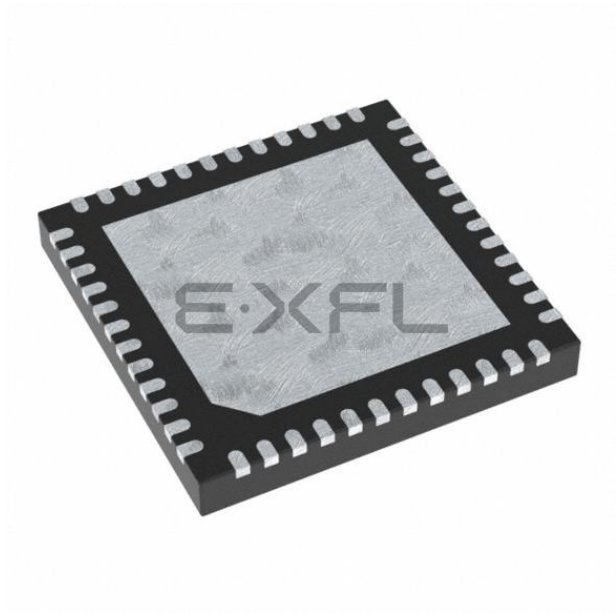


Welcome to E-XFL.COM

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®-M0+
Core Size	32-Bit Single-Core
Speed	48MHz
Connectivity	I²C, SCI, SPI, UART/USART, USB
Peripherals	Brown-out Detect/Reset, DMA, I²S, POR, PWM, WDT
Number of I/O	38
Program Memory Size	64KB (64K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	8K x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 3.63V
Data Converters	A/D 14x12b; D/A 1x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	48-VFQFN Exposed Pad
Supplier Device Package	48-QFN (7x7)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/atsamda1g16b-mbt

Table of Contents

Name: WPCLR
Offset: 0x00 [ID-00000a18]
Reset: 0x00800000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
				I2S	PTC	DAC	AC	ADC
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	TC7	TC4	TC5	TC4	TC3	TCC2	TCC1	TCC0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
							EVSYS	
Access							R/W	
Reset							0	

Bit 20 – I2S
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

Value	Description
0	Write-protection is disabled.
1	Write-protection is enabled.

Bit 19 – PTC
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

Value	Description
0	Write-protection is disabled.
1	Write-protection is enabled.

Bit 18 – DAC:
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

Value	Description
0	Write-protection is disabled.
1	Write-protection is enabled.

Table 18-4. Sleep Mode Overview

Sleep Mode	CPU Clock	AHB Clock	APB Clock	Oscillators				Main Clock	Regulator Mode	RAM Mode
				ONDEMAND = 0		ONDEMAND = 1				
				RUNSTDBY=0	RUNSTDBY=1	RUNSTDBY=0	RUNSTDBY=1			
Idle 0	Stop	Run	Run	Run	Run	Run if requested	Run if requested	Run	Normal	Normal
Idle 1	Stop	Stop	Run	Run	Run	Run if requested	Run if requested	Run	Normal	Normal
Idle 2	Stop	Stop	Stop	Run	Run	Run if requested	Run if requested	Run	Normal	Normal
Standby	Stop	Stop	Stop	Stop	Run	Stop	Run if requested	Stop	Low power	Low power

IDLE Mode

The IDLE modes allow power optimization with the fastest wake-up time.

The CPU is stopped. To further reduce power consumption, the user can disable the clocking of modules and clock sources by configuring the SLEEP.IDLE bit group. The module will be halted regardless of the bit settings of the mask registers in the Power Manager (PM.AHBMASK, PM.APBxMASK).

Regulator operates in normal mode.

- Entering IDLE mode: The IDLE mode is entered by executing the WFI instruction. Additionally, if the SLEEPONEXIT bit in the ARM Cortex System Control register (SCR) is set, the IDLE mode will also be entered when the CPU exits the lowest priority ISR. This mechanism can be useful for applications that only require the processor to run when an interrupt occurs. Before entering the IDLE mode, the user must configure the IDLE mode configuration bit group and must write a zero to the SCR.SLEEPDEEP bit.
- Exiting IDLE mode: The processor wakes the system up when it detects the occurrence of any interrupt that is not masked in the NVIC Controller with sufficient priority to cause exception entry. The system goes back to the ACTIVE mode. The CPU and affected modules are restarted.

STANDBY Mode

The STANDBY mode allows achieving very low power consumption.

In this mode, all clocks are stopped except those which are kept running if requested by a running module or have the ONDEMAND bit set to zero. For example, the RTC can operate in STANDBY mode. In this case, its Generic Clock clock source will also be enabled.

The regulator and the RAM operate in low-power mode.

A SLEEPONEXIT feature is also available.

- Entering STANDBY mode: This mode is entered by executing the WFI instruction with the SCR.SLEEPDEEP bit of the CPU is written to 1.
- Exiting STANDBY mode: Any peripheral able to generate an asynchronous interrupt can wake up the system. For example, a module running on a Generic clock can trigger an interrupt. When the enabled asynchronous wake-up event occurs and the system is woken up, the device will either execute the interrupt service routine or continue the normal program execution according to the Priority Mask Register (PRIMASK) configuration of the CPU.

18.6.3 SleepWalking

SleepWalking is the capability for a device to temporarily wak-eup clocks for peripheral to perform a task without waking-up the CPU in STANDBY sleep mode. At the end of the sleepwalking task, the device can either be waken-up by an interrupt (from a peripheral involved in SleepWalking) or enter again into STANDBY sleep mode.

Value	Description
0	The oscillator is always on, if enabled.
1	The oscillator is enabled when a peripheral is requesting the oscillator to be used as a clock source. The oscillator is disabled if no peripheral is requesting the clock source.

Bit 6 – RUNSTDBY: Run in Standby
This bit controls how the XOSC behaves during standby sleep mode:

Value	Description
0	The oscillator is disabled in standby sleep mode.
1	The oscillator is not stopped in standby sleep mode. If XOSC.ONDEMAND is one, the clock source will be running when a peripheral is requesting the clock. If XOSC.ONDEMAND is zero, the clock source will always be running in standby sleep mode.

Bit 2 – XTALEN: Crystal Oscillator Enable
This bit controls the connections between the I/O pads and the external clock or crystal oscillator:

Value	Description
0	External clock connected on XIN. XOUT can be used as general-purpose I/O.
1	Crystal connected to XIN/XOUT.

Bit 1 – ENABLE: Oscillator Enable

Value	Description
0	The oscillator is disabled.
1	The oscillator is enabled.

19.8.6 32kHz External Crystal Oscillator (XOSC32K) Control

Name: XOSC32K
Offset: 0x14 [ID-00003d5d]
Reset: 0x0080
Property: Write-Protected

Bit	15	14	13	12	11	10	9	8
				WRTLOCK		STARTUP[2:0]		
Access				R/W		R/W	R/W	R/W
Reset				0		0	0	0

Bit	7	6	5	4	3	2	1	0
	ONDEMAND	RUNSTDBY	AAMPEN		EN32K	XTALEN	ENABLE	
Access	R/W	R/W	R/W		R/W	R/W	R/W	
Reset	1	0	0		0	0	0	

Bit 12 – WRTLOCK: Write Lock
This bit locks the XOSC32K register for future writes to fix the XOSC32K configuration.

Value	Description
0	The XOSC32K configuration is not locked.
1	The XOSC32K configuration is locked.

Bits 10:8 – STARTUP[2:0]: Oscillator Start-Up Time
These bits select the start-up time for the oscillator.

The OSCULP32K oscillator is used to clock the start-up counter.

Table 19-6. Start-Up Time for 32kHz External Crystal Oscillator

STARTUP[2:0]	Number of OSCULP32K Clock Cycles	Number of XOSC32K Clock Cycles	Approximate Equivalent Time (OSCULP = 32kHz)(1)(2)(3)
0x0	1	3	122µs
0x1	32	3	1068µs
0x2	2048	3	62592µs
0x3	4096	3	125092µs
0x4	16384	3	500092µs
0x5	32768	3	1000092µs
0x6	65536	3	2000092µs
0x7	131072	3	4000092µs

Notes: 1. Number of cycles for the start-up counter.

2. Number of cycles for the synchronization delay, before PCLKSR.XOSC32KRDY is set.

3. Start-up time is n OSCULP32K cycles + 3 XOSC32K cycles.

Bit 7 – ONDEMAND: On Demand Control
The On Demand operation mode allows an oscillator to be enabled or disabled depending on peripheral clock requests.

In On Demand operation mode, i.e., if the ONDEMAND bit has been previously written to one, the oscillator will only be running when requested by a peripheral. If there is no peripheral requesting the oscillator s clock source, the oscillator will be in a disabled state.

If On Demand is disabled the oscillator will always be running when enabled.

In standby sleep mode, the On Demand operation is still active if the XOSC32K.RUNSTDBY bit is one. If XOSC32K.RUNSTDBY is zero, the oscillator is disabled.

Value	Description
0	The oscillator is always on, if enabled.
1	The oscillator is enabled when a peripheral is requesting the oscillator to be used as a clock source. The oscillator is disabled if no peripheral is requesting the clock source.

Bit 6 – RUNSTDBY: Run in Standby
This bit controls how the XOSC32K behaves during standby sleep mode:

Value	Description
0	The oscillator is disabled in standby sleep mode.
1	The oscillator is not stopped in standby sleep mode. If XOSC32K.ONDEMAND is one, the clock source will be running when a peripheral is requesting the clock. If

Bit	7	6	5	4	3	2	1	0
								EW
Access								R/W
Reset								0

Bit 0 – EW: Early Warning Interrupt Enable
Writing a zero to this bit has no effect.
Writing a one to this bit disables the Early Warning interrupt.

Value	Description
0	The Early Warning interrupt is disabled.
1	The Early Warning interrupt is enabled.

20.8.5 Interrupt Enable Set

Name: INTENSET
Offset: 0x5 [ID-000004d0]
Reset: 0x00
Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
								EW
Access								R/W
Reset								0

Bit 0 – EW: Early Warning Interrupt Enable
Writing a zero to this bit has no effect.
Writing a one to this bit enables the Early Warning interrupt.

Value	Description
0	The Early Warning interrupt is disabled.
1	The Early Warning interrupt is enabled.

20.8.6 Interrupt Flag Status and Clear

Name: INTFLAG
Offset: 0x6 [ID-000004d0]
Reset: 0x00
Property: –

Bit	7	6	5	4	3	2	1	0
								EW
Access								R/W
Reset								0

Bit 0 – EW: Early Warning
This flag is set when an Early Warning interrupt occurs, as defined by the EWOFFSET bit group in EWCTRL.

2. Clock representation must be selected by writing the Clock Representation bit in the Control register (CTRL.CLKREP)
3. Prescaler value must be selected by writing the Prescaler bit group in the Control register (CTRL.PRESCALER)

The RTC prescaler divides the source clock for the RTC counter.
Note: In Clock/Calendar mode, the prescaler must be configured to provide a 1Hz clock to the counter for correct operation.

The frequency of the RTC clock (CLK_RTC_CNT) is given by the following formula:

$$f_{CLK\_RTC\_CNT} = \frac{f_{GCLK\_RTC}}{2^{PRESCALER}}$$

The frequency of the generic clock, GCLK_RTC, is given by f_{GCLK_RTC}, and $f_{CLK\_RTC\_CNT}$ is the frequency of the internal prescaled RTC clock, CLK_RTC_CNT.

21.6.2.2 Enabling, Disabling and Resetting

The RTC is enabled by setting the Enable bit in the Control register (CTRL.ENABLE=1). The RTC is disabled by writing CTRL.ENABLE=0.

The RTC is reset by setting the Software Reset bit in the Control register (CTRL.SWRST=1). All registers in the RTC, except DEBUG, will be reset to their initial state, and the RTC will be disabled. The RTC must be disabled before resetting it.

21.6.3 Operating Modes

The RTC counter supports three RTC operating modes: 32-bit Counter, 16-bit Counter and Clock/Calendar. The operating mode is selected by writing to the Operating Mode bit group in the Control register (CTRL.MODE).

21.6.3.1 32-Bit Counter (Mode 0)

When the RTC Operating Mode bits in the Control register are zero (CTRL.MODE=00), the counter operates in 32-bit Counter mode. The block diagram of this mode is shown in Figure 21-1. When the RTC is enabled, the counter will increment on every 0-to-1 transition of CLK_RTC_CNT. The counter will increment until it reaches the top value of 0xFFFFFFFF, and then wrap to 0x00000000. This sets the Overflow Interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.OVF).

The RTC counter value can be read from or written to the Counter Value register (COUNT) in 32-bit format.

The counter value is continuously compared with the 32-bit Compare register (COMP0). When a compare match occurs, the Compare 0interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.CMP0) is set on the next 0-to-1 transition of CLK_RTC_CNT.

If the Clear on Match bit in the Control register (CTRL.MATCHCLR) is '1', the counter is cleared on the next counter cycle when a compare match with COMP0 occurs. This allows the RTC to generate periodic interrupts or events with longer periods than are possible with the prescaler events. Note that when CTRL.MATCHCLR is '1', INTFLAG.CMP0 and INTFLAG.OVF will both be set simultaneously on a compare match with COMP0.

21.6.3.2 16-Bit Counter (Mode 1)

When the RTC Operating Mode bits in the Control register (CTRL.MODE) are 1, the counter operates in 16-bit Counter mode as shown in Figure 21-2. When the RTC is enabled, the counter will increment on every 0-to-1 transition of CLK_RTC_CNT. In 16-bit Counter mode, the 16-bit Period register (PER) holds the maximum value of the counter. The counter will increment until it reaches the PER value, and then

An AHB clock (CLK_DMAC_AHB) is required to clock the DMAC. This clock must be configured and enabled in the power manager before using the DMAC, and the default state of CLK_DMAC_AHB can be found in *Peripheral Clock Masking*.

This bus clock (CLK_DMAC_APB) is always synchronous to the module clock (CLK_DMAC_AHB), but can be divided by a prescaler and may run even when the module clock is turned off.

Related Links

Peripheral Clock Masking

22.5.4 DMA

Not applicable.

22.5.5 Interrupts

The interrupt request line is connected to the interrupt controller. Using the DMAC interrupt requires the interrupt controller to be configured first.

Related Links

Nested Vector Interrupt Controller

22.5.6 Events

The events are connected to the event system.

Related Links

EVSYS – Event System

22.5.7 Debug Operation

When the CPU is halted in debug mode the DMAC will halt normal operation. The DMAC can be forced to continue operation during debugging. Refer to DBGCTRL for details.

22.5.8 Register Access Protection

All registers with write-access can be write-protected optionally by the Peripheral Access Controller (PAC), except for the following registers:

- Interrupt Pending register (INTPEND)
- Channel ID register (CHID)
- Channel Interrupt Flag Status and Clear register (CHINTFLAG)

Optional write-protection by the Peripheral Access Controller (PAC) is denoted by the "PAC Write-Protection" property in each individual register description.

PAC write-protection does not apply to accesses through an external debugger.

Related Links

PAC - Peripheral Access Controller

22.5.9 Analog Connections

Not applicable.

22.6 Functional Description

22.6.1 Principle of Operation

The DMAC consists of a DMA module and a CRC module.

Bit	31	30	29	28	27	26	25	24
	BTCNT[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	BTCNT[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	ABUSY			ID[4:0]				
Access	R			R	R	R	R	R
Reset	0			0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
					LVLEXx	LVLEXx	LVLEXx	LVLEXx
Access					R	R	R	R
Reset					0	0	0	0

Bits 31:16 – BTCNT[15:0]: Active Channel Block Transfer Count
These bits hold the 16-bit block transfer count of the ongoing transfer. This value is stored in the active channel and written back in the corresponding Write-Back channel memory location when the arbiter grants a new channel access. The value is valid only when the active channel active busy flag (ABUSY) is set.

Bit 15 – ABUSY: Active Channel Busy
This bit is cleared when the active transfer count is written back in the write-back memory section.

This bit is set when the next descriptor transfer count is read from the write-back memory section.

Bits 12:8 – ID[4:0]: Active Channel ID
These bits hold the channel index currently stored in the active channel registers. The value is updated each time the arbiter grants a new channel transfer access request.

Bits 3,2,1,0 – LVLEXx: Level x Channel Trigger Request Executing [x=3..0]
This bit is set when a level-x channel trigger request is executing or pending.

This bit is cleared when no request is pending or being executed.

22.8.15 Descriptor Memory Section Base Address

Name: BASEADDR
Offset: 0x34 [ID-00001ece]
Reset: 0x00000000
Property: PAC Write-Protection, Enable-Protected

Value	Description
0	Hardware $\overline{SS}$ control is disabled.
1	Hardware $\overline{SS}$ control is enabled.

Bit 9 – SSDE: Slave Select Low Detect Enable
This bit enables wake up when the slave select ($\overline{SS}$) pin transitions from high to low.

Value	Description
0	$\overline{SS}$ low detector is disabled.
1	$\overline{SS}$ low detector is enabled.

Bit 6 – PLOADEN: Slave Data Preload Enable
Setting this bit will enable preloading of the slave shift register when there is no transfer in progress. If the $\overline{SS}$ line is high when DATA is written, it will be transferred immediately to the shift register.

Bits 2:0 – CHSIZE[2:0]: Character Size

CHSIZE[2:0]	Name	Description
0x0	8BIT	8 bits
0x1	9BIT	9 bits
0x2-0x7	-	Reserved

29.8.3 Baud Rate

Name: BAUD
Offset: 0x0C [ID-00000e74]
Reset: 0x00
Property: PAC Write-Protection, Enable-Protected

Bit	7	6	5	4	3	2	1	0
	BAUD[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 7:0 – BAUD[7:0]: Baud Register
These bits control the clock generation, as described in the *SERCOM Clock Generation – Baud-Rate Generator*.

29.8.4 Interrupt Enable Clear

Name: INTENCLR
Offset: 0x14 [ID-00000e74]
Reset: 0x00
Property: PAC Write-Protection

In Controller mode, only the Clock generation unit needs to be configured by writing to the CTRLA and CLKCTRLn registers, where parameters such as clock division factors, Number of slots, Slot size, Frame Sync signal, clock enable are selected.

MCKn Clock Frequency

When the I²S is in Master mode, writing a '1' to CLKCTRLn.MCKEN will output GCLK_I2S_n as Master Clock to the MCKn pin. The Master Clock to MCKn pin can be divided by writing to CLKCTRLn.MCKSEL and CLKCTRLn.MCKOUTOUT. The Master Clock (MCKn) frequency is GCLK_I2S_n frequency divided by (MCLKOUTDIV+1).

$$f(\mathrm{MCKn}) = \frac{f(\mathrm{GCLK\_I2S\_}n)}{(\mathrm{MCKOUTDIV}+1)}$$

SCKn Clock Frequency

When the Serial Clock (SCKn) is generated from GCLK_I2S_n and both CLKCTRLn.MCKSEL and CLKCTRLn.SCKSEL are zero, the Serial Clock (SCKn) frequency is GCLK_I2S_n frequency divided by (MCKDIV+1).

i.e.

$$f(\mathrm{SCKn}) = \frac{f(\mathrm{GCLK\_I2S\_}n)}{(\mathrm{MCKDIV}+1)}$$

Relation Between MCKn, SCKn, and Sampling Frequency *fs*

Based on sampling frequency ***fs***, the SCKn frequency requirement can be calculated:

- SCKn frequency: $f_{\mathrm{SCKn}} = fs \times \mathrm{total\_number\_of\_bits\_per\_frame}$,
- Where $\mathrm{total\_number\_of\_bits\_per\_frame} = \mathrm{number\_of\_slots} \times \mathrm{number\_of\_bits\_per\_slots}$.
- The number of slots is selected by writing to the Number of Slots in Frame bit field in the Clock Unit n Control (CLKCTRLn) register: $\mathrm{number\_of\_slots} = \mathrm{NBSLOTS} + 1$.
- The number of bits per slot (8, 16, 24, or 32 bit) is selected by writing to the Slot Size bit field in CLKCTRLn: .
- Consequently, $f_{\mathrm{SCKn}} = 8 \times fs \times (\mathrm{NBSLOTS}+1) \times (\mathrm{SLOTSIZE}+1)$.

The clock frequencies f_{SCKn} and f_{MCKn} are derived from the generic clock frequency $f_{\mathrm{GCLK\_I2S\_n}}$:

- $f_{\mathrm{GCLK\_I2S\_n}} = f_{\mathrm{SCKn}} \times (\mathrm{CLKCTRLn.MCKDIV}+1)$
  $= 8 \times fs \times (\mathrm{NBSLOTS}+1) \times (\mathrm{SLOTSIZE}+1) \times (\mathrm{MCKDIV}+1)$
  , and
- $f_{\mathrm{GCLK\_I2S\_n}} = f_{\mathrm{MCKn}} \times (\mathrm{MCKOUTDIV}+1)$.

Substituting the right hand sides of the two last equations yields:

$$f_{\mathrm{MCKn}} = \frac{f_{\mathrm{GCLK\_I2S\_n}}}{\mathrm{MCKOUTDIV}+1}$$

$$f_{\mathrm{MCKn}} = \frac{8 \cdot (\mathrm{SLOTSIZE}+1) \cdot (\mathrm{NBSLOTS}+1) \cdot (\mathrm{MCKDIV}+1)}{\mathrm{MCKOUTDIV}+1}$$

If a Master Clock output is not required, the GCLK_I2S generic clock can be configured as SCKn by writing a '0'to CLKCTRLn.MCKDIV. Alternatively, if the frequency of the generic clock is a multiple of the required SCKn frequency, the MCKn-to-SCKn divider can be used with the ratio defined by writing the CLKCTRLn.MCKDIV field.

enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled, or the I²S is reset. See INTFLAG register for details on how to clear interrupt flags. All interrupt requests from the peripheral are ORed together on system level to generate one combined interrupt request to the NVIC. Refer to *Nested Vector Interrupt Controller* for details. The user must read the INTFLAG register to determine which interrupt condition is present.

Note: Interrupts must be globally enabled for interrupt requests to be generated. Refer to *Nested Vector Interrupt Controller* for details.

Related Links

Nested Vector Interrupt Controller

31.6.8.3 Events

Not applicable.

31.6.9 Sleep Mode Operation

The I²S continues to operate in all sleep modes that still provide its clocks.

31.6.10 Synchronization

Due to asynchronicity between the main clock domain and the peripheral clock domains, some registers need to be synchronized when written or read.

When executing an operation that requires synchronization, the corresponding Synchronization Busy bit in the Synchronization Busy register (SYNCBUSY) will be set immediately, and cleared when synchronization is complete.

If an operation that requires synchronization is executed while the corresponding SYNCBUSY bit is '1', a peripheral bus error is generated.

The following bits are synchronized when written:

- Software Reset bit in the Control A register (CTRLA.SWRST). SYNCBUSY.SWRST is set to '1' while synchronization is in progress.
- Enable bit in the Control A register (CTRLA.ENABLE). SYNCBUSY.ENABLE is set to '1' while synchronization is in progress.
- Clock Unit x Enable bits in the Control A register (CTRLA.CKENx). SYNCBUSY.CKENx is set to '1' while synchronization is in progress.
- Serializer x Enable bits in the Control A register (CTRLA.SERENx). SYNCBUSY.SERENx is set to '1' while synchronization is in progress.

The following registers require synchronization when read or written:

- Data n registers (DATAn), Read-Synchronized when Serializer n is in Rx mode or Write-Synchronized when in Tx mode. SYNCBUSY.DATAn is set to '1' while synchronization is in progress.

Synchronization is denoted by the Read-Synchronized or Write-Synchronized property in the register description.

Related Links

Register Synchronization

31.6.11 Loop-Back Mode

For debugging purposes, the I²S can be configured to loop back the Transmitter to the Receiver. Writing a '1' to the Loop-Back Test Mode bit in the Serializer m Control register (SERCTRLm.RXLOOP) configures SDm as input and the remaining SD as output. Both SD will be connected internally, so the transmitted

32.8.12 Counter Value

32.8.12.1 Counter Value, 8-bit Mode

Name: COUNT
Offset: 0x10 [ID-00001cd8]
Reset: 0x00
Property: PAC Write-Protection, Write-Synchronized, Read-Synchronized

Bit	7	6	5	4	3	2	1	0
	COUNT[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 7:0 – COUNT[7:0]: Counter Value
These bits contain the current counter value.

32.8.12.2 Counter Value, 16-bit Mode

Name: COUNT
Offset: 0x10 [ID-00001cd8]
Reset: 0x00
Property: PAC Write-Protection, Write-Synchronized, Read-Synchronized

Bit	15	14	13	12	11	10	9	8
	COUNT[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	COUNT[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – COUNT[15:0]: Counter Value
These bits contain the current counter value.

32.8.12.3 Counter Value, 32-bit Mode

Name: COUNT
Offset: 0x10 [ID-00001cd8]
Reset: 0x00
Property: PAC Write-Protection, Write-Synchronized, Read-Synchronized

Table 34-4. Device Endpoint n Descriptor Bank 1

Offset 0x n0 + 0x10 + index	Name	Bit Pos.								
0x00	ADDR	7:0	ADD[7:0]							
0x01		15:8	ADD[15:8]							
0x02		23:16	ADD[23:16]							
0x03		31:24	ADD[31:24]							
0x04	PCKSIZE	7:0	BYTE_COUNT[7:0]							
0x05		15:8	MULTI_PACKET_SIZE[1:0]		BYTE_COUNT[13:8]					
0x06		23:16	MULTI_PACKET_SIZE[9:2]							
0x07		31:24	AUTO_ZLP	SIZE[2:0]			MULTI_PACKET_SIZE[13:10]			
0x08	Reserved	7:0								
0x09	Reserved	15:8								
0x0A	STATUS_BK	7:0							ERRORFLOW	CRCERR
0x0B	Reserved	7:0								
0x0C	Reserved	7:0								
0x0D	Reserved	7:0								
0x0E	Reserved	7:0								
0x0F	Reserved	7:0								

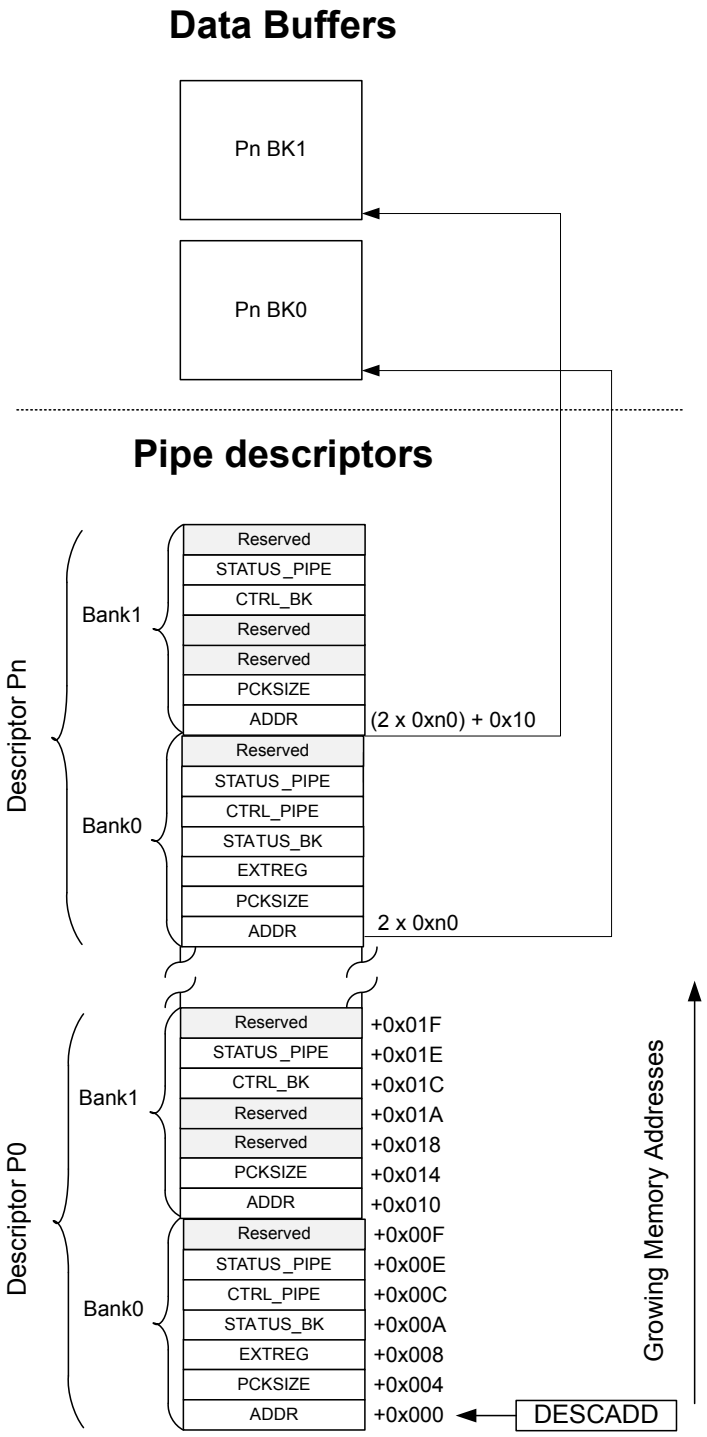
34.7.3 Host Summary

Table 34-5. General Host Registers

Offset	Name	Bit Pos.								
0x04	Reserved									
0x05	Reserved									
0x06	Reserved									
0x07	Reserved									
0x08	CTRLB	7:0		TSTK	TSTJ		SPDCONF[1:0]		RESUME	
0x09		15:8					L1RESUME	VBUSOK	BUSRESET	SOFE
0x0A	HSOFC	7:0	FLENCE				FLENC[3:0]			
0x0B	Reserved									
0x0C	STATUS	7:0	LINESTATE[1:0]				SPEED[1:0]			
0x0E	Reserved									
0x0F	Reserved									
0x10	FNUM	7:0	FNUM[4:0]							
0x11		15:8			FNUM[10:5]					
0x12	FLENHIGH	7:0	FLENHIGH[7:0]							
0x14	INTENCLR	7:0	RAMACER	UPRSM	DNRSM	WAKEUP	RST	HSOF		
0x15		15:8							DDISC	DCONN
0x16	Reserved									
0x17	Reserved									
0x18	INTENSET	7:0	RAMACER	UPRSM	DNRSM	WAKEUP	RST	HSOF		
0x19		15:8							DDISC	DCONN
0x1A	Reserved									

34.8.7 Host Registers - Pipe RAM

34.8.7.1 Pipe Descriptor Structure



34.8.7.2 Address of the Data Buffer

(OFFSETCORR). The offset correction value is subtracted from the converted data before writing the Result register (RESULT).

The gain error is defined as the deviation of the last output step's midpoint from the ideal straight line, after compensating for offset error. The gain error cancellation is handled by the Gain Correction register (GAINCORR).

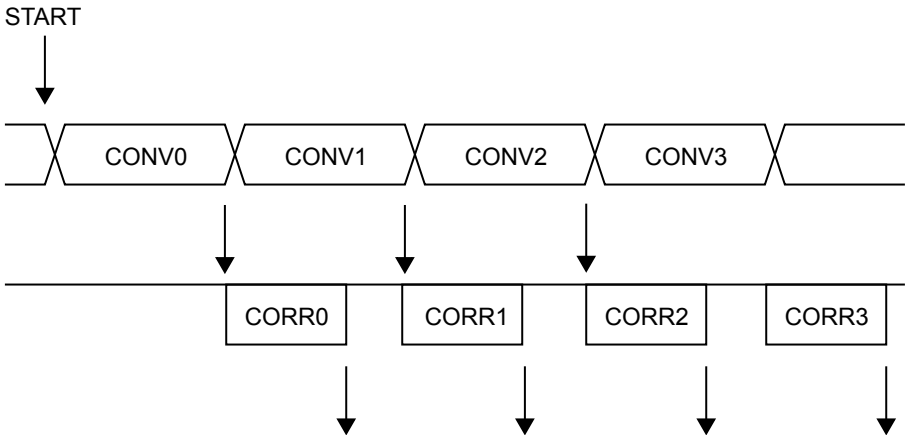
To correct these two errors, the Digital Correction Logic Enabled bit in the Control B register (CTRLB.CORREN) must be set to ".

Offset and gain error compensation results are both calculated according to:

$$\text{Result} = (\text{Conversion value} + - \text{OFFSETCORR}) \cdot \text{GAINCORR}$$

The correction will introduce a latency of 13 CLK_ADC clock cycles. In free running mode this latency is introduced on the first conversion only, since its duration is always less than the propagation delay. In single conversion mode this latency is introduced for each conversion.

Figure 35-8. ADC Timing Correction Enabled



35.6.11 DMA Operation

The ADC generates the following DMA request:

- Result Conversion Ready (RESRDY): the request is set when a conversion result is available and cleared when the RESULT register is read. When the averaging operation is enabled, the DMA request is set when the averaging is completed and result is available.

35.6.12 Interrupts

The ADC has the following interrupt sources:

- Result Conversion Ready: RESRDY
- Window Monitor: WINMON
- Overrun: OVERRUN

Each interrupt source has an interrupt flag associated with it. The interrupt flag in the Interrupt Flag Status and Clear (INTFLAG) register is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a one to the corresponding bit in the Interrupt Enable Set (INTENSET) register, and disabled by writing a one to the corresponding bit in the Interrupt Enable Clear (INTENCLR) register. An interrupt request is generated when the interrupt flag is set and the corresponding interrupt is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled, or the ADC is reset. An interrupt flag is cleared by writing a one to the corresponding bit in the INTFLAG register. Each peripheral can have one interrupt request line per interrupt source or one common interrupt request line for all the interrupt sources. This is device dependent.

37. DAC – Digital-to-Analog Converter

37.1 Overview

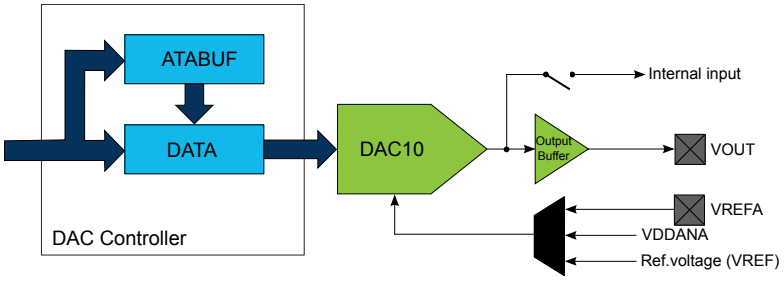
The Digital-to-Analog Converter (DAC) converts a digital value to a voltage. The DAC has one channel with 10-bit resolution, and it is capable of converting up to 350,000 samples per second (350ksps).

37.2 Features

- DAC with 10-bit resolution
- Up to 350ksps conversion rate
- Multiple trigger sources
- High-drive capabilities
- Output can be used as input to the Analog Comparator (AC)
- DMA support

37.3 Block Diagram

Figure 37-1. DAC Block Diagram



37.4 Signal Description

Signal Name	Type	Description
VOUT	Analog output	DAC output
VREFA	Analog input	External reference

Related Links

I/O Multiplexing and Considerations

37.5 Product Dependencies

In order to use this peripheral, other parts of the system must be configured correctly, as described below.

37.5.1 I/O Lines

Using the DAC Controller's I/O lines requires the I/O pins to be configured using the port configuration (PORT).

Figure 41-3. External Analog Reference Schematic With One Reference

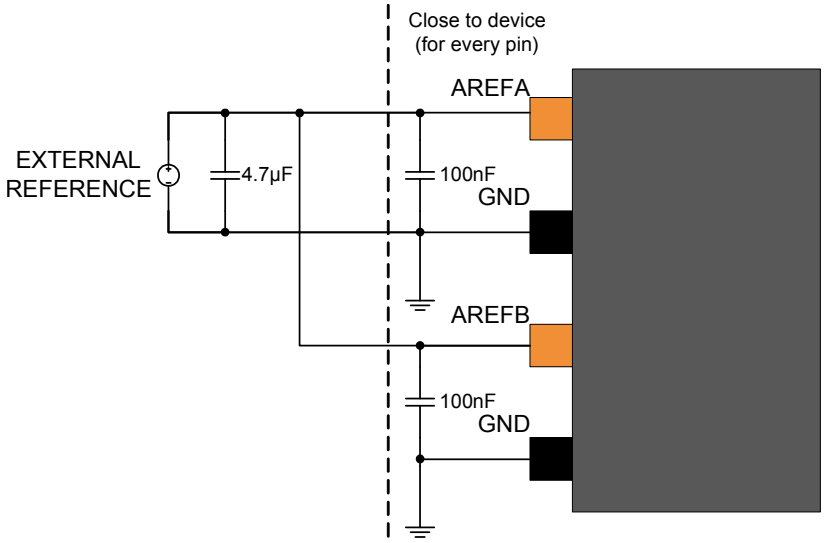


Table 41-2. External Analog Reference Connections

Signal Name	Recommended Pin Connection	Description
AREFx	1.0V to V_{DDANA} - 0.6V for ADC 1.0V to V_{DDANA}- 0.6V for DAC Decoupling/filtering capacitors 100nF(1)(2) and 4.7µF(1)	External reference from AREFx pin on the analog port
GND		Ground

1. These values are given as a typical example.
2. Decoupling capacitor should be placed close to the device for each supply pin pair in the signal group.

41.4 External Reset Circuit

The external reset circuit is connected to the $\overline{\text{RESET}}$ pin when the external reset function is used. If the external reset function has been disabled, the circuit is not necessary. The reset switch can also be removed, if the manual reset is not necessary. The $\overline{\text{RESET}}$ pin itself has an internal pull-up resistor, hence it is optional to also add an external pull-up resistor.

This happens if the channel number of the channel being enabled is lower than the channel already active.
Errata reference: 15683
Fix/Workaround:
When enabling a DMA channel while other channels using linked descriptors are already active, the channel number of the new channel enabled must be greater than the other channel numbers.

2 – If data is written to CRCDATAIN in two consecutive instructions, the CRC computation may be incorrect.
Errata reference: 13507
Fix/Workaround:
Add a NOP instruction between each write to CRCDATAIN register.

42.2.7 EIC

1 – When the EIC is configured to generate an interrupt on a low level or rising edge or both edges (CONFIGn.SENSEx) with the filter enabled (CONFIGn.FILTENx), a spurious flag might appear for the dedicated pin on the INTFLAG.EXTINT[x] register as soon as the EIC is enabled using CTRLA ENABLE bit.
Errata reference: 15341
Fix/Workaround:
Clear the INTFLAG bit once the EIC enabled and before enabling the interrupts.

42.2.8 NVMCTRL

1 – Default value of MANW in NVM.CTRLB is 0.
This can lead to spurious writes to the NVM if a data write is done through a pointer with a wrong address corresponding to NVM area.
Errata reference: 13134
Fix/Workaround:
Set MANW in the NVM.CTRLB to 1 at startup

42.2.9 SERCOM

1 – In USART autobaud mode, missing stop bits are not recognized as inconsistent sync (ISF) or framing (FERR) errors.
Errata reference: 13852
Fix/Workaround:
None

42.2.10 TCC

1 – FCTRLX.CAPTURE[CAPTMARK] does not work as described in the datasheet. CAPTMARK cannot be used to identify captured values triggered by fault inputs source A or B on the same channel.
Errata reference: 13316
Fix/Workaround:
Use two different channels to timestamp FaultA and FaultB.

2 – Using TCC in dithering mode with external retrigger events can lead to unexpected stretch of right aligned pulses, or shrink of left aligned pulses.