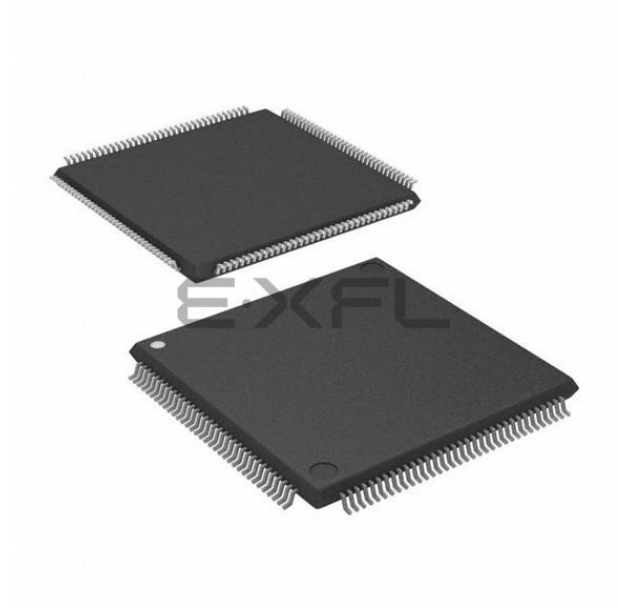


Welcome to E-XFL.COM

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, Ethernet, I²C, SPI, SSC, UART/USART, USB OTG
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	109
Program Memory Size	512KB (512K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	64K 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	144-LQFP
Supplier Device Package	144-LQFP (20x20)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/at32uc3a0512-alut

6. Package and Pinout

The device pins are multiplexed with peripheral functions as described in ”Peripheral Multiplexing on I/O lines” on page 31.

Figure 6-1. TQFP100 Pinout

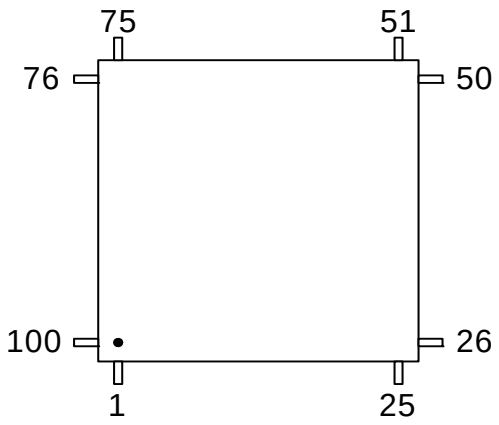


Table 6-1. TQFP100 Package Pinout

1	PB20	**26**	PA05	**51**	PA21	**76**	PB08
2	PB21	**27**	PA06	**52**	PA22	**77**	PB09
3	PB22	**28**	PA07	**53**	PA23	**78**	PB10
4	VDDIO	**29**	PA08	**54**	PA24	**79**	VDDIO
5	GND	**30**	PA09	**55**	PA25	**80**	GND
6	PB23	**31**	PA10	**56**	PA26	**81**	PB11
7	PB24	**32**	N/C	**57**	PA27	**82**	PB12
8	PB25	**33**	PA11	**58**	PA28	**83**	PA29
9	PB26	**34**	VDDCORE	**59**	VDDANA	**84**	PA30
10	PB27	**35**	GND	**60**	ADVREF	**85**	PC02
11	VDDOUT	**36**	PA12	**61**	GNDANA	**86**	PC03
12	VDDIN	**37**	PA13	**62**	VDDPLL	**87**	PB13
13	GND	**38**	VDDCORE	**63**	PC00	**88**	PB14
14	PB28	**39**	PA14	**64**	PC01	**89**	TMS
15	PB29	**40**	PA15	**65**	PB00	**90**	TCK
16	PB30	**41**	PA16	**66**	PB01	**91**	TDO
17	PB31	**42**	PA17	**67**	VDDIO	**92**	TDI
18	RESET_N	**43**	PA18	**68**	VDDIO	**93**	PC04
19	PA00	**44**	PA19	**69**	GND	**94**	PC05
20	PA01	**45**	PA20	**70**	PB02	**95**	PB15
21	GND	**46**	VBUS	**71**	PB03	**96**	PB16
22	VDDCORE	**47**	VDDIO	**72**	PB04	**97**	VDDCORE

Table 10-2. Local bus mapped GPIO registers

Port	Register	Mode	Local Bus Address	Access
3	Output Driver Enable Register (ODER)	WRITE	0x4000_0340	Write-only
		SET	0x4000_0344	Write-only
		CLEAR	0x4000_0348	Write-only
		TOGGLE	0x4000_034C	Write-only
	Output Value Register (OVR)	WRITE	0x4000_0350	Write-only
		SET	0x4000_0354	Write-only
		CLEAR	0x4000_0358	Write-only
		TOGGLE	0x4000_035C	Write-only
	Pin Value Register (PVR)	-	0x4000_0360	Read-only

10.3 Interrupt Request Signal Map

The various modules may output Interrupt request signals. These signals are routed to the Interrupt Controller (INTC), described in a later chapter. The Interrupt Controller supports up to 64 groups of interrupt requests. Each group can have up to 32 interrupt request signals. All interrupt signals in the same group share the same autovector address and priority level. Refer to the documentation for the individual submodules for a description of the semantics of the different interrupt requests.

The interrupt request signals are connected to the INTC as follows.

Table 10-3. Interrupt Request Signal Map

Group	Line	Module	Signal
0	0	AVR32 UC CPU with optional MPU and optional OCD	SYSBLOCK COMPARE
1	0	External Interrupt Controller	EIC 0
	1	External Interrupt Controller	EIC 1
	2	External Interrupt Controller	EIC 2
	3	External Interrupt Controller	EIC 3
	4	External Interrupt Controller	EIC 4
	5	External Interrupt Controller	EIC 5
	6	External Interrupt Controller	EIC 6
	7	External Interrupt Controller	EIC 7
	8	Real Time Counter	RTC
	9	Power Manager	PM
	10	Frequency Meter	FREQM

Table 10-3. Interrupt Request Signal Map

2	0	General Purpose Input/Output	GPIO 0
	1	General Purpose Input/Output	GPIO 1
	2	General Purpose Input/Output	GPIO 2
	3	General Purpose Input/Output	GPIO 3
	4	General Purpose Input/Output	GPIO 4
	5	General Purpose Input/Output	GPIO 5
	6	General Purpose Input/Output	GPIO 6
	7	General Purpose Input/Output	GPIO 7
	8	General Purpose Input/Output	GPIO 8
	9	General Purpose Input/Output	GPIO 9
	10	General Purpose Input/Output	GPIO 10
	11	General Purpose Input/Output	GPIO 11
	12	General Purpose Input/Output	GPIO 12
	13	General Purpose Input/Output	GPIO 13
3	0	Peripheral DMA Controller	PDCA 0
	1	Peripheral DMA Controller	PDCA 1
	2	Peripheral DMA Controller	PDCA 2
	3	Peripheral DMA Controller	PDCA 3
	4	Peripheral DMA Controller	PDCA 4
	5	Peripheral DMA Controller	PDCA 5
	6	Peripheral DMA Controller	PDCA 6
	7	Peripheral DMA Controller	PDCA 7
	8	Peripheral DMA Controller	PDCA 8
	9	Peripheral DMA Controller	PDCA 9
	10	Peripheral DMA Controller	PDCA 10
	11	Peripheral DMA Controller	PDCA 11
	12	Peripheral DMA Controller	PDCA 12
	13	Peripheral DMA Controller	PDCA 13
	14	Peripheral DMA Controller	PDCA 14
4	0	Flash Controller	FLASHC
5	0	Universal Synchronous/Asynchronous Receiver/Transmitter	USART0
6	0	Universal Synchronous/Asynchronous Receiver/Transmitter	USART1
7	0	Universal Synchronous/Asynchronous Receiver/Transmitter	USART2
8	0	Universal Synchronous/Asynchronous Receiver/Transmitter	USART3

10.4.3 SPIs

Each SPI can be connected to an internally divided clock:

Table 10-6. SPI clock connections

SPI	Source	Name	Connection
0	Internal	CLK_DIV	PBA clock or PBA clock / 32
1			

10.5 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 10-7. Nexus OCD AUX port connections

Pin	AXS=0	AXS=1
EVTI_N	PB19	PA08
MDO[5]	PB16	PA27
MDO[4]	PB14	PA26
MDO[3]	PB13	PA25
MDO[2]	PB12	PA24
MDO[1]	PB11	PA23
MDO[0]	PB10	PA22
EVTO_N	PB20	PB20
MCKO	PB21	PA21
MSEO[1]	PB04	PA07
MSEO[0]	PB17	PA28

10.6 PDC handshake signals

The PDC and the peripheral modules communicate through a set of handshake signals. The following table defines the valid settings for the Peripheral Identifier (PID) in the PDC Peripheral Select Register (PSR).

Table 10-8. PDC Handshake Signals

PID Value	Peripheral module & direction
0	ADC
1	SSC - RX
2	USART0 - RX
3	USART1 - RX

Table 10-8. PDC Handshake Signals

PID Value	Peripheral module & direction
4	USART2 - RX
5	USART3 - RX
6	TWI - RX
7	SPI0 - RX
8	SPI1 - RX
9	SSC - TX
10	USART0 - TX
11	USART1 - TX
12	USART2 - TX
13	USART3 - TX
14	TWI - TX
15	SPI0 - TX
16	SPI1 - TX
17	ABDAC

10.7 Peripheral Multiplexing on I/O lines

Each GPIO line can be assigned to one of 3 peripheral functions; A, B or C. The following table define how the I/O lines on the peripherals A, B and C are multiplexed by the GPIO.

Table 10-9. GPIO Controller Function Multiplexing

TQFP100	VQFP144	PIN	GPIO Pin	Function A	Function B	Function C
19	25	PA00	GPIO 0	USART0 - RXD	TC - CLK0	
20	27	PA01	GPIO 1	USART0 - TXD	TC - CLK1	
23	30	PA02	GPIO 2	USART0 - CLK	TC - CLK2	
24	32	PA03	GPIO 3	USART0 - RTS	EIM - EXTINT[4]	DAC - DATA[0]
25	34	PA04	GPIO 4	USART0 - CTS	EIM - EXTINT[5]	DAC - DATAN[0]
26	39	PA05	GPIO 5	USART1 - RXD	PWM - PWM[4]	
27	41	PA06	GPIO 6	USART1 - TXD	PWM - PWM[5]	
28	43	PA07	GPIO 7	USART1 - CLK	PM - GCLK[0]	SPI0 - NPCS[3]
29	45	PA08	GPIO 8	USART1 - RTS	SPI0 - NPCS[1]	EIM - EXTINT[7]
30	47	PA09	GPIO 9	USART1 - CTS	SPI0 - NPCS[2]	MACB - WOL
31	48	PA10	GPIO 10	SPI0 - NPCS[0]	EIM - EXTINT[6]	
33	50	PA11	GPIO 11	SPI0 - MISO	USB - USB_ID	
36	53	PA12	GPIO 12	SPI0 - MOSI	USB - USB_VBOF	
37	54	PA13	GPIO 13	SPI0 - SCK		
39	56	PA14	GPIO 14	SSC - TX_FRAME_SYNC	SPI1 - NPCS[0]	EBI - NCS[0]
40	57	PA15	GPIO 15	SSC - TX_CLOCK	SPI1 - SCK	EBI - ADDR[20]

10.10 GPIO

The GPIO open drain feature (GPIO ODMER register (Open Drain Mode Enable Register)) is not available for this device.

10.11 Peripheral overview

10.11.1 External Bus Interface

- **Optimized for Application Memory Space support**
- **Integrates Two External Memory Controllers:**
  - **Static Memory Controller**
  - **SDRAM Controller**
- **Optimized External Bus:**
  - **16-bit Data Bus**
  - **24-bit Address Bus, Up to 16-Mbytes Addressable**
  - **Optimized pin multiplexing to reduce latencies on External Memories**
- **4 SRAM Chip Selects, 1SDRAM Chip Select:**
  - **Static Memory Controller on NCS0**
  - **SDRAM Controller or Static Memory Controller on NCS1**
  - **Static Memory Controller on NCS2**
  - **Static Memory Controller on NCS3**

10.11.2 Static Memory Controller

- **4 Chip Selects Available**
- **64-Mbyte Address Space per Chip Select**
- **8-, 16-bit Data Bus**
- **Word, Halfword, Byte Transfers**
- **Byte Write or Byte Select Lines**
- **Programmable Setup, Pulse And Hold Time for Read Signals per Chip Select**
- **Programmable Setup, Pulse And Hold Time for Write Signals per Chip Select**
- **Programmable Data Float Time per Chip Select**
- **Compliant with LCD Module**
- **External Wait Request**
- **Automatic Switch to Slow Clock Mode**
- **Asynchronous Read in Page Mode Supported: Page Size Ranges from 4 to 32 Bytes**

10.11.3 SDRAM Controller

- **Numerous Configurations Supported**
  - **2K, 4K, 8K Row Address Memory Parts**
  - **SDRAM with Two or Four Internal Banks**
  - **SDRAM with 16-bit Data Path**
- **Programming Facilities**
  - **Word, Half-word, Byte Access**
  - **Automatic Page Break When Memory Boundary Has Been Reached**
  - **Multibank Ping-pong Access**
  - **Timing Parameters Specified by Software**
  - **Automatic Refresh Operation, Refresh Rate is Programmable**
- **Energy-saving Capabilities**
  - **Self-refresh, Power-down and Deep Power Modes Supported**

12.5 Power Consumption

The values in Table 12-9 and Table 12-10 on page 46 are measured values of power consumption with operating conditions as follows:

- V_{DDIO} = 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 12-1. Measurement setup

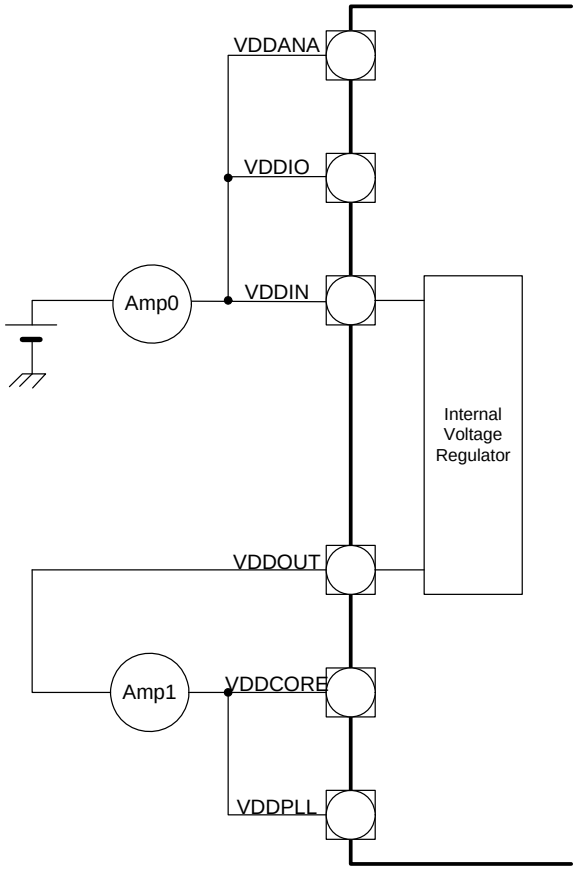


Table 12-21. Transfer Characteristics in 10-bit mode

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

12.9 EBI Timings

These timings are given for worst case process, T = 85·C, VDDCORE = 1.65V, VDDIO = 3V and 40 pF load capacitance.

Table 12-22. SMC Clock Signal.

Symbol	Parameter	Max[1]	Units
$1/(t_{CPSMC})$	SMC Controller Clock Frequency	$1/(t_{cpcpu})$	MHz

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

Table 12-23. SMC Read Signals with Hold Settings

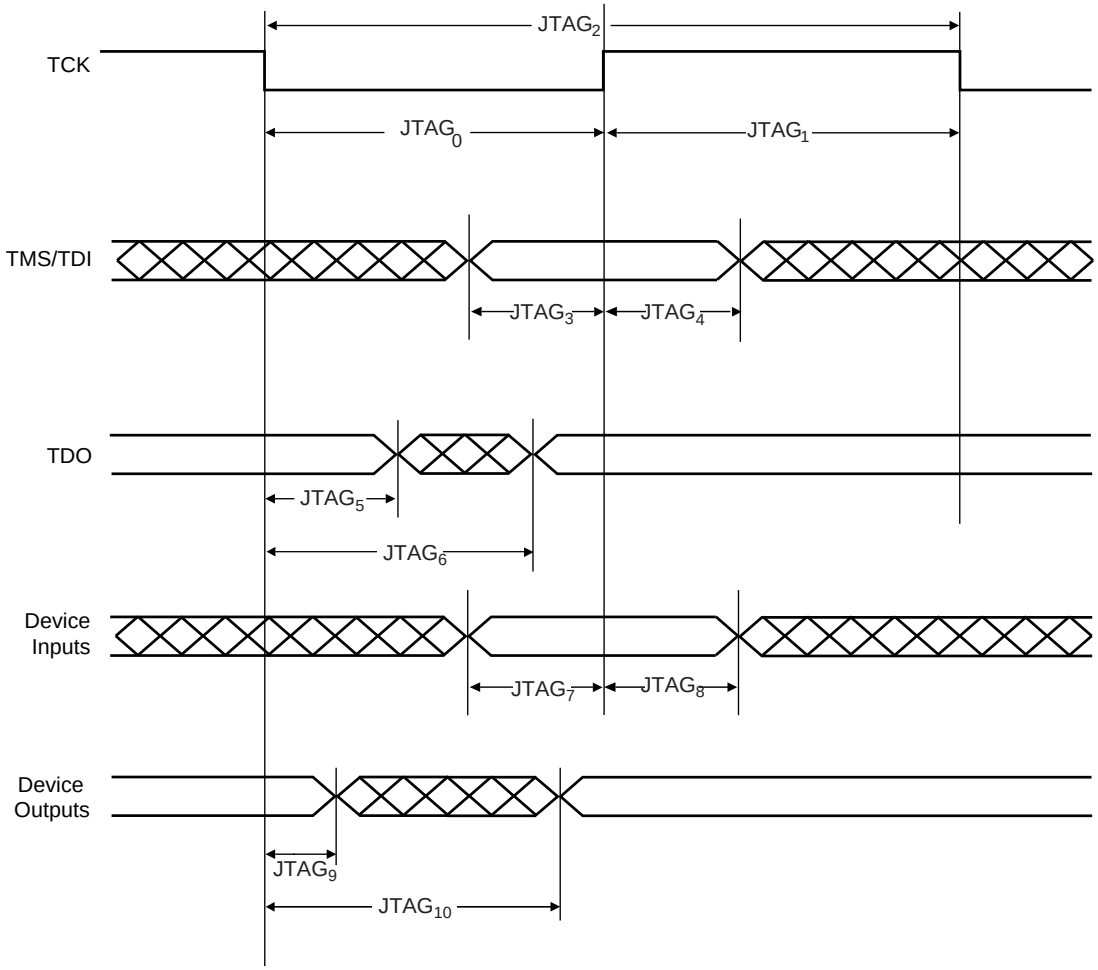
Symbol	Parameter	Min	Units
NRD Controlled (READ_MODE = 1)			
SMC_1	Data Setup before NRD High	12	ns
SMC_2	Data Hold after NRD High	0	
SMC_3	NRD High to NBS0/A0 Change[1]	nrd hold length * t_{CPSMC} - 1.3	
SMC_4	NRD High to NBS1 Change[1]	nrd hold length * t_{CPSMC} - 1.3	
SMC_5	NRD High to NBS2/A1 Change[1]	nrd hold length * t_{CPSMC} - 1.3	
SMC_6	NRD High to NBS3 Change[1]	nrd hold length * t_{CPSMC} - 1.3	
SMC_7	NRD High to A2 - A25 Change[1]	nrd hold length * t_{CPSMC} - 1.3	
SMC_8	NRD High to NCS Inactive[1]	(nrd hold length - ncs rd hold length) * t_{CPSMC} - 2.3	
SMC_9	NRD Pulse Width	nrd pulse length * t_{CPSMC} - 1.4	
NRD Controlled (READ_MODE = 0)			
SMC_{10}	Data Setup before NCS High	11.5	ns
SMC_{11}	Data Hold after NCS High	0	
SMC_{12}	NCS High to NBS0/A0 Change[1]	ncs rd hold length * t_{CPSMC} - 2.3	
SMC_{13}	NCS High to NBS0/A0 Change[1]	ncs rd hold length * t_{CPSMC} - 2.3	
SMC_{14}	NCS High to NBS2/A1 Change[1]	ncs rd hold length * t_{CPSMC} - 2.3	
SMC_{15}	NCS High to NBS3 Change[1]	ncs rd hold length * t_{CPSMC} - 2.3	
SMC_{16}	NCS High to A2 - A25 Change[1]	ncs rd hold length * t_{CPSMC} - 4	
SMC_{17}	NCS High to NRD Inactive[1]	ncs rd hold length - nrd hold length)* t_{CPSMC} - 1.3	
SMC_{18}	NCS Pulse Width	ncs rd pulse length * t_{CPSMC} - 3.6	

Note: 1. hold length = total cycle duration - setup duration - pulse duration. “hold length” is for “ncs rd hold length” or “nrd hold length”.

Table 12-28. SDRAM Clock Signal.

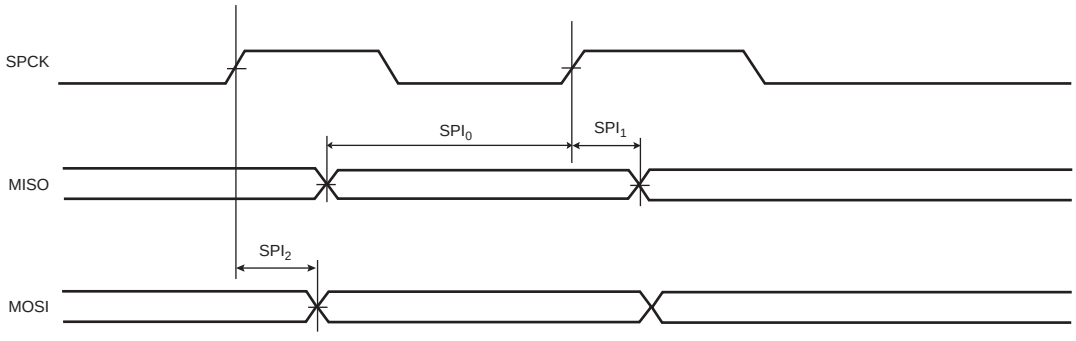
Symbol	Parameter	Min	Units
$SDRAMC_{11}$	Address Change before SDCK Rising Edge	6.2	ns
$SDRAMC_{12}$	Address Change after SDCK Rising Edge	2.2	
$SDRAMC_{13}$	Bank Change before SDCK Rising Edge	6.3	
$SDRAMC_{14}$	Bank Change after SDCK Rising Edge	2.4	
$SDRAMC_{15}$	CAS Low before SDCK Rising Edge	7.4	
$SDRAMC_{16}$	CAS High after SDCK Rising Edge	1.9	
$SDRAMC_{17}$	DQM Change before SDCK Rising Edge	6.4	
$SDRAMC_{18}$	DQM Change after SDCK Rising Edge	2.2	
$SDRAMC_{19}$	D0-D15 in Setup before SDCK Rising Edge	9	
$SDRAMC_{20}$	D0-D15 in Hold after SDCK Rising Edge	0	
$SDRAMC_{23}$	SDWE Low before SDCK Rising Edge	7.6	
$SDRAMC_{24}$	SDWE High after SDCK Rising Edge	1.8	
$SDRAMC_{25}$	D0-D15 Out Valid before SDCK Rising Edge	7.1	
$SDRAMC_{26}$	D0-D15 Out Valid after SDCK Rising Edge	1.5	

Figure 12-5. JTAG Interface Signals



12.11 SPI Characteristics

Figure 12-6. SPI Master mode with (CPOL = NCPHA = 0) or (CPOL= NCPHA= 1)



15. Errata

All industrial parts labelled with -UES (engineering samples) are revision E parts.

15.1 Rev. K, L, M

15.1.1 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. **PWM 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.

15.1.2 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.

15.1.3 SPI

1. **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.

2. **SPI FDIV option does not work**
   Selecting clock signal using FDIV = 1 does not work as specified.

   Fix/Workaround
   Do not set FDIV = 1.

3. **SPI Bad Serial Clock Generation on 2nd chip_select when SCBR = 1, CPOL=1 and NCPHA=0**
   When multiple CS are in use, if one of the baudrate equals to 1 and one of the others doesn't equal to 1, and CPOL=1 and CPHA=0, then an aditional pulse will be generated on SCK.
   Fix/workaround
   When multiple CS are in use, if one of the baudrate equals 1, the other must also equal 1 if CPOL=1 and CPHA=0.

4. **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 befin and RXREADY will now behave as expected.

5. **SPI Disable does not work in Slave mode**
   Fix/workaround
   Read the last received data then perform a Software reset.

15.1.4 Power Manager

1. **If the BOD level is higher than VDDCORE, the part is constantly under 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.

15.1.5 PDCA

1. **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.

15.1.6 TWI

1. **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.

15.1.7 USART

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

15.1.8 Processor and Architecture

1. **LDM instruction with PC in the register list and without ++ increments Rp**

15.2 Rev. J

15.2.1 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. **PWM 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.

15.2.2 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.

15.2.3 SPI

1. **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.

2. **SPI FDIV option does not work**
   Selecting clock signal using FDIV = 1 does not work as specified.

   Fix/Workaround
   Do not set FDIV = 1.

3. **SPI Bad Serial Clock Generation on 2nd chip_select when SCBR = 1, CPOL=1 and NCPHA=0**
   When multiple CS are in use, if one of the baudrate equals to 1 and one of the others doesn't equal to 1, and CPOL=1 and CPHA=0, then an aditional pulse will be generated on SCK.
   Fix/workaround

None.

15.2.9 USART

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

15.2.10 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. **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/Woraround
   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.

15.3 Rev. I

15.3.1 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. **PWM 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.

15.3.2 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.

15.3.3 SPI

1. **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.

2. **SPI FDIV option does not work**
   Selecting clock signal using FDIV = 1 does not work as specified.

   Fix/Workaround
   Do not set FDIV = 1.

3. **SPI Bad Serial Clock Generation on 2nd chip_select when SCBR = 1, CPOL=1 and NCPHA=0**
   When multiple CS are in use, if one of the baudrate equals to 1 and one of the others doesn't equal to 1, and CPOL=1 and CPHA=0, then an aditional pulse will be generated on SCK.
   Fix/workaround

Workaround/fix
The same PID should not be assigned to more than one channel.

15.3.7 GPIO

1. **Some GPIO VIH (input high voltage) are 3.6V max instead of 5V tolerant**
Only 11 GPIOs remain 5V tolerant (VIHmax=5V):PB01, PB02, PB03, PB10, PB19, PB20, PB21, PB22, PB23, PB27, PB28**.**
Workaround/fix
None.

15.3.8 USART

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

15.3.9 TWI

1. **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.

15.3.10 SDRAMC

1. **Code execution from external SDRAM does not work**
Code execution from SDRAM does not work.

Fix/Workaround
Do not run code from SDRAM.

15.3.11 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. **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/Woraround
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

2. **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 E**.**

15.5.14 OCD

1. **Stalled memory access instruction writeback fails if followed by a HW breakpoint.**
   Consider the following assembly code sequence:
   A
   B
   If a hardware breakpoint is placed on instruction B, and instruction A is a memory access instruction, register file updates from instruction A can be discarded.
   Fix/Workaround
   Do not place hardware breakpoints, use software breakpoints instead.
   Alternatively, place a hardware breakpoint on the instruction before the memory access instruction and then single step over the memory access instruction.

15.5.15 PDCA

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

15.5.16 TWI

1. **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.

16.6 Rev. C – 10/07

1. Updated ”Signal Description List” on page 8. Removed RXDN and TXDN from USART section.
2. Updated ”Errata” on page 70. Rev G replaced by rev H.

16.7 Rev. B – 10/07

1. Updated ”Features” on page 1.
2. Update ”Blockdiagram” on page 4 with local bus.
3. Updated ”Peripherals” on page 34 with local bus.
4. Add SPI feature in ”Universial Synchronous/Asynchronous Receiver/Transmitter (USART)” on page 315.
5. Updated ”USB On-The-Go Interface (USBB)” on page 517.
6. Updated ”JTAG and Boundary Scan” on page 750 with programming procedure .
7. Add description for silicon Rev G.

16.8 Rev. A – 03/07

1. Initial revision.