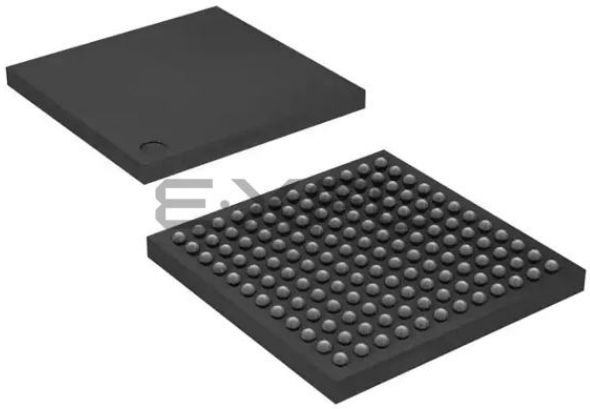


Welcome to E-XFL.COM

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-TFBGA
Supplier Device Package	144-FFBGA (11x11)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/at32uc3a364s-ctur

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.

Table 3-1. GPIO Controller Function Multiplexing

BGA 144	QFP 144	BGA 100	PIN	GPIO	Supply	PIN Type (2)	GPIO function			
							A	B	C	D
C7	14	A4	PA30	30	VDDIO	x1	MCI - DATA[1]	USART3 - CLK	DMACA - DMAACK[0]	MSI - DATA[1]
B3	29	C2	PA31	31	VDDIO	x1	MCI - DATA[2]	USART2 - RXD	DMACA - DMARQ[0]	MSI - DATA[2]
A2	30	B1	PB00	32	VDDIO	x1	MCI - DATA[3]	USART2 - TXD	ADC - TRIGGER	MSI - DATA[3]
C4	27	B2	PB01	33	VDDIO	x1	MCI - DATA[4]	ABDAC - DATA[1]	EIC - SCAN[0]	MSI - INS
B4	25	B3	PB02	34	VDDIO	x1	MCI - DATA[5]	ABDAC - DATAN[1]	EIC - SCAN[1]	
A5	24	C4	PB03	35	VDDIO	x1	MCI - DATA[6]	USART2 - CLK	EIC - SCAN[2]	
B6	22	A3	PB04	36	VDDIO	x1	MCI - DATA[7]	USART3 - RXD	EIC - SCAN[3]	
H12	121	F7(1)	PB05	37	VDDIO	x3	USB - ID	TC0 - A0	EIC - SCAN[4]	
D12	134	D7	PB06	38	VDDIO	x1	USB - VBOF	TC0 - B0	EIC - SCAN[5]	
D11	135	D6	PB07	39	VDDIO	x3	SPI1 - SPCK	SSC - TX_CLOCK	EIC - SCAN[6]	
C8	11	C6	PB08	40	VDDIO	x2	SPI1 - MISO	SSC - TX_DATA	EIC - SCAN[7]	
E7	17	C5	PB09	41	VDDIO	x2	SPI1 - NPCS[0]	SSC - RX_DATA	EBI - NCS[4]	
D7	16	D5	PB10	42	VDDIO	x2	SPI1 - MOSI	SSC - RX_FRAME_SYNC	EBI - NCS[5]	
B2	31	C1	PB11	43	VDDIO	x1	USART1 - RXD	SSC - TX_FRAME_SYNC	PM - GCLK[1]	
K5	98	K5(1)	PC00	45	VDDIO	x1				
H6	99	K6	PC01	46	VDDIO	x1				
A7	18	A5	PC02	47	VDDIO	x1				
B7	19	A6	PC03	48	VDDIO	x1				
A8	13	B7	PC04	49	VDDIO	x1				
A9	12	A7	PC05	50	VDDIO	x1				
G1	55	G4	PX00	51	VDDIO	x2	EBI - DATA[10]	USART0 - RXD	USART1 - RI	
H1	59	G2	PX01	52	VDDIO	x2	EBI - DATA[9]	USART0 - TXD	USART1 - DTR	
J2	62	G3	PX02	53	VDDIO	x2	EBI - DATA[8]	USART0 - CTS	PM - GCLK[0]	
K1	63	J1	PX03	54	VDDIO	x2	EBI - DATA[7]	USART0 - RTS		
J1	60	H1	PX04	55	VDDIO	x2	EBI - DATA[6]	USART1 - RXD		
G2	58	G1	PX05	56	VDDIO	x2	EBI - DATA[5]	USART1 - TXD		
F3	53	F3	PX06	57	VDDIO	x2	EBI - DATA[4]	USART1 - CTS		
F2	54	F4	PX07	58	VDDIO	x2	EBI - DATA[3]	USART1 - RTS		
D1	50	E3	PX08	59	VDDIO	x2	EBI - DATA[2]	USART3 - RXD		
C1	49	E4	PX09	60	VDDIO	x2	EBI - DATA[1]	USART3 - TXD		
B1	37	D2	PX10	61	VDDIO	x2	EBI - DATA[0]	USART2 - RXD		
L1	67	K7(1)	PX11	62	VDDIO	x2	EBI - NWE1	USART2 - TXD		
D6	34	D1	PX12	63	VDDIO	x2	EBI - NWE0	USART2 - CTS	MCI - CLK	
C6	33	D3	PX13	64	VDDIO	x2	EBI - NRD	USART2 - RTS	MCI - CLK	
M4	68	K5(1)	PX14	65	VDDIO	x2	EBI - NCS[1]		TC0 - A0	
E6	40	K4(1)	PX15	66	VDDIO	x2	EBI - ADDR[19]	USART3 - RTS	TC0 - B0	
C5	32	D4(1)	PX16	67	VDDIO	x2	EBI - ADDR[18]	USART3 - CTS	TC0 - A1	
K6	83	J10(1)	PX17	68	VDDIO	x2	EBI - ADDR[17]	DMACA - DMARQ[1]	TC0 - B1	

Table 3-6. Signal Description List

Signal Name	Function	Type	Active Level	Comments
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 - TWI0, TWI1				
TWCK	Serial Clock	I/O		
TWD	Serial Data	I/O		
TWALM	SMBALERT signal	I/O		
Universal Synchronous Asynchronous Receiver Transmitter - USART0, USART1, USART2, USART3				
CLK	Clock	I/O		
CTS	Clear To Send	Input		
DCD	Data Carrier Detect			Only USART1
DSR	Data Set Ready			Only USART1
DTR	Data Terminal Ready			Only USART1
RI	Ring Indicator			Only USART1
RTS	Request To Send	Output		
RXD	Receive Data	Input		
TXD	Transmit Data	Output		
Analog to Digital Converter - ADC				
AD0 - AD7	Analog input pins	Analog input		
Audio Bitstream DAC (ABDAC)				
DATA0-DATA1	D/A Data out	Output		
DATAN0-DATAN1	D/A Data inverted out	Output		
Universal Serial Bus Device - USB				
DMFS	USB Full Speed Data -	Analog		
DPFS	USB Full Speed Data +	Analog		

4. Processor and Architecture

Rev: 1.4.2.0

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 extention with saturating arithmetic, and a wide variety of multiply instructions**
- **3-stage pipeline allows 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**

4.2 AVR32 Architecture

AVR32 is a 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 micro-architectures, 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 single cycle. Load and store instructions have several different formats in order to reduce code size and speed up execution.

All interrupt levels are by default disabled when debug state is entered, but they can individually be switched on by the monitor routine by clearing the respective mask bit in the status register.

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

Debug state is exited by the *retd* instruction.

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
24	96	JAVA_LV1	Unused in AVR32UC
25	100	JAVA_LV2	Unused in AVR32UC

The user must also make sure that the system stack is large enough so that any event is able to push the required registers to stack. If the system stack is full, and an event occurs, the system will enter an UNDEFINED state.

4.5.2 Exceptions and Interrupt Requests

When an event other than *scall* or debug request is received by the core, the following actions are performed atomically:

1. The pending event will not be accepted if it is masked. The I3M, I2M, I1M, I0M, EM, and GM bits in the Status Register are used to mask different events. Not all events can be masked. A few critical events (NMI, Unrecoverable Exception, TLB Multiple Hit, and Bus Error) can not be masked. When an event is accepted, hardware automatically sets the mask bits corresponding to all sources with equal or lower priority. This inhibits acceptance of other events of the same or lower priority, except for the critical events listed above. Software may choose to clear some or all of these bits after saving the necessary state if other priority schemes are desired. It is the event source's responsability to ensure that their events are left pending until accepted by the CPU.
2. When a request is accepted, the Status Register and Program Counter of the current context is stored to the system stack. If the event is an INT0, INT1, INT2, or INT3, registers R8-R12 and LR are also automatically stored to stack. Storing the Status Register ensures that the core is returned to the previous execution mode when the current event handling is completed. When exceptions occur, both the EM and GM bits are set, and the application may manually enable nested exceptions if desired by clearing the appropriate bit. Each exception handler has a dedicated handler address, and this address uniquely identifies the exception source.
3. The Mode bits are set to reflect the priority of the accepted event, and the correct register file bank is selected. The address of the event handler, as shown in Table 4-4, is loaded into the Program Counter.

The execution of the event handler routine then continues from the effective address calculated.

The *rete* instruction signals the end of the event. When encountered, the Return Status Register and Return Address Register are popped from the system stack and restored to the Status Register and Program Counter. If the *rete* instruction returns from INT0, INT1, INT2, or INT3, registers R8-R12 and LR are also popped from the system stack. The restored Status Register contains information allowing the core to resume operation in the previous execution mode. This concludes the event handling.

4.5.3 Supervisor Calls

The AVR32 instruction set provides a supervisor mode call instruction. The *scall* instruction is designed so that privileged routines can be called from any context. This facilitates sharing of code between different execution modes. The *scall* mechanism is designed so that a minimal execution cycle overhead is experienced when performing supervisor routine calls from time-critical event handlers.

The *scall* instruction behaves differently depending on which mode it is called from. The behaviour is detailed in the instruction set reference. In order to allow the *scall* routine to return to the correct context, a return from supervisor call instruction, *rets*, is implemented. In the AVR32UC CPU, *scall* and *rets* uses the system stack to store the return address and the status register.

4.5.4 Debug Requests

The AVR32 architecture defines a dedicated Debug mode. When a debug request is received by the core, Debug mode is entered. Entry into Debug mode can be masked by the DM bit in the

Table 5-3. Local Bus Mapped GPIO Registers

Port	Register	Mode	Local Bus Address	Access
2	Output Driver Enable Register (ODER)	WRITE	0x40000240	Write-only
		SET	0x40000244	Write-only
		CLEAR	0x40000248	Write-only
		TOGGLE	0x4000024C	Write-only
	Output Value Register (OVR)	WRITE	0x40000250	Write-only
		SET	0x40000254	Write-only
		CLEAR	0x40000258	Write-only
		TOGGLE	0x4000025C	Write-only
	Pin Value Register (PVR)	-	0x40000260	Read-only
3	Output Driver Enable Register (ODER)	WRITE	0x40000340	Write-only
		SET	0x40000344	Write-only
		CLEAR	0x40000348	Write-only
		TOGGLE	0x4000034C	Write-only
	Output Value Register (OVR)	WRITE	0x40000350	Write-only
		SET	0x40000354	Write-only
		CLEAR	0x40000358	Write-only
		TOGGLE	0x4000035C	Write-only
	Pin Value Register (PVR)	-	0x40000360	Read-only

7. Electrical Characteristics

7.1 Absolute Maximum Ratings*

Operating Temperature	-40°C to +85°C
Storage Temperature	-60°C to +150°C
Voltage on Input Pin with respect to Ground	-0.3V to 3.6V
Maximum Operating Voltage (VDDCORE)	1.95V
Maximum Operating Voltage (VDDIO)	3.6V
Total DC Output Current on all I/O Pin	
for TQFP144 package	370 mA
for TFBGA144 package	370 mA

*NOTICE: Stresses beyond those listed under “Absolute Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

7.2.1 I/O Pin Output Level Typical Characteristics

Figure 7-1. I/O Pin drive x2 Output Low Level Voltage (VOL) vs. Source Current

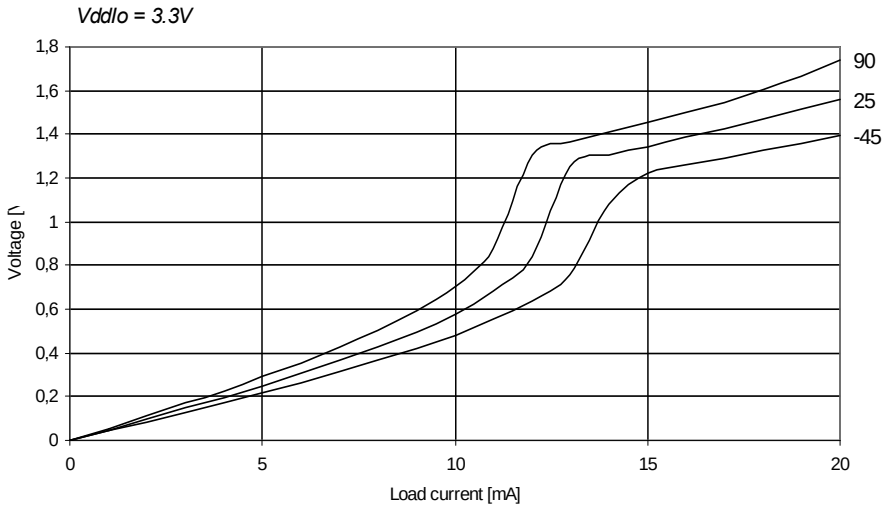
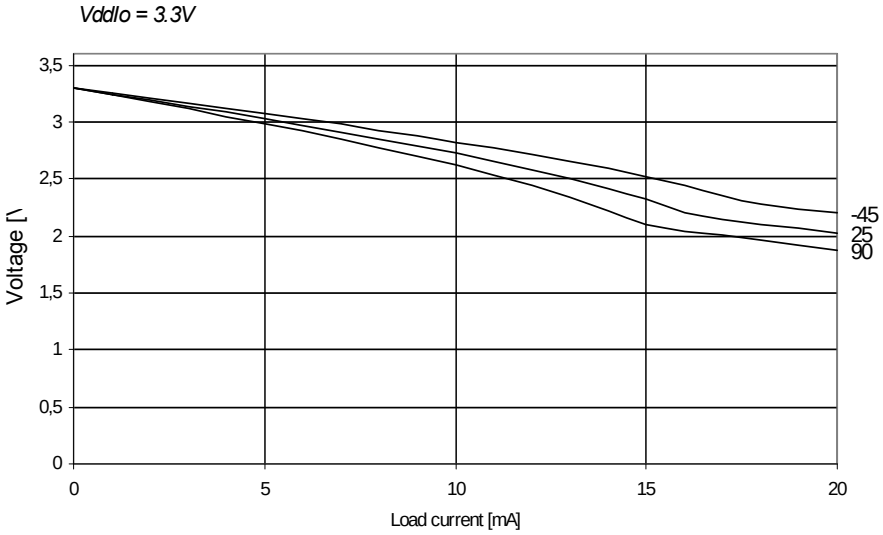


Figure 7-2. I/O Pin drive x2 Output High Level Voltage (VOH) vs. Source Current



7.3 I/O pin Characteristics

These parameters are given in the following conditions:

- V_{DDCORE} = 1.8V
- V_{DDIO} = 3.3V
- Ambient Temperature = 25°C

7.5.4 RESET_N Characteristics

Table 7-11. RESET_N Waveform Parameters

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

7.7 System Clock Characteristics

These parameters are given in the following conditions:

- V_{DDCORE} = 1.8V

7.7.1 CPU/HSB Clock Characteristics

Table 7-14. Core Clock Waveform Parameters

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Unit
$1/(t_{CPCPU})$	CPU Clock Frequency	-40°C < Ambient Temperature < 70°C			84	MHz
$1/(t_{CPCPU})$	CPU Clock Frequency	-40°C < Ambient Temperature < 85°C			66	MHz

7.7.2 PBA Clock Characteristics

Table 7-15. PBA Clock Waveform Parameters

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Unit
$1/(t_{CPPBA})$	PBA Clock Frequency	-40°C < Ambient Temperature < 70°C			84	MHz
$1/(t_{CPPBA})$	PBA Clock Frequency	-40°C < Ambient Temperature < 85°C			66	MHz

7.7.3 PBB Clock Characteristics

Table 7-16. PBB Clock Waveform Parameters

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Unit
$1/(t_{CPPBB})$	PBB Clock Frequency	-40°C < Ambient Temperature < 70°C			84	MHz
$1/(t_{CPPBB})$	PBB Clock Frequency	-40°C < Ambient Temperature < 85°C			66	MHz

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

Table 7-26. 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 = 5 MHz	-2		2	LSB

7.10 USB Transceiver Characteristics

7.10.1 Electrical Characteristics

Table 7-27. Electrical Parameters

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Unit
R_{EXT}	Recommended External USB Series Resistor	In series with each USB pin with ±5%		39		Ω
R_{BIAS}	VBIAS External Resistor (1)	±1%		6810		Ω
C_{BIAS}	VBIAS External Capcitor			10		pF

1. The USB on-chip buffers comply with the Universal Serial Bus (USB) v2.0 standard. All AC parameters related to these buffers can be found within the USB 2.0 electrical specifications.

7.10.2 Static Power Consumption

Table 7-28. Static Power Consumption

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Unit
I_{BIAS}	Bias current consumption on VBG				1	µA
$I_{VDDUTMI}$	HS Transceiver and I/O current consumption				8	µA
	FS/HS Transceiver and I/O current consumption	If cable is connected, add 200µA (typical) due to Pull-up/Pull-down current consumption			3	µA

7.10.3 Dynamic Power Consumption

Table 7-29. Dynamic Power Consumption

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Unit
I_{BIAS}	Bias current consumption on VBG			0.7	0.8	mA

Table 7-34. SMC Write Signals with No Hold Settings (NWE Controlled only)

Symbol	Parameter	Min.	Unit
SMC_{43}	Data Out Valid before NWE Rising	(nwe pulse length - 1) * t_{CPSMC} - 1.2	ns
SMC_{44}	Data Out Valid after NWE Rising	5	ns
SMC_{45}	NWE Pulse Width	nwe pulse length * t_{CPSMC} - 0.9	ns

Figure 7-7. SMC Signals for NCS Controlled Accesses.

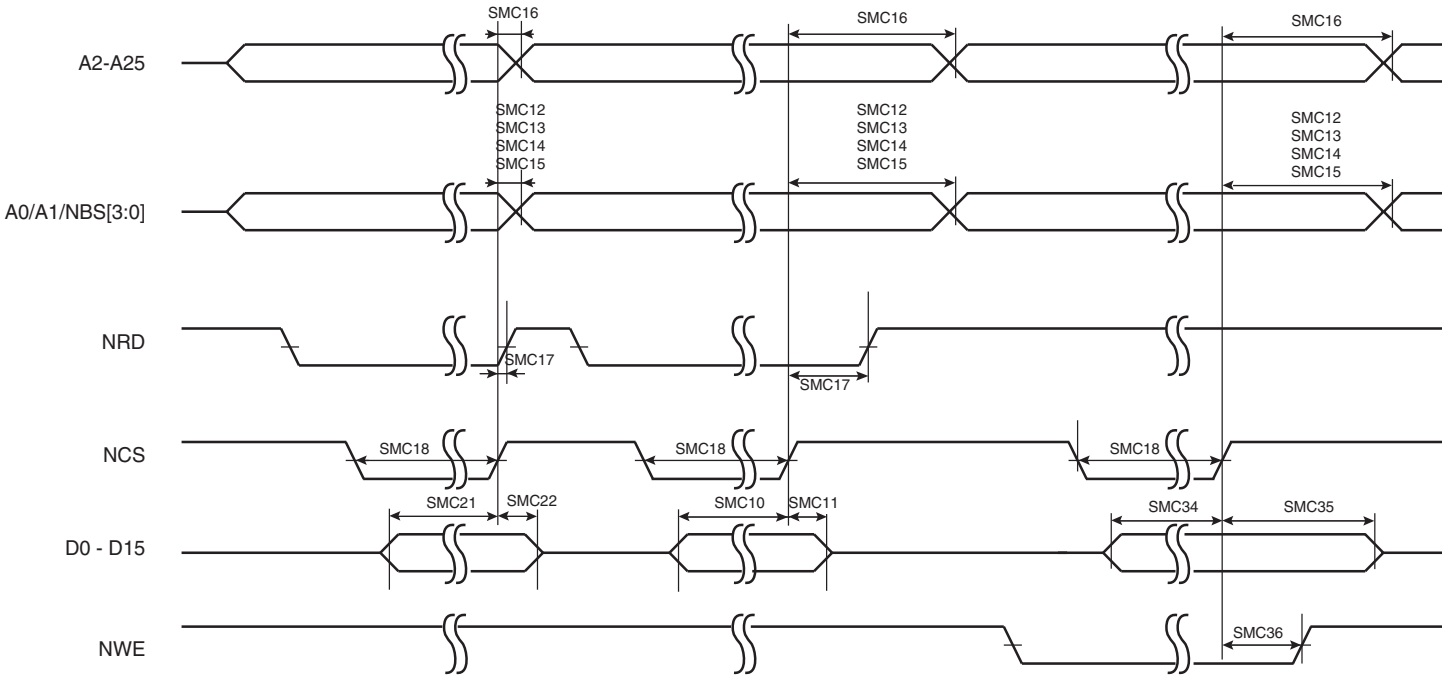
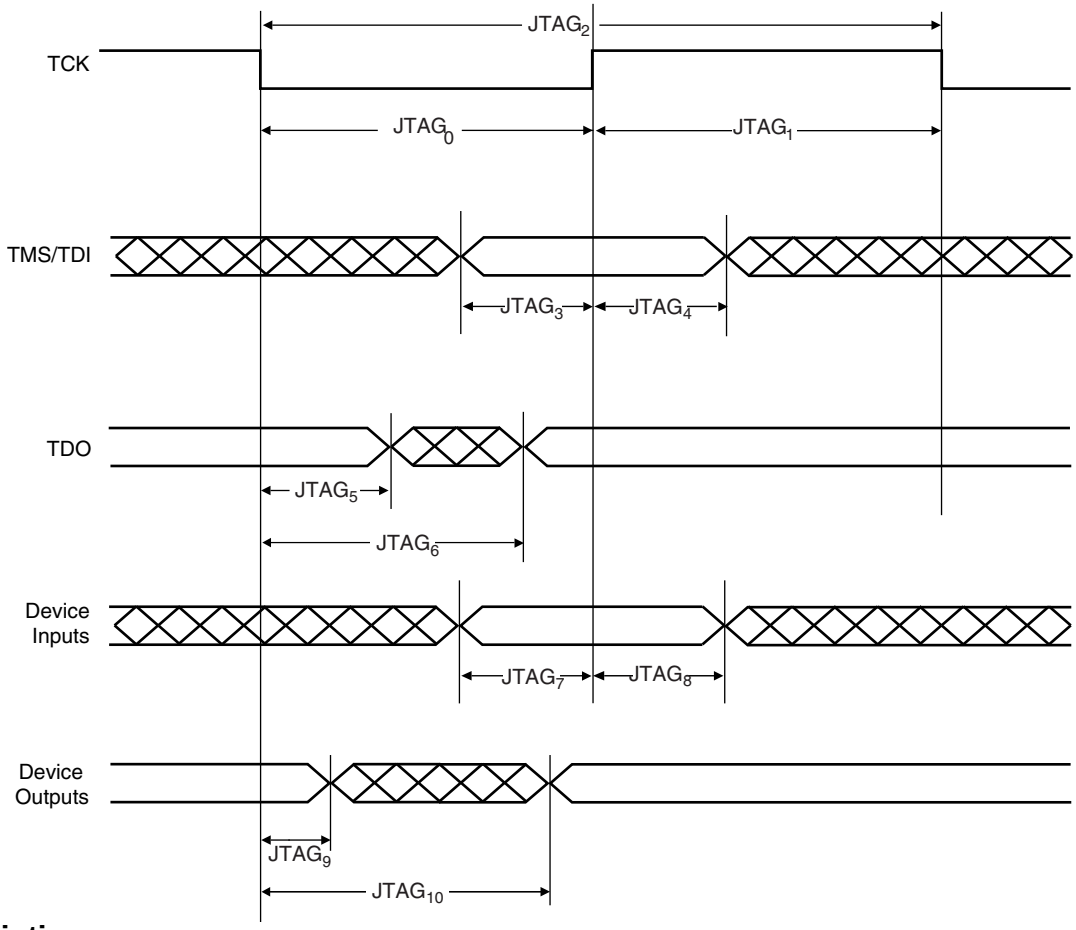


Figure 7-10. JTAG Interface Signals



7.13 SPI Characteristics

Figure 7-11. SPI Master mode with (CPOL= NCPHA= 0) or (CPOL= NCPHA= 1)

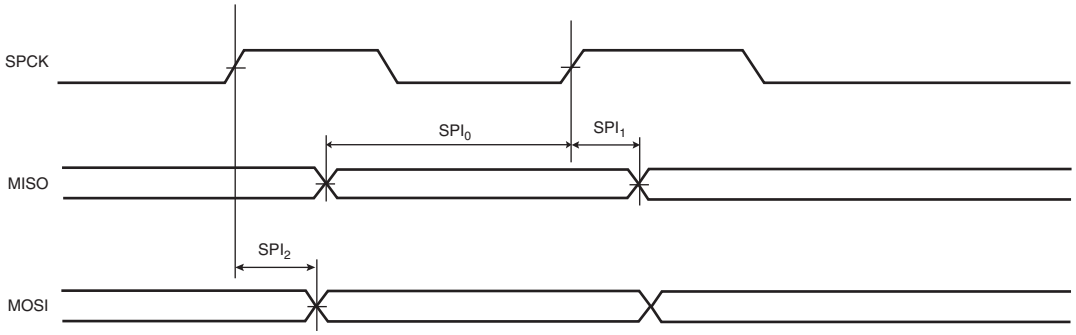


Table 7-38. SPI Timings

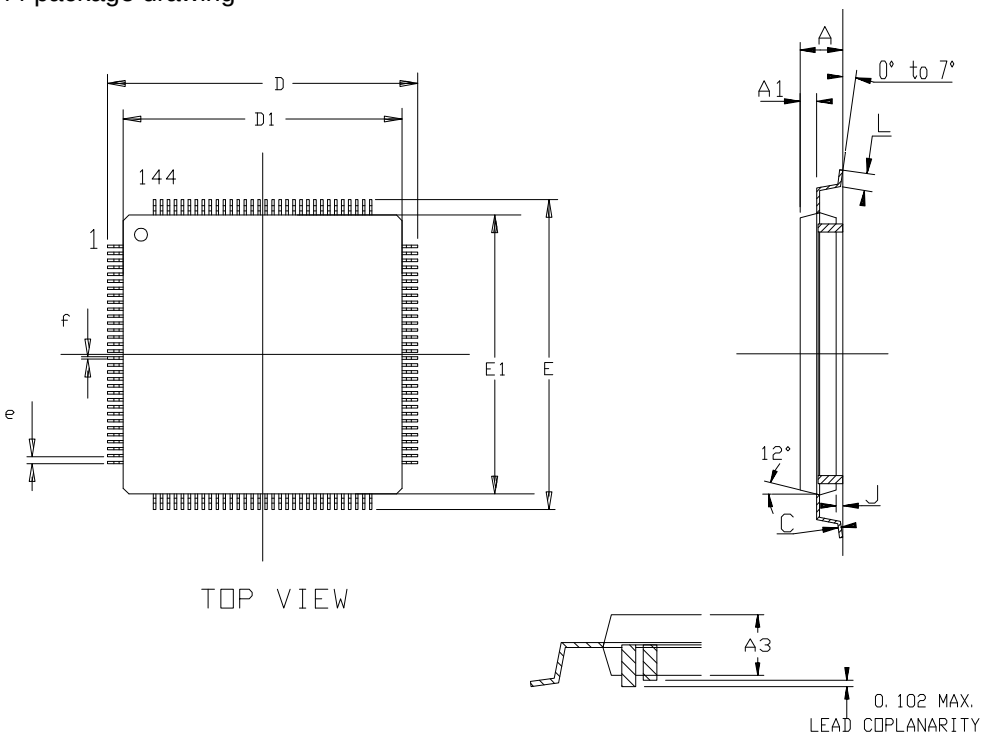
Symbol	Parameter	Conditions [1]	Min.	Max.	Unit
SPI_0	MISO Setup time before SPCK rises (master)	3.3V domain	22 + $(t_{CPMCK})/2$ [2]		ns
SPI_1	MISO Hold time after SPCK rises (master)	3.3V domain	0		ns
SPI_2	SPCK rising to MOSI Delay (master)	3.3V domain		7	ns
SPI_3	MISO Setup time before SPCK falls (master)	3.3V domain	22 + $(t_{CPMCK})/2$ [3]		ns
SPI_4	MISO Hold time after SPCK falls (master)	3.3V domain	0		ns
SPI_5	SPCK falling to MOSI Delay master)	3.3V domain		7	ns
SPI_6	SPCK falling to MISO Delay (slave)	3.3V domain		26.5	ns
SPI_7	MOSI Setup time before SPCK rises (slave)	3.3V domain	0		ns
SPI_8	MOSI Hold time after SPCK rises (slave)	3.3V domain	1.5		ns
SPI_9	SPCK rising to MISO Delay (slave)	3.3V domain		27	ns
SPI_{10}	MOSI Setup time before SPCK falls (slave)	3.3V domain	0		ns
SPI_{11}	MOSI Hold time after SPCK falls (slave)	3.3V domain	1		ns

1. 3.3V domain: V_{VDDIO} from 3.0V to 3.6V, maximum external capacitor = 40 pF
2. t_{CPMCK}: Master Clock period in ns.
3. t_{CPMCK}: Master Clock period in ns.

7.14 MCI

The High Speed MultiMedia Card Interface (MCI) supports the MultiMedia Card (MMC) Specification V4.2, the SD Memory Card Specification V2.0, the SDIO V1.1 specification and CE-ATA V1.1.

Figure 8-2. LQFP-144 package drawing



	MM			INCH		
	Min	Nom	Max	Min	Nom	Max
A	–	–	1.60	–	–	.063
C	0.09	–	0.20	.004	–	.008
A3	1.35	1.40	1.45	.053	.055	.057
D	21.90	22.00	22.10	.862	.866	.870
D1	19.90	20.00	20.10	.783	.787	.791
E	21.90	22.00	22.10	.862	.866	.870
E1	19.90	20.00	20.10	.783	.787	.791
J	0.05	–	0.15	.002	–	.006
L	0.45	0.60	0.75	.018	.024	.030
e	0.50 BSC			.0197 BSC		
f	0.22 BSC			.009 BSC		

Table 8-2. Device and Package Maximum Weight

1300	mg

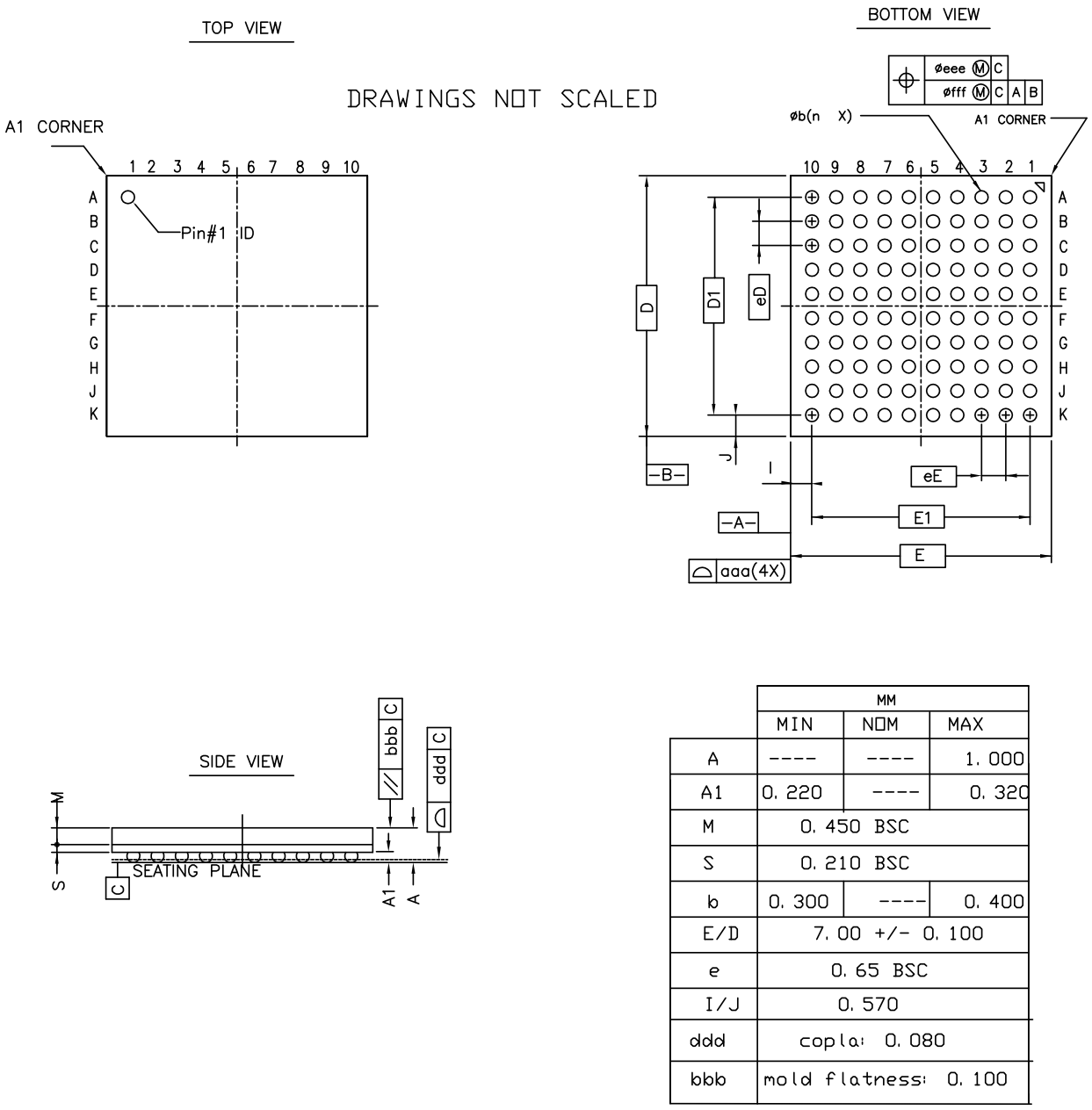
Table 8-3. Package Characteristics

Moisture Sensitivity Level	MSL3

Table 8-4. Package Reference

JEDEC Drawing Reference	MS-026
JESD97 Classification	E3

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		

SMBALERT bit may be set after reset
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.

10.2.13 TWIS

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.

TWIS stretch on Address match error
When the TWIS stretches TWCK due to a slave address match, it also holds TWD low for the same duration if it is to be receiving data. When TWIS releases TWCK, it releases TWD at the same time. This can cause a TWI timing violation.
Fix/Workaround
None.

10.2.14 MCI

MCI_CLK features is not available on PX12, PX13 and PX40
Fix/Workaround
MCI_CLK feature is available on PA27 only.

The busy signal of the responses R1b is not taken in account for CMD12 STOP_TRANSFER
It is not possible to know the busy status of the card during the response (R1b) for the commands CMD12.
Fix/Workaround
The card busy line should be polled through the GPIO Input Value register (IVR) for commands CMD12.

10.2.15 SSC

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.

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.