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"

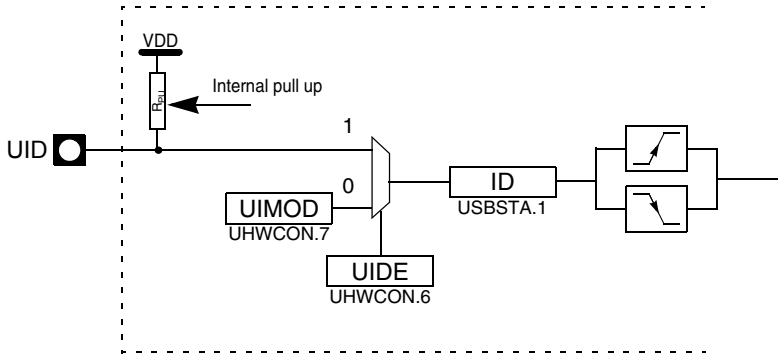
Details	
Product Status	Obsolete
Core Processor	AVR
Core Size	8-Bit
Speed	16MHz
Connectivity	EBI/EMI, I²C, SPI, UART/USART, USB, USB OTG
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	48
Program Memory Size	128KB (128K x 8)
Program Memory Type	FLASH
EEPROM Size	4K x 8
RAM Size	8K x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	64-TQFP
Supplier Device Package	64-TQFP (14x14)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/at90usb1287-16aur

  - Real time counter with separate oscillator
  - Four 8-bit PWM channels
  - Six PWM channels with programmab le resolution from 2 to 16 bits
  - Output compare modulator
  - 8-channels, 10-bit ADC
  - Programmable serial USART
  - Master/slave SPI serial interface
  - Byte oriented 2-wi re serial interface
  - Programmable watchdog timer with separate on-chip oscillator
  - On-chip analog comparator
  - Interrupt and wake-up on pin change
- Special microcontroller features
  - Power-on reset and programmable brown-out detection
  - Internal calibra ted oscillator
  - External and internal interrupt sources
  - Six sleep modes: Idle, ADC Noise Reduction, Powe r-save, Power-down, Standby, and Extended Standby
- I/O and packages
  - 48 programmable I/O lines
  - 64-lead TQFP and 64-lead QFN
- Operating voltages
  - 2.7 - 5.5V
- Operating temperature
  - Industrial (- 40°C to +85°C)
- Maximum frequency
  - 8MHz at 2.7V - industrial range
  - 16MHz at 4.5V - industrial range

22.11 ID detection

The ID pin transition is detected thanks to the following architecture:

Figure 22-17. ID detection input block diagram.



The ID pin can be used to detect the USB mode (Peripheral or Host) or software selected. This allows the UID pin to be used has general purpose I/O even when USB interface is enable. When the UID pin is selected, by default, (no A-plug or B-plug), the macro is in the Peripheral mode (internal pull-up). The IDTI interrupt is triggered when a A-plug (Host) is plugged or unplugged. The interrupt is not triggered when a B-plug (Periph) is plugged or unplugged.

ID detection is independent of USB global interface enable.

22.12 Registers description

22.12.1 USB general registers

Bit	7	6	5	4	3	2	1	0	
	UIMOD	**UIDE**		**UVCONE**				**UVREGE**	**UHWCON**
Read/write	R/W	R/W	R	R/W	R	R	R	R/W	
Initial value	1	0	0	0	0	0	0	0	

- **7 – UIMOD: USB Mode bit**

This bit has no effect when the UIDE bit is set (external UID pin activated). Set to enable the USB device mode. Clear to enable the USB host mode

- **6 – UIDE: UID pin Enable**

Set to enable the USB mode selection (peripheral/host) through the UID pin. Clear to enable the USB mode selection (peripheral/host) with UIMOD bit register.

UIDE should be modified only when the USB interface is disabled (USBE bit cleared).

- **5 – Reserved**

The value read from this bit is always 0. Do not set this bit.

- **4 – UVCONE: UVCON pin Enable**

Set to enable the UVCON pin control. Clear to disable the UVCON pin control. This bit should be set only when the USB interface is enable.

- **4 – SRPREQ: SRP Request bit**

Set to initiate the SRP when the controller is in Device mode. Cleared by hardware when the controller is initiating a SRP.

- **3 – SRPSEL: SRP Selection bit**

Set to choose VBUS pulsing as SRP method.
Clear to choose data line pulsing as SRP method.

- **2 – VBUSHWC: VBus Hardware Control bit**

Set to disable the hardware control over the UVCON pin.

Clear to enable the hardware control over the UVCON pin.

See for more details

- **1 – VBUSREQ: VBUS Request bit**

Set to assert the UVCON pin in order to enable the VBUS power supply generation. This bit shall be used when the controller is in the Host mode.

Cleared by hardware when VBUSRQC is set.

- **0 – VBUSRQC: VBUS Request Clear bit**

Set to deassert the UVCON pin in order to enable the VBUS power supply generation. This bit shall be used when the controller is in the Host mode.

Cleared by hardware immediately after the set.

Bit	7	6	5	4	3	2	1	0	
	-	PAGE		-	-	-	VALUE		OTGTCON
Read/write	R	R/W	R/W	R	R	R/W	R/W	R/W	
Initial value	1	0	0	0	0	0	0	0	

- **7 – Reserved**

This bit is reserved and always set.

- **6-5 – PAGE: Timer page access bit**

Set/clear to access a special timer register. See Section 22.9, page 254 for more details.

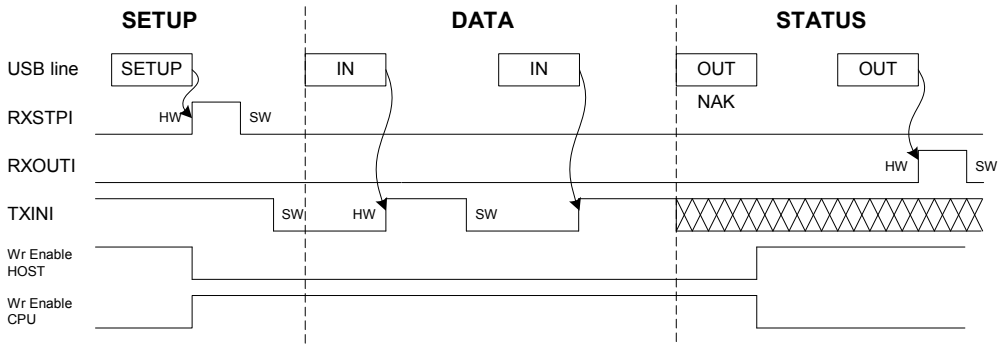
- **4-3 - Reserved**

The value read from these bits is always 0. Do not set these bits.

- **1-0 – VALUE: Value bit**

Set to initialize the new value of the timer. See Section 22.9, page 254 for more details.

Figure 23-5. Control read transaction.



A NAK handshake is always generated at the first status stage command.

When the controller detect the status stage, all the data writen by the CPU are erased, and clearing TXINI has no effects.

The firmware checks if the transmission is complete or if the reception is complete.

The OUT retry is always ack'ed. This reception:

- set the RXOUTI flag (received OUT data)

- set the TXINI flag (data sent, ready to accept new data)

software algorithm:

```
set transmit ready
wait (transmit complete OR Receive complete)
if receive complete, clear flag and return
if transmit complete, continue
```

Once the OUT status stage has been received, the USB controller waits for a SETUP request. The SETUP request have priority over any other request and has to be ACK'ed. This means that any other flag should be cleared and the fifo reset when a SETUP is received.

WARNING: the byte counter is reset when the OUT Zero Length Packet is received. The firmware has to take care of this.

23.13 OUT endpoint management

OUT packets are sent by the host. All the data can be read by the CPU, which acknowledges or not the bank when it is empty.

23.13.1 Overview

The Endpoint must be configured first.

Each time the current bank is full, the RXOUTI and the FIFOCON bits are set. This triggers an interrupt if the RXOUTE bit is set. The firmware can acknowledge the USB interrupt by clearing the RXOUTI bit. The Firmware read the data and clear the FIFOCON bit in order to free the current bank. If the OUT Endpoint is composed of multiple banks, clearing the FIFOCON bit will switch to the next bank. The RXOUTI and FIFOCON bits are then updated by hardware in accordance with the status of the new bank.

23.14.1 Overview

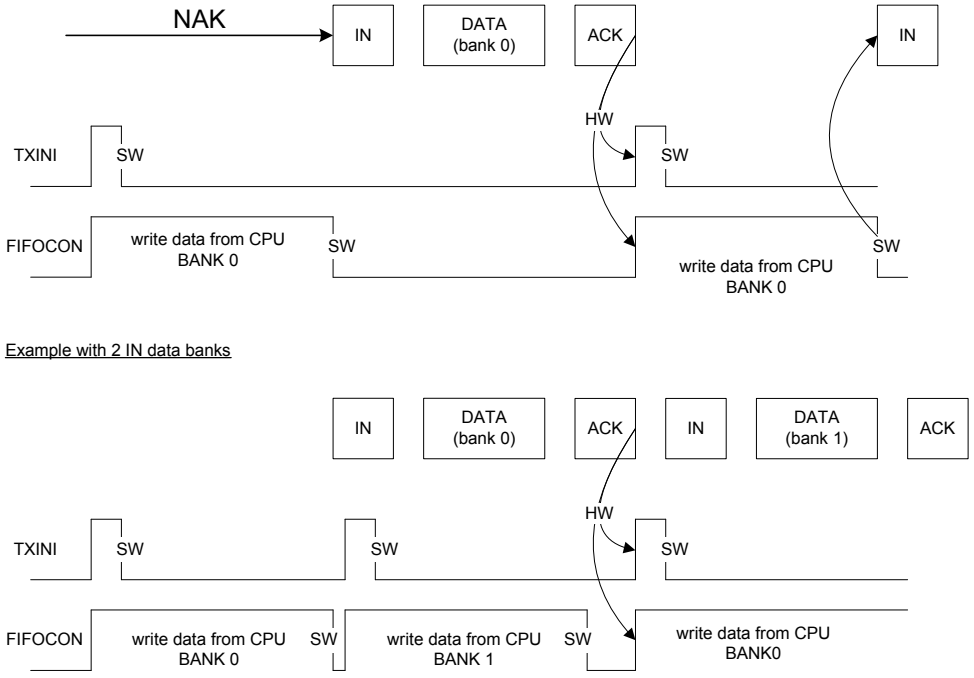
The Endpoint must be configured first.

The TXINI bit is set by hardware when the current bank becomes free. This triggers an interrupt if the TXINE bit is set. The FIFOCON bit is set at the same time. The CPU writes into the FIFO and clears the FIFOCON bit to allow the USB controller to send the data. If the IN Endpoint is composed of multiple banks, this also switches to the next data bank. The TXINI and FIFOCON bits are automatically updated by hardware regarding the status of the next bank.

TXINI shall always be cleared before clearing FIFOCON.

The RWAL bit always reflects the state of the current bank. This bit is set if the firmware can write data to the bank, and cleared by hardware when the bank is full.

Figure 23-7. Example with 1 and 2 IN data bank.

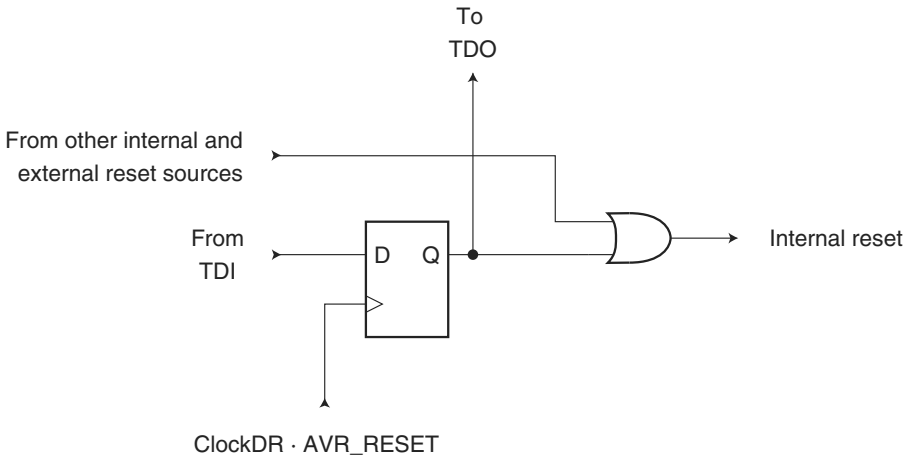


23.14.2 Detailed description

The data are written by the CPU, following the next flow:

- When the bank is empty, an endpoint interrupt (EPINTx) is triggered, if enabled (TXINE set) and TXINI is set. The CPU can also poll TXINI or FIFOCON, depending the software architecture choice
- The CPU acknowledges the interrupt by clearing TXINI
- The CPU can write the data into the current bank (write in UEDATX)
- The CPU can free the bank by clearing FIFOCON when all the data are written, that is:
  - after “N” write into UEDATX
  - as soon as RWAL is cleared by hardware

Figure 28-2. Reset register.



28.3.4 Boundary-scan Chain

The Boundary-scan Chain has the capability of driving and observing the logic levels on the digital I/O pins, as well as the boundary between digital and analog logic for analog circuitry having off-chip connections.

See “Boundary-scan chain” on page 337 for a complete description.

28.4 Boundary-scan specific JTAG instructions

The Instruction Register is 4-bit wide, supporting up to 16 instructions. Listed below are the JTAG instructions useful for Boundary-scan operation. Note that the optional HIGHZ instruction is not implemented, but all outputs with tri-state capability can be set in high-impedant state by using the AVR_RESET instruction, since the initial state for all port pins is tri-state.

As a definition in this datasheet, the LSB is shifted in and out first for all Shift Registers.

The OPCODE for each instruction is shown behind the instruction name in hex format. The text describes which Data Register is selected as path between TDI and TDO for each instruction.

28.4.1 EXTEST; 0x0

Mandatory JTAG instruction for selecting the Boundary-scan Chain as Data Register for testing circuitry external to the AVR package. For port-pins, Pull-up Disable, Output Control, Output Data, and Input Data are all accessible in the scan chain. For Analog circuits having off-chip connections, the interface between the analog and the digital logic is in the scan chain. The contents of the latched outputs of the Boundary-scan chain is driven out as soon as the JTAG IR-Register is loaded with the EXTEST instruction.

The active states are:

- Capture-DR: Data on the external pins are sampled into the Boundary-scan Chain
- Shift-DR: The Internal Scan Chain is shifted by the TCK input
- Update-DR: Data from the scan chain is applied to output pins

28.4.2 IDCODE; 0x1

Optional JTAG instruction selecting the 32-bit ID-Register as Data Register. The ID-Register consists of a version number, a device number and the manufacturer code chosen by JEDEC. This is the default instruction after power-up.

Table 29-4. Boot reset fuse (1).

BOOTRST	Reset address
1	Reset Vector = Application reset (address 0x0000)
0	Reset Vector = Boot loader reset (see Table 29-8 on page 357)

Note: 1. “1” means unprogrammed, “0” means programmed.

29.5.3 External hardware conditions

The Hardware Boot Enable Fuse (HWBE) can be programmed (see Table 29-5) so that upon special hardware conditions under reset, the boot loader execution is forced after reset.

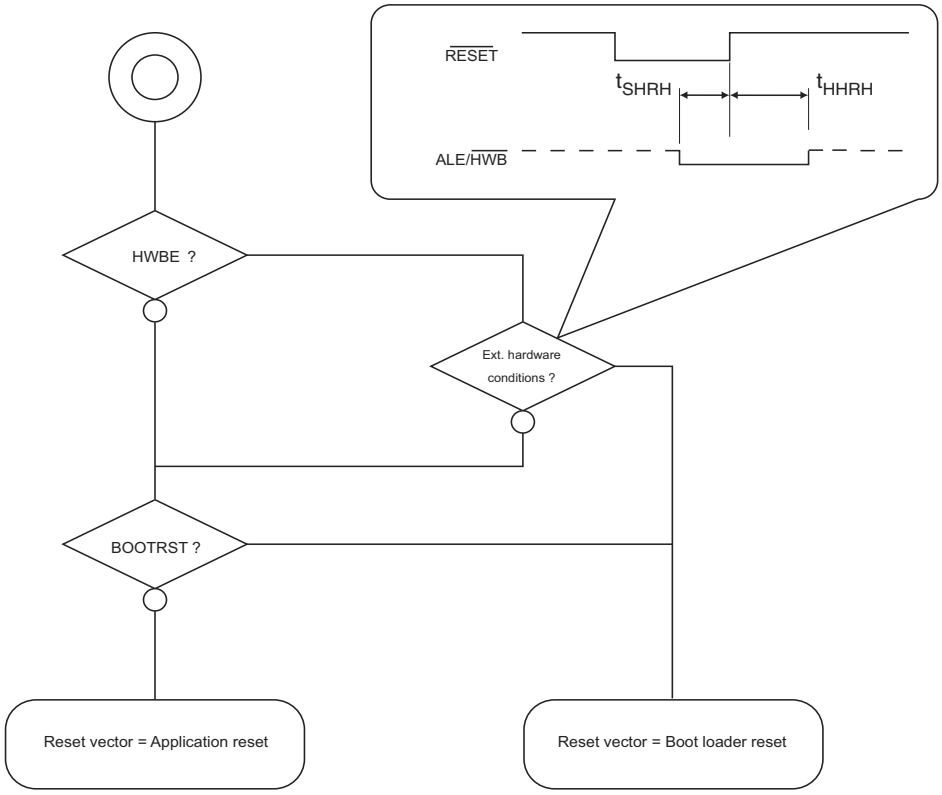
Table 29-5. Hardware boot enable fuse (1).

HWBE	Reset address
1	ALE/$\overline{\text{HWB}}$ pin can not be used to force boot loader execution after reset
0	ALE/$\overline{\text{HWB}}$ pin is used during reset to force boot loader execution after reset

Note: 1. “1” means unprogrammed, “0” means programmed.

When the HWBE fuse is enable the ALE/$\overline{\text{HWB}}$ pin is configured as input during reset and sampled during reset rising edge. When ALE/$\overline{\text{HWB}}$ pin is ‘0’ during reset rising edge, the reset vector will be set as the Boot Loader Reset address and the Boot Loader will be executed (see Figure 29-3).

Figure 29-3. Boot process description.



page. See “Simple Assembly Code example for a Boot Loader” on page 355 for an assembly code example.

29.7.1 Performing page erase by SPM

To execute Page Erase, set up the address in the Z-pointer, write “X0000011” to SPMCSR and execute SPM within four clock cycles after writing SPMCSR. The data in R1 and R0 is ignored. The page address must be written to PCPAGE in the Z-register. Other bits in the Z-pointer will be ignored during this operation.

- Page Erase to the RWW section: The NRWW section can be read during the Page Erase
- Page Erase to the NRWW section: The CPU is halted during the operation

29.7.2 Filling the Temporary Buffer (page loading)

To write an instruction word, set up the address in the Z-pointer and data in R1:R0, write “00000001” to SPMCSR and execute SPM within four clock cycles after writing SPMCSR. The content of PCWORD in the Z-register is used to address the data in the temporary buffer. The temporary buffer will auto-erase after a Page Write operation or by writing the RWWSRE bit in SPMCSR. It is also erased after a system reset. Note that it is not possible to write more than one time to each address without erasing the temporary buffer.

If the EEPROM is written in the middle of an SPM Page Load operation, all data loaded will be lost.

29.7.3 Performing a Page Write

To execute Page Write, set up the address in the Z-pointer, write “X0000101” to SPMCSR and execute SPM within four clock cycles after writing SPMCSR. The data in R1 and R0 is ignored. The page address must be written to PCPAGE. Other bits in the Z-pointer must be written to zero during this operation.

- Page Write to the RWW section: The NRWW section can be read during the Page Write
- Page Write to the NRWW section: The CPU is halted during the operation

29.7.4 Using the SPM interrupt

If the SPM interrupt is enabled, the SPM interrupt will generate a constant interrupt when the SPMEN bit in SPMCSR is cleared. This means that the interrupt can be used instead of polling the SPMCSR Register in software. When using the SPM interrupt, the Interrupt Vectors should be moved to the BLS section to avoid that an interrupt is accessing the RWW section when it is blocked for reading. How to move the interrupts is described in “Interrupts” on page 68.

29.7.5 Consideration while updating BLS

Special care must be taken if the user allows the Boot Loader section to be updated by leaving Boot Lock bit11 unprogrammed. An accidental write to the Boot Loader itself can corrupt the entire Boot Loader, and further software updates might be impossible. If it is not necessary to change the Boot Loader software itself, it is recommended to program the Boot Lock bit11 to protect the Boot Loader software from any internal software changes.

29.7.6 Prevent reading the RWW section during self-programming

During Self-Programming (either Page Erase or Page Write), the RWW section is always blocked for reading. The user software itself must prevent that this section is addressed during the self programming operation. The RWWSB in the SPMCSR will be set as long as the RWW section is busy. During Self-Programming the Interrupt Vector table should be moved to the BLS

30.9.8 Reset Register

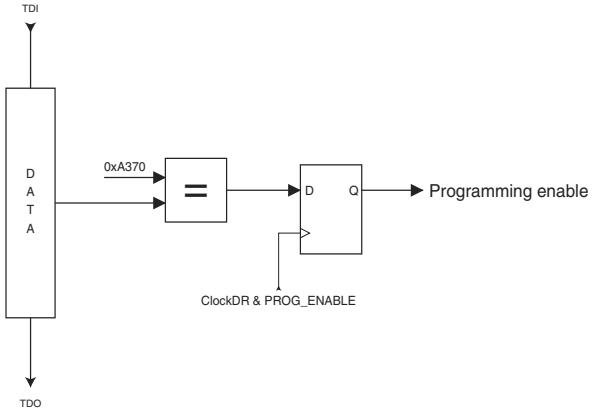
The Reset Register is a Test Data Register used to reset the part during programming. It is required to reset the part before entering Programming mode.

A high value in the Reset Register corresponds to pulling the external reset low. The part is reset as long as there is a high value present in the Reset Register. Depending on the Fuse settings for the clock options, the part will remain reset for a Reset Time-out period (refer to “Clock sources” on page 41) after releasing the Reset Register. The output from this Data Register is not latched, so the reset will take place immediately, as shown in Figure 9-1 on page 58.

30.9.9 Programming Enable Register

The Programming Enable Register is a 16-bit register. The contents of this register is compared to the programming enable signature, binary code 0b1010_0011_0111_0000. When the contents of the register is equal to the programming enable signature, programming via the JTAG port is enabled. The register is reset to 0 on Power-on Reset, and should always be reset when leaving Programming mode.

Figure 30-13. Programming enable register.



30.9.10 Programming Command Register

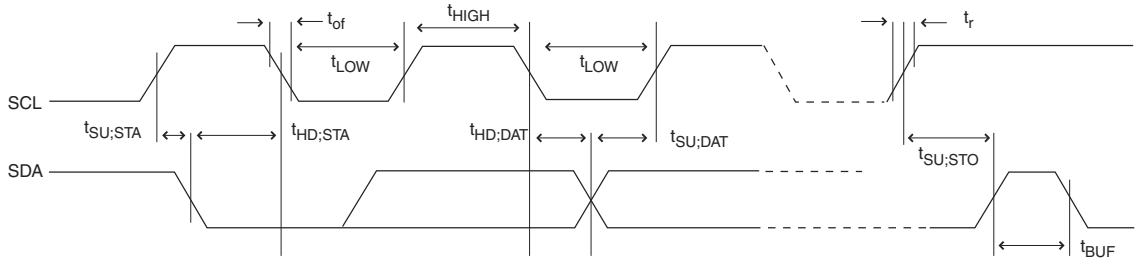
The Programming Command Register is a 15-bit register. This register is used to serially shift in programming commands, and to serially shift out the result of the previous command, if any. The JTAG Programming Instruction Set is shown in Table 30-17 on page 382. The state sequence when shifting in the programming commands is illustrated in Figure 30-15 on page 385.

Table 31-2. 2-wire serial bus requirements. (Continued)

Symbol	Parameter	Condition	Min	Max	Units
$t_{HD;STA}$	Hold Time (repeated) START Condition	$f_{SCL} \leq 100kHz$	4.0	–	µs
		$f_{SCL} > 100kHz$	0.6	–	
t_{LOW}	Low Period of the SCL Clock	$f_{SCL} \leq 100kHz$ (6)	4.7	–	
		$f_{SCL} > 100kHz$ (7)	1.3	–	
t_{HIGH}	High period of the SCL clock	$f_{SCL} \leq 100kHz$	4.0	–	
		$f_{SCL} > 100kHz$	0.6	–	
$t_{SU;STA}$	Set-up time for a repeated START condition	$f_{SCL} \leq 100kHz$	4.7	–	
		$f_{SCL} > 100kHz$	0.6	–	
$t_{HD;DAT}$	Data hold time	$f_{SCL} \leq 100kHz$	0	3.45	
		$f_{SCL} > 100kHz$	0	0.9	
$t_{SU;DAT}$	Data setup time	$f_{SCL} \leq 100kHz$	250	–	ns
		$f_{SCL} > 100kHz$	100	–	
$t_{SU;STO}$	Setup time for STOP condition	$f_{SCL} \leq 100kHz$	4.0	–	µs
		$f_{SCL} > 100kHz$	0.6	–	
t_{BUF}	Bus free time between a STOP and START condition	$f_{SCL} \leq 100kHz$	4.7	–	
		$f_{SCL} > 100kHz$	1.3	–	

Notes:
1. In Atmel AT90USB64/128, this parameter is characterized and not 100% tested.
2. Required only for f_{SCL} >100kHz.
3. C_b = capacitance of one bus line in pF.
4. f_{CK} = CPU clock frequency
5. This requirement applies to all AT90USB64/128 2-wire Serial Interface operation. Other devices connected to the 2-wire Serial Bus need only obey the general f_{SCL} requirement.
6. The actual low period generated by the AT90USB64/128 2-wire Serial Interface is ($1/f_{SCL} - 2/f_{CK}$), thus f_{CK} must be greater than 6MHz for the low time requirement to be strictly met at f_{SCL} = 100kHz.
7. The actual low period generated by the AT90USB64/128 2-wire Serial Interface is ($1/f_{SCL} - 2/f_{CK}$), thus the low time requirement will not be strictly met for f_{SCL} > 308kHz when f_{CK} = 8MHz. Still, AT90USB64/128 devices connected to the bus may communicate at full speed (400kHz) with other AT90USB64/128 devices, as well as any other device with a proper t_{LOW} acceptance margin.

Figure 31-3. 2-wire serial bus timing.



28.4 Boundary-scan specific JTAG instructions 335
28.5 Boundary-scan Related Register in I/O memory 336
28.6 Boundary-scan chain 337
28.7 Atmel AT90USB64/128 Boundary-scan order 340
28.8 Boundary-scan description language files 342

29 Boot Loader support – read-while-write self-programming 343

29.1 Boot Loader features 343
29.2 Application and Boot Loader flash sections 343
29.3 Read-while-write and no read-while-write flash sections 343
29.4 Boot Loader lock bits 346
29.5 Entering the Boot Loader program 347
29.6 Addressing the flash during self-programming 350
29.7 Self-programming the flash 351

30 Memory programming 359

30.1 Program and data memory lock bits 359
30.2 Fuse bits 360
30.3 Signature bytes 362
30.4 Calibration byte 362
30.5 Parallel programming parameters, pin mapping, and commands 362
30.6 Parallel programming 365
30.7 Serial downloading 373
30.8 Serial programming pin mapping 374
30.9 Programming via the JTAG interface 377

31 Electrical characteristics for Atmel AT90USB64/128 390

31.1 Absolute maximum ratings* 390
31.2 DC characteristics 390
31.3 External clock drive waveforms 392
31.4 External clock drive 392
31.5 Maximum speed vs. V_{CC} 392
31.6 2-wire serial interface characteristics 393
31.7 SPI timing characteristics 395
31.8 Hardware boot entrance timing characteristics 396
31.9 ADC characteristics 397
31.10 External data memory timing 399

32 Atmel AT90USB64/128 typical characteristics 404