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, LINbus, SPI, UART/USART, USB
Peripherals	Brown-out Detect/Reset, DMA, I²S, POR, PWM, WDT
Number of I/O	52
Program Memory Size	128KB (128K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	16K x 8
Voltage - Supply (Vcc/Vdd)	1.62V ~ 3.6V
Data Converters	A/D 20x12b; 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/atsamd21j17a-aut

Ordering Code	FLASH (bytes)	SRAM (bytes)	Package	Carrier Type
ATSAMD21E16A-AU	64K	8K	TQFP32	Tray
ATSAMD21E16A-AUT				Tape & Reel
ATSAMD21E16A-AF				Tray
ATSAMD21E16A-AFT				Tape & Reel
ATSAMD21E16A-MU			QFN32	Tray
ATSAMD21E16A-MUT				Tape & Reel
ATSAMD21E16A-MF				Tray
ATSAMD21E16A-MFT				Tape & Reel
ATSAMD21E17A-AU	128K	16K	TQFP32	Tray
ATSAMD21E17A-AUT				Tape & Reel
ATSAMD21E17A-AF				Tray
ATSAMD21E17A-AFT				Tape & Reel
ATSAMD21E17A-MU			QFN32	Tray
ATSAMD21E17A-MUT				Tape & Reel
ATSAMD21E17A-MF				Tray
ATSAMD21E17A-MFT				Tape & Reel
ATSAMD21E18A-AU	256K	32K	TQFP32	Tray
ATSAMD21E18A-AUT				Tape & Reel
ATSAMD21E18A-AF				Tray
ATSAMD21E18A-AFT				Tape & Reel
ATSAMD21E18A-MU			QFN32	Tray
ATSAMD21E18A-MUT				Tape & Reel
ATSAMD21E18A-MF				Tray
ATSAMD21E18A-MFT				Tape & Reel

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.

Bit 17 – AC
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 16 – ADC
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.

Bits 11, 12, 13, 14, 15 – TC3, TC4, TC5, TC6, TC7
Writing a zero to these bits has no effect.

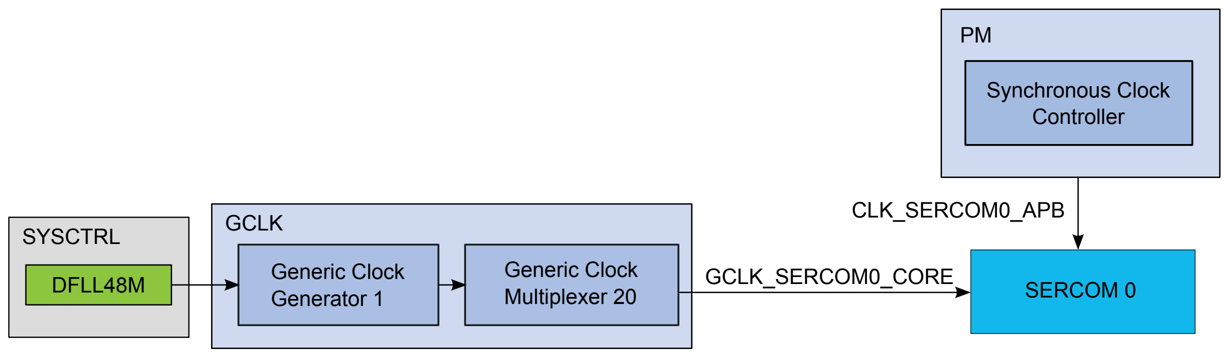
Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

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

Bits 8, 9, 10 – TCCn
Writing a zero to these bits has no effect.

Offset	Name	Bit Pos.								
0x100C ... 0x1FCB	Reserved									
0x1FCC	MEMTYPE	7:0								SMEMP
0x1FCD		15:8								
0x1FCE		23:16								
0x1FCF		31:24								
0x1FD0	PID4	7:0	FKBC[3:0]				JEPCC[3:0]			
0x1FD1		15:8								
0x1FD2		23:16								
0x1FD3		31:24								
0x1FD4 ... 0x1FDF	Reserved									
0x1FE0	PID0	7:0	PARTNBL[7:0]							
0x1FE1		15:8								
0x1FE2		23:16								
0x1FE3		31:24								
0x1FE4	PID1	7:0	JEPIDCL[3:0]				PARTNBH[3:0]			
0x1FE5		15:8								
0x1FE6		23:16								
0x1FE7		31:24								
0x1FE8	PID2	7:0	REVISION[3:0]				JEPU	JEPIDCH[2:0]		
0x1FE9		15:8								
0x1FEA		23:16								
0x1FEB		31:24								
0x1FEC	PID3	7:0	REVAND[3:0]				CUSMOD[3:0]			
0x1FED		15:8								
0x1FEE		23:16								
0x1FEF		31:24								
0x1FF0	CID0	7:0	PREAMBLEB0[7:0]							
0x1FF1		15:8								
0x1FF2		23:16								
0x1FF3		31:24								
0x1FF4	CID1	7:0	CCLASS[3:0]				PREAMBLE[3:0]			
0x1FF5		15:8								
0x1FF6		23:16								
0x1FF7		31:24								
0x1FF8	CID2	7:0	PREAMBLEB2[7:0]							
0x1FF9		15:8								
0x1FFA		23:16								
0x1FFB		31:24								
0x1FFC	CID3	7:0	PREAMBLEB3[7:0]							
0x1FFD		15:8								
0x1FFE		23:16								
0x1FFF		31:24								

Figure 14-2. Example of SERCOM clock



14.2 Synchronous and Asynchronous Clocks

As the CPU and the peripherals can be in different clock domains, i.e. they are clocked from different clock sources and/or with different clock speeds, some peripheral accesses by the CPU need to be synchronized. In this case the peripheral includes a SYNCBUSY status register that can be used to check if a sync operation is in progress.

For a general description, see Register Synchronization. Some peripherals have specific properties described in their individual sub-chapter “Synchronization”.

In the datasheet, references to Synchronous Clocks are referring to the CPU and bus clocks, while asynchronous clocks are generated by the Generic Clock Controller (GCLK).

14.3 Register Synchronization

There are two different register synchronization schemes implemented on this device: *common synchronizer register synchronization* and *distributed synchronizer register synchronization*.

The modules using a common synchronizer register synchronization are: GCLK, WDT, RTC, EIC, TC, ADC, AC and DAC.

The modules adopting a distributed synchronizer register synchronization are: SERCOM USART, SERCOM SPI, SERCOM I2C, I2S, TCC, USB.

14.3.1 Common Synchronizer Register Synchronization

14.3.1.1 Overview

All peripherals are composed of one digital bus interface connected to the APB or AHB bus and running from a corresponding clock in the Main Clock domain, and one peripheral core running from the peripheral Generic Clock (GCLK).

Communication between these clock domains must be synchronized. This mechanism is implemented in hardware, so the synchronization process takes place even if the peripheral generic clock is running from the same clock source and on the same frequency as the bus interface.

All registers in the bus interface are accessible without synchronization. All registers in the peripheral core are synchronized when written. Some registers in the peripheral core are synchronized when read. Each individual register description will have the properties "Read-Synchronized" and/or "Write-Synchronized" if a register is synchronized.

As shown in the figure below, the common synchronizer is used for all registers in one peripheral. Therefore, status register (STATUS) of each peripheral can be synchronized at a time.

Table 16-4. 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.

16.6.3 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	Output clock before the DFLL is locked.
1	Output clock when DFLL is locked.

Bit 10 – BPLCKC: Bypass Coarse Lock
This bit controls the coarse lock procedure:

Value	Description
0	Bypass coarse lock is disabled.
1	Bypass coarse lock is enabled.

Bit 9 – QLDIS: Quick Lock Disable

Value	Description
0	Quick Lock is enabled.
1	Quick Lock is disabled.

Bit 8 – CCDIS: Chill Cycle Disable

Value	Description
0	Chill Cycle is enabled.
1	Chill Cycle is disabled.

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 DFLLCTRL.RUNSTDBY bit is one. If DFLLCTRL.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 DFLL 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 DFLLCTRL.ONDEMAND is one, the clock source will be running when a peripheral is requesting the clock. If DFLLCTRL.ONDEMAND is zero, the clock source will always be running in standby sleep mode.

Bit 5 – USBCRM: USB Clock Recovery Mode

Bit 6 – RUNSTDBY: Run in Standby

Value	Description
0	The BOD33 is disabled in standby sleep mode.
1	The BOD33 is enabled in standby sleep mode.

Bits 4:3 – ACTION[1:0]: BOD33 Action
These bits are used to select the BOD33 action when the supply voltage crosses below the BOD33 threshold.

These bits are loaded from Flash User Row at start-up.

ACTION[1:0]	Name	Description
0x0	NONE	No action
0x1	RESET	The BOD33 generates a reset
0x2	INTERRUPT	The BOD33 generates an interrupt
0x3		Reserved

Bit 2 – HYST: Hysteresis
This bit indicates whether hysteresis is enabled for the BOD33 threshold voltage:

This bit is loaded from Flash User Row at start-up. Refer to *NVM User Row Mapping* for more details.

Value	Description
0	No hysteresis.
1	Hysteresis enabled.

Bit 1 – ENABLE: Enable
This bit is loaded from Flash User Row at startup. Refer to *NVM User Row Mapping* for more details.

Value	Description
0	BOD33 is disabled.
1	BOD33 is enabled.

17.8.15 Voltage Regulator System (VREG) Control

Name: VREG
Offset: 0x3C
Reset: 0x0X00
Property: Write-Protected

Bit	15	14	13	12	11	10	9	8
			FORCELDO					
Access			R/W					
Reset			0					

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

Reset: 0x00X0
Property: PAC Write-Protection, Enable-Protected

Bit	15	14	13	12	11	10	9	8
					LVLEN3	LVLEN2	LVLEN1	LVLEN0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	7	6	5	4	3	2	1	0
						CRCENABLE	DMAENABLE	SWRST
Access						R/W	R/W	R/W
Reset						0	0	0

Bits 8, 9, 10, 11 – LVLENx: Priority Level x Enable
When this bit is set, all requests with the corresponding level will be fed into the arbiter block. When cleared, all requests with the corresponding level will be ignored.

For details on arbitration schemes, refer to the Arbitration section.

These bits are not enable-protected.

Value	Description
0	Transfer requests for Priority level x will not be handled.
1	Transfer requests for Priority level x will be handled.

Bit 2 – CRCENABLE: CRC Enable
Writing a '0' to this bit will disable the CRC calculation when the CRC Status Busy flag is cleared (CRCSTATUS. CRCBUSY). The bit is zero when the CRC is disabled.

Writing a '1' to this bit will enable the CRC calculation.

Value	Description
0	The CRC calculation is disabled.
1	The CRC calculation is enabled.

Bit 1 – DMAENABLE: DMA Enable
Setting this bit will enable the DMA module.

Writing a '0' to this bit will disable the DMA module. When writing a '0' during an ongoing transfer, the bit will not be cleared until the internal data transfer buffer is empty and the DMA transfer is aborted. The internal data transfer buffer will be empty once the ongoing burst transfer is completed.

This bit is not enable-protected.

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 when both the DMAC and the CRC module are disabled (DMAENABLE and CRCENABLE are '0') resets all registers in the DMAC (except DBGCTRL) to their initial state. If either the

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

27.8.3 Baud Rate

Name: BAUD
Offset: 0x0C
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*.

27.8.4 Interrupt Enable Clear

Name: INTENCLR
Offset: 0x14
Reset: 0x00
Property: PAC Write-Protection

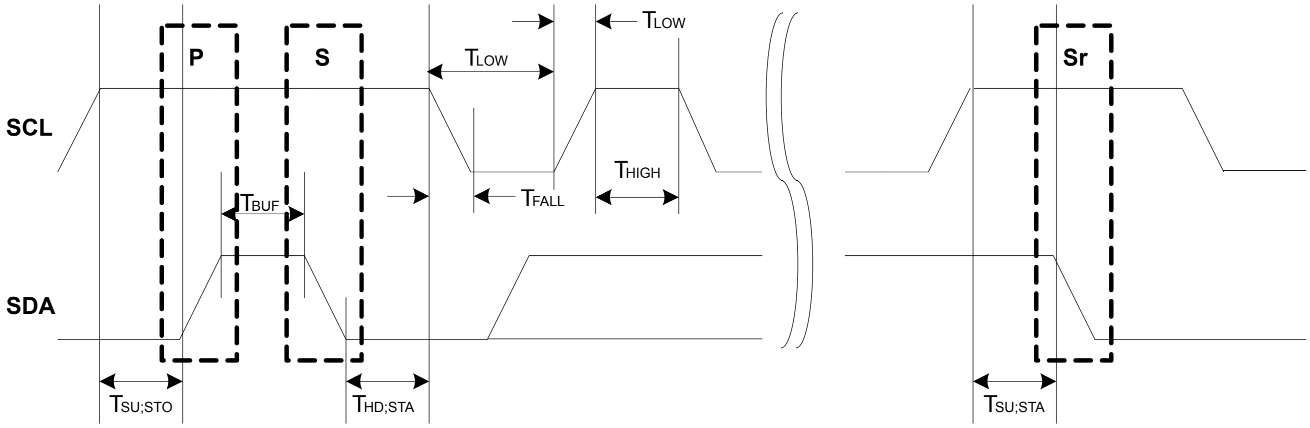
Bit	7	6	5	4	3	2	1	0
	ERROR				SSL	RXC	TXC	DRE
Access	R/W				R/W	R/W	R/W	R/W
Reset	0				0	0	0	0

Bit 7 – ERROR: Error Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the Error Interrupt Enable bit, which disables the Error interrupt.

Value	Description
0	Error interrupt is disabled.
1	Error interrupt is enabled.

Figure 28-6. SCL Timing



The following parameters are timed using the SCL low time period T_{LOW}. This comes from the Master Baud Rate Low bit group in the Baud Rate register (BAUD.BAUDLOW). When BAUD.BAUDLOW=0, or the Master Baud Rate bit group in the Baud Rate register (BAUD.BAUD) determines it.

- T_{LOW} – Low period of SCL clock
- $T_{SU;STO}$ – Set-up time for stop condition
- T_{BUF} – Bus free time between stop and start conditions
- $T_{HD;STA}$ – Hold time (repeated) start condition
- $T_{SU;STA}$ – Set-up time for repeated start condition
- T_{HIGH} is timed using the SCL high time count from BAUD.BAUD
- T_{RISE} is determined by the bus impedance; for internal pull-ups. Refer to *Electrical Characteristics*.
- T_{FALL} is determined by the open-drain current limit and bus impedance; can typically be regarded as zero. Refer to *Electrical Characteristics* for details.

The SCL frequency is given by:

$$f_{\mathrm{SCL}} = \frac{1}{T_{\mathrm{LOW}} + T_{\mathrm{HIGH}} + T_{\mathrm{RISE}}}$$

When BAUD.BAUDLOW is zero, the BAUD.BAUD value is used to time both SCL high and SCL low. In this case the following formula will give the SCL frequency:

$$f_{\mathrm{SCL}} = \frac{f_{\mathrm{GCLK}}}{10 + \ 2BAUD + f_{\mathrm{GCLK}} \cdot T_{\mathrm{RISE}}}$$

When BAUD.BAUDLOW is non-zero, the following formula determines the SCL frequency:

$$f_{\mathrm{SCL}} = \frac{f_{\mathrm{GCLK}}}{10 + \ BAUD + BAUDLOW + f_{\mathrm{GCLK}} \cdot T_{\mathrm{RISE}}}$$

The following formulas can determine the SCL T_{LOW} and T_{HIGH} times:

$$T_{\mathrm{LOW}} = \frac{BAUDLOW + 5}{f_{\mathrm{GCLK}}}$$

$$T_{\mathrm{HIGH}} = \frac{BAUD + 5}{f_{\mathrm{GCLK}}}$$

Note: The I²C standard *Fm+* (Fast-mode plus) requires a nominal high to low SCL ratio of 1:2, and BAUD should be set accordingly. At a minimum, BAUD.BAUD and/or BAUD.BAUDLOW must be non-zero.

arbitration is lost during the transmission. In this case, a lost arbitration will prevent setting INTFLAG.SB. Instead, INTFLAG.MB will indicate a change in arbitration. Handling of lost arbitration is the same as for data bit transmission.

Receiving Data Packets (SCLSM=1)

When INTFLAG.SB is set, the I^{2}C master will already have received one data packet and transmitted an ACK or NACK, depending on CTRLB.ACKACT. At this point, CTRLB.ACKACT must be set to the correct value for the next ACK bit, and the transaction can continue by reading DATA and issuing a command if not in the smart mode.

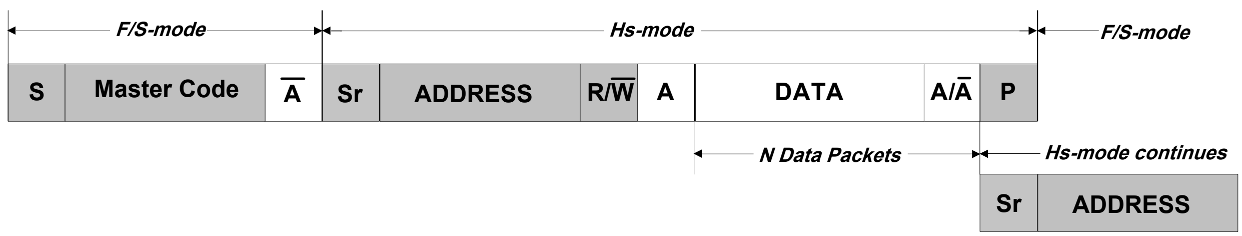
High-Speed Mode

High-speed transfers are a multi-step process, see High Speed Transfer.

First, a master code (0b00001nnn, where 'nnn' is a unique master code) is transmitted in Full-speed mode, followed by a NACK since no slaveshould acknowledge. Arbitration is performed only during the Full-speed Master Code phase. The master code is transmitted by writing the master code to the address register (ADDR.ADDR) and writing the high-speed bit (ADDR.HS) to '0'.

After the master code and NACK have been transmitted, the master write interrupt will be asserted. In the meanwhile, the slave address can be written to the ADDR.ADDR register together with ADDR.HS=1. Now in High-speed mode, the master will generate a repeated start, followed by the slave address with RW-direction. The bus will remain in High-speed mode until a stop is generated. If a repeated start is desired, the ADDR.HS bit must again be written to '1', along with the new address ADDR.ADDR to be transmitted.

Figure 28-7. High Speed Transfer



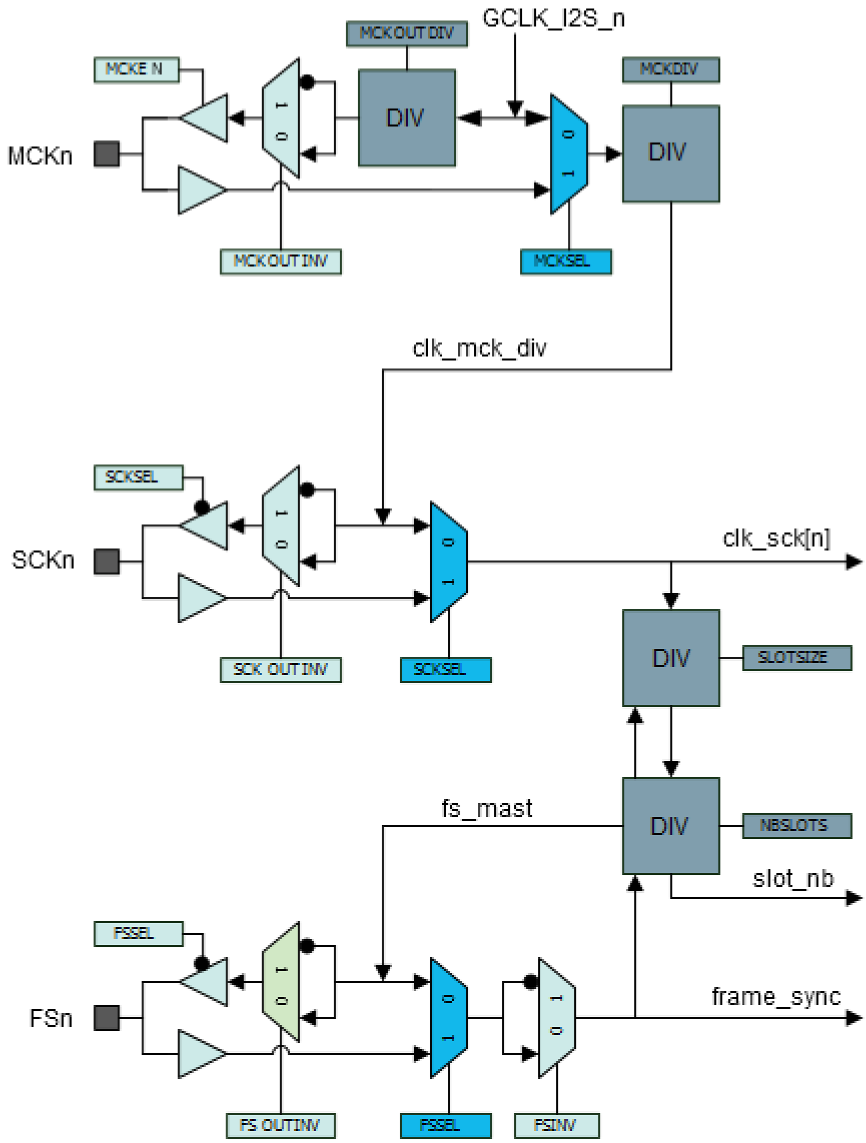
Transmitting in High-speed mode requires the I^{2}C master to be configured in High-speed mode (CTRLA.SPEED=0x2) and the SCL clock stretch mode (CTRLA.SCLSM) bit set to '1'.

10-Bit Addressing

When 10-bit addressing is enabled by the Ten Bit Addressing Enable bit in the Address register (ADDR.TENBITEN=1) and the Address bit field ADDR.ADDR is written, the two address bytes will be transmitted, see 10-bit Address Transmission for a Read Transaction. The addressed slave acknowledges the two address bytes, and the transaction continues. Regardless of whether the transaction is a read or write, the master must start by sending the 10-bit address with the direction bit (ADDR.ADDR[0]) being zero.

If the master receives a NACK after the first byte, the write interrupt flag will be raised and the STATUS.RXNACK bit will be set. If the first byte is acknowledged by one or more slaves, then the master will proceed to transmit the second address byte and the master will first see the write interrupt flag after the second byte is transmitted. If the transaction direction is read-from-slave, the 10-bit address transmission must be followed by a repeated start and the first 7 bits of the address with the read/write bit equal to '1'.

Figure 29-4. I²S Clocks Generation



Slave Mode

In Slave mode, the Serial Clock and Frame Sync (Word Select in I²S mode and Frame Sync in TDM mode) are driven by an external master. SCKn and FSn pins are inputs and no generic clock is required by the I²S.

Master Mode and Controller Mode

In Master Mode, the Master Clock (MCKn), the Serial Clock (SCKn), and the Frame Sync Clock (FSn) are generated by the I²S controller. The user can configure the Master Clock, Serial Clock, and Word Select Frame Sync signal (Word Select in I²S mode and Frame Sync in TDM mode) using the Clock Unit n Control register (CLKCTRLn). MCKn, SCKn, and FSn pins are outputs and a generic clock is used to derive the I²S clocks.

In some applications, audio CODECs connected to the I²S pins may require a Master Clock signal with a frequency multiple of the audio sample frequency *fs*, such as 256×*fs*.

data is also received. For instance, writing SERCTRL0.RXLOOP=1 will connect SD1 output to SD0 input, or writing SERCTRL1.RXLOOP=1 will connect SD0 output to SD1 input.

RXLOOP=1 will connect the Transmitter output of the other Serializer to the Receiver input of the current Serializer. For the Loop-back Mode to work, the current Serializer must be configured as receiver and the other Serializer as transmitter.

Writing SERCTRLm.RXLOOP=0 will restore normal behavior and connection between Serializer m and SDm pin input.

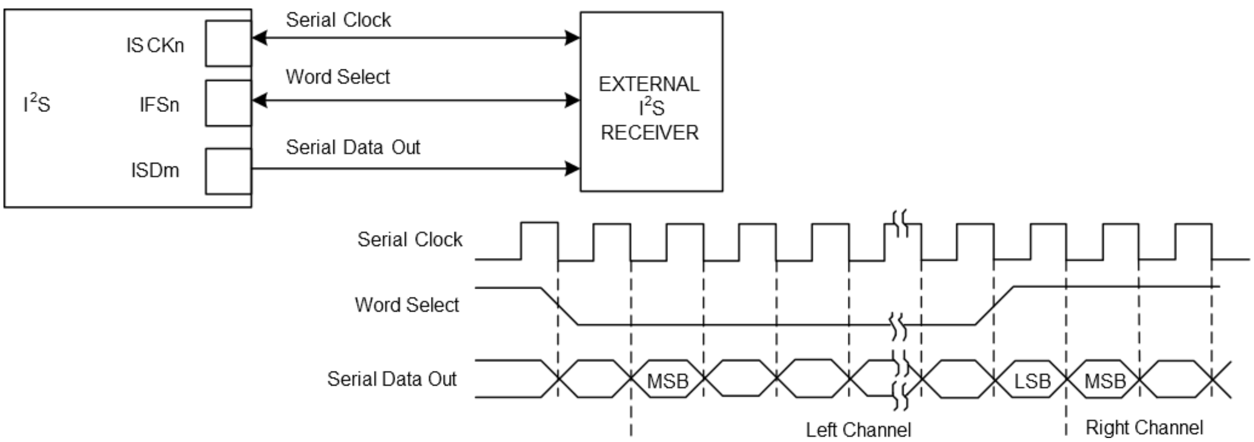
As for other changes to the Serializer configuration, Serializer m must be disabled before writing the SERCTRLm register to update SERCTRLm.RXLOOP.

29.7 I²S Application Examples

The I²S can support several serial communication modes used in audio or high-speed serial links. Some standard applications are shown in the following figures.

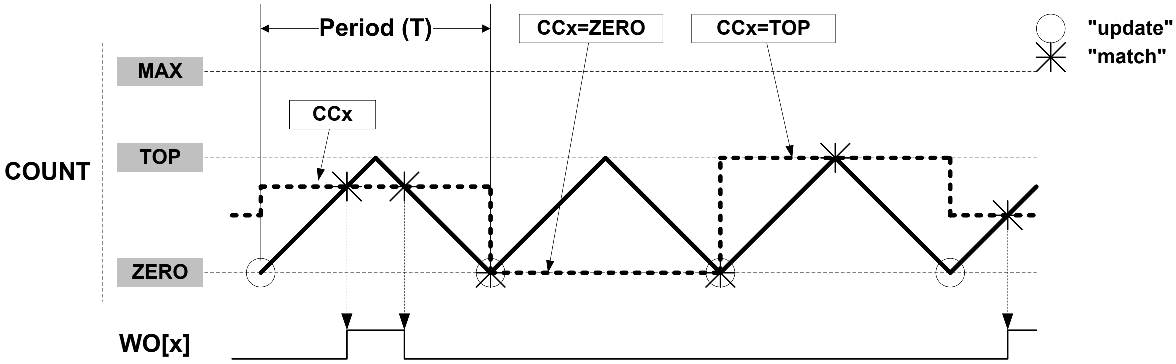
Note: The following examples are not a complete list of serial link applications supported by the I²S.

Figure 29-7. Audio Application Block Diagram



In DSBOTH operation, a second update time occurs on TOP when circular buffer is enabled.

Figure 31-7. Dual-Slope Pulse Width Modulation



Using dual-slope PWM results in a lower maximum operation frequency compared to single-slope PWM generation. The period (TOP) defines the PWM resolution. The minimum resolution is 1 bit (TOP=0x00000001).

The following equation calculates the exact resolution for dual-slope PWM ($R_{\mathrm{PWM\_DS}}$):

$$R_{\mathrm{PWM\_DS}} = \frac{\log(\mathrm{PER}+1)}{\log(2)}.$$

The PWM frequency $f_{\mathrm{PWM\_DS}}$ depends on the period setting (TOP) and the peripheral clock frequency $f_{\mathrm{GCLK\_TCC}}$, and can be calculated by the following equation:

$$f_{\mathrm{PWM\_DS}} = \frac{f_{\mathrm{GCLK\_TCC}}}{2N \cdot \mathrm{PER}}$$

N represents the prescaler divider used. The waveform generated will have a maximum frequency of half of the TCC clock frequency ($f_{\mathrm{GCLK\_TCC}}$) when TOP is set to 0x00000001 and no prescaling is used.

The pulse width ($P_{\mathrm{PWM\_DS}}$) depends on the compare channel (CCx) register value and the peripheral clock frequency ($f_{\mathrm{GCLK\_TCC}}$), and can be calculated by the following equation:

$$P_{\mathrm{PWM\_DS}} = \frac{2N \cdot (\mathrm{TOP} - \mathrm{CCx})}{f_{\mathrm{GCLK\_TCC}}}$$

N represents the prescaler divider used.

Note: In DSTOP, DSBOTTOM and DSBOTH operation, when TOP is lower than MAX/2, the CCx MSB bit defines the ramp on which the CCx Match interrupt or event is generated. (Rising if CCx[MSB]=0, falling if CCx[MSB]=1.)

Related Links

Circular Buffer

Dual-Slope Critical PWM Generation

Dual-Slope Critical PWM Generation

Critical mode generation allows generation of non-aligned centered pulses. In this mode, the period time is controlled by PER while CCx control the generated waveform output edge during up-counting and CC(x+CC_NUM/2) control the generated waveform output edge during down-counting.

Notes:
1. DMA request set on overflow, underflow or re-trigger conditions.
2. Can perform capture or generate recoverable fault on an event input.
3. In capture or circular modes.
4. On event input, either action can be executed:
   – re-trigger counter
   – control counter direction
   – stop the counter
   – decrement the counter
   – perform period and pulse width capture
   – generate non-recoverable fault
5. On event input, either action can be executed:
   – re-trigger counter
   – increment or decrement counter depending on direction
   – start the counter
   – increment or decrement counter based on direction
   – increment counter regardless of direction
   – generate non-recoverable fault

31.6.4.1 DMA Operation

The TCC can generate the following DMA requests:

Counter overflow (OVF): If the Ones-shot Trigger mode in the control A register (CTRLA.DMAOS) is written to '0', the TCC generates a DMA request on each cycle when an update condition (overflow, underflow or re-trigger) is detected.
When an update condition (overflow, underflow or re-trigger) is detected while CTRLA.DMAOS=1, the TCC generates a DMA trigger on the cycle following the DMA One-Shot Command written to the Control B register (CTRLBSET.CMD=DMAOS).

In both cases, the request is cleared by hardware on DMA acknowledge.

Channel Match (MCx): A DMA request is set only on a compare match if CTRLA.DMAOS=0. The request is cleared by hardware on DMA acknowledge.
When CTRLA.DMAOS=1, the DMA requests are not generated.

Channel Capture (MCx): For a capture channel, the request is set when valid data is present in the CCx register, and cleared once the CCx register is read.
In this operation mode, the CTRLA.DMAOS bit value is ignored.

DMA Operation with Circular Buffer

When circular buffer operation is enabled, the buffer registers must be written in a correct order and synchronized to the update times of the timer. The DMA triggers of the TCC provide a way to ensure a safe and correct update of circular buffers.

Note: Circular buffer are intended to be used with RAMP2, RAMP2A and DSBOTH operation only.

DMA Operation with Circular Buffer in RAMP and RAMP2A Mode

When a CCx channel is selected as a circular buffer, the related DMA request is not set on a compare match detection, but on start of ramp B.

Offset: 0x40
Reset: 0xFFFFFFFF
Property: Write-Synchronized

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	PER[17:10]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8
	PER[9:2]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

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

Bits 23:6 – PER[17:0]: Period Value
These bits hold the value of the period buffer register.

Note: When the TCC is configured as 16-bit timer/counter, the excess bits are read zero.

Note: This bit field occupies the MSB of the register, [23:m]. m is dependent on the Resolution bit in the Control A register (CTRLA.RESOLUTION):

CTRLA.RESOLUTION	Bits [23:m]
0x0 - NONE	23:0
0x1 - DITH4	23:4
0x2 - DITH5	23:5
0x3 - DITH6	23:6 (depicted)

Bits 5:0 – DITHER[5:0]: Dithering Cycle Number
These bits hold the number of extra cycles that are added on the PWM pulse period every 64 PWM frames.
Note: This bit field consists of the n LSB of the register. n is dependent on the value of the Resolution bits in the Control A register (CTRLA.RESOLUTION):

CTRLA.RESOLUTION	Bits [n:0]
0x0 - NONE	-
0x1 - DITH4	3:0

Bit 4 – WAKEUP: Wake Up Interrupt Flag
This flag is cleared by writing a one to the flag.

This flag is set when the USB is reactivated by a filtered non-idle signal from the lines and will generate an interrupt if INTENCLR/SET.WAKEUP is one.

Writing a zero to this bit has no effect.

Bit 3 – EORST: End of Reset Interrupt Flag
This flag is cleared by writing a one to the flag.

This flag is set when a USB “End of Reset” has been detected and will generate an interrupt if INTENCLR/SET.EORST is one.

Writing a zero to this bit has no effect.

Bit 2 – SOF: Start-of-Frame Interrupt Flag
This flag is cleared by writing a one to the flag.

This flag is set when a USB “Start-of-Frame” has been detected (every 1 ms) and will generate an interrupt if INTENCLR/SET.SOF is one.

The FNUM is updated. In High Speed mode, the MFNUM register is cleared.

Writing a zero to this bit has no effect.

Bit 0 – SUSPEND: Suspend Interrupt Flag
This flag is cleared by writing a one to the flag.

This flag is set when a USB “Suspend” idle state has been detected for 3 frame periods (J state for 3 ms) and will generate an interrupt if INTENCLR/SET.SUSPEND is one.

Writing a zero to this bit has no effect.

32.8.2.8 Endpoint Interrupt Summary

Name: EPINTSMRY
Offset: 0x20
Reset: 0x0000
Property: -

Bit	15	14	13	12	11	10	9	8
	EPINT[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
	EPINT[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – EPINT[15:0]: EndPoint Interrupt
The flag EPINT[n] is set when an interrupt is triggered by the EndPoint n. See EPINTFLAGn register in the device EndPoint section.

This bit will be cleared when no interrupts are pending for EndPoint n.

Writing this bit to zero will have no effect.

Value	Description
0	No flush action.
1	"Writing a '1' to this bit will flush the ADC pipeline. A flush will restart the ADC clock on the next peripheral clock edge, and all conversions in progress will be aborted and lost. This bit will be cleared after the ADC has been flushed. After the flush, the ADC will resume where it left off; i.e., if a conversion was pending, the ADC will start a new conversion.

33.8.8 Input Control

Name: INPUTCTRL
Offset: 0x10
Reset: 0x00000000
Property: Write-Protected, Write-Synchronized

Bit	31	30	29	28	27	26	25	24
					GAIN[3:0]			
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	23	22	21	20	19	18	17	16
	INPUTOFFSET[3:0]				INPUTSCAN[3:0]			
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

Bit	15	14	13	12	11	10	9	8
				MUXNEG[4:0]				
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

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

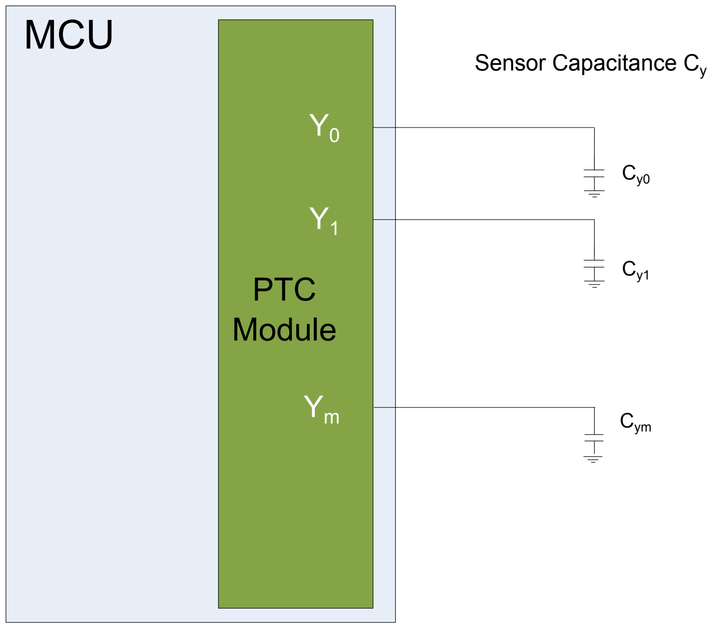
Bits 27:24 – GAIN[3:0]: Gain Factor Selection
These bits set the gain factor of the ADC gain stage.

GAIN[3:0]	Name	Description
0x0	1X	1x
0x1	2X	2x
0x2	4X	4x
0x3	8X	8x
0x4	16X	16x

36.5.1.2 Self-capacitance Sensor Arrangement

The self-capacitance sensor is connected to a single pin on the Peripheral Touch Controller through the Y electrode for receiving the signal. The sense electrode capacitance is measured by the Peripheral Touch Controller.

Figure 36-4. Self-capacitance Sensor Arrangement



For more information about designing the touch sensor, refer to Buttons, Sliders and Wheels Touch Sensor Design Guide on http://www.atmel.com.

36.5.2 Clocks

The PTC is clocked by the GCLK_PTC clock. The PTC operates from an asynchronous clock source and the operation is independent of the main system clock and its derivative clocks, such as the peripheral bus clock (CLK_APB). A number of clock sources can be selected as the source for the asynchronous GCLK_PTC. The clock source is selected by configuring the Generic Clock Selection ID in the Generic Clock Control register. For more information about selecting the clock sources, refer to *GCLK - Generic Clock Controller*.

The selected clock must be enabled in the Power Manager, before it can be used by the PTC. By default these clocks are disabled. The frequency range of GCLK_PTC is 400kHz to 4MHz.

Related Links

GCLK - Generic Clock Controller

PM – Power Manager

36.6 Functional Description

In order to access the PTC, the user must use the QTouch Composer tool to configure and link the QTouch Library firmware with the application code. QTouch Library can be used to implement buttons, sliders, wheels in a variety of combinations on a single interface.

The SAM D21 oscillator is optimized for very low power consumption, hence close attention should be made when selecting crystals, see the table below for maximum ESR recommendations on 9pF and 12.5pF crystals.

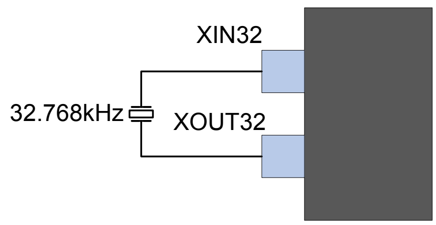
The Low-frequency Crystal Oscillator provides an internal load capacitance of typical values available in Table , *32kHz Crystal Oscillator Characteristics*. This internal load capacitance and PCB capacitance can allow to use a Crystal inferior to 12.5pF load capacitance without external capacitors as shown in the following figure.

Table 39-6. Maximum ESR Recommendation for 32.768kHz Crystal

Crystal C_L (pF)	Max ESR [kΩ]
12.5	313

Note: Maximum ESR is typical value based on characterization. These values are not covered by test limits in production.

Figure 39-7. External Real Time Oscillator without Load Capacitor



However, to improve Crystal accuracy and Safety Factor, it can be recommended by crystal datasheet to add external capacitors as shown in the next figure.

To find suitable load capacitance for a 32.768kHz crystal, consult the crystal datasheet.

Figure 39-8. External Real Time Oscillator with Load Capacitor

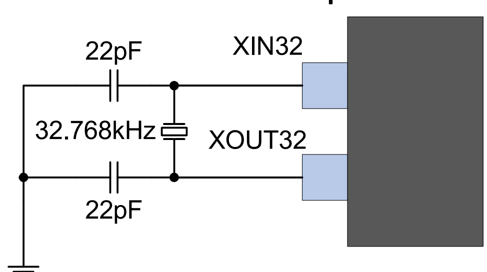


Table 39-7. External Real Time Oscillator Checklist

Signal Name	Recommended Pin Connection	Description
XIN32	Load capacitor 22pF(1)(2)	Timer oscillator input
XOUT32	Load capacitor 22pF(1)(2)	Timer oscillator output

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

Note: In order to minimize the cycle-to-cycle jitter of the external oscillator, keep the neighboring pins as steady as possible. For neighboring pin details, refer to the Oscillator Pinout section.

Related Links