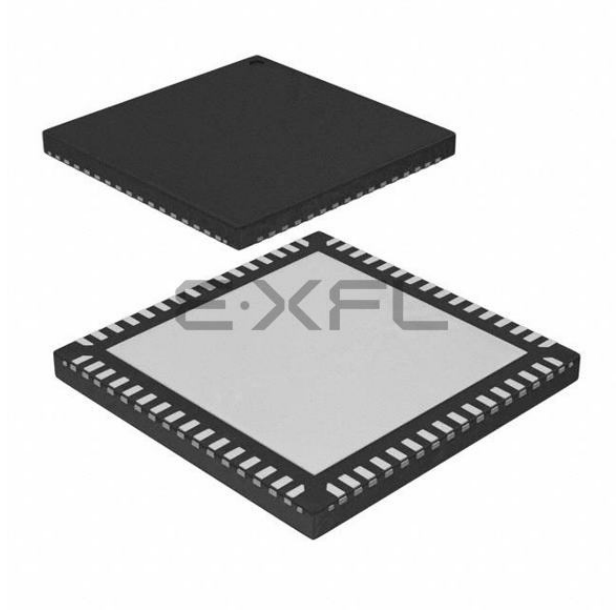


Welcome to E-XFL.COM

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	I²C, IrDA, Memory Card, SPI, SSC, UART/USART, USB
Peripherals	Brown-out Detect/Reset, DMA, POR, PWM, WDT
Number of I/O	47
Program Memory Size	2MB (2M x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	160K x 8
Voltage - Supply (Vcc/Vdd)	1.62V ~ 3.6V
Data Converters	A/D 11x12b; D/A 2x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	64-VFQFN Exposed Pad
Supplier Device Package	64-QFN (9x9)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/atsam4sd32ba-mu

12.6.5 General Data Processing Instructions

The table below shows the data processing instructions.

Table 12-20. Data Processing Instructions

Mnemonic	Description
ADC	Add with Carry
ADD	Add
ADDW	Add
AND	Logical AND
ASR	Arithmetic Shift Right
BIC	Bit Clear
CLZ	Count leading zeros
CMN	Compare Negative
CMP	Compare
EOR	Exclusive OR
LSL	Logical Shift Left
LSR	Logical Shift Right
MOV	Move
MOVT	Move Top
MOVW	Move 16-bit constant
MVN	Move NOT
ORN	Logical OR NOT
ORR	Logical OR
RBIT	Reverse Bits
REV	Reverse byte order in a word
REV16	Reverse byte order in each halfword
REVSH	Reverse byte order in bottom halfword and sign extend
ROR	Rotate Right
RRX	Rotate Right with Extend
RSB	Reverse Subtract
SADD16	Signed Add 16
SADD8	Signed Add 8
SASX	Signed Add and Subtract with Exchange
SSAX	Signed Subtract and Add with Exchange
SBC	Subtract with Carry
SHADD16	Signed Halving Add 16
SHADD8	Signed Halving Add 8
SHASX	Signed Halving Add and Subtract with Exchange
SHSAX	Signed Halving Subtract and Add with Exchange

12.6.5.1 ADD, ADC, SUB, SBC, and RSB

Add, Add with carry, Subtract, Subtract with carry, and Reverse Subtract.

Syntax

```
op{S}{cond} {Rd,} Rn, Operand2
op{cond} {Rd,} Rn, #imm12              ; ADD and SUB only
```

where:

op is one of:

ADD Add.

ADC Add with Carry.

SUB Subtract.

SBC Subtract with Carry.

RSB Reverse Subtract.

S is an optional suffix. If S is specified, the condition code flags are updated on the result of the operation, see “Conditional Execution” .

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

Rd is the destination register. If *Rd* is omitted, the destination register is *Rn*.

Rn is the register holding the first operand.

Operand2 is a flexible second operand. See “Flexible Second Operand” for details of the options.

imm12 is any value in the range 0–4095.

Operation

The ADD instruction adds the value of *Operand2* or *imm12* to the value in *Rn*.

The ADC instruction adds the values in *Rn* and *Operand2*, together with the carry flag.

The SUB instruction subtracts the value of *Operand2* or *imm12* from the value in *Rn*.

The SBC instruction subtracts the value of *Operand2* from the value in *Rn*. If the carry flag is clear, the result is reduced by one.

The RSB instruction subtracts the value in *Rn* from the value of *Operand2*. This is useful because of the wide range of options for *Operand2*.

Use ADC and SBC to synthesize multiword arithmetic, see *Multiword arithmetic examples* on.

See also “ADR” .

Note: ADDW is equivalent to the ADD syntax that uses the *imm12* operand. SUBW is equivalent to the SUB syntax that uses the *imm12* operand.

Restrictions

In these instructions:

- *Operand2* must not be SP and must not be PC
- *Rd* can be SP only in ADD and SUB, and only with the additional restrictions:
  - *Rn* must also be SP
  - Any shift in *Operand2* must be limited to a maximum of 3 bits using LSL
- *Rn* can be SP only in ADD and SUB
- *Rd* can be PC only in the ADD{*cond*} PC, PC, Rm instruction where:
  - The user must not specify the S suffix
  - *Rm* must not be PC and must not be SP

In these instructions:

- Do not use SP and do not use PC.
- *RdHi* and *RdLo* must be different registers.

Examples

```
SMULBT      R0, R4, R5  ; Multiplies the bottom halfword of R4 with the
                        ; top halfword of R5, multiplies results and
                        ; writes to R0
SMULBB      R0, R4, R5  ; Multiplies the bottom halfword of R4 with the
                        ; bottom halfword of R5, multiplies results and
                        ; writes to R0
SMULTT      R0, R4, R5  ; Multiplies the top halfword of R4 with the top
                        ; halfword of R5, multiplies results and writes
                        ; to R0
SMULTB      R0, R4, R5  ; Multiplies the top halfword of R4 with the
                        ; bottom halfword of R5, multiplies results and
                        ; and writes to R0
SMULWT      R4, R5, R3  ; Multiplies R5 with the top halfword of R3,
                        ; extracts top 32 bits and writes to R4
SMULWB      R4, R5, R3  ; Multiplies R5 with the bottom halfword of R3,
                        ; extracts top 32 bits and writes to R4.
```

12.7 Cortex-M4 Core Peripherals

12.7.1 Peripherals

- Nested Vectored Interrupt Controller (NVIC)
  The Nested Vectored Interrupt Controller (NVIC) is an embedded interrupt controller that supports low latency interrupt processing. See Section 12.8 ”Nested Vectored Interrupt Controller (NVIC)”.
- System Control Block (SCB)
  The System Control Block (SCB) is the programmers model interface to the processor. It provides system implementation information and system control, including configuration, control, and reporting of system exceptions. See Section 12.9 ”System Control Block (SCB)”.
- System Timer (SysTick)
  The System Timer, SysTick, is a 24-bit count-down timer. Use this as a Real Time Operating System (RTOS) tick timer or as a simple counter. See Section 12.10 ”System Timer (SysTick)”.
- Memory Protection Unit (MPU)
  The Memory Protection Unit (MPU) improves system reliability by defining the memory attributes for different memory regions. It provides up to eight different regions, and an optional predefined background region. See Section 12.11 ”Memory Protection Unit (MPU)”.

12.7.2 Address Map

The address map of the *Private peripheral bus* (PPB) is given in the following table.

Table 12-28. Core Peripheral Register Regions

Address	Core Peripheral
0xE000E008–0xE000E00F	System Control Block
0xE000E010–0xE000E01F	System Timer
0xE000E100–0xE000E4EF	Nested Vectored Interrupt Controller
0xE000ED00–0xE000ED3F	System control block
0xE000ED90–0xE000EDB8	Memory Protection Unit
0xE000EF00–0xE000EF03	Nested Vectored Interrupt Controller

In register descriptions:

- The *required privilege* gives the privilege level required to access the register, as follows:
  - Privileged: Only privileged software can access the register.
  - Unprivileged: Both unprivileged and privileged software can access the register.

the final product manufacturing by means of measurement equipment embedding such a reference clock. The correction of value must be programmed into the (RTC_MR), and this value is kept as long as the circuitry is powered (backup area). Removing the backup power supply cancels this calibration. This room temperature calibration can be further processed by means of the networking capability of the target application.

To ease the comparison of the inherent crystal accuracy with the reference clock/signal during manufacturing, an internal prescaled 32.768 kHz clock derivative signal can be assigned to drive RTC output. To accommodate the measure, several clock frequencies can be selected among 1 Hz, 32 Hz, 64 Hz, 512 Hz.

The clock calibration correction drives the internal RTC counters but can also be observed in the RTC output when one of the following three frequencies 1 Hz, 32 Hz or 64 Hz is configured. The correction is not visible in the RTC output if 512 Hz frequency is configured.

In any event, this adjustment does not take into account the temperature variation.

The frequency drift (up to -200 ppm) due to temperature variation can be compensated using a reference time if the application can access such a reference. If a reference time cannot be used, a temperature sensor can be placed close to the crystal oscillator in order to get the operating temperature of the crystal oscillator. Once obtained, the temperature may be converted using a lookup table (describing the accuracy/temperature curve of the crystal oscillator used) and RTC_MR configured accordingly. The calibration can be performed on-the-fly. This adjustment method is not based on a measurement of the crystal frequency/drift and therefore can be improved by means of the networking capability of the target application.

If no crystal frequency adjustment has been done during manufacturing, it is still possible to do it. In the case where a reference time of the day can be obtained through LAN/WAN network, it is possible to calculate the drift of the application crystal oscillator by comparing the values read on RTC Time Register (RTC_TIMR) and programming the HIGHPPM and CORRECTION fields on RTC_MR according to the difference measured between the reference time and those of RTC_TIMR.

16.5.8 Waveform Generation

Waveforms can be generated by the RTC in order to take advantage of the RTC inherent prescalers while the RTC is the only powered circuitry (low power mode of operation, backup mode) or in any active modes. Going into backup or low power operating modes does not affect the waveform generation outputs.

The RTC outputs (RTCOUT0 and RTCOUT1) have a source driver selected among seven possibilities.

The first selection choice sticks the associated output at 0 (This is the reset value and it can be used at any time to disable the waveform generation).

Selection choices 1 to 4 respectively select 1 Hz, 32 Hz, 64 Hz and 512 Hz.

32 Hz or 64 Hz can drive, for example, a TN LCD backplane signal while 1 Hz can be used to drive a blinking character like “:” for basic time display (hour, minute) on TN LCDs.

Selection choice 5 provides a toggling signal when the RTC alarm is reached.

Selection choice 6 provides a copy of the alarm flag, so the associated output is set high (logical 1) when an alarm occurs and immediately cleared when software clears the alarm interrupt source.

Selection choice 7 provides a 1 Hz periodic high pulse of 15 µs duration that can be used to drive external devices for power consumption reduction or any other purpose.

PIO lines associated to RTC outputs are automatically selecting these waveforms as soon as RTC_MR corresponding fields OUT0 and OUT1 differ from 0.

26.16.2 SMC Pulse Register

Name: SMC_PULSE[0..3]

Address: 0x400E0004 [0], 0x400E0014 [1], 0x400E0024 [2], 0x400E0034 [3]

Access: Read/Write

31	30	29	28	27	26	25	24
–	NCS_RD_PULSE						

23	22	21	20	19	18	17	16
–	NRD_PULSE						

15	14	13	12	11	10	9	8
–	NCS_WR_PULSE						

7	6	5	4	3	2	1	0
–	NWE_PULSE						

This register can only be written if the WPEN bit is cleared in the “SMC Write Protection Mode Register” .

- **NWE_PULSE: NWE Pulse Length**

The NWE signal pulse length is defined as:

NWE pulse length = (256* NWE_PULSE[6] + NWE_PULSE[5:0]) clock cycles

The NWE pulse length must be at least 1 clock cycle.

- **NCS_WR_PULSE: NCS Pulse Length in WRITE Access**

In write access, the NCS signal pulse length is defined as:

NCS pulse length = (256* NCS_WR_PULSE[6] + NCS_WR_PULSE[5:0]) clock cycles

The NCS pulse length must be at least 1 clock cycle.

- **NRD_PULSE: NRD Pulse Length**

In standard read access, the NRD signal pulse length is defined in clock cycles as:

NRD pulse length = (256* NRD_PULSE[6] + NRD_PULSE[5:0]) clock cycles

The NRD pulse length must be at least 1 clock cycle.

In Page mode read access, the NRD_PULSE parameter defines the duration of the subsequent accesses in the page.

- **NCS_RD_PULSE: NCS Pulse Length in READ Access**

In standard read access, the NCS signal pulse length is defined as:

NCS pulse length = (256* NCS_RD_PULSE[6] + NCS_RD_PULSE[5:0]) clock cycles

The NCS pulse length must be at least 1 clock cycle.

In Page mode read access, the NCS_RD_PULSE parameter defines the duration of the first access to one page.

29.17.17PMC Interrupt Mask Register

Name: PMC_IMR

Address: 0x400E046C

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	CFDEV	MOSCRCS	MOSCSELS
15	14	13	12	11	10	9	8
–	–	–	–	–	PCKRDY2	PCKRDY1	PCKRDY0
7	6	5	4	3	2	1	0
–	–	–	–	MCKRDY	LOCKB	LOCKA	MOSCXTS

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

0: No effect.

1: Enables the corresponding interrupt.

- **MOSCXTS: Main Crystal Oscillator Status Interrupt Mask**
- **LOCKA: PLLA Lock Interrupt Mask**
- **LOCKB: PLLB Lock Interrupt Mask**
- **MCKRDY: Master Clock Ready Interrupt Mask**
- **PCKRDYx: Programmable Clock Ready x Interrupt Mask**
- **MOSCSELS: Main Oscillator Selection Status Interrupt Mask**
- **MOSCRCS: Main On-Chip RC Status Interrupt Mask**
- **CFDEV: Clock Failure Detector Event Interrupt Mask**

31.6.12 PIO Output Data Status Register

Name: PIO_ODSR

Address: 0x400E0E38 (PIOA), 0x400E1038 (PIOB), 0x400E1238 (PIOC)

Access: Read-only or Read/Write

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: Output Data Status**

0: The data to be driven on the I/O line is 0.

1: The data to be driven on the I/O line is 1.

31.6.36 PIO Additional Interrupt Modes Enable Register

Name: PIO_AIMER

Address: 0x400E0EB0 (PIOA), 0x400E10B0 (PIOB), 0x400E12B0 (PIOC)

Access: 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: Additional Interrupt Modes Enable**

0: No effect.

1: The interrupt source is the event described in PIO_ELSR and PIO_FRLHSR.

Figure 34-10. Master Write with One, Two or Three Bytes Internal Address and One Data Byte

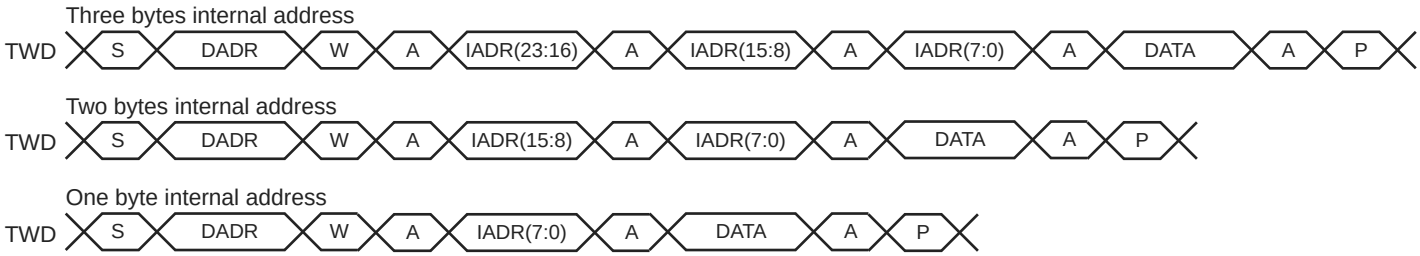
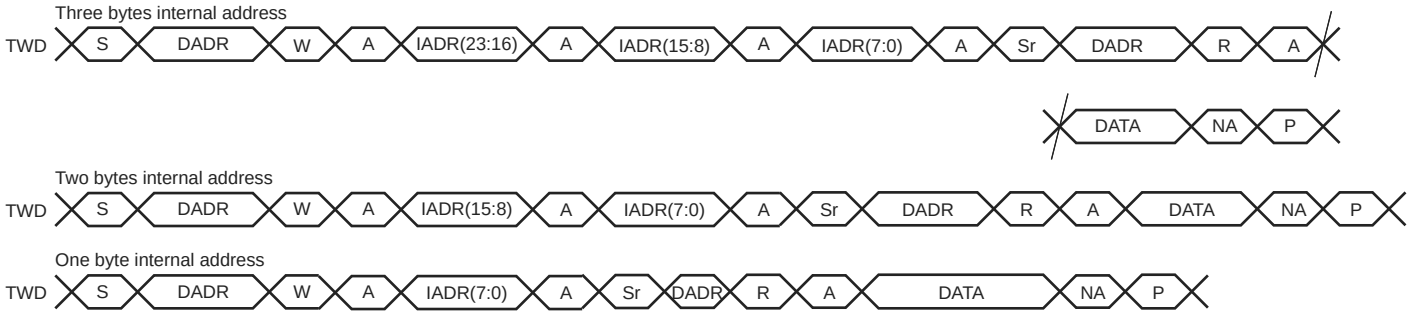


Figure 34-11. Master Read with One, Two or Three Bytes Internal Address and One Data Byte



10-bit Slave Addressing

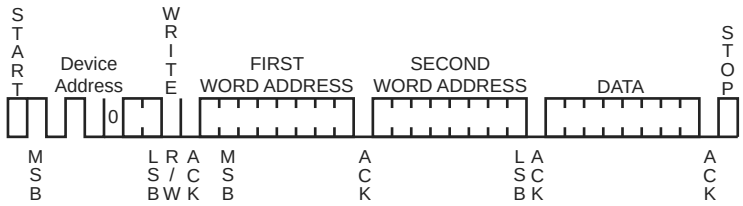
For a slave address higher than seven bits, the user must configure the address size **(**IADRSZ) and set the other slave address bits in the Internal Address register (TWI_IADR). The two remaining internal address bytes, IADR[15:8] and IADR[23:16] can be used the same way as in 7-bit slave addressing.

Example: Address a 10-bit device (10-bit device address is b1 b2 b3 b4 b5 b6 b7 b8 b9 b10)

1. Program IADRSZ = 1,
2. Program DADR with 1 1 1 1 0 b1 b2 (b1 is the MSB of the 10-bit address, b2, etc.)
3. Program TWI_IADR with b3 b4 b5 b6 b7 b8 b9 b10 (b10 is the LSB of the 10-bit address)

Figure 34-12 below shows a byte write to an Atmel AT24LC512 EEPROM. This demonstrates the use of internal addresses to access the device.

Figure 34-12. Internal Address Usage



34.7.3.7 Using the Peripheral DMA Controller (PDC)

The use of the PDC significantly reduces the CPU load.

To ensure correct implementation, proceed as follows.

Data Transmit with the PDC

1. Initialize the transmit PDC (memory pointers, transfer size - 1).
2. Configure the master (DADR, CKDIV, MREAD = 0, etc.)
3. Start the transfer by setting the PDC TXTEN bit.

Figure 34-20. Programmer Sends Data While the Bus is Busy

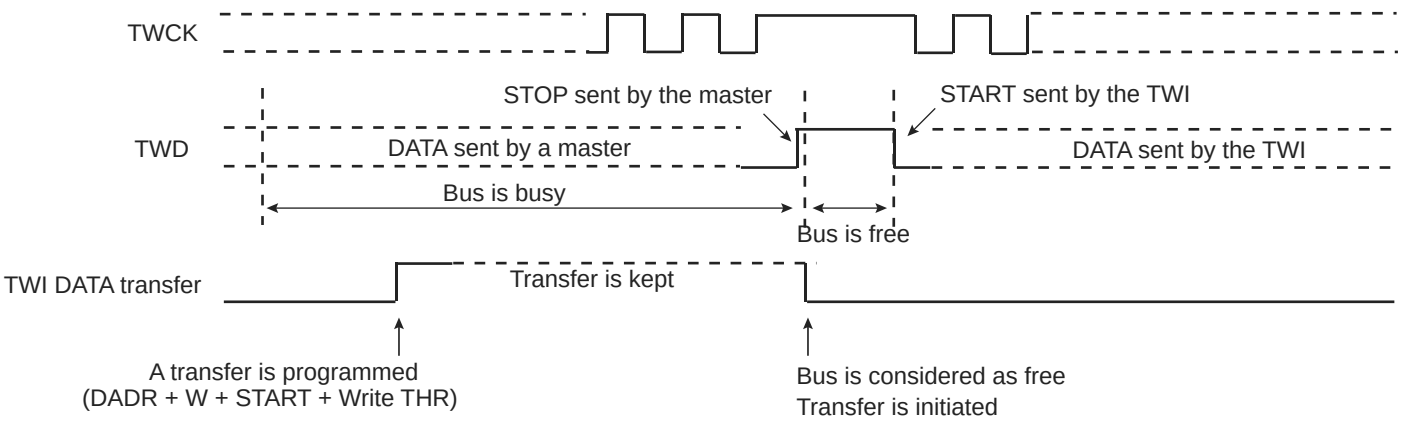
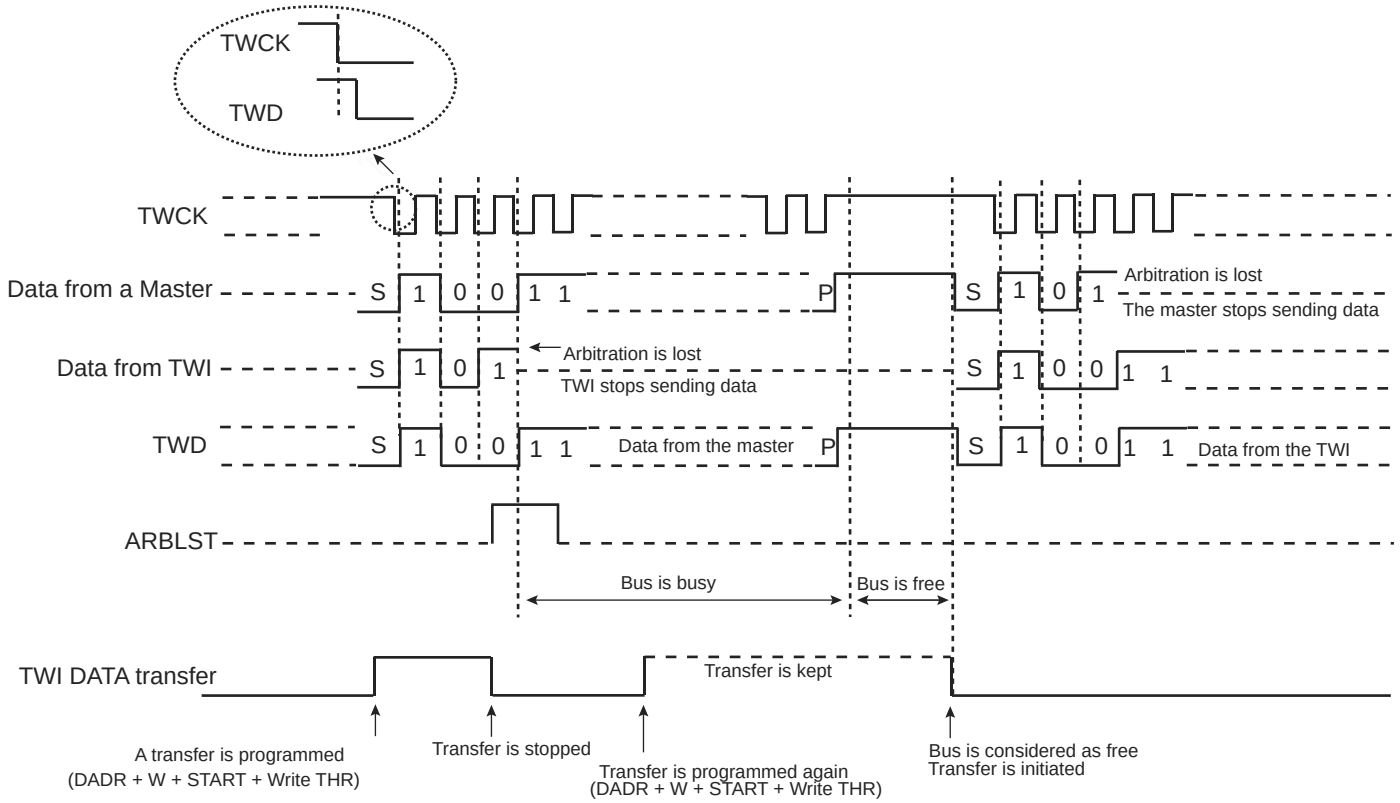
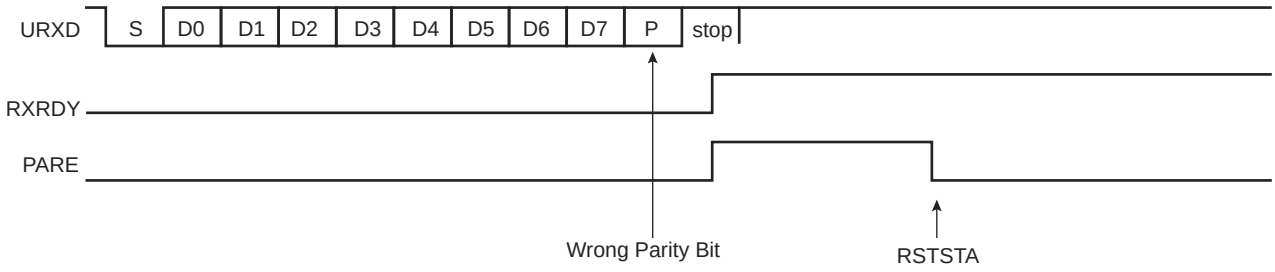


Figure 34-21. Arbitration Cases



The flowchart shown in Figure 34-22 gives an example of read and write operations in Multi-master mode.

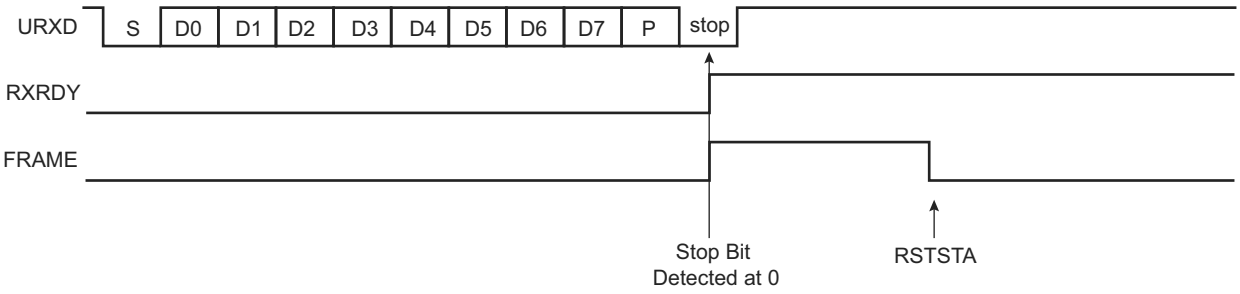
Figure 35-7. Parity Error



35.5.2.6 Receiver Framing Error

When a start bit is detected, it generates a character reception when all the data bits have been sampled. The stop bit is also sampled and when it is detected at 0, the FRAME (Framing Error) bit in UART_SR is set at the same time the RXRDY bit is set. The FRAME bit remains high until the Control Register (UART_CR) is written with the bit RSTSTA at 1.

Figure 35-8. Receiver Framing Error



35.5.3 Transmitter

35.5.3.1 Transmitter Reset, Enable and Disable

After device reset, the UART transmitter is disabled and must be enabled before being used. The transmitter is enabled by writing UART_CR with the bit TXEN at 1. From this command, the transmitter waits for a character to be written in the Transmit Holding Register (UART_THR) before actually starting the transmission.

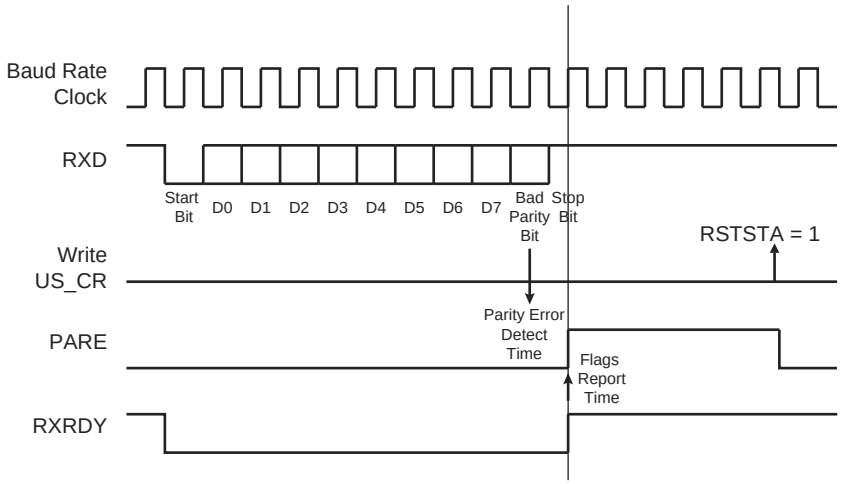
The programmer can disable the transmitter by writing UART_CR with the bit TXDIS at 1. If the transmitter is not operating, it is immediately stopped. However, if a character is being processed into the internal shift register and/or a character has been written in the UART_THR, the characters are completed before the transmitter is actually stopped.

The programmer can also put the transmitter in its reset state by writing the UART_CR with the bit RSTTX at 1. This immediately stops the transmitter, whether or not it is processing characters.

35.5.3.2 Transmit Format

The UART transmitter drives the pin UTXD at the baud rate clock speed. The line is driven depending on the format defined in UART_MR and the data stored in the internal shift register. One start bit at level 0, then the 8 data bits, from the lowest to the highest bit, one optional parity bit and one stop bit at 1 are consecutively shifted out as shown in the following figure. The field PARE in UART_MR defines whether or not a parity bit is shifted out. When a parity bit is enabled, it can be selected between an odd parity, an even parity, or a fixed space or mark bit.

Figure 36-21. Parity Error



36.6.3.9 Multidrop Mode

If the value 0x6 or 0x07 is written to the PAR field in the US_MR, the USART runs in Multidrop mode. This mode differentiates the data characters and the address characters. Data is transmitted with the parity bit at 0 and addresses are transmitted with the parity bit at 1.

If the USART is configured in Multidrop mode, the receiver sets the PARE parity error bit when the parity bit is high and the transmitter is able to send a character with the parity bit high when a 1 is written to the SENTA bit in the US_CR.

To handle parity error, the PARE bit is cleared when a 1 is written to the RSTSTA bit in the US_CR.

The transmitter sends an address byte (parity bit set) when SENDA is written to in the US_CR. In this case, the next byte written to the US_THR is transmitted as an address. Any character written in the US_THR without having written the command SENDA is transmitted normally with the parity at 0.

36.6.3.10 Transmitter Timeguard

The timeguard feature enables the USART interface with slow remote devices.

The timeguard function enables the transmitter to insert an idle state on the TXD line between two characters. This idle state actually acts as a long stop bit.

The duration of the idle state is programmed in the TG field of the Transmitter Timeguard register (US_TTGR). When this field is written to zero no timeguard is generated. Otherwise, the transmitter holds a high level on TXD after each transmitted byte during the number of bit periods programmed in TG in addition to the number of stop bits.

As illustrated in Figure 36-22, the behavior of TXRDY and TXEMPTY status bits is modified by the programming of a timeguard. TXRDY rises only when the start bit of the next character is sent, and thus remains to 0 during the timeguard transmission if a character has been written in US_THR. TXEMPTY remains low until the timeguard transmission is completed as the timeguard is part of the current character being transmitted.

38.13 Register Write Protection

To prevent any single software error from corrupting HSMCI behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the HSMCI Write Protection Mode Register (HSMCI_WPMR).

If a write access to a write-protected register is detected, the WPVS bit in the HSMCI Write Protection Status Register (HSMCI_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading the HSMCI_WPSR.

The following registers can be protected:

- HSMCI Mode Register
- HSMCI Data Timeout Register
- HSMCI SDCard/SDIO Register
- HSMCI Completion Signal Timeout Register
- HSMCI Configuration Register

All of the PWM outputs may or may not be enabled. If an application requires only four channels, then only four PIO lines will be assigned to PWM outputs.

Table 39-2. I/O Lines

Instance	Signal	I/O Line	Peripheral
PWM	PWMFI0	PA9	C
PWM	PWMFI1	PA10	C
PWM	PWMFI2	PA18	D
PWM	PWMH0	PA0	A
PWM	PWMH0	PA11	B
PWM	PWMH0	PA23	B
PWM	PWMH0	PB0	A
PWM	PWMH0	PC18	B
PWM	PWMH1	PA1	A
PWM	PWMH1	PA12	B
PWM	PWMH1	PA24	B
PWM	PWMH1	PB1	A
PWM	PWMH1	PC19	B
PWM	PWMH2	PA2	A
PWM	PWMH2	PA13	B
PWM	PWMH2	PA25	B
PWM	PWMH2	PB4	B
PWM	PWMH2	PC20	B
PWM	PWMH3	PA7	B
PWM	PWMH3	PA14	B
PWM	PWMH3	PA17	C
PWM	PWMH3	PB14	B
PWM	PWMH3	PC21	B
PWM	PWML0	PA19	B
PWM	PWML0	PB5	B
PWM	PWML0	PC0	B
PWM	PWML0	PC13	B
PWM	PWML1	PA20	B
PWM	PWML1	PB12	A
PWM	PWML1	PC1	B
PWM	PWML1	PC15	B
PWM	PWML2	PA16	C
PWM	PWML2	PA30	A
PWM	PWML2	PB13	A
PWM	PWML2	PC2	B

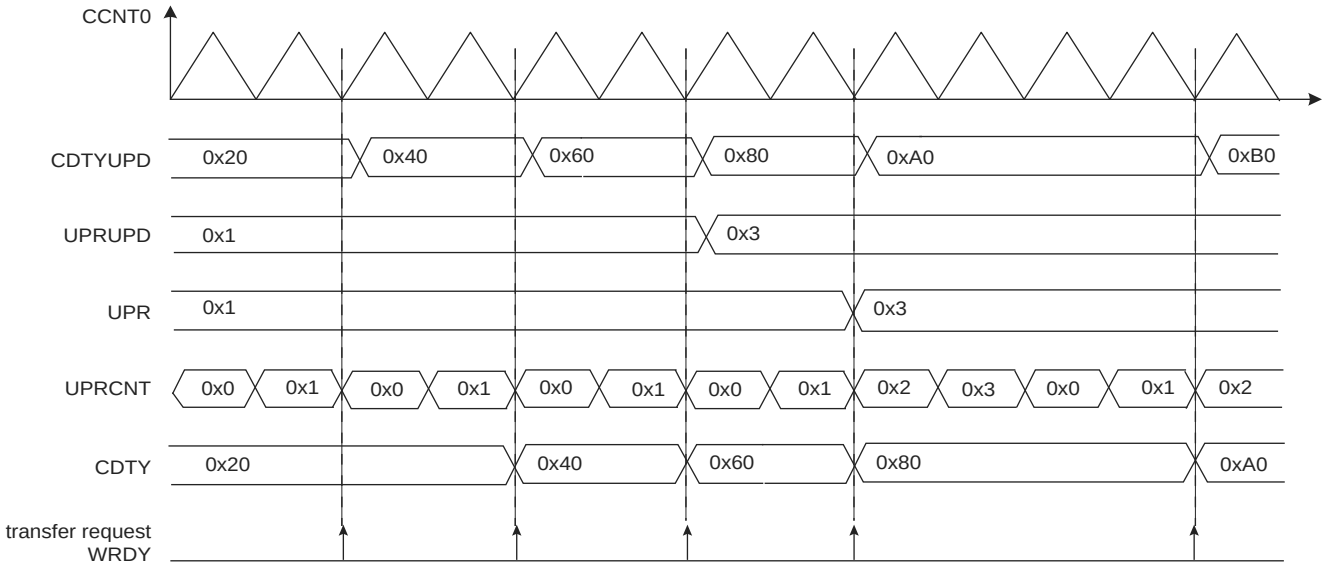
- UNRE: this flag is set to ‘1’ when the update period defined by the UPR field has elapsed while the whole data has not been written by the Peripheral DMA Controller. It is reset to ‘0’ when the PWM_ISR2 is read.

Depending on the interrupt mask in the PWM_IMR2, an interrupt can be generated by these flags.

Sequence for Method 3:

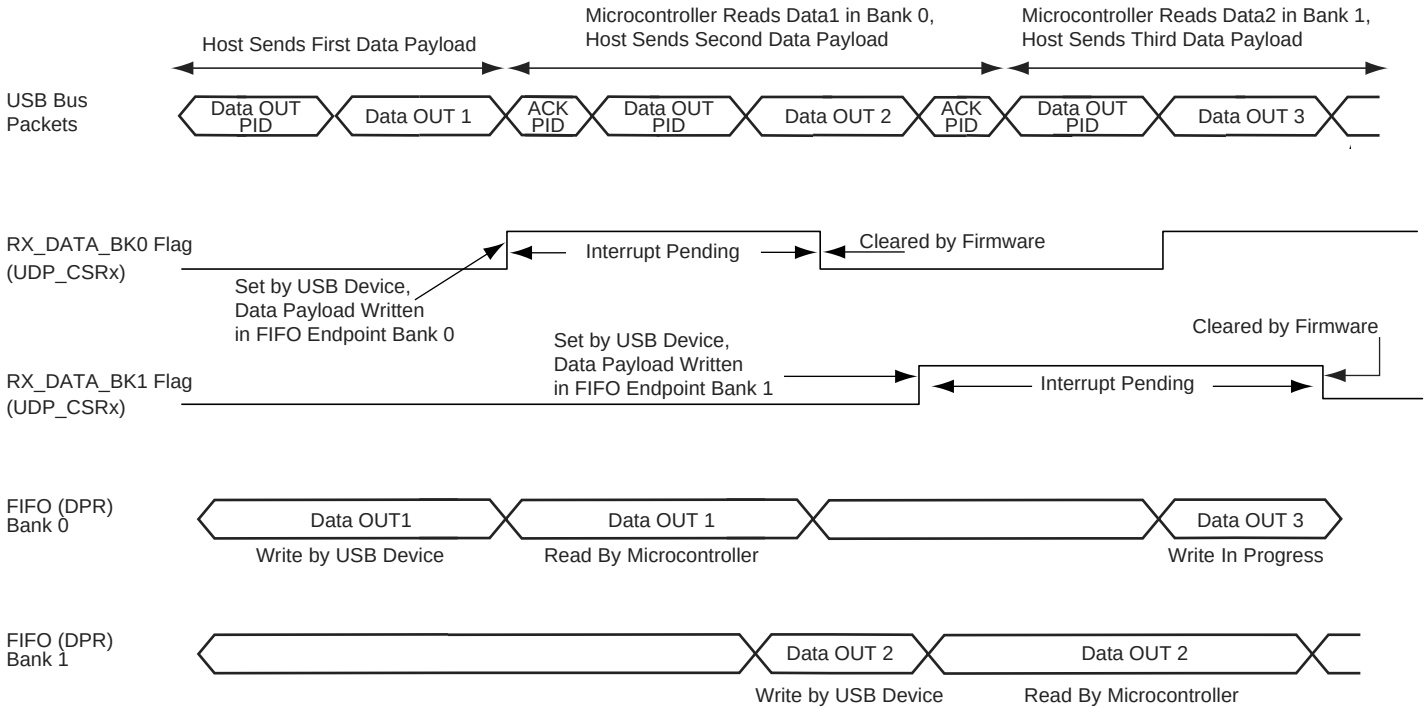
1. Select the automatic write of duty-cycle values and automatic update by setting the field UPDM to 2 in the PWM_SCM register.
2. Define the synchronous channels by the bits SYNCx in the PWM_SCM register.
3. Define the update period by the field UPR in the PWM_SCUP register.
4. Define when the WRDY flag and the corresponding Peripheral DMA Controller transfer request must be set in the update period by the PTRM bit and the PTRCS field in the PWM_SCM register (at the end of the update period or when a comparison matches).
5. Define the Peripheral DMA Controller transfer settings for the duty-cycle values and enable it in the Peripheral DMA Controller registers
6. Enable the synchronous channels by writing CHID0 in the PWM_ENA register.
7. If an update of the period value and/or of the dead-time values is required, write registers that need to be updated (PWM_CPRDUPDx, PWM_DTUPDx), else go to Step 10.
8. Set UPDULOCK to ‘1’ in PWM_SCUC.
9. The update of these registers will occur at the beginning of the next PWM period. At this moment the bit UPDULOCK is reset, go to Step 7. for new values.
10. If an update of the update period value is required, check first that write of a new update value is possible by polling the flag WRDY (or by waiting for the corresponding interrupt) in the PWM_ISR2, else go to Step 13.
11. Write the register that needs to be updated (PWM_SCUPUPD).
12. The update of this register will occur at the next PWM period of the synchronous channels when the Update Period is elapsed. Go to Step 10. for new values.
13. Check the end of the PDC transfer by the flag ENDTX. If the transfer has ended, define a new PDC transfer in the PDC registers for new duty-cycle values. Go to Step 5.

Figure 39-12. Method 3 (UPDM = 2 and PTRM = 0)



5. The number of bytes available in the FIFO is made available by reading RXBYTECNT in the endpoint's UDP_CSRx.
6. The microcontroller transfers out data received from the endpoint's memory to the microcontroller's memory. Data received is made available by reading the endpoint's UDP_FDRx.
7. The microcontroller notifies the USB peripheral device that it has finished the transfer by clearing RX_DATA_BK0 in the endpoint's UDP_CSRx.
8. A third Data OUT packet can be accepted by the USB peripheral device and copied in the FIFO Bank 0.
9. If a second Data OUT packet has been received, the microcontroller is notified by the flag RX_DATA_BK1 set in the endpoint's UDP_CSRx. An interrupt is pending for this endpoint while RX_DATA_BK1 is set.
10. The microcontroller transfers out data received from the endpoint's memory to the microcontroller's memory. Data received is available by reading the endpoint's UDP_FDRx.
11. The microcontroller notifies the USB device it has finished the transfer by clearing RX_DATA_BK1 in the endpoint's UDP_CSRx.
12. A fourth Data OUT packet can be accepted by the USB device and copied in the FIFO Bank 1.

Figure 40-11. Data OUT Transfer for Ping-pong Endpoint



Note: An interrupt is pending while the RX_DATA_BK0 or RX_DATA_BK1 flag is set.

Warning: When RX_DATA_BK0 and RX_DATA_BK1 are both set, there is no way to determine which one to clear first. Thus the software must keep an internal counter to be sure to clear alternatively RX_DATA_BK0 then RX_DATA_BK1. This situation may occur when the software application is busy elsewhere and the two banks are filled by the USB host. Once the application comes back to the USB driver, the two flags are set.

- **TRANSFER: Hold Time**

  The TRANSFER field should be set to 2 to guarantee the optimal hold time.

- **USEQ: Use Sequence Enable**

Value	Name	Description
0	NUM_ORDER	Normal Mode: The controller converts channels in a simple numeric order depending only on the channel index.
1	REG_ORDER	User Sequence Mode: The sequence respects what is defined in ADC_SEQR1 and ADC_SEQR2 registers and can be used to convert the same channel several times.

Table 44-18. SAM4S4/S2 Typical Current Consumption in Wait Mode

Conditions	@ 25°C		@ 85°C	@ 105°C	Unit
	VDDOUT Consumption (AMP1)	Total Consumption (AMP2)	Total Consumption (AMP2)	Total Consumption (AMP2	
See Figure 44-9 on page 1155 There is no activity on the I/Os of the device. With the Flash in standby mode	14.9	28.4	211	436	µA
See Figure 44-9 on page 1155 There is no activity on the I/Os of the device. With the Flash in deep power down mode	14.9	24.1	205	432	

Table 44-19. SAM4S16/S8 Typical Current Consumption in Wait Mode

Conditions	@ 25°C		@ 85°C	@ 105°C	Unit
	VDDOUT Consumption (AMP1)	Total Consumption (AMP2)	Total Consumption (AMP2)	Total Consumption (AMP2	
See Figure 44-9 on page 1155 There is no activity on the I/Os of the device. With the Flash in standby mode	20.5	32.7	344	654	µA
See Figure 44-9 on page 1155 There is no activity on the I/Os of the device. With the Flash in deep power down mode	20.5	27.8	438	589	

Table 44-20. SAM4SD32/SD16/SA16 Typical Current Consumption in Wait Mode

Conditions	@ 25°C		@ 85°C	@ 105°C	Unit
	VDDOUT Consumption (AMP1)	Total Consumption (AMP2)	Total Consumption (AMP2	Total Consumption (AMP2	
See Figure 44-9 on page 1155 There is no activity on the I/Os of the device. With the Flash in standby mode	–	42.1	633	1105	µA
See Figure 44-9 on page 1155 There is no activity on the I/Os of the device. With the Flash in deep power down mode	–	35.3	608	1085	

Table 49-6. SAM4S Datasheet Rev. 11100F 29-Jan-14 Revision History (Continued)

Doc. Date	Changes
	Section 43. “Digital-to-Analog Converter Controller (DACC)” Section 43.7.7 “DACC Interrupt Enable Register”, Section 43.7.8 “DACC Interrupt Disable Register” and Section 43.7.9 “DACC Interrupt Mask Register”: modified bit descriptions. Rework of all “refresh” related paragraphs, Section 43.7.3 “DACC Channel Enable Register” and Section 43.6.7 “DACC Timings”. Modified description for ”REFRESH: Automatic Refresh Period” field in Section 43.7.2 “DACC Mode Register”. Re-worked Section 43.6.8 “Register Write Protection” and associated registers and bit/field descriptions in Section 43.7.12 “DACC Write Protection Mode Register” and Section 43.7.13 “DACC Write Protection Status Register”.
	Section 44. “Electrical Characteristics” Added Section 44.2 “Recommended Operating Conditions”. Section 44.4 “Power Consumption”: Added power consumption values for SAM4S4/SAM4S2. Updated Section 44.4.1 “Backup Mode Current Consumption”. Removed Supply Ripple Voltage parameter from Table 44-30, “3 to 20 MHz Crystal Oscillator Characteristics” Table 44-32 ”XIN Clock Electrical Characteristics (In Bypass Mode)”: Added $C_{PARASTANDBY}$ AND $R_{PARASTANDBY}$ parameters. Updated and re-worked Section 44.8 “12-bit ADC Characteristics”: Updated Section 44.9 “12-bit DAC Characteristics”. Removed Max Voltage Ripple parameter from Table 44-55, “Analog Power Supply Characteristics”. Added Refresh Time to Table 44-56, “Channel Conversion Time and DAC Clock”. In Section 44.12 “AC Characteristics” modified ● Table 44-64, “SPI Timings”. ● Table 44-65, “SSC Timings” ● Table 44-66, “SMC Read Signals - NRD Controlled (READ_MODE = 1)” ● Table 44-68, “SMC Write Signals - NWE Controlled (WRITE_MODE = 1)” ● Table 44-69, “SMC Write Signals - NCS Controlled (WRITE_MODE = 0)” ● Table 44-70, “USART SPI Timings” Table 44-71 ”Two-wire Serial Bus Requirements”: Added parameter t_{BUF} Section 44.12.9 “Embedded Flash Characteristics”: modified Table 44-72, “Embedded Flash Wait State at 105°C”. Table 44-73, “AC Flash Characteristics”: Full Chip Erase: Added values for 256 Kbytes and 128 Kbytes. Added new parameter Page Program Time.
	Section 45. “Mechanical Characteristics” Table 45-20 ”64-ball WLCSP Package Dimensions (in mm)” Added body size for SAM4S4 for WLCSP64 package. Figure 45-8 ”48-lead LQFP Package Drawing” and corresponding characteristics added. Figure 45-9 ”48-lead QFN Package Drawing” and corresponding characteristics added.
	Section 48. “Errata” Added Section 48.3 “Errata SAM4S4/S2 Rev. A Parts”.
	Section 47. “Ordering Information” Added information on carrier type availability. Updated Table 47-1 ”Ordering Codes for SAM4S Devices”. Added new ordering codes for SAM4S4 and SAM4S2 devices.