Features

- 80C52X2 Core (6 Clocks per Instruction)
  - Maximum Core Frequency 48 MHz in X1 Mode, 24MHz in X2 Mode
  - Dual Data Pointer
  - Full-duplex Enhanced UART (EUART)
  - Three 16-bit Timer/Counters: T0, T1 and T2
  - 256 Bytes of Scratchpad RAM
- 16/32-Kbyte On-chip Flash In-System Programming through USB
  - Byte and Page (128 bytes) Erase and Write
  - 100k Write Cycles
- 4-Kbyte EEPROM for Bootloader (3-Kbyte) and Data (1-Kbyte)
  - Byte and Page (128 bytes) Erase and Write
  - 100k Write Cycles
- On-chip Expanded RAM (ERAM): 1024 Bytes
- Integrated Power Monitor (POR/PFD) to Supervise Internal Power Supply
- USB 1.1 and 2.0 Full Speed Compliant Module with Interrupt on Transfer Completion
  - Endpoint 0 for Control Transfers: 32-byte FIFO
  - 6 Programmable Endpoints with In or Out Directions and with Bulk, Interrupt or Isochronous Transfers
    - Endpoint 1, 2, 3: 32-byte FIFO
    - Endpoint 4, 5: 2 x 64-byte FIFO with Double Buffering (Ping-pong Mode)
    - Endpoint 6: 2 x 512-byte FIFO with Double Buffering (Ping-pong Mode)
  - Suspend/Resume Interrupts
  - 48 MHz PLL for Full-speed Bus Operation
  - Bus Disconnection on Microcontroller Request
- 5 Channels Programmable Counter Array (PCA) with 16-bit Counter, High-speed Output, Compare/Capture, PWM and Watchdog Timer Capabilities
- Programmable Hardware Watchdog Timer (One-time Enabled with Reset-out): 100 ms to 3s at 8 MHz
- Keyboard Interrupt Interface on Port P1 (8 Bits)
- TWI (Two Wire Interface) 400Kbit/s
- SPI Interface (Master/Slave Mode)
- 34 I/O Pins
- 4 Direct-drive LED Outputs with Programmable Current Sources: 2-6-10 mA Typical
- 4-level Priority Interrupt System (11 sources)
- Idle and Power-down Modes
- 0 to 24 MHz On-chip Oscillator with Analog PLL for 48 MHz Synthesis
- Industrial Temperature Range
- Extended Range Power Supply: 2.7V to 5.5V (3.3V to 5.5V required for USB)
- Packages: PLCC52, VQFP64, QFN32

8-bit Flash Microcontroller with Full Speed USB Device

AT89C5130A-M
AT89C5131A-M

Rev. 4337C–USB–02/05

Description

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

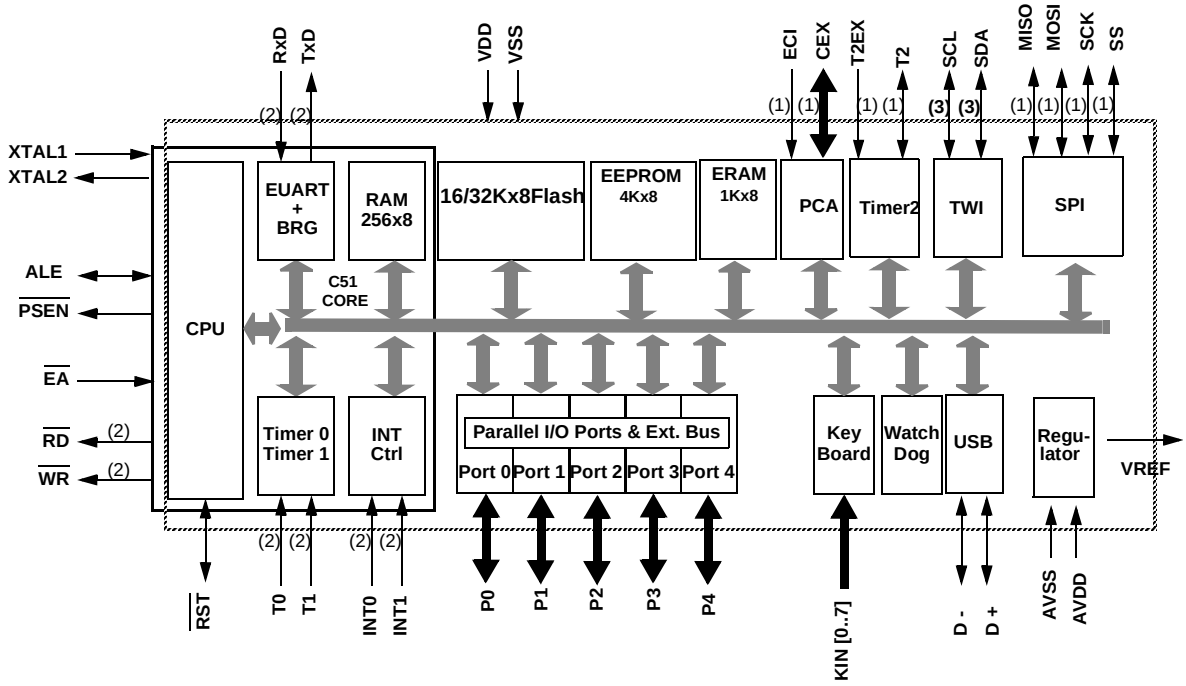
AT89C5130A/31A-M features a full-speed USB module compatible with the USB specifications Version 1.1 and 2.0. This module integrates the USB transceivers with a 3.3V voltage regulator and the Serial Interface Engine (SIE) with Digital Phase Locked Loop and 48 MHz clock recovery. USB Event detection logic (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.

Block Diagram



Notes:
1. Alternate function of Port 1
2. Alternate function of Port 3
3. Alternate function of Port 4

Pinout Description

Pinout

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

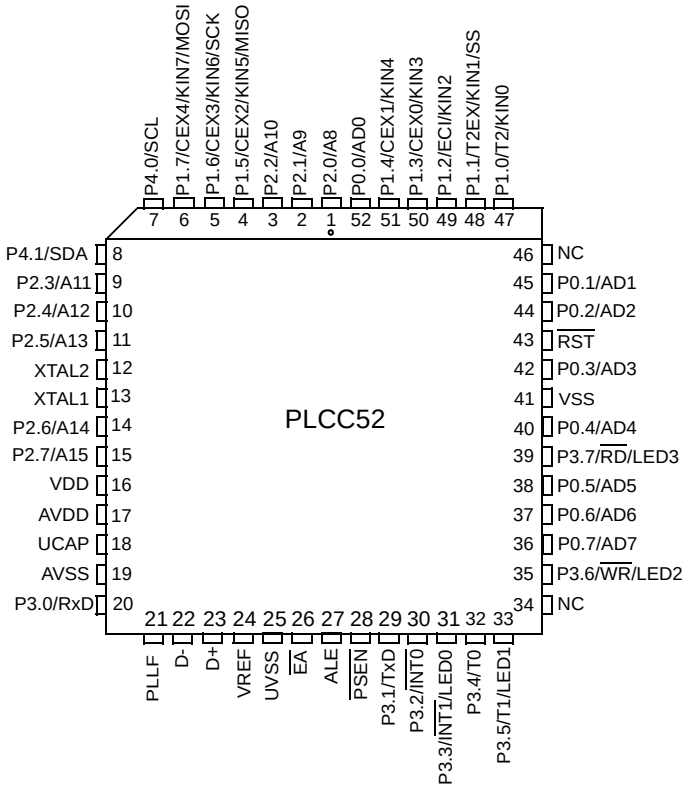


Figure 2. AT89C5130A/31A-M 64-pin VQFP Pinout

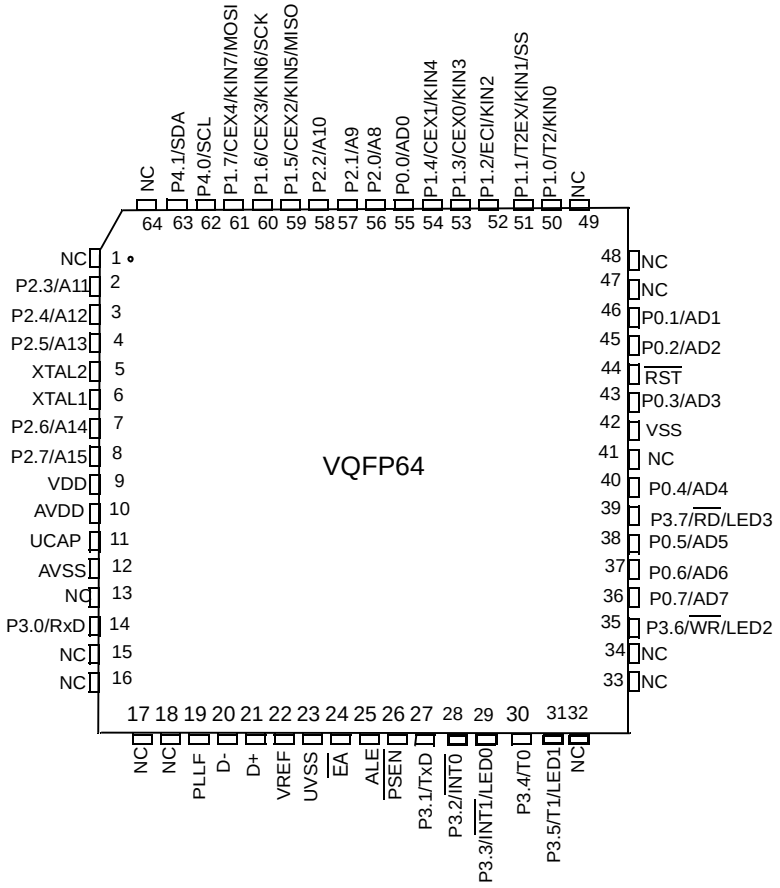
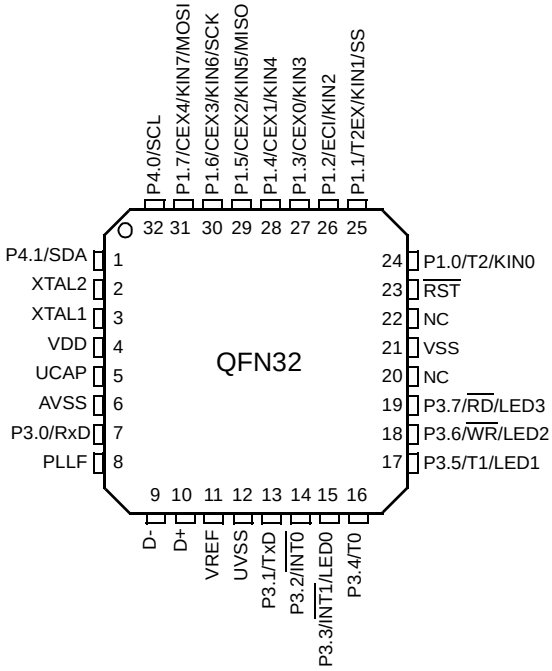


Figure 3. AT89C5130A/31A-M 32-pin QFN Pinout



Note : The metal plate can be connected to Vss

Signals

All the AT89C5130A/31A-M signals are detailed by functionality on Table 1 through Table 12.

Table 1. Keypad Interface Signal Description

Signal Name	Type	Description	Alternate Function
KIN[7:0]	I	**Keypad Input Lines** Holding one of these pins high or low for 24 oscillator periods triggers a keypad interrupt if enabled. Held line is reported in the KBCON register.	P1[7:0]

Table 2. Programmable Counter Array Signal Description

Signal Name	Type	Description	Alternate Function
ECI	I	**External Clock Input**	P1.2
CEX[4:0]	I/O	Capture External Input Compare External Output	P1.3 P1.4 P1.5 P1.6 P1.7

Table 3. Serial I/O Signal Description

Signal Name	Type	Description	Alternate Function
RxD	I	**Serial Input** The serial input is P3.0 after reset, but it can also be configured to P4.0 by software.	P3.0
TxD	O	**Serial Output** The serial output is P3.1 after reset, but it can also be configured to P4.1 by software.	P3.1

Table 4. Timer 0, Timer 1 and Timer 2 Signal Description

Signal Name	Type	Description	Alternate Function
INT0	I	**Timer 0 Gate Input** $\overline{INT0}$ serves as external run control for timer 0, when selected by GATE0 bit in TCON register. **External Interrupt 0** $\overline{INT0}$ input set IE0 in the TCON register. If bit IT0 in this register is set, bits IE0 are set by a falling edge on $\overline{INT0}$. If bit IT0 is cleared, bits IE0 is set by a low level on $\overline{INT0}$.	P3.2
INT1	I	**Timer 1 Gate Input** $\overline{INT1}$ serves as external run control for Timer 1, when selected by GATE1 bit in TCON register. **External Interrupt 1** $\overline{INT1}$ input set IE1 in the TCON register. If bit IT1 in this register is set, bits IE1 are set by a falling edge on $\overline{INT1}$. If bit IT1 is cleared, bits IE1 is set by a low level on $\overline{INT1}$.	P3.3

Table 4. Timer 0, Timer 1 and Timer 2 Signal Description (Continued)

Signal Name	Type	Description	Alternate Function
T0	I	**Timer Counter 0 External Clock Input** When Timer 0 operates as a counter, a falling edge on the T0 pin increments the count.	P3.4
T1	I	**Timer/Counter 1 External Clock Input** When Timer 1 operates as a counter, a falling edge on the T1 pin increments the count.	P3.5
T2	I O	**Timer/Counter 2 External Clock Input** Timer/Counter 2 Clock Output	P1.0
T2EX	I	Timer/Counter 2 Reload/Capture/Direction Control Input	P1.1

Table 5. LED Signal Description

Signal Name	Type	Description	Alternate Function
LED[3:0]	O	**Direct Drive LED Output** These pins can be directly connected to the Cathode of standard LEDs without external current limiting resistors. The typical current of each output can be programmed by software to 2, 6 or 10 mA. Several outputs can be connected together to get higher drive capabilities.	P3.3 P3.5 P3.6 P3.7

Table 6. TWI Signal Description

Signal Name	Type	Description	Alternate Function
SCL	I/O	**SCL: TWI Serial Clock** SCL output the serial clock to slave peripherals. SCL input the serial clock from master.	P4.0
SDA	I/O	**SDA: TWI Serial Data** SCL is the bidirectional TWI data line.	P4.1

Table 7. SPI Signal Description

Signal Name	Type	Description	Alternate Function
SS	I/O	**$\overline{\text{SS}}$:** SPI Slave Select	P1.1
MISO	I/O	**MISO:** SPI Master Input Slave Output line When SPI is in master mode, MISO receives data from the slave peripheral. When SPI is in slave mode, MISO outputs data to the master controller.	P1.5
SCK	I/O	**SCK:** SPI Serial Clock SCK outputs clock to the slave peripheral or receive clock from the master	P1.6
MOSI	I/O	**MOSI:** SPI Master Output Slave Input line When SPI is in master mode, MOSI outputs data to the slave peripheral. When SPI is in slave mode, MOSI receives data from the master controller	P1.7

Table 8. Ports Signal Description

Signal Name	Type	Description	Alternate Function
P0[7:0]	I/O	**Port 0** P0 is an 8-bit open-drain bidirectional I/O port. Port 0 pins that have 1s written to them float and can be used as high impedance inputs. To avoid any parasitic current consumption, Floating P0 inputs must be pulled to V_{DD} or V_{SS}.	AD[7:0]
P1[7:0]	I/O	**Port 1** P1 is an 8-bit bidirectional I/O port with internal pull-ups.	KIN[7:0] T2 T2EX ECI CEX[4:0]
P2[7:0]	I/O	**Port 2** P2 is an 8-bit bidirectional I/O port with internal pull-ups.	A[15:8]
P3[7:0]	I/O	**Port 3** P3 is an 8-bit bidirectional I/O port with internal pull-ups.	LED[3:0] RxD TxD $\overline{INT0}$ INT1 T0 T1 $\overline{WR}$ $\overline{RD}$
P4[1:0]	I/O	**Port 4** P4 is an 2-bit open drain port.	SCL SDA

Table 9. Clock Signal Description

Signal Name	Type	Description	Alternate Function
XTAL1	I	**Input to the on-chip inverting oscillator amplifier** To use the internal oscillator, a crystal/resonator circuit is connected to this pin. If an external oscillator is used, its output is connected to this pin.	-
XTAL2	O	**Output of the on-chip inverting oscillator amplifier** To use the internal oscillator, a crystal/resonator circuit is connected to this pin. If an external oscillator is used, leave XTAL2 unconnected.	-
PLLF	I	**PLL Low Pass Filter input** Receive the RC network of the PLL low pass filter.	-

Table 10. USB Signal Description

Signal Name	Type	Description	Alternate Function
D+	I/O	USB Data + signal	-
D-	I/O	USB Data - signal	-
VREF	O	**USB Reference Voltage** Connect this pin to D+ using a 1.5 kΩ resistor to use the Detach function.	-

Table 11. System Signal Description

Signal Name	Type	Description	Alternate Function
AD[7:0]	I/O	Multiplexed Address/Data LSB for external access Data LSB for Slave port access (used for 8-bit and 16-bit modes)	P0[7:0]
A[15:8]	I/O	Address Bus MSB for external access Data MSB for Slave port access (used for 16-bit mode only)	P2[7:0]
RD	I/O	**Read Signal** Read signal asserted during external data memory read operation. Control input for slave port read access cycles.	P3.7
WR	I/O	**Write Signal** Write signal asserted during external data memory write operation. Control input for slave write access cycles.	P3.6
$\overline{RST}$	O	**Reset Input** Holding this pin low for 64 oscillator periods while the oscillator is running resets the device. The Port pins are driven to their reset conditions when a voltage lower than V_{IL} is applied, whether or not the oscillator is running. This pin has an internal pull-up resistor which allows the device to be reset by connecting a capacitor between this pin and VSS. Asserting $\overline{RST}$ when the chip is in Idle mode or Power-down mode returns the chip to normal operation. This pin is tied to 0 for at least 12 oscillator periods when an internal reset occurs (hardware watchdog or power monitor).	-
ALE	O	**Address Latch Enable Output** The falling edge of ALE strobes the address into external latch. This signal is active only when reading or writing external memory using MOVX instructions.	-
PSEN	O	**Program** Test mode entry signal. This pin must be set to V_{DD} for normal operation.	-
EA	I	**External Access Enable** This pin must be held low to force the device to fetch code from external program memory starting at address 0000h. It is latched during reset and cannot be dynamically changed during operation.	-

Table 12. Power Signal Description

Signal Name	Type	Description	Alternate Function
AVSS	GND	**Analog Ground** AVSS is used to supply the on-chip PLL and the USB PAD.	-
AVDD	PWR	**Analog Supply Voltage** AVDD is used to supply the on-chip PLL and the USB PAD.	-
VSS	GND	**Digital Ground** VSS is used to supply the buffer ring and the digital core.	-
UVSS	GND	**USB Digital Ground** UVSS is used to supply the USB pads.	-
UCAP	PWR	**USB Pad Power Capacitor** UCAP must be connect to an external capacitor for USB pad power supply (for typical application see Figure 4 on page 12)	-

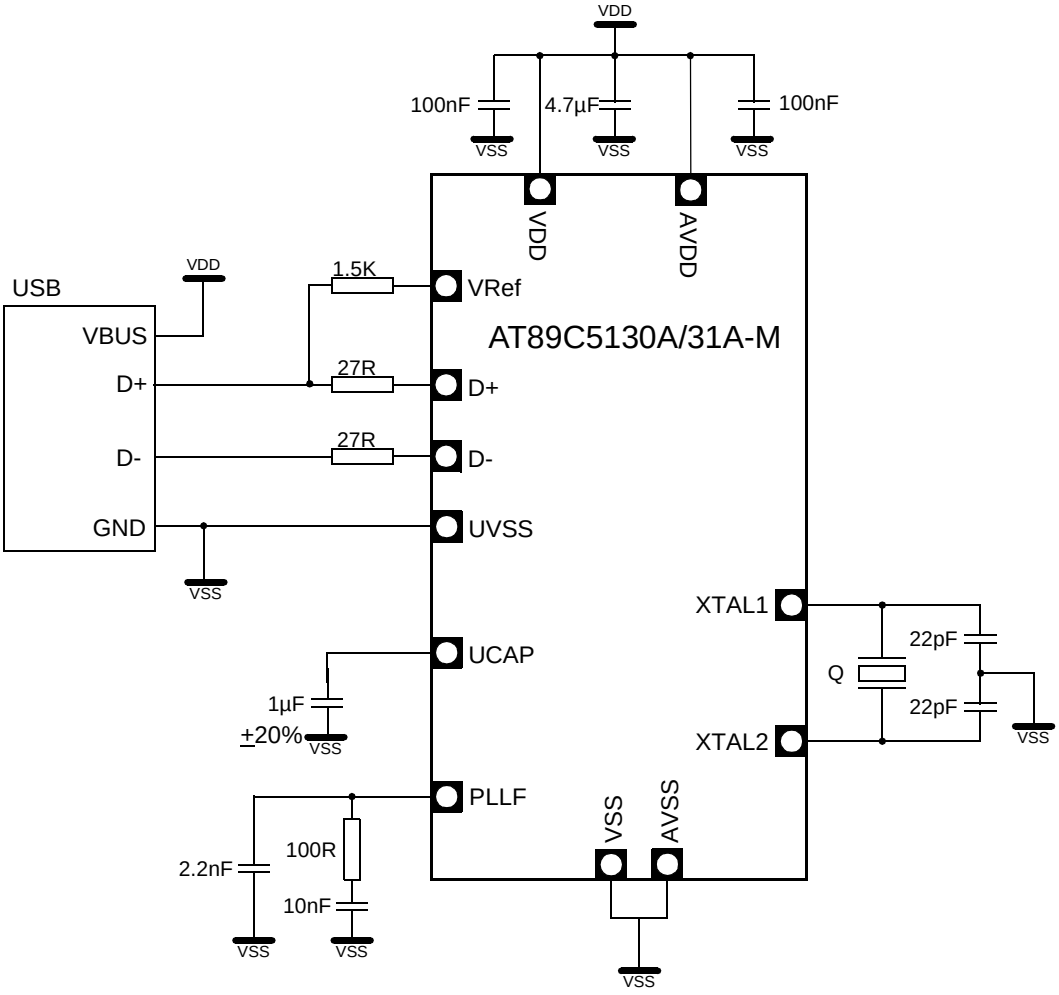
Table 12. Power Signal Description (Continued)

Signal Name	Type	Description	Alternate Function
VDD	PWR	**Digital Supply Voltage** VDD is used to supply the buffer ring on all versions of the device. It is also used to power the on-chip voltage regulator of the Standard versions or the digital core of the Low Power versions.	-
VREF	O	**USB pull-up Controlled Output** VREF is used to control the USB D+ 1.5 kΩ pull up. The Vref output is in high impedance when the bit DETACH is set in the USBCON register.	-

Typical Application

The following figure represents the typical wiring schematic.

Figure 4. Typical Application



Clock Controller

Introduction

The AT89C5130A/31A-M clock controller is based on an on-chip oscillator feeding an on-chip Phase Lock Loop (PLL). All the internal clocks to the peripherals and CPU core are generated by this controller.

The AT89C5130A/31A-M X1 and X2 pins are the input and the output of a single-stage on-chip inverter (see Figure 5) that can be configured with off-chip components as a Pierce oscillator (see Figure 6). Value of capacitors and crystal characteristics are detailed in the section “DC Characteristics”.

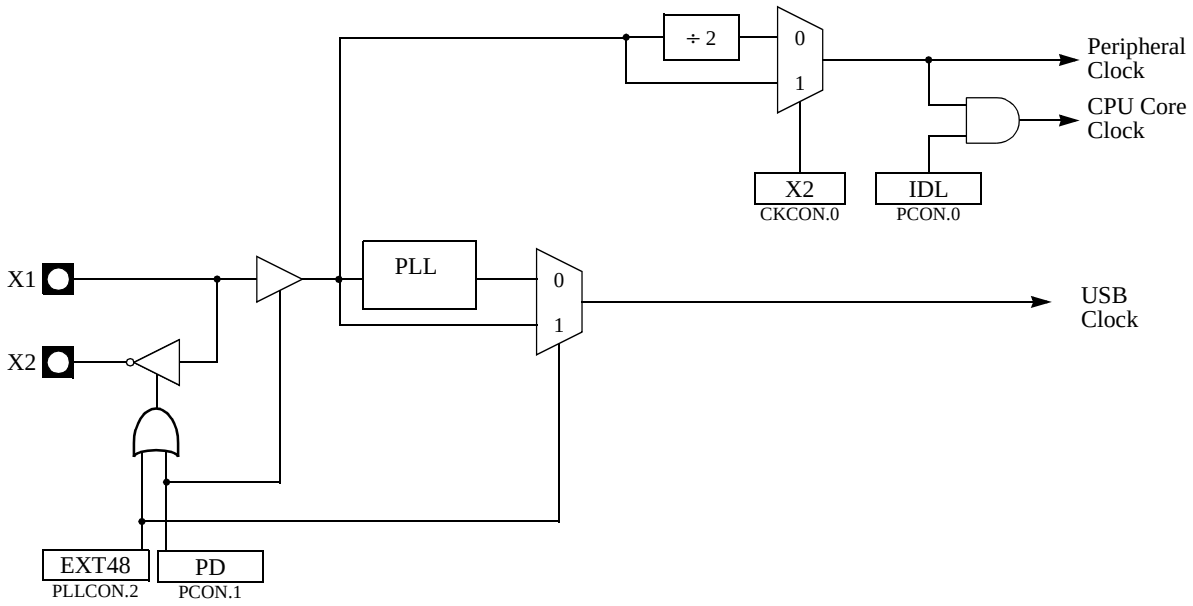
The X1 pin can also be used as input for an external 48 MHz clock.

The clock controller outputs three different clocks as shown in Figure 5:

- a clock for the CPU core
- a clock for the peripherals which is used to generate the Timers, PCA, WD, and Port sampling clocks
- a clock for the USB controller

These clocks are enabled or disabled depending on the power reduction mode as detailed in Section “Power Management”, page 153.

Figure 5. Oscillator Block Diagram



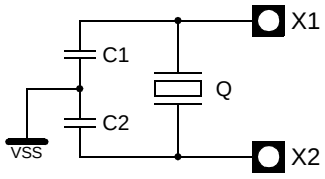
Oscillator

Two clock sources are available for CPU:

- Crystal oscillator on X1 and X2 pins: Up to 32 MHz
- External 48 MHz clock on X1 pin

In order to optimize the power consumption, the oscillator inverter is inactive when the PLL output is not selected for the USB device.

Figure 6. Crystal Connection



PLL

PLL Description

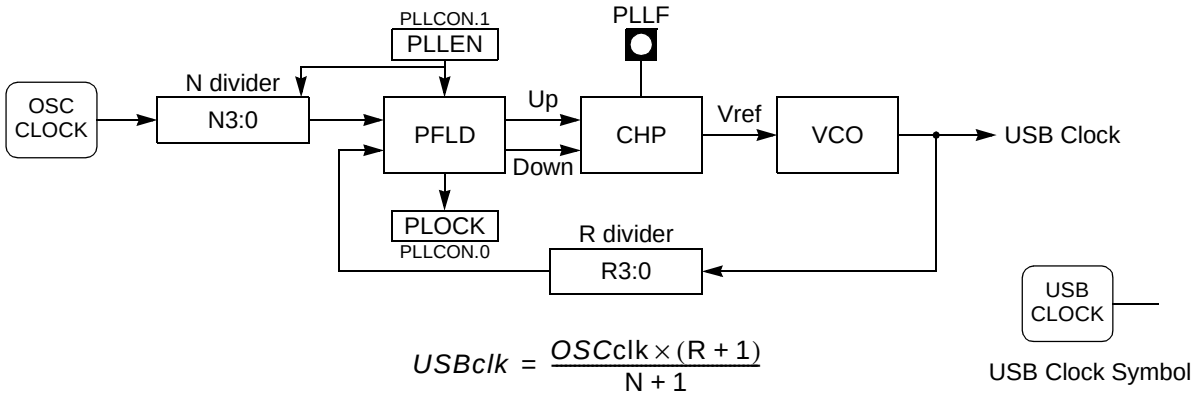
The AT89C5130A/31A-M PLL is used to generate internal high frequency clock (the USB Clock) synchronized with an external low-frequency (the Peripheral Clock). The PLL clock is used to generate the USB interface clock. Figure 7 shows the internal structure of the PLL.

The PFLD block is the Phase Frequency Comparator and Lock Detector. This block makes the comparison between the reference clock coming from the N divider and the reverse clock coming from the R divider and generates some pulses on the Up or Down signal depending on the edge position of the reverse clock. The PLLEN bit in PLLCON register is used to enable the clock generation. When the PLL is locked, the bit PLOCK in PLLCON register (see Figure 7) is set.

The CHP block is the Charge Pump that generates the voltage reference for the VCO by injecting or extracting charges from the external filter connected on PLLF pin (see Figure 8). Value of the filter components are detailed in the Section “DC Characteristics”.

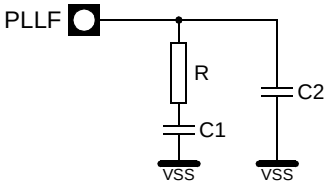
The VCO block is the Voltage Controlled Oscillator controlled by the voltage V_{REF} produced by the charge pump. It generates a square wave signal: the PLL clock.

Figure 7. PLL Block Diagram and Symbol



$$USBclk = \frac{OSCclk \times (R + 1)}{N + 1}$$

Figure 8. PLL Filter Connection

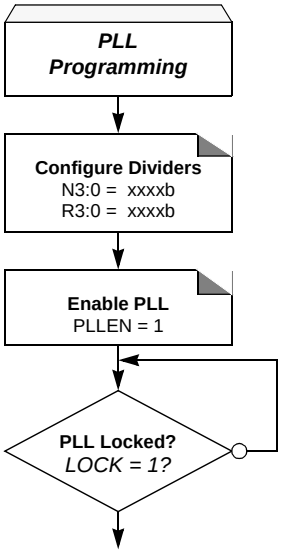


The typical values are: R = 100 Ω, C1 = 10 nf, C2 = 2.2 nF.

PLL Programming

The PLL is programmed using the flow shown in Figure 9. As soon as clock generation is enabled user must wait until the lock indicator is set to ensure the clock output is stable.

Figure 9. PLL Programming Flow



Divider Values

To generate a 48 MHz clock using the PLL, the divider values have to be configured following the oscillator frequency. The typical divider values are shown in Table 13.

Table 13. Typical Divider Values

Oscillator Frequency	R+1	N+1	PLLDIV
3 MHz	16	1	F0h
6 MHz	8	1	70h
8 MHz	6	1	50h
12 MHz	4	1	30h
16 MHz	3	1	20h
18 MHz	8	3	72h
20 MHz	12	5	B4h
24 MHz	2	1	10h
32 MHz	3	2	21h
40 MHz	12	10	B9h

Registers

Table 14. CKCON0 (S:8Fh)
Clock Control Register 0

7	6	5	4	3	2	1	0
-	WDX2	PCAX2	SIX2	T2X2	T1X2	T0X2	X2

Bit Number	Bit Mnemonic	Description
7	-	**Reserved** The value read from this bit is always 0. Do not set this bit.
6	WDX2	**Watchdog Clock** **This control bit is validated when the CPU clock X2 is set. When X2 is low, this bit has no effect.** Clear to select 6 clock periods per peripheral clock cycle. Set to select 12 clock periods per peripheral clock cycle.
5	PCAX2	**Programmable Counter Array Clock** **This control bit is validated when the CPU clock X2 is set. When X2 is low, this bit has no effect.** Clear to select 6 clock periods per peripheral clock cycle. Set to select 12 clock periods per peripheral clock cycle.
4	SIX2	**Enhanced UART Clock (Mode 0 and 2)** **This control bit is validated when the CPU clock X2 is set. When X2 is low, this bit has no effect.** Clear to select 6 clock periods per peripheral clock cycle. Set to select 12 clock periods per peripheral clock cycle.
3	T2X2	**Timer2 Clock** **This control bit is validated when the CPU clock X2 is set. When X2 is low, this bit has no effect.** Clear to select 6 clock periods per peripheral clock cycle. Set to select 12 clock periods per peripheral clock cycle.
2	T1X2	**Timer1 Clock** **This control bit is validated when the CPU clock X2 is set. When X2 is low, this bit has no effect.** Clear to select 6 clock periods per peripheral clock cycle. Set to select 12 clock periods per peripheral clock cycle.
1	T0X2	**Timer0 Clock** **This control bit is validated when the CPU clock X2 is set. When X2 is low, this bit has no effect.** Clear to select 6 clock periods per peripheral clock cycle. Set to select 12 clock periods per peripheral clock cycle.
0	X2	**System Clock Control bit** Clear to select 12 clock periods per machine cycle (STD mode, $F_{CPU} = F_{PER} = F_{OSC}/2$). Set to select 6 clock periods per machine cycle (X2 mode, $F_{CPU} = F_{PER} = F_{OSC}$).

Reset Value = 0000 0000b

Table 15. CKCON1 (S:AFh)
Clock Control Register 1

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	SPIX2

Bit Number	Bit Mnemonic	Description
7-1	-	**Reserved** The value read from this bit is always 0. Do not set this bit.
0	SPIX2	**SPI Clock** **This control bit is validated when the CPU clock X2 is set. When X2 is low, this bit has no effect.** Clear to select 6 clock periods per peripheral clock cycle. Set to select 12 clock periods per peripheral clock cycle.

Reset Value = 0000 0000b

Table 16. PLLCON (S:A3h)
PLL Control Register

Table 17.

7	6	5	4	3	2	1	0
-	-	-	-	-	EXT48	PLLEN	PLOCK

Bit Number	Bit Mnemonic	Description
7-3	-	**Reserved** The value read from this bit is always 0. Do not set this bit.
2	EXT48	**External 48 MHz Enable Bit** Set this bit to bypass the PLL and disable the crystal oscillator. Clear this bit to select the PLL output as USB clock and to enable the crystal oscillator.
1	PLLEN	**PLL Enable Bit** Set to enable the PLL. Clear to disable the PLL.
0	PLOCK	**PLL Lock Indicator** Set by hardware when PLL is locked. Clear by hardware when PLL is unlocked.

Reset Value = 0000 0000b

Table 18. PLLDIV (S:A4h)
PLL Divider Register

Table 19.

7	6	5	4	3	2	1	0
R3	R2	R1	R0	N3	N2	N1	N0

Bit Number	Bit Mnemonic	Description
7-4	R3:0	**PLL R Divider Bits**

Bit Number	Bit Mnemonic	Description
3-0	N3:0	**PLL N Divider Bits**

Reset Value = 0000 0000

SFR Mapping

The Special Function Registers (SFRs) of the AT89C5130A/31A-M fall into the following categories:

- C51 core registers: ACC, B, DPH, DPL, PSW, SP
- I/O port registers: P0, P1, P2, P3, P4
- Timer registers: T2CON, T2MOD, TCON, TH0, TH1, TH2, TMOD, TL0, TL1, TL2, RCAP2L, RCAP2H
- Serial I/O port registers: SADDR, SADEN, SBUF, SCON
- PCA (Programmable Counter Array) registers: CCON, CMOD, CCAPMx, CL, CH, CCAPxH, CCAPxL (x: 0 to 4)
- Power and clock control registers: PCON
- Hardware Watchdog Timer registers: WDTRST, WDTPRG
- Interrupt system registers: IEN0, IPL0, IPH0, IEN1, IPL1, IPH1
- Keyboard Interface registers: KBE, KBF, KBLS
- LED register: LEDCON
- Two Wire Interface (TWI) registers: SSCON, SSCS, SSDAT, SSADR
- Serial Port Interface (SPI) registers: SPCON, SPSTA, SPDAT
- USB registers: Uxxx (17 registers)
- PLL registers: PLLCON, PLLDIV
- BRG (Baud Rate Generator) registers: BRL, BDRCON
- Flash register: FCON (FCON access is reserved for the Flash API and ISP software)
- EEPROM register: EECON
- Others: AUXR, AUXR1, CKCON0, CKCON1

The table below shows all SFRs with their address and their reset value.

Table 20. SFR Descriptions

	Bit Addressable	**Non-Bit Addressable**							
	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F	
F8h	UEPINT 0000 0000	CH 0000 0000	CCAP0H XXXX XXXX	CCAP1H XXXX XXXX	CCAP2H XXXX XXXX	CCAP3H XXXX XXXX	CCAP4H XXXX XXXX		FFh
F0h	B 0000 0000	LEDCON 0000 0000							F7h
E8h		CL 0000 0000	CCAP0L XXXX XXXX	CCAP1L XXXX XXXX	CCAP2L XXXX XXXX	CCAP3L XXXX XXXX	CCAP4L XXXX XXXX		EFh
E0h	ACC 0000 0000		UBYCTLX 0000 0000	UBYCTHX 0000 0000					E7h
D8h	CCON 00X0 0000	CMOD 00XX X000	CCAPM0 X000 0000	CCAPM1 X000 0000	CCAPM2 X000 0000	CCAPM3 X000 0000	CCAPM4 X000 0000		DFh
D0h	PSW 0000 0000	FCON (1) XXXX 0000	EECON XXXX XX00		UEPCONX 1000 0000	UEPRST 0000 0000			D7h
C8h	T2CON 0000 0000	T2MOD XXXX XX00	RCAP2L 0000 0000	RCAP2H 0000 0000	TL2 0000 0000	TH2 0000 0000	UEPSTAX 0000 0000	UEPDATX 0000 0000	CFh
C0h	P4 XXXX 1111		UEPIEN 0000 0000	SPCON 0001 0100	SPSTA 0000 0000	SPDAT XXXX XXXX	USBADDR 1000 0000	UEPNUM 0000 0000	C7h
B8h	IPL0 X000 000	SADEN 0000 0000	UFNUML 0000 0000	UFNUMH 0000 0000	USBCON 0000 0000	USBINT 0000 0000	USBIEN 0000 0000		BFh
B0h	P3 1111 1111	IEN1 X0XX X000	IPL1 X0XX X000	IPH1 X0XX X000				IPH0 X000 0000	B7h
A8h	IEN0 0000 0000	SADDR 0000 0000						CKCON1 0000 0000	AFh
A0h	P2 1111 1111		AUXR1 XXXX X0X0	PLLCON XXXX XX00	PLLDIV 0000 0000		WDTRST XXXX XXXX	WDTPRG XXXX X000	A7h
98h	SCON 0000 0000	SBUF XXXX XXXX	BRL 0000 0000	BDRCON XXX0 0000	KBLS 0000 0000	KBE 0000 0000	KBF 0000 0000		9Fh
90h	P1 1111 1111			SSCON 0000 0000	SSCS 1111 1000	SSDAT 1111 1111	SSADR 1111 1110		97h
88h	TCON 0000 0000	TMOD 0000 0000	TL0 0000 0000	TL1 0000 0000	TH0 0000 0000	TH1 0000 0000	AUXR XX0X 0000	CKCON0 0000 0000	8Fh
80h	P0 1111 1111	SP 0000 0111	DPL 0000 0000	DPH 0000 0000				PCON 00X1 0000	87h
	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F	

Note: 1. FCON access is reserved for the Flash API and ISP software.

Reserved

The Special Function Registers (SFRs) of the AT89C5131 fall into the following categories:

Table 21. C51 Core SFRs

Mnemonic	Add	Name	7	6	5	4	3	2	1	0
ACC	E0h	Accumulator								
B	F0h	B Register								
PSW	D0h	Program Status Word								
SP	81h	Stack Pointer LSB of SPX								
DPL	82h	Data Pointer Low byte LSB of DPTR								
DPH	83h	Data Pointer High byte MSB of DPTR								

Table 22. I/O Port SFRs

Mnemonic	Add	Name	7	6	5	4	3	2	1	0
P0	80h	Port 0								
P1	90h	Port 1								
P2	A0h	Port 2								
P3	B0h	Port 3								
P4	C0h	Port 4 (2bits)								

Table 23. Timer SFR's

Mnemonic	Add	Name	7	6	5	4	3	2	1	0
TH0	8Ch	Timer/Counter 0 High byte								
TL0	8Ah	Timer/Counter 0 Low byte								
TH1	8Dh	Timer/Counter 1 High byte								
TL1	8Bh	Timer/Counter 1 Low byte								
TH2	CDh	Timer/Counter 2 High byte								
TL2	CCh	Timer/Counter 2 Low byte								
TCON	88h	Timer/Counter 0 and 1 control	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
TMOD	89h	Timer/Counter 0 and 1 Modes	GATE1	C/T1#	M11	M01	GATE0	C/T0#	M10	M00
T2CON	C8h	Timer/Counter 2 control	TF2	EXF2	RCLK	TCLK	EXEN2	TR2	C/T2#	CP/RL2#
T2MOD	C9h	Timer/Counter 2 Mode							T2OE	DCEN
RCAP2H	CBh	Timer/Counter 2 Reload/Capture High byte								
RCAP2L	CAh	Timer/Counter 2 Reload/Capture Low byte								
WDTRST	A6h	WatchDog Timer Reset								
WDTPRG	A7h	WatchDog Timer Program						S2	S1	S0

Table 24. Serial I/O Port SFR's

Mnemonic	Add	Name	7	6	5	4	3	2	1	0
SCON	98h	Serial Control	FE/SM0	SM1	SM2	REN	TB8	RB8	TI	RI
SBUF	99h	Serial Data Buffer								
SADEN	B9h	Slave Address Mask								
SADDR	A9h	Slave Address								

Table 25. Baud Rate Generator SFR's

Mnemonic	Add	Name	7	6	5	4	3	2	1	0
BRL	9Ah	Baud Rate Reload								
BDRCON	9Bh	Baud Rate Control				BRR	TBCK	RBCK	SPD	SRC

Table 26. PCA SFR's

Mnemo-nic	Add	Name	7	6	5	4	3	2	1	0
CCON	D8h	PCA Timer/Counter Control	CF	CR		CCF4	CCF3	CCF2	CCF1	CCF0
CMOD	D9h	PCA Timer/Counter Mode	CIDL	WDTE				CPS1	CPS0	ECF
CL	E9h	PCA Timer/Counter Low byte								
CH	F9h	PCA Timer/Counter High byte								
CCAPM0 CCAPM1 CCAPM2 CCAPM3 CCAPM4	DAh DBh DCh DDh DEh	PCA Timer/Counter Mode 0 PCA Timer/Counter Mode 1 PCA Timer/Counter Mode 2 PCA Timer/Counter Mode 3 PCA Timer/Counter Mode 4		ECOM0 ECOM1 ECOM2 ECOM3 ECOM4	CAPP0 CAPP1 CAPP2 CAPP3 CAPP4	CAPN0 CAPN1 CAPN2 CAPN3 CAPN4	MAT0 MAT1 MAT2 MAT3 MAT4	TOG0 TOG1 TOG2 TOG3 TOG4	PWM0 PWM1 PWM2 PWM3 PWM4	ECCF0 ECCF1 ECCF2 ECCF3 ECCF4
CCAP0H CCAP1H CCAP2H CCAP3H CCAP4H	FAh FBh FCh FDh FEh	PCA Compare Capture Module 0 H PCA Compare Capture Module 1 H PCA Compare Capture Module 2 H PCA Compare Capture Module 3 H PCA Compare Capture Module 4 H	CCAP0H7 CCAP1H7 CCAP2H7 CCAP3H7 CCAP4H7	CCAP0H6 CCAP1H6 CCAP2H6 CCAP3H6 CCAP4H6	CCAP0H5 CCAP1H5 CCAP2H5 CCAP3H5 CCAP4H5	CCAP0H4 CCAP1H4 CCAP2H4 CCAP3H4 CCAP4H4	CCAP0H3 CCAP1H3 CCAP2H3 CCAP3H3 CCAP4H3	CCAP0H2 CCAP1H2 CCAP2H2 CCAP3H2 CCAP4H2	CCAP0H1 CCAP1H1 CCAP2H1 CCAP3H1 CCAP4H1	CCAP0H0 CCAP1H0 CCAP2H0 CCAP3H0 CCAP4H0
CCAP0L CCAP1L CCAP2L CCAP3L CCAP4L	EAh EBh ECh EDh EEh	PCA Compare Capture Module 0 L PCA Compare Capture Module 1 L PCA Compare Capture Module 2 L PCA Compare Capture Module 3 L PCA Compare Capture Module 4 L	CCAP0L7 CCAP1L7 CCAP2L7 CCAP3L7 CCAP4L7	CCAP0L6 CCAP1L6 CCAP2L6 CCAP3L6 CCAP4L6	CCAP0L5 CCAP1L5 CCAP2L5 CCAP3L5 CCAP4L5	CCAP0L4 CCAP1L4 CCAP2L4 CCAP3L4 CCAP4L4	CCAP0L3 CCAP1L3 CCAP2L3 CCAP3L3 CCAP4L3	CCAP0L2 CCAP1L2 CCAP2L2 CCAP3L2 CCAP4L2	CCAP0L1 CCAP1L1 CCAP2L1 CCAP3L1 CCAP4L1	CCAP0L0 CCAP1L0 CCAP2L0 CCAP3L0 CCAP4L0

Table 27. Interrupt SFR's

Mnemo-nic	Add	Name	7	6	5	4	3	2	1	0
IEN0	A8h	Interrupt Enable Control 0	EA	EC	ET2	ES	ET1	EX1	ET0	EX0
IEN1	B1h	Interrupt Enable Control 1		EUSB				ESPI	ETWI	EKB
IPL0	B8h	Interrupt Priority Control Low 0		PPCL	PT2L	PSL	PT1L	PX1L	PT0L	PX0L
IPH0	B7h	Interrupt Priority Control High 0		PPCH	PT2H	PSH	PT1H	PX1H	PT0H	PX0H
IPL1	B2h	Interrupt Priority Control Low 1		PUSBL				PSPIL	PTWIL	PKBL
IPH1	B3h	Interrupt Priority Control High 1		PUSBH				PSPIH	PTWIH	PKBH

Table 28. PLL SFRs

Mnemonic	Add	Name	7	6	5	4	3	2	1	0
PLLCON	A3h	PLL Control						EXT48	PLLEN	PLOCK
PLLDIV	A4h	PLL Divider	R3	R2	R1	R0	N3	N2	N1	N0

Table 29. Keyboard SFRs

Mnemonic	Add	Name	7	6	5	4	3	2	1	0
KBF	9Eh	Keyboard Flag Register	KBF7	KBF6	KBF5	KBF4	KBF3	KBF2	KBF1	KBF0
KBE	9Dh	Keyboard Input Enable Register	KBE7	KBE6	KBE5	KBE4	KBE3	KBE2	KBE1	KBE0
KBLS	9Ch	Keyboard Level Selector Register	KBLS7	KBLS6	KBLS5	KBLS4	KBLS3	KBLS2	KBLS1	KBLS0

Table 30. TWI SFRs

Mnemonic	Add	Name	7	6	5	4	3	2	1	0
SSCON	93h	Synchronous Serial Control	CR2	SSIE	STA	STO	SI	AA	CR1	CR0
SSCS	94h	Synchronous Serial Control-Status	SC4	SC3	SC2	SC1	SC0	-	-	-
SSDAT	95h	Synchronous Serial Data	SD7	SD6	SD5	SD4	SD3	SD2	SD1	SD0
SSADR	96h	Synchronous Serial Address	A7	A6	A5	A4	A3	A2	A1	A0

Table 31. SPI SFRs

Mnemonic	Add	Name	7	6	5	4	3	2	1	0
SPCON	C3h	Serial Peripheral Control	SPR2	SPEN	SSDIS	MSTR	CPOL	CPHA	SPR1	SPR0
SPSTA	C4h	Serial Peripheral Status-Control	SPIF	WCOL	SSERR	MODF	-	-	-	-
SPDAT	C5h	Serial Peripheral Data	R7	R6	R5	R4	R3	R2	R1	R0

Table 32. USB SFR's

Mnemonic	Add	Name	7	6	5	4	3	2	1	0
USBCON	BCh	USB Global Control	USBE	SUSPCLK	SDRMWUP	DETACH	UPRSM	RMWUPE	CONFG	FADDEN
USBADDR	C6h	USB Address	FEN	UADD6	UADD5	UADD4	UADD3	UADD2	UADD1	UADD0
USBINT	BDh	USB Global Interrupt	-	-	WUPCPU	EORINT	SOFINT	-	-	SPINT
USBIEN	BEh	USB Global Interrupt Enable	-	-	EWUPCPU	EEORINT	ESOFINT	-	-	ESPINT
UEPNUM	C7h	USB Endpoint Number	-	-	-	-	EPNUM3	EPNUM2	EPNUM1	EPNUM0
UEPCONX	D4h	USB Endpoint X Control	EPEN	-	-	-	DTGL	EPDIR	EPTYPE1	EPTYPE0
UEPSTAX	CEh	USB Endpoint X Status	DIR	RXOUTB1	STALLRQ	TXRDY	STLCRC	RXSETUP	RXOUTB0	TXCMP
UEPRST	D5h	USB Endpoint Reset	-	EP6RST	EP5RST	EP4RST	EP3RST	EP2RST	EP1RST	EP0RST
UEPINT	F8h	USB Endpoint Interrupt	-	EP6INT	EP5INT	EP4INT	EP3INT	EP2INT	EP1INT	EP0INT

Table 32. USB SFR's

Mnemonic	Add	Name	7	6	5	4	3	2	1	0
UEPIEN	C2h	USB Endpoint Interrupt Enable	-	EP6INTE	EP5INTE	EP4INTE	EP3INTE	EP2INTE	EP1INTE	EP0INTE
UEPDATX	CFh	USB Endpoint X FIFO Data	FDAT7	FDAT6	FDAT5	FDAT4	FDAT3	FDAT2	FDAT1	FDAT0
UBYCTLX	E2h	USB Byte Counter Low (EP X)	BYCT7	BYCT6	BYCT5	BYCT4	BYCT3	BYCT2	BYCT1	BYCT0
UBYCTHX	E3h	USB Byte Counter High (EP X)	-	-	-	-	-	BYCT10	BYCT9	BYCT8
UFNUML	BAh	USB Frame Number Low	FNUM7	FNUM6	FNUM5	FNUM4	FNUM3	FNUM2	FNUM1	FNUM0
UFNUMH	BBh	USB Frame Number High	-	-	CRCOK	CRCERR	-	FNUM10	FNUM9	FNUM8

Table 33. Other SFR's

Mnemonic	Add	Name	7	6	5	4	3	2	1	0
PCON	87h	Power Control	SMOD1	SMOD0	-	POF	GF1	GF0	PD	IDL
AUXR	8Eh	Auxiliary Register 0	DPU	-	M0	-	XRS1	XRS2	EXTRAM	A0
AUXR1	A2h	Auxiliary Register 1	-	-	ENBOOT	-	GF3	-	-	DPS
CKCON0	8Fh	Clock Control 0	-	WDX2	PCAX2	SIX2	T2X2	T1X2	T0X2	X2
CKCON1	AFh	Clock Control 1	-	-	-	-	-	-	-	SPIX2
LEDCON	F1h	LED Control	LED3		LED2		LED1		LED0	
FCON	D1h	Flash Control	FPL3	FPL2	FPL1	FPL0	FPS	FMOD1	FMOD0	FBUSY
EECON	D2h	EEPROM Contol	EEPL3	EEPL2	EEPL1	EEPL0	-	-	EEE	EEBUSY

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

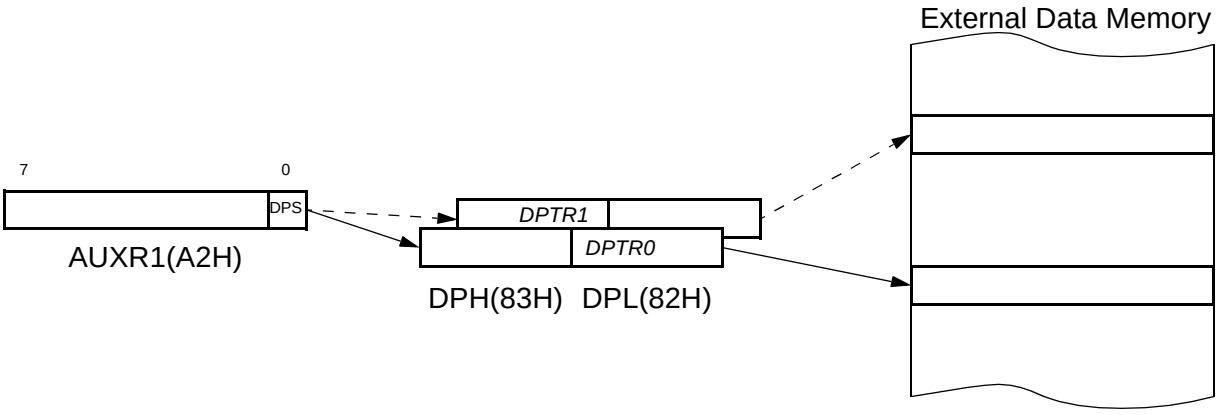


Table 34. AUXR1 Register

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.

ASSEMBLY LANGUAGE

```
; Block move using dual data pointers
; Modifies DPTR0, DPTR1, A and PSW
; note: DPS exits opposite of entry state
; unless an extra INC AUXR1 is added
;
00A2    AUXR1 EQU 0A2H
;
0000 909000MOV DPTR,#SOURCE ; address of SOURCE
0003 05A2 INC AUXR1 ; switch data pointers
0005 90A000 MOV DPTR,#DEST ; address of DEST
0008    LOOP:
0008 05A2 INC AUXR1 ; switch data pointers
000A E0 MOVX A,@DPTR ; get a byte from SOURCE
000B A3 INC DPTR ; increment SOURCE address
000C 05A2 INC AUXR1 ; switch data pointers
000E F0 MOVX @DPTR,A ; write the byte to DEST
000F A3 INC DPTR ; increment DEST address
0010 70F6JNZ LOOP ; check for 0 terminator
0012 05A2 INC AUXR1 ; (optional) restore DPS
```

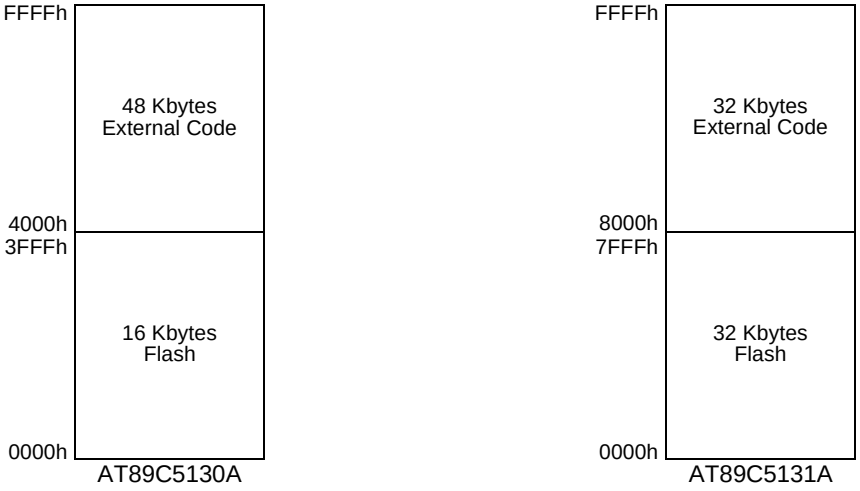
INC is a short (2 bytes) and fast (12 clocks) way to manipulate the DPS bit in the AUXR1 SFR. However, note that the INC instruction does not directly force the DPS bit to a particular state, but simply toggles it. In simple routines, such as the block move example, only the fact that DPS is toggled in the proper sequence matters, not its actual value. In other words, the block move routine works the same whether DPS is '0' or '1' on entry. Observe that without the last instruction (INC AUXR1), the routine will exit with DPS in the opposite state.

Program/Code Memory

The AT89C5130A/31A-M implement 16/ 32 Kbytes of on-chip program/code memory. Figure 11 shows the split of internal and external program/code memory spaces depending on the product.

The Flash memory increases EPROM and ROM functionality by in-circuit electrical erasure and programming. Thanks to the internal charge pump, the high voltage needed for programming or erasing Flash cells is generated on-chip using the standard V_{DD} voltage. Thus, the Flash Memory can be programmed using only one voltage and allows In-application Software Programming commonly known as IAP. Hardware programming mode is also available using specific programming tool.

Figure 11. Program/Code Memory Organization



Note: If the program executes exclusively from on-chip code memory (not from external memory), beware of executing code from the upper byte of on-chip memory (3FFFh/7FFFh) and thereby disrupting I/O Ports 0 and 2 due to external prefetch. Fetching code constant from this location does not affect Ports 0 and 2.

External Code Memory Access

Memory Interface

The external memory interface comprises the external bus (Port 0 and Port 2) as well as the bus control signals (PSEN, and ALE).

Figure 12 shows the structure of the external address bus. P0 carries address A7:0 while P2 carries address A15:8. Data D7:0 is multiplexed with A7:0 on P0. Table 35 describes the external memory interface signals.

Figure 12. External Code Memory Interface Structure

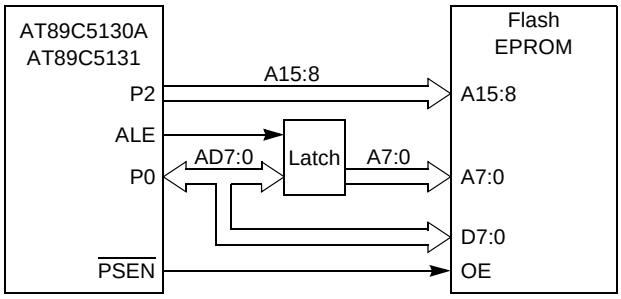


Table 35. External Data Memory Interface Signals

Signal Name	Type	Description	Alternate Function
A15:8	O	**Address Lines** Upper address lines for the external bus.	P2.7:0
AD7:0	I/O	**Address/Data Lines** Multiplexed lower address lines and data for the external memory.	P0.7:0
ALE	O	**Address Latch Enable** ALE signals indicates that valid address information are available on lines AD7:0.	-
$\overline{\text{PSEN}}$	O	**Program Store Enable Output** This signal is active low during external code fetch or external code read (MOVC instruction).	-

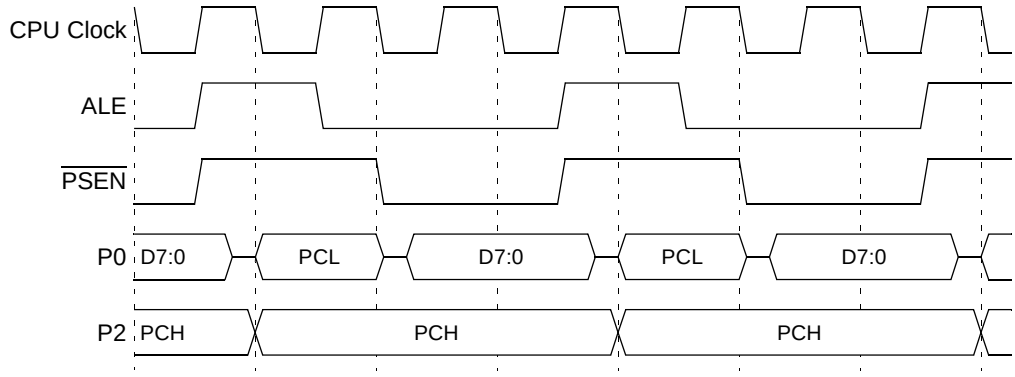
External Bus Cycles

This section describes the bus cycles the AT89C5130A/31A-M executes to fetch code (see Figure 13) in the external program/code memory.

External memory cycle takes 6 CPU clock periods. This is equivalent to 12 oscillator clock periods in standard mode or 6 oscillator clock periods in X2 mode. For further information on X2 mode (see the clock Section).

For simplicity, the accompanying figure depicts the bus cycle waveforms in idealized form and do not provide precise timing information.

Figure 13. External Code Fetch Waveforms



Flash Memory Architecture

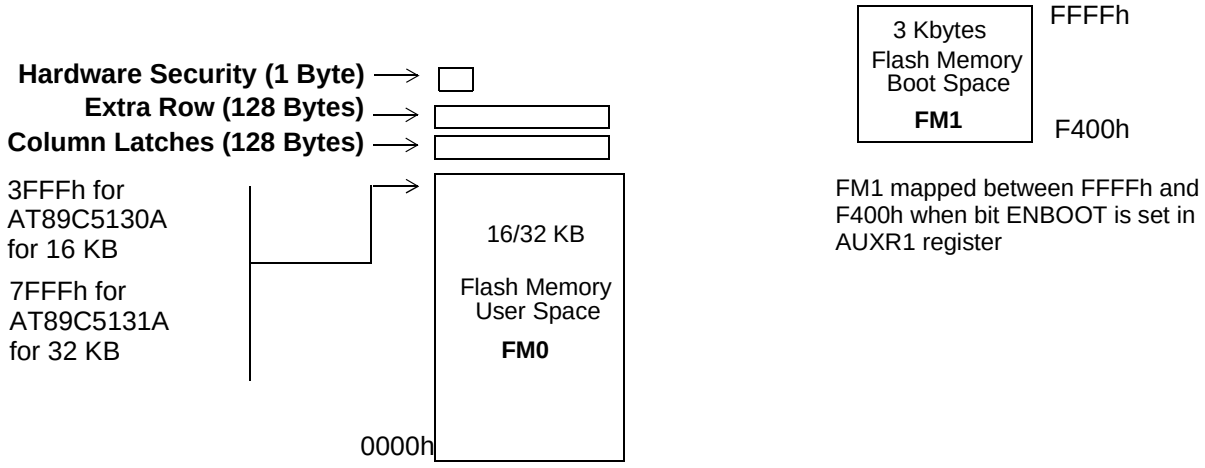
AT89C5130A/31A-M features two on-chip Flash memories:

- Flash memory FM0:
  containing 32 Kbytes of program memory (user space) organized into 128-byte pages,
- Flash memory FM1:
  3 Kbytes for bootloader and Application Programming Interfaces (API).

The FM0 supports both parallel programming and Serial In-System Programming (ISP) whereas FM1 supports only parallel programming by programmers. The ISP mode is detailed in the “In-System Programming” section.

All Read/Write access operations on Flash memory by user application are managed by a set of API described in the “In-System Programming” section.

Figure 14. Flash Memory Architecture



FM0 Memory Architecture

The Flash memory is made up of 4 blocks (see Figure 14):

1. The memory array (user space) 32 Kbytes
2. The Extra Row
3. The Hardware security bits
4. The column latch registers

User Space

This space is composed of a 16/32 Kbytes Flash memory organized in 128/256 pages of 128 bytes. It contains the user's application code.

Extra Row (XRow)

This row is a part of FM0 and has a size of 128 bytes. The extra row may contain information for bootloader usage.

Hardware Security Space

The hardware security space is a part of FM0 and has a size of 1 byte.
The 4 MSB can be read/written by software. The 4 LSB can only be read by software and written by hardware in parallel mode.

Column Latches

The column latches, also part of FM0, have a size of full page (128 bytes).
The column latches are the entrance buffers of the three previous memory locations (user array, XRow and Hardware security byte).

Overview of FM0 Operations

The CPU interfaces to the Flash memory through the FCON register and AUXR1 register.

These registers are used to:

- Map the memory spaces in the adressable space
- Launch the programming of the memory spaces
- Get the status of the Flash memory (busy/not busy)
- Select the Flash memory FM0/FM1.

Mapping of the Memory Space

By default, the user space is accessed by MOVC instruction for read only. The column latches space is made accessible by setting the FPS bit in FCON register. Writing is possible from 0000h to 3FFFH/7FFFh, address bits 6 to 0 are used to select an address within a page while bits 14 to 7 are used to select the programming address of the page.

Setting this bit takes precedence on the EXTRAM bit in AUXR register.

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.

Loading the Column Latches

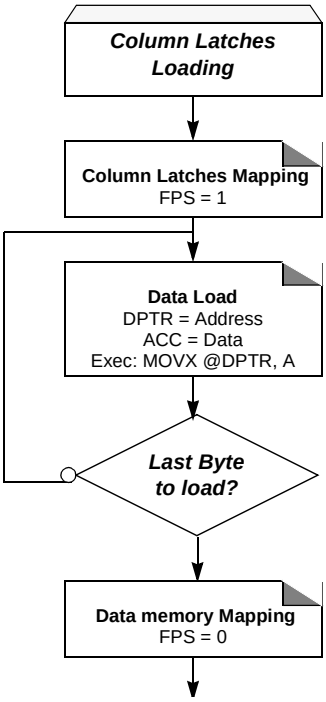
Any number of data from 1 byte to 128 bytes can be loaded in the column latches. This provides the capability to program the whole memory by byte, by page or by any number of bytes in a page.

When programming is launched, an automatic erase of the locations loaded in the column latches is first performed, then programming is effectively done. Thus, no page or block erase is needed and only the loaded data are programmed in the corresponding page.

The following procedure is used to load the column latches and is summarized in Figure 15:

- Map the column latch space by setting FPS bit.
- Load the DPTR with the address to load.
- Load Accumulator register with the data to load.
- Execute the MOVX @DPTR, A instruction.
- If needed loop the three last instructions until the page is completely loaded.

Figure 15. Column Latches Loading Procedure



Programming the Flash Spaces

User

The following procedure is used to program the User space and is summarized in Figure 16:

- Load data in the column latches from address 0000h to 7FFFh(1).
- Disable the interrupts.
- Launch the programming by writing the data sequence 50h followed by A0h in FCON register.
  The end of the programming indicated by the FBUSY flag cleared.
- Enable the interrupts.

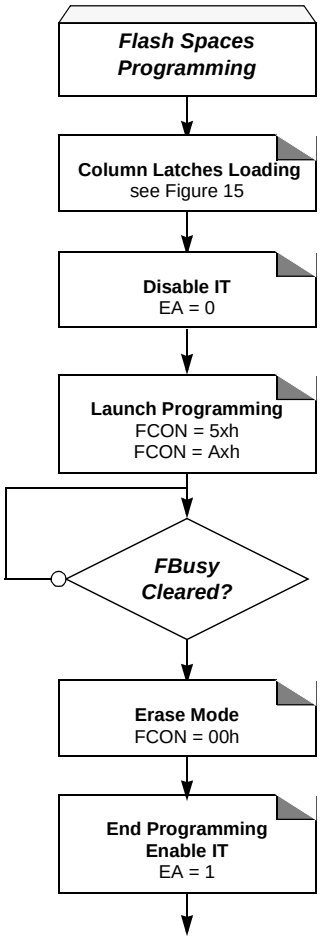
Note: 1. The last page address used when loading the column latch is the one used to select the page programming address.

Extra Row

The following procedure is used to program the Extra Row space and is summarized in Figure 16:

- Load data in the column latches from address FF80h to FFFFh.
- Disable the interrupts.
- Launch the programming by writing the data sequence 52h followed by A2h in FCON register.
  The end of the programming indicated by the FBUSY flag cleared.
- Enable the interrupts.

Figure 16. Flash and Extra Row Programming Procedure

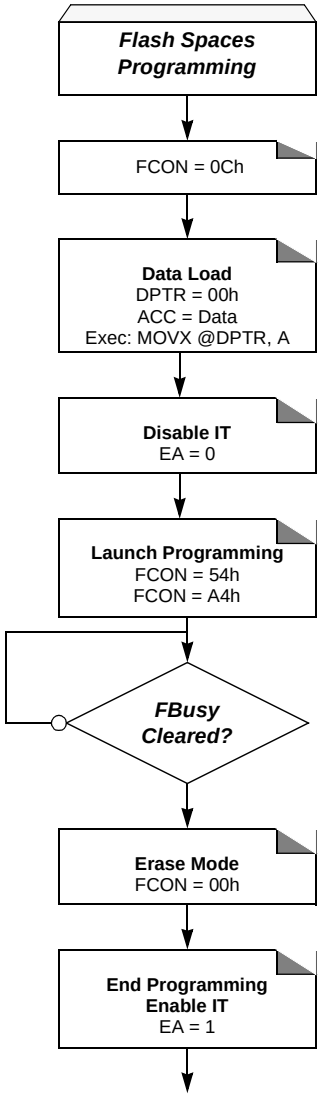


Hardware Security

The following procedure is used to program the Hardware Security space and is summarized in Figure 17:

- Set FPS and map Hardware byte (FCON = 0x0C)
- Disable the interrupts.
- Load DPTR at address 0000h.
- Load Accumulator register with the data to load.
- Execute the MOVX @DPTR, A instruction.
- Launch the programming by writing the data sequence 54h followed by A4h in FCON register.
  The end of the programming indicated by the FBusy flag cleared.
- Enable the interrupts.

Figure 17. Hardware Programming Procedure



Reading the Flash Spaces

User

The following procedure is used to read the User space and is summarized in Figure 18:

- Map the User space by writing 00h in FCON register.
- Read one byte in Accumulator by executing MOVC A, @A+DPTR with A = 0 & DPTR = 0000h to FFFFh.

Extra Row

The following procedure is used to read the Extra Row space and is summarized in Figure 18:

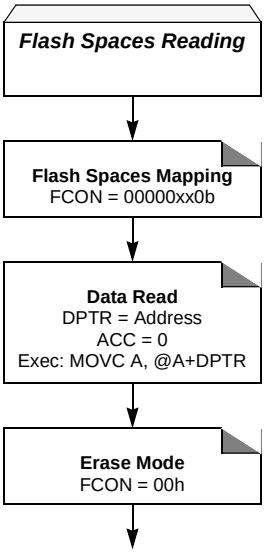
- Map the Extra Row space by writing 02h in FCON register.
- Read one byte in Accumulator by executing MOVC A, @A+DPTR with A = 0 & DPTR = FF80h to FFFFh.

Hardware Security

The following procedure is used to read the Hardware Security space and is summarized in Figure 18:

- Map the Hardware Security space by writing 04h in FCON register.
- Read the byte in Accumulator by executing MOVC A, @A+DPTR with A = 0 & DPTR = 0000h.

Figure 18. Reading Procedure



Registers

Table 38. FCON (S:D1h)
Flash Control Register

7	6	5	4	3	2	1	0
FPL3	**FPL2**	**FPL1**	**FPL0**	**FPS**	**FMOD1**	**FMOD0**	**FBUSY**

Bit Number	Bit Mnemonic	Description
7-4	FPL3:0	**Programming Launch Command Bits** Write 5Xh followed by AXh to launch the programming according to FMOD1:0. (see Table 37.)
3	FPS	**Flash Map Program Space** Set to map the column latch space in the data memory space. Clear to re-map the data memory space.
2-1	FMOD1:0	**Flash Mode** See Table 36 or Table 37.
0	FBUSY	**Flash Busy** Set by hardware when programming is in progress. Clear by hardware when programming is done. Can not be cleared by software.

Reset Value = 0000 0000b

Flash EEPROM Memory

General Description

The Flash memory increases EPROM functionality with in-circuit electrical erasure and programming. It contains 16/32 Kbytes of program memory organized in 128/256 pages of 128 bytes, respectively. This memory is both parallel and serial In-System Programmable (ISP). ISP allows devices to alter their own program memory in the actual end product under software control. A default serial loader (bootloader) program allows ISP of the Flash.

The programming does not require 12V external programming voltage. The necessary high programming voltage is generated on-chip using the standard V_{CC} pins of the microcontroller.

Features

- Flash EEPROM internal program memory.
- Boot vector allows user-provided Flash loader code to reside anywhere in the Flash memory space. This configuration provides flexibility to the user.
- Default loader in Boot EEPROM allows programming via the serial port without the need of a user provided loader.
- Up to 64K bytes external program memory if the internal program memory is disabled (EA = 0).
- Programming and erase voltage with standard power supply.
- Read/Program/Erase:
- Byte-wise read (without wait state).
- Byte or page erase and programming (10 ms).
- Typical programming time (32 Kbytes) in 4.5 sec.
- Parallel programming with 87C51 compatible hardware interface to programmer.
- Programmable security for the code in the Flash.
- 100K write cycles
- 10 years data retention

Flash Programming and Erasure

The 16/32 Kbytes Flash is programmed by bytes or by pages of 128 bytes. It is not necessary to erase a byte or a page before programming. The programming of a byte or a page includes a self erase before programming.

There are three methods of programming the Flash memory:

1. The on-chip ISP bootloader may be invoked which will use low level routines to program the pages. The interface used for serial downloading of Flash is the USB.
2. The Flash may be programmed or erased in the end-user application by calling low-level routines through a common entry point in the Boot Flash.
3. The Flash may be programmed using the parallel method.

The bootloader and the Application Programming Interface (API) routines are located in the Flash Bootloader.

Flash Registers and Memory Map

The AT89C5130A/31A-M Flash memory uses several registers:

- Hardware registers can only be accessed through the parallel programming modes which are handled by the parallel programmer.
- Software registers are in a special page of the Flash memory which can be accessed through the API or with the parallel programming modes. This page, called “Extra Flash Memory”, is not in the internal Flash program memory addressing space.

Hardware Registers

The only hardware registers of the AT89C5130A/31A-M is called Hardware Security Byte (HSB).

Table 39. Hardware Security Byte (HSB)

7	6	5	4	3	2	1	0
X2	**BLJB**	**OSCON1**	**OSCON0**	**-**	**LB2**	**LB1**	**LB0**

Bit Number	Bit Mnemonic	Description
7	X2	**X2 Mode** Cleared to force X2 mode (6 clocks per instruction) Set to force X1 mode, Standard Mode (Default).
6	BLJB	**Bootloader Jump Bit** Set this bit to start the user’s application on next reset at address 0000h. Cleared this bit to start the bootloader at address F400h (default).
5-4	OSCON1-0	**Oscillator Control Bits** These two bits are used to control the oscillator in order to reduce consumption. **OSCONOSCON0Description** 1 1 The oscillator is configured to run from 0 to 32 MHz 1 0 The oscillator is configured to run from 0 to 16 MHz 0 1 The oscillator is configured to run from 0 to 8 MHz 0 0 This configuration shouldn’t be set
3	-	**Reserved**
2-0	LB2-0	**User Memory Lock Bits** See Table 40

Bootloader Jump Bit (BLJB)

One bit of the HSB, the BLJB bit, is used to force the boot address:

- When this bit is set the boot address is 0000h.
- When this bit is reset the boot address is F400h. By default, this bit is cleared and the ISP is enabled.

Flash Memory Lock Bits

The three lock bits provide different levels of protection for the on-chip code and data, when programmed as shown in Table 40.

Table 40. Program Lock bits

Program Lock Bits				
Security level	**LB0**	**LB1**	**LB2**	**Protection Description**
1	U	U	U	No program lock features enabled.
2	P	U	U	MOVC instruction executed from external program memory is disabled from fetching code bytes from any internal memory, $\overline{EA}$ is sampled and latched on reset, and further parallel programming of the Flash and of the EEPROM (boot and Xdata) is disabled. ISP and software programming with API are still allowed.
3	X	P	U	Same as 2, also verify through parallel programming interface is disabled and serial programming ISP is still enabled.
4	X	X	P	Same as 3, also external execution is disabled.

Notes:
1. U: unprogrammed or “one” level.
2. P: programmed or “zero” level.
3. X: don’t care
4. WARNING: Security level 2 and 3 should only be programmed after Flash and code verification.

These security bits protect the code access through the parallel programming interface. They are set by default to level 4. The code access through the ISP is still possible and is controlled by the “software security bits” which are stored in the extra Flash memory accessed by the ISP firmware.

To load a new application with the parallel programmer, a chip erase must be done first. This will set the HSB in its inactive state and will erase the Flash memory. The part reference can always be read using Flash parallel programming modes.

Default Values

The default value of the HSB provides parts ready to be programmed with ISP:

- BLJB: Cleared to force ISP operation.
- X2: Set to force X1 mode (Standard Mode)
- OSCON1-0: Set to start with 32 MHz oscillator configuration value.
- LB2-0: Security level four to protect the code from a parallel access with maximum security.

Software Registers

Several registers are used, in factory and by parallel programmers, to make copies of hardware registers contents. These values are used by Atmel ISP (see Section “In-System Programming (ISP)”).

These registers are in the “Extra Flash Memory” part of the Flash memory. This block is also called ”XAF” or eXtra Array Flash. They are accessed in the following ways:

- Commands issued by the parallel memory programmer.
- Commands issued by the ISP software.
- Calls of API issued by the application software.

Several software registers are described in Table 41.

Table 41. Software Registers

Mnemonic	Description	Default value	
SBV	Software Boot Vector	FCh	–
HSB	Copy of the Hardware Security Byte	1011 1000b	–
BSB	Boot Status Byte	0FFh	–
SSB	Software Security Byte	FFh	–
–	Copy of the Manufacturer Code	58h	Atmel
–	Copy of the Device ID #1: Family Code	D7h	C51 X2, Electrically Erasable
–	Copy of the Device ID #2: Memories	F7h	AT89C5130A/31A-M 32 Kbyte
–	Copy of the Device ID #3: Name	EFh	AT89C5130A/31A-M 32 Kbyte, revision 0

After programming the part by ISP, the BSB must be cleared (00h) in order to allow the application to boot at 0000h.

The content of the Software Security Byte (SSB) is described in Table 42 and Table 43.

To assure code protection from a parallel access, the HSB must also be at the required level.

Table 42. Software Security Byte (SSB)

7	6	5	4	3	2	1	0
-	-	-	-	-	-	**LB1**	**LB0**

Bit Number	Bit Mnemonic	Description
7	-	Reserved Do not clear this bit.
6	-	Reserved Do not clear this bit.
5	-	Reserved Do not clear this bit.
4	-	Reserved Do not clear this bit.
3	-	Reserved Do not clear this bit.
2	-	Reserved Do not clear this bit.
1-0	LB1-0	User Memory Lock Bits See Table 43

The two lock bits provide different levels of protection for the on-chip code and data, when programmed as shown to Table 43.

Table 43. Program Lock Bits of the SSB

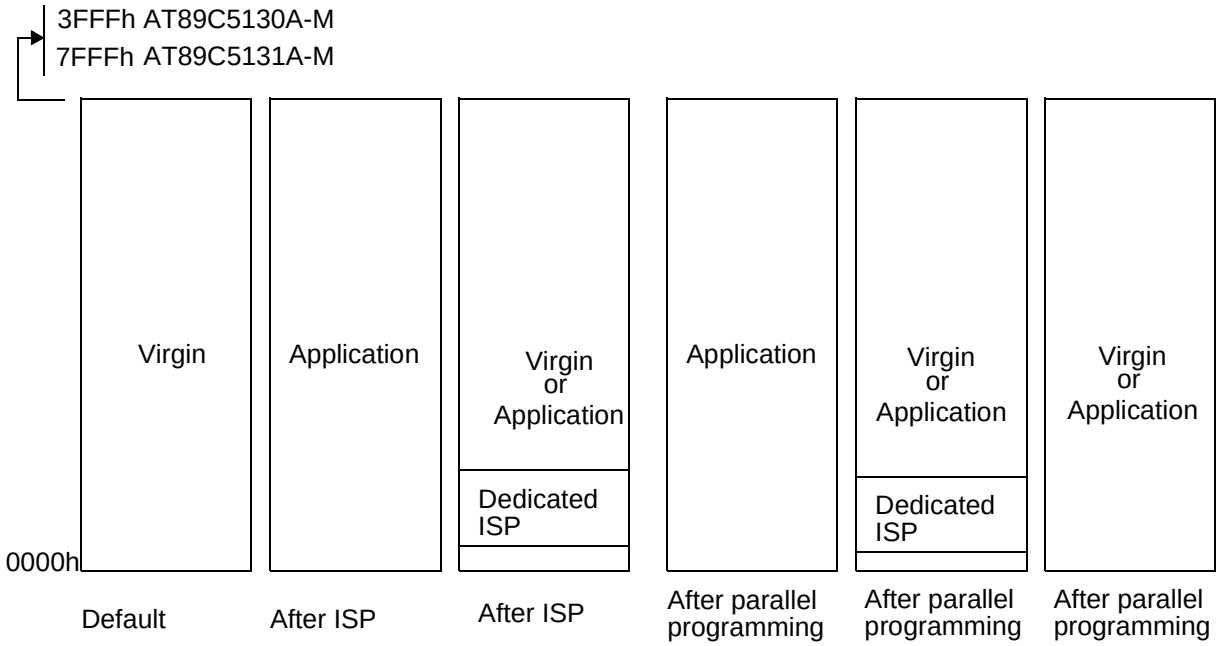
Program Lock Bits			
Security Level	**LB0**	**LB1**	**Protection Description**
1	U	U	No program lock features enabled.
2	P	U	ISP programming of the Flash is disabled.
3	X	P	Same as 2, also verify through ISP programming interface is disabled.

Notes:
1. U: unprogrammed or "one" level.
2. P: programmed or “zero” level.
3. X: don’t care
4. WARNING: Security level 2 and 3 should only be programmed after Flash and code verification.

Flash Memory Status

AT89C5130A/31A-M parts are delivered with the ISP boot in the Flash memory. After ISP or parallel programming, the possible contents of the Flash memory are summarized in Figure 19:

Figure 19. Flash Memory Possible Contents



Memory Organization

In the AT89C5130A/31A-M, the lowest 16/32K of the 64 Kbyte program memory address space is filled by internal Flash.

When the $\overline{EA}$ is pin high, the processor fetches instructions from internal program Flash. Bus expansion for accessing program memory from 16/32K upward is automatic since external instruction fetches occur automatically when the program counter exceeds 3FFFh (16K) or 7FFFh (32K). If the $\overline{EA}$ pin is tied low, all program memory fetches are from external memory. If all storage is on chip, then byte location 3FFFh (16K) or 7FFFh (32K) should be left vacant to prevent and undesired pre-fetch from external program memory address 4000h (16K) or 8000h (32K).

EEPROM Data Memory

Description

The 1-Kbyte on-chip EEPROM memory block is located at addresses 0000h to 03FFh of the ERAM memory space and is selected by setting control bits in the EECON register.

A read in the EEPROM memory is done with a MOVX instruction.

A physical write in the EEPROM memory is done in two steps: write data in the column latches and transfer of all data latches into an EEPROM memory row (programming).

The number of data written on the page may vary from 1 to 128 bytes (the page size). When programming, only the data written in the column latch is programmed and a ninth bit is used to obtain this feature. This provides the capability to program the whole memory by bytes, by page or by a number of bytes in a page. Indeed, each ninth bit is set when the writing the corresponding byte in a row and all these ninth bits are reset after the writing of the complete EEPROM row.

Write Data in the Column Latches

Data is written by byte to the column latches as for an external RAM memory. Out of the 11 address bits of the data pointer, the 4 MSBs are used for page selection (row) and 7 are used for byte selection. Between two EEPROM programming sessions, all the addresses in the column latches must stay on the same page, meaning that the 4 MSB must not be changed.

The following procedure is used to write to the column latches:

- Set bit EEE of EECON register
- Load DPTR with the address to write
- Store A register with the data to be written
- Execute a MOVX @DPTR, A
- If needed, loop the three last instructions until the end of a 128 bytes page

Programming

The EEPROM programming consists on the following actions:

- Writing one or more bytes of one page in the column latches. Normally, all bytes must belong to the same page; if not, the first page address will be latched and the others discarded.
- Launching programming by writing the control sequence (54h followed by A4h) to the EECON register.
- EEBUSY flag in EECON is then set by hardware to indicate that programming is in progress and that the EEPROM segment is not available for reading.
- The end of programming is indicated by a hardware clear of the EEBUSY flag.

Read Data

The following procedure is used to read the data stored in the EEPROM memory:

- Set bit EEE of EECON register
- Stretch the MOVX to accommodate the slow access time of the column latch (Set bit M0 of AUXR register)
- Load DPTR with the address to read
- Execute a MOVX A, @DPTR

Registers

Table 44. EECON (S:0D2h)
EECON Register

7	6	5	4	3	2	1	0
EEPL3	**EEPL2**	**EEPL1**	**EEPL0**	**-**	**-**	**EEE**	**EEBUSY**

Bit Number	Bit Mnemonic	Description
7-4	EEPL3-0	**Programming Launch command bits** Write 5Xh followed by AXh to EEPL to launch the programming.
3	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
2	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
1	EEE	**Enable EEPROM Space bit** Set to map the EEPROM space during MOVX instructions (Write in the column latches) Clear to map the ERAM space during MOVX.
0	EEBUSY	**Programming Busy flag** Set by hardware when programming is in progress. Cleared by hardware when programming is done. Cannot be set or cleared by software.

Reset Value = XXXX XX00b
Not bit addressable

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.

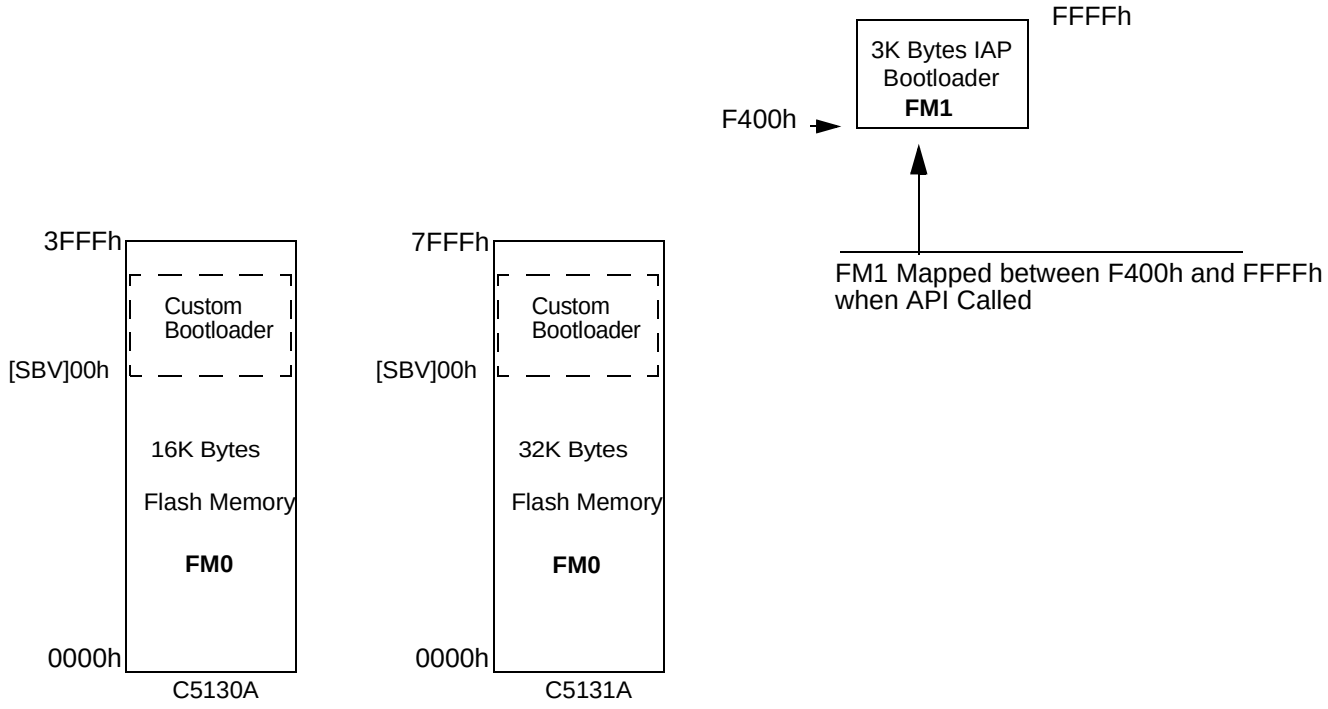
This allow the customer to have a full use of the 32-Kbyte user memory.

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



Boot Process

Software Boot Process Example

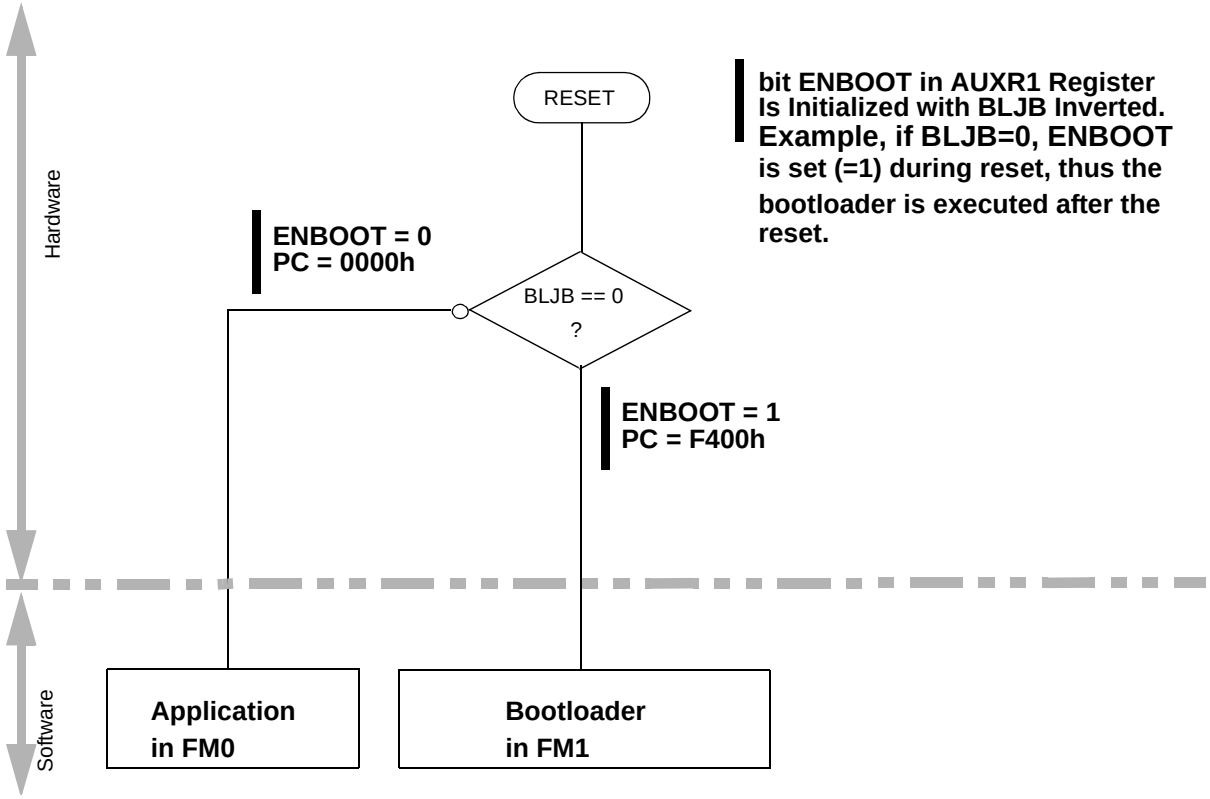
Many algorithms can be used for the software boot process. Below are descriptions of the different flags and Bytes.

Boot Loader Jump bit (BLJB):
- This bit indicates if on RESET the user wants to jump to this application at address @0000h on FM0 or execute the boot loader at address @F400h on FM1.
- BLJB = 0 (i.e. bootloader FM1 executed after a reset) is the default Atmel factory programming.
-To read or modify this bit, the APIs are used.

Boot Vector Address (SBV):
- This byte contains the MSB of the user boot loader address in FM0.
- The default value of SBV is FFh (no user boot loader in FM0).
- To read or modify this byte, the APIs are used.

Extra Byte (EB) & Boot Status Byte (BSB):
- These Bytes are reserved for customer use.
- To read or modify these Bytes, the APIs are used.

Figure 21. Hardware Boot Process Algorithm



Application-Programming-Interface

Several Application Program Interface (API) calls are available for use by an application program to permit selective erasing and programming of Flash pages. All calls are made by functions.

All these APIs are described in detail in the following document on the Atmel web site.

– Datasheet Bootloader USB AT89C5131.

XROW Bytes

The EXTRA ROW (XROW) includes 128 bytes. Some of these bytes are used for specific purpose in conjonction with the bootloader.

Table 45. XROW Mapping

Description	Default Value	Address
Copy of the Manufacturer Code	58h	30h
Copy of the Device ID#1: Family code	D7h	31h
Copy of the Device ID#2: Memories size and type	BBh	60h
Copy of the Device ID#3: Name and Revision	FFh	61h

Hardware Conditions

It is possible to force the controller to execute the bootloader after a Reset with hardware conditions. Depending on the product type (low pin count or high pin count package), there are two methods to apply the hardware conditions.

High Pin Count Hardware Conditions (PLCC52, QFP64)

For high pin count packages, the hardware conditons (EA = 1, PSEN = 0) are sampled during the RESET rising edge to force the on-chip bootloader execution (See Figure 79 on page 173). In this way the bootloader can be carried out regardless of the user Flash memory content.

As PSEN is an output port in normal operating mode (running user application or bootloader code) after reset, it is recommended to release PSEN after rising edge of reset signal.

Low Pin Count Hardware Conditions (QFN32)

Low pin count products do not have PSEN signal, thus for these products, the bootloader is always executed after reset thanks to the BLJB bit. The Hardware Condition are detected at the begining of the bootloader execution from reset.

The default factory Hardware Condition is assigned to port P1.

- P1 must be equal to FEh

In order to offer the best flexibility, the user can define its own Hardware Condition on one of the following Ports:

- Port1
- Port3
- Port4 (only bit0 and bit1)

The Hardware Condition configuration are stored in three bytes called P1_CF, P3_CF, P4_CF.

These bytes can be modified by the user through a set of API or through an ISP command.

Note: The BLJB must be at 0 (programmed) to be able to restart the bootloader.
If the BLJB is equal to 1 (unprogrammed) only the hardware parallel programmer can change this bit.

See a detailed description in the applicable Document.

– Datasheet Bootloader USB AT89C5131.

On-chip Expanded RAM (ERAM)

The AT89C5130A/31A-M provides additional Bytes of random access memory (RAM) space for increased data parameters handling and high level language usage.

AT89C5130A/31A-M devices have expanded RAM in external data space; maximum size and location are described in Table 46.

Table 46. Description of Expanded RAM

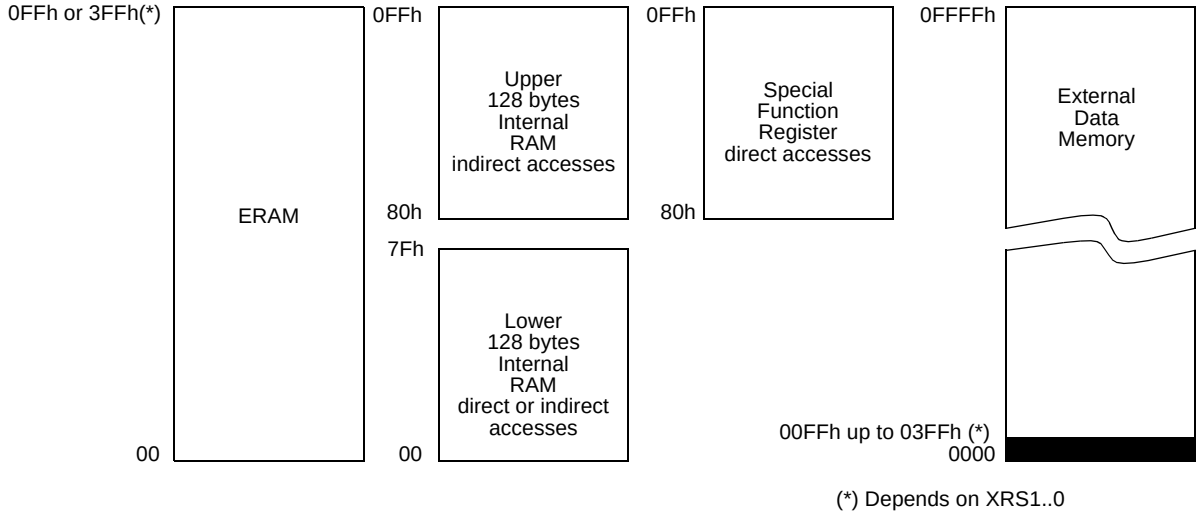
Part Number	ERAM Size	Address	
		Start	End
AT89C5130A/31A-M	1024	00h	3FFh

The AT89C5130A/31A-M has on-chip data memory which is mapped into the following four separate segments.

1. The Lower 128 bytes of RAM (addresses 00h to 7Fh) are directly and indirectly addressable.
2. The Upper 128 bytes of RAM (addresses 80h to FFh) are indirectly addressable only.
3. The Special Function Registers, SFRs, (addresses 80h to FFh) are directly addressable only.
4. The expanded RAM bytes are indirectly accessed by MOVX instructions, and with the EXTRAM bit cleared in the AUXR register (see Table 46)

The lower 128 bytes can be accessed by either direct or indirect addressing. The Upper 128 bytes can be accessed by indirect addressing only. The Upper 128 bytes occupy the same address space as the SFR. That means they have the same address, but are physically separate from SFR space.

Figure 22. Internal and External Data Memory Address



When an instruction accesses an internal location above address 7Fh, the CPU knows whether the access is to the upper 128 bytes of data RAM or to SFR space by the addressing mode used in the instruction.

- Instructions that use direct addressing access SFR space. For example: MOV 0A0H, # data, accesses the SFR at location 0A0h (which is P2).
- Instructions that use indirect addressing access the Upper 128 bytes of data RAM. For example: MOV atR0, # data where R0 contains 0A0h, accesses the data byte at address 0A0h, rather than P2 (whose address is 0A0h).
- The ERAM bytes can be accessed by indirect addressing, with EXTRAM bit cleared and MOVX instructions. This part of memory which is physically located on-chip, logically occupies the first bytes of external data memory. The bits XRS0 and XRS1 are used to hide a part of the available ERAM as explained in Table 46. This can be useful if external peripherals are mapped at addresses already used by the internal ERAM.
- With EXTRAM = 0, the ERAM is indirectly addressed, using the MOVX instruction in combination with any of the registers R0, R1 of the selected bank or DPTR. An access to ERAM will not affect ports P0, P2, P3.6 (WR) and P3.7 (RD). For example, with EXTRAM = 0, MOVX atR0, # data where R0 contains 0A0H, accesses the ERAM at address 0A0H rather than external memory. An access to external data memory locations higher than the accessible size of the ERAM will be performed with the MOVX DPTR instructions in the same way as in the standard 80C51, with P0 and P2 as data/address busses, and P3.6 and P3.7 as write and read timing signals. Accesses to ERAM above 0FFH can only be done by the use of DPTR.
- With EXTRAM = 1, MOVX @Ri and MOVX @DPTR will be similar to the standard 80C51. MOVX at Ri will provide an eight-bit address multiplexed with data on Port0 and any output port pins can be used to output higher order address bits. This is to provide the external paging capability. MOVX @DPTR will generate a sixteen-bit address. Port2 outputs the high-order eight address bits (the contents of DPH) while Port0 multiplexes the low-order eight address bits (DPL) with data. MOVX at Ri and MOVX @DPTR will generate either read or write signals on P3.6 ($\overline{WR}$) and P3.7 ($\overline{RD}$).

The stack pointer (SP) may be located anywhere in the 256 bytes RAM (lower and upper RAM) internal data memory. The stack may not be located in the ERAM.

The M0 bit allows to stretch the ERAM timings; if M0 is set, the read and write pulses are extended from 6 to 30 clock periods. This is useful to access external slow peripherals.

Table 47. AUXR Register

AUXR - Auxiliary Register (8Eh)

7	6	5	4	3	2	1	0
DPU	**-**	**M0**	**-**	**XRS1**	**XRS0**	**EXTRAM**	**AO**

Bit Number	Bit Mnemonic	Description
7	DPU	**Disable Weak Pull Up** Cleared to enabled weak pull up on standard Ports. Set to disable weak pull up on standard Ports.
6	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit
5	M0	**Pulse length** Cleared to stretch MOVX control: the $\overline{RD}$ and the $\overline{WR}$ pulse length is 6 clock periods (default). Set to stretch MOVX control: the $\overline{RD}$ and the $\overline{WR}$ pulse length is 30 clock periods.
4	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit
3	XRS1	**ERAM Size** **XRS1 XRS0 ERAM size** 0 0 256 bytes 0 1 512 bytes 1 0 768 bytes 1 1 1024 bytes (default)
2	XRS0	
1	EXTRAM	**EXTRAM bit** Cleared to access internal ERAM using MOVX at $\overline{Ri}$ at DPTR. Set to access external memory.
0	AO	**ALE Output bit** Cleared, ALE is emitted at a constant rate of 1/6 the oscillator frequency (or 1/3 if X2 mode is used) (default). Set, ALE is active only when a MOVX or MOVC instruction is used.

Reset Value = 0X0X 1100b

Not bit addressable

Timer 2

The Timer 2 in the AT89C5130A/31A-M is the standard C52 Timer 2. It is a 16-bit timer/counter: the count is maintained by two cascaded eight-bit timer registers, TH2 and TL2. It is controlled by T2CON (Table 48) and T2MOD (Table 49) registers. Timer 2 operation is similar to Timer 0 and Timer 1. C/$\overline{T2}$ selects F_{OSC}/12 (timer operation) or external pin T2 (counter operation) as the timer clock input. Setting TR2 allows TL2 to be incremented by the selected input.

Timer 2 has 3 operating modes: capture, auto reload and Baud Rate Generator. These modes are selected by the combination of RCLK, TCLK and CP/$\overline{RL2}$ (T2CON).

Refer to the Atmel 8-bit microcontroller hardware documentation for the description of Capture and Baud Rate Generator Modes.

Timer 2 includes the following enhancements:

- Auto-reload mode with up or down counter
- Programmable Clock-output

Auto-reload Mode

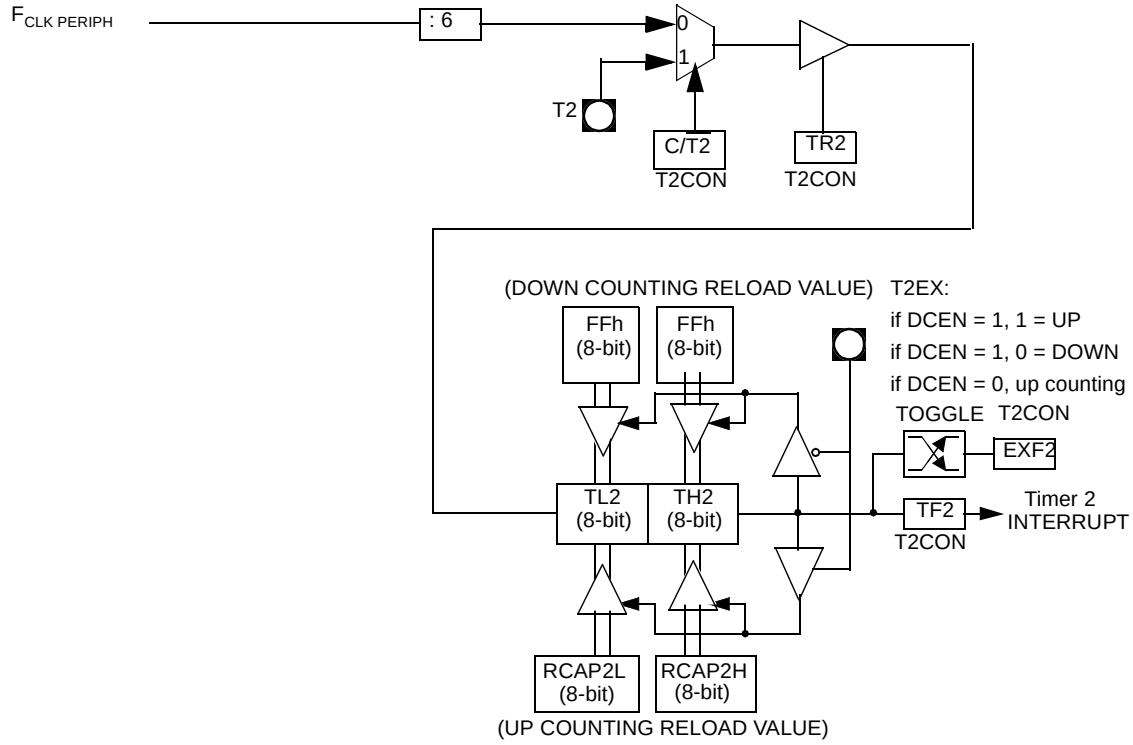
The Auto-reload mode configures Timer 2 as a 16-bit timer or event counter with automatic reload. If DCEN bit in T2MOD is cleared, Timer 2 behaves as in 80C52 (refer to the Atmel 8-bit microcontroller hardware description). If DCEN bit is set, Timer 2 acts as an Up/down timer/counter as shown in Figure 23. In this mode the T2EX pin controls the direction of count.

When T2EX is high, Timer 2 counts up. Timer overflow occurs at FFFFh which sets the TF2 flag and generates an interrupt request. The overflow also causes the 16-bit value in RCAP2H and RCAP2L registers to be loaded into the timer registers TH2 and TL2.

When T2EX is low, Timer 2 counts down. Timer underflow occurs when the count in the timer registers TH2 and TL2 equals the value stored in RCAP2H and RCAP2L registers. The underflow sets TF2 flag and reloads FFFFh into the timer registers.

The EXF2 bit toggles when Timer 2 overflows or underflows according to the direction of the count. EXF2 does not generate any interrupt. This bit can be used to provide 17-bit resolution.

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/$\overline{T2}$ 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.

It is possible to use Timer 2 as a baud rate generator and a clock generator simultaneously. For this configuration, the baud rates and clock frequencies are not independent since both functions use the values in the RCAP2H and RCAP2L registers.

Figure 24. Clock-out Mode C/$\overline{T2}$ = 0

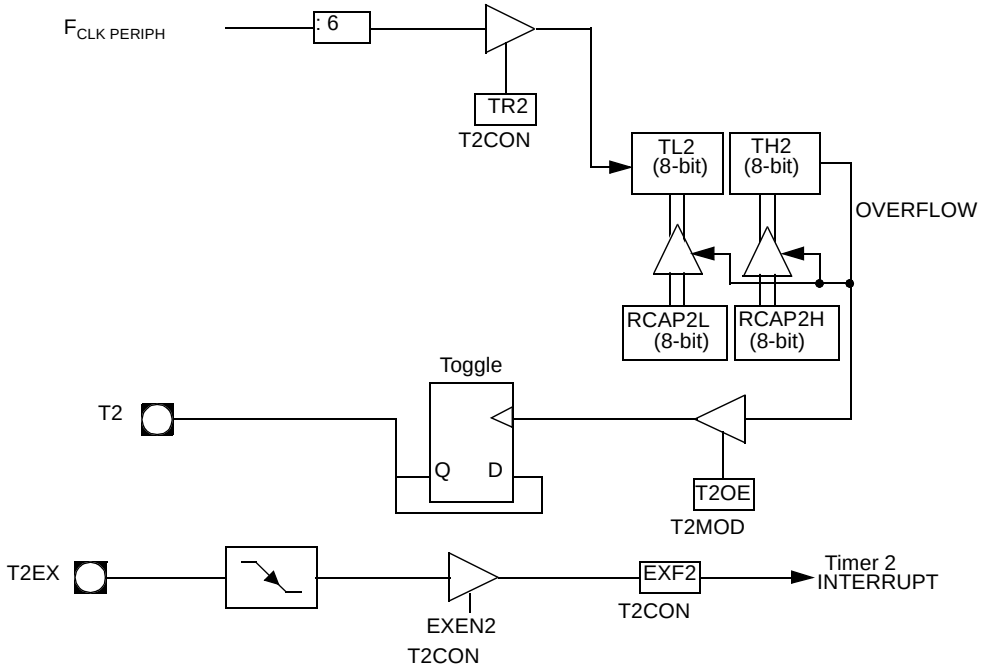


Table 48. T2CON Register

T2CON - Timer 2 Control Register (C8h)

7	6	5	4	3	2	1	0
TF2	**EXF2**	**RCLK**	**TCLK**	**EXEN2**	**TR2**	**C/T2#**	**CP/RL2#**

Bit Number	Bit Mnemonic	Description
7	TF2	**Timer 2 overflow Flag** Must be cleared by software. Set by hardware on Timer 2 overflow, if RCLK = 0 and TCLK = 0.
6	EXF2	**Timer 2 External Flag** Set when a capture or a reload is caused by a negative transition on T2EX pin if EXEN2 = 1. When set, causes the CPU to vector to Timer 2 interrupt routine when Timer 2 interrupt is enabled. Must be cleared by software. EXF2 doesn't cause an interrupt in Up/down counter mode (DCEN = 1).
5	RCLK	**Receive Clock bit** Cleared to use Timer 1 overflow as receive clock for serial port in mode 1 or 3. Set to use Timer 2 overflow as receive clock for serial port in mode 1 or 3.
4	TCLK	**Transmit Clock bit** Cleared to use Timer 1 overflow as transmit clock for serial port in mode 1 or 3. Set to use Timer 2 overflow as transmit clock for serial port in mode 1 or 3.
3	EXEN2	**Timer 2 External Enable bit** Cleared to ignore events on T2EX pin for Timer 2 operation. Set to cause a capture or reload when a negative transition on T2EX pin is detected, if Timer 2 is not used to clock the serial port.
2	TR2	**Timer 2 Run control bit** Cleared to turn off Timer 2. Set to turn on Timer 2.
1	C/T2#	**Timer/Counter 2 select bit** Cleared for timer operation (input from internal clock system: $F_{CLK\ PERIPH}$). Set for counter operation (input from T2 input pin, falling edge trigger). Must be 0 for clock out mode.
0	CP/RL2#	**Timer 2 Capture/Reload bit** If RCLK = 1 or TCLK = 1, CP/RL2# is ignored and timer is forced to Auto-reload on Timer 2 overflow. Cleared to Auto-reload on Timer 2 overflows or negative transitions on T2EX pin if EXEN2 = 1. Set to capture on negative transitions on T2EX pin if EXEN2 = 1.

Reset Value = 0000 0000b

Bit addressable

Table 49. T2MOD Register

T2MOD - Timer 2 Mode Control Register (C9h)

7	6	5	4	3	2	1	0
-	-	-	-	-	-	T2OE	DCEN

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	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
4	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
3	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
2	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
1	T2OE	**Timer 2 Output Enable bit** Cleared to program P1.0/T2 as clock input or I/O port. Set to program P1.0/T2 as clock output.
0	DCEN	**Down Counter Enable bit** Cleared to disable Timer 2 as up/down counter. Set to enable Timer 2 as up/down counter.

Reset Value = XXXX XX00b

Not bit addressable

Programmable Counter Array (PCA)

The PCA provides more timing capabilities with less CPU intervention than the standard timer/counters. Its advantages include reduced software overhead and improved accuracy. The PCA consists of a dedicated timer/counter which serves as the time base for an array of five compare/capture modules. Its clock input can be programmed to count any one of the following signals:

- Peripheral clock frequency ($F_{CLK\ PERIPH}$) ÷ 6
- Peripheral clock frequency ($F_{CLK\ PERIPH}$) ÷ 2
- Timer 0 overflow
- External input on ECI (P1.2)

Each compare/capture modules can be programmed in any one of the following modes:

- rising and/or falling edge capture,
- software timer
- high-speed output, or
- pulse width modulator

Module 4 can also be programmed as a watchdog timer (see Section "PCA Watchdog Timer", page 66).

When the compare/capture modules are programmed in the capture mode, software timer, or high speed output mode, an interrupt can be generated when the module executes its function. All five modules plus the PCA timer overflow share one interrupt vector.

The PCA timer/counter and compare/capture modules share Port 1 for external I/O. These pins are listed below. If the port pin is not used for the PCA, it can still be used for standard I/O.

PCA Component	External I/O Pin
16-bit Counter	P1.2/ECI
16-bit Module 0	P1.3/CEX0
16-bit Module 1	P1.4/CEX1
16-bit Module 2	P1.5/CEX2
16-bit Module 3	P1.6/CEX3
16-bit Module 4	P1.7/CEX4

The PCA timer is a common time base for all five modules (see Figure 25). The timer count source is determined from the CPS1 and CPS0 bits in the CMOD register (Table 50) and can be programmed to run at:

- 1/6 the peripheral clock frequency ($F_{CLK\ PERIPH}$).
- 1/2 the peripheral clock frequency ($F_{CLK\ PERIPH}$).
- The Timer 0 overflow
- The input on the ECI pin (P1.2)

Figure 25. PCA Timer/Counter

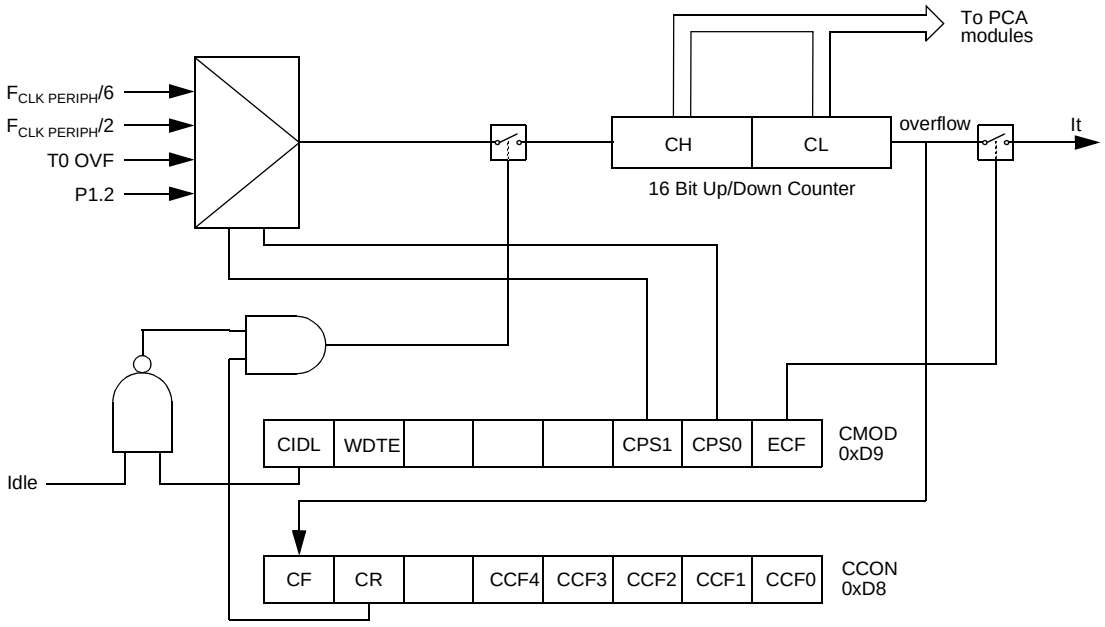


Table 50. CMOD Register

CMOD - PCA Counter Mode Register (D9h)

7	6	5	4	3	2	1	0
CIDL	**WDTE**	**-**	**-**	**-**	**CPS1**	**CPS0**	**ECF**

Bit Number	Bit Mnemonic	Description
7	CIDL	**Counter Idle Control** Cleared to program the PCA Counter to continue functioning during idle Mode. Set to program PCA to be gated off during idle.
6	WDTE	**Watchdog Timer Enable** Cleared to disable Watchdog Timer function on PCA Module 4. Set to enable Watchdog Timer function on PCA Module 4.
5	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
4	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
3	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
2	CPS1	**PCA Count Pulse Select** CPS1 CPS0 Selected PCA input 0 0 Internal clock $f_{CLK\ PERIPH}/6$ 0 1 Internal clock $f_{CLK\ PERIPH}/2$ 1 0 Timer 0 Overflow 1 1 External clock at ECI/P1.2 pin (max rate = $f_{CLK\ PERIPH}/4$)
1	CPS0	
0	ECF	**PCA Enable Counter Overflow Interrupt** Cleared to disable CF bit in CCON to inhibit an interrupt. Set to enable CF bit in CCON to generate an interrupt.

Reset Value = 00XX X000b

Not bit addressable

The CMOD register includes three additional bits associated with the PCA (See Figure 25 and Table 50).

- The CIDL bit allows the PCA to stop during idle mode.
- The WDTE bit enables or disables the watchdog function on module 4.
- The ECF bit when set causes an interrupt and the PCA overflow flag CF (in the CCON SFR) to be set when the PCA timer overflows.

The CCON register contains the run control bit for the PCA and the flags for the PCA timer (CF) and each module (see Table 51).

- Bit CR (CCON.6) must be set by software to run the PCA. The PCA is shut off by clearing this bit.
- Bit CF: The CF bit (CCON.7) is set when the PCA counter overflows and an interrupt will be generated if the ECF bit in the CMOD register is set. The CF bit can only be cleared by software.
- Bits 0 through 4 are the flags for the modules (bit 0 for module 0, bit 1 for module 1, etc.) and are set by hardware when either a match or a capture occurs. These flags can only be cleared by software.

Table 51. CCON Register

CCON - PCA Counter Control Register (D8h)

7	6	5	4	3	2	1	0
CF	**CR**	–	**CCF4**	**CCF3**	**CCF2**	**CCF1**	**CCF0**

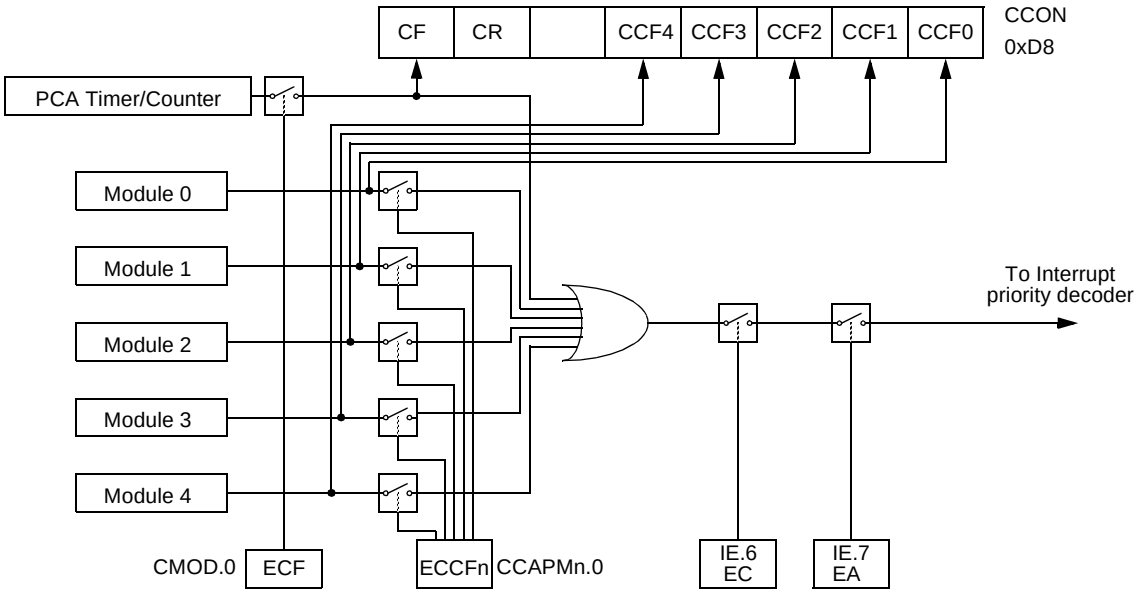
Bit Number	Bit Mnemonic	Description
7	CF	**PCA Counter Overflow flag** Set by hardware when the counter rolls over. CF flags an interrupt if bit ECF in CMOD is set. CF may be set by either hardware or software but can only be cleared by software.
6	CR	**PCA Counter Run control bit** Must be cleared by software to turn the PCA counter off. Set by software to turn the PCA counter on.
5	–	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
4	CCF4	**PCA Module 4 interrupt flag** Must be cleared by software. Set by hardware when a match or capture occurs.
3	CCF3	**PCA Module 3 interrupt flag** Must be cleared by software. Set by hardware when a match or capture occurs.
2	CCF2	**PCA Module 2 interrupt flag** Must be cleared by software. Set by hardware when a match or capture occurs.
1	CCF1	**PCA Module 1 Interrupt Flag** Must be cleared by software. Set by hardware when a match or capture occurs.
0	CCF0	**PCA Module 0 Interrupt Flag** Must be cleared by software. Set by hardware when a match or capture occurs.

Reset Value = 000X 0000b
Not bit addressable

The watchdog timer function is implemented in module 4 (See Figure 28).

The PCA interrupt system is shown in Figure 26.

Figure 26. PCA Interrupt System



PCA Modules: each one of the five compare/capture modules has six possible functions. It can perform:

- 16-bit capture, positive-edge triggered
- 16-bit capture, negative-edge triggered
- 16-bit capture, both positive and negative-edge triggered
- 16-bit Software Timer
- 16-bit High-speed Output
- 8-bit Pulse Width Modulator

In addition, module 4 can be used as a Watchdog Timer.

Each module in the PCA has a special function register associated with it. These registers are: CCAPM0 for module 0, CCAPM1 for module 1, etc. (see Table 52). The registers contain the bits that control the mode that each module will operate in.

- The ECCF bit (CCAPMn.0 where n = 0, 1, 2, 3, or 4 depending on the module) enables the CCF flag in the CCON SFR to generate an interrupt when a match or compare occurs in the associated module.
- PWM (CCAPMn.1) enables the pulse width modulation mode.
- The TOG bit (CCAPMn.2) when set causes the CEX output associated with the module to toggle when there is a match between the PCA counter and the module's capture/compare register.
- The match bit MAT (CCAPMn.3) when set will cause the CCFn bit in the CCON register to be set when there is a match between the PCA counter and the module's capture/compare register.
- The next two bits CAPN (CCAPMn.4) and CAPP (CCAPMn.5) determine the edge that a capture input will be active on. The CAPN bit enables the negative edge, and

the CAPP bit enables the positive edge. If both bits are set both edges will be enabled and a capture will occur for either transition.

- The last bit in the register ECOM (CCAPMn.6) when set enables the comparator function.

Table 53 shows the CCAPMn settings for the various PCA functions.

Table 52. CCAPMn Registers (n = 0-4)

CCAPM0 - PCA Module 0 Compare/Capture Control Register (0DAh)
CCAPM1 - PCA Module 1 Compare/Capture Control Register (0DBh)
CCAPM2 - PCA Module 2 Compare/Capture Control Register (0DCh)
CCAPM3 - PCA Module 3 Compare/Capture Control Register (0DDh)
CCAPM4 - PCA Module 4 Compare/Capture Control Register (0DEh)

7	6	5	4	3	2	1	0
-	ECOMn	CAPPn	CAPNn	MATn	TOGn	PWMn	ECCFn

Bit Number	Bit Mnemonic	Description
7	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
6	ECOMn	**Enable Comparator** Cleared to disable the comparator function. Set to enable the comparator function.
5	CAPPn	**Capture Positive** Cleared to disable positive edge capture. Set to enable positive edge capture.
4	CAPNn	**Capture Negative** Cleared to disable negative edge capture. Set to enable negative edge capture.
3	MATn	**Match** When MATn = 1, a match of the PCA counter with this module's compare/capture register causes the CCFn bit in CCON to be set, flagging an interrupt.
2	TOGn	**Toggle** When TOGn = 1, a match of the PCA counter with this module's compare/capture register causes the CEXn pin to toggle.
1	PWMn	**Pulse Width Modulation Mode** Cleared to disable the CEXn pin to be used as a pulse width modulated output. Set to enable the CEXn pin to be used as a pulse width modulated output.
0	ECCFn	**Enable CCF Interrupt** Cleared to disable compare/capture flag CCFn in the CCON register to generate an interrupt. Set to enable compare/capture flag CCFn in the CCON register to generate an interrupt.

Reset Value = X000 0000b
Not bit addressable

Table 53. PCA Module Modes (CCAPMn Registers)

ECOMn	CAPPn	CAPNn	MATn	TOGn	PWMm	ECCFn	Module Function
0	0	0	0	0	0	0	No Operation
X	1	0	0	0	0	X	16-bit capture by a positive-edge trigger on CEXn
X	0	1	0	0	0	X	16-bit capture by a negative trigger on CEXn
X	1	1	0	0	0	X	16-bit capture by a transition on CEXn
1	0	0	1	0	0	X	16-bit Software Timer/Compare mode.
1	0	0	1	1	0	X	16-bit High Speed Output
1	0	0	0	0	1	0	8-bit PWM
1	0	0	1	X	0	X	Watchdog Timer (module 4 only)

There are two additional registers associated with each of the PCA modules. They are CCAPnH and CCAPnL and these are the registers that store the 16-bit count when a capture occurs or a compare should occur. When a module is used in the PWM mode these registers are used to control the duty cycle of the output (see Table 54 and Table 55)

Table 54. CCAPnH Registers (n = 0-4)

CCAP0H - PCA Module 0 Compare/Capture Control Register High (0FAh)
CCAP1H - PCA Module 1 Compare/Capture Control Register High (0FBh)
CCAP2H - PCA Module 2 Compare/Capture Control Register High (0FCh)
CCAP3H - PCA Module 3 Compare/Capture Control Register High (0FDh)
CCAP4H - PCA Module 4 Compare/Capture Control Register High (0FEh)

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

Bit Number	Bit Mnemonic	Description
7 - 0	-	**PCA Module n Compare/Capture Control** CCAPnH Value

Reset Value = XXXX XXXXb
Not bit addressable

Table 55. CCAPnL Registers (n = 0-4)

CCAP0L - PCA Module 0 Compare/Capture Control Register Low (0EAh)
CCAP1L - PCA Module 1 Compare/Capture Control Register Low (0EBh)
CCAP2L - PCA Module 2 Compare/Capture Control Register Low (0ECh)
CCAP3L - PCA Module 3 Compare/Capture Control Register Low (0EDh)
CCAP4L - PCA Module 4 Compare/Capture Control Register Low (0EEh)

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

Bit Number	Bit Mnemonic	Description
7 - 0	-	**PCA Module n Compare/Capture Control** CCAPnL Value

Reset Value = XXXX XXXXb
Not bit addressable

Table 56. CH Register

CH - PCA Counter Register High (0F9h)

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

Bit Number	Bit Mnemonic	Description
7 - 0	-	**PCA counter** CH Value

Reset Value = 0000 0000b
Not bit addressable

Table 57. CL Register

CL - PCA Counter Register Low (0E9h)

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

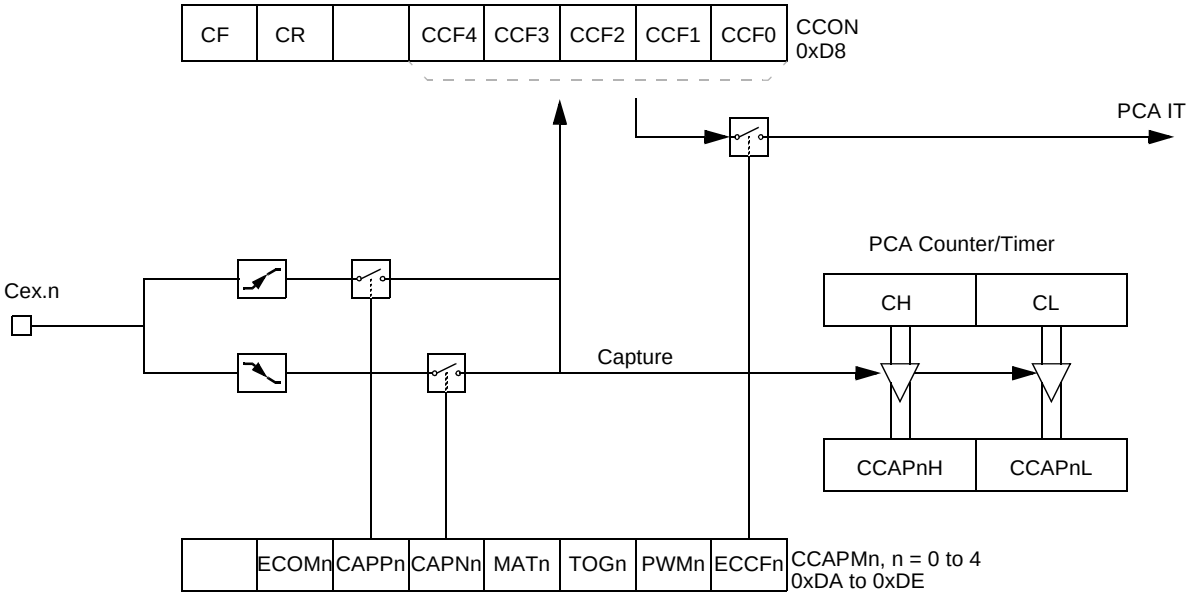
Bit Number	Bit Mnemonic	Description
7 - 0	-	**PCA Counter** CL Value

Reset Value = 0000 0000b
Not bit addressable

PCA Capture Mode

To use one of the PCA modules in the capture mode either one or both of the CCAPM bits CAPN and CAPP for that module must be set. The external CEX input for the module (on port 1) is sampled for a transition. When a valid transition occurs the PCA hardware loads the value of the PCA counter registers (CH and CL) into the module's capture registers (CCAPnL and CCAPnH). If the CCFn bit for the module in the CCON SFR and the ECCFn bit in the CCAPMn SFR are set then an interrupt will be generated (see Figure 27).

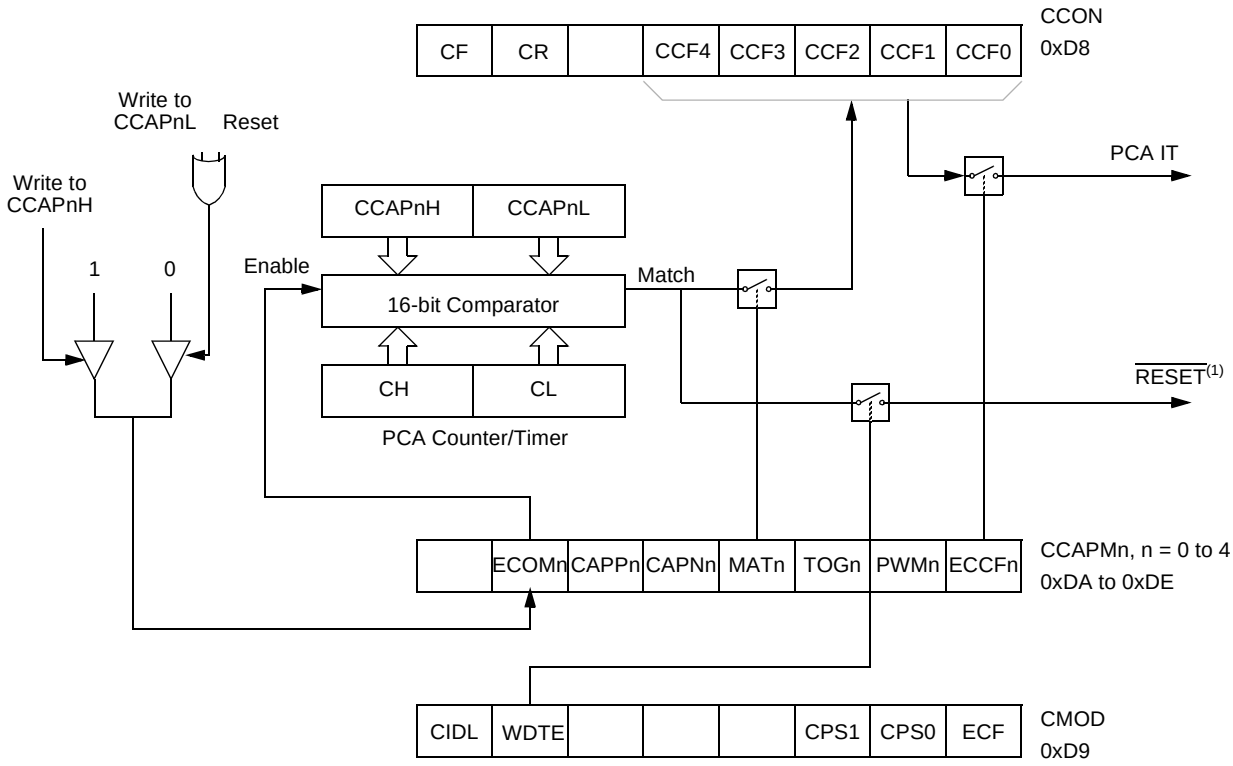
Figure 27. PCA Capture Mode



16-bit Software Timer/Compare Mode

The PCA modules can be used as software timers by setting both the ECOM and MAT bits in the modules CCAPMn register. The PCA timer will be compared to the module's capture registers and when a match occurs an interrupt will occur if the CCFn (CCON SFR) and the ECCFn (CCAPMn SFR) bits for the module are both set (see Figure 28).

Figure 28. PCA Compare Mode and PCA Watchdog Timer



Note: 1. Only for Module 4

Before enabling ECOM bit, CCAPnL and CCAPnH should be set with a non zero value, otherwise an unwanted match could happen. Writing to CCAPnH will set the ECOM bit.

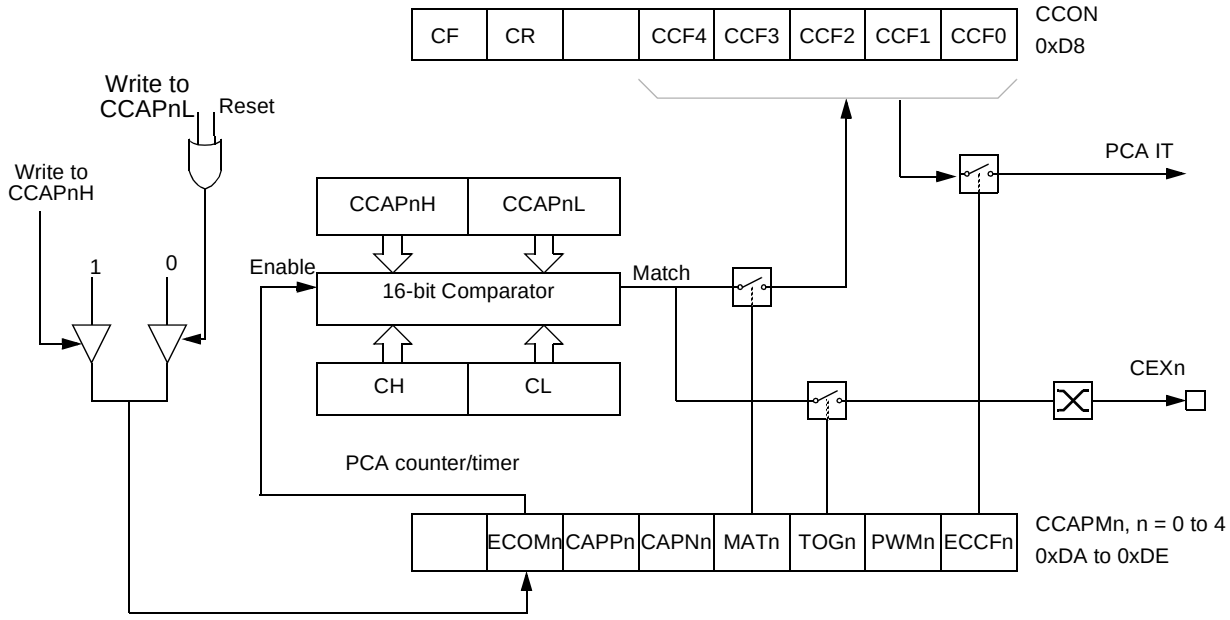
Once ECOM set, writing CCAPnL will clear ECOM so that an unwanted match doesn't occur while modifying the compare value. Writing to CCAPnH will set ECOM. For this reason, user software should write CCAPnL first, and then CCAPnH. Of course, the ECOM bit can still be controlled by accessing to CCAPMn register.

High Speed Output Mode

In this mode, the CEX output (on port 1) associated with the PCA module will toggle each time a match occurs between the PCA counter and the module's capture registers. To activate this mode the TOG, MAT, and ECOM bits in the module's CCAPMn SFR must be set (see Figure 29).

A prior write must be done to CCAPnL and CCAPnH before writing the ECOMn bit.

Figure 29. PCA High-speed Output Mode



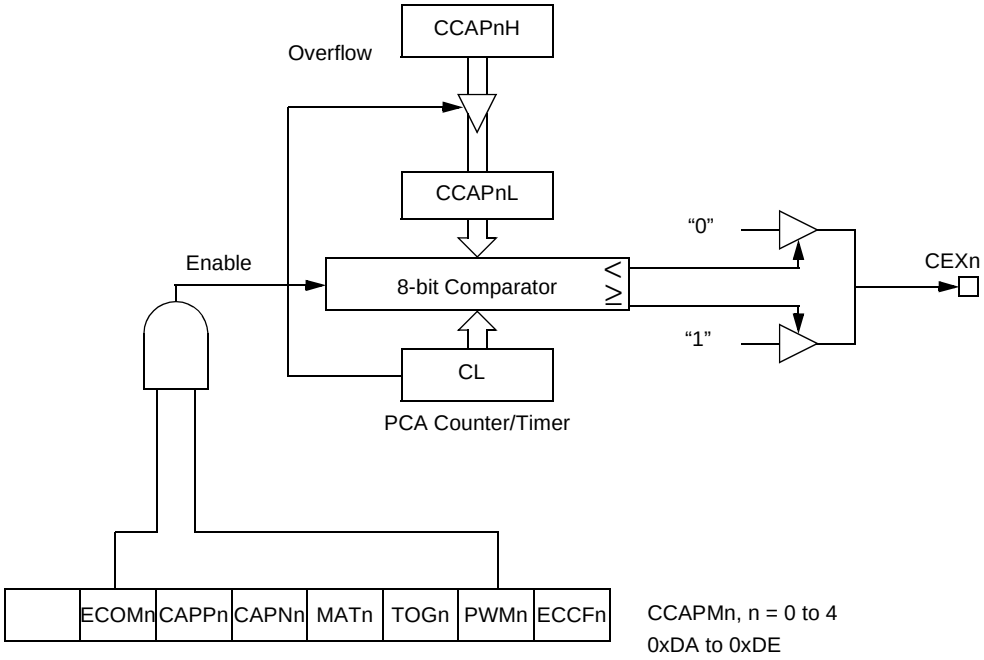
Before enabling ECOM bit, CCAPnL and CCAPnH should be set with a non zero value, otherwise an unwanted match could happen.

Once ECOM set, writing CCAPnL will clear ECOM so that an unwanted match doesn't occur while modifying the compare value. Writing to CCAPnH will set ECOM. For this reason, user software should write CCAPnL first, and then CCAPnH. Of course, the ECOM bit can still be controlled by accessing to CCAPMn register.

Pulse Width Modulator Mode

All of the PCA modules can be used as PWM outputs. Figure 30 shows the PWM function. The frequency of the output depends on the source for the PCA timer. All of the modules will have the same frequency of output because they all share the PCA timer. The duty cycle of each module is independently variable using the module's capture register CCAPLn. When the value of the PCA CL SFR is less than the value in the module's CCAPLn SFR the output will be low, when it is equal to or greater than the output will be high. When CL overflows from FF to 00, CCAPLn is reloaded with the value in CCAPHn. This allows updating the PWM without glitches. The PWM and ECOM bits in the module's CCAPMn register must be set to enable the PWM mode.

Figure 30. PCA PWM Mode



PCA Watchdog Timer

An on-board watchdog timer is available with the PCA to improve the reliability of the system without increasing chip count. Watchdog timers are useful for systems that are susceptible to noise, power glitches, or electrostatic discharge. Module 4 is the only PCA module that can be programmed as a watchdog. However, this module can still be used for other modes if the watchdog is not needed. Figure 28 shows a diagram of how the watchdog works. The user pre-loads a 16-bit value in the compare registers. Just like the other compare modes, this 16-bit value is compared to the PCA timer value. If a match is allowed to occur, an internal reset will be generated. This will not cause the RST pin to be driven low.

In order to hold off the reset, the user has three options:

1. Periodically change the compare value so it will never match the PCA timer
2. Periodically change the PCA timer value so it will never match the compare values, or
3. Disable the watchdog by clearing the WDTE bit before a match occurs and then re-enable it

The first two options are more reliable because the watchdog timer is never disabled as in option #3. If the program counter ever goes astray, a match will eventually occur and cause an internal reset. The second option is also not recommended if other PCA modules are being used. Remember, the PCA timer is the time base for all modules; changing the time base for other modules would not be a good idea. Thus, in most applications the first solution is the best option.

This watchdog timer won't generate a reset out on the reset pin.

Serial I/O Port

The serial I/O port in the AT89C5130A/31A-M is compatible with the serial I/O port in the 80C52.
It provides both synchronous and asynchronous communication modes. It operates as an Universal Asynchronous Receiver and Transmitter (UART) in three full-duplex modes (modes 1, 2 and 3). Asynchronous transmission and reception can occur simultaneously and at different baud rates.

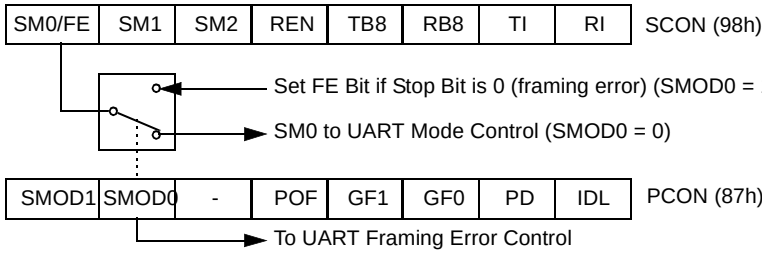
Serial I/O port includes the following enhancements:

- Framing error detection
- Automatic address recognition

Framing Error Detection

Framing bit error detection is provided for the three asynchronous modes (modes 1, 2 and 3). To enable the framing bit error detection feature, set SMOD0 bit in PCON register (see Figure 31).

Figure 31. Framing Error Block Diagram



When this feature is enabled, the receiver checks each incoming data frame for a valid stop bit. An invalid stop bit may result from noise on the serial lines or from simultaneous transmission by two CPUs. If a valid stop bit is not found, the Framing Error bit (FE) in SCON register (See Table 58) bit is set.
Software may examine FE bit after each reception to check for data errors. Once set, only software or a reset can clear FE bit. Subsequently received frames with valid stop bits cannot clear FE bit. When FE feature is enabled, RI rises on stop bit instead of the last data bit (See Figure 32 and Figure 33).

Figure 32. UART Timings in Mode 1

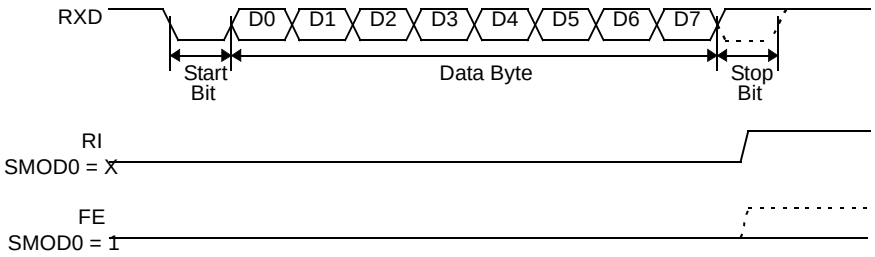
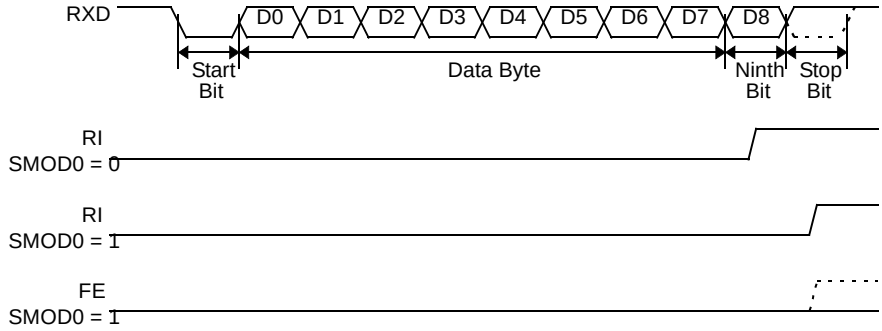


Figure 33. UART Timings in Modes 2 and 3



Automatic Address Recognition

The automatic address recognition feature is enabled when the multiprocessor communication feature is enabled (SM2 bit in SCON register is set).

Implemented in hardware, automatic address recognition enhances the multiprocessor communication feature by allowing the serial port to examine the address of each incoming command frame. Only when the serial port recognizes its own address, the receiver sets RI bit in SCON register to generate an interrupt. This ensures that the CPU is not interrupted by command frames addressed to other devices.

If desired, you may enable the automatic address recognition feature in mode 1. In this configuration, the stop bit takes the place of the ninth data bit. Bit RI is set only when the received command frame address matches the device's address and is terminated by a valid stop bit.

To support automatic address recognition, a device is identified by a given address and a broadcast address.

Note: The multiprocessor communication and automatic address recognition features cannot be enabled in mode 0 (i.e., setting SM2 bit in SCON register in mode 0 has no effect).

Given Address

Each device has an individual address that is specified in SADDR register; the SADEN register is a mask byte that contains don't care bits (defined by zeros) to form the device's given address. The don't care bits provide the flexibility to address one or more slaves at a time. The following example illustrates how a given address is formed.

To address a device by its individual address, the SADEN mask byte must be `1111 1111b`.

For example:

SADDR0101 0110b
SADEN1111 1100b
Given0101 01XXb

The following is an example of how to use given addresses to address different slaves:

Slave A:SADDR1111 0001b
SADEN1111 1010b
Given1111 0X0Xb

Slave B:SADDR1111 0011b
SADEN1111 1001b
Given1111 0XX1b

Slave C:SADDR1111 0011b
SADEN1111 1101b
Given1111 00X1b

The SADEN byte is selected so that each slave may be addressed separately.

For slave A, bit 0 (the LSB) is a don't care bit; for slaves B and C, bit 0 is a 1. To communicate with slave A only, the master must send an address where bit 0 is clear (e.g. 1111 0000b).

For slave A, bit 1 is a 1; for slaves B and C, bit 1 is a don't care bit. To communicate with slaves B and C, but not slave A, the master must send an address with bits 0 and 1 both set (e.g. 1111 0011b).

To communicate with slaves A, B and C, the master must send an address with bit 0 set, bit 1 clear, and bit 2 clear (e.g. 1111 0001b).

Broadcast Address

A broadcast address is formed from the logical OR of the SADDR and SADEN registers with zeros defined as don't care bits, e.g.:

SADDR0101 0110b
SADEN1111 1100b

Broadcast = SADDR OR SADEN1111 111Xb

The use of don't care bits provides flexibility in defining the broadcast address, in most applications, a broadcast address is FFh. The following is an example of using broadcast addresses:

Slave A:SADDR1111 0001b
SADEN1111 1010b
Broadcast1111 1X11b,

Slave B:SADDR1111 0011b
SADEN1111 1001b
Broadcast1111 1X11B,

Slave C:SADDR = 1111 0010b
SADEN1111 1101b
Broadcast1111 1111b

For slaves A and B, bit 2 is a don't care bit; for slave C, bit 2 is set. To communicate with all of the slaves, the master must send an address FFh. To communicate with slaves A and B, but not slave C, the master can send and address FBh.

Reset Addresses

On reset, the SADDR and SADEN registers are initialized to 00h, i.e. the given and broadcast addresses are XXXX XXXXb (all don't care bits). This ensures that the serial port will reply to any address, and so, that it is backwards compatible with the 80C51 microcontrollers that do not support automatic address recognition.

SADEN - Slave Address Mask Register (B9h)

7	6	5	4	3	2	1	0

Reset Value = 0000 0000b
Not bit addressable

SADDR - Slave Address Register (A9h)

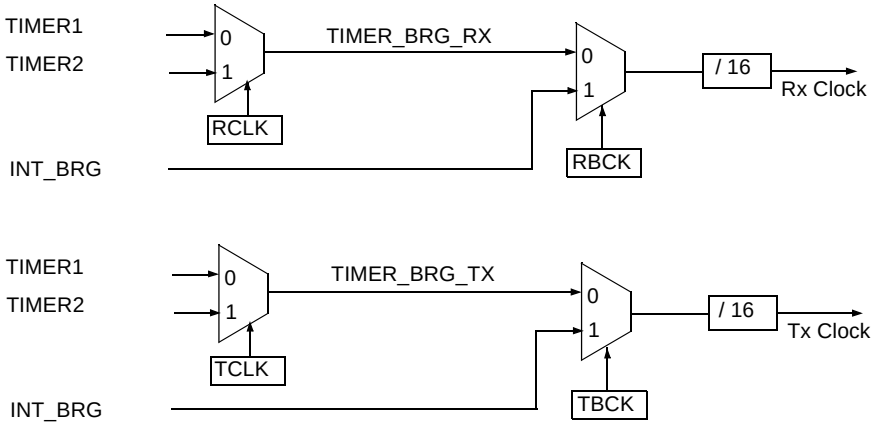
7	6	5	4	3	2	1	0

Reset Value = 0000 0000b
Not bit addressable

Baud Rate Selection for UART for Mode 1 and 3

The Baud Rate Generator for transmit and receive clocks can be selected separately via the T2CON and BDRCON registers.

Figure 34. Baud Rate Selection



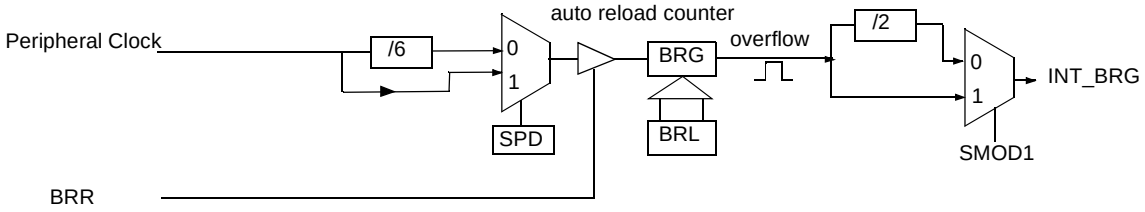
Baud Rate Selection Table for UART

TCLK (T2CON)	RCLK (T2CON)	TBCK (BDRCON)	RBCK (BDRCON)	Clock Source UART Tx	Clock Source UART Rx
0	0	0	0	Timer 1	Timer 1
1	0	0	0	Timer 2	Timer 1
0	1	0	0	Timer 1	Timer 2
1	1	0	0	Timer 2	Timer 2
X	0	1	0	INT_BRG	Timer 1
X	1	1	0	INT_BRG	Timer 2
0	X	0	1	Timer 1	INT_BRG
1	X	0	1	Timer 2	INT_BRG
X	X	1	1	INT_BRG	INT_BRG

Internal Baud Rate Generator (BRG)

When the internal Baud Rate Generator is used, the Baud Rates are determined by the BRG overflow depending on the BRL reload value, the value of SPD bit (Speed Mode) in BDRCON register and the value of the SMOD1 bit in PCON register.

Figure 35. Internal Baud Rate



- The baud rate for UART is token by formula:

$$Baud_Rate = \frac{2^{SMOD1} \times F_{CLK\ PERIPH}}{2 \times 2 \times 6^{(1-SPD)} \times 16 \times [256 - (BRL)]}$$

$$(BRL) = 256 - \frac{2^{SMOD1} \times F_{CLK\ PERIPH}}{2 \times 2 \times 6^{(1-SPD)} \times 16 \times Baud_Rate}$$

Table 58. SCON Register – SCON Serial Control Register (98h)

7	6	5	4	3	2	1	0
FE/SM0	**SM1**	**SM2**	**REN**	**TB8**	**RB8**	**TI**	**RI**

Bit Number	Bit Mnemonic	Description
7	FE	**Framing Error bit (SMOD0 = 1)** Clear to reset the error state, not cleared by a valid stop bit. Set by hardware when an invalid stop bit is detected. SMOD0 must be set to enable access to the FE bit
	SM0	**Serial port Mode bit 0** Refer to SM1 for serial port mode selection. SMOD0 must be cleared to enable access to the SM0 bit
6	SM1	**Serial port Mode bit 1** SM0 SM1 Mode Description Baud Rate 0 0 0 Shift Register $F_{CPU\ PERIPH}/6$ 0 1 1 8-bit UART Variable 1 0 2 9-bit UART $F_{CPU\ PERIPH}/32$ or/16 1 1 3 9-bit UART Variable
5	SM2	**Serial port Mode 2 bit/Multiprocessor Communication Enable bit** Clear to disable multiprocessor communication feature. Set to enable multiprocessor communication feature in mode 2 and 3, and eventually mode 1. This bit should be cleared in mode 0.
4	REN	**Reception Enable bit** Clear to disable serial reception. Set to enable serial reception.
3	TB8	**Transmitter Bit 8/Ninth bit to Transmit in Modes 2 and 3** Clear to transmit a logic 0 in the 9th bit. Set to transmit a logic 1 in the 9th bit.
2	RB8	**Receiver Bit 8/Ninth bit received in modes 2 and 3** Cleared by hardware if 9th bit received is a logic 0. Set by hardware if 9th bit received is a logic 1. In mode 1, if SM2 = 0, RB8 is the received stop bit. In mode 0 RB8 is not used.
1	TI	**Transmit Interrupt flag** Clear to acknowledge interrupt. Set by hardware at the end of the 8th bit time in mode 0 or at the beginning of the stop bit in the other modes.
0	RI	**Receive Interrupt flag** Clear to acknowledge interrupt. Set by hardware at the end of the 8th bit time in mode 0, see Figure 32. and Figure 33. in the other modes.

Reset Value = 0000 0000b
Bit addressable

Example of computed value when X2 = 1, SMOD1 = 1, SPD = 1

Baud Rates	F_{osc} = 16.384 MHz		F_{osc} = 24 MHz	
	BRL	Error (%)	BRL	Error (%)
115200	247	1.23	243	0.16
57600	238	1.23	230	0.16
38400	229	1.23	217	0.16
28800	220	1.23	204	0.16
19200	203	0.63	178	0.16
9600	149	0.31	100	0.16
4800	43	1.23	-	-

Example of computed value when X2 = 0, SMOD1 = 0, SPD = 0

Baud Rates	F_{osc} = 16.384 MHz		F_{osc} = 24 MHz	
	BRL	Error (%)	BRL	Error (%)
4800	247	1.23	243	0.16
2400	238	1.23	230	0.16
1200	220	1.23	202	3.55
600	185	0.16	152	0.16

The baud rate generator can be used for mode 1 or 3 (refer to Figure 34.), but also for mode 0 for UART, thanks to the bit SRC located in BDRCON register (Table 61.)

UART Registers

SADEN - Slave Address Mask Register for UART (B9h)

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	–

Reset Value = 0000 0000b

SADDR - Slave Address Register for UART (A9h)

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	–

Reset Value = 0000 0000b

SBUF - Serial Buffer Register for UART (99h)

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	–

Reset Value = XXXX XXXXb

BRL - Baud Rate Reload Register for the internal baud rate generator, UART (9Ah)

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	–

Reset Value = 0000 0000b

Table 59. T2CON Register

T2CON - Timer 2 Control Register (C8h)

7	6	5	4	3	2	1	0
TF2	**EXF2**	**RCLK**	**TCLK**	**EXEN2**	**TR2**	**C/T2#**	**CP/RL2#**

Bit Number	Bit Mnemonic	Description
7	TF2	**Timer 2 overflow Flag** Must be cleared by software. Set by hardware on Timer 2 overflow, if RCLK = 0 and TCLK = 0.
6	EXF2	**Timer 2 External Flag** Set when a capture or a reload is caused by a negative transition on T2EX pin if EXEN2 = 1. When set, causes the CPU to vector to Timer 2 interrupt routine when Timer 2 interrupt is enabled. Must be cleared by software. EXF2 doesn't cause an interrupt in Up/down counter mode (DCEN = 1)
5	RCLK	**Receive Clock bit for UART** Cleared to use Timer 1 overflow as receive clock for serial port in mode 1 or 3. Set to use Timer 2 overflow as receive clock for serial port in mode 1 or 3.
4	TCLK	**Transmit Clock bit for UART** Cleared to use Timer 1 overflow as transmit clock for serial port in mode 1 or 3. Set to use Timer 2 overflow as transmit clock for serial port in mode 1 or 3.
3	EXEN2	**Timer 2 External Enable bit** Cleared to ignore events on T2EX pin for Timer 2 operation. Set to cause a capture or reload when a negative transition on T2EX pin is detected, if Timer 2 is not used to clock the serial port.
2	TR2	**Timer 2 Run control bit** Cleared to turn off Timer 2. Set to turn on Timer 2.
1	C/T2#	**Timer/Counter 2 select bit** Cleared for timer operation (input from internal clock system: $F_{CLK\ PERIPH}$). Set for counter operation (input from T2 input pin, falling edge trigger). Must be 0 for clock out mode.
0	CP/RL2#	**Timer 2 Capture/Reload bit** If RCLK = 1 or TCLK = 1, CP/RL2# is ignored and timer is forced to Auto-reload on Timer 2 overflow. Cleared to Auto-reload on Timer 2 overflows or negative transitions on T2EX pin if EXEN2 = 1. Set to capture on negative transitions on T2EX pin if EXEN2 = 1.

Reset Value = 0000 0000b
Bit addressable

Table 60. PCON Register

PCON - Power Control Register (87h)

7	6	5	4	3	2	1	0
SMOD1	SMOD0	-	POF	GF1	GF0	PD	IDL

Bit Number	Bit Mnemonic	Description
7	SMOD1	**Serial port Mode bit 1 for UART** Set to select double baud rate in mode 1, 2 or 3.
6	SMOD0	**Serial port Mode bit 0 for UART** Cleared to select SM0 bit in SCON register. Set to select FE bit in SCON register.
5	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
4	POF	**Power-Off Flag** Cleared to recognize next reset type. Set by hardware when V_{CC} rises from 0 to its nominal voltage. Can also be set by software.
3	GF1	**General-purpose Flag** Cleared by user for general-purpose usage. Set by user for general-purpose usage.
2	GF0	**General-purpose Flag** Cleared by user for general-purpose usage. Set by user for general-purpose usage.
1	PD	**Power-down Mode Bit** Cleared by hardware when reset occurs. Set to enter power-down mode.
0	IDL	**Idle Mode Bit** Cleared by hardware when interrupt or reset occurs. Set to enter idle mode.

Reset Value = 00X1 0000b
Not bit addressable

Power-off flag reset value will be 1 only after a power on (cold reset). A warm reset doesn't affect the value of this bit.

Table 61. BDRCON Register

BDRCON - Baud Rate Control Register (9Bh)

7	6	5	4	3	2	1	0
-	-	-	BRR	TBCK	RBCK	SPD	SRC

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	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
4	BRR	**Baud Rate Run Control bit** Cleared to stop the internal Baud Rate Generator. Set to start the internal Baud Rate Generator.
3	TBCK	**Transmission Baud rate Generator Selection bit for UART** Cleared to select Timer 1 or Timer 2 for the Baud Rate Generator. Set to select internal Baud Rate Generator.
2	RBCK	**Reception Baud Rate Generator Selection bit for UART** Cleared to select Timer 1 or Timer 2 for the Baud Rate Generator. Set to select internal Baud Rate Generator.
1	SPD	**Baud Rate Speed Control bit for UART** Cleared to select the SLOW Baud Rate Generator. Set to select the FAST Baud Rate Generator.
0	SRC	**Baud Rate Source select bit in Mode 0 for UART** Cleared to select $F_{OSC}/12$ as the Baud Rate Generator ($F_{CLK\ PERIPH}/6$ in X2 mode). Set to select the internal Baud Rate Generator for UARTs in mode 0.

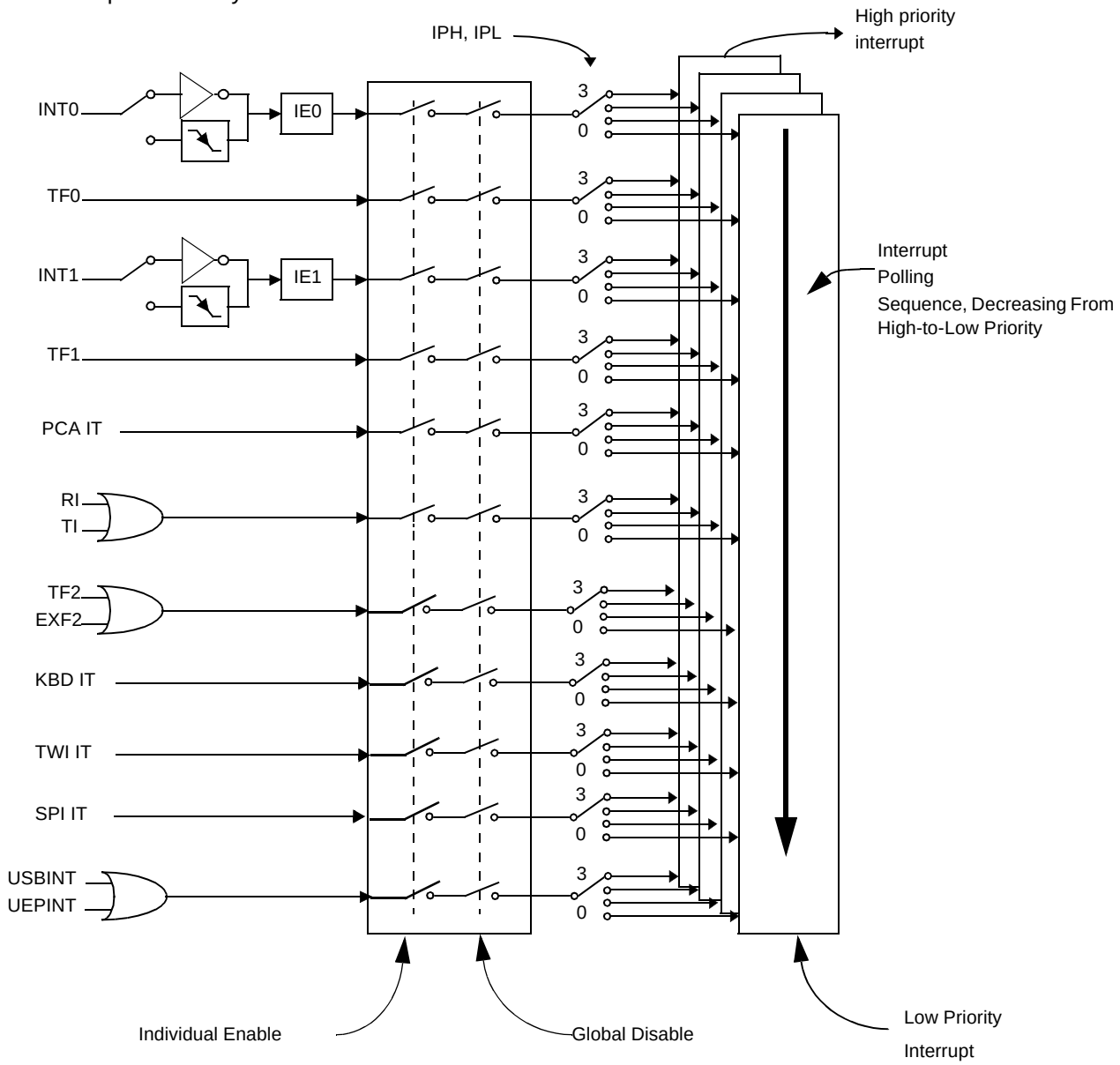
Reset Value = XXX0 0000b
Not bit addressable

Interrupt System

Overview

The AT89C5130A/31A-M has a total of 11 interrupt vectors: two external interrupts (INT0 and INT1), three timer interrupts (timers 0, 1 and 2), the serial port interrupt, SPI interrupt, Keyboard interrupt, USB interrupt and the PCA global interrupt. These interrupts are shown in Figure 36.

Figure 36. Interrupt Control System



Each of the interrupt sources can be individually enabled or disabled by setting or clearing a bit in the Interrupt Enable register (Table 63). This register also contains a global disable bit, which must be cleared to disable all interrupts at once.

Each interrupt source can also be individually programmed to one out of four priority levels by setting or clearing a bit in the Interrupt Priority register (Table 64.) and in the Interrupt Priority High register (Table 65). Table 62. shows the bit values and priority levels associated with each combination.

Registers

The PCA interrupt vector is located at address 0033H, the SPI interrupt vector is located at address 004BH and Keyboard interrupt vector is located at address 003BH. All other vectors addresses are the same as standard C52 devices.

Table 62. Priority Level Bit Values

IPH.x	IPL.x	Interrupt Level Priority
0	0	0 (Lowest)
0	1	1
1	0	2
1	1	3 (Highest)

A low-priority interrupt can be interrupted by a high priority interrupt, but not by another low-priority interrupt. A high-priority interrupt can't be interrupted by any other interrupt source.

If two interrupt requests of different priority levels are received simultaneously, the request of higher priority level is serviced. If interrupt requests of the same priority level are received simultaneously, an internal polling sequence determines which request is serviced. Thus within each priority level there is a second priority structure determined by the polling sequence.

Table 63. IEN0 Register

IEN0 - Interrupt Enable Register (A8h)

7	6	5	4	3	2	1	0
EA	EC	ET2	ES	ET1	EX1	ET0	EX0

Bit Number	Bit Mnemonic	Description
7	EA	**Enable All interrupt bit** Cleared to disable all interrupts. Set to enable all interrupts.
6	EC	**PCA interrupt enable bit** Cleared to disable. Set to enable.
5	ET2	**Timer 2 overflow interrupt Enable bit** Cleared to disable Timer 2 overflow interrupt. Set to enable Timer 2 overflow interrupt.
4	ES	**Serial port Enable bit** Cleared to disable serial port interrupt. Set to enable serial port interrupt.
3	ET1	**Timer 1 overflow interrupt Enable bit** Cleared to disable Timer 1 overflow interrupt. Set to enable Timer 1 overflow interrupt.
2	EX1	**External interrupt 1 Enable bit** Cleared to disable external interrupt 1. Set to enable external interrupt 1.
1	ET0	**Timer 0 overflow interrupt Enable bit** Cleared to disable timer 0 overflow interrupt. Set to enable timer 0 overflow interrupt.
0	EX0	**External interrupt 0 Enable bit** Cleared to disable external interrupt 0. Set to enable external interrupt 0.

Reset Value = 0000 0000b
Bit addressable

Table 64. IPL0 Register

IPL0 - Interrupt Priority Register (B8h)

7	6	5	4	3	2	1	0
-	PPCL	PT2L	PSL	PT1L	PX1L	PT0L	PX0L

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

Reset Value = X000 0000b
Bit addressable

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

Table 66. IEN1 Register

IEN1 - Interrupt Enable Register (B1h)

7	6	5	4	3	2	1	0
-	EUSB	-	-	-	ESPI	ETWI	EKB

Bit Number	Bit Mnemonic	Description
7	-	Reserved
6	EUSB	USB Interrupt Enable bit Cleared to disable USB interrupt. Set to enable USB interrupt.
5	-	Reserved
4	-	Reserved
3	-	Reserved
2	ESPI	**SPI interrupt Enable bit** Cleared to disable SPI interrupt. Set to enable SPI interrupt.
1	ETWI	**TWI interrupt Enable bit** Cleared to disable TWI interrupt. Set to enable TWI interrupt.
0	EKB	**Keyboard interrupt Enable bit** Cleared to disable keyboard interrupt. Set to enable keyboard interrupt.

Reset Value = X0XX X000b
Not bit addressable

Table 67. IPL1 Register

IPL1 - Interrupt Priority Register (B2h)

7	6	5	4	3	2	1	0
-	PUSBL	-	-	-	PSPIL	PTWIL	PKBDL

Bit Number	Bit Mnemonic	Description
7	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
6	PUSBL	**USB Interrupt Priority bit** Refer to PUSBH for priority level.
5	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
4	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
3	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
2	PSPIL	**SPI Interrupt Priority bit** Refer to PSPIH for priority level.
1	PTWIL	**TWI Interrupt Priority bit** Refer to PTWIH for priority level.
0	PKBL	**Keyboard Interrupt Priority bit** Refer to PKBH for priority level.

Reset Value = X0XX X000b
Not bit addressable

Table 68. IPH1 Register

IPH1 - Interrupt Priority High Register (B3h)

7	6	5	4	3	2	1	0
-	PUSBH	-	-	-	PSPIH	PTWIH	PKBH

Bit Number	Bit Mnemonic	Description
7	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
6	PUSBH	**USB Interrupt Priority High bit** PUSBH PUSBL Priority Level 0 0 Lowest 0 1 1 0 1 1 Highest
5	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
4	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
3	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
2	PSPIH	**SPI Interrupt Priority High bit** PSPIH PSPIL Priority Level 0 0 Lowest 0 1 1 0 1 1 Highest
1	PTWIH	**TWI Interrupt Priority High bit** PTWIH PTWIL Priority Level 0 0 Lowest 0 1 1 0 1 1 Highest
0	PKBH	**Keyboard Interrupt Priority High bit** PKBH PKBL Priority Level 0 0 Lowest 0 1 1 0 1 1 Highest

Reset Value = X0XX X000b
Not bit addressable

Interrupt Sources and Vector Addresses

Table 69. Vector Table

Number	Polling Priority	Interrupt Source	Interrupt Request	Vector Address
0	0	Reset		0000h
1	1	INT0	IE0	0003h
2	2	Timer 0	TF0	000Bh
3	3	INT1	IE1	0013h
4	4	Timer 1	IF1	001Bh
5	6	UART	RI+TI	0023h
6	7	Timer 2	TF2+EXF2	002Bh
7	5	PCA	CF + CCFn (n = 0-4)	0033h
8	8	Keyboard	KBDIT	003Bh
9	9	TWI	TWIIT	0043h
10	10	SPI	SPIIT	004Bh
11	11			0053h
12	12			005Bh
13	13			0063h
14	14	USB	UEPINT + USBINT	006Bh
15	15			0073h

Keyboard Interface

Introduction

The AT89C5130A/31A-M implements a keyboard interface allowing the connection of a 8 x n matrix keyboard. It is based on 8 inputs with programmable interrupt capability on both high or low level. These inputs are available as an alternate function of P1 and allow to exit from idle and power down modes.

Description

The keyboard interface communicates with the C51 core through 3 special function registers: KBLS, the Keyboard Level Selection register (Table 72), KBE, The Keyboard interrupt Enable register (Table 71), and KBF, the Keyboard Flag register (Table 70).

Interrupt

The keyboard inputs are considered as 8 independent interrupt sources sharing the same interrupt vector. An interrupt enable bit (KBD in IE1) allows global enable or disable of the keyboard interrupt (see Figure 37). As detailed in Figure 38 each keyboard input has the capability to detect a programmable level according to KBLS.x bit value. Level detection is then reported in interrupt flags KBF.x that can be masked by software using KBE.x bits.

This structure allow keyboard arrangement from 1 by n to 8 by n matrix and allow usage of P1 inputs for other purpose.

Figure 37. Keyboard Interface Block Diagram

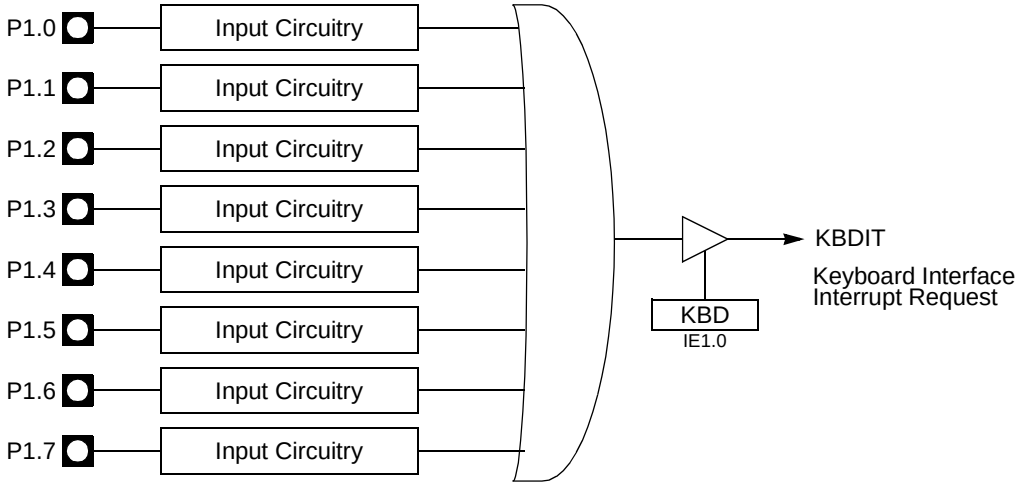
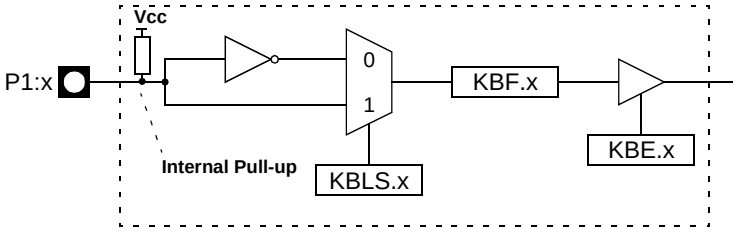


Figure 38. Keyboard Input Circuitry



Power Reduction Mode

P1 inputs allow exit from idle and power down modes as detailed in section “Power-down Mode”.

Registers

Table 70. KBF Register

KBF - Keyboard Flag Register (9Eh)

7	6	5	4	3	2	1	0
KBF7	KBF6	KBF5	KBF4	KBF3	KBF2	KBF1	KBF0

Bit Number	Bit Mnemonic	Description
7	KBF7	**Keyboard line 7 flag** Set by hardware when the Port line 7 detects a programmed level. It generates a Keyboard interrupt request if the KBKBIE.7 bit in KBIE register is set. Must be cleared by software.
6	KBF6	**Keyboard line 6 flag** Set by hardware when the Port line 6 detects a programmed level. It generates a Keyboard interrupt request if the KBIE.6 bit in KBIE register is set. Must be cleared by software.
5	KBF5	**Keyboard line 5 flag** Set by hardware when the Port line 5 detects a programmed level. It generates a Keyboard interrupt request if the KBIE.5 bit in KBIE register is set. Must be cleared by software.
4	KBF4	**Keyboard line 4 flag** Set by hardware when the Port line 4 detects a programmed level. It generates a Keyboard interrupt request if the KBIE.4 bit in KBIE register is set. Must be cleared by software.
3	KBF3	**Keyboard line 3 flag** Set by hardware when the Port line 3 detects a programmed level. It generates a Keyboard interrupt request if the KBIE.3 bit in KBIE register is set. Must be cleared by software.
2	KBF2	**Keyboard line 2 flag** Set by hardware when the Port line 2 detects a programmed level. It generates a Keyboard interrupt request if the KBIE.2 bit in KBIE register is set. Must be cleared by software.
1	KBF1	**Keyboard line 1 flag** Set by hardware when the Port line 1 detects a programmed level. It generates a Keyboard interrupt request if the KBIE.1 bit in KBIE register is set. Must be cleared by software.
0	KBF0	**Keyboard line 0 flag** Set by hardware when the Port line 0 detects a programmed level. It generates a Keyboard interrupt request if the KBIE.0 bit in KBIE register is set. Must be cleared by software.

Reset Value = 0000 0000b

Table 71. KBE Register

KBE **-** Keyboard Input Enable Register (9Dh)

7	6	5	4	3	2	1	0
KBE7	KBE6	KBE5	KBE4	KBE3	KBE2	KBE1	KBE0

Bit Number	Bit Mnemonic	Description
7	KBE7	**Keyboard line 7 Enable bit** Cleared to enable standard I/O pin. Set to enable KBF.7 bit in KBF register to generate an interrupt request.
6	KBE6	**Keyboard line 6 Enable bit** Cleared to enable standard I/O pin. Set to enable KBF.6 bit in KBF register to generate an interrupt request.
5	KBE5	**Keyboard line 5 Enable bit** Cleared to enable standard I/O pin. Set to enable KBF.5 bit in KBF register to generate an interrupt request.
4	KBE4	**Keyboard line 4 Enable bit** Cleared to enable standard I/O pin. Set to enable KBF.4 bit in KBF register to generate an interrupt request.
3	KBE3	**Keyboard line 3 Enable bit** Cleared to enable standard I/O pin. Set to enable KBF.3 bit in KBF register to generate an interrupt request.
2	KBE2	**Keyboard line 2 Enable bit** Cleared to enable standard I/O pin. Set to enable KBF.2 bit in KBF register to generate an interrupt request.
1	KBE1	**Keyboard line 1 Enable bit** Cleared to enable standard I/O pin. Set to enable KBF.1 bit in KBF register to generate an interrupt request.
0	KBE0	**Keyboard line 0 Enable bit** Cleared to enable standard I/O pin. Set to enable KBF.0 bit in KBF register to generate an interrupt request.

Reset Value = 0000 0000b

Table 72. KBLS Register

KBLS-Keyboard Level Selector Register (9Ch)

7	6	5	4	3	2	1	0
KBLS7	KBLS6	KBLS5	KBLS4	KBLS3	KBLS2	KBLS1	KBLS0

Bit Number	Bit Mnemonic	Description
7	KBLS7	**Keyboard line 7 Level Selection bit** Cleared to enable a low level detection on Port line 7. Set to enable a high level detection on Port line 7.
6	KBLS6	**Keyboard line 6 Level Selection bit** Cleared to enable a low level detection on Port line 6. Set to enable a high level detection on Port line 6.
5	KBLS5	**Keyboard line 5 Level Selection bit** Cleared to enable a low level detection on Port line 5. Set to enable a high level detection on Port line 5.
4	KBLS4	**Keyboard line 4 Level Selection bit** Cleared to enable a low level detection on Port line 4. Set to enable a high level detection on Port line 4.
3	KBLS3	**Keyboard line 3 Level Selection bit** Cleared to enable a low level detection on Port line 3. Set to enable a high level detection on Port line 3.
2	KBLS2	**Keyboard line 2 Level Selection bit** Cleared to enable a low level detection on Port line 2. Set to enable a high level detection on Port line 2.
1	KBLS1	**Keyboard line 1 Level Selection bit** Cleared to enable a low level detection on Port line 1. Set to enable a high level detection on Port line 1.
0	KBLS0	**Keyboard line 0 Level Selection bit** Cleared to enable a low level detection on Port line 0. Set to enable a high level detection on Port line 0.

Reset Value = 0000 0000b

Programmable LED

AT89C5130A/31A-M have up to 4 programmable LED current sources, configured by the register LEDCON.

Table 73. LEDCON Register

LEDCON (S:F1h) LED Control Register

7	6	5	4	3	2	1	0
LED3		LED2		LED1		LED0	

Bit Number	Bit Mnemonic	Description
7:6	LED3	**Port** **LED3** **Configuration** 0 0 Standard C51 Port 0 1 2 mA current source when P3.7 is low 1 0 4 mA current source when P3.7 is low 1 1 10 mA current source when P3.7 is low
5:4	LED2	**Port** **/LED2** **Configuration** 0 0 Standard C51 Port 0 1 2 mA current source when P3.6 is low 1 0 4 mA current source when P3.6 is low 1 1 10 mA current source when P3.6 is low
3:2	LED1	**Port/** **LED1** **Configuration** 0 0 Standard C51 Port 0 1 2 mA current source when P3.5 is low 1 0 4 mA current source when P3.5 is low 1 1 10 mA current source when P3.5 is low
1:0	LED0	**Port/** **LED0** **Configuration** 0 0 Standard C51 Port 0 1 2 mA current source when P3.3 is low 1 0 4 mA current source when P3.3 is low 1 1 10 mA current source when P3.3 is low

Reset Value = 00h

Serial Peripheral Interface (SPI)

The Serial Peripheral Interface module (SPI) allows full-duplex, synchronous, serial communication between the MCU and peripheral devices, including other MCUs.

Features

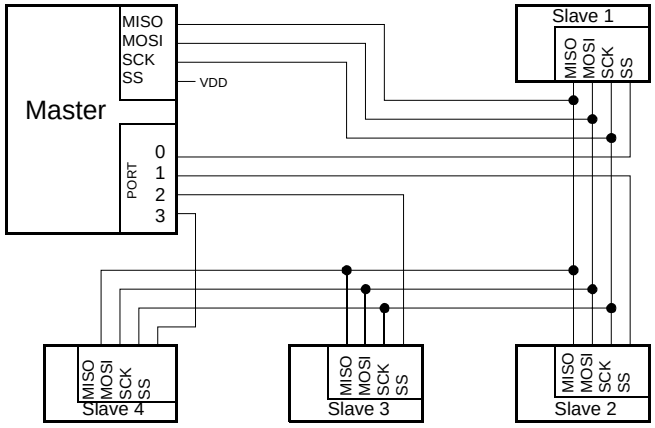
Features of the SPI module include the following:

- Full-duplex, three-wire synchronous transfers
- Master or Slave operation
- Eight programmable Master clock rates
- Serial clock with programmable polarity and phase
- Master mode fault error flag with MCU interrupt capability
- Write collision flag protection

Signal Description

Figure 39 shows a typical SPI bus configuration using one Master controller and many Slave peripherals. The bus is made of three wires connecting all the devices:

Figure 39. SPI Master/Slaves Interconnection



The Master device selects the individual Slave devices by using four pins of a parallel port to control the four $\overline{SS}$ pins of the Slave devices.

Master Output Slave Input (MOSI)

This 1-bit signal is directly connected between the Master Device and a Slave Device. The MOSI line is used to transfer data in series from the Master to the Slave. Therefore, it is an output signal from the Master, and an input signal to a Slave. A byte (8-bit word) is transmitted most significant bit (MSB) first, least significant bit (LSB) last.

Master Input Slave Output (MISO)

This 1-bit signal is directly connected between the Slave Device and a Master Device. The MISO line is used to transfer data in series from the Slave to the Master. Therefore, it is an output signal from the Slave, and an input signal to the Master. A byte (8-bit word) is transmitted most significant bit (MSB) first, least significant bit (LSB) last.

SPI Serial Clock (SCK)

This signal is used to synchronize the data movement both in and out the devices through their MOSI and MISO lines. It is driven by the Master for eight clock cycles which allows to exchange one byte on the serial lines.

Slave Select ($\overline{SS}$)

Each Slave peripheral is selected by one Slave Select pin ($\overline{SS}$). This signal must stay low for any message for a Slave. It is obvious that only one Master ($\overline{SS}$ high level) can drive the network. The Master may select each Slave device by software through port

pins (Figure 39). To prevent bus conflicts on the MISO line, only one slave should be selected at a time by the Master for a transmission.

In a Master configuration, the $\overline{SS}$ line can be used in conjunction with the MODF flag in the SPI Status register (SPSTA) to prevent multiple masters from driving MOSI and SCK (see Section “Error Conditions”, page 96).

A high level on the $\overline{SS}$ pin puts the MISO line of a Slave SPI in a high-impedance state.

The $\overline{SS}$ pin could be used as a general-purpose if the following conditions are met:

- The device is configured as a Master and the SSDIS control bit in SPCON is set. This kind of configuration can be found when only one Master is driving the network and there is no way that the $\overline{SS}$ pin could be pulled low. Therefore, the MODF flag in the SPSTA will never be set(1).
- The Device is configured as a Slave with CPHA and SSDIS control bits set(2) This kind of configuration can happen when the system comprises one Master and one Slave only. Therefore, the device should always be selected and there is no reason that the Master uses the $\overline{SS}$ pin to select the communicating Slave device.

Notes: 1. Clearing SSDIS control bit does not clear MODF.
2. Special care should be taken not to set SSDIS control bit when CPHA =’0’ because in this mode, the $\overline{SS}$ is used to start the transmission.

Baud Rate

In Master mode, the baud rate can be selected from a baud rate generator which is controlled by three bits in the SPCON register: SPR2, SPR1 and SPR0. The Master clock is chosen from one of seven clock rates resulting from the division of the internal clock by 2, 4, 8, 16, 32, 64 or 128.

Table 74 gives the different clock rates selected by SPR2:SPR1:SPR0:

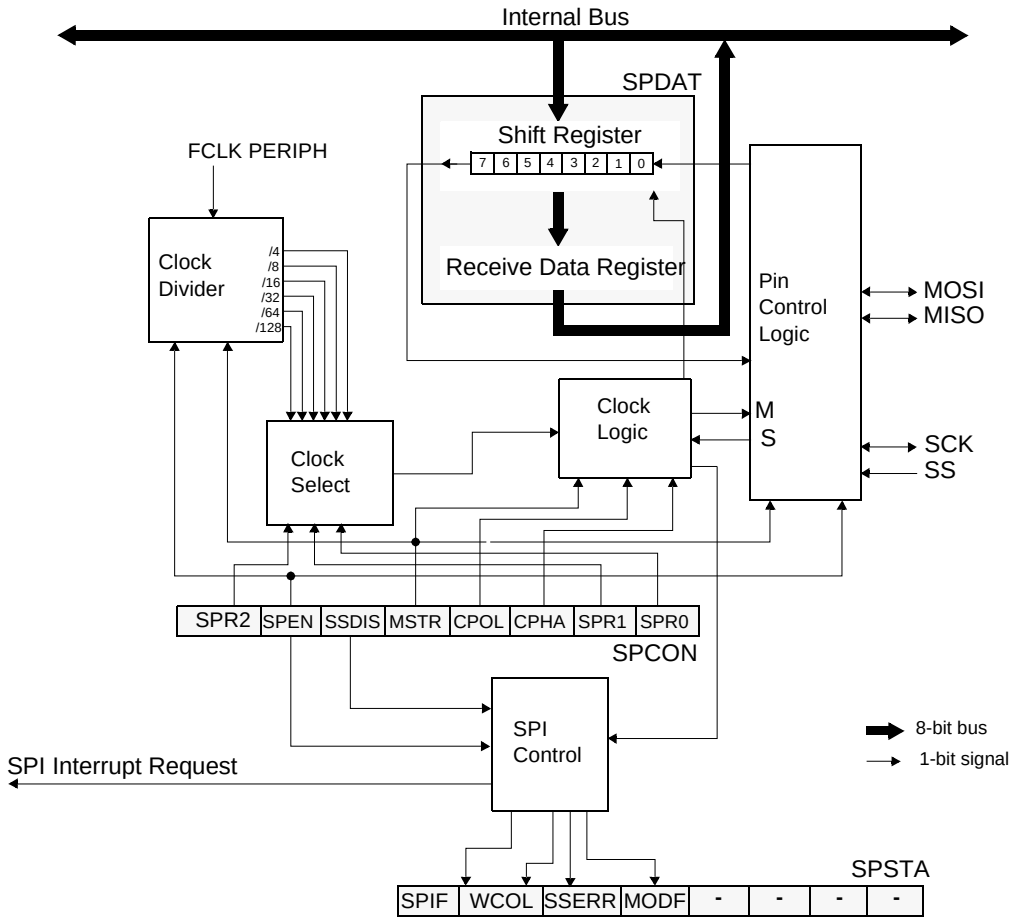
Table 74. SPI Master Baud Rate Selection

SPR2	SPR1	SPR0	Clock Rate	Baud Rate Divisor (BD)
0	0	0	Don’t Use	No BRG
0	0	1	$F_{CLK\ PERIPH}/4$	4
0	1	0	$F_{CLK\ PERIPH}/8$	8
0	1	1	$F_{CLK\ PERIPH}/16$	16
1	0	0	$F_{CLK\ PERIPH}/32$	32
1	0	1	$F_{CLK\ PERIPH}/64$	64
1	1	0	$F_{CLK\ PERIPH}/128$	128
1	1	1	Don’t Use	No BRG

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)

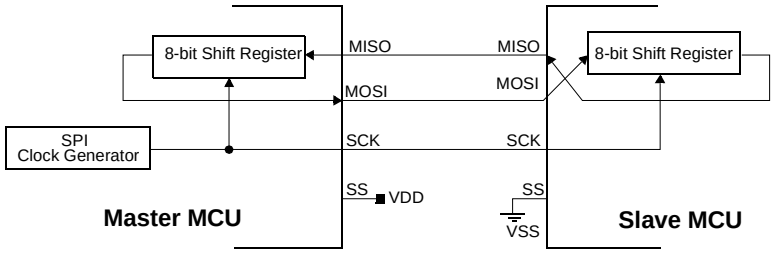
Once the SPI is configured, the data exchange is made using:

- 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 ($\overline{SS}$) 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).

Figure 41. Full-duplex Master/Slave Interconnection



Master Mode

The SPI operates in Master mode when the Master bit, MSTR (1), in the SPCON register is set. Only one Master SPI device can initiate transmissions. Software begins the transmission from a Master SPI module by writing to the Serial Peripheral Data Register (SPDAT). If the shift register is empty, the byte is immediately transferred to the shift register. The byte begins shifting out on MOSI pin under the control of the serial clock, SCK. Simultaneously, another byte shifts in from the Slave on the Master's MISO pin. The transmission ends when the Serial Peripheral transfer data flag, SPIF, in SPSTA becomes set. At the same time that SPIF becomes set, the received byte from the Slave is transferred to the receive data register in SPDAT. Software clears SPIF by reading the Serial Peripheral Status register (SPSTA) with the SPIF bit set, and then reading the SPDAT.

Slave Mode

The SPI operates in Slave mode when the Master bit, MSTR (2), in the SPCON register is cleared. Before a data transmission occurs, the Slave Select pin, $\overline{SS}$, of the Slave device must be set to'0'. $\overline{SS}$ must remain low until the transmission is complete.

In a Slave SPI module, data enters the shift register under the control of the SCK from the Master SPI module. After a byte enters the shift register, it is immediately transferred to the receive data register in SPDAT, and the SPIF bit is set. To prevent an overflow condition, Slave software must then read the SPDAT before another byte enters the shift register (3). A Slave SPI must complete the write to the SPDAT (shift register) at least one bus cycle before the Master SPI starts a transmission. If the write to the data register is late, the SPI transmits the data already in the shift register from the previous transmission.

Transmission Formats

Software can select any of four combinations of serial clock (SCK) phase and polarity using two bits in the SPCON: the Clock POLarity (CPOL (4)) and the Clock PHAse (CPHA[4]). CPOL defines the default SCK line level in idle state. It has no significant effect on the transmission format. CPHA defines the edges on which the input data are sampled and the edges on which the output data are shifted (Figure 42 and Figure 43). The clock phase and polarity should be identical for the Master SPI device and the communicating Slave device.

1. The SPI module should be configured as a Master before it is enabled (SPEN set). Also the Master SPI should be configured before the Slave SPI.
2. The SPI module should be configured as a Slave before it is enabled (SPEN set).
3. The maximum frequency of the SCK for an SPI configured as a Slave is the bus clock speed.
4. Before writing to the CPOL and CPHA bits, the SPI should be disabled (SPEN ='0').

Figure 42. Data Transmission Format (CPHA = 0)

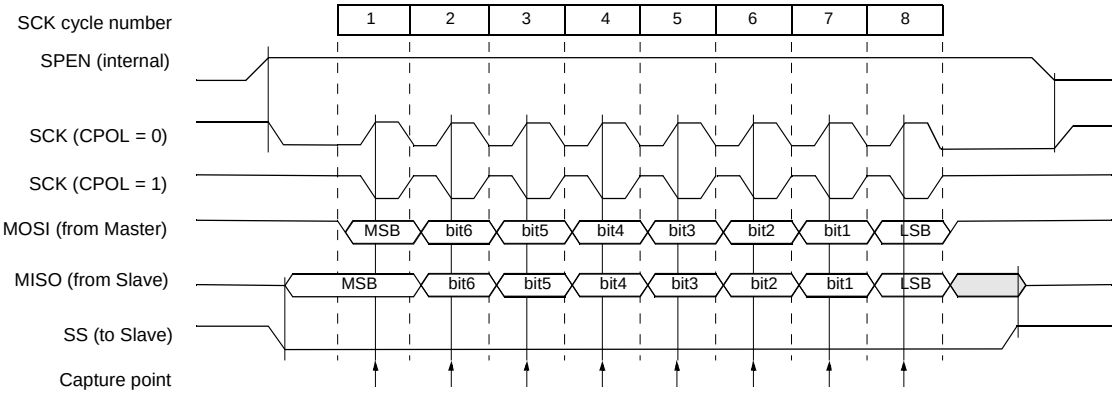


Figure 43. Data Transmission Format (CPHA = 1)

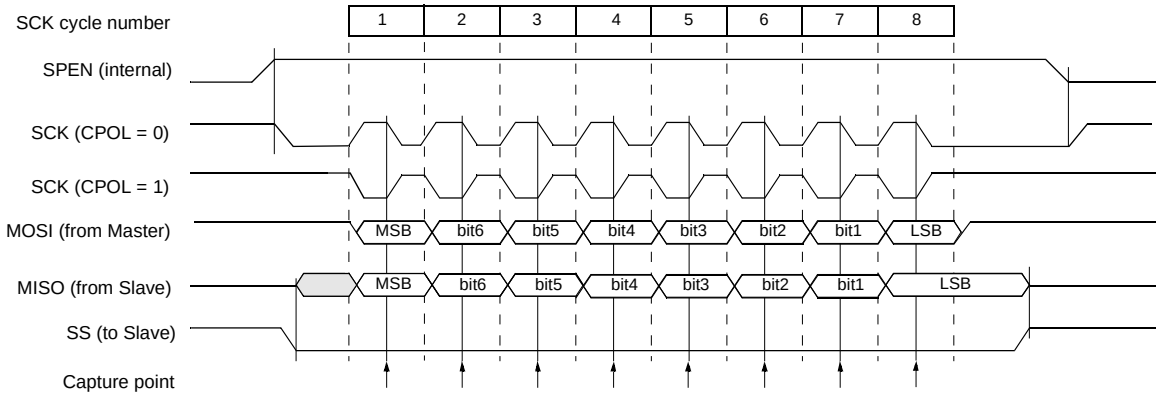
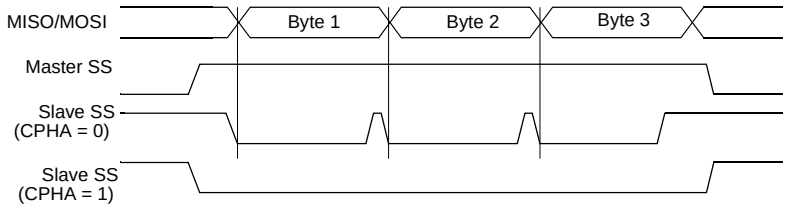


Figure 44. CPHA/$\overline{SS}$ Timing



As shown in Figure 43, the first SCK edge is the MSB capture strobe. Therefore the Slave must begin driving its data before the first SCK edge, and a falling edge on the $\overline{SS}$ pin is used to start the transmission. The $\overline{SS}$ pin must be toggled high and then low between each byte transmitted (Figure 40).

Figure 44 shows an SPI transmission in which CPHA is'1'. In this case, the Master begins driving its MOSI pin on the first SCK edge. Therefore the Slave uses the first SCK edge as a start transmission signal. The $\overline{SS}$ pin can remain low between transmissions (Figure 39). This format may be preferable in systems having only one Master and only one Slave driving the MISO data line.

Error Conditions

The following flags in the SPSTA signal SPI error conditions:

Mode Fault (MODF)

Mode Fault error in Master mode SPI indicates that the level on the Slave Select ($\overline{SS}$) pin is inconsistent with the actual mode of the device. MODF is set to warn that there may have a multi-master conflict for system control. In this case, the SPI system is affected in the following ways:

- An SPI receiver/error CPU interrupt request is generated,
- The SPEN bit in SPCON is cleared. This disable the SPI,
- The MSTR bit in SPCON is cleared

When $\overline{SS}$ DISable (SSDIS) bit in the SPCON register is cleared, the MODF flag is set when the $\overline{SS}$ signal becomes “0”.

However, as stated before, for a system with one Master, if the $\overline{SS}$ pin of the Master device is pulled low, there is no way that another Master attempt to drive the network. In this case, to prevent the MODF flag from being set, software can set the SSDIS bit in the SPCON register and therefore making the $\overline{SS}$ pin as a general-purpose I/O pin.

Clearing the MODF bit is accomplished by a read of SPSTA register with MODF bit set, followed by a write to the SPCON register. SPEN Control bit may be restored to its original set state after the MODF bit has been cleared.

Write Collision (WCOL)

A Write Collision (WCOL) flag in the SPSTA is set when a write to the SPDAT register is done during a transmit sequence.

WCOL does not cause an interruption, and the transfer continues uninterrupted.

Clearing the WCOL bit is done through a software sequence of an access to SPSTA and an access to SPDAT.

Overrun Condition

An overrun condition occurs when the Master device tries to send several data bytes and the Slave devise has not cleared the SPIF bit issuing from the previous data byte transmitted. In this case, the receiver buffer contains the byte sent after the SPIF bit was last cleared. A read of the SPDAT returns this byte. All others bytes are lost.

This condition is not detected by the SPI peripheral.

Interrupts

Two SPI status flags can generate a CPU interrupt requests:

Table 75. SPI Interrupts

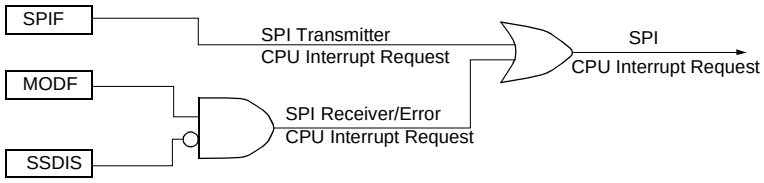
Flag	Request
SPIF (SP Data Transfer)	SPI Transmitter Interrupt request
MODF (Mode Fault)	SPI Receiver/Error Interrupt Request (if SSDIS = “0”)

Serial Peripheral data transfer flag, SPIF: This bit is set by hardware when a transfer has been completed. SPIF bit generates transmitter CPU interrupt requests.

Mode Fault flag, MODF: This bit becomes set to indicate that the level on the SS is inconsistent with the mode of the SPI. MODF with SSDIS reset, generates receiver/error CPU interrupt requests.

Figure 45 gives a logical view of the above statements.

Figure 45. SPI Interrupt Requests Generation



Registers

There are three registers in the module that provide control, status and data storage functions. These registers are describes in the following paragraphs.

Serial Peripheral Control Register (SPCON)

- The Serial Peripheral Control Register does the following:
  - Selects one of the Master clock rates
  - Configure the SPI module as Master or Slave
  - Selects serial clock polarity and phase
  - Enables the SPI module
  - Frees the SS pin for a general-purpose

Table 76 describes this register and explains the use of each bit.

Table 76. SPCON Register

7	6	5	4	3	2	1	0
SPR2	**SPEN**	**SSDIS**	**MSTR**	**CPOL**	**CPHA**	**SPR1**	**SPR0**

Bit Number	Bit Mnemonic	Description
7	SPR2	**Serial Peripheral Rate 2** Bit with SPR1 and SPR0 define the clock rate.
6	SPEN	**Serial Peripheral Enable** Cleared to disable the SPI interface. Set to enable the SPI interface.
5	SSDIS	**$\overline{SS}$ Disable** Cleared to enable $\overline{SS}$ in both Master and Slave modes. Set to disable $\overline{SS}$ in both Master and Slave modes. In Slave mode, this bit has no effect if CPHA = “0”.
5	MSTR	**Serial Peripheral Master** Cleared to configure the SPI as a Slave. Set to configure the SPI as a Master.
4	CPOL	**Clock Polarity** Cleared to have the SCK set to “0” in idle state. Set to have the SCK set to “1” in idle state.
3	CPHA	**Clock Phase** Cleared to have the data sampled when the SCK leaves the idle state (see CPOL). Set to have the data sampled when the SCK returns to idle state (see CPOL).

Bit Number	Bit Mnemonic	Description			
2	SPR1	**SPR2**	**SPR1**	**SPR0**	**Serial Peripheral Rate**
		0	**0**	**0**	**Reserved**
		0	**0**	**1**	**$F_{CLK\ PERIPH}/4$**
		0	**1**	**0**	**$F_{CLK\ PERIPH}/8$**
1	SPR0	**0**	**1**	**1**	**$F_{CLK\ PERIPH}/16$**
		1	**0**	**0**	**$F_{CLK\ PERIPH}/32$**
		1	**0**	**1**	**$F_{CLK\ PERIPH}/64$**
		1	**1**	**0**	**$F_{CLK\ PERIPH}/128$**
		1	**1**	**1**	**Reserved**

Reset Value = 0001 0100b

Not bit addressable

Serial Peripheral Status Register (SPSTA)

The Serial Peripheral Status Register contains flags to signal the following conditions:

- Data transfer complete
- Write collision
- Inconsistent logic level on $\overline{SS}$ pin (mode fault error)

Table 77 describes the SPSTA register and explains the use of every bit in the register.

Table 77. SPSTA Register

SPSTA - Serial Peripheral Status and Control register (0C4H)

7	6	5	4	3	2	1	0
SPIF	**WCOL**	**SSERR**	**MODF**	**-**	**-**	**-**	**-**

Bit Number	Bit Mnemonic	Description
7	SPIF	**Serial Peripheral data transfer flag** Cleared by hardware to indicate data transfer is in progress or has been approved by a clearing sequence. Set by hardware to indicate that the data transfer has been completed.
6	WCOL	**Write Collision flag** Cleared by hardware to indicate that no collision has occurred or has been approved by a clearing sequence. Set by hardware to indicate that a collision has been detected.
5	SSERR	**Synchronous Serial Slave Error flag** Set by hardware when $\overline{SS}$ is de- asserted before the end of a received data. Cleared by disabling the SPI (clearing SPEN bit in SPCON).
4	MODF	**Mode Fault** Cleared by hardware to indicate that the $\overline{SS}$ pin is at appropriate logic level, or has been approved by a clearing sequence. Set by hardware to indicate that the $\overline{SS}$ pin is at inappropriate logic level.
3	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit

Bit Number	Bit Mnemonic	Description
2	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit
1	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
0	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.

Reset Value = 00X0 XXXXb
Not Bit addressable

Serial Peripheral Data Register (SPDAT)

The Serial Peripheral Data Register (Table 78) is a read/write buffer for the receive data register. A write to SPDAT places data directly into the shift register. No transmit buffer is available in this model.

A Read of the SPDAT returns the value located in the receive buffer and not the content of the shift register.

Table 78. SPDAT Register

SPDAT - Serial Peripheral Data Register (0C5H)

7	6	5	4	3	2	1	0
R7	**R6**	**R5**	**R4**	**R3**	**R2**	**R1**	**R0**

Reset Value = Indeterminate

R7:R0: Receive data bits

SPCON, SPSTA and SPDAT registers may be read and written at any time while there is no on-going exchange. However, special care should be taken when writing to them while a transmission is on-going:

- Do not change SPR2, SPR1 and SPR0
- Do not change CPHA and CPOL
- Do not change MSTR
- Clearing SPEN would immediately disable the peripheral
- Writing to the SPDAT will cause an overflow

Two Wire Interface (TWI)

This section describes the 2-wire interface. The 2-wire bus is a bi-directional 2-wire serial communication standard. It is designed primarily for simple but efficient integrated circuit (IC) control. The system is comprised of two lines, SCL (Serial Clock) and SDA (Serial Data) that carry information between the ICs connected to them. The serial data transfer is limited to 100 Kbit/s in standard mode. Various communication configuration can be designed using this bus. Figure 46 shows a typical 2-wire bus configuration. All the devices connected to the bus can be master and slave.

Figure 46. 2-wire Bus Configuration

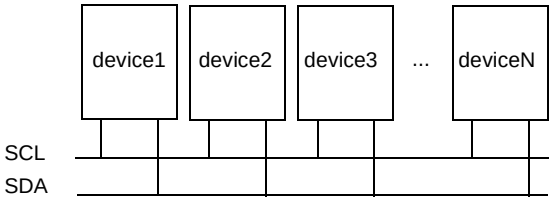
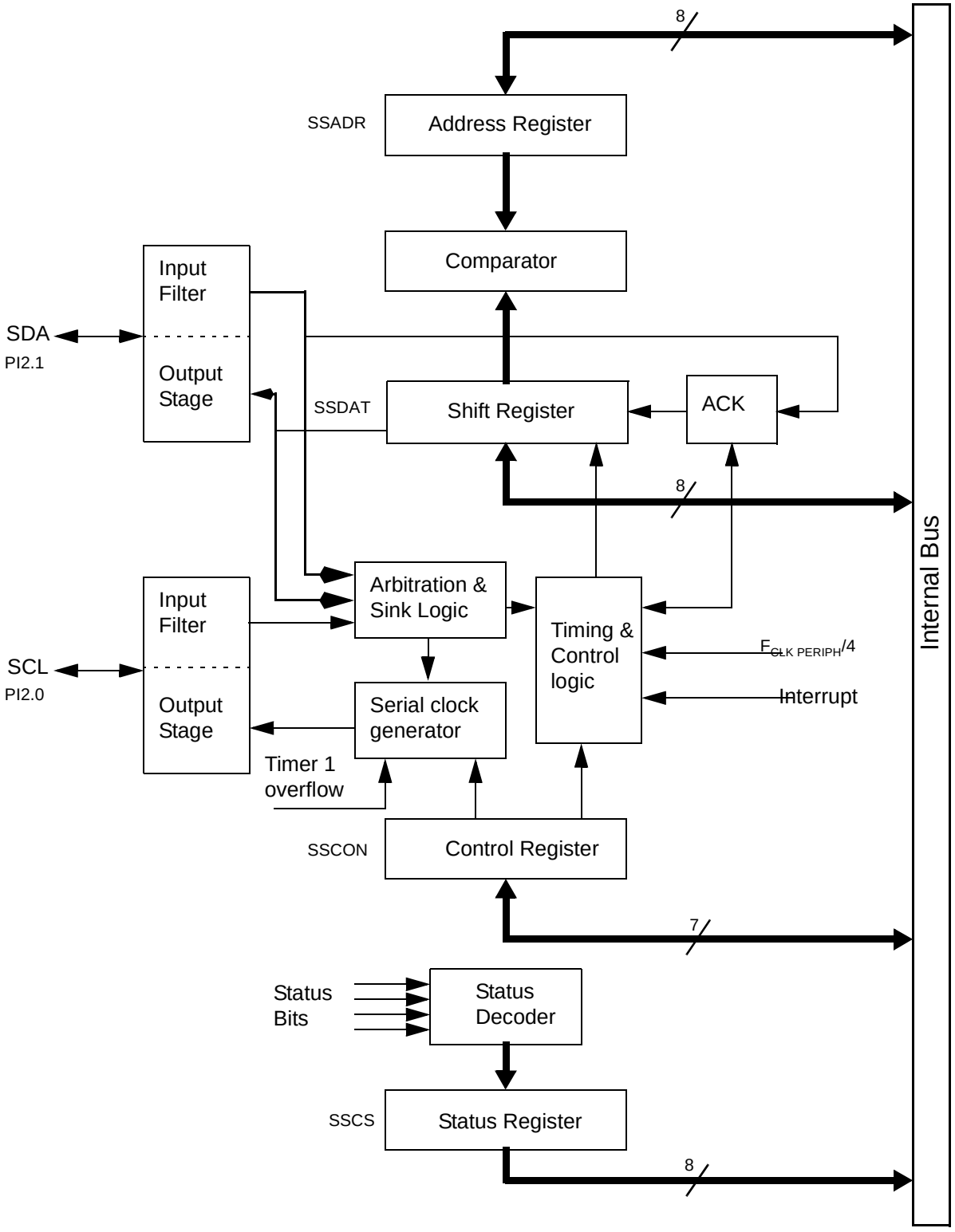


Figure 47. Block Diagram



Description

The CPU interfaces to the 2-wire logic via the following four 8-bit special function registers: the Synchronous Serial Control register (SSCON; Table 88), the Synchronous Serial Data register (SSDAT; Table 89), the Synchronous Serial Control and Status register (SSCS; Table 90) and the Synchronous Serial Address register (SSADR Table 91).

SSCON is used to enable the TWI interface, to program the bit rate (see Table 81), to enable slave modes, to acknowledge or not a received data, to send a START or a STOP condition on the 2-wire bus, and to acknowledge a serial interrupt. A hardware reset disables the TWI module.

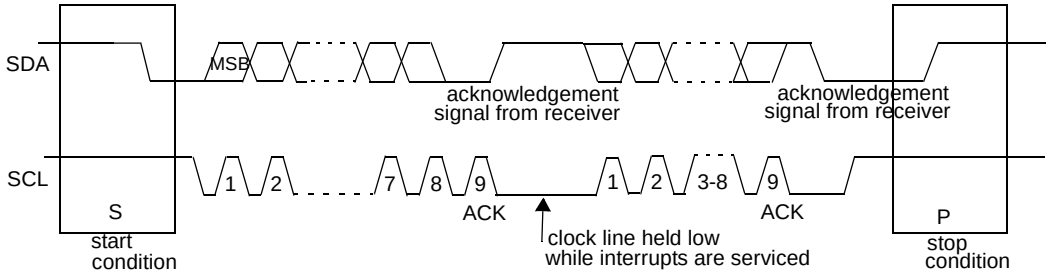
SSCS contains a status code which reflects the status of the 2-wire logic and the 2-wire bus. The three least significant bits are always zero. The five most significant bits contains the status code. There are 26 possible status codes. When SSCS contains F8h, no relevant state information is available and no serial interrupt is requested. A valid status code is available in SSCS one machine cycle after SI is set by hardware and is still present one machine cycle after SI has been reset by software. to Table 87. give the status for the master modes and miscellaneous states.

SSDAT contains a byte of serial data to be transmitted or a byte which has just been received. It is addressable while it is not in process of shifting a byte. This occurs when 2-wire logic is in a defined state and the serial interrupt flag is set. Data in SSDAT remains stable as long as SI is set. While data is being shifted out, data on the bus is simultaneously shifted in; SSDAT always contains the last byte present on the bus.

SSADR may be loaded with the 7-bit slave address (7 most significant bits) to which the TWI module will respond when programmed as a slave transmitter or receiver. The LSB is used to enable general call address (00h) recognition.

Figure 48 shows how a data transfer is accomplished on the 2-wire bus.

Figure 48. Complete Data Transfer on 2-wire Bus



The four operating modes are:

- Master Transmitter
- Master Receiver
- Slave transmitter
- Slave receiver

Data transfer in each mode of operation is shown in Table to Table 87 and Figure 49. to Figure 52.. These figures contain the following abbreviations:

S : START condition

R : Read bit (high level at SDA)

W: Write bit (low level at SDA)

A: Acknowledge bit (low level at SDA)

$\overline{A}$: Not acknowledge bit (high level at SDA)

Data: 8-bit data byte

P : STOP condition

In Figure 49 to Figure 52, circles are used to indicate when the serial interrupt flag is set. The numbers in the circles show the status code held in SSCS. At these points, a service routine must be executed to continue or complete the serial transfer. These service routines are not critical since the serial transfer is suspended until the serial interrupt flag is cleared by software.

When the serial interrupt routine is entered, the status code in SSCS is used to branch to the appropriate service routine. For each status code, the required software action and details of the following serial transfer are given in Table to Table 87.

Master Transmitter Mode

In the master transmitter mode, a number of data bytes are transmitted to a slave receiver (Figure 49). Before the master transmitter mode can be entered, SSCON must be initialised as follows:

Table 79. SSCON Initialization

CR2	SSIE	STA	STO	SI	AA	CR1	CR0
bit rate	1	0	0	0	X	bit rate	bit rate

CR0, CR1 and CR2 define the internal serial bit rate if external bit rate generator is not used. SSIE must be set to enable TWI. STA, STO and SI must be cleared.

The master transmitter mode may now be entered by setting the STA bit. The 2-wire logic will now test the 2-wire bus and generate a START condition as soon as the bus becomes free. When a START condition is transmitted, the serial interrupt flag (SI bit in SSCON) is set, and the status code in SSCS will be 08h. This status must be used to vector to an interrupt routine that loads SSDAT with the slave address and the data direction bit (SLA+W).

When the slave address and the direction bit have been transmitted and an acknowledgement bit has been received, SI is set again and a number of status code in SSCS are possible. There are 18h, 20h or 38h for the master mode and also 68h, 78h or B0h if the slave mode was enabled (AA=logic 1). The appropriate action to be taken for each of these status code is detailed in Table . This scheme is repeated until a STOP condition is transmitted.

SSIE, CR2, CR1 and CR0 are not affected by the serial transfer and are referred to Table 7 to Table 11. After a repeated START condition (state 10h) the TWI module may switch to the master receiver mode by loading SSDAT with SLA+R.

Master Receiver Mode

In the master receiver mode, a number of data bytes are received from a slave transmitter (Figure 50). The transfer is initialized as in the master transmitter mode. When the START condition has been transmitted, the interrupt routine must load SSDAT with the 7-bit slave address and the data direction bit (SLA+R). The serial interrupt flag SI must then be cleared before the serial transfer can continue.

When the slave address and the direction bit have been transmitted and an acknowledgement bit has been received, the serial interrupt flag is set again and a number of status code in SSCS are possible. There are 40h, 48h or 38h for the master mode and also 68h, 78h or B0h if the slave mode was enabled (AA=logic 1). The appropriate action to be taken for each of these status code is detailed in Table . This scheme is repeated until a STOP condition is transmitted.

SSIE, CR2, CR1 and CR0 are not affected by the serial transfer and are referred to Table 7 to Table 11. After a repeated START condition (state 10h) the TWI module may switch to the master transmitter mode by loading SSDAT with SLA+W.

Slave Receiver Mode

In the slave receiver mode, a number of data bytes are received from a master transmitter (Figure 51). To initiate the slave receiver mode, SSADR and SSCON must be loaded as follows:

Table 80. SSADR: Slave Receiver Mode Initialization

A6	A5	A4	A3	A2	A1	A0	GC
own slave address							

The upper 7 bits are the address to which the TWI module will respond when addressed by a master. If the LSB (GC) is set the TWI module will respond to the general call address (00h); otherwise it ignores the general call address.

Table 81. SSCON: Slave Receiver Mode Initialization

CR2	SSIE	STA	STO	SI	AA	CR1	CR0
bit rate	1	0	0	0	1	bit rate	bit rate

CR0, CR1 and CR2 have no effect in the slave mode. SSIE must be set to enable the TWI. The AA bit must be set to enable the own slave address or the general call address acknowledgement. STA, STO and SI must be cleared.

When SSADR and SSCON have been initialised, the TWI module waits until it is addressed by its own slave address followed by the data direction bit which must be at logic 0 (W) for the TWI to operate in the slave receiver mode. After its own slave address and the W bit have been received, the serial interrupt flag is set and a valid status code can be read from SSCS. This status code is used to vector to an interrupt service routine.The appropriate action to be taken for each of these status code is detailed in Table . The slave receiver mode may also be entered if arbitration is lost while TWI is in the master mode (states 68h and 78h).

If the AA bit is reset during a transfer, TWI module will return a not acknowledge (logic 1) to SDA after the next received data byte. While AA is reset, the TWI module does not respond to its own slave address. However, the 2-wire bus is still monitored and address recognition may be resume at any time by setting AA. This means that the AA bit may be used to temporarily isolate the module from the 2-wire bus.

Slave Transmitter Mode

In the slave transmitter mode, a number of data bytes are transmitted to a master receiver (Figure 52). Data transfer is initialized as in the slave receiver mode. When SSADR and SSCON have been initialized, the TWI module waits until it is addressed by

its own slave address followed by the data direction bit which must be at logic 1 (R) for TWI to operate in the slave transmitter mode. After its own slave address and the R bit have been received, the serial interrupt flag is set and a valid status code can be read from SSCS. This status code is used to vector to an interrupt service routine. The appropriate action to be taken for each of these status code is detailed in Table . The slave transmitter mode may also be entered if arbitration is lost while the TWI module is in the master mode.

If the AA bit is reset during a transfer, the TWI module will transmit the last byte of the transfer and enter state C0h or C8h. the TWI module is switched to the not addressed slave mode and will ignore the master receiver if it continues the transfer. Thus the master receiver receives all 1's as serial data. While AA is reset, the TWI module does not respond to its own slave address. However, the 2-wire bus is still monitored and address recognition may be resume at any time by setting AA. This means that the AA bit may be used to temporarily isolate the TWI module from the 2-wire bus.

Miscellaneous States

There are two SSCS codes that do not correspond to a define TWI hardware state (Table 87). These codes are discuss hereafter.

Status F8h indicates that no relevant information is available because the serial interrupt flag is not set yet. This occurs between other states and when the TWI module is not involved in a serial transfer.

Status 00h indicates that a bus error has occurred during a TWI serial transfer. A bus error is caused when a START or a STOP condition occurs at an illegal position in the format frame. Examples of such illegal positions happen during the serial transfer of an address byte, a data byte, or an acknowledge bit. When a bus error occurs, SI is set. To recover from a bus error, the STO flag must be set and SI must be cleared. This causes the TWI module to enter the not addressed slave mode and to clear the STO flag (no other bits in SSCON are affected). The SDA and SCL lines are released and no STOP condition is transmitted.

Notes

the TWI module interfaces to the external 2-wire bus via two port pins: SCL (serial clock line) and SDA (serial data line). To avoid low level asserting on these lines when the TWI module is enabled, the output latches of SDA and SLC must be set to logic 1.

Table 82. Bit Frequency Configuration

			Bit Frequency (kHz)		
CR2	**CR1**	**CR0**	**F_{OSCA}= 12 MHz**	**F_{OSCA} = 16 MHz**	**F_{OSCA} divided by**
0	0	0	47	62.5	256
0	0	1	53.5	71.5	224
0	1	0	62.5	83	192
0	1	1	75	100	160
1	0	0	-	-	Unused
1	0	1	100	133.3	120
1	1	0	200	266.6	60
1	1	1	0.5 <. < 62.5	0.67 <. < 83	96 · (256 - reload valueTimer 1) (reload value range: 0-254 in mode 2)

Figure 49. Format and State in the Master Transmitter Mode

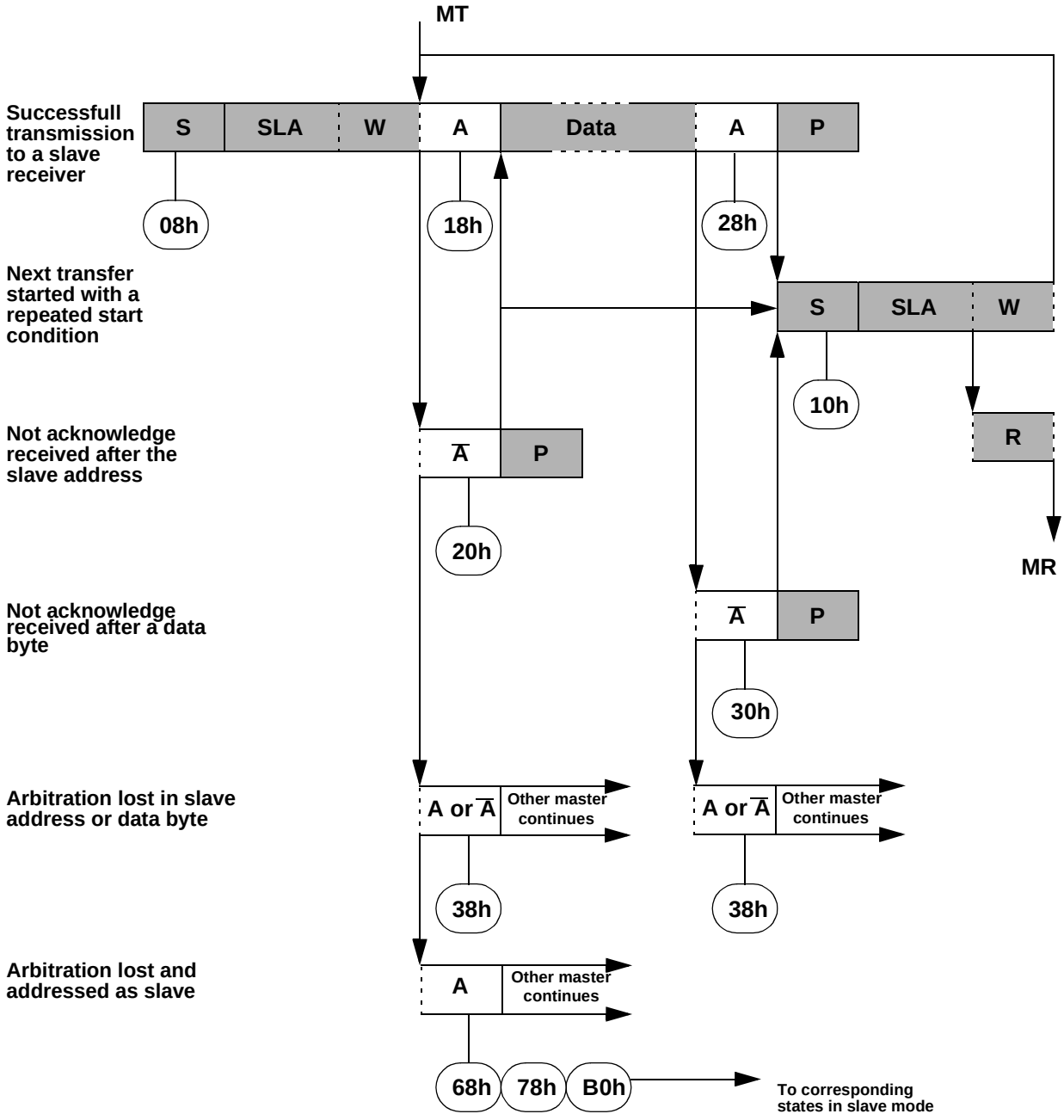
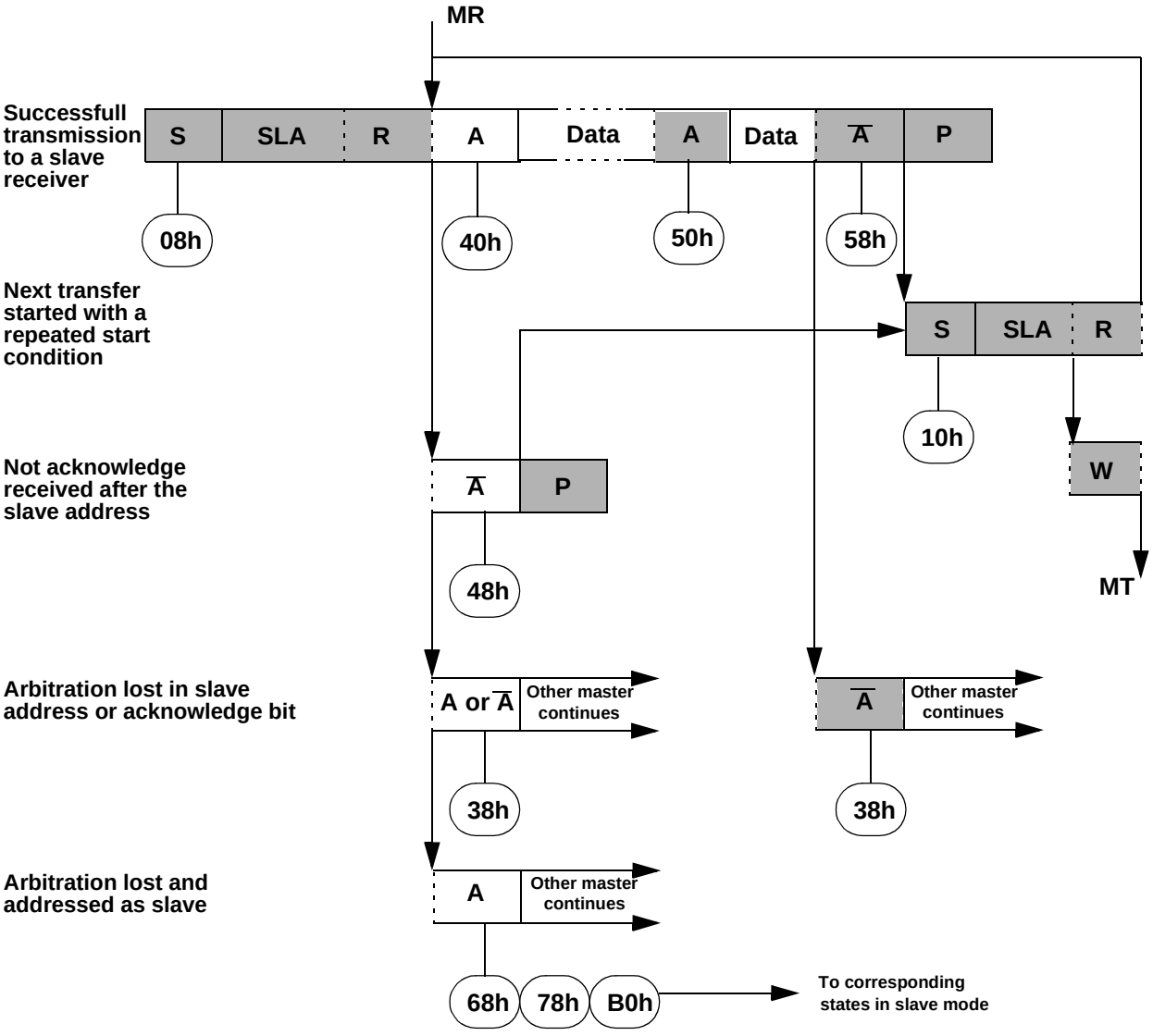


Table 83. Status in Master Transmitter Mode

Status Code SSSTA	Status of the Two-wire Bus and Two-wire Hardware	Application software response					Next Action Taken by Two-wire Hardware
		To/From SSDAT	To SSCON				
			SSSTA	SSSTO	SSI	SSAA	
08h	A START condition has been transmitted	Write SLA+W	X	0	0	X	SLA+W will be transmitted.
10h	A repeated START condition has been transmitted	Write SLA+W	X	0	0	X	SLA+W will be transmitted.
		Write SLA+R	X	0	0	X	SLA+R will be transmitted. Logic will switch to master receiver mode
18h	SLA+W has been transmitted; ACK has been received	Write data byte	0	0	0	X	Data byte will be transmitted.
		No SSDAT action	1	0	0	X	Repeated START will be transmitted.
		No SSDAT action	0	1	0	X	STOP condition will be transmitted and SSSTO flag will be reset.
		No SSDAT action	1	1	0	X	STOP condition followed by a START condition will be transmitted and SSSTO flag will be reset.
20h	SLA+W has been transmitted; NOT ACK has been received	Write data byte	0	0	0	X	Data byte will be transmitted.
		No SSDAT action	1	0	0	X	Repeated START will be transmitted.
		No SSDAT action	0	1	0	X	STOP condition will be transmitted and SSSTO flag will be reset.
		No SSDAT action	1	1	0	X	STOP condition followed by a START condition will be transmitted and SSSTO flag will be reset.
28h	Data byte has been transmitted; ACK has been received	Write data byte	0	0	0	X	Data byte will be transmitted.
		No SSDAT action	1	0	0	X	Repeated START will be transmitted.
		No SSDAT action	0	1	0	X	STOP condition will be transmitted and SSSTO flag will be reset.
		No SSDAT action	1	1	0	X	STOP condition followed by a START condition will be transmitted and SSSTO flag will be reset.
30h	Data byte has been transmitted; NOT ACK has been received	Write data byte	0	0	0	X	Data byte will be transmitted.
		No SSDAT action	1	0	0	X	Repeated START will be transmitted.
		No SSDAT action	0	1	0	X	STOP condition will be transmitted and SSSTO flag will be reset.
		No SSDAT action	1	1	0	X	STOP condition followed by a START condition will be transmitted and SSSTO flag will be reset.
38h	Arbitration lost in SLA+W or data bytes	No SSDAT action	0	0	0	X	Two-wire bus will be released and not addressed slave mode will be entered.
		No SSDAT action	1	0	0	X	A START condition will be transmitted when the bus becomes free.

Figure 50. Format and State in the Master Receiver Mode



From master to slave

From slave to master

Data A — Any number of data bytes and their associated acknowledge bits

n — This number (contained in SSCS) corresponds to a defined state of the 2-wire bus

Table 84. Status in Master Receiver Mode

Status Code SSSTA	Status of the Two-wire Bus and Two-wire Hardware	Application software response					Next Action Taken by Two-wire Hardware
		To/From SSDAT	To SSCON				
			SSSTA	SSSTO	SSI	SSAA	
08h	A START condition has been transmitted	Write SLA+R	X	0	0	X	SLA+R will be transmitted.
10h	A repeated START condition has been transmitted	Write SLA+R	X	0	0	X	SLA+R will be transmitted.
		Write SLA+W	X	0	0	X	SLA+W will be transmitted. Logic will switch to master transmitter mode.
38h	Arbitration lost in SLA+R or NOT ACK bit	No SSDAT action	0	0	0	X	Two-wire bus will be released and not addressed slave mode will be entered.
		No SSDAT action	1	0	0	X	A START condition will be transmitted when the bus becomes free.
40h	SLA+R has been transmitted; ACK has been received	No SSDAT action	0	0	0	0	Data byte will be received and NOT ACK will be returned.
		No SSDAT action	0	0	0	1	Data byte will be received and ACK will be returned.
48h	SLA+R has been transmitted; NOT ACK has been received	No SSDAT action	1	0	0	X	Repeated START will be transmitted.
		No SSDAT action	0	1	0	X	STOP condition will be transmitted and SSSTO flag will be reset.
		No SSDAT action	1	1	0	X	STOP condition followed by a START condition will be transmitted and SSSTO flag will be reset.
50h	Data byte has been received; ACK has been returned	Read data byte	0	0	0	0	Data byte will be received and NOT ACK will be returned.
		Read data byte	0	0	0	1	Data byte will be received and ACK will be returned.
58h	Data byte has been received; NOT ACK has been returned	Read data byte	1	0	0	X	Repeated START will be transmitted.
		Read data byte	0	1	0	X	STOP condition will be transmitted and SSSTO flag will be reset.
		Read data byte	1	1	0	X	STOP condition followed by a START condition will be transmitted and SSSTO flag will be reset.

Figure 51. Format and State in the Slave Receiver Mode

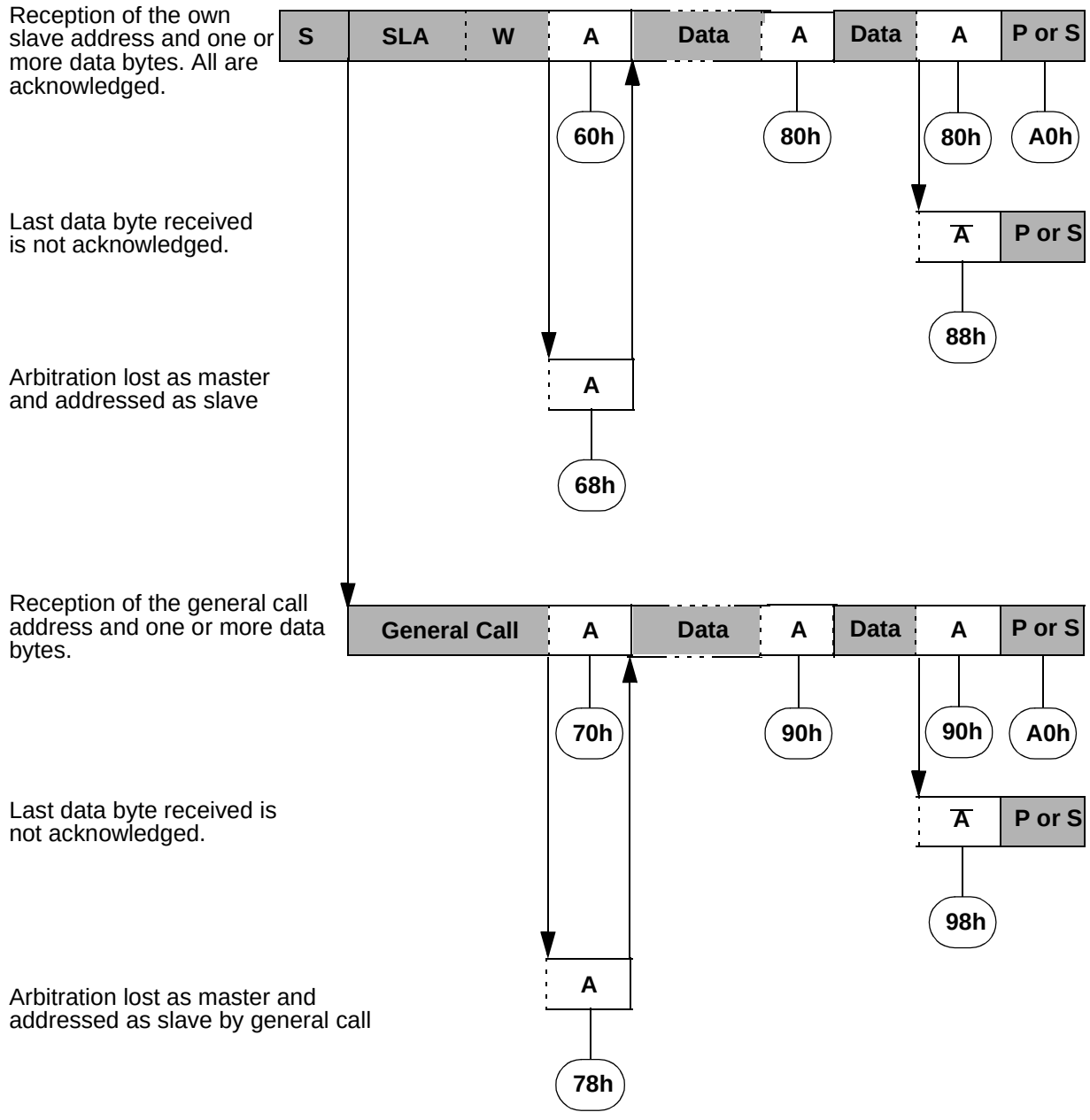


Table 85. Status in Slave Receiver Mode

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

Figure 52. Format and State in the Slave Transmitter Mode

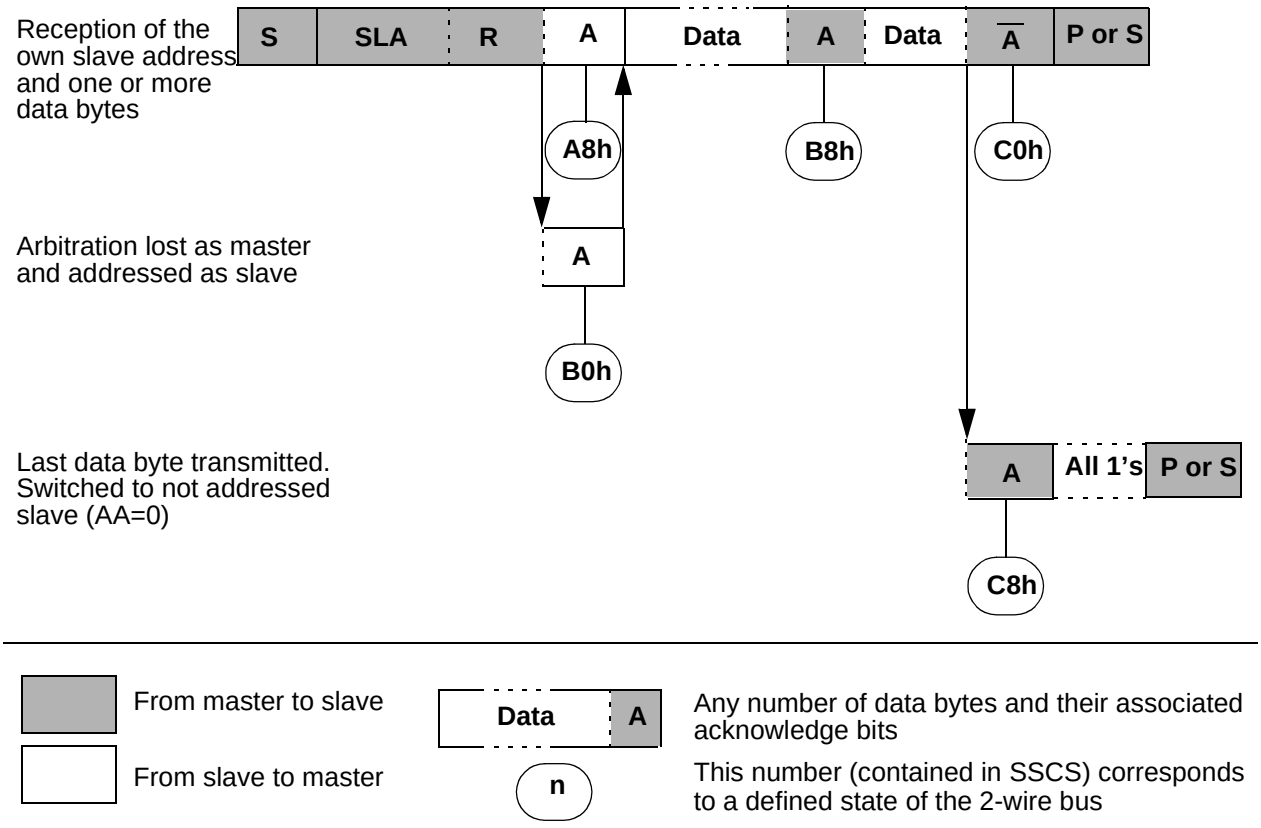


Table 86. Status in Slave Transmitter Mode

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	
A8h	Own SLA+R has been received; ACK has been returned	Load data byte or	X	0	0	0	Last data byte will be transmitted and NOT ACK will be received
		Load data byte	X	0	0	1	Data byte will be transmitted and ACK will be received
B0h	Arbitration lost in SLA+R/W as master; own SLA+R has been received; ACK has been returned	Load data byte or	X	0	0	0	Last data byte will be transmitted and NOT ACK will be received
		Load data byte	X	0	0	1	Data byte will be transmitted and ACK will be received
B8h	Data byte in SSDAT has been transmitted; NOT ACK has been received	Load data byte or	X	0	0	0	Last data byte will be transmitted and NOT ACK will be received
		Load data byte	X	0	0	1	Data byte will be transmitted and ACK will be received

Table 86. Status in Slave Transmitter 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	
C0h	Data byte in SSDAT has been transmitted; NOT ACK has been received	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
C8h	Last data byte in SSDAT has been transmitted (AA=0); ACK has been received	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

Table 87. Miscellaneous Status

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	
F8h	No relevant state information available; SI= 0	No SSDAT action	No SSCON action				Wait or proceed current transfer
00h	Bus error due to an illegal START or STOP condition	No SSDAT action	0	1	0	X	Only the internal hardware is affected, no STOP condition is sent on the bus. In all cases, the bus is released and STO is reset.

Registers

Table 88. SSCON Register

SSCON - Synchronous Serial Control Register (93h)

7	6	5	4	3	2	1	0
CR2	SSIE	STA	STO	SI	AA	CR1	CR0

Bit Number	Bit Mnemonic	Description
7	CR2	**Control Rate bit 2** See .
6	SSIE	**Synchronous Serial Interface Enable bit** Clear to disable SSLC. Set to enable SSLC.
5	STA	**Start flag** Set to send a START condition on the bus.
4	ST0	**Stop flag** Set to send a STOP condition on the bus.
3	SI	**Synchronous Serial Interrupt flag** Set by hardware when a serial interrupt is requested. Must be cleared by software to acknowledge interrupt.
2	AA	**Assert Acknowledge flag** Clear in master and slave receiver modes, to force a not acknowledge (high level on SDA). Clear to disable SLA or GCA recognition. Set to recognise SLA or GCA (if GC set) for entering slave receiver or transmitter modes. Set in master and slave receiver modes, to force an acknowledge (low level on SDA). This bit has no effect when in master transmitter mode.
1	CR1	**Control Rate bit 1** See Table 82
0	CR0	**Control Rate bit 0** See Table 82

Table 89. SSDAT (095h) - Synchronous Serial Data Register (read/write)

SD7	SD6	SD5	SD4	SD3	SD2	SD1	SD0
7	6	5	4	3	2	1	0

Bit Number	Bit Mnemonic	Description
7	SD7	Address bit 7 or Data bit 7.
6	SD6	Address bit 6 or Data bit 6.
5	SD5	Address bit 5 or Data bit 5.
4	SD4	Address bit 4 or Data bit 4.
3	SD3	Address bit 3 or Data bit 3.
2	SD2	Address bit 2 or Data bit 2.

Bit Number	Bit Mnemonic	Description
1	SD1	Address bit 1 or Data bit 1.
0	SD0	Address bit 0 (R/W) or Data bit 0.

Table 90. SSCS (094h) Read - Synchronous Serial Control and Status Register

7	6	5	4	3	2	1	0
SC4	SC3	SC2	SC1	SC0	0	0	0

Bit Number	Bit Mnemonic	Description
0	0	Always zero
1	0	Always zero
2	0	Always zero
3	SC0	**Status Code bit 0** See Table 83 to Table 87
4	SC1	**Status Code bit 1** See Table 83 to Table 87
5	SC2	**Status Code bit 2** See Table 83 to Table 87
6	SC3	**Status Code bit 3** See Table 83 to Table 87
7	SC4	**Status Code bit 4** See Table 83 to Table 87

Table 91. SSADR (096h) - Synchronous Serial Address Register (read/write)

7	6	5	4	3	2	1	0
A7	A6	A5	A4	A3	A2	A1	A0

Bit Number	Bit Mnemonic	Description
7	A7	**Slave address bit 7.**
6	A6	**Slave address bit 6.**
5	A5	**Slave address bit 5.**
4	A4	**Slave address bit 4.**
3	A3	**Slave address bit 3.**
2	A2	**Slave address bit 2.**
1	A1	**Slave address bit 1.**
0	GC	**General call bit** Clear to disable the general call address recognition. Set to enable the general call address recognition.

USB Controller

.

Description

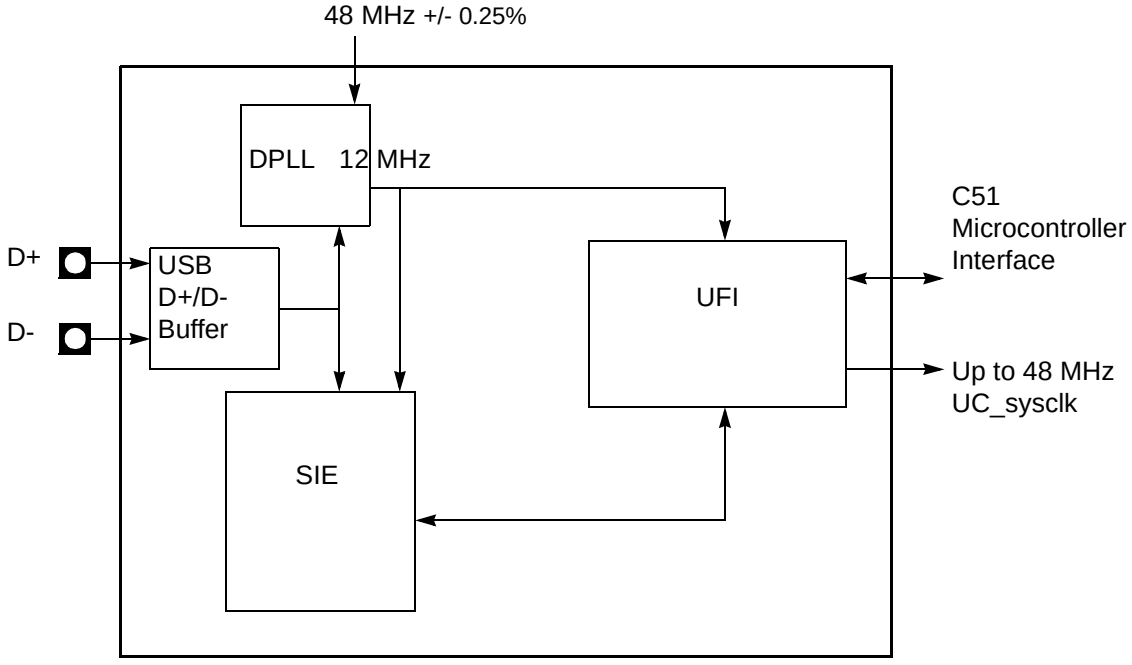
The USB device controller provides the hardware that the AT89C5131 needs to interface a USB link to a data flow stored in a double port memory (DPRAM).

The USB controller requires a 48 MHz ±0.25% reference clock, which is the output of the AT89C5131 PLL (see Section “PLL”, page 14) divided by a clock prescaler. This clock is used to generate a 12 MHz Full-speed bit clock from the received USB differential data and to transmit data according to full speed USB device tolerance. Clock recovery is done by a Digital Phase Locked Loop (DPLL) block, which is compliant with the jitter specification of the USB bus.

The Serial Interface Engine (SIE) block performs NRZI encoding and decoding, bit stuffing, CRC generation and checking, and the serial-parallel data conversion.

The Universal Function Interface (UFI) realizes the interface between the data flow and the Dual Port RAM.

Figure 53. USB Device Controller Block Diagram

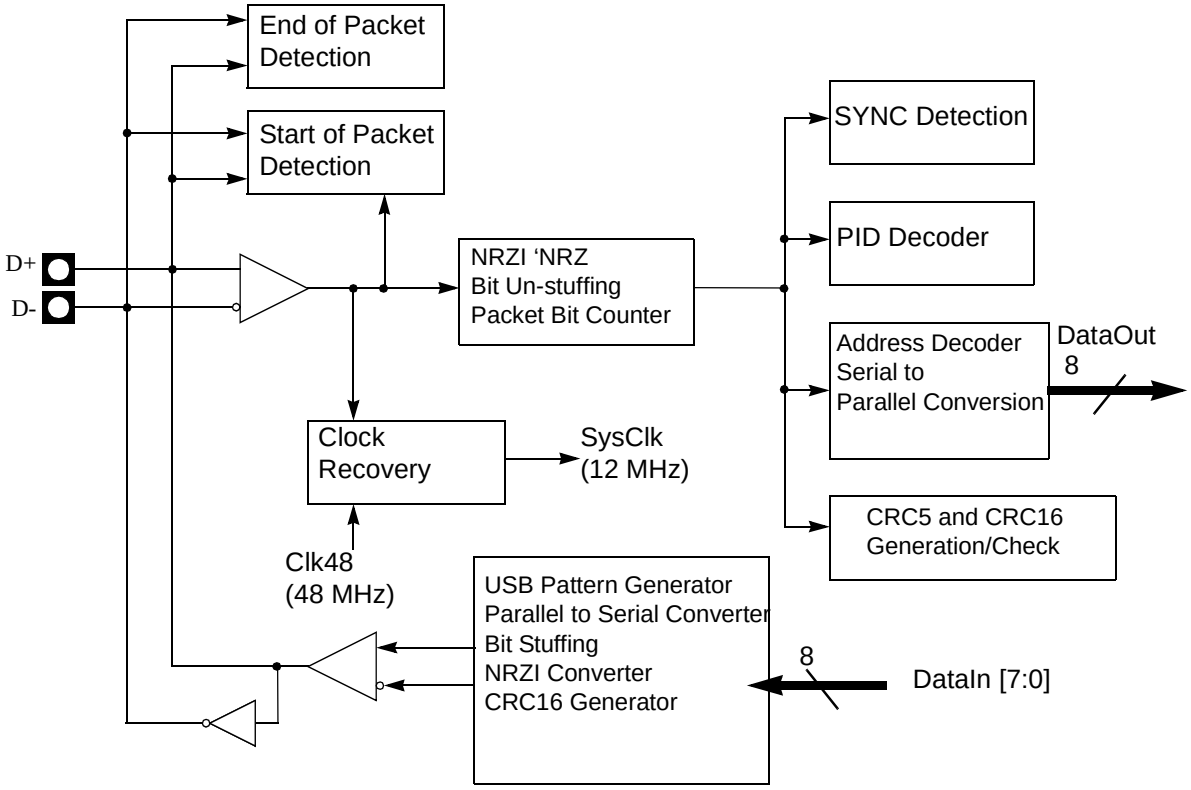


Serial Interface Engine (SIE)

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



Function Interface Unit (FIU) The Function Interface Unit provides the interface between the AT89C5131 and the SIE. It manages transactions at the packet level with minimal intervention from the device firmware, which reads and writes the endpoint FIFOs.

Figure 55. UFI Block Diagram

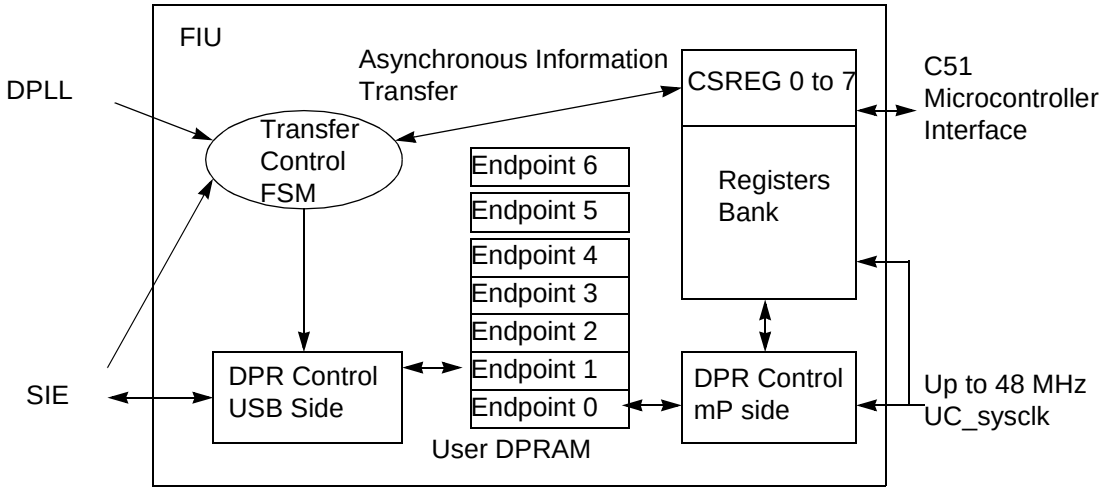
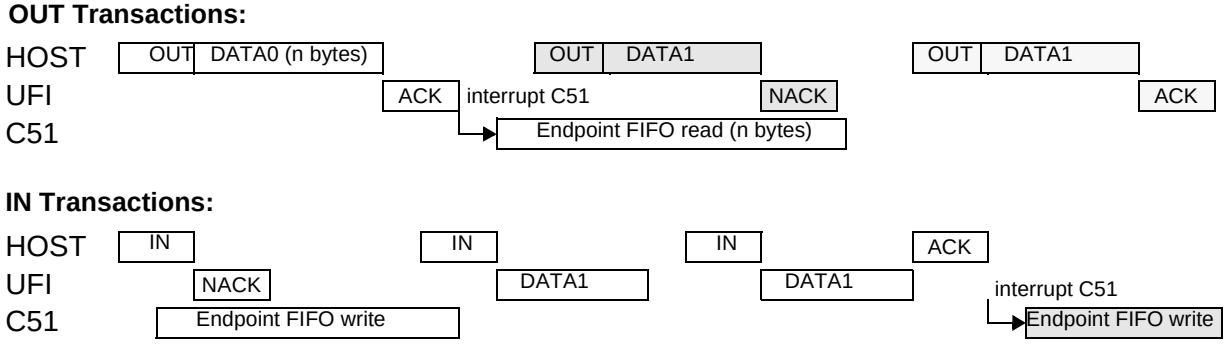


Figure 56. Minimum Intervention from the USB Device Firmware



Configuration

General Configuration

- USB controller enable

  Before any USB transaction, the 48 MHz required by the USB controller must be correctly generated (See “Clock Controller” on page 13.).

  The USB controller will be then enabled by setting the EUSB bit in the USBCON register.

- Set address

  After a Reset or a USB reset, the software has to set the FEN (Function Enable) bit in the USBADDR register. This action will allow the USB controller to answer to the requests sent at the address 0.

  When a SET_ADDRESS request has been received, the USB controller must only answer to the address defined by the request. The new address will be stored in the USBADDR register. The FEN bit and the FADDEN bit in the USBCON register will be set to allow the USB controller to answer only to requests sent at the new address.

- Set configuration

  The CONFG bit in the USBCON register has to be set after a SET_CONFIGURATION request with a non-zero value. Otherwise, this bit has to be cleared.

Endpoint Configuration

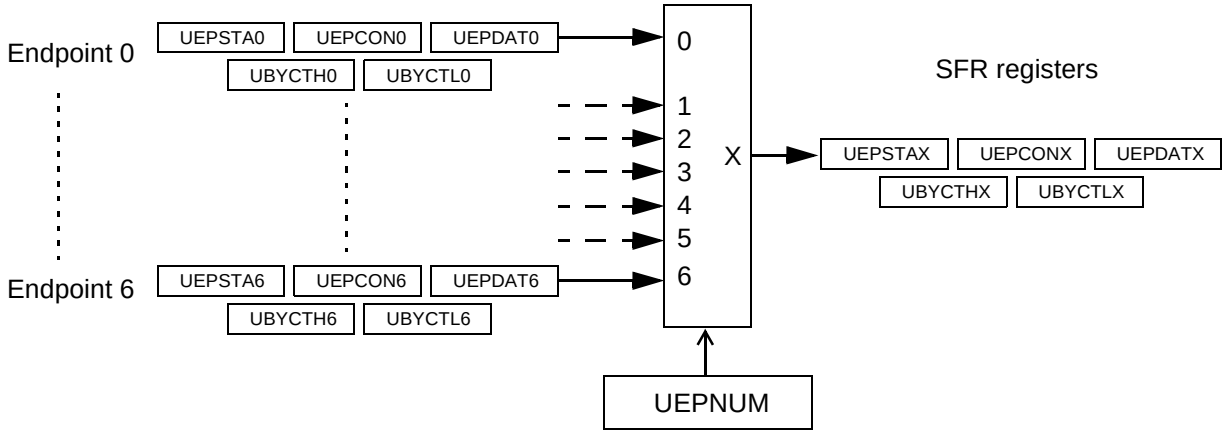
- Selection of an Endpoint

  The endpoint register access is performed using the UEPNUM register. The registers

  – UEPSTAX
  – UEPCONX
  – UEPDATX
  – UBYCTLX
  – UBYCTHX

  These registers correspond to the endpoint whose number is stored in the UEPNUM register. To select an Endpoint, the firmware has to write the endpoint number in the UEPNUM register.

Figure 57. Endpoint Selection



- Endpoint enable

  Before using an endpoint, this one will be enabled by setting the EPEN bit in the UEPCONX register.

  An endpoint which is not enabled won't answer to any USB request. The Default Control Endpoint (Endpoint 0) will always be enabled in order to answer to USB standard requests.

- Endpoint type configuration

  All Standard Endpoints can be configured in Control, Bulk, Interrupt or Isochronous mode. The Ping-pong Endpoints can be configured in Bulk, Interrupt or Isochronous mode. The configuration of an endpoint is performed by setting the field EPTYPE with the following values:

  – Control:EPTYPE = 00b
  – Isochronous:EPTYPE = 01b
  – Bulk:EPTYPE = 10b
  – Interrupt:EPTYPE = 11b

  The Endpoint 0 is the Default Control Endpoint and will always be configured in Control type.

- Endpoint direction configuration

  For Bulk, Interrupt and Isochronous endpoints, the direction is defined with the EPDIR bit of the UEPCONX register with the following values:

  – IN:EPDIR = 1b
  – OUT:EPDIR = 0b

  For Control endpoints, the EPDIR bit has no effect.

- Summary of Endpoint Configuration:

  Do not forget to select the correct endpoint number in the UEPNUM register before accessing to endpoint specific registers.

Table 92. Summary of Endpoint Configuration

Endpoint Configuration	EPEN	EPDIR	EPTYPE	UEPCONX
Disabled	0b	Xb	XXb	0XXX XXXb
Control	1b	Xb	00b	80h
Bulk-in	1b	1b	10b	86h
Bulk-out	1b	0b	10b	82h
Interrupt-In	1b	1b	11b	87h
Interrupt-Out	1b	0b	11b	83h
Isochronous-In	1b	1b	01b	85h
Isochronous-Out	1b	0b	01b	81h

- Endpoint FIFO reset

  Before using an endpoint, its FIFO will be reset. This action resets the FIFO pointer to its original value, resets the byte counter of the endpoint (UBYCTLX and UBYCTHX registers), and resets the data toggle bit (DTGL bit in UEPCONX).

  The reset of an endpoint FIFO is performed by setting to 1 and resetting to 0 the corresponding bit in the UEPRST register.

  For example, in order to reset the Endpoint number 2 FIFO, write 0000 0100b then 0000 0000b in the UEPRST register.

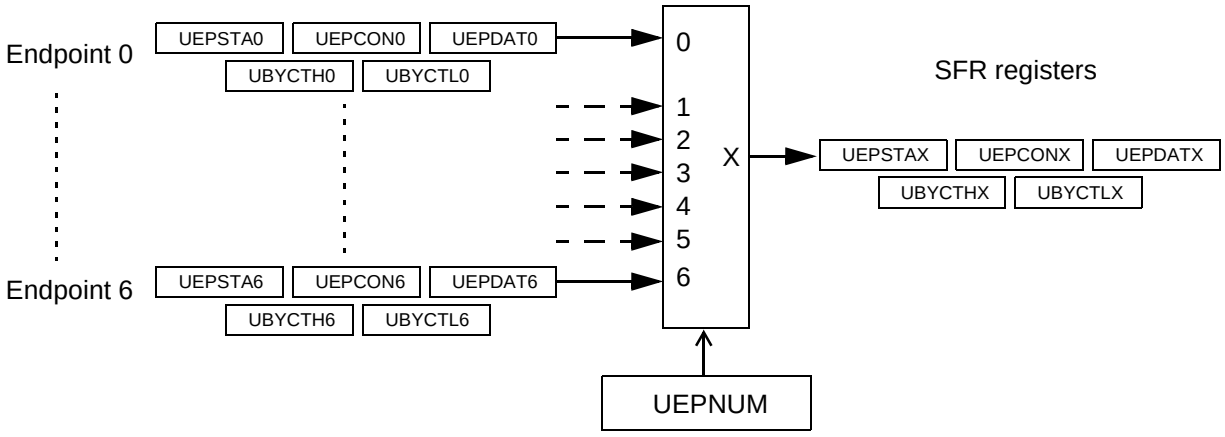
  Note that the endpoint reset doesn't reset the bank number for ping-pong endpoints.

Read/Write Data FIFO

FIFO Mapping

Depending on the selected endpoint through the UEPNUM register, the UEPDATX register allows to access the corresponding endpoint data fifo.

Figure 58. Endpoint FIFO Configuration



Read Data FIFO

The read access for each OUT endpoint is performed using the UEPDATX register.

After a new valid packet has been received on an Endpoint, the data are stored into the FIFO and the byte counter of the endpoint is updated (UBYCTLX and UBYCTHX registers). The firmware has to store the endpoint byte counter before any access to the endpoint FIFO. The byte counter is not updated when reading the FIFO.

To read data from an endpoint, select the correct endpoint number in UEPNUM and read the UEPDATX register. This action automatically decreases the corresponding address vector, and the next data is then available in the UEPDATX register.

Write Data FIFO

The write access for each IN endpoint is performed using the UEPDATX register.

To write a byte into an IN endpoint FIFO, select the correct endpoint number in UEPNUM and write into the UEPDATX register. The corresponding address vector is automatically increased, and another write can be carried out.

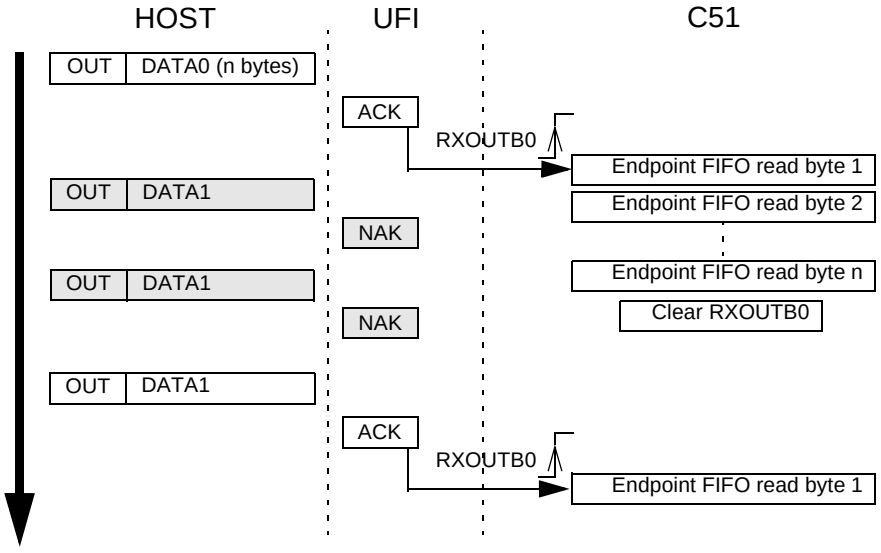
Warning 1: The byte counter is not updated.
Warning 2: Do not write more bytes than supported by the corresponding endpoint.

Bulk/Interrupt Transactions

Bulk and Interrupt transactions are managed in the same way.

Bulk/Interrupt OUT Transactions in Standard Mode

Figure 59. Bulk/Interrupt OUT transactions in Standard Mode



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

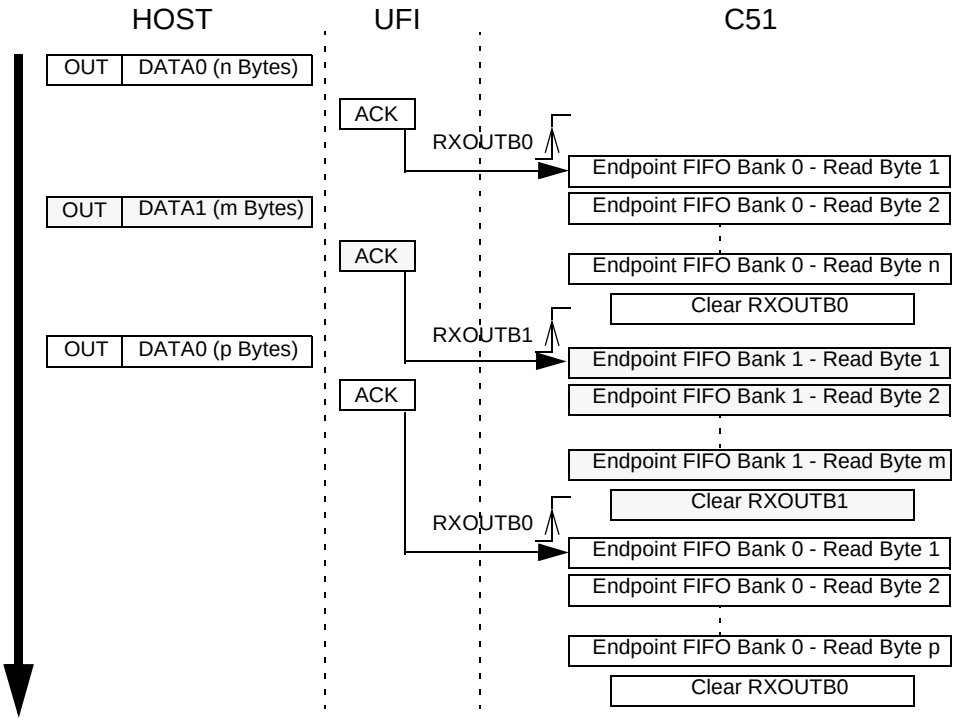
When a valid OUT packet is received on an endpoint, the RXOUTB0 bit is set by the USB controller. This triggers an interrupt if enabled. The firmware has to select the corresponding endpoint, store the number of data bytes by reading the UBYCTLX and UBYCTHX registers. If the received packet is a ZLP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

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

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 and the endpoint byte counter contains the number of bytes sent by the Host.

Bulk/Interrupt OUT Transactions in Ping-pong Mode

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



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.

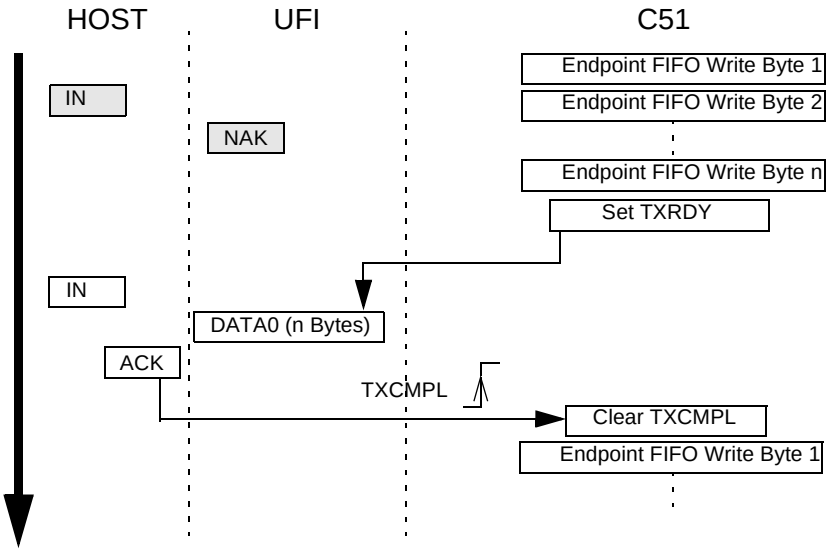
The RXOUTB0 and RXOUTB1 bits are alternatively set by the USB controller at each new valid packet receipt.
The firmware has to clear one of these two bits after having read all the data FIFO to allow a new valid packet to be stored in the corresponding bank.

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.

Bulk/Interrupt IN Transactions in Standard Mode

Figure 61. Bulk/Interrupt IN Transactions in Standard Mode



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

The firmware will fill the FIFO with the data to be sent and set the TXRDY bit in the UEPSTAX register to allow the USB controller to send the data stored in FIFO at the next IN request concerning this endpoint. To send a Zero Length Packet, the firmware will set the TXRDY bit without writing any data into the endpoint FIFO.

Until the TXRDY bit has been set by the firmware, the USB controller will answer a NAK handshake for each IN requests.

To cancel the sending of this packet, the firmware has to reset the TXRDY bit. The packet stored in the endpoint FIFO is then cleared and a new packet can be written and sent.

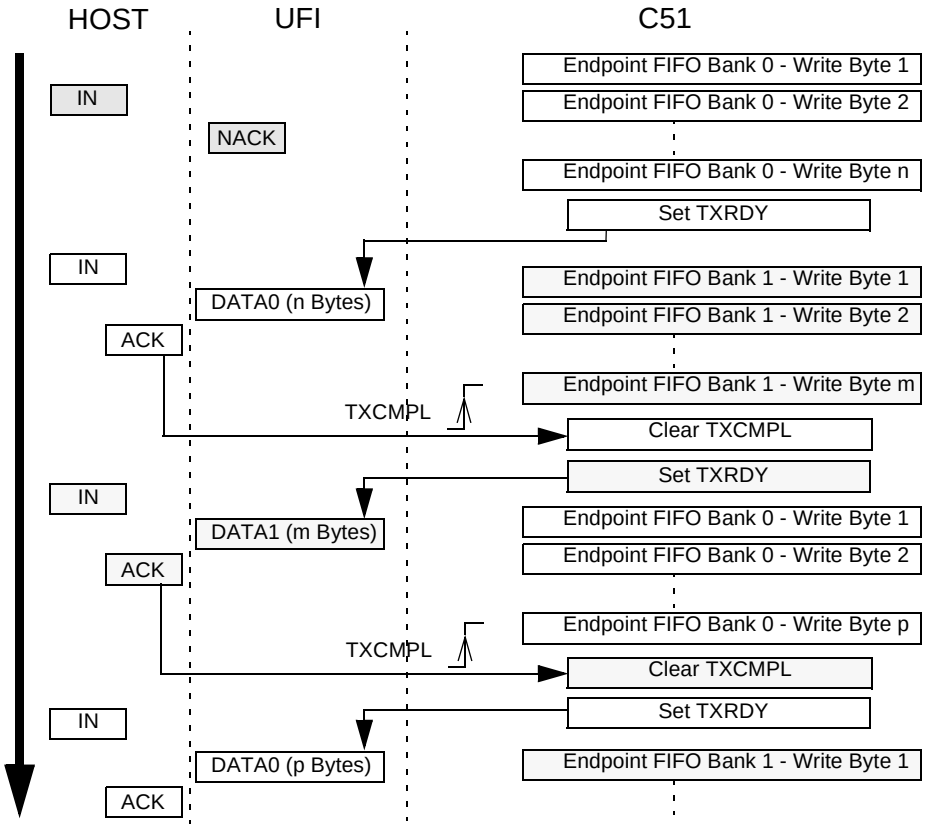
When the IN packet has been sent and acknowledged by the Host, the TXCMPL bit in the UEPSTAX register is set by the USB controller. This triggers a USB interrupt if enabled. The firmware will clear the TXCMPL bit before filling the endpoint FIFO with new data.

The firmware will never write more bytes than supported by the endpoint FIFO.

All USB retry mechanisms are automatically managed by the USB controller.

Bulk/Interrupt IN Transactions in Ping-pong Mode

Figure 62. Bulk/Interrupt IN Transactions in Ping-pong Mode



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

The firmware will fill the FIFO bank 0 with the data to be sent and set the TXRDY bit in the UEPSTAX register to allow the USB controller to send the data stored in FIFO at the next IN request concerning the endpoint. The FIFO banks are automatically switched, and the firmware can immediately write into the endpoint FIFO bank 1.

When the IN packet concerning the bank 0 has been sent and acknowledged by the Host, the TXCMPL bit is set by the USB controller. This triggers a USB interrupt if enabled. The firmware will clear the TXCMPL bit before filling the endpoint FIFO bank 0 with new data. The FIFO banks are then automatically switched.

When the IN packet concerning the bank 1 has been sent and acknowledged by the Host, the TXCMPL bit is set by the USB controller. This triggers a USB interrupt if enabled. The firmware will clear the TXCMPL bit before filling the endpoint FIFO bank 1 with new data.

The bank switch is performed by the USB controller each time the TXRDY bit is set by the firmware. Until the TXRDY bit has been set by the firmware for an endpoint bank, the USB controller will answer a NAK handshake for each IN requests concerning this bank.

Note that in the example above, the firmware clears the Transmit Complete bit (TXCMPL) before setting the Transmit Ready bit (TXRDY). This is done in order to avoid the firmware to clear at the same time the TXCMPL bit for bank 0 and the bank 1.

The firmware will never write more bytes than supported by the endpoint FIFO.

Control Transactions

Setup Stage

The DIR bit in the UEPSTAX register will be at 0.

Receiving Setup packets is the same as receiving Bulk Out packets, except that the RXSETUP bit in the UEPSTAX register is set by the USB controller instead of the RXOUTB0 bit to indicate that an Out packet with a Setup PID has been received on the Control endpoint. When the RXSETUP bit has been set, all the other bits of the UEPSTAX register are cleared and an interrupt is triggered if enabled.

The firmware has to read the Setup request stored in the Control endpoint FIFO before clearing the RXSETUP bit to free the endpoint FIFO for the next transaction.

Data Stage: Control Endpoint Direction

The data stage management is similar to Bulk management.

A Control endpoint is managed by the USB controller as a full-duplex endpoint: IN and OUT. All other endpoint types are managed as half-duplex endpoint: IN or OUT. The firmware has to specify the control endpoint direction for the data stage using the DIR bit in the UEPSTAX register.
The firmware has to use the DIR bit before data IN in order to meet the data-toggle requirements:

- If the data stage consists of INs,
  the firmware has to set the DIR bit in the UEPSTAX register before writing into the FIFO and sending the data by setting to 1 the TXRDY bit in the UEPSTAX register. The IN transaction is complete when the TXCMPL has been set by the hardware. The firmware will clear the TXCMPL bit before any other transaction.
- If the data stage consists of OUTs,
  the firmware has to leave the DIR bit at 0. The RXOUTB0 bit is set by hardware when a new valid packet has been received on the endpoint. The firmware must read the data stored into the FIFO and then clear the RXOUTB0 bit to reset the FIFO and to allow the next transaction.

To send a STALL handshake, see “STALL Handshake” on page 130.

Status Stage

The DIR bit in the UEPSTAX register will be reset at 0 for IN and OUT status stage.

The status stage management is similar to Bulk management.

- For a Control Write transaction or a No-Data Control transaction, the status stage consists of a IN Zero Length Packet (see “Bulk/Interrupt IN Transactions in Standard Mode” on page 125). To send a STALL handshake, see “STALL Handshake” on page 130.
- For a Control Read transaction, the status stage consists of a OUT Zero Length Packet (see “Bulk/Interrupt OUT Transactions in Standard Mode” on page 123).

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.

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 IN Transactions in Standard Mode

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

The firmware will fill the FIFO with the data to be sent and set the TXRDY bit in the UEPSTAX register to allow the USB controller to send the data stored in FIFO at the next IN request concerning this endpoint.
If the TXRDY bit is not set when the IN request occurs, nothing will be sent by the USB controller.

When the IN packet has been sent, the TXCMPL bit in the UEPSTAX register is set by the USB controller. This triggers a USB interrupt if enabled. The firmware will clear the TXCMPL bit before filling the endpoint FIFO with new data.

The firmware will never write more bytes than supported by the endpoint FIFO

Isochronous IN Transactions in Ping-pong Mode

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

The firmware will fill the FIFO bank 0 with the data to be sent and set the TXRDY bit in the UEPSTAX register to allow the USB controller to send the data stored in FIFO at the next IN request concerning the endpoint. The FIFO banks are automatically switched, and the firmware can immediately write into the endpoint FIFO bank 1.
If the TXRDY bit is not set when the IN request occurs, nothing will be sent by the USB controller.

When the IN packet concerning the bank 0 has been sent, the TXCMPL bit is set by the USB controller. This triggers a USB interrupt if enabled. The firmware will clear the TXCMPL bit before filling the endpoint FIFO bank 0 with new data. The FIFO banks are then automatically switched.

When the IN packet concerning the bank 1 has been sent, the TXCMPL bit is set by the USB controller. This triggers a USB interrupt if enabled. The firmware will clear the TXCMPL bit before filling the endpoint FIFO bank 1 with new data.

The bank switch is performed by the USB controller each time the TXRDY bit is set by the firmware. Until the TXRDY bit has been set by the firmware for an endpoint bank, the USB controller won't send anything at each IN requests concerning this bank.

The firmware will never write more bytes than supported by the endpoint FIFO.

Miscellaneous

USB Reset

The EORINT bit in the USBINT register is set by hardware when a End Of Reset has been detected on the USB bus. This triggers a USB interrupt if enabled. The USB controller is still enabled, but all the USB registers are reset by hardware. The firmware will clear the EORINT bit to allow the next USB reset detection.

STALL Handshake

This function is only available for Control, Bulk, and Interrupt endpoints.

The firmware has to set the STALLRQ bit in the UEPSTAX register to send a STALL handshake at the next request of the Host on the endpoint selected with the UEPNUM register. The RXSETUP, TXRDY, TXCMPL, RXOUTB0 and RXOUTB1 bits must be first reset to 0. The bit STLCRC is set at 1 by the USB controller when a STALL has been sent. This triggers an interrupt if enabled.
The firmware will clear the STALLRQ and STLCRC bits after each STALL sent.

The STALLRQ bit is cleared automatically by hardware when a valid SETUP PID is received on a CONTROL type endpoint.

Important note: when a Clear Halt Feature occurs for an endpoint, the firmware will reset this endpoint using the UEPRST register in order to reset the data toggle management.

Start of Frame Detection

The SOFINT bit in the USBINT register is set when the USB controller detects a Start of Frame PID. This triggers an interrupt if enabled. The firmware will clear the SOFINT bit to allow the next Start of Frame detection.

Frame Number

When receiving a Start of Frame, the frame number is automatically stored in the UFNUML and UFNUMH registers. The CRCOK and CRCERR bits indicate if the CRC of the last Start of Frame is valid (CRCOK set at 1) or corrupted (CRCERR set at 1). The UFNUML and UFNUMH registers are automatically updated when receiving a new Start of Frame.

Data Toggle Bit

The Data Toggle bit is set by hardware when a DATA0 packet is received and accepted by the USB controller and cleared by hardware when a DATA1 packet is received and accepted by the USB controller. This bit is reset when the firmware resets the endpoint FIFO using the UEPRST register.

For Control endpoints, each SETUP transaction starts with a DATA0 and data toggling is then used as for Bulk endpoints until the end of the Data stage (for a control write transfer). The Status stage completes the data transfer with a DATA1 (for a control read transfer).

For Isochronous endpoints, the device firmware will ignore the data-toggle.

Suspend/Resume Management

Suspend

The Suspend state can be detected by the USB controller if all the clocks are enabled and if the USB controller is enabled. The bit SPINT is set by hardware when an idle state is detected for more than 3 ms. This triggers a USB interrupt if enabled.

In order to reduce current consumption, the firmware can put the USB PAD in idle mode, stop the clocks and put the C51 in Idle or Power-down mode. The Resume detection is still active.
The USB PAD is put in idle mode when the firmware clear the SPINT bit. In order to avoid a new suspend detection 3ms later, the firmware has to disable the USB clock input using the SUSPCLK bit in the USBCON Register. The USB PAD automatically exits of idle mode when a wake-up event is detected.

The stop of the 48 MHz clock from the PLL should be done in the following order:

1. Clear suspend interrupt bit in USBINT (required to allow the USB pads to enter power down mode).
2. Enable USB resume interrupt.
3. Disable of the 48 MHz clock input of the USB controller by setting to 1 the SUSPCLK bit in the USBCON register.
4. Disable the PLL by clearing the PLLEN bit in the PLLCON register.
5. Make the CPU core enter power down mode by setting PDOWN bit in PCON.

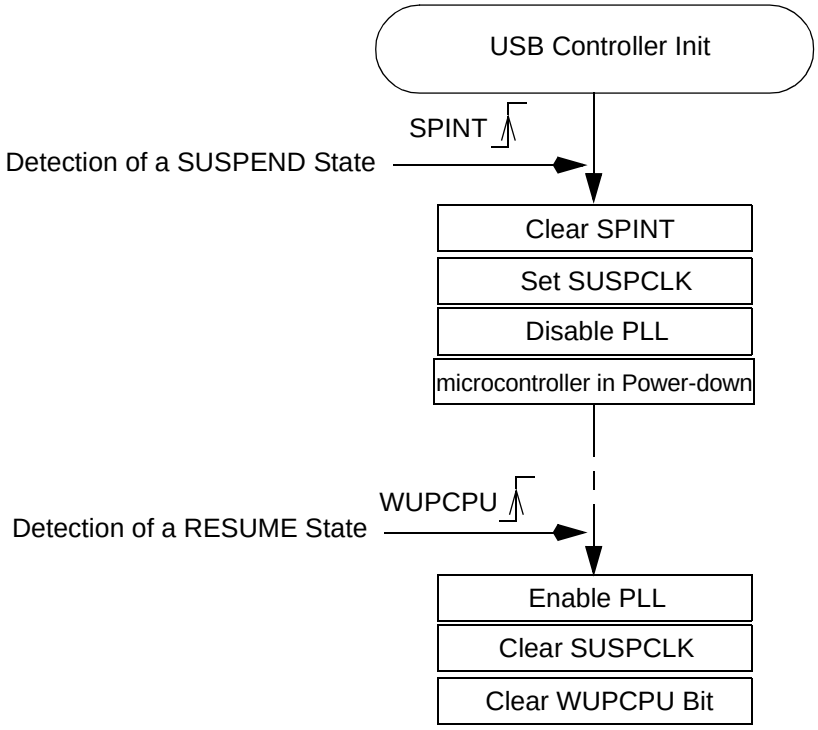
Resume

When the USB controller is in Suspend state, the Resume detection is active even if all the clocks are disabled and if the C51 is in Idle or Power-down mode. The WUPCPU bit is set by hardware when a non-idle state occurs on the USB bus. This triggers an interrupt if enabled. This interrupt wakes up the CPU from its Idle or Power-down state and the interrupt function is then executed. The firmware will first enable the 48 MHz generation and then reset to 0 the SUSPCLK bit in the USBCON register if needed.

The firmware has to clear the SPINT bit in the USBINT register before any other USB operation in order to wake up the USB controller from its Suspend mode.

The USB controller is then re-activated.

Figure 63. Example of a Suspend/Resume Management



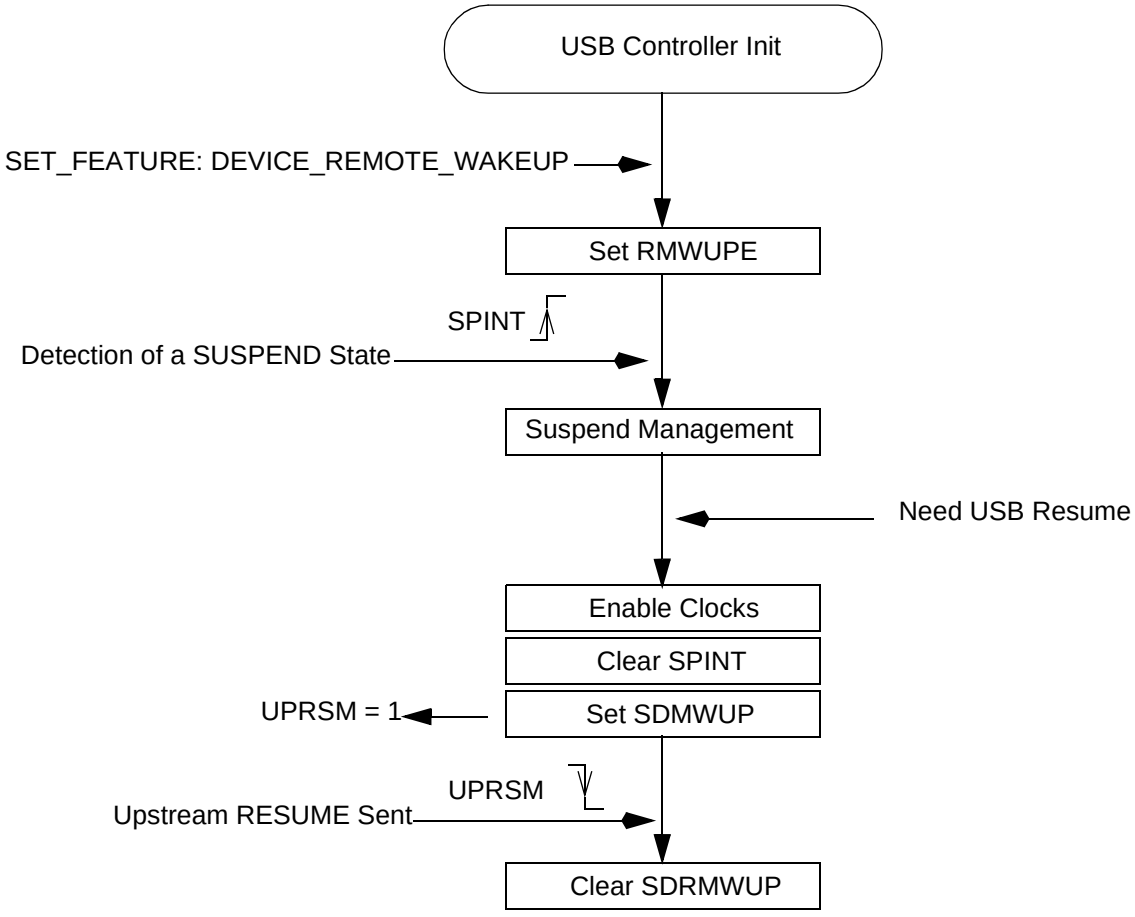
Upstream Resume

A USB device can be allowed by the Host to send an upstream resume for Remote Wake Up purpose.

When the USB controller receives the SET_FEATURE request: DEVICE_REMOTE_WAKEUP, the firmware will set to 1 the RMWUPE bit in the USBCON register to enable this functionality. RMWUPE value will be 0 in the other cases.

If the device is in SUSPEND mode, the USB controller can send an upstream resume by clearing first the SPINT bit in the USBINT register and by setting then to 1 the SDRMWUP bit in the USBCON register. The USB controller sets to 1 the UPRSM bit in the USBCON register. All clocks must be enabled first. The Remote Wake is sent only if the USB bus was in Suspend state for at least 5 ms. When the upstream resume is completed, the UPRSM bit is reset to 0 by hardware. The firmware will then clear the SDRMWUP bit.

Figure 64. Example of REMOTE WAKEUP Management



Detach Simulation

In order to be re-enumerated by the Host, the AT89C5130A/31A-M has the possibility to simulate a DETACH - ATTACH of the USB bus.

The V_{REF} output voltage is between 3.0V and 3.6V. This output can be connected to the D+ pull-up as shown in Figure 65. This output can be put in high-impedance when the DETACH bit is set to 1 in the USBCON register. Maintaining this output in high impedance for more than 3 µs will simulate the disconnection of the device. When resetting the DETACH bit, an attach is then simulated.

Figure 65. Example of V_{REF} Connection

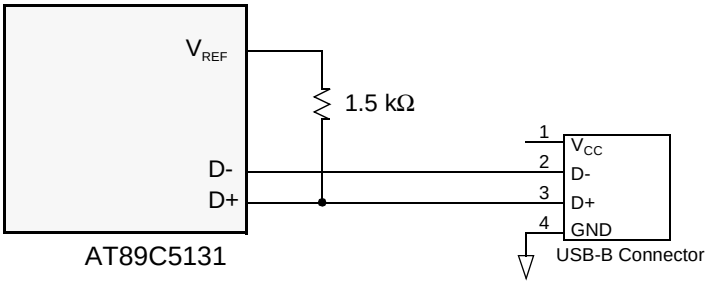
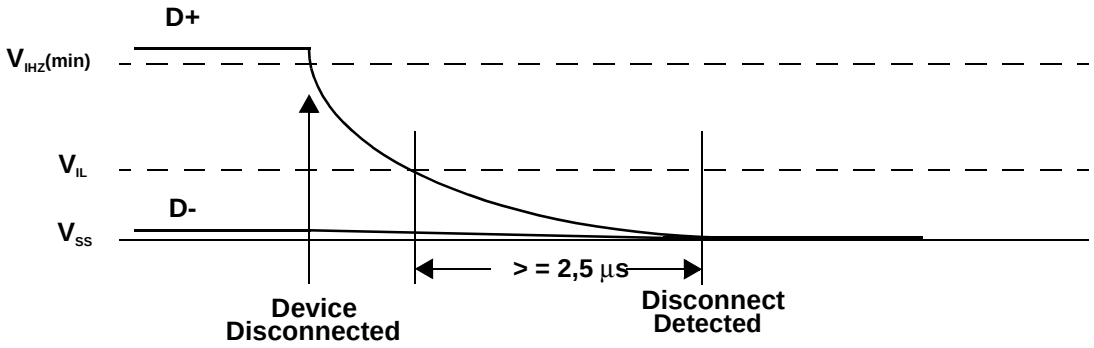


Figure 66. Disconnect Timing



USB Interrupt System

Interrupt System Priorities

Figure 67. USB Interrupt Control System

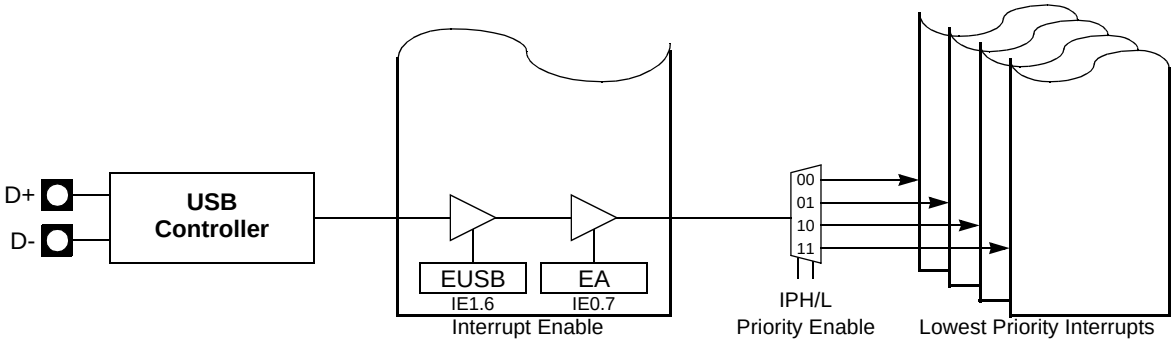


Table 93. Priority Levels

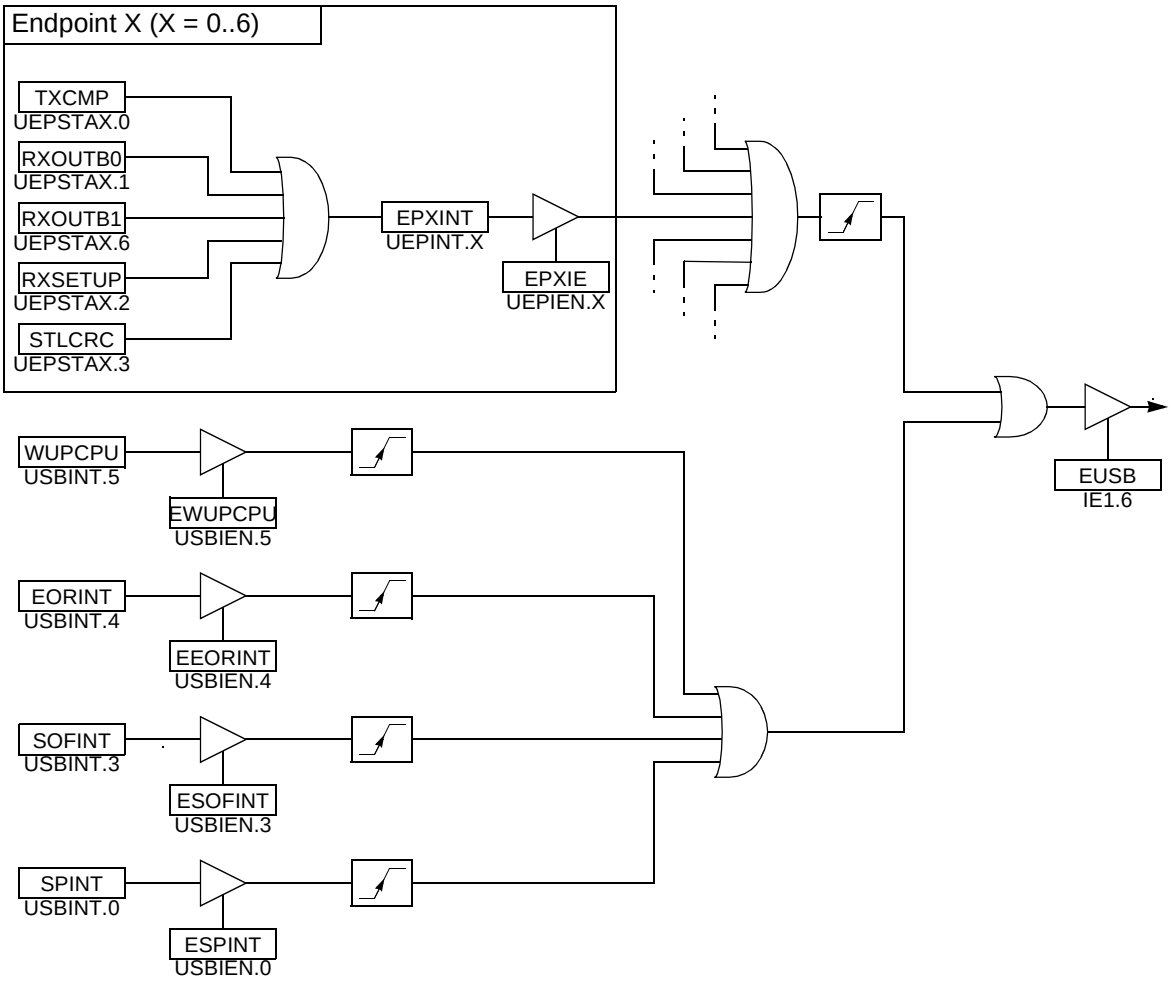
IPHUSB	IPLUSB	USB Priority Level
0	0	0 Lowest
0	1	1
1	0	2
1	1	3 Highest

USB Interrupt Control System

As shown in Figure 68, many events can produce a USB interrupt:

- TXCMPL: Transmitted In Data (see Table 100 on page 142). This bit is set by hardware when the Host accept a In packet.
- RXOUTB0: Received Out Data Bank 0 (see Table 100 on page 142). This bit is set by hardware when an Out packet is accepted by the endpoint and stored in bank 0.
- RXOUTB1: Received Out Data Bank 1 (only for Ping-pong endpoints) (see Table 100 on page 142). This bit is set by hardware when an Out packet is accepted by the endpoint and stored in bank 1.
- RXSETUP: Received Setup (see Table 100 on page 142). This bit is set by hardware when an SETUP packet is accepted by the endpoint.
- STLCRC: STALLED (only for Control, Bulk and Interrupt endpoints) (see Table 100 on page 142). This bit is set by hardware when a STALL handshake has been sent as requested by STALLRQ, and is reset by hardware when a SETUP packet is received.
- SOFINT: Start of Frame Interrupt (See “USBIEN Register USBIEN (S:BEh) USB Global Interrupt Enable Register” on page 139.). This bit is set by hardware when a USB Start of Frame packet has been received.
- WUPCPU: Wake-Up CPU Interrupt (See “USBIEN Register USBIEN (S:BEh) USB Global Interrupt Enable Register” on page 139.). This bit is set by hardware when a USB resume is detected on the USB bus, after a SUSPEND state.
- SPINT: Suspend Interrupt (See “USBIEN Register USBIEN (S:BEh) USB Global Interrupt Enable Register” on page 139.). This bit is set by hardware when a USB suspend is detected on the USB bus.

Figure 68. USB Interrupt Control Block Diagram



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 95. USBINT Register
USBINT (S:BDh)
USB Global Interrupt Register

7	6	5	4	3	2	1	0
-	-	WUPCPU	EORINT	SOFINT	-	-	SPINT

Bit Number	Bit Mnemonic	Description
7-6	-	**Reserved** The value read from these bits is always 0. Do not set these bits.
5	WUPCPU	**Wake Up CPU Interrupt** This bit is set by hardware when the USB controller is in SUSPEND state and is re-activated by a non-idle signal FROM USB line (not by an upstream resume). This triggers a USB interrupt when EWUPCPU is set in Figure 96 on page 139. When receiving this interrupt, user has to enable all USB clock inputs. This bit will be cleared by software (USB clocks must be enabled before).
4	EORINT	**End Of Reset Interrupt** This bit is set by hardware when a End Of Reset has been detected by the USB controller. This triggers a USB interrupt when EEORINT is set (see Table 96 on page 139). This bit will be cleared by software.
3	SOFINT	**Start of Frame Interrupt** This bit is set by hardware when an USB Start of Frame PID (SOF) has been detected. This triggers a USB interrupt when ESOFINT is set (see Table 96 on page 139). This bit will be cleared by software.
2	-	**Reserved** The value read from this bit is always 0. Do not set this bit.
1	-	**Reserved** The value read from this bit is always 0. Do not set this bit.
0	SPINT	**Suspend Interrupt** This bit is set by hardware when a USB Suspend (Idle bus for three frame periods: a J state for 3 ms) is detected. This triggers a USB interrupt when ESPINT is set in see Table 96 on page 139. This bit will be cleared by software BEFORE any other USB operation to re-activate the macro.

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	EEOFINT	**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 99. UEPCONX Register
UEPCONX (S:D4h)
USB Endpoint X Control Register

7	6	5	4	3	2	1	0
EPEN	**-**	**-**	**-**	**DTGL**	**EPDIR**	**EPTYPE1**	**EPTYPE0**

Bit Number	Bit Mnemonic	Description
7	EPEN	**Endpoint Enable** Set this bit to enable the endpoint according to the device configuration. Endpoint 0 will always be enabled after a hardware or USB bus reset and participate in the device configuration. Clear this bit to disable the endpoint according to the device configuration.
6	-	**Reserved** The value read from this bit is always 0. Do not set this bit.
5	-	**Reserved** The value read from this bit is always 0. Do not set this bit.
4	-	**Reserved** The value read from this bit is always 0. Do not set this bit.
3	DTGL	**Data Toggle (Read-only)** This bit is set by hardware when a valid DATA0 packet is received and accepted. This bit is cleared by hardware when a valid DATA1 packet is received and accepted.
2	EPDIR	**Endpoint Direction** Set this bit to configure IN direction for Bulk, Interrupt and Isochronous endpoints. Clear this bit to configure OUT direction for Bulk, Interrupt and Isochronous endpoints. This bit has no effect for Control endpoints.
1-0	EPTYPE[1:0]	**Endpoint Type** Set this field according to the endpoint configuration (Endpoint 0 will always be configured as control): 00Control endpoint 01Isochronous endpoint 10Bulk endpoint 11Interrupt endpoint

Note: 1. (X = EPNUM set in UEPNUM Register UEPNUM (S:C7h) USB Endpoint Number)

Reset Value = 80h when UEPNUM = 0 (default Control Endpoint)
Reset Value = 00h otherwise for all other endpoints

Table 100. UEPSTAX (S:CEh) USB Endpoint X Status Register

7	6	5	4	3	2	1	0
DIR	RXOUTB1	STALLRQ	TXRDY	STL/CRC	RXSETUP	RXOUTB0	TXCMP

Bit Number	Bit Mnemonic	Description
7	DIR	**Control Endpoint Direction** This bit is used only if the endpoint is configured in the control type (seeSection “UEPCONX Register UEPCONX (S:D4h) USB Endpoint X Control Register”). This bit determines the Control data and status direction. The device firmware will set this bit ONLY for the IN data stage, before any other USB operation. Otherwise, the device firmware will clear this bit.
6	RXOUTB1	**Received OUT Data Bank 1 for Endpoints 4, 5 and 6 (Ping-pong mode)** This bit is set by hardware after a new packet has been stored in the endpoint FIFO data bank 1 (only in Ping-pong mode). Then, the endpoint interrupt is triggered if enabled (see“UEPINT Register UEPINT (S:F8h read-only) USB Endpoint Interrupt Register” on page 146) and all the following OUT packets to the endpoint bank 1 are rejected (NAK’ed) until this bit has been cleared, excepted for Isochronous Endpoints. This bit will be cleared by the device firmware after reading the OUT data from the endpoint FIFO.
5	STALLRQ	**Stall Handshake Request** Set this bit to request a STALL answer to the host for the next handshake.Clear this bit otherwise. For CONTROL endpoints: cleared by hardware when a valid SETUP PID is received.
4	TXRDY	**TX Packet Ready** Set this bit after a packet has been written into the endpoint FIFO for IN data transfers. Data will be written into the endpoint FIFO only after this bit has been cleared. Set this bit without writing data to the endpoint FIFO to send a Zero Length Packet. This bit is cleared by hardware, as soon as the packet has been sent for Isochronous endpoints, or after the host has acknowledged the packet for Control, Bulk and Interrupt endpoints. When this bit is cleared, the endpoint interrupt is triggered if enabled (see“UEPINT Register UEPINT (S:F8h read-only) USB Endpoint Interrupt Register” on page 146).
3	STLCRC	**Stall Sent/CRC error flag** - For Control, Bulk and Interrupt Endpoints: This bit is set by hardware after a STALL handshake has been sent as requested by STALLRQ. Then, the endpoint interrupt is triggered if enabled (see“UEPINT Register UEPINT (S:F8h read-only) USB Endpoint Interrupt Register” on page 146) It will be cleared by the device firmware. - For Isochronous Endpoints **(Read-Only)**: This bit is set by hardware if the last received data is corrupted (CRC error on data). This bit is updated by hardware when a new data is received.
2	RXSETUP	**Received SETUP** This bit is set by hardware when a valid SETUP packet has been received from the host. Then, all the other bits of the register are cleared by hardware and the endpoint interrupt is triggered if enabled (see“UEPINT Register UEPINT (S:F8h read-only) USB Endpoint Interrupt Register” on page 146). It will be cleared by the device firmware after reading the SETUP data from the endpoint FIFO.
1	RXOUTB0	**Received OUT Data Bank 0** (see also RXOUTB1 bit for Ping-pong Endpoints) This bit is set by hardware after a new packet has been stored in the endpoint FIFO data bank 0. Then, the endpoint interrupt is triggered if enabled (see“UEPINT Register UEPINT (S:F8h read-only) USB Endpoint Interrupt Register” on page 146) and all the following OUT packets to the endpoint bank 0 are rejected (NAK’ed) until this bit has been cleared, excepted for Isochronous Endpoints. However, for control endpoints, an early SETUP transaction may overwrite the content of the endpoint FIFO, even if its Data packet is received while this bit is set. This bit will be cleared by the device firmware after reading the OUT data from the endpoint FIFO.
0	TXCMPL	**Transmitted IN Data Complete** This bit is set by hardware after an IN packet has been transmitted for Isochronous endpoints and after it has been accepted (ACK’ed) by the host for Control, Bulk and Interrupt endpoints. Then, the endpoint interrupt is triggered if enabled (see“UEPINT Register UEPINT (S:F8h read-only) USB Endpoint Interrupt Register” on page 146). This bit will be cleared by the device firmware before setting TXRDY.

Reset Value = 00h

Table 101. UEPDATX Register
UEPDATX (S:CFh)
USB FIFO Data Endpoint X (X = EPNUM set in UEPNUM Register UEPNUM (S:C7h)

7	6	5	4	3	2	1	0
FDAT7	FDAT6	FDAT5	FDAT4	FDAT3	FDAT2	FDAT1	FDAT0

Bit Number	Bit Mnemonic	Description
7 - 0	FDAT[7:0]	**Endpoint X FIFO data** Data byte to be written to FIFO or data byte to be read from the FIFO, for the Endpoint X (see EPNUM).

USB Endpoint Number)

Reset Value = XXh

Table 102. UBYCTLX Register
UBYCTLX (S:E2h)
USB Byte Count Low Register X (X = EPNUM set in UEPNUM Register UEPNUM

7	6	5	4	3	2	1	0
BYCT7	BYCT6	BYCT5	BYCT4	BYCT3	BYCT2	BYCT1	BYCT0

Bit Number	Bit Mnemonic	Description
7 - 0	BYCT[7:0]	**Byte Count LSB** Least Significant Byte of the byte count of a received data packet. The most significant part is provided by the UBYCTHX Register UBYCTHX (S:E3h) USB Byte Count High Register X (X = EPNUM set in UEPNUM Register UEPNUM (S:C7h) USB Endpoint Number) (see Figure 102 on page 143). This byte count is equal to the number of data bytes received after the Data PID.

(S:C7h) USB Endpoint Number)

Reset Value = 00h

Table 103. UBYCTHX Register
UBYCTHX (S:E3h)
USB Byte Count High Register X (X = EPNUM set in UEPNUM Register UEPNUM

7	6	5	4	3	2	1	0
-	-	-	-	-	-	BYCT9	BYCT8

Bit Number	Bit Mnemonic	Description
7-2	-	**Reserved** The value read from these bits is always 0. Do not set these bits.
2-0	BYCT[10:8]	**Byte Count MSB** Most Significant Byte of the byte count of a received data packet. The Least significant part is provided by UBYCTLX Register UBYCTLX (S:E2h) USB Byte Count Low Register X (X = EPNUM set in UEPNUM Register UEPNUM (S:C7h) USB Endpoint Number) (see Figure 102 on page 143).

(S:C7h) USB Endpoint Number)

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 105. UEPINT Register
UEPINT (S:F8h read-only)
USB Endpoint Interrupt Register

7	6	5	4	3	2	1	0
-	EP6INT	EP5INT	EP4INT	EP3INT	EP2INT	EP1INT	EP0INT

Bit Number	Bit Mnemonic	Description
7	-	**Reserved** The value read from this bit is always 0. Do not set this bit.
6	EP6INT	**Endpoint 6 Interrupt** This bit is set by hardware when an endpoint interrupt source has been detected on the endpoint 6. The endpoint interrupt sources are in the UEPSTAX register and can be: TXCMP, RXOUTB0, RXOUTB1, RXSETUP or STLCRC. A USB interrupt is triggered when the EP6IE bit in the UEPIEN register is set. This bit is cleared by hardware when all the endpoint interrupt sources are cleared
5	EP5INT	**Endpoint 5 Interrupt** This bit is set by hardware when an endpoint interrupt source has been detected on the endpoint 5. The endpoint interrupt sources are in the UEPSTAX register and can be: TXCMP, RXOUTB0, RXOUTB1, RXSETUP or STLCRC. A USB interrupt is triggered when the EP5IE bit in the UEPIEN register is set. This bit is cleared by hardware when all the endpoint interrupt sources are cleared
4	EP4INT	**Endpoint 4 Interrupt** This bit is set by hardware when an endpoint interrupt source has been detected on the endpoint 4. The endpoint interrupt sources are in the UEPSTAX register and can be: TXCMP, RXOUTB0, RXOUTB1, RXSETUP or STLCRC. A USB interrupt is triggered when the EP4IE bit in the UEPIEN register is set. This bit is cleared by hardware when all the endpoint interrupt sources are cleared
3	EP3INT	**Endpoint 3 Interrupt** This bit is set by hardware when an endpoint interrupt source has been detected on the endpoint 3. The endpoint interrupt sources are in the UEPSTAX register and can be: TXCMP, RXOUTB0, RXOUTB1, RXSETUP or STLCRC. A USB interrupt is triggered when the EP3IE bit in the UEPIEN register is set. This bit is cleared by hardware when all the endpoint interrupt sources are cleared
2	EP2INT	**Endpoint 2 Interrupt** This bit is set by hardware when an endpoint interrupt source has been detected on the endpoint 2. The endpoint interrupt sources are in the UEPSTAX register and can be: TXCMP, RXOUTB0, RXOUTB1, RXSETUP or STLCRC. A USB interrupt is triggered when the EP2IE bit in the UEPIEN register is set. This bit is cleared by hardware when all the endpoint interrupt sources are cleared
1	EP1INT	**Endpoint 1 Interrupt** This bit is set by hardware when an endpoint interrupt source has been detected on the endpoint 1. The endpoint interrupt sources are in the UEPSTAX register and can be: TXCMP, RXOUTB0, RXOUTB1, RXSETUP or STLCRC. A USB interrupt is triggered when the EP1IE bit in the UEPIEN register is set. This bit is cleared by hardware when all the endpoint interrupt sources are cleared
0	EP0INT	**Endpoint 0 Interrupt** This bit is set by hardware when an endpoint interrupt source has been detected on the endpoint 0. The endpoint interrupt sources are in the UEPSTAX register and can be: TXCMP, RXOUTB0, RXOUTB1, RXSETUP or STLCRC. A USB interrupt is triggered when the EP0IE bit in the UEPIEN register is set. This bit is cleared by hardware when all the endpoint interrupt sources are cleared

Reset Value = 00h

Table 106. UEPIEN Register
UEPIEN (S:C2h)
USB Endpoint Interrupt Enable Register

7	6	5	4	3	2	1	0
-	EP6INTE	EP5INTE	EP4INTE	EP3INTE	EP2INTE	EP1INTE	EP0INTE

Bit Number	Bit Mnemonic	Description
7	-	**Reserved** The value read from this bit is always 0. Do not set this bit.
6	EP6INTE	**Endpoint 6 Interrupt Enable** Set this bit to enable the interrupts for this endpoint. Clear this bit to disable the interrupts for this endpoint.
5	EP5INTE	**Endpoint 5 Interrupt Enable** Set this bit to enable the interrupts for this endpoint. Clear this bit to disable the interrupts for this endpoint.
4	EP4INTE	**Endpoint 4 Interrupt Enable** Set this bit to enable the interrupts for this endpoint. Clear this bit to disable the interrupts for this endpoint.
3	EP3INTE	**Endpoint 3 Interrupt Enable** Set this bit to enable the interrupts for this endpoint. Clear this bit to disable the interrupts for this endpoint.
2	EP2INTE	**Endpoint 2 Interrupt Enable** Set this bit to enable the interrupts for this endpoint. Clear this bit to disable the interrupts for this endpoint.
1	EP1INTE	**Endpoint 1 Interrupt Enable** Set this bit to enable the interrupts for this endpoint. Clear this bit to disable the interrupts for this endpoint.
0	EP0INTE	**Endpoint 0 Interrupt Enable** Set this bit to enable the interrupts for this endpoint. Clear this bit to disable the interrupts for this endpoint.

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.).

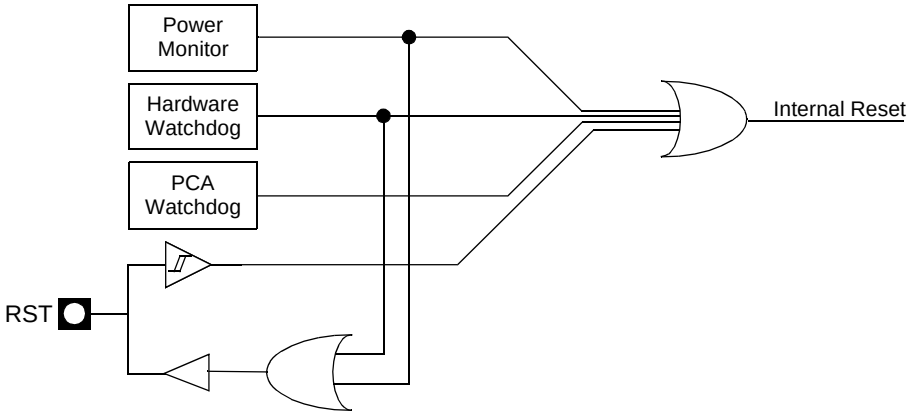
Reset Value = 00h

Reset

Introduction

The reset sources are: Power Management, Hardware Watchdog, PCA Watchdog and Reset input.

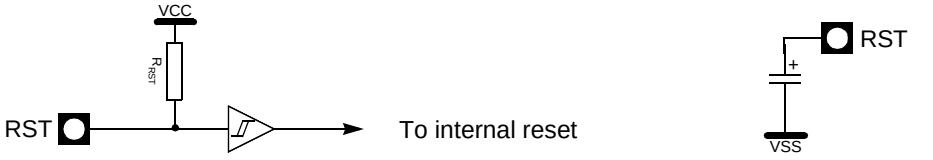
Figure 69. Reset schematic



Reset Input

The Reset input can be used to force a reset pulse longer than the internal reset controlled by the Power Monitor. RST input has a pull-up resistor allowing power-on reset by simply connecting an external capacitor to V_{SS} as shown in Figure 70. Resistor value and input characteristics are discussed in the Section “DC Characteristics” of the AT89C5130A/31A-M datasheet.

Figure 70. Reset Circuitry and Power-On Reset



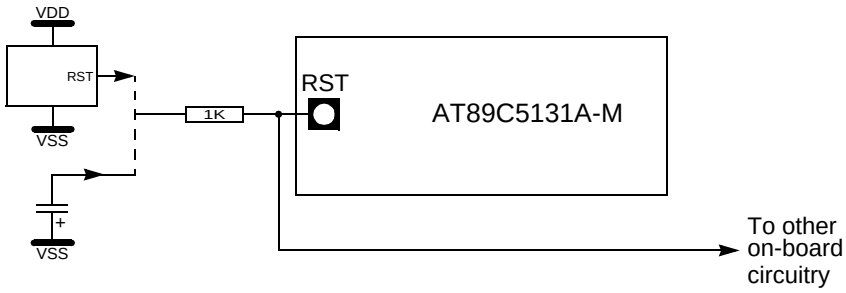
a. RST input circuitry

b. Power-on Reset

Reset Output

As detailed in Section “Hardware Watchdog Timer”, page 156, the WDT generates a 96-clock period pulse on the RST pin. In order to properly propagate this pulse to the rest of the application in case of external capacitor or power-supply supervisor circuit, a 1 kΩ resistor must be added as shown Figure 71.

Figure 71. Recommended Reset Output Schematic



Power Monitor

The POR/PFD function monitors the internal power-supply of the CPU core memories and the peripherals, and if needed, suspends their activity when the internal power supply falls below a safety threshold. This is achieved by applying an internal reset to them.

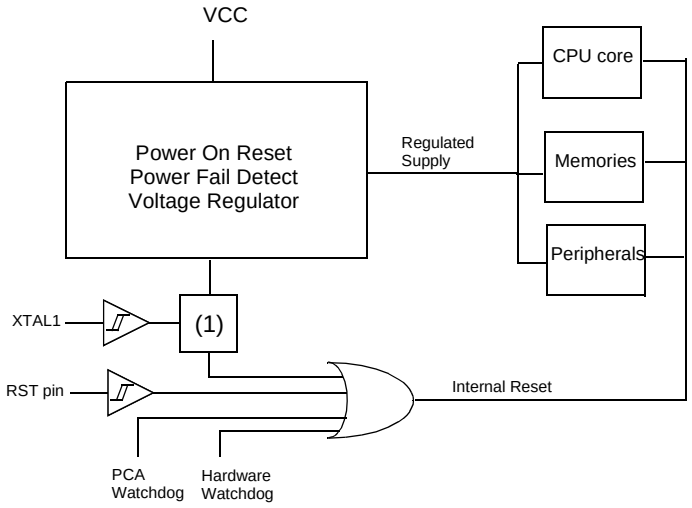
By generating the Reset the Power Monitor insures a correct start up when AT89C5131 is powered up.

Description

In order to startup and maintain the microcontroller in correct operating mode, V_{CC} has to be stabilized in the V_{CC} operating range and the oscillator has to be stabilized with a nominal amplitude compatible with logic level VIH/VIL.

These parameters are controlled during the three phases: power-up, normal operation and power going down. See Figure 72.

Figure 72. Power Monitor Block Diagram

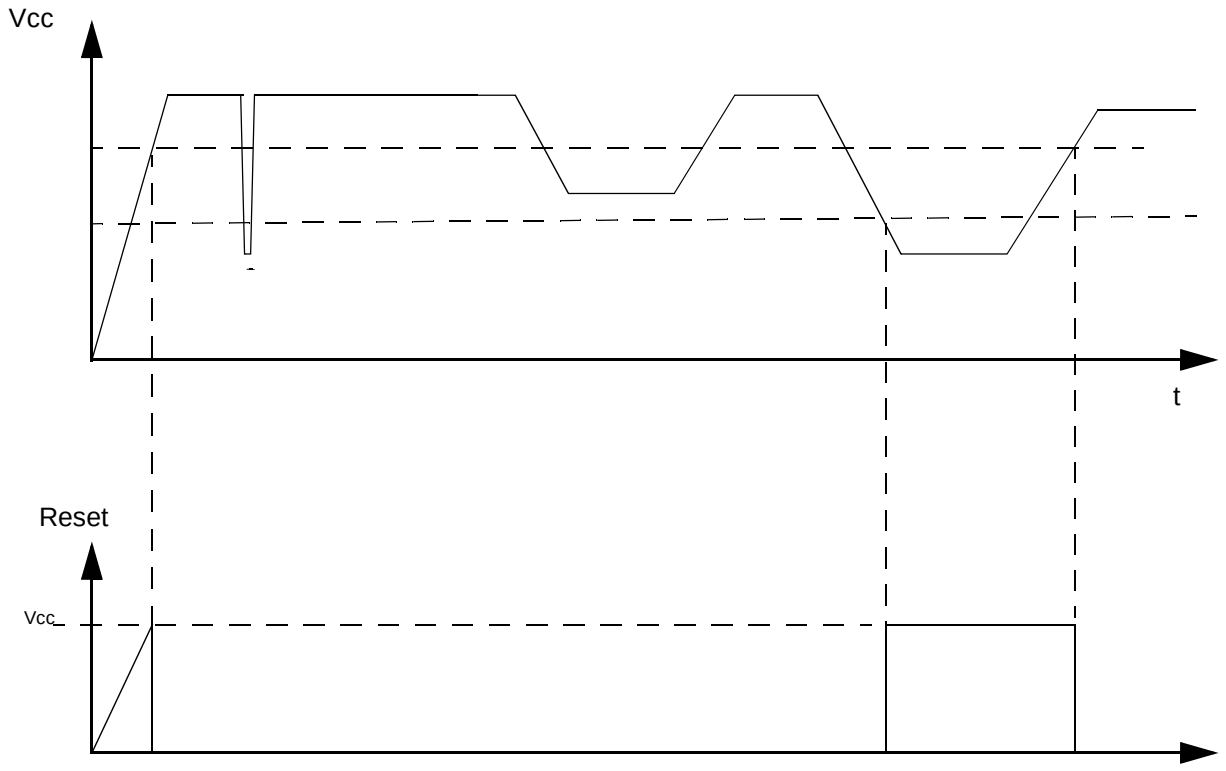


Note: 1. Once XTAL1 High and low levels reach above and below VIH/VIL. a 1024 clock period delay will extend the reset coming from the Power Fail Detect. If the power falls below the Power Fail Detect threshold level, the Reset will be applied immediately.

The Voltage regulator generates a regulated internal supply for the CPU core the memories and the peripherals. Spikes on the external Vcc are smoothed by the voltage regulator.

The Power fail detect monitor the supply generated by the voltage regulator and generate a reset if this supply falls below a safety threshold as illustrated in the Figure 73 below.

Figure 73. Power Fail Detect



When the power is applied, the Power Monitor immediately asserts a reset. Once the internal supply after the voltage regulator reach a safety level, the power monitor then looks at the XTAL clock input. The internal reset will remain asserted until the Xtal1 levels are above and below VIH and VIL. Further more. An internal counter will count 1024 clock periods before the reset is de-asserted.

If the internal power supply falls below a safety level, a reset is immediately asserted.

.

Power Management

Idle Mode

An instruction that sets PCON.0 indicates that it is the last instruction to be executed before going into the Idle mode. In the Idle mode, the internal clock signal is gated off to the CPU, but not to the interrupt, Timer, and Serial Port functions. The CPU status is preserved in its entirety: the Stack Pointer, Program Counter, Program Status Word, Accumulator and all other registers maintain their data during Idle. The port pins hold the logical states they had at the time Idle was activated. ALE and PSEN hold at logic high level.

There are two ways to terminate the Idle mode. Activation of any enabled interrupt will cause PCON.0 to be cleared by hardware, terminating the Idle mode. The interrupt will be serviced, and following RETI the next instruction to be executed will be the one following the instruction that put the device into idle.

The flag bits GF0 and GF1 can be used to give an indication if an interrupt occurred during normal operation or during an Idle. For example, an instruction that activates Idle can also set one or both flag bits. When Idle is terminated by an interrupt, the interrupt service routine can examine the flag bits.

The other way of terminating the Idle mode is with a hardware reset. Since the clock oscillator is still running, the hardware reset needs to be held active for only two machine cycles (24 oscillator periods) to complete the reset.

Power-down Mode

To save maximum power, a power-down mode can be invoked by software (refer to Table 13, PCON register).

In power-down mode, the oscillator is stopped and the instruction that invoked power-down mode is the last instruction executed. The internal RAM and SFRs retain their value until the power-down mode is terminated. V_{CC} can be lowered to save further power. Either a hardware reset or an external interrupt can cause an exit from power-down. To properly terminate power-down, the reset or external interrupt should not be executed before V_{CC} is restored to its normal operating level and must be held active long enough for the oscillator to restart and stabilize.

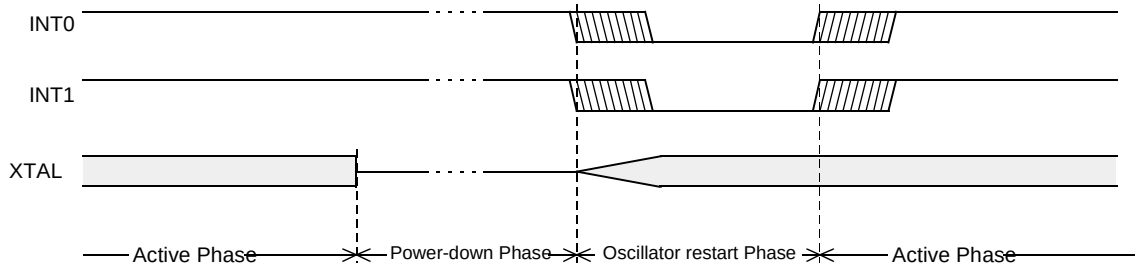
Only:

- external interrupt $\overline{\text{INT0}}$,
- external interrupt $\overline{\text{INT1}}$,
- Keyboard interrupt and
- USB Interrupt

are useful to exit from power-down. For that, interrupt must be enabled and configured as level or edge sensitive interrupt input. When Keyboard Interrupt occurs after a power down mode, 1024 clocks are necessary to exit to power-down mode and enter in operating mode.

Holding the pin low restarts the oscillator but bringing the pin high completes the exit as detailed in Figure 74. When both interrupts are enabled, the oscillator restarts as soon as one of the two inputs is held low and power-down exit will be completed when the first input is released. In this case, the higher priority interrupt service routine is executed. Once the interrupt is serviced, the next instruction to be executed after RETI will be the one following the instruction that put AT89C5130A/31A-M into power-down mode.

Figure 74. Power-down Exit Waveform



Exit from power-down by reset redefines all the SFRs, exit from power-down by external interrupt does no affect the SFRs.

Exit from power-down by either reset or external interrupt does not affect the internal RAM content.

Note: If idle mode is activated with power-down mode (IDL and PD bits set), the exit sequence is unchanged, when execution is vectored to interrupt, PD and IDL bits are cleared and idle mode is not entered.

This table shows the state of ports during idle and power-down modes.

Table 109. State of Ports

Mode	Program Memory	ALE	PSEN	PORT0	PORT1	PORT2	PORT3	PORTI2
Idle	Internal	1	1	Port Data(1)	Port Data	Port Data	Port Data	Port Data
Idle	External	1	1	Floating	Port Data	Address	Port Data	Port Data
Power-down	Internal	0	0	Port Data(1)	Port Data	Port Data	Port Data	Port Data
Power-down	External	0	0	Floating	Port Data	Port Data	Port Data	Port Data

Note: 1. Port 0 can force a 0 level. A “one” will leave port floating.

Registers

Table 110. PCON Register

PCON (S:87h)
Power Control Register

7	6	5	4	3	2	1	0
SMOD1	**SMOD0**	**-**	**POF**	**GF1**	**GF0**	**PD**	**IDL**

Bit Number	Bit Mnemonic	Description
7	SMOD1	**Serial Port Mode bit 1** Set to select double baud rate in mode 1, 2 or 3.
6	SMOD0	**Serial Port Mode bit 0** Set to select FE bit in SCON register. Clear to select SM0 bit in SCON register
5	-	**Reserved** The value read from this bit is always 0. Do not set this bit.
4	POF	**Power-Off Flag** Set by hardware when V_{CC} rises from 0 to its nominal voltage. Can also be set by software. Clear to recognize next reset type.
3	GF1	**General-purpose Flag 1** Set by software for general-purpose usage. Cleared by software for general-purpose usage.
2	GF0	**General-purpose Flag 0** Set by software for general-purpose usage. Cleared by software for general-purpose usage.
1	PD	**Power-down mode bit** Set this bit to enter in power-down mode. Cleared by hardware when reset occurs.
0	IDL	**Idle mode bit** Set this bit to enter in Idle mode. Cleared by hardware when interrupt or reset occurs.

Reset Value = 10h

Hardware Watchdog Timer

The WDT is intended as a recovery method in situations where the CPU may be subjected to software upset. The WDT consists of a 14-bit counter and the WatchDog Timer ReSeT (WDTRST) SFR. The WDT is by default disabled from exiting reset. To enable the WDT, user must write 01EH and 0E1H in sequence to the WDTRST, SFR location 0A6H. When WDT is enabled, it will increment every machine cycle while the oscillator is running and there is no way to disable the WDT except through reset (either hardware reset or WDT overflow reset). When WDT overflows, it will drive an output RESET LOW pulse at the RST-pin.

Using the WDT

To enable the WDT, user must write 01EH and 0E1H in sequence to the WDTRST, SFR location 0A6H. When WDT is enabled, the user needs to service it by writing to 01EH and 0E1H to WDTRST to avoid WDT overflow. The 14-bit counter overflows when it reaches 16383 (3FFFH) and this will reset the device. When WDT is enabled, it will increment every machine cycle while the oscillator is running. This means the user must reset the WDT at least every 16383 machine cycle. To reset the WDT the user must write 01EH and 0E1H to WDTRST. WDTRST is a write only register. The WDT counter cannot be read or written. When WDT overflows, it will generate an output RESET pulse at the RST-pin. The RESET pulse duration is 96 x $T_{CLK\ PERIPH}$, where $T_{CLK\ PERIPH}$ = $1/F_{CLK\ PERIPH}$. To make the best use of the WDT, it should be serviced in those sections of code that will periodically be executed within the time required to prevent a WDT reset.

To have a more powerful WDT, a 2^7 counter has been added to extend the Time-out capability, ranking from 16 ms to 2s at F_{OSCA} = 12 MHz. To manage this feature, refer to WDTPRG register description, Table 112.

Table 111. WDTRST Register

WDTRST - Watchdog Reset Register (0A6h)

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

Reset Value = XXXX XXXXb

Write only, this SFR is used to reset/enable the WDT by writing 01EH then 0E1H in sequence.

Table 112. WDTPRG Register

WDTPRG - Watchdog Timer Out Register (0A7h)

7	6	5	4	3	2	1	0
-	-	-	-	-	S2	S1	S0

Bit Number	Bit Mnemonic	Description
7	-	**Reserved** The value read from this bit is undetermined. Do not try to set this bit.
6	-	
5	-	
4	-	
3	-	
2	S2	WDT Time-out select bit 2
1	S1	WDT Time-out select bit 1
0	S0	WDT Time-out select bit 0
		S2 S1 S0 Selected Time-out 0 0 0 16384x2^(214 - 1) machine cycles, 16.3 ms at FOSC = 12 MHz 0 0 1 16384x2^(215 - 1) machine cycles, 32.7 ms at FOSC = 12 MHz 0 1 0 16384x2^(216 - 1) machine cycles, 65.5 ms at FOSC = 12 MHz 0 1 1 16384x2^(217 - 1) machine cycles, 131 ms at FOSC = 12 MHz 1 0 0 16384x2^(218 - 1) machine cycles, 262 ms at FOSC = 12 MHz 1 0 1 16384x2^(219 - 1) machine cycles, 542 ms at FOSC = 12 MHz 1 1 0 16384x2^(220 - 1) machine cycles, 1.05 s at FOSC = 12 MHz 1 1 1 16384x2^(221 - 1) machine cycles, 2.09 s at FOSC = 12 MHz 16384x2^S machine cycles

Reset value = XXXX X000

WDT During Power-down and Idle

In Power-down mode the oscillator stops, which means the WDT also stops. While in Power-down mode the user does not need to service the WDT. There are 2 methods of exiting Power-down mode: by a hardware reset or via a level activated external interrupt which is enabled prior to entering Power-down mode. When Power-down is exited with hardware reset, servicing the WDT should occur as it normally should whenever the AT89C5130A/31A-M is reset. Exiting Power-down with an interrupt is significantly different. The interrupt is held low long enough for the oscillator to stabilize. When the interrupt is brought high, the interrupt is serviced. To prevent the WDT from resetting the device while the interrupt pin is held low, the WDT is not started until the interrupt is pulled high. It is suggested that the WDT be reset during the interrupt service routine.

To ensure that the WDT does not overflow within a few states of exiting of power-down, it is better to reset the WDT just before entering power-down.

In the Idle mode, the oscillator continues to run. To prevent the WDT from resetting the AT89C5130A/31A-M while in Idle mode, the user should always set up a timer that will periodically exit Idle, service the WDT, and re-enter Idle mode.

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

Reduced EMI Mode

The ALE signal is used to demultiplex address and data buses on port 0 when used with external program or data memory. Nevertheless, during internal code execution, ALE signal is still generated. In order to reduce EMI, ALE signal can be disabled by setting AO bit.

The AO bit is located in AUXR register at bit location 0. As soon as AO is set, ALE is no longer output but remains active during MOVX and MOVC instructions and external fetches. During ALE disabling, ALE pin is weakly pulled high.

Table 114. AUXR Register

AUXR - Auxiliary Register (8Eh)

7	6	5	4	3	2	1	0
DPU	-	M0	-	XRS1	XRS0	EXTRAM	AO

Bit Number	Bit Mnemonic	Description
7	DPU	**Disable Weak Pull Up** Cleared to enabled weak pull up on standard Ports Set to disable weak pull up on standard Ports
6	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
5	M0	**Pulse length** Cleared to stretch MOVX control: the $\overline{RD}$ and the $\overline{WR}$ pulse length is 6 clock periods (default). Set to stretch MOVX control: the $\overline{RD}$ and the $\overline{WR}$ pulse length is 30 clock periods.
4	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
3	XRS1	**ERAM Size** **XRS1 XRS0 ERAM size** 0 0 256 bytes 0 1 512 bytes 1 0 768 bytes 1 1 1024 bytes (default)
2	XRS0	
1	EXTRAM	**EXTRAM bit** Cleared to access internal ERAM using MOVX at Ri at DPTR. Set to access external memory.
0	AO	**ALE Output bit** Cleared, ALE is emitted at a constant rate of 1/6 the oscillator frequency (or 1/3 if X2 mode is used) (default). Set, ALE is active only during a MOVX or MOVC instruction is used.

Reset Value = 0X0X 1100b
Not bit addressable

Electrical Characteristics

Absolute Maximum Ratings

Ambient Temperature Under Bias:

I = industrial .....-40°C to 85°C

Storage Temperature ..... -65°C to + 150°C

Voltage on V_{CC} from V_{SS}.....-0.5V to + 6V

Voltage on Any Pin from V_{SS} .....-0.5V to V_{CC} + 0.2V

Note: Stresses at or above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions may affect device reliability.

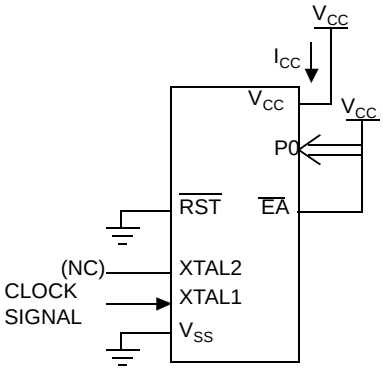
DC Parameters

T_A = -40°C to +85°C; V_{SS} = 0V; V_{CC} = 2.7V to 5.5 V; F = 0 to 48 MHz

Symbol	Parameter	Min	Typ[5]	Max	Unit	Test Conditions
V_{IL}	Input Low Voltage	-0.5		0.2Vcc - 0.1	V	
V_{IH}	Input High Voltage except XTAL1, $\overline{RST}$	0.2 V_{CC} + 0.9		V_{CC} + 0.5	V	
V_{IH1}	Input High Voltage, XTAL1, $\overline{RST}$	0.7 V_{CC}		V_{CC} + 0.5	V	
V_{OL}	Output Low Voltage, ports 1, 2, 3 and 4[6]			0.3 0.45 1.0	V V V	I_{OL} = 100 µA[4] I_{OL} = 0.8 mA[4] I_{OL} = 1.6mA[4]
V_{OL1}	Output Low Voltage, port 0, ALE, $\overline{PSEN}$ [6]			0.3 0.45 1.0	V V V	I_{OL} = 200 µA[4] I_{OL} = 1.6 mA[4] I_{OL} = 3.5 mA[4]
V_{OH}	Output High Voltage, ports 1, 2, 3, 4 and 5	V_{CC} - 0.3 V_{CC} - 0.7 V_{CC} - 1.5			V V V	I_{OH} = -10 µA I_{OH} = -30 µA I_{OH} = -60 µA V_{CC} = 3.3V ± 10%
V_{OH1}	Output High Voltage, port 0, ALE, $\overline{PSEN}$	V_{CC} - 0.3 V_{CC} - 0.7 V_{CC} - 1.5			V V V	I_{OH} = -200 µA I_{OH} = -1.6 mA I_{OH} = -3.5 mA V_{CC} = 3.3V ± 10%
R_{RST}	$\overline{RST}$ Pullup Resistor	50	100	200	kΩ	
I_{IL}	Logical 0 Input Current ports 1, 2, 3 and 4			-50	µA	Vin = 0.45V
I_{LI}	Input Leakage Current			±10	µA	0.45V < Vin < V_{CC}
I_{TL}	Logical 1 to 0 Transition Current, ports 1, 2, 3 and 4			-650	µA	Vin = 2.0V
C_{IO}	Capacitance of I/O Buffer			10	pF	Fc = 1 MHz T_A = 25°C
I_{PD}	Power-down Current			100µA	µA	3.0V < V_{CC} < 3.6V[3]
I_{CC}	Power Supply Current	I_{CCOP} = 0.4xF(MHz)+5 I_{CCIDLE} = 0.3xF(MHz)+5 $I_{CCwrite}$ = 0.8xF(MHz)+15				V_{CC} = 3.3V [1][2]

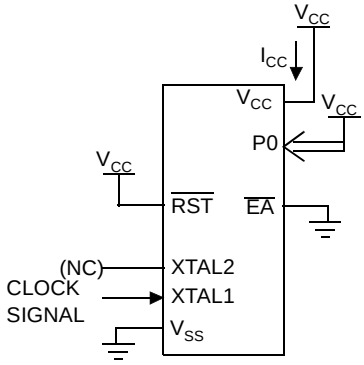
Notes:
1. Operating I_{CC} is measured with all output pins disconnected; XTAL1 driven with T_{CLCH}, T_{CHCL} = 5 ns (see Figure 78.), V_{IL} = V_{SS} + 0.5V,
   V_{IH} = V_{CC} - 0.5V; XTAL2 N.C.; $\overline{EA}$ = $\overline{RST}$ = Port 0 = V_{CC}. I_{CC} would be slightly higher if a crystal oscillator used (see Figure 75.).
2. Idle I_{CC} is measured with all output pins disconnected; XTAL1 driven with T_{CLCH}, T_{CHCL} = 5 ns, V_{IL} = V_{SS} + 0.5V, V_{IH} = V_{CC} - 0.5V; XTAL2 N.C; Port 0 = V_{CC}; $\overline{EA}$ = $\overline{RST}$ = V_{SS} (see Figure 76).
3. Power-down I_{CC} is measured with all output pins disconnected; $\overline{EA}$ = V_{CC}, PORT 0 = V_{CC}; XTAL2 NC.; $\overline{RST}$ = V_{SS} (see Figure 77.). In addition, the WDT must be inactive and the POF flag must be set.
4. Under steady state (non-transient) conditions, I_{OL} must be externally limited as follows:
   Maximum I_{OL} per port pin: 10 mA
   Maximum I_{OL} per 8-bit port:
   Port 0: 26 mA
   Ports 1, 2 and 3: 15 mA
   Maximum total I_{OL} for all output pins: 71 mA
   If I_{OL} exceeds the test condition, V_{OL} may exceed the related specification. Pins are not guaranteed to sink current greater than the listed test conditions.

Figure 75. I_{CC} Test Condition, Active Mode



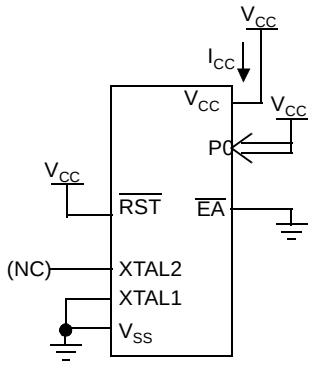
All other pins are disconnected.

Figure 76. I_{CC} Test Condition, Idle Mode



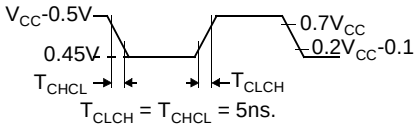
All other pins are disconnected.

Figure 77. I_{CC} Test Condition, Power-down Mode



All other pins are disconnected.

Figure 78. Clock Signal Waveform for I_{CC} Tests in Active and Idle Modes



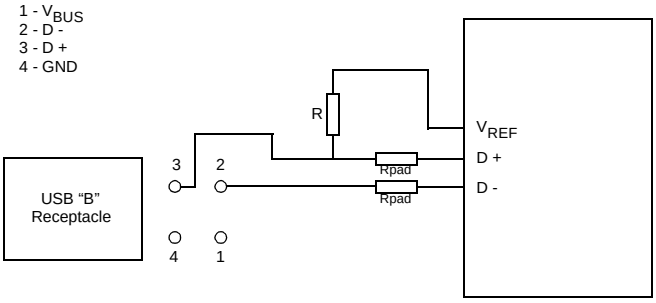
LED's

Table 115. LED Outputs DC Parameters

Symbol	Parameter	Min	Typ	Max	Unit	Test Conditions
I_{OL}	Output Low Current, P3.6 and P3.7 LED modes	1	2	4	mA	2 mA configuration
		2	4	8	mA	4 mA configuration
		5	10	20	mA	10 mA configuration

Note: 1. (T_A = -20°C to +50°C, V_{CC} - V_{OL} = 2 V ± 20%)

USB DC Parameters



R = 1.5 kΩ
R_{pad} = 27Ω

Symbol	Parameter	Min	Typ	Max	Unit
V_{REF}	USB Reference Voltage	3.0		3.6	V
V_{IH}	Input High Voltage for D+ and D- (Driven)	2.0		4.0	V
V_{IHZ}	Input High Voltage for D+ and D- (Floating)	2.7		3.6	V
V_{IL}	Input Low Voltage for D+ and D-			0.8	V
V_{OH}	Output High Voltage for D+ and D-	2.8		3.6	V
V_{OL}	Output Low Voltage for D+ and D-	0.0		0.3	V

AC Parameters

Explanation of the AC Symbols

Each timing symbol has 5 characters. The first character is always a “T” (stands for time). The other characters, depending on their positions, stand for the name of a signal or the logical status of that signal. The following is a list of all the characters and what they stand for.

Example: T_{AVLL} = Time for Address Valid to ALE Low.
T_{LLPL} = Time for ALE Low to $\overline{PSEN}$ Low.

T_A = -40°C to +85°C; V_{SS} = 0V; V_{CC} = 3.3V ±10%; F = 0 to 48 MHz.

T_A = -40°C to +85°C; V_{SS} = 0V; V_{CC} = 3.3V ± 10%.

(Load Capacitance for port 0, ALE and PSEN = 60 pF; Load Capacitance for all other outputs = 60 pF.)

Table 116, Table 119 and Table 122 give the description of each AC symbols.

Table 117, Table 121 and Table 123 give for each range the AC parameter.

Table 118, Table 121 and Table 124 give the frequency derating formula of the AC parameter for each speed range description. To calculate each AC symbols. take the x value and use this value in the formula.

Example: T_{LLIV} and 20 MHz, Standard clock.
x = 30 ns
T = 50 ns
T_{CCIV} = 4T - x = 170 ns

External Program Memory Characteristics

Table 116. Symbol Description

Symbol	Parameter
T	Oscillator Clock Period
T_{LHLL}	ALE Pulse Width
T_{AVLL}	Address Valid to ALE
T_{LLAX}	Address Hold after ALE
T_{LLIV}	ALE to Valid Instruction In
T_{LLPL}	ALE to $\overline{PSEN}$
T_{PLPH}	$\overline{PSEN}$ Pulse Width
T_{PLIV}	$\overline{PSEN}$ to Valid Instruction In
T_{PXIX}	Input Instruction Hold after $\overline{PSEN}$
T_{PXIZ}	Input Instruction Float after $\overline{PSEN}$
T_{AVIV}	Address to Valid Instruction In
T_{PLAZ}	$\overline{PSEN}$ Low to Address Float

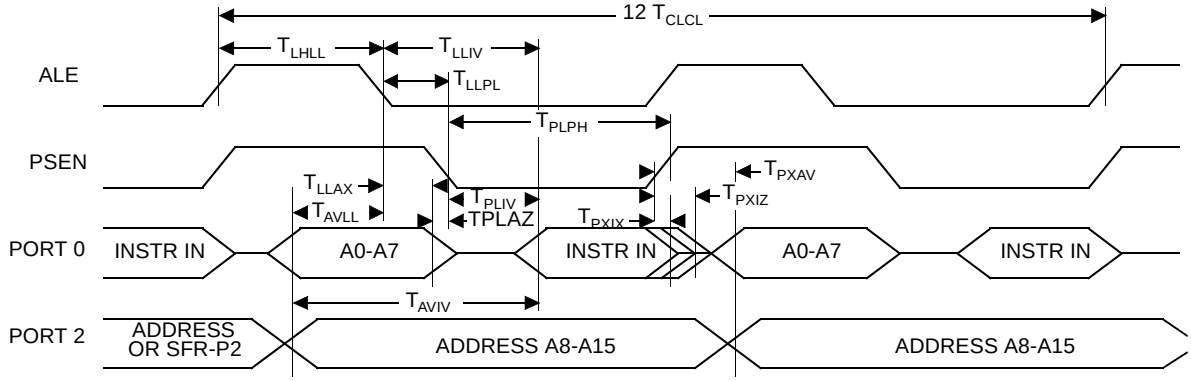
Table 117. AC Parameters for a Fix Clock (F = 40 MHz)

Symbol	Min	Max	Units
T	25		ns
T_{LHLL}	40		ns
T_{AVLL}	10		ns
T_{LLAX}	10		ns
T_{LLIV}		70	ns
T_{LLPL}	15		ns
T_{PLPH}	55		ns
T_{PLIV}		35	ns
T_{PXIX}	0		ns
T_{PXIZ}		18	ns
T_{AVIV}		85	ns
T_{PLAZ}		10	ns

Table 118. AC Parameters for a Variable Clock

Symbol	Type	Standard Clock	X2 Clock	X Parameter	Units
T_{LHLL}	Min	2 T - x	T - x	10	ns
T_{AVLL}	Min	T - x	0.5 T - x	15	ns
T_{LLAX}	Min	T - x	0.5 T - x	15	ns
T_{LLIV}	Max	4 T - x	2 T - x	30	ns
T_{LLPL}	Min	T - x	0.5 T - x	10	ns
T_{PLPH}	Min	3 T - x	1.5 T - x	20	ns
T_{PLIV}	Max	3 T - x	1.5 T - x	40	ns
T_{PXIX}	Min	x	x	0	ns
T_{PXIZ}	Max	T - x	0.5 T - x	7	ns
T_{AVIV}	Max	5 T - x	2.5 T - x	40	ns
T_{PLAZ}	Max	x	x	10	ns

External Program Memory Read Cycle



External Data Memory Characteristics

Table 119. Symbol Description

Symbol	Parameter
T_{RLRH}	$\overline{RD}$ Pulse Width
T_{WLWH}	$\overline{WR}$ Pulse Width
T_{RLDV}	$\overline{RD}$ to Valid Data In
T_{RHDX}	Data Hold After $\overline{RD}$
T_{RHDZ}	Data Float After $\overline{RD}$
T_{LLDV}	ALE to Valid Data In
T_{AVDV}	Address to Valid Data In
T_{LLWL}	ALE to $\overline{WR}$ or $\overline{RD}$
T_{AVWL}	Address to $\overline{WR}$ or $\overline{RD}$
T_{QVWX}	Data Valid to $\overline{WR}$ Transition
T_{QVWH}	Data set-up to $\overline{WR}$ High
T_{WHQX}	Data Hold After $\overline{WR}$
T_{RLAZ}	$\overline{RD}$ Low to Address Float
T_{WHLH}	$\overline{RD}$ or $\overline{WR}$ High to ALE high

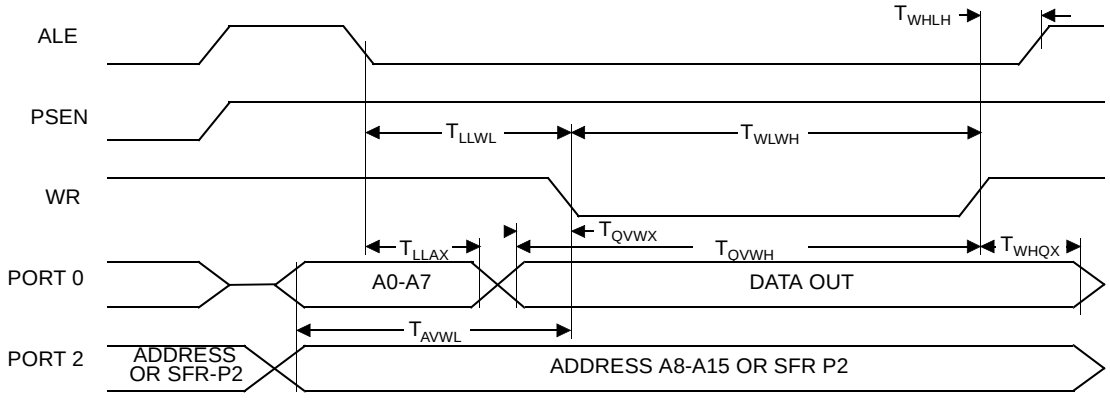
Table 120. AC Parameters for a Variable Clock (F = 40 MHz)

Symbol	Min	Max	Units
T_{RLRH}	130		ns
T_{WLWH}	130		ns
T_{RLDV}		100	ns
T_{RHDX}	0		ns
T_{RHDZ}		30	ns
T_{LLDV}		160	ns
T_{AVDV}		165	ns
T_{LLWL}	50	100	ns
T_{AVWL}	75		ns
T_{QVWX}	10		ns
T_{QVWH}	160		ns
T_{WHQX}	15		ns
T_{RLAZ}		0	ns
T_{WHLH}	10	40	ns

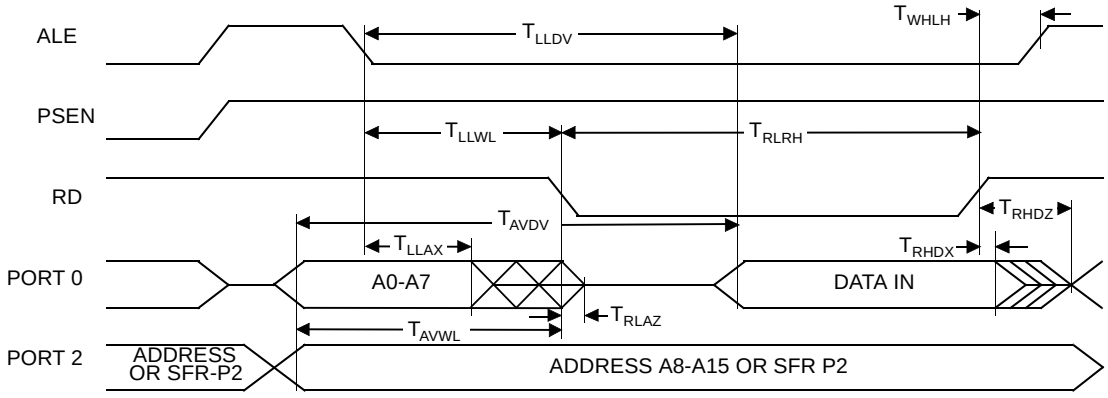
Table 121. AC Parameters for a Variable Clock

Symbol	Type	Standard Clock	X2 Clock	X Parameter	Units
T_{RLRH}	Min	6 T - x	3 T - x	20	ns
T_{WLWH}	Min	6 T - x	3 T - x	20	ns
T_{RLDV}	Max	5 T - x	2.5 T - x	25	ns
T_{RHDX}	Min	x	x	0	ns
T_{RHDZ}	Max	2 T - x	T - x	20	ns
T_{LLDV}	Max	8 T - x	4T -x	40	ns
T_{AVDV}	Max	9 T - x	4.5 T - x	60	ns
T_{LLWL}	Min	3 T - x	1.5 T - x	25	ns
T_{LLWL}	Max	3 T + x	1.5 T + x	25	ns
T_{AVWL}	Min	4 T - x	2 T - x	25	ns
T_{QVWX}	Min	T - x	0.5 T - x	15	ns
T_{QVWH}	Min	7 T - x	3.5 T - x	25	ns
T_{WHQX}	Min	T - x	0.5 T - x	10	ns
T_{RLAZ}	Max	x	x	0	ns
T_{WHLH}	Min	T - x	0.5 T - x	15	ns
T_{WHLH}	Max	T + x	0.5 T + x	15	ns

External Data Memory Write Cycle



External Data Memory Read Cycle



Serial Port Timing - Shift Register Mode

Table 122. Symbol Description (F = 40 MHz)

Symbol	Parameter
T_{XLXL}	Serial port clock cycle time
T_{QVHX}	Output data set-up to clock rising edge
T_{XHQX}	Output data hold after clock rising edge
T_{XHDX}	Input data hold after clock rising edge
T_{XHDV}	Clock rising edge to input data valid

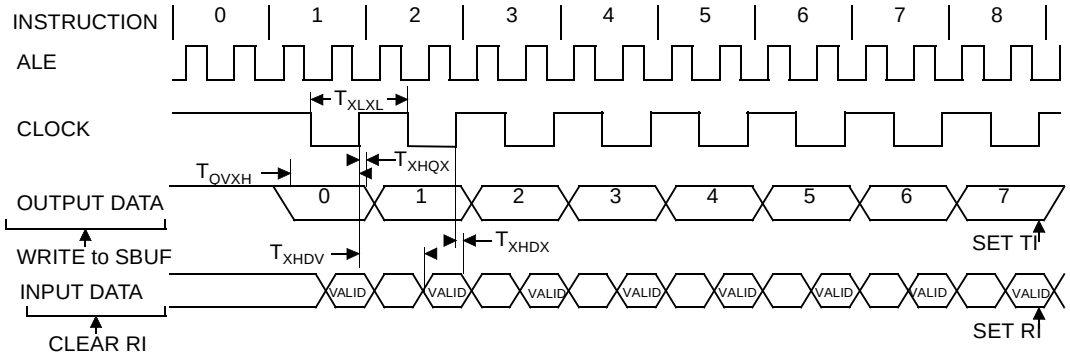
Table 123. AC Parameters for a Fix Clock (F = 40 MHz)

Symbol	Min	Max	Units
T_{XLXL}	300		ns
T_{QVHX}	200		ns
T_{XHQX}	30		ns
T_{XHDX}	0		ns
T_{XHDV}		117	ns

Table 124. AC Parameters for a Variable Clock

Symbol	Type	Standard Clock	X2 Clock	X Parameter for -M Range	Units
T_{XLXL}	Min	12 T	6 T		ns
T_{QVHX}	Min	10 T - x	5 T - x	50	ns
T_{XHQX}	Min	2 T - x	T - x	20	ns
T_{XHDX}	Min	x	x	0	ns
T_{XHDV}	Max	10 T - x	5 T- x	133	ns

Shift Register Timing Waveform

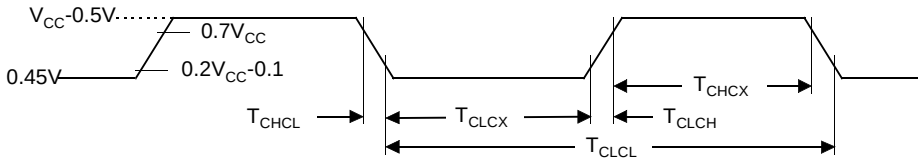


External Clock Drive Characteristics (XTAL1)

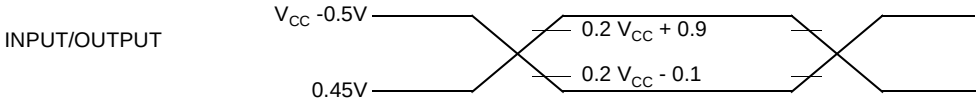
Table 125. AC Parameters

Symbol	Parameter	Min	Max	Units
T_{CLCL}	Oscillator Period	21		ns
T_{CHCX}	High Time	5		ns
T_{CLCX}	Low Time	5		ns
T_{CLCH}	Rise Time		5	ns
T_{CHCL}	Fall Time		5	ns
T_{CHCX}/T_{CLCX}	Cyclic ratio in X2 mode	40	60	%

External Clock Drive Waveforms

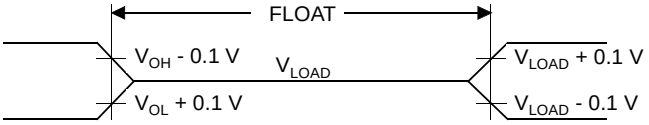


AC Testing Input/Output Waveforms



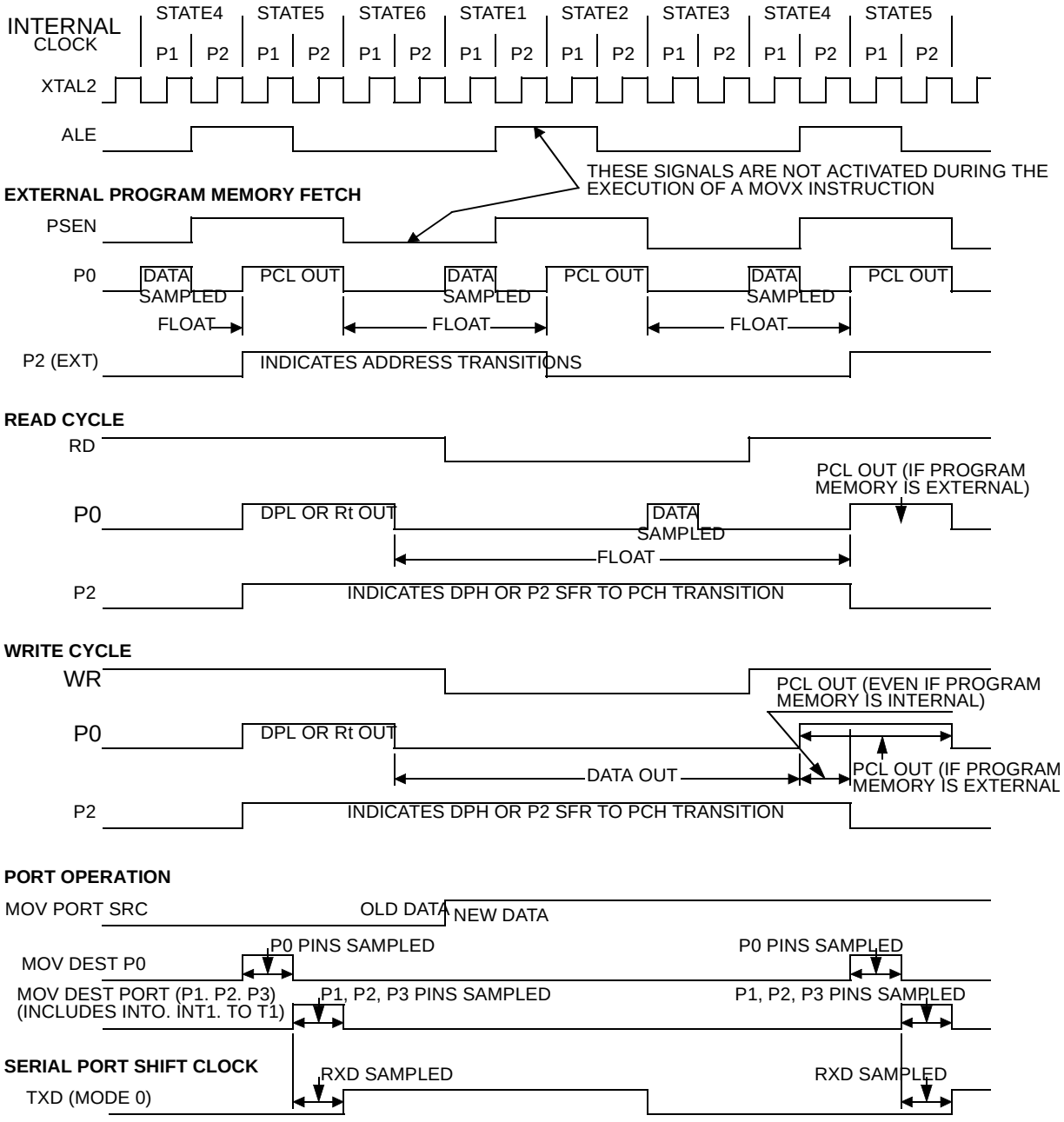
AC inputs during testing are driven at V_{CC} - 0.5 for a logic “1” and 0.45V for a logic “0”. Timing measurement are made at V_{IH} min for a logic “1” and V_{IL} max for a logic “0”.

Float Waveforms



For timing purposes as port pin is no longer floating when a 100 mV change from load voltage occurs and begins to float when a 100 mV change from the loaded V_{OH}/V_{OL} level occurs. $I_{OL}/I_{OH} \geq \pm 20$ mA.

Clock Waveforms Valid in normal clock mode. In X2 mode XTAL2 must be changed to XTAL2/2.



This diagram indicates when signals are clocked internally. The time it takes the signals to propagate to the pins, however, ranges from 25 to 125 ns. This propagation delay is dependent on variables such as temperature and pin loading. Propagation also varies from output to output and component. Typically though (T_A = 25°C fully loaded) $\overline{RD}$ and $\overline{WR}$ propagation delays are approximately 50 ns. The other signals are typically 85 ns. Propagation delays are incorporated in the AC specifications.

Flash Memory

Table 126. Timing Symbol Definitions

Signals	
S (Hardware Condition)	$\overline{PSEN}$, EA
R	$\overline{RST}$
B	FBUSY Flag

Conditions	
L	Low
V	Valid
X	No Longer Valid

Table 127. Memory AC Timing
Vcc = 3.3V ± 10%, T_A = -40 to +85°C

Symbol	Parameter	Min	Typ	Max	Unit
T_{SVRL}	Input $\overline{PSEN}$ Valid to $\overline{RST}$ Edge	50			ns
T_{RLSX}	Input $\overline{PSEN}$ Hold after $\overline{RST}$ Edge	50			ns
T_{BHBL}	Flash Internal Busy (Programming) Time		10		ms

Figure 79. Flash Memory - ISP Waveforms

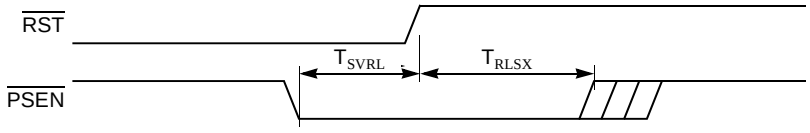
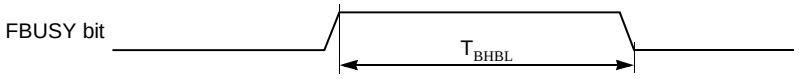


Figure 80. Flash Memory - Internal Busy Waveforms



USB AC Parameters

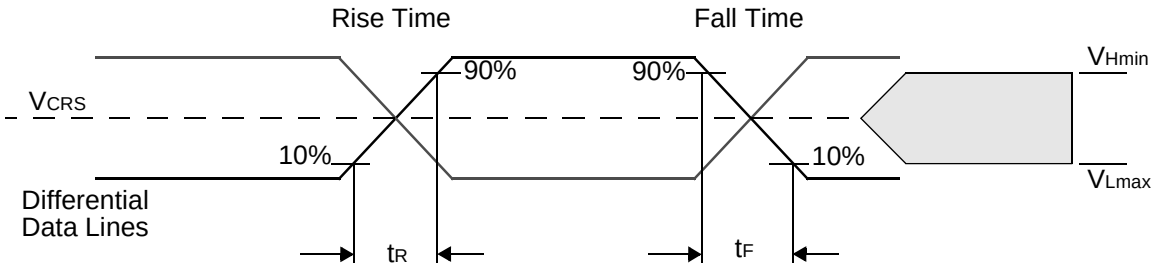


Table 128. USB AC Parameters

Symbol	Parameter	Min	Typ	Max	Unit	Test Conditions
t_R	Rise Time	4		20	ns	
t_F	Fall Time	4		20	ns	
t_{FDRATE}	Full-speed Data Rate	11.9700		12.0300	Mb/s	
V_{CRS}	Crossover Voltage	1.3		2.0	V	
t_{DJ1}	Source Jitter Total to Next Transaction	-3.5		3.5	ns	
t_{DJ2}	Source Jitter Total for Paired Transactions	-4		4	ns	
t_{JR1}	Receiver Jitter to Next Transaction	-18.5		18.5	ns	
t_{JR2}	Receiver Jitter for Paired Transactions	-9		9	ns	

Ordering Information

Table 129. Possible Order Entries

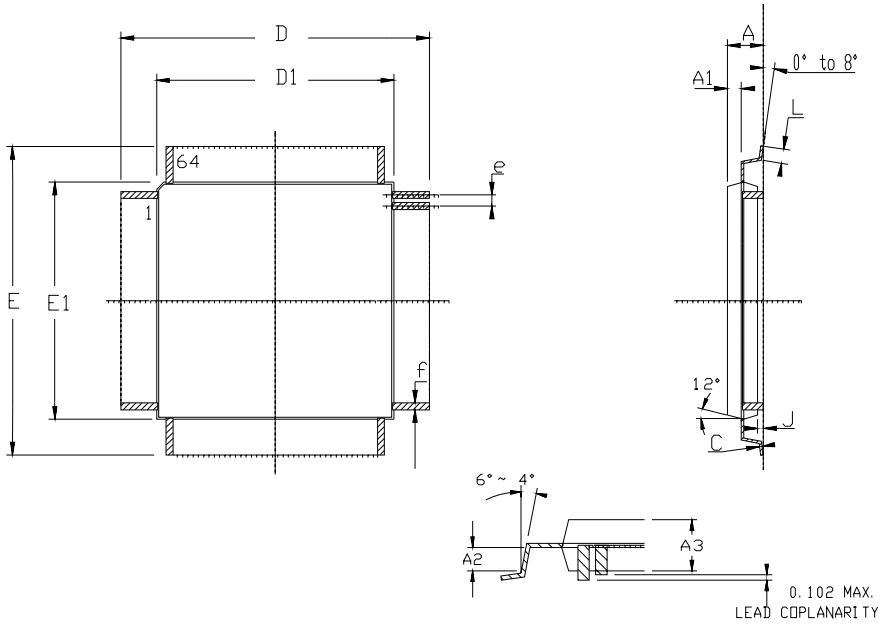
Part Number	Memory Size (Kbytes)	Supply Voltage	Temperature Range	Package	Packing
AT89C5130A-RDTIM	16	2.7 to 5.5V	Industrial	VQFP64	Tray
AT89C5130A-PUTIM	16	2.7 to 5.5V	Industrial	QFN32	Stick
AT89C5130A-S3SIM	16	2.7 to 5.5V	Industrial	PLCC52	Stick
AT89C5131A-RDTIM	32	2.7 to 5.5V	Industrial	VQFP64	Tray
AT89C5131A-PUTIM	32	2.7 to 5.5V	Industrial	QFN32	Stick
AT89C5131A-S3SIM	32	2.7 to 5.5V	Industrial	PLCC52	Stick
AT89C5130A-RDTUM(1)	16	2.7 to 5.5V	Industrial	LHF VQFP64	Tray & Dry Pack
AT89C5131A-RDTUM(1)	32	2.7 to 5.5V	Industrial	LHF VQFP64	Tray & Dry Pack

Note: 1. Indicates “Green” product version.

2. Optional Packing and Package options (please consult Atmel sales representative)
   -Tape and Reel
   -Dry Pack
   -Die form

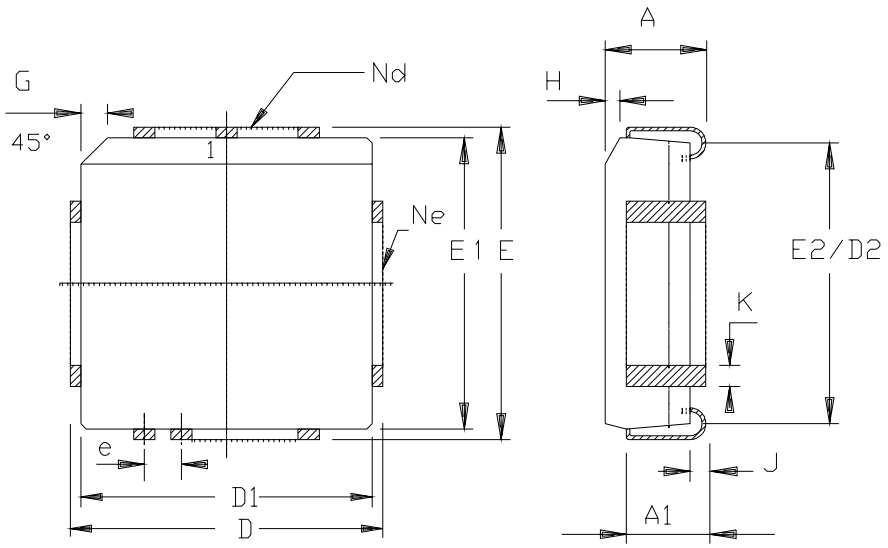
Packaging Information

64-lead VQFP



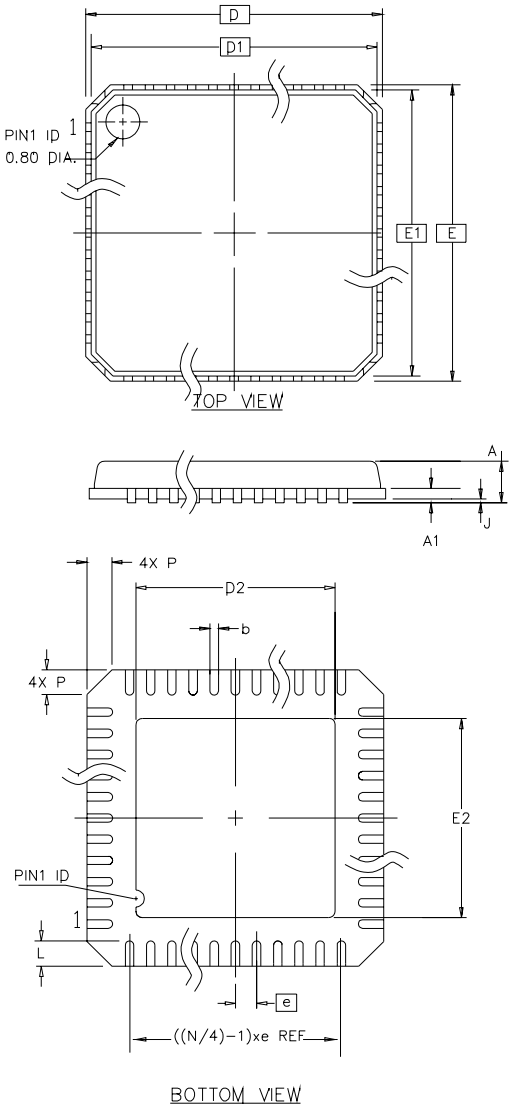
	MM		INCH	
	Min	Max	Min	Max
A	-	1.60	-	.063
A1	0.64 REF		.025 REF	
A2	0.64 REF		.025 REF	
A3	1.35	1.45	.053	.057
D	11.75	12.25	.463	.483
D1	9.90	10.10	.390	.398
E	11.75	12.25	.463	.483
E1	9.90	10.10	.390	.398
J	0.05	-	.002	-
L	0.45	0.75	.018	.030
e	0.50 BSC		.0197 BSC	
f	0.25 BSC		.010 BSC	

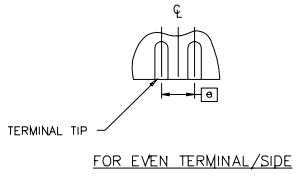
52-lead PLCC



	MM		INCH	
A	4.20	4.57	.165	.180
A1	2.29	3.30	.090	.130
D	19.94	20.19	.785	.795
D1	19.05	19.25	.750	.758
D2	17.53	18.54	.690	.730
E	19.94	20.19	.785	.795
E1	19.05	19.25	.750	.758
E2	17.53	18.54	.690	.730
e	1.27	BSC	.050	BSC
G	1.07	1.22	.042	.048
H	1.07	1.42	.042	.056
J	0.51	-	.020	-
K	0.33	0.53	.013	.021
Nd	13		13	
Ne	13		13	
PKG STD	00			

32-lead QFN





	MM			INCH		
	MIN	NOM	MAX	MIN	NOM	MAX
A	-	0.85	0.90	-	.033	.035
J	0.00	0.01	0.05	.000	.000	.002
A1		0.20 ref			.008 ref	
D/E		7.00 BSC			.276 BSC	
D1/E1		6.75 BSC			.266 BSC	
D2/E2	4.95	5.10	5.25	.195	.201	.207
N	32					
P	0.24	0.42	0.60	.009	.016	.024
e		0.65 BSC			.026 BSC	
L	0.50	0.60	0.75	.020	.024	.030
b	0.23	0.28	0.35	.009	.011	.014

Table of Contents

Features ***1***

Description ***2***

Block Diagram ***3***

Pinout Description ***4***

Pinout 4

Signals 7

Typical Application ***12***

..... 12

Clock Controller ***13***

Introduction 13

Oscillator 13

PLL 14

Registers 16

SFR Mapping ***19***

Dual Data Pointer Register ***26***

Program/Code Memory ***28***

External Code Memory Access 28

Flash Memory Architecture 29

Overview of FM0 Operations 30

Registers 36

Flash EEPROM Memory ***37***

General Description 37

Features 37

Flash Programming and Erasure 37

Flash Registers and Memory Map 38

Flash Memory Status 41

Memory Organization 41

EEPROM Data Memory ***42***

Description 42

Write Data in the Column Latches 42

Programming 42

Read Data 42

Registers 43

In-System Programming (ISP) 44

Flash Programming and Erasure 44
Boot Process 45
Application-Programming-Interface 46
XROW Bytes 46
Hardware Conditions 46

On-chip Expanded RAM (ERAM) 48

Timer 2 51

Auto-reload Mode 51
Programmable Clock Output 52

Programmable Counter Array (PCA) 56

PCA Capture Mode 63
16-bit Software Timer/Compare Mode 63
High Speed Output Mode 64
Pulse Width Modulator Mode 65
PCA Watchdog Timer 66

Serial I/O Port 67

Framing Error Detection 67
Automatic Address Recognition 68
Baud Rate Selection for UART for Mode 1 and 3 70
UART Registers 73

Interrupt System 77

Overview 77
Registers 78
Interrupt Sources and Vector Addresses 85

Keyboard Interface 86

Introduction 86
Description 86
Registers 87

Programmable LED 90

Serial Peripheral Interface (SPI) 91

Features 91

Table of Contents

Signal Description 91
Functional Description 93

Two Wire Interface (TWI) 100
Description 102
Notes 105
Registers 115

USB Controller 117
Description 117
Configuration 120
Read/Write Data FIFO 122
Bulk/Interrupt Transactions 123
Control Transactions 127
Isochronous Transactions 128
Miscellaneous 130
Suspend/Resume Management 131
Detach Simulation 134
USB Interrupt System 134
USB Registers 137

Reset 149
Introduction 149
Reset Input 149
Reset Output 150

Power Monitor 151
Description 151

Power Management 153
Idle Mode 153
Power-down Mode 153
Registers 155

Hardware Watchdog Timer 156
Using the WDT 156
WDT During Power-down and Idle 157

ONCE Mode (ON Chip Emulation) 158

Reduced EMI Mode 159

Electrical Characteristics 160
Absolute Maximum Ratings 160
DC Parameters 160
USB DC Parameters 163
AC Parameters 164
USB AC Parameters 174

Ordering Information 175

Packaging Information 176
64-lead VQFP 176
52-lead PLCC 177
32-lead QFN 178

Atmel Corporation

2325 Orchard Parkway
San Jose, CA 95131, USA
Tel: 1(408) 441-0311
Fax: 1(408) 487-2600

Regional Headquarters

Europe
Atmel Sarl
Route des Arsenaux 41
Case Postale 80
CH-1705 Fribourg
Switzerland
Tel: (41) 26-426-5555
Fax: (41) 26-426-5500

Asia
Room 1219
Chinachem Golden Plaza
77 Mody Road Tsimshatsui
East Kowloon
Hong Kong
Tel: (852) 2721-9778
Fax: (852) 2722-1369

Japan
9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chuo-ku, Tokyo 104-0033
Japan
Tel: (81) 3-3523-3551
Fax: (81) 3-3523-7581

Atmel Operations

Memory
2325 Orchard Parkway
San Jose, CA 95131, USA
Tel: 1(408) 441-0311
Fax: 1(408) 436-4314

Microcontrollers
2325 Orchard Parkway
San Jose, CA 95131, USA
Tel: 1(408) 441-0311
Fax: 1(408) 436-4314

La Chantrerie
BP 70602
44306 Nantes Cedex 3, France
Tel: (33) 2-40-18-18-18
Fax: (33) 2-40-18-19-60

ASIC/ASSP/Smart Cards
Zone Industrielle
13106 Rousset Cedex, France
Tel: (33) 4-42-53-60-00
Fax: (33) 4-42-53-60-01

1150 East Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906, USA
Tel: 1(719) 576-3300
Fax: 1(719) 540-1759

Scottish Enterprise Technology Park
Maxwell Building
East Kilbride G75 0QR, Scotland
Tel: (44) 1355-803-000
Fax: (44) 1355-242-743

RF/Automotive
Theresienstrasse 2
Postfach 3535
74025 Heilbronn, Germany
Tel: (49) 71-31-67-0
Fax: (49) 71-31-67-2340

1150 East Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906, USA
Tel: 1(719) 576-3300
Fax: 1(719) 540-1759

Biometrics/Imaging/Hi-Rel MPU/
High Speed Converters/RF Datacom
Avenue de Rochepleine
BP 123
38521 Saint-Egreve Cedex, France
Tel: (33) 4-76-58-30-00
Fax: (33) 4-76-58-34-80

Literature Requests
www.atmel.com/literature

Disclaimer: Atmel Corporation makes no warranty for the use of its products, other than those expressly contained in the Company's standard warranty which is detailed in Atmel's Terms and Conditions located on the Company's web site. The Company assumes no responsibility for any errors which may appear in this document, reserves the right to change devices or specifications detailed herein at any time without notice, and does not make any commitment to update the information contained herein. No licenses to patents or other intellectual property of Atmel are granted by the Company in connection with the sale of Atmel products, expressly or by implication. Atmel's products are not authorized for use as critical components in life support devices or systems.

© **Atmel Corporation 2005**. **All rights reserved.** Atmel® and combinations thereof are the registered trademarks of Atmel Corporation or its subsidiaries. Other terms and product names may be the trademarks of others.

Printed on recycled paper.

4337C–USB–02/05