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	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 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/atsam4sd32cb-anr

12.6.5.22 USAD8

Unsigned Sum of Absolute Differences

Syntax

```
USAD8{cond}{Rd,} Rn, Rm
```

where:

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

Rd is the destination register.

Rn is the first operand register.

Rm is the second operand register.

Operation

The USAD8 instruction:

1. Subtracts each byte of the second operand register from the corresponding byte of the first operand register.
2. Adds the absolute values of the differences together.
3. Writes the result to the destination register.

Restrictions

Do not use SP and do not use PC.

Condition Flags

These instructions do not change the flags.

Examples

```
USAD8 R1, R4, R0 ; Subtracts each byte in R0 from corresponding byte of R4
                 ; adds the differences and writes to R1
USAD8 R0, R5     ; Subtracts bytes of R5 from corresponding byte in R0,
                 ; adds the differences and writes to R0.
```

op is one of:

QADD Saturating 32-bit add.

QADD8 Saturating four 8-bit integer additions.

QADD16 Saturating two 16-bit integer additions.

QSUB Saturating 32-bit subtraction.

QSUB8 Saturating four 8-bit integer subtraction.

QSUB16 Saturating two 16-bit integer subtraction.

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

These instructions add or subtract two, four or eight values from the first and second operands and then writes a signed saturated value in the destination register.

The QADD and QSUB instructions apply the specified add or subtract, and then saturate the result to the signed range $-2^{n-1} \leq x \leq 2^{n-1}-1$, where x is given by the number of bits applied in the instruction, 32, 16 or 8.

If the returned result is different from the value to be saturated, it is called *saturation*. If saturation occurs, the QADD and QSUB instructions set the Q flag to 1 in the APSR. Otherwise, it leaves the Q flag unchanged. The 8-bit and 16-bit QADD and QSUB instructions always leave the Q flag unchanged.

To clear the Q flag to 0, the MSR instruction must be used; see “MSR” .

To read the state of the Q flag, the MRS instruction must be used; see “MRS” .

Restrictions

Do not use SP and do not use PC.

Condition Flags

These instructions do not affect the condition code flags.

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

```
STM R0, {R1-R2}     ; Region base address, region number and VALID bit,
                    ; and Region Attribute, Size and Enable
```

12.11.1.5 Subregions

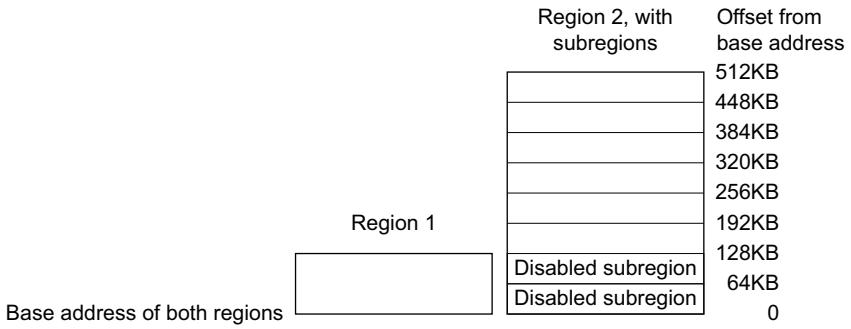
Regions of 256 bytes or more are divided into eight equal-sized subregions. Set the corresponding bit in the SRD field of the MPU_RASR field to disable a subregion. See “MPU Region Attribute and Size Register” . The least significant bit of SRD controls the first subregion, and the most significant bit controls the last subregion. Disabling a subregion means another region overlapping the disabled range matches instead. If no other enabled region overlaps the disabled subregion, the MPU issues a fault.

Regions of 32, 64, and 128 bytes do not support subregions. With regions of these sizes, the SRD field must be set to 0x00, otherwise the MPU behavior is unpredictable.

12.11.1.6 Example of SRD Use

Two regions with the same base address overlap. Region 1 is 128 KB, and region 2 is 512 KB. To ensure the attributes from region 1 apply to the first 128 KB region, set the SRD field for region 2 to b00000011 to disable the first two subregions, as in Figure 12-13 below:

Figure 12-13. SRD Use



12.11.1.7 MPU Design Hints And Tips

To avoid unexpected behavior, disable the interrupts before updating the attributes of a region that the interrupt handlers might access.

Ensure the software uses aligned accesses of the correct size to access MPU registers:

- Except for the MPU_RASR, it must use aligned word accesses
- For the MPU_RASR, it can use byte or aligned halfword or word accesses.

The processor does not support unaligned accesses to MPU registers.

When setting up the MPU, and if the MPU has previously been programmed, disable unused regions to prevent any previous region settings from affecting the new MPU setup.

MPU Configuration for a Microcontroller

Usually, a microcontroller system has only a single processor and no caches. In such a system, program the MPU as follows:

Table 12-39. Memory Region Attributes for a Microcontroller

Memory Region	TEX	C	B	S	Memory Type and Attributes
Flash memory	b000	1	0	0	Normal memory, non-shareable, write-through
Internal SRAM	b000	1	0	1	Normal memory, shareable, write-through
External SRAM	b000	1	1	1	Normal memory, shareable, write-back, write-allocate
Peripherals	b000	0	1	1	Device memory, shareable

- **OSCSEL: 32-kHz Oscillator Selection Status**

0 (RC): The slow clock, SLCK, is generated by the embedded 32 kHz RC oscillator.

1 (CRYST): The slow clock, SLCK, is generated by the 32 kHz crystal oscillator.

- **LPDBCS0: Low-power Debouncer Wake-up Status on WKUP0 (cleared on read)**

0 (NO): No wake-up due to the assertion of the WKUP0 pin has occurred since the last read of SUPC_SR.

1 (PRESENT): At least one wake-up due to the assertion of the WKUP0 pin has occurred since the last read of SUPC_SR.

- **LPDBCS1: Low-power Debouncer Wake-up Status on WKUP1 (cleared on read)**

0 (NO): No wake-up due to the assertion of the WKUP1 pin has occurred since the last read of SUPC_SR.

1 (PRESENT): At least one wake-up due to the assertion of the WKUP1 pin has occurred since the last read of SUPC_SR.

- **WKUPISx: WKUPx Input Status (cleared on read)**

0 (DIS): The corresponding wake-up input is disabled, or was inactive at the time the debouncer triggered a wake-up event.

1 (EN): The corresponding wake-up input was active at the time the debouncer triggered a wake-up event since the last read of SUPC_SR.

24.5.1 UART0 Serial Port

Communication is performed through the UART0 initialized to 115200 Baud, 8, n, 1.

The Send and Receive File commands use the Xmodem protocol to communicate. Any terminal performing this protocol can be used to send the application file to the target. The size of the binary file to send depends on the SRAM size embedded in the product. In all cases, the size of the binary file must be lower than the SRAM size because the Xmodem protocol requires some SRAM memory to work. See Section 24.2 ”Hardware and Software Constraints”.

24.5.2 Xmodem Protocol

The Xmodem protocol supported is the 128-byte length block. This protocol uses a two-character CRC-16 to guarantee detection of a maximum bit error.

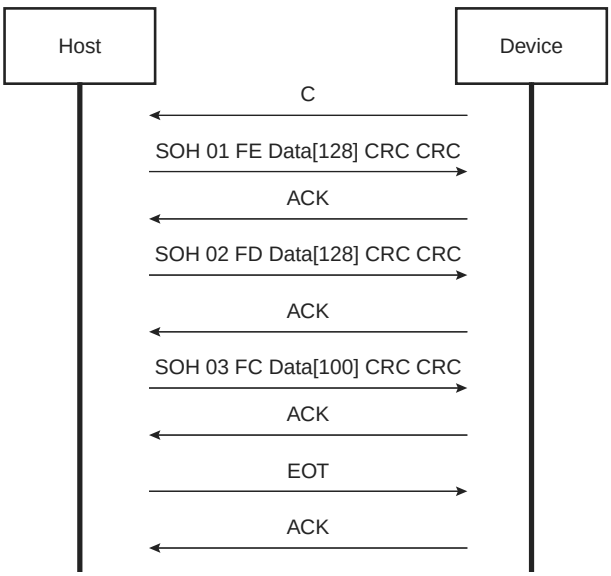
Xmodem protocol with CRC is accurate provided both sender and receiver report successful transmission. Each block of the transfer looks like:

<SOH><blk #><255-blk #><--128 data bytes--><checksum> in which:

- <SOH> = 01 hex
- <blk #> = binary number, starts at 01, increments by 1, and wraps 0FFH to 00H (not to 01)
- <255-blk #> = 1’s complement of the blk#.
- <checksum> = 2 bytes CRC16

Figure 24-2 shows a transmission using this protocol.

Figure 24-2. Xmodem Transfer Example

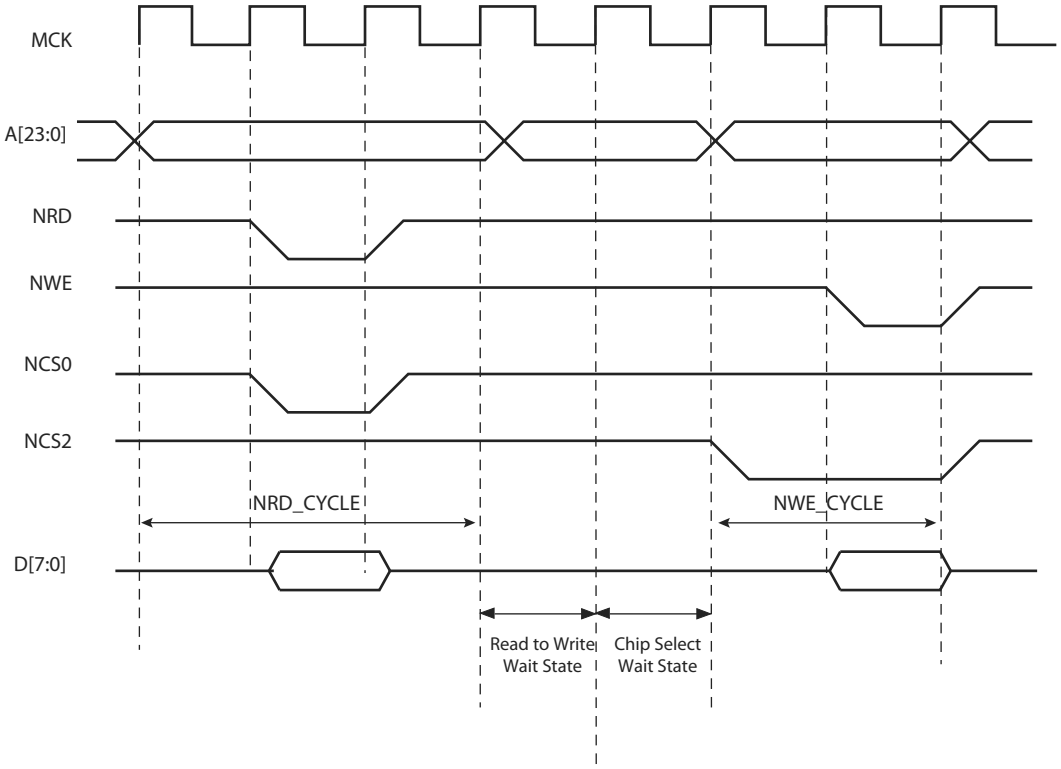


24.5.3 USB Device Port

The device uses the USB communication device class (CDC) drivers to take advantage of the installed PC RS-232 software to talk over the USB. The CDC class is implemented in all releases of Windows®, beginning with Windows 98 SE. The CDC document, available at www.usb.org, describes a way to implement devices such as ISDN modems and virtual COM ports.

The Vendor ID (VID) is Atmel’s vendor ID 0x03EB. The product ID (PID) is 0x6124. These references are used by the host operating system to mount the correct driver. On Windows systems, the INF files contain the correspondence between vendor ID and product ID.

Figure 26-13. Chip Select Wait State between a Read Access on NCS0 and a Write Access on NCS2



26.11.2 Early Read Wait State

In some cases, the SMC inserts a wait state cycle between a write access and a read access to allow time for the write cycle to end before the subsequent read cycle begins. This wait state is not generated in addition to a chip select wait state. The early read cycle thus only occurs between a write and read access to the same memory device (same chip select).

An early read wait state is automatically inserted if at least one of the following conditions is valid:

- if the write controlling signal has no hold time and the read controlling signal has no setup time (Figure 26-14).
- in NCS Write controlled mode (WRITE_MODE = 0), if there is no hold timing on the NCS signal and the NCS_RD_SETUP parameter is set to 0, regardless of the Read mode (Figure 26-15). The write operation must end with a NCS rising edge. Without an Early Read Wait State, the write operation could not complete properly.
- in NWE controlled mode (WRITE_MODE = 1) and if there is no hold timing (NWE_HOLD = 0), the feedback of the write control signal is used to control address, data, and chip select lines. If the external write control signal is not inactivated as expected due to load capacitances, an Early Read Wait State is inserted and address, data and control signals are maintained one more cycle. See Figure 26-16.

29.17.6 PMC Peripheral Clock Status Register 0

Name: PMC_PCSR0

Address: 0x400E0418

Access: Read-only

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

- **PIDx: Peripheral Clock x Status**

0: The corresponding peripheral clock is disabled.

1: The corresponding peripheral clock is enabled.

Note: PIDx refers to identifiers defined in the section “Peripheral Identifiers”. Other peripherals status can be read in PMC_PCSR1 (Section 29.17.25 ”PMC Peripheral Clock Status Register 1”).

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.

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).

36.5 Product Dependencies

36.5.1 I/O Lines

The pins used for interfacing the USART may be multiplexed with the PIO lines. The programmer must first program the PIO controller to assign the desired USART pins to their peripheral function. If I/O lines of the USART are not used by the application, they can be used for other purposes by the PIO Controller.

To prevent the TXD line from falling when the USART is disabled, the use of an internal pull up is mandatory. If the hardware handshaking feature or Modem mode is used, the internal pull up on TXD must also be enabled.

All the pins of the modems may or may not be implemented on the USART. Only USART1 fully equipped with all the modem signals. On USARTs not equipped with the corresponding pin, the associated control bits and statuses have no effect on the behavior of the USART.

Table 36-2. I/O Lines

Instance	Signal	I/O Line	Peripheral
USART0	CTS0	PA8	A
USART0	RTS0	PA7	A
USART0	RXD0	PA5	A
USART0	SCK0	PA2	B
USART0	TXD0	PA6	A
USART1	CTS1	PA25	A
USART1	DCD1	PA26	A
USART1	DSR1	PA28	A
USART1	DTR1	PA27	A
USART1	RI1	PA29	A
USART1	RTS1	PA24	A
USART1	RXD1	PA21	A
USART1	SCK1	PA23	A
USART1	TXD1	PA22	A

36.5.2 Power Management

The USART is not continuously clocked. The programmer must first enable the USART clock in the Power Management Controller (PMC) before using the USART. However, if the application does not require USART operations, the USART clock can be stopped when not needed and be restarted later. In this case, the USART will resume its operations where it left off.

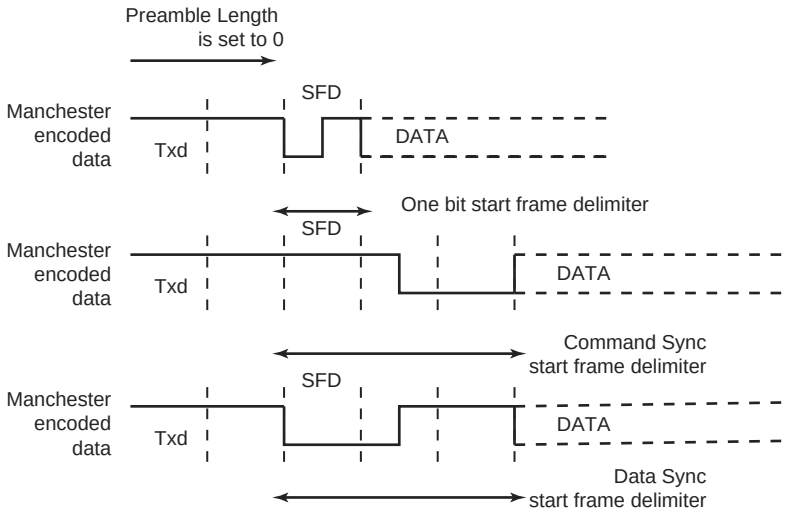
36.5.3 Interrupt Sources

The USART interrupt line is connected on one of the internal sources of the Interrupt Controller. Using the USART interrupt requires the Interrupt Controller to be programmed first.

Table 36-3. Peripheral IDs

Instance	ID
USART0	14
USART1	15

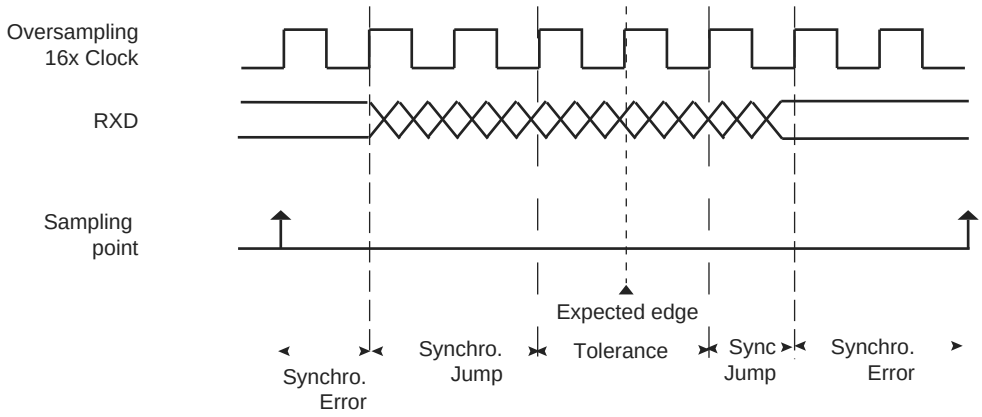
Figure 36-9. Start Frame Delimiter



Drift Compensation

Drift compensation is available only in 16X oversampling mode. An hardware recovery system allows a larger clock drift. To enable the hardware system, the bit in the USART_MAN register must be set. If the RXD edge is one 16X clock cycle from the expected edge, this is considered as normal jitter and no corrective actions is taken. If the RXD event is between 4 and 2 clock cycles before the expected edge, then the current period is shortened by one clock cycle. If the RXD event is between 2 and 3 clock cycles after the expected edge, then the current period is lengthened by one clock cycle. These intervals are considered to be drift and so corrective actions are automatically taken.

Figure 36-10. Bit Resynchronization



36.6.3.3 Asynchronous Receiver

If the USART is programmed in Asynchronous operating mode (SYNC = 0), the receiver oversamples the RXD input line. The oversampling is either 16 or 8 times the baud rate clock, depending on the OVER bit in the US_MR.

The receiver samples the RXD line. If the line is sampled during one half of a bit time to 0, a start bit is detected and data, parity and stop bits are successively sampled on the bit rate clock.

If the oversampling is 16 (OVER = 0), a start is detected at the eighth sample to 0. Data bits, parity bit and stop bit are assumed to have a duration corresponding to 16 oversampling clock cycles. If the oversampling is 8 (OVER = 1), a start bit is detected at the fourth sample to 0. Data bits, parity bit and stop bit are assumed to have a duration corresponding to 8 oversampling clock cycles.

- **DSRIC: Data Set Ready Input Change Mask**
- **DCDIC: Data Carrier Detect Input Change Interrupt Mask**
- **CTSIC: Clear to Send Input Change Interrupt Mask**
- **MANE: Manchester Error Interrupt Mask**

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.

39.7.28 PWM Event Line x Register

Name: PWM_ELMRx

Address: 0x4002007C

Access: Read/Write

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
CSEL7	CSEL6	CSEL5	CSEL4	CSEL3	CSEL2	CSEL1	CSEL0

- **CSELy: Comparison y Selection**

0: A pulse is not generated on the event line x when the comparison y matches.

1: A pulse is generated on the event line x when the comparison y match.

42.6.11 ADC Timings

Each ADC has its own minimal startup time that is programmed through the field STARTUP in ADC_MR.

A minimal tracking time is necessary for the ADC to guarantee the best converted final value between two channel selections. This time must be programmed in the TRACKTIM field in ADC_MR.

When the gain, offset or differential input parameters of the analog cell change between two channels, the analog cell may need a specific settling time before starting the tracking phase. In that case, the controller automatically waits during the settling time defined in ADC_MR. Obviously, if the ANACH option is not set, this time is unused.

Warning: No input buffer amplifier to isolate the source is included in the ADC. This must be taken into consideration to program a precise value in the TRACKTIM field. See the section 'ADC Characteristics' in the 'Electrical Characteristics'.

42.6.12 Automatic Calibration

The ADC features an Automatic calibration (AUTOCALIB) mode for gain errors (calibration).

The automatic calibration sequence can be started at any time by writing a '1' to the AUTOCAL bit of ADC_CR. The automatic calibration sequence requires a software reset command (SWRST in ADC_CR) prior to writing the AUTOCAL bit. The end of calibration sequence is given by the EOCAL bit in ADC_ISR, and an interrupt is generated if EOCAL interrupt has been enabled (ADC_IER).

The calibration sequence performs an automatic calibration on all enabled channels. The channels required for conversion do not need to be all enabled during the calibration process if they are programmed with the same gain. Only channels with different gain settings need to be enabled. The gain settings of all enabled channels must be set before starting the AUTOCALIB sequence. If the gain settings (ADC_CGR and ADC_COR) for a given channel are changed, the AUTOCALIB sequence must then be started again.

The calibration data on one or more enabled channels is stored in the internal ADC memory.

Then, when a new conversion is started on one or more enabled channels, the converted value (in ADC_LCDR or ADC_CDRx) is a calibrated value.

Autocalibration is for settings, not for channels. Therefore, if a specific combination of gain has already been calibrated and a new channel with the same settings is enabled after the initial calibration, there is no need to restart a calibration. If different enabled channels have different gain settings, the corresponding channels must be enabled before starting the calibration.

If a software reset is performed (bit SWRST = 1 in ADC_CR) or after power-up (or wake-up from Backup mode), the calibration data in the ADC memory is lost.

Changing the ADC reference voltage (ADVREF pin) requires a new calibration sequence.

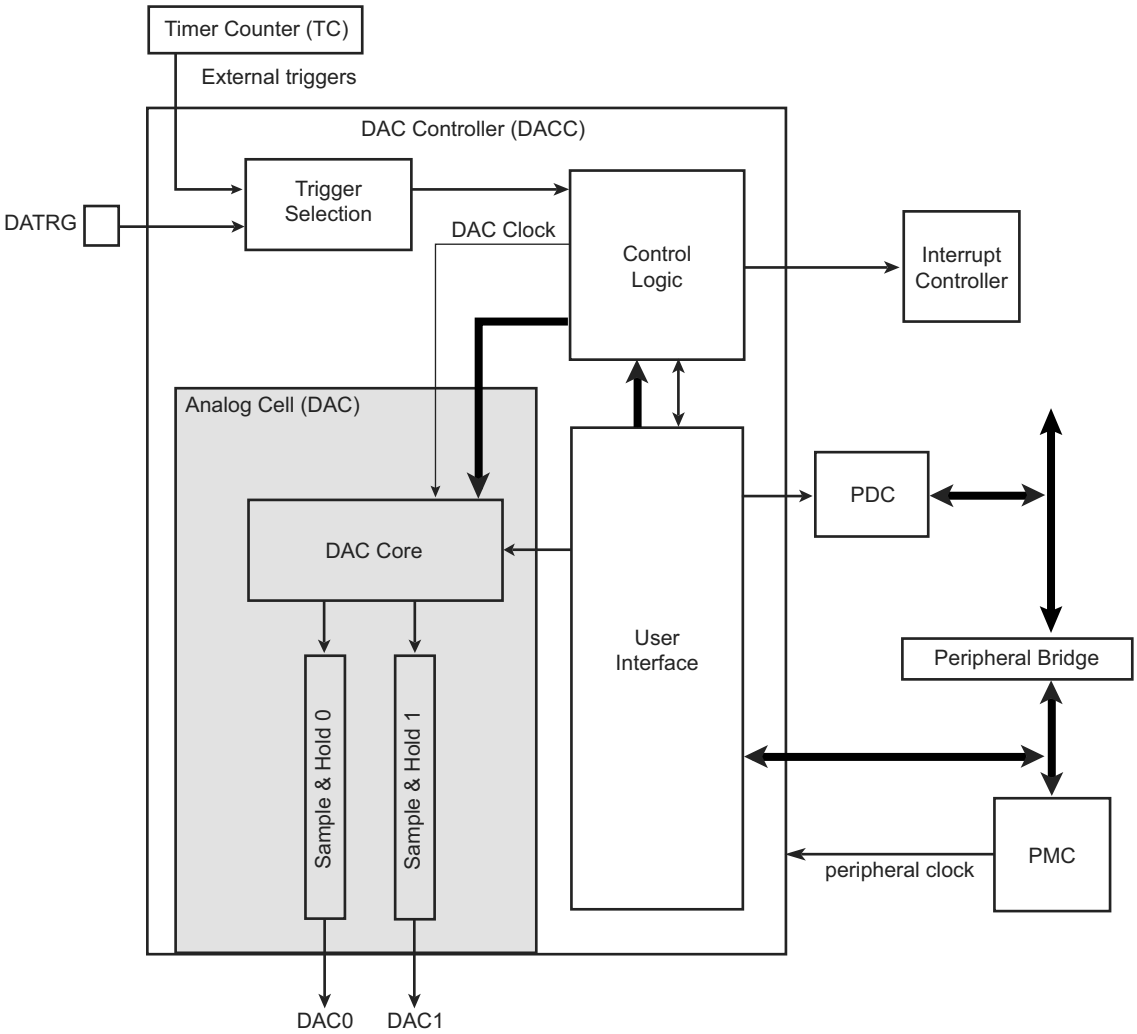
For calibration time and gain error after calibration, refer to the section on the 12-bit ADC in the Electrical Characteristics.

42.6.13 Buffer Structure

The PDC read channel is triggered each time a new data is stored in ADC_LCDR. The same structure of data is repeatedly stored in ADC_LCDR each time a trigger event occurs. Depending on user mode of operation (ADC_MR, ADC_CHSR, ADC_SEQR1, ADC_SEQR2) the structure differs. Each data read to PDC buffer, carried on a half-word (16-bit), consists of last converted data right aligned and when TAG is set in ADC_EMR, the four most significant bits are carrying the channel number thus allowing an easier post-processing in the PDC buffer or better checking the PDC buffer integrity.

43.3 Block Diagram

Figure 43-1. DACC Block Diagram



43.4 Signal Description

Table 43-1. DACC Pin Description

Pin Name	Description
DAC0–DAC1	Analog output channels
DATRG	External triggers

43.5 Product Dependencies

43.5.1 Power Management

The user must first enable the DAC Controller Clock in the Power Management Controller (PMC) before using the DACC.

The DACC becomes active as soon as a conversion is requested and at least one channel is enabled. The DACC is automatically deactivated when no channels are enabled.

43.7.4 DACC Channel Disable Register

Name: DACC_CHDR

Address: 0x4003C014

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
–	–	–	–	–	–	CH1	CH0

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

- **CHx: Channel x Disable**

0: No effect

1: Disables the corresponding channel

Warning: If the corresponding channel is disabled during a conversion, or disabled then re-enabled during a conversion, the associated analog value and the corresponding EOC flags in the DACC_ISR are unpredictable.

Table 44-24. SAM4SD32/SA16/SD16 Typical Active Power Consumption with VDDCORE@ 1.2V running from Flash Memory (AMP2) or SRAM

	CoreMark					
	Cache Enable (CE)		Cache Disable (CD)			
Core Clock (MHz)	128-bit Flash access(1)	64-bit Flash access(1)	128-bit Flash access(1)	64-bit Flash access(1)	SRAM	Unit
120	23.2	23.2	27.8	20.9	22.1	mA
100	19.6	19.6	25.3	19.0	18.5	
84	16.6	16.5	21.6	16.2	15.7	
64	12.8	12.8	18.0	13.7	12.1	
48	9.7	9.7	14.9	11.9	9.2	
32	6.7	6.7	11.2	9.5	6.3	
24	5.2	5.2	9.5	8.4	4.9	
12	2.5	2.5	5.4	4.6	2.4	
8	1.8	1.8	4.5	3.9	1.7	
4	1.1	1.1	2.8	2.8	1.0	
2	0.7	0.7	2.0	2.0	0.7	
1	0.5	0.5	1.2	1.2	0.5	
0.5	0.4	0.4	0.8	0.8	0.4	

Note: 1. Flash Wait State (FWS) in EEFC_FMR adjusted versus core frequency

44.8.4 ADC Transfer Function

The first operation of the ADC is a sampling function relative to a common mode voltage. The common mode voltage (V_{CM}) is equal to $V_{ADVREF}/2$ when the bits OFFx = 1, in Differential and in Single-ended mode. When the bits OFFx = 0, sampling is done versus $V_{ADVREF}/4$ for gain = 2, and $V_{ADVREF}/8$ for gain = 4, in Single-ended mode only.

The code in ADC_CDRx is a 12-bit positive integer.

44.8.4.1 Differential Mode

A differential input voltage $V_I = V_{I+} - V_{I-}$ can be applied between two selected differential pins, e.g., AD0 and AD1. The ideal code Ci is calculated by using the following formula and rounding the result to the nearest positive integer.

$$Ci = \frac{4096}{V_{ADVREF}} \times V_I \times Gain + 2047$$

Table 44-42 is a computation example for the above formula, where V_{ADVREF} = 3V.

Table 44-42. Input Voltage Values in Differential Mode

Ci	Gain = 0.5	Gain = 1	Gain = 2
0	-3	-1.5	-0.75
2047	0	0	0
4095	3	1.5	0.75

44.8.4.2 Single-ended Mode

A single input voltage V_I can be applied to selected pins, e.g., AD0 or AD1. The ideal code Ci is calculated by using the following formula and rounding the result to the nearest positive integer.

The single-ended ideal code conversion formula for OFFx = 1 is:

$$Ci = \frac{4096}{V_{ADVREF}} \times \left(V_I - \frac{V_{ADVREF}}{2}\right) \times Gain + 2047$$

Table 44-43 is a computation example for the above formula, where V_{ADVREF} = 3V.

Table 44-43. Input Voltage Values in Single-ended Mode, OFFx = 1

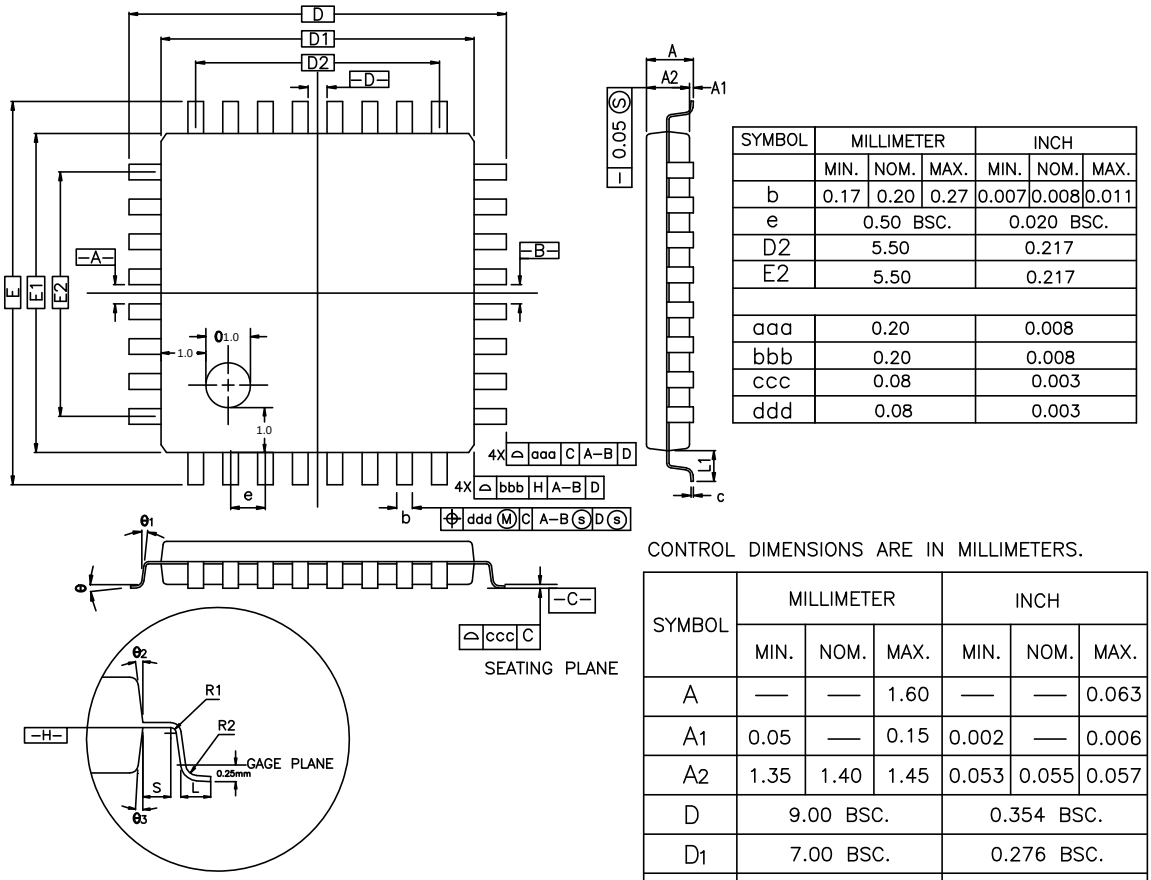
Ci	Gain = 1	Gain = 2	Gain = 4
0	0	0.75	1.125
2047	1.5	1.5	1.5
4095	3	2.25	1.875

The single-ended ideal code conversion formula for OFFx = 0 is:

$$Ci = V_I \times Gain \times \frac{4096}{V_{ADVREF}} - 1$$

45.7 48-lead LQFP Mechanical Characteristics

Figure 45-9. 48-lead LQFP Package Drawing



SYMBOL	MILLIMETER			INCH		
	MIN.	NOM.	MAX.	MIN.	NOM.	MAX.
b	0.17	0.20	0.27	0.007	0.008	0.011
e		0.50 BSC.			0.020 BSC.	
D2		5.50			0.217	
E2		5.50			0.217	
aaa		0.20			0.008	
bbb		0.20			0.008	
ccc		0.08			0.003	
ddd		0.08			0.003	

CONTROL DIMENSIONS ARE IN MILLIMETERS.

SYMBOL	MILLIMETER			INCH		
	MIN.	NOM.	MAX.	MIN.	NOM.	MAX.
A	—	—	1.60	—	—	0.063
A_1	0.05	—	0.15	0.002	—	0.006
A_2	1.35	1.40	1.45	0.053	0.055	0.057
D		9.00 BSC.			0.354 BSC.	
D_1		7.00 BSC.			0.276 BSC.	
E		9.00 BSC.			0.354 BSC.	
E_1		7.00 BSC.			0.276 BSC.	
R_2	0.08	—	0.20	0.003	—	0.008
R_1	0.08	—	—	0.003	—	—
θ	0°	3.5°	7°	0°	3.5°	7°
$θ_1$	0°	—	—	0°	—	—
$θ_2$	11°	12°	13°	11°	12°	13°
$θ_3$	11°	12°	13°	11°	12°	13°
c	0.09	—	0.20	0.004	—	0.008
L	0.45	0.60	0.75	0.018	0.024	0.030
L_1		1.00 REF			0.039 REF	
S	0.20	—	—	0.008	—	—

NOTES :

1. DIMENSIONS D1 AND E1 DO NOT INCLUDE MOLD PROTRUSION. ALLOWABLE PROTRUSION IS 0.25mm PER SIDE. D1 AND E1 ARE MAXIMUN PLASTIC BODY SIZE DIMENSIONS INCLUDING MOLD MISMATCH.
2. DIMENSION b DOES NOT INCLUDE DAMBAR PROTRUSION. ALLOWABLE DAMBAR PROTRUSION SHALL NOT CAUSE THE LEAD WIDTH TO EXCEED THE MAXIMUM b DIMENSION BY MORE THAN 0.08mm.
   DAMBAR CAN NOT BE LOCATED ON THE LOWER RADIUS OR THE FOOT. MINIMUM SPACE BETWEEN PROTRUSION AND AN ADJACENT LEAD IS 0.07mm FOR 0.4mm and 0.5mm PITCH PACKAGES.
3. ALL DIMENSION OF 44L WERE BASE ON THOSE OF 48L SINCE THEY ARE NOT MENTIONED IN JEDEC SPEC MS–026.

Table 45-25. Device and 48-lead LQFP Package Maximum Weight

SAM4S	190	mg

Table 45-26. 48-lead LQFP Package Characteristics

Moisture Sensitivity Level	3