Microchip Technology - AT32UC3B1256-AUT Datasheet

What is "Embedded - Microcontrollers"?

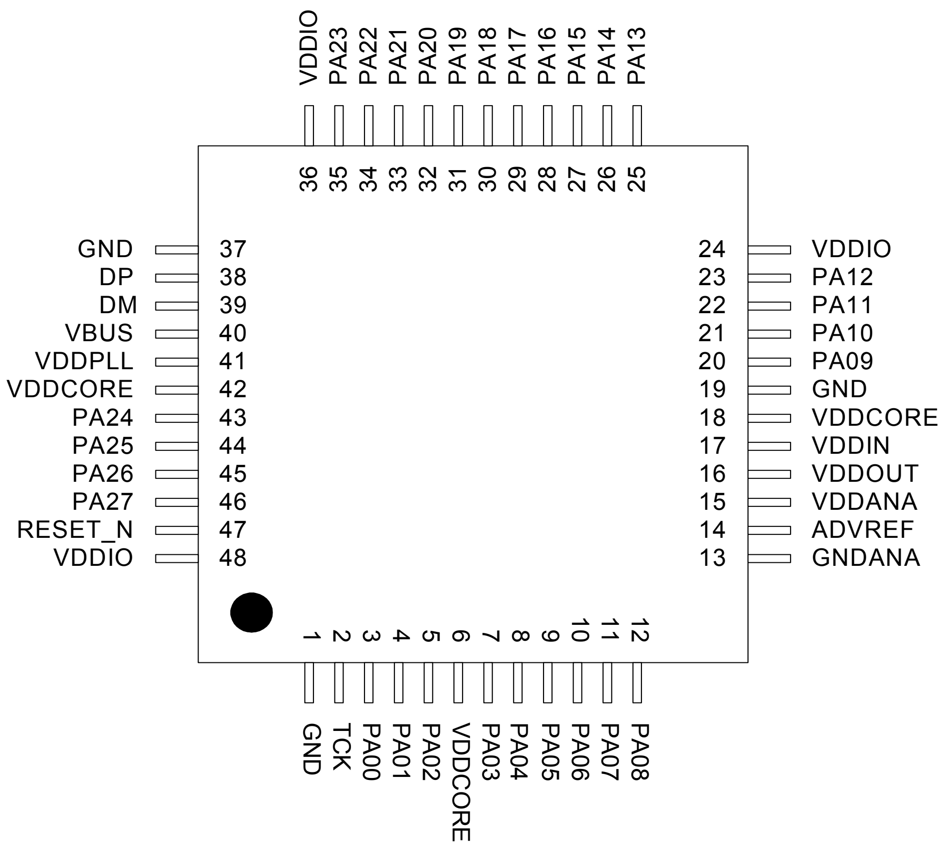
"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" 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	28
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 6x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	48-TQFP
Supplier Device Package	48-TQFP (7x7)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/at32uc3b1256-aut

- **On-Chip Debug System (JTAG interface)**
  - **Nexus Class 2+, Runtime Control, Non-Intrusive Data and Program Trace**
- **64-pin TQFP/QFN (44 GPIO pins), 48-pin TQFP/QFN (28 GPIO pins)**
- **5V Input Tolerant I/Os, including 4 high-drive pins**
- **Single 3.3V Power Supply or Dual 1.8V-3.3V Power Supply**

Figure 4-2. TQFP48 / QFN48 Pinout



Note: The exposed pad is not connected to anything internally, but should be soldered to ground to increase board level reliability.

4.2 Peripheral Multiplexing on I/O lines

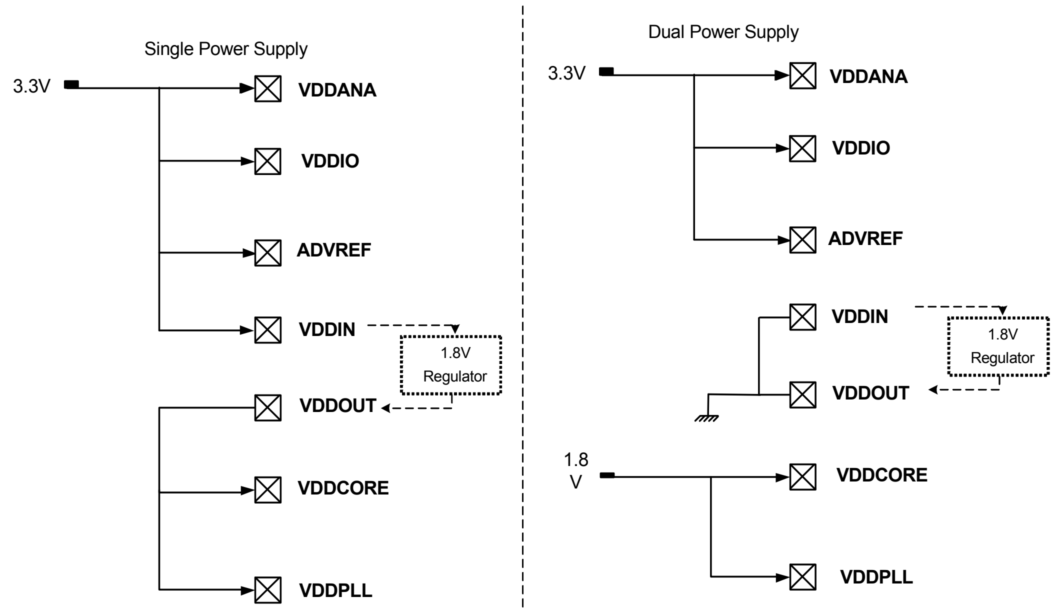
4.2.1 Multiplexed signals

Each GPIO line can be assigned to one of 4 peripheral functions; A, B, C or D (D is only available for UC3Bx512 parts). The following table define how the I/O lines on the peripherals A, B,C or D are multiplexed by the GPIO.

Table 4-1. GPIO Controller Function Multiplexing

48-pin	64-pin	PIN	GPIO Pin	Function A	Function B	Function C	Function D (only for UC3Bx512)
3	3	PA00	GPIO 0				
4	4	PA01	GPIO 1				
5	5	PA02	GPIO 2				
7	9	PA03	GPIO 3	ADC - AD[0]	PM - GCLK[0]	USBB - USB_ID	ABDAC - DATA[0]
8	10	PA04	GPIO 4	ADC - AD[1]	PM - GCLK[1]	USBB - USB_VBOF	ABDAC - DATAN[0]
9	11	PA05	GPIO 5	EIC - EXTINT[0]	ADC - AD[2]	USART1 - DCD	ABDAC - DATA[1]

Figure 5-1. Power Supply



5.6.2 Voltage Regulator

5.6.2.1 Single Power Supply

The AT32UC3B embeds a voltage regulator that converts from 3.3V to 1.8V. The regulator takes its input voltage from VDDIN, and supplies the output voltage on VDDOUT that should be externally connected to the 1.8V domains.

Adequate input supply decoupling is mandatory for VDDIN in order to improve startup stability and reduce source voltage drop. Two input decoupling capacitors must be placed close to the chip.

Adequate output supply decoupling is mandatory for VDDOUT to reduce ripple and avoid oscillations. The best way to achieve this is to use two capacitors in parallel between VDDOUT and GND as close to the chip as possible

Figure 5-2. Supply Decoupling

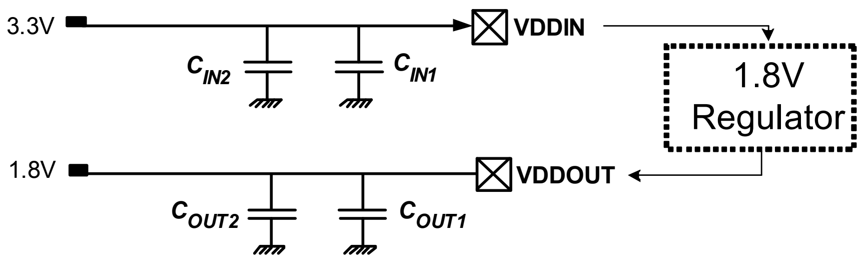
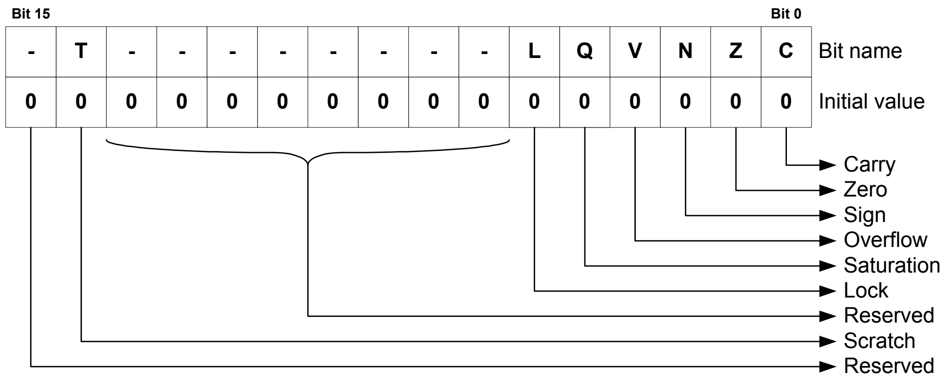


Figure 6-5. The Status Register Low Halfword



6.4.3 Processor States

6.4.3.1 Normal RISC State

The AVR32 processor supports several different execution contexts as shown in Table 6-2 on page 23.

Table 6-2. Overview of Execution Modes, their Priorities and Privilege Levels.

Priority	Mode	Security	Description
1	Non Maskable Interrupt	Privileged	Non Maskable high priority interrupt mode
2	Exception	Privileged	Execute exceptions
3	Interrupt 3	Privileged	General purpose interrupt mode
4	Interrupt 2	Privileged	General purpose interrupt mode
5	Interrupt 1	Privileged	General purpose interrupt mode
6	Interrupt 0	Privileged	General purpose interrupt mode
N/A	Supervisor	Privileged	Runs supervisor calls
N/A	Application	Unprivileged	Normal program execution mode

Mode changes can be made under software control, or can be caused by external interrupts or exception processing. A mode can be interrupted by a higher priority mode, but never by one with lower priority. Nested exceptions can be supported with a minimal software overhead.

When running an operating system on the AVR32, user processes will typically execute in the application mode. The programs executed in this mode are restricted from executing certain instructions. Furthermore, most system registers together with the upper halfword of the status register cannot be accessed. Protected memory areas are also not available. All other operating modes are privileged and are collectively called System Modes. They have full access to all privileged and unprivileged resources. After a reset, the processor will be in supervisor mode.

6.4.3.2 Debug State

The AVR32 can be set in a debug state, which allows implementation of software monitor routines that can read out and alter system information for use during application development. This implies that all system and application registers, including the status registers and program counters, are accessible in debug state. The privileged instructions are also available.

status register. Upon entry into Debug mode, hardware sets the SR[D] bit and jumps to the Debug Exception handler. By default, Debug mode executes in the exception context, but with dedicated Return Address Register and Return Status Register. These dedicated registers remove the need for storing this data to the system stack, thereby improving debuggability. The mode bits in the status register can freely be manipulated in Debug mode, to observe registers in all contexts, while retaining full privileges.

Debug mode is exited by executing the *retd* instruction. This returns to the previous context.

6.5.5 Entry Points for Events

Several different event handler entry points exists. In AVR32UC, the reset address is 0x8000_0000. This places the reset address in the boot flash memory area.

TLB miss exceptions and *scall* have a dedicated space relative to EVBA where their event handler can be placed. This speeds up execution by removing the need for a jump instruction placed at the program address jumped to by the event hardware. All other exceptions have a dedicated event routine entry point located relative to EVBA. The handler routine address identifies the exception source directly.

AVR32UC uses the ITLB and DTLB protection exceptions to signal a MPU protection violation. ITLB and DTLB miss exceptions are used to signal that an access address did not map to any of the entries in the MPU. TLB multiple hit exception indicates that an access address did map to multiple TLB entries, signalling an error.

All external interrupt requests have entry points located at an offset relative to EVBA. This autovector offset is specified by an external Interrupt Controller. The programmer must make sure that none of the autovector offsets interfere with the placement of other code. The autovector offset has 14 address bits, giving an offset of maximum 16384 bytes.

Special considerations should be made when loading EVBA with a pointer. Due to security considerations, the event handlers should be located in non-writeable flash memory, or optionally in a privileged memory protection region if an MPU is present.

If several events occur on the same instruction, they are handled in a prioritized way. The priority ordering is presented in Table 6-4. If events occur on several instructions at different locations in the pipeline, the events on the oldest instruction are always handled before any events on any younger instruction, even if the younger instruction has events of higher priority than the oldest instruction. An instruction B is younger than an instruction A if it was sent down the pipeline later than A.

The addresses and priority of simultaneous events are shown in Table 6-4. Some of the exceptions are unused in AVR32UC since it has no MMU, coprocessor interface, or floating-point unit.

7.3 Peripheral Address Map

Table 7-2. Peripheral Address Mapping

Address		Peripheral Name
0xFFFE0000	USB	USB 2.0 Interface - USB
0xFFFE1000	HMATRIX	HSB Matrix - HMATRIX
0xFFFE1400	HFLASHC	Flash Controller - HFLASHC
0xFFFF0000	PDCA	Peripheral DMA Controller - PDCA
0xFFFF0800	INTC	Interrupt controller - INTC
0xFFFF0C00	PM	Power Manager - PM
0xFFFF0D00	RTC	Real Time Counter - RTC
0xFFFF0D30	WDT	Watchdog Timer - WDT
0xFFFF0D80	EIM	External Interrupt Controller - EIM
0xFFFF1000	GPIO	General Purpose Input/Output Controller - GPIO
0xFFFF1400	USART0	Universal Synchronous/Asynchronous Receiver/Transmitter - USART0
0xFFFF1800	USART1	Universal Synchronous/Asynchronous Receiver/Transmitter - USART1
0xFFFF1C00	USART2	Universal Synchronous/Asynchronous Receiver/Transmitter - USART2
0xFFFF2400	SPI0	Serial Peripheral Interface - SPI0
0xFFFF2C00	TWI	Two-wire Interface - TWI
0xFFFF3000	PWM	Pulse Width Modulation Controller - PWM
0xFFFF3400	SSC	Synchronous Serial Controller - SSC
0xFFFF3800	TC	Timer/Counter - TC

9.2 DC Characteristics

The following characteristics are applicable to the operating temperature range: T_A = -40°C to 85°C, unless otherwise specified and are certified for a junction temperature up to T_J = 100°C.

Table 9-1. DC Characteristics

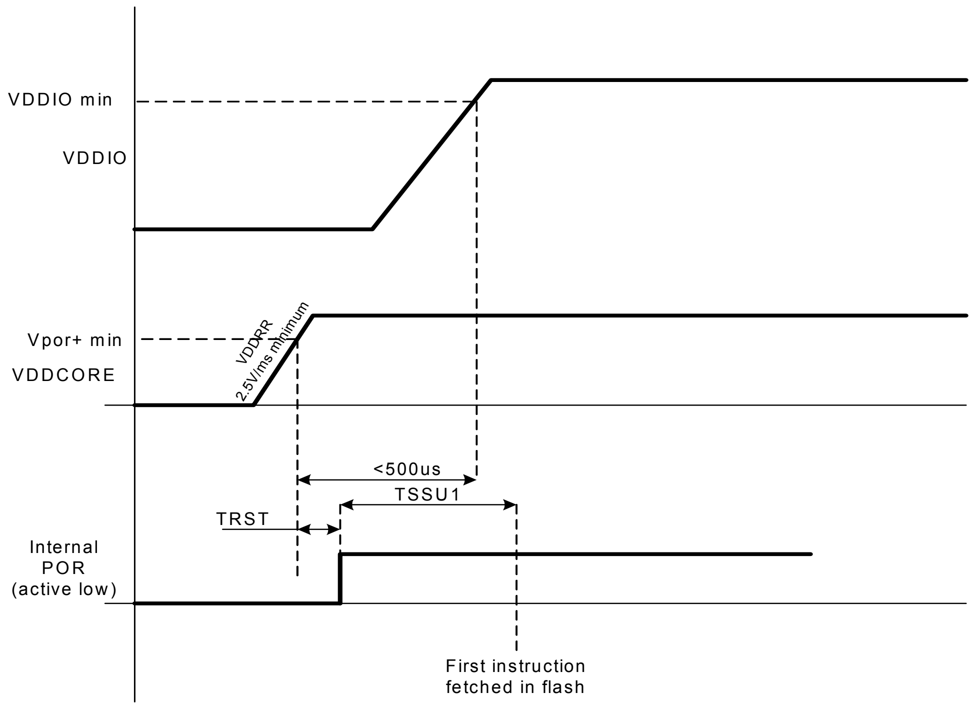
Symbol	Parameter	Conditions		Min.	Typ.	Max.	Unit
$V_{VDDCORE}$	DC Supply Core			1.65		1.95	V
V_{VDDPLL}	DC Supply PLL			1.65		1.95	V
V_{VDDIO}	DC Supply Peripheral I/Os			3.0		3.6	V
V_{IL}	Input Low-level Voltage			-0.3		+0.8	V
V_{IH}	Input High-level Voltage	AT32UC3B064 AT32UC3B0128 AT32UC3B0256 AT32UC3B164 AT32UC3B1128 AT32UC3B1256	All I/O pins except TCK, RESET_N, PA03, PA04, PA05, PA06, PA07, PA08, PA11, PA12, PA18, PA19, PA28, PA29, PA30, PA31	2.0		5.5	V
			TCK, RESET_N, PA03, PA04, PA05, PA06, PA07, PA08, PA11, PA12, PA18, PA19, PA28, PA29, PA30, PA31	2.0		3.6	V
		AT32UC3B0512 AT32UC3B1512	All I/O pins except TCK, RESET_N, PA03, PA04, PA05, PA06, PA07, PA08, PA11, PA12, PA18, PA19, PA28, PA29, PA30, PA31	2.0		5.5	V
			TCK, RESET_N	2.5		3.6	V
			PA03, PA04, PA05, PA06, PA07, PA08, PA11, PA12, PA18, PA19, PA28, PA29, PA30, PA31	2.0		3.6	V
V_{OL}	Output Low-level Voltage	I_{OL}= -4mA for all I/O except PA20, PA21, PA22, PA23				0.4	V
		I_{OL}= -8mA for PA20, PA21, PA22, PA23				0.4	V
V_{OH}	Output High-level Voltage	I_{OL}= -4mA for all I/O except PA20, PA21, PA22, PA23		V_{VDDIO} -0.4			V
		I_{OL}= -8mA for PA20, PA21, PA22, PA23		V_{VDDIO} -0.4			V
I_{OL}	Output Low-level Current	All I/O pins except PA20, PA21, PA22, PA23				-4	mA
		PA20, PA21, PA22, PA23				-8	mA
I_{OH}	Output High-level Current	All I/O pins except for PA20, PA21, PA22, PA23				4	mA
		PA20, PA21, PA22, PA23				8	mA
I_{LEAK}	Input Leakage Current	Pullup resistors disabled				1	µA

Table 9-1. DC Characteristics

Symbol	Parameter	Conditions			Min.	Typ.	Max.	Unit
C_{IN}	Input Capacitance	QFP64					7	pF
		QFP48					7	pF
		QFN64					7	pF
		QFN48					7	pF
R_{PULLUP}	Pull-up Resistance	AT32UC3B064 AT32UC3B0128 AT32UC3B0256 AT32UC3B164 AT32UC3B1128 AT32UC3B1256	All I/O pins except RESET_N, TCK, TDI, TMS pins		13	19	25	KΩ
			RESET_N pin, TCK, TDI, TMS pins		5	12	25	KΩ
		AT32UC3B0512 AT32UC3B1512	All I/O pins except PA20, PA21, PA22, PA23, RESET_N, TCK, TDI, TMS pins		10	15	20	KΩ
			PA20, PA21, PA22, PA23		5	7.5	12	KΩ
			RESET_N pin, TCK, TDI, TMS pins		5	10	25	KΩ
I_{SC}	Static Current	AT32UC3B064 AT32UC3B0128 AT32UC3B0256 AT32UC3B164 AT32UC3B1128 AT32UC3B1256	On $V_{VDDCORE}$ = 1.8V, device in static mode All inputs driven including JTAG; RESET_N=1	T_A = 25°C		6		µA
				T_A = 85°C		42.5		µA
		AT32UC3B0512 AT32UC3B1512	On $V_{VDDCORE}$ = 1.8V, device in static mode All inputs driven including JTAG; RESET_N=1	T_A = 25°C		7.5		µA
				T_A = 85°C		39		µA

Therefore VDDCORE rise rate (VDDRR) must be equal or superior to 2.5V/ms and VDDIO must reach VDDIO mini value before 500 us (< TRST + TSSU1) after VDDCORE has reached V_{POR+} min value.

Figure 9-4. Dual Supply Configuration

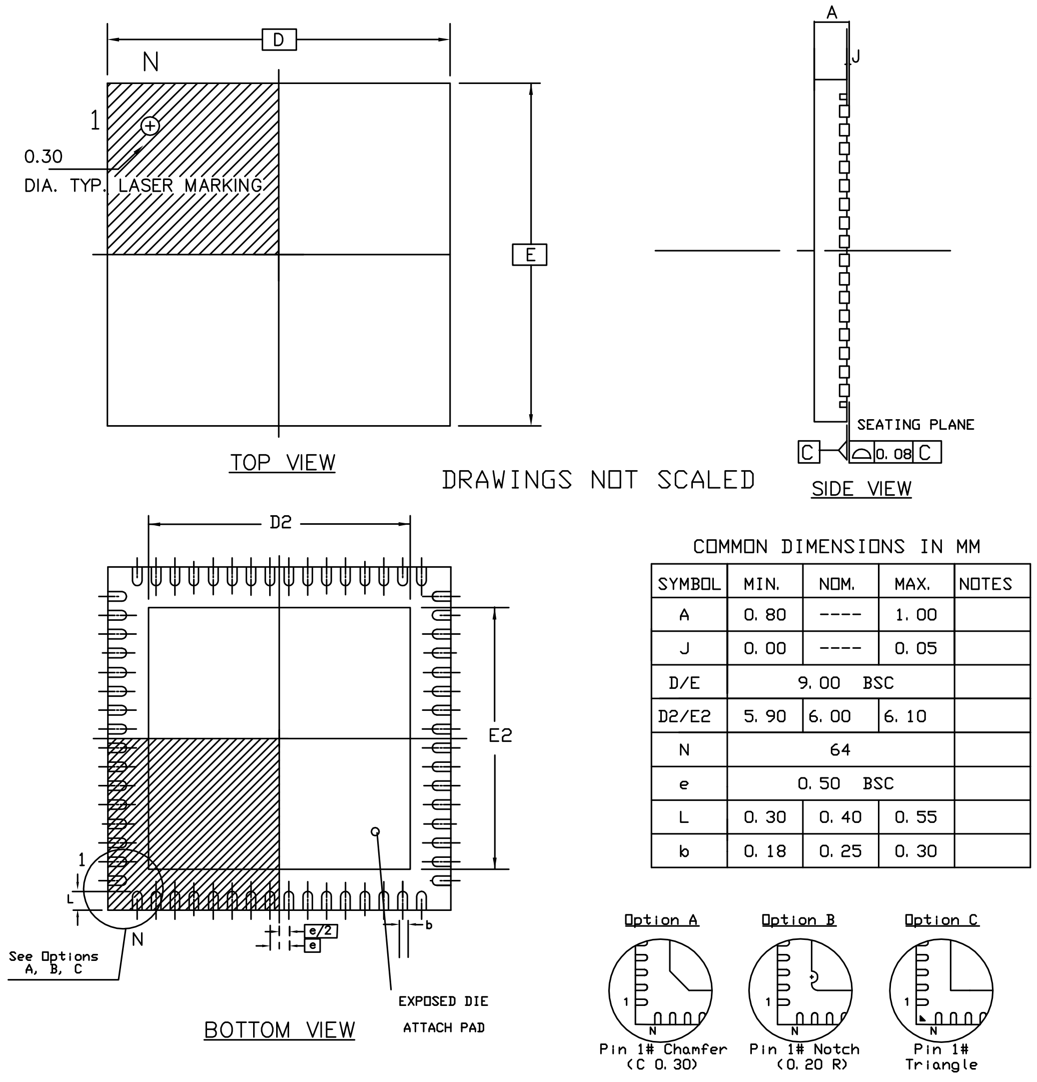


9.4.4 RESET_N Characteristics

Table 9-9. RESET_N Waveform Parameters

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Unit
t_{RESET}	RESET_N minimum pulse width		10			ns

Figure 10-3. QFN-64 package drawing



COMMON DIMENSIONS IN MM

SYMBOL	MIN.	NOM.	MAX.	NOTES
A	0. 80	----	1. 00	
J	0. 00	----	0. 05	
D/E		9. 00 BSC		
D2/E2	5. 90	6. 00	6. 10	
N		64		
e		0. 50 BSC		
L	0. 30	0. 40	0. 55	
b	0. 18	0. 25	0. 30	

Compliant JEDEC Standard MO-220 variation VMMD-3

Table 10-8. Device and Package Maximum Weight

Weight	200 mg

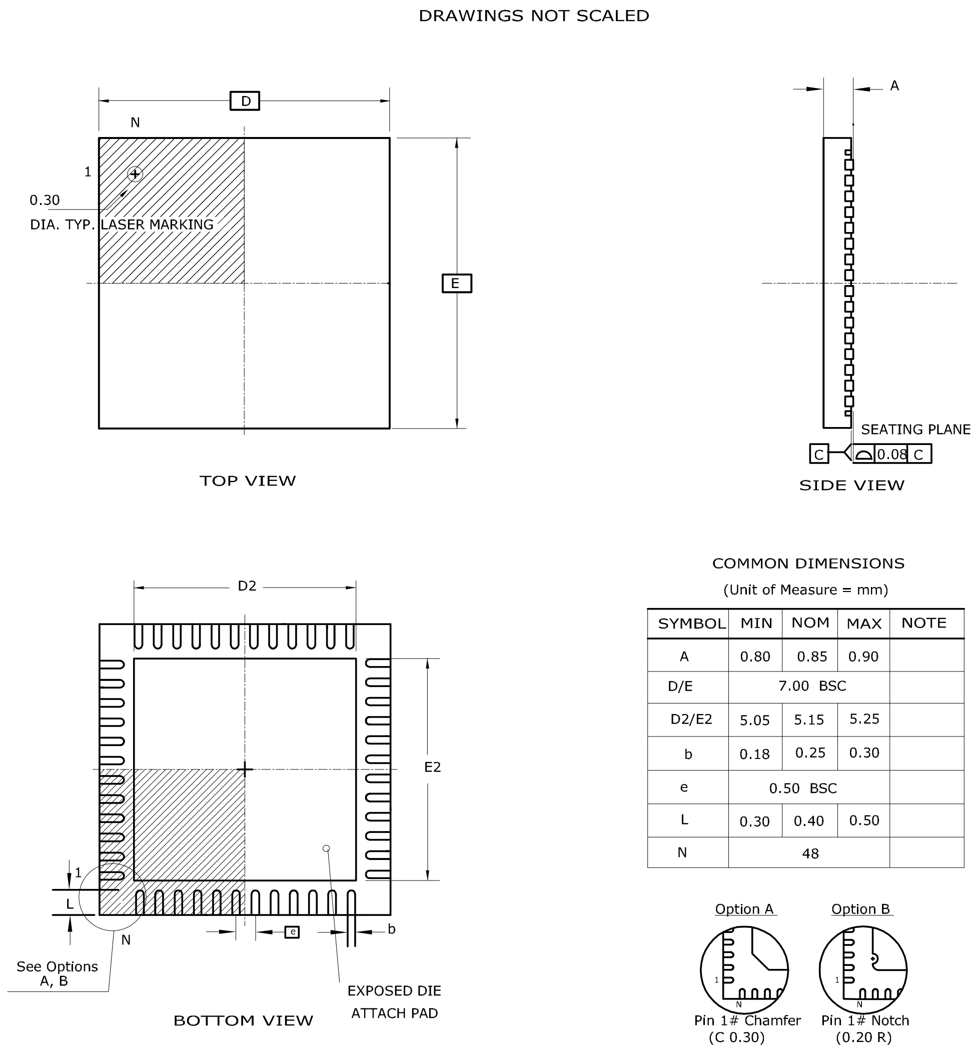
Table 10-9. Package Characteristics

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

Table 10-10. Package Reference

JEDEC Drawing Reference	M0-220
JESD97 Classification	e3

Figure 10-4. QFN-48 package drawing



COMMON DIMENSIONS

(Unit of Measure = mm)

SYMBOL	MIN	NOM	MAX	NOTE
A	0.80	0.85	0.90	
D/E		7.00 BSC		
D2/E2	5.05	5.15	5.25	
b	0.18	0.25	0.30	
e		0.50 BSC		
L	0.30	0.40	0.50	
N		48		

Notes : 1. This drawing is for general information only. Refer to JEDEC Drawing MO-220, Variation VKKD-4, for proper dimensions, tolerances, datums, etc.
2. Dimension b applies to metallized terminal and is measured between 0.15mm and 0.30mm from the terminal tip.
If the terminal has the optical radius on the other end of the terminal, the dimension should not be measured in that radius area.

Table 10-11. Device and Package Maximum Weight

Weight	100 mg

Table 10-12. Package Characteristics

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

Table 10-13. Package Reference

JEDEC Drawing Reference	M0-220
JESD97 Classification	e3

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.

8. **SPI disable does not work in SLAVE mode**
   SPI disable does not work in SLAVE mode.
   Fix/Workaround
   Read the last received data, then perform a software reset by writing a one to the Software Reset bit in the Control Register (CR.SWRST).

9. **SPI data transfer hangs with CSR0.CSAAT==1 and MR.MODFDIS==0**
   When CSR0.CSAAT==1 and mode fault detection is enabled (MR.MODFDIS==0), the SPI module will not start a data transfer.
   Fix/Workaround
   Disable mode fault detection by writing a one to MR.MODFDIS.

10. **Disabling SPI has no effect on the SR.TDRE bit**
    Disabling SPI has no effect on the SR.TDRE bit whereas the write data command is filtered when SPI is disabled. Writing to TDR when SPI is disabled will not clear SR.TDRE. If SPI is disabled during a PDCA transfer, the PDCA will continue to write data to TDR until its buffer is empty, and this data will be lost.
    Fix/Workaround
    Disable the PDCA, add two NOPs, and disable the SPI. To continue the transfer, enable the SPI and PDCA.

11. **Power Manager**

12. **If the BOD level is higher than VDDCORE, the part is constantly reset**
    If the BOD level is set to a value higher than VDDCORE and enabled by fuses, the part will be in constant reset.
    Fix/Workaround
    Apply an external voltage on VDDCORE that is higher than the BOD level and is lower than VDDCORE max and disable the BOD.

13. **When the main clock is RCSYS, TIMER_CLOCK5 is equal to PBA clock**
    When the main clock is generated from RCSYS, TIMER_CLOCK5 is equal to PBA Clock and not PBA Clock / 128.
    Fix/Workaround
    None.

14. **Clock sources will not be stopped in STATIC sleep mode if the difference between CPU and PBx division factor is too high**
    If the division factor between the CPU/HSB and PBx frequencies is more than 4 when going to a sleep mode where the system RC oscillator is turned off, then high speed clock sources

20. USB

21. 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.

2. Transfer error will stall a transmit peripheral handshake interface
If a transfer error is encountered on a channel transmitting to a peripheral, the peripheral handshake of the active channel will stall and the PDCA will not do any more transfers on the affected peripheral handshake interface.
Fix/Workaround
Disable and then enable the peripheral after the transfer error.

3. TWI

4. The TWI RXRDY flag in SR register is not reset when a software reset is performed
The TWI RXRDY flag in SR register is not reset when a software reset is performed.
Fix/Workaround
After a Software Reset, the register TWI RHR must be read.

5. TWI in master mode will continue to read data
TWI in master mode will continue to read data on the line even if the shift register and the RHR register are full. This will generate an overrun error.
Fix/Workaround
To prevent this, read the RHR register as soon as a new RX data is ready.

6. TWI slave behaves improperly if master acknowledges the last transmitted data byte before a STOP condition
In I2C slave transmitter mode, if the master acknowledges the last data byte before a STOP condition (what the master is not supposed to do), the following TWI slave receiver mode frame may contain an inappropriate clock stretch. This clock stretch can only be stopped by resetting the TWI.
Fix/Workaround
If the TWI is used as a slave transmitter with a master that acknowledges the last data byte before a STOP condition, it is necessary to reset the TWI before entering slave receiver mode.

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 transfer is started and a receive buffer is available, write a one to the RTSEN bit in the USART CR so that RTS will be driven low.

8. **Corruption after receiving too many bits in SPI slave mode**
   If the USART is in SPI slave mode and receives too much data bits (ex: 9bitsinstead of 8 bits) by the SPI master, an error occurs. After that, the next reception may be corrupted even if the frame is correct and the USART has been disabled, reset by a soft reset and re-enabled.
   Fix/Workaround
   None.

9. **USART slave synchronous mode external clock must be at least 9 times lower in frequency than CLK_USART**
   When the USART is operating in slave synchronous mode with an external clock, the frequency of the signal provided on CLK must be at least 9 times lower than CLK_USART.
   Fix/Workaround
   When the USART is operating in slave synchronous mode with an external clock, provide a signal on CLK that has a frequency at least 9 times lower than CLK_USART.

10. **HMATRIX**

11. **In the PRAS and PRBS registers, the MxPR fields are only two bits**
    In the PRAS and PRBS registers, the MxPR fields are only two bits wide, instead of four bits. The unused bits are undefined when reading the registers.
    Fix/Workaround
    Mask undefined bits when reading PRAS and PRBS.

- *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.

12.1.2 Rev C

- *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.

14. **SSC**

15. **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.

16. **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.

17. **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.

18. **USB**

19. **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.

even if the frame is correct and the USART has been disabled, reset by a soft reset and re-enabled.
Fix/Workaround
None.

9. **USART slave synchronous mode external clock must be at least 9 times lower in frequency than CLK_USART**
When the USART is operating in slave synchronous mode with an external clock, the frequency of the signal provided on CLK must be at least 9 times lower than CLK_USART.
Fix/Workaround
When the USART is operating in slave synchronous mode with an external clock, provide a signal on CLK that has a frequency at least 9 times lower than CLK_USART.

10. **HMATRIX**

11. **In the PRAS and PRBS registers, the MxPR fields are only two bits**
In the PRAS and PRBS registers, the MxPR fields are only two bits wide, instead of four bits. The unused bits are undefined when reading the registers.
Fix/Workaround
Mask undefined bits when reading PRAS and PRBS.

- *FLASHC*

1. **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.

- *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.

- USART

1. **USART Manchester Encoder Not Working**
   Manchester encoding/decoding is not working.
   Fix/Workaround
   Do not use manchester encoding.

2. **USART RXBREAK problem when no timeguard**
   In asynchronous mode the RXBREAK flag is not correctly handled when the timeguard is 0 and the break character is located just after the stop bit.
   Fix/Workaround
   If the NBSTOP is 1, timeguard should be different from 0.

3. **USART Handshaking: 2 characters sent / CTS rises when TX**
   If CTS switches from 0 to 1 during the TX of a character, if the Holding register is not empty, the TXHOLDING is also transmitted.
   Fix/Workaround
   None.

4. **USART PDC and TIMEGUARD not supported in MANCHESTER**
   Manchester encoding/decoding is not working.
   Fix/Workaround
   Do not use manchester encoding.

5. **USART SPI mode is non functional on this revision**
   USART SPI mode is non functional on this revision.
   Fix/Workaround
   Do not use the USART SPI mode.

- HMATRIX

1. **HMatrix fixed priority arbitration does not work**
   Fixed priority arbitration does not work.
   Fix/Workaround
   Use Round-Robin arbitration instead.

- Clock characteristic

1. **PBA max frequency**
   The Peripheral bus A (PBA) max frequency is 30MHz instead of 60MHz.
   Fix/Workaround
   Do not set the PBA maximum frequency higher than 30MHz.

- FLASHC

1. **The address of Flash General Purpose Fuse Register Low (FGPFRLO) is 0xFFFE140C on revB instead of 0xFFFE1410**
   The address of Flash General Purpose Fuse Register Low (FGPFRLO) is 0xFFFE140C on revB instead of 0xFFFE1410.
   Fix/Workaround
   None.

13. Datasheet Revision History

Please note that the referring page numbers in this section are referred to this document. The referring revision in this section are referring to the document revision.

13.1 Rev. L– 01/2012

1. Updated Mechanical Characteristics section.

13.2 Rev. K– 02/2011

1. Updated USB section.
2. Updated Configuration Summary section.
3. Updated Electrical Characteristics section.
4. Updated Errata section.

13.3 Rev. J– 12/2010

1. Updated USB section.
2. Updated USART section.
3. Updated TWI section.
4. Updated PWM section.
5. Updated Electrical Characteristics section.

13.4 Rev. I – 06/2010

1. Updated SPI section.

2 Updated Electrical Characteristics section.

13.5 Rev. H – 10/2009

1. Update datasheet architecture.

2 Add AT32UC3B0512 and AT32UC3B1512 devices description.

Table of Contents