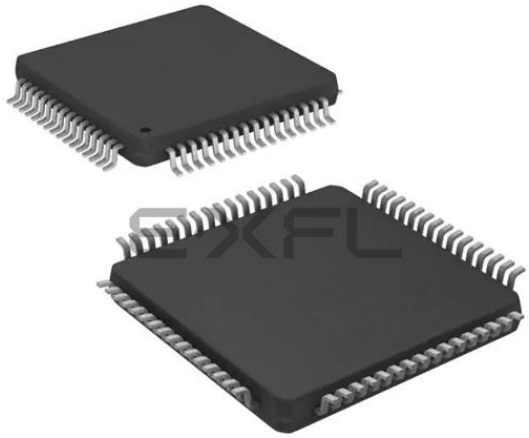


Welcome to E-XFL.COM

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®-M4F
Core Size	32-Bit Single-Core
Speed	120MHz
Connectivity	EBI/EMI, Ethernet, I²C, IrDA, LINbus, MMC/SD, QSPI, SPI, UART/USART, USB
Peripherals	Brown-out Detect/Reset, DMA, I²S, POR, PWM, WDT
Number of I/O	51
Program Memory Size	1MB (1M x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	256K x 8
Voltage - Supply (Vcc/Vdd)	1.71V ~ 3.63V
Data Converters	A/D 24x12b; D/A 2x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	64-TQFP
Supplier Device Package	64-TQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/atsame53j20a-aut

Figure 12-3. Hot-Plugging Detection Timing Diagram

The presence of a debugger probe is detected when either Hot-Plugging or Cold-Plugging is detected. Once detected, the Debugger Present bit of the Status B register (STATUSB.DBGPRES) is set. For security reasons, Hot-Plugging is not available when the device is protected by the NVMCTRL security bit.

This detection requires that pads are correctly powered. Thus, at cold startup, this detection cannot be done until POR is released. If the device is protected, Cold-Plugging is the only way to detect a debugger probe, and so the external reset timing must be longer than the POR timing. If external reset is deasserted before POR release, the user must retry the procedure above until it gets connected to the device.

Related Links

25. NVMCTRL – Nonvolatile Memory Controller

12.7 Chip Erase

Chip-Erase consists of removing all sensitive information stored in the chip and clearing the NVMCTRL security bit. Therefore, all volatile memories and the Flash memory (including the EEPROM emulation area) will be erased. The Flash auxiliary rows, including the user row, will not be erased.

When the device is protected, the debugger must first reset the device in order to be detected. This ensures that internal registers are reset after the protected state is removed. The Chip-Erase operation is triggered by writing a '1' to the Chip-Erase bit in the Control register (CTRL.CE). This command will be discarded if the DSU is protected by the Peripheral Access Controller (PAC). Once issued, the module clears volatile memories prior to erasing the Flash array. To ensure that the Chip-Erase operation is completed, check the Done bit of the Status A register (STATUSA.DONE).

The Chip-Erase operation depends on clocks and power management features that can be altered by the CPU. For that reason, it is recommended to issue a Chip- Erase after a Cold-Plugging procedure to ensure that the device is in a known and safe state.

The recommended sequence is as follows:

1. Issue the Cold-Plugging procedure (refer to 12.6.3.1 Cold Plugging). The device then:
   1.1. Detects the debugger probe.
   1.2. Holds the CPU in reset.
2. Issue the Chip-Erase command by writing a '1' to CTRL.CE. The device then:
   2.1. Clears the system volatile memories.
   2.2. Erases the whole Flash array (including the EEPROM emulation area, not including auxiliary rows).

14.6.5.3 Entering Standby Mode

There may occur a delay when the device is put into Standby, until the power is turned off. This delay is caused by running Clock Generators: if the Run in Standby bit in the Generator Control register (GENCTRLn.RUNSTDBY) is '0', GCLK must verify that the clock is turned of properly. The duration of this verification is frequency-dependent.

Related Links

18. PM – Power Manager

14.6.6 Synchronization

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

An exception is the Channel Enable bit in the Peripheral Channel Control registers (PCHCTRLm.CHEN). When changing this bit, the bit value must be read-back to ensure the synchronization is complete and to assert glitch free internal operation. Note that changing the bit value under ongoing synchronization will *not* generate an error.

The following registers are synchronized when written:

- Generic Clock Generator Control register (GENCTRLn)
- Control A register (CTRLA)

Required write-synchronization is denoted by the "Write-Synchronized" property in the register description.

Related Links

14.8.1 CTRLA

14.8.4 PCHCTRLm

15.8.11 APBD Mask

Name: APBDMASK
Offset: 0x20
Reset: 0x00000000
Property: PAC Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
					PCC	I2S	DAC	ADCn1
Access					R/W	R/W	R	R/W
Reset					0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADCn0	TC7	TC6	TCC4	SERCOM7	SERCOM6	SERCOM5	SERCOM4
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 11 – PCC PCC APBD Mask Clock Enable

Value	Description
0	The APBD clock for the PCC is stopped.
1	The APBD clock for the PCC is enabled.

Bit 10 – I2S I2S APBD Mask Clock Enable

Value	Description
0	The APBD clock for the I2S is stopped.
1	The APBD clock for the I2S is enabled.

Bit 9 – DAC DAC APBD Mask Clock Enable

Value	Description
0	The APBD clock for the DAC is stopped.
1	The APBD clock for the DAC is enabled.

Bits 7, 8 – ADCn ADCn APBD Mask Clock Enable

Value	Description
0	The APBD clock for the ADCn is stopped.
1	The APBD clock for the ADCn is enabled.

Value	Name	Description
0xB	DIV1024	CLK_RTC_CNT = GCLK_RTC/1024
0xC-0xF	-	Reserved

Bit 7 – MATCHCLR Clear on Match
This bit is valid only in Mode 0 (COUNT32) and Mode 2 (CLOCK). This bit can be written only when the peripheral is disabled. This bit is not synchronized.

Value	Description
0	The counter is not cleared on a Compare/Alarm 0 match
1	The counter is cleared on a Compare/Alarm 0 match

Bit 6 – CLKREP Clock Representation
This bit is valid only in Mode 2 and determines how the hours are represented in the Clock Value (CLOCK) register. This bit can be written only when the peripheral is disabled. This bit is not synchronized.

Value	Description
0	24 Hour
1	12 Hour (AM/PM)

Bits 3:2 – MODE[1:0] Operating Mode
This field defines the operating mode of the RTC. This bit is not synchronized.

Value	Name	Description
0x0	COUNT32	Mode 0: 32-bit counter
0x1	COUNT16	Mode 1: 16-bit counter
0x2	CLOCK	Mode 2: Clock/calendar
0x3	-	Reserved

Bit 1 – ENABLE Enable
Due to synchronization there is delay from writing CTRLA.ENABLE until the peripheral is enabled/disabled. The value written to CTRLA.ENABLE will read back immediately and the Enable bit in the Synchronization Busy register (SYNCBUSY.ENABLE) will be set. SYNCBUSY.ENABLE will be cleared when the operation is complete.

Value	Description
0	The peripheral is disabled
1	The peripheral is enabled

Bit 0 – SWRST Software Reset
Writing a '0' to this bit has no effect.

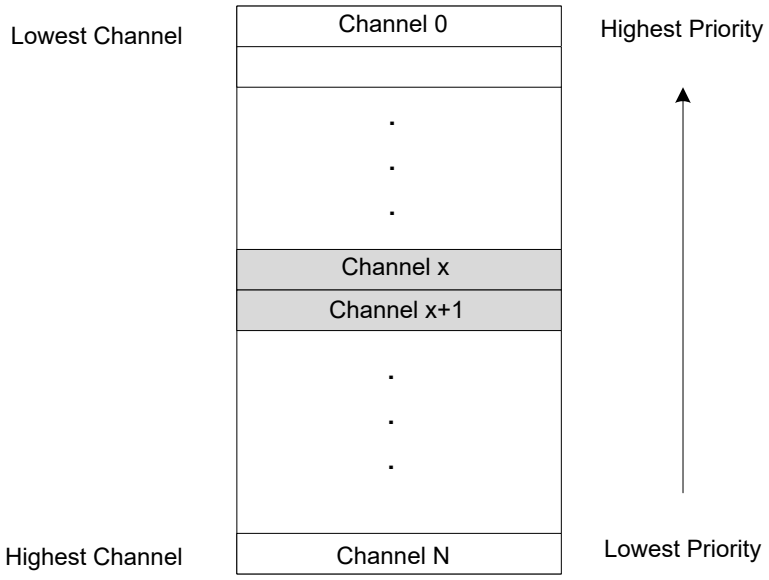
Writing a '1' to this bit resets all registers in the RTC, except DBGCTRL, to their initial state, and the RTC will be disabled.

Writing a '1' to CTRLA.SWRST will always take precedence, meaning that all other writes in the same write-operation will be discarded.

Due to synchronization there is a delay from writing CTRLA.SWRST until the reset is complete. CTRLA.SWRST will be cleared when the reset is complete.

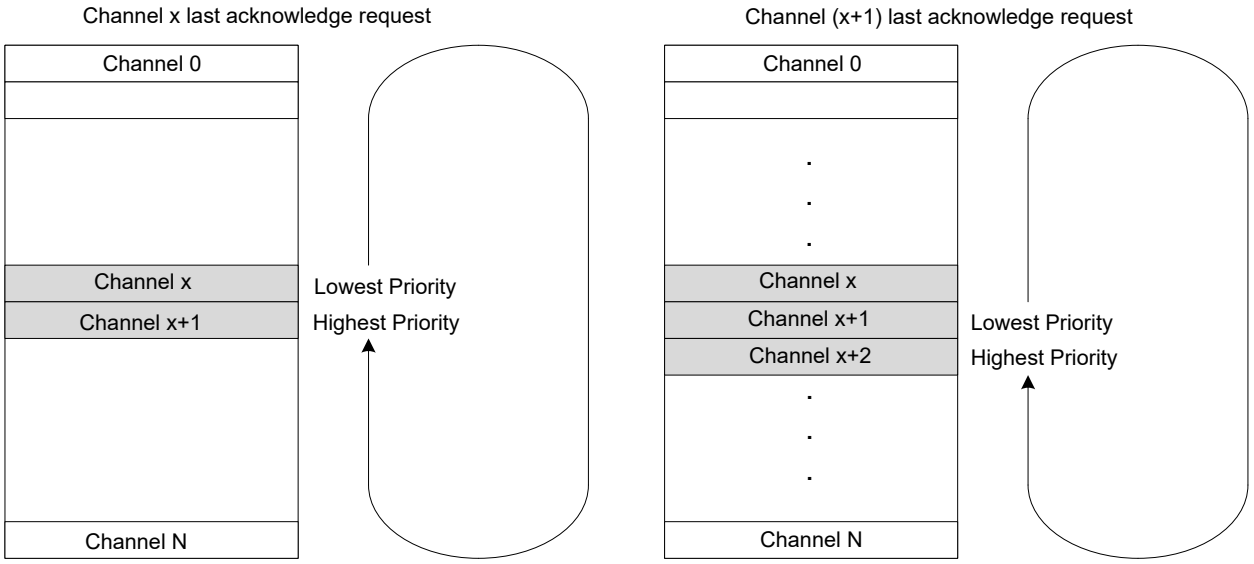
Within each priority level, the DMAC's arbiter can be configured to prioritize statically or dynamically. For the arbiter to perform static arbitration within a priority level, the Level X Round-Robin Scheduling Enable bit in the Priority Control x register (PRICTRL0.RRLVLENx) has to be written to '0'. When static arbitration is enabled (PRICTRL0.RRLVLENx is '0'), the arbiter will prioritize a low channel number over a high channel number as shown in Static Priority Scheduling. When using the static scheme, there is a risk of high channel numbers never being granted access as the active channel. This can be avoided using a dynamic arbitration scheme.

Figure 22-5. Static Priority Scheduling



The dynamic arbitration scheme in the DMAC is round-robin. Round-robin arbitration is enabled by writing PRICTRL0.RRLVLEN to '1', for a given priority level x. With the round-robin scheme, the channel number of the last channel being granted access will have the lowest priority the next time the arbiter has to grant access to a channel within the same priority level, as shown in Figure 22-6. The channel number of the last channel being granted access as the active channel is stored in the Level x Channel Priority Number bit group in the Priority Control 0 register (PRICTRL0.LVLPRIx) for the corresponding priority level.

Figure 22-6. Dynamic (Round-Robin) Priority Scheduling



22.8.11 Busy Channels

Name: BUSYCH
Offset: 0x28
Reset: 0x00000000
Property: -

Bit	31	30	29	28	27	26	25	24
	BUSYCHn[31:24]							
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
	BUSYCHn[23:16]							
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
	BUSYCHn[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	BUSYCHn[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – BUSYCHn[31:0] Busy Channel n [x=31..0]
This bit is cleared when the channel trigger action for DMA channel n is complete, when a bus error for DMA channel n is detected, or when DMA channel n is disabled.

This bit is set when DMA channel n starts a DMA transfer.

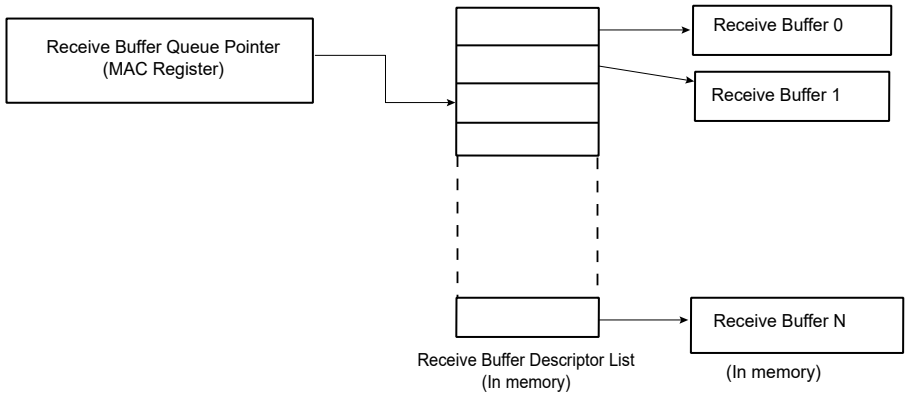
1. Write to Network Control register to disable transmit and receive circuits.
2. Write to Network Control register to change loop back mode.
3. Write to Network Control register to re-enable transmit or receive circuits.
   Note: These writes to the Network Control register cannot be combined in any way.

24.7.1.2 Receive Buffer List

Receive data is written to areas of data (i.e., buffers) in system memory. These buffers are listed in another data structure that also resides in main memory. This data structure (receive buffer queue) is a sequence of descriptor entries as defined in Table 1-6 “Receive Buffer Descriptor Entry”.

The Receive Buffer Queue Pointer register points to this data structure.

Figure 24-3. Receive Buffer List



To create the list of buffers:

1. Allocate a number (N) of buffers of X bytes in system memory, where X is the DMA buffer length programmed in the DMA Configuration register.
2. Allocate an area 8N bytes for the receive buffer descriptor list in system memory and create N entries in this list. Mark all entries in this list as owned by GMAC, i.e., bit 0 of word 0 set to 0.
3. Mark the last descriptor in the queue with the wrap bit (bit 1 in word 0 set to 1).
4. Write address of receive buffer descriptor list and control information to GMAC register receive buffer queue pointer
5. The receive circuits can then be enabled by writing to the address recognition registers and the Network Control register.

24.7.1.3 Transmit Buffer List

Transmit data is read from areas of data (the buffers) in system memory. These buffers are listed in another data structure that also resides in main memory. This data structure (Transmit Buffer Queue) is a sequence of descriptor entries as defined in Table 1-7 “Transmit Buffer Descriptor Entry”.

The Transmit Buffer Queue Pointer register points to this data structure.

To create this list of buffers:

1. Allocate a number (N) of buffers of between 1 and 2047 bytes of data to be transmitted in system memory. Up to 128 buffers per frame are allowed.
2. Allocate an area 8N bytes for the transmit buffer descriptor list in system memory and create N entries in this list. Mark all entries in this list as owned by GMAC, i.e., bit 31 of word 1 set to 0.
3. Mark the last descriptor in the queue with the wrap bit (bit 30 in word 1 set to 1).

Offset	Name	Bit Pos.								
		23:16	NFTX[23:16]							
		31:24	NFTX[31:24]							
0x0124	TBFT511	7:0	NFTX[7:0]							
		15:8	NFTX[15:8]							
		23:16	NFTX[23:16]							
		31:24	NFTX[31:24]							
0x0128	TBFT1023	7:0	NFTX[7:0]							
		15:8	NFTX[15:8]							
		23:16	NFTX[23:16]							
		31:24	NFTX[31:24]							
0x012C	TBFT1518	7:0	NFTX[7:0]							
		15:8	NFTX[15:8]							
		23:16	NFTX[23:16]							
		31:24	NFTX[31:24]							
0x0130	GTBFT1518	7:0	NFTX[7:0]							
		15:8	NFTX[15:8]							
		23:16	NFTX[23:16]							
		31:24	NFTX[31:24]							
0x0134	TUR	7:0	TXUNR[7:0]							
		15:8							TXUNR[9:8]	
		23:16								
		31:24								
0x0138	SCF	7:0	SCOL[7:0]							
		15:8	SCOL[15:8]							
		23:16							SCOL[17:16]	
		31:24								
0x013C	MCF	7:0	MCOL[7:0]							
		15:8	MCOL[15:8]							
		23:16							MCOL[17:16]	
		31:24								
0x0140	EC	7:0	XCOL[7:0]							
		15:8							XCOL[9:8]	
		23:16								
		31:24								
0x0144	LC	7:0	LCOL[7:0]							
		15:8							LCOL[9:8]	
		23:16								
		31:24								
0x0148	DTF	7:0	DEFT[7:0]							
		15:8	DEFT[15:8]							
		23:16							DEFT[17:16]	
		31:24								
0x014C	CSE	7:0	CSR[7:0]							
		15:8							CSR[9:8]	
		23:16								
		31:24								

31.7.2 Software Event

Name: SWEVT
Offset: 0x04
Reset: 0x00000000
Property: PAC Write-Protection

Bit	31	30	29	28	27	26	25	24
	CHANNEL[31:24]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	CHANNEL[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

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

Bit	7	6	5	4	3	2	1	0
	CHANNEL[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – CHANNEL[31:0] Channel x Software Selection
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will trigger a software event for channel x.

These bits always return '0' when read.

34.8.12 Length

Name: LENGTH
Offset: 0x22
Reset: 0x00
Property: PAC Write-Protection, Write-Synchronized

Bit	15	14	13	12	11	10	9	8
							LENEN[1:0]	
Access							R/W	R/W
Reset							0	0

Bit	7	6	5	4	3	2	1	0
	LEN[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

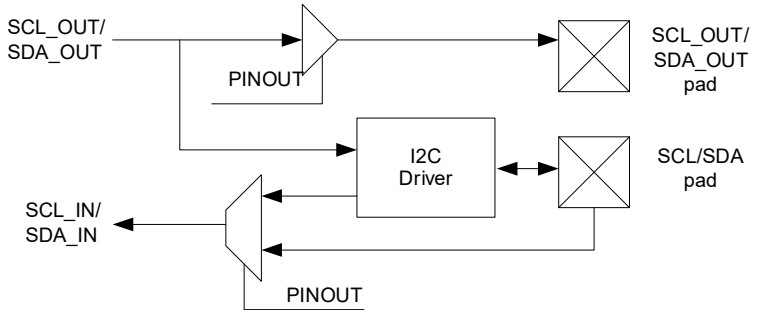
Bits 9:8 – LENEN[1:0] Data Length Enable
In 32-bit Extension mode, this bit field configures the length counter either for transmit or receive transactions.

Value	Description
0x0	Length counter disabled
0x1	Length counter enabled for transmit
0x2	Length counter enabled for receive
0x3	Reserved

Bits 7:0 – LEN[7:0] Data Length
In 32-bit Extension mode, this bit field configures the data length after which the flags INTFLAG.RXC or INTFLAG.DRE are raised.

Value	Description
0x00	Reserved if LENEN=0x1 or LENEN=0x2
0x01-0xFF	Data Length

Figure 36-14. I²C Pad Interface



36.6.3.4 Quick Command

Setting the Quick Command Enable bit in the Control B register (CTRLB.QCEN) enables quick command. When quick command is enabled, the corresponding interrupt flag (INTFLAG.SB or INTFLAG.MB) is set immediately after the slave acknowledges the address. At this point, the software can either issue a stop command or a repeated start by writing CTRLB.CMD or ADDR.ADDR.

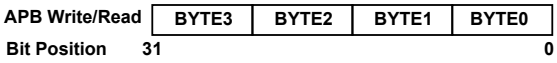
36.6.3.5 32-bit Extension

For better system bus utilization, 32-bit data receive and transmit can be enabled by writing to the Data 32-bit bit field in the Control C register (CTRLC.DATA32B=1). When enabled, write and read transaction to/from the DATA register are 32 bit in size.

If frames are not multiples of 4 Bytes, the Length Counter (LENGTH.LEN) and Length Enable (LENGTH.LENEN) must be configured before data transfer begins. LENGTH.LEN must be enabled only when CTRLC.DATA32B is enabled.

The figure below shows the order of transmit and receive when using 32-bit mode. Bytes are transmitted or received and stored in order from 0 to 3.

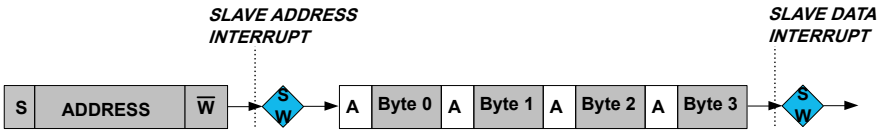
Figure 36-15. 32-bit Extension Byte Ordering



32-bit Extension Slave Operation

The figure below shows a transaction with 32-bit Extension enabled (CTRLC.DATA32B=1). In slave operation, the Address Match interrupt in the Interrupt Flag Status and Clear register (INTFLAG.AMATCH) is set after the address is received and available in the DATA register. The Data Ready interrupt (INTFLAG.DRDY) will then be raised for every 4 Bytes transferred.

Figure 36-16. 32-bit Extension Slave Operation



The LENGTH register can be written before the frame begins, or when the AMATCH interrupt is set. If the frame size is not LENGTH.LEN Bytes, the Length Error status bit (STATUS.LENERR) is raised. If LENGTH.LEN is not a multiple of 4 Bytes, the final INTFLAG.DRDY interrupt is raised when the last Byte is received for master reads. For master writes, the last data byte will be automatically NACKed. On address recognition, the internal length counter is reset in preparation for the incoming frame.

High Speed transactions start with a Full Speed Master Code. When a Master Code is detected, no data is received and the next expected operation is a repeated start. For this reason, the length is not counted

Bit 17 – MRAFE Message RAM Access Failure Interrupt Enable

Value	Description
0	Interrupt disabled.
1	Interrupt enabled.

Bit 16 – TSWE Timestamp Wraparound Interrupt Enable

Value	Description
0	Interrupt disabled.
1	Interrupt enabled.

Bit 15 – TEFLE Tx Event FIFO Event Lost Interrupt Enable

Value	Description
0	Interrupt disabled.
1	Interrupt enabled.

Bit 14 – TEFFE Tx Event FIFO Full Interrupt Enable

Value	Description
0	Interrupt disabled.
1	Interrupt enabled.

Bit 13 – TEFWE Tx Event FIFO Watermark Reached Interrupt Enable

Value	Description
0	Interrupt disabled.
1	Interrupt enabled.

Bit 12 – TEFNE Tx Event FIFO New Entry Interrupt Enable

Value	Description
0	Interrupt disabled.
1	Interrupt enabled.

Bit 11 – TFEE Tx FIFO Empty Interrupt Enable

Value	Description
0	Interrupt disabled.
1	Interrupt enabled.

Bit 10 – TCFE Transmission Cancellation Finished Interrupt Enable

Value	Description
0	Interrupt disabled.
1	Interrupt enabled.

Bit 9 – TCE Transmission Completed Interrupt Enable

Value	Description
0	Interrupt disabled.
1	Interrupt enabled.

43.3.4.12.4 Parameters Definition

Table 43-37. GCD Service Parameters

Parameter	Type	Dir.	Location	Data Length	Before Executing the Service	After Executing the Service
Specific/Gf2n	Bit	I	–	–	GF(2n) Bit	–
nu1XBase	nu1	I	Crypto RAM	u2Length	Base of X Number X	Base of X Filled with the GCD D
u2Length	u2	I	–	–	Length of the Areas X, Y, A, Z	Length of the Areas X, Y, A, Z
nu1YBase	nu1	I	Crypto RAM	u2Length	Base of Y Number Y	Base of Y Cleared area
nu1ABase	nu1	I	Crypto RAM	u2Length	Base of A	Base of A Filled with the result
nu1ZBase	nu1	I	Crypto RAM	u2Length + 4 (see **Note 1**)	Base of Z	Base of Z Filled with the result
nu1WorkSpace	nu1	I	Crypto RAM	32 bytes	Base of the workspace	Base of the workspace corrupted

Note:

1. The additional word is 4 zero bytes.

The parameters X and Y must have their most significant 32-bit word cleared to zero. The length u2Length is the length of the longer of the parameters X and Y including this zero word.

To clarify here is an example:

- X is an 8 bytes number.
- Y is a 12 bytes number.

This example is processed this way before the use of the GCD service:

- The longer number is Y so its length is taken and increased by 4 bytes for the 32-bit word cleared to zero, this gives u2Length = 16 bytes. Therefore, X, Y, A and Z areas have a length of 16 bytes.
- Y is padded with 4 bytes cleared to zero on the MSB side and the u2Length = 16 bytes are written in memory (LSB first).
- X is padded with 8 bytes cleared to zero on the MSB side and the u2Length = 16 bytes are written in memory (LSB first).
- The areas A and Z are mapped in memory with a size of u2Length = 16 bytes.
- The workspace is mapped in memory with its constant size of 32 bytes

43.3.4.12.5 Code Example

```
PUKCL_PARAM PUKCLParam;
PPUKCL_PARAM pvPUKCLParam = &PUKCLParam;
// Fill all the fields
PUKCL(u2Option) = 0;
PUKCL_GCD(nu1XBase) = <Base of the ram location of X>;
PUKCL_GCD(nu1YBase) = <Base of the ram location of Y>;
PUKCL_GCD(nu1ABase) = <Base of the ram location of A>;
PUKCL_GCD(nu1ZBase) = <Base of the ram location of Z>;
```

The DAC can only maintain its output within one LSB of the desired value for approximately 100µs. When a DAC is used to generate a static voltage or at a rate less than 20kSPS, the conversion must be refreshed periodically. The OSCULP32K clock can start new conversions automatically after a specified period. Write a value to the Refresh bit field in the DAC Control x register (DACCTRLx.REFRESH[3:0]) to select the refresh period according to the formula:

$$T_{\mathrm{REFRESH}} = \mathrm{REFRESH} \times T_{\mathrm{OSCULP32K}}$$

The actual period will depend on the tolerance of the OSCULP32K (see Electrical Characteristics).

If DACCTRLx.REFRESH=0, there is no conversion refresh. DACCTRLx.REFRESH=1 is Reserved.

If no new conversion is started before the refresh period is completed, DACx will convert the DATAx value again.

In standby sleep mode, the refresh mode remains enabled if DACCTRLx.RUNSTDBY=1.

If DATAx is written while a refresh conversion is ongoing, the conversion of the new content of DATAx is postponed until DACx is ready to start the next conversion.

47.6.9.4 Differential Mode

DAC0 and DAC1 can be configured to operate in differential mode, i.e. the combined output is a voltage balanced around VREF/2, see also the figure below.

In differential mode, DAC0 and DAC1 are converting synchronously the DATA0 value. DATA0 must therefore be a signed value, represented in two's complement format with DATA0[11] as the signed bit. DATA0 has therefore the range [-2047:2047].

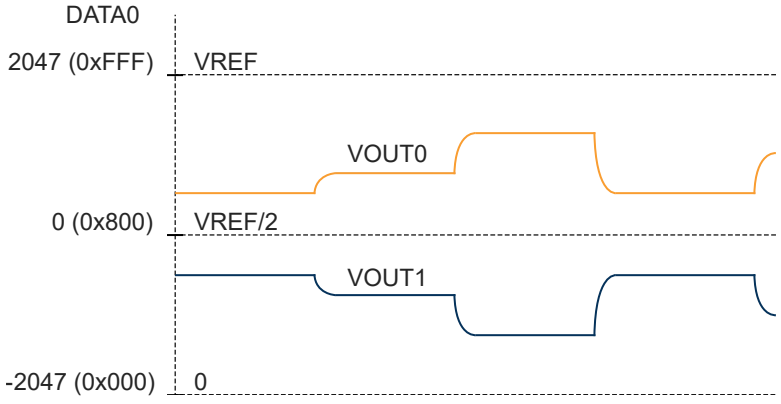
VOUT0 is the positive output and VOUT1 the negative output. The differential output voltage is therefore:

$$V_{\mathrm{OUT}} = \frac{\mathrm{DATA0}}{2047} \times \mathrm{VREF} = (V_{\mathrm{OUT0}} - V_{\mathrm{OUT1}})$$

DACCTRL0 serves as the configuration register for both DAC0 and DAC1. Therefore DACCTRL1 does not need to be written.

The differential mode is enabled by writing a '1' to the Differential bit in the Control B register (CTRLB.DIFF).

Figure 47-4. DAC Conversions in Differential Mode



Value	Description
0	Filter 0 Result Ready interrupt is disabled.
1	Filter 0 Result Ready interrupt is enabled.

Bit 3 – EMPTY1 Data Buffer 1 Empty Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Data Buffer 1 Empty Interrupt Enable bit, which disables the Data Buffer 1 Empty interrupt.

Value	Description
0	The Data Buffer 1 Empty interrupt is disabled.
1	The Data Buffer 1 Empty interrupt is enabled.

Bit 2 – EMPTY0 Data Buffer 0 Empty Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Data Buffer 0 Empty Interrupt Enable bit, which disables the Data Buffer 0 Empty interrupt.

Value	Description
0	The Data Buffer 0 Empty interrupt is disabled.
1	The Data Buffer 0 Empty interrupt is enabled.

Bit 1 – UNDERRUN1 Underrun Interrupt Enable for DAC1
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Data Buffer 1 Underrun Interrupt Enable bit, which disables the Data Buffer 1 Underrun interrupt.

Value	Description
0	The Data Buffer 1 Underrun interrupt is disabled.
1	The Data Buffer 1 Underrun interrupt is enabled.

Bit 0 – UNDERRUN0 Underrun Interrupt Enable for DAC0
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Data Buffer 0 Underrun Interrupt Enable bit, which disables the Data Buffer 0 Underrun interrupt.

Value	Description
0	The Data Buffer 0 Underrun interrupt is disabled.
1	The Data Buffer 0 Underrun interrupt is enabled.

47.8.7 Status

Name: STATUS
Offset: 0x07
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
					EOC1	EOC0	READY1	READY0
Access					R	R	R	R
Reset					0	0	0	0

Bit 3 – EOC1 DAC1 End of Conversion
This bit is cleared when DATA1 register is written.

Value	Description
0	No conversion completed since last load of DATA1.
1	DAC1 conversion is complete, VOUT1 is stable.

Bit 2 – EOC0 DAC0 End of Conversion
This bit is cleared when DATA0 register is written.

Value	Description
0	No conversion completed since last load of DATA0.
1	DAC0 conversion is complete, VOUT0 is stable.

Bit 1 – READY1 DAC1 Startup Ready

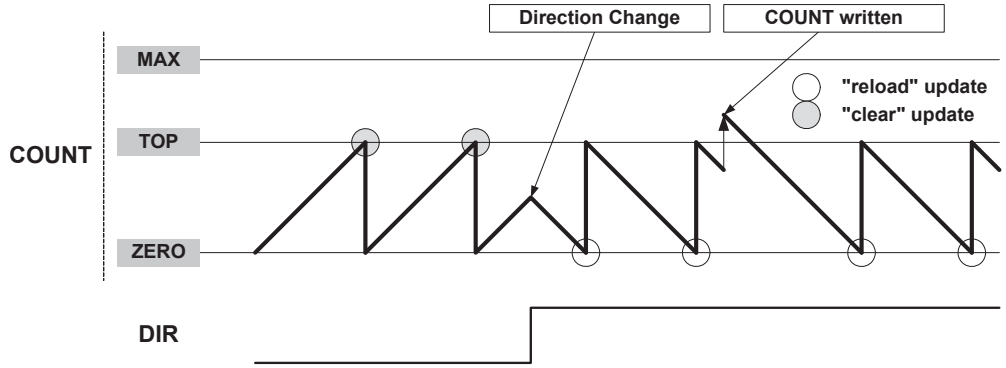
Value	Description
0	DAC1 is not ready for conversion.
1	Startup time has elapsed, DAC1 is ready for conversion.

Bit 0 – READY0 DAC0 Startup Ready

Value	Description
0	DAC0 is not ready for conversion.
1	Startup time has elapsed, DAC0 is ready for conversion.

INTFLAG.OVF can be used to trigger an interrupt, or an event. An overflow/underflow occurrence (i.e. a compare match with TOP/ZERO) will stop counting if the One-Shot bit in the Control B register is set (CTRLBSET.ONESHOT). The One-Shot feature is explained in the Additional Features section.

Figure 49-3. Counter Operation



It is possible to change the counter value (by writing directly in the COUNT register) even when the counter is running. The COUNT value will always be ZERO or TOP, depending on direction set by CTRLBSET.DIR or CTRLBCLR.DIR, when starting the TCC, unless a different value has been written to it, or the TCC has been stopped at a value other than ZERO. The write access has higher priority than count, clear, or reload. The direction of the counter can also be changed during normal operation. See also Figure 49-3.

Stop Command

A stop command can be issued from software by using TCC Command bits in Control B Set register (CTRLBSET.CMD=0x2, STOP).

When a stop is detected while the counter is running, the counter will maintain its current value. If the waveform generation (WG) is used, all waveforms are set to a state defined in Non-Recoverable State x Output Enable bit and Non- Recoverable State x Output Value bit in the Driver Control register (DRVCTRL.NREx and DRVCTRL.NRVx), and the Stop bit in the Status register is set (STATUS.STOP).

Pause Event Action

A pause command can be issued when the stop event action is configured in the Input Event Action 1 bits in Event Control register (EVCTRL.EVACT1=0x3, STOP).

When a pause is detected, the counter can stop immediatly maintaining its current value and all waveforms keep their current state, as long as a start event action is detected: Input Event Action 0 bits in Event Control register (EVCTRL.EVACT0=0x3, START).

Re-Trigger Command and Event Action

A re-trigger command can be issued from software by using TCC Command bits in Control B Set register (CTRLBSET.CMD=0x1, RETRIGGER), or from event when the re-trigger event action is configured in the Input Event 0/1 Action bits in Event Control register (EVCTRL.EVACTn=0x1, RETRIGGER).

When the command is detected during counting operation, the counter will be reloaded or cleared, depending on the counting direction (CTRLBSET.DIR or CTRLBCLR.DIR). The Re-Trigger bit in the Interrupt Flag Status and Clear register will be set (INTFLAG.TRG). It is also possible to generate an event by writing a '1' to the Re-Trigger Event Output Enable bit in the Event Control register (EVCTRL.TRGEO). If the re-trigger command is detected when the counter is stopped, the counter will resume counting operation from the value in COUNT.

Figure 56-14. Cortex Debug Connector (10-pin)

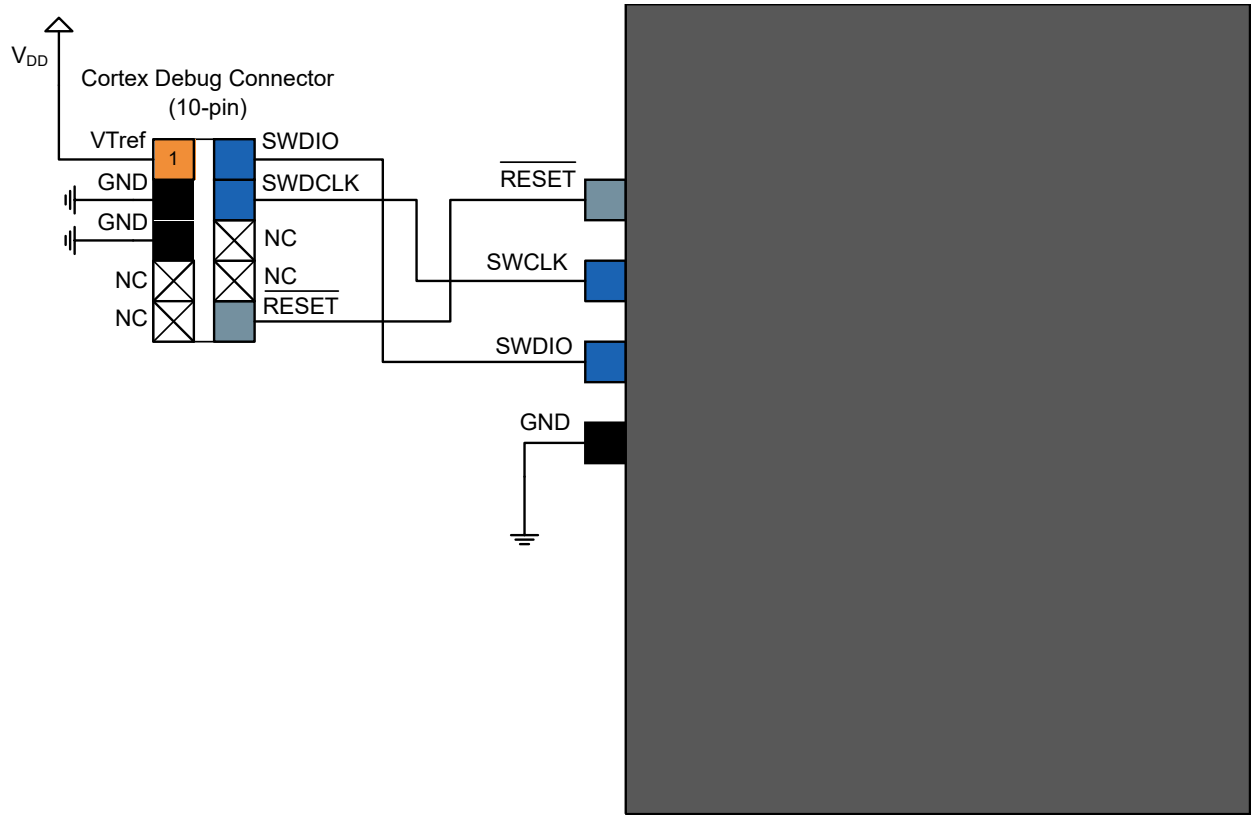


Table 56-8. Cortex Debug Connector (10-pin)

Header Signal Name	Description
SWDCLK	Serial wire clock pin
SWDIO	Serial wire bidirectional data pin
$\overline{\text{RESET}}$	Target device reset pin, active low
VTref	Target voltage sense, should be connected to the device V_{DD}
GND	Ground

56.7.2 20-pin IDC JTAG Connector

For debuggers and/or programmers that support the 20-pin IDC JTAG Connector, e.g. the SAM-ICE, the signals should be connected as shown in Figure 56-15 with details described in Table 56-9.