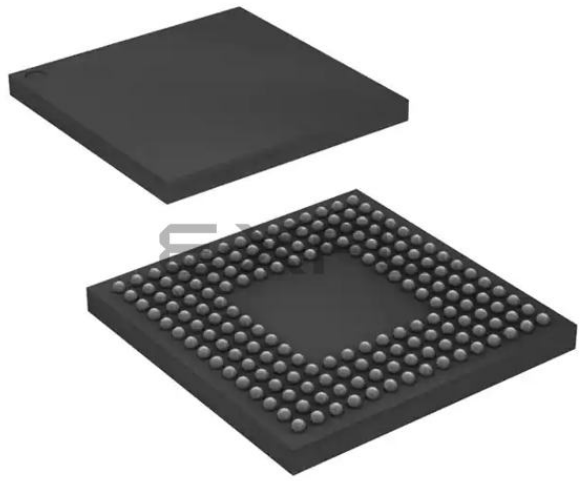


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Obsolete
Core Processor	ARM7®
Core Size	16/32-Bit
Speed	33MHz
Connectivity	EBI/EMI, SPI, UART/USART
Peripherals	POR, WDT
Number of I/O	58
Program Memory Size	-
Program Memory Type	ROMless
EEPROM Size	-
RAM Size	8K x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 3.6V
Data Converters	A/D 8x10b; D/A 2x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	176-LFBGA
Supplier Device Package	176-BGA (13x13)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/at91m55800a-33cj-t

5. Architectural Overview

The AT91M55800A microcontroller integrates an ARM7TDMI with its EmbeddedICE interface, memories and peripherals. Its architecture consists of two main buses, the Advanced System Bus (ASB) and the Advanced Peripheral Bus (APB). Designed for maximum performance and controlled by the memory controller, the ASB interfaces the ARM7TDMI processor with the on-chip 32-bit memories, the External Bus Interface (EBI) and the AMBA™ Bridge. The AMBA Bridge drives the APB, which is designed for accesses to on-chip peripherals and optimized for low power consumption.

The AT91M55800A microcontroller implements the ICE port of the ARM7TDMI processor on dedicated pins, offering a complete, low cost and easy-to-use debug solution for target debugging.

5.1 Memory

The AT91M55800A microcontroller embeds 8K bytes of internal SRAM. The internal memory is directly connected to the 32-bit data bus and is single-cycle accessible.

The AT91M55800A microcontroller features an External Bus Interface (EBI), which enables connection of external memories and application-specific peripherals. The EBI supports 8- or 16-bit devices and can use two 8-bit devices to emulate a single 16-bit device. The EBI implements the early read protocol, enabling faster memory accesses than standard memory interfaces.

5.2 Peripherals

The AT91M55800A microcontroller integrates several peripherals, which are classified as system or user peripherals. All on-chip peripherals are 32-bit accessible by the AMBA Bridge, and can be programmed with a minimum number of instructions. The peripheral register set is composed of control, mode, data, status and enable/disable/status registers.

An on-chip, 8-channel Peripheral Data Controller (PDC) transfers data between the on-chip USARTs/SPI and the on and off-chip memories without processor intervention. One PDC channel is connected to the receiving channel and one to the transmitting channel of each USART and of the SPI.

Most importantly, the PDC removes the processor interrupt handling overhead and significantly reduces the number of clock cycles required for a data transfer. It can transfer up to 64K contiguous bytes. As a result, the performance of the microcontroller is increased and the power consumption reduced.

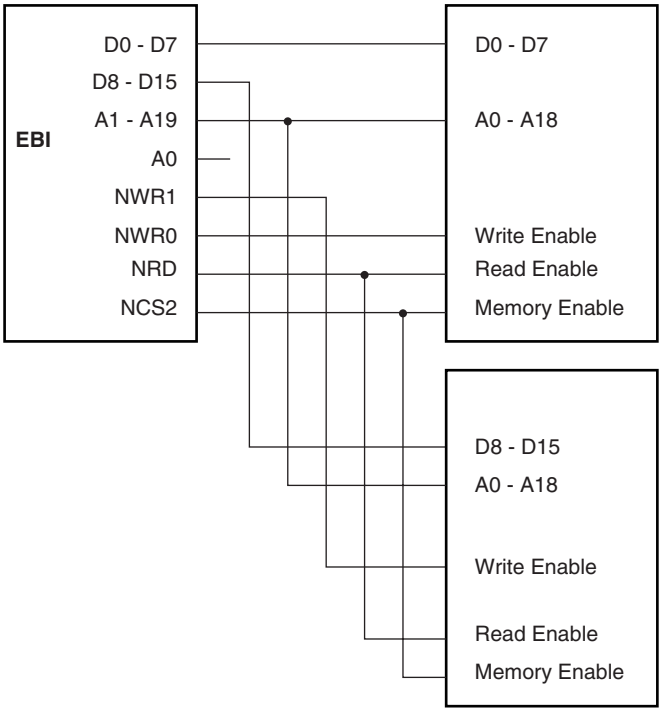
5.2.1 System Peripherals

The External Bus Interface (EBI) controls the external memory and peripheral devices via an 8- or 16-bit data bus and is programmed through the APB. Each chip select line has its own programming register.

The Advanced Power Management Controller (APMC) optimizes power consumption of the product by controlling the clocking elements such as the oscillators and the PLL, system and user peripheral clocks, and the power supplies.

The Advanced Interrupt Controller (AIC) controls the internal interrupt sources from the internal peripherals and the eight external interrupt lines (including the FIQ), to provide an interrupt and/or fast interrupt request to the ARM7TDMI. It integrates an 8-level priority controller and, using the Auto-vectoring feature, reduces the interrupt latency time.

Figure 11-4. Memory Connection for 2 x 8-bit Data Busses



Byte-select Access is used to connect 16-bit devices in a memory page.

- The signal A0/NLB is used as NLB and enables the lower byte for both read and write operations.
- The signal NWR1/NUB is used as NUB and enables the upper byte for both read and write operations.
- The signal NWR0/NWE is used as NWE and enables writing for byte or half word.
- The signal NRD/NOE is used as NOE and enables reading for byte or half word.

Figure 11-5 shows how to connect a 16-bit device with byte and half-word access (e.g. 16-bit SRAM) on NCS2.

Figure 11-5. Connection for a 16-bit Data Bus with Byte and Half-word Access

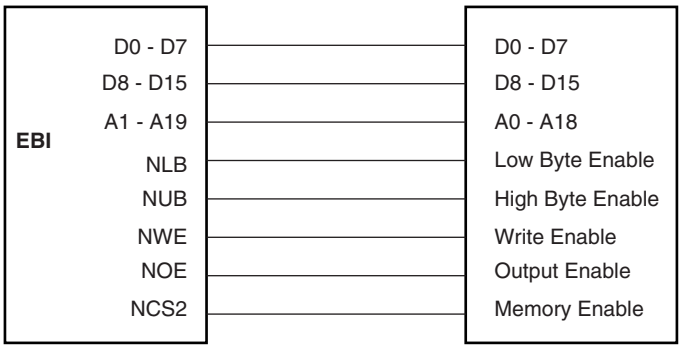
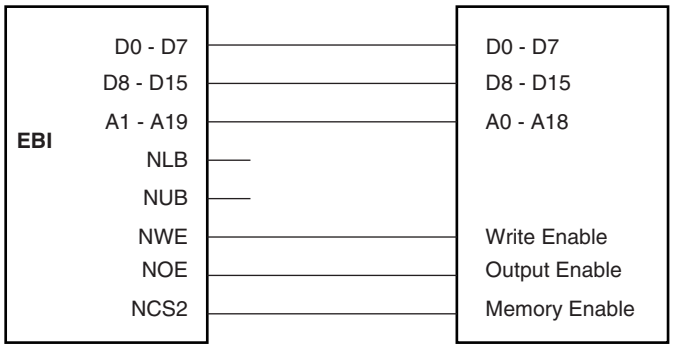


Figure 11-6 shows how to connect a 16-bit device without byte access (e.g. Flash) on NCS2.

Figure 11-6. Connection for a 16-bit Data Bus Without Byte-write Capability.



11.5 Boot on NCS0

Depending on the device and the BMS pin level during the reset, the user can select either an 8-bit or 16-bit external memory device connected on NCS0 as the Boot Memory. In this case, EBI_CSR0 (Chip-select Register 0) is reset at the following configuration for chip select 0:

- 8 wait states (WSE = 1, NWS = 7)
- 8-bit or 16-bit data bus width, depending on BMS

Byte access type and number of data float time are respectively set to Byte-write Access and 0. With a nonvolatile memory interface, any value can be programmed for these parameters.

Before the remap command, the user can modify the chip select 0 configuration, programming the EBI_CSR0 with exact boot memory characteristics. The base address becomes effective after the remap command, but the new number of wait states can be changed immediately. This is useful if a boot sequence needs to be faster.

12.9.3 APMC System Clock Status Register

Register Name: APMC_SCSR

Access Type: Read-only

Reset Value: 0x1

Offset: 0x08

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
–	–	–	–	–	–	–	CPU

- **CPU: System Clock Status Bit**

0 = System Clock is disabled.

1 = System Clock is enabled.

12.9.4 APMC Peripheral Clock Enable Register

Register Name: APMC_PCER

Access Type: Write-only

Offset: 0x10

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	DAC1	DAC0	ADC1
15	14	13	12	11	10	9	8
ADC0	PIOB	PIOA	–	TC5	TC4	TC3	TC2
7	6	5	4	3	2	1	0
TC1	TC0	SPI	US2	US1	US0	–	–

- **Peripheral Clock Enable (per peripheral)**

0 = No effect.

1 = Enables the peripheral clock.

13.3.1 RTC Mode Register

Register Name: RTC_MR

Access: Read/Write

Offset: 0x00

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

- **UPDTIM: Update Request Time Register (Code Label `RTC_UPDTIM`)**

0 = Enables the RTC time counting.

1 = Stops the RTC time counting.

Time counting consists of second, minute and hour counters. Time counters can be programmed once this bit is set.

- **UPDCAL: Update Request Calendar Register (Code Label `RTC_UPDCAL`)**

0 = Disables the RTC calendar counting.

1 = Stops the RTC calendar counting.

Calendar counting consists of day, date, month, year and century counters. Calendar counters can be programmed once this bit is set.

- **TEVSEL: Time Event Selection**

The event which generates the flag TIMEV in RTC_SR (Status Register) depends on the value of TEVSEL.

TEVSEL		Event	Code Label
0	0	Minute change	RTC_TEVSEL_MN_CHG
0	1	Hour change	RTC_TEVSEL_HR_CHG
1	0	Every day at midnight	RTC_TEVSEL_EVDAY_MD
1	1	Every day at noon	RTC_TEVSEL_EVDAY_NOON

- **CEVSEL: Calendar Event Selection**

The event which generates the flag CALEV in RTC_SR depends on the value of CEVSEL.

CEVSEL		Event	Code Label
0	0	Week change (every Monday at time 00:00:00)	RTC_CEVSEL_WEEK_CHG
0	1	Month change (every 01 of each month at time 00:00:00)	RTC_CEVSEL_MONTH_CHG
1	0	Year change (every January 1st at time 00:00:00)	RTC_CEVSEL_YEAR_CHG
1	1	Reserved	–

13.3.10 RTC Interrupt Disable Register

Register Name: RTC_IDR

Access Type: Write-only

Offset: 0x24

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
–	–	–	CALEV	TIMEV	SEC	ALARM	ACKUPD

- **ACKUPD: Acknowledge Update Interrupt Disable (Code Label `RTC_ACKUPD`)**

0 = No effect.

1 = The acknowledge for update interrupt is disabled.

- **ALARM: Alarm Interrupt Disable (Code Label `RTC_ALARM`)**

0 = No effect.

1 = The alarm interrupt is disabled.

- **SEC: Second Event Interrupt Disable (Code Label `RTC_SEC`)**

0 = No effect.

1 = The second periodic interrupt is disabled.

- **TIMEV: Time Event Interrupt Disable (Code Label `RTC_TIMEV`)**

0 = No effect.

1 = The selected time event interrupt is disabled.

- **CALEV: Calendar Event Interrupt Disable (Code Label `RTC_CALEV`)**

0 = No effect.

1 = The selected calendar event interrupt is disabled.

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).

16.8 PIO Connection Tables

Table 16-1. PIO Controller A Connection Table

PIO Controller		Peripheral					
Bit Number	Port Name	Port Name	Signal Description	Signal Direction	OFF Value[1]	Reset State	Pin Number
0	PA0	TCLK3	Timer 3 Clock signal	Input	0	PIO Input	66
1	PA1	TIOA3	Timer 3 Signal A	Bi-directional	0	PIO Input	67
2	PA2	TIOB3	Timer 3 Signal B	Bi-directional	0	PIO Input	68
3	PA3	TCLK4	Timer 4 Clock signal	Input	0	PIO Input	69
4	PA4	TIOA4	Timer 4 Signal A	Bi-directional	0	PIO Input	70
5	PA5	TIOB4	Timer 4 Signal B	Bi-directional	0	PIO Input	71
6	PA6	TCLK5	Timer 5 Clock signal	Input	0	PIO Input	72
7	PA7	TIOA5	Timer 5 Signal A	Bi-directional	0	PIO Input	75
8	PA8	TIOB5	Timer 5 Signal B	Bi-directional	0	PIO Input	76
9	PA9	IRQ0	External Interrupt 0	Input	0	PIO Input	77
10	PA10	IRQ1	External Interrupt 1	Input	0	PIO Input	78
11	PA11	IRQ2	External Interrupt 2	Input	0	PIO Input	79
12	PA12	IRQ3	External Interrupt 3	Input	0	PIO Input	80
13	PA13	FIQ	Fast Interrupt	Input	0	PIO Input	81
14	PA14	SCK0	USART 0 Clock signal	Bi-directional	0	PIO Input	82
15	PA15	TXD0	USART 0 transmit data	Output	–	PIO Input	83
16	PA16	RXD0	USART 0 receive data	Input	0	PIO Input	84
17	PA17	SCK1	USART 1 Clock signal	Bi-directional	0	PIO Input	85
18	PA18	TXD1	USART 1 transmit data	Output	–	PIO Input	86
19	PA19	RXD1	USART 1 receive data	Input	0	PIO Input	91
20	PA20	SCK2	USART 2 Clock signal	Bi-directional	0	PIO Input	92
21	PA21	TXD2	USART 2 transmit data	Output	–	PIO Input	93
22	PA22	RXD2	USART 2 receive data	Input	0	PIO Input	94
23	PA23	SPCK	SPI Clock signal	Bi-directional	0	PIO Input	95
24	PA24	MISO	SPI Master In Slave Out	Bi-directional	0	PIO Input	96
25	PA25	MOSI	SPI Master Out Slave In	Bi-directional	0	PIO Input	97
26	PA26	NPCS0	SPI Peripheral Chip Select 0	Bi-directional	1	PIO Input	98
27	PA27	NPCS1	SPI Peripheral Chip Select 1	Output	–	PIO Input	99
28	PA28	NPCS2	SPI Peripheral Chip Select 2	Output	–	PIO Input	100
29	PA29	NPCS3	SPI Peripheral Chip Select 3	Output	–	PIO Input	101
30	–	–	–	–	–	–	–
31	–	–	–	–	–	–	–

Note: 1. The OFF value is the default level seen on the peripheral input when the PIO line is enabled.

- **ARCH: Chip Architecture**

Code of Architecture: Two BCD digits

ARCH	Selected ARCH	Code Label: SF_ARCH
0110 0011	AT91x63yyy	SF_ARCH_AT91x63
0100 0000	AT91x40yyy	SF_ARCH_AT91x40
0101 0101	AT91x55yyy	SF_ARCH_AT91x55

- **NVPTYP: Nonvolatile Program Memory Type**

NVPTYP			Type	Code Label: SF_NVPTYP
0	0	1	“M” Series or “F” Series	SF_NVPTYP_M
1	0	0	“R” Series	SF_NVPTYP_R

Note: All other codes are reserved.

- **EXT: Extension Flag (Code Label SF_EXT)**

0 = Chip ID has a single-register definition without extensions

1 = An extended Chip ID exists (to be defined in the future).

17.2.2 Chip ID Extension Register

Register Name: SF_EXID

Access Type: Read-only

Offset: 0x04

This register is reserved for future use. It will be defined when needed.

- **TIMEOUT: Enable Time-out Interrupt (Code Label `US_TIMEOUT`)**

0 = No effect.

1 = Enables Reception Time-out Interrupt.

- **TXEMPTY: Enable TXEMPTY Interrupt (Code Label `US_TXEMPTY`)**

0 = No effect.

1 = Enables TXEMPTY Interrupt.

18.10.6 USART Channel Status Register

Name: US_CSR
Access Type: Read-only
Reset: 0x18
Offset: 0x14

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

- **RXRDY: Receiver Ready (Code Label `US_RXRDY`)**

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 (Code Label `US_TXRDY`)**

0 = US_THR contains a character waiting to be transferred to the Transmit Shift Register, or an STTBRK command has been requested.

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

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

- **RXBRK: Break Received/End of Break (Code Label `US_RXBRK`)**

0 = No Break Received nor End of Break detected since the last “Reset Status Bits” command in the Control Register.

1 = Break Received or End of Break detected since the last “Reset Status Bits” command in the Control Register.

- **ENDRX: End of Receive Transfer (Code Label `US_ENDRX`)**

0 = The End of Transfer signal from the Peripheral Data Controller channel dedicated to the receiver is inactive.

1 = The End of Transfer signal from the Peripheral Data Controller channel dedicated to the receiver is active.

- **ENDTX: End of Transmit Transfer (Code Label `US_ENDTX`)**

0 = The End of Transfer signal from the Peripheral Data Controller channel dedicated to the transmitter is inactive.

1 = The End of Transfer signal from the Peripheral Data Controller channel dedicated to the transmitter is active.

- **OVRE: Overrun Error (Code Label `US_OVRE`)**

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 (Code Label `US_FRAME`)**

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.

18.10.12 USART Receive Pointer Register

Name: US_RPR
Access Type: Read/Write
Reset State: 0
Offset: 0x30

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.

18.10.13 USART Receive Counter Register

Name: US_RCR
Access Type: Read/Write
Reset State: 0
Offset: 0x34

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							

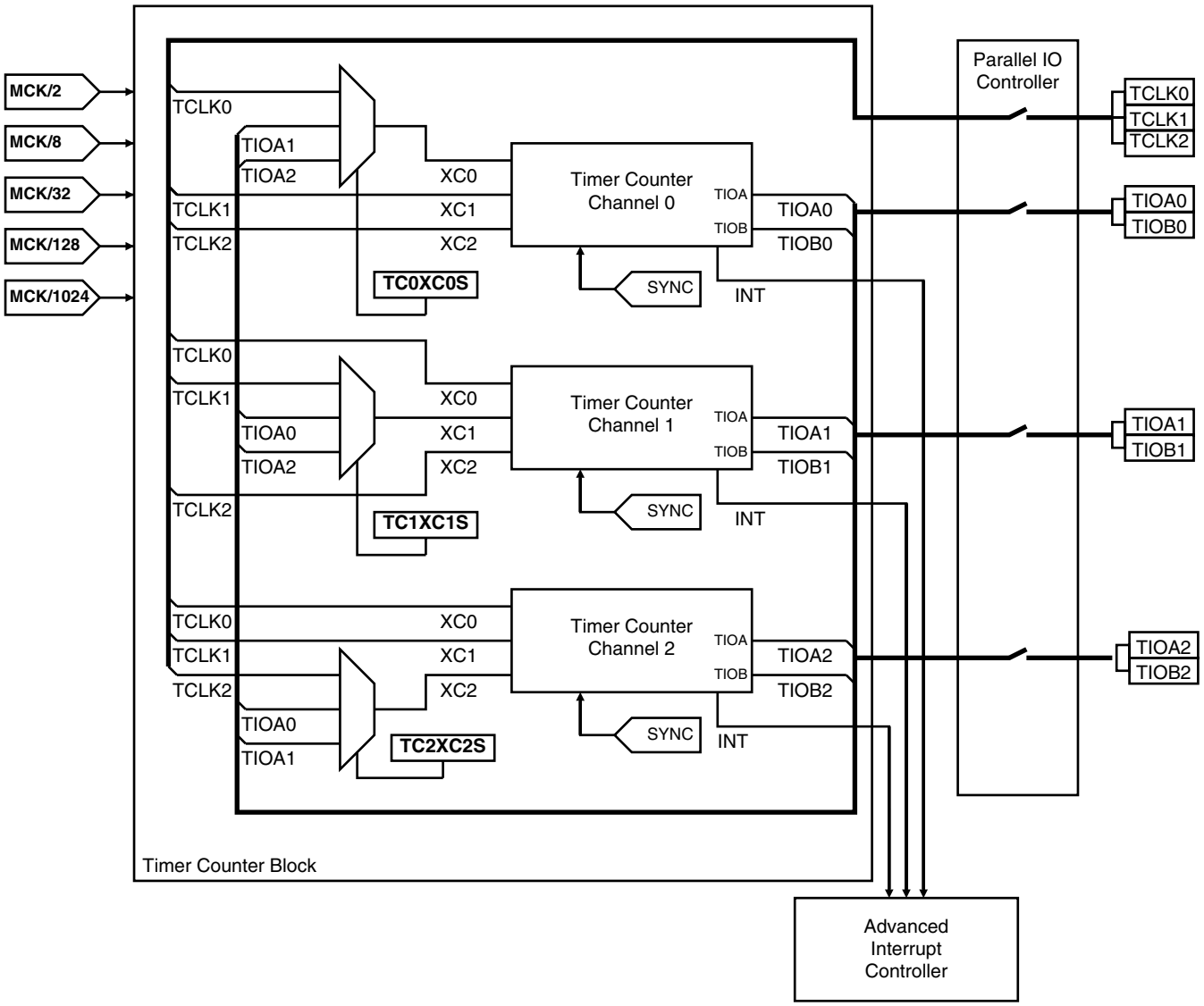
- **RXCTR: Receive Counter**

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.

Figure 19-1. TC Block Diagram



19.5 TC User Interface

TC Block 0 Base Address: 0xFFFD0000 (Code Label `TCB0_BASE`)
TC Block 1 Base Address: 0xFFFD4000 (Code Label `TCB1_BASE`)

Table 19-2. TC Global Register Mapping

Offset	Channel/Register	Name	Access	Reset State
0x00	TC Channel 0	See Table 19-3		
0x40	TC Channel 1	See Table 19-3		
0x80	TC Channel 2	See Table 19-3		
0xC0	TC Block Control Register	TC_BCR	Write-only	–
0xC4	TC Block Mode Register	TC_BMR	Read/Write	0

TC_BCR (Block Control Register) and TC_BMR (Block Mode Register) control the TC block. TC Channels are controlled by the registers listed in Table 19-3. The offset of each of the Channel registers in Table 19-3 is in relation to the offset of the corresponding channel as mentioned in Table 19-2.

Table 19-3. TC Channel Register Mapping

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

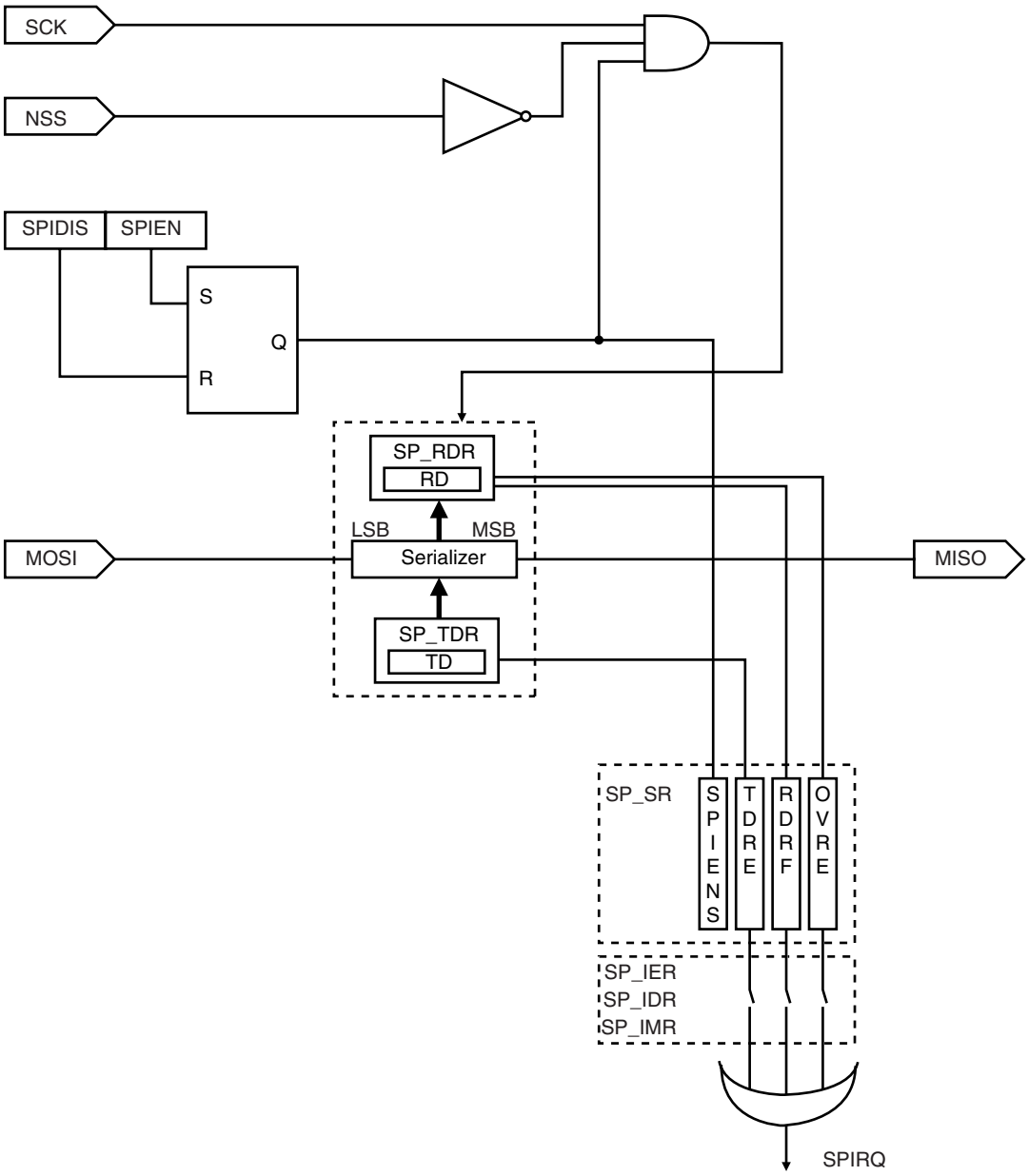
Note: 1. Read-only if WAVE = 0

20.3 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 SP_CSR0 are used to define the transfer characteristics. The other Chip Select Registers are not used in slave mode.

Figure 2. SPI in Slave Mode



20.7.3 SPI Receive Data Register

Register Name: SP_RDR
Access Type: Read-only
Reset State: 0
Offset: 0x08

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 (Code Label `SP_RD`)**

Data received by the SPI Interface is stored in this register right-justified. Unused bits read zero.

- **PCS: Peripheral Chip Select Status**

In Master Mode only, these bits indicate the value on the NPCS pins at the end of a transfer. Otherwise, these bits read zero.

- **SLEEP: Sleep Mode**

SLEEP	Selected SLEEP	Code Label
0	Normal Mode	ADC_NORMAL_MODE
1	Sleep Mode	ADC_SLEEP_MODE

- **PRESCAL: Prescaler Rate Selection (`ADC_PRESCAL`)**

This field defines the conversion clock in function of the Master Clock (MCK):

$$ADCClock = MCK / ((PRESCAL + 1) \times 2)$$

The ADC clock range is between MCK/2 (PRESCAL = 0) and MCK /128 (PRESCAL = 63). PRESCAL must be programmed in order to provide an ADC clock frequency according to the parameters given in the AT91M55800A Electrical Datasheet, literature number 1727.

Table 23-1. JTAG Boundary-scan Register (Continued)

Bit Number	Pin Name	Pin Type	Associated BSR Cells
129	PA9/IRQ0	IN/OUT	OUTPUT
128			INPUT
127			CTRL
126	PA8/TIOB5	IN/OUT	OUTPUT
125			INPUT
124			CTRL
123	PA7/TIOA5	IN/OUT	OUTPUT
122			INPUT
121			CTRL
120	PA6/CLK5	IN/OUT	OUTPUT
119			INPUT
118			CTRL
117	PA5/TIOB4	IN/OUT	OUTPUT
116			INPUT
115			CTRL
114	PA4/TIOA4	IN/OUT	OUTPUT
113			INPUT
112			CTRL
111	PA3/TCLK4	IN/OUT	OUTPUT
110			INPUT
109			CTRL
108	PA2/TIOB3	IN/OUT	OUTPUT
107			INPUT
106			CTRL
105	PA1/TIOA3	IN/OUT	OUTPUT
104			INPUT
103			CTRL
102	PA0/TCLK3	IN/OUT	OUTPUT
101			INPUT
100			CTRL
99	PB27/TIOB2	IN/OUT	OUTPUT
98			INPUT
97			CTRL
96	PB26/TIOA2	IN/OUT	OUTPUT

In other cases, the following erroneous behavior occurs:

– 32-bit read accesses are not managed correctly and the first 16-bit data sampling takes into account only the standard wait states. 16- and 8-bit accesses are not affected.

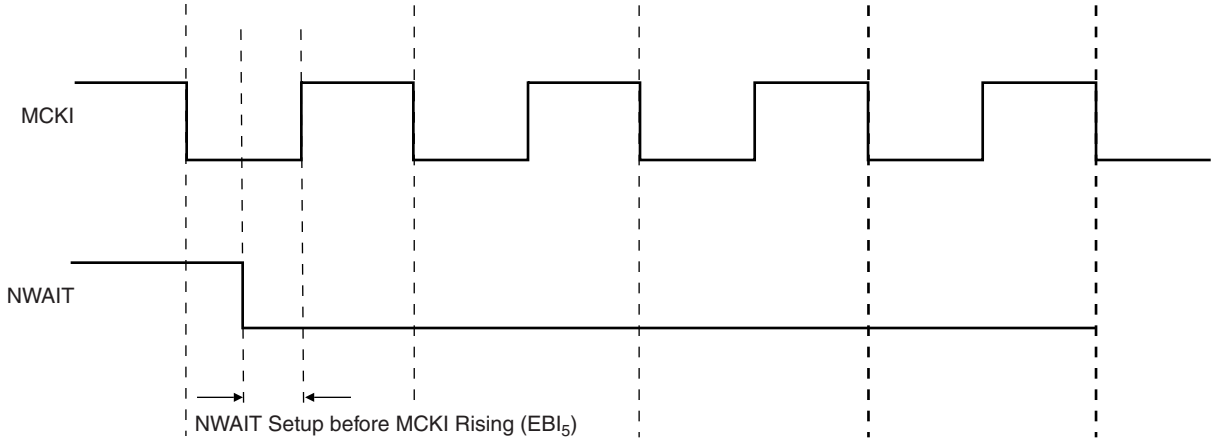
– During write accesses of any type, the NWE rises on the rising edge of the last cycle as defined by the programmed number of wait states. However, NWAIT assertion does affect the length of the total access. Only the NWE pulse length is inaccurate.

At maximum speed, asserting the NWAIT in the first access cycle is not possible, as the sum of the timings “MCKI Falling to Chip Select” and “NWAIT setup to MCKI rising” are generally higher than one half of a clock period. This leads to using at least one standard wait state. However, this is not sufficient except to perform byte or half-word read accesses. Word and write accesses require at least two standard wait states.

The following waveforms further explain the issue:

If the NWAIT setup time is satisfied on the first rising edge of MCKI, the behavior is accurate. The EBI operations are not affected when the NWAIT rises.

Figure 27-1. NWAIT Rising



If the NWAIT setup time is satisfied on the following edges of MCKI and if at least one standard wait state remains to be executed, the behavior is accurate. In the following example, the number of standard wait states is two. The NWAIT setup time on the second rising edge of MCKI must be met.