Features

- **ARM7TDMI® ARM® Thumb® Processor Core**
  - **In-Circuit Emulator, 36 MHz operation**
- **Ethernet Bridge**
  - **Dual Ethernet 10/100 Mbps MAC Interface**
  - **16-Kbyte Frame Buffer**
- **1 K-Byte Boot ROM, Embedding a Boot Program**
  - **Enable Application Download from DataFlash®**
- **External Bus Interface**
  - **On-chip 32-bit SDRAM Controller**
  - **4-Chip Select Static Memory Controller**
- **Multi-level Priority, Individually Maskable, Vectored Interrupt Controller**
- **Three 16-bit Timer/Counters**
- **Two UARTs with Modem Control Lines**
- **Serial Peripheral Interface (SPI)**
- **Two PIO Controllers, Managing up to 48 General-purpose I/O Pins**
- **Available in a 256-ball BGA Package**
- **Power Supplies**
  - **VDDIO 3.3V nominal**
  - **VDDCORE and VDDOSC 1.8V nominal**
- **-40°C to + 85°C Operating Temperature Range**

Description

The AT91C140 is a member of the Atmel AT91 16- and 32-bit microcontroller family based on the ARM7TDMI processor core. This processor has a high performance 32-bit RISC architecture with a high density 16-bit instruction set and very low power consumption.

In addition, the AT91C140 integrates a double Ethernet 10/100 base-T MAC capable of operating as an Ethernet bridge, thus making it ideally suited for networking applications. It supports a wide range of memory devices such as SDRAM, SRAM and Flash and embeds an extensive array of peripherals.

The device is manufactured using Atmel's high-density CMOS technology. By combining the ARM7TDMI processor core with an expansive assortment of peripheral functions and low-power oscillators and PLL on a monolithic chip, the Atmel AT91C140 is a powerful microcontroller that provides a highly flexible and cost effective solution to many networking applications.

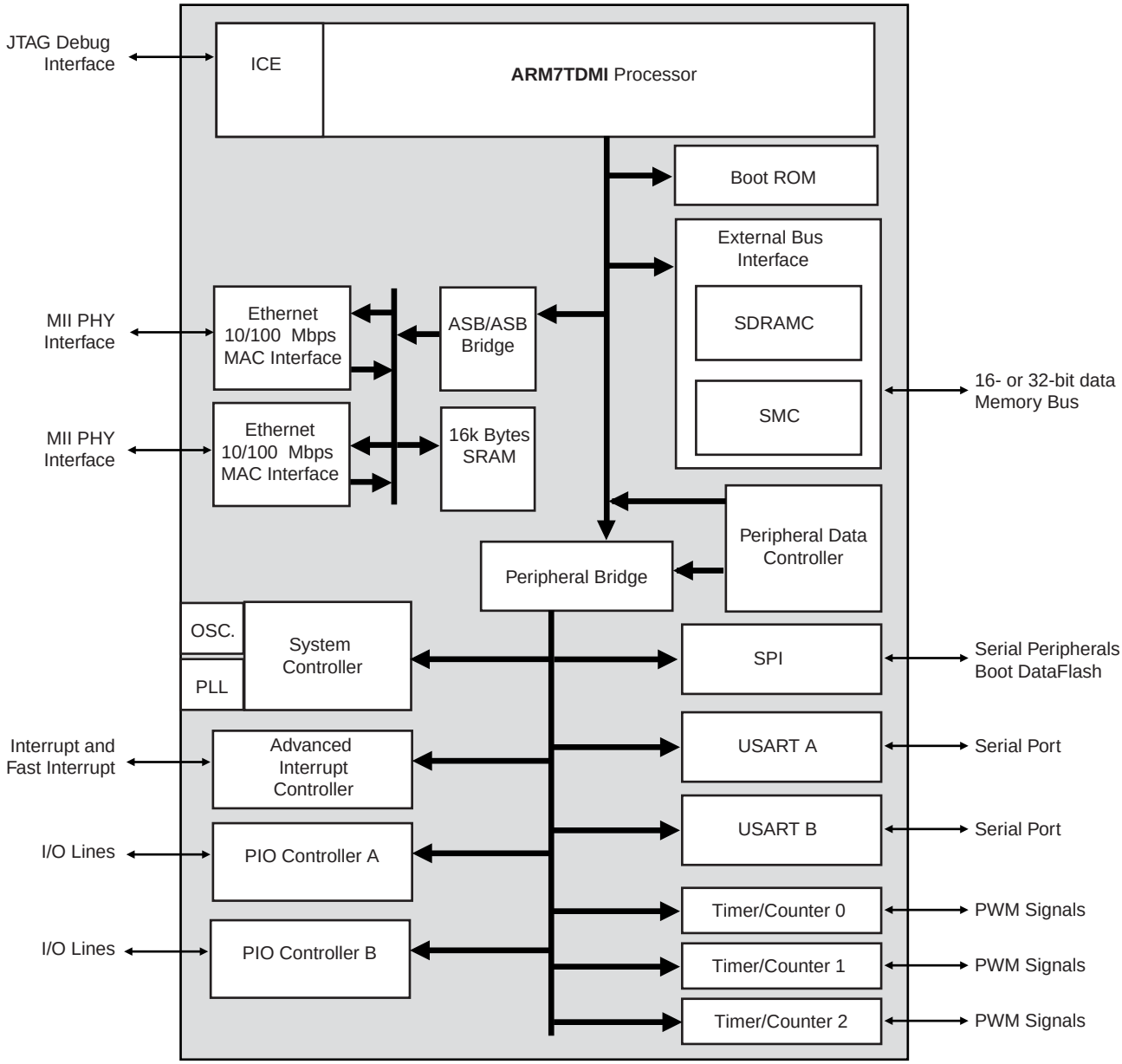
AT91® ARM® Thumb Microcontrollers

AT91C140

6069A–ATARM–05/04

Block Diagram

Figure 1. AT91C140 Block Diagram



Pinout

256-ball BGA Package Pinout

Table 1. Pinout for 256-ball BGA Package

Pin	Signal Name
A1	GND
A2	PB9
A3	PB4
A4	PB1
A5	NDSRB
A6	NRSTB
A7	RXDB
A8	NDSRA
A9	TXDA
A10	PA2
A11	PA3
A12	PA6
A13	PA10
A14	PA13
A15	PA15
A16	PA19
A17	NC(1)
A18	PA23
A19	TDO
A20	NC(1)
B1	VDDIO
B2	PB8
B3	PB7
B4	PB3
B5	PB0
B6	NDTRB
B7	TXDB
B8	NDCDA
B9	NRSTA
B10	PA1
B11	PA5
B12	PA7
B13	PA11
B14	VDDCORE
B15	PA16

Pin	Signal Name
B18	TDI
B19	NC(1)
B20	NC(1)
C1	PB10
C2	PA28
C3	DBW32
C4	PB6
C5	PB2
C6	NRIB
C7	NCTSB
C8	NRIA
C9	NCTSA
C10	PA0
C11	PA4
C12	PA8
C13	PA12
C14	PA14
C15	PA18
C16	PA21
C17	TCK
C18	NC(1)
C19	NC(1)
C20	PA31
D1	PB11
D2	PA27
D3	PA26
D4	GND
D5	PB5
D6	VDDIO
D7	NDCDB
D8	GND
D9	NDTRA
D10	RXDA
D11	VDDIO
D12	PA9

Pin	Signal Name
D15	VDDIO
D16	PA24
D17	GND
D18	PA29
D19	VDDCORE
D20	IRQ1
E1	NC(1)
E2	GND
E3	GND
E4	PA25
E17	PA30
E18	TST
E19	IRQ0
E20	NC(1)
F1	PB13
F2	PB12
F3	GND
F4	VDDIO
F17	VDDIO
F18	FIQ
F19	NC(1)
F20	SPCK
G1	MA_COL
G2	PB15
G3	PB14
G4	NTRST
G17	NRST
G18	PA22
G19	MOSI
G20	MISO
H1	MA_TXD0
H2	MA_TXER
H3	MA_CRS
H4	GND
H17	GND

Pin	Signal Name
H20	NSOE
J1	MA_TXEN
J2	MA_TXD3
J3	MA_TXD2
J4	MA_TXD1
J17	NWR
J18	NWE3
J19	NC(1)
J20	NWE2
K1	MA_RXD0
K2	MA_TXCLK
K3	NC(1)
K4	VDDIO
K17	NWE1
K18	NWE0
K19	NCE3
K20	NCE2
L1	MA_RXD1
L2	MA_RXD2
L3	MA_RXD3
L4	MA_RXER
L17	VDDIO
L18	NCE0
L19	NC(1)
L20	NCE1
M1	MA_RXCLK
M2	VDDCORE
M3	MA_RXDV
M4	MA_MDC
M17	PLLRC
M18	NC(1)
M19	XTALOUT
M20	XTALIN
N1	MA_MDIO
N2	MA_LINK

Table 1. Pinout for 256-ball BGA Package (Continued)

Pin	Signal Name	Pin	Signal Name	Pin	Signal Name	Pin	Signal Name
B16	PA20	D13	GND	H18	VDDIO	N3	MB_COL
B17	TMS	D14	PA17	H19	VDDCORE	N4	GND
N17	GND	T20	SDCS	V7	A11	W14	D12
N18	DQM3	U1	MB_RXD0	V8	A14	W15	VDDCORE
N19	VDDCORE	U2	MB_RXD2	V9	A18	W16	D17
N20	VDDOSC	U3	MB_RXCLK	V10	A22	W17	D20
P1	MB_CRS	U4	GND	V11	D2	W18	D24
P2	VDDCORE	U5	A1	V12	D6	W19	VDDIO
P3	MB_TXD0	U6	VDDIO	V13	D10	W20	NC(1)
P4	MB_TXD3	U7	A8	V14	D14	Y1	NC(1)
P17	RAS	U8	GND	V15	NC(1)	Y2	MB_MDIO
P18	DQM0	U9	A17	V16	D19	Y3	A2
P19	DQM1	U10	VDDIO	V17	D23	Y4	A3
P20	DQM2	U11	D3	V18	D26	Y5	A6
R1	MB_TXER	U12	D7	V19	NC(1)	Y6	A10
R2	MB_TXD1	U13	GND	V20	D29	Y7	A13
R3	MB_TXEN	U14	D16	W1	MB_MDC	Y8	A16
R4	VDDIO	U15	VDDIO	W2	NC(1)	Y9	A20/BA1
R17	VDDIO	U16	D22	W3	NC(1)	Y10	A23
R18	SDA10	U17	GND	W4	MB_LINK	Y11	D0
R19	CAS	U18	D27	W5	A5	Y12	D4
R20	WE	U19	NC(1)	W6	A9	Y13	D8
T1	MB_TXD2	U20	D30	W7	A12	Y14	D11
T2	MB_TXCLK	V1	MB_RXD3	W8	A15	Y15	D13
T3	MB_RXD1	V2	MB_RXDV	W9	A19/BA0	Y16	D15
T4	MB_RXER	V3	NC(1)	W10	A21	Y17	D18
T17	D28	V4	A0	W11	D1	Y18	D21
T18	D31	V5	A4	W12	D5	Y19	D25
T19	SDCK	V6	A7	W13	D9	Y20	NC(1)

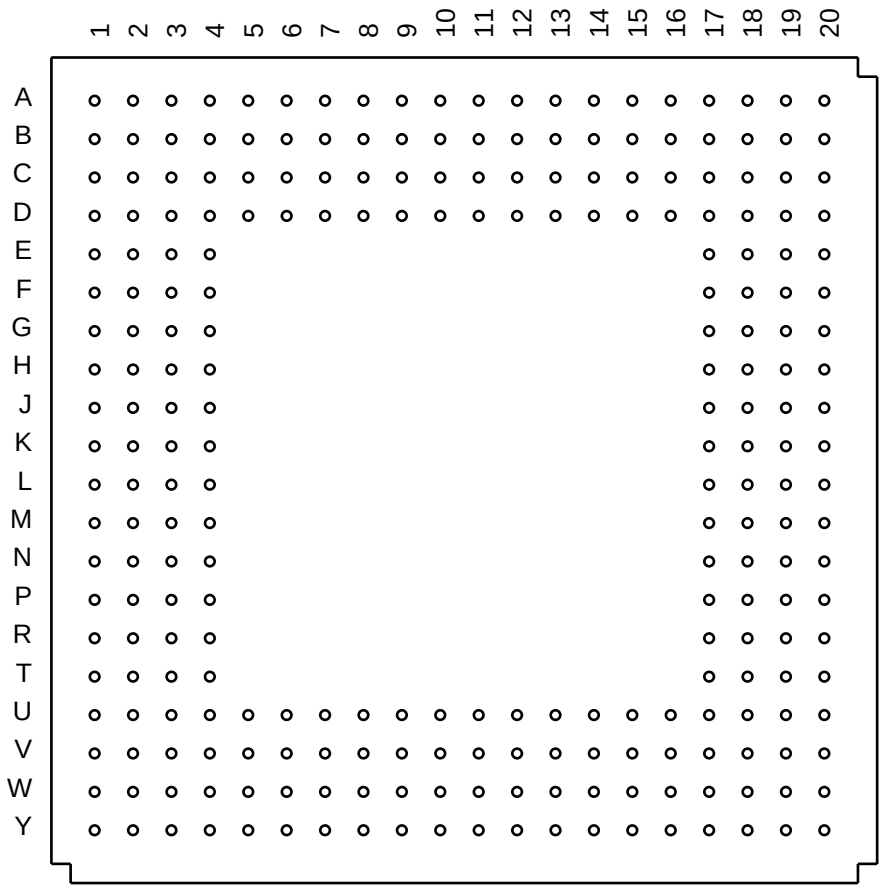
Note: 1. NC Balls should be left unconnected.

Mechanical Overview of the 256-ball BGA Package

Figure 2 below shows the orientation of the 256-ball BGA Package.

For a detailed mechanical description, see “Mechanical Characteristics and Packaging” on page 162.

Figure 2. 256-ball BGA Package Orientation (Top View)



Peripheral Multiplexing on PIO Lines

The AT91C140 features two PIO Controllers, PIOA and PIOB, multiplexing I/O lines of the peripheral set.

The PIO Controller A manages 32 I/O lines, PA0 to PA31.

The PIO Controller B manages only 16 I/O lines, PB0 to PB15.

Each I/O line of a PIO Controller can be multiplexed with a peripheral I/O. Multiplexing of the PIO Controller A is given in Table 2 on page 7. Multiplexing of the PIO Controller B is given in Table 3 on page 8.

PIO Controller A Multiplexing

Table 2. Multiplexing on PIO Controller A

I/O Line	Peripheral		
Name	**Signal Name**	**Description**	**Type**
PA0			
PA1			
PA2			
PA3			
PA4			
PA5			
PA6			
PA7			
PA8	TCLK0	Timer Counter Clock Input 0	Input
PA9	TIOA0	Timer Counter I/O LIne A 0	I/O
PA10	TIOB0	Timer Counter I/O LIne B 0	I/O
PA11	SCKA	UART A Serial Clock	I/O
PA12	NPCS1	Serial Peripehral Chip Select 1	Output
PA13			
PA14	NPCS2	Serial Peripheral Chip Select 2	Output
PA15	NPCS3	Serial Peripheral Chip Select 3	Output
PA16	TCLK2	Timer Counter Clock 2	Input
PA17	TIOA2	Timer Counter I/O Line A 2	I/O
PA18	TIOB2	Timer Counter I/O Line B 2	I/O
PA19	ACLKO	ARM System Clock	Output
PA20			
PA21			
PA22	NPCS0	Serial Peripheral Chip Select 0	I/O
PA23			
PA24			
PA25			
PA26			
PA27			
PA28			
PA29			
PA30			
PA31			

PIO Controller B Multiplexing

Table 3. Multiplexing on PIO Controller B

I/O Line	Peripheral		
Name	**Signal Name**	**Description**	**Type**
PB0	TCLK1	Timer Counter Clock Input 1	Input
PB1	TIOA1	Timer Counter I/O LIne A 1	I/O
PB2	TIOB1	Timer Counter I/O LIne B 1	I/O
PB3			
PB4			
PB5	NRIA	UART A Ring Indicator	Input
PB6			
PB7			
PB8			
PB9			
PB10			
PB11			
PB12			
PB13			
PB14			
PB15			

Signal Description

Table 4. Signal Description

Block	Signal Name	Function	Type
Power Supplies	VDDIO	I/O Lines Power Supply	
	VDDCORE	Device Core Power Supply	
	VDDOSC	PLL and Oscillator Power Supply	
	GND	Ground	
External Bus Interface	A0-A23	Address Bus	Output
	D0-D31	Data Bus	Input/Output
Synchronous Dynamic Memory Controller	SDCK	SDRAM Clock	Output
	DQM0-DQM3	SDRAM Byte Masks	Output
	SDCS	SDRAM Chip Select	Output
	SDA10	SDRAM Address Line 10	Output
	RAS	Row Address Strobes	Output
	CAS	Column Address Strobes	Output
	WE	Write Enable	Output
	BA0-BA1	Bank Address Line	Output
Static Memory Controller	NCE0-NCE3	Chip Selects	Output
	NWE0-NWE3	Byte Select/Write Enable	Output
	NSOE	Output Enable	Output
	NWR	Memory Block Write Enable	Output
PIO Controller A	PA0-PA31	PIO Controller A I/O Lines	Input/Output
PIO Controller B	PB0-PB15	PIO Controller B I/O Lines	Input/Output
Timer Counter	TCLK0-TCLK2	Timer Counter Clock 0 to 2	Input
	TIOA0-TIOA2	Timer Counter I/O Line A 0 to 2	Input/Output
	TIOB0-TIOA2	Timer Counter I/O Line B 0 to 2	Input/Output
Serial Peripheral Interface	MISO	Master In/Slave Out	Input/Output
	MOSI	Master Out/Slave In	Input/Output
	SPCK	Serial Clock	Input/Output
	NPCS0/NSS	Peripheral Chip Select 0/Slave Select	Input/Output
	NPCS1-NPCS3	Peripheral Chip Select 1 to 3	Output

Table 4. Signal Description (Continued)

Block	Signal Name	Function	Type
UART A and UART B	RXDA-RXDB	Receive Data	Input
	TXDA-TXDB	Transmit Data	Output
	NRTSA-NRSTB	Ready to Send	Output
	NCTSA-NCTSB	Clear to Send	Input
	NDTRA-NDTRB	Data Terminal Ready	Output
	NDSRA-NDSRB	Data Set Ready	Input
	NDCDA-NDCDB	Data Carrier Detect	Input
	NRIA-NRIB	Ring Indicator	Input
MAC A Interface	MA_COL	MAC A Collision Detect	Input
	MA_CRS	MAC A Carrier Sense	Input
	MA_TXER	MAC A Transmit Error	Output
	MA_TXD0-MA_TXD3	MAC A Transmit Data Bus	Output
	MA_TXEN	MAC A Transmit Enable	Output
	MA_TXCLK	MAC A Transmit Clock	Input
	MA_RXD0-MA_RXD3	MAC A Receive Data Bus	Input
	MA_RXER	MAC A Receive Error	Input
	MA_RXCLK	MAC A Receive Clock	Input
	MA_RXDV	MAC A Receive Data Valid	Output
	MA_MDC	MAC A Management Data Clock	Output
	MA_MDIO	MAC A Management Data Bus	Input/Output
	MA_LINK	MAC A Link Interrupt	Input
MAC B Interface	MB_COL	MAC B Collision Detect	Input
	MB_CRS	MAC B Carrier Sense	Input
	MB_TXER	MAC B Transmit Error	Output
	MB_TXD0-MB_TXD3	MAC B Transmit Data Bus	Output
	MB_TXEN	MAC B Transmit Enable	Output
	MB_TXCLK	MAC B Transmit Clock	Input
	MB_RXD0-MB_RXD3	MAC B Receive Data Bus	Input
	MB_RXER	MAC B Receive Error	Input
	MB_RXCLK	MAC B Receive Clock	Input
	MB_RXDV	MAC B Receive Data Valid	Output
	MB_MDC	MAC B Management Data Clock	Output
	MB_MDIO	MAC B Management Data Bus	Input/Output
	MB_LINK	MAC B Link Interrupt	Input

Table 4. Signal Description (Continued)

Block	Signal Name	Function	Type
In-Circuit Emulator	NTRST	Test Reset	Input
	TCK	Test Clock	Input
	TMS	Test Mode Select	Input
	TDI	Test Data Input	Input
	TDO	Test Data Output	Output
Miscellaneous	NRST	Reset	Input
	FIQ	Fast Interrupt	Input
	IRQ0-IRQ1	Interrupt Lines	Input
	PLLRC	PLL RC Filter	Analog
	XTALIN	Crystal Input	Analog
	XTALOUT	External Crystal	Analog
	TST	Test Mode	Input
	DBW32	External Data Bus Width for CS0 (1 = 32 bits)	Input
	ACLKO	ARM Clock Output	Output

ARM7TDMI Core

The ARM7TDMI is a three-stage pipeline, 32-bit RISC processor. The processor architecture is Von Neumann load/store architecture, characterized by a single data and address bus for instructions and data. The CPU has two instruction sets: the ARM and the Thumb instruction set. The ARM instruction set has 32-bit wide instructions and provides maximum performance. Thumb instructions are 16-bit wide and give maximum code density.

Instructions operate on 8-bit, 16-bit and 32-bit data types.

The CPU has seven operating modes. Each operating mode has dedicated banked registers for fast exception handling. The processor has a total of 37 32-bit registers, including six status registers.

Power Supplies

The AT91C140 has three types of power supply pins:

- VDDCORE pins power the core, including the ARM7TDMI processor, the memories and the peripherals; voltage is between 1.65V and 1.95V, 1.8V nominal.
- VDDIO pins power the I/O lines, including those of the External Bus Interface and those of the peripherals; voltage is between 3V and 3.6V, 3.3V nominal.
- VDDOSC pins power the PLL and oscillator cells; voltage is between 1.65V and 1.95V, 1.8V nominal.

Ground pins are common to all power supplies.

System Controller

The AT91C140 features a System Controller that takes care of and controls:

- The Test Mode
- The Reset
- The System Clocks
- The Chip Identifier

The System Controller manages the reset of all the system and integrates a clock generator, made up of an oscillator and a PLL.

Test

The AT91C140 features a test pin (TST). This pin must be tied low for normal operations. Using the AT91C140 with the TST pin at a high level might lead to unpredictable results.

Reset Controller

NRST Pin

The AT91C140 is reset by asserting the NRST pin low. It should be asserted for a time adequate to ensure the startup of the oscillator on a power on, and at least 1 ACLK cycle for a warm reset. As the ACLK switches on the 31,25kHz (assuming the crystal is at 16 MHz) as soon as the reset is asserted, it must remain low for at least 32 µs. The first instruction fetch happens 10 ACLK cycles after the reset releases.

System Reset

A reset initializes the user interface registers to their default states as defined in the peripheral sections of this datasheet and forces the ARM7TDMI to perform the next instruction fetch from address zero. Except for the program counter and the Current Program Status Register, the ARM processor registers do not have defined reset states. When NRST is active, the inputs of the AT91C140 must be held at valid logic levels to reduce the power consumption to a minimum.

Boot Memory and Remap Command

When NRST is released, the PA0 pin is sampled to determine if the ARM processor should boot from internal ROM or from external memory connected to NCE0. The details of the boot operations are described in “Memory Controller (MC)” on page 17. The Boot Program is described in “Boot Program” on page 24.

After a reset, the RM bit in the Mode Register reflects the state of the PA0 pin. Then, writing this bit at 1 removes the ROM from the address 0. Writing it at 0 remaps the ROM at address 0x0.

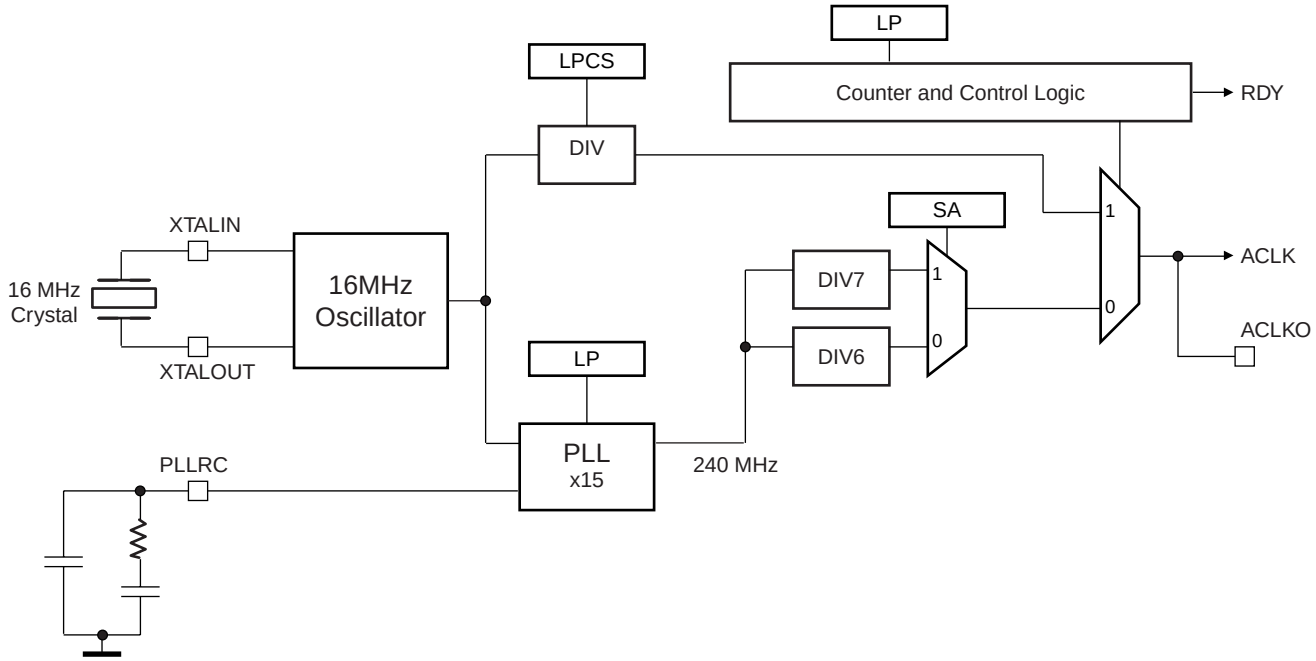
Clock Generator

The AT91C140 features a Clock Generator based on a 16 MHz oscillator and a PLL. It provides all the clocks of the system, including a clock signal named ACLK, to the ARM processor, to the memory controller and to the External Bus Interface and to all the embedded peripherals

The ACLK signal is also provided on the ACLKO pin, through PIO Controller A.

Figure 3 below shows the architecture of the Clock Generator.

Figure 3. Clock Generator



After the reset, the ACLK clock is running at 31.25 kHz. The user can program the LPCS field to speed the boot sequence.

The ACLKST (ARM Clock Status) bit reflects the clock being used for the ARM. When read at 0, ACLK is 40 MHz if SA is 0 and 34.3 MHz if SA is 1. When read at 1, ACLK is at a frequency according to the value programmed in the LPCS field in the System Mode Register (SYS_MR).

Chip ID

The System Controller features a Chip ID Register that reads a value of 0x00010221

System Controller User Interface

Base Address: 0xFF00 0000.

Table 5. System Controller Register Mapping

Offset	Register Name	Register Description	Access	Reset Value
0x0	SYS_MD	System Mode Register	Read/Write	0x0000 034x
0x4	SYS_ID	System ID Register	Read-only	0x0001 0221
0x8	Reserved			
0xC	SYS_CLKF	System Clock Status Register	Read-only	0x0000001

System Mode Register

Register Name: SYS_MD
Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	0	–	0	–	–	LPCS	
7	6	5	4	3	2	1	0
SA	LP	–	–	0	–	0	RM

- **RM: Remap**

0 =The ROM is mapped only at its normal address.

1 =The ROM is mapped at its address and at address 0x0.

- **LP: Low Power Mode**

0 =The PLL is enabled and ACLK is the output of the PLL divided by 6 or 7.

1 =The PLL is disabled and ACLK is defined by LPCS.

- **SA: Slow ARM**

0 =The ARM divider is 6.

1 =The ARM divider is 7.

- **LPCS: Low Power Clock Select**

LPCS		Divisor	ACLK
0	0	2	8 MHz
0	1	16	1 MHz
1	0	64	250 kHz
1	1	512	31,25 kHz

System ID Register

Register Name: SYS_ID
Access: Read-only

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0

23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	1

15	14	13	12	11	10	9	8
0	0	0	0	0	0	1	0

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

System Clock Status Register

Register Name: SYS_CLKF
Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	ACLKST

- **ACLKST: ARM Clock Status**

0 = ARM Clock currently using the 240 MHz source (PLL).

1 = ARM Clock currently using the 16 MHz source (oscillator).

Memory Controller (MC)

Architecture

The AT91C140 architecture is made up of two Advanced System Buses, the ARM ASB and the MAC ASB. Both handle a single memory space.

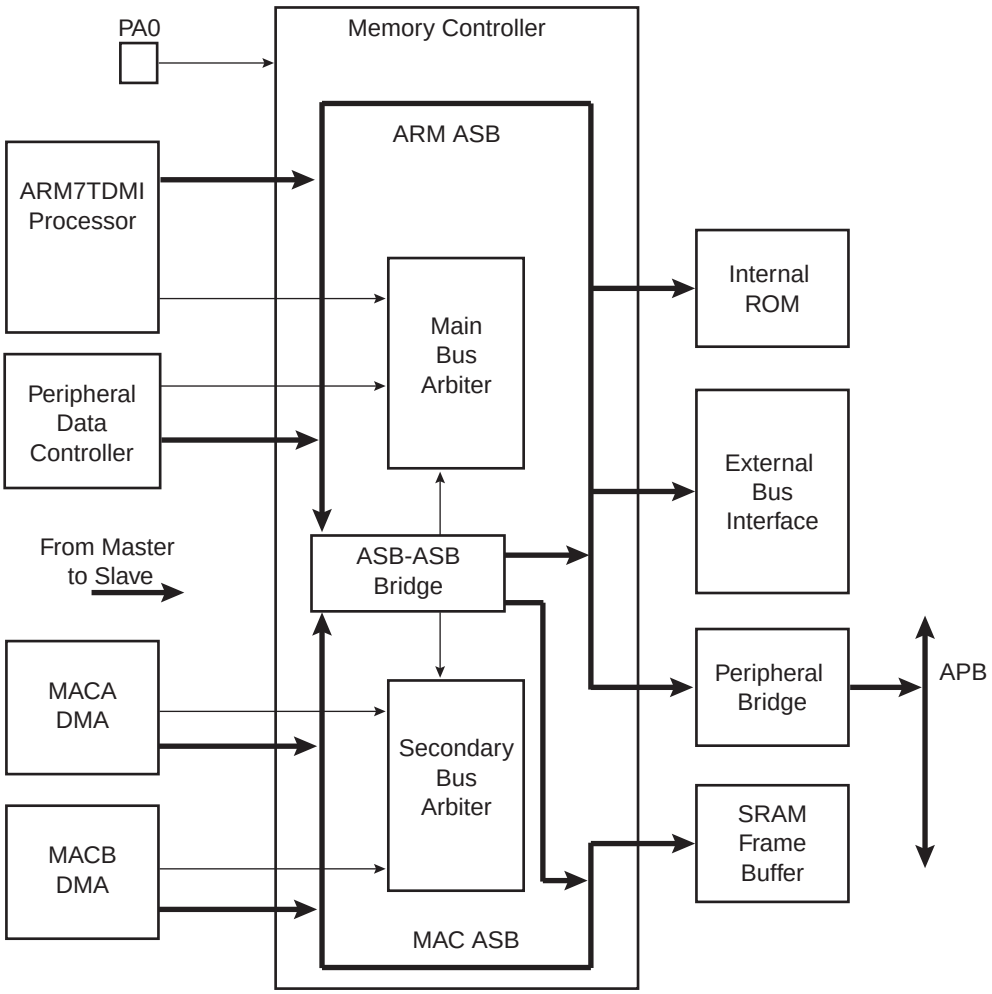
The ARM ASB handles the access requests of the ARM7TDMI and the PDC. It also handles the access requests coming from the MAC ASB. It connects with the External Bus Interface, the Peripheral Bridge and the Internal Memories. It also connects with the MAC ASB.

The MAC ASB handles the access requests of the DMAs of both Ethernet MACs. It also handles the access requests coming from the the ARM ASB. It connects essentially with the Frame Buffer, but also connects with the ARM ASB.

The major advantage of this double-ASB architecture is that the Ethernet traffic does not occupy the main ASB bandwidth, ensuring that the ARM7TDMI can perform at its maximum speed while the Ethernet traffic goes through the Frame Buffer.

The AT91C140 architecture is shown in Figure 4.

Figure 4. AT91C140 Memory Controller Architecture



Memory Map

The AT91C140 memory map is divided into regions of 256 megabytes. The top memory region (0xF000_0000) is reserved and subdivided for the internal memories and shared memory and the embedded peripherals.

The device can define up to five other active external memory regions by means of the static memory controller and SDRAM memory controller.

The memory map is divided between both ASBs, as shown in Figure 4. All regions except the 16 megabytes between 0xFC00 0000 and 0xFCFF FFFF are located on the Main ASB. Accesses to locations between 0xFC00 0000 and 0xFCFF FFFF are routed to the MAC ASB.

The memory map assumes default values on reset. External memory regions can be reprogrammed to other base addresses in the Static Memory Controller or in the SDRAM Controller. Note that the internal memory regions have fixed locations that cannot be reprogrammed.

There are no hardware locks to prevent incorrect programming of the regions. Programming two or more regions to have the same base address or overlapping two memory regions results in undefined behavior.

The ARM processor reset vector at address 0x00000000 is mapped into the internal ROM or external memory connected on NCE0. This selection depends on the PA0 signal pin. After booting, the ROM region can be disabled and any external memory can be mapped to the bottom of the memory map by programming SMC_CSRx or SDRAMC_ADDR.

Figure 5. AT91C140 Memory Map

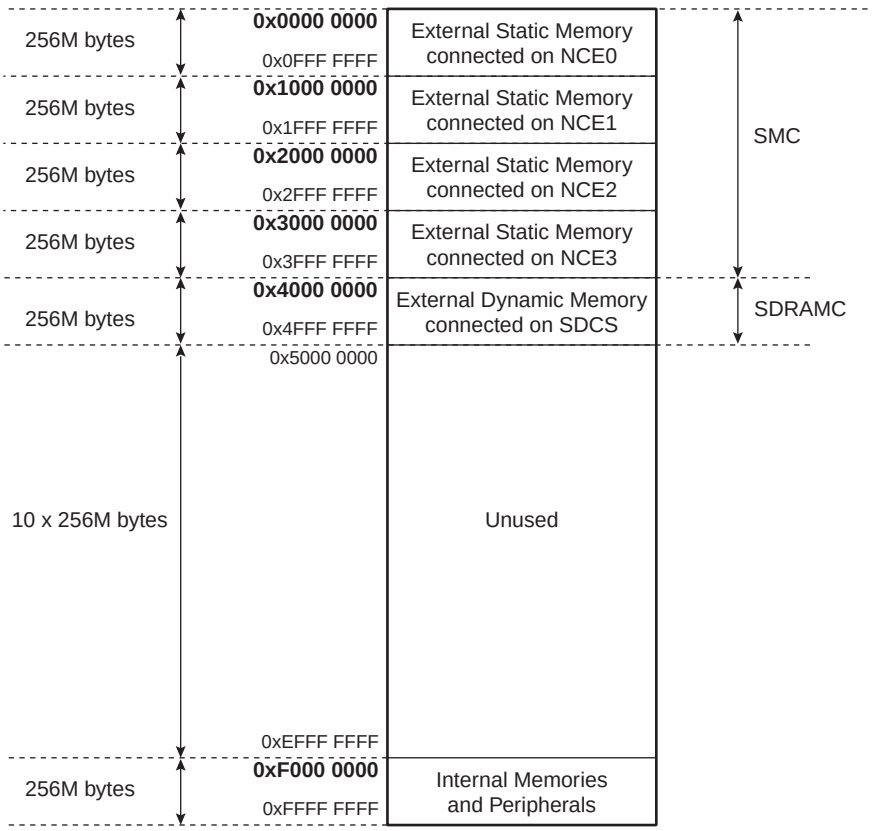
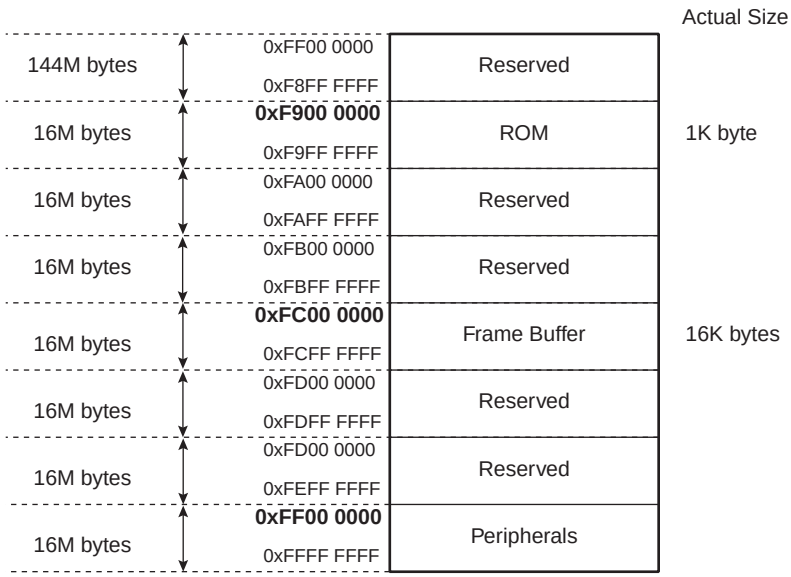


Figure 6 below shows the mapping of the internal memories and the address space reserved for the Peripheral Bridge.

Figure 6. Internal Memory Mapping



ARM ASB Arbitration

The ARM ASB is arbitrated with the following priorities:

- The PDC has the highest priority.
- The Bridge from the MAC ASB has the middle priority.
- The ARM processor has the lowest priority.

MAC ASB Arbitration

The MAC ASB is arbitrated with the following priorities:

- The Bridge from the ARM ASB has the highest priority.
- The MAC A has the middle priority.
- The MAC B has the lowest priority.

ASB-ASB Bridge Arbitration

The MAC ASB has two priority levels; the two MACs share low priority access and the ASB-ASB Bridge has high priority. The MACs do not burst more than four words per access and release the bus request between accesses so the MACs share a priority level with a simple round-robin arbitration scheme.

The ARM is likely to be the only master accessing the MAC bus via the bridge and should not perform more than a couple of cycles before releasing the MAC bus. Care should be taken to prevent other masters on the ARM bus holding the MAC bus for more than a few cycles. Otherwise, the MACs drop frames due to FIFO overflow or underflow.

When a master on one bus accesses a slave on the other bus, the following operations occur:

- The local bus arbiter arbitrates the master's request for the local ASB bus if it does not already have access to the bus.
- When the local bus arbiter grants the local bus to the master, the master initiates a cycle with an address corresponding to a slave on the remote bus.
- The bridge is selected as the slave on the local bus and responds by inserting wait cycles. The bridge also requests the remote bus from the remote bus arbiter.

- When the bridge is granted the remote bus, the two ASB buses are coupled and the transfer completes.

The ASB performs pipelined arbitration. The ASB-ASB Bridge can only request the bus when the address of the slave is available. For this reason, the ASB-ASB Bridge inserts a wait cycle during the arbitration cycle on the remote bus because it cannot request the bus early.

Boot Mode

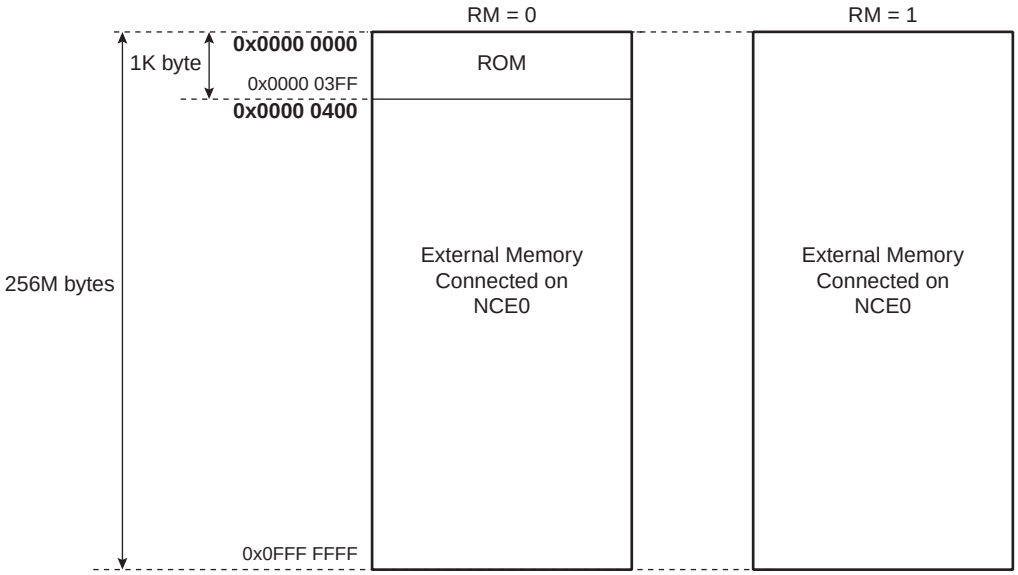
The AT91C140 has an integrated 1-Kbyte ROM to support the boot software. When the device is released from reset, the pin PA0 is sampled by the Memory Controller. If sampled low, the internal ROM becomes accessible from the address 0x0, so that the ARM processor starts execution of the Boot Program. Note that the ROM remains accessible at its normal address. If the pin PA0 is sampled high at reset, the mapping does not change and the external memory connected on NCE0 should contain a valid boot sequence.

The level of the pin PA0 at resets is indicated by the RM flag in the System Mode Register (SYS_MD). Then, the RM bit can be written at any value to map to or remove the ROM from address 0x0.

If PA0 is asserted on reset, the Boot Program in ROM is executed. The Boot Program is described in “Boot Program” on page 24.

Figure 7 below shows the mapping of the ROM depending on the Boot Mode.

Figure 7. ROM Mapping Depending on the Boot Mode



Endianness

The AT91C140 Memory Controller operates in little-endian mode only. The user has to make sure that the data structures used by the ARM7TDMI, the Ethernet DMAs and the PDC are compliant with this mode of byte arrangement.

Peripherals

The Peripheral Bridge allows access to the embedded peripheral user interfaces. It is optimized for low power consumption, as it is built without usage of any clock. However, any access on the peripheral is performed in two cycles.

The AT91C140 peripherals are designed to be programmed with a minimum number of instructions. Each peripheral has 16K bytes of address space allocated in the upper part of the address space.

Peripheral Registers

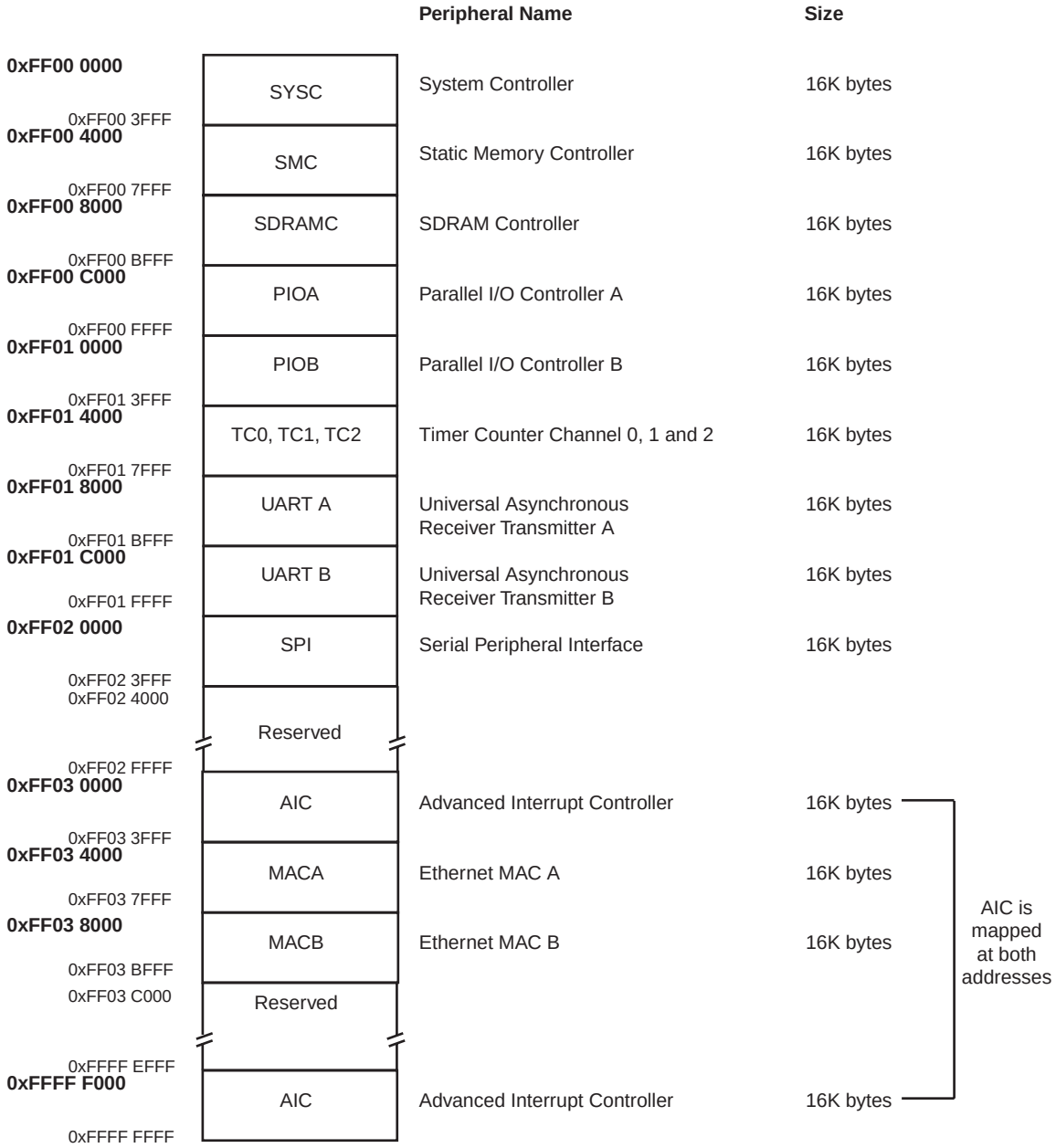
All of the peripheral registers are 32-bits wide and support only aligned accesses. When a misaligned access is performed within the peripheral address space, the access is automatically performed at the lower aligned address.

All undefined or unused register bits (marked “-”) read 0. It is recommended to write them at 0 for software upward compatibility.

Peripheral Memory Map

Figure 8 below gives the mapping of the peripherals integrated in the AT91C140.

Figure 8. Peripheral Memory Map



Peripheral Data Controller (PDC)

PDC Overview

The AT91C140 features a six-channel Peripheral Data Controller (PDC) dedicated to the two on-chip UARTs and the SPI. One PDC channel is connected to the receiving channel and one to the transmitting channel of each UART and of the SPI.

Each PDC channel operates as DMA (Direct Memory Access).

The User Interface of a PDC channel is integrated in the memory space of each peripheral. It contains a 32-bit address pointer register and a 16-bit count register. When the programmed number of bytes is transferred, an end-of-transfer signal is sent to the peripheral and is visible in the peripheral status register. This status bit might trigger an interrupt.

PDC Channel Priority

The transfer requests from the peripherals are treated in the order they happen.

When several transfer requests happen in the same cycle, the following priority order is applied:

- the UART A receiver
- the UART A transmitter
- the UART B receiver
- the UART B transmitter
- the SPI receiver
- the SPI transmitter

Boot Program

The AT91C140 can boot in several ways as explained below. When the ARM7TDMI processor is released from reset it basically attempts a fetch from address 0x00000000. Depending on an hardware configuration, the memory mapping can be altered and thus modify how the system boots.

Boot Mode

When the master reset is released, the pin PA0 is latched. Its state defines how the system boots. When PA0 is latched at 1, the AT91C140 is said to be configured in external boot mode. The initial state of the EBI maps the 1-Mbyte address range starting from 0x00000000 in the external device selected by NCE0. In this boot mode, NCE0 is assumed to be connected to an external memory device containing the suitable boot code.

When PA0 is latched at 0, the AT91C140 is said to be configured in internal boot mode. The internal boot ROM normally located at base address 0xf9000000 is aliased at address 0x00000000. In this boot mode, the ARM Processor executes the first instructions out of the internal boot ROM.

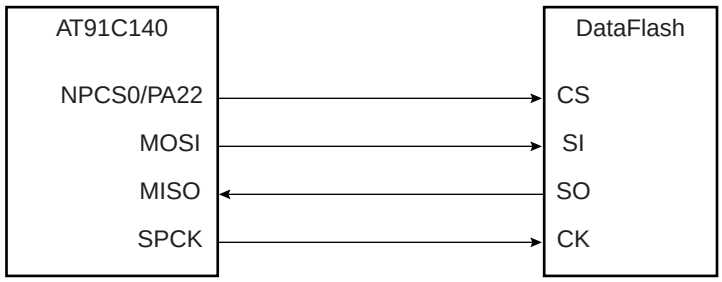
The boot mode is reflected by the RM bit in register SYS_MD. Reading RM at 0 indicates that the boot ROM is aliased at base address 0x00000000, eventually overlapping the memory layout defined by the SMC and the SDRAMC registers. Reading RM at 1 indicates that the boot ROM can be accessed only from base address 0xf9000000. Writing RM allows to select the mapping of the boot ROM under software control.

Hardware Connection of the DataFlash

The internal boot software provides the AT91C140 with the capability of booting from an external serial DataFlash connected on the on-chip SPI interface as described above.

When the internal boot software is used in conjunction with DataFlash, the latter must be connected to the AT91C140 as shown below in Figure 9.

Figure 9. DataFlash Connection



Internal Boot Software

The internal boot code goes through the following steps in sequence:

- The processor enters the supervisor mode and all the interrupts are masked.
- A branch is executed into the ROM alias based from 0xf9000000.
- The ROM alias based at 0x00000000 is removed by writing the RM bit at 1.
- The clock is programmed at the highest frequency achievable without using the on-chip PLL (i.e. the frequency of the crystal divided by 2).
- The on-chip SPI interface is setup to prepare for communications with DataFlash.
- A bunch of data is downloaded from the DataFlash. This data is expected to contain a formatted header describing the contents of the DataFlash.
- This header is analyzed to verify whether a DataFlash is actually present and contains valid executable code.

- If the DataFlash is there and contains valid executable code, this code is downloaded into a location specified by the header, and an absolute branch to this code is performed.
- If the DataFlash is missing, or if the header is not valid, an absolute branch to address 0x00000000 is performed. A suitable memory device should be mapped at this address and contain the expected code.

DataFlash Header Details

To ensure correct operation of the boot out of DataFlash, the DataFlash must contain a valid header. This header contains several fields which define how the application software residing further must be handled. The structure of the DataFlash header is illustrated below in Table 6.

Table 6. Header Structure

Field Address	Field Name	Field Length
0x00 (1)	MAGIC	4 bytes
0x04	DSRC	4 bytes
0x08	DDST	4 bytes
0x0c	DSIZE	4 bytes
0x10	ENTRY	4 bytes

Note: 1. The field address is respective to the DataFlash space. 0x00 corresponds to the first location of the DataFlash.

The MAGIC field contains a predefined magic number which allows identification of the suitability of the DataFlash. The value of this field must be 0x0075C221 to allow the boot routine to proceed. If another value is read, the boot code gives up the download and branches to 0x0000 0000 where the real application code is expected.

The DSRC field contains the address where the code to be downloaded resides in DataFlash. This address is respective to the DataFlash address space (**not** the ARM Processor address space) and follows the non-linear addressing scheme defined in the documentation of the DataFlash. Note that all bits are not necessarily significant, depending on the specific DataFlash device.

The DDST field contains the destination address where the downloaded code will be copied. This address is respective to the ARM Processor address space. Typically, this address should point into some internal RAM.

The DSIZE field contains the number of bytes to be downloaded. This value is exclusive of the header. It must be even.

The ENTRY field contains the address where the boot routine must branch when the download is complete. It is the entry point of the newly downloaded software. Although this is not required, the ENTRY field equals the DDST field in most cases.

Reserved Resources

The internal boot code needs some resources to operate correctly, especially as it programs some on-chip peripherals. These must not be assumed to be in their reset state when the control is given to the application code. The concerned peripherals are:

- the clock management system
- the SPI interface
- the PIO pin PA22
- the RM bit

The internal boot code also uses some internal RAM locations to store temporary data. These reside in the first 64 bytes of RAM, i.e. from 0xFD00 FFC0 to 0xFD00 FFFF. The DDST, DSIZE fields of the DataFlash header must not define a memory area overlapping the locations used by the internal boot routine. The ENTRY field must not point into this area.

External Bus Interface (EBI)

The External Bus Interface (EBI) generates the signals that control access to external memories or peripheral devices. It contains two controllers, the SDRAM Controller and the Static Memory Controller and manages the sharing of data and address busses between both of these controllers.

Signal Multiplexing

Table 7. Signal Description and Multiplexing

Name	Description	Controlled by SMC	Controlled by SDRAMC
[D31:0]	Data Bus	[D31:0]	[D31:0]
[A9:0]	Address Lines 0 to 9	[A9:0]	[A9:0]
A10	Address Line 10	A10	
SDA10	SDRAM Controller Address Line 10		A10
[A12:11]	Address Lines 11 to 12	[A12:11]	[A12:11]
[A18:13]	Address Lines 13 to 18	[A18:13]	
A19/BA0	Address Line 19 or Bank Address 0	A19	BA0
A20/BA1	Address Line 20 or Bank Address 1	A20	BA1
[A23:21]	Address Lines 21 to 23	[A23:21]	
SDCK	SDRAM Clock		SDCK
SDCS	SDRAM Controller Chip Select		SDCS
RAS	SDRAM Row Signal		RAS
CAS	SDRAM Column Signal		CAS
WE	SDRAM Write Enable		WE
DQM0-DQM3	SDRAM Data Mask Enable Signals		DQM0-DQM3
NCE0-NCE3	Active low chip enable	NCE0-NCE3	
NWE0-NWE3	Active low byte select/write strobe signals	NWE0-NWE3	
NWR	Active low write strobe signals	NWR	
NSOE	Active low read enable signal	NSOE	

SDRAM Controller (SDRAMC)

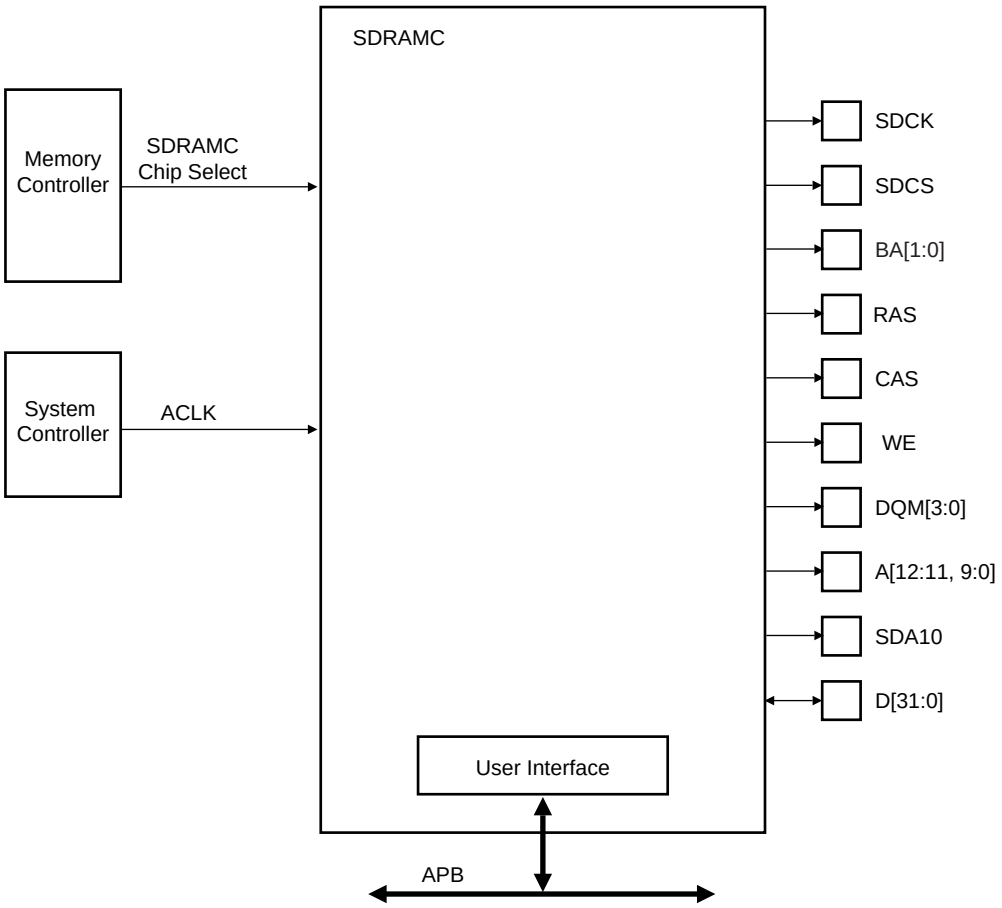
Description

The SDRAM Controller (SDRAMC) extends the memory capabilities of a chip by providing the interface to an external 16-bit or 32-bit SDRAM device. The page size supports ranges from 2048 to 8192 and the number of columns from 256 to 2048. It supports byte (8-bit), half-word (16-bit) and word (32-bit) accesses. The maximum addressable SDRAM size is 256M bytes.

The SDRAM Controller supports a read or write burst length of one location. It keeps track of the active row in each bank, thus maximizing SDRAM performance, e.g., the application may be placed in one bank and data in the other banks. So as to optimize performance, it is advisable to avoid accessing different rows in the same bank.

Block Diagram

Figure 10. SDRAM Controller Block Diagram



I/O Lines Description

Table 8. I/O Line Description

Name	Description	Type	Active Level
SDCK	SDRAM Clock	Output	
SDCS	SDRAM Controller Chip Select	Output	Low
BA[1:0]	Bank Select Signals	Output	High
RAS	Row Signal	Output	Low
CAS	Column Signal	Output	Low
WE	SDRAM Write Enable	Output	Low
DQM[3:0]	Data Mask Enable Signals	Output	Low
A [12:11] SDA10 A[9:0]	Address Bus	Output	
D[31:0]	Data Bus	I/O	

Application Example

Hardware Interface

Figure 11 below shows an example of an SDRAM device connection to the SDRAM Controller by using a 32-bit data bus width. Figure 12 shows an example of an SDRAM device connection by using a 16-bit data bus width.

Figure 11. SDRAM Controller Connections to SDRAM Devices: 32-bit Data Bus Width

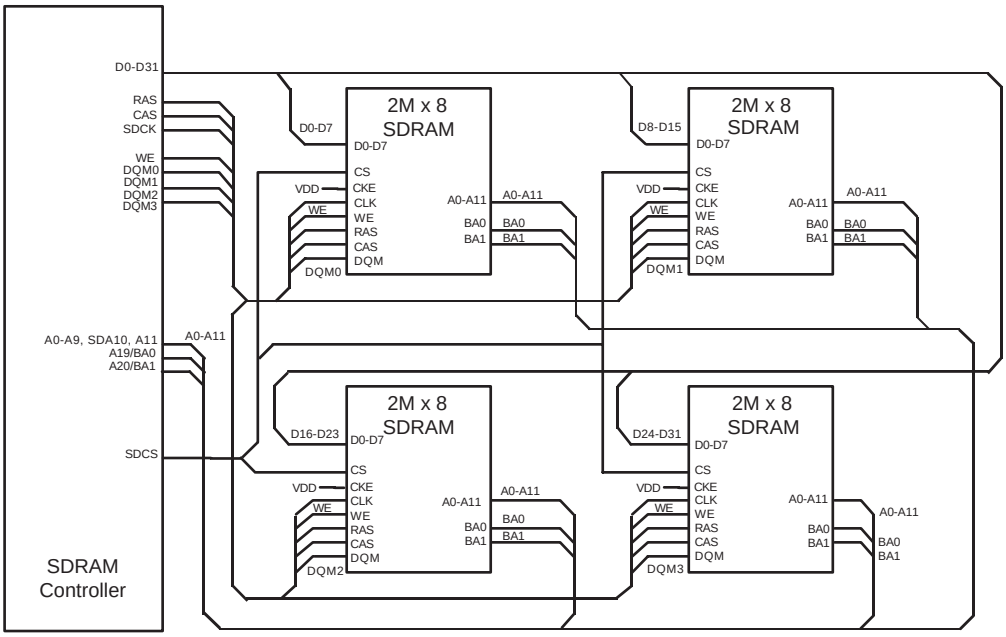
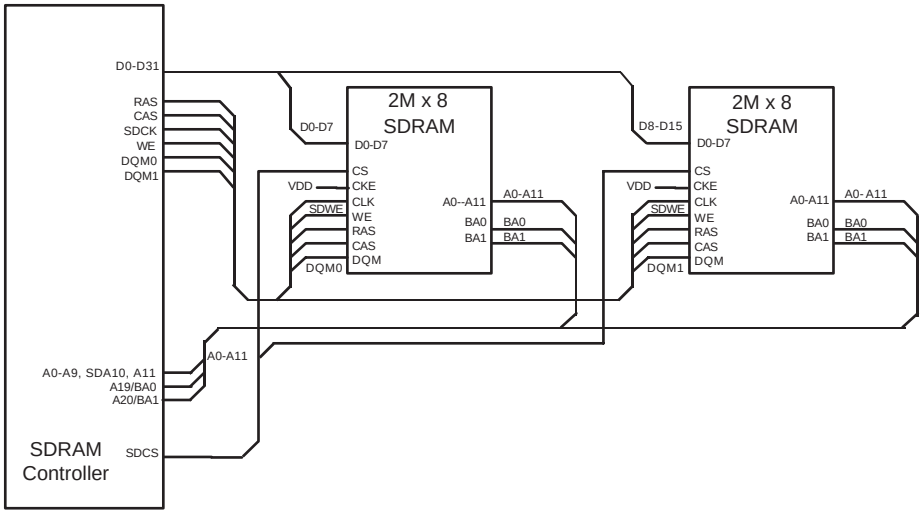


Figure 12. SDRAM Controller Connections to SDRAM Devices: 16-bit Data Bus Width



Software Interface

The SDRAM Controller's function is to make the SDRAM device access protocol transparent to the user. Table 9 to Table 13 illustrate the SDRAM device memory mapping therefore seen by the user in correlation with the device structure. Various configurations are illustrated.

32-bit Memory Data Bus Width

Table 9. SDRAM Configuration Mapping: 2K Rows, 256/512/1024/2048 Columns

CPU Address Line																											
27	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
					Bk[1:0]		Row[10:0]											Column[7:0]								M[1:0]	
				Bk[1:0]		Row[10:0]											Column[8:0]									M[1:0]	

Table 10. SDRAM Configuration Mapping: 4K Rows, 256/512/1024/2048 Columns

CPU Address Line																											
27	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
				Bk[1:0]		Row[11:0]												Column[7:0]								M[1:0]	

16-bit Memory Data Bus Width

Table 11. SDRAM Configuration Mapping: 2K Rows, 256/512/1024/2048 Columns

CPU Address Line																											
27	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
						Bk[1:0]		Row[10:0]											Column[7:0]								M0
					Bk[1:0]		Row[10:0]											Column[8:0]								M0		
				Bk[1:0]		Row[10:0]											Column[9:0]								M0		

Table 12. SDRAM Configuration Mapping: 4K Rows, 256/512/1024/2048 Columns

CPU Address Line																										
27	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
					Bk[1:0]	Row[11:0]													Column[7:0]								M0
				Bk[1:0]	Row[11:0]													Column[8:0]									M0

Table 13. SDRAM Configuration Mapping: 8K Rows, 256/512/1024/2048 Columns

CPU Address Line																										
27	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
				Bk[1:0]	Row[12:0]														Column[7:0]								M0

SDRAM Device Initialization

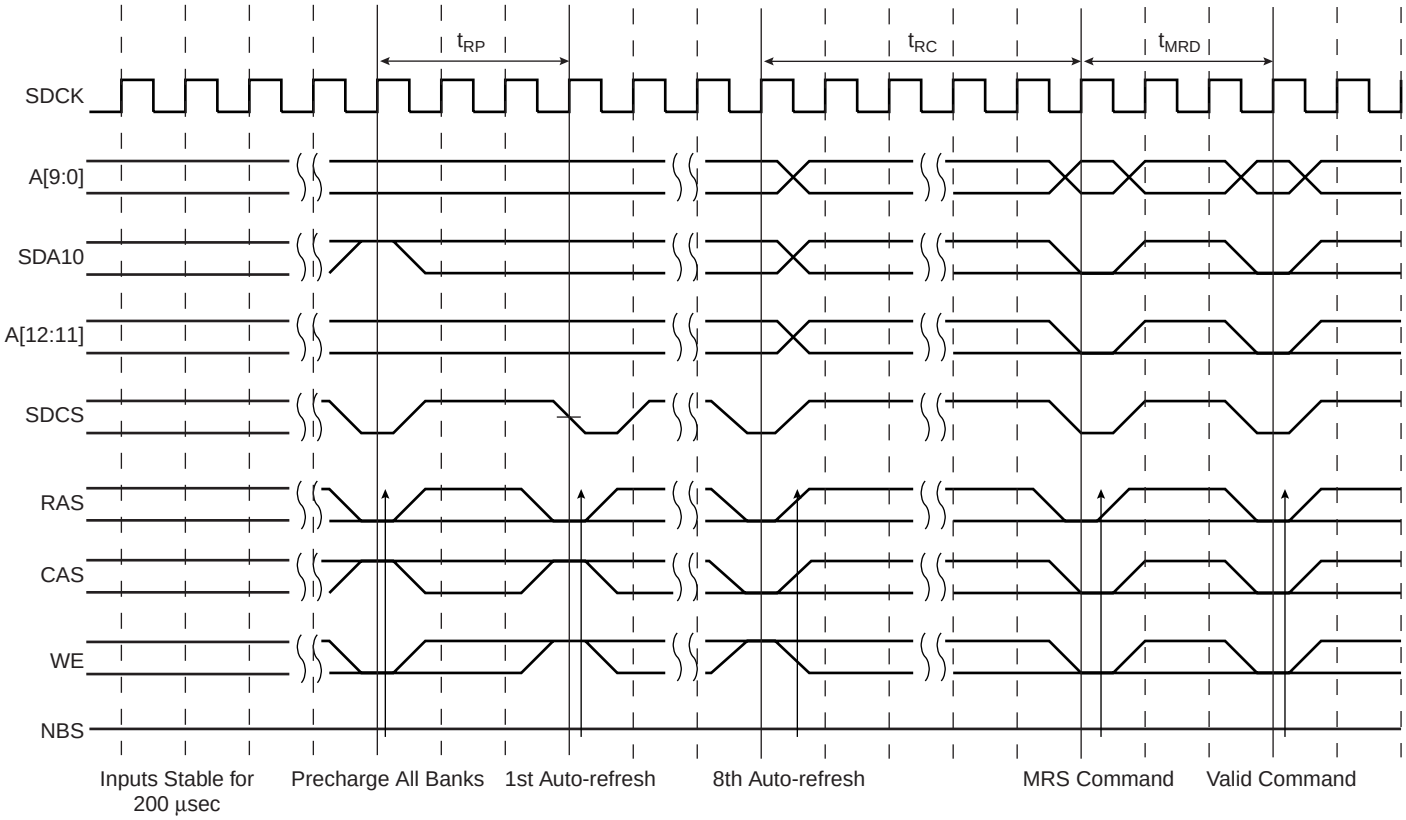
The initialization sequence is generated by software. The SDRAM devices are initialized by the following sequence:

1. A minimum pause of 200 µs is provided to precede any signal toggle.
2. An All Banks Precharge command is issued to the SDRAM devices.
3. Eight auto-refresh (CBR) cycles are provided.
4. A mode register set (MRS) cycle is issued to program the parameters of the SDRAM devices, in particular CAS latency and burst length.
5. A Normal Mode command is provided, 3 clocks after t_{MRD} is met.
6. Perform a dummy access in the SDRAM Memory Space to initialize the state machine.
7. Write refresh rate into the count field in the SDRAMC Refresh Timer register. (Refresh rate = delay between refresh cycles).

After these six steps, the SDRAM devices are fully functional.

The commands (NOP, MRS, CBR, normal mode) are generated by programming the command field in the SDRAMC Mode register.

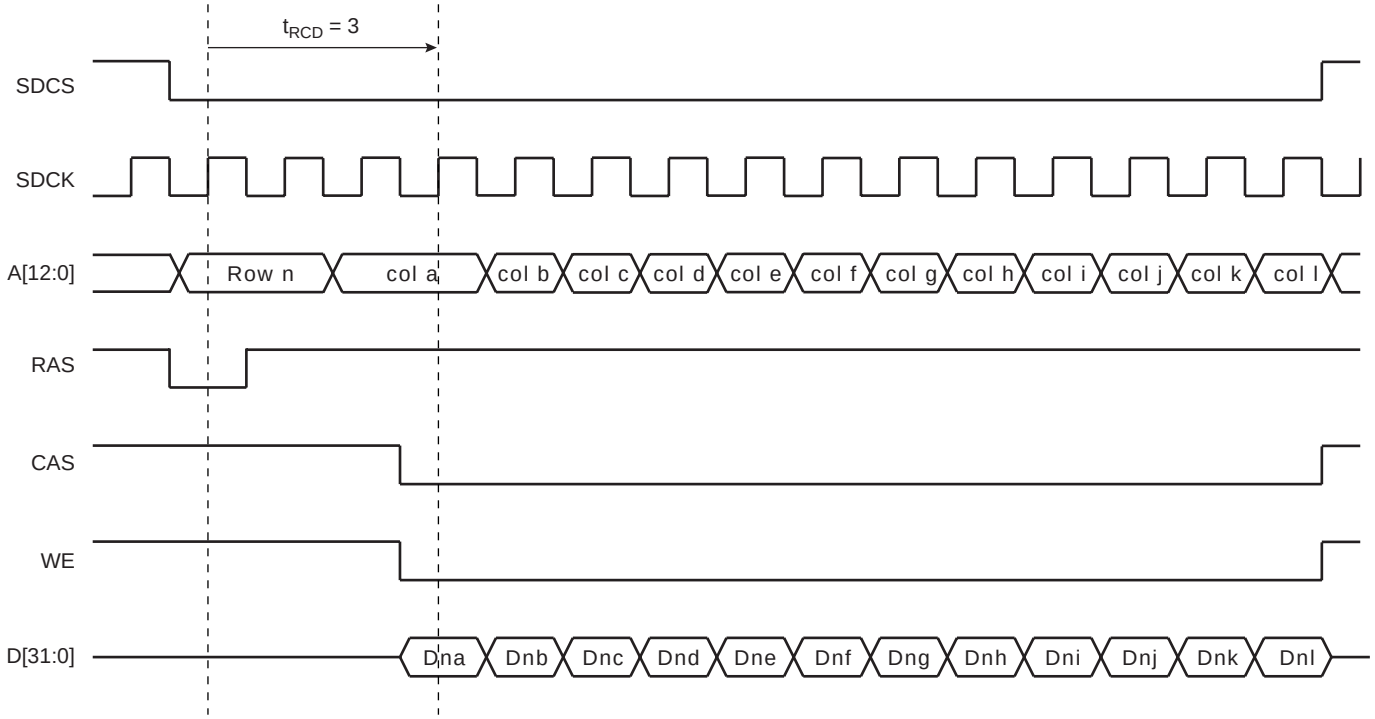
Figure 13. SDRAM Device Initialization Sequence



SDRAM Controller Write Cycle

The SDRAM Controller allows burst access or single access. To initiate a burst access, the SDRAM Controller uses the transfer type signal provided by the master requesting the access. If the next access is a sequential write access, writing to the SDRAM device is carried out. If the next access is a write-sequential access, but the current access is to a boundary page, or if the next access is in another row, then the SDRAM Controller generates a precharge command, activates the new row and initiates a write command. To comply with SDRAM timing parameters, additional clock cycles are inserted between precharge/active (t_{RP}) commands and active/write (t_{RCD}) commands. For a definition of these timing parameters, refer to the “SDRAMC Configuration Register” on page 39. This is described in Figure 14 below.

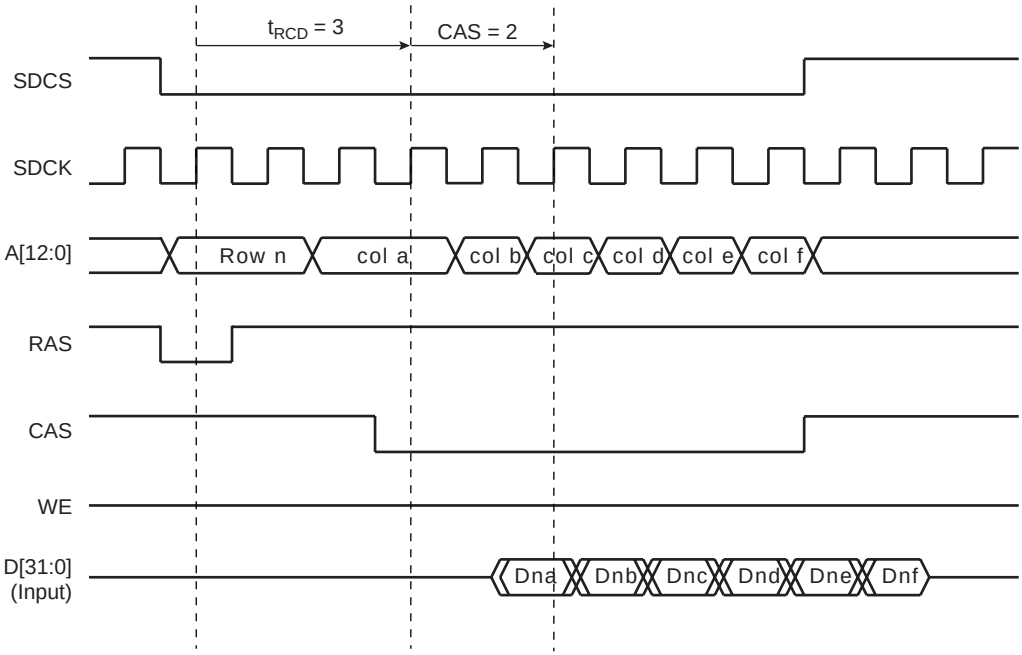
Figure 14. Write Burst, 32-bit SDRAM Access



SDRAM Controller Read Cycle

The SDRAM Controller allows burst access or single access. To initiate a burst access, the SDRAM Controller uses the transfer type signal provided by the master requesting the access. If the next access is a sequential read access, reading to the SDRAM device is carried out. If the next access is a sequential read access, but the current access is to a boundary page, or if the next access is in another row, then the SDRAM Controller generates a precharge command, activates the new row and initiates a read command. To comply with SDRAM timing parameters, an additional clock cycle is inserted between the precharge/active (t_{RP}) command and the active/read (t_{RCD}) command, After a read command, additional wait states are generated to comply with cas latency. The SDRAM Controller supports a cas latency of two. For definition of these timing parameters, refer to “SDRAMC Configuration Register” on page 39. This is described in Figure 15 below.

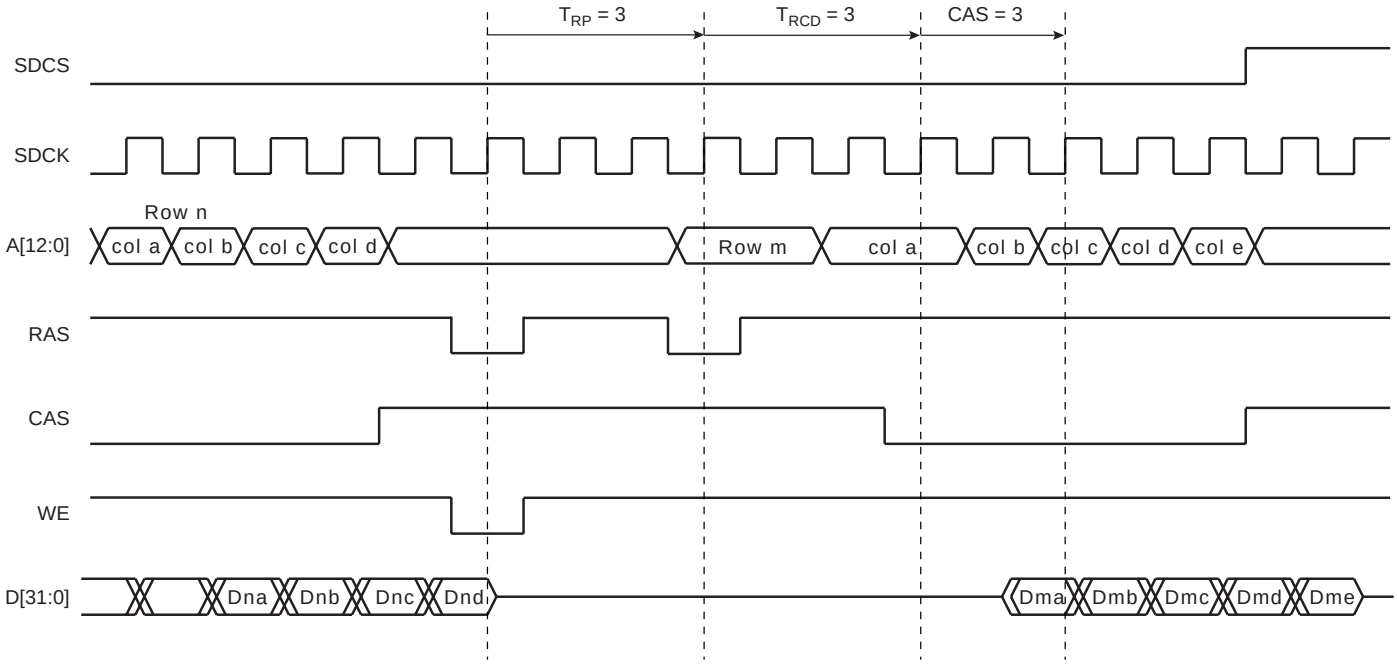
Figure 15. Read Burst, 32-bit SDRAM access



Border Management

When the memory row boundary has been reached, an automatic page break is inserted. In this case, the SDRAM controller generates a precharge command, activates the new row and initiates a read or write command. To comply with SDRAM timing parameters, an additional clock cycle is inserted between the precharge/active (t_{RP}) command and the active/read (t_{RCD}) command. This is described in Figure 16 below.

Figure 16. Read Burst with Boundary Row Access

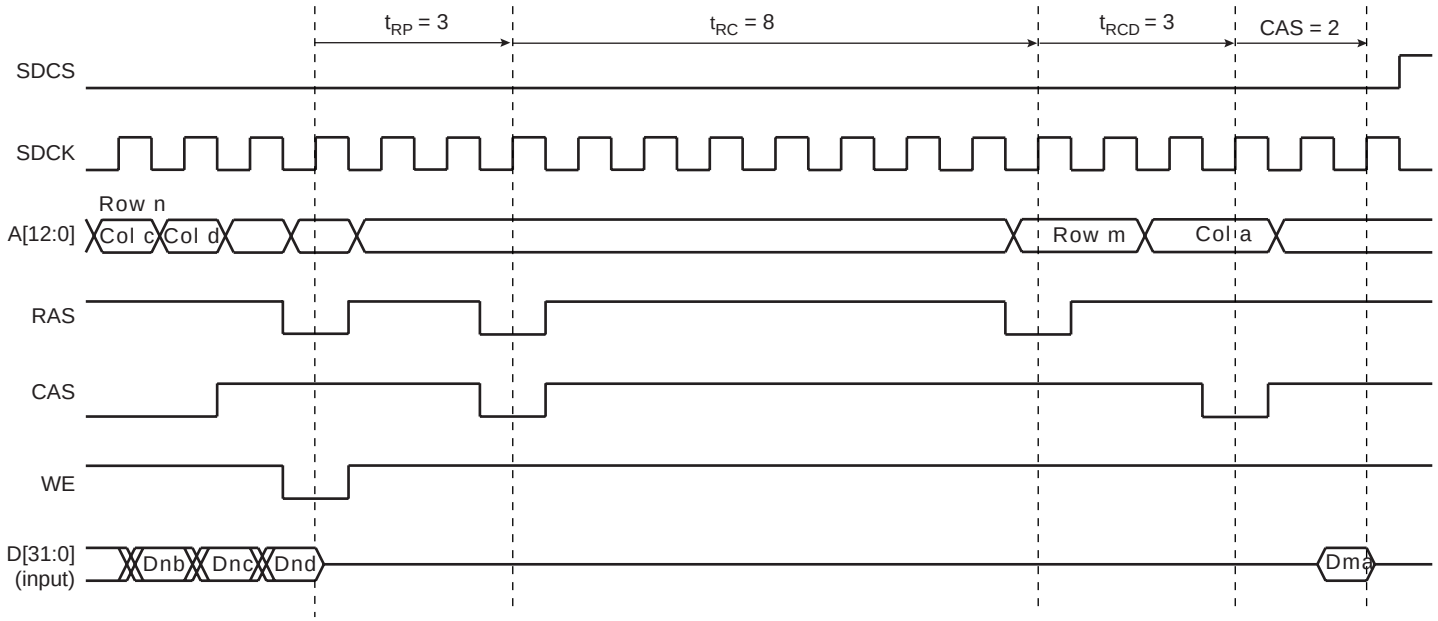


SDRAM Controller Refresh Cycles

An auto-refresh command is used to refresh the SDRAM device. Refresh addresses are generated internally by the SDRAM device and incremented after each auto-refresh automatically. The SDRAM Controller generates these auto-refresh commands periodically. A timer is loaded with the value in the register SDRAMC_TR that indicates the number of clock cycles between refresh cycles.

When the SDRAM Controller initiates a refresh of the SDRAM device, internal memory accesses are not delayed. However, if the ARM tries to access the SDRAM, it is held until the refresh cycle has completed. See Figure 17 below.

Figure 17. Refresh Cycle Followed by a Read Access



SDRAM User Interface

Base Address: 0xFF00 8000

Table 14. SDRAM Controller Register Mapping

Offset	Register Name	Register Description	Access	Reset State
0x00	SDRAMC_MR	SDRAMC Mode Register	Read/Write	0x00000000
0x04	SDRAMC_TR	SDRAMC Refresh Timer Register	Read/Write	0x00000800
0x08	SDRAMC_CR	SDRAMC Configuration Register	Read/Write	0x0299C140
0x0C	SDRAM_16BIT	SDRAM 16-bit configuration	Read/Write	0x00000001
0x10	SDRAMC_ADDR	Base address for SDCS	Read/Write	0x00000040

SDRAMC Mode Register

Register Name: SDRAMC_MR
Access Type: Read/Write
Reset Value: 0x00000010

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	MODE

7	6	5	4	3	2	1	0
MODE		–	–	–	–	–	–

- **MODE: SDRAMC Command Mode**

This field defines the command issued by the SDRAM Controller when the SDRAM device is accessed.

MODE			Description
0	0	0	Normal mode. Any access to the SDRAM is decoded normally.
0	0	1	The SDRAM Controller issues a NOP command when the SDRAM device is accessed regardless of the cycle.
0	1	0	The SDRAM Controller issues an “All Banks Precharge” command when the SDRAM device is accessed regardless of the cycle.
0	1	1	The SDRAM Controller issues a “Load Mode Register” command when the SDRAM device is accessed regardless of the cycle. The address offset with respect to the SDRAM device base address is used to program the Mode Register. For instance, when this mode is activated, an access to the “SDRAM_Base + offset” address generates a “Load Mode Register” command with the value “offset” written to the SDRAM device Mode Register.
1	0	0	The SDRAM Controller issues a “Refresh” Command when the SDRAM device is accessed regardless of the cycle. Prior to this, an “All Banks Precharge” command must be issued.
Others			Reserved

SDRAMC Refresh Timer Register

Register Name: SDRAMC_TR
Access Type: Read/Write
Reset Value: 0x00000800

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	COUNT			

7	6	5	4	3	2	1	0
COUNT							

- **COUNT: SDRAMC Refresh Timer Count**

This 12-bit field is loaded into a timer that generates the refresh pulse. Each time the refresh pulse is generated, a refresh burst is initiated. The value to be loaded depends on the SDRAMC clock frequency (MCK: Master Clock), the refresh rate of the SDRAM device and the refresh burst length where 15.6 µs per row is a typical value for a burst of one length.

To refresh the SDRAM device even if the reset value is not equal to 0, this 12-bit field must be written. If this condition is not satisfied, no refresh command is issued and no refresh of the SDRAM device is carried out.

SDRAMC Configuration Register

Register Name: SDRAMC_CR
Access Type: Read/Write
Reset Value: 0x0299C140

31	30	29	28	27	26	25	24
–	–	–	–	–	TRAS		

23	22	21	20	19	18	17	16
TRAS	TRCD				TRP		

15	14	13	12	11	10	9	8
TRP	TRC				TWR		

7	6	5	4	3	2	1	0
TWR	1	0	NB	NR		NC	

- **NC: Number of Column Bits**

Reset value is 8 column bits.

NC		Column Bits
0	0	8
0	1	9
1	0	10
1	1	11

- **NR: Number of Row Bits**

Reset value is 11 row bits.

NR		Row Bits
0	0	11
0	1	12
1	0	13
1	1	Reserved

- **NB: Number of Banks**

Reset value is two banks.

NB	Number of Banks
0	2
1	4

- **TWR: Write Recovery Delay**

Reset value is two cycles.

This field defines the Write Recovery Time in number of cycles. Number of cycles is between 2 and 15.

If TWR is less than or equal to 2, two clock periods are inserted by default.

- **TRC: Row Cycle Delay**

Reset value is eight cycles.

This field defines the delay between a Refresh and an Activate Command in number of cycles. Number of cycles is between 2 and 15.

If TRC is less than or equal to 2, two clock periods are inserted by default.

- **TRP: Row Precharge Delay**

Reset value is three cycles.

This field defines the delay between a Precharge Command and another Command in number of cycles. Number of cycles is between 2 and 15.

If TRP is less than or equal to 2, two clock periods are inserted by default.

- **TRCD: Row to Column Delay**

Reset value is three cycles.

This field defines the delay between an Activate Command and a Read/Write Command in number of cycles. Number of cycles is between 2 and 15.

If TRCD is less than or equal to 2, two clock periods are inserted by default.

- **TRAS: Active to Precharge Delay**

Reset value is five cycles.

This field defines the delay between an Activate Command and a Precharge Command in number of cycles. Number of cycles is between 2 and 15.

If TRAS is less than or equal to 2, two clock periods are inserted by default.

SDRAMC Address Register

Register Name: SDRAMC_ADDR
Access Type: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
SDCS_ADDR							

- **SDCS_ADDR**

This field defines the eight most significant bits of the base address of the SDRAMC.

Static Memory Controller (SMC)

The AT91C140 features a Static Memory Controller (SMC), that enables interfacing with a wide range of external static memory on peripheral devices, including Flash, ROM, static RAM, and parallel peripherals.

The SMC provides a glueless memory interface to external memory using common address, data bus and dedicated control signals. The SMC is highly programmable and has up to 24 bits of address bus, a 32- or 16-bit data bus and up to four chip select lines. The SMC supports different access protocols allowing single clock-cycle accesses. The SMC is programmed as an internal peripheral that has a standard APB bus interface and a set of memory-mapped registers. It shares the external address and data buses with the SDRAMC and any external bus master.

External Memory Mapping

The memory map associates the internal 32-bit address space with the external 24-bit address bus. The memory map is defined by programming the base address and page size of the external memories. Note that address bits A2 to A23 are significant for 32-bit memories whereas address bits A1 to A23 are significant for 16-bit memories.

If the physical memory-mapped device is smaller than the programmed page size, it wraps around and appears to be repeated within the page. The SMC correctly handles any valid access to the memory device within the page.

In the event of an access request to an address outside any programmed page, an abort signal is generated by the internal decoder. Two types of abort are possible: instruction prefetch abort and data abort. The corresponding exception vector addresses are 0x0000000C and 0x00000010. It is up to the system programmer to program the exception handling routine used in case of an abort.

Pin Description

Table 15 below lists the pins used by the SMC to control external memories.

Table 15. SMC Pin Description

FPDRAM	Description	Type
[A23:0]	Address bus	Output
[D31:0]	Data bus	I/O
NCE0-NCE3	Active low chip enable	Output
NWE0-NWE3	Active low byte select/write strobe signals	Output
NWR	Active low write strobe signals	Output
NSOE	Active low read enable signal	Output

Data Bus Width

A data bus width of 32 or 16 bits can be selected for each chip select. This option is controlled by the DBW field in the Chip Select Register (SMC_CSR) of the corresponding chip select.

The AT91C140 boots up with a data bus as defined by the DBW32 pin. If tied high, Chip Select 0 is automatically setup to be 32-bit wide. If tied low, the Chip Select 0 is configured to be 16-bit wide.

The DBW bit in SMC_CSR resets accordingly to the level of DBW32.

Byte Write or Byte Select Mode

Each chip select can be individually programmed to operate in Byte Write or Byte Select Mode.

- The Byte Write Mode supports four (32-bit bus) or two (16-bit bus) byte writes and a single read signal.
- The Byte Select Mode selects the appropriate byte(s) using four (32-bit bus) or two (16-bit bus) byte-select lines and separate read and write signals.

This option is controlled by the BAT bit in the Chip Select Register (SMC_CSR0 to SMC_CSR3).

The Byte Write Mode is used to connect four 8-bit devices on a 32-bit bus or two 8-bit devices on a 16-bit bus.

For a 32-bit bus:

- The NWE0 signal is used as the write enable signal for byte 0.
- The NWE1 signal is used as the write enable signal for byte 1.
- The NWE2 signal is used as the write enable signal for byte 2.
- The NWE3 signal is used as the write enable signal for byte 3.
- The NSOE signal enables memory reads to all memory blocks.

For a 16-bit bus:

- The NWE0 signal is used as the write enable signal for byte 0.
- The NWE1 signal is used as the write enable signal for byte 1.
- The NSOE signal enables memory reads to all memory blocks.

The Byte Select Mode is used to connect one 32-bit device or two 16-bit devices on a 32-bit data bus or one 16-bit device on a 16-bit data bus.

For a 32-bit bus:

- The NWE0 signal is used to select byte 0 for read and write operations.
- The NWE1 signal is used to select byte 1 for read and write operations.
- The NWE2 signal is used to select byte 2 for read and write operations.
- The NWE3 signal is used to select byte 3 for read and write operations.
- The NWR signal is used as the write enable signal for the memory block.
- The NSOE signal enables memory reads to the memory block.

For a 16-bit bus:

- The NWE0 signal is used to select byte 0 for read and write operations.
- The NWE1 signal is used to select byte 1 for read and write operations.
- The NWR signal is used as the write enable signal for the memory block.
- The NSOE signal enables memory reads to the memory block.

Read Protocols

The SMC provides two alternative protocols for external memory read access; standard and early read. The difference between the two protocols lies in the timing of the NSOE (read cycle) waveform.

The protocol is selected by the DRP field in the Memory Control Register (SMC_MCR) and is valid for all memory devices. Standard read protocol is the default protocol after reset.

Standard Read Protocol

Standard read protocol implements a read cycle in which NSOE and the write strobes are similar. Both are active during the second half of the clock cycle. The first half of the clock cycle allows time to ensure completion of the previous access, as well as the output of address and NCE before the read cycle begins.

During a standard read protocol external memory access, the chip enable signal sNCE0 to NCE3 are set low and the address lines are valid at the beginning of the access, whereas NSOE goes low only in the second half of the master clock cycle to avoid bus conflict. The write strobes are the same in both protocols. The write strobes always go low in the second half of the master clock cycle.

Early Read Protocol

Early read protocol provides more time for a read access from the memory by asserting NSOE at the beginning of the clock cycle. In the case of successive read cycles in the same memory, NSOE remains active continuously. Since a read cycle normally limits the speed of operation of the external memory system, early read protocol allows a faster clock frequency to be used. However, an extra wait state is required in some cases to avoid contention on the external bus.

In early read protocol, an early read wait state is automatically inserted when an external write cycle is followed by a read cycle to allow time for the write cycle to end before the subsequent read cycle begins. This wait state is generated in addition to any other programmed wait states (i.e., data float wait). No wait state is added when a read cycle is followed by a write cycle, between consecutive accesses of the same type or between external and internal memory accesses.

Write Protocol

During a write cycle, the data becomes valid after the falling edge of the write strobe signal and remains valid after the rising edge of the write strobe. The external write strobe waveform on the appropriate write strobe pin is used to control the output data timing to guarantee this operation.

Thus, it is necessary to avoid excessive loading of the write strobe pins, which could delay the write signal too long and cause a contention with a subsequent read cycle in standard protocol. In early read protocol, the data can remain valid longer than in standard read protocol due to the additional wait cycle that follows a write access.

Wait States

The SMC can automatically insert wait states. The different types of wait states are:

- Standard wait states
- Data float wait states
- Chip select change wait states
- Early read wait states, as described in “Early Read Protocol” above.

Standard Wait States

Each chip select can be programmed to insert one or more wait states during an access on the corresponding device. This is done by setting the WSE field in the corresponding SMC_CSR. The number of cycles to insert is programmed in the NWS field in the same register.

When no wait state is programmed (WSE = 0), the NWE signal lasts only one-half cycle. If at least one wait state is programmed, the NWE signal lasts an integer number of cycles, accordingly to the number of wait states programmed.

Data Float Wait States

Some memory devices are slow to release the external bus. For such devices it is necessary to add wait states (data float waits) after a read access before starting a write access or a read access to a different external memory.

The Data Float Output Time (TDF) for each external memory device is programmed in the TDF field of the SMC_CSR register for the corresponding chip select. The value (0 - 7 clock cycles) indicates the number of data float waits to be inserted and represents the time allowed for the data output to go to high impedance after the memory is disabled.

The SMC keeps track of the programmed external data float time even when it makes internal accesses to ensure that the external memory system is not accessed while it is still busy.

Internal memory accesses and consecutive accesses to the same external memory do not insert added data float wait states.

When data float wait states are being used, the SMC prevents the SDRAM Controller from accessing the external data bus.

Chip Select Change Wait States

A chip select wait state is automatically inserted when consecutive accesses are made to two different external memories (if no wait states have already been inserted). If any wait states have already been inserted (e.g., data float wait), then none are added.

Signal Waveforms

Figure 18 on page 45 shows a write to memory 0 followed by a write and a read to memory 1. SMC_CSR0 is programmed for one wait state with BAT = 0 and TDF = 0. SMC_CSR1 is programmed for zero wait states with BAT = 1 and TDF = 0. Early Read Protocol is enabled.

The write to memory 0 is a word access and therefore all four NWE strobes are active. As BAT = 0, they are configured as write strobes and have the same timing as NWR. As the access employs a single wait state, the write strobe pulse is one clock cycle long.

There is a chip select change wait state between the memory 0 write and the memory 1 write. The new address is output at the end of the memory 0 access, but the strobes are delayed for one clock cycle.

The write to memory 1 is a half-word access to an odd half-word address and, therefore, NWE2 and NWE3 are active. As BAT = 1, they are configured as byte select signals and have the same timing as NCE. As the access has no internal wait states, the write strobe pulse is one- half clock cycle long. Data and address are driven until the write strobe rising edge is sensed at the AT91C140 pin to guarantee positive hold times.

There is an early read wait state between memory 1 write and memory 1 read to provide time for the AT91C140 to disable the output data before the memory is read. If the read was normal mode, i.e., not early, the NSOE strobe would not fall until the rising edge of ACLK and no wait state would be inserted. If the write and early read were to different memories, then the early read wait state is not required as a chip select wait state will be implemented.

The read from memory 1 is a byte access to an address with a byte offset of 2 and therefore only NWE2 is active.

Figure 18. Write to Memory 0, Write and Read to Memory 1

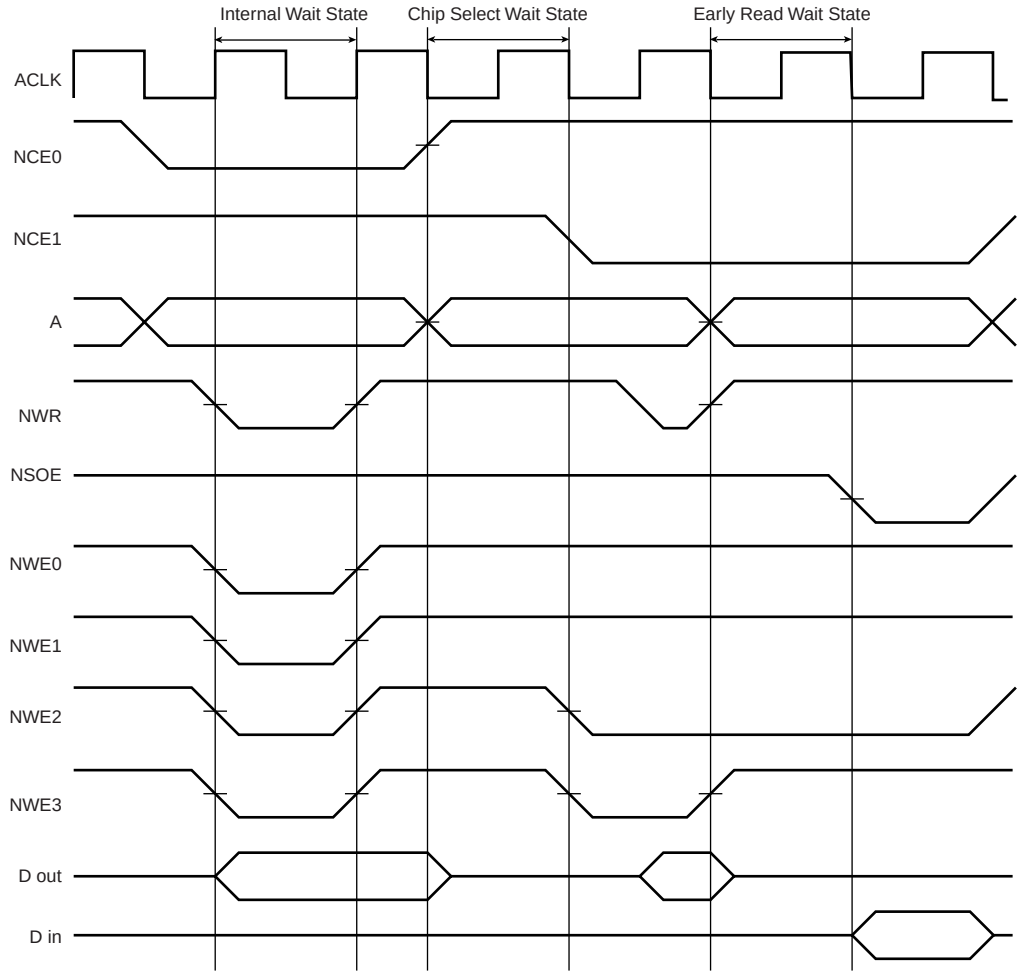


Figure 19 on page 46 shows a write and a read to memory 0 followed by a read and a write to memory 1. SMC_CSR0 is programmed for zero wait states with BAT = 0 and DFT = 0. SMC_CSR1 is programmed for zero wait states with BAT = 1 and DFT = 1. SMC_MCR is programmed for normal reads from all memories.

The write to memory 0 is a byte access and, therefore, only one NWE strobe is active. As BAT = 0, they are configured as write strobes and have the same timing as NWR.

The memory 0 read immediately follows the write as early reads are not configured and an early read wait state is not required. As early reads are not configured, the read strobe pulse is one-half clock cycle long.

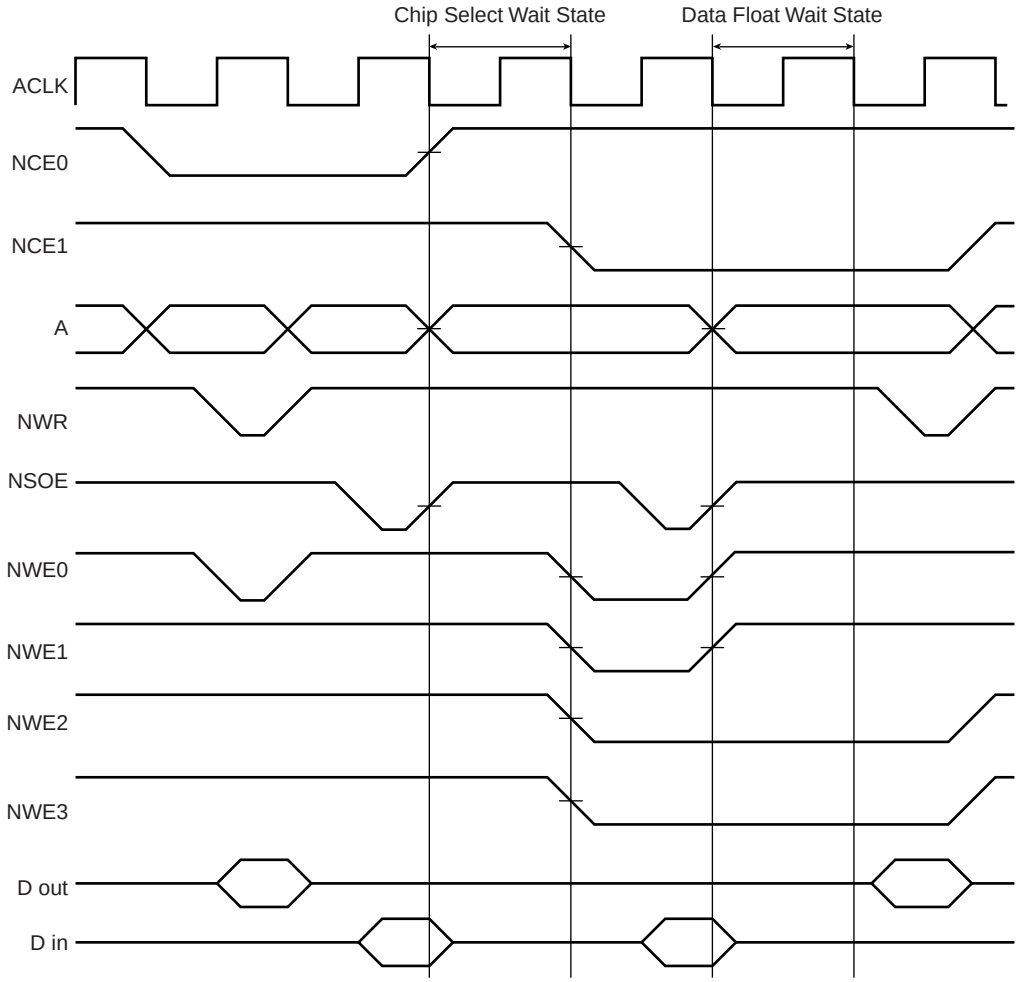
There is a chip select change wait state between the memory 0 write and the memory 1 read. The new address is output at the end of the memory 0 access but the strobes are delayed for one clock cycle.

The write to memory 1 is a half-word access to an odd half-word address and, therefore, NWE2 and NWE3 are active. As BAT = 1, they are configured as byte select signals and have the same timing as NCE.

As DFT = 1 for memory 1, a wait state is implemented between the read and write to provide time for the memory to stop driving the data bus. DFT wait states are only implemented at the end of read accesses.

The read from memory 1 is a byte access to an address with a byte offset of 2 and, therefore, only NWE2 is active.

Figure 19. Write and Read to Memory 0, Read and Write to Memory 1



SMC User Interface

The memory control register (SMC_MCR) is used to program the number of active chip selects and data read protocol. Four chip select registers (SMC_CSR0 to SMC_CSR3) are used to program the parameters for the individual external memories. Each SMC_CSR must be programmed with a different base address, even for unused chip selects.

The SMC_CSR register resets according to the DBW32 pin.

During the boot sequence, the Chip Select Registers must be programmed as required depending on the devices connected on the external bus. The chip select addresses that are programmed take effect immediately. Wait states also take effect immediately when they are programmed to optimize boot program execution.

Table 16. SMC Register Mapping

Offset	Register Name	Register Description	Access	Reset Value
0x00	SMC_CSR0	Chip Select Register	Read/Write	0x0000203D 0x0000203E
0x04	SMC_CSR1	Chip Select Register	Read/Write	0x10000000
0x08	SMC_CSR2	Chip Select Register	Read/Write	0x20000000
0x0C	SMC_CSR3	Chip Select Register	Read/Write	0x30000000
0x10	–	Reserved	–	–
0x14	–	Reserved	–	–
0x18	–	Reserved	–	–
0x1C	–	Reserved	–	–
0x20	–	Reserved	–	–
0x24	SMC_MCR	Memory Control Register	Read/ Write	0x0

SMC Chip Select Register

Register Name: SMC_CSR0..SMC_CSR3
Access: Read/Write

31	30	29	28	27	26	25	24
BA							

23	22	21	20	19	18	17	16
BA				–	–	–	–

15	14	13	12	11	10	9	8
–	–	CSEN	BAT	TDF			PAGES

7	6	5	4	3	2	1	0
PAGES	MWS	WSE	NWS			DBW	

- **DBW: Data Bus Width**

DBW		Data Bus Width
0	0	Reserved
0	1	16-bit external bus
1	0	32-bit external bus
1	1	Reserved

- **NWS: Number of Wait States**
- **WSE: Wait State Enable**
- **MWS: Multiply Wait States**

NWS			WSE	Wait States Number	
				MWS = 0	MWS = 1
X	X	X	0	0	0
0	0	0	1	1	8
0	0	1	1	2	16
0	1	0	1	3	24
0	1	1	1	4	32
1	0	0	1	5	40
1	0	1	1	6	48
1	1	0	1	7	56
1	1	1	1	8	64

- **PAGES: Page Size**

PAGES		Page Size	Base Address
0	0	1M byte	BA20-BA31
0	1	4M bytes	BA22-BA31
1	0	16M bytes	BA24-BA31
1	1	Reserved	–

- **TDF: Data Float Output Time**

TDF			Cycles after Transfer
0	0	0	0
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

- **BAT: Byte Access Mode**

0 = Byte Write Mode

1= Byte Select Mode

- **CSEN: Chip Select Enable**

0 = Chip Select is disabled

1 = Chip Select is enabled

- **BA: Base Address**

This field contains the high-order bits of the base address. If the page size is larger than 1M byte, then the unused bits of the base address are ignored by the SMC decoder.

SMC Memory Control Register

Register Name: SMC_MCR
Access Type: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	DRP	–	–	–	–

- **DRP: Data Read Protocol**

0 =Standard Read Mode

1 =Early Read Mode

Ethernet MAC (EMAC)

The AT91C140 features two identical Ethernet MACs, both of which feature the following:

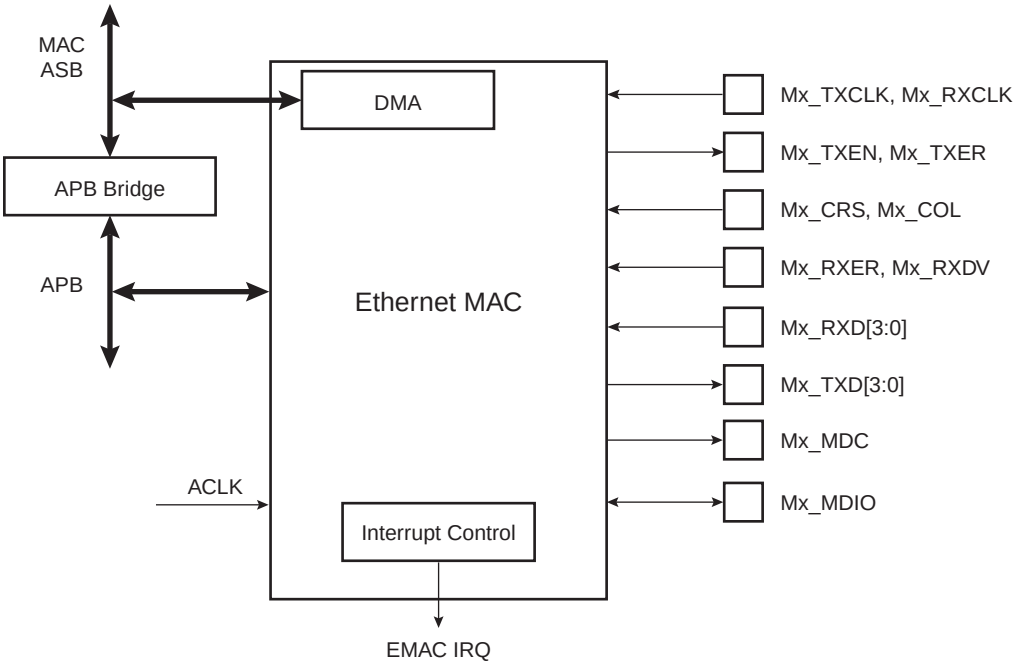
- Compatible with IEEE Standard 802.3
- 10 and 100 Mbits per Second Data Throughput Capability
- Full- and Half-duplex Operation
- Media Independent Interface to the Physical Layer
- Register Interface to Address, Status and Control Registers
- DMA Interface
- Interrupt Generation to Signal Receive and Transmit Completion
- 28-byte Transmit and 28-byte Receive FIFOs
- Automatic Pad and CRC Generation on Transmitted Frames
- Address Checking Logic to Recognize Four 48-bit Addresses
- Supports Promiscuous Mode Where All Valid Frames are Copied to Memory
- Supports Physical Layer Management through MDIO Interface

The Ethernet MAC is the hardware implementation of the MAC sub-layer OSI reference model between the physical layer (PHY) and the logical link layer (LLC). It controls the data exchange between a host and a PHY layer according to Ethernet IEEE 802.3 data frame format. The Ethernet MAC contains the required logic and transmit and receive FIFOs for DMA management. In addition, it is interfaced through MDIO/MDC pins for PHY layer management.

The Ethernet MAC transfers data in media-independent interface (MII).

Block Diagram

Figure 20. Block Diagram



Media Independent Interface

Table 17. Pin Configuration

MII Signal	Signal Name	Pin Name EMAC A	Pin Name EMAC B
Transmit Clock	ETXCK	MA_TXCLK	MB_TXCLK
Carrier Sense	ECRS	MA_CRS	MB_CRS
Collision Detect	ECOL	MA_COL	MB_COL
Receive Data Valid	ERXDV	MA_RXDV	MB_RXDV
4-bit Receive Data	ERX0-ERX3	MA_RXD[0:3]	MB_RXD[0:3]
Receive Error	ERXER	MA_RXER	MB_RXER
Receive Clock	ERXCK	MA_RXCLK	MB_RXCLK
Transmit Enable	ETXEN	MA_TXEN	MB_TXEN
4-bit Transmit Data	ETX0-ETX3	MA_TXD[0:3]	MB_TXD[0:3]
Transmit Error	ETXER	MA_TXER	MB_TXER

Transmit/Receive Operation

A standard IEEE 802.3 packet consists of the following fields: preamble, start of frame delimiter (SFD), destination address (DA), source address (SA), length, data (Logical Link Control Data) and frame check sequence CRC32 (FCS).

Table 18. Packet Format

Preamble		Frame[1]					
Alternating 1s/0s	SFD	DA	SA	Length/type	LLC Data	PAD	FCS
Up to 7 bytes	1 byte	6 bytes	6 bytes	2 bytes			4 bytes

Note: 1. Frame Length between 64 bytes and 1518 bytes.

The packets are Manchester-encoded and -decoded and transferred serially using NRZ data with a clock. All fields are of fixed length except for the data field. The MAC generates and appends the preamble, SFD and CRC fields during transmission.

The preamble and SFD fields are stripped during reception.

Preamble and Start of Frame Delimiter (SFD)

The preamble field is used to acquire bit synchronization with an incoming packet. When transmitted, each packet contains 62 bits of alternating 1,0 preamble. Some of this preamble is lost as the packet travels through the network. Byte alignment is performed with the Start of Frame Delimiter (SFD) pattern that consists of two consecutive 1's.

Destination Address (DA)

The destination address (DA) indicates the destination of the packet on the network and is used to filter unwanted packets. There are three types of address formats: physical, multicast and broadcast. The physical address is a unique address that corresponds only to a single node. All physical addresses have an MSB of 0.

Multicast addresses begin with an MSB of 1. The MAC filters multicast addresses using a standard hashing algorithm that maps all multicast addresses into a 6-bit value. This 6-bit value indexes a 64-bit array that filters the value. If the address consists of all ones, it is a broadcast address, indicating that the packet is intended for all nodes.

Source Address (SA)

The source address (SA) is the physical address of the node that sent the packet. Source addresses cannot be multicast or broadcast addresses. This field is passed to buffer memory.

Length/Type

If the value of this field is less than or equal to 1500, then the Length/Type field indicates the number of bytes in the subsequent LLC Data field. If the value of this field is greater than or equal to 1536, then the Length/Type field indicates the nature of the MAC client protocol (protocol type).

LLC Data

The data field consists of anywhere from 46 to 1500 bytes. Messages longer than 1500 bytes need to be broken into multiple packets. Messages shorter than 46 bytes require appending a pad to bring the data field to the minimum length of 46 bytes. If the data field is padded, the number of valid data bytes is indicated in the length field.

Frame Check Sequence Field (FCS)

The Frame Check Sequence (FCS) is a 32-bit CRC field, calculated and appended to a packet during transmission to allow detection of errors when a packet is received. During reception, error free packets result in a specific pattern in the CRC generator. Packets with improper CRC will be rejected.

Frame Format Extensions

The original Ethernet standards define the minimum frame size as 64 bytes and the maximum as 1518 bytes. These numbers include all bytes from the Destination MAC Address field through the Frame Check Sequence field. The Preamble and Start Frame Delimiter fields are not included when quoting the size of a frame. The IEEE 802.3ac standard extended the maximum allowable frame size to 1522 bytes to allow a VLAN tag to be inserted into the Ethernet frame format. The BIG bit defined in the ETH_CFG register processes packets with a VLAN tag.

The VLAN protocol permits insertion of an identifier, or tag, into the Ethernet frame format to identify the VLAN to which the frame belongs. It allows frames from stations to be assigned to logical groups. This provides various benefits, such as easing network administration, allowing formation of work groups, enhancing network security, and providing a means of limiting broadcast domains (refer to IEEE standard 802.1Q for definition of the VLAN protocol). The 802.3ac standard defines only the implementation details of the VLAN protocol that are specific to Ethernet.

If present, the 4-byte VLAN tag is inserted into the Ethernet frame between the Source MAC Address field and the Length field. The first 2 bytes of the VLAN tag consist of the “802.1Q Tag Type” and are always set to a value of 0x8100. The 0x8100 value is a reserved Length/Type field assignment that indicates the presence of the VLAN tag, and signals that the traditional Length/Type field can be found at an offset of four bytes further into the frame. The last two bytes of the VLAN tag contain the following information.

- The first three bits are a User Priority Field that may be used to assign a priority level to the Ethernet frame.
- The following one bit is a Canonical Format Indicator (CFI) used in Ethernet frames to indicate the presence of a Routing Information Field (RIF).
- The last twelve bits are the VLAN Identifier (VID) that uniquely identifies the VLAN to which the Ethernet frame belongs.

With the addition of VLAN tagging, the 802.3ac standard permits the maximum length of an Ethernet frame to be extended from 1518 bytes to 1522 bytes. Table 19 on page 53 illustrates the format of an Ethernet frame that has been “tagged” with a VLAN identifier according to the IEEE 802.3ac standard.

Table 19. Ethernet Frame with VLAN Tagging

Preamble	7 bytes
Start Frame Delimiter	1 byte
Dest. MAC Address	6 bytes
Source MAC Address	6 bytes
Length/Type = 802.1Q Tag Type	2 byte
Tag Control Information	2 bytes
Length / Type	2 bytes
MAC Client Data	0 - n bytes
Pad	0 - p bytes
Frame Check Sequence	4 bytes

DMA Operations

Frame data is transferred to and from the Ethernet MAC via the DMA interface. All transfers are 32-bit words and may be single accesses or bursts of two, three or four words. Burst accesses do not cross 16-byte boundaries.

The DMA controller performs four types of operations on the ASB bus. In order of priority, these operations are receive buffer manager read, receive buffer manager write, transmit data DMA read and receive data DMA write.

Transmitter Mode

Transmit frame data needs to be stored in contiguous memory locations. It does not need to be word-aligned.

The transmit address register is written with the address of the first byte to be transmitted.

Transmit is initiated by writing the number of bytes to transfer (length) to the transmit control register.

The transmit channel then reads data from memory 32 bits at a time and places them in the transmit FIFO.

The transmit block starts frame transmission when three words have been loaded into the FIFO.

The transmit address register must be written before the transmit control register. While a frame is being transmitted, it is possible to set up one other frame for transmission by writing new values to the transmit address and control registers. Reading the transmit address register returns the address of the buffer currently being accessed by the transmit FIFO.

Reading the transmit control register returns the total number of bytes to be transmitted. The BNQ bit in the Transmit Status Register indicates whether another buffer can be safely queued. An interrupt is generated whenever this bit is set.

Frame assembly starts by adding preamble and the start frame delimiter. Data is taken from the transmit FIFO word-by-word. If necessary, padding is added to make the frame length 60 bytes. The CRC is calculated as a 32-bit polynomial. This is inverted and appended to the end of the frame, making the frame length a minimum of 64 bytes. The CRC is not appended if the NCRC bit is set in the transmit control register.

In full-duplex mode, frames are transmitted immediately. Back-to-back frames are transmitted at least 96 bit times apart to guarantee the inter-frame gap.

In half-duplex mode, the transmitter checks carrier sense. If asserted, it waits for it to de-assert and then starts transmission after the inter-frame gap of 96 bit-times.

If the collision signal is asserted during transmission, the transmitter transmits a jam sequence of 32 bits taken from the data register and then retries transmission after the backoff time has elapsed. An error is indicated and any further attempts aborted if 16 attempts cause collisions.

If transmit DMA underruns, bad CRC is automatically appended using the same mechanism as jam insertion. Underrun also causes TXER to be asserted.

Receiver Mode

When a packet is received, it is checked for valid preamble, CRC, alignment, length and address. If all these criteria are met, the packet is stored successfully in a receive buffer. If at the end of reception the CRC is bad, then the received buffer is recovered. Each received frame including CRC is written to a single receive buffer.

Receive buffers are word-aligned and are capable of containing 1518 or 1522 bytes (BIG = 1 in ETH_CFG) of data (the maximum length of an Ethernet frame).

The start location for each received frame is stored in memory in a list of receive buffer descriptors at a location pointed to by the receive buffer queue pointer register. Each entry in the list consists of two words. The first word is the address of the received buffer; the second is the receive status. Table 20 defines an entry in the received buffer descriptor list.

To receive frames, the buffer queue must be initialized by writing an appropriate address to bits [31:2] in the first word of each list entry. Bit zero of word zero must be written with zero.

After a frame is received, bit zero becomes set and the second word indicates what caused the frame to be copied to memory. The start location of the received buffer descriptor list should be written to the received buffer queue pointer register before receive is enabled (by setting the receive enable bit in the network control register). As soon as the received block starts writing received frame data to the receive FIFO, the received buffer manager reads the first receive buffer location pointed to by the received buffer queue pointer register. If the filter block is active, the frame should be copied to memory; the receive data DMA operation starts writing data into the receive buffer. If an error occurs, the buffer is recovered. If the frame is received without error, the queue entry is updated. The buffer pointer is rewritten to memory with its low-order bit set to indicate successful frame reception and a used buffer. The next word is written with the length of the frame and how the destination address was recognized. The next receive buffer location is then read from the following word or, if the current buffer pointer had its wrap bit set, the beginning of the table. The maximum number of buffer pointers before a wrap bit is seen is 1024. If a wrap bit is not seen by then, a wrap bit is assumed in that entry. The received buffer queue pointer register must be written with zero in its lower-order bit positions to enable the wrap function to work correctly.

If bit zero is set when the receive buffer manager reads the location of the receive buffer, then the buffer has already been used and cannot be used again until software has processed the frame and cleared bit zero. In this case, the DMA block sets the buffer unavailable bit in the received status register and triggers an interrupt. The frame is discarded and the queue entry is reread on reception of the next frame to see if the buffer is now available. Each discarded frame increments a statistics register that is cleared on being read. When there is network congestion, it is possible for the MAC to be programmed to apply back pressure.

This is when half-duplex mode collisions are forced on all received frames by transmitting 64 bits of data (a default pattern).

Reading the received buffer queue register returns the location of the queue entry currently being accessed. The queue wraps around to the start after either 1024 entries (i.e., 2048 words) or when the wrap bit is found to be set in bit 1 of the first word of an entry.

Table 20. Received Buffer Descriptor List

Bit	Function
Word 0	
31:2	Base address of receive buffer
1	Wrap bit. If this bit is set, the counter that is ORed with the received buffer queue pointer register to give the pointer to entries in this table is cleared after the buffer is used.
0	Ownership bit. 1 indicates software owns the pointer, 0 indicates that the DMA owns the buffer. If this bit is not zero when the entry is read by the receiver, the buffer unavailable bit is set in the received status register and the receiver goes inactive.
Word 1	
31	Global all ones broadcast address detected
30	Multicast hash match
29	Unicast hash match
28	External address
27	Unknown source address (reserved for future use)
26	Local address match (Specific address 1 match)
25	Local address match (Specific address 2 match)
24	Local address match (Specific address 3 match)
23	Local address match (Specific address 4 match)
22:11	Reserved; written to 0
10:0	Length of frame including FCS

Address Checking

Whether or not a frame is stored depends on what is enabled in the network configuration register, the contents of the specific address and hash registers and the frame destination address. In this implementation of the MAC the frame source address is not checked.

A frame is not copied to memory if the MAC is transmitting in half-duplex mode at the time a destination address is received.

The hash register is 64 bits long and takes up two locations in the memory map.

There are four 48-bit specific address registers, each taking up two memory locations. The first location contains the first four bytes of the address; the second location contains the last two bytes of the address stored in its least significant byte positions. The addresses stored can be specific, group, local or universal.

Ethernet frames are transmitted a byte at a time, LSB first. The first bit (i.e., the LSB of the first byte) of the destination address is the group/individual bit and is set one for multicast addresses and zero for unicast. This bit corresponds to bit 24 of the first word of

the specific address register. The MSB of the first byte of the destination address corresponds to bit 31 of the specific address register.

The specific address registers are compared to the destination address of received frames once they have been activated. Addresses are deactivated at reset or when the first byte [47:40] is written and activated or when the last byte [7:0] is written. If a receive frame address matches an active address, the local match signal is set and the store frame pulse signal is sent to the DMA block via the ACLK synchronization block.

A frame can also be copied if a unicast or multicast hash match occurs, it has the broadcast address of all ones, or the copy all frames bit in the network configuration register is set.

The broadcast address of 0xFFFFFFFF is recognized if the no broadcast bit in the network configuration register is zero. This sets the broadcast match signal and triggers the store frame signal.

The unicast hash enable and the multicast hash enable bits in the network configuration register enable the reception of hash matched frames. So all multicast frames can be received by setting all bits in the hash register.

The CRC algorithm reduces the destination address to a 6-bit index into a 64-bit hash register.If the equivalent bit in the register is set, the frame is matched depending on whether the frame is multicast or unicast and the appropriate match signals are sent to the DMA block. If the copy all frames bit is set in the network configuration register, the store frame pulse is always sent to the DMA block as soon as any destination address is received.

EMAC User Interface

MACA Memory Address: 0xFF034000
MACB Memory Address: 0xFF038000

Table 21. Ethernet MAC Register Mapping

Offset	Register Name	Register Description	Read/Write	Reset
0x00	ETH_CTL	Network Control Register	Read/Write	0x0
0x04	ETH_CFG	Network Configuration Register	Read/Write	0x800
0x08	ETH_SR	Network Status Register	Read-only	0x6
0x0C	ETH_TAR	Transmit Address Register	Read/Write	0x0
0x10	ETH_TCR	Transmit Control Register	Read/Write	0x0
0x14	ETH_TSR	Transmit Status Register	Read/Write	0x18
0x18	ETH_RBQP	Receive Buffer Queue Pointer	Read/Write	0x0
0x1C	–	Reserved	Read-only	0x0
0x20	ETH_RSR	Receive Status Register	Read/Write	0x0
0x24	ETH_ISR	Interrupt Status Register	Read/Write	0x0
0x28	ETH_IER	Interrupt Enable Register	Write-only	–
0x2C	ETH_IDR	Interrupt Disable Register	Write-only	–
0x30	ETH_IMR	Interrupt Mask Register	Read-only	0xFFF
0x34	ETH_MAN	PHY Maintenance Register	Read/Write	0x0
		Statistics Registers[1]		
0x40	ETH_FRA	Frames Transmitted OK Register	Read/Write	0x0
0x44	ETH_SCOL	Single Collision Frame Register	Read/Write	0x0
0x48	ETH_MCOL	Multiple Collision Frame Register	Read/Write	0x0
0x4C	ETH_OK	Frames Received OK Register	Read/Write	0x0
0x50	ETH_SEQE	Frame Check Sequence Error Register	Read/Write	0x0
0x54	ETH_ALE	Alignment Error Register	Read/Write	0x0
0x58	ETH_DTE	Deferred Transmission Frame Register	Read/Write	0x0
0x5C	ETH_LCOL	Late Collision Register	Read/Write	0x0
0x60	ETH_ECOL	Excessive Collision Register	Read/Write	0x0
0x64	ETH_CSE	Carrier Sense Error Register	Read/Write	0x0
0x68	ETH_TUE	Transmit Underrun Error Register	Read/Write	0x0
0x6C	ETH_CDE	Code Error Register	Read/Write	0x0
0x70	ETH_ELR	Excessive Length Error Register	Read/Write	0x0
0x74	ETH_RJB	Receive Jabber Register	Read/Write	0x0
0x78	ETH_USF	Undersize Frame Register	Read/Write	0x0
0x7C	ETH_SQEE	SQE Test Error Register	Read/Write	0x0
0x80	ETH_DRFC	Discarded RX Frame Register	Read/Write	0x0
		Address Registers		
0x90	ETH_HSH	Hash Address High [63:32]	Read/Write	0x0
0x94	ETH_HSL	Hash Address Low [31:0]	Read/Write	0x0
0x98	ETH_SA1L	Specific Address 1 Low, First 4 Bytes	Read/Write	0x0
0x9C	ETH_SA1H	Specific Address 1 High, Last 2 Bytes	Read/Write	0x0
0xA0	ETH_SA2L	Specific Address 2 Low, First 4 Bytes	Read/Write	0x0
0xA4	ETH_SA2H	Specific Address 2 High, Last 2 Bytes	Read/Write	0x0
0xA8	ETH_SA3L	Specific Address 3 Low, First 4 Bytes	Read/Write	0x0
0xAC	ETH_SA3H	Specific Address 3 High, Last 2 Bytes	Read/Write	0x0
0xB0	ETH_SA4L	Specific Address 4 Low, First 4 Bytes	Read/Write	0x0
0xB4	ETH_SA4H	Specific Address 4 High, Last 2 Bytes	Read/Write	0x0

Note: 1. For further details on the statistics registers, see Table 22, “Statistics Register Block,” on page 71.

EMAC Control Register

Register Name: ETH_CTL
Access Type: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	BP
7	6	5	4	3	2	1	0
WES	ISR	CSR	MPE	TE	RE	LBL	LB

- **LB: Loopback**

. When set, loopback signal is at high level.

- **LBL: Loopback Local**

When set, connects ETX[3:0] to ERX[3:0], ETXEN to ERXDV, forces full duplex and drives ERXCK and ETXCK_REFCK with ACK divided by 4.

- **RE: Receive Enable**

When set, enables the Ethernet MAC to receive data.

- **TE: Transmit Enable**

When set, enables the Ethernet transmitter to send data.

- **MPE: Management Port Enable**

Set to one to enable the management port. When zero, forces MDIO to high impedance state.

- **CSR: Clear Statistics Registers**

This bit is write-only. Writing a one clears the statistics registers.

- **ISR: Increment Statistics Registers**

This bit is write-only. Writing a one increments all the statistics registers by one for test purposes.

- **WES: Write Enable for Statistics Registers**

Setting this bit to one makes the statistics registers writable for functional test purposes.

- **BP: Back Pressure**

If this field is set, then in half-duplex mode collisions are forced on all received frames by transmitting 64 bits of data (default pattern).

EMAC Mode Register

Name: ETH_CFG
Access Type: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	RTY	CLK		EAE	BIG
7	6	5	4	3	2	1	0
UNI	MTI	NBC	CAF	–	BR	FD	SPD

- **SPD: Speed**

Set to 1 to indicate 100 Mbit/sec, 0 for 10 Mbit/sec. Has no other functional effect.

- **FD: Full Duplex**

If set to 1, the transmit block ignores the state of collision and carrier sense and allows receive while transmitting.

- **BR: Bit Rate**
- **CAF: Copy All Frames**

When set to 1, all valid frames are received.

- **NBC: No Broadcast**

When set to 1, frames addressed to the broadcast address of all ones are not received.

- **MTI: Multicast Hash Enable**

When set multicast frames are received when six bits of the CRC of the destination address point to a bit that is set in the hash register.

- **UNI: Unicast Hash Enable**

When set, unicast frames are received when six bits of the CRC of the destination address point to a bit that is set in the hash register.

- **BIG: Receive 1522 Bytes**

When set, the MAC receives up to 1522 bytes. Normally the MAC receives frames up to 1518 bytes in length.

This bit allows to receive extended Ethernet frame with “VLAN tag” (IEEE 802.3ac)

- **EAE: External Address Match Enable**
- **CLK**

The ARM clock is divided down to generate MDC (the clock for the MDIO). To conform with IEEE standard 802.3 MDC must not exceed 2.5 MHz. At reset this field is set to 10 so that ACK is divided by 32.

CLK	MDC
00	ACK divided by 8
01	ACK divided by 16
10	ACK divided by 32
11	ACK divided by 64

- **RTY: Retry Test**

When set, the time between frames is always one time slot. For test purposes only. Must be cleared for normal operation.

EMAC Status Register

Name: ETH_SR
Access Type: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	IDLE	MDIO	LINK

- **LINK**

0 = LINK is at 0.

1 = LINK is at 1.

- **MDIO**

0 = MDIO pin not set.

1 = MDIO pin set.

- **IDLE**

0 = PHY logic is idle.

1 = PHY logic is running.

EMAC Transmit Address Register

Name: ETH_TAR
Access Type: Read/Write

31	30	29	28	27	26	25	24
ADDRESS							
23	22	21	20	19	18	17	16
ADDRESS							
15	14	13	12	11	10	9	8
ADDRESS							
7	6	5	4	3	2	1	0
ADDRESS							

- **ADDRESS: Transmit Address Register**

Written with the address of the frame to be transmitted, read as the base address of the buffer being accessed by the transmit FIFO. Note that if the two least significant bits are not zero, transmit starts at the byte indicated.

EMAC Transmit Control Register

Name: ETH_TCR
Access Type: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
NCRC	–	–	–	–	LEN		
7	6	5	4	3	2	1	0
LEN							

- **LEN: Transmit Frame Length**

This register is written to the number of bytes to be transmitted excluding the four CRC bytes unless the no CRC bit is asserted. Writing these bits to any non-zero value initiates a transmission. If the value is greater than 1514 (1518 if no CRC is being generated), an oversize frame is transmitted. This field is buffered so that a new frame can be queued while the previous frame is still being transmitted. Must always be written in address-then-length order. Reads as the total number of bytes to be transmitted (i.e., this value does not change as the frame is transmitted.) Frame transmission does not start until two 32-bit words have been loaded into the transmit FIFO. The length must be great enough to ensure two words are loaded.

- **NCRC: No CRC**

If this bit is set, it is assumed that the CRC is included in the length being written in the low-order bits and the MAC does not append CRC to the transmitted frame. If the buffer is not at least 64 bytes long, a short frame is sent. This field is buffered so that a new frame can be queued while the previous frame is still being transmitted. Reads as the value of the frame currently being transmitted.

EMAC Transmit Status Register

Name: ETH_TSR
Access Type: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	UND	COMP	BNQ	IDLE	RLE	COL	OVR

- **OVR: Ethernet Transmit Buffer Overrun**

Software has written to the Transmit Address Register (ETH_TAR) or Transmit Control Register (ETH_TCR) when bit BNQ was not set. Cleared by writing a one to this bit.

- **COL: Collision Occurred**

Set by the assertion of a collision. Cleared by writing a one to this bit.

- **RLE: Retry Limit Exceeded**

Cleared by writing a one to this bit.

- **IDLE: Transmitter Idle**

Asserted when the transmitter has no frame to transmit. Cleared when a length is written to transmit frame length portion of the Transmit Control register. This bit is read-only.

- **BNQ: Ethernet Transmit Buffer not Queued**

Software may write a new buffer address and length to the transmit DMA controller when set. Cleared by having one frame ready to transmit and another in the process of being transmitted. This bit is read-only.

- **COMP: Transmit Complete**

Set when a frame has been transmitted. Cleared by writing a one to this bit.

- **UND: Transmit Underrun**

Set when transmit DMA was not able to read data from memory in time. If this happens, the transmitter forces bad CRC. Cleared by writing a one to this bit.

EMAC Receive Buffer Queue Pointer Register

Name: ETH_RBQP
Access Type: Read/Write

31	30	29	28	27	26	25	24
ADDRESS							
23	22	21	20	19	18	17	16
ADDRESS							
15	14	13	12	11	10	9	8
ADDRESS							
7	6	5	4	3	2	1	0
ADDRESS							

- **ADDRESS: Receive Buffer Queue Pointer**

Written with the address of the start of the receive queue, reads as a pointer to the current buffer being used. The receive buffer is forced to word alignment.

EMAC Receive Status Register

Name: ETH_RSR
Access Type: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	OVR	REC	BNA

- **BNA: Buffer Not Available**

An attempt was made to get a new buffer and the pointer indicated that it was owned by the processor. The DMA rereads the pointer each time a new frame starts until a valid pointer is found. This bit is set at each attempt that fails even if it has not had a successful pointer read since it has been cleared. Cleared by writing a one to this bit.

- **REC: Frame Received**

One or more frames have been received and placed in memory. Cleared by writing a one to this bit.

- **OVR: RX Overrun**

The DMA block was unable to store the receive frame to memory, either because the MAC ASB bus was not granted in time or because an abort occurred. The buffer is recovered if this happens. Cleared by writing a one to this bit.

EMAC Interrupt Status Register

Name: ETH_ISR
Access Type: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	ABT	ROVR	LINK	TIDLE
7	6	5	4	3	2	1	0
TCOM	TBRE	RTRY	TUND	TOVR	RBNA	RCOM	DONE

- **DONE: Management Done**

The PHY maintenance register has completed its operation. Cleared on read.

- **RCOM: Receive Complete**

A frame has been stored in memory. Cleared on read.

- **RBNA: Receive Buffer Not Available**

Cleared on read.

- **TOVR: Transmit Buffer Overrun**

Software has written to the Transmit Address Register (ETH_TAR) or Transmit Control Register (ETH_TCR) when BNQ of the Transmit Status Register (ETH_TSR) was not set. Cleared on read.

- **TUND: Transmit Buffer Underrun**

Ethernet transmit buffer underrun. The transmit DMA did not complete fetch frame data in time for it to be transmitted. Cleared on read.

- **RTRY: Retry Limit**

Retry limit exceeded. Cleared on read.

- **TBRE: Transmit Buffer Register Empty**

Software may write a new buffer address and length to the transmit DMA controller. Cleared by having one frame ready to transmit and another in the process of being transmitted. Cleared on read.

- **TCOM: Transmit Complete**

Set when a frame has been transmitted. Cleared on read.

- **TIDLE: Transmit Idle**

Set when all frames have been transmitted. Cleared on read.

- **LINK**

Set when LINK pin changes value.

- **ROVR: RX Overrun**

Set when the RX overrun status bit is set. Cleared on read.

- **ABT: Abort**

Set when the DMA generates an Abort. Cleared on read.

EMAC Interrupt Enable Register

Name: ETH_IER
Access Type: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	ABT	ROVR	LINK	TIDLE
7	6	5	4	3	2	1	0
TCOM	TBRE	RTRY	TUND	TOVR	RBNA	RCOM	DONE

- **DONE: Management Done Interrupt Enable**
- **RCOM: Receive Complete Interrupt Enable**
- **RBNA: Receive Buffer Not Available Interrupt Enable**
- **TOVR: Transmit Buffer Overrun Interrupt Enable**
- **TUND: Transmit Buffer Underrun Interrupt Enable**
- **RTRY: Retry Limit Interrupt Enable**
- **TBRE: Transmit Buffer Register Empty Interrupt Enable**
- **TCOM: Transmit Complete Interrupt Enable**
- **TIDLE: Transmit Idle Interrupt Enable**
- **LINK: LINK Interrupt Enable**
- **ROVR: RX Overrun Interrupt Enable**
- **ABT: Abort Interrupt Enable**

0 =No effect.

1 =Enables the corresponding interrupt.

EMAC Interrupt Disable Register

Name: ETH_IDR
Access Type: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	ABT	ROVR	LINK	TIDLE
7	6	5	4	3	2	1	0
TCOM	TBRE	RTRY	TUND	TOVR	RBNA	RCOM	DONE

- **DONE: Management Done Interrupt Disable**
- **RCOM: Receive Complete Interrupt Disable**
- **RBNA: Receive Buffer Not Available Interrupt Disable**
- **TOVR: Transmit Buffer Overrun Interrupt Disable**
- **TUND: Transmit Buffer Underrun Interrupt Disable**
- **RTRY: Retry Limit Interrupt Disable**
- **TBRE: Transmit Buffer Register Empty Interrupt Disable**
- **TCOM: Transmit Complete Interrupt Disable**
- **TIDLE: Transmit Idle Interrupt Disable**
- **LINK: LINK Interrupt Disable**
- **ROVR: RX Overrun Interrupt Disable**
- **ABT: Abort Interrupt Disable**

0 =No effect.

1 =Disables the corresponding interrupt.

EMAC Interrupt Mask Register

Name: ETH_IMR
Access Type: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	ABT	ROVR	LINK	TIDLE
7	6	5	4	3	2	1	0
TCOM	TBRE	RTRY	TUND	TOVR	RBNA	RCOM	DONE

- **DONE: Management Done Interrupt Mask**
- **RCOM: Receive Complete Interrupt Mask**
- **RBNA: Receive Buffer Not Available Interrupt Mask**
- **TOVR: Transmit Buffer Overrun Interrupt Mask**
- **TUND: Transmit Buffer Underrun Interrupt Mask**
- **RTRY: Retry Limit Interrupt Mask**
- **TBRE: Transmit Buffer Register Empty Interrupt Mask**
- **TCOM: Transmit Complete Interrupt Mask**
- **TIDLE: Transmit Idle Interrupt Mask**
- **LINK: LINK Interrupt Mask**
- **ROVR: RX Overrun Interrupt Mask**
- **ABT: Abort Interrupt Mask**

0 =The corresponding interrupt is enabled.

1 =The corresponding interrupt is not enabled.

Important Note: The interrupt is disabled when the corresponding bit is set. This is non-standard with other peripherals of the product, as generally a mask bit set enables the interrupt.

EMAC PHY Maintenance Register

Name: ETH_MAN
Access Type: Read/Write

31	30	29	28	27	26	25	24
LOW	HIGH	RW		PHYA			

23	22	21	20	19	18	17	16
PHYA	REGA					CODE	

15	14	13	12	11	10	9	8
DATA							

7	6	5	4	3	2	1	0
DATA							

Writing to this register starts the shift register that controls the serial connection to the PHY. On each shift cycle the MDIO pin becomes equal to the MSB of the shift register and LSB of the shift register becomes equal to the value of the MDIO pin. When the shifting is complete an interrupt is generated and the IDLE field is set in the Network Status register.

When read, gives current shifted value.

- **DATA**

For a write operation this is written with the data to be written to the PHY. After a read operation this contains the data read from the PHY.

- **CODE**

Must be written to 10 in accordance with IEEE standard 802.3. Reads as written.

- **REGA**

Register address. Specifies the register in the PHY to access.

- **PHYA**

PHY address. Normally is 0.

- **RW**

Read/Write Operation. 10 is read. 01 is write. Any other value is an invalid PHY management frame.

- **HIGH**

Must be written with 1 to make a valid PHY management frame. Conforms with IEEE standard 802.3.

- **LOW**

Must be written with 0 to make a valid PHY management frame. Conforms with IEEE standard 802.3.

EMAC Hash Address High Register

Register Name: ETH_HSH
Access Type: Read/Write

31	30	29	28	27	26	25	24
ADDR							
23	22	21	20	19	18	17	16
ADDR							
15	14	13	12	11	10	9	8
ADDR							
7	6	5	4	3	2	1	0
ADDR							

- **ADDR**

Hash address bits 63 to 32.

EMAC Hash Address Low Register

Register Name: ETH_HSL
Access Type: Read/Write

31	30	29	28	27	26	25	24
ADDR							
23	22	21	20	19	18	17	16
ADDR							
15	14	13	12	11	10	9	8
ADDR							
7	6	5	4	3	2	1	0
ADDR							

- **ADDR**

Hash address bits 31 to 0.

EMAC Specific Address (1, 2, 3 and 4) High Register

Register Name: ETH_SA1H,...ETH_SA4H
Access Type: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
ADDR							
7	6	5	4	3	2	1	0
ADDR							

- **ADDR**

Unicast addresses (1, 2, 3 and 4), Bits 47:32.

EMAC Specific Address (1, 2, 3 and 4) Low Register

Register Name: ETH_SA1L,...ETH_SA4L
Access Type: Read/Write

31	30	29	28	27	26	25	24
ADDR							
23	22	21	20	19	18	17	16
ADDR							
15	14	13	12	11	10	9	8
ADDR							
7	6	5	4	3	2	1	0
ADDR							

- **ADDR**

Unicast addresses (1, 2, 3 and 4), Bits 31:0.

EMAC Statistics Register Block Registers

These registers reset to zero on a read and remain at all ones when they count to their maximum value. They should be read frequently enough to prevent loss of data.

The statistics register block contains the registers found in Table 21, “Ethernet MAC Register Mapping,” on page 57.

Table 22. Statistics Register Block

Register	Register Name	Description
Frames Transmitted OK Register	ETH_FRA	A 24-bit register counting the number of frames successfully transmitted.
Single Collision Frame Register	ETH_SCOL	A 16-bit register counting the number of frames experiencing a single collision before being transmitted and experiencing no carrier loss nor underrun.
Multiple Collision Frame Register	ETH_MCOL	A 16-bit register counting the number of frames experiencing between two and fifteen collisions prior to being transmitted (62 - 1518 bytes, no carrier loss, no underrun).
Frames Received OK Register	ETH_OK	A 24-bit register counting the number of good frames received, i.e., address recognized. A good frame is of length 64 to 1518 bytes and has no FCS, alignment or code errors.
Frame Check Sequence Error Register	ETH_SEQE	An 8-bit register counting address-recognized frames that are an integral number of bytes long, that have bad CRC and that are 64 to 1518 bytes long.
Alignment Error Register	ETH_ALE	An 8-bit register counting frames that: - are address-recognized, - are not an integral number of bytes long, - have bad CRC when their length is truncated to an integral number of bytes, - are between 64 and 1518 bytes long.
Deferred Transmission Frame Register	ETH_DTE	A 16-bit register counting the number of frames experiencing deferral due to carrier sense active on their first attempt at transmission (no underrun or collision).
Late Collision Register	ETH_LCOL	An 8-bit register counting the number of frames that experience a collision after the slot time (512 bits) has expired. No carrier loss or underrun. A late collision is counted twice, i.e., both as a collision and a late collision.
Excessive Collision Register	ETH_ECOL	An 8-bit register counting the number of frames that failed to be transmitted because they experienced 16 collisions (64 - 1518 bytes, no carrier loss or underrun).
Carrier Sense Error Register	ETH_CSE	An 8-bit register counting the number of frames for which carrier sense was not detected and that were maintained in half-duplex mode one slot time (512 bits) after the start of transmission (no excessive collision).
Transmit Underrun Error Register	ETH_TUE	An 8-bit register counting the number of frames not transmitted due to a transmit DMA underrun. If this register is incremented, then no other register is incremented.
Code Error Register	ETH_CDE	An 8-bit register counting the number of frames that are address-recognized, had RXER asserted during reception. If this counter is incremented, then no other counters are incremented.
Excessive Length Error Register	ETH_ELR	An 8-bit register counting the number of frames received exceeding 1518 bytes in length but that do not have either a CRC error, an alignment error or a code error.
Receive Jabber Register	ETH_RJB	An 8-bit register counting the number of frames received exceeding 1518 bytes in length and having either a CRC error, an alignment error or a code error.
Undersize Frame Register	ETH_USF	An 8-bit register counting the number of frames received that are less than 64 bytes in length but that do not have either a CRC error, an alignment error or a code error.
SQE Test Error Register	ETH_SQEE	An 8-bit register counting the number of frames where pin ECOL was not asserted within a slot time of pin ETXEN being deasserted.
Discarded RX Frame Register	ETH_DRFC	This 16-bit counter is incremented every time an address-recognized frame is received but cannot be copied to memory because the receive buffer is available.

Advanced Interrupt Controller (AIC)

The AT91C140 integrates the Atmel advanced interrupt controller (AIC).

The interrupt controller is connected to the fast interrupt request (nFIQ) and the standard interrupt request (NIRQ) inputs of the ARM7TDMI processor. The processor's nFIQ line can only be asserted by the external fast interrupt request input (FIQ). The nIRQ line can be asserted by the interrupts generated by the on-chip peripherals and the two external interrupt request lines, IRQ0 to IRQ1.

An 8-level priority encoder allows the user to define the priority between the different interrupt sources. Internal sources are programmed to be level-sensitive or edge-triggered. External sources can be programmed to be positive- or negative-edge triggered or high- or low-level sensitive.

Figure 21. Advanced Interrupt Controller Block Diagram

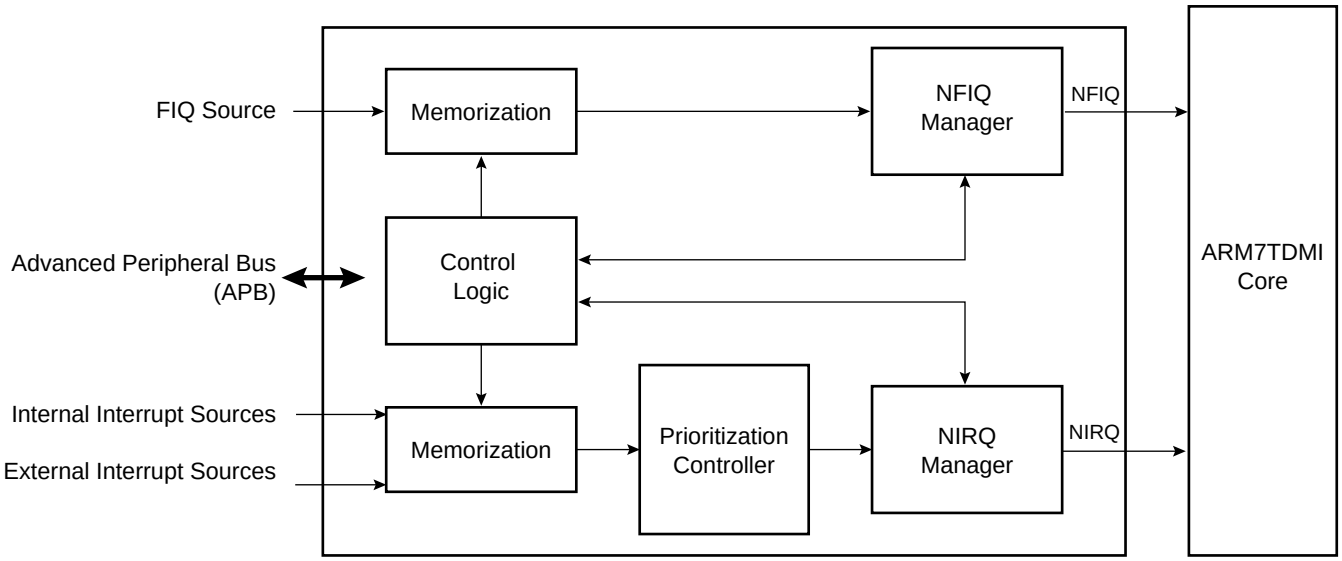


Table 23. Interrupt Sources

Interrupt Source	Interrupt Name	Interrupt Description
0	FIQ	Fast Interrupt (LOWP)
1	--	
2	SWI	Software Interrupt
3	UARTA	UART A Interrupt
4	TC0	Timer Channel 0 Interrupt
5	TC1	Timer Channel 1 Interrupt
6	TC2	Timer Channel 2 Interrupt
7	PIOA	PIO A Interrupt
8	MACA	MAC A Interrupt
9	SPI	Serial Peripheral Interface
10	IRQ0	External Interrupt

Table 23. Interrupt Sources (Continued)

Interrupt Source	Interrupt Name	Interrupt Description
11	IRQ1	External Interrupt
12	Reserved	
13	MACB	MAC B Interrupt
14	UARTB	UART B Interrupt
15	PIOB	PIO B Interrupt
16 - 31	Reserved	

Priority Controller

The nIRQ line is controlled by an 8-level priority encoder. Each source has a programmable priority level of 7 to 0. Level 7 is the highest priority and level 0 the lowest.

When the AIC receives more than one unmasked interrupt at a time, the interrupt with the highest priority is serviced first. If both interrupts have equal priority, the interrupt with the lowest interrupt source number is serviced first.

The current priority level is defined as the priority level of the current interrupt at the time the register AIC_IVR is read (the interrupt which will be serviced). In the case when a higher priority unmasked interrupt occurs while an interrupt already exists, there are two possible outcomes depending on whether the AIC_IVR has been read.

1. If the nIRQ line has been asserted but the AIC_IVR has not been read, then the processor will read the new higher priority interrupt handler number in the AIC_IVR register and the current interrupt level is updated.
2. If the processor has already read the AIC_IVR, then the nIRQ line is reasserted. When the processor has authorized nested interrupts to occur and reads the AIC_IVR again, it reads the new, higher priority interrupt handler address. At the same time the current priority value is pushed onto a first-in last-out stack and the current priority is updated to the higher priority.

When the End of Interrupt Command Register (AIC_EOICR) is written, the current interrupt level is updated with the current interrupt level from the stack (if any). Hence, at the end of a higher priority interrupt, the AIC returns to the previous state corresponding to the preceding lower priority interrupt which had been interrupted.

Interrupt Handling

The interrupt handler must read the AIC_IVR as soon as possible. This deasserts the nIRQ request to the processor and clears the interrupt in case it is programmed to be edge-triggered. This permits the AIC to assert the nIRQ line again when a higher priority unmasked interrupt occurs.

At the end of the interrupt service routine, the End of Interrupt Command Register (AIC_EOICR) must be written. This allows pending interrupts to be serviced.

Interrupt Masking

Each interrupt source, including FIQ, can be enabled or disabled using the command registers AIC_IECR and AIC_IDCR. The interrupt mask can be read in the read only register AIC_IMR. A disabled interrupt does not affect the servicing of other interrupts.

Interrupt Clearing and Setting

All interrupt sources which are programmed to be edge-triggered (including FIQ) can be individually set or cleared by respectively writing to the registers AIC_ISCR and AIC_ICCR. This function of the interrupt controller is available for auto-test or software debug purposes.

Standard Interrupt Sequence

It is assumed that:

- The advanced interrupt controller has been programmed, AIC_SVR registers are loaded with corresponding interrupt service routine addresses and interrupts are enabled.

When nIRQ is asserted and if the I bit of CPSR is 0, the sequence is as follows:

1. The CPSR is stored in SPSR_irq, the current value of the Program Counter is loaded in the IRQ link register (R14_IRQ) and the Program Counter (R15) is loaded with 0x18. In the following cycle during fetch at address 0x1C, the ARM core adjusts R14_IRQ, decrementing it by 4.
2. The ARM core enters IRQ mode if it is not already.
3. When the instruction at 0x18 is executed, the Program Counter is loaded with the value read in the AIC_IVR. Reading the AIC_IVR has the following effects:

   Sets the current interrupt to be the pending one with the highest priority. The current level is the priority level of the current interrupt.

   De-asserts the nIRQ line on the processor (even if vectoring is not used, AIC_IVR must be read in order to de-assert nIRQ).

   Automatically clears the interrupt if it has been programmed to be edge-triggered.

   Pushes the current level on to the stack.

   Returns the AIC_SVR corresponding to the current interrupt.
4. The previous step establishes a connection to the corresponding ISR. This begins by saving the link register (R14_IRQ) and the SPSR (SPSR_IRQ). Note that the link register must be decremented by 4 when it is saved if it is to be restored directly into the Program Counter at the end of the interrupt.
5. Further interrupts can then be unmasked by clearing the I bit in the CPSR, allowing re-assertion of the nIRQ to be taken into account by the core. This can occur if an interrupt with a higher priority than the current one occurs.
6. The interrupt handler then proceeds as required, saving the registers which are used and restoring them at the end. During this phase, an interrupt of priority higher than the current level will restart the sequence from step 1. Note that if the interrupt is programmed to be level-sensitive, the source of the interrupt must be cleared during this phase.
7. The I bit in the CPSR must be set in order to mask interrupts before exiting to ensure that the interrupt is completed in an orderly manner.
8. The service routine should then connect to the common exit routine.
9. The End Of Interrupt Command Register (AIC_EOICR) must be written in order to indicate to the AIC that the current interrupt is finished. This causes the current level to be popped from the stack, restoring the previous current level if one exists. If another interrupt with lower or equal priority than the old current level is pending, the nIRQ line is re-asserted but the interrupt sequence does not immediately start because the I bit is set in the core.
10. The SPSR (SPSR_IRQ) is restored. Finally, the saved value of the Link Register is restored directly into the PC. This has the effect of returning from the interrupt to the step previously executed, of loading the CPSR with the stored SPSR and of masking or unmasking the interrupts depending on the state saved in the SPSR (the previous state of the ARM core).

Note: The I bit in the SPSR is significant. If it is set, it indicates that the ARM core was just about to mask IRQ interrupts when the mask instruction was interrupted. Hence, when the SPSR is restored, the mask instruction is completed (IRQ is masked).

Fast Interrupt

The external FIQ line is the only source which can raise a fast interrupt request to the processor. Therefore it has no priority controller. It can be programmed to be positive- or negative-edge triggered or high- or low-level sensitive in the AIC_SMR0 register.

The fast interrupt handler address can be stored in the AIC_SVR0 register. The value written into this register is available by reading the AIC_FVR register when an FIQ interrupt is raised. By storing the following instruction at address 0x0000001C, the processor will load the program counter with the interrupt handler address stored in the AIC_FVR register.

LDR PC, [PC, #-&F20]

Alternatively, the interrupt handler can be stored starting from address 0x0000001C as described in the ARM7TDMI datasheet.

Fast Interrupt Sequence

It is assumed that:

- The advanced interrupt controller has been programmed, AIC_SVR[0] is loaded with the fast interrupt service routine address and the fast interrupt is enabled.
- Nested fast interrupts are not needed by the user.

When nFIQ is asserted, if the F bit of CPSR is 0, the sequence is:

1. The CPSR is stored in SPSR_fiq, the current value of the Program Counter is loaded in the FIQ link register (R14_FIQ) and the Program Counter (R15) is loaded with 0x1C. In the following cycle, during fetch at address 0x20, the ARM core adjusts R14_FIQ, decrementing it by 4.
2. The ARM core enters FIQ mode.
3. When the instruction loaded at address 0x1C is executed, the Program Counter is loaded with the value read in AIC_FVR. Reading the AIC_FVR has the effect of clearing the fast interrupt (source 0 connected to the FIQ line) if it has been programmed to be edge-triggered. In this case only, it de-asserts the nFIQ line on the processor.
4. The previous step establishes a connection to the corresponding interrupt service routine. It is not necessary to save the Link Register (R14_FIQ) and the SPSR (SPSR_FIQ) if nested fast interrupts are not needed.
5. The interrupt handler can then proceed as required. It is not necessary to save registers R8 to R13 because FIQ mode has its own dedicated registers and the user R8 to R13 are banked. The other registers, R0 to R7, must be saved before being used and restored at the end (before the next step). Note that if the fast interrupt is programmed to be level- sensitive, the source of the interrupt must be cleared during this phase in order to de-assert the nFIQ line.
6. Finally, the Link Register (R14_FIQ) is restored into the PC after decrementing it by 4 (e.g., with instruction SUB PC, LR, #4). This has the effect of returning from the interrupt to the step previously executed, of loading the CPSR with the SPSR and of masking or unmasking the fast interrupt depending on the state saved in the SPSR.

Note: The F bit in the SPSR is significant. If it is set, it indicates that the ARM core was just about to mask FIQ interrupts when the mask instruction was interrupted. Hence, when the SPSR is restored, the interrupted instruction is completed (FIQ is masked).

Software Interrupt

Any interrupt source of the AIC can be a software interrupt. It must be programmed to be edge-triggered in order to set or clear it by writing to the AIC_ISCR and AIC_ICCR. This is totally independent of the SWI instruction of the ARM7TDMI processor.

Spurious Interrupt

A spurious interrupt is a signal of very short duration on one of the interrupt input lines. A spurious interrupt also arises when an interrupt is triggered and masked in the same cycle.

Spurious Interrupt Sequence

A spurious interrupt is handled by the following sequence of actions.

1. When an interrupt is active, the AIC asserts the nIRQ (or nFIQ) line and the ARM7TDMI enters IRQ (or FIQ) mode. At this moment, if the interrupt source disappears, the nIRQ (or nFIQ) line is de-asserted but the ARM7TDMI continues with the interrupt handler.
2. If the IRQ Vector Register (AIC_IVR) is read when the nIRQ is not asserted, the AIC_IVR is read with the contents of the Spurious Interrupt Vector Register.
3. If the FIQ Vector Register (AIC_FVR) is read when the nFIQ is not asserted, the AIC_FVR is read with the contents of the Spurious Interrupt Vector Register.
4. The Spurious ISR must write an End of Interrupt command as a minimum, however, it is sufficient to write to the End of Interrupt Command Register (AIC_EOICR). Until the AIC_EOICR write is received by the interrupt controller, the nIRQ (or nFIQ) line is not re-asserted.
5. This causes the ARM7TDMI to jump into the Spurious Interrupt Routine.
6. During a spurious ISR, the AIC_ISR reads 0.

AIC User Interface

Base Address: 0xFF030000 with double mapping at address 0xFFFF F000

Table 1. AIC Register Mapping

Offset	Register	Register Name	Access	Reset State
0x000	Source Mode Register 0	AIC_SMR0	Read/Write	0
0x004	Source Mode Register 1	AIC_SMR1	Read/Write	0
–	–	–	–	–
0x07C	Source Mode Register 31	AIC_SMR31	Read/Write	0
0x080	Source Vector Register 0	AIC_SVR0	Read/Write	0
0x084	Source Vector Register 1	AIC_SVR1	Read/Write	0
–	–	–	–	–
0xFC0	Source Vector Register 31	AIC_SVR31	Read/Write	0
0x100	IRQ Vector Register	AIC_IVR	Read-only	0
0x104	FIQ Vector Register	AIC_FVR	Read-only	0
0x108	Interrupt Status Register	AIC_ISR	Read-only	0
0x10C	Interrupt Pending Register	AIC_IPR	Read-only	(1)
0x110	Interrupt Mask Register	AIC_IMR	Read-only	0
0x114	Core Interrupt Status Register	AIC_CISR	Read-only	0
0x118	Reserved	–	–	–
0x11C	Reserved	–	–	–
0x120	Interrupt Enable Command Register	AIC_IECR	Write-only	–
0x124	Interrupt Disable Command Register	AIC_IDCR	Write-only	–
0x128	Interrupt Clear Command Register	AIC_ICCR	Write-only	–
0x12C	Interrupt Set Command Register	AIC_ISCR	Write-only	–
0x130	End-of-interrupt Command Register	AIC_EOICR	Write-only	–
0x134	Spurious Interrupt Vector Register	AIC_SPU	Read/Write	0

Note: 1. The reset value of this register depends on the level of the external IRQ lines. All other sources are cleared at reset.

AIC Source Mode Register

Register Name: AIC_SMR0...AIC_SMR31
Access Type: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	SRCTYPE		–	–	PRIOR		

- **PRIOR: Priority Level**

Programs the priority level for all sources except source 0 (FIQ).
The priority level can be between 0 (lowest) and 7 (highest).
The priority level is not used for the FIQ in the SMR0.

- **SRCTYPE: Interrupt Source Type**

Programs the input to be positive- or negative-edge triggered or positive- or negative-level sensitive.
The active level or edge is not programmable for the internal sources.

SRCTYPE		Internal Sources	External Sources
0	0	Level-sensitive	Low-level sensitive
0	1	Edge-triggered	Negative-edge triggered
1	0	Level-sensitive	High-level sensitive
1	1	Edge-triggered	Positive-edge triggered

AIC Source Vector Registers

Register Name: AIC_SVR0...AIC_SVR31
Access Type: Read/Write

31	30	29	28	27	26	25	24
Vector							
23	22	21	20	19	18	17	16
Vector							
15	14	13	12	11	10	9	8
Vector							
7	6	5	4	3	2	1	0
Vector							

- **Vector**

In these registers, the user may store the addresses of the corresponding handler for each interrupt source.

AIC Interrupt Vector Registers

Register Name: AIC_IVR
Access Type: Read-only
Reset Value: 0

31	30	29	28	27	26	25	24
IRQV							

23	22	21	20	19	18	17	16
IRQV							

15	14	13	12	11	10	9	8
IRQV							

7	6	5	4	3	2	1	0
IRQV							

- **IRQV**

The IRQ Vector Register contains the vector programmed by the user in the Source Vector Register corresponding to the current interrupt. The SVR Register (1 to 31) is indexed by the current interrupt number when the IVR register is read. When there is no interrupt, the IRQ register reads 0.

AIC FIQ Vector Register

Register Name: AIC_FVR
Access Type: Read-only
Reset Value: 0

31	30	29	28	27	26	25	24
FIQV							

23	22	21	20	19	18	17	16
FIQV							

15	14	13	12	11	10	9	8
FIQV							

7	6	5	4	3	2	1	0
FIQV							

- **FIQ**

The vector register contains the vector programmed by the user in SVR Register 0 which corresponds to FIQ.

AIC Interrupt Status Register

Register Name: AIC_ISR
Access Type: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	IRQID				

- **IRQID**

The interrupt status register returns the current interrupt source register.

AIC Interrupt Pending Register
Register Name: AIC_IPR
Access Type: Read-only

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
15	14	13	12	11	10	9	8
PIOB	UARTB	MACB	0	IRQ1	IRQ0	SPI	MACA
7	6	5	4	3	2	1	0
PIOA	TC2	TC1	TC0	UARTA	SWI	0	FIQ

- **Interrupt Pending**

0 = Corresponding interrupt is not pending.
1 = Corresponding interrupt is pending.

AIC Interrupt Mask Register

Register Name: AIC_IMR
Access Type: Read-only

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
15	14	13	12	11	10	9	8
PIOB	UARTB	MACB	0	IRQ1[1]	IRQ0	SPI	MACA
7	6	5	4	3	2	1	0
PIOA	TC2	TC1	TC0	UARTA	SWI	0	FIQ

- **Interrupt Mask**

0 = Corresponding interrupt is disabled.
1 = Corresponding interrupt is enabled.

AIC Core Interrupt Status Register

Register Name: AIC_CISR
Access Type: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	NIRQ	nFIQ

- **nFIQ: nFIQ Status**

0 = nFIQ line inactive.

1 = nFIQ line active.

- **NIRQ: nIRQ Status**

0 = nIRQ line inactive.

1 = nIRQ line active.

AIC Interrupt Enable Command Register

Register Name: AIC_IECR
Access Type: Write-only

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
15	14	13	12	11	10	9	8
PIOB	UARTB	MACB	0	IRQ1	IRQ0	SPI	MACA
7	6	5	4	3	2	1	0
PIOA	TC2	TC1	TC0	UARTA	SWI	0	FIQ

- **Interrupt Enable**

0 = No effect.

1 = Enables the corresponding interrupt.

AIC Interrupt Disable Command Register

Register Name: AIC_IDCR
Access Type: Write-only

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
15	14	13	12	11	10	9	8
PIOB	UARTB	MACB	0	IRQ1	IRQ0	SPI	MACA
7	6	5	4	3	2	1	0
PIOA	TC2	TC1	TC0	UARTA	SWI	0	FIQ

- **Interrupt Disable**

0 = No effect.
1 = Disables the corresponding interrupt.

AIC Interrupt Clear Command Register

Register Name: AIC_ICCR
Access Type: Write-only

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
15	14	13	12	11	10	9	8
PIOB	UARTB	MACB	0	IRQ1	IRQ0	SPI	MACA
7	6	5	4	3	2	1	0
PIOA	TC2	TC1	TC0	UARTA	SWI	0	FIQ

- **Interrupt Clear**

0 = No effect.
1 = Clears the corresponding interrupt.

AIC Interrupt Set Command Register

Register Name: AIC_ISCR
Access Type: Write-only

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
15	14	13	12	11	10	9	8
PIOB	UARTB	MACB	0	IRQ1	IRQ0	SPI	MACA
7	6	5	4	3	2	1	0
PIOA	TC2	TC1	TC0	UARTA	SWI	0	FIQ

- **Interrupt Set**

0 = No effect.
1 = Sets the corresponding interrupt.

AIC End of Interrupt Command Register

Register Name: AIC_EOICR
Access Type: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	–

The End of Interrupt Command Register is used by the interrupt routine to indicate that the interrupt treatment is complete. Any value can be written as it is only necessary to make a write to this register location to signal the end of interrupt treatment.

AIC Spurious Interrupt Vector Register

Register Name: AIC_SPU
Access Type: Read/Write

31	30	29	28	27	26	25	24
SIQV							
23	22	21	20	19	18	17	16
SIQV							
15	14	13	12	11	10	9	8
SIQV							
7	6	5	4	3	2	1	0
SIQV							

- **SIQV**

This register contains the 32-bit address of an interrupt routine which is used to treat cases of spurious interrupts.
The programmed address is read in the AIC_IVR if it is read when the nIRQ line is not asserted.
The programmed address is read in the AIC_FVR if it is read when the nFIQ line is not asserted.

Parallel I/O Controller (PIO)

The AT91C140 integrates two PIO controllers, PIOA and PIOB. PIOA controls 32 I/O lines and PIOB controls 16 FI/O lines. Each I/O line can be programmed as an input or an output and can generate an interrupt on level change.

These pins are used for several functions:

- External I/O for Internal Peripherals
- Keypad Controller Function
- General Purpose I/O

Output Selection

The user can enable each individual I/O signal as an output with the PIO_OER and PIO_ODR registers. The output status of the I/O signals can be read in the PIO_OSR register. The direction defined has an effect only if the pin is configured to be controlled by the PIO controller.

I/O Levels

Each pin can be configured to be driven high or low. The level is defined in four different ways, according to the following conditions:

- If a pin is controlled by the PIO controller and is defined as an output, the level is programmed using the PIO_SODR and PIO_CODR registers. In this case, the programmed value can be read in the PIO_ODSR register.
- If a pin is controlled by the PIO controller and is not defined as an output, the level is determined by the external circuit.
- If a pin is not controlled by the PIO controller, the state of the pin is defined by the peripheral.

In all cases, the level on the pin can be read in the register PIO_PDSR.

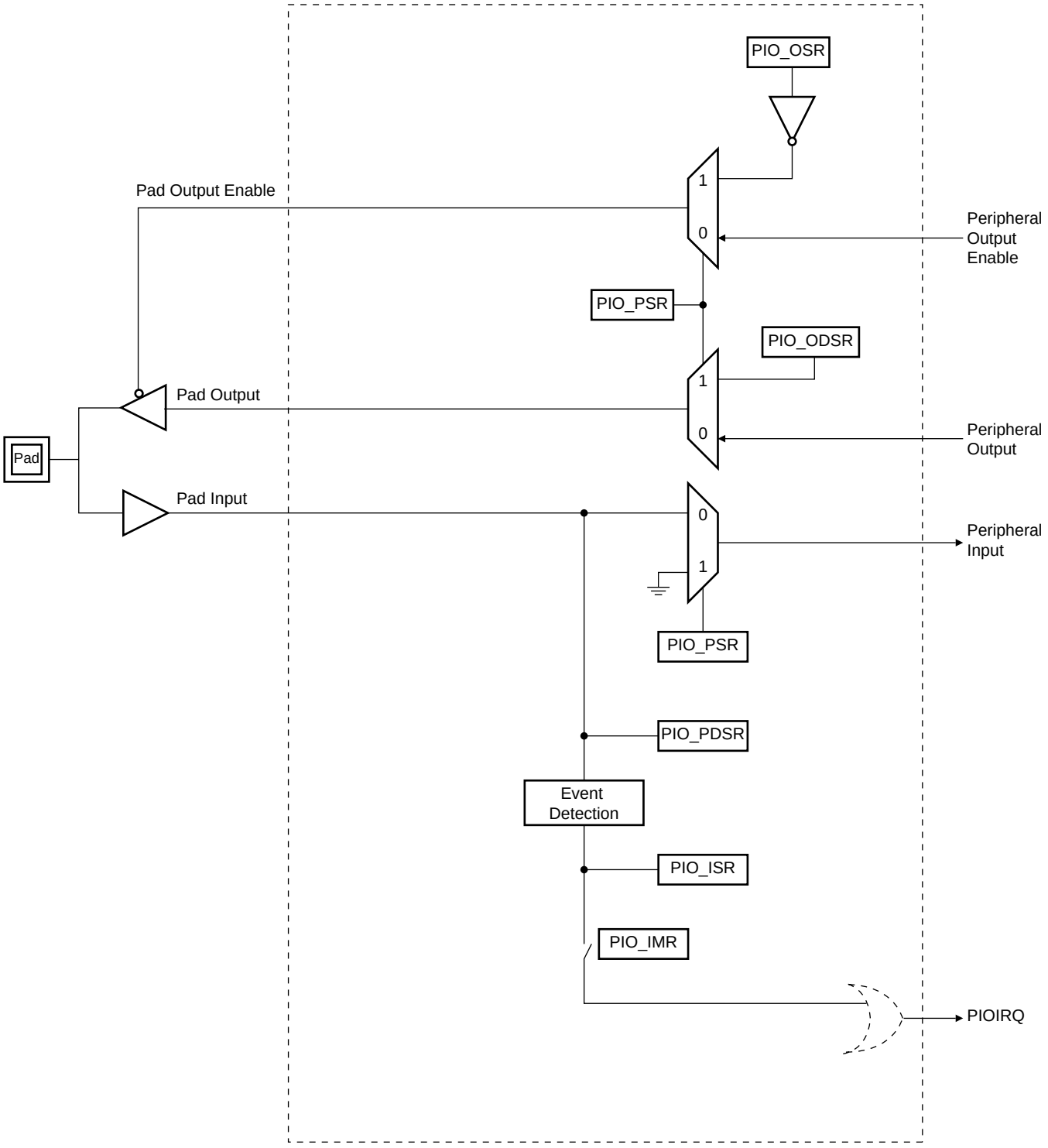
Interrupts

Each parallel I/O can be programmed to generate an interrupt when a level change occurs. This is controlled by the PIO_IER and PIO_IDR registers which enable/disable the I/O interrupt by setting/clearing the corresponding bit in PIO_IMR. When a change in level occurs, the corresponding bit in PIO_ISR is set depending on whether the pin is used as a PIO or a peripheral, and whether it is defined as input or output. If the corresponding interrupt in PIO_IMR is enabled, the PIO interrupt is asserted.

When PIO_ISR is read, the register is automatically cleared.

I/O Line Control

Figure 22. I/O Line Block Diagram



Parallel I/O Controller (PIO) User Interface

Each individual I/O is associated with a bit position in the parallel I/O user interface registers. Each of these registers is 32 bits wide. If a parallel I/O line is not defined, writing to the corresponding bits has no effect. Undefined bits read as zero.

Table 24. PIO Controller Memory Map

Offset	Register Name	Description	Access	Reset Value
0x00	PIO_PER	PIO Enable Register	Write-only	–
0x04	PIO_PDR	PIO Disable Register	Write-only	–
0x08	PIO_PSR	PIO Status Register	Read-only	–
0x0C	–	Reserved	–	–
0x10	PIO_OER	Output Enable Register	Write-only	–
0x14	PIO_ODR	Output Disable Register	Write-only	–
0x18	PIO_OSR	Output Status Register	Read-only	0x0
0x1C	–	Reserved	–	–
0x20	–	Reserved	–	–
0x24	–	Reserved	–	–
0x28	–	Reserved	–	0x0
0x2C	–	Reserved	–	–
0x30	PIO_SODR	Set Output Data Register	Write-only	–
0x34	PIO_CODR	Clear Output Data Register	Write-only	–
0x38	PIO_ODSR	Output Data Status Register	Read-only	0x0
0x3C	PIO_PDSR(1)	Pin Data Status Register	Read-only	See Note 1
0x40	PIO_IER	Interrupt Enable Register	Write-only	–
0x44	PIO_IDR	Interrupt Disable Register	Write-only	–
0x48	PIO_IMR	Interrupt Mask Register	Read-only	–
0x4C	PIO_ISR(2)	Interrupt Status Register	Read-only	See Note 2

Notes: 1. The reset value of this register depends on the level of the external pins at reset.
2. This register is cleared at reset. However, the first read of the register can give a value not equal to zero if any changes have occurred on any pins between the reset and the read.

PIO Enable Register

Register Name: PIO_PER
Access Type: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

This register is used to enable individual pins to be controlled by the PIO controller instead of the associated peripheral. When the PIO is enabled, the associated peripheral (if any) is held at logic zero.

1 = Enables the PIO to control the corresponding pin (disables peripheral control of the pin).

0 = No effect.

PIO Disable Register

Register Name: PIO_PDR
Access Type: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

This register is used to disable PIO control of individual pins. When the PIO control is disabled, the normal peripheral function is enabled on the corresponding pin.

1 = Disables PIO control (enables peripheral control) on the corresponding pin.

0 = No effect.

PIO Status Register

Register Name: PIO_PSR
Access Type: Read-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

This register indicates which pins are enabled for PIO control. This register is updated when PIO lines are enabled or disabled.

1 = PIO is active on the corresponding line (peripheral is inactive).

0 = PIO is inactive on the corresponding line (peripheral is active).

PIO Output Enable Register

Register Name: PIO_OER
Access Type: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

This register is used to enable PIO output drivers. If the pin is driven by a peripheral, there is no effect on the pin but the information is stored. The register is programmed as follows:

1 = Enables the PIO output on the corresponding pin.

0 = No effect.

PIO Output Disable Register

Register Name: PIO_ODR
Access Type: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

This register is used to disable PIO output drivers. If the pin is driven by the peripheral, there is no effect on the pin, but the information is stored. The register is programmed as follows:

1 = Disables the PIO output on the corresponding pin.

0 = No effect.

PIO Output Status Register

Register Name: PIO_OSR
Access Type: Read-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

This register shows the PIO pin control (output enable) status which is programmed in PIO_OER and PIO ODR. The defined value is effective only if the pin is controlled by the PIO. The register reads as follows:

1 = The corresponding PIO is output on this line.

0 = The corresponding PIO is input on this line.

PIO Set Output Data Register

Register Name: PIO_SODR
Access Type: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

This register is used to set PIO output data. It affects the pin only if the corresponding PIO output line is enabled and if the pin is controlled by the PIO. Otherwise, the information is stored.

1 = PIO output data on the corresponding pin is set.

0 = No effect.

PIO Clear Output Data Register

Register Name: PIO_CODR
Access Type: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

This register is used to clear PIO output data. It affects the pin only if the corresponding PIO output line is enabled and if the pin is controlled by the PIO. Otherwise, the information is stored.

1 = PIO output data on the corresponding pin is cleared.

0 = No effect.

PIO Output Data Status Register

Register Name: PIO_ODSR
Access Type: Read-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

This register shows the output data status which is programmed in PIO_SODR or PIO_CODR. The defined value is effective only if the pin is controlled by the PIO Controller and only if the pin is defined as an output.

1 = The output data for the corresponding line is programmed to 1.

0 = The output data for the corresponding line is programmed to 0.

PIO Pin Data Status Register

Register Name: PIO_PDSR
Access Type: Read-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

This register shows the state of the physical pin of the chip. The pin values are always valid, regardless of whether the pins are enabled as PIO, peripheral, input or output. The register reads as follows:

1 = The corresponding pin is at logic 1.

0 = The corresponding pin is at logic 0.

PIO Interrupt Enable Register

Register Name: PIO_IER
Access Type: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

This register is used to enable PIO interrupts on the corresponding pin. It has an effect whether PIO is enabled or not.

1 = Enables an interrupt when a change of logic level is detected on the corresponding pin.

0 = No effect.

PIO Interrupt Disable Register

Register Name: PIO_IDR
Access Type: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

This register is used to disable PIO interrupts on the corresponding pin. It has an effect whether the PIO is enabled or not.

1 = Disables the interrupt on the corresponding pin. Logic level changes are still detected.

0 = No effect.

PIO Interrupt Mask Register

Register Name: PIO_IMR
Access Type: Read-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

This register shows which pins have interrupts enabled. It is updated when interrupts are enabled or disabled by writing to PIO_IER or PIO_IDR.

1 = Interrupt is enabled on the corresponding pin.

0 = Interrupt is not enabled on the corresponding pin.

PIO Interrupt Status Register

Register Name: PIO_ISR
Access Type: Read-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

This register indicates for each pin when a logic value change has been detected (rising or falling edge). This is valid whether the PIO is selected for the pin or not and whether the pin is an input or an output.

The register is reset to zero following a read and at reset.

1 = At least one input change has been detected on the corresponding pin since the register was last read.

0 = No input change has been detected on the corresponding pin since the register was last read.

Universal Asynchronous Receiver Transmitter (UART)

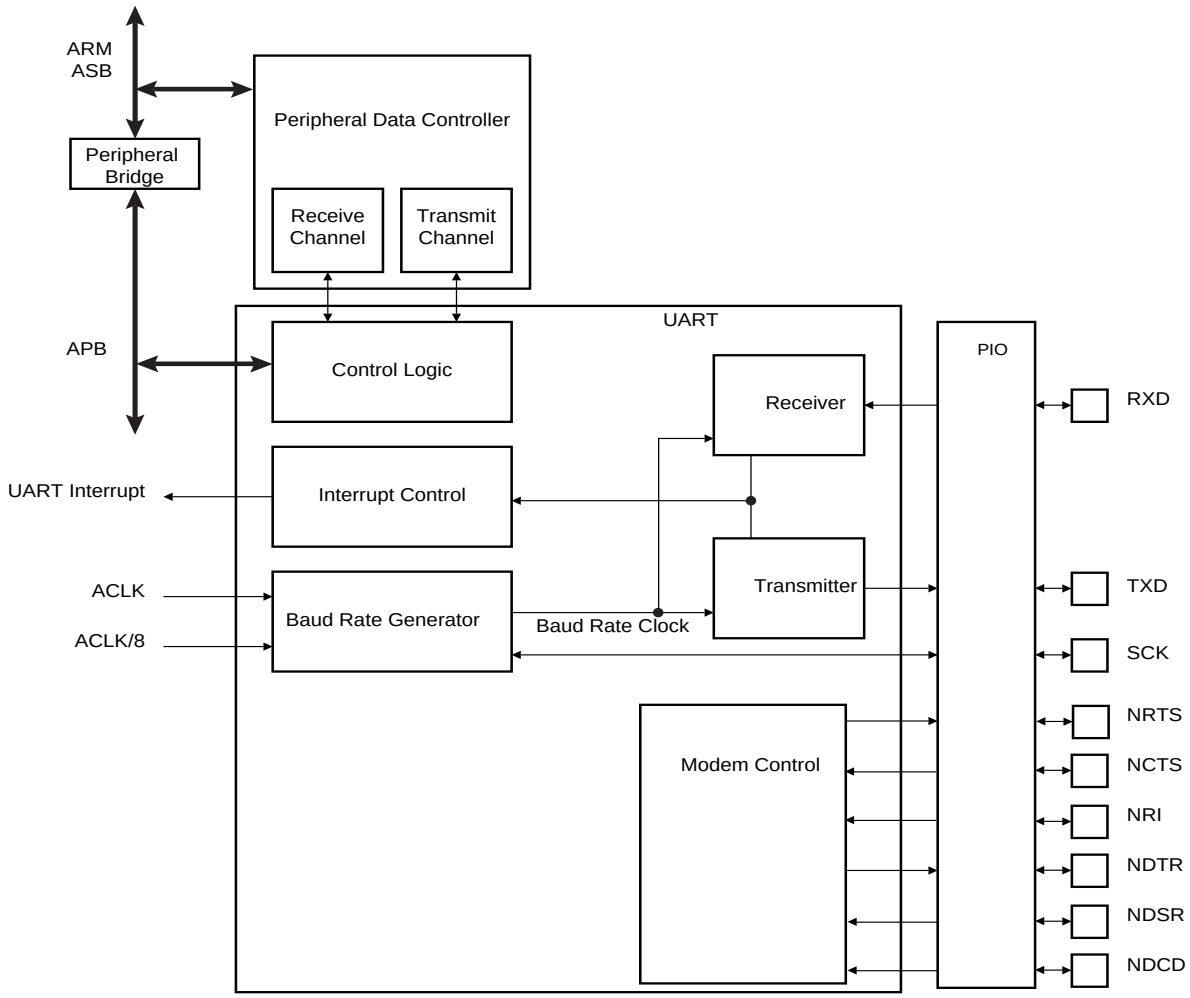
The AT91C140 provides two identical full-duplex Universal Asynchronous Receiver Transmitters, UART A and UART B. These peripherals sit on the APB bus but are also connected to the ASB bus (and hence external memory) via a dedicated PDC.

The main features are:

- Programmable Baud Rate Generator
- Parity, Framing and Overrun Error Detection
- Line Break Generation and Detection
- Automatic Echo, Local Loopback and Remote Loopback Channel Modes
- Interrupt Generation
- Two Dedicated Peripheral Data Controller Channels
- 6-, 7- and 8-bit Character Length
- Modem Control Signals

Block Diagram

Figure 23. UART Block Diagram



Pin Description

Each UART channel has external signals as defined in Table 25.

Table 25. UART Pins

Signal Name	Description	Type
SCK	UART Serial Clock. Can be configured as input or output. See US_MR	I/O
TXD	Transmit Serial Data	Output
RXD	Receive Serial Data	Input
NRTS	Request to Send	Output
NCTS	Clear to Send	Input
NDTR	Data Terminal Ready	Output
NDSR	Data Set Ready	Input
NDCD	Data Carrier Detect	Input
NRI	Ring Indicator	Input

Baud Rate Generator

The baud rate generator provides the bit period clock (the baud rate clock) to both the receiver and the transmitter.

The baud rate generator can select between external and internal clock sources. The external clock source is SCK. The internal clock sources can be either the ARM Clock (ACLK) or the ARM Clock divided by 8 (ACLK/8).

Note: In all cases, if an external clock is used, the duration of each of its levels must be longer than the system clock (ACLK) period. The external clock frequency must be at least 2.5 times lower than the system clock.

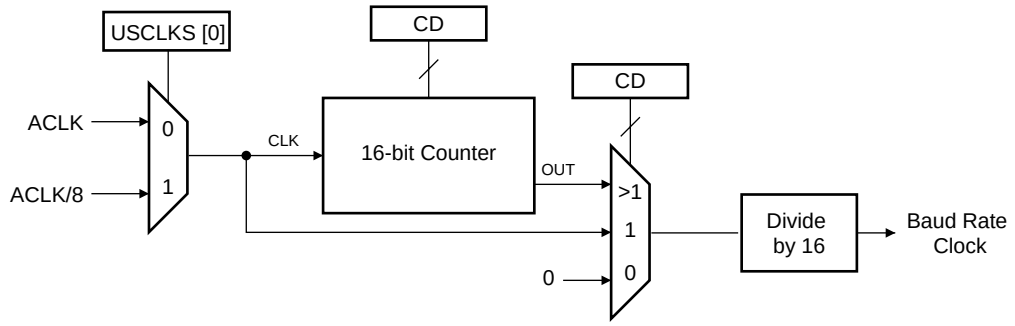
The selected clock is divided by 16 times the value (CD) written in US_BRGR (Baud Rate Generator Register). If US_BRGR is set to 0, the baud rate clock is disabled.

$$\text{Baud Rate} = \frac{\text{Selected Clock}}{16 \times \text{CD}}$$

Table 26. Clock Generator Table with Crystal Frequency of 16 MHz

Required Baud Rate (bps)	CD	Actual CD	Actual Baud Rate (bps)	Error (bps)	% Error
9600	260.42	260	9615.385	15,38	0.16
19200	130.21	130	19230.77	30.77	0.16
38400	65.10	65	38461.54	61.54	0.16
57600	43.41	43	58139.53	539.53	0.94
115200	21.70	22	113636.40	-1163.64	-1.36

Figure 24. Baud Rate Generator



Receiver Operations

The UART detects the start of a received character by sampling the RXD signal until it detects a valid start bit. A low level (space) on RXD is interpreted as a valid start bit if it is detected for more than seven cycles of the sampling clock, which is 16 times the baud rate. Hence, a space which is longer than 7/16 of the bit period is detected as a valid start bit. A space which is 7/16 of a bit period or shorter is ignored and the receiver continues to wait for a valid start bit.

When a valid start bit has been detected, the receiver samples the RXD at the theoretical mid-point of each bit. It is assumed that each bit lasts 16 cycles of the sampling clock (1-bit period) so the sampling point is eight cycles (0.5-bit periods) after the start of the bit. The first sampling point is therefore 24 cycles (1.5-bit periods) after the falling edge of the start bit was detected. Each subsequent bit is sampled 16 cycles (1-bit period) after the previous one.

Figure 25. Start Bit Detection

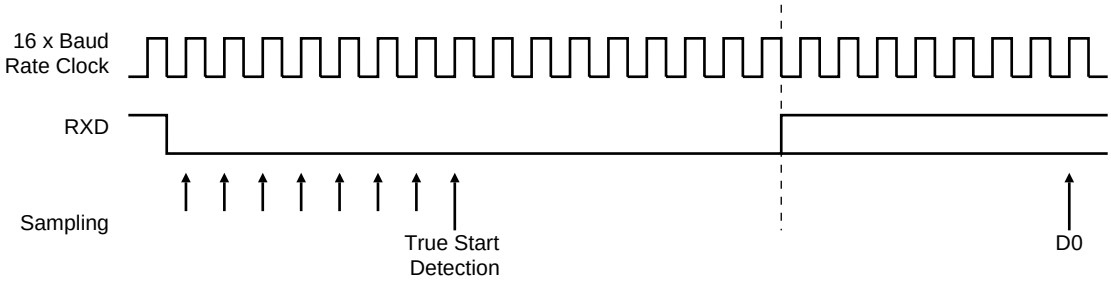
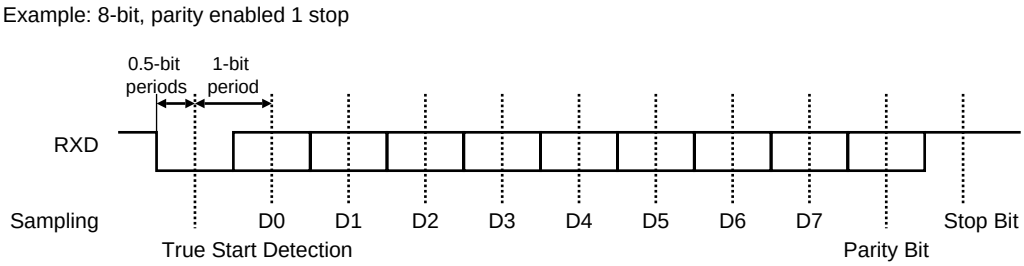


Figure 26. Character Reception



Receiver Ready

When a complete character is received, it is transferred to the US_RHR and the RXRDY status bit in US_CSR is set. If US_RHR has not been read since the last transfer, the OVRE status bit in US_CSR is set.

Parity Error

Each time a character is received, the receiver calculates the parity of the received data bits in accordance with the field PAR in US_MR. It then compares the result with the received parity bit. If different, the parity error bit PARE in US_CSR is set.

Framing Error

If a character is received with a stop bit at low level and with at least one data bit at high level, a framing error is generated. This sets FRAME in US_CSR.

Time-out

This function allows an idle condition on the RXD line to be detected. The maximum delay for which the UART should wait for a new character to arrive while the RXD line is inactive (high level) is programmed in US_RTOR. When this register is set to 0, no time-out is detected. Otherwise, the receiver waits for a first character and then initializes a counter which is decremented at each bit period and reloaded at each byte reception. When the counter reaches 0, the TIMEOUT bit in US_CSR is set. The user can restart the wait for a first character with the STTTO (Start Time-out) bit in US_CR.

Calculation of time-out duration:

$$\text{Duration} = \text{Value} \times 4 \times \text{Bit Period}$$

Transmitter

Start bit, data bits, parity bit and stop bits are serially shifted, lowest significant bit first, on the falling edge of the serial clock.

The number of data bits is selected in the CHRL field in US_MR.

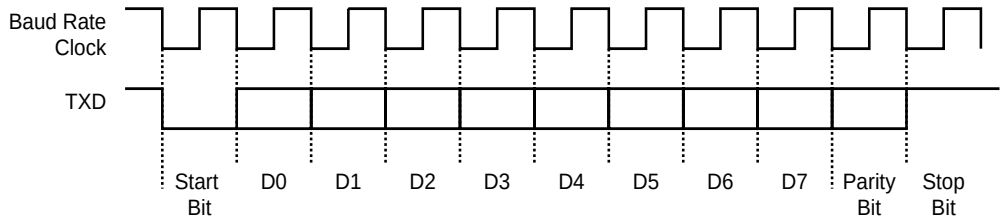
The parity bit is set according to the PAR field in US_MR.

The number of stop bits is selected in the NBSTOP field in US_MR.

When a character is written to US_THR, it is transferred to the Shift Register as soon as it is empty. When the transfer occurs, the TXRDY bit in US_CSR is set until a new character is written to US_THR. If the Transmit Shift Register and US_THR are both empty, the TXEMPTY bit in US_CSR is set.

Figure 27. Character Transmission

Example: 8-bit, parity enabled 1 stop



Time-guard

The time-guard function allows the transmitter to insert an idle state on the TXD line between two characters. The duration of the idle state is programmed in US_TTGR. When this register is set to zero, no time-guard is generated. Otherwise, the transmitter holds a high level on TXD after each transmitted byte during the number of bit periods programmed in US_TTGR.

$$\text{Idle state duration between two characters} = \text{Time-guard value} \times \text{Bit period}$$

Channel Modes

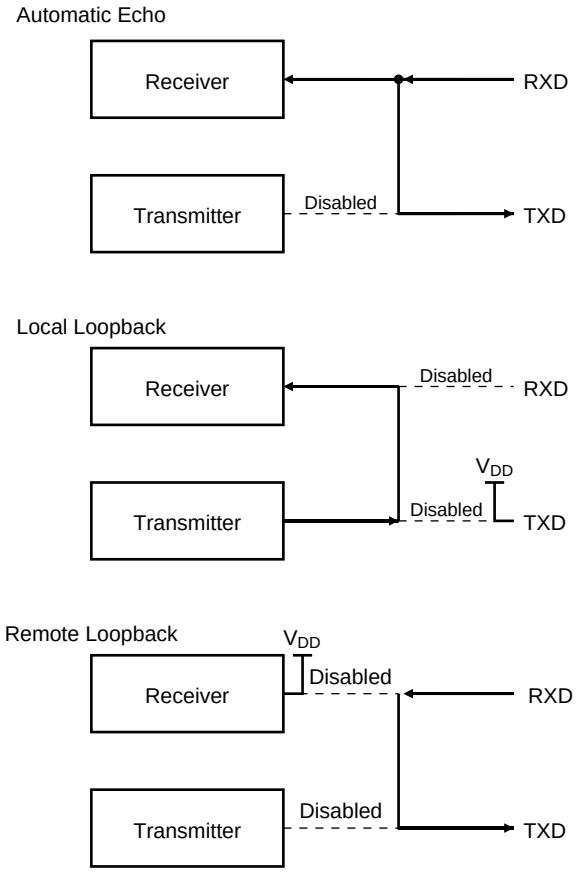
The UART can be programmed to operate in three different test modes using the field CHMODE in US_MR.

Automatic echo mode allows bit-by-bit re-transmission. When a bit is received on the RXD line, it is sent to the TXD line. Programming the transmitter has no effect.

Local loopback mode allows the transmitted characters to be received. TXD and RXD pins are not used and the output of the transmitter is internally connected to the input of the receiver. The RXD pin level has no effect and the TXD pin is held high, as in idle state.

Remote loopback mode directly connects the RXD pin to the TXD pin. The transmitter and the receiver are disabled and have no effect. This mode allows bit-by-bit re-transmission.

Figure 28. Channel Modes



Peripheral Data Controller

Each UART channel is closely connected to a corresponding peripheral data controller channel. One is dedicated to the receiver, the other is dedicated to the transmitter.

The PDC channel is programmed using US_TPR and US_TCR for the transmitter and US_RPR and US_RCR for the receiver. The status of the PDC is given in US_CSR by the ENDTX bit for the transmitter and by the ENDRX bit for the receiver.

The pointer registers US_TPR and US_RPR are used to store the address of the transmit or receive buffers. The counter registers US_TCR and US_RCR are used to store the size of these buffers.

The receiver data transfer is triggered by the RXRDY bit and the transmitter data transfer is triggered by TXRDY. When a transfer is performed, the counter is decremented and the pointer is incremented. When the counter reaches 0, the status bit is set

(ENDRX for the receiver, ENDTX for the transmitter in US_CSR) and can be programmed to generate an interrupt. Transfers are then disabled until a new non-zero counter value is programmed.

Modem Control and Status Signals

NCTS: Clear to Send

When low, this indicates that the modem or data set is ready to exchange data. The NCTS signal is a modem status input; its conditions can be tested by the CPU reading bit 4 (CTS) of the Modem Status Register. Bit 4 is the complement of the NCTS signal. Bit 0 (DCTS) of the Modem Status Register indicates whether the NCTS input has changed state since the previous read of the Modem Status Register. NCTS has no effect on the transmitter.

NDCD: Data Carrier Detect

When low, this indicates that the data carrier has been detected by the modem or data set. The NDCD signal is a modem status input; its condition can be tested by the CPU reading bit 7 (DCD) of the Modem Status Register. Bit 7 is the complement of the NDCD signal. Bit 3 (DDCD) of the Modem Status Register indicates whether the NDCD input pin has changed since the previous reading of the Modem Status Register. NDCD has no effect on the receiver.

NDSR: Data Set Ready

When low, this informs the modem or data set that the UART is ready to communicate. The NDSR signal is a modem status input; its condition can be tested by the CPU reading bit 5 (DSR) of the Modem Status Register. Bit 5 is the complement of the NDSR signal. Bit 1 (DDSR of the Modem Status Register) indicates whether the NDSR input has changed state since the previous read of the Modem Status Register.

NDTR: Data Terminal Ready

When low, this informs the modem or data set that the UART is ready to communicate. The NDTR output signal can be set to active low by programming bit 0 (DTR) of the Modem Control Register to a high level. A master reset operation sets this signal to its inactive (high) state. Loop mode operation holds this signal in its inactive state.

NRI: Ring Indicator

When low, this indicates that a telephone ringing signal has been received by the modem or data set. The NRI signal is a modem status input; its condition can be tested by the CPU reading bit 6 (RI) of the Modem Status Register. Bit 6 is the complement of the NRI signal. Bit 2 (TERI) of the Modem Status Register indicates whether the NRI input signal has changed from a low to a high state since the previous read of the Modem Status Register.

NRTS: Request to Send

When low, this informs the modem or data set that the UART is ready to exchange data. The NRTS output signal can be set to an active low by programming bit 1 (RTS) of the Modem Control Register. A master reset operation sets this signal to its inactive (high) state.

Universal Asynchronous Receiver/Transmitter (UART) User Interface

Table 27. UART Memory Map

Offset	Register Name	Description	Access	Reset Value
0x00	US_CR	Control Register	Write-only	–
0x04	US_MR	Mode Register	Read/Write	0
0x08	US_IER	Interrupt Enable Register	Write-only	–
0x0C	US_IDR	Interrupt Disable Register	Write-only	–
0x10	US_IMR	Interrupt Mask Register	Read-only	0
0x14	US_CSR	Channel Status Register	Read-only	0x18
0x18	US_RHR	Receiver Holding Register	Read-only	0
0x1C	US_THR	Transmitter Holding Register	Write-only	–
0x20	US_BRGR	Baud Rate Generator Register	Read/Write	0
0x24	US_RTOR	Receiver Time-out Register	Read/Write	0
0x28	US_TTGR	Transmitter Time-guard Register	Read/Write	0
0x2C	–	Reserved	–	–
0x30	US_RPR	Receive Pointer Register	Read/Write	0
0x34	US_RCR	Receive Counter Register	Read/Write	0
0x38	US_TPR	Transmit Pointer Register	Read/Write	0
0x3C	US_TCR	Transmit Counter Register	Read/Write	0
0x40	US_MC	Modem Control Register	Write-only	–
0x44	US_MS	Modem Status Register	Read-only	

UART Control Register

Name: US_CR

Access Type: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	RSTSTA
7	6	5	4	3	2	1	0
TXDIS	TXEN	RXDIS	RXEN	RSTTX	RSTRX	–	–

- **RSTRX: Reset Receiver**

0 = No effect.

1 = The receiver logic is reset.

- **RSTTX: Reset Transmitter**

0 = No effect.

1 = The transmitter logic is reset.

- **RXEN: Receiver Enable**

0 = No effect.

1 = The receiver is enabled if RXDIS is 0.

- **RXDIS: Receiver Disable**

0 = No effect.

1 = The receiver is disabled.

- **TXEN: Transmitter Enable**

0 = No effect.

1 = The transmitter is enabled if TXDIS is 0.

- **TXDIS: Transmitter Disable**

0 = No effect.

1 = The transmitter is disabled.

- **RSTSTA: Reset Status Bits**

0 = No effect.

1 = Resets the status bits PARE, FRAME, OVRE and RXBRK in the US_CSR.

UART Mode Register

Name: US_MR
Access Type: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
CHMODE		NBSTOP		PAR			–
7	6	5	4	3	2	1	0
CHRL		USCLKS		–	–	–	–

- **USCLKS: Clock Selection**

USCLKS		Selected Clock
0	0	ACLK
0	1	ACLK/8
1	X	External (SCK)

- **CHRL: Character Length**

CHRL		Character Length
0	0	Five bits
0	1	Six bits
1	0	Seven bits
1	1	Eight bits

- **PAR: Parity Type**

PAR			Parity Type
0	0	0	Even parity
0	0	1	Odd parity
0	1	0	Parity forced to 0 (space)
0	1	1	Parity forced to 1 (mark)
1	0	x	No parity

- **NBSTOP: Number of Stop Bits**

NBSTOP		
0	0	1 stop bit
0	1	1.5 stop bits
1	0	2 stop bits
1	1	Reserved

- **CHMODE: Channel Mode**

CHMODE		Mode Description
0	0	Normal Mode The UART channel operates as an Rx/Tx UART.
0	1	Automatic Echo Receiver data input is connected to TXD pin.
1	0	Local Loopback Transmitter output signal is connected to receiver input signal.
1	1	Remote Loopback RXD pin is internally connected to TXD pin.

UART Interrupt Enable Register

Name: US_IER

Access Type: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	DMSI	TXEMPTY	–
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	ENDTX	ENDRX	–	TXRDY	RXRDY

- **RXRDY: Enable RXRDY Interrupt**
- **TXRDY: Enable TXRDY Interrupt**
- **ENDRX: Enable End of Receive Transfer Interrupt**
- **ENDTX: Enable End of Transmit Transfer Interrupt**
- **OVRE: Enable Overrun Error Interrupt**
- **FRAME: Enable Framing Error Interrupt**
- **PARE: Enable Parity Error Interrupt**
- **TXEMPTY: Enable TXEMPTY Interrupt**
- **DMSI: Delta Modem Interrupt**

0 = No effect.

1 = Enables the corresponding interrupt.

UART Interrupt Disable Register

Name: US_IDR

Access Type: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	DMSI	TXEMPTY	–
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	ENDTX	ENDRX	–	TXRDY	RXRDY

- **RXRDY: Disable RXRDY Interrupt**
- **TXRDY: Disable TXRDY Interrupt**
- **ENDRX: Disable End of Receive Transfer Interrupt**
- **ENDTX: Disable End of Transmit Transfer Interrupt**
- **OVRE: Disable Overrun Error Interrupt**
- **FRAME: Disable Framing Error Interrupt**
- **PARE: Disable Parity Error Interrupt**
- **TXEMPTY: Disable TXEMPTY Interrupt**
- **DMSI: Disable Delta Modem Interrupt**

0 = No effect.

1 = Disables the corresponding interrupt.

UART Interrupt Mask Register

Name: US_IMR
Access Type: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	DMSI	TXEMPTY	–
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	ENDTX	ENDRX	RXBRK	–	RXRDY

- **RXRDY: RXRDY Interrupt Mask**
- **TXRDY: TXRDY Interrupt Mask**
- **ENDRX: End of Receive Transfer Interrupt Mask**
- **ENDTX: End of Transmit Transfer Interrupt Mask**
- **OVRE: Overrun Error Interrupt Mask**
- **FRAME: Framing Error Interrupt Mask**
- **PARE: Parity Error Interrupt Mask**
- **TXEMPTY: TXEMPTY Interrupt Mask**
- **DMSI: Delta Modem Status Indication Interrupt Mask**

0 = The corresponding interrupt is disabled.

1 = The corresponding interrupt is enabled.

UART Channel Status Register

Name: US_CSR
Access Type: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	DMSI	TXEMPTY	–
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	ENDTX	ENDRX	–	TXRDY	RXRDY

- **RXRDY: Receiver Ready**

0 = No complete character has been received since the last read of the US_RHR or the receiver is disabled.

1 = At least one complete character has been received and the US_RHR has not yet been read.

- **TXRDY: Transmitter Ready**

0 = US_THR contains a character waiting to be transferred to the Transmit Shift Register.

1 = US_THR is empty and there is no break request pending TSR availability.

Equal to zero when the UART is disabled or at reset. Transmitter enable command (in US_CR) sets this bit to one.

- **ENDRX: End-of-receive Transfer**

0 = The end-of-transfer signal from the PDC channel dedicated to the receiver is inactive.

1 = The end-of-transfer signal from the PDC channel dedicated to the receiver is active.

- **ENDTX: End-of-transmit Transfer**

0 = The end-of-transfer signal from the PDC channel dedicated to the transmitter is inactive.

1 = The end-of-transfer signal from the PDC channel dedicated to the transmitter is active.

- **OVRE: Overrun Error**

0 = No byte has been transferred from the Receive Shift Register to the US_RHR when RxRDY was asserted since the last reset status bits command.

1 = At least one byte has been transferred from the Receive Shift Register to the US_RHR when RxRDY was asserted since the last reset status bits command.

- **FRAME: Framing Error**

0 = No stop bit has been detected low since the last reset status bits command.

1 = At least one stop bit has been detected low since the last reset status bits command.

- **PARE: Parity Error**

1 = At least one parity bit has been detected false (or a parity bit high in multi-drop mode) since the last reset status bit" command.

0 = No parity bit has been detected false (or a parity bit high in multi-drop mode) since the last reset status bits command.

- **TXEMPTY: Transmitter Empty**

0 = There are characters in either US_THR or the Transmit Shift Register or a break is being transmitted.

1 = There are no characters in US_THR and the Transmit Shift Register and break is not active.

Equal to zero when the UART is disabled or at reset. Transmitter enable command (in US_CR) sets this bit to one.

- **DMSI: Delta Modem Status Indication Interrupt**

0 = No effect.

1 = There has been a change in the modem status delta bits since the last reset status bits command.

UART Receiver Holding Register

Name: US_RHR
Access Type: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
RXCHR							

- **RXCHR: Received Character**

Last character received if RXRDY is set. When number of data bits is less than eight, the bits are right-aligned.

UART Transmitter Holding Register

Name: US_THR
Access Type: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
TXCHR							

- **TXCHR: Character to be Transmitted**

Next character to be transmitted after the current character if TXRDY is not set. When number of data bits is less than eight, the bits are right-aligned.

UART Baud Rate Generator Register

Name: US_BRGR
Access Type: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
CD							
7	6	5	4	3	2	1	0
CD							

- **CD: Clock Divisor**

This register has no effect if synchronous mode is selected with an external clock.

CD	Effect
0	Disables clock
1	Clock divisor bypass
2 to 65535	Baud rate = Selected clock/(16 x CD)

UART Receive Pointer Register

Name: US_RPR
Access Type: Read/Write

31	30	29	28	27	26	25	24
RXPTR							
23	22	21	20	19	18	17	16
RXPTR							
15	14	13	12	11	10	9	8
RXPTR							
7	6	5	4	3	2	1	0
RXPTR							

- **RXPTR: Receive Pointer**

RXPTR must be loaded with the address of the receive buffer.

UART Receive Counter Register

Name: US_RCR
Access Type: Read/Write
Reset Value: 0x0

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
RXCTR							
7	6	5	4	3	2	1	0
RXCTR							

- **RXCTR: Receive Counter**

RXCTR must be loaded with the size of the receive buffer.

0 =Stop peripheral data transfer dedicated to the receiver.

1 - 65535: Start peripheral data transfer if RXRDY is active.

UART Transmit Pointer Register

Name: US_TPR
Access Type: Read/Write
Reset Value: 0x0

31	30	29	28	27	26	25	24
TXPTR							
23	22	21	20	19	18	17	16
TXPTR							
15	14	13	12	11	10	9	8
TXPTR							
7	6	5	4	3	2	1	0
TXPTR							

- **TXPTR: Transmit Pointer**

TXPTR must be loaded with the address of the transmit buffer.

UART Transmit Counter Register

Name: US_TCR
Access Type: Read/Write
Reset Value: 0x0

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
TXCTR							
7	6	5	4	3	2	1	0
TXCTR							

- **TXCTR: Transmit Counter**

TXCTR must be loaded with the size of the transmit buffer.

0 =Stop peripheral data transfer dedicated to the transmitter.

1 - 65535: Start peripheral data transfer if TXRDY is active.

Modem Control Register

Register Name: US_MC
Access Type: Write-only
Reset Value: Undefined

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	RTS	DTR

This register controls the interface with the modem or data set (or a peripheral device emulating a modem). The contents of the Control Register are indicated below.

- **DTR: Data Terminal Ready**

This bit controls the NDTR output. When bit 0 is set to a logic 1, the NDTR output is forced to a logic 0.

When bit 0 is reset to a logic 0, the NDTR output is forced to a logic 1.

The NDTR output of the UART can be applied to an EIA inverting line driver to obtain proper polarity input at the succeeding modem or data set.

- **RTS: Request to Send**

This bit controls the NRTS output. Bit 1 affects the NRTS output in a manner identical to that described above for bit 0.

Modem Status Register

Register Name: US_MS
Access Type: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
DCD	RI	DSR	CTS	DDCD	TERI	DDSR	DCTS

This register provides the current state of the control lines from the modem (or peripheral device) to the CPU. In addition to this current-state information, four bits of the Modem Status Register provide change information. These bits are set to a logic 1 whenever a control input from the modem changes state. They are reset to logic 0 whenever the CPU reads the Modem Status Register.

- **DCTS: Delta Clear to Send**

Bit 0 indicates that the NCTS input to the chip has changed state since the last time it was read by the CPU.

- **DDSR: Delta Data Set Ready**

Bit 1 indicates that the NDSR input to the chip has changed state since the last time it was read by the CPU.

- **TERI: Trailing Edge Ring Indicator**

Bit 2 indicates that the NRI input to the chip has changed from a low to a high state.

- **DDCD: Delta Data Carrier Detect**

Bit 3 indicates that the NDCD input has changed state.

Note that whenever bit 0, 1, 2, or 3 is set to logic 1, a modem status interrupt is generated. This is reflected in the modem status register.

- **CTS: Clear to Send**

This bit is the complement of the Clear to Send (NCTS) input.

- **DSR: Data Set Ready**

This bit is the complement of the Data Set Ready (NDSR) input.

- **RI: Ring Indicator**

This bit is the complement of the Ring Indicator (NRI) input.

- **DCD: Data Carrier Detect**

This bit is the complement of the Data Carrier Detect (NDCD) input.

Timer/Counter (TC)

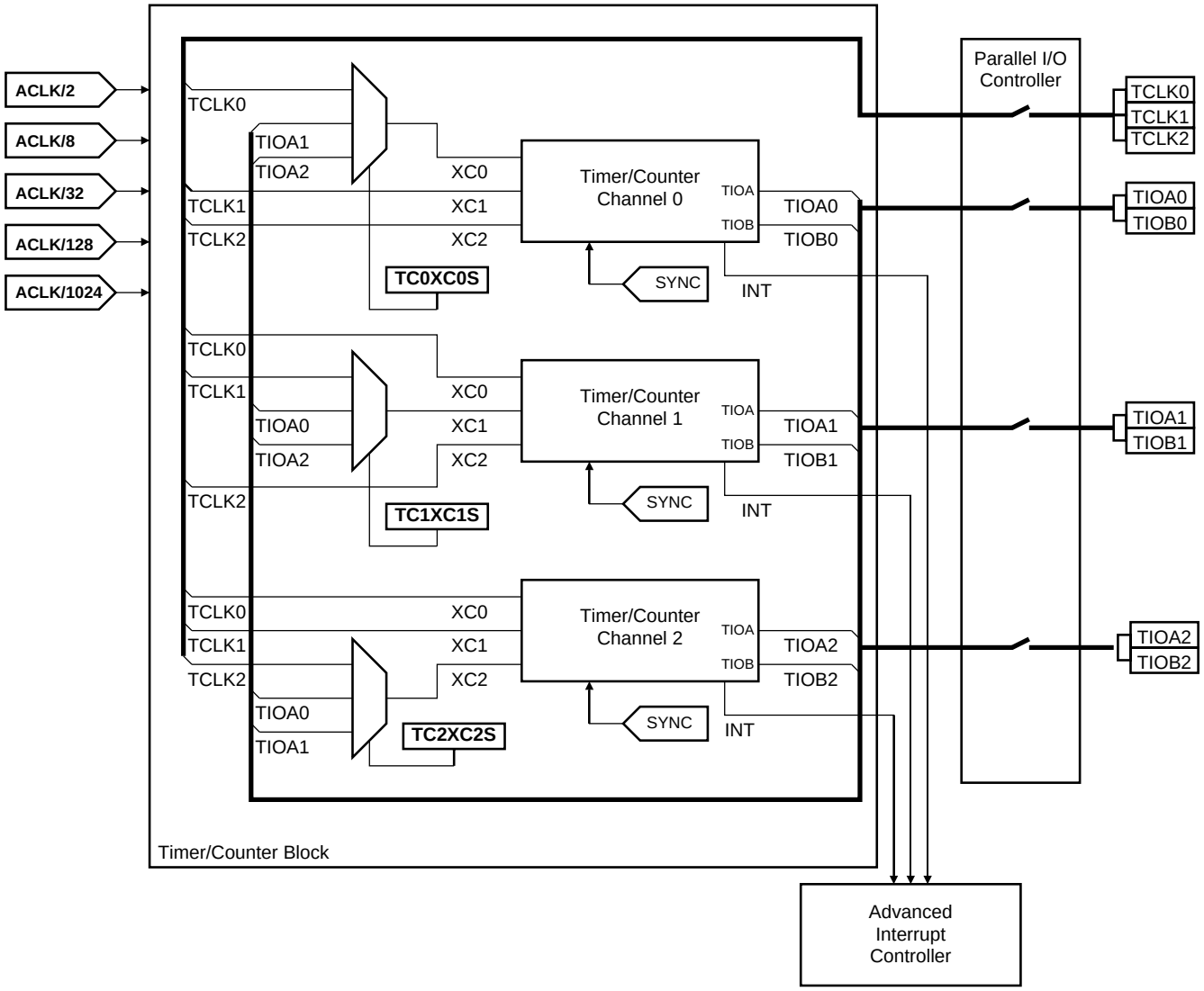
The AT91C140 features a timer/counter block that includes three identical 16-bit timer/counter channels. Each channel can be independently programmed to perform a wide range of functions including frequency measurement, event counting, interval measurement, pulse generation, delay timing and pulse-width modulation.

Each timer/counter channel has three external clock inputs, five internal clock inputs, and two multi-purpose input/output signals that can be configured by the user. Each channel drives an internal interrupt signal that can be programmed to generate processor interrupts via the AIC.

The timer/counter block has two global registers which act upon all three TC channels. The Block Control Register allows the three channels to be started simultaneously with the same instruction. The Block Mode Register defines the external clock inputs for each timer/counter channel, allowing them to be chained.

Block Diagram

Figure 29. Timer/Counter Block Diagram



Signal Name Description

Table 28. Timer Counter Signal Description

Channel Signal	Description	Type
XC0, XC1, XC2	External clock inputs	I
TIOA	Capture mode: General-purpose input Waveform mode: General-purpose output	I O
TIOB	Capture mode: General-purpose input Waveform mode: General-purpose input/output	I O
INT	Interrupt signal output	O
SYNC	Synchronization input signal	I

Table 28. Timer Counter Signal Description

Block Signal		
TCLK0, TCLK1, TCLK2	External clock inputs	I
TIOA0	TIOA signal for Channel 0	I/O
TIOB0	TIOB signal for Channel 0	I/O
TIOA1	TIOA signal for Channel 1	I/O
TIOB1	TIOB signal for Channel 1	I/O
TIOA2	TIOA signal for Channel 2	I/O
TIOB2	TIOB signal for Channel 2	I/O

Note: After a hardware reset, the timer/counter block pins are controlled by the PIO controller. They must be configured to be controlled by the peripheral before being used.

Description

The three timer/counter channels are independent and identical in operation.

Counter

Each timer/counter channel is organized around a 16-bit counter. The value of the counter is incremented at each positive edge of the selected clock. When the counter has reached the value 0xFFFF and passes to 0x0000, an overflow occurs and the COVFS bit in TC_SR (Status Register) is set.

The current value of the counter is accessible in real time by reading TC_CV. The counter can be reset by a trigger. In this case, the counter value passes to 0x0000 on the next valid edge of the selected clock.

Clock Selection

At block level, input clock signals of each channel can either be connected to the external inputs TCLK0, TCLK1 or TCLK2, or be connected to the configurable I/O signals TIOA0, TIOA1 or TIOA2 for chaining by programming the TC_BMR (Block Mode).

Each channel can independently select an internal or external clock source for its counter:

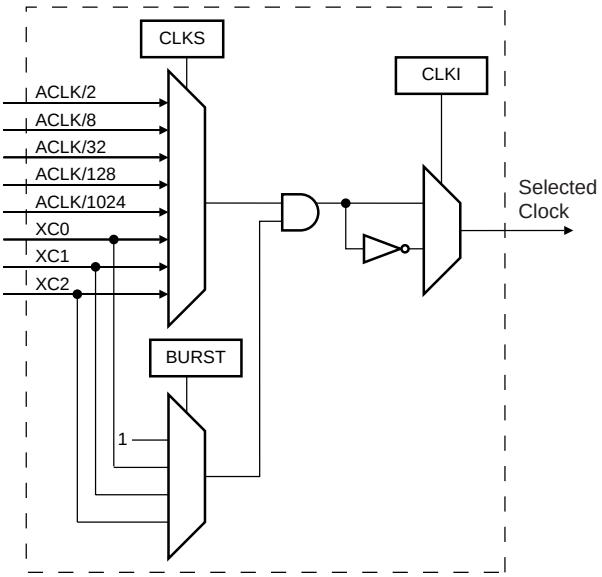
- Internal clock signals: ACLK/2, ACLK/8, ACLK/32, ACLK/128, ACLK/1024
- External clock signals: XC0, XC1 or XC2

The selected clock can be inverted with the CLKI bit in TC_CMR (Channel Mode). This allows counting on the opposite edges of the clock.

The burst function allows the clock to be validated when an external signal is high. The BURST parameter in the Mode Register defines this signal (none, XC0, XC1, XC2).

Note: In all cases, if an external clock is used, the duration of each of its levels must be longer than the system clock (ACLK) period. The external clock frequency must be at least 2.5 times lower than the system clock (ACLK).

Figure 30. Clock Selection



Clock Control

The clock of each counter can be controlled in two different ways: it can be enabled/disabled and started/stopped.

1. The clock can be enabled or disabled by the user with the CLKEN and the CLKDIS commands in the Control Register. In capture mode, it can be disabled by an RB load event if LDBDIS is set to 1 in TC_CMR. In waveform mode, it can be disabled by an RC Compare event if CPCDIS is set to 1 in TC_CMR. When disabled, the start or the stop actions have no effect: only a CLKEN command in the Control Register can re-enable the clock. When the clock is enabled, the CLKSTA bit is set in the Status Register.
2. The clock can also be started or stopped: a trigger (software, synchro, external or compare) always starts the clock. The clock can be stopped by an RB load event in capture mode (LDBSTOP = 1 in TC_CMR) or a RC compare event in waveform mode (CPCSTOP = 1 in TC_CMR). The start and the stop commands have an effect only if the clock is enabled.

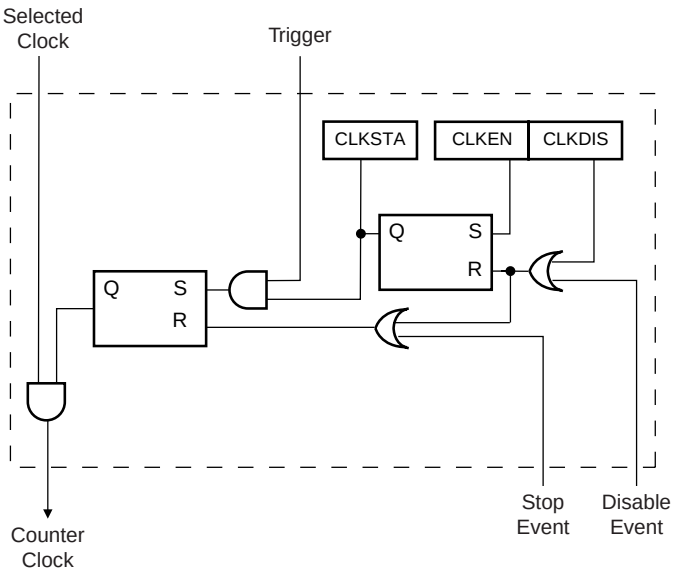
Timer/Counter Operating Modes

Each timer/counter channel can operate independently in two different modes:

1. Capture mode allows measurement on signals
2. Waveform mode allows wave generation

The timer/counter operating mode is programmed with the WAVE bit in the TC Mode Register. In capture mode, TIOA and TIOB are configured as inputs. In waveform mode, TIOA is always configured to be an output and TIOB is an output if it is not selected to be the external trigger.

Figure 31. Clock Control



Trigger

A trigger resets the counter and starts the counter clock. Three types of triggers are common to both modes, and a fourth external trigger is available to each mode.

The following triggers are common to both modes:

1. Software trigger: Each channel has a software trigger, available by setting SWTRG in TC_CCR.
2. SYNC: Each channel has a synchronization signal, SYNC. When asserted, this signal has the same effect as a software trigger. The SYNC signals of all channels are asserted simultaneously by writing TC_BCR (Block Control) with SYNC set.
3. Compare RC trigger: RC is implemented in each channel and can provide a trigger when the counter value matches the RC value if CPCTRG is set in TC_CMR.

The timer/counter channel can also be configured to have an external trigger. In capture mode, the external trigger signal can be selected between TIOA and TIOB. In waveform mode, an external event can be programmed on one of the following signals: TIOB, XC0, XC1 or XC2. This external event can then be programmed to perform a trigger by setting ENETRG in TC_CMR.

If an external trigger is used, the duration of the pulses must be longer than the system clock (ACLK) period in order to be detected.

Whatever the trigger used, it will be taken into account at the following active edge of the selected clock. This means that the counter value may not read zero just after a trigger, especially when a low-frequency signal is selected as the clock.

Capture Operating Mode

This mode is entered by clearing the WAVE parameter in TC_CMR (Channel Mode Register). Capture mode allows the TC Channel to perform measurements such as pulse timing, frequency, period, duty cycle and phase on TIOA and TIOB signals which are inputs.

Figure 32 shows the configuration of the TC Channel when programmed in capture mode.

Capture Registers A and B (RA and RB)

Registers A and B are used as capture registers; thus, they can be loaded with the counter value when a programmable event occurs on the TIOA signal.

The parameter LDRA in TC_CMR defines the TIOA edge for the loading of Register A, and the parameter LDRB defines the TIOA edge for the loading of Register B.

RA is loaded only if it has not been loaded since the last trigger or if RB has been loaded since the last loading of RA.

RB is loaded only if RA has been loaded since the last trigger or the last loading of RB.

Loading RA or RB before the read of the last value loaded sets the Overrun Error Flag (LOVRS) in TC_SR (Status Register). In this case, the old value is overwritten.

Trigger Conditions

In addition to the SYNC signal, the software trigger and the RC compare trigger, an external trigger can be defined.

Bit ABETRG in TC_CMR selects input signal TIOA or TIOB as an external trigger. Parameter ETRGEDG defines the edge (rising, falling or both) detected to generate an external trigger. If ETRGEDG = 0 (none), the external trigger is disabled.

Status Register

The following bits in the status register are significant in capture operating mode.

- CPCS: RC Compare Status

  There has been an RC Compare match at least once since the last read of the status.
- COVFS: Counter Overflow Status

  The counter has attempted to count past $FFFF since the last read of the status.
- LOVRS: Load Overrun Status

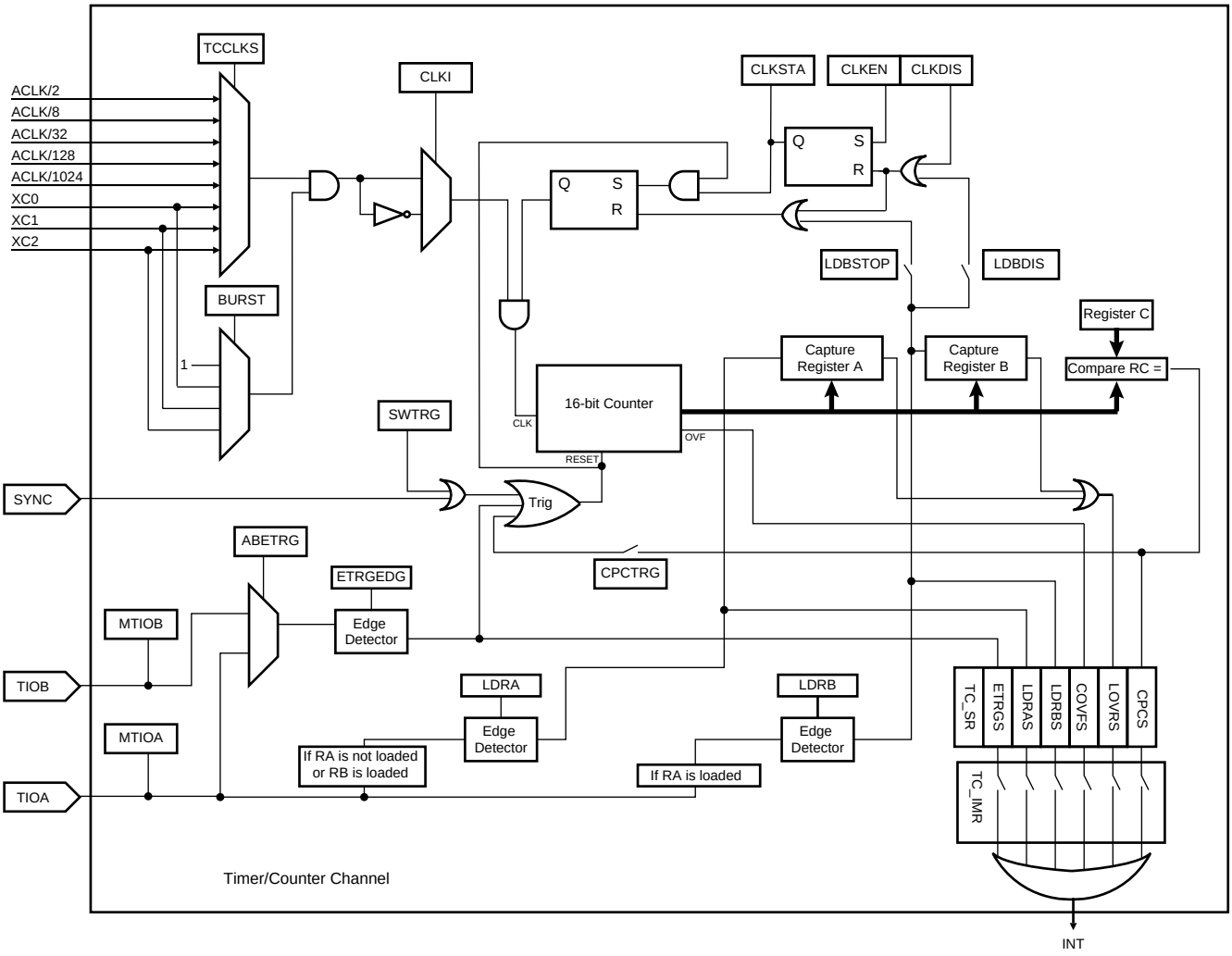
  RA or RB has been loaded at least twice without any read of the corresponding register since the last read of the status.
- LDRAS: Load RA Status

  RA has been loaded at least once without any read since the last read of the status.
- LDRBS: Load RB Status

  RB has been loaded at least once without any read since the last read of the status.
- ETRGS: External Trigger Status

  An external trigger on TIOA or TIOB has been detected since the last read of the status.

Figure 32. Capture Mode



Waveform Operating Mode

This mode is entered by setting the WAVE parameter in TC_CMR (Channel Mode Register).

Waveform operating mode allows the TC channel to generate 1 or 2 PWM signals with the same frequency and independently programmable duty cycles, or to generate different types of one-shot or repetitive pulses.

In this mode, TIOA is configured as output and TIOB is defined as output if it is not used as an external event (EEVT parameter in TC_CMR).

Figure 33 on page 123 shows the configuration of the TC channel when programmed in waveform operating mode.

Compare Register A, B and C (RA, RB, and RC)

In waveform operating mode, RA, RB and RC are all used as compare registers.

RA Compare is used to control the TIOA output. RB Compare is used to control the TIOB (if configured as output). RC Compare can be programmed to control TIOA and/or TIOB outputs.

RC Compare can also stop the counter clock (CPCSTOP = 1 in TC_CMR) and/or disable the counter clock (CPCDIS = 1 in TC_CMR).

As in capture mode, RC Compare can also generate a trigger if CPCTRG = 1. A trigger resets the counter so RC can control the period of PWM waveforms.

External Event/Trigger Conditions

An external event can be programmed to be detected on one of the clock sources (XC0, XC1, XC2) or TIOB. The external event selected can then be used as a trigger.

The parameter EEVT in TC_CMR selects the external trigger. The parameter EEVTEDG defines the trigger edge for each of the possible external triggers (rising, falling or both). If EEVTEDG is cleared (none), no external event is defined.

If TIOB is defined as an external event signal (EEVT = 0), TIOB is no longer used as output and the TC channel can only generate a waveform on TIOA.

When an external event is defined, it can be used as a trigger by setting bit ENETRG in TC_CMR.

As in capture mode, the SYNC signal, the software trigger and the RC compare trigger are also available as triggers.

Output Controller

The output controller defines the output level changes on TIOA and TIOB following an event. TIOB control is used only if TIOB is defined as output (not as an external event).

The following events control TIOA and TIOB: software trigger, external event and RC compare. RA compare controls TIOA and RB compare controls TIOB. Each of these events can be programmed to set, clear or toggle the output as defined in the corresponding parameter in TC_CMR.

Table 29 and Table 30 show which parameter in TC_CMR is used to define the effect of each event.

Table 29. TIOA Events

Parameter	TIOA Event
ASWTRG	Software trigger
AEEVT	External event
ACPC	RC compare
ACPA	RA compare

Table 30. TIOB Events

Parameter	TIOB Event
BSWTRG	Software trigger
BEEVT	External event
BCPC	RC compare
BCPB	RB compare

If two or more events occur at the same time, the priority level is defined as follows:

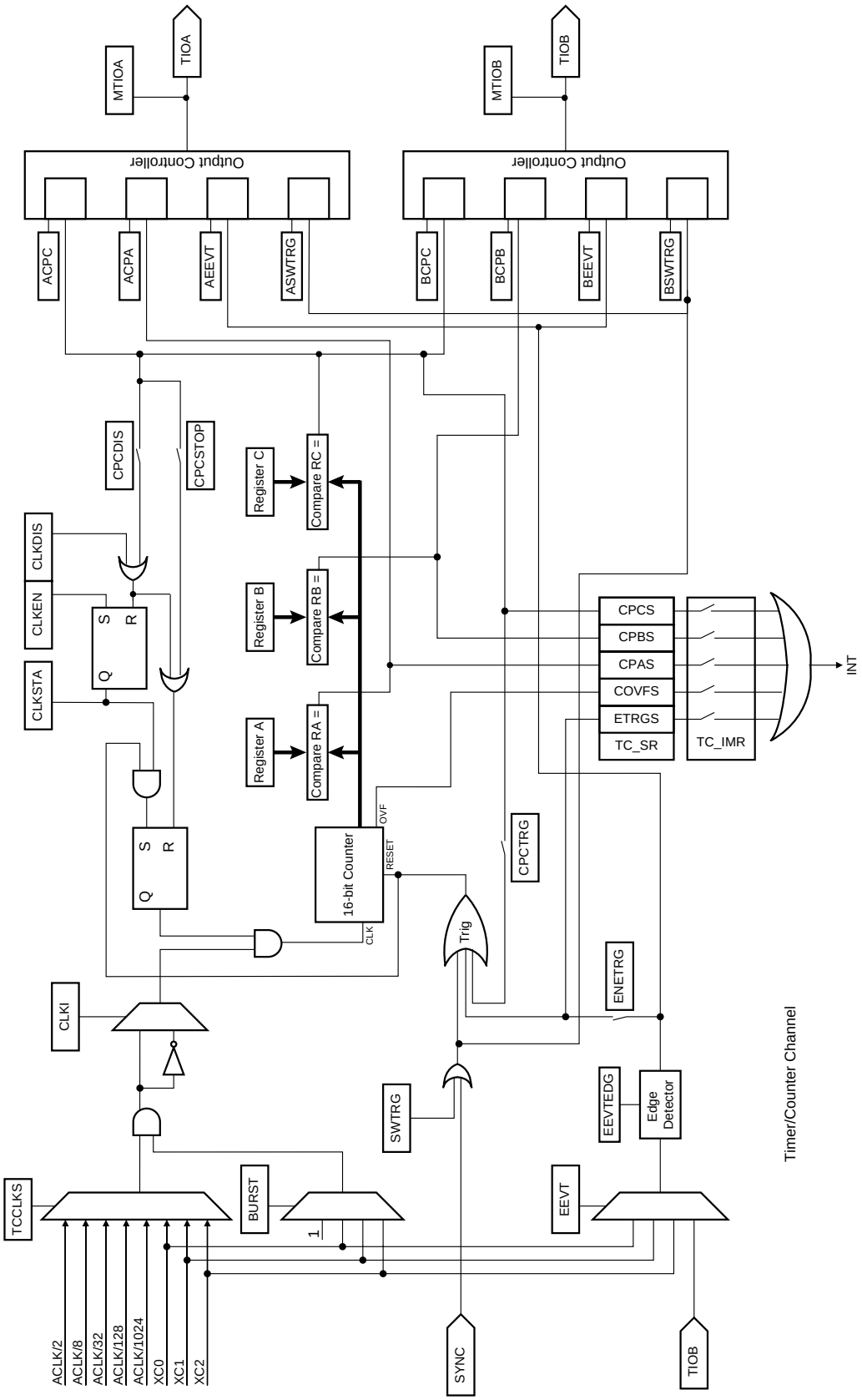
1. Software trigger
2. External event
3. RC compare
4. RA or RB compare

Status

The following bits in the status register are significant in waveform mode:

- CPAS: RA Compare Status
  There has been a RA Compare match at least once since the last read of the status
- CPBS: RB Compare Status
  There has been a RB Compare match at least once since the last read of the status
- CPCS: RC Compare Status
  There has been a RC Compare match at least once since the last read of the status
- COVFS: Counter Overflow
  Counter has attempted to count past $FFFF since the last read of the status
- ETRGS: External Trigger
  External trigger has been detected since the last read of the status

Figure 33. Waveform Mode



Timer/Counter (TC) User Interface

Table 31. TC Global Memory Map

Offset	Register Name	Channel/Register	Access	Reset Value
0x00		TC Channel 0	See Table 32	
0x40		TC Channel 1	See Table 32	
0x80		TC Channel 2	See Table 32	
0xC0	TC_BCR	TC Block Control Register	Write-only	–
0xC4	TC_BMR	TC Block Mode Register	Read/Write	0

TC_BCR (Block Control Register) and TC_BMR (Block Mode Register) control the whole TC block. TC channels are controlled by the registers listed in Table 32. The offset of each of the channel registers in Table 32 is in relation to the offset of the corresponding channel as specified in Table 31.

Table 32. TC Channel Memory Map

Offset	Register Name	Description	Access	Reset Value
0x00	TC_CCR	Channel Control Register	Write-only	–
0x04	TC_CMR	Channel Mode Register	Read/Write	0
0x08		Reserved		–
0x0C		Reserved		–
0x10	TC_CVR	Counter Value Register	Read/Write	0
0x14	TC_RA	Register A	Read/Write(1)	0
0x18	TC_RB	Register B	Read/Write(1)	0
0x1C	TC_RC	Register C	Read/Write	0
0x20	TC_SR	Status Register	Read-only	–
0x24	TC_IER	Interrupt Enable Register	Write-only	–
0x28	TC_IDR	Interrupt Disable Register	Write-only	–
0x2C	TC_IMR	Interrupt Mask Register	Read-only	0

Note: 1. Read only if WAVE = 0.

TC Block Control Register

Register Name: TC_BCR
Access Type: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	SYNC

- **SYNC: Synchro Command**

0 = No effect.

1 = Asserts the SYNC signal which generates a software trigger simultaneously for each of the channels.

TC Block Mode Register

Register Name: TC_BMR
Access Type: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	TC2XC2S		TC1XC1S		TC0XC0S	

- **TC0XC0S: External Clock Signal 0 Selection**

TC0XC0S		Signal Connected to XC0
0	0	TCLK0
0	1	None
1	0	TIOA1
1	1	TIOA2

- **TC1XC1S: External Clock Signal 1 Selection**

TC1XC1S		Signal Connected to XC1
0	0	TCLK1
0	1	none
1	0	TIOA0
1	1	TIOA2

- **TC2XC2S: External Clock Signal 2 Selection**

TC2XC2S		Signal Connected to XC2
0	0	TCLK2
0	1	none
1	0	TIOA0
1	1	TIOA1

TC Channel Control Register

Register Name: TC_CCR
Access Type: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	SWTRG	CLKDIS	CLKEN

- **CLKEN: Counter Clock Enable Command**

0 = No effect.

1 = Enables the clock if CLKDIS is not 1.

- **CLKDIS: Counter Clock Disable Command**

0 = No effect.

1 = Disables the clock.

- **SWTRG: Software Trigger Command**

0 = No effect.

1 = A software trigger is performed: the counter is reset and clock is started.

TC Channel Mode Register: Capture Mode

Register Name: TC_CMR
Access Type: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	LDRB		LDRA	
15	14	13	12	11	10	9	8
WAVE	CPCTRG	–	–	–	ABETRG	ETRGEDG	
7	6	5	4	3	2	1	0
LDBDIS	LDBSTOP	BURST		CLKI	TCCLKS		

- **TCCLKS: Clock Selection**

TCCLKS			Clock Selected
0	0	0	ACLK/2
0	0	1	ACLK/8
0	1	0	ACLK/32
0	1	1	ACLK/128
1	0	0	ACLK/1024
1	0	1	XC0
1	1	0	XC1
1	1	1	XC2

- **CLKI: Clock Invert**

0 = Counter is incremented on rising edge of the clock.

1 = Counter is incremented on falling edge of the clock.

- **BURST: Burst Signal Selection**

BURST		
0	0	The clock is not gated by an external signal.
0	1	XC0 is ANDed with the selected clock.
1	0	XC1 is ANDed with the selected clock.
1	1	XC2 is ANDed with the selected clock.

- **LDBSTOP: Counter Clock Stopped with RB Loading**

0 = Counter clock is not stopped when RB loading occurs.

1 = Counter clock is stopped when RB loading occurs.

- **LDBDIS: Counter Clock Disable with RB Loading**

0 = Counter clock is not disabled when RB loading occurs.

1 = Counter clock is disabled when RB loading occurs.

- **ETRGEDG: External Trigger Edge Selection**

ETRGEDG		Edge
0	0	None
0	1	Rising edge
1	0	Falling edge
1	1	Each edge

- **ABETRG: TIOA or TIOB External Trigger Selection**

0 = TIOB is used as an external trigger.

1 = TIOA is used as an external trigger.

- **CPCTRG: RC Compare Trigger Enable**

0 = RC Compare has no effect on the counter and its clock.

1 = RC Compare resets the counter and starts the counter clock.

- **WAVE**

0 = Capture mode is enabled.

1 = Capture mode is disabled (waveform mode is enabled).

- **LDRA: RA Loading Selection**

LDRA		Edge
0	0	None
0	1	Rising edge of TIOA
1	0	Falling edge of TIOA
1	1	Each edge of TIOA

- **LDRB: RB Loading Selection**

LDRB		Edge
0	0	None
0	1	Rising edge of TIOA
1	0	Falling edge of TIOA
1	1	Each edge of TIOA

TC Channel Mode Register: Waveform Mode

Register Name: TC_CMR
Access Type: Read/Write

31	30	29	28	27	26	25	24
BSWTRG		BEEVT		BCPC		BCPB	
23	22	21	20	19	18	17	16
ASWTRG		AEEVT		ACPC		ACPA	
15	14	13	12	11	10	9	8
WAVE	CPCTRG	–	ENETRG	EEVT		EEVTEDG	
7	6	5	4	3	2	1	0
CPCDIS	CPCSTOP	BURST		CLKI	TCCLKS		

- **TCCLKS: Clock Selection**

TCCLKS			Clock Selected
0	0	0	ACLK/2
0	0	1	ACLK/8
0	1	0	ACLK/32
0	1	1	ACLK/128
1	0	0	ACLK/1024
1	0	1	XC0
1	1	0	XC1
1	1	1	XC2

- **CLKI: Clock Invert**

0 = Counter is incremented on rising edge of the clock.

1 = Counter is incremented on falling edge of the clock.

- **BURST: Burst Signal Selection**

BURST		
0	0	The clock is not gated by an external signal.
0	1	XC0 is ANDed with the selected clock.
1	0	XC1 is ANDed with the selected clock.
1	1	XC2 is ANDed with the selected clock.

- **CPCSTOP: Counter Clock Stopped with RC Compare**

0 = Counter clock is not stopped when counter reaches RC.

1 = Counter clock is stopped when counter reaches RC.

- **CPCDIS: Counter Clock Disable with RC Compare**

0 = Counter clock is not disabled when counter reaches RC.

1 = Counter clock is disabled when counter reaches RC.

- **EEVTEDG: External Event Edge Selection**

EEVTEDG		Edge
0	0	None
0	1	Rising edge
1	0	Falling edge
1	1	Each edge

- **EEVT: External Event Selection**

EEVT		Signal Selected as External Event	TIOB Direction
0	0	TIOB	Input(1)
0	1	XC0	Output
1	0	XC1	Output
1	1	XC2	Output

Note: If TIOB is chosen as the external event signal, it is configured as an input and no longer generates waveforms.

- **ENETRG: External Event Trigger Enable**

0 = The external event has no effect on the counter and its clock. In this case, the selected external event only controls the TIOA output.

1 = The external event resets the counter and starts the counter clock.

- **CPCTRG: RC Compare Trigger Enable**

0 = RC Compare has no effect on the counter and its clock.

1 = RC Compare resets the counter and starts the counter clock.

- **WAVE**

0 = Waveform mode is disabled (Capture mode is enabled).

1 = Waveform mode is enabled.

- **ACPA: RA Compare Effect on TIOA**

ACPA		Effect
0	0	None
0	1	Set
1	0	Clear
1	1	Toggle

- **ACPC: RC Compare Effect on TIOA**

ACPC		Effect
0	0	None
0	1	Set
1	0	Clear
1	1	Toggle

- **AEEVT: External Event Effect on TIOA**

AEEVT		Effect
0	0	None
0	1	Set
1	0	Clear
1	1	Toggle

- **ASWTRG: Software Trigger Effect on TIOA**

ASWTRG		Effect
0	0	None
0	1	Set
1	0	Clear
1	1	Toggle

- **BCPB: RB Compare Effect on TIOB**

BCPB		Effect
0	0	None
0	1	Set
1	0	Clear
1	1	Toggle

- **BCPC: RC Compare Effect on TIOB**

BCPC		Effect
0	0	None
0	1	Set
1	0	Clear
1	1	Toggle

- **BEEVT: External Event Effect on TIOB**

BEEVT		Effect
0	0	None
0	1	Set
1	0	Clear
1	1	Toggle

- **BSWTRG: Software Trigger Effect on TIOB**

BSWTRG		Effect
0	0	None
0	1	Set
1	0	Clear
1	1	Toggle

TC Counter Value Register

Register Name: TC_CVR
Access Type: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
CV							
7	6	5	4	3	2	1	0
CV							

- **CV: Counter Value**

CV contains the counter value in real-time.

TC Register A

Register Name: TC_RA
Access Type: Read-only if WAVE = 0, Read/Write if WAVE = 1

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
RA							
7	6	5	4	3	2	1	0
RA							

- **RA: Register A**

RA contains the Register A value in real-time.

TC Register B

Register Name: TC_RB
Access Type: Read-only if WAVE = 0, Read/Write if WAVE = 1

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
RB							
7	6	5	4	3	2	1	0
RB							

- **RB: Register B**

RB contains the Register B value in real-time.

TC Register C

Register Name: TC_RC
Access Type: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
RC							
7	6	5	4	3	2	1	0
RC							

- **RC: Register C**

RC contains the Register C value in real-time.

TC Status Register

Register Name: TC_SR
Access Type: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	MTIOB	MTIOA	CLKSTA
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS

- **COVFS: Counter Overflow Status**

0 = No counter overflow has occurred since the last read of the Status Register.

1 = A counter overflow has occurred since the last read of the Status Register.

- **LOVRS: Load Overrun Status**

0 = Load overrun has not occurred since the last read of the Status Register or WAVE = 1.

1 = RA or RB have been loaded at least twice without any read of the corresponding register since the last read of the Status Register if WAVE = 0.

- **CPAS: RA Compare Status**

0 = RA compare has not occurred since the last read of the Status Register or WAVE = 0.

1 = RA compare has occurred since the last read of the Status Register if WAVE = 1.

- **CPBS: RB Compare Status**

0 = RB compare has not occurred since the last read of the Status Register or WAVE = 0.

1 = RB compare has occurred since the last read of the Status Register if WAVE = 1.

- **CPCS: RC Compare Status**

0 = RC compare has not occurred since the last read of the Status Register.

1 = RC compare has occurred since the last read of the Status Register.

- **LDRAS: RA Loading Status**

0 = RA Load has not occurred since the last read of the Status Register or WAVE = 1.

1 = RA Load has occurred since the last read of the Status Register, if WAVE = 0.

- **LDRBS: RB Loading Status**

0 = RB load has not occurred since the last read of the Status Register or WAVE = 1.

1 = RB load has occurred since the last read of the Status Register if WAVE = 0.

- **ETRGS: External Trigger Status**

0 = External trigger has not occurred since the last read of the Status Register.

1 = External trigger has occurred since the last read of the Status Register.

- **CLKSTA: Clock Enabling Status**

0 = Clock is disabled.

1 = Clock is enabled.

- **MTIOA: TIOA Mirror**

0 = TIOA is low. If WAVE = 0, then TIOA pin is low. If WAVE = 1, then TIOA is driven low.

1 = TIOA is high. If WAVE = 0, then TIOA pin is high. If WAVE = 1, then TIOA is driven high.

- **MTIOB: TIOB Mirror**

0 = TIOB is low. If WAVE = 0, then TIOB pin is low. If WAVE = 1, then TIOB is driven low.

1 = TIOB is high. If WAVE = 0, then TIOB pin is high. If WAVE = 1, then TIOB is driven high.

TC Interrupt Enable Register

Register Name: TC_IER
Access Type: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS

- **COVFS: Counter Overflow**
- **LOVRS: Load Overrun**
- **CPAS: RA Compare**
- **CPBS: RB Compare**
- **CPCS: RC Compare**
- **LDRAS: RA Loading**
- **LDRBS: RB Loading**
- **ETRGS: External Trigger**

0 = No effect.

1 = Enables the corresponding interrupt.

TC Interrupt Disable Register

Register Name: TC_IDR
Access Type: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS

- **COVFS: Counter Overflow**
- **LOVRS: Load Overrun**
- **CPAS: RA Compare**
- **CPBS: RB Compare**
- **CPCS: RC Compare**
- **LDRAS: RA Loading**
- **LDRBS: RB Loading**
- **ETRGS: External Trigger**

0 = No effect.

1 = Disables the corresponding interrupt.

TC Interrupt Mask Register

Register Name: TC_IMR
Access Type: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS

- **COVFS: Counter Overflow**
- **LOVRS: Load Overrun**
- **CPAS: RA Compare**
- **CPBS: RB Compare**
- **CPCS: RC Compare**
- **LDRAS: RA Loading**
- **LDRBS: RB Loading**
- **ETRGS: External Trigger**

0 = The corresponding interrupt is disabled.

1 = The corresponding interrupt is enabled.

Serial Peripheral Interface (SPI)

The AT91C140 embeds a Serial Peripheral Interface featuring:

- Four Chip Selects with External Decoder Support Allowing Communication with Up to 15 Peripherals
- Serial Memories, such as DataFlash and 3-wire EEPROMS
- Serial Peripherals, such as ADCS, DACS, LCD Controllers, CAN Controllers And Sensors
- External Co-processors
- Master or Slave Serial Peripheral Bus Interface
- 8- to 16-bit Programmable Data Length Per Chip Select
- Programmable Phase and Polarity Per Chip Select
- Programmable Transfer Delays Between Consecutive Transfers and Between Clock and Data Per Chip Select
- Programmable Delay Between Consecutive Transfers
- Selectable Mode Fault Detection
- Connection to PDC Channel Capabilities Optimizes Data Transfers
- One Channel for the Receiver, One Channel for the Transmitter

Overview

The Serial Peripheral Interface (SPI) circuit is a synchronous serial data link that provides communication with external devices in Master or Slave Mode. It also allows communication between processors if an external processor is connected to the system.

The Serial Peripheral Interface is a shift register that serially transmits data bits to other SPIs. During a data transfer, one SPI system acts as the “master"' that controls the data flow, while the other system acts as the “slave'' that has data shifted into and out of it by the master. Different CPUs can take turn being masters (Multiple Master Protocol versus Single Master Protocol, where one CPU is always the master while all of the others are always slaves), and one master may simultaneously shift data into multiple slaves. However, only one slave may drive its output to write data back to the master at any given time.

A slave device is selected when the master asserts its NSS signal. If multiple slave devices exist, the master generates a separate slave select signal for each slave (NPCS).

The SPI system consists of two data lines and two control lines:

Master Out Slave In (MOSI): This data line supplies the output data from the master shifted into the input(s) of the slave(s).

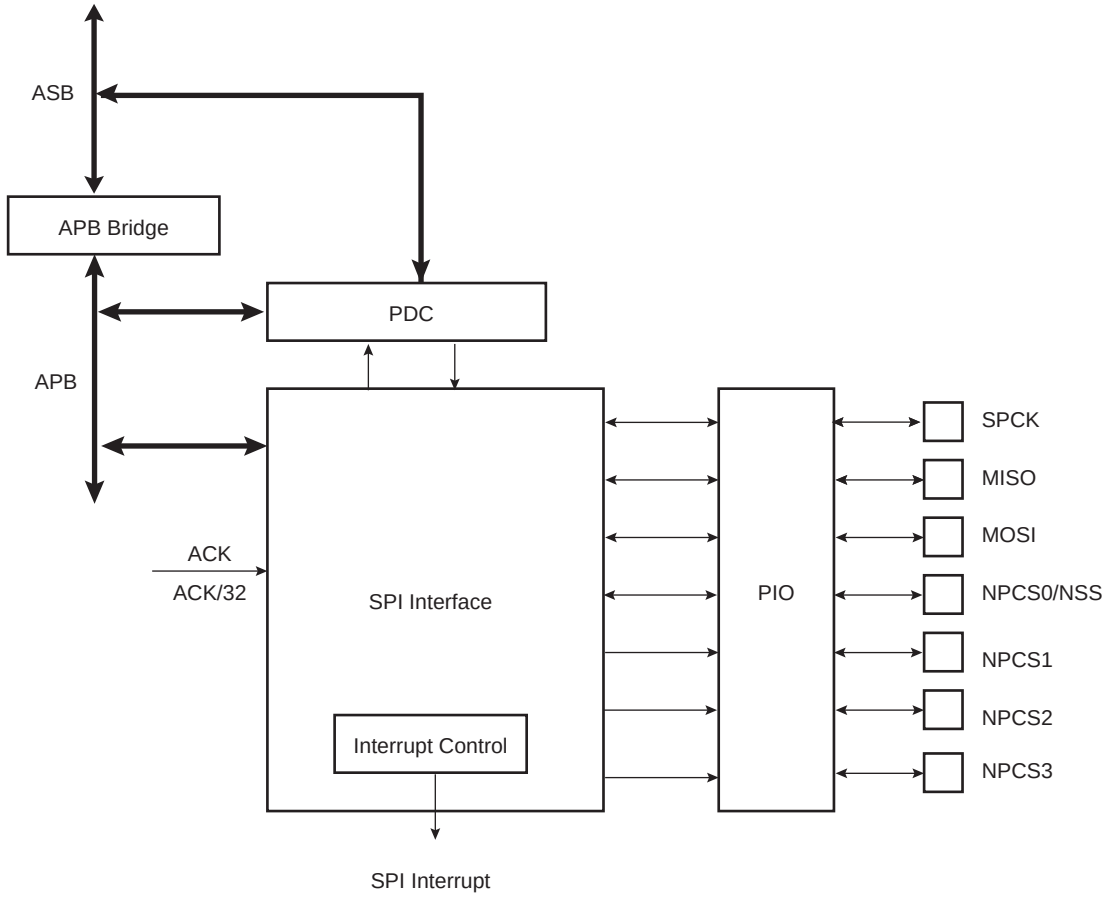
Master In Slave Out (MISO): This data line supplies the output data from a slave to the input of the master. There may be no more than one slave transmitting data during any particular transfer.

Serial Clock (SPCK): This control line is driven by the master and regulates the flow of the data bits. The master may transmit data at a variety of baud rates; the SPCK line cycles once for each bit that is transmitted.

Slave Select (NSS): This control line allows slaves to be turned on and off by hardware.

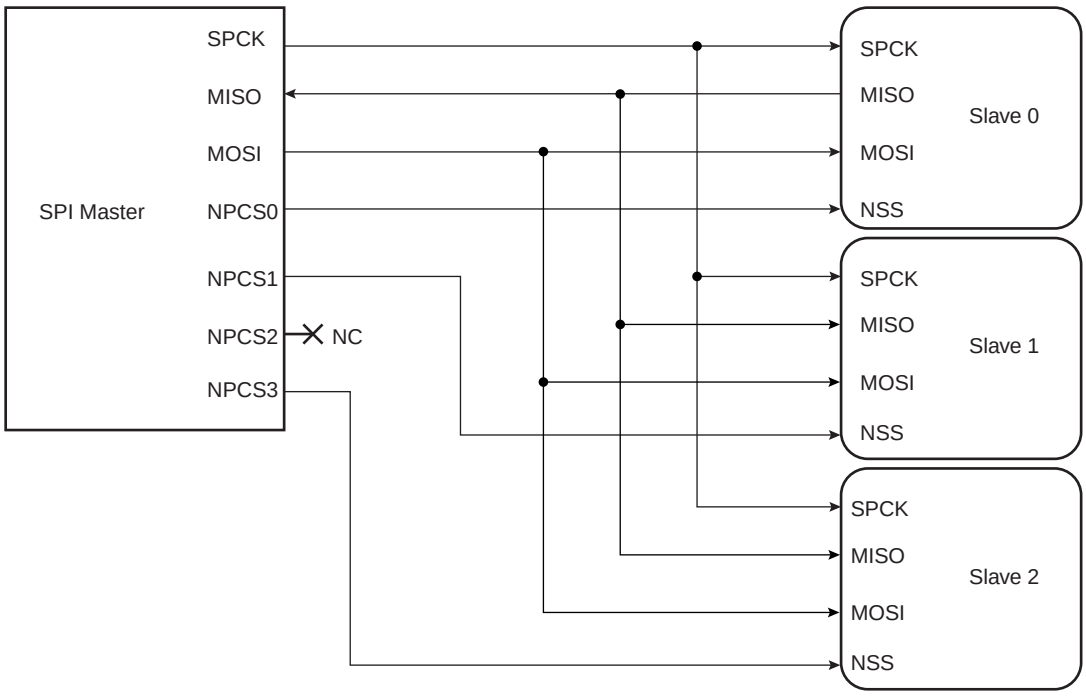
Block Diagram

Figure 34. Block Diagram



Connections

Figure 35. Application Block Diagram: Single Master/Multiple Slave Implementation



Pin Name List

Table 33. I/O Lines Description

Pin Name	Pin Description	Type	Mode	Comments
MISO	Master In Slave Out	I/O	Master Slave	Serial data input to SPI Serial data output from SPI
MOSI	Master Out Slave In	I/O	Master Slave	Serial data output from SPI Serial data input to SPI
SPCK	Serial Clock	I/O	Master Slave	Clock output from SPI Clock input to SPI
NPCS1-NPCS3	Peripheral Chip Selects	Input	Master	Select peripherals
NPCS0/NSS	Peripheral Chip Select/Slave Select	I/O	Master Master Slave	Output: Selects peripheral Input: low causes mode fault Input: chip select for SPI

Master Mode Operations

When configured in Master Mode, the Serial Peripheral Interface controls data transfers to and from the slave(s) connected to the SPI bus. The SPI drives the chip select(s) to the slave(s) and the serial clock (SPCK). After enabling the SPI, a data transfer begins when the core writes to the SPI_TDR (Transmit Data Register).

Transmit and Receive buffers maintain the data flow at a constant rate with a reduced requirement for high-priority interrupt servicing. When new data is available in the

SPI_TDR, the SPI continues to transfer data. If the SPI_RDR (Receive Data Register) has not been read before new data is received, the Overrun Error (OVRES) flag is set.

Note: As long as this flag is set, no data is loaded in the SPI_RDR. The user has to read the status register to clear it.

The delay between the activation of the chip select and the start of the data transfer (DLYBS), as well as the delay between each data transfer (DLYBCT), can be programmed for each of the four external chip selects. All data transfer characteristics, including the two timing values, are programmed in registers SPI_CSR0 to SPI_CSR3 (Chip Select Registers).

In Master Mode, the peripheral selection can be defined in two different ways:

- Fixed Peripheral Select: SPI exchanges data with only one peripheral
- Variable Peripheral Select: Data can be exchanged with more than one peripheral

Figure 39 and Figure 40 show the operation of the SPI in Master Mode. For details concerning the flag and control bits in these diagrams, see “SPI Chip Select Register” on page 157.

Fixed Peripheral Select

This mode is used for transferring memory blocks without the extra overhead in the transmit data register to determine the peripheral.

Fixed Peripheral Select is activated by setting bit PS to zero in SPI_MR (Mode Register). The peripheral is defined by the PCS field in SPI_MR.

This option is only available when the SPI is programmed in Master Mode.

Variable Peripheral Select

Variable Peripheral Select is activated by setting the PS bit to one. The PCS field in SPI_TDR is used to select the destination peripheral. The data transfer characteristics are changed when the selected peripheral changes, depending on the associated chip select register.

The PCS field in the SPI_MR has no effect.

This option is available only when the SPI is programmed in Master Mode.

Chip Selects

The Chip Select lines are driven by the SPI only if it is programmed in Master Mode. These lines are used to select the destination peripheral. The PCSDEC field in SPI_MR (Mode Register) selects one to four peripherals (PCSDEC = 0) or up to 15 peripherals (PCSDEC = 1).

If Variable Peripheral Select is active, the chip select signals are defined for each transfer in the PCS field in SPI_TDR. Chip select signals can thus be defined independently for each transfer.

If Fixed Peripheral Select is active, Chip Select signals are defined for all transfers by the field PCS in SPI_MR. If a transfer with a new peripheral is necessary, the software must wait until the current transfer is completed, then change the value of PCS in SPI_MR before writing new data in SPI_TDR.

The value on the NPCS pins at the end of each transfer can be read in the SPI_RDR (Receive Data Register).

By default, all NPCS signals are high (equal to one) before and after each transfer.

Mode Fault Detection

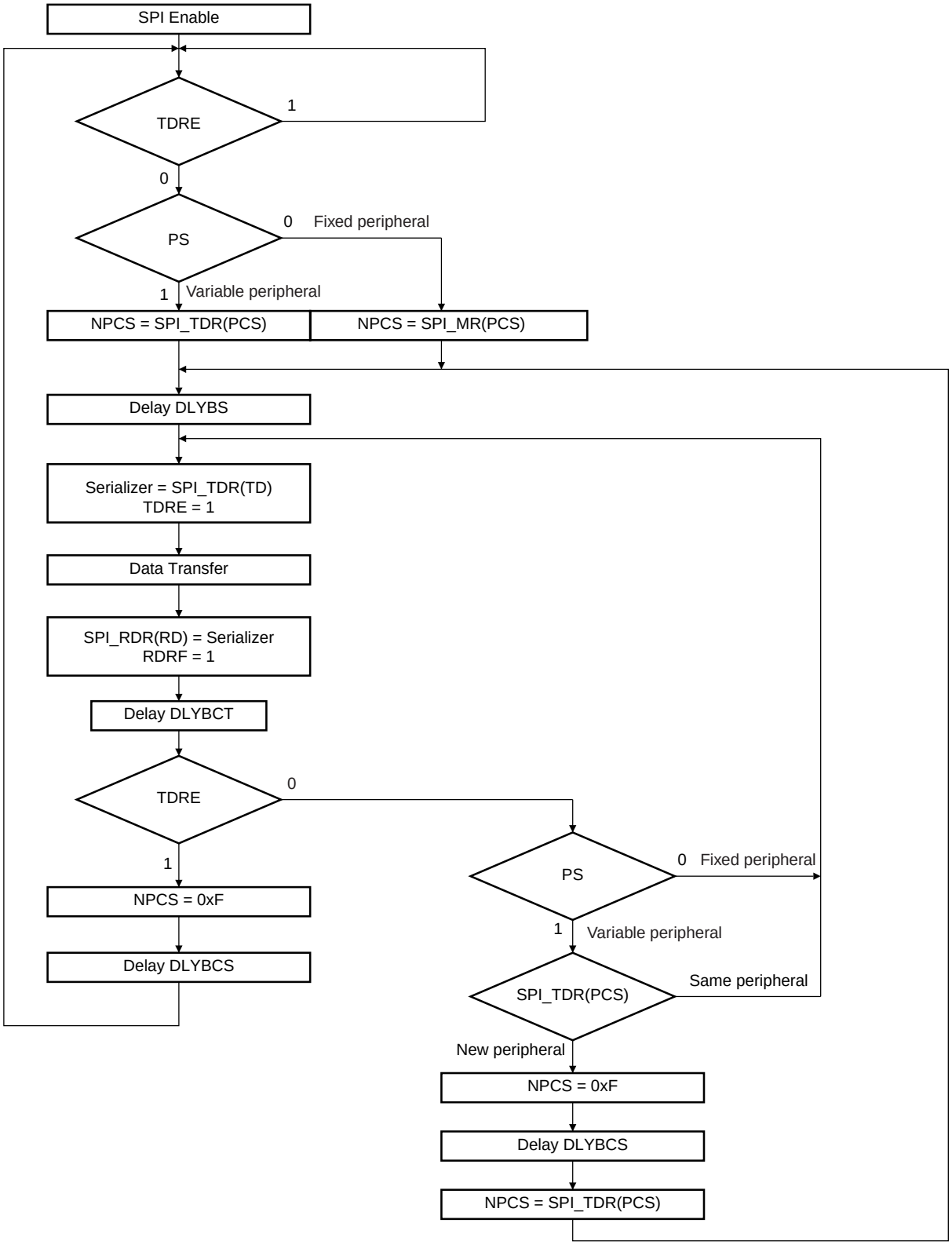
A mode fault is detected when the SPI is programmed in Master Mode and a low level is driven by an external master on the NPCS[0]/NSS signal.

When a mode fault is detected, the MODF bit in the SPI_SR is set until the SPI_SR is read and the SPI is disabled until re-enabled by bit SPIEN in the SPI_CR (Control Register).

By default, Mode Fault Detection is enabled. It is disabled by setting the MODFDIS bit in the SPI Mode Register.

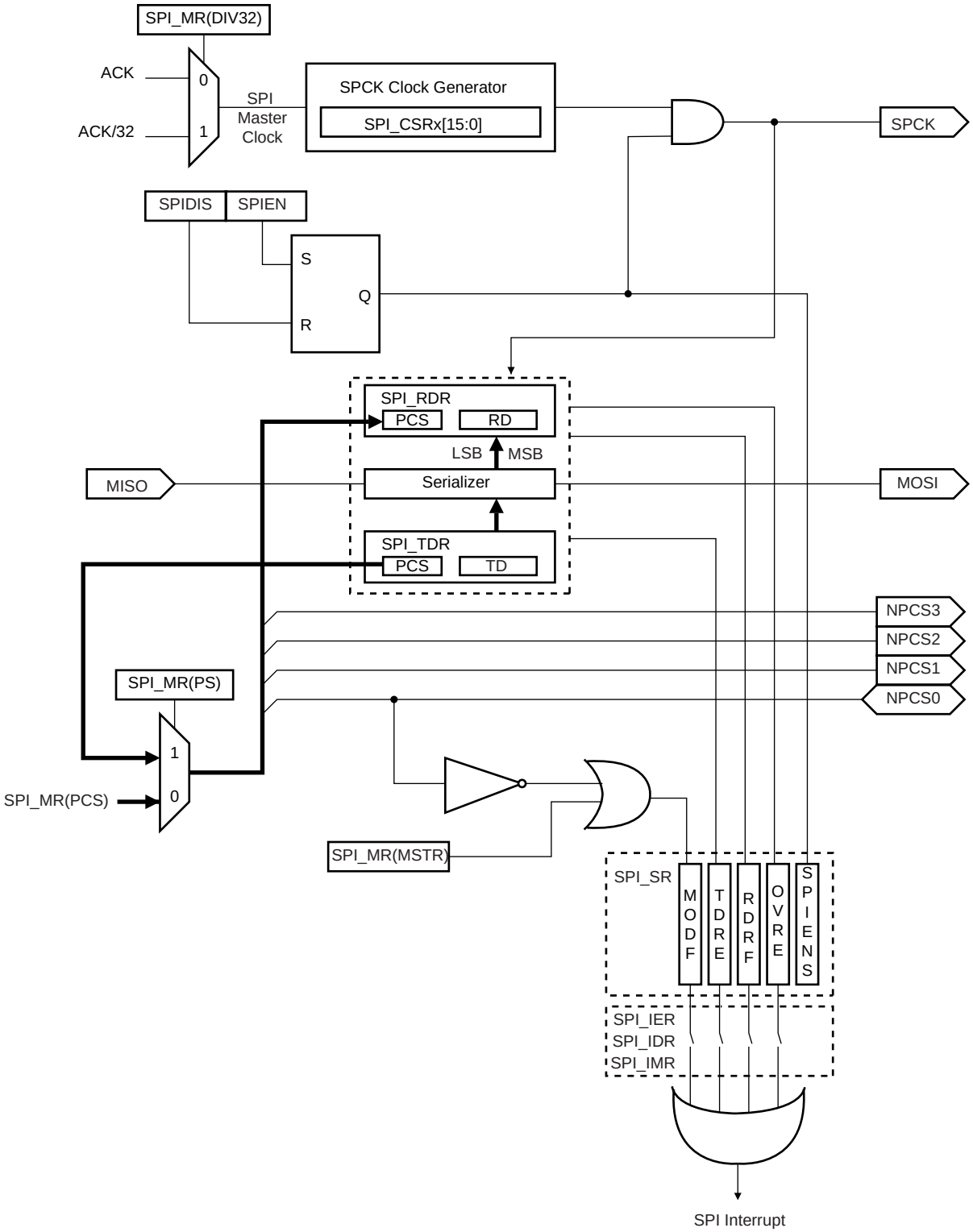
Master Mode Flow Diagram

Figure 36. Master Mode Flow Diagram



Master Mode Block Diagram

Figure 37. Master Mode Block Diagram

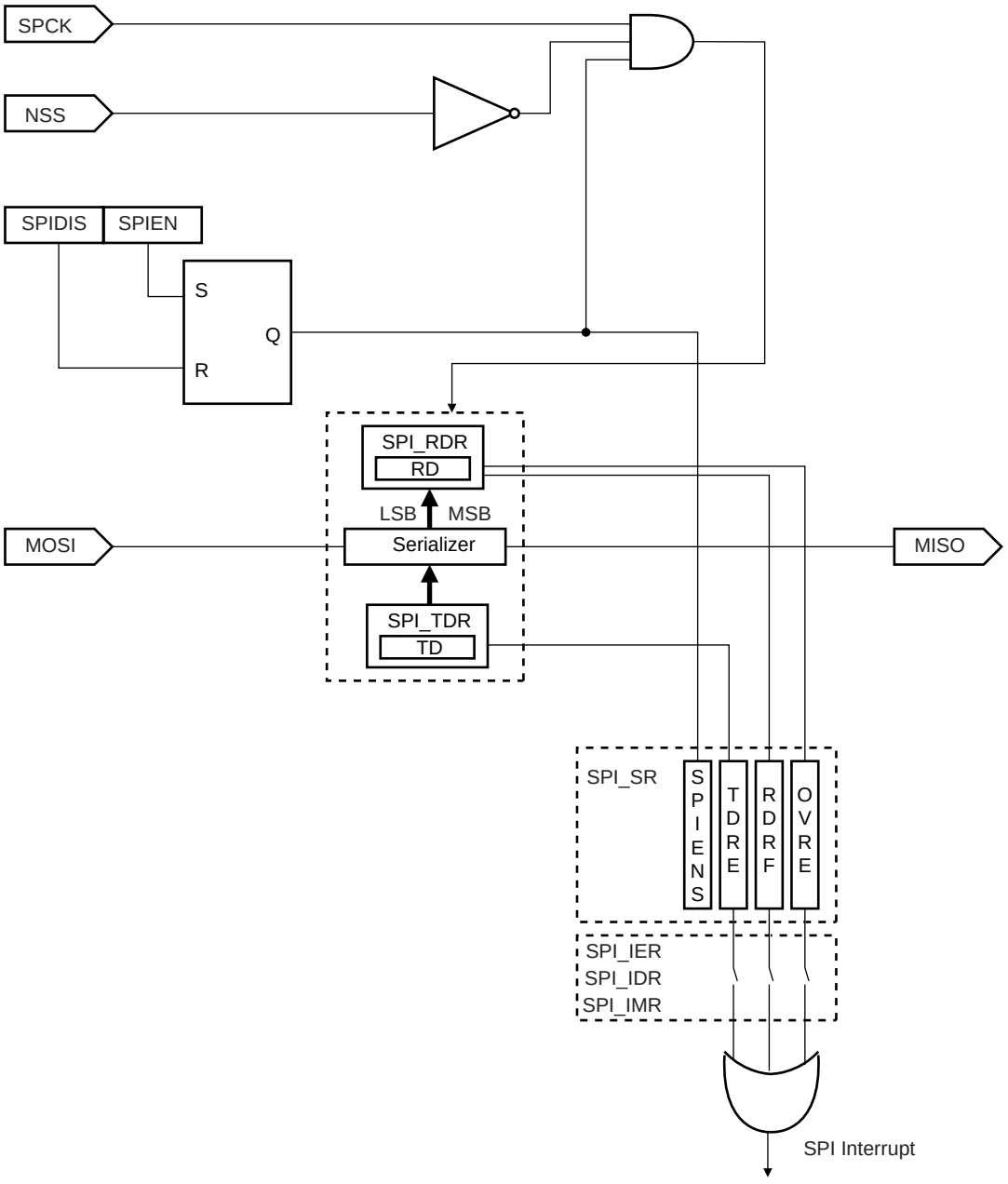


SPI Slave Mode

In Slave Mode, the SPI waits for NSS to go active low before receiving the serial clock from an external master.

In Slave Mode, CPOL, NCPHA and BITS fields of SPI_CSR0 are used to define the transfer characteristics. The other Chip Select Registers are not used in Slave Mode.

Figure 38. Slave Mode Block Diagram



Data Transfers

Four modes are used for data transfers. These modes correspond to combinations of a pair of parameters called clock polarity (CPOL) and clock phase (CPHA) that determine the edges of the clock signal on which the data are driven and sampled. Each of the two parameters has two possible states, resulting in four possible combinations that are incompatible with one another. Thus a master/slave pair must use the same parameter

pair values to communicate. If multiple slaves are used and fixed in different configurations, the master must reconfigure itself each time it needs to communicate with a different slave.

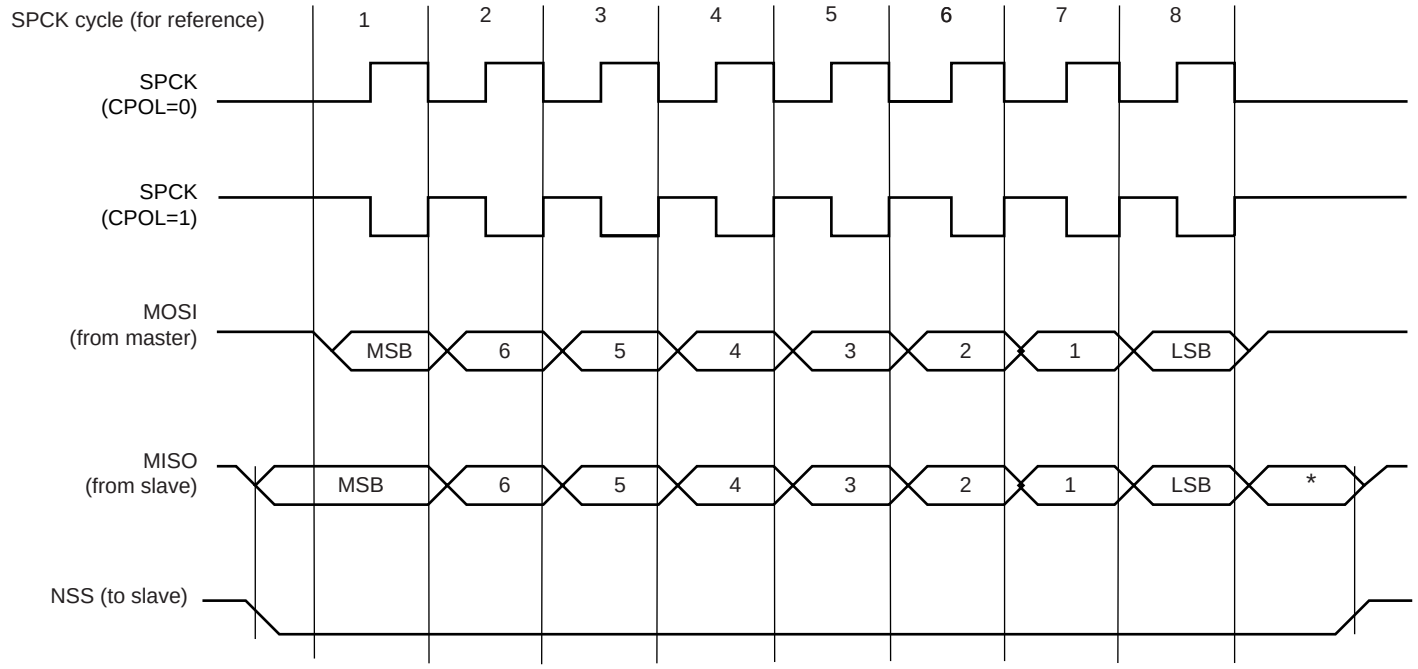
Table 34 shows the four modes and corresponding parameter settings.

Table 34. SPI Bus Protocol Mode

SPI Mode	CPOL	CPHA
0	0	0
1	0	1
2	1	0
3	1	1

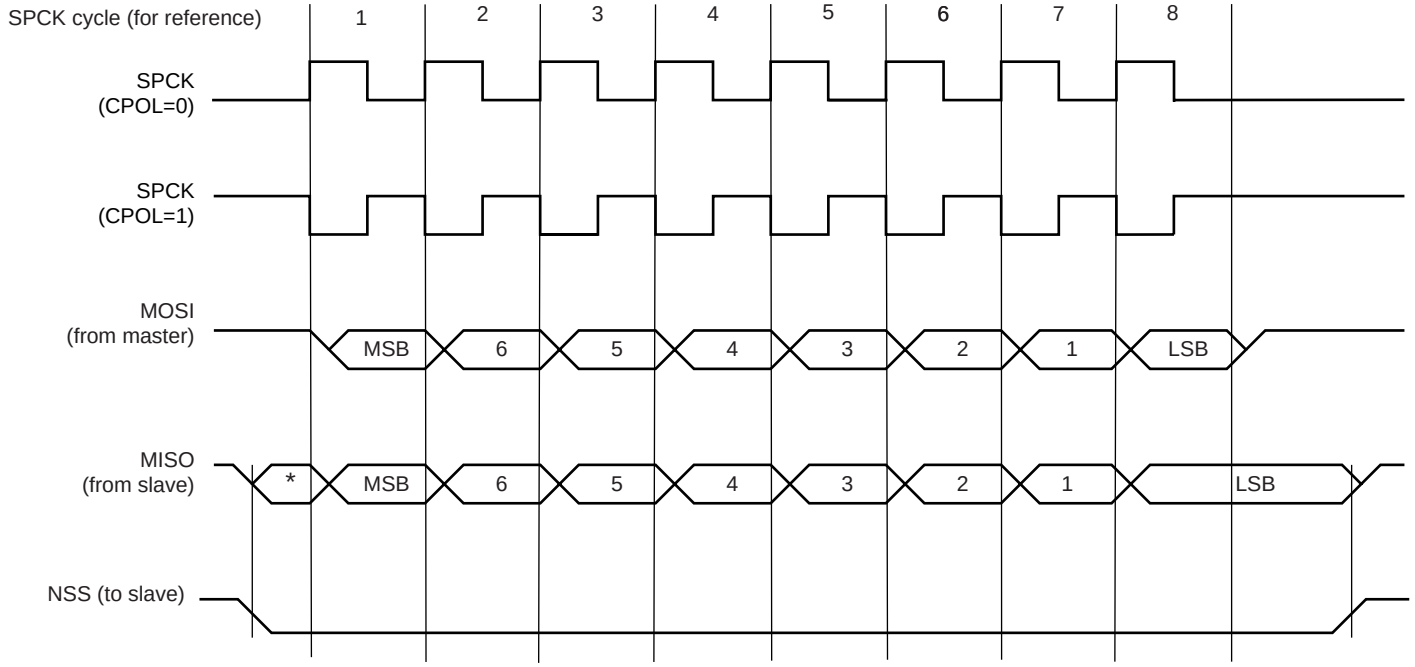
Figure 39 and Figure 40 show examples of data transfers.

Figure 39. SPI Transfer Format (NCPHA = 1, 8 bits per transfer



* Not defined, but normally MSB of previous character received.

Figure 40. SPI Transfer Format (NCPHA = 0, 8 bits per transfer)



* Not defined but normally LSB of previous character transmitted.

Clock Generation

In Master Mode, the SPI Master Clock is either CLOCK or FDIV, as defined by the DIV32 field of SPI_MR. The SPI baud rate clock is generated by dividing the SPI Master Clock by a value between 4 and 510. The divisor is defined in the SCBR field in each Chip Select Register. The transfer speed can thus be defined independently for each chip select signal.

CPOL and NCPHA in the Chip Select Registers define the clock/data relationship between master and slave devices. CPOL defines the inactive value of the SPCK. NCPHA defines which edge causes data to change and which edge causes data to be captured.

In Slave Mode, the input clock low and high pulse duration must be longer than two system clock (CLOCK) periods.

Serial Peripheral Interface (SPI) User Interface

Table 35. SPI Memory Map

Offset	Register	Register Name	Access	Reset
0x00	Control Register	SPI_CR	Write-only	---
0x04	Mode Register	SPI_MR	Read/Write	0x0
0x08	Receive Data Register	SPI_RDR	Read-only	0x0
0x0C	Transmit Data Register	SPI_TDR	Write-only	---
0x10	Status Register	SPI_SR	Read-only	0x000000F0
0x14	Interrupt Enable Register	SPI_IER	Write-only	---
0x18	Interrupt Disable Register	SPI_IDR	Write-only	---
0x1C	Interrupt Mask Register	SPI_IMR	Read-only	0x0
0x20	Receive Pointer Register	SPI_RPR	Read/Write	0x0
0x24	Receive Counter Register	SPI_RCR	Read/Write	0x0
0x28	Transmit Pointer Register	SPI_TPR	Read/Write	0x0
0x2C	Transmit Counter Register	SPI_TCR	Read/Write	0x0
0x30	Chip Select Register 0	SPI_CSR0	Read/Write	0x0
0x34	Chip Select Register 1	SPI_CSR1	Read/Write	0x0
0x38	Chip Select Register 2	SPI_CSR2	Read/Write	0x0
0x3C	Chip Select Register 3	SPI_CSR3	Read/Write	0x0

SPI Control Register

Name: SPI_CR

Access Type: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
SWRST	–	–	–	–	–	SPIDIS	SPIEN

- **SPIEN: SPI Enable**

0 = No effect.

1 = Enables the SPI to transfer and receive data.

- **SPIDIS: SPI Disable**

0 = No effect.

1 = Disables the SPI.

All pins are set in input mode and no data is received or transmitted.

If a transfer is in progress, the transfer is finished before the SPI is disabled.

If both SPIEN and SPIDIS are equal to one when the control register is written, the SPI is disabled

- **SWRST: SPI Software Reset**

0 = No effect.

1 = Resets the SPI. A software-triggered hardware reset of the SPI interface is performed.

SPI Mode Register

Name: SPI_MR
Access Type: Read/Write

31	30	29	28	27	26	25	24
DLYBCS							
23	22	21	20	19	18	17	16
–	–	–	–	PCS			
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
LLB	–	–	MODFDIS	DIV32	PCSDEC	PS	MSTR

- **MSTR: Master/Slave Mode**

0 = SPI is in Slave mode.

1 = SPI is in Master mode.

- **PS: Peripheral Select**

0 = Fixed Peripheral Select.

1 = Variable Peripheral Select.

- **PCSDEC: Chip Select Decode**

0 = The chip selects are directly connected to a peripheral device.

1 = The four chip select lines are connected to a 4- to 16-bit decoder.

When PCSDEC equals one, up to 16 Chip Select signals can be generated with the four lines using an external 4- to 16-bit decoder.

The Chip Select Registers define the characteristics of the 16 chip selects according to the following rules:

SPI_CSR0 defines peripheral chip select signals 0 to 3.

SPI_CSR1 defines peripheral chip select signals 4 to 7.

SPI_CSR2 defines peripheral chip select signals 8 to 11.

SPI_CSR3 defines peripheral chip select signals 12 to 15*.

***Note**: T*he 16th state corresponds to a state in which all chip selects are inactive. This allows a different clock configuration to be defined by each chip select register.

- **DIV32: Clock Selection**

0 = SPI Master Clock equals ACK.

1 = SPI Master Clock equals ACK/32.

- **MODFDIS: Mode Fault Detection**

0 = Mode fault detection is enabled.

1 = Mode fault detection is disabled.

- **LLB: Local Loopback Enable**

0 = Local loopback path disabled

1 = Local loopback path enabled

LLB controls the local loopback on the data serializer for testing in Master Mode only.

- **PCS: Peripheral Chip Select**

This field is only used if Fixed Peripheral Select is active (PS = 0).

If PCSDEC = 0:

PCS = xxx0 NPCS[3:0] = 1110

PCS = xx01 NPCS[3:0] = 1101

PCS = x011 NPCS[3:0] = 1011

PCS = 0111 NPCS[3:0] = 0111

PCS = 1111 forbidden (no peripheral is selected)

(x = don't care)

If PCSDEC = 1:

NPCS[3:0] output signals = PCS

- **DLYBCS: Delay Between Chip Selects**

This field defines the delay from NPCS inactive to the activation of another NPCS. The DLYBCS time guarantees non-overlapping chip selects and solves bus contentions in case of peripherals having long data float times.

If DLYBCS is less than or equal to six, six SPI Master Clock periods will be inserted by default.

Otherwise, the following equation determines the delay:

NPCS_to_SCK_Delay = DLYBCS * SPI_Master_Clock_period

SPI Receive Data Register

Name: SPI_RDR
Access Type: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	PCS			
15	14	13	12	11	10	9	8
RD							
7	6	5	4	3	2	1	0
RD							

- **RD: Receive Data**

Data received by the SPI Interface is stored in this register right-justified. Unused bits read zero.

- **PCS: Peripheral Chip Select**

In Master Mode only, these bits indicate the value on the NPCS pins at the end of a transfer. Otherwise, these bits read zero.

SPI Transmit Data Register

Name: SPI_TDR
Access Type: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	PCS			
15	14	13	12	11	10	9	8
TD							
7	6	5	4	3	2	1	0
TD							

- **TD: Transmit Data**

Data to be transmitted by the SPI is stored in this register. Information to be transmitted must be written to the transmit data register in a right-justified format.

PCS: Peripheral Chip Select

This field is only used if Variable Peripheral Select is active (PS = 1).

If PCSDEC = 0:

PCS = xxx0 NPCS[3:0] = 1110
PCS = xx01 NPCS[3:0] = 1101
PCS = x011 NPCS[3:0] = 1011
PCS = 0111 NPCS[3:0] = 0111
PCS = 1111 forbidden (no peripheral is selected)
(x = don't care)

If PCSDEC = 1:

NPCS[3:0] output signals = PCS

SPI Status Register

Name: SPI_SR

Access Type: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	SPIENS
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	ENDTX	ENDRX	OVRES	MODF	TDRE	RDRF

- **RDRF: Receive Data Register Full**

0 = No data has been received since the last read of SPI_RDR

1 = Data has been received and the received data has been transferred from the serializer to SPI_RDR since the last read of SPI_RDR.

- **TDRE: Transmit Data Register Empty**

0 = Data has been written to SPI_TDR and not yet transferred to the serializer.

1 = The last data written in the Transmit Data Register has been transferred to the serializer.

TDRE equals zero when the SPI is disabled or at reset. The SPI enable command sets this bit to one.

- **MODF: Mode Fault Error**

0 = No Mode Fault has been detected since the last read of SPI_SR.

1 = A Mode Fault occurred since the last read of the SPI_SR.

- **OVRES: Overrun Error Status**

0 = No overrun has been detected since the last read of SPI_SR.

1 = An overrun has occurred since the last read of SPI_SR.

An overrun occurs when SPI_RDR is loaded at least twice from the serializer since the last read of the SPI_RDR.

- **ENDRX: End of RX buffer**

0 = The Receive Counter Register has not reached 0 since the last write in SPI_RCR.

1 = The Receive Counter Register has reached 0 since the last write in SPI_RCR.

- **ENDTX: End of TX buffer**

0 = The Transmit Counter Register has not reached 0 since the last write in SPI_TCR.

1 = The Transmit Counter Register has reached 0 since the last write in SPI_TCR.

- **SPIENS: SPI Enable Status**

0 = SPI is disabled.

1 = SPI is enabled.

SPI Interrupt Enable Register

Name: SPI_IER
Access Type: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	ENDTX	ENDRX	OVRES	MODF	TDRE	RDRF

- **RDRF: Receive Data Register Full Interrupt Enable**
- **TDRE: SPI Transmit Data Register Empty Interrupt Enable**
- **MODF: Mode Fault Error Interrupt Enable**
- **OVRES: Overrun Error Interrupt Enable**
- **ENDRX: End of Receive Buffer Interrupt Enable**
- **ENDTX: End of Transmit Buffer Interrupt Enable**

0 = No effect.

1 = Enables the corresponding interrupt.

SPI Interrupt Disable Register

Name: SPI_IDR
Access Type: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	ENDTX	ENDRX	OVRES	MODF	TDRE	RDRF

- **RDRF: Receive Data Register Full Interrupt Disable**
- **TDRE: SPI Transmit Data Register Empty Interrupt Disable**
- **MODF: Mode Fault Error Interrupt Disable**
- **OVRES: Overrun Error Interrupt Disable**
- **ENDRX: End of Receive Buffer Interrupt Disable**
- **ENDTX: End of Transmit Buffer Interrupt Disable**

0 = No effect.

1 = Disables the corresponding interrupt.

SPI Interrupt Mask Register

Name: SPI_IMR
Access Type: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	ENDTX	ENDRX	OVRES	MODF	TDRE	RDRF

- **RDRF: Receive Data Register Full Interrupt Mask**
- **TDRE: SPI Transmit Data Register Empty Interrupt Mask**
- **MODF: Mode Fault Error Interrupt Mask**
- **OVRES: Overrun Error Interrupt Mask**
- **ENDRX: End of Receive Buffer Interrupt Mask**
- **ENDTX: End of Transmit Buffer Interrupt Mask**

0 = The corresponding interrupt is not enabled.

1 = The corresponding interrupt is enabled.

SPI Chip Select Register

Name: SPI_CSR0... SPI_CSR3
Access Type: Read/Write

31	30	29	28	27	26	25	24
DLYBCT							
23	22	21	20	19	18	17	16
DLYBS							
15	14	13	12	11	10	9	8
SCBR							
7	6	5	4	3	2	1	0
BITS				–	–	NCPHA	CPOL

- **CPOL: Clock Polarity**

0 = The inactive state value of SCK is logic level zero.

1 = The inactive state value of SCK is logic level one.

CPOL is used to determine the inactive state value of the serial clock (SCK). It is used with NCPHA to produce the required clock/data relationship between master and slave devices.

- **NCPHA: Clock Phase**

0 = Data is changed on the leading edge of SCK and captured on the following edge of SCK.

1 = Data is captured on the leading edge of SCK and changed on the following edge of SCK.

NCPHA determines which edge of SCK causes data to change and which edge causes data to be captured. NCPHA is used with CPOL to produce the required clock/data relationship between master and slave devices.

- **BITS: Bits Per Transfer**

The BITS field determines the number of data bits transferred. Reserved values should not be used.

BITS[3:0]	Bits Per Transfer
0000	8
0001	9
0010	10
0011	11
0100	12
0101	13
0110	14
0111	15
1000	16
1001	Reserved
1010	Reserved
1011	Reserved
1100	Reserved
1101	Reserved
1110	Reserved
1111	Reserved

- **SCBR: Serial Clock Baud Rate**

In Master Mode, the SPI Interface uses a modulus counter to derive the SPCK baud rate from the SPI Master Clock (selected between CLOCK and FDIV). The Baud rate is selected by writing a value from 2 to 255 in the field SCBR. The following equation determines the SPCK baud rate:

SPCK Baudrate = SPI_Master_Clock / (2 * SCBR)

Giving SCBR a value of zero or one disables the baud rate generator. SPCK is disabled and assumes its inactive state value. No serial transfers may occur. At reset, baud rate is disabled.

- **DLYBS: Delay Before SCK**

This field defines the delay from NPCS valid to the first valid SCK transition.

When DLYBS equals zero, the NPCS valid to SCK transition is 1/2 the SCK clock period.

Otherwise, the following equation determines the delay:

NPCS_to_SCK_Delay = DLYBS * SPI_Master_Clock_period

- **DLYBCT: Delay Between Consecutive Transfers**

This field defines the delay between two consecutive transfers with the same peripheral without removing the chip select. The delay is always inserted after each transfer and before removing the chip select if needed.

When DLYBCT equals zero, a delay of four SPI Master Clock periods are inserted.

Otherwise, the following equation determines the delay:

Delay_After_Transfer = 32 * DLYBCT * SPI_Master_Clock_period.

SPI Receive Pointer Register

Register Name: SPI_RPR
Access Type: Read/Write

31	30	29	28	27	26	25	24
RXPTR							
23	22	21	20	19	18	17	16
RXPTR							
15	14	13	12	11	10	9	8
RXPTR							
7	6	5	4	3	2	1	0
RXPTR							

- **RXPTR: Receive Pointer**

RXPTR must be loaded with the address of the receive buffer.

SPI Receive Counter Register

Register Name: SPI_RCR
Access Type: Read/Write

31	30	29	28	27	26	25	24
--	--	--	--	--	--	--	--
23	22	21	20	19	18	17	16
--	--	--	--	--	--	--	--
15	14	13	12	11	10	9	8
RXCTR							
7	6	5	4	3	2	1	0
RXCTR							

- **RXCTR: Receive Counter Register**

RXCTR must be loaded with the size of the receive buffer.

0 = Stops peripheral data transfer

1 - 65535 = Start peripheral data transfer if RDRF is active.

SPI Transmit Pointer Register

Register Name: SP_TPR
Access Type: Read/Write

31	30	29	28	27	26	25	24
TXPTR							
23	22	21	20	19	18	17	16
TXPTR							
15	14	13	12	11	10	9	8
TXPTR							
7	6	5	4	3	2	1	0
TXPTR							

- **TXPTR: Transmit Pointer Register**

TXPTR must be loaded with the address of the transmit buffer.

SPI Transmit Counter Register

Register Name: SP_TCR
Access Type: Read/Write

31	30	29	28	27	26	25	24
--	--	--	--	--	--	--	--
23	22	21	20	19	18	17	16
--	--	--	--	--	--	--	--
15	14	13	12	11	10	9	8
TXCTR							
7	6	5	4	3	2	1	0
TXCTR							

- **TXCTR: Transmit Counter Register**

TXCTR must be loaded with the size of the receive buffer.

0 = Stops peripheral data transfer

1 - 65535 = Start peripheral data transfer if TDRE is active.

Ordering Information

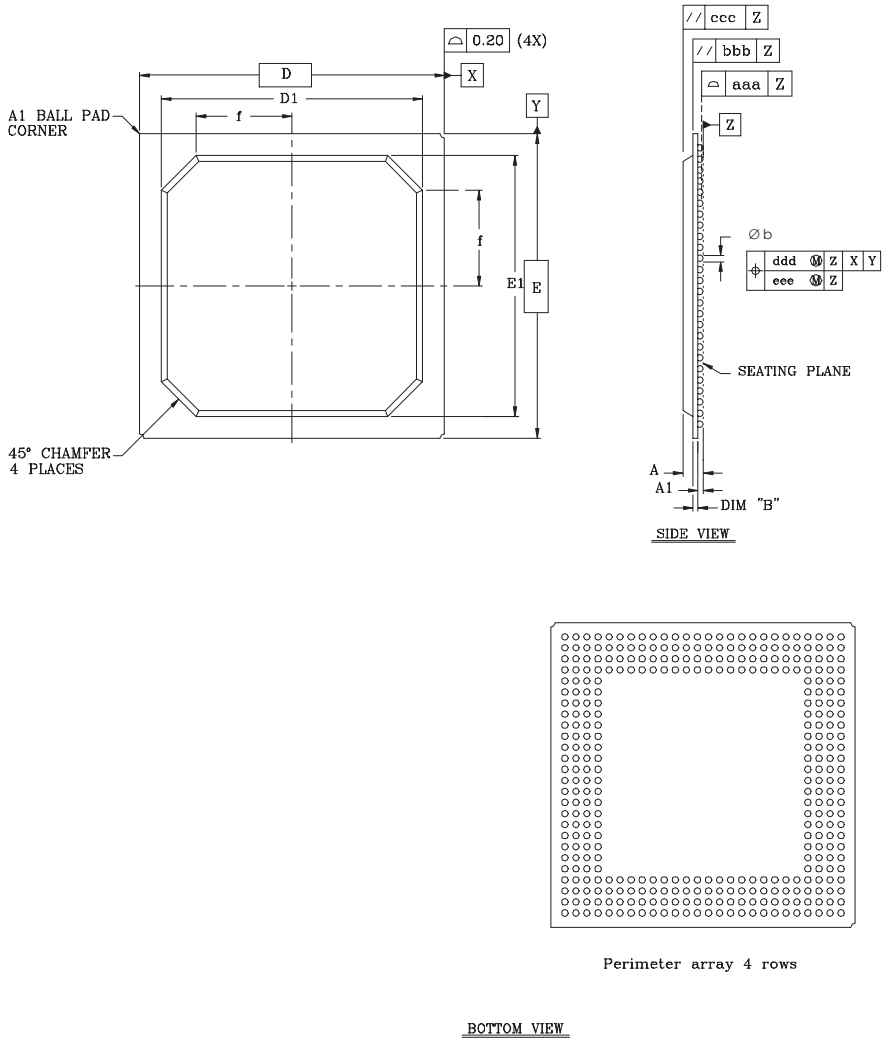
Table 36. Ordering Information

Ordering Reference	Temp	Package
AT91C140-CI	-40°C to +85°C	BGA256

Mechanical Characteristics and Packaging

BGA Packaging Information

Figure 41. AT91C140 BGA Package



For BGA package data, see Table 37 on page 163,

BGA Package Data .

Table 37. Dimensions (mm)

Symbol	Min	Nom	Max
A1	0.50	0.60	0.70
∅b	0.60	0.75	0.90
aaa		0.30	
bbb		0.25	
ccc		0.35	
ddd		0.30	
eee		0.15	
A	1.92	2.13	2.34
B	0.28	0.32	0.38
D/E	26.8	27.0	27.2
D1/E1		24.0	24.7
e		1.27	
f		8.05	

Table of Contents

Table of Contents i

Atmel Corporation

2325 Orchard Parkway
San Jose, CA 95131, USA
Tel: 1(408) 441-0311
Fax: 1(408) 487-2600

Regional Headquarters

Europe
Atmel Sarl
Route des Arsenaux 41
Case Postale 80
CH-1705 Fribourg
Switzerland
Tel: (41) 26-426-5555
Fax: (41) 26-426-5500

Asia
Room 1219
Chinachem Golden Plaza
77 Mody Road Tsimshatsui
East Kowloon
Hong Kong
Tel: (852) 2721-9778
Fax: (852) 2722-1369

Japan
9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chuo-ku, Tokyo 104-0033
Japan
Tel: (81) 3-3523-3551
Fax: (81) 3-3523-7581

Atmel Operations

Memory
2325 Orchard Parkway
San Jose, CA 95131, USA
Tel: 1(408) 441-0311
Fax: 1(408) 436-4314

Microcontrollers
2325 Orchard Parkway
San Jose, CA 95131, USA
Tel: 1(408) 441-0311
Fax: 1(408) 436-4314

La Chantrerie
BP 70602
44306 Nantes Cedex 3, France
Tel: (33) 2-40-18-18-18
Fax: (33) 2-40-18-19-60

ASIC/ASSP/Smart Cards
Zone Industrielle
13106 Rousset Cedex, France
Tel: (33) 4-42-53-60-00
Fax: (33) 4-42-53-60-01

1150 East Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906, USA
Tel: 1(719) 576-3300
Fax: 1(719) 540-1759

Scottish Enterprise Technology Park
Maxwell Building
East Kilbride G75 0QR, Scotland
Tel: (44) 1355-803-000
Fax: (44) 1355-242-743

RF/Automotive
Theresienstrasse 2
Postfach 3535
74025 Heilbronn, Germany
Tel: (49) 71-31-67-0
Fax: (49) 71-31-67-2340

1150 East Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906, USA
Tel: 1(719) 576-3300
Fax: 1(719) 540-1759

Biometrics/Imaging/Hi-Rel MPU/
High Speed Converters/RF Datacom
Avenue de Rochepleine
BP 123
38521 Saint-Egreve Cedex, France
Tel: (33) 4-76-58-30-00
Fax: (33) 4-76-58-34-80

Literature Requests
www.atmel.com/literature

6069A–ATARM–05/04