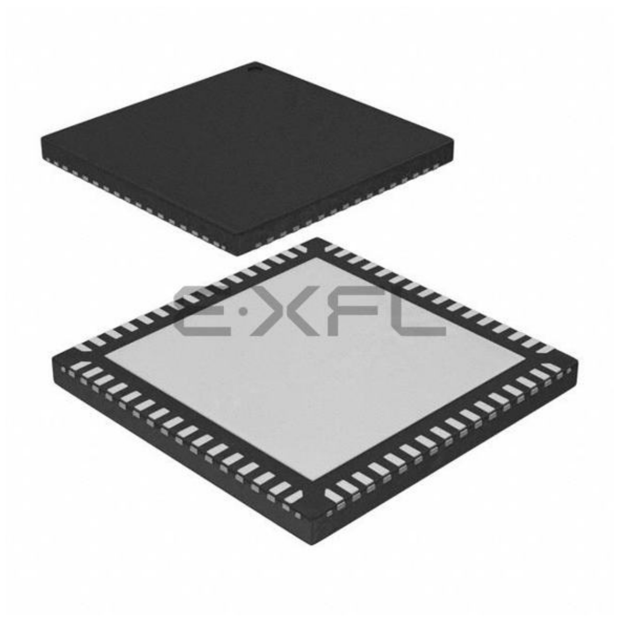
Microchip Technology - AT32UC3B0256-Z2UT Datasheet



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	60MHz
Connectivity	I²C, IrDA, SPI, SSC, UART/USART, USB
Peripherals	Brown-out Detect/Reset, DMA, POR, PWM, WDT
Number of I/O	44
Program Memory Size	256KB (256K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	32K x 8
Voltage - Supply (Vcc/Vdd)	1.65V ~ 3.6V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	64-VFQFN Exposed Pad
Supplier Device Package	64-QFN (9x9)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/at32uc3b0256-z2ut

Table 4-1. GPIO Controller Function Multiplexing

	55	PB09	GPIO 41	SSC - TX_CLOCK	USART1 - RI	EIC - SCAN[7]	ABDAC - DATAN[1]
	57	PB10	GPIO 42	SSC - TX_DATA	TC - A2	USART0 - RXD	
	58	PB11	GPIO 43	SSC - TX_FRAME_SYNC	TC - B2	USART0 - TXD	

4.2.2 JTAG Port Connections

If the JTAG is enabled, the JTAG will take control over a number of pins, irrespective of the I/O Controller configuration.

Table 4-2. JTAG Pinout

64QFP/QFN	48QFP/QFN	Pin name	JTAG pin
2	2	TCK	TCK
3	3	PA00	TDI
4	4	PA01	TDO
5	5	PA02	TMS

4.2.3 Nexus OCD AUX port connections

If the OCD trace system is enabled, the trace system will take control over a number of pins, irrespectively of the PIO configuration. Two different OCD trace pin mappings are possible, depending on the configuration of the OCD AXS register. For details, see the AVR32 UC Technical Reference Manual.

Table 4-3. Nexus OCD AUX port connections

Pin	AXS=0	AXS=1
EVTI_N	PB05	PA14
MDO[5]	PB04	PA08
MDO[4]	PB03	PA07
MDO[3]	PB02	PA06
MDO[2]	PB01	PA05
MDO[1]	PB00	PA04
MDO[0]	PA31	PA03
EVTO_N	PA15	PA15
MCKO	PA30	PA13
MSEO[1]	PB06	PA09
MSEO[0]	PB07	PA10

4.2.4 Oscillator Pinout

The oscillators are not mapped to the normal A, B or C functions and their muxings are controlled by registers in the Power Manager (PM). Please refer to the power manager chapter for more information about this.

Table 5-1. Signal Description List (Continued)

Signal Name	Function	Type	Active Level	Comments
Serial Peripheral Interface - SPI0				
MISO	Master In Slave Out	I/O		
MOSI	Master Out Slave In	I/O		
NPCS0 - NPCS3	SPI Peripheral Chip Select	I/O	Low	
SCK	Clock	Output		
Synchronous Serial Controller - SSC				
RX_CLOCK	SSC Receive Clock	I/O		
RX_DATA	SSC Receive Data	Input		
RX_FRAME_SYNC	SSC Receive Frame Sync	I/O		
TX_CLOCK	SSC Transmit Clock	I/O		
TX_DATA	SSC Transmit Data	Output		
TX_FRAME_SYNC	SSC Transmit Frame Sync	I/O		
Timer/Counter - TIMER				
A0	Channel 0 Line A	I/O		
A1	Channel 1 Line A	I/O		
A2	Channel 2 Line A	I/O		
B0	Channel 0 Line B	I/O		
B1	Channel 1 Line B	I/O		
B2	Channel 2 Line B	I/O		
CLK0	Channel 0 External Clock Input	Input		
CLK1	Channel 1 External Clock Input	Input		
CLK2	Channel 2 External Clock Input	Input		
Two-wire Interface - TWI				
SCL	Serial Clock	I/O		
SDA	Serial Data	I/O		
Universal Synchronous Asynchronous Receiver Transmitter - USART0, USART1, USART2				
CLK	Clock	I/O		
CTS	Clear To Send	Input		

The register file is organized as sixteen 32-bit registers and includes the Program Counter, the Link Register, and the Stack Pointer. In addition, register R12 is designed to hold return values from function calls and is used implicitly by some instructions.

6.3 The AVR32UC CPU

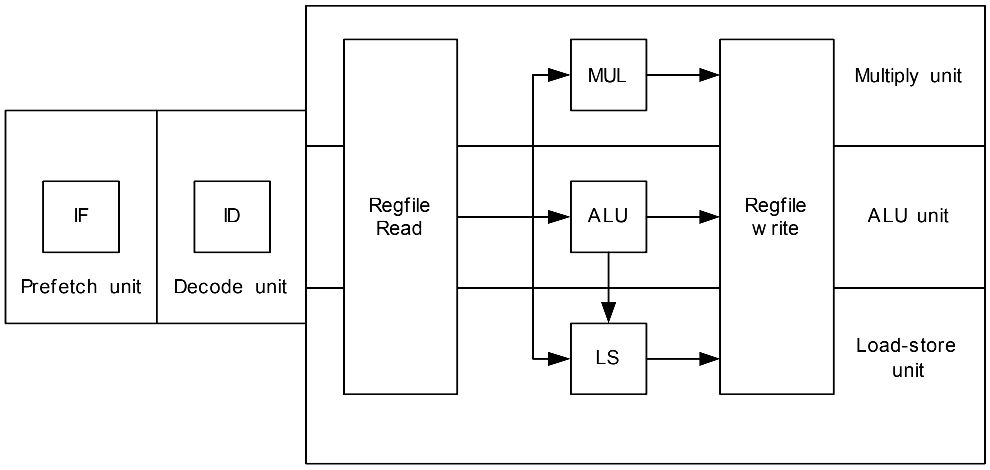
The AVR32UC CPU targets low- and medium-performance applications, and provides an advanced OCD system, no caches, and a Memory Protection Unit (MPU). Java acceleration hardware is not implemented.

AVR32UC provides three memory interfaces, one High Speed Bus master for instruction fetch, one High Speed Bus master for data access, and one High Speed Bus slave interface allowing other bus masters to access data RAMs internal to the CPU. Keeping data RAMs internal to the CPU allows fast access to the RAMs, reduces latency, and guarantees deterministic timing. Also, power consumption is reduced by not needing a full High Speed Bus access for memory accesses. A dedicated data RAM interface is provided for communicating with the internal data RAMs.

A local bus interface is provided for connecting the CPU to device-specific high-speed systems, such as floating-point units and fast GPIO ports. This local bus has to be enabled by writing the LOCEN bit in the CPUCR system register. The local bus is able to transfer data between the CPU and the local bus slave in a single clock cycle. The local bus has a dedicated memory range allocated to it, and data transfers are performed using regular load and store instructions. Details on which devices that are mapped into the local bus space is given in the Memories chapter of this data sheet.

Figure 6-1 on page 19 displays the contents of AVR32UC.

Figure 6-2. The AVR32UC Pipeline



6.3.2 AVR32A Microarchitecture Compliance

AVR32UC implements an AVR32A microarchitecture. The AVR32A microarchitecture is targeted at cost-sensitive, lower-end applications like smaller microcontrollers. This microarchitecture does not provide dedicated hardware registers for shadowing of register file registers in interrupt contexts. Additionally, it does not provide hardware registers for the return address registers and return status registers. Instead, all this information is stored on the system stack. This saves chip area at the expense of slower interrupt handling.

Upon interrupt initiation, registers R8-R12 are automatically pushed to the system stack. These registers are pushed regardless of the priority level of the pending interrupt. The return address and status register are also automatically pushed to stack. The interrupt handler can therefore use R8-R12 freely. Upon interrupt completion, the old R8-R12 registers and status register are restored, and execution continues at the return address stored popped from stack.

The stack is also used to store the status register and return address for exceptions and *scall*. Executing the *rete* or *rets* instruction at the completion of an exception or system call will pop this status register and continue execution at the popped return address.

6.3.3 Java Support

AVR32UC does not provide Java hardware acceleration.

6.3.4 Memory Protection

The MPU allows the user to check all memory accesses for privilege violations. If an access is attempted to an illegal memory address, the access is aborted and an exception is taken. The MPU in AVR32UC is specified in the AVR32UC Technical Reference manual.

6.3.5 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.

6.6 Module Configuration

All AT32UC3B parts do not implement the same CPU and Architecture Revision.

Table 6-5. CPU and Architecture Revision

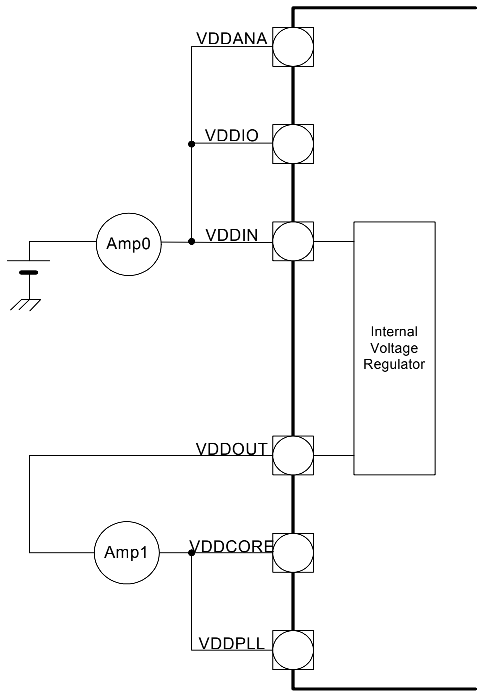
Part Name	Architecture Revision
AT32UC3Bx512	2
AT32UC3Bx256	1
AT32UC3Bx128	1
AT32UC3Bx64	1

9.5 Power Consumption

The values in Table 9-10, Table 9-11 on page 43 and Table 9-12 on page 44 are measured values of power consumption with operating conditions as follows:

- $V_{DDIO} = V_{DDANA} = 3.3V$
- $V_{DDCORE} = V_{DDPLL} = 1.8V$
- $T_A = 25°C$, $T_A = 85°C$
- I/Os are configured in input, pull-up enabled.

Figure 9-5. Measurement Setup



The following tables represent the power consumption measured on the power supplies.

9.6 System Clock Characteristics

These parameters are given in the following conditions:

- V_{DDCORE} = 1.8V
- Ambient Temperature = 25°C

9.6.1 CPU/HSB Clock Characteristics

Table 9-13. Core Clock Waveform Parameters

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Unit
$1/(t_{CPCPU})$	CPU Clock Frequency				60	MHz
t_{CPCPU}	CPU Clock Period		16.6			ns

9.6.2 PBA Clock Characteristics

Table 9-14. PBA Clock Waveform Parameters

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Unit
$1/(t_{CPPBA})$	PBA Clock Frequency				60	MHz
t_{CPPBA}	PBA Clock Period		16.6			ns

9.6.3 PBB Clock Characteristics

Table 9-15. PBB Clock Waveform Parameters

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Unit
$1/(t_{CPPBB})$	PBB Clock Frequency				60	MHz
t_{CPPBB}	PBB Clock Period		16.6			ns

Table 9-23. Transfer Characteristics in 8-bit Mode

Parameter	Conditions	Min.	Typ.	Max.	Unit
Differential Non-linearity	ADC Clock = 5 MHz		0.3	0.5	LSB
	ADC Clock = 8 MHz		0.5	1.0	LSB
Offset Error	ADC Clock = 5 MHz	-0.5		0.5	LSB
Gain Error	ADC Clock = 5 MHz	-0.5		0.5	LSB

Table 9-24. Transfer Characteristics in 10-bit Mode

Parameter	Conditions	Min.	Typ.	Max.	Unit
Resolution			10		Bit
Absolute Accuracy	ADC Clock = 5 MHz			3	LSB
Integral Non-linearity	ADC Clock = 5 MHz		1.5	2	LSB
Differential Non-linearity	ADC Clock = 5 MHz		1	2	LSB
	ADC Clock = 2.5 MHz		0.6	1	LSB
Offset Error	ADC Clock = 5 MHz	-2		2	LSB
Gain Error	ADC Clock = 5MHz	-2		2	LSB

9.10 JTAG Characteristics

9.10.1 JTAG Timing

Figure 9-6. JTAG Interface Signals

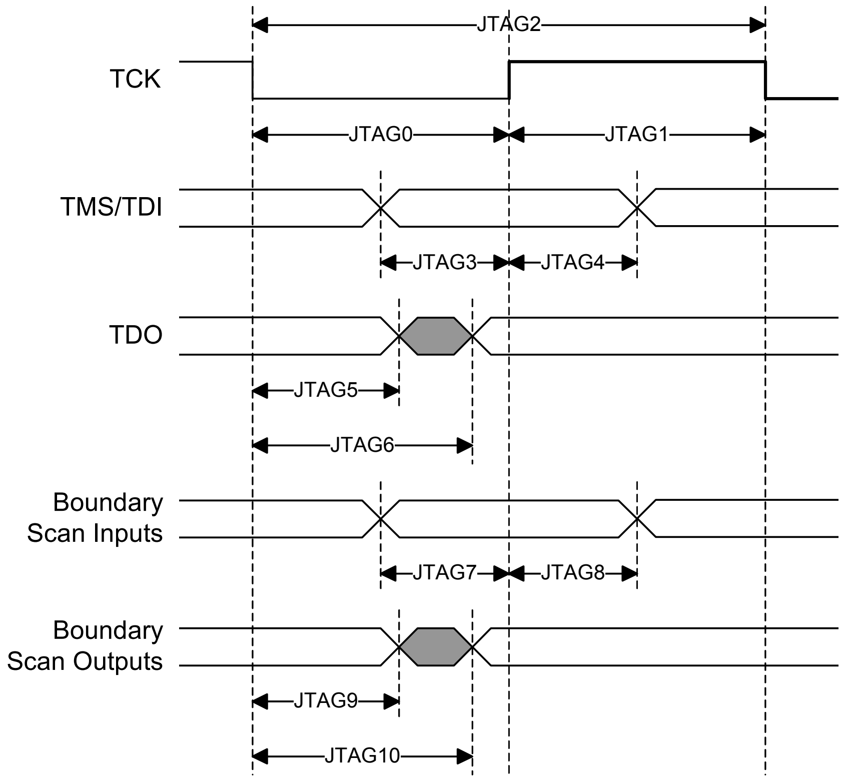


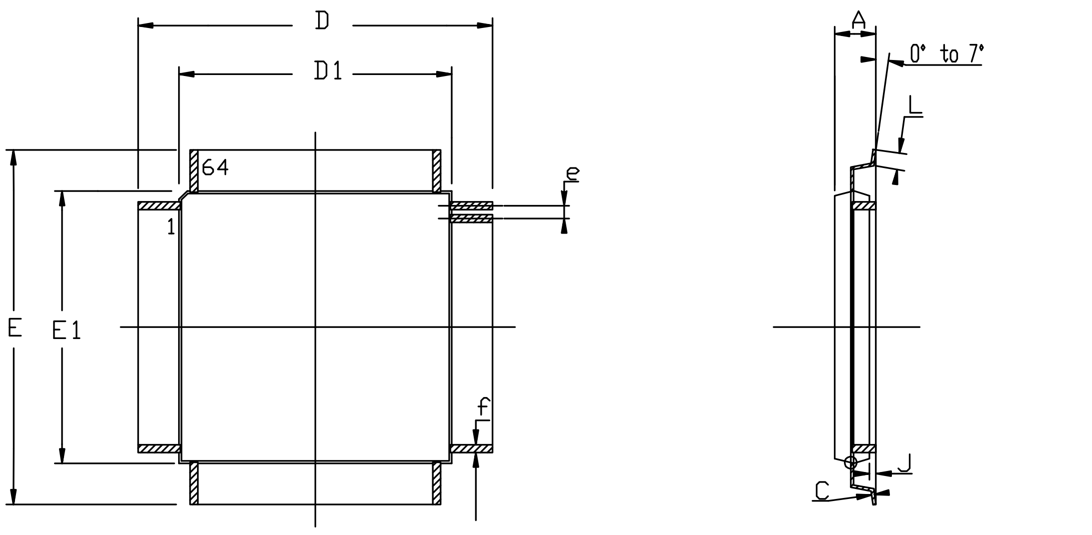
Table 9-26. JTAG Timings[(1)]

Symbol	Parameter	Conditions	Min	Max	Units
JTAG0	TCK Low Half-period	V_{VDDIO} from 3.0V to 3.6V, maximum external capacitor = 40pF	23.2		ns
JTAG1	TCK High Half-period		8.8		ns
JTAG2	TCK Period		32.0		ns
JTAG3	TDI, TMS Setup before TCK High		3.9		ns
JTAG4	TDI, TMS Hold after TCK High		0.6		ns
JTAG5	TDO Hold Time		4.5		ns
JTAG6	TCK Low to TDO Valid			23.2	ns
JTAG7	Boundary Scan Inputs Setup Time		0		ns
JTAG8	Boundary Scan Inputs Hold Time		5.0		ns
JTAG9	Boundary Scan Outputs Hold Time		8.7		ns
JTAG10	TCK to Boundary Scan Outputs Valid			17.7	ns

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

10.2 Package Drawings

Figure 10-1. TQFP-64 package drawing



COMMON DIMENSIONS IN MM

SYMBOL	Min	Max	NOTES
A	----	1.20	
A1	0.95	1.05	
C	0.09	0.20	
D	12.00 BSC		
D1	10.00 BSC		
E	12.00 BSC		
E1	10.00 BSC		
J	0.05	0.15	
L	0.45	0.75	
e	0.50 BSC		
f	0.17	0.27	

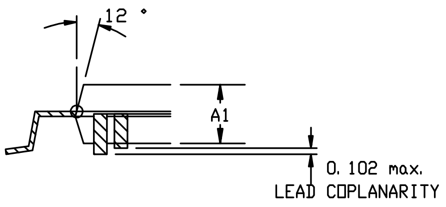


Table 10-2. Device and Package Maximum Weight

Weight	300 mg

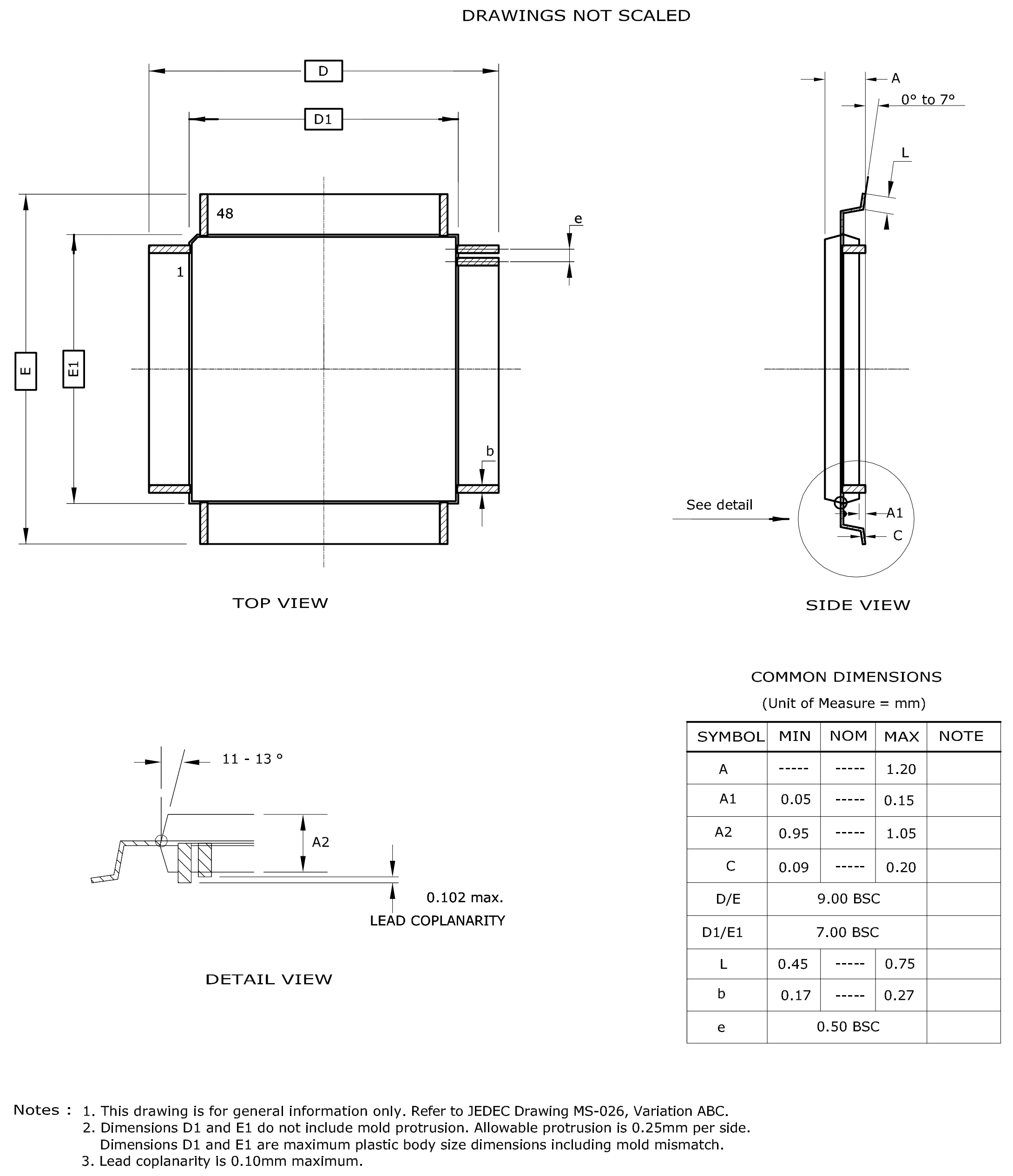
Table 10-3. Package Characteristics

Moisture Sensitivity Level	Jedec J-STD-20D-MSL3

Table 10-4. Package Reference

JEDEC Drawing Reference	MS-026
JESD97 Classification	e3

Figure 10-2. TQFP-48 package drawing



SYMBOL	MIN	NOM	MAX	NOTE
A	-----	-----	1.20	
A1	0.05	-----	0.15	
A2	0.95	-----	1.05	
C	0.09	-----	0.20	
D/E		9.00 BSC		
D1/E1		7.00 BSC		
L	0.45	-----	0.75	
b	0.17	-----	0.27	
e		0.50 BSC		

Notes :
1. This drawing is for general information only. Refer to JEDEC Drawing MS-026, Variation ABC.
2. Dimensions D1 and E1 do not include mold protrusion. Allowable protrusion is 0.25mm per side. Dimensions D1 and E1 are maximum plastic body size dimensions including mold mismatch.
3. Lead coplanarity is 0.10mm maximum.

Table 10-5. Device and Package Maximum Weight

Weight	100 mg

Table 10-6. Package Characteristics

Moisture Sensitivity Level	Jedec J-STD-20D-MSL3

Table 10-7. Package Reference

JEDEC Drawing Reference	MS-026
JESD97 Classification	e3

- *Processor and Architecture*

1. **LDM instruction with PC in the register list and without ++ increments Rp**
For LDM with PC in the register list: the instruction behaves as if the ++ field is always set, ie the pointer is always updated. This happens even if the ++ field is cleared. Specifically, the increment of the pointer is done in parallel with the testing of R12.
Fix/Workaround
None.

2. **RETE instruction does not clear SREG[L] from interrupts**
The RETE instruction clears SREG[L] as expected from exceptions.
Fix/Workaround
When using the STCOND instruction, clear SREG[L] in the stacked value of SR before returning from interrupts with RETE.

3. **Privilege violation when using interrupts in application mode with protected system stack**
If the system stack is protected by the MPU and an interrupt occurs in application mode, an MPU DTLB exception will occur.
Fix/Workaround
Make a DTLB Protection (Write) exception handler which permits the interrupt request to be handled in privileged mode.

4. **Flash**

5. **Reset vector is 80000020h rather than 80000000h**
Reset vector is 80000020h rather than 80000000h.
Fix/Workaround
The flash program code must start at the address 80000020h. The flash memory range 80000000h-80000020h must be programmed with 00000000h.

- *USART*

1. **ISO7816 info register US_NER cannot be read**
The NER register always returns zero.
Fix/Workaround
None.

2. **ISO7816 Mode T1: RX impossible after any TX**
RX impossible after any TX.
Fix/Workaround
SOFT_RESET on RX+ Config US_MR + Config_US_CR.

3. **The RTS output does not function correctly in hardware handshaking mode**
The RTS signal is not generated properly when the USART receives data in hardware handshaking mode. When the Peripheral DMA receive buffer becomes full, the RTS output should go high, but it will stay low.
Fix/Workaround
Do not use the hardware handshaking mode of the USART. If it is necessary to drive the RTS output high when the Peripheral DMA receive buffer becomes full, use the normal mode of the USART. Configure the Peripheral DMA Controller to signal an interrupt when the receive buffer is full. In the interrupt handler code, write a one to the RTSDIS bit in the USART Control Register (CR). This will drive the RTS output high. After the next DMA trans-

12.2.2 Rev. G

- *PWM*

1. **PWM channel interrupt enabling triggers an interrupt**
When enabling a PWM channel that is configured with center aligned period (CALG=1), an interrupt is signalled.
Fix/Workaround
When using center aligned mode, enable the channel and read the status before channel interrupt is enabled.

2. **PWN counter restarts at 0x0001**
The PWM counter restarts at 0x0001 and not 0x0000 as specified. Because of this the first PWM period has one more clock cycle.
Fix/Workaround
- The first period is 0x0000, 0x0001, ..., period.
- Consecutive periods are 0x0001, 0x0002, ..., period.

3. **PWM update period to a 0 value does not work**
It is impossible to update a period equal to 0 by the using the PWM update register (PWM_CUPD).
Fix/Workaround
Do not update the PWM_CUPD register with a value equal to 0.

4. **SPI**

5. **SPI Slave / PDCA transfer: no TX UNDERRUN flag**
There is no TX UNDERRUN flag available, therefore in SPI slave mode, there is no way to be informed of a character lost in transmission.
Fix/Workaround
For PDCA transfer: none.

6. **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.

7. **SPI Glitch on RXREADY flag in slave mode when enabling the SPI or during the first transfer**
In slave mode, the SPI can generate a false RXREADY signal during enabling of the SPI or during the first transfer.
Fix/Workaround
1. Set slave mode, set required CPOL/CPHA.
2. Enable SPI.
3. Set the polarity CPOL of the line in the opposite value of the required one.
4. Set the polarity CPOL to the required one.
5. Read the RXHOLDING register.
Transfers can now begin and RXREADY will now behave as expected.

15. **SSC**

16. **Additional delay on TD output**
    A delay from 2 to 3 system clock cycles is added to TD output when:
    TCMR.START = Receive Start,
    TCMR.STTDLY = more than ZERO,
    RCMR.START = Start on falling edge / Start on Rising edge / Start on any edge,
    RFMR.FSOS = None (input).
    Fix/Workaround
    None.

17. **TF output is not correct**
    TF output is not correct (at least emitted one serial clock cycle later than expected) when:
    TFMR.FSOS = Driven Low during data transfer/ Driven High during data transfer
    TCMR.START = Receive start
    RFMR.FSOS = None (Input)
    RCMR.START = any on RF (edge/level)
    Fix/Workaround
    None.

18. **Frame Synchro and Frame Synchro Data are delayed by one clock cycle**
    The frame synchro and the frame synchro data are delayed from 1 SSC_CLOCK when:
    - Clock is CKDIV
    - The START is selected on either a frame synchro edge or a level
    - Frame synchro data is enabled
    - Transmit clock is gated on output (through CKO field)
    Fix/Workaround
    Transmit or receive CLOCK must not be gated (by the mean of CKO field) when START condition is performed on a generated frame synchro.

19. **USB**

20. **UPCFGn.INTFRQ is irrelevant for isochronous pipe**
    As a consequence, isochronous IN and OUT tokens are sent every 1ms (Full Speed), or every 125uS (High Speed).
    Fix/Workaround
    For higher polling time, the software must freeze the pipe for the desired period in order to prevent any "extra" token.

- *ADC*

1. **Sleep Mode activation needs additional A to D conversion**
   If the ADC sleep mode is activated when the ADC is idle the ADC will not enter sleep mode before after the next AD conversion.
   Fix/Workaround
   Activate the sleep mode in the mode register and then perform an AD conversion.

- *PDCA*

1. **Wrong PDCA behavior when using two PDCA channels with the same PID**
   Wrong PDCA behavior when using two PDCA channels with the same PID.
   Fix/Workaround
   The same PID should not be assigned to more than one channel.

- *OCD*

1. **The auxiliary trace does not work for CPU/HSB speed higher than 50MHz**
   The auxiliary trace does not work for CPU/HSB speed higher than 50MHz.
   Fix/Workaround
   Do not use the auxiliary trace for CPU/HSB speed higher than 50MHz.

- *Processor and Architecture*

1. **LDM instruction with PC in the register list and without ++ increments Rp**
   For LDM with PC in the register list: the instruction behaves as if the ++ field is always set, ie the pointer is always updated. This happens even if the ++ field is cleared. Specifically, the increment of the pointer is done in parallel with the testing of R12.
   Fix/Workaround
   None.

2. **RETE instruction does not clear SREG[L] from interrupts**
   The RETE instruction clears SREG[L] as expected from exceptions.
   Fix/Workaround
   When using the STCOND instruction, clear SREG[L] in the stacked value of SR before returning from interrupts with RETE.

3. **RETS behaves incorrectly when MPU is enabled**
   RETS behaves incorrectly when MPU is enabled and MPU is configured so that system stack is not readable in unprivileged mode.
   Fix/Workaround
   Make system stack readable in unprivileged mode, or return from supervisor mode using rete instead of rets. This requires:
   1. Changing the mode bits from 001 to 110 before issuing the instruction. Updating the mode bits to the desired value must be done using a single mtsr instruction so it is done atomically. Even if this step is generally described as not safe in the UC technical reference manual, it is safe in this very specific case.
   2. Execute the RETE instruction.

4. **Privilege violation when using interrupts in application mode with protected system stack**
   If the system stack is protected by the MPU and an interrupt occurs in application mode, an MPU DTLB exception will occur.
   Fix/Workaround
   Make a DTLB Protection (Write) exception handler which permits the interrupt request to be handled in privileged mode.

5. **USART**

6. **ISO7816 info register US_NER cannot be read**
   The NER register always returns zero.
   Fix/Workaround
   None.

7. **ISO7816 Mode T1: RX impossible after any TX**
   RX impossible after any TX.
   Fix/Workaround
   SOFT_RESET on RX+ Config US_MR + Config_US_CR.

- *DSP Operations*

1. **Hardware breakpoints may corrupt MAC results**
   Hardware breakpoints on MAC instructions may corrupt the destination register of the MAC instruction.
   Fix/Workaround
   Place breakpoints on earlier or later instructions.

2. **The command Quick Page Read User Page(QPRUP) is not functional**
   The command Quick Page Read User Page(QPRUP) is not functional.
   Fix/Workaround
   None.

3. **PAGEN Semantic Field for Program GP Fuse Byte is WriteData[7:0], ByteAddress[1:0] on revision B instead of WriteData[7:0], ByteAddress[2:0]**
   PAGEN Semantic Field for Program GP Fuse Byte is WriteData[7:0], ByteAddress[1:0] on revision B instead of WriteData[7:0], ByteAddress[2:0].
   Fix/Workaround
   None.

4. **Reading from on-chip flash may fail after a flash fuse write operation (FLASHC LP, UP, WGPB, EGPB, SSB, PGPFB, EAGPF commands).**
   After a flash fuse write operation (FLASHC LP, UP, WGPB, EGPB, SSB, PGPFB, EAGPF commands), the following flash read access may return corrupted data. This erratum does not affect write operations to regular flash memory.
   Fix/Workaround
   The flash fuse write operation (FLASHC LP, UP, WGPB, EGPB, SSB, PGPFB, EAGPF commands) must be issued from internal RAM. After the write operation, perform a dummy flash page write operation (FLASHC WP). Content and location of this page is not important and filling the write buffer with all one (FFh) will leave the current flash content unchanged. It is then safe to read and fetch code from the flash.

5.

- *RTC*

1. **Writes to control (CTRL), top (TOP) and value (VAL) in the RTC are discarded if the RTC peripheral bus clock (PBA) is divided by a factor of four or more relative to the HSB clock**
   Writes to control (CTRL), top (TOP) and value (VAL) in the RTC are discarded if the RTC peripheral bus clock (PBA) is divided by a factor of four or more relative to the HSB clock.
   Fix/Workaround
   Do not write to the RTC registers using the peripheral bus clock (PBA) divided by a factor of four or more relative to the HSB clock.

2. **The RTC CLKEN bit (bit number 16) of CTRL register is not available**
   The RTC CLKEN bit (bit number 16) of CTRL register is not available.
   Fix/Workaround
   Do not use the CLKEN bit of the RTC on Rev B.

Figure 12-1. Timer/Counter clock connections on RevB

Source	Name	Connection
Internal	TIMER_CLOCK1	32KHz Oscillator
	TIMER_CLOCK2	PBA Clock / 4
	TIMER_CLOCK3	PBA Clock / 8
	TIMER_CLOCK4	PBA Clock / 16
	TIMER_CLOCK5	PBA Clock / 32
External	XC0	
	XC1	
	XC2	

7. **Spurious interrupt may corrupt core SR mode to exception**
   If the rules listed in the chapter `Masking interrupt requests in peripheral modules' of the AVR32UC Technical Reference Manual are not followed, a spurious interrupt may occur. An interrupt context will be pushed onto the stack while the core SR mode will indicate an exception. A RETE instruction would then corrupt the stack.
   Fix/Workaround
   Follow the rules of the AVR32UC Technical Reference Manual. To increase software robustness, if an exception mode is detected at the beginning of an interrupt handler, change the stack interrupt context to an exception context and issue a RETE instruction.

8. **CPU cannot operate on a divided slow clock (internal RC oscillator)**
   CPU cannot operate on a divided slow clock (internal RC oscillator).
   Fix/Workaround
   Do not run the CPU on a divided slow clock.

9. **LDM instruction with PC in the register list and without ++ increments Rp**
   For LDM with PC in the register list: the instruction behaves as if the ++ field is always set, i.e. the pointer is always updated. This happens even if the ++ field is cleared. Specifically, the increment of the pointer is done in parallel with the testing of R12.
   Fix/Workaround
   None.

10. **RETE instruction does not clear SREG[L] from interrupts**
   The RETE instruction clears SREG[L] as expected from exceptions.
   Fix/Workaround
   When using the STCOND instruction, clear SREG[L] in the stacked value of SR before returning from interrupts with RETE.

11. **Exceptions when system stack is protected by MPU**
   RETS behaves incorrectly when MPU is enabled and MPU is configured so that system stack is not readable in unprivileged mode.
   Fix/Workaround
   Workaround 1: Make system stack readable in unprivileged mode,
   or
   Workaround 2: Return from supervisor mode using rete instead of rets. This requires: 1. Changing the mode bits from 001b to 110b before issuing the instruction.
   Updating the mode bits to the desired value must be done using a single mtsr instruction so

it is done atomically. Even if this step is described in general as not safe in the UC technical reference guide, it is safe in this very specific case.
2. Execute the RETE instruction.

Atmel Corporation
2325 Orchard Parkway
San Jose, CA 95131
USA
Tel: (+1)(408) 441-0311
Fax: (+1)(408) 487-2600
www.atmel.com

Atmel Asia Limited
Unit 1-5 & 16, 19/F
BEA Tower, Millennium City 5
418 Kwun Tong Road
Kwun Tong, Kowloon
HONG KONG
Tel: (+852) 2245-6100
Fax: (+852) 2722-1369

Atmel Munich GmbH
Business Campus
Parkring 4
D-85748 Garching b. Munich
GERMANY
Tel: (+49) 89-31970-0
Fax: (+49) 89-3194621

Atmel Japan
16F, Shin Osaki Kangyo Bldg.
1-6-4 Osaka Shinagawa-ku
Tokyo 104-0032
JAPAN
Tel: (+81) 3-6417-0300
Fax: (+81) 3-6417-0370