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	32-Bit Single-Core
Speed	66MHz
Connectivity	CANbus, EBI/EMI, Ethernet, I²C, IrDA, LINbus, SPI, UART/USART, USB
Peripherals	Brown-out Detect/Reset, DMA, I²S, POR, PWM, WDT
Number of I/O	123
Program Memory Size	128KB (128K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	32K x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 5.5V
Data Converters	A/D 16x12b; D/A 4x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	144-LQFP
Supplier Device Package	144-LQFP (20x20)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/at32uc3c0128c-alut

Table 3-1. GPIO Controller Function Multiplexing

TQFP / QFN 64	TQFP 100	LQFP 144	PIN	GPIO	Supply	Pin Type (1)	GPIO function					
							A	B	C	D	E	F
16	25	36	PA19	19	VDDANA	x1/x2	ADCIN8	EIC - EXTINT[1]				
19	28	39	PA20	20	VDDANA	x1/x2	ADCIN9	AC0AP0	AC0AP0 or DAC0A			
20	29	40	PA21	21	VDDANA	x1/x2	ADCIN10	AC0BN0	AC0BN0 or DAC0B			
21	30	41	PA22	22	VDDANA	x1/x2	ADCIN11	AC0AN0	PEVC - PAD_EVT [4]		MACB - SPEED	
22	31	42	PA23	23	VDDANA	x1/x2	ADCIN12	AC0BP0	PEVC - PAD_EVT [5]		MACB - WOL	
	32	43	PA24	24	VDDANA	x1/x2	ADCIN13	SPI1 - NPCS[2]				
	33	44	PA25	25	VDDANA	x1/x2	ADCIN14	SPI1 - NPCS[3]	EIC - EXTINT[0]			
		45	PA26	26	VDDANA	x1/x2	AC0AP1	EIC - EXTINT[1]				
		46	PA27	27	VDDANA	x1/x2	AC0AN1	EIC - EXTINT[2]				
		47	PA28	28	VDDANA	x1/x2	AC0BP1	EIC - EXTINT[3]				
		48	PA29	29	VDDANA	x1/x2	AC0BN1	EIC - EXTINT[0]				
62	96	140	PB00	32	VDDIO1	x1	USART0 - CLK	CANIF - RXLINE[1]	EIC - EXTINT[8]	PEVC - PAD_EVT [10]		
63	97	141	PB01	33	VDDIO1	x1		CANIF - TXLINE[1]		PEVC - PAD_EVT [11]		
	99	143	PB02	34	VDDIO1	x1		USBC - ID	PEVC - PAD_EVT [6]	TC1 - A1		
	100	144	PB03	35	VDDIO1	x1		USBC - VBOF	PEVC - PAD_EVT [7]			
	7	7	PB04	36	VDDIO1	x1/x2	SPI1 - MOSI	CANIF - RXLINE[0]	QDEC1 - QEPI		MACB - TXD[2]	
	8	8	PB05	37	VDDIO1	x1/x2	SPI1 - MISO	CANIF - TXLINE[0]	PEVC - PAD_EVT [12]	USART3 - CLK	MACB - TXD[3]	
	9	9	PB06	38	VDDIO1	x2/x4	SPI1 - SCK		QDEC1 - QEPA	USART1 - CLK	MACB - TX_ER	
		10	PB07	39	VDDIO1	x1/x2	SPI1 - NPCS[0]	EIC - EXTINT[2]	QDEC1 - QEPB		MACB - RX_DV	
		11	PB08	40	VDDIO1	x1/x2	SPI1 - NPCS[1]	PEVC - PAD_EVT [1]	PWM - PWML[0]		MACB - RXD[0]	
		12	PB09	41	VDDIO1	x1/x2	SPI1 - NPCS[2]		PWM - PWMH[0]		MACB - RXD[1]	
		13	PB10	42	VDDIO1	x1/x2	USART1 - DTR	SPI0 - MOSI	PWM - PWML[1]			

Table 3-1. GPIO Controller Function Multiplexing

TQFP / QFN 64	TQFP 100	LQFP 144	PIN	GPIO	Supply	Pin Type (1)	GPIO function					
							A	B	C	D	E	F
		14	PB11	43	VDDIO1	x1/x2	USART1 - DSR	SPI0 - MISO	PWM - PWMH[1]			
		15	PB12	44	VDDIO1	x1/x2	USART1 - DCD	SPI0 - SCK	PWM - PWML[2]			
		16	PB13	45	VDDIO1	x1/x2	USART1 - RI	SPI0 - NPCS[0]	PWM - PWMH[2]		MACB - RX_ER	
		17	PB14	46	VDDIO1	x1/x2	USART1 - RTS	SPI0 - NPCS[1]	PWM - PWML[3]		MACB - MDC	
		18	PB15	47	VDDIO1	x1/x2	USART1 - CTS	USART1 - CLK	PWM - PWMH[3]		MACB - MDIO	
		19	PB16	48	VDDIO1	x1/x2	USART1 - RXD	SPI0 - NPCS[2]	PWM - EXT_ FAULTS[0]		CANIF - RXLINE[0]	
		20	PB17	49	VDDIO1	x1/x2	USART1 - TXD	SPI0 - NPCS[3]	PWM - EXT_ FAULTS[1]		CANIF - TXLINE[0]	
		57	PB18	50	VDDIO2	x1/x2	TC0 - CLK2		EIC - EXTINT[4]			
	42	58	PB19	51	VDDIO2	x1/x2	TC0 - A0	SPI1 - MOSI	IISC - ISDO		MACB - CRS	
	43	59	PB20	52	VDDIO2	x1/x2	TC0 - B0	SPI1 - MISO	IISC - ISDI	ACIFA1 - ACAOUT	MACB - COL	
	44	60	PB21	53	VDDIO2	x2/x4	TC0 - CLK1	SPI1 - SCK	IISC - IMCK	ACIFA1 - ACBOUT	MACB - RXD[2]	
	45	61	PB22	54	VDDIO2	x1/x2	TC0 - A1	SPI1 - NPCS[3]	IISC - ISCK	SCIF - GCLK[0]	MACB - RXD[3]	
	46	62	PB23	55	VDDIO2	x1/x2	TC0 - B1	SPI1 - NPCS[2]	IISC - IWS	SCIF - GCLK[1]	MACB - RX_CLK	
		63	PB24	56	VDDIO2	x1/x2	TC0 - CLK0	SPI1 - NPCS[1]				
		64	PB25	57	VDDIO2	x1/x2	TC0 - A2	SPI1 - NPCS[0]	PEVC - PAD_EVT [8]			
		65	PB26	58	VDDIO2	x2/x4	TC0 - B2	SPI1 - SCK	PEVC - PAD_EVT [9]		MACB - TX_EN	
		66	PB27	59	VDDIO2	x1/x2	QDEC0 - QEPA	SPI1 - MISO	PEVC - PAD_EVT [10]	TC1 - CLK0	MACB - TXD[0]	
		67	PB28	60	VDDIO2	x1/x2	QDEC0 - QEPB	SPI1 - MOSI	PEVC - PAD_EVT [11]	TC1 - B0	MACB - TXD[1]	
		68	PB29	61	VDDIO2	x1/x2	QDEC0 - QEPI	SPI0 - NPCS[0]	PEVC - PAD_EVT [12]	TC1 - A0		
31	47	69	PB30	62	VDDIO2	x1						
32	48	70	PB31	63	VDDIO2	x1						
	49	71	PC00	64	VDDIO2	x1/x2	USBC - ID	SPI0 - NPCS[1]	USART2 - CTS	TC1 - B2	CANIF - TXLINE[1]	
	50	72	PC01	65	VDDIO2	x1/x2	USBC - VBOF	SPI0 - NPCS[2]	USART2 - RTS	TC1 - A2	CANIF - RXLINE[1]	

depending on the configuration of the OCD AXS register. For details, see the AVR32UC Technical Reference Manual.

Table 3-5. Nexus OCD AUX port connections

Pin	AXS=0	AXS=1	AXS=2
EVTI_N	PA08	PB19	PA10
MDO[5]	PC05	PC31	PB06
MDO[4]	PC04	PC12	PB15
MDO[3]	PA23	PC11	PB14
MDO[2]	PA22	PB23	PA27
MDO[1]	PA19	PB22	PA26
MDO[0]	PA09	PB20	PA19
EVTO_N	PD29	PD29	PD29
MCKO	PD13	PB21	PB26
MSEO[1]	PD30	PD08	PB25
MSEO[0]	PD14	PD07	PB18

3.2.6 Other Functions

The functions listed in Table 3-6 are not mapped to the normal GPIO functions. The aWire DATA pin will only be active after the aWire is enabled. The aWire DATAOUT pin will only be active after the aWire is enabled and the 2_PIN_MODE command has been sent.

Table 3-6. Other Functions

QFN64/ TQFP64 pin	TQFP100 pin	LQFP144 pin	Pad	Oscillator pin
64	98	142	RESET_N	aWire DATA
3	3	3	PA02	aWire DATAOUT

3.3 Signals Description

The following table give details on the signal name classified by peripherals.

Table 3-7. Signal Description List

Signal Name	Function	Type	Active Level	Comments
Power				
VDDIO1 VDDIO2 VDDIO3	I/O Power Supply	Power Input		4.5V to 5.5V or 3.0V to 3.6 V
VDDANA	Analog Power Supply	Power Input		4.5V to 5.5V or 3.0V to 3.6 V

Table 3-7. Signal Description List

Signal Name	Function	Type	Active Level	Comments
RX_CLK	Receive Clock	Input		
RX_DV	Receive Data Valid	Input		
RX_ER	Receive Coding Error	Input		
SPEED	Speed	Output		
TXD[3:0]	Transmit Data	Output		
TX_CLK	Transmit Clock or Reference Clock	Input		
TX_EN	Transmit Enable	Output		
TX_ER	Transmit Coding Error	Output		
WOL	Wake-On-LAN	Output		
Peripheral Event Controller - PEVC				
PAD_EVT[15:0]	Event Input Pins	Input		
Power Manager - PM				
RESET_N	Reset Pin	Input	Low	
Pulse Width Modulator - PWM				
PWMH[3:0] PWML[3:0]	PWM Output Pins	Output		
EXT_FAULT[1:0]	PWM Fault Input Pins	Input		
Quadrature Decoder- QDEC0/QDEC1				
QEPA	QEPA quadrature input	Input		
QEPB	QEPB quadrature input	Input		
QEPI	Index input	Input		
System Controller Interface- SCIF				
XIN0, XIN1, XIN32	Crystal 0, 1, 32K Inputs	Analog		
XOUT0, XOUT1, XOUT32	Crystal 0, 1, 32K Output	Analog		
GCLK0 - GCLK1	Generic Clock Pins	Output		
Serial Peripheral Interface - SPI0, SPI1				
MISO	Master In Slave Out	I/O		
MOSI	Master Out Slave In	I/O		

4. Processor and Architecture

Rev: 2.1.2.0

This chapter gives an overview of the AVR32UC CPU. AVR32UC is an implementation of the AVR32 architecture. A summary of the programming model, instruction set, and MPU is presented. For further details, see the *AVR32 Architecture Manual* and the *AVR32UC Technical Reference Manual*.

4.1 Features

- **32-bit load/store AVR32A RISC architecture**
  - **15 general-purpose 32-bit registers**
  - **32-bit Stack Pointer, Program Counter and Link Register reside in register file**
  - **Fully orthogonal instruction set**
  - **Privileged and unprivileged modes enabling efficient and secure operating systems**
  - **Innovative instruction set together with variable instruction length ensuring industry leading code density**
  - **DSP extension with saturating arithmetic, and a wide variety of multiply instructions**
- **3-stage pipeline allowing one instruction per clock cycle for most instructions**
  - **Byte, halfword, word, and double word memory access**
  - **Multiple interrupt priority levels**
- **MPU allows for operating systems with memory protection**
- **FPU enables hardware accelerated floating point calculations**
- **Secure State for supporting FlashVault technology**

4.2 AVR32 Architecture

AVR32 is a new, high-performance 32-bit RISC microprocessor architecture, designed for cost-sensitive embedded applications, with particular emphasis on low power consumption and high code density. In addition, the instruction set architecture has been tuned to allow a variety of microarchitectures, enabling the AVR32 to be implemented as low-, mid-, or high-performance processors. AVR32 extends the AVR family into the world of 32- and 64-bit applications.

Through a quantitative approach, a large set of industry recognized benchmarks has been compiled and analyzed to achieve the best code density in its class. In addition to lowering the memory requirements, a compact code size also contributes to the core's low power characteristics. The processor supports byte and halfword data types without penalty in code size and performance.

Memory load and store operations are provided for byte, halfword, word, and double word data with automatic sign- or zero extension of halfword and byte data. The C-compiler is closely linked to the architecture and is able to exploit code optimization features, both for size and speed.

In order to reduce code size to a minimum, some instructions have multiple addressing modes. As an example, instructions with immediates often have a compact format with a smaller immediate, and an extended format with a larger immediate. In this way, the compiler is able to use the format giving the smallest code size.

Another feature of the instruction set is that frequently used instructions, like add, have a compact format with two operands as well as an extended format with three operands. The larger format increases performance, allowing an addition and a data move in the same instruction in a

4.3.2.5 *Unaligned Reference Handling*

AVR32UC does not support unaligned accesses, except for doubleword accesses. AVR32UC is able to perform word-aligned *st.d* and *ld.d*. Any other unaligned memory access will cause an address exception. Doubleword-sized accesses with word-aligned pointers will automatically be performed as two word-sized accesses.

The following table shows the instructions with support for unaligned addresses. All other instructions require aligned addresses.

Table 4-1. Instructions with Unaligned Reference Support

Instruction	Supported Alignment
ld.d	Word
st.d	Word

4.3.2.6 *Unimplemented Instructions*

The following instructions are unimplemented in AVR32UC, and will cause an Unimplemented Instruction Exception if executed:

- All SIMD instructions
- All coprocessor instructions if no coprocessors are present
- retj, incjosp, popjc, pushjc
- tlbr, tlbs, tlbw
- cache

4.3.2.7 *CPU and Architecture Revision*

Three major revisions of the AVR32UC CPU currently exist. The device described in this datasheet uses CPU revision 3.

The Architecture Revision field in the CONFIG0 system register identifies which architecture revision is implemented in a specific device.

AVR32UC CPU revision 3 is fully backward-compatible with revisions 1 and 2, ie. code compiled for revision 1 or 2 is binary-compatible with revision 3 CPUs.

Debug state can be entered as described in the *AVR32UC Technical Reference Manual*.

Debug state is exited by the *retd* instruction.

4.4.3.3 Secure State

The AVR32 can be set in a secure state, that allows a part of the code to execute in a state with higher security levels. The rest of the code can not access resources reserved for this secure code. Secure State is used to implement FlashVault technology. Refer to the *AVR32UC Technical Reference Manual* for details.

4.4.4 System Registers

The system registers are placed outside of the virtual memory space, and are only accessible using the privileged *mfsr* and *mtsr* instructions. The table below lists the system registers specified in the AVR32 architecture, some of which are unused in AVR32UC. The programmer is responsible for maintaining correct sequencing of any instructions following a *mtsr* instruction. For detail on the system registers, refer to the *AVR32UC Technical Reference Manual*.

Table 4-3. System Registers

Reg #	Address	Name	Function
0	0	SR	Status Register
1	4	EVBA	Exception Vector Base Address
2	8	ACBA	Application Call Base Address
3	12	CPUCR	CPU Control Register
4	16	ECR	Exception Cause Register
5	20	RSR_SUP	Unused in AVR32UC
6	24	RSR_INT0	Unused in AVR32UC
7	28	RSR_INT1	Unused in AVR32UC
8	32	RSR_INT2	Unused in AVR32UC
9	36	RSR_INT3	Unused in AVR32UC
10	40	RSR_EX	Unused in AVR32UC
11	44	RSR_NMI	Unused in AVR32UC
12	48	RSR_DBG	Return Status Register for Debug mode
13	52	RAR_SUP	Unused in AVR32UC
14	56	RAR_INT0	Unused in AVR32UC
15	60	RAR_INT1	Unused in AVR32UC
16	64	RAR_INT2	Unused in AVR32UC
17	68	RAR_INT3	Unused in AVR32UC
18	72	RAR_EX	Unused in AVR32UC
19	76	RAR_NMI	Unused in AVR32UC
20	80	RAR_DBG	Return Address Register for Debug mode
21	84	JECR	Unused in AVR32UC
22	88	JOSP	Unused in AVR32UC
23	92	JAVA_LV0	Unused in AVR32UC

Table 4-3. System Registers (Continued)

Reg #	Address	Name	Function
24	96	JAVA_LV1	Unused in AVR32UC
25	100	JAVA_LV2	Unused in AVR32UC
26	104	JAVA_LV3	Unused in AVR32UC
27	108	JAVA_LV4	Unused in AVR32UC
28	112	JAVA_LV5	Unused in AVR32UC
29	116	JAVA_LV6	Unused in AVR32UC
30	120	JAVA_LV7	Unused in AVR32UC
31	124	JTBA	Unused in AVR32UC
32	128	JBCR	Unused in AVR32UC
33-63	132-252	Reserved	Reserved for future use
64	256	CONFIG0	Configuration register 0
65	260	CONFIG1	Configuration register 1
66	264	COUNT	Cycle Counter register
67	268	COMPARE	Compare register
68	272	TLBEHI	Unused in AVR32UC
69	276	TLBELO	Unused in AVR32UC
70	280	PTBR	Unused in AVR32UC
71	284	TLBEAR	Unused in AVR32UC
72	288	MMUCR	Unused in AVR32UC
73	292	TLBARLO	Unused in AVR32UC
74	296	TLBARHI	Unused in AVR32UC
75	300	PCCNT	Unused in AVR32UC
76	304	PCNT0	Unused in AVR32UC
77	308	PCNT1	Unused in AVR32UC
78	312	PCCR	Unused in AVR32UC
79	316	BEAR	Bus Error Address Register
80	320	MPUAR0	MPU Address Register region 0
81	324	MPUAR1	MPU Address Register region 1
82	328	MPUAR2	MPU Address Register region 2
83	332	MPUAR3	MPU Address Register region 3
84	336	MPUAR4	MPU Address Register region 4
85	340	MPUAR5	MPU Address Register region 5
86	344	MPUAR6	MPU Address Register region 6
87	348	MPUAR7	MPU Address Register region 7
88	352	MPUPSR0	MPU Privilege Select Register region 0
89	356	MPUPSR1	MPU Privilege Select Register region 1

Table 7-2. Supply Rise Rates and Order

Symbol	Parameter	Rise Rate		
		Min	Max	Comment
V_{VDDIN_5}	DC supply internal 3.3V regulator	0.01 V/ms	1.25 V/us	
V_{VDDIN_33}	DC supply internal 1.8V regulator	0.01 V/ms	1.25 V/us	
V_{VDDIO1} V_{VDDIO2} V_{VDDIO3}	DC supply peripheral I/O	0.01 V/ms	1.25 V/us	Rise after or at the same time as VDDIN_5, VDDIN_33
V_{VDDANA}	DC supply peripheral I/O and analog part	0.01 V/ms	1.25 V/us	Rise after or at the same time as VDDIN_5, VDDIN_33

7.3 Maximum Clock Frequencies

These parameters are given in the following conditions:

- $V_{VDDCORE}$ > 1.85V
- Temperature = -40°C to 85°C

Table 7-3. Clock Frequencies

Symbol	Parameter	Conditions	Min	Max	Units
f_{CPU}	CPU clock frequency			66	MHz
f_{PBA}	PBA clock frequency			66	MHz
f_{PBB}	PBB clock frequency			66	MHz
f_{PBC}	PBC clock frequency			66	MHz
f_{GCLK0}	GCLK0 clock frequency	Generic clock for USBC		50(1)	MHz
f_{GCLK1}	GCLK1 clock frequency	Generic clock for CANIF		66(1)	MHz
f_{GCLK2}	GCLK2 clock frequency	Generic clock for AST		80(1)	MHz
f_{GCLK4}	GCLK4 clock frequency	Generic clock for PWM		133(1)	MHz
f_{GCLK11}	GCLK11 clock frequency	Generic clock for IISC		50(1)	MHz

Note: 1. These values are based on simulation and characterization of other AVR microcontrollers manufactured in the same process technology. These values are not covered by test limits in production.

7.4 Power Consumption

The values in Table 7-4 are measured values of power consumption under the following conditions, except where noted:

- Operating conditions core supply (Figure 7-1)
  - V_{VDDIN_5} = V_{VDDIN_33} = 3.3V
  - $V_{VDDCORE}$ = 1.85V, supplied by the internal regulator
  - V_{VDDIO1} = V_{VDDIO2} = V_{VDDIO3} = 3.3V
  - V_{VDDANA} = 3.3V

7.8.4 3.3V Brown Out Detector (BOD33) Characteristics

The values in Table 7-23 describe the values of the BOD33.LEVEL field in the SCIF module.

Table 7-23. BOD33.LEVEL Values

BOD33.LEVEL Value	Parameter	Min	Max	Units
17		2.21	2.55	V
22		2.30	2.64	
27		2.39	2.74	
31	threshold at power-up sequence	2.46	2.82	
33		2.50	2.86	
39		2.60	2.98	
44		2.69	3.08	
49		2.78	3.18	
53		2.85	3.27	
60		2.98	3.41	

7.8.5 5V Brown Out Detector (BOD50) Characteristics

The values in Table 7-25 describe the values of the BOD50.LEVEL field in the SCIF module.

Table 7-25. BOD50.LEVEL Values

BOD50.LEVEL Value	Parameter	Min	Max	Units
16		3.20	3.65	V
25		3.42	3.92	
35		3.68	4.22	
44		3.91	4.48	
53		4.15	4.74	
61		4.36	4.97	

7.8.6 Analog to Digital Converter (ADC) and sample and hold (S/H) Characteristics

Table 7-27. ADC and S/H characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Units
f_{ADC}	ADC clock frequency	12-bit resolution mode, V_{VDDANA} = 3V			1.2	MHz
		10-bit resolution mode, V_{VDDANA} = 3V			1.6	
		8-bit resolution mode, V_{VDDANA} = 3V			2.2	
		12-bit resolution mode, V_{VDDANA} = 4.5V			1.5	
		10-bit resolution mode, V_{VDDANA} = 4.5V			2	
		8-bit resolution mode, V_{VDDANA} = 4.5V			2.4	
$t_{STARTUP}$	Startup time	ADC cold start-up			1	ms
		ADC hot start-up			24	ADC clock cycles
t_{CONV}	Conversion time (latency)	(ADCIFA.SEQCFGn.SRES)/2 + 2, ADCIFA.CFG.SHD = 1	6		8	ADC clock cycles
		(ADCIFA.SEQCFGn.SRES)/2 + 3, ADCIFA.CFG.SHD = 0	7		9	
	Throughput rate	12-bit resolution, ADC clock = 1.2 MHz, V_{VDDANA} = 3V			1.2	MSPS
		10-bit resolution, ADC clock = 1.6 MHz, V_{VDDANA} = 3V			1.6	
		12-bit resolution, ADC clock = 1.5 MHz, V_{VDDANA} = 4.5V			1.5	
		10-bit resolution, ADC clock = 2 MHz, V_{VDDANA} = 4.5V			2	

Table 7-28. ADC Reference Voltage

Symbol	Parameter	Conditions	Min	Typ	Max	Units
$V_{ADCREF0}$	ADCREF0 input voltage range	5V Range	1		3.5	V
		3V Range	1		V_{VDDANA}-0.7	
$V_{ADCREF1}$	ADCREF1 input voltage range	5V Range	1		3.5	V
		3V Range	1		V_{VDDANA}-0.7	
$V_{ADCREFP}$	ADCREFP input voltage	5V Range - Voltage reference applied on ADCREFP	1		3.5	V
		3V Range - Voltage reference applied on ADCREFP	1		V_{VDDANA}-0.7	
$V_{ADCREFN}$	ADCREFN input voltage	Voltage reference applied on ADCREFN		GNDANA		V
	Internal 1V reference			1.0		V
	Internal 0.6*VDDANA reference			0.6*V_{VDDANA}		V

Figure 7-12. SPI Master Mode With (CPOL= 0 and NCPHA= 1) or (CPOL= 1 and NCPHA= 0)

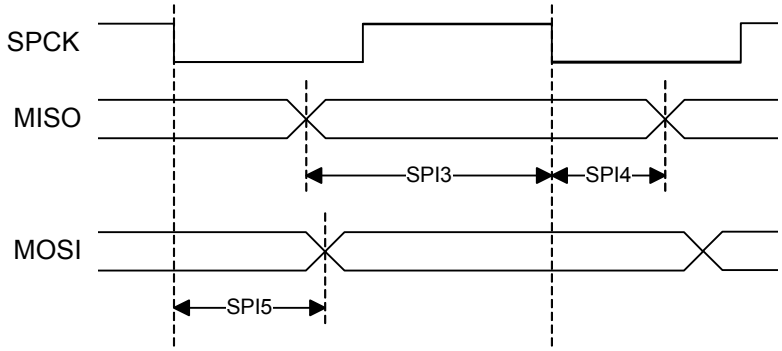


Table 7-48. SPI Timing, Master Mode[1]

Symbol	Parameter	Conditions	Min	Max	Units
SPI0	MISO setup time before SPCK rises	external capacitor = 40pF	28.5+ (t_{CLK_SPI})/2		ns
SPI1	MISO hold time after SPCK rises		0		ns
SPI2	SPCK rising to MOSI delay			10.5	ns
SPI3	MISO setup time before SPCK falls		28.5 + (t_{CLK_SPI})/2		ns
SPI4	MISO hold time after SPCK falls		0		ns
SPI5	SPCK falling to MOSI delay			10.5	ns

Note: 1. These values are based on simulation and characterization of other AVR microcontrollers manufactured in the same process technology. These values are not covered by test limits in production.

Maximum SPI Frequency, Master Output

The maximum SPI master output frequency is given by the following formula:

$$f_{SPCKMAX} = MIN(f_{PINMAX}, \frac{1}{SPIn})$$

Where $SPIn$ is the MOSI delay, SPI2 or SPI5 depending on CPOL and NCPHA. f_{PINMAX} is the maximum frequency of the SPI pins. Please refer to the I/O Pin Characteristics section for the maximum frequency of the pins.

Maximum SPI Frequency, Master Input

The maximum SPI master input frequency is given by the following formula:

$$f_{SPCKMAX} = \frac{1}{SPIn + t_{VALID}}$$

Where $SPIn$ is the MISO setup and hold time, SPI0 + SPI1 or SPI3 + SPI4 depending on CPOL and NCPHA. t_{VALID} is the SPI slave response time. Please refer to the SPI slave datasheet for t_{VALID} .

7.9.6 JTAG Timing

Figure 7-16. JTAG Interface Signals

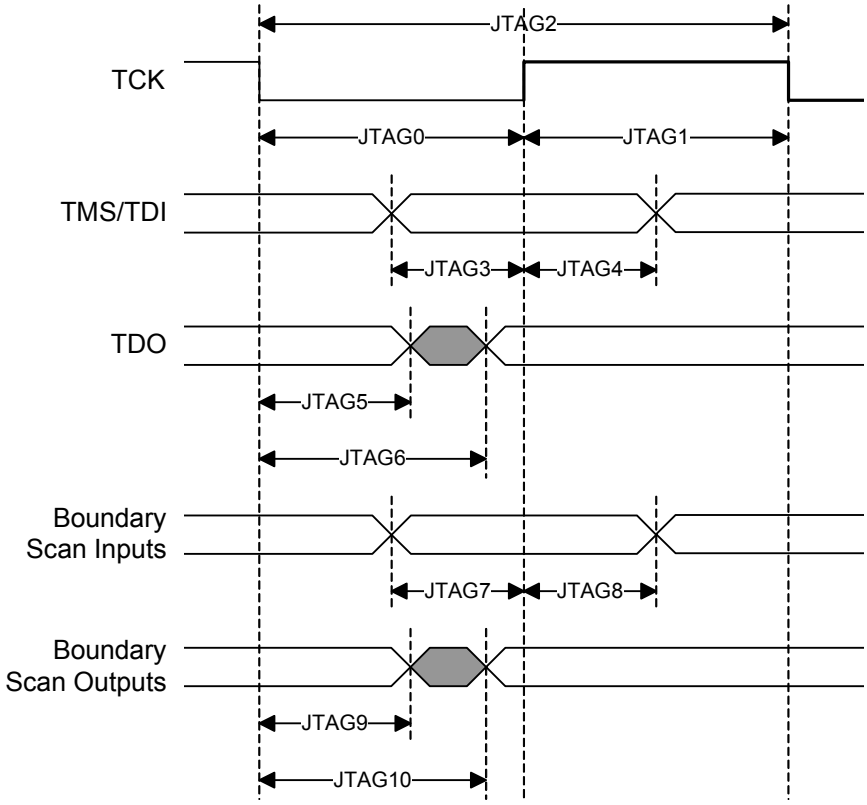


Table 7-51. JTAG Timings[1]

Symbol	Parameter	Conditions	Min	Max	Units
JTAG0	TCK Low Half-period	external capacitor = 40pF	21.5		ns
JTAG1	TCK High Half-period		8.5		ns
JTAG2	TCK Period		29		ns
JTAG3	TDI, TMS Setup before TCK High		6.5		ns
JTAG4	TDI, TMS Hold after TCK High		0		ns
JTAG5	TDO Hold Time		12.5		ns
JTAG6	TCK Low to TDO Valid			21.5	ns
JTAG7	Boundary Scan Inputs Setup Time		0		ns
JTAG8	Boundary Scan Inputs Hold Time		4.5		ns
JTAG9	Boundary Scan Outputs Hold Time		11		ns
JTAG10	TCK to Boundary Scan Outputs Valid			18	ns

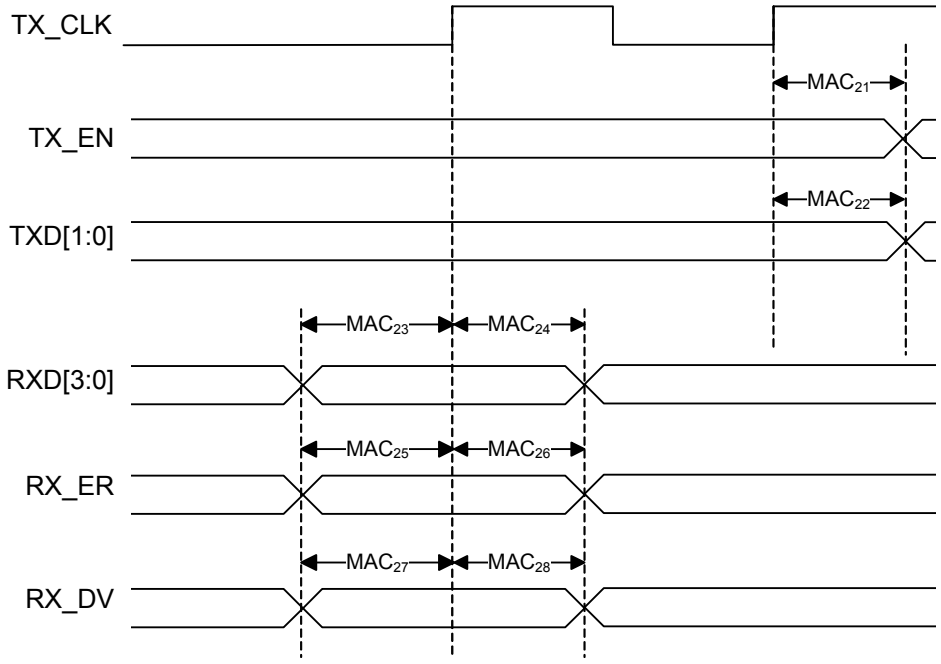
Note: 1. These values are based on simulation and characterization of other AVR microcontrollers manufactured in the same process technology. These values are not covered by test limits in production.

Table 7-61. Ethernet MAC RMII Specific Signals[1]

Symbol	Parameter	Conditions	Min.	Max.	Unit
MAC_{21}	TX_EN toggling from TX_CLK rising	V_{VDD} = 3.0V, drive strength of the pads set to the highest, external capacitor = 10pF on MACB pins	11.7	12.5	ns
MAC_{22}	TXD toggling from TX_CLK rising		11.7	12.5	ns
MAC_{23}	Setup for RXD from TX_CLK		4.5		ns
MAC_{24}	Hold for RXD from TX_CLK		0		ns
MAC_{25}	Setup for RX_ER from TX_CLK		3.4		ns
MAC_{26}	Hold for RX_ER from TX_CLK		0		ns
MAC_{27}	Setup for RX_DV from TX_CLK		4.4		ns
MAC_{28}	Hold for RX_DV from TX_CLK		0		ns

Note: 1. These values are based on simulation and characterization of other AVR microcontrollers manufactured in the same process technology. These values are not covered by test limits in production.

Figure 7-21. Ethernet MAC RMII Mode



8. Mechanical Characteristics

8.1 Thermal Considerations

8.1.1 Thermal Data

Table 8-1 summarizes the thermal resistance data depending on the package.

Table 8-1. Thermal Resistance Data

Symbol	Parameter	Condition	Package	Typ	Unit
θ_{JA}	Junction-to-ambient thermal resistance	No air flow	QFN64	20.0	°C/W
θ_{JC}	Junction-to-case thermal resistance		QFN64	0.8	
θ_{JA}	Junction-to-ambient thermal resistance	No air flow	TQFP64	40.5	°C/W
θ_{JC}	Junction-to-case thermal resistance		TQFP64	8.7	
θ_{JA}	Junction-to-ambient thermal resistance	No air flow	TQFP100	39.3	°C/W
θ_{JC}	Junction-to-case thermal resistance		TQFP100	8.5	
θ_{JA}	Junction-to-ambient thermal resistance	No air flow	LQFP144	38.1	°C/W
θ_{JC}	Junction-to-case thermal resistance		LQFP144	8.4	

8.1.2 Junction Temperature

The average chip-junction temperature, T_J, in °C can be obtained from the following:

1. $T_J = T_A + (P_D \times \theta_{JA})$
2. $T_J = T_A + (P_D \times (\theta_{HEATSINK} + \theta_{JC}))$

where:

- θ_{JA} = package thermal resistance, Junction-to-ambient (°C/W), provided in Table 8-1 on page 90.
- θ_{JC} = package thermal resistance, Junction-to-case thermal resistance (°C/W), provided in Table 8-1 on page 90.
- $\theta_{HEAT\ SINK}$ = cooling device thermal resistance (°C/W), provided in the device datasheet.
- P_D = device power consumption (W) estimated from data provided in the section "Power Consumption" on page 51.
- T_A = ambient temperature (°C).

From the first equation, the user can derive the estimated lifetime of the chip and decide if a cooling device is necessary or not. If a cooling device is to be fitted on the chip, the second equation should be used to compute the resulting average chip-junction temperature T_J in °C.

10. Errata

10.1 rev E

10.1.1 ADCIFA

1 **ADCREFP/ADCREFN can not be selected as an external ADC reference by setting the ADCIFA.CFG.EXREF bit to one**
Fix/Workaround
A voltage reference can be applied on ADCREFP/ADCREFN pins if the ADCIFA.CFG.EXREF bit is set to zero, the ADCIFA.CFG.RS bit is set to zero and the voltage reference applied on ADCREFP/ADCREFN pins is higher than the internal 1V reference.

10.1.2 AST

1 **AST wake signal is released one AST clock cycle after the BUSY bit is cleared**
After writing to the Status Clear Register (SCR) the wake signal is released one AST clock cycle after the BUSY bit in the Status Register (SR.BUSY) is cleared. If entering sleep mode directly after the BUSY bit is cleared the part will wake up immediately.
Fix/Workaround
Read the Wake Enable Register (WER) and write this value back to the same register. Wait for BUSY to clear before entering sleep mode.

10.1.3 aWire

1 **aWire MEMORY_SPEED_REQUEST command does not return correct CV**
The aWire MEMORY_SPEED_REQUEST command does not return a CV corresponding to the formula in the aWire Debug Interface chapter.
Fix/Workaround
Issue a dummy read to address 0x100000000 before issuing the MEMORY_SPEED_REQUEST command and use this formula instead:

$$f_{sab} = \frac{7 f_{aw}}{CV - 3}$$

10.1.4 Power Manager

1 **TWIS may not wake the device from sleep mode**
If the CPU is put to a sleep mode (except Idle and Frozen) directly after a TWI Start condition, the CPU may not wake upon a TWIS address match. The request is NACKed.
Fix/Workaround
When using the TWI address match to wake the device from sleep, do not switch to sleep modes deeper than Frozen. Another solution is to enable asynchronous EIC wake on the TWIS clock (TWCK) or TWIS data (TWD) pins, in order to wake the system up on bus events.

4 SPI bad serial clock generation on 2nd chip_select when SCBR=1, CPOL=1, and NCPHA=0

When multiple chip selects (CS) are in use, if one of the baudrates equal 1 while one (CSRn.SCBR=1) of the others do not equal 1, and CSRn.CPOL=1 and CSRn.NCPHA=0, then an additional pulse will be generated on SCK.

Fix/Workaround

When multiple CS are in use, if one of the baudrates equals 1, the others must also equal 1 if CSRn.CPOL=1 and CSRn.NCPHA=0.

10.1.7 TC

1 Channel chaining skips first pulse for upper channel

When chaining two channels using the Block Mode Register, the first pulse of the clock between the channels is skipped.

Fix/Workaround

Configure the lower channel with RA = 0x1 and RC = 0x2 to produce a dummy clock cycle for the upper channel. After the dummy cycle has been generated, indicated by the SR.CPCS bit, reconfigure the RA and RC registers for the lower channel with the real values.

10.1.8 TWIM

1 SMBALERT bit may be set after reset

For TWIM0 and TWIM1 modules, the SMBus Alert (SMBALERT) bit in the Status Register (SR) might be erroneously set after system reset.

Fix/Workaround

After system reset, clear the SR.SMBALERT bit before commencing any TWI transfer.

For TWIM2 module, the SMBus Alert (SMBALERT) is not implemented but the bit in the Status Register (SR) is erroneously set once TWIM2 is enabled.

Fix/Workaround

None.

10.1.9 TWIS

1 Clearing the NAK bit before the BTF bit is set locks up the TWI bus

When the TWIS is in transmit mode, clearing the NAK Received (NAK) bit of the Status Register (SR) before the end of the Acknowledge/Not Acknowledge cycle will cause the TWIS to attempt to continue transmitting data, thus locking up the bus.

Fix/Workaround

Clear SR.NAK only after the Byte Transfer Finished (BTF) bit of the same register has been set.

10.1.10 USBC

1 UPINRQx.INRQ field is limited to 8-bits

In Host mode, when using the UPINRQx.INRQ feature together with the multi-packet mode to launch a finite number of packet among multi-packet, the multi-packet size (located in the descriptor table) is limited to the UPINRQx.INRQ value multiply by the pipe size.

Fix/Workaround

UPINRQx.INRQ value shall be less than the number of configured multi-packet.

2 In USB host mode, downstream resume feature does not work (UHCON.RESUME=1).

Fix/Workaround
None.

3 **In host mode, the disconnection during OUT transition is not supported**
In USB host mode, a pipe can not work if the previous USB device disconnection has occurred during a USB transfer.
Fix/Workaround
Reset the USBC (USBCON.USB=0 and =1) after a device disconnection (UHINT.DDISCI).

4 **In USB host mode, entering suspend mode can fail**
In USB host mode, entering suspend mode can fail when UHCON.SOFE=0 is done just after a SOF reception (UHINT.HSOFI).
Fix/Workaround
Check that UHNUM.FLENHIGH is below 185 in Full speed and below 21 in Low speed before clearing UHCON.SOFE.

5 **In USB host mode, entering suspend mode for low speed device can fail when the USB freeze (USBCON.FRZCLK=1) is done just after UHCON.SOFE=0.**
Fix/Workaround
When entering suspend mode (UHCON.SOFE is cleared), check that USBFSM.DRDSTATE is not equal to three before freezing the clock (USBCON.FRZCLK=1).

10.1.11 WDT

1 **WDT Control Register does not have synchronization feedback**
When writing to the Timeout Prescale Select (PSEL), Time Ban Prescale Select (TBAN), Enable (EN), or WDT Mode (MODE) fieldss of the WDT Control Register (CTRL), a synchronizer is started to propagate the values to the WDT clcok domain. This synchronization takes a finite amount of time, but only the status of the synchronization of the EN bit is reflected back to the user. Writing to the synchronized fields during synchronization can lead to undefined behavior.
Fix/Workaround
-When writing to the affected fields, the user must ensure a wait corresponding to 2 clock cycles of both the WDT peripheral bus clock and the selected WDT clock source.
-When doing writes that changes the EN bit, the EN bit can be read back until it reflects the written value.

5	AST: Updated digital tuner formula
6	SDRAMC: cleaned-up SDCS/NCS names. Added VERSION register
7	SAU: Updated SR.IDLE
8	USART: Updated
9	CANIF: Updated address map figure
10	USBC: Updated
11	DACIFB: Updated
12	Programming and Debugging: Added JTAG Data Registers section
13	Electrical Characteristics: Updated
14	Ordering Information: Updated
15	Errata: Updated

11.4 Rev. A – 10/10

1	Initial revision