Semiconductors

UM10109

P89LPC932A1
8-bit microcontroller with two-clock 80C51 core

Rev. 02 — 23 May 2005 User manual

Document information

Info	Content
Keywords	P89LPC932, P89LPC932A1
Abstract	Technical information for the P89LPC932A1 device.

PHILIPS

Revision history

Rev	Date	Description
2	20050523	• Corrected typographical error in Table 35 “Capture compare control register (CCRx - address Exh) bit description”. • Corrected Table 92 “Data EEPROM control register (DEECON address F1h) bit allocation” and Table 93 “Data EEPROM control register (DEECON address F1h) bit description”. • Removed “with 8-bit A/D” from title. • Revised Table 37 “Output compare pin behavior” for OCMx1:0 =10.
1	20040802	Initial version

Contact information

For additional information, please visit: **http://www.semiconductors.philips.com**

For sales office addresses, please send an email to: **sales.addresses@www.semiconductors.philips.com**

1. Introduction

The P89LPC932A1 is a single-chip microcontroller designed for applications demanding high-integration, low cost solutions over a wide range of performance requirements. The P89LPC932A1 is based on a high performance processor architecture that executes instructions in two to four clocks, six times the rate of standard 80C51 devices. Many system-level functions have been incorporated into the P89LPC932A1 in order to reduce component count, board space, and system cost.

1.1 Comparison to the P89LPC932 device

The P89LPC932A1 includes several improvements compared to the P89LPC932. These improvements are described below.

1.1.1 Byte-erasability (IAP-Lite)

The original P89LPC932 allowed from 1 byte to 64 bytes of user code memory, in a single page, to be programmed using an IAP function call. The bytes to be programmed needed to have been previously erased using either a page erase, sector erase, or chip erase (in a parallel programmer) command. Thus code memory was erased in 64 byte, 1 kB, or 8 kB groups. The P89LPC932A1 allows from 1 byte to 64 bytes of a page of user code memory to be erased and reprogrammed in a single operation. The bytes to be erased and reprogrammed may be randomly addressed within a single page. Only the bytes so addressed will be affected. See Section 18.4 “Using Flash as data storage: IAP-Lite” on page 109.

1.1.2 Serial in-circuit programming (ICP)

In-Circuit Programming is a method intended to allow low cost commercial programmers to program and erase these devices without removing the microcontroller from the system. The In-Circuit Programming facility consists of a series of internal hardware resources to facilitate remote programming of the P89LPC932A1 through a two-wire serial interface. Philips has made in-circuit programming in an embedded application possible with a minimum of additional expense in components and circuit board area. The ICP function uses five pins (V_{DD}, V_{SS}, P0.5, P0.4, and $\overline{RST}$). Only a small connector needs to be available to interface your application to an external programmer in order to use this feature. This function was not available on the P89LPC932 device.

1.1.3 ‘On-the-fly’ clock selection

The RC Oscillator can be selected as the source for the CPU clock (CCLK) by using the RCCLK bit in the TRIM register (TRIM.7). This bit allows for fast ‘on-the-fly’ switching between the RC Oscillator and the clock source selected by the oscillator type select bits, FOSC[2:0], in UCFG1, without the need to reset the device. This functionality was not available on the P89LPC932. See Table 5 “On-chip RC oscillator trim register (TRIM - address 96h) bit description” on page 22.

1.1.4 Increased ISP/IAP functionality

1.1.4.1 Support for the watchdog timer

The ISP code has been modified to set the WDT prescaler (in WDCON) and WDL register to their maximum values. Other WDCON bits are unchanged and the ISP code does not explicitly enable or disable the WDT. Periodic feeds are provided within the ISP code to support applications that entered the ISP code with an enabled WDT. This functionality was not provided in the ISP code on the P89LPC932.

1.1.4.2 XDATA data buffer option added for programming code memory

The “program user code page” function on the P89LPC932 used IDATA as the 64 byte data buffer. An option is provided to allow the user to specify that XDATA is to be used instead as the buffer source. If the F1 flag (PSW.1) is set, then XDATA is used. If the F1 flag (PSW.1) is cleared, then IDATA is used.

1.1.4.3 Port 0 initialization

On the P89LPC932 the ISP code during initialization programmed all bits of Port 0 to the quasi-bidirectional mode and set these port pins HIGH. This has been changed such that only the TxD and RxD pins have their port mode programmed during ISP initialization. All other Port 0 pins remain in their previous state (for example, input-only mode following a reset).

1.1.4.4 Direct load of UART baud rate fix

A bug identified in the “direct load of baud rate” ISP function has been fixed. The baud rate source for this function has been changed from Timer 1 to the BRG.

1.1.4.5 Boot Vector and IAP entry points modified

To protect against errant code execution incrementing into the ISP or IAP routines, software reset instructions have been added to the beginning of these code blocks. This required that the ISP and IAP entry points be changed. The ISP entry point has changed to 1F00H resulting in a default Boot Vector of 1FH. The IAP entry point has changed to FF03H.

1.1.4.6 IAP authorization key

IAP functions which write or erase code memory require an authorization key be set by the calling routine prior to performing the IAP function call. This authorization key is set by writing 96H to RAM location FFH. See Section 18.13 “IAP authorization key” on page 118

After the function call is processed by the IAP routine, the authorization key will be cleared. Thus it is necessary for the authorization key to be set prior to EACH call to PGM_MTP that requires a key. If an IAP routine that requires an authorization key is called without a valid authorization key present, the MCU will perform a reset.

1.1.4.7 Hardware write enable (WE) key

This device has hardware write enable protection. This protection applies to both ISP and IAP modes and applies to both the user code memory space and the user configuration bytes (UCFG1, BOOTVEC, and BOOTSTAT). This protection does not apply to commercial programmer modes. When enabled, user code requesting a write function via IAP or IAP-Lite will need to explicitly set a Write Enable flag prior to requesting the write function. See Section 18.14 “Flash write enable” on page 119

1.1.4.8 Configuration byte protection

A separate write protection bit has been provided for the “configuration bytes”. These bytes include UCFG1, BootStat, Boot Vector, and the sector security bytes. This write protection applies for ISP and IAP modes. It does not apply to commercial programmer modes. See Section 18.15 “Configuration byte protection” on page 119

1.1.5 Previous errata fix

Most known errata on the P89LPC932 devices has been fixed on the P89LPC932A1 device. For current errata information on the P89LPC932A1, if any, please see the P89LPC932A1 errata sheet.

1.2 Pin configuration

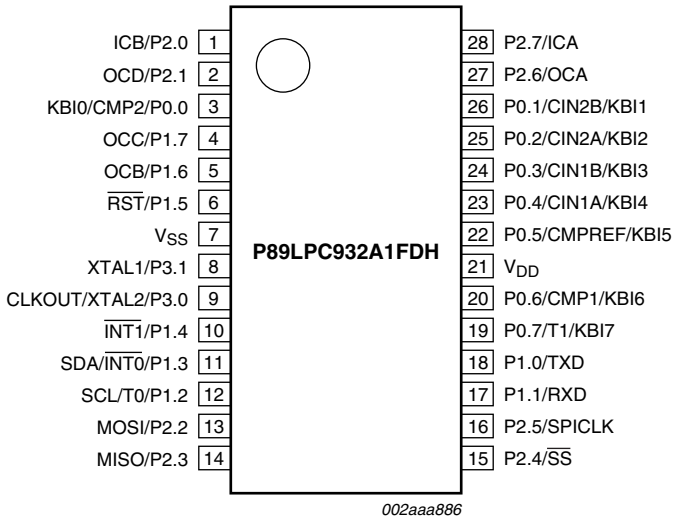


Fig 1. P89LPC932A1 TSSOP28 pin configuration.

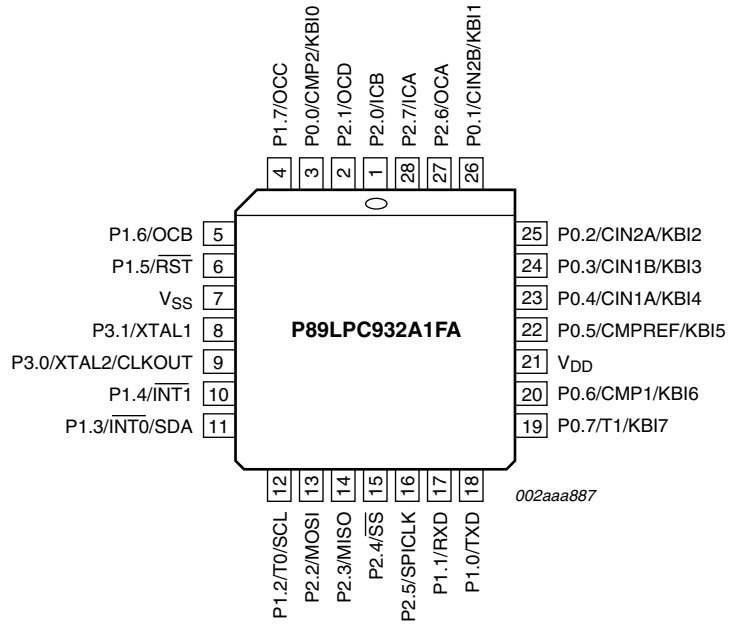


Fig 2. P89LPC932A1 PLCC28 pin configuration.

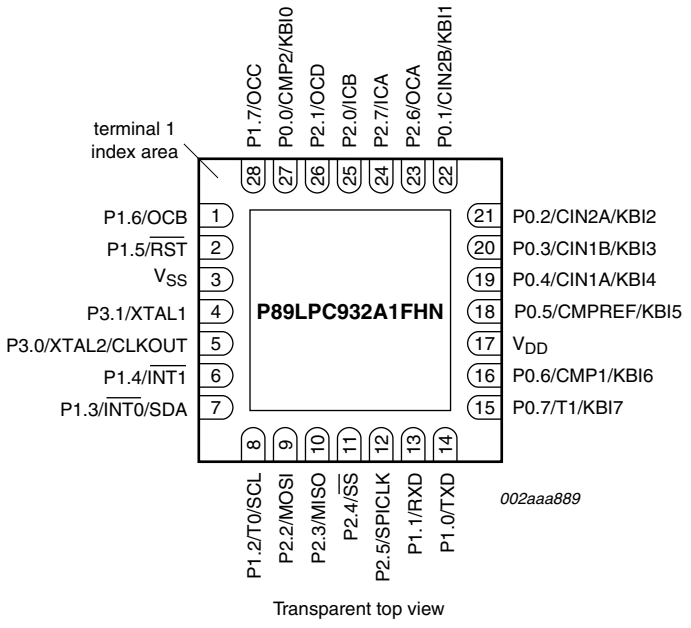


Fig 3. P89LPC932A1 HVQFN28 pin configuration.

1.3 Pin description

Table 1: Pin description

Symbol	Pin		Type	Description
	TSSOP28, PLCC28	HVQFN28		
P0.0 to P0.7	3, 26, 25, 24, 23, 22, 20, 19	27, 22, 21, 20, 19, 18, 16, 15	I/O	**Port 0:** Port 0 is an 8-bit I/O port with a user-configurable output type. During reset Port 0 latches are configured in the input only mode with the internal pull-up disabled. The operation of Port 0 pins as inputs and outputs depends upon the port configuration selected. Each port pin is configured independently. Refer to Section 4.1 “Port configurations” and the *P89LPC932A1 data sheet, Static characteristics* for details. The Keypad Interrupt feature operates with Port 0 pins. All pins have Schmitt triggered inputs. Port 0 also provides various special functions as described below:
	3	27	I/O	**P0.0 —** Port 0 bit 0.
			O	**CMP2 —** Comparator 2 output.
			I	**KBI0 —** Keyboard input 0.
	26	22	I/O	**P0.1 —** Port 0 bit 1.
			I	**CIN2B —** Comparator 2 positive input B.
			I	**KBI1 —** Keyboard input 1.
	25	21	I/O	**P0.2 —** Port 0 bit 2.
			I	**CIN2A —** Comparator 2 positive input A.
			I	**KBI2 —** Keyboard input 2.
	24	20	I/O	**P0.3 —** Port 0 bit 3.
			I	**CIN1B —** Comparator 1 positive input B.
			I	**KBI3 —** Keyboard input 3.
	23	19	I/O	**P0.4 —** Port 0 bit 4.
			I	**CIN1A —** Comparator 1 positive input A.
			I	**KBI4 —** Keyboard input 4.
	22	18	I/O	**P0.5 —** Port 0 bit 5.
			I	**CMPREF —** Comparator reference (negative) input.
			I	**KBI5 —** Keyboard input 5.

Table 1: Pin description *…continued*

Symbol	Pin		Type	Description
	TSSOP28, PLCC28	HVQFN28		
P0.0 to P0.7 (continued)	20	16	I/O	**P0.6 —** Port 0 bit 6.
			O	**CMP1 —** Comparator 1 output.
			I	**KBI6 —** Keyboard input 6.
	19	15	I/O	**P0.7 —** Port 0 bit 7.
			I/O	**T1 —** Timer/counter 1 external count input or overflow output.
			I	**KBI7 —** Keyboard input 7.

Table 1: Pin description *...continued*

Symbol	Pin		Type	Description
	TSSOP28, PLCC28	HVQFN28		
P1.0 to P1.7	18, 17, 12, 11, 10, 6, 5, 4	14, 13, 8, 7, 6, 2, 1, 28	I/O, I [1]	**Port 1:** Port 1 is an 8-bit I/O port with a user-configurable output type, except for three pins as noted below. During reset Port 1 latches are configured in the input only mode with the internal pull-up disabled. The operation of the configurable Port 1 pins as inputs and outputs depends upon the port configuration selected. Each of the configurable port pins are programmed independently. Refer to Section 4.1 "Port configurations" and the *P89LPC932A1 data sheet, Static characteristics* for details. P1.2 to P1.3 are open drain when used as outputs. P1.5 is input only. All pins have Schmitt triggered inputs. Port 1 also provides various special functions as described below:
	18	14	I/O	**P1.0 —** Port 1 bit 0.
			O	**TXD —** Transmitter output for the serial port.
	17	13	I/O	**P1.1 —** Port 1 bit 1.
			I	**RXD —** Receiver input for the serial port.
	12	8	I/O	**P1.2 —** Port 1 bit 2 (open-drain when used as output).
			I/O	**T0 —** Timer/counter 0 external count input or overflow output (open-drain when used as output).
			I/O	**SCL —** I^2C serial clock input/output.
	11	7	I/O	**P1.3 —** Port 1 bit 3 (open-drain when used as output).
			I	**$\overline{INT0}$ —** External interrupt 0 input.
			I/O	**SDA —** I^2C serial data input/output.
	10	6	I	**P1.4 —** Port 1 bit 4.
			I	**$\overline{INT1}$ —** External interrupt 1 input.
	6	2	I	**P1.5 —** Port 1 bit 5 (input only).
			I	**$\overline{RST}$ —** External Reset input during power-on or if selected via UCFG1. When functioning as a reset input, a LOW on this pin resets the microcontroller, causing I/O ports and peripherals to take on their default states, and the processor begins execution at address 0. Also used during a power-on sequence to force In-System Programming mode. **When using an oscillator frequency above 12 MHz, the reset input function of P1.5 must be enabled. An external circuit is required to hold the device in reset at powerup until V_{DD} has reached its specified level. When system power is removed V_{DD} will fall below the minimum specified operating voltage. When using an oscillator frequency above 12 MHz, in some applications, an external brownout detect circuit may be required to hold the device in reset when V_{DD} falls below the minimum specified operating voltage.**
	5	1	I/O	**P1.6 —** Port 1 bit 6.
			O	**OCB —** Output Compare B
	4	28	I/O	**P1.7 —** Port 1 bit 7.
			O	**OCC —** Output Compare C

Table 1: Pin description *…continued*

Symbol	Pin		Type	Description
	TSSOP28, PLCC28	HVQFN28		
P2.0 to P2.7	1, 2, 13, 14, 15, 16, 27, 28	25, 26, 9, 10, 11, 12, 23, 24	I/O	**Port 2:** Port 2 is an 8-bit I/O port with a user-configurable output type. During reset Port 2 latches are configured in the input only mode with the internal pull-up disabled. The operation of Port 2 pins as inputs and outputs depends upon the port configuration selected. Each port pin is configured independently. Refer to Section 4.1 "Port configurations" and the *P89LPC932A1 data sheet, Static characteristics* for details. All pins have Schmitt triggered inputs. Port 2 also provides various special functions as described below:
	1	25	I/O	**P2.0 —** Port 2 bit 0.
			I	**ICB —** Input Capture B
	2	26	I/O	**P2.1 —** Port 2 bit 1.
			O	**OCD —** Output Compare D
	13	9	I/O	**P2.2 —** Port 2 bit 2.
			I/O	**MOSI —** SPI master out slave in. When configured as master, this pin is output; when configured as slave, this pin is input.
	14	10	I/O	**P2.3 —** Port 2 bit 3.
			I/O	**MISO —** When configured as master, this pin is input, when configured as slave, this pin is output.
	15	11	I/O	**P2.4 —** Port 2 bit 4.
			I	$\overline{\textbf{SS}}$ **—** SPI Slave select.
	16	12	I/O	**P2.5 —** Port 2 bit 5.
			I/O	**SPICLK —** SPI clock. When configured as master, this pin is output; when configured as slave, this pin is input.
	27	23	I/O	**P2.6 —** Port 2 bit 6.
			O	**OCA —** Output Compare A
	28	24	I/O	**P2.7 —** Port 2 bit 7.
			I	**ICA —** Input Capture A

Table 1: Pin description *...continued*

Symbol	Pin		Type	Description
	TSSOP28, PLCC28	HVQFN28		
P3.0 to P3.1	9, 8	5, 4	I/O	**Port 3:** Port 3 is a 2-bit I/O port with a user-configurable output type. During reset Port 3 latches are configured in the input only mode with the internal pull-up disabled. The operation of Port 3 pins as inputs and outputs depends upon the port configuration selected. Each port pin is configured independently. Refer to Section 4.1 and the *P89LPC932A1 data sheet, Static characteristics* for details. All pins have Schmitt triggered inputs. Port 3 also provides various special functions as described below:
	9	5	I/O	**P3.0 —** Port 3 bit 0.
			O	**XTAL2 —** Output from the oscillator amplifier (when a crystal oscillator option is selected via the FLASH configuration.
			O	**CLKOUT —** CPU clock divided by 2 when enabled via SFR bit (ENCLK - TRIM.6). It can be used if the CPU clock is the internal RC oscillator, watchdog oscillator or external clock input, except when XTAL1/XTAL2 are used to generate clock source for the Real-Time clock/system timer.
	8	4	I/O	**P3.1 —** Port 3 bit 1.
			I	**XTAL1 —** Input to the oscillator circuit and internal clock generator circuits (when selected via the FLASH configuration). It can be a port pin if internal RC oscillator or watchdog oscillator is used as the CPU clock source, **and** if XTAL1/XTAL2 are not used to generate the clock for the Real-Time clock/system timer.
V_{SS}	7	3	I	**Ground:** 0 V reference.
V_{DD}	21	17	I	**Power Supply:** This is the power supply voltage for normal operation as well as Idle and Power-down modes.

[1] Input/Output for P1.0 to P1.4, P1.6, P1.7. Input for P1.5.

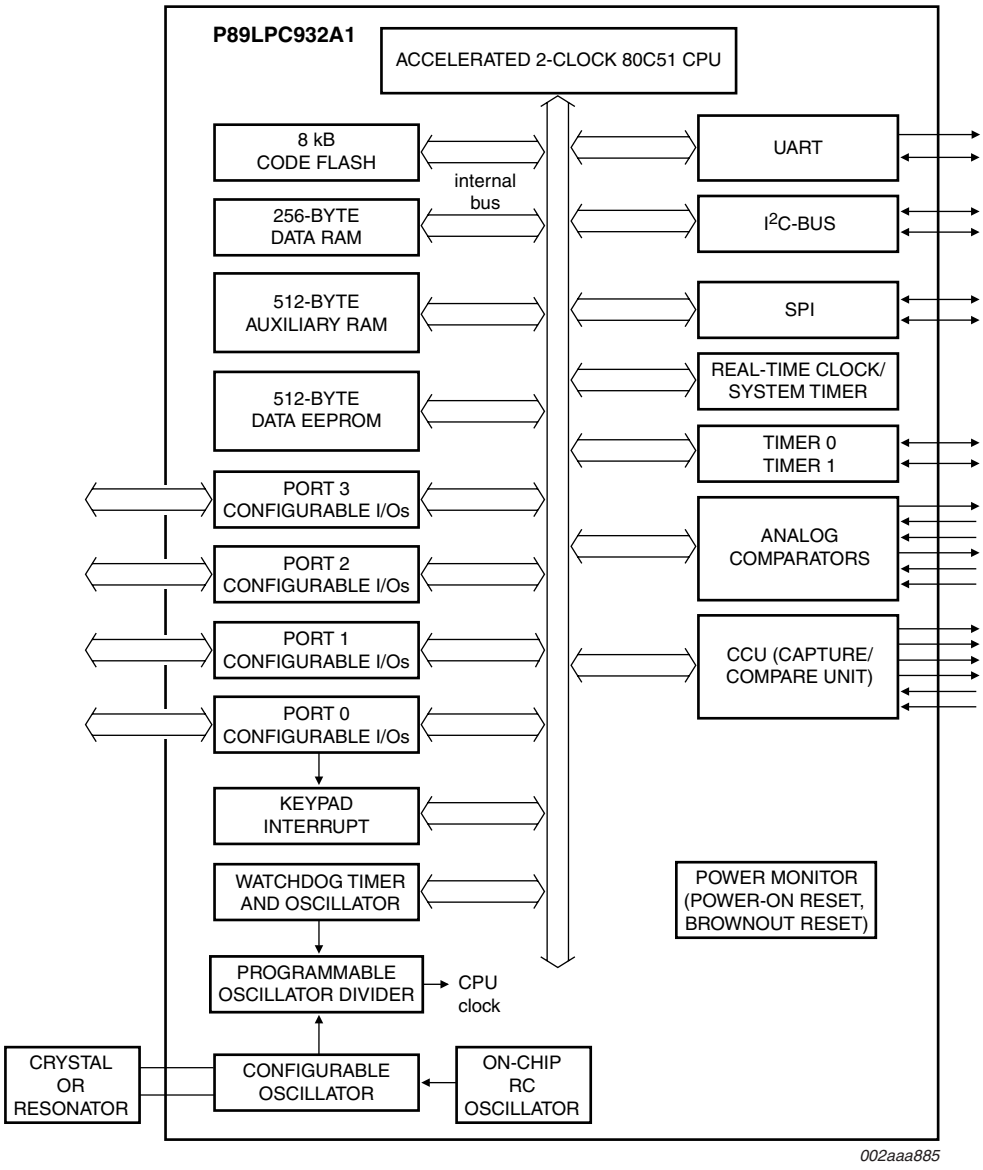


Fig 4. P89LPC932A1 block diagram.

1.4 Special function registers

Remark: Special Function Registers (SFRs) accesses are restricted in the following ways:

- User must **not** attempt to access any SFR locations not defined.
- Accesses to any defined SFR locations must be strictly for the functions for the SFRs.
- SFR bits labeled '-', '0' or '1' can **only** be written and read as follows:
  - '-' Unless otherwise specified, **must** be written with '0', but can return any value when read (even if it was written with '0'). It is a reserved bit and may be used in future derivatives.
  - '0' **must** be written with '0', and will return a '0' when read.
  - '1' **must** be written with '1', and will return a '1' when read.

Table 2: P89LPC932A1 Special function registers

** indicates SFRs that are bit addressable.*

Name	Description	SFR addr.	Bit functions and addresses								Reset value	
			MSB							LSB	Hex	Binary
	Bit address		E7	E6	E5	E4	E3	E2	E1	E0		
ACC*	Accumulator	E0H									00	0000 0000
AUXR1	Auxiliary function register	A2H	CLKLP	EBRR	ENT1	ENT0	SRST	0	-	DPS	00	0000 00x0
	Bit address		F7	F6	F5	F4	F3	F2	F1	F0		
B*	B register	F0H									00	0000 0000
BRGR0[2]	Baud rate generator rate low	BEH									00	0000 0000
BRGR1[2]	Baud rate generator rate high	BFH									00	0000 0000
BRGCON	Baud rate generator control	BDH	-	-	-	-	-	-	SBRGS	BRGEN	00[2]	xxxx xx00
CCCRA	Capture compare A control register	EAH	ICECA2	ICECA1	ICECA0	ICESA	ICNFA	FCOA	OCMA1	OCMA0	00	0000 0000
CCCRB	Capture compare B control register	EBH	ICECB2	ICECB1	ICECB0	ICESB	ICNFB	FCOB	OCMB1	OCMB0	00	0000 0000
CCCRC	Capture compare C control register	ECH	-	-	-	-	-	FCOC	OCMC1	OCMC0	00	xxxx x000
CCCRD	Capture compare D control register	EDH	-	-	-	-	-	FCOD	OCMD1	OCMD0	00	xxxx x000
CMP1	Comparator 1 control register	ACH	-	-	CE1	CP1	CN1	OE1	CO1	CMF1	00[1]	xx00 0000
CMP2	Comparator 2 control register	ADH	-	-	CE2	CP2	CN2	OE2	CO2	CMF2	00[1]	xx00 0000
DEECON	Data EEPROM control register	F1H	EEIF	HVERR	ECTL1	ECTL0	-	-	-	EADR8	0E	0000 1110
DEEDAT	Data EEPROM data register	F2H									00	0000 0000
DEEADR	Data EEPROM address register	F3H									00	0000 0000
DIVM	CPU clock divide-by-M control	95H									00	0000 0000
DPTR	Data pointer (2 bytes)											

Table 2: P89LPC932A1 Special function registers *…continued*

** indicates SFRs that are bit addressable.*

Name	Description	SFR addr.	Bit functions and addresses								Reset value	
			MSB							LSB	Hex	Binary
DPH	Data pointer high	83H									00	0000 0000
DPL	Data pointer low	82H									00	0000 0000
I2ADR	I^2C slave address register	DBH	I2ADR.6	I2ADR.5	I2ADR.4	I2ADR.3	I2ADR.2	I2ADR.1	I2ADR.0	GC	00	0000 0000
	Bit address		**DF**	**DE**	**DD**	**DC**	**DB**	**DA**	**D9**	**D8**		
I2CON*	I^2C control register	D8H	-	I2EN	STA	STO	SI	AA	-	CRSEL	00	x000 00x0
I2DAT	I^2C data register	DAH										
I2SCLH	Serial clock generator/SCL duty cycle register high	DDH									00	0000 0000
I2SCLL	Serial clock generator/SCL duty cycle register low	DCH									00	0000 0000
I2STAT	I^2C status register	D9H	STA.4	STA.3	STA.2	STA.1	STA.0	0	0	0	F8	1111 1000
ICRAH	Input capture A register high	ABH									00	0000 0000
ICRAL	Input capture A register low	AAH									00	0000 0000
ICRBH	Input capture B register high	AFH									00	0000 0000
ICRBL	Input capture B register low	AEH									00	0000 0000
	Bit address		**AF**	**AE**	**AD**	**AC**	**AB**	**AA**	**A9**	**A8**		
IEN0*	Interrupt enable 0	A8H	EA	EWDRT	EBO	ES/ESR	ET1	EX1	ET0	EX0	00	0000 0000
	Bit address		**EF**	**EE**	**ED**	**EC**	**EB**	**EA**	**E9**	**E8**		
IEN1*	Interrupt enable 1	E8H	EIEE	EST	-	ECCU	ESPI	EC	EKBI	EI2C	00[1]	00x0 0000
	Bit address		**BF**	**BE**	**BD**	**BC**	**BB**	**BA**	**B9**	**B8**		
IP0*	Interrupt priority 0	B8H	-	PWDRT	PBO	PS/PSR	PT1	PX1	PT0	PX0	00[1]	x000 0000
IP0H	Interrupt priority 0 high	B7H	-	PWDRT H	PBOH	PSH/ PSRH	PT1H	PX1H	PT0H	PX0H	00[1]	x000 0000
	Bit address		**FF**	**FE**	**FD**	**FC**	**FB**	**FA**	**F9**	**F8**		
IP1*	Interrupt priority 1	F8H	PIEE	PST	-	PCCU	PSPI	PC	PKBI	PI2C	00[1]	00x0 0000
IP1H	Interrupt priority 1 high	F7H	PIEEH	PSTH	-	PCCUH	PSPIH	PCH	PKBIH	PI2CH	00[1]	00x0 0000

Table 2: P89LPC932A1 Special function registers *…continued*

** indicates SFRs that are bit addressable.*

Name	Description	SFR addr.	Bit functions and addresses								Reset value	
			MSB							LSB	Hex	Binary
KBCON	Keypad control register	94H	-	-	-	-	-	-	PATN_SEL	KBIF	00[1]	xxxx xx00
KBMASK	Keypad interrupt mask register	86H									00	0000 0000
KBPATN	Keypad pattern register	93H									FF	1111 1111
OCRAH	Output compare A register high	EFH									00	0000 0000
OCRAL	Output compare A register low	EEH									00	0000 0000
OCRBH	Output compare B register high	FBH									00	0000 0000
OCRBL	Output compare B register low	FAH									00	0000 0000
OCRCH	Output compare C register high	FDH									00	0000 0000
OCRCL	Output compare C register low	FCH									00	0000 0000
OCRDH	Output compare D register high	FFH									00	0000 0000
OCRDL	Output compare D register low	FEH									00	0000 0000
	Bit address		**87**	**86**	**85**	**84**	**83**	**82**	**81**	**80**		
P0*	Port 0	80H	T1/KB7	CMP1 /KB6	CMPREF /KB5	CIN1A /KB4	CIN1B /KB3	CIN2A /KB2	CIN2B /KB1	CMP2 /KB0		[1]
	Bit address		**97**	**96**	**95**	**94**	**93**	**92**	**91**	**90**		
P1*	Port 1	90H	OCC	OCB	$\overline{\text{RST}}$	$\overline{\text{INT1}}$	$\overline{\text{INT0}}$/ SDA	T0/SCL	RXD	TXD		[1]
	Bit address		**97**	**96**	**95**	**94**	**93**	**92**	**91**	**90**		
P2*	Port 2	A0H	ICA	OCA	SPICLK	$\overline{\text{SS}}$	MISO	MOSI	OCD	ICB		[1]
	Bit address		**B7**	**B6**	**B5**	**B4**	**B3**	**B2**	**B1**	**B0**		
P3*	Port 3	B0H	-	-	-	-	-	-	XTAL1	XTAL2		[1]
P0M1	Port 0 output mode 1	84H	(P0M1.7)	(P0M1.6)	(P0M1.5)	(P0M1.4)	(P0M1.3)	(P0M1.2)	(P0M1.1)	(P0M1.0)	FF[1]	1111 1111

Table 2: P89LPC932A1 Special function registers *…continued*

** indicates SFRs that are bit addressable.*

Name	Description	SFR addr.	Bit functions and addresses								Reset value	
			MSB							LSB	Hex	Binary
P0M2	Port 0 output mode 2	85H	(P0M2.7)	(P0M2.6)	(P0M2.5)	(P0M2.4)	(P0M2.3)	(P0M2.2)	(P0M2.1)	(P0M2.0)	00[1]	0000 0000
P1M1	Port 1 output mode 1	91H	(P1M1.7)	(P1M1.6)	-	(P1M1.4)	(P1M1.3)	(P1M1.2)	(P1M1.1)	(P1M1.0)	D3[1]	11x1 xx11
P1M2	Port 1 output mode 2	92H	(P1M2.7)	(P1M2.6)	-	(P1M2.4)	(P1M2.3)	(P1M2.2)	(P1M2.1)	(P1M2.0)	00[1]	00x0 xx00
P2M1	Port 2 output mode 1	A4H	(P2M1.7)	(P2M1.6)	(P2M1.5)	(P2M1.4)	(P2M1.3)	(P2M1.2)	(P2M1.1)	(P2M1.0)	FF[1]	1111 1111
P2M2	Port 2 output mode 2	A5H	(P2M2.7)	(P2M2.6)	(P2M2.5)	(P2M2.4)	(P2M2.3)	(P2M2.2)	(P2M2.1)	(P2M2.0)	00[1]	0000 0000
P3M1	Port 3 output mode 1	B1H	-	-	-	-	-	-	(P3M1.1)	(P3M1.0)	03[1]	xxxx xx11
P3M2	Port 3 output mode 2	B2H	-	-	-	-	-	-	(P3M2.1)	(P3M2.0)	00[1]	xxxx xx00
PCON	Power control register	87H	SMOD1	SMOD0	BOPD	BOI	GF1	GF0	PMOD1	PMOD0	00	0000 0000
PCONA	Power control register A	B5H	RTCPD	DEEPD	VCPD	-	I2PD	SPPD	SPD	CCUPD	00[1]	0000 0000
	Bit address		**D7**	**D6**	**D5**	**D4**	**D3**	**D2**	**D1**	**D0**		
PSW*	Program status word	D0H	CY	AC	F0	RS1	RS0	OV	F1	P	00	0000 0000
PT0AD	Port 0 digital input disable	F6H	-	-	PT0AD.5	PT0AD.4	PT0AD.3	PT0AD.2	PT0AD.1	-	00	xx00 000x
RSTSRC	Reset source register	DFH	-	-	BOF	POF	R_BK	R_WD	R_SF	R_EX		[3]
RTCCON	Real-time clock control	D1H	RTCF	RTCS1	RTCS0	-	-	-	ERTC	RTCEN	60[1][6]	011x xx00
RTCH	Real-time clock register high	D2H									00[6]	0000 0000
RTCL	Real-time clock register low	D3H									00[6]	0000 0000
SADDR	Serial port address register	A9H									00	0000 0000
SADEN	Serial port address enable	B9H									00	0000 0000
SBUF	Serial Port data buffer register	99H									xx	xxxx xxxx

Table 2: P89LPC932A1 Special function registers *…continued*

** indicates SFRs that are bit addressable.*

Name	Description	SFR addr.	Bit functions and addresses								Reset value	
			MSB							LSB	Hex	Binary
	Bit address		9F	9E	9D	9C	9B	9A	99	98		
SCON*	Serial port control	98H	SM0/FE	SM1	SM2	REN	TB8	RB8	TI	RI	00	0000 0000
SSTAT	Serial port extended status register	BAH	DBMOD	INTLO	CIDIS	DBISEL	FE	BR	OE	STINT	00	0000 0000
SP	Stack pointer	81H									07	0000 0111
SPCTL	SPI control register	E2H	SSIG	SPEN	DORD	MSTR	CPOL	CPHA	SPR1	SPR0	04	0000 0100
SPSTAT	SPI status register	E1H	SPIF	WCOL	-	-	-	-	-	-	00	00xx xxxx
SPDAT	SPI data register	E3H									00	0000 0000
TAMOD	Timer 0 and 1 auxiliary mode	8FH	-	-	-	T1M2	-	-	-	T0M2	00	xxx0 xxx0
	Bit address		8F	8E	8D	8C	8B	8A	89	88		
TCON*	Timer 0 and 1 control	88H	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0	00	0000 0000
TCR20*	CCU control register 0	C8H	PLEEN	HLTRN	HLTEN	ALTCD	ALTAB	TDIR2	TMOD21	TMOD20	00	0000 0000
TCR21	CCU control register 1	F9H	TCOU2	-	-	-	PLLDV.3	PLLDV.2	PLLDV.1	PLLDV.0	00	0xxx 0000
TH0	Timer 0 high	8CH									00	0000 0000
TH1	Timer 1 high	8DH									00	0000 0000
TH2	CCU timer high	CDH									00	0000 0000
TICR2	CCU interrupt control register	C9H	TOIE2	TOCIE2D	TOCIE2C	TOCIE2B	TOCIE2A	-	TICIE2B	TICIE2A	00	0000 0x00
TIFR2	CCU interrupt flag register	E9H	TOIF2	TOCF2D	TOCF2C	TOCF2B	TOCF2A	-	TICF2B	TICF2A	00	0000 0x00
TISE2	CCU interrupt status encode register	DEH	-	-	-	-	-	ENCINT.2	ENCINT.1	ENCINT.0	00	xxxx x000
TL0	Timer 0 low	8AH									00	0000 0000
TL1	Timer 1 low	8BH									00	0000 0000
TL2	CCU timer low	CCH									00	0000 0000
TMOD	Timer 0 and 1 mode	89H	T1GATE	T1C/$\overline{T}$	T1M1	T1M0	T0GATE	T0C/$\overline{T}$	T0M1	T0M0	00	0000 0000
TOR2H	CCU reload register high	CFH									00	0000 0000
TOR2L	CCU reload register low	CEH									00	0000 0000
TPCR2H	Prescaler control register high	CBH	-	-	-	-	-	-	TPCR2H.1	TPCR2H.0	00	xxxx xx00

Table 2: P89LPC932A1 Special function registers *…continued*

** indicates SFRs that are bit addressable.*

Name	Description	SFR addr.	Bit functions and addresses								Reset value	
			MSB							LSB	Hex	Binary
TPCR2L	Prescaler control register low	CAH	TPCR2L.7	TPCR2L.6	TPCR2L.5	TPCR2L.4	TPCR2L.3	TPCR2L.2	TPCR2L.1	TPCR2L.0	00	0000 0000
TRIM	Internal oscillator trim register	96H	RCCLK	ENCLK	TRIM.5	TRIM.4	TRIM.3	TRIM.2	TRIM.1	TRIM.0		[5] [6]
WDCON	Watchdog control register	A7H	PRE2	PRE1	PRE0	-	-	WDRUN	WDTOF	WDCLK		[4] [6]
WDL	Watchdog load	C1H									FF	1111 1111
WFEED1	Watchdog feed 1	C2H										
WFEED2	Watchdog feed 2	C3H										

[1] All ports are in input only (high-impedance) state after power-up.

[2] BRGR1 and BRGR0 must only be written if BRGEN in BRGCON SFR is logic 0. If any are written while BRGEN = 1, the result is unpredictable.

[3] The RSTSRC register reflects the cause of the P89LPC932A1 reset. Upon a power-up reset, all reset source flags are cleared except POF and BOF; the power-on reset value is xx110000.

[4] After reset, the value is 111001x1, i.e., PRE2-PRE0 are all logic 1, WDRUN = 1 and WDCLK = 1. WDTOF bit is logic 1 after watchdog reset and is logic 0 after power-on reset. Other resets will not affect WDTOF.

[5] On power-on reset, the TRIM SFR is initialized with a factory preprogrammed value. Other resets will not cause initialization of the TRIM register.

[6] The only reset source that affects these SFRs is power-on reset.

1.5 Memory organization

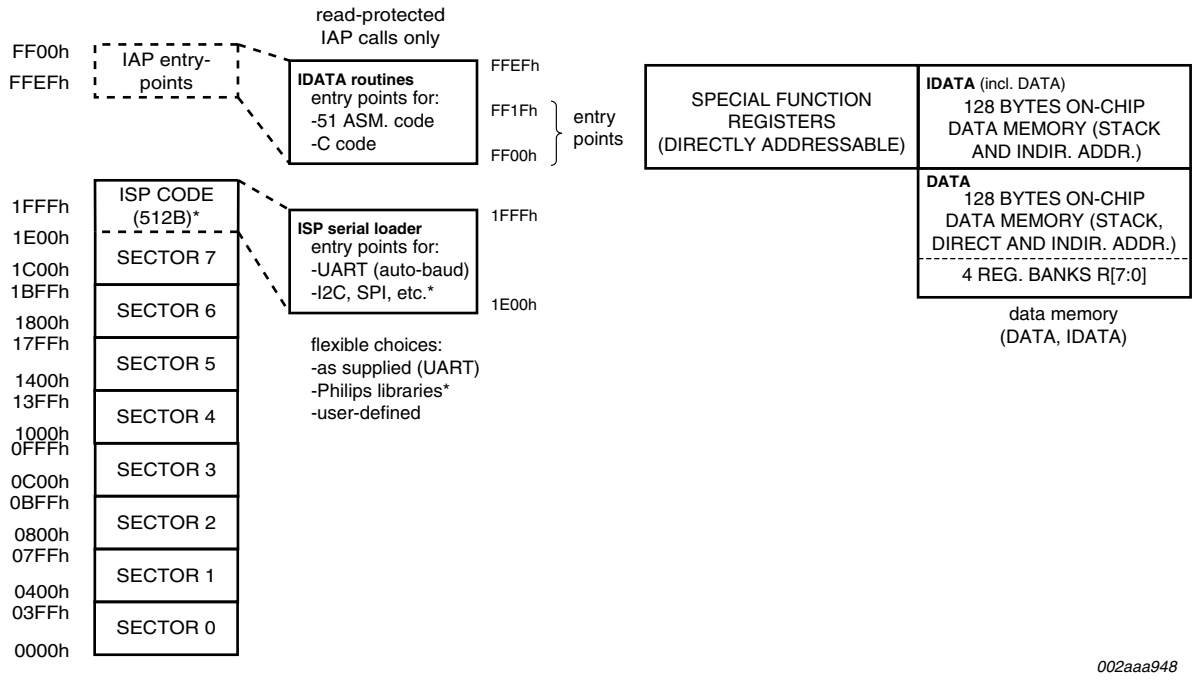


Fig 5. P89LPC932A1 memory map.

The various P89LPC932A1 memory spaces are as follows:

DATA — 128 bytes of internal data memory space (00h:7Fh) accessed via direct or indirect addressing, using instruction other than MOVX and MOVC. All or part of the Stack may be in this area.

IDATA — Indirect Data. 256 bytes of internal data memory space (00h:FFh) accessed via indirect addressing using instructions other than MOVX and MOVC. All or part of the Stack may be in this area. This area includes the DATA area and the 128 bytes immediately above it.

SFR — Special Function Registers. Selected CPU registers and peripheral control and status registers, accessible only via direct addressing.

CODE — 64 kB of Code memory space, accessed as part of program execution and via the MOVC instruction. The P89LPC932A1 has 8 kB of on-chip Code memory.

Table 3: Data RAM arrangement

Type	Data RAM	Size (bytes)
DATA	Directly and indirectly addressable memory	128
IDATA	Indirectly addressable memory	256

2. Clocks

2.1 Enhanced CPU

The P89LPC932A1 uses an enhanced 80C51 CPU which runs at six times the speed of standard 80C51 devices. A machine cycle consists of two CPU clock cycles, and most instructions execute in one or two machine cycles.

2.2 Clock definitions

The P89LPC932A1 device has several internal clocks as defined below:

OSCCLK — Input to the DIVM clock divider. OSCCLK is selected from one of four clock sources and can also be optionally divided to a slower frequency (see Figure 6 and Section 2.8 “CPU Clock (CCLK) modification: DIVM register”). **Note:** f_{osc} is defined as the OSCCLK frequency.

CCLK — CPU clock; output of the DIVM clock divider. There are two CCLK cycles per machine cycle, and most instructions are executed in one to two machine cycles (two or four CCLK cycles).

RCCLK — The internal 7.373 MHz RC oscillator output.

PCLK — Clock for the various peripheral devices and is $^{CCLK}/_{2}$.

2.2.1 Oscillator Clock (OSCCLK)

The P89LPC932A1 provides several user-selectable oscillator options. This allows optimization for a range of needs from high precision to lowest possible cost. These options are configured when the FLASH is programmed and include an on-chip watchdog oscillator, an on-chip RC oscillator, an oscillator using an external crystal, or an external clock source. The crystal oscillator can be optimized for low, medium, or high frequency crystals covering a range from 20 kHz to 12 MHz.

2.2.2 Low speed oscillator option

This option supports an external crystal in the range of 20 kHz to 100 kHz. Ceramic resonators are also supported in this configuration.

2.2.3 Medium speed oscillator option

This option supports an external crystal in the range of 100 kHz to 4 MHz. Ceramic resonators are also supported in this configuration.

2.2.4 High speed oscillator option

This option supports an external crystal in the range of 4 MHz to 12 MHz. Ceramic resonators are also supported in this configuration.

2.3 Clock output

The P89LPC932A1 supports a user-selectable clock output function on the XTAL2 / CLKOUT pin when the crystal oscillator is not being used. This condition occurs if a different clock source has been selected (on-chip RC oscillator, watchdog oscillator, external clock input on X1) and if the Real-time Clock is not using the crystal oscillator as its clock source. This allows external devices to synchronize to the P89LPC932A1. This output is enabled by the ENCLK bit in the TRIM register

The frequency of this clock output is $^1/_2$ that of the CCLK. If the clock output is not needed in Idle mode, it may be turned off prior to entering Idle, saving additional power. Note: on reset, the TRIM SFR is initialized with a factory preprogrammed value. Therefore when setting or clearing the ENCLK bit, the user should retain the contents of other bits of the TRIM register. This can be done by reading the contents of the TRIM register (into the ACC for example), modifying bit 6, and writing this result back into the TRIM register. Alternatively, the 'ANL direct' or 'ORL direct' instructions can be used to clear or set bit 6 of the TRIM register.

2.4 On-chip RC oscillator option

The P89LPC932A1 has a TRIM register that can be used to tune the frequency of the RC oscillator. During reset, the TRIM value is initialized to a factory pre-programmed value to adjust the oscillator frequency to 7.373 MHz, ± 1 %. (Note: the initial value is better than 1 %; please refer to the *P89LPC932A1 data sheet* for behavior over temperature). End user applications can write to the TRIM register to adjust the on-chip RC oscillator to other frequencies. Increasing the TRIM value will decrease the oscillator frequency.

Table 4: On-chip RC oscillator trim register (TRIM - address 96h) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	RCCLK	ENCLK	TRIM.5	TRIM.4	TRIM.3	TRIM.2	TRIM.1	TRIM.0
Reset	0	0	Bits 5:0 loaded with factory stored value during reset.					

Table 5: On-chip RC oscillator trim register (TRIM - address 96h) bit description

Bit	Symbol	Description
0	TRIM.0	Trim value. Determines the frequency of the internal RC oscillator. During reset, these bits are loaded with a stored factory calibration value. When writing to either bit 6 or bit 7 of this register, care should be taken to preserve the current TRIM value by reading this register, modifying bits 6 or 7 as required, and writing the result to this register.
1	TRIM.1	
2	TRIM.2	
3	TRIM.3	
4	TRIM.4	
5	TRIM.5	
6	ENCLK	when = 1, $^{CCLK}/_2$ is output on the XTAL2 pin provided the crystal oscillator is not being used.
7	RCCLK	when = 1, selects the RC Oscillator output as the CPU clock (CCLK). This allows for fast switching between any clock source and the internal RC oscillator without needing to go through a reset cycle. **The original P89LPC932 required a reset cycle in order to switch between clock sources.**

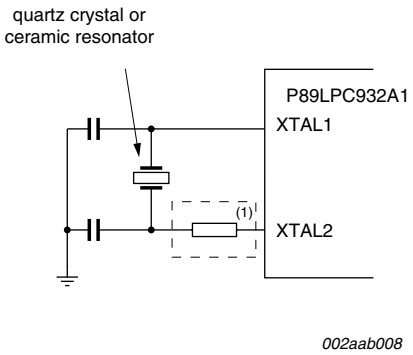
2.5 Watchdog oscillator option

The watchdog has a separate oscillator which has a frequency of 400 kHz. This oscillator can be used to save power when a high clock frequency is not needed.

2.6 External clock input option

In this configuration, the processor clock is derived from an external source driving the XTAL1 / P3.1 pin. The rate may be from 0 Hz up to 18 MHz. The XTAL2 / P3.0 pin may be used as a standard port pin or a clock output. **When using an oscillator frequency above 12 MHz, the reset input function of P1.5 must be enabled. An external circuit is required to hold the device in reset at powerup until V_{DD} has reached its specified**

level. When system power is removed V_{DD} will fall below the minimum specified operating voltage. When using an oscillator frequency above 12 MHz, in some applications, an external brownout detect circuit may be required to hold the device in reset when VDD falls below the minimum specified operating voltage.



Note: The oscillator must be configured in one of the following modes: Low frequency crystal, medium frequency crystal, or high frequency crystal.

(1) A series resistor may be required to limit crystal drive levels. This is especially important for low frequency crystals (see text).

Fig 6. Using the crystal oscillator.

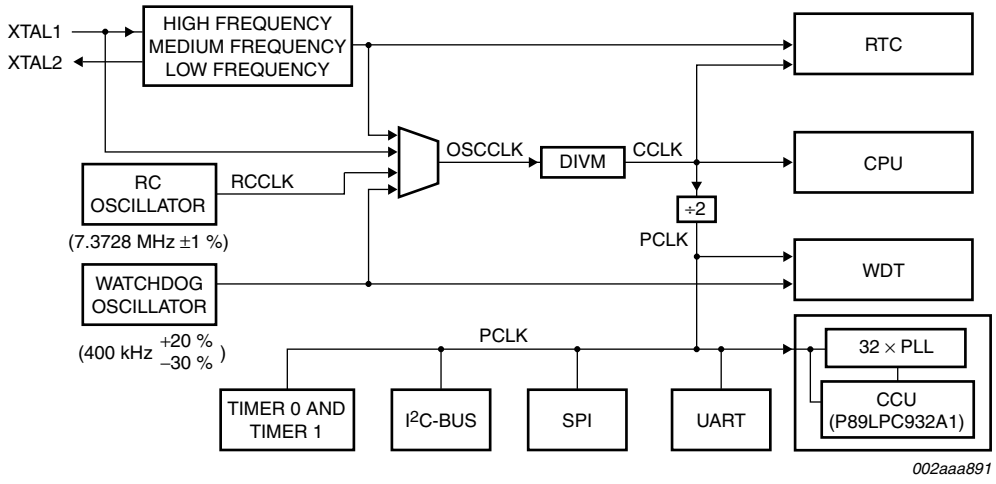


Fig 7. Block diagram of oscillator control.

2.7 Oscillator Clock (OSCCLK) wake-up delay

The P89LPC932A1 has an internal wake-up timer that delays the clock until it stabilizes depending to the clock source used. If the clock source is any of the three crystal selections, the delay is 992 OSCCLK cycles plus 60 μs to 100 μs. If the clock source is either the internal RC oscillator or the Watchdog oscillator, the delay is 224 OSCCLK cycles plus 60 μs to 100 μs.

2.8 CPU Clock (CCLK) modification: DIVM register

The OSCCLK frequency can be divided down, by an integer, up to 510 times by configuring a dividing register, DIVM, to provide CCLK. This produces the CCLK frequency using the following formula:

CCLK frequency = f_{osc} / (2N)

Where: f_{osc} is the frequency of OSCCLK, N is the value of DIVM.

Since N ranges from 0 to 255, the CCLK frequency can be in the range of f_{osc} to f_{osc}/510. (for N = 0, CCLK = f_{osc}).

This feature makes it possible to temporarily run the CPU at a lower rate, reducing power consumption. By dividing the clock, the CPU can retain the ability to respond to events other than those that can cause interrupts (i.e. events that allow exiting the Idle mode) by executing its normal program at a lower rate. This can often result in lower power consumption than in Idle mode. This can allow bypassing the oscillator start-up time in cases where Power-down mode would otherwise be used. The value of DIVM may be changed by the program at any time without interrupting code execution.

2.9 Low power select

The P89LPC932A1 is designed to run at 12 MHz (CCLK) maximum. However, if CCLK is 8 MHz or slower, the CLKLP SFR bit (AUXR1.7) can be set to a logic 1 to lower the power consumption further. On any reset, CLKLP is logic 0 allowing highest performance. This bit can then be set in software if CCLK is running at 8 MHz or slower.

3. Interrupts

The P89LPC932A1 uses a four priority level interrupt structure. This allows great flexibility in controlling the handling of the P89LPC932A1's 15 interrupt sources.

Each interrupt source can be individually enabled or disabled by setting or clearing a bit in the interrupt enable registers IEN0 or IEN1. The IEN0 register also contains a global enable bit, EA, which enables all interrupts.

Each interrupt source can be individually programmed to one of four priority levels by setting or clearing bits in the interrupt priority registers IP0, IP0H, IP1, and IP1H. An interrupt service routine in progress can be interrupted by a higher priority interrupt, but not by another interrupt of the same or lower priority. The highest priority interrupt service cannot be interrupted by any other interrupt source. If two requests of different priority levels are received simultaneously, the request of higher priority level is serviced.

If requests of the same priority level are pending at the start of an instruction cycle, an internal polling sequence determines which request is serviced. This is called the arbitration ranking. Note that the arbitration ranking is only used for pending requests of the same priority level. Table 7 summarizes the interrupt sources, flag bits, vector addresses, enable bits, priority bits, arbitration ranking, and whether each interrupt may wake-up the CPU from a Power-down mode.

3.1 Interrupt priority structure

Table 6: Interrupt priority level

Priority bits		Interrupt priority level
IPxH	IPx	
0	0	Level 0 (lowest priority)
0	1	Level 1
1	0	Level 2
1	1	Level 3

There are four SFRs associated with the four interrupt levels: IP0, IP0H, IP1, IP1H. Every interrupt has two bits in IPx and IPxH (x = 0, 1) and can therefore be assigned to one of four levels, as shown in Table 7.

The P89LPC932A1 has two external interrupt inputs in addition to the Keypad Interrupt function. The two interrupt inputs are identical to those present on the standard 80C51 microcontrollers.

These external interrupts can be programmed to be level-triggered or edge-triggered by clearing or setting bit IT1 or IT0 in Register TCON. If ITn = 0, external interrupt n is triggered by a low level detected at the $\overline{\text{INTn}}$ pin. If ITn = 1, external interrupt n is edge triggered. In this mode if consecutive samples of the $\overline{\text{INTn}}$ pin show a high level in one cycle and a low level in the next cycle, interrupt request flag IEn in TCON is set, causing an interrupt request.

Since the external interrupt pins are sampled once each machine cycle, an input high or low level should be held for at least one machine cycle to ensure proper sampling. If the external interrupt is edge-triggered, the external source has to hold the request pin high for at least one machine cycle, and then hold it low for at least one machine cycle. This is to ensure that the transition is detected and that interrupt request flag IEn is set. IEn is automatically cleared by the CPU when the service routine is called.

If the external interrupt is level-triggered, the external source must hold the request active until the requested interrupt is generated. If the external interrupt is still asserted when the interrupt service routine is completed, another interrupt will be generated. It is not necessary to clear the interrupt flag IEn when the interrupt is level sensitive, it simply tracks the input pin level.

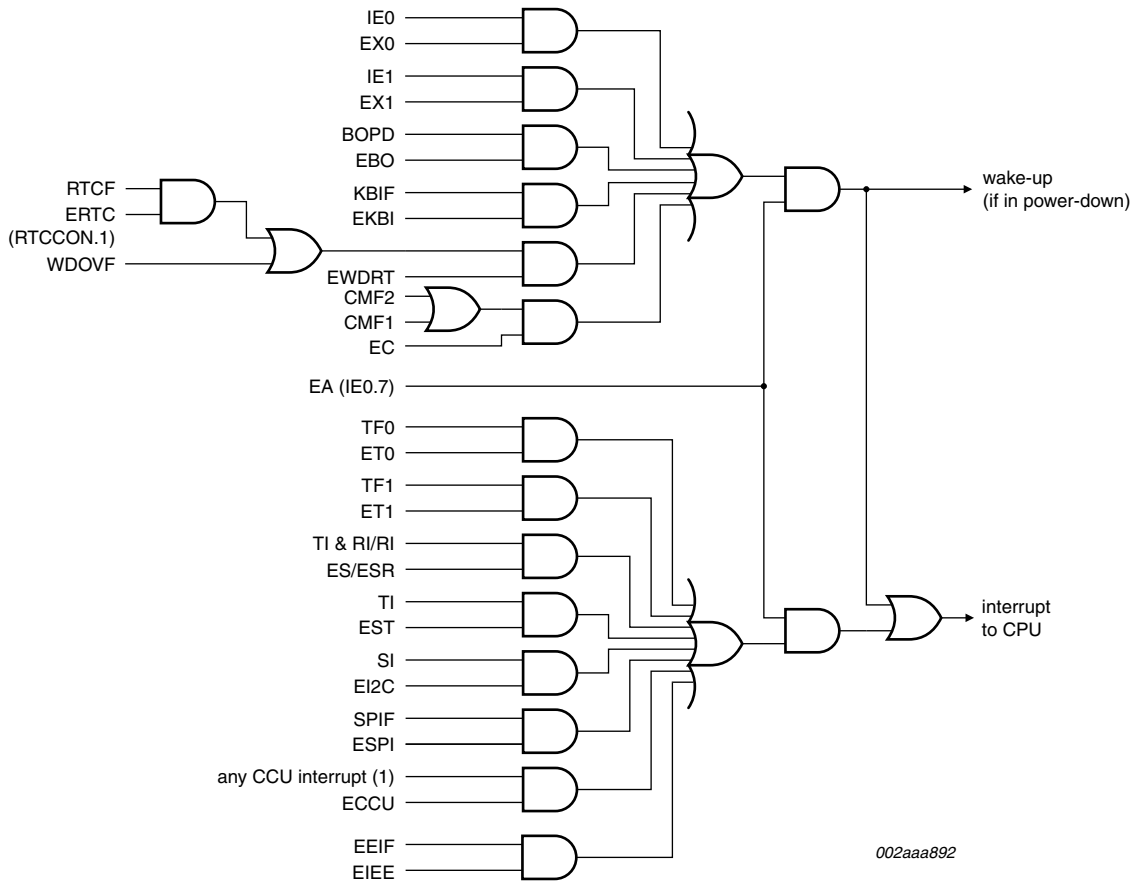
If an external interrupt has been programmed as level-triggered and is enabled when the P89LPC932A1 is put into Power-down mode or Idle mode, the interrupt occurrence will cause the processor to wake-up and resume operation. Refer to Section 5.3 “Power reduction modes” for details.

3.2 External Interrupt pin glitch suppression

Most of the P89LPC932A1 pins have glitch suppression circuits to reject short glitches (please refer to the *P89LPC932A1 data sheet, Dynamic characteristics* for glitch filter specifications). However, pins SDA/$\overline{\text{INT0}}$/P1.3 and SCL/T0/P1.2 do not have the glitch suppression circuits. Therefore, $\overline{\text{INT1}}$ has glitch suppression while $\overline{\text{INT0}}$ does not.

Table 7: Summary of interrupts

Description	Interrupt flag bit(s)	Vector address	Interrupt enable bit(s)	Interrupt priority	Arbitration ranking	Power-down wake-up
External interrupt 0	IE0	0003h	EX0 (IEN0.0)	IP0H.0, IP0.0	1 (highest)	Yes
Timer 0 interrupt	TF0	000Bh	ET0 (IEN0.1)	IP0H.1, IP0.1	4	No
External interrupt 1	IE1	0013h	EX1 (IEN0.2)	IP0H.2, IP0.2	7	Yes
Timer 1 interrupt	TF1	001Bh	ET1 (IEN0.3)	IP0H.3, IP0.3	10	No
Serial port Tx and Rx	TI and RI	0023h	ES/ESR (IEN0.4)	IP0H.4, IP0.4	13	No
Serial port Rx	RI					
Brownout detect	BOF	002Bh	EBO (IEN0.5)	IP0H.5, IP0.5	2	Yes
Watchdog timer/Real-time clock	WDOVF/RTCF	0053h	EWDRT (IEN0.6)	IP0H.6, IP0.6	3	Yes
I^2C interrupt	SI	0033h	EI2C (IEN1.0)	IP0H.0, IP0.0	5	No
KBI interrupt	KBIF	003Bh	EKBI (IEN1.1)	IP0H.0, IP0.0	8	Yes
Comparators 1 and 2 interrupts	CMF1/CMF2	0043h	EC (IEN1.2)	IP0H.0, IP0.0	11	Yes
SPI interrupt	SPIF	004Bh	ESPI (IEN1.3)	IP1H.3, IP1.3	14	No
Capture/Compare Unit		005Bh	ECCU(IEN1.4)	IP1H.4, IP1.4	6	No
Serial port Tx	TI	006Bh	EST (IEN1.6)	IP0H.0, IP0.0	12	No
Data EEPROM	ADCI1, BNDI1	0073h	EAD (IEN1.7)	IP1H.7, IP1.7	15 (lowest)	No



(1) See Section 9 “Capture/Compare Unit (CCU)”.

Fig 8. Interrupt sources, interrupt enables, and power-down wake-up sources.

4. I/O ports

The P89LPC932A1 has four I/O ports: Port 0, Port 1, Port 2, and Port 3. Ports 0, 1, and 2 are 8-bit ports and Port 3 is a 2-bit port. The exact number of I/O pins available depends upon the clock and reset options chosen (see Table 8).

Table 8: Number of I/O pins available

Clock source	Reset option	Number of I/O pins
On-chip oscillator or watchdog oscillator	No external reset (except during power up)	26
	External $\overline{\text{RST}}$ pin supported	25
External clock input	No external reset (except during power up)	25
	External $\overline{\text{RST}}$ pin supported[1]	24
Low/medium/high speed oscillator (external crystal or resonator)	No external reset (except during power up)	24
	External $\overline{\text{RST}}$ pin supported[1]	23

[1] Required for operation above 12 MHz.

4.1 Port configurations

All but three I/O port pins on the P89LPC932A1 may be configured by software to one of four types on a pin-by-pin basis, as shown in Table 9. These are: quasi-bidirectional (standard 80C51 port outputs), push-pull, open drain, and input-only. Two configuration registers for each port select the output type for each port pin.

P1.5 ($\overline{RST}$) can only be an input and cannot be configured.

P1.2 (SCL/T0) and P1.3 (SDA/$\overline{INT0}$) may only be configured to be either input-only or open drain.

Table 9: Port output configuration settings

PxM1.y	PxM2.y	Port output mode
0	0	Quasi-bidirectional
0	1	Push-pull
1	0	Input only (high-impedance)
1	1	Open drain

4.2 Quasi-bidirectional output configuration

Quasi-bidirectional outputs can be used both as an input and output without the need to reconfigure the port. This is possible because when the port outputs a logic high, it is weakly driven, allowing an external device to pull the pin low. When the pin is driven low, it is driven strongly and able to sink a large current. There are three pull-up transistors in the quasi-bidirectional output that serve different purposes.

One of these pull-ups, called the ‘very weak’ pull-up, is turned on whenever the port latch for the pin contains a logic 1. This very weak pull-up sources a very small current that will pull the pin high if it is left floating.

A second pull-up, called the ‘weak’ pull-up, is turned on when the port latch for the pin contains a logic 1 and the pin itself is also at a logic 1 level. This pull-up provides the primary source current for a quasi-bidirectional pin that is outputting a 1. If this pin is pulled low by an external device, the weak pull-up turns off, and only the very weak pull-up remains on. In order to pull the pin low under these conditions, the external device has to sink enough current to overpower the weak pull-up and pull the port pin below its input threshold voltage.

The third pull-up is referred to as the ‘strong’ pull-up. This pull-up is used to speed up low-to-high transitions on a quasi-bidirectional port pin when the port latch changes from a logic 0 to a logic 1. When this occurs, the strong pull-up turns on for two CPU clocks quickly pulling the port pin high.

The quasi-bidirectional port configuration is shown in Figure 9.

Although the P89LPC932A1 is a 3 V device most of the pins are 5 V-tolerant. If 5 V is applied to a pin configured in quasi-bidirectional mode, there will be a current flowing from the pin to V_{DD} causing extra power consumption. Therefore, applying 5 V to pins configured in quasi-bidirectional mode is discouraged.

A quasi-bidirectional port pin has a Schmitt-triggered input that also has a glitch suppression circuit

(Please refer to the *P89LPC932A1 data sheet, Dynamic characteristics* for glitch filter specifications).

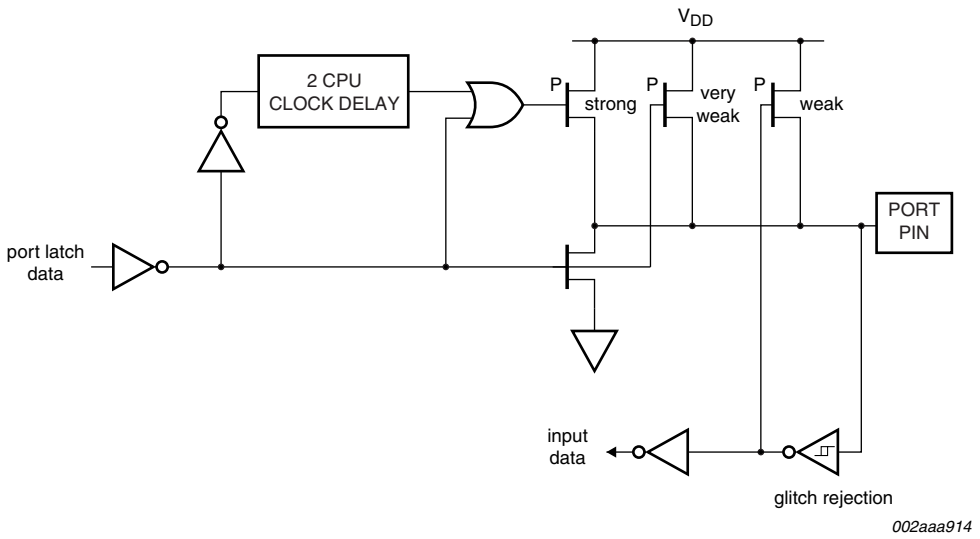


Fig 9. Quasi-bidirectional output.

4.3 Open drain output configuration

The open drain output configuration turns off all pull-ups and only drives the pull-down transistor of the port pin when the port latch contains a logic 0. To be used as a logic output, a port configured in this manner must have an external pull-up, typically a resistor tied to V_{DD}. The pull-down for this mode is the same as for the quasi-bidirectional mode.

The open drain port configuration is shown in Figure 10.

An open drain port pin has a Schmitt-triggered input that also has a glitch suppression circuit.

Please refer to the *P89LPC932A1 data sheet, Dynamic characteristics* for glitch filter specifications.

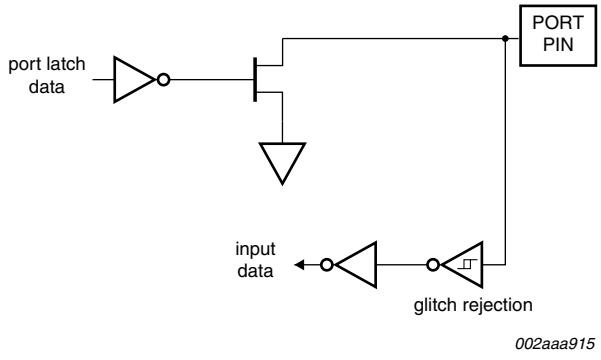


Fig 10. Open drain output.

4.4 Input-only configuration

The input port configuration is shown in Figure 11. It is a Schmitt-triggered input that also has a glitch suppression circuit.

(Please refer to the *P89LPC932A1 data sheet, Dynamic characteristics* for glitch filter specifications).

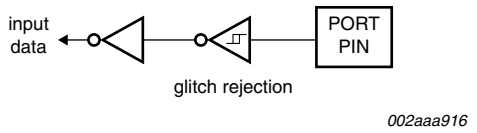


Fig 11. Input only.

4.5 Push-pull output configuration

The push-pull output configuration has the same pull-down structure as both the open drain and the quasi-bidirectional output modes, but provides a continuous strong pull-up when the port latch contains a logic 1. The push-pull mode may be used when more source current is needed from a port output.

The push-pull port configuration is shown in Figure 12.

A push-pull port pin has a Schmitt-triggered input that also has a glitch suppression circuit.

(Please refer to the *P89LPC932A1 data sheet, Dynamic characteristics* for glitch filter specifications).

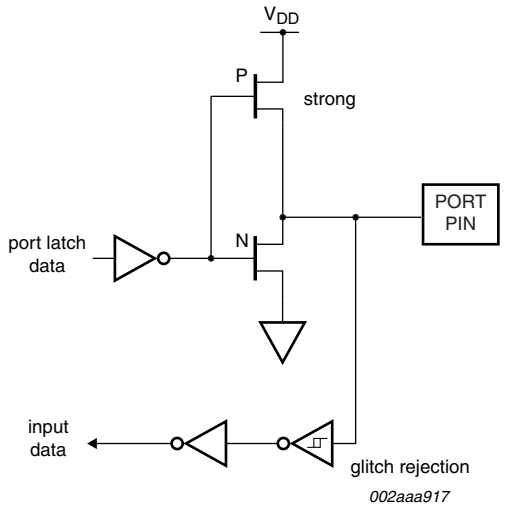


Fig 12. Push-pull output.

4.6 Port 0 and Analog Comparator functions

The P89LPC932A1 incorporates two Analog Comparators. In order to give the best analog performance and minimize power consumption, pins that are being used for analog functions must have both the digital outputs and digital inputs disabled.

Digital outputs are disabled by putting the port pins into the input-only mode as described in the Port Configurations section (see Figure 11).

Digital inputs on Port 0 may be disabled through the use of the PT0AD register. Bits 1 through 5 in this register correspond to pins P0.1 through P0.5 of Port 0, respectively. Setting the corresponding bit in PT0AD disables that pin's digital input. Port bits that have their digital inputs disabled will be read as 0 by any instruction that accesses the port.

On any reset, PT0AD bits 1 through 5 default to logic 0s to enable the digital functions.

4.7 Additional port features

After power-up, all pins are in Input-Only mode. **Please note that this is different from the LPC76x series of devices.**

- After power-up, all I/O pins except P1.5, may be configured by software.
- Pin P1.5 is input only. Pins P1.2 and P1.3 are configurable for either input-only or open drain.

Every output on the P89LPC932A1 has been designed to sink typical LED drive current. However, there is a maximum total output current for all ports which must not be exceeded. Please refer to the *P89LPC932A1 data sheet* for detailed specifications.

All ports pins that can function as an output have slew rate controlled outputs to limit noise generated by quickly switching output signals. The slew rate is factory-set to approximately 10 ns rise and fall times.

Table 10: Port output configuration

Port pin	Configuration SFR bits		Alternate usage	Notes
	PxM1.y	PxM2.y		
P0.0	P0M1.0	P0M2.0	KBIO, CMP2	
P0.1	P0M1.1	P0M2.1	KBI1, CIN2B	Refer to Section 4.6 "Port 0 and Analog Comparator functions" for usage as analog inputs.
P0.2	P0M1.2	P0M2.2	KBI2, CIN2A	
P0.3	P0M1.3	P0M2.3	KBI3, CIN1B	
P0.4	P0M1.4	P0M2.4	KBI4, CIN1A	
P0.5	P0M1.5	P0M2.5	KBI5, CMPREF	
P0.6	P0M1.6	P0M2.6	KBI6, CMP1	
P0.7	P0M1.7	P0M2.7	KBI7, T1	
P1.0	P1M1.0	P1M2.0	TxD	
P1.1	P1M1.1	P1M2.1	RxD	
P1.2	P1M1.2	P1M2.2	T0, SCL	Input-only or open-drain
P1.3	P1M1.3	P1M2.3	$\overline{\text{INTO}}$, SDA	input-only or open-drain
P1.4	P1M1.4	P1M2.4	$\overline{\text{INT1}}$	
P1.5	P1M1.5	P1M2.5	$\overline{\text{RST}}$	

Table 10: Port output configuration *...continued*

Port pin	Configuration SFR bits		Alternate usage	Notes
	PxM1.y	PxM2.y		
P1.6	P1M1.6	P1M2.6		
P1.7	P1M1.7	P1M2.7		
P3.0	P3M1.0	P3M2.0	CLKOUT, XTAL2	
P3.1	P3M1.1	P3M2.1	XTAL1	

5. Power monitoring functions

The P89LPC932A1 incorporates power monitoring functions designed to prevent incorrect operation during initial power-on and power loss or reduction during operation. This is accomplished with two hardware functions: Power-on Detect and Brownout Detect.

5.1 Brownout detection

The Brownout Detect function determines if the power supply voltage drops below a certain level. The default operation for a Brownout Detection is to cause a processor reset. However, it may alternatively be configured to generate an interrupt by setting the BOI (PCON.4) bit and the EBO (IEN0.5) bit.

Enabling and disabling of Brownout Detection is done via the BOPD (PCON.5) bit, bit field PMOD1/PMOD0 (PCON[1:0]) and user configuration bit BOE (UCFG1.5). If BOE is in an unprogrammed state, brownout is disabled regardless of PMOD1/PMOD0 and BOPD. If BOE is in a programmed state, PMOD1/PMOD0 and BOPD will be used to determine whether Brownout Detect will be disabled or enabled. PMOD1/PMOD0 is used to select the power reduction mode. If PMOD1/PMOD0 = ‘11’, the circuitry for the Brownout Detection is disabled for lowest power consumption. BOPD defaults to logic 0, indicating brownout detection is enabled on power-on if BOE is programmed.

If Brownout Detection is enabled, the brownout condition occurs when V_{DD} falls below the Brownout trip voltage, VBO (see *P89LPC932A1 data sheet, Static characteristics*), and is negated when V_{DD} rises above VBO. If the P89LPC932A1 device is to operate with a power supply that can be below 2.7 V, BOE should be left in the unprogrammed state so that the device can operate at 2.4 V, otherwise continuous brownout reset may prevent the device from operating.

If Brownout Detect is enabled (BOE programmed, PMOD1/PMOD0 ≠ ‘11’, BOPD = 0), BOF (RSTSRC.5) will be set when a brownout is detected, regardless of whether a reset or an interrupt is enabled. BOF will stay set until it is cleared in software by writing a logic 0 to the bit. Note that if BOE is unprogrammed, BOF is meaningless. If BOE is programmed, and a initial power-on occurs, BOF will be set in addition to the power-on flag (POF - RSTSRC.4).

For correct activation of Brownout Detect, certain V_{DD} rise and fall times must be observed. Please see the data sheet for specifications.

Table 11: Brownout options[1]

BOE (UCFG1.5)	PMOD1/ PMOD0 (PCON[1:0])	BOPD (PCON.5)	BOI (PCON.4)	EBO (IEN0.5)	EA (IEN0.7)	Description
0 (erased)	XX	X	X	X	X	Brownout disabled. V_{DD} operating range is 2.4 V to 3.6 V.
1(programmed)	11 (total power-down)	X	X	X	X	
	≠ 11 (any mode other than total power-down)	1 (brownout detect power-down)	X	X	X	Brownout disabled. V_{DD} operating range is 2.4 V to 3.6 V. However, BOPD is default to logic 0 upon power-up.
		0 (brownout detect active)	0 (brownout detect generates reset)	X	X	Brownout reset enabled. V_{DD} operating range is 2.7 V to 3.6 V. Upon a brownout reset, BOF (RSTSRC.5) will be set to indicate the reset source. BOF can be cleared by writing a logic 0 to the bit.
			1 (brownout detect generates an interrupt)	1 (enable brownout interrupt)	1 (global interrupt enable)	Brownout interrupt enabled. V_{DD} operating range is 2.7 V to 3.6 V. Upon a brownout interrupt, BOF (RSTSRC.5) will be set. BOF can be cleared by writing a logic 0 to the bit.
				0	X	Both brownout reset and interrupt disabled. V_{DD} operating range is 2.4 V to 3.6 V. However, BOF (RSTSRC.5) will be set when V_{DD} falls to the Brownout Detection trip point. BOF can be cleared by writing a logic 0 to the bit.
				X	0	

[1] Cannot be used with operation above 12 MHz as this requires V_{DD} of 3.0 V or above.

5.2 Power-on detection

The Power-On Detect has a function similar to the Brownout Detect, but is designed to work as power initially comes up, before the power supply voltage reaches a level where the Brownout Detect can function. The POF flag (RSTSRC.4) is set to indicate an initial power-on condition. The POF flag will remain set until cleared by software by writing a logic 0 to the bit. Note that if BOE (UCFG1.5) is programmed, BOF (RSTSRC.5) will be set when POF is set. If BOE is unprogrammed, BOF is meaningless.

5.3 Power reduction modes

The P89LPC932A1 supports three different power reduction modes as determined by SFR bits PCON[1:0] (see Table 12).

Table 12: Power reduction modes

PMOD1 (PCON.1)	PMOD0 (PCON.0)	Description
0	0	Normal mode (default) - no power reduction.
0	1	Idle mode. The Idle mode leaves peripherals running in order to allow them to activate the processor when an interrupt is generated. Any enabled interrupt source or reset may terminate Idle mode.
1	0	Power-down mode: The Power-down mode stops the oscillator in order to minimize power consumption. The P89LPC932A1 exits Power-down mode via any reset, or certain interrupts - external pins $\overline{INT0}$/$\overline{INT1}$, brownout Interrupt, keyboard, Real-time Clock/System Timer), watchdog, and comparator trips. Waking up by reset is only enabled if the corresponding reset is enabled, and waking up by interrupt is only enabled if the corresponding interrupt is enabled and the EA SFR bit (IEN0.7) is set. External interrupts should be programmed to level-triggered mode to be used to exit Power-down mode. In Power-down mode the internal RC oscillator is disabled unless both the RC oscillator has been selected as the system clock AND the RTC is enabled. In Power-down mode, the power supply voltage may be reduced to the RAM keep-alive voltage VRAM. This retains the RAM contents at the point where Power-down mode was entered. SFR contents are not guaranteed after V_{DD} has been lowered to VRAM, therefore it is recommended to wake-up the processor via Reset in this situation. V_{DD} must be raised to within the operating range before the Power-down mode is exited. When the processor wakes up from Power-down mode, it will start the oscillator immediately and begin execution when the oscillator is stable. Oscillator stability is determined by counting 1024 CPU clocks after start-up when one of the crystal oscillator configurations is used, or 256 clocks after start-up for the internal RC or external clock input configurations. Some chip functions continue to operate and draw power during Power-down mode, increasing the total power used during power-down. These include: • Brownout Detect • Watchdog Timer if WDCLK (WDCON.0) is logic 1. • Comparators (Note: Comparators can be powered down separately with PCONA.5 set to logic 1 and comparators disabled); • Real-time Clock/System Timer (and the crystal oscillator circuitry if this block is using it, unless RTCPD, i.e., PCONA.7 is logic 1).
1	1	Total Power-down mode: This is the same as Power-down mode except that the Brownout Detection circuitry and the voltage comparators are also disabled to conserve additional power. Note that a brownout reset or interrupt will not occur. Voltage comparator interrupts and Brownout interrupt cannot be used as a wake-up source. The internal RC oscillator is disabled unless both the RC oscillator has been selected as the system clock AND the RTC is enabled. The following are the wake-up options supported: • Watchdog Timer if WDCLK (WDCON.0) is logic 1. Could generate Interrupt or Reset, either one can wake up the device • External interrupts $\overline{INT0}$/$\overline{INT1}$ (when programmed to level-triggered mode). • Keyboard Interrupt • Real-time Clock/System Timer (and the crystal oscillator circuitry if this block is using it, unless RTCPD, i.e., PCONA.7 is logic 1). Note: Using the internal RC-oscillator to clock the RTC during power-down may result in relatively high power consumption. Lower power consumption can be achieved by using an external low frequency clock when the Real-time Clock is running during power-down.

Table 13: Power Control register (PCON - address 87h) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	SMOD1	SMOD0	BOPD	BOI	GF1	GF0	PMOD1	PMOD0
Reset	0	0	0	0	0	0	0	0

Table 14: Power Control register (PCON - address 87h) bit description

Bit	Symbol	Description
0	PMOD0	Power Reduction Mode (see Section 5.3)
1	PMOD1	
2	GF0	General Purpose Flag 0. May be read or written by user software, but has no effect on operation
3	GF1	General Purpose Flag 1. May be read or written by user software, but has no effect on operation
4	BOI	Brownout Detect Interrupt Enable. When logic 1, Brownout Detection will generate a interrupt. When logic 0, Brownout Detection will cause a reset
5	BOPD	Brownout Detect power-down. When logic 1, Brownout Detect is powered down and therefore disabled. When logic 0, Brownout Detect is enabled. (Note: BOPD must be logic 0 before any programming or erasing commands can be issued. Otherwise these commands will be aborted.)
6	SMOD0	Framing Error Location: • When logic 0, bit 7 of SCON is accessed as SM0 for the UART. • When logic 1, bit 7 of SCON is accessed as the framing error status (FE) for the UART
7	SMOD1	Double Baud Rate bit for the serial port (UART) when Timer 1 is used as the baud rate source. When logic 1, the Timer 1 overflow rate is supplied to the UART. When logic 0, the Timer 1 overflow rate is divided by two before being supplied to the UART. (See Section 10)

Table 15: Power Control register A (PCONA - address B5h) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	RTCPD	DEEPD	VCPD	-	I2PD	SPPD	SPD	CCUPD
Reset	0	0	0	0	0	0	0	0

Table 16: Power Control register A (PCONA - address B5h) bit description

Bit	Symbol	Description
0	CCUPD	Compare/Capture Unit (CCU) power-down: When logic 1, the internal clock to the CCU is disabled. Note that in either Power-down mode or Total Power-down mode, the CCU clock will be disabled regardless of this bit. (Note: This bit is overridden by the CCUDIS bit in FCFG1. If CCUDIS = 1, CCU is powered down.)
1	SPD	Serial Port (UART) power-down: When logic 1, the internal clock to the UART is disabled. Note that in either Power-down mode or Total Power-down mode, the UART clock will be disabled regardless of this bit.
2	SPPD	SPI power-down: When logic 1, the internal clock to the SPI is disabled. Note that in either Power-down mode or Total Power-down mode, the SPI clock will be disabled regardless of this bit.
3	I2PD	I^2C power-down: When logic 1, the internal clock to the I^2C-bus is disabled. Note that in either Power-down mode or Total Power-down mode, the I^2C clock will be disabled regardless of this bit.

Table 16: Power Control register A (PCONA - address B5h) bit description *...continued*

Bit	Symbol	Description
4	-	reserved
5	VCPD	Analog Voltage Comparators power-down: When logic 1, the voltage comparators are powered down. User must disable the voltage comparators prior to setting this bit.
6	DEEPD	Data EEPROM power-down: When logic 1, the Data EEPROM is powered down. Note that in either Power-down mode or Total Power-down mode, the Data EEPROM will be powered down regardless of this bit.
7	RTCPD	Real-time Clock power-down: When logic 1, the internal clock to the Real-time Clock is disabled.

6. Reset

The P1.5/$\overline{\text{RST}}$ pin can function as either an active low reset input or as a digital input, P1.5. The RPE (Reset Pin Enable) bit in UCFG1, when set to 1, enables the external reset input function on P1.5. When cleared, P1.5 may be used as an input pin. **When using an oscillator frequency above 12 MHz, the reset input function of P1.5 must be enabled. An external circuit is required to hold the device in reset at powerup until V_{DD} has reached its specified level. When system power is removed V_{DD} will fall below the minimum specified operating voltage. When using an oscillator frequency above 12 MHz, in some applications, an external brownout detect circuit may be required to hold the device in reset when V_{DD} falls below the minimum specified operating voltage.**

Note: During a power-on sequence, The RPE selection is overridden and this pin will always functions as a reset input. An external circuit connected to this pin should not hold this pin low during a Power-on sequence as this will keep the device in reset. After power-on this input will function either as an external reset input or as a digital input as defined by the RPE bit. Only a power-on reset will temporarily override the selection defined by RPE bit. Other sources of reset will not override the RPE bit.

Note: During a power cycle, V_{DD} must fall below V_{POR} (see *P89LPC932A1 data sheet, Static characteristics*) before power is reapplied, in order to ensure a power-on reset.

Reset can be triggered from the following sources (see Figure 13):

- External reset pin (during power-on or if user configured via UCFG1);
- Power-on Detect;
- Brownout Detect;
- Watchdog Timer;
- Software reset;
- UART break detect reset.

For every reset source, there is a flag in the Reset Register, RSTSRC. The user can read this register to determine the most recent reset source. These flag bits can be cleared in software by writing a logic 0 to the corresponding bit. More than one flag bit may be set:

- During a power-on reset, both POF and BOF are set but the other flag bits are cleared.

- For any other reset, any previously set flag bits that have not been cleared will remain set.

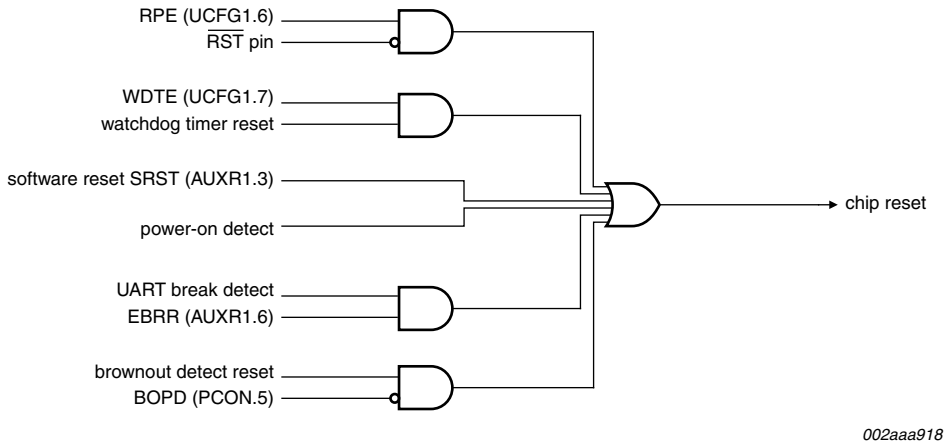


Fig 13. Block diagram of reset.

Table 17: Reset Sources register (RSTSRC - address DFh) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	-	-	BOF	POF	R_BK	R_WD	R_SF	R_EX
Reset[1]	x	x	1	1	0	0	0	0

[1] The value shown is for a power-on reset. Other reset sources will set their corresponding bits.

Table 18: Reset Sources register (RSTSRC - address DFh) bit description

Bit	Symbol	Description
0	R_EX	external reset Flag. When this bit is logic 1, it indicates external pin reset. Cleared by software by writing a logic 0 to the bit or a Power-on reset. If $\overline{RST}$ is still asserted after the Power-on reset is over, R_EX will be set.
1	R_SF	software reset Flag. Cleared by software by writing a logic 0 to the bit or a Power-on reset
2	R_WD	Watchdog Timer reset flag. Cleared by software by writing a logic 0 to the bit or a Power-on reset.(NOTE: UCFG1.7 must be = 1)
3	R_BK	break detect reset. If a break detect occurs and EBRR (AUXR1.6) is set to logic 1, a system reset will occur. This bit is set to indicate that the system reset is caused by a break detect. Cleared by software by writing a logic 0 to the bit or on a Power-on reset.
4	POF	Power-on Detect Flag. When Power-on Detect is activated, the POF flag is set to indicate an initial power-up condition. The POF flag will remain set until cleared by software by writing a logic 0 to the bit. (Note: On a Power-on reset, both BOF and this bit will be set while the other flag bits are cleared.)
5	BOF	Brownout Detect Flag. When Brownout Detect is activated, this bit is set. It will remain set until cleared by software by writing a logic 0 to the bit. (Note: On a Power-on reset, both POF and this bit will be set while the other flag bits are cleared.)
6:7	-	reserved

6.1 Reset vector

Following reset, the P89LPC932A1 will fetch instructions from either address 0000h or the Boot address. The Boot address is formed by using the Boot Vector as the high byte of the address and the low byte of the address = 00h. The Boot address will be used if a UART break reset occurs or the non-volatile Boot Status bit (BOOTSTAT.0) = 1, or the device has been forced into ISP mode. Otherwise, instructions will be fetched from address 0000H.

7. Timers 0 and 1

The P89LPC932A1 has two general-purpose counter/timers which are upward compatible with the 80C51 Timer 0 and Timer 1. Both can be configured to operate either as timers or event counters (see Table 20). An option to automatically toggle the Tx pin upon timer overflow has been added.

In the 'Timer' function, the timer is incremented every PCLK.

In the 'Counter' function, the register is incremented in response to a 1-to-0 transition on its corresponding external input pin (T0 or T1). The external input is sampled once during every machine cycle. When the pin is high during one cycle and low in the next cycle, the count is incremented. The new count value appears in the register during the cycle following the one in which the transition was detected. Since it takes two machine cycles (four CPU clocks) to recognize a 1-to-0 transition, the maximum count rate is $\frac{1}{4}$ of the CPU clock frequency. There are no restrictions on the duty cycle of the external input signal, but to ensure that a given level is sampled at least once before it changes, it should be held for at least one full machine cycle.

The 'Timer' or 'Counter' function is selected by control bits TnC/$\overline{T}$ (x = 0 and 1 for Timers 0 and 1 respectively) in the Special Function Register TMOD. Timer 0 and Timer 1 have five operating modes (modes 0, 1, 2, 3 and 6), which are selected by bit-pairs (TnM1, TnM0) in TMOD and TnM2 in TAMOD. Modes 0, 1, 2 and 6 are the same for both Timers/Counters. Mode 3 is different. The operating modes are described later in this section.

Table 19: Timer/Counter Mode register (TMOD - address 89h) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	T1GATE	T1C/$\overline{T}$	T1M1	T1M0	T0GATE	T0C/$\overline{T}$	T0M1	T0M0
Reset	0	0	0	0	0	0	0	0

Table 20: Timer/Counter Mode register (TMOD - address 89h) bit description

Bit	Symbol	Description
0	T0M0	Mode Select for Timer 0. These bits are used with the T0M2 bit in the TAMOD register to determine the Timer 0 mode (see Table 22).
1	T0M1	
2	T0C/$\overline{T}$	Timer or Counter selector for Timer 0. Cleared for Timer operation (input from CCLK). Set for Counter operation (input from T0 input pin).
3	T0GATE	Gating control for Timer 0. When set, Timer/Counter is enabled only while the $\overline{INT0}$ pin is high and the TR0 control pin is set. When cleared, Timer 0 is enabled when the TR0 control bit is set.

Table 20: Timer/Counter Mode register (TMOD - address 89h) bit description *...continued*

Bit	Symbol	Description
4	T1M0	Mode Select for Timer 1. These bits are used with the T1M2 bit in the TAMOD register to determine the Timer 1 mode (see Table 22).
5	T1M1	
6	T1C/$\overline{T}$	Timer or Counter Selector for Timer 1. Cleared for Timer operation (input from CCLK). Set for Counter operation (input from T1 input pin).
7	T1GATE	Gating control for Timer 1. When set, Timer/Counter is enabled only while the $\overline{INT1}$ pin is high and the TR1 control pin is set. When cleared, Timer 1 is enabled when the TR1 control bit is set.

Table 21: Timer/Counter Auxiliary Mode register (TAMOD - address 8Fh) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	--	-	-	T1M2	-	-	-	T0M2
Reset	x	x	x	0	x	x	x	0

Table 22: Timer/Counter Auxiliary Mode register (TAMOD - address 8Fh) bit description

Bit	Symbol	Description
0	T0M2	Mode Select for Timer 0. These bits are used with the T0M2 bit in the TAMOD register to determine the Timer 0 mode (see Table 22).
1:3	-	reserved
4	T1M2	Mode Select for Timer 1. These bits are used with the T1M2 bit in the TAMOD register to determine the Timer 1 mode (see Table 22).
		The following timer modes are selected by timer mode bits TnM[2:0]: **000 —** 8048 Timer 'TLn' serves as 5-bit prescaler. (Mode 0) **001 —** 16-bit Timer/Counter 'THn' and 'TLn' are cascaded; there is no prescaler.(Mode 1) **010 —** 8-bit auto-reload Timer/Counter. THn holds a value which is loaded into TLn when it overflows. (Mode 2) **011 —** Timer 0 is a dual 8-bit Timer/Counter in this mode. TL0 is an 8-bit Timer/Counter controlled by the standard Timer 0 control bits. TH0 is an 8-bit timer only, controlled by the Timer 1 control bits (see text). Timer 1 in this mode is stopped. (Mode 3) **100 —** Reserved. User must not configure to this mode. **101 —** Reserved. User must not configure to this mode. **110 —** PWM mode (see Section 7.5). **111 —** Reserved. User must not configure to this mode.
5:7	-	reserved

7.1 Mode 0

Putting either Timer into Mode 0 makes it look like an 8048 Timer, which is an 8-bit Counter with a divide-by-32 prescaler. Figure 14 shows Mode 0 operation.

In this mode, the Timer register is configured as a 13-bit register. As the count rolls over from all 1s to all 0s, it sets the Timer interrupt flag TFn. The count input is enabled to the Timer when TRn = 1 and either TnGATE = 0 or $\overline{INTn}$ = 1. (Setting TnGATE = 1 allows the Timer to be controlled by external input $\overline{INTn}$, to facilitate pulse width measurements). TRn is a control bit in the Special Function Register TCON (Table 24). The TnGATE bit is in the TMOD register.

The 13-bit register consists of all 8 bits of THn and the lower 5 bits of TLn. The upper 3 bits of TLn are indeterminate and should be ignored. Setting the run flag (TRn) does not clear the registers.

Mode 0 operation is the same for Timer 0 and Timer 1. See Figure 14. There are two different GATE bits, one for Timer 1 (TMOD.7) and one for Timer 0 (TMOD.3).

7.2 Mode 1

Mode 1 is the same as Mode 0, except that all 16 bits of the timer register (THn and TLn) are used. See Figure 15.

7.3 Mode 2

Mode 2 configures the Timer register as an 8-bit Counter (TLn) with automatic reload, as shown in Figure 16. Overflow from TLn not only sets TFn, but also reloads TLn with the contents of THn, which must be preset by software. The reload leaves THn unchanged. Mode 2 operation is the same for Timer 0 and Timer 1.

7.4 Mode 3

When Timer 1 is in Mode 3 it is stopped. The effect is the same as setting TR1 = 0.

Timer 0 in Mode 3 establishes TL0 and TH0 as two separate 8-bit counters. The logic for Mode 3 on Timer 0 is shown in Figure 17. TL0 uses the Timer 0 control bits: T0C/$\overline{T}$, T0GATE, TR0, $\overline{INT0}$, and TF0. TH0 is locked into a timer function (counting machine cycles) and takes over the use of TR1 and TF1 from Timer 1. Thus, TH0 now controls the 'Timer 1' interrupt.

Mode 3 is provided for applications that require an extra 8-bit timer. With Timer 0 in Mode 3, an P89LPC932A1 device can look like it has three Timer/Counters.

Note: When Timer 0 is in Mode 3, Timer 1 can be turned on and off by switching it into and out of its own Mode 3. It can still be used by the serial port as a baud rate generator, or in any application not requiring an interrupt.

7.5 Mode 6

In this mode, the corresponding timer can be changed to a PWM with a full period of 256 timer clocks (see Figure 18). Its structure is similar to mode 2, except that:

- TFn (n = 0 and 1 for Timers 0 and 1 respectively) is set and cleared in hardware;
- The low period of the TFn is in THn, and should be between 1 and 254, and;
- The high period of the TFn is always 256–THn.
- Loading THn with 00h will force the Tx pin high, loading THn with FFh will force the Tx pin low.

Note that interrupt can still be enabled on the low to high transition of TFn, and that TFn can still be cleared in software like in any other modes.

Table 23: Timer/Counter Control register (TCON) - address 88h) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
Reset	0	0	0	0	0	0	0	0

Table 24: Timer/Counter Control register (TCON - address 88h) bit description

Bit	Symbol	Description
0	IT0	Interrupt 0 Type control bit. Set/cleared by software to specify falling edge/low level triggered external interrupts.
1	IE0	Interrupt 0 Edge flag. Set by hardware when external interrupt 0 edge is detected. Cleared by hardware when the interrupt is processed, or by software.
2	IT1	Interrupt 1 Type control bit. Set/cleared by software to specify falling edge/low level triggered external interrupts.
3	IE1	Interrupt 1 Edge flag. Set by hardware when external interrupt 1 edge is detected. Cleared by hardware when the interrupt is processed, or by software.
4	TR0	Timer 0 Run control bit. Set/cleared by software to turn Timer/Counter 0 on/off.
5	TF0	Timer 0 overflow flag. Set by hardware on Timer/Counter overflow. Cleared by hardware when the processor vectors to the interrupt routine, or by software. (except in mode 6, where it is cleared in hardware)
6	TR1	Timer 1 Run control bit. Set/cleared by software to turn Timer/Counter 1 on/off
7	TF1	Timer 1 overflow flag. Set by hardware on Timer/Counter overflow. Cleared by hardware when the interrupt is processed, or by software (except in mode 6, see above, when it is cleared in hardware).

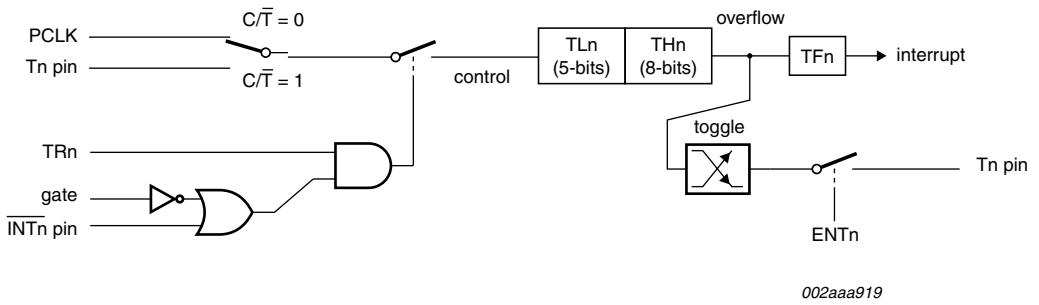


Fig 14. Timer/counter 0 or 1 in Mode 0 (13-bit counter).

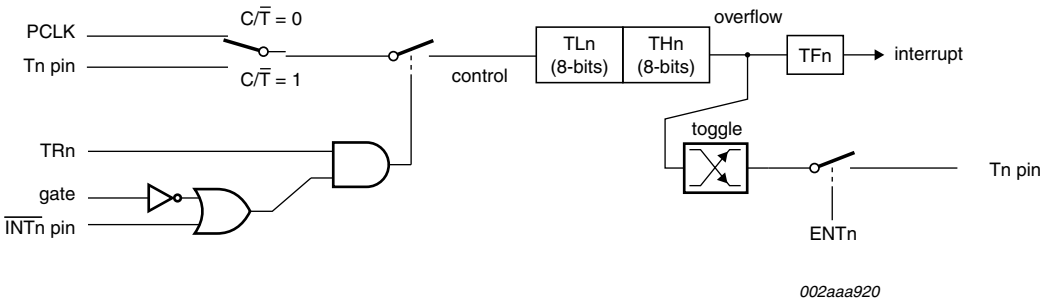


Fig 15. Timer/counter 0 or 1 in mode 1 (16-bit counter).

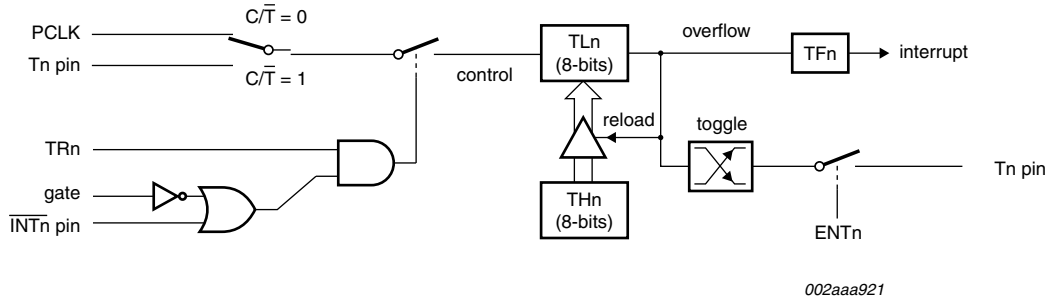


Fig 16. Timer/counter 0 or 1 in Mode 2 (8-bit auto-reload).

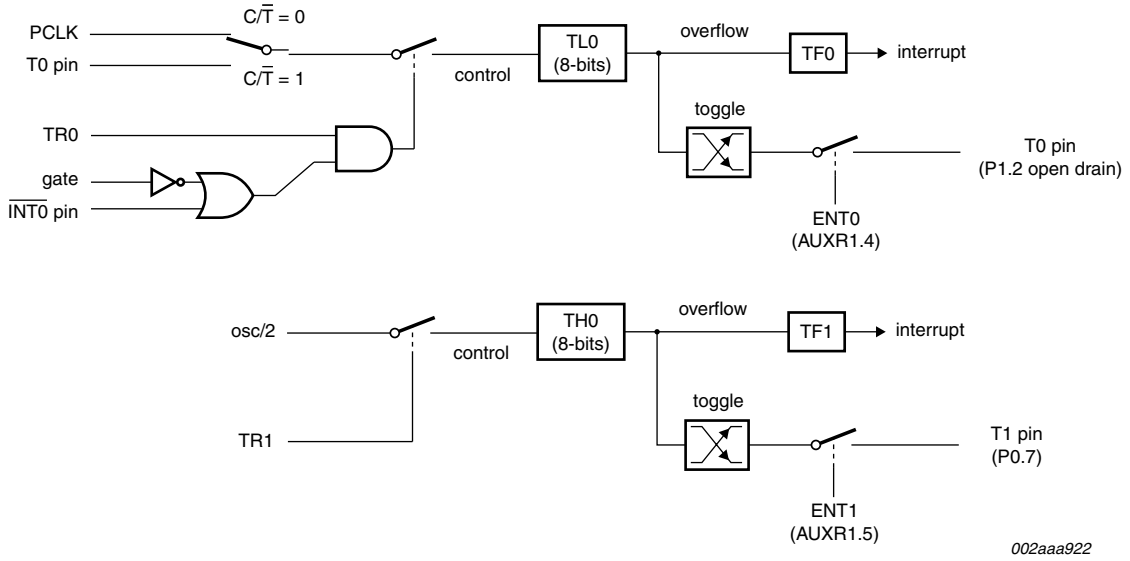


Fig 17. Timer/counter 0 Mode 3 (two 8-bit counters).

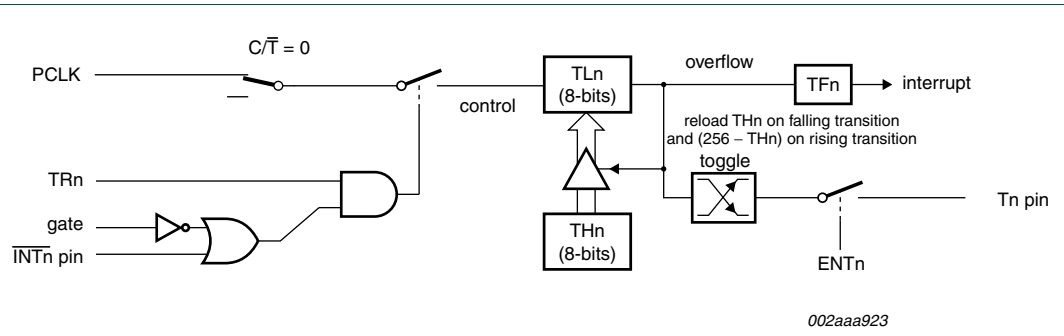


Fig 18. Timer/counter 0 or 1 in mode 6 (PWM auto-reload).

7.6 Timer overflow toggle output

Timers 0 and 1 can be configured to automatically toggle a port output whenever a timer overflow occurs. The same device pins that are used for the T0 and T1 count inputs and PWM outputs are also used for the timer toggle outputs. This function is enabled by

control bits ENT0 and ENT1 in the AUXR1 register, and apply to Timer 0 and Timer 1 respectively. The port outputs will be a logic 1 prior to the first timer overflow when this mode is turned on. In order for this mode to function, the C/T bit must be cleared selecting PCLK as the clock source for the timer.

8. Real-time clock system timer

The P89LPC932A1 has a simple Real-time Clock/System Timer that allows a user to continue running an accurate timer while the rest of the device is powered down. The Real-time Clock can be an interrupt or a wake-up source (see Figure 19).

The Real-time Clock is a 23-bit down counter. The clock source for this counter can be either the CPU clock (CCLK) or the XTAL1-2 oscillator, provided that the XTAL1-2 oscillator is not being used as the CPU clock. If the XTAL1-2 oscillator is used as the CPU clock, then the RTC will use CCLK as its clock source regardless of the state of the RTCS1:0 in the RTCCON register. There are three SFRs used for the RTC:

RTCCON — Real-time Clock control.

RTCH — Real-time Clock counter reload high (bits 22 to 15).

RTCL — Real-time Clock counter reload low (bits 14 to 7).

The Real-time clock system timer can be enabled by setting the RTCEN (RTCCON.0) bit. The Real-time Clock is a 23-bit down counter (initialized to all 0's when RTCEN = 0) that is comprised of a 7-bit prescaler and a 16-bit loadable down counter. When RTCEN is written with logic 1, the counter is first loaded with (RTCH, RTCL, '1111111') and will count down. When it reaches all 0's, the counter will be reloaded again with (RTCH, RTCL, '1111111') and a flag - RTCF (RTCCON.7) - will be set.

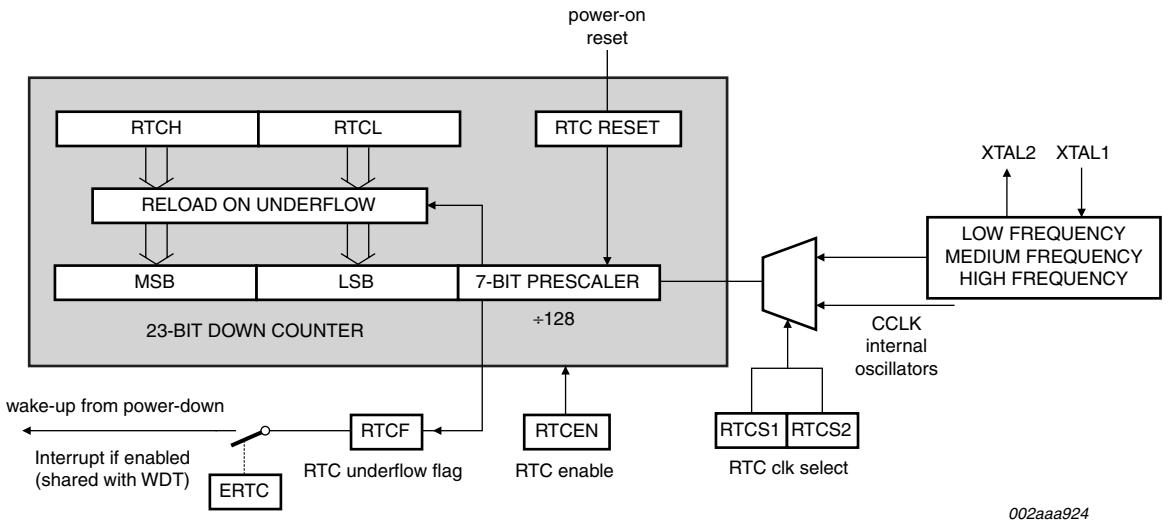


Fig 19. Real-time clock/system timer block diagram.

8.1 Real-time clock source

RTCS1/RTCS0 (RTCCON[6:5]) are used to select the clock source for the RTC if either the Internal RC oscillator or the internal WD oscillator is used as the CPU clock. If the internal crystal oscillator or the external clock input on XTAL1 is used as the CPU clock, then the RTC will use CCLK as its clock source.

8.2 Changing RTCS1/RTCS0

RTCS1/RTCS0 cannot be changed if the RTC is currently enabled (RTCCON.0 = 1). Setting RTCEN and updating RTCS1/RTCS0 may be done in a single write to RTCCON. However, if RTCEN = 1, this bit must first be cleared before updating RTCS1/RTCS0.

8.3 Real-time clock interrupt/wake-up

If ERTC (RTCCON.1), EWDRT (IEN1.0.6) and EA (IEN0.7) are set to logic 1, RTCF can be used as an interrupt source. This interrupt vector is shared with the watchdog timer. It can also be a source to wake-up the device.

8.4 Reset sources affecting the Real-time clock

Only power-on reset will reset the Real-time Clock and its associated SFRs to their default state.

Table 25: Real-time Clock/System Timer clock sources

FOSC2:0	RCCLK	RTCS1:0	RTC clock source	CPU clock source
000	0	00	High frequency crystal	High frequency crystal /DIVM
		01		
		10		
		11	High frequency crystal /DIVM	
	1	00	High frequency crystal	Internal RC oscillator
		01		
		10		
		11	Internal RC oscillator	
001	0	00	Medium frequency crystal	Medium frequency crystal /DIVM
		01		
		10		
		11	Medium frequency crystal /DIVM	
	1	00	Medium frequency crystal	Internal RC oscillator
		01		
		10		
		11	Internal RC oscillator	

Table 25: Real-time Clock/System Timer clock sources *...continued*

FOSC2:0	RCCLK	RTCS1:0	RTC clock source	CPU clock source
010	0	00	Low frequency crystal	Low frequency crystal /DIVM
		01		
		10		
		11	Low frequency crystal /DIV	
	1	00	Low frequency crystal	Internal RC oscillator
		01		
		10		
		11	Internal RC oscillator	
011	0	00	High frequency crystal	Internal RC oscillator /DIVM
		01	Medium frequency crystal	
		10	Low frequency crystal	
		11	Internal RC oscillator /DIVM	
	1	00	High frequency crystal	Internal RC oscillator
		01	Medium frequency crystal	
		10	Low frequency crystal	
		11	Internal RC oscillator	
100	0	00	High frequency crystal	Watchdog oscillator /DIVM
		01	Medium frequency crystal	
		10	Low frequency crystal	
		11	Watchdog oscillator /DIVM	
	1	00	High frequency crystal	Internal RC oscillator
		01	Medium frequency crystal	
		10	Low frequency crystal	
		11	Internal RC oscillator	
101	x	xx	undefined	undefined
110	x	xx	undefined	undefined
111	0	00	External clock input	External clock input /DIVM
		01		
		10		
		11	External clock input /DIVM	
	1	00	External clock input	Internal RC oscillator
		01		
		10		
		11	Internal RC oscillator	

Table 26: Real-time Clock Control register (RTCCON - address D1h) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	RTCF	RTCS1	RTCS0	-	-	-	ERTC	RTCEN
Reset	0	1	1	x	x	x	0	0

Table 27: Real-time Clock Control register (RTCCON - address D1h) bit description

Bit	Symbol	Description
0	RTCEN	Real-time Clock enable. The Real-time Clock will be enabled if this bit is logic 1. Note that this bit will not power-down the Real-time Clock. The RTCPD bit (PCONA.7) if set, will power-down and disable this block regardless of RTCEN.
1	ERTC	Real-time Clock interrupt enable. The Real-time Clock shares the same interrupt as the watchdog timer. Note that if the user configuration bit WDTE (UCFG1.7) is logic 0, the watchdog timer can be enabled to generate an interrupt. Users can read the RTCF (RTCCON.7) bit to determine whether the Real-time Clock caused the interrupt.
2:4	-	reserved
5	RTCS0	Real-time Clock source select (see Table 25).
6	RTCS1	
7	RTCF	Real-time Clock Flag. This bit is set to logic 1 when the 23-bit Real-time Clock reaches a count of logic 0. It can be cleared in software.

9. Capture/Compare Unit (CCU)

This unit features:

- A 16-bit timer with 16-bit reload on overflow
- Selectable clock (CCUCLK), with a prescaler to divide the clock source by any integer between 1 and 1024.
- Four Compare / PWM outputs with selectable polarity
- Symmetrical / Asymmetrical PWM selection
- Seven interrupts with common interrupt vector (one Overflow, 2xCapture, 4xCompare), safe 16-bit read/write via shadow registers.
- Two Capture inputs with event counter and digital noise rejection filter

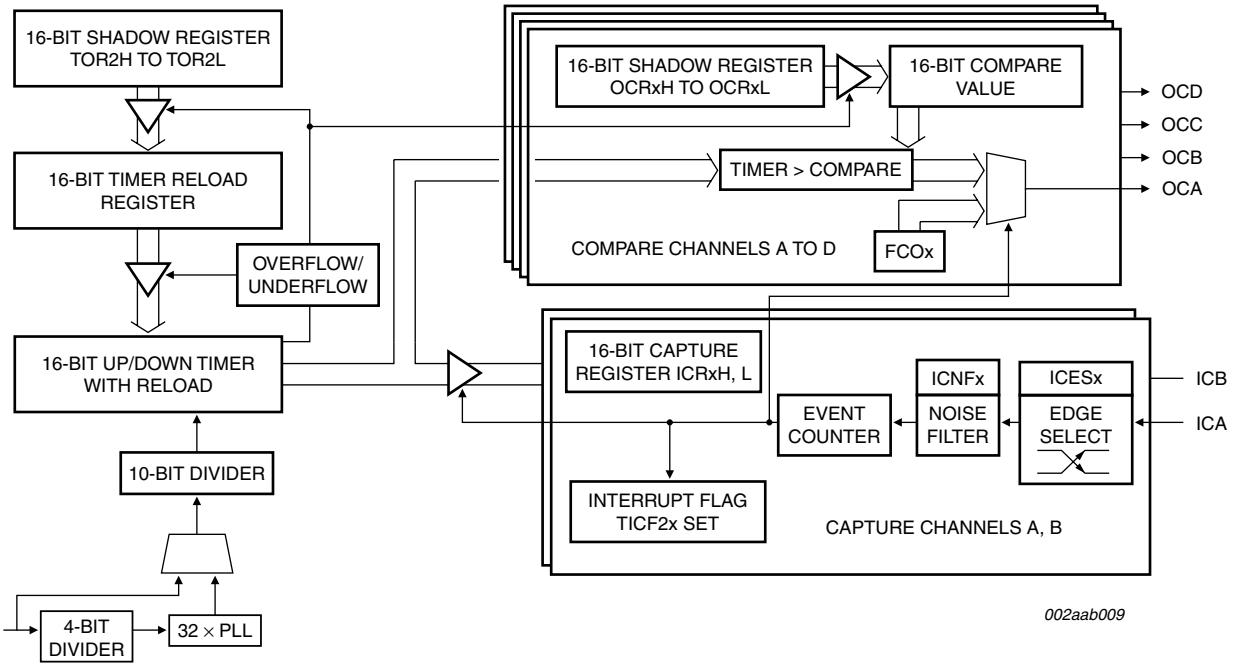


Fig 20. Capture Compare Unit block diagram.

9.1 CCU Clock (CCUCLK)

The CCU runs on the CCUCLK, which can be either PCLK in basic timer mode or the output of a PLL (see Figure 20). The PLL is designed to use a clock source between 0.5 MHz to 1 MHz that is multiplied by 32 to produce a CCUCLK between 16 MHz and 32 MHz in PWM mode (asymmetrical or symmetrical). The PLL contains a 4-bit divider (PLLDV3:0 bits in the TCR21 register) to help divide PCLK into a frequency between 0.5 MHz and 1 MHz

9.2 CCU Clock prescaling

This CCUCLK can further be divided down by a prescaler. The prescaler is implemented as a 10-bit free-running counter with programmable reload at overflow. Writing a value to the prescaler will cause the prescaler to restart.

9.3 Basic timer operation

The Timer is a free-running up/down counter counting at the pace determined by the prescaler. The timer is started by setting the CCU Mode Select bits TMOD21 and TMOD20 in the CCU Control Register 0 (TCR20) as shown in the table in the TCR20 register description (Table 32).

The CCU direction control bit, TDIR2, determines the direction of the count. TDIR2 = 0: Count up, TDIR2 = 1: Count down. If the timer counting direction is changed while the counter is running, the count sequence will be reversed in the CCUCLK cycle following the write of TDIR2. The timer can be written or read at any time and newly-written values will take effect when the prescaler overflows. The timer is accessible through two SFRs, TL2(low byte) and TH2(high byte). A third 16-bit SFR, TOR2H:TOR2L, determines the overflow reload value. TL2, TH2 and TOR2H, TOR2L will be 0 after a reset

Up-counting: When the timer contents are FFFFH, the next CCUCLK cycle will set the counter value to the contents of TOR2H:TOR2L.

Down-counting: When the timer contents are 0000H, the next CCUCLK cycle will set the counter value to the contents of TOR2H:TOR2L. During the CCUCLK cycle when the reload is performed, the CCU Timer Overflow Interrupt Flag (TOIF2) in the CCU Interrupt Flag Register (TIFR2) will be set, and, if the EA bit in the IEN0 register and ECCU bit in the IEN1 register (IEN1.4) are set, program execution will vector to the overflow interrupt. The user has to clear the interrupt flag in software by writing a logic 0 to it.

When writing to the reload registers, TOR2H and TOR2L, the values written are stored in two 8-bit shadow registers. In order to latch the contents of the shadow registers into TOR2H and TOR2L, the user must write a logic 1 to the CCU Timer Compare/Overflow Update bit TCOU2, in CCU Timer Control Register 1 (TCR21). The function of this bit depends on whether the timer is running in PWM mode or in basic timer mode. In basic timer mode, writing a one to TCOU2 will cause the values to be latched immediately and the value of TCOU2 will always read as zero. In PWM mode, writing a one to TCOU2 will cause the contents of the shadow registers to be updated on the next CCU Timer overflow. As long as the latch is pending, TCOU2 will read as one and will return to zero when the latching takes place. TCOU2 also controls the latching of the Output Compare registers OCR2A, OCR2B and OCR2C.

When writing to timer high byte, TH2, the value written is stored in a shadow register. When TL2 is written, the contents of TH2's shadow register is transferred to TH2 at the same time that TL2 gets updated. Thus, TH2 should be written prior to writing to TL2. If a write to TL2 is followed by another write to TL2, without TH2 being written in between, the value of TH2 will be transferred directly to the high byte of the timer.

If the 16-bit CCU Timer is to be used as an 8-bit timer, the user can write FFh (for upcounting) or 00h (for downcounting) to TH2. When TL2 is written, FFh:TH2 (for upcounting) and 00h (for downcounting) will be loaded to CCU Timer. The user will not need to rewrite TH2 again for an 8-bit timer operation unless there is a change in count direction

When reading the timer, TL2 must be read first. When TL2 is read, the contents of the timer high byte are transferred to a shadow register in the same PCLK cycle as the read is performed. When TH2 is read, the contents of the shadow register are read instead. If a read from TL2 is followed by another read from TL2 without TH2 being read in between, the high byte of the timer will be transferred directly to TH2.

Table 28: CCU prescaler control register, high byte (TPCR2H - address CBh) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	-	-	-	-	-	-	TPCR2H.1	TPCR2H.0
Reset	x	x	x	x	x	x	0	0

Table 29: CCU prescaler control register, high byte (TPCR2H - address CBh) bit description

Bit	Symbol	Description
0	TPCR2H.0	Prescaler bit 8
1	TPCR2H.1	Prescaler bit 9

Table 30: CCU prescaler control register, low byte (TPCR2L - address CAh) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	TPCR2L.7	TPCR2L.6	TPCR2L.5	TPCR2L.4	TPCR2L.3	TPCR2L.2	TPCR2L.1	TPCR2L.0
Reset	0	0	0	0	0	0	0	0

Table 31: CCU prescaler control register, low byte (TPCR2L - address CAh) bit description

Bit	Symbol	Description
0	TPCR2L.0	Prescaler bit 0
1	TPCR2L.1	Prescaler bit 1
2	TPCR2L.2	Prescaler bit 2
3	TPCR2L.3	Prescaler bit 3
4	TPCR2L.4	Prescaler bit 4
5	TPCR2L.5	Prescaler bit 5
6	TPCR2L.6	Prescaler bit 6
7	TPCR2L.7	Prescaler bit 7

Table 32: CCU control register 0 (TCR20 - address C8h) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	PLLEN	HLTRN	HLTEN	ALTCD	ALTAB	TDIR2	TMOD21	TMOD20
Reset	0	0	0	0	0	0	0	0

Table 33: CCU control register 0 (TCR20 - address C8h) bit description

Bit	Symbol	Description
1:2	TMOD20/21	CCU Timer mode (TMOD21, TMOD20): **00** — Timer is stopped **01** — Basic timer function **10** — Asymmetrical PWM (uses PLL as clock source) **11** — Symmetrical PWM (uses PLL as clock source)
2	TDIR2	Count direction of the CCU Timer. When logic 0, count up, When logic 1, count down.
3	ALTAB	PWM channel A/B alternately output enable. When this bit is set, the output of PWM channel A and B are alternately gated on every counter cycle.
4	ALTCD	PWM channel C/D alternately output enable. When this bit is set, the output of PWM channel C and D are alternately gated on every counter cycle.
5	HLTEN	PWM Halt Enable. When logic 1, a capture event as enabled for Input Capture A pin will immediately stop all activity on the PWM pins and set them to a predetermined state.
6	HLTRN	PWM Halt. When set indicates a halt took place. In order to re-activate the PWM, the user must clear the HLTRN bit.
7	PLLEN	Phase Locked Loop Enable. When set to logic 1, starts PLL operation. After the PLL is in lock this bit it will read back a one.

9.4 Output compare

The four output compare channels A, B, C and D are controlled through four 16-bit SFRs, OCRAH:OCRAL, OCRBH:OCRBL, OCRCH:OCRCL, OCRDH: OCRDL. Each output compare channel needs to be enabled in order to operate. The channel is enabled by selecting a Compare Output Action by setting the OCMx1:0 bits in the Capture Compare x Control Register – CCCRx (x = A, B, C, D). When a compare channel is enabled, the user

will have to set the associated I/O pin to the desired output mode to connect the pin. (Note: The SFR bits for port pins P2.6, P1.6, P1.7, P2.1 must be set to logic 1 in order for the compare channel outputs to be visible at the port pins.) When the contents of TH2:TL2 match that of OCRxH:OCRxL, the Timer Output Compare Interrupt Flag - TOCFx is set in TIFR2. This happens in the CCUCLK cycle after the compare takes place. If EA and the Timer Output Compare Interrupt Enable bit – TOCIE2x (in TICR2 register), as well as ECCU bit in IEN1 are all set, the program counter will be vectored to the corresponding interrupt. The user must manually clear the bit by writing a logic 0 to it.

Two bits in OCCRx, the Output Compare x Mode bits OCMx1 and OCMx0 select what action is taken when a compare match occurs. Enabled compare actions take place even if the interrupt is disabled.

In order for a Compare Output Action to occur, the compare values must be within the counting range of the CCU timer.

When the compare channel is enabled, the I/O pin (which must be configured as an output) will be connected to an internal latch controlled by the compare logic. The value of this latch is zero from reset and can be changed by invoking a forced compare. A forced compare is generated by writing a logic 1 to the Force Compare x Output bit – FCOx bit in OCCRx. Writing a one to this bit generates a transition on the corresponding I/O pin as set up by OCMx1/OCMx0 without causing an interrupt. In basic timer operating mode the FCOx bits always read zero. (Note: This bit has a different function in PWM mode.) When an output compare pin is enabled and connected to the compare latch, the state of the compare pin remains unchanged until a compare event or forced compare occurs.

Table 34: Capture compare control register (CCRx - address Exh) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	ICECx2	ICECx1	ICECx0	ICESx	ICNFx	FCOx	OCMx1	OCMx0
Reset	0	0	0	0	0	0	0	0

Table 35: Capture compare control register (CCRx - address Exh) bit description

Bit	Symbol	Description
0	OCMx0	Output Compare x Mode. See Table 37 “Output compare pin behavior”
1	OCMx1	
2	FCOx	Force Compare X Output Bit. When set, invoke a force compare.
3	ICNFx	Input Capture x Noise Filter Enable Bit. When logic 1, the capture logic needs to see four consecutive samples of the same value in order to recognize an edge as a capture event. The inputs are sampled every two CCLK periods regardless of the speed of the timer.
4	ICESx	Input Capture x Edge Select Bit. When logic 0: Negative edge triggers a capture, When logic 1: Positive edge triggers a capture.
5	ICECx0	Capture Delay Setting Bit 0. See Table 36 for details.
6	ICECx1	Capture Delay Setting Bit 1. See Table 36 for details.
7	ICECx2	Capture Delay Setting Bit 2. See Table 36 for details.

When the user writes to change the output compare value, the values written to OCRH2x and OCRL2x are transferred to two 8-bit shadow registers. In order to latch the contents of the shadow registers into the capture compare register, the user must write a logic 1 to the CCU Timer Compare/Overflow Update bit TCOU2, in the CCU Control Register 1 - TCR21. The function of this bit depends on whether the timer is running in PWM mode or

in basic timer mode. In basic timer mode, writing a one to TCOU2 will cause the values to be latched immediately and the value of TCOU2 will always read as zero. In PWM mode, writing a one to TCOU2 will cause the contents of the shadow registers to be updated on the next CCU Timer overflow. As long as the latch is pending, TCOU2 will read as one and will return to zero when the latch takes place. TCOU2 also controls the latching of all the Output Compare registers as well as the Timer Overflow Reload registers - TOR2.

9.5 Input capture

Input capture is always enabled. Each time a capture event occurs on one of the two input capture pins, the contents of the timer is transferred to the corresponding 16-bit input capture register ICRAH:ICRAL or ICRBH:ICRBL. The capture event is defined by the Input Capture Edge Select – ICESx bit (x being A or B) in the CCCRx register. The user will have to configure the associated I/O pin as an input in order for an external event to trigger a capture.

A simple noise filter can be enabled on the input capture input. When the Input Capture Noise Filter ICNFx bit is set, the capture logic needs to see four consecutive samples of the same value in order to recognize an edge as a capture event. The inputs are sampled every two CCLK periods regardless of the speed of the timer.

An event counter can be set to delay a capture by a number of capture events. The three bits ICECx2, ICECx1 and ICECx0 in the CCCRx register determine the number of edges the capture logic has to see before an input capture occurs.

When a capture event is detected, the Timer Input Capture x (x is A or B) Interrupt Flag – TICF2x (TIFR2.1 or TIFR2.0) is set. If EA and the Timer Input Capture x Enable bit – TICIE2x (TICR2.1 or TICR2.0) is set as well as the ECCU (IEN1.4) bit is set, the program counter will be vectored to the corresponding interrupt. The interrupt flag must be cleared manually by writing a logic 0 to it.

When reading the input capture register, ICRxL must be read first. When ICRxL is read, the contents of the capture register high byte are transferred to a shadow register. When ICRxH is read, the contents of the shadow register are read instead. (If a read from ICRxL is followed by another read from ICRxL without ICRxH being read in between, the new value of the capture register high byte (from the last ICRxL read) will be in the shadow register).

Table 36: Event delay counter for input capture

ICECx2	ICECx1	ICECx0	Delay (numbers of edges)
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	7
1	1	1	15

9.6 PWM operation

PWM Operation has two main modes, asymmetrical and symmetrical. These modes of timer operation are selected by writing 10H or 11H to TMOD21:TMOD20 as shown in Section 9.3 “Basic timer operation”.

In asymmetrical PWM operation, the CCU Timer operates in downcounting mode regardless of the setting of TDIR2. In this case, TDIR2 will always read 1.

In symmetrical mode, the timer counts up/down alternately and the value of TDIR2 has no effect. The main difference from basic timer operation is the operation of the compare module, which in PWM mode is used for PWM waveform generation. Table 37 shows the behavior of the compare pins in PWM mode.

The user will have to configure the output compare pins as outputs in order to enable the PWM output. As with basic timer operation, when the PWM (compare) pins are connected to the compare logic, their logic state remains unchanged. However, since the bit FCO is used to hold the halt value, only a compare event can change the state of the pin.

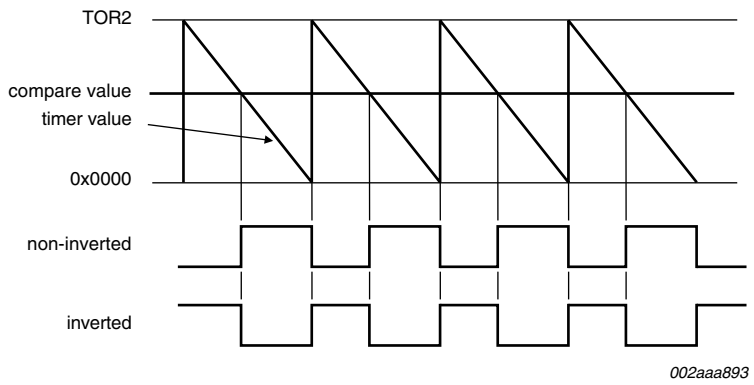


Fig 21. Asymmetrical PWM, downcounting.

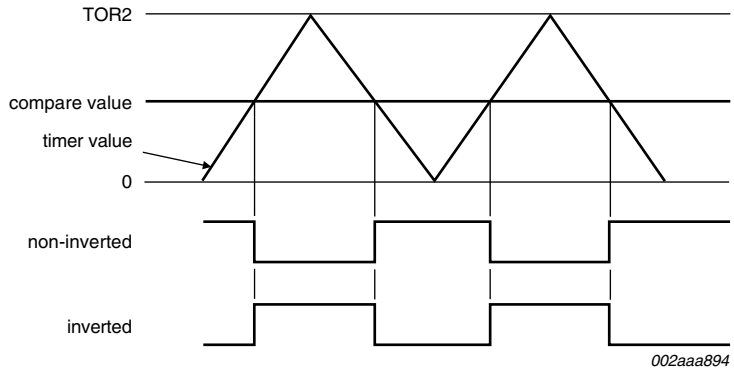


Fig 22. Symmetrical PWM.

The CCU Timer Overflow interrupt flag is set when the counter changes direction at the top. For example, if TOR contains 01FFH, CCU Timer will count: …01FEH, 01FFH, 01FEH,… The flag is set in the counter cycle after the change from TOR to TOR-1.

When the timer changes direction at the bottom, in this example, it counts …,0001H, 0000H, 0001H,… The CCU Timer overflow interrupt flag is set in the counter CCUCLK cycle after the transition from 0001H to 0000H.

The status of the TDIR2 bit in TCR20 reflects the current counting direction. Writing to this bit while operating in symmetrical mode has no effect.

9.7 Alternating output mode

In asymmetrical mode, the user can program PWM channels A/B and C/D as alternating pairs for bridge drive control. By setting ALTAB or ALTCD bits in TCR20, the output of these PWM channels are alternately gated on every counter cycle. This is shown in the following figure:

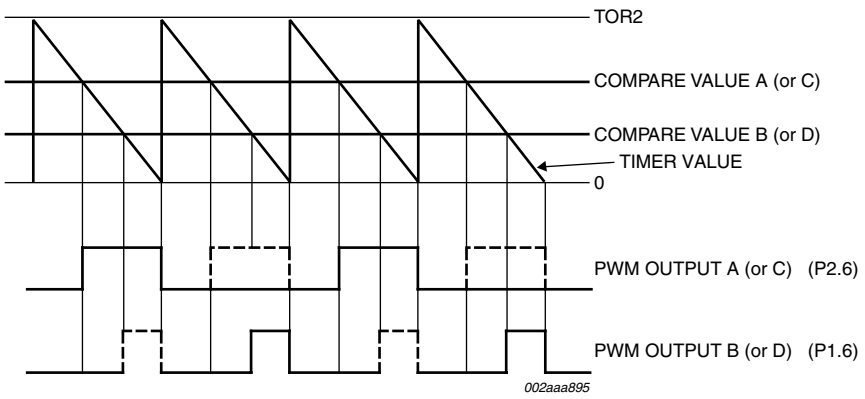


Fig 23. Alternate output mode.

Table 37: Output compare pin behavior

OCMx1[1]	OCMx0[1]	Output Compare pin behavior		
		Basic timer mode	Asymmetrical PWM	Symmetrical PWM
0	0	Output compare disabled. On power-on, this is the default state, and pins are configured as inputs.		
0	1	Set when compare in operation. Cleared on compare match.[2]	Non-Inverted PWM. Set on compare match. Cleared on CCU Timer underflow.	Non-Inverted PWM. Cleared on compare match, upcounting. Set on compare match, downcounting.
1	0	invalid configuration		
1	1	Toggles on compare match[2]	Inverted PWM. Cleared on compare match. Set on CCU Timer underflow.[2]	Inverted PWM. Set on compare match, upcounting. Cleared on compare match, downcounting.[2]

[1] x = A, B, C, D

[2] 'ON' means in the CCUCLK cycle after the event takes place.

9.8 Synchronized PWM register update

When the OCRx registers are written, a built in mechanism ensures that the value is not updated in the middle of a PWM pulse. This could result in an odd-length pulse. When the registers are written, the values are placed in two shadow registers, as is the case in basic timer operation mode. Writing to TCOU2 will cause the contents of the shadow registers to be updated on the next CCU Timer overflow. If OCRxH and/or OCRxL are read before the value is updated, the most currently written value is read.

9.9 HALT

Setting the HLTEN bit in TCR20 enables the PWM Halt Function. When halt function is enabled, a capture event as enabled for the Input Capture A pin will immediately stop all activity on the PWM pins and set them to a predetermined state defined by FCOx bit. In PWM Mode, the FCOx bits in the CCCRx register hold the value the pin is forced to during halt. The value of the setting can be read back. The capture function and the interrupt will still operate as normal even if it has this added functionality enabled. When the PWM unit is halted, the timer will still run as normal. The HLTRN bit in TCR20 will be set to indicate that a halt took place. In order to re-activate the PWM, the user must clear the HLTRN bit. The user can force the PWM unit into halt by writing a logic 1 to HLTRN bit.

9.10 PLL operation

The PWM module features a Phase Locked Loop that can be used to generate a CCUCLK frequency between 16 MHz and 32 MHz. At this frequency the PWM module provides ultrasonic PWM frequency with 10-bit resolution provided that the crystal frequency is 1 MHz or higher (The PWM resolution is programmable up to 16 bits by writing to TOR2H:TOR2L). The PLL is fed an input signal of 0.5 MHz to 1 MHz and generates an output signal of 32 times the input frequency. This signal is used to clock the timer. The user will have to set a divider that scales PCLK by a factor of 1 to 16. This divider is found in the SFR register TCR21. The PLL frequency can be expressed as follows:

PLL frequency = PCLK / (N+1)

Where: N is the value of PLLDV3:0.

Since N ranges in 0 to 15, the CCLK frequency can be in the range of PCLK to $^{PCLK}/_{16}$.

Table 38: CCU control register 1 (TCR21 - address F9h) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	TCOU2	-	-	-	PLLDV.3	PLLDV.2	PLLDV.1	PLLDV.0
Reset	0	x	x	x	0	0	0	0

Table 39: CCU control register 1 (TCR21 - address F9h) bit description

Bit	Symbol	Description
0:3	PLLDV.3:0	PLL frequency divider.
4:6	-	Reserved.
7	TCOU2	In basic timer mode, writing a logic 1 to TCOU2 will cause the values to be latched immediately and the value of TCOU2 will always read as logic 0. In PWM mode, writing a logic 1 to TCOU2 will cause the contents of the shadow registers to be updated on the next CCU Timer overflow. As long as the latch is pending, TCOU2 will read as logic 1 and will return to logic 0 when the latching takes place. TCOU2 also controls the latching of the Output Compare registers OCRAx, OCRBx and OCRCx.

Setting the PLLEN bit in TCR20 starts the PLL. When PLLEN is set, it will not read back a one until the PLL is in lock. At this time, the PWM unit is ready to operate and the timer can be enabled. The following start-up sequence is recommended:

1. Set up the PWM module without starting the timer.
2. Calculate the right division factor so that the PLL receives an input clock signal of 500 kHz - 1 MHz. Write this value to PLLDV.
3. Set PLLEN. Wait until the bit reads one
4. Start the timer by writing a value to bits TMOD21, TMOD20

When the timer runs from the PLL, the timer operates asynchronously to the rest of the microcontroller. Some restrictions apply:

- The user is discouraged from writing or reading the timer in asynchronous mode. The results may be unpredictable
- Interrupts and flags are asynchronous. There will be delay as the event may not actually be recognized until some CCLK cycles later (for interrupts and reads)

9.11 CCU interrupt structure

There are seven independent sources of interrupts in the CCU: timer overflow, captured input events on Input Capture blocks A/B, and compare match events on Output Compare blocks A through D. One common interrupt vector is used for the CCU service routine and interrupts can occur simultaneously in system usage. To resolve this situation, a priority encode function of the seven interrupt bits in TIFR2 SFR is implemented (after each bit is AND-ed with the corresponding interrupt enable bit in the TICR2 register). The order of priority is fixed as follows, from highest to lowest:

- TOIF2
- TICF2A
- TICF2B
- TOCF2A
- TOCF2B
- TOCF2C
- TOCF2D

An interrupt service routine for the CCU can be as follows:

1. Read the priority-encoded value from the TISE2 register to determine the interrupt source to be handled.

2. After the current (highest priority) event is serviced, write a logic 0 to the corresponding interrupt flag bit in the TIFR2 register to clear the flag.
3. Read the TISE2 register. If the priority-encoded interrupt source is '000', all CCU interrupts are serviced and a return from interrupt can occur. Otherwise, return to step 2 for the next interrupt.

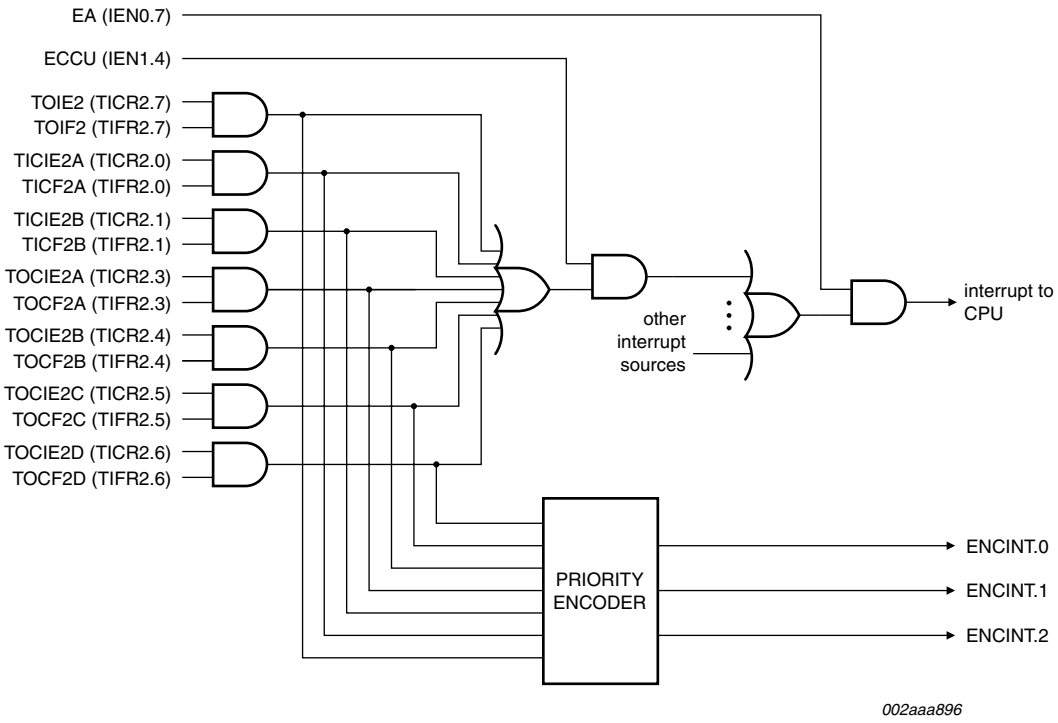


Fig 24. Capture/compare unit interrupts.

Table 40: CCU interrupt status encode register (TISE2 - address DEh) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	-	-	-	-	-	ENCINT.2	ENCINT.1	ENCINT.0
Reset	x	x	x	x	x	0	0	0

Table 41: CCU interrupt status encode register (TISE2 - address DEh) bit description

Bit	Symbol	Description
2:0	ENCINT.2:0	CCU Interrupt Encode output. When multiple interrupts happen, more than one interrupt flag is set in CCU Interrupt Flag Register (TIFR2). The encoder output can be read to determine which interrupt is to be serviced. The user must write a logic 0 to clear the corresponding interrupt flag bit in the TIFR2 register after the corresponding interrupt has been serviced. Refer to Table 43 for TIFR2 description. **000** — No interrupt pending. **001** — Output Compare Event D interrupt (lowest priority) **010** — Output Compare Event C interrupt. **011** — Output Compare Event B interrupt. **100** — Output Compare Event A interrupt. **101** — Input Capture Event B interrupt. **110** — Input Capture Event A interrupt. **111** — CCU Timer Overflow interrupt (highest priority).
3:7	-	Reserved.

Table 42: CCU interrupt flag register (TIFR2 - address E9h) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	TOIF2	TOCF2D	TOCF2C	TOCF2B	TOCF2A	-	TICF2B	TICF2A
Reset	0	0	0	0	0	x	0	0

Table 43: CCU interrupt flag register (TIFR2 - address E9h) bit description

Bit	Symbol	Description
0	TICF2A	Input Capture Channel A Interrupt Flag Bit. Set by hardware when an input capture event is detected. Cleared by software.
1	TICF2B	Input Capture Channel B Interrupt Flag Bit. Set by hardware when an input capture event is detected. Cleared by software.
2	-	Reserved for future use. Should not be set to logic 1 by user program.
3	TOCF2A	Output Compare Channel A Interrupt Flag Bit. Set by hardware when the contents of TH2:TL2 match that of OCRHA:OCRLA. Compare channel A must be enabled in order to generate this interrupt. If EA bit in IEN0, ECCU bit in IEN1 and TOCIE2A bit are all set, the program counter will vectored to the corresponding interrupt. Cleared by software.
4	TOCF2B	Output Compare Channel B Interrupt Flag Bit. Set by hardware when the contents of TH2:TL2 match that of OCRHB:OCRLB. Compare channel B must be enabled in order to generate this interrupt. If EA bit in IEN0, ECCU bit in IEN1 and TOCIE2B bit are set, the program counter will vectored to the corresponding interrupt. Cleared by software.
5	TOCF2C	Output Compare Channel C Interrupt Flag Bit. Set by hardware when the contents of TH2:TL2 match that of OCRHC:OCRLC. Compare channel C must be enabled in order to generate this interrupt. If EA bit in IEN0, ECCU bit in IEN1 and TOCIE2C bit are all set, the program counter will vectored to the corresponding interrupt. Cleared by software.
6	TOCF2D	Output Compare Channel D Interrupt Flag Bit. Set by hardware when the contents of TH2:TL2 match that of OCRHD:OCRLD. Compare channel D must be enabled in order to generate this interrupt. If EA bit in IEN0, ECCU bit in IEN1 and TOCIE2D bit are all set, the program counter will vectored to the corresponding interrupt. Cleared by software.
7	TOIF2	CCU Timer Overflow Interrupt Flag bit. Set by hardware on CCU Timer overflow. Cleared by software.

Table 44: CCU interrupt control register (TICR2 - address C9h) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	TOIE2	TOCIE2D	TOCIE2C	TOCIE2B	TOCIE2A	-	TICIE2B	TICIE2A
Reset	0	0	0	0	0	x	0	0

Table 45: CCU interrupt control register (TICR2 - address C9h) bit description

Bit	Symbol	Description
0	TICIE2A	Input Capture Channel A Interrupt Enable Bit. If EA bit and this bit all be set, when a capture event is detected, the program counter will vectored to the corresponding interrupt.
1	TICIE2B	Input Capture Channel B Interrupt Enable Bit. If EA bit and this bit all be set, when a capture event is detected, the program counter will vectored to the corresponding interrupt.
2	-	Reserved for future use. Should not be set to logic 1 by user program.
3	TOCIE2A	Output Compare Channel A Interrupt Enable Bit. If EA bit and this bit are set to 1, when compare channel is enabled and the contents of TH2:TL2 match that of OCRHA:OCRLA, the program counter will vectored to the corresponding interrupt.
4	TOCIE2B	Output Compare Channel B Interrupt Enable Bit. If EA bit and this bit are set to 1, when compare channel B is enabled and the contents of TH2:TL2 match that of OCRHB:OCRLB, the program counter will vectored to the corresponding interrupt.
5	TOCIE2C	Output Compare Channel C Interrupt Enable Bit. If EA bit and this bit are set to 1, when compare channel C is enabled and the contents of TH2:TL2 match that of OCRHC:OCRLC, the program counter will vectored to the corresponding interrupt.
6	TOCIE2D	Output Compare Channel D Interrupt Enable Bit. If EA bit and this bit are set to 1, when compare channel D is enabled and the contents of TH2:TL2 match that of OCRHD:OCRLD, the program counter will vectored to the corresponding interrupt.
7	TOIE2	CCU Timer Overflow Interrupt Enable bit.

10. UART

The P89LPC932A1 has an enhanced UART that is compatible with the conventional 80C51 UART except that Timer 2 overflow cannot be used as a baud rate source. The P89LPC932A1 does include an independent Baud Rate Generator. The baud rate can be selected from the oscillator (divided by a constant), Timer 1 overflow, or the independent Baud Rate Generator. In addition to the baud rate generation, enhancements over the standard 80C51 UART include Framing Error detection, break detect, automatic address recognition, selectable double buffering and several interrupt options.

The UART can be operated in 4 modes, as described in the following sections.

10.1 Mode 0

Serial data enters and exits through RxD. TxD outputs the shift clock. 8 bits are transmitted or received, LSB first. The baud rate is fixed at $^1/_{16}$ of the CPU clock frequency.

10.2 Mode 1

10 bits are transmitted (through TxD) or received (through RxD): a start bit (logic 0), 8 data bits (LSB first), and a stop bit (logic 1). When data is received, the stop bit is stored in RB8 in Special Function Register SCON. The baud rate is variable and is determined by the Timer 1 overflow rate or the Baud Rate Generator (see Section 10.6 “Baud Rate generator and selection” on page 59).

10.3 Mode 2

11 bits are transmitted (through TxD) or received (through RxD): start bit (logic 0), 8 data bits (LSB first), a programmable 9th data bit, and a stop bit (logic 1). When data is transmitted, the 9th data bit (TB8 in SCON) can be assigned the value of 0 or 1. Or, for example, the parity bit (P, in the PSW) could be moved into TB8. When data is received, the 9th data bit goes into RB8 in Special Function Register SCON and the stop bit is not saved. The baud rate is programmable to either $1/16$ or $1/32$ of the CCLK frequency, as determined by the SMOD1 bit in PCON.

10.4 Mode 3

11 bits are transmitted (through TxD) or received (through RxD): a start bit (logic 0), 8 data bits (LSB first), a programmable 9th data bit, and a stop bit (logic 1). Mode 3 is the same as Mode 2 in all respects except baud rate. The baud rate in Mode 3 is variable and is determined by the Timer 1 overflow rate or the Baud Rate Generator (see Section 10.6 “Baud Rate generator and selection” on page 59).

In all four modes, transmission is initiated by any instruction that uses SBUF as a destination register. Reception is initiated in Mode 0 by the condition RI = 0 and REN = 1. Reception is initiated in the other modes by the incoming start bit if REN = 1.

10.5 SFR space

The UART SFRs are at the following locations:

Table 46: UART SFR addresses

Register	Description	SFR location
PCON	Power Control	87H
SCON	Serial Port (UART) Control	98H
SBUF	Serial Port (UART) Data Buffer	99H
SADDR	Serial Port (UART) Address	A9H
SADEN	Serial Port (UART) Address Enable	B9H
SSTAT	Serial Port (UART) Status	BAH
BRGR1	Baud Rate Generator Rate High Byte	BFH
BRGR0	Baud Rate Generator Rate Low Byte	BEH
BRGCON	Baud Rate Generator Control	BDH

10.6 Baud Rate generator and selection

The P89LPC932A1 enhanced UART has an independent Baud Rate Generator. The baud rate is determined by a value programmed into the BRGR1 and BRGR0 SFRs. The UART can use either Timer 1 or the baud rate generator output as determined by BRGCON[2:1] (see Figure 25). Note that Timer T1 is further divided by 2 if the SMOD1 bit (PCON.7) is set. The independent Baud Rate Generator uses CCLK.

10.7 Updating the BRGR1 and BRGR0 SFRs

The baud rate SFRs, BRGR1 and BRGR0 must only be loaded when the Baud Rate Generator is disabled (the BRGEN bit in the BRGCON register is logic 0). This avoids the loading of an interim value to the baud rate generator. **(CAUTION: If either BRGR0 or BRGR1 is written when BRGEN = 1, the result is unpredictable.)**

Table 47: UART baud rate generation

SCON.7 (SM0)	SCON.6 (SM1)	PCON.7 (SMOD1)	BRGCON.1 (SBRGS)	Receive/transmit baud rate for UART
0	0	X	X	$CCLK/16$
0	1	0	0	$CCLK/(256-TH1)64$
		1	0	$CCLK/(256-TH1)32$
		X	1	$CCLK/((BRGR1, BRGR0)+16)$
1	0	0	X	$CCLK/32$
		1	X	$CCLK/16$
1	1	0	0	$CCLK/(256-TH1)64$
		1	0	$CCLK/(256-TH1)32$
		X	1	$CCLK/((BRGR1, BRGR0)+16)$

Table 48: Baud Rate Generator Control register (BRGCON - address BDh) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	--	-	-	-	-	-	SBRGS	BRGEN
Reset	x	x	x	x	x	x	0	0

Table 49: Baud Rate Generator Control register (BRGCON - address BDh) bit description

Bit	Symbol	Description
0	BRGEN	Baud Rate Generator Enable. Enables the baud rate generator. BRGR1 and BRGR0 can only be written when BRGEN = 0.
1	SBRGS	Select Baud Rate Generator as the source for baud rates to UART in modes 1 and 3 (see Table 47 for details)
2:7	-	reserved

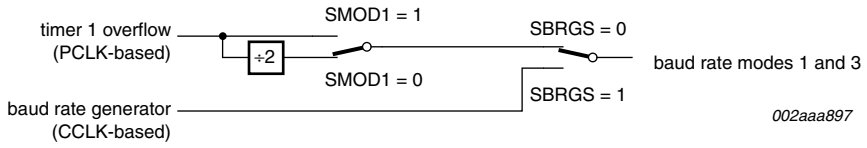


Fig 25. Baud rate generation for UART (Modes 1, 3).

10.8 Framing error

A Framing error occurs when the stop bit is sensed as a logic 0. A Framing error is reported in the status register (SSTAT). In addition, if SMOD0 (PCON.6) is 1, framing errors can be made available in SCON.7. If SMOD0 is 0, SCON.7 is SM0. It is recommended that SM0 and SM1 (SCON[7:6]) are programmed when SMOD0 is logic 0.

10.9 Break detect

A break detect is reported in the status register (SSTAT). A break is detected when any 11 consecutive bits are sensed low. Since a break condition also satisfies the requirements for a framing error, a break condition will also result in reporting a framing error. Once a break condition has been detected, the UART will go into an idle state and remain in this idle state until a stop bit has been received. The break detect can be used to reset the device and force the device into ISP mode by setting the EBRR bit (AUXR1.6)

Table 50: Serial Port Control register (SCON - address 98h) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	SM0/FE	SM1	SM2	REN	TB8	RB8	TI	RI
Reset	x	x	x	x	x	x	0	0

Table 51: Serial Port Control register (SCON - address 98h) bit description

Bit	Symbol	Description
0	RI	Receive interrupt flag. Set by hardware at the end of the 8th bit time in Mode 0, or approximately halfway through the stop bit time in Mode 1. For Mode 2 or Mode 3, if SMOD0, it is set near the middle of the 9th data bit (bit 8). If SMOD0 = 1, it is set near the middle of the stop bit (see SM2 - SCON.5 - for exceptions). Must be cleared by software.
1	TI	Transmit interrupt flag. Set by hardware at the end of the 8th bit time in Mode 0, or at the stop bit (see description of INTLO bit in SSTAT register) in the other modes. Must be cleared by software.
2	RB8	The 9th data bit that was received in Modes 2 and 3. In Mode 1 (SM2 must be 0), RB8 is the stop bit that was received. In Mode 0, RB8 is undefined.
3	TB8	The 9th data bit that will be transmitted in Modes 2 and 3. Set or clear by software as desired.
4	REN	Enables serial reception. Set by software to enable reception. Clear by software to disable reception.
5	SM2	Enables the multiprocessor communication feature in Modes 2 and 3. In Mode 2 or 3, if SM2 is set to 1, then RI will not be activated if the received 9th data bit (RB8) is 0. In Mode 0, SM2 should be 0. In Mode 1, SM2 must be 0.
6	SM1	With SM0 defines the serial port mode, see Table 52.
7	SM0/FE	The use of this bit is determined by SMOD0 in the PCON register. If SMOD0 = 0, this bit is read and written as SM0, which with SM1, defines the serial port mode. If SMOD0 = 1, this bit is read and written as FE (Framing Error). FE is set by the receiver when an invalid stop bit is detected. Once set, this bit cannot be cleared by valid frames but is cleared by software. (Note: UART mode bits SM0 and SM1 should be programmed when SMOD0 is logic 0 - default mode on any reset.)

Table 52: Serial Port modes

SM0, SM1	UART mode	UART baud rate
00	Mode 0: shift register	$^{CCLK}/_{16}$ (default mode on any reset)
01	Mode 1: 8-bit UART	Variable (see Table 47)
10	Mode 2: 9-bit UART	$^{CCLK}/_{32}$ or $^{CCLK}/_{16}$
11	Mode 3: 9-bit UART	Variable (see Table 47)

Table 53: Serial Port Status register (SSTAT - address BAh) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	DBMOD	INTLO	CIDIS	DBISEL	FE	BR	OE	STINT
Reset	x	x	x	x	x	x	0	0

Table 54: Serial Port Status register (SSTAT - address BAh) bit description

Bit	Symbol	Description
0	STINT	Status Interrupt Enable. When set = 1, FE, BR, or OE can cause an interrupt. The interrupt used (vector address 0023h) is shared with RI (CIDIS = 1) or the combined TI/RI (CIDIS = 0). When cleared = 0, FE, BR, OE cannot cause an interrupt. (Note: FE, BR, or OE is often accompanied by a RI, which will generate an interrupt regardless of the state of STINT). Note that BR can cause a break detect reset if EBRR (AUXR1.6) is set to logic 1.
1	OE	Overrun Error flag is set if a new character is received in the receiver buffer while it is still full (before the software has read the previous character from the buffer), i.e., when bit 8 of a new byte is received while RI in SCON is still set. Cleared by software.
2	BR	Break Detect flag. A break is detected when any 11 consecutive bits are sensed low. Cleared by software.
3	FE	Framing error flag is set when the receiver fails to see a valid STOP bit at the end of the frame. Cleared by software.
4	DBISEL	Double buffering transmit interrupt select. Used only if double buffering is enabled. This bit controls the number of interrupts that can occur when double buffering is enabled. When set, one transmit interrupt is generated after each character written to SBUF, and there is also one more transmit interrupt generated at the beginning (INTLO = 0) or the end (INTLO = 1) of the STOP bit of the last character sent (i.e., no more data in buffer). This last interrupt can be used to indicate that all transmit operations are over. When cleared = 0, only one transmit interrupt is generated per character written to SBUF. Must be logic 0 when double buffering is disabled. Note that except for the first character written (when buffer is empty), the location of the transmit interrupt is determined by INTLO. When the first character is written, the transmit interrupt is generated immediately after SBUF is written.
5	CIDIS	Combined Interrupt Disable. When set = 1, Rx and Tx interrupts are separate. When cleared = 0, the UART uses a combined Tx/Rx interrupt (like a conventional 80C51 UART). This bit is reset to logic 0 to select combined interrupts.
6	INTLO	Transmit interrupt position. When cleared = 0, the Tx interrupt is issued at the beginning of the stop bit. When set = 1, the Tx interrupt is issued at end of the stop bit. Must be logic 0 for mode 0. Note that in the case of single buffering, if the Tx interrupt occurs at the end of a STOP bit, a gap may exist before the next start bit.
7	DBMOD	Double buffering mode. When set = 1 enables double buffering. Must be logic 0 for UART mode 0. In order to be compatible with existing 80C51 devices, this bit is reset to logic 0 to disable double buffering.

10.10 More about UART Mode 0

In Mode 0, a write to SBUF will initiate a transmission. At the end of the transmission, TI (SCON.1) is set, which must be cleared in software. Double buffering must be disabled in this mode.

Reception is initiated by clearing RI (SCON.0). Synchronous serial transfer occurs and RI will be set again at the end of the transfer. When RI is cleared, the reception of the next character will begin. Refer to Figure 26

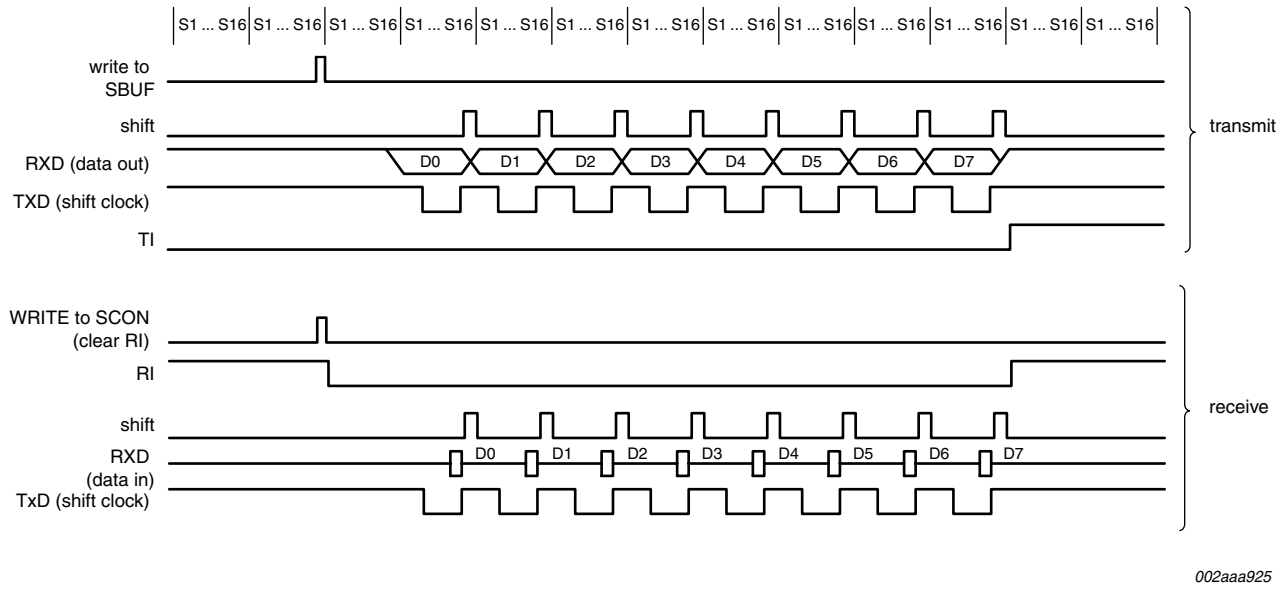


Fig 26. Serial Port Mode 0 (double buffering must be disabled).

10.11 More about UART Mode 1

Reception is initiated by detecting a 1-to-0 transition on RxD. RxD is sampled at a rate 16 times the programmed baud rate. When a transition is detected, the divide-by-16 counter is immediately reset. Each bit time is thus divided into 16 counter states. At the 7th, 8th, and 9th counter states, the bit detector samples the value of RxD. The value accepted is the value that was seen in at least 2 of the 3 samples. This is done for noise rejection. If the value accepted during the first bit time is not 0, the receive circuits are reset and the receiver goes back to looking for another 1-to-0 transition. This provides rejection of false start bits. If the start bit proves valid, it is shifted into the input shift register, and reception of the rest of the frame will proceed.

The signal to load SBUF and RB8, and to set RI, will be generated if, and only if, the following conditions are met at the time the final shift pulse is generated: RI = 0 and either SM2 = 0 or the received stop bit = 1. If either of these two conditions is not met, the received frame is lost. If both conditions are met, the stop bit goes into RB8, the 8 data bits go into SBUF, and RI is activated.

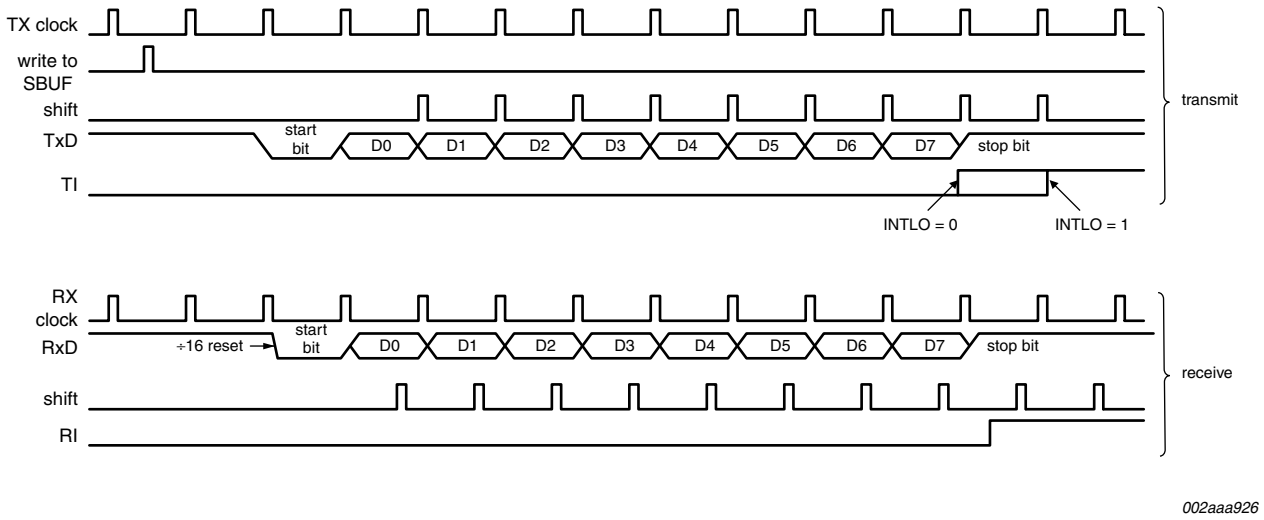


Fig 27. Serial Port Mode 1 (only single transmit buffering case is shown).

10.12 More about UART Modes 2 and 3

Reception is the same as in Mode 1.

The signal to load SBUF and RB8, and to set RI, will be generated if, and only if, the following conditions are met at the time the final shift pulse is generated. (a) RI = 0, and (b) Either SM2 = 0, or the received 9th data bit = 1. If either of these conditions is not met, the received frame is lost, and RI is not set. If both conditions are met, the received 9th data bit goes into RB8, and the first 8 data bits go into SBUF.

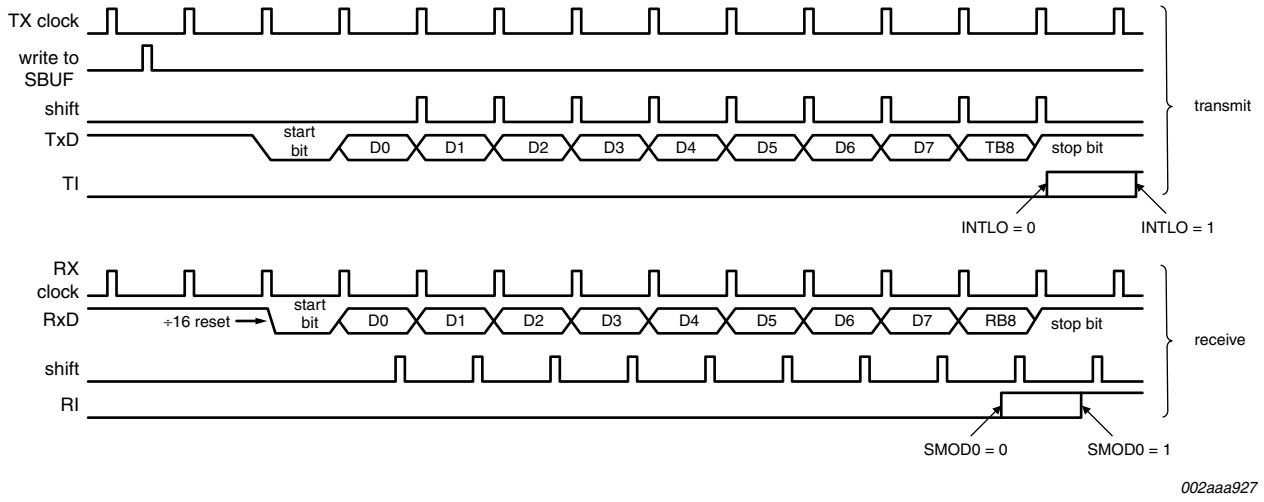


Fig 28. Serial Port Mode 2 or 3 (only single transmit buffering case is shown).

10.13 Framing error and RI in Modes 2 and 3 with SM2 = 1

If SM2 = 1 in modes 2 and 3, RI and FE behaves as in the following table.

Table 55: FE and RI when SM2= 1 in Modes 2 and 3

Mode	PCON.6 (SMOD0)	RB8	RI	FE
2	0	0	No RI when RB8 = 0	Occurs during STOP bit
		1	Similar to Figure 28, with SMOD0 = 0, RI occurs during RB8, one bit before FE	Occurs during STOP bit
3	1	0	No RI when RB8 = 0	Will NOT occur
		1	Similar to [28], with SMOD0 = 1, RI occurs during STOP bit	Occurs during STOP bit

10.14 Break detect

A break is detected when 11 consecutive bits are sensed low and is reported in the status register (SSTAT). For Mode 1, this consists of the start bit, 8 data bits, and two stop bit times. For Modes 2 and 3, this consists of the start bit, 9 data bits, and one stop bit. The break detect bit is cleared in software or by a reset. The break detect can be used to reset the device and force the device into ISP mode. This occurs if the UART is enabled and the the EBRR bit (AUXR1.6) is set and a break occurs.

10.15 Double buffering

The UART has a transmit double buffer that allows buffering of the next character to be written to SBUF while the first character is being transmitted. Double buffering allows transmission of a string of characters with only one stop bit between any two characters, provided the next character is written between the start bit and the stop bit of the previous character.

Double buffering can be disabled. If disabled (DBMOD, i.e. SSTAT.7 = 0), the UART is compatible with the conventional 80C51 UART. If enabled, the UART allows writing to SnBUF while the previous data is being shifted out.

10.16 Double buffering in different modes

Double buffering is only allowed in Modes 1, 2 and 3. When operated in Mode 0, double buffering must be disabled (DBMOD = 0).

10.17 Transmit interrupts with double buffering enabled (Modes 1, 2, and 3)

Unlike the conventional UART, when double buffering is enabled, the Tx interrupt is generated when the double buffer is ready to receive new data. The following occurs during a transmission (assuming eight data bits):

1. The double buffer is empty initially.
2. The CPU writes to SBUF.
3. The SBUF data is loaded to the shift register and a Tx interrupt is generated immediately.
4. If there is more data, go to 6, else continue.
5. If there is no more data, then:
   - If DBISEL is logic 0, no more interrupts will occur.

– If DBISEL is logic 1 and INTLO is logic 0, a Tx interrupt will occur at the beginning of the STOP bit of the data currently in the shifter (which is also the last data).
– If DBISEL is logic 1 and INTLO is logic 1, a Tx interrupt will occur at the end of the STOP bit of the data currently in the shifter (which is also the last data).
– Note that if DBISEL is logic 1 and the CPU is writing to SBUF when the STOP bit of the last data is shifted out, there can be an uncertainty of whether a Tx interrupt is generated already with the UART not knowing whether there is any more data following.

6. If there is more data, the CPU writes to SBUF again. Then:
   – If INTLO is logic 0, the new data will be loaded and a Tx interrupt will occur at the beginning of the STOP bit of the data currently in the shifter.
   – If INTLO is logic 1, the new data will be loaded and a Tx interrupt will occur at the end of the STOP bit of the data currently in the shifter.
   – Go to 3.

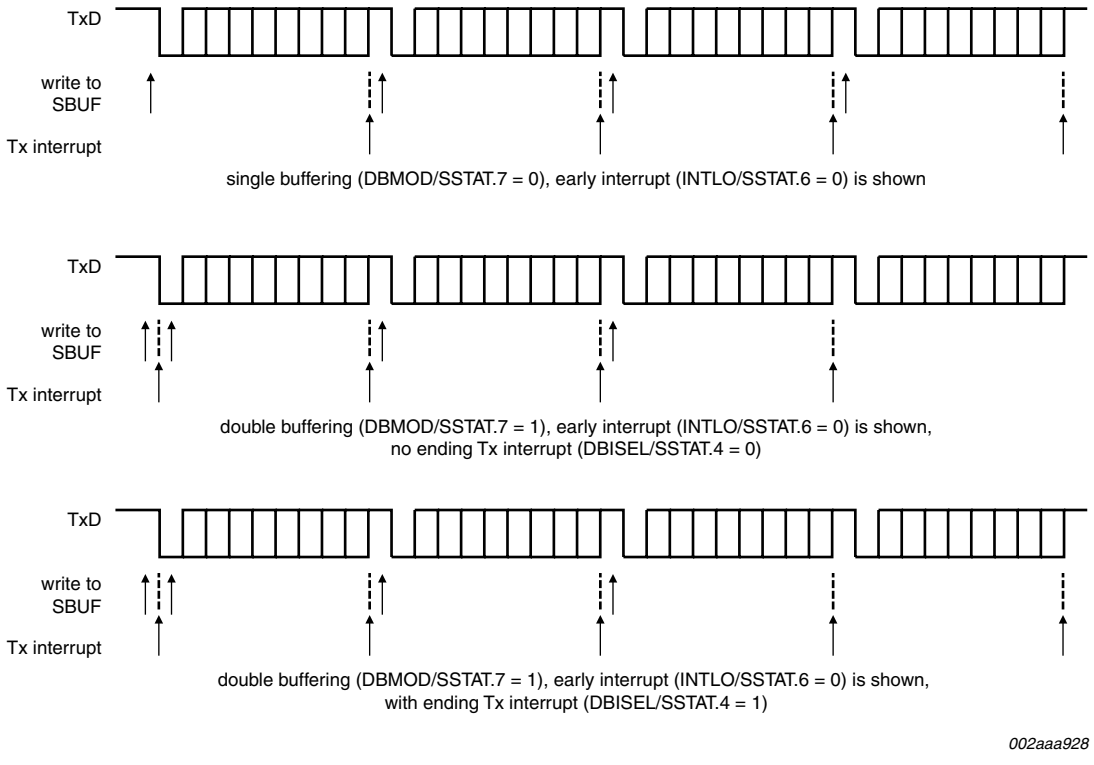


Fig 29. Transmission with and without double buffering.

10.18 The 9th bit (bit 8) in double buffering (Modes 1, 2, and 3)

If double buffering is disabled (DBMOD, i.e. SSTAT.7 = 0), TB8 can be written before or after SBUF is written, provided TB8 is updated before that TB8 is shifted out. TB8 must not be changed again until after TB8 shifting has been completed, as indicated by the Tx interrupt.

If double buffering is enabled, TB8 MUST be updated before SBUF is written, as TB8 will be double-buffered together with SBUF data. The operation described in the Section 10.17 “Transmit interrupts with double buffering enabled (Modes 1, 2, and 3)” on page 65 becomes as follows:

1. The double buffer is empty initially.
2. The CPU writes to TB8.
3. The CPU writes to SBUF.
4. The SBUF/TB8 data is loaded to the shift register and a Tx interrupt is generated immediately.
5. If there is more data, go to 7, else continue on 6.
6. If there is no more data, then:
   - If DBISEL is logic 0, no more interrupt will occur.
   - If DBISEL is logic 1 and INTLO is logic 0, a Tx interrupt will occur at the beginning of the STOP bit of the data currently in the shifter (which is also the last data).
   - If DBISEL is logic 1 and INTLO is logic 1, a Tx interrupt will occur at the end of the STOP bit of the data currently in the shifter (which is also the last data).
7. If there is more data, the CPU writes to TB8 again.
8. The CPU writes to SBUF again. Then:
   - If INTLO is logic 0, the new data will be loaded and a Tx interrupt will occur at the beginning of the STOP bit of the data currently in the shifter.
   - If INTLO is logic 1, the new data will be loaded and a Tx interrupt will occur at the end of the STOP bit of the data currently in the shifter.
9. Go to 4.
10. Note that if DBISEL is logic 1 and the CPU is writing to SBUF when the STOP bit of the last data is shifted out, there can be an uncertainty of whether a Tx interrupt is generated already with the UART not knowing whether there is any more data following.

10.19 Multiprocessor communications

UART modes 2 and 3 have a special provision for multiprocessor communications. In these modes, 9 data bits are received or transmitted. When data is received, the 9th bit is stored in RB8. The UART can be programmed such that when the stop bit is received, the serial port interrupt will be activated only if RB8 = 1. This feature is enabled by setting bit SM2 in SCON. One way to use this feature in multiprocessor systems is as follows:

When the master processor wants to transmit a block of data to one of several slaves, it first sends out an address byte which identifies the target slave. An address byte differs from a data byte in that the 9th bit is 1 in an address byte and 0 in a data byte. With SM2 = 1, no slave will be interrupted by a data byte. An address byte, however, will interrupt all slaves, so that each slave can examine the received byte and see if it is being addressed. The addressed slave will clear its SM2 bit and prepare to receive the data bytes that follow. The slaves that weren’t being addressed leave their SM2 bits set and go on about their business, ignoring the subsequent data bytes.

Note that SM2 has no effect in Mode 0, and must be logic 0 in Mode 1.

10.20 Automatic address recognition

Automatic address recognition is a feature which allows the UART to recognize certain addresses in the serial bit stream by using hardware to make the comparisons. This feature saves a great deal of software overhead by eliminating the need for the software to examine every serial address which passes by the serial port. This feature is enabled by setting the SM2 bit in SCON. In the 9 bit UART modes (mode 2 and mode 3), the Receive Interrupt flag (RI) will be automatically set when the received byte contains either the 'Given' address or the 'Broadcast' address. The 9 bit mode requires that the 9th information bit is a 1 to indicate that the received information is an address and not data.

Using the Automatic Address Recognition feature allows a master to selectively communicate with one or more slaves by invoking the Given slave address or addresses. All of the slaves may be contacted by using the Broadcast address. Two special Function Registers are used to define the slave's address, SADDR, and the address mask, SADEN. SADEN is used to define which bits in the SADDR are to be used and which bits are 'don't care'. The SADEN mask can be logically ANDed with the SADDR to create the 'Given' address which the master will use for addressing each of the slaves. Use of the Given address allows multiple slaves to be recognized while excluding others. The following examples will help to show the versatility of this scheme:

Table 56: Slave 0/1 examples

Example 1				Example 2			
Slave 0	SADDR	=	1100 0000	Slave 1	SADDR	=	1100 0000
	SADEN	=	1111 1101		SADEN	=	1111 1110
	Given	=	1100 00X0		Given	=	1100 000X

In the above example SADDR is the same and the SADEN data is used to differentiate between the two slaves. Slave 0 requires a 0 in bit 0 and it ignores bit 1. Slave 1 requires a 0 in bit 1 and bit 0 is ignored. A unique address for Slave 0 would be 1100 0010 since slave 1 requires a 0 in bit 1. A unique address for slave 1 would be 1100 0001 since a 1 in bit 0 will exclude slave 0. Both slaves can be selected at the same time by an address which has bit 0 = 0 (for slave 0) and bit 1 = 0 (for slave 1). Thus, both could be addressed with 1100 0000.

In a more complex system the following could be used to select slaves 1 and 2 while excluding slave 0:

Table 57: Slave 0/1/2 examples

Example 1				Example 2				Example 3			
Slave 0	SADDR	=	1100 0000	Slave 1	SADDR	=	1110 0000	Slave 2	SADDR	=	1100 0000
	SADEN	=	1111 1001		SADEN	=	1111 1010		SADEN	=	1111 1100
	Given	=	1100 0XX0		Given	=	1110 0X0X		Given	=	1110 00XX

In the above example the differentiation among the 3 slaves is in the lower 3 address bits. Slave 0 requires that bit 0 = 0 and it can be uniquely addressed by 1110 0110. Slave 1 requires that bit 1 = 0 and it can be uniquely addressed by 1110 and 0101. Slave 2 requires that bit 2 = 0 and its unique address is 1110 0011. To select Slaves 0 and 1 and exclude Slave 2 use address 1110 0100, since it is necessary to make bit 2 = 1 to exclude slave 2. The Broadcast Address for each slave is created by taking the logical OR of SADDR and SADEN. Zeros in this result are treated as don't-cares. In most cases,

interpreting the don't-cares as ones, the broadcast address will be FF hexadecimal. Upon reset SADDR and SADEN are loaded with 0s. This produces a given address of all 'don't cares' as well as a Broadcast address of all 'don't cares'. This effectively disables the Automatic Addressing mode and allows the microcontroller to use standard UART drivers which do not make use of this feature.

11. I²C interface

The I²C-bus uses two wires, serial clock (SCL) and serial data (SDA) to transfer information between devices connected to the bus, and has the following features:

- Bidirectional data transfer between masters and slaves
- Multimaster bus (no central master)
- Arbitration between simultaneously transmitting masters without corruption of serial data on the bus
- Serial clock synchronization allows devices with different bit rates to communicate via one serial bus
- Serial clock synchronization can be used as a handshake mechanism to suspend and resume serial transfer
- The I²C-bus may be used for test and diagnostic purposes

A typical I²C-bus configuration is shown in Figure 30. Depending on the state of the direction bit (R/W), two types of data transfers are possible on the I²C-bus:

- Data transfer from a master transmitter to a slave receiver. The first byte transmitted by the master is the slave address. Next follows a number of data bytes. The slave returns an acknowledge bit after each received byte.
- Data transfer from a slave transmitter to a master receiver. The first byte (the slave address) is transmitted by the master. The slave then returns an acknowledge bit. Next follows the data bytes transmitted by the slave to the master. The master returns an acknowledge bit after all received bytes other than the last byte. At the end of the last received byte, a 'not acknowledge' is returned. The master device generates all of the serial clock pulses and the START and STOP conditions. A transfer is ended with a STOP condition or with a repeated START condition. Since a repeated START condition is also the beginning of the next serial transfer, the I²C-bus will not be released.

The P89LPC932A1 device provides a byte-oriented I²C interface. It has four operation modes: Master Transmitter Mode, Master Receiver Mode, Slave Transmitter Mode and Slave Receiver Mode

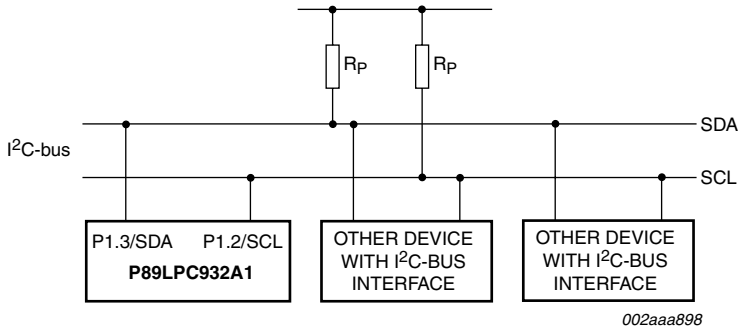


Fig 30. I²C-bus configuration.

The P89LPC932A1 CPU interfaces with the I²C-bus through six Special Function Registers (SFRs): I2CON (I²C Control Register), I2DAT (I²C Data Register), I2STAT (I²C Status Register), I2ADR (I²C Slave Address Register), I2SCLH (SCL Duty Cycle Register High Byte), and I2SCLL (SCL Duty Cycle Register Low Byte).

11.1 I²C data register

I2DAT register contains the data to be transmitted or the data received. The CPU can read and write to this 8-bit register while it is not in the process of shifting a byte. Thus this register should only be accessed when the SI bit is set. Data in I2DAT remains stable as long as the SI bit is set. Data in I2DAT is always shifted from right to left: the first bit to be transmitted is the MSB (bit 7), and after a byte has been received, the first bit of received data is located at the MSB of I2DAT.

Table 58: I²C data register (I2DAT - address DAh) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	I2DAT.7	I2DAT.6	I2DAT.5	I2DAT.4	I2DAT.3	I2DAT.2	I2DAT.1	I2DAT.0
Reset	0	0	0	0	0	0	0	0

11.2 I²C slave address register

I2ADR register is readable and writable, and is only used when the I²C interface is set to slave mode. In master mode, this register has no effect. The LSB of I2ADR is general call bit. When this bit is set, the general call address (00h) is recognized.

Table 59: I²C slave address register (I2ADR - address DBh) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	I2ADR.6	I2ADR.5	I2ADR.4	I2ADR.3	I2ADR.2	I2ADR.1	I2ADR.0	GC
Reset	0	0	0	0	0	0	0	0

Table 60: I²C slave address register (I2ADR - address DBh) bit description

Bit	Symbol	Description
0	GC	General call bit. When set, the general call address (00H) is recognized, otherwise it is ignored.
1:7	I2ADR1:7	7 bit own slave address. When in master mode, the contents of this register has no effect.

11.3 I²C control register

The CPU can read and write this register. There are two bits are affected by hardware: the SI bit and the STO bit. The SI bit is set by hardware and the STO bit is cleared by hardware.

CRSEL determines the SCL source when the I²C-bus is in master mode. In slave mode this bit is ignored and the bus will automatically synchronize with any clock frequency up to 400 kHz from the master I²C device. When CRSEL = 1, the I²C interface uses the Timer 1 overflow rate divided by 2 for the I²C clock rate. Timer 1 should be programmed by the user in 8 bit auto-reload mode (Mode 2).

Data rate of I²C-bus = Timer overflow rate / 2 = PCLK / (2*(256-reload value)).

If f_{osc} = 12 MHz, reload value is 0 to 255, so I²C data rate range is 11.72 Kbit/sec to 3000 Kbit/sec.

When CRSEL = 0, the I²C interface uses the internal clock generator based on the value of I2SCLL and I2CSCLH register. The duty cycle does not need to be 50 %.

The STA bit is START flag. Setting this bit causes the I²C interface to enter master mode and attempt transmitting a START condition or transmitting a repeated START condition when it is already in master mode.

The STO bit is STOP flag. Setting this bit causes the I²C interface to transmit a STOP condition in master mode, or recovering from an error condition in slave mode.

If the STA and STO are both set, then a STOP condition is transmitted to the I²C-bus if it is in master mode, and transmits a START condition afterwards. If it is in slave mode, an internal STOP condition will be generated, but it is not transmitted to the bus.

Table 61: I²C Control register (I2CON - address D8h) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	-	I2EN	STA	STO	SI	AA	-	CRSEL
Reset	x	0	0	0	0	0	x	0

Table 62: I²C Control register (I2CON - address D8h) bit description

Bit	Symbol	Description
0	CRSEL	SCL clock selection. When set = 1, Timer 1 overflow generates SCL, when cleared = 0, the internal SCL generator is used base on values of I2SCLH and I2SCLL.
1	-	reserved
2	AA	The Assert Acknowledge Flag. When set to 1, an acknowledge (low level to SDA) will be returned during the acknowledge clock pulse on the SCL line on the following situations: (1)The 'own slave address' has been received. (2)The general call address has been received while the general call bit (GC) in I2ADR is set. (3) A data byte has been received while the I²C interface is in the Master Receiver Mode. (4)A data byte has been received while the I²C interface is in the addressed Slave Receiver Mode. When cleared to 0, an not acknowledge (high level to SDA) will be returned during the acknowledge clock pulse on the SCL line on the following situations: (1) A data byte has been received while the I²C interface is in the Master Receiver Mode. (2) A data byte has been received while the I²C interface is in the addressed Slave Receiver Mode.

Table 62: I²C Control register (I2CON - address D8h) bit description …continued

Bit	Symbol	Description
3	SI	I²C Interrupt Flag. This bit is set when one of the 25 possible I²C states is entered. When EA bit and EI2C (IEN1.0) bit are both set, an interrupt is requested when SI is set. Must be cleared by software by writing 0 to this bit.
4	STO	STOP Flag. STO = 1: In master mode, a STOP condition is transmitted to the I²C-bus. When the bus detects the STOP condition, it will clear STO bit automatically. In slave mode, setting this bit can recover from an error condition. In this case, no STOP condition is transmitted to the bus. The hardware behaves as if a STOP condition has been received and it switches to 'not addressed' Slave Receiver Mode. The STO flag is cleared by hardware automatically.
5	STA	Start Flag. STA = 1: I²C-bus enters master mode, checks the bus and generates a START condition if the bus is free. If the bus is not free, it waits for a STOP condition (which will free the bus) and generates a START condition after a delay of a half clock period of the internal clock generator. When the I²C interface is already in master mode and some data is transmitted or received, it transmits a repeated START condition. STA may be set at any time, it may also be set when the I²C interface is in an addressed slave mode. STA = 0: no START condition or repeated START condition will be generated.
6	I2EN	I²C Interface Enable. When set, enables the I²C interface. When clear, the I²C function is disabled.
7	-	reserved

11.4 I²C Status register

This is a read-only register. It contains the status code of the I²C interface. The least three bits are always 0. There are 26 possible status codes. When the code is F8H, there is no relevant information available and SI bit is not set. All other 25 status codes correspond to defined I²C states. When any of these states entered, the SI bit will be set. Refer to Table 68 to Table 71 for details.

Table 63: I²C Status register (I2STAT - address D9h) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	STA.4	STA.3	STA.2	STA.1	STA.0	0	0	0
Reset	0	0	0	0	0	0	0	0

Table 64: I²C Status register (I2STAT - address D9h) bit description

Bit	Symbol	Description
0:2	-	Reserved, are always set to 0.
3:7	STA[0:4]	I²C Status code.

11.5 I²C SCL duty cycle registers I2SCLH and I2SCLL

When the internal SCL generator is selected for the I²C interface by setting CRSEL = 0 in the I2CON register, the user must set values for registers I2SCLL and I2SCLH to select the data rate. I2SCLH defines the number of PCLK cycles for SCL = high, I2SCLL defines the number of PCLK cycles for SCL = low. The frequency is determined by the following formula:

$$\text{Bit Frequency} = f_{PCLK} / (2*(I2SCLH + I2SCLL))$$

Where f_{PCLK} is the frequency of PCLK.

The values for I2SCLL and I2SCLH do not have to be the same; the user can give different duty cycles for SCL by setting these two registers. However, the value of the register must ensure that the data rate is in the I²C data rate range of 0 to 400 kHz. Thus the values of I2SCLL and I2SCLH have some restrictions and values for both registers greater than three PCLKs are recommended.

Table 65: I²C clock rates selection

		Bit data rate (Kbit/sec) at f_{osc}				
I2SCLL+ I2SCLH	**CRSEL**	**7.373 MHz**	**3.6865 MHz**	**1.8433 MHz**	**12 MHz**	**6 MHz**
6	0	-	307	154	-	-
7	0	-	263	132	-	-
8	0	-	230	115	-	375
9	0	-	205	102	-	333
10	0	369	184	92	-	300
15	0	246	123	61	400	200
25	0	147	74	37	240	120
30	0	123	61	31	200	100
50	0	74	37	18	120	60
60	0	61	31	15	100	50
100	0	37	18	9	60	30
150	0	25	12	6	40	20
200	0	18	9	5	30	15
-	1	3.6 Kbps to 922 Kbps Timer 1 in mode 2	1.8 Kbps to 461 Kbps Timer 1 in mode 2	0.9 Kbps to 230 Kbps Timer 1 in mode 2	5.86 Kbps to 1500 Kbps Timer 1 in mode 2	2.93 Kbps to 750 Kbps Timer 1 in mode 2

11.6 I²C operation modes

11.6.1 Master Transmitter mode

In this mode data is transmitted from master to slave. Before the Master Transmitter mode can be entered, I2CON must be initialized as follows:

Table 66: I²C Control register (I2CON - address D8h)

Bit	7	6	5	4	3	2	1	0
	-	I2EN	STA	STO	SI	AA	-	CRSEL
value	-	1	0	0	0	x	-	bit rate

CRSEL defines the bit rate. I2EN must be set to 1 to enable the I²C function. If the AA bit is 0, it will not acknowledge its own slave address or the general call address in the event of another device becoming master of the bus and it can not enter slave mode. STA, STO, and SI bits must be cleared to 0.

The first byte transmitted contains the slave address of the receiving device (7 bits) and the data direction bit. In this case, the data direction bit (R/W) will be logic 0 indicating a write. Data is transmitted 8 bits at a time. After each byte is transmitted, an acknowledge bit is received. START and STOP conditions are output to indicate the beginning and the end of a serial transfer.

The I^2C-bus will enter Master Transmitter Mode by setting the STA bit. The I^2C logic will send the START condition as soon as the bus is free. After the START condition is transmitted, the SI bit is set, and the status code in I2STAT should be 08h. This status code must be used to vector to an interrupt service routine where the user should load the slave address to I2DAT (Data Register) and data direction bit (SLA+W). The SI bit must be cleared before the data transfer can continue.

When the slave address and R/W bit have been transmitted and an acknowledgment bit has been received, the SI bit is set again, and the possible status codes are 18h, 20h, or 38h for the master mode or 68h, 78h, or 0B0h if the slave mode was enabled (setting AA = Logic 1). The appropriate action to be taken for each of these status codes is shown in Table 68.

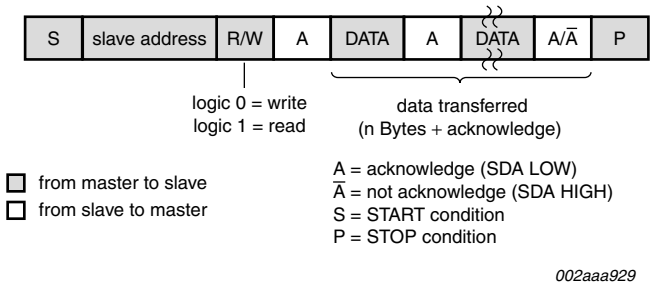


Fig 31. Format in the Master Transmitter mode.

11.6.2 Master Receiver mode

In the Master Receiver Mode, data is received from a slave transmitter. The transfer started in the same manner as in the Master Transmitter Mode. When the START condition has been transmitted, the interrupt service routine must load the slave address and the data direction bit to I^2C Data Register (I2DAT). The SI bit must be cleared before the data transfer can continue.

When the slave address and data direction bit have been transmitted and an acknowledge bit has been received, the SI bit is set, and the Status Register will show the status code. For master mode, the possible status codes are 40H, 48H, or 38H. For slave mode, the possible status codes are 68H, 78H, or B0H. Refer to Table 70 for details.

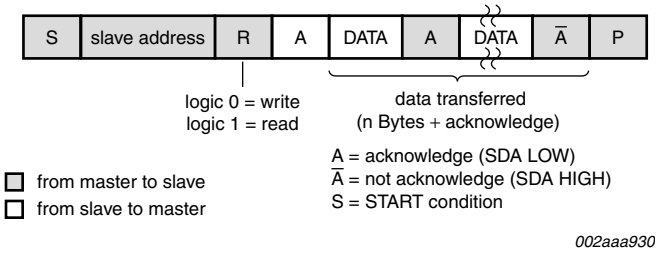


Fig 32. Format of Master Receiver mode.

After a repeated START condition, I²C-bus may switch to the Master Transmitter Mode.

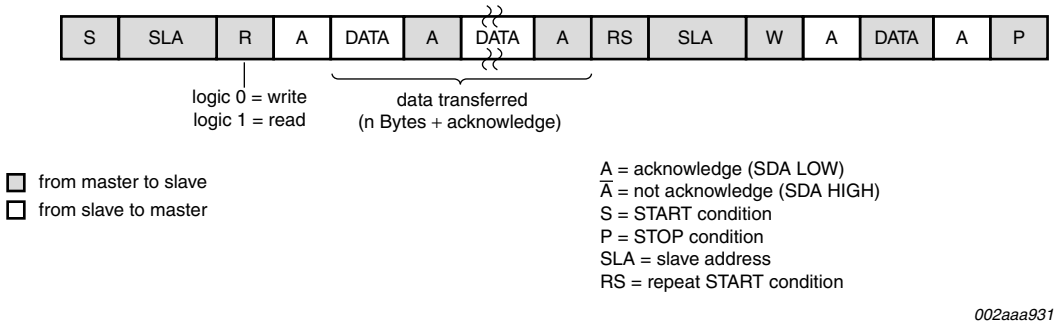


Fig 33. A Master Receiver switches to Master Transmitter after sending Repeated Start.

11.6.3 Slave Receiver mode

In the Slave Receiver Mode, data bytes are received from a master transmitter. To initialize the Slave Receiver Mode, the user should write the slave address to the Slave Address Register (I2ADR) and the I²C Control Register (I2CON) should be configured as follows:

Table 67: I²C Control register (I2CON - address D8h)

Bit	7	6	5	4	3	2	1	0
	-	I2EN	STA	STO	SI	AA	-	CRSEL
value	-	1	0	0	0	1	-	-

CRSEL is not used for slave mode. I2EN must be set = 1 to enable I²C function. AA bit must be set = 1 to acknowledge its own slave address or the general call address. STA, STO and SI are cleared to 0.

After I2ADR and I2CON are initialized, the interface waits until it is addressed by its own address or general address followed by the data direction bit which is 0(W). If the direction bit is 1(R), it will enter Slave Transmitter Mode. After the address and the direction bit have been received, the SI bit is set and a valid status code can be read from the Status Register(I2STAT). Refer to Table 71 for the status codes and actions.

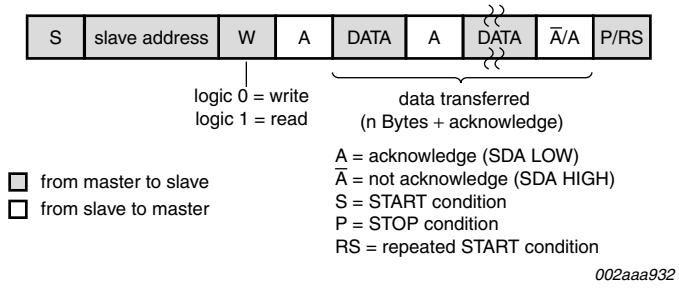


Fig 34. Format of Slave Receiver mode.

11.6.4 Slave Transmitter mode

The first byte is received and handled as in the Slave Receiver Mode. However, in this mode, the direction bit will indicate that the transfer direction is reversed. Serial data is transmitted via P1.3/SDA while the serial clock is input through P1.2/SCL. START and STOP conditions are recognized as the beginning and end of a serial transfer. In a given application, the I^2C-bus may operate as a master and as a slave. In the slave mode, the I^2C hardware looks for its own slave address and the general call address. If one of these addresses is detected, an interrupt is requested. When the microcontrollers wishes to become the bus master, the hardware waits until the bus is free before the master mode is entered so that a possible slave action is not interrupted. If bus arbitration is lost in the master mode, the I^2C-bus switches to the slave mode immediately and can detect its own slave address in the same serial transfer.

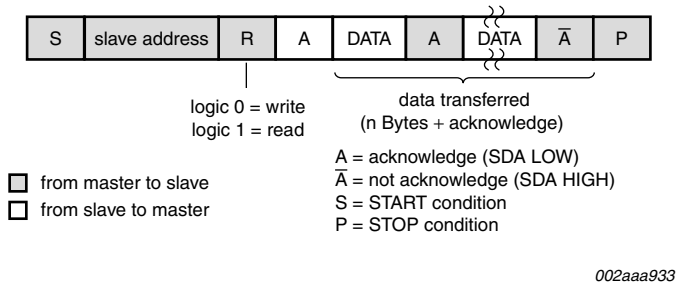


Fig 35. Format of Slave Transmitter mode.

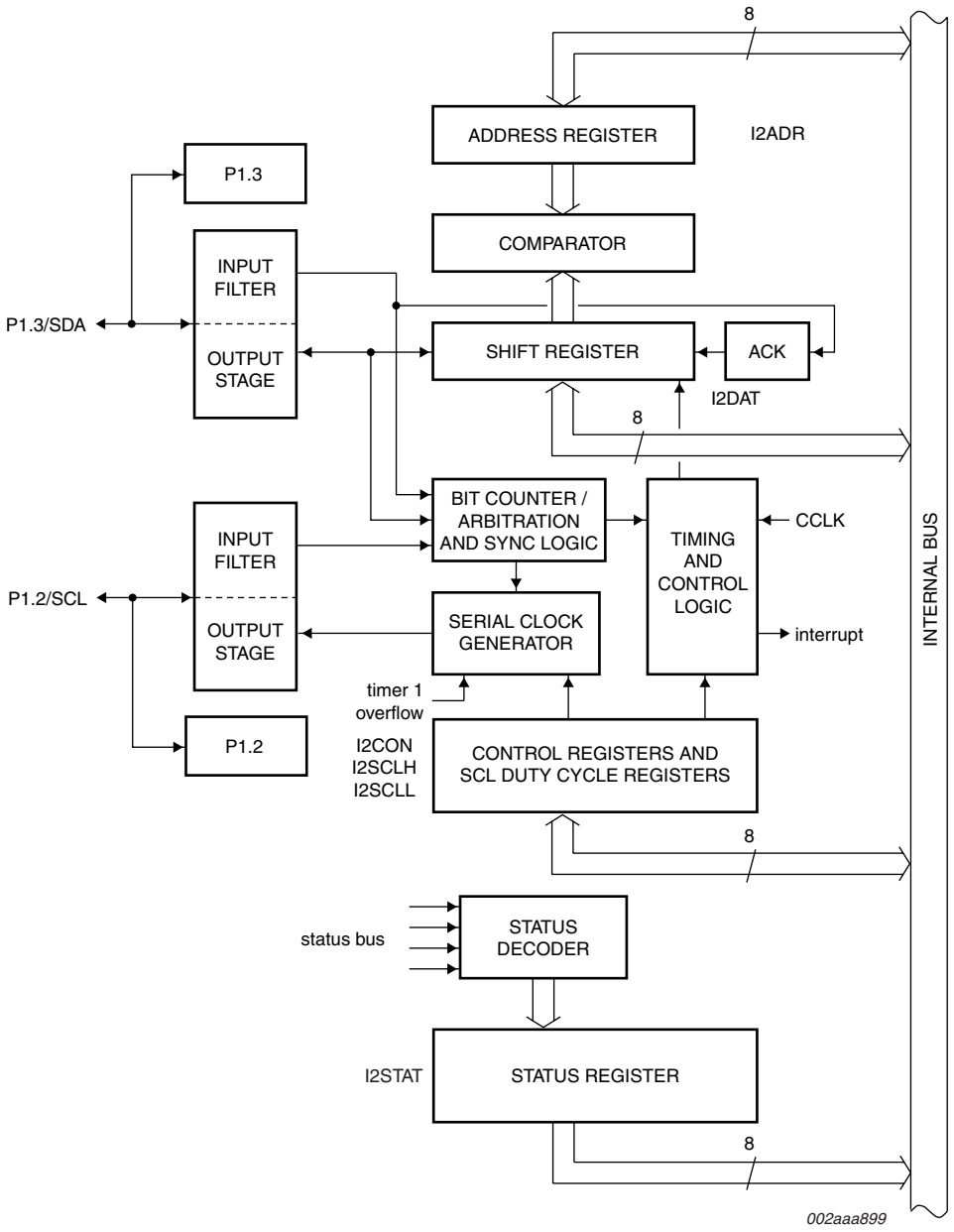


Fig 36. I²C serial interface block diagram.

Table 68: Master Transmitter mode

Status code (I2STAT)	Status of the I²C hardware	Application software response					Next action taken by I²C hardware
		to/from I2DAT	to I2CON				
			STA	STO	SI	AA	
08H	A START condition has been transmitted	Load SLA+W	x	0	0	x	SLA+W will be transmitted; ACK bit will be received
10H	A repeat START condition has been transmitted	Load SLA+W or Load SLA+R	x	0	0	x	As above; SLA+W will be transmitted; I²C-bus switches to Master Receiver Mode
18h	SLA+W has been transmitted; ACK has been received	Load data byte or	0	0	0	x	Data byte will be transmitted; ACK bit will be received
		no I2DAT action or	1	0	0	x	Repeated START will be transmitted;
		no I2DAT action or	0	1	0	x	STOP condition will be transmitted; STO flag will be reset
		no I2DAT action	1	1	0	x	STOP condition followed by a START condition will be transmitted; STO flag will be reset.
20h	SLA+W has been transmitted; NOT-ACK has been received	Load data byte or	0	0	0	x	Data byte will be transmitted; ACK bit will be received
		no I2DAT action or	1	0	0	x	Repeated START will be transmitted;
		no I2DAT action or	0	1	0	x	STOP condition will be transmitted; STO flag will be reset
		no I2DAT action	1	1	0	x	STOP condition followed by a START condition will be transmitted; STO flag will be reset
28h	Data byte in I2DAT has been transmitted; ACK has been received	Load data byte or	0	0	0	x	Data byte will be transmitted; ACK bit will be received
		no I2DAT action or	1	0	0	x	Repeated START will be transmitted;
		no I2DAT action or	0	1	0	x	STOP condition will be transmitted; STO flag will be reset
		no I2DAT action	1	1	0	x	STOP condition followed by a START condition will be transmitted; STO flag will be reset

Table 68: Master Transmitter mode *…continued*

Status code (I2STAT)	Status of the I²C hardware	Application software response					Next action taken by I²C hardware
		to/from I2DAT	to I2CON				
			STA	STO	SI	AA	
30h	Data byte in I2DAT has been transmitted, NOT ACK has been received	Load data byte or	0	0	0	x	Data byte will be transmitted; ACK bit will be received
		no I2DAT action or	1	0	0	x	Repeated START will be transmitted;
		no I2DAT action or	0	1	0	x	STOP condition will be transmitted; STO flag will be reset
		no I2DAT action	1	1	0	x	STOP condition followed by a START condition will be transmitted. STO flag will be reset.
38H	Arbitration lost in SLA+R/W or data bytes	No I2DAT action or	0	0	0	x	I²C-bus will be released; not addressed slave will be entered
		No I2DAT action	1	0	0	x	A START condition will be transmitted when the bus becomes free.

Table 69: Master Receiver mode

Status code (I2STAT)	Status of the I²C hardware	Application software response					Next action taken by I²C hardware
		to/from I2DAT	to I2CON				
			STA	STO	SI	STA	
08H	A START condition has been transmitted	Load SLA+R	x	0	0	x	SLA+R will be transmitted; ACK bit will be received
10H	A repeat START condition has been transmitted	Load SLA+R or	x	0	0	x	As above
		Load SLA+W					SLA+W will be transmitted; I²C-bus will be switched to Master Transmitter Mode
38H	Arbitration lost in NOT ACK bit	no I2DAT action or	0	0	0	x	I²C-bus will be released; it will enter a slave mode
		no I2DAT 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 I2DAT action or	0	0	0	0	Data byte will be received; NOT ACK bit will be returned
		no I2DAT action or	0	0	0	1	Data byte will be received; ACK bit will be returned
48h	SLA+R has been transmitted; NOT ACK has been received	No I2DAT action or	1	0	0	x	Repeated START will be transmitted
		no I2DAT action or	0	1	0	x	STOP condition will be transmitted; STO flag will be reset
		no I2DAT action or	1	1	0	x	STOP condition followed by a START condition will be transmitted; STO flag will be reset

Table 69: Master Receiver mode *…continued*

Status code (I2STAT)	Status of the I²C hardware	Application software response					Next action taken by I²C hardware
		to/from I2DAT	to I2CON				
			STA	STO	SI	STA	
50h	Data byte has been received; ACK has been returned	Read data byte	0	0	0	0	Data byte will be received; NOT ACK bit will be returned
		read data byte	0	0	0	1	Data byte will be received; ACK bit will be returned
58h	Data byte has been received; NACK has been returned	Read data byte or	1	0	0	x	Repeated START will be transmitted;
		read data byte or	0	1	0	x	STOP condition will be transmitted; STO flag will be reset
		read data byte	1	1	0	x	STOP condition followed by a START condition will be transmitted; STO flag will be reset

Table 70: Slave Receiver mode

Status code (I2STAT)	Status of the I²C hardware	Application software response					Next action taken by I²C hardware
		to/from I2DAT	to I2CON				
			STA	STO	SI	AA	
60H	Own SLA+W has been received; ACK has been received	no I2DAT action or	x	0	0	0	Data byte will be received and NOT ACK will be returned
		no I2DAT action	x	0	0	1	Data byte will be received and ACK will be returned
68H	Arbitration lost in SLA+R/Was master; Own SLA+W has been received, ACK returned	No I2DAT action or	x	0	0	0	Data byte will be received and NOT ACK will be returned
		no I2DAT action	x	0	0	1	Data byte will be received and ACK will be returned
70H	General call address(00H) has been received, ACK has been returned	No I2DAT action or	x	0	0	0	Data byte will be received and NOT ACK will be returned
		no I2DAT 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 bit has been returned	no I2DAT action or	x	0	0	0	Data byte will be received and NOT ACK will be returned
		no I2DAT action	x	0	0	1	Data byte will be received and ACK will be returned
80H	Previously addressed with own SLA address; 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; ACK bit will be returned

Table 70: Slave Receiver mode *…continued*

Status code (I2STAT)	Status of the I²C hardware	Application software response					Next action taken by I²C hardware
		to/from I2DAT	to I2CON				
			STA	STO	SI	AA	
88H	Previously addressed with own SLA address; Data has been received; NACK has been returned	Read data byte or	0	0	0	0	Switched to not addressed SLA mode; no recognition of own SLA or general address
		read data byte or	0	0	0	1	Switched to not addressed SLA mode; Own SLA will be recognized; general call address will be recognized if I2ADR.0 = 1
		read data byte or	1	0	0	0	Switched to not addressed SLA mode; no recognition of own SLA or General call address. A START condition will be transmitted when the bus becomes free
		read data byte	1	0	0	1	Switched to not addressed SLA mode; Own slave address will be recognized; General call address will be recognized if I2ADR.0 = 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
98H	Previously addressed with General call; Data has been received; NACK has been returned	Read data byte	0	0	0	0	Switched to not addressed SLA mode; no recognition of own SLA or General call address
		read data byte	0	0	0	1	Switched to not addressed SLA mode; Own slave address will be recognized; General call address will be recognized if I2ADR.0 = 1.
		read data byte	1	0	0	0	Switched to not addressed SLA mode; no recognition of own SLA or General call address. A START condition will be transmitted when the bus becomes free.
		read data byte	1	0	0	1	Switched to not addressed SLA mode; Own slave address will be recognized; General call address will be recognized if I2ADR.0 = 1. A START condition will be transmitted when the bus becomes free.

Table 70: Slave Receiver mode *…continued*

Status code (I2STAT)	Status of the I²C hardware	Application software response					Next action taken by I²C hardware
		to/from I2DAT	to I2CON				
			STA	STO	SI	AA	
A0H	A STOP condition or repeated START condition has been received while still addressed as SLA/REC or SLA/TRX	No I2DAT action	0	0	0	0	Switched to not addressed SLA mode; no recognition of own SLA or General call address
		no I2DAT action	0	0	0	1	Switched to not addressed SLA mode; Own slave address will be recognized; General call address will be recognized if I2ADR.0 = 1.
		no I2DAT action	1	0	0	0	Switched to not addressed SLA mode; no recognition of own SLA or General call address. A START condition will be transmitted when the bus becomes free.
		no I2DAT action	1	0	0	1	Switched to not addressed SLA mode; Own slave address will be recognized; General call address will be recognized if I2ADR.0 = 1. A START condition will be transmitted when the bus becomes free.

Table 71: Slave Transmitter mode

Status code (I2STAT)	Status of the I²C hardware	Application software response					Next action taken by I²C hardware
		to/from I2DAT	to I2CON				
			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 ACK bit will be received
		load data byte	x	0	0	1	Data byte will be transmitted; 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 ACK bit will be received
		load data byte	x	0	0	1	Data byte will be transmitted; ACK bit will be received
B8H	Data byte in I2DAT has been transmitted; ACK has been received	Load data byte or	x	0	0	0	Last data byte will be transmitted and ACK bit will be received
		load data byte	x	0	0	1	Data byte will be transmitted; ACK will be received

Table 71: Slave Transmitter mode *…continued*

Status code (I2STAT)	Status of the I²C hardware	Application software response					Next action taken by I²C hardware
		to/from I2DAT	to I2CON				
			STA	STO	SI	AA	
C0H	Data byte in I2DAT has been transmitted; NACK has been received	No I2DAT action or	0	0	0	0	Switched to not addressed SLA mode; no recognition of own SLA or General call address.
		no I2DAT action or	0	0	0	1	Switched to not addressed SLA mode; Own slave address will be recognized; General call address will be recognized if I2ADR.0 = 1.
		no I2DAT action or	1	0	0	0	Switched to not addressed SLA mode; no recognition of own SLA or General call address. A START condition will be transmitted when the bus becomes free.
		no I2DAT action	1	0	0	1	Switched to not addressed SLA mode; Own slave address will be recognized; General call address will be recognized if I2ADR.0 = 1. A START condition will be transmitted when the bus becomes free.
C8H	Last data byte in I2DAT has been transmitted (AA = 0); ACK has been received	No I2DAT action or	0	0	0	0	Switched to not addressed SLA mode; no recognition of own SLA or General call address.
		no I2DAT action or	0	0	0	1	Switched to not addressed SLA mode; Own slave address will be recognized; General call address will be recognized if I2ADR.0 = 1.
		no I2DAT action or	1	0	0	0	Switched to not addressed SLA mode; no recognition of own SLA or General call address. A START condition will be transmitted when the bus becomes free.
		no I2DAT action	1	0	0	1	Switched to not addressed SLA mode; Own slave address will be recognized; General call address will be recognized if I2ADR.0 = 1. A START condition will be transmitted when the bus becomes free.

12. Serial Peripheral Interface (SPI)

The P89LPC932A1 provides another high-speed serial communication interface, the SPI interface. SPI is a full-duplex, high-speed, synchronous communication bus with two operation modes: Master mode and Slave mode. Up to 3 Mbit/s can be supported in either Master or Slave mode. It has a Transfer Completion Flag and Write Collision Flag Protection.

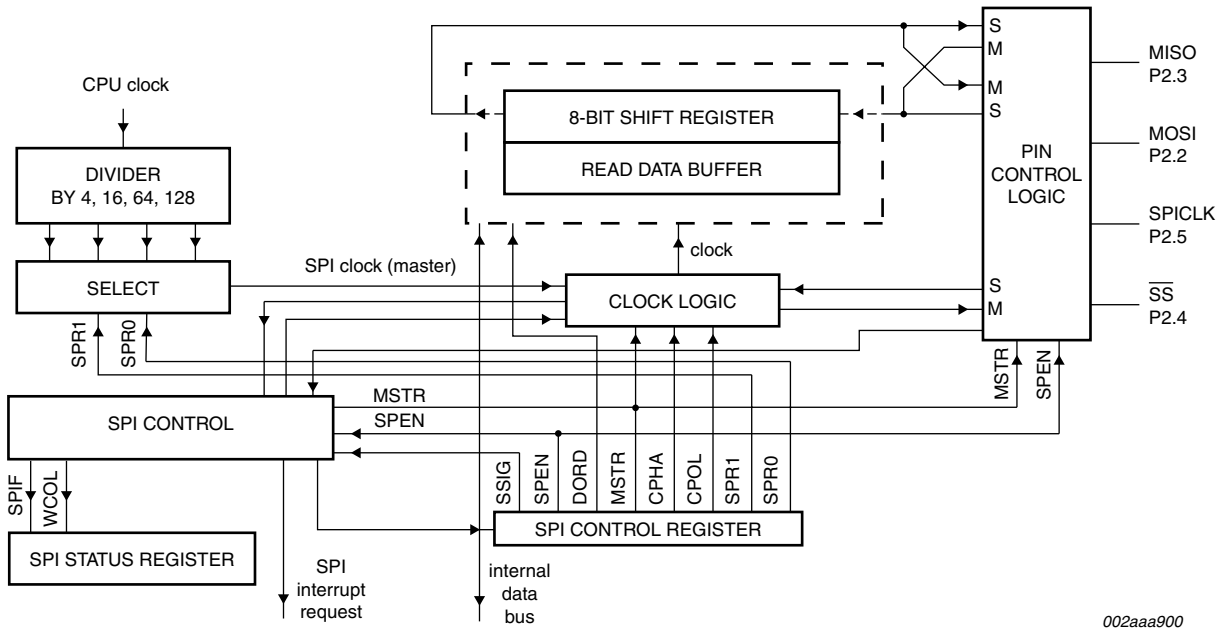


Fig 37. SPI block diagram.

The SPI interface has four pins: SPICLK, MOSI, MISO and $\overline{SS}$:

- SPICLK, MOSI and MISO are typically tied together between two or more SPI devices. Data flows from master to slave on the MOSI (Master Out Slave In) pin and flows from slave to master on the MISO (Master In Slave Out) pin. The SPICLK signal is output in the master mode and is input in the slave mode. If the SPI system is disabled, i.e. SPEN (SPCTL.6) = 0 (reset value), these pins are configured for port functions.
- $\overline{SS}$ is the optional slave select pin. In a typical configuration, an SPI master asserts one of its port pins to select one SPI device as the current slave. An SPI slave device uses its $\overline{SS}$ pin to determine whether it is selected. The $\overline{SS}$ is ignored if any of the following conditions are true:
  - If the SPI system is disabled, i.e. SPEN (SPCTL.6) = 0 (reset value)
  - If the SPI is configured as a master, i.e., MSTR (SPCTL.4) = 1, and P2.4 is configured as an output (via the P2M1.4 and P2M2.4 SFR bits);
  - If the $\overline{SS}$ pin is ignored, i.e. SSIG (SPCTL.7) bit = 1, this pin is configured for port functions.

Note that even if the SPI is configured as a master (MSTR = 1), it can still be converted to a slave by driving the $\overline{SS}$ pin low (if P2.4 is configured as input and SSIG = 0). Should this happen, the SPIF bit (SPSTAT.7) will be set (see Section 12.4 “Mode change on SS”)

Typical connections are shown in Figure 38 to Figure 40.

Table 72: SPI Control register (SPCTL - address E2h) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	SSIG	SPEN	DORD	MSTR	CPOL	CPHA	SPR1	SPR0
Reset	0	0	0	0	0	1	0	0

Table 73: SPI Control register (SPCTL - address E2h) bit description

Bit	Symbol	Description
0	SPR0	SPI Clock Rate Select
1	SPR1	SPR1, SPR0: **00** — $CCLK/4$ **01** — $CCLK/16$ **10** — $CCLK/64$ **11** — $CCLK/128$
2	CPHA	SPI Clock PHAse select (see Figure 41 to Figure 44): **1** — Data is driven on the leading edge of SPICLK, and is sampled on the trailing edge. **0** — Data is driven when $\overline{SS}$ is low (SSIG = 0) and changes on the trailing edge of SPICLK, and is sampled on the leading edge. (Note: If SSIG = 1, the operation is not defined.
3	CPOL	SPI Clock POLarity (see Figure 41 to Figure 44): **1** — SPICLK is high when idle. The leading edge of SPICLK is the falling edge and the trailing edge is the rising edge. **0** — SPICLK is low when idle. The leading edge of SPICLK is the rising edge and the trailing edge is the falling edge.
4	MSTR	Master/Slave mode Select (see Table 77).
5	DORD	SPI Data ORDer. **1** — The LSB of the data word is transmitted first. **0** — The MSB of the data word is transmitted first.
6	SPEN	SPI Enable. **1** — The SPI is enabled. **0** — The SPI is disabled and all SPI pins will be port pins.
7	SSIG	$\overline{SS}$ IGnore. **1** — MSTR (bit 4) decides whether the device is a master or slave. **0** — The $\overline{SS}$ pin decides whether the device is master or slave. The $\overline{SS}$ pin can be used as a port pin (see Table 77).

Table 74: SPI Status register (SPSTAT - address E1h) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	SPIF	WCOL	-	-	-	-	-	-
Reset	0	0	x	x	x	x	x	x

Table 75: SPI Status register (SPSTAT - address E1h) bit description

Bit	Symbol	Description
0:5	-	reserved
6	WCOL	SPI Write Collision Flag. The WCOL bit is set if the SPI data register, SPDAT, is written during a data transfer (see Section 12.5 “Write collision”). The WCOL flag is cleared in software by writing a logic 1 to this bit.
7	SPIF	SPI Transfer Completion Flag. When a serial transfer finishes, the SPIF bit is set and an interrupt is generated if both the ESPI (IEN1.3) bit and the EA bit are set. If $\overline{SS}$ is an input and is driven low when SPI is in master mode, and SSIG = 0, this bit will also be set (see Section 12.4 “Mode change on SS”). The SPIF flag is cleared in software by writing a logic 1 to this bit.

Table 76: SPI Data register (SPDAT - address E3h) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	MSB							LSB
Reset	0	0	0	0	0	0	0	0

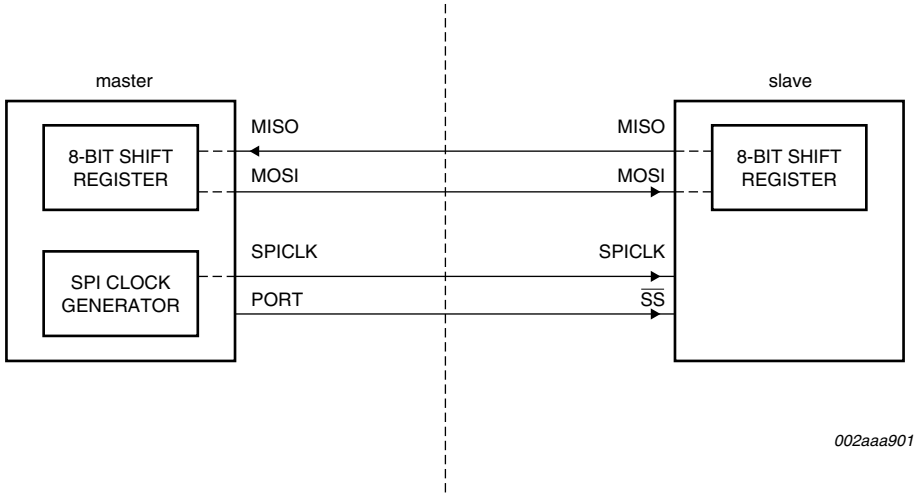


Fig 38. SPI single master single slave configuration.

In Figure 38, SSIG (SPCTL.7) for the slave is logic 0, and $\overline{SS}$ is used to select the slave. The SPI master can use any port pin (including P2.4/$\overline{SS}$) to drive the $\overline{SS}$ pin.

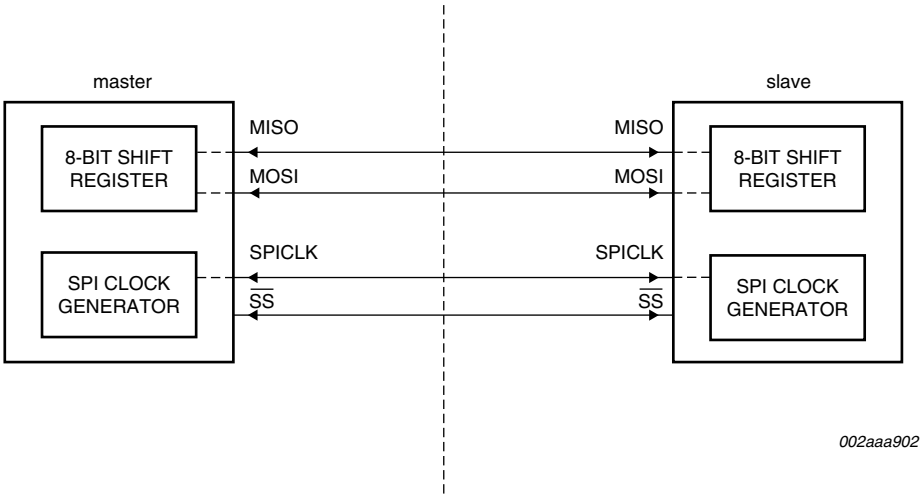


Fig 39. SPI dual device configuration, where either can be a master or a slave.

Figure 39 shows a case where two devices are connected to each other and either device can be a master or a slave. When no SPI operation is occurring, both can be configured as masters (MSTR = 1) with SSIG cleared to 0 and P2.4 ($\overline{SS}$) configured in quasi-bidirectional mode. When a device initiates a transfer, it can configure P2.4 as an output and drive it low, forcing a mode change in the other device (see Section 12.4 “Mode change on SS”) to slave.

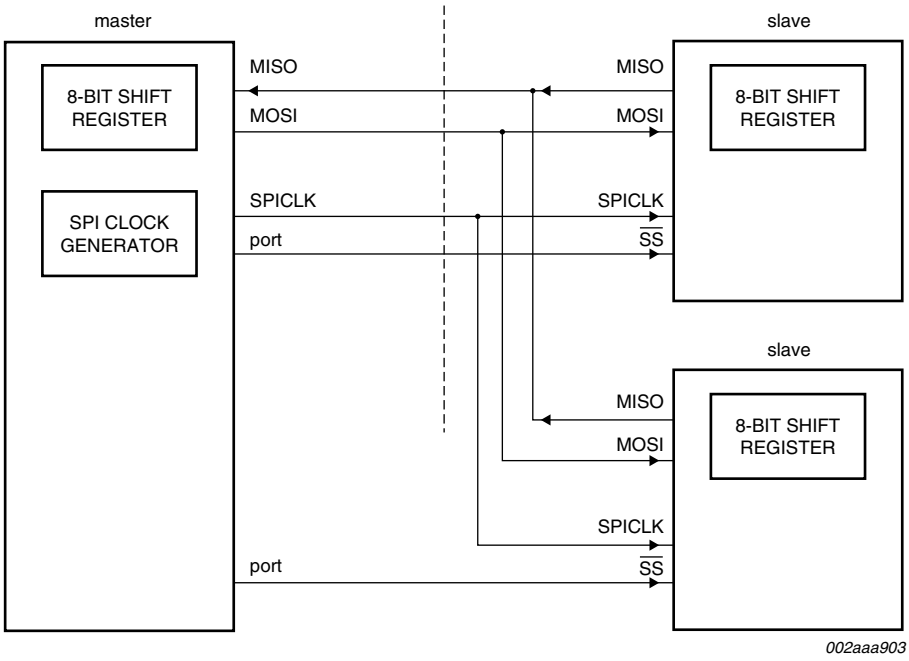


Fig 40. SPI single master multiple slaves configuration.

In Figure 40, SSIG (SPCTL.7) bits for the slaves are logic 0, and the slaves are selected by the corresponding $\overline{SS}$ signals. The SPI master can use any port pin (including P2.4/$\overline{SS}$) to drive the $\overline{SS}$ pins.

12.1 Configuring the SPI

Table 77 shows configuration for the master/slave modes as well as usages and directions for the modes.

Table 77: SPI master and slave selection

SPEN	SSIG	$\overline{SS}$ Pin	MSTR	Master or Slave Mode	MISO	MOSI	SPICLK	Remarks
0	x	P2.4[1]	x	SPI Disabled	P2.3[1]	P2.2[1]	P2.5[1]	SPI disabled. P2.2, P2.3, P2.4, P2.5 are used as port pins.
1	0	0	0	Slave	output	input	input	Selected as slave.
1	0	1	0	Slave	Hi-Z	input	input	Not selected. MISO is high-impedance to avoid bus contention.
1	0	0	1 (-> 0)[2]	Slave	output	input	input	P2.4/$\overline{SS}$ is configured as an input or quasi-bidirectional pin. SSIG is 0. Selected externally as slave if $\overline{SS}$ is selected and is driven low. The MSTR bit will be cleared to logic 0 when $\overline{SS}$ becomes low.

Table 77: SPI master and slave selection *...continued*

SPEN	SSIG	$\overline{SS}$ Pin	MSTR	Master or Slave Mode	MISO	MOSI	SPICLK	Remarks
1	0	1	1	Master (idle)	input	Hi-Z	Hi-Z	MOSI and SPICLK are at high-impedance to avoid bus contention when the MAster is idle. The application must pull-up or pull-down SPICLK (depending on CPOL - SPCTL.3) to avoid a floating SPICLK.
				Master (active)		output	output	MOSI and SPICLK are push-pull when the Master is active.
1	1	P2.4[1]	0	Slave	output	input	input	
1	1	P2.4[1]	1	Master	input	output	output	

[1] Selected as a port function

[2] The MSTR bit changes to logic 0 automatically when $\overline{SS}$ becomes low in input mode and SSIG is logic 0.

12.2 Additional considerations for a slave

When CPHA equals zero, SSIG must be logic 0 and the $\overline{SS}$ pin must be negated and reasserted between each successive serial byte. If the SPDAT register is written while $\overline{SS}$ is active (low), a write collision error results. The operation is undefined if CPHA is logic 0 and SSIG is logic 1.

When CPHA equals one, SSIG may be set to logic 1. If SSIG = 0, the $\overline{SS}$ pin may remain active low between successive transfers (can be tied low at all times). This format is sometimes preferred in systems having a single fixed master and a single slave driving the MISO data line.

12.3 Additional considerations for a master

In SPI, transfers are always initiated by the master. If the SPI is enabled (SPEN = 1) and selected as master, writing to the SPI data register by the master starts the SPI clock generator and data transfer. The data will start to appear on MOSI about one half SPI bit-time to one SPI bit-time after data is written to SPDAT.

Note that the master can select a slave by driving the $\overline{SS}$ pin of the corresponding device low. Data written to the SPDAT register of the master is shifted out of the MOSI pin of the master to the MOSI pin of the slave, at the same time the data in SPDAT register in slave side is shifted out on MISO pin to the MISO pin of the master.

After shifting one byte, the SPI clock generator stops, setting the transfer completion flag (SPIF) and an interrupt will be created if the SPI interrupt is enabled (ESPI, or IEN1.3 = 1). The two shift registers in the master CPU and slave CPU can be considered as one distributed 16-bit circular shift register. When data is shifted from the master to the slave, data is also shifted in the opposite direction simultaneously. This means that during one shift cycle, data in the master and the slave are interchanged.

12.4 Mode change on $\overline{SS}$

If SPEN = 1, SSIG = 0 and MSTR = 1, the SPI is enabled in master mode. The $\overline{SS}$ pin can be configured as an input (P2M2.4, P2M1.4 = 00) or quasi-bidirectional (P2M2.4, P2M1.4 = 01). In this case, another master can drive this pin low to select this device as an SPI

slave and start sending data to it. To avoid bus contention, the SPI becomes a slave. As a result of the SPI becoming a slave, the MOSI and SPICLK pins are forced to be an input and MISO becomes an output.

The SPIF flag in SPSTAT is set, and if the SPI interrupt is enabled, an SPI interrupt will occur.

User software should always check the MSTR bit. If this bit is cleared by a slave select and the user wants to continue to use the SPI as a master, the user must set the MSTR bit again, otherwise it will stay in slave mode.

12.5 Write collision

The SPI is single buffered in the transmit direction and double buffered in the receive direction. New data for transmission can not be written to the shift register until the previous transaction is complete. The WCOL (SPSTAT.6) bit is set to indicate data collision when the data register is written during transmission. In this case, the data currently being transmitted will continue to be transmitted, but the new data, i.e., the one causing the collision, will be lost.

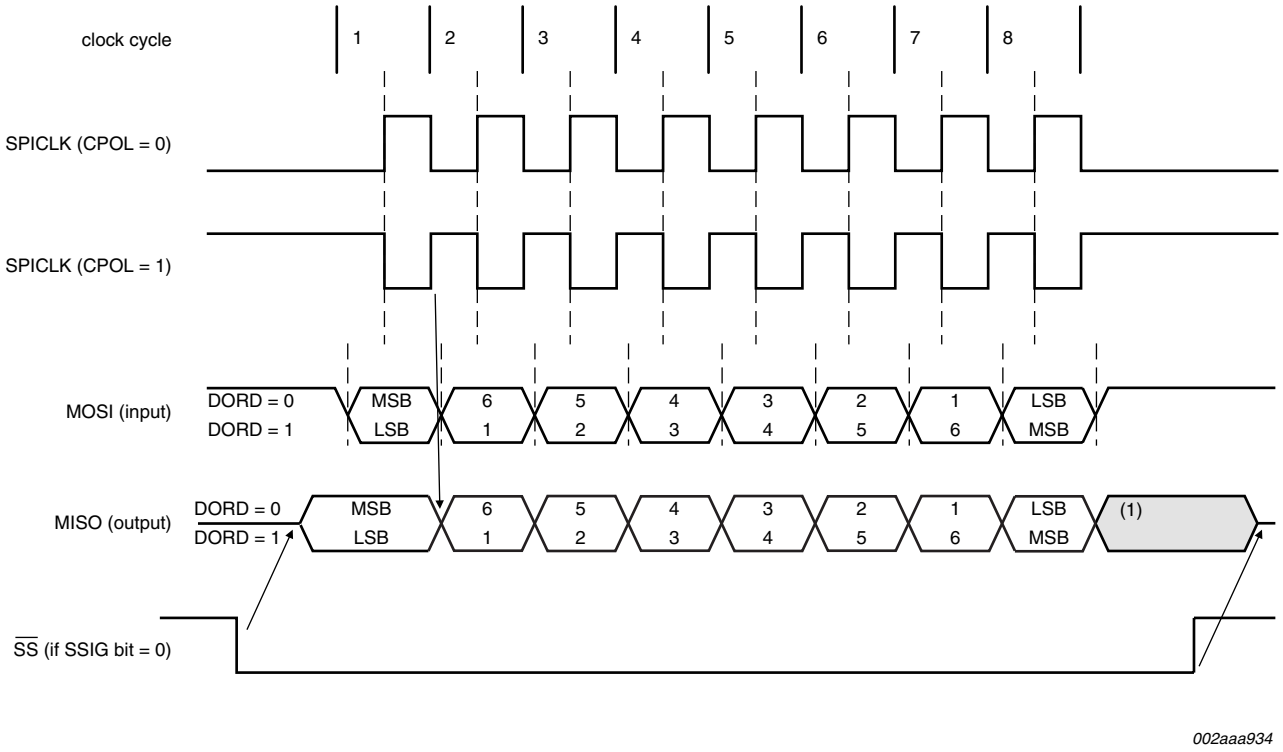
While write collision is detected for both a master or a slave, it is uncommon for a master because the master has full control of the transfer in progress. The slave, however, has no control over when the master will initiate a transfer and therefore collision can occur.

For receiving data, received data is transferred into a parallel read data buffer so that the shift register is free to accept a second character. However, the received character must be read from the Data Register before the next character has been completely shifted in. Otherwise. the previous data is lost.

WCOL can be cleared in software by writing a logic 1 to the bit.

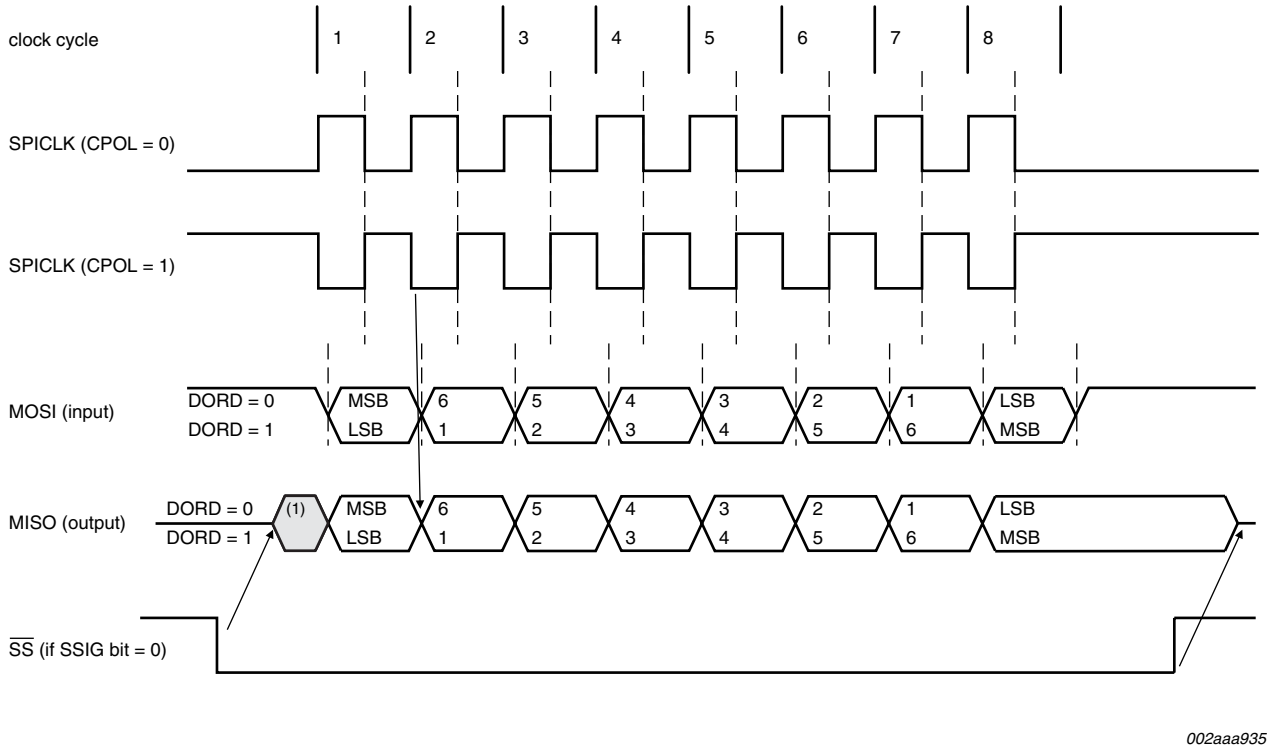
12.6 Data mode

Clock Phase Bit (CPHA) allows the user to set the edges for sampling and changing data. The Clock Polarity bit, CPOL, allows the user to set the clock polarity. Figure 41 - Figure 44 show the different settings of Clock Phase bit CPHA.



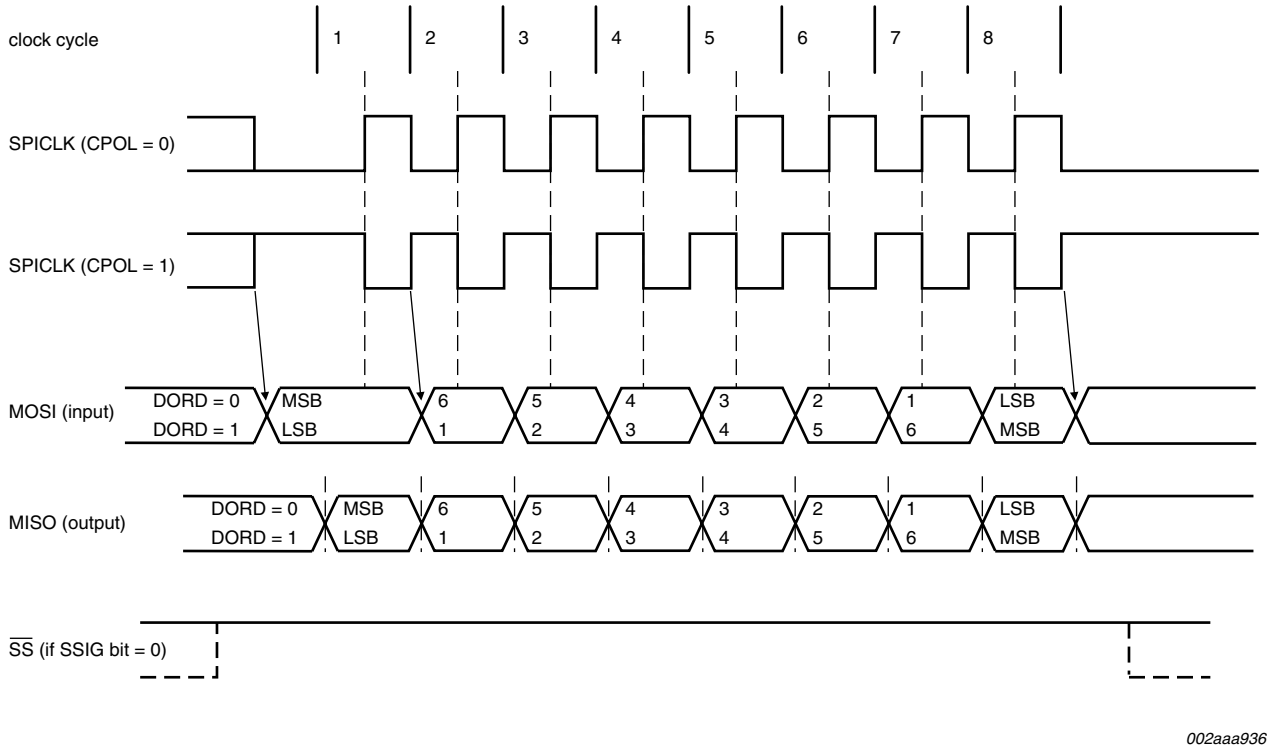
(1) Not defined

Fig 41. SPI slave transfer format with CPHA = 0.



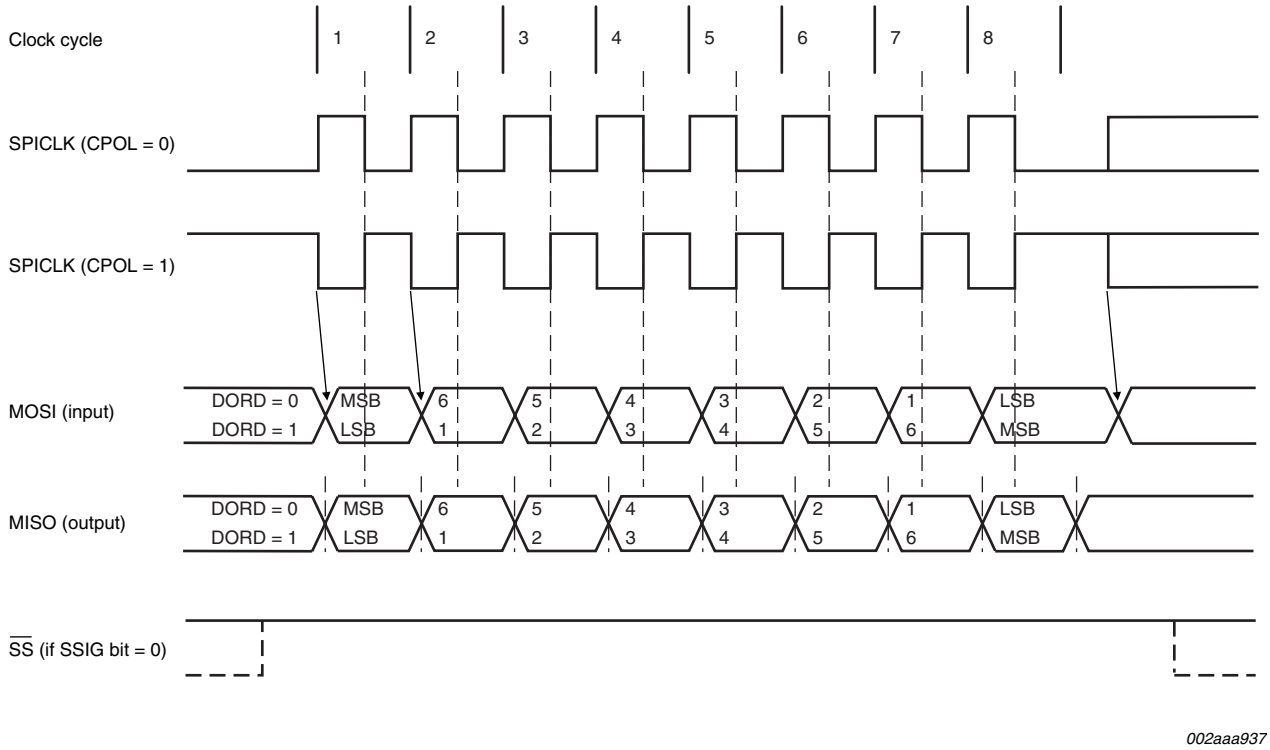
(1) Not defined

Fig 42. SPI slave transfer format with CPHA = 1.



(1) Not defined

Fig 43. SPI master transfer format with CPHA = 0.



(1) Not defined

Fig 44. SPI master transfer format with CPHA = 1.

12.7 SPI clock prescaler select

The SPI clock prescalar selection uses the SPR1-SPR0 bits in the SPCTL register (see Table 73).

13. Analog comparators

Two analog comparators are provided on the P89LPC932A1. Input and output options allow use of the comparators in a number of different configurations. Comparator operation is such that the output is a logic 1 (which may be read in a register and/or routed to a pin) when the positive input (one of two selectable pins) is greater than the negative input (selectable from a pin or an internal reference voltage). Otherwise the output is a zero. Each comparator may be configured to cause an interrupt when the output value changes.

13.1 Comparator configuration

Each comparator has a control register, CMP1 for comparator 1 and CMP2 for comparator 2. The control registers are identical and are shown in Table 79.

The overall connections to both comparators are shown in Figure 45. There are eight possible configurations for each comparator, as determined by the control bits in the corresponding CMPn register: CPn, CNn, and OEn. These configurations are shown in Figure 46.

When each comparator is first enabled, the comparator output and interrupt flag are not guaranteed to be stable for 10 microseconds. The corresponding comparator interrupt should not be enabled during that time, and the comparator interrupt flag must be cleared before the interrupt is enabled in order to prevent an immediate interrupt service.

Table 78: Comparator Control register (CMP1 - address ACh, CMP2 - address ADh) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	-	-	CEn	CPn	CNn	OEn	COn	CMFn
Reset	x	x	0	0	0	0	0	0

Table 79: Comparator Control register (CMP1 - address ACh, CMP2 - address ADh) bit description

Bit	Symbol	Description
0	CMFn	Comparator interrupt flag. This bit is set by hardware whenever the comparator output COn changes state. This bit will cause a hardware interrupt if enabled. Cleared by software.
1	COn	Comparator output, synchronized to the CPU clock to allow reading by software.
2	OEn	Output enable. When logic 1, the comparator output is connected to the CMPn pin if the comparator is enabled (CEn = 1). This output is asynchronous to the CPU clock.
3	CNn	Comparator negative input select. When logic 0, the comparator reference pin CMPREF is selected as the negative comparator input. When logic 1, the internal comparator reference, Vref, is selected as the negative comparator input.
4	CPn	Comparator positive input select. When logic 0, CINnA is selected as the positive comparator input. When logic 1, CINnB is selected as the positive comparator input.
5	CEn	Comparator enable. When set, the corresponding comparator function is enabled. Comparator output is stable 10 microseconds after CEn is set.
6:7	-	reserved

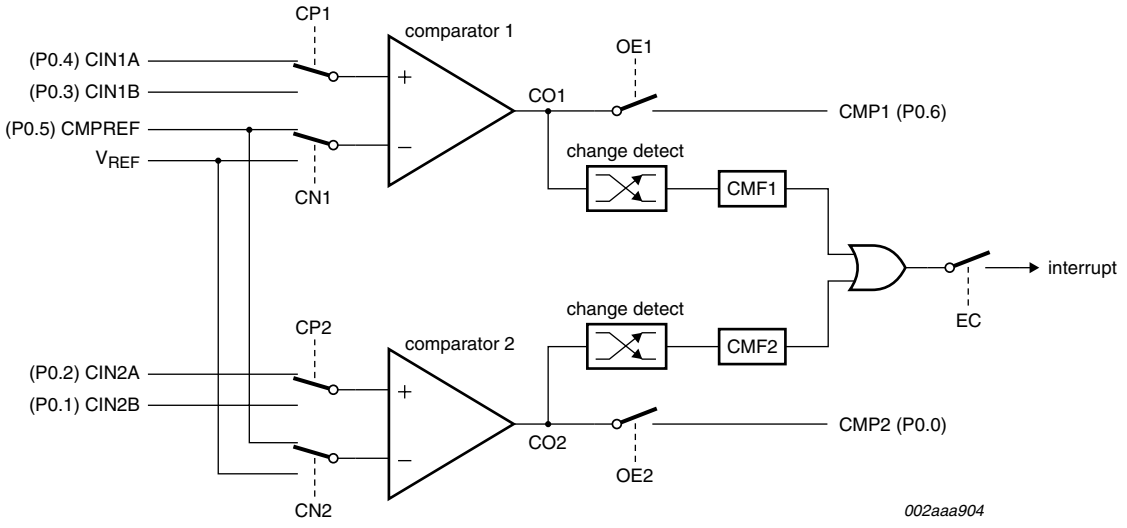


Fig 45. Comparator input and output connections.

13.2 Internal reference voltage

An internal reference voltage, Vref, may supply a default reference when a single comparator input pin is used. Please refer to the *P89LPC932A1 data sheet* for specifications

13.3 Comparator input pins

Comparator input and reference pins maybe be used as either digital I/O or as inputs to the comparator. When used as digital I/O these pins are 5 V tolerant. However, when selected as comparator input signals in CMPn lower voltage limits apply. Please refer to the *P89LPC932A1 data sheet* for specifications.

13.4 Comparator interrupt

Each comparator has an interrupt flag CMFn contained in its configuration register. This flag is set whenever the comparator output changes state. The flag may be polled by software or may be used to generate an interrupt. The two comparators use one common interrupt vector. The interrupt will be generated when the interrupt enable bit EC in the IEN1 register is set and the interrupt system is enabled via the EA bit in the IEN0 register. If both comparators enable interrupts, after entering the interrupt service routine, the user will need to read the flags to determine which comparator caused the interrupt.

When a comparator is disabled the comparator's output, COx, goes high. If the comparator output was low and then is disabled, the resulting transition of the comparator output from a low to high state will set the comparator flag, CMFx. This will cause an interrupt if the comparator interrupt is enabled. The user should therefore disable the comparator interrupt prior to disabling the comparator. Additionally, the user should clear the comparator flag, CMFx, after disabling the comparator.

13.5 Comparators and power reduction modes

Either or both comparators may remain enabled when Power-down mode or Idle mode is activated, but both comparators are disabled automatically in Total Power-down mode.

If a comparator interrupt is enabled (except in Total Power-down mode), a change of the comparator output state will generate an interrupt and wake-up the processor. If the comparator output to a pin is enabled, the pin should be configured in the push-pull mode in order to obtain fast switching times while in Power-down mode. The reason is that with the oscillator stopped, the temporary strong pull-up that normally occurs during switching on a quasi-bidirectional port pin does not take place.

Comparators consume power in Power-down mode and Idle mode, as well as in the normal operating mode. This should be taken into consideration when system power consumption is an issue. To minimize power consumption, the user can power-down the comparators by disabling the comparators and setting PCONA.5 to logic 1, or simply putting the device in Total Power-down mode.

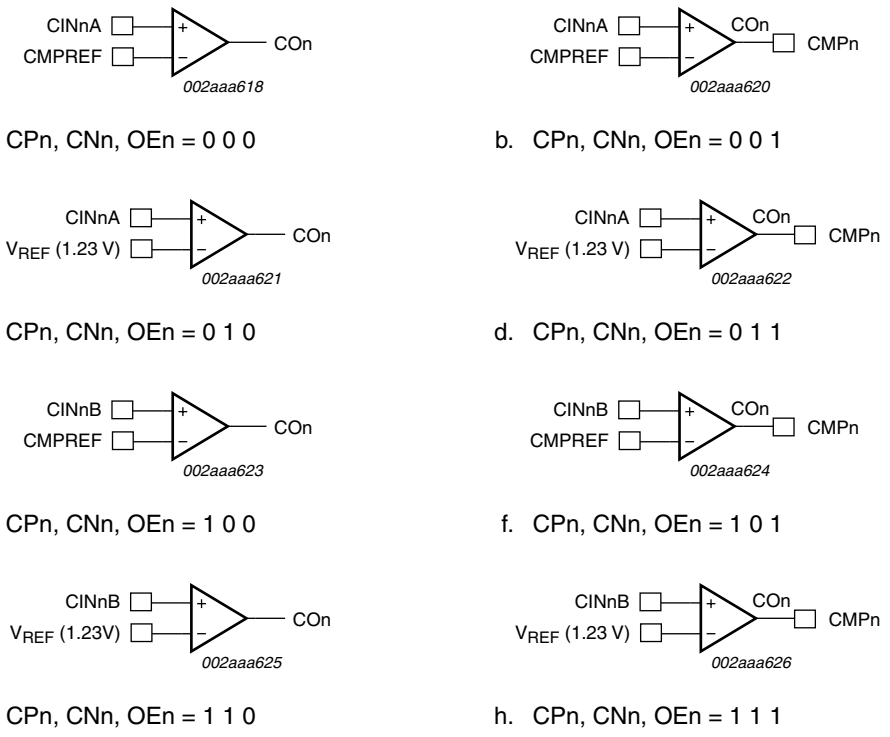


Fig 46. Comparator configurations.

13.6 Comparators configuration example

The code shown below is an example of initializing one comparator. Comparator 1 is configured to use the CIN1A and CMPREF inputs, outputs the comparator result to the CMP1 pin, and generates an interrupt when the comparator output changes.

```
CMPINIT:
 MOV PT0AD,#030h          ;Disable digital INPUTS on CIN1A, CMPREF.
 ANL P0M2,#0CFh           ;Disable digital OUTPUTS on pins that are used
 ORL P0M1,#030h           ;for analog functions: CIN1A, CMPREF.
 MOV CMP1,#024h           ;Turn on comparator 1 and set up for:
                          ;Positive input on CIN1A.
                          ;Negative input from CMPREF pin.
                          ;Output to CMP1 pin enabled.
 CALL delay10us           ;The comparator needs at least 10 microseconds before use.
 ANL CMP1,#0FEh           ;Clear comparator 1 interrupt flag.
 SETB EC                  ;Enable the comparator interrupt,
 SETB EA                  ;Enable the interrupt system (if needed).
 RET                      ;Return to caller.
```

The interrupt routine used for the comparator must clear the interrupt flag (CMF1 in this case) before returning

14. Keypad interrupt (KBI)

The Keypad Interrupt function is intended primarily to allow a single interrupt to be generated when Port 0 is equal to or not equal to a certain pattern. This function can be used for bus address recognition or keypad recognition. The user can configure the port via SFRs for different tasks.

There are three SFRs used for this function. The Keypad Interrupt Mask Register (KBMASK) is used to define which input pins connected to Port 0 are enabled to trigger the interrupt. The Keypad Pattern Register (KBPATN) is used to define a pattern that is compared to the value of Port 0. The Keypad Interrupt Flag (KBIF) in the Keypad Interrupt Control Register (KBCON) is set when the condition is matched while the Keypad Interrupt function is active. An interrupt will be generated if it has been enabled by setting the EKBI bit in IEN1 register and EA = 1. The PATN_SEL bit in the Keypad Interrupt Control Register (KBCON) is used to define equal or not-equal for the comparison.

In order to use the Keypad Interrupt as an original KBI function like in the 87LPC76x series, the user needs to set KBPATN = 0FFH and PATN_SEL = 0 (not equal), then any key connected to Port0 which is enabled by KBMASK register is will cause the hardware to set KBIF = 1 and generate an interrupt if it has been enabled. The interrupt may be used to wake-up the CPU from Idle or Power-down modes. This feature is particularly useful in handheld, battery powered systems that need to carefully manage power consumption yet also need to be convenient to use.

In order to set the flag and cause an interrupt, the pattern on Port 0 must be held longer than 6 CCLKs

Table 80: Keypad Pattern register (KBPATN - address 93h) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	KBPATN.7	KBPATN.6	KBPATN.5	KBPATN.4	KBPATN.3	KBPATN.2	KBPATN.1	KBPATN.0
Reset	1	1	1	1	1	1	1	1

Table 81: Keypad Pattern register (KBPATN - address 93h) bit description

Bit	Symbol	Access	Description
0:7	KBPATN.7:0	R/W	Pattern bit 0 - bit 7

Table 82: Keypad Control register (KBCON - address 94h) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	-	-	-	-	-	-	PATN_SEL	KBIF
Reset	x	x	x	x	x	x	0	0

Table 83: Keypad Control register (KBCON - address 94h) bit description

Bit	Symbol	Access	Description
0	KBIF	R/W	Keypad Interrupt Flag. Set when Port 0 matches user defined conditions specified in KBPATN, KBMASK, and PATN_SEL. Needs to be cleared by software by writing logic 0.
1	PATN_SEL	R/W	Pattern Matching Polarity selection. When set, Port 0 has to be equal to the user-defined Pattern in KBPATN to generate the interrupt. When clear, Port 0 has to be not equal to the value of KBPATN register to generate the interrupt.
2:7	-	-	reserved

Table 84: Keypad Interrupt Mask register (KBMASK - address 86h) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	KBMASK.7	KBMASK.6	KBMASK.5	KBMASK.4	KBMASK.3	KBMASK.2	KBMASK.1	KBMASK.0
Reset	0	0	0	0	0	0	0	0

Table 85: Keypad Interrupt Mask register (KBMASK - address 86h) bit description

Bit	Symbol	Description
0	KBMASK.0	When set, enables P0.0 as a cause of a Keypad Interrupt.
1	KBMASK.1	When set, enables P0.1 as a cause of a Keypad Interrupt.
2	KBMASK.2	When set, enables P0.2 as a cause of a Keypad Interrupt.
3	KBMASK.3	When set, enables P0.3 as a cause of a Keypad Interrupt.
4	KBMASK.4	When set, enables P0.4 as a cause of a Keypad Interrupt.
5	KBMASK.5	When set, enables P0.5 as a cause of a Keypad Interrupt.
6	KBMASK.6	When set, enables P0.6 as a cause of a Keypad Interrupt.
7	KBMASK.7	When set, enables P0.7 as a cause of a Keypad Interrupt.

[1] The Keypad Interrupt must be enabled in order for the settings of the KBMASK register to be effective.

15. Watchdog timer (WDT)

The watchdog timer subsystem protects the system from incorrect code execution by causing a system reset when it underflows as a result of a failure of software to feed the timer prior to the timer reaching its terminal count. The watchdog timer can only be reset by a power-on reset.

15.1 Watchdog function

The user has the ability using the WDCON and UCFG1 registers to control the run /stop condition of the WDT, the clock source for the WDT, the prescaler value, and whether the WDT is enabled to reset the device on underflow. In addition, there is a safety mechanism which forces the WDT to be enabled by values programmed into UCFG1 either through IAP or a commercial programmer.

The WDTE bit (UCFG1.7), if set, enables the WDT to reset the device on underflow. Following reset, the WDT will be running regardless of the state of the WDTE bit.

The WDRUN bit (WDCON.2) can be set to start the WDT and cleared to stop the WDT. Following reset this bit will be set and the WDT will be running. All writes to WDCON need to be followed by a feed sequence (see Section 15.2). Additional bits in WDCON allow the user to select the clock source for the WDT and the prescaler.

When the timer is not enabled to reset the device on underflow, the WDT can be used in ‘timer mode’ and be enabled to produce an interrupt (IEN0.6) if desired

The Watchdog Safety Enable bit, WDSE (UCFG1.4) along with WDTE, is designed to force certain operating conditions at power-up. Refer to Table 86 for details.

Figure 49 shows the watchdog timer in watchdog mode. It consists of a programmable 13-bit prescaler, and an 8-bit down counter. The down counter is clocked (decremented) by a tap taken from the prescaler. The clock source for the prescaler is either PCLK or the watchdog oscillator selected by the WDCLK bit in the WDCON register. (Note that switching of the clock sources will not take effect immediately - see Section 15.3).

The watchdog asserts the watchdog reset when the watchdog count underflows and the watchdog reset is enabled. When the watchdog reset is enabled, writing to WDL or WDCON must be followed by a feed sequence for the new values to take effect.

If a watchdog reset occurs, the internal reset is active for at least one watchdog clock cycle (PCLK or the watchdog oscillator clock). If CCLK is still running, code execution will begin immediately after the reset cycle. If the processor was in Power-down mode, the watchdog reset will start the oscillator and code execution will resume after the oscillator is stable.

Table 86: Watchdog timer configuration

WDTE	WDSE	FUNCTION
0	x	The watchdog reset is disabled. The timer can be used as an internal timer and can be used to generate an interrupt. WDSE has no effect.
1	0	The watchdog reset is enabled. The user can set WDCLK to choose the clock source.
1	1	The watchdog reset is enabled, along with additional safety features: 1. WDCLK is forced to 1 (using watchdog oscillator) 2. WDCON and WDL register can only be written once 3. WDRUN is forced to 1

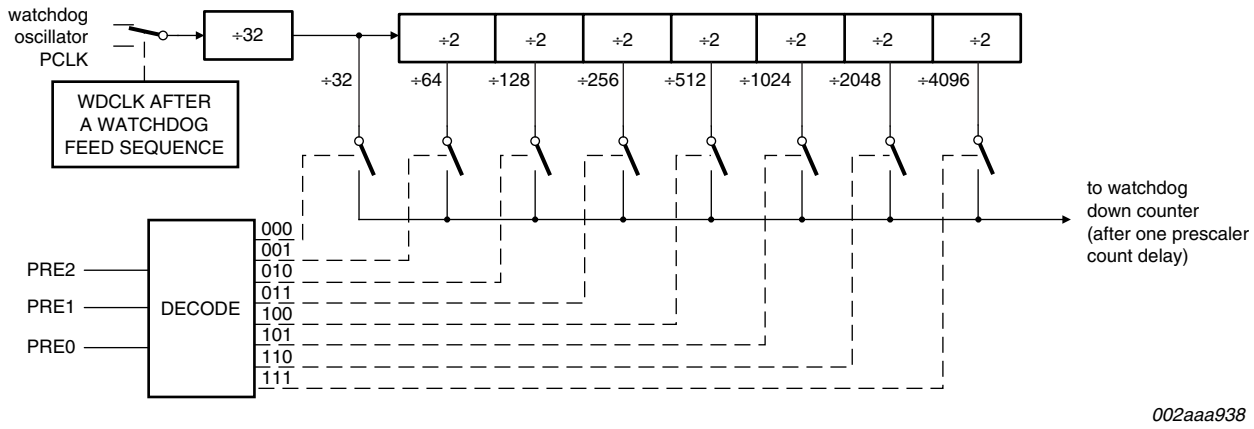


Fig 47. Watchdog Prescaler.

15.2 Feed sequence

The watchdog timer control register and the 8-bit down counter (See Figure 48) are not directly loaded by the user. The user writes to the WDCON and the WDL SFRs. At the end of a feed sequence, the values in the WDCON and WDL SFRs are loaded to the control register and the 8-bit down counter. Before the feed sequence, any new values written to

these two SFRs will not take effect. To avoid a watchdog reset, the watchdog timer needs to be fed (via a special sequence of software action called the feed sequence) prior to reaching an underflow.

To feed the watchdog, two write instructions must be sequentially executed successfully. Between the two write instructions, SFR reads are allowed, but writes are not allowed. The instructions should move A5H to the WFEED1 register and then 5AH to the WFEED2 register. An incorrect feed sequence will cause an immediate watchdog reset. The program sequence to feed the watchdog timer is as follows:

```
CLR EA ;disable interrupt
MOV WFEED1,#0A5h ;do watchdog feed part 1
MOV WFEED2,#05Ah ;do watchdog feed part 2
SETB EA ;enable interrupt
```

This sequence assumes that the P89LPC932A1 interrupt system is enabled and there is a possibility of an interrupt request occurring during the feed sequence. If an interrupt was allowed to be serviced and the service routine contained any SFR writes, it would trigger a watchdog reset. If it is known that no interrupt could occur during the feed sequence, the instructions to disable and re-enable interrupts may be removed.

In watchdog mode (WDTE = 1), writing the WDCON register must be IMMEDIATELY followed by a feed sequence to load the WDL to the 8-bit down counter, and the WDCON to the shadow register. If writing to the WDCON register is not immediately followed by the feed sequence, a watchdog reset will occur.

For example: setting WDRUN = 1:

```
MOV ACC,WDCON ;get WDCON
SETB ACC.2 ;set WD_RUN=1
MOV WDL,#0FFh ;New count to be loaded to 8-bit down counter
CLR EA ;disable interrupt
MOV WDCON,ACC ;write back to WDCON (after the watchdog is enabled, a feed
must occur ; immediately)
MOV WFEED1,#0A5h ;do watchdog feed part 1
MOV WFEED2,#05Ah ;do watchdog feed part 2
SETB EA ;enable interrupt
```

In timer mode (WDTE = 0), WDCON is loaded to the control register every CCLK cycle (no feed sequence is required to load the control register), but a feed sequence is required to load from the WDL SFR to the 8-bit down counter before a time-out occurs.

The number of watchdog clocks before timing out is calculated by the following equations:

$$tclks = (2^{(5+PRE)})(WDL+1)+1 \quad (1)$$

where:

PRE is the value of prescaler (PRE2 to PRE0) which can be the range 0 to 7, and;

WDL is the value of watchdog load register which can be the range of 0 to 255.

The minimum number of tclks is:

$$tclks = (2^{(5+0)})(0+1)+1 = 33 \quad (2)$$

The maximum number of tclks is:

$$tclks = (2^{(5+7)})(255+1)+1 = 1048577 \quad (3)$$

Table 89 shows sample P89LPC932A1 timeout values.

Table 87: Watchdog Timer Control register (WDCON - address A7h) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	PRE2	PRE1	PRE0	-	-	WDRUN	WDTOF	WDCLK
Reset	1	1	1	x	x	1	1/0	1

Table 88: Watchdog Timer Control register (WDCON - address A7h) bit description

Bit	Symbol	Description
0	WDCLK	Watchdog input clock select. When set, the watchdog oscillator is selected. When cleared, PCLK is selected. (If the CPU is powered down, the watchdog is disabled if WDCLK = 0, see Section 15.5). (Note: If both WDTE and WDSE are set to 1, this bit is forced to 1.) Refer to Section 15.3 for details.
1	WDTOF	Watchdog Timer Time-Out Flag. This bit is set when the 8-bit down counter underflows. In watchdog mode, a feed sequence will clear this bit. It can also be cleared by writing a logic 0 to this bit in software.
2	WDRUN	Watchdog Run Control. The watchdog timer is started when WDRUN = 1 and stopped when WDRUN = 0. This bit is forced to 1 (watchdog running) and cannot be cleared to zero if both WDTE and WDSE are set to 1.
3:4	-	reserved
5	PRE0	Clock Prescaler Tap Select. Refer to Table 89 for details.
6	PRE1	
7	PRE2	

Table 89: Watchdog timeout vales

PRE2 to PRE0	WDL in decimal)	Timeout Period (in watchdog clock cycles)	Watchdog Clock Source	
			400 KHz Watchdog Oscillator Clock (Nominal)	12 MHz CCLK (6 MHz $^{CCLK}/_{2}$ Watchdog Clock)
000	0	33	82.5 µs	5.50 µs
	255	8,193	20.5 ms	1.37 ms
001	0	65	162.5 µs	10.8 µs
	255	16,385	41.0 ms	2.73 ms
010	0	129	322.5 µs	21.5 µs
	255	32,769	81.9 ms	5.46 ms
011	0	257	642.5 µs	42.8 µs
	255	65,537	163.8 ms	10.9 ms
100	0	513	1.28 ms	85.5 µs
	255	131,073	327.7 ms	21.8 ms
101	0	1,025	2.56 ms	170.8 µs
	255	262,145	655.4 ms	43.7 ms

Table 89: Watchdog timeout vales *...continued*

PRE2 to PRE0	WDL in decimal)	Timeout Period (in watchdog clock cycles)	Watchdog Clock Source	
			400 KHz Watchdog Oscillator Clock (Nominal)	12 MHz CCLK (6 MHz $^{CCLK}/_{2}$ Watchdog Clock)
110	0	2,049	5.12 ms	341.5 µs
	255	524,289	1.31 s	87.4 ms
111	0	4097	10.2 ms	682.8 µs
	255	1,048,577	2.62 s	174.8 ms

15.3 Watchdog clock source

The watchdog timer system has an on-chip 400 KHz oscillator. The watchdog timer can be clocked from either the watchdog oscillator or from PCLK (refer to Figure 47) by configuring the WDCLK bit in the Watchdog Control Register WDCON. When the watchdog feature is enabled, the timer must be fed regularly by software in order to prevent it from resetting the CPU.

After changing WDCLK (WDCON.0), switching of the clock source will not immediately take effect. As shown in Figure 49, the selection is loaded after a watchdog feed sequence. In addition, due to clock synchronization logic, it can take two old clock cycles before the old clock source is deselected, and then an additional two new clock cycles before the new clock source is selected.

Since the prescaler starts counting immediately after a feed, switching clocks can cause some inaccuracy in the prescaler count. The inaccuracy could be as much as 2 old clock source counts plus 2 new clock cycles.

Note: When switching clocks, it is important that the old clock source is left enabled for two clock cycles after the feed completes. Otherwise, the watchdog may become disabled when the old clock source is disabled. For example, suppose PCLK (WCLK = 0) is the current clock source. After WCLK is set to logic 1, the program should wait at least two PCLK cycles (4 CCLKs) after the feed completes before going into Power-down mode. Otherwise, the watchdog could become disabled when CCLK turns off. The watchdog oscillator will never become selected as the clock source unless CCLK is turned on again first.

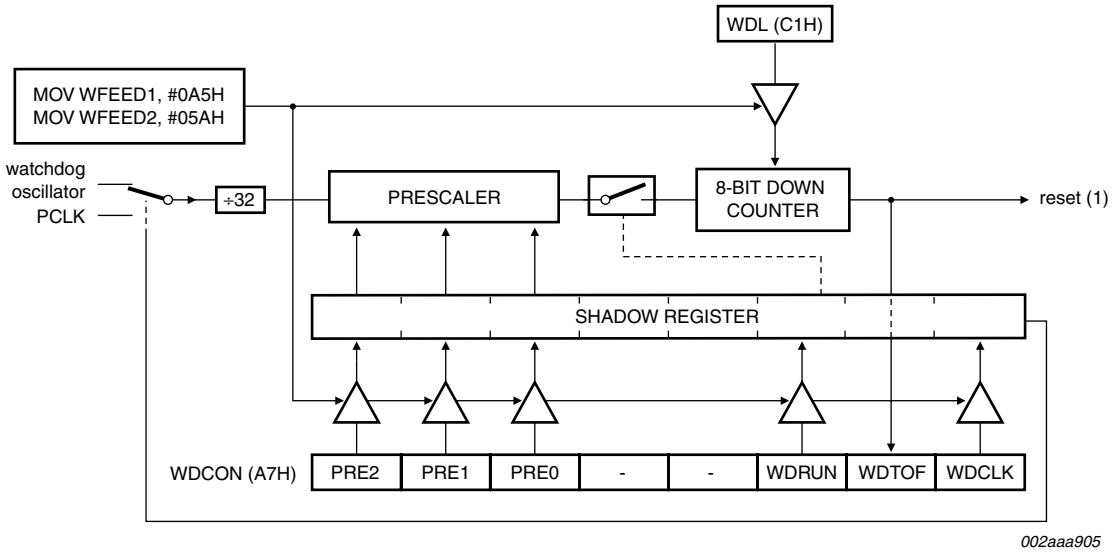


Fig 48. Watchdog Timer in Watchdog Mode (WDTE = 1).

15.4 Watchdog Timer in Timer mode

Figure 49 shows the Watchdog Timer in Timer Mode. In this mode, any changes to WDCON are written to the shadow register after one watchdog clock cycle. A watchdog underflow will set the WDTOF bit. If IEN0.6 is set, the watchdog underflow is enabled to cause an interrupt. WDTOF is cleared by writing a logic 0 to this bit in software. When an underflow occurs, the contents of WDL is reloaded into the down counter and the watchdog timer immediately begins to count down again.

A feed is necessary to cause WDL to be loaded into the down counter before an underflow occurs. Incorrect feeds are ignored in this mode.

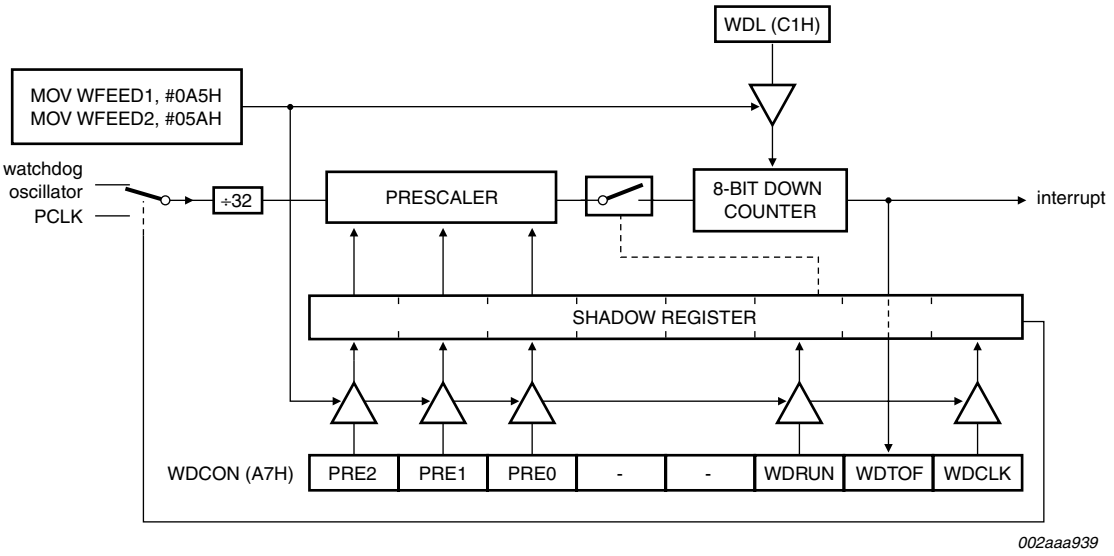


Fig 49. Watchdog Timer in Timer Mode (WDTE = 0).

15.5 Power-down operation

The WDT oscillator will continue to run in power-down, consuming approximately 50 μA, as long as the WDT oscillator is selected as the clock source for the WDT. Selecting PCLK as the WDT source will result in the WDT oscillator going into power-down with the rest of the device (see Section 15.3). Power-down mode will also prevent PCLK from running and therefore the watchdog is effectively disabled.

15.6 Periodic wake-up from power-down without an external oscillator

Without using an external oscillator source, the power consumption required in order to have a periodic wake-up is determined by the power consumption of the internal oscillator source used to produce the wake-up. The Real-time clock running from the internal RC oscillator can be used. The power consumption of this oscillator is approximately 300 μA. Instead, if the WDT is used to generate interrupts the current is reduced to approximately 50 μA. Whenever the WDT underflows, the device will wake-up.

16. Additional features

The AUXR1 register contains several special purpose control bits that relate to several chip features. AUXR1 is described in Table 91

Table 90: AUXR1 register (address A2h) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	CLKLP	EBRR	ENT1	ENT0	SRST	0	-	DPS
Reset	0	0	0	0	0	0	x	0

Table 91: AUXR1 register (address A2h) bit description

Bit	Symbol	Description
0	DPS	Data Pointer Select. Chooses one of two Data Pointers.
1	-	Not used. Allowable to set to a logic 1.
2	0	This bit contains a hard-wired 0. Allows toggling of the DPS bit by incrementing AUXR1, without interfering with other bits in the register.
3	SRST	Software Reset. When set by software, resets the P89LPC932A1 as if a hardware reset occurred.
4	ENT0	When set the P1.2 pin is toggled whenever Timer 0 overflows. The output frequency is therefore one half of the Timer 0 overflow rate. Refer to Section 7 “Timers 0 and 1” for details.
5	ENT1	When set, the P0.7 pin is toggled whenever Timer 1 overflows. The output frequency is therefore one half of the Timer 1 overflow rate. Refer to Section 7 “Timers 0 and 1” for details.
6	EBRR	UART Break Detect Reset Enable. If logic 1, UART Break Detect will cause a chip reset and force the device into ISP mode.
7	CLKLP	Clock Low Power Select. When set, reduces power consumption in the clock circuits. Can be used when the clock frequency is 8 MHz or less. After reset this bit is cleared to support up to 12 MHz operation.

16.1 Software reset

The SRST bit in AUXR1 gives software the opportunity to reset the processor completely, as if an external reset or watchdog reset had occurred. If a value is written to AUXR1 that contains a 1 at bit position 3, all SFRs will be initialized and execution will resume at program address 0000. Care should be taken when writing to AUXR1 to avoid accidental software resets.

16.2 Dual Data Pointers

The dual Data Pointers (DPTR) adds to the ways in which the processor can specify the address used with certain instructions. The DPS bit in the AUXR1 register selects one of the two Data Pointers. The DPTR that is not currently selected is not accessible to software unless the DPS bit is toggled.

Specific instructions affected by the Data Pointer selection are:

INC DPTR — Increments the Data Pointer by 1

JMP@A+DPTR — Jump indirect relative to DPTR value

MOV DPTR, #data16 — Load the Data Pointer with a 16-bit constant

MOVC A, @A+DPTR — Move code byte relative to DPTR to the accumulator

MOVX A, @DPTR — Move accumulator to data memory relative to DPTR

MOVX @DPTR, A — Move from data memory relative to DPTR to the accumulator

Also, any instruction that reads or manipulates the DPH and DPL registers (the upper and lower bytes of the current DPTR) will be affected by the setting of DPS. The MOVX instructions have limited application for the P89LPC932A1 since the part does not have an external data bus. However, they may be used to access Flash configuration information (see Flash Configuration section) or auxiliary data (XDATA) memory.

Bit 2 of AUXR1 is permanently wired as a logic 0. This is so that the DPS bit may be toggled (thereby switching Data Pointers) simply by incrementing the AUXR1 register, without the possibility of inadvertently altering other bits in the register.

17. Data EEPROM

The P89LPC932A1 has 512 bytes of on-chip Data EEPROM that can be used to save configuration parameters. The Data EEPROM is SFR based, byte readable, byte writable, and erasable (via row fill and sector fill). The user can read, write, and fill the memory via three SFRs and one interrupt:

- Address Register (DEEADR) is used for address bits 7 to 0 (bit 8 is in the DEECON register).
- Control Register (DEECON) is used for address bit 8, setup operation mode, and status flag bit (see Table 92).
- Data Register (DEEDAT) is used for writing data to, or reading data from, the Data EEPROM.

Table 92: Data EEPROM control register (DEECON address F1h) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	EEIF	HVERR	ECTL1	ECTL0	-	-	-	EADR8
Reset	0	0	0	0	0	0	x	0

Table 93: Data EEPROM control register (DEECON address F1h) bit description

Bit	Symbol	Description
0	EADR8	Most significant address (bit 8) of the Data EEPROM. EADR7-0 are in DEEADR.
1:3		Reserved.
5:4	ECTL1:0	Operation mode selection:
		The following modes are selected by ECTL[1:0]: **00 —** Byte read / write mode. **01 —** Reserved. **10 —** Row (64 bytes) fill. **11 —** Block fill (512 bytes).
6	HVERR	High voltage error. Indicates a programming voltage error during program or erase.
7	EEIF	Data EEPROM interrupt flag. Set when a read or write finishes, reset by software.

Byte Mode: In this mode data can be read and written to one byte at a time. Data is in the DEEDAT register and the address is in the DEEADR register. Each write requires approximately 4 ms to complete. Each read requires three machines after writing the address to the DEEADR register.

Row Fill: In this mode the addressed row (64 bytes, with address DEEADR[5:0] ignored) is filled with the DEEDAT pattern. To erase the entire row to 00h or program the entire row to FFh, write 00h or FFh to DEEDAT prior to row fill. Each row fill requires approximately 4 ms to complete.

Block Fill: In this mode all 512 bytes are filled with the DEEDAT pattern. To erase the block to 00h or program the block to FFh, write 00h or FFh to DEEDAT prior to the block fill. Prior to using this command EADR8 must be set = 1. Each Block Fill requires approximately 4 ms to complete.

In any mode, after the operation finishes, the hardware will set EEIF bit. An interrupt can be enabled via the IEN1.7 bit. If IEN1.7 and the EA bits are set, it will generate an interrupt request. The EEIF bit will need to be cleared by software.

17.1 Data EEPROM read

A byte can be read via polling or interrupt:

1. Write to DEECON with ECTL1/ECTL0 (DEECON[5:4]) = ‘00’ and correct bit 8 address to EADR8. (Note that if the correct values are already written to DEECON, there is no need to write to this register.)
2. Without writing to the DEEDAT register, write address bits 7 to 0 to DEEADR.
3. If both the EIEE (IEN1.7) bit and the EA (IEN0.7) bit are logic 1s, wait for the Data EEPROM interrupt then read/poll the EEIF (DEECON.7) bit until it is set to logic 1. If EIEE or EA is logic 0, the interrupt is disabled, only polling is enabled.
4. Read the Data EEPROM data from the DEEDAT SFR.

Note that if DEEDAT is written prior to a write to DEEADR (if DEECON[5:4] = '00'), a Data EEPROM write operation will commence. The user must take caution that such cases do not occur during a read. An example is if the Data EEPROM is read in an interrupt service routine with the interrupt occurring in the middle of a Data EEPROM cycle. The user should disable interrupts during a Data EEPROM write operation (see Section 17.2).

17.2 Data EEPROM write

A byte can be written via polling or interrupt:

1. Write to DEECON with ECTL1/ECTL0 (DEECON[5:4]) = '00' and correct bit 8 address to EADR8. (Note that if the correct values are already written to DEECON, there is no need to write to this register.)
2. Write the data to the DEEDAT register.
3. Write address bits 7 to 0 to DEEADR.
4. If both the EIEE (IEN1.7) bit and the EA (IEN0.7) bit are logic 1s, wait for the Data EEPROM interrupt then read/poll the EEIF (DEECON.7) bit until it is set to logic 1. If EIEE or EA is logic 0, the interrupt is disabled and only polling is enabled. When EEIF is logic 1, the operation is complete and data is written.

As a write to the DEEDAT register followed by a write to the DEEADR register will automatically set off a write (if DEECON[5:4] = '00'), the user must take great caution in a write to the DEEDAT register. It is strongly recommended that the user disables interrupts prior to a write to the DEEDAT register and enable interrupts after all writes are over. An example is as follows:

```
CLR EA          ;disable interrupt
MOV DEEDAT,@R0  ;write data pattern
MOV DEEADR,@R1  ;write address for the data
SETB EA         ;wait for the interrupt orpoll the DEECON.7 (EEIF) bit
```

17.3 Hardware reset

During any hardware reset, including watchdog and system timer reset, the state machine that 'remembers' a write to the DEEDAT register will be initialized. If a write to the DEEDAT register occurs followed by a hardware reset, a write to the DEEADR register without a prior write to the DEEDAT register will result in a read cycle.

17.4 Multiple writes to the DEEDAT register

If there are multiple writes to the DEEDAT register before a write to the DEEADR register, the last data written to the DEEDAT register will be written to the corresponding address.

17.5 Sequences of writes to DEECON and DEEDAT registers

A write to the DEEDAT register is considered a valid write (i.e, will trigger the state machine to 'remember' a write operation is to commence) if DEECON[5:4] = '00'. If these mode bits are already '00' and address bit 8 is correct, there is no need to write to the DEECON register prior to a write to the DEEDAT register.

17.6 Data EEPROM Row Fill

A row (64 bytes) can be filled with a predetermined data pattern via polling or interrupt:

1. Write to DEECON with ECTL1/ECTL0 (DEECON[5:4]) = '10' and correct bit 8 address to EADR8. (Note that if the correct values are already written to DEECON, there is no need to write to this register.)
2. Write the fill pattern to the DEEDAT register. (Note that if the correct values are already written to DEEDAT, there is no need to write to this register.)
3. Write address bits 7 to 0 to DEEADR. Note that address bits 5 to 0 are ignored.
4. If both the EIEE (IEN1.7) bit and the EA (IEN0.7) bit are logic 1s, wait for the Data EEPROM interrupt then read/poll the EEIF (DEECON.7) bit until it is set to logic 1. If EIEE or EA is logic 0, the interrupt is disabled and only polling is enabled. When EEIF is logic 1, the operation is complete and row is filled with the DEEDAT pattern.

17.7 Data EEPROM Block Fill

The Data EEPROM array can be filled with a predetermined data pattern via polling or interrupt:

1. Write to DEECON with ECTL1/ECTL0 (DEECON[5:4]) = '11'. Set bit EADR8 = 1.
2. Write the fill pattern to the DEEDAT register.
3. Write any address to DEEADR. Note that the entire address is ignored in a block fill operation.
4. If both the EIEE (IEN1.7) bit and the EA (IEN0.7) bit are logic 1s, wait for the Data EEPROM interrupt then read/poll the EEIF (DEECON.7) bit until it is set to logic 1. If EIEE or EA is logic 0, the interrupt is disabled and only polling is enabled. When EEIF is logic 1, the operation is complete.

18. Flash memory

18.1 General description

The P89LPC932A1 Flash memory provides in-circuit electrical erasure and programming. The Flash can be read and written as bytes. The Sector and Page Erase functions can erase any Flash sector (1 kB) or page (64 bytes). The Chip Erase operation will erase the entire program memory. Five Flash programming methods are available. On-chip erase and write timing generation contribute to a user-friendly programming interface. The P89LPC932A1 Flash reliably stores memory contents even after 100,000 erase and program cycles. The cell is designed to optimize the erase and programming mechanisms. P89LPC932A1 uses V_{DD} as the supply voltage to perform the Program/Erase algorithms

18.2 Features

- Parallel programming with industry-standard commercial programmers
- In-Circuit serial Programming (ICP) with industry-standard commercial programmers.**(This feature was not present in the original P89LPC932).**
- IAP-Lite allows individual and multiple bytes of code memory to be used for data storage and programmed under control of the end application.**(This feature was not present in the original P89LPC932).**

- Internal fixed boot ROM, containing low-level In-Application Programming (IAP) routines that can be called from the end application (in addition to IAP-Lite).
- Default serial loader providing In-System Programming (ISP) via the serial port, located in upper end of user program memory.
- Boot vector allows user provided Flash loader code to reside anywhere in the Flash memory space, providing flexibility to the user.
- Programming and erase over the full operating voltage range
- Read/Programming/Erase using ISP/IAP/IAP-Lite
- Any flash program operation in 2 ms (4 ms for erase/program)
- Programmable security for the code in the Flash for each sector
- > 100,000 typical erase/program cycles for each byte
- 10-year minimum data retention

18.3 Flash programming and erase

The P89LPC932A1 program memory consists 1 kB sectors. Each sector can be further divided into 64-byte pages. In addition to sector erase and page erase, a 64-byte page register is included which allows from 1 to 64 bytes of a given page to be programmed at the same time, substantially reducing overall programming time. Five methods of programming this device are available.

- Parallel programming with industry-standard commercial programmers.
- In-Circuit serial Programming (ICP) with industry-standard commercial programmers. **(This feature was not present in the original P89LPC932).**
- IAP-Lite allows individual and multiple bytes of code memory to be used for data storage and programmed under control of the end application.**(This feature was not present in the original P89LPC932).**
- Internal fixed boot ROM, containing low-level In-Application Programming (IAP) routines that can be called from the end application (in addition to IAP-Lite).
- A factory-provided default serial loader, located in upper end of user program memory, providing In-System Programming (ISP) via the serial port.

18.4 Using Flash as data storage: IAP-Lite

(This feature was not present in the original P89LPC932).

The Flash code memory array of this device supports IAP-Lite in addition to standard IAP functions. Any byte in a non-secured sector of the code memory array may be read using the MOVC instruction and thus is suitable for use as non-volatile data storage. IAP-Lite provides an erase-program function that makes it easy for one or more bytes within a page to be erased and programmed in a single operation without the need to erase or program any other bytes in the page. IAP-Lite is performed in the application under the control of the microcontroller's firmware using four SFRs and an internal 64-byte 'page register' to facilitate erasing and programing within unsecured sectors. These SFRs are:

- FMCON (Flash Control Register). When read, this is the status register. When written, this is a command register. Note that the status bits are cleared to logic 0s when the command is written.

- FMADRL, FMADRH (Flash memory address low, Flash memory address high). Used to specify the byte address within the page register or specify the page within user code memory
- FMDATA (Flash Data Register). Accepts data to be loaded into the page register.

The page register consists of 64 bytes and an update flag for each byte. When a LOAD command is issued to FMCON the page register contents and all of the update flags will be cleared. When FMDATA is written, the value written to FMDATA will be stored in the page register at the location specified by the lower 6 bits of FMADRL. In addition, the update flag for that location will be set. FMADRL will auto-increment to the next location. Auto-increment after writing to the last byte in the page register will 'wrap-around' to the first byte in the page register, but will not affect FMADRL[7:6]. Bytes loaded into the page register do not have to be continuous. Any byte location can be loaded into the page register by changing the contents of FMADRL prior to writing to FMDATA. However, each location in the page register can only be written once following each LOAD command. Attempts to write to a page register location more than once should be avoided.

FMADRH and FMADRL[7:6] are used to select a page of code memory for the erase-program function. When the erase-program command is written to FMCON, the locations within the code memory page that correspond to updated locations in the page register, will have their contents erased and programmed with the contents of their corresponding locations in the page register. Only the bytes that were loaded into the page register will be erased and programmed in the user code array. Other bytes within the user code memory will not be affected.

Writing the erase-program command (68H) to FMCON will start the erase-program process and place the CPU in a program-idle state. The CPU will remain in this idle state until the erase-program cycle is either completed or terminated by an interrupt. When the program-idle state is exited FMCON will contain status information for the cycle.

If an interrupt occurs during an erase/programming cycle, the erase/programming cycle will be aborted and the OI flag (Operation Interrupted) in FMCON will be set. If the application permits interrupts during erasing-programming the user code should check the OI flag (FMCON.0) after each erase-programming operation to see if the operation was aborted. If the operation was aborted, the user's code will need to repeat the process starting with loading the page register.

The erase-program cycle takes 4 ms (2 ms for erase, 2 ms for programming) to complete, regardless of the number of bytes that were loaded into the page register.

Erasing-programming of a single byte (or multiple bytes) in code memory is accomplished using the following steps:

- Write the LOAD command (00H) to FMCON. The LOAD command will clear all locations in the page register and their corresponding update flags.
- Write the address within the page register to FMADRL. Since the loading the page register uses FMADRL[5:0], and since the erase-program command uses FMADRH and FMADRL[7:6], the user can write the byte location within the page register (FMADRL[5:0]) and the code memory page address (FMADRH and FMADRL[7:6]) at this time.
- Write the data to be programmed to FMDATA. This will increment FMADRL pointing to the next byte in the page register.

- Write the address of the next byte to be programmed to FMADRL, if desired. (Not needed for contiguous bytes since FMADRL is auto-incremented). All bytes to be programmed must be within the same page.
- Write the data for the next byte to be programmed to FMDATA.
- Repeat writing of FMADRL and/or FMDATA until all desired bytes have been loaded into the page register.
- Write the page address in user code memory to FMADRH and FMADRL[7:6], if not previously included when writing the page register address to FMADRL[5:0].
- Write the erase-program command (68H) to FMCON, starting the erase-program cycle.
- Read FMCON to check status. If aborted, repeat starting with the LOAD command.

Table 94: Flash Memory Control register (FMCON - address E4h) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol (R)	-	-	-	-	HVA	HVE	SV	OI
Symbol (W)	FMCMD.7	FMCMD.6	FMCMD.5	FMCMD.4	FMCMD.3	FMCMD.2	FMCMD.1	FMCMD.0
Reset	0	0	0	0	0	0	0	0

Table 95: Flash Memory Control register (FMCON - address E4h) bit description

Bit	Symbol	Access	Description
0	OI	R	Operation interrupted. Set when cycle aborted due to an interrupt or reset.
	FMCMD.0	W	Command byte bit 0.
1	SV	R	Security violation. Set when an attempt is made to program, erase, or CRC a secured sector or page.
	FMCMD.1	W	Command byte bit 1
2	HVE	R	High voltage error. Set when an error occurs in the high voltage generator.
	FMCMD.2	W	Command byte bit 2.
3	HVA	R	High voltage abort. Set if either an interrupt or a brown-out is detected during a program or erase cycle. Also set if the brown-out detector is disabled at the start of a program or erase cycle.
	FMCMD.3	W	Command byte bit 3.
4:7	-	R	reserved
4:7	FMCMD.4	W	Command byte bit 4.
4:7	FMCMD.5	W	Command byte bit 5.
4:7	FMCMD.6	W	Command byte bit 6.
4:7	FMCMD.7	W	Command byte bit 7.

An assembly language routine to load the page register and perform an erase/program operation is shown below.

```
;*************************************************
;* pgm user code  *
;*************************************************
;*                                               *
;* Inputs:                                       *
;*R3 = number of bytes to program (byte)         *
;*R4 = page address MSB(byte)                    *
```

```
;*R5 = page address LSB(byte)                        *
;*R7 = pointer to data buffer in RAM(byte)           *
;* Outputs:                                           *
;*R7 = status (byte)                                 *
;* C = clear on no error, set on error               *
;*****************************************************

LOAD     EQU     00H
EP       EQU     68H

PGM_USER:
     MOV     FMCON,#LOAD     ;load command, clears page register
     MOV     FMADRH,R4     ;get high address
     MOV     FMADRL,R5     ;get low address
     MOV     A,R7          ;
     MOV     R0,A          ;get pointer into R0
 LOAD_PAGE:
     MOV     FMDAT,@R0     ;write data to page register
     INC     R0            ;point to next byte
     DJNZ    R3,LOAD_PAGE  ;do until count is zero
     MOV     FMCON,#EP     ;else erase & program the page

     MOV     R7,FMCON      ;copy status for return
     MOV     A,R7          ;read status
     ANL     A,#0FH        ;save only four lower bits
     JNZ     BAD           ;
     CLR     C             ;clear error flag if good
     RET                   ;and return
BAD:
     SETB    C             ;set error flag
     RET                   ;and return
```

A C-language routine to load the page register and perform an erase/program operation is shown below.

```
#include <REG931.H>
unsigned char idata dbytes[64]; // data buffer
unsigned char Fm_stat; // status result
bit PGM_USER (unsigned char, unsigned char);
bit prog_fail;

void main ()
{
     prog_fail=PGM_USER(0x1F,0xC0);
}

bit PGM_USER (unsigned char page_hi, unsigned char page_lo)
      {
           #define LOAD0x00// clear page register, enable loading
           #define EP0x68// erase & program page
           unsigned char i;// loop count
```

```
        FMCON = LOAD; //load command, clears page reg
        FMADRH = page_hi; //
        FMADRL = page_lo; //write my page address to addr regs

        for(i=0;i<64;i=i+1)
       {
                  FMDATA = dbytes[i];
        }
        FMCON = EP; //erase & prog page command
        Fm_stat = FMCON; //read the result status
        if ((Fm_stat & 0x0F)!=0) prog_fail=1; else prog_fail=0;
        return(prog_fail);
}
```

18.5 In-circuit programming (ICP)

(This feature was not present in the original P89LPC932).

In-Circuit Programming is a method intended to allow commercial programmers to program and erase these devices without removing the microcontroller from the system. The In-Circuit Programming facility consists of a series of internal hardware resources to facilitate remote programming of the P89LPC932A1 through a two-wire serial interface. Philips has made in-circuit programming in an embedded application possible with a minimum of additional expense in components and circuit board area. The ICP function uses five pins (V_{DD}, V_{SS}, P0.5, P0.4, and $\overline{RST}$). Only a small connector needs to be available to interface your application to an external programmer in order to use this feature.

18.6 ISP and IAP capabilities of the P89LPC932A1

An In-Application Programming (IAP) interface is provided to allow the end user's application to erase and reprogram the user code memory. In addition, erasing and reprogramming of user-programmable bytes including UCFG1, the Boot Status Bit, and the Boot Vector is supported. As shipped from the factory, the upper 512 bytes of user code space contains a serial In-System Programming (ISP) loader allowing for the device to be programmed in circuit through the serial port. This ISP boot loader will, in turn, call low-level routines through the same common entry point that can be used by the end-user application.

18.7 Boot ROM

When the microcontroller contains a a 256 byte Boot ROM that is separate from the user's Flash program memory. This Boot ROM contains routines which handle all of the low level details needed to erase and program the user Flash memory. A user program simply calls a common entry point in the Boot ROM with appropriate parameters to accomplish the desired operation. Boot ROM operations include operations such as erase sector, erase page, program page, CRC, program security bit, etc. The Boot ROM occupies the program memory space at the top of the address space from FF00 to FFFFh, thereby not conflicting with the user program memory space. This function is in addition to the IAP-Lite feature.

18.8 Power on reset code execution

The P89LPC932A1 contains two special Flash elements: the BOOT VECTOR and the Boot Status Bit. Following reset, the P89LPC932A1 examines the contents of the Boot Status Bit. If the Boot Status Bit is set to zero, power-up execution starts at location 0000H, which is the normal start address of the user's application code. When the Boot Status Bit is set to a va one, the contents of the Boot Vector is used as the high byte of the execution address and the low byte is set to 00H.

The factory default settings for this device is shown in Table 96, below.

Note: These settings are different from the original P89LPC932.

The factory pre-programmed boot loader can be erased by the user. Users who wish to use this loader should take cautions to avoid erasing the last 1 kB sector on the device. Instead, the page erase function can be used to erase the eight 64-byte pages located in this sector. A custom boot loader can be written with the Boot Vector set to the custom boot loader, if desired.

Table 96: Boot loader address and default Boot vector

Product	Flash size	End address	Signature bytes			Sector size	Page size	Pre-programmed serial loader	Default Boot vector
			Mfg id	Id 1	Id 2				
P89LPC932A1	8 kB × 8	1FFFh	15h	DDh	1Fh	1 kB × 8	64 × 8	1E00h to 1FFFh	1Fh

18.9 Hardware activation of Boot Loader

The boot loader can also be executed by forcing the device into ISP mode during a power-on sequence (see Figure 50). This is accomplished by powering up the device with the reset pin initially held low and holding the pin low for a fixed time after V_{DD} rises to its normal operating value. This is followed by three, and only three, properly timed low-going pulses. Fewer or more than three pulses will result in the device not entering ISP mode. Timing specifications may be found in the data sheet for this device.

This has the same effect as having a non-zero status bit. This allows an application to be built that will normally execute the user code but can be manually forced into ISP operation. If the factory default setting for the Boot Vector is changed, it will no longer point to the factory pre-programmed ISP boot loader code. If this happens, the only way it is possible to change the contents of the Boot Vector is through the parallel or ICP programming method, provided that the end user application does not contain a customized loader that provides for erasing and reprogramming of the Boot Vector and Boot Status Bit. After programming the Flash, the status byte should be programmed to zero in order to allow execution of the user's application code beginning at address 0000H.

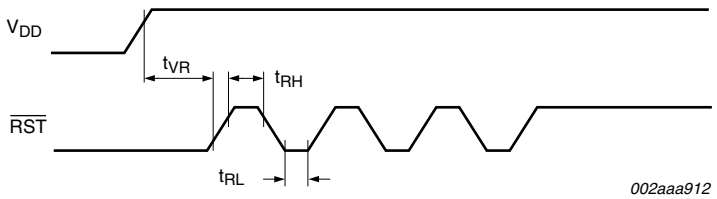


Fig 50. Forcing ISP mode.

18.10 In-system programming (ISP)

In-System Programming is performed without removing the microcontroller from the system. The In-System Programming facility consists of a series of internal hardware resources coupled with internal firmware to facilitate remote programming of the P89LPC932A1 through the serial port. This firmware is provided by Philips and embedded within each P89LPC932A1 device. The Philips In-System Programming facility has made in-circuit programming in an embedded application possible with a minimum of additional expense in components and circuit board area. The ISP function uses five pins (V_{DD}, V_{SS}, TxD, RxD, and $\overline{RST}$). Only a small connector needs to be available to interface your application to an external circuit in order to use this feature.

18.11 Using the In-system programming (ISP)

The ISP feature allows for a wide range of baud rates to be used in your application, independent of the oscillator frequency. It is also adaptable to a wide range of oscillator frequencies. This is accomplished by measuring the bit-time of a single bit in a received character. This information is then used to program the baud rate in terms of timer counts based on the oscillator frequency. The ISP feature requires that an initial character (an uppercase U) be sent to the P89LPC932A1 to establish the baud rate. The ISP firmware provides auto-echo of received characters. Once baud rate initialization has been performed, the ISP firmware will only accept Intel Hex-type records. Intel Hex records consist of ASCII characters used to represent hexadecimal values and are summarized below:

:NNAAAARRDD..DDCC<crlf>

In the Intel Hex record, the ‘NN’ represents the number of data bytes in the record. The P89LPC932A1 will accept up to 64 (40H) data bytes. The ‘AAAA’ string represents the address of the first byte in the record. If there are zero bytes in the record, this field is often set to 0000. The ‘RR’ string indicates the record type. A record type of ‘00’ is a data record. A record type of ‘01’ indicates the end-of-file mark. In this application, additional record types will be added to indicate either commands or data for the ISP facility. The maximum number of data bytes in a record is limited to 64 (decimal). ISP commands are summarized in Table 97. As a record is received by the P89LPC932A1, the information in the record is stored internally and a checksum calculation is performed. The operation indicated by the record type is not performed until the entire record has been received. Should an error occur in the checksum, the P89LPC932A1 will send an ‘X’ out the serial port indicating a checksum error. If the checksum calculation is found to match the checksum in the record, then the command will be executed. In most cases, successful reception of the record will be indicated by transmitting a ‘.’ character out the serial port.

Table 97: In-system Programming (ISP) hex record formats

Record type	Command/data function
00	Program User Code Memory Page :nnaaaa00dd..ddcc Where: nn = number of bytes to program aaaa = page address dd..dd= data bytes cc = checksum Example: :10000000010203040506070809cc
01	Read Version Id :00xxxx01cc Where: xxxx = required field but value is a ‘don’t care’ cc = checksum Example: :00000001cc
02	Miscellaneous Write Functions :02xxxx02ssddcc Where: xxxx = required field but value is a ‘don’t care’ ss= subfunction code dd= data cc = checksum Subfunction codes: 00= UCFG1 01= reserved 02= Boot Vector 03= Status Byte 04= reserved 05= reserved 06= reserved 07= reserved 08= Security Byte 0 09= Security Byte 1 0A= Security Byte 2 0B= Security Byte 3 0C= Security Byte 4 0D= Security Byte 5 0E= Security Byte 6 0F= Security Byte 7 0A= Clear Configuration Protection Example: :020000020347cc

Table 97: In-system Programming (ISP) hex record formats *…continued*

Record type	Command/data function
03	Miscellaneous Read Functions :01xxxx03sscc Where xxxx = required field but value is a ‘don’t care’ ss= subfunction code cc = checksum Subfunction codes: 00= UCFG1 01= reserved 02= Boot Vector 03= Status Byte 04= reserved 05= reserved 06= reserved 07= reserved 08= Security Byte 0 09= Security Byte 1 0A= Security Byte 2 0B= Security Byte 3 0C= Security Byte 4 0D= Security Byte 5 0E= Security Byte 6 0F= Security Byte 7 10= Manufacturer Id 11= Device Id 12= Derivative Id Example: :0100000312cc
04	Erase Sector/Page :03xxxx04ssaaaacc Where: xxxx = required field but value is a ‘don’t care’ aaaa = sector/page address ss= 01 erase sector = 00 erase page cc = checksum Example: :03000004010000F8

Table 97: In-system Programming (ISP) hex record formats *...continued*

Record type	Command/data function
05	Read Sector CRC :01xxxx05aacc Where: xxxx = required field but value is a 'don't care' aa= sector address high byte cc= checksum Example: :0100000504F6cc
06	Read Global CRC :00xxxx06cc Where: xxxx = required field but value is a 'don't care' cc= checksum Example: :00000006FA
07	Direct Load of Baud Rate :02xxxx07HHLLcc Where: xxxx = required field but value is a 'don't care' HH= high byte of timer LL = low byte of timer cc = checksum Example: :02000007FFFFcc
08	Reset MCU :00xxxx08cc Where: xxxx = required field but value is a 'don't care' cc = checksum Example: :00000008F8

18.12 In-application programming (IAP)

Several In-Application Programming (IAP) calls are available for use by an application program to permit selective erasing and programming of Flash sectors, pages, security bits, configuration bytes, and device id. All calls are made through a common interface, PGM_MTP. The programming functions are selected by setting up the microcontroller's registers before making a call to PGM_MTP at FF03H. The IAP calls are shown in Table 99.

18.13 IAP authorization key

(This feature was not present in the original P89LPC932).

IAP functions which write or erase code memory require an authorization key be set by the calling routine prior to performing the IAP function call. This authorization key is set by writing 96H to RAM location FFH. The following example was written using the Keil C compiler. The methods used to access a specific physical address in memory may vary with other compilers.

```
#include <ABSACC.H> /* enable absolute memory access */
#define key DBYTE[0xFF] /* force key to be at address 0xFF */
short (*pgm_mtp) (void) = 0xFF00; /* set pointer to IAP entry point */;
key = 0x96; /* set the authorization key */
pgm_mtp (); /* execute the IAP function call */
```

After the function call is processed by the IAP routine, the authorization key will be cleared. Thus it is necessary for the authorization key to be set prior to EACH call to PGM_MTP that requires a key. If an IAP routine that requires an authorization key is called without a valid authorization key present, the MCU will perform a reset.

18.14 Flash write enable

(This feature was not present in the original P89LPC932).

This device has hardware write enable protection. This protection applies to both ISP and IAP modes and applies to both the user code memory space and the user configuration bytes (UCFG1, BOOTVEC, and BOOTSTAT). This protection does not apply to ICP or parallel programmer modes. If the Activate Write Enable (AWE) bit in BOOTSTAT.7 is a logic 0, an internal Write Enable (WE) flag is forced set and writes to the flash memory and configuration bytes are enabled. If the Active Write Enable (AWE) bit is a logic 1, then the state of the internal WE flag can be controlled by the user.

The WE flag is SET by writing the Set Write Enable (08H) command to FMCON followed by a key value (96H) to FMDATA:

```
FMCON = 0x08;
FMDATA = 0x96;
```

The WE flag is CLEARED by writing the Clear Write Enable (0BH) command to FMCON followed by a key value (96H) to FMDATA, or by a reset:

```
FMCON = 0x0B;
FMDATA = 0x96;
```

The ISP function in this device sets the WE flag prior to calling the IAP routines. The IAP function in this device executes a Clear Write Enable command following any write operation. If the Write Enable function is active, user code which calls IAP routines will need to set the Write Enable flag prior to each IAP write function call.

18.15 Configuration byte protection

(This feature was not present in the original P89LPC932).

In addition to the hardware write enable protection, described above, the ‘configuration bytes’ may be separately write protected. These configuration bytes include UCFG1, BOOTVEC, and BOOTSTAT. This protection applies to both ISP and IAP modes and does not apply to ICP or parallel programmer modes.

If the Configuration Write Protect bit (CWP) in BOOTSTAT.6 is a logic 1, writes to the configuration bytes are disabled. If the Configuration Write Protect bit (CWP) is a logic 0, writes to the configuration bytes are enabled. The CWP bit is set by programming the BOOTSTAT register. This bit is cleared by using the Clear Configuration Protection (CCP) command in IAP or ISP.

The Clear Configuration Protection command can be disabled in ISP or IAP mode by programming the Disable Clear Configuration Protection bit (DCCP) in BOOTSTAT.7 to a logic 1. When DCCP is set, the CCP command may still be used in ICP or parallel programming modes. This bit is cleared by writing the Clear Configuration Protection (CCP) command in either ICP or parallel programming modes.

18.16 IAP error status

It is not possible to use the Flash memory as the source of program instructions while programming or erasing this same Flash memory. During an IAP erase, program, or CRC the CPU enters a program-idle state. The CPU will remain in this program-idle state until the erase, program, or CRC cycle is completed. These cycles are self timed. When the cycle is completed, code execution resumes. If an interrupt occurs during an erase, programming or CRC cycle, the erase, programming, or CRC cycle will be aborted so that the Flash memory can be used as the source of instructions to service the interrupt. An IAP error condition will be flagged by setting the carry flag and status information returned. The status information returned is shown in Table 98. If the application permits interrupts during erasing, programming, or CRC cycles, the user code should check the carry flag after each erase, programming, or CRC operation to see if an error occurred. If the operation was aborted, the user's code will need to repeat the operation.

Table 98: IAP error status

Bit	Flag	Description
0	OI	Operation Interrupted. Indicates that an operation was aborted due to an interrupt occurring during a program or erase cycle.
1	SV	Security Violation. Set if program or erase operation fails due to security settings. Cycle is aborted. Memory contents are unchanged. CRC output is invalid.
2	HVE	High Voltage Error. Set if error detected in high voltage generation circuits. Cycle is aborted. Memory contents may be corrupted.
3	VE	Verify error. Set during IAP programming of user code if the contents of the programmed address does not agree with the intended programmed value. IAP uses the MOVC instruction to perform this verify. Attempts to program user code that is MOVC protected can be programmed but will generate this error after the programming cycle has been completed.
4 to 7	-	unused; reads as a logic 0

Table 99: IAP function calls

IAP function	IAP call parameters
Program User Code Page (requires 'key')	Input parameters: ACC = 00h R3= number of bytes to program R4= page address (MSB) R5= page address (LSB) R7= pointer to data buffer in RAM F1= 0h = use IDATA Return parameter(s): R7= status Carry= set on error, clear on no error
Read Version Id	Input parameters: ACC = 01h Return parameter(s): R7= IAP code version id
Misc. Write (requires 'key')	Input parameters: ACC = 02h R5= data to write R7= register address 00= UCFG1 01= reserved 02= Boot Vector 03= Status Byte 04= reserved 05= reserved 06= reserved 07= reserved 08= Security Byte 0 09= Security Byte 1 0A= Security Byte 2 0B= Security Byte 3 0C= Security Byte 4 0D= Security Byte 5 0E= Security Byte 6 0F= Security Byte 7 0A = Clear Configuration Protection Return parameter(s): R7= status Carry= set on error, clear on no error

Table 99: IAP function calls *…continued*

IAP function	IAP call parameters
Misc. Read	Input parameters: ACC = 03h R7= register address 00= UCFG1 01= reserved 02= Boot Vector 03= Status Byte 04= reserved 05= reserved 06= reserved 07= reserved 08= Security Byte 0 09= Security Byte 1 0A= Security Byte 2 0B= Security Byte 3 0C= Security Byte 4 0D= Security Byte 5 0E= Security Byte 6 0F= Security Byte 7 Return parameter(s): R7= register data if no error, else error status Carry= set on error, clear on no error
Erase Sector/Page (requires ‘key’)	Input parameters: ACC = 04h R7= 00H (erase page) or 01H (erase sector) R4= sector/page address (MSB) R5=sector/page address (LSB) Return parameter(s): R7= status Carry= set on error, clear on no error

Table 99: IAP function calls *…continued*

IAP function	IAP call parameters
Read Sector CRC	Input parameters: ACC = 05h R7= sector address Return parameter(s): R4= CRC bits 31:24 R5= CRC bits 23:16 R6= CRC bits 15:8 R7= CRC bits 7:0 (if no error) R7= error status (if error) Carry= set on error, clear on no error
Read Global CRC	Input parameters: ACC = 06h Return parameter(s): R4= CRC bits 31:24 R5= CRC bits 23:16 R6= CRC bits 15:8 R7= CRC bits 7:0 (if no error) R7= error status (if error) Carry= set on error, clear on no error
Read User Code	Input parameters: ACC = 07h R4= address (MSB) R5= address (LSB) Return parameter(s): R7= data

18.17 User configuration bytes

A number of user-configurable features of the P89LPC932A1 must be defined at power-up and therefore cannot be set by the program after start of execution. These features are configured through the use of an Flash byte UCFG1 shown in Table 101

Table 100: Flash User Configuration Byte (UCFG1) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	WDTE	RPE	BOE	WDSE	-	FOSC2	FOSC1	FOSC0
Unprogrammed value	0	1	1	0	0	0	1	1

Table 101: Flash User Configuration Byte (UCFG1) bit description

Bit	Symbol	Description
0	FOSC0	CPU oscillator type select. See Section 2 “Clocks” for additional information. Combinations other than those shown in Table 102 are reserved for future use should not be used.
1	FOSC1	
2	FOSC2	
3	-	reserved

Table 101: Flash User Configuration Byte (UCFG1) bit description *...continued*

Bit	Symbol	Description
4	WDSE	Watchdog Safety Enable bit. Refer to Table 86 “Watchdog timer configuration” for details.
5	BOE	Brownout Detect Enable (see Section 5.1 “Brownout detection”)
6	RPE	Reset pin enable. When set = 1, enables the reset function of pin P1.5. When cleared, P1.5 may be used as an input pin. NOTE: During a power-up sequence, the RPE selection is overridden and this pin will always functions as a reset input. After power-up the pin will function as defined by the RPE bit. Only a power-up reset will temporarily override the selection defined by RPE bit. Other sources of reset will not override the RPE bit.
7	WDTE	Watchdog timer reset enable. When set = 1, enables the watchdog timer reset. When cleared = 0, disables the watchdog timer reset. The timer may still be used to generate an interrupt. Refer to Table 86 “Watchdog timer configuration” for details.

Table 102: Oscillator type selection

FOSC[2:0]	Oscillator configuration
111	External clock input on XTAL1.
100	Watchdog Oscillator, 400 kHz (+20/ −30 % tolerance).
011	Internal RC oscillator, 7.373 MHz ± 2.5 %.
010	Low frequency crystal, 20 kHz to 100 kHz.
001	Medium frequency crystal or resonator, 100 kHz to 4 MHz.
000	High frequency crystal or resonator, 4 MHz to 12 MHz.

18.18 User security bytes

This device has three security bits associated with each of its eight sectors, as shown in Table 103

Table 103: Sector Security Bytes (SECx) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	-	-	-	-	-	EDISx	SPEDISx	MOVCDISx
Unprogrammed value	0	0	0	0	0	0	0	0

Table 104: Sector Security Bytes (SECx) bit description

Bit	Symbol	Description
0	MOVCDISx	MOVC Disable. Disables the MOVC command for sector x. Any MOVC that attempts to read a byte in a MOVC protected sector will return invalid data. This bit can only be erased when sector x is erased.
1	SPEDISx	Sector Program Erase Disable x. Disables program or erase of all or part of sector x. This bit and sector x are erased by either a sector erase command (ISP, IAP, commercial programmer) or a 'global' erase command (commercial programmer).
2	EDISx	Erase Disable ISP. Disables the ability to perform an erase of sector x in ISP or IAP mode. When programmed, this bit and sector x can only be erased by a 'global' erase command using a commercial programmer. This bit and sector x CANNOT be erased in ISP or IAP modes.
3:7	-	reserved

Table 105: Effects of Security Bits

EDISx	SPEDISx	MOVCDISx	Effects on Programming
0	0	0	None.
0	0	1	Security violation flag set for sector CRC calculation for the specific sector. Security violation flag set for global CRC calculation if any MOVCDISx bit is set. Cycle aborted. Memory contents unchanged. CRC invalid. Program/erase commands will not result in a security violation.
0	1	x	Security violation flag set for program commands or an erase page command. Cycle aborted. Memory contents unchanged. Sector erase and global erase are allowed.
1	x	x	Security violation flag set for program commands or an erase page command. Cycle aborted. Memory contents unchanged. Global erase is allowed.

18.19 Boot Vector register

Table 106: Boot Vector (BOOTVEC) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	-	-	-	BOOTV4	BOOTV3	BOOTV2	BOOTV1	BOOTV0
Factory default value	0	0	0	1	1	1	1	1

Table 107: Boot Vector (BOOTVEC) bit description

Bit	Symbol	Description
0:4	BOOTV[0:4]	Boot vector. If the Boot Vector is selected as the reset address, the P89LPC932A1 will start execution at an address comprised of 00h in the lower eight bits and this BOOTVEC as the upper eight bits after a reset.
5:7	-	reserved

18.20 Boot status register

Table 108: Boot Status (BOOTSTAT) bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	DCCP	CWP	AWP	-	-	-	--	BSB
Factory default value	0	0	0	0	0	0	0	1

Table 109: Boot Status (BOOTSTAT) bit description

Bit	Symbol	Description
0	BSB	Boot Status Bit. If programmed to logic 1, the P89LPC932A1 will always start execution at an address comprised of 00H in the lower eight bits and BOOTVEC as the upper bits after a reset. (See Section 6.1 "Reset vector").
1:4	-	reserved

Table 109: Boot Status (BOOTSTAT) bit description *…continued*

Bit	Symbol	Description
5	AWP	Activate Write Protection bit. When this bit is cleared, the internal Write Enable flag is forced to the set state, thus writes to the flash memory are always enabled. When this bit is set, the Write Enable internal flag can be set or cleared using the Set Write Enable (SWE) or Clear Write Enable (CWE) commands.
6	CWP	Configuration Write Protect bit. Protects inadvertent writes to the user programmable configuration bytes (UCFG1, BOOTVEC, and BOOTSTAT). If programmed to a logic 1, the writes to these registers are disabled. If programmed to a logic 0, writes to these registers are enabled. This bit is set by programming the BOOTSTAT register. This bit is cleared by writing the Clear Configuration Protection (CCP) command to FMCON followed by writing 96H to FMDATA.
7	DCCP	Disable Clear Configuration Protection command. If Programmed to ‘1’, the Clear Configuration Protection (CCP) command is disabled during ISP or IAP modes. This command can still be used in ICP or parallel programmer modes. If programmed to ‘0’, the CCP command can be used in all programming modes. This bit is set by programming the BOOTSTAT register. This bit is cleared by writing the Clear Configuration Protection (CCP) command in either ICP or parallel programmer modes.

19. Instruction set

Table 110: Instruction set summary

Mnemonic	Description	Bytes	Cycles	Hex code
	ARITHMETIC			
ADD A,Rn	Add register to A	1	1	28 to 2F
ADD A,dir	Add direct byte to A	2	1	25
ADD A,@Ri	Add indirect memory to A	1	1	26 to 27
ADD A,#data	Add immediate to A	2	1	24
ADDC A,Rn	Add register to A with carry	1	1	38 to 3F
ADDC A,dir	Add direct byte to A with carry	2	1	35
ADDC A,@Ri	Add indirect memory to A with carry	1	1	36 to 37
ADDC A,#data	Add immediate to A with carry	2	1	34
SUBB A,Rn	Subtract register from A with borrow	1	1	98 to 9F
SUBB A,dir	Subtract direct byte from A with borrow	2	1	95
SUBB A,@Ri	Subtract indirect memory from A with borrow	1	1	96 to 97
SUBB A,#data	Subtract immediate from A with borrow	2	1	94
INC A	Increment A	1	1	04
INC Rn	Increment register	1	1	08 to 0F
INC dir	Increment direct byte	2	1	05
INC @Ri	Increment indirect memory	1	1	06 to 07
DEC A	Decrement A	1	1	14
DEC Rn	Decrement register	1	1	18 to 1F
DEC dir	Decrement direct byte	2	1	15
DEC @Ri	Decrement indirect memory	1	1	16 to 17
INC DPTR	Increment data pointer	1	2	A3
MUL AB	Multiply A by B	1	4	A4
DIV AB	Divide A by B	1	4	84
DA A	Decimal Adjust A	1	1	D4
	LOGICAL			
ANL A,Rn	AND register to A	1	1	58 to 5F
ANL A,dir	AND direct byte to A	2	1	55
ANL A,@Ri	AND indirect memory to A	1	1	56 to 57
ANL A,#data	AND immediate to A	2	1	54
ANL dir,A	AND A to direct byte	2	1	52
ANL dir,#data	AND immediate to direct byte	3	2	53
ORL A,Rn	OR register to A	1	1	48 to 4F
ORL A,dir	OR direct byte to A	2	1	45

Table 110: Instruction set summary *...continued*

Mnemonic	Description	Bytes	Cycles	Hex code
ORL A,@Ri	OR indirect memory to A	1	1	46 to 47
ORL A,#data	OR immediate to A	2	1	44
ORL dir,A	OR A to direct byte	2	1	42
ORL dir,#data	OR immediate to direct byte	3	2	43
XRL A,Rn	Exclusive-OR register to A	1	1	68 to 6F
XRL A,dir	Exclusive-OR direct byte to A	2	1	65
XRL A, @Ri	Exclusive-OR indirect memory to A	1	1	66 to 67
XRL A,#data	Exclusive-OR immediate to A	2	1	64
XRL dir,A	Exclusive-OR A to direct byte	2	1	62
XRL dir,#data	Exclusive-OR immediate to direct byte	3	2	63
CLR A	Clear A	1	1	E4
CPL A	Complement A	1	1	F4
SWAP A	Swap Nibbles of A	1	1	C4
RL A	Rotate A left	1	1	23
RLC A	Rotate A left through carry	1	1	33
Rotate A right	RR A	1	1	03
RRC A	Rotate A right through carry	1	1	13
	DATA TRANSFER			
MOV A,Rn	Move register to A	1	1	E8 to EF
MOV A,dir	Move direct byte to A	2	1	E5
Move indirect memory to A	MOV A,@Ri	1	1	E6 to E7
MOV A,#data	Move immediate to A	2	1	74
MOV Rn,A	Move A to register	1	1	F8 to FF
MOV Rn,dir	Move direct byte to register	2	2	A8 to AF
MOV Rn,#data	Move immediate to register	2	1	78 to 7F
MOV dir,A	Move A to direct byte	2	1	F5
MOV dir,Rn	Move register to direct byte	2	2	88 to 8F
MOV dir,dir	Move direct byte to direct byte	3	2	85
MOV dir,@Ri	Move indirect memory to direct byte	2	2	86 to 87
MOV dir,#data	Move immediate to direct byte	3	2	75
MOV @Ri,A	Move A to indirect memory	1	1	F6 to F7
MOV @Ri,dir	Move direct byte to indirect memory	2	2	A6 to A7
MOV @Ri,#data	Move immediate to indirect memory	2	1	76 to 77
MOV DPTR,#data	Move immediate to data pointer	3	2	90
MOVC A,@A+DPTR	Move code byte relative DPTR to A	1	2	93
MOVC A,@A+PC	Move code byte relative PC to A	1	2	94

Table 110: Instruction set summary *...continued*

Mnemonic	Description	Bytes	Cycles	Hex code
MOVX A,@Ri	Move external data(A8) to A	1	2	E2 to E3
MOVX A,@DPTR	Move external data(A16) to A	1	2	E0
MOVX @Ri,A	Move A to external data(A8)	1	2	F2 to F3
MOVX @DPTR,A	Move A to external data(A16)	1	2	F0
PUSH dir	Push direct byte onto stack	2	2	C0
POP dir	Pop direct byte from stack	2	2	D0
XCH A,Rn	Exchange A and register	1	1	C8 to CF
XCH A,dir	Exchange A and direct byte	2	1	C5
XCH A,@Ri	Exchange A and indirect memory	1	1	C6 to C7
XCHD A,@Ri	Exchange A and indirect memory nibble	1	1	D6 to D7
	BOOLEAN			
Mnemonic	Description	Bytes	Cycles	Hex code
CLR C	Clear carry	1	1	C3
CLR bit	Clear direct bit	2	1	C2
SETB C	Set carry	1	1	D3
SETB bit	Set direct bit	2	1	D2
CPL C	Complement carry	1	1	B3
CPL bit	Complement direct bit	2	1	B2
ANL C,bit	AND direct bit to carry	2	2	82
ANL C,/bit	AND direct bit inverse to carry	2	2	B0
ORL C,bit	OR direct bit to carry	2	2	72
ORL C,/bit	OR direct bit inverse to carry	2	2	A0
MOV C,bit	Move direct bit to carry	2	1	A2
MOV bit,C	Move carry to direct bit	2	2	92
	BRANCHING			
ACALL addr 11	Absolute jump to subroutine	2	2	116F1
LCALL addr 16	Long jump to subroutine	3	2	12
RET	Return from subroutine	1	2	22
RETI	Return from interrupt	1	2	32
AJMP addr 11	Absolute jump unconditional	2	2	016E1
LJMP addr 16	Long jump unconditional	3	2	02
SJMP rel	Short jump (relative address)	2	2	80
JC rel	Jump on carry = 1	2	2	40
JNC rel	Jump on carry = 0	2	2	50
JB bit,rel	Jump on direct bit = 1	3	2	20
JNB bit,rel	Jump on direct bit = 0	3	2	30
JBC bit,rel	Jump on direct bit = 1 and clear	3	2	10
JMP @A+DPTR	Jump indirect relative DPTR	1	2	73

Table 110: Instruction set summary *…continued*

Mnemonic	Description	Bytes	Cycles	Hex code
JZ rel	Jump on accumulator = 0	2	2	60
JNZ rel	Jump on accumulator ≠ 0	2	2	70
CJNE A,dir,rel	Compare A, direct jne relative	3	2	B5
CJNE A,#d,rel	Compare A, immediate jne relative	3	2	B4
CJNE Rn,#d,rel	Compare register, immediate jne relative	3	2	B8 to BF
CJNE @Ri,#d,rel	Compare indirect, immediate jne relative	3	2	B6 to B7
DJNZ Rn,rel	Decrement register, jnz relative	2	2	D8 to DF
DJNZ dir,rel	Decrement direct byte, jnz relative	3	2	D5
	MISCELLANEOUS			
NOP	No operation	1	1	00

20. Disclaimers

Life support — These products are not designed for use in life support appliances, devices, or systems where malfunction of these products can reasonably be expected to result in personal injury. Philips Semiconductors customers using or selling these products for use in such applications do so at their own risk and agree to fully indemnify Philips Semiconductors for any damages resulting from such application.

Right to make changes — Philips Semiconductors reserves the right to make changes in the products - including circuits, standard cells, and/or software - described or contained herein in order to improve design and/or performance. When the product is in full production (status 'Production'), relevant changes will be communicated via a Customer Product/Process Change Notification (CPCN). Philips Semiconductors assumes no responsibility or liability for the use of any of these products, conveys no licence or title under any patent, copyright, or mask work right to these products, and makes no representations or warranties that these products are free from patent, copyright, or mask work right infringement, unless otherwise specified.

Application information — Applications that are described herein for any of these products are for illustrative purposes only. Philips Semiconductors make no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

21. Trademarks

Notice — All referenced brands, product names, service names and trademarks are the property of their respective owners.
I^2C-bus (logo) — is a trademark of Philips Semicondoctors.

22. Contents

continued >>

Date of release: 23 May 2005

Published in the Netherlands