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	AVR
Core Size	8-Bit
Speed	20MHz
Connectivity	I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	12
Program Memory Size	2KB (2K x 8)
Program Memory Type	FLASH
EEPROM Size	64 x 8
RAM Size	128 x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 5.5V
Data Converters	A/D 10x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	14-SOIC (0.154", 3.90mm Width)
Supplier Device Package	14-SOIC
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/attiny204-ssnr

6.10.1 SIGROW - Signature Row Summary

Offset	Name	Bit Pos.								
0x00	DEVICEID0	7:0	DEVICEID[7:0]							
0x01	DEVICEID1	7:0	DEVICEID[7:0]							
0x02	DEVICEID2	7:0	DEVICEID[7:0]							
0x03	SERNUM0	7:0	SERNUM[7:0]							
0x04	SERNUM1	7:0	SERNUM[7:0]							
0x05	SERNUM2	7:0	SERNUM[7:0]							
0x06	SERNUM3	7:0	SERNUM[7:0]							
0x07	SERNUM4	7:0	SERNUM[7:0]							
0x08	SERNUM5	7:0	SERNUM[7:0]							
0x09	SERNUM6	7:0	SERNUM[7:0]							
0x0A	SERNUM7	7:0	SERNUM[7:0]							
0x0B	SERNUM8	7:0	SERNUM[7:0]							
0x0C	SERNUM9	7:0	SERNUM[7:0]							
0x0D ... 0x21	Reserved									
0x22	OSC16ERR3V	7:0	OSC16ERR3V[7:0]							
0x23	OSC16ERR5V	7:0	OSC16ERR5V[7:0]							
0x24	OSC20ERR3V	7:0	OSC20ERR3V[7:0]							
0x25	OSC20ERR5V	7:0	OSC20ERR5V[7:0]							

6.10.2 Signature Row Description

Vector Number	Peripheral Source	Definition
16	-	-
17	AC0_AC	AC0 – Analog Comparator
18	-	-
19	-	-
20	ADC0_RESRDY	ADC0 – Analog-to-Digital Converter, RESRDY
21	ADC0_WCOMP	ADC0, WCOMP
22	-	-
23	-	-
24	TWI0_TWIS	TWI0 - Two-Wire Interface/I^{2}C, TWIS
25	TWI0_TWIM	TWI0, TWIM
26	SPI0_INT	SPI0 - Serial Peripheral Interface
27	USART0_RXC	USART0 - Universal Asynchronous Receiver-Transmitter, RXC
28	USART0_DRE	USART0, DRE
29	USART0_TXC	USART0, TXC
30	NVMCTRL_EE	NVM - Nonvolatile Memory

Related Links

Nonvolatile Memory Controller (NVMCTRL)
I/O Pin Configuration (PORT)
Real-Time Counter (RTC)
Serial Peripheral Interface (SPI)
Universal Synchronous and Asynchronous Receiver and Transmitter (USART)
Two-Wire Interface (TWI)
Cyclic Redundancy Check Memory Scan (CRCSCAN)
16-bit Timer/Counter Type A (TCA)
16-bit Timer/Counter Type B (TCB)
Analog Comparator (AC)
Analog-to-Digital Converter (ADC)

7.3 System Configuration (SYSCFG)

The system configuration contains the revision ID of the part. The revision ID is readable from the CPU, making it useful for implementing application changes between part revisions.

12. Reset Controller (RSTCTRL)

12.1 Features

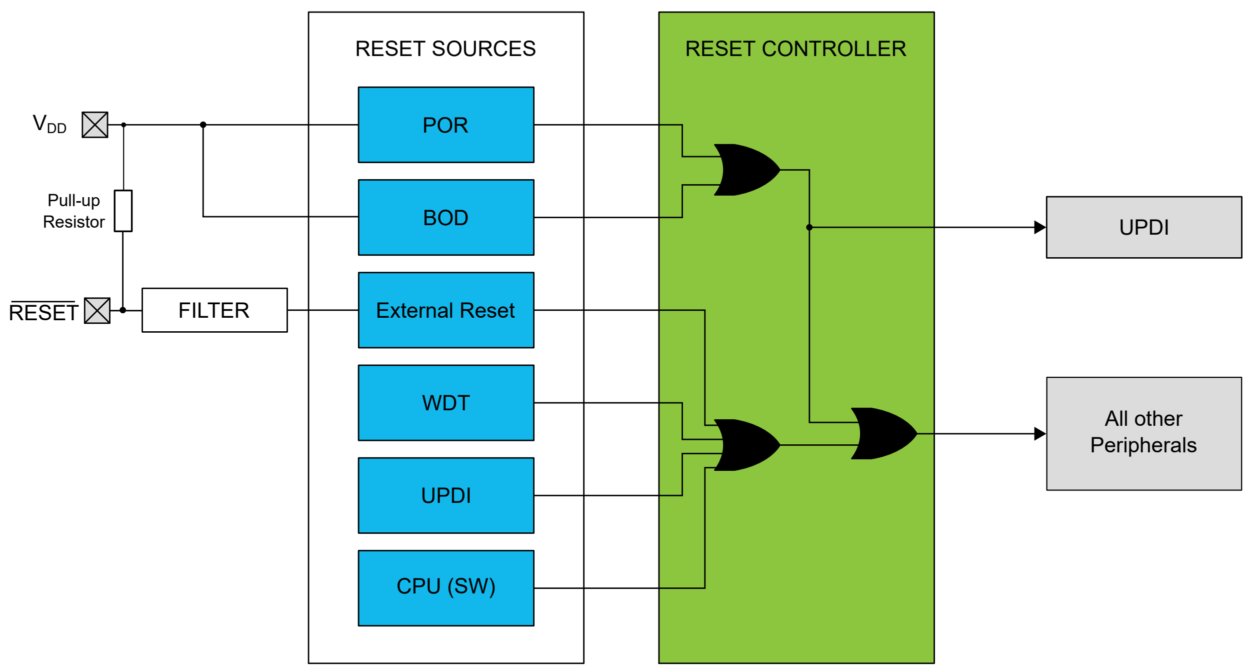
- Reset the device and set it to an initial state.
- Reset Flag register for identifying the Reset source in software.
- Multiple Reset sources:
  - Power supply Reset sources: Brown-out Detect (BOD), Power-on Reset (POR)
  - User Reset sources: External Reset pin ($\overline{\text{RESET}}$), Watchdog Reset (WDT), Software Reset (SW), and UPDI Reset.

12.2 Overview

The Reset Controller (RSTCTRL) manages the Reset of the device. It issues a device Reset, sets the device to its initial state, and allows the Reset source to be identified by the software.

12.2.1 Block Diagram

Figure 12-1. Reset System Overview



12.2.2 Signal Description

Signal	Description	Type
$\overline{\text{RESET}}$	External Reset (active-low)	Digital input

16. I/O Pin Configuration (PORT)

16.1 Features

- General Purpose Input and Output Pins with Individual Configuration
- Output Driver with Configurable Inverted I/O and Pullup
- Input with Interrupts and Events:
  - Sense both edges
  - Sense rising edges
  - Sense falling edges
  - Sense low level
- Asynchronous Pin Change Sensing That Can Wake the Device From all Sleep Modes
- Efficient and Safe Access to Port Pins
  - Hardware read-modify-write through dedicated toggle/clear/set registers
  - Mapping of often-used PORT registers into bit-accessible I/O memory space (virtual ports)

16.2 Overview

The I/O pins of the device are controlled by instances of the port peripheral registers. This device has the following instances of the I/O pin configuration (PORT): PORTA, PORTB.

Refer to the I/O Multiplexing table to see which pins are controlled by what instance of port. The offsets of the port instances and of the corresponding virtual port instances are listed in the Peripherals and Architecture section.

Each of the port pins has a corresponding bit in the Data Direction (PORT.DIR) and Data Output Value (PORT.OUT) registers to enable that pin as an output and to define the output state. For example, pin PA3 is controlled by DIR[3] and OUT[3] of the PORTA instance.

The Data Input Value (PORT.IN) is set as the input value of a port pin with resynchronization to the main clock. To reduce power consumption, these input synchronizers are not clocked if the Input Sense Configuration bit field (ISC) in PORT.PINnCTRL is INPUT_DISABLE. The value of the pin can always be read, whether the pin is configured as input or output.

The port also supports synchronous and asynchronous input sensing with interrupts for selectable pin change conditions. Asynchronous pin-change sensing means that a pin change can wake the device from all sleep modes, including the modes where no clocks are running.

All pin functions are configurable individually per pin. The pins have hardware read-modify-write (RMW) functionality for a safe and correct change of drive value and/or pull resistor configuration. The direction of one port pin can be changed without unintentionally changing the direction of any other pin.

The port pin configuration also controls input and output selection of other device functions.

Related Links

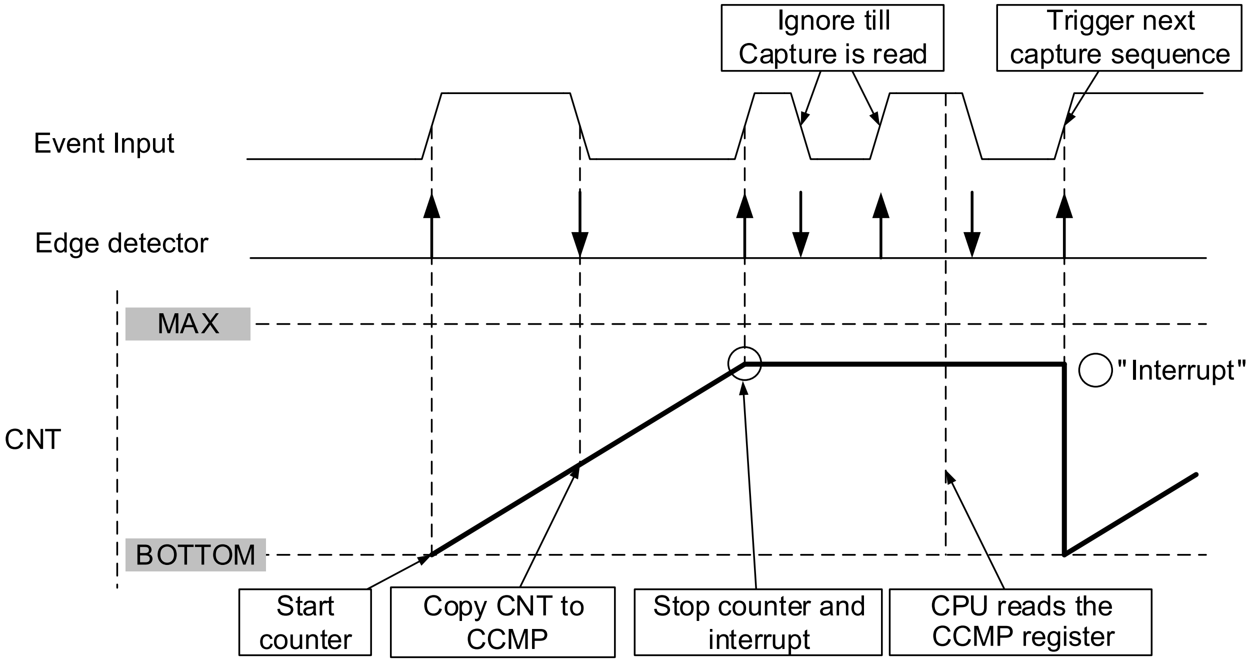
I/O Multiplexing and Considerations
Peripherals and Architecture

18.4 Register Summary - VREF

Offset	Name	Bit Pos.								
0x00	CTRLA	7:0		ADC0REFSEL[2:0]				DAC0REFSEL[2:0]		
0x01	CTRLB	7:0							ADC0REFEN	DAC0REFEN

18.5 Register Description

Figure 21-7. Input Capture Frequency and Pulse-Width Measurement



Single-Shot Mode

This mode can be used to generate a pulse with a duration that is defined by the Compare register (TCBn.CCMP), every time a rising or falling edge is observed on a connected event channel.

When the counter is stopped, the output pin is driven to low. If an event is detected on the connected event channel, the timer will reset and start counting from zero to TOP while driving its output high. The RUN bit in the STATUS register can be read to see if the counter is counting or not. When the counter register reaches the CCMP register value, the counter will stop and the output pin will go low for at least one prescaler cycle. If a new event arrives during this time, that event will be ignored. The following figure shows an example waveform. There is a two clock cycle delay from when the event is received until the output is set high.

The counter will start counting as soon as the module is enabled, even without triggering an event. This is prevented by writing TOP to the counter register. Similar behavior is seen if the EDGE bit in the TCBn.EVCTRL register is '1' while the module is enabled. Writing TOP to the Counter register prevents this as well.

If the ASYNC bit in TCBn.CTRLB is written to '1', the timer is reacting asynchronously to an incoming event. An edge on the event will immediately cause the output signal to be set. The counter will still start counting two clock cycles after the event is received.

Count Mode	EDGE	Positive Edge	Negative Edge
	1	Start counter	Start counter
8-Bit PWM mode	0	-	-
	1	-	-

Bit 0 – CAPTEI Capture Event Input Enable
Writing this bit to '1' enables the input capture event.

21.5.7 Debug Control

Name: DBGCTRL
Offset: 0x08
Reset: 0x00
Property: -

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

Bit 0 – DBGRUN Debug Run

Value	Description
0	The peripheral is halted in Break Debug mode and ignores events
1	The peripheral will continue to run in Break Debug mode when the CPU is halted

21.5.10 Capture/Compare

Name: CCMP
Offset: 0x0C
Reset: 0x00
Property: -

The TCBn.CCMPL and TCBn.CCMPH register pair represents the 16-bit value TCBn.CCMP. The low byte [7:0] (suffix L) is accessible at the original offset. The high byte [15:8] (suffix H) can be accessed at offset + 0x01.

This register has different functions depending on the mode of operation:

- For capture operation, these registers contain the captured value of the counter at the time the capture occurs
- In periodic interrupt/time-out and Single-Shot mode, this register acts as the TOP value
- In 8-bit PWM mode, TCBn.CCMPL and TCBn.CCMPH act as two independent registers

Bit	15	14	13	12	11	10	9	8
	CCMP[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
	CCMP[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 15:8 – CCMP[15:8] Capture/Compare Value High Byte
These bits hold the MSB of the 16-bit compare, capture, and top value.

Bits 7:0 – CCMP[7:0] Capture/Compare Value Low Byte
These bits hold the LSB of the 16-bit compare, capture, and top value.

22.10 Register Summary - RTC

Offset	Name	Bit Pos.								
0x00	CTRLA	7:0	RUNSTDBY	PRESCALER[3:0]						RTCEN
0x01	STATUS	7:0					CMPBUSY	PERBUSY	CNTBUSY	CTRLABUSY
0x02	INTCTRL	7:0							CMP	OVF
0x03	INTFLAGS	7:0							CMP	OVF
0x04	TEMP	7:0	TEMP[7:0]							
0x05	DBGCTRL	7:0								DBGRUN
0x06	Reserved									
0x07	CLKSEL	7:0							CLKSEL[1:0]	
0x08	CNT	7:0	CNT[7:0]							
		15:8	CNT[15:8]							
0x0A	PER	7:0	PER[7:0]							
		15:8	PER[15:8]							
0x0C	CMP	7:0	CMP[7:0]							
		15:8	CMP[15:8]							
0x0E ... 0x0F	Reserved									
0x10	PITCTRLA	7:0		PERIOD[3:0]						PITEN
0x11	PITSTATUS	7:0								CTRLBUSY
0x12	PITINTCTRL	7:0								PI
0x13	PITINTFLAGS	7:0								PI
0x14	Reserved									
0x15	PITDBGCTRL	7:0								DBGRUN

22.11 Register Description

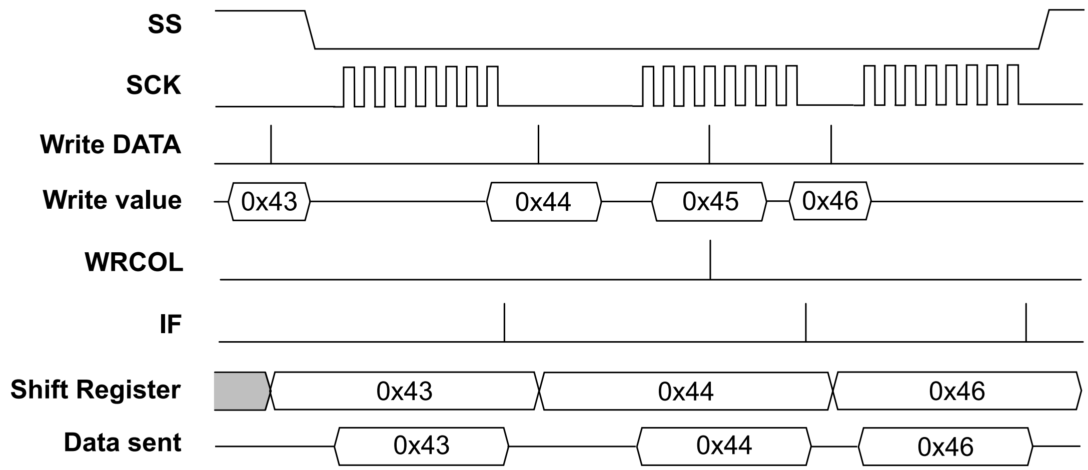
23.5.12 IrDA Control Register

Name: EVCTRL
Offset: 0x0C
Reset: 0x00
Property: -

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

Bit 0 – IREI IrDA Event Input Enable
This bit enables the event source for the IRCOM Receiver. If event input is selected for the IRCOM Receiver, the input from the USART's RX pin is automatically disabled.

Figure 24-2. SPI Timing Diagram in Normal Mode (Buffer Mode Not Enabled)

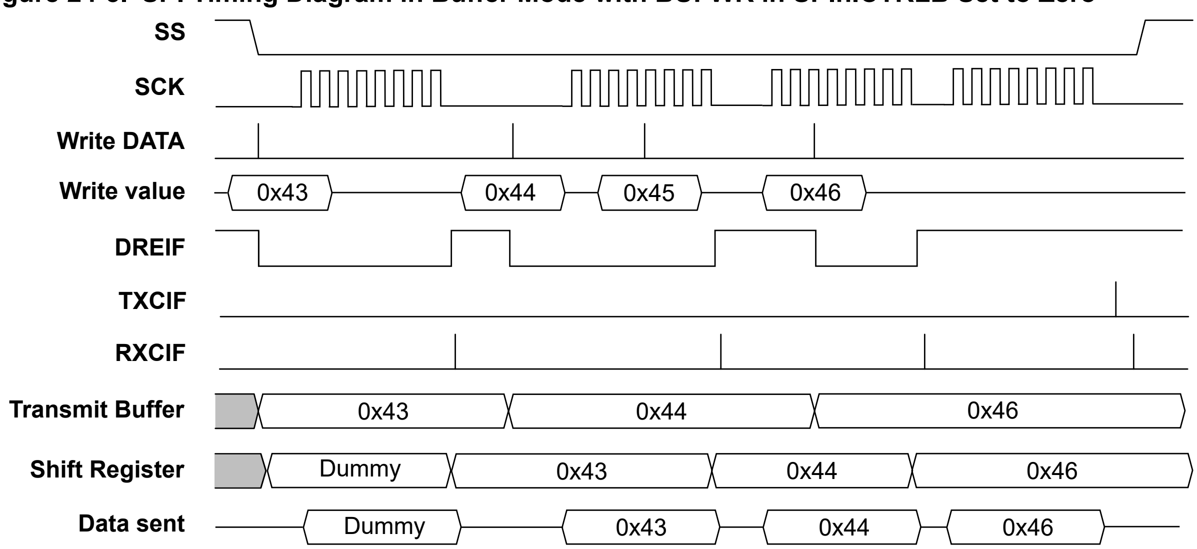


The figure above shows three transfers and one write to the DATA register while the SPI is busy with a transfer. This write will be ignored and the Write Collision Flag (WRCOL in SPIn.INTFLAGS) is set.

Buffer Mode

To avoid data collisions, the SPI peripheral can be configured in buffered mode by writing a '1' to the Buffer Mode Enable bit in the Control B register (BUFEN in SPIn.CTRLB). In this mode, the SPI has additional interrupt flags and extra buffers. The extra buffers are shown in Figure 24-1. There are two different modes for the Buffer mode, selected with the Buffer mode Wait for Receive bit (BUFWR). The two different modes are described below with timing diagrams.

Figure 24-3. SPI Timing Diagram in Buffer Mode with BUFWR in SPIn.CTRLB Set to Zero



All writes to the Data register goes to the Transmit Buffer register. The figure above shows that the value 0x43 is written to the Data register, but it is not immediately transferred to the shift register so the first byte sent will be a dummy byte. The value of the dummy byte is whatever was in the shift register at the time, usually the last received byte. After the first dummy transfer is completed the value 0x43 is transferred to the Shift register. Then 0x44 is written to the Data register and goes to the Transmit Buffer register. A new transfer is started and 0x43 will be sent. The value 0x45 is written to the Data register, but the Transmit Buffer register is not updated since it is already full containing 0x44 and the Data Register Empty Interrupt Flag (DREIF in SPIn.INTFLAGS) is low. The value 0x45 will be lost. After the transfer, the value 0x44 is moved to the Shift register. During the next transfer, 0x46 is written to the Data register and

Case S3: Collision

If the slave is not able to send a high level or NACK, the collision flag is set, and it will disable the data and acknowledge output from the slave logic. The clock hold is released. A Start or repeated Start condition will be accepted.

Case S4: STOP Condition Received

When the Stop condition is received, the slave address/stop flag will be set, indicating that a Stop condition, and not an address match, occurred.

Receiving Data Packets

The slave will know when an address packet with R/$\overline{W}$ direction bit cleared has been successfully received. After acknowledging this, the slave must be ready to receive data. When a data packet is received, the data interrupt flag is set and the slave must indicate ACK or NACK. After indicating a NACK, the slave must expect a Stop or repeated Start condition.

Transmitting Data Packets

The slave will know when an address packet with R/$\overline{W}$ direction bit set has been successfully received. It can then start sending data by writing to the slave data register. When a data packet transmission is completed, the data interrupt flag is set. If the master indicates NACK, the slave must stop transmitting data and expect a Stop or repeated Start condition.

25.3.4.4 Smart Mode

The TWI interface has a Smart mode that simplifies application code and minimizes the user interaction needed to adhere to the I^{2}C protocol. For TWI Master, Smart mode accomplishes this by automatically sending an ACK as soon as data register TWI.MDATA is read. This feature is only active when the ACKACT bit in TWIn.MCTRLA register is set to ACK. If ACKACT is set to NACK, the TWI Master will not generate a NACK bit followed by reading the Data register.

With Smart mode enabled for TWI Slave (SMEN bit in TWIn.SCTRLA), DIF (Data Interrupt Flag) will automatically be cleared if Data register (TWIn.SDATA) is read or written.

25.3.5 Events

Not applicable.

25.3.6 Interrupts

Table 25-2. Available Interrupt Vectors and Sources

Offset	Name	Vector Description	Conditions
0x00	Slave	TWI Slave interrupt	• DIF: Data Interrupt Flag in SSTATUS set • APIF: Address or Stop Interrupt Flag in SSTATUS set
0x02	Master	TWI Master interrupt	• RIF: Read Interrupt Flag in MSTATUS set • WIF: Write Interrupt Flag in MSTATUS set

When an interrupt condition occurs, the corresponding interrupt flag is set in the Master register (TWI.MSTATUS) or Slave Status register (TWI.SSTATUS).
When several interrupt request conditions are supported by an interrupt vector, the interrupt requests are ORed together into one combined interrupt request to the interrupt controller. The user must read the peripheral's INTFLAGS register to determine which of the interrupt conditions are present.

Related Links

CPU Interrupt Controller (CPUINT)

SREG

25.5.5 Master Status

Name: MSTATUS
Offset: 0x05
Reset: 0x00
Property: -

Normal TWI operation dictates that this register is regarded purely as a read-only register. Clearing any of the status flags is done indirectly by accessing the Master Transmits Address (TWIn.MADDR), Master Data register (TWIn.MDATA), or the Command bits (CMD) in the Master Control B register (TWIn.MCTRLB).

Bit	7	6	5	4	3	2	1	0
	RIF	WIF	CLKHOLD	RXACK	ARBLOST	BUSERR	BUSSTATE[1:0]	
Access	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 7 – RIF Read Interrupt Flag
This bit is set to '1' when the master byte read operation is successfully completed (i.e., no arbitration lost or bus error occurred during the operation). The read operation is triggered by software reading DATA or writing to ADDR registers with bit ADDR[0] written to '1'. A slave device must have responded with an ACK to the address and direction byte transmitted by the master for this flag to be set.

Writing a '1' to this bit will clear the RIF. However, normal use of the TWI does not require the flag to be cleared by this method.

Clearing the RIF bit will follow the same software interaction as the CLKHOLD flag.

The RIF flag can generate a master read interrupt (see the description of the RIEN control bit in the TWIn.MCTRLA register).

Bit 6 – WIF Write Interrupt Flag
This bit is set when a master transmit address or byte write is completed, regardless of the occurrence of a bus error or an arbitration lost condition.

Writing a '1' to this bit will clear the WIF. However, normal use of the TWI does not require the flag to be cleared by this method.

Clearing the WIF bit will follow the same software interaction as the CLKHOLD flag.

The WIF flag can generate a master write interrupt (see the description of the WIEN control bit in the TWIn.MCTRLA register).

Bit 5 – CLKHOLD Clock Hold
If read as '1', this bit indicates that the master is currently holding the TWI clock (SCL) low, stretching the TWI clock period.

Writing a '1' to this bit will clear the CLKHOLD flag. However, normal use of the TWI does not require the CLKHOLD flag to be cleared by this method, since the flag is automatically cleared when accessing several other TWI registers. The CLKHOLD flag can be cleared by:

1. Writing a '1' to it.
2. Writing to the TWIn.MADDR register.

3. Writing to the TWIn.MDATA register.
4. Reading the TWIn.DATA register while the ACKACT control bits in TWIn.MCTRLB are set to either send ACK or NACK.
5. Writing a valid command to the TWIn.MCTRLB register.

Bit 4 – RXACK Received Acknowledge
This bit is read-only and contains the most recently received Acknowledge bit from the slave. When read as zero, the most recent acknowledge bit from the slave was ACK. When read as one, the most recent acknowledge bit was NACK.

Bit 3 – ARBLOST Arbitration Lost
If read as '1' this bit indicates that the master has lost arbitration while transmitting a high data or NACK bit, or while issuing a Start or repeated Start condition (S/Sr) on the bus.

Writing a '1' to it will clear the ARBLOST flag. However, normal use of the TWI does not require the flag to be cleared by this method. However, as for the CLKHOLD flag, clearing the ARBLOST flag is not required during normal use of the TWI.

Clearing the ARBLOST bit will follow the same software interaction as the CLKHOLD flag.

Given the condition where the bus ownership is lost to another master, the software must either abort operation or resend the data packet. Either way, the next required software interaction is in both cases to write to the TWIn.MADDR register. A write access to the TWIn.MADDR register will then clear the ARBLOST flag.

Bit 2 – BUSERR Bus Error
The BUSERR flag indicates that an illegal bus condition has occurred. An illegal bus condition is detected if a protocol violating Start (S), repeated Start (Sr), or Stop (P) is detected on the TWI bus lines. A Start condition directly followed by a Stop condition is one example of protocol violation.

Writing a '1' to this bit will clear the BUSERR. However, normal use of the TWI does not require the BUSERR to be cleared by this method.

A robust TWI driver software design will treat the bus error flag similarly to the ARBLOST flag, assuming the bus ownership is lost when the bus error flag is set. As for the ARBLOST flag, the next software operation of writing the TWIn.MADDR register will consequently clear the BUSERR flag. For bus error to be detected, the bus state logic must be enabled and the system frequency must be 4x the SCL frequency.

Bits 1:0 – BUSSTATE[1:0] Bus State
These bits indicate the current TWI bus state as defined in the table below. After a System Reset or re-enabling, the TWI master bus state will be unknown. The change of bus state is dependent on bus activity.

Writing 0x1 to the BUSSTATE bits forces the bus state logic into its Idle state. However, the bus state logic cannot be forced into any other state. When the master is disabled, the bus state is 'unknown'.

Value	Name	Description
0x0	UNKNOWN	Unknown bus state
0x1	IDLE	Bus is idle
0x2	OWNER	This TWI controls the bus
0x3	BUSY	The bus is busy

Bit 3 – COLL Collision
If read as '1', the transmit collision flag indicates that the slave has not been able to transmit a high data or NACK bit. If a slave transmit collision is detected, the slave will commence its operation as normal, except no low values will be shifted out onto the SDA line (i.e., when the COLL flag is set to '1' it disables the data and acknowledge output from the slave logic). The DIF flag will be set to '1' at the end as a result of the internal completion of an unsuccessful transaction. Similarly, when a collision occurs because the slave has not been able to transmit NACK bit, it means the address match already happened and APIF flag is set as a result. APIF/DIF flags can only generate interrupt whose handlers can be used to check for the collision. Writing a '1' to its bit location will clear the COLL flag. However, the flag is automatically cleared if any Start condition (S/Sr) is detected.

This flag is intended for systems where the address resolution protocol (ARP) is employed. However, a detected collision in non-ARP situations indicates that there has been a protocol violation and should be treated as a bus error.

Bit 2 – BUSERR Bus Error
The BUSERR flag indicates that an illegal bus condition has occurred. An illegal bus condition is detected if a protocol violating Start (S), Repeated Start (Sr), or Stop (P) is detected on the TWI bus lines. A Start condition directly followed by a Stop condition is one example of protocol violation. Writing a '1' to its bit location will clear the BUSERR flag. However, normal use of the TWI does not require the BUSERR to be cleared by this method. A robust TWI driver software design will assume that the entire packet of data has been corrupted and restart by waiting for a new Start condition (S). The TWI bus error detector is part of the TWI Master circuitry. For bus errors to be detected, the TWI Master must be enabled (ENABLE bit in TWIn.MCTRLA is '1'), and the system clock frequency must be at least four times the SCL frequency.

Bit 1 – DIR Read/Write Direction
This bit is read-only and indicates the current bus direction state. The DIR bit reflects the direction bit value from the last address packet received from a master TWI device. If this bit is read as '1', a master read operation is in progress. Consequently, a '0' indicates that a master write operation is in progress.

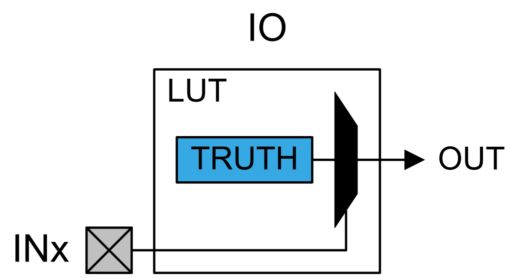
Bit 0 – AP Address or Stop
When the TWI slave address or Stop Interrupt Flag (APIF) is set, this bit determines whether the interrupt is due to address detection or a Stop condition.

Value	Name	Description
0	STOP	A Stop condition generated the interrupt on APIF
1	ADR	Address detection generated the interrupt on APIF

input option by writing to the LUT CONTROL A or B register (CCL.LUTnCTRLB or LUTnCTRLC), the Event System must be configured.

I/O Pin Inputs (I/O)
When selecting the I/O option, the LUT input will be connected to its corresponding I/O pin. Refer to the I/O Multiplexing section for more details about where the LUTnINy is located.

Figure 27-4. I/O Pin Input Selection



Peripherals
The different peripherals on the three input lines of each LUT are selected by writing to the respective LUT n Input y bit fields in the LUT n Control B and C registers:
- INSEL0 in CCL.LUTnCTRLB
- INSEL1 in CCL.LUTnCTRLB
- INSEL2 in CCL.LUTnCTRLC

Related Links
I/O Multiplexing and Considerations
I/O Pin Configuration (PORT)
Clock Controller (CLKCTRL)
Analog Comparator (AC)
16-bit Timer/Counter Type A (TCA)
Universal Synchronous and Asynchronous Receiver and Transmitter (USART)
Serial Peripheral Interface (SPI)
Two-Wire Interface (TWI)
I/O Multiplexing and Considerations

27.3.2.4 Filter

By default, the LUT output is a combinational function of the LUT inputs. This may cause some short glitches when the inputs change the value. These glitches can be removed by clocking through filters if demanded by application needs.

The Filter Selection bits (FILTSEL) in the LUT Control registers (CCL.LUTnCTRLA) define the digital filter options. When a filter is enabled, the output will be delayed by two to five CLK cycles (peripheral clock or alternative clock). One clock cycle after the corresponding LUT is disabled, all internal filter logic is cleared.

A single BREAK character is enough to reset the UPDI, but in some special cases where the BREAK character is sent when the UPDI has not yet entered the error state, a double BREAK character might be needed. A double BREAK is ensured to reset the UPDI from any state. When sending a double BREAK it is required to have at least one Stop bit between the BREAK characters.

No SYNCH character is required before the BREAK because the BREAK is used to reset the UPDI from any state. This means that the UPDI will sample the BREAK based on the last stored baud rate setting, derived from the last received valid SYNCH character. If the communication error was due to an incorrect sampling of the SYNCH character, the baud rate is unknown to the connected debugger. For this reason, the BREAK character should be transmitted at the slowest recommended baud rate setting for the selected UPDI clock according to Table 30-4:

Table 30-4. Recommended BREAK Character Duration

UPDICLKSEL[1:0]	Recommended BREAK Character Duration
0x1 (16 MHz)	6.15 ms
0x2 (8 MHz)	12.30 ms
0x3 (4 MHz) - Default	24.60 ms

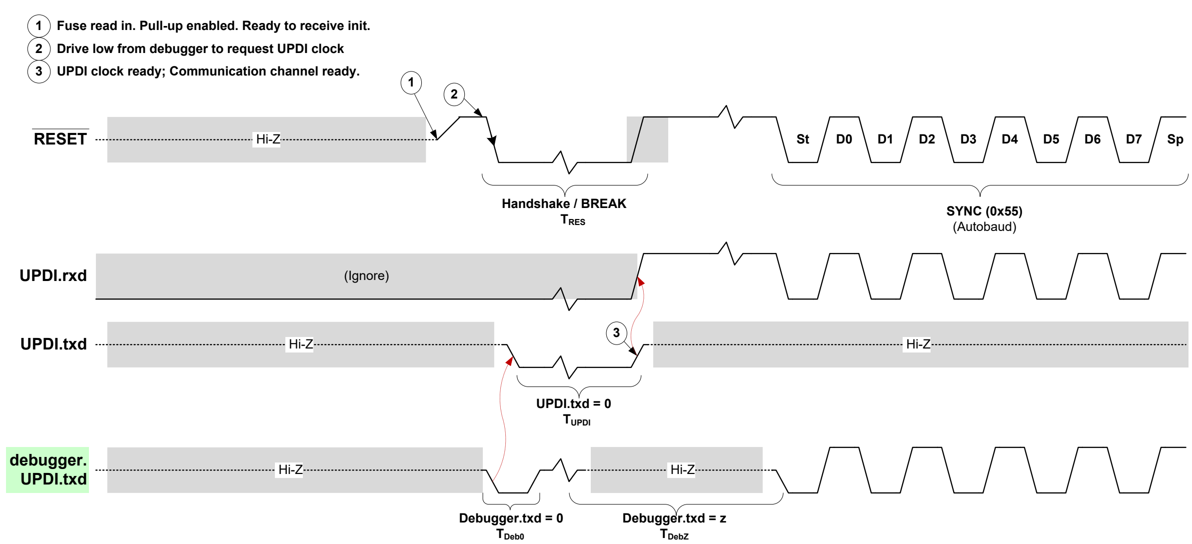
30.3.2 Operation

The UPDI must be enabled before the UART communication can start.

30.3.2.1 UPDI Enable with Fuse Override of $\overline{\text{RESET}}$ Pin

When the $\overline{\text{RESET}}$ Pin Configuration (RSTPINCFG) bits in FUSE.SYSCFG0 are 0x1, the $\overline{\text{RESET}}$ pin will be overridden, and the UPDI will take control of the pin and configure it as input with pull-up. When the pull-up is detected by a connected debugger, the UPDI enable sequence, as depicted below, is started.

Figure 30-5. UPDI Enable Sequence with UPDI PAD Enabled By Fuse



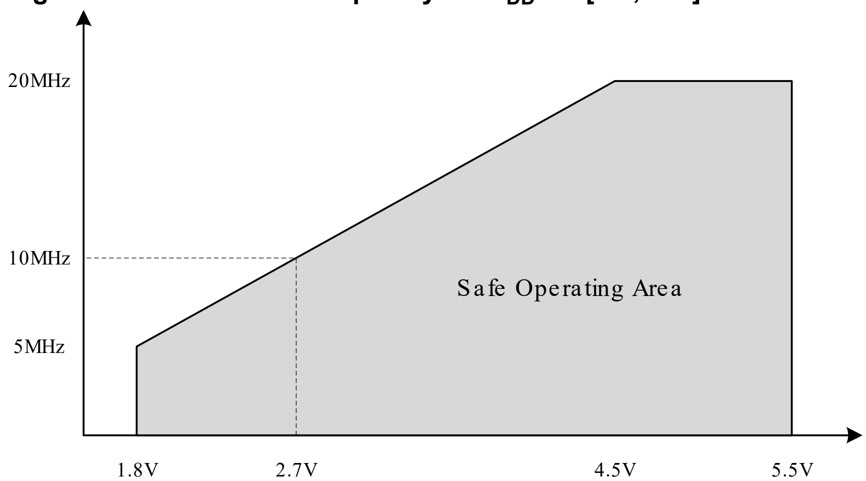
When the pull-up is detected, the debugger initiates the enable sequence by driving the line low for a duration of T_{Deb0} to ensure that the line is released from the debugger before the UPDI enable sequence is done.

The negative edge is detected by the UPDI, which requests the UPDI clock. The UPDI will continue to drive the line low until the clock is stable and ready for the UPDI to use. The duration of this T_{UPDI} will

2. Operation ensured down to BOD triggering level, V_{BOD} with BODLEVEL2.
3. Operation ensured down to BOD triggering level, V_{BOD} with BODLEVEL7.

The maximum CPU clock frequency depends on V_{DD}. As shown in the following figure, the Maximum Frequency vs. V_{DD} is linear between 1.8V < V_{DD} < 2.7V and 2.7V < V_{DD} < 4.5V.

Figure 31-1. Maximum Frequency vs. V_{DD} for [-40, 105]°C



31.4 Power Consumption for ATtiny204/ATtiny404

The values are measured power consumption under the following conditions, except where noted:

- V_{DD} = 3V
- T = 25°C
- OSC20M used as system clock source, except where otherwise specified
- System power consumption measured with peripherals disabled and without I/O drive

Table 31-4. Power Consumption in Active and Idle Mode

Mode	Description	Condition		Typ.	Max.	Unit
Active	Active power consumption	CLK_CPU=20 MHz (OSC20M)	V_{DD}=5V	9.0	-	mA
		CLK_CPU=10 MHz (OSC20M div2)	V_{DD}=5V	4.8	-	mA
			V_{DD}=3V	2.7	-	mA
		CLK_CPU=5 MHz (OSC20M div4)	V_{DD}=5V	2.8	-	mA
			V_{DD}=3V	1.6	-	mA
			V_{DD}=2V	1.0	-	mA
		CLK_CPU=32 KHz (OSCULP32K)	V_{DD}=5V	18	-	μA
			V_{DD}=3V	10	-	μA
			V_{DD}=2V	7	-	μA
Idle	Idle power consumption	CLK_CPU=20 MHz (OSC20M)	V_{DD}=5V	2.8	-	mA
		CLK_CPU=10 MHz (OSC20M div2)	V_{DD}=5V	1.7	-	mA

Figure 32-59. OSC20M Internal Oscillator: Frequency vs. Temperature

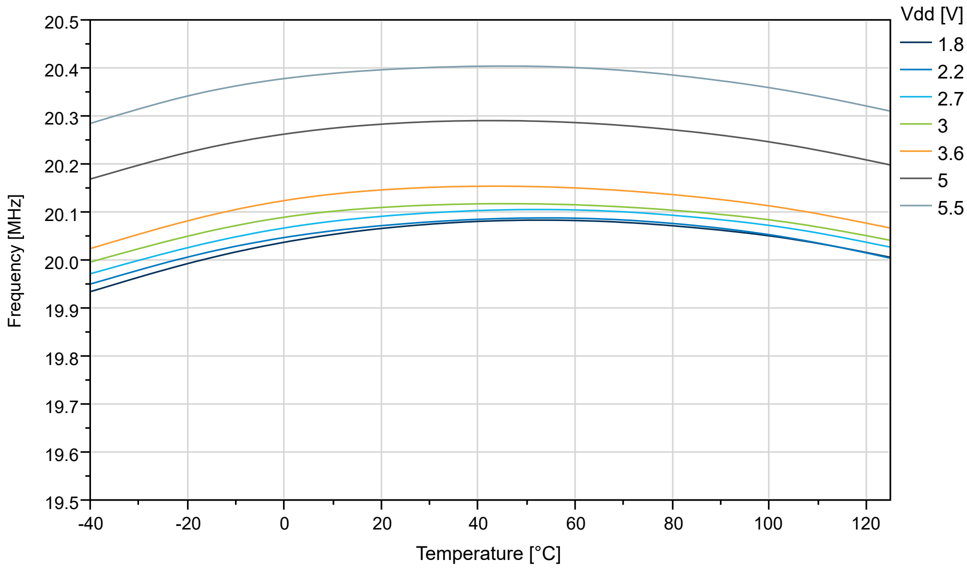


Figure 32-60. OSC20M Internal Oscillator: Frequency vs. V_{DD}