Features

- Utilizes the ARM7TDMI™ ARM Thumb Processor Core
  - High-performance 32-bit RISC Architecture
  - High-density 16-bit Instruction Set
  - Leader in MIPS/Watt
  - Embedded ICE (In-Circuit Emulation)
- 8K Bytes Internal SRAM
- Fully-programmable External Bus Interface (EBI)
  - Maximum External Address Space of 128M Bytes
  - Eight Chip Selects
  - Software Programmable 8/16-bit External Databus
- 8-level Priority, Individually Maskable, Vectored Interrupt Controller
  - Seven External Interrupts, Including a High-priority, Low-latency Interrupt Request
- Fifty-eight Programmable I/O Lines
- 6-channel 16-bit Timer/Counter
  - Six External Clock Inputs and Two Multi-purpose I/O Pins per Channel
- Three USARTs
- Master/Slave SPI Interface
  - 8-bit to 16-bit Programmable Data Length
  - Four External Slave Chip Selects
- Programmable Watchdog Timer
- 8-channel 10-bit ADC
- 2-channel 10-bit DAC
- Clock Generator with On-chip Main Oscillator and PLL for Multiplication
  - 3 to 20 MHz Frequency Range Main Oscillator
- Real-time Clock with On-chip 32 kHz Oscillator
  - Battery Backup Operation and External Alarm
- 8-channel Peripheral Data Controller for USARTs and SPIs
- Advanced Power Management Controller (APMC)
  - Normal, Wait, Slow, Standby and Power-down modes
- IEEE 1149.1 JTAG Boundary-scan on all Digital Pins
- Fully Static Operation: 0 Hz to 33 MHz
- 2.7V to 3.6V Core Operating Range
- 2.7V to 5.5V I/O Operating Range
- 2.7V to 3.6V Analog Operating Range
- 1.8V to 3.6V Backup Battery Operating Range
- 2.7V to 3.6V Oscillator and PLL Operating Range
- -40°C to +85°C Temperature Range
- Available in a 176-lead TQFP or 176-ball BGA Package

Description

The AT91M55800A is a member of the Atmel AT91 16/32-bit microcontroller family, which is 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, a large number of internally banked registers result in very fast exception handling, making the device ideal for real-time control applications.

The fully programmable External Bus Interface provides a direct connection to off-chip memory in as fast as one clock cycle for a read or write operation. An eight-level priority vectored interrupt controller in conjunction with the peripheral data controller significantly improve the real-time performance of the device.

The device is manufactured using Atmel's high-density CMOS technology. By combining the ARM7TDMI processor core with an on-chip SRAM, a wide range of peripheral functions, analog interfaces and low-power oscillators on a monolithic chip, the Atmel AT91M55800A is a powerful microcontroller that provides a highly-flexible and cost-effective solution to many ultra low-power applications.

AT91 ARM® Thumb® Microcontrollers

AT91M55800A

Rev. 1745C–ATARM–12/02

Pin Configurations

Table 1. Pin Configuration for 176-lead TQFP Package

Pin	AT91M55800A	Pin	AT91M55800A	Pin	AT91M55800A	Pin	AT91M55800A
1	GND	45	GND	89	GND	133	GND
2	GND	46	GND	90	GND	134	GND
3	NCS0	47	D8	91	PA19/RXD1	135	NCS4
4	NCS1	48	D9	92	PA20/SCK2	136	NCS5
5	NCS2	49	D10	93	PA21/TXD2	137	NCS6
6	NCS3	50	D11	94	PA22/RXD2	138	NCS7
7	NLB/A0	51	D12	95	PA23/SPCK	139	PB0
8	A1	52	D13	96	PA24/MISO	140	PB1
9	A2	53	D14	97	PA25/MOSI	141	PB2
10	A3	54	D15	98	PA26/NPCS0/NSS	142	PB3/IRQ4
11	A4	55	PB19/TCLK0	99	PA27/NPCS1	143	PB4/IRQ5
12	A5	56	PB20/TIOA0	100	PA28/NPCS2	144	PB5
13	A6	57	PB21/TIOB0	101	PA29/NPCS3	145	PB6/AD0TRIG
14	A7	58	PB22/TCLK1	102	VDDIO	146	PB7/AD1TRIG
15	VDDIO	59	VDDIO	103	GND	147	VDDIO
16	GND	60	GND	104	VDDPLL	148	GND
17	A8	61	PB23/TIOA1	105	XIN	149	PB8
18	A9	62	PB24/TIOB1	106	XOUT	150	PB9
19	A10	63	PB25/TCLK2	107	GNDPLL	151	PB10
20	A11	64	PB26/TIOA2	108	PLLRC	152	PB11
21	A12	65	PB27/TIOB2	109	VDDBU[2]	153	PB12
22	A13	66	PA0/TCLK3	110	XIN32[2]	154	PB13
23	A14	67	PA1/TIOA3	111	XOUT32[2]	155	PB14
24	A15	68	PA2/TIOB3	112	NRSTBU[2]	156	PB15
25	A16	69	PA3/TCLK4	113	GNDBU[2]	157	PB16
26	A17	70	PA4/TIOA4	114	WAKEUP[2]	158	PB17
27	A18	71	PA5/TIOB4	115	SHDN[2]	159	NWDOVF
28	A19	72	PA6/TCLK5	116	GNDBU[2]	160	MCKO
29	VDDIO	73	VDDIO	117	VDDA[1]	161	VDDIO
30	GND	74	GND	118	AD0[1]	162	GND
31	A20	75	PA7/TIOA5	119	AD1[1]	163	PB18/BMS
32	A21	76	PA8/TIOB5	120	AD2[1]	164	JTAGSEL
33	A22	77	PA9/IRQ0	121	AD3[1]	165	TMS
34	A23	78	PA10/IRQ1	122	AD4[1]	166	TDI
35	D0	79	PA11/IRQ2	123	AD5[1]	167	TDO
36	D1	80	PA12/IRQ3	124	AD6[1]	168	TCK
37	D2	81	PA13/FIQ	125	AD7[1]	169	NTRST
38	D3	82	PA14/SCK0	126	ADVREF[1]	170	NRST
39	D4	83	PA15/TXD0	127	DAVREF[1]	171	NWAIT
40	D5	84	PA16/RXD0	128	DA0[1]	172	NOE/NRD
41	D6	85	PA17/SCK1	129	DA1[1]	173	NWE/NWR0
42	D7	86	PA18/TXD1/NTRI	130	GNDA[1]	174	NUB/NWR1
43	VDDCORE	87	VDDCORE	131	VDDCORE	175	VDDCORE
44	VDDIO	88	VDDIO	132	VDDIO	176	VDDIO

Notes: 1. Analog pins
2. Battery backup pins

Table 2. Pin Configuration for 176-ball BGA Package

Pin	AT91M55800A
A1	NCS1
A2	NWAIT
A3	NRST
A4	NTRST
A5	PB18/BMS
A6	NWDOVF
A7	PB16
A8	PB12
A9	PB10
A10	PB9
A11	PB8
A12	NCS7
A13	NCS6
A14	GND
A15	DAVREF
B1	NCS2
B2	NUB/NWR1
B3	NWE/NWR0
B4	NOE/NRD
B5	TD0
B6	TDI
B7	PB17
B8	PB11
B9	PB7/AD1TRIG
B10	PB3/IRQ4
B11	PB2
B12	NCS5
B13	NCS4
B14	DA1
B15	GNDA

Pin	AT91M55800A
C1	A0/NLB
C2	NCS0
C3	VDDIO
C4	VDDCORE
C5	TMS
C6	VDDIO
C7	MCK0
C8	PB13
C9	PB6/AD0TRIG
C10	VDDIO
C11	PB4/IRQ5
C12	PB0
C13	VDDIO
C14	DA0
C15	ADVREF
D1	A2
D2	A1
D3	NCS3
D4	GND
D5	TCK
D6	JTAGSEL
D7	GND
D8	PB15
D9	PB14
D10	PB5
D11	PB1
D12	GND
D13	VDDCORE
D14	AD7
D15	VDDA

Pin	AT91M55800A
E1	A4
E2	A3
E3	A5
E4	GND
E5	–
E6	–
E7	–
E8	–
E9	–
E10	–
E11	–
E12	AD6
E13	AD5
E14	NRSTBU
E15	GNDBU
F1	A10
F2	A7
F3	VDDIO
F4	A6
F5	–
F6	–
F7	–
F8	–
F9	–
F10	–
F11	–
F12	GND
F13	AD4
F14	VDDBU
F15	XOUT32

Pin	AT91M55800A
G1	A12
G2	A9
G3	A8
G4	GND
G5	–
G6	–
G7	–
G8	–
G9	–
G10	–
G11	–
G12	AD3
G13	AD2
G14	GND
G15	XIN32
H1	A15
H2	A14
H3	A13
H4	A11
H5	–
H6	–
H7	–
H8	–
H9	–
H10	–
H11	–
H12	AD1
H13	AD0
H14	WAKEUP
H15	GND

Table 2. Pin Configuration for 176-ball BGA Package (Continued)

Pin	AT91M55800A
J1	A17
J2	A18
J3	VDDIO
J4	A16
J5	–
J6	–
J7	–
J8	–
J9	–
J10	–
J11	–
J12	PA29/NPCS3
J13	SHDN
J14	VDDPLL
J15	PLLRC
K1	A19
K2	A22
K3	A21
K4	GND
K5	–
K6	–
K7	–
K8	–
K9	–
K10	–
K11	–
K12	PA28/NPCS2
K13	VDDIO
K14	PA27/NPCS1
K15	GNDPLL

Pin	AT91M55800A
L1	A20
L2	A23
L3	D0
L4	D1
L5	–
L6	–
L7	–
L8	–
L9	–
L10	–
L11	–
L12	PA25/MOSI
L13	PA22RXD2
L14	PA26/NPCS0/NSS
L15	XOUT
M1	D2
M2	D3
M3	VDDCORE
M4	GND
M5	GND
M6	PB21/TIOB0
M7	GND
M8	PB27/TIOB2
M9	PA0/TCLK3
M10	GND
M11	PA23/SPCK
M12	GND
M13	PA21/TXD2
M14	PA24/MISO
M15	XIN

Pin	AT91M55800A
N1	D4
N2	D6
N3	VDDIO
N4	D14
N5	PB19/TCLK0
N6	VDDIO
N7	PB25/TCLK2
N8	PA1/TIOA3
N9	VDDIO
N10	PA8/TIOB5
N11	PA9/IRQ0
N12	VDDCORE
N13	VDDIO
N14	PA19/RXD1
N15	GND
P1	D5
P2	D7
P3	D8
P4	D9
P5	D15
P6	PB22/TCLK1
P7	PB26/TIOA2
P8	PA2/TIOB3
P9	PA7/TIOA5
P10	PA10/IRQ1
P11	PA11/IRQ2
P12	PA13/FIQ
P13	PA17SCK1
P14	PA18/TXD1/NTRI
P15	PA20/SCK2

Pin	AT91M55800A
R1	D10
R2	D11
R3	D12
R4	D13
R5	PB20/TIOA0
R6	PB23/TIOA1
R7	PB24/TIOB1
R8	PA3/TCLK4
R9	PA4/TIOA4
R10	PA5/TIOB4
R11	PA6/TCLK5
R12	PA12/IRQ3
R13	PA14/SCK0
R14	PA15/TXD0
R15	PA16/RXD0

Figure 1. 176-lead TQFP Pinout

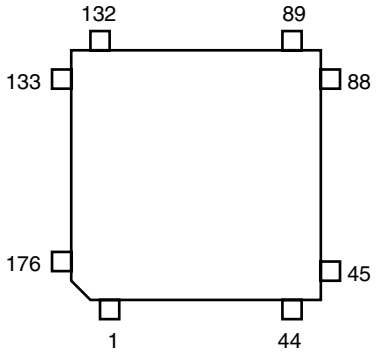
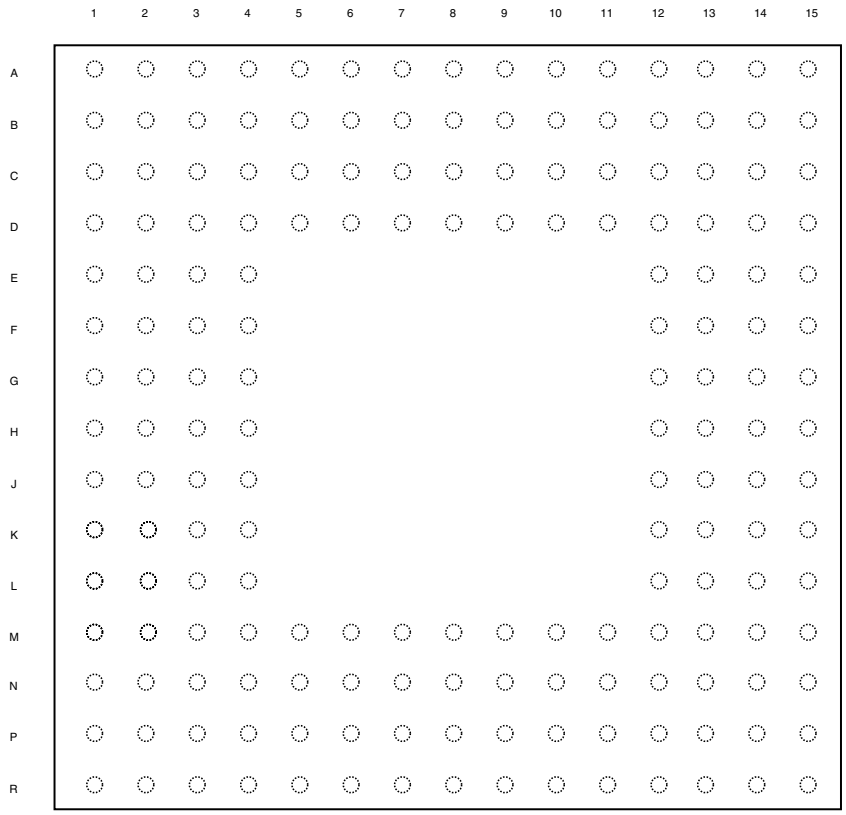


Figure 2. 176-ball BGA Pinout



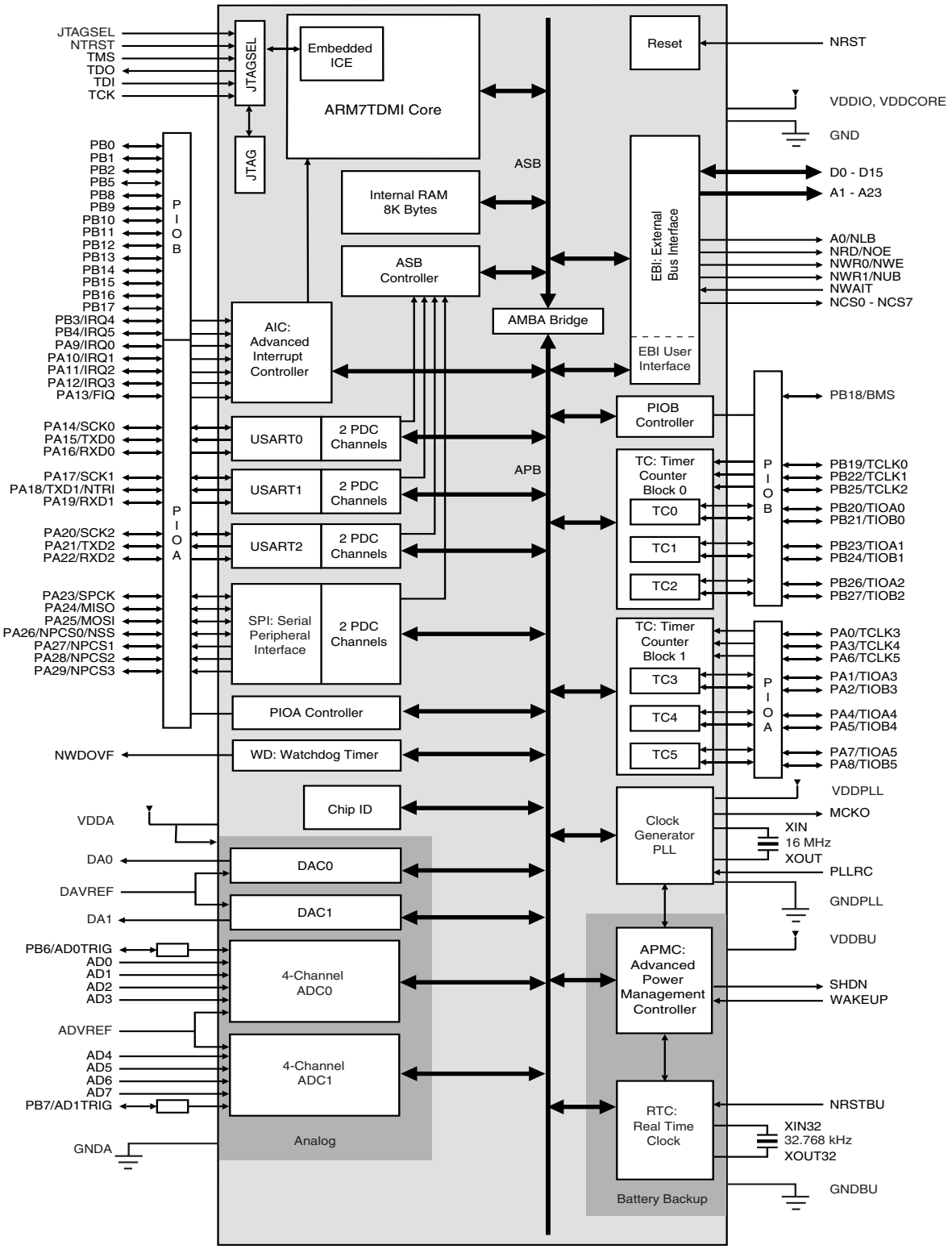
Pin Description

Module	Name	Function	Type	Active Level	Comments
EBI	A0 - A23	Address bus	Output	–	
	D0 - D15	Data bus	I/O	–	
	NCS0 - NCS7	Chip select	Output	Low	
	NWR0	Lower byte 0 write signal	Output	Low	Used in Byte-write option
	NWR1	Lower byte 1 write signal	Output	Low	Used in Byte-write option
	NRD	Read signal	Output	Low	Used in Byte-write option
	NWE	Write enable	Output	Low	Used in Byte-select option
	NOE	Output enable	Output	Low	Used in Byte-select option
	NUB	Upper byte-select	Output	Low	Used in Byte-select option
	NLB	Lower byte-select	Output	Low	Used in Byte-select option
	NWAIT	Wait input	Input	Low	
	BMS	Boot mode select	Input	–	Sampled during reset
AIC	IRQ0 - IRQ5	External interrupt request	Input	–	PIO-controlled after reset
	FIQ	Fast external interrupt request	Input	–	PIO-controlled after reset
Timer	TCLK0 - TCLK5	Timer external clock	Input	–	PIO-controlled after reset
	TIOA0 - TIOA5	Multipurpose timer I/O pin A	I/O	–	PIO-controlled after reset
	TIOB0 - TIOB5	Multipurpose timer I/O pin B	I/O	–	PIO-controlled after reset
USART	SCK0 - SCK2	External serial clock	I/O	–	PIO-controlled after reset
	TXD0 - TXD2	Transmit data output	Output	–	PIO-controlled after reset
	RXD0 - RXD2	Receive data input	Input	–	PIO-controlled after reset
SPI	SPCK	SPI clock	I/O	–	PIO-controlled after reset
	MISO	Master in slave out	I/O	–	PIO-controlled after reset
	MOSI	Master out slave in	I/O	–	PIO-controlled after reset
	NSS	Slave select	Input	Low	PIO-controlled after reset
	NPCS0 - NPCS3	Peripheral chip select	Output	Low	PIO-controlled after reset
PIO	PA0 - PA29	Parallel I/O port A	I/O	–	Input after reset
	PB0 - PB27	Parallel I/O port B	I/O	–	Input after reset
WD	NWDOVF	Watchdog timer overflow	Output	Low	Open drain
ADC	AD0-AD7	Analog input channels 0 - 7	Analog in	–	
	AD0TRIG	ADC0 external trigger	Input	–	PIO-controlled after reset
	AD1TRIG	ADC1 external trigger	Input	–	PIO-controlled after reset
	ADVREF	Analog reference	Analog ref	–	
DAC	DA0 - DA1	Analog output channels 0 - 1	Analog out	–	
	DAVREF	Analog reference	Analog ref	–	

Pin Description (Continued)

Module	Name	Function	Type	Active Level	Comments
Clock	XIN	Main oscillator input	Input	–	
	XOUT	Main oscillator output	Output	–	
	PLLRC	RC filter for PLL	Input	–	
	XIN32	32 kHz oscillator input	Input	–	
	XOUT32	32 kHz oscillator output	Output	–	
	MCKO	System clock	Output	–	
APMC	WAKEUP	Wakeup request	Input	–	
	SHDN	Shutdown request	Output	–	Tri-state after backup reset
Reset	NRST	Hardware reset input	Input	Low	Schmidt trigger
	NRSTBU	Hardware reset input for battery part	Input	Low	Schmidt trigger
	NTRI	Tri-state mode select	Input	Low	Sampled during reset
JTAG/ICE	JTAGSEL	Selects between ICE and JTAG mode	Input	–	
	TMS	Test mode select	Input	–	Schmidt trigger, internal pull-up
	TDI	Test data input	Input	–	Schmidt trigger, internal pull-up
	TDO	Test data output	Output	–	
	TCK	Test clock	Input	–	Schmidt trigger, internal pull-up
	NTRST	Test reset input	Input	Low	Schmidt trigger, internal pull-up
Power	VDDA	Analog power	Analog pwr	–	
	GNDA	Analog ground	Analog gnd	–	
	VDDBU	Power backup	Power	–	
	GNDBU	Ground backup	Ground	–	
	VDDCORE	Digital core power	Power	–	
	VDDIO	Digital I/O power	Power	–	
	VDDPLL	Main oscillator and PLL power	Power	–	
	GND	Digital ground	Ground	–	
	GNDPLL	PLL ground	Ground	–	

Block Diagram



Architectural Overview

The AT91M55800A microcontroller integrates an ARM7TDMI with its embedded ICE 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.

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.

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.

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.

The Real-time Clock (RTC) peripheral is designed for very low power consumption, and combines a complete time-of-day clock with alarm and a two-hundred year Gregorian calendar, complemented by a programmable periodic interrupt.

The Parallel Input/Output Controllers (PIOA and PIOB) control the 58 I/O lines. They enable the user to select specific pins for on-chip peripheral input/output functions, and

general-purpose input/output signal pins. The PIO controllers can be programmed to detect an interrupt on a signal change from each line.

The Watchdog (WD) can be used to prevent system lock-up if the software becomes trapped in a deadlock.

The Special Function (SF) module integrates the Chip ID and Reset Status registers.

User Peripherals

Three USARTs, independently configurable, enable communication at a high baud rate in synchronous or asynchronous mode. The format includes start, stop and parity bits and up to 8 data bits. Each USART also features a Timeout and a Time Guard Register, facilitating the use of the two dedicated Peripheral Data Controller (PDC) channels.

The six 16-bit Timer/Counters (TC) are highly programmable and support capture or waveform modes. Each TC channel can be programmed to measure or generate different kinds of waves, and can detect and control two input/output signals. Each TC also has three external clock signals.

The SPI provides communication with external devices in master or slave mode. It has four external chip selects which can be connected to up to 15 devices. The data length is programmable, from 8- to 16-bits.

The two identical 4-channel 10-bit analog-to-digital converters (ADC) are based on a Successive Approximation Register (SAR) approach.

Associated Documentation

Information	Document Title	Literature Number
Internal architecture of processor ARM/Thumb instruction sets Embedded in-circuit-emulator	ARM7TDMI (Thumb) Datasheet	0673B
Mapping Peripheral operation Peripheral user interface	AT91M55800A Datasheet	1745A
Mechanical characteristics Ordering information	AT91M55800A Summary Datasheet	1745AS
Timings DC Characteristics	AT91M55800A Electrical Characteristics Datasheet	1727A

Product Overview

Power Supplies

The AT91M55800A has 5 kinds of power supply pins:

- VDDCORE pins, which power the chip core
- VDDIO pins, which power the I/O Lines
- VDDPLL pins, which power the oscillator and PLL cells
- VDDA pins, which power the analog peripherals ADC and DAC
- VDDBU pins, which power the RTC, the 32768 Hz oscillator and the Shut-down Logic of the APMC

VDDIO and VDDCORE are separated to permit the I/O lines to be powered with 5V, thus resulting in full TTL compliance.

The following ground pins are provided:

- GND for both VDDCORE and VDDIO
- GNDPLL for VDDPLL
- GNDA for VDDA
- GNDBU for VDDBU

All of these ground pins must be connected to the same voltage (generally the board electric ground) with wires as short as possible. GNDPLL, GNDA and GNDBU are provided separately in order to allow the user to add a decoupling capacitor directly between the power and ground pads. In the same way, the PLL filter resistor and capacitors must be connected to the device and to GNDBU with wires as short as possible. Also, the main oscillator crystal and the 32768 Hz crystal external load capacitances must be connected respectively to GNDPLL and to GNDBU with wires as short as possible.

The main constraints applying to the different voltages of the device are:

- VDDBU must be lower than or equal to VDDCORE
- VDDA must be higher than or equal to VDDCORE
- VDDCORE must be lower than or equal to VDDIO

The nominal power combinations supported by the AT91M55800A are described in the following table:

Table 3. Nominal Power Combinations

VDDIO	VDDCORE	VDDA	VDDPLL	VDDBU	Maximum Operating Frequency
3V	3V	3V	3V	3V	33 MHz
3.3V	3.3V	3.3V	3.3V	3.3V	33 MHz
5V	3.3V	3.3V	3.3V	3.3V	33 MHz

Input/Output Considerations

After the reset, the peripheral I/Os are initialized as inputs to provide the user with maximum flexibility. It is recommended that in any application phase, the inputs to the AT91M55800A microcontroller be held at valid logic levels to minimize the power consumption.

Master Clock

Master Clock is generated in one of the following ways, depending on programming in the APMC registers:

- From the 32768 Hz low-power oscillator that clocks the RTC
- The on-chip main oscillator together with a PLL generate a software-programmable main clock in the 500 Hz to 33 MHz range. The main oscillator can be bypassed to allow the user to enter an external clock signal.

The Master Clock (MCK) is also provided as an output of the device on the pin MCKO, whose state is controlled by the APMC module.

Reset

Reset restores the default states of the user interface registers (defined in the user interface of each peripheral), and forces the ARM7TDMI to perform the next instruction fetch from address zero. Aside from the program counter, the ARM7TDMI registers do not have defined reset states.

NRST Pin

NRST is active low-level input. It is asserted asynchronously, but exit from reset is synchronized internally to the MCK. At reset, the source of MCK is the Slow Clock (32768 Hz crystal), and the signal presented on MCK must be active within the specification for a minimum of 10 clock cycles up to the rising edge of NRST, to ensure correct operation.

Watchdog Reset

The watchdog can be programmed to generate an internal reset. In this case, the reset has the same effect as the NRST pin assertion, but the pins BMS and NTRI are not sampled. Boot Mode and Tri-state Mode are not updated. If the NRST pin is asserted and the watchdog triggers the internal reset, the NRST pin has priority.

Emulation Functions

Tri-state Mode

The AT91M55800A provides a Tri-state Mode, which is used for debug purposes. This enables the connection of an emulator probe to an application board without having to desolder the device from the target board. In Tri-state Mode, all the output pin drivers of the AT91M55800A microcontroller are disabled.

To enter Tri-state Mode, the pin NTRI must be held low during the last 10 clock cycles before the rising edge of NRST. For normal operation the pin NTRI must be held high during reset, by a resistor of up to 400K Ohm.

NTRI is multiplexed with I/O line PA18 and USART 1 serial data transmit line TXD1.

Standard RS232 drivers generally contain internal 400K Ohm pull-up resistors. If TXD1 is connected to a device not including this pull-up, the user must make sure that a high level is tied on NTRI while NRST is asserted.

JTAG/ICE Debug Mode

ARM Standard Embedded In-Circuit Emulation is supported via the JTAG/ICE port. It is connected to a host computer via an external ICE Interface. The JTAG/ICE debug mode is enabled when JTAGSEL is low.

In ICE Debug Mode the ARM Core responds with a non-JTAG chip ID which identifies the core to the ICE system. This is not JTAG compliant.

IEEE 1149.1 JTAG Boundary-scan

JTAG Boundary-scan is enabled when JTAGSEL is high. The functions SAMPLE, EXTEST and BYPASS are implemented. There is no JTAG chip ID. The Special Function module provides a chip ID which is independent of JTAG.

It is not possible to switch directly between JTAG and ICE operations. A chip reset must be performed (NRST and NTRST) after JTAGSEL is changed.

Memory Controller

The ARM7TDMI processor address space is 4G bytes. The memory controller decodes the internal 32-bit address bus and defines three address spaces:

- Internal memories in the four lowest megabytes
- Middle space reserved for the external devices (memory or peripherals) controlled by the EBI
- Internal peripherals in the four highest megabytes

In any of these address spaces, the ARM7TDMI operates in Little-Endian mode only.

Internal Memories

The AT91M55800A microcontroller integrates an 8-Kbyte SRAM bank. This memory bank is mapped at address 0x0 (after the remap command), allowing ARM7TDMI exception vectors between 0x0 and 0x20 to be modified by the software. The rest of the bank can be used for stack allocation (to speed up context saving and restoring), or as data and program storage for critical algorithms. All internal memory is 32 bits wide and single-clock cycle accessible. Byte (8-bit), half-word (16-bit) or word (32-bit) accesses are supported and are executed within one cycle. Fetching Thumb or ARM instructions is supported and internal memory can store twice as many Thumb instructions as ARM ones.

Boot Mode Select

The ARM reset vector is at address 0x0. After the NRST line is released, the ARM7TDMI executes the instruction stored at this address. This means that this address must be mapped in nonvolatile memory after the reset.

The input level on the BMS pin during the last 10 clock cycles before the rising edge of the NRST selects the type of boot memory (see Table 4).

The pin BMS is multiplexed with the I/O line PB18 that can be programmed after reset like any standard PIO line.

Table 4. Boot Mode Select

BMS	Boot Mode
1	External 8-bit memory on NCS0
0	External 16-bit memory on NCS0

Remap Command

The ARM vectors (Reset, Abort, Data Abort, Prefetch Abort, Undefined Instruction, Interrupt, Fast Interrupt) are mapped from address 0x0 to address 0x20. In order to allow these vectors to be redefined dynamically by the software, the AT91M55800A microcontroller uses a remap command that enables switching between the boot memory and the internal RAM bank addresses. The remap command is accessible through the EBI User Interface, by writing one in RCB of EBI_RCR (Remap Control Register). Performing a remap command is mandatory if access to the other external devices (connected to chip selects 1 to 7) is required. The remap operation can only be changed back by an internal reset or an NRST assertion.

Abort Control

The abort signal providing a Data Abort or a Prefetch Abort exception to the ARM7TDMI is asserted when accessing an undefined address in the EBI address space.

No abort is generated when reading the internal memory or by accessing the internal peripherals, whether the address is defined or not.

External Bus Interface

The External Bus Interface handles the accesses between addresses 0x0040 0000 and 0xFFC0 0000. It generates the signals that control access to the external devices, and can configure up to eight 16-Mbyte banks. In all cases it supports byte, half-word and word aligned accesses.

For each of these banks, the user can program:

- Number of wait states
- Number of data float times (wait time after the access is finished to prevent any bus contention in case the device is too long in releasing the bus)
- Data bus width (8-bit or 16-bit)
- With a 16-bit wide data bus, the user can program the EBI to control one 16-bit device (Byte Access Select Mode) or two 8-bit devices in parallel that emulate a 16-bit memory (Byte-write Access mode).

The External Bus Interface features also the Early Read Protocol, configurable for all the devices, that significantly reduces access time requirements on an external device.

Peripherals

The AT91M55800A peripherals are connected to the 32-bit wide Advanced Peripheral Bus. Peripheral registers are only word accessible – byte and half-word accesses are not supported. If a byte or a half-word access is attempted, the memory controller automatically masks the lowest address bits and generates a word access.

Each peripheral has a 16-Kbyte address space allocated (the AIC only has a 4-Kbyte address space).

Peripheral Registers

The following registers are common to all peripherals:

- Control Register – Write-only register that triggers a command when a one is written to the corresponding position at the appropriate address. Writing a zero has no effect.
- Mode Register – read/write register that defines the configuration of the peripheral. Usually has a value of 0x0 after a reset.
- Data Register – read and/or write register that enables the exchange of data between the processor and the peripheral.
- Status Register – Read-only register that returns the status of the peripheral.
- Enable/Disable/Status Registers – shadow command registers. Writing a one in the Enable Register sets the corresponding bit in the Status Register. Writing a one in the Disable Register resets the corresponding bit and the result can be read in the Status Register. Writing a bit to zero has no effect. This register access method maximizes the efficiency of bit manipulation, and enables modification of a register with a single non-interruptible instruction, replacing the costly read-modify-write operation.

Unused bits in the peripheral registers are shown as “–” and must be written at 0 for upward compatibility. These bits read 0.

Peripheral Interrupt Control

The Interrupt Control of each peripheral is controlled from the status register using the interrupt mask. The status register bits are ANDed to their corresponding interrupt mask bits and the result is then ORed to generate the Interrupt Source signal to the Advanced Interrupt Controller.

The interrupt mask is read in the Interrupt Mask Register and is modified with the Interrupt Enable Register and the Interrupt Disable Register. The enable/disable/status (or mask) makes it possible to enable or disable peripheral interrupt sources with a non-

interruptible single instruction. This eliminates the need for interrupt masking at the AIC or Core level in real-time and multi-tasking systems.

Peripheral Data Controller

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 SPI.

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 data is transferred, an end of transfer interrupt is generated by the corresponding peripheral.

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.

System Peripherals

APMC: Advanced Power Management Controller

The AT91M55800A Advanced Power Management Controller allows optimization of power consumption. The APMC enables/disables the clock inputs of most of the peripherals and the ARM Core. Moreover, the main oscillator, the PLL and the analog peripherals can be put in standby mode allowing minimum power consumption to be obtained. The APMC provides the following operating modes:

- Normal: clock generator provides clock to the entire chip except the RTC.
- Wait mode: ARM Core clock deactivated
- Slow Clock mode: clock generator deactivated, master clock 32 kHz
- Standby mode: RTC active, all other clocks disabled
- Power down: RTC active, supply on the rest of the circuit deactivated

RTC: Real-time Clock

The AT91M55800A features a Real-time Clock (RTC) peripheral that is designed for very low power consumption. It combines a complete time-of-day clock with alarm and a two-hundred year Gregorian calendar, complemented by a programmable periodic interrupt.

The time and calendar values are coded in Binary-Coded Decimal (BCD) format. The time format can be 24-hour mode or 12-hour mode with an AM/PM indicator.

Updating time and calendar fields and configuring the alarm fields is performed by a parallel capture on the 32-bit data bus. An entry control is performed to avoid loading registers with incompatible BCD format data or with an incompatible date according to the current month/ year/century.

AIC: Advanced Interrupt Controller

The AIC has an 8-level priority, individually maskable, vectored interrupt controller, and drives the NIRQ and NFIQ pins of the ARM7TDMI from:

- The external fast interrupt line (FIQ)
- The six external interrupt request lines (IRQ0 - IRQ5)
- The interrupt signals from the on-chip peripherals.

The AIC is largely programmable offering maximum flexibility, and its vectoring features reduce the real-time overhead in handling interrupts.

The AIC also features a spurious vector, which reduces Spurious Interrupt handling to a minimum, and a protect mode that facilitates the debug capabilities.

PIO: Parallel I/O Controller

The AT91M55800A has 58 programmable I/O lines. 13 pins are dedicated as general-purpose I/O pins. The other I/O lines are multiplexed with an external signal of a peripheral to optimize the use of available package pins. The PIO lines are controlled by two separate and identical PIO Controllers called PIOA and PIOB. The PIO controller enables the generation of an interrupt on input change and insertion of a simple input glitch filter on any of the PIO pins.

WD: Watchdog

The Watchdog is built around a 16-bit counter, and is used to prevent system lock-up if the software becomes trapped in a deadlock. It can generate an internal reset or interrupt, or assert an active level on the dedicated pin NWDOVF. All programming registers are password-protected to prevent unintentional programming.

SF: Special Function

The AT91M55800A provides registers which implement the following special functions.

- Chip identification
- RESET status

User Peripherals

USART: Universal Synchronous/ Asynchronous Receiver Transmitter

The AT91M55800A provides three identical, full-duplex, universal synchronous/asynchronous receiver/transmitters.

Each USART has its own baud rate generator, and two dedicated Peripheral Data Controller channels. The data format includes a start bit, up to 8 data bits, an optional programmable parity bit and up to 2 stop bits.

The USART also features a Receiver Timeout register, facilitating variable-length frame support when it is working with the PDC, and a Time-guard register, used when interfacing with slow remote equipment.

TC: Timer Counter

The AT91M55800A features two Timer Counter blocks that include 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.

The Timer Counters can be used in Capture or Waveform mode, and all three counter channels can be started simultaneously and chained together.

SPI: Serial Peripheral Interface

The SPI provides communication with external devices in master or slave mode. It has four external chip selects that can be connected to up to 15 devices. The data length is programmable, from 8- to 16-bit.

ADC: Analog-to-digital Converter

The two identical 4-channel 10-bit analog-to-digital converters (ADC) are based on a Successive Approximation Register (SAR) approach.

Each ADC has 4 analog input pins, AD0 to AD3 and AD4 to AD7, digital trigger input pins AD0TRIG and AD1TRIG, and provides an interrupt signal to the AIC. Both ADCs share the analog power supply pins VDDA and GNDA, and the input reference voltage pin ADVREF.

Each channel can be enabled or disabled independently, and has its own data register. The ADC can be configured to automatically enter Sleep mode after a conversion

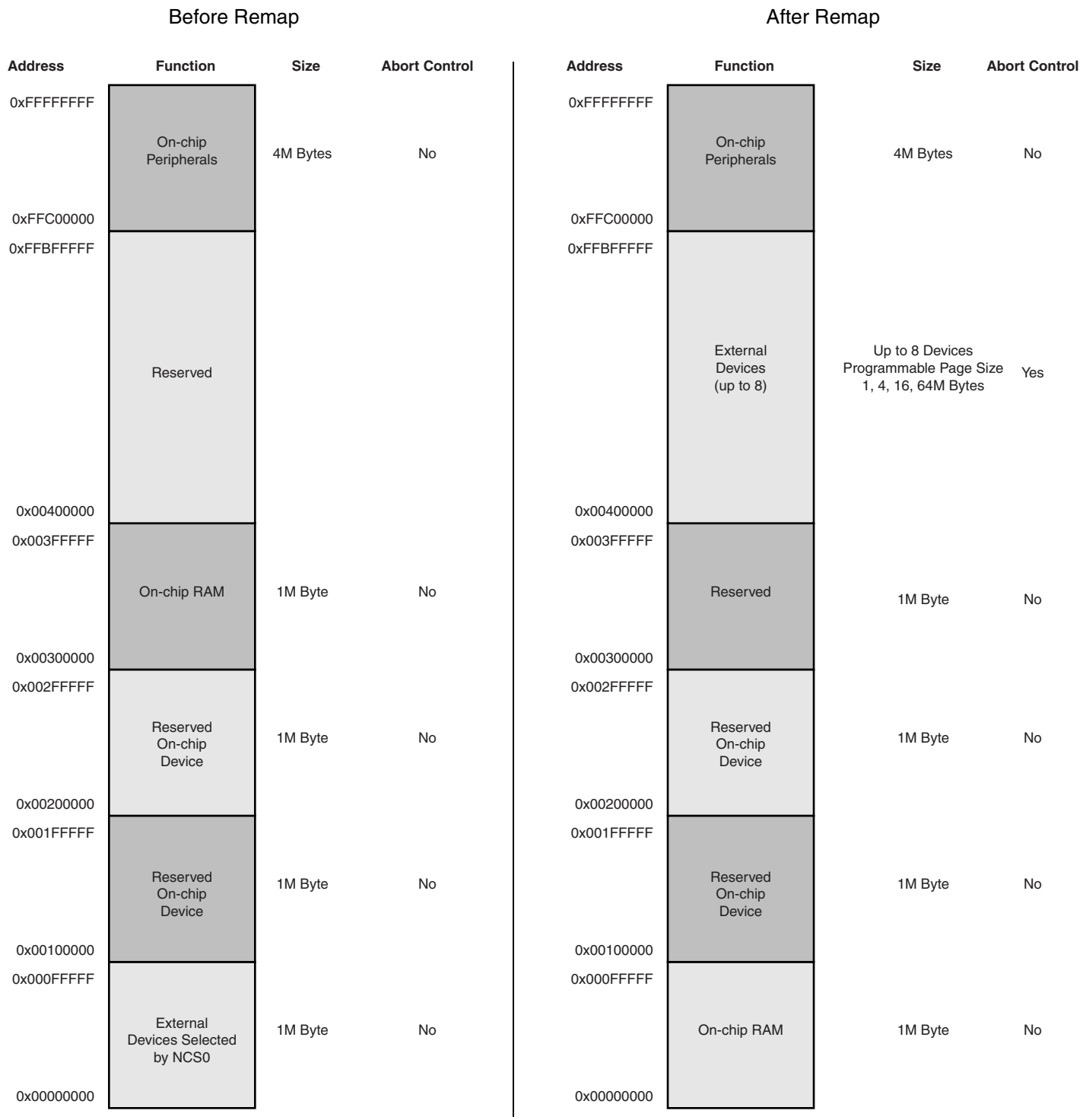
sequence, and can be triggered by the software, the Timer Counter, or an external signal.

DAC: Digital-to-analog Converter

Each DAC has an analog output pin, DA0 and DA1, and provides an interrupt signal to the AIC DA0IRQ and DA1IRQ. Both DACs share the analog power supply pins VDDA and GNDA, and the input reference DAVREF.

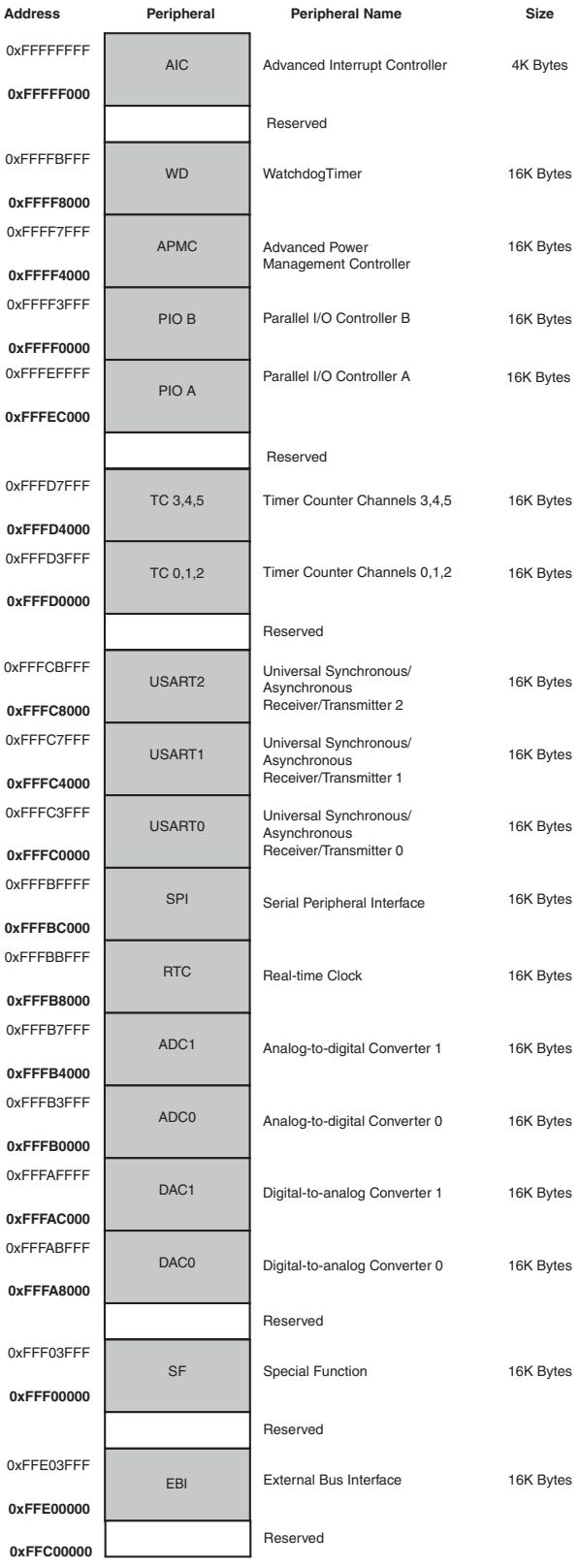
Memory Map

Figure 3. AT91M55800A Memory Map Before and after Remap Command



Peripheral Memory Map

Figure 4. AT91M55800A Peripheral Memory Map



EBI: External Bus Interface

The EBI generates the signals that control the access to the external memory or peripheral devices. The EBI is fully-programmable and can address up to 128M bytes. It has eight chip selects and a 24-bit address bus.

The 16-bit data bus can be configured to interface with 8- or 16-bit external devices. Separate read and write control signals allow for direct memory and peripheral interfacing.

The EBI supports different access protocols allowing single-clock cycle memory accesses.

The main features are:

- External memory mapping
- 8 active-low chip select lines
- 8- or 16-bit data bus
- Byte-write or byte-select lines
- Remap of boot memory
- Two different read protocols
- Programmable wait state generation
- External wait request
- Programmable data float time

The EBI User Interface is described on page 46.

External Memory Mapping

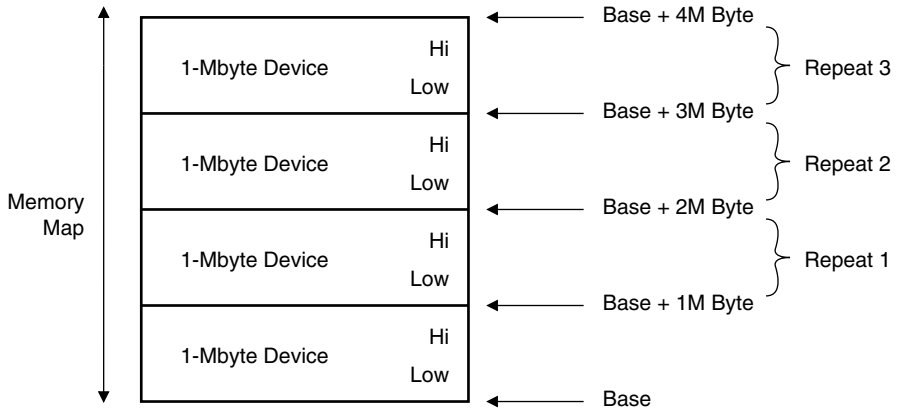
The memory map associates the internal 32-bit address space with the external 24-bit address bus.

The memory map is defined by programming the base address and page size of the external memories (see EBI User Interface registers EBI_CSR0 to EBI_CSR7). Note that A0 - A23 is only significant for 8-bit memory; A1 - A23 is used for 16-bit memory.

If the physical memory device is smaller than the programmed page size, it wraps around and appears to be repeated within the page. The EBI correctly handles any valid access to the memory device within the page (see Figure 5).

In the event of an access request to an address outside any programmed page, an Abort signal is generated. Two types of Abort are possible: instruction prefetch abort and data abort. The corresponding exception vector addresses are respectively 0x0000 000C and 0x0000 0010. It is up to the system programmer to program the error handling routine to use in case of an Abort (see the ARM7TDMI datasheet for further information).

Figure 5. External Memory Smaller than Page Size



EBI Pin Description

Name	Description	Type
A0 - A23	Address bus (output)	Output
D0 - D15	Data bus (input/output)	I/O
NCS0 - NCS7	Active low chip selects (output)	Output
NRD	Read Enable (output)	Output
NWR0 - NWR1	Lower and upper write enable (output)	Output
NOE	Output enable (output)	Output
NWE	Write enable (output)	Output
NUB, NLB	Upper and lower byte-select (output)	Output
NWAIT	Wait request (input)	Input

The following table shows how certain EBI signals are multiplexed:

Multiplexed Signals		Functions
A0	NLB	8- or 16-bit data bus
NRD	NOE	Byte-write or byte-select access
NWR0	NWE	Byte-write or byte-select access
NWR1	NUB	Byte-write or byte-select access

Data Bus Width

A data bus width of 8 or 16 bits can be selected for each chip select. This option is controlled by the DBW field in the EBI_CSR (Chip-select Register) for the corresponding chip select.

Figure 6 shows how to connect a 512K x 8-bit memory on NCS2.

Figure 6. Memory Connection for an 8-bit Data Bus

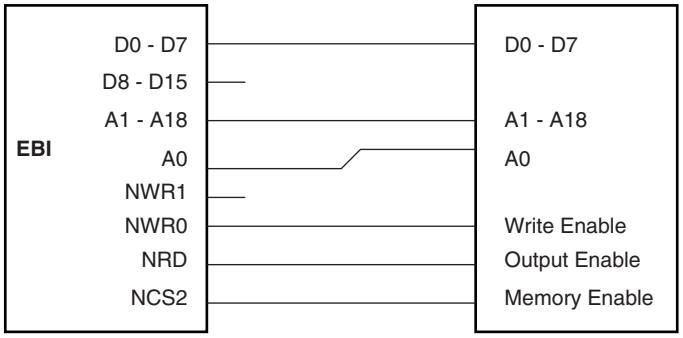
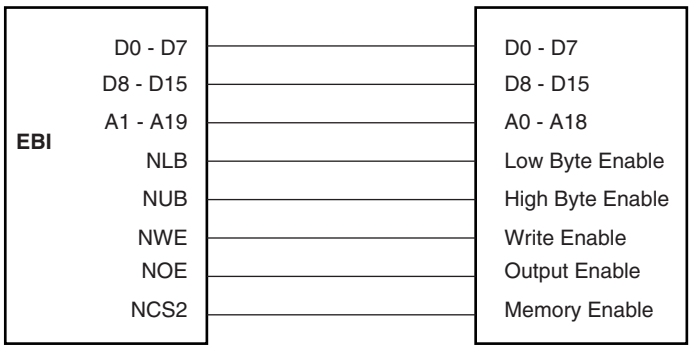


Figure 7 shows how to connect a 512K x 16-bit memory on NCS2.

Figure 7. Memory Connection for a 16-bit Data Bus



Byte-write or Byte-select Access

Each chip select with a 16-bit data bus can operate with one of two different types of write access:

- Byte-write Access supports two Byte-write and a single read signal.
- Byte-select Access selects upper and/or lower byte with two byte-select lines, and separate read and write signals.

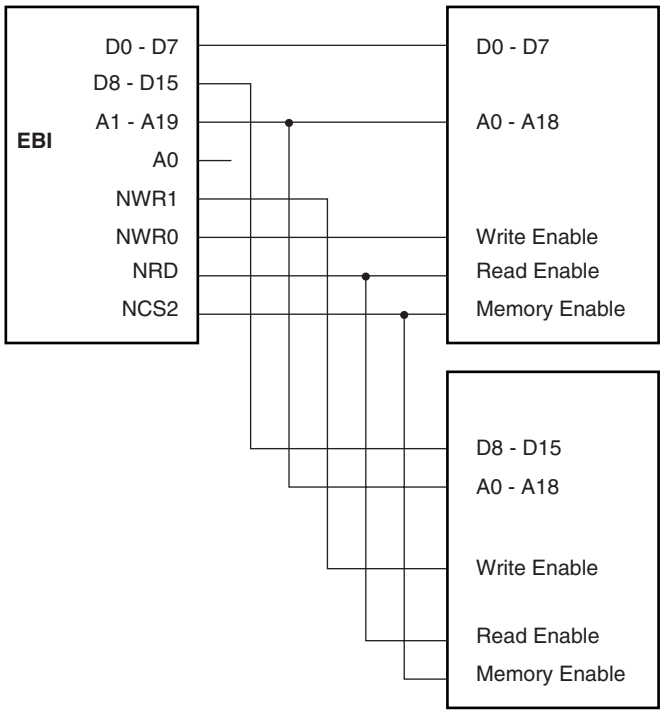
This option is controlled by the BAT field in the EBI_CSR (Chip-select Register) for the corresponding chip select.

Byte-write Access is used to connect 2 x 8-bit devices as a 16-bit memory page.

- The signal A0/NLB is not used.
- The signal NWR1/NUB is used as NWR1 and enables upper byte writes.
- The signal NWR0/NWE is used as NWR0 and enables lower byte writes.
- The signal NRD/NOE is used as NRD and enables half-word and byte reads.

Figure 8 shows how to connect two 512K x 8-bit devices in parallel on NCS2.

Figure 8. 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 9 shows how to connect a 16-bit device with byte and half-word access (e.g. 16-bit SRAM) on NCS2.

Figure 9. Connection for a 16-bit Data Bus with Byte and Half-word Access

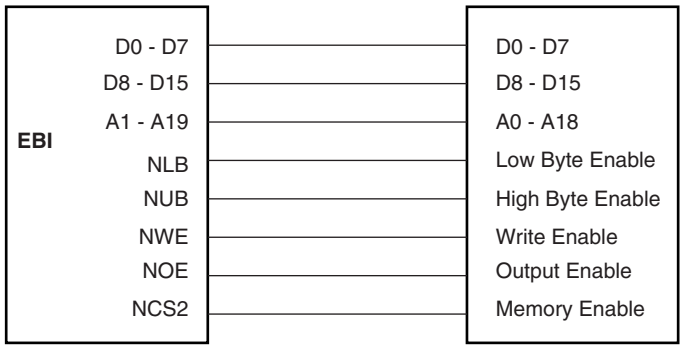
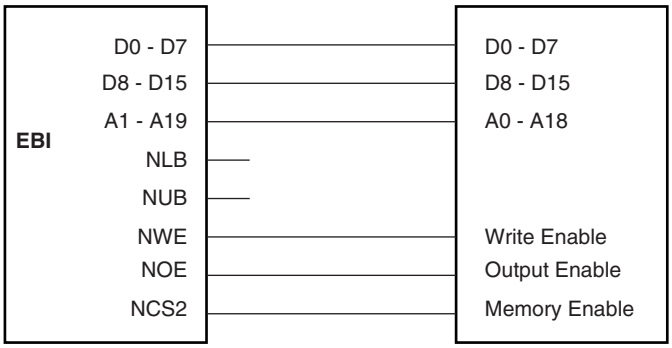


Figure 10 shows how to connect a 16-bit device without byte access (e.g. Flash) on NCS2.

Figure 10. Connection for a 16-bit Data Bus Without Byte-write Capability.



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 values 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.

Read Protocols

The EBI 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 NRD (read cycle) waveform.

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

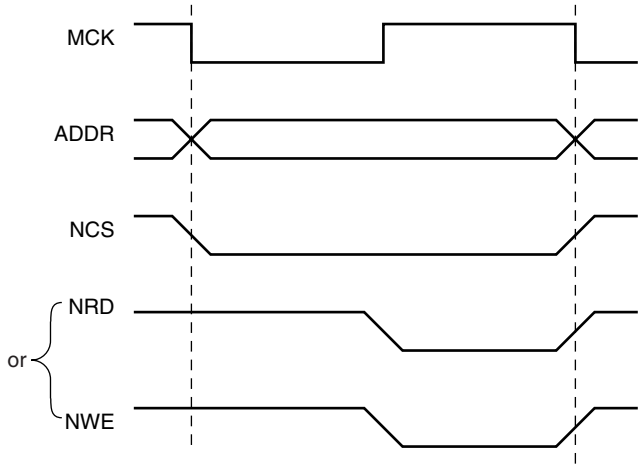
Note: In the following waveforms and descriptions, **NRD** represents NRD and NOE since the two signals have the same waveform. Likewise, **NWE** represents NWE, NWR0 and NWR1 unless NWR0 and NWR1 are otherwise represented. **ADDR** represents A0 - A23 and/or A1 - A23.

Standard Read Protocol

Standard read protocol implements a read cycle in which NRD and NWE 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 NCS before the read cycle begins.

During a standard read protocol, external memory access, NCS is set low and ADDR is valid at the beginning of the access while NRD goes low only in the second half of the master clock cycle to avoid bus conflict (see Figure 11). NWE is the same in both protocols. NWE always goes low in the second half of the master clock cycle (see Figure 12).

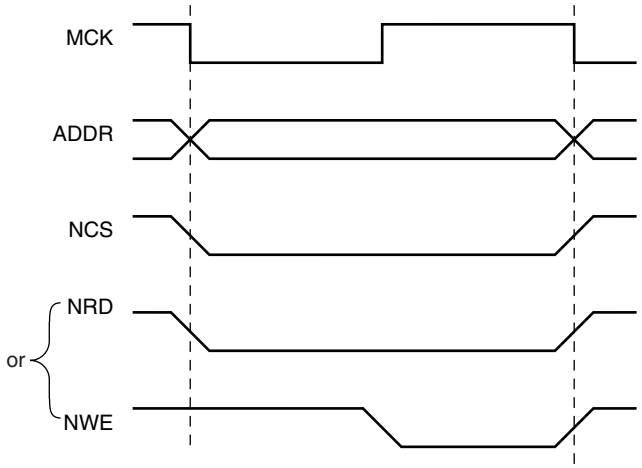
Figure 11. Standard Read Protocol



Early Read Protocol

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

Figure 12. Early Read Protocol



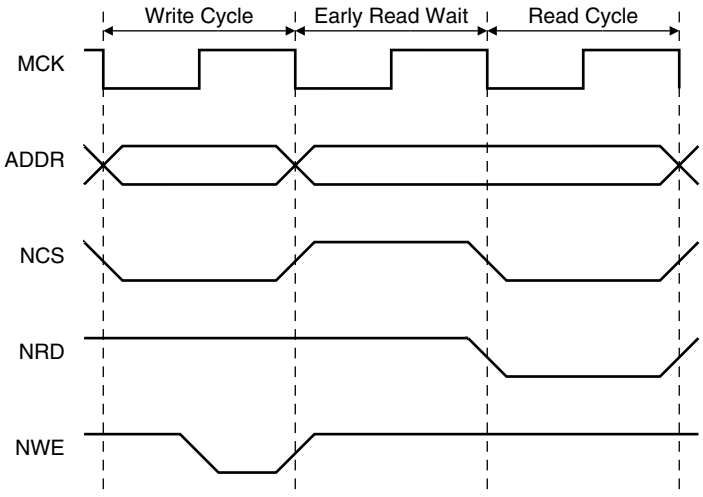
Early Read Wait State

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 (see Figure 13). 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.

Early read wait states affect the external bus only. They do not affect internal bus timing.

Figure 13. Early Read Wait State

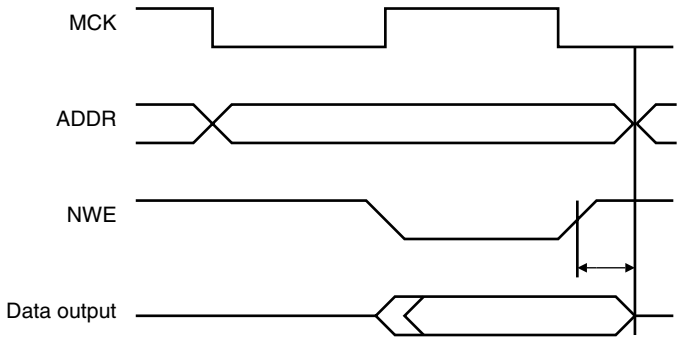


Write Data Hold Time

During write cycles in both protocols, output data becomes valid after the falling edge of the NWE signal and remains valid after the rising edge of NWE, as illustrated in the figure below. The external NWE waveform (on the NWE pin) is used to control the output data timing to guarantee this operation.

It is therefore necessary to avoid excessive loading of the NWE pins, which could delay the write signal too long and cause a contention with a subsequent read cycle in standard protocol.

Figure 14. Data Hold Time



In early read protocol the data can remain valid longer than in standard read protocol due to the additional wait cycle which follows a write access.

Wait States

The EBI can automatically insert wait states. The different types of wait states are listed below:

- Standard wait states
- Data float wait states
- External wait states
- Chip select change wait states
- Early read wait states (as described in Read Protocols)

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 EBI_CSR. The number of cycles to insert is programmed in the NWS field in the same register.

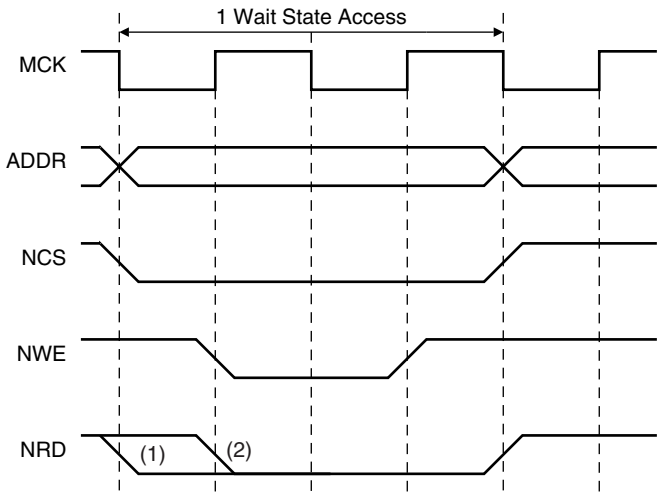
Below is the correspondence between the number of standard wait states programmed and the number of cycles during which the NWE pulse is held low:

0 wait states 1/2 cycle

1 wait state 1 cycle

For each additional wait state programmed, an additional cycle is added.

Figure 15. One Wait State Access



Notes: 1. Early Read Protocol
2. Standard Read Protocol

Data Float Wait State

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.

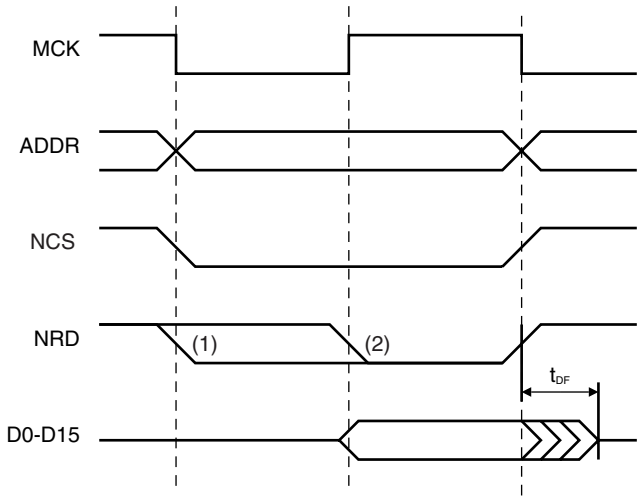
The Data Float Output Time (t_{DF}) for each external memory device is programmed in the TDF field of the EBI_CSR register for the corresponding chip select. The value (0 - 7 clock cycles) indicates the number of data float waits to be inserted and represents the time allowed for the data output to go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

Data float wait states do not delay internal memory accesses. Hence, a single access to an external memory with long t_{DF} will not slow down the execution of a program from internal memory.

The EBI keeps track of the programmed external data float time during 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 have added Data Float wait states.

Figure 16. Data Float Output Time



Notes: 1. Early Read Protocol
2. Standard Read Protocol

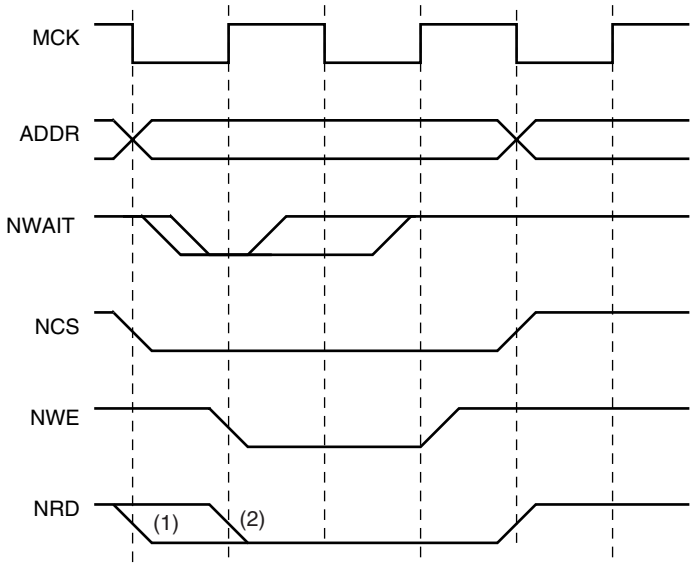
External Wait

The NWAIT input can be used to add wait states at any time. NWAIT is active low and is detected on the rising edge of the clock.

If NWAIT is low at the rising edge of the clock, the EBI adds a wait state and changes neither the output signals nor its internal counters and state. When NWAIT is de-asserted, the EBI finishes the access sequence.

The NWAIT signal must meet setup and hold requirements on the rising edge of the clock.

Figure 17. External Wait

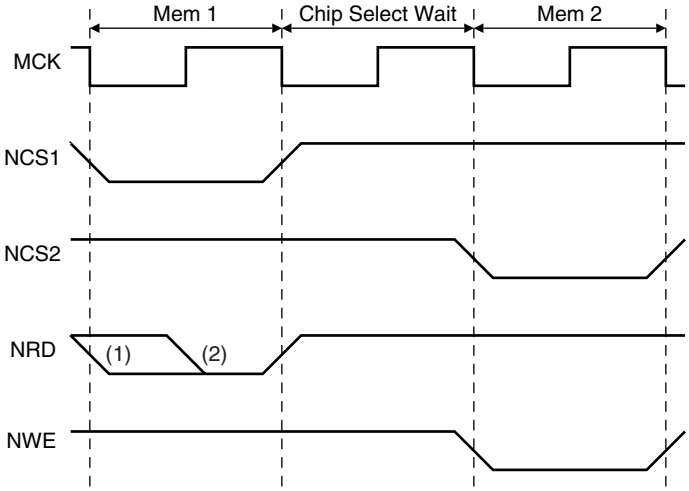


Notes: 1. Early Read Protocol
2. Standard Read Protocol

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.

Figure 18. Chip Select Wait



Notes: 1. Early Read Protocol
2. Standard Read Protocol

Memory Access Waveforms

Figures 19 through 22 show examples of the two alternative protocols for external memory read access.

Figure 19. Standard Read Protocol with no t_{DF}

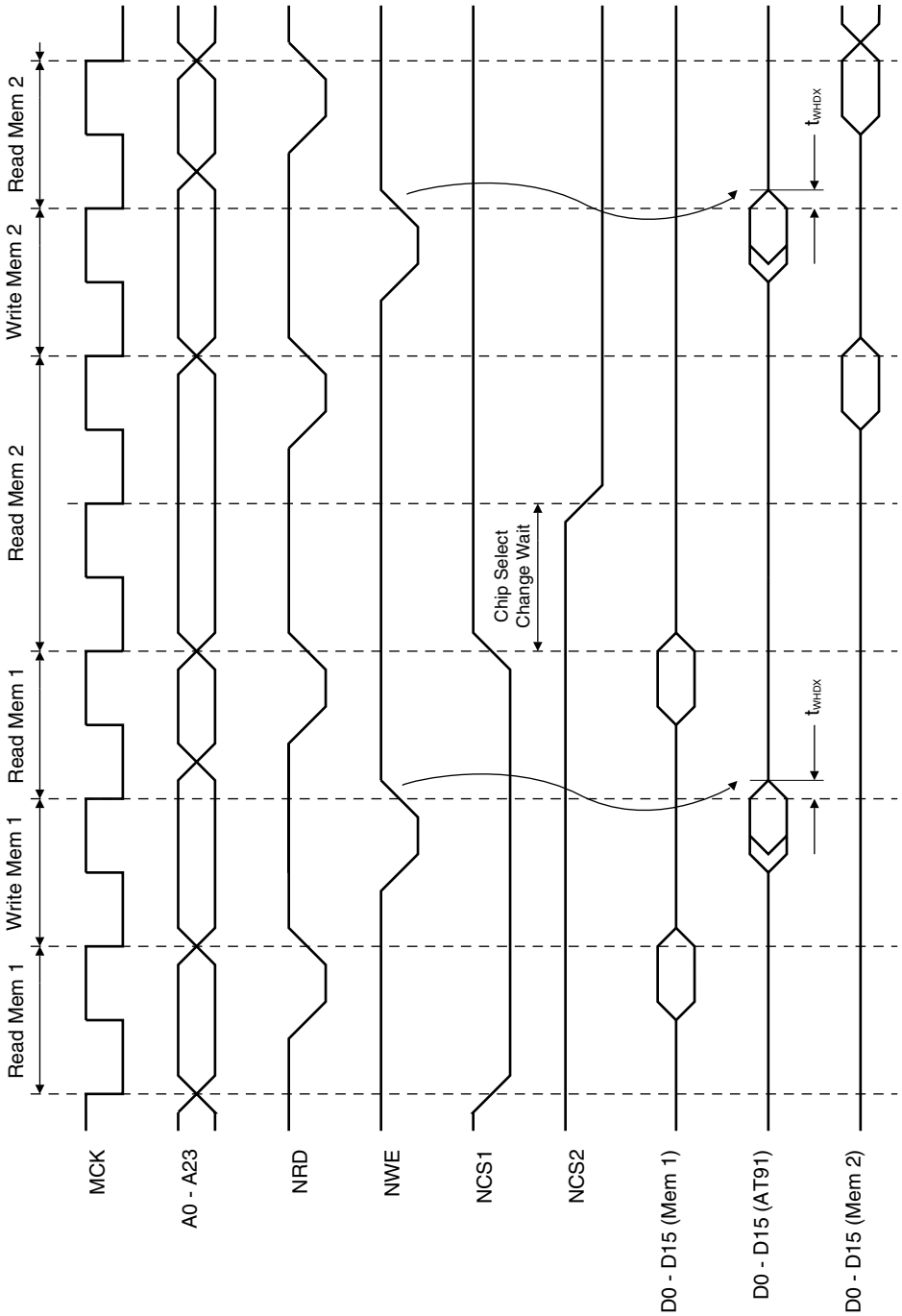


Figure 20. Early Read Protocol with no t_{DF}

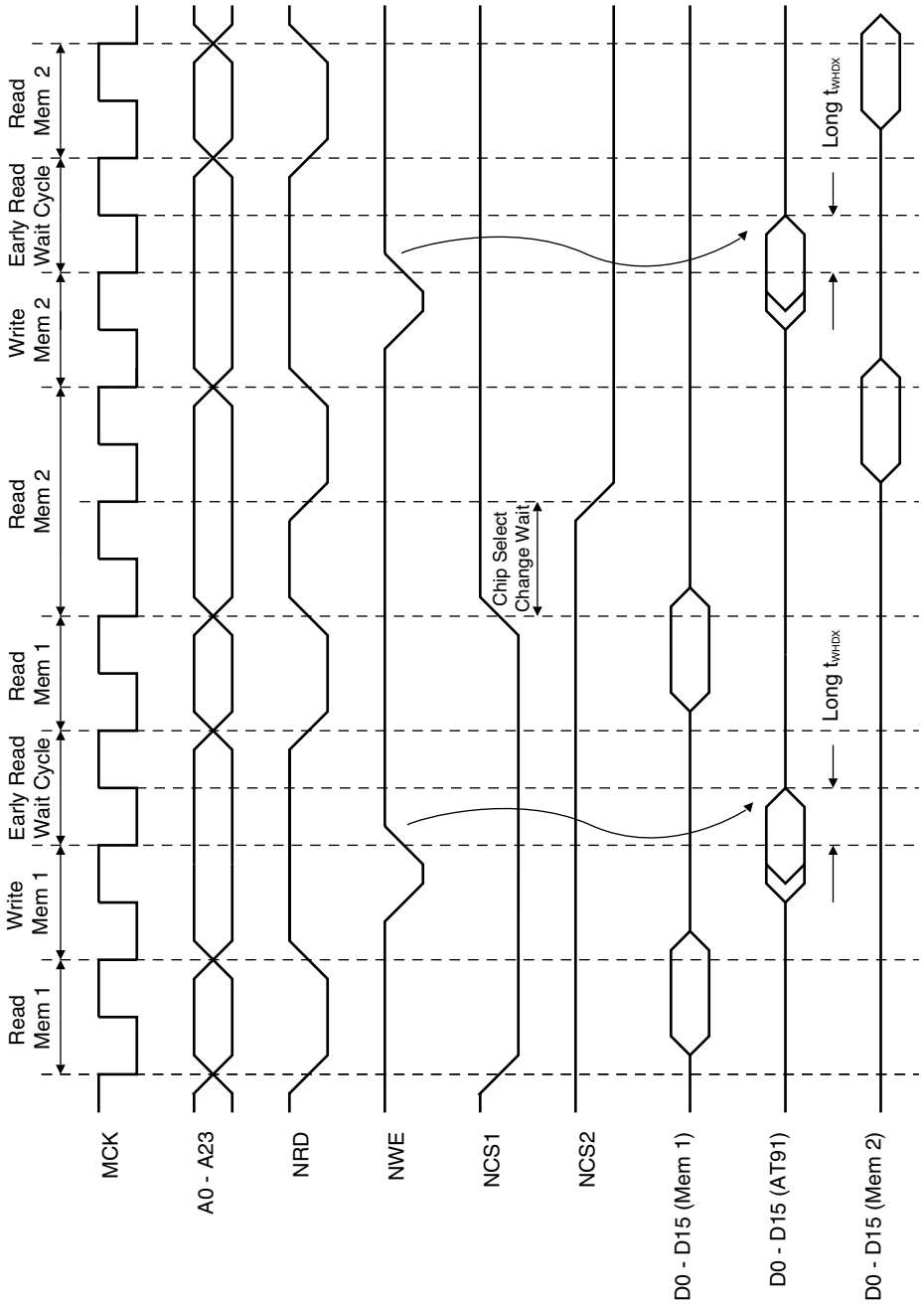


Figure 21. Standard Read Protocol with t_{DF}

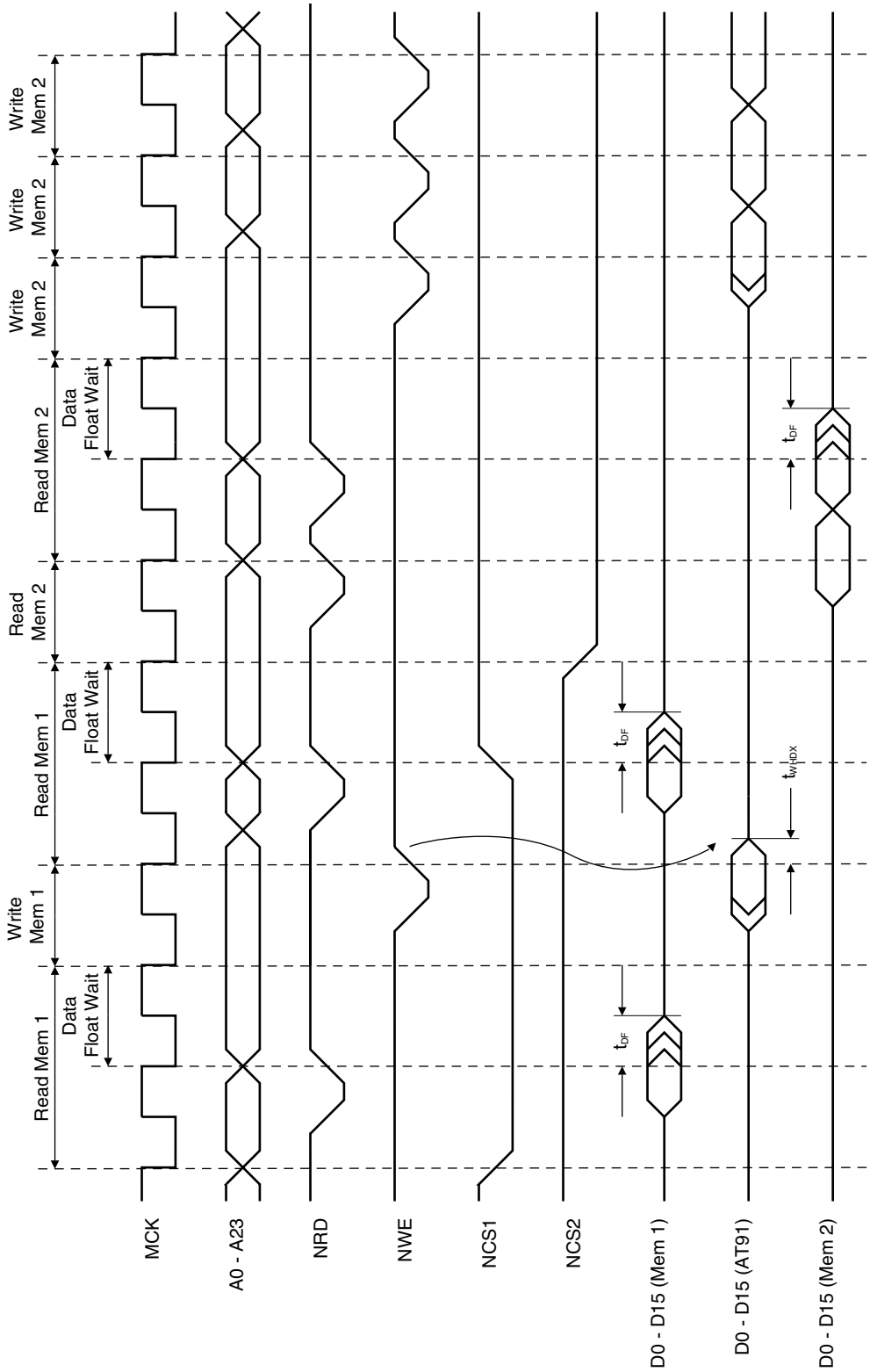
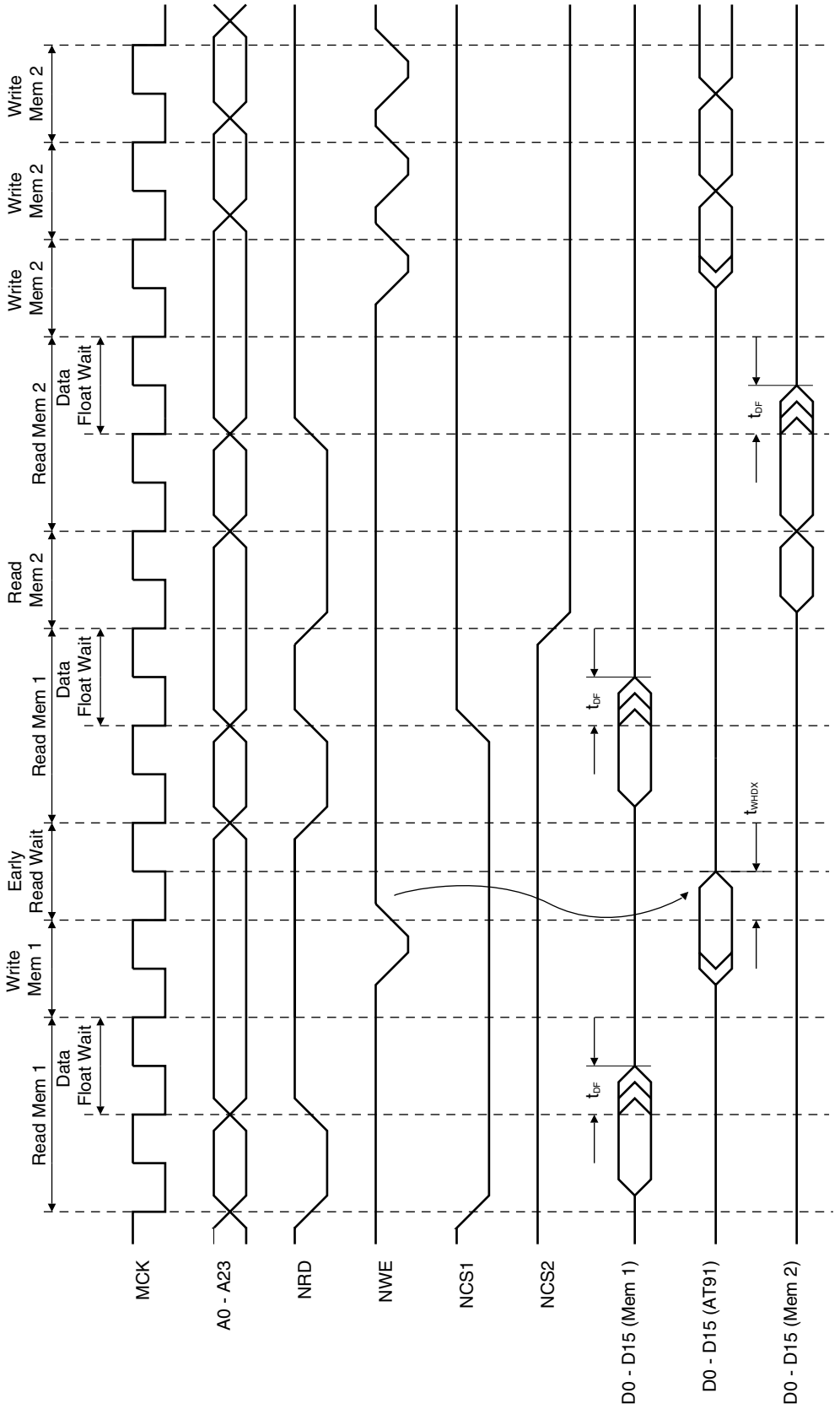


Figure 22. Early Read Protocol with t_{DF}



Figures 23 through 29 show the timing cycles and wait states for read and write access to the various AT91M55800A external memory devices. The configurations described are as follows:

Table 5. Memory Access Waveforms

Figure Number	Number of Wait States	Bus Width	Size of Data Transfer
23	0	16	Word
24	1	16	Word
25	1	16	Half-word
26	0	8	Word
27	1	8	Half-word
28	1	8	Byte
29	0	16	Byte

Figure 23. 0 Wait States, 16-bit Bus Width, Word Transfer

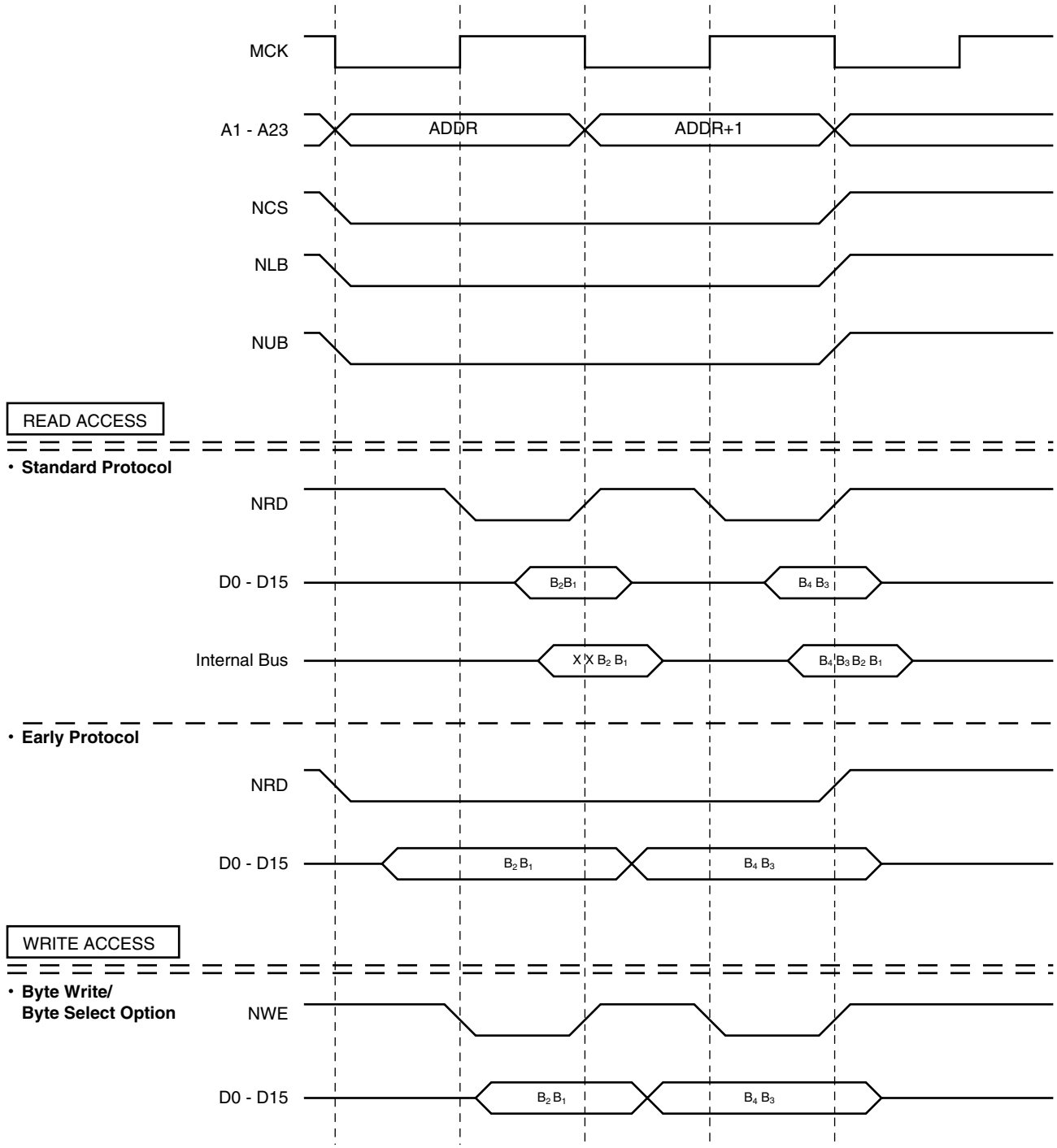


Figure 24. 1 Wait State, 16-bit Bus Width, Word Transfer

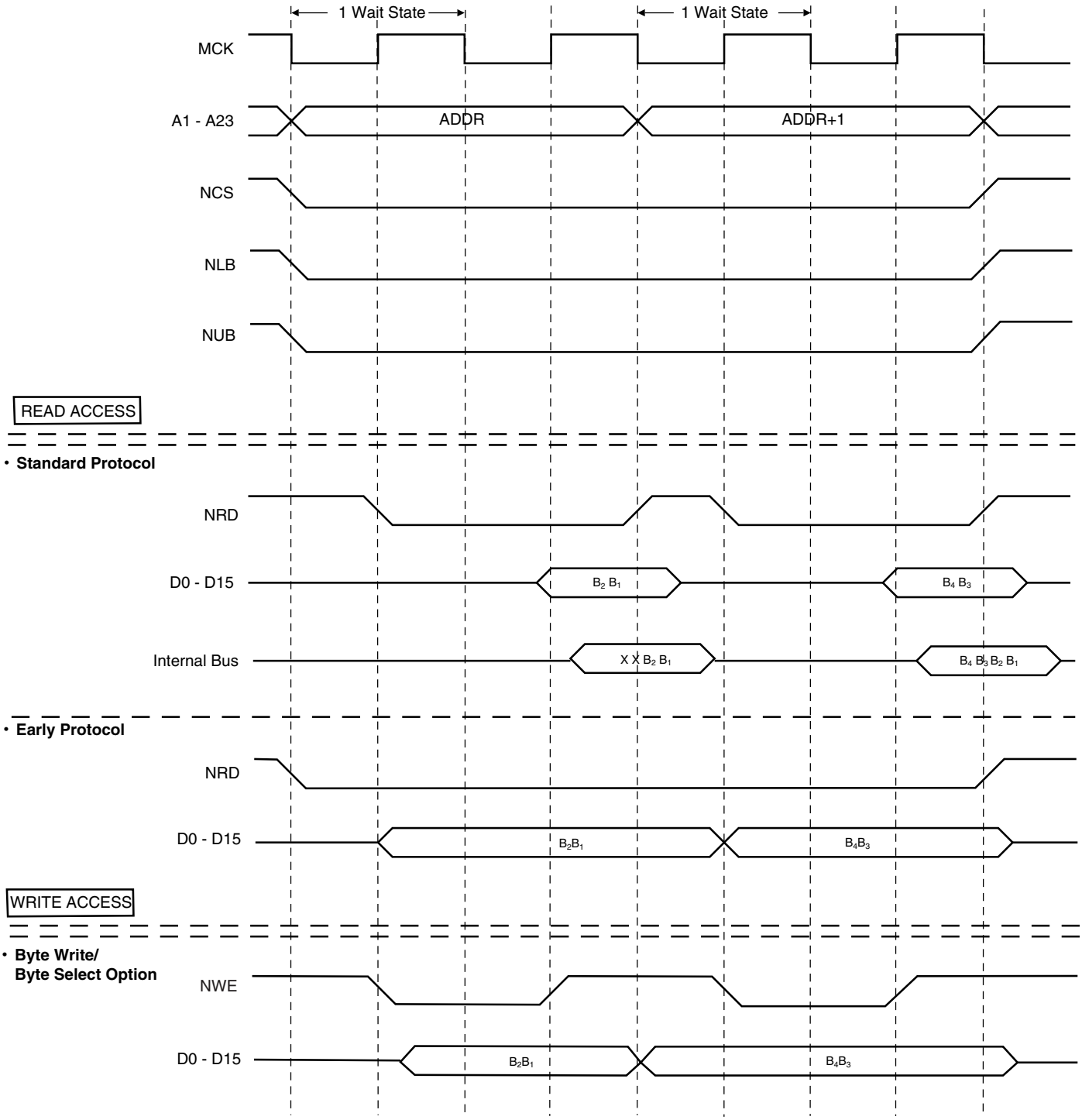


Figure 25. 1 Wait State, 16-bit Bus Width, Half-word Transfer

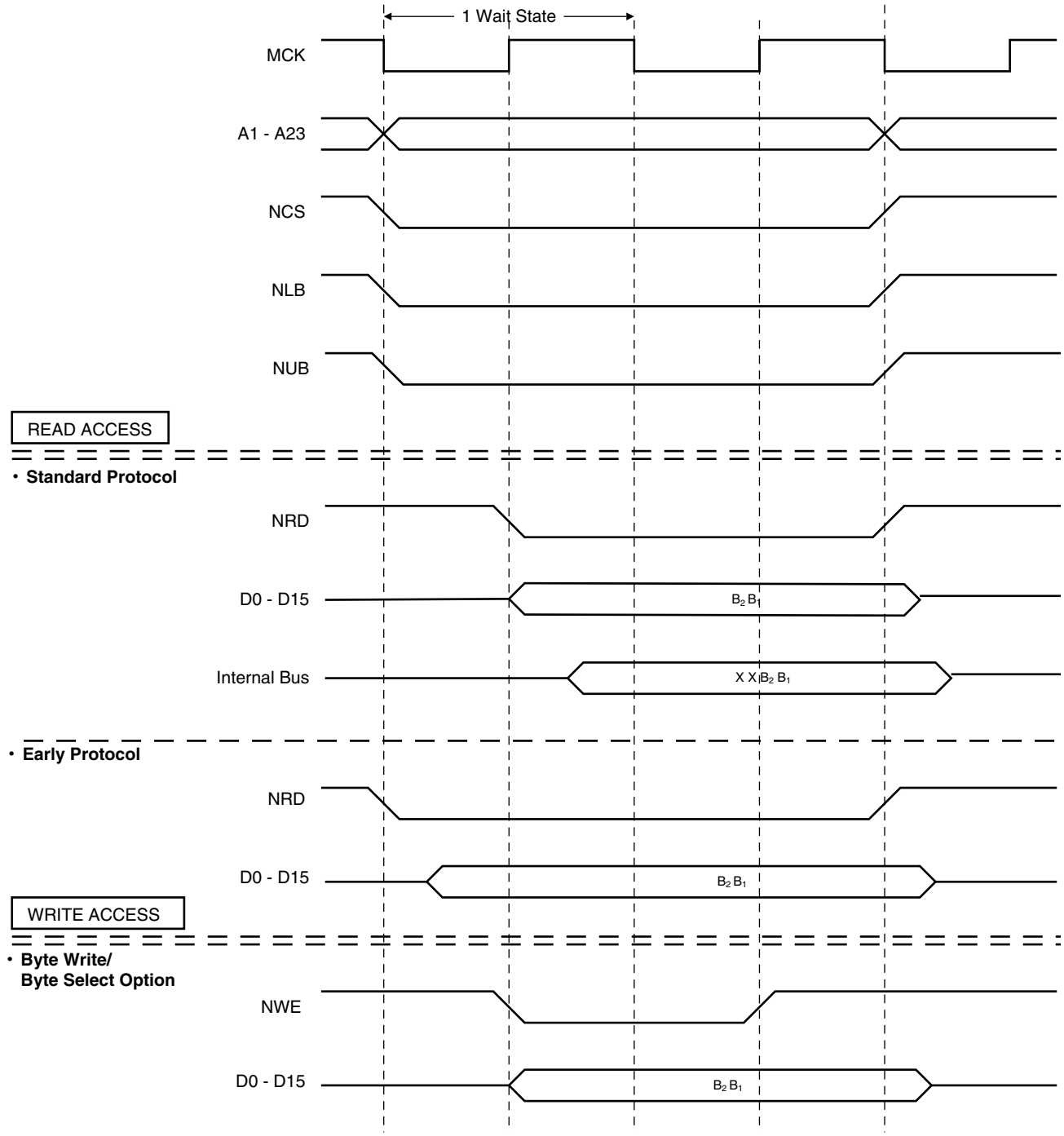


Figure 26. 0 Wait States, 8-bit Bus Width, Word Transfer

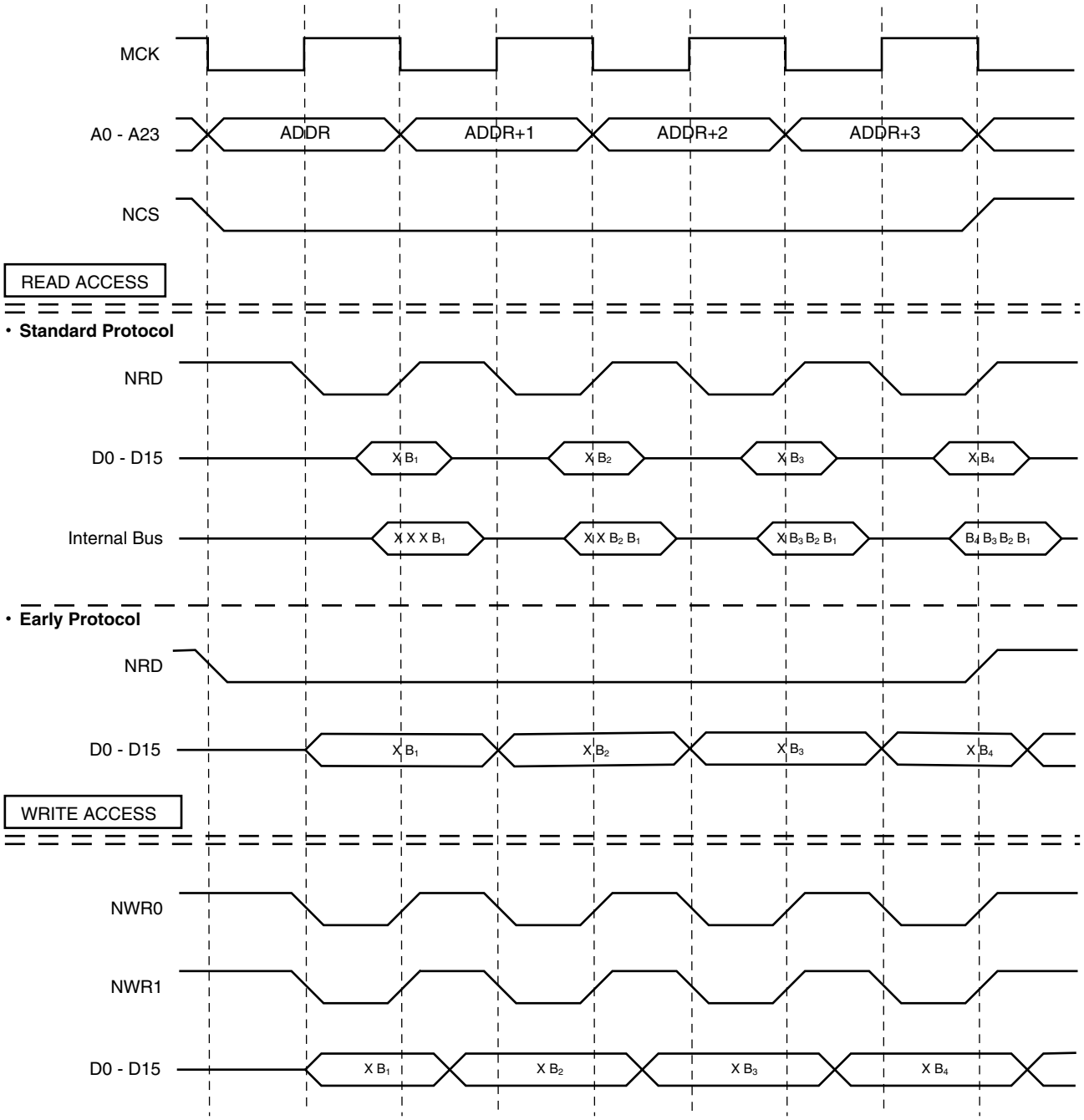


Figure 27. 1 Wait State, 8-bit Bus Width, Half-word Transfer

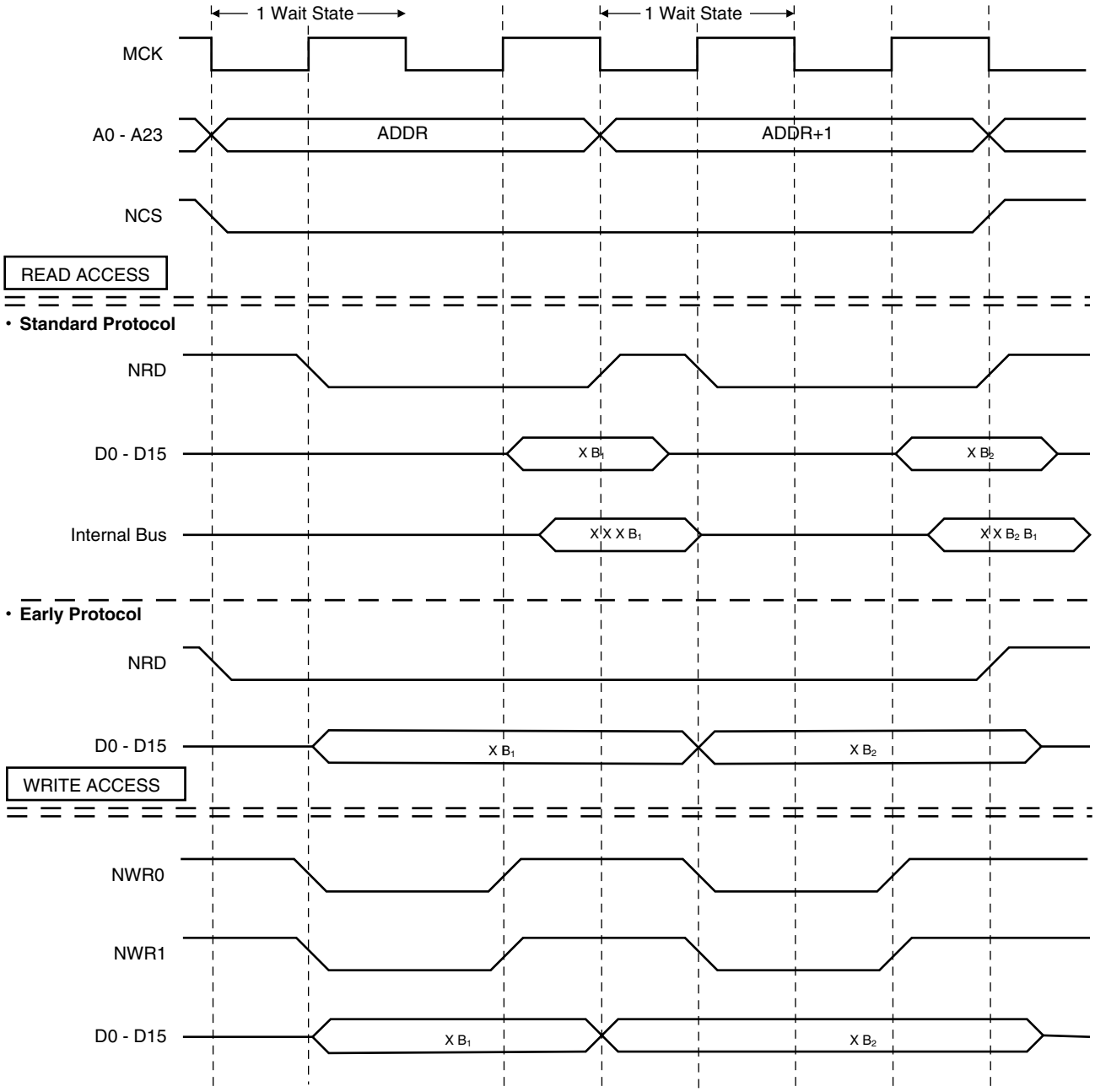


Figure 28. 1 Wait State, 8-bit Bus Width, Byte Transfer

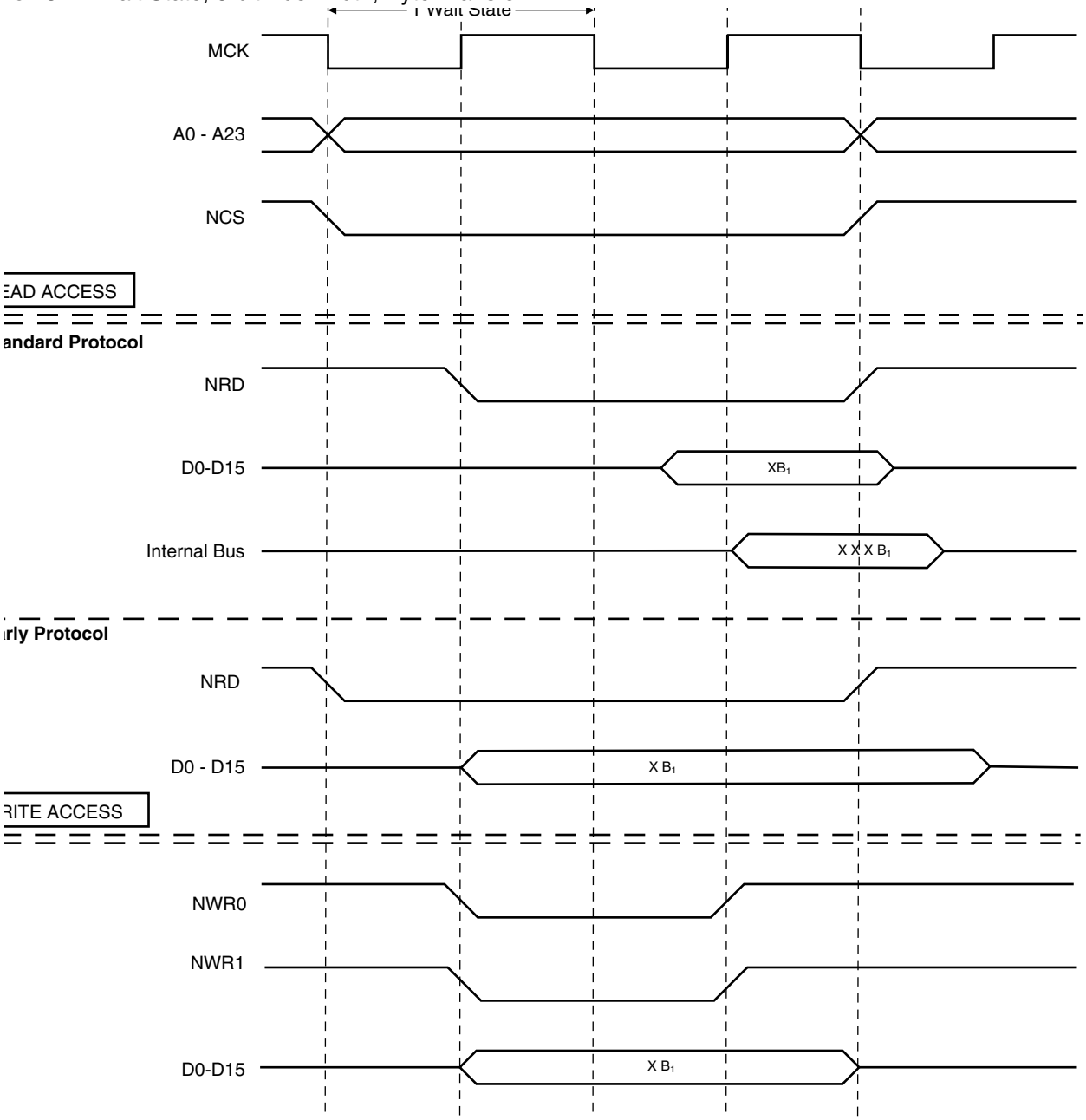
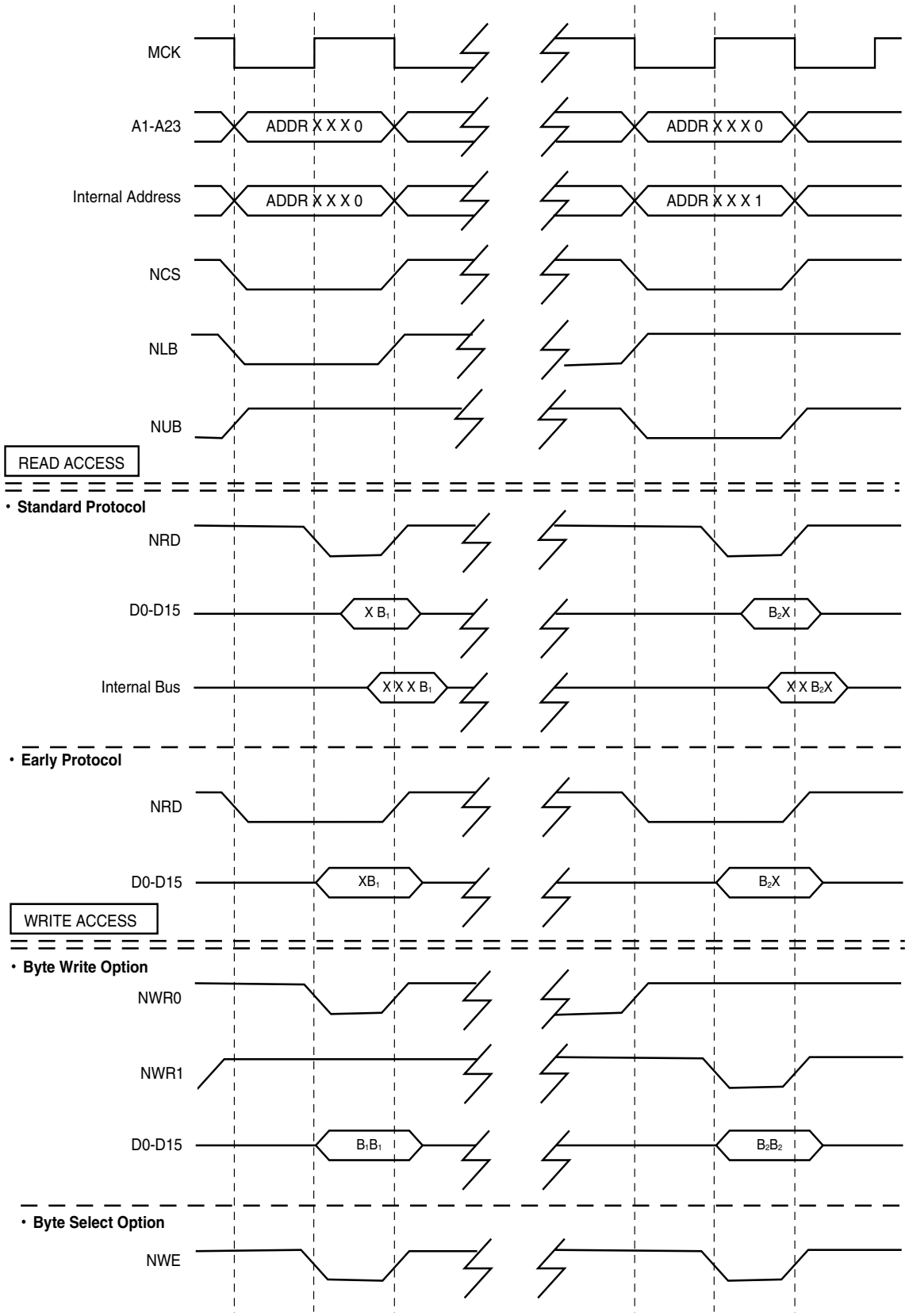


Figure 29. 0 Wait States, 16-bit Bus Width, Byte Transfer



EBI User Interface

The EBI is programmed using the registers listed in the table below. The Remap Control Register (EBI_RCR) controls exit from Boot Mode (See “Boot on NCS0” on page 26.) The Memory Control Register (EBI_MCR) is used to program the number of active chip selects and data read protocol. Eight Chip-select Registers (EBI_CSR0 to EBI_CSR7) are used to program the parameters for the individual external memories. Each EBI_CSR must be programmed with a different base address, even for unused chip selects.

Base Address: 0xFFE00000 (Code Label `EBI_BASE`)

Table 6. EBI Memory Map

Offset	Register	Name	Access	Reset State
0x00	Chip-select Register 0	EBI_CSR0	Read/Write	0x0000203E(1) 0x0000203D(2)
0x04	Chip-select Register 1	EBI_CSR1	Read/Write	0x10000000
0x08	Chip-select Register 2	EBI_CSR2	Read/Write	0x20000000
0x0C	Chip-select Register 3	EBI_CSR3	Read/Write	0x30000000
0x10	Chip-select Register 4	EBI_CSR4	Read/Write	0x40000000
0x14	Chip-select Register 5	EBI_CSR5	Read/Write	0x50000000
0x18	Chip-select Register 6	EBI_CSR6	Read/Write	0x60000000
0x1C	Chip-select Register 7	EBI_CSR7	Read/Write	0x70000000
0x20	Remap Control Register	EBI_RCR	Write-only	–
0x24	Memory Control Register	EBI_MCR	Read/Write	0

Notes: 1. 8-bit boot (if BMS is detected high)
2. 16-bit boot (if BMS is detected low)

EBI Chip Select Register

Register Name: EBI_CSR0 - EBI_CSR7
Access Type: Read/Write
Reset Value: See Table 6
Absolute Address: 0xFFE00000 - 0xFFE0001C

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	–	WSE	NWS			DBW	

- **DBW: Data Bus Width**

DBW		Data Bus Width	Code Label
			EBI_DBW
0	0	Reserved	–
0	1	16-bit data bus width	EBI_DBW_16
1	0	8-bit data bus width	EBI_DBW_8
1	1	Reserved	–

- **NWS: Number of Wait States**

  This field is valid only if WSE is set.

NWS			Number of Standard Wait States	Code Label
				EBI_NWS
0	0	0	1	EBI_NWS_1
0	0	1	2	EBI_NWS_2
0	1	0	3	EBI_NWS_3
0	1	1	4	EBI_NWS_4
1	0	0	5	EBI_NWS_5
1	0	1	6	EBI_NWS_6
1	1	0	7	EBI_NWS_7
1	1	1	8	EBI_NWS_8

- **WSE: Wait State Enable (Code Label EBI_WSE)**

  0 = Wait state generation is disabled. No wait states are inserted.
  1 = Wait state generation is enabled.

- **PAGES: Page Size**

PAGES		Page Size	Active Bits in Base Address	Code Label
				EBI_PAGES
0	0	1M Byte	12 Bits (31-20)	EBI_PAGES_1M
0	1	4M Bytes	10 Bits (31-22)	EBI_PAGES_4M
1	0	16M Bytes	8 Bits (31-24)	EBI_PAGES_16M
1	1	64M Bytes	6 Bits (31-26)	EBI_PAGES_64M

- **TDF: Data Float Output Time**

TDF			Number of Cycles Added after the Transfer	Code Label
				EBI_TDF
0	0	0	0	EBI_TDF_0
0	0	1	1	EBI_TDF_1
0	1	0	2	EBI_TDF_2
0	1	1	3	EBI_TDF_3
1	0	0	4	EBI_TDF_4
1	0	1	5	EBI_TDF_5
1	1	0	6	EBI_TDF_6
1	1	1	7	EBI_TDF_7

- **BAT: Byte Access Type**

BAT	Selected BAT	Code Label
		EBI_BAT
0	Byte-write access type	EBI_BAT_BYTE_WRITE
1	Byte-select access type	EBI_BAT_BYTE_SELECT

- **CSEN: Chip Select Enable (Code Label `EBI_CSEN`)**

  0 = Chip select is disabled.
  1 = Chip select is enabled.

- **BA: Base Address (Code Label `EBI_BA`)**

  These bits contain the highest bits of the base address. If the page size is larger than 1M byte, the unused bits of the base address are ignored by the EBI decoder.

EBI Remap Control Register

Register Name:EBI_RCR

Access Type: Write-only

Absolute Address:0xFFE00020

Offset: 0x20

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
–	–	–	–	–	–	–	RCB

- **RCB: Remap Command Bit (Code Label `EBI_RCB`)**

  0 = No effect.
  1 = Cancels the remapping (performed at reset) of the page zero memory devices.

EBI Memory Control Register

Register Name:EBI_MCR

Access Type: Read/Write

Reset Value:0

Absolute Address:0xFFE00024

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
–	–	–	DRP	–	–	–	–

- **DRP: Data Read Protocol**

DRP	Selected DRP	Code Label
		`EBI_DRP`
0	Standard read protocol for all external memory devices enabled	`EBI_DRP_STANDARD`
1	Early read protocol for all external memory devices enabled	`EBI_DRP_EARLY`

APMC: Advanced Power Management Controller

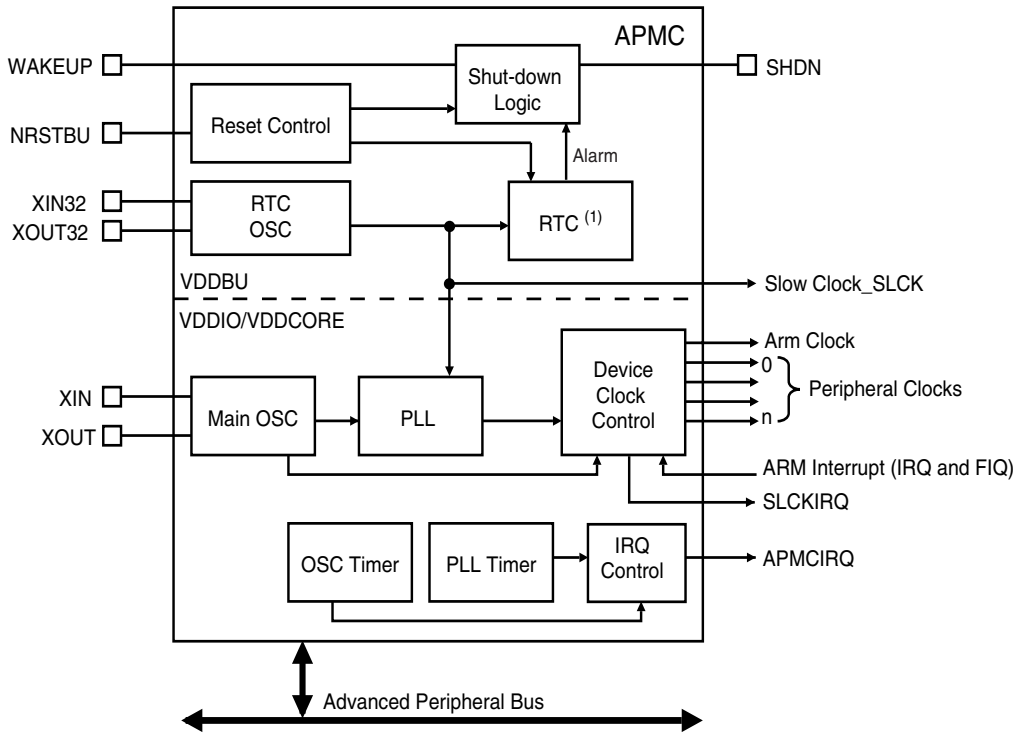
The AT91M55800A features an Advanced Power Management Controller, which optimizes both the power consumption of the device and the complete system. The APMC controls the clocking elements such as the oscillators and the PLL, the core and the peripheral clocks, and has the capability to control the system power supply.

Main Power is used throughout this document to identify the voltages powering the AT91M55800A and other components of the system, with the exception of the Battery Backup voltage, which is applied to the VDDBU. Main Power supplies VDDIO, VDDCORE and, if required, the analog voltage VDDA. A battery or battery capacitor generally supplies the Battery Backup Power.

The APMC consists of the following elements:

- The RTC Oscillator, which provides the Slow Clock at 32768 Hz.
- The Main Oscillator, which provides a clock that depends on the frequency of the crystal connected to the XIN and XOUT pins.
- The Phase Lock Loop.
- The ARM Core Clock Controller, which allows entry to the Idle Mode.
- The Peripheral Clock Controller, which conserves the power consumption of unused peripherals.
- The Master Clock Output Controller.
- The Shut-down Logic, which controls the Main Power.

Figure 30. APMC Module



Note: The RTC peripheral is described in a separate section.

Operating Modes

Five operating modes are supported by the APMC and offer different power consumption levels and event response latency times.

- Normal Mode:

  The Main Power supply is switched on; the ARM Core Clock is enabled and the peripheral clocks are enabled according to the application requirements.

- Idle Mode:

  The Main Power is switched on; the ARM Core Clock is disabled and waiting for the next interrupt (or a main reset); the peripheral clocks are enabled according to the application requirements and the PDC transfers are still possible.

- Slow Clock Mode:

  Similar to Normal Mode, but the Main Oscillator and the PLL are switched off to save power; the device core and peripheral run in Slow Clock Mode; Note that Slow Clock Mode is the mode selected after the reset.

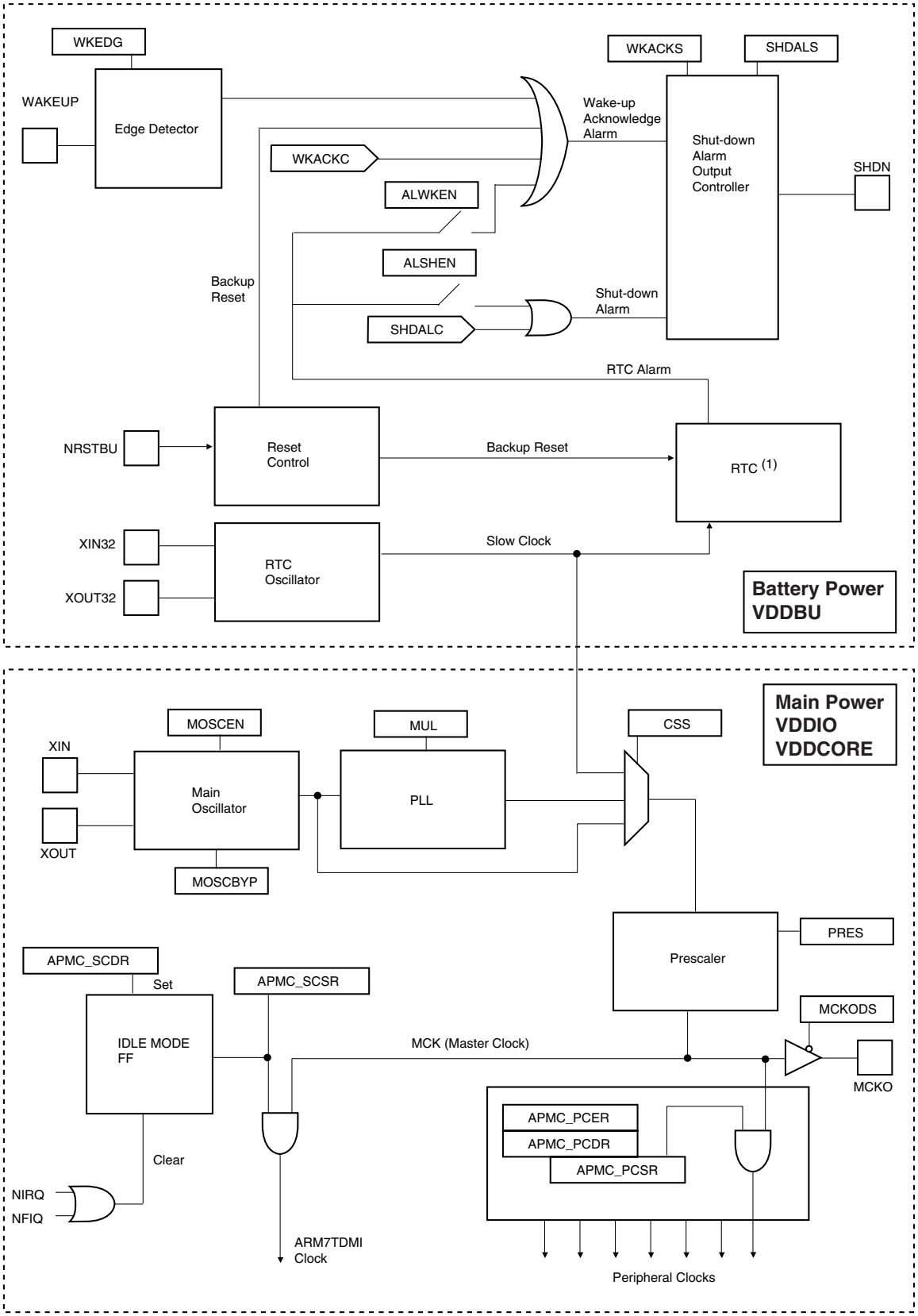
- Standby Mode:

  A combination of the Slow Clock Mode and the Idle Mode, which enables the processor to respond quickly to a wake-up event by keeping very low power consumption.

- Power-down Mode:

  The Main Power supply is turned off at the external power source until a programmable edge on the wake-up signal or a programmable RTC Alarm occurs.

Figure 31. APMC Block Diagram

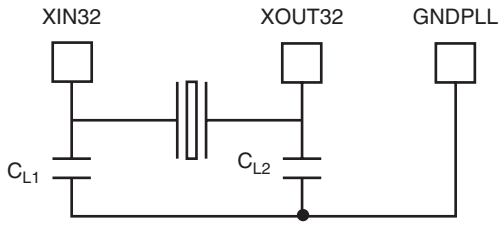


Note: 1. The RTC is described in another chapter

Slow Clock Generator

The AT91M55800A has a very low power 32 kHz oscillator powered by the backup battery voltage supplied on the VDDBU pins. The XIN32 and XOUT32 pins must be connected to a 32768 Hz crystal. The oscillator has been especially designed to connect to a 6 pF typical load capacitance crystal and does not require any external capacitor, as it integrates the XIN32 and XOUT32 capacitors to ground. For a higher typical load capacitance, two external capacitances must be wired as shown in Figure 32:

Figure 32. Higher Typical Load Capacitance



Backup Reset Controller

The backup reset controller initializes the logic supplied by the backup battery power. A simple RC circuit connected to the NRSTBU pin provides a power-on reset signal to the RTC and the shutdown logic. When the reset signal increases and as the startup time of the 32 kHz oscillator is around 300 ms, the AT91M55800A maintains the internal backup reset signal for 32768 oscillator clock cycles in order to guarantee the backup power supplied logic does not operate before the oscillator output is stabilized.

Alternatively, a reset supervisor can be connected to the NRSTBU pin in place of the RC.

Slow Clock

The Slow Clock is the only clock considered permanent in an AT91M55800A-based system and is essential in the operations of the APMC (Advanced Power Management Controller). In any use-case, a 32768 Hz crystal must be connected to the XIN32 and XOUT32 pins in order to ensure that the Slow Clock is present.

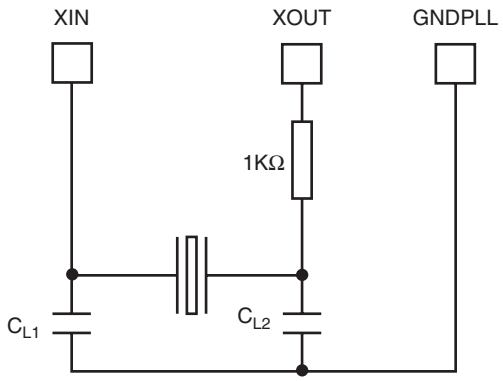
Clock Generator

The clock generator consists of the main oscillator, the PLL and the clock selection logic with its prescaler. It aims at selecting the Master Clock, called MCK throughout this datasheet. The clock generator also contains the circuitry needed to drive the MCKO pin with the master clock signal.

Main Oscillator

The Main Oscillator is designed for a 3 to 20 MHz fundamental crystal. The typical crystal connection is illustrated in Figure 33. The 1 kΩ resistor is only required for crystals with frequencies lower than 8 MHz. The oscillator contains 25 pF capacitances on each XIN and XOUT pin. Consequently, CL1 and CL2 can be removed when a crystal with a load capacitance of 12.5 pF is used.

Figure 33. Typical Crystal Connection of Main Oscillator



The Main Oscillator can be bypassed if the MOSCBYP bit in the Clock Generator Mode Register (APMC_CGMR) is set to 1. In this case, any frequency (up to the maximum specified in the electrical characteristics datasheet) can be input on the XIN pin. If the PLL is used, a minimum input frequency is required.

To minimize the power required to start up the system, the Main Oscillator is disabled after the reset. The software can deactivate the Main Oscillator to reduce the power consumption by clearing the MOSCEN bit in APMC_CGMR. The MOSCS (Main Oscillator Status) bit in APMC_SR is automatically cleared, indicating that the Main Oscillator is off.

Writing the MOSCEN bit in APMC_CGMR reactivates the Main Oscillator and loads the value written in the OSCOUNT field in APMC_CGMR in the oscillator counter. Then, the oscillator counter decrements every 8 clock cycles and when it reaches 0, the MOSCS bit is set and can provide an interrupt.

Phase Lock Loop

The Main Oscillator output signal feeds the phase lock loop, which aims at multiplying the frequency of its input signal by a number up to 64. This number is programmed in the MUL field of APMC_CGMR and the multiplication ratio is the programmed value plus one (MUL+1). If a null value is programmed into MUL, the PLL is automatically disabled to save power.

The PLL is disabled at reset to minimize the power consumption.

A start-up sequence must be executed to enable the PLL if it is disabled. This sequence is started by writing a new MUL value in APMC_CGMR. This automatically clears the LOCK bit in APMC_SR and loads the PLL counter with the value programmed in the PLLCOUNT field. Then, the PLL counter decrements at each Slow Clock cycle.

Note: Programming one in PLLCOUNT is the minimum allowed and guarantees at least 2 Slow Clock cycles before the lock bit is set. Programming *n* in PLLCOUNT guarantees (*n*+1) the delay of Slow Clock cycles. When the PLL Counter reaches 0, the LOCK bit in APMC_SR is set and can cause an interrupt. Programming MUL or PLLCOUNT before the LOCK bit is set may lead to unpredictable behavior.

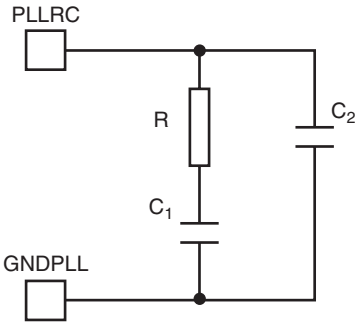
If the PLL multiplication is changed while the PLL is already active, the LOCK bit in APMC_SR is automatically cleared and the same sequence is restarted. The PLL is automatically bypassed while the frequency is changing (while LOCK is 0). If the Main Oscillator is reactivated at the same time the PLL is enabled, the LOCK bit is set only when both the Main Oscillator and the PLL are stabilized.

PLL Filter

The Phase Lock Loop has a dedicated PLLRC pin which must connect with an appropriate second order filter made up of one resistor and two capacitors. If the integrated PLL

is not used, it can remain disabled. The PLLRC pin must be grounded if the resistor and the capacitors need to be saved. The following figure shows a typical filter connection.

Figure 34. Typical Filter Connection



In order to obtain optimal results with a 16 MHz input frequency and a 32 MHz output frequency, the typical component values for the PLL filter are:

R = 287Ω - C1 = 680 nF - C2 = 68 nF

The lock time with these values is about 3.5 µs in this example.

Master Clock Selection

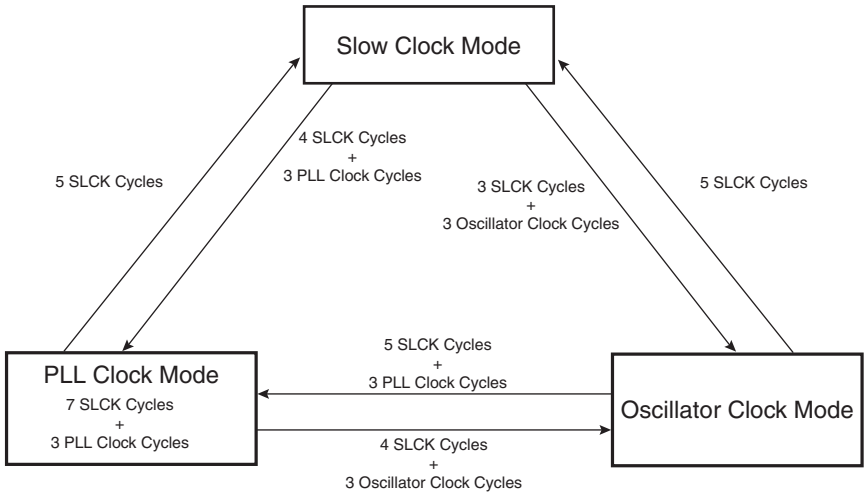
The MCK (Master Clock) can be selected through the CSS field in APMC_CGMR between the Slow Clock, the output of the Main Oscillator or the output of the PLL.

The following CSS field definitions are forbidden and the write operations are not taken into account by the APMC:

- deselect the Slow Clock if the Main Oscillator is disabled or its output is not stabilized
- disable the PLL without having first selected the Slow Clock or the Main Oscillator clock
- select the PLL clock and, in the same register, write disable the PLL
- select either the Main Oscillator or the PLL clocks and, in the same register, write disable the Main Oscillator
- disable the Main Oscillator without having first selected the Slow Clock

This clock switch is performed in some Slow Clocks and PLLs or Main Oscillator clock cycles as described in the state machine diagram below:

Figure 35. Clock Switch



Slow Clock Interrupt

The APMC also features the Slow Clock interrupt, allowing the user to detect when the Master Clock is actually switched to the Slow Clock. Switching from the Slow Clock to a higher frequency is generally performed safely, as the processor is running slower than the target frequency. However, switching from a high frequency to the Slow Clock requires the high frequency to be valid during the switch time. The Slow Clock interrupt permits the user to know exactly when the switch has been achieved, thus, when the Main Oscillator or the PLL can be disabled.

Prescaler

The prescaler is the last stage to provide the master clock. It permits the selected clock to be divided by a power of 2 between 1 and 64. The default value is 1 after the reset. The prescaler allows the microcontroller operating frequency to reach down to 512 Hz.

Precautions must be taken when defining a master clock lower than the Slow Clock, as some peripherals (RTC and APMC) can still operate at Slow Clock frequency. In this case, access to the peripheral registers that are updated at 32 kHz cannot be ensured.

Master Clock Output

The Master Clock can be output to the MCKO pad. The MCKO pad can be tri-stated to minimize power consumption by setting the bit MCKODS (Master Clock Output Disable) in APMC_CGMR (default is MCKO enabled).

System Clock

The AT91M55800A has only one system clock: the ARM Core clock. It can be enabled and disabled by writing to the System Clock Enable (APMC_SCER) and System Clock Disable Registers (APMC_SCDR). The status of the ARM Core clock (at least for debug purposes) can be read in the System Clock Status Register (APMC_SCSR).

The ARM Core clock is enabled after a reset and is automatically re-enabled by any enabled interrupt.

When the ARM Core clock is disabled, the current instruction is finished before the clock is stopped.

Note: Stopping the ARM Core does not prevent PDC transfers.

Peripheral Clocks

Each peripheral clock integrated in the AT91M55800A can be individually enabled and disabled by writing to the Peripheral Clock Enable (APMC_PCER) and Peripheral Clock Disable (APMC_PCDR) Registers. The status of the peripheral clocks can be read in the Peripheral Clock Status Register (APMC_PCSR).

When a peripheral clock is disabled, the clock is immediately stopped. When the clock is re-enabled, the peripheral resumes action where it left off.

In order to stop a peripheral, it is recommended that the system software waits until the peripheral has executed its last programmed operation before disabling the clock. This is to avoid data corruption or erroneous behavior of the system.

The peripheral clocks are automatically disabled after a reset.

The bits that control the peripheral clocks are the same as those that control the Interrupt Sources in the AIC.

Shut-down and Wake-up

The APMC (Advanced Power Management Controller) integrates shut-down and wake-up logic to control an external main power supply. This logic is supplied by the Battery Backup Power. This feature makes the Power-down mode possible.

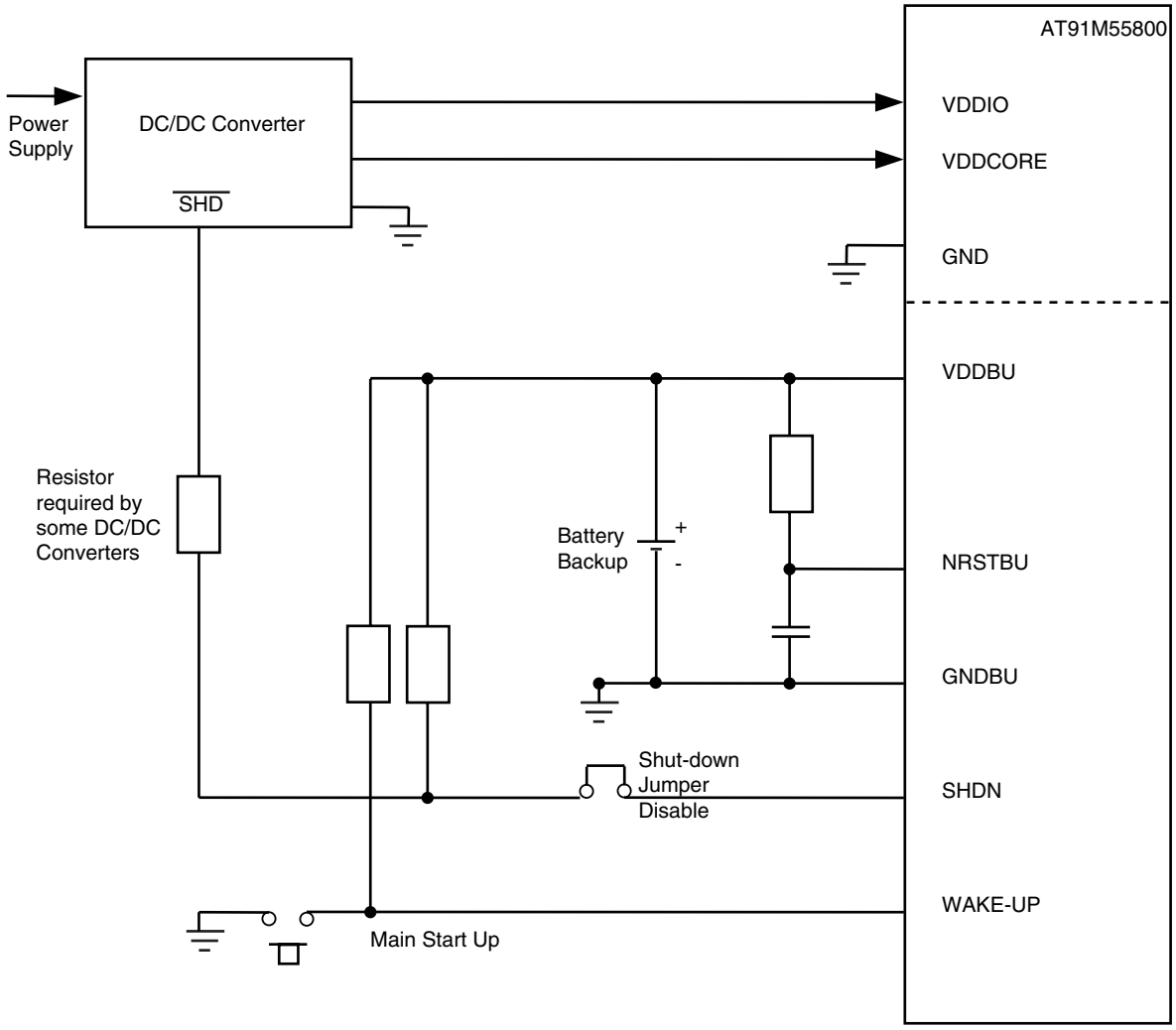
If the SHDN pin is connected to the shut-down pin of the main power supply, the Shut-down command (SHDALC) in APMC_PCR disables the main power. The shut-down input of the converter is generally pulled up or down by a resistor, depending on its active level.

There are 3 ways to exit Power-down mode and restart the main power:

- An alarm programmed in the RTC occurs and the bit ALWKEN in APMC_PMR is set.
- An edge defined by the field WKEDG in APMC_PMR occurs on the pin WAKEUP.
- The user opens the Shut-down line with an external jumper or push-button.

Figure 36 shows a typical application using the Shut-down and Wake-up features.

Figure 36. Shut-down and Wake-up Features



To accommodate the different types of main power supply available, and different signals that can command the shut-down of this device, tri-state, level 0 and level 1 are user-definable for the Shut-down pin. The Wake-up pin can be configured as detected on the positive or negative edge, and at high or low level. They are selected by the SHDALS and WKACKS fields in APMC_PMR.

Alarm

If the Shut-down feature is not used, the pin SHDN can be used as an Alarm Output Signal from the RTC Alarm. The Alarm State corresponds to Shut-down, and the Acknowledge or Non-Alarm State corresponds to Wake-up.

The alarm control logic is the same as that for Shut-down. The SHDALC command in APMC_PCR (defined by the field SHDALS in APMC_PMR) and the WKACKS command in APMC_PCR (defined by the field WKACKS field in APMC_PMR) control the SHDN pin.

The alarm can be positioned by an RTC Alarm and be acknowledged by a programmable edge on the WAKEUP pin. The Backup Reset initializes the logic in Non-Alarm State.

First Start-up Sequence

At initial startup, or after VDDBU has been disconnected, the battery-supplied logic must be initialized.

The Battery Backup Reset sets the following default state:

- Shut-down Logic
  Initialized in the Wake-up state (or Non Alarm)
- The Power Mode Register
  Shut-down defines SHDN as level 0 (SHDALS = 1)
  Wake-up defines SHDN as tri-state (WKACKS = 0)
- The Real-time Clock Configuration and Data Registers

A simple RC network can be used as a power-on reset for the battery supply.

The pin SHDN is tri-stated by default. An external resistor must hold the main power supply shut-down pin in the inactive state. The shut-down logic can be programmed with the correct active level of the power supply shut-down input during the first start-up sequence.

The first time the system is powered up, the SHDN pin is tri-stated because different power supplies use different logic levels for their shut-down input signals. To minimize backup battery power consumption, there is no internal pull-up or pull-down on this signal.

If the power supply needs a logic level on its shut-down input in order to start the main power supply then an external “Force Start Up” jumper is required to provide this level.

The jumper provides the necessary level on the SHDN to maintain the power supply when the AT91 boots, and it can be removed until the next loss of battery power.

APMC User Interface

Base Address:0xFFFF4000 (Code Label `APMC_BASE`)

Table 7. APMC Memory Map

Offset	Register	Name	Access	Main Reset	Backup Reset
0x00	System Clock Enable Register	APMC_SCER	W	–	–
0x04	System Clock Disable Register	APMC_SCDR	W	–	–
0x08	System Clock Status Register	APMC_SCSR	R	0x1	–
0x0C	Reserved	–	–	–	–
0x10	Peripheral Clock Enable Register	APMC_PCER	W	–	–
0x14	Peripheral Clock Disable Register	APMC_PCDR	W		–
0x18	Peripheral Clock Status Register	APMC_PCSR	R	0	–
0x1C	Reserved	–	W		–
0x20	Clock Generator Mode Register	APMC_CGMR	R/W	0	–
0x24	Reserved	–	–	–	–
0x28	Power Control Register	APMC_PCR	W		–
0x2C	Power Mode Register	APMC_PMR	R/W		0x1
0x30	Status Register	APMC_SR	R	–	–
0x34	Interrupt Enable Register	APMC_IER	W	–	–
0x38	Interrupt Disable Register	APMC_IDR	W	–	–
0x3C	Interrupt Mask Register	APMC_IMR	R	0	–

APMC System Clock Enable Register

Register Name:APMC_SCER

Access Type: Write-only

Offset: 0x00

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 Enable Bit**

  0 = No effect.
  1 = Enables the System Clock.

APMC System Clock Disable Register

Register Name:APMC_SCDR

Access Type: Write-only

Offset: 0x04

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 Disable Bit**

  0 = No effect.
  1 = Disables the System Clock.

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.

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.

APMC Peripheral Clock Disable Register

Register Name:APMC_PCDR

Access Type: Write-only

Offset: 0x14

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 Disable (per peripheral)**

  0 = No effect.
  1 = Disables the peripheral clock.

APMC Peripheral Clock Status Register

Register Name:APMC_PCSR

Access Type: Read-only

Reset Value:0x0

Offset: 0x18

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 Status (per peripheral)**

  0 = The peripheral clock is disabled.
  1 = The peripheral clock is enabled.

APMC Clock Generator Mode Register

Register Name:APMC_CGMR

Access Type: Read/Write

Reset Value:0x0

Offset: 0x20

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

- **MOSCBYP: Main Oscillator Bypass (Code Label `APMC_MOSC_BYP`)**

  0 = Crystal must be connected between XIN and XOUT.
  1 = External clock must be provided on XIN.

- **MOSCEN: Main Oscillator Enable (Code Label `APMC_MOSC_EN`)**

  0 = Main Oscillator is disabled.
  1 = Main Oscillator is enabled.

Note: When operating in Bypass Mode, the Main Oscillator must be disabled. MOSCEN and MOSCBYP bits must never be set together.

- **MCKODS: Master Clock Output Disable (Code Label `APMC_MCKO_DIS`)**

  0 = The MCKO pin is driven with the Master Clock (MCK).
  1 = The MCKO pin is tri-stated.

- **PRES: Prescaler Selection**

PRES			Prescaler Selected	Code Label
0	0	0	None. Prescaler Output is the selected clock.	APMC_PRES_NONE
0	0	1	Selected clock is divided by 2	APMC_PRES_DIV2
0	1	0	Selected clock is divided by 4	APMC_PRES_DIV4
0	1	1	Selected clock is divided by 8	APMC_PRES_DIV8
1	0	0	Selected clock is divided by 16	APMC_PRES_DIV16
1	0	1	Selected clock is divided by 32	APMC_PRES_DIV32
1	1	0	Selected clock is divided by 64	APMC_PRES_DIV64
1	1	1	Reserved	–

- **MUL: Phase Lock Loop Factor**

  0 = The PLL is deactivated, reducing power consumption to a minimum.
  1 - 63 = The PLL output is at a higher frequency (MUL+1) than the input if the bit lock is set in APMC_SR.

- **CSS: Clock Source Selection**

CSS		Clock Source Selection	Code Label
0	0	Low-frequency clock provided by the RTC	APMC_CSS_LF
0	1	Main oscillator Output or external clock	APMC_CSS_MOSC
1	0	Phase Lock Loop Output	APMC_CSS_PLL
1	1	Reserved	–

- **OSCOUNT: Main Oscillator Counter**

  Specifies the number of 32,768 Hz divided by 8 clock cycles for the main oscillator start-up timer to count before the main oscillator is stabilized, after the oscillator is enabled. The main oscillator counter is a down-counter which is preloaded with the OSCOUNT value when the MOSCEN bit in the Clock Generator Mode register (CGMR) is set, but only if the OSCOUNT value is different from 0x0.

- **PLLCOUNT: PLL Lock Counter**

  Specifies the number of 32,768 Hz clock cycles for the PLL lock timer to count before the PLL is locked, after the PLL is started. The PLL counter is a down-counter which is preloaded with the PLLCOUNT value when the MUL field in the Clock Generator Mode register (CGMR) is modified, but only if the MUL value is different from 0 (PLL disabled) and also the PLLCOUNT value itself different from 0x0. PLLCOUNT must be loaded with a minimum value of 2 in order to guarantee a time of at least one slow clock period.

APMC Power Control Register

Register Name:APMC_PCR

Access Type: Write-only

Offset: 0x28

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
–	–	–	–	–	–	WKACKC	SHDALC

- **SHDALC: Shut-down or Alarm Command (Code Label `APMC_SHDALC`)**

  0 = No effect.
  1 = Configures the SHDN pin as defined by the field SHDALS in APMC_PMR.

- **WKACKC: Wake-up or Alarm Acknowledge Command (Code Label `APMC_WKACKC`)**

  0 = No effect.
  1 = Configures the SHDN pin as defined by the field WKACKS in APMC_PMR.

Note: If both the SHDALC and WKACKS bits are set, the WKACKS command has priority.

APMC Power Mode Register

Register Name:APMC_PMR

Access Type: Read/Write

Backup Reset Value:0x1

Offset: 0x2C

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
WKEDG		ALSHEN	ALWKEN	WKACKS		SHDALS	

- **SHDALS: Shut-down or Alarm Output Selection**

  This field defines the state of the SHDAL pin when shut-down or alarm is requested.

SHDALS		Shut-down or Alarm Output Selected	Code Label
0	0	Tri-stated	APMC_SHDALS_OUT_TRIS
0	1	Level 0	APMC_SHDALS_OUT_LEVEL0
1	0	Level 1	APMC_SHDALS_OUT_LEVEL1
1	1	Reserved	–

- **WKACKS: Wake-up or Alarm Acknowledge Output Selection**

  This field defines the state of the WKACKS pin when wake-up or alarm acknowledge is requested.

WKACKS		Wake-up or Alarm Acknowledge Output Selected	Code Label
0	0	Tri-stated	APMC_WKACKS_OUT_TRIS
0	1	Level 0	APMC_WKACKS_OUT_LEVEL_0
1	0	Level 1	APMC_WKACKS_OUT_LEVEL_1
1	1	Reserved	–

- **ALWKEN: Alarm Wake-up Enable (Code Label `APMC_WKEN`)**

  0 = The alarm from the RTC has no wake-up effect.
  1 = The alarm from the RTC commands a wake-up.

- **ALSHEN: Alarm Shut-down Enable (Code Label `APMC_ALSHEN`)**

  0 = The alarm from the RTC has no shut-down effect.
  1 = If ALWKEN is 0, the alarm from the RTC commands a shut-down.

- **WKEDG: Wake-up Input Edge Selection**

  This field defines the edge to detect on the Wake-up pin (WAKEUP) to provoke a wake-up.

WKEDG		Wake-up Input Edge Selection	Code Label
0	0	None. No edge is detected on wake-up.	APMC_WKEDG_NONE
0	1	Positive edge	APMC_WKEDG_POS_EDG
1	0	Negative edge	APMC_WKEDG_NEG_EDG
1	1	Both edges	APMC_WKEDG_BOTH_EDG

APMC Status Register

Register Name:APMC_SR

Access Type: Read-only

Offset: 0x30

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
–	–	–	–	–	–	LOCK	MOSCS

- **MOSCS: Main Oscillator Status (Code Label APMC_MOSCS)**

  0 = Main Oscillator output signal is not stabilized or the Main Oscillator is disabled.
  1 = The Main Oscillator is enabled and its output is stabilized. Actually, this bit indicates that the Main Oscillator counter reached 0.

- **LOCK: PLL Lock Status (Code Label APMC_PLL_LOCK)**

  0 = PLL output signal or main oscillator output signal is not stabilized, or the main oscillator is disabled.
  1 = Main Oscillator is enabled and its output is stabilized and the PLL output signal is stabilized. Actually, this bit is set when the PLL Lock Counter reaches 0.

APMC Interrupt Enable Register

Register Name:APMC_IER

Access Type: Write-only

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
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	LOCK	MOSCS

- **MOSCS: Main Oscillator Interrupt Enable (Code Label `APMC_MOSCS`)**

  0 = No effect.
  1 = Enables the Main Oscillator Stabilized Interrupt.
- **LOCK: PLL Lock Interrupt Enable (Code Label `APMC_PLL_LOCK`)**

  0 = No effect.
  1 = Enables the PLL Lock Interrupt.

APMC Interrupt Disable Register

Register Name:APMC_IDR

Access Type: Write-only

Offset: 0x38

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
–	–	–	–	–	–	LOCK	MOSCS

- **MOSCS: Main Oscillator Interrupt Disable (Code Label `APMC_MOSCS`)**

  0 = No effect.
  1 = Disables the Main Oscillator Stabilized Interrupt.
- **LOCK: PLL Lock Interrupt Disable (Code Label `APMC_PLL_LOCK`)**

  0 = No effect.
  1 = Disables the PLL Lock Interrupt.

APMC Interrupt Mask Register

Register Name:APMC_IMR

Access Type: Read-only

Reset Value:0x0

Offset: 0x3C

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
–	–	–	–	–	–	LOCK	MOSCS

- **MOSCS: Main Oscillator Interrupt Mask (Code Label `APMC_MOSCS`)**

  0 = The Main Oscillator Interrupt is disabled.
  1 = The Main Oscillator Interrupt is enabled.

- **LOCK: PLL Lock Interrupt Mask (Code Label `APMC_PLL_LOCK`)**

  0 = The PLL Lock Interrupt is disabled.
  1 = The PLL Lock Interrupt is enabled.

RTC: Real-time Clock

The AT91M55800A features a Real-time Clock (RTC) peripheral that is designed for very low power consumption. It combines a complete time-of-day clock with alarm and a two-hundred year Gregorian calendar, complemented by a programmable periodic interrupt.

The time and calendar values are coded in Binary-Coded Decimal (BCD) format. The time format can be 24-hour mode or 12-hour mode with an AM/PM indicator.

Updating time and calendar fields and configuring the alarm fields is performed by a parallel capture on the 32-bit data bus. An entry control is performed to avoid loading registers with incompatible BCD format data or with an incompatible date according to the current month/ year/century.

Year 2000 Conformity

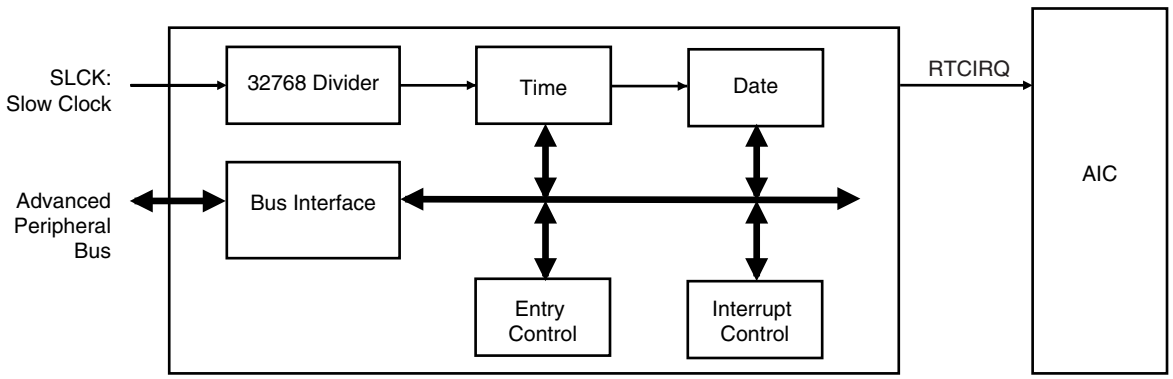
The Real-time Clock complies fully with the Year 2000 Conformity Requirements as stated in the British Standards Institution Document Ref BSI-DISC PD2000-1: “Year 2000 conformity shall mean that neither performance nor functionality is affected by dates prior to, during and after the year 2000”.

It has been tested to be compliant with the four associated rules:

1. No value for current date will cause any interruption in operation.
2. Date-based functionality must behave consistently for dates prior to, during and after year 2000.
3. In all interfaces and data storage, the century in any date must be specified either explicitly or by unambiguous algorithms or inferencing rules.
4. Year 2000 must be recognized as a leap year.

The RTC represents the year as a four-digit number (1998, 1999, 2000, 2001, etc.) so that the century is unambiguously identified, in accordance with Rule 3.

Figure 37. RTC Block Diagram



Functional Description

The RTC provides a full Binary-Coded Decimal (BCD) clock which includes century (19/20), year (with leap years), month, date, day, hours, minutes and seconds.

The valid year range is 1900 to 2099, a two-hundred year Gregorian calendar achieving full Y2K compliance.

The RTC can operate in 24-hour mode or in 12-hour mode with an AM/PM indicator.

Corrections for leap years are included (all years divisible by 4 being leap years, including year 2000). This is correct up to the year 2099.

Timing

The RTC is updated in real-time at one second intervals in normal mode for the counters of seconds, at 1 minute intervals for the counter of minutes and so on.

Due to the asynchronous operation of the RTC with respect to the rest of the chip, to be certain that the value read in the RTC registers (century, year, month, date, day, hours, minutes, seconds) are valid and stable, it is necessary to read these registers twice. If the data is the same both times, then it is valid. Therefore, a minimum of two and a maximum of three accesses is required.

Alarm

The RTC has five programmable fields with which to program an alarm: MONTH and DATE in the Calendar Alarm Register (RTC_CAR), and SEC, MIN and HOUR in the Time Alarm Register (RTC_TAR). Each of these fields can be enabled or disabled using the bits MTHEN, DATEN, SECEN, MINEN, HOUREN to match the alarm condition.

- If all the fields are enabled, an alarm flag is generated (the corresponding flag is asserted and an interrupt generated if enabled) at a given month, date, hour, minute and second.
- If only the “seconds” field is enabled, then an alarm is generated every minute.
- Depending on the combination of fields enabled, a large number of possibilities are available to the user ranging from minutes to 365/366 days.

Error Checking

A verification on user interface data is performed when accessing the century, year, month, date, day, hours, minutes, seconds and alarms. A check is performed on illegal BCD entries such as illegal date of the month with regards to the year and century configured.

If one of the time fields is not correct, the data is not loaded into the register/counter and a flag is set in the Valid Entry Register (RTC_VER). This flag cannot be reset by the user. It is reset as soon as an acceptable value is programmed. This avoids any further side effects in the hardware. The same processing is done for the alarm.

The following checks are processed:

1. Century (check if it is in range 19 - 20)
2. Year (BCD entry check)
3. Date (check range 01 - 31)
4. Month (check if it is in BCD range 01 - 12,
   check validity regarding “date”)
5. Day (check range 1 - 7)
6. Hour (BCD check, in 24-hour mode check range 00 - 23 and check that AM/PM flag is not set if RTC is set in 24-Hour mode, in 12-Hour mode check range 01 - 12)
7. Minute (check BCD and range 00 - 59)

8. Second (check BCD and range 00 - 59)

Note: If the 12-hour mode is selected by means of the RTC_MODE register, a 12-hour value can be programmed and the returned value on RTC_TIME will be the corresponding 24-hour value. The entry control checks the value of the AM/PM indicator (bit 22 of RTC_TIME register) to determine the range to be checked.

Updating Time/Calendar

To update any of the time/calendar fields, the user must first stop the RTC by setting the corresponding field in the Mode Register (RTC_MR). Bit UPDTIM must be set to update time fields (hour, minute, second) and bit UPDCAL must be set to update calendar fields (century, year, month, date, day).

Then the user must poll or wait for the interrupt (if enabled) of bit ACKUPD in the Status Register (RTC_SR). Once the bit reads 1 (the user must clear this status bit by writing ACKUPD to 1 in RTC_SCR), the user can write to the appropriate register.

Once the update is finished, the user must reset (0) UPDTIM and/or UPDCAL in the Mode Register (RTC_MR).

When programming the calendar fields, the time fields remain enabled. This avoids a time slip in case the user stays in the calendar update phase for several tens of seconds or more.

RTC User Interface

Base Address:0xFFFB8000 (Code Label `RTC_BASE`)

Table 8. RTC Memory Map

Offset	Register	Name	Access	Reset State
0x0000	Mode Register	RTC_MR	Read/Write	0x00000000
0x0004	Hour Mode Register	RTC_HMR	Read/Write	0x00000000
0x0008	Time Register	RTC_TIMR	Read/Write	0x00000000
0x000C	Calendar Register	RTC_CALR	Read/Write	0x01819819
0x0010	Time Alarm Register	RTC_TAR	Read/Write	0x00000000
0x0014	Calendar Alarm Register	RTC_CAR	Read/Write	0x00000000
0x0018	Status Register	RTC_SR	Read-only	0x00000000
0x001C	Status Clear Register	RTC_SCR	Write-only	–
0x0020	Interrupt Enable Register	RTC_IER	Write-only	–
0x0024	Interrupt Disable Register	RTC_IDR	Write-only	–
0x0028	Interrupt Mask Register	RTC_IMR	Read-only	0x00000000
0x002C	Valid Entry Register	RTC_VER	Read-only	0x00000000

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	–

RTC Hour Mode Register

Register Name:RTC_HMR

Access Type: Read/Write

Reset State:0x0

Offset: 0x04

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
–	–	–	–	–	–	–	HRMOD

- **HRMOD: 12/24 Hour Mode**

HRMOD	Selected HRMOD	Code Label
0	24-Hour mode is selected	RTC_24_HRMOD
1	12-Hour mode is selected	RTC_12_HRMOD

RTC Time Register

Register Name:RTC_TIMR

Access Type: Read/Write

Reset State:0x0

Offset: 0x08

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

- **SEC: Current Second (Code Label `RTC_SEC`)**

  The range that can be set is 0 - 59 (BCD).
  The lowest four bits encode the units. The higher bits encode the tens.
- **MIN: Current Minute (Code Label `RTC_MIN`)**

  The range that can be set is 0-59 (BCD).
  The lowest four bits encode the units. The higher bits encode the tens.
- **HOUR: Current Hour (Code Label `RTC_HOUR`)**

  The range that can be set is 1 - 12 (BCD) in 12-hour mode or 0 - 23 (BCD) in 24-hour mode.
- **AMPM: Ante Meridiem Post Meridiem Indicator (Code Label `RTC_AMPM`)**

  This bit is the AM/PM indicator in 12-hour mode. It must be written at 0 if HRMOD in RTC_HMR defines 24-Hour mode.
  0 = AM.
  1 = PM.

RTC Calendar Register

Register Name:RTC_CALR

Access Type: Read/Write

Reset State:0x01819819

Offset: 0x0C

31	30	29	28	27	26	25	24
–	–	DATE					

23	22	21	20	19	18	17	16
DAY			MONTH				

15	14	13	12	11	10	9	8
YEAR							

7	6	5	4	3	2	1	0
–	–	CENT					

- **CENT: Current Century (Code Label `RTC_CENT`)**

  The range that can be set is 19 - 20 (BCD).
  The lowest four bits encode the units. The higher bits encode the tens.
- **YEAR: Current Year (Code Label `RTC_YEAR`)**

  The range that can be set is 00 - 99 (BCD).
  The lowest four bits encode the units. The higher bits encode the tens.
- **MONTH: Current Month (Code Label `RTC_MONTH`)**

  The range that can be set is 01 - 12 (BCD).
  The lowest four bits encode the units. The higher bits encode the tens.
- **DAY: Current Day (Code Label `RTC_DAY`)**

  The range that can be set is 1 - 7 (BCD).
  The significance of the number (which number represents which day) is user defined as it has no effect on the date counter.
- **DATE: Current Date (Code Label `RTC_DATE`)**

  The range that can be set is 01 - 31 (BCD).
  The lowest four bits encode the units. The higher bits encode the tens.

RTC Time Alarm Register

Register Name:RTC_TAR

Access Type: Read/Write

Reset State:0x0

Offset: 0x10

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

- **SEC: Second Alarm**

  This field is the alarm field corresponding to the BCD-coded second counter.
- **SECEN: Second Alarm Enable**

SECEN	Selected SECEN	Code Label
0	The second matching alarm is disabled.	RTC_SEC_ALARM_DIS
1	The second matching alarm is enabled.	RTC_SEC_ALARM_EN

- **MIN: Minute Alarm**

  This field is the alarm field corresponding to the BCD-coded minute counter.
- **MINEN: Minute Alarm Enable**

MINEN	Selected MINEN	Code Label
0	The minute matching alarm is disabled.	RTC_MIN_ALARM_DIS
1	The minute matching alarm is enabled.	RTC_MIN_ALARM_EN

- **HOUR: Hour Alarm**

  This field is the alarm field corresponding to the BCD-coded hour counter.
- **AMPM: AM/PM Indicator**

  This bit is the AM/PM indicator in 12-Hour mode. It must be written at 0 if HRMOD in RTC_HMR defines 24-Hour mode.
- **HOUREN: Hour Alarm Enable**

HOUREN	Selected HOUREN	Code Label
0	The hour matching alarm is disabled.	RTC_HOUR_ALARM_DIS
1	The hour matching alarm is enabled.	RTC_HOUR_ALARM_EN

RTC Calendar Alarm Register

Register Name:RTC_CAR

Access Type: Read/Write

Reset State:0x0

Offset: 0x14

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

- **MONTH: Month Alarm**

  This field is the alarm field corresponding to the BCD-coded month counter.
- **MTHEN: Month Alarm Enable**

MTHEN	Selected MTHEN	Code Label
0	The month matching alarm is disabled.	RTC_MONTH_ALARM_DIS
1	The month matching alarm is enabled.	RTC_MONTH_ALARM_EN

- **DATE: Date Alarm**

  This field is the alarm field corresponding to the BCD-coded date counter.
- **DATEN: Date Alarm Enable**

DATEN	Selected DATEN	Code Label
0	The date matching alarm is disabled.	RTC_DATE_ALARM_DIS
1	The date matching alarm is enabled.	RTC_DATE_ALARM_EN

RTC Status Register

Register Name:RTC_SR

Access Type: Read-only

Reset State:0x0

Offset: 0x18

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 for Update (Code Label `RTC_ACKUPD`)**

  0 = Time and Calendar registers cannot be updated.
  1 = Time and Calendar registers can be updated.
- **ALARM: Alarm Flag (Code Label `RTC_ALARM`)**

  0 = No alarm matching condition occurred.
  1 = An alarm matching condition has occurred.
- **SEC: Second Event (Code Label `RTC_SEC`)**

  0 = No second event has occurred since the last clear.
  1 = At least one second event has occurred since the last clear.
- **TIMEV: Time Event (Code Label `RTC_TIMEV`)**

  0 = No time event has occurred since the last clear.
  1 = At least one time event has occurred since the last clear.
  The time event is selected in the TEVSEV field in RTC_CR and can be any one of the following events: minute change, hour change, noon, midnight (day change).
- **CALEV: Calendar Event (Code Label `RTC_CALEV`)**

  0 = No calendar event has occurred since the last clear.
  1 = At least one calendar event has occurred since the last clear.
  The calendar event is selected in the CEVSEL field in RTC_CR and can be any one of the following events: week change, month change, year change.

RTC Status Clear Register

Register Name:RTC_SCR

Access Type: Write-only

Offset: 0x1C

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 for Update Interrupt Clear (Code Label `RTC_ACKUPD`)**

  0 = No effect.
  1 = Clears Acknowledge for Update status bit.
- **ALARM: Alarm Flag Interrupt Clear (Code Label `RTC_ALARM`)**

  0 = No effect.
  1 = Clears Alarm Flag bit.
- **SEC: Second Event Interrupt Clear (Code Label `RTC_SEC`)**

  0 = No effect.
  1 = Clears Second Event bit.
- **TIMEV: Time Event Interrupt Clear (Code Label `RTC_TIMEV`)**

  0 = No effect.
  1 = Clears Time Event bit.
- **CALEV: Calendar Event Interrupt Clear (Code Label `RTC_CALEV`)**

  0 = No effect.
  1 = Clears Calendar Event bit.

RTC Interrupt Enable Register

Register Name:RTC_IER

Access Type: Write-only

Offset: 0x20

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 Enable (Code Label `RTC_ACKUPD`)**

  0 = No effect.
  1 = The acknowledge for update interrupt is enabled.
- **ALARM: Alarm Interrupt Enable (Code Label `RTC_ALARM`)**

  0 = No effect.
  1 = The alarm interrupt is enabled.
- **SEC: Second Event Interrupt Enable (Code Label `RTC_SEC`)**

  0 = No effect.
  1 = The second periodic interrupt is enabled.
- **TIMEV: Time Event Interrupt Enable (Code Label `RTC_TIMEV`)**

  0 = No effect.
  1 = The selected time event interrupt is enabled.
- **CALEV: Calendar Event Interrupt Enable (Code Label `RTC_CALEV`)**

  0 = No effect.
  1 = The selected calendar event interrupt is enabled.

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.

RTC Interrupt Mask Register

Register Name:RTC_IMR

Access Type: Read-only

Reset State:0x0

Offset: 0x28

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 Mask (Code Label `RTC_ACKUPD`)**

  0 = The acknowledge for update interrupt is disabled.
  1 = The acknowledge for update interrupt is enabled.
- **ALARM: Alarm Interrupt Mask (Code Label `RTC_ALARM`)**

  0 = The alarm interrupt is disabled.
  1 = The alarm interrupt is enabled.
- **SEC: Second Event Interrupt Mask (Code Label `RTC_SEC`)**

  0 = The second periodic interrupt is disabled.
  1 = The second periodic interrupt is enabled.
- **TIMEV: Time Event Interrupt Mask (Code Label `RTC_TIMEV`)**

  0 = The selected time event interrupt is disabled.
  1 = The selected time event interrupt is enabled.
- **CALEV: Calendar Event Interrupt Mask (Code Label `RTC_CALEV`)**

  0 = The selected calendar event interrupt is disabled.
  1 = The selected calendar event interrupt is enabled.

RTC Valid Entry Register

Register Name:RTC_VER

Access Type: Read-only

Reset State:0x0

Offset: 0x2C

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
–	–	–	–	NVCAL	NVTAL	NVC	NVT

- **NVT: Non-Valid Time (Code Label `RTC_NVT`)**

  0 = No invalid data has been detected in RTC_TIMR.
  1 = RTC_TIMR has contained invalid data since it was last programmed.
- **NVC: Non-Valid Calendar (Code Label `RTC_NVC`)**

  0 = No invalid data has been detected in RTC_CALR.
  1 = RTC_CALR has contained invalid data since it was last programmed.
- **NVTAL: Non-Valid Time Alarm (Code Label `RTC_NVTAL`)**

  0 = No invalid data has been detected in RTC_TAR.
  1 = RTC_TAR has contained invalid data since it was last programmed.
- **NVCAL: Non-Valid Calendar Alarm (Code Label `RTC_NVCAL`)**

  0 = No invalid data has been detected in RTC_CAR.
  1 = RTC_CAR has contained invalid data since it was last programmed.

WD: Watchdog Timer

The AT91M55800A has an internal Watchdog Timer that can be used to prevent system lock-up if the software becomes trapped in a deadlock.

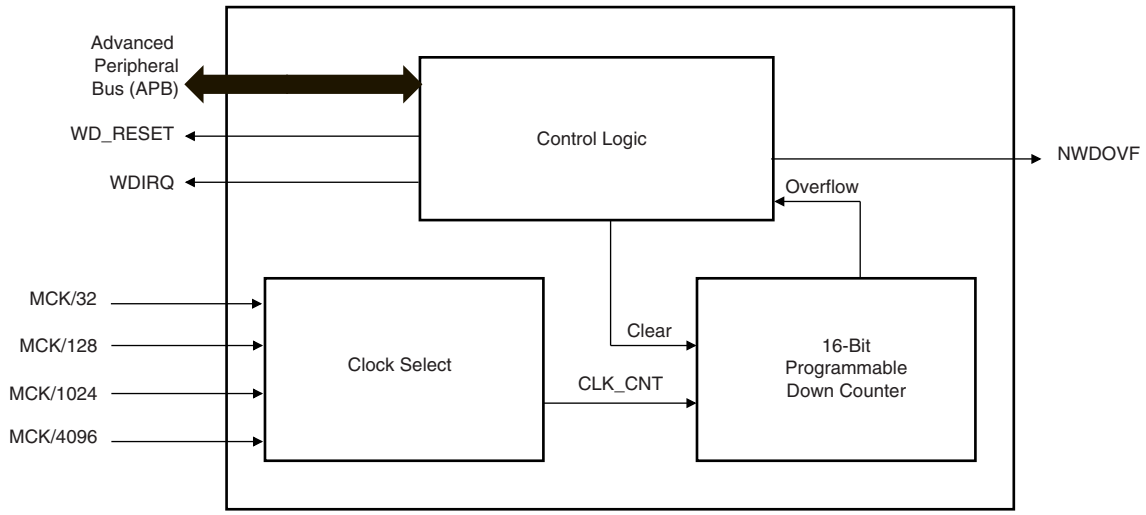
In normal operation the user reloads the watchdog at regular intervals before the timer overflow occurs. If an overflow does occur, the watchdog timer generates one or a combination of the following signals, depending on the parameters in WD_OMR (Overflow Mode Register):

- If RSTEN is set, an internal reset is generated (WD_RESET as shown in Figure 38).
- If IRQEN is set, a pulse is generated on the signal WDIRQ which is connected to the Advanced Interrupt Controller
- If EXTEN is set, a low level is driven on the NWDOVF signal for a duration of 8 MCK cycles.

The watchdog timer has a 16-bit down counter. Bits 12 - 15 of the value loaded when the watchdog is restarted are programmable using the HPVC parameter in WD_CMR (Clock Mode). Four clock sources are available to the watchdog counter: MCK/32, MCK/128, MCK/1024 or MCK/4096. The selection is made using the WDCLKS parameter in WD_CMR. This provides a programmable time-out period of 4 ms to 8 sec. with a 33 MHz system clock.

All write accesses are protected by control access keys to help prevent corruption of the watchdog should an error condition occur. To update the contents of the mode and control registers it is necessary to write the correct bit pattern to the control access key bits at the same time as the control bits are written (the same write access).

Figure 38. Watchdog Timer Block Diagram



WD User Interface

WD Base Address: 0xFFFF8000 (Code Label `WD_BASE`)

Table 9. WD Memory Map

Offset	Register	Name	Access	Reset State
0x00	Overflow Mode Register	WD_OMR	Read/Write	0
0x04	Clock Mode Register	WD_CMR	Read/Write	0
0x08	Control Register	WD_CR	Write-only	–
0x0C	Status Register	WD_SR	Read-only	0

WD Overflow Mode Register

Name: WD_OMR

Access: Read/Write

Reset Value:0

Offset: 0x00

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

- **WDEN: Watchdog Enable (Code Label `WD_WDEN`)**

  0 = Watchdog is disabled and does not generate any signals.
  1 = Watchdog is enabled and generates enabled signals.
- **RSTEN: Reset Enable (Code Label `WD_RSTEN`)**

  0 = Generation of an internal reset by the Watchdog is disabled.
  1 = When overflow occurs, the Watchdog generates an internal reset.
- **IRQEN: Interrupt Enable (Code Label `WD_IRQEN`)**

  0 = Generation of an interrupt by the Watchdog is disabled.
  1 = When overflow occurs, the Watchdog generates an interrupt.
- **EXTEN: External Signal Enable (Code Label `WD_EXTEN`)**

  0 = Generation of a pulse on the pin NWDOVF by the Watchdog is disabled.
  1 = When an overflow occurs, a pulse on the pin NWDOVF is generated.
- **OKEY: Overflow Access Key (Code Label `WD_OKEY`)**

  Used only when writing WD_OMR. OKEY is read as 0.
  0x234 = Write access in WD_OMR is allowed.
  Other value = Write access in WD_OMR is prohibited.

WD Clock Mode Register

Name: WD_CMR

Access: Read/Write

Reset Value:0

Offset: 0x04

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

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

15	14	13	12	11	10	9	8
CKEY							

7	6	5	4	3	2	1	0
CKEY	–	HPCV				WDCLKS	

- **WDCLKS: Clock Selection**

WDCLKS		Clock Selected	Code Label
			WD_WDCLKS
0	0	MCK/32	WD_WDCLKS_MCK32
0	1	MCK/128	WD_WDCLKS_MCK128
1	0	MCK/1024	WD_WDCLKS_MCK1024
1	1	MCK/4096	WD_WDCLKS_MCK4096

- **HPCV: High Pre-load Counter Value (Code Label `WD_HPCV`)**

  Counter is preloaded when watchdog counter is restarted with bits 0 to 11 set (FFF) and bits 12 to 15 equaling HPCV.
- **CKEY: Clock Access Key (Code Label `WD_CKEY`)**

  Used only when writing WD_CMR. CKEY is read as 0.
  0x06E: Write access in WD_CMR is allowed.
  Other value: Write access in WD_CMR is prohibited.

WD Control Register

Name: WD_CR

Access: Write-only

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
RSTKEY							

7	6	5	4	3	2	1	0
RSTKEY							

- **RSTKEY: Restart Key (Code Label `WD_RSTKEY`)**

0xC071 = Watch Dog counter is restarted.
Other value = No effect.

WD Status Register

Name: WD_SR

Access: Read-only

Reset Value:0x0

Offset: 0x0C

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
–	–	–	–	–	–	–	WDOVF

- **WDOVF: Watchdog Overflow (Code Label `WD_WDOVF`)**

  0 = No watchdog overflow.
  1 = A watchdog overflow has occurred since the last restart of the watchdog counter or since internal or external reset.

WD Enabling Sequence

To enable the Watchdog Timer, the sequence is as follows:

1. Disable the Watchdog by clearing the bit WDEN:
   Write 0x2340 to WD_OMR
   This step is unnecessary if the WD is already disabled (reset state).
2. Initialize the WD Clock Mode Register:
3. Write 0x373C to WD_CMR
   (HPCV = 15 and WDCLKS = MCK/8)
4. Restart the timer:
   Write 0xC071 to WD_CR
5. Enable the watchdog:
   Write 0x2345 to WD_OMR (interrupt enabled)

AIC: Advanced Interrupt Controller

The AT91M55800A has an 8-level priority, individually maskable, vectored interrupt controller. This feature substantially reduces the software and real-time overhead in handling internal and external interrupts.

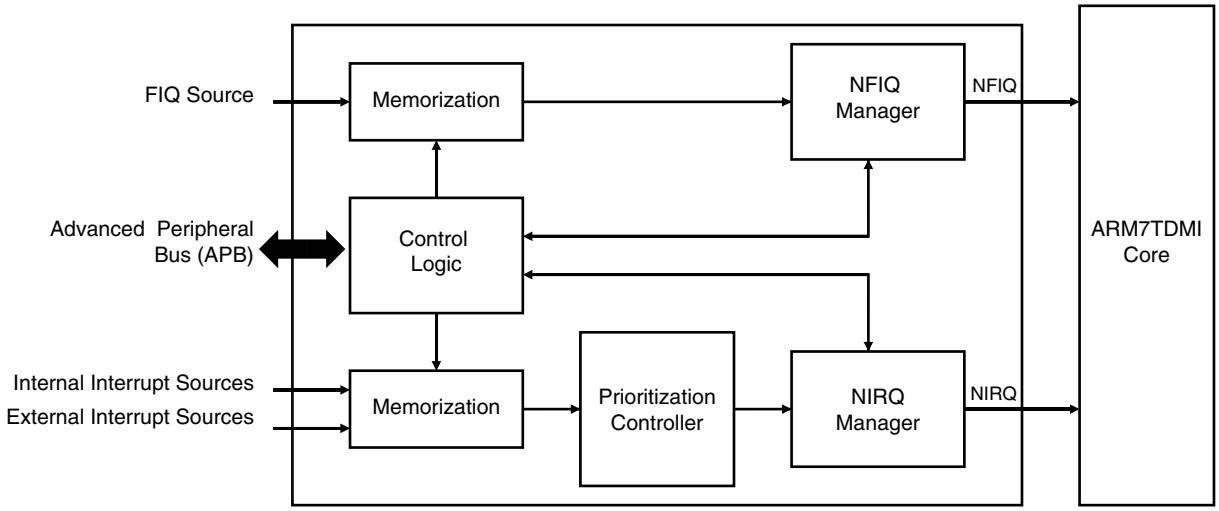
The interrupt controller is connected to the NFIQ (fast interrupt request) and the NIRQ (standard interrupt request) inputs of the ARM7TDMI processor. The processor's NFIQ line can only be asserted by the external fast interrupt request input: FIQ. The NIRQ line can be asserted by the interrupts generated by the on-chip peripherals and the external interrupt request lines: IRQ0 to IRQ5.

An 8-level priority encoder allows the customer to define the priority between the different NIRQ 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.

The interrupt sources are listed in Table 10 and the AIC programmable registers in Table 11.

Figure 39. Advanced Interrupt Controller Block Diagram



Note: After a hardware reset, the AIC pins are controlled by the PIO Controller. They must be configured to be controlled by the peripheral before being used.

Table 10. AIC Interrupt Sources

Interrupt Source	Interrupt Name	Interrupt Description
0	FIQ	Fast interrupt
1	SWIRQ	Software interrupt
2	US0IRQ	USART Channel 0 interrupt
3	US1IRQ	USART Channel 1 interrupt
4	US2IRQ	USART Channel 2 interrupt
5	SPIRQ	SPI interrupt
6	TC0IRQ	Timer Channel 0 interrupt
7	TC1IRQ	Timer Channel 1 interrupt
8	TC2IRQ	Timer Channel 2 interrupt
9	TC3IRQ	Timer Channel 3 interrupt
10	TC4IRQ	Timer Channel 4 interrupt
11	TC5IRQ	Timer Channel 5 interrupt
12	WDIRQ	Watchdog interrupt
13	PIOAIRQ	Parallel I/O Controller A interrupt
14	PIOBIRQ	Parallel I/O Controller B interrupt
15	AD0IRQ	Analog-to-digital Converter Channel 0 interrupt
16	AD1IRQ	Analog-to-digital Converter Channel 1 interrupt
17	DA0IRQ	Digital-to-analog Converter Channel 0 interrupt
18	DA1IRQ	Digital-to-analog Converter Channel 1 interrupt
19	RTCIRQ	Real-time Clock interrupt
20	APMCIRQ	Advanced Power Management Controller interrupt
21	–	Reserved
22	–	Reserved
23	SLCKIRQ	Slow Clock Interrupt
24	IRQ5	External interrupt 5
25	IRQ4	External interrupt 4
26	IRQ3	External interrupt 3
27	IRQ2	External interrupt 2
28	IRQ1	External interrupt 1
29	IRQ0	External interrupt 0
30	COMMRX	RX Debug Communication Channel interrupt
31	COMMTX	TX Debug Communication Channel interrupt

Hardware Interrupt Vectoring

The hardware interrupt vectoring reduces the number of instructions to reach the interrupt handler to only one. By storing the following instruction at address 0x00000018, the processor loads the program counter with the interrupt handler address stored in the AIC_IVR register. Execution is then vectored to the interrupt handler corresponding to the current interrupt.

```
ldr PC,[PC,# -&F20]
```

The current interrupt is the interrupt with the highest priority when the Interrupt Vector Register (AIC_IVR) is read. The value read in the AIC_IVR corresponds to the address stored in the Source Vector Register (AIC_SVR) of the current interrupt. Each interrupt source has its corresponding AIC_SVR. In order to take advantage of the hardware interrupt vectoring it is necessary to store the address of each interrupt handler in the corresponding AIC_SVR, at system initialization.

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 (see Table Table 10) 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 is 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.

- If the NIRQ line has been asserted but the AIC_IVR has not been read, then the processor reads the new higher priority interrupt handler address in the AIC_IVR register and the current interrupt level is updated.
- 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 last stored 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 de-asserts 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.

Fast Interrupt Request

The external FIQ line is the only source which can raise a fast interrupt request to the processor. Therefore, it has no priority controller.

The external FIQ line 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 loads 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.

Software Interrupt

Interrupt source 1 of the advanced interrupt controller is 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

When the AIC asserts the NIRQ line, the ARM7TDMI enters IRQ mode and the interrupt handler reads the IVR. It may happen that the AIC de-asserts the NIRQ line after the core has taken into account the NIRQ assertion and before the read of the IVR.

This behavior is called a Spurious Interrupt.

The AIC is able to detect these Spurious Interrupts and returns the Spurious Vector when the IVR is read. The Spurious Vector can be programmed by the user when the vector table is initialized.

A Spurious Interrupt may occur in the following cases:

- With any sources programmed to be level sensitive, if the interrupt signal of the AIC input is de-asserted at the same time as it is taken into account by the ARM7TDMI.
- If an interrupt is asserted at the same time as the software is disabling the corresponding source through AIC_IDCR (this can happen due to the pipelining of the ARM Core).

The same mechanism of Spurious Interrupt occurs if the ARM7TDMI reads the IVR (application software or ICE) when there is no interrupt pending. This mechanism is also valid for the FIQ interrupts.

Once the AIC enters the Spurious Interrupt management, it asserts neither the NIRQ nor the NFIQ lines to the ARM7TDMI as long as the Spurious Interrupt is not acknowledged. Therefore, it is mandatory for the Spurious Interrupt Service Routine to acknowledge the “Spurious” behavior by writing to the AIC_EOICR (End of Interrupt) before returning to the interrupted software. It also can perform other operation(s), e.g. trace possible undesirable behavior.

Protect Mode

The Protect Mode permits reading of the Interrupt Vector Register without performing the associated automatic operations. This is necessary when working with a debug system.

When a Debug Monitor or an ICE reads the AIC User Interface, the IVR could be read. This would have the following consequences in normal mode:

- If an enabled interrupt with a higher priority than the current one is pending, it would be stacked.
- If there is no enabled pending interrupt, the spurious vector would be returned.

In either case, an End of Interrupt Command would be necessary to acknowledge and to restore the context of the AIC. This operation is generally not performed by the debug system. Hence the debug system would become strongly intrusive, and could cause the application to enter an undesired state.

This is avoided by using Protect Mode.

The Protect Mode is enabled by setting the AIC bit in the SF Protect Mode Register.

When Protect Mode is enabled, the AIC performs interrupt stacking only when a write access is performed on the AIC_IVR. Therefore, the Interrupt Service Routines must write (arbitrary data) to the AIC_IVR just after reading it.

The new context of the AIC, including the value of the Interrupt Status Register (AIC_ISR), is updated with the current interrupt only when IVR is written.

An AIC_IVR read on its own (e.g. by a debugger), modifies neither the AIC context nor the AIC_ISR.

Extra AIC_IVR reads performed in between the read and the write can cause unpredictable results. Therefore, it is strongly recommended not to set a breakpoint between these 2 actions, nor to stop the software.

The debug system must not write to the AIC_IVR as this would cause undesirable effects.

The following table shows the main steps of an interrupt and the order in which they are performed according to the mode:

Action	Normal Mode	Protect Mode
Calculate active interrupt (higher than current or spurious)	Read AIC_IVR	Read AIC_IVR
Determine and return the vector of the active interrupt	Read AIC_IVR	Read AIC_IVR
Memorize interrupt	Read AIC_IVR	Read AIC_IVR
Push on internal stack the current priority level	Read AIC_IVR	Write AIC_IVR
Acknowledge the interrupt (1)	Read AIC_IVR	Write AIC_IVR
No effect(2)	Write AIC_IVR	–

Notes: 1. NIRQ de-assertion and automatic interrupt clearing if the source is programmed as level sensitive
2. Note that software which has been written and debugged using Protect Mode will run correctly in Normal Mode without modification. However in Normal Mode the AIC_IVR write has no effect and can be removed to optimize the code.

AIC User Interface

Base Address: 0xFFFFF000 (Code Label `AIC_BASE`)

Table 11. AIC Memory Map

Offset	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
–	–	–	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
–	–	–	Read/Write	0
0x0FC	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	(see Note 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 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

Reset Value: 0

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 (Code Label `AIC_PRIOR`)**

  Program 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 (Code Label `AIC_SRCTYPE`)**

  Program 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	Code Label Internal	External Sources	Code Label External
0	0	Level Sensitive	AIC_SRCTYPE_INT_LEVEL_SENSITIVE	Low-level Sensitive	AIC_SRCTYPE_EXT_LOW_LEVEL
0	1	Edge-triggered	AIC_SRCTYPE_INT_EDGE_TRIGGERED	Negative Edge-triggered	AIC_SRCTYPE_EXT_NEGATIVE_EDGE
1	0	Level Sensitive	AIC_SRCTYPE_INT_LEVEL_SENSITIVE	High-level Sensitive	AIC_SRCTYPE_EXT_HIGH_LEVEL
1	1	Edge-triggered	AIC_SRCTYPE_INT_EDGE_TRIGGERED	Positive Edge-triggered	AIC_SRCTYPE_EXT_POSITIVE_EDGE

AIC Source Vector Register

Register Name: AIC_SVR0..AIC_SVR31

Access Type: Read/Write

Reset Value:0

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: Interrupt Handler Address**

  The user may store in these registers the addresses of the corresponding handler for each interrupt source.

AIC Interrupt Vector Register

Register Name: AIC_IVR

Access Type: Read-only

Reset Value:0

Offset: 0x100

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: Interrupt Vector Register**

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

AIC FIQ Vector Register

Register Name: AIC_FVR

Access Type: Read-only

Reset Value:0

Offset: 0x104

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							

- **FIQV: FIQ Vector Register**

  The FIQ Vector Register contains the vector programmed by the user in the Source Vector Register 0 which corresponds to FIQ.

AIC Interrupt Status Register

Register Name: AIC_ISR

Access Type: Read-only

Reset Value:0

Offset: 0x108

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: Current IRQ Identifier (Code Label `AIC_IRQID`)**

  The Interrupt Status Register returns the current interrupt source number.

AIC Interrupt Pending Register

Register Name: AIC_IPR

Access Type: Read-only

Reset Value: Undefined

Offset: 0x10C

31	30	29	28	27	26	25	24
COMMRX	COMMTX	IRQ0	IRQ1	IRQ2	IRQ3	IRQ4	IRQ5
23	22	21	20	19	18	17	16
SLCKIRQ	–	–	APMCIRQ	RTCIRQ	DAC1IRQ	DAC0IRQ	ADC1IRQ
15	14	13	12	11	10	9	8
ADC0IRQ	PIOBIRQ	PIOAIRQ	WDIRQ	TC5IRQ	TC4IRQ	TC3IRQ	TC2IRQ
7	6	5	4	3	2	1	0
TC1IRQ	TC0IRQ	SPIRQ	US2IRQ	US1IRQ	US0IRQ	SWIRQ	FIQ

- **Interrupt Pending**

  0 = Corresponding interrupt is inactive.
  1 = Corresponding interrupt is pending.

AIC Interrupt Mask Register

Register Name: AIC_IMR

Access Type: Read-only

Reset Value:0

Offset: 0x110

31	30	29	28	27	26	25	24
COMMRX	COMMTX	IRQ0	IRQ1	IRQ2	IRQ3	IRQ4	IRQ5
23	22	21	20	19	18	17	16
SLCKIRQ	–	–	APMCIRQ	RTCIRQ	DAC1IRQ	DAC0IRQ	ADC1IRQ
15	14	13	12	11	10	9	8
ADC0IRQ	PIOBIRQ	PIOAIRQ	WDIRQ	TC5IRQ	TC4IRQ	TC3IRQ	TC2IRQ
7	6	5	4	3	2	1	0
TC1IRQ	TC0IRQ	SPIRQ	US2IRQ	US1IRQ	US0IRQ	SWIRQ	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

Reset Value:0

Offset: 0x114

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 (Code Label `AIC_NFIQ`)**

  0 = NFIQ line inactive.
  1 = NFIQ line active.
- **NIRQ: NIRQ Status (Code Label `AIC_NIRQ`)**

  0 = NIRQ line inactive.
  1 = NIRQ line active.

AIC Interrupt Enable Command Register

Register Name: AIC_IECR

Access Type: Write-only

Offset: 0x120

31	30	29	28	27	26	25	24
COMMRX	COMMTX	IRQ0	IRQ1	IRQ2	IRQ3	IRQ4	IRQ5
23	22	21	20	19	18	17	16
SLCKIRQ	–	–	APMCIRQ	RTCIRQ	DAC1IRQ	DAC0IRQ	ADC1IRQ
15	14	13	12	11	10	9	8
ADC0IRQ	PIOBIRQ	PIOAIRQ	WDIRQ	TC5IRQ	TC4IRQ	TC3IRQ	TC2IRQ
7	6	5	4	3	2	1	0
TC1IRQ	TC0IRQ	SPIRQ	US2IRQ	US1IRQ	US0IRQ	SWIRQ	FIQ

- **Interrupt Enable**

  0 = No effect.
  1 = Enables corresponding interrupt.

AIC Interrupt Disable Command Register

Register Name: AIC_IDCR

Access Type: Write-only

Offset: 0x124

31	30	29	28	27	26	25	24
COMMRX	COMMTX	IRQ0	IRQ1	IRQ2	IRQ3	IRQ4	IRQ5
23	22	21	20	19	18	17	16
SLCKIRQ	–	–	APMCIRQ	RTCIRQ	DAC1IRQ	DAC0IRQ	ADC1IRQ
15	14	13	12	11	10	9	8
ADC0IRQ	PIOBIRQ	PIOAIRQ	WDIRQ	TC5IRQ	TC4IRQ	TC3IRQ	TC2IRQ
7	6	5	4	3	2	1	0
TC1IRQ	TC0IRQ	SPIRQ	US2IRQ	US1IRQ	US0IRQ	SWIRQ	FIQ

- **Interrupt Disable**

  0 = No effect.
  1 = Disables corresponding interrupt.

AIC Interrupt Clear Command Register

Register Name: AIC_ICCR

Access Type: Write-only

Offset: 0x128

31	30	29	28	27	26	25	24
COMMRX	COMMTX	IRQ0	IRQ1	IRQ2	IRQ3	IRQ4	IRQ5
23	22	21	20	19	18	17	16
SLCKIRQ	–	–	APMCIRQ	RTCIRQ	DAC1IRQ	DAC0IRQ	ADC1IRQ
15	14	13	12	11	10	9	8
ADC0IRQ	PIOBIRQ	PIOAIRQ	WDIRQ	TC5IRQ	TC4IRQ	TC3IRQ	TC2IRQ
7	6	5	4	3	2	1	0
TC1IRQ	TC0IRQ	SPIRQ	US2IRQ	US1IRQ	US0IRQ	SWIRQ	FIQ

- **Interrupt Clear**

  0 = No effect.
  1 = Clears corresponding interrupt.

AIC Interrupt Set Command Register

Register Name: AIC_ISCR

Access Type: Write-only

Offset: 0x12C

31	30	29	28	27	26	25	24
COMMRX	COMMTX	IRQ0	IRQ1	IRQ2	IRQ3	IRQ4	IRQ5
23	22	21	20	19	18	17	16
SLCKIRQ	–	–	APMCIRQ	RTCIRQ	DAC1IRQ	DAC0IRQ	ADC1IRQ
15	14	13	12	11	10	9	8
ADC0IRQ	PIOBIRQ	PIOAIRQ	WDIRQ	TC5IRQ	TC4IRQ	TC3IRQ	TC2IRQ
7	6	5	4	3	2	1	0
TC1IRQ	TC0IRQ	SPIRQ	US2IRQ	US1IRQ	US0IRQ	SWIRQ	FIQ

- **Interrupt Set**

  0 = No effect.
  1 = Sets corresponding interrupt.

AIC End of Interrupt Command Register

Register Name: AIC_EOICR

Access Type: Write-only

Offset: 0x130

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 because it is only necessary to make a write to this register location to signal the end of interrupt treatment.

AIC Spurious Vector Register

Register Name:AIC_SPU

Access Type: Read/Write

Reset Value:0

Offset: 0x134

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

- **SPUVEC: Spurious Interrupt Vector Handler Address**
  The user may store the address of the Spurious Interrupt handler in this register.

Standard Interrupt Sequence

It is assumed that:

- The Advanced Interrupt Controller has been programmed, AIC_SVR are loaded with corresponding interrupt service routine addresses and interrupts are enabled.
- The Instruction at address 0x18(IRQ exception vector address) is

```
ldr pc, [pc, #-&F20]
```

When NIRQ is asserted, if the bit I of CPSR is 0, the sequence is:

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 loaded at address 0x18 is executed, the Program Counter is loaded with the value read in AIC_IVR. Reading the AIC_IVR has the following effects:

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

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

   Automatically clear the interrupt, if it has been programmed to be edge-triggered

   Push the current level on to the stack

   Return the value written in the AIC_SVR corresponding to the current interrupt
4. The previous step has effect to branch to the corresponding interrupt service routine. This should start 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 can then proceed 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 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 on the stack. If another interrupt is pending, with lower or equal priority than old current level but with higher priority than the new current level, the NIRQ line is reasserted, but the interrupt sequence does not immediately start because the I bit is set in the core.

9. The SPSR (SPSR_irq) is restored. Finally, the saved value of the Link Register is restored directly into the PC. This has effect of returning from the interrupt to whatever was being executed before, and of loading the CPSR with the stored SPSR, 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).

PIO: Parallel I/O Controller

The AT91M55800A has 58 programmable I/O lines. 13 pins are dedicated as general-purpose I/O pins. The other I/O lines are multiplexed with an external signal of a peripheral to optimize the use of available package pins. The PIO lines are controlled by two separate and identical PIO Controllers called PIOA and PIOB. The PIO controller enables the generation of an interrupt on input change and insertion of a simple input glitch filter on any of the PIO pins.

Multiplexed I/O Lines

Some I/O lines are multiplexed with an I/O signal of a peripheral. After reset, the pin is controlled by the PIO Controller and is in input mode.

When a peripheral signal is not used in an application, the corresponding pin can be used as a parallel I/O. Each parallel I/O line is bi-directional, whether the peripheral defines the signal as input or output. Figure 40 shows the multiplexing of the peripheral signals with Parallel I/O signals.

If a pin is multiplexed between the PIO Controller and a peripheral, the pin is controlled by the registers PIO_PER (PIO Enable) and PIO_PDR (PIO Disable). The register PIO_PSR (PIO Status) indicates whether the pin is controlled by the corresponding peripheral or by the PIO Controller.

If a pin is a general multi-purpose parallel I/O pin (not multiplexed with a peripheral), PIO_PER and PIO_PDR have no effect and PIO_PSR returns 1 for the bits corresponding to these pins.

When the PIO is selected, the peripheral input line is connected to zero.

Output Selection

The user can enable each individual I/O signal as an output with the registers PIO_OER (Output Enable) and PIO_ODR (Output Disable). The output status of the I/O signals can be read in the register PIO_OSR (Output Status). The direction defined has 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 (see Output Selection above), the level is programmed using the registers PIO_SODR (Set Output Data) and PIO_CODR (Clear Output Data). In this case, the programmed value can be read in PIO_ODSR (Output Data Status).

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 (see peripheral datasheets).

In all cases, the level on the pin can be read in the register PIO_PDSR (Pin Data Status).

Filters

Optional input glitch filtering is available on each pin and is controlled by the registers PIO_IFER (Input Filter Enable) and PIO_IFDR (Input Filter Disable). The input glitch filtering can be selected whether the pin is used for its peripheral function or as a parallel I/O line. The register PIO_IFSR (Input Filter Status) indicates whether or not the filter is activated for each pin.

Interrupts

Each parallel I/O can be programmed to generate an interrupt when a level change occurs. This is controlled by the PIO_IER (Interrupt Enable) and PIO_IDR (Interrupt Disable) registers which enable/disable the I/O interrupt by setting/clearing the corresponding bit in the PIO_IMR. When a change in level occurs, the corresponding bit in the PIO_ISR (Interrupt Status) is set 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 (Interrupt Mask) is enabled, the PIO interrupt is asserted.

When PIO_ISR is read, the register is automatically cleared.

User Interface

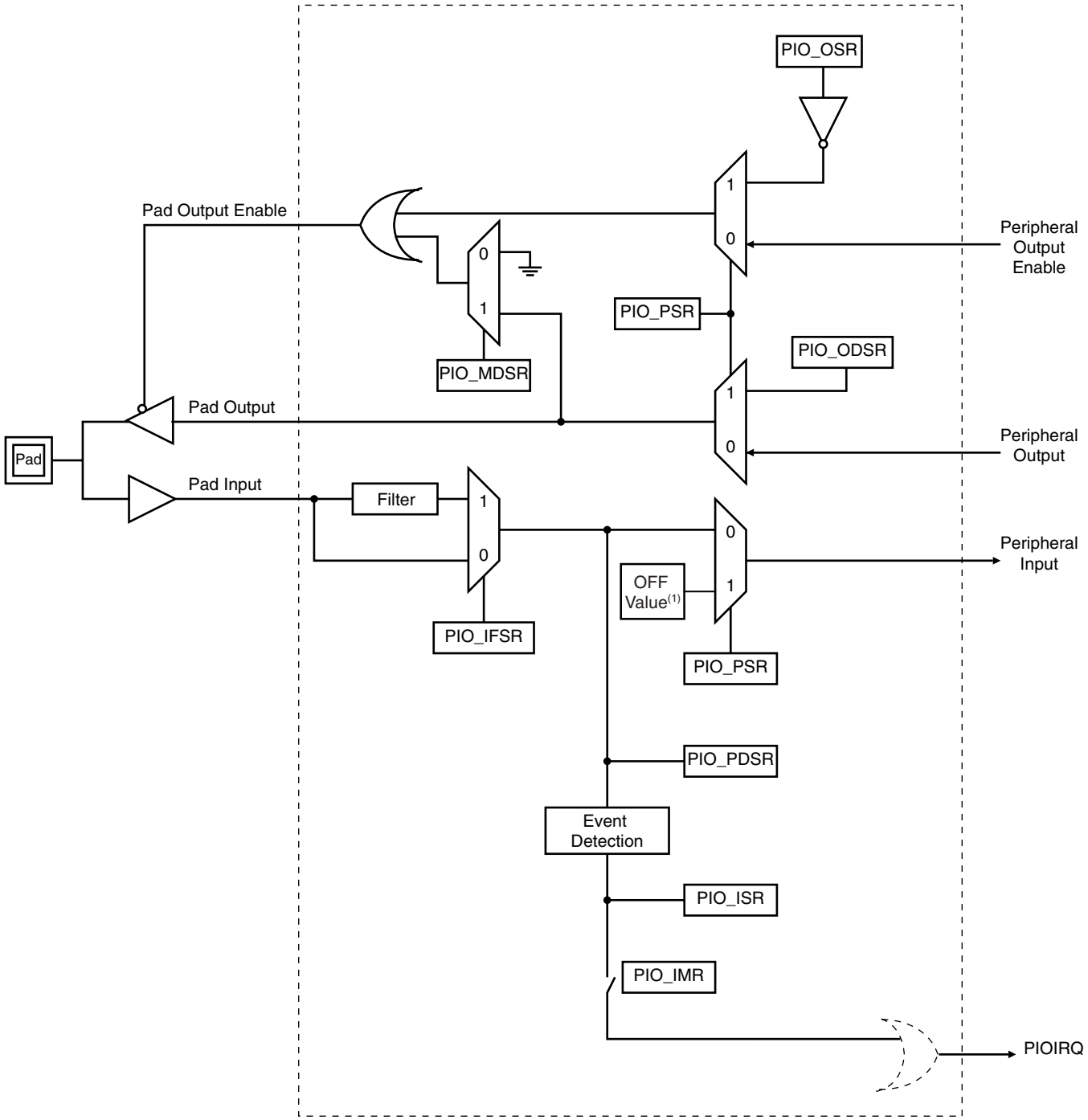
Each individual I/O is associated with a bit position in the Parallel I/O user interface registers. Each of these registers are 32 bits wide. If a parallel I/O line is not defined, writing to the corresponding bits has no effect. Undefined bits read zero.

Multi-driver (Open Drain)

Each I/O can be programmed for multi-driver option. This means that the I/O is configured as open drain (can only drive a low level) in order to support external drivers on the same pin. An external pull-up is necessary to guarantee a logic level of one when the pin is not being driven.

Registers PIO_MDER (Multi-driver Enable) and PIO_MDDR (Multi-driver Disable) control this option. Multi-driver can be selected whether the I/O pin is controlled by the PIO Controller or the peripheral. PIO_MDSR (Multi-driver Status) indicates which pins are configured to support external drivers.

Figure 40. Parallel I/O Multiplexed with a Bi-directional Signal



Note: 1. See “PIO Connection Tables.”

PIO Connection Tables

Table 12. 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.

Table 13. PIO Controller B Connection Table

PIO Controller		Peripheral					
Bit Number	Port Name	Port Name	Signal Description	Signal Direction	OFF Value[1]	Reset State	Pin Number
0	PB0	–	–	–	–	PIO Input	139
1	PB1	–	–	–	–	PIO Input	140
2	PB2	–	–	–	–	PIO Input	141
3	PB3	IRQ4	External Interrupt 4	Input	0	PIO Input	142
4	PB4	IRQ5	External Interrupt 5	Input	0	PIO Input	143
5	PB5	–	–	–	0	PIO Input	144
6	PB6	AD0TRIG	ADC0 External Trigger	Input	0	PIO Input	145
7	PB7	AD1TRIG	ADC1 External Trigger	Input	0	PIO Input	146
8	PB8	–	–	–	–	PIO Input	149
9	PB9	–	–	–	–	PIO Input	150
10	PB10	–	–	–	–	PIO Input	151
11	PB11	–	–	–	–	PIO Input	152
12	PB12	–	–	–	–	PIO Input	153
13	PB13	–	–	–	–	PIO Input	154
14	PB14	–	–	–	–	PIO Input	155
15	PB15	–	–	–	–	PIO Input	156
16	PB16	–	–	–	–	PIO Input	157
17	PB17	–	–	–	–	PIO Input	158
18	PB18	BMS	Boot Mode Select	Input	0	PIO Input	163
19	PB19	TCLK0	Timer 0 Clock signal	Input	0	PIO Input	55
20	PB20	TIOA0	Timer 0 Signal A	Bi-directional	0	PIO Input	56
21	PB21	TIOB0	Timer 0 Signal B	Bi-directional	0	PIO Input	57
22	PB22	TCLK1	Timer 1 Clock signal	Input	0	PIO Input	58
23	PB23	TIOA1	Timer 1 Signal A	Bi-directional	0	PIO Input	61
24	PB24	TIOB1	Timer 1 Signal B	Bi-directional	0	PIO Input	62
25	PB25	TCLK2	Timer 2 Clock signal	Input	0	PIO Input	63
26	PB26	TIOA2	Timer 2 Signal A	Bi-directional	0	PIO Input	64
27	PB27	TIOB2	Timer 2 Signal B	Bi-directional	0	PIO Input	65
28	–	–	–	–	–	–	–
29	–	–	–	–	–	–	–
30	–	–	–	–	–	–	–
31	–	–	–	–	–	–	–

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

PIO User Interface

PIO Controller A Base Address:0xFFFEC000 (Code Label `PIOA_BASE`)

PIO Controller B Base Address:0xFFFF0000 (Code Label `PIOB_BASE`)

Table 14. PIO Controller Memory Map

Offset	Register	Name	Access	Reset State
0x00	PIO Enable Register	PIO_PER	Write-only	–
0x04	PIO Disable Register	PIO_PDR	Write-only	–
0x08	PIO Status Register	PIO_PSR	Read-only	0x3FFF FFFF (A) 0x0FFF FFFF (B)
0x0C	Reserved	–	–	–
0x10	Output Enable Register	PIO_OER	Write-only	–
0x14	Output Disable Register	PIO_ODR	Write-only	–
0x18	Output Status Register	PIO_OSR	Read-only	0
0x1C	Reserved	–	–	–
0x20	Input Filter Enable Register	PIO_IFER	Write-only	–
0x24	Input Filter Disable Register	PIO_IFDR	Write-only	–
0x28	Input Filter Status Register	PIO_IFSR	Read-only	0
0x2C	Reserved	–	–	–
0x30	Set Output Data Register	PIO_SODR	Write-only	–
0x34	Clear Output Data Register	PIO_CODR	Write-only	–
0x38	Output Data Status Register	PIO_ODSR	Read-only	0
0x3C	Pin Data Status Register	PIO_PDSR	Read-only	(see Note 1)
0x40	Interrupt Enable Register	PIO_IER	Write-only	–
0x44	Interrupt Disable Register	PIO_IDR	Write-only	–
0x48	Interrupt Mask Register	PIO_IMR	Read-only	0
0x4C	Interrupt Status Register	PIO_ISR	Read-only	(see Note 2)
0x50	Multi-driver Enable Register	PIO_MDER	Write-only	–
0x54	Multi-driver Disable Register	PIO_MDDR	Write-only	–
0x58	Multi-driver Status Register	PIO_MDSR	Read-only	0
0x5C	Reserved	–	–	–

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

Offset: 0x00

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

Offset: 0x04

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-onlyRead-only

Offset: 0x08

Reset Value:0x3FFFFFFF (A)

0x0FFFFFFF (B)

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

Offset: 0x10

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, this has 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

Offset: 0x14

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, this has 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

Offset: 0x18

Reset Value:0

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 Input Filter Enable Register

Register Name:PIO_IFER

Access Type: Write-only

Offset: 0x20

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 input glitch filters. It affects the pin whether or not the PIO is enabled. The register is programmed as follows:

1 = Enables the glitch filter on the corresponding pin.
0 = No effect.

PIO Input Filter Disable Register

Register Name:IO_IFDR

Access Type: Write-only

Offset: 0x24

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 input glitch filters. It affects the pin whether or not the PIO is enabled. The register is programmed as follows:

1 = Disables the glitch filter on the corresponding pin.
0 = No effect.

PIO Input Filter Status Register

Register Name:PIO_IFSR

Access Type: Read-only

Offset: 0x28

Reset Value:0

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 have glitch filters selected. It is updated when PIO outputs are enabled or disabled by writing to PIO_IFER or PIO_IFDR.

1 = Filter is selected on the corresponding input (peripheral and PIO).
0 = Filter is not selected on the corresponding input.

Note: When the glitch filter is selected, and the PIO Controller clock is disabled, either the signal on the peripheral input or the corresponding bit in PIO_PDSR remains at the current state.

PIO Set Output Data Register

Register Name:PIO_SODR

Access Type: Write-only

Offset: 0x30

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

Offset: 0x34

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

Offset: 0x38

Reset Value:0

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

Offset: 0x3C

Reset Value: Undefined

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

Offset: 0x40

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 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

Offset: 0x44

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 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

Offset: 0x48

Reset Value:0

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 input pin.
0 = Interrupt is not enabled on the corresponding input pin.

PIO Interrupt Status Register

Register Name:PIO_ISR

Access Type: Read-only

Offset: 0x4C

Reset Value:0

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.

PIO Multi-driver Enable Register

Register Name:PIO_MDER

Access Type: Write-only

Offset: 0x50

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 to be configured as open drain to support external drivers on the same pin.

1 = Enables multi-drive option on the corresponding pin.
0 = No effect.

PIO Multi-driver Disable Register

Register Name: PIO_MDDR

Access Type: Write-only

Offset: 0x54

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 the open drain configuration of the output buffer.
1 = Disables the multi-driver option on the corresponding pin.
0 = No effect.

PIO Multi-driver Status Register

Register Name:PIO_MDSR

Access Type: Read-only

Reset Value:0x0

Offset: 0x58

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 configured with open drain drivers.
1 = PIO is configured as an open drain.
0 = PIO is not configured as an open drain.

SF: Special Function Registers

The AT91M55800A provides registers which implement the following special functions.

- Chip identification
- RESET status

Chip Identifier

The following chip identifier values are covered in this datasheet:

Product	Revision	Chip ID
AT91M55800A	A	0x15580040

SF User Interface

Chip ID Base Address = 0xFFF00000 (Code Label `SF_BASE`)

Table 15. SF Memory Map

Offset	Register	Name	Access	Reset State
0x00	Chip ID Register	SF_CIDR	Read-only	Hardwired
0x04	Chip ID Extension Register	SF_EXID	Read-only	Hardwired
0x08	Reset Status Register	SF_RSR	Read-only	See register description
0x0C	Reserved	–	–	–
0x10	Reserved	–	–	–
0x14	Reserved	–	–	–
0x18	Protect Mode Register	SF_PMR	Read/Write	0x0

Chip ID Register

Register Name:SF_CIDR

Access Type: Read-only

Offset: 0x00

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

- **VERSION: Version of the chip (Code Label `SF_VERSION`)**

  This value is incremented by one with each new version of the chip (from zero to a maximum value of 31).
- **NVPSIZ: Nonvolatile Program Memory Size**

NVPSIZ				Size	Code Label: `SF_NVPSIZ`
0	0	0	0	None	SF_NVPSIZ_NONE
0	0	1	1	32K Bytes	SF_NVPSIZ_32K
0	1	0	1	64K Bytes	SF_NVP_SIZ_64K
0	1	1	1	128K Bytes	SF_NVP_SIZ_128K
1	0	0	1	256K Bytes	SF_NVP_SIZ_256K
Others				Reserved	–

- **NVDSIZ: Nonvolatile Data Memory Size**

NVDSIZ				Size	Code Label: `SF_NVDSIZ`
0	0	0	0	None	SF_NVDSIZ_NONE
Others				Reserved	–

- **VDSIZ: Volatile Data Memory Size**

VDSIZ				Size	Code Label: `SF_VDSIZ`
0	0	0	0	None	SF_VDSIZ_NONE
0	0	0	1	1K Bytes	SF_VDSIZ_1K
0	0	1	0	2K Bytes	SF_VDSIZ_2K
0	1	0	0	4K Bytes	SF_VDSIZ_4K
1	0	0	0	8K Bytes	SF_VDSIZ_8K
Others				Reserved	–

- **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).

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.

Reset Status Register

Register Name:SF_RSR

Access Type: Read-only

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
RESET							

- **RESET: Reset Status Information**

This field indicates whether the reset was demanded by the external system (via NRST) or by the Watchdog internal reset request.

Reset	Cause of Reset	Code Label
0x6C	External Pin	SF_EXT_RESET
0x53	Internal Watchdog	SF_WD_RESET

SF Protect Mode Register

Register Name:SF_PMR

Access Type: Read/Write

Reset Value:0x0

Offset: 0x18

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

- **PMRKEY: Protect Mode Register Key**

Used only when writing SF_PMR. PMRKEY is reads 0.

0x27A8: Write access in SF_PMR is allowed.

Other value: Write access in SF_PMR is prohibited.

- **AIC: AIC Protect Mode Enable (Code Label `SF_AIC`)**

0 = The Advanced Interrupt Controller runs in Normal Mode.

1 = The Advanced Interrupt Controller runs in Protect Mode.

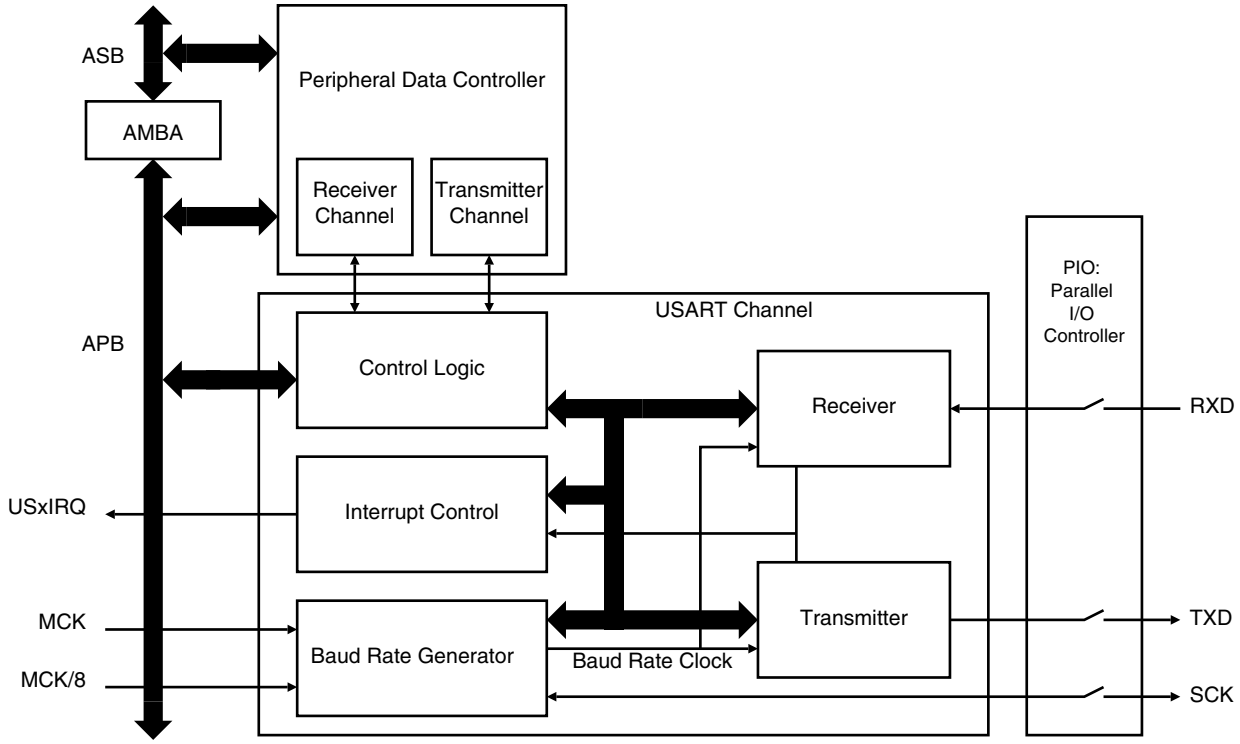
USART: Universal Synchronous/ Asynchronous Receiver/Transmitter

The AT91M55800AA provides three identical, full-duplex, universal synchronous/asynchronous receiver/transmitters which are connected to the Peripheral Data Controller.

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
- Multi-drop Mode: Address Detection and Generation
- Interrupt Generation
- Two Dedicated Peripheral Data Controller channels
- 5-, 6-, 7-, 8- and 9-bit character length

Figure 41. USART Block Diagram



Pin Description

Table 16. USART Channel External Signals

Name	Description
SCK	USART Serial clock can be configured as input or output: SCK is configured as input if an External clock is selected (USCLKS[1] = 1) SCK is driven as output if the External Clock is disabled (USCLKS[1] = 0) and Clock output is enabled (CLKO = 1)
TXD	Transmit Serial Data is an output
RXD	Receive Serial Data is an input

Notes: 1. After a hardware reset, the USART clock is disabled by default. The user must configure the Power Management Controller before any access to the User Interface of the USART.

2. After a hardware reset, the USART pins are deselected by default (see “PIO: Parallel I/O Controller” on page 105). The user must configure the PIO Controller before enabling the transmitter or receiver. If the user selects one of the internal clocks, SCK can be configured as a PIO.

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 master clock MCK or the master clock divided by 8 (MCK/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 (MCK) period. The external clock frequency must be at least 2.5 times lower than the system clock.

When the USART is programmed to operate in Asynchronous Mode (SYNC = 0 in the Mode Register US_MR), 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.

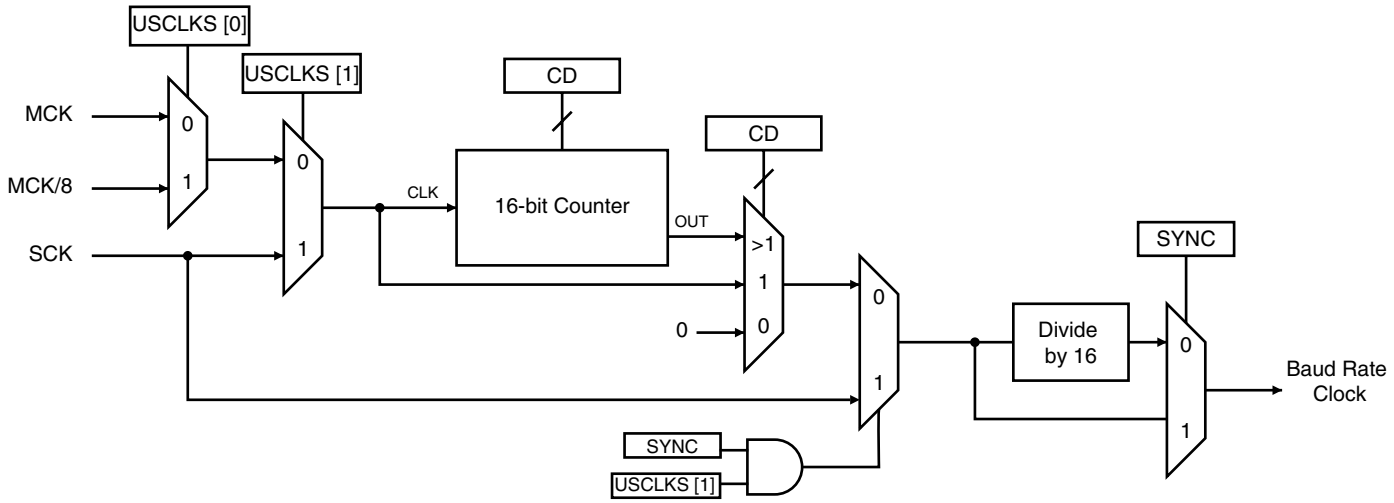
$$\text{Baud Rate} = \frac{\text{Selected Clock}}{16 \times \text{CD}}$$

When the USART is programmed to operate in Synchronous Mode (SYNC = 1) and the selected clock is internal (USCLKS[1] = 0 in the Mode Register US_MR), the Baud Rate Clock is the internal selected clock divided by the value written in US_BRGR. If US_BRGR is set to 0, the Baud Rate Clock is disabled.

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

In Synchronous Mode with external clock selected (USCLKS[1] = 1), the clock is provided directly by the signal on the SCK pin. No division is active. The value written in US_BRGR has no effect.

Figure 42. Baud Rate Generator



Receiver

Asynchronous Receiver

The USART is configured for asynchronous operation when SYNC = 0 (bit 7 of US_MR). In asynchronous mode, the USART 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 7 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 (one bit period) so the sampling point is 8 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 43. Asynchronous Mode: Start Bit Detection

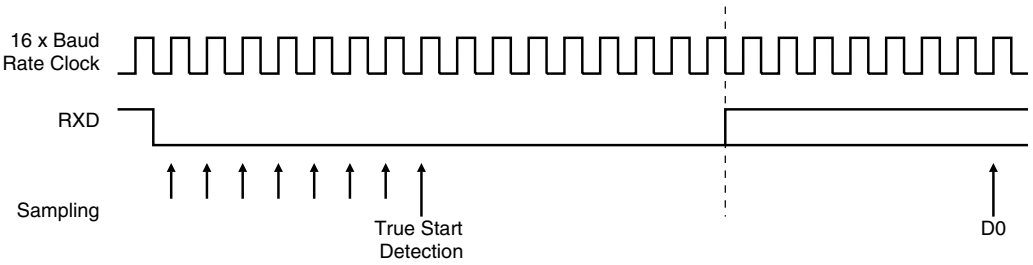
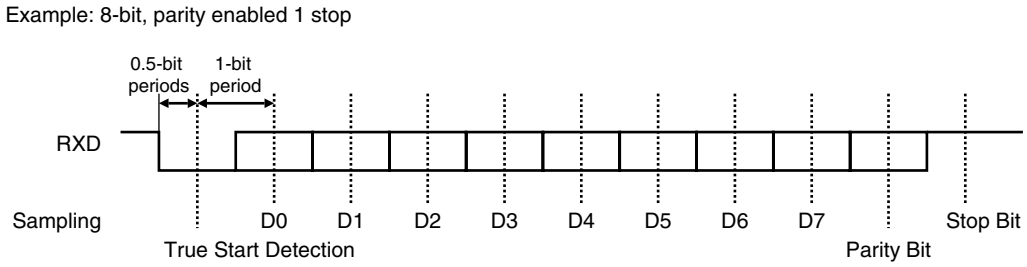


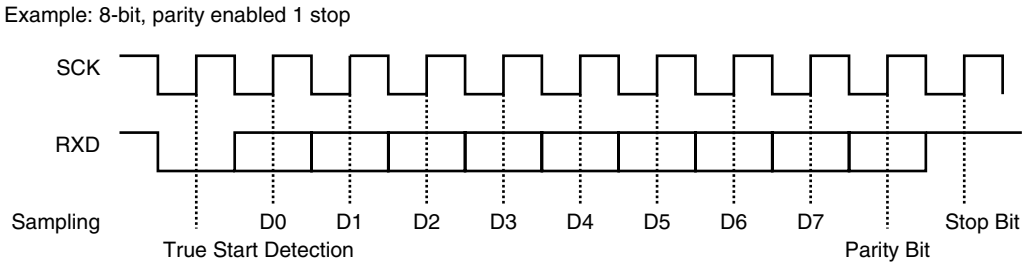
Figure 44. Asynchronous Mode: Character Reception



Synchronous Receiver

When configured for synchronous operation (SYNC = 1), the receiver samples the RXD signal on each rising edge of the Baud Rate clock. If a low level is detected, it is considered as a start. Data bits, parity bit and stop bit are sampled and the receiver waits for the next start bit. See example in Figure 45.

Figure 45. Synchronous Mode: 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. When the character is completed and as soon as the character is read, the parity status bit is cleared.

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 USART should wait for a new character to arrive while the RXD line is inactive (high level) is programmed in US_RTOR (Receiver Time-out). 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:

$$Duration = Value \bullet 4 \bullet BitPeriod$$

Transmitter

The transmitter has the same behavior in both synchronous and asynchronous operating modes. Start bit, data bits, parity bit and stop bits are serially shifted, lowest significant bit first, on the falling edge of the serial clock. See example in Figure 46.

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 (Transmit Holding), 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 Transmit Shift Register and US_THR are both empty, the TXEMPTY bit in US_CSR is set.

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 (Transmitter Time-guard). 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.

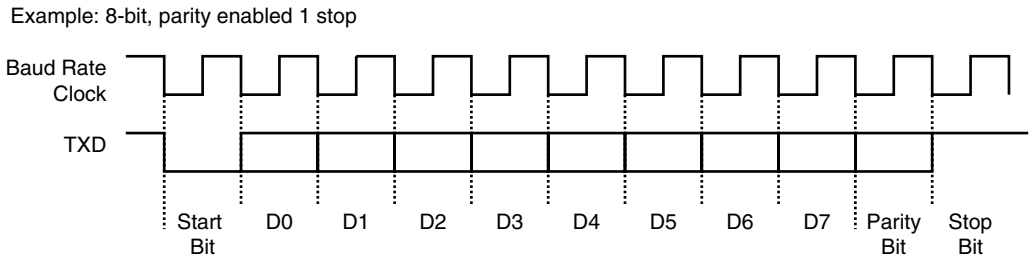
$$\text{Idle state duration between two characters} = \text{Time-guard value} \bullet \text{Bit period}$$

Multi-drop Mode

When the field PAR in US_MR equals 11X (binary value), the USART is configured to run in multi-drop mode. In this case, the parity error bit PARE in US_CSR is set when data is detected with a parity bit set to identify an address byte. PARE is cleared with the Reset Status Bits Command (RSTSTA) in US_CR. If the parity bit is detected low, identifying a data byte, PARE is not set.

The transmitter sends an address byte (parity bit set) when a Send Address Command (SENDA) is written to US_CR. In this case, the next byte written to US_THR will be transmitted as an address. After this any byte transmitted will have the parity bit cleared.

Figure 46. Synchronous and Asynchronous Modes: Character Transmission



Break

A break condition is a low signal level which has a duration of at least one character (including start/stop bits and parity).

Transmit Break

The transmitter generates a break condition on the TXD line when STTBRK is set in US_CR (Control Register). In this case, the character present in the Transmit Shift Register is completed before the line is held low.

To cancel a break condition on the TXD line, the STPBRK command in US_CR must be set. The USART completes a minimum break duration of one character length. The TXD line then returns to high level (idle state) for at least 12-bit periods to ensure that the end of break is correctly detected. Then the transmitter resumes normal operation.

The BREAK is managed like a character:

- The STTBRK and the STPBRK commands are performed only if the transmitter is ready (bit TXRDY = 1 in US_CSR)
- The STTBRK command blocks the transmitter holding register (bit TXRDY is cleared in US_CSR) until the break has started
- A break is started when the Shift Register is empty (any previous character is fully transmitted). US_CSR.TXEMPTY is cleared. The break blocks the transmitter shift register until it is completed (high level for at least 12-bit periods after the STPBRK command is requested)

In order to avoid unpredictable states:

- STTBRK and STPBRK commands must not be requested at the same time
- Once an STTBRK command is requested, further STTBRK commands are ignored until the BREAK is ended (high level for at least 12-bit periods)
- All STPBRK commands requested without a previous STTBRK command are ignored
- A byte written into the Transmit Holding Register while a break is pending but not started (bit TXRDY = 0 in US_CSR) is ignored
- It is *not permitted* to write new data in the Transmit Holding Register while a break is in progress (STPBRK has not been requested), even though TXRDY = 1 in US_CSR.
- A new STTBRK command *must not* be issued until an existing break has ended (TXEMPTY=1 in US_CSR).

The standard break transmission sequence is:

1. Wait for the transmitter ready
   (US_CSR.TXRDY = 1)
2. Send the STTBRK command
   (write 0x0200 to US_CR)
3. Wait for the transmitter ready
   (bit TXRDY = 1 in US_CSR)
4. Send the STPBRK command
   (write 0x0400 to US_CR)

The next byte can then be sent:

5. Wait for the transmitter ready
   (bit TXRDY = 1 in US_CSR)
6. Send the next byte
   (write byte to US_THR)

Each of these steps can be scheduled by using the interrupt if the bit TXRDY in US_IMR is set.

For character transmission, the USART channel must be enabled before sending a break.

Receive Break

The receiver detects a break condition when all data, parity and stop bits are low. When the low stop bit is detected, the receiver asserts the RXBRK bit in US_CSR. An end of receive break is detected by a high level for at least 1-bit + 1/16 of a bit period in asynchronous operating mode or at least one sample in synchronous operating mode. RXBRK is also asserted when an end of break is detected.

Both the beginning and the end of a break can be detected by interrupt if the bit US_IMR.RXBRK is set.

Peripheral Data Controller

Each USART channel is closely connected to a corresponding Peripheral Data Controller channel. One is dedicated to the receiver. The other is dedicated to the transmitter.

Note: The PDC is disabled if 9-bit character length is selected (MODE9 = 1) in US_MR.

The PDC channel is programmed using US_TPR (Transmit Pointer) and US_TCR (Transmit Counter) for the transmitter and US_RPR (Receive Pointer) and US_RCR (Receive Counter) 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.

Interrupt Generation

Each status bit in US_CSR has a corresponding bit in US_IER (Interrupt Enable) and US_IDR (Interrupt Disable) which controls the generation of interrupts by asserting the USART interrupt line connected to the Advanced Interrupt Controller. US_IMR (Interrupt Mask Register) indicates the status of the corresponding bits.

When a bit is set in US_CSR and the same bit is set in US_IMR, the interrupt line is asserted.

Channel Modes

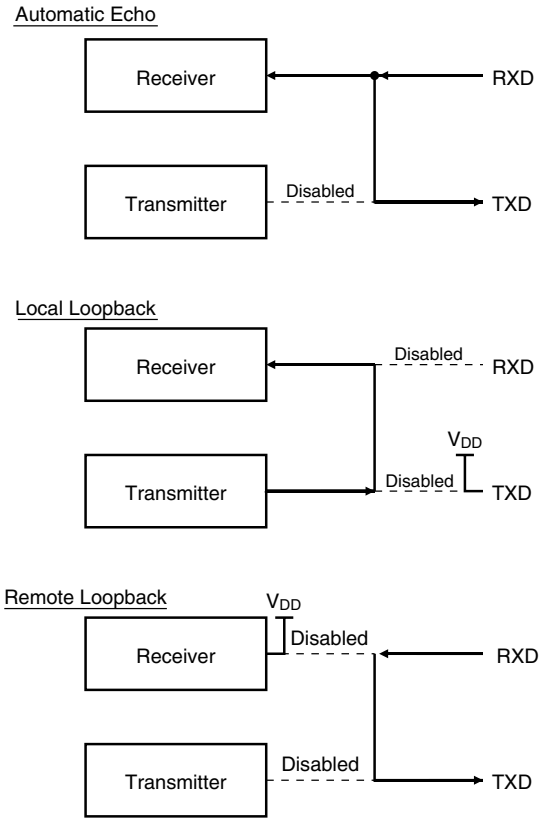
The USART can be programmed to operate in three different test modes, using the field CHMODE in US_MR.

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 retransmission.

Figure 47. Channel Modes



USART User Interface

Base Address USART0: 0xFFFC0000 (Code Label `USART0_BASE`)
Base Address USART1: 0xFFFC4000 (Code Label `USART1_BASE`)
Base Address USART2: 0xFFFC8000 (Code Label `USART2_BASE`)

Table 17. USART Memory Map

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

USART Control Register

Name: US_CR
Access Type: Write-only
Offset: 0x00

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

- **RSTRX: Reset Receiver (Code Label `US_RSTRX`)**
  0 = No effect.
  1 = The receiver logic is reset.
- **RSTTX: Reset Transmitter (Code Label `US_RSTTX`)**
  0 = No effect.
  1 = The transmitter logic is reset.
- **RXEN: Receiver Enable (Code Label `US_RXEN`)**
  0 = No effect.
  1 = The receiver is enabled if RXDIS is 0.
- **RXDIS: Receiver Disable (Code Label `US_RXDIS`)**
  0 = No effect.
  1 = The receiver is disabled.
- **TXEN: Transmitter Enable (Code Label `US_TXEN`)**
  0 = No effect.
  1 = The transmitter is enabled if TXDIS is 0.
- **TXDIS: Transmitter Disable (Code Label `US_TXDIS`)**
  0 = No effect.
  1 = The transmitter is disabled.
- **RSTSTA: Reset Status Bits (Code Label `US_RSTSTA`)**
  0 = No effect.
  1 = Resets the status bits PARE, FRAME, OVRE and RXBRK in the US_CSR.
- **STTBRK: Start Break (Code Label `US_STTBRK`)**
  0 = No effect.
  1 = If break is not being transmitted, start transmission of a break after the characters present in US_THR and the Transmit Shift Register have been transmitted.
- **STPBRK: Stop Break (Code Label `US_STPBRK`)**
  0 = No effect.
  1 = If a break is being transmitted, stop transmission of the break after a minimum of one character length and transmit a high level during 12 bit periods.
- **STTTO: Start Time-out (Code Label `US_STTTO`)**
  0 = No effect.
  1 = Start waiting for a character before clocking the time-out counter.
- **SENDA: Send Address (Code Label `US_SENDA`)**
  0 = No effect.
  1 = In Multi-drop Mode only, the next character written to the US_THR is sent with the address bit set.

USART Mode Register

Name: US_MR
Access Type: Read/Write
Reset State: 0
Offset: 0x04

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

23	22	21	20	19	18	17	16
–	–	–	–	–	CLKO	MODE9	–

15	14	13	12	11	10	9	8
CHMODE		NBSTOP		PAR			SYNC

7	6	5	4	3	2	1	0
CHRL		USCLKS		–	–	–	–

- **USCLKS: Clock Selection (Baud Rate Generator Input Clock)**

USCLKS		Selected Clock	Code Label: US_CLKS
0	0	MCK	US_CLKS_MCK
0	1	MCK/8	US_CLKS_MCK8
1	X	External (SCK)	US_CLKS_SCK

- **CHRL: Character Length**

CHRL		Character Length	Code Label: US_CHRL
0	0	Five bits	US_CHRL_5
0	1	Six bits	US_CHRL_6
1	0	Seven bits	US_CHRL_7
1	1	Eight bits	US_CHRL_8

Start, stop and parity bits are added to the character length.

- **SYNC: Synchronous Mode Select (Code Label US_SYNC)**
  0 = USART operates in Asynchronous Mode.
  1 = USART operates in Synchronous Mode.
- **PAR: Parity Type**

PAR			Parity Type	Code Label: US_PAR
0	0	0	Even Parity	US_PAR_EVEN
0	0	1	Odd Parity	US_PAR_ODD
0	1	0	Parity forced to 0 (Space)	US_PAR_SPACE
0	1	1	Parity forced to 1 (Mark)	US_PAR_MARK
1	0	x	No parity	US_PAR_NO
1	1	x	Multi-drop mode	US_PAR_MULTIDROP

- **NBSTOP: Number of Stop Bits**
  The interpretation of the number of stop bits depends on SYNC.

NBSTOP		Asynchronous (SYNC = 0)	Synchronous (SYNC = 1)	Code Label: US_NBSTOP
0	0	1 stop bit	1 stop bit	US_NBSTOP_1
0	1	1.5 stop bits	Reserved	US_NBSTOP_1_5
1	0	2 stop bits	2 stop bits	US_NBSTOP_2
1	1	Reserved	Reserved	–

- **CHMODE: Channel Mode**

CHMODE		Mode Description	Code Label: US_CHMODE
0	0	Normal Mode The USART Channel operates as an Rx/Tx USART.	US_CHMODE_NORMAL
0	1	Automatic Echo Receiver Data Input is connected to TXD pin.	US_CHMODE_AUTOMATIC_ECHO
1	0	Local Loopback Transmitter Output Signal is connected to Receiver Input Signal.	US_CHMODE_LOCAL_LOOPBACK
1	1	Remote Loopback RXD pin is internally connected to TXD pin.	US_CHMODE_REMODE_LOOPBACK

- **MODE9: 9-Bit Character Length (Code Label US_MODE9)**
  0 = CHRL defines character length.
  1 = 9-Bit character length.
- **CKLO: Clock Output Select (Code Label US_CLKO)**
  0 = The USART does not drive the SCK pin.
  1 = The USART drives the SCK pin if USCLKS[1] is 0.

USART Interrupt Enable Register

Name: US_IER
Access Type: Write-only
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
–	–	–	–	–	–	TXEMPTY	TIMEOUT
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	ENDTX	ENDRX	RXBRK	TXRDY	RXRDY

- **RXRDY: Enable RXRDY Interrupt (Code Label `US_RXRDY`)**
  0 = No effect.
  1 = Enables RXRDY Interrupt.
- **TXRDY: Enable TXRDY Interrupt (Code Label `US_TXRDY`)**
  0 = No effect.
  1 = Enables TXRDY Interrupt.
- **RXBRK: Enable Receiver Break Interrupt (Code Label `US_RXBRK`)**
  0 = No effect.
  1 = Enables Receiver Break Interrupt.
- **ENDRX: Enable End of Receive Transfer Interrupt (Code Label `US_ENDRX`)**
  0 = No effect.
  1 = Enables End of Receive Transfer Interrupt.
- **ENDTX: Enable End of Transmit Transfer Interrupt (Code Label `US_ENDTX`)**
  0 = No effect.
  1 = Enables End of Transmit Transfer Interrupt.
- **OVRE: Enable Overrun Error Interrupt (Code Label `US_OVRE`)**
  0 = No effect.
  1 = Enables Overrun Error Interrupt.
- **FRAME: Enable Framing Error Interrupt (Code Label `US_FRAME`)**
  0 = No effect.
  1 = Enables Framing Error Interrupt.
- **PARE: Enable Parity Error Interrupt (Code Label `US_PARE`)**
  0 = No effect.
  1 = Enables Parity Error Interrupt.
- **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.

USART Interrupt Disable Register

Name: US_IDR
Access Type: Write-only
Offset: 0x0C

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: Disable RXRDY Interrupt (Code Label `US_RXRDY`)**
  0 = No effect.
  1 = Disables RXRDY Interrupt.
- **TXRDY: Disable TXRDY Interrupt (Code Label `US_TXRDY`)**
  0 = No effect.
  1 = Disables TXRDY Interrupt.
- **RXBRK: Disable Receiver Break Interrupt (Code Label `US_RXBRK`)**
  0 = No effect.
  1 = Disables Receiver Break Interrupt.
- **ENDRX: Disable End of Receive Transfer Interrupt (Code Label `US_ENDRX`)**
  0 = No effect.
  1 = Disables End of Receive Transfer Interrupt.
- **ENDTX: Disable End of Transmit Transfer Interrupt (Code Label `US_ENDTX`)**
  0 = No effect.
  1 = Disables End of Transmit Transfer Interrupt.
- **OVRE: Disable Overrun Error Interrupt (Code Label `US_OVRE`)**
  0 = No effect.
  1 = Disables Overrun Error Interrupt.
- **FRAME: Disable Framing Error Interrupt (Code Label `US_FRAME`)**
  0 = No effect.
  1 = Disables Framing Error Interrupt.
- **PARE: Disable Parity Error Interrupt (Code Label `US_PARE`)**
  0 = No effect.
  1 = Disables Parity Error Interrupt.
- **TIMEOUT: Disable Time-out Interrupt (Code Label `US_TIMEOUT`)**
  0 = No effect.
  1 = Disables Receiver Time-out Interrupt.
- **TXEMPTY: Disable TXEMPTY Interrupt (Code Label `US_TXEMPTY`)**
  0 = No effect.
  1 = Disables TXEMPTY Interrupt.

USART Interrupt Mask Register

Name: US_IMR
Access Type: Read-only
Reset Value: 0x0
Offset: 0x10

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: RXRDY Interrupt Mask (Code Label `US_RXRDY`)**
  0 = RXRDY Interrupt is Disabled.
  1 = RXRDY Interrupt is Enabled.
- **TXRDY: TXRDY Interrupt Mask (Code Label `US_TXRDY`)**
  0 = TXRDY Interrupt is Disabled.
  1 = TXRDY Interrupt is Enabled.
- **RXBRK: Receiver Break Interrupt Mask (Code Label `US_RXBRK`)**
  0 = Receiver Break Interrupt is Disabled.
  1 = Receiver Break Interrupt is Enabled.
- **ENDRX: End of Receive Transfer Interrupt Mask (Code Label `US_ENDRX`)**
  0 = End of Receive Transfer Interrupt is Disabled.
  1 = End of Receive Transfer Interrupt is Enabled.
- **ENDTX: End of Transmit Transfer Interrupt Mask (Code Label `US_ENDTX`)**
  0 = End of Transmit Transfer Interrupt is Disabled.
  1 = End of Transmit Transfer Interrupt is Enabled.
- **OVRE: Overrun Error Interrupt Mask (Code Label `US_OVRE`)**
  0 = Overrun Error Interrupt is Disabled.
  1 = Overrun Error Interrupt is Enabled.
- **FRAME: Framing Error Interrupt Mask (Code Label `US_FRAME`)**
  0 = Framing Error Interrupt is Disabled.
  1 = Framing Error Interrupt is Enabled.
- **PARE: Parity Error Interrupt Mask (Code Label `US_PARE`)**
  0 = Parity Error Interrupt is Disabled.
  1 = Parity Error Interrupt is Enabled.
- **TIMEOUT: Time-out Interrupt Mask (Code Label `US_TIMEOUT`)**
  0 = Receive Time-out Interrupt is Disabled.
  1 = Receive Time-out Interrupt is Enabled.
- **TXEMPTY: TXEMPTY Interrupt Mask (Code Label `US_TXEMPTY`)**
  0 = TXEMPTY Interrupt is Disabled.
  1 = TXEMPTY Interrupt is Enabled.

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.
- **PARE: Parity Error (Code Label `US_PARE`)**
  1 = At least one parity bit has been detected false (or a parity bit high in multi-drop mode) since the last “Reset Status Bits” 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.
- **TIMEOUT: Receiver Time-out (Code Label `US_TIMEOUT`)**
  0 = There has not been a time-out since the last “Start Time-out” command or the Time-out Register is 0.
  1 = There has been a time-out since the last “Start Time-out” command.
- **TXEMPTY: Transmitter Empty (Code Label `US_TXEMPTY`)**
  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 USART is disabled or at reset. Transmitter Enable command (in US_CR) sets this bit to one.

USART Receiver Holding Register

Name: US_RHR
Access Type: Read-only
Reset State: 0
Offset: 0x18

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	RXCHR
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 9 bits, the bits are right-aligned. All unused bits read zero.

USART Transmitter Holding Register

Name: US_THR
Access Type: Write-only
Offset: 0x1C

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	TXCHR
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 9 bits, the bits are right-aligned.

USART Baud Rate Generator Register

Name: US_BRGR
Access Type: Read/Write
Reset State: 0
Offset: 0x20

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	
0	Disables Clock
1	Clock Divisor bypass
2 to 65535	Baud Rate (Asynchronous Mode) = Selected clock/(16 x CD) Baud Rate (Synchronous Mode) = Selected clock/CD

Notes: 1. In Synchronous Mode, the value programmed must be even to ensure a 50:50 mark:space ratio.
2. Clock divisor bypass (CD = 1) must not be used when internal clock MCK is selected (USCLKS = 0).

USART Receiver Time-out Register

Name: US_RTOR
Access Type: Read/Write
Reset State: 0
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
TO							

- **TO: Time-out Value**
  When a value is written to this register, a Start Time-out Command is automatically performed.

TO	
0	Disables the RX Time-out function.
1 - 255	The Time-out counter is loaded with TO when the Start Time-out Command is given or when each new data character is received (after reception has started).

Time-out duration = TO x 4 x Bit period

USART Transmitter Time-guard Register

Name: US_TTGR
Access Type: Read/Write
Reset State: 0
Offset: 0x28

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
TG							

- **TG: Time-guard Value**

TG	
0	Disables the TX Time-guard function.
1 - 255	TXD is inactive high after the transmission of each character for the time-guard duration.

Time-guard duration = TG x Bit period

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.

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.

USART Transmit Pointer Register

Name: US_TPR
Access Type: Read/Write
Reset State: 0
Offset: 0x38

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.

USART Transmit Counter Register

Name: US_TCR
Access Type: Read/Write
Reset State: 0
Offset: 0x3C

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.

TC: Timer Counter

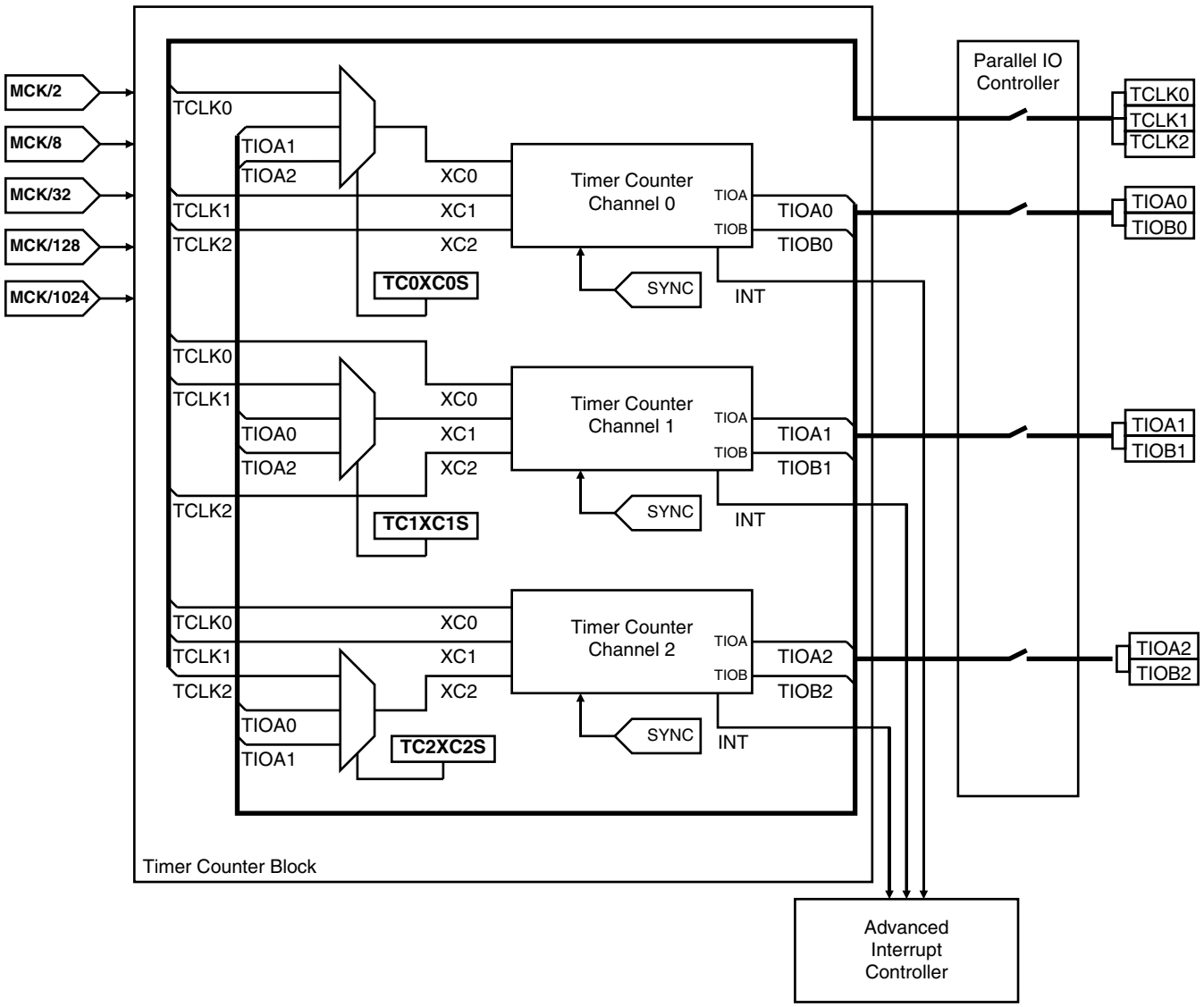
The AT91M55800A features two Timer Counter Blocks, each containing three identical 16-bit timer counter channels. Each channel can be independently programmed to perform a wide range of functions including frequency measurement, event counting, interval measurement, pulse generation, delay timing and pulse-width modulation.

Each Timer Counter channel has three external clock inputs, five internal clock inputs, and two multi-purpose input/output signals which can be configured by the user. Each channel drives an internal interrupt signal which can be programmed to generate processor interrupts via the AIC (Advanced Interrupt Controller).

Each 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.

The internal configuration of a single Timer Counter Block is shown in Figure 48.

Figure 48. TC Block Diagram



Signal Name Description

Channel Signals	Description
XC0, XC1, XC2	External Clock Inputs
TIOA	Capture Mode: General-purpose input Waveform Mode: General-purpose output
TIOB	Capture Mode: General-purpose input Waveform Mode: General-purpose input/output
INT	Interrupt signal output
SYNC	Synchronization input signal
Block 0 Signals	**Description**
TCLK0, TCLK1, TCLK2	External Clock Inputs for Channels 0, 1, 2
TIOA0	TIOA signal for Channel 0
TIOB0	TIOB signal for Channel 0
TIOA1	TIOA signal for Channel 1
TIOB1	TIOB signal for Channel 1
TIOA2	TIOA signal for Channel 2
TIOB2	TIOB signal for Channel 2
Block 1 Signals	**Description**
TCLK3, TCLK4, TCLK5	External Clock Inputs for Channels 3, 4, 5
TIOA3	TIOA signal for Channel 3
TIOB3	TIOB signal for Channel 3
TIOA4	TIOA signal for Channel 4
TIOB4	TIOB signal for Channel 4
TIOA5	TIOA signal for Channel 5
TIOB5	TIOB signal for Channel 5

Notes: 1. After a hardware reset, the TC clock is disabled by default (See “APMC: Advanced Power Management Controller” on page 50.). The user must configure the Power Management Controller before any access to the User Interface of the TC.
2. 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.

Timer Counter Description

Each Timer Counter channel is identical in operation. The registers for channel programming are listed below “Signal Name Description” on page 149.

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 input clock. When the counter reaches the value 0xFFFF and passes to 0x0000, an overflow occurs and the bit COVFS 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 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:

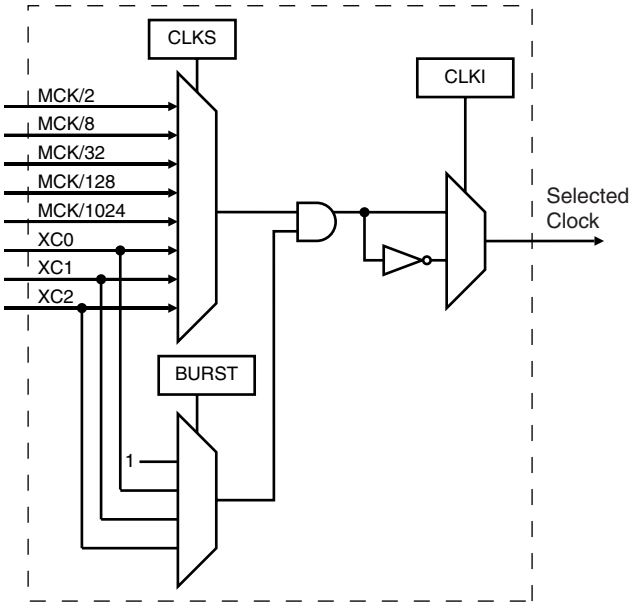
- Internal clock signals: MCK/2, MCK/8, MCK/32, MCK/128, MCK/1024
- External clock signals: XC0, XC1 or XC2

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 (MCK) period. The external clock frequency must be at least 2.5 times lower than the system clock.

Figure 49. Clock Selection

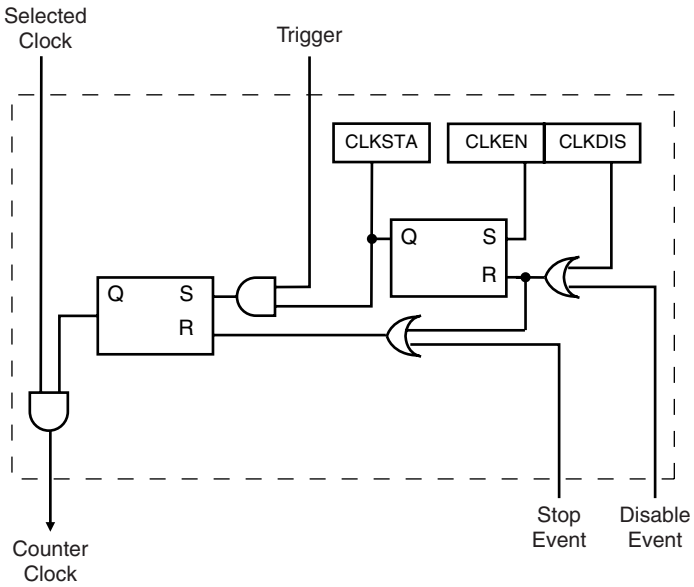


Clock Control

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

- 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.
- 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 effect only if the clock is enabled.

Figure 50. Clock Control



Timer Counter Operating Modes

Each Timer Counter channel can independently operate in two different modes:

- Capture Mode allows measurement on signals
- Waveform Mode allows wave generation

The Timer Counter 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.

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:

- Software Trigger: Each channel has a software trigger, available by setting SWTRG in TC_CCR.
- 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.
- 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 (MCK) period in order to be detected.

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 considered as input.

Figure 51 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. This means that they can be loaded with the counter value when a programmable event occurs on the signal TIOA.

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.

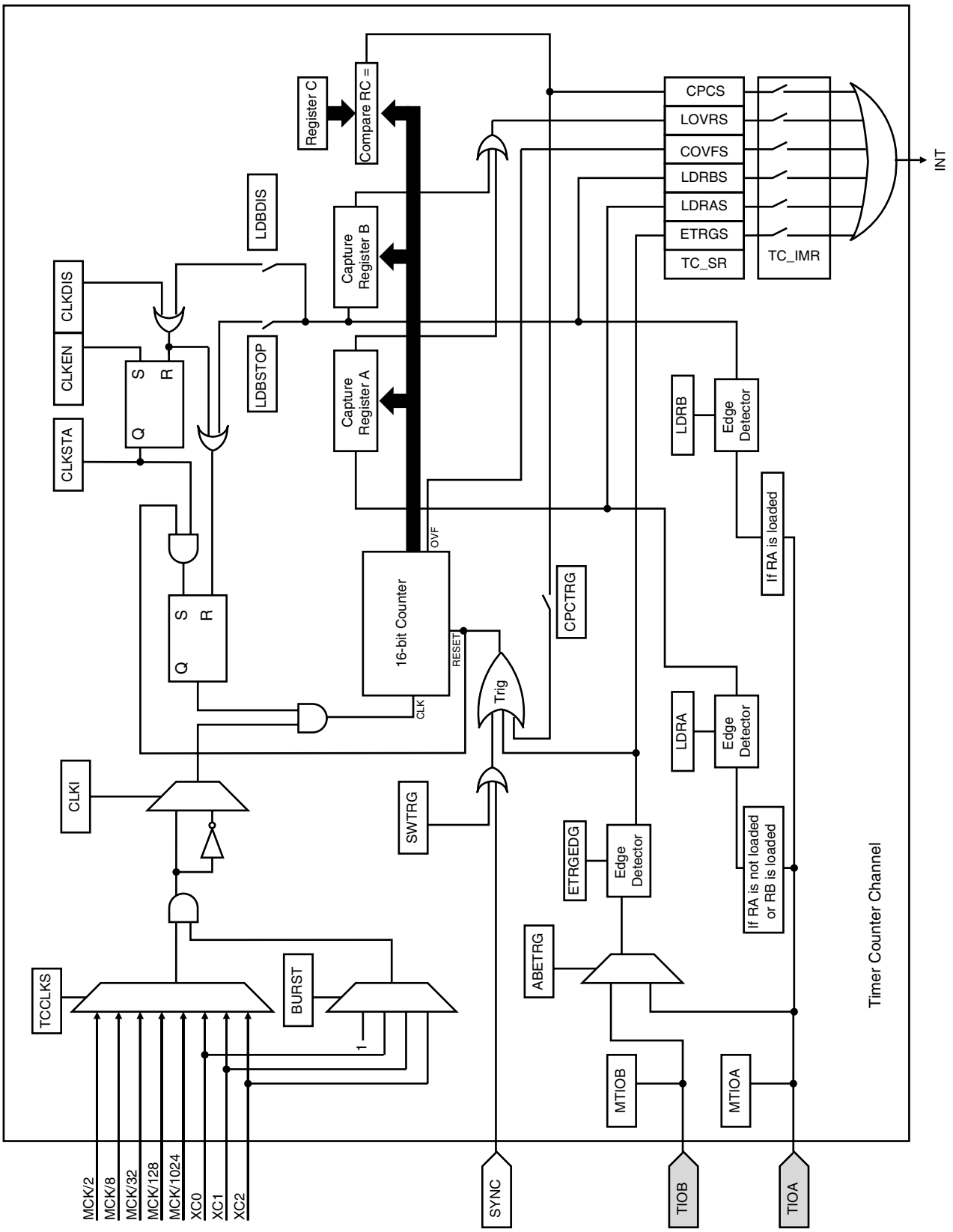
Bit ABETRG in TC_CMR selects input signal TIOA or TIOB as an external trigger. Parameter ETRGEDG defines the edge (rising, falling or both) detected to generate an external trigger. If ETRGEDG = 0 (none), the external trigger is disabled.

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 51. 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 52 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. 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.

The tables below show which parameter in TC_CMR is used to define the effect of each event.

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

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

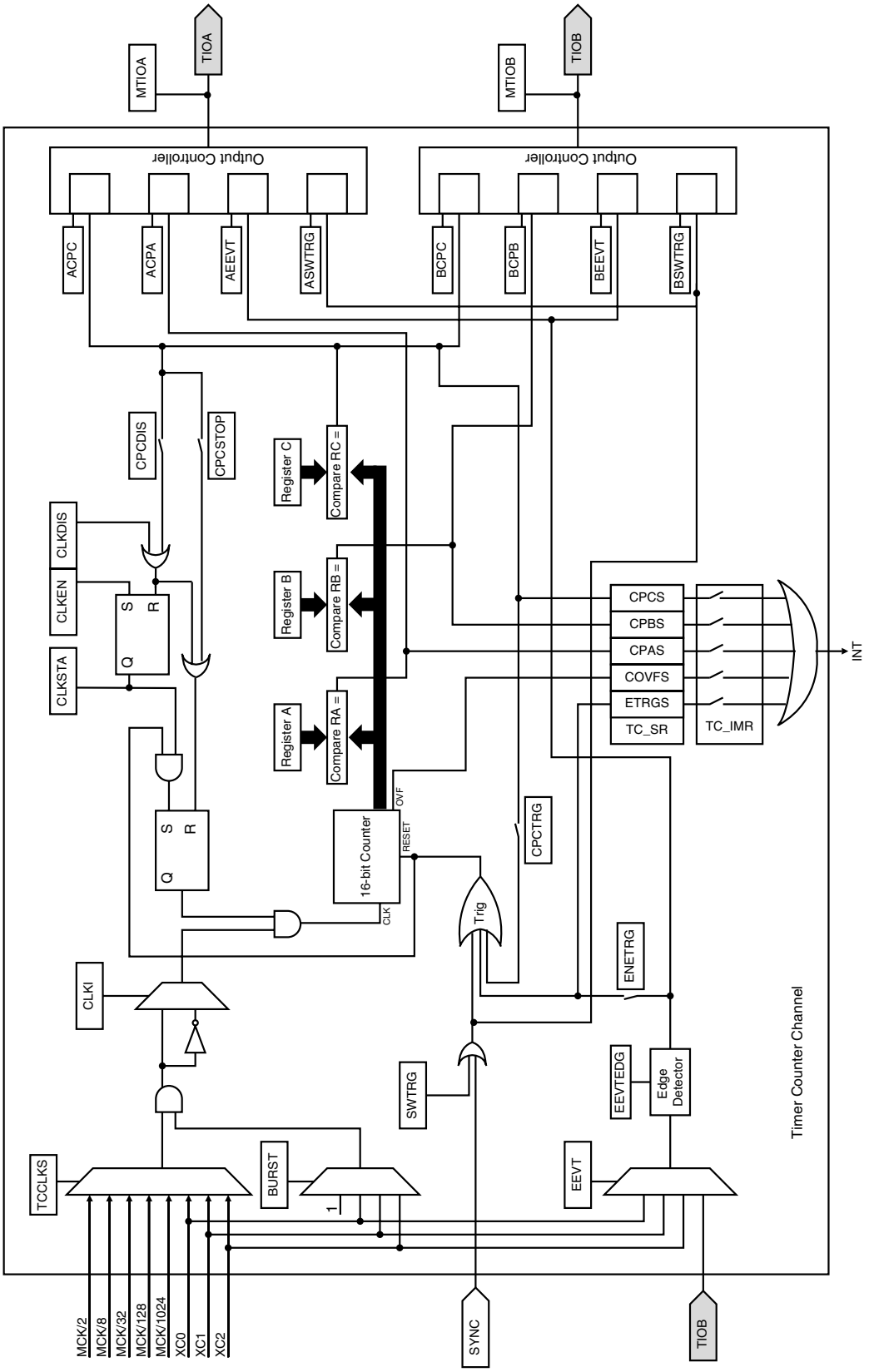
If two or more events occur at the same time, the priority level is defined as follows:
1. Software trigger
2. External event
3. RC compare
4. RA or RB compare

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 52. Waveform Mode



TC User Interface

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

Table 18. TC Global Memory Map

Offset	Channel/Register	Name	Access	Reset State
0x00	TC Channel 0	See Table 19		
0x40	TC Channel 1	See Table 19		
0x80	TC Channel 2	See Table 19		
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. The offset of each of the Channel registers in Table 19 is in relation to the offset of the corresponding channel as mentioned in Table 18.

Table 19. TC Channel Memory Map

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

TC Block Control Register

Register Name: TC_BCR
Access Type: Write-only
Offset: 0xC0

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 (Code Label `TC_SYNC`)**
  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
Reset State: 0
Offset: 0xC4

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	Code Label: `TC_TC0XC0S`
0	0	TCLK0	`TC_TCLK0XC0`
0	1	None	`TC_NONEXC0`
1	0	TIOA1	`TC_TIOA1XC0`
1	1	TIOA2	`TC_TIOA2XC0`

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

TC1XC1S		Signal Connected to XC1	Code Label: TC_TC1XC1S
0	0	TCLK1	TC_TCLK1XC1
0	1	None	TC_NONEXC1
1	0	TIOA0	TC_TIOA0XC1
1	1	TIOA2	TC_TIOA2XC1

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

TC2XC2S		Signal Connected to XC2	Code Label: TC_TC2XC2S
0	0	TCLK2	TC_TCLK2XC2
0	1	None	TC_NONEXC2
1	0	TIOA0	TC_TIOA0XC2
1	1	TIOA1	TC_TIOA1XC2

TC Channel Control Register

Register Name: TC_CCR
Access Type: Write-only
Offset: 0x00

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 (Code Label TC_CLKEN)**
  0 = No effect.
  1 = Enables the clock if CLKDIS is not 1.
- **CLKDIS: Counter Clock Disable Command (Code Label TC_CLKDIS)**
  0 = No effect.
  1 = Disables the clock.
- **SWTRG: Software Trigger Command (Code Label TC_SWTRG)**
  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
Reset State: 0
Offset: 0x04

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=0	CPCTRG	–	–	–	ABETRG	ETRGEDG	

7	6	5	4	3	2	1	0
LDBDIS	LDBSTOP	BURST		CLKI	TCCLKS		

- **TCCLKS: Clock Selection**

TCCLKS			Clock Selected	Code Label: TC_CLKS
0	0	0	MCK/2	TC_CLKS_MCK2
0	0	1	MCK/8	TC_CLKS_MCK8
0	1	0	MCK/32	TC_CLKS_MCK32
0	1	1	MCK/128	TC_CLKS_MCK128
1	0	0	MCK/1024	TC_CLKS_MCK1024
1	0	1	XC0	TC_CLKS_XC0
1	1	0	XC1	TC_CLKS_XC1
1	1	1	XC2	TC_CLKS_XC2

- **CLKI: Clock Invert (Code Label TC_CLKI)**
  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		Selected BURST	Code Label: TC_BURST
0	0	The clock is not gated by an external signal.	TC_BURST_NONE
0	1	XC0 is ANDed with the selected clock.	TC_BURST_XC0
1	0	XC1 is ANDed with the selected clock.	TC_BURST_XC1
1	1	XC2 is ANDed with the selected clock.	TC_BURST_XC2

- **LDBSTOP: Counter Clock Stopped with RB Loading (Code Label TC_LDBSTOP)**
  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 (Code Label TC_LDBDIS)**
  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	Code Label: TC_ETRGEDG
0	0	None	TC_ETRGEDG_EDGE_NONE
0	1	Rising edge	TC_ETRGEDG_RISING_EDGE
1	0	Falling edge	TC_ETRGEDG_FALLING_EDGE
1	1	Each edge	TC_ETRGEDG_BOTH_EDGE

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

ABETRG	Selected ABETRG	Code Label: TC_ABETRG
0	TIOB is used as an external trigger.	TC_ABETRG_TIOB
1	TIOA is used as an external trigger.	TC_ABETRG_TIOA

- **CPCTRG: RC Compare Trigger Enable (Code Label TC_CPCTRG)**
  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 (Code Label TC_WAVE)**
  0 = Capture Mode is enabled.
  1 = Capture Mode is disabled (Waveform Mode is enabled).
- **LDRA: RA Loading Selection**

LDRA		Edge	Code Label: TC_LDRA
0	0	None	TC_LDRA_EDGE_NONE
0	1	Rising edge of TIOA	TC_LDRA_RISING_EDGE
1	0	Falling edge of TIOA	TC_LDRA_FALLING_EDGE
1	1	Each edge of TIOA	TC_LDRA_BOTH_EDGE

- **LDRB: RB Loading Selection**

LDRB		Edge	Code Label: TC_LDRB
0	0	None	TC_LDRB_EDGE_NONE
0	1	Rising edge of TIOA	TC_LDRB_RISING_EDGE
1	0	Falling edge of TIOA	TC_LDRB_FALLING_EDGE
1	1	Each edge of TIOA	TC_LDRB_BOTH_EDGE

TC Channel Mode Register: Waveform Mode

Register Name: TC_CMR
Access Type: Read/Write
Reset State: 0
Offset: 0x4

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=1	CPCTRG	–	ENETRG	EEVT		EEVTEDG	
7	6	5	4	3	2	1	0
CPCDIS	CPCSTOP	BURST		CLKI	TCCLKS		

- **TCCLKS: Clock Selection**

TCCLKS			Clock Selected	Code Label: TC_CLKS
0	0	0	MCK/2	TC_CLKS_MCK2
0	0	1	MCK/8	TC_CLKS_MCK8
0	1	0	MCK/32	TC_CLKS_MCK32
0	1	1	MCK/128	TC_CLKS_MCK128
1	0	0	MCK/1024	TC_CLKS_MCK1024
1	0	1	XC0	TC_CLKS_XC0
1	1	0	XC1	TC_CLKS_XC1
1	1	1	XC2	TC_CLKS_XC2

- **CLKI: Clock Invert (Code Label TC_CLKI)**
  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		Selected BURST	Code Label: TC_BURST
0	0	The clock is not gated by an external signal.	TC_BURST_NONE
0	1	XC0 is ANDed with the selected clock.	TC_BURST_XC0
1	0	XC1 is ANDed with the selected clock.	TC_BURST_XC1
1	1	XC2 is ANDed with the selected clock.	TC_BURST_XC2

- **CPCSTOP: Counter Clock Stopped with RC Compare (Code Label TC_CPCSTOP)**
  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 (Code Label TC_CPCDIS)**
  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	Code Label: TC_EEVTEDG
0	0	None	TC_EEVTEDG_EDGE_NONE
0	1	Rising edge	TC_EEVTEDG_RISING_EDGE
1	0	Falling edge	TC_EEVTEDG_FALLING_EDGE
1	1	Each edge	TC_EEVTEDG_BOTH_EDGE

- **EEVT: External Event Selection**

EEVT		Signal Selected as External Event	TIOB Direction	Code Label: TC_EEVT
0	0	TIOB	Input(1)	TC_EEVT_TIOB
0	1	XC0	Output	TC_EEVT_XC0
1	0	XC1	Output	TC_EEVT_XC1
1	1	XC2	Output	TC_EEVT_XC2

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 (Code Label TC_ENETRG)**
  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 (Code Label TC_CPCTRG)**
  0 = RC Compare has no effect on the counter and its clock.
  1 = RC Compare resets the counter and starts the counter clock.
- **WAVE = 1 (Code Label TC_WAVE)**
  0 = Waveform Mode is disabled (Capture Mode is enabled).
  1 = Waveform Mode is enabled.
- **ACPA: RA Compare Effect on TIOA**

ACPA		Effect	Code Label: TC_ACPA
0	0	None	TC_ACPA_OUTPUT_NONE
0	1	Set	TC_ACPA_SET_OUTPUT
1	0	Clear	TC_ACPA_CLEAR_OUTPUT
1	1	Toggle	TC_ACPA_TOGGLE_OUTPUT

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

ACPC		Effect	Code Label: TC_ACPC
0	0	None	TC_ACPC_OUTPUT_NONE
0	1	Set	TC_ACPC_SET_OUTPUT
1	0	Clear	TC_ACPC_CLEAR_OUTPUT
1	1	Toggle	TC_ACPC_TOGGLE_OUTPUT

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

AEEVT		Effect	Code Label: TC_AEEVT
0	0	None	TC_AEEVT_OUTPUT_NONE
0	1	Set	TC_AEEVT_SET_OUTPUT
1	0	Clear	TC_AEEVT_CLEAR_OUTPUT
1	1	Toggle	TC_AEEVT_TOGGLE_OUTPUT

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

ASWTRG		Effect	Code Label: TC_ASWTRG
0	0	None	TC_ASWTRG_OUTPUT_NONE
0	1	Set	TC_ASWTRG_SET_OUTPUT
1	0	Clear	TC_ASWTRG_CLEAR_OUTPUT
1	1	Toggle	TC_ASWTRG_TOGGLE_OUTPUT

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

BCPB		Effect	Code Label: TC_BCPB
0	0	None	TC_BCPB_OUTPUT_NONE
0	1	Set	TC_BCPB_SET_OUTPUT
1	0	Clear	TC_BCPB_CLEAR_OUTPUT
1	1	Toggle	TC_BCPB_TOGGLE_OUTPUT

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

BCPC		Effect	Code Label: TC_BCPC
0	0	None	TC_BCPC_OUTPUT_NONE
0	1	Set	TC_BCPC_SET_OUTPUT
1	0	Clear	TC_BCPC_CLEAR_OUTPUT
1	1	Toggle	TC_BCPC_TOGGLE_OUTPUT

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

BEEVT		Effect	Code Label: TC_BEEVT
0	0	None	TC_BEEVT_OUTPUT_NONE
0	1	Set	TC_BEEVT_SET_OUTPUT
1	0	Clear	TC_BEEVT_CLEAR_OUTPUT
1	1	Toggle	TC_BEEVT_TOGGLE_OUTPUT

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

BSWTRG		Effect	Code Label: TC_BSWTRG
0	0	None	TC_BSWTRG_OUTPUT_NONE
0	1	Set	TC_BSWTRG_SET_OUTPUT
1	0	Clear	TC_BSWTRG_CLEAR_OUTPUT
1	1	Toggle	TC_BSWTRG_TOGGLE_OUTPUT

TC Counter Value Register

Register Name: TC_CVR
Access Type: Read-only
Reset State: 0
Offset: 0x10

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 (Code Label `TC_CV`)**
  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
Reset State: 0
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
RA							

7	6	5	4	3	2	1	0
RA							

- **RA: Register A (Code Label `TC_RA`)**
  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
Reset State: 0
Offset: 0x18

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 (Code Label `TC_RB`)**
  RB contains the Register B value in real-time.

TC Register C

Register Name: TC_RC
Access Type: Read/Write
Reset State: 0
Offset: 0x1C

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 (Code Label `TC_RC`)**
  RC contains the Register C value in real-time.

TC Status Register

Register Name: TC_SR
Access Type: Read/Write
Offset: 0x20

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 (Code Label `TC_COVFS`)**
  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 (Code Label `TC_LOVRS`)**
  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 (Code Label `TC_CPAS`)**
  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 (Code Label `TC_CPBS`)**
  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 (Code Label `TC_CPCS`)**
  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 (Code Label `TC_LDRAS`)**
  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 (Code Label `TC_LDRBS`)**
  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 (Code Label `TC_ETRGS`)**
  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 (Code Label `TC_CLKSTA`)**
  0 = Clock is disabled.
  1 = Clock is enabled.
- **MTIOA: TIOA Mirror (Code Label `TC_MTIOA`)**
  0 = TIOA is low. If WAVE = 0, this means that TIOA pin is low. If WAVE = 1, this means that TIOA is driven low.
  1 = TIOA is high. If WAVE = 0, this means that TIOA pin is high. If WAVE = 1, this means that TIOA is driven high.
- **MTIOB: TIOB Mirror (Code Label `TC_MTIOB`)**
  0 = TIOB is low. If WAVE = 0, this means that TIOB pin is low. If WAVE = 1, this means that TIOB is driven low.
  1 = TIOB is high. If WAVE = 0, this means that TIOB pin is high. If WAVE = 1, this means that TIOB is driven high.

TC Interrupt Enable Register

Register Name: TC_IER
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
ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS

- **COVFS: Counter Overflow (Code Label `TC_COVFS`)**
  0 = No effect.
  1 = Enables the Counter Overflow Interrupt.
- **LOVRS: Load Overrun (Code Label `TC_LOVRS`)**
  0 = No effect.
  1: Enables the Load Overrun Interrupt.
- **CPAS: RA Compare (Code Label `TC_CPAS`)**
  0 = No effect.
  1 = Enables the RA Compare Interrupt.
- **CPBS: RB Compare (Code Label `TC_CPBS`)**
  0 = No effect.
  1 = Enables the RB Compare Interrupt.
- **CPCS: RC Compare (Code Label `TC_CPCS`)**
  0 = No effect.
  1 = Enables the RC Compare Interrupt.
- **LDRAS: RA Loading (Code Label `TC_LDRAS`)**
  0 = No effect.
  1 = Enables the RA Load Interrupt.
- **LDRBS: RB Loading (Code Label `TC_LDRBS`)**
  0 = No effect.
  1 = Enables the RB Load Interrupt.
- **ETRGS: External Trigger (Code Label `TC_ETRGS`)**
  0 = No effect.
  1 = Enables the External Trigger Interrupt.

TC Interrupt Disable Register

Register Name: TC_IDR
Access Type: Write-only
Offset: 0x28

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 (Code Label `TC_COVFS`)**
  0 = No effect.
  1 = Disables the Counter Overflow Interrupt.
- **LOVRS: Load Overrun (Code Label `TC_LOVRS`)**
  0 = No effect.
  1 = Disables the Load Overrun Interrupt (if WAVE = 0).
- **CPAS: RA Compare (Code Label `TC_CPAS`)**
  0 = No effect.
  1 = Disables the RA Compare Interrupt (if WAVE = 1).
- **CPBS: RB Compare (Code Label `TC_CPBS`)**
  0 = No effect.
  1 = Disables the RB Compare Interrupt (if WAVE = 1).
- **CPCS: RC Compare (Code Label `TC_CPCS`)**
  0 = No effect.
  1 = Disables the RC Compare Interrupt.
- **LDRAS: RA Loading (Code Label `TC_LDRAS`)**
  0 = No effect.
  1 = Disables the RA Load Interrupt (if WAVE = 0).
- **LDRBS: RB Loading (Code Label `TC_LDRBS`)**
  0 = No effect.
  1 = Disables the RB Load Interrupt (if WAVE = 0).
- **ETRGS: External Trigger (Code Label `TC_ETRGS`)**
  0 = No effect.
  1 = Disables the External Trigger Interrupt.

TC Interrupt Mask Register

Register Name: TC_IMR
Access Type: Read-only
Reset State: 0
Offset: 0x2C

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 (Code Label `TC_COVFS`)**
  0 = The Counter Overflow Interrupt is disabled.
  1 = The Counter Overflow Interrupt is enabled.
- **LOVRS: Load Overrun (Code Label `TC_LOVRS`)**
  0 = The Load Overrun Interrupt is disabled.
  1 = The Load Overrun Interrupt is enabled.
- **CPAS: RA Compare (Code Label `TC_CPAS`)**
  0 = The RA Compare Interrupt is disabled.
  1 = The RA Compare Interrupt is enabled.
- **CPBS: RB Compare (Code Label `TC_CPBS`)**
  0 = The RB Compare Interrupt is disabled.
  1 = The RB Compare Interrupt is enabled.
- **CPCS: RC Compare (Code Label `TC_CPCS`)**
  0 = The RC Compare Interrupt is disabled.
  1 = The RC Compare Interrupt is enabled.
- **LDRAS: RA Loading (Code Label `TC_LDRAS`)**
  0 = The Load RA Interrupt is disabled.
  1 = The Load RA Interrupt is enabled.
- **LDRBS: RB Loading (Code Label `TC_LDRBS`)**
  0 = The Load RB Interrupt is disabled.
  1 = The Load RB Interrupt is enabled.
- **ETRGS: External Trigger (Code Label `TC_ETRGS`)**
  0 = The External Trigger Interrupt is disabled.
  1 = The External Trigger Interrupt is enabled.

SPI: Serial Peripheral Interface

The AT91M55800A includes an SPI which provides communication with external devices in master or slave mode.

The SPI has four external chip selects which can be connected to up to 15 devices. The data length is programmable, from 8- to 16-bit.

As for the USART, a 2-channel PDC can be used to move data between memory and the SPI without CPU intervention.

Pin Description

Seven pins are associated with the SPI Interface. When not needed for the SPI function, each of these pins can be configured as a PIO.

Support for an external master is provided by the PIO Controller Multi-driver option. To configure an SPI pin as open-drain to support external drivers, set the corresponding bits in the PIO_MDSR register (see).

An input filter can be enabled on the SPI input pins by setting the corresponding bits in the PIO_IFSR.

The NPCS0/NSS pin can function as a peripheral chip select output or slave select input. Refer to Table 20 for a description of the SPI pins.

Figure 53. SPI Block Diagram

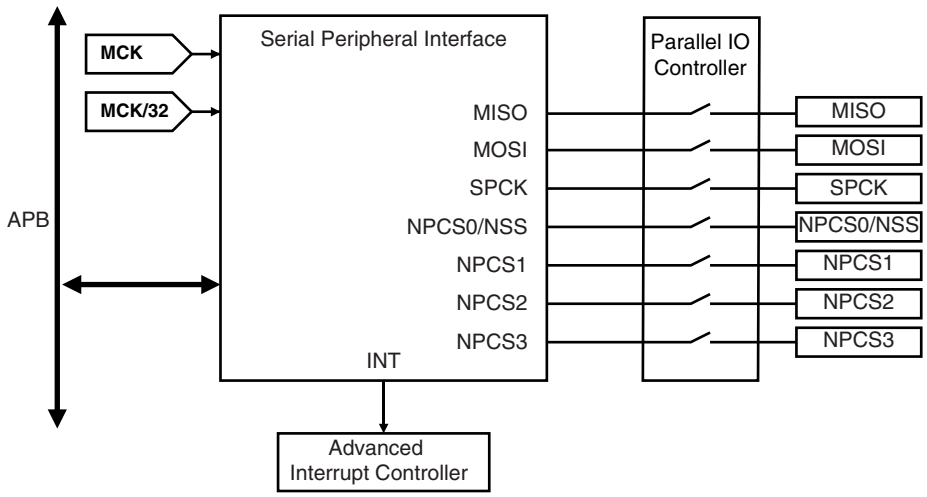


Table 20. SPI Pins

Pin Name	Mnemonic	Mode	Function
Master In Slave Out	MISO	Master Slave	Serial data input to SPI Serial data output from SPI
Master Out Slave In	MOSI	Master Slave	Serial data output from SPI Serial data input to SPI
Serial Clock	SPCK	Master Slave	Clock output from SPI Clock input to SPI
Peripheral Chip Selects	NPCS[3:1]	Master	Select peripherals
Peripheral Chip Select/ Slave Select	NPCS0/ NSS	Master Master Slave	Output: Selects peripheral Input: low causes mode fault Input: chip select for SPI

Notes: 1. After a hardware reset, the SPI clock is disabled by default. The user must configure the Power Management Controller before any access to the User Interface of the SPI.
2. After a hardware reset, the SPI pins are deselected by default (see “PIO: Parallel I/O Controller” on page 105). The user must configure the PIO Controller to enable the corresponding pins for their SPI function. NPCS0/NSS must be configured as open drain in the Parallel I/O Controller for multi-master operation.

Master Mode

In Master Mode, the SPI 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 ARM core writes to the SP_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 SP_TDR (Transmit Data Register) the SPI continues to transfer data. If the SP_RDR (Receive Data Register) has not been read before new data is received, the Overrun Error (OVRES) flag is set.

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 SP_CSR0 to SP_CSR3 (Chip Select Registers). See Table 21.

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

Figures 54 and 55 show the operation of the SPI in Master Mode. For details concerning the flag and control bits in these diagrams, see the tables in the Programmer’s Model, starting on page 181.

Fixed Peripheral Select

This mode is ideal 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 SP_MR (Mode Register). The peripheral is defined by the PCS field, also in SP_MR.

This option is only available when the SPI is programmed in master mode.

Variable Peripheral Select

Variable Peripheral Select is activated by setting bit PS to one. The PCS field in SP_TDR (Transmit Data Register) is used to select the destination peripheral. The data transfer characteristics are changed when the selected peripheral changes, according to the associated chip select register.

The PCS field in the SP_MR has no effect.

This option is only available 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 SP_MR (Mode Register) selects 1 to 4 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 SP_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 SP_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 SP_MR before writing new data in SP_TDR.

The value on the NPCS pins at the end of each transfer can be read in the SP_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 NPCS0/NSS signal.

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

Figure 54. Functional Flow Diagram in Master Mode

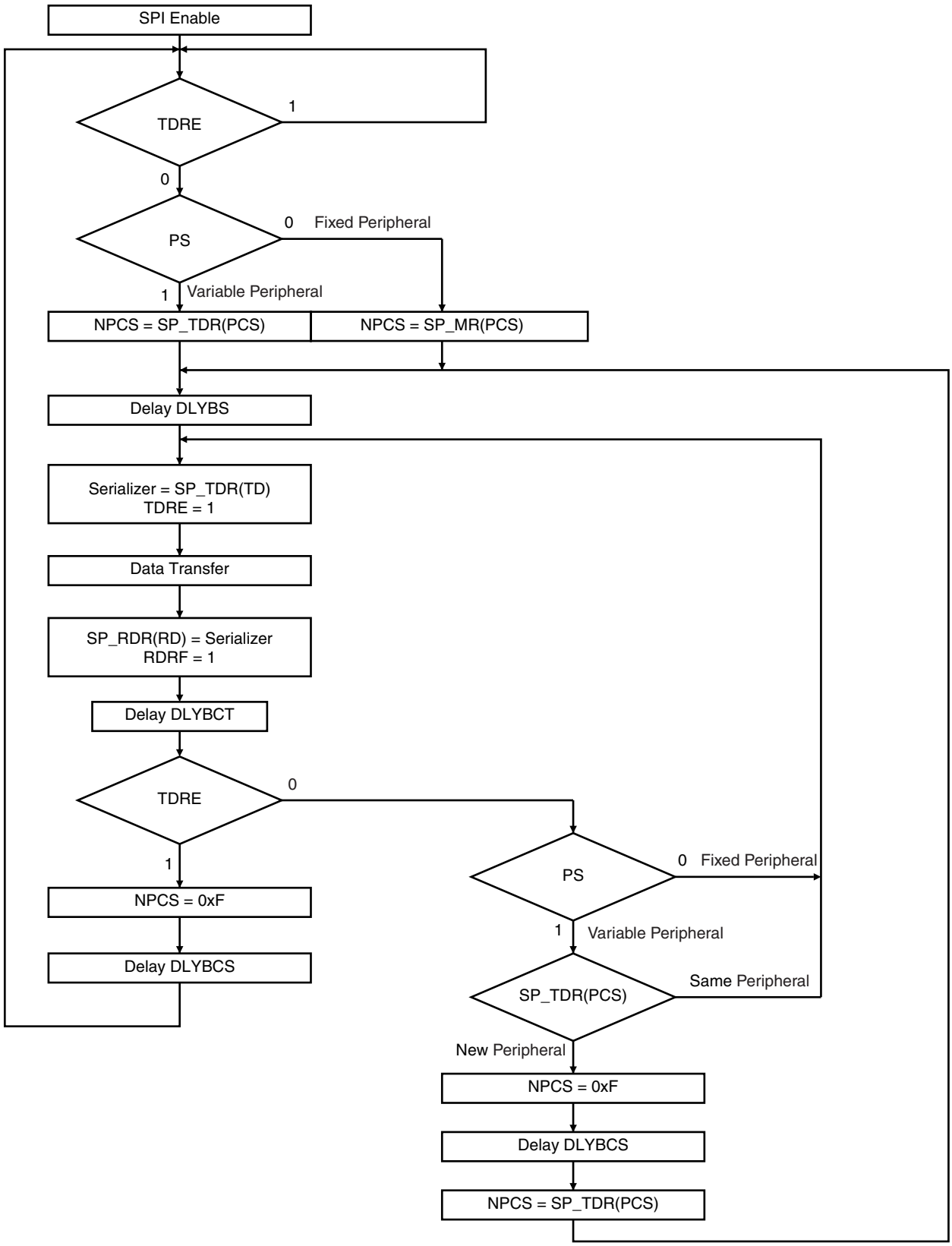
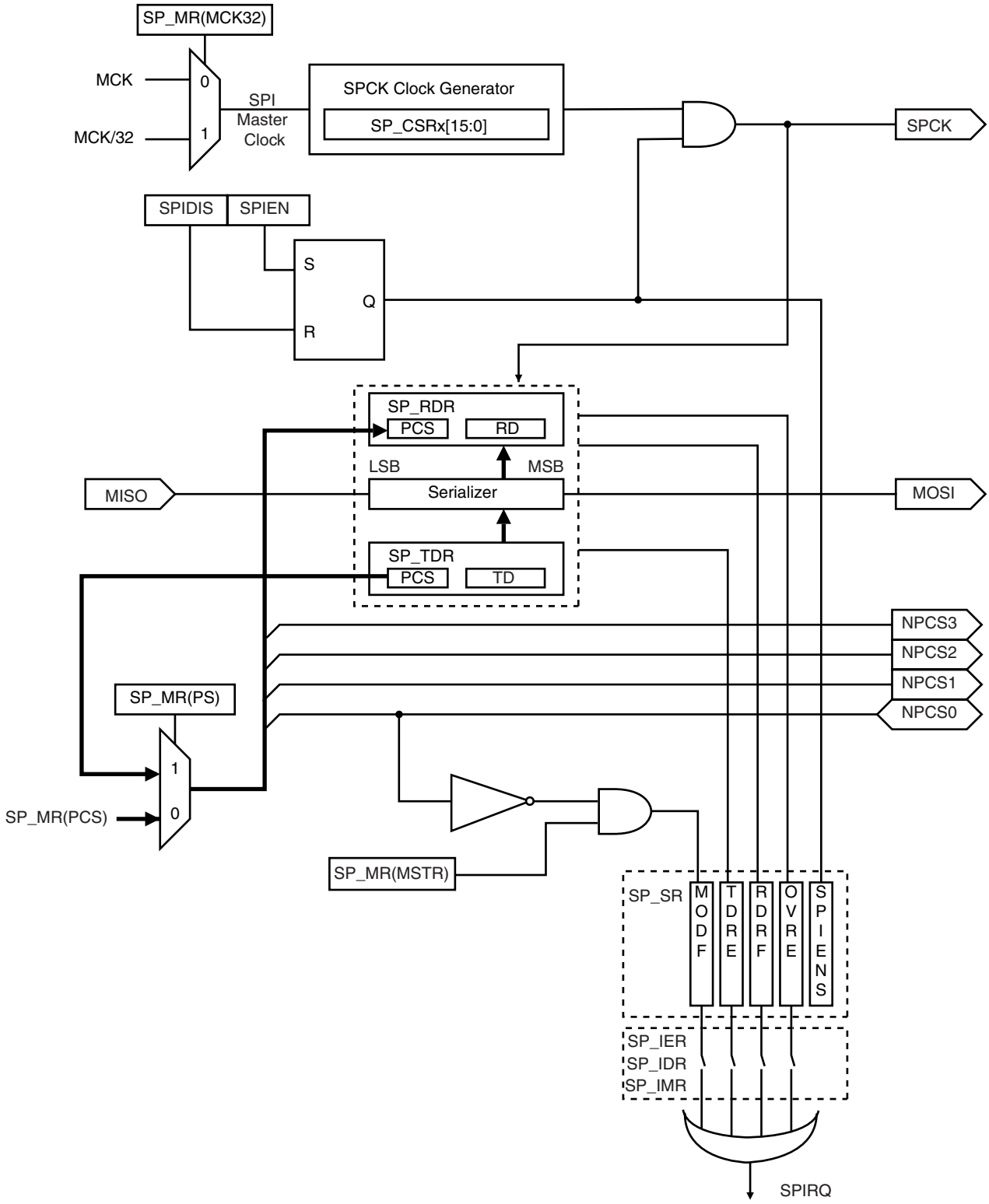


Figure 55. SPI in Master Mode

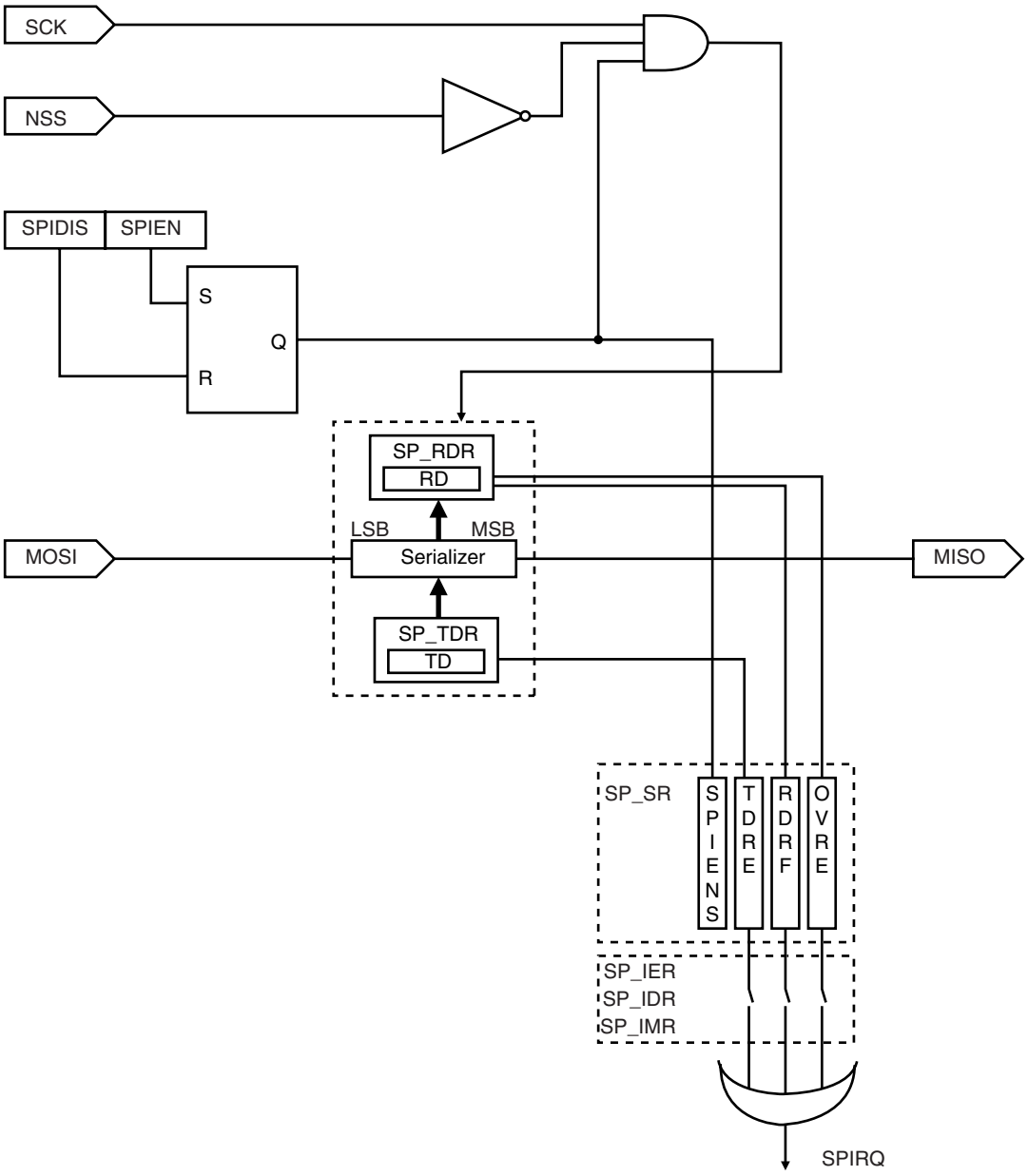


Slave Mode

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 56. SPI in Slave Mode



Data Transfer

The following waveforms show examples of data transfers.

Figure 57. SPI Transfer Format (NCPHA equals One, 8 bits per transfer)

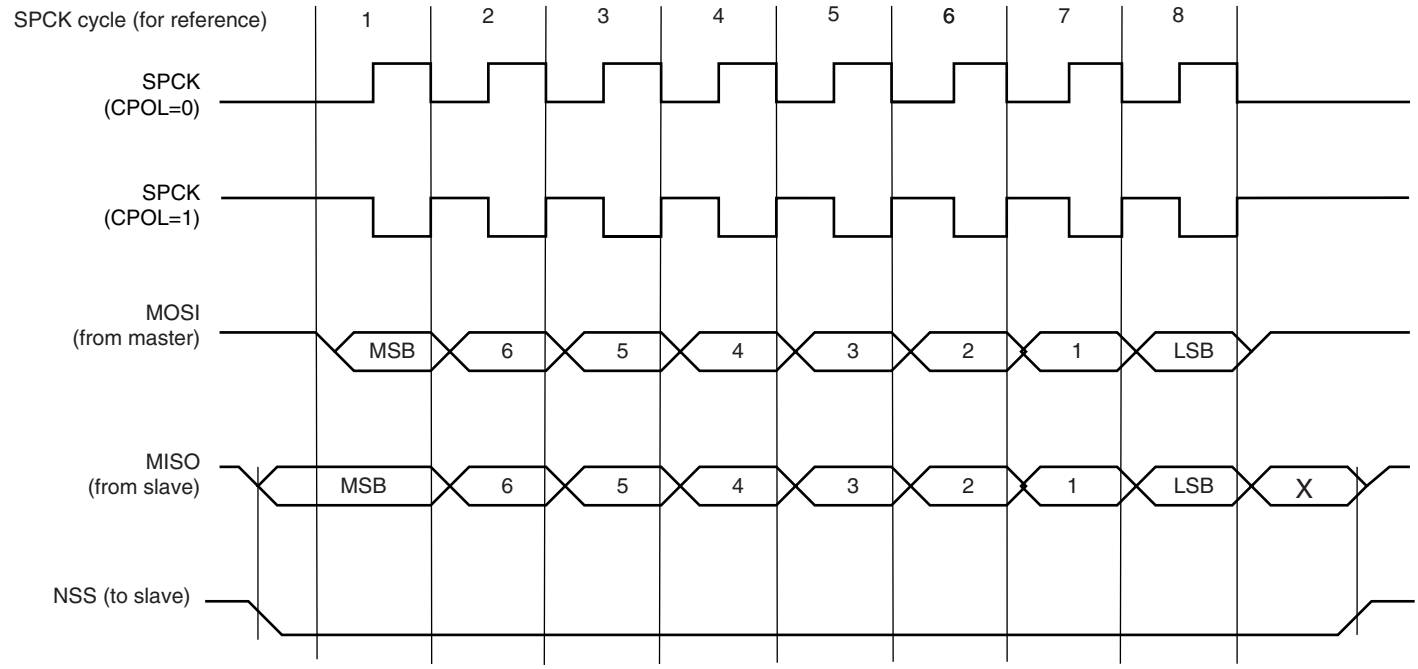


Figure 58. SPI Transfer Format (NCPHA equals Zero, 8 bits per transfer)

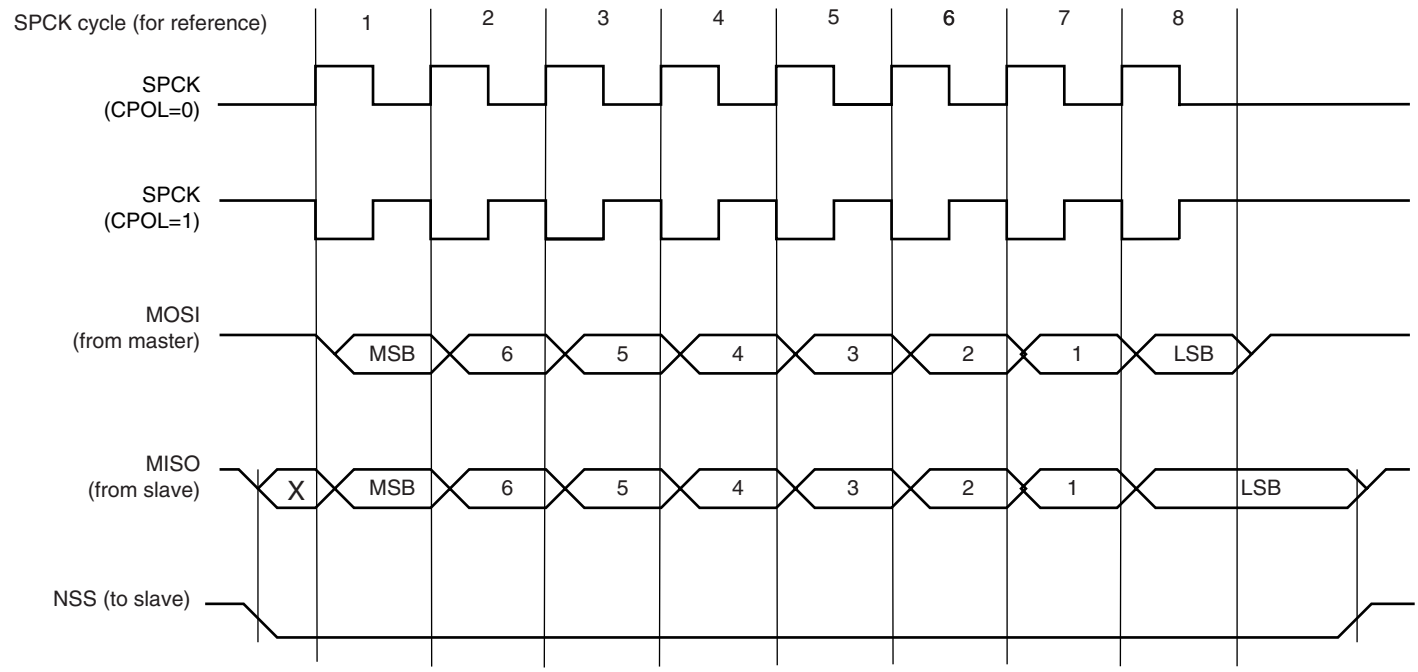
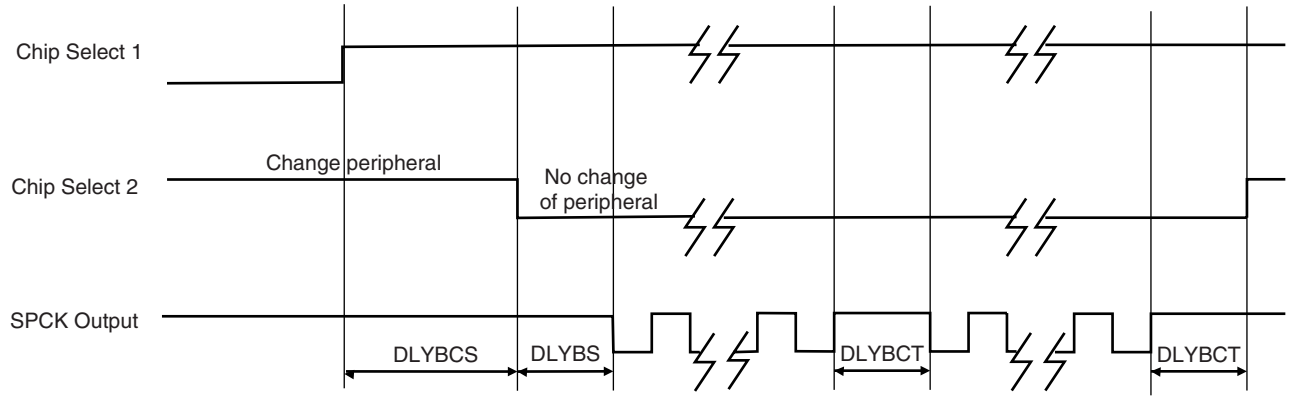


Figure 59. Programmable Delays (DLYBCS, DLYBS and DLYBCT)



Clock Generation

In master mode the SPI Master Clock is either MCK or MCK/32, as defined by the MCK32 field of SP_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 strictly be longer than two system clock (MCK) periods.

Peripheral Data Controller

The SPI is closely connected to two Peripheral Data Controller channels. One is dedicated to the receiver. The other is dedicated to the transmitter.

The PDC channel is programmed using SP_TPR (Transmit Pointer) and SP_TCR (Transmit Counter) for the transmitter and SP_RPR (Receive Pointer) and SP_RCR (Receive Counter) for the receiver. The status of the PDC is given in SP_SR by the SPENDTX bit for the transmitter and by the SPENDRX bit for the receiver.

The pointer registers (SP_TPR and SP_RPR) are used to store the address of the transmit or receive buffers. The counter registers (SP_TCR and SP_RCR) are used to store the size of these buffers.

The receiver data transfer is triggered by the RDRF bit and the transmitter data transfer is triggered by TDRE. When a transfer is performed, the counter is decremented and the pointer is incremented. When the counter reaches 0, the status bit is set (SPENDRX for the receiver, SPENDTX for the transmitter in SP_SR) and can be programmed to generate an interrupt. While the counter is at zero, the status bit is asserted and transfers are disabled.

SPI Programmer's Model

SPI Base Address: 0xFFFBC000 (Code Label `SPI_BASE`)

Table 21. SPI Memory Map

Offset	Register	Name	Access	Reset State
0x00	Control Register	SP_CR	Write-only	–
0x04	Mode Register	SP_MR	Read/Write	0
0x08	Receive Data Register	SP_RDR	Read-only	0
0x0C	Transmit Data Register	SP_TDR	Write-only	–
0x10	Status Register	SP_SR	Read-only	0
0x14	Interrupt Enable Register	SP_IER	Write-only	–
0x18	Interrupt Disable Register	SP_IDR	Write-only	–
0x1C	Interrupt Mask Register	SP_IMR	Read-only	0
0x20	Receive Pointer Register	SP_RPR	Read/Write	0
0x24	Receive Counter Register	SP_RCR	Read/Write	0
0x28	Transmit Pointer Register	SP_TPR	Read/Write	0
0x2C	Transmit Counter Register	SP_TCR	Read/Write	0
0x30	Chip Select Register 0	SP_CSR0	Read/Write	0
0x34	Chip Select Register 1	SP_CSR1	Read/Write	0
0x38	Chip Select Register 2	SP_CSR2	Read/Write	0
0x3C	Chip Select Register 3	SP_CSR3	Read/Write	0

SPI Control Register

Register Name: SP_CR
Access Type: Write-only
Offset: 0x00

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 (Code Label `SP_SPIEN`)**
  0 = No effect.
  1 = Enables the SPI to transfer and receive data.
- **SPIDIS: SPI Disable (Code Label `SP_SPIDIS`)**
  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 (Code Label `SP_SWRST`)**
  0 = No effect.
  1 = Resets the SPI.
  A software triggered hardware reset of the SPI interface is performed.

SPI Mode Register

Register Name: SP_MR
Access Type: Read/Write
Reset State: 0
Offset: 0x04

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	–	–	–	MCK32	PCSDEC	PS	MSTR

- **MSTR: Master/Slave Mode (Code Label `SP_MSTR`)**
  0 = SPI is in Slave mode.
  1 = SPI is in Master mode.
  MSTR configures the SPI Interface for either master or slave mode operation.

- **PS: Peripheral Select**

PS	Selected PS	Code Label: SP_PS
0	Fixed Peripheral Select	SP_PS_FIXED
1	Variable Peripheral Select	SP_PS_VARIABLE

- **PCSDEC: Chip Select Decode (Code Label SP_PCSDEC)**
  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:
  SP_CSR0 defines peripheral chip select signals 0 to 3.
  SP_CSR1 defines peripheral chip select signals 4 to 7.
  SP_CSR2 defines peripheral chip select signals 8 to 11.
  SP_CSR3 defines peripheral chip select signals 12 to 15[1].

Note: 1. The 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.

- **MCK32: Clock Selection (Code Label SP_DIV32)**
  0 = SPI Master Clock equals MCK.
  1 = SPI Master Clock equals MCK/32.
- **LLB: Local Loopback Enable (Code Label SP_LLB)**
  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 (Code Label SP_PCS)**
  This field is only used if Fixed Peripheral Select is active (PS=0).
  If PCSDEC=0:
  PCS = xxx0 NPCS[3:0] = 1110 (Code Label SP_PCS0)
  PCS = xx01 NPCS[3:0] = 1101 (Code Label SP_PCS1)
  PCS = x011 NPCS[3:0] = 1011 (Code Label SP_PCS2)
  PCS = 0111 NPCS[3:0] = 0111 (Code Label SP_PCS3)
  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 (Code Label SP_DLYBCS)**
  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:
  Delay_ Between_Chip_Selects = DLYBCS * SPI_Master_Clock_period

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.

SPI Transmit Data Register

Register Name: SP_TDR
Access Type: Write-only
Offset: 0x0C

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 (Code Label `SP_TD`)**
  Data which is to be transmitted by the SPI Interface 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) and if the SPI is in Master Mode.
  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

Register Name: SP_SR
Access Type: Read-only
Reset State: 0
Offset: 0x10

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
–	–	SPENDTX	SPENDRX	OVRES	MODF	TDRE	RDRF

- **RDRF: Receive Data Register Full (Code Label `SP_RDRF`)**
  0 = No data has been received since the last read of SP_RDR.
  1 = Data has been received and the received data has been transferred from the serializer to SP_RDR since the last read of SP_RDR.
- **TDRE: Transmit Data Register Empty (Code Label `SP_TDRE`)**
  0 = Data has been written to SP_TDR and not yet transferred to the serializer.
  1 = The last data written in the Transmit Data Register has been transferred in 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 (Code Label `SP_MODF`)**
  0 = No Mode Fault has been detected since the last read of SP_SR.
  1 = A Mode Fault occurred since the last read of the SP_SR.
- **OVRES: Overrun Error Status (Code Label `SP_OVRES`)**
  0 = No overrun has been detected since the last read of SP_SR.
  1 = An overrun has occurred since the last read of SP_SR.
  An overrun occurs when SP_RDR is loaded at least twice from the serializer since the last read of the SP_RDR.
- **SPENDRX: End of Receiver Transfer (Code Label `SP_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.
- **SPENDTX: End of Transmitter Transfer (Code Label `SP_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.
- **SPIENS: SPI Enable Status (Code Label `SP_SPIENS`)**
  0 = SPI is disabled.
  1 = SPI is enabled.

SPI Interrupt Enable Register

Register Name: SP_IER
Access Type: Write-only
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
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	SPENDTX	SPENDRX	OVRES	MODF	TDRE	RDRF

- **RDRF: Receive Data Register Full Interrupt Enable (Code Label `SP_RDRF`)**
  0 = No effect.
  1 = Enables the Receiver Data Register Full Interrupt.
- **TDRE: SPI Transmit Data Register Empty Interrupt Enable (Code Label `SP_TDRE`)**
  0 = No effect.
  1 = Enables the Transmit Data Register Empty Interrupt.
- **MODF: Mode Fault Error Interrupt Enable (Code Label `SP_MODF`)**
  0 = No effect.
  1 = Enables the Mode Fault Interrupt.
- **OVRES: Overrun Error Interrupt Enable (Code Label `SP_OVRES`)**
  0 = No effect.
  1 = Enables the Overrun Error Interrupt.
- **SPENDRX: End of Receiver Transfer Interrupt Enable (Code Label `SP_ENDRX`)**
  0 = No effect.
  1 = Enables the End of Receiver Transfer Interrupt.
- **SPENDTX: End of Transmitter Transfer Interrupt Enable (Code Label `SP_ENDTX`)**
  0 = No effect.
  1 = Enables the End of Transmitter Transfer Interrupt.

SPI Interrupt Disable Register

Register Name: SP_IDR
Access Type: Write-only
Offset: 0x18

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
–	–	SPENDTX	SPENDRX	OVRES	MODF	TDRE	RDRF

- **RDRF: Receive Data Register Full Interrupt Disable (Code Label `SP_RDRF`)**
  0 = No effect.
  1 = Disables the Receiver Data Register Full Interrupt.
- **TDRE: Transmit Data Register Empty Interrupt Disable (Code Label `SP_TDRE`)**
  0 = No effect.
  1 = Disables the Transmit Data Register Empty Interrupt.
- **MODF: Mode Fault Interrupt Disable (Code Label `SP_MODF`)**
  0 = No effect.
  1 = Disables the Mode Fault Interrupt.
- **OVRES: Overrun Error Interrupt Disable (Code Label `SP_OVRES`)**
  0 = No effect.
  1 = Disables the Overrun Error Interrupt.
- **SPENDRX: End of Receiver Transfer Interrupt Disable (Code Label `SP_ENDRX`)**
  0 = No effect.
  1 = Disables the End of Receiver Transfer Interrupt.
- **SPENDTX: End of Transmitter Transfer Interrupt Disable (Code Label `SP_ENDTX`)**
  0 = No effect.
  1 = Disables the End of Transmitter Transfer Interrupt.

SPI Interrupt Mask Register

Register Name: SP_IMR
Access Type: Read-only
Reset State: 0
Offset: 0x1C

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
–	–	SPENDTX	SPENDRX	OVRES	MODF	TDRE	RDRF

- **RDRF: Receive Data Register Full Interrupt Mask (Code Label `SP_RDRF`)**
  0 = Receive Data Register Full Interrupt is disabled.
  1 = Receive Data Register Full Interrupt is enabled.
- **TDRE: Transmit Data Register Empty Interrupt Mask (Code Label `SP_TDRE`)**
  0 = Transmit Data Register Empty Interrupt is disabled.
  1 = Transmit Data Register Empty Interrupt is enabled.
- **MODF: Mode Fault Interrupt Mask (Code Label `SP_MODF`)**
  0 = Mode Fault Interrupt is disabled.
  1 = Mode Fault Interrupt is enabled.
- **OVRES: Overrun Error Interrupt Mask (Code Label `SP_OVRES`)**
  0 = Overrun Error Interrupt is disabled.
  1 = Overrun Error Interrupt is enabled.
- **SPENDRX: End of Receiver Transfer Interrupt Mask (Code Label `SP_ENDRX`)**
  0 = End of Receiver Transfer Interrupt is disabled.
  1 = End of Receiver Transfer Interrupt is enabled.
- **SPENDTX: End of Transmitter Transfer Interrupt Mask (Code Label `SP_ENDTX`)**
  0 = End of Transmitter Transfer Interrupt is disabled.
  1 = End of Transmitter Transfer Interrupt is enabled.

SPI Receive Pointer Register

Name: SP_RPR
Access Type: Read/Write
Reset State: 0
Offset: 0x20

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

Name: SP_RCR
Access Type: Read/Write
Reset State: 0
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
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 RDRF is active.

SPI Transmit Pointer Register

Name: SP_TPR
Access Type: Read/Write
Reset State: 0
Offset: 0x28

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.

SPI Transmit Counter Register

Name: SP_TCR
Access Type: Read/Write
Reset State: 0
Offset: 0x2C

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 TDRE is active.

SPI Chip Select Register

Register Name: SP_CSR0.. SP_CSR3
Access Type: Read/Write
Reset State: 0
Offset: 0x30.....0x3C

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 (Code Label `SP_CPOL`)**
  0 = The inactive state value of SPCK is logic level zero.
  1 = The inactive state value of SPCK is logic level one.
  CPOL is used to determine the inactive state value of the serial clock (SPCK). It is used with NCPHA to produce a desired clock/data relationship between master and slave devices.
- **NCPHA: Clock Phase (Code Label `SP_NCPHA`)**
  0 = Data is changed on the leading edge of SPCK and captured on the following edge of SPCK.
  1 = Data is captured on the leading edge of SPCK and changed on the following edge of SPCK.
  NCPHA determines which edge of SPCK causes data to change and which edge causes data to be captured. NCPHA is used with CPOL to produce a desired 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	Code Label: `SP_BITS`
0000	8	SP_BITS_8
0001	9	SP_BITS_9
0010	10	SP_BITS_10
0011	11	SP_BITS_11
0100	12	SP_BITS_12
0101	13	SP_BITS_13
0110	14	SP_BITS_14
0111	15	SP_BITS_15
1000	16	SP_BITS_16
1001	Reserved	–
1010	Reserved	–
1011	Reserved	–
1100	Reserved	–
1101	Reserved	–
1110	Reserved	–
1111	Reserved	–

- **SCBR: Serial Clock Baud Rate (Code Label `SP_SCBR`)**
  In Master Mode, the SPI Interface uses a modulus counter to derive the SPCK baud rate from the SPI Master Clock (selected between MCK and MCK/32). 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:

$$\text{SPCK\_Baud\_Rate} = \frac{\text{SPI\_Master\_Clock\_frequency}}{2 \times \text{SCBR}}$$

  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 SPCK (Code Label `SP_DLYBS`)**
  This field defines the delay from NPCS valid to the first valid SPCK transition.
  When DLYBS equals zero, the NPCS valid to SPCK transition is 1/2 the SPCK clock period.
  Otherwise, the following equation determines the delay:
  NPCS_to_SPCK_Delay = DLYBS * SPI_Master_Clock_period
- **DLYBCT: Delay Between Consecutive Transfers (Code Label `SP_DLYBCT`)**
  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

ADC: Analog-to-digital Converter

The AT91M55800A features two identical 4-channel 10-bit Analog-to-digital converters (ADC) based on a Successive Approximation Register (SAR) approach.

Each ADC has 4 analog input pins (AD0 to AD3 and AD4 to AD7), digital trigger input pins (AD0TRIG and AD1TRIG), and provides an interrupt signal to the AIC. Both ADCs share the analog power supply pins (VDDA and GNDA) and the input reference voltage pin (ADVREF).

Figure 60. Block Diagram

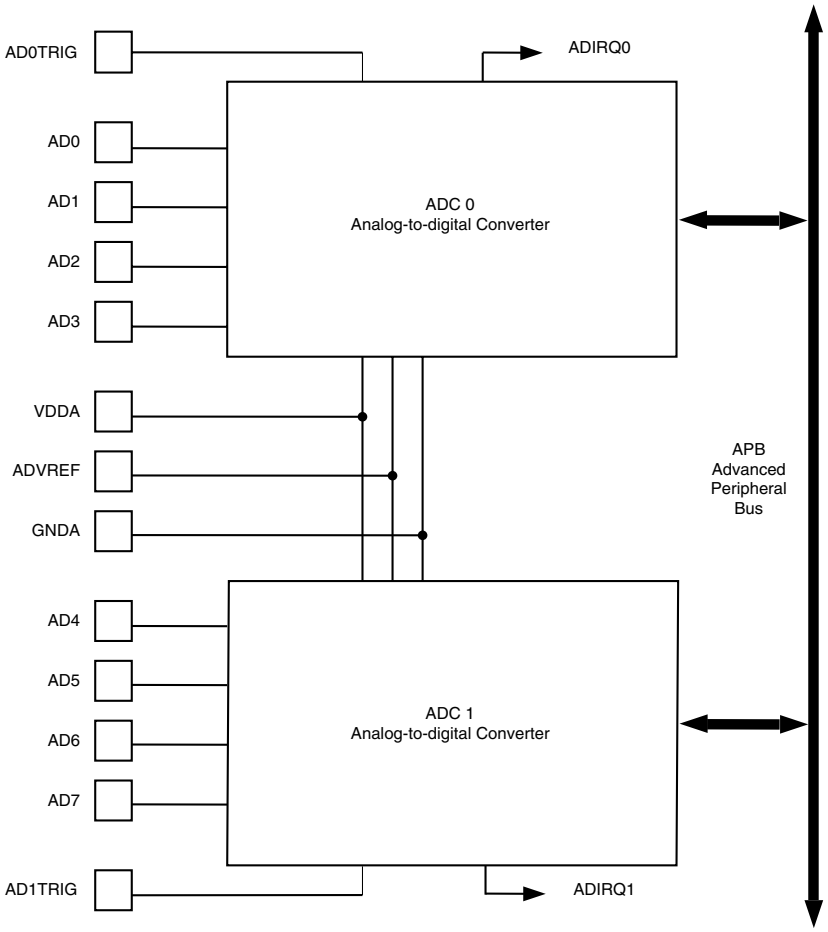


Table 22. ADC Pin Description

Pin Name	Description
VDDA	Analog power supply
GNDA	Analog ground
ADVREF	Reference voltage
AD0 - AD7	Analog input channels
AD0TRIG, AD1TRIG	External triggers

Analog-to-digital Conversion

The ADC has an internal sample-and-hold circuit that holds the sampled analog value during a complete conversion.

The reference voltage pin ADVREF allows the analog input conversion range to be set between 0 and ADVREF. Analog inputs between these voltages convert to values based on a linear conversion.

The ADC uses the ADC Clock to perform the conversion. To convert a single analog value to a 10-bit digital data requires 11 ADC clock cycles. The ADC Clock frequency is selected in the PRESCAL field of the Mode Register (ADC_MR).

Conversion Results

When a conversion is complete, the resulting 10-bit digital value is stored in the Convert Data Register (ADC_CDR) of the selected channel, and the corresponding EOC flag in the Status Register (ADC_SR) is set. This bit can provide an interrupt signal and is automatically cleared when the corresponding Convert Data Register (ADC_CDR) is read.

If the ADC_CDR is not read before further incoming data is converted, the corresponding Overrun Error (OVRE) flag is set in the Status Register (ADC_SR).

The ADC offers an 8-bit or 10-bit operating mode. By default after a reset, the ADC operates in 10-bit mode. If the bit RES in ADC_MR is set, the 8-bit mode is selected.

When operating in 10-bit mode, the field DATA in ADC_CDR is fully significant.

When operating in 8-bit mode, only the 8 lowest bits of DATA are significant and the 2 highest bits read 0.

Conversion Triggers

Conversions of the active analog channels are started with a software or a hardware trigger. The software trigger is provided by writing the bit START in the Control Register (ADC_CR).

The hardware trigger can be one of the TIOA outputs of the Timer Counter channels, or the external trigger input of the ADC (AD0TRIG for the ADC0 or AD1TRIG for ADC1). The hardware trigger is selected with the field TRGSEL in the Mode Register (ADC_MR). The selected hardware trigger is enabled with the bit TRGEN in the Mode Register (ADC_MR).

If a hardware trigger is selected, the start of a conversion is detected at each rising edge of the selected signal. If one of the TIOA outputs is selected, the corresponding Timer Counter channel must be programmed in Waveform Mode.

Only one start command is necessary to initiate a conversion sequence on all the channels. The ADC hardware logic automatically performs the conversions on the active channels, then waits for a new request. The Channel Enable (ADC_CHER) and Channel Disable (ADC_CHDR) Registers enable the analog channels to be enabled or disabled independently.

Sleep Mode

The AT91 ADC Sleep Mode maximizes power saving by deactivating the ADC when it is not being used for conversions. Sleep Mode is selected by setting the bit SLEEP in the Mode Register ADC_MR.

When a start conversion request occurs, the ADC is automatically activated. As the analog cell requires a start-up time, the logic waits during this time and starts the conversion sequence on the enabled channel. When all conversions are complete, the ADC is deactivated until the next trigger.

This permits an automatic conversion sequence with minimum CPU intervention and optimized power consumption.

ADC User Interface

Base Address ADC 0:0xFFFB0000 (Code Label `ADC0_BASE`)
Base Address ADC 1:0xFFFB4000 (Code Label `ADC1_BASE`)

Table 23. ADC Memory Map

Offset	Register	Name	Access	Reset State
0x00	Control Register	ADC_CR	Write-only	–
0x04	Mode Register	ADC_MR	Read/Write	0
0x08	Reserved	–	–	–
0x0C	Reserved	–	–	–
0x10	Channel Enable Register	ADC_CHER	Write-only	–
0x14	Channel Disable Register	ADC_CHDR	Write-only	–
0x18	Channel Status Register	ADC_CHSR	Read-only	0
0x1C	Reserved	–	–	–
0x20	Status Register	ADC_SR	Read-only	0
0x24	Interrupt Enable Register	ADC_IER	Write-only	–
0x28	Interrupt Disable Register	ADC_IDR	Write-only	–
0x2C	Interrupt Mask Register	ADC_IMR	Read-only	0
0x30	Convert Data Register 0	ADC_CDR0	Read-only	0
0x34	Convert Data Register 1	ADC_CDR1	Read-only	0
0x38	Convert Data Register 2	ADC_CDR2	Read-only	0
0x3C	Convert Data Register 3	ADC_CDR3	Read-only	0

ADC Control Register

Register Name: ADC_CR
Access Type: Write-only
Offset: 0x00

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
–	–	–	–	–	–	START	SWRST

- **SWRST: Software Reset (Code Label `ADC_SWRST`)**
  0 = No effect.
  1 = Resets the ADC simulating a hardware reset.
- **START: Start Conversion (Code Label `ADC_START`)**
  0 = No effect.
  1 = Begins analog-to-digital conversion and clears all EOC bits.

ADC Mode Register

Register Name: ADC_MR
Access Type: Read/Write
Reset State: 0
Offset: 0x04

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

- **TRGEN: Trigger Enable .**

TRGEN	Selected TRGEN	Code Label
0	Hardware triggers are disabled. Starting a conversion is only possible by software.	ADC_TRGEN_DIS
1	Hardware trigger selected by TRGSEL field is enabled.	ADC_TRGEN_EN

- **TRGSEL: Trigger Selection**
  This field selects the hardware trigger.

TTRGSEL			Selected TRGSEL	Code Label: ADC_B_TTRGSEL
0	0	0	TIOA0	ADC_TRG_TIOA0
0	0	1	TIOA1	ADC_TRG_TIOA1
0	1	0	TIOA2	ADC_TRG_TIOA2
0	1	1	TIOA3	ADC_TRG_TIOA3
1	0	0	TIOA4	ADC_TRG_TIOA4
1	0	1	TIOA5	ADC_TRG_TIOA5
1	1	0	External trigger	ADC_TRG_EXT
1	1	1	Reserved	–

- **RES: Resolution.**

RES	Selected RES	Code Label
0	10-bit resolution	ADC_10_BIT_RES
1	8-bit resolution	ADC_8_BIT_RES

- **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.

ADC Channel Enable Register

Register Name: ADC_CHER
Access Type: Write-only
Offset: 0x10

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
–	–	–	–	CH3	CH2	CH1	CH0

- **CH: Channel Enable (Code Label `ADC_CHx`)**
  0 = No effect.
  1 = Enables the corresponding channel.

ADC Channel Disable Register

Register Name: ADC_CHDR
Access Type: Write-only
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
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	CH3	CH2	CH1	CH0

- **CH: Channel Disable (Code Label `ADC_CHx`)**
  0 = No effect.
  1 = Disables the corresponding channel.

ADC Channel Status Register

Register Name: ADC_CHSR
Access Type: Read-only
Reset State: 0
Offset: 0x18

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
–	–	–	–	CH3	CH2	CH1	CH0

- **CH: Channel Status (Code Label `ADC_CHx`)**
  0 = Corresponding channel is disabled.
  1 = Corresponding channel is enabled.

ADC Status Register

Register Name: ADC_SR
Access Type: Read-only
Reset State: 0
Offset: 0x20

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	OVRE3	OVRE2	OVRE1	OVRE0
7	6	5	4	3	2	1	0
–	–	–	–	EOC3	EOC2	EOC1	EOC0

- **EOC: End of Conversion (Code Label `ADC_EOCx`)**
  0 = Corresponding analog channel is disabled, or the conversion is not finished.
  1 = Corresponding analog channel is enabled and conversion is complete.
- **OVRE: Enable Overrun Error Interrupt (Code Label `ADC_OVREx`)**
  0 = No overrun on the corresponding channel since the last read of ADC_SR.
  1 = There has been an overrun on the corresponding channel since the last read of ADC_SR.

ADC Interrupt Enable Register

Register Name: ADC_IER
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
–	–	–	–	OVRE3	OVRE2	OVRE1	OVRE0
7	6	5	4	3	2	1	0
–	–	–	–	EOC3	EOC2	EOC1	EOC0

- **EOC: End of Conversion Interrupt Enable (Code Label `ADC_EOCx`)**
  0 = No effect.
  1 = Enables the End of Conversion Interrupt.
- **OVRE: Overrun Error Interrupt Enable (Code Label `ADC_OVREx`)**
  0 = No effect.
  1 = Enables the Overrun Error Interrupt.

ADC Interrupt Disable Register

Register Name: ADC_IDR
Access Type: Write-only
Offset: 0x28

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	OVRE3	OVRE2	OVRE1	OVRE0
7	6	5	4	3	2	1	0
–	–	–	–	EOC3	EOC2	EOC1	EOC0

- **EOC: End of Conversion Interrupt Disable (Code Label `ADC_EOCx`)**
  0 = No effect.
  1 = Disables the End of Conversion Interrupt.
- **OVRE: Overrun Error Interrupt Disable (Code Label `ADC_OVREx`)**
  0 = No effect.
  1 = Disables the Overrun Error Interrupt.

ADC Interrupt Mask Register

Register Name: ADC_IMR
Access Type: Read-only
Reset State: 0
Offset: 0x2C

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	OVRE3	OVRE2	OVRE1	OVRE0
7	6	5	4	3	2	1	0
–	–	–	–	EOC3	EOC2	EOC1	EOC0

- **EOC: End of Conversion Interrupt Mask (Code Label `ADC_EOCx`)**
  0 = End of Conversion Interrupt is disabled.
  1 = End of Conversion Interrupt is enabled.
- **OVRE: Overrun Error Interrupt Mask (Code Label `ADC_OVREx`)**
  0 = Overrun Error Interrupt is disabled.
  1 = Overrun Error Interrupt is enabled.

ADC Convert Data Register

Register Name: ADC_CDR0 to ADC_CDR3
Access Type: Read-only
Reset State: 0
Offset: 0x30 to 0x3C

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

- **DATA: Converted Data**
  The analog-to-digital conversion data is placed into this register at the end of a conversion and remains until a new conversion is completed. The Convert Data Register (CDR) is only loaded if the corresponding analog channel is enabled.

DATA	Selected DATA	Code Label: `ADC_CDRx`
0 or 1	10-bits Data	ADC_DATA_10BITS
0 or 1	8-bits Data	ADC_DATA_8BITS

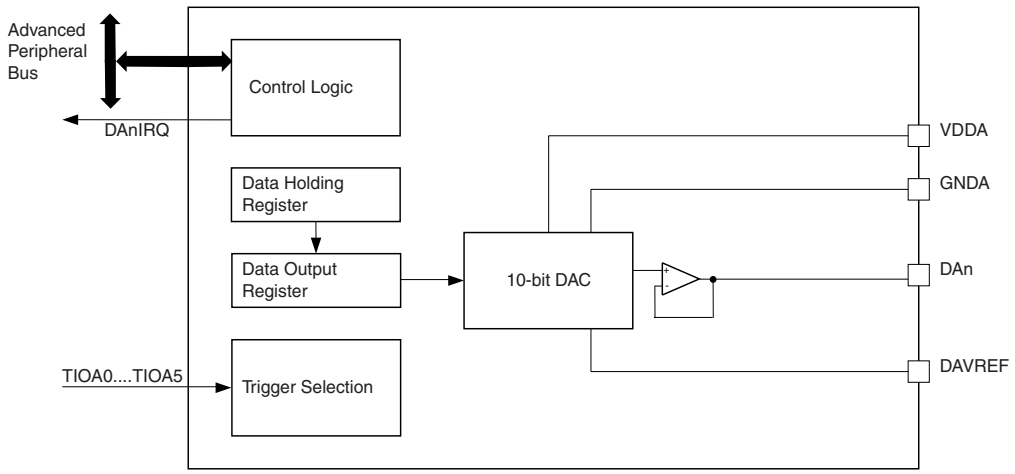
DAC: Digital-to-analog Converter

The AT91M55800A features two identical 1-channel 10-bit Digital-to-analog converters (DAC).

Each DAC has an analog output pin (DA0 and DA1) and provides an interrupt signal to the AIC (DA0IRQ and DA1IRQ). Both DACs share the analog power supply pins VDDA and GNDA, and the input reference pin DAVREF.

Pin Name	Meaning
VDDA	Analog power supply
GNDA	Analog ground
DAVREF	Reference voltage
DA0	Analog output, channel 0
DA1	Analog output, channel 1

Figure 61. DAC Block Diagram



Conversion Details

Digital-to-analog conversions are possible only if the DAC has been enabled in the APMC and the startup time has elapsed. This startup time is a maximum of 5 µsec, and is indicated more precisely in the Electrical Characteristics datasheet of the device as parameter t_{DASU}.

Digital inputs are converted to output voltages on a linear conversion between 0 and DAVREF. The analog output voltages on DA0 and DA1 pins are determined by the following equation:

DA = DAVREF x (DAC_DOR / 1024)

When DAC_DOR (Data Output Register) is loaded, the analog output voltage is available after a settling time of approximately 5 µsec. The exact value depends on the power supply voltage and the analog output load, and is indicated in the Electrical Characteristics Sheet of the device as parameter t_{DAST}.

The output register cannot be written directly and any data transfer to the DAC must be performed by writing in DAC_DHR (Data Holding Register). The transfer from DAC_DHR to DAC_DOR is performed automatically or when an hardware trigger occurs, depending on the bit TRGEN in DAC_MR (Mode Register).

The DAC integrates an output buffer enabling the reduction of the output impedance, and the possibility of driving external loads directly, without having to add an external operational amplifier. The maximum load supported by the output buffer is indicated in the Electrical Characteristics of the device.

8- or 10-bit Conversion Mode

Bit RES in the Mode Register (DAC_MR) selects between 8-bit or 10-bit modes of operation. In 8-bit mode, the data written in DAC_DHR is automatically shifted left two bits and the two lowest bits are written 0. The bit RES also affects the type of transfers performed by the PDC channel:

- in 8-bit mode, only a byte transfer is performed .
- in 10-bit mode, a half-word transfer (16 bits) is performed.

Trigger Selection

A conversion is triggered when data is written in DAC_DHR and TRGEN in DAC_MR is 0.

If TRGEN is 1, a hardware trigger is selected by the field TTRGSEL between the Timer Counter Channel outputs TIOA. In this case, the corresponding Timer Counter channel must be programmed in Waveform Mode, and each time the DAC detects a rising edge on the TC output, it transfers the last data written in DAC_DHR into DAC_DOR.

The bit DATRDY traces the fact that a valid data has been written in DAC_DHR and not yet been transferred in DAC_DOR. An interrupt can be generated from this status bit to tell the software to load the following value.

DAC User Interface

Base Address DAC 0:0xFFFA8000 (Code Label `DAC0_BASE`)
Base Address DAC 1:0xFFFAC000 (Code Label `DAC1_BASE`)

Table 24. DAC Memory Map

Offset	Register	Name	Access	Reset State
0x00	Control Register	DAC_CR	Write-only	–
0x04	Mode Register	DAC_MR	Read/Write	0
0x08	Data Holding Register	DAC_DHR	Read/Write	0
0x0C	Data Output Register	DAC_DOR	Read-only	0
0x10	Status Register	DAC_SR	Read-only	0
0x14	Interrupt Enable Register	DAC_IER	Write-only	–
0x18	Interrupt Disable Register	DAC_IDR	Write-only	–
0x1C	Interrupt Mask Register	DAC_IMR	Read-only	0

DAC Control Register

Register Name: DAC_CR
Access Type: Write-only
Offset: 0x00

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

- **SWRST: Software Reset (Code Label `DAC_SWRST`)**
  0 = No effect.
  1 = Resets the DAC. A software-triggered reset of the DAC interface is performed.

DAC Mode Register

Register Name: DAC_MR
Access Type: Read/Write
Reset State: 0
Offset: 0x04

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
–	–	–	RES	TTRGSEL			TTRGEN

- **TTRGEN: Timer Trigger Enable (Code Label `DAC_TTRGEN_EN`)**

TTRGEN	Selected TTRGEN	Code Label
0	The data written into the Data Holding Register (DAC_DHR) is transferred one main clock cycle later to the data output register (DAC_DOR).	DAC_TTRGEN_DIS
1	The data transfer from the DAC_DHR to the DAC_DOR is synchronized by the timer trigger.	DAC_TTRGEN_EN

- **TTRGSEL: Timer Trigger Selection**
  Only used if TTRGEN = 1

TTRGSEL			Selected Timer Trigger	Code Label
				`DAC_TTRGSEL`
0	0	0	TIOA0	DAC_TRG_TIOA0
0	0	1	TIOA1	DAC_TRG_TIOA1
0	1	0	TIOA2	DAC_TRG_TIOA2
0	1	1	TIOA3	DAC_TRG_TIOA3
1	0	0	TIOA4	DAC_TRG_TIOA4
1	0	1	TIOA5	DAC_TRG_TIOA5
1	1	X	Reserved	–

- **RES: Resolution**

RES	Selected RES	Code Label
0	10-bit resolution	DAC_10_BIT_RES
1	8-bit resolution	DAC_8_BIT_RES

DAC Data Holding Register

Register Name: DAC_DHR
Access Type: Read/Write
Reset State: 0
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
–	–	–	–	–	–	DATA	

7	6	5	4	3	2	1	0
DATA							

- **DATA: Data to be Converted (Code Label `DAC_DATA_10BITS` or `DAC_DATA_8BITS` depending on RES)**
  Data that is to be converted by the DAC is stored in this register. Data to be converted must be written in a right-aligned format.
  In 8-bit resolution mode (RES = 1), data written into the Data Holding Register will be shifted to the left by 2 bits and the two LSBs will be 0.
  In both 8-bit and 10-bit modes, data will be read as written after the adjustments are done. All non-significant bits read 0.

DAC Output Register

Register Name: DAC_DOR
Access Type: Read-only
Reset State: 0
Offset: 0x0C

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

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

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

7	6	5	4	3	2	1	0
DATA							

- **DATA: Data being Converted (Code Label `DAC_DATA_10BITS` or `DAC_DATA_8BITS` depending on RES)**
  Data being converted is stored, in a right-aligned format, in this register.
  All non-significant bits read 0.

DAC Status Register

Register Name: DAC_SR
Access Type: Read-only
Reset State: 0
Offset: 0x10

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
–	–	–	–	–	–	–	DATRDY

- **DATRDY: Data Ready for Conversion (Code Label `DAC_DATRDY`)**
  0 = Data has been written to the Data Holding Register and not yet transferred to the Data Output Register.
  1 = The last data written in the Data Holding Register has been transferred to the Data Output Register. This is equal to 0 when the Timer Trigger is disabled or at reset. Enabling the Timer Trigger sets this bit to 1.

DAC Interrupt Enable Register

Register Name: DAC_IER
Access Type: Write-only
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
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DATRDY

- **DATRDY: Data Ready for Conversion Interrupt Enable (Code Label `DAC_DATRDY`)**
  0 = No effect.
  1 = Enables the Data Ready for Conversion Interrupt.

DAC Interrupt Disable Register

Register Name: DAC_IDR
Access Type: Write-only
Offset: 0x18

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
–	–	–	–	–	–	–	DATRDY

- **DATRDY: Data Ready for Conversion Interrupt Disable (Code Label `DAC_DATRDY`)**
  0 = No effect.
  1 = Disables the Data Ready for Conversion Interrupt.

DAC Interrupt Mask Register

Register Name: DAC_IMR
Access Type: Read-only
Reset State: 0
Offset: 0x1C

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
–	–	–	–	–	–	–	DATRDY

- **DATRDY: Data Ready for Conversion Interrupt Mask (Code Label `DAC_DATRDY`)**
  0 = Data Ready for Conversion Interrupt is disabled.
  1 = Data Ready for Conversion Interrupt is enabled.

JTAG Boundary-scan Register

The Boundary-scan Register (BSR) contains 256 bits which correspond to active pins and associated control signals.

Each AT91M55800A input pin has a corresponding bit in the Boundary-scan Register for observability.

Each AT91M55800A output pin has a corresponding 2-bit register in the BSR. The OUTPUT bit contains data which can be forced on the pad. The CTRL bit can put the pad into high impedance.

Each AT91M55800A in/out pin corresponds to a 3-bit register in the BSR. The OUTPUT bit contains data that can be forced on the pad. The INPUT bit is for the observability of data applied to the pad. The CTRL bit selects the direction of the pad.

Table 25. JTAG Boundary-scan Register

Bit Number	Pin Name	Pin Type	Associated BSR Cells
256	NWAIT	INPUT	INPUT
255	NRST	INPUT	INPUT
254	PB18/BMS	IN/OUT	OUTPUT
253			INPUT
252			CTRL
251	MCKO	OUTPUT	OUTPUT
250			CTRL
249	NWDOVF	OUTPUT	OUTPUT
248			CTRL
247	PB17	IN/OUT	OUTPUT
246			INPUT
245			CTRL
244	PB16	IN/OUT	OUTPUT
243			INPUT
242			CTRL
241	PB15	IN/OUT	OUTPUT
240			INPUT
239			CTRL
238	PB14	IN/OUT	OUTPUT
237			INPUT
236			CTRL
235	PB13	IN/OUT	OUTPUT
234			INPUT
233	PB13	IN/OUT	CTRL
232	PB12	IN/OUT	OUTPUT
231			INPUT
230			CTRL

Table 25. JTAG Boundary-scan Register (Continued)

Bit Number	Pin Name	Pin Type	Associated BSR Cells
229	PB11	IN/OUT	OUTPUT
228			INPUT
227			CTRL
226	PB10	IN/OUT	OUTPUT
225			INPUT
224			CTRL
223	PB9	IN/OUT	OUTPUT
222			INPUT
221			CTRL
220	PB8	IN/OUT	OUTPUT
219			INPUT
218			CTRL
217	PB7/AD1TRIG	IN/OUT	OUTPUT
216			INPUT
215			CTRL
214	PB6/AD0TRIG	IN/OUT	OUTPUT
213			INPUT
212			CTRL
211	PB5	IN/OUT	OUTPUT
210			INPUT
209			CTRL
208	PB4/IRQ5	IN/OUT	OUTPUT
207			INPUT
206			CTRL
205	PB3	IN/OUT	OUTPUT
204			INPUT
203			CTRL
202	PB2	IN/OUT	OUTPUT
201			INPUT
200			CTRL
199	PB1	IN/OUT	OUTPUT
198			INPUT
197			CTRL

Table 25. JTAG Boundary-scan Register (Continued)

Bit Number	Pin Name	Pin Type	Associated BSR Cells
196	PB0	IN/OUT	OUTPUT
195			INPUT
194			CTRL
193	NCS7	OUTPUT	OUTPUT
192	NCS6	OUTPUT	OUTPUT
191	NCS5	OUTPUT	OUTPUT
190	NCS4	OUTPUT	OUTPUT
189	PA29NPCS3	IN/OUT	OUTPUT
188			INPUT
187			CTRL
186	PA28NPCS2	IN/OUT	OUTPUT
185			INPUT
184			CTRL
183	PA27NPCS1	IN/OUT	OUTPUT
182			INPUT
181			CTRL
180	PA26NPCS0	IN/OUT	OUTPUT
179			INPUT
178			CTRL
177	PA25MOSI	IN/OUT	OUTPUT
176			INPUT
175			CTRL
174	PA24MISO	IN/OUT	OUTPUT
173			INPUT
172			CTRL
171	PA23SPCK	IN/OUT	OUTPUT
170			INPUT
169			CTRL
168	PA22RXD2	IN/OUT	OUTPUT
167			INPUT
166			CTRL
165	PA21TXD2	IN/OUT	OUTPUT
164	PA21TXD2	IN/OUT	INPUT
163			CTRL

Table 25. JTAG Boundary-scan Register (Continued)

Bit Number	Pin Name	Pin Type	Associated BSR Cells
162	PA20SCK2	IN/OUT	OUTPUT
161			INPUT
160			CTRL
159	PA19RXD1	IN/OUT	OUTPUT
158			INPUT
157			CTRL
156	PA18/TXD1/NTRI	IN/OUT	OUTPUT
155			INPUT
154			CTRL
153	PA17/SCK1	IN/OUT	OUTPUT
152			INPUT
151			CTRL
150	PA16/RXD0	IN/OUT	OUTPUT
149			INPUT
148			CTRL
147	PA15/TXD0	IN/OUT	OUTPUT
146			INPUT
145			CTRL
144	PA14/SCK0	IN/OUT	OUTPUT
143			INPUT
142			CTRL
141	PA13/FIQ	IN/OUT	OUTPUT
140			INPUT
139			CTRL
138	PA12/IRQ3	IN/OUT	OUTPUT
137			INPUT
136			CTRL
135	PA11/IRQ2	IN/OUT	OUTPUT
134			INPUT
133			CTRL
132	PA10/IRQ1	IN/OUT	OUTPUT
131			INPUT
130			CTRL

Table 25. 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

Table 25. JTAG Boundary-scan Register (Continued)

Bit Number	Pin Name	Pin Type	Associated BSR Cells
95			INPUT
94			CTRL
93	PB25/TCLK2	IN/OUT	OUTPUT
92			INPUT
91			CTRL
90	PB24/TIOB1	IN/OUT	OUTPUT
89			INPUT
88			CTRL
87	PB23/TIOA1	IN/OUT	OUTPUT
86			INPUT
85			CTRL
84	PB22/TCLK1	IN/OUT	OUTPUT
83			INPUT
82			CTRL
81	PB21TIOB0	IN/OUT	OUTPUT
80			INPUT
79			CTRL
78	PB20/TIOA0	IN/OUT	OUTPUT
77			INPUT
76			CTRL
75	PB19/TCLK0	IN/OUT	OUTPUT
74			INPUT
73			CTRL
72	D15	IN/OUT	INPUT
71			OUTPUT
70	D14	IN/OUT	INPUT
69			OUTPUT
68	D13	IN/OUT	INPUT
67			OUTPUT
66	D12	IN/OUT	INPUT
65			OUTPUT
64	D11	IN/OUT	INPUT
63			OUTPUT
62	D10	IN/OUT	INPUT
61			OUTPUT

Table 25. JTAG Boundary-scan Register (Continued)

Bit Number	Pin Name	Pin Type	Associated BSR Cells
60	D9	IN/OUT	INPUT
59			OUTPUT
58	D8	IN/OUT	INPUT
57			OUTPUT
56	D[15:8]	IN/OUT	CTRL
55	D7	IN/OUT	INPUT
54			OUTPUT
53	D6	IN/OUT	INPUT
52			OUTPUT
51	D5	IN/OUT	INPUT
50			OUTPUT
49	D4	IN/OUT	INPUT
48			OUTPUT
47	D3	IN/OUT	INPUT
46			OUTPUT
45	D2	IN/OUT	INPUT
44			OUTPUT
43	D1	IN/OUT	INPUT
42			OUTPUT
41	D0	IN/OUT	INPUT
40			OUTPUT
39	D[7:0]	IN/OUT	CTRL
38	A23	OUTPUT	OUTPUT
37	A22	OUTPUT	OUTPUT
36	A21	OUTPUT	OUTPUT
35	A20	OUTPUT	OUTPUT
34	A19	OUTPUT	OUTPUT
33	A18	OUTPUT	OUTPUT
32	A17	OUTPUT	OUTPUT
31	A16	OUTPUT	OUTPUT
30	A[23:16]	OUTPUT	CTRL
29	A15	OUTPUT	OUTPUT
28	A14	OUTPUT	OUTPUT
27	A13	OUTPUT	OUTPUT
26	A12	OUTPUT	OUTPUT

Table 25. JTAG Boundary-scan Register (Continued)

Bit Number	Pin Name	Pin Type	Associated BSR Cells
25	A11	OUTPUT	OUTPUT
24	A10	OUTPUT	OUTPUT
23	A9	OUTPUT	OUTPUT
22	A8	OUTPUT	OUTPUT
21	A[15:8]	OUTPUT	CTRL
20	A7	OUTPUT	OUTPUT
19	A6	OUTPUT	OUTPUT
18	A5	OUTPUT	OUTPUT
17	A4	OUTPUT	OUTPUT
16	A3	OUTPUT	OUTPUT
15	A2	OUTPUT	OUTPUT
14	A1	OUTPUT	OUTPUT
13	NLB/A0	OUTPUT	OUTPUT
12	A[7:0]	OUTPUT	CTRL
11	NCS3	OUTPUT	OUTPUT
10	NCS2	OUTPUT	OUTPUT
9	NCS1	OUTPUT	OUTPUT
8	NCS0	OUTPUT	OUTPUT
7	NUB/NWR1	IN/OUT	OUTPUT
6			INPUT
5	NUB/NWR0	IN/OUT	OUTPUT
4			INPUT
3	NOE/NRD	IN/OUT	OUTPUT
2			INPUT
1	NCS[7:0] NUB/NWR1 NWE/NWR0 NOE/NRD	IN/OUT	CTRL

Table of Contents

Atmel Headquarters

Corporate Headquarters
2325 Orchard Parkway
San Jose, CA 95131
TEL 1(408) 441-0311
FAX 1(408) 487-2600

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 Tsimhatsui
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
TEL 1(408) 441-0311
FAX 1(408) 436-4314

Microcontrollers
2325 Orchard Parkway
San Jose, CA 95131
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
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
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

e-mail
literature@atmel.com

Web Site
http://www.atmel.com

1745C–ATARM–12/02 0M