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	ARM® Cortex®-M4
Core Size	32-Bit Single-Core
Speed	120MHz
Connectivity	EBI/EMI, I²C, IrDA, Memory Card, SPI, SSC, UART/USART, USB
Peripherals	Brown-out Detect/Reset, DMA, POR, PWM, WDT
Number of I/O	79
Program Memory Size	512KB (512K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	128K x 8
Voltage - Supply (Vcc/Vdd)	1.62V ~ 3.6V
Data Converters	A/D 16x12b; D/A 2x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 105°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/atsam4s8cb-anr

12.6.5.11 SHASX and SHSAX

Signed Halving Add and Subtract with Exchange and Signed Halving Subtract and Add with Exchange.

Syntax

```
op{cond} {Rd}, Rn, Rm
```

where:

op is any of:

SHASX Add and Subtract with Exchange and Halving.

SHSAX Subtract and Add with Exchange and Halving.

cond is an optional condition code, see “Conditional Execution” .

Rd is the destination register.

Rn, Rm are registers holding the first and second operands.

Operation

The SHASX instruction:

1. Adds the top halfword of the first operand with the bottom halfword of the second operand.
2. Writes the halfword result of the addition to the top halfword of the destination register, shifted by one bit to the right causing a divide by two, or halving.
3. Subtracts the top halfword of the second operand from the bottom highword of the first operand.
4. Writes the halfword result of the division in the bottom halfword of the destination register, shifted by one bit to the right causing a divide by two, or halving.

The SHSAX instruction:

1. Subtracts the bottom halfword of the second operand from the top highword of the first operand.
2. Writes the halfword result of the addition to the bottom halfword of the destination register, shifted by one bit to the right causing a divide by two, or halving.
3. Adds the bottom halfword of the first operand with the top halfword of the second operand.
4. Writes the halfword result of the division in the top halfword of the destination register, shifted by one bit to the right causing a divide by two, or halving.

Restrictions

Do not use SP and do not use PC.

Condition Flags

These instructions do not affect the condition code flags.

12.6.8 Packing and Unpacking Instructions

The table below shows the instructions that operate on packing and unpacking data.

Table 12-23. Packing and Unpacking Instructions

Mnemonic	Description
PKH	Pack Halfword
SXTAB	Extend 8 bits to 32 and add
SXTAB16	Dual extend 8 bits to 16 and add
SXTAH	Extend 16 bits to 32 and add
SXTB	Sign extend a byte
SXTB16	Dual extend 8 bits to 16 and add
SXTH	Sign extend a halfword
UXTAB	Extend 8 bits to 32 and add
UXTAB16	Dual extend 8 bits to 16 and add
UXTAH	Extend 16 bits to 32 and add
UXTB	Zero extend a byte
UXTB16	Dual zero extend 8 bits to 16 and add
UXTH	Zero extend a halfword

Examples

```
        ADR.W  R0, BranchTable_Byte
        TBB    [R0, R1]     ; R1 is the index, R0 is the base address of the
                            ; branch table
    Case1
    ; an instruction sequence follows
    Case2
    ; an instruction sequence follows
    Case3
    ; an instruction sequence follows
    BranchTable_Byte
        DCB    0                  ; Case1 offset calculation
        DCB    ((Case2-Case1)/2)  ; Case2 offset calculation
        DCB    ((Case3-Case1)/2)  ; Case3 offset calculation


        TBH    [PC, R1, LSL #1]   ; R1 is the index, PC is used as base of the
                                  ; branch table
    BranchTable_H
        DCI    ((CaseA - BranchTable_H)/2)  ; CaseA offset calculation
        DCI    ((CaseB - BranchTable_H)/2)  ; CaseB offset calculation
        DCI    ((CaseC - BranchTable_H)/2)  ; CaseC offset calculation

    CaseA
    ; an instruction sequence follows
    CaseB
    ; an instruction sequence follows
    CaseC
    ; an instruction sequence follows
```

19.3.1 General Purpose Backup Register x

Name: SYS_GPBRx

Address: 0x400E1490

Access: Read/Write

31	30	29	28	27	26	25	24
GPBR_VALUE							

23	22	21	20	19	18	17	16
GPBR_VALUE							

15	14	13	12	11	10	9	8
GPBR_VALUE							

7	6	5	4	3	2	1	0
GPBR_VALUE							

These registers are reset at first power-up and on each loss of VVDIO.

- **GPBR_VALUE: Value of GPBR x**

If a Tamper event has been detected, it is not possible to write GPBR_VALUE as long as the LPDBCS0 or LPDBCS1 flag has not been cleared in the Supply Controller Status Register (SUPC_SR).

23.7.14 CRCCU Interrupt Mask Register

Name: CRCCU_IMR

Address: 0x40044048

Access: Read-only

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

- **ERRIMR: CRC Error Interrupt Mask**

0: Interrupt disabled

1: Interrupt enabled

28.4 Slow Clock

The Supply Controller embeds a slow clock generator that is supplied with the VDDIO power supply. As soon as VDDIO is supplied, both the crystal oscillator and the embedded RC oscillator are powered up, but only the embedded RC oscillator is enabled. This allows the slow clock to be valid in a short time (about 100 µs).

The slow clock is generated either by the slow clock crystal oscillator or by the slow clock RC oscillator.

The selection between the RC or the crystal oscillator is made by writing the XTALSEL bit in the Supply Controller Control Register (SUPC_CR).

28.4.1 Slow Clock RC Oscillator

By default, the slow clock RC oscillator is enabled and selected. The user has to take into account the possible drifts of the RC oscillator. More details are given in the section “DC Characteristics” of the product datasheet.

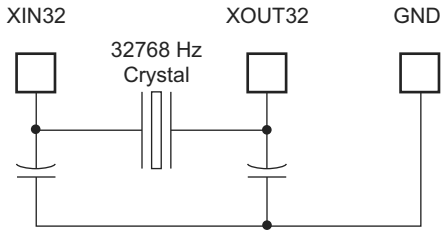
It can be disabled via the XTALSEL bit in SUPC_CR.

28.4.2 Slow Clock Crystal Oscillator

The Clock Generator integrates a 32768 Hz low-power oscillator. To use this oscillator, the XIN32 and XOUT32 pins must be connected to a 32768 Hz crystal. Two external capacitors must be wired as shown in Figure 28-2. More details are given in the section “DC Characteristics” of the product datasheet.

Note that the user is not obliged to use the slow clock crystal and can use the RC oscillator instead.

Figure 28-2. Typical Slow Clock Crystal Oscillator Connection



The user can select the crystal oscillator to be the source of the slow clock, as it provides a more accurate frequency. The command is made by writing a 1 to the SUPC_CR.XTALSEL bit. This results in a sequence which first configures the PIO lines multiplexed with XIN32 and XOUT32 to be driven by the oscillator, then enables the crystal oscillator and then disables the RC oscillator to save power. The switch of the slow clock source is glitch free. The OSCSEL bit of the Supply Controller Status Register (SUPC_SR) or the OSCSEL bit of the PMC Status Register (PMC_SR) tracks the oscillator frequency downstream. It must be read in order to be informed when the switch sequence, initiated when a new value is written in the SUPC_CR.XTALSEL bit, is done.

Coming back on the RC oscillator is only possible by shutting down the VDDIO power supply. If the user does not need the crystal oscillator, the XIN32 and XOUT32 pins can be left unconnected since by default the XIN32 and XOUT32 system I/O pins are in PIO input mode with pull-up after reset.

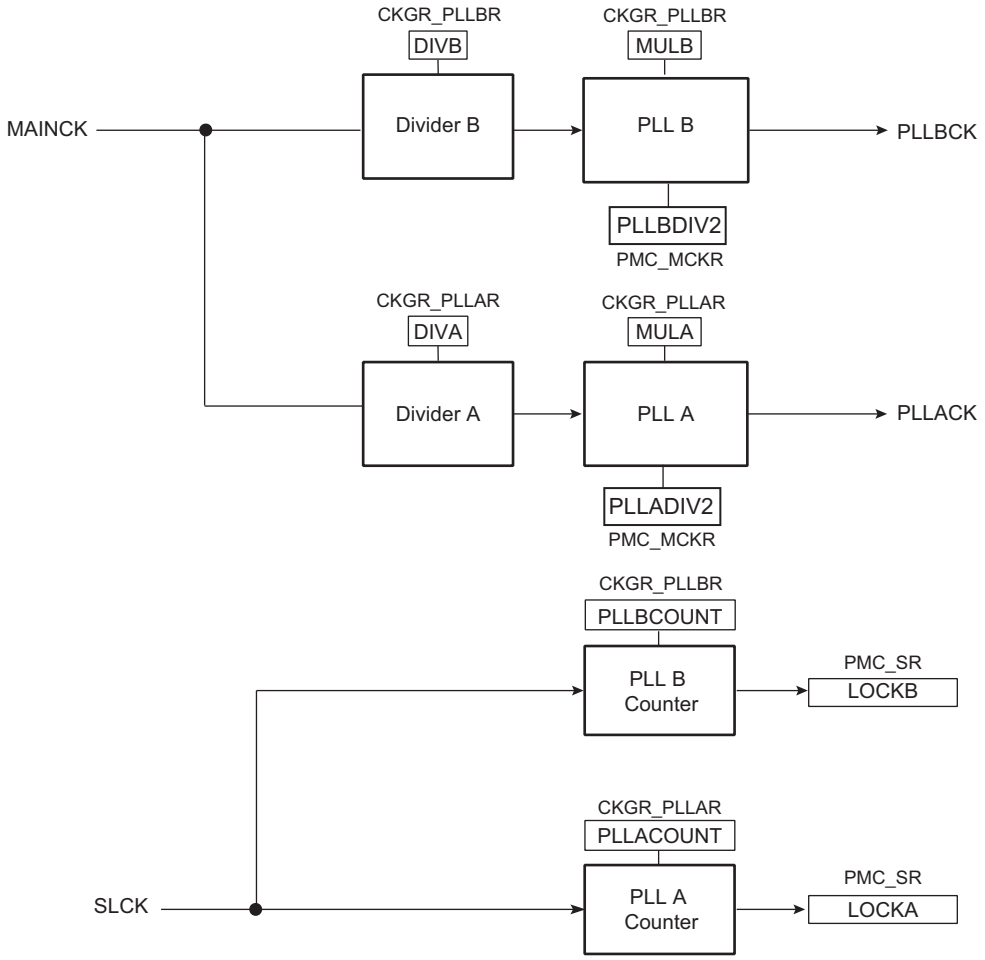
The user can also set the crystal oscillator in Bypass mode instead of connecting a crystal. In this case, the user has to provide the external clock signal on XIN32. The input characteristics of the XIN32 pin are given in the product electrical characteristics section. In order to set the Bypass mode, the OSCBYPASS bit of the Supply Controller Mode Register (SUPC_MR) needs to be set at 1 prior to writing a 1 in bit XTALSEL.

28.6 Divider and PLL Block

The device features two divider blocks and two PLL blocks that permit a wide range of frequencies to be selected on either the master clock, the processor clock or the programmable clock outputs. A 48 MHz clock signal is provided to the embedded USB device port regardless of the frequency of the main clock.

Figure 28-4 shows the block diagram of the divider and PLL blocks.

Figure 28-4. Dividers and PLL Block Diagram



28.6.1 Divider and Phase Lock Loop Programming

The divider can be set between 1 and 255 in steps of 1. When a divider field (DIV) is cleared, the output of the corresponding divider and the PLL output is a continuous signal at level 0. On reset, each DIV field is cleared, thus the corresponding PLL input clock is stuck at 0.

The PLLs (PLLA, PLLB) allow multiplication of the divider's outputs. The PLL clock signal has a frequency that depends on the respective source signal frequency and on the parameters DIV (DIVA, DIVB) and MUL (MULA, MULB) . The factor applied to the source signal frequency is (MUL + 1)/DIV. When MUL is written to 0 or DIV = 0, the PLL is disabled and its power consumption is saved. Note that there is a delay of two SLCK clock cycles between the disable command and the real disable of the PLL. Re-enabling the PLL can be performed by writing a value higher than 0 in the MUL field and DIV higher than 0.

Whenever the PLL is re-enabled or one of its parameters is changed, the LOCK (LOCKA, LOCKB) bit in PMC_SR is automatically cleared. The values written in the PLLCOUNT field (PLLACOUNT, PLLBCOUNT) in CKGR_PLLR (CKGR_PLLAR, CKGR_PLLBR) are loaded in the PLL counter. The PLL counter then decrements at the speed of

29.15 Clock Switching Details

29.15.1 Master Clock Switching Timings

Table 29-1 and Table 29-2 give the worst case timings required for the master clock to switch from one selected clock to another one. This is in the event that the prescaler is de-activated. When the prescaler is activated, an additional time of 64 clock cycles of the newly selected clock has to be added.

Table 29-1. Clock Switching Timings (Worst Case)

From To	Main Clock	SLCK	PLL Clock
Main Clock	–	4 x SLCK + 2.5 x Main Clock	3 x PLL Clock + 4 x SLCK + 1 x Main Clock
SLCK	0.5 x Main Clock + 4.5 x SLCK	–	3 x PLL Clock + 5 x SLCK
PLL Clock	0.5 x Main Clock + 4 x SLCK + PLLCOUNT x SLCK + 2.5 x PLLx Clock	2.5 x PLL Clock + 5 x SLCK + PLLCOUNT x SLCK	2.5 x PLL Clock + 4 x SLCK + PLLCOUNT x SLCK

Notes: 1. PLL designates either the PLLA or the PLLB Clock.
2. PLLCOUNT designates either PLLACOUNT or PLLBCOUNT.

Table 29-2. Clock Switching Timings between Two PLLs (Worst Case)

From To	PLLA Clock	PLLB Clock
PLLA Clock	2.5 x PLLA Clock + 4 x SLCK + PLLACOUNT x SLCK	3 x PLLA Clock + 4 x SLCK + 1.5 x PLLA Clock
PLLB Clock	3 x PLLB Clock + 4 x SLCK + 1.5 x PLLB Clock	2.5 x PLLB Clock + 4 x SLCK + PLLBCOUNT x SLCK

- **PLLBDIV2: PLLB Divisor by 2**

PLLBDIV2	PLLB Clock Division
0	PLLB clock frequency is divided by 1.
1	PLLB clock frequency is divided by 2.

32.9.17 SSC Write Protection Mode Register

Name: SSC_WPMR

Address: 0x400040E4

Access: Read/Write

31	30	29	28	27	26	25	24
WPKEY							

23	22	21	20	19	18	17	16
WPKEY							

15	14	13	12	11	10	9	8
WPKEY							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protection Enable**

0: Disables the write protection if WPKEY corresponds to 0x535343 (“SSC” in ASCII).

1: Enables the write protection if WPKEY corresponds to 0x535343 (“SSC” in ASCII).

See Section 32.8.10 “Register Write Protection” for the list of registers that can be protected.

- **WPKEY: Write Protection Key**

Value	Name	Description
0x535343	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0.

33.8.4 SPI Transmit Data Register

Name: SPI_TDR

Address: 0x4000800C

Access: Write-only

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

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

15	14	13	12	11	10	9	8
TD							

7	6	5	4	3	2	1	0
TD							

- **TD: Transmit Data**

Data to be transmitted by the SPI 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).

If SPI_MR.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 SPI_MR.PCSDEC = 1:

NPCS[3:0] output signals = PCS.

- **LASTXFER: Last Transfer**

0: No effect

1: The current NPCS is de-asserted after the transfer of the character written in TD. When SPI_CSRx.CSAAT is set, the communication with the current serial peripheral can be closed by raising the corresponding NPCS line as soon as TD transfer is completed.

This field is only used if variable peripheral select is active (SPI_MR.PS = 1).

35.6.5 UART Interrupt Mask Register

Name: UART_IMR

Address: 0x400E0610 (0), 0x400E0810 (1)

Access: Read-only

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

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

- **RXRDY: Mask RXRDY Interrupt**
- **TXRDY: Disable TXRDY Interrupt**
- **ENDRX: Mask End of Receive Transfer Interrupt**
- **ENDTX: Mask End of Transmit Interrupt**
- **OVRE: Mask Overrun Error Interrupt**
- **FRAME: Mask Framing Error Interrupt**
- **PARE: Mask Parity Error Interrupt**
- **TXEMPTY: Mask TXEMPTY Interrupt**
- **TXBUFE: Mask TXBUFE Interrupt**
- **RXBUFF: Mask RXBUFF Interrupt**

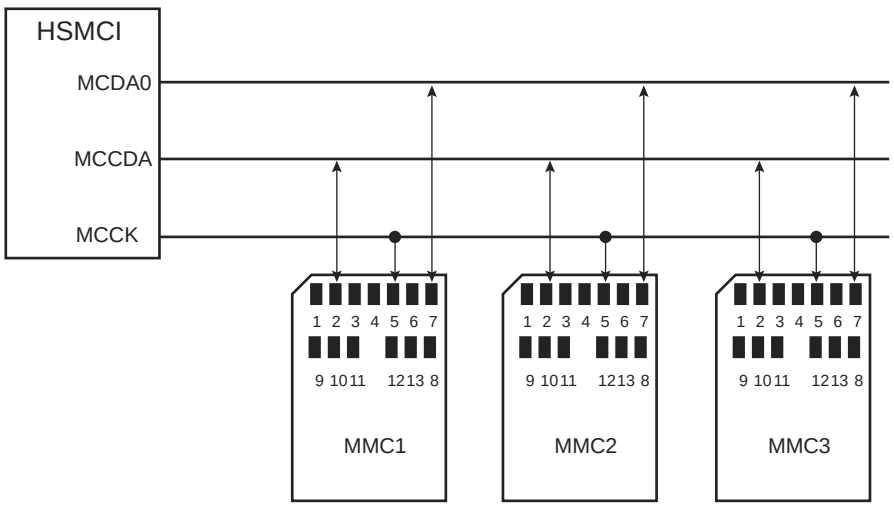
The High Speed MultiMedia Card communication is based on a 13-pin serial bus interface. It has three communication lines and four supply lines.

Table 38-4. Bus Topology

Pin Number	Name	Type[1]	Description	HSMCI Pin Name[2] (Slot z)
1	DAT[3]	I/O/PP	Data	MCDz3
2	CMD	I/O/PP/OD	Command/response	MCCDz
3	VSS1	S	Supply voltage ground	VSS
4	VDD	S	Supply voltage	VDD
5	CLK	I/O	Clock	MCCK
6	VSS2	S	Supply voltage ground	VSS
7	DAT[0]	I/O/PP	Data 0	MCDz0
8	DAT[1]	I/O/PP	Data 1	MCDz1
9	DAT[2]	I/O/PP	Data 2	MCDz2
10	DAT[4]	I/O/PP	Data 4	MCDz4
11	DAT[5]	I/O/PP	Data 5	MCDz5
12	DAT[6]	I/O/PP	Data 6	MCDz6
13	DAT[7]	I/O/PP	Data 7	MCDz7

Notes: 1. I: Input, O: Output, PP: Push/Pull, OD: Open Drain.
2. When several HSMCI (x HSMCI) are embedded in a product, MCCK refers to HSMCIx_CK, MCCDA to HSMCIx_CDA, MCDAy to HSMCIx_DAy.

Figure 38-4. MMC Bus Connections (One Slot)



Note: When several HSMCI (x HSMCI) are embedded in a product, MCCK refers to HSMCIx_CK, MCCDA to HSMCIx_CDA MCDAy to HSMCIx_DAy.

38.11.1 Boot Procedure, Processor Mode

1. Configure the HSMCI data bus width programming SDCBUS Field in the HSMCI_SDCR. The BOOT_BUS_WIDTH field located in the device Extended CSD register must be set accordingly.
2. Set the byte count to 512 bytes and the block count to the desired number of blocks, writing BLKLEN and BCNT fields of the HSMCI_BLKR.
3. Issue the Boot Operation Request command by writing to the HSMCI_CMDR with SPCMD field set to BOOTREQ, TRDIR set to READ and TRCMD set to “start data transfer”.
4. The BOOT_ACK field located in the HSMCI_CMDR must be set to one, if the BOOT_ACK field of the MMC device located in the Extended CSD register is set to one.
5. Host processor can copy boot data sequentially as soon as the RXRDY flag is asserted.
6. When Data transfer is completed, host processor shall terminate the boot stream by writing the HSMCI_CMDR with SPCMD field set to BOOTEND.

38.12 HSMCI Transfer Done Timings

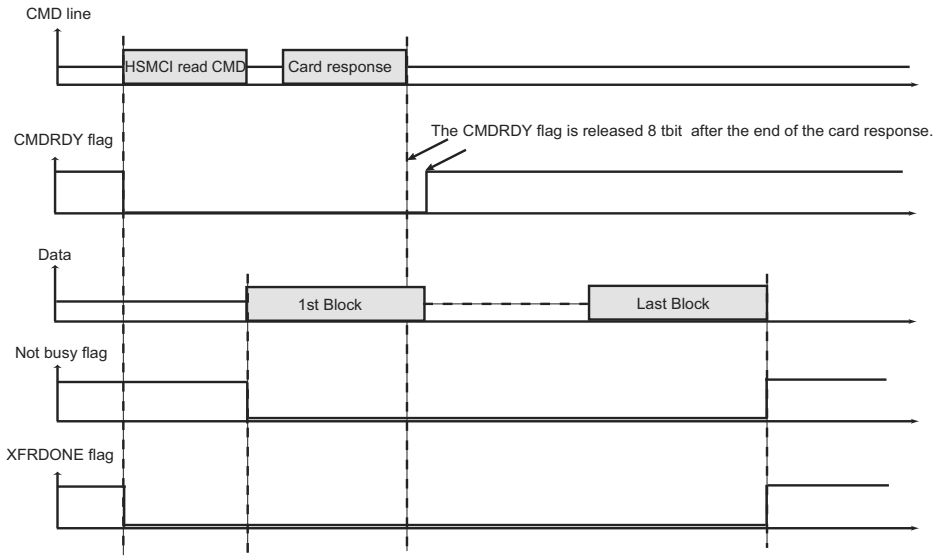
38.12.1 Definition

The XFRDONE flag in the HSMCI_SR indicates exactly when the read or write sequence is finished.

38.12.2 Read Access

During a read access, the XFRDONE flag behaves as shown in Figure 38-11.

Figure 38-11. XFRDONE During a Read Access



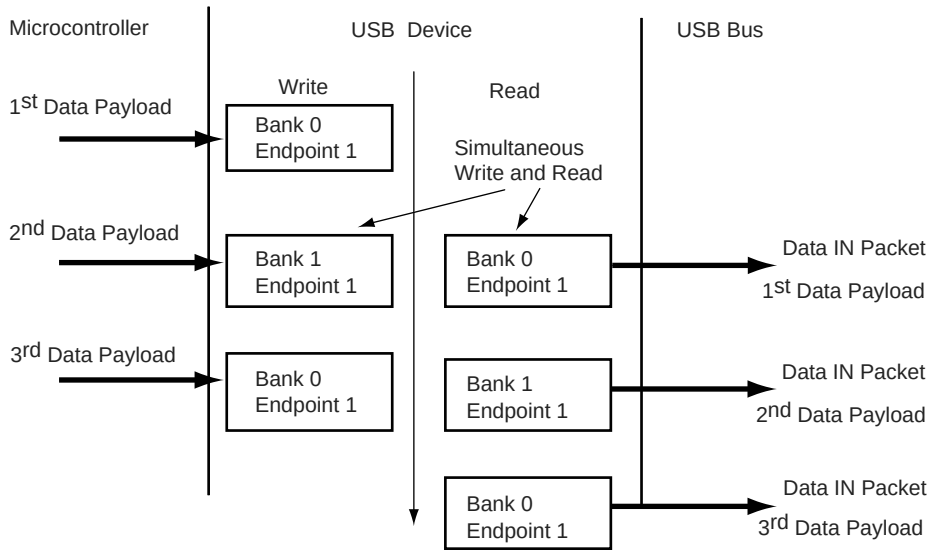
38.12.3 Write Access

During a write access, the XFRDONE flag behaves as shown in Figure 38-12.

Using Endpoints With Ping-pong Attribute

The use of an endpoint with ping-pong attributes is necessary during isochronous transfer. This also allows handling the maximum bandwidth defined in the USB specification during bulk transfer. To be able to guarantee a constant or the maximum bandwidth, the microcontroller must prepare the next data payload to be sent while the current one is being sent 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.

Figure 40-7. Bank Swapping Data IN Transfer for Ping-pong Endpoints



When using a ping-pong endpoint, the following procedures are required to perform Data IN transactions:

1. The microcontroller checks if it is possible to write in the FIFO by polling TXPKTRDY to be cleared in the endpoint's UDP_CSRx.
2. The microcontroller writes the first data payload to be sent in the FIFO (Bank 0), writing zero or more byte values in the endpoint's UDP_FDRx.
3. The microcontroller notifies the USB peripheral it has finished writing in Bank 0 of the FIFO by setting the TXPKTRDY in the endpoint's UDP_CSRx.
4. Without waiting for TXPKTRDY to be cleared, the microcontroller writes the second data payload to be sent in the FIFO (Bank 1), writing zero or more byte values in the endpoint's UDP_FDRx.
5. The microcontroller is notified that the first Bank has been released by the USB device when TXCOMP in the endpoint's UDP_CSRx is set. An interrupt is pending while TXCOMP is being set.
6. Once the microcontroller has received TXCOMP for the first Bank, it notifies the USB device that it has prepared the second Bank to be sent, raising TXPKTRDY in the endpoint's UDP_CSRx.
7. At this step, Bank 0 is available and the microcontroller can prepare a third data payload to be sent.

40.6.3.2 Entering Attached State

To enable integrated pull-up, the PUON bit in the UDP_TXVC register must be set.

Warning: To write to the UDP_TXVC register, MCK clock must be enabled on the UDP. This is done in the Power Management Controller.

After pull-up connection, the device enters the powered state. In this state, the UDPCK and MCK must be enabled in the Power Management Controller. The transceiver can remain disabled.

40.6.3.3 From Powered State to Default State

After its connection to a USB host, the USB device waits for an end-of-bus reset. The unmaskable flag ENDBUSRES is set in the UDP_ISR and an interrupt is triggered.

Once the ENDBUSRES interrupt has been triggered, the device enters Default State. In this state, the UDP software must:

- Enable the default endpoint, setting the EPEDS flag in the UDP_CSR0 and, optionally, enabling the interrupt for endpoint 0 by writing 1 to the UDP_IER. The enumeration then begins by a control transfer.
- Configure the interrupt mask register which has been reset by the USB reset detection
- Enable the transceiver clearing the TXVDIS flag in the UDP_TXVC register.

In this state UDPCK and MCK must be enabled.

Warning: Each time an ENDBUSRES interrupt is triggered, the Interrupt Mask Register and UDP_CSRs have been reset.

40.6.3.4 From Default State to Address State

After a set address standard device request, the USB host peripheral enters the address state.

Warning: Before the device enters in address state, it must achieve the Status IN transaction of the control transfer, i.e., the UDP device sets its new address once the TXCOMP flag in the UDP_CSR0 has been received and cleared.

To move to address state, the driver software sets the FADDEN flag in the UDP_GLB_STAT register, sets its new address, and sets the FEN bit in the UDP_FADDR register.

40.6.3.5 From Address State to Configured State

Once a valid Set Configuration standard request has been received and acknowledged, the device enables endpoints corresponding to the current configuration. This is done by setting the EPEDS and EPTYPE fields in the UDP_CSRx and, optionally, enabling corresponding interrupts in the UDP_IER.

40.6.3.6 Entering in Suspend State

When a Suspend (no bus activity on the USB bus) is detected, the RXSUSP signal in the UDP_ISR is set. This triggers an interrupt if the corresponding bit is set in the UDP_IMR. This flag is cleared by writing to the UDP_ICR. Then the device enters Suspend Mode.

In this state bus powered devices must drain no more than 2.5 mA from the 5V VBUS. As an example, the microcontroller switches to slow clock, disables the PLL and main oscillator, and goes into Idle Mode. It may also switch off other devices on the board.

The USB device peripheral clocks can be switched off. Resume event is asynchronously detected. MCK and UDPCK can be switched off in the Power Management controller and the USB transceiver can be disabled by setting the TXVDIS bit in the UDP_TXVC register.

Warning: Read, write operations to the UDP registers are allowed only if MCK is enabled for the UDP peripheral. Switching off MCK for the UDP peripheral must be one of the last operations after writing to the UDP_TXVC register and acknowledging the RXSUSP.

44.12.3.2 SPI Timings

SPI timings are given for the following domains:

- 1.8V domain: VDDIO from 1.65 to 1.95 V, maximum external capacitor = 20 pF
- 3.3V domain: VDDIO from 2.85 to 3.6 V, maximum external capacitor = 40 pF

Table 44-65. SPI Timings

Symbol	Parameter	Conditions	Min	Max	Unit
SPI_0	MISO Setup Time before SPCK Rises (Master)	3.3V domain	11.3	–	ns
		1.8V domain	13.3	–	ns
SPI_1	MISO Hold Time after SPCK Rises (Master)	3.3V domain	0	–	ns
		1.8V domain	0	–	ns
SPI_2	SPCK Rising to MOSI Delay (Master)	3.3V domain	-2.0	1.9	ns
		1.8V domain	-1.9	1.0	ns
SPI_3	MISO Setup Time before SPCK Falls (Master)	3.3V domain	16.2	–	ns
		1.8V domain	21.6	–	ns
SPI_4	MISO Hold Time after SPCK Falls (Master)	3.3V domain	0	–	ns
		1.8V domain	0	–	ns
SPI_5	SPCK Falling to MOSI Delay (Master)	3.3V domain	-7	-3.6	ns
		1.8V domain	-6.7	-4.2	ns
SPI_6	SPCK Falling to MISO Delay (Slave)	3.3V domain	3.4	11.1	ns
		1.8V domain	4.1	13.1	ns
SPI_7	MOSI Setup Time before SPCK Rises (Slave)	3.3V domain	0	–	ns
		1.8V domain	0	–	ns
SPI_8	MOSI Hold Time after SPCK Rises (Slave)	3.3V domain	1.3	–	ns
		1.8V domain	0.9	–	ns
SPI_9	SPCK Rising to MISO Delay (Slave)	3.3V domain	3.6	11.5	ns
		1.8V domain	4.1	12.9	ns
SPI_{10}	MOSI Setup Time before SPCK Falls (Slave)	3.3V domain	0	–	ns
		1.8V domain	0	–	ns
SPI_{11}	MOSI Hold Time after SPCK Falls (Slave)	3.3V domain	0.8	–	ns
		1.8V domain	0.9	–	ns
SPI_{12}	NPCS Setup to SPCK Rising (Slave)	3.3V domain	3.3	–	ns
		1.8V domain	3.5	–	ns
SPI_{13}	NPCS Hold after SPCK Falling (Slave)	3.3V domain	0	–	ns
		1.8V domain	0	–	ns
SPI_{14}	NPCS Setup to SPCK Falling (Slave)	3.3V domain	4	–	ns
		1.8V domain	3.6	–	ns
SPI_{15}	NPCS Hold after SPCK Falling (Slave)	3.3V domain	0	–	ns
		1.8V domain	0	–	ns

45.9 Soldering Profile

Table 45-31 gives the recommended soldering profile from J-STD-020C.

Table 45-31. Soldering Profile

Profile Feature	Green Package
Average Ramp-up Rate (217°C to Peak)	3°C/sec. max.
Preheat Temperature 175°C ± 25°C	180 sec. max.
Temperature Maintained Above 217°C	60 sec. to 150 sec.
Time within 5°C of Actual Peak Temperature	20 sec. to 40 sec.
Peak Temperature Range	260°C
Ramp-down Rate	6°C/sec. max.
Time 25°C to Peak Temperature	8 min. max.

Note: The package is certified to be backward compatible with Pb/Sn soldering profile.

A maximum of three reflow passes is allowed per component.

45.10 Packaging Resources

Land Pattern Definition.

Refer to the following IPC Standards:

- IPC-7351A and IPC-782 (*Generic Requirements for Surface Mount Design and Land Pattern Standards*) http://landpatterns.ipc.org/default.asp
- Atmel Green and RoHS Policy and Package Material Declaration Datasheet available on www.atmel.com

48. Errata

48.1 Errata SAM4SD32/SD16/SA16/S16/S8 Rev. A Parts

The errata are applicable to the devices in Table 48-1.

Table 48-1. Device List for Errata Described in Section 48.1

Device Name	Revision	Chip ID
SAM4SD32C	A	0x29A7_0EE0
SAM4SD32B	A	0x2997_0EE0
SAM4SD16C	A	0x29A7_0CE0
SAM4SD16B	A	0x2997_0CE0
SAM4SA16C	A	0x28A7_0CE0
SAM4SA16B	A	0x2897_0CE0
SAM4S16C	A	0x28AC_0CE0
SAM4S16B	A	0x289C_0CE0
SAM4S8C	A	0x28AC_0AE0
SAM4S8B	A	0x289C_0AE0

48.1.1 Flash Controller (EEFC)

Issue: Flash Buffer Not Cleared

The Write Buffer in the embedded Flash is not cleared after trying to write to a locked region. Therefore, the data that was previously loaded into the Write Buffer would remain in the buffer while the next page write command (e.g., WP) is being executed.

Workaround: Do not do partial programming (Fill completely the Write Buffer). Note that this problem occurs only if the software tries to write into a locked region.

Issue: Code Loop Optimization Cannot Be Disabled

The EFC does not work after the buffer for loop optimization is disabled; in Flash Mode Register (EEFC_FMR), CLOE = 0.

Workaround: The CLOE bit must be kept at 1.

Issue: Erase Sector Command Cannot Be Performed If a Subsector Is Locked (ONLY in Flash Sector0)

If one of subsector (Small Sector 0, Small Sector1 and Larger Sector) is locked, the Erase Sector Command (ES) is not possible on non-locked subsectors.

Workaround: All the lock bits of the sector0 must be cleared prior to issuing the ES command. After the ES command has been issued, the first sector lock bits must be reverted to the state before clearing them.

48.2.5 PIO

Issue: PB4 Input Low-level Voltage Range

The undershoot is limited to -0.1V.

In normal operating conditions, the V_{IL} minimum value on PB4 is limited to 0V.

Workaround: The voltage on PB4 with respect to ground must be in the range -0.1V to + VDDIO + 0.4V instead of -0.3V to + VDDIO + 0.4V for all other input pins, as shown in Table 44.1 “Absolute Maximum Ratings”.

The minimum V_{IL} on PB4 must be 0V instead of -0.3V for all other input pins, as shown in Table 44.3 “DC Characteristics”.