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	128KB (128K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	64K 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/atsam4s2cb-an

12.4.1.9 Application Program Status Register

Name: APSR

Access: Read/Write

Reset: 0x000000000

31	30	29	28	27	26	25	24
N	Z	C	V	Q	–		

23	22	21	20	19	18	17	16
–				GE[3:0]			

15	14	13	12	11	10	9	8
–							

7	6	5	4	3	2	1	0
–							

The APSR contains the current state of the condition flags from previous instruction executions.

- **N: Negative Flag**

0: Operation result was positive, zero, greater than, or equal

1: Operation result was negative or less than.

- **Z: Zero Flag**

0: Operation result was not zero

1: Operation result was zero.

- **C: Carry or Borrow Flag**

Carry or borrow flag:

0: Add operation did not result in a carry bit or subtract operation resulted in a borrow bit

1: Add operation resulted in a carry bit or subtract operation did not result in a borrow bit.

- **V: Overflow Flag**

0: Operation did not result in an overflow

1: Operation resulted in an overflow.

- **Q: DSP Overflow and Saturation Flag**

Sticky saturation flag:

0: Indicates that saturation has not occurred since reset or since the bit was last cleared to zero

1: Indicates when an SSAT or USAT instruction results in saturation.

This bit is cleared to zero by software using an MRS instruction.

- **GE[19:16]: Greater Than or Equal Flags**

See “SEL” for more information.

12.6.7.5 QDADD and QDSUB

Saturating Double and Add and Saturating Double and Subtract, signed.

Syntax

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

where:

op is one of:

QDADD Saturating Double and Add.

QDSUB Saturating Double and Subtract.

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

Rd is the destination register.

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

Operation

The QDADD instruction:

- Doubles the second operand value.
- Adds the result of the doubling to the signed saturated value in the first operand.
- Writes the result to the destination register.

The QDSUB instruction:

- Doubles the second operand value.
- Subtracts the doubled value from the signed saturated value in the first operand.
- Writes the result to the destination register.

Both the doubling and the addition or subtraction have their results saturated to the 32-bit signed integer range $-2^{31} \leq x \leq 2^{31} - 1$. If saturation occurs in either operation, it sets the Q flag in the APSR.

Restrictions

Do not use SP and do not use PC.

Condition Flags

If saturation occurs, these instructions set the Q flag to 1.

Examples

```
QDADD     R7, R4, R2   ; Doubles and saturates R4 to 32 bits, adds R2,
                       ; saturates to 32 bits, writes to R7
QDSUB     R0, R3, R5   ; Subtracts R3 doubled and saturated to 32 bits
                       ; from R5, saturates to 32 bits, writes to R0.
```

12.8.3.3 Interrupt Set-pending Registers

Name: NVIC_ISPRx [x=0..7]

Access: Read/Write

Reset: 0x000000000

31	30	29	28	27	26	25	24
SETPEND							

23	22	21	20	19	18	17	16
SETPEND							

15	14	13	12	11	10	9	8
SETPEND							

7	6	5	4	3	2	1	0
SETPEND							

These registers force interrupts into the pending state, and show which interrupts are pending.

- **SETPEND: Interrupt Set-pending**

Write:

0: No effect.

1: Changes the interrupt state to pending.

Read:

0: Interrupt is not pending.

1: Interrupt is pending.

Notes:
1. Writing a 1 to an ISPR bit corresponding to an interrupt that is pending has no effect.
2. Writing a 1 to an ISPR bit corresponding to a disabled interrupt sets the state of that interrupt to pending.

- **PENDSVCLR: PendSV Clear-pending**

Write:

0: No effect.

1: Removes the pending state from the PendSV exception.

- **PENDSTSET: SysTick Exception Set-pending**

Write:

0: No effect.

1: Changes SysTick exception state to pending.

Read:

0: SysTick exception is not pending.

1: SysTick exception is pending.

- **PENDSTCLR: SysTick Exception Clear-pending**

Write:

0: No effect.

1: Removes the pending state from the SysTick exception.

This bit is Write-only. On a register read, its value is Unknown.

- **ISRPENDING: Interrupt Pending Flag (Excluding NMI and Faults)**

0: Interrupt not pending.

1: Interrupt pending.

- **VECTPENDING: Exception Number of the Highest Priority Pending Enabled Exception**

0: No pending exceptions.

Nonzero: The exception number of the highest priority pending enabled exception.

The value indicated by this field includes the effect of the BASEPRI and FAULTMASK registers, but not any effect of the PRIMASK register.

- **RETTOBASE: Preempted Active Exceptions Present or Not**

0: There are preempted active exceptions to execute.

1: There are no active exceptions, or the currently-executing exception is the only active exception.

- **VECTACTIVE: Active Exception Number Contained**

0: Thread mode.

Nonzero: The exception number of the currently active exception. The value is the same as IPSR bits [8:0]. See “Interrupt Program Status Register” .

Subtract 16 from this value to obtain the IRQ number required to index into the Interrupt Clear-Enable, Set-Enable, Clear-Pending, Set-Pending, or Priority Registers, see “Interrupt Program Status Register” .

Note: When the user writes to the SCB_ICSR, the effect is unpredictable if:
- Writing a 1 to the PENDSVSET bit and writing a 1 to the PENDSVCLR bit
- Writing a 1 to the PENDSTSET bit and writing a 1 to the PENDSTCLR bit.

14.4 Functional Description

14.4.1 Reset Controller Overview

The Reset Controller is made up of an NRST manager and a reset state manager. It runs at slow clock and generates the following reset signals:

- proc_nreset: processor reset line (also resets the Watchdog Timer)
- periph_nreset: affects the whole set of embedded peripherals
- nrst_out: drives the NRST pin

These reset signals are asserted by the Reset Controller, either on events generated by peripherals, events on NRST pin, or on software action. The reset state manager controls the generation of reset signals and provides a signal to the NRST manager when an assertion of the NRST pin is required.

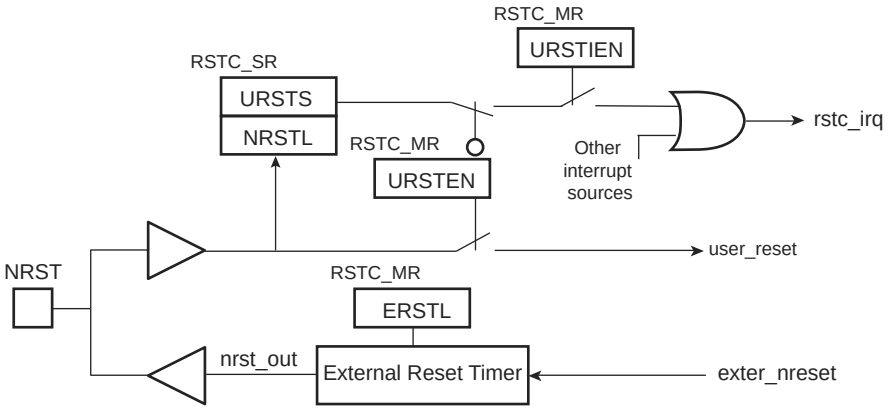
The NRST manager shapes the NRST assertion during a programmable time, thus controlling external device resets.

The Reset Controller Mode Register (RSTC_MR), used to configure the Reset Controller, is powered with VDDIO, so that its configuration is saved as long as VDDIO is on.

14.4.2 NRST Manager

The NRST manager samples the NRST input pin and drives this pin low when required by the reset state manager. Figure 14-2 shows the block diagram of the NRST manager.

Figure 14-2. NRST Manager



14.4.2.1 NRST Signal or Interrupt

The NRST manager samples the NRST pin at slow clock speed. When the line is detected low, a User Reset is reported to the reset state manager.

However, the NRST manager can be programmed to not trigger a reset when an assertion of NRST occurs. Writing a 0 to the URSTEN bit in the RSTC_MR disables the User Reset trigger.

The level of the pin NRST can be read at any time in the bit NRSTL (NRST level) in the Reset Controller Status Register (RSTC_SR). As soon as the NRST pin is asserted, bit URSTS in the RSTC_SR is set. This bit is cleared only when the RSTC_SR is read.

The Reset Controller can also be programmed to generate an interrupt instead of generating a reset. To do so, set the URSTIEN bit in the RSTC_MR.

16.6.8 RTC Status Clear Command Register

Name: RTC_SCCR

Address: 0x400E147C

Access: 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
–	–	TDERRCLR	CALCLR	TIMCLR	SECCLR	ALRCLR	ACKCLR

- **ACKCLR: Acknowledge Clear**

0: No effect.

1: Clears corresponding status flag in the Status Register (RTC_SR).

- **ALRCLR: Alarm Clear**

0: No effect.

1: Clears corresponding status flag in the Status Register (RTC_SR).

- **SECCLR: Second Clear**

0: No effect.

1: Clears corresponding status flag in the Status Register (RTC_SR).

- **TIMCLR: Time Clear**

0: No effect.

1: Clears corresponding status flag in the Status Register (RTC_SR).

- **CALCLR: Calendar Clear**

0: No effect.

1: Clears corresponding status flag in the Status Register (RTC_SR).

- **TDERRCLR: Time and/or Date Free Running Error Clear**

0: No effect.

1: Clears corresponding status flag in the Status Register (RTC_SR).

- **LCKDOWN: Lockdown Supported**

0: Lockdown is not supported.

1: Lockdown is supported.

- **CSIZE: Data Cache Size**

Value	Name	Description
0	CSIZE_1KB	Data cache size is 1 Kbyte
1	CSIZE_2KB	Data cache size is 2 Kbytes
2	CSIZE_4KB	Data cache size is 4 Kbytes
3	CSIZE_8KB	Data cache size is 8 Kbytes

- **CLSIZE: Cache LIne Size**

Value	Name	Description
0	CLSIZE_1KB	Cache line size is 4 bytes
1	CLSIZE_2KB	Cache line size is 8 bytes
2	CLSIZE_4KB	Cache line size is 16 bytes
3	CLSIZE_8KB	Cache line size is 32 bytes

associated interrupt source (MCKRDY) has been enabled in PMC_IER. PMC_MCKR must not be programmed in a single write operation. The programming sequence for PMC_MCKR is as follows:

- If a new value for CSS field corresponds to PLL clock,
  - Program the PRES field in PMC_MCKR.
  - Wait for the MCKRDY bit to be set in PMC_SR.
  - Program the CSS field in PMC_MCKR.
  - Wait for the MCKRDY bit to be set in PMC_SR.
- If a new value for CSS field corresponds to main clock or slow clock,
  - Program the CSS field in PMC_MCKR.
  - Wait for the MCKRDY bit to be set in the PMC_SR.
  - Program the PRES field in PMC_MCKR.
  - Wait for the MCKRDY bit to be set in PMC_SR.

If at some stage, parameters CSS or PRES are modified, the MCKRDY bit goes low to indicate that the master clock and the processor clock are not yet ready. The user must wait for MCKRDY bit to be set again before using the master and processor clocks.

Note: IF PLLx clock was selected as the master clock and the user decides to modify it by writing in CKGR_PLLxR, the MCKRDY flag will go low while PLLx is unlocked. Once PLLx is locked again, LOCKx goes high and MCKRDY is set. While PLLx is unlocked, the master clock selection is automatically changed to slow clock for PLLA and main clock for PLLB. For further information, see Section 29.15.2 ”Clock Switching Waveforms”.

Code Example:

```
write_register(PMC_MCKR,0x00000001)
wait (MCKRDY=1)
write_register(PMC_MCKR,0x00000011)
wait (MCKRDY=1)
```

The master clock is main clock divided by 2.

8. Select the programmable clocks

   Programmable clocks are controlled via registers, PMC_SCER, PMC_SCDR and PMC_SCSR.

   Programmable clocks can be enabled and/or disabled via PMC_SCER and PMC_SCDR. Three programmable clocks can be used. PMC_SCSR indicates which programmable clock is enabled. By default all programmable clocks are disabled.

   PMC_PCKx registers are used to configure programmable clocks.

   The CSS field is used to select the programmable clock divider source. Several clock options are available: main clock, slow clock, master clock, PLLACK, PLLBCK. The slow clock is the default clock source.

   The PRES field is used to control the programmable clock prescaler. It is possible to choose between different values (1, 2, 4, 8, 16, 32, 64). Programmable clock output is prescaler input divided by PRES parameter. By default, the PRES value is cleared which means that PCKx is equal to slow clock.

   Once PMC_PCKx register has been configured, the corresponding programmable clock must be enabled and the user is constrained to wait for the PCKRDYx bit to be set in the PMC_SR. This can be done either by polling PCKRDYx in PMC_SR or by waiting for the interrupt line to be raised if the associated interrupt source (PCKRDYx) has been enabled in PMC_IER. All parameters in PMC_PCKx can be programmed in a single write operation.

   If the CSS and PRES parameters are to be modified, the corresponding programmable clock must be disabled first. The parameters can then be modified. Once this has been done, the user must re-enable the programmable clock and wait for the PCKRDYx bit to be set.

9. Enable the peripheral clocks

29.17.10PMC Clock Generator PLLB Register

Name: CKGR_PLLBR

Address: 0x400E042C

Access: Read/Write

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

23	22	21	20	19	18	17	16
MULB							

15	14	13	12	11	10	9	8
–	–	PLLBCOUNT					

7	6	5	4	3	2	1	0
DIVB							

Possible limitations on PLLB input frequencies and multiplier factors should be checked before using the PMC.

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

- **DIVB: PLLB Front-End Divider**

0: Divider output is stuck at 0 and PLLB is disabled.

1: Divider is bypassed (divide by 1)

2–255: Clock is divided by DIVB

- **PLLBCOUNT: PLLB Counter**

Specifies the number of Slow Clock cycles before the LOCKB bit is set in PMC_SR after CKGR_PLLBR is written.

- **MULB: PLLB Multiplier**

0: The PLLB is deactivated (PLLB also disabled if DIVB = 0).

7 up to 62: The PLLB Clock frequency is the PLLB input frequency multiplied by MULB + 1.

Unlisted values are forbidden.

- **LLB: Local Loopback Enable**

0: Local loopback path disabled.

1: Local loopback path enabled.

LLB controls the local loopback on the data shift register for testing in Master mode only (MISO is internally connected on MOSI).

- **PCS: Peripheral Chip Select**

This field is only used if fixed peripheral select is active (PS = 0).

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.

- **DLYBCS: Delay Between Chip Selects**

This field defines the delay between the inactivation and the activation of 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 lower than 6, six peripheral clock periods are inserted by default.

Otherwise, the following equation determines the delay:

$$\text{Delay Between Chip Selects} = \frac{\text{DLYBCS}}{f_{\text{peripheral clock}}}$$

33.8.7 SPI Interrupt Disable Register

Name: SPI_IDR

Address: 0x40008018

Access: 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
–	–	–	–	–	UNDES	TXEMPTY	NSSR
7	6	5	4	3	2	1	0
TXBUFE	RXBUFF	ENDTX	ENDRX	OVRES	MODF	TDRE	RDRF

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

0: No effect.

1: Disables the corresponding interrupt.

- **RDRF: Receive Data Register Full Interrupt Disable**
- **TDRE: SPI Transmit Data Register Empty Interrupt Disable**
- **MODF: Mode Fault Error Interrupt Disable**
- **OVRES: Overrun Error Interrupt Disable**
- **ENDRX: End of Receive Buffer Interrupt Disable**
- **ENDTX: End of Transmit Buffer Interrupt Disable**
- **RXBUFF: Receive Buffer Full Interrupt Disable**
- **TXBUFE: Transmit Buffer Empty Interrupt Disable**
- **NSSR: NSS Rising Interrupt Disable**
- **TXEMPTY: Transmission Registers Empty Disable**
- **UNDES: Underrun Error Interrupt Disable**

34.8.8 TWI Interrupt Disable Register

Name: TWI_IDR

Address: 0x40018028 (0), 0x4001C028 (1)

Access: 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
TXBUFE	RXBUFF	ENDTX	ENDRX	EOSACC	SCL_WS	ARBLST	NACK

7	6	5	4	3	2	1	0
–	OVRE	GACC	SVACC	–	TXRDY	RXRDY	TXCOMP

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

0: No effect.

1: Disables the corresponding interrupt.

- **TXCOMP: Transmission Completed Interrupt Disable**
- **RXRDY: Receive Holding Register Ready Interrupt Disable**
- **TXRDY: Transmit Holding Register Ready Interrupt Disable**
- **SVACC: Slave Access Interrupt Disable**
- **GACC: General Call Access Interrupt Disable**
- **OVRE: Overrun Error Interrupt Disable**
- **NACK: Not Acknowledge Interrupt Disable**
- **ARBLST: Arbitration Lost Interrupt Disable**
- **SCL_WS: Clock Wait State Interrupt Disable**
- **EOSACC: End Of Slave Access Interrupt Disable**
- **ENDRX: End of Receive Buffer Interrupt Disable**
- **ENDTX: End of Transmit Buffer Interrupt Disable**
- **RXBUFF: Receive Buffer Full Interrupt Disable**
- **TXBUFE: Transmit Buffer Empty Interrupt Disable**

36.7.14 USART Transmit Holding Register

Name: US_THR

Address: 0x4002401C (0), 0x4002801C (1)

Access: 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
TXSYNH	–	–	–	–	–	–	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.

- **TXSYNH: Sync Field to be Transmitted**

0: The next character sent is encoded as a data. Start frame delimiter is DATA SYNC.

1: The next character sent is encoded as a command. Start frame delimiter is COMMAND SYNC.

38.10.1 Executing an ATA Polling Command

1. Issue READ_DMA_EXT with RW_MULTIPLE_REGISTER (CMD60) for 8 KB of DATA.
2. Read the ATA status register until DRQ is set.
3. Issue RW_MULTIPLE_BLOCK (CMD61) to transfer DATA.
4. Read the ATA status register until DRQ && BSY are configured to 0.

38.10.2 Executing an ATA Interrupt Command

1. Issue READ_DMA_EXT with RW_MULTIPLE_REGISTER (CMD60) for 8 KB of DATA with nIEN field set to zero to enable the command completion signal in the device.
2. Issue RW_MULTIPLE_BLOCK (CMD61) to transfer DATA.
3. Wait for Completion Signal Received Interrupt.

38.10.3 Aborting an ATA Command

If the host needs to abort an ATA command prior to the completion signal it must send a special command to avoid potential collision on the command line. The SPCMD field of the HSMCI_CMDR must be set to 3 to issue the CE-ATA completion Signal Disable Command.

38.10.4 CE-ATA Error Recovery

Several methods of ATA command failure may occur, including:

- No response to an MMC command, such as RW_MULTIPLE_REGISTER (CMD60).
- CRC is invalid for an MMC command or response.
- CRC16 is invalid for an MMC data packet.
- ATA Status register reflects an error by setting the ERR bit to one.
- The command completion signal does not arrive within a host specified time out period.

Error conditions are expected to happen infrequently. Thus, a robust error recovery mechanism may be used for each error event. The recommended error recovery procedure after a timeout is:

- Issue the command completion signal disable if nIEN was cleared to zero and the RW_MULTIPLE_BLOCK (CMD61) response has been received.
- Issue STOP_TRANSMISSION (CMD12) and successfully receive the R1 response.
- Issue a software reset to the CE-ATA device using FAST_IO (CMD39).

If STOP_TRANMISSION (CMD12) is successful, then the device is again ready for ATA commands. However, if the error recovery procedure does not work as expected or there is another timeout, the next step is to issue GO_IDLE_STATE (CMD0) to the device. GO_IDLE_STATE (CMD0) is a hard reset to the device and completely resets all device states.

Note that after issuing GO_IDLE_STATE (CMD0), all device initialization needs to be completed again. If the CE-ATA device completes all MMC commands correctly but fails the ATA command with the ERR bit set in the ATA Status register, no error recovery action is required. The ATA command itself failed implying that the device could not complete the action requested, however, there was no communication or protocol failure. After the device signals an error by setting the ERR bit to one in the ATA Status register, the host may attempt to retry the command.

38.11 HSMCI Boot Operation Mode

In boot operation mode, the processor can read boot data from the slave (MMC device) by keeping the CMD line low after power-on before issuing CMD1. The data can be read from either the boot area or user area, depending on register setting. As it is not possible to boot directly on SD-CARD, a preliminary boot code must be stored in internal Flash.

- **RTOE: Response Time-out Error Interrupt Enable**
- **DCRCE: Data CRC Error Interrupt Enable**
- **DTOE: Data Time-out Error Interrupt Enable**
- **CSTOE: Completion Signal Timeout Error Interrupt Enable**
- **FIFOEMPTY: FIFO empty Interrupt enable**
- **XFRDONE: Transfer Done Interrupt enable**
- **ACKRCV: Boot Acknowledge Interrupt Enable**
- **ACKRCVE: Boot Acknowledge Error Interrupt Enable**
- **OVRE: Overrun Interrupt Enable**
- **UNRE: Underrun Interrupt Enable**

39. Pulse Width Modulation Controller (PWM)

39.1 Description

The Pulse Width Modulation Controller (PWM) generates output pulses on 4 channels independently according to parameters defined per channel. Each channel controls two complementary square output waveforms. Characteristics of the output waveforms such as period, duty-cycle, polarity and dead-times (also called dead-bands or non-overlapping times) are configured through the user interface. Each channel selects and uses one of the clocks provided by the clock generator. The clock generator provides several clocks resulting from the division of the PWM peripheral clock.

All accesses to the PWM are made through registers mapped on the peripheral bus. All channels integrate a double buffering system in order to prevent an unexpected output waveform while modifying the period, the duty-cycle or the dead-times.

Channels can be linked together as synchronous channels to be able to update their duty-cycle or dead-times at the same time.

The update of duty-cycles of synchronous channels can be performed by the Peripheral DMA Controller channel which offers buffer transfer without processor Intervention.

The PWM provides 8 independent comparison units capable of comparing a programmed value to the counter of the synchronous channels (counter of channel 0). These comparisons are intended to generate software interrupts, to trigger pulses on the 2 independent event lines (in order to synchronize ADC conversions with a lot of flexibility independently of the PWM outputs) and to trigger Peripheral DMA Controllertransfer requests.

PWM outputs can be overridden synchronously or asynchronously to their channel counter.

The PWM provides a fault protection mechanism with 8 fault inputs**,** capable to detect a fault condition and to override the PWM outputs asynchronously (outputs forced to ‘0’, ‘1’).

For safety usage, some configuration registers are write-protected.

42.6.2 ADC Clock

The ADC uses the ADC clock (ADCCLK) to perform conversions. The ADC clock frequency is selected in the PRESCAL field of ADC_MR.

The ADC clock frequency is between $f_{peripheral\ clock}/2$, if PRESCAL is 0, and $f_{peripheral\ clock}/512$, if PRESCAL is set to 255 (0xFF).

PRESCAL must be programmed to provide the ADC clock frequency parameter given in the section ‘Electrical Characteristics’.

42.6.3 ADC Reference Voltage

The conversion is performed on a full range between 0V and the reference voltage pin ADVREF. Analog inputs between these voltages convert to values based on a linear conversion.

42.6.4 Conversion Resolution

The ADC analog cell features12-bit resolution.

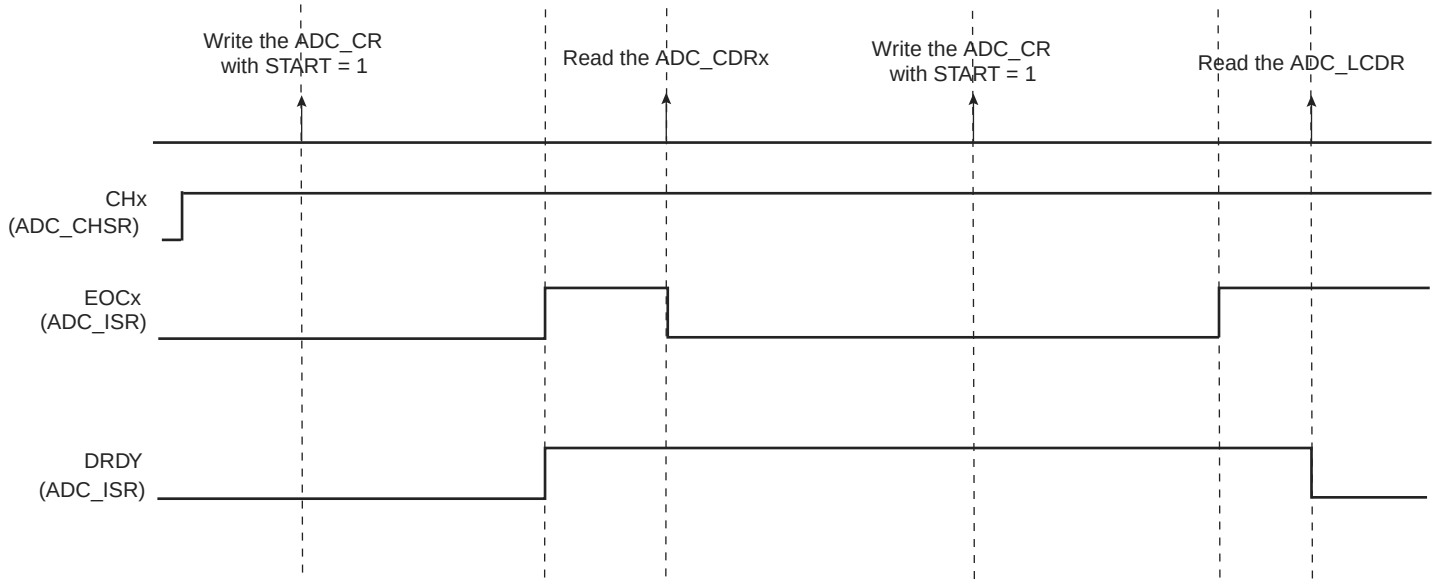
42.6.5 Conversion Results

When a conversion is completed, the resulting digital value is stored in the Channel Data register (ADC_CDRx) of the current channel and in the ADC Last Converted Data register (ADC_LCDR). By setting the TAG option in the Extended Mode Register (ADC_EMR), the ADC_LCDR presents the channel number associated with the last converted data in the CHNB field.

The channel EOC bit and the DRDY bit in the Interrupt Status register (ADC_ISR) are set. In the case of a connected PDC channel, DRDY rising triggers a data request. In any case, either EOC and DRDY can trigger an interrupt.

Reading one of the ADC_CDRx clears the corresponding EOC bit. Reading ADC_LCDR clears the DRDY bit.

Figure 42-4. EOCx and DRDY Flag Behavior



If ADC_CDR is not read before further incoming data is converted, the corresponding OVREx flag is set in the Overrun Status register (ADC_OVER).

New data converted when DRDY is high sets the GOVRE bit in ADC_ISR.

Table 49-3. SAM4S Datasheet Rev. 11100I Revision History (Continued)

Doc. Date	Changes
03-Apr-15	**Section 18., “Supply Controller (SUPC)”** Updated Figure 18-1 “Supply Controller Block Diagram”. Section 18.4.2, “Slow Clock Generator”: modified last paragraph with information on entering Bypass mode. Modfied Section 18.4.7.1, “Wake-up Inputs”, Section 18.5.5, “Supply Controller Mode Register”, Section 18.5.7, “Supply Controller Wake-up Inputs Register” and Section 18.5.8, “Supply Controller Status Register” Section 18.4.7.2, “Low-power Tamper Detection and Anti-Tampering”: corrected ‘...LPDBCCLR bit must be set in SUPC_MR’ to ‘...LPDBCCLR bit must be set in SUPC_WUMR’Updated Figure 18-4 “Wake-up Sources” Section 18.5.2, “Supply Controller (SUPC) User Interface”: corrected reset value for SUPC_MR.
	Section 19., “General Purpose Backup Registers (GPBR)” Modified Section 19.1, “Description” and Section 19.2, “Embedded Characteristics” Table 19-1 “Register Mapping”: added reset value 0x00000000 for all registers SYS_GPBRx Section 19.3.1, “General Purpose Backup Register x”: inserted sentence “These registers are reset at first power-up and on each loss of VVDIO”
	Section 20., “Enhanced Embedded Flash Controller (EEFC)”. Updated Section 20.2, “Embedded Characteristics” Added Figure 20-1 Flash Memory Areas Section 20.3.2, “Interrupt Sources”: changed last sentence Section 20.4.1, “Embedded Flash Organization”: corrected instance of command name “Get descriptor” to “Get Flash Descriptor” Figure 20-7 Command State Chart: replaced two instances of “MC_FSR” with “EEFC_FSR” Modified Section 20.4.3.2, “Write Commands”, Section 20.4.3.3, “Erase Commands”, Section 20.4.3.8, “Unique Identifier Area” and Section 20.4.3.9, “User Signature Area” Section 20.5, “Enhanced Embedded Flash Controller (EEFC) User Interface”: deleted address offsets from individual register descriptions (offsets are provided in Section 20-6, “Register Mapping”) Modified Section 20.5.1, “EEFC Flash Mode Register”, Section 20.5.2, “EEFC Flash Command Register” and Section 20.5.3, “EEFC Flash Status Register”: added new field ‘ZERO’ (register bits 26:25) Updated Section 20-6, “Register Mapping”
	Section 21., “Fast Flash Programming Interface (FFPI)” Section 21-1, “16-bit Parallel Programming Interface”: renamed figure
	Section 22., “Cortex-M Cache Controller (CMCC)” Section 22.4.3, “Cache Performance Monitoring”: corrected “MODE field of the CMCC_CFG register” to “MODE field of the CMCC_MCFG register” Table 22-1 "Register Mapping": removed CMCC_CTRL reset value for this write-only register; replaced “Cache ...” with “Cache Controller ...” in “Register” column contents Updated Section 22.5.1, “Cache Controller Type Register” Updated bit descriptions in Section 22.5.3, “Cache Controller Control Register”, Section 22.5.4, “Cache Controller Status Register”, Section 22.5.5, “Cache Controller Maintenance Register 0”, Section 22.5.8, “Cache Controller Monitor Enable Register” and Section 22.5.9, “Cache Controller Monitor Control Register”. Section 22.5.7, “Cache Controller Monitor Configuration Register”: changed access from Write-only to Read/Write. Section 22.5.8, “Cache Controller Monitor Enable Register”: removed reset value from several write-only registers (reset values are found in Table 22-1 "Register Mapping") and changed access from Write-only to Read/Write.

Table 49-5. SAM4S Datasheet Rev. 11100G Revision History

Doc. Date	Changes
27-May-14	Table 3-1 ”Signal Description List”: WKUP[15:0] voltage reference type added.
	In Figure 5-4 “Backup Battery”, modified ADC, DAC, Analog Comparator Supply from 2.0V to 2.4V
	Modified Section 6.5 “ERASE Pin”.
	Modified bullet list on use of erase commands depending on sector size in Section 8.1.3.1 “Flash Overview”
	Modified Section 8.1.3.5 “Security Bit”, Section 8.1.3.11 “GPNVM Bits” and Section 8.1.4 “Boot Strategies”.
	Section 24. “Boot Program” Section 24.5.4 “In Application Programming (IAP) Feature”: 5th sentence: added “...the EFC number...”
	Section 29. “Power Management Controller (PMC)” Section 29.17.9 “PMC Clock Generator PLLA Register”: Min value for bit MULA corrected to 4 from 7. Section 29.17.10 “PMC Clock Generator PLLB Register”: Min value for bit MULB corrected to 4 from 1.
	Section 44. “Electrical Characteristics” Added Table 44-24 ”Typical Power Consumption on VDDCORE (VDDIO = 3.3V, TA = 25°C)”. Table 44-73 ”AC Flash Characteristics”: Added parameter Erase Pin Assertion Time.
	Section 48. “Errata” Added Section Issue: and Section Issue: “Incorrect Flash Read May Occur Depending on VDDIO Voltage and Flash Wait State”.

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

Doc. Date	Changes
	Section 12. "ARM Cortex-M4 Processor" Section 12.5.3 "Power Management Programming Hints": in 2nd instruction line, replaced "WFE(void)" with "WFI(void)" to match 'Wait For Interrupt' and in 2nd instruction line, replaced "WFE(void)" with "WFI(void)" to match 'Wait For Interrupt' Section 12.9.1.2 "CPUID Base Register": updated 'Constant' field description Section 12.9.1.5 "Application Interrupt and Reset Control Register": updated 'VECTCLRACTIVE' and 'VECTRESET' field descriptions Section 12.9.1.7 "Configuration and Control Register": updated 'USERSETMPEND' field description Section 12.9.1.16 "MemManage Fault Address Register": updated 'ADDRESS' field description Section 12.9.1.17 "Bus Fault Address Register": updated 'ADDRESS' field description Section 12.10.1.1 "SysTick Control and Status": updated 'TICKINT' and 'ENABLE' field descriptions Section 12.10.1.2 "SysTick Reload Value Registers": updated 'RELOAD' field description Section 12.10.1.3 "SysTick Current Value Register": updated 'CURRENT' field description Section 12.10.1.4 "SysTick Calibration Value Register": updated register reset value; updated 'TENMS' and 'SKEW' field descriptions. Section 12.11.2.2 "MPU Control Register": updated 'ENABLE' field description. Section 12.11.2.3 "MPU Region Number Register": updated 'REGION' field description. Updated Section 12.11.2.4 "MPU Region Base Address Register". Added Section 12.11.2.6 "MPU Region Base Address Register Alias 1"to Section 12.11.2.11 "MPU Region Attribute and Size Register Alias 3". Corrected "Sterling Pound" symbol (£) to "less than or equal to" symbol (≤) in operation description in Section 12.6.7 "Saturating Instructions". Table 12-30 "Mapping of Interrupts to the Interrupt Variables": updated count range in "Interrupts" column.
	Section 14. "Reset Controller (RSTC)" Figure 14-3 "General Reset State": replaced "backup_nreset" with "vddbu_nreset". Section 14.4.2.2 "NRST External Reset Control": replaced "ext_nreset" with "exter_nreset". Section 14.4.4.2 "Backup Reset": replaced "core_backup_reset" with "vddcore_nreset"; reworded content to improve comprehension. RSTTYP information corrected in Section 14.4.6 "Reset Controller Status Register". Section 14.5.1 "Reset Controller Control Register": updated EXTRST value 1 description (deleted phrase "and resets the processor and the peripherals"). Section 14.5.3 "Reset Controller Mode Register": inserted sentence "This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register (SYSC_WPMR)."
	Section 15. "Real-time Timer (RTT)" Figure 15-1 "Real-time Timer": replaced "16-bit Divider" with "16-bit Prescaler. Revised Section 15.4 "Functional Description". Section 15.5.4 "Real-time Timer Status Register": updated RTTINC bit description.