Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

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

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	ARM7®
Core Size	16/32-Bit
Speed	55MHz
Connectivity	CANbus, Ethernet, I²C, SPI, SSC, UART/USART, USB
Peripherals	Brown-out Detect/Reset, DMA, POR, PWM, WDT
Number of I/O	62
Program Memory Size	512KB (512K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	128K x 8
Voltage - Supply (Vcc/Vdd)	1.65V ~ 1.95V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	100-LQFP
Supplier Device Package	100-LQFP (14x14)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/at91sam7x512b-au

Features

- Incorporates the ARM7TDMI ARM Thumb® Processor
  - High-performance 32-bit RISC Architecture
  - High-density 16-bit Instruction Set
    - Leader in MIPS/Watt
  - EmbeddedICE™ In-circuit Emulation, Debug Communication Channel Support
- Internal High-speed Flash
  - 512 Kbytes (SAM7X512) Organized in Two Banks of 1024 Pages of 256 Bytes (Dual Plane)
  - 256 Kbytes (SAM7X256) Organized in 1024 Pages of 256 Bytes (Single Plane)
  - 128 Kbytes (SAM7X128) Organized in 512 Pages of 256 Bytes (Single Plane)
    - Single Cycle Access at Up to 30 MHz in Worst Case Conditions
    - Prefetch Buffer Optimizing Thumb Instruction Execution at Maximum Speed
    - Page Programming Time: 6 ms, Including Page Auto-erase, Full Erase Time: 15 ms
    - 10,000 Write Cycles, 10-year Data Retention Capability, Sector Lock Capabilities, Flash Security Bit
    - Fast Flash Programming Interface for High Volume Production
- Internal High-speed SRAM, Single-cycle Access at Maximum Speed
  - 128 Kbytes (SAM7X512)
  - 64 Kbytes (SAM7X256)
  - 32 Kbytes (SAM7X128)
- Memory Controller (MC)
  - Embedded Flash Controller, Abort Status and Misalignment Detection
- Reset Controller (RSTC)
  - Based on Power-on Reset Cells and Low-power Factory-calibrated Brownout Detector
  - Provides External Reset Signal Shaping and Reset Source Status
- Clock Generator (CKGR)
  - Low-power RC Oscillator, 3 to 20 MHz On-chip Oscillator and one PLL
- Power Management Controller (PMC)
  - Power Optimization Capabilities, Including Slow Clock Mode (Down to 500 Hz) and Idle Mode
  - Four Programmable External Clock Signals
- Advanced Interrupt Controller (AIC)
  - Individually Maskable, Eight-level Priority, Vectored Interrupt Sources
  - Two External Interrupt Sources and One Fast Interrupt Source, Spurious Interrupt Protected
- Debug Unit (DBGU)
  - 2-wire UART and Support for Debug Communication Channel interrupt, Programmable ICE Access Prevention
  - Mode for General Purpose 2-wire UART Serial Communication
- Periodic Interval Timer (PIT)
  - 20-bit Programmable Counter plus 12-bit Interval Counter
- Windowed Watchdog (WDT)
  - 12-bit key-protected Programmable Counter
  - Provides Reset or Interrupt Signals to the System
  - Counter May Be Stopped While the Processor is in Debug State or in Idle Mode
- Real-time Timer (RTT)
  - 32-bit Free-running Counter with Alarm
  - Runs Off the Internal RC Oscillator
- Two Parallel Input/Output Controllers (PIO)
  - Sixty-two Programmable I/O Lines Multiplexed with up to Two Peripheral I/Os
  - Input Change Interrupt Capability on Each I/O Line

11. ARM7TDMI Processor Overview

11.1 Overview

The ARM7TDMI core executes both the 32-bit ARM and 16-bit Thumb instruction sets, allowing the user to trade off between high performance and high code density.The ARM7TDMI processor implements Von Neuman architecture, using a three-stage pipeline consisting of Fetch, Decode, and Execute stages.

The main features of the ARM7tDMI processor are:

- ARM7TDMI Based on ARMv4T Architecture
- Two Instruction Sets
  - ARM High-performance 32-bit Instruction Set
  - Thumb High Code Density 16-bit Instruction Set
- Three-Stage Pipeline Architecture
  - Instruction Fetch (F)
  - Instruction Decode (D)
  - Execute (E)

15.4.2 Periodic Interval Timer Status Register

Register Name: PIT_SR

Access Type: Read-only

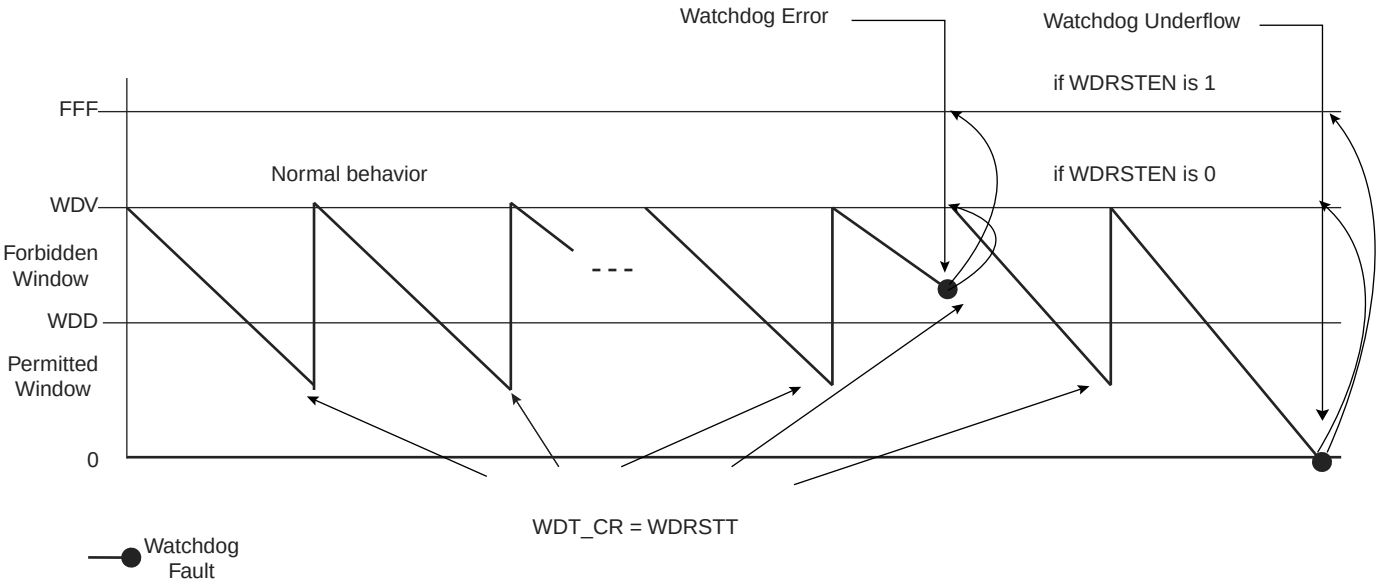
31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	PITS

- **PITS: Periodic Interval Timer Status**

0 = The Periodic Interval timer has not reached PIV since the last read of PIT_PIVR.

1 = The Periodic Interval timer has reached PIV since the last read of PIT_PIVR.

Figure 16-2. Watchdog Behavior



19.3.1 MC Flash Mode Register

Register Name: MC_FMR

Access Type: Read-write

Offset: (EFC0) 0x60

Offset: (EFC1) 0x70

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

23	22	21	20	19	18	17	16
FMCN							

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

7	6	5	4	3	2	1	0
NEBP	–	–	–	PROGE	LOCKE	–	FRDY

- **FRDY: Flash Ready Interrupt Enable**

0: Flash Ready does not generate an interrupt.

1: Flash Ready generates an interrupt.

- **LOCKE: Lock Error Interrupt Enable**

0: Lock Error does not generate an interrupt.

1: Lock Error generates an interrupt.

- **PROGE: Programming Error Interrupt Enable**

0: Programming Error does not generate an interrupt.

1: Programming Error generates an interrupt.

- **NEBP: No Erase Before Programming**

0: A page erase is performed before programming.

1: No erase is performed before programming.

- **FWS: Flash Wait State**

This field defines the number of wait states for read and write operations:

FWS	Read Operations	Write Operations
0	1 cycle	2 cycles
1	2 cycles	3 cycles
2	3 cycles	4 cycles
3	4 cycles	4 cycles

- Assert Erase during a period of more than 220 ms
- Power-off the chip

Then it is possible to return to FFPI mode and check that Flash is erased.

20.3.4.7 AT91SAM7X512 Select EFC Command

The commands WPx, EA, xLB, xFB are executed using the current EFC controller. The default EFC controller is EFC0. The **Select EFC** command (SEFC) allows selection of the current EFC controller.

Table 20-28. Select EFC Command

Step	Handshake Sequence	MODE[3:0]	DATA[15:0]
1	Write handshaking	CMDE	SEFC
2	Write handshaking	DATA	0 = Select EFC0 1 = Select EFC1

20.3.4.8 Memory Write Command

This command is used to perform a write access to any memory location.

The **Memory Write** command **(WRAM)** is optimized for consecutive writes. An internal address buffer is automatically increased.

Table 20-29. Write Command

Read/Write	DR Data
Write	(Number of Words to Write) << 16 | (WRAM)
Write	Address
Write	Memory [address]
Write	Memory [address+4]
Write	Memory [address+8]
Write	Memory [address+(Number of Words to Write - 1)* 4]

20.3.4.9 Get Version Command

The **Get Version** (GVE) command retrieves the version of the FFPI interface.

Table 20-30. Get Version Command

Read/Write	DR Data
Write	GVE
Read	Version

23.8.14 AIC Interrupt Set Command Register

Register Name: AIC_ISCR

Access Type: Write-only

31	30	29	28	27	26	25	24
PID31	PID30	PID29	PID28	PID27	PID26	PID25	PID24
23	22	21	20	19	18	17	16
PID23	PID22	PID21	PID20	PID19	PID18	PID17	PID16
15	14	13	12	11	10	9	8
PID15	PID14	PID13	PID12	PID11	PID10	PID9	PID8
7	6	5	4	3	2	1	0
PID7	PID6	PID5	PID4	PID3	PID2	SYS	FIQ

- **FIQ, SYS, PID2-PID31: Interrupt Set**

0 = No effect.

1 = Sets corresponding interrupt.

23.8.15 AIC End of Interrupt Command Register

Register Name: AIC_EOICR

Access Type: Write-only

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

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

27.6.7 PIO Controller Input Filter Enable Register

Name: PIO_IFER

Access Type: Write-only

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

- **P0-P31: Input Filter Enable**

0 = No effect.

1 = Enables the input glitch filter on the I/O line.

27.6.8 PIO Controller Input Filter Disable Register

Name: PIO_IFDR

Access Type: Write-only

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

- **P0-P31: Input Filter Disable**

0 = No effect.

1 = Disables the input glitch filter on the I/O line.

SPI_SR is set. As long as this flag is set, data is loaded in SPI_RDR. The user has to read the status register to clear the OVRES bit.

When a transfer starts, the data shifted out is the data present in the Shift Register. If no data has been written in the Transmit Data Register (SPI_TDR), the last data received is transferred. If no data has been received since the last reset, all bits are transmitted low, as the Shift Register resets at 0.

When a first data is written in SPI_TDR, it is transferred immediately in the Shift Register and the TDRE bit rises. If new data is written, it remains in SPI_TDR until a transfer occurs, i.e. NSS falls and there is a valid clock on the SPCK pin. When the transfer occurs, the last data written in SPI_TDR is transferred in the Shift Register and the TDRE bit rises. This enables frequent updates of critical variables with single transfers.

Then, new data is loaded in the Shift Register from the Transmit Data Register. In case no character is ready to be transmitted, i.e. no character has been written in SPI_TDR since the last load from SPI_TDR to the Shift Register, the Shift Register is not modified and the last received character is retransmitted.

Figure 28-9 shows a block diagram of the SPI when operating in Slave Mode.

Figure 28-9. Slave Mode Functional Block Diagram

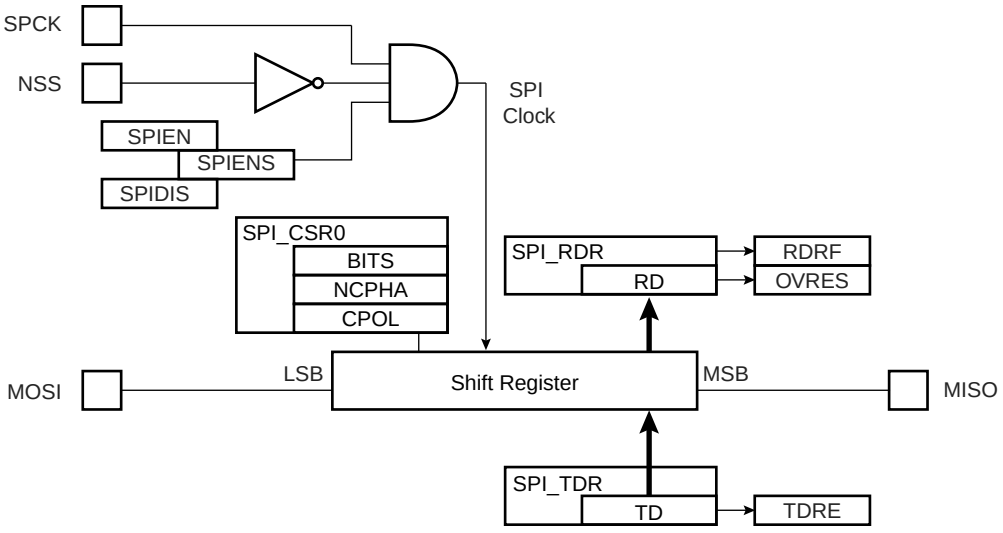
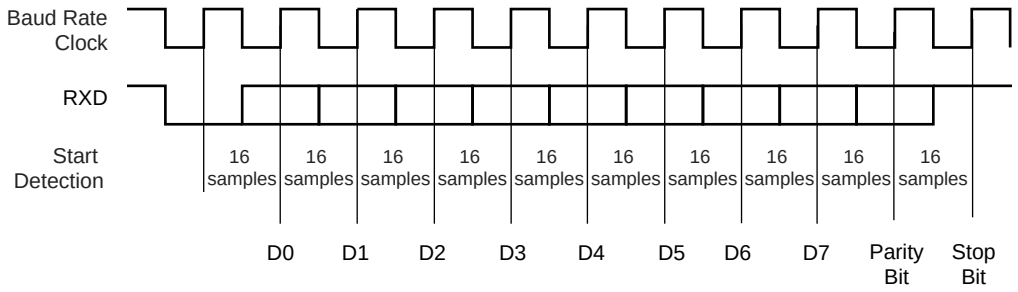


Figure 30-9. Asynchronous Character Reception

Example: 8-bit, Parity Enabled



30.6.3.3 Synchronous Receiver

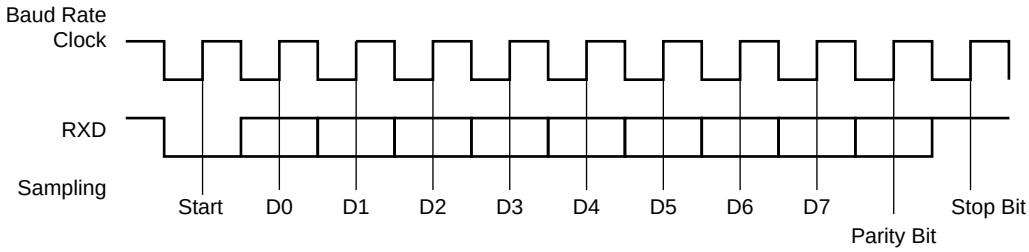
In synchronous mode (SYNC = 1), the receiver samples the RXD signal on each rising edge of the Baud Rate Clock. If a low level is detected, it is considered as a start. All data bits, the parity bit and the stop bits are sampled and the receiver waits for the next start bit. Synchronous mode operations provide a high speed transfer capability.

Configuration fields and bits are the same as in asynchronous mode.

Figure 30-10 illustrates a character reception in synchronous mode.

Figure 30-10. Synchronous Mode Character Reception

Example: 8-bit, Parity Enabled 1 Stop



30.6.3.4 Receiver Operations

When a character reception is completed, it is transferred to the Receive Holding Register (US_RHR) and the RXRDY bit in the Status Register (US_CSR) rises. If a character is completed while the RXRDY is set, the OVRE (Overrun Error) bit is set. The last character is transferred into US_RHR and overwrites the previous one. The OVRE bit is cleared by writing the Control Register (US_CR) with the RSTSTA (Reset Status) bit at 1.

30.7.7 USART Receive Holding Register

Name: US_RHR

Access Type: Read-only

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

- **RXCHR: Received Character**

Last character received if RXRDY is set.

- **RXSYNH: Received Sync**

0: Last Character received is a Data.

1: Last Character received is a Command.

32.6.3 TC Channel Control Register

Register Name: TC_CCR [x=0..2]

Access Type: Write-only

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

- **CLKEN: Counter Clock Enable Command**

0 = No effect.

1 = Enables the clock if CLKDIS is not 1.

- **CLKDIS: Counter Clock Disable Command**

0 = No effect.

1 = Disables the clock.

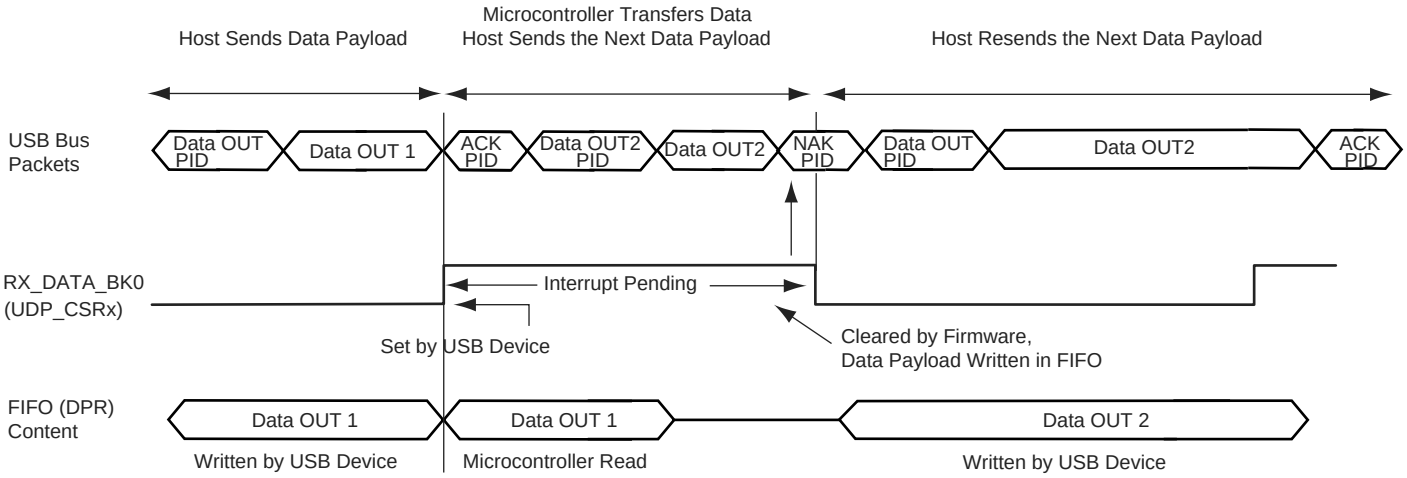
- **SWTRG: Software Trigger Command**

0 = No effect.

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

1. The host generates a Data OUT packet.
2. This packet is received by the USB device endpoint. While the FIFO associated to this endpoint is being used by the microcontroller, a NAK PID is returned to the host. Once the FIFO is available, data are written to the FIFO by the USB device and an ACK is automatically carried out to the host.
3. The microcontroller is notified that the USB device has received a data payload polling RX_DATA_BK0 in the endpoint's UDP_ CSRx register. An interrupt is pending for this endpoint while RX_DATA_BK0 is set.
4. The number of bytes available in the FIFO is made available by reading RXBYTECNT in the endpoint's UDP_ CSRx register.
5. The microcontroller carries out data received from the endpoint's memory to its memory. Data received is available by reading the endpoint's UDP_ FDRx register.
6. The microcontroller notifies the USB device that it has finished the transfer by clearing RX_DATA_BK0 in the endpoint's UDP_ CSRx register.
7. A new Data OUT packet can be accepted by the USB device.

Figure 34-9. Data OUT Transfer for Non Ping-pong Endpoints



An interrupt is pending while the flag RX_DATA_BK0 is set. Memory transfer between the USB device, the FIFO and microcontroller memory can not be done after RX_DATA_BK0 has been cleared. Otherwise, the USB device would accept the next Data OUT transfer and overwrite the current Data OUT packet in the FIFO.

Using Endpoints With Ping-pong Attributes

During isochronous transfer, using an endpoint with ping-pong attributes is obligatory. To be able to guarantee a constant bandwidth, the microcontroller must read the previous data payload sent by the host, while the current data payload is received by the USB device. Thus two banks of memory are used. While one is available for the microcontroller, the other one is locked by the USB device.

34.6.7 UDP Interrupt Status Register

Register Name: UDP_ ISR

Access Type: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	WAKEUP	ENDBUSRES	SOFINT	–	RXRSM	RXSUSP
7	6	5	4	3	2	1	0
		EP5INT	EP4INT	EP3INT	EP2INT	EP1INT	EP0INT

- **EP0INT: Endpoint 0 Interrupt Status**
- **EP1INT: Endpoint 1 Interrupt Status**
- **EP2INT: Endpoint 2 Interrupt Status**
- **EP3INT: Endpoint 3 Interrupt Status**
- **EP4INT: Endpoint 4 Interrupt Status**
- **EP5INT: Endpoint 5 Interrupt Status**

0 = No Endpoint0 Interrupt pending.

1 = Endpoint0 Interrupt has been raised.

Several signals can generate this interrupt. The reason can be found by reading UDP_ CSR0:

RXSETUP set to 1

RX_DATA_BK0 set to 1

RX_DATA_BK1 set to 1

TXCOMP set to 1

STALLSENT set to 1

EP0INT is a sticky bit. Interrupt remains valid until EP0INT is cleared by writing in the corresponding UDP_ CSR0 bit.

- **RXSUSP: UDP Suspend Interrupt Status**

0 = No UDP Suspend Interrupt pending.

1 = UDP Suspend Interrupt has been raised.

The USB device sets this bit when it detects no activity for 3ms. The USB device enters Suspend mode.

- **RXRSM: UDP Resume Interrupt Status**

0 = No UDP Resume Interrupt pending.

1 =UDP Resume Interrupt has been raised.

The USB device sets this bit when a UDP resume signal is detected at its port.

After reset, the state of this bit is undefined, the application must clear this bit by setting the RXRSM flag in the UDP_ ICR register.

36.8 Controller Area Network (CAN) User Interface

Table 36-2. Register Mapping

Offset	Register	Name	Access	Reset
0x0000	Mode Register	CAN_MR	Read-write	0x0
0x0004	Interrupt Enable Register	CAN_IER	Write-only	-
0x0008	Interrupt Disable Register	CAN_IDR	Write-only	-
0x000C	Interrupt Mask Register	CAN_IMR	Read-only	0x0
0x0010	Status Register	CAN_SR	Read-only	0x0
0x0014	Baudrate Register	CAN_BR	Read-write	0x0
0x0018	Timer Register	CAN_TIM	Read-only	0x0
0x001C	Timestamp Register	CAN_TIMESTP	Read-only	0x0
0x0020	Error Counter Register	CAN_ECR	Read-only	0x0
0x0024	Transfer Command Register	CAN_TCR	Write-only	-
0x0028	Abort Command Register	CAN_ACR	Write-only	-
0x0100 - 0x01FC	Reserved	–	–	–
0x0200	Mailbox 0 Mode Register	CAN_MMR0	Read-write	0x0
0x0204	Mailbox 0 Acceptance Mask Register	CAN_MAM0	Read-write	0x0
0x0208	Mailbox 0 ID Register	CAN_MID0	Read-write	0x0
0x020C	Mailbox 0 Family ID Register	CAN_MFID0	Read-only	0x0
0x0210	Mailbox 0 Status Register	CAN_MSR0	Read-only	0x0
0x0214	Mailbox 0 Data Low Register	CAN_MDL0	Read-write	0x0
0x0218	Mailbox 0 Data High Register	CAN_MDH0	Read-write	0x0
0x021C	Mailbox 0 Control Register	CAN_MCR0	Write-only	-
0x0220	Mailbox 1 Mode Register	CAN_MMR1	Read-write	0x0
0x0224	Mailbox 1 Acceptance Mask Register	CAN_MAM1	Read-write	0x0
0x0228	Mailbox 1 ID register	CAN_MID1	Read-write	0x0
0x022C	Mailbox 1 Family ID Register	CAN_MFID1	Read-only	0x0
0x0230	Mailbox 1 Status Register	CAN_MSR1	Read-only	0x0
0x0234	Mailbox 1 Data Low Register	CAN_MDL1	Read-write	0x0
0x0238	Mailbox 1 Data High Register	CAN_MDH1	Read-write	0x0
0x023C	Mailbox 1 Control Register	CAN_MCR1	Write-only	-
...	...	...	...	-

This flag is set depending on TEC counter value. A node is bus off when TEC counter is greater or equal to 256 (decimal).

This flag is automatically reset when above condition is not satisfied. Refer to Section 36.6.4.6 “Error Interrupt Handler” on page 492 for more information.

- **SLEEP: CAN controller in Low power Mode**

0 = CAN controller is not in low power mode.

1 = CAN controller is in low power mode.

This flag is automatically reset when Low power mode is disabled

- **WAKEUP: CAN controller is not in Low power Mode**

0 = CAN controller is in low power mode.

1 = CAN controller is not in low power mode.

When a WAKEUP event occurs, the CAN controller is synchronized with the bus activity. Messages can be transmitted or received. The CAN controller clock must be available when a WAKEUP event occurs. This flag is automatically reset when the CAN Controller enters Low Power mode.

- **TOVF: Timer Overflow**

0 = The timer has not rolled-over FFFFh to 0000h.

1 = The timer rolls-over FFFFh to 0000h.

This flag is automatically cleared by reading CAN_SR register.

- **TSTP Timestamp**

0 = No bus activity has been detected.

1 = A start of frame or an end of frame has been detected (according to the TEOF field in the CAN_MR register).

This flag is automatically cleared by reading the CAN_SR register.

- **CERR: Mailbox CRC Error**

0 = No CRC error occurred during a previous transfer.

1 = A CRC error occurred during a previous transfer.

A CRC error has been detected during last reception.

This flag is automatically cleared by reading CAN_SR register.

- **SERR: Mailbox Stuffing Error**

0 = No stuffing error occurred during a previous transfer.

1 = A stuffing error occurred during a previous transfer.

A form error results from the detection of more than five consecutive bit with the same polarity.

This flag is automatically cleared by reading CAN_SR register.

- **AERR: Acknowledgment Error**

0 = No acknowledgment error occurred during a previous transfer.

1 = An acknowledgment error occurred during a previous transfer.

An acknowledgment error is detected when no detection of the dominant bit in the acknowledge slot occurs.

This flag is automatically cleared by reading CAN_SR register.

- **FERR: Form Error**

0 = No form error occurred during a previous transfer

1 = A form error occurred during a previous transfer

36.8.16 CAN Message Status Register

Name: CAN_MSRx

Access Type: Read-only

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

23	22	21	20	19	18	17	16
MRDY	MABT	–	MRTR	MDLC			

15	14	13	12	11	10	9	8
MTIMESTAMP15	MTIMESTAMP14	MTIMESTAMP13	MTIMESTAMP12	MTIMESTAMP11	MTIMESTAMP10	MTIMESTAMP9	MTIMESTAMP8

7	6	5	4	3	2	1	0
MTIMESTAMP7	MTIMESTAMP6	MTIMESTAMP5	MTIMESTAMP4	MTIMESTAMP3	MTIMESTAMP2	MTIMESTAMP1	MTIMESTAMP0

These register fields are updated each time a message transfer is received or aborted.

MMI is cleared by reading the CAN_MSRx register.

MRDY, MABT are cleared by writing MTCR or MACR in the CAN_MCRx register.

Warning: MRTR and MDLC state depends partly on the mailbox object type.

- **MTIMESTAMP: Timer value**

This field is updated only when time-triggered operations are disabled (TTM cleared in CAN_MR register). If the TEOF field in the CAN_MR register is cleared, TIMESTAMP is the internal timer value at the start of frame of the last message received or sent by the mailbox. If the TEOF field in the CAN_MR register is set, TIMESTAMP is the internal timer value at the end of frame of the last message received or sent by the mailbox.

In Time Triggered Mode, MTIMESTAMP is set to 0.

- **MDLC: Mailbox Data Length Code**

Mailbox Object Type	Description
Receive	Length of the first mailbox message received
Receive with overwrite	Length of the last mailbox message received
Transmit	No action
Consumer	Length of the mailbox message received
Producer	Length of the mailbox message to be sent after the remote frame reception

41.3.9 Two-wire Interface (TWI)

41.3.9.1 TWI: Clock Divider

The value of CLDIV x 2^{CKDIV} must be less than or equal to 8191, the value of CHDIV x 2^{CKDIV} must be less than or equal to 8191·

Problem Fix/Workaround

None.

41.3.9.2 TWI: Disabling Does not Operate Correctly

Any transfer in progress is immediately frozen if the Control Register (TWI_CR) is written with the bit MSDIS at 1. Furthermore, the status bits TXCOMP and TXRDY in the Status Register (TWI_SR) are not reset.

Problem Fix/Workaround

The user must wait for the end of transfer before disabling the TWI. In addition, the interrupts must be disabled before disabling the TWI.

41.3.9.3 TWI: NACK Status Bit Lost

During a master frame, if TWI_SR is read between the Non Acknowledge condition detection and the TXCOMP bit rising in the TWI_SR, the NACK bit is not set.

Problem Fix/Workaround

The user must wait for the TXCOMP status bit by interrupt and must not read the TWI_SR as long as transmission is not completed.

TXCOMP and NACK fields are set simultaneously and the NACK field is reset after the read of the TWI_SR.

41.3.9.4 TWI: Possible Receive Holding Register Corruption

When loading the TWI_RHR, the transfer direction is ignored. The last data byte received in the TWI_RHR is corrupted at the end of the first subsequent transmit data byte. Neither RXRDY nor OVERRUN status bits are set if this occurs.

Problem Fix/Workaround

The user must be sure that received data is read before transmitting any new data.

41.6 AT91SAM7X512 Errata - Rev. B Parts

Refer to Section 41.1 “Marking” on page 607.

Note: AT91SAM7X512 Revision B chip ID is 0x0x275C 0A41.

41.6.1 Analog-to-Digital Converter (ADC)

41.6.1.1 ADC: DRDY Bit Cleared

The DRDY Flag should be clear only after a read of ADC_LCDR (Last Converted Data Register). A read of any ADC_CDRx register (Channel Data Register) automatically clears the DRDY flag.

Problem Fix/Workaround:

None

41.6.1.2 ADC: DRDY not Cleared on Disable

When reading LCDR at the same instant as an end of conversion, with DRDY already active, DRDY is kept active regardless of the enable status of the current channel. This sets DRDY, whereas new data is not stored.

Problem Fix/Workaround

None

41.6.1.3 ADC: DRDY Possibly Skipped due to CDR Read

Reading CDR for channel "y" at the same instant as an end of conversion on channel "x" with EOC[x] already active, leads to skipping to set the DRDY flag if channel "x" is enabled.

Problem Fix/Workaround

Use of DRDY functionality with access to CDR registers should be avoided.

41.6.1.4 ADC: Possible Skip on DRDY when Disabling a Channel

DRDY does not rise when disabling channel "y" at the same time as an end of "x" channel conversion, although data is stored into CDRx and LCDR.

Problem Fix/Workaround

None.

41.6.1.5 ADC: GOVRE Bit is not Updated

Read of the Status Register at the same instant as an end of conversion leads to skipping the update of the GOVRE (general overrun) flag. GOVRE is neither reset nor set.

For example, if reading the status while an end of conversion is occurring and:

1. GOVRE is active but DRDY is inactive, does not correspond to a new general overrun condition but the GOVRE flag is not reset.
2. GOVRE is inactive but DRDY is active, does correspond to a new general overrun condition but the GOVRE flag is not set.

Problem Fix/Workaround

None

41.6.1.6 ADC: GOVRE Bit is not Set when Reading CDR

When reading CDRy (Channel Data Register y) at the same instant as an end of conversion on channel "x" with the following conditions:

- EOC[x] already active,

Version 6120F (Continued)	Comments	Change Request Ref.
	DBGU: Corrected references from ice_nreset to Power-on Reset in Figure 26-1 on page 196, Functional Block Diagram, and in FNTRST bit description in Section 26.5.12 “Debug Unit Force NTRST Register” on page 218.	2832
	PIO:Section 27.4.4 “Output Control” on page 222, typo corrected	05-346
	Section 27.4.1 “Pull-up Resistor Control” on page 221 reference to resistor value removed.	05-497
	Figure 27-3 on page 221 0 and 1 inverted in the MUX controlled by PIO_MDSR..	3053
	SPI: Section 28.7.5 “SPI Status Register” on page 262 SPI_RCR, SPI_RNCR, SPI_TCR, SPI_TNCR location defined.	04-183
	Section 28.7.4 “SPI Transmit Data Register” on page 261, LASTXFER: Last Transfer text added.	05-434
	Section 28.7.2 “SPI Mode Register” on page 258, PCSDEC: Chip Select Decode changed.	05-476
	Updated Figure 28-1, ”Block Diagram” on page 244, removed Note. Removed bit FDIV from Section 28.7.2 “SPI Mode Register” on page 258 and Section 28.7.9 “SPI Chip Select Register” on page 267. LLB description modified in Section 28.7.2 “SPI Mode Register” on page 258.	05-484
	Updated Figure 28-9, ”Slave Mode Functional Block Diagram” on page 255 to remove FLOAD.	1542
	Updated information on SPI_RDR in Section 28.6.3 “Master Mode Operations” on page 249. Added information to SWRST bit description in Section 28.7.1 “SPI Control Register” on page 257. Corrected equations in DLYBCT bit description, Section 28.7.9 ”SPI Chip Select Register” on page 268.	1543
	Changes to Section 28.6.3.8 “Mode Fault Detection” on page 254.	1676
	USART:	2768
	Manchester Functionality Removed.	
	Section 30.4 “I/O Lines Description” on page 293, text concerning TXD line added.	
	Section 30.6.1.3 “Fractional Baud Rate in Asynchronous Mode” on page 297, using USART “functional mode” changed to USART “normal mode”.	
	Table 30-3, “Binary and Decimal Values for Di,” on page 298 and Table 30-4, “Binary and Decimal Values for Fi,” on page 299: DI and Fi properly referenced in titles.	
	Figure 30-25, ”IrDA Demodulator Operations” on page 314 modified.	
	Section 30.6.4.1 “ISO7816 Mode Overview” on page 310 clarification of PAR configuration added.	1552
	Section 30.6.7 “Modem Mode” on page 316 Control of DTR and RTS output pins.	1770
	Table 30-2, “Baud Rate Example (OVER = 0),” on page 296 60k and 70k MHz clock speeds removed.	2942
	“Asynchronous Receiver” on page 301 2nd line in 4th paragraph changed.	3023
	“Receiver Time-out” on page 306 list of user options rewritten.	
	Section 30.7.1 ”USART Control Register” STTTO bit function related to TIMEOUT in US_CSR register	
	Section 30.7.6 ”USART Channel Status Register” TIMEOUT bit function related to STTO in US_CR register	
	TC: Section 32.5.12 “External Event/Trigger Conditions” on page 388 “....(EEVT = 0), TIOB is no longer used as an output and the compare register B is not used to generate waveforms and subsequently no IRQs. Note (1) attached to ”EEVT: External Event Selection” in Section 32.6.5 “TC Channel Mode Register: Waveform Mode” on page 395 further clarifies this condition.	2704
	PWM: Section 33.5.3.3 ”Changing the Duty Cycle or the Period”: updated info on waveform generation.	1677