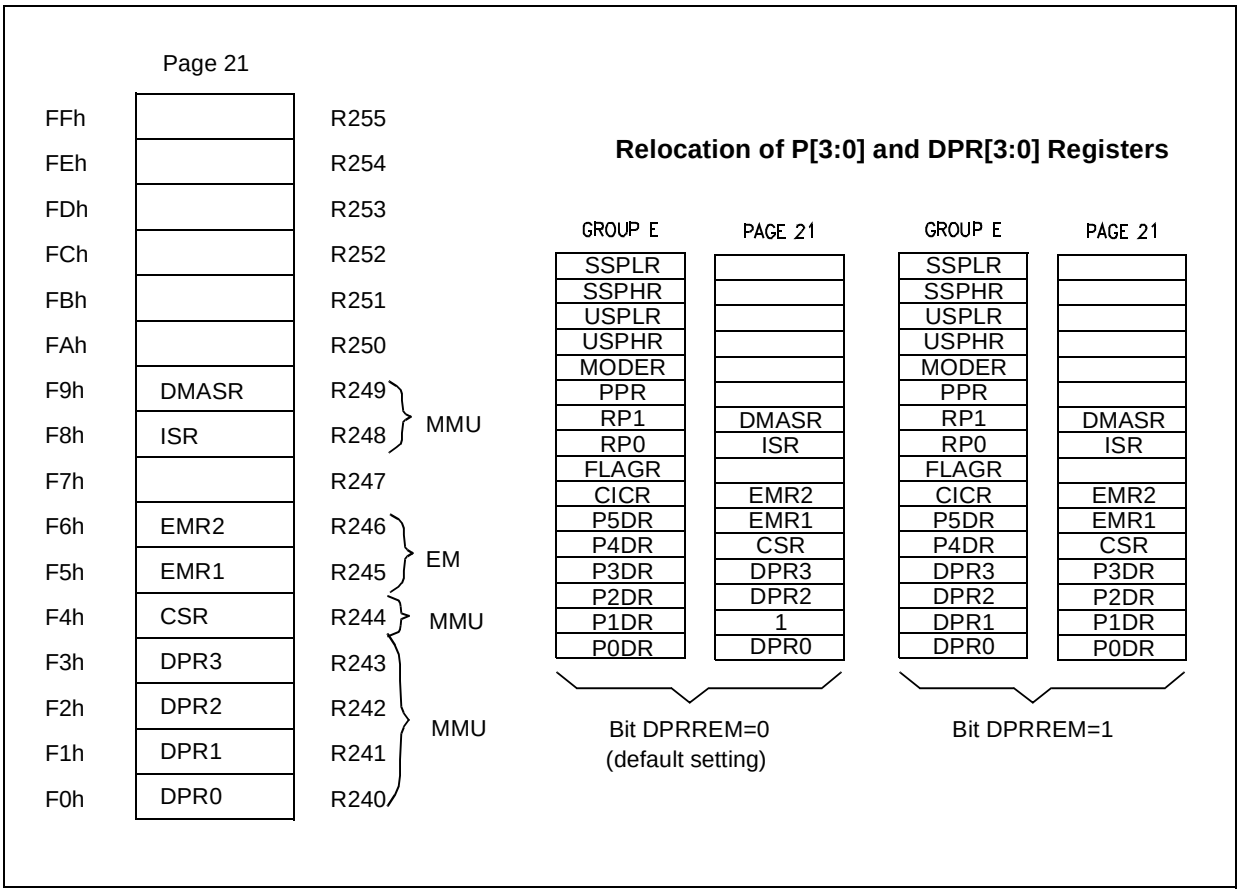
Refer to the Register and Memory Map Chapter for more details on the memory map.

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 13. 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;

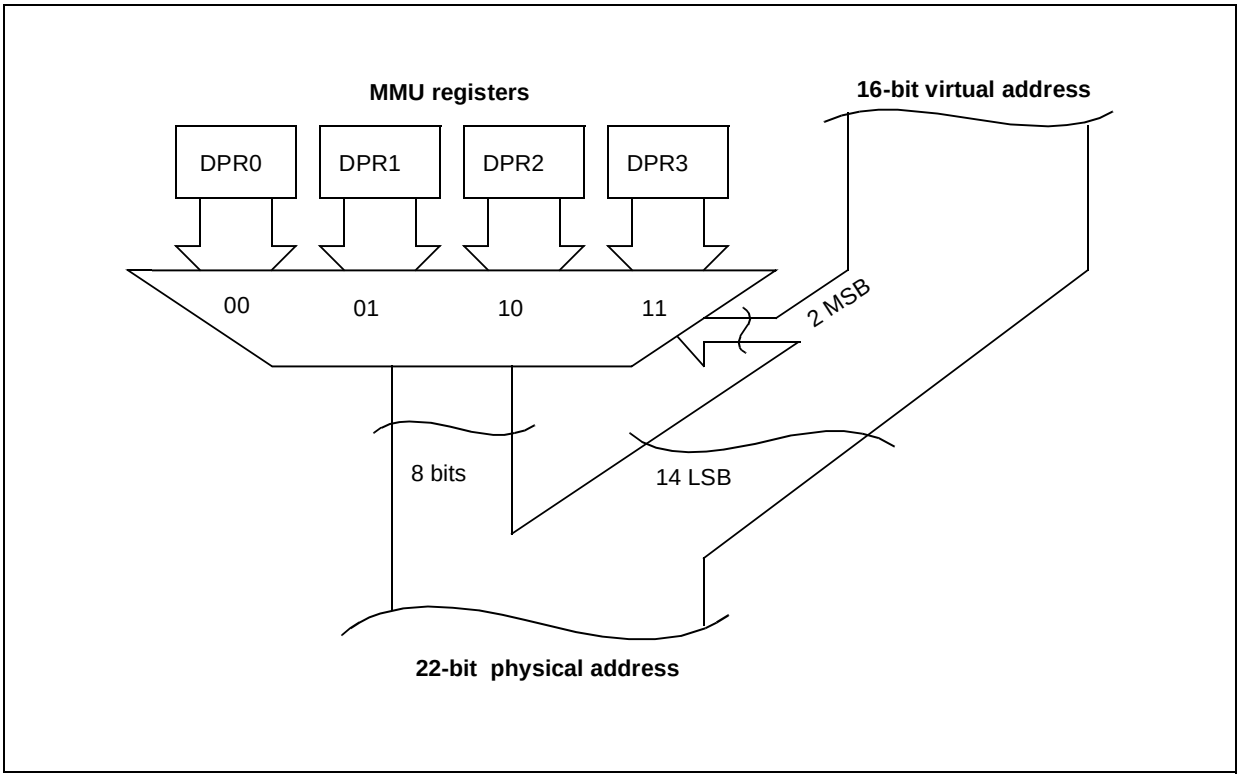
DPR2: from 8000h to BFFFh;

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 14).

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 14. 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 15).

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

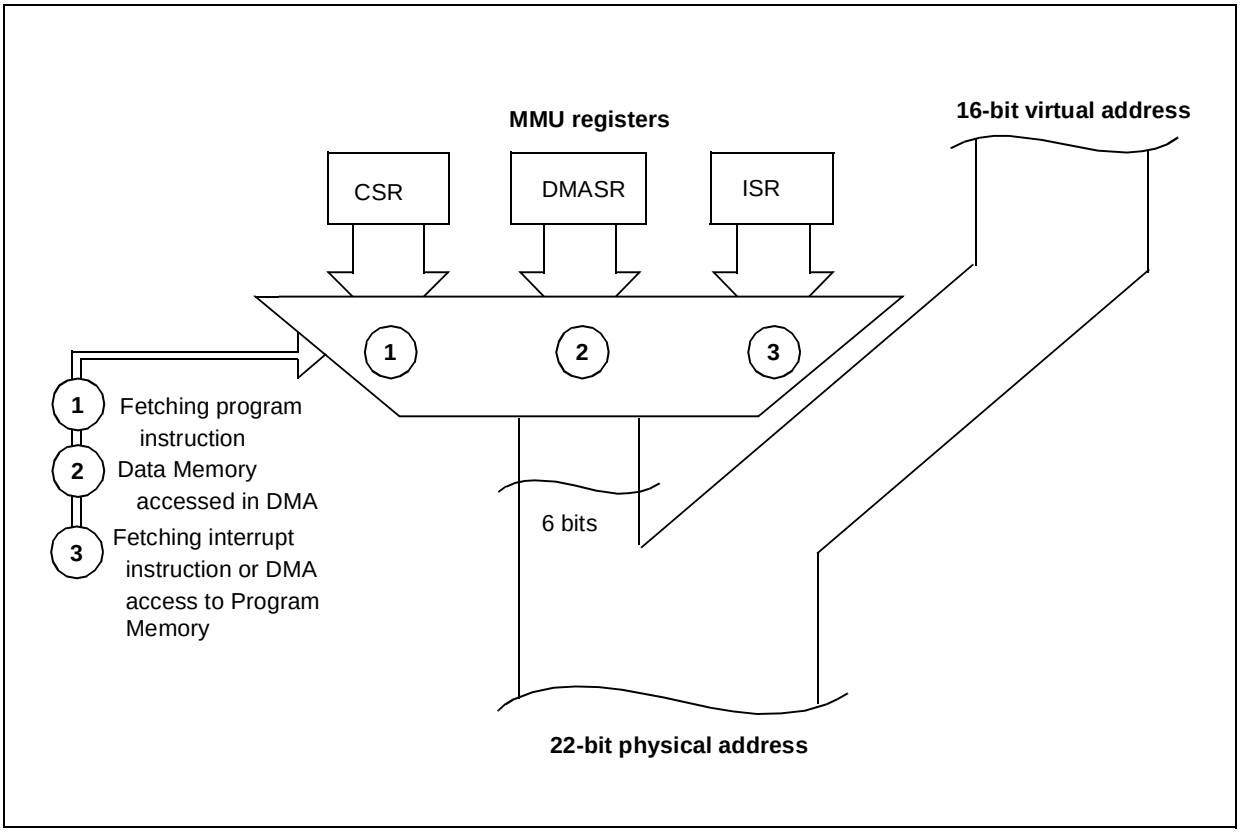
The DPR[3:0] registers allow access to the entire 4 Mbyte memory space composed of 256 pages of 16 Kbytes.

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 13.

Figure 15. 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

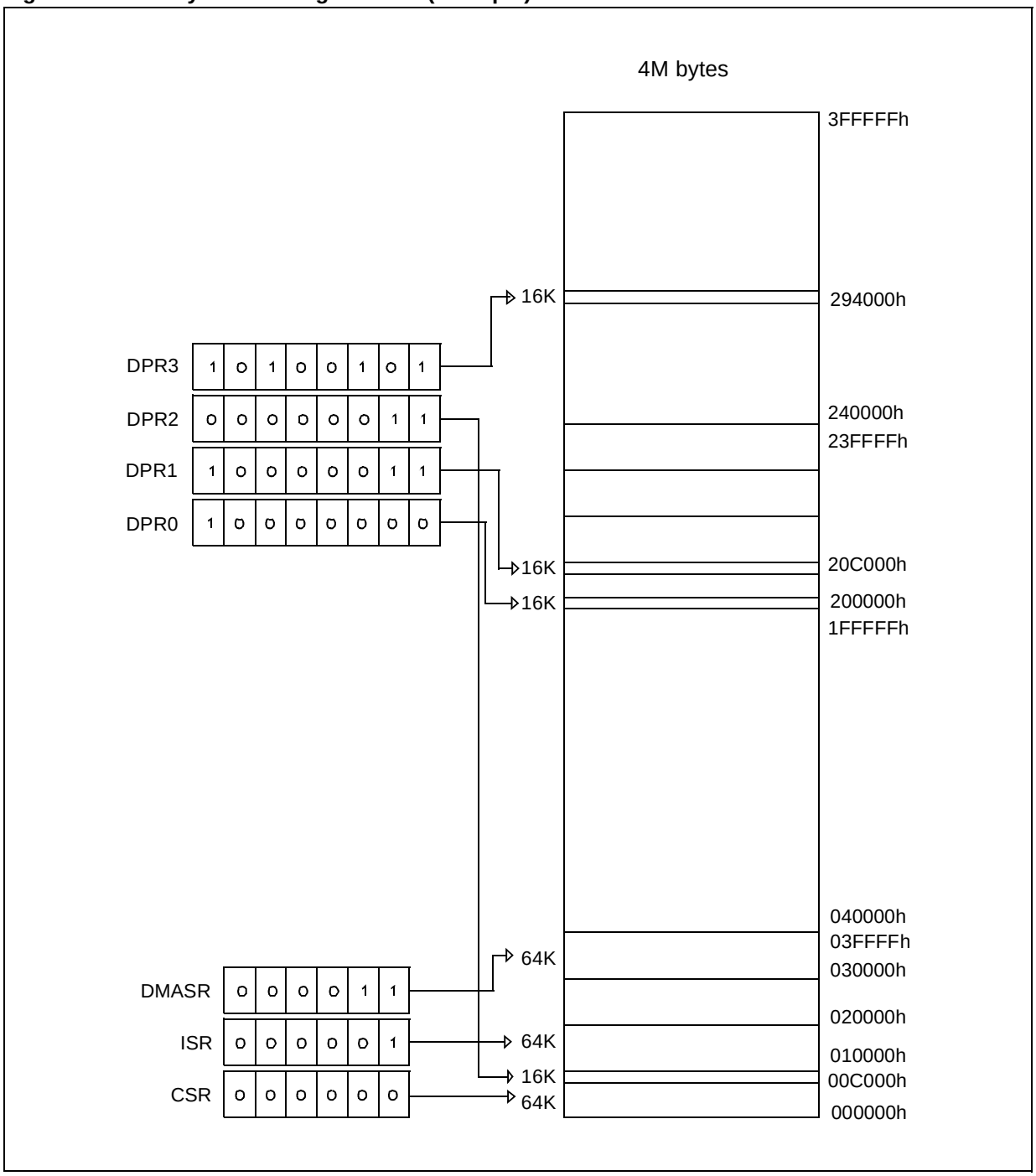
7							0
0	0	DMA SR_5	DMA SR_4	DMA SR_3	DMA SR_2	DMA SR_1	DMA SR_0

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 16. 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 INTERRUPTS

3.1 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

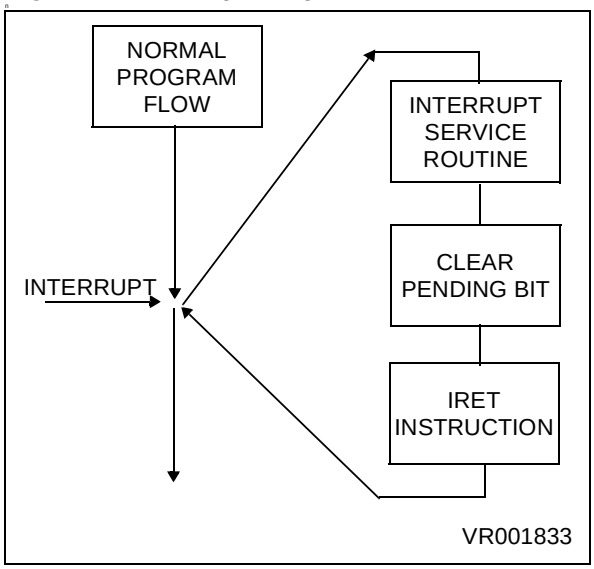
– External pins

– 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 17. Interrupt Response



3.2 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 is stored in the first two physical bytes in memory, 000000h and 000001h.

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

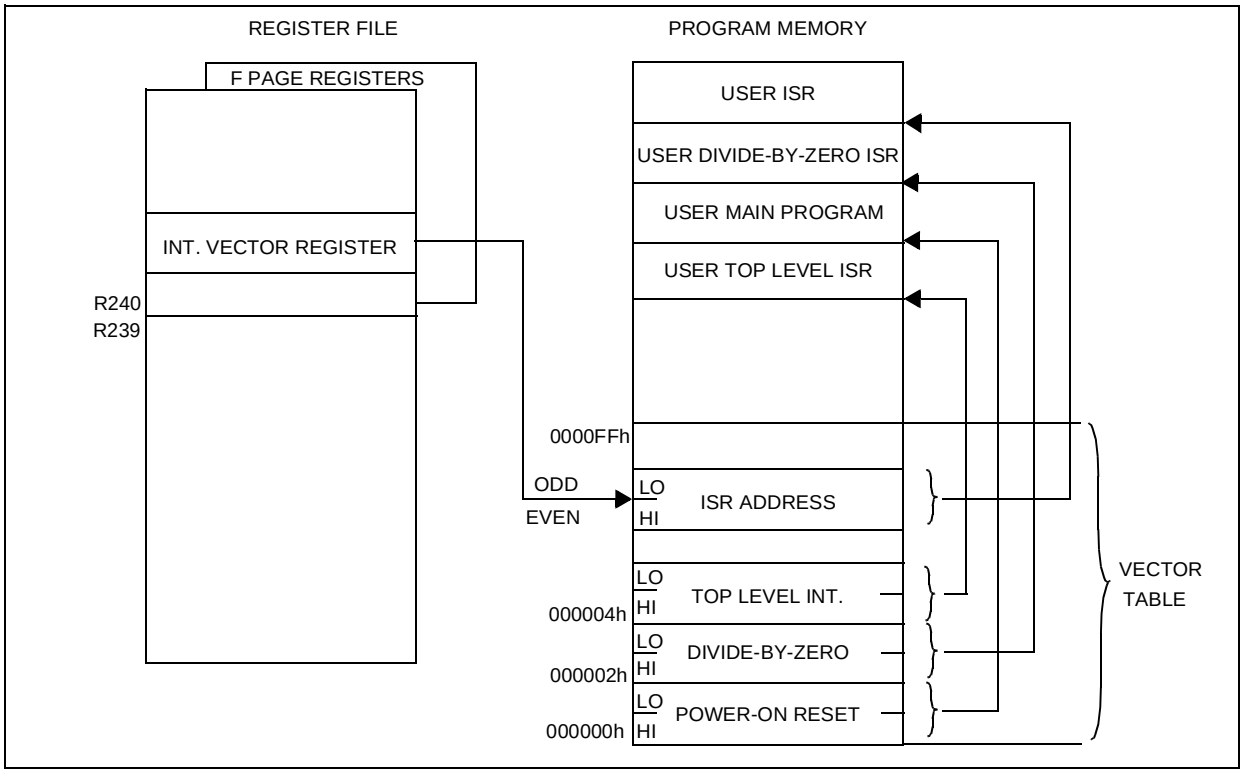
With one Interrupt Vector register, it is possible to address several interrupt service routines; in fact, peripherals can share the same interrupt vector register among several interrupt channels. The most significant bits of the vector are user programmable to define the base vector address within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

3.2.1 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.

Warning. *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` *).*



3.2.2 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

3.3 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).

3.4 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.

3.4.1 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.

3.4.2 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.

3.4.3 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 Figure 10

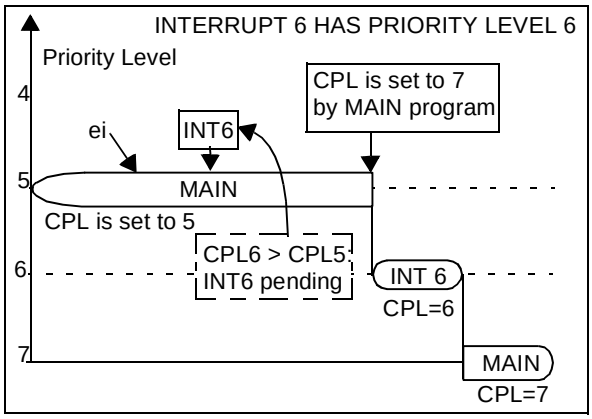
Table 10. Daisy Chain Priority

Highest Position	INTA0	INT0/WDT
	INTA1	INT1/STIM
	INTB0	INT2/SPI
	INTB1	INT3/I2C
	INTC0	INT4/OSD
	INTC1	INT5/ADC
	INTD0	INT6/DS0, DS1
	INTD1	INT7/IR
	SCI	
Lowest Position	MFT	

3.4.4 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 18

Figure 18. Example of Dynamic priority level modification in Nested Mode



3.5 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.

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

ARBITRATION MODES (Cont'd)

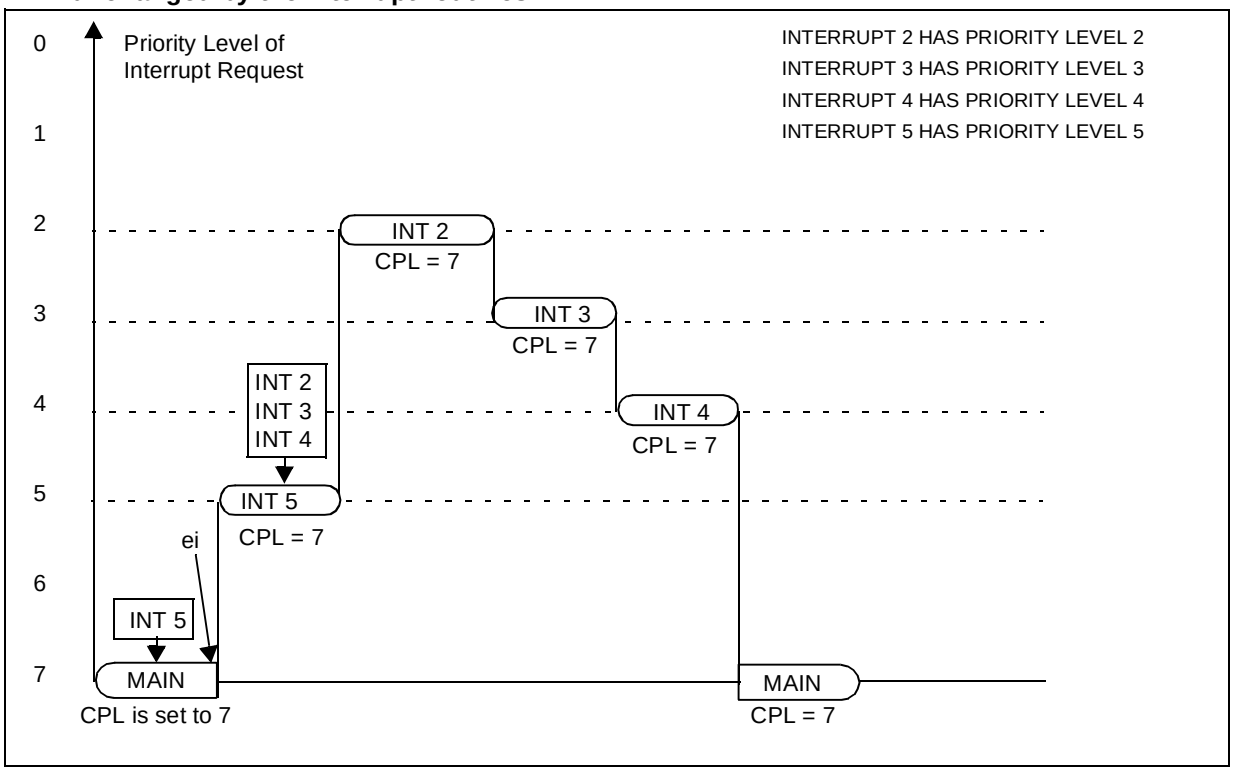
Examples

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 19) 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 19. 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

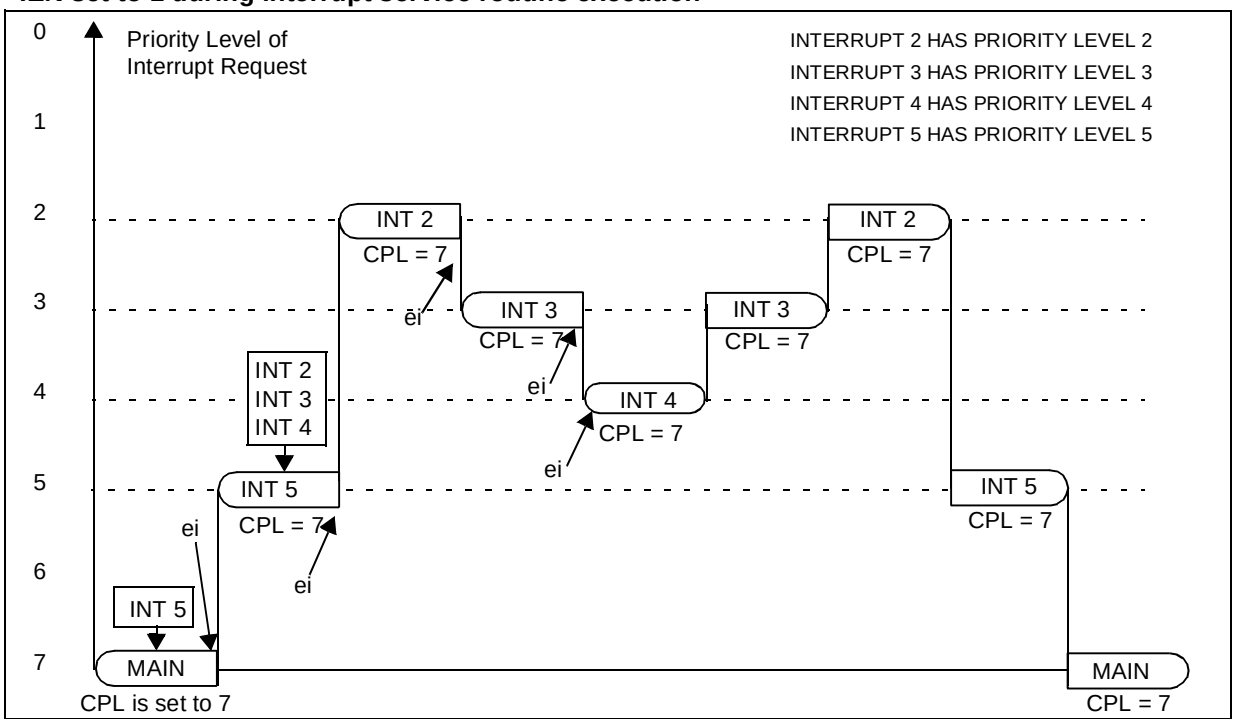
In the second example, (more complex, Figure 20), each interrupt service routine sets Interrupt Enable with the `ei` instruction at the beginning of the routine. Placed here, it minimizes response time for requests with a higher priority than the one being serviced.

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.

WARNING: 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 20. 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)

3.5.2 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.

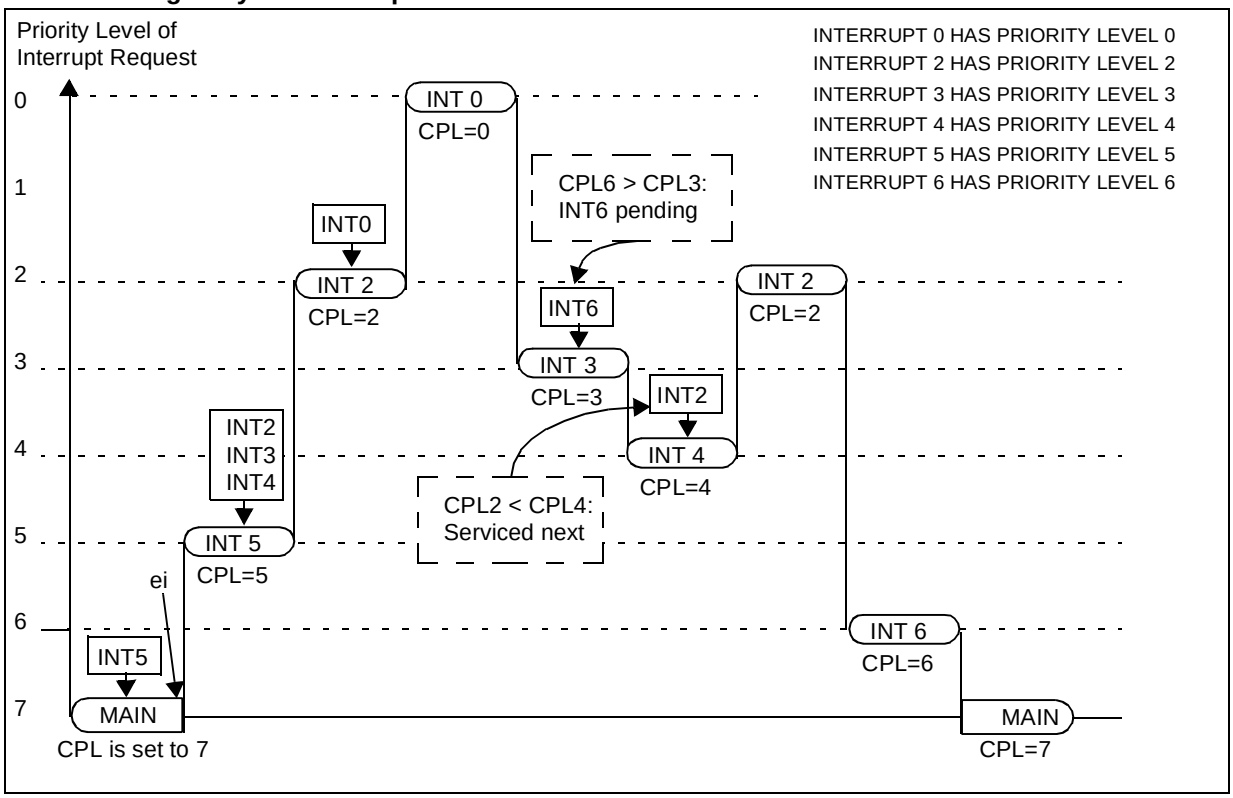
– If ENCSR is set, CSR is pushed onto system stack.

– 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 21. 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.

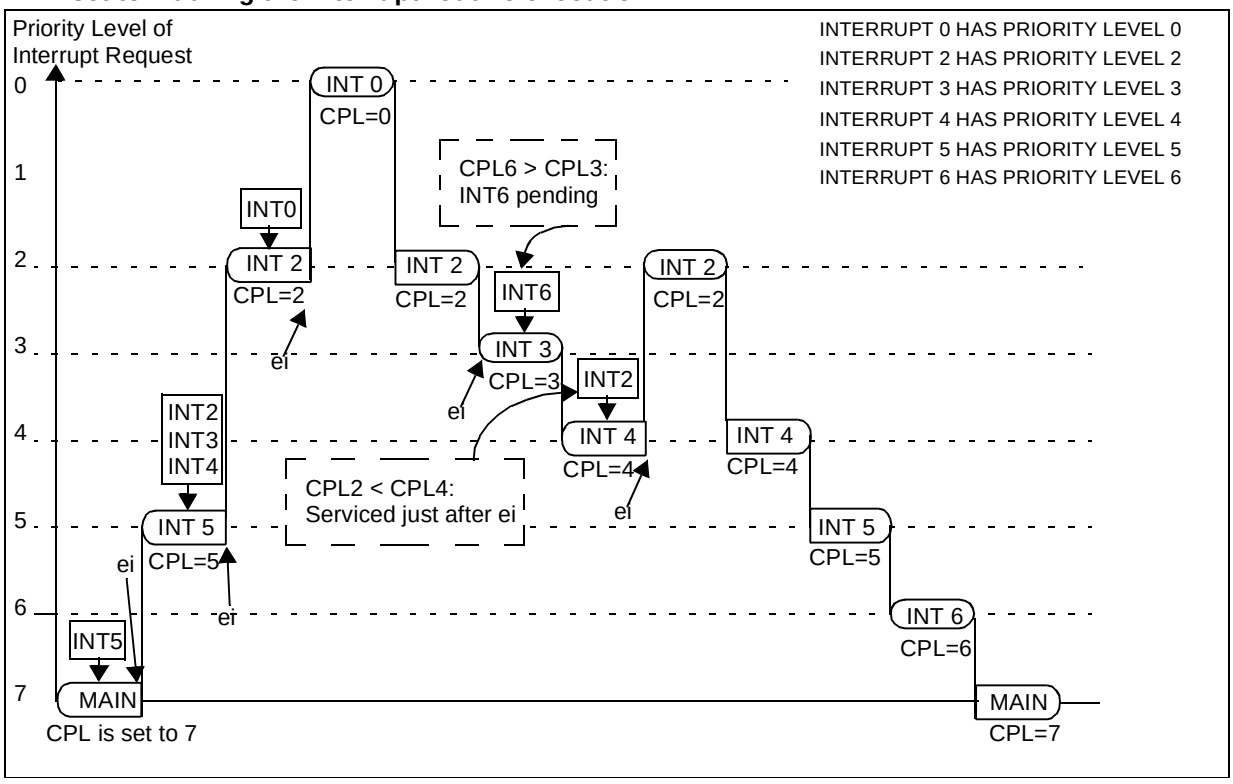
– If ENCSR is reset, CSR is used instead of ISR, unless the program returns to another nested routine.

The suspended routine thus resumes at the interrupted instruction.

Figure 21 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 22 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 22. Complex Example of a Sequence of Interrupt Requests with:
- Nested mode
- IEN set to 1 during the interrupt routine execution



3.6 EXTERNAL INTERRUPTS

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

Table 11. External Interrupt Channel Grouping

External Interrupt	Channel
INT7 INT6	INTD1 INTD0
INT5 INT4	INTC1 INTC0
INT3 INT2	INTB1 INTB0
INT1 INT0	INTA1 INTA0

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 1.

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 23. Priority Level Examples

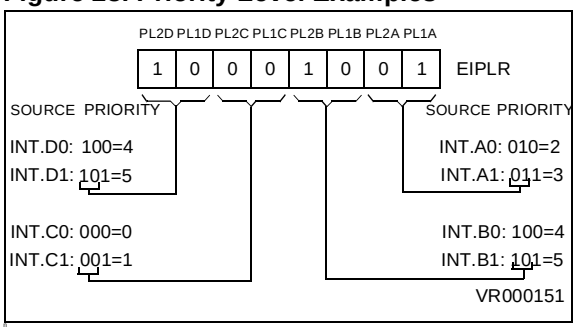


Figure 23 shows an example of priority levels.

Figure 1 gives an overview of the External interrupt control bits and vectors.

– The source of the interrupt channel A1 can be selected between the external pin INT4 (when INTS = 1) or the on-chip Standard Timer.

– The source of the interrupt channel B0 can be selected between the external pin INT2 (when (SPEN,BMS)=(0,0)) or the on-chip SPI peripheral.

– The source of the interrupt channel INTB1 can be selected between the INT3 external pin (CLEAR=1) or the I2C interrupt (CLEAR=0) by programming the CLEAR bit in the I2CCTR register.

– The source of the interrupt channel INTC0 can be selected between the INT4 external pin (DION=OSDE=0) or the Display Controller interrupt (all other cases) by programming the DION, OSDE bits in the OSDER register.

– The source of the interrupt channel C1 can be selected between the external pin INT5 (when the AD_INT bit in the AD-INT register=0) or the on-chip ADC (when AD-INT=1).

– The source of the interrupt channel D0 can be selected between the external pin INT6 (when the CCID bit in the DS0CR2 or DS1CR2 register=1) or the on-chip Data Slicers (when CCID=0).

– The source of the interrupt channel D1 can be selected between the external pin INT7 (when the IRWDIS bit in the IRSC register = 1) or the on-chip IR (when IRWDIS=0).

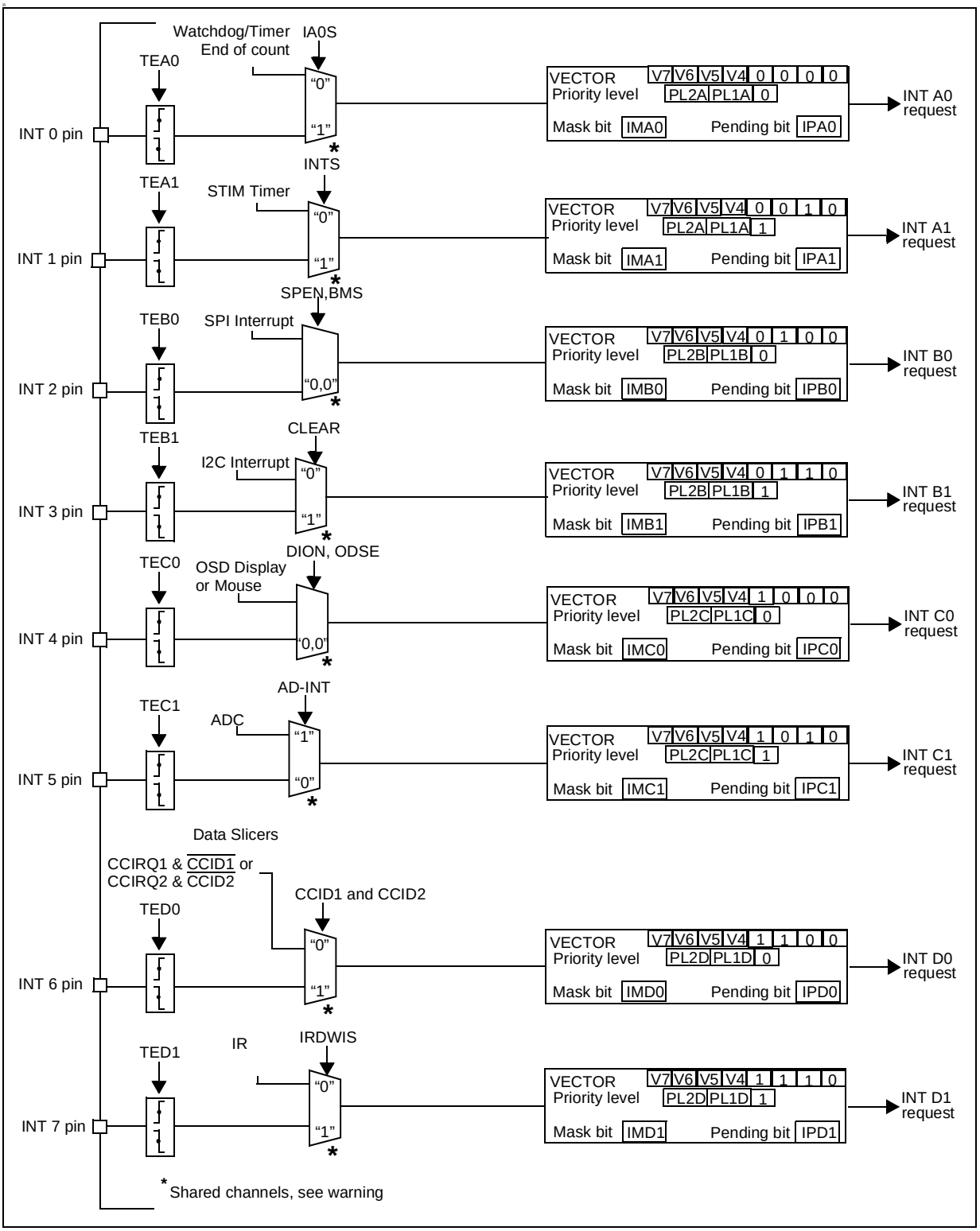
Warning: When using channels shared by both external interrupts and peripherals, special care must be taken to configure their control registers for both peripherals and interrupts.

Table 12. Multiplexed Interrupt Sources

Channel	Internal Interrupt Source	External Interrupt Source	Related Pin
INTA0	Timer/Watch-dog	INT0	P3.2
INTA1	STIM Timer	INT1	P3.4
INTB0	SPI Interrupt	INT2	P2.4
INTB1	I2C	INT3	P2.2
INTC0	OSD	INT4	P2.3
INTC1	ADC	INT5	P2.7
INTD0	DS0 & DS1	INT6	P2.1
INTD1	IR	INT7	P2.0

EXTERNAL INTERRUPTS (Cont'd)

Figure 1. External Interrupts Control Bits and Vectors



3.7 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.

Warning. 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.

3.8 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:

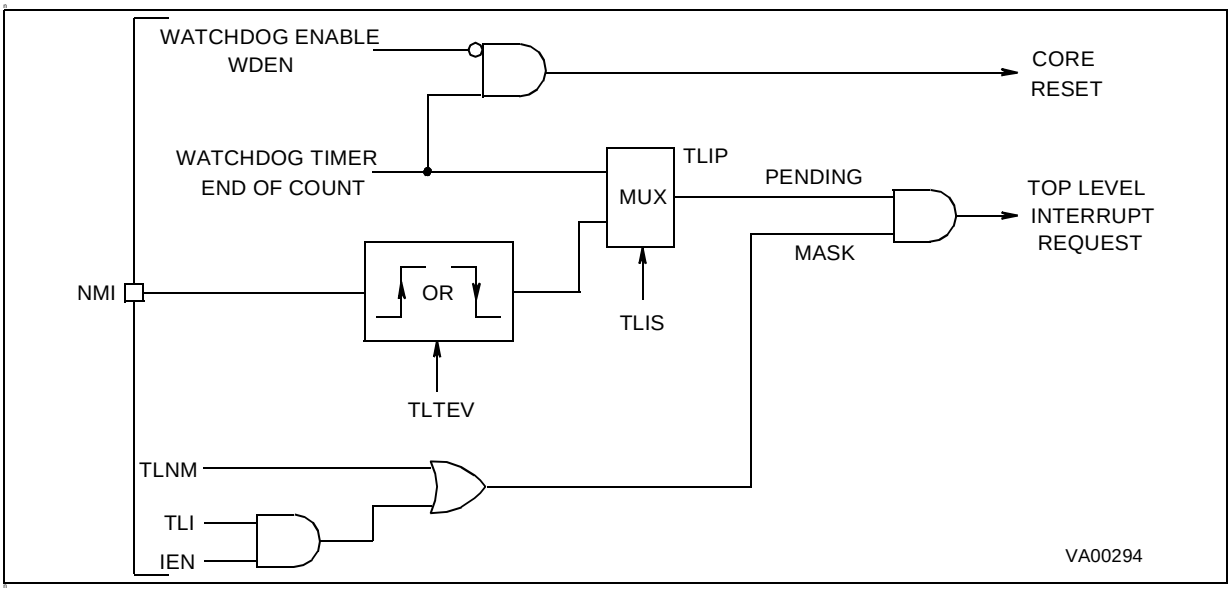
– **Interrupt Pending bit** (IP). Set by hardware when the Trigger Event occurs. Can be set/ cleared by software to generate/cancel pending interrupts and give the status for Interrupt polling.

– **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 24. Top Level Interrupt Structure



3.9 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.

3.10 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

Bit 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.

INTERRUPT REGISTERS (Cont'd)

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

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)

INTERRUPT REGISTERS (Cont'd)

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

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

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

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

Bit 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)

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

Bit 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 1.

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
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

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

Bit 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.

INTERRUPT REGISTERS (Cont'd)

EXTERNAL MEMORY REGISTER 2 (EMR2)

R246 - Read/Write
Register Page: 21
Reset value: 0000 1111 (0Fh)

7							0
0	ENCSR	0	0	1	1	1	1

Bit 7, 5:0 = Reserved, keep in reset state. Refer to the external Memory Interface Chapter.

Bit 6 = **ENCSR**: *Enable Code Segment Register.* This bit is set and cleared by software. It affects the ST9 CPU behaviour whenever an interrupt request is issued.

0: 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. 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: 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; this difference, however, should not affect the vast majority of programs.

4 ON-CHIP DIRECT MEMORY ACCESS (DMA)

4.1 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.

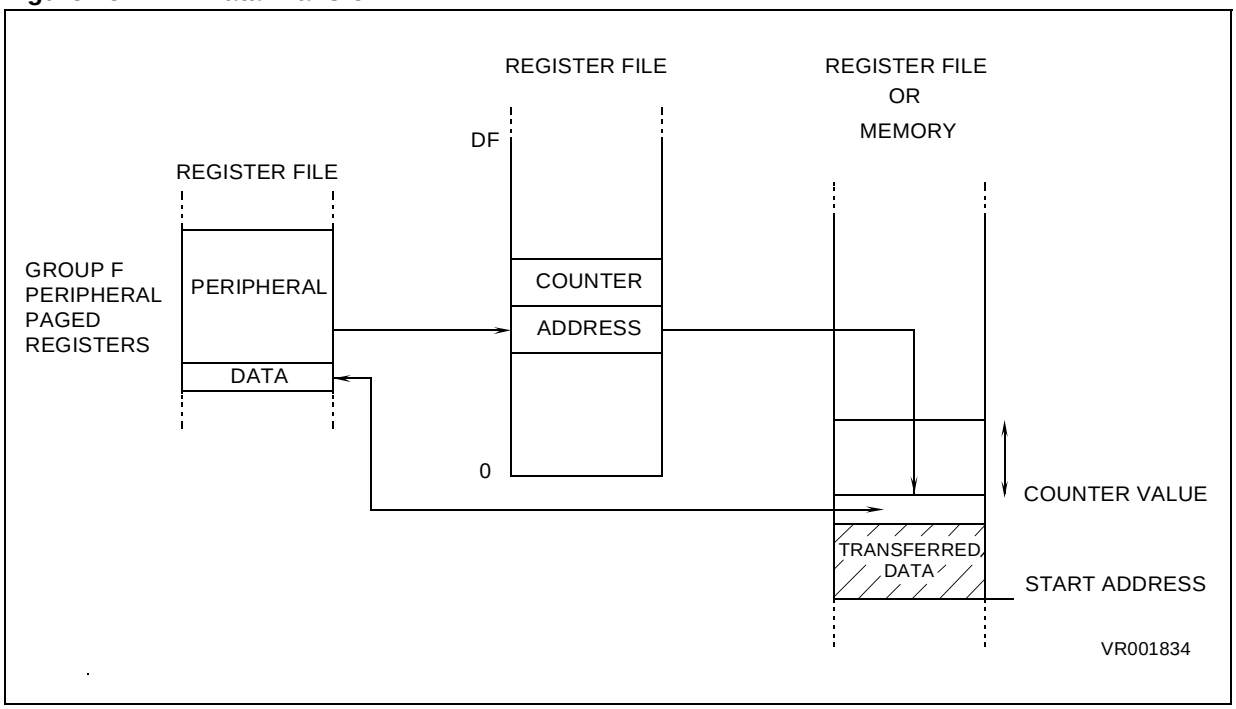
4.2 DMA PRIORITY LEVELS

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 25. DMA Data Transfer



4.3 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)

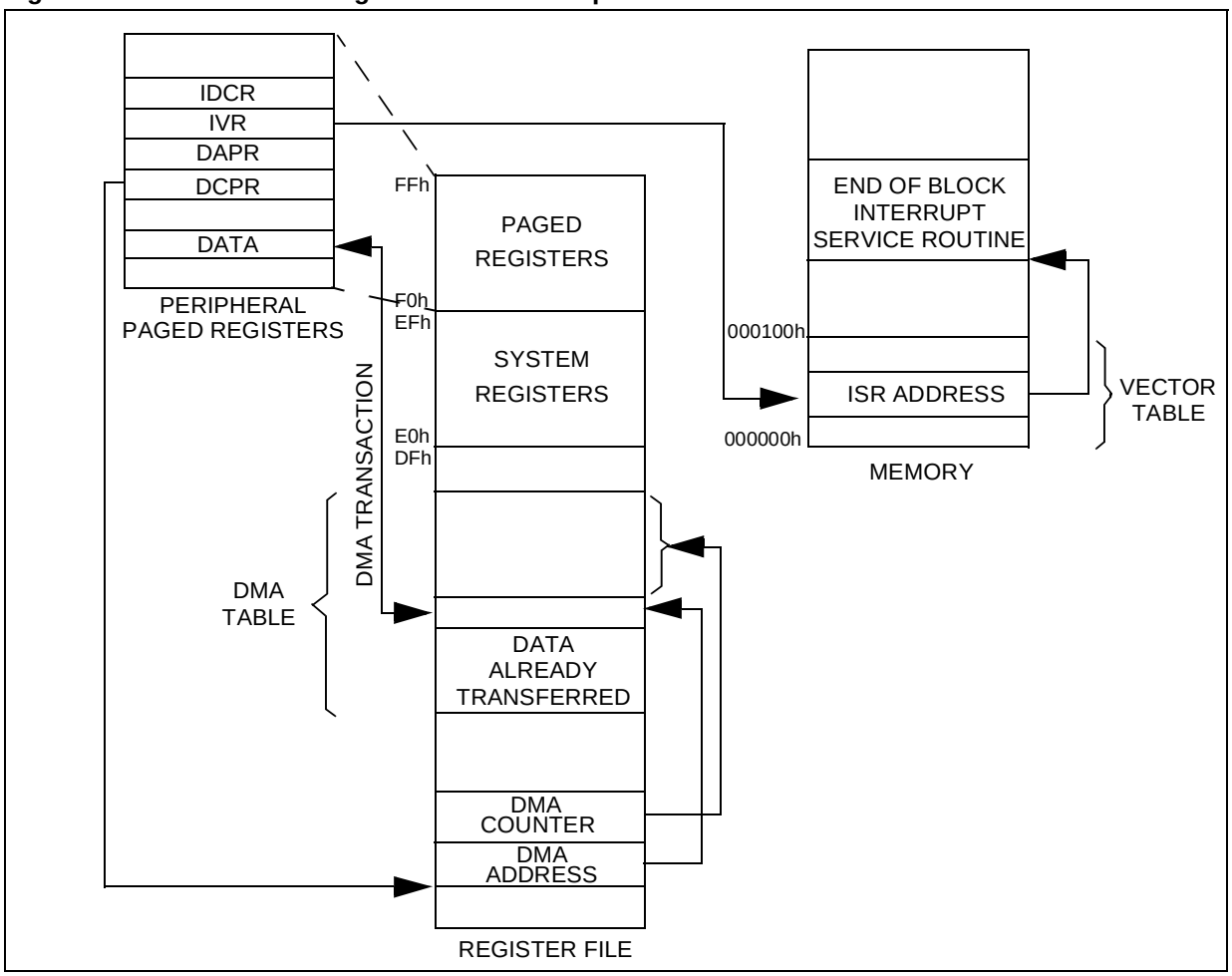
– A post-decrement of the DMA transaction counter, which contains the number of transactions that have still to be performed.

If the DMA transaction is carried out between **the peripheral and the Register File** (Figure 26), 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 27). 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 26. 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.

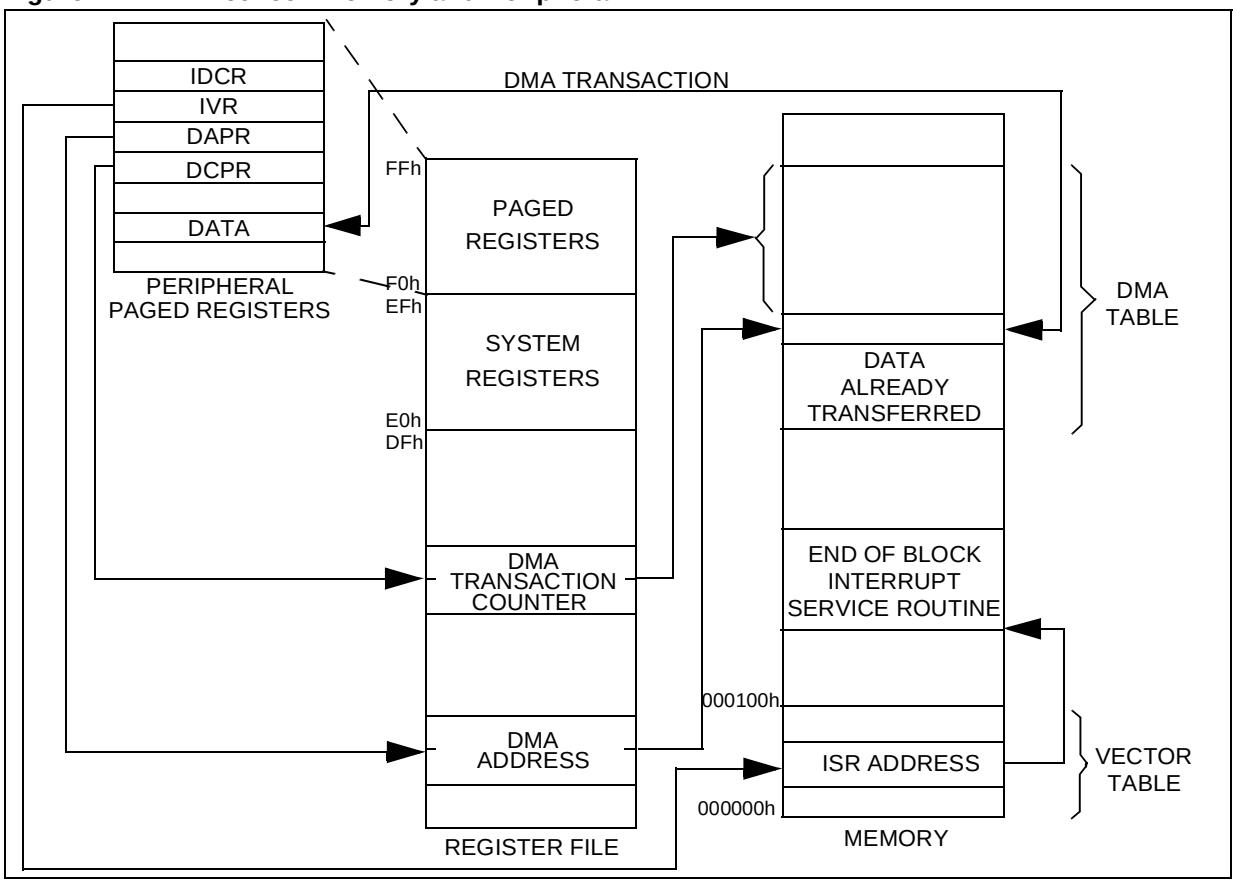
Once a DMA channel is initialized, a transfer can start. The direction of the transfer is automatically defined by the type of peripheral and programming mode.

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 27. DMA Between Memory and Peripheral



DMA TRANSACTIONS (Cont'd)

4.4 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.

4.5 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.

4.6 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 4.2 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).

5 RESET AND CLOCK CONTROL UNIT (RCCU)

5.1 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.

5.2 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

Bit 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
-	-	-	-	SRE-SEN	-	-	-

Bit 7:4 = **Reserved.**
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 2:0 = **Reserved.**
Must be kept reset for normal operation.

CLOCK FLAG REGISTER (CLK_FLAG)
R242 - Read/Write
Register Page: 55
Reset Value: 0100 1000 after a Watchdog Reset
Reset Value: 0010 1000 after a Software Reset
Reset Value: 0000 1000 after a Power-On Reset

7							0
-	WDG RES	SOFT RES	-	-	-	-	-

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

Bit 7 = **Reserved.**
Must be kept reset for normal operation.

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 4:0 = **Reserved.**
Must be kept reset for normal operation.

5.3 OSCILLATOR CHARACTERISTICS

Because of the real time need of the application, it is assumed the ST92196A will be used with a 4 MHz crystal fed to the Core by the frequency multiplier output after it is started and stabilized.

5.3.1 HALT State

When a HALT instruction is processed, it stops the main crystal oscillator preventing any derived clock into the chip. Exit from the HALT state can be obtained through a main system reset.

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 28. Crystal Oscillator

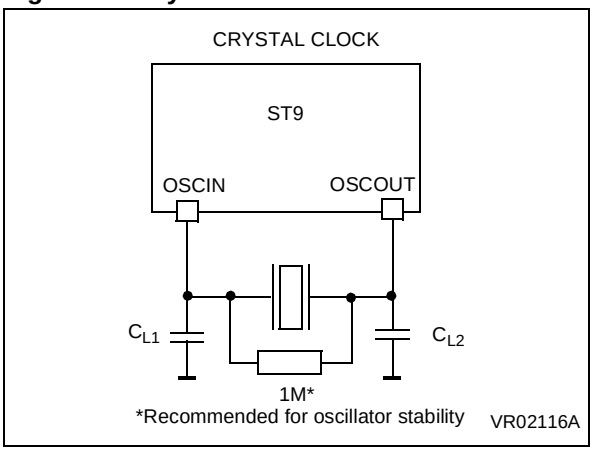


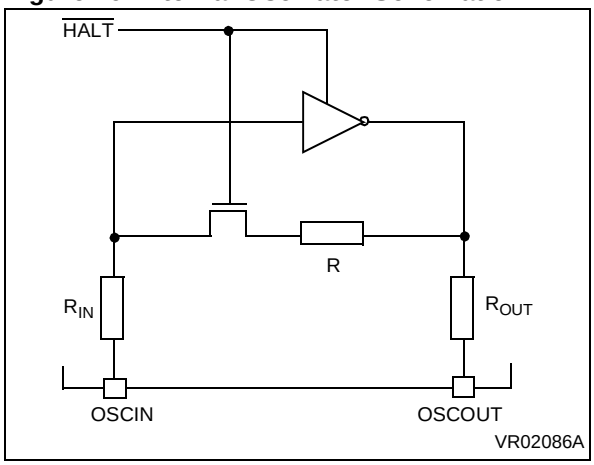
Table 13. Crystal Specification

	C_1=C_2=56pF	C_1=C_2=47pF
Rs max (ohm)	200	260

Legend:
C_{L1}, C_{L2}: Maximum Total Capacitances on pins OSCIN and OSCOUT (the value includes the external capacitance tied to the pin CL1 and CL2 plus the parasitic capacitance of the board and of the device).

Note: The table is relative to the fundamental quartz crystal only (not ceramic resonator).

Figure 29. Internal Oscillator Schematic



5.4 RESET/STOP MANAGER

The RESET/STOP Manager resets the device when one of the three following triggering events occurs:

– A hardware reset, consequence of a falling edge on the **RESET** pin.

– A software reset, consequence of an HALT instruction when enabled.

– A Watchdog end of count.

The **RESET** input is schmitt triggered.

Note: The memorized Internal Reset (called RESETI) will be maintained active for a duration of 32768 Oscin periods (about 8 ms for a 4 MHz crystal) after the external input is released (set high).

This RESETI internal Reset signal is output on the I/O port bit P5.0 (active low) during the whole reset phase until the P5.0 configuration is changed by software. The true internal reset (to all macrocells) will only be released 511 Reference clock periods after the Memorized Internal reset is released.

It is possible to know which was the last **RESET** triggering event, by reading bits 5 and 6 of register CLK_FLAG.

Figure 30. Reset Overview

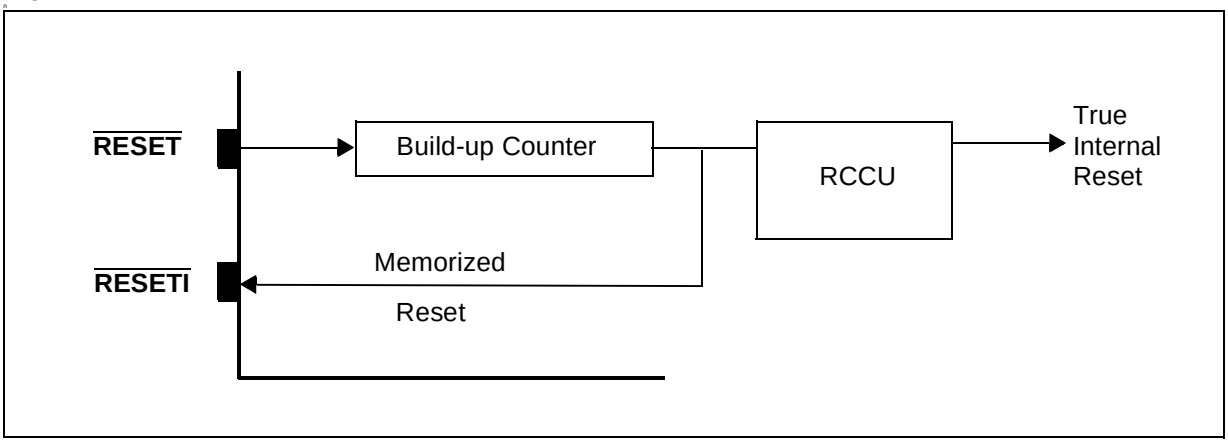
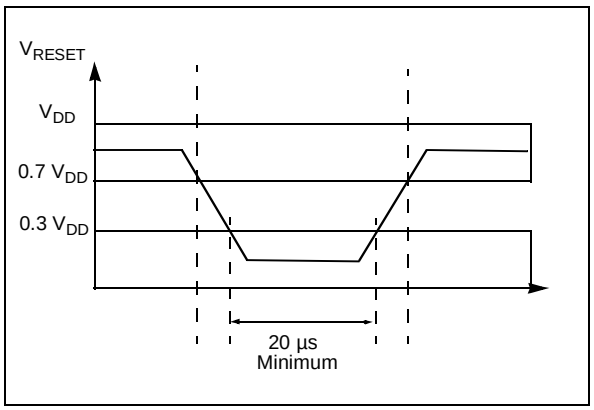


Figure 31. Recommended Signal to be Applied on Reset Pin



6 TIMING AND CLOCK CONTROLLER (TCC)

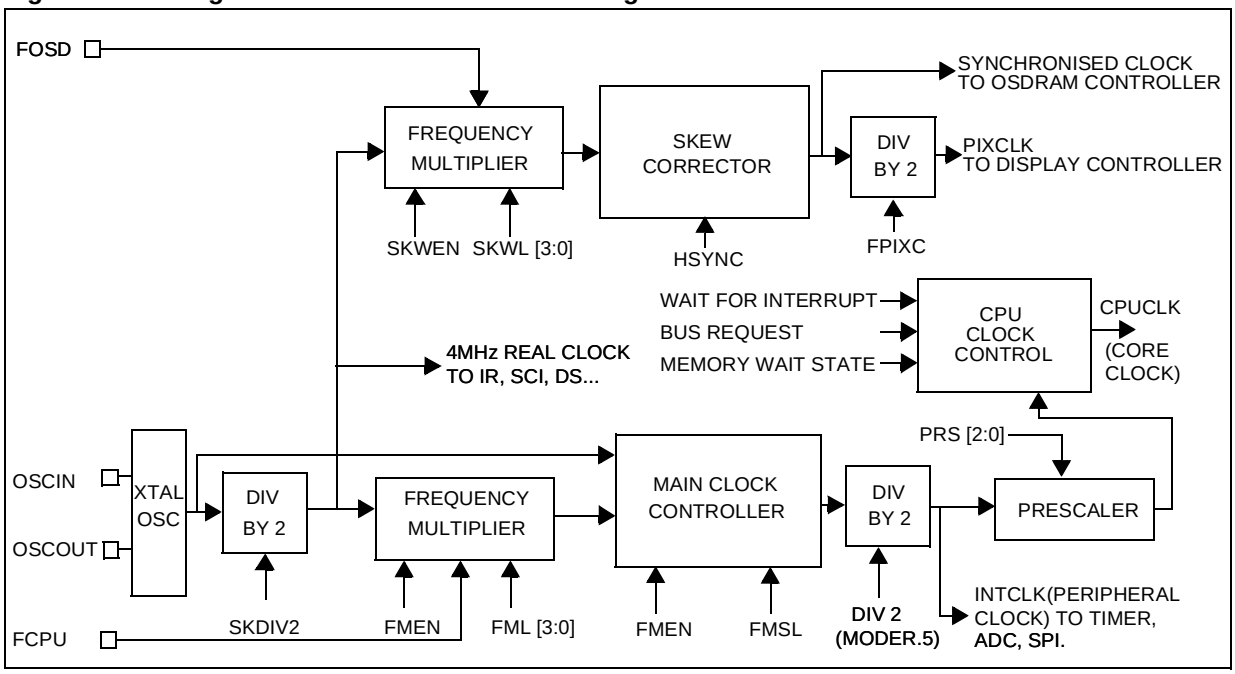
6.1 FREQUENCY MULTIPLIERS

Two on-chip frequency multipliers generate the proper frequencies for: the Core/Real time Peripherals and the Display related time base.

They follow the same basic scheme based on an integrated VCO driven by a three state phase comparator and a charge-pump (1 pin used for off-chip filtering components; a resistor in series with a capacitor tied to ground).

For both the Core and the Display frequency multipliers, a 4 bit programmable feed-back counter allows the adjustment of the multiplying factor to the application needs (a 4 MHz crystal is assumed).

Figure 32. Timing and Clock Controller Block Diagram



FREQUENCY MULTIPLIERS (Cont'd)

Off-chip filter components (to be confirmed)

– Core frequency multiplier (FCPU pin) : 1.2K ohms; 47 nF plus 100 pF between the FCPU pin and GND.

– Skew frequency multiplier (FOSD pin) : 1.2K ohms; 47 nF plus 100 pF between the FOSD pin and GND.

The frequency multipliers are off during and upon exiting from the reset phase. The user must program the desired multiplying factor, start the multiplier and then wait for its stability (refer to the Electrical Characteristics chapter for the specified delay).

Once the Core/Peripherals multiplier is stabilized, the Main Clock controller can be re-programmed through the FMSL bit in the MCCR register to provide the final frequency (CPUCLK) to the CPU.

The frequency multipliers are automatically switched off when the microprocessor enters HALT mode (the HALT mode forces the control register to its reset status).

Table 14. Examples of CPU Speed Choices

Crystal Frequency SKDIV2=0	FML (3:0)	CPUCLK
4 MHz	4	10 MHz
	5	12 MHz
	6	14 MHz
	7	16 MHz
	8	18 MHz
	9	20 MHz
	10	22 MHz
	11	24 MHz

Note: 24 MHz is the max. authorized frequency.
Caution: The values indicated in this table are the only authorized values.

Table 15. PIXCLK Frequency Choices

Crystal Frequency	SKW (3:0)	FPIXC	PIXCLK
SKDIV2=0			
4 MHz	6	0	14 MHz
	7	0	16 MHz
	8	0	18 MHz
	9	0	20 MHz
	10	0	22 MHz
	11	0	24 MHz
	6	1	28 MHz
	7	1	32 MHz
	8	1	36 MHz
	9	1	40 MHz

6.2 REGISTER DESCRIPTION

SKEW CLOCK CONTROL REGISTER (SKCCR)

R254 - Read/ Write
Register Page: 43
Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
SKWEN	SKDIV2	0	0	SKW3	SKW2	SKW1	SKW0

The HALT mode forces the register to its initialization state.

Bit 7= **SKWEN:** *Frequency Multiplier Enable bit.*
0: FM disabled (reset state), low-power consumption mode.
1: FM is enabled providing clock to the Skew corrector. The SKWEN bit must be set only after programming the SKW(3-0) bits.

Bit 6 = **SKDIV2:** *Skew Divide-by-2 Enable bit.*
0: Divide-by-2 disabled.
1: Divide-by-2 enabled.
This bit must be kept in reset state.

Bit 5:4 = Reserved. These bits are forced to 0 by hardware.

Bit 3:0 = **SKW:** *Skew Counter.*
These 4 bits program the down-counter inserted in the feedback loop of the Frequency Multiplier which generates the internal multiplied frequency PIXCLK. The PIXCLK value is calculated as follows :

If FPIXC=0 :

F(PIXCLK)=Crystal frequency * [(SKW(3:0)+1)]/2

If FPIXC=1 :

F(PIXCLK)=Crystal frequency * [(SKW(3:0)+1)]

Note: To program the FPIXC bit, refer to the description of the OSDER register in the OSD chapter.

MAIN CLOCK CONTROL REGISTER (MCCR)

R253 - Read/ Write
Register Page: 43
Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
FMEN	FMSL	0	0	FML3	FML2	FML1	FML0

The HALT mode forces the register to its initialization state.

Bit 7 = **FMEN:** *Frequency Multiplier Enable bit.*
0: FM disabled (reset state), low-power consumption mode.
1: FM is enabled, providing clock to the CPU. The FMEN bit must be set only after programming the FML(3:0) bits.

Bit 6= **FMSL:** *Frequency Multiplier Select bit.*
This bit controls the choice of the ST9 core internal frequency between the external crystal frequency and the Main Clock issued by the frequency multiplier.

In order to secure the application, the ST9 core internal frequency is automatically switched back to the external crystal frequency if the frequency multiplier is switched off (FMEN =0) regardless of the value of the FMSL bit. Care must be taken to reset the FMSL bit before any frequency multiplier can restart (FMEN set back to 1).

After reset, the external crystal frequency is always sent to the ST9 Core.

Bit 5:4 = Reserved. These bits are forced to 0 by hardware.

Bit 3:0 = **FML:** *FM Counter.*
These 4 bits program the down-counter inserted in the feed-back loop of the Frequency Multiplier which generates the internal multiplied frequency Fimf. The Fimf value is calculated as follows :

Fimf = Crystal frequency * [(FML(3:0) + 1)] /2

7 I/O PORTS

7.1 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.

7.2 SPECIFIC PORT CONFIGURATIONS

Refer to the Pin Description chapter for a list of the specific port styles and reset values.

7.3 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 33. 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 33. I/O Register Map

	GROUP E			GROUP F PAGE 2	GROUP F PAGE 3	GROUP F PAGE 43	
	System Registers		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	Reserved	R247
			F6h	P1C2	P5C2		R246
E5h	P5DR	R229	F5h	P1C1	P5C1		R245
E4h	P4DR	R228	F4h	P1C0	P5C0		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

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.

7.4 INPUT/OUTPUT BIT CONFIGURATION

By programming the control bits PxC0.n and PxC1.n (see Figure 34) 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 7.5). 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 35.

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 36).

– 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 34. Control Bits

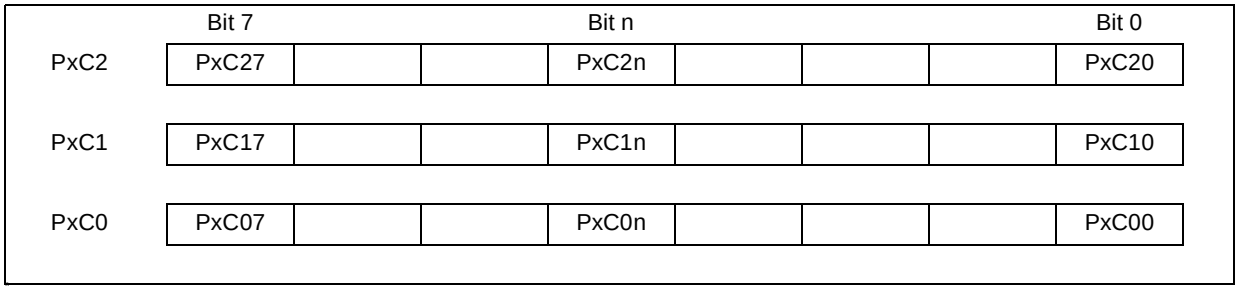


Table 16. Port Bit Configuration Table (n = 0, 1... 7; X = port number)

	General Purpose I/O Pins								A/D Pins
PXC2n	0	1	0	1	0	1	0	1	1
PXC1n	0	0	1	1	0	0	1	1	1
PXC0n	0	0	0	0	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 35. Basic Structure of an I/O Port Pin

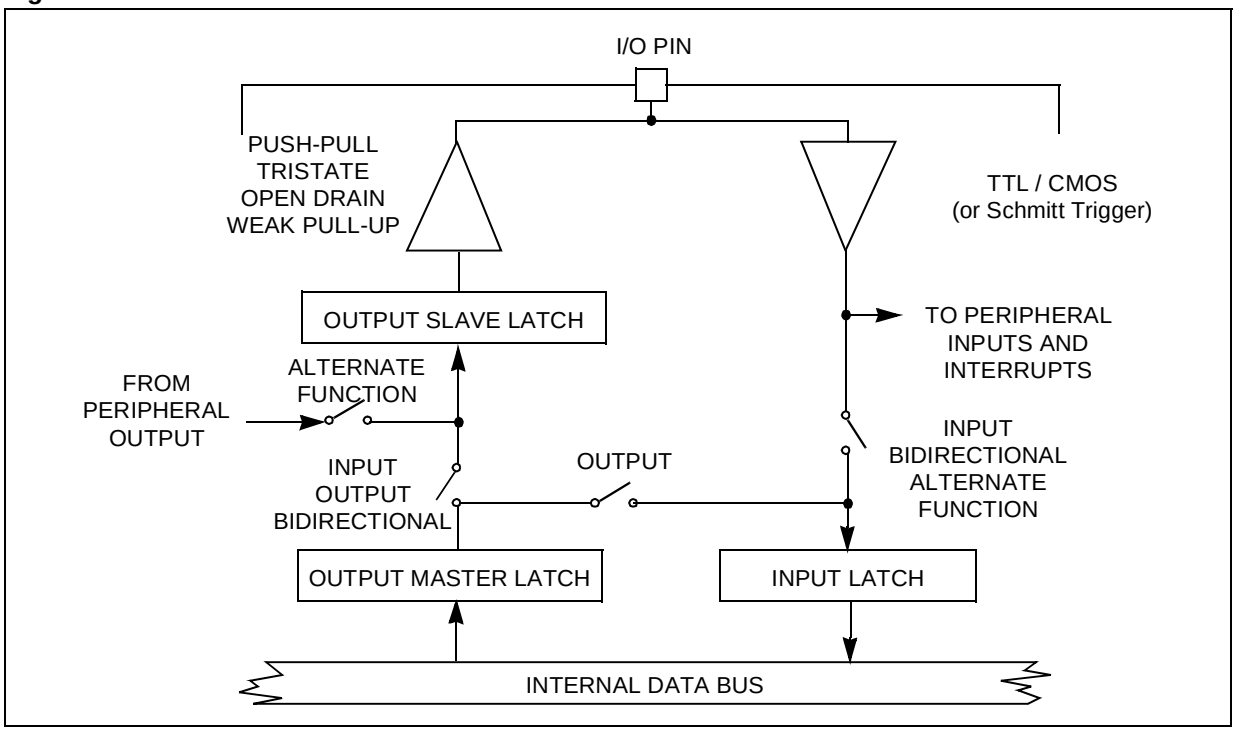


Figure 36. Input Configuration

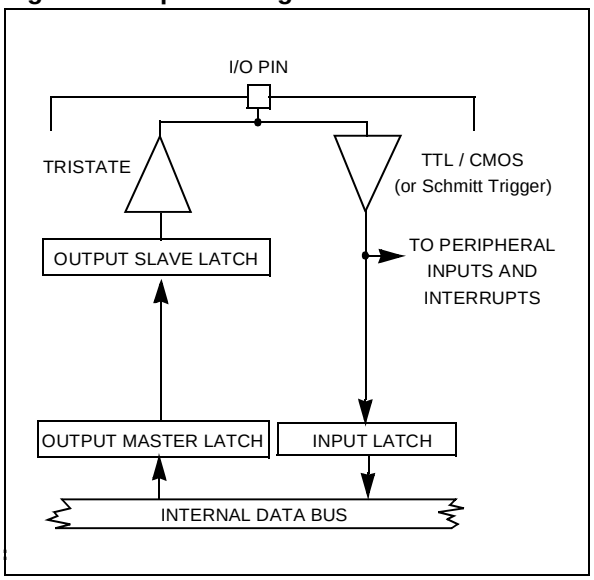
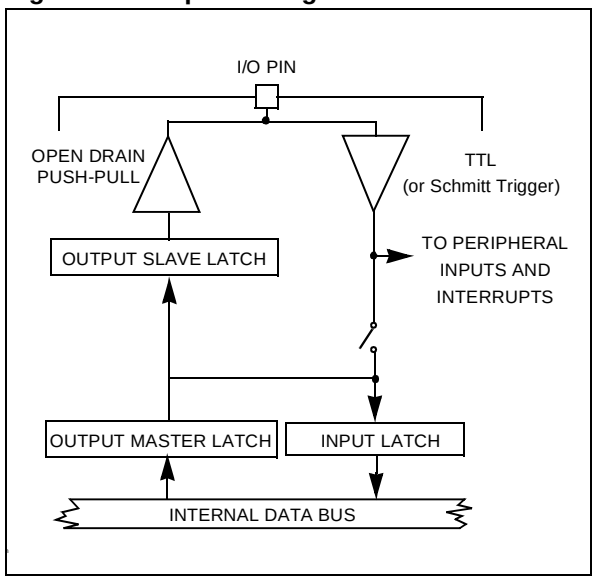


Figure 37. Output Configuration



INPUT/OUTPUT BIT CONFIGURATION (Cont'd)

When Px.n is programmed as an Output: (Figure 37)

– 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 38)

– 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 39)

– 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 38. Bidirectional Configuration

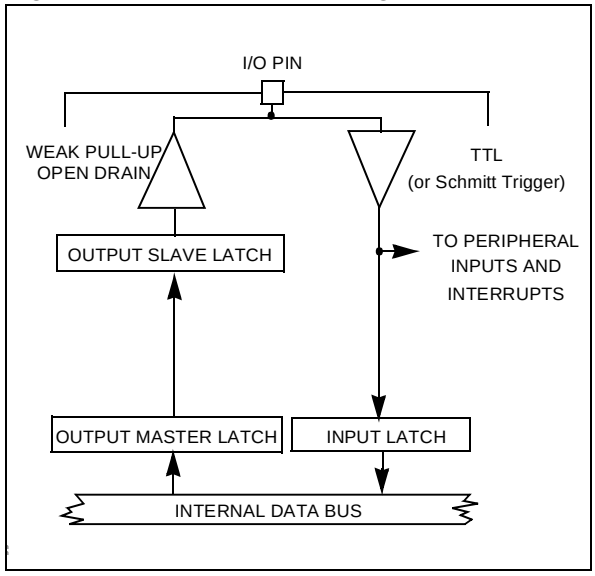
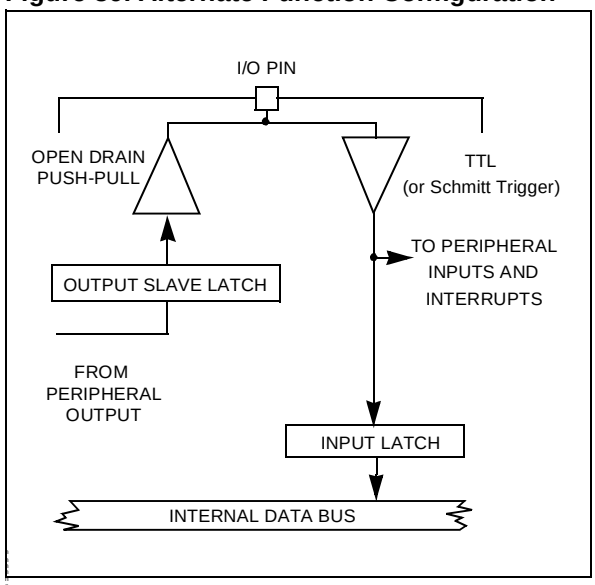


Figure 39. Alternate Function Configuration



7.5 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

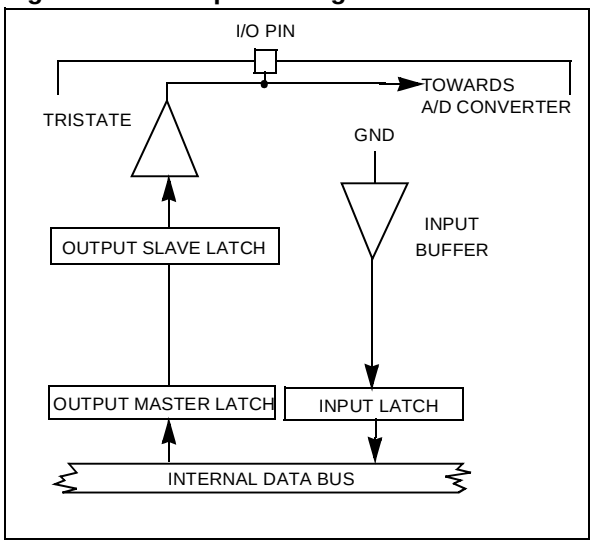
7.5.1 Pin Declared as I/O

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).

7.5.2 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 40).

Figure 40. A/D Input Configuration



7.5.3 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.*

7.6 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	
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).

8 ON-CHIP PERIPHERALS

8.1 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.

8.1.1 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

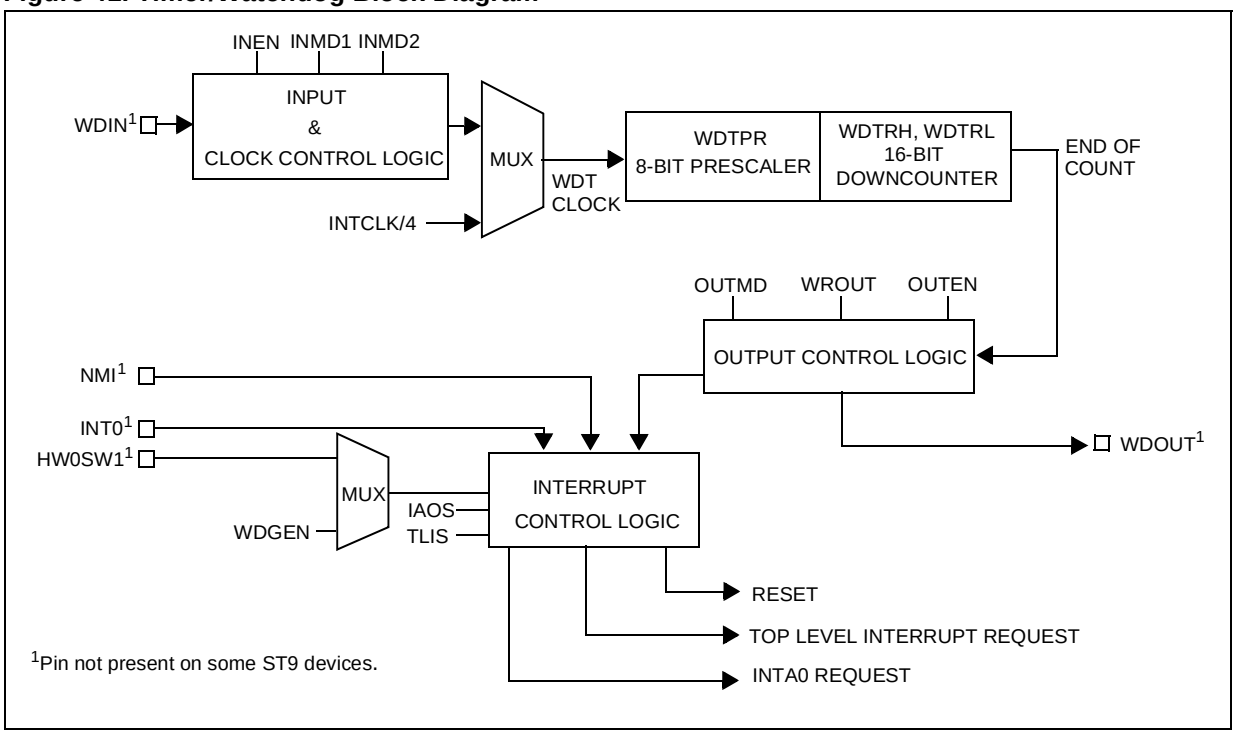
– WDOUT Square wave or PWM signal output

– INT0 External interrupt input

– NMI Non-Maskable Interrupt input

– HW0SW1 Hardware/Software Watchdog enable.

Figure 41. Timer/Watchdog Block Diagram



TIMER/WATCHDOG (Cont'd)

8.1.2 Functional Description

8.1.2.1 External Signals

The HW0SW1 pin can be used to permanently enable Watchdog mode. Refer to section 8.1.3.1 on page 78.

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.

8.1.2.2 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.

8.1.2.3 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.

8.1.2.4 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.

8.1.2.5 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.

8.1.2.6 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)

8.1.2.7 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.

8.1.2.8 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.

8.1.2.9 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.

8.1.2.10 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.

8.1.3 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.

8.1.3.1 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.

8.1.3.2 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)

8.1.3.3 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.

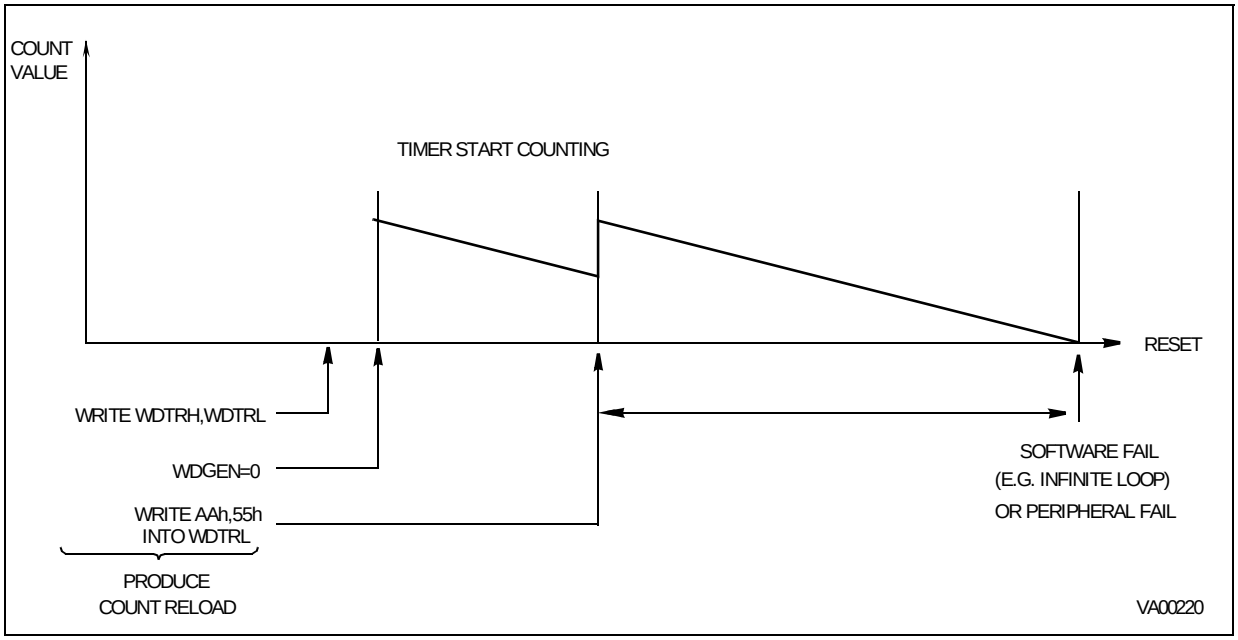
To reload the counter, the two writing operations must be performed sequentially without inserting other instructions that modify the value of the WDTLR register between the writing operations. The maximum allowed time between two reloads of the counter depends on the Watchdog timeout period.

8.1.3.4 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 42. Watchdog Timer Mode



TIMER/WATCHDOG (Cont'd)

8.1.4 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 43.

Note: Software traps can be generated by setting the appropriate interrupt pending bit.

Table 17 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 43. Interrupt Sources

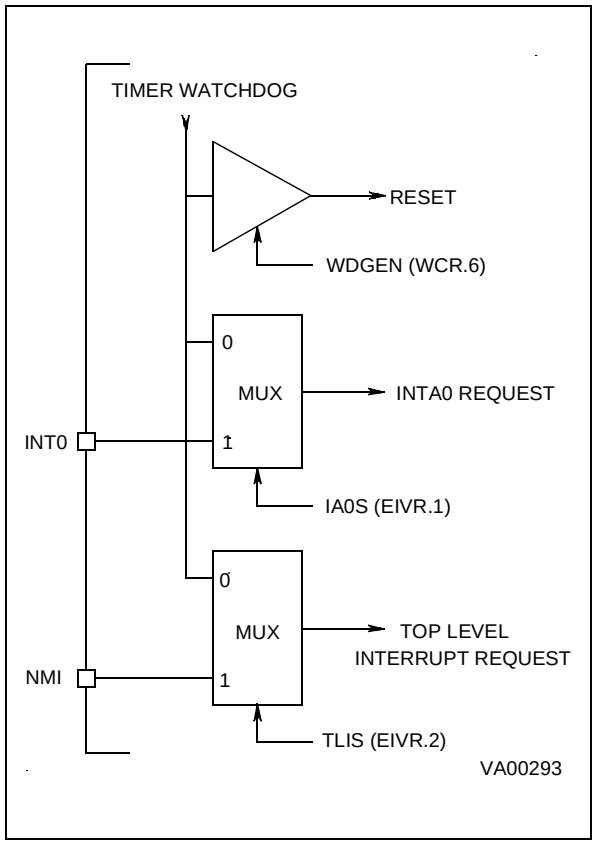


Table 17. Interrupt Configuration

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)

8.1.5 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.

8.2 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.

8.2.1 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.

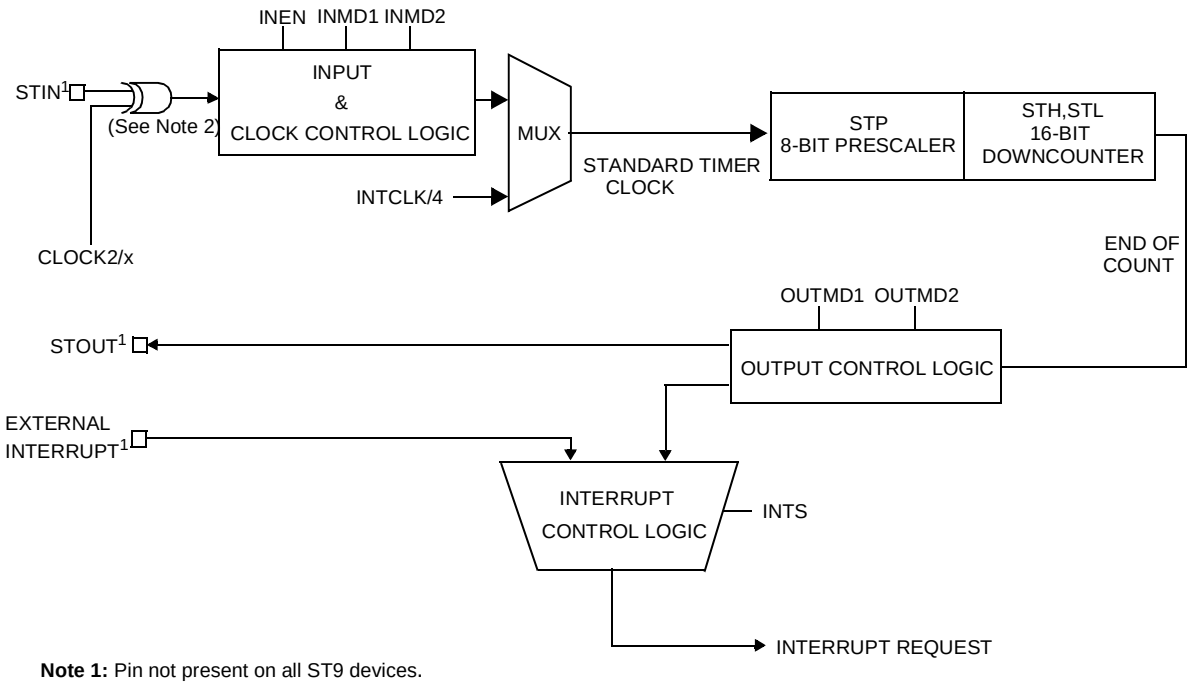
STOUT can be used to generate a Square Wave or Pulse Width Modulated signal.

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 44. 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)

8.2.2 Functional Description

8.2.2.1 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.

8.2.2.2 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.

8.2.2.3 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)

8.2.2.4 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.

8.2.3 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.

8.2.4 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)

8.2.5 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 8.2.2.2, 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 8.2.2.4.

OUTMD1	OUTMD2	Mode
0	**0**	No output mode
0	**1**	Square wave output mode
1	**x**	PWM output mode

8.3 MULTIFUNCTION TIMER (MFT)

8.3.1 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.

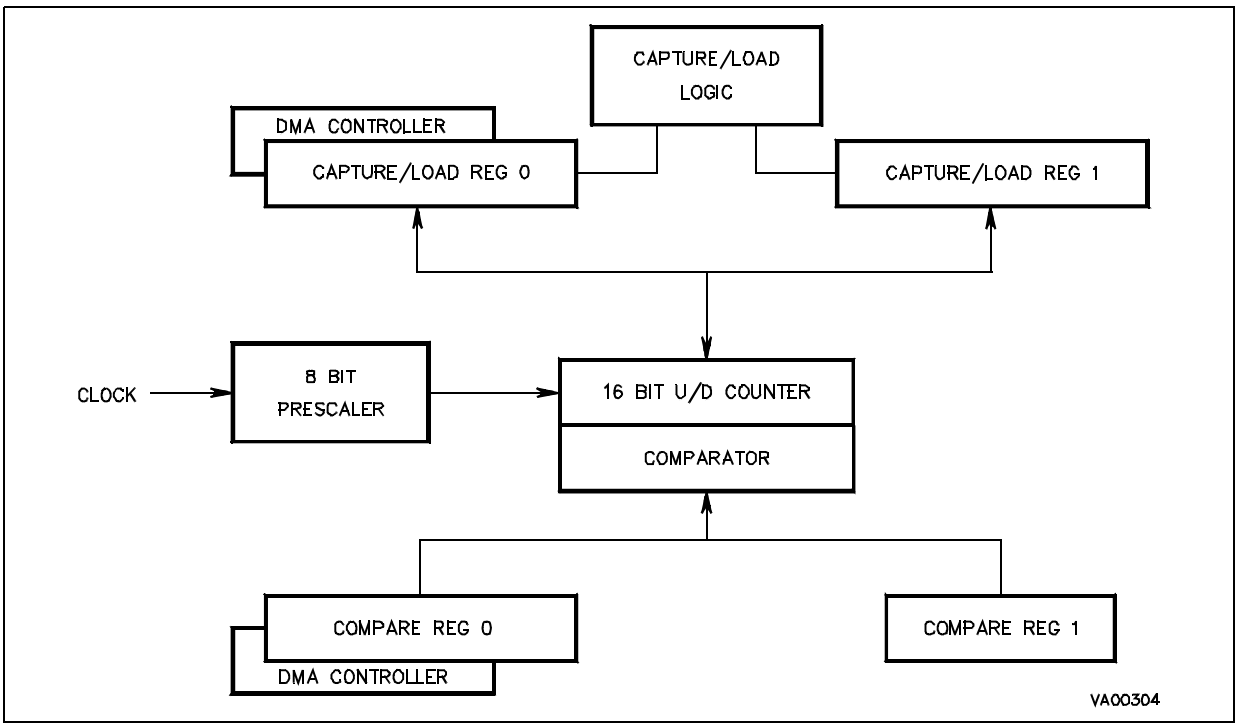
The two external inputs may be individually programmed to detect any of the following:

– rising edges

– falling edges

– both rising and falling edges

Figure 45. 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:

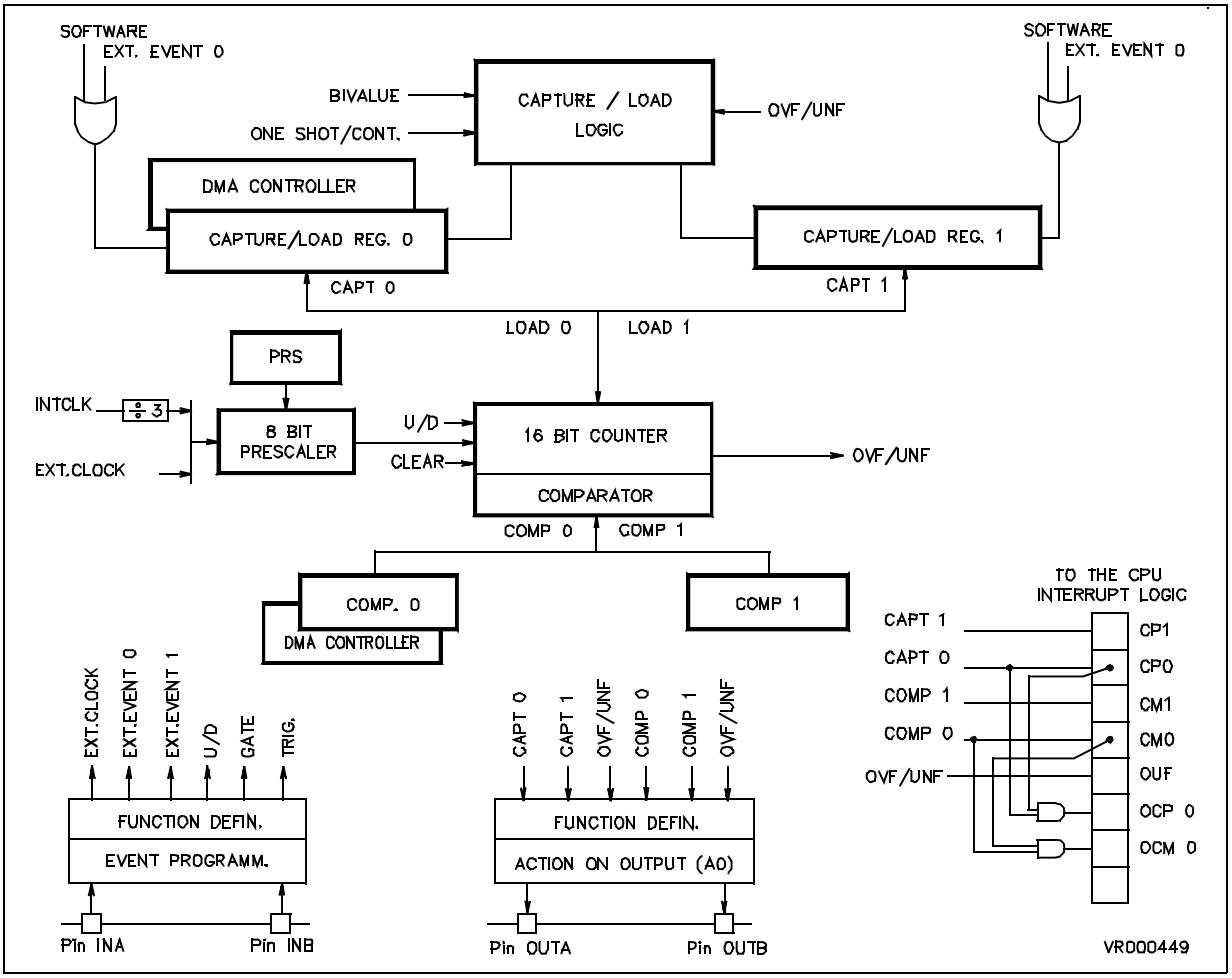
– Nop, Set, Reset, Toggle

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 46. Detailed Block Diagram



MULTIFUNCTION TIMER (Cont'd)

8.3.2 Functional Description

The MFT operating modes are selected by programming the Timer Control Register (TCR) and the Timer Mode Register (TMR).

8.3.2.1 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 21).

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 8.3.2.11).

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."

8.3.2.2 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.

8.3.2.3 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.

8.3.2.4 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.

8.3.2.5 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.

8.3.2.6 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.

8.3.2.7 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)

8.3.2.8 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).

8.3.2.9 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".

8.3.2.10 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).

8.3.2.11 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 18 below:

Table 18. 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.

8.3.2.12 Parallel Mode

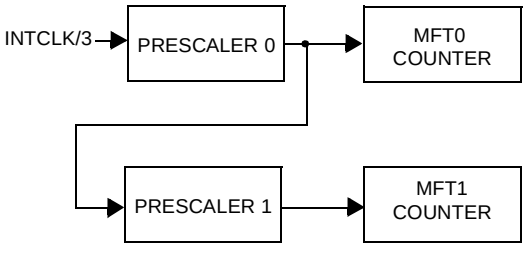
When two MFTs are present on an ST9 device, the parallel mode is entered when the ECK bit in the TMR register of Timer 1 is set. The Timer 1 prescaler input is internally connected to the Timer 0 prescaler output. Timer 0 prescaler input is connected to the system clock line.

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}.

8.3.2.13 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 47. 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)

8.3.3 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 19

Table 19. 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)

8.3.3.1 TxINA = I/O - TxINB = I/O

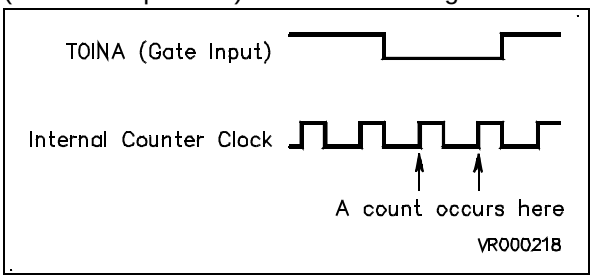
Input pins A and B are not used by the Timer. The counter clock is internally generated and the up/down selection may be made only by software via the UDC (Software Up/Down) bit in the TCR register.

8.3.3.2 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.

8.3.3.3 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.



8.3.3.4 TxINA = Gate - TxINB = Trigger

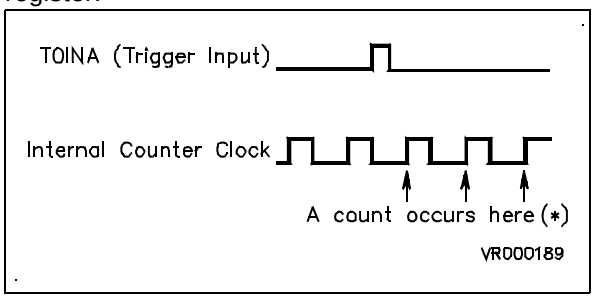
Both input pins A and B are connected to the timer, with the resulting effect of combining the actions relating to the previously described configurations.

8.3.3.5 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.

8.3.3.6 TxINA = Trigger - TxINB = I/O

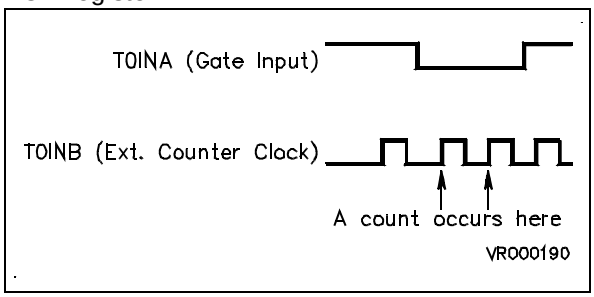
The signal applied to input pin A acts as a trigger for REG0R, initiating the action for which the register was programmed (i.e. a reload or capture). The prescaler clock is internally generated and the up/down selection may be made only by software via the UDC (Software Up/Down) bit in the TCR register.



(*) The timer is in One shot mode and REGOR in Reload mode

8.3.3.7 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.



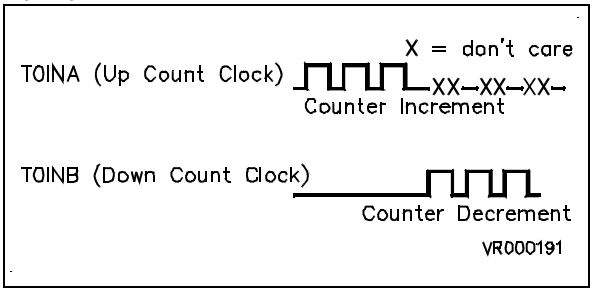
8.3.3.8 TxINA = Trigger - TxINB = Trigger

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)

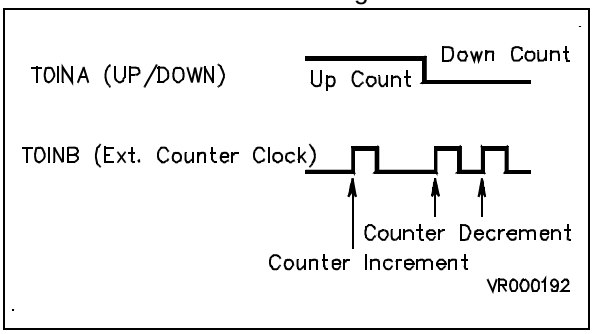
8.3.3.9 TxINA = Clock Up - TxINB = Clock Down

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.



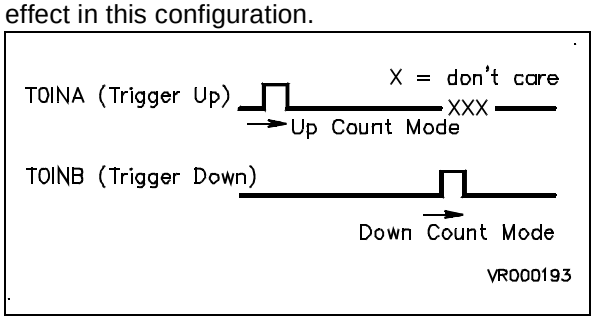
8.3.3.10 TxINA = Up/Down - TxINB = Ext Clock

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.



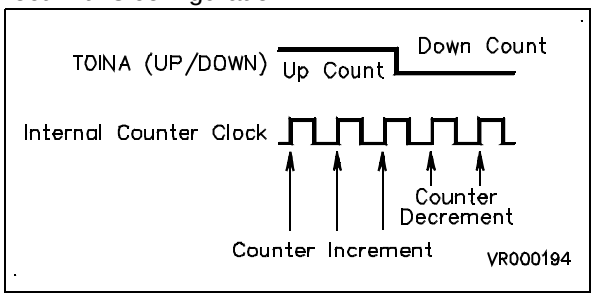
8.3.3.11 TxINA = Trigger Up - TxINB = Trigger Down

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.



8.3.3.12 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.

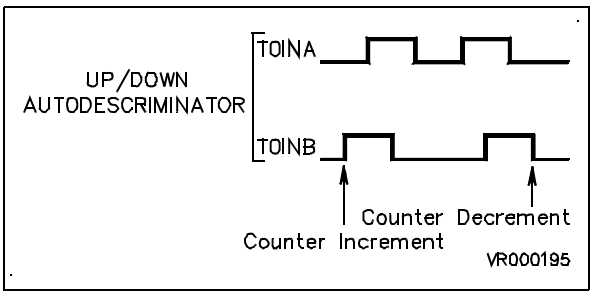


MULTIFUNCTION TIMER (Cont'd)

8.3.3.13 Autodiscrimination Mode

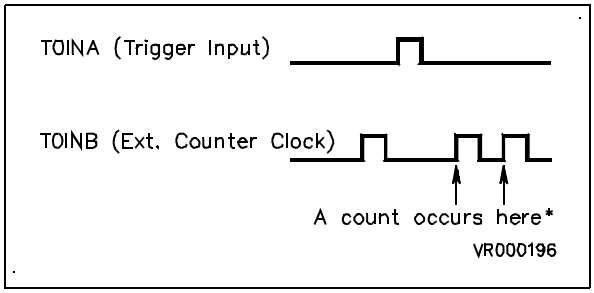
The phase between two pulses (respectively on input pin B and input pin A) generates a one step up (or down) count, so that the up/down control and the counter clock are both external. Thus, if the rising edge of TxINB arrives when TxINA is at a low level, the timer is incremented (no action if the rising 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.



8.3.3.14 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

8.3.3.15 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.

8.3.3.16 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)

8.3.4 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

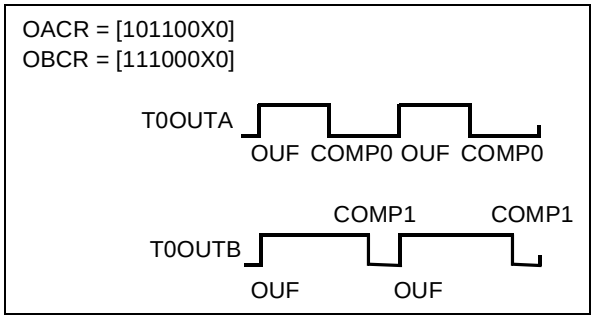
Furthermore an On Chip Event signal can be driven by two of the three sources: the Over/Underflow event and Compare 0 event by programming the CEV bit of the OACR register and the OEV bit of OBCR register respectively. This signal can be used internally to synchronise another on-chip peripheral.

Output Waveforms

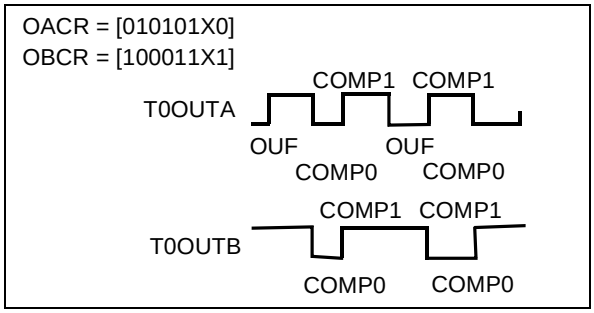
Depending on the programming of OACR and OBCR, the following example waveforms can be generated on TxOUTA and TxOUTB pins.

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.

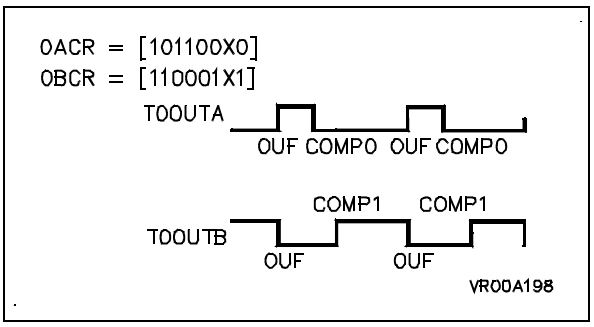


For a configuration where TxOUTA is driven by the Over/Underflow, by Compare 0 and by Compare 1; TxOUTB is driven by both Compare 0 and Compare 1. OACR is programmed with TxOUTA preset to "0". OUF toggles Output 0, as do CM0 and CM1. OBCR is programmed with TxOUTB preset to "1". OUF does not affect the output; CM0 resets TxOUTB and CM1 sets it.

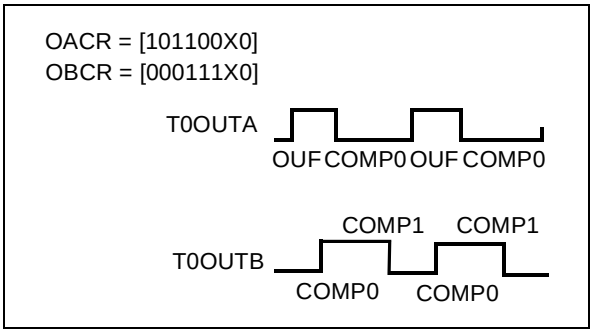


MULTIFUNCTION TIMER (Cont'd)

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.



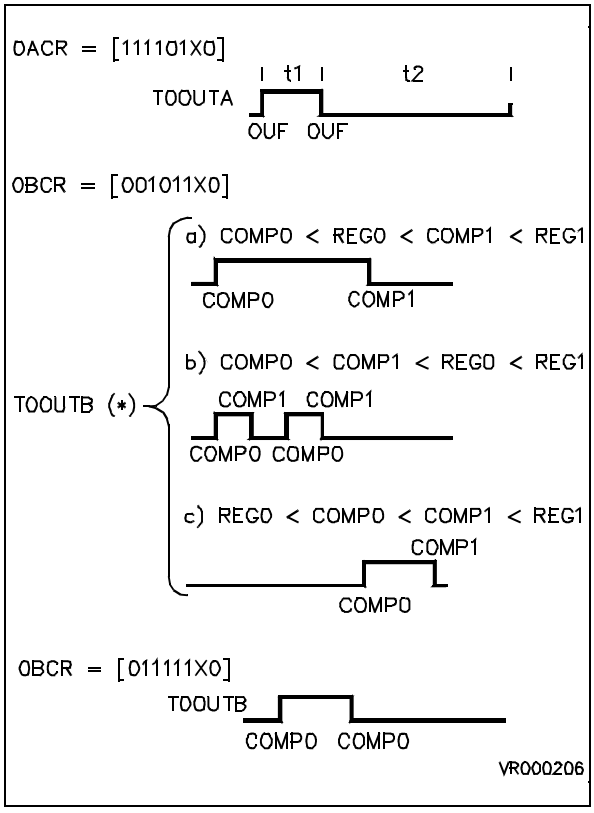
For a configuration where TxOUTA is driven by the Over/Underflow and by Compare 0, and TxOUTB is driven by Compare 0 and 1. OACR is programmed with TxOUTA preset to "0". OUF sets TxOUTA, CM0 resets it and CM1 has no effect. OBCR is programmed with TxOUTB preset to "0". OUF has no effect, CM0 sets TxOUTB 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)

8.3.5 Interrupt and DMA

8.3.5.1 Timer Interrupt

The timer has 5 different Interrupt sources, belonging to 3 independent groups, which are assigned to the following Interrupt vectors:

Table 20. 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).

8.3.5.2 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.

8.3.5.3 DMA Pointers

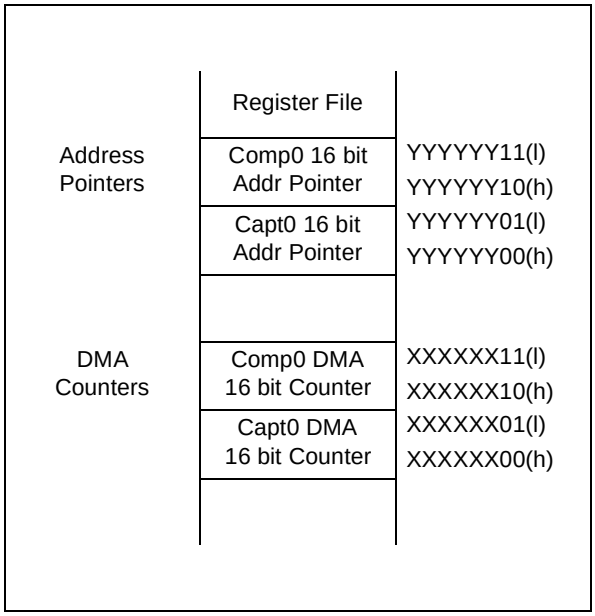
The 6 programmable most significant bits of the DMA Counter Pointer Register (DCPR) and of the DMA Address Pointer Register (DAPR) are common to both channels (Comp0 and Capt0). The Comp0 and Capt0 Address Pointers are mapped as a pair in the Register File, as are the Comp0 and Capt0 DMA Counter pair.

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.

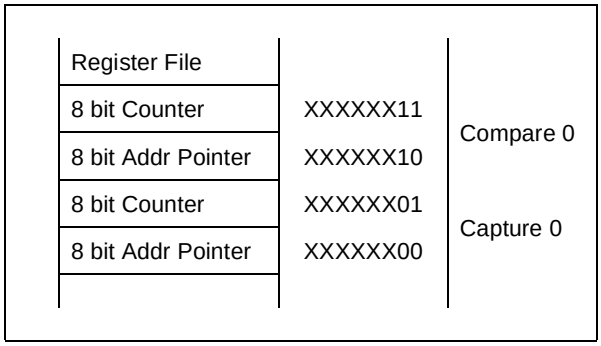
The DMA Address Pointer Register (DAPR) is not used in this case, but must be considered reserved.

Figure 48. Pointer Mapping for Transfers between Registers and Memory



MULTIFUNCTION TIMER (Cont'd)

Figure 49. Pointer Mapping for Register to Register Transfers



8.3.5.4 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.

8.3.5.5 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)

8.3.5.6 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 8.3.5.7).

– 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.

8.3.5.7 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.

8.3.6 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 21).
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 21.

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 21.

Table 21. 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 8.3.2.12.

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 22 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 22 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 22 for the list of actions that can be configured.

Table 22. 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 23 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 23 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 23 for the list of actions that can be configured.

Table 23. 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	OCP 0	OCM 0	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
GT-IEN	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
DAP 7	DAP 6	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.

8.4 OSDRAM CONTROLLER

8.4.1 Introduction

The OSDRAM Controller handles the interface between the Display Controller, the CPU and the OSDRAM.

The time slots are allocated to each unit in order to optimize the response time.

The main features of the OSDRAM Controller are the following:

- Memory mapped in Memory Space (segment 22h of the MMU)
- DMA access for Display control
- Direct CPU access

8.4.2 Functional Description

The OSDRAM controller manages the data flows between the different sub-units (display controller, CPU) and the OSDRAM. A specific set of buses (16-bit data, 9-bit addresses) is dedicated to these data flows. The OSDRAM controller accesses these buses in real time. The OSDRAM controller has registers mapped in the ST9 register file.

As this OSDRAM controller has also to deal with TV real time signals (On-Screen-Display), a specific controller manages all exchanges:

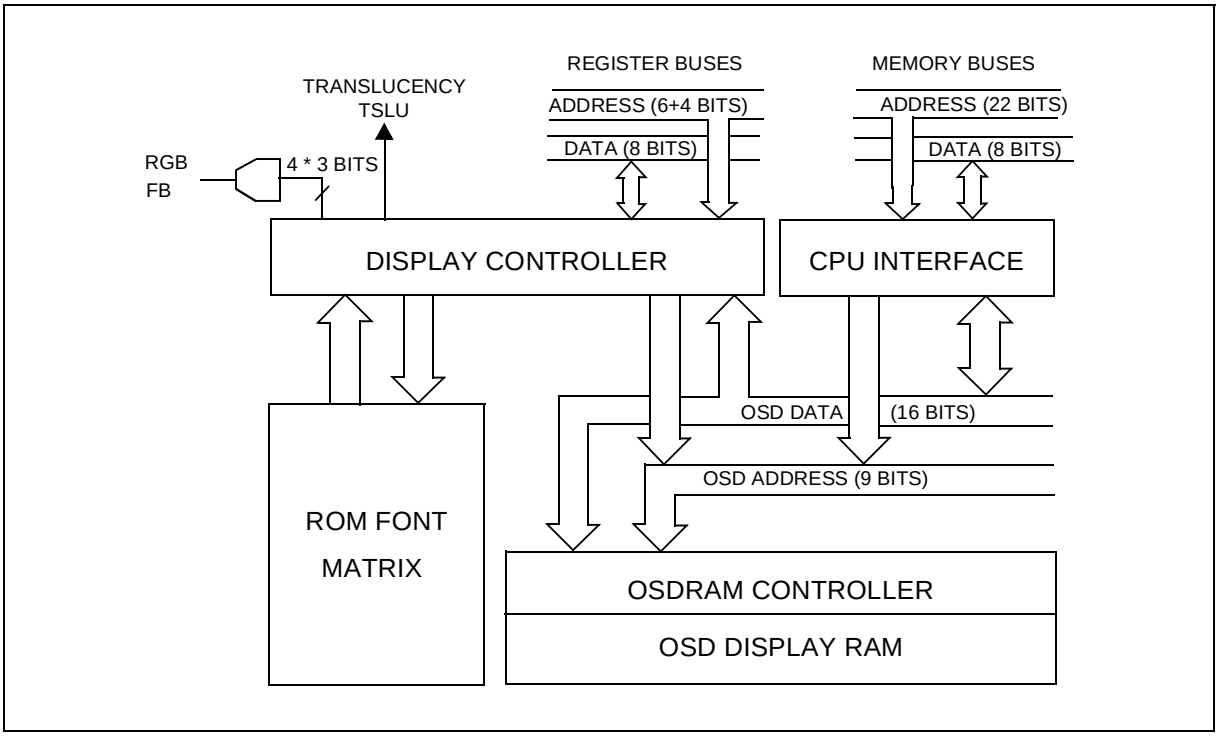
– Its timing generator uses the same frequency generator as the Display (Pixel frequency multiplier),

– Its controller can work in two TV modes:
  – **Single mode**: all time slots are dedicated to the CPU.
  – **Shared mode**: time slots are shared between the CPU and the Display. The shared mode is controlled by the Display controller.

– Its architecture gives priority to the TV real time constraints: whenever there is access contention between the CPU and the Display (shared mode), the CPU is automatically forced in a “wait” configuration until its request is served.

– Its controller enables a third operating mode (stand-alone mode) which allows the application to access the OSDRAM while the Display is turned off. In this case, the OSDRAM controller uses the CPU main clock.

Figure 50. Display Architecture Overview



OSDRAM CONTROLLER (Cont'd)

8.4.2.1 Time Sharing during Display

The time necessary to display a character on the screen defines the basic repetitive cycle of the OSDRAM controller. This whole cycle represents therefore 18 clock periods. This cycle is divided in 9 sub-cycles called "slots". Each slot is allocated in real-time either to the CPU or the Display:

– In single mode, this 9-slot cycle is repeated continuously providing only CPU slots (single cycle), until the OSDRAM controller is switched off by the main program execution.

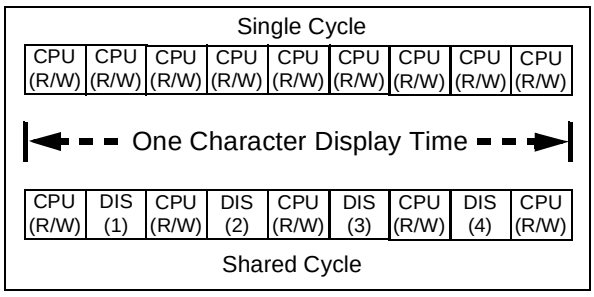
– In shared mode, this 9-slot cycle provides Display slots followed by CPU slots.

Each slot represents a two-byte exchange (read or write) between the OSDRAM memory and the other units:

Display Reading slot (DIS): 16 bits are read from the OSDRAM and sent to the display unit, the address being defined by the display address generator.

Direct CPU Access slot (CPU): 16 bits are exchanged (read or write) between the OSDRAM and its controller but only 8 bits are exchanged with the CPU, the address being defined by the CPU memory address bus.

Figure 51. Time Sharing during Display



Display reading is handled as follows:

– DIS(1) & DIS(2) are dedicated to reading the character code, its parallel attributes & associated palette pointer.

– DIS(3) provides the foreground palette.

– DIS(4) provides the background palette. In case of Underline activation (refer to the OSD controller paragraph for more details), the DIS(4) slot is no longer provides the background palette content (useless information) but recovers the Underline color set data.

The CPU write accesses are handled as follows:

Because of the 16-bit word width inside the OSDRAM matrix, it is obviously necessary to perform a CPU write access in 2 steps:

– Reading the OSDRAM word

– Rewriting it with the same values except for the 8 modified bits.

Each time a CPU write operation is started, the next following CPU slot will be used as a read slot, the effective write to the OSDRAM being completed at the next CPU slot.

8.4.2.2 Time Sharing within the TV line

At the beginning of each TV line, the OSDRAM is accessed (read) by the Display controller in order to get all the row attributes. When the TV line is recognized as the one where data have to be displayed, the Shared cycle is activated at the time the data has to be processed for display.

OSDRAM CONTROLLER (Cont'd)

8.4.3 OSDRAM Controller Reset Configuration

During and after a reset, the OSDRAM access is disabled.

When the OSDRAM controller is software disabled, it will:

1. Complete the current slot.
2. Complete any pending write operation (a few slots may elapse).
3. Switch off any OSDRAM interface activity.

8.4.3.1 OSDRAM Controller Running Modes

2 control bits called "OSDE" (OSD Enable) and "DION" (Display ON) are used to enable the OSDRAM controller. Both are also shared by the Display controller.

These 2 bits are located in the OSDER register. This register is described in the On Screen Display Controller Chapter.

8.4.3.2 CPU Slowdown on OSDRAM access

As described above, the OSDRAM controller puts priority on TV real time constraints and may slowdown the CPU (through "wait" cycle insertion) when any OSDRAM access is requested. The effective duration of the CPU slowdown is a complex function of the OSDRAM controller working mode and of the respective PIXCLK frequency (OSDRAM frequency) and the Core INTCLK frequency.

8.4.3.3 OSDRAM Mapping

The OSDRAM is mapped in the memory space, segment 22h, starting from address 0000h to address 017Fh (384 bytes).

The OSDRAM mapping is described in the On Screen Display Controller Chapter.

8.5 ON SCREEN DISPLAY CONTROLLER (OSD)

8.5.1 Introduction

The OSD displays Closed Captioning (EIA708 specification) or other character data and menus on a TV screen.

Each row can be defined through three different Display configurations:

– **Serial mode:** each character is defined by an 8-bit word which provides the character address into the Font ROM memory. Some codes are reserved for color controls and do not address any character description. They are displayed as spaces and as a direct consequence are active on a “word” basis. This mode fully supports the Closed Captioning format.

– **Basic parallel mode:** each character is defined by a 16-bit word which provides the character address in the Font ROM memory and its color attribute. This mode is called “parallel” as the colors are definable on a “per character” basis.

– **Extended parallel mode:** each character is defined by a 24-bit word which provides the character address in the Font ROM memory and its color and shape attributes. This mode is called “parallel” as the attributes are definable on a “per character” basis.

8.5.2 General Features

- 50/60 Hz and 100/120 Hz* operation
- 525/625 lines operation, 4/3 or 16/9 format
- Interlaced or progressive scanning
- 18x26 or 9x13 character matrix user definable in ROM. Both matrixes can be mixed.
- Up to 63 characters per row
- 7 character sizes in 18x26, 4 in 9x13
- 512 possible colors in 4x16-entry palettes
- 8 levels of translucency on Fast Blanking
- Serial character mode supporting Closed Captioning format
- Basic Parallel Mode for character based color definition
- Extended parallel mode for character based color and shape definition
- Mouse pointer (user definable in ROM)
- Rounding, fringing, shadowing, flashing, scrolling, italics, and various underlining modes

Definition of Terms used in this Chapter

– **Pixel**: minimum displayed element that the Display Controller can handle. Its vertical physical size is always one TV line. Its horizontal physical size is directly linked to the basic clock frequency (called Pixel clock) which synchronizes the OSD and is therefore independent of any magnification factor which may be applied to the displayed element.

– **Dot**: a dot defines the displayed element which corresponds to a single bit read from the Font ROM memory. A dot is represented on the screen by a "matrix" of pixels. The matrix size depends on the magnification factor applied.

8.5.3 Functional Description

All characters are user definable by masking the Font ROM content (except the one corresponding to code 00h which is reserved for test). Two different matrixes can be used and mixed:

– 18x26 character matrix

– 9x13 character matrix

The hardware display system has the capability of displaying one character row and requires the CPU to update the next display buffer prior to displaying the next row. Using a real time routine, the On Screen Display supports the display of as many character rows as the TV screen can physically handle.

The OSD can display up to 63 characters per row, depending on the row RAM buffer size (user definable, see Section 8.5.5.1).

The OSD is also designed to handle a mouse pointer (see Section 8.5.7).

A smart pixel processing unit provides extended features such as rounding, fringe, or shadowing for better picture quality. Other smart function such as flashing, scrolling, italics, underlining and mouse pointer allow the designing of a high quality display application.

The screen insertion of the displayed characters is fully synchronized by the vertical and horizontal TV synchronizing signals. The OSD controller generates the Red, Green, Blue and Fast Blanking video signals through 8-level DAC outputs. The Fast Blanking video signal can also be generated as a digital signal if needed.

*Available on some devices

OSD CONTROLLER (Cont'd)

8.5.3.1 Display Attributes

- Global screen attributes:
  - Border color
  - Border translucency
  - Turn all background color into border color
- Row parameters and attributes:
  - Row mode control (serial, basic parallel, extended parallel)
  - Row character count
  - Horizontal and Vertical shift
  - Active Range (used for vertical scrolling)
  - Font matrix selection
  - Size control (dot height and width definition)
  - Flashing control
  - Rounding control
  - Fringe control
- Serial character attributes:
  - Background color (16 user definable colors)
  - Foreground color (16 user definable colors)
  - Flashing control
  - Italics control
- Palette parallel character attributes:
  - Background color (16 user definable colors)
  - Foreground color (16 user definable colors)
- Shape parallel character attributes:
  - Character code extension
  - Double height
  - Double width
  - Foreground palette extension (32 user definable colors)
  - Background palette extension (32 user definable colors)
  - Flashing control
  - Shadowing control
  - Fringe control
- Mouse Pointer attributes:
  - Fringe control
  - Rounding control
  - Foreground color (32 user definable colors)
  - Double size
  - Horizontal and Vertical position

8.5.3.2 OSD Area

When the Display controller is turned on, the TV screen will show a specific color prior to any data display which is called the "Border Color". The effective border color is fully software programmable from a palette of 512 colors through 2 control registers.

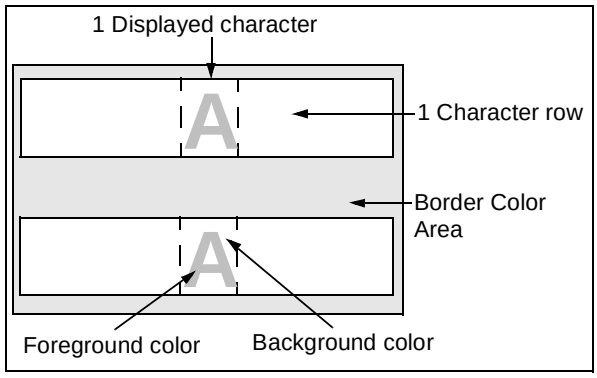
The border area translucency can be chosen from 8 different levels, from fully transparent to fully opaque. The 3 translucency control bits are accessible through the border color control registers.

When data are displayed by the OSD, they form rows of characters. All characters of one row are horizontally aligned.

For each displayed character, two kinds of colors must be programmed which are defined as:

- Background color
- Foreground color

Figure 52. OSD Area Description



8.5.3.3 Color Processing

Further color elements may be generated by the Display controller as a result of real time pixel calculations (they are not stored in the Font ROM memory). These are:

- Rounding pixels: they must be considered as "calculated" foreground pixels.
- Fringe pixels: they are always displayed with a black color and are never translucent.
- Underline pixels: they must be considered as "calculated" pixels. Their colors are defined through independent underline color values. Translucency levels are also programmable for underline pixels.

OSD CONTROLLER (Cont'd)

All colors are taken from a double Palette set (Background and Foreground) which are both OSD-RAM mapped and thus definable in real-time.

The priority of all color layers is, from highest to lowest:

Mouse, underline, foreground & rounding, fringe, background and border.

Character code 00h is a test character and has no user customized content. It is always displayed with a border color and will appear as a "non-displayed character".

8.5.3.4 Character Matrix Definition

A character is described by a matrix of dots stored in the Font ROM memory. Two matrix sizes are can be used to define each character pattern: either a 9x13 matrix or a 18x26 matrix. The matrix size can be redefined for each row Display buffer; mixing the 2 matrix sizes on a same screen is therefore possible.

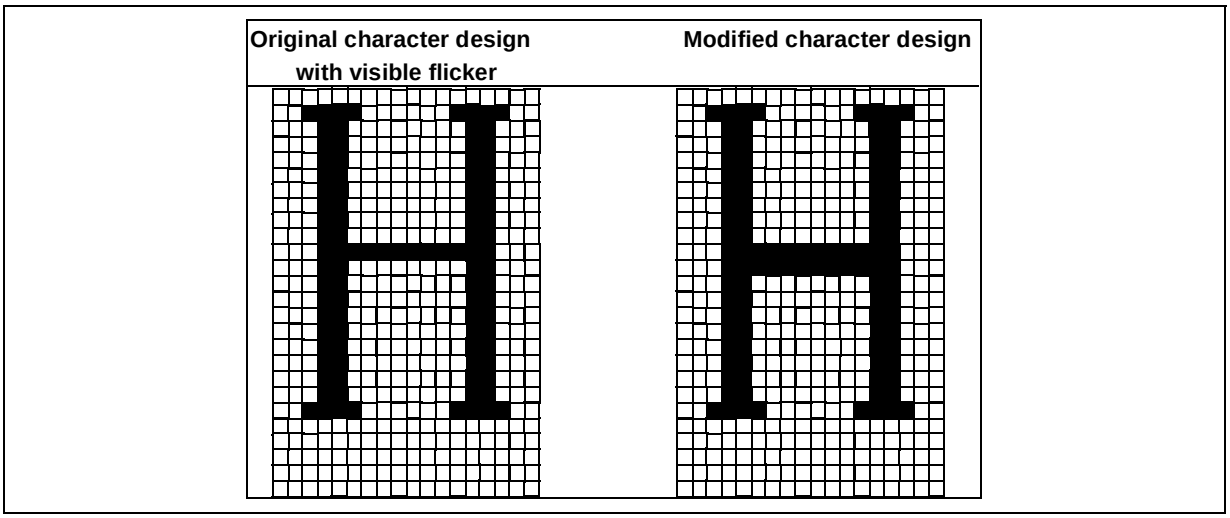
Refer to Figure 61, for an example of the Font ROM content.

The 9x13 matrix is a compressed format where each bit represents a 2x2 pixel dot (with no magnification applied). In interlaced mode, for each dot, 2 pixels are generated on a field, the 2 others on the other field; each row of the matrix is used for both fields.

The 18x26 matrix is an "expanded" format where each bit represents a 1x1 pixel dot (if no magnification). In interlaced mode, if no magnification is applied, each odd row of the matrix is displayed on a field, each even row on the other field.

WARNING: As a result, when displaying the 18x26 matrix with no vertical magnification and in interlaced mode, flicker may be visible on characters containing long single-pixel rows. To avoid this effect, either use double pixel horizontal lines, if possible, when designing the font (see Figure 53). Alternatively, use the double height attribute or choose the foreground and background colors for reduced contrast.

Figure 53. Avoiding Flicker on unmagnified 18x26 characters



OSD CONTROLLER (Cont'd)

8.5.3.5 Cursor & Flash

A cursor facility may be emulated under software control, using the "flash" attribute. This allows to have a "flash-on-word" in serial mode or a "flash-on-character" in extended parallel mode.

The cursor facility first requires activating the "flash on" row control bit (refer to Section 8.5.8.3), which acts as a general flash enable.

Then program the characters with Flashing Character attribute(s) (serial or extended parallel) at the required locations.

The flash effect is obtained by toggling the general flash enable bit.

Several flashing words or characters per screen can easily be implemented.

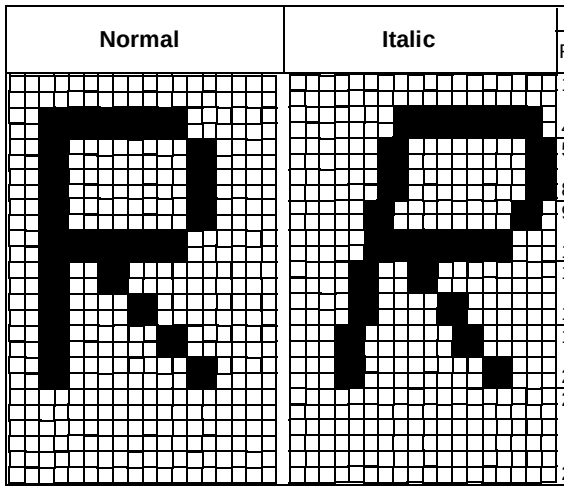
8.5.3.6 Italic Mode

The italics attribute is a serial attribute; this means that Italic mode is available in serial mode only.

In the matrix description that follows, line 1 is at the top of the character, line 13 (or line 26) is at the bottom.

The codes (seen as spaces) needed to activate and deactivate the Italic attribute provide a convenient method for solving border problems between italic and non-italic characters.

Figure 54. Italic Character Mode



18x26		9x13	
Row	Dot Shift	Row	Dot Shift
1 to 4	5	1 to 2	2.5
5 to 8	4	3 to 4	2
9 to 12	3	5 to 6	1.5
13 to 16	2	7 to 8	1
17 to 20	1	9 to 10	0.5
21 to 26	0	11 to 13	0

OSD CONTROLLER (Cont'd)

If the italic attribute is still active at the end of the row, the last character code to be displayed is truncated to the vertical position where it would have finished without the italic attribute.

If the background color is changed while italics are enabled, then the color attribute (seen as a space) is not slanted.

The right edge background corresponding to a control character is never slanted.

When a background code follows an italic character, then the background color of the italic character extends half way into the displayed background code location, regardless of the background Palette M bit (refer to Section 8.5.6.4 for a description of the M bit).

In the case where a background code is followed by a second background code while italics are on, the first background color will extend half way into the second background location.

The edges of the 00h code (test character seen as a border color) are never slanted.

The right edge of the 00h code is never slanted.

When a 00h code follows an italic character, then the background color of the italic character extends half way into the 00h code location.

In case a 00h code is followed by a background code while italics are on, the 00h code is never extended into the following code location.

8.5.3.7 Rounding and Fringe

Rounding can be enabled or disabled row by row (see Section 8.5.8.3).

For a 18x26 matrix size, there is no rounding facility when the character size is (1X,1Y).

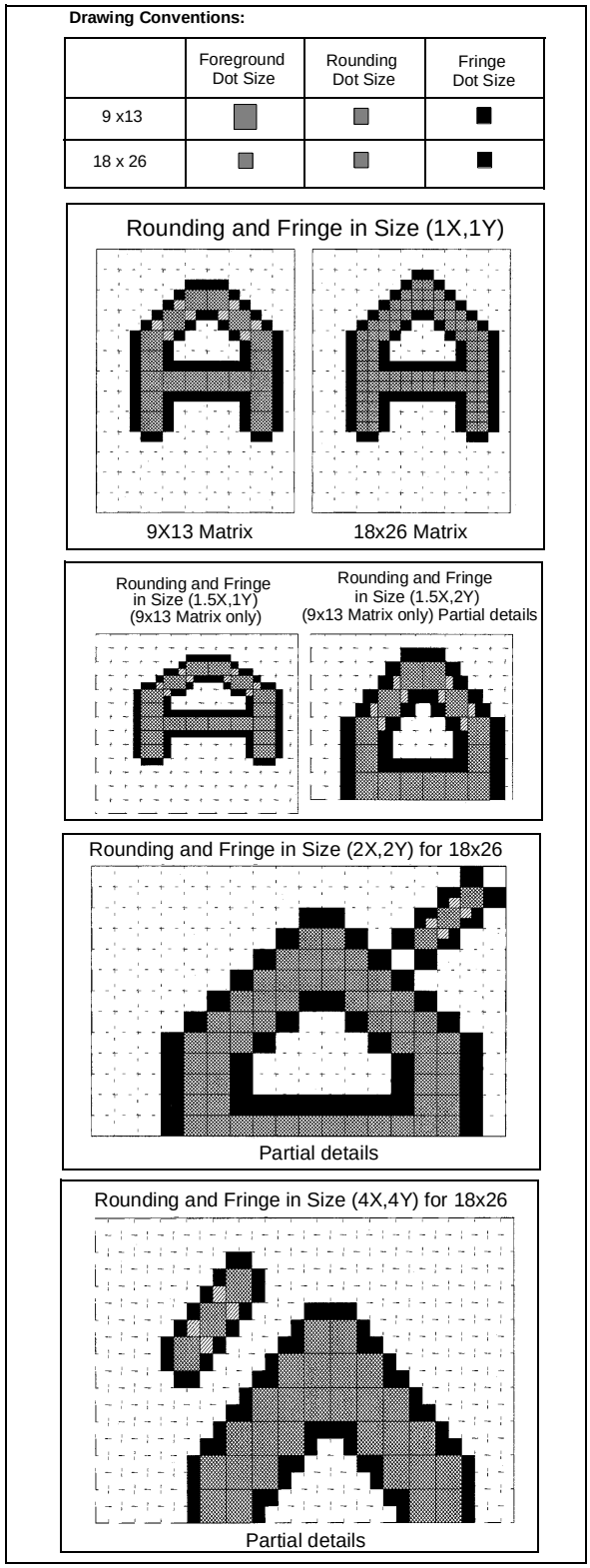
For any other (X,Y) size combinations, the rounding facility is allowed for the 18x26 font matrix, and rounding is available in any of the 3 row modes (serial, basic parallel, extended parallel).

The fringe mechanism can be enabled or disabled row by row (see row attributes description), but it can also be defined on a character basis in extended parallel mode (see shape parallel attributes for more details).

The fringe mechanism can be activated for any size and both matrix formats.

Please note that, for both matrixes, in the case of fringe usage in 1Y vertical size and interlaced mode, a flicker may appear on the screen as the fringe information is built on a field basis.

Figure 55. Rounding and Fringe



OSD CONTROLLER (Cont'd)

8.5.3.8 Scrolling

The row RAM buffer architecture of the Display allows all scrolling operations to be performed very easily by software: scroll up, scroll down, scroll left, scroll right and any horizontal/vertical mix.

In addition to the character row scrolling, a vertical "smooth" scrolling has been implemented. It requires defining the "active range" for each displayed row (refer to Section 8.5.8.4).

8.5.3.9 Color Palettes

The Display controller provides 4 user-definable palettes: two 16-color foreground palettes (one basic and one extended), and two 16-color background palettes (one basic and one extended). Each color is defined using 16 bits:

– **Foreground palette:** 3 bits for red level, 3 bits for green level, 3 bits for blue level, 3 bits for translucency, and 3 bits for the underline mode control.

– **Background palette:** 3 bits for red level, 3 bits for green level, 3 bits for blue level, 3 bits for translucency, and 1 bit for immediate color change control.

Two palettes are always available (one foreground palette and one background palette). The 2 other palettes are only accessible in extended parallel mode.

The palettes available in serial and basic parallel modes depend on the value of the PASW bit in the OSDDR register.

8.5.3.10 Underline Mode Control

The OSD is able to underline any character of the 9x13 or 18x26 matrix with 3 different colors selected from a palette, and 2 different dot lines.

Using 3 bits (see the Color Palettes paragraph above), it is possible to define the underline mode associated with each foreground palette entry.

In 9x13 mode, single or double underline can be set on lines 12 and 13 with the foreground color or two specific colors defined in underline color sets 1 & 2.

In 18x26 mode, single or double underline can be set on lines 23-24 and 25-26 with the foreground color or two specific colors defined in underline color sets 1 & 2.

For more details please refer to Section 8.5.6.

OSD CONTROLLER (Cont'd)

8.5.3.11 Translucency Function

The translucency feature is designed to provide a better OSD quality while displaying rows in mixed mode.

Instead of forcing the background color of any character to any full intensity color, (which will prevent the viewer from seeing a significant amount of the video picture), or having a fully transparent background (i.e. no background) which makes the OSD more difficult to read for the viewer, the translucency provides a real time mix between both the video and the OSD background information. This feature will appear as a "boxing" effect around all the displayed characters.

The translucency can be handled in two different ways at application level, depending on the video processor used in the application.

1. When the video processor accepts an analog control of the fast switch OSD signal (called "FB"), the translucency can be handled directly through the real time amplitude of the FB signal (refer to Color attribute control for Border, Background palette and Underline color settings).
2. When the video processor accepts only a digital FB signal, the translucency function may be implemented on the chassis with the help of an additional digital output of the MCU, which is provided as an I/O pin alternate function.

This digital output called "TSLU" is active (set to "1") when the OSD displays the Background and Foreground or when the mute is active (refer to the description of the LSM[2:0] bits in the OSDMR register) and is inactive the rest of the time (during foreground or if no display), including character 00h.

This TSLU signal is controlled by the TSLE bit in the OSDER register. When not used, (TSLE=0), the TSLU signal is held at "1" by hardware.

Application Note

In order to enable the Translucency Function (See Example No. 2, above), the following procedure must be performed:

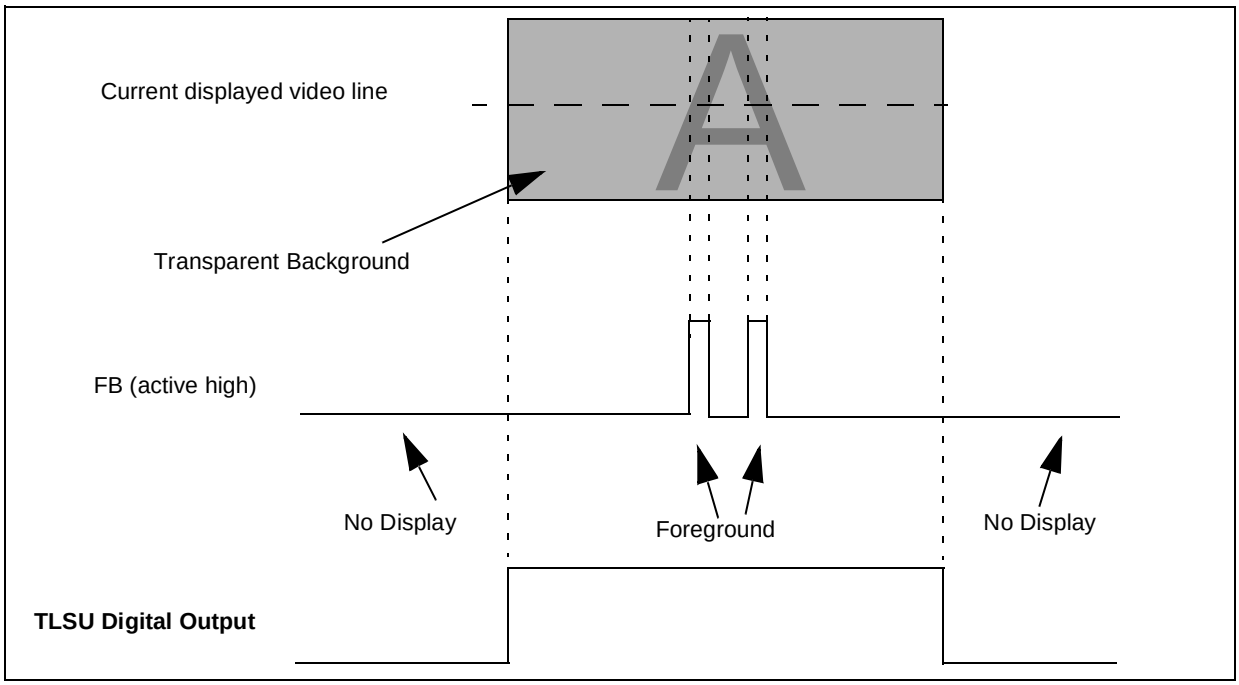
– Fast Blanking must be set to Digital mode. Set bit DIFB (bit 7) of register OSDBCR1 (R247, page 42) to 1. Refer to Section 8.5.9 Register Description.

– Initialize port 3.0 as a push-pull type Alternate Function output. Refer to Section 8.5.3 Functional Description.

– Set bit TSLE (bit 2) of register OSDER (R248, page 42) to 1. Refer to Section 8.5.9 Register Description.

– For the selected colors (i.e. those which will appear as transparent with contrast reduction), set bits BT[2:0] to 0. Refer to Section 8.5.6.4 Background Palettes.

Figure 56. Digital Translucency Output Pin Example



OSD CONTROLLER (Cont'd)

Figure 57. Digital Translucency Display Scheme using STV2238D in PQFP64 Package

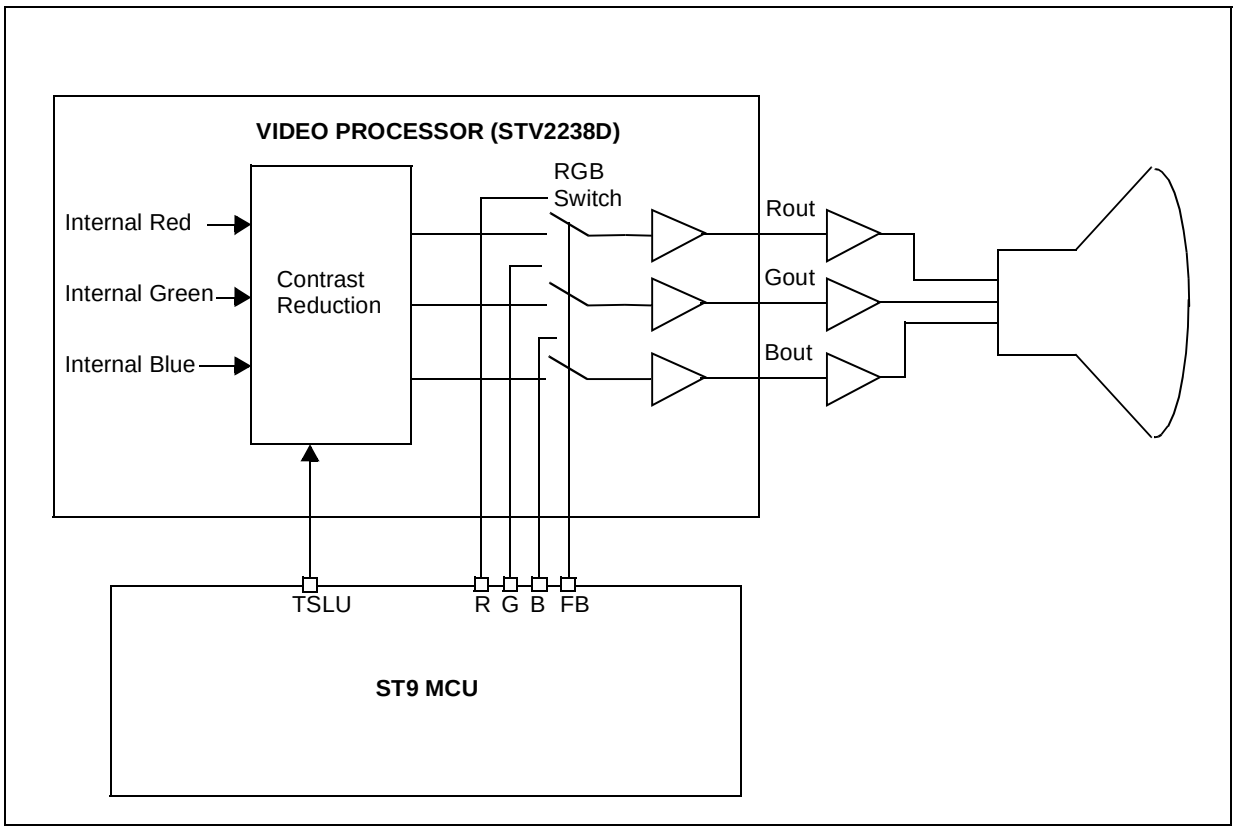
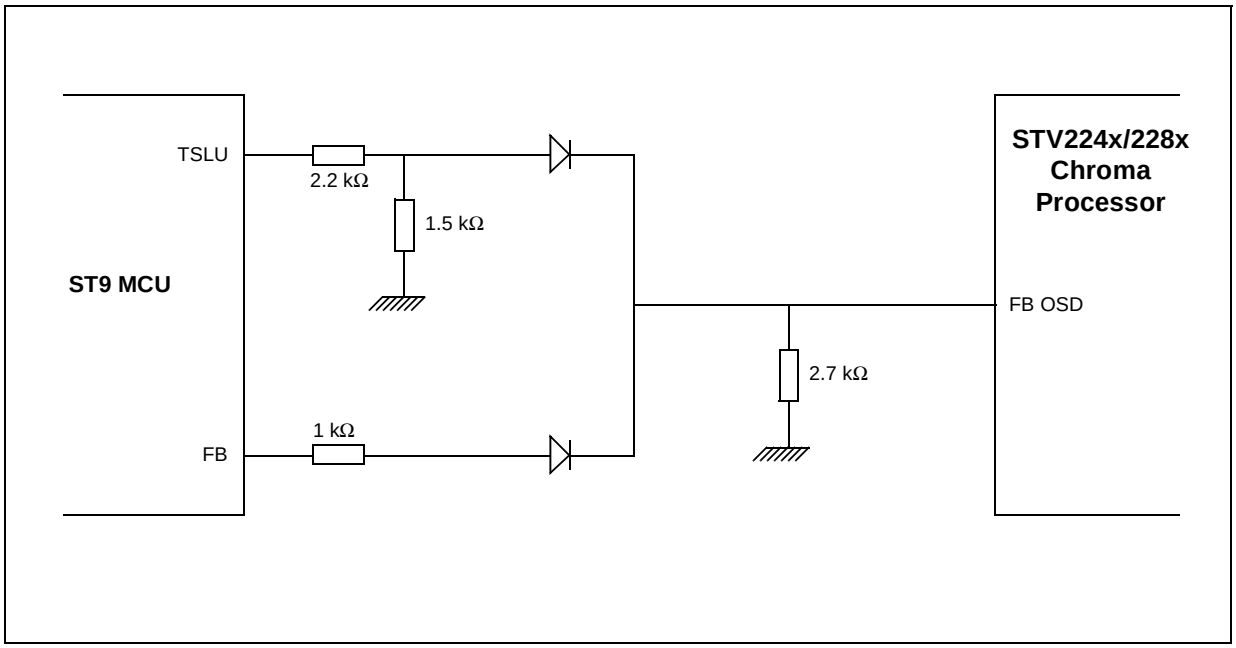


Figure 58. Application Example using STV224x/228x in SDIP56 Package



8.5.3.12 Mouse Pointer

The Mouse Pointer icon is built as an 18x26 dot matrix; it is fully user definable, selected from any of the OSD characters in Font ROM memory (except 00h). This allows a multiple mouse configuration.

The Mouse Pointer dots are processed like any 9x13 character, i.e. any dot is displayed on both fields and represents a final 2x2 pixel area on the screen. The Mouse Pointer represents a 18x26 character displayed in 2Y-2X size.

Only the Mouse Pointer foreground is displayed. The Mouse Pointer background is transparent.

The Mouse Pointer dot processing includes the standard foreground pixel processing and also the rounding and fringe. The algorithms used are the same as for characters (for more details, refer to the rounding and fringe section above, assuming a 9x13 character matrix case extended to an 18x26 matrix).

The Mouse Pointer uses one of the foreground Palette entries to define its pixel color. All features of the foreground Palette are retained except the Underline.

The priority of all color layers is from highest-to-lowest:

Mouse foreground, Mouse fringe, OSD characters and border.

8.5.4 Horizontal and Vertical Sync

8.5.4.1 Pixel Clock Control

The Pixel clock is issued from a frequency multiplier which is locked to the main crystal frequency. The synthesized frequency is software programmable (4-bit value defining the multiplying factor) which provides flexibility for supporting various application conditions, from a basic 4/3 screen format and a “1H” horizontal sweep to a 16/9 format with a “2H” sweep, interlaced or progressive scanning. For more information, refer to the Timing and Clock Controller chapter.

Note: It is recommended to wait for a stable clock (approx. 35 ms) from the frequency multiplier before enabling the OSDRAM controller.

OSD CONTROLLER (Cont'd)

Vertical & Horizontal Sync Pulse Inputs

A spike filter has been implemented on the vertical Sync input. This circuit is inserted after internal polarity compensation of the VSYNC input signal (see VPOL bit of the Delay Register, OSDDR). It masks any spike on the vertical sync pulse with a duration smaller than 3µs. The leading edge of the VSYNC pin is affected by the vertical Sync pulse cleaner. The VSYNC edges are internally delayed by 3µs.

A Schmitt trigger provides noise immunity on the horizontal Sync pulse input and will add a delay between the deflection pulse and the effective count start of the OSD line processing.

8.5.4.2 Field Detection in interlaced mode

For TV sets working in interlaced mode, the Display controller has to retrieve the field information (some pixel information, like 18x26 matrix characters, rounding or fringe, is field based).

The Display is synchronized to HSYNC and VSYNC inputs. The phase relationship of these signals may be different from one chassis type to another. Therefore, in order to prevent vertical OSD jitter, some circuitry is implemented to provide a stable and secure field detection (OSD jitter may appear if the rising edge of an external vertical sync pulse coincides with that of an external horizontal sync pulse). This circuitry delays the vertical sync leading edge internally. The delay applied is software programmable through a 4-bit value (refer to bit DBLS in the OSDDR register).

The field information is then extracted by appropriate hardware logic.

8.5.4.3 Display Behaviour in 2H modes

The "2H" mode corresponds to a double scan display mode: the line frequency is doubled (to 31.5 kHz) compared to the traditional 60 Hz field, 262.5 lines per field. This mode requires doubling the pixel frequency and also adjusting some timing operations (refer to Section 8.5.4.1 and Section 8.5.4.2).

This feature is controlled by the DBLS bit in the OSDER register.

The double scan may be used in interlaced mode (100/120 Hz field frequency) or in progressive scanning ("non-interlaced" mode, 50/60 Hz field frequency).

This feature is enabled by the NIDS bit in the OSDER register.

RGB-FB Line Start Mute

The R, G, B & FB outputs are muted after each horizontal Sync pulse received on the HSYNC pin. The mute duration is controlled by software through a 3-bit value; these bits are called "LSM(2:0)" and are located in the Mute register OSDMR.

When the Display works in 1H mode (bit DBLS reset), the mute duration can be adjusted in 2µs steps from 2 to 14 µs. When the Display works in 2H mode (bit DBLS set), the mute duration can be adjusted in 1µs steps from 1 to 7 µs.

When the 3-bit mute value is "zero", the R, G, B & FB display outputs are muted during the duration of the horizontal Sync pulse received on the HSYNC pin.

For more details, refer to the DBLS bit in the OSDER and the LSM bits in the OSDMR register.

The HSY bit in the OSDFBR register provides an image of the mute.

OSD CONTROLLER (Cont'd)

8.5.5 Programming the Display

The row-wise RAM buffer contains the description of the characters to display:

– Row and character attributes (color, shape etc.)

– Horizontal shift code

– Character codes (addressing the Font ROM)

While one row buffer is displayed on the screen, the CPU has time to prepare the content of the next character row by filling up the second row buffer. At the time the next row must be displayed, the Display controller will point to the second row buffer, allowing the CPU to start loading data into the first row buffer for the following row. An interrupt request is generated each time the buffer pointer toggles.

Note: The display and the mouse share the same interrupt line. In configurations with mouse, the DINT and MOINT bits in the OSDFBR register can be used to determine the interrupt cause.

The Vertical location of the next character row on the screen is programmed by software through the Event Line value (Refer to Figure 65). The vertical position of the beam is memorized by the Line counter which counts the TV horizontal synchronization pulses (called here "Scan Line"). When the Scan Line counter matches the Event Line value the buffer toggle mechanism is activated.

8.5.5.1 OSDRAM Mapping

The OSDRAM is mapped in segment 22h of the memory space.

In addition to row buffers, it is used to store color palettes and information concerning mouse pointer.

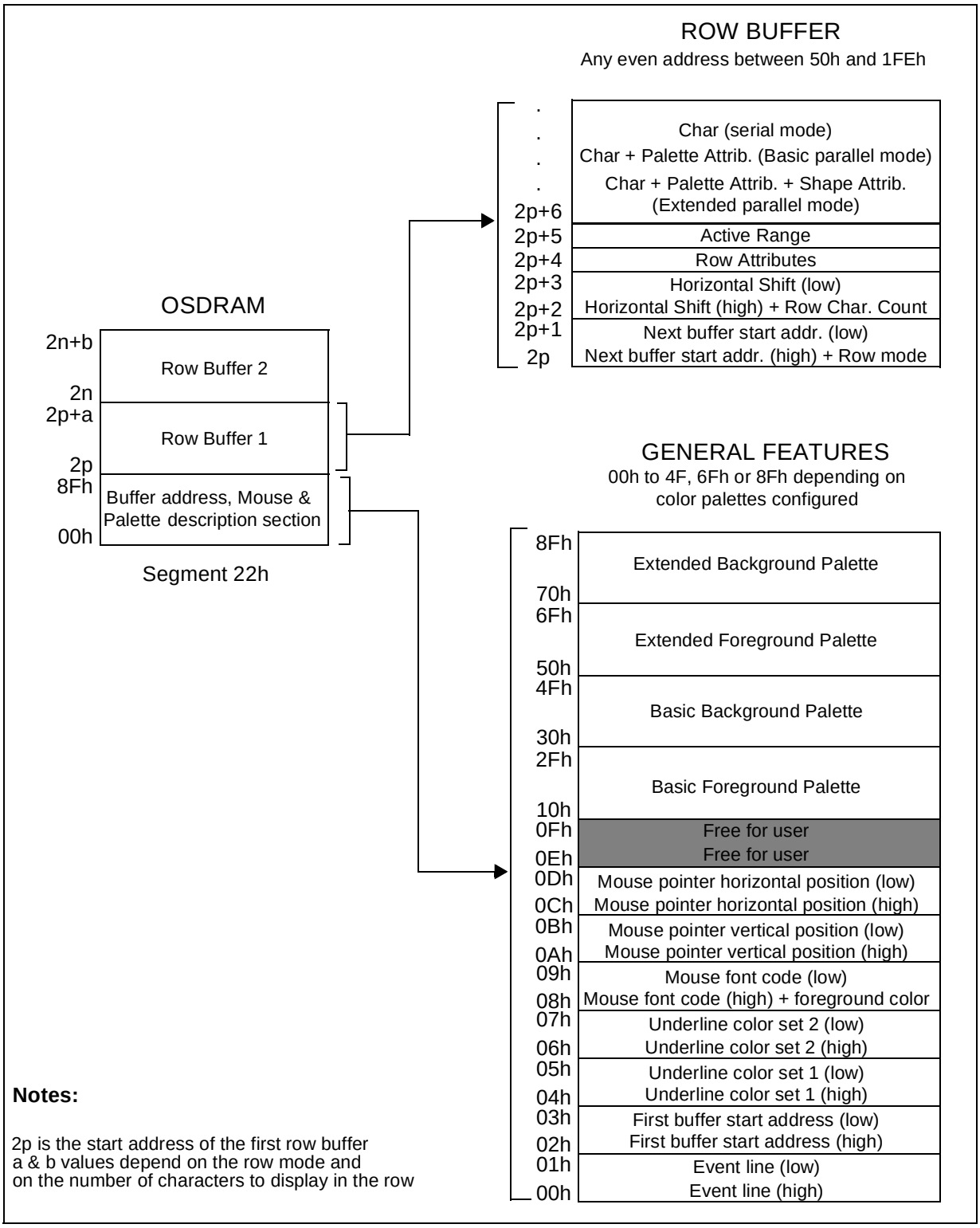
Note: The reset value of the OSDRAM contents is undefined.

General Overview

The Figure 59 gives a general overview of the OSDRAM mapping.

OSD CONTROLLER (Cont'd)

Figure 59. OSDRAM Mapping



OSD CONTROLLER (Cont'd)

8.5.5.2 Row Buffer Description

The start address for each buffer must be even.

Starting from address 2p+6, write a string corresponding to the character codes and the possible attributes as in the example below:

Byte	Serial Mode	Basic Parallel Mode	Extended Parallel Mode
1	Char or Attrib1	Char1	Char1
2	Char or Attrib2	PaletteAttrib1	PaletteAttrib1
3	Char or Attrib3	Char2	ShapeAttrib1
4	Char or Attrib4	PaletteAttrib2	Char2
5	Char or Attrib5	Char3	PaletteAttrib2
6	Char or Attrib6	PaletteAttrib3	ShapeAttrib2
...	...	...	...

8.5.5.3 OSDRAM Mapping Example 1

The left column of Figure 60 gives an example of OSDRAM mapping for the following configuration:

– The display uses basic parallel mode

– Only the two basic color palettes are used

– 36 characters are displayed per row

– The First Buffer Start Address (to be stored in 0002h and 0003h) is 0050h

– The Next Buffer Start Address to be stored in buffer 1 (address 0050h and 0051h) is 409Eh (009Eh is the address of buffer 2, and 40h is the code for basic parallel mode. Refer to Section 8.5.8.1).

– The Next Buffer Start Address to be stored in buffer 2 (address 009Eh and 009Fh) is 4050h (0050h is the address of buffer 1, and 40h is the code for basic parallel mode. Refer to Section 8.5.8.1 for more details).

8.5.5.4 OSDRAM Mapping Example 2

The right column of Figure 60 gives an example of OSDRAM mapping for the following configuration:

– The display uses extended parallel mode

– The four color palettes are used

– 36 characters are displayed per row

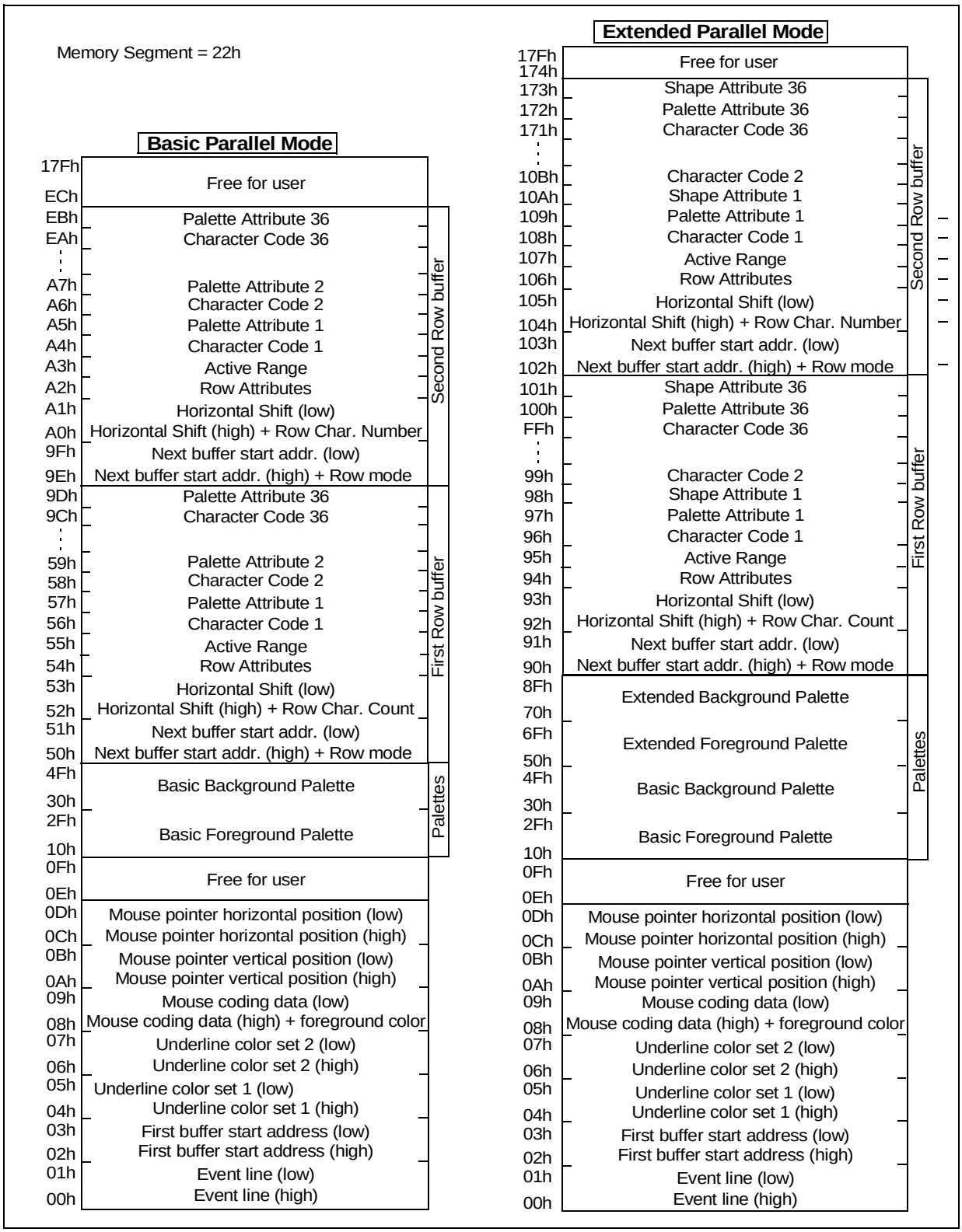
– The First Buffer Start Address (to be stored in 0002h and 0003h) is 0090h

– The Next Buffer Start Address to be stored in buffer 1 (address 0090h and 0091h) is 8102h (0102h is the address of buffer 2, and 80h is the code for extended parallel mode. Refer to Section 8.5.8.1 for more details).

– The Next Buffer Start Address to be stored in buffer 2 (address 0102h and 0103h) is 8090h (0090h is the address of buffer 1, and 80h is the code for extended parallel mode. Refer to Section 8.5.8.1 for more details).

Note: To keep some OSDRAM locations free, configure only those features that you really use (color palettes, underline palettes and mouse pointer data), as shown in the two examples.

Figure 60. Parallel Mode Mapping Examples



OSD CONTROLLER (Cont'd)

8.5.5.5 Font ROM

To address the characters in Font ROM refer to Figure 61. To obtain the character code, add the line code to the column code.

Example 1: The code for the 'A' character is:

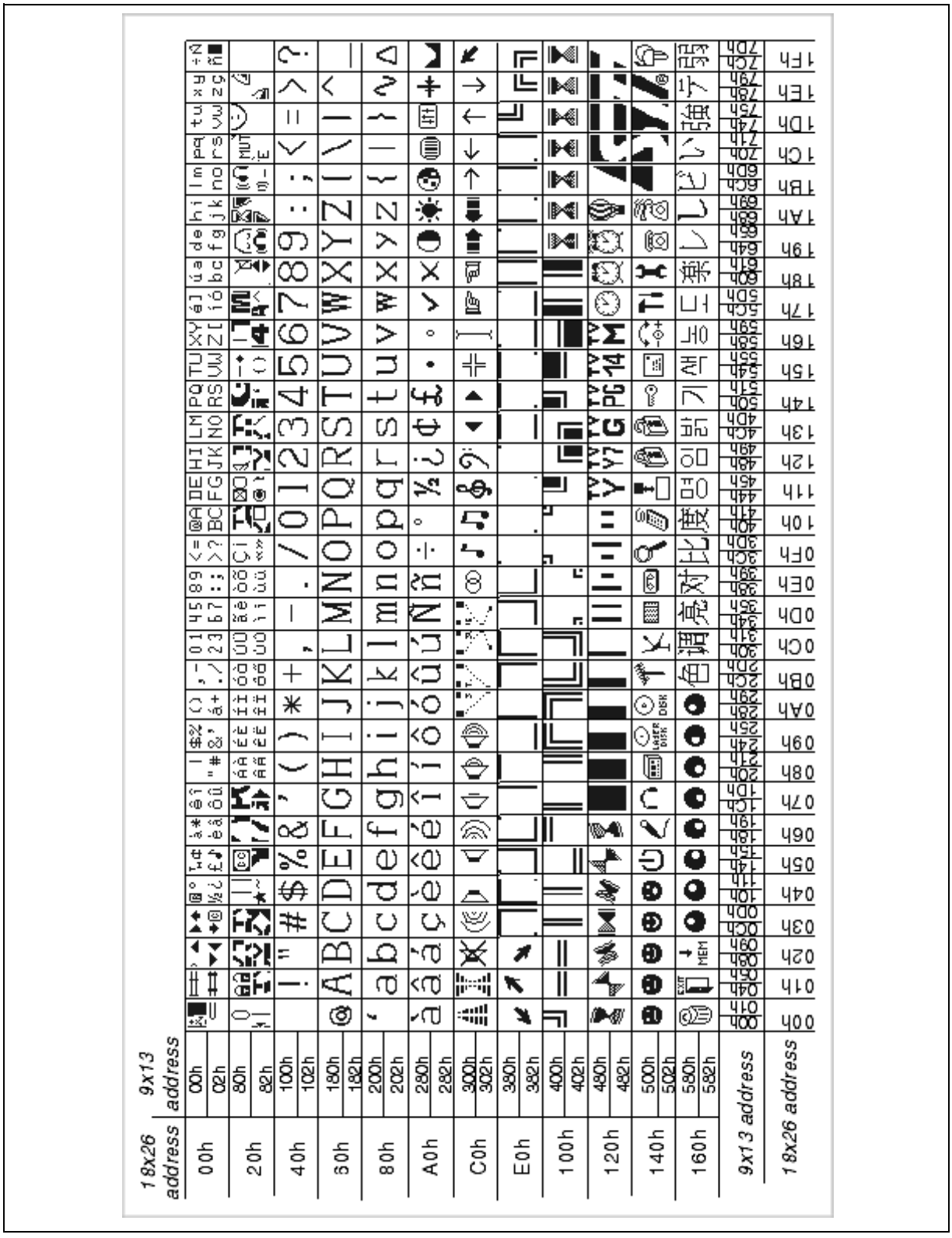
Matrix	Line Code	+	Column Code	=	Character Code
9x13	00h		41h		41h
18x26	60h		01h		61h

Example 2: The code for the '{' character is:

Matrix	Line Code	+	Column Code	=	Character Code
9x13	82h		54h		D6h
18x26	80h		1Bh		9Bh

Note: The first two 9x13 characters (addresses 00h and 01h) are the control characters. They cannot be modified by the user.

Figure 61. ST92196A Font ROM Contents



OSD CONTROLLER (Cont'd)

8.5.5.6 Event Line

Address in Segment 22h: 00h (Bits 15:8), 01h (Bits 7:0)

15							8	7							0
-	-	-	-	-	-	-	EL8	EL7	EL6	EL5	EL4	EL3	EL2	EL1	EL0

EL[8:0] is a 9-bit number specifying at which TV line number the character row display should start.

For more details refer to Section 8.5.8.8

Bits 15:8: are at address 00h in Segment 22h.

Bits 7:0 are at address 01h

Bits 15:9 are reserved.

8.5.5.7 First Buffer Start Address

Address in Segment 22h: 02h (Bits 15:8), 03h (Bits 7:0)

15							8	7							0
-	-	-	-	-	-	-	FSA8	FSA7	FSA6	FSA5	FSA4	FSA3	FSA2	FSA1	0

To handle the display properly, the user must store the start address of the first OSDRAM row buffer (the row buffer containing the first row to be displayed when the Display controller is turned on). Two bytes are reserved for this in the OSDRAM. (See Figure 59).

Bits 15:8 are at address 02h in Segment 22h

Bits 7:0 are at address 03h

Bits 15:9 are reserved.

As row buffer start addresses are always even addresses, Bit 0 is forced to 0 by hardware.

OSD CONTROLLER (Cont'd)

8.5.6 Programming the Color Palettes

The Palette attributes are coded inside the two Palettes (Basic background and foreground, and Extended background and foreground); they are therefore accessible in all modes, parallel as well as serial.

A palette contains 16 colors each defined with a 16-bit word. For each color, you can define the red level (1 of 8 values), the green level (1 value among 8), the blue level (1 value among 8), and the translucency level (1 value among 8).

The color palettes also bring improvements in underline control to allow for "Windows-like" buttons.

Once programmed, color palettes can be changed in real-time when the OSD is running, that is to say when the software is filling one row buffer while the other one is displayed (but take care that the row currently displayed may be using some of the colors which you want to modify).

OSD CONTROLLER (Cont'd)

8.5.6.1 Underline Color Set 1 (USC1)

Address in Segment 22h: 04h (Bits 15:8), 05h (Bits 7:0)

15							8	7							0
				U1T2	U1T1	U1T0	U1R2	U1R1	U1R0	U1G2	U1G1	U1G0	U1B2	U1B1	U1B0

To support "windows-like" button effects, the color of the 2 (or 4 in 18x26 matrix) bottom dot lines of a character row may be defined using the Underline attributes. Two dedicated 2-byte words define 2 color sets, "Underline color set 1" called "UCS1" and "Underline color set 2" called "UCS2". They are used by the Underline mode in addition to the current foreground color.

This provides a four color choice for both rows 12 and 13 (9x13 character matrix) or pair of rows 23-24 and 25-26 (18x26 character matrix): none (background), foreground, UCS1 or UCS2, as shown in Table 24.

The UCS1 data is mapped in a fixed OSDRAM location (See Figure 59).

Bits 15:12 = Free for the user

Bits 11:9 = **U1T[2:0]** *Underline color set 1 Translucency*
These bits configure the background translucency level applied to the color (refer to Section 8.5.3.11 for more details).
U1T[2:0] = 7 means that this color will be fully opaque (no video mixed in it on the display)
U1T[2:0] = 0 means that this color will be fully transparent (the video is displayed instead of this color)

Bits 8:6 = **U1R[2:0]** *Underline color set 1 Red color*
These bits configure the background red intensity for the color.
U1R[2:0] = 0 means that no red is used to define the color.
U1R[2:0] = 7 means that the maximum red intensity is used in the color.

Bits 5:3 = **U1G[2:0]** *Underline color set 1 Green color*
These bits configure the background green intensity for the color.
U1G[2:0] = 0 means that no green is used to define the color.
U1G[2:0] = 7 means that the maximum green intensity is used in the color.

Bits 2:0 = **U1B[2:0]** *Underline color set 1 Blue color*
These bits configure the background blue intensity for the color.
U1B[2:0] = 0 means that no blue is used to define the color.
U1B[2:0] = 7 means that the maximum blue intensity is used in the color.

OSD CONTROLLER (Cont'd)

8.5.6.2 Underline Color Set 2 (UCS2)

Address in Segment 22h: 06h (Bits 15:8), 07h (Bits 7:0)

15								8	7						0
-	-	-	-	U2T2	U2T1	U2T0	U2R2	U2R1	U2R0	U2G2	U2G1	U2G0	U2B2	U2B1	U2B0

Bits 15:12 = Free for the user

Bits 11:9 = **U2T[2:0]** *Underline color set 1 Translucency*
These bits configure the background translucency level applied to the color (refer to Section 8.5.3.11 for more details).
U2T[2:0] = 7 means that this color will be fully opaque (no video mixed in it on the display)
U2T[2:0] = 0 means that this color will be fully transparent (the video is displayed instead of this color)

Bits 8:6 = **U2R[2:0]** *Underline color set 1 Red color*
These bits configure the background red intensity for the color.
U2R[2:0] = 0 means that no red is used to define the color.
U2R[2:0] = 7 means that the maximum red intensity is used in the color.

Bits 5:3 = **U2G[2:0]** *Underline color set 1 Green color*
These bits configure the background green intensity for the color.
U2G[2:0] = 0 means that no green is used to define the color.
U2G[2:0] = 7 means that the maximum green intensity is used in the color.

Bits 2:0 = **U2B[2:0]** *Underline color set 1 Blue color*
These bits configure the background blue intensity for the color.
U2B[2:0] = 0 means that no blue is used to define the color.
U2B[2:0] = 7 means that the maximum blue intensity is used in the color.

Warning: The UCS1 and UCS2 data may be used as "row attributes" providing more than 3 underline colors on screen. This requires taking some care when their contents are modified.

The UCS1 and UCS2 contents are fetched for display only when dot lines 12/13 of the character are being processed, but at that time (while dot lines 12 and 13 are processed) any change to UCS1 or UCS2 is forbidden.

Software should change the UCS1 or UCS2 while the display processes character dot lines 1 to 11. It is recommended to associate the UCS1 or UCS2 management of the currently displayed buffer with the routine which handles the next buffer preparation. Refer to Section 8.5.8.9.

OSD CONTROLLER (Cont'd)

8.5.6.3 Foreground Palettes

15							8	7							0
	U2	U1	U0	FT2	FT1	FT0	FR2	FR1	FR0	FG2	FG1	FG0	FB2	FB1	FB0

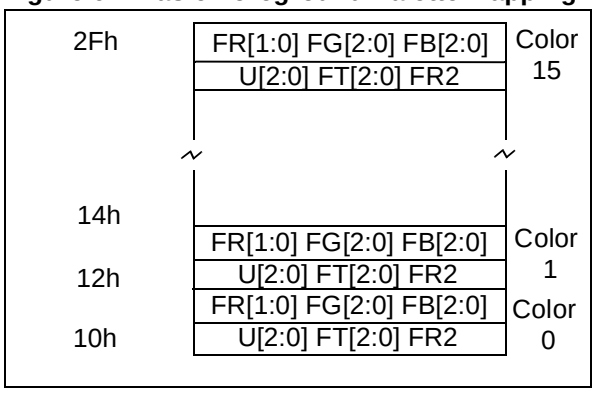
The foreground palettes (Basic and Extended) both use the same principle:

– The basic foreground palette is stored in OSDRAM (segment 22h) starting from 10h to 2Fh (see Figure 59).

– The extended foreground palette is stored in OSDRAM (segment 22h) starting from 50h to 6Fh (see Figure 59).

A 16-bit word is used to define each color in the palette, located at even addresses between 10h and 2Eh (even value) for the basic foreground palette, and between 50h and 6Eh (even value) for the extended foreground one.

Figure 62. Basic Foreground Palette Mapping



Bits 15 = Free for the user

Bits 14:12 = **U[2:0]** *Underline mode control bits*.
These bits configure the underline mode for the dot line 12 (lines 23 and 24 if using the 18x26 matrix) and line 13 (lines 25 and 26 if using the 18x26 matrix).
See Table 24.

Bits 11:9 = **FT[2:0]** *Foreground Translucency*
These bits configure the foreground translucency level applied to the color (refer to Section 8.5.3.11 for more details).
FT[2:0] = 7 means that this color will be fully opaque (no video mixed in it on the display)
FT[2:0] = 0 means that this color will be fully transparent (the video is displayed instead of this color)

Bits 8:6 = **FR[2:0]** *Foreground Red color*
These bits configure the foreground red intensity for the color.
FR[2:0] = 0 means that no red is used to define the color.
FR[2:0] = 7 means that the maximum red intensity is used in the color.

Bits 5:3 = **FG[2:0]** *Foreground Green color*
These bits configure the foreground green intensity for the color.
FG[2:0] = 0 means that no green is used to define the color.
FG[2:0] = 7 means that the maximum green intensity is used in the color.

Bits 2:0 = **FB[2:0]** *Foreground Blue color*
These bits configure the foreground blue intensity for the color.
FB[2:0] = 0 means that no blue is used to define the color.
FB[2:0] = 7 means that the maximum blue intensity is used in the color.

OSD CONTROLLER (Cont'd)

Table 24. Underline Mode Control Bits Description

U2	U1	U0	Underline color for a 9x13 dot matrix	Underline color for a 18x26 dot matrix
0	0	0	No underline	No underline
0	0	1	Line 12: foreground color	Lines 23-24: foreground color
0	1	0	Line 13: underline color set 1	Lines 25-26: underline color set 1
0	1	1	Line 13: underline color set 2	Lines 25-26: underline color set 2
1	0	0	Line 12-13: underline color set 1	Lines 23-24-25-26: underline color set 1
1	0	1	Line 12-13: underline color set 2	Lines 23-24-25-26: underline color set 2
1	1	0	Line 12: underline color set 1; line 13: underline color set 2	Lines 23-24: underline color set 1; lines 25-26: underline color set 2
1	1	1	Line 12: underline color set 2; line 13: underline color set 1	Lines 23-24: underline color set 2; lines 25-26: underline color set 1

Note: Take care when changing (or stopping) underline mode for a character, when the background color is displayed with "underline mode change in the center of the character" (M bit in Background palette).

In this case, the underline doesn't stop at the end of the character but stops in the middle of the following character. The underline color used is the last background color displayed in the current pixel line.

This always happens, when changing underline mode in the following ways:

– From THICK to THIN or no underline

– From one THIN color to another when the underlining is not on the same line.

(THICK underline, in a 9 x 13 matrix, is underlining on 2 lines and, in a 18 x 26 matrix, on 4 lines. THIN underline, in a 9 x 13 matrix, is underlining on 1 line and, in a 18 x 26 matrix, on 2 lines.)

OSD CONTROLLER (Cont'd)

8.5.6.4 Background Palettes

15							8	7							0
M				BT2	BT1	BT0	BR2	BR1	BR0	BG2	BG1	BG0	BB2	BB1	BB0

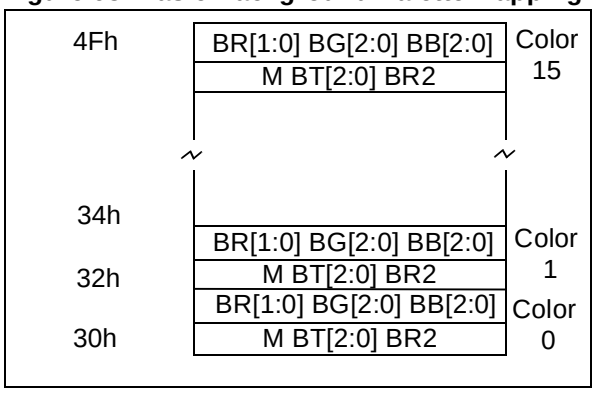
The background palettes (Basic and Extended) both use the same principle:

The basic background palette is stored in OSDRAM (segment 22h) starting from 30h to 4Fh (see Figure 59).

The extended background palette is stored in OSDRAM (segment 22h) starting from 70h to 8Fh (see Figure 59).

A 16-bit word is used to define each color in the palette, located at even addresses between 30h and 4Eh (even value) for the basic background palette, and between 70h and 8Eh (even value) for the extended background one.

Figure 63. Basic Background Palette Mapping



Bit 15 = **M** *Background color change bit.*
This bit determines where the background color change occurs.
0: The background color change takes effect immediately.
1: The background color change occurs in the center of the character.

Notes:

– When the preceding character is slanted (italics on, only available in serial mode), a background color change only occurs in the center of the character regardless of the M bit value.

– If the M bit is used in parallel mode at the beginning of a row, it is strongly recommended to insert a space or null character before setting the M bit in order to control the first half background color.

Bits 14:12 = free for the user

Bits 11:9 = **BT[2:0]** *Background Translucency*
These bits configure the background translucency level applied to the color (refer to Section 8.5.3.11 for more details).
BT[2:0] = 7 means that this color will be fully opaque (no video mixed in it on the display)
BT[2:0] = 0 means that this color will be fully transparent (the video is displayed instead of this color)

Bits 8:6 = **BR[2:0]** *Background Red color*
These bits configure the background red intensity for the color.
BR[2:0] = 0 means that no red is used to define the color.
BR[2:0] = 7 means that the maximum red intensity is used in the color.

Bits 5:3 = **BG[2:0]** *Background Green color*
These bits configure the background green intensity for the color.
BG[2:0] = 0 means that no green is used to define the color.
BG[2:0] = 7 means that the maximum green intensity is used in the color.

Bits 2:0 = **BB[2:0]** *Background Blue color*
These bits configure the background blue intensity for the color.
BB[2:0] = 0 means that no blue is used to define the color.
BB[2:0] = 7 means that the maximum blue intensity is used in the color.

OSD CONTROLLER (Cont'd)

8.5.7 Programming the Mouse Pointer

The Mouse Pointer is programmed using two control bits stored in registers, and three 16-bit words located in OSDRAM (Figure 59).

The mouse pointer can be defined as any of the ROM Font characters. The 18x26 matrix is used to represent it.

Mouse Pointer Attributes

The three 16-bit word attributes are located in OSDRAM:

– Mouse Coding Data

– Mouse Pointer Vertical Position

– Mouse Pointer Horizontal Position

Mouse pointer interrupt control

The Mouse Pointer has a dedicated interrupt source which uses the same interrupt line as the OSD. Both interrupts (Mouse Pointer and OSD) are simply ORed at hardware level and forwarded to the CPU.

The Mouse pointer interrupt is generated as soon as the Mouse matrix processing is completed.

The Mouse Pointer interrupt generation is automatically enabled as soon as the Mouse Pointer is activated.

To help identify the actual interrupt source, a flag (MOIT) is associated with the Mouse Pointer interrupt: this flag is activated when the mouse interrupt is generated and must be cleared by software. This bit is not the image of the interrupt transient condition but it keeps trace of the interrupt event until the software erases it. This bit must be reset by writing to the Enable register. The Mouse Pointer interrupt is generated regardless of this flag value.

OSD CONTROLLER (Cont'd)

8.5.7.1 Mouse Coding Data

Address in Segment 22h: 08h (Bits 15:8), 09h (Bits 7:0)

15							8	7							0
MPFE	MPRE	MFC4	MFC3	MFC2	MFC1	MFC0	MPF8	MPF7	MPF6	MPF5	MPF4	MPF3	MPF2	MPF1	MPF0

Bit 15 = **MPFE** *Mouse Pointer Fringe Enable bit*
This bit is used to enable or disable the fringe on the mouse pointer.
0: No fringe.
1: The fringe is activated.

Bit 14 = **MPRE** *Mouse Pointer Rounding Enable bit*
This bit is used to enable or disable the rounding on the mouse pointer.
0: No rounding.
1: The rounding is activated.

Bit 13 = **MFC4** *Mouse Foreground Palette*
This bit determines which palette is used for the mouse pointer color (independently of the setting of the PASW bit in the OSDDR register).
0: The basic foreground palette is used (address range in OSDRAM: 10h - 2Fh)
1: The extended foreground palette is used (address range in OSDRAM: 50h - 6Fh)

Bits 12:9 = **MFC[3:0]** *Mouse Foreground Color code*
These bits determine the foreground color of the mouse pointer.
The MFC[3:0] value points to one of the 16 predefined entries of the foreground palette.
For example, MFC[3:0] = 0 points to 10h & 11h if MFC4 = 0 (basic foreground palette), and points to 50h & 51h if MFC4 = 1 (extended foreground palette).

Bit 8 = **MPF8** *Mouse Pointer Character code Extension*
This bit selects or deselects the character code extension.
0: Character code extension disabled. The addressed character number range is 0 - 255.
1: Character code extension enabled. The addressed character number range is 256 - 383.

Bits 7:0 = **MPF[7:0]** *Mouse Pointer Character code*
This bit selects a character in the character font. The character is displayed in 18x26.
If MPF8 = 0, the MFP[7:0] value range is 0 - 255.
If MPF8 = 1, the MFP[7:0] value range is 0 - 127.

OSD CONTROLLER (Cont'd)

8.5.7.2 Mouse Pointer Vertical Position

Address in Segment 22h: 0Ah (Bits 15:8), 0Bh (Bits 7:0)

15								8	7						0
MDS						MPY9	MPY8	MPY7	MPY6	MPY5	MPY4	MPY4	MPY2	MPY1	MPY0

Bit 15 = **MDS** *Mouse Double Size enable bit*
This bit enables or disables the double size display for the mouse pointer.
0: The mouse pointer is displayed in a 2Y-2X size.
1: The mouse pointer is displayed in a 4Y-4X size.

Bits 14:10 = free for the user

Bits 9:0 = **MPY[9:0]** *Mouse Pointer vertical position*
These bits define the mouse pointer vertical start position expressed as a "TV lines-per-field" count (refer to Figure 64). When the Display controller works in non-interlaced mode, all MPY bits are used.

When the Display controller works in interlaced mode the MPY0 bit becomes meaningless The actual Mouse Pointer starting position is given by MPY[9:1].
The minimum vertical shift is therefore a 2-pixel step, in interlaced mode, as incrementing the Mouse Pointer vertical position will act on both fields, producing a one line shift per field. In non-interlaced mode, the minimum vertical shift becomes a 1-pixel step.

Warning: In interlaced mode the MPY0 bit must be forced by software to 0.

OSD CONTROLLER (Cont'd)

8.5.7.3 Mouse Pointer Horizontal Position

15								8	7						0
					MPX10	MPX9	MPX8	MPX7	MPX6	MPX5	MPX4	MPX4	MPX2	MPX1	MPX0

Bits 15:8: are at address 0Ch in Segment 22h.

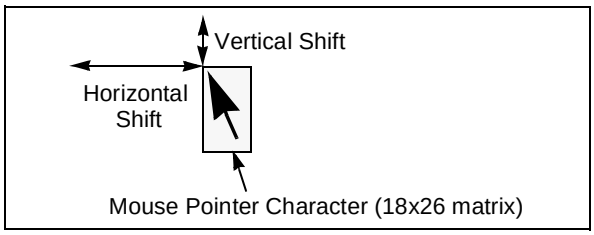
Bits 7:0 are at address 0Dh

Bits 15:11 = free for the user

Bits 10:0 = **MPX[10:0]** *Mouse Pointer horizontal position*
These bits define the mouse pointer horizontal start position value expressed as a number of pixel clock periods.

Refer to Figure 64.

Figure 64. Mouse Pointer Position



OSD CONTROLLER (Cont'd)

8.5.8 Programming the Row Buffers

The 2 row buffers are based on the same structure (Figure 59):

– Next Buffer Start Address
– Row Mode
– Horizontal Shift
– Row Character Count

8.5.8.1 Next Buffer Start Address And Row Mode

Address in Segment 22h: 2p (Bits 15:8), 2p + 1 (Bits 7:0). See Figure 59.

15							8	7							0
WM2	WM1						NSA8	NSA7	NSA6	NSA5	NSA4	NSA3	NSA2	NSA1	0

To display more than 1 character row on the screen, you must specify the start address of the next OSDRAM row buffer (the row buffer containing the next row to be displayed when the current row is completely processed). Two bytes are reserved for this in the OSDRAM. (See Figure 59).

As it is possible to display each row using different modes (serial, basic parallel, and extended parallel), the row mode has to be specified for the current row buffer.

Bits 15:14 = **WM[2:1]** *Serial/parallel Row Mode control*
These bits define the row mode for the buffer defined by the Next Buffer Start Address (NSA[8:0] bits). Refer to Table 25 for details.

Table 25. Serial/Parallel mode control

WM2	WM1	Mode
1	1	(reserved)
1	0	Extended Parallel
0	1	Basic Parallel
0	0	Serial

Bits 13:9 = Free for the user

Bits 8:0 = **NSA[8:0]** *Next buffer Start Address*
These bits define the start address of the next row buffer.
As row buffer start addresses are always even addresses, the NSA0 is not implemented, and Bit 0 is forced to 0 by hardware.

OSD CONTROLLER (Cont'd)

8.5.8.2 Horizontal Shift and Row Character Count

Address in Segment 22h: 2p + 2 (Bits 15:8), 2p + 3 (Bits 7:0). See Figure 59.

15							8	7							0
RCN5	RCN4	RCN3	RCN2	RCN1	RCN0	HS9	HS8	HS7	HS6	HS5	HS4	HS3	HS2	HS1	HS0

For each row to be displayed, the number of characters in the row, and the horizontal position of the row on the screen need to be specified.

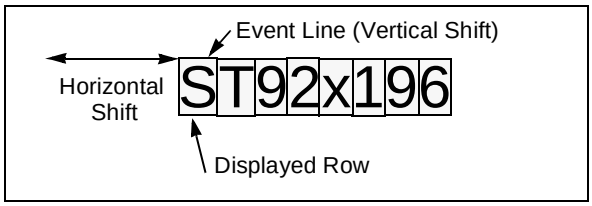
Let's assume that the start address of the current row buffer is 2p (even address).

Bits 15:10 = **RCN[5:0]** *Row Character Count*
These bits define the number of characters to display in the current row.
The Display controller allows to display from 1 to 63 characters (in parallel mode) or 61 (in serial mode, as the first 2 bytes are taken as attributes).. For example, a 36 character row requires programming RCN[5:0] = 24h.

Bits 9:0 = **HS[9:0]** *Horizontal Shift*
These bits define the horizontal shift value. They specify, in terms of number of Pixel clock periods, the horizontal shift applied from the leading edge of the Hsync pulse to the beginning of the first displayed character (1st character in parallel modes, 3rd character in serial mode). Refer to Figure 65. Loading any value smaller than 01h is forbidden.

The result is given by the formula:
Horizontal shift = [HS[9:0]+ 47] * (2*PIXCLK)
(where PIXCLK is the clock issued from the Skew Corrector). Refer to the Timing and Clock Control chapter for programming information.

Figure 65. Row Position



OSD CONTROLLER (Cont'd)

8.5.8.3 Row Attributes

Address in Segment 22h: 2p + 4. See Figure 59.

7							0
FM	UH		SY	SX	FON	ROU	FR

For each row to be displayed, specify the font matrix used (9x13 or 18x26), the size of the characters, the rounding, the fringe and the flash mode.

Let's assume that the start address of the current row buffer is 2p (even address).

Bit 7 = **FM** *Font Matrix*
This bit selects the 9x13 or the 18x26 font matrix for the current row.
The FM bit is not an address extension bit but it affects how the Font ROM content is addressed.
0: the characters use a 9x13 font matrix.
1: the characters use an 18x26 font matrix

Bit 6 = **UH** *Upper Half*
When 18x26 characters are displayed in double height (i.e. if the DBLY parallel attribute is set), the UH bit defines if the current row displays the lower or upper half of the character.
This bit has no effect when a 9x13 matrix is used or when the character has normal height (when DBLY= 0).
0: the lower half of the double-height character is displayed.
1: the upper half of the double-height character is displayed.

Bit 5 = Free for the user

Bit 4 = **SY** *Vertical Size control bit*
This bit defines the character Dot height
Refer to Table 26, Table 27, and Table 28 for complete details.

Note: The dot height is also affected by DBLY, defined in the parallel attribute section.

Bit 3 = **SX** *Horizontal Size control bit*
This bit defines the character Dot width
Refer to Table 26, Table 27, and Table 28 for complete details.

Note: The dot width is also affected by DBLX, defined in the parallel attribute section.

Table 26. 9x13 Font Matrix

SY	SX	Dot Height	Dot Width	Matrix size
0	0	2 lines	2 pixels	1Y-1X
0	1	2 lines	3 pixels	1Y-1.5X
1	0	4 lines	3 pixels	2Y-1.5X
1	1	4 lines	4 pixels	2Y-2X

Table 27. 18x26 Font Matrix (1)

SX	Character width DBLX=0	Dot width		Character width DBLX=1	Dot width
0	1X	1 pixel		2X	2 pixels
1	2X	2 pixels		4X	4 pixels

Table 28. 18x26 Font Matrix (2)

SY	Vertical Dot Size DBLY = 0	Vertical Dot Size DBLY = 1
0	1 line	2 lines
1	2 lines	4 lines

Note: In 18x26 matrix mode, this mechanism provides 7 different sizes which are: 1Y-1X, 2Y-2X, 4Y-4X but also 2Y-1X, 1Y-2X, 4Y-2X, 2Y-4X.

Note: When the Display controller works in basic parallel mode, the DBLX and DBLY bits are not accessible, and are assumed to be always programmed to "0".

Bit 2 = **FON** *Flash ON control bit*
This bit is used to control the flashing period by software. It is only available in serial and extended parallel modes. This bit has no effect in basic parallel mode.
0: the flashing mechanism is disabled for the whole row in all modes.
1: the flashing mechanism is enabled for the whole row. All characters described in the row with a serial or a parallel flash attribute are displayed as space with background color. Underline also flashes.

OSD CONTROLLER (Cont'd)

Bit 1 = **ROU** *Rounding control bit*
This bit enables or disables the rounding for the whole row.
0: the rounding is disabled
1: the rounding is enabled

Note: For a 18x26 matrix size, there is no rounding when the character size is (1X,1Y). For any other (X,Y) size combinations, the rounding is possible for the 18x26 font matrix.

Bit 0 = **FR** *Fringe control bit*
This bit enables or disables the fringe for the whole row. The fringe mechanism can be activated for any size and matrix format.
0: the character fringe is disabled
1: the character fringe is enabled.

Note: In case of fringe usage in 1Y vertical size and interlaced mode, a flicker may appear on screen as the fringe information is built on a field basis.

8.5.8.4 Active Range

Address in Segment 22h: 2p + 5. See Figure 59.

7							0
RS3	RS2	RS1	RS0	RE3	RE2	RE1	RE0

The active range feature is useful for software controlled smooth vertical scrolling (up or down).

For each row to be displayed, the first line (RS) and the last line (RE) to be displayed for the current row have to be specified.

The two values (RS[3:0] and RE[3:0]) are compared to the row line counter value. If the value of the counter is outside the active range (less than RS[3:0] or greater than or equal to RE[3:0]), the border color is displayed as defined by its attributes. Otherwise, if the counter value is inside the active range, the normal character pixel processing and display is done (See Figure 66).

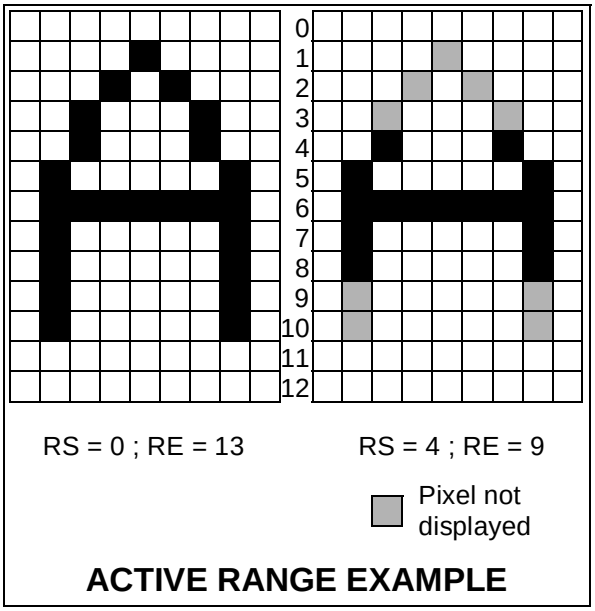
Let's assume that the start address of the current row buffer is 2p (even address).

Bits 7:4 = **RS[3:0]** *Active Range Start value*
The RS[3:0] value range is 0h-Ch (0-12) in all cases (9x13 or 18x26 character matrix).

Bits 3:0 = **RE[3:0]** *Active Range End value*
The RE[3:0] value range is 1h-Dh (1-13) in all cases (9x13 or 18x26 character matrix).

Note: For 18x26 matrix characters, the active range is calculated by pair of TV lines, i.e. the active range always starts on the first field and finishes on the second field (in interlaced mode).
In case of non-interlaced display mode, the active range is calculated by pair of TV lines.

Figure 66. Active Range Example



OSD CONTROLLER (Cont'd)

8.5.8.5 Serial Mode

In serial mode, only 1 byte is used to describe the character code or the attribute (Figure 59):

– If the most significant bit (Bit 7) of this byte is 0, the byte represents a character code.

– If the most significant bit (Bit 7) of this byte is 1, the byte represents a serial attribute. Then the display controller uses Bit 6 to determine whether the attribute is a foreground serial attribute (Bit 6 = 0) or a background serial attribute (Bit 6 = 1).

A consequence of this structure is that in serial mode, only the first 128 characters stored in the font ROM can be accessed (the value of the 7 least significant bits of the character code = the character number in font ROM).

The first two bytes of the row buffer describing the row are displayed as border color. The first character to be displayed in serial mode is in fact the 3rd of the row buffer.

All the attributes (background and foreground) are displayed as space with background color.

Let's assume that the start address of the current row buffer is 2p (even address).

In this case the character codes and attributes are stored in the OSDRAM at the address 2p+5+z, where z value is 1 to RCN (RCN is the row character count defined in Section 8.5.8.2).

Character Code in Serial Mode

~~A~~ddress in Segment 22h: 2p + 5 + z. See Figure 59.

7							0
0	CHC6	CHC5	CHC4	CHC3	CHC2	CHC1	CHC0

Bits 6:0 = **CHC[6:0]** *Character Code in serial mode*
The CHC[6:0] value points to one of the first 128 characters stored in the font ROM.

BACKGROUND SERIAL ATTRIBUTE

Address in Segment 22h: 2p + 5 + z. See Figure 59.

7							0
1	1	x	x	BP3	BP2	BP1	BP0

Bits 5:4 Reserved

Bits 3:0 = **BP[3:0]** *Background Palette pointer*
The BP[3:0] value points to one of the 16 predefined entries of the background palette for the background color and the translucency.

For example, BP[3:0] = 0 points to the first background palette entry, 30h & 31h if the palette swap bit PASW of the OSDDR register is reset (see registers description for more details).

Note: the display of the background serial attribute is affected by the use of the Italics (see the foreground serial attribute) and also by the "M" bit located in the Background Palette (see Section 8.5.6.4 for further details).

OSD CONTROLLER (Cont'd)

Foreground Serial Attribute

Address in Segment 22h: 2p + 5 + z. See Figure 59

7							0
1	0	FLA	IT	FP3	FP2	FP1	FP0

Bit 5 = **FLA** *Flash control bit*
This bit controls the flashing feature (see Section 0.2.4.2).
0: All the following characters in the row are not affected by the flashing mechanism.
1: All the following characters in the row follow the flashing mechanism.

Note: Flashing characters are alternatively displayed as spaces and in normal mode (non-flashing), depending on the value of the FON bit in the Row Attribute byte (see Section 8.5.8.3).
The flash rate is controlled by software by toggling the "FON" bit.

Bit 4 = **IT** *Italics control bit*
This bit enables the italic feature for the row. (See Section 8.5.3.6)
0: Italics are disabled.
1: All the following characters, until the end of the row, or the next foreground serial attribute are displayed in italics.

Note: Italics mode is only available in serial mode.

Bits 3:0 = **FP[3:0]** *Foreground Palette pointer*
The FP[3:0] value points to one of the 16 predefined entries of the foreground palette for foreground color, translucency and underline style of all the following characters (see Section 8.5.6.3).

For example, FP[3:0] = 0 points to the first foreground palette entry, 10h & 11h if the palette swap bit PASW of the OSDDR register is reset.

8.5.8.6 Basic Parallel Mode

In basic parallel mode, each character code (1 byte) is followed by a palette attribute (1 byte). See Figure 59.

Let's assume that the start address of the current row buffer is 2p (even address).

In this case the character codes are stored in OSDRAM at the address 2p+4+2z, and the palette attributes are stored in the OSDRAM at the address 2p+5+2z, where z ranges from 1 to RCN (RCN is the row character count defined in Section 8.5.8.2).

The character code structure allows pointing to the first 256 characters of the font ROM (the character code value = the character number in font ROM).

CHARACTER CODE IN BASIC PARALLEL MODE
Address in Segment 22h: 2p+4+2z. See Figure 59.

7							0
CHC7	CHC6	CHC5	CHC4	CHC3	CHC2	CHC1	CHC0

Bits 7:0 = **CHC[7:0]** *Character Code in basic parallel mode*
The CHC[7:0] value points to one of the first 256 characters stored in the font ROM.

PALETTE ATTRIBUTE
Address in segment 22h: 2p+5+2z. See Figure 59.

7							0
FP3	FP2	FP1	FP0	BP3	BP2	BP1	BP0

Bits 7:4 = **FP[3:0]** *Foreground Palette pointer*
The FP[3:0] value points to one of the 16 predefined entries of the foreground palette for the foreground color, the translucency and the underline style of all the following characters (see Section 8.5.6.3 for further details).

For example, FP[3:0] = 0 points to the first foreground palette entry, 10h & 11h if the palette swap bit PASW of the OSDDR register is reset (see registers description for more details).

Bits 3:0 = **BP[3:0]** *Background Palette pointer*
The BP[3:0] value points to one of the 16 predefined entries of the background palette for the background color and the translucency.

For example, BP[3:0] = 0 points to the first background palette entry, 30h & 31h if the palette swap bit PASW of the OSDDR register is reset (see registers description for more details).

Note: the background color of the character is affected by the use of the "M" bit located in the Background Palette (see Section 8.5.6.4).

OSD CONTROLLER (Cont'd)

8.5.8.7 Extended Parallel Mode

In extended parallel mode, each character code (1 byte) is followed by a palette attribute (1 byte) and a shape attribute (1 byte). Refer to Figure 59.

Let's assume that the start address of the current row buffer is 2p (even address).

The character codes are stored in the OSDRAM at the address 2p+3+3z, the palette attributes are stored at the address 2p+4+3z, and the shape attributes are stored at the address 2p+5+3z, where z range value is 1 to RCN (RCN is the row character count defined in Section 8.5.8.2).

The character code structure, using the shape attribute, allows you to point to any of the font ROM characters.

The shape attribute definition depends on the font matrix used for the row (it depends on the FM bit, see Section 8.5.8.3).

CHARACTER CODE IN EXTENDED PARALLEL MODE

Address in segment 22h: 2p+3+3z. See Figure 59.

7							0
CHC7	CHC6	CHC5	CHC4	CHC3	CHC2	CHC1	CHC0

Bits 7:0 = **CHC[7:0]** *Character Code in extended parallel mode*
The CHC[7:0] bits are used, combined with 3 or 1 bits of the shape attribute (for a 9x13 or 18x26 matrix), to point to any of the characters stored in the font ROM (refer to the shape attribute description for more details).

PALETTE ATTRIBUTE

Address in segment 22h: 2p+4+3z. See Figure 59.

7							0
FP3	FP2	FP1	FP0	BP3	BP2	BP1	BP0

Refer to Section 8.5.8.6 for the bit description.

OSD CONTROLLER (Cont'd)

SHAPE ATTRIBUTE - 9x13 MATRIX

Address in segment 22h: 2p+5+3z. See Figure 59.

7							0
CODX	SPL1	SPL0	FXP	BXP	FOC	SHA	FRC

Bit 7 = **CODX** *Character Code Extension*
This bit is used with SPL[1:0] as the character address extension. Thus it is possible to address any of the 9x13 characters of the font ROM. (See the SPL[1:0] bit description for more details)

Bits 6:5 = **SPL[1:0]** *Character Split control bits*
These "character split" bits are used with the CODX bit as the character address extension. It is then possible to address any of the 9x13 characters of the font ROM.
If the character code is ChC[7:0] (1 byte), then the character addressed with this structure is: CODX.SPL1.SPL0.ChC[7:0]

Bit 4 = **FXP** *Foreground Extended Palette addressing bit*
This bit is combined with the PASW bit in the OSDDR register to give the MSB bit of the foreground palette address. It allows the foreground palette, (basic or extended), to be selected on a per-character basis. See Table 29 for details.

Table 29. Foreground Palette Selection

PASW	FXP	Foreground palette used
0	0	basic
0	1	extended
1	0	extended
1	1	basic

Bit 3 = **BXP** *Background Extended Palette addressing bit*
This bit is combined with the PASW bit in the OSDDR register to give the MSB bit of the background palette address. It allows the background palette, (basic or extended), to be selected on a per-character basis. See Table 30 for details.

PASW	BXP	Background palette used
0	0	basic
0	1	extended
1	0	extended
1	1	basic

Bit 2 = **FOC** *Flash On Character control bit*
This bit enables the flash mechanism for the current character.
0: The flash mechanism is disabled. The current character is displayed, whatever the flash row attribute value (FON) is (see Section 8.5.8.3).
1: The flash mechanism is enabled. If the flash row attribute (FON) is "on" (see Section 8.5.8.3), the character is displayed as space using the background color.

Bit 1 = **SHA** *shadow mode control bit*
This bit enables or disables a black shadow shape on the right and bottom edges of the current character foreground.
0: No shadow is added
1: A black shadow is added on the current character.

Note: This shadow shape follows the same algorithm as the fringe (see Section 8.5.3.7 for further details).

Bit 0 = **FRC** *Fringe on Character control bit*
This bit enables or disables a fringe on the current character foreground.
0: No fringe is added.
1: A fringe is added on the current character if the fringe row attribute bit FR is set (see Section 8.5.8.3 for more details).

Note: The fringe follows the algorithm described in Section 8.5.3.7.

OSD CONTROLLER (Cont'd)

SHAPE ATTRIBUTE - 18x26 MATRIX

Address in segment 22h: 2p+5+3z. See Figure 59.

7							0
CODX	DBLY	DBLX	FXP	BXP	FOC	SHA	FRC

Bit 7 = **CODX** *Character Code Extension*
This bit is used as the character address extension. Then it is possible to address any of the 18x26 characters of the font ROM.
Let's assume that the character code is ChC[7:0] (1 byte), then the character addressed with this structure is: CODX.ChC[7:0]

Bit 6 = **DBLY** *Double height control bit*
This bit controls the double height feature applied on the current row height.
0: The character is displayed with the current row height, as defined by SY (Section 8.5.8.3).
1: The current character is displayed with a double height than defined by SY. The display of the lower or upper half of the character is controlled by the UH row attribute bit (refer to Section 8.5.8.3).

Bit 5 = **DBLX** *Double width control bit*
This bit controls the double width feature applied to the current row character width.
0: The character is displayed with the current width as defined by SX (Section 8.5.8.3).
1: The character is displayed in double width, according to the following rules:
- It covers the next character location
- The next character location is read and decoded but not processed,
- If the character is the last one in the row, it will be truncated to its left half.

Bits 4:0 = Please refer to the bit descriptions in the 9x13 matrix shape attributes.

OSD CONTROLLER (Cont'd)

8.5.8.8 Row Buffer Management

To Start the Display:

0. Write the (DION, OSDE) bits to (1,0) to access the OSDRAM with the CPU clock,
1. Initialize the color palettes,
2. Initialize the Mouse Pointer Data (if needed),
3. Initialize the "first buffer start address" with the address of the first byte of the above Row Buffer (address 0002h & 0003h of the segment 22h),
4. Fill up one of the row buffers with the data to display the desired row (only in case the TE bit in the OSDER register has been set),
5. Initialize the Event Line value to the desired one (address 0000h & 0001h of the segment 22h),
6. Set the MOPE bit (if needed),
7. Start the Display controller by programming the mode control bits (DION, OSDE) and the transfer enable bit (TE) to the desired working mode. It is mandatory to start the display following the algorithm below:

```
unsigned char tmp;

spp(OSD_PG);                    /* select the OSD register page */
OSDFBR &= ~0x06;                /* reset DINT & MOIT bits */

while (OSDFBR & OSDm_Vsy);      /* wait a Low to High transition on VSYNC */
while (!(OSDFBR & OSDm_Vsy));

tmp= OSDMR;                     /* save LSM bits */
OSDMR &= ~0x07;                 /* reset the LSM bits so that the Hsy bit will be an image of the HSYNC pulse */

OSDER = 0x40;                   /* OSDRAM interface enabled with PIXEL clock */

di();                           /* disable all interrupts */

while (OSDFBR & OSDm_Hsy);      /* wait a HSYNC pulse : Low -> High -> Low transition */
while (!(OSDFBR & OSDm_Hsy));
while (OSDFBR & OSDm_Hsy);

OSDMR = tmp;                    /* recover old LSM bit value */

spp(OSD_PG);                    /* select the OSD register page */
OSDER |= 0xE2;                  /* start the display by setting the appropriate bits,
                                   set at least OSDE, DION, and TE bits. Then set the other bits as required
                                   for your application. Here the MOPE (MOuse Pointer Enable) bit is also set */

ei();                           /* enable interrupts again*/
```

OSD CONTROLLER (Cont'd)

The real time control provides:

– A continuous search of matching values between Scan line and Event Line (this condition being evaluated at each TV line start).

– A display interrupt generation when the match condition is detected.

– In "full OSD mode", if the TE bit in the OSDER register is set, the switch from one row buffer to the second row buffer when the match occurs. If the TE bit in the OSDER register is reset, when a matching condition occurs, the previous row buffer will be kept.

8.5.8.9 Handling the Row Buffers in Continuous Mode:

– When the line match condition is detected, an interrupt is sent to the CPU.

Let us then assume the TE bit in the OSDER register is set. When the interrupt is executed:

– The Event Line value must be programmed to the next desired value.

– The next Row Buffer content must be filled up by the data of the next row to display. The next row buffer is easily identified using the BUFL bit in the OSDFBR register.

Let us then assume the TE bit in the OSDER register is reset. When the interrupt is fetched:

– The Event Line value must be programmed to the next desired value.

– The Next Row Buffer content might be filled up by the data of the next row to display, if desired.

– The content of the current Row Buffer is NOT displayed but simply ignored as it should have already been displayed in a previous cycle.

Note: In case the TE bit in the OSDER register is kept reset, there is no need to manage the second row buffer as it will never be used.

OSD CONTROLLER (Cont'd)

8.5.9 Register Description

To run the Display controller properly, you need to program the 7 registers that configure the display

BORDER COLOR REGISTER 2 (OSDBCR2)
R246 - Read/Write
Register Page: 42
Reset Value: 0x00 0000

7							0
B2BC		BOS2	BOS1	BOS0	BOR2	BOR1	BOR0

Bit 7 = **B2BC** *Background to Border Color control bit.*
This bit allows to force the background color of all the characters to the border color.
0: All characters backgrounds are normally displayed
1: All character backgrounds are forced to the current border color and translucency level.

Bit 6 = Reserved

Bits 5:3 = **BOS[2:0]** *Border color translucency*
These bits control the Border color translucency.
BOS[2:0] = 7 means that the border color will be fully opaque (no video mixed in it on the display)
BOS[2:0] = 0 means that the border color will be fully transparent (the video is displayed instead of this color)

Bits 2:0 = **BOR[2:0]** *Red Border color*
These bits configure the red intensity for the border color.
BOR[2:0] = 0 means that no red is used to define the border color.
BOR[2:0] = 7 means that the maximum red intensity is used in the border color.

BORDER COLOR REGISTER 1 (OSDBCR1)
R247 - Read/Write
Register Page: 42
Reset Value: 0x00 0000

7							0
DIFB	-	BOG2	BOG1	BOG0	BOB2	BOB1	BOB0

Bit 7 = **DIFB** *Digital FB control bit*
This bit selects the Fast Blanking (FB) output as analog or digital.
0: The FB DAC works as an 8-level DAC output from 0V up to 1V (with a 500ohms internal impedance to ground).
1: The FB DAC works as a 2-level DAC output, the high level providing an amplitude higher than 2.7 volts. All translucency control bits are managed as follows:
- the code (0,0,0) generates a "0" output (0 volt),
- all other codes generate a "1" output (>2.7 V).

Note: This applies to BT[2:0], FT[2:0], U2T[2:0], U2T[2:0] and BOS[2:0] (refer to Section 8.5.6.3, Section 8.5.6.4 and Section 8.5.6.1, and to the OSDBCR2 register).

Bit 6 = Reserved

Bits 5:3 = **BOG[2:0]** *Green Border color*
These bits configure the green intensity for the border color.
BOG[2:0] = 0 means that no green is used to define the border color.
BOG[2:0] = 7 means that the maximum green intensity is used in the border color.

Bits 2:0 = **BOB[2:0]** *Blue Border color*
These bits configure the blue intensity for the border color.
BOB[2:0] = 0 means that no blue is used to define the border color.
BOB[2:0] = 7 means that the maximum blue intensity is used in the border color.

OSD CONTROLLER (Cont'd)

ENABLE REGISTER (OSDER)
R248 - Read/Write
Register Page: 42
Reset Value: 0000 0000 (00h)

7							0
DION	OSDE	TE	DBLS	NIDS	TSLE	MOPE	FPIXC

Bit 7 = **DION** *Display ON*
This bit is used in combination with the OSDE bit to control the display working mode. See Table 31.

Warning: after a reset, a valid HSYNC signal is required to write to the OSDRAM, whatever the clock rate (CPU or Pixel clock rate).

Bit 6 = **OSDE** *OSD Enable*
This bit is used in combination with the DION bit to control the display working mode. See Table 24.

Note 1: When the (DION,OSDE) bits switch from any other value to (1,1), i.e. when the controller is switched to a full OSD function, the "first buffer start address" content is used to locate the first Row buffer to process.
While the full Display function is running, the DION & OSDE bits remain set and the first buffer start address is not used again, even if both bits are rewritten to "1".

Note 2: It is strongly recommended to use state 3 **only** if the OSDRAM has been initialized using state 2.

Warning 1: States 3 and 4 (refer to Table 31) can only be used if HSYNC and VSYNC are applied on the external pins.

Warning 2: After a reset, a valid HSYNC signal is required to write to the OSDRAM, regardless of the clock rate (CPU or Pixel clock rate).

Warning 3: When the OSD is displayed, it is advised not to write to the OSDRAM when a VSYNC pulse occurs. In Normal operating mode, this configration will never happen.

Bit 5 = **TE** *Transfer Enable bit*
This bit controls the "swap to next row buffer" function whenever the Scan Line counter content matches the Event Line parameter value.
An interrupt request pulse is generated and forwarded to the core each time the match occurs regardless of the value of TE.
0: Row buffer swap disabled. The current row buffer content is simply ignored and the screen will display the border color, as if the current buffer content was already processed.
1: A Row buffer swap enabled

Note: Refer to Section 8.5.8.8 for more details about using the TE bit.

Bit 4 = **DBLS** *Double Scan bit*
This bit defines if the display works in 1H or 2H mode.
0: The display works in "single scan" or "1H" mode.
1: The display works in "double scan" or "2H" mode. The 2H mode is used in progressive scan display (60Hz field, 525 lines).

Note: the DBLS bit acts on the display vertical delay for field determination (refer to the VD[3:0] bits of the Delay register OSDDR).
The DBLS bit also acts on the Line Start Mute (refer to the LSM[2:0] bits of the Mute register OSDMR) and the HSY flag bit.

Table 31. OSDRAM Interface Configuration

DION	OSDE	OSDRAM Interface clock	OSD Function	Detailed Configuration	State
0	0	off, no RAM access	off	The OSDRAM controller and the Display are both disabled. The CPU has no access to the OSDRAM.	1
1	0	on, CPU clock	off	The Display is disabled. The OSDRAM controller is running using the CPU clock, allowing for CPU accesses.	2

0	1	on, Pixel clock	no OSD, time control on	The Display is partially enabled. The OSDRAM controller is running with the Pixel clock, allowing CPU accesses. The OSD pixel processing is disabled (RGB & FB outputs are turned off), the TV oriented time control is still running, such as event line control, interrupt generation, flag bits and field calculation. The border control is inactive.	3
1	1	on, Pixel clock	fully on	The OSDRAM controller and the Display are both enabled. The OSDRAM controller is running with the Pixel clock, allowing CPU accesses. The RGB & FB outputs are turned on. The border control is activated.	4

OSD CONTROLLER (Cont'd)

Bit 3 = **NIDS** *Non Interlaced Display control bit*
This bit selects the interlaced or non-interlaced mode.
0: The display works in interlaced mode (line counting, fringe and rounding algorithms are 2-field based)
1: The display works in non-interlaced mode (line counting, fringe and rounding algorithms are 1-field based).

Bit 2 = **TSLE** *Translucency Enable bit*
This bit enables or disables the digital translucency signal (TSLU) generation. (Refer to the Section 8.5.3.11).
0: The TSLU signal built by the Display controller remains continuously idle regardless of the OSD activity.
1: The TSLU signal carries the real time background information and can be output through the I/O pin alternate function.

Bit 1 = **MOPE** *Mouse Pointer Enable*.
This bit enables or disables the Mouse Pointer.
0: The Mouse Pointer is disabled
1: The Mouse Pointer is enabled.

Note: When the Mouse Pointer feature is not implemented, this bit becomes "reserved".
The MOPE bit also acts as a Mouse Pointer interrupt Enable: enabling the Mouse pointer allows automatically enables the Mouse Pointer interrupt generation.

Bit 0 = **FPIXC** *Fast Pixel Clock control bit*
This bit handles the divide-by-2 prescaler inserted between the Skew Corrector output and the Display Pixel clock input.
0: The Skew corrector clock output is divided by 2 to provide the Pixel clock.
1: The Skew corrector clock output is directly taken as the Pixel Clock.

Note: For further information, refer to the Timing and Clock Control chapter.

OSD CONTROLLER (Cont'd)

DELAY REGISTER (OSDDR)
R249 - Read/Write
Register Page: 42
Reset Value: 0xxx xxxx

7							0
PASW	HPOL	VPOL	FBPOL	VD3	VD2	VD1	VD0

Note: The display may flicker if you write to the Delay Register while OSD is fully on.

Bit 7 = **PASW** *Palette Swap bit*
The PASW bit is used in serial or basic parallel modes to provide access to the extended palettes. It is still active when you work in extended parallel mode, however it not needed as the FXP and BXP bits are available (refer to Section 8.5.8.7).
0: The basic palette sets are used.
1: The extended palette sets are used.

Bit 6 = **HPOL** *Hsync signal Polarity selection bit*
This bit has to be configured according to the polarity of the Hsync input signal.
0: Hsync input pulses are of positive polarity
1: Hsync input pulses are of negative polarity

Bit 5 = **VPOL** *Vsync signal POLarity selection bit*
This bit has to be configured according to the polarity of the Vsync input signal.
0: Vsync input pulses are of positive polarity
1: Vsync input pulses are of negative polarity

Bit 4 = **FBPOL** *Fast Blanking signal Polarity selection bit*
This bit selects the polarity of the Fast Blanking (FB) output signal.
0: FB output pulses are of positive polarity (FB active high)
1: FB output pulses are of negative polarity (FB active low)

Note: the FB signal is kept active during the VSYNC vertical retrace.

Bits 3:0 = **VD[3:0]** *Vertical Delay control bits*
This 4-bit value is used to program an internal delay on vertical sync pulses applied to VSYNC input pin.

The purpose of the programmable delay is to prevent vertical OSD jitter in case the rising edge of external vertical sync pulse coincides with thatofan external horizontal sync pulse.
The delay applied is expressed by the following equations (4 MHz is the frequency issued directly from the crystal oscillator):

for 2H display mode:
[VD[3:0]+1] * 8*(1/4MHz) =< d =< [VD[3:0]+2] * 8*(1/4MHz)

for 1H display mode:
[VD[3:0]+1] *16*(1/4MHz) =< d =< [VD[3:0]+2] *16*(1/4MHz)

Note: programming the Vertical Delay to Fh will freeze the scan line counter disabling any further RGB output.

Note: It is mandatory for the CPU to initialize the Vertical Delay Register to avoid any problems.

OSD CONTROLLER (Cont'd)

FLAG BIT REGISTER (OSDFBR)
R250 - Read/Write
Register Page: 42
Reset Value: xxxx xxxx (xxh)

7							0
BUFL	VSY	HSY	VSDL	FIELD	DINT	MOIT	SL8

Bit 7 = **BUFL** *Buffer Flag bit*
This bit indicates which Row Buffer of the OSD RAM is being used by the Display.
The BUFL flag is automatically re-evaluated each time the Scan & Event line matching condition is fulfilled. In case the "TE" bit is reset (see the OSDER register), the "BUFL" flag remains unchanged as NO row buffer change occurs.
0: The OSD displays the content of the second row buffer (the one NOT pointed by the "first buffer start address" value).
1: The OSD displays the content of the Row Buffer location pointed by the "first buffer start address" value.

Note: The BUFL flag is automatically reset when the (DION,OSDE) bits are switching from any other value to (1,1); it will be set at the first Buffer transfer (scan & event match and TE=1).

Bit 6 = **VSY** *Vsync status bit*
This bit gives the status of the VSYNC input signal.
0: the VSYNC input signal is inactive.
1: the VSYNC input signal is active.

Note: The VSYNC signal polarity is compensated to always provide VSY=1 during the vertical pulse.

Bit 5 = **HSY** *Hsync status bit*
This bit gives the status of the internal HS signal generated by the skew corrector and locked to the external HSYNC signal.
0: The HS signal is inactive.
1: The HS signal is active.

Note: the HSYNC signal polarity is compensated to provide HSY=1 during the horizontal pulse.

Note: HSY remains active during the whole "Line Start Mute" timing which is software controlled through both the DBLS bit and the LSM[2:0] value (see OSDER and OSDMR registers).

Bit 4 = **VSDL** *Delayed Vertical Pulse status bit*
This bit indicates the status of the VDPLS internal signal (it is the delayed vertical pulse issued from the programmable vertical delay unit as described by the OSDDR register bits VD[3:0])
0: The VDPLS internal signal is inactive
1: The VDPLS internal signal is active

Note: the VDPLS signal polarity is compensated to provide VSDL =1 during the vertical pulse.

Bit 3 = **FIELD** *Field status bit*
This bit indicates the current TV field.
0: TV beam is in field 2 (even field)
1: TV beam is in field 1 (odd field)

Bit 2 = **DINT** *Display Interrupt flag bit*
This bit is set by hardware when an OSD interrupt occurs. This bit must be reset by Software. Refer to Table 32.

Table 32. Display and Mouse Interrupt Flags

DINT	MOIT	Meaning
0	0	No Interrupt
0	1	Mouse Interrupt
1	0	Display Interrupt
1	1	Mouse and Display Interrupts

Note: To handle OSD interrupts, it is recommended to clear the pending bit associated with the external interrupt channel used for the OSD (see the chapter on Interrupts) and then poll the two flag bits (DINT & MOIT) in series within the Display interrupt routine.

Bit 1 = **MOIT** *Mouse pointer Interrupt flag*.
This bit indicates which is the source of the OSD interrupt (See Table 32). This bit must be reset by Software.

Bit 0 = **SL8** *Most Significant Bit of the Scan Line counter*
Refer to the description of the OSDSLR register.

OSD CONTROLLER (Cont'd)

SCAN LINE REGISTER (OSDSLR)
R251 - Read Only
Register Page: 42
Reset Value: xxxx xxxx (xxh)

7							0
SL7	SL6	SL5	SL4	SL3	SL2	SL1	SL0

Bits 7:0 = **SL[7:0]** *Scan Line Counter Value*
These bits indicate the current vertical position of the TV beam.
The most significant bit SL8 of this counter is located in the Flag Bit register OSDFBR (see below).
This counter starts from 0 at the top of the screen (i.e. after the Vsync pulse) and is incremented by HSYNC.

MUTE REGISTER (OSDMR)
R252 - Read/Write
Register Page: 42
Reset Value: 00xx x000

7							0
ADMULT	ODEVN	-	-	-	LSM2	LSM1	LSM0

Bit 7 = **ADMULT** *Address Multiply control bit*
This bit, together with the ODEVN bit, controls the, OSDRAM address generation. (Refer to the note below for more details)
0: The CPU address is used to address the OSDRAM, i.e. the CPU address LSB bit is used as OSDRAM address LSB.
1: The CPU address is multiplied by 2 and the ODEVN control bit is used as the OSDRAM LSB address bit.

Note: This mechanism is intended for improved software compatibility with the ST9296 when the Display works in basic parallel mode; it allows to do a buffer transfer to the OSDRAM using a basic "ld memory-to-memory with post-increment" instruction; i.e.: ld(..)+, (")+ to fill up the row buffer, making an automatic reservation for the attribute byte associated with the character.

Bit 6 = **ODEVN** *Odd/Even address control bit*
This bit controls the OSDRAM address LSB bit when the ADMULT bit is high.
Refer to the ADMULT bit description for further details.

Bits 5:3 are reserved

Bits 2:0 = **LSM[2:0]** *Line Start Mute value*
These bits are used to program the mute duration after the beginning of each TV line.
When the Display works in 1H mode the mute duration can be adjusted in 2µs steps from 2 to 14 µs.
When the Display works in 2H mode the mute duration can be adjusted in 1µs steps from 1 to 7 µs.
The LSM bits also define the HSY flag duration.
The Mute duration is expressed by the following equation (the "1µs" is a frequency issued from the crystal oscillator):
for 2H display mode: T_{mute} = LSM[2:0] * (1 µs)
for 1H display mode: T_{mute} = LSM[2:0] * 2*(1 µs)
in both 1H/2H modes, if LSM[2:0] = 0 then T_{mute} = Hsync width.

OSD CONTROLLER (Cont'd)

Table 33. OSD Register Map

Register Number Page 42	Register Name	7	6	5	4	3	2	1	0
246	**OSDBCR2**	B2BC	-	BOS2	BOS1	BOS0	BOR2	BOR1	BOR0
247	**OSDBCR1**	DIFB	-	BOG2	BOG1	BOG0	BOB2	BOB1	BOB0
248	**OSDER**	DION	OSDE	TE	DBLS	NIDS	TSLE	MOPE	FPIXC
249	**OSDDR**	PASW	HPOL	VPOL	FBPOL	VD3	VD2	VD1	VD0
250	**OSDFBR**	BUFL	VSY	HSY	VSDL	FIELD	DINT	MOIT	SL8
251	**OSDSLR**	SL7	SL6	SL5	SL4	SL3	SL2	SL1	SL0
252	**OSDMR**	ADMULT	ODEVN	-	-	-	LSM2	LSM1	LSM0

8.6 CLOSED CAPTION DATA SLICER (DS)

8.6.1 Introduction

Depending on the ST9 device, one or two Data Slicers may be available in the MCU (refer the device feature list and Register map).

Each Data Slicer can extract either

- Closed caption data from a composite video signal broadcast in the EIA-608 format. Used in conjunction with the OSD, it allows closed caption information to be displayed on a TV screen.
- Gemstar format data transmitted on one or more horizontal lines in the vertical blanking interval. In this format, one line contains 4 bytes of data.

The Data Slicer automatically determines the data format in the specified line and sets the appropriate flag.

8.6.2 Functional Description

Inputs

CCVIDEO: Composite video signal AC coupled through a 1µF capacitor.

HSYNC: Horizontal deflection pulse.

VSYNC: Vertical deflection pulse.

F_4MHz: 4 MHz clock from frequency multiplier.

Outputs

IRQ: Conditional negative edge interrupt request, connected to a CPU interrupt channel (See Interrupts Chapter).

As shown in Figure 67, the Data Slicer accepts the incoming composite video as an AC coupled signal through a 1µF capacitor to the CCVIDEO pin. The OSD synchronization signals and a 4 MHz crystal derived clock are used in the data slicer signal extraction logic.

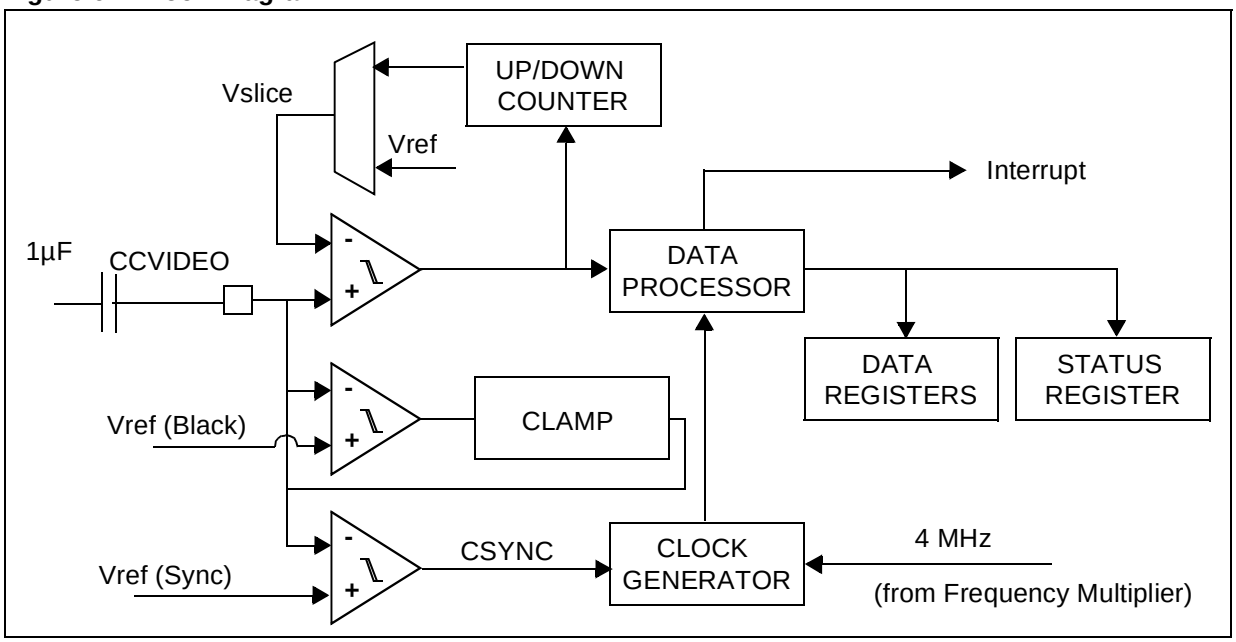
Data extraction can be programmed for a selectable line in either field for a video signal of amplitude of 2V +/-3dB. The slicing level for data is controlled automatically by hardware.

The output signal DSOUT is high when the input signal exceeds the level of the reference voltage Vslice, and low when it is less than Vslice.

The clamp is disabled from the time of detection of vertical sync in CCVIDEO (a sync pulse wider than 12µs) up to line 28.

Note: For good slicing results, it is advised to set VDDA >= 5.3V. Remember also that VDDA < VDD + 0.3V.

Figure 67. Block Diagram



DATA SLICER (Cont'd)

8.6.3 Data Slicer Operation

The data slicer is enabled or disabled using the EDS bit in the DR1 register. The Data Slicer clock frequencies are generated by the clock generator starting from a basic 4 MHz clock.

The decoder is activated when the output of a half-line counter matches the value written by the user in the CR1 register and a horizontal sync pulse is detected while the match is valid. Odd values of half-line counts in register CR1 are used to decode lines in Field#1, even values to decode lines in Field#2. The CC decoder includes logic for recognizing the current field and generates a corresponding FIELD1 signal.

The slicing level is automatically adjusted during the clock run-in window to obtain approximately a 50% duty cycle waveform corresponding to the clock run-in signal at the data slicer output. Refer to Figure 68.

DSOUT is fed into the data processor where it is processed for the selected line. The waveforms of DSOUT at the output of the comparator for signals received in either closed caption or Gemstar format are shown in Figure 68. The closed caption signal starts with 7 cycles of clock run-in signal with a frequency of about 500KHz, and ends with 16 bits (2 bytes) of data where each bit has a period of approximately 2µs. The Gemstar signal has only 5 cycles of the 500KHz clock run-in signal and transmits 32 bits (4 bytes) of data with the period of each bit reduced to about 1µs.

Closed caption and Gemstar signals are detected by looking for their distinctive frame codes. "Frame code" refers to the characteristic of the signal waveform in a time interval between the clock run-in and data sections of the signal as shown in Figure 68. The frame code detector examines during a 5µs window selected outputs of a 32-stage shift register which holds consecutive samples of DSOUT obtained with a 4MHz clock at.25µs intervals. In normal operation the frame code detector requires the 5 bits marked with an upward pointing arrow (^) to be correct, but in a search mode covering all lines in the vertical blanking interval all 8 bits of the frame code are examined by when the SEARCH control bit in the CR2 register is set.

Identification of the frame code results in setting either the CCMODE or GSMODE bit in the MR register. Then the clock for recovering the following 2 or 4 data bytes is activated in correct phase relative to the data signal. Parity of all data bytes is checked and appropriate flags EVNP[4:1] are set correspondingly in the MR register.

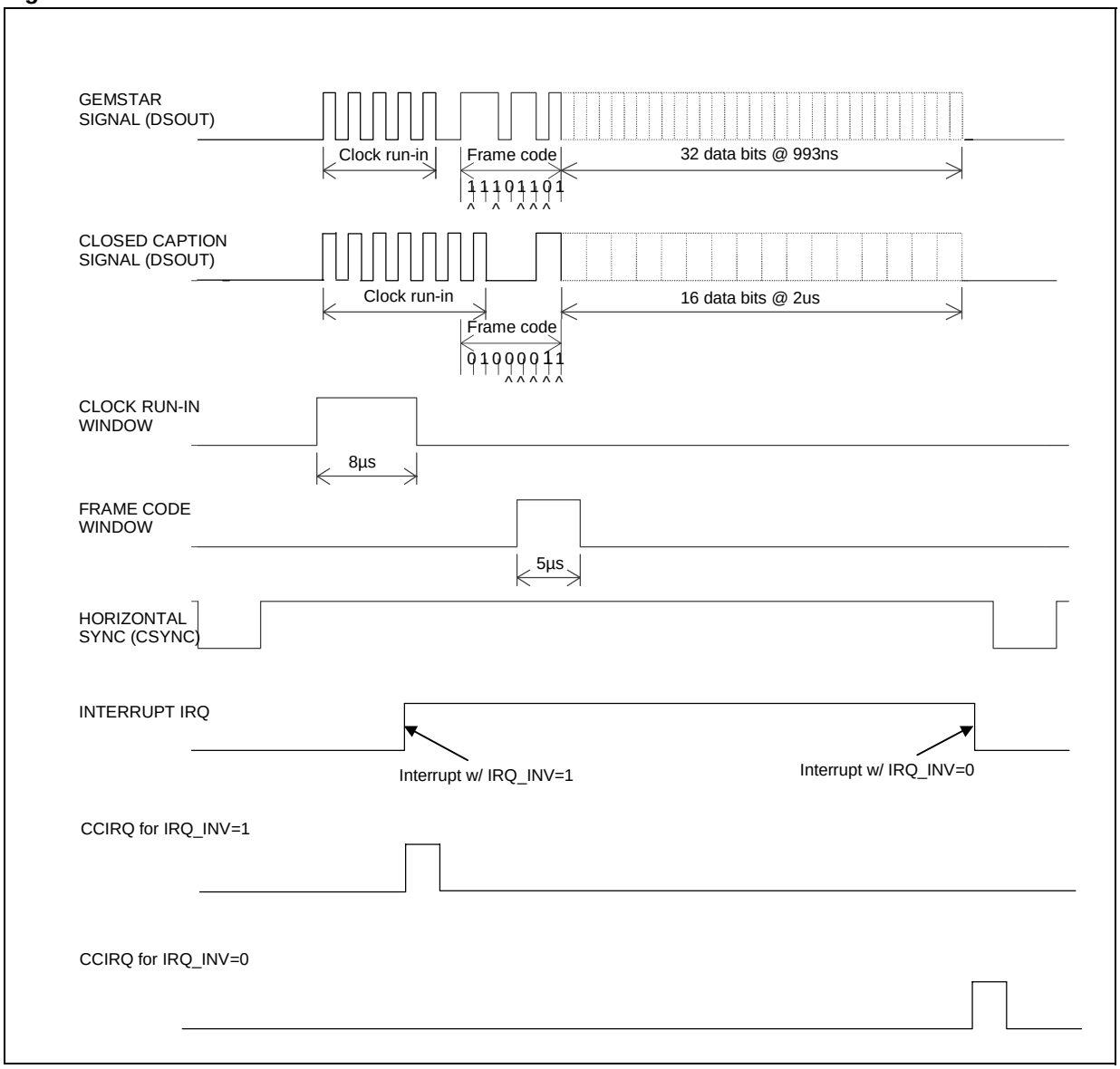
An interrupt is always generated at the end of the specified line and on the corresponding line of the opposite field, even if no data has been recovered.

The Data Slicer is able to recover data from 2 or more adjacent lines. However, except for the last line to be decoded in the current field, in such a situation recovered data bytes must be read out within a time period of 18µs after an interrupt has occurred.

During normal operation, the interrupt should occur at the end of the selected line. However, if data must be recovered from adjacent lines in the same field, then it becomes necessary for some of the lines to generate an interrupt at the leading edge of interrupt signal, and thereafter for the same line at the trailing edge of the interrupt signal. The IRQ_INV bit in the CR1 Register controls the polarity of the interrupt signal. The TED0 bit in the EITR register should be set to "1" and INV_IRQ should be used to control the data slicer interrupt polarity.

DATA SLICER (Cont'd)

Figure 68. Data Slicer Waveforms



DATA SLICER (Cont'd)

8.6.4 Interrupt handling

If the device has two data slicers, both use the same interrupt channel. So some additional software is needed, when using both slicers at the same time. Using only one of them (i.e. DS0) allows for software compatibility with the ST9296.

The interrupt requests, issued by the two slicers are ORed in order to generate an interrupt signal on the same ST9 interrupt channel.

The internal signals IRQ1 for DS0 / IRQ2 for DS1 (see Figure 68) are activated when the line number matches the value set by the user in the first control register and this signal is reset at the end of the line: the IRQ_INV bit in the CR1 register allows to select whether the rising edge or the falling edge of this signal will cause a positive pulse on CCIRQ1 or CCIRQ2. As the CCIRQ pulses are very short, the TED0 control bit in the EITR register is of very limited use. It is recommended to keep this bit always at "1" (interrupt on rising edge).

In order to find out the source of an interrupt request, each of the two slicers sets the IRFL bit in the CR2 register when an interrupt request is issued. Both of these flags (IRFL for DS0 and DS1) must be scanned and then cleared by the software during every interrupt service routine.

Note: There are situations where data has to be recovered from one or more lines in only one field, i.e. either Field#1 or Field#2. In such cases an unnecessary interrupt is generated in the field that does not contain any lines with data that is of interest. For example, to recover closed caption data on Line#21 in Field#1, the half line count LN[5:0] in the CR1 regsiter can be permanently set to 33. This value of LN[5:0] will also cause an interrupt to be generated in the middle of Line#21 in Field#2. The interrupt routine must therefore determine the status of the FIELD1 bit in the MR register and immediately return to the main program if this bit is low.

Note: An interrupt is always generated at the end of the specified line (and on the corresponding line of the opposite field).

DATA SLICER (Cont'd)

8.6.5 Register Description

The register description lists the register page for Data Slicer 0 (DS0) If a second Data Slicer is available (DS1), it is mapped in Page 46.

DATA REGISTER 1 (DR1)

R240 - Read only

Register Page: 45
Reset Value: 0000 0000 (00h)

7							0
D1.7	D1.6	D1.5	D1.4	D1.3	D1.2	D1.1	D1.0

Bit 7:0 = **D1[7:0]:** *First data byte for Gemstar*

DATA REGISTER 2 (DR2)

R241 - Read only

Register Page: 45
Reset Value: 0000 0000 (00h)

7							0
D2.7	D2.6	D2.5	D2.4	D2.3	D2.2	D2.1	D2.0

Bit 7:0 = **D2[7:0]:** *Second data byte for Gemstar*

DATA REGISTER 3 (DR3)

R242 - Read only

Register Page: 45
Reset Value: 0000 0000 (00h)

7							0
D3.7	D3.6	D3.5	D3.4	D3.3	D3.2	D3.1	D3.0

Bit 7:0 = **D3[7:0]:** *Third data byte for Gemstar, first data byte for closed caption*

DATA SLICER (Cont'd)

DATA REGISTER 4 (DR4)

R243 - Read only

Register Page: 45
Reset Value: 0000 0000 (00h)

7							0
D4.7	D4.6	D4.5	D4.4	D4.3	D4.2	D4.1	D4.0

Bit 7:0 = **D4[7:0]:** *Fourth data byte for Gemstar, second data byte for closed caption*

CONTROL REGISTER 1 (CR1)

R244 - Read/Write
Register Page: 45
Reset Value: 0000 0000 (00h)

7							0
STNDBY	IRQ_INV	LN5	LN4	LN3	LN2	LN1	LN0

Bit 7 = **STNDBY**: Standby mode

This bit selects the standby mode (operation when main power supplies have been turned off).
0: The horizontal deflection pulses (HPLS) are used for synchronization
1: Horizontal synchronization pulses are internally generated from the incoming video signal (vertical synchronization signal for the Data Slicer is always derived from incoming video).

Bit 6 = **IRQ_INV:** *Interrupt Signal Polarity.*

This bit is set and cleared by software. It controls the position of the data slicer's interrupt signal to the µP (CCIRQ) to be on the rising or falling edge of the slicers interrupt signal (IRQ). Refer to Figure 68. It is used to setup the interrupt to occur at the beginning of the specified line or at the end of the line.
0: The interrupt request occurs at the end of the specified line
1: The interrupt request occurs at the beginning of the specified line.

Note: Data recovery from adjacent lines:
In situations where closed caption or Gemstar data has to be recovered from adjacent lines, e.g. from Line#20 and Line#21 in Field#1, it becomes necessary to generate an interrupt both at the rising edge as well as at the falling edge of the IRQ signal shown in Figure 68. The interrupt at the rising edge of IRQ for Line#20 is obtained by setting the IRQ_INV bit and writing the LN[5:0] value for Line#21 in the CR1 Register. Usually it will be too late to write this value at the end of Line#20. After the LN[5:0] value for Line#21 has been written, the software must also clear the IRQ_INV bit before exiting the interrupt routine. This will cause the next interrupt to be generated at the falling edge of IRQ which occurs at the end of Line#20. At this time the character codes transmitted in Line#20 are available for reading.

Bit 5:0 = **LN[5:0]:** *Closed caption line selector.*

The data slicer current half-line count is compared with the LN[5:0] value; the half-line counter is released from the reset state when the first vertical sync pulse is detected. With the NTSC standard, to select line "N" the LN[5:0] value must be set to LN[5:0] = [(2*N)-9] for Field#1, and [(2*N)-8] for Field#2.

Examples: LN[5:0] = 33 (21hex) for Line#21 in Field 1, LN[5:0] = 34 (22hex) for Line#21 in Field 2.

CONTROL REGISTER 2 (CR2)

R245 - Read/Write
Register Page: 45
Reset Value: 0000 0000 (00h)

7							0
EDS	IRFL	CCID	SEARCH	SCG_EN	PHD2	PHD1	PHD0

Bit 7 = **EDS:** *Enable Data Slicer.*
This bit is set and cleared by software.
0: Disable Data Slicer (analog part)
1: Enable Data Slicer

Bit 6 = **IRFL:** *Interrupt Flag*.
This bit is set by hardware, when an interrupt request is issued. It must be cleared by software when the corresponding interrupt service routine has been finished. A 'clear' can be performed by any write operation to this register.
0: No Data Slicer interrupt pending
1: Data Slicer interrupt pending

Bit 5 = **CCID:** *External Interrupt Source Selection.*
This bit is set and cleared by software.
0: The interrupt request from the Data Slicer is forwarded to the CPU
1: The interrupt from the external interrupt pin is forwarded to the CPU.

DATA SLICER (Cont'd)

Bit 4 = **SEARCH:** Enhanced Signal Search.

This bit is set and cleared by software. This bit should be set to improve the reliability of properly identifying signals in closed caption and Gemstar format when doing a search encompassing all horizontal lines in the vertical blanking interval.
0: Check 5 bits in the Frame code
1: Check 8 bits in the Frame code

Bit 3 = **SCG_EN:** *Block Copy-Guard signals.*

In video signals with Copy-Guard protection pulses similar to horizontal sync pulses may have been inserted into certain lines in the vertical blanking interval.
0: No blocking
1: Copy-Guard pulse occurring 8 µs before the anticipated occurrence of normal horizontal sync pulses are blocked from reaching the line detection logic

Bit 2 = **PHD[2:0]:** *Horizontal phase compensation.*

These bits specify the delay added in 4µs increments to the clock for the half-line counter in logic that identifies the specified line. This feature is used in 2H mode of operation to compensate for a large phase difference which may exist between 2H deflection pulses and horizontal sync pulses extracted from video.

MONITOR REGISTER (MR)

R246 - Read/Write

Register Page: 45
Reset Value: 0000 0000 (00h)

7							0
CC MODE	GS MODE	FIELD 1	IN_SY NC	EVNP 4	EVNP 3	EVNP 2	EVNP 1

Bit 7 = **CCMODE:** *Closed Caption Mode.*

This bit is set by hardware and cleared by software.
0: No closed caption format detected
1: Closed caption format detected

Bit 6 = **GSMODE:** *Gemstar Mode.*

This bit is set by hardware and cleared by software.
0: No Gemstar format detected
1: Gemstar format detected

Bit 5= **FIELD1:** *Field 1 flag.*

This bit is set and cleared by hardware.
0: Field 2 detected
1: Field 1 detected

WARNING: The bit value is not valid after tuning of a new channel or in case of a momentary drop-out of the tuned signal. In this case, the use of the IN_SYNC bit is required.

Bit 4= **IN_SYNC:** *Phase of line monitoring relative to video field.*

This bit is set and cleared by hardware. Software should read this bit periodically in case data in closed caption and/or Gemstar format must be recovered from lines in both fields (e.g. closed captions from Line#21 of Field1 and extended data service information from Line#21 of Field2).
0: Periodic writing of the appropriate line codes into Register CR1 is out of phase with the field sequence of the video input signal. In this case the phasing can be corrected by skipping just once the change of the line code stored in the CR1 register and writing the immediately following field.
1: The start of a specified line has been detected in one or both of the last 2 fields.

Note: In situations where the closed caption or Gemstar data is on lines which are all in the same field, e.g. Field#1, the hardware in the data slicer cell will automatically correct an out-of-phase situation. Such a situation will be corrected within the period of one frame of video.

In a more general situation, data must be recovered from lines in both fields, and then the hardware cannot on its own correct the polarity of an out-of-phase FIELD#1 signal. The correction must therefore be done by software on basis of information provided by the IN_SYNC status bit.

Bit 3:0= **EVNP[4:1]**: *Parity Event Bits*

These bits are set and cleared by hardware. They indicate the detected parity of data bytes 1 through 4 (only 3 and 4 for closed caption format). These bits should normally be low.
0: Odd parity detected
1: Even parity detected

Table 34. DS Register Map

Register Number Page 45	Register Name	7	6	5	4	3	2	1	0
240	**DR1**	D1.7	D1.6	D1.5	D1.4	D1.3	D1.2	D1.1	D1.0
241	**DR2**	D2.7	D2.6	D2.5	D2.4	D2.3	D2.2	D2.1	D2.0
242	**DR3**	D3.7	D3.6	D3.5	D3.4	D3.3	D3.2	D3.1	D3.0
243	**DR4**	D4.7	D4.6	D4.5	D4.4	D4.3	D4.2	D4.1	D4.0
244	**CR1**	STNDBY	IRQ_INV	LN5	LN4	LN3	LN2	LN1	LN0
245	**CR2**	EDS	IRFL	CCID	SEARCH	SCG_EN	PHD2	PHD1	PHD0
246	**MR**	CCMODE	GSMODE	FIELD1	IN_SYNC	EVNP4	EVNP3	EVNP2	EVNP1

8.7 VIDEO SYNC ERROR DETECTOR (SYNCERR)

8.7.1 Functional Description

The Sync Error Detector provides information to the tuning system whether an IF signal is a picture carrier or not. The CSYNC source for the detector is selected using the SYSEL[1:0] bits in the IRSCR register: it is internally extracted fro CCVIDEO1 and 2 or directly taken from the SYNDET1 or SYNDET0 pins.

The number of positive transitions of CSYNC in a 78µsec window is checked. One or two transitions should occur in every window (horizontal sync pulses occur at intervals of 63.5µs). An error counter is incremented at end of a window in case of one or more of the following situations :

1.No low to high transitions of the signal were detected within the 78µs wide window.

2.More than 2 low-to-high transitions were detected within the window.

3.More than 2 consecutive samples of CSYNC taken at 8µs intervals within the window were high.

The errors are accumulated for a period of one field as defined by two adjacent vertical deflection pulses. Then the error count for the field is latched into the SYNCER register and the VALID flag is set.

The frequency of pulses applied to SYNDET0 or SYNDET1 input not producing any errors is in the range of 13KHz to 25Khz. The width of the pulses applied to these inputs and not producing any errors is less than 16us.

Inevitably, error counts are generated in the vertical sync interval in presence of double frequency equalization and wide sync pulses. With a standard video signal the typical error count is 5. The error count threshold for an acceptable video signal can be set on basis of experimental results with a typical value of about 30.

8.7.2 Register Description

SYNC ERROR REGISTER (SYNCER)

R249 - Read only, except bit 7

Register Page: 43
Reset Value: 0000 0000 (00h)

7							0
VALID	SD6	SD5	SD4	SD3	SD2	SD1	SD0

Bit 7 = **VALID:** *Data valid bit*

This bit is set by hardware on the leading edge of a vertical deflection pulse. It is cleared by software

Bit 6:0 = **SD[6:0]**: Sync error count

These bits are updated by hardware on the leading edge of a vertical deflection pulse.

IR/SYNC CONTROL REGISTER (IRSCR)

R250 - Read/Write
Register Page: 43
Reset Value: 0000 0000 (00h)

7							0
0	0	SYSEL1	-	-	-	-	SYSEL0

Bit 7:6 = Reserved. Forced by hardware to 0.

Bit 5, 0 = **SYSEL[1:0]**: *Sync error detector Input selection*

SYSEL1	SYSEL0	
0	0	CSYNC signal on SYNDET0 input
0	1	CSYNC signal on SYNDET1 input
1	0	CSYNC extracted from CCVIDEO1
1	1	CSYNC extracted from CCVIDEO2

Bit 4:1 = Reserved (used for IR Preprocessor). Refer to the IR Preprocessor chapter.

8.8 IR PREPROCESSOR (IR)

8.8.1 Functional Description

The IR Preprocessor measures the interval between adjacent edges of the demodulated output signal from the IR amplifier/detector. You can specify the polarity using the POSED and NEGED bit in the IRSCR register The measurement is represented in terms of a count obtained with a 12.5KHz clock and stored in the IRPR register.

Whenever an edge of specified polarity is detected, the count accumulated since the previously detected edge is latched into an 8-bit register and an interrupt request IRQ is generated if the IRWDIS bit is reset in the IRSCR register.

Note: Any count less than 255 stored in the latch register is over-written in case the µP fails to execute the read before the next edge occurs.

In case an edge is not detected in about 20ms (the count reaches its maximum value of 255) the count is latched immediately and the IRQ flag is set. An overflow flag (not accessable) is also set internally.

Each time an interrupt is received, it must be acknowleged by writing any value in the IRPR register. Otherwise no further interrupts will be generated.

Warning: The content of the latch cannot be changed as long as the overflow flag remains set. To clear the IRQ and internal overflow flags, just write any value in the IRPR register. As long as the internal overflow flag is set, no interrupt is generated.

The IR input signal is preprocessed by a spike filter. The FLSEL bit of the IRSCR register determines the width of filtered pulse.

8.8.2 Register Description

IR PULSE REGISTER (IRPR)

R248 - Read only
Register Page: 43
Reset Value: 0000 0000 (00h)

7							0
IR7	IR6	IR5	IR4	IR3	IR2	IR1	IR0

Bits 7:0 = **IR[7:0]**: IR pulse width in terms of number of 12.5KHz clock cycles.

IR/SYNC CONTROL REGISTER (IRSCR)

R250 - Read/Write
Register Page: 43
Reset Value: 0000 0000 (00h)

7							0
0	0	-	IRWDIS	FLSEL	POSED	NEGED	-

Bit 7:6 = Reserved. Forced by hardware to 0.

Bit 5 = Reserved (used for Sync Error Detector). Refer to the Sync Error Detector chapter.

Bit 4 = **IRWDIS**: *External Interrupt Source.*

This bit is set and cleared by software. It selects the source of the interrupt assigned to the external interrupt channel. Refer to the Interrupt Chapter.
0: The interrupt request from the IR preprocessor is forwarded to the CPU
1: The interrupt from the external interrupt pin is forwarded to the CPU

Bit 3 = **FLSEL**: *Spike filter pulse width selection*

This bit is set and cleared by software. It selects the spike filter width.
0: Filter pulses narrower than 2µs
1: Filter pulses narrower than 160µs

Bit 2:1 = **POSED**, **NEGED** *Edge selection for the duration measurement*

NEGED	POSED	Count latch at ...
1	1	Positive or negative transition of IR or when count reaches 255
1	0	Negative transition of IR or when count reaches 255
0	1	Positive transition of IR or when count reaches 255
0	0	Only when count reaches 255

Bit 0 = Reserved (used for Sync Error Detector).

8.9 FOUR-CHANNEL I2C BUS INTERFACE (I2C)

8.9.1 Introduction

The I2C Bus Master/Slave Interface supports up to 4 serial I2C buses used for communication with various external devices. It meets all of the requirements of the I2C bus specification (except extended 10-bit addressing compatibility for slave operation and CBUS compatibility).

8.9.1.1 General Features

– Conversion of internal 8-bit parallel data to/from external I2C bus serial data
– Realtime interrupt generation and handling
– Software selectable operation one of four I2C buses
– Software selectable acknowledge bit generation
– Internal general reset
– 8-bit data read/write register
– 8-bit control register,
– 8-bit status register,
– Operates by default in slave mode and is automatically switched to master mode by loading the ‘data write’ register when the bus is idle.

8.9.1.2 Master Operation

– 4-bit Frequency Control register to select 1 of 16 clock frequencies for the SCL line ranging from 20 kHz to 800 kHz derived from a 4MHz crystal clock
– Compatible with standard 7 or extended 10-bit address protocol
– Handles stretching of SCL bus clock pulses by slaves without restrictions
– Bus arbitration with arbitration loss detection in multimaster environment
– Bus error detection
– Optional push-pull bus drive capability for faster communication

8.9.1.3 Slave Operation

– 7-bit address register (cannot be assigned a 10-bit address)
– The first SCL clock pulse in every data byte is stretched until the MCU has finished processing the previously received byte
– Bus error detection
– Optional general call detection
– Operates optionally as Bus Monitor without interfering in any way with bus traffic
– Setup time for any first transmitted data bit can be adjusted

I²C BUS INTERFACE (Cont'd)

8.9.2 General Description

In addition to receiving and transmitting data, this interface convert them from serial to parallel format and vice versa. The interface is connected, through a multiplexer, to one I²C bus among 4 by a data pin, SDAx, and by a clock pin, SCLx, where x range value is 1 to 4.
It can be connected both with a standard I²C bus and a Fast I²C bus. This selection is made by software.

8.9.2.1 Mode Selection

The interface can operate in the four following modes:

– Slave transmitter/receiver

– Master transmitter/receiver

By default, it operates in slave mode.

The interface automatically switches from inactive slave to master after it generates a START condition and from master to inactive slave in case of arbitration loss or a STOP generation, this allows Multi-Master capability.

8.9.2.2 Communication Flow

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. Start condition is automatically generated by the interface when the data register is loaded by the slave address (see register description for further details). Stop condition is generated in master mode by writing by software in the control register.

In Slave mode, the interface is capable of recognising its own address (7-bit), and the General Call address. The General Call address detection may be enabled or disabled by software.

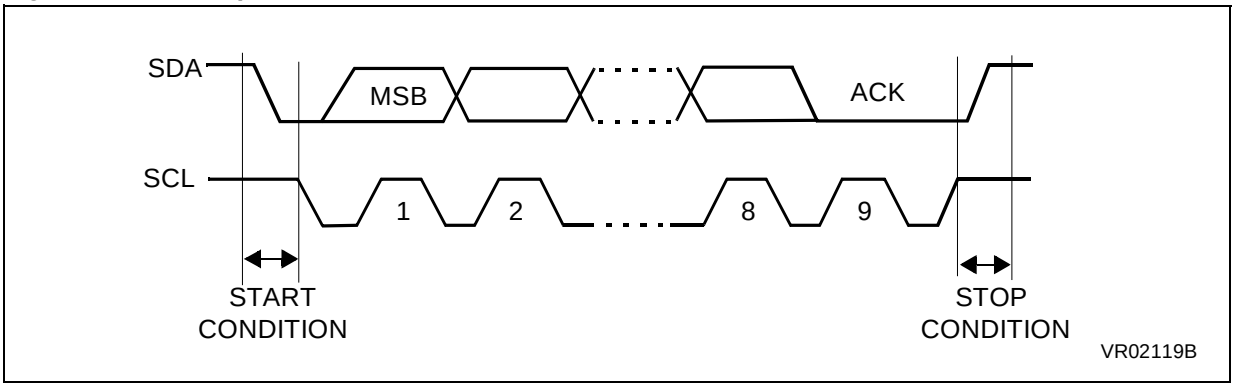
Data and addresses are transferred as 8-bit bytes, MSB first. The first byte following the start condition is the address byte; it is always transmitted in Master mode.

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. Refer to Figure 69.

The acknowledge may be enabled and disabled by software.

The speed of the I²C interface may be selected between Standard (15.625 - 100 kHz), Fast I²C (100- 400 kHz), or Extended I²C (500 - 800 kHz).

Figure 69. I²C bus protocol



I²C BUS INTERFACE (Cont'd)

8.9.3 Functional Description

Refer to Section 8.9.6 for the bit definitions.

Figure 70 gives the block diagram of the cell.

By default, the I²C interface is in inactive slave mode, except when it initiates a transmit or receive sequence.

After the microcontroller power-on reset state, the I²C interface is in reset state until the CLEAR bit (I2CCTR register) is reset.

8.9.3.1 Configuring the interface

Before using the I²C interface, configure it as follows.

If it is to be used in slave mode, write the address assigned to the interface in the I2COAR register.

If it is to be used in master mode, write the SCL clock frequency in the I2CFQR register.

Then, select one of the four buses available and configure the corresponding pins to the alternate function (refer to the I/O port chapter).

Depending on your application, you may use the advanced features (see the UNPROC and UNEXP bits of the I2CSTR2 register) by setting the AFEN bit of the I2CCTR register.

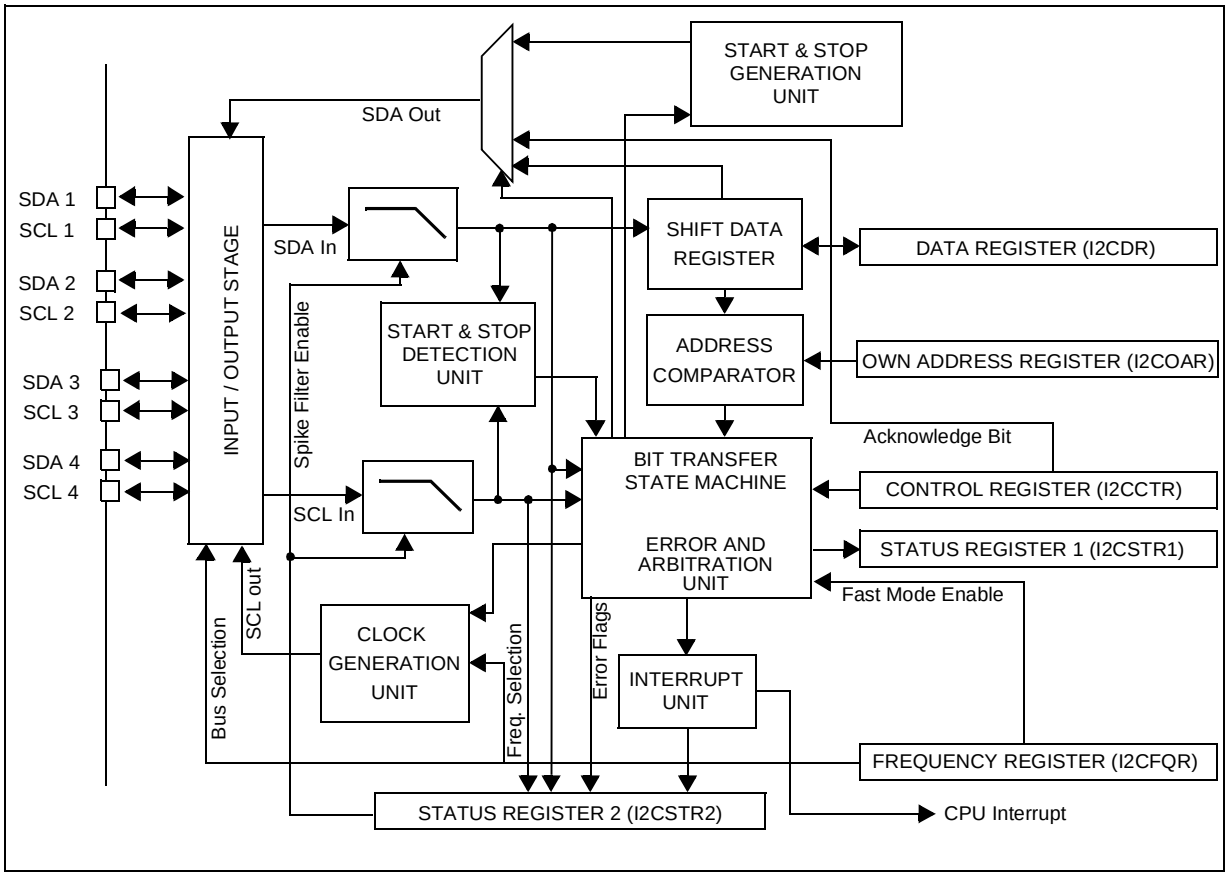
You may also optionally set the RSRT and STOP bits of the I2CCTR register.

You can enable the interrupt on stop condition and the Spike filter by setting the ISCEN and SFEN bits of the I2CSTR1 register.

If you want to use the monitor feature, then set the MONITOR bit in the I2CCTR register.

In all cases reset the CLEAR bit of the I2CCTR register to enable the I²C interface.

Figure 70. I²C interface block diagram



I²C BUS INTERFACE (Cont'd)

8.9.3.2 Slave Mode

As soon as a start condition is detected, the address is received from the SDA line and sent to the shift register; then it is compared with:

– The 7 MSB of the interface address (see I2COAR register) if the ADR0 bit = 0

– The 4 MSB of the interface address (see I2COAR register) if the ADR0 bit = 1

– The General Call address

Address not matched: the interface ignores it and waits for another Start condition.

Address matched: the interface generates in sequence:

– Acknowledge pulse if the GENC_ACK bit (I2CCTR register) is set and a general call is detected, or if the SEND_ACK bit (I2CCTR register) is reset and "normal" address if detected.

– An interrupt is generated and the INT bit of the I2CSTR2 register is set.

Then check the I2CSTR1 register to know the interface status:

– Read the FIRST bit of the I2CSTR1 register to know whether the byte stored in the I2CDR register is the address (first byte transferred in an I2C transaction) or a data.

– If the GEN_CALL bit is set, a general call has been requested by a master.

– If the ACT_SLV bit is set and the READ bit is set, the interface is an active slave transmitter, else, if the ACT_SLV bit is set and the READ bit is reset, the interface is an active slave receiver.

Slave receiver

After the address, the slave receives bytes from the SDA line into the I2CDR register via the internal shift register. After each received byte the interface generates in sequence:

– Acknowledge pulse according to the SEND_ACK bit value

– An interrupt is generated and the INT bit of the I2CSTR2 register is set.

Using the FIRST bit of the I2CSTR1 register, you know whether the byte stored in the I2CDR register is the address (first byte transferred in an I2C transaction) or data.

Slave Transmitter

Following the address reception, the slave sends bytes from the DR register to the SDA line via the internal shift register.

The slave writes in the I2CDR register the data to send on the SDA bus.

When the acknowledge pulse is received:

– an interrupt is generated and the INT bit of the I2CSTR2 register is set.

Then you need to check the ACK_BIT of the I2CSTR2 register to know whether the last byte has been acknowledged or not. If some data have to be sent again, write the value in the OSDDR register.

Closing a slave communication

The I2C interface returns to inactive slave state as soon as a stop condition has been detected.

If the ISCEN bit of the I2CSTR2 is set, an interrupt is generated on detecting the stop condition, allowing the user to know if the transaction was successful by checking the ERROR and ACTIVE flags of the I2CSTR1 register.

8.9.3.3 Master Mode

To switch from default inactive slave mode to Master mode: load a slave address in the I2CDR register.

If the bus is free (ACTIVE bit of the I2CSTR2 register reset), then the I²C interface automatically generates a start condition followed by the I2CDR byte.

Then, on the 9th clock pulse, an interrupt is generated and the INT bit of the I2CSTR2 register is set.

Check the ACK_BIT bit of the I2CSTR1 register to know whether the slave address has been acknowledged or not, in order to manage the transaction.

If needed, generate a stop condition on the bus with the STOP bit of the I2CSTR1 register.

Note: If the RSRT bit of the I2CCTR register is set, the master will generate a repeated start sequence as soon as a new byte is loaded in the I2CDR register.

I²C BUS INTERFACE (Cont'd)

Master Receiver

Following the address transmission and acknowledgment, the master receives bytes from the SDA line into the I2CDR register via the internal shift register. After each byte the interface generates in sequence:

– Acknowledge pulse according to the SEND_ACK bit value,

– An interrupt is generated and the INT bit of the I2CSTR2 register is set.

Then read the I2CDR register to store the transmitted data.

Note: In order to generate the non-acknowledge pulse after the last received data byte, the SEND_ACK bit must be set just before reading the second last data byte.

Master Transmitter

Following the address transmission and acknowledgment, the master sends bytes from the I2CDR register to the SDA line via the internal shift register.

When the acknowledge bit is received, the interface generates an interrupt and sets the INT bit of the I2CSTR2 register.

The user can check the ACK_BIT bit of the I2CSTR1 register in order to handle the transaction properly.

Closing a master communication

The master interface will generate a stop condition on the bus when the user sets the STOP bit of the I2CSTR1 register.

8.9.4 Interrupt Handling

To acknowledge interrupts generated by the I2C interface, software must write any value in the I2CDR register before leaving the I2C interrupt subroutine. This is necessary in all modes including slave or master receiver mode.

8.9.5 Error Cases

Each time an error occurs, an interrupt is generated. Then by checking the following bits, the user can identify the problem:

– If the **ERROR** bit in the I2CSTR1 register is set, an illegal start or stop condition has been detected.
If the AFEN bit in the I2CCTR register is set, the UNPROC, UNEXP, and MISP bits of the I2CSTR2 register indicate what kind of illegal condition has been detected.

– If the **ARB_LOST** bit, in the I2CSTR1 register is set, an arbitration lost occurred on the bus.

Note: the ERROR bit has higher priority than the ARB_LOST bit, so if ERROR is set, ARB_LOST has to be ignored.

I²C BUS INTERFACE (Cont'd)

8.9.6 Register Description

OWN ADDRESS REGISTER (I2COAR)
R240 - Read/Write
Register Page: 44
Reset Value: 0000 0000(00h)

7							0
ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	ADR0

Bit 7:1 = **ADR[7:1]** *Interface Slave Address*
These bits are the 7 most significant bits of the 8-bit address assigned to interface when it works in slave mode.

Bit 0 = **ADR0** *Address match bit*
This bit selects when the I²C interface becomes active if it works in slave mode, and if its slave address is transmitted on the bus.
0: The interface becomes an active slave when the 7 most significant bits ADR[7:1] match the address transmitted by a master.
1: The interface becomes an active slave when only the 4 most significant bits ADR[7:4] match the address transmitted by a master. This feature allows the master to send data simultaneously to up to 8 slaves with identical ADR(7:4).

I2C BUS INTERFACE (Cont'd)

FREQUENCY REGISTER (I2CFQR)
R241 - Read/Write
Register Page: 44
Reset Value: 0000 0000(00h)

7							0
BUS_S0	BUS_S1	FMEN	PP_DRV	Q3	Q2	Q1	Q0

Bits 7:6 = **BUS_S[1:0]** *I2C BUS Selection bits*
These bits connect the I2C interface to one of the four possible buses as described in Table 35.

Table 35. I2C bus selection

BUS_S1	BUS_S0	Selected Bus
0	0	SCL1/SDA1
0	1	SCL2/SDA2
1	0	SCL3/SDA3
1	1	SCL4/SDA4

Bit 5 = **FMEN** *Fast Mode Enable bit*
This bit enables or disables the fast mode for the SCL bus frequency.
0: Standard Mode (up to 100 kHz).
1: Fast Mode (over 100 kHz)

Bit 4 = **PP_DRV** *Push-Pull Drive mode bit*
This bit determines if the master drives the SCL/SDA buses in push-pull mode or in normal mode. This allows the master to send data to the slave at a faster speed.
0: The push-pull drive mode is disabled
1: The push-pull drive mode is enabled. All "normal" bus frequencies are doubled with the only exception that the push-pull drive mode is automatically disabled when Q[3:0]=1110 or Q[3:0]=1111 to yield an SCL frequency of 500 kHz or 800 kHz. Refer to Q[3:0] bit description.

Note: The master automatically switches temporarily to normal bus driving mode with active pull-up disabled and SCL frequency reduced by factor of 2 when receiving acknowledges or data from the addressed slave.

Bit 3:0 = **Q[3:0]** *SCL clock frequency bits*
These bits select the SCL clock frequency when the interface works in master mode. In slave transmitter mode, they can be used to adjust the setting up time between the first data byte and the clock. Refer to Table 36.

In push-pull mode, the frequency values presented in the following table correspond to an approximate frequency assuming that :

– the first data bit is transferred at a lower frequency (clock stretching capability),

– the acknowledge bit is transferred at the slave speed without push-pull mode,

– other data bits are transferred with a real period 250 ns shorter than the values indicated in this table.

Using the spike filter will add an internal delay acting as a period increase by 250-ns steps.

Table 36. SCL Clock Frequency Selection

	SCL BUS FREQUENCY (in kHz)	
Q[3:0]	PP_DRV = 0 SCL max Frequency	PP_DRV = 1 SCL Frequency (kHz) (period: +0/-250ns
0000	20.10	40.40
0001	30.53	61.54
0010	40.40	81.63
0011	50.63	102.56
0100	63.49	129.03
0101	72.73	148.15
0110	85.11	173.91
0111	102.56	210.53
1000	129.03	266.67
1001	173.91	363.64
1010	210.53	444.44
1011	266.67	444.44
1100	400.00	571.43
1101	444.44	666.67
1110	444.44*	666.67
1111	666.67*	666.67

* These values are not covered by the Philips I2C specification

Notes:

– The maximum allowed frequency depends on the state of the FMEN control bit (If PP_DRV=0, standard mode: 100 KHz; fast mode: 666.6 kHz)

– All frequency values depend on the bus line load (except push-pull mode).

– All above values are obtained with loads corresponding to a rise time from 0 to 250 ns.

– Any higher rise time (especially in standard mode) will increase the period of the bus line frequency by 250-ns steps.

I²C BUS INTERFACE (Cont'd)

CONTROL REGISTER (I2CCTR)
R242 - Read/Write
Register Page: 44
Reset Value: 0000 0001(01h)

7							0
AFEN	RTI	GENC_ ACK	SEND_ ACK	MONI TOR	RSRT	STOP	CLEAR

Bit 7 = **AFEN** *Advanced Features Enable bit*
This bit enables or disables the unexpected & unprocessed error detection. Refer to the description of the UNPROC and UNEXP bits in the I2CSTR2 register.
0: Advanced features disabled
1: Advanced features enabled

Bit 6 = **RTI** *Return To Inactive state bit*
This bit determines the interface status after an interrupt is processed (either after a complete transfer or an error occured).
0: The interface keeps its active state
1: The interface (master or slave) returns to the inactive slave state

Note: The state of the Active Flag (I2CSTR1.0) is maintained.The RTI bit is automatically cleared.

Bit 5 = **GENC_ACK** *General Call Acknowledge bit*
This bit determines the response of the I²C interface when a general call is detected on the bus.
0: The interface will acknowledge the reception of a 'General Call' immediately after receiving the address 00h. An interrupt is generated at the end of the acknowledge interval that follows the address.
1: The interface will not acknowledge a 'General Call' and does not generate an interrupt, i.e. the interface will remain an inactive slave.

Bit 4 = **SEND_ACK** *Send Acknowledge bit*
This bit is set by software to define if the acknowledge bit is placed on the bus when the interface is operating as a master receiver, active slave receiver or an active slave.
0: An inactive interface will acknowledge the reception of its address and switch to active slave mode.
1: The interface will not acknowledge the reception of its address and remains inactive.

Note: The interface operating as a master slave receiver is free to acknowledge or not all data bytes. In a normal I²C transaction, it acknowledges all data bytes except the last received from a slave/master transmitter.
SEND_ACK should be programmed before receiving the relevant byte (data or address).

Bit 3 = **MONITOR** *Bus Monitor mode bit*
This bit determines if the interface acts as a bus monitor or not.
0: The bus monitor mode is disabled.
1: The interface behaves as a bus monitor. The interface becomes a slave regardless of the address received, but neither the address or the following data is acknowledged (this is equivalent to SEND_ACK=1). If a read address is received, the high state of the least significant bit of this address is suppressed inside the interface and all data bytes are processed by the MCU as received data.

Bit 2 = **RSRT** *Repeated Start bit*
This bit determines if the interface generates automatically a repeated start condition on the I²C bus (in master mode) as soon as a new byte is ready to be send.
0: Repeated start disabled
1: Repeated start enabled

Note: This bit is automatically cleared.

Bit 1 = **STOP** *STOP condition generation bit*
When working in master mode, this bit enables or disables a STOP condition generation on the I²C bus.
0: No Stop condition is generated
1: The master will generate a stop condition to terminate the bus transaction. The master will automatically revert to an inactive slave and the STOP bit will be cleared.

Bit 0 = **CLEAR** *Clear interface bit*
This bit enables or disables the I²C interface.
0: The interface is enabled
1: A general reset is generated. The interface becomes an inactive slave and the SCL and SDA buses drive signals are removed. The system is kept in reset state until the CLEAR bit is written to "0".

Note: The CLEAR bit is "1" (i.e. the interface is disabled) when exiting from the MCUs power-on reset state.

I²C BUS INTERFACE (Cont'd)

DATA REGISTER (I2CDR)
R243 - Read/Write
Register Page: 44
Reset Value: 0000 0000(00h)

7							0
SR8	SR7	SR6	SR5	SR4	SR3	SR2	SR1

Bit 7:0 = **SR[8:1]** *address or data byte*
These bits contains the address or data byte loaded by software for sending on the I²C bus, and also the address or data byte received on the bus to be read by software.
When read, this register reflects the last byte which has been transferred on the bus. Reading this register is equivalent to reading the shift register of the interface.
When written, the contents of this register will be transferred into the shift register of the interface.

STATUS REGISTER 2 (I2CSTR2)
R244 - Read/Write (Bit 7:6),Read Only (Bit 5:0)
Register Page: 44
Reset Value: 0000 0000(00h)

7							0
ISCEN	SFEN	SCLIN	SDAIN	INT	UNPROC	UNEXP	MISP

Bit 7 = **ISCEN** *Interrupt on Stop Condition Enable bit*
This bit determines if an interrupt is generated as soon as a stop condition has been detected on the bus.
0: No interrupt generated on a bus stop condition
1: An interrupt is generated on a bus stop condition.

Note: When the interface is involved in a transaction, checking the ERROR status flag related to the error detection allows to determine if the transaction has been successfully completed. This interrupt can be useful for an interface waiting for a "bus free" condition in order to become a master as soon as possible. Checking the ACTIVE bit (in the I2CSTR1 register) allows to correctly identify an interrupt generated by a stop condition.

Bit 6 = **SFEN** *Spike Filter Enable bit*
This bit enables or disables the spike filters on the SDAx and SCLx inputs (x is 1 to 4).
0: spike filters disabled
1: spike filters enabled

Note: The length of a pulse identified as a spike depends on the CPUCLK frequency used (CPUCLK frequencies from 10 to 20 Mhz allow to filter pulses smaller than 100 to 40 ns).

Bit 5 = **SCLIN** *SCL Input status bit*
This read-only bit describes the current logic state on the SCL bus.
It can be used to sample the signal on a newly selected SCL bus for a quick determination concerning the bus use and the bus clock frequency.

Bit 4 = **SDAIN** *SDA Input status bit*
This read-only bit describes the current logic state on the SDA bus.
It can be used to sample the signal on a newly selected SDA bus for a quick determination of the state of this bus, prior starting a transaction.

Bit 3 = **INT** *Interrupt status bit*

This (read-only) bit indicates if an event has occurred.
0: No interrupt requested or an interrupt resulting from a stop condition occurred.
1: The interface enters an interrupt state resulting from any error (bus error or arbitration loss) or any byte transfer completed.

Bit 2 = **UNPROC** *Unprocessed flag bit*
This bit is useful in a multimaster mode system, to solve conflicts between a "Repeated Start" or a "Stop" condition and any bit of an address or data byte from other concurrent masters.
0: No error occurred.
1: A master interface tried to generate a "Repeated Start" or a "Stop" condition, which never occurred.

Note: If this bit is set, it will automatically activate the ERROR bit.

Note: This bit is only valid when the Advanced Features Enable bit AFEN is set in the I2CCTR register.

I²C BUS INTERFACE (Cont'd)

Bit 1 = **UNEXP** *Unexpected flag bit*
This bit is useful for error detection in a multimaster mode system, when a master is continuing its transaction while an other concurrent master wants to finish or restart a transaction by sending a "Start" or a "Stop" condition.
Together with the MISP bit, it covers all possible cases, where unexpected "Start" or "Stop" conditions occur, while the interface is a master.
0: No Unexpected error detected
1: A master interface receives a "Start" or a "Stop" condition, while sending the first bit of a data byte.

Notes:

– If this bit is set, it will automatically activate the ERROR bit.

– This bit is only valid when the Advanced Features Enable bit AFEN is set.

Bit 0 = **MISP** *Misplaced flag bit*
This bit indicates if the interface has received a misplaced "Start" or "Stop" condition during address transfer or any data byte transfer (besides first data bit). This error detection is also activated during the acknowledge bit transfer.
Together with the UNEXP bit, it covers all possible cases, where unexpected "Start" or "Stop" conditions occur, while the interface is a master.
0: No misplaced "start" or "stop" condition has been detected
1: A misplaced "Start" or "Stop" condition has been received.

Note: If this bit is set, it will automatically activate the ERROR bit.

STATUS REGISTER 1 (I2CSTR1)
R245 - Read Only
Register Page: 44
Reset Value: 0000 0000(00h)

7							0
ERROR	ARB_ LOST	READ	FIRST	GEN_ CALL	ACK_ BIT	ACT_ SLV	ACTIVE

Bit 7 = **ERROR** *ERROR detection bit*
This bit indicates if an error occurred on the bus or not.
0: No error detected
1: An error is detected. It is an illegal start or stop condition, i.e. a signal level transition occurs on the SDA bus during presence of a clock pulse on the SCL bus.
An interrupt is generated in this case. The interface stays in the error state until the error flag is reset by either a CLEAR operation, a STOP request or a "Return To Inactive State" operation.

Note: the ERROR bit has higher priority than the ARB_LOST bit (i.e. when ERROR=1, the value of ARB_LOST has to be ignored).

I²C BUS INTERFACE (Cont'd)

Bit 6 = **ARB_LOST** *Arbitration LOST detection bit*
This bit indicates if an arbitration lost occurred on the bus.
0: No arbitration lost occurred
1: An arbitration lost occurred. The bit is set when the interface operating as a master loses arbitration to another master on the bus.
If a loss of arbitration occurs during the address byte and if the interface has been addressed by the winning master (ACT_SLV=1), then the ARB_LOST flag is cleared by any "data load" operation into I2CDR.
In all other cases it is up to the user to return the interface into the status of an inactive slave via either a CLEAR operation, a "Return To Inactive State" operation or a STOP request.
If a loss of arbitration occurs, an interrupt is generated: when occurring during the address byte, the interrupt is generated at the end of the acknowledge bit; when occurring during a data byte, the interrupt is generated immediately.

Note: the ERROR bit has higher priority than the ARB_LOST bit (i.e. when ERROR=1, the value of ARB_LOST has to be ignored).

Bit 5 = **READ** *Read/write status bit*
This flag represents the state of the read/write bit of the address byte. It is updated either for a master or an active slave after the end of the address byte. It is cleared, when the interface returns to the inactive slave status (i.e. after the normal completion of a transaction, when exiting from any error state, ...).
0: Write operation
1: Read operation

Bit 4 = **FIRST** *transmission status bit*
This bit indicates if the byte transmitted on the bus is an address byte or a data byte.
0: The byte is a data byte
1: The byte is the address part of an I²C bus transaction.

Note: the FIRST bit is automatically cleared at the end of the interrupt, after the address, and when the interface returns into inactive slave state.

Bit 3 = **GEN_CALL** *General CALL status bit*
This bit indicates if a general call has been detected on the bus.
This bit is updated only if GENC_ACK=0 (see I2CCTR register for more details)
0: No general call detected, or GENC_ACK=1.
1: The general call address 00h has been recognized by the slave.

Note: This bit is cleared by hardware when the interface returns to the inactive slave status.

Bit 2 = **ACK_BIT** *Acknowledge BIT*
This bit reflects the logic level of the acknowledge bit detected at the end of the last byte (either address or data) transmitted on the I²C bus. It remains valid until the interface exits from the interrupt state.
0: Acknowledge detected
1: No acknowledge detected

Bit 1 = **ACT_SLV** *Active Slave status bit*
This bit indicates the slave status of the interface.
0: The interface is not working in slave mode. It may be inactive or in master mode (see the ACTIVE bit for more details).
1: The address assigned to the interface has been received on the bus and has been acknowledged by the interface (SEND_ACK=0).

Note: This bit is cleared when the interface returns into inactive slave state.

Bit 0 = **ACTIVE** *Interface Activity status bit*
This bit indicates if the interface is active or not.
0: The I²C interface is inactive
1: The Interface is active. The bit is set throughout the interval between a start condition and the first stop condition that follows on the I²C bus.

Note: It is reset by the CLEAR bit.

I²C BUS INTERFACE (Cont'd)

Table 37. I2C Interface Register Map and Reset Values

Address	Register Name	7	6	5	4	3	2	1	0
240	I2COAR Reset Value	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	ADR0
241	I2CFQR Reset Value	BUS_S0	BUS_S1	FMEN	PP_DRV	Q3	Q2	Q1	Q0
242	I2CCTR Reset Value	AFEN	RTI	GENC_ACK	SEND_ACK	MONITOR	RSRT	STOP	CLEAR
243	I2CDR Reset Value	SR8	SR7	SR6	SR5	SR4	SR3	SR2	SR1
244	I2CSTR2 Reset Value	ISCEN	SFEN	SCLIN	SDAIN	INT	UNPROC	UNEXP	MISP
245	I2CSTR1 Reset Value	ERROR	ARB_LOST	READ	FIRST	GEN_ CALL	ACK_BIT	ACT_SLV	ACTIVE

8.10 SERIAL PERIPHERAL INTERFACE (SPI)

8.10.1 Introduction

The Serial Peripheral Interface (SPI) is a general purpose on-chip shift register peripheral. It allows communication with external peripherals via an SPI protocol bus.

In addition, special operating modes allow reduced software overhead when implementing I^2C-bus and IM-bus communication standards.

The SPI uses up to 3 pins: Serial Data In (SDI), Serial Data Out (SDO) and Synchronous Serial Clock (SCK). Additional I/O pins may act as device selects or IM-bus address identifier signals.

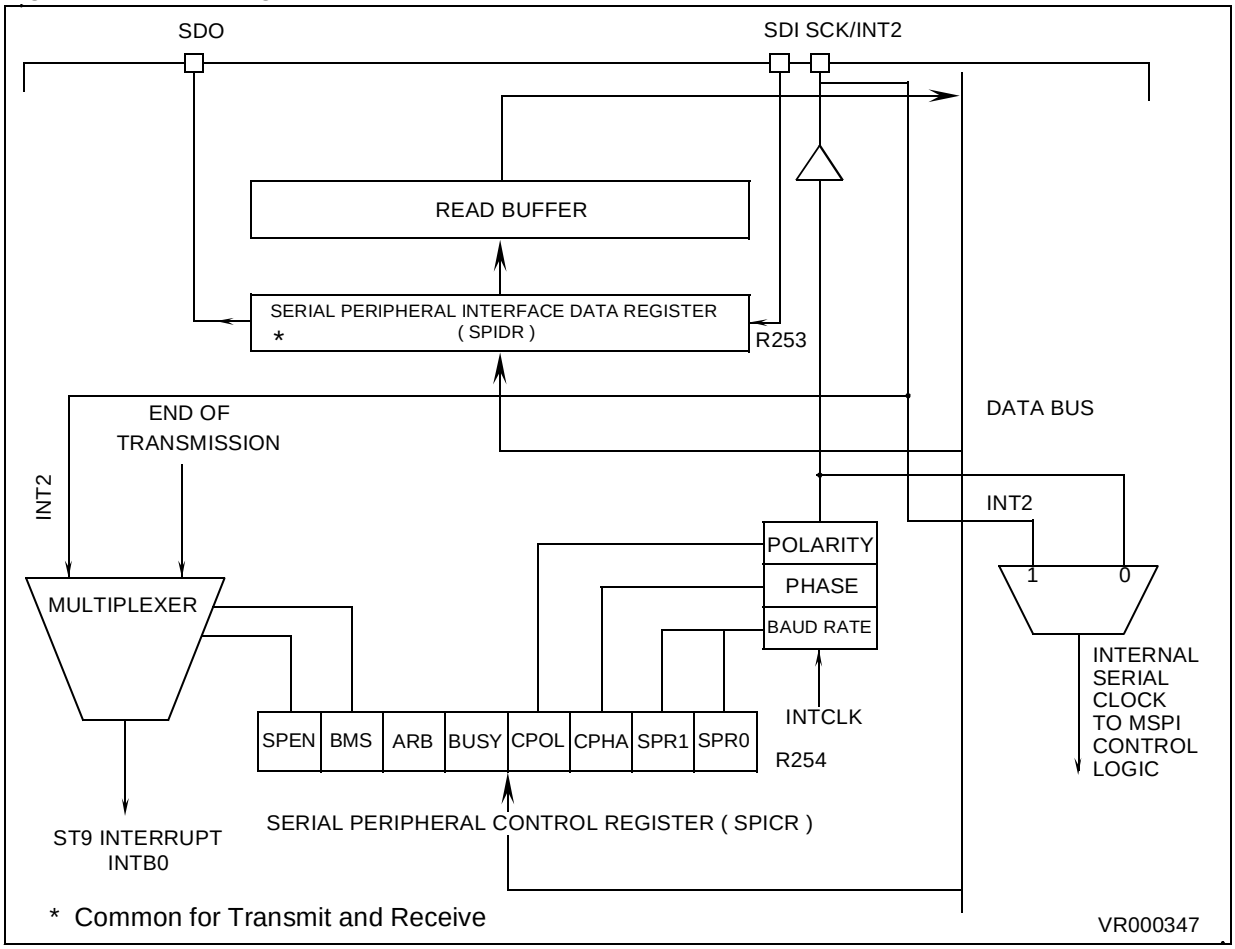
The main features are:

- Full duplex synchronous transfer if 3 I/O pins are used
- Master operation only
- 4 Programmable bit rates
- Programmable clock polarity and phase
- Busy Flag
- End of transmission interrupt
- Additional hardware to facilitate more complex protocols

8.10.2 Device-Specific Options

Depending on the ST9 variant and package type, the SPI interface signals may not be connected to separate external pins. Refer to the Peripheral Configuration Chapter for the device pin-out.

Figure 71. Block Diagram



SERIAL PERIPHERAL INTERFACE (Cont'd)

8.10.3 Functional Description

The SPI, when enabled, receives input data from the internal data bus to the SPI Data Register (SPIDR). A Serial Clock (SCK) is generated by controlling through software two bits in the SPI Control Register (SPICR). The data is parallel loaded into the 8 bit shift register during a write cycle. This is shifted out serially via the SDO pin, MSB first, to the slave device, which responds by sending its data to the master device via the SDI pin. This implies full duplex transmission if 3 I/O pins are used with both the data-out and data-in synchronized with the same clock signal, SCK. Thus the transmitted byte is replaced by the received byte, eliminating the need for separate "Tx empty" and "Rx full" status bits.

When the shift register is loaded, data is parallel transferred to the read buffer and becomes available to the CPU during a subsequent read cycle.

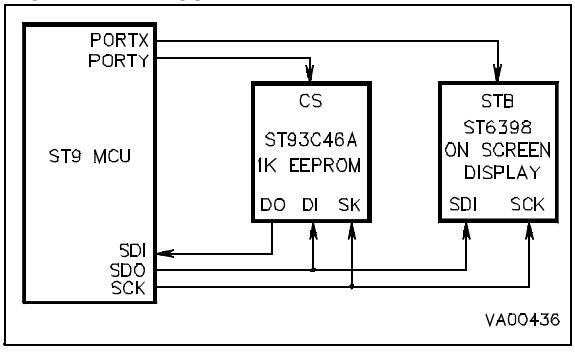
The SPI requires three I/O port pins:

SCK	Serial Clock signal
SDO	Serial Data Out
SDI	Serial Data In

An additional I/O port output bit may be used as a slave chip select signal. Data and Clock pins I²C Bus protocol are open-drain to allow arbitration and multiplexing.

Figure 72 below shows a typical SPI network.

Figure 72. A Typical SPI Network



8.10.3.1 Input Signal Description

Serial Data In (SDI)

Data is transferred serially from a slave to a master on this line, most significant bit first. In an S-BUS/I^2C-bus configuration, the SDI line senses the value forced on the data line (by SDO or by another peripheral connected to the S-bus/I^2C-bus).

8.10.3.2 Output Signal Description

Serial Data Out (SDO)

The SDO pin is configured as an output for the master device. This is obtained by programming the corresponding I/O pin as an output alternate function. Data is transferred serially from a master to a slave on SDO, most significant bit first. The master device always allows data to be applied on the SDO line one half cycle before the clock edge, in order to latch the data for the slave device. The SDO pin is forced to high impedance when the SPI is disabled.

During an S-Bus or I^2C-Bus protocol, when arbitration is lost, SDO is set to one (thus not driving the line, as SDO is configured as an open drain).

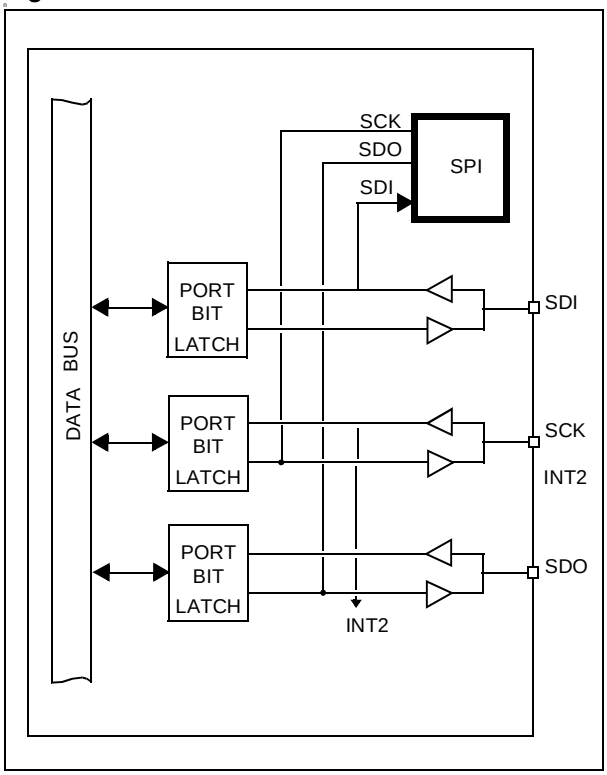
Master Serial Clock (SCK)

The master device uses SCK to latch the incoming data on the SDI line. This pin is forced to a high impedance state when SPI is disabled (SPEN, SPICR.7 = "0"), in order to avoid clock contention from different masters in a multi-master system. The master device generates the SCK clock from INTCLK. The SCK clock is used to synchronize data transfer, both in to and out of the device, through its SDI and SDO pins. The SCK clock type, and its relationship with data is controlled by the CPOL (Clock Polarity) and CPHA (Clock Phase) bits in the Serial Peripheral Control Register (SPICR). This input is provided with a digital filter which eliminates spikes lasting less than one INTCLK period.

Two bits, SPR1 and SPR0, in the Serial Peripheral Control Register (SPICR), select the clock rate. Four frequencies can be selected, two in the high frequency range (mostly used with the SPI protocol) and two in the medium frequency range (mostly used with more complex protocols).

SERIAL PERIPHERAL INTERFACE (Cont'd)

Figure 73. SPI I/O Pins



8.10.4 Interrupt Structure

The SPI peripheral is associated with external interrupt channel B0 (pin INT2). Multiplexing between the external pin and the SPI internal source is controlled by the SPEN and BMS bits, as shown in Table 38.

The two possible SPI interrupt sources are:

– End of transmission (after each byte).

– S-bus/I^2C-bus start or stop condition.

Care should be taken when toggling the SPEN and/or BMS bits from the "0,0" condition. Before changing the interrupt source from the external pin to the internal function, the B0 interrupt channel should be masked. EIMR.2 (External Interrupt Mask Register, bit 2, IMBO) and EIPR.2 (External Interrupt Pending Register bit 2, IMP0) should be "0" before changing the source. This sequence of events is to avoid the generating and reading of spurious interrupts.

A delay instruction lasting at least 4 clock cycles (e.g. 2 NOPs) should be inserted between the SPEN toggle instruction and the Interrupt Pending bit reset instruction.

The INT2 input Function is always mapped together with the SCK input Function, to allow Start/Stop bit detection when using S-bus/I^2C-bus protocols.

A start condition occurs when SDI goes from "1" to "0" and SCK is "1". The Stop condition occurs when SDI goes from "0" to "1" and SCK is "1". For both Stop and Start conditions, SPEN = "0" and BMS = "1".

Table 38. Interrupt Configuration

SPEN	BMS	Interrupt Source
0	0	External channel INT2
0	1	S-bus/I^2C bus start or stop condition
1	X	End of a byte transmission

SERIAL PERIPHERAL INTERFACE (Cont'd)

8.10.5 Working With Other Protocols

The SPI peripheral offers the following facilities for operation with S-bus/I^{2}C-bus and IM-bus protocols:

- Interrupt request on start/stop detection
- Hardware clock synchronisation
- Arbitration lost flag with an automatic set of data line

Note that the I/O bit associated with the SPI should be returned to a defined state as a normal I/O pin before changing the SPI protocol.

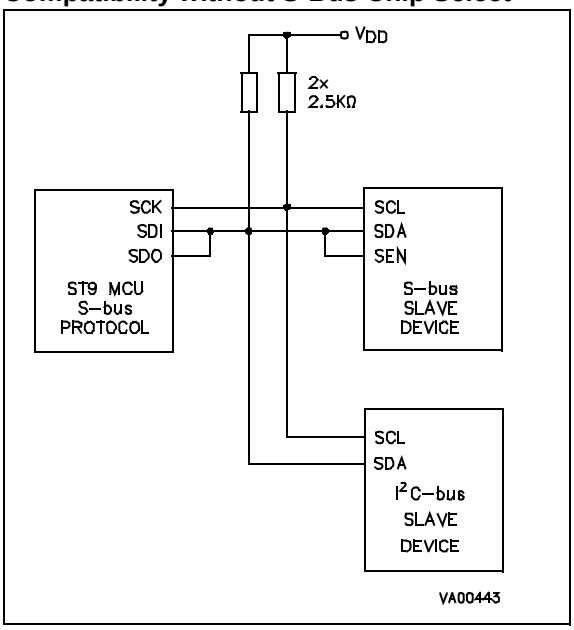
The following paragraphs provide information on how to manage these protocols.

8.10.6 I^{2}C-bus Interface

The I^{2}C-bus is a two-wire bidirectional data-bus, the two lines being SDA (Serial DAta) and SCL (Serial CLock). Both are open drain lines, to allow arbitration. As shown in Figure 75, data is toggled with clock low. An I²C bus start condition is the transition on SDI from 1 to 0 with the SCK held high. In a stop condition, the SCK is also high and the transition on SDI is from 0 to 1. During both of these conditions, if SPEN = 0 and BMS = 1 then an interrupt request is performed.

Each transmission consists of nine clock pulses (SCL line). The first 8 pulses transmit the byte (MSB first), the ninth pulse is used by the receiver to acknowledge.

Figure 74. S-Bus / I^{2}C-bus Peripheral Compatibility without S-Bus Chip Select

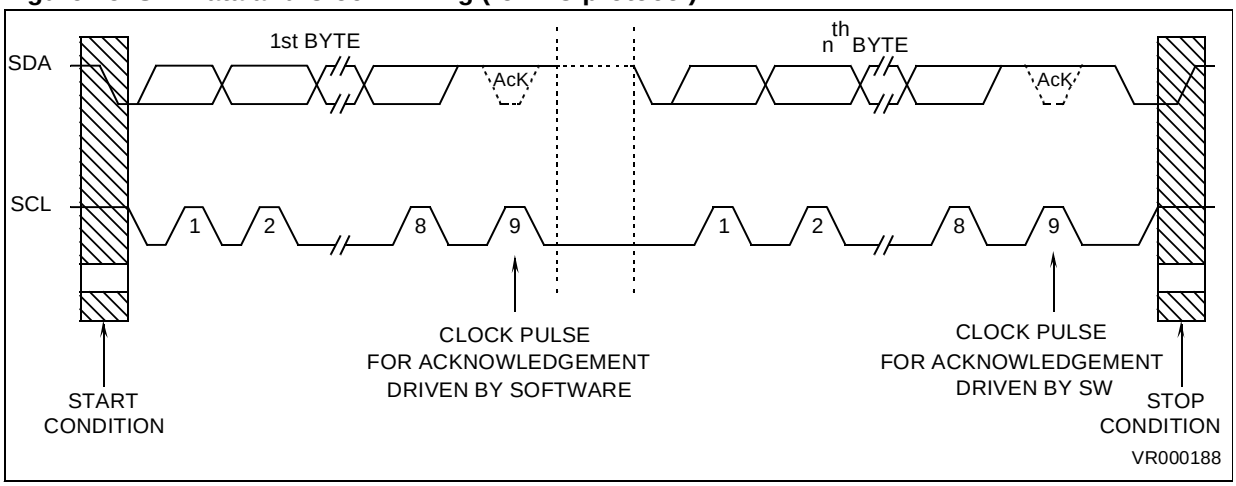


SERIAL PERIPHERAL INTERFACE (Cont'd)

Table 39. Typical I^2C-bus Sequences

Phase	Software	Hardware	Notes
INITIALIZE	SPICR.CPOL, CPHA = 0, 0 SPICR.SPEN = 0 SPICR.BMS = 1 SCK pin set as AF output SDI pin set as input Set SDO port bit to 1	SCK, SDO in HI-Z SCL, SDA = 1, 1	Set polarity and phase SPI disable START/STOP interrupt Enable
START	SDO pin set as output Open Drain Set SDO port bit to 0	SDA = 0, SCL = 1 interrupt request	START condition receiver START detection
TRANSMISSION	SPICR.SPEN = 1 SDO pin as Alternate Function output load data into SPIDR	SCL = 0 Start transmission Interrupt request at end of byte transmission	Managed by interrupt routine load FFh when receiving end of transmission detection
ACKNOWLEDGE	SPICR.SPEN = 0 Poll SDA line Set SDA line SPICR.SPEN = 1	SCK, SDO in HI-Z SCL, SDA = 1 SCL = 0	SPI disable only if transmitting only if receiving only if transmitting
STOP	SDO pin set as output Open Drain SPICR.SPEN = 0 Set SDO port bit to 1	SDA = 1 interrupt request	STOP condition

Figure 75. SPI Data and Clock Timing (for I2C protocol)



SERIAL PERIPHERAL INTERFACE (Cont'd)

The data on the SDA line is sampled on the low to high transition of the SCL line.

SPI working with an I²C-bus

To use the SPI with the I²C-bus protocol, the SCK line is used as SCL; the SDI and SDO lines, externally wire-ORed, are used as SDA. All output pins must be configured as open drain (see Figure 74).

Figure 39 illustrates the typical I²C-bus sequence, comprising 5 phases: Initialization, Start, Transmission, Acknowledge and Stop. It should be noted that only the first 8 bits are handled by the SPI peripheral; the ACKNOWLEDGE bit must be managed by software, by polling or forcing the SCL and SDO lines via the corresponding I/O port bits.

During the transmission phase, the following I²C-bus features are also supported by hardware.

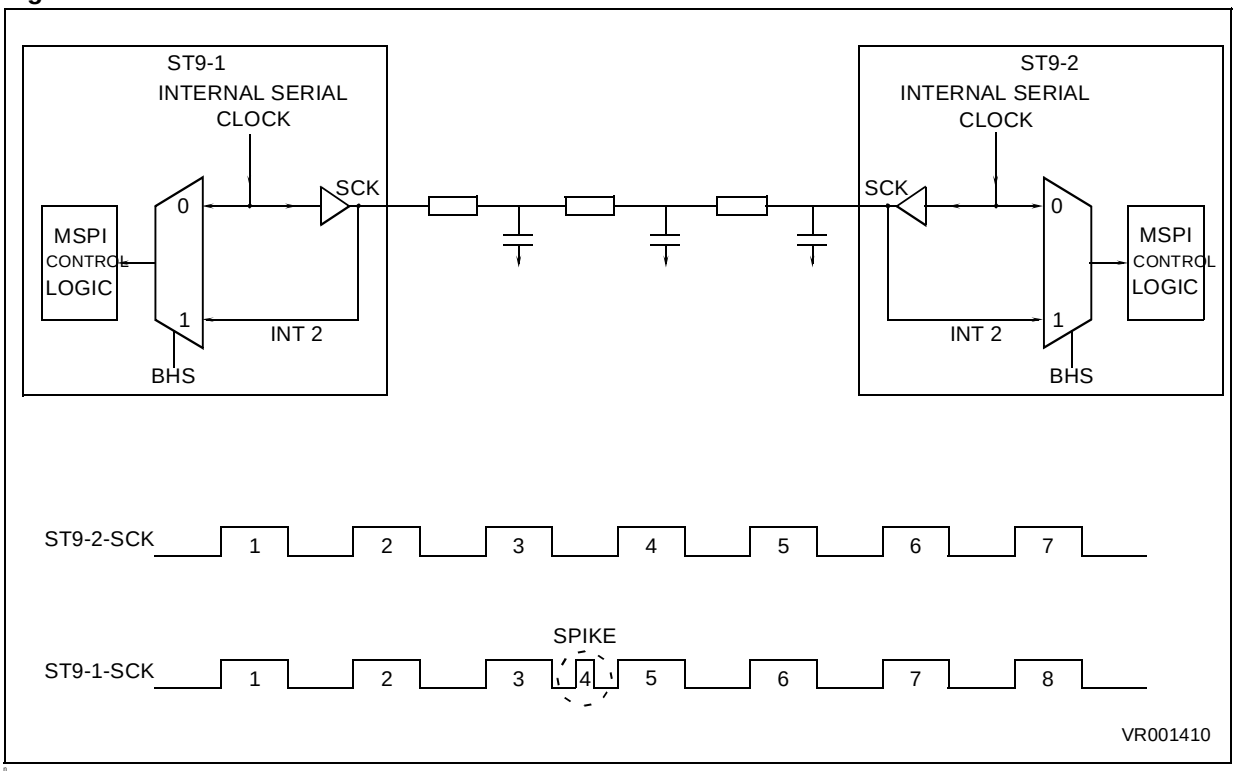
Clock Synchronization

In a multimaster I²C-bus system, when several masters generate their own clock, synchronization is required. The first master which releases the SCL line stops internal counting, restarting only when the SCL line goes high (released by all the other masters). In this manner, devices using different clock sources and different frequencies can be interfaced.

Arbitration Lost

When several masters are sending data on the SDA line, the following takes place: if the transmitter sends a "1" and the SDA line is forced low by another device, the ARB flag (SPICR.5) is set and the SDO buffer is disabled (ARB is reset and the SDO buffer is enabled when SPIDR is written to again). When BMS is set, the peripheral clock is supplied through the INT2 line by the external clock line (SCL). Due to potential noise spikes (which must last longer than one INTCLK period to be detected), RX or TX may gain a clock pulse. Referring to Figure 76, if device ST9-1 detects a noise spike and therefore gains a clock pulse, it will stop its transmission early and hold the clock line low, causing device ST9-2 to freeze on the 7th bit. To exit and recover from this condition, the BMS bit must be reset; this will cause the SPI logic to be reset, thus aborting the current transmission. An End of Transmission interrupt is generated following this reset sequence.

Figure 76. SPI Arbitration



SERIAL PERIPHERAL INTERFACE (Cont'd)

8.10.7 S-Bus Interface

The S-bus is a three-wire bidirectional data-bus, possessing functional features similar to the I^2C-bus. As opposed to the I^2C-bus, the Start/Stop conditions are determined by encoding the information on 3 wires rather than on 2, as shown in Figure 78. The additional line is referred as SEN.

Figure 77. Mixed S-bus and I^2C-bus System

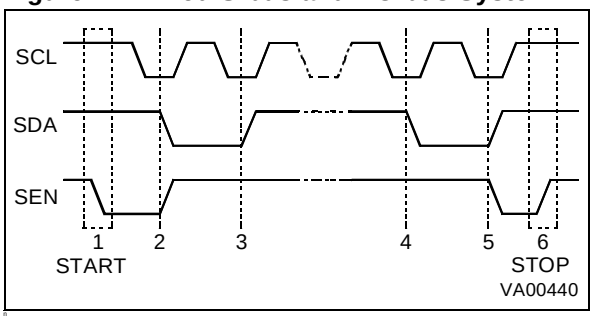
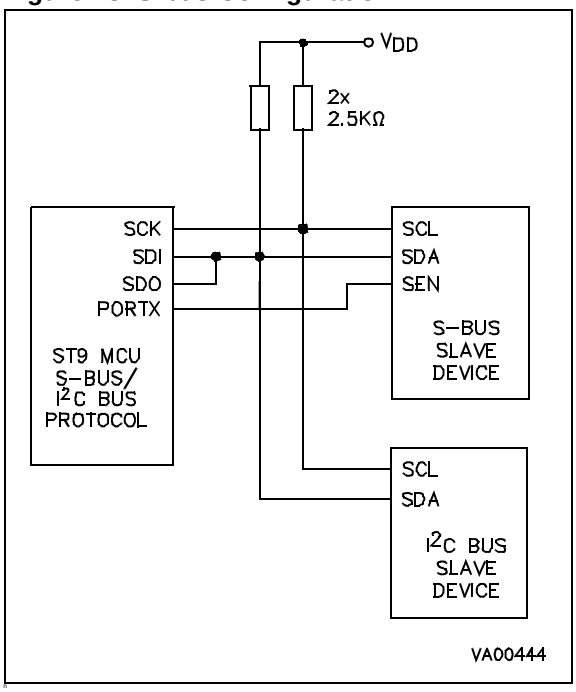


Figure 78. S-bus Configuration



SPI Working with S-bus

The S-bus protocol uses the same pin configuration as the I^2C-bus for generating the SCL and SDA lines. The additional SEN line is managed through a standard ST9 I/O port line, under software control (see Figure 74).

SERIAL PERIPHERAL INTERFACE (Cont'd)

8.10.8 IM-bus Interface

The IM-bus features a bidirectional data line and a clock line; in addition, it requires an IDENT line to distinguish an address byte from a data byte (Figure 80). Unlike the I^2C-bus protocol, the IM-bus protocol sends the least significant bit first; this requires a software routine which reverses the bit order before sending, and after receiving, a data byte. Figure 79 shows the connections between an IM-bus peripheral and an ST9 SPI. The SDO and SDI pins are connected to the bidirectional data pin of the peripheral device. The SDO alternate function is configured as Open-Drain (external 2.5KΩ pull-up resistors are required).

With this type of configuration, data is sent to the peripheral by writing the data byte to the SPIDR register. To receive data from the peripheral, the user should write FFh to the SPIDR register, in order to generate the shift clock pulses. As the SDO line is set to the Open-Drain configuration, the incoming data bits that are set to "1" do not affect the SDO/SDI line status (which defaults to a high level due to the FFh value in the transmit register), while incoming bits that are set to "0" pull the input line low.

In software it is necessary to initialise the ST9 SPI by setting both CPOL and CPHA to "1". By using a general purpose I/O as the IDENT line, and forcing it to a logical "0" when writing to the SPIDR register, an address is sent (or read). Then, by setting this bit to "1" and writing to SPIDR, data is sent to the peripheral. When all the address and data pairs are sent, it is necessary to drive the IDENT line low and high to create a short pulse. This will generate the stop condition.

Figure 79. ST9 and IM-bus Peripheral

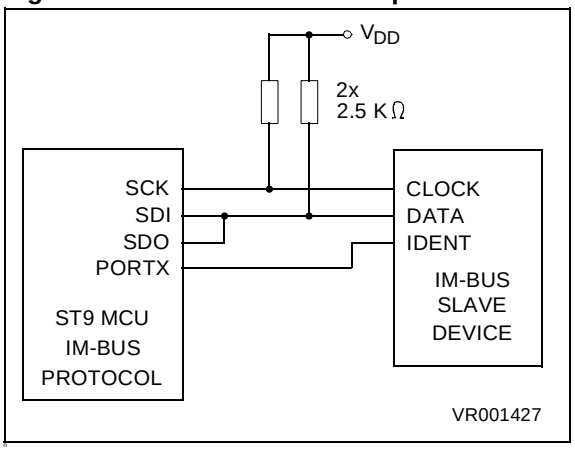
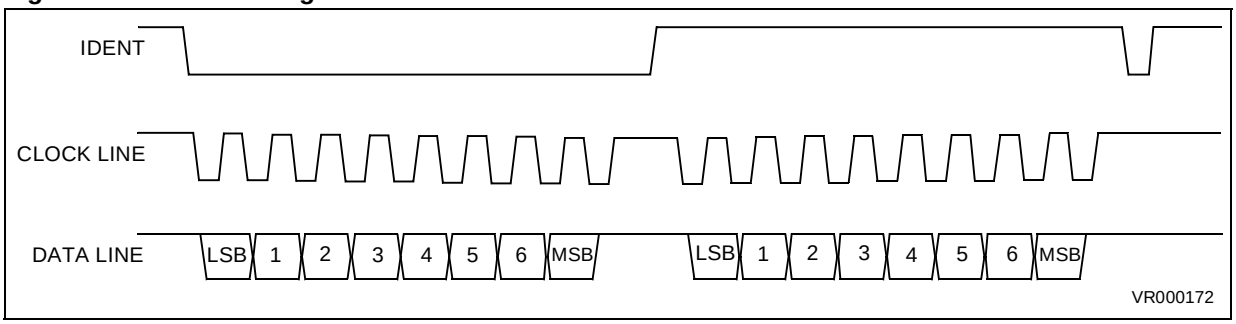


Figure 80. IM bus Timing



SERIAL PERIPHERAL INTERFACE (Cont'd)

8.10.9 Register Description

It is possible to have up to 3 independent SPIs in the same device (refer to the device block diagram). In this case they are named SPI0 thru SPI2. If the device has one SPI converter it uses the register adresses of SPI0. The register map is the following:

Register	SPIn	Page
SPIDR R253	SPI0	0
SPICR R254	SPI0	0
SPIDR1 R253	SPI1	7
SPICR1 R254	SPI1	7
SPIDR2 R245	SPI2	7
SPICR2 R246	SPI2	7

Note: In the register description on the following pages, register and page numbers are given using the example of SPI0.

SPI DATA REGISTER (SPIDR)
R253 - Read/Write
Register Page: 0
Reset Value: undefined

7							0
D7	D6	D5	D4	D3	D2	D1	D0

Bit 7:0 = **D[0:7]**: *SPI Data*.
This register contains the data transmitted and received by the SPI. Data is transmitted bit 7 first, and incoming data is received into bit 0. Transmission is started by writing to this register.

Note: SPIDR state remains undefined until the end of transmission of the first byte.

SPI CONTROL REGISTER (SPICR)
R254 **-** Read/Write
Register Page: 0
Reset Value: 0000 0000 (00h)

7							0
SPEN	BMS	ARB	BUSY	CPOL	CPHA	SPR1	SPR0

Bit 7 = **SPEN**: *Serial Peripheral Enable*.
0: SCK and SDO are kept tristate.
1: Both alternate functions SCK and SDO are enabled.

Note: furthermore, SPEN (together with the BMS bit) affects the selection of the source for interrupt channel B0. Transmission starts when data is written to the SPIDR Register.

Bit 6 = **BMS**: *S-bus/I^{2}C-bus Mode Selector*.
0: Perform a re-initialisation of the SPI logic, thus allowing recovery procedures after a RX/TX failure.
1: Enable S-bus/I^{2}C-bus arbitration, clock synchronization and Start/ Stop detection (SPI used in an S-bus/I^{2}C-bus protocol).

Note: when the BMS bit is reset, it affects (together with the SPEN bit) the selection of the source for interrupt channel B0.

Bit 5 = **ARB**: *Arbitration flag bit.*
This bit is set by hardware and can be reset by software.
0: S-bus/I^{2}C-bus stop condition is detected.
1: Arbitration lost by the SPI in S-bus/I^{2}C-bus mode.

Note: when ARB is set automatically, the SDO pin is set to a high value until a write instruction on SPIDR is performed.

Bit 4 = **BUSY**: *SPI Busy Flag*.
This bit is set by hardware. It allows the user to monitor the SPI status by polling its value.
0: No transmission in progress.
1: Transmission in progress.

Bit 3 = **CPOL**: *Transmission Clock Polarity*.
CPOL controls the normal or steady state value of the clock when data is *not* being transferred. Please refer to the following table and to Figure 81 to see this bit action (together with the CPHA bit).

Note: As the SCK line is held in a high impedance state when the SPI is disabled (SPEN = "0"), the SCK pin must be connected to V_{SS} or to V_{CC} through a resistor, depending on the CPOL state. Polarity should be set during the initialisation routine, in accordance with the setting of all peripherals, and should not be changed during program execution.

SERIAL PERIPHERAL INTERFACE (Cont'd)

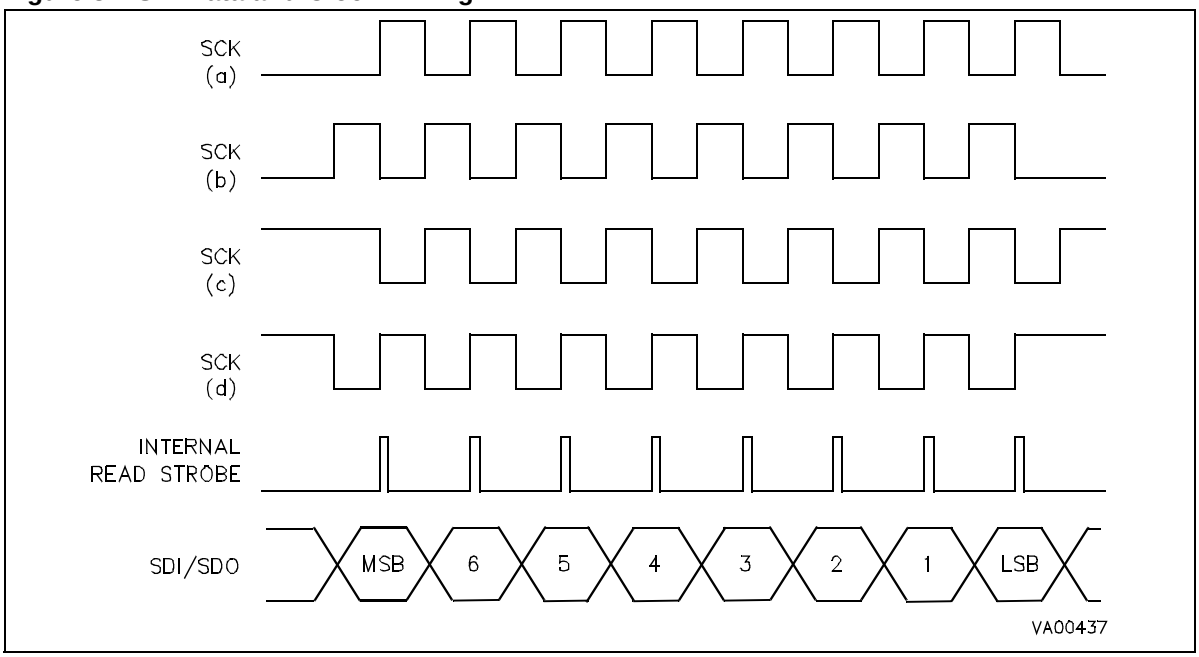
Bit 2 = **CPHA**: *Transmission Clock Phase.*
CPHA controls the relationship between the data on the SDI and SDO pins, and the clock signal on the SCK pin. The CPHA bit selects the clock edge used to capture data. It has its greatest impact on the first bit transmitted (MSB), because it does (or does not) allow a clock transition before the first data capture edge. Figure 81 shows the relationship between CPHA, CPOL and SCK, and indicates active clock edges and strobe times.

CPOL	CPHA	SCK (in Figure 81)
0	0	(a)
0	1	(b)
1	0	(c)
1	1	(d)

Bit 1:0 = **SPR[1:0]:** *SPI Rate.*
These two bits select one (of four) baud rates, to be used as SCK.

SPR1	SPR0	Clock Divider	SCK Frequency (@ INTCLK = 24MHz)	
0	0	8	3000kHz	(T = 0.33µs)
0	1	16	1500kHz	(T = 0.67µs)
1	0	128	187.5kHz	(T = 5.33µs)
1	1	256	93.75kHz	(T = 10.66µs)

Figure 81. SPI Data and Clock Timing



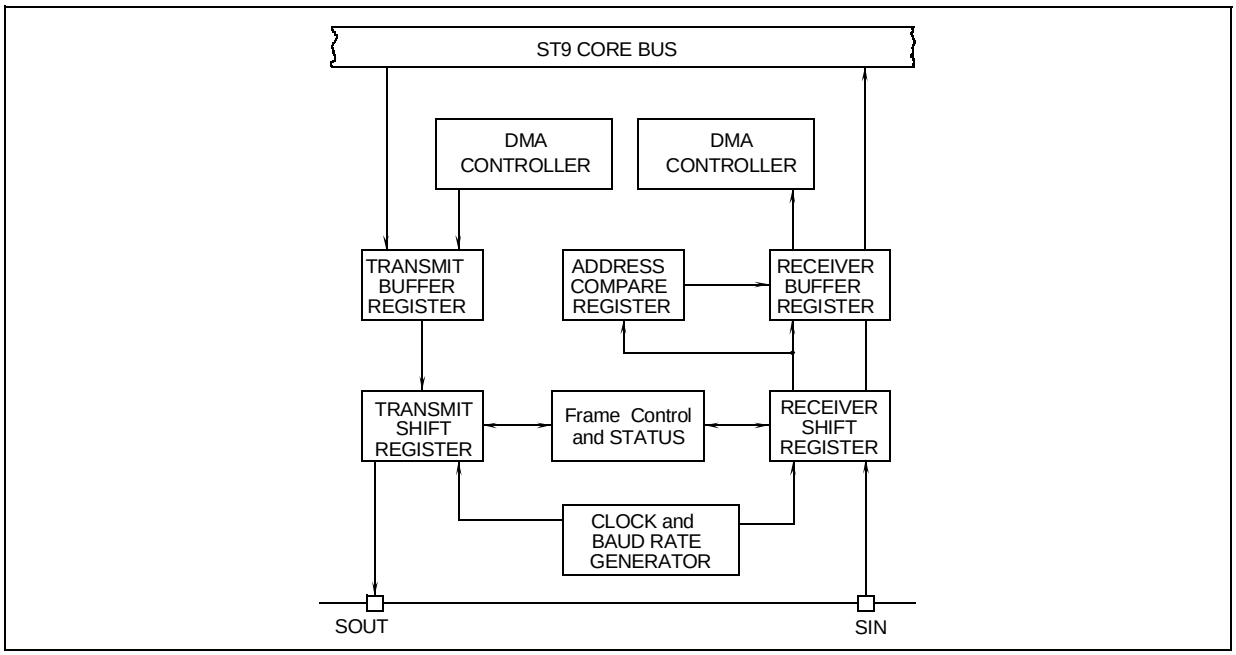
8.11 SERIAL COMMUNICATIONS INTERFACE (SCI)

8.11.1 Introduction

The Serial Communications Interface (SCI) offers full-duplex asynchronous serial data exchange for interfacing a wide range of external equipment. It has the following principal features:

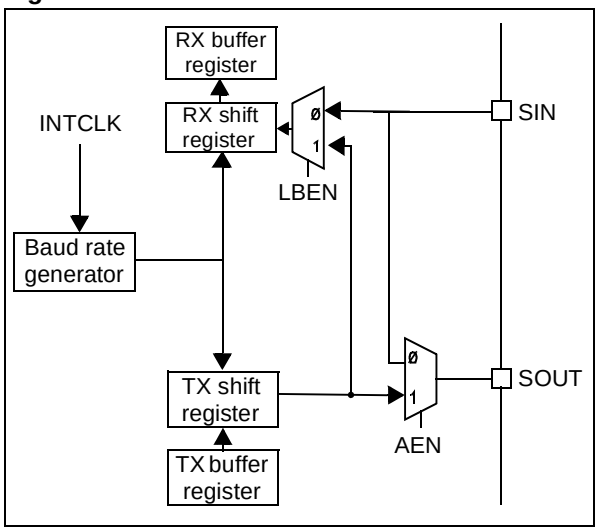
- Full duplex asynchronous operation.
- Transmit, receive, line status, and device address interrupt generation.
- Integral Baud Rate Generator capable of dividing the input clock by any value from 2 to 2^{16}-1 (16 bit word) and generating the internal 16X data sampling clock for asynchronous operation.
- Fully programmable serial interface:
  - 5, 6, 7, or 8 bit word length.
  - Even, odd, or no parity generation and detection.
  - 0, 1, 1-1/2, 2, 2-1/2, 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.

Figure 82. SCI Block Diagram



SERIAL COMMUNICATIONS INTERFACE (Cont'd)

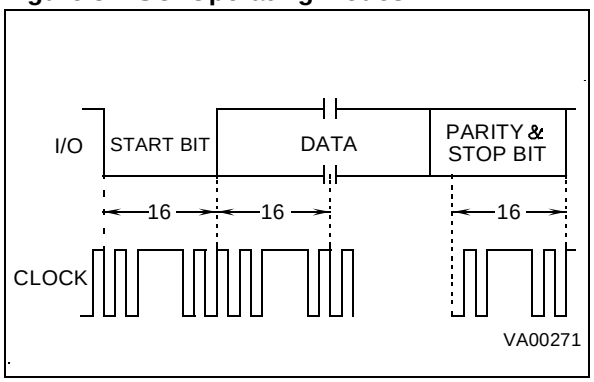
Figure 83. SCI Functional Schematic



8.11.2 SCI Operation

Each data bit is sampled 16 times per clock period.

Figure 84. SCI Operating Modes



8.11.3 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. The START condition is detected as a high to low transition.

DATA: the DATA word length is programmable from 5 to 8 bits. LSB are transmitted first.

PARITY: The Parity Bit 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 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. 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.

Figure 85. SCI Character Formats

	START	DATA(1)	PARITY	ADDRESS	STOP
# bits	0, 1	5, 6, 7, 8	0, 1	0, 1	0, 1, 1.5, 2, 2.5
states			NONE ODD EVEN	ON OFF	

(1) LSB First

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

8.11.3.1 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 uses INTCLK as the clock source.

The Address bit/D9 is optional and may be added to any word. 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).

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.

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.

Table 40. Address Interrupt Modes

If 9th Data Bit is set
If Character Match
If Character Match and 9th Data Bit is set
If Character Match Immediately Follows BREAK

Figure 86. Auto Echo Configuration

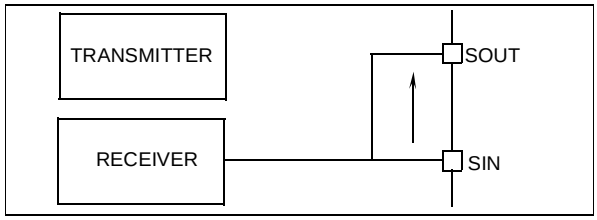


Figure 87. Loop Back Configuration

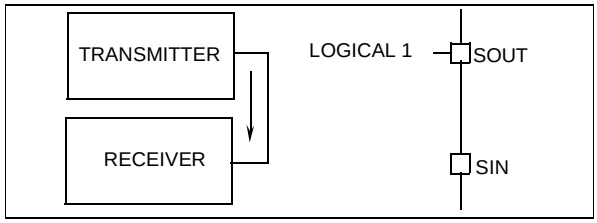
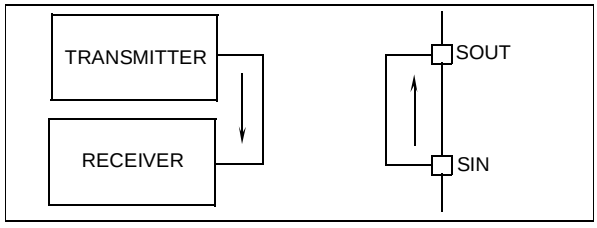


Figure 88. Auto Echo and Loop-Back Configuration



SERIAL COMMUNICATIONS INTERFACE (Cont'd)

8.11.4 Clocks And Serial Transmission Rates

The communication bit rate of the SCI transmitter and receiver sections is provided from the internal Baud Rate Generator divided by 16. x

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.

With INTCLK running at 20MHz or 10MHz, a maximum bit rate of 625 KBaud or 312.5K Baud respectively is possible.

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. 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 41.

8.11.5 SCI 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.

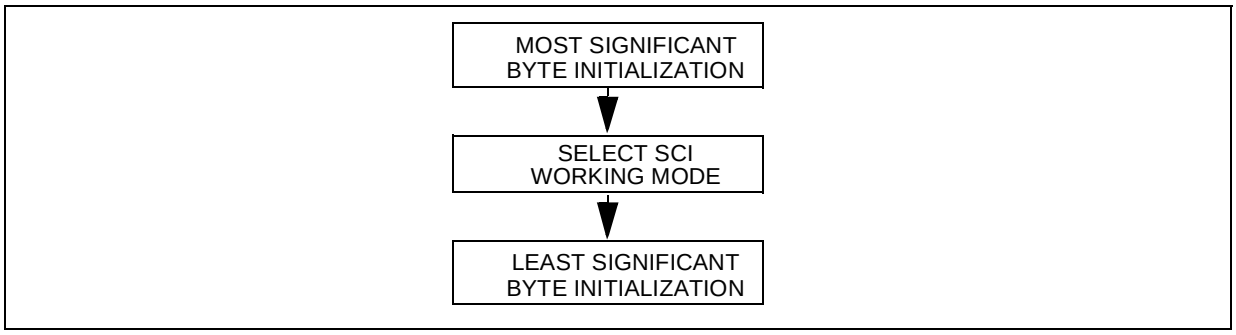
'On-the-Fly' modifications of the control registers' content during transmitter/receiver operations, although possible, can corrupt data and produce undesirable spikes on the I/O lines. Furthermore, modifying the control registers' content without reinitialising the SCI circuitry (during stand-by cycles, waiting to transmit or receive data) must be kept carefully under control by software to avoid spurious data being transmitted or received.

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

Table 41. Practical Example of SCI Baud Rate Generator Divider Values

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%

Figure 89. SCI Baud Rate Generator Initialization Sequence



SERIAL COMMUNICATIONS INTERFACE (Cont'd)

8.11.6 Input Signals

SIN: Serial Data Input. This pin is the serial data input to the SCI receiver shift register.

8.11.7 Output Signals

SOUT: Serial Data Output. This Alternate Function output signal is the serial data output for the SCI transmitter in all operating modes.

8.11.8 Interrupts and DMA

8.11.8.1 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 90.

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 8.11.2 for more details relating to Synchronous mode.

Table 42. SCI Interrupt Internal Priority

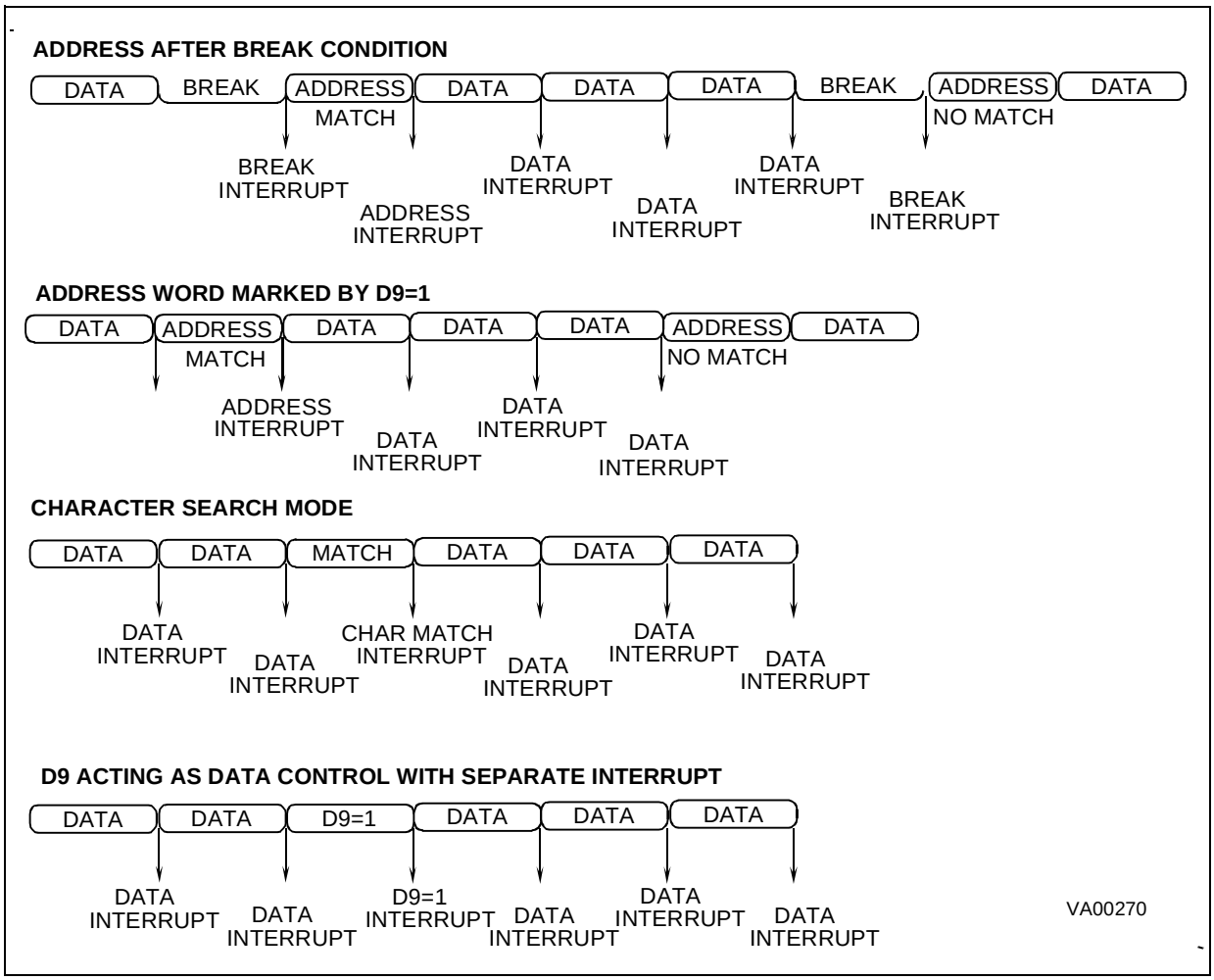
Receive DMA Request	Highest Priority
Transmit DMA Request	
Receive Interrupt	
Transmit Interrupt	Lowest Priority

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

Table 43. SCI 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 90. SCI Interrupts: Example of Typical Usage



SERIAL COMMUNICATIONS INTERFACE (Cont'd)

8.11.8.2 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.

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

8.11.9 Register Description

The SCI registers are located in the following pages in the ST9:

SCI number 0: page 24 (18h)

SCI 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)	Reserved
R255 (FFh)	Reserved

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.

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.

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.

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).

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.

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

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
0	0	1
0	1	1.5
1	0	2
1	1	2.5

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

CLOCK CONFIGURATION REGISTER (CCR)

R251 - Read/Write

Reset value: 0000 0000 (00h)

7							0
-	-	-	-	-	AEN	LBEN	STPEN

Bit 7:3 = Reserved. Must be left at reset value.

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

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.

INPUT CONTROL (SICR)

R254 - Read/Write

Reset value: 0000 0011 (03h)

7							0
-	INPL	-	-	-	INPEN	-	-

Bit 7 = Reserved.

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 5:3 = Reserved.

Bit 2 = **INPEN**: *Input Disable*.
0: Enable SIN input
1: Disable SIN input

Bit 1:0 = Reserved.

OUTPUT CONTROL (SOCR)

R255 - Read/Write

Reset value: 0000 0001 (01h)

7							0
OUTPL	OUTSB	-	-	-	-	-	-

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 5:0 = Reserved.

8.12 VOLTAGE SYNTHESIS TUNING CONVERTER (VS)

8.12.1 Description

The on-chip Voltage Synthesis (VS) converter allows the generation of a tuning reference voltage in a TV set application. The peripheral is composed of a 14-bit counter that allows the conversion of the digital content in a tuning voltage, available at the VS output pin, by using PWM (Pulse Width Modulation) and BRM (Bit Rate Modulation) techniques. The 14-bit counter gives 16384 steps which allow a resolution of approximately 2 mV over a tuning voltage of 32 V. This corresponds to a tuning resolution of about 40 KHz per step in UHF band (the actual value will depend on the characteristics of the tuner).

The tuning word consists of a 14-bit word contained in the registers VSDR1 (R254) and VSDR2 (R255) both located in page 59.

Coarse tuning (PWM) is performed using the seven most significant bits. Fine tuning (BRM) is performed using the the seven least significant bits. With all “0”s loaded, the output is 0. As the tuning voltage increases from all “0”s, the number of pulses in one period increases to 128 with all pulses being the same width. For values larger than 128, the PWM takes over and the number of pulses in one period remains constant at 128, but the width changes. At the other end of the scale, when almost all “1”s are loaded, the pulses will start to link together and the number of pulses will decrease.

When all “1”s are loaded, the output will be almost 100% high but will have a low pulse (1/16384 of the high pulse).

8.12.2 Output Waveforms

Included inside the VS are the register latches, a reference counter, PWM and BRM control circuitry. The clock for the 14-bit reference counter is derived from the main system clock (referred to as INTCLK) after a division by 4. For example, using an internal 12 MHz on-chip clock (see Timing & Clock Controller chapter) leads to a 3 MHz input for the VS counter.

From the point of view of the circuit, the seven most significant bits control the coarse tuning, while the seven least significant bits control the fine tuning. From the application and software point of view, the 14 bits can be considered as one binary number.

As already mentioned the coarse tuning consists of a PWM signal with 128 steps: we can consider the fine tuning to cover 128 coarse tuning cycles.

The VS Tuning Converter is implemented with 2 separate outputs (VSO1 and VSO2) that can drive 2 separate Alternate Function outputs of 2 standard I/O port bits. A control bit allows you to choose which output is activated (only one output can be activated at a time).

When a VS output is not selected because the VS is disabled or because the second output is selected, it stays at a logical “one” level, allowing you to use the corresponding I/O port bit either as a normal I/O port bit or for a possible second Alternate Function output.

A second control bit allows the VS function to be started (or stopped) by software.

VOLTAGE SYNTHESIS (Cont'd)

PWM Generation

The counter increments continuously, clocked at INTCLK divided by 4. Whenever the 7 least significant bits of the counter overflow, the VS output is set.

The state of the PWM counter is continuously compared to the value programmed in the 7 most significant bits of the tuning word. When a match occurs, the output is reset thus generating the PWM output signal on the VS pin.

This Pulse Width modulated signal must be filtered, using an external RC network placed as close as possible to the associated pin. This provides an analog voltage proportional to the average charge passed to the external capacitor. Thus for a higher mark/space ratio (High time much greater than Low time) the average output voltage is higher. The external components of the RC network should be selected for the filtering level required for control of the system variable.

Figure 91. Typical PWM Output Filter

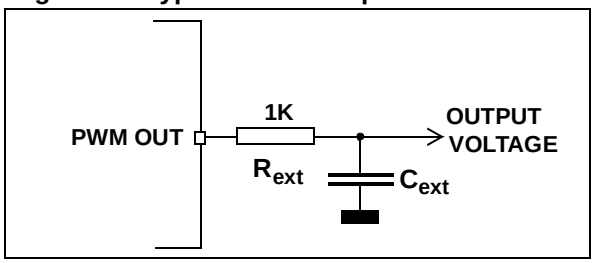
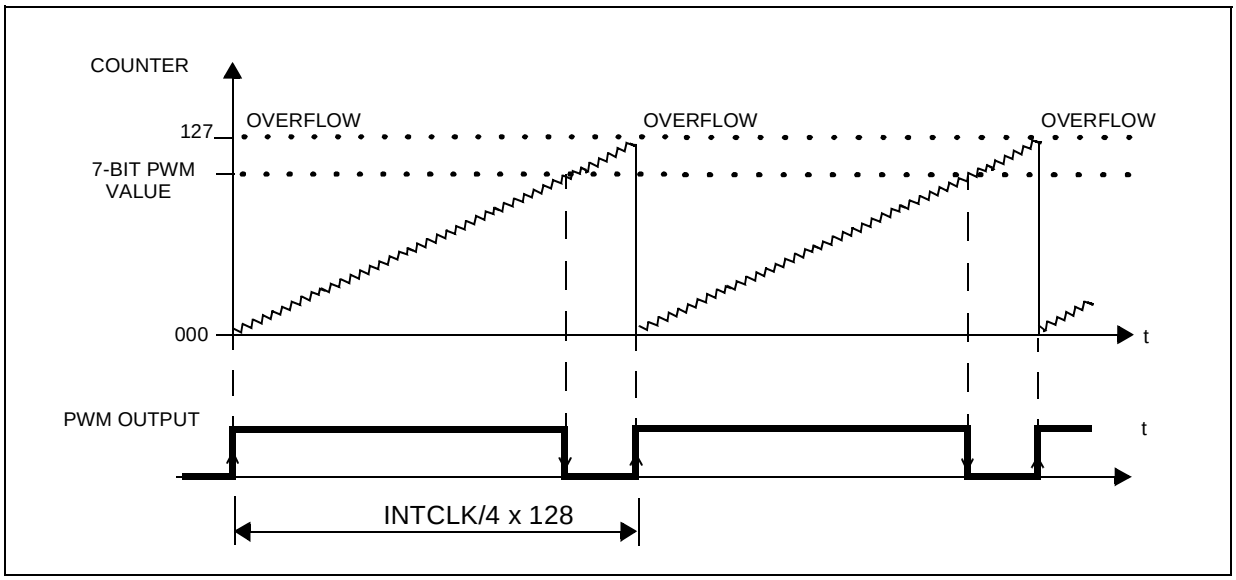
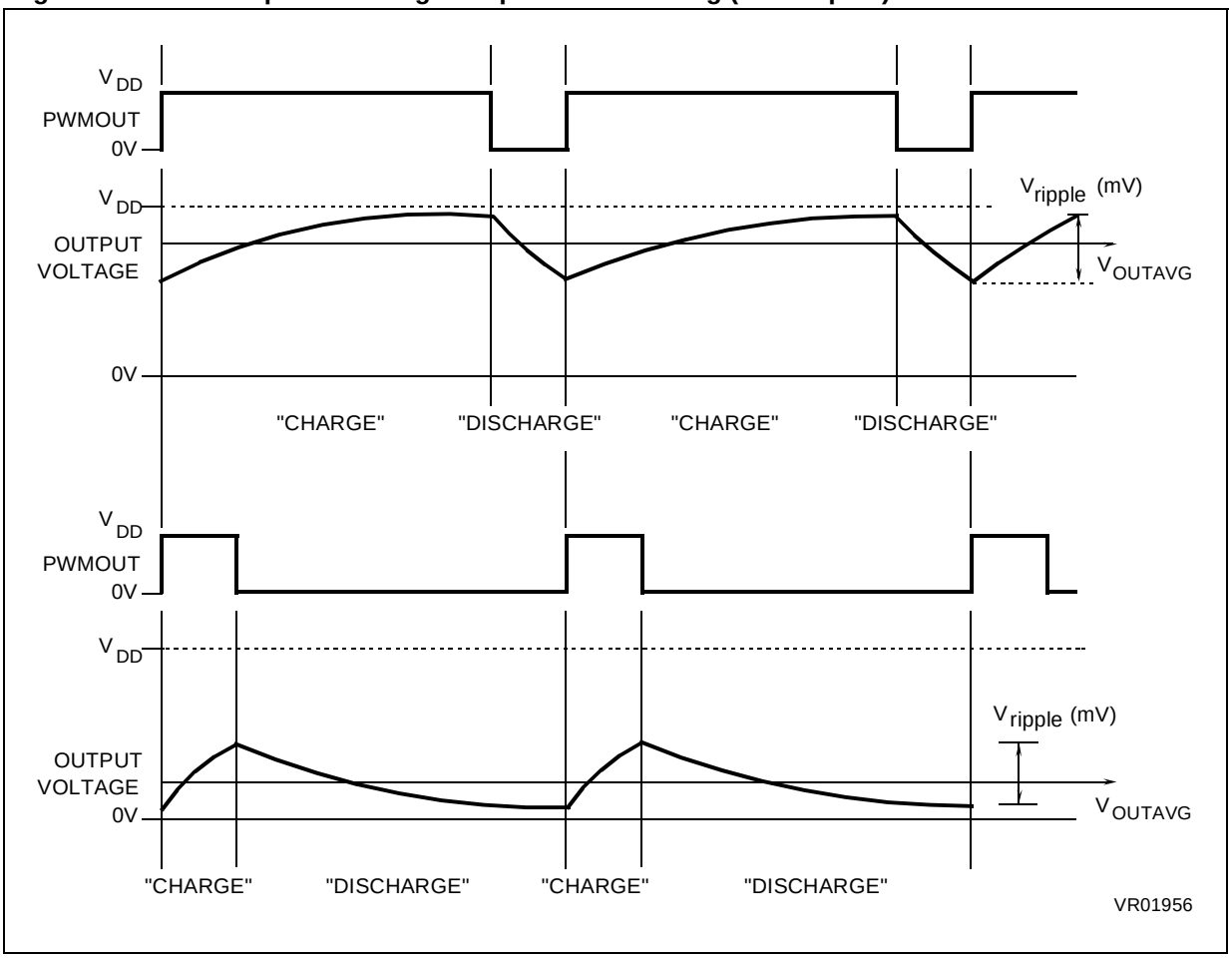


Figure 92. PWM Generation



VOLTAGE SYNTHESIS (Cont'd)

Figure 93. PWM Simplified Voltage Output After Filtering (2 examples)



VOLTAGE SYNTHESIS (Cont'd)

BRM Generation

The BRM bits allow the addition of a pulse to widen a standard PWM pulse for specific PWM cycles. This has the effect of "fine-tuning" the PWM Duty cycle (without modifying the base duty cycle), thus, with the external filtering, providing additional fine voltage steps.

The incremental pulses (with duration of T_{INTCLK}/4) are added to the beginning of the original PWM pulse and thus cause the PWM high time to be extended by this time with a corresponding reduction in the low time. The PWM intervals which are added to are specified in the lower 7 bits of the tuning word and are encoded as shown in the following table.

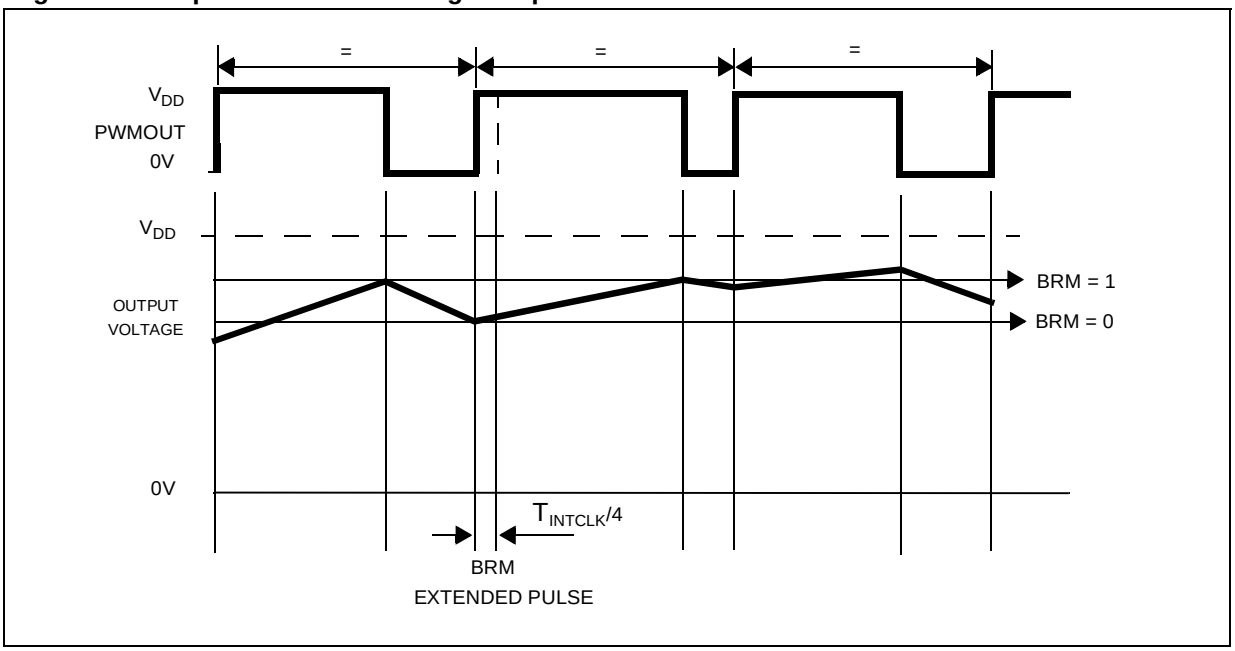
Table 44. 7-Bit BRM Pulse Addition Positions

Fine Tuning	No. of Pulses added at the following Cycles
0000001	64
0000010	32, 96
0000100	16, 48, 80, 112
0001000	8, 24,... 104, 120
0010000	4, 12,... 116, 124
0100000	2, 6,... 122, 126
1000000	1, 3,... 125, 127

The BRM values shown may be combined together to provide a summation of the incremental pulse intervals specified.

The pulse increment corresponds to the PWM resolution.

Figure 94. Simplified Filtered Voltage Output Schematic with BRM added



VOLTAGE SYNTHESIS (Cont'd)

8.12.3 Register Description

VS DATA AND CONTROL REGISTER 1 (VSDR1)
R254 - Read/Write
Register Page: 59
Reset Value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
VSE	VSWP	VD13	VD12	VD11	VD10	VD9	VD8

Bit 7 = **VSE**: *VS enable bit.*
0: VS Tuning Converter disabled (i.e. the clock is not forwarded to the VS counter and the 2 outputs are set to 1 (idle state)
1: VS Tuning Converter enabled.

Bit 6 = **VSWP**: *VS Output Select*
This bit controls which VS output is enabled to output the VS signal.
0: VSO1 output selected
1: VSO2 output selected

Bit 5:0 = **VD[13:8]** *Tuning word bits.*
These bits are the 6 most significant bits of the Tuning word forming the PWM selection. The VD13 bit is the MSB.

VS DATA AND CONTROL REGISTER 2 (VSDR2)
R255 - Read/Write
Register Page: 59
Reset Value: 0000 0000 (00h)

7							0
VD7	VD6	VD5	VD4	VD3	VD2	VD1	VD0

Bit 7:0 = **VD[7:0]** *Tuning word bits.*
These bits are the 8 least significant data bits of the VS Tuning word. All bits are accessible. Bits VD6 - VD0 form the BRM pulse selection. VD7 is the LSB of the 7 bits forming the PWM selection.

8.13 PWM GENERATOR

8.13.1 Introduction

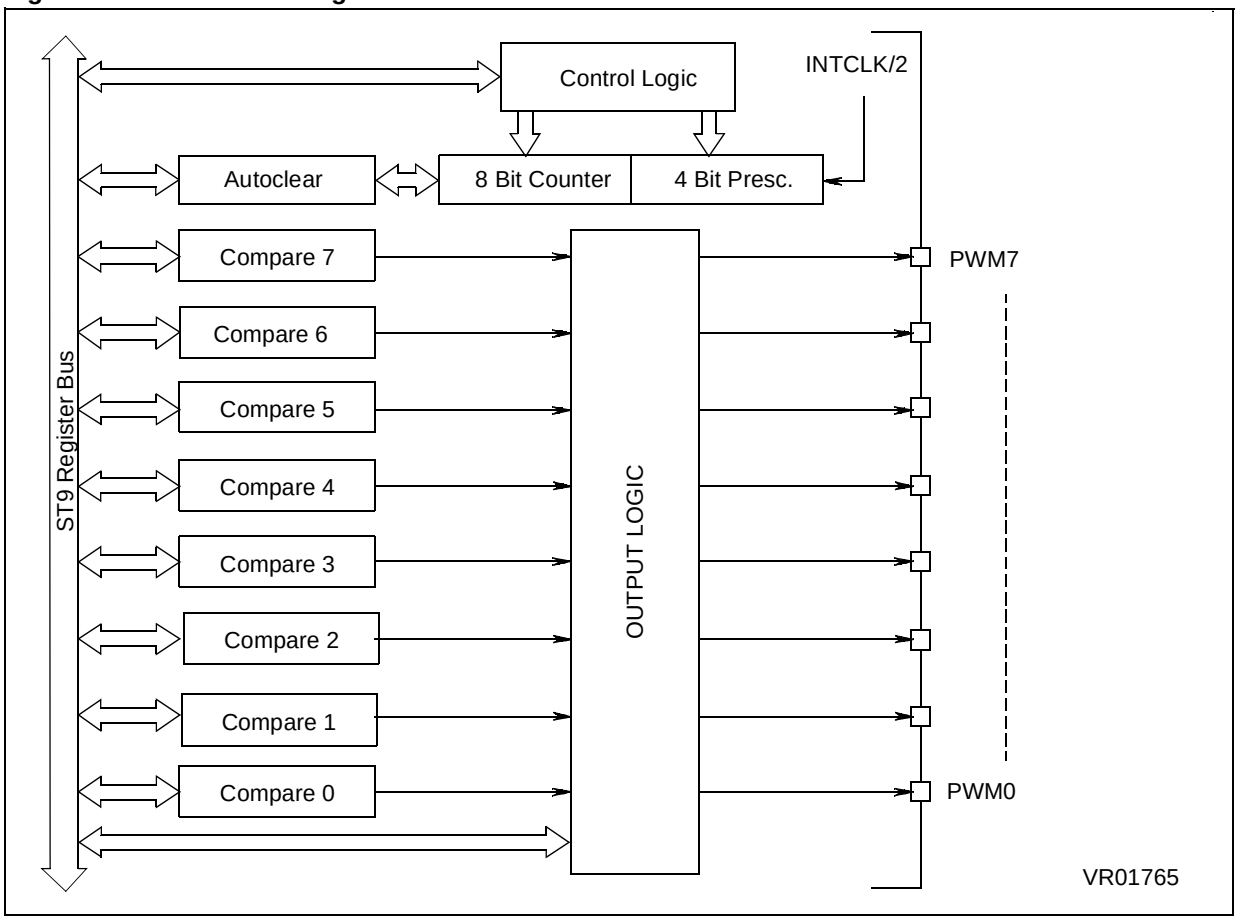
The PWM (Pulse Width Modulated) signal generator allows the digital generation of up to 8 analog outputs when used with an external filtering network.

The unit is based around an 8-bit counter which is driven by a programmable 4-bit prescaler, with an input clock signal equal to the internal clock INTCLK divided by 2. For example, with a 12 MHz Internal clock, using the full 8-bit resolution, a frequency range from 1465 Hz up to 23437 Hz can be achieved.

Higher frequencies, with lower resolution, can be achieved by using the autoclear register. As an example, with a 12 MHz Internal clock, a maximum PWM repetition rate of 93750 Hz can be reached with 6-bit resolution.

Note: The number of output pins is device dependant. Refer to the device pinout description.

Figure 95. PWM Block Diagram.



PWM GENERATOR (Cont'd)

Up to 8 PWM outputs can be selected as Alternate Functions of an I/O port. Each output bit is independently controlled by a separate Compare Register. When the value programmed into the Compare Register and the counter value are equal, the corresponding output bit is set. The output bit is reset by a counter clear (by overflow or autoclear), generating the variable PWM signal.

Each output bit can also be complemented or disabled under software control.

8.13.2 Register Mapping

The ST9 can have one or two PWM Generators. Each has 13 registers mapped in page 59 (PWM0) or page 58 (PWM1). In the register description on the following pages, the register page refers to PWM0 only.

Register Address	Register	Function
R240	CM0	Ch. 0 Compare Register
R241	CM1	Ch. 1 Compare Register
R242	CM2	Ch. 2 Compare Register
R243	CM3	Ch. 3 Compare Register
R244	CM4	Ch. 4 Compare Register
R245	CM5	Ch. 5 Compare Register
R246	CM6	Ch. 6 Compare Register
R247	CM7	Ch. 7 Compare Register
R248	ACR	Autoclear Register
R249	CRR	Counter Read Register
R250	PCTLR	Prescaler/ Reload Reg.
R251	OCPLR	Output Complement Reg.
R252	OER	Output Enable Register
R253- R255	—	Reserved

Figure 96. PWM Action When Compare Register = 0 (no complement)

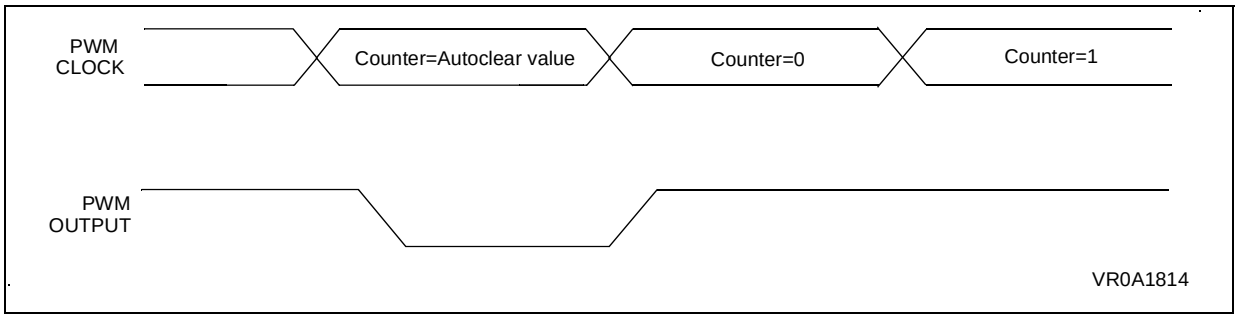
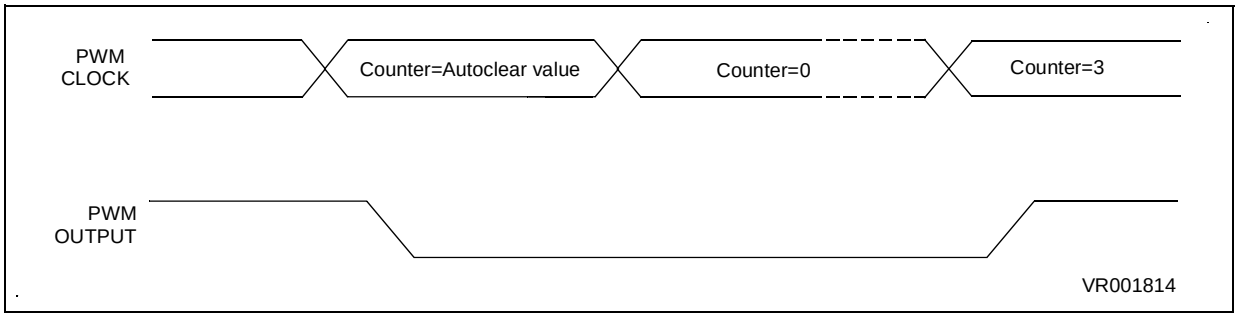


Figure 97. PWM Action When Compare Register = 3 (no complement)



PWM GENERATOR (Cont'd)

8.13.2.1 Register Description

COMPARE REGISTER 0 (CM0)
R240 - Read/Write
Register Page: 59
Reset Value: 0000 0000 (00h)

7							0
CM0.7	CM0.6	CM0.5	CM0.4	CM0.3	CM0.2	CM0.1	CM0.0

This is the compare register controlling PWM output 0. When the programmed content is equal to the counter content, a SET operation is performed on PWM output 0 (if the output has not been complemented or disabled).

Bit 7:0 = **CM0.[7:0]**: PWM Compare value Channel 0.

COMPARE REGISTER 1 (CM1)
R241 - Read/Write
Register Page: 59
Reset Value: 0000 0000 (00h)

7							0
CM1.7	CM1.6	CM1.5	CM1.4	CM1.3	CM1.2	CM1.1	CM1.0

This is the compare register controlling PWM output 1.

COMPARE REGISTER 2 (CM2)
R242 - Read/Write
Register Page: 59
Reset Value: 0000 0000 (00h)

7							0
CM2.7	CM2.6	CM2.5	CM2.4	CM2.3	CM2.2	CM2.1	CM2.0

This is the compare register controlling PWM output 2.

COMPARE REGISTER 3 (CM3)
R243 - Read/Write
Register Page: 59
Reset Value: 0000 0000 (00h)

7							0
CM3.7	CM3.6	CM3.5	CM3.4	CM3.3	CM3.2	CM3.1	CM3.0

This is the compare register controlling PWM output 3.

COMPARE REGISTER 4 (CM4)
R244 - Read/Write
Register Page: 59
Reset Value: 0000 0000 (00h)

7							0
CM4.7	CM4.6	CM4.5	CM4.4	CM4.3	CM4.2	CM4.1	CM4.0

This is the compare register controlling PWM output 4.

COMPARE REGISTER 5 (CM5)
R245 - Read/Write
Register Page: 59
Reset Value: 0000 0000 (00h)

7							0
CM5.7	CM5.6	CM5.5	CM5.4	CM5.3	CM5.2	CM5.1	CM5.0

This is the compare register controlling PWM output 5.

COMPARE REGISTER 6 (CM6)
R246 - Read/Write
Register Page: 59
Reset Value: 0000 0000 (00h)

7							0
CM6.7	CM6.6	CM6.5	CM6.4	CM6.3	CM6.2	CM6.1	CM6.0

This is the compare register controlling PWM output 6.

COMPARE REGISTER 7 (CM7)
R247 - Read/Write
Register Page: 59
Reset Value: 0000 0000 (00h)

7							0
CM7.7	CM7.6	CM7.5	CM7.4	CM7.3	CM7.2	CM7.1	CM7.0

This is the compare register controlling PWM output 7.

PWM GENERATOR (Cont'd)

AUTOCLEAR REGISTER (ACR)
R248 - Read/Write
Register Page: 59
Reset Value: 1111 1111 (FFh)

7							0
AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0

This register behaves exactly as a 9th compare Register, but its effect is to clear the CRR counter register, so causing the desired PWM repetition rate.

The reset condition generates the free running mode. So, FFh means count by 256.

Bit 7:0 = **AC[7:0]**: *Autoclear Count Value.*

When 00 is written to the Compare Register, if the ACR register = FFh, the PWM output bit is always set except for the last clock count (255 set and 1 reset; the converse when the output is complemented). If the ACR content is less than FFh, the PWM output bit is set for a number of clock counts equal to that content (see Figure 2).

Writing the Compare register constant equal to the ACR register value causes the output bit to be always reset (or set if complemented).

Example: If 03h is written to the Compare Register, the output bit is reset when the CRR counter reaches the ACR register value and set when it reaches the Compare register value (after 4 clock counts, see Figure 97). The action will be reversed if the output is complemented. The PWM mark/space ratio will remain constant until changed by software writing a new value in the ACR register.

COUNTER REGISTER (CRR)
R249 - Read Only
Register Page: 59
Reset Value: 0000 0000 (00h)

7							0
CR7	CR6	CR5	CR4	CR3	CR2	CR1	CR0

This read-only register returns the current counter value when read.

The 8 bit Counter is initialized to 00h at reset, and is a free running UP counter.

Bit 7:0 = **CR[7:0]**: *Current Counter Value.*

PRESCALER AND CONTROL REGISTER (PCTL)
R250 - Read/Write
Register Page: 59
Reset Value: 0000 1100 (0Ch)

7							0
PR3	PR2	PR1	PR0	1	1	CLR	CE

Bit 7:4 = **PR[3:0]** *PWM Prescaler value.*
These bits hold the Prescaler preset value. This is reloaded into the 4-bit prescaler whenever the prescaler (DOWN Counter) reaches the value 0, so determining the 8-bit Counter count frequency. The value 0 corresponds to the maximum counter frequency which is INTCLK/2. The value Fh corresponds to the maximum frequency divided by 16 (INTCLK/32).

The reset condition initializes the Prescaler to the Maximum Counter frequency.

PR[3:0]	Divider Factor	Frequency
0	1	INTCLK/2 (Max.)
1	2	INTCLK/4
2	3	INTCLK/6
..	..	..
Fh	16	INTCLK/32 (Min.)

Bit 3:2 = Reserved. Forced by hardware to "1"

Bit 1 = **CLR**: *Counter Clear.*
This bit when set, allows both to clear the counter, and to reload the prescaler. The effect is also to clear the PWM output. It returns "0" if read.

Bit 0 = **CE**: *Counter Enable.*
This bit enables the counter and the prescaler when set to "1". It stops both when reset without affecting their current value, allowing the count to be suspended and then restarted by software "on fly".

PWM GENERATOR (Cont'd)

OUTPUT COMPLEMENT REGISTER (OCPL)
R251- Read/Write
Register Page 59
Reset Value: 0000 0000 (00h)

7							0
OCPL.7	OCPL.6	OCPL.5	OCPL.4	OCPL.3	OCPL.2	OCPL.1	OCPL.0

This register allows the PWM output level to be complemented on an individual bit basis.

In default mode (reset configuration), each comparison true between a Compare register and the counter has the effect of setting the corresponding output.

At counter clear (either by autoclear comparison true, software clear or overflow when in free running mode), all the outputs are cleared.

By setting each individual bit (OCPL.x) in this register, the logic value of the corresponding output will be inverted (i.e. reset on comparison true and set on counter clear).

Example: When set to "1", the OCPL.1 bit complements the PWM output 1.

Bit 7 = **OCPL.7**: *Complement PWM Output 7.*

Bit 6 = **OCPL.6**: *Complement PWM Output 6.*

Bit 5 = **OCPL.5**: *Complement PWM Output 5.*

Bit 4 = **OCPL.4**: *Complement PWM Output 4.*

Bit 3 = **OCPL.3**: *Complement PWM Output 3.*

Bit 2 = **OCPL.2**: *Complement PWM Output 2.*

Bit 1 = **OCPL.1**: *Complement PWM Output 1.*

Bit 0 = **OCPL.0**: *Complement PWM Output 0.*

OUTPUT ENABLE REGISTER (OER)
R252 - Read/Write
Register Page: 59
Reset Value: 0000 0000 (00h)

7							0
OE.7	OE.6	OE.5	OE.4	OE.3	OE.2	OE.1	OE.0

These bits are set and cleared by software.
0: Force the corresponding PWM output to logic level 1. This allows the port pins to be used for normal I/O functions or other alternate functions (if available).
1: Enable the corresponding PWM output.

Example: Writing 03h into the OE Register will enable only PWM outputs 0 and 1, while outputs 2, 3, 4, 5, 6 and 7 will be forced to logic level "1".

Bit 7 = **OE.7**: *Output Enable PWM Output 7.*

Bit 6 = **OE.6**: *Output Enable PWM Output 6.*

Bit 5 = **OE.5**: *Output Enable PWM Output 5.*

Bit 4 = **OE.4**: *Output Enable PWM Output 4.*

Bit 3 = **OE.3**: *Output Enable PWM Output 3.*

Bit 2 = **OE.2**: *Output Enable PWM Output 2.*

Bit 1 = **OE.1**: *Output Enable PWM Output 1.*

Bit 0 = **OE.0**: *Output Enable PWM Output 0.*

8.14 A/D CONVERTER (A/D)

8.14.1 Introduction

The 8 bit Analog to Digital Converter uses a fully differential analog configuration for the best noise immunity and precision performance. The analog voltage references of the converter are connected to the internal AV_{DD} & AV_{SS} analog supply pins of the chip if they are available, otherwise to the ordinary V_{DD} and V_{SS} supply pins of the chip. The guaranteed accuracy depends on the device (see Electrical Characteristics). A fast Sample/Hold allows quick signal sampling for minimum warping effect and conversion error.

8.14.2 Main Features

- 8-bit resolution A/D Converter
- Single Conversion Time (including Sampling Time):
  - 138 internal system clock periods in slow mode (~5.6 µs @25Mhz internal system clock);
  - 78 INTCLK periods in fast mode (~6.5 µs @ 12MHZ internal system clock)
- Sample/Hold: Tsample=
  - 84 INTCLK periods in slow mode (~3.4 µs @25Mhz internal system clock)
  - 48 INTCLK periods in fast mode (~4 µs @12Mhz internal system clock)
- Up to 8 Analog Inputs (the number of inputs is device dependent, see device pinout)
- Single/Continuous Conversion Mode
- External source Trigger (Alternate synchronization)
- Power Down mode (Zero Power Consumption)
- 1 Control Logic Register
- 1 Data Register

8.14.3 General Description

Depending on the device, up to 8 analog inputs can be selected by software.

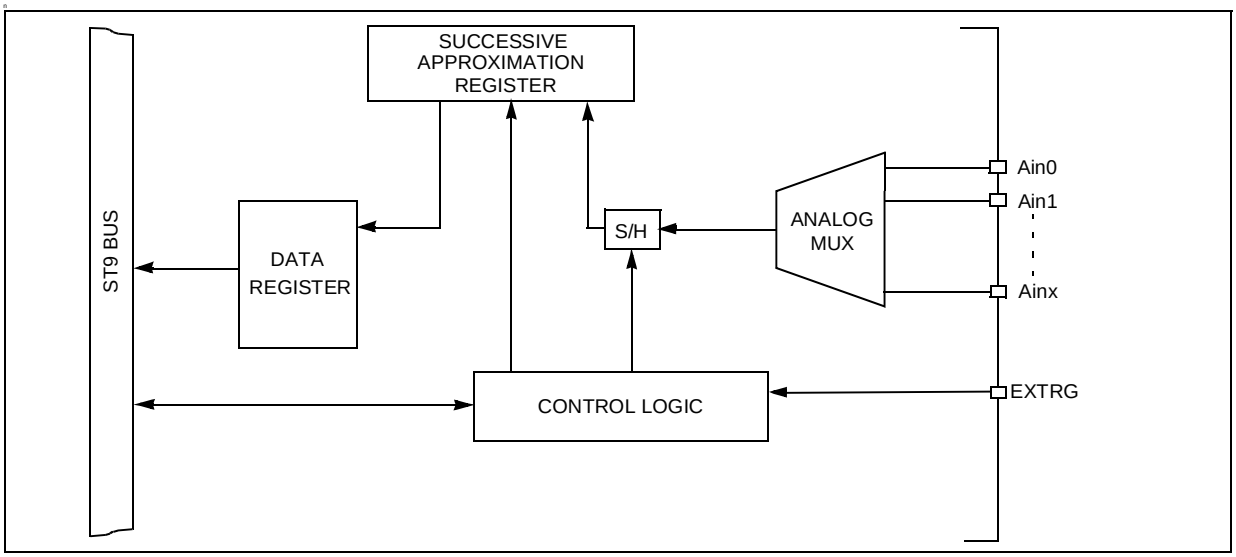
Different conversion modes are provided: single, continuous, or triggered. The continuous mode performs a continuous conversion flow of the selected channel, while in the single mode the selected channel is converted once and then the logic waits for a new hardware or software restart.

A data register (ADDTR) is available, mapped in page 62, allowing data storage (in single or continuous mode).

The start conversion event can be managed either

- by software, writing the START/STOP bit of the Control Logic Register
- or by hardware using an external signal on the EXTRG triggered input (negative edge sensitive) connected as an Alternate Function to an I/O port bit

Figure 98. A/D Converter Block Diagram



A/D CONVERTER (Cont'd**)**

The conversion technique used is successive approximation, with AC coupled analog fully differential comparators blocks plus a Sample and Hold logic and a reference generator.

The internal reference (DAC) is based on the use of a binary-ratioed capacitor array. This technique allows the specified monotonicity (using the same ratioed capacitors as sampling capacitor). A Power Down programmable bit sets the A/D converter analog section to a zero consumption idle status.

8.14.3.1 Operating Modes

The two main operating modes, single and continuous, can be selected by writing 0 (reset value) or 1 into the CONT bit of the Control Logic Register.

Single Mode

In single mode (CONT=0 in ADCLR) the STR bit is forced to '0' after the end of channel i-th conversion; then the A/D waits for a new start event. This mode is useful when a set of signals must be sampled at a fixed frequency imposed by a Timer unit or an external generator (through the alternate synchronization feature). A simple software routine monitoring the STR bit can be used to save the current value before a new conversion ends (so to create a signal samples table within the internal memory or the Register File). Furthermore, if the R242.0 bit (register AD-INT, bit 0) is set, at the end of conversion a negative edge on the connected external interrupt channel (see Interrupts Chapter) is generated to allow the reading of the converted data by means of an interrupt routine.

Continuous Mode

In continuous mode (CONT=1 in ADCLR) a continuous conversion flow is entered by a start event on the selected channel until the STR bit is reset by software.

At the end of each conversion, the Data Register (ADCDR) content is updated with the last conversion result, while the former value is lost. When the conversion flow is stopped, an interrupt request is generated with the same modality previously described.

8.14.3.2 Alternate Synchronization

This feature is available in both single/continuous modes. The negative edge of external EXTRG signal can be used to synchronize the conversion start with a trigger pulse. This event can be enabled or masked by programming the TRG bit in the ADCLR Register.

The effect of alternate synchronization is to set the STR bit, which is cleared by hardware at the end of each conversion in single mode. In continuous mode any trigger pulse following the first one will be ignored. The synchronization source must provide a pulse (1.5 internal system clock, 125ns @ 12 MHz internal clock) of minimum width, and a period greater (in single mode) than the conversion time (~6.5us @ 12 MHz internal clock). If a trigger occurs when the STR bit is still '1' (conversions still in progress), it is ignored (see Electrical Characteristics).

WARNING: If the EXTRG signal is already active when TRG bit is set, the conversion starts immediately.

8.14.3.3 Power-Up Operations

Before enabling any A/D operation mode, set the POW bit of the ADCLR Register at least 60 µs before the first conversion starts to enable the biasing circuits inside the analog section of the converter. Clearing the POW bit is useful when the A/D is not used so reducing the total chip power consumption. This state is also the reset configuration and it is forced by hardware when the core is in HALT state (after a HALT instruction execution).

8.14.3.4 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 map is the following:

Register Address	ADn	Page 62 (3Eh)
F0h	A/D 0	ADDTR0
F1h	A/D 0	ADCLR0
F2h	A/D 0	ADINT0
F3-F7h	A/D 0	Reserved
F8h	A/D 1	ADDTR1
F9h	A/D 1	ADCLR1
FAh	A/D 1	ADINT1
FB-FFh	A/D 1	Reserved

If two A/D converters are present, the registers are renamed, adding the suffix 0 to the A/D 0 registers and 1 to the A/D 1 registers.

A/D CONVERTER (Cont'd**)**

8.14.4 Register Description

A/D CONTROL LOGIC REGISTER (ADCLR)
R241 - Read/Write
Register Page: 62
Reset value: 0000 0000 (00h)

7							0
C2	C1	C0	FS	TRG	POW	CONT	STR

This 8-bit register manages the A/D logic operations. Any write operation to it will cause the current conversion to be aborted and the logic to be re-initialized to the starting configuration.

Bit 7:5 = **C[2:0]**: *Channel Address.*
These bits are set and cleared by software. They select channel `i` conversion as follows:

C2	C1	C0	Channel Enabled
0	0	0	Channel 0
0	0	1	Channel 1
0	1	0	Channel 2
0	1	1	Channel 3
1	0	0	Channel 4
1	0	1	Channel 5
1	1	0	Channel 6
1	1	1	Channel 7

Bit 4 = **FS**: *Fast/Slow*.
This bit is set and cleared by software.
0: Fast mode. Single conversion time: 78 x INTCLK (5.75µs at INTCLK = 12 MHz)
1: Slow mode. Single conversion time: 138 x INTCLK (11.5µs at INTCLK = 12 MHz)

Note: Fast conversion mode is only allowed for internal speeds which do not exceed 12 MHz.

Bit 3 = **TRG**: *External Trigger Enable*.
This bit is set and cleared by software.
0: External Trigger disabled.
1: A negative (falling) edge on the EXTRG pin writes a "1" into the STR bit, enabling start of conversion.

Note: Triggering by on chip event is available on devices with the multifunction timer (MFT) peripheral.

Bit 2 = **POW**: *Power Enable*.
This bit is set and cleared by software.
0: Disables all power consuming logic.
1: Enables the A/D logic and analog circuitry.

Bit 1 = **CONT**: *Continuous/Single Mode Select*.
This bit it set and cleared by software.
0: Single mode: after the current conversion ends, the STR bit is reset by hardware and the converter logic is put in a wait status. To start another conversion, the STR bit has to be set by software or hardware.
1: Select Continuous Mode, a continuous flow of A/D conversions on the selected channel, starting when the STR bit is set.

Bit 0 = **STR**: *Start/Stop*.
This bit is set and cleared by software. It is also set by hardware when the A/D is synchronized with an external trigger.
0: Stop conversion on channel i. An interrupt is generated if the STR was previously set and the AD-INT bit is set.
1: Start conversion on channel i

WARNING: When accessing this register, it is recommended to keep the related A/D interrupt channel masked or disabled to avoid spurious interrupt requests.

A/D CHANNEL `i` DATA REGISTER (ADDTR)
R240 - Read/Write
Register Page: 62
Reset value: undefined

7							0
R.7	R.6	R.5	R.4	R.3	R.2	R.1	R.0

The result of the conversion of the selected channel is stored in the 8-bit ADDTR, which is reloaded with a new value every time a conversion ends.

A/D CONVERTER (Cont'd**)**

A/D INTERRUPT REGISTER (ADINT)
Register Page: 62
R242 - Read/write
Reset value: 0000 0001 (01h)

7							0
-	-	-	-	-	-	-	AD-INT

Bit 7:1 = Reserved.

Bit 0 = **AD-INT**: *AD Converter Interrupt Enable*.

This bit is set and cleared by software. It allows the interrupt source to be switched between the A/D Converter and an external interrupt pin (See Interrupts chapter).
0: A/D Interrupt disabled. External pin selected as interrupt source.
1: A/D Interrupt enabled

9 ELECTRICAL CHARACTERISTICS

The ST92196A device contains circuitry to protect the inputs against damage due to high static voltage or electric field. Nevertheless it is advised to take normal precautions and to avoid applying to this high impedance voltage circuit any voltage higher than the maximum rated voltages. It is recommended for proper operation that V_{IN} and V_{OUT} be constrained to the range.

$V_{SS} \leq (V_{IN}$ or $V_{OUT}) \leq V_{DD}$

To enhance reliability of operation, it is recommended to connect unused inputs to an appropriate logic voltage level such as V_{SS} or V_{DD}.
All the voltages in the following table, are referenced to V_{SS}.

ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
V_{DD}1,2	Supply Voltage	V_{SS} – 0.3 to V_{SS} + 7.0	V
V_{DDA}	Analog Supply Voltage	V_{SS} – 0.3 to V_{DD} +0.3	V
V_I	Input Voltage	V_{SS} – 0.3 to V_{DD} +0.3	V
V_I	Input Voltage	V_{SS} – 0.7 to V_{DD} +0.7 *	V
V_O	Output Voltage	V_{SS} – 0.3 to V_{DD} +0.3	V
V_O	Output Voltage	V_{SS} – 0.7 to V_{DD} +0.7 *	V
T_{STG}	Storage Temperature	– 55 to + 150	°C
I_{INJ}	Pin Injection Current Digital and Analog Input	-5 to +5	mA
	Maximum Accumulated Pin injection Current in the device	-50 to +50	mA

*Current is limited to |<200µA| into the pin

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.

RECOMMENDED OPERATING CONDITIONS

Symbol	Parameter	Value		Unit
		Min.	Max.	
T_A	Operating Temperature	-10	75	°C
V_{DD}	Operating Supply Voltage	4.5	5.5	V
V_{DDa}	Analog Supply Voltage	4.5	5.5	V
f_{OSCE}	External Oscillator Frequency		4.0	MHz
f_{INTCLK}	Internal Clock Frequency	0[1]	24	MHz

Note 1. 1MHz when A/D is used
Note 2. For good slicing results, it is advised to set VDDA >= 5.3V. Remember also that VDDA < VDD + 0.3V.

DC ELECTRICAL CHARACTERISTICS

(V_{DD} = 5V ± 10% T_A = -10°C + 75°C, unless otherwise specified)

Symbol	Parameter	Test Conditions	Value		Unit
			Min.	Max.	
V_{IH}	Input High Level	TTL	2.0		V
		CMOS	0.7 V_{DD}		V
V_{IL}	Input Low Level	TTL		0.8	V
		CMOS		0.3 V_{DD}	V
V_{IHRS}	$\overline{\text{RESET}}$ Input High Level		0.8 V_{DD}		V
V_{ILRS}	$\overline{\text{RESET}}$ Input Low Level			0.3 V_{DD}	V
V_{HYRS}	$\overline{\text{RESET}}$ Input Hysteresis		0.5		V
V_{IH}	P2.0 Input High Level		0.8 V_{DD}		V
V_{IL}	P2.0 Input Low Level			0.3 V_{DD}	V
V_{IHY}	P2.0 Input Hysteresis		0.5		V
V_{IHIIC}	SDA4,1/SCL4,1 Input High Level		0.8 V_{DD}		V
V_{ILIIC}	SDA4,1/SCL4,1 Input Low Level			0.3 V_{DD}	V
V_{IHYIIC}	SDA4,1/SCL4,1 Hysteresis		0.5		V
V_{IHVH}	HSYNC/VSYNC Input High Level		0.8 V_{DD}		V
V_{ILVH}	HSYNC/VSYNC Input Low Level			0.3 V_{DD}	V
V_{HYHV}	HSYNC/VSYNC Input Hysteresis		1.2		V
V_{IHDT}	SYNDET1,0 Input High Level		0.8 V_{DD}		V
V_{ILDT}	SYNDET1,0 Input Low Level			0.3 V_{DD}	V
V_{HYDT}	SYNDET1,0 Input Hysteresis		0.5		V
V_{OH}	Output High Level	Push Pull, Iload = – 0.8mA	V_{DD} – 0.8		V
V_{OL}	Output Low Level	Push Pull or Open Drain, Iload = 1.6mA		0.4	V
V_{pp}	EPROM programming voltage			12.8	V
I_{LKIO}	I/O Pin Input Leakage Current	Hi-Z Input, $0V < V_{IN} < V_{DD}$		± 1.0	µA
I_{LKRS}	$\overline{\text{RESET}}$ Pin Input Leakage Current	$0V < V_{IN} < V_{DD}$		± 1.0	µA
$I_{LKA/D}$	A/D Pin Input Leakage Current	Alternate function open drain		± 1.0	µA
I_{LKOS}	OSCIN Pin Input Leakage Current	$0V < V_{IN} < V_{DD}$		± 1.0	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. No peripheral working.

ELECTRICAL CHARACTERISTICS

AC ELECTRICAL CHARACTERISTICS

PIN CAPACITANCE

(V_{DD} = 5V ± 10%; T_A = -10°C + 75°C, unless otherwise specified)

Symbol	Parameter	Conditions	Value		Unit
			Typ.	Max	
C_{IO}	Pin Capacitance Digital Input/Output		8	10	pF

CURRENT CONSUMPTION

(V_{DD} = 5V ± 10%; T_A = -10°C + 75°C, unless otherwise specified)

Symbol	Parameter	Conditions	Value		Unit
			Typ.	Max	
I_{CC1}	Run Mode Current	(Notes 1, 2) INTCLK=16MHz	80	100	mA
I_{CC2}	Run Mode Current	(Notes 1, 2) INTCLK=24MHz	100	120	mA
I_{CC3}	Run Mode Current	(Notes 1, 2) INTCLK=4MHz	25	30	mA
I_{CC4}	Run Mode Current	(Notes 1, 5) INTCLK=16MHz	65	78	mA
I_{CCA1}	Analog Current (V_{DDA} pin)	Freq. Multipliers , A/D, OSD, All DACs & Slicers On.	45	55	mA
I_{CCA2}	Analog Current (V_{DDA} pin)	Freq. Multipliers , A/D, OSD, DACs & Slicers Off.	1	10	µA
I_{CCA3}	Analog Current (V_{DDA} pin)	Freq. Multipliers , A/D, OSD, two DACs & one Slicer On.	25	30	mA
I_{ILPR}	Reset Mode Current	(Note 3)	10	100	µA
I_{HALT}	HALT Mode Current	(Note 4)	10	100	µA

Notes :

1. All ports are configured in push-pull output mode (output is high). VSYNC and HSYNC are tied to V_{SS}, CCVIDEO is floating. The internal clock prescaler is in divide-by-1 mode. The external CLOCK pin (OSCIN) is driven by a square wave external clock at 4 MHz.

2. The CPU is fed by a frequency issued by the on-chip Frequency Multiplier. The Skew Corrector Frequency Multiplier provides a 28 MHz clock. All peripherals are working.

3. All ports are configured in push-pull output mode (output is high). VSYNC and HSYNC are tied to V_{SS}, CCVIDEO is floating. External CLOCK pin (OSCIN) and Reset pins are held low. All peripherals are disabled.

4. All ports are configured in push-pull output mode (output is high). VSYNC and HSYNC are tied to V_{SS}, CCVIDEO is floating. All peripherals are disabled.

5. The CPU is fed by a frequency issued by the on-chip Frequency Multiplier. The Skew Corrector Frequency Multiplier provides a 14Mhz clock. OSD, A/D, PWM, Sync Error Detector, Std Timer and WDG Timer peripherals are running.

AC ELECTRICAL CHARACTERISTICS (Cont'd)

CLOCK TIMING

(V_{DD} = 5V ± 10% T_A = -10°C + 75°C, unless otherwise specified)

Symbol	Parameter	Conditions	Value		Unit
			Min	Max	
TpC	OSCIN Clock Period	intern. div. by 2	41.7		ns
		intern. div. by 1	83.3		ns
TrC	OSCIN rise time			12	ns
TfC	OSCIN fall time			12	ns
TwCL	OSCIN low width	intern. div. by 2	17		ns
		intern. div. by 1	38		ns
TwCH	OSCIN high width	intern. div. by 2	17		ns
		intern. div. by 1	38		ns

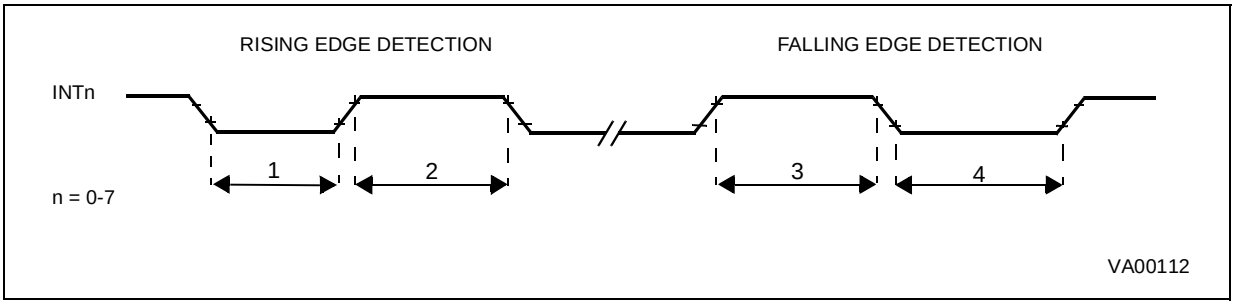
EXTERNAL INTERRUPT TIMING

(Rising or falling edge mode; V_{DD} = 5V ± 10%; T_A = -10°C + 75°C, unless otherwise specified)

N°	Symbol	Parameter	Conditions				Unit
			OSCIN Divided by 2 Min.	OSCIN Not Divided by 2 Min.	Min	Max	
1	TwLR	Low Level Minimum Pulse width in Rising Edge Mode	2TpC + 12	TpC + 12	95		ns
2	TwHR	High Level Minimum Pulse width in Rising Edge Mode	2TpC + 12	TpC + 12	95		ns
3	TwLF	Low Level Minimum Pulse width in Falling Edge Mode	2TpC + 12	TpC + 12	95		ns
4	TwHF	High Level Minimum Pulse width in Falling Edge Mode	2TpC + 12	TpC + 12	95		ns

Note: The value in the left hand two columns shows the formula used to calculate the minimum or maximum timing from the oscillator clock period, prescale value and number of wait cycles inserted. The value in the rignt hand two columns shows the minimum and maximum for an external clock at 24 MHz divided by 2, prescale value of zero and zero wait status.

EXTERNAL INTERRUPT TIMING



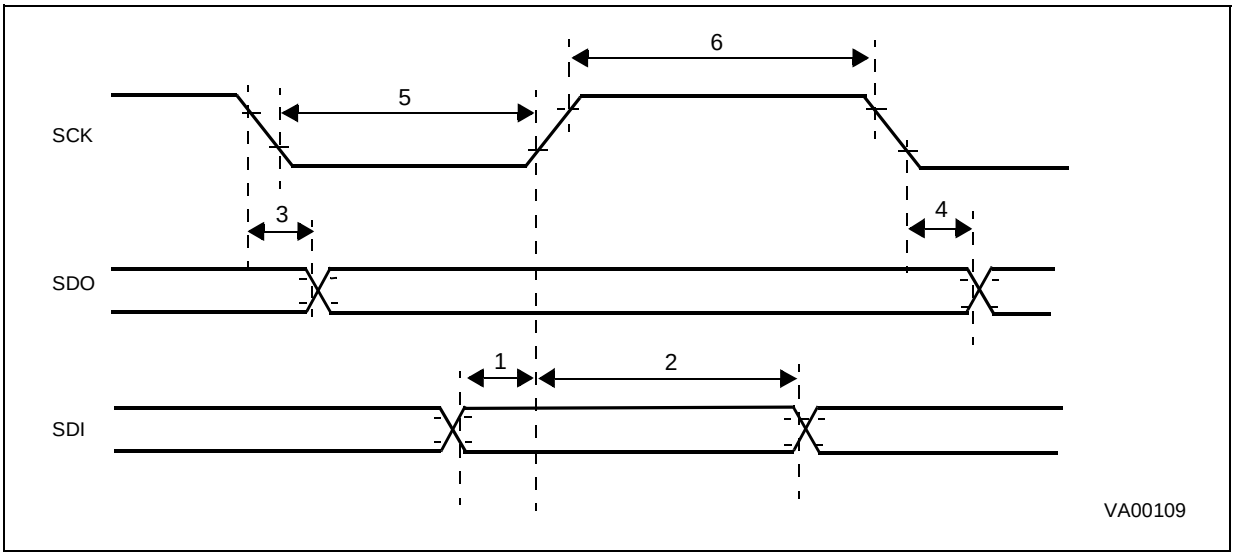
AC ELECTRICAL CHARACTERISTICS (Cont'd)

SPI TIMING

(V_{DD} = 5V ± 10% ; T_A = -10°C + 75°C, unless otherwise specified)

N°	Symbol	Parameter	Conditions	Value		Unit
				Min	Max	
1	TsDI	Input Data Set-up Time		100		
2	ThDI	Input Data Hold Time		1/2 TpC + 100		
3	TdOV	SCK to Output Data Valid			100	
4	ThDO	Output Data Hold Time		-20		
5	TwSKL	SCK Low Pulse Width		300		
6	TwSKH	SCK High Pulse Width		300		

SPI TIMING



SKEW CORRECTOR TIMING TABLE

(V_{DD} = 5V ± 10%; T_A = -10°C + 75°C, unless otherwise specified)

Symbol	Parameter	Conditions	Value Max	Unit
Tjskw	Jitter on RGB output	28 MHz Skew corrector clock frequency	<12 *	ns

The OSD jitter is measured from leading edge to leading edge of a single character row on consecutive TV lines. The value is an envelope of 100 fields
*Max. value at all CPU operating frequencies

AC ELECTRICAL CHARACTERISTICS (Cont'd)

OSD DAC CHARACTERISTICS

(V_{DD} = 5V ± 10%; T_A = -10°C + 75°C, unless otherwise specified)

Symbol	Parameter	Conditions	Value			Unit
			Min	Typ	Max	
	Output impedance FB,R,G,B				100	Ohm
	Output voltage FB,R,G,B	Cload = 20 pF RL=100K				
	code = 111		0.976	1.170	1.364	V
	code = 110		0.863	1.034	1.205	
	code = 101		0.751	0.899	1.046	
	code = 100		0.638	0.763	0.887	
	code = 011		0.525	0.627	0.729	
	code = 010		0.412	0.491	0.570	
	code = 001		0.300	0.356	0.411	
	code = 000		0.157	0.220	0.252	
	FB = 1 FB = 0			5.0 0.2		V
	Relative voltage accuracy	(*)			+/-5	%
	R/G/B to FB 50% point matching	FB DAC mode (**)			5	ns
	R/G/B to FB 50% point matching	FB digital mode (***)			5	ns
	Pixel Frequency	Cload = 20 pF			20****	MHz.
		Cload = 10 pF			40****	MHz.

(*) Output voltage matching of the R,G and B levels on a single device for each of the 8 levels
(**) Phase matching (50% point on both rise & fall time) on R, G, B, FB lines (FB in DAC mode)
(***) Phase matching (50% point on both rise & fall time) on R, G, B, FB lines (FB in digital mode)
(****) 95% of the signal amplitude is reached within the specified clock period

AC ELECTRICAL CHARACTERISTICS (Cont'd)

	I^2C Interface Electrical specifications					
Symbol	**Parameter**	**Standard mode I2C**		**Fast mode I2C**		**Unit**
		Min	**Max**	**Min**	**Max**	
V_{IL}	Low level input voltage: fixed input levels V_{DD}-related input levels	 -0.5 -0.5	 1.5 0.3 V_{DD}	 -0.5 -0.5	 1.5 0.3 V_{DD}	V
V_{IH}	High level input voltage: V_{DD}-related input levels	 0.8 V_{DD}	 V_{DD}+0.5	 0.8 V_{DD}	 V_{DD}+0.5	V
V_{HYS}	Hysteresis of Schmitt trigger inputs fixed input levels V_{DD}-related input levels	 N/A N/A	 N/A N/A	 0.2 0,05 V_{DD}		V
T_{SP}	Pulse width of spikes which must be suppressed by the input filter	N/A	N/A	0 ns	50 ns	ns
 V_{OL1} V_{OL2}	Low level output voltage (open drain and open collector) at 3 mA sink current at 6 mA sink current	 0 N/A	 0.4 N/A	 0 0	 0.4 0.6	V
T_{OF}	Output fall time from VIH min to VIL max with a bus capacitance from 10 pF to 400 pF with up to 3 mA sink current at V_{OL1} with up to 6 mA sink current at V_{OL2}	 N/A	 250 N/A	 20+0.1Cb 20+0.1Cb	 250 250	ns
I	Input current each I/O pin with an input voltage between 0.4V and 0.9 V_{DD} max	- 10	10	-10	10	µA
C	Capacitance for each I/O pin		10		10	pF

N/A = not applicable

Cb = capacitance of one bus in pF

AC ELECTRICAL CHARACTERISTICS (Cont'd)

	I^2C Bus Timings					
Symbol	**Parameter**	**Standard I^2C**		**Fast I^2C**		**Unit**
		Min	**Max**	**Min**	**Max**	
T_{BUF}	Bus free time between a STOP and START condition	4.7		1.3		ms
$T_{HD:STA}$	Hold time START condition. After this period, the first clock pulse is generated	4.0		0.6		µs
T_{LOW}	LOW period of the SCL clock	4.7		1.3		µs
T_{HIGH}	HIGH period of the SCL clock	4.0		0.6		µs
$T_{SU:STA}$	Set-up time for a repeated START condition	4.7		0.6		µs
$T_{HD:DAT}$	Data hold time	0 (1)		0 (1)	0.9(2)	ns
$T_{SU:DAT}$	Data set-up time	250		100		ns
T_R	Rise time of both SDA and SCL signals		1000	20+0.1Cb	300	ns
TF	Fall time of both SDA and SCL signals		300	20+0.1Cb	300	ns
$T_{SU:STO}$	Set-up time for STOP condition	4.0		0.6		ns
Cb	Capacitive load for each bus line		400		400	pF

1)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

2)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

Cb = total capacitance of one bus line in pF

Table 45. Characteristics of Analog Input Section

(V_{DD} = 5V, TA = -10°C to 75°C)

Parameter	**Value**			**Unit**
	Min.	**Typ.**	**Max.**	
Voltage comparator reference voltage:				
Unit#1 - Video black level clamp	1.90	2.00	2.10	V
Unit#2 - Data slicer (**)	2.25	2.35	2.45	V
Unit#3 - Sync slicer	1.70	1.80	1.90	V
Voltage comparator delay (all units) (*)	150	200	250	ns
Video clamp:				
Sink current (CCVIDEO pin at 2.1V DC)	21	42	80	µA
Source current (CCVIDEO pin at 1.9V DC)	150	300	600	µA
Sink to source current ratio	0.1	0.14	0.18	

Measurement conditions: (*) Same DC level on both comparator inputs AC level 40mV for applicated measurement signal (**) corresponds to 25 IRE tap voltage

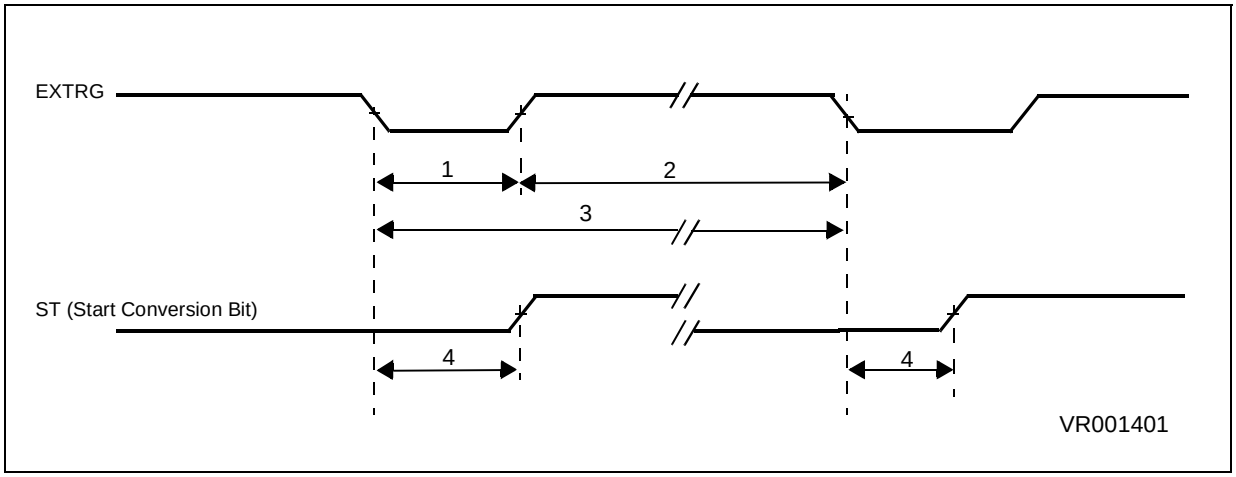
AC ELECTRICAL CHARACTERISTICS (Cont'd)

A/D CONVERTER, EXTERNAL TRIGGER TIMING TABLE

(V_{DD}= 5V +/-10%; T_A= -10°C to 75°C, unless otherwise specified)

N°	Symbol	Parameter	Conditions	Value		Unit
				min	max	
1	T_{low}	Pulse Width		1.5		INTCLK
2	T_{high}	Pulse Distance		1.5		INTCLK
3	T_{ext}	Period/fast Mode		78+1		INTCLK
4	T_{str}	Start Conversion Delay		0.5	1.5	INTCLK

A/D CONVERTER, EXTERNAL TRIGGER TIMING TABLE



A/D CONVERTER. ANALOG PARAMETERS TABLE

(V_{DD}= 5V +/-10% ; T_A= -10°C to 75°C, unless otherwise specified))

Parameter	Value			Unit	Note
	typ (*)	min	max	(**)	
Analog Input Range		V_{SS}	V_{DD}	V	
Conversion Time		138		INTCLK	(1,2)
Sample Time		87.51		INTCLK	(1)
Power-up Time		60		µs	
Resolution	8			bits	
Differential Non Linearity	0.5	0.3	1.5	LSBs	(4)
Integral Non Linearity			2	LSBs	(4)
Absolute Accuracy			2	LSBs	(4)
Input Resistance			1.5	Kohm	(3)
Hold Capacitance			1.92	pF	

Notes:
(*) The values are expected at 25 Celsius degrees with V_{DD}= 5V
(**)'LSBs' , as used here, as a value of V_{DD}/256
(1) @ 24 MHz external clock
(2) including Sample time
(3) it must be considered as the on-chip series resistance before the sampling capacitor
(4) DNL ERROR= max {[V(i) -V(i-1)] / LSB-1}INL ERROR= max {[V(i) -V(0)] / LSB-i}
ABSOLUTE ACCURACY= overall max conversion error

AC ELECTRICAL CHARACTERISTICS (Cont'd)

LATCH-UP AND ESD

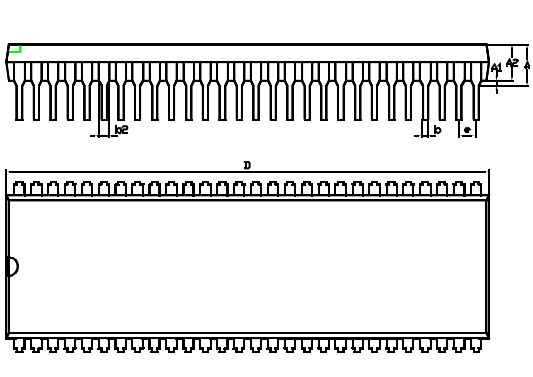
Parameter	Conditions	Value	Unit
ESD Sensitivity	for $\pm$ 10µA	4	kV
	for $\pm$ 1µA	2	
Latch-up performance	STMicroelectronics specification for Class A	No Latch-Up	

PPM REQUIREMENTS

Parameter	Conditions	Value	Unit
PPM Requirements		100	ppm

10 PACKAGE DESCRIPTION

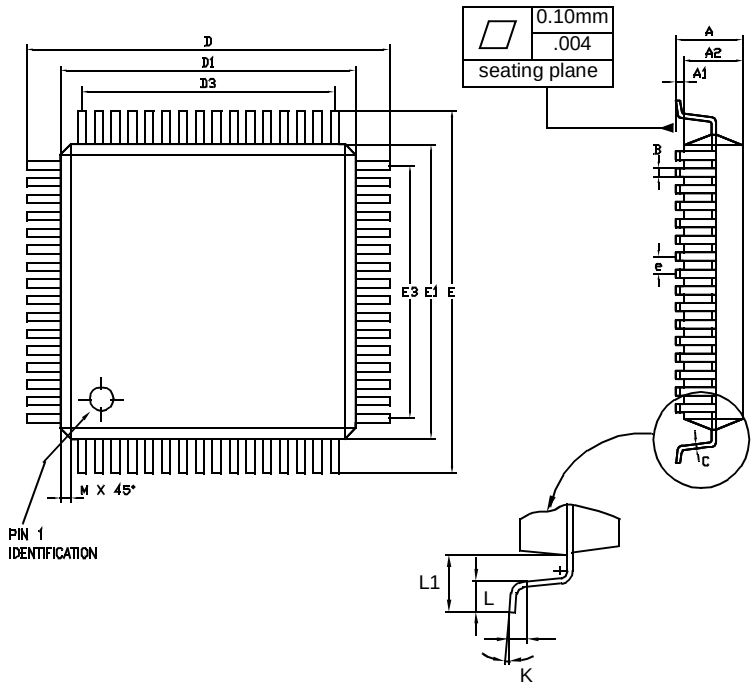
Figure 99. 56-Pin Shrink Plastic Dual In Line Package, 600-mil Width



PDIP56S

Dim.	mm			inches		
	Min	Typ	Max	Min	Typ	Max
A			6.35			0.250
A1	0.38			0.015		
A2	3.18		4.95	0.125		0.195
b		0.41			0.016	
b2		0.89			0.035	
C	0.20		0.38	0.008		0.015
D	50.29		53.21	1.980		2.095
E		15.01			0.591	
E1	12.32		14.73	0.485		0.580
e		1.78			0.070	
eA		15.24			0.600	
eB			17.78			0.700
L	2.92		5.08	0.115		0.200
	Number of Pins					
N	56					

Figure 100. 64-Pin Thin Quad Flat Package

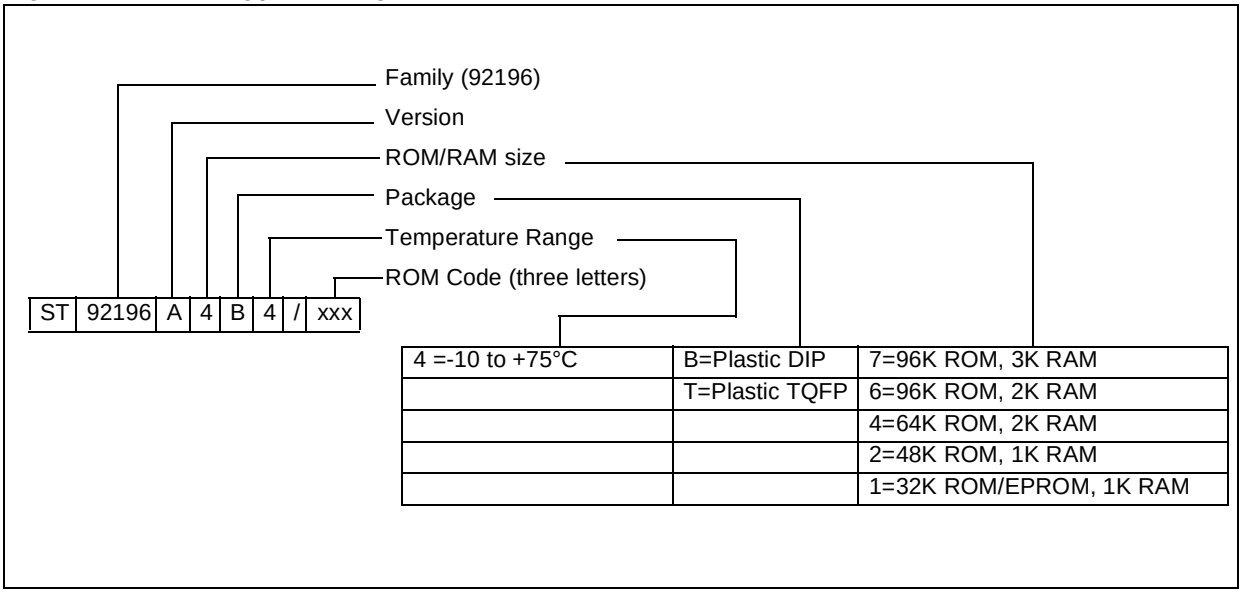


Dim	mm			inches		
	Min	Typ	Max	Min	Typ	Max
A			1.60			0.063
A1	0.05		0.15	0.002		0.006
A2	1.35	1.40	1.45	0.053	0.055	0.057
B	0.30	0.37	0.45	0.012	0.015	0.018
C	0.09		0.20	0.004		0.008
D		16.00			0.630	
D1		14.00			0.551	
D3		12.00			0.472	
E		16.00			0.630	
E1		14.00			0.551	
E3		12.00			0.472	
e		0.80			0.031	
K	0°	3.5°	7°			
L	0.45	0.60	0.75	0.018	0.024	0.030
L1		1.00			0.039	
	Number of Pins					
N	64		ND	16	NE	16

11 ORDERING INFORMATION

Device	ROM (Kbytes)	RAM (Kbytes)	Data Slicers	SCI	MFT
ST92196A7	96	3	2	1	1
ST92196A6		2	2	1	1
ST92196A4	64		2	-	1
ST92196A3			1	-	-
ST92196A2	48	1	1	-	-
ST92196A1	32		1	-	-

Figure 101. Sales Type Coding Rules



ORDERING INFORMATION

ST92196 OPTION LIST

Customer: . . . . . .
Address: . . . . . .
. . . . . .
Contact: . . . . . .
Phone No: . . . . . .
Reference/ROM Code* : . . . . . .
*The ROM code name is assigned by STMicroelectronics.

Please confirm characteristics of device:

Device:
[ ] ST92196A1
[ ] ST92196A2
[ ] ST92196A3
[ ] ST92196A4
[ ] ST92196A6
[ ] ST92196A7

Package:
[ ] PSDIP56
[ ] TQFP64

Temperature Range: -10°C to 75 °C

OSD Code: [ ] OSD filename _ _ _ _ _ _ _ _ _.OSD

Special Marking: [ ] No [ ] Yes "_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _"
For marking, one line is possible with maximum 14 characters.
Authorized characters are letters, digits, '.', '-', '/' and spaces only.

Quantity forecast: [_ _ _ _ _ _ _] K units per year for a period of [_ _] years.
Preferred production start date: [_ _ _ _ _ _ _] (YYYY/MM/DD)

Customer Signature . . . . . .

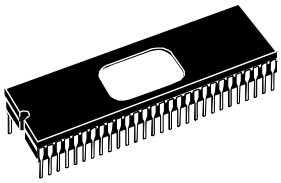
Date . . . . . .

ST92E196A/B & ST92T196A/B

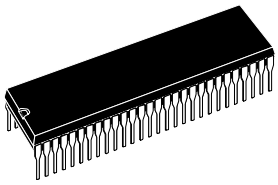
8/16-BIT MCU FOR TV APPLICATIONS WITH 128 K EPROM/OTP, ON-SCREEN-DISPLAY AND 1 OR 2 DATA SLICERS

DATASHEET

- Register file based 8/16 bit Core Architecture with RUN, WFI, and HALT modes
- -10 to 75°C Operating Temperature Range
- 24 MHz Operation @5V ±10%
- Min. instruction cycle time: 165 ns at 24 MHz
- 128 Kbytes EPROM/OTP, 3 or 4 Kbytes static RAM
- 256 bytes of Register file
- 384 bytes of display RAM (OSDRAM)
- 56-pin Shrink DIP packages
- 37 fully programmable I/O pins
- Flexible Clock controller for OSD, Data slicer and Core clocks, running from one single low frequency external crystal
- Enhanced Display Controller with rows of up to 63 characters per row
  - 50/60Hz and 100/120 Hz operation
  - 525/625 lines operation, 4:3 or 16:9 format
  - interlaced and progressive scanning
  - 18x26 or 9x13 character matrix
  - 384 (18x26) characters, or 1536 (9x13) characters definable in ROM by user
  - 512 possible colors, in 4x16-entry palettes
  - 2 x 16-entry palettes for Foreground, and 2 x 16-entry palettes for Background
  - 8 levels of translucency on Fast Blanking
  - Serial, Parallel and Extended Parallel Attribute modes
  - Mouse pointers user-definable in ROM
  - 7 character sizes in 18x26 mode, 4 in 9x13
  - Rounding, Fringe, Scrolling, Flashing, Shadowing, Italics, Semi-transparent
- I^2C Multi-Master / Slave with 4 channels
- Serial Communications Interface (SCI)
- Serial Peripheral Interface (SPI)
- 8-channel A/D converter with 6-bit accuracy
- 16-bit Watchdog timer with 8-bit prescaler
- 14-bit Voltage Synthesis for tuning reference voltage with 2 outputs for 2 tuners
- 16-bit standard timer with 8-bit prescaler
- 16-bit Multi-Function timer
- Eight 8-bit programmable PWM outputs



CSDIP56W



PSDIP56

- NMI and 6 external interrupts
- 2 Data Slicers for Closed Captioning and Extended Data Service data extraction, on 2 independent video sources. Support for FCC V-Chip and Gemstar bitstream decoding
- Infra-Red signal digital pre-processor
- 2-channel Sync error detection with integrated Sync extractor
- Rich instruction set and 14 addressing modes
- Versatile Development Tools, including C-Compiler, Assembler, Linker, Source Level Debugger, Emulator and Real-Time Operating Systems from third-parties
- Windows Based OSD Font and Screen Editor

DEVICE SUMMARY

Device	Memory	RAM	Prog. OSD Font
ST92E196A9	128K EPROM	4K	No
ST92T196A9	128K OTP		
ST92E196B7	128K EPROM	3K	Yes
ST92T196B7	128K OTP		

Rev. 3.1

1GENERAL DESCRIPTION

1.1 INTRODUCTION

The ST92E196A/B and ST92T196A/B microcontrollers are the EPROM/OTP versions of the ST92196A ROM devices and are suitable for product prototyping and low volume production. Their 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 ST92E196A/B and ST92T196A/B devices support low power consumption and low voltage operation for power-efficient and low-cost embedded systems.

1.1.1 Core Architecture

The nucleus of the ST92196A/B is the enhanced ST9 Core that includes the Central Processing Unit (CPU), the register file, the interrupt and DMA controller.

Three independent buses are controlled by the Core: a 16-bit memory bus, an 8-bit register addressing 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. Many opcodes specify byte or word operations, the hardware automatically handles 16-bit operations and accesses.

For interrupts or subroutine calls, the CPU uses a system stack in conjunction with the stack pointer (SP). A separate user stack has its own SP. The separate stacks, without size limitations, can be in on-chip RAM (or in Register File) or off-chip memory.

1.1.2 Instruction Set

The ST9 instruction set consists of 94 instruction types, including instructions for bit handling, byte (8-bit) and word (16-bit) data, as well as BCD and Boolean formats. Instructions have been added to facilitate large program and data handling through the MMU, as well as to improve the performance and code density of C Function calls. 14 addressing modes are available, including powerful indirect addressing capabilities.

The ST9's bit-manipulation instructions are set, clear, complement, test and set, load, and various logic instructions (AND, OR, and XOR). Math functions include add, subtract, increment, decrement, decimal adjust, multiply, and divide.

1.1.3 Operating Modes

To optimize performance versus the power consumption of the device, ST9 devices now support a range of 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).

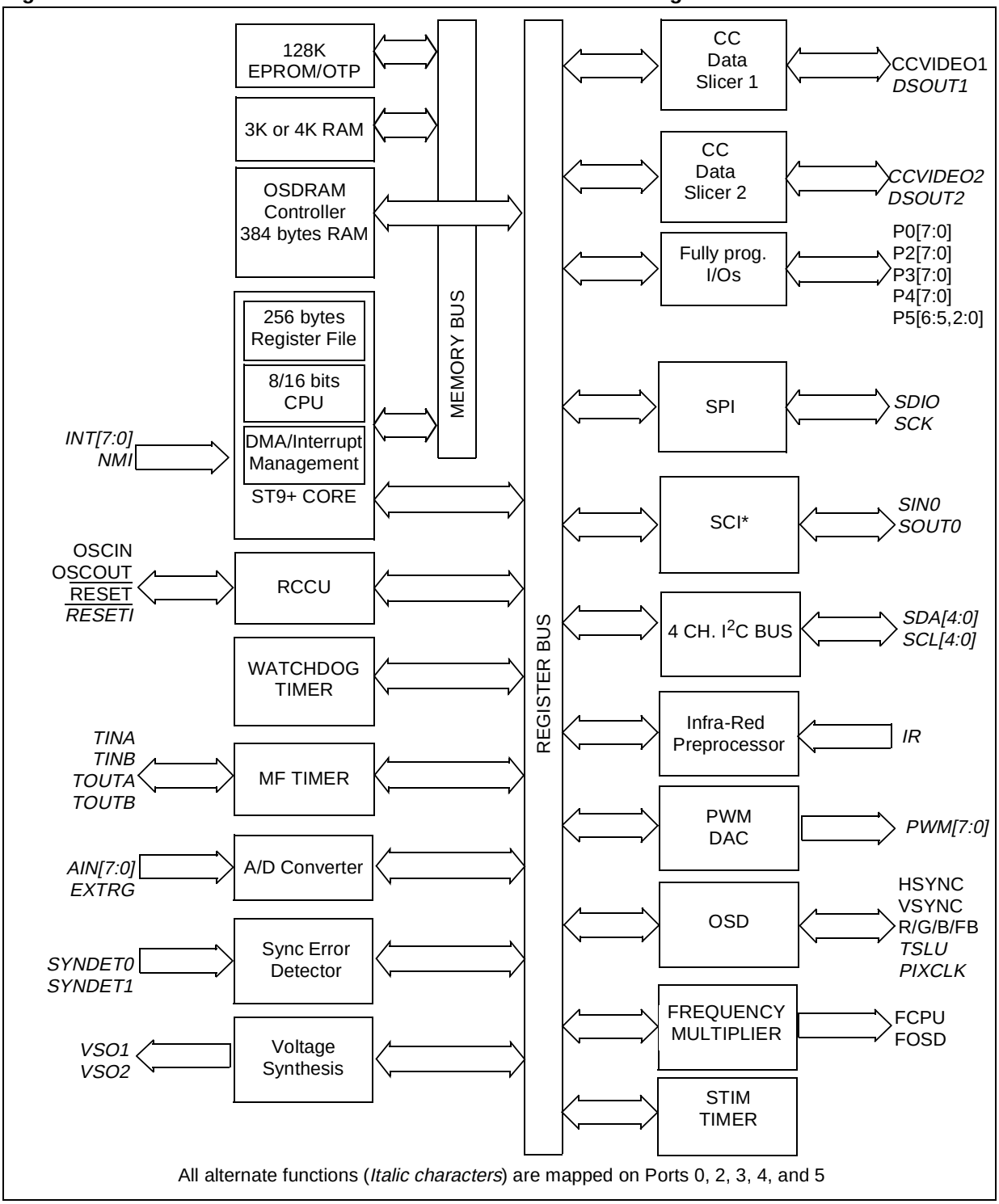
Slow Mode. Power consumption can be significantly reduced by running the CPU and the peripherals at reduced clock speed using the CPU Prescaler and CCU Clock Divider.

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 programmable via the CCU. In this mode, the power consumption of the device can be reduced by more than 95% (Low Power WFI).

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.

INTRODUCTION (Cont'd)

Figure 102. ST92E196A/B & ST92T196A/B Architectural Block Diagram



All alternate functions (*Italic characters*) are mapped on Ports 0, 2, 3, 4, and 5

INTRODUCTION (Cont'd)

1.1.4 On-chip Peripherals

OSD Controller

The On Screen Display displays closed caption or extended service format data received from the on-chip data slicers or any text or menu data generated by the application. Rows of up to 63 characters can be displayed with two user-definable fonts. Colors, character shape and other attributes are software programmable. Support is provided for mouse or other pointing devices.

Parallel I/O Ports

The ST9 is provided with dedicated lines for input/output. These lines, grouped into 8-bit ports, can be independently programmed to provide parallel input/output or to carry input/output signals to or from the on-chip peripherals and core e.g. SCI and Multifunction Timer. All ports have active pull-ups and pull-down resistors compatible with TTL loads. In addition pull-ups can be turned off for open drain operation and weak pull-ups can be turned on to save chip resistive pull-ups. Input buffers can be either TTL or CMOS compatible.

Multifunction Timer

The multifunction timer has a 16-bit Up/Down counter supported by two 16-bit Compare registers and two 16-bit input capture registers. Timing resolution can be programmed using an 8-bit prescaler.

Serial Communications Controller

The SCI provides an asynchronous serial I/O port using two DMA channels. Baud rates and data formats are programmable. Controller applications can further benefit from the self test and address wake-up facility offered by the character search mode.

I^2C Bus Interface

The I^2C bus is a synchronous serial bus for connecting multiple devices using a data line and a clock line. Multimaster and slave modes are supported. Up to four channels are supported. The I^2C interface supports 7-bit addressing. It operates in multimaster or slave mode and supports speeds of up to 666.67 kHz. Bus events (Bus busy, slave address recognised) and error conditions are automatically flagged in peripheral registers and interrupts are optionally generated.

Analog/Digital Converter

The ADC provides up to 8 analog inputs with on-chip sample and hold. Conversion can be triggered by a signal from the MFT.

1.2 PIN DESCRIPTION

Figure 103. 56-Pin Package Pin-Out

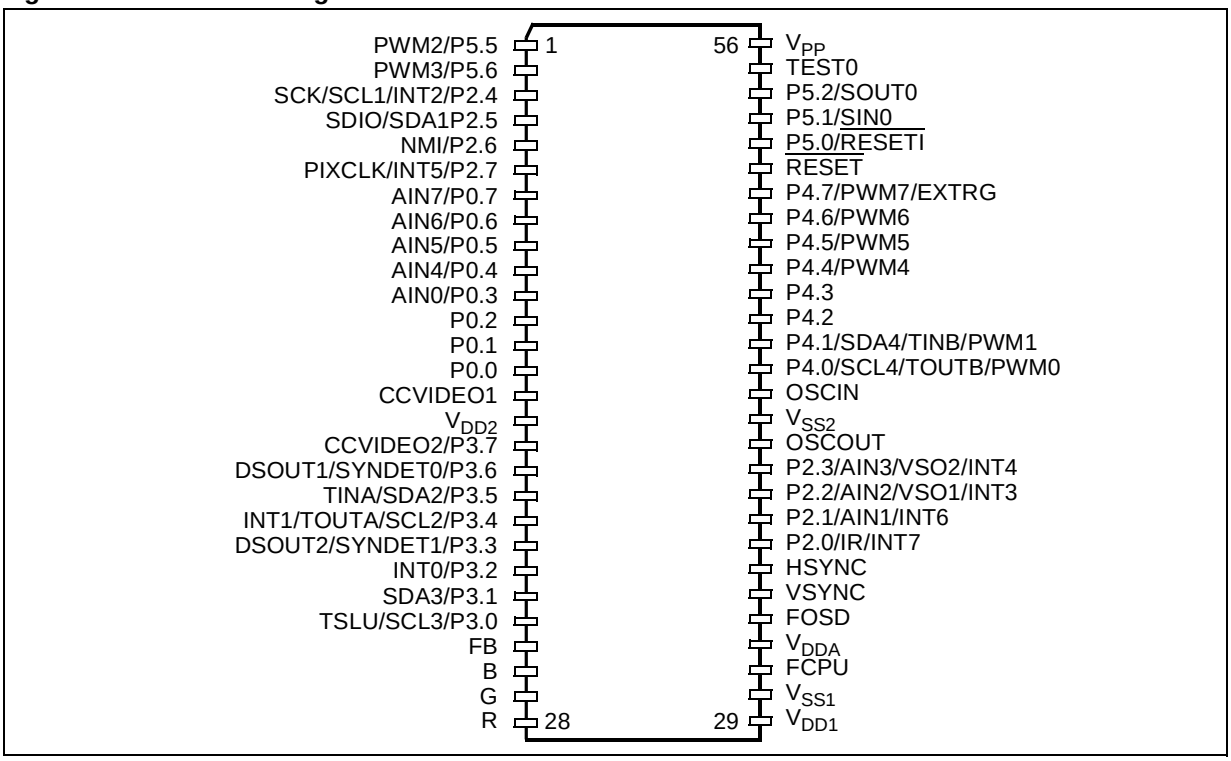


Table 46. Power Supply Pins

Name	Function	SDIP56
V_{DD1}	Main Power Supply Voltage	29
V_{DD2}	(2 pins internally connected)	16
V_{SS1}	Analog and Digital Circuit Ground	30
V_{SS2}	(2 pins internally connected)	41
V_{DDA}	Analog Circuit Supply Voltage	32
V_{PP}	EPROM Programming Voltage. Must be connected to V_{DD} in normal operating mode.	56

Table 47. Primary Function Pins

Name	Function	SDIP56
OSCIN	Oscillator input	42
OSCOUT	Oscillator output	40
$\overline{RESET}$	Reset to initialize the ST9	51
HSYNC	Video Horizontal Sync Input (Schmitt trigger)	35
VSYNC	Video Vertical Sync input (Schmitt trigger)	34
R	Red video analog DAC output	28
G	Green video analog DAC output	27
B	Blue video analog DAC output	26
FB	Fast Blanking analog DAC output	25
CCVIDEO1	Closed Caption Composite Video input 1 (2V +/- 3 dB)	15
FCPU	CPU frequency multiplier filter output	31
FOSD	OSD frequency multiplier filter output	33
TEST0	Test input (must be tied to V_{DD})	55

PIN DESCRIPTION (Cont'd)

1.2.1 I/O Port Configuration

All ports can be individually configured as input, bidirectional, output, or alternate function. Refer to the Port Bit Configuration Table in the I/O Port Chapter.

No I/O pins have any physical weak pull-up capability (they will show no pull-up if they are programmed in the "weak pull-up" software mode).

Input levels can be selected on a bit basis by choosing between TTL or CMOS input levels for I/O port pin except for P2.(5:4,0), P3.(6:3,1:0), P4.(1:0) which are implemented with a Schmitt trigger function.

All port output configurations can be software selected on a bit basis to provide push-pull or open drain driving capabilities. For all ports, when configured as open-drain, the voltage on the pin must never exceed the V_{DD} power line value (refer to Electrical characteristics section).

1.2.2 I/O Port Reset State

I/Os are reset asynchronously as soon as the $\overline{\text{RESET}}$ pin is asserted low.

All I/O are forced by the Reset in bidirectional, high impedance output due to the lack of physical pull-up except P5.0 (refer to the Reset section) which is forced into the "Push-Pull Alternate Function" mode until being reconfigured by software.

Warning

When a common pin is declared to be connected to an alternate function input and to an alternate function output, the user must be aware of the fact that the alternate function output signal always inputs to the alternate function module declared as input.

When any given pin is declared to be connected to a digital alternate function input, the user must be aware of the fact that the alternate function input is always connected to the pin. When a given pin is declared to be connected to an analog alternate function input (ADC input for example) and if this pin is programmed in the "AF-OD" mode, the digital input path is disconnected from the pin to prevent any DC consumption.

Table 48. I/O Port Characteristics

	Input	Output	Weak Pull-Up	Reset State
Port 0[7:0]	TTL/CMOS	Push-Pull/OD	No	Bidirectional
Port 2.0	Schmitt trigger	Push-Pull/OD	No	Bidirectional
Port 2[3:1]	TTL/CMOS	Push-Pull/OD	No	Bidirectional
Port 2[5:4]	Schmitt trigger	Push-Pull/OD	No	Bidirectional
Port 2[7:6]	TTL/CMOS	Push-Pull/OD	No	Bidirectional
Port 3.0	Schmitt trigger	Push-Pull/OD	No	Bidirectional
Port 3.1	Schmitt trigger	Push-Pull/OD	No	Bidirectional
Port 3.2	TTL/CMOS	Push-Pull/OD	No	Bidirectional
Port 3[6:3]	Schmitt trigger	Push-Pull/OD	No	Bidirectional
Port 3.7	TTL/CMOS	Push-Pull/OD	No	Bidirectional
Port 4.[1:0]	Schmitt trigger	Push-Pull/OD	No	Bidirectional
Port 4.[7:2]	TTL/CMOS	Push-Pull/OD	No	Bidirectional
Port 5.0	TTL/CMOS	Push-Pull/OD	No	Push-Pull AF Out
Port 5[6:1]	TTL/CMOS	Push-Pull/OD	No	Bidirectional

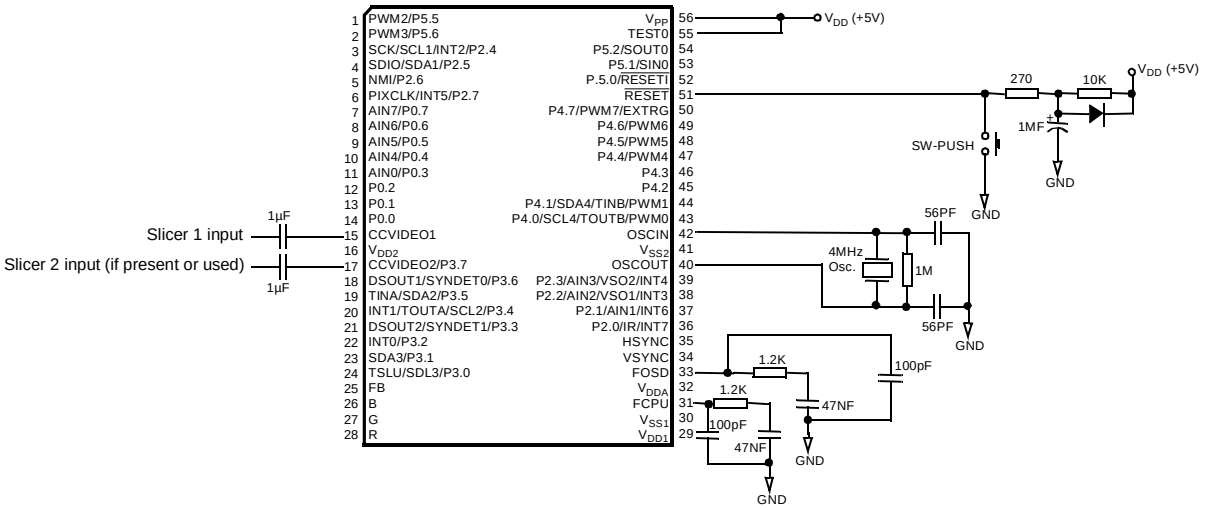
Legend: OD = Open Drain, AF = Alternate Function

Table 49. I/O Port Alternate Functions

Port Name	General Purpose I/O	Pin No. SDIP56	Alternate Functions		
P0.0	All ports useable for general purpose I/O (input, output or bidirectional)	14		I/O	
P0.1		13		I/O	
P0.2		12		I/O	
P0.3		11	AIN0	I	A/D Analog Data Input 0
P0.4		10	AIN4	I	A/D Analog Data Input 4
P0.5		9	AIN5	I	A/D Analog Data Input 5
P0.6		8	AIN6	I	A/D Analog Data Input 6
P0.7		7	AIN7	I	A/D Analog Data Input 7
P2.0		36	IR	I	IFR Infrared Input
			INT7	I	External Interrupt 7
P2.1		37	AIN1	I	A/D Analog Data Input 1
			INT6	I	External Interrupt 6
P2.2		38	INT3	I	External Interrupt 3
			AIN2		A/D Analog Data Input 2
			VSO1	O	Voltage Synthesis Converter Output 1
P2.3		39	INT4	I	External Interrupt 4
			AIN3	I	A/D Analog Data Input 3
			VSO2	O	Voltage Synthesis Converter Output 2
P2.4		3	INT2	I	External Interrupt 2
			SCL1	I/O	I^2C Channel 1 Serial Clock
			SCK	O	SPI Serial Clock Output
P2.5		4	SDIO	I/O	SPI Serial Data
			SDA1	I/O	I^2C Channel 1 Serial Data
P2.6		5	NMI	I	Non Maskable Interrupt Input
P2.7		6	INT5	I	External Interrupt 5
			PIXCLK	O	Pixel Clock (after divide-by-2) Output
P3.0		24	SCL3	I/O	I^2C Channel 3 Serial Clock
			TSLU	O	Translucency Digital Video Output
P3.1		23	SDA3	I/O	I^2C Channel 3 Serial Data
P3.2		22	INT0	I	External Interrupt 0
P3.3		21	SYNDET1	I	Sync Error Detector Input 1
			DSOUT2	O	Data Slicer Comparator Output 2
P3.4		20	INT1	I	External Interrupt 1
			SCL2	I/O	I^2C Channel 2 Serial Clock
			TOUTA	O	MFT Timer output A
P3.5		19	TINA	I	MFT Timer input A
			SDA2	I/O	I^2C Channel 2 Serial Data

Port Name	General Purpose I/O	Pin No. SDIP56	Alternate Functions		
P3.6	All ports useable for general purpose I/O (input, output or bidirectional)	18	SYNDET0	I	Sync Error Detector Input 0
			DSOUT1	O	Data Slicer Comparator Output 1
P3.7		17	CCVIDEO2	I	Closed Caption Composite Video input 1 (2V +/- 3 dB)
P4.0		43	SCL4	I/O	I^2C Channel 4 Serial Clock
			TOUTB	O	MFT Timer output B
			PWM0	O	PWM D/A Converter Output 0
P4.1		44	TINB	I	MFT Timer input B
			SDA4	I/O	I^2C Channel 4 Serial Data
			PWM1	O	PWM D/A Converter Output 1
P4.2		45		I/O	
P4.3		46		I/O	
P4.4		47	PWM4	O	PWM D/A Converter Output 4
P4.5		48	PWM5	O	PWM D/A Converter Output 5
P4.6		49	PWM6	O	PWM D/A Converter Output 6
P4.7		50	EXTRG	I	A/D Converter External Trigger Input
			PWM7	O	PWM D/A Converter Output 7
P5.0		52	RESETI	O	Internal Delayed Reset Output
P5.1		53	SIN0	I	SCI Serial Comm. Interface Input
P5.2		54	SOUT0	O	SCI Serial Comm. Interface Output
P5.5		1	PWM2	O	PWM D/A Converter Output 2
P5.6		2	PWM3	O	PWM D/A Converter Output 3

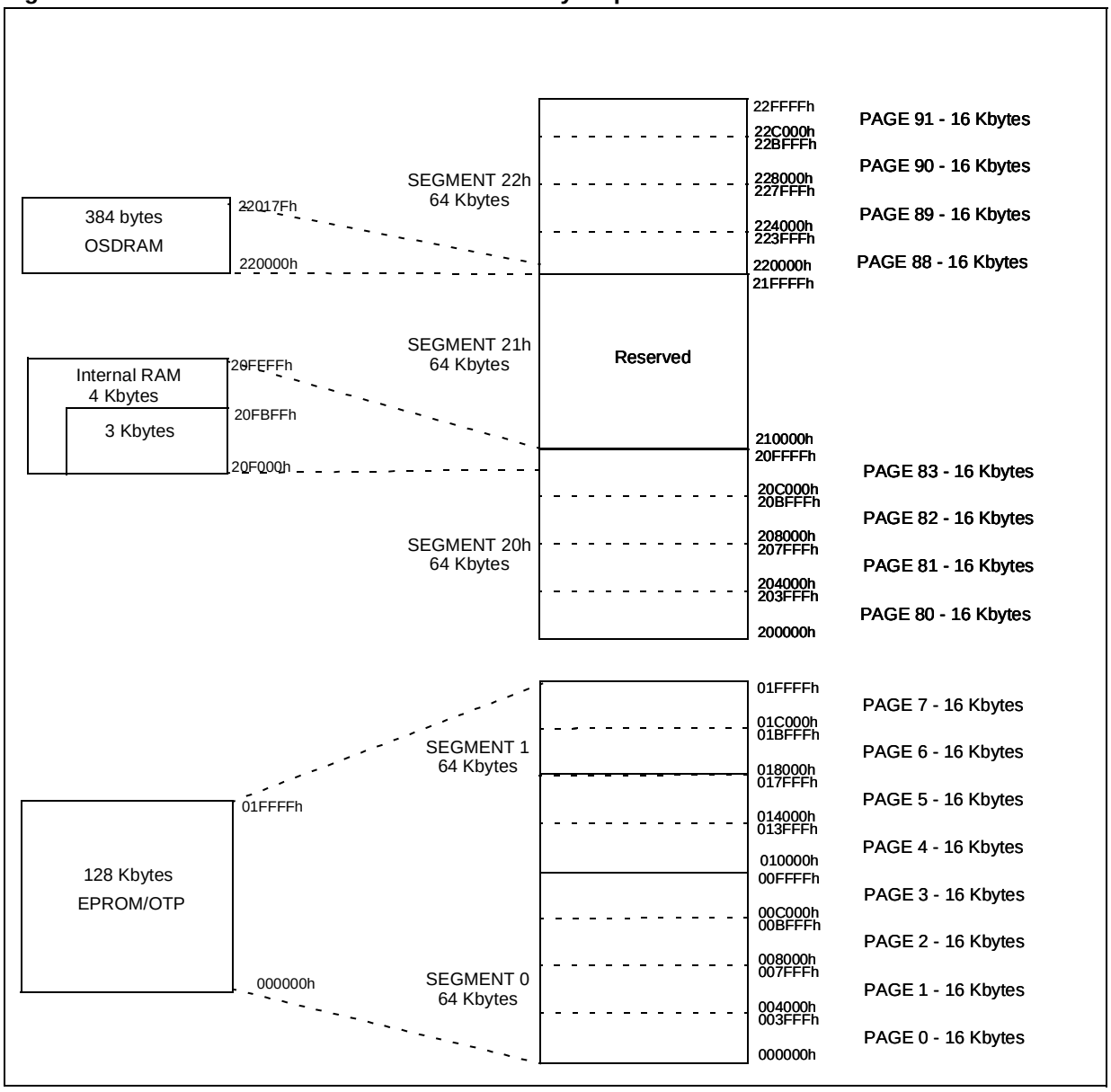
1.3 REQUIRED EXTERNAL COMPONENTS



Warning : The decoupling capacitors between analog and digital +5V (V_{DDA}, V_{DD1}, V_{DD2}), and ground (V_{SS1}, V_{SS2}) are not shown. Add a 100NF and a 4.7µF capacitor close to the corresponding pins if needed.

1.4 MEMORY MAP

Figure 104. ST92E196A/B & ST92T196A/B Memory Map



1.5 ST92E196A/B & ST92T196A/B REGISTER MAP

Table 51 contains the map of the group F peripheral pages.

The common registers used by each peripheral are listed in Table 50.

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.

Warning: 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 50. Common Registers

Function or Peripheral	Common Registers
SCI, MFT	CICR + NICR + DMA REGISTERS + I/O PORT REGISTERS
ADC	CICR + NICR + I/O PORT REGISTERS
WDT	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

ST92196A/B REGISTER MAP (Cont'd)

Table 51. Group F Pages Register Map

Resources available on the ST92E196A/B & ST92T196A/B device:

Register	Page															
	0	2	3	9	10	11	21	24	42	43	44	45	46	55	59	62
R255	Res.	Res.	Res.	Res.	MFT	Res.	MMU	SCI0	Res.	Res.	Res.	Res.	Res.	Res.	VS	Res.
R254	SPI	Port 3								TCC						
R253															Res.	
R252	WCR								OSD	Res.					PWM	
R251	WDT	Res.								IR/ SYNC ERR						
R250		Port 2														
R249																
R248				MFT												
R247	EXT INT	Res.	Port 5	Res.					Res.	Res.	I2C	DS0	DS1	RCCU		
R246																
R245																
R244																
R243			Res.	MFT		STIM										
R242		Port 0	Port 4						Res.							ADC
R241	Res.															
R240																

ST92E196A/B/T196 REGISTER MAP (Cont'd)

Table 52. Detailed Register Map

Group F Page Dec.	Block	Reg. No.	Register Name	Description	Reset Value Hex.	Doc. Page
N/A	I/O Port 0:5	R224	P0DR	Port 0 Data Register	FF	69
		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	
	Core	R230	CICR	Central Interrupt Control Register	87	27
		R231	FLAGR	Flag Register	00	28
		R232	RP0	Pointer 0 Register	00	30
		R233	RP1	Pointer 1 Register	00	30
		R234	PPR	Page Pointer Register	54	32
		R235	MODER	Mode Register	E0	32
		R236	USPHR	User Stack Pointer High Register	xx	34
		R237	USPLR	User Stack Pointer Low Register	xx	34
		R238	SSPHR	System Stack Pointer High Reg.	xx	34
		R239	SSPLR	System Stack Pointer Low Reg.	xx	34
0	INT	R242	EITR	External Interrupt Trigger Register	00	56
		R243	EIPR	External Interrupt Pending Reg.	00	56
		R244	EIMR	External Interrupt Mask-bit Reg.	00	56
		R245	EIPLR	External Interrupt Priority Level Reg.	FF	57
		R246	EIVR	External Interrupt Vector Register	x6	57
		R247	NICR	Nested Interrupt Control	00	57
	WDT	R248	WDTHR	Watchdog Timer High Register	FF	81
		R249	WDTLR	Watchdog Timer Low Register	FF	81
		R250	WDTPR	Watchdog Timer Prescaler Reg.	FF	81
		R251	WDTCR	Watchdog Timer Control Register	12	81
		R252	WCR	Wait Control Register	7F	82
	SPI	R253	SPIDR	SPI Data Register	xx	190
		R254	SPICR	SPI Control Register	00	190
2	I/O Port 0	R240	P0C0	Port 0 Configuration Register 0	00	69
		R241	P0C1	Port 0 Configuration Register 1	00	
		R242	P0C2	Port 0 Configuration Register 2	00	
	I/O Port 2	R248	P2C0	Port 2 Configuration Register 0	00	
		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	00	
		R253	P3C1	Port 3 Configuration Register 1	00	
		R254	P3C2	Port 3 Configuration Register 2	00	

Group F 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	00	69
		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	00	
		R245	P5C1	Port 5 Configuration Register 1	00	
		R246	P5C2	Port 5 Configuration Register 2	00	
9	MFT	R240	DCPR	DMA Counter Pointer Register	xx	108
		R241	DAPR	DMA Address Pointer Register	xx	109
		R242	T_IVR	Interrupt Vector Register	xx	109
		R243	IDCR	Interrupt/DMA Control Register	C7	110
		R248	IOCR	I/O Connection Register	FC	110
10		R240	REG0HR	Capture Load Register 0 High	xx	101
		R241	REG0LR	Capture Load Register 0 Low	xx	101
		R242	REG1HR	Capture Load Register 1 High	xx	101
		R243	REG1LR	Capture Load Register 1 Low	xx	101
		R244	CMP0HR	Compare 0 Register High	00	101
		R245	CMP0LR	Compare 0 Register Low	00	101
		R246	CMP1HR	Compare 1 Register High	00	101
		R247	CMP1LR	Compare 1 Register Low	00	101
		R248	TCR	Timer Control Register	0x	102
		R249	TMR	Timer Mode Register	00	103
		R250	T_ICR	External Input Control Register	0x	104
		R251	PRSR	Prescaler Register	00	104
		R252	OACR	Output A Control Register	xx	105
		R253	OBCR	Output B Control Register	xx	106
		R254	T_FLAGR	Flags Register	00	107
		R255	IDMR	Interrupt/DMA Mask Register	00	108
11	STIM	R240	STH	Counter High Byte Register	FF	86
		R241	STL	Counter Low Byte Register	FF	86
		R242	STP	Standard Timer Prescaler Register	FF	86
		R243	STC	Standard Timer Control Register	14	86
21	MMU	R240	DPR0	Data Page Register 0	00	39
		R241	DPR1	Data Page Register 1	01	39
		R242	DPR2	Data Page Register 2	02	39
		R243	DPR3	Data Page Register 3	83	39
		R244	CSR	Code Segment Register	00	40
		R248	ISR	Interrupt Segment Register	x0	40
		R249	DMASR	DMA Segment Register	x0	40
	EXTMI	R246	EMR2	External Memory Register 2	0F	58

Group F Page Dec.	Block	Reg. No.	Register Name	Description	Reset Value Hex.	Doc. Page
24	SCI0	R240	RDCPR	Receiver DMA Transaction Counter Pointer	xx	201
		R241	RDAPR	Receiver DMA Source Address Pointer	xx	201
		R242	TDCPR	Transmitter DMA Transaction Counter Pointer	xx	201
		R243	TDAPR	Transmitter DMA Destination Address Pointer	xx	201
		R244	S_IVR	Interrupt Vector Register	xx	202
		R245	ACR	Address/Data Compare Register	xx	203
		R246	IMR	Interrupt Mask Register	x0	203
		R247	S_ISR	Interrupt Status Register	xx	204
		R248	RXBR	Receive Buffer Register	xx	205
		R248	TXBR	Transmitter Buffer Register	xx	205
		R249	IDPR	Interrupt/DMA Priority Register	xx	206
		R250	CHCR	Character Configuration Register	xx	207
		R251	CCR	Clock Configuration Register	00	207
		R252	BRGHR	Baud Rate Generator High Reg.	xx	208
		R253	BRGLR	Baud Rate Generator Low Register	xx	208
		R254	SICR	Input Control	03	208
		R255	SOCR	Output Control	01	208
42	OSD	R246	OSDBCR2	Border Color Register 2	x0	152
		R247	OSDBCR1	Border Color Register 1	x0	152
		R248	OSDER	Enable Register	00	153
		R249	OSDDR	Delay Register	xx	156
		R250	OSDFBR	Flag Bit Register	xx	157
		R251	OSDSLR	Scan Line Register	xx	158
		R252	OSDMR	Mute Register	xx	158
43	IR/SYNC ERR	R248	IRPR	Infrared Pulse Register	00	169
		R249	SYNCER	Sync Error Register	00	168
		R250	IRSCR	Infrared / Sync Control Register	00	168
	TCC	R253	MCCR	Main Clock Control Register	00	69
		R254	SKCCR	Skew Clock Control Register	00	69
44	I2C	R240	I2COAR	Own Address Register	00	175
		R241	I2CFQR	Frequency Register	00	176
		R242	I2CCTR	Control Register	01	177
		R243	I2CDR	Data Register	00	178
		R244	I2CSTR2	Status Register 2	00	178
		R245	I2CSTR1	Status Register 1	00	179

Group F Page Dec.	Block	Reg. No.	Register Name	Description	Reset Value Hex.	Doc. Page
45	DS0	R240	DS0DR1	Data Register 1	00	164
		R241	DS0DR2	Data Register 2	00	164
		R242	DS0DR3	Data Register 3	00	164
		R243	DS0DR4	Data Register 4	00	165
		R244	DS0CR1	Control Register 1	00	165
		R245	DS0CR2	Control Register 2	00	165
		R246	DS0MR	Monitor Register	00	166
46	DS1	R240	DS1DR1	Data Register 1	00	164
		R241	DS1DR2	Data Register 2	00	164
		R242	DS1DR3	Data Register 3	00	164
		R243	DS1DR4	Data Register 4	00	165
		R244	DS1CR1	Control Register 1	00	165
		R245	DS1CR2	Control Register 2	00	165
		R246	DS1MR	Monitor Register	00	166
55	RCCU	R240	CLKCTL	Clock Control Register	00	64
		R242	CLK_FLAG	Clock Flag Register	48, 28 or 08	64
59	PWM	R240	CM0	Compare Register 0	00	216
		R241	CM1	Compare Register 1	00	216
		R242	CM2	Compare Register 2	00	216
		R243	CM3	Compare Register 3	00	216
		R244	CM4	Compare Register 4	00	216
		R245	CM5	Compare Register 5	00	216
		R246	CM6	Compare Register 6	00	216
		R247	CM7	Compare Register 7	00	216
		R248	ACR	Autoclear Register	FF	217
		R249	CCR	Counter Register	00	217
		R250	PCTL	Prescaler and Control Register	0C	217
		R251	OCPL	Output Complement Register	00	218
		R252	OER	Output Enable Register	00	218
	VS	R254	VSDR1	Data and Control Register 1	00	213
		R255	VSDR2	Data Register 2	00	213
62	ADC	R240	ADDTR	Channel i Data Register	xx	221
		R241	ADCLR	Control Logic Register	00	221
		R242	ADINT	AD Interrupt Register	01	222

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

2 ELECTRICAL CHARACTERISTICS

The ST92E196A/B & ST92T196A/B devices contain circuitry to protect the inputs against damage due to high static voltage or electric field. Nevertheless, it is advised to take normal precautions and to avoid applying to this high impedance voltage circuit any voltage higher than the maximum rated voltages. It is recommended for proper operation that V_{IN} and V_{OUT} be constrained to the range.

$V_{SS} \leq (V_{IN}$ or $V_{OUT}) \leq V_{DD}$

To enhance reliability of operation, it is recommended to connect unused inputs to an appropriate logic voltage level such as V_{SS} or V_{DD}.
All the voltages in the following table, are referenced to V_{SS}.

ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
V_{DD}1,2	Supply Voltage	V_{SS} – 0.3 to V_{SS} + 7.0	V
V_{DDA}	Analog Supply Voltage	V_{SS} – 0.3 to V_{DD} +0.3	V
V_I	Input Voltage	V_{SS} – 0.3 to V_{DD} +0.3	V
V_I	Input Voltage	V_{SS} – 0.7 to V_{DD} +0.7 *	V
V_O	Output Voltage	V_{SS} – 0.3 to V_{DD} +0.3	V
V_O	Output Voltage	V_{SS} – 0.7 to V_{DD} +0.7 *	V
T_{STG}	Storage Temperature	– 55 to + 150	°C
I_{INJ}	Pin Injection Current Digital and Analog Input	-5 to +5	mA
	Maximum Accumulated Pin injection Current in the device	-50 to +50	mA

*Current is limited to |<200µA| into the pin

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.

RECOMMENDED OPERATING CONDITIONS

Symbol	Parameter	Value		Unit
		Min.	Max.	
T_A	Operating Temperature	-10	75	°C
V_{DD}	Operating Supply Voltage	4.5	5.5	V
V_{DDa}	Analog Supply Voltage	4.5	5.5	V
f_{OSCE}	External Oscillator Frequency		4.0	MHz
f_{INTCLK}	Internal Clock Frequency	0[1]	24	MHz

Note 1. 1MHz when A/D is used
Note 2. For good slicing results, it is advised to set VDDA >= 5.3V. Remember also that VDDA < VDD + 0.3V.

DC ELECTRICAL CHARACTERISTICS

(V_{DD} = 5V ± 10% T_A = -10°C + 75°C, unless otherwise specified)

Symbol	Parameter	Test Conditions	Value		Unit
			Min.	Max.	
V_{IH}	Input High Level	TTL	2.0		V
		CMOS	0.7 V_{DD}		V
V_{IL}	Input Low Level	TTL		0.8	V
		CMOS		0.3 V_{DD}	V
V_{IHRS}	$\overline{\text{RESET}}$ Input High Level		0.8 V_{DD}		V
V_{ILRS}	$\overline{\text{RESET}}$ Input Low Level			0.3 V_{DD}	V
V_{HYRS}	$\overline{\text{RESET}}$ Input Hysteresis		0.5		V
V_{IH}	P2.0 Input High Level		0.8 V_{DD}		V
V_{IL}	P2.0 Input Low Level			0.3 V_{DD}	V
V_{IHY}	P2.0 Input Hysteresis		0.5		V
V_{IHIIC}	SDA4,1/SCL4,1 Input High Level		0.8 V_{DD}		V
V_{ILIIC}	SDA4,1/SCL4,1 Input Low Level			0.3 V_{DD}	V
V_{IHYIIC}	SDA4,1/SCL4,1 Hysteresis		0.5		V
V_{IHVH}	HSYNC/VSYNC Input High Level		0.8 V_{DD}		V
V_{ILVH}	HSYNC/VSYNC Input Low Level			0.3 V_{DD}	V
V_{HYHV}	HSYNC/VSYNC Input Hysteresis		1.2		V
V_{IHDT}	SYNDET1,0 Input High Level		0.8 V_{DD}		V
V_{ILDT}	SYNDET1,0 Input Low Level			0.3 V_{DD}	V
V_{HYDT}	SYNDET1,0 Input Hysteresis		0.5		V
V_{OH}	Output High Level	Push Pull, Iload = – 0.8mA	V_{DD} – 0.8		V
V_{OL}	Output Low Level	Push Pull or Open Drain, Iload = 1.6mA		0.4	V
V_{pp}	EPROM programming voltage			12.8	V
I_{LKIO}	I/O Pin Input Leakage Current	Hi-Z Input, 0V < V_{IN} < V_{DD}		± 1.0	µA
I_{LKRS}	$\overline{\text{RESET}}$ Pin Input Leakage Current	0V < V_{IN} < V_{DD}		± 1.0	µA
$I_{LKA/D}$	A/D Pin Input Leakage Current	Alternate function open drain		± 1.0	µA
I_{LKOS}	OSCIN Pin Input Leakage Current	0V < V_{IN} < V_{DD}		± 1.0	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. No peripheral working.

AC ELECTRICAL CHARACTERISTICS

PIN CAPACITANCE

(V_{DD} = 5V ± 10%; T_A = -10°C + 75°C, unless otherwise specified)

Symbol	Parameter	Conditions	Value		Unit
			Typ.	Max	
C_{IO}	Pin Capacitance Digital Input/Output		8	10	pF

CURRENT CONSUMPTION

(V_{DD} = 5V ± 10%; T_A = -10°C + 75°C, unless otherwise specified)

Symbol	Parameter	Conditions	Value		Unit
			Typ.	Max	
I_{CC1}	Run Mode Current	(Notes 1, 2) INTCLK=16MHz	80	100	mA
I_{CC2}	Run Mode Current	(Notes 1, 2) INTCLK=24MHz	100	120	mA
I_{CC3}	Run Mode Current	(Notes 1, 2) INTCLK=4MHz	25	30	mA
I_{CC4}	Run Mode Current	(Notes 1, 5) INTCLK=16MHz	65	78	mA
I_{CCA1}	Analog Current (V_{DDA} pin)	Freq. Multipliers , A/D, OSD, All DACs & Slicers On.	45	55	mA
I_{CCA2}	Analog Current (V_{DDA} pin)	Freq. Multipliers , A/D, OSD, DACs & Slicers Off.	1	10	µA
I_{CCA3}	Analog Current (V_{DDA} pin)	Freq. Multipliers , A/D, OSD, two DACs & one Slicer On.	25	30	mA
I_{ILPR}	Reset Mode Current	(Note 3)	10	100	µA
I_{HALT}	HALT Mode Current	(Note 4)	10	100	µA

Notes :

1. All ports are configured in push-pull output mode (output is high). VSYNC and HSYNC are tied to V_{SS}, CCVIDEO is floating. The internal clock prescaler is in divide-by-1 mode. The external CLOCK pin (OSCIN) is driven by a square wave external clock at 4 MHz.

2. The CPU is fed by a frequency issued by the on-chip Frequency Multiplier. The Skew Corrector Frequency Multiplier provides a 28 MHz clock. All peripherals are working.

3. All ports are configured in push-pull output mode (output is high). VSYNC and HSYNC are tied to V_{SS}, CCVIDEO is floating. External CLOCK pin (OSCIN) and Reset pins are held low. All peripherals are disabled.

4. All ports are configured in push-pull output mode (output is high). VSYNC and HSYNC are tied to V_{SS}, CCVIDEO is floating. All peripherals are disabled.

5. The CPU is fed by a frequency issued by the on-chip Frequency Multiplier. The Skew Corrector Frequency Multiplier provides a 14Mhz clock. OSD, A/D, PWM, Sync Error Detector, Std Timer and WDG Timer peripherals are running.

AC ELECTRICAL CHARACTERISTICS (Cont'd)

CLOCK TIMING

(V_{DD} = 5V ± 10% T_A = -10°C + 75°C, unless otherwise specified)

Symbol	Parameter	Conditions	Value		Unit
			Min	Max	
TpC	OSCIN Clock Period	intern. div. by 2	41.7		ns
		intern. div. by 1	83.3		ns
TrC	OSCIN rise time			12	ns
TfC	OSCIN fall time			12	ns
TwCL	OSCIN low width	intern. div. by 2	17		ns
		intern. div. by 1	38		ns
TwCH	OSCIN high width	intern. div. by 2	17		ns
		intern. div. by 1	38		ns

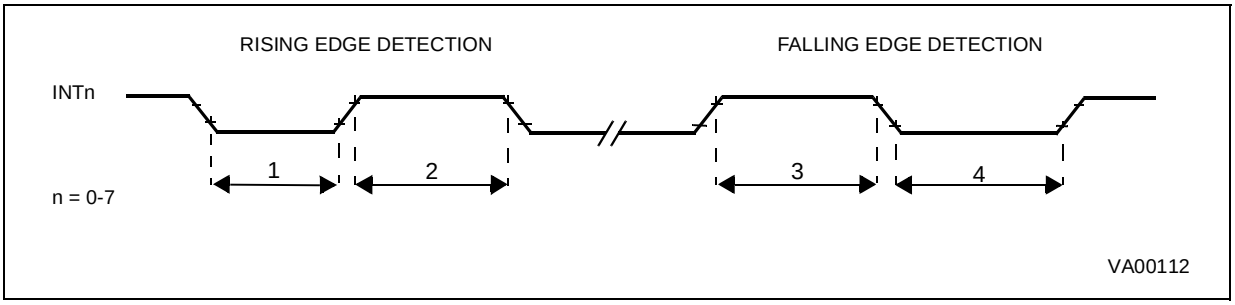
EXTERNAL INTERRUPT TIMING

(Rising or falling edge mode; V_{DD} = 5V ± 10%; T_A = -10°C + 75°C, unless otherwise specified)

N°	Symbol	Parameter	Conditions				Unit
			OSCIN Divided by 2 Min.	OSCIN Not Divided by 2 Min.	Min	Max	
1	TwLR	Low Level Minimum Pulse width in Rising Edge Mode	2TpC + 12	TpC + 12	95		ns
2	TwHR	High Level Minimum Pulse width in Rising Edge Mode	2TpC + 12	TpC + 12	95		ns
3	TwLF	Low Level Minimum Pulse width in Falling Edge Mode	2TpC + 12	TpC + 12	95		ns
4	TwHF	High Level Minimum Pulse width in Falling Edge Mode	2TpC + 12	TpC + 12	95		ns

Note: The value in the left hand two columns shows the formula used to calculate the minimum or maximum timing from the oscillator clock period, prescale value and number of wait cycles inserted. The value in the rignt hand two columns shows the minimum and maximum for an external clock at 24 MHz divided by 2, prescale value of zero and zero wait status.

EXTERNAL INTERRUPT TIMING



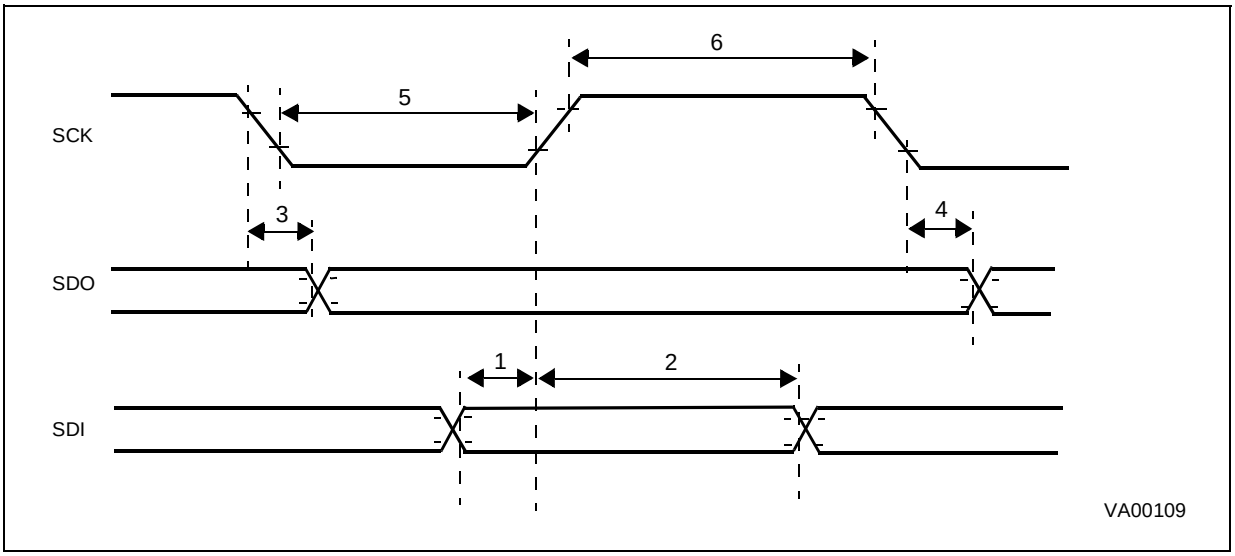
AC ELECTRICAL CHARACTERISTICS (Cont'd)

SPI TIMING

(V_{DD} = 5V ± 10% ; T_A = -10°C + 75°C, unless otherwise specified)

N°	Symbol	Parameter	Conditions	Value		Unit
				Min	Max	
1	TsDI	Input Data Set-up Time		100		
2	ThDI	Input Data Hold Time		1/2 TpC + 100		
3	TdOV	SCK to Output Data Valid			100	
4	ThDO	Output Data Hold Time		-20		
5	TwSKL	SCK Low Pulse Width		300		
6	TwSKH	SCK High Pulse Width		300		

SPI TIMING



SKEW CORRECTOR TIMING TABLE

(V_{DD} = 5V ± 10%; T_A = -10°C + 75°C, unless otherwise specified)

Symbol	Parameter	Conditions	Value Max	Unit
Tjskw	Jitter on RGB output	28 MHz Skew corrector clock frequency	<12 *	ns

The OSD jitter is measured from leading edge to leading edge of a single character row on consecutive TV lines. The value is an envelope of 100 fields
*Max. value at all CPU operating frequencies

AC ELECTRICAL CHARACTERISTICS (Cont'd)

OSD DAC CHARACTERISTICS

(V_{DD} = 5V ± 10%; T_A = -10°C + 75°C, unless otherwise specified)

Symbol	Parameter	Conditions	Value			Unit
			Min	Typ	Max	
	Output impedance FB,R,G,B				100	Ohm
	Output voltage FB,R,G,B	Cload = 20 pF RL=100K				
	code = 111		0.976	1.170	1.364	V
	code = 110		0.863	1.034	1.205	
	code = 101		0.751	0.899	1.046	
	code = 100		0.638	0.763	0.887	
	code = 011		0.525	0.627	0.729	
	code = 010		0.412	0.491	0.570	
	code = 001		0.300	0.356	0.411	
	code = 000		0.157	0.220	0.252	
	FB = 1 FB = 0			5.0 0.2		V
	Relative voltage accuracy	(*)			+/-5	%
	R/G/B to FB 50% point matching	FB DAC mode (**)			5	ns
	R/G/B to FB 50% point matching	FB digital mode (***)			5	ns
	Pixel Frequency	Cload = 20 pF			20****	MHz.
		Cload = 10 pF			40****	MHz.

(*) Output voltage matching of the R,G and B levels on a single device for each of the 8 levels
(**) Phase matching (50% point on both rise & fall time) on R, G, B, FB lines (FB in DAC mode)
(***) Phase matching (50% point on both rise & fall time) on R, G, B, FB lines (FB in digital mode)
(****) 95% of the signal amplitude is reached within the specified clock period

AC ELECTRICAL CHARACTERISTICS (Cont'd)

	I^2C Interface Electrical specifications					
Symbol	**Parameter**	**Standard mode I2C**		**Fast mode I2C**		**Unit**
		Min	**Max**	**Min**	**Max**	
V_{IL}	Low level input voltage: fixed input levels V_{DD}-related input levels	 -0.5 -0.5	 1.5 0.3 V_{DD}	 -0.5 -0.5	 1.5 0.3 V_{DD}	V
V_{IH}	High level input voltage: V_{DD}-related input levels	 0.8 V_{DD}	 V_{DD}+0.5	 0.8 V_{DD}	 V_{DD}+0.5	V
V_{HYS}	Hysteresis of Schmitt trigger inputs fixed input levels V_{DD}-related input levels	 N/A N/A	 N/A N/A	 0.2 0,05 V_{DD}		V
T_{SP}	Pulse width of spikes which must be suppressed by the input filter	N/A	N/A	0 ns	50 ns	ns
 V_{OL1} V_{OL2}	Low level output voltage (open drain and open collector) at 3 mA sink current at 6 mA sink current	 0 N/A	 0.4 N/A	 0 0	 0.4 0.6	V
T_{OF}	Output fall time from VIH min to VIL max with a bus capacitance from 10 pF to 400 pF with up to 3 mA sink current at V_{OL1} with up to 6 mA sink current at V_{OL2}	 N/A	 250 N/A	 20+0.1Cb 20+0.1Cb	 250 250	ns
I	Input current each I/O pin with an input voltage between 0.4V and 0.9 V_{DD} max	- 10	10	-10	10	µA
C	Capacitance for each I/O pin		10		10	pF

N/A = not applicable

Cb = capacitance of one bus in pF

AC ELECTRICAL CHARACTERISTICS (Cont'd)

	I^2C Bus Timings					
Symbol	**Parameter**	**Standard I^2C**		**Fast I^2C**		**Unit**
		Min	**Max**	**Min**	**Max**	
T_{BUF}	Bus free time between a STOP and START condition	4.7		1.3		ms
$T_{HD:STA}$	Hold time START condition. After this period, the first clock pulse is generated	4.0		0.6		µs
T_{LOW}	LOW period of the SCL clock	4.7		1.3		µs
T_{HIGH}	HIGH period of the SCL clock	4.0		0.6		µs
$T_{SU:STA}$	Set-up time for a repeated START condition	4.7		0.6		µs
$T_{HD:DAT}$	Data hold time	0 (1)		0 (1)	0.9(2)	ns
$T_{SU:DAT}$	Data set-up time	250		100		ns
T_R	Rise time of both SDA and SCL signals		1000	20+0.1Cb	300	ns
TF	Fall time of both SDA and SCL signals		300	20+0.1Cb	300	ns
$T_{SU:STO}$	Set-up time for STOP condition	4.0		0.6		ns
Cb	Capacitive load for each bus line		400		400	pF

1)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

2)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

Cb = total capacitance of one bus line in pF

Table 53. Characteristics of Analog Input Section

(V_{DD} = 5V, TA = -10°C to 75°C)

Parameter	**Value**			**Unit**
	Min.	**Typ.**	**Max.**	
Voltage comparator reference voltage:				
Unit#1 - Video black level clamp	1.90	2.00	2.10	V
Unit#2 - Data slicer (**)	2.25	2.35	2.45	V
Unit#3 - Sync slicer	1.70	1.80	1.90	V
Voltage comparator delay (all units) (*)	150	200	250	ns
Video clamp:				
Sink current (CCVIDEO pin at 2.1V DC)	21	42	80	µA
Source current (CCVIDEO pin at 1.9V DC)	150	300	600	µA
Sink to source current ratio	0.1	0.14	0.18	

Measurement conditions: (*) Same DC level on both comparator inputs AC level 40mV for applicated measurement signal (**) corresponds to 25 IRE tap voltage

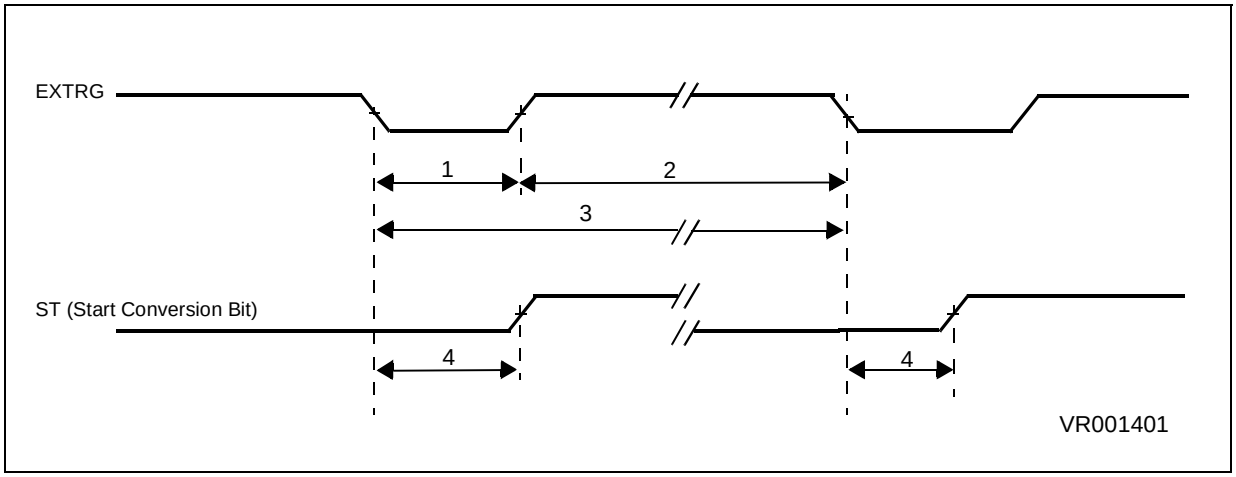
AC ELECTRICAL CHARACTERISTICS (Cont'd)

A/D CONVERTER, EXTERNAL TRIGGER TIMING TABLE

(V_{DD}= 5V +/-10%; T_A= -10°C to 75°C, unless otherwise specified)

N°	Symbol	Parameter	Conditions	Value		Unit
				min	max	
1	T_{low}	Pulse Width		1.5		INTCLK
2	T_{high}	Pulse Distance		1.5		INTCLK
3	T_{ext}	Period/fast Mode		78+1		INTCLK
4	T_{str}	Start Conversion Delay		0.5	1.5	INTCLK

A/D CONVERTER, EXTERNAL TRIGGER TIMING TABLE



A/D CONVERTER. ANALOG PARAMETERS TABLE

(V_{DD}= 5V +/-10% ; T_A= -10°C to 75°C, unless otherwise specified))

Parameter	Value			Unit	Note
	typ (*)	min	max	(**)	
Analog Input Range		V_{SS}	V_{DD}	V	
Conversion Time		138		INTCLK	(1,2)
Sample Time		87.51		INTCLK	(1)
Power-up Time		60		µs	
Resolution	8			bits	
Differential Non Linearity	0.5	0.3	1.5	LSBs	(4)
Integral Non Linearity			2	LSBs	(4)
Absolute Accuracy			2	LSBs	(4)
Input Resistance			1.5	Kohm	(3)
Hold Capacitance			1.92	pF	

Notes:
(*) The values are expected at 25 Celsius degrees with V_{DD}= 5V
(**)'LSBs' , as used here, as a value of V_{DD}/256
(1) @ 24 MHz external clock
(2) including Sample time
(3) it must be considered as the on-chip series resistance before the sampling capacitor
(4) DNL ERROR= max {[V(i) -V(i-1)] / LSB-1}INL ERROR= max {[V(i) -V(0)] / LSB-i}
ABSOLUTE ACCURACY= overall max conversion error

AC ELECTRICAL CHARACTERISTICS (Cont'd)

LATCH-UP AND ESD

Parameter	Conditions	Value	Unit
ESD Sensitivity	for ± 10µA	4	kV
	for ± 1µA	2	
Latch-up performance	STMicroelectronics specification for Class A	No Latch-Up	

PPM REQUIREMENTS

Parameter	Conditions	Value	Unit
PPM Requirements		100	ppm

3 EPROM/OTP PROGRAMMING

The EPROM/OTP of the ST92E196A/B & ST92T196A/B devices may be programmed using the EPROM programming boards available from STMicroelectronics.

EPROM Erasing

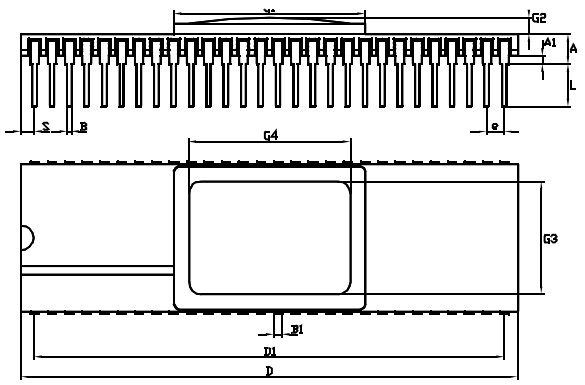
The EPROM of the windowed package of the ST92E196A/B can be erased by exposure to Ultra-Violet light.

The erasure characteristic of the ST92E196A/B is such that erasure begins when the memory is exposed to light with wave lengths shorter than approximately 4000Å. It should be noted that sunlight and some types of fluorescent lamps have wavelengths in the range 3000-4000 Å. It is recommended to cover the window of the ST92E196A/B packages by an opaque label to prevent unintentional erasure problems when testing the application in such an environment.

The recommended erasure procedure of the EPROM is the exposure to short wave ultraviolet light which have a wave-length 2537Å. The integrated dose (i.e. U.V. intensity x exposure time) for erasure should be a minimum of 15W-sec/cm2. The erasure time with this dosage is approximately 30 minutes using an ultraviolet lamp with a 12000 mW/cm2 power rating. The device should be placed within 2.5 cm (1 inch) of the lamp tubes during erasure.

4 PACKAGE DESCRIPTION

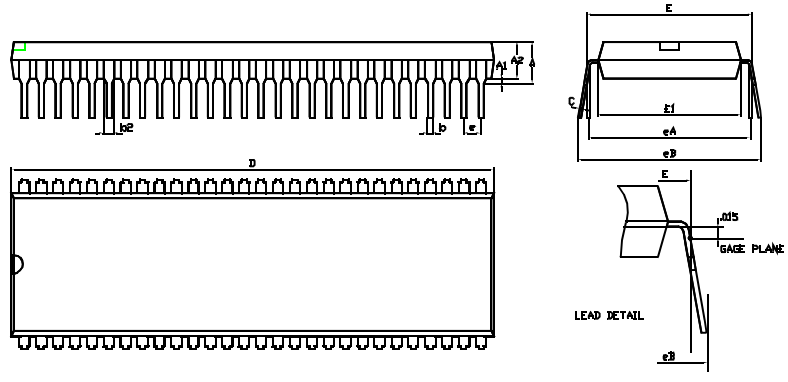
Figure 105. 56-Pin Shrink Ceramic Dual In-Line Package, 600-mil Width



CDIP56SW

Dim.	mm			inches		
	Min	Typ	Max	Min	Typ	Max
A			4.17			0.164
A1	0.76			0.030		
B	0.38	0.46	0.56	0.015	0.018	0.022
B1	0.76	0.89	1.02	0.030	0.035	0.040
C	0.23	0.25	0.38	0.009	0.010	0.015
D	50.04	50.80	51.56	1.970	2.000	2.030
D1		48.01			1.890	
E1	14.48	14.99	15.49	0.570	0.590	0.610
e		1.78			0.070	
G	14.12	14.38	14.63	0.556	0.566	0.576
G1	18.69	18.95	19.20	0.736	0.746	0.756
G2		1.14			0.045	
G3	11.05	11.30	11.56	0.435	0.445	0.455
G4	15.11	15.37	15.62	0.595	0.605	0.615
L	2.92		5.08	0.115		0.200
S		1.40			0.055	
	Number of Pins					
N	56					

Figure 106. 56-Pin Shrink Plastic Dual In Line Package, 600-mil Width



PSDIP56

Dim.	mm			inches		
	Min	Typ	Max	Min	Typ	Max
A			6.35			0.250
A1	0.38			0.015		
A2	3.18		4.95	0.125		0.195
b		0.41			0.016	
b2		0.89			0.035	
C	0.20		0.38	0.008		0.015
D	50.29		53.21	1.980		2.095
E		15.01			0.591	
E1	12.32		14.73	0.485		0.580
e		1.78			0.070	
eA		15.24			0.600	
eB			17.78			0.700
L	2.92		5.08	0.115		0.200
	Number of Pins					
N	56					

5 ORDERING INFORMATION

Device	Memory (Kbytes)	RAM (Kbytes)	Data Slicers	SCI	MFT	Package
ST92E196A9	128 (EPROM)	4	2	1	1	CSDIP56W
ST92T196A9	128 (OTP)					PSDIP56
ST92E196B7	128 (EPROM)	3	2	1	1	CSDIP56W
ST92T196B7	128 (OTP)					PSDIP56

ST9E2196A/B & ST92T196A/B OPTION LIST

Customer: . . . . . .
Address: . . . . . .
. . . . . .
Contact: . . . . . .
Phone No: . . . . . .
Reference/ROM Code* : . . . . . .
*The ROM code name is assigned by STMicroelectronics.

Please confirm characteristics of device:

Device:	[ ] ST92E196A9 128K EPROM	[ ] ST92T196A9 128K OTP
	[ ] ST92E196B7 128K EPROM	[ ] ST92T196B7 128K OTP
Package:	[ ] CSDIP56W	[ ] PSDIP56
Temperature Range:	-10°C to 75 °C	
OSD Code (A Version Only):	[ ] OSD filename _ _ _ _ _ _ _ _ _.OSD	
Special Marking:	[ ] No	[ ] Yes "_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _"

For marking, one line is possible with maximum 14 characters.
Authorized characters are letters, digits, '.', '-', '/' and spaces only.

Quantity forecast: [_ _ _ _ _ _ _] K units per year for a period of [_ _] years.
Preferred production start date: [_ _ _ _ _ _ _] (YYYY/MM/DD)

Date . . . . . .

Customer Signature . . . . . .

12 SUMMARY OF CHANGES

3.0	Addition of ST92T196B and ST92E196B salestypes. I²C Bus Interface speed changed from up to 800 kHz to 666.67 kHz. QFP64 package removed.	31 Aug 2001
3.1	VDDA recommendation for good slicing results added.	27 Feb 2003