Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

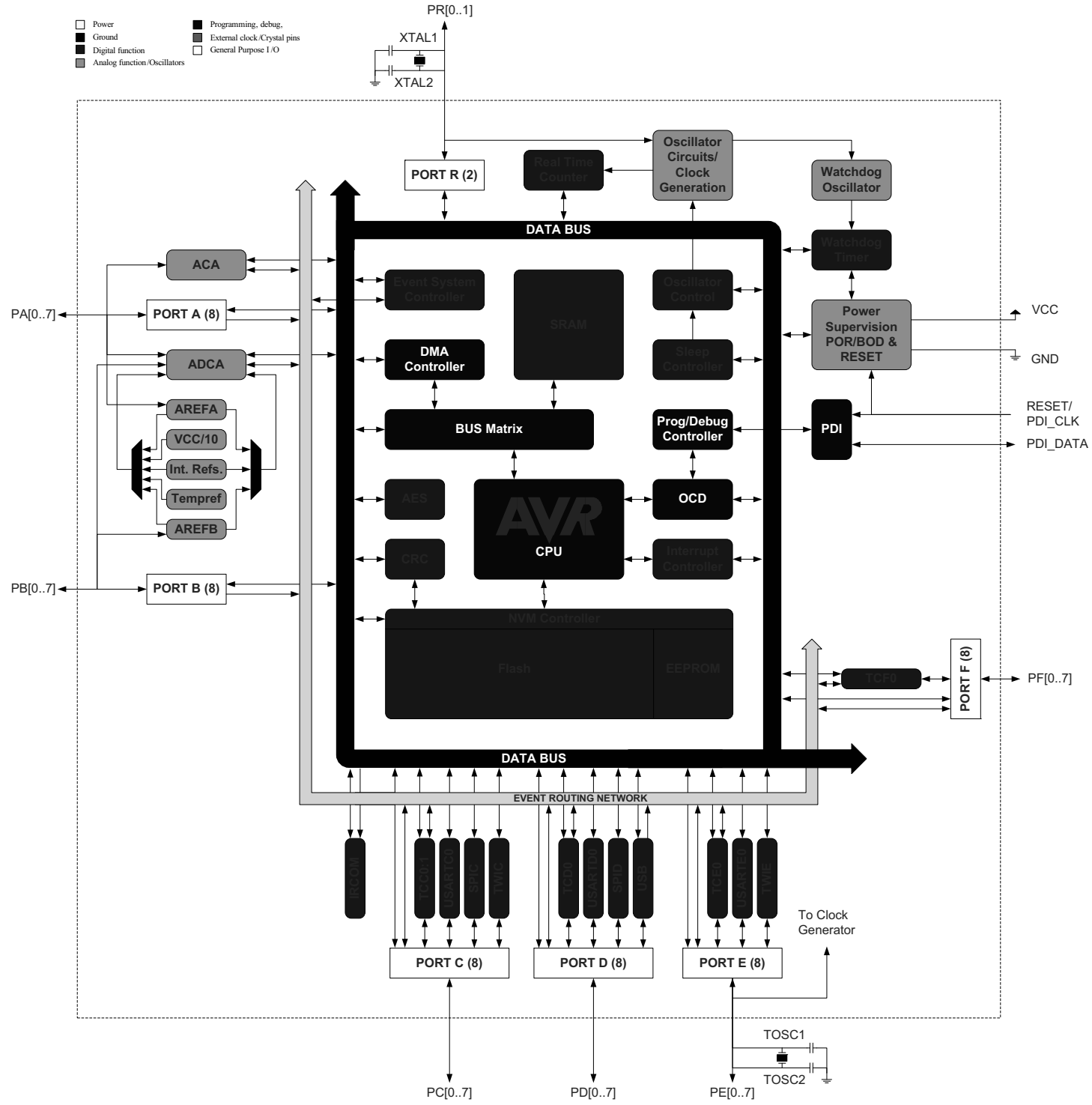
"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" 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	8/16-Bit
Speed	32MHz
Connectivity	I²C, IrDA, SPI, UART/USART, USB
Peripherals	Brown-out Detect/Reset, DMA, POR, PWM, WDT
Number of I/O	50
Program Memory Size	384KB (384K x 8)
Program Memory Type	FLASH
EEPROM Size	4K x 8
RAM Size	32K x 8
Voltage - Supply (Vcc/Vdd)	1.6V ~ 3.6V
Data Converters	A/D 16x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	64-TQFP
Supplier Device Package	64-TQFP (14x14)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/atxmega384c3-aur

3.1 Block Diagram

Figure 3-1. XMEGA C3 Block Diagram



5. Capacitive Touch Sensing

The Atmel QTouch library provides a simple to use solution to realize touch sensitive interfaces on most Atmel AVR microcontrollers. The patented charge-transfer signal acquisition offers robust sensing and includes fully debounced reporting of touch keys and includes Adjacent Key Suppression® (AKS®) technology for unambiguous detection of key events. The QTouch library includes support for the QTouch and QMatrix acquisition methods.

Touch sensing can be added to any application by linking the appropriate Atmel QTouch library for the AVR microcontroller. This is done by using a simple set of APIs to define the touch channels and sensors, and then calling the touch sensing API's to retrieve the channel information and determine the touch sensor states.

The QTouch library is FREE and downloadable from the Atmel website at the following location: www.atmel.com/qtouchlibrary. For implementation details and other information, refer to the QTouch library user guide - also available for download from the Atmel website.

7.12 Flash and EEPROM Page Size

The flash program memory and EEPROM data memory are organized in pages. The pages are word accessible for the flash and byte accessible for the EEPROM.

Table 7-2 on page 15 shows the Flash Program Memory organization and Program Counter (PC) size. Flash write and erase operations are performed on one page at a time, while reading the Flash is done one byte at a time. For Flash access the Z-pointer (Z[m:n]) is used for addressing. The most significant bits in the address (FPAGE) give the page number and the least significant address bits (FWORD) give the word in the page.

Table 7-2. Number of Words and Pages in the Flash

Devices	PC size	Flash size	Page size	FWORD	FPAGE	Application		Boot	
	bits	bytes	words			Size	No. of pages	Size	No. of pages
ATxmega384C3	18	384K + 8K	256	Z[8:1]	Z[19:9]	384K	768	8K	16

Table 7-3 shows EEPROM memory organization. EEEPROM write and erase operations can be performed one page or one byte at a time, while reading the EEPROM is done one byte at a time. For EEPROM access the NVM address register (ADDR[m:n]) is used for addressing. The most significant bits in the address (E2PAGE) give the page number and the least significant address bits (E2BYTE) give the byte in the page.

Table 7-3. Number of Bytes and Pages in the EEPROM

Devices	EEPROM	Page size	E2BYTE	E2PAGE	No. of pages
	Size	bytes			
ATxmega384C3	4K	32	ADDR[4:0]	ADDR[11:5]	128

8. DMAC – Direct Memory Access Controller

8.1 Features

- **Allows high speed data transfers with minimal CPU intervention**
  - **from data memory to data memory**
  - **from data memory to peripheral**
  - **from peripheral to data memory**
  - **from peripheral to peripheral**
- **Two DMA channels with separate**
  - **transfer triggers**
  - **interrupt vectors**
  - **addressing modes**
- **Programmable channel priority**
- **From one byte to 16MB of data in a single transaction**
  - **Up to 64KB block transfers with repeat**
  - **1, 2, 4, or 8 byte burst transfers**
- **Multiple addressing modes**
  - **Static**
  - **Incremental**
  - **Decremental**
- **Optional reload of source and destination addresses at the end of each**
  - **Burst**
  - **Block**
  - **Transaction**
- **Optional interrupt on end of transaction**
- **Optional connection to CRC generator for CRC on DMA data**

8.2 Overview

The two-channel direct memory access (DMA) controller can transfer data between memories and peripherals, and thus off-load these tasks from the CPU. It enables high data transfer rates with minimum CPU intervention, and frees up CPU time. The four DMA channels enable up to four independent and parallel transfers.

The DMA controller can move data between SRAM and peripherals, between SRAM locations and directly between peripheral registers. With access to all peripherals, the DMA controller can handle automatic transfer of data to/from communication modules. The DMA controller can also read from memory mapped EEPROM.

Data transfers are done in continuous bursts of 1, 2, 4, or 8 bytes. They build block transfers of configurable size from 1 byte to 64KB. A repeat counter can be used to repeat each block transfer for single transactions up to 16MB. Source and destination addressing can be static, incremental or decremental. Automatic reload of source and/or destination addresses can be done after each burst or block transfer, or when a transaction is complete. Application software, peripherals, and events can trigger DMA transfers.

The two DMA channels have individual configuration and control settings. This include source, destination, transfer triggers, and transaction sizes. They have individual interrupt settings. Interrupt requests can be generated when a transaction is complete or when the DMA controller detects an error on a DMA channel.

To allow for continuous transfers, two channels can be interlinked so that the second takes over the transfer when the first is finished, and vice versa.

14. Interrupts and Programmable Multilevel Interrupt Controller

14.1 Features

- **Short and predictable interrupt response time**
- **Separate interrupt configuration and vector address for each interrupt**
- **Programmable multilevel interrupt controller**
  - **Interrupt prioritizing according to level and vector address**
  - **Three selectable interrupt levels for all interrupts: low, medium and high**
  - **Selectable, round-robin priority scheme within low-level interrupts**
  - **Non-maskable interrupts for critical functions**
- **Interrupt vectors optionally placed in the application section or the boot loader section**

14.2 Overview

Interrupts signal a change of state in peripherals, and this can be used to alter program execution. Peripherals can have one or more interrupts, and all are individually enabled and configured. When an interrupt is enabled and configured, it will generate an interrupt request when the interrupt condition is present. The programmable multilevel interrupt controller (PMIC) controls the handling and prioritizing of interrupt requests. When an interrupt request is acknowledged by the PMIC, the program counter is set to point to the interrupt vector, and the interrupt handler can be executed.

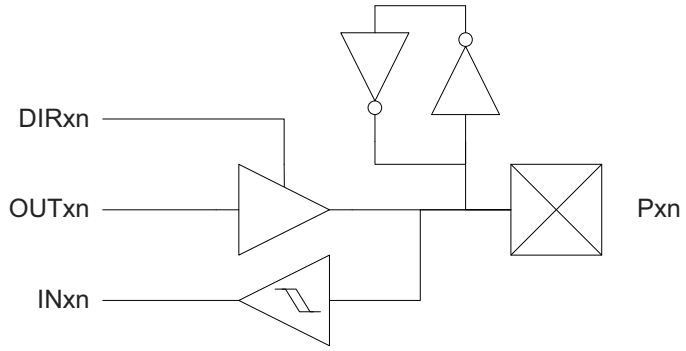
All peripherals can select between three different priority levels for their interrupts: low, medium, and high. Interrupts are prioritized according to their level and their interrupt vector address. Medium-level interrupts will interrupt low-level interrupt handlers. High-level interrupts will interrupt both medium- and low-level interrupt handlers. Within each level, the interrupt priority is decided from the interrupt vector address, where the lowest interrupt vector address has the highest interrupt priority. Low-level interrupts have an optional round-robin scheduling scheme to ensure that all interrupts are serviced within a certain amount of time.

Non-maskable interrupts (NMI) are also supported, and can be used for system critical functions.

14.3 Interrupt Vectors

The interrupt vector is the sum of the peripheral's base interrupt address and the offset address for specific interrupts in each peripheral. The base addresses for the Atmel AVR XMEGA C3 devices are shown in Table 14-1 on page 27. Offset addresses for each interrupt available in the peripheral are described for each peripheral in the XMEGA C manual. For peripherals or modules that have only one interrupt, the interrupt vector is shown in Table 14-1 on page 27. The program address is the word address.

Figure 15-4. I/O Configuration - Totem-pole with Bus-keeper



15.3.5 Others

Figure 15-5. Output Configuration - Wired-OR with Optional Pull-down

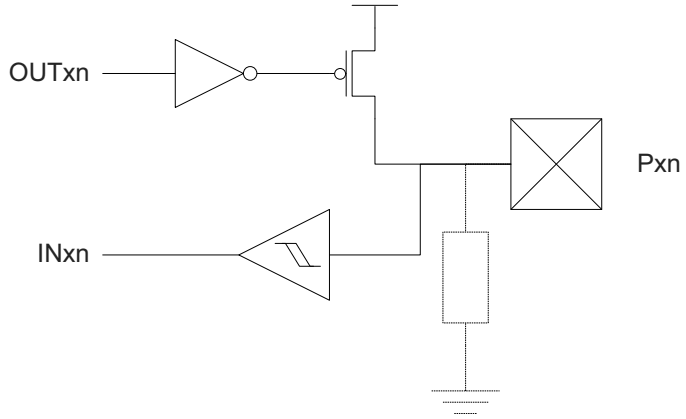
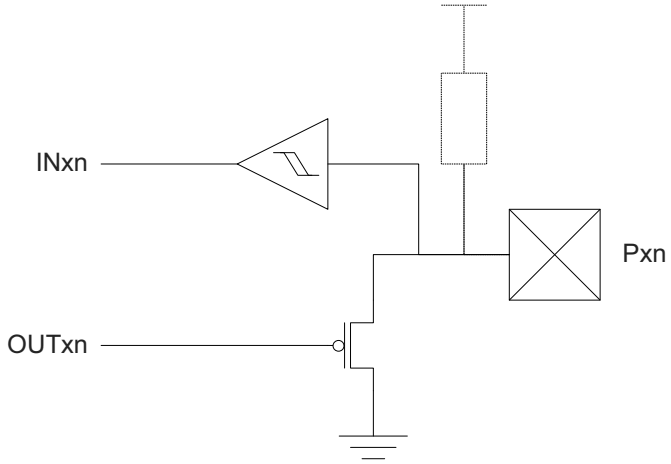


Figure 15-6. I/O Configuration - Wired-AND with Optional Pull-up



24. USART

24.1 Features

- **Three identical USART peripherals**
- **Full-duplex operation**
- **Asynchronous or synchronous operation**
  - **Synchronous clock rates up to 1/2 of the device clock frequency**
  - **Asynchronous clock rates up to 1/8 of the device clock frequency**
- **Supports serial frames with 5, 6, 7, 8, or 9 data bits and 1 or 2 stop bits**
- **Fractional baud rate generator**
  - **Can generate desired baud rate from any system clock frequency**
  - **No need for external oscillator with certain frequencies**
- **Built-in error detection and correction schemes**
  - **Odd or even parity generation and parity check**
  - **Data overrun and framing error detection**
  - **Noise filtering includes false start bit detection and digital low-pass filter**
- **Separate interrupts for**
  - **Transmit complete**
  - **Transmit data register empty**
  - **Receive complete**
- **Multiprocessor communication mode**
  - **Addressing scheme to address a specific devices on a multidevice bus**
  - **Enable unaddressed devices to automatically ignore all frames**
- **Master SPI mode**
  - **Double buffered operation**
  - **Operation up to 1/2 of the peripheral clock frequency**
- **IRCOM module for IrDA compliant pulse modulation/demodulation**

24.2 Overview

The universal synchronous and asynchronous serial receiver and transmitter (USART) is a fast and flexible serial communication module. The USART supports full-duplex communication and asynchronous and synchronous operation. The USART can be configured to operate in SPI master mode and used for SPI communication.

Communication is frame based, and the frame format can be customized to support a wide range of standards. The USART is buffered in both directions, enabling continued data transmission without any delay between frames. Separate interrupts for receive and transmit complete enable fully interrupt driven communication. Frame error and buffer overflow are detected in hardware and indicated with separate status flags. Even or odd parity generation and parity check can also be enabled.

The clock generator includes a fractional baud rate generator that is able to generate a wide range of USART baud rates from any system clock frequencies. This removes the need to use an external crystal oscillator with a specific frequency to achieve a required baud rate. It also supports external clock input in synchronous slave operation.

When the USART is set in master SPI mode, all USART-specific logic is disabled, leaving the transmit and receive buffers, shift registers, and baud rate generator enabled. Pin control and interrupt generation are identical in both modes. The registers are used in both modes, but their functionality differs for some control settings.

An IRCOM module can be enabled for one USART to support IrDA 1.4 physical compliant pulse modulation and demodulation for baud rates up to 115.2kbps.

PORTC, PORTD, and PORTE each has one USART. Notation of these peripherals are USARTC0, USARTD0, and USARTE0, respectively.

31. Pinout and Pin Functions

The device pinout is shown in “Pinout/Block Diagram” on page 3. In addition to general purpose I/O functionality, each pin can have several alternate functions. This will depend on which peripheral is enabled and connected to the actual pin. Only one of the pin functions can be used at time.

31.1 Alternate Pin Function Description

The tables below show the notation for all pin functions available and describe its function.

31.1.1 Operation/Power Supply

V_{CC}	Digital supply voltage
AV_{CC}	Analog supply voltage
GND	Ground

31.1.2 Port Interrupt Functions

SYNC	Port pin with full synchronous and limited asynchronous interrupt function
ASYNC	Port pin with full synchronous and full asynchronous interrupt function

31.1.3 Analog Functions

ACn	Analog Comparator input pin n
ACnOUT	Analog Comparator n Output
ADCn	Analog to Digital Converter input pin n
A_{REF}	Analog Reference input pin

31.1.4 Timer/Counter and AWEX Functions

OCnxLS	Output Compare Channel x Low Side for Timer/Counter n
OCnxHS	Output Compare Channel x High Side for Timer/Counter n

Table 31-5. Port E - Alternate Functions

PORT E	PIN #	INTERRUPT	TCE0	USARTE0	TOSC	TWIE	CLOCKOUT	EVENTOUT
PE3	39	SYNC	OC0D	TXD0				
PE4	40	SYNC						
PE5	41	SYNC						
PE6	42	SYNC			TOSC2			
PE7	43	SYNC			TOSC1		Clk_{PER}	EVOUT
GND	44							
VCC	45							

Table 31-6. Port F - Alternate Functions

PORT F	PIN #	INTERRUPT	TCF0
PF0	46	SYNC	OC0A
PF1	47	SYNC	OC0B
PF2	48	SYNC/ ASYNC	OC0C
PF3	49	SYNC	OC0D
PF4	50	SYNC	
PF5	51	SYNC	
PF6	54	SYNC	
PF7	55	SYNC	
GND	52		
VCC	53		

Table 31-7. Port R - Alternate functions

PORT R	PIN #	INTERRUPT	PDI	XTAL
PDI	56		PDI_DATA	
$\overline{\text{RESET}}$	57		PDI_CLOCK	
PRO	58	SYNC		XTAL2
PR1	59	SYNC		XTAL1

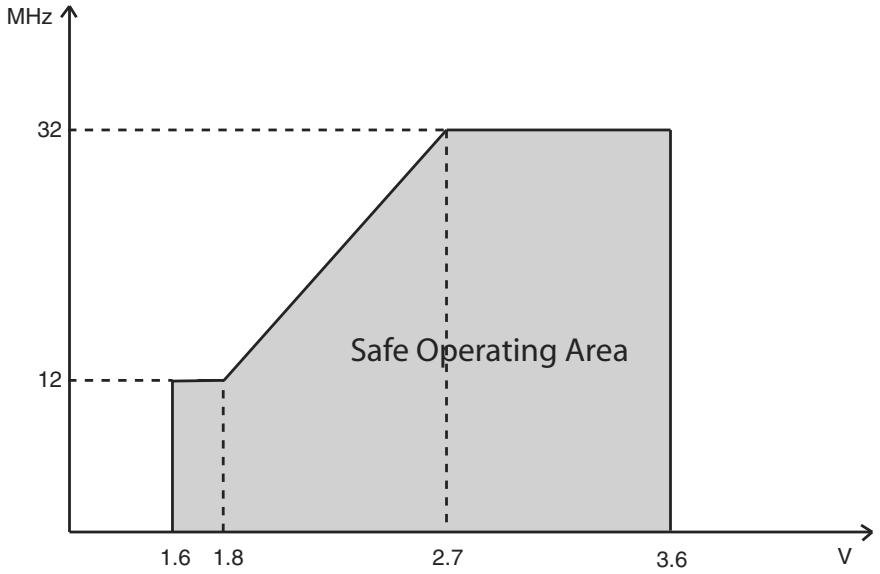
33. Instruction Set Summary

Mnemonics	Operands	Description	Operation	Flags	#Clocks
Arithmetic and Logic Instructions					
ADD	Rd, Rr	Add without Carry	Rd ← Rd + Rr	Z,C,N,V,S,H	1
ADC	Rd, Rr	Add with Carry	Rd ← Rd + Rr + C	Z,C,N,V,S,H	1
ADIW	Rd, K	Add Immediate to Word	Rd ← Rd + 1:Rd + K	Z,C,N,V,S	2
SUB	Rd, Rr	Subtract without Carry	Rd ← Rd - Rr	Z,C,N,V,S,H	1
SUBI	Rd, K	Subtract Immediate	Rd ← Rd - K	Z,C,N,V,S,H	1
SBC	Rd, Rr	Subtract with Carry	Rd ← Rd - Rr - C	Z,C,N,V,S,H	1
SBCI	Rd, K	Subtract Immediate with Carry	Rd ← Rd - K - C	Z,C,N,V,S,H	1
SBIW	Rd, K	Subtract Immediate from Word	Rd + 1:Rd ← Rd + 1:Rd - K	Z,C,N,V,S	2
AND	Rd, Rr	Logical AND	Rd ← Rd • Rr	Z,N,V,S	1
ANDI	Rd, K	Logical AND with Immediate	Rd ← Rd • K	Z,N,V,S	1
OR	Rd, Rr	Logical OR	Rd ← Rd v Rr	Z,N,V,S	1
ORI	Rd, K	Logical OR with Immediate	Rd ← Rd v K	Z,N,V,S	1
EOR	Rd, Rr	Exclusive OR	Rd ← Rd ⊕ Rr	Z,N,V,S	1
COM	Rd	One's Complement	Rd ← $FF - Rd	Z,C,N,V,S	1
NEG	Rd	Two's Complement	Rd ← $00 - Rd	Z,C,N,V,S,H	1
SBR	Rd,K	Set Bit(s) in Register	Rd ← Rd v K	Z,N,V,S	1
CBR	Rd,K	Clear Bit(s) in Register	Rd ← Rd • ($FFh - K)	Z,N,V,S	1
INC	Rd	Increment	Rd ← Rd + 1	Z,N,V,S	1
DEC	Rd	Decrement	Rd ← Rd - 1	Z,N,V,S	1
TST	Rd	Test for Zero or Minus	Rd ← Rd • Rd	Z,N,V,S	1
CLR	Rd	Clear Register	Rd ← Rd ⊕ Rd	Z,N,V,S	1
SER	Rd	Set Register	Rd ← $FF	None	1
MUL	Rd,Rr	Multiply Unsigned	R1:R0 ← Rd x Rr (UU)	Z,C	2
MULS	Rd,Rr	Multiply Signed	R1:R0 ← Rd x Rr (SS)	Z,C	2
MULSU	Rd,Rr	Multiply Signed with Unsigned	R1:R0 ← Rd x Rr (SU)	Z,C	2
FMUL	Rd,Rr	Fractional Multiply Unsigned	R1:R0 ← Rd x Rr<<1 (UU)	Z,C	2
FMULS	Rd,Rr	Fractional Multiply Signed	R1:R0 ← Rd x Rr<<1 (SS)	Z,C	2
FMULSU	Rd,Rr	Fractional Multiply Signed with Unsigned	R1:R0 ← Rd x Rr<<1 (SU)	Z,C	2
DES	K	Data Encryption	if (H = 0) then R15:R0 ← Encrypt(R15:R0, K) else if (H = 1) then R15:R0 ← Decrypt(R15:R0, K)		1/2
Branch instructions					
RJMP	k	Relative Jump	PC ← PC + k + 1	None	2
IJMP		Indirect Jump to (Z)	PC(15:0) ← Z, PC(21:16) ← 0	None	2
EIJMP		Extended Indirect Jump to (Z)	PC(15:0) ← Z, PC(21:16) ← EIND	None	2
JMP	k	Jump	PC ← k	None	3

Mnemonics	Operands	Description	Operation	Flags	#Clocks
RCALL	k	Relative Call Subroutine	PC ← PC + k + 1	None	2 / 3(1)
ICALL		Indirect Call to (Z)	PC(15:0) ← Z, PC(21:16) ← 0	None	2 / 3(1)
EICALL		Extended Indirect Call to (Z)	PC(15:0) ← Z, PC(21:16) ← EIND	None	3(1)
CALL	k	call Subroutine	PC ← k	None	3 / 4(1)
RET		Subroutine Return	PC ← STACK	None	4 / 5(1)
RETI		Interrupt Return	PC ← STACK	I	4 / 5(1)
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	2 / 3 / 4
SBIS	A, b	Skip if Bit in I/O Register Set	If (I/O(A,b) =1) PC ← PC + 2 or 3	None	2 / 3 / 4
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
BRVC	k	Branch if Overflow Flag is Cleared	if (V = 0) then PC ← PC + k + 1	None	1 / 2
BRIE	k	Branch if Interrupt Enabled	if (I = 1) then PC ← PC + k + 1	None	1 / 2
BRID	k	Branch if Interrupt Disabled	if (I = 0) then PC ← PC + k + 1	None	1 / 2
Data transfer instructions					
MOV	Rd, Rr	Copy Register	Rd ← Rr	None	1
MOVW	Rd, Rr	Copy Register Pair	Rd+1:Rd ← Rr+1:Rr	None	1

The maximum CPU clock frequency depends on V_{CC}. As shown in Figure 35-1 the Frequency vs. V_{CC} curve is linear between 1.8V < V_{CC} < 2.7V.

Figure 35-1. Maximum Frequency vs. V_{CC}



35.6 ADC Characteristics

Table 35-8. Power Supply, Reference, and Input Range

Symbol	Parameter	Condition	Min.	Typ.	Max.	Units
AV_{CC}	Analog supply voltage		V_{CC}-0.3		V_{CC}+0.3	V
V_{REF}	Reference voltage		1		AV_{CC}-0.6	
R_{in}	Input resistance	Switched			4.5	kΩ
C_{in}	Input capacitance	Switched			5	pF
R_{AREF}	Reference input resistance	(leakage only)		>10		MΩ
C_{AREF}	Reference input capacitance	Static load		7		pF
Vin	Input range		0		V_{REF}	V
	Conversion range	Differential mode, Vinp - Vinn	$-V_{REF}$		V_{REF}	
	Conversion range	Single ended unsigned mode, Vinp	-ΔV		V_{REF}-ΔV	
ΔV	Fixed offset voltage			200		lsb

Table 35-9. Clock and Timing

Symbol	Parameter	Condition	Min.	Typ.	Max.	Units
Clk_{ADC}	ADC clock frequency	Maximum is 1/4 of peripheral clock frequency	100		1800	kHz
		Measuring internal signals	100		125	
f_{ClkADC}	Sample rate		16		300	ksps
f_{ADC}	Sample rate	Current limitation (CURRLIMIT) off	16		300	
		CURRLIMIT = LOW	16		250	
		CURRLIMIT = MEDIUM	16		150	
		CURRLIMIT = HIGH	16		50	
	Sampling time	Configurable in steps of 1/2 Clk_{ADC} cycles up to 32 Clk_{ADC} cycles	0.28		320	µs
	Conversion time (latency)	(RES+1)/2 + GAIN RES (Resolution) = 8 or 12, GAIN=0 to 3	5.5		10	Clk_{ADC} cycles
	Start-up time	ADC clock cycles		12	24	
	ADC settling time	After changing reference or input mode		7	7	

Table 35-10. Accuracy Characteristics

Symbol	Parameter	Condition[2]		Min.	Typ.	Max.	Units
RES	Resolution	12-bit resolution	Differential	8	12	12	Bits
			Single ended signed	7	11	11	
			Single ended unsigned	8	12	12	
INL[1]	Integral non-linearity	Differential mode	16ksps, V_{REF} = 3V		0.5	1	lsb
			16ksps, all V_{REF}		0.8	2	
			300ksps, V_{REF} = 3V		0.6	1	
			300ksps, all V_{REF}		1	2	
		Single ended unsigned mode	16ksps, V_{REF} = 3.0V		0.5	1	
			16ksps, all V_{REF}		1.3	2	
DNL[1]	Differential non-linearity	Differential mode	16ksps, V_{REF} = 3V		0.3	1	
			16ksps, all V_{REF}		0.5	1	
			300ksps, V_{REF} = 3V		0.35	1	
			300ksps, all V_{REF}		0.5	1	
		Single ended unsigned mode	16ksps, V_{REF} = 3.0V		0.6	1	
			16ksps, all V_{REF}		0.6	1	
	Offset error	Differential mode	300ksps, V_{REF}=3V		-7		mV
			Temperature drift, V_{REF}=3V		0.01		mV/K
			Operating voltage drift		0.16		mV/V
	Gain error	Differential mode	External reference		-5		mV
			AV_{CC}/1.6		-5		
			AV_{CC}/2.0		-6		
			Bandgap		±10		
			Temperature drift		0.02		mV/K
			Operating voltage drift		2		mV/V
	Gain error	Single ended unsigned mode	External reference		-8		mV
			AV_{CC}/1.6		-8		
			AV_{CC}/2.0		-8		
			Bandgap		±10		
			Temperature drift		0.03		mV/K
			Operating voltage drift		2		mV/V

Notes:
1. Maximum numbers are based on characterisation and not tested in production, and valid for 5% to 95% input voltage range.
2. Unless otherwise noted all linearity, offset and gain error numbers are valid under the condition that external VREF is used.

35.8 Bandgap and Internal 1.0V Reference Characteristics

Table 35-13. Bandgap and Internal 1.0V Reference Characteristics

Symbol	Parameter	Condition	Min.	Typ.	Max.	Units
	Startup time	As reference for ADC		1 Clk_{PER} + 2.5µs		µs
		As input voltage to ADC and AC		1.5		
	Bandgap voltage			1.1		V
INT1V	Internal 1.00V reference	T= 85°C, after calibration	0.99	1	1.01	
	Variation over voltage and temperature	Calibrated at T= 85°C		2		%

35.9 Brownout Detection Characteristics

Table 35-14. Brownout Detection Characteristics(1)

Symbol	Parameter	Condition	Min.	Typ.	Max.	Units
V_{BOT}	BOD level 0 falling V_{CC}		1.60	1.62	1.72	V
	BOD level 1 falling V_{CC}			1.8		
	BOD level 2 falling V_{CC}			2.0		
	BOD level 3 falling V_{CC}			2.2		
	BOD level 4 falling V_{CC}			2.4		
	BOD level 5 falling V_{CC}			2.6		
	BOD level 6 falling V_{CC}			2.8		
	BOD level 7 falling V_{CC}			3.0		
t_{BOD}	Detection time	Continuous mode		0.4		µs
		Sampled mode		1000		
V_{HYST}	Hysteresis			1.0		%

Note: 1. BOD is calibrated at 85°C within BOD level 0 values, and BOD level 0 is the default level.

35.10 External Reset Characteristics

Table 35-15. External Reset Characteristics

Symbol	Parameter	Condition	Min.	Typ.	Max.	Units
t_{EXT}	Minimum reset pulse width		1000	90		ns
V_{RST}	Reset threshold voltage	V_{CC} = 2.7 - 3.6V		0.45*V_{CC}		V
		V_{CC} = 1.6 - 2.7V		0.45*V_{CC}		
R_{RST}	Reset pin pull-up resistor			25		kΩ

36. Typical Characteristics

36.1 Current Consumption

36.1.1 Active Mode Supply Current

Figure 36-1. Active Supply Current vs. Frequency

f_{SYS} = 0 - 1MHz external clock, T = 25°C

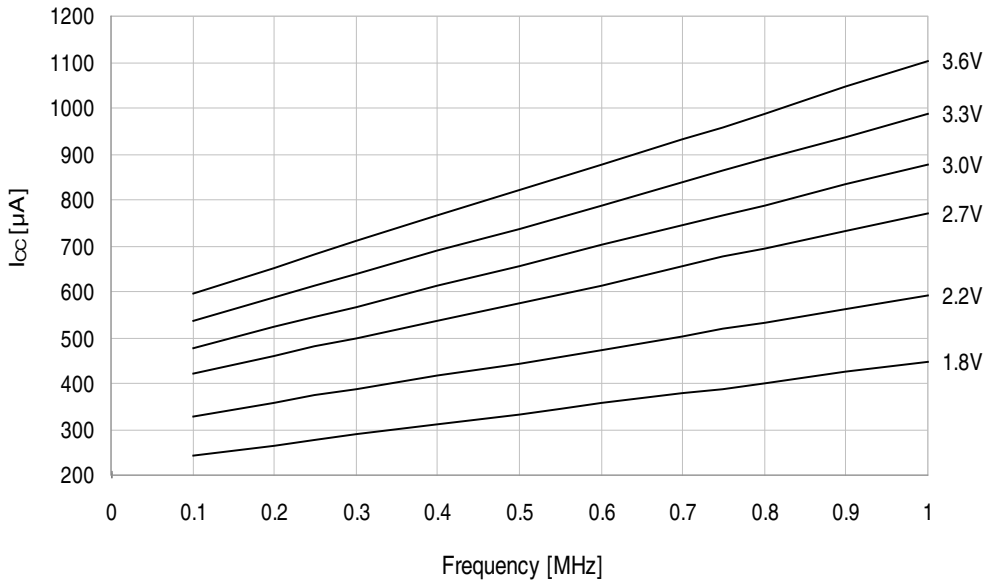


Figure 36-2. Active Supply Current vs. Frequency

f_{SYS} = 1 - 32MHz external clock, T = 25°C

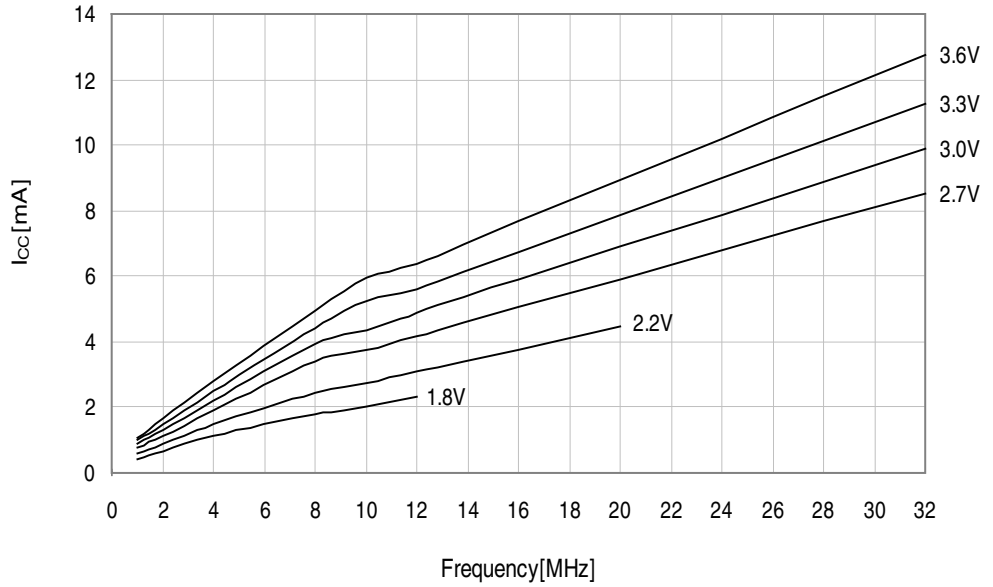
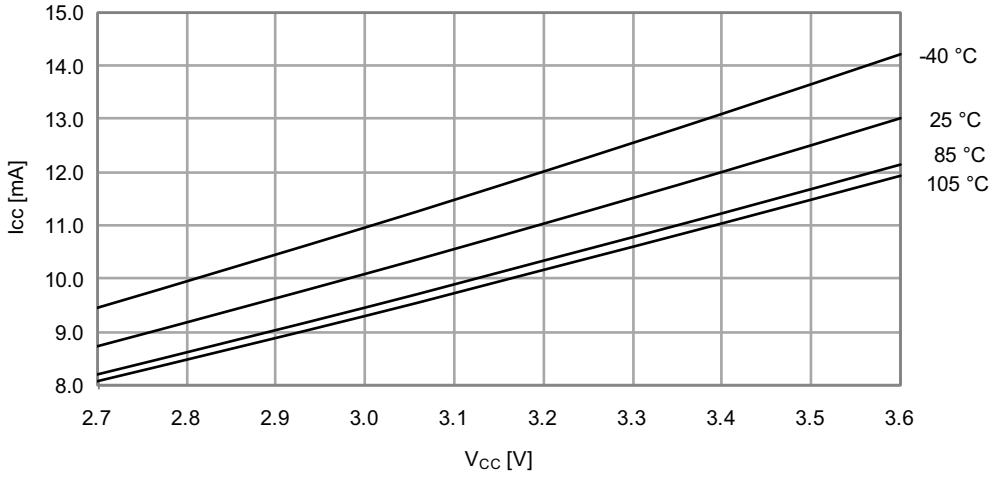


Figure 36-7. Active Mode Supply Current vs. V_{CC}

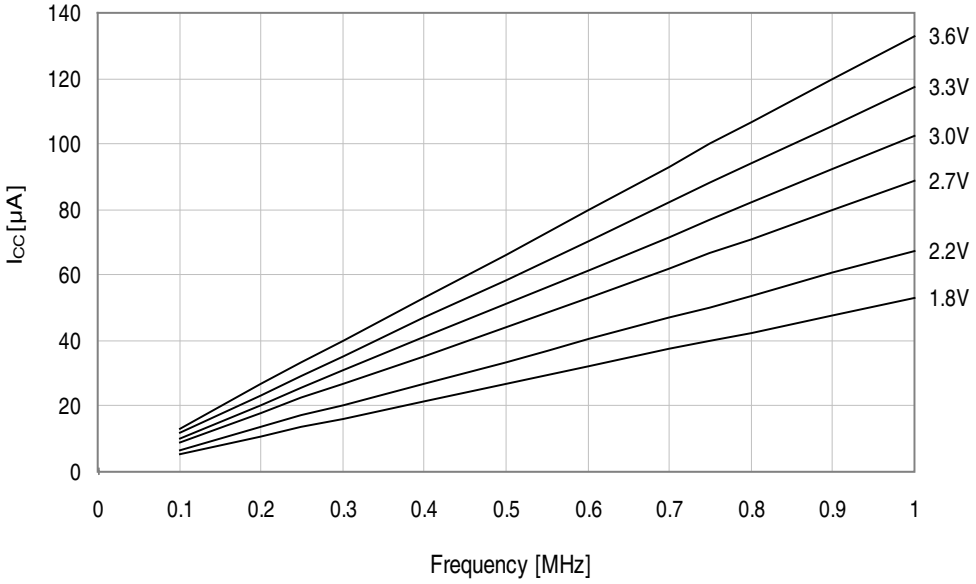
f_{SYS} = 32MHz internal oscillator



36.1.2 Idle Mode Supply Current

Figure 36-8. Idle Mode Supply Current vs. Frequency

f_{SYS} = 0 - 1MHz external clock, T = 25°C



36.2.2 Output Voltage vs. Sink/Source Current

Figure 36-21.I/O Pin Output Voltage vs. Source Current

V_{CC} = 1.8V

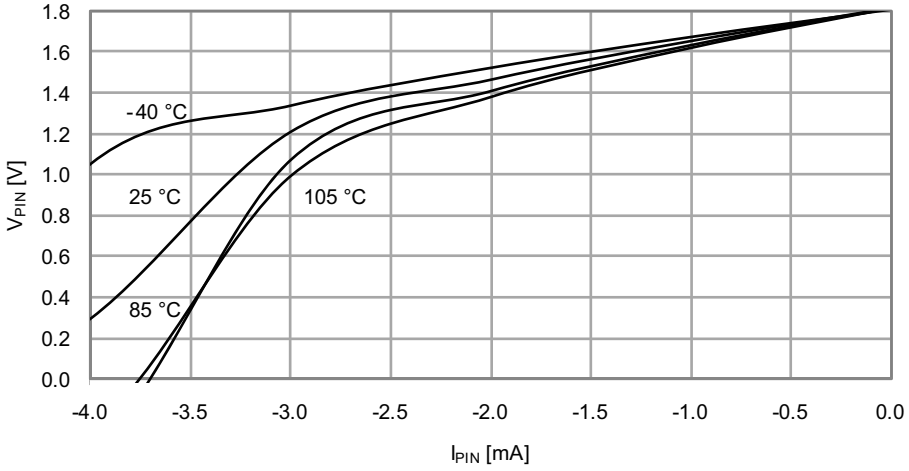


Figure 36-22.I/O Pin Output Voltage vs. Source Current

V_{CC} = 3.0V

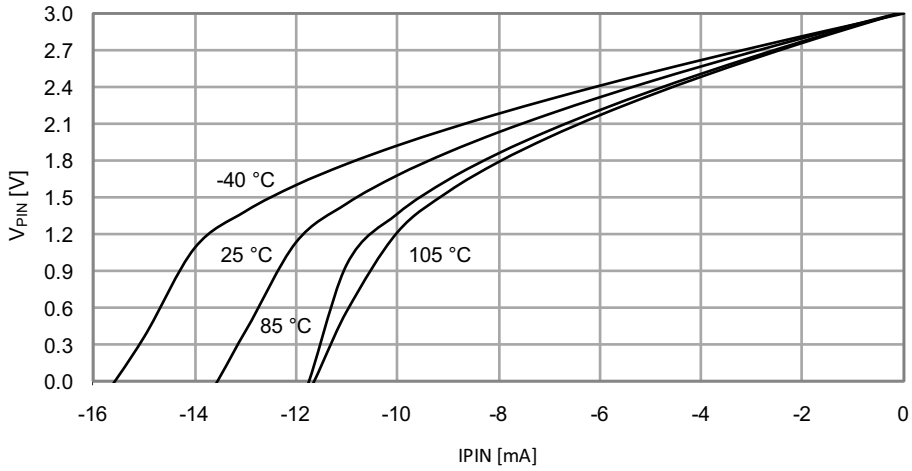


Figure 36-49.Reset Pin Pull-up Resistor Current vs. Reset Pin Voltage

V_{CC} = 1.8V

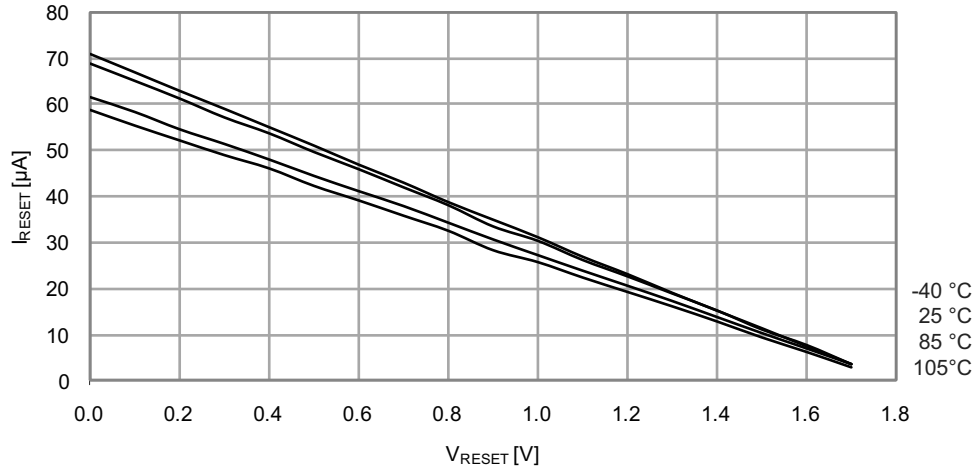


Figure 36-50.Reset Pin Pull-up Resistor Current vs. Reset Pin Voltage

V_{CC} = 3.0V