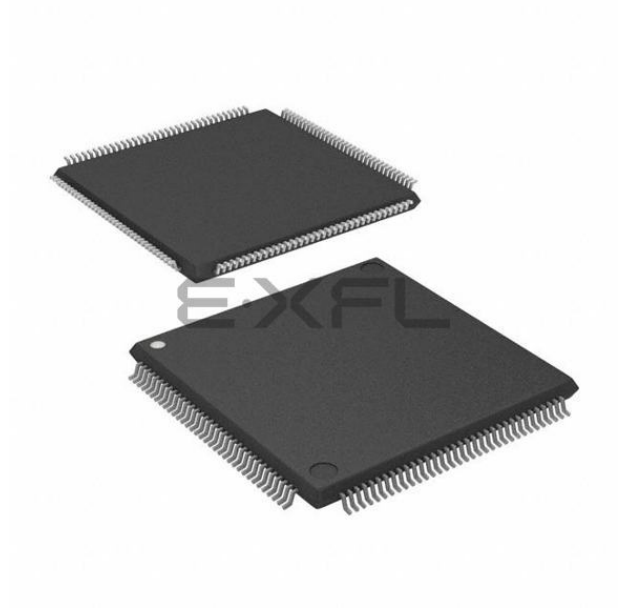


Welcome to E-XFL.COM

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	EBI/EMI, I²C, IrDA, Memory Card, SPI, SSC, UART/USART, USB OTG
Peripherals	Brown-out Detect/Reset, DMA, POR, WDT
Number of I/O	110
Program Memory Size	64KB (64K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	128K x 8
Voltage - Supply (Vcc/Vdd)	1.75V ~ 3.6V
Data Converters	A/D 8x10b
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/at32uc3a364-alut

1. Description

The AT32UC3A3/A4 is a complete System-On-Chip microcontroller based on the AVR32 UC RISC processor running at frequencies up to 84MHz. AVR32 UC is a high-performance 32-bit RISC microprocessor core, designed for cost-sensitive embedded applications, with particular emphasis on low power consumption, high code density and high performance.

The processor implements a Memory Protection Unit (MPU) and a fast and flexible interrupt controller for supporting modern operating systems and real-time operating systems. Higher computation capabilities are achievable using a rich set of DSP instructions.

The AT32UC3A3/A4 incorporates on-chip Flash and SRAM memories for secure and fast access. 64 KBytes of SRAM are directly coupled to the AVR32 UC for performances optimization. Two blocks of 32 Kbytes SRAM are independently attached to the High Speed Bus Matrix, allowing real ping-pong management.

The Peripheral Direct Memory Access Controller (PDCA) enables data transfers between peripherals and memories without processor involvement. The PDCA drastically reduces processing overhead when transferring continuous and large data streams.

The Power Manager improves design flexibility and security: the on-chip Brown-Out Detector monitors the power supply, the CPU runs from the on-chip RC oscillator or from one of external oscillator sources, a Real-Time Clock and its associated timer keeps track of the time.

The device includes two sets of three identical 16-bit Timer/Counter (TC) channels. Each channel can be independently programmed to perform frequency measurement, event counting, interval measurement, pulse generation, delay timing and pulse width modulation. 16-bit channels are combined to operate as 32-bit channels.

The AT32UC3A3/A4 also features many communication interfaces for communication intensive applications like UART, SPI or TWI. The USART supports different communication modes, like SPI Mode and LIN Mode. Additionally, a flexible Synchronous Serial Controller (SSC) is available. The SSC provides easy access to serial communication protocols and audio standards like I2S.

The AT32UC3A3/A4 includes a powerfull External Bus Interface to interface all standard memory device like SRAM, SDRAM, NAND Flash or parallel interfaces like LCD Module.

The peripheral set includes a High Speed MCI for SDIO/SD/MMC and a hardware encryption module based on AES algorithm.

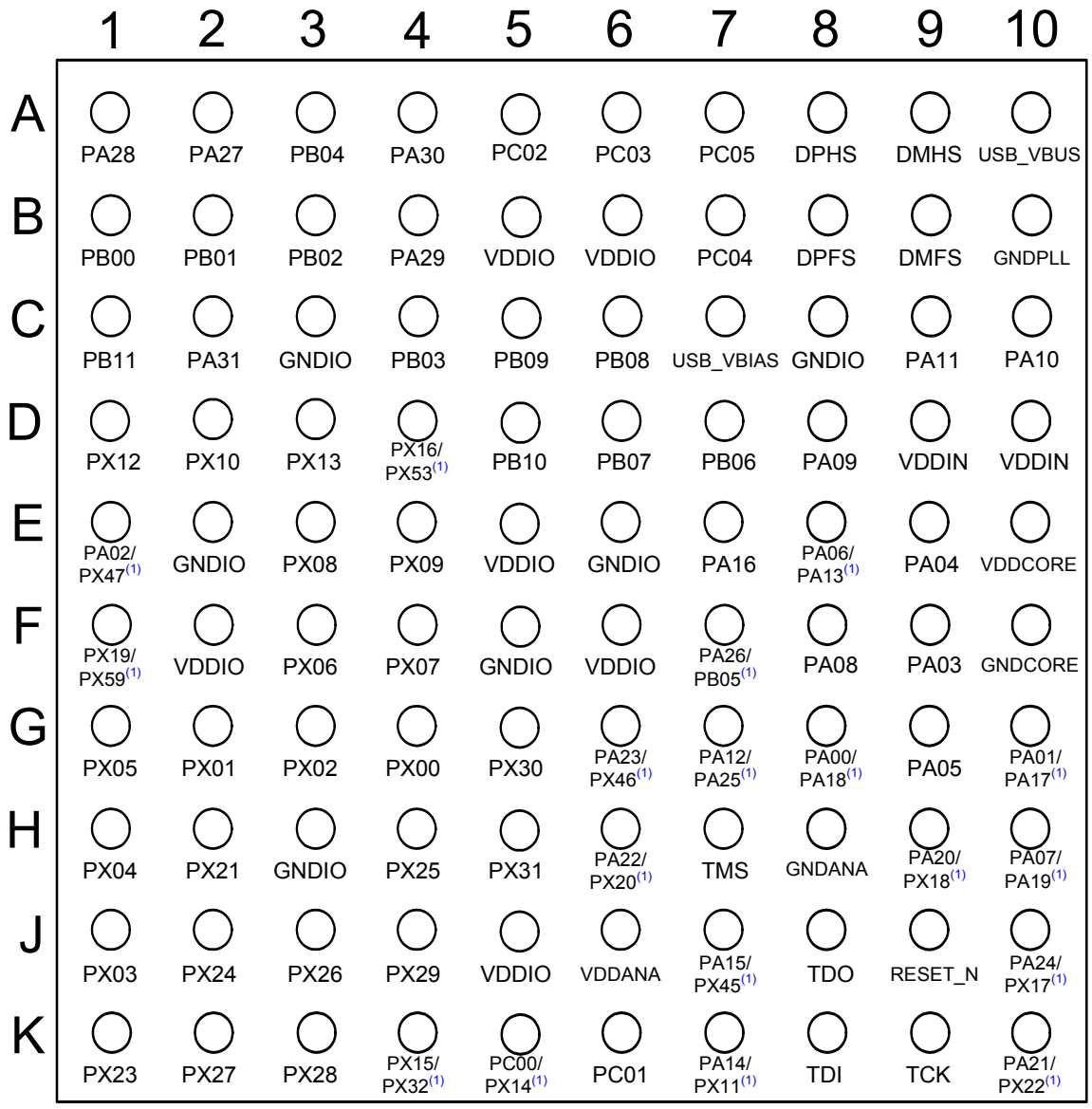
The device embeds a 10-bit ADC and a Digital Audio bistream DAC.

The Direct Memory Access controller (DMACA) allows high bandwidth data flows between high speed peripherals (USB, External Memories, MMC, SDIO, ...) and through high speed internal features (AES, internal memories).

The High-Speed (480MBit/s) USB 2.0 Device and Host interface supports several USB Classes at the same time thanks to the rich Endpoint configuration. The Embedded Host interface allows device like a USB Flash disk or a USB printer to be directly connected to the processor. This periphal has its own dedicated DMA and is perfect for Mass Storage application.

AT32UC3A3/A4 integrates a class 2+ Nexus 2.0 On-Chip Debug (OCD) System, with non-intrusive real-time trace, full-speed read/write memory access in addition to basic runtime control.

Figure 3-3. VFBGA100 Pinout (top view)



Note: 1. Those balls are physically connected to 2 GPIOs. Software must managed carrefully the GPIO configuration to avoid electrical conflict

3.2.4 JTAG port connections

Table 3-4. JTAG Pinout

TFBGA144	QFP144	VFBGA100	Pin name	JTAG pin
K12	107	K9	TCK	TCK
L12	108	K8	TDI	TDI
J11	105	J8	TDO	TDO
J10	104	H7	TMS	TMS

3.2.5 Nexus OCD AUX port connections

If the OCD trace system is enabled, the trace system will take control over a number of pins, irrespective of the GPIO configuration. Three differents 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 3-5. Nexus OCD AUX port connections

Pin	AXS=0	AXS=1	AXS=2
EVTI_N	PB05	PA08	PX00
MDO[5]	PA00	PX56	PX06
MDO[4]	PA01	PX57	PX05
MDO[3]	PA03	PX58	PX04
MDO[2]	PA16	PA24	PX03
MDO[1]	PA13	PA23	PX02
MDO[0]	PA12	PA22	PX01
MSEO[1]	PA10	PA07	PX08
MSEO[0]	PA11	PX55	PX07
MCKO	PB07	PX00	PB09
EVTO_N	PB06	PB06	PB06

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.

4.3 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 4-1 on page 23 displays the contents of AVR32UC.

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

4.3.6 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

4.3.7 CPU and Architecture Revision

Three major revisions of the AVR32UC CPU currently exist.

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.

Table 4-3. System Registers (Continued)

Reg #	Address	Name	Function
92	368	MPUPSR4	MPU Privilege Select Register region 4
93	372	MPUPSR5	MPU Privilege Select Register region 5
94	376	MPUPSR6	MPU Privilege Select Register region 6
95	380	MPUPSR7	MPU Privilege Select Register region 7
96	384	MPUCRA	Unused in this version of AVR32UC
97	388	MPUCRB	Unused in this version of AVR32UC
98	392	MPUBRA	Unused in this version of AVR32UC
99	396	MPUBRB	Unused in this version of AVR32UC
100	400	MPUAPRA	MPU Access Permission Register A
101	404	MPUAPRB	MPU Access Permission Register B
102	408	MPUCR	MPU Control Register
103-191	448-764	Reserved	Reserved for future use
192-255	768-1020	IMPL	IMPLEMENTATION DEFINED

4.5 Exceptions and Interrupts

AVR32UC incorporates a powerful exception handling scheme. The different exception sources, like Illegal Op-code and external interrupt requests, have different priority levels, ensuring a well-defined behavior when multiple exceptions are received simultaneously. Additionally, pending exceptions of a higher priority class may preempt handling of ongoing exceptions of a lower priority class.

When an event occurs, the execution of the instruction stream is halted, and execution control is passed to an event handler at an address specified in Table 4-4 on page 33. Most of the handlers are placed sequentially in the code space starting at the address specified by EVBA, with four bytes between each handler. This gives ample space for a jump instruction to be placed there, jumping to the event routine itself. A few critical handlers have larger spacing between them, allowing the entire event routine to be placed directly at the address specified by the EVBA-relative offset generated by hardware. All external interrupt sources have autovectored interrupt service routine (ISR) addresses. This allows the interrupt controller to directly specify the ISR address as an address relative to EVBA. The autovector offset has 14 address bits, giving an offset of maximum 16384 bytes. The target address of the event handler is calculated as (EVBA | event_handler_offset), not (EVBA + event_handler_offset), so EVBA and exception code segments must be set up appropriately. The same mechanisms are used to service all different types of events, including external interrupt requests, yielding a uniform event handling scheme.

An interrupt controller does the priority handling of the external interrupts and provides the autovector offset to the CPU.

4.5.1 System Stack Issues

Event handling in AVR32UC uses the system stack pointed to by the system stack pointer, SP_SYS, for pushing and popping R8-R12, LR, status register, and return address. Since event code may be timing-critical, SP_SYS should point to memory addresses in the IRAM section, since the timing of accesses to this memory section is both fast and deterministic.

Figure 7-3. MCU Cold Start-Up

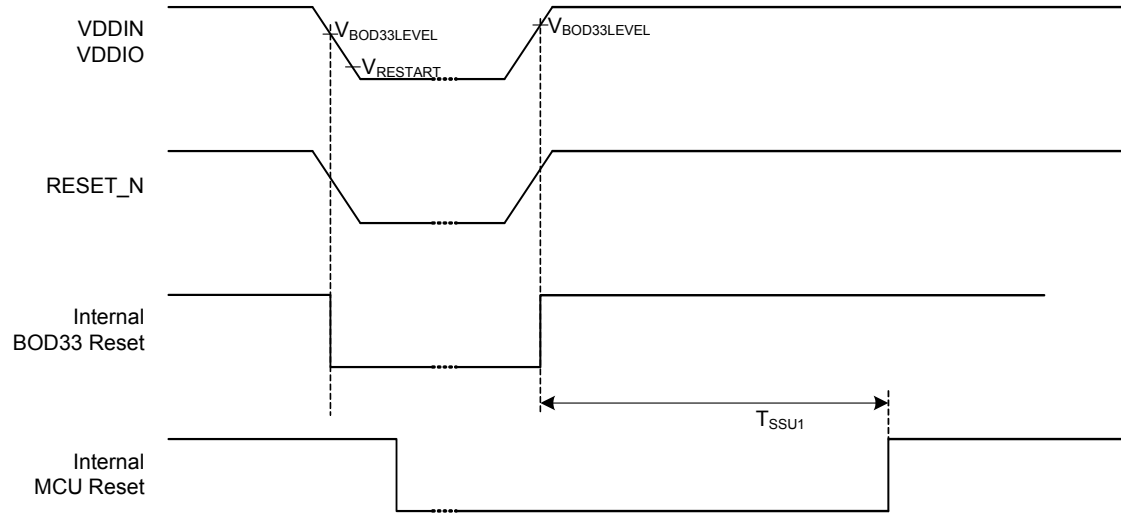


Figure 7-4. MCU Cold Start-Up RESET_N Externally Driven

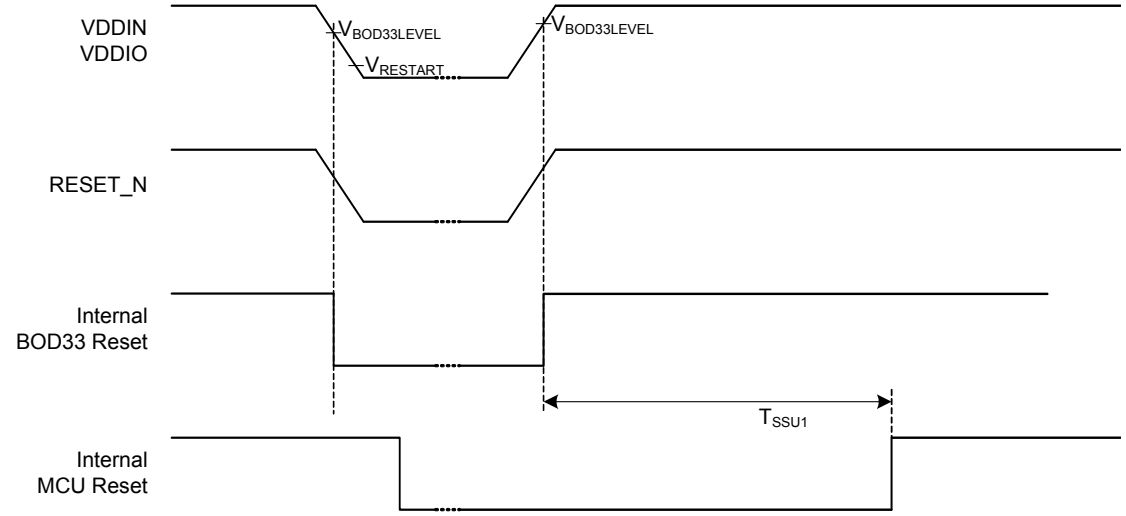
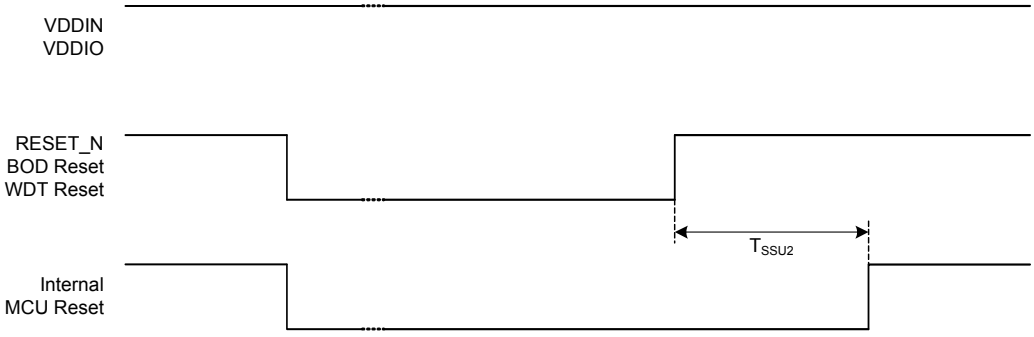


Figure 7-5. MCU Hot Start-Up

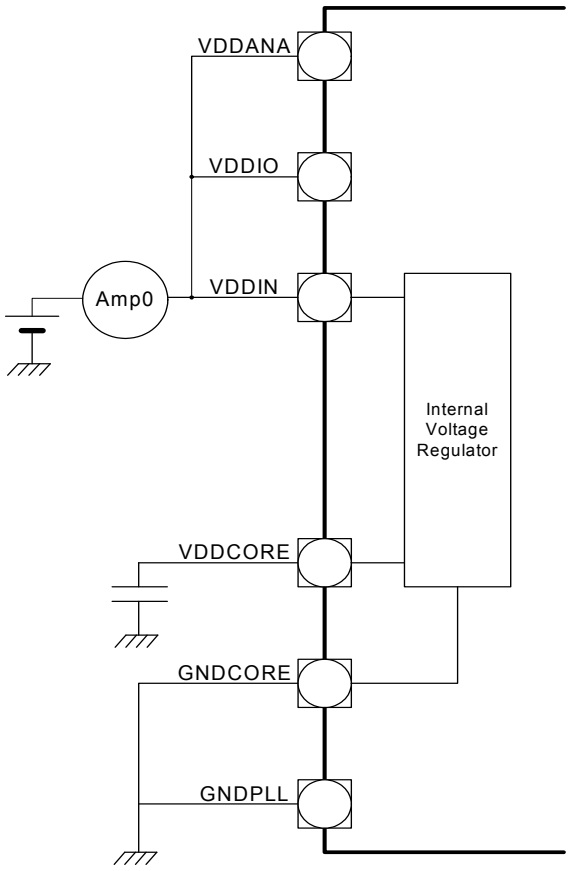


7.6 Power Consumption

The values in Table 7-12 and Table 7-13 on page 50 are measured values of power consumption with operating conditions as follows:

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

Figure 7-6. Measurement Setup



These figures represent the power consumption measured on the power supplies

7.8.3 Main Oscillators

Table 7-19. Main Oscillators Characteristics

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Unit
$1/(t_{CPMAIN})$	Oscillator Frequency	External clock on XIN			50	MHz
		Crystal	0.4		20	MHz
C_{L1}, C_{L2}	Internal Load Capacitance ($C_{L1} = C_{L2}$)			7		pF
ESR	Crystal Equivalent Series Resistance				75	Ω
	Duty Cycle		40	50	60	%
t_{ST}	Startup Time	f = 400 KHz f = 8 MHz f = 16 MHz f = 20 MHz		25 4 1.4 1		ms
t_{CH}	XIN Clock High Half-period		0.4 t_{CP}		0.6 t_{CP}	
t_{CL}	XIN Clock Low Half-period		0.4 t_{CP}		0.6 t_{CP}	
C_{IN}	XIN Input Capacitance			7		pF
I_{OSC}	Current Consumption	Active mode at 400 KHz. Gain = G0 Active mode at 8 MHz. Gain = G1 Active mode at 16 MHz. Gain = G2 Active mode at 20 MHz. Gain = G3		30 45 95 205		µA

7.8.4 Phase Lock Loop (PLL0, PLL1)

Table 7-20. PLL Characteristics

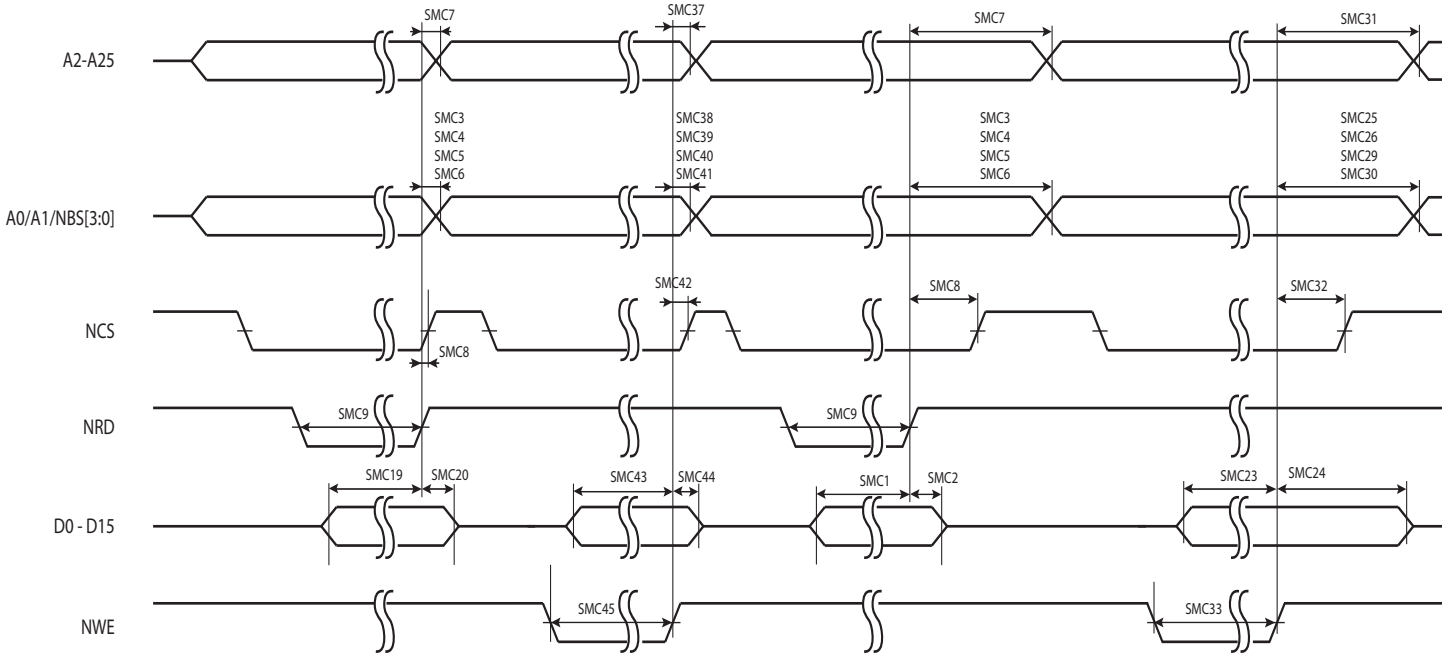
Symbol	Parameter	Conditions	Min.	Typ.	Max.	Unit
F_{OUT}	VCO Output Frequency		80		240	MHz
F_{IN}	Input Frequency (after input divider)		4		16	MHz
I_{PLL}	Current Consumption	Active mode (Fout=80 MHz)		250		µA
		Active mode (Fout=240 MHz)		600		µA

7.8.5 USB Hi-Speed Phase Lock Loop

Table 7-21. PLL Characteristics

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Unit
F_{OUT}	VCO Output Frequency			480		MHz
F_{IN}	Input Frequency			12		MHz
Delta F_{IN}	Input Frequency Accuracy (applicable to Clock signal on XIN or to Quartz tolerance)		-500		+500	ppm
I_{PLL}	Current Consumption	Active mode @480MHz @1.8V		2.5		mA

Figure 7-8. SMC Signals for NRD and NRW Controlled Accesses.



7.11.2 SDRAM Signals

These timings are given for 10 pF load on SDCK and 40 pF on other signals.

Table 7-35. SDRAM Clock Signal.

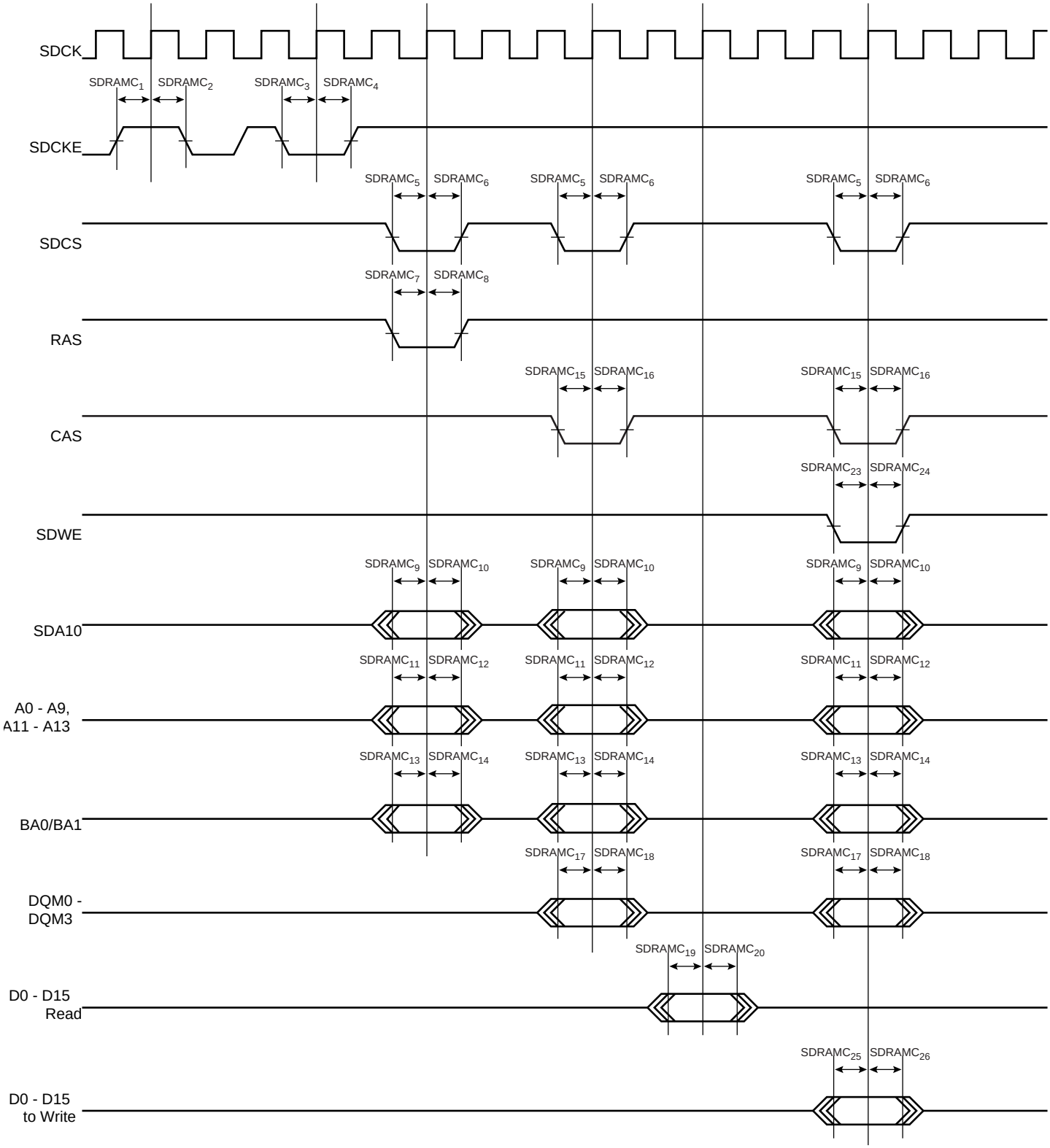
Symbol	Parameter	Conditions	Min.	Max.(1)	Unit
$1/(t_{CPSDCK})$	SDRAM Controller Clock Frequency			$1/(t_{cpcpu})$	MHz

Note: 1. The maximum frequency of the SDRAMC interface is the same as the max frequency for the HSB.

Table 7-36. SDRAM Clock Signal

Symbol	Parameter	Conditions	Min.	Max.	Unit
$SDRAMC_1$	SDCKE High before SDCK Rising Edge		7.4		ns
$SDRAMC_2$	SDCKE Low after SDCK Rising Edge		3.2		ns
$SDRAMC_3$	SDCKE Low before SDCK Rising Edge		7		ns
$SDRAMC_4$	SDCKE High after SDCK Rising Edge		2.9		ns
$SDRAMC_5$	SDCS Low before SDCK Rising Edge		7.5		ns
$SDRAMC_6$	SDCS High after SDCK Rising Edge		1.6		ns
$SDRAMC_7$	RAS Low before SDCK Rising Edge		7.2		ns
$SDRAMC_8$	RAS High after SDCK Rising Edge		2.3		ns
$SDRAMC_9$	SDA10 Change before SDCK Rising Edge		7.6		ns
$SDRAMC_{10}$	SDA10 Change after SDCK Rising Edge		1.9		ns
$SDRAMC_{11}$	Address Change before SDCK Rising Edge		6.2		ns
$SDRAMC_{12}$	Address Change after SDCK Rising Edge		2.2		ns

Figure 7-9. SDRAMC Signals relative to SDCK.



7.15 Flash Memory Characteristics

The following table gives the device maximum operating frequency depending on the field FWS of the Flash FSR register. This field defines the number of wait states required to access the Flash Memory. Flash operating frequency equals the CPU/HSB frequency.

Table 7-39. Flash Operating Frequency

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Unit
F_{FOP}	Flash Operating Frequency	FWS = 0 High Speed Read Mode Disable -40°C < Ambient Temperature < 85°C			36	MHz
		FWS = 1 High Speed Read Mode Disable -40°C < Ambient Temperature < 85°C			66	MHz
		FWS = 0 High Speed Read Mode Enable -40°C < Ambient Temperature < 70°C			42	MHz
		FWS = 1 High Speed Read Mode Enable -40°C < Ambient Temperature < 70°C			84	MHz

Table 7-40. Parts Programming Time

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Unit
T_{FPP}	Page Programming Time			5		ms
T_{FFP}	Fuse Programming Time			0.5		ms
T_{FCE}	Chip erase Time			8		ms

Table 7-41. Flash Parameters

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Unit
N_{FARRAY}	Flash Array Write/Erase cycle				100K	cycle
N_{FFUSE}	General Purpose Fuses write cycle				1000	cycle
T_{FDR}	Flash Data Retention Time			15		year

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	Still Air	TQFP144	40.3	°C/W
θ_{JC}	Junction-to-case thermal resistance		TQFP144	9.5	
θ_{JA}	Junction-to-ambient thermal resistance	Still Air	TFBGA144	28.5	°C/W
θ_{JC}	Junction-to-case thermal resistance		TFBGA144	6.9	
θ_{JA}	Junction-to-ambient thermal resistance	Still Air	VFBGA100	31.1	°C/W
θ_{JC}	Junction-to-case thermal resistance		VFBGA100	6.9	

8.1.2 Junction Temperature

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

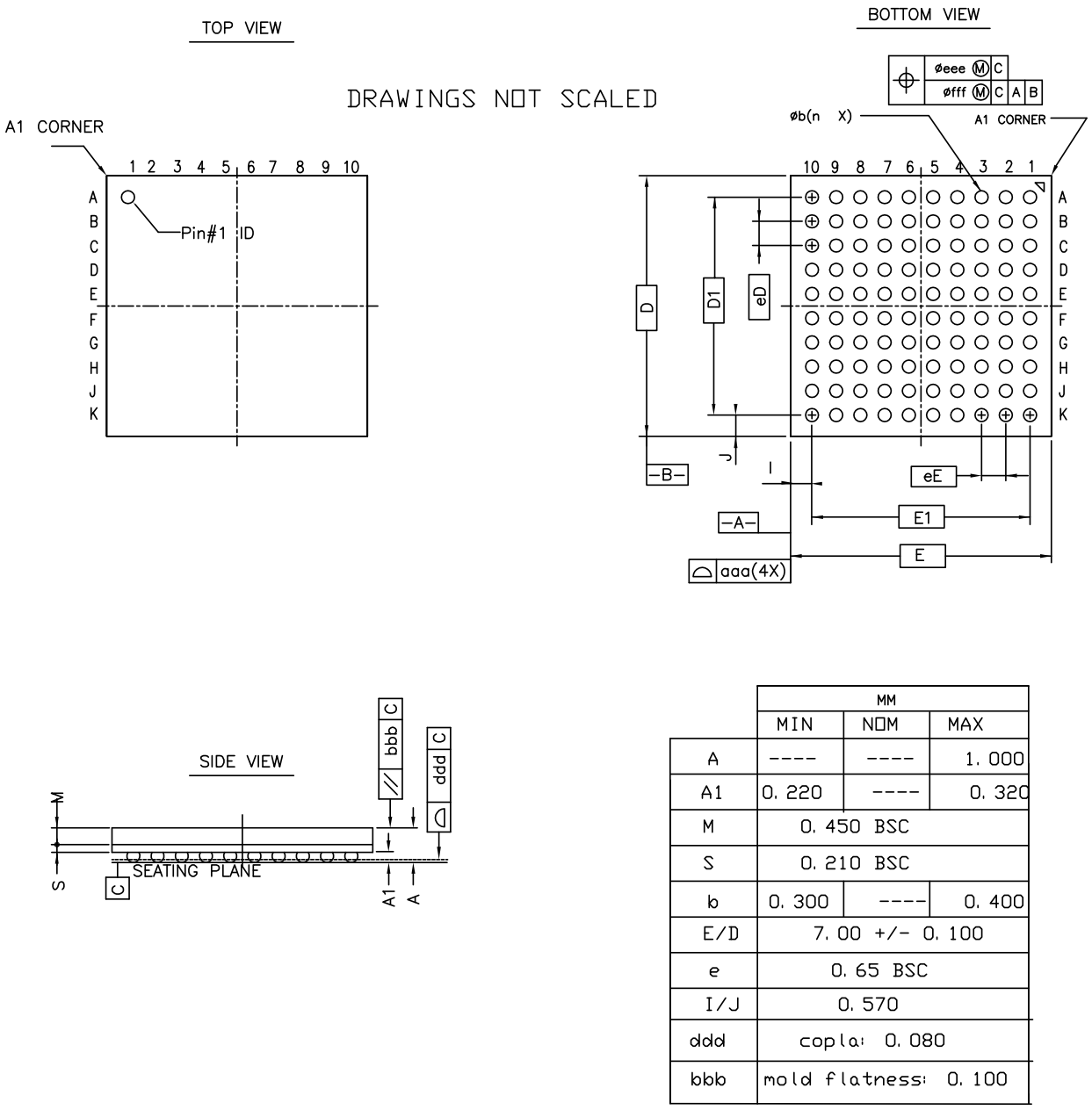
1. $T_J = T_A + (P_D \times \theta_{JA})$
2. $T_J = T_A + (P_D \times (\theta_{HEATSINK} + \theta_{JC}))$

where:

- θ_{JA} = package thermal resistance, Junction-to-ambient (°C/W), provided in Table 8-1 on page 68.
- θ_{JC} = package thermal resistance, Junction-to-case thermal resistance (°C/W), provided in Table 8-1 on page 68.
- $\theta_{HEAT\ SINK}$ = cooling device thermal resistance (°C/W), provided in the device datasheet.
- P_D = device power consumption (W) estimated from data provided in the section "Regulator characteristics" on page 43.
- 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.

Figure 8-3. VFBGA-100 package drawing



	MM		
	MIN	NOM	MAX
A	----	----	1.000
A1	0.220	----	0.320
M	0.450 BSC		
S	0.210 BSC		
b	0.300	----	0.400
E/D	7.00 +/- 0.100		
e	0.65 BSC		
I/J	0.570		
ddd	copla: 0.080		
bbb	mold flatness: 0.100		

9. Ordering Information

Device	Ordering Code	Package	Conditioning	Temperature Operating Range
AT32UC3A3256S	AT32UC3A3256S-ALUT	144-lead LQFP	Tray	Industrial (-40·C to 85·C)
	AT32UC3A3256S-ALUR	144-lead LQFP	Reels	Industrial (-40·C to 85·C)
	AT32UC3A3256S-CTUT	144-ball TFBGA	Tray	Industrial (-40·C to 85·C)
	AT32UC3A3256S-CTUR	144-ball TFBGA	Reels	Industrial (-40·C to 85·C)
AT32UC3A3256	AT32UC3A3256-ALUT	144-lead LQFP	Tray	Industrial (-40·C to 85·C)
	AT32UC3A3256-ALUR	144-lead LQFP	Reels	Industrial (-40·C to 85·C)
	AT32UC3A3256-CTUT	144-ball TFBGA	Tray	Industrial (-40·C to 85·C)
	AT32UC3A3256-CTUR	144-ball TFBGA	Reels	Industrial (-40·C to 85·C)
AT32UC3A3128S	AT32UC3A3128S-ALUT	144-lead LQFP	Tray	Industrial (-40·C to 85·C)
	AT32UC3A3128S-ALUR	144-lead LQFP	Reels	Industrial (-40·C to 85·C)
	AT32UC3A3128S-CTUT	144-ball TFBGA	Tray	Industrial (-40·C to 85·C)
	AT32UC3A3128S-CTUR	144-ball TFBGA	Reels	Industrial (-40·C to 85·C)
AT32UC3A3128	AT32UC3A3128-ALUT	144-lead LQFP	Tray	Industrial (-40·C to 85·C)
	AT32UC3A3128-ALUR	144-lead LQFP	Reels	Industrial (-40·C to 85·C)
	AT32UC3A3128-CTUT	144-ball TFBGA	Tray	Industrial (-40·C to 85·C)
	AT32UC3A3128-CTUR	144-ball TFBGA	Reels	Industrial (-40·C to 85·C)
AT32UC3A364S	AT32UC3A364S-ALUT	144-lead LQFP	Tray	Industrial (-40·C to 85·C)
	AT32UC3A364S-ALUR	144-lead LQFP	Reels	Industrial (-40·C to 85·C)
	AT32UC3A364S-CTUT	144-ball TFBGA	Tray	Industrial (-40·C to 85·C)
	AT32UC3A364S-CTUR	144-ball TFBGA	Reels	Industrial (-40·C to 85·C)
AT32UC3A364	AT32UC3A364-ALUT	144-lead LQFP	Tray	Industrial (-40·C to 85·C)
	AT32UC3A364-ALUR	144-lead LQFP	Reels	Industrial (-40·C to 85·C)
	AT32UC3A364-CTUT	144-ball TFBGA	Tray	Industrial (-40·C to 85·C)
	AT32UC3A364-CTUR	144-ball TFBGA	Reels	Industrial (-40·C to 85·C)
AT32UC3A4256S	AT32UC3A4256S-C1UT	100-ball VFBGA	Tray	Industrial (-40·C to 85·C)
	AT32UC3A4256S-C1UR	100-ball VFBGA	Reels	Industrial (-40·C to 85·C)
AT32UC3A4256	AT32UC3A4256-C1UT	100-ball VFBGA	Tray	Industrial (-40·C to 85·C)
	AT32UC3A4256-C1UR	100-ball VFBGA	Reels	Industrial (-40·C to 85·C)
AT32UC3A4128S	AT32UC3A4128S-C1UT	100-ball VFBGA	Tray	Industrial (-40·C to 85·C)
	AT32UC3A4128S-C1UR	100-ball VFBGA	Reels	Industrial (-40·C to 85·C)
AT32UC3A4128	AT32UC3A4128-C1UT	100-ball VFBGA	Tray	Industrial (-40·C to 85·C)
	AT32UC3A4128-C1UR	100-ball VFBGA	Reels	Industrial (-40·C to 85·C)
AT32UC3A464S	AT32UC3A464S-C1UT	100-ball VFBGA	Tray	Industrial (-40·C to 85·C)
	AT32UC3A464S-C1UR	100-ball VFBGA	Reels	Industrial (-40·C to 85·C)
AT32UC3A464	AT32UC3A464-C1UT	100-ball VFBGA	Tray	Industrial (-40·C to 85·C)
	AT32UC3A464-C1UR	100-ball VFBGA	Reels	Industrial (-40·C to 85·C)

10. Errata

10.1 Rev. H

10.1.1 General

Devices with Date Code lower than 1233 cannot operate with CPU frequency higher than 66MHz in 1WS and 36MHz in 0WS in the whole temperature range
Fix/Workaround
None

DMACA data transfer fails when CTLx.SRC_TR_WIDTH is not equal to CTLx.DST_TR_WIDTH
Fix/Workaround
For any DMACA transfer make sure CTLx.SRC_TR_WIDTH = CTLx.DST_TR_WIDTH.

10.1.2 Processor and Architecture

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.

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.

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.

10.1.3 MPU

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.

10.1.4 USB

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

Fix/Workaround
For higher polling time, the software must freeze the pipe for the desired period in order to prevent any "extra" token.

10.2.5 ADC

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.

10.2.6 USART

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

The LIN ID is not transmitted in mode PDCM='0'
Fix/Workaround
Using USART in mode LIN master with the PDCM bit = '0', the LINID written at the first address of the transmit buffer is not used. The LINID must be written in the LINIR register, after the configuration and start of the PDCA transfer. Writing the LINID in the LINIR register will start the transfer whenever the PDCA transfer is ready.

The LINID interrupt is only available for the header reception and not available for the header transmission
Fix/Workaround
None.

USART LIN mode is not functional with the PDCA if PDCM bit in LINMR register is set to 1
If a PDCA transfer is initiated in USART LIN mode with PDCM bit set to 1, the transfer never starts.
Fix/Workaround
Only use PDCM=0 configuration with the PDCA transfer.

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 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.

ISO7816 Mode T1: RX impossible after any TX
RX impossible after any TX.

10.3.7 SPI

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).

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.

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.

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.

10.3.8 Power Manager

OSC32 not functionnal in Crystal Modes (OSC32CTRL.MODE=1 or OSC32CTRL.MODE=2)
OSC32 clock output is not active even if the oscillation signal is present on XIN32/XOUT32 pins.
OSC32RDY bit may still set even if the CLK32 is not active.
External clock mode (OSC32CTRL.MODE=0) is not affected.

Fix/Workaround
None.

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 will not be turned off. This will result in a significantly higher power consumption during the sleep mode.
Fix/Workaround
Before going to sleep modes where the system RC oscillator is stopped, make sure that the factor between the CPU/HSB and PBx frequencies is less than or equal to 4.

11. 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.

11.1 Rev. H– 10/12

1. Updated max frequency
2. Added Flash Read High Speed Mode description in FLASHC chapter
3. Updated Electrical Characteristics accordingly to new max frequency
4. Fixed wrong description of PLLOPT[0] in PM chapter
5. Updated Errata section according to new maximum frequency
6. Added USB hi-speed PLL electrical characteristics
7. Added OSC32 Errata in Power Management sections for Rev D,E and H

11.2 Rev. G– 11/11

1. Add recommandation for MCI connection with more than 1 slot

11.3 Rev. F – 08/11

1. Final version

11.4 Rev. E – 06/11

1. Updated Errata for E and D
2. Updated FLASHC chapter with HSEN and HSDIS commands

11.5 Rev. D – 04/11

1. Updated Errata for revision H and E
2. Updated Reset Sequence
3. Updated Peripherals' current consumption and others minor electrical charateristics
4. Updated Peripherals chapters

11.6 Rev. C – 03/10

1. Updated the datasheet with new revision H features.