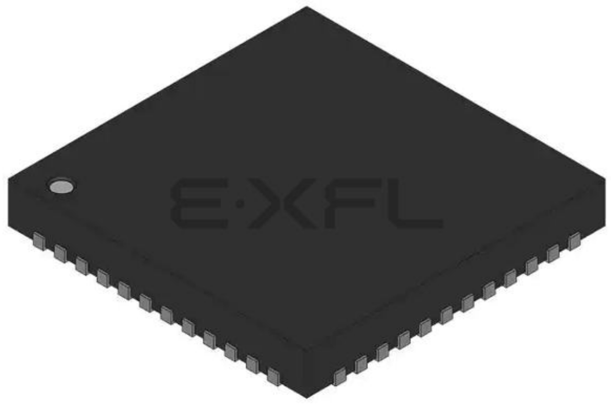
Atmel - ATUC64D4-Z1UR Datasheet



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	48MHz
Connectivity	I²C, SPI, UART/USART, USB
Peripherals	Brown-out Detect/Reset, DMA, I²S, POR, PWM, WDT
Number of I/O	35
Program Memory Size	64KB (64K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	16K 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-VFQFN Exposed Pad
Supplier Device Package	48-QFN (7x7)
Purchase URL	https://www.e-xfl.com/product-detail/atmel/atuc64d4-z1ur

The Capacitive Touch (CAT) module senses touch on external capacitive touch sensors, using the QTouch® technology. Capacitive touch sensors use no external mechanical components, unlike normal push buttons, and therefore demand less maintenance in the user application. The CAT module allows up to 25 touch sensors. One touch sensor can be configured to operate autonomously without software interaction,allowing wakeup from sleep modes when activated.

Atmel also offers the QTouch library for embedding capacitive touch buttons, sliders, and wheels functionality into AVR microcontrollers. The patented charge-transfer signal acquisition offers robust sensing and included fully debounced reporting of touch keys and includes Adjacent Key Suppression® (AKS®) technology for unambiguous detection of key events. The easy-to-use QTouch Suite toolchain allows you to explore, develop, and debug your own touch applications.

The UC3D integrates a class 2+ Nexus 2.0 On-Chip Debug (OCD) System, with full-speed read/write memory access, in addition to basic runtime control. The single-pin aWire interface allows all features available through the JTAG interface to be accessed through the RESET pin, allowing the JTAG pins to be used for GPIO or peripherals.

contains information allowing the core to resume operation in the previous execution mode. This concludes the event handling.

5.5.3 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.

5.5.4 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 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.

5.5.5 Entry Points for Events

Several different event handler entry points exist. In AVR32UC, the reset address is 0x80000000. 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.

All interrupt requests have entry points located at an offset relative to EVBA. This autovector offset is specified by an 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.

If several events occur on the same instruction, they are handled in a prioritized way. The priority ordering is presented in Table 5-4 on page 32. 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.

rent transfer, if no other master request is pending, the slave remains connected to the last master that performed the access. Other non privileged masters still get one latency cycle if they want to access the same slave. This technique can be used for masters that mainly perform single accesses.

- Round-Robin Arbitration with Fixed Default Master

This is another biased round-robin algorithm. It allows the Bus Matrix arbiters to remove the one latency cycle for the fixed default master per slave. At the end of the current access, the slave remains connected to its fixed default master. Every request attempted by this fixed default master will not cause any latency whereas other non privileged masters will still get one latency cycle. This technique can be used for masters that mainly perform single accesses.

9.4.2.3 Fixed Priority Arbitration

This algorithm allows the Bus Matrix arbiters to dispatch the requests from different masters to the same slave by using the fixed priority defined by the user. If two or more master requests are active at the same time, the master with the highest priority number is serviced first. If two or more master requests with the same priority are active at the same time, the master with the highest number is serviced first.

For each slave, the priority of each master may be defined through the Priority Registers for Slaves (PRAS and PRBS).

9.4.3 Slave and Master assignation

The index number assigned to Bus Matrix slaves and masters are described in the Module Configuration section at the end of this chapter.

10.5.10 Priority

If more than one PDCA channel is requesting transfer at a given time, the PDCA channels are prioritized by their channel number. Channels with lower numbers have priority over channels with higher numbers, giving channel zero the highest priority.

10.5.11 Error Handling

If the Memory Address Register (MAR) is set to point to an invalid location in memory, an error will occur when the PDCA tries to perform a transfer. When an error occurs, the Transfer Error bit in the Interrupt Status Register (ISR.TERR) will be set and the DMA channel that caused the error will be stopped. In order to restart the channel, the user must program the Memory Address Register to a valid address and then write a one to the Error Clear bit in the Control Register (CR.ECLR). If the Transfer Error interrupt is enabled, an interrupt request will be generated when a transfer error occurs.

10.6.15 Interrupt Status Register

Name:	ISR
Access Type:	Read-only
Offset:	0x02C + n*0x040
Reset Value:	0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	TERR	TRC	RCZ

- **TERR: Transfer Error**
  This bit is cleared when no transfer errors have occurred since the last write to CR.ECLR.
  This bit is set when one or more transfer errors has occurred since reset or the last write to CR.ECLR.
- **TRC: Transfer Complete**
  This bit is cleared when the TCR and/or the TCRR holds a non-zero value.
  This bit is set when both the TCR and the TCRR are zero.
- **RCZ: Reload Counter Zero**
  This bit is cleared when the TCRR holds a non-zero value.
  This bit is set when TCRR is zero.

Table 12-3. Wake-up Sources

Index[1]	Sleep Mode	Wake-up Sources
0	**Idle**	Synchronous, Asynchronous
1	**Frozen**	Synchronous[2], Asynchronous
2	**Standby**	Asynchronous
3	**Stop**	Asynchronous
4	**DeepStop**	Asynchronous
5	**Static**	Asynchronous[3]

Notes: 1. The sleep mode index is used as argument for the sleep instruction.
2. Only PB modules operational, as HSB module clocks are stopped.
3. WDT only available if clocked from pre-enabled OSC32K.

12.6.3.4 SleepWalking

In all sleep modes where the PBx clocks are stopped, the device can partially wake up if a PBx module asynchronously discovers that it needs its clock. Only the requested clocks and clock sources needed will be started, all other clocks will remain masked to zero. E.g. if the main clock source is OSC0, only OSC0 will be started even if other clock sources were enabled in normal mode. Generic clocks can also be started in a similar way. The state where only requested clocks are running is referred to as SleepWalking.

The time spent to start the requested clock is mostly limited by the startup time of the given clock source. This allows PBx modules to handle incoming requests, while still keeping the power consumption at a minimum.

When the device is SleepWalking any asynchronous wake-up can wake the device up at any time without stopping the requested PBx clock.

All requests to start clocks can be masked by writing to the Peripheral Power Control Register (PPCR), all requests are enabled at reset.

During SleepWalking the interrupt controller clock will be running. If an interrupt is pending when entering SleepWalking, it will wake the whole device up.

12.6.3.5 Precautions when entering sleep mode

Modules communicating with external circuits should normally be disabled before entering a sleep mode that will stop the module operation. This will prevent erratic behavior caused by entering or exiting sleep modes. Please refer to the relevant module documentation for recommended actions.

Communication between the synchronous clock domains is disturbed when entering and exiting sleep modes. Bus transactions over clock domains affected by the sleep mode are therefore not recommended. The system may hang if the bus clocks are stopped during a bus transaction.

The CPU is automatically stopped in a safe state to ensure that all CPU bus operations are complete when the sleep mode goes into effect. Thus, when entering Idle mode, no further action is necessary.

13.6.6 Power and Clocks Status Register

Name:	PCLKSR
Access Type:	Read-only
Offset:	0x0014
Reset Value:	0x00000000

31	30	29	28	27	26	25	24
AE	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	PLL1_LOCK LOST	PLL0_LOCK LOST	BODDET	PLL1_LOCK	PLL0_LOCK	OSC32RDY	OSC0RDY

- **AE: SCIF Access Error value**
  0: No access error has occurred on the SCIF.
  1: An access error has occurred on the SCIF.
- **PLLL1_LOCKLOST: PLL1 lock lost value**
  0: PLL1 has not lost its lock or has never been enabled.
  1: PLL1 has lost its lock, either by disabling the PLL1 or due to faulty operation.
- **PLLL0_LOCKLOST: PLL0 lock lost value**
  0: PLL1 has not lost its lock or has never been enabled.
  1: PLL1 has lost its lock, either by disabling the PLL1 or due to faulty operation.
- **BODDET: 1.8V Brown out detection**
  0: 1.8V BOD not enabled or the 1.8V power supply is above the 1.8V BOD threshold.
  1: 1.8V BOD enabled and the 1.8V power supply is going below 1.8V BOD threshold.
- **PLL1_LOCK: PLL1 Locked on Accurate value**
  0: PLL1 is unlocked on Accurate value.
  1: PLL1 is locked on Accurate value, and is ready to be selected as clock source with an accurate output clock.
- **PLL0_LOCK: PLL0 Locked on Accurate value**
  0: PLL0 is unlocked on Accurate value.
  1: PLL0 is locked on Accurate value, and is ready to be selected as clock source with an accurate output clock.
- **OSC32RDY: 32 KHz oscillator Ready**
  0: Oscillator 32 not enabled or not ready.
  1: Oscillator 32 is stable and ready to be used as clock source.
- **OSC0RDY: OSC0Ready**
  0: Oscillator not enabled or not ready.
  1: Oscillator is stable and ready to be used as clock source.

13.6.10 1.8V BOD Control Register

Name: BOD

Access Type: Read/Write

Offset: 0x0028

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
SFV	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	FCD

15	14	13	12	11	10	9	8
-	-	-	-	-	-	CTRL	

7	6	5	4	3	2	1	0
-	HYST	LEVEL					

- **SFV: Store Final Value**
  0: The register is read/write
  1: The register is read-only, to protect against further accidental writes.
- **FCD: BOD Fuse Calibration Done**
  This bit is set to 1 when the CTRL, HYST and LEVEL fields have been updated by the flash fuses after a reset.
  0: The flash calibration will be redone after any reset.
  1: The flash calibration will not be redone after a BOD reset.
- **CTRL: BOD Control**

Table 13-6. Operation mode for BOD

CTRL	Description
0x0	BOD is off
0x1	BOD is enabled and can reset the chip
0x2	BOD is enabled and but cannot reset the chip. Only interrupt will be sent to interrupt controller, if enabled in the IMR register.
0x3	Reserved

- **HYST: BOD Hysteresis**
  0: No hysteresis
  1: Hysteresis on.
- **LEVEL: BOD Level**
  This field sets the triggering threshold of the BOD. See Electrical Characteristics for actual voltage levels.
  Note that any change to the LEVEL field of the BOD register should be done with the BOD deactivated to avoid spurious reset or interrupt.

- **CEN: Clock Enable**

  0: Clock is stopped.

  1: Clock is running.

16.7.5 Interrupt Clear Register

Name:	ICR
Access Type:	Write-only
Offset:	0x010
Reset Value:	0x00000000

31	30	29	28	27	26	25	24
-	INT30	INT29	INT28	INT27	INT26	INT25	INT24

23	22	21	20	19	18	17	16
INT23	INT22	INT21	INT20	INT19	INT18	INT17	INT16

15	14	13	12	11	10	9	8
INT15	INT14	INT13	INT12	INT11	INT10	INT9	INT8

7	6	5	4	3	2	1	0
INT7	INT6	INT5	INT4	INT3	INT2	INT1	NMI

- **INTn: External Interrupt n**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the corresponding bit in ISR.
  Please refer to the Module Configuration section for the number of external interrupts.
- **NMI: Non-Maskable Interrupt**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the corresponding bit in ISR.

18.7.6 Peripheral Mux Register 1

Name:	PMR1
Access:	Read/Write, Set, Clear, Toggle
Offset:	0x020, 0x024, 0x028, 0x02C
Reset Value:	-

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24

23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16

15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8

7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0-31: Peripheral Multiplexer Select bit 1**

18.7.11 Pull-up Enable Register

Name:	PUER
Access:	Read/Write, Set, Clear, Toggle
Offset:	0x070, 0x074, 0x078, 0x07C
Reset Value:	-

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24

23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16

15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8

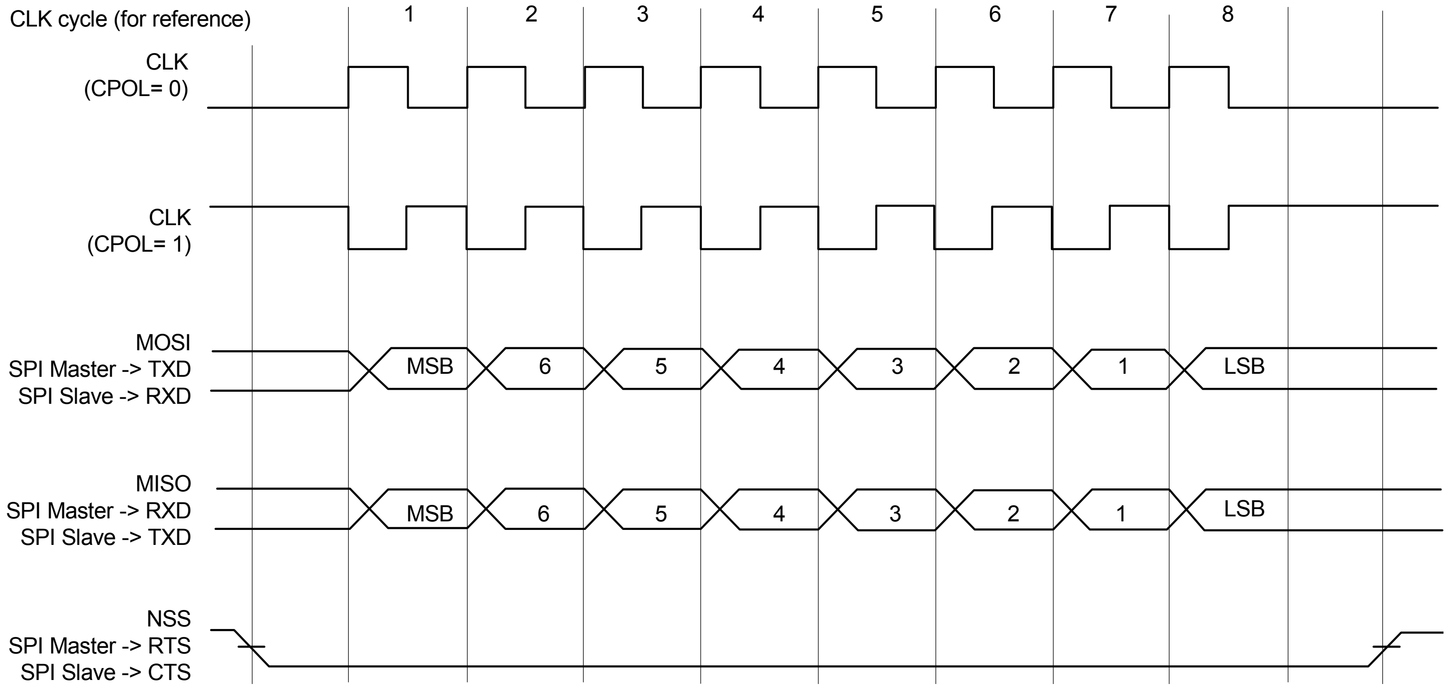
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0-31: Pull-up Enable**

  Writing a zero to a bit in this register will disable pull-up on the corresponding pin.

  Writing a one to a bit in this register will enable pull-up on the corresponding pin.

Figure 20-19. SPI Transfer Format (CPHA=0, 8 bits per transfer)



20.6.4.4 *Receiver and Transmitter Control*

See ”Transmitter Operations” on page 342, and ”Receiver Operations” on page 344.

20.6.4.5 *Character Transmission and Reception*

In SPI master mode, the slave select line (NSS) is asserted low one bit period before the start of transmission, and released high one bit period after every character transmission. A delay for at least three bit periods is always inserted in between characters. In order to address slave devices supporting the Chip Select Active After Transfer (CSAAT) mode, NSS can be forced low by writing a one to the Force SPI Chip Select bit (CR.RTSEN/FCS). Releasing NSS when FCS is one, is only possible by writing a one to the Release SPI Chip Select bit (CR.RTSDIS/RCS).

In SPI slave mode, a low level on NSS for at least one bit period will allow the slave to initiate a transmission or reception. The Underrun Error bit (CSR.UNRE) is set if a character must be sent while THR is empty, and TXD will be high during character transmission, as if 0xFF was being sent. If a new character is written to THR it will be sent correctly during the next transmission slot. Writing a one to CR.RSTSTA will clear UNRE. To ensure correct behavior of the receiver in SPI slave mode, the master device sending the frame must ensure a minimum delay of one bit period in between each character transmission.

20.6.4.6 *Receiver Time-out*

Receiver Time-out’s are not possible in SPI mode as the baud rate clock is only active during data transfers.

20.8 Module Configuration

The specific configuration for each USART instance is listed in the following tables.The module bus clocks listed here are connected to the system bus clocks. Please refer to the Power Manager chapter for details.

Table 20-16. Module Configuration

Feature	USART0	USART1	USART2
SPI Logic	Implemented	Implemented	Implemented
LIN Logic	Not Implemented	Not Implemented	Not Implemented
RS485 Logic	Not Implemented	Not Implemented	Not Implemented
Manchester Logic	Not Implemented	Not Implemented	Not Implemented
Modem Logic	Not Implemented	Not Implemented	Not Implemented
IRDA Logic	Not Implemented	Not Implemented	Not Implemented
Fractional Baudrate	Not Implemented	Not Implemented	Not Implemented
ISO7816	Not Implemented	Not Implemented	Not Implemented
DIV	8	8	8
Receiver Time-out Counter Size (Size of the RTOR.TO field)	8-bits	8-bits	8-bits

Table 20-17. Module Clock Name

Module Name	Clock Name
USART0	**CLK_USART0**
USART1	**CLK_USART1**
USART2	**CLK_USART2**

20.8.1 Clock Connections

Each USART can be connected to an internally divided clock:

Table 20-18. USART Clock Connections

USART	Source	Name	Connection
0	Internal	CLK_DIV	PBA Clock / 8
1			
2			

20.8.2 Register Reset Values

Table 20-19.

Register	Reset Value
VERSION	0x00000440

24. Two-wire Slave Interface (TWIS)

Rev.: 1.2.0.1

24.1 Features

- **Compatible with I²C standard**
  - **Transfer speeds of 100 and 400 kbit/s**
  - **7 and 10-bit and General Call addressing**
- **Compatible with SMBus standard**
  - **Hardware Packet Error Checking (CRC) generation and verification with ACK response25 ms clock low timeout delay**
  - **25 ms slave cumulative clock low extend time**
- **Compatible with PMBus**
- **DMA interface for reducing CPU load**
- **Arbitrary transfer lengths, including 0 data bytes**
- **Optional clock stretching if transmit or receive buffers not ready for data transfer**
- **32-bit Peripheral Bus interface for configuration of the interface**

24.2 Overview

The Atmel Two-wire Slave Interface (TWIS) interconnects components on a unique two-wire bus, made up of one clock line and one data line with speeds of up to 400 kbit/s, based on a byte-oriented transfer format. It can be used with any Atmel Two-wire Interface bus, I²C, or SMBus-compatible master. The TWIS is always a bus slave and can transfer sequential or single bytes.

Below, Table 24-1 lists the compatibility level of the Atmel Two-wire Slave Interface and a full I²C compatible device.

Table 24-1. Atmel TWIS Compatibility with I²C Standard

I²C Standard	**Atmel TWIS**
Standard-mode (100 kbit/s)	Supported
Fast-mode (400 kbit/s)	Supported
7 or 10 bits Slave Addressing	Supported
START BYTE(1)	Not Supported
Repeated Start (Sr) Condition	Supported
ACK and NAK Management	Supported
Slope control and input filtering (Fast mode)	Supported
Clock stretching	Supported

Note: 1. START + b000000001 + Ack + Sr

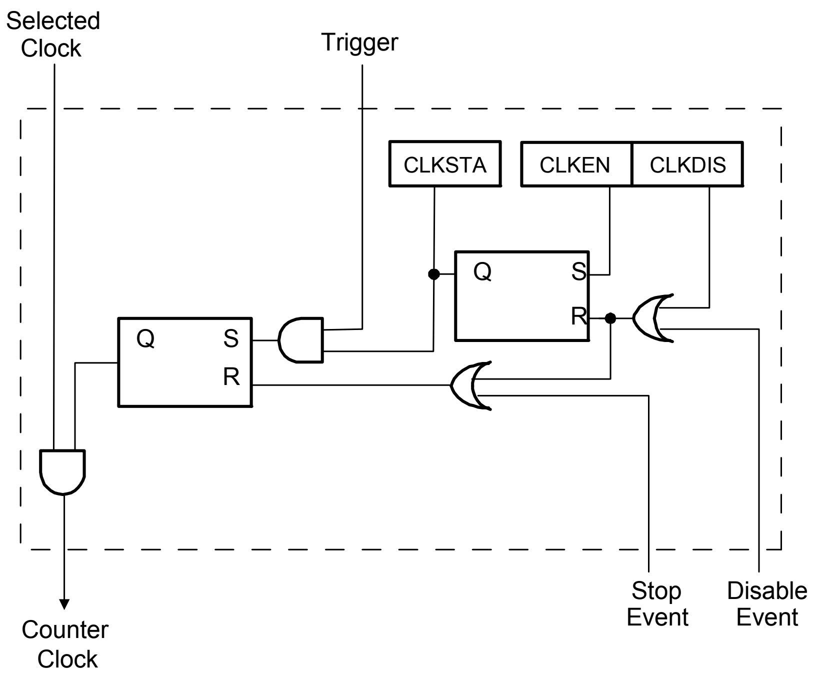
Below, Table 24-2 lists the compatibility level of the Atmel Two-wire Slave Interface and a full SMBus compatible device.

Table 24-2. Atmel TWIS Compatibility with SMBus Standard

SMBus Standard	**Atmel TWIS**
Bus Timeouts	Supported
Address Resolution Protocol	Supported
Packet Error Checking	Supported

- The clock can be enabled or disabled by the user by writing to the Counter Clock Enable/Disable Command bits in the Channel n Clock Control Register (CCRn.CLKEN and CCRn.CLKDIS). In Capture mode it can be disabled by an RB load event if the Counter Clock Disable with RB Loading bit in CMRn is written to one (CMRn.LDBDIS). In Waveform mode, it can be disabled by an RC Compare event if the Counter Clock Disable with RC Compare bit in CMRn is written to one (CMRn.CPCDIS). When disabled, the start or the stop actions have no effect: only a CLKEN command in CCRn can re-enable the clock. When the clock is enabled, the Clock Enabling Status bit is set in SRn (SRn.CLKSTA).
- The clock can also be started or stopped: a trigger (software, synchro, external or compare) always starts the clock. In Capture mode the clock can be stopped by an RB load event if the Counter Clock Stopped with RB Loading bit in CMRn is written to one (CMRn.LDBSTOP). In Waveform mode it can be stopped by an RC compare event if the Counter Clock Stopped with RC Compare bit in CMRn is written to one (CMRn.CPCSTOP). The start and the stop commands have effect only if the clock is enabled.

Figure 26-3. Clock Control



26.6.1.5 *TC operating modes*

Each channel can independently operate in two different modes:

- Capture mode provides measurement on signals.
- Waveform mode provides wave generation.

The TC operating mode selection is done by writing to the Wave bit in the CCRn register (CCRn.WAVE).

In Capture mode, TIOA and TIOB are configured as inputs.

In Waveform mode, TIOA is always configured to be an output and TIOB is an output if it is not selected to be the external trigger.

to be set. If the power reduction mode is on, only SR.EN can tell if the ADCIFD is ready for operation since startup-time will be performed only when a sequencer trigger event occurs. Please note that all ADCIFD controls will be ignored until SR.EN goes to '1'.

Before the ADCIFD can be used, the I/O Controller must be configured correctly and the Reference Voltage (ADVREF) signal must be connected. Refer to I/O Controller section for details. Note that once configured, ADCIFD configuration registers should not be written during operation since they are permanently used by the ADCIFD. The user must ensure that ADCIFD is stopped during configuration unless he knows what he is doing.

28.6.2 Basic Operation

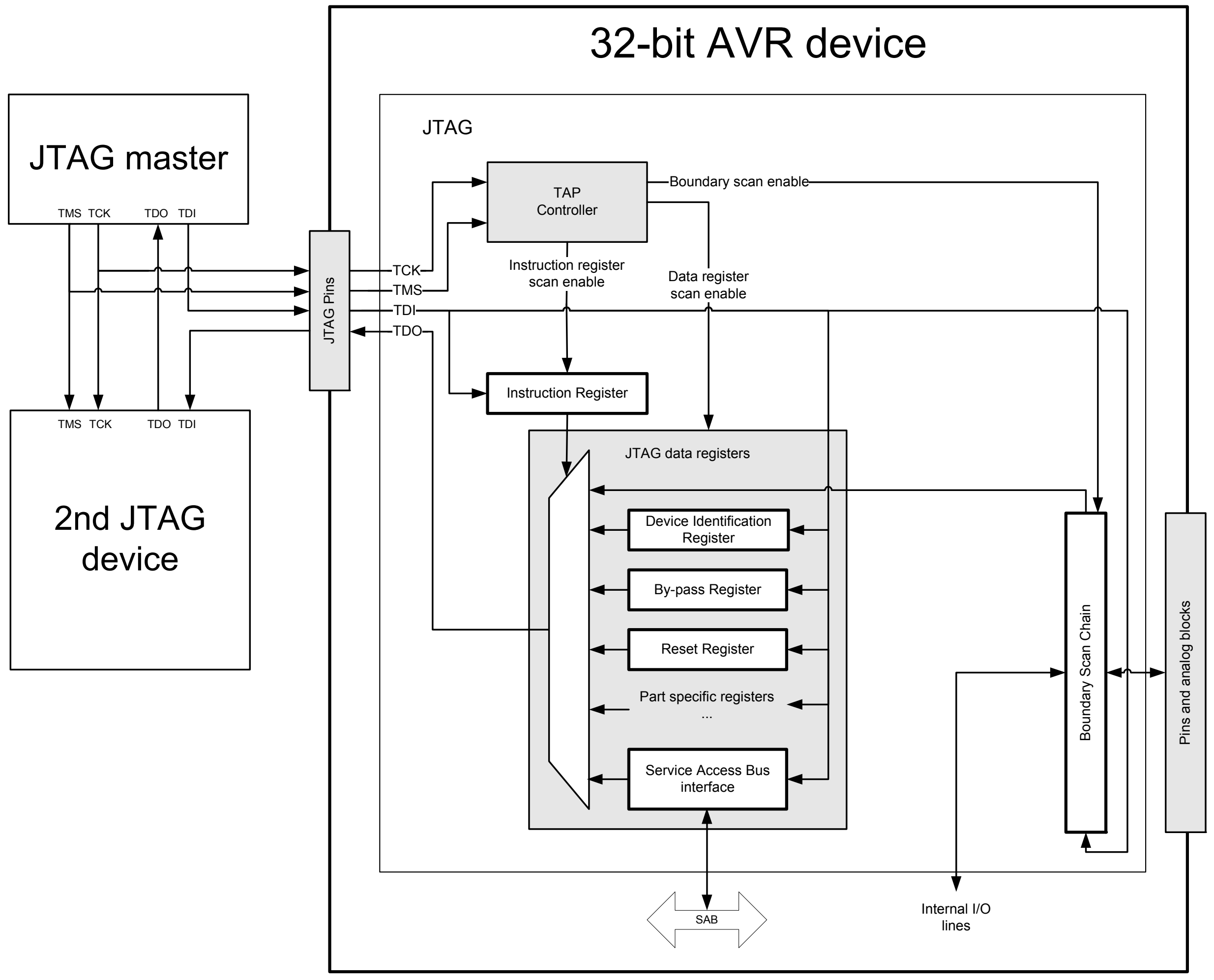
To convert analog values to digital values the user must first initialize the ADCIFD as described in Section 28.6.1. When the ADCIFD is initialized the sequencer must be configured by writing the Number of Conversions in the Sequence (CNVNB) in the Sequencer Configuration Register (SEQCFG) and by writing the index channels in the Channel Selection Per Low/High conversion registers, respectively CSPLC and CSPHC. Configuring channel N for a given conversion instructs the ADCIFD to convert the analog voltage applied to AD pin N. To start converting data the user can either manually start a conversion sequence by write a one to the sequencer trigger event (STRIG) bit in the Control Register (CR) or configure an automatic trigger to initiate the conversions. The automatic trigger can be configured to trig on many different conditions. Refer to Section 28.6.13 for details. The result of the conversions are stored in the Last Converted Value register (LCV) as they become available, overwriting the result from the previous conversion. To avoid data loss if more than one channel is enabled, the user must read the conversion results as they become available either by using an interrupt handler or by using a Peripheral DMA channel to copy the results to memory. Failing to do so will result in an Overrun Error condition, indicated by the LOVR bit in the Status Register (SR). To use an interrupt handler the user must enable the End Of Conversion (EOC) interrupt request by writing a one to the corresponding bit in the Interrupt Enable Register (IER). To clear the interrupt after the conversion result is read, the user must write a one to the corresponding bit in the Status Clear Register (SCR). To use a Peripheral DMA Controller channel the user must configure the Peripheral DMA Controller appropriately. The DMA Controller will, when configured, automatically read converted data as they become available. There is no need to manually clear any bits in the Interrupt Status Register as this is performed by the hardware. If an Overrun Error condition happens during DMA operation, the LOVR bit in the SR will be set.

28.6.3 ADC resolution

The ADC supports 8-bit and 10-bit resolution. Resolution can be changed by writing the resolution field (RS) in the Sequencer Configuration register (SEQCFG). By default, after a reset, the resolution is set to 10-bit. Note that an external decoupling capacitor connected to ADVREF and GNDANA is mandatory to achieve maximum resolution.

31.5.3 Block Diagram

Figure 31-9. JTAG and Boundary-scan Access



31.5.4 I/O Lines Description

Table 31-37. I/O Line Description

Pin Name	Pin Description	Type	Active Level
RESET_N	External reset pin. Used when enabling and disabling the JTAG.	Input	Low
TCK	Test Clock Input. Fully asynchronous to system clock frequency.	Input	
TMS	Test Mode Select, sampled on rising TCK.	Input	
TDI	Test Data In, sampled on rising TCK.	Input	
TDO	Test Data Out, driven on falling TCK.	Output	

31.5.5 Product Dependencies

In order to use this module, other parts of the system must be configured correctly, as described below.

34. Ordering Information

Table 34-1. Ordering Information

Device	Ordering Code	Carrier Type	Package	Package Type	Temperature Operating Range
ATUC128D3	ATUC128D3-A2UT	Tray	TQFP 64	JESD97 Classification E3	Industrial (-40·C to 85·C)
	ATUC128D3-A2UR	Tape & Reel	TQFP 64		
	ATUC128D3-Z2UT	Tray	QFN 64		
	ATUC128D3-Z2UR	Tape & Reel	QFN 64		
ATUC128D4	ATUC128D4-AUT	Tray	TQFP 48		
	ATUC128D4-AUR	Tape & Reel	TQFP 48		
	ATUC128D4-Z1UT	Tray	QFN 48		
	ATUC128D4-Z1UR	Tape & Reel	QFN 48		
ATUC64D3	ATUC64D3-A2UT	Tray	TQFP 64	JESD97 Classification E3	Industrial (-40·C to 85·C)
	ATUC64D3-A2UR	Tape & Reel	TQFP 64		
	ATUC64D3-Z2UT	Tray	QFN 64		
	ATUC64D3-Z2UR	Tape & Reel	QFN 64		
ATUC64D4	ATUC64D4-AUT	Tray	TQFP 48		
	ATUC64D4-AUR	Tape & Reel	TQFP 48		
	ATUC64D4-Z1UT	Tray	QFN 48		
	ATUC64D4-Z1UR	Tape & Reel	QFN 48		