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	CANbus, I²C, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, DMA, POR, WDT
Number of I/O	52
Program Memory Size	256KB (256K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	32K x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	A/D 20x12b, 3x16b; D/A 1x10b
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/atsamc21j18a-aut

The “set protection” operation will set the write access protection for the peripheral selected by WRCTRL.PERID. Write accesses are not allowed for the registers with write protection property in this peripheral.

The “set and lock protection” operation will set the write access protection for the peripheral selected by WRCTRL.PERID and locks the access rights of the selected peripheral registers. The write access protection will only be cleared by a hardware reset.

The peripheral access control status can be read from the corresponding STATUSn register.

11.5.2.6 Write Access Protection Management Errors

Only word-wise writes to the WRCTRL register will effectively change the access protection. Other type of accesses will have no effect and will cause a PAC write access error. This error is reported in the INTFLAGn.PAC bit corresponding to the PAC module.

PAC also offers an additional safety feature for correct program execution with an interrupt generated on double write clear protection or double write set protection. If a peripheral is write protected and a subsequent set protection operation is detected then the PAC returns an error, and similarly for a double clear protection operation.

In addition, an error is generated when writing a “set and lock” protection to a write-protected peripheral or when a write access is done to a locked set protection. This can be used to ensure that the application follows the intended program flow by always following a write protect with an unprotect and conversely. However in applications where a write protected peripheral is used in several contexts, e.g. interrupt, care should be taken so that either the interrupt can not happen while the main application or other interrupt levels manipulates the write protection status or when the interrupt handler needs to unprotect the peripheral based on the current protection status by reading the STATUS register.

The errors generated while accessing the PAC module registers (eg. key error, double protect error...) will set the INTFLAGn.PAC flag.

11.5.2.7 AHB Slave Bus Errors

The PAC module reports errors occurring at the AHB Slave bus level. These errors are generated when an access is performed at an address where no slave (bridge or peripheral) is mapped . These errors are reported in the corresponding bits of the INTFLAGAHB register.

11.5.2.8 Generating Events

The PAC module can also generate an event when any of the Interrupt Flag registers bit are set. To enable the PAC event generation, the control bit EVCTRL.ERREO must be set a '1'.

11.5.3 DMA Operation

Not applicable.

11.5.4 Interrupts

The PAC has the following interrupt source:

- Error (ERR): Indicates that a peripheral access violation occurred in one of the peripherals controlled by the PAC module, or a bridge error occurred in one of the bridges reported by the PAC
  - This interrupt is a synchronous wake-up source.

Each interrupt source has an interrupt flag associated with it. The interrupt flag in the Interrupt Flag Status and Clear (INTFLAGAHB and INTFLAGn) registers is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a '1' to the corresponding bit in the Interrupt Enable Set (INTENSET) register, and disabled by writing a '1' 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,

Bit 1 – SERCOM0: SERCOM0 APBC Mask Clock Enable

Value	Description
0	The APBC clock for the SERCOM0 is stopped.
1	The APBC clock for the SERCOM0 is enabled.

Bit 0 – EVSYS: EVSYS APBC Clock Enable

Value	Description
0	The APBC clock for the EVSYS is stopped.
1	The APBC clock for the EVSYS is enabled.

17.8.10 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
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				TC7	TC6	TC5	SERCOM7	SERCOM6
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bit 4 – TC7: TC7 APBD Mask Clock Enable

Value	Description
0	The APBD clock for the TC7 is stopped.
1	The APBD clock for the TC7 is enabled.

Bit 3 – TC6: TC6 APBD Mask Clock Enable

Value	Description
0	The APBD clock for the TC6 is stopped.
1	The APBD clock for the TC6 is enabled.

23.6.2.4 Normal Mode

In Normal mode operation, the length of a time-out period is configured in CONFIG.PER. The WDT is enabled by writing a '1' to the Enable bit in the Control A register (CTRLA.ENABLE). Once enabled, the WDT will issue a system reset if a time-out occurs. This can be prevented by clearing the WDT at any time during the time-out period.

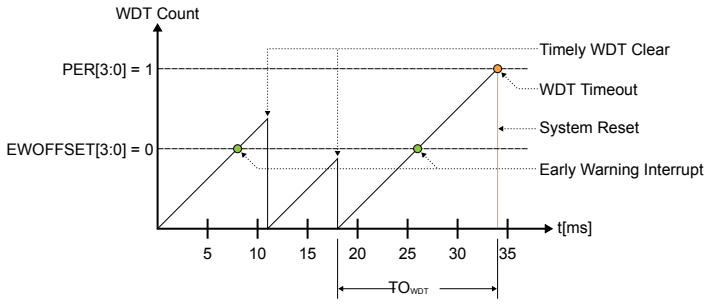
The WDT is cleared and a new WDT time-out period is started by writing 0xA5 to the Clear register (CLEAR). Writing any other value than 0xA5 to CLEAR will issue an immediate system reset.

There are 12 possible WDT time-out (TO_{WDT}) periods, selectable from 8ms to 16s.

By default, the early warning interrupt is disabled. If it is desired, the Early Warning Interrupt Enable bit in the Interrupt Enable register (INTENSET.EW) must be written to '1'. The Early Warning Interrupt is disabled again by writing a '1' to the Early Warning Interrupt bit in the Interrupt Enable Clear register (INTENCLR.EW).

If the Early Warning Interrupt is enabled, an interrupt is generated prior to a WDT time-out condition. In Normal mode, the Early Warning Offset bits in the Early Warning Interrupt Control register, EWCTRL.EWOFFSET, define the time when the early warning interrupt occurs. The Normal mode operation is illustrated in the figure Normal-Mode Operation.

Figure 23-2. Normal-Mode Operation



23.6.2.5 Window Mode

In Window mode operation, the WDT uses two different time specifications: the WDT can only be cleared by writing 0xA5 to the CLEAR register *after* the closed window time-out period (TO_{WDTW}), during the subsequent Normal time-out period (TO_{WDT}). If the WDT is cleared before the time window opens (before TO_{WDTW} is over), the WDT will issue a system reset.

Both parameters TO_{WDTW} and TO_{WDT} are periods in a range from 8ms to 16s, so the total duration of the WDT time-out period is the sum of the two parameters.

The closed window period is defined by the Window Period bits in the Configuration register (CONFIG.WINDOW), and the open window period is defined by the Period bits in the Configuration register (CONFIG.PER).

By default, the Early Warning interrupt is disabled. If it is desired, the Early Warning Interrupt Enable bit in the Interrupt Enable register (INTENSET.EW) must be written to '1'. The Early Warning Interrupt is disabled again by writing a '1' to the Early Warning Interrupt bit in the Interrupt Enable Clear (INTENCLR.EW) register.

If the Early Warning interrupt is enabled in Window mode, the interrupt is generated at the start of the open window period, i.e. after TO_{WDTW}. The Window mode operation is illustrated in figure Window-Mode Operation.

24.6.2.2 Enabling, Disabling, and Resetting

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

The RTC is reset by setting the Software Reset bit in the Control A register (CTRLA.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.

24.6.2.3 32-Bit Counter (Mode 0)

When the RTC Operating Mode bits in the Control A register (CTRLA.MODE) are written to 0x0, the counter operates in 32-bit Counter mode. The block diagram of this mode is shown in Figure 24-1. 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 0 Interrupt 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 A register (CTRLA.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 the prescaler events. Note that when CTRLA.MATCHCLR is '1', INTFLAG.CMP0 and INTFLAG.OVF will both be set simultaneously on a compare match with COMP0.

24.6.2.4 16-Bit Counter (Mode 1)

When the RTC Operating Mode bits in the Control A register (CTRLA.MODE) are written to 0x1, the counter operates in 16-bit Counter mode as shown in Figure 24-2. 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 wrap to 0x0000. 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 16-bit format.

The counter value is continuously compared with the 16-bit Compare registers (COMPn, n=0..1). When a compare match occurs, the Compare n Interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.CMPn, n=0..1) is set on the next 0-to-1 transition of CLK_RTC_CNT.

24.6.2.5 Clock/Calendar (Mode 2)

When the RTC Operating Mode bits in the Control A register (CTRLA.MODE) are written to 0x2, the counter operates in Clock/Calendar mode, as shown in Figure 24-3. When the RTC is enabled, the counter will increment on every 0-to-1 transition of CLK_RTC_CNT. The selected clock source and RTC prescaler must be configured to provide a 1Hz clock to the counter for correct operation in this mode.

The time and date can be read from or written to the Clock Value register (CLOCK) in a 32-bit time/date format. Time is represented as:

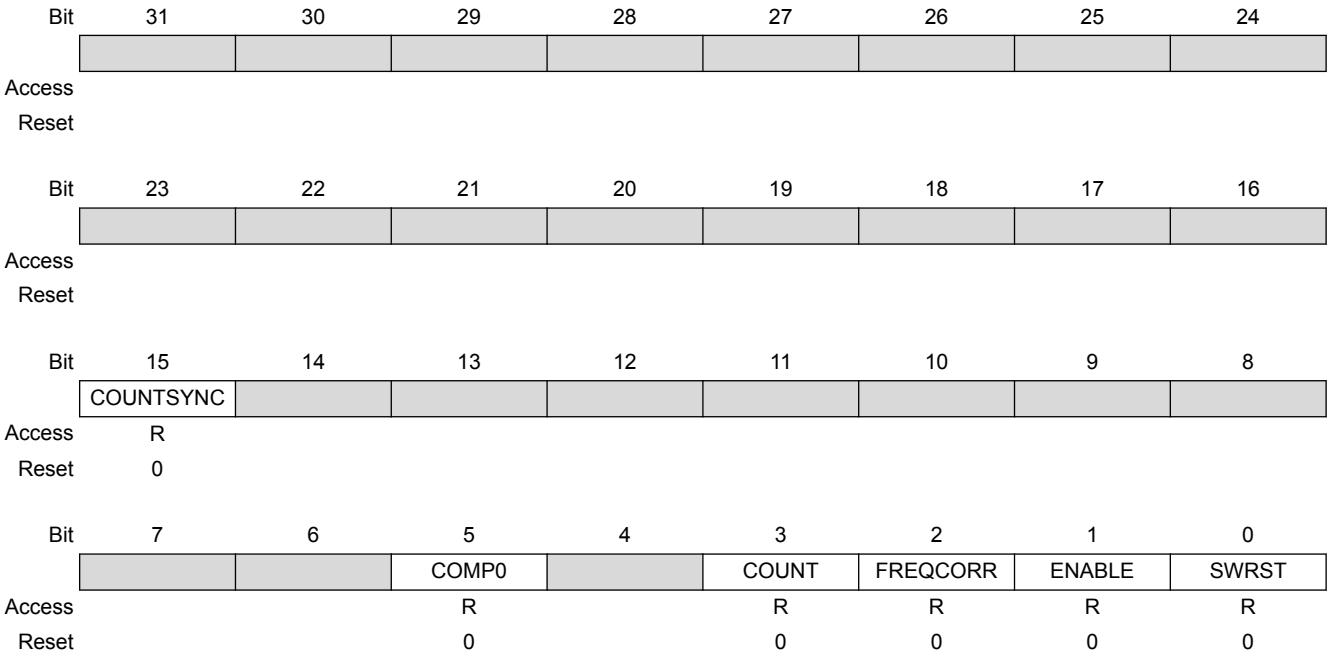
- Seconds
- Minutes
- Hours

This bit controls the functionality when the CPU is halted by an external debugger.

Value	Description
0	The RTC is halted when the CPU is halted by an external debugger.
1	The RTC continues normal operation when the CPU is halted by an external debugger.

24.8.7 Synchronization Busy in COUNT32 mode (CTRLA.MODE=0)

Name: SYNCBUSY
Offset: 0x10
Reset: 0x00000000
Property: -



Bit 15 – COUNTSYNC: Count Read Sync Enable Synchronization Busy Status

Value	Description
0	Write synchronization for CTRLA.COUNTSYNC bit is complete.
1	Write synchronization for CTRLA.COUNTSYNC bit is ongoing.

Bit 5 – COMP0: Compare 0 Synchronization Busy Status

Value	Description
0	Write synchronization for COMP0 register is complete.
1	Write synchronization for COMP0 register is ongoing.

Bit 3 – COUNT: Count Value Synchronization Busy Status

Value	Description
0	Read/write synchronization for COUNT register is complete.
1	Read/write synchronization for COUNT register is ongoing.

Name: COUNT
Offset: 0x18
Reset: 0x0000
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 define the value of the 16-bit RTC counter in COUNT16 mode (CTRLA.MODE=1).

24.10.10 Counter Period in COUNT16 mode (CTRLA.MODE=1)

Name: PER
Offset: 0x1C
Reset: 0x0000
Property: PAC Write-Protection, Write-Synchronized

Bit	15	14	13	12	11	10	9	8
	PER[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
	PER[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 – PER[15:0]: Counter Period
These bits define the value of the 16-bit RTC period in COUNT16 mode (CTRLA.MODE=1).

24.10.11 Compare n Value in COUNT16 mode (CTRLA.MODE=1)

Name: COMP
Offset: 0x20 + n*0x02 [n=0..1]
Reset: 0x0000
Property: PAC Write-Protection, Write-Synchronized

25. DMAC – Direct Memory Access Controller

25.1 Overview

The Direct Memory Access Controller (DMAC) contains both a Direct Memory Access engine and a Cyclic Redundancy Check (CRC) engine. The DMAC can transfer data between memories and peripherals, and thus off-load these tasks from the CPU. It enables high data transfer rates with minimum CPU intervention, and frees up CPU time. With access to all peripherals, the DMAC can handle automatic transfer of data between communication modules.

The DMA part of the DMAC has several DMA channels which all can receive different types of transfer triggers to generate transfer requests from the DMA channels to the arbiter, see also the Block Diagram. The arbiter will grant one DMA channel at a time to act as the active channel. When an active channel has been granted, the fetch engine of the DMAC will fetch a transfer descriptor from the SRAM and store it in the internal memory of the active channel, which will execute the data transmission.

An ongoing data transfer of an active channel can be interrupted by a higher prioritized DMA channel. The DMAC will write back the updated transfer descriptor from the internal memory of the active channel to SRAM, and grant the higher prioritized channel to start transfer as the new active channel. Once a DMA channel is done with its transfer, interrupts and events can be generated optionally.

The DMAC has four bus interfaces:

- The *data transfer bus* is used for performing the actual DMA transfer.
- The *AHB/APB Bridge bus* is used when writing and reading the I/O registers of the DMAC.
- The *descriptor fetch bus* is used by the fetch engine to fetch transfer descriptors before data transfer can be started or continued.
- The *write-back bus* is used to write the transfer descriptor back to SRAM.

All buses are AHB master interfaces but the AHB/APB Bridge bus, which is an APB slave interface.

The CRC engine can be used by software to detect an accidental error in the transferred data and to take corrective action, such as requesting the data to be sent again or simply not using the incorrect data.

25.2 Features

- Data transfer from:
  - Peripheral to peripheral
  - Peripheral to memory
  - Memory to peripheral
  - Memory to memory
- Transfer trigger sources
  - Software
  - Events from Event System
  - Dedicated requests from peripherals
- SRAM based transfer descriptors
  - Single transfer using one descriptor
  - Multi-buffer or circular buffer modes by linking multiple descriptors
- Up to 12 channels

Name: INTSTATUS
Offset: 0x24
Reset: 0x00000000
Property: -

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
					CHINTn	CHINTn	CHINTn	CHINTn
Access					R	R	R	R
Reset					0	0	0	0

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

Bits 11:0 – CHINTn: Channel n Pending Interrupt [n=11..0]
This bit is set when Channel n has a pending interrupt/the interrupt request is received.

This bit is cleared when the corresponding Channel n interrupts are disabled or the interrupts sources are cleared.

25.8.12 Busy Channels

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

26. EIC – External Interrupt Controller

26.1 Overview

The External Interrupt Controller (EIC) allows external pins to be configured as interrupt lines. Each interrupt line can be individually masked and can generate an interrupt on rising, falling, or both edges, or on high or low levels. Each external pin has a configurable filter to remove spikes. Each external pin can also be configured to be asynchronous in order to wake up the device from sleep modes where all clocks have been disabled. External pins can also generate an event.

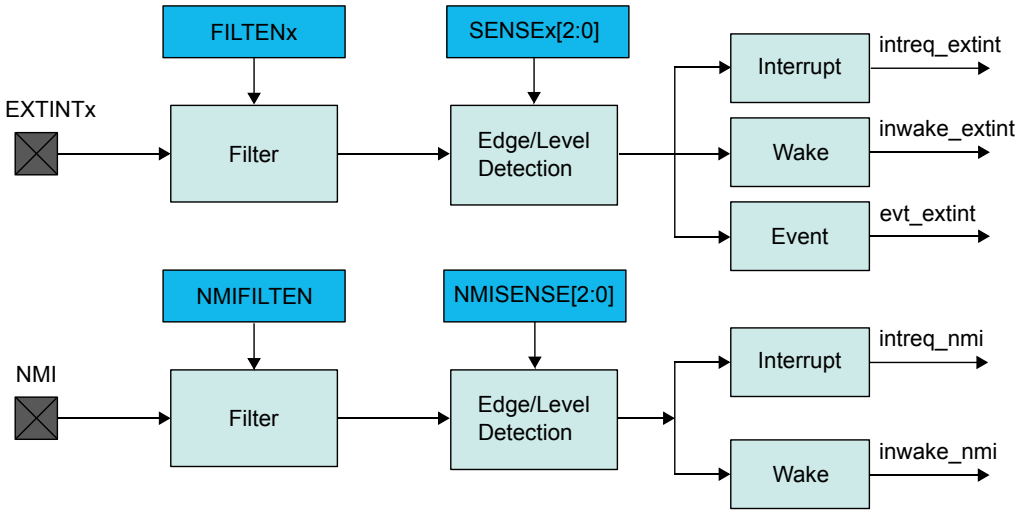
A separate non-maskable interrupt (NMI) is also supported. It has properties similar to the other external interrupts, but is connected to the NMI request of the CPU, enabling it to interrupt any other interrupt mode.

26.2 Features

- Up to 32 external pins (EXTINTx), plus one non-maskable pin (NMI)
- Dedicated, individually maskable interrupt for each pin
- Interrupt on rising, falling, or both edges
- Synchronous or asynchronous edge detection mode
- Interrupt pin debouncing
- Interrupt on high or low levels
- Asynchronous interrupts for sleep modes without clock
- Filtering of external pins
- Event generation from EXTINTx

26.3 Block Diagram

Figure 26-1. EIC Block Diagram



(NMIFLAG.NMI) is set when the last sampled state of the pin differs from the previously sampled state. In this mode, the EIC clock is required.

The Synchronous Edge Detection Mode can be used in Idle and Standby sleep modes.

In *Asynchronous Edge Detection Mode*, the external interrupt (EXTINT) pins or the non-maskable interrupt (NMI) pins set the External Interrupt flag or Non-Maskable Interrupt flag (INTFLAG.EXTINT[x] or NMIFLAG) directly. In this mode, the EIC clock is not requested.

The asynchronous edge detection mode can be used in Idle and Standby sleep modes.

26.6.4.3 Interrupt Pin Debouncing

The external interrupt pin (EXTINT) edge detection can use a debouncer to improve input noise immunity. When selected, the debouncer can work in the synchronous mode or the asynchronous mode, depending on the configuration of the ASYNCH.ASYNCH[x] bit for the pin. The debouncer uses the EIC clock as defined by the bit CTRLA.CKSEL to clock the debouncing circuitry. The debouncing time frame is set with the debouncer prescaler DPRESCALER.DPRESCALERn, which provides the *low frequency clock* tick that is used to reject higher frequency signals.

The debouncing mode for pin EXTINT x can be selected only if the Sense bits in the Configuration y register (CONFIGy.SENSEx) are set to RISE, FALL or BOTH. If the debouncing mode for pin EXTINT x is selected, the filter mode for that pin (CONFIGy.FILTENx) can not be selected.

The debouncer manages an internal “valid pin state” that depends on the external interrupt (EXTINT) pin transitions, the debouncing mode and the debouncer prescaler frequency. The valid pin state reflects the pin value after debouncing. The external interrupt pin (EXTINT) is sampled continously on EIC clock. The sampled value is evaluated on each *low frequency clock* tick to detect a transitional edge when the sampled value is different of the current valid pin state. The sampled value is evaluated on each EIC clock when DPRESCALER.TICKON=0 or on each *low frequency clock* tick when DPRESCALER.TICKON=1, to detect a bounce when the sampled value is equal to the current valid pin state. Transitional edge detection increments the transition counter of the EXTINT pin, while bounce detection resets the transition counter. The transition counter must exceed the transition count threshold as defined by the DPRESCALER.STATESn bitfield. In the synchronous mode the threshold is 4 when DPRESCALER.STATESn=0 or 8 when DPRESCALER.STATESn=1. In the asynchronous mode the threshold is 4.

The valid pin state for the pins can be accessed by reading the register PINSTATE for both synchronous or asynchronous debouncing mode.

Synchronous edge detection In this mode the external interrupt (EXTINT) pin is sampled continously on EIC clock.

1. A pin edge transition will be validated when the sampled value is consistently different of the current valid pin state for 4 (or 8 depending on bit DPRESCALER.STATESn) consecutive ticks of the low frequency clock.
2. Any pin sample, at the *low frequency clock* tick rate, with a value opposite to the current valid pin state will increment the transition counter.
3. Any pin sample, at EIC clock rate (when DPRESCALER.TICKON=0) or the *low frequency clock* tick (when DPRESCALER.TICKON=1), with a value identical to the current valid pin state will return the transition counter to zero.
4. When the transition counter meets the count threshold, the pin edge transition is validated and the pin state PINSTATE.PINSTATE[x] is changed to the detected level.
5. The external interrupt flag (INTFLAG.EXTINT[x]) is set when the pin state PINSTATE.PINSTATE[x] is changed.

Bit	31	30	29	28	27	26	25	24
	OUTSET[31:24]							
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	OUTSET[23:16]							
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	OUTSET[15:8]							
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	OUTSET[7:0]							
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – OUTSET[31:0]: PORT Data Output Value Set
Writing '0' to a bit has no effect.

Writing '1' to a bit will set the corresponding bit in the OUT register, which sets the output drive level high for I/O pins configured as outputs via the Data Direction register (DIR). For pins configured as inputs via Data Direction register (DIR) with pull enabled via the Pull Enable register (PULLEN), these bits will set the input pull direction to an internal pull-up.

Value	Description
0	The corresponding I/O pin in the group will keep its configuration.
1	The corresponding I/O pin output is driven high, or the input is connected to an internal pull-up.

28.9.8 Data Output Value Toggle

This register allows the user to toggle the drive level of one or more output I/O pins, without doing a read-modify-write operation. Changes in this register will also be reflected in the Data Output Value (OUT), Data Output Value Set (OUTSET) and Data Output Value Clear (OUTCLR) registers.

Tip: The I/O pins are assembled in pin groups (”PORT groups”) with up to 32 pins. Group 0 consists of the PA pins, group 1 is for the PB pins, etc. Each pin group has its own PORT registers. For example, the register address offset for the Data Direction (DIR) register for group 0 (PA00 to PA31) is 0x00, and the register address offset for the DIR register for group 1 (PB00 to PB31) is 0x80.

Name: OUTTGL
Offset: 0x1C
Reset: 0x00000000
Property: PAC Write-Protection

held low by the master (STATUS.CLKHOLD is set). An exception is reading the last data byte after the stop condition has been sent.

Accessing DATA.DATA auto-triggers I^2C bus operations. The operation performed depends on the state of CTRLB.ACKACT, CTRLB.SMEN and the type of access (read/write).

Writing or reading DATA.DATA when not in smart mode does not require synchronization.

33.10.11 Debug Control

Name: DBGCTRL
Offset: 0x30 [ID-00001bb3]
Reset: 0x00
Property: PAC Write-Protection

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

Bit 0 – DBGSTOP: Debug Stop Mode
This bit controls functionality when the CPU is halted by an external debugger.

Value	Description
0	The baud-rate generator continues normal operation when the CPU is halted by an external debugger.
1	The baud-rate generator is halted when the CPU is halted by an external debugger.

Table 34-4. Example Filter Configuration for Rx Buffers

Filter Element	SFID1[10:0] / EFID1[28:0]	SFID2[10:9] / EFID2[10:9]	SFID2[5:0] / EFID2[5:0]
0	ID message 1	00	00 0000
1	ID message 2	00	00 0001
2	ID message 3	00	00 0010

After the last word of a matching received message has been written to the Message RAM, the respective New Data flag in register NDAT1, NDAT2 is set. As long as the New Data flag is set, the respective Rx Buffer is locked against updates from received matching frames. The New Data flags have to be reset by the CPU by writing a '1' to the respective bit position.

While an Rx Buffer's New Data flag is set, a Message ID Filter Element referencing this specific Rx Buffer will not match, causing the acceptance filtering to continue. Following Message ID Filter Elements may cause the received message to be stored into another Rx Buffer, or into an Rx FIFO, or the message may be rejected, depending on filter configuration.

Rx Buffer Handling

- Reset interrupt flag IR.DRX
- Read New Data registers
- Read messages from Message RAM
- Reset New Data flags of processed messages

34.6.5.4 Debug on CAN Support

Debug messages are stored into Rx Buffers. For debug handling three consecutive Rx buffers (e.g. #61, #62, #63) have to be used for storage of debug messages A, B, and C. The format is the same as for an Rx Buffer or an Rx FIFO element (see Rx Buffer and FIFO Element).

Advantage: Fixed start address for the DMA transfers (relative to RXBC.RBSA), no additional configuration required.

For filtering of debug messages Standard / Extended Filter Elements with SFEC / EFEC = "111" have to be set up. Messages matching these filter elements are stored into the Rx Buffers addressed by SFID2 / EFID2[5:0].

After message C has been stored, the DMA request output is activated and the three messages can be read from the Message RAM under DMA control. The RAM words holding the debug messages will not be changed by the CAN while DMA request is activated. The behavior is similar to that of an Rx Buffers with its New Data flag set.

After the DMA has completed the DMA unit sets the DMA acknowledge. This resets DMA request. Now the CAN is prepared to receive the next set of debug messages.

Filtering for Debug Messages

Filtering for debug messages is done by configuring one Standard / Extended Message ID Filter Element for each of the three debug messages. To enable a filter element to filter for debug messages SFEC / EFEC has to be programmed to "111". In this case fields SFID1 / SFID2 and EFID1 / EFID2 have a different meaning (see Standard Message ID Filter Element and Extended Message ID Filter Element). While SFID2 / EFID2[10:9] controls the debug message handling state machine, SFID2 / EFID2[5:0] controls the location for storage of a received debug message.

When a debug message is stored, neither the respective New Data flag nor IR.DRX are set. The reception of debug messages can be monitored via RXF1S.DMS.

This register allows the user to clear bits in the CTRLB register without doing a read-modify-write operation. Changes in this register will also be reflected in the Control B Set register (CTRLBSET).

Name: CTRLBCLR
Offset: 0x04
Reset: 0x00
Property: PAC Write-Protection, Read-Synchronized, Write-Synchronized

Bit	7	6	5	4	3	2	1	0
	CMD[2:0]					ONESHOT	LUPD	DIR
Access	R/W	R/W	R/W			R/W	R/W	R/W
Reset	0	0	0			0	0	0

Bits 7:5 – CMD[2:0]: Command
These bits are used for software control of the TC. The commands are executed on the next prescaled GCLK_TC clock cycle. When a command has been executed, the CMD bit group will be read back as zero.

Writing 0x0 to these bits has no effect.

Writing a '1' to any of these bits will clear the pending command.

Bit 2 – ONESHOT: One-Shot on Counter
This bit controls one-shot operation of the TC.

Writing a '0' to this bit has no effect

Writing a '1' to this bit will disable one-shot operation.

Value	Description
0	The TC will wrap around and continue counting on an overflow/underflow condition.
1	The TC will wrap around and stop on the next underflow/overflow condition.

Bit 1 – LUPD: Lock Update
This bit controls the update operation of the TC buffered registers.

When CTRLB.LUPD is set, no any update of the registers with value of its buffered register is performed on hardware UPDATE condition. Locking the update ensures that all buffer registers are valid before an hardware update is performed. After all the buffer registers are loaded correctly, the buffered registers can be unlocked.

This bit has no effect when input capture operation is enabled.

Writing a '0' to this bit has no effect.

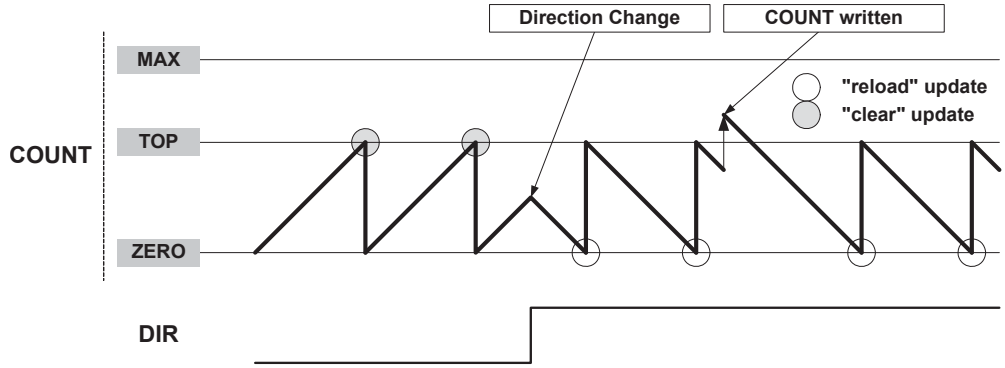
Writing a '1' to this bit will clear the LUPD bit.

Value	Description
0	The CCBUFx and PERBUF buffer registers value are copied into CCx and PER registers on hardware update condition.
1	The CCBUFx and PERBUF buffer registers value are not copied into CCx and PER registers on hardware update condition.

Bit 0 – DIR: Counter Direction
This bit is used to change the direction of the counter.

INTFLAG.OVF can be used to trigger an interrupt, a DMA request, 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).

Figure 36-3. 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 36-3.

Stop Command

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

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).

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).

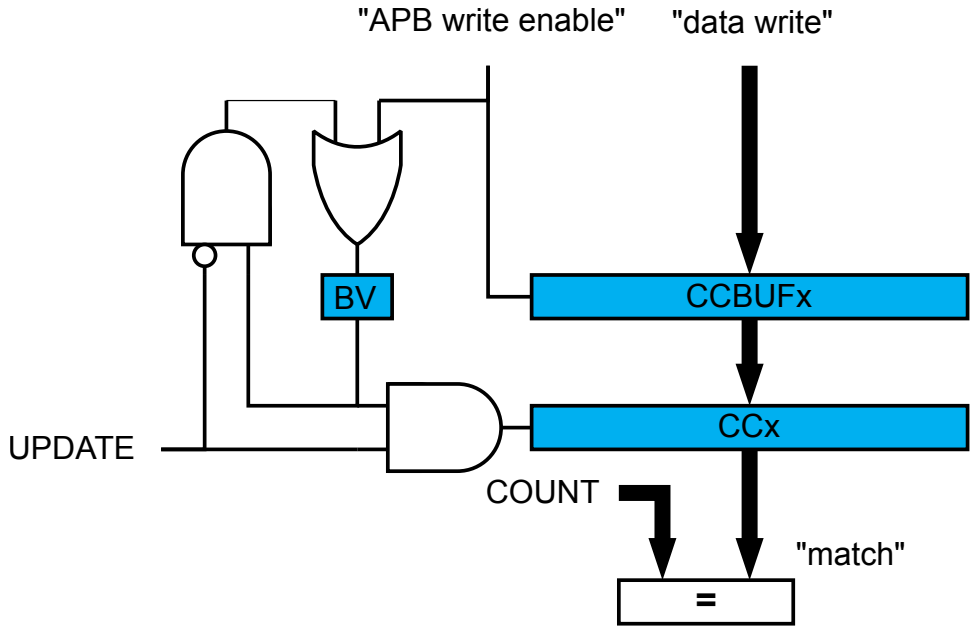
When the command is detected during counting operation, the counter will be reloaded or cleared, depending on the counting direction (CTRLBSET.DIR or CTRLBCLR.DIR). The Re-Trigger bit in the Interrupt Flag Status and Clear register will be set (INTFLAG.TRG). It is also possible to generate an event by 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.

Note:
When a re-trigger event action is configured in the Event Action bits in the Event Control register (EVCTRL.EVACTn=0x1, RETRIGGER), enabling the counter will not start the counter. The counter will start on the next incoming event and restart on corresponding following event.

Start Event Action

The start action can be selected in the Event Control register (EVCTRL.EVACT0=0x3, START) and can start the counting operation when previously stopped. The event has no effect if the counter is already

Figure 36-9. Compare Channel Double Buffering



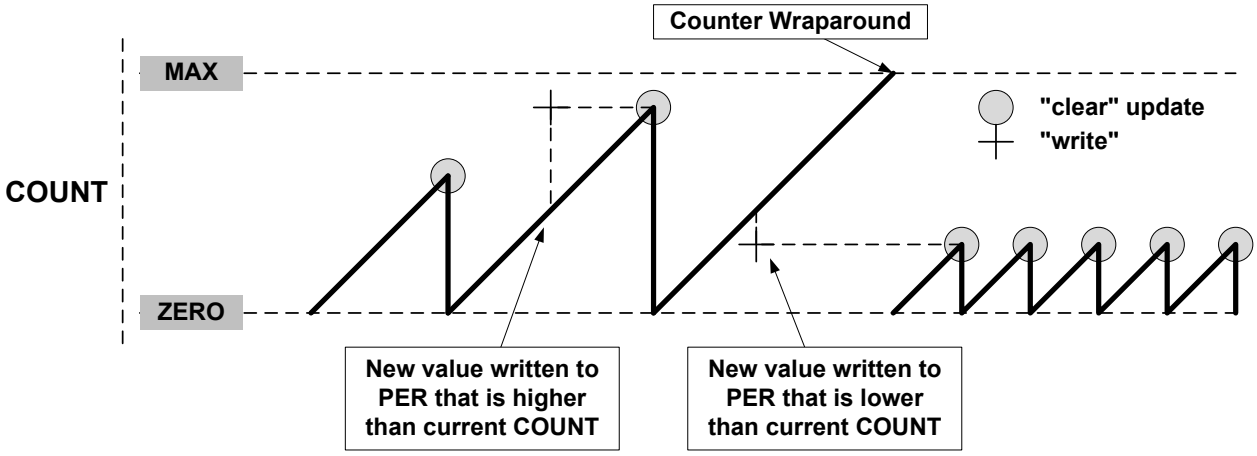
Both the registers (PATT/WAVE/PER/CCx) and corresponding buffer registers (PATTBUF/WAVEBUFV/PERBUF/CCBUFx) are available in the I/O register map, and the double buffering feature is not mandatory. The double buffering is disabled by writing a '1' to CTRLSET.LUPD.

Note: In NFRQ, MFRQ or PWM down-counting counter mode (CTRLBSET.DIR=1), when double buffering is enabled (CTRLBCLR.LUPD=1), PERBUF register is continuously copied into the PER independently of update conditions.

Changing the Period

The counter period can be changed by writing a new Top value to the Period register (PER or CC0, depending on the waveform generation mode), any period update on registers (PER or CCx) is effective after the synchronization delay, whatever double buffering enabling is.

Figure 36-10. Unbuffered Single-Slope Up-Counting Operation



Name: EVCTRL
Offset: 0x02 [ID-00000bc7]
Reset: 0x00
Property: PAC Write-Protection

Bit	7	6	5	4	3	2	1	0
						INVEI	EMPTYEO	STARTEI
Access						R/W	R/W	R/W
Reset						0	0	0

Bit 2 – INVEI: Enable Inversion Data Buffer Empty Event Output
This bit defines the edge detection of the input event for STARTEI.

Value	Description
0	Rising edge.
1	Falling edge.

Bit 1 – EMPTYEO: Data Buffer Empty Event Output
This bit indicates whether or not the Data Buffer Empty event is enabled and will be generated when the Data Buffer register is empty.

Value	Description
0	Data Buffer Empty event is disabled and will not be generated.
1	Data Buffer Empty event is enabled and will be generated.

Bit 0 – STARTEI: Start Conversion Event Input
This bit indicates whether or not the Start Conversion event is enabled and data are loaded from the Data Buffer register to the Data register upon event reception.

Value	Description
0	A new conversion will not be triggered on any incoming event.
1	A new conversion will be triggered on any incoming event.

41.8.4 Interrupt Enable Clear

This register allows the user to disable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Set register (INTENSET).

Name: INTENCLR
Offset: 0x04 [ID-00000bc7]
Reset: 0x00
Property: PAC Write-Protection

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

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

Symbol	Parameters	Conditions	Ta	Typ.	Max	Units
	STANDBY, Mode SAMPL	VDD = 2.7V		0.8	2.1	
		VDD = 5.0V		3.5	4.9	

Note:

1. These values are based on characterization.

Table 46-4. BODVDD Characteristics (see Note 2)

Symbol	Parameters	Conditions	Min	Typ	Max	Unit
VBOD+ (see **Note 1**)	BODVDD high threshold Level	VDD level, BOD setting = 8 (default)	-	2.86	2.98	V
		VDD level, BOD setting = 9	-	2.92	3.01	
		VDD level, BOD setting = 44	-	4.57	4.82	
VBOD- / VBOD (see **Note 1**)	BODVDD low threshold Level	VDD level, BOD setting = 8 (default)	2.71	2.8	2.90	
		VDD level, BOD setting = 9	2.75	2.85	2.96	
		VDD level, Bod setting = 44	4.37	4.51	4.66	
	Step size		-	60	-	mV
VHys (see **Note 1**)	Hysteresis (VBOD+ - VBOD-) BODVDD.LEVEL = 8 to 48	VDD	40	-	75	mV
Tstart (see **Note 3**)	Startup time	Time from enable to RDY	-	3.1	-	µs

Note:

1. These values are based on characterization.
2. BODVDD in Continuous mode.
3. These values are based on simulation, and are not covered by test or characterization.

Related Links

NVM User Row Mapping

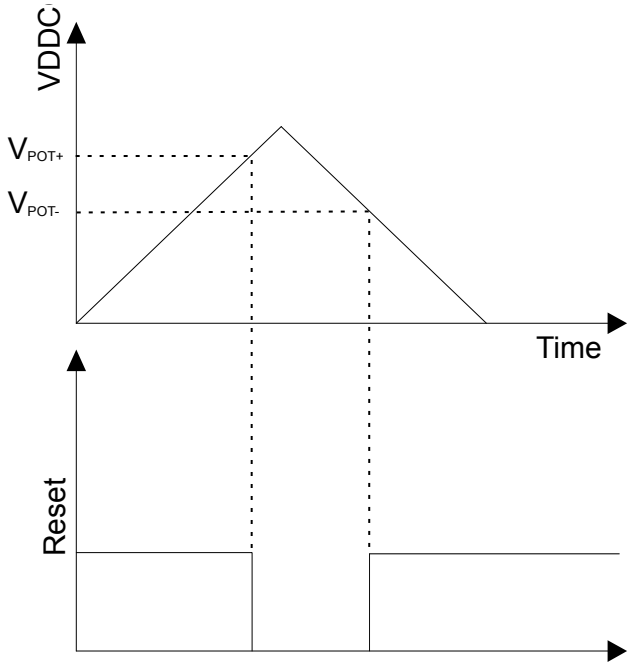
NVM User Row Mapping

46.4.2 Analog-to-Digital Converter (ADC) Characteristics

Table 46-5. Power Consumption(1)

Symbol	Parameters	Conditions	Ta	Typ.	Max	Units
IDD VDDANA	Differential mode	fs = 1 Msps / Reference buffer disabled / BIASREFBUF = '111',	Max 105°C Typ 25°C	905	1034	µA

Figure 47-1. POR Operating Principle



47.4.2 Brown Out Detectors (BOD) Characteristics

See NVM User Row Mapping for the BODVDD default value settings. These values are based on simulation and are not covered by test limits in production or characterization.

Figure 47-2. BODVDD Hysteresis OFF

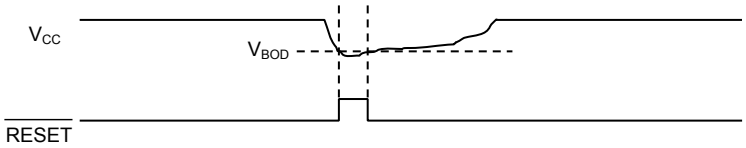


Figure 47-3. BODVDD Hysteresis ON

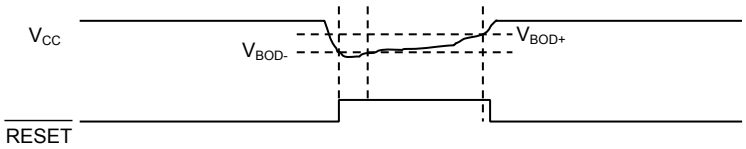


Table 47-4. BODVDD Characteristics(2)

Symbol	Parameters	Conditions	Min	Typ	Max	Unit
VBOD+(1)	BODVDD high threshold Level	VDD level, Bod setting = 8 (default)	-	2.86	2.98	V
		VDD level, Bod setting = 9	-	2.92	3.01	
		VDD level, Bod setting = 44	-	4.57	4.82	
VBOD- / VBOD(1)	BODVDD low threshold Level	VDD level, Bod setting = 8 (default)	2.71	2.80	2.90	
		VDD level, Bod setting = 9	2.75	2.85	2.96	