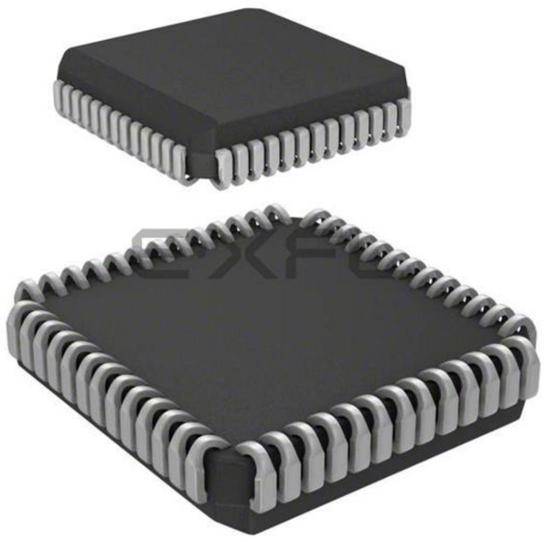
Microchip Technology - AT89C5131A-S3SIM Datasheet



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	80C51
Core Size	8-Bit
Speed	48MHz
Connectivity	I²C, SPI, UART/USART, USB
Peripherals	LED, POR, PWM, WDT
Number of I/O	34
Program Memory Size	32KB (32K x 8)
Program Memory Type	FLASH
EEPROM Size	4K x 8
RAM Size	1.25K x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	-
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	52-LCC (J-Lead)
Supplier Device Package	52-PLCC (19.15x19.15)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/at89c5131a-s3sim

Description

AT89C5130A/31A-M is a high-performance Flash version of the 80C51 single-chip 8-bit microcontrollers with full speed USB functions.

AT89C5130A/31A-M features a full-speed USB module compatible with the USB specifications Version 1.1 and 2.0. This module integrates the USB transceivers with a 3.3V voltage regulator and the Serial Interface Engine (SIE) with Digital Phase Locked Loop and 48 MHz clock recovery. USB Event detection logic (Reset and Suspend/Resume) and FIFO buffers supporting the mandatory control Endpoint (EP0) and up to 6 versatile Endpoints (EP1/EP2/EP3/EP4/EP5/EP6) with minimum software overhead are also part of the USB module.

AT89C5130A/31A-M retains the features of the Atmel 80C52 with extended Flash capacity (16/32-Kbytes), 256 bytes of internal RAM, a 4-level interrupt system, two 16-bit timer/counters (T0/T1), a full duplex enhanced UART (EUART) and an on-chip oscillator.

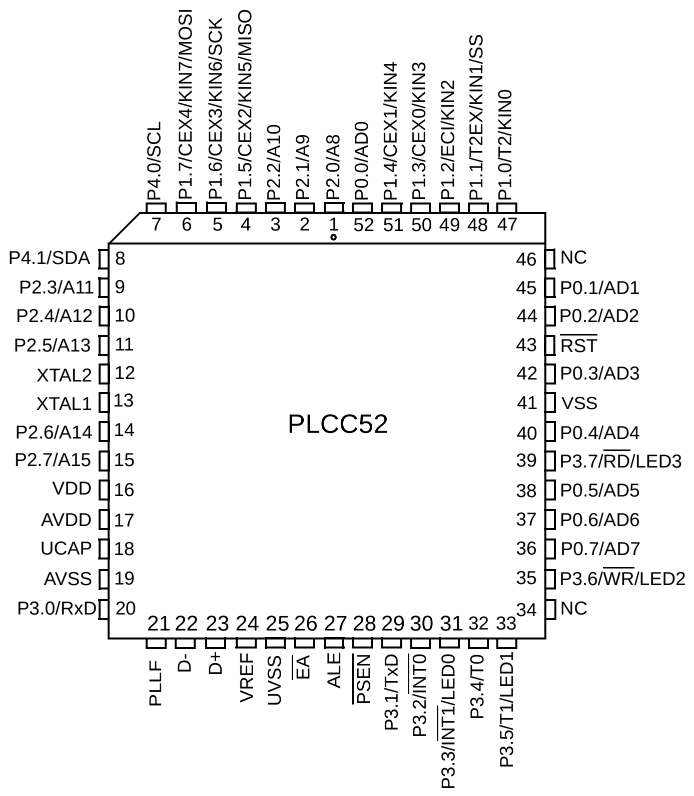
In addition, AT89C5130A/31A-M has an on-chip expanded RAM of 1024 bytes (ERAM), a dual data pointer, a 16-bit up/down Timer (T2), a Programmable Counter Array (PCA), up to 4 programmable LED current sources, a programmable hardware watchdog and a power-on reset.

AT89C5130A/31A-M has two software-selectable modes of reduced activity for further reduction in power consumption. In the idle mode the CPU is frozen while the timers, the serial ports and the interrupt system are still operating. In the power-down mode the RAM is saved, the peripheral clock is frozen, but the device has full wake-up capability through USB events or external interrupts.

Pinout Description

Pinout

Figure 1. AT89C5130A/31A-M 52-pin PLCC Pinout



Dual Data Pointer Register

The additional data pointer can be used to speed up code execution and reduce code size.

The dual DPTR structure is a way by which the chip will specify the address of an external data memory location. There are two 16-bit DPTR registers that address the external memory, and a single bit called DPS = AUXR1.0 (see Table 34) that allows the program code to switch between them (see Figure 10).

Figure 10. Use of Dual Pointer

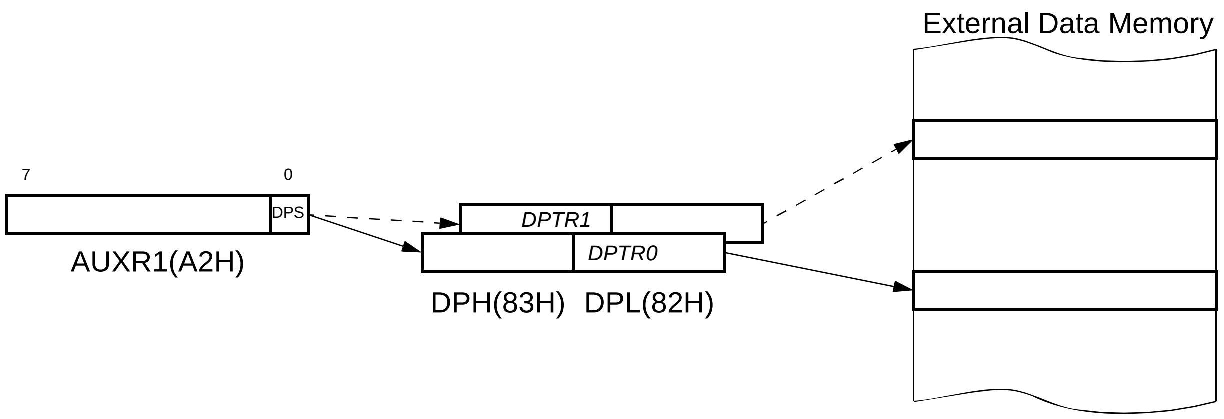


Table 34. AUXR1 Register
AUXR1- Auxiliary Register 1(0A2h)

7	6	5	4	3	2	1	0
-	-	ENBOOT	-	GF3	0	-	DPS

Bit Number	Bit Mnemonic	Description
7	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
6	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
5	ENBOOT	**Enable Boot Flash** Cleared to disable boot ROM. Set to map the boot ROM between F800h - 0FFFFh.
4	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
3	GF3	This bit is a general-purpose user flag.
2	0	Always cleared.
1	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
0	DPS	**Data Pointer Selection** Cleared to select DPTR0. Set to select DPTR1.

Reset Value = XX[BLJB]X X0X0b
Not bit addressable

a. Bit 2 stuck at 0; this allows to use INC AUXR1 to toggle DPS without changing GF3.

The other memory spaces (user, extra row, hardware security) are made accessible in the code segment by programming bits FMOD0 and FMOD1 in FCON register in accordance with Table 36. A MOVC instruction is then used for reading these spaces.

Table 36. FM0 Blocks Select Bits

FMOD1	FMOD0	FM0 Adressable Space
0	0	User (0000h-FFFFh)
0	1	Extra Row(FF80h-FFFFh)
1	0	Hardware Security (0000h)
1	1	reserved

Launching Programming

FPL3:0 bits in FCON register are used to secure the launch of programming. A specific sequence must be written in these bits to unlock the write protection and to launch the programming. This sequence is 5 followed by A. Table 37 summarizes the memory spaces to program according to FMOD1:0 bits.

Table 37. Programming Spaces

	Write to FCON				
	FPL3:0	FPS	FMOD1	FMOD0	Operation
User	5	X	0	0	No action
	A	X	0	0	Write the column latches in user space
Extra Row	5	X	0	1	No action
	A	X	0	1	Write the column latches in extra row space
Security Space	5	X	1	0	No action
	A	X	1	0	Write the fuse bits space
Reserved	5	X	1	1	No action
	A	X	1	1	No action

The Flash memory enters a busy state as soon as programming is launched. In this state, the memory is not available for fetching code. Thus to avoid any erratic execution during programming, the CPU enters Idle mode. Exit is automatically performed at the end of programming.

Note: Interrupts that may occur during programming time must be disabled to avoid any spurious exit of the idle mode.

Status of the Flash Memory

The bit FBUSY in FCON register is used to indicate the status of programming.

FBUSY is set when programming is in progress.

Selecting FM0/FM1

The bit ENBOOT in AUXR1 register is used to choose between FM0 and FM1 mapped up to F800h.

In-System Programming (ISP)

With the implementation of the User Space (FM0) and the Boot Space (FM1) in Flash technology the AT89C5130A/31A-M allows the system engineer the development of applications with a very high level of flexibility. This flexibility is based on the possibility to alter the customer program at any stages of a product's life:

- Before mounting the chip on the PCB, FM0 flash can be programmed with the application code. FM1 is always preprogrammed by Atmel with a USB bootloader.(1)
- Once the chip is mounted on the PCB, it can be programmed by serial mode via the USB bus.

Note: 1. The user can also program his own bootloader in FM1.

This ISP allows code modification over the total lifetime of the product.

Besides the default Bootloaders Atmel provide customers all the needed Application-Programming-Interfaces (API) which are needed for the ISP. The API are located in the Boot memory.

This allow the customer to have a full use of the 32-Kbyte user memory.

Flash Programming and Erasure

There are three methods for programming the Flash memory:

- The Atmel bootloader located in FM1 is activated by the application. Low level API routines (located in FM1)will be used to program FM0. The interface used for serial downloading to FM0 is the USB. API can be called also by user's bootloader located in FM0 at [SBV]00h.
- A further method exist in activating the Atmel boot loader by hardware activation. See the Section "Hardware Registers".
- The FM0 can be programmed also by the parallel mode using a programmer.

Figure 20. Flash Memory Mapping

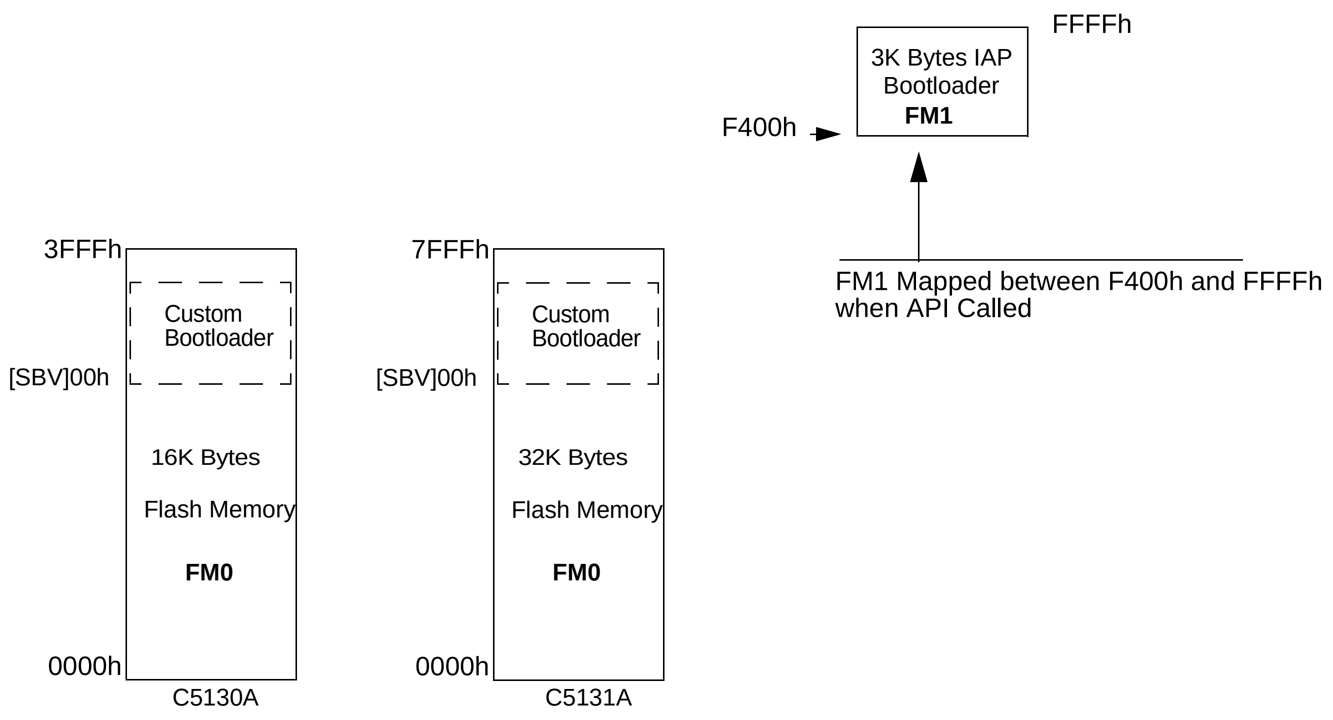
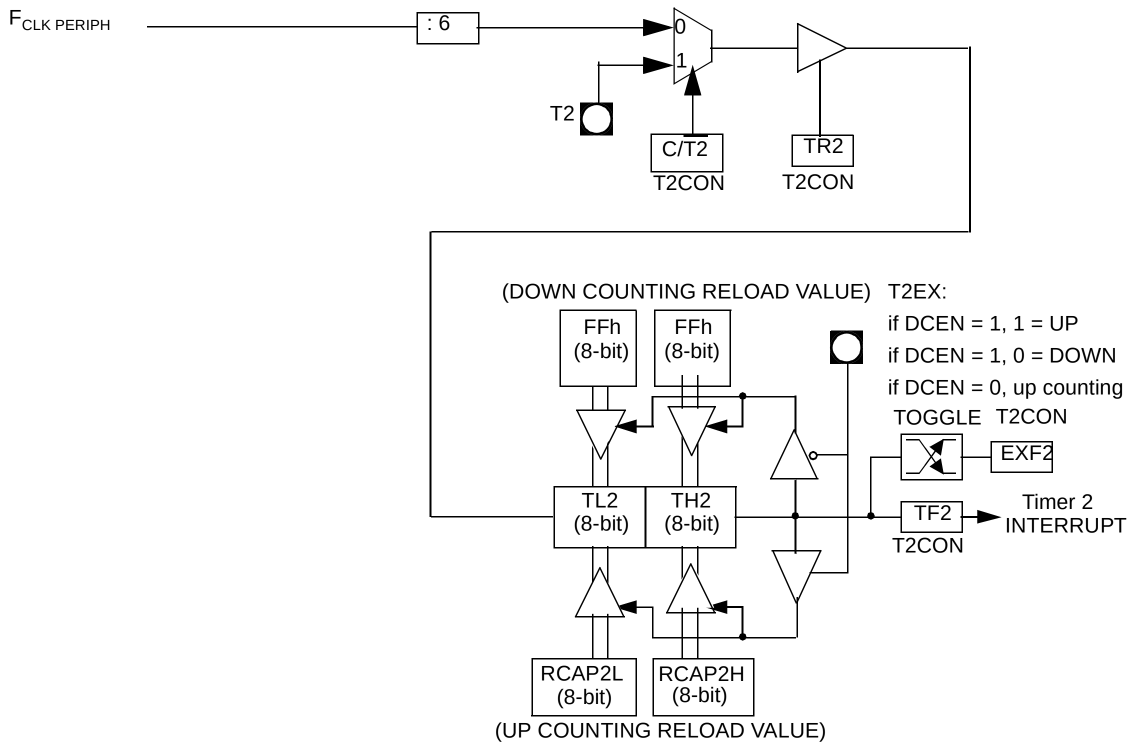


Figure 23. Auto-reload Mode Up/Down Counter (DCEN = 1)



Programmable Clock Output

In the Clock-out mode, Timer 2 operates as a 50%-duty-cycle, programmable clock generator (See Figure 24). The input clock increments TL2 at frequency $F_{CLK\ PERIPH}/2$. The timer repeatedly counts to overflow from a loaded value. At overflow, the contents of RCAP2H and RCAP2L registers are loaded into TH2 and TL2. In this mode, Timer 2 overflows do not generate interrupts. The following formula gives the Clock-out frequency as a function of the system oscillator frequency and the value in the RCAP2H and RCAP2L registers

$$Clock - OutFrequency = \frac{F_{CLKPERIPH}}{4 \times (65536 - RCAP2H/RCAP2L)}$$

For a 16 MHz system clock, Timer 2 has a programmable frequency range of 61 Hz ($F_{CLK\ PERIPH}/2^{16)}$ to 4 MHz ($F_{CLK\ PERIPH}/4$). The generated clock signal is brought out to T2 pin (P1.0).

Timer 2 is programmed for the Clock-out mode as follows:

- Set T2OE bit in T2MOD register.
- Clear C/$\overline{\text{T2}}$ bit in T2CON register.
- Determine the 16-bit reload value from the formula and enter it in RCAP2H/RCAP2L registers.
- Enter a 16-bit initial value in timer registers TH2/TL2. It can be the same as the reload value or a different one depending on the application.
- To start the timer, set TR2 run control bit in T2CON register.

Table 65. IPH0 Register

IPH0 - Interrupt Priority High Register (B7h)

7	6	5	4	3	2	1	0
-	PPCH	PT2H	PSH	PT1H	PX1H	PT0H	PX0H

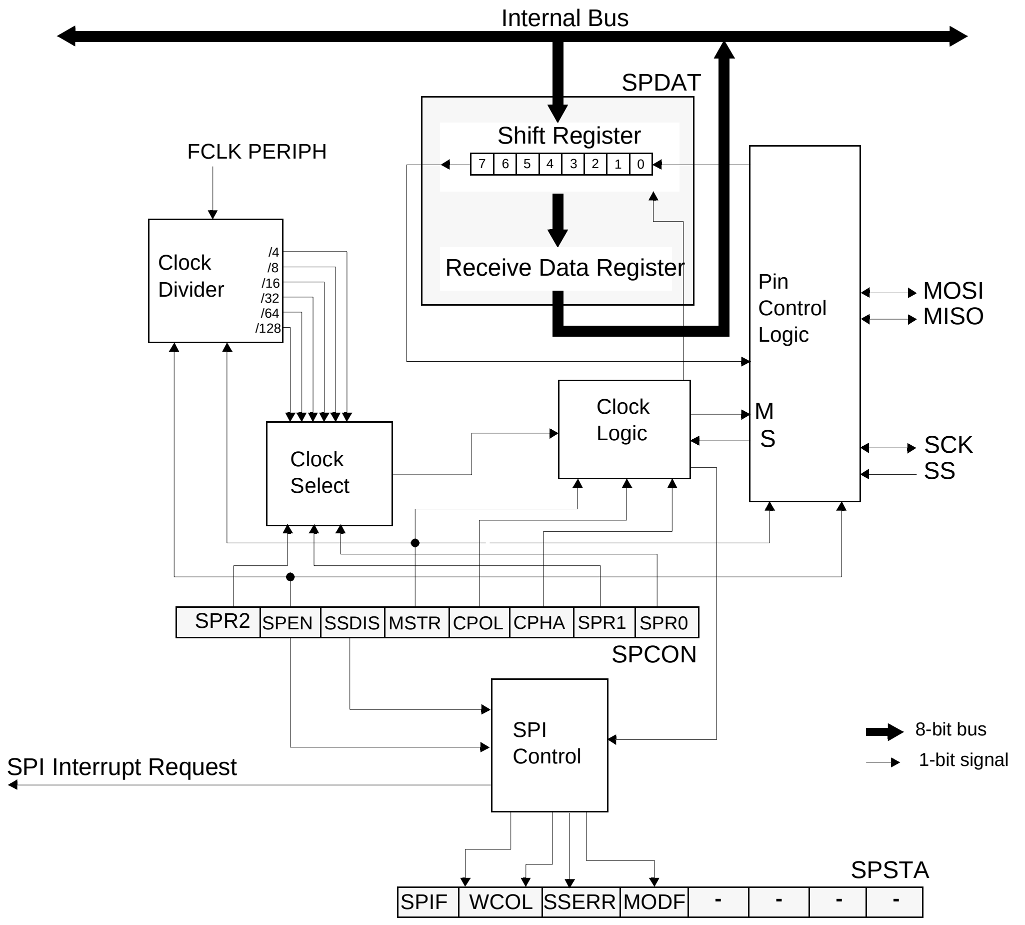
Bit Number	Bit Mnemonic	Description
7	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
6	PPCH	**PCA interrupt Priority high bit.** PPCH PPCL Priority Level 0 0 Lowest 0 1 1 0 1 1 Highest
5	PT2H	**Timer 2 overflow interrupt Priority High bit** PT2H PT2L Priority Level 0 0 Lowest 0 1 1 0 1 1 Highest
4	PSH	**Serial port Priority High bit** PSH PSL Priority Level 0 0 Lowest 0 1 1 0 1 1 Highest
3	PT1H	**Timer 1 overflow interrupt Priority High bit** PT1H PT1L Priority Level 0 0 Lowest 0 1 1 0 1 1 Highest
2	PX1H	**External interrupt 1 Priority High bit** PX1H PX1L Priority Level 0 0 Lowest 0 1 1 0 1 1 Highest
1	PT0H	**Timer 0 overflow interrupt Priority High bit** PT0H PT0L Priority Level 0 0 Lowest 0 1 1 0 1 1 Highest
0	PX0H	**External interrupt 0 Priority High bit** PX0H PX0L Priority Level 0 0 Lowest 0 1 1 0 1 1 Highest

Reset Value = X000 0000b
Not bit addressable

Functional Description

Figure 40 shows a detailed structure of the SPI module.

Figure 40. SPI Module Block Diagram



Operating Modes

The Serial Peripheral Interface can be configured as one of the two modes: Master mode or Slave mode. The configuration and initialization of the SPI module is made through one register:

- The Serial Peripheral CONtrol register (SPCON)

Once the SPI is configured, the data exchange is made using:

- SPCON
- The Serial Peripheral STAtus register (SPSTA)
- The Serial Peripheral DATa register (SPDAT)

During an SPI transmission, data is simultaneously transmitted (shifted out serially) and received (shifted in serially). A serial clock line (SCK) synchronizes shifting and sampling on the two serial data lines (MOSI and MISO). A Slave Select line (SS) allows individual selection of a Slave SPI device; Slave devices that are not selected do not interfere with SPI bus activities.

When the Master device transmits data to the Slave device via the MOSI line, the Slave device responds by sending data to the Master device via the MISO line. This implies full-duplex transmission with both data out and data in synchronized with the same clock (Figure 41).

Table 85. Status in Slave Receiver Mode

Status Code (SSCS)	Status of the 2-wire bus and 2-wire hardware	Application Software Response					Next Action Taken By 2-wire Software
		To/from SSDAT	To SSCON				
			STA	STO	SI	AA	
60h	Own SLA+W has been received; ACK has been returned	No SSDAT action or	X	0	0	0	Data byte will be received and NOT ACK will be returned
		No SSDAT action	X	0	0	1	Data byte will be received and ACK will be returned
68h	Arbitration lost in SLA+R/W as master; own SLA+W has been received; ACK has been returned	No SSDAT action or	X	0	0	0	Data byte will be received and NOT ACK will be returned
		No SSDAT action	X	0	0	1	Data byte will be received and ACK will be returned
70h	General call address has been received; ACK has been returned	No SSDAT action or	X	0	0	0	Data byte will be received and NOT ACK will be returned
		No SSDAT action	X	0	0	1	Data byte will be received and ACK will be returned
78h	Arbitration lost in SLA+R/W as master; general call address has been received; ACK has been returned	No SSDAT action or	X	0	0	0	Data byte will be received and NOT ACK will be returned
		No SSDAT action	X	0	0	1	Data byte will be received and ACK will be returned
80h	Previously addressed with own SLA+W; data has been received; ACK has been returned	No SSDAT action or	X	0	0	0	Data byte will be received and NOT ACK will be returned
		No SSDAT action	X	0	0	1	Data byte will be received and ACK will be returned
88h	Previously addressed with own SLA+W; data has been received; NOT ACK has been returned	Read data byte or	0	0	0	0	Switched to the not addressed slave mode; no recognition of own SLA or GCA
		Read data byte or	0	0	0	1	Switched to the not addressed slave mode; own SLA will be recognised; GCA will be recognised if GC=logic 1
		Read data byte or	1	0	0	0	Switched to the not addressed slave mode; no recognition of own SLA or GCA. A START condition will be transmitted when the bus becomes free
		Read data byte	1	0	0	1	Switched to the not addressed slave mode; own SLA will be recognised; GCA will be recognised if GC=logic 1. A START condition will be transmitted when the bus becomes free
90h	Previously addressed with general call; data has been received; ACK has been returned	Read data byte or	X	0	0	0	Data byte will be received and NOT ACK will be returned
		Read data byte	X	0	0	1	Data byte will be received and ACK will be returned

Table 85. Status in Slave Receiver Mode (Continued)

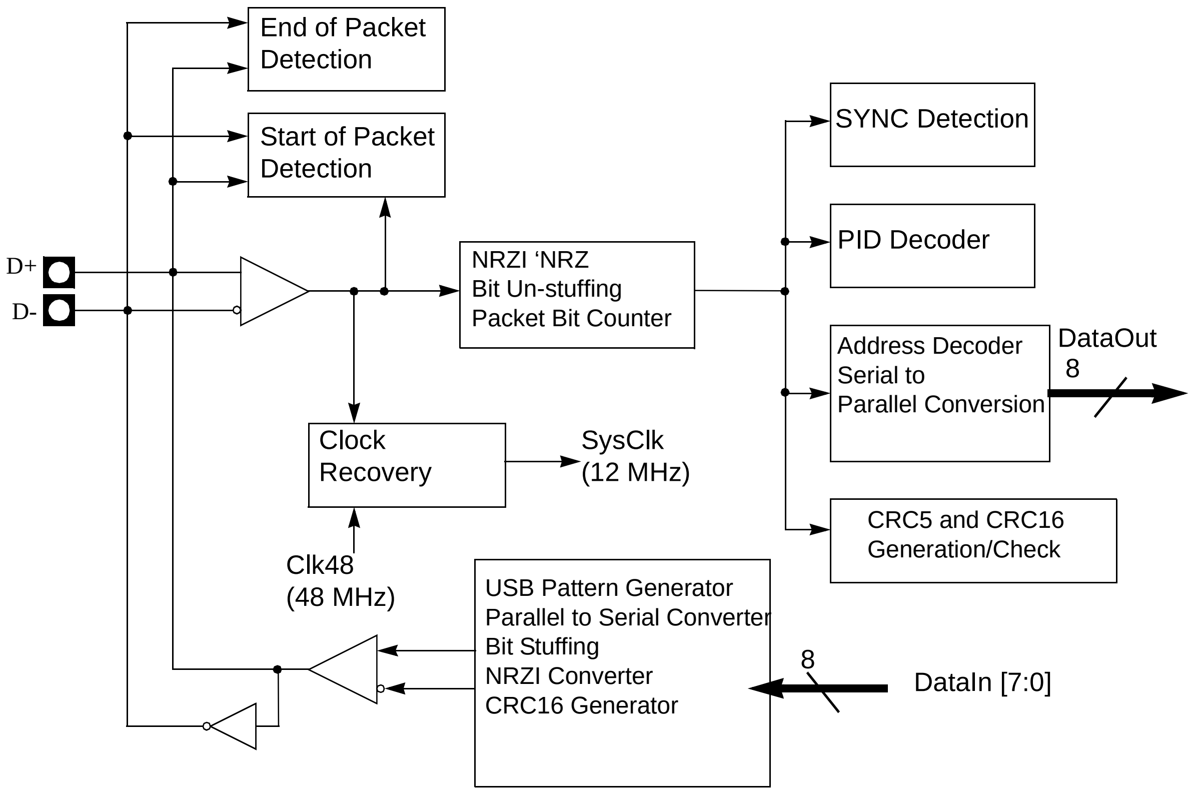
Status Code (SSCS)	Status of the 2-wire bus and 2-wire hardware	Application Software Response					Next Action Taken By 2-wire Software
		To/from SSDAT	To SSCON				
			STA	STO	SI	AA	
98h	Previously addressed with general call; data has been received; NOT ACK has been returned	Read data byte or	0	0	0	0	Switched to the not addressed slave mode; no recognition of own SLA or GCA
		Read data byte or	0	0	0	1	Switched to the not addressed slave mode; own SLA will be recognised; GCA will be recognised if GC=logic 1
		Read data byte or	1	0	0	0	Switched to the not addressed slave mode; no recognition of own SLA or GCA. A START condition will be transmitted when the bus becomes free
		Read data byte	1	0	0	1	Switched to the not addressed slave mode; own SLA will be recognised; GCA will be recognised if GC=logic 1. A START condition will be transmitted when the bus becomes free
A0h	A STOP condition or repeated START condition has been received while still addressed as slave	No SSDAT action or	0	0	0	0	Switched to the not addressed slave mode; no recognition of own SLA or GCA
		No SSDAT action or	0	0	0	1	Switched to the not addressed slave mode; own SLA will be recognised; GCA will be recognised if GC=logic 1
		No SSDAT action or	1	0	0	0	Switched to the not addressed slave mode; no recognition of own SLA or GCA. A START condition will be transmitted when the bus becomes free
		No SSDAT action	1	0	0	1	Switched to the not addressed slave mode; own SLA will be recognised; GCA will be recognised if GC=logic 1. A START condition will be transmitted when the bus becomes free

Serial Interface Engine (SIE)

The SIE performs the following functions:

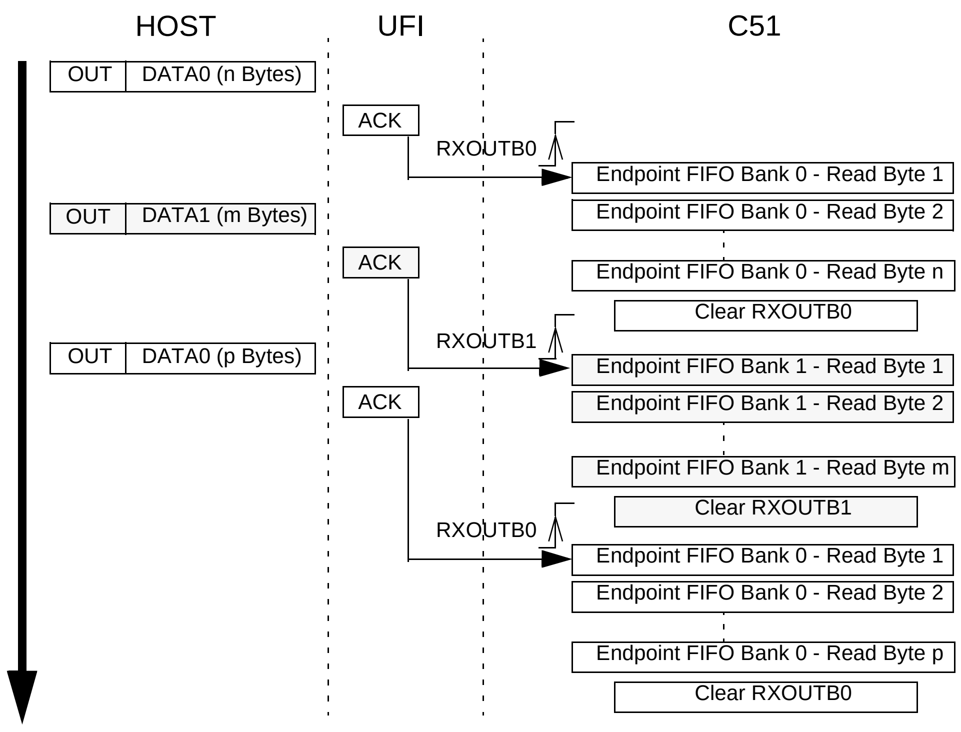
- NRZI data encoding and decoding.
- Bit stuffing and un-stuffing.
- CRC generation and checking.
- Handshakes.
- TOKEN type identifying.
- Address checking.
- Clock generation (via DPLL).

Figure 54. SIE Block Diagram



Bulk/Interrupt OUT Transactions in Ping-pong Mode

Figure 60. Bulk/Interrupt OUT Transactions in Ping-pong Mode


HOST
UFI
C51
OUT
DATA0 (n Bytes)
ACK
RXOUTB0
Endpoint FIFO Bank 0 - Read Byte 1
OUT
DATA1 (m Bytes)
Endpoint FIFO Bank 0 - Read Byte 2
ACK
Endpoint FIFO Bank 0 - Read Byte n
Clear RXOUTB0
RXOUTB1
OUT
DATA0 (p Bytes)
Endpoint FIFO Bank 1 - Read Byte 1
ACK
Endpoint FIFO Bank 1 - Read Byte 2
Endpoint FIFO Bank 1 - Read Byte m
Clear RXOUTB1
RXOUTB0
Endpoint FIFO Bank 0 - Read Byte 1
Endpoint FIFO Bank 0 - Read Byte 2
Endpoint FIFO Bank 0 - Read Byte p
Clear RXOUTB0

An endpoint will be first enabled and configured before being able to receive Bulk or Interrupt packets.

When a valid OUT packet is received on the endpoint bank 0, the RXOUTB0 bit is set by the USB controller. This triggers an interrupt if enabled. The firmware has to select the corresponding endpoint, store the number of data bytes by reading the UBYCTLX and UBYCTHX registers. If the received packet is a ZLP (Zero Length Packet), the UBYCTLX and UBYCTHX register values are equal to 0 and no data has to be read.

When all the endpoint FIFO bytes have been read, the firmware will clear the RXOUB0 bit to allow the USB controller to accept the next OUT packet on the endpoint bank 0. This action switches the endpoint bank 0 and 1. Until the RXOUTB0 bit has been cleared by the firmware, the USB controller will answer a NAK handshake for each OUT requests on the bank 0 endpoint FIFO.

When a new valid OUT packet is received on the endpoint bank 1, the RXOUTB1 bit is set by the USB controller. This triggers an interrupt if enabled. The firmware empties the bank 1 endpoint FIFO before clearing the RXOUTB1 bit. Until the RXOUTB1 bit has been cleared by the firmware, the USB controller will answer a NAK handshake for each OUT requests on the bank 1 endpoint FIFO.

The RXOUTB0 and RXOUTB1 bits are alternatively set by the USB controller at each new valid packet receipt.
The firmware has to clear one of these two bits after having read all the data FIFO to allow a new valid packet to be stored in the corresponding bank.

A NAK handshake is sent by the USB controller only if the banks 0 and 1 has not been released by the firmware.

If the Host sends more bytes than supported by the endpoint FIFO, the overflow data won't be stored, but the USB controller will consider that the packet is valid if the CRC is correct.

Isochronous Transactions

Isochronous OUT Transactions in Standard Mode

An endpoint will be first enabled and configured before being able to receive Isochronous packets.

When a OUT packet is received on an endpoint, the RXOUTB0 bit is set by the USB controller. This triggers an interrupt if enabled. The firmware has to select the corresponding endpoint, store the number of data bytes by reading the UBYCTLX and UBYCTHX registers. If the received packet is a ZLP (Zero Length Packet), the UBYCTLX and UBYCTHX register values are equal to 0 and no data has to be read.

The STLCRC bit in the UEPSTAX register is set by the USB controller if the packet stored in FIFO has a corrupted CRC. This bit is updated after each new packet receipt.

When all the endpoint FIFO bytes have been read, the firmware will clear the RXOUTB0 bit to allow the USB controller to store the next OUT packet data into the endpoint FIFO. Until the RXOUTB0 bit has been cleared by the firmware, the data sent by the Host at each OUT transaction will be lost.

If the RXOUTB0 bit is cleared while the Host is sending data, the USB controller will store only the remaining bytes into the FIFO.

If the Host sends more bytes than supported by the endpoint FIFO, the overflow data won’t be stored, but the USB controller will consider that the packet is valid if the CRC is correct.

Isochronous OUT Transactions in Ping-pong Mode

An endpoint will be first enabled and configured before being able to receive Isochronous packets.

When a OUT packet is received on the endpoint bank 0, the RXOUTB0 bit is set by the USB controller. This triggers an interrupt if enabled. The firmware has to select the corresponding endpoint, store the number of data bytes by reading the UBYCTLX and UBYCTHX registers. If the received packet is a ZLP (Zero Length Packet), the UBYCTLX and UBYCTHX register values are equal to 0 and no data has to be read.

The STLCRC bit in the UEPSTAX register is set by the USB controller if the packet stored in FIFO has a corrupted CRC. This bit is updated after each new packet receipt.

When all the endpoint FIFO bytes have been read, the firmware will clear the RXOUB0 bit to allow the USB controller to store the next OUT packet data into the endpoint FIFO bank 0. This action switches the endpoint bank 0 and 1. Until the RXOUTB0 bit has been cleared by the firmware, the data sent by the Host on the bank 0 endpoint FIFO will be lost.

If the RXOUTB0 bit is cleared while the Host is sending data on the endpoint bank 0, the USB controller will store only the remaining bytes into the FIFO.

When a new OUT packet is received on the endpoint bank 1, the RXOUTB1 bit is set by the USB controller. This triggers an interrupt if enabled. The firmware empties the bank 1 endpoint FIFO before clearing the RXOUTB1 bit. Until the RXOUTB1 bit has been cleared by the firmware, the data sent by the Host on the bank 1 endpoint FIFO will be lost.

The RXOUTB0 and RXOUTB1 bits are alternatively set by the USB controller at each new packet receipt.

The firmware has to clear one of these two bits after having read all the data FIFO to allow a new packet to be stored in the corresponding bank.

USB Registers

Table 94. USBCON Register
USBCON (S:BCh)
USB Global Control Register

7	6	5	4	3	2	1	0
USBE	SUSPCLK	SDRMWUP	DETACH	UPRSM	RMWUPE	CONFG	FADDEN

Bit Number	Bit Mnemonic	Description
7	USBE	**USB Enable** Set this bit to enable the USB controller. Clear this bit to disable and reset the USB controller, to disable the USB transceiver an to disable the USB controller clock inputs.
6	SUSPCLK	**Suspend USB Clock** Set this bit to disable the 48 MHz clock input (Resume Detection is still active). Clear this bit to enable the 48 MHz clock input.
5	SDRMWUP	**Send Remote Wake Up** Set this bit to force an external interrupt on the USB controller for Remote Wake UP purpose. An upstream resume is send only if the bit RMWUPE is set, all USB clocks are enabled AND the USB bus was in SUSPEND state for at least 5 ms. See UPRSM below. This bit is cleared by software.
4	DETACH	**Detach Command** Set this bit to simulate a Detach on the USB line. The V_{REF} pin is then in a floating state. Clear this bit to maintain V_{REF} at high level.
3	UPRSM	**Upstream Resume (read only)** This bit is set by hardware when SDRMWUP has been set and if RMWUPE is enabled. This bit is cleared by hardware after the upstream resume has been sent.
2	RMWUPE	**Remote Wake-Up Enable** Set this bit to enabled request an upstream resume signaling to the host. Clear this bit otherwise. Note: Do not set this bit if the host has not set the DEVICE_REMOTE_WAKEUP feature for the device.
1	CONFG	**Configured** This bit will be set by the device firmware after a SET_CONFIGURATION request with a non-zero value has been correctly processed. It will be cleared by the device firmware when a SET_CONFIGURATION request with a zero value is received. It is cleared by hardware on hardware reset or when an USB reset is detected on the bus (SE0 state for at least 32 Full Speed bit times: typically 2.7 µs).
0	FADDEN	**Function Address Enable** This bit will be set by the device firmware after a successful status phase of a SET_ADDRESS transaction. It will not be cleared afterwards by the device firmware. It is cleared by hardware on hardware reset or when an USB reset is received (see above). When this bit is cleared, the default function address is used (0).

Reset Value = 00h

Table 96. USBIEN Register
USBIEN (S:BEh)
USB Global Interrupt Enable Register

7	6	5	4	3	2	1	0
-	-	EWUPCPU	EEORINT	ESOFINT	-	-	ESPINT

Bit Number	Bit Mnemonic	Description
7-6	-	**Reserved** The value read from these bits is always 0. Do not set these bits.
5	EWUPCPU	**Enable Wake Up CPU Interrupt** Set this bit to enable Wake Up CPU Interrupt. (See “USBIEN Register USBIEN (S:BEh) USB Global Interrupt Enable Register” on page 139.) Clear this bit to disable Wake Up CPU Interrupt.
4	EEORINT	**Enable End Of Reset Interrupt** Set this bit to enable End Of Reset Interrupt. (See “USBIEN Register USBIEN (S:BEh) USB Global Interrupt Enable Register” on page 139.). This bit is set after reset. Clear this bit to disable End Of Reset Interrupt.
3	ESOFINT	**Enable SOF Interrupt** Set this bit to enable SOF Interrupt. (See “USBIEN Register USBIEN (S:BEh) USB Global Interrupt Enable Register” on page 139.). Clear this bit to disable SOF Interrupt.
2	-	**Reserved** The value read from these bits is always 0. Do not set these bits.
1	-	
0	ESPINT	**Enable Suspend Interrupt** Set this bit to enable Suspend Interrupts (see the “USBIEN Register USBIEN (S:BEh) USB Global Interrupt Enable Register” on page 139). Clear this bit to disable Suspend Interrupts.

Reset Value = 10h

Table 97. USBADDR Register
USBADDR (S:C6h)
USB Address Register

7	6	5	4	3	2	1	0
FEN	UADD6	UADD5	UADD4	UADD3	UADD2	UADD1	UADD0

Bit Number	Bit Mnemonic	Description
7	FEN	**Function Enable** Set this bit to enable the address filtering function. Cleared this bit to disable the function.
6-0	UADD[6:0]	**USB Address** This field contains the default address (0) after power-up or USB bus reset. It will be written with the value set by a SET_ADDRESS request received by the device firmware.

Reset Value = 80h

Table 98. UEPNUM Register
UEPNUM (S:C7h)
USB Endpoint Number

7	6	5	4	3	2	1	0
-	-	-	-	EPNUM3	EPNUM2	EPNUM1	EPNUM0

Bit Number	Bit Mnemonic	Description
7-4	-	**Reserved** The value read from these bits is always 0. Do not set these bits.
3-0	EPNUM[3:0]	**Endpoint Number** Set this field with the number of the endpoint which will be accessed when reading or writing to, UEPDATX Register UEPDATX (S:CFh) USB FIFO Data Endpoint X (X = EPNUM set in UEPNUM Register UEPNUM (S:C7h) USB Endpoint Number), UBYCTLX Register UBYCTLX (S:E2h) USB Byte Count Low Register X (X = EPNUM set in UEPNUM Register UEPNUM (S:C7h) USB Endpoint Number), UBYCTHX Register UBYCTHX (S:E3h) USB Byte Count High Register X (X = EPNUM set in UEPNUM Register UEPNUM (S:C7h) USB Endpoint Number) or UEPCONX Register UEPCONX (S:D4h) USB Endpoint X Control Register. This value can be 0, 1, 2, 3, 4, 5 or 6.

Reset Value = 00h

Table 104. UEPRST Register
UEPRST (S:D5h)
USB Endpoint FIFO Reset Register

7	6	5	4	3	2	1	0
-	EP6RST	EP5RST	EP4RST	EP3RST	EP2RST	EP1RST	EP0RST

Bit Number	Bit Mnemonic	Description
7	-	**Reserved** The value read from this bit is always 0. Do not set this bit.
6	EP6RST	**Endpoint 6 FIFO Reset** Set this bit and reset the endpoint FIFO prior to any other operation, upon hardware reset or when an USB bus reset has been received. Then, clear this bit to complete the reset operation and start using the FIFO.
5	EP5RST	**Endpoint 5 FIFO Reset** Set this bit and reset the endpoint FIFO prior to any other operation, upon hardware reset or when an USB bus reset has been received. Then, clear this bit to complete the reset operation and start using the FIFO.
4	EP4RST	**Endpoint 4 FIFO Reset** Set this bit and reset the endpoint FIFO prior to any other operation, upon hardware reset or when an USB bus reset has been received. Then, clear this bit to complete the reset operation and start using the FIFO.
3	EP3RST	**Endpoint 3 FIFO Reset** Set this bit and reset the endpoint FIFO prior to any other operation, upon hardware reset or when an USB bus reset has been received. Then, clear this bit to complete the reset operation and start using the FIFO.
2	EP2RST	**Endpoint 2 FIFO Reset** Set this bit and reset the endpoint FIFO prior to any other operation, upon hardware reset or when an USB bus reset has been received. Then, clear this bit to complete the reset operation and start using the FIFO.
1	EP1RST	**Endpoint 1 FIFO Reset** Set this bit and reset the endpoint FIFO prior to any other operation, upon hardware reset or when an USB bus reset has been received. Then, clear this bit to complete the reset operation and start using the FIFO.
0	EP0RST	**Endpoint 0 FIFO Reset** Set this bit and reset the endpoint FIFO prior to any other operation, upon hardware reset or when an USB bus reset has been received. Then, clear this bit to complete the reset operation and start using the FIFO.

Reset Value = 00h

Table 107. UFNUMH Register
UFNUMH (S:BBh, read-only)
USB Frame Number High Register

7	6	5	4	3	2	1	0
-	-	CRCOK	CRCERR	-	FNUM10	FNUM9	FNUM8

Bit Number	Bit Mnemonic	Description
5	CRCOK	**Frame Number CRC OK** This bit is set by hardware when a new Frame Number in Start of Frame Packet is received without CRC error. This bit is updated after every Start of Frame packet receipt. Important note: the Start of Frame interrupt is generated just after the PID receipt.
4	CRCERR	**Frame Number CRC Error** This bit is set by hardware when a corrupted Frame Number in Start of Frame packet is received. This bit is updated after every Start of Frame packet receipt. Important note: the Start of Frame interrupt is generated just after the PID receipt.
3	-	**Reserved** The value read from this bit is always 0. Do not set this bit.
2-0	FNUM[10:8]	**Frame Number** FNUM[10:8] are the upper 3 bits of the 11-bit Frame Number (see the “UFNUML Register UFNUML (S:BAh, read-only) USB Frame Number Low Register” on page 148). It is provided in the last received SOF packet (see SOFINT in the “USBIEN Register USBIEN (S:BEh) USB Global Interrupt Enable Register” on page 139). FNUM is updated if a corrupted SOF is received.

Reset Value = 00h

Table 108. UFNUML Register
UFNUML (S:BAh, read-only)
USB Frame Number Low Register

7	6	5	4	3	2	1	0
FNUM7	FNUM6	FNUM5	FNUM4	FNUM3	FNUM2	FNUM1	FNUM0

Bit Number	Bit Mnemonic	Description
7 - 0	FNUM[7:0]	**Frame Number** FNUM[7:0] are the lower 8 bits of the 11-bit Frame Number (See “UFNUMH Register UFNUMH (S:BBh, read-only) USB Frame Number High Register” on page 148.).

Reset Value = 00h

ONCE Mode (ON Chip Emulation)

The ONCE mode facilitates testing and debugging of systems using AT89C5130A/31A-M without removing the circuit from the board. The ONCE mode is invoked by driving certain pins of the AT89C5130A/31A-M; the following sequence must be exercised:

- Pull ALE low while the device is in reset (RST high) and $\overline{\text{PSEN}}$ is high.
- Hold ALE low as RST is deactivated.

While the AT89C5130A/31A-M is in ONCE mode, an emulator or test CPU can be used to drive the circuit Table 113 shows the status of the port pins during ONCE mode.

Normal operation is restored when normal reset is applied.

Table 113. External Pin Status during ONCE Mode

ALE	PSEN	Port 0	Port 1	Port 2	Port 3	Port I2	XTAL1/2
Weak pull-up	Weak pull-up	Float	Weak pull-up	Weak pull-up	Weak pull-up	Float	Active