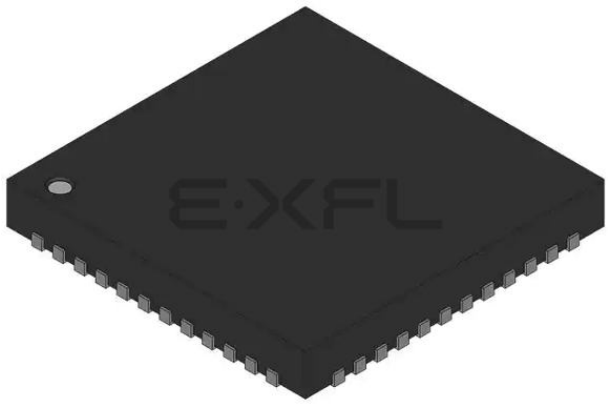


Welcome to E-XFL.COM

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	50MHz
Connectivity	I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, DMA, POR, PWM, WDT
Number of I/O	36
Program Memory Size	64KB (64K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	16K x 8
Voltage - Supply (Vcc/Vdd)	1.62V ~ 3.6V
Data Converters	A/D 8x12b
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/at32uc3l064-zaut

7.7.30 PDCA Version Register

Name: VERSION

Access Type: Read-only

Offset: 0x834

Reset Value: -

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

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

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant Number**
  Reserved. No functionality associated.
- **VERSION: Version Number**
  Version number of the module. No functionality associated.

8.8.1 Flash Control Register

Name: FCR

Access Type: Read/Write

Offset: 0x00

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
-	-	-	-	-	BRBUF	SEQBUF	-

7	6	5	4	3	2	1	0
-	FWS	-	-	PROGE	LOCKE	-	FRDY

- **BRBUF: Branch Target Instruction Buffer Enable**
  0: The Branch Target Instruction Buffer is disabled.
  1: The Branch Target Instruction Buffer is enabled.
- **SEQBUF: Sequential Instruction Fetch Buffer Enable**
  0: The Sequential Instruction Fetch Buffer is disabled.
  1: The Sequential Instruction Fetch Buffer is enabled.
- **FWS: Flash Wait State**
  0: The flash is read with 0 wait states.
  1: The flash is read with 1 wait state.
- **PROGE: Programming Error Interrupt Enable**
  0: Programming Error does not generate an interrupt request.
  1: Programming Error generates an interrupt request.
- **LOCKE: Lock Error Interrupt Enable**
  0: Lock Error does not generate an interrupt request.
  1: Lock Error generates an interrupt request.
- **FRDY: Flash Ready Interrupt Enable**
  0: Flash Ready does not generate an interrupt request.
  1: Flash Ready generates an interrupt request.

10.6 Module Configuration

The specific configuration for each HMATRIX 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 10-3. HMATRIX Clocks

Clock Name	Description
CLK_HMATRIX	Clock for the HMATRIX bus interface

10.6.1 Bus Matrix Connections

The bus matrix has the several masters and slaves. Each master has its own bus and its own decoder, thus allowing a different memory mapping per master. The master number in the table below can be used to index the HMATRIX control registers. For example, HMATRIX MCFG0 register is associated with the CPU Data master interface.

Table 10-4. High Speed Bus Masters

Master 0	CPU Data
Master 1	CPU Instruction
Master 2	CPU SAB
Master 3	SAU
Master 4	PDCA

Each slave has its own arbiter, thus allowing a different arbitration per slave. The slave number in the table below can be used to index the HMATRIX control registers. For example, SCFG3 is associated with the Internal SRAM Slave Interface.

Accesses to unused areas returns an error result to the master requesting such an access.

Table 10-5. High Speed Bus Slaves

Slave 0	Internal Flash
Slave 1	HSB-PB Bridge A
Slave 2	HSB-PB Bridge B
Slave 3	Internal SRAM
Slave 4	SAU

11. Interrupt Controller (INTC)

Rev: 1.0.2.5

11.1 Features

- **Autovectored low latency interrupt service with programmable priority**
  - **4 priority levels for regular, maskable interrupts**
  - **One Non-Maskable Interrupt**
- **Up to 64 groups of interrupts with up to 32 interrupt requests in each group**

11.2 Overview

The INTC collects interrupt requests from the peripherals, prioritizes them, and delivers an interrupt request and an autovector to the CPU. The AVR32 architecture supports 4 priority levels for regular, maskable interrupts, and a Non-Maskable Interrupt (NMI).

The INTC supports up to 64 groups of interrupts. Each group can have up to 32 interrupt request lines, these lines are connected to the peripherals. Each group has an Interrupt Priority Register (IPR) and an Interrupt Request Register (IRR). The IPRs are used to assign a priority level and an autovector to each group, and the IRRs are used to identify the active interrupt request within each group. If a group has only one interrupt request line, an active interrupt group uniquely identifies the active interrupt request line, and the corresponding IRR is not needed. The INTC also provides one Interrupt Cause Register (ICR) per priority level. These registers identify the group that has a pending interrupt of the corresponding priority level. If several groups have a pending interrupt of the same level, the group with the lowest number takes priority.

11.3 Block Diagram

Figure 11-1 gives an overview of the INTC. The grey boxes represent registers that can be accessed via the user interface. The interrupt requests from the peripherals (IREQn) and the NMI are input on the left side of the figure. Signals to and from the CPU are on the right side of the figure.

13.6.28 Brown-Out Detector Version Register

Name: BODIFAVERSION

Access Type: Read-only

Offset: 0x03D4

Reset Value: -

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

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

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant number**
  Reserved. No functionality associated.
- **VERSION: Version number**
  Version number of the module. No functionality associated.

15.6.2 Clear Register

Name: CLR

Access Type: Write-only

Offset: 0x004

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
KEY							

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

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

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	WDTCLR

When the Watchdog Timer is enabled, this Register must be periodically written within the window time frame or within the watchdog timeout period, to prevent a watchdog reset.

- **KEY**
  This field must be written twice, first with key value 0x55, then 0xAA, for a write operation to be effective.
- **WDTCLR: Watchdog Clear**
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the WDT counter.

16.4 I/O Lines Description

Table 16-1. I/O Lines Description

Pin Name	Pin Description	Type
NMI	Non-Maskable Interrupt	Input
EXTINTn	External Interrupt	Input

16.5 Product Dependencies

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

16.5.1 I/O Lines

The external interrupt pins (EXTINTn and NMI) may be multiplexed with I/O Controller lines. The programmer must first program the I/O Controller to assign the desired EIC pins to their peripheral function. If I/O lines of the EIC are not used by the application, they can be used for other purposes by the I/O Controller.

It is only required to enable the EIC inputs actually in use. If an application requires two external interrupts, then only two I/O lines will be assigned to EIC inputs.

16.5.2 Power Management

All interrupts are available in all sleep modes as long as the EIC module is powered. However, in sleep modes where CLK_SYNC is stopped, the interrupt must be configured to asynchronous mode.

16.5.3 Clocks

The clock for the EIC bus interface (CLK_EIC) is generated by the Power Manager. This clock is enabled at reset, and can be disabled in the Power Manager.

The filter and synchronous edge/level detector runs on a clock which is stopped in any of the sleep modes where the system RC oscillator (RCSYS) is not running. This clock is referred to as CLK_SYNC.

16.5.4 Interrupts

The external interrupt request lines are connected to the interrupt controller. Using the external interrupts requires the interrupt controller to be programmed first.

Using the Non-Maskable Interrupt does not require the interrupt controller to be programmed.

16.5.5 Debug Operation

When an external debugger forces the CPU into debug mode, the EIC continues normal operation. If the EIC is configured in a way that requires it to be periodically serviced by the CPU through interrupts or similar, improper operation or data loss may result during debugging.

16.6 Functional Description

16.6.1 External Interrupts

The external interrupts are not enabled by default, allowing the proper interrupt vectors to be set up by the CPU before the interrupts are enabled.

Each external interrupt INTn can be configured to produce an interrupt on rising or falling edge, or high or low level. External interrupts are configured by the MODE, EDGE, and LEVEL registers. Each interrupt has a bit INTn in each of these registers. Writing a zero to the INTn bit in the MODE register enables edge triggered interrupts, while writing a one to the bit enables level triggered interrupts.

If INTn is configured as an edge triggered interrupt, writing a zero to the INTn bit in the EDGE register will cause the interrupt to be triggered on a falling edge on EXTINTn, while writing a one to the bit will cause the interrupt to be triggered on a rising edge on EXTINTn.

If INTn is configured as a level triggered interrupt, writing a zero to the INTn bit in the LEVEL register will cause the interrupt to be triggered on a low level on EXTINTn, while writing a one to the bit will cause the interrupt to be triggered on a high level on EXTINTn.

Each interrupt has a corresponding bit in each of the interrupt control and status registers. Writing a one to the INTn bit in the Interrupt Enable Register (IER) enables the external interrupt from pin EXTINTn to propagate from the EIC to the interrupt controller, while writing a one to INTn bit in the Interrupt Disable Register (IDR) disables this propagation. The Interrupt Mask Register (IMR) can be read to check which interrupts are enabled. When an interrupt triggers, the corresponding bit in the Interrupt Status Register (ISR) will be set. This bit remains set until a one is written to the corresponding bit in the Interrupt Clear Register (ICR) or the interrupt is disabled.

Writing a one to the INTn bit in the Enable Register (EN) enables the external interrupt on pin EXTINTn, while writing a one to INTn bit in the Disable Register (DIS) disables the external interrupt. The Control Register (CTRL) can be read to check which interrupts are enabled. If a bit in the CTRL register is set, but the corresponding bit in IMR is not set, an interrupt will not propagate to the interrupt controller. However, the corresponding bit in ISR will be set, and EIC_WAKE will be set. Note that an external interrupt should not be enabled before it has been configured correctly.

If the CTRL.INTn bit is zero, the corresponding bit in ISR will always be zero. Disabling an external interrupt by writing a one to the DIS.INTn bit will clear the corresponding bit in ISR.

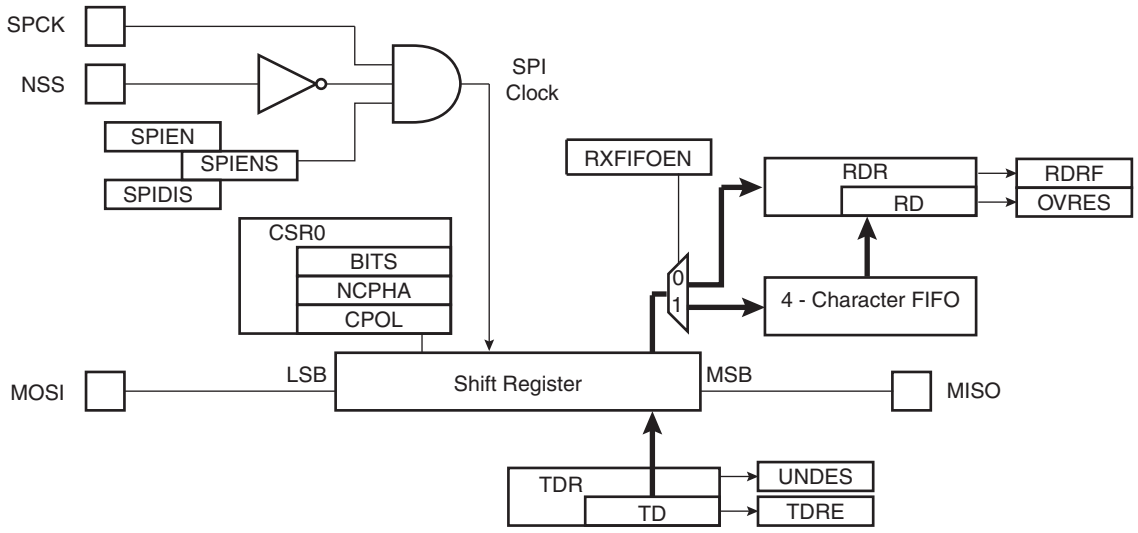
Please refer to the Module Configuration section for the number of external interrupts.

16.6.2 Synchronization and Filtering of External Interrupts

In synchronous mode the pin value of the EXTINTn pin is synchronized to CLK_SYNC, so spikes shorter than one CLK_SYNC cycle are not guaranteed to produce an interrupt. The synchronization of the EXTINTn to CLK_SYNC will delay the propagation of the interrupt to the interrupt controller by two cycles of CLK_SYNC, see Figure 16-2 and Figure 16-3 for examples (FILTER off).

It is also possible to apply a filter on EXTINTn by writing a one to the INTn bit in the FILTER register. This filter is a majority voter, if the condition for an interrupt is true for more than one of the latest three cycles of CLK_SYNC the interrupt will be set. This will additionally delay the propagation of the interrupt to the interrupt controller by one or two cycles of CLK_SYNC, see Figure 16-2 and Figure 16-3 for examples (FILTER on).

Figure 20-9. Slave Mode Functional Block Diagram



20.8.4 Transmit Data Register

Name: TDR

Access Type: Write-only

Offset: 0x0C

Reset Value: 0x00000000

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

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

15	14	13	12	11	10	9	8
TD[15:8]							

7	6	5	4	3	2	1	0
TD[7:0]							

- **LASTXFER: Last Transfer**

  1: The current NPCS will be deasserted after the character written in TD has been transferred. When CSRn.CSAAT is one, this allows to close the communication with the current serial peripheral by raising the corresponding NPCS line as soon as TD transfer has completed.
  0: Writing a zero to this bit has no effect.

  This field is only used if Variable Peripheral Select is active (MR.PS = 1).

- **PCS: Peripheral Chip Select**

  If PCSDEC = 0:
  PCS = xxx0NPCS[3:0] = 1110
  PCS = xx01NPCS[3:0] = 1101
  PCS = x011NPCS[3:0] = 1011
  PCS = 0111NPCS[3:0] = 0111
  PCS = 1111forbidden (no peripheral is selected)
  (x = don't care)
  If PCSDEC = 1:
  NPCS[3:0] output signals = PCS

  This field is only used if Variable Peripheral Select is active (MR.PS = 1).

- **TD: Transmit Data**

  Data to be transmitted by the SPI Interface is stored in this register. Information to be transmitted must be written to the TDR register in a right-justified format.

- **PPNCONF: Polarity Positive if Polarity not Configurable**
  0: If polarity is not configurable, polarity is negative.
  1: If polarity is not configurable, polarity is positive.
- **PCONF: Polarity Configurable**
  0: Polarity is not configurable.
  1: Polarity is configurable.
- **NCS: Number of Chip Selects**
  This field indicates the number of chip selects implemented.

slave to pull it down in order to generate the acknowledge. The master polls the data line during this clock pulse.

The SR.RXRDY bit indicates that a data byte is available in the RHR. The RXRDY bit is also used as Receive Ready for the Peripheral DMA Controller receive channel.

Figure 22-10. Slave Receiver with One Data Byte

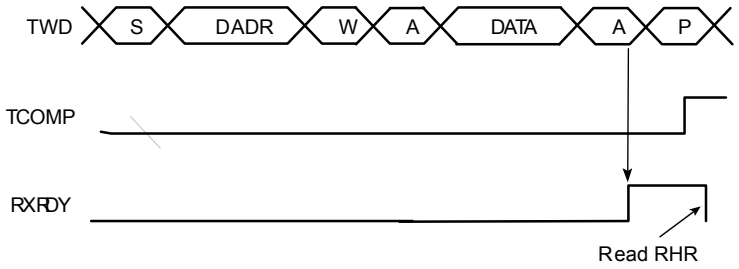
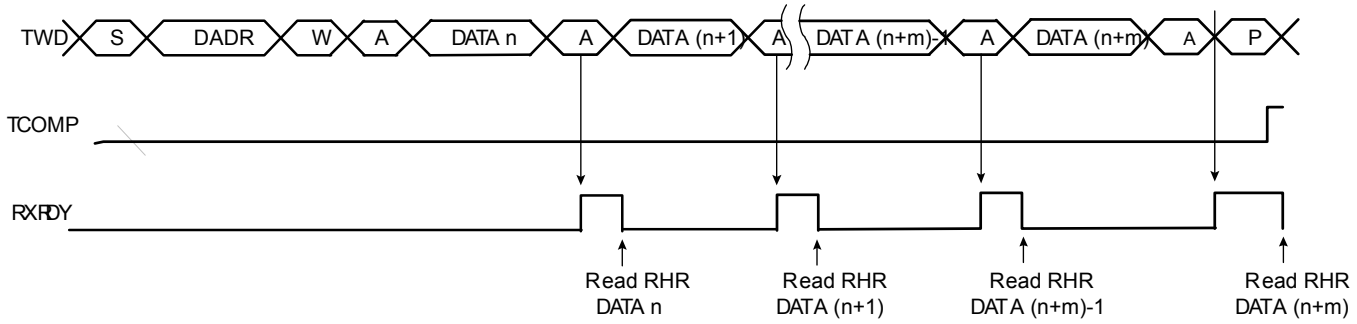


Figure 22-11. Slave Receiver with Multiple Data Bytes



22.8.5 Using the Peripheral DMA Controller

The use of the Peripheral DMA Controller significantly reduces the CPU load. The user can set up ring buffers for the Peripheral DMA Controller, containing data to transmit or free buffer space to place received data. By initializing NBYTES to zero before a transfer, and writing a one to CR.CUP, NBYTES is incremented by one each time a data has been transmitted or received. This allows the user to detect how much data was actually transferred by the DMA system.

To assure correct behavior, respect the following programming sequences:

22.8.5.1 Data Transmit with the Peripheral DMA Controller

1. Initialize the transmit Peripheral DMA Controller (memory pointers, size, etc.).
2. Configure the TWIS (ADR, NBYTES, etc.).
3. Start the transfer by enabling the Peripheral DMA Controller to transmit.
4. Wait for the Peripheral DMA Controller end-of-transmit flag.
5. Disable the Peripheral DMA Controller.

22.8.5.2 Data Receive with the Peripheral DMA Controller

1. Initialize the receive Peripheral DMA Controller (memory pointers, size - 1, etc.).
2. Configure the TWIS (ADR, NBYTES, etc.).

A trigger such as an external event or a software trigger can modify CVn at any time. If a trigger occurs while CVn is incrementing, CVn then decrements. If a trigger is received while CVn is decrementing, CVn then increments. See Figure 24-11 on page 554.

RC Compare cannot be programmed to generate a trigger in this configuration.

At the same time, RC Compare can stop the counter clock (CMRn.CPCSTOP = 1) and/or disable the counter clock (CMRn.CPCDIS = 1).

Figure 24-10. WAVSEL = 1 Without Trigger

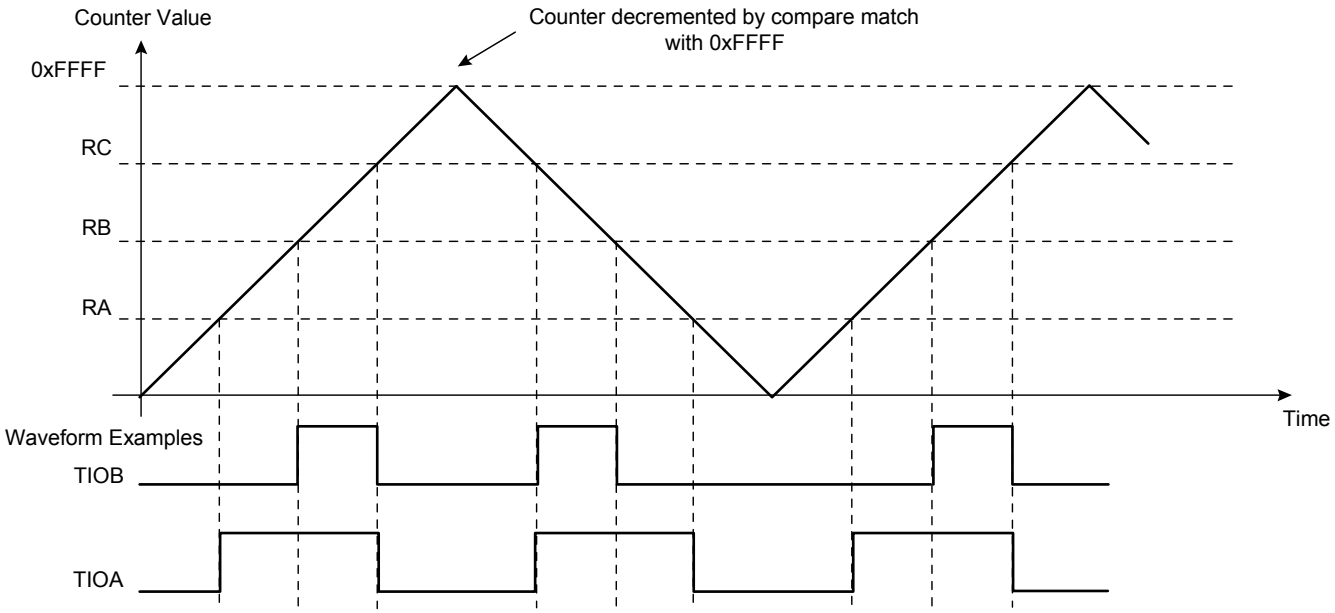


Figure 24-11. WAVSEL = 1 With Trigger

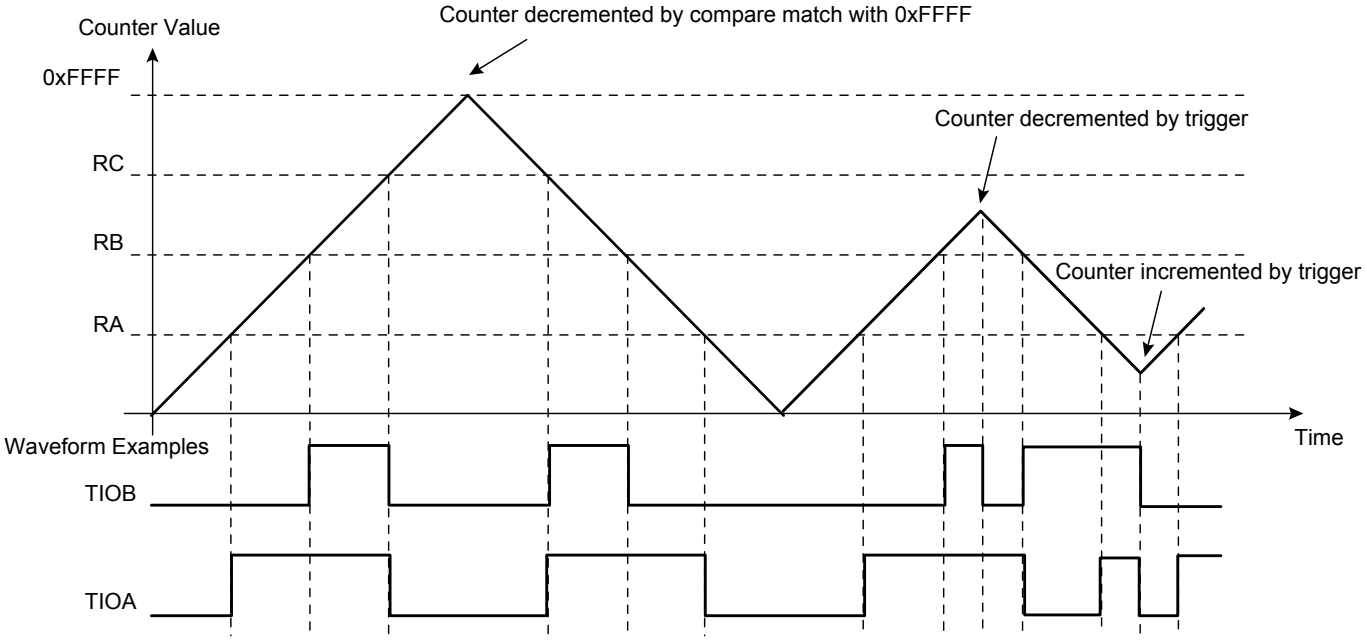


Table 25-1. Peripheral Event Mapping from ACIFB to PWMA

Generator	Generated Event	User	Effect	Asynchronous
ACIFB channel 0	AC0 V_{INP} > AC0 V_{INN}	PWMA channel 0	PWMA duty cycle value increased by one	No
	AC0 V_{INN} > AC0 V_{INP}		PWMA duty cycle value decreased by one	
ACIFB channel 1	AC1 V_{INP} > AC1 V_{INN}	PWMA channel 6	PWMA duty cycle value increased by one	
	AC1 V_{INN} > AC1 V_{INP}		PWMA duty cycle value decreased by one	
ACIFB channel 2	AC2 V_{INP} > AC2 V_{INN}	PWMA channel 8	PWMA duty cycle value increased by one	
	AC2 V_{INN} > AC2 V_{INP}		PWMA duty cycle value decreased by one	
ACIFB channel 3	AC3 V_{INP} > AC3 V_{INN}	PWMA channel 9	PWMA duty cycle value increased by one	
	AC3 V_{INN} > AC3 V_{INP}		PWMA duty cycle value decreased by one	
ACIFB channel 4	AC4 V_{INP} > AC4 V_{INN}	PWMA channel 11	PWMA duty cycle value increased by one	
	AC4 V_{INN} > AC4 V_{INP}		PWMA duty cycle value decreased by one	
ACIFB channel 5	AC5 V_{INP} > AC5 V_{INN}	PWMA channel 14	PWMA duty cycle value increased by one	
	AC5 V_{INN} > AC5 V_{INP}		PWMA duty cycle value decreased by one	
ACIFB channel 6	AC6 V_{INP} > AC6 V_{INN}	PWMA channel 19	PWMA duty cycle value increased by one	
	AC6 V_{INN} > AC6 V_{INP}		PWMA duty cycle value decreased by one	
ACIFB channel 7	AC7 V_{INP} > AC7 V_{INN}	PWMA channel 20	PWMA duty cycle value increased by one	
	AC7 V_{INN} > AC7 V_{INP}		PWMA duty cycle value decreased by one	
ACIFB channel n	ACn V_{INN} > ACn V_{INP}	CAT	Automatically used by the CAT when performing QMatrix acquisition.	No

Table 25-2. Peripheral Event Mapping from GPIO to TC

Generator	Generated Event	User	Effect	Asynchronous
GPIO	Pin change on PA00-PA07	TC0	A0 capture	No
	Pin change on PA08-PA15		A1 capture	
	Pin change on PA16-PA23		A2 capture	
	Pin change on PB00-PB07	TC1	A1 capture	
	Pin change on PB08-PB15		A2 capture	

26.9.1 Control Register

Register Name: CR

Access Type: Write-only

Offset: 0x00

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
-	-	-	-	-	-	DIS	EN
7	6	5	4	3	2	1	0
-	-	-	-	-	-	START	SWRST

- **DIS: ADCDIFB Disable**
  Writing a zero to this bit has no effect.
  Writing a one to this bit disables the ADCIFB.
  Note: Disabling the ADCIFB effectively stops all clocks in the module so the user must make sure the ADCIFB is idle before disabling the ADCIFB.
- **EN: ADCIFB Enable**
  Writing a zero to this bit has no effect.
  Writing a one to this bit enables the ADCIFB.
  Note: The ADCIFB must be enabled before use.
- **START: Start Conversion**
  Writing a zero to this bit has no effect.
  Writing a one to this bit starts an Analog-to-Digital conversion.
- **SWRST: Software Reset**
  Writing a zero to this bit has no effect.
  Writing a one to this bit resets the ADCIFB, simulating a hardware reset.

This bit is set when pen contact is detected and pen detect is enabled.

- **OVRE: Overrun Error Status**
  This bit is cleared when no Overrun Error has occurred since the start of a conversion sequence.
  This bit is set when one or more Overrun Error has occurred since the start of a conversion sequence.
- **DRDY: Data Ready Status**
  0: No data has been converted since the last reset.
  1: One or more conversions have completed since the last reset and data is available in LCDR.
  This bit is cleared when CR.SWRST is written to one.
  This bit is set when one or more conversions have completed and data is available in LCDR.

- Perform a CHIP_ERASE to clear the security bit. **NOTE**: This will erase all the contents of the non-volatile memory.

31.5 JTAG Instruction Summary

The implemented JTAG instructions in the 32-bit AVR are shown in the table below.

Table 31-9. JTAG Instruction Summary

Instruction OPCODE	Instruction	Description
0x01	IDCODE	Select the 32-bit Device Identification register as data register.
0x02	SAMPLE_PRELOAD	Take a snapshot of external pin values without affecting system operation.
0x03	EXTEST	Select boundary-scan chain as data register for testing circuitry external to the device.
0x04	INTEST	Select boundary-scan chain for internal testing of the device.
0x06	CLAMP	Bypass device through Bypass register, while driving outputs from boundary-scan register.
0x0C	AVR_RESET	Apply or remove a static reset to the device
0x0F	CHIP_ERASE	Erase the device
0x10	NEXUS_ACCESS	Select the SAB Address and Data registers as data register for the TAP. The registers are accessed in Nexus mode.
0x11	MEMORY_WORD_ACCESS	Select the SAB Address and Data registers as data register for the TAP.
0x12	MEMORY_BLOCK_ACCESS	Select the SAB Data register as data register for the TAP. The address is auto-incremented.
0x13	CANCEL_ACCESS	Cancel an ongoing Nexus or Memory access.
0x14	MEMORY_SERVICE	Select the SAB Address and Data registers as data register for the TAP. The registers are accessed in Memory Service mode.
0x15	MEMORY_SIZED_ACCESS	Select the SAB Address and Data registers as data register for the TAP.
0x17	SYNC	Synchronization counter
0x1C	HALT	Halt the CPU for safe programming.
0x1F	BYPASS	Bypass this device through the bypass register.
Others	N/A	Acts as BYPASS

31.5.1 Security Restrictions

When the security fuse in the Flash is programmed, the following JTAG instructions are restricted:

- NEXUS_ACCESS
- MEMORY_WORD_ACCESS
- MEMORY_BLOCK_ACCESS
- MEMORY_SIZED_ACCESS

For description of what memory locations remain accessible, please refer to the SAB address map.

Full access to these instructions is re-enabled when the security fuse is erased by the CHIP_ERASE JTAG instruction.

Note that the security bit will read as programmed and block these instructions also if the Flash Controller is statically reset.

Other security mechanisms can also restrict these functions. If such mechanisms are present they are listed in the SAB address map section.

31.5.1.1 Notation

Table 31-11 on page 737 shows bit patterns to be shifted in a format like "**peb01**". Each character corresponds to one bit, and eight bits are grouped together for readability. The least significantbit is always shifted first, and the most significant bit shifted last. The symbols used are shown in Table 31-10.

Table 31-10. Symbol Description

Symbol	Description
0	Constant low value - always reads as zero.
1	Constant high value - always reads as one.
a	An address bit - always scanned with the least significant bit first
b	A busy bit. Reads as one if the SAB was busy, or zero if it was not. See Section 31.4.11.4 for details on how the busy reporting works.
d	A data bit - always scanned with the least significant bit first.
e	An error bit. Reads as one if an error occurred, or zero if not. See Section 31.4.11.5 for details on how the error reporting works.
p	The chip protected bit. Some devices may be set in a protected state where access to chip internals are severely restricted. See the documentation for the specific device for details. On devices without this possibility, this bit always reads as zero.
r	A direction bit. Set to one to request a read, set to zero to request a write.
s	A size bit. The size encoding is described where used.
x	A don't care bit. Any value can be shifted in, and output data should be ignored.

In many cases, it is not required to shift all bits through the data register. Bit patterns are shown using the full width of the shift register, but the suggested or required bits are emphasized using **bold** text. I.e. given the pattern "**aaaaaaar** xxxxxxxx xxxxxxxx xxxxxxxx xx", the shift register is 34 bits, but the test or debug unit may choose to shift only 8 bits "**aaaaaaar**".

The following describes how to interpret the fields in the instruction description tables:

Table 31-11. Instruction Description

Instruction	Description
IR input value	Shows the bit pattern to shift into IR in the Shift-IR state in order to select this instruction. The pattern is show both in binary and in hexadecimal form for convenience. Example: **10000** (0x10)
IR output value	Shows the bit pattern shifted out of IR in the Shift-IR state when this instruction is active. Example: peb01

Table 32-42. TWI-Bus Timing Requirements

Symbol	Parameter	Mode	Minimum		Maximum		Unit
			Requirement	Device	Requirement	Device	
$t_{SU-DAT-TWI}$	Data set-up time	Standard	250	$2t_{clkpb}$	-		ns
		Fast	100				
t_{SU-DAT}		-	-	t_{clkpb}	-		-
$t_{LOW-TWI}$	TWCK LOW period	Standard	4.7	$4t_{clkpb}$	-		µs
		Fast	1.3				
t_{LOW}		-	-	t_{clkpb}	-		-
t_{HIGH}	TWCK HIGH period	Standard	4.0	$8t_{clkpb}$	-		µs
		Fast	0.6				
f_{TWCK}	TWCK frequency	Standard	-		100	$\frac{1}{12t_{clkpb}}$	kHz
		Fast			400		

Notes: 1. Standard mode: $f_{TWCK} \leq 100$ kHz ; fast mode: $f_{TWCK} > 100$ kHz .
2. A device must internally provide a hold time of at least 300 ns for TWD with reference to the falling edge of TWCK.

Notations:

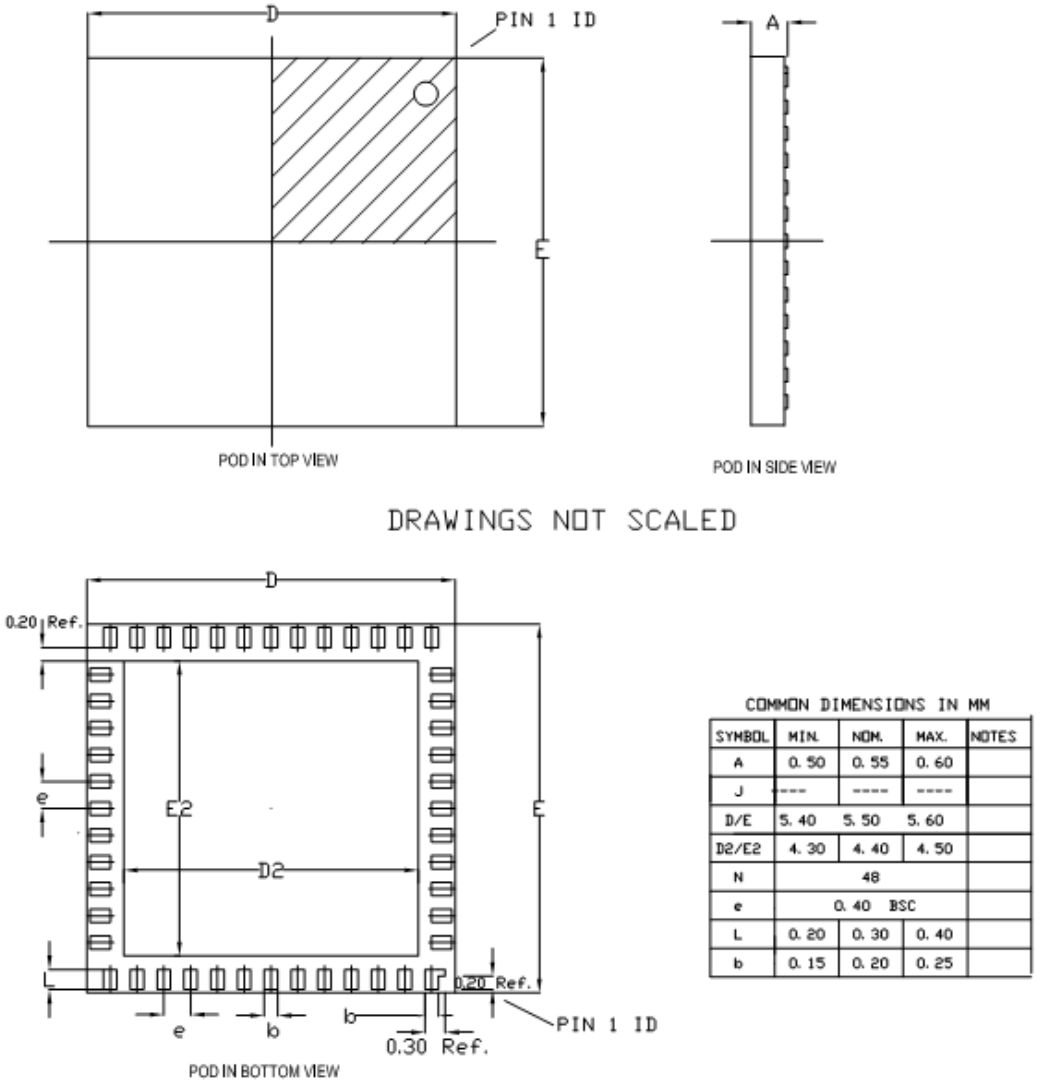
C_b = total capacitance of one bus line in pF

t_{clkpb} = period of TWI peripheral bus clock

$t_{prescaled}$ = period of TWI internal prescaled clock (see chapters on TWIM and TWIS)

The maximum $t_{HD;DAT}$ has only to be met if the device does not stretch the LOW period ($t_{LOW-TWI}$) of TWCK.

Figure 33-3. TLLGA-48 Package Drawing



COMMON DIMENSIONS IN MM

SYMBOL	MIN.	NOM.	MAX.	NOTES
A	0.50	0.55	0.60	
J	----	----	----	
D/E	5.40	5.50	5.60	
D2/E2	4.30	4.40	4.50	
N	48			
e	0.40 BSC			
L	0.20	0.30	0.40	
b	0.15	0.20	0.25	

Table 33-8. Device and Package Maximum Weight

39.3	mg

Table 33-9. Package Characteristics

Moisture Sensitivity Level	MSL3

Table 33-10. Package Reference

JEDEC Drawing Reference	N/A
JESD97 Classification	E4