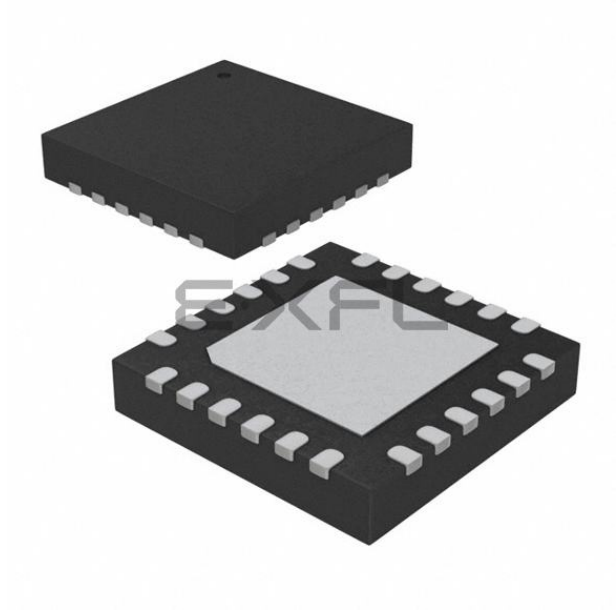


Welcome to E-XFL.COM

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	16MHz
Connectivity	I²C, IrDA, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, WDT
Number of I/O	22
Program Memory Size	8KB (8K x 8)
Program Memory Type	FLASH
EEPROM Size	128 x 8
RAM Size	512 x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	A/D 12x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	24-VFQFN Exposed Pad
Supplier Device Package	24-VQFN (4x4)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/attiny807-mfr

6.10.4.5 System Configuration 1

Name: SYSCFG1
Offset: 0x06
Reset: -
Property: -

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

Bits 2:0 – SUT[2:0] Start-Up Time Setting
These bits select the start-up time between power-on and code execution.

Value	Description
0x0	0 ms
0x1	1 ms
0x2	2 ms
0x3	4 ms
0x4	8 ms
0x5	16 ms
0x6	32 ms
0x7	64 ms

8.5 Functional Description

8.5.1 Program Flow

After Reset, the CPU will execute instructions from the lowest address in the Flash program memory, 0x0000. The Program Counter (PC) addresses the next instruction to be fetched.

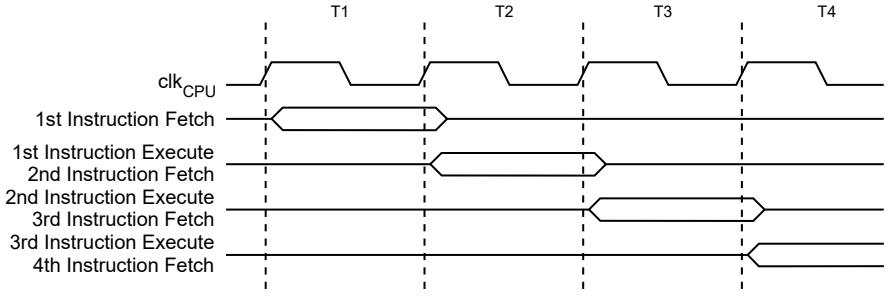
Program flow is supported by conditional and unconditional `JUMP` and `CALL` instructions, capable of addressing the whole address space directly. Most AVR instructions use a 16-bit word format, and a limited number use a 32-bit format.

During interrupts and subroutine calls, the return address PC is stored on the stack as a word pointer. The stack is allocated in the general data SRAM, and consequently, the stack size is only limited by the total SRAM size and the usage of the SRAM. After Reset, the Stack Pointer (SP) points to the highest address in the internal SRAM. The SP is read/write accessible in the I/O memory space, enabling easy implementation of multiple stacks or stack areas. The data SRAM can easily be accessed through the five different addressing modes supported by the AVR CPU.

8.5.2 Instruction Execution Timing

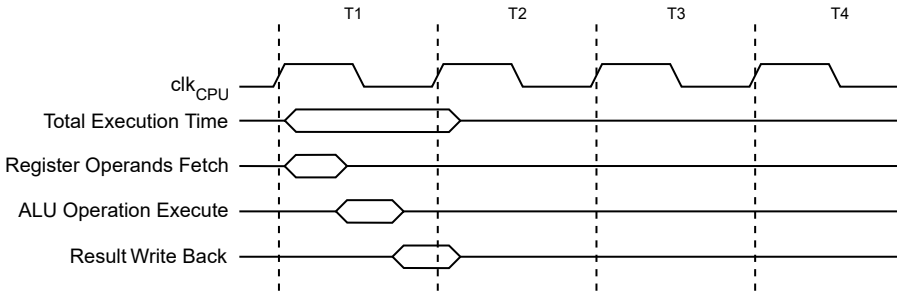
The AVR CPU is clocked by the CPU clock: CLK_CPU. No internal clock division is applied. The figure below shows the parallel instruction fetches and instruction executions enabled by the Harvard architecture and the fast-access register file concept. This is the basic pipelining concept enabling up to 1 MIPS/MHz performance with high efficiency.

Figure 8-2. The Parallel Instruction Fetches and Instruction Executions



The following figure shows the internal timing concept for the register file. In a single clock cycle, an ALU operation using two register operands is executed and the result is stored in the destination register.

Figure 8-3. Single Cycle ALU Operation



8.5.3 Status Register

The Status register (CPU.SREG) contains information about the result of the most recently executed arithmetic or logic instruction. This information can be used for altering program flow in order to perform conditional operations.

Table 12-1. RSTCTRL - Registers Under Configuration Change Protection

Register	Key
RSTCTRL.SWRR	IOREG

Related Links

8.5.7.1 Sequence for Write Operation to Configuration Change Protected I/O Registers

15.2 Register Summary - PORTMUX

Offset	Name	Bit Pos.								
0x00	CTRLA	7:0			LUT1	LUT0		EVOUT2	EVOUT1	EVOUT0
0x01	CTRLB	7:0						SPI0		USART0
0x02	CTRLC	7:0			TCA05	TCA04	TCA03	TCA02	TCA01	TCA00
0x03	CTRLD	7:0								TCB0

15.3 Register Description

respective Reference Select bit field (ADC0REFSEL, DAC0REFSEL) in the Control A register (VREF.CTRLA).

20.7.8 Interrupt Flag Register - Split Mode

Name: INTFLAGS
Offset: 0x0B
Reset: 0x00
Property: -

The individual Status bit can be cleared by writing a ‘1’ to its bit location. This allows each bit to be set without the use of a read-modify-write operation on a single register.

Bit	7	6	5	4	3	2	1	0
		LCMP2	LCMP1	LCMP0			HUNF	LUNF
Access		R/W	R/W	R/W			R/W	R/W
Reset		0	0	0			0	0

Bit 6 – LCMP2 Low byte Compare Channel 0 Interrupt Flag
See LCMP0 flag description.

Bit 5 – LCMP1 Low byte Compare Channel 0 Interrupt Flag
See LCMP0 flag description.

Bit 4 – LCMP0 Low byte Compare Channel 0 Interrupt Flag
The Compare Interrupt flag (LCMPn) is set on a compare match on the corresponding compare channel.

For all modes of operation, the LCMPn flag will be set when a compare match occurs between the Low Byte Count register (TCAn.LCNT) and the corresponding compare register (TCAn.LCMPn). The LCMPn flag will not be cleared automatically and has to be cleared by software. This is done by writing a ‘1’ to its bit location.

Bit 1 – HUNF High byte Underflow Interrupt Flag
This flag is set on a high byte timer BOTTOM (underflow) condition. HUNF is not automatically cleared and needs to be cleared by software. This is done by writing a ‘1’ to its bit location.

Bit 0 – LUNF Low byte Underflow Interrupt Flag
This flag is set on a low byte timer BOTTOM (underflow) condition. LUNF is not automatically cleared and needs to be cleared by software. This is done by writing a ‘1’ to its bit location.

20.7.10 Low Byte Timer Counter Register - Split Mode

Name: LCNT
Offset: 0x20
Reset: 0x00
Property: -

TCAn.LCNT contains the counter value in low byte timer. CPU and UPDI write access has priority over count, clear, or reload of the counter.

Bit	7	6	5	4	3	2	1	0
	LCNT[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 – LCNT[7:0] Counter Value for Low Byte Timer
These bits define the counter value of the low byte timer.

22.11.14 PIT Interrupt Flag

Name: PITINTFLAGS
Offset: 0x13
Reset: 0x00
Property: -

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

Bit 0 – PI Periodic interrupt Flag
This flag is set when a periodic interrupt is issued.

Writing a '1' clears the flag.

23.5.6 Control A

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

Bit	7	6	5	4	3	2	1	0
	RXCIE	TXCIE	DREIE	RXSIE	LBME	ABEIE	RS485[1:0]	
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 7 – RXCIE Receive Complete Interrupt Enable
The bit enables the Receive Complete Interrupt (interrupt vector RXC). The enabled interrupt will be triggered when RXCIF in the USARTn.STATUS register is set.

Bit 6 – TXCIE Transmit Complete Interrupt Enable
This bit enables the Transmit Complete Interrupt (interrupt vector TXC). The enabled interrupt will be triggered when the TXCIF in the USARTn.STATUS register is set.

Bit 5 – DREIE Data Register Empty Interrupt Enable
This bit enables the Data Register Empty Interrupt (interrupt vector DRE). The enabled interrupt will be triggered when the DREIF in the USART.STATUS register is set.

Bit 4 – RXSIE Receiver Start Frame Interrupt Enable
Writing a '1' to this bit enables the Start Frame Detector to generate an interrupt on interrupt vector RXC when a start-of-frame condition is detected.

Bit 3 – LBME Loop-back Mode Enable
Writing this bit to '1' enables an internal connection between the TxD and RxD pin.

Bit 2 – ABEIE Auto-baud Error Interrupt Enable
Writing this bit to '1' enables the auto-baud error interrupt on interrupt vector RXC. The enabled interrupt will trigger for conditions where the ISFIF flag is set.

Bits 1:0 – RS485[1:0] RS-485 Mode
These bits enable the RS-485 and select the operation mode.

Value	Name	Description
0x0	OFF	Disabled.
0x1	EXT	Enables RS-485 mode with control of an external line driver through a dedicated Transmit Enable (TE) pin.
0x2	INT	Enables RS-485 mode with control of the internal USART transmitter.
0x3	-	Reserved.

24.5.2 Control B

Name: CTRLB
Offset: 0x01
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	BUFEN	BUFWR				SSD	MODE[1:0]	
Access	R/W	R/W				R/W	R/W	R/W
Reset	0	0				0	0	0

Bit 7 – BUFEN Buffer Mode Enable
Writing this bit to '1' enables Buffer mode, meaning two buffers in receive direction, one buffer in transmit direction, and separate interrupt flags for both transmit complete and receive complete.

Bit 6 – BUFWR Buffer Mode Wait for Receive
When writing this bit to '0' the first data transferred will be a dummy sample.

Value	Description
0	One SPI transfer must be completed before the data is copied into the Shift register.
1	When writing to the data register when the SPI is enabled and $\overline{SS}$ is high, the first write will go directly to the Shift register.

Bit 2 – SSD Slave Select Disable
When this bit is set and when operating as SPI Master (MASTER=1 in SPIn.CTRLA), $\overline{SS}$ does not disable Master mode.

Value	Description
0	Enable the Slave Select line when operating as SPI Master
1	Disable the Slave Select line when operating as SPI Master

Bits 1:0 – MODE[1:0] Mode
These bits select the Transfer mode. The four combinations of SCK phase and polarity with respect to the serial data are shown in the table below. These bits decide whether the first edge of a clock cycle (leading edge) is rising or falling and whether data setup and sample occur on the leading or trailing edge. When the leading edge is rising, the SCK signal is low when idle, and when the leading edge is falling, the SCK signal is high when idle.

Value	Name	Description
0x0	0	Leading edge: Rising, sample Trailing edge: Falling, setup
0x1	1	Leading edge: Rising, setup Trailing edge: Falling, sample
0x2	2	Leading edge: Falling, sample Trailing edge: Rising, setup
0x3	3	Leading edge: Falling, setup Trailing edge: Rising, sample

24.5.5 Data

Name: DATA
Offset: 0x04
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	DATA[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 – DATA[7:0] SPI Data
The SPIn.DATA register is used for sending and receiving data. Writing to the register initiates the data transmission, and the byte written to the register will be shifted out on the SPI output line.

Reading this register in Buffer mode will read the second receive buffer and the contents of the first receive buffer will be moved to the second receive buffer.

A bus error will behave in the same way as an arbitration lost condition, but the Bus Error Flag (BUSERR in TWIn.MSTATUS) is set in addition to the write interrupt and arbitration lost flags.

Case M2: Address Packet Transmit Complete - Address not Acknowledged by Slave
If no slave device responds to the address, the Master Write Interrupt Flag (WIF in TWIn.MSTATUS) and the Master Received Acknowledge Flag (RXACK in TWIn.MSTATUS) are set. The RXACK flag reflects the physical state of the ACK bit (i.e.< no slave did pull the ACK bit low). The clock hold is active at this point, preventing further activity on the bus.

Case M3: Address Packet Transmit Complete - Direction Bit Cleared
If the master receives an ACK from the slave, the Master Write Interrupt Flag (WIF in TWIn.MSTATUS) is set and the Master Received Acknowledge Flag (RXACK in TWIn.MSTATUS) is cleared. The clock hold is active at this point, preventing further activity on the bus.

Case M4: Address Packet Transmit Complete - Direction Bit Set
If the master receives an ACK from the slave, the master proceeds to receive the next byte of data from the slave. When the first data byte is received, the Master Read Interrupt Flag (RIF in TWIn.MSTATUS) is set and the Master Received Acknowledge Flag (RXACK in TWIn.MSTATUS) is cleared. The clock hold is active at this point, preventing further activity on the bus.

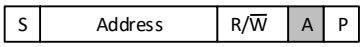
25.3.4.2.3 Transmitting Data Packets
The slave will know when an address packet with R/$\overline{W}$ direction bit set has been successfully received. It can then start sending data by writing to the slave data register. When a data packet transmission is completed, the data interrupt flag is set. If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

25.3.4.2.4 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.

25.3.4.2.5 Quick Command Mode
With Quick Command enabled (QCEN in TWIn.MCTRLA), the R/W# bit of the slave address denotes the command. This is a SMBus specific command where the R/$\overline{W}$ bit may be used to simply turn a device function ON or OFF, or enable/disable a low-power Standby mode. There is no data sent or received.

After the master receives an acknowledge from the slave, either RIF or WIF flag in TWIn.MSTATUS will be set depending on the polarity of R/$\overline{W}$. When either RIF or WIF flag is set after issuing a Quick Command, the TWI will accept a stop command through writing the CMD bits in TWIn.MCTRLB.

Figure 25-15. Quick Command Frame Format



25.3.4.3 TWI Slave Operation
The TWI slave is byte-oriented with optional interrupts after each byte. There are separate slave data and address/stop interrupt flags. Interrupt flags can also be used for polled operation. There are dedicated status flags for indicating ACK/NACK received, clock hold, collision, bus error, and read/write direction.

When an interrupt flag is set, the SCL line is forced low. This will give the slave time to respond or handle data, and will in most cases require software interaction. Figure 25-16 shows the TWI slave operation. The diamond-shaped symbols (SW) indicate where software interaction is required.

26.5.2 Control B

Name: CTRLB
Offset: 0x01
Reset: 0x00
Property: -

The CTRLB register contains the mode and source settings for the CRC. It is not writable when the CRC is busy or when an NMI has been triggered.

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

Bits 1:0 – SRC[1:0] CRC Source
The SRC bit field selects which section of the Flash the CRC module should check. To set up section sizes, refer to the fuse description.
The CRC can be enabled during internal Reset initialization to verify Flash sections before letting the CPU start (see fuse description). If the CRC is enabled during internal Reset initialization, the SRC bit field will read out as FLASH, BOOTAPP, or BOOT when normal code execution starts (depending on the configuration).

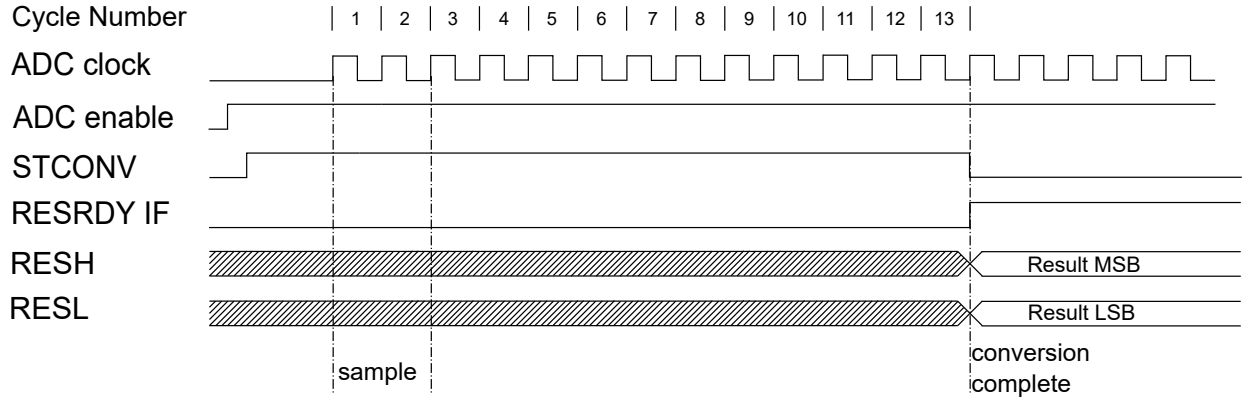
Value	Name	Description
0x0	FLASH	The CRC is performed on the entire Flash (boot, application code, and application data sections).
0x1	BOOTAPP	The CRC is performed on the boot and application code sections of Flash.
0x2	BOOT	The CRC is performed on the boot section of Flash.
0x3	SIGROW	The CRC is performed on the Flash signature row (AUX).

Related Links

6.10 Configuration and User Fuses (FUSE)
12.3.2.2 Reset Time

be cleared when the result is read from the Result registers, or by writing a '1' to the RESRDY bit in ADC.INTFLAG.

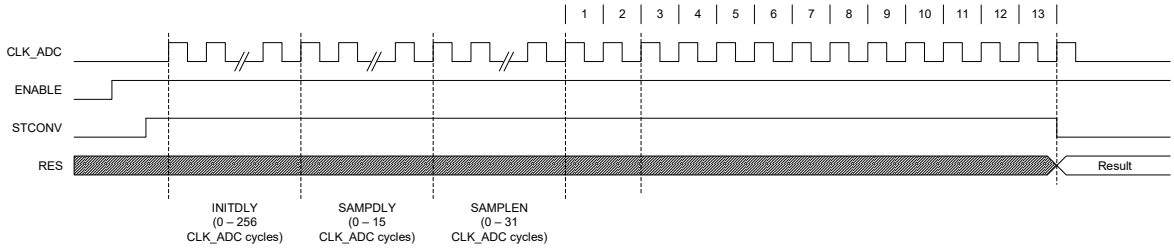
Figure 29-8. ADC Timing Diagram - Single Conversion



Both sampling time and sampling length can be adjusted using the Sample Delay bit field in Control D (ADC.CTRLD) and sampling the Sample Length bit field in the Sample Control register (ADC.SAMPCTRL). Both of these control the ADC sampling time in a number of CLK_ADC cycles. This allows sampling high-impedance sources without relaxing conversion speed. See the register description for further information. Total sampling time is given by:

$$\text{SampleTime} = \frac{(2 + \text{SAMPDLY} + \text{SAMPLEN})}{f_{\text{CLK\_ADC}}}$$

Figure 29-9. ADC Timing Diagram - Single Conversion With Delays



29.3.2.4 Changing Channel or Reference Selection

The MUXPOS bits in the ADCn.MUXPOS register and the REFSEL bits in the ADCn.CTRLC register are buffered through a temporary register to which the CPU has random access. This ensures that the channel and reference selections only take place at a safe point during the conversion. The channel and reference selections are continuously updated until a conversion is started.

Once the conversion starts, the channel and reference selections are locked to ensure sufficient sampling time for the ADC. Continuous updating resumes in the last CLK_ADC clock cycle before the conversion completes (RESRDY in ADCn.INTFLAGS is set). The conversion starts on the following rising CLK_ADC clock edge after the STCONV bit is written to '1'.

29.3.2.4.1 ADC Input Channels

When changing channel selection, the user should observe the following guideline to ensure that the correct channel is selected:

The channel should be selected before starting the conversion. The channel selection may be changed one ADC clock cycle after writing '1' to the STCONV bit.

The ADC requires a settling time after switching the input channel - refer to the Electrical Characteristics section for details.

6. In ADCn.CTRLC select $SAMPCAP = 1$
7. Acquire the temperature sensor output voltage by starting a conversion.
8. Process the measurement result as described below.

The measured voltage has a linear relationship to the temperature. Due to process variations, the temperature sensor output voltage varies between individual devices at the same temperature. The individual compensation factors are determined during the production test and saved in the Signature Row:

- SIGROW.TEMPSENSE0 is a gain/slope correction
- SIGROW.TEMPSENSE1 is an offset correction

In order to achieve accurate results, the result of the temperature sensor measurement must be processed in the application software using factory calibration values. The temperature (in Kelvin) is calculated by this rule:

```
Temp = (((RESH << 8) | RESL) - TEMPSENSE1) * TEMPSENSE0) >> 8
```

RESH and RESL are the high and low bytes of the Result register (ADCn.RES), and TEMPSENSEn are the respective values from the Signature row.

It is recommended to follow these steps in user code:

```
int8_t sigrow_offset = SIGROW.TEMPSENSE1;  // Read signed value from signature row
uint8_t sigrow_gain = SIGROW.TEMPSENSE0;    // Read unsigned value from signature row
uint16_t adc_reading = 0;   // ADC conversion result with 1.1 V internal reference

uint32_t temp = adc_reading - sigrow_offset;
temp *= sigrow_gain;  // Result might overflow 16 bit variable (10bit+8bit)
temp += 0x80;               // Add 1/2 to get correct rounding on division below
temp >>= 8;                 // Divide result to get Kelvin
uint16_t temperature_in_K = temp;
```

Related Links

6.10.2.3 TEMPSENSEn

29.3.2.7 Window Comparator Mode

The ADC can raise the WCOMP flag in the Interrupt and Flag register (ADCn.INTFLAG) and request an interrupt (WCOMP) when the result of a conversion is above and/or below certain thresholds. The available modes are:

- The result is under a threshold
- The result is over a threshold
- The result is inside a window (above a lower threshold, but below the upper one)
- The result is outside a window (either under the lower or above the upper threshold)

The thresholds are defined by writing to the Window Comparator Threshold registers (ADCn.WINLT and ADCn.WINHT). Writing to the Window Comparator mode bit field (WINCM) in the Control E register (ADCn.CTRLE) selects the conditions when the flag is raised and/or the interrupt is requested.

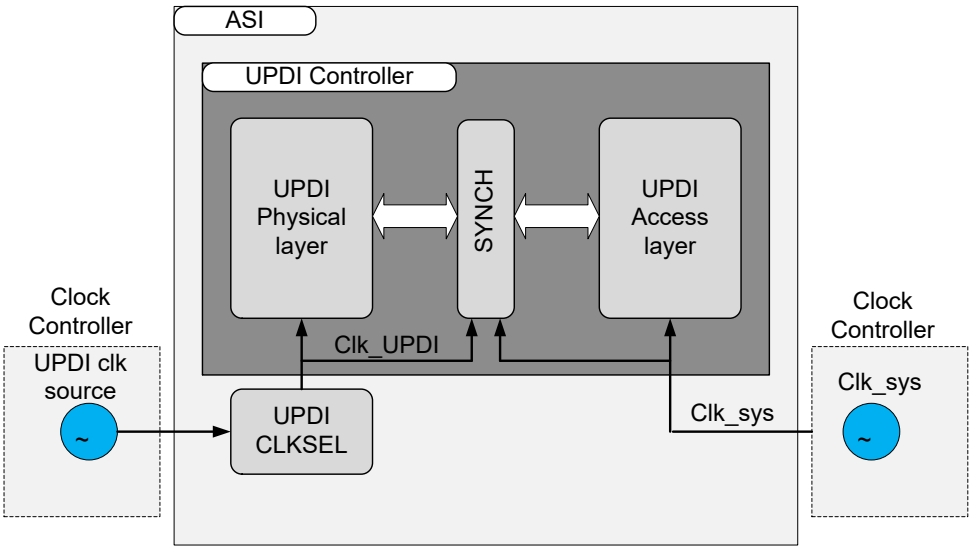
Assuming the ADC is already configured to run, follow these steps to use the Window Comparator mode:

1. Choose which Window Comparator to use (see the WINCM description in ADCn.CTRLE), and set the required threshold(s) by writing to ADCn.WINLT and/or ADCn.WINHT.
2. Optional: enable the interrupt request by writing a '1' to the Window Comparator Interrupt Enable bit (WCOMP) in the Interrupt Control register (ADCn.INTCTRL).

30.2.2.1 Clocks

The UPDI Physical (UPDI PHY) layer and UPDI Access (UPDI ACC) layer can operate on different clock domains. The UPDI PHY layer clock is derived from an internal oscillator, and the UPDI ACC layer clock is the same as the system clock. There is a synchronization boundary between the UPDI PHY layer and the UPDI ACC layer, which ensures correct operation between the clock domains. The UPDI clock output frequency is selected through the ASI, and the default UPDI clock start-up frequency is 4 MHz after enabling the UPDI. The UPDI clock frequency is changed by writing the UPDICLKSEL bits in the ASI_CTRLA register.

Figure 30-2. UPDI Clock Domains



Related Links

10. CLKCTRL - Clock Controller

30.2.2.2 I/O Lines and Connections

To operate the UPDI, the $\overline{\text{RESET}}$ pin must be set to UPDI mode. This is not done through the port I/O pin configuration as regular I/O pins, but through setting the $\overline{\text{RESET}}$ Pin Configuration (RSTPINCFG) bits in FUSE.SYSCFG0, as described in 30.3.2.1 UPDI Enable with Fuse Override of RESET Pin, or by following the UPDI 12V enable sequence from 30.3.2.2 UPDI Enable with 12V Override of RESET Pin. Pull enable, input enable, and output enable settings are automatically controlled by the UPDI when active.

30.2.2.3 Events

The events of this peripheral are connected to the Event System.

Related Links

14. EVSYS - Event System

30.2.2.4 Power Management

The UPDI physical layer continues to operate in any Sleep mode and is always accessible for a connected debugger, but read/write access to the system bus is restricted in Sleep modes where the CPU clock is switched OFF. The UPDI can be enabled at any time, independent of the system Sleep state. See 30.3.9 Sleep Mode Operation for details on UPDI operation during Sleep modes.

vary, depending on the status of the oscillator when the UPDI is enabled. After this duration, the data line will be released by the UPDI and pulled high.

When the debugger detects that the line is high, the initial SYNCH character (0x55) must be sent to properly enable the UPDI for communication. If the Start bit of the SYNCH character is not sent well within maximum T_{DebZ}, the UPDI will disable itself, and the enable sequence must be repeated. This time is based on counted cycles on the 4 MHz UPDI clock, which is the default when enabling the UPDI. The disable is performed to avoid the UPDI being enabled unintentionally.

After successful SYNCH character transmission, the first instruction frame can be transmitted.

Related Links

31.17 UPDI Timing

31.17 UPDI Timing

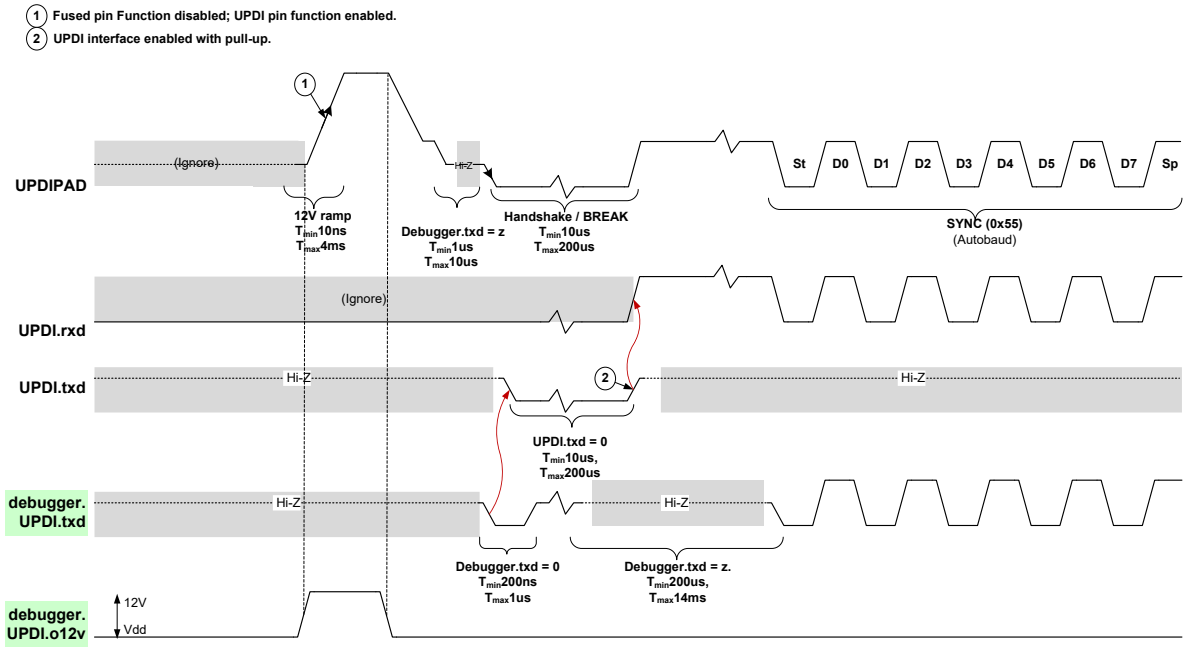
30.3.2.2 UPDI Enable with 12V Override of $\overline{\text{RESET}}$ Pin

GPIO or Reset functionality on the $\overline{\text{RESET}}$ pin can be overridden by the UPDI by using 12V programming. By applying a 12V pulse to the $\overline{\text{RESET}}$ pin, the pin functionality is switched to UPDI, independent of RSTPINCFG in FUSE.SYSCFG0. It is recommended to always reset the device before starting the 12V enable sequence.

During power-up, the Power-on Reset (POR) must be released before the 12V pulse can be applied. The duration of the pulse is recommended in the range from 100 µs to 1 ms, before tri-stating. When applying the rising edge of the 12V pulse, the UPDI will be reset. After tri-stating, the UPDI will remain in Reset until the $\overline{\text{RESET}}$ pin is driven low by the debugger. This will release the UPDI Reset and initiate the same enable sequence as explained in 30.3.2.1 UPDI Enable with Fuse Override of RESET Pin.

The following figure shows the 12V enable sequence.

Figure 30-6. UPDI Enable Sequence by 12V Programming



When enabled by 12V, only a POR will disable the UPDI configuration on the $\overline{\text{RESET}}$ pin, and restore the default setting. If issuing a UPDI Disable command through the UPDIDIS bit in UPDI.CTRLB, the UPDI will be reset and the clock request will be canceled, but the $\overline{\text{RESET}}$ pin will remain in UPDI configuration.

31. Electrical Characteristics

31.1 Disclaimer

All typical values are measured at T = 25°C and V_{DD} = 3V unless otherwise specified. All minimum and maximum values are valid across operating temperature and voltage unless otherwise specified.

31.2 Absolute Maximum Ratings

Stresses beyond those listed in this section may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

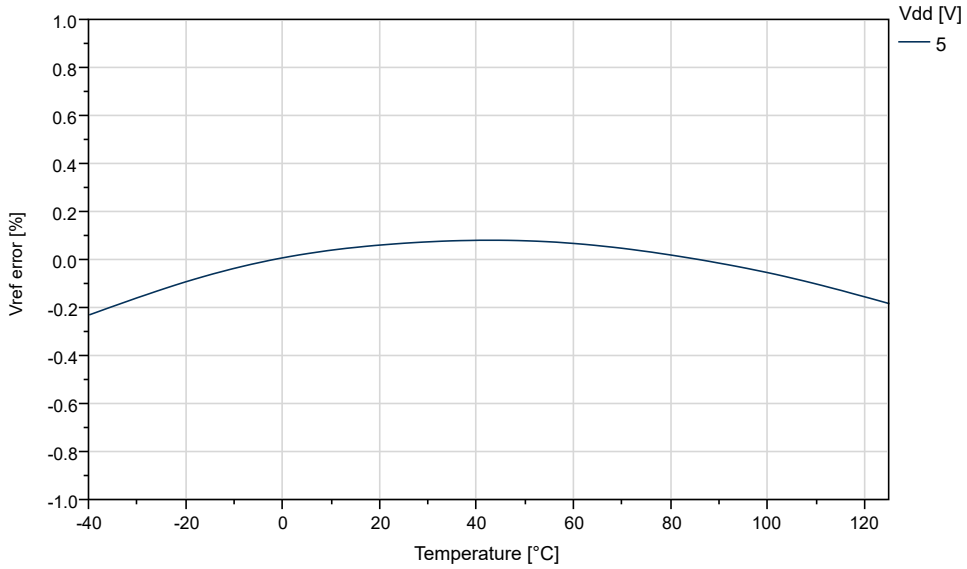
Table 31-1. Absolute Maximum Ratings

Symbol	Description	Conditions	Min.	Max.	Unit
V_{DD}	Power Supply Voltage		-0.5	6	V
I_{VDD}	Current into a V_{DD} pin	T=[-40, 85]°C	-	200	mA
		T=[85, 125]°C	-	100	mA
I_{GND}	Current out of a GND pin	T=[-40, 85]°C	-	200	mA
		T=[85, 125]°C	-	100	mA
V_{RST}	RESET pin voltage with respect to GND		-0.5	13	V
V_{PIN}	Pin voltage with respect to GND		-0.5	V_{DD}+0.5	V
I_{PIN}	I/O pin sink/source current		-40	40	mA
I_{c1}(1)	I/O pin injection current except RESET pin	V_{pin}<GND-0.6V or 5.5V<V_{pin}≤6.1V 4.9V<V_{DD}≤5.5V	-1	1	mA
I_{c2}(1)	I/O pin injection current except RESET pin	V_{pin}<GND-0.6V or V_{pin}≤5.5V V_{DD}≤4.9V	-15	15	mA
I_{ctot}	Sum of I/O pin injection current except RESET pin		-45	45	mA
$T_{storage}$	Storage temperature		-65	150	°C

Note:

1. – If V_{PIN} is lower than GND-0.6V, then a current limiting resistor is required. The negative DC injection current limiting resistor is calculated as R = (GND-0.6V – V_{pin})/I_{Cn}.
   – If V_{PIN} is greater than V_{DD}+0.6V, then a current limiting resistor is required. The positive DC injection current limiting resistor is calculated as R = (V_{pin}-(V_{DD}+0.6))/I_{Cn}.

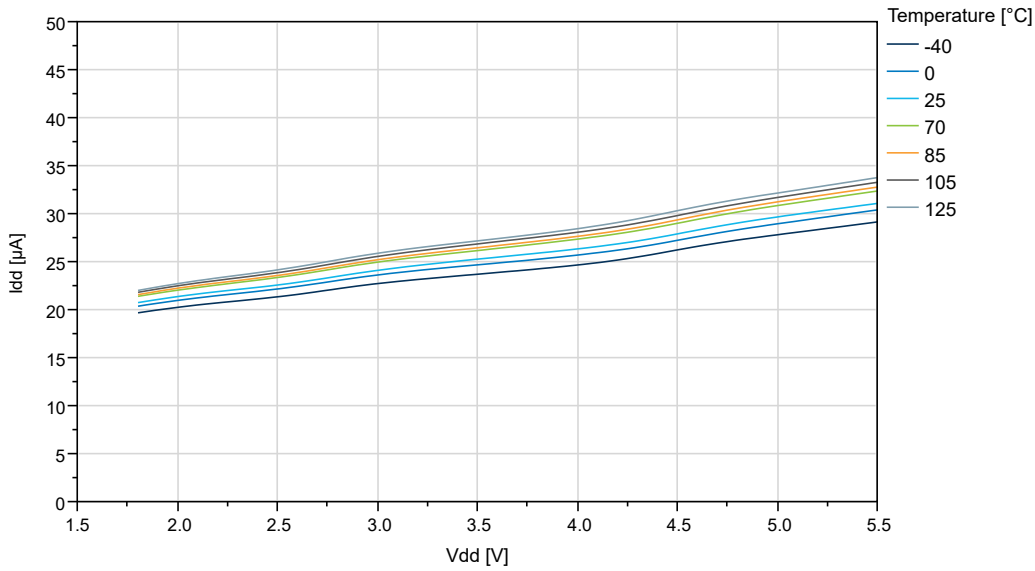
Figure 32-33. Internal 4.3V Reference vs. Temperature



32.4 BOD Characteristics

BOD Current vs. V_{DD}

Figure 32-34. BOD Current vs. V_{DD} (Continuous Mode Enabled)



Mnemonic	Operands	Description		Op		Flags	#Clocks
			PC(21:16)	←	0		
CALL	k	Call Subroutine	PC	←	k	None	3 / 4
RET		Subroutine Return	PC	←	STACK	None	4 / 5
RETI		Interrupt Return	PC	←	STACK	I	4 / 5
CPSE	Rd,Rr	Compare, skip if Equal	if (Rd = Rr) PC	←	PC + 2 or 3	None	1 / 2 / 3
CP	Rd,Rr	Compare	Rd - Rr			Z,C,N,V,S,H	1
CPC	Rd,Rr	Compare with Carry	Rd - Rr - C			Z,C,N,V,S,H	1
CPI	Rd,K	Compare with Immediate	Rd - K			Z,C,N,V,S,H	1
SBRC	Rr, b	Skip if Bit in Register Cleared	if (Rr(b) = 0) PC	←	PC + 2 or 3	None	1 / 2 / 3
SBRS	Rr, b	Skip if Bit in Register Set	if (Rr(b) = 1) PC	←	PC + 2 or 3	None	1 / 2 / 3
SBIC	A, b	Skip if Bit in I/O Register Cleared	if (I/O(A,b) = 0) PC	←	PC + 2 or 3	None	1 / 2 / 3
SBIS	A, b	Skip if Bit in I/O Register Set	If (I/O(A,b) =1) PC	←	PC + 2 or 3	None	1 / 2 / 3
BRBS	s, k	Branch if Status Flag Set	if (SREG(s) = 1) then PC	←	PC + k + 1	None	1 / 2
BRBC	s, k	Branch if Status Flag Cleared	if (SREG(s) = 0) then PC	←	PC + k + 1	None	1 / 2
BREQ	k	Branch if Equal	if (Z = 1) then PC	←	PC + k + 1	None	1 / 2
BRNE	k	Branch if Not Equal	if (Z = 0) then PC	←	PC + k + 1	None	1 / 2
BRCS	k	Branch if Carry Set	if (C = 1) then PC	←	PC + k + 1	None	1 / 2
BRCC	k	Branch if Carry Cleared	if (C = 0) then PC	←	PC + k + 1	None	1 / 2
BRSH	k	Branch if Same or Higher	if (C = 0) then PC	←	PC + k + 1	None	1 / 2
BRLO	k	Branch if Lower	if (C = 1) then PC	←	PC + k + 1	None	1 / 2
BRMI	k	Branch if Minus	if (N = 1) then PC	←	PC + k + 1	None	1 / 2
BRPL	k	Branch if Plus	if (N = 0) then PC	←	PC + k + 1	None	1 / 2
BRGE	k	Branch if Greater or Equal, Signed	if (N ⊕ V= 0) then PC	←	PC + k + 1	None	1 / 2
BRLT	k	Branch if Less Than, Signed	if (N ⊕ V= 1) then PC	←	PC + k + 1	None	1 / 2
BRHS	k	Branch if Half Carry Flag Set	if (H = 1) then PC	←	PC + k + 1	None	1 / 2
BRHC	k	Branch if Half Carry Flag Cleared	if (H = 0) then PC	←	PC + k + 1	None	1 / 2
BRTS	k	Branch if T Flag Set	if (T = 1) then PC	←	PC + k + 1	None	1 / 2
BRTC	k	Branch if T Flag Cleared	if (T = 0) then PC	←	PC + k + 1	None	1 / 2
BRVS	k	Branch if Overflow Flag is Set	if (V = 1) then PC	←	PC + k + 1	None	1 / 2