DATA SHEET

P89C660/P89C662/P89C664/P89C668

80C51 8-bit Flash microcontroller family

16KB/32KB/64KB ISP/IAP FLASH with 512B/1KB/2KB/8KB RAM

Product data

Replaces P89C660/P89C662/P89C664 of 2001 Jul 19
and P89C668 of 2001 Jul 27

2002 Oct 28

Philips
Semiconductors

PHILIPS

80C51 8-bit Flash microcontroller family
16KB/32KB/64KB ISP/IAP Flash with 512B/1KB/2KB/8KB RAM

P89C660/P89C662/P89C664/ P89C668

DESCRIPTION

The P89C660/662/664/668 device contains a non-volatile 16KB/32KB/64KB Flash program memory that is both parallel programmable and serial In-System and In-Application Programmable. In-System Programming (ISP) allows the user to download new code while the microcontroller sits in the application. In-Application Programming (IAP) means that the microcontroller fetches new program code and reprograms itself while in the system. This allows for remote programming over a modem link. A default serial loader (boot loader) program in ROM allows serial In-System Programming of the Flash memory via the UART without the need for a loader in the Flash code. For In-Application Programming, the user program erases and reprograms the Flash memory by use of standard routines contained in ROM.

This device executes one instruction in 6 clock cycles, hence providing twice the speed of a conventional 80C51. An OTP configuration bit gives the user the option to select conventional 12-clock timing.

This device is a Single-Chip 8-Bit Microcontroller manufactured in advanced CMOS process and is a derivative of the 80C51 microcontroller family. The instruction set is 100% executing and timing compatible with the 80C51 instruction set.

The device also has four 8-bit I/O ports, three 16-bit timer/event counters, a multi-source, four-priority-level, nested interrupt structure, an enhanced UART and on-chip oscillator and timing circuits.

The added features of the P89C660/662/664/668 makes it a powerful microcontroller for applications that require pulse width modulation, high-speed I/O and up/down counting capabilities such as motor control.

FEATURES

- 80C51 Central Processing Unit
- On-chip Flash program memory with In-System Programming (ISP) and In-Application Programming (IAP) capability
- Boot ROM contains low level Flash programming routines for downloading via the UART
- Can be programmed by the end-user application (IAP)
- Parallel programming with 87C51 compatible hardware interface to programmer
- Six clocks per machine cycle operation (standard)
- 12 clocks per machine cycle operation (optional)
- Speed up to 20 MHz with 6 clock cycles per machine cycle (40 MHz equivalent performance); up to 33 MHz with 12 clocks per machine cycle
- Fully static operation
- RAM externally expandable to 64 kbytes
- Four interrupt priority levels
- Eight interrupt sources
- Four 8-bit I/O ports
- Full-duplex enhanced UART
  - Framing error detection
  - Automatic address recognition
- Power control modes
  - Clock can be stopped and resumed
  - Idle mode
  - Power-Down mode
- Programmable clock out
- Second DPTR register
- Asynchronous port reset
- Low EMI (inhibit ALE)
- I^2C serial interface
- Programmable Counter Array (PCA)
  - PWM
  - Capture/compare
- Well-suited for IPMI applications

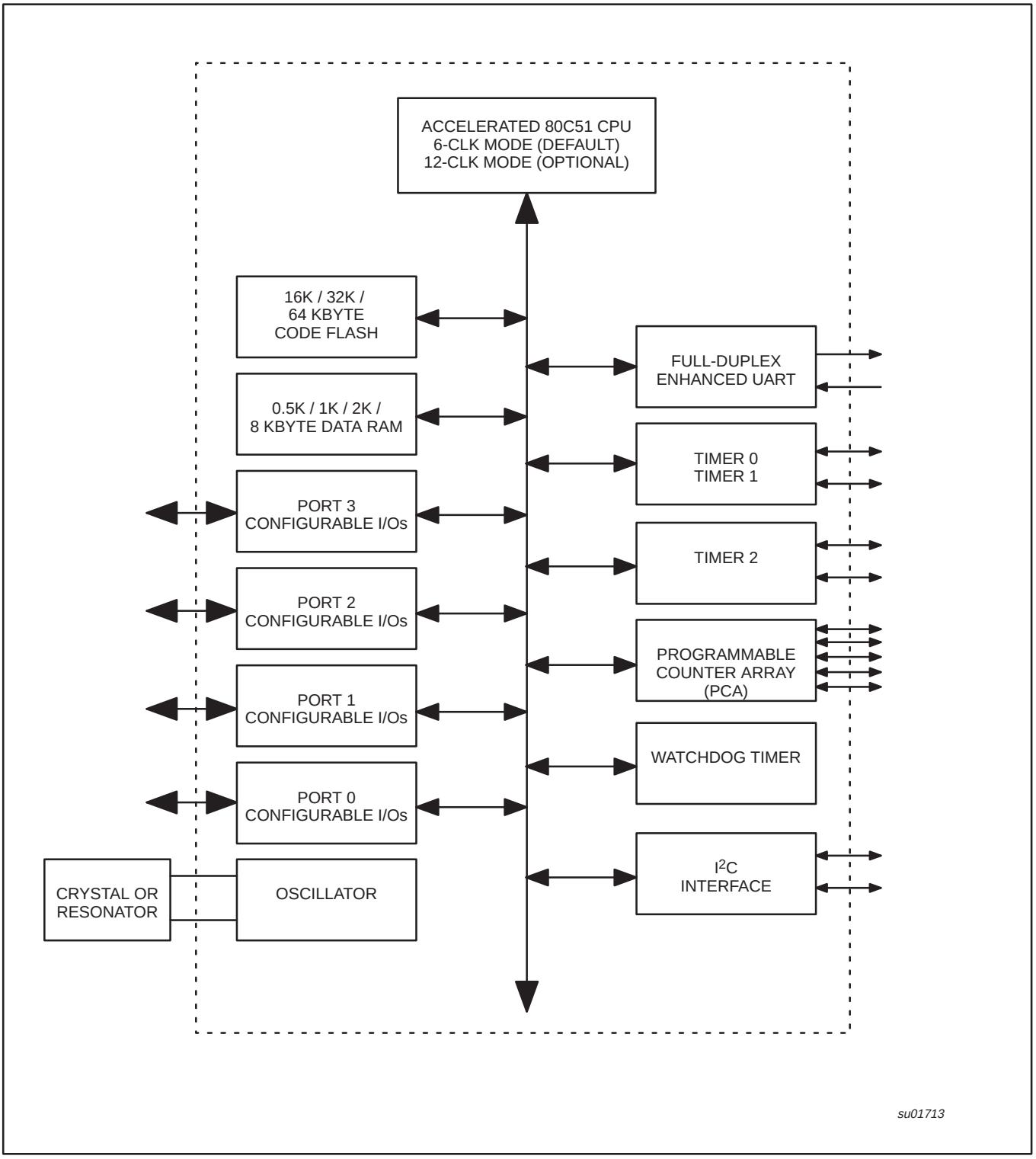
SELECTION TABLE

Type	Memory				Timers				Serial Interfaces				ADC bits/ch.	I/O Pins	Interrupts (External)	Program Security	Default Clock Rate	Optional Clock Rate	Reset active low/high?	Max. Freq. at 6-clk / 12-clk (MHz)	Freq. Range at 3V (MHz)	Freq. Range at 5V (MHz)
	RAM	ROM	OTP	Flash	# of Timers	PWM	PCA	WD	UART	I²C	CAN	SPI										
P89C668	8K	–	–	64K	4	√	√	√	√	√	–	–	–	32	8(2)/4	√	6-clk	12-clk	H	20/33	–	0-20/33
P89C664	2K	–	–	64K	4	√	√	√	√	√	–	–	–	32	8(2)/4	√	6-clk	12-clk	H	20/33	–	0-20/33
P89C662	1K	–	–	32K	4	√	√	√	√	√	–	–	–	32	8(2)/4	√	6-clk	12-clk	H	20/33	–	0-20/33
P89C660	512B	–	–	16K	4	√	√	√	√	√	–	–	–	32	8(2)/4	√	6-clk	12-clk	H	20/33	–	0-20/33

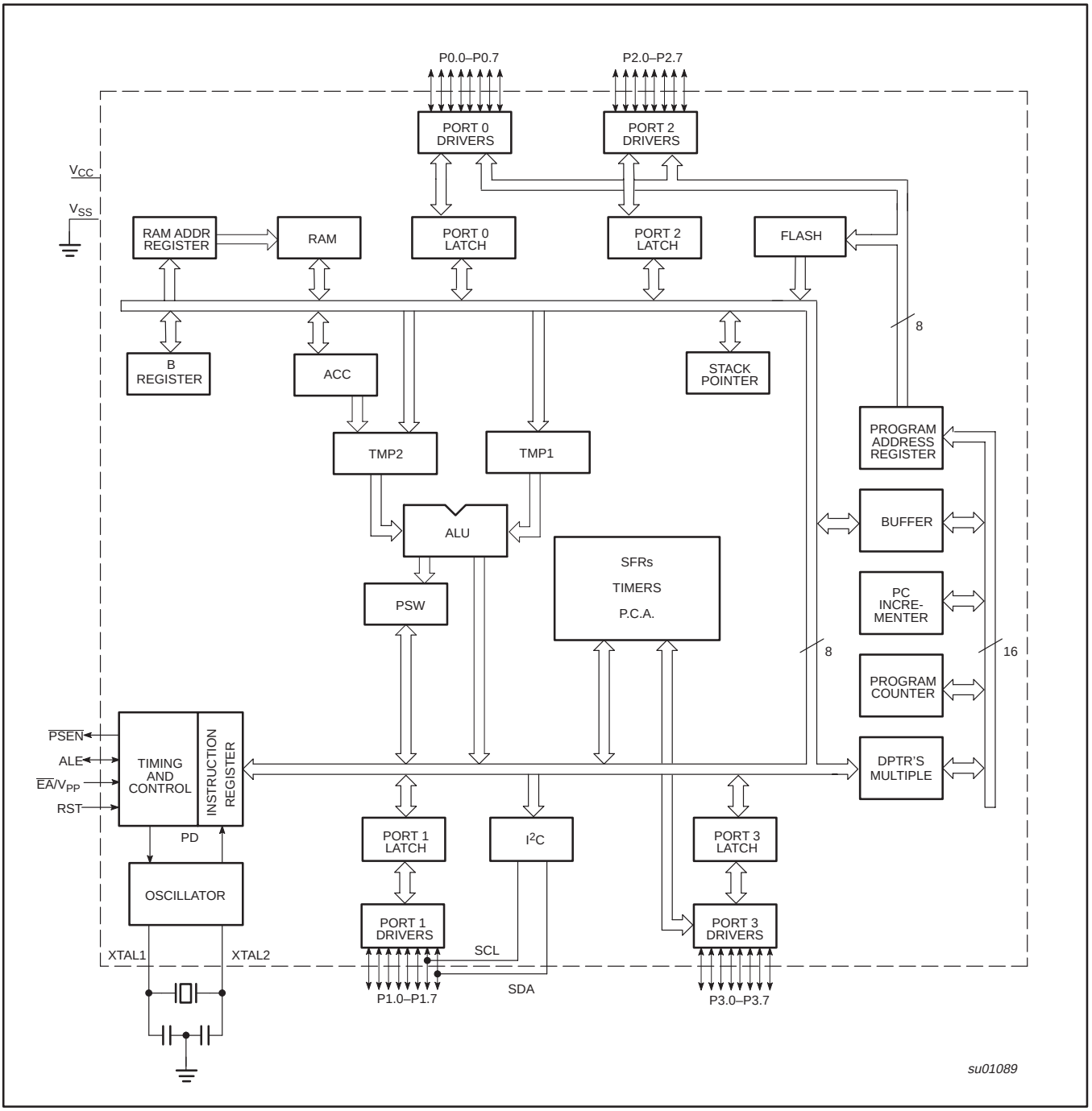
ORDERING INFORMATION

DEVICE	MEMORY		TEMPERATURE RANGE (°C) AND PACKAGE	VOLTAGE RANGE	FREQUENCY (MHz)		DWG #
	FLASH	RAM			6 CLOCK MODE	12 CLOCK MODE	
P89C660HBA	16 KB	512 B	0 to +70, PLCC	4.5–5.5 V	0 to 20 MHz	0 to 33 MHz	SOT187-2
P89C660HFA	16 KB	512 B	–40 to +85, PLCC	4.75–5.25 V	0 to 20 MHz	0 to 33 MHz	SOT187-2
P89C660HBBD	16 KB	512 B	0 to +70, LQFP	4.5–5.5 V	0 to 20 MHz	0 to 33 MHz	SOT389-1
P89C662HBA	32 KB	1 KB	0 to +70, PLCC	4.5–5.5 V	0 to 20 MHz	0 to 33 MHz	SOT187-2
P89C662HFA	32 KB	1 KB	–40 to +85, PLCC	4.75–5.25 V	0 to 20 MHz	0 to 33 MHz	SOT187-2
P89C662HBBD	32 KB	1 KB	0 to +70, LQFP	4.5–5.5 V	0 to 20 MHz	0 to 33 MHz	SOT389-1
P89C662HFBD	32 KB	1 KB	–40 to +85, LQFP	4.75–5.25 V	0 to 20 MHz	0 to 33 MHz	SOT389-1
P89C664HBA	64 KB	2 KB	0 to +70, PLCC	4.5–5.5 V	0 to 20 MHz	0 to 33 MHz	SOT187-2
P89C664HFA	64 KB	2 KB	–40 to +85, PLCC	4.75–5.25 V	0 to 20 MHz	0 to 33 MHz	SOT187-2
P89C664HBBD	64 KB	2 KB	0 to +70, LQFP	4.5–5.5 V	0 to 20 MHz	0 to 33 MHz	SOT389-1
P89C664HFBD	64 KB	2 KB	–40 to +85, LQFP	4.75–5.25 V	0 to 20 MHz	0 to 33 MHz	SOT389-1
P89C668HBA	64 KB	8 KB	0 to +70, PLCC	4.5–5.5 V	0 to 20 MHz	0 to 33 MHz	SOT187-2
P89C668HFA	64 KB	8 KB	–40 to +85, PLCC	4.5–5.5 V	0 to 20 MHz	0 to 33 MHz	SOT187-2
P89C668HBBD	64 KB	8 KB	0 to +70, LQFP	4.5–5.5 V	0 to 20 MHz	0 to 33 MHz	SOT389-1

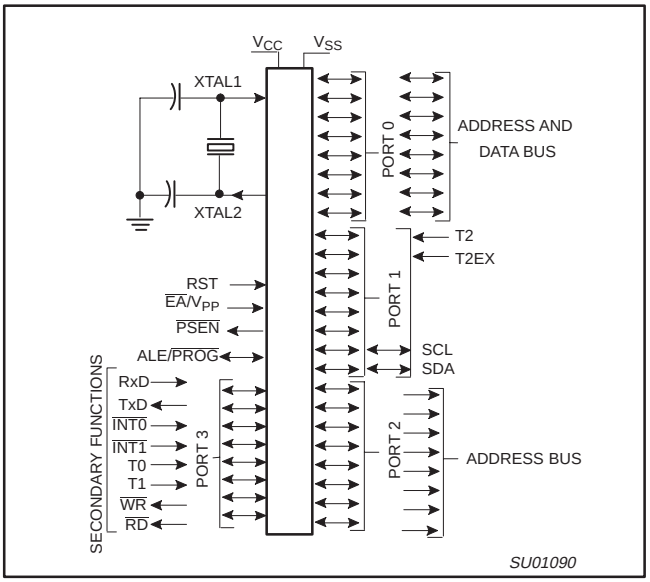
BLOCK DIAGRAM 1



BLOCK DIAGRAM (CPU-ORIENTED)

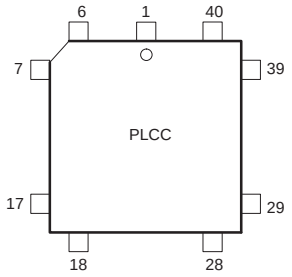


LOGIC SYMBOL



PINNING

Plastic Leaded Chip Carrier

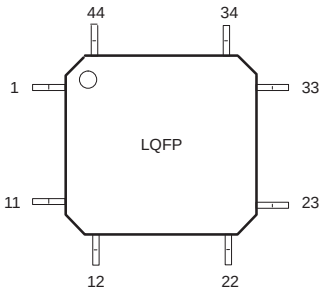


Pin	Function	Pin	Function	Pin	Function
1	NIC*	16	P3.4/T0/CEX3	31	P2.7/A15
2	P1.0/T2	17	P3.5/T1/CEX4	32	$\overline{PSEN}$
3	P1.1/T2EX	18	P3.6/$\overline{WR}$	33	ALE
4	P1.2/ECI	19	P3.7/$\overline{RD}$	34	NIC*
5	P1.3/CEX0	20	XTAL2	35	$\overline{EA}$/V_{PP}
6	P1.4/CEX1	21	XTAL1	36	P0.7/AD7
7	P1.5/CEX2	22	V_{SS}	37	P0.6/AD6
8	P1.6/SCL	23	NIC*	38	P0.5/AD5
9	P1.7/SDA	24	P2.0/A8	39	P0.4/AD4
10	RST	25	P2.1/A9	40	P0.3/AD3
11	P3.0/RxD	26	P2.2/A10	41	P0.2/AD2
12	NIC*	27	P2.3/A11	42	P0.1/AD1
13	P3.1/TxD	28	P2.4/A12	43	P0.0/AD0
14	P3.2/$\overline{INT0}$	29	P2.5/A13	44	V_{CC}
15	P3.3/$\overline{INT1}$	30	P2.6/A14		

* NO INTERNAL CONNECTION

SU01091

Low Quad Flat Pack



Pin	Function	Pin	Function	Pin	Function
1	P1.5/CEX2	16	V_{SS}	31	P0.6/AD6
2	P1.6/SCL	17	NIC*	32	P0.5/AD5
3	P1.7/SDA	18	P2.0/A8	33	P0.4/AD4
4	RST	19	P2.1/A9	34	P0.3/AD3
5	P3.0/RxD	20	P2.2/A10	35	P0.2/AD2
6	NIC*	21	P2.3/A11	36	P0.1/AD1
7	P3.1/TxD	22	P2.4/A12	37	P0.0/AD0
8	P3.2/$\overline{INT0}$	23	P2.5/A13	38	V_{CC}
9	P3.3/$\overline{INT1}$	24	P2.6/A14	39	NIC*
10	P3.4/T0/CEX3	25	P2.7/A15	40	P1.0/T2
11	P3.5/T1/CEX4	26	$\overline{PSEN}$	41	P1.1/T2EX
12	P3.6/$\overline{WR}$	27	ALE	42	P1.2/ECI
13	P3.7/$\overline{RD}$	28	NIC*	43	P1.3/CEX0
14	XTAL2	29	$\overline{EA}$/V_{PP}	44	P1.4/CEX1
15	XTAL1	30	P0.7/AD7		

* NO INTERNAL CONNECTION

SU01401

PIN DESCRIPTIONS

MNEMONIC	PIN NUMBER		TYPE	NAME AND FUNCTION
	PLCC	LQFP		
V_{SS}	22	16	I	**Ground:** 0 V reference.
V_{CC}	44	38	I	**Power Supply:** This is the power supply voltage for normal, idle, and power-down operation.
P0.0–0.7	43–36	37–30	I/O	**Port 0:** Port 0 is an open-drain, bidirectional I/O port. Port 0 pins that have 1s written to them float and can be used as high-impedance inputs. Port 0 is also the multiplexed low-order address and data bus during accesses to external program and data memory. In this application, it uses strong internal pull-ups when emitting 1s.
P1.0–P1.7	2–9	40–44, 1–3	I/O	**Port 1:** Port 1 is an 8-bit bidirectional I/O port with internal pull-ups on all pins except P1.6 and P1.7 which are open drain. Port 1 pins that have 1s written to them are pulled high by the internal pull-ups and can be used as inputs. As inputs, port 1 pins that are externally pulled low will source current because of the internal pull-ups. (See DC Electrical Characteristics: I_{IL}).
				Alternate functions for P89C660/662/664/668 Port 1 include:
	2	40	I/O	**T2 (P1.0):** Timer/Counter 2 external count input/Clockout (see Programmable Clock-Out)
	3	41	I	**T2EX (P1.1):** Timer/Counter 2 Reload/Capture/Direction Control
	4	42	I	**ECI (P1.2):** External Clock Input to the PCA
	5	43	I/O	**CEX0 (P1.3):** Capture/Compare External I/O for PCA module 0
	6	44	I/O	**CEX1 (P1.4):** Capture/Compare External I/O for PCA module 1
	7	1	I/O	**CEX2 (P1.5):** Capture/Compare External I/O for PCA module 2
	8	2	I/O	**SCL (P1.6):** I^2C bus clock line (open drain)
	9	3	I/O	**SDA (P1.7):** I^2C bus data line (open drain)
P2.0–P2.7	24–31	18–25	I/O	**Port 2:** Port 2 is an 8-bit bidirectional I/O port with internal pull-ups. Port 2 pins that have 1s written to them are pulled high by the internal pull-ups and can be used as inputs. As inputs, port 2 pins that are externally being pulled low will source current because of the internal pull-ups. (See DC Electrical Characteristics: I_{IL}). Port 2 emits the high-order address byte during fetches from external program memory and during accesses to external data memory that use 16-bit addresses (MOVX @DPTR). In this application, it uses strong internal pull-ups when emitting 1s. During accesses to external data memory that use 8-bit addresses (MOV @Ri), port 2 emits the contents of the P2 special function register.
P3.0–P3.7	11, 13–19	5, 7–13	I/O	**Port 3:** Port 3 is an 8-bit bidirectional I/O port with internal pull-ups. Port 3 pins that have 1s written to them are pulled high by the internal pull-ups and can be used as inputs. As inputs, port 3 pins that are externally being pulled low will source current because of the pull-ups. (See DC Electrical Characteristics: I_{IL}). Port 3 also serves the special features of the P89C660/662/664/668, as listed below:
	11	5	I	**RxD (P3.0):** Serial input port
	13	7	O	**TxD (P3.1):** Serial output port
	14	8	I	**$\overline{INT0}$ (P3.2):** External interrupt
	15	9	I	**$\overline{INT1}$ (P3.3):** External interrupt
	16	10	I	**CEX3/T0 (P3.4):** Timer 0 external input; Capture/Compare External I/O for PCA module 3
	17	11	I	**CEX4/T1 (P3.5):** Timer 1 external input; Capture/Compare External I/O for PCA module 4
	18	12	O	**$\overline{WR}$ (P3.6):** External data memory write strobe
	19	13	O	**$\overline{RD}$ (P3.7):** External data memory read strobe
RST	10	4	I	**Reset:** A high on this pin for two machine cycles while the oscillator is running, resets the device. An internal resistor to V_{SS} permits a power-on reset using only an external capacitor to V_{CC}.
ALE	33	27	O	**Address Latch Enable:** Output pulse for latching the low byte of the address during an access to external memory. In normal operation, ALE is emitted twice every machine cycle, and can be used for external timing or clocking. Note that one ALE pulse is skipped during each access to external data memory. ALE can be disabled by setting SFR auxiliary.0. With this bit set, ALE will be active only during a MOVX instruction.
$\overline{PSEN}$	32	26	O	**Program Store Enable:** The read strobe to external program memory. When executing code from the external program memory, $\overline{PSEN}$ is activated twice each machine cycle, except that two $\overline{PSEN}$ activations are skipped during each access to external data memory. $\overline{PSEN}$ is not activated during fetches from internal program memory.

MNEMONIC	PIN NUMBER		TYPE	NAME AND FUNCTION
	PLCC	LQFP		
$\overline{EA}/V_{PP}$	35	29	I	**External Access Enable/Programming Supply Voltage:** $\overline{EA}$ must be externally held low to enable the device to fetch code from external program memory locations. If $\overline{EA}$ is held high, the device executes from internal program memory. The value on the $\overline{EA}$ pin is latched when RST is released and any subsequent changes have no effect. This pin also receives the programming supply voltage (V_{PP}) during Flash programming.
XTAL1	21	15	I	**Crystal 1:** Input to the inverting oscillator amplifier and input to the internal clock generator circuits.
XTAL2	20	14	O	**Crystal 2:** Output from the inverting oscillator amplifier.

NOTE:
To avoid “latch-up” effect at power-on, the voltage on any pin (other than V_{PP}) must not be higher than V_{CC} + 0.5 V or less than V_{SS} – 0.5 V.

Table 1. Special Function Registers

SYMBOL	DESCRIPTION	DIRECT ADDRESS	BIT ADDRESS, SYMBOL, OR ALTERNATIVE PORT FUNCTION MSB							LSB	RESET VALUE
ACC*	Accumulator	E0H	E7	E6	E5	E4	E3	E2	E1	E0	00H
AUXR#	Auxiliary	8EH	–	–	–	–	–	–	EXTRAM	AO	xxxxxx10B
AUXR1#	Auxiliary 1	A2H	–	–	ENBOOT	–	GF2	0	–	DPS	xxxxx0x0B
B*	B register	F0H	F7	F6	F5	F4	F3	F2	F1	F0	00H
CCAP0H#	Module 0 Capture High	FAH									xxxxxxxxB
CCAP1H#	Module 1 Capture High	FBH									xxxxxxxxB
CCAP2H#	Module 2 Capture High	FCH									xxxxxxxxB
CCAP3H#	Module 3 Capture High	FDH									xxxxxxxxB
CCAP4H#	Module 4 Capture High	FEH									xxxxxxxxB
CCAP0L#	Module 0 Capture Low	EAH									xxxxxxxxB
CCAP1L#	Module 1 Capture Low	EBH									xxxxxxxxB
CCAP2L#	Module 2 Capture Low	ECH									xxxxxxxxB
CCAP3L#	Module 3 Capture Low	EDH									xxxxxxxxB
CCAP4L#	Module 4 Capture Low	EEH									xxxxxxxxB
CCAPM0#	Module 0 Mode	C2H	–	ECOM	CAPP	CAPN	MAT	TOG	PWM	ECCF	x0000000B
CCAPM1#	Module 1 Mode	C3H	–	ECOM	CAPP	CAPN	MAT	TOG	PWM	ECCF	x0000000B
CCAPM2#	Module 2 Mode	C4H	–	ECOM	CAPP	CAPN	MAT	TOG	PWM	ECCF	x0000000B
CCAPM3#	Module 3 Mode	C5H	–	ECOM	CAPP	CAPN	MAT	TOG	PWM	ECCF	x0000000B
CCAPM4#	Module 4 Mode	C6H	–	ECOM	CAPP	CAPN	MAT	TOG	PWM	ECCF	x0000000B
			C7	C6	C5	C4	C3	C2	C1	C0	
CCON*#	PCA Counter Control	C0H	CF	CR	–	CCF4	CCF3	CCF2	CCF1	CCF0	00x00000B
CH#	PCA Counter High	F9H									00H
CL#	PCA Counter Low	E9H									00H
CMOD#	PCA Counter Mode	C1H	CIDL	WDTE	–	–	–	CPS1	CPS0	ECF	00xxx000B
DPTR:	Data Pointer (2 bytes)										
DPH	Data Pointer High	83H									00H
DPL	Data Pointer Low	82H									00H
			AF	AE	AD	AC	AB	AA	A9	A8	
IEN0*	Interrupt Enable 0	A8H	EA	EC	ES1	ES0	ET1	EX1	ET0	EX0	00H
IEN1*	Interrupt Enable 1	E8	–	–	–	–	–	–	–	ET2	xxxxxxx0B
			BF	BE	BD	BC	BB	BA	B9	B8	
IP*	Interrupt Priority	B8H	PT2	PPC	PS1	PS0	PT1	PX1	PT0	PX0	x0000000B
IPH#	Interrupt Priority High	B7H	PT2H	PPCH	PS1H	PS0H	PT1H	PX1H	PT0H	PX0H	x0000000B
			87	86	85	84	83	82	81	80	
P0*	Port 0	80H	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0	FFH
			97	96	95	94	93	92	91	90	
P1*	Port 1	90H	SDA	SCL	CEX2	CEX1	CEX0	ECI	T2EX	T2	FFH
			A7	A6	A5	A4	A3	A2	A1	A0	
P2*	Port 2	A0H	AD15	AD14	AD13	AD12	AD11	AD10	AD9	AD8	FFH
			B7	B6	B5	B4	B3	B2	B1	B0	
P3*	Port 3	B0H	$\overline{RD}$	$\overline{WR}$	T1/ CEX4	T0/ CEX3	$\overline{INT1}$	$\overline{INT0}$	TxD	RxD	FFH
PCON#[1]	Power Control	87H	SMOD1	SMOD0	–	POF	GF1	GF0	PD	IDL	00xxx000B

* SFRs are bit addressable.
\# SFRs are modified from or added to the 80C51 SFRs.
– Reserved bits.
1. Reset value depends on reset source.

Table 1 Special Function Registers (Continued)

SYMBOL	DESCRIPTION	DIRECT ADDRESS	BIT ADDRESS, SYMBOL, OR ALTERNATIVE PORT FUNCTION MSB							LSB	RESET VALUE
			D7	D6	D5	D4	D3	D2	D1	D0	
PSW*	Program Status Word	D0H	CY	AC	F0	RS1	RS0	OV	F1	P	00000000B
RCAP2H#	Timer 2 Capture High	CBH									00H
RCAP2L#	Timer 2 Capture Low	CAH									00H
SADDR#	Slave Address	A9H									00H
SADEN#	Slave Address Mask	B9H									00H
S0BUF	Serial Data Buffer	99H									xxxxxxxxB
			9F	9E	9D	9C	9B	9A	99	98	
S0CON*	Serial Control	98H	SM0/FE	SM1	SM2	REN	TB8	RB8	TI	RI	00H
SP	Stack Pointer	81H									07H
S1DAT#	Serial 1 Data	DAH									00H
S1ADR#	Serial 1 Address	DBH	SLAVE ADDRESS							GC	00H
S1STA#	Serial 1 Status	D9H	SC4	SC3	SC2	SC1	SC0	0	0	0	F8H
			DF	DE	DD	DC	DB	DA	D9	D8	
S1CON*#	Serial 1 Control	D8H	CR2	ENS1	STA	STO	SI	AA	CR1	CR0	00000000B
			8F	8E	8D	8C	8B	8A	89	88	
TCON*	Timer Control	88H	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0	00H
			CF	CE	CD	CC	CB	CA	C9	C8	
T2CON*	Timer 2 Control	C8H	TF2	EXF2	RCLK	TCLK	EXEN2	TR2	C/$\overline{T2}$	CP/$\overline{RL2}$	00H
T2MOD#	Timer 2 Mode Control	C9H	–	–	–	–	–	–	T2OE	DCEN	xxxxxx00B
TH0	Timer High 0	8CH									00H
TH1	Timer High 1	8DH									00H
TH2#	Timer High 2	CDH									00H
TL0	Timer Low 0	8AH									00H
TL1	Timer Low 1	8BH									00H
TL2#	Timer Low 2	CCH									00H
TMOD	Timer Mode	89H	GATE	C/$\overline{T}$	M1	M0	GATE	C/$\overline{T}$	M1	M0	00H
WDTRST	Watchdog Timer Reset	A6H									

* SFRs are bit addressable.
\# SFRs are modified from or added to the 80C51 SFRs.
– Reserved bits.

OSCILLATOR CHARACTERISTICS

XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier. The pins can be configured for use as an on-chip oscillator.

To drive the device from an external clock source, XTAL1 should be driven while XTAL2 is left unconnected. Minimum and maximum high and low times specified in the data sheet must be observed.

This device is configured at the factory to operate using 6 clock periods per machine cycle, referred to in this datasheet as “6 clock mode”. (This yields performance equivalent to twice that of standard 80C51 family devices). It may be optionally configured on commercially-available EPROM programming equipment to operate at 12 clock periods per machine cycle, referred to in this datasheet as “12 clock mode”. Once 12 clock mode has been configured, it cannot be changed back to 6 clock mode.

RESET

A reset is accomplished by holding the RST pin high for at least two machine cycles (12 oscillator periods in 6 clock mode, or 24 oscillator periods in 12 clock mode), while the oscillator is running. To insure a good power-on reset, the RST pin must be high long enough to allow the oscillator time to start up (normally a few milliseconds) plus two machine cycles. At power-on, the voltage on V_{CC} and RST must come up at the same time for a proper start-up. Ports 1, 2, and 3 will asynchronously be driven to their reset condition when a voltage above V_{IH1} (min.) is applied to RST.

The value on the $\overline{EA}$ pin is latched when RST is deasserted and has no further effect.

LOW POWER MODES

Stop Clock Mode

The static design enables the clock speed to be reduced down to 0 MHz (stopped). When the oscillator is stopped, the RAM and Special Function Registers retain their values. This mode allows step-by-step utilization and reduces system power consumption by lowering the clock frequency down to any value. For lowest power consumption the Power-Down mode is suggested.

Idle Mode

In the idle mode (see Table 2), the CPU puts itself to sleep while all of the on-chip peripherals stay active. The instruction to invoke the idle mode is the last instruction executed in the normal operating mode before the idle mode is activated. The CPU contents, the on-chip RAM, and all of the special function registers remain intact during this mode. The idle mode can be terminated either by any enabled interrupt (at which time the process is picked up at the interrupt service routine and continued), or by a hardware reset which starts the processor in the same manner as a power-on reset.

Power-Down Mode

To save even more power, a Power-Down mode (see Table 2) can be invoked by software. In this mode, the oscillator is stopped and the instruction that invoked Power-Down is the last instruction executed. The on-chip RAM and Special Function Registers retain their values down to 2.0 V and care must be taken to return V_{CC} to the minimum specified operating voltages before the Power-Down mode is terminated.

Either a hardware reset or external interrupt can be used to exit from Power-Down. Reset redefines all the SFRs but does not change the on-chip RAM. An external interrupt allows both the SFRs and the on-chip RAM to retain their values.

To properly terminate Power-Down the reset or external interrupt should not be executed before V_{CC} is restored to its normal operating level and must be held active long enough for the oscillator to restart and stabilize (normally less than 10ms).

With an external interrupt, INT0 and INT1 must be enabled and configured as level-sensitive. Holding the pin low restarts the oscillator, but bringing the pin back high completes the exit. Once the interrupt is serviced, the next instruction to be executed after RETI will be the one following the instruction that put the device into Power-Down.

POWER-ON FLAG

The Power-On Flag (POF) is set by on-chip circuitry when the V_{CC} level on the P89C660/662/664/668 rises from 0 to 5 V. The POF bit can be set or cleared by software allowing a user to determine if the reset is the result of a power-on or a warm start after Power-Down. The V_{CC} level must remain above 3 V for the POF to remain unaffected by the V_{CC} level.

Design Consideration

When the idle mode is terminated by a hardware reset, the device normally resumes program execution, from where it left off, up to two machine cycles before the internal reset algorithm takes control. On-chip hardware inhibits access to internal RAM in this event, however, access to the port pins is not inhibited. To eliminate the possibility of an unexpected write when the idle mode is terminated by reset, the instruction following the one that invokes the idle mode should not be one that writes to a port pin or to external memory.

ONCE™ Mode

The ONCE (“On-Circuit Emulation”) mode facilitates testing and debugging of systems without the device having to be removed from the circuit. The ONCE mode is invoked by:

1. Pulling ALE low while the device is in reset and $\overline{PSEN}$ is high;
2. Holding ALE low as RST is deactivated.

While the device is in ONCE mode, the Port 0 pins go into a float state, and the other port pins and ALE and $\overline{PSEN}$ are weakly pulled high. The oscillator circuit remains active. While the device is in this mode, an emulator or test CPU can be used to drive the circuit. Normal operation is restored when a normal reset is applied.

Programmable Clock-Out

A 50% duty cycle clock can be programmed to come out on P1.0. This pin, besides being a regular I/O pin, has two alternate functions. It can be programmed:

1. to input the external clock for Timer/Counter 2, or
2. to output a 50% duty cycle clock ranging from 122 Hz to 8 MHz at a 16 MHz operating frequency (61 Hz to 4 MHz in 12 clock mode).

To configure the Timer/Counter 2 as a clock generator, bit C/$\overline{T}$2 (in T2CON) must be cleared and bit T20E in T2MOD must be set. Bit TR2 (T2CON.2) also must be set to start the timer.

The Clock-Out frequency depends on the oscillator frequency and the reload value of Timer 2 capture registers (RCAP2H, RCAP2L) as shown in this equation:

$$\frac{\text{Oscillator Frequency}}{n \times (65536 - \text{RCAP2H, RCAP2L})}$$

n = 2 in 6 clock mode
4 in 12 clock mode

Where (RCAP2H,RCAP2L) = the content of RCAP2H and RCAP2L taken as a 16-bit unsigned integer.

In the Clock-Out mode Timer 2 roll-overs will not generate an interrupt. This is similar to when it is used as a baud-rate generator. It is possible to use Timer 2 as a baud-rate generator and a clock generator simultaneously. Note, however, that the baud-rate and the Clock-Out frequency will be the same.

Table 2. External Pin Status During Idle and Power-Down mode

MODE	PROGRAM MEMORY	ALE	$\overline{PSEN}$	PORT 0	PORT 1	PORT 2	PORT 3
Idle	Internal	1	1	Data	Data	Data	Data
Idle	External	1	1	Float	Data	Address	Data
Power-Down	Internal	0	0	Data	Data	Data	Data
Power-Down	External	0	0	Float	Data	Data	Data

I²C SERIAL COMMUNICATION — SIO1

The I²C serial port is identical to the I²C serial port on the 8XC554, 8XC654, and 8XC652 devices.

Note that the P89C660/662/664/668 I²C pins are alternate functions to port pins P1.6 and P1.7. Because of this, P1.6 and P1.7 on these parts do not have a pull-up structure as found on the 80C51. Therefore P1.6 and P1.7 have open drain outputs on the P89C660/662/664/668.

The I²C bus uses two wires (SDA and SCL) to transfer information between devices connected to the bus. The main features of the bus are:

– 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

The output latches of P1.6 and P1.7 must be set to logic 1 in order to enable SIO1.

The P89C66x on-chip I²C logic provides a serial interface that meets the I²C bus specification and supports all transfer modes (other than the low-speed mode) from and to the I²C bus. The SIO1 logic handles bytes transfer autonomously. It also keeps track of serial transfers, and a status register (S1STA) reflects the status of SIO1 and the I²C bus.

The CPU interfaces to the I²C logic via the following four special function registers: S1CON (SIO1 control register), S1STA (SIO1 status register), S1DAT (SIO1 data register), and S1ADR (SIO1 slave address register). The SIO1 logic interfaces to the external I²C bus via two port 1 pins: P1.6/SCL (serial clock line) and P1.7/SDA (serial data line).

A typical I²C bus configuration is shown in Figure 1. Figure 2 shows how a data transfer is accomplished on the bus. Depending on the state of the direction bit (R/W), two types of data transfers are possible on the I²C bus:

1. 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.
2. 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.

Modes of Operation

The on-chip SIO1 logic may operate in the following four modes:

1. Master Transmitter mode:

   Serial data output through P1.7/SDA while P1.6/SCL outputs the serial clock. The first transmitted byte contains the slave address of the receiving device (7 bits) and the data direction bit. In this mode the data direction bit (R/$\overline{W}$) will be logic 0, and we say that a “W” is transmitted. Thus the first byte transmitted is SLA+W. Serial 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.

2. Master Receiver Mode:

   The first transmitted byte contains the slave address of the transmitting device (7 bits) and the data direction bit. In this mode the data direction bit (R/$\overline{W}$) will be logic 1, and we say that an “R” is transmitted. Thus the first byte transmitted is SLA+R. Serial data is received via P1.7/SDA while P1.6/SCL outputs the serial clock. Serial data is received 8 bits at a time. After each byte is received, an acknowledge bit is transmitted. START and STOP conditions are output to indicate the beginning and end of a serial transfer.

3. Slave Receiver mode:

   Serial data and the serial clock are received through P1.7/SDA and P1.6/SCL. After each byte is received, an acknowledge bit is transmitted. START and STOP conditions are recognized as the beginning and end of a serial transfer. Address recognition is performed by hardware after reception of the slave address and direction bit.

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 P1.7/SDA while the serial clock is input through P1.6/SCL. START and STOP conditions are recognized as the beginning and end of a serial transfer.

In a given application, SIO1 may operate as a master and as a slave. In the Slave mode, the SIO1 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 microcontroller 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, SIO1 switches to the Slave mode immediately and can detect its own slave address in the same serial transfer.

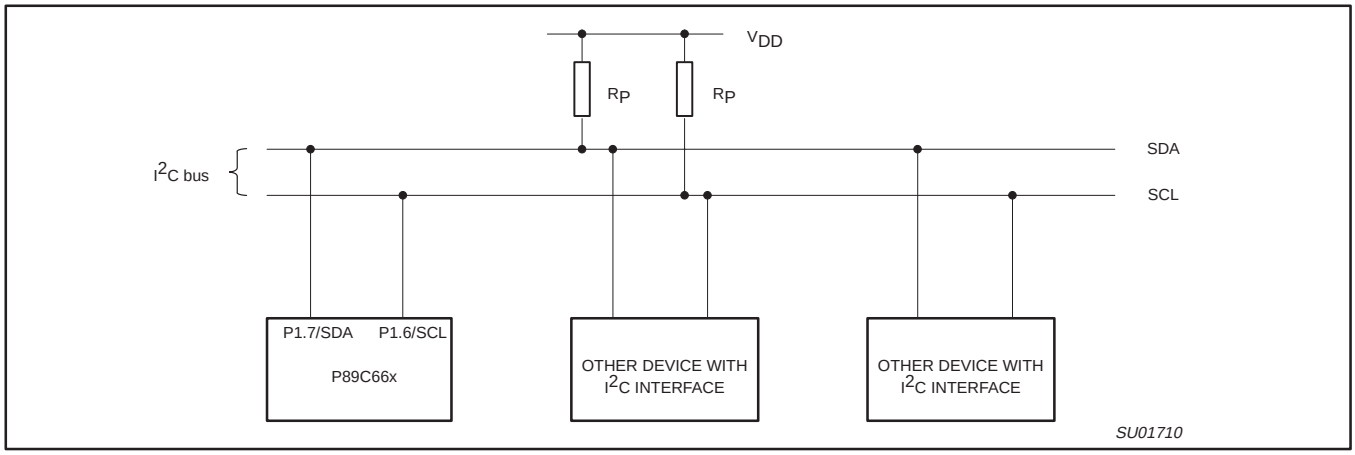


Figure 1. Typical I²C Bus Configuration

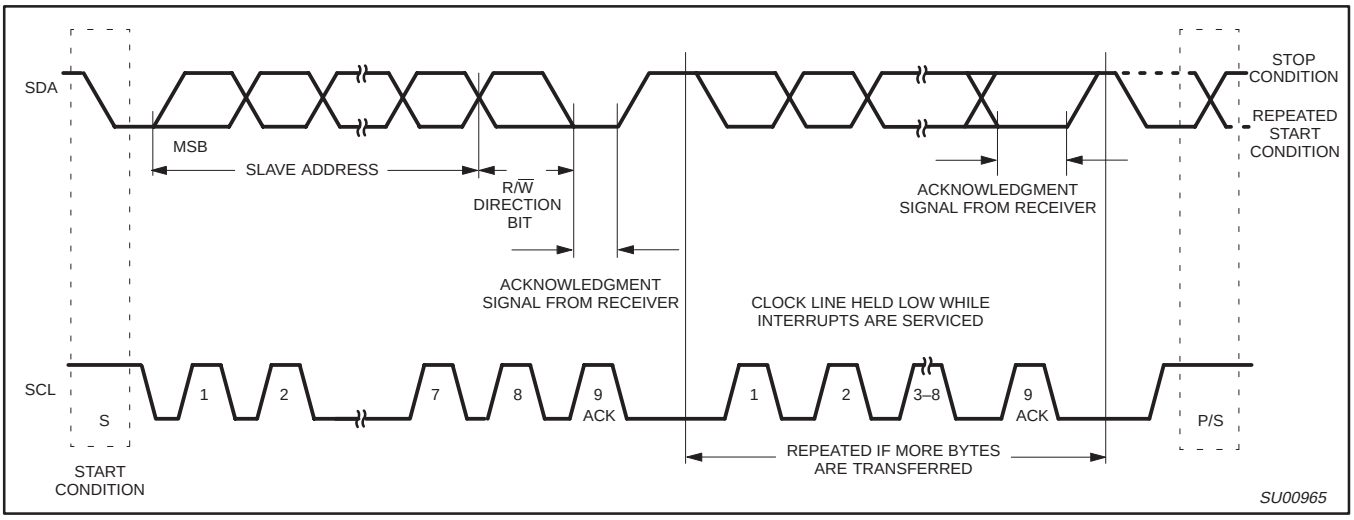


Figure 2. Data Transfer on the I²C Bus

SIO1 Implementation and Operation

Figure 3 shows how the on-chip I²C bus interface is implemented, and the following text describes the individual blocks.

Input Filters and Output Stages

The input filters have I²C compatible input levels. If the input voltage is less than 1.5 V, the input logic level is interpreted as 0; if the input voltage is greater than 3.0 V, the input logic level is interpreted as 1. Input signals are synchronized with the internal clock ($f_{OSC}/4$), and spikes shorter than three oscillator periods are filtered out.

The output stages consist of open drain transistors that can sink 3mA at V_{OUT} < 0.4 V. These open drain outputs do not have clamping diodes to V_{DD}. Thus, if the device is connected to the I²C bus and V_{DD} is switched off, the I²C bus is not affected.

Address Register, S1ADR

This 8-bit special function register may be loaded with the 7-bit slave address (7 most significant bits) to which SIO1 will respond when programmed as a slave transmitter or receiver. The LSB (GC) is used to enable general call address (00H) recognition.

Comparator

The comparator compares the received 7-bit slave address with its own slave address (7 most significant bits in S1ADR). It also compares the first received 8-bit byte with the general call address (00H). If an equality is found, the appropriate status bits are set and an interrupt is requested.

Shift Register, S1DAT

This 8-bit special function register contains a byte of serial data to be transmitted or a byte which has just been received. Data in S1DAT 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 S1DAT. While data is being shifted out, data on the bus is simultaneously being shifted in; S1DAT always contains the last byte present on the bus. Thus, in the event of lost arbitration, the transition from master transmitter to slave receiver is made with the correct data in S1DAT.

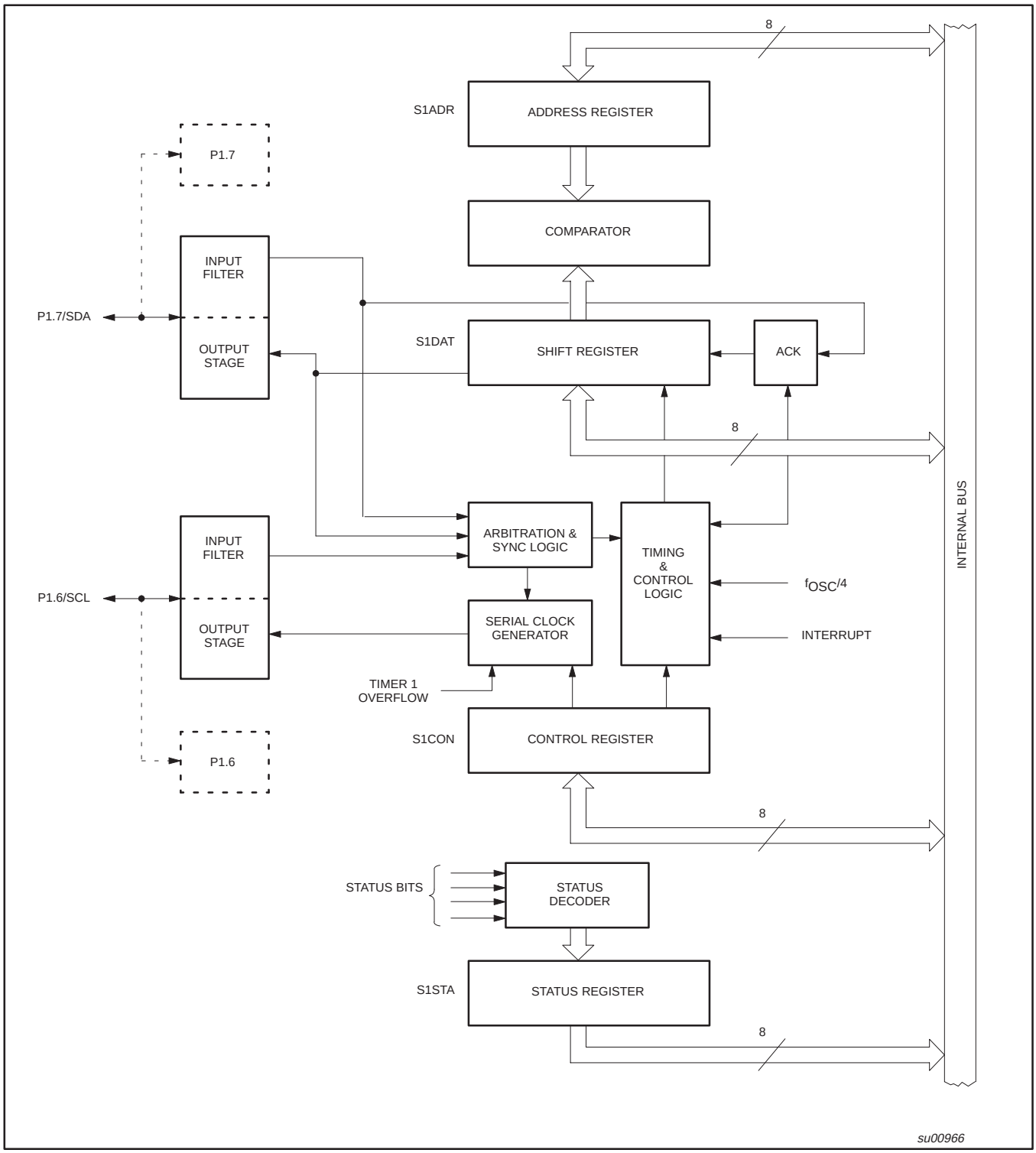


Figure 3. I²C Bus Serial Interface Block Diagram

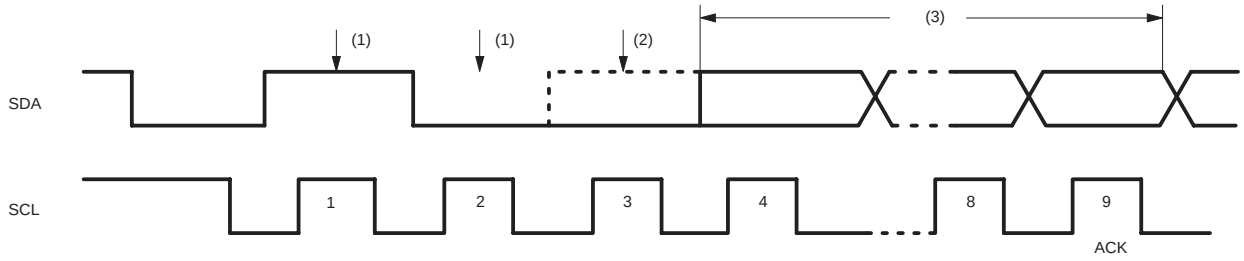
Arbitration and Synchronization Logic

In the Master Transmitter mode, the arbitration logic checks that every transmitted logic 1 actually appears as a logic 1 on the I^2C bus. If another device on the bus overrules a logic 1 and pulls the SDA line low, arbitration is lost, and SIO1 immediately changes from master transmitter to slave receiver. SIO1 will continue to output clock pulses (on SCL) until transmission of the current serial byte is complete.

Arbitration may also be lost in the Master Receiver mode. Loss of arbitration in this mode can only occur while SIO1 is returning a “not acknowledge: (logic 1) to the bus. Arbitration is lost when another device on the bus pulls this signal LOW. Since this can occur only at the end of a serial byte, SIO1 generates no further clock pulses. Figure 4 shows the arbitration procedure.

The synchronization logic will synchronize the serial clock generator with the clock pulses on the SCL line from another device. If two or more master devices generate clock pulses, the “mark” duration is determined by the device that generates the shortest “marks,” and the “space” duration is determined by the device that generates the longest “spaces.” Figure 5 shows the synchronization procedure.

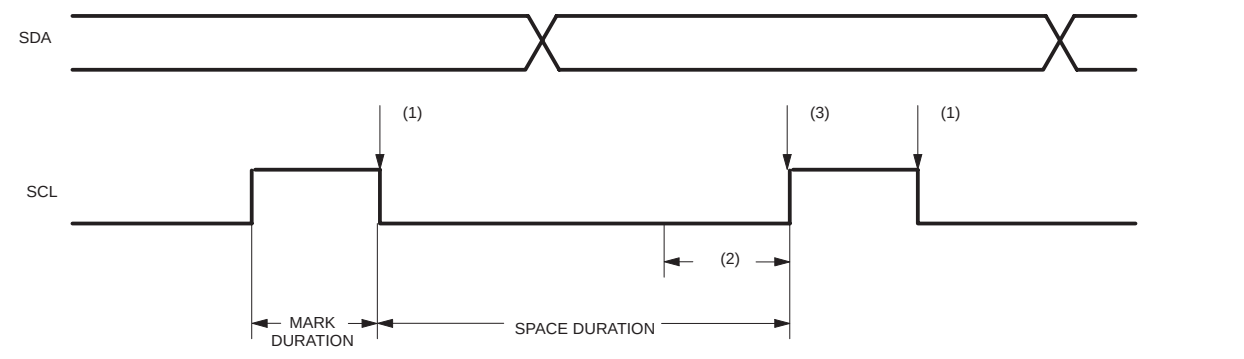
A slave may stretch the space duration to slow down the bus master. The space duration may also be stretched for handshaking purposes. This can be done after each bit or after a complete byte transfer. SIO1 will stretch the SCL space duration after a byte has been transmitted or received and the acknowledge bit has been transferred. The serial interrupt flag (SI) is set, and the stretching continues until the serial interrupt flag is cleared.



1. Another device transmits identical serial data.
2. Another device overrules a logic 1 (dotted line) transmitted by SIO1 (master) by pulling the SDA line low. Arbitration is lost, and SIO1 enters the slave receiver mode.
3. SIO1 is in the slave receiver mode but still generates clock pulses until the current byte has been transmitted. SIO1 will not generate clock pulses for the next byte. Data on SDA originates from the new master once it has won arbitration.

SU00967

Figure 4. Arbitration Procedure



1. Another service pulls the SCL line low before the SIO1 “mark” duration is complete. The serial clock generator is immediately reset and commences with the “space” duration by pulling SCL low.
2. Another device still pulls the SCL line low after SIO1 releases SCL. The serial clock generator is forced into the wait state until the SCL line is released.
3. The SCL line is released, and the serial clock generator commences with the mark duration.

SU00968

Figure 5. Serial Clock Synchronization

Serial Clock Generator
This programmable clock pulse generator provides the SCL clock pulses when SIO1 is in the Master Transmitter or Master Receiver mode. It is switched off when SIO1 is in a Slave mode. The programmable output clock frequencies are: $f_{OSC}/120$, $f_{OSC}/9600$ (12-clock mode) or $f_{OSC}/60$, $f_{OSC}/4800$ (6-clock mode) and the Timer 1 overflow rate divided by eight. The output clock pulses have a 50% duty cycle unless the clock generator is synchronized with other SCL clock sources as described above.

Timing and Control
The timing and control logic generates the timing and control signals for serial byte handling. This logic block provides the shift pulses for S1DAT, enables the comparator, generates and detects start and stop conditions, receives and transmits acknowledge bits, controls the master and Slave modes, contains interrupt request logic, and monitors the I^2C bus status.

Control Register, S1CON
This 7-bit special function register is used by the microcontroller to control the following SIO1 functions: start and restart of a serial transfer, termination of a serial transfer, bit rate, address recognition, and acknowledgment.

Status Decoder and Status Register
The status decoder takes all of the internal status bits and compresses them into a 5-bit code. This code is unique for each I^2C bus status. The 5-bit code may be used to generate vector addresses for fast processing of the various service routines. Each service routine processes a particular bus status. There are 26 possible bus states if all four modes of SIO1 are used. The 5-bit status code is latched into the five most significant bits of the status register when the serial interrupt flag is set (by hardware) and remains stable until the interrupt flag is cleared by software. The three least significant bits of the status register are always zero. If the status code is used as a vector to service routines, then the routines are displaced by eight address locations. Eight bytes of code is sufficient for most of the service routines.

The Four SIO1 Special Function Registers

The microcontroller interfaces to SIO1 via four special function registers. These four SFRs (S1ADR, S1DAT, S1CON, and S1STA) are described individually in the following sections.

The Address Register, S1ADR
The CPU can read from and write to this 8-bit, directly addressable SFR. S1ADR is not affected by the SIO1 hardware. The contents of this register are irrelevant when SIO1 is in a Master mode. In the Slave modes, the seven most significant bits must be loaded with the microcontroller's own slave address, and, if the least significant bit is set, the general call address (00H) is recognized; otherwise it is ignored.

	7	6	5	4	3	2	1	0
S1ADR (DBH)	X	X	X	X	X	X	X	GC
	own slave address							

The most significant bit corresponds to the first bit received from the I^2C bus after a start condition. A logic 1 in S1ADR corresponds to a high level on the I^2C bus, and a logic 0 corresponds to a low level on the bus.

The Data Register, S1DAT
S1DAT contains a byte of serial data to be transmitted or a byte which has just been received. The CPU can read from and write to this 8-bit, directly addressable SFR while it is not in the process of shifting a byte. This occurs when SIO1 is in a defined state and the serial interrupt flag is set. Data in S1DAT remains stable as long as SI is set. Data in S1DAT 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 S1DAT. While data is being shifted out, data on the bus is simultaneously being shifted in; S1DAT always contains the last data byte present on the bus. Thus, in the event of lost arbitration, the transition from master transmitter to slave receiver is made with the correct data in S1DAT.

	7	6	5	4	3	2	1	0
S1DAT (DAH)	SD7	SD6	SD5	SD4	SD3	SD2	SD1	SD0
	shift direction							

SD7 - SD0:
Eight bits to be transmitted or just received. A logic 1 in S1DAT corresponds to a high level on the I^2C bus, and a logic 0 corresponds to a low level on the bus. Serial data shifts through S1DAT from right to left. Figure 6 shows how data in S1DAT is serially transferred to and from the SDA line.

S1DAT and the ACK flag form a 9-bit shift register which shifts in or shifts out an 8-bit byte, followed by an acknowledge bit. The ACK flag is controlled by the SIO1 hardware and cannot be accessed by the CPU. Serial data is shifted through the ACK flag into S1DAT on the rising edges of serial clock pulses on the SCL line. When a byte has been shifted into S1DAT, the serial data is available in S1DAT, and the acknowledge bit is returned by the control logic during the ninth clock pulse. Serial data is shifted out from S1DAT via a buffer (BSD7) on the falling edges of clock pulses on the SCL line.

When the CPU writes to S1DAT, BSD7 is loaded with the content of S1DAT.7, which is the first bit to be transmitted to the SDA line (see Figure 7). After nine serial clock pulses, the eight bits in S1DAT will have been transmitted to the SDA line, and the acknowledge bit will be present in ACK. Note that the eight transmitted bits are shifted back into S1DAT.

The Control Register, S1CON
The CPU can read from and write to this 8-bit, directly addressable SFR. Two bits are affected by the SIO1 hardware: the SI bit is set when a serial interrupt is requested, and the STO bit is cleared when a STOP condition is present on the I^2C bus. The STO bit is also cleared when ENS1 = "0".

	7	6	5	4	3	2	1	0
S1CON (D8H)	CR2	ENS1	STA	STO	SI	AA	CR1	CR0

ENS1, the SIO1 Enable Bit: ENS1 = "0": When ENS1 is "0", the SDA and SCL outputs are in a high impedance state. SDA and SCL input signals are ignored, SIO1 is in the "not addressed" slave state, and the STO bit in S1CON is forced to "0". No other bits are affected. P1.6 and P1.7 may be used as open drain I/O ports.

ENS1 = "1": When ENS1 is "1", SIO1 is enabled. The P1.6 and P1.7 port latches must be set to logic 1.

ENS1 should not be used to temporarily release SIO1 from the I2C bus since, when ENS1 is reset, the I2C bus status is lost. The AA flag should be used instead (see description of the AA flag in the following text).

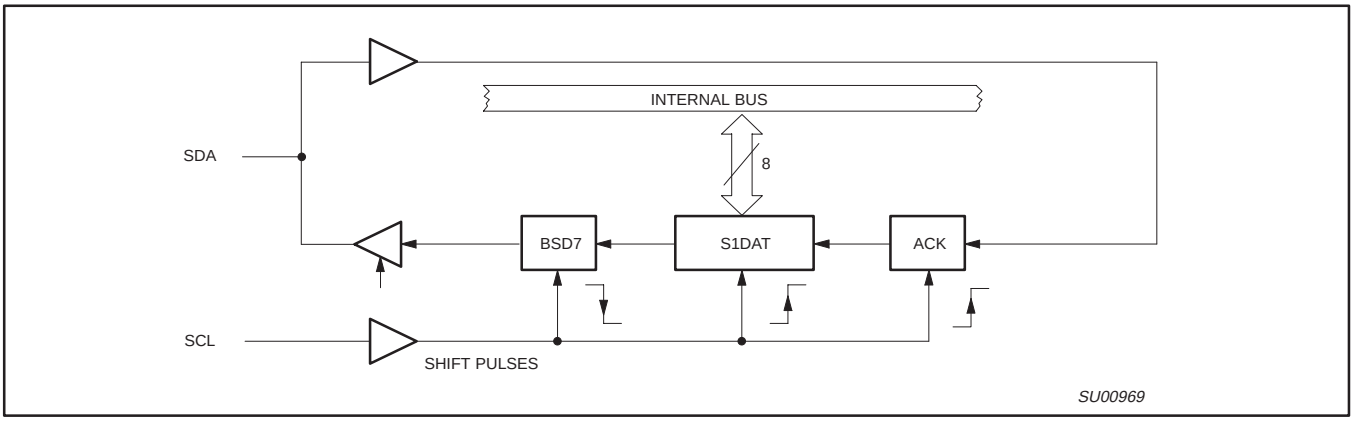


Figure 6. Serial Input/Output Configuration

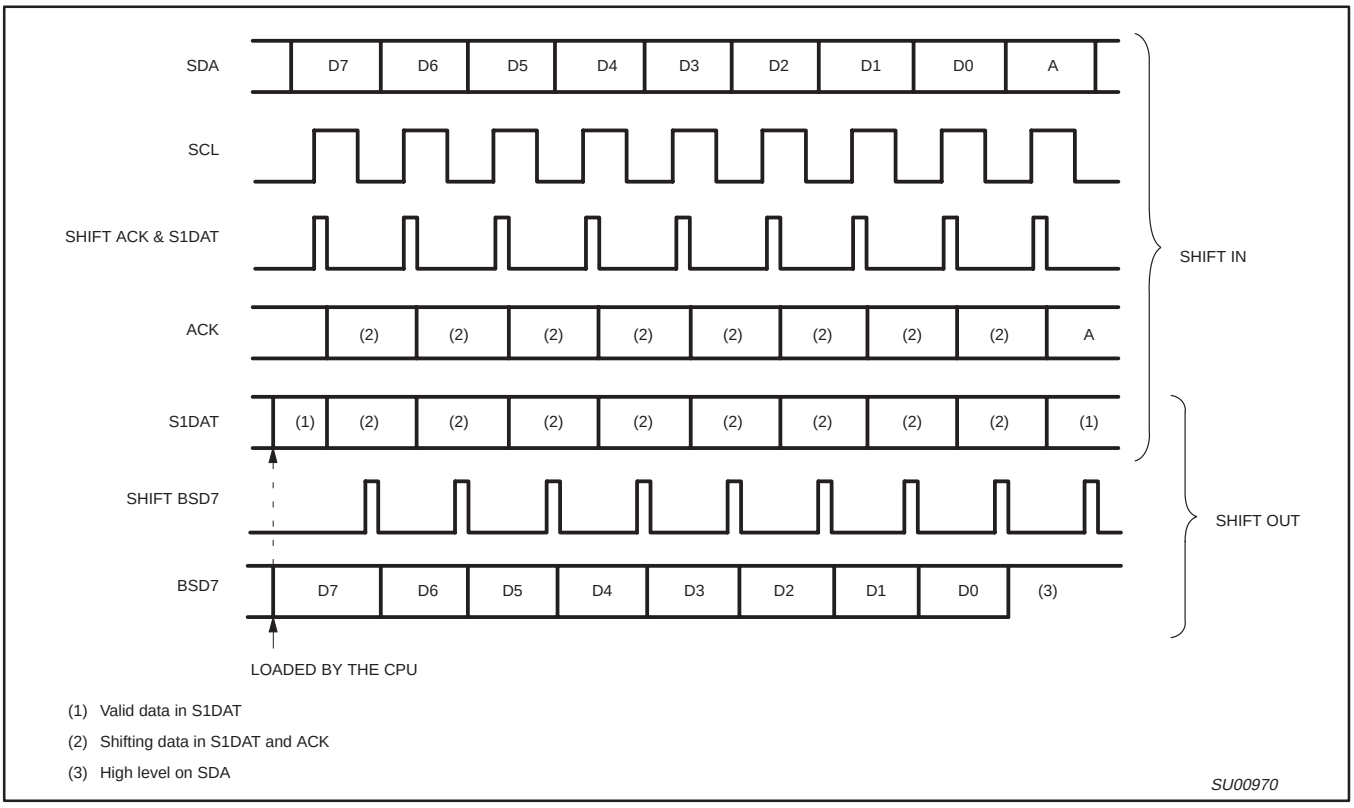


Figure 7. Shift-in and Shift-out Timing

In the following text, it is assumed that ENS1 = “1”.

The “START” Flag, STA: STA = “1”: When the STA bit is set to enter a Master mode, the SIO1 hardware checks the status of the I2C bus and generates a START condition if the bus is free. If the bus is not free, then SIO1 waits for a STOP condition (which will free the bus) and generates a START condition after a delay of half a clock period of the internal serial clock generator.

If STA is set while SIO1 is already in a Master mode and one or more bytes are transmitted or received, SIO1 transmits a repeated START condition. STA may be set at any time. STA may also be set when SIO1 is an addressed slave.

STA = “0”: When the STA bit is reset, no START condition or repeated START condition will be generated.

The STOP Flag, STO: STO = “1”: When the STO bit is set while SIO1 is in a Master mode, a STOP condition is transmitted to the I^2C bus. When the STOP condition is detected on the bus, the SIO1 hardware clears the STO flag. In a Slave mode, the STO flag may be set to recover from an error condition. In this case, no STOP condition is transmitted to the I^2C bus. However, the SIO1 hardware behaves as if a STOP condition has been received and switches to the defined “not addressed” Slave Receiver mode. The STO flag is automatically cleared by hardware.

If the STA and STO bits are both set, the a STOP condition is transmitted to the I^2C bus if SIO1 is in a Master mode (in a Slave mode, SIO1 generates an internal STOP condition which is not transmitted). SIO1 then transmits a START condition.

STO = “0”: When the STO bit is reset, no STOP condition will be generated.

The Serial Interrupt Flag, SI: SI = “1”: When the SI flag is set, then, if the EA and ES1 (interrupt enable register) bits are also set, a serial interrupt is requested. SI is set by hardware when one of 25 of the 26 possible SIO1 states is entered. The only state that does not cause SI to be set is state F8H, which indicates that no relevant state information is available.

While SI is set, the low period of the serial clock on the SCL line is stretched, and the serial transfer is suspended. A high level on the SCL line is unaffected by the serial interrupt flag. SI must be reset by software.

SI = “0”: When the SI flag is reset, no serial interrupt is requested, and there is no stretching of the serial clock on the SCL line.

The Assert Acknowledge Flag, AA: AA = “1”: If the AA flag is set, an acknowledge (low level to SDA) will be returned during the acknowledge clock pulse on the SCL line when:
– The “own slave address” has been received
– The general call address has been received while the general call bit (GC) in S1ADR is set
– A data byte has been received while SIO1 is in the Master Receiver mode
– A data byte has been received while SIO1 is in the addressed Slave Receiver mode

AA = “0”: if the AA flag is reset, a not acknowledge (high level to SDA) will be returned during the acknowledge clock pulse on SCL when:
– A data has been received while SIO1 is in the Master Receiver mode
– A data byte has been received while SIO1 is in the addressed Slave Receiver mode

When SIO1 is in the addressed Slave Transmitter mode, state C8H will be entered after the last serial is transmitted (see Figure 11).

When SI is cleared, SIO1 leaves state C8H, enters the not addressed Slave Receiver mode, and the SDA line remains at a high level. In state C8H, the AA flag can be set again for future address recognition.

When SIO1 is in the not addressed Slave mode, its own slave address and the general call address are ignored. Consequently, no acknowledge is returned, and a serial interrupt is not requested. Thus, SIO1 can be temporarily released from the I^2C bus while the bus status is monitored. While SIO1 is released from the bus, START and STOP conditions are detected, and serial data is shifted in. Address recognition can be resumed at any time by setting the AA flag. If the AA flag is set when the part’s own Slave address or the general call address has been partly received, the address will be recognized at the end of the byte transmission.

The Clock Rate Bits CR0, CR1, and CR2: These three bits determine the serial clock frequency when SIO1 is in a Master mode. The various serial rates are shown in Table 3.

A 12.5 kHz bit rate may be used by devices that interface to the I^2C bus via standard I/O port lines which are software driven and slow. 100 kHz is usually the maximum bit rate and can be derived from a 16 MHz, 12 MHz, or a 6 MHz oscillator. A variable bit rate (0.5 kHz to 62.5 kHz) may also be used if Timer 1 is not required for any other purpose while SIO1 is in a Master mode.

The frequencies shown in Table 3 are unimportant when SIO1 is in a Slave mode. In the Slave modes, SIO1 will automatically synchronize with any clock frequency up to 100 kHz.

The Status Register, S1STA

S1STA is an 8-bit read-only special function register. The three least significant bits are always zero. The five most significant bits contain the status code. There are 26 possible status codes. When S1STA contains F8H, no relevant state information is available and no serial interrupt is requested. All other S1STA values correspond to defined SIO1 states. When each of these states is entered, a serial interrupt is requested (SI = “1”). A valid status code is present in S1STA one machine cycle after SI is set by hardware and is still present one machine cycle after SI has been reset by software.

Table 3. Serial Clock Rates

6-clock mode								
			BIT FREQUENCY (kHz) AT f_{OSC}					
CR2	CR1	CR0	3 MHz	6 MHz	8 MHz	12 MHz[2]	15 MHz[2]	f_{OSC} DIVIDED BY
0	0	0	23	47	62.5	94	117[1]	128
0	0	1	27	54	71	107[1]	134[1]	112
0	1	0	31	63	83.3	125[1]	156[1]	96
0	1	1	37	75	100	150[1]	188[1]	80
1	0	0	6.25	12.5	17	25	31	480
1	0	1	50	100	133[1]	200[1]	250[1]	60
1	1	0	100	200	267[1]	400[1]	500[1]	30
1	1	1	0.24 < 62.5 0 < 255	0.49 < 62.5 0 < 254	0.65 < 55.6 0 < 253	0.98 < 50.0 0 < 251	1.22 < 52.1 0 < 250	48 × (256 – (reload value Timer 1)) Reload value Timer 1 in Mode 2.

12-clock mode								
			BIT FREQUENCY (kHz) AT f_{OSC}					
CR2	CR1	CR0	6 MHz	12 MHz	16 MHz	24 MHz[3]	30 MHz[3]	f_{OSC} DIVIDED BY
0	0	0	23	47	62.5	94	117[1]	256
0	0	1	27	54	71	107[1]	134[1]	224
0	1	0	31	63	83.3	125[1]	156[1]	192
0	1	1	37	75	100	150[1]	188[1]	160
1	0	0	6.25	12.5	17	25	31	960
1	0	1	50	100	133[1]	200[1]	250[1]	120
1	1	0	100	200	267[1]	400[1]	500[1]	60
1	1	1	0.24 < 62.5 0 < 255	0.49 < 62.5 0 < 254	0.65 < 55.6 0 < 253	0.98 < 50.0 0 < 251	1.22 < 52.1 0 < 250	96 × (256 – (reload value Timer 1)) Reload value Timer 1 in Mode 2.

NOTES:
1. These frequencies exceed the upper limit of 100 kHz of the I^2C-bus specification and cannot be used in an I^2C-bus application.
2. At f_{OSC} = 12 MHz/15 MHz the maximum I^2C bus rate of 100 kHz cannot be realized due to the fixed divider rates.
3. At f_{OSC} = 24 MHz/30 MHz the maximum I^2C bus rate of 100 kHz cannot be realized due to the fixed divider rates.

More Information on SIO1 Operating Modes

The four operating modes are:
– Master Transmitter
– Master Receiver
– Slave Receiver
– Slave Transmitter

Data transfers in each mode of operation are shown in Figures 8-11. These figures contain the following abbreviations:

Abbreviation	Explanation
S	Start condition
SLA	7-bit slave address
R	Read bit (high level at SDA)
W	Write bit (low level at SDA)
A	Acknowledge bit (low level at SDA)
$\overline{A}$	Not acknowledge bit (high level at SDA)
Data	8-bit data byte
P	Stop condition

In Figures 8-11, circles are used to indicate when the serial interrupt flag is set. The numbers in the circles show the status code held in the S1STA register. At these points, a service routine must be executed to continue or complete the serial transfer. These service routines are not critical since the serial transfer is suspended until the serial interrupt flag is cleared by software.

When a serial interrupt routine is entered, the status code in S1STA is used to branch to the appropriate service routine. For each status code, the required software action and details of the following serial transfer are given in Tables 4-8.

Master Transmitter mode

In the Master Transmitter mode, a number of data bytes are transmitted to a slave receiver (see Figure 8). Before the Master Transmitter mode can be entered, S1CON must be initialized as follows:

	7	6	5	4	3	2	1	0
S1CON (D8H)	CR2	ENS1	STA	STO	SI	AA	CR1	CR0
	bit rate	1	0	0	0	X	bit rate	

CR0, CR1, and CR2 define the serial bit rate. ENS1 must be set to logic 1 to enable SIO1. If the AA bit is reset, SIO1 will not acknowledge its own slave address or the general call address in the event of another device becoming master of the bus. In other words, if AA is reset, SIO0 cannot enter a Slave mode. STA, STO, and SI must be reset.

The Master Transmitter mode may now be entered by setting the STA bit using the SETB instruction. The SIO1 logic will now test the I^2C bus and generate a start condition as soon as the bus becomes free. When a START condition is transmitted, the serial interrupt flag (SI) is set, and the status code in the status register (S1STA) will be 08H. This status code must be used to vector to an interrupt service routine that loads S1DAT with the slave address and the data direction bit (SLA+W). The SI bit in S1CON must then be reset before the serial transfer can continue.

When the slave address and the direction bit have been transmitted and an acknowledgment bit has been received, the serial interrupt flag (SI) is set again, and a number of status codes in S1STA are possible. There are 18H, 20H, or 38H for the Master mode and also 68H, 78H, or B0H if the Slave mode was enabled (AA = logic 1). The appropriate action to be taken for each of these status codes is detailed in Table 4. After a repeated start condition (state 10H). SIO1 may switch to the Master Receiver mode by loading S1DAT with SLA+R).

Master Receiver mode

In the Master Receiver mode, a number of data bytes are received from a slave transmitter (see Figure 9). The transfer is initialized as in the Master Transmitter mode. When the start condition has been transmitted, the interrupt service routine must load S1DAT with the 7-bit slave address and the data direction bit (SLA+R). The SI bit in S1CON must then be cleared before the serial transfer can continue.

When the slave address and the data direction bit have been transmitted and an acknowledgment bit has been received, the serial interrupt flag (SI) is set again, and a number of status codes in S1STA are possible. These are 40H, 48H, or 38H for the Master mode and also 68H, 78H, or B0H if the Slave mode was enabled (AA = logic 1). The appropriate action to be taken for each of these status codes is detailed in Table 5. ENS1, CR1, and CR0 are not affected by the serial transfer and are not referred to in Table 5. After a repeated start condition (state 10H), SIO1 may switch to the Master Transmitter mode by loading S1DAT with SLA+W.

Slave Receiver mode

In the Slave Receiver mode, a number of data bytes are received from a master transmitter (see Figure 10). To initiate the Slave Receiver mode, S1ADR and S1CON must be loaded as follows:

	7	6	5	4	3	2	1	0
S1ADR (DBH)	X	X	X	X	X	X	X	GC
	own slave address							

The upper 7 bits are the address to which SIO1 will respond when addressed by a master. If the LSB (GC) is set, SIO1 will respond to the general call address (00H); otherwise it ignores the general call address.

	7	6	5	4	3	2	1	0
S1CON (D8H)	CR2	ENS1	STA	STO	SI	AA	CR1	CR0
	X	1	0	0	0	1	X	X

CR0, CR1, and CR2 do not affect SIO1 in the Slave mode. ENS1 must be set to logic 1 to enable SIO1. The AA bit must be set to enable SIO1 to acknowledge its own slave address or the general call address. STA, STO, and SI must be reset.

When S1ADR and S1CON have been initialized, SIO1 waits until it is addressed by its own slave address followed by the data direction bit which must be “0” (W) for SIO1 to operate in the Slave Receiver mode. After its own slave address and the W bit have been received, the serial interrupt flag (I) is set and a valid status code can be read from S1STA. This status code is used to vector to an interrupt service routine, and the appropriate action to be taken for each of these status codes is detailed in Table 6. The Slave Receiver mode may also be entered if arbitration is lost while SIO1 is in the Master mode (see status 68H and 78H).

If the AA bit is reset during a transfer, SIO1 will return a not acknowledge (logic 1) to SDA after the next received data byte. While AA is reset, SIO1 does not respond to its own slave address or a general call address. However, the I^2C bus is still monitored and address recognition may be resumed at any time by setting AA. This means that the AA bit may be used to temporarily isolate SIO1 from the I^2C bus.

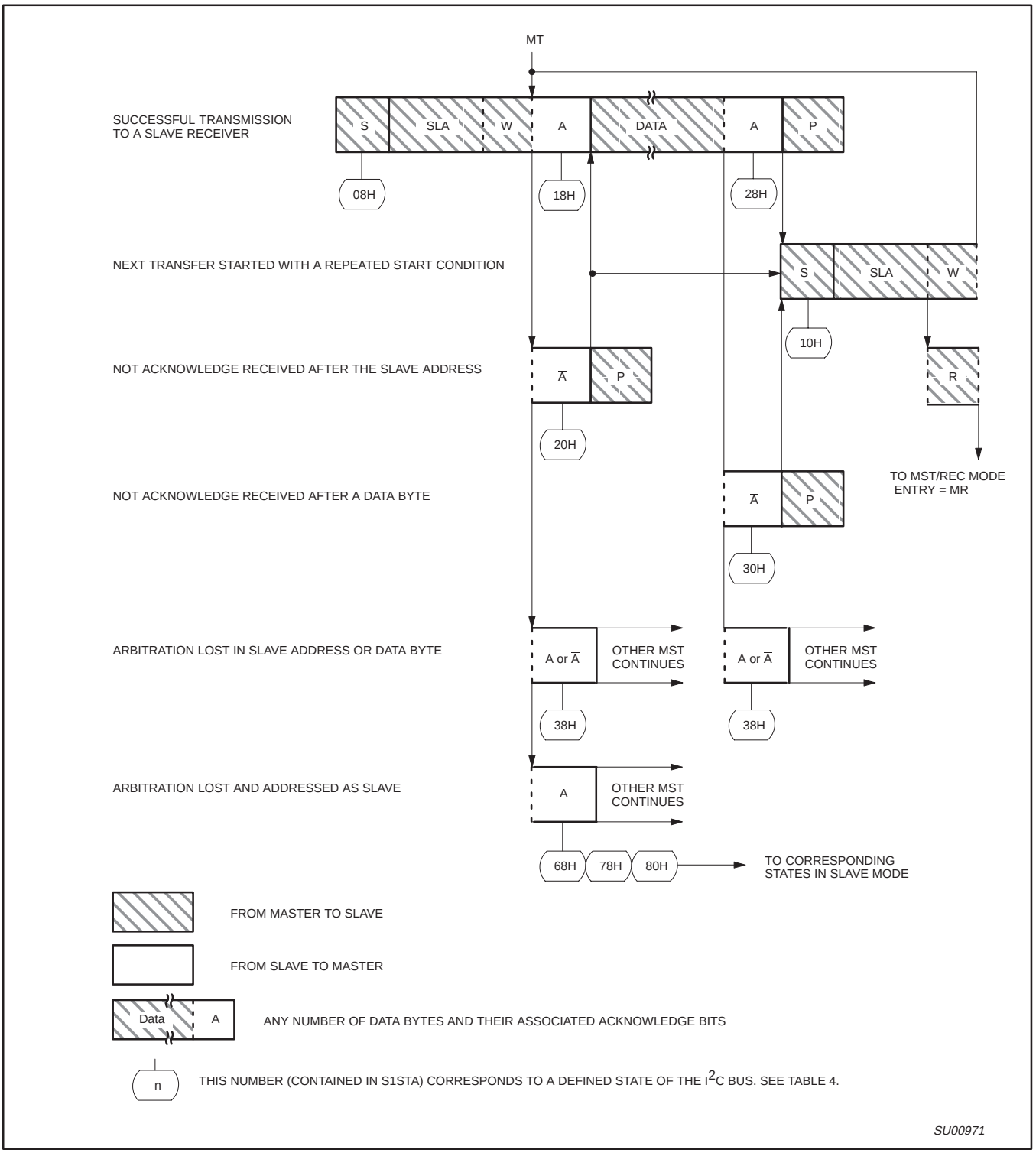


Figure 8. Format and States in the Master Transmitter mode

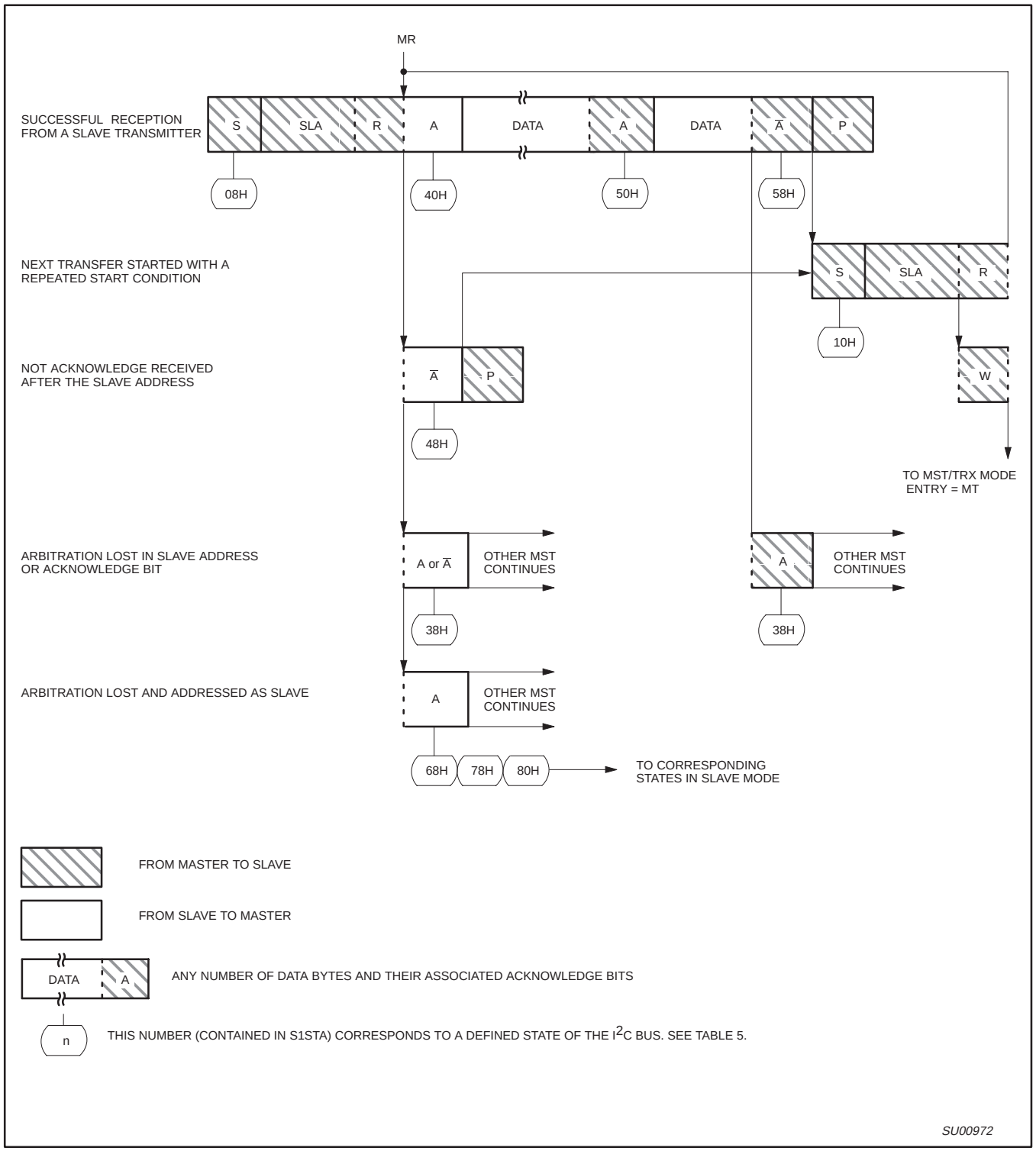


Figure 9. Format and States in the Master Receiver Mode

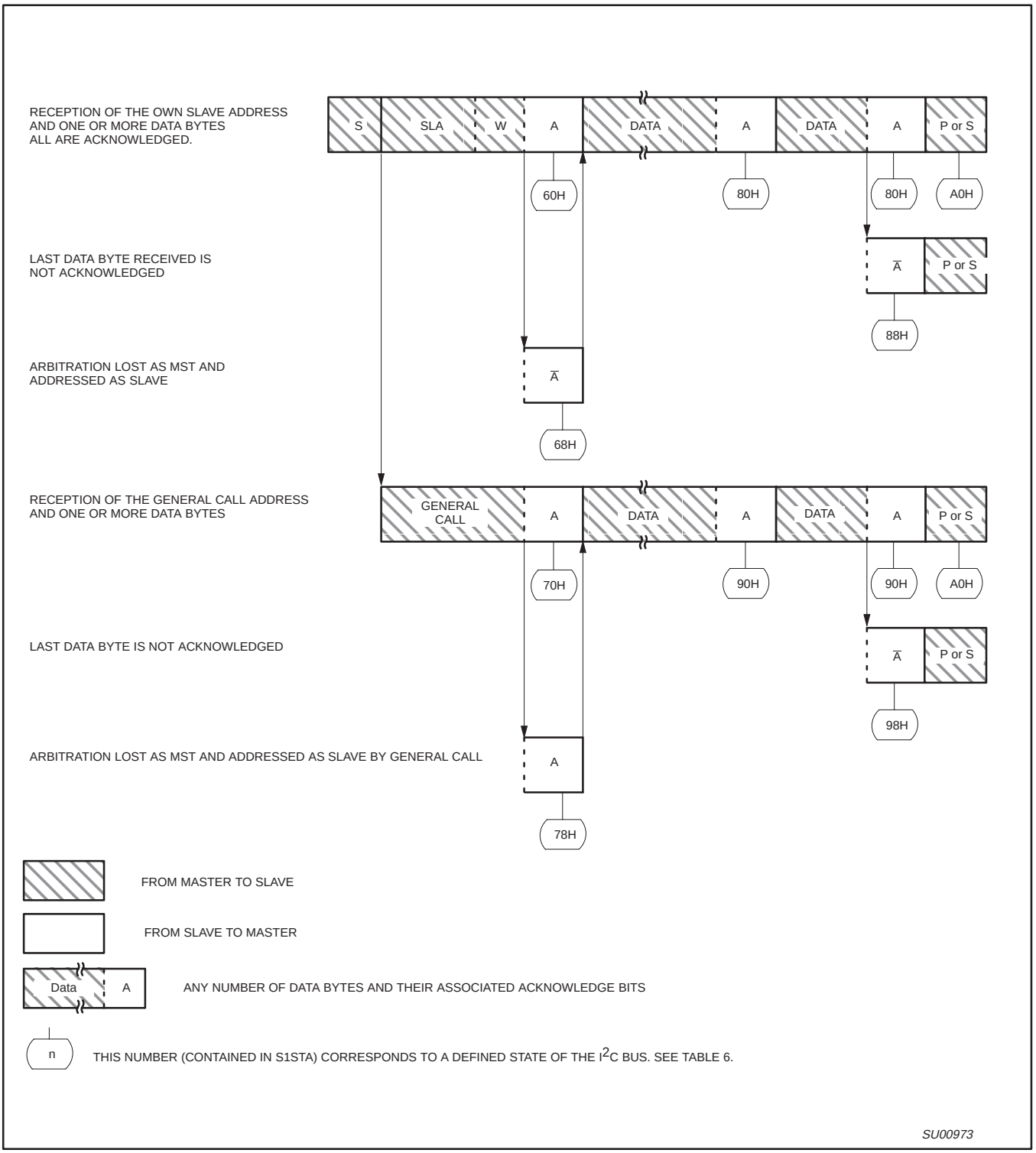


Figure 10. Format and States in the Slave Receiver mode

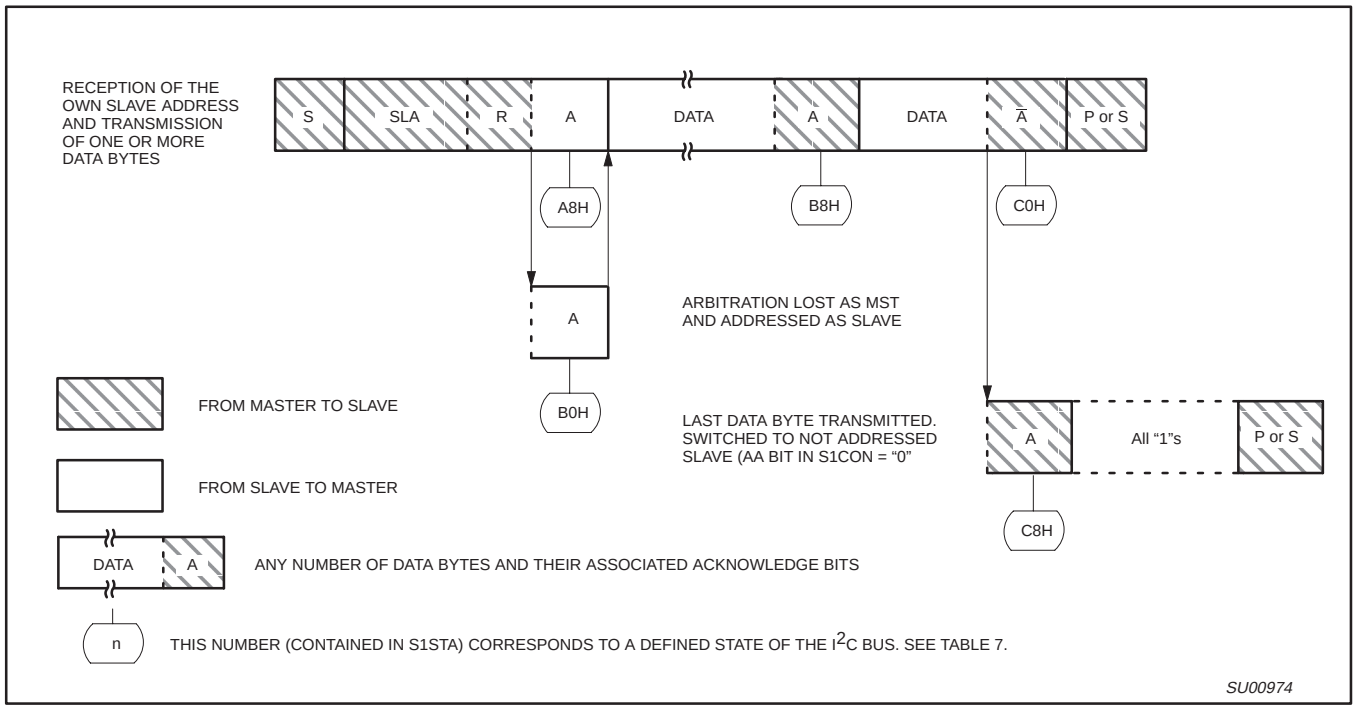


Figure 11. Format and States of the Slave Transmitter mode

Table 4. Master Transmitter mode

STATUS CODE (S1STA)	STATUS OF THE I^2C BUS AND SIO1 HARDWARE	APPLICATION SOFTWARE RESPONSE					NEXT ACTION TAKEN BY SIO1 HARDWARE
		TO/FROM S1DAT	TO S1CON				
			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 repeated START condition has been transmitted	Load SLA+W or	X	0	0	X	As above
		Load SLA+R	X	0	0	X	SLA+W will be transmitted; SIO1 will be switched to MST/REC 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 S1DAT action or	1	0	0	X	Repeated START will be transmitted;
		no S1DAT action or	0	1	0	X	STOP condition will be transmitted; STO flag will be reset
		no S1DAT 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 S1DAT action or	1	0	0	X	Repeated START will be transmitted;
		no S1DAT action or	0	1	0	X	STOP condition will be transmitted; STO flag will be reset
		no S1DAT action	1	1	0	X	STOP condition followed by a START condition will be transmitted; STO flag will be reset
28H	Data byte in S1DAT 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 S1DAT action or	1	0	0	X	Repeated START will be transmitted;
		no S1DAT action or	0	1	0	X	STOP condition will be transmitted; STO flag will be reset
		no S1DAT action	1	1	0	X	STOP condition followed by a START condition will be transmitted; STO flag will be reset
30H	Data byte in S1DAT 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 S1DAT action or	1	0	0	X	Repeated START will be transmitted;
		no S1DAT action or	0	1	0	X	STOP condition will be transmitted; STO flag will be reset
		no S1DAT 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/$\overline{W}$ or Data bytes	No S1DAT action or	0	0	0	X	I^2C bus will be released; not addressed slave will be entered
		No S1DAT action	1	0	0	X	A START condition will be transmitted when the bus becomes free

Table 5. Master Receiver Mode

STATUS CODE (S1STA)	STATUS OF THE I^2C BUS AND SIO1 HARDWARE	APPLICATION SOFTWARE RESPONSE					NEXT ACTION TAKEN BY SIO1 HARDWARE
		TO/FROM S1DAT	TO S1CON				
			STA	STO	SI	AA	
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 repeated START condition has been transmitted	Load SLA+R or	X	0	0	X	As above
		Load SLA+W	X	0	0	X	SLA+W will be transmitted; SIO1 will be switched to MST/TRX mode
38H	Arbitration lost in NOT ACK bit	No S1DAT action or	0	0	0	X	I^2C bus will be released; SIO1 will enter a Slave mode
		No S1DAT 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 S1DAT action or	0	0	0	0	Data byte will be received; NOT ACK bit will be returned
		no S1DAT action	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 S1DAT action or	1	0	0	X	Repeated START condition will be transmitted
		no S1DAT action or	0	1	0	X	STOP condition will be transmitted; STO flag will be reset
		no S1DAT action	1	1	0	X	STOP condition followed by a START condition will be transmitted; STO flag will be reset
50H	Data byte has been received; ACK has been returned	Read data byte or	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; NOT ACK has been returned	Read data byte or	1	0	0	X	Repeated START condition 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 6. Slave Receiver mode

STATUS CODE (S1STA)	STATUS OF THE I^2C BUS AND SIO1 HARDWARE	APPLICATION SOFTWARE RESPONSE					NEXT ACTION TAKEN BY SIO1 HARDWARE
		TO/FROM S1DAT	TO S1CON				
			STA	STO	SI	AA	
60H	Own SLA+W has been received; ACK has been returned	No S1DAT action or	X	0	0	0	Data byte will be received and NOT ACK will be returned
		no S1DAT action	X	0	0	1	Data byte will be received and ACK will be returned
68H	Arbitration lost in SLA+R/$\overline{W}$ as master; Own SLA+W has been received, ACK returned	No S1DAT action or	X	0	0	0	Data byte will be received and NOT ACK will be returned
		no S1DAT 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 S1DAT action or	X	0	0	0	Data byte will be received and NOT ACK will be returned
		no S1DAT action	X	0	0	1	Data byte will be received and ACK will be returned
78H	Arbitration lost in SLA+R/$\overline{W}$ as master; General call address has been received, ACK has been returned	No S1DAT action or	X	0	0	0	Data byte will be received and NOT ACK will be returned
		no S1DAT action	X	0	0	1	Data byte will be received and ACK will be returned
80H	Previously addressed with own SLV 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 and ACK will be returned
88H	Previously addressed with own SLA; DATA byte has been received; NOT ACK has been returned	Read data byte or	0	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or General call address
		read data byte or	0	0	0	1	Switched to not addressed SLV mode; Own SLA will be recognized; General call address will be recognized if S1ADR.0 = logic 1
		read data byte or	1	0	0	0	Switched to not addressed SLV 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 SLV mode; Own SLA will be recognized; General call address will be recognized if S1ADR.0 = logic 1. A START condition will be transmitted when the bus becomes free.
90H	Previously addressed with General Call; DATA byte 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 byte has been received; NOT ACK has been returned	Read data byte or	0	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or General call address
		read data byte or	0	0	0	1	Switched to not addressed SLV mode; Own SLA will be recognized; General call address will be recognized if S1ADR.0 = logic 1
		read data byte or	1	0	0	0	Switched to not addressed SLV 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 SLV mode; Own SLA will be recognized; General call address will be recognized if S1ADR.0 = logic 1. A START condition will be transmitted when the bus becomes free.

Table 6. Slave Receiver mode (Continued)

STATUS CODE (S1STA)	STATUS OF THE I^2C BUS AND SIO1 HARDWARE	APPLICATION SOFTWARE RESPONSE					NEXT ACTION TAKEN BY SIO1 HARDWARE
		TO/FROM S1DAT	TO S1CON				
			STA	STO	SI	AA	
A0H	A STOP condition or repeated START condition has been received while still addressed as SLV/REC or SLV/TRX	No STDAT action or	0	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or General call address
		No STDAT action or	0	0	0	1	Switched to not addressed SLV mode; Own SLA will be recognized; General call address will be recognized if S1ADR.0 = logic 1
		No STDAT action or	1	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or General call address. A START condition will be transmitted when the bus becomes free
		No STDAT action	1	0	0	1	Switched to not addressed SLV mode; Own SLA will be recognized; General call address will be recognized if S1ADR.0 = logic 1. A START condition will be transmitted when the bus becomes free.

Table 7. Slave Transmitter mode

STATUS CODE (S1STA)	STATUS OF THE I^2C BUS AND SIO1 HARDWARE	APPLICATION SOFTWARE RESPONSE					NEXT ACTION TAKEN BY SIO1 HARDWARE
		TO/FROM S1DAT	TO S1CON				
			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/$\overline{W}$ 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 S1DAT 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 bit will be received
C0H	Data byte in S1DAT has been transmitted; NOT ACK has been received	No S1DAT action or	0	0	0	01	Switched to not addressed SLV mode; no recognition of own SLA or General call address
		no S1DAT action or	0	0	0	1	Switched to not addressed SLV mode; Own SLA will be recognized; General call address will be recognized if S1ADR.0 = logic 1
		no S1DAT action or	1	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or General call address. A START condition will be transmitted when the bus becomes free
		no S1DAT action	1	0	0	1	Switched to not addressed SLV mode; Own SLA will be recognized; General call address will be recognized if S1ADR.0 = logic 1. A START condition will be transmitted when the bus becomes free.
C8H	Last data byte in S1DAT has been transmitted (AA = 0); ACK has been received	No S1DAT action or	0	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or General call address
		no S1DAT action or	0	0	0	1	Switched to not addressed SLV mode; Own SLA will be recognized; General call address will be recognized if S1ADR.0 = logic 1
		no S1DAT action or	1	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or General call address. A START condition will be transmitted when the bus becomes free
		no S1DAT action	1	0	0	1	Switched to not addressed SLV mode; Own SLA will be recognized; General call address will be recognized if S1ADR.0 = logic 1. A START condition will be transmitted when the bus becomes free.

Table 8. Miscellaneous States

STATUS CODE (S1STA)	STATUS OF THE I^2C BUS AND SIO1 HARDWARE	APPLICATION SOFTWARE RESPONSE					NEXT ACTION TAKEN BY SIO1 HARDWARE
		TO/FROM S1DAT	TO S1CON				
			STA	STO	SI	AA	
F8H	No relevant state information available; SI = 0	No S1DAT action	No S1CON action				Wait or proceed current transfer
00H	Bus error during MST or selected Slave modes, due to an illegal START or STOP condition. State 00H can also occur when interference causes SIO1 to enter an undefined state.	No S1DAT action	0	1	0	X	Only the internal hardware is affected in the MST or addressed SLV modes. In all cases, the bus is released and SIO1 is switched to the not addressed SLV mode. STO is reset.

Slave Transmitter mode
In the Slave Transmitter mode, a number of data bytes are transmitted to a master receiver (see Figure 11). Data transfer is initialized as in the Slave Receiver mode. When S1ADR and S1CON have been initialized, SIO1 waits until it is addressed by its own slave address followed by the data direction bit which must be "1" (R) for SIO1 to operate in the Slave Transmitter mode. After its own slave address and the R bit have been received, the serial interrupt flag (SI) is set and a valid status code can be read from S1STA. This status code is used to vector to an interrupt service routine, and the appropriate action to be taken for each of these status codes is detailed in Table 7. The Slave Transmitter mode may also be entered if arbitration is lost while SIO1 is in the Master mode (see state B0H).

If the AA bit is reset during a transfer, SIO1 will transmit the last byte of the transfer and enter state C0H or C8H. SIO1 is switched to the "not addressed" Slave mode and will ignore the master receiver if it continues the transfer. Thus the master receiver receives all 1s as serial data. While AA is reset, SIO1 does not respond to its own slave address or a general call address. However, the I^2C bus is still monitored, and address recognition may be resumed at any time by setting AA. This means that the AA bit may be used to temporarily isolate SIO1 from the I^2C bus.

Miscellaneous States

There are two S1STA codes that do not correspond to a defined SIO1 hardware state (see Table 8). These are discussed below.

S1STA = F8H
This status code indicates that no relevant information is available because the serial interrupt flag, SI, is not yet set. This occurs between other states and when SIO1 is not involved in a serial transfer.

S1STA = 00H
This status code indicates that a bus error has occurred during an SIO1 serial transfer. A bus error is caused when a START or STOP condition occurs at an illegal position in the format frame. Examples of such illegal positions are during the serial transfer of an address byte, a data byte, or an acknowledge bit. A bus error may also be caused when external interference disturbs the internal SIO1 signals. When a bus error occurs, SI is set. To recover from a bus error, the STO flag must be set and SI must be cleared. This causes SIO1 to enter the "not addressed" Slave mode (a defined state) and to clear the STO flag (no other bits in S1CON are affected). The SDA and SCL lines are released (a STOP condition is not transmitted).

Some Special Cases

The SIO1 hardware has facilities to handle the following special cases that may occur during a serial transfer.

Simultaneous Repeated START Conditions from Two Masters
A repeated START condition may be generated in the Master Transmitter or Master Receiver modes. A special case occurs if another master simultaneously generates a repeated START condition (see Figure 12). Until this occurs, arbitration is not lost by either master since they were both transmitting the same data.

If the SIO1 hardware detects a repeated START condition on the I^2C bus before generating a repeated START condition itself, it will release the bus, and no interrupt request is generated. If another master frees the bus by generating a STOP condition, SIO1 will transmit a normal START condition (state 08H), and a retry of the total serial data transfer can commence.

Data Transfer After Loss of Arbitration
Arbitration may be lost in the Master Transmitter and Master Receiver modes (see Figure 4). Loss of arbitration is indicated by the following states in S1STA: 38H, 68H, 78H, and B0H (see Figures 8 and 9).

If the STA flag in S1CON is set by the routines which service these states, then, if the bus is free again, a START condition (state 08H) is transmitted without intervention by the CPU, and a retry of the total serial transfer can commence.

Forced Access to the I^2C Bus
In some applications, it may be possible for an uncontrolled source to cause a bus hang-up. In such situations, the problem may be caused by interference, temporary interruption of the bus or a temporary short-circuit between SDA and SCL.

If an uncontrolled source generates a superfluous START or masks a STOP condition, then the I^2C bus stays busy indefinitely. If the STA flag is set and bus access is not obtained within a reasonable amount of time, then a forced access to the I^2C bus is possible. This is achieved by setting the STO flag while the STA flag is still set. No STOP condition is transmitted. The SIO1 hardware behaves as if a STOP condition was received and is able to transmit a START condition. The ST0 flag is cleared by hardware (see Figure 13).

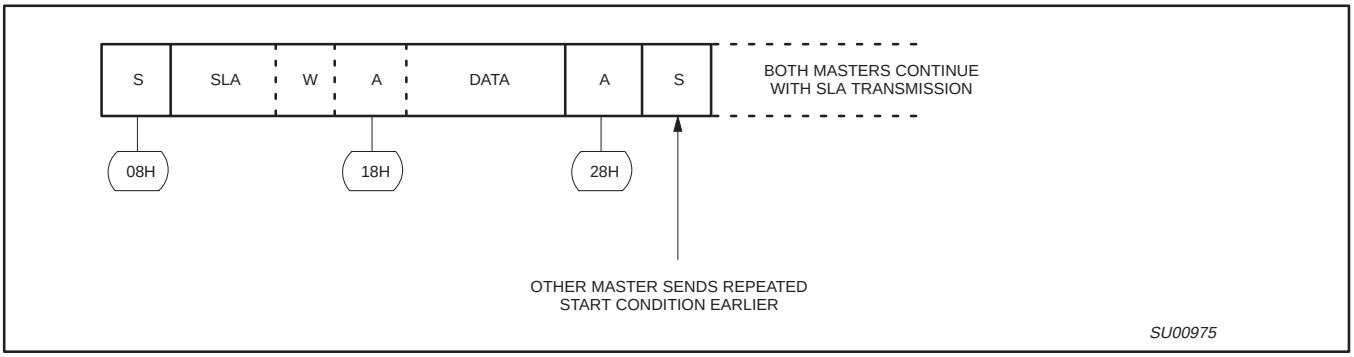


Figure 12. Simultaneous Repeated START Conditions from 2 Masters

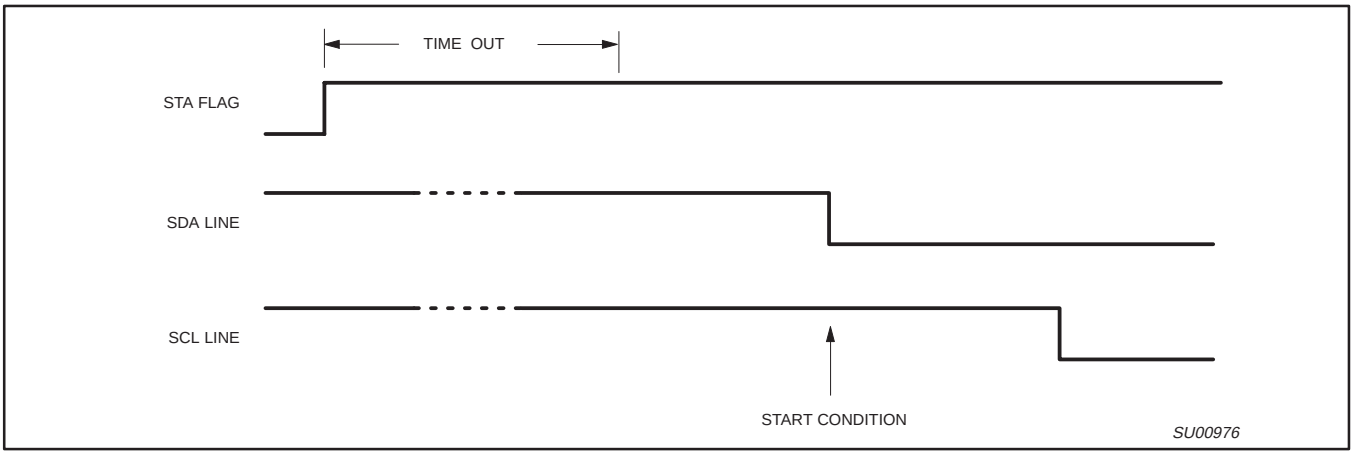


Figure 13. Forced Access to a Busy I²C Bus

I²C Bus Obstructed by a Low Level on SCL or SDA

An I²C bus hang-up occurs if SDA or SCL is pulled LOW by an uncontrolled source. If the SCL line is obstructed (pulled LOW) by a device on the bus, no further serial transfer is possible, and the SIO1 hardware cannot resolve this type of problem. When this occurs, the problem must be resolved by the device that is pulling the SCL bus line LOW.

If the SDA line is obstructed by another device on the bus (e.g., a slave device out of bit synchronization), the problem can be solved by transmitting additional clock pulses on the SCL line (see Figure 14). The SIO1 hardware transmits additional clock pulses when the STA flag is set, but no START condition can be generated because the SDA line is pulled LOW while the I²C bus is considered free. The SIO1 hardware attempts to generate a START condition after every two additional clock pulses on the SCL line. When the SDA line is eventually released, a normal START condition is transmitted, state 08H is entered, and the serial transfer continues.

If a forced bus access occurs or a repeated START condition is transmitted while SDA is obstructed (pulled LOW), the SIO1 hardware performs the same action as described above. In each case, state 08H is entered after a successful START condition is transmitted and normal serial transfer continues. Note that the CPU is not involved in solving these bus hang-up problems.

Bus Error

A bus error occurs when a START or STOP condition is present at an illegal position in the format frame. Examples of illegal positions are during the serial transfer of an address byte, a data, or an acknowledge bit.

The SIO1 hardware only reacts to a bus error when it is involved in a serial transfer either as a master or an addressed slave. When a bus error is detected, SIO1 immediately switches to the “not addressed” Slave mode, releases the SDA and SCL lines, sets the interrupt flag, and loads the status register with 00H. This status code may be used to vector to a service routine which either attempts the aborted serial transfer again or simply recovers from the error condition as shown in Table 8.

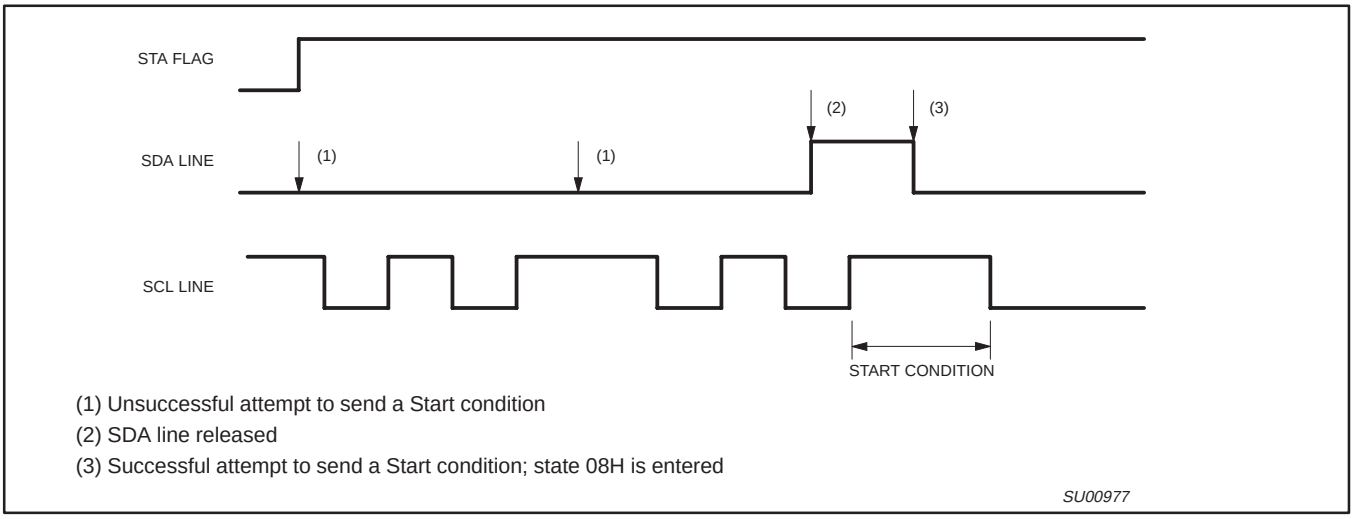


Figure 14. Recovering from a Bus Obstruction Caused by a Low Level on SDA

An I^2C byte-oriented system driver is described in application note AN435. Please visit
http://www.semiconductors.philips.com/products/all_appnotes.html

TIMER 0 AND TIMER 1 OPERATION

Timer 0 and Timer 1

The "Timer" or "Counter" function is selected by control bits C/$\overline{T}$ in the Special Function Register TMOD (see Figure 15). These two Timer/Counters have four operating modes, which are selected by bit-pairs (M1, M0) in TMOD. Modes 0, 1, and 2 are the same for both Timers/Counters. Mode 3 is different. The four operating modes are described in the following text.

Mode 0

Putting either Timer into Mode 0 makes it behave as an 8048 Timer, which is an 8-bit Counter with a divide-by-32 prescaler. Figure 16 shows the 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 counted input is enabled to the Timer when TRn = 1 and either GATE = 0 or $\overline{INTn}$ = 1. TRn is a control bit in the Special Function Register TCON (Figure 17). (Setting GATE = 1 allows the Timer to be controlled by external input $\overline{INTn}$, to facilitate pulse width measurements).

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 as for Timer 1. There are two different GATE bits, one for Timer 1 (TMOD.7) and one for Timer 0 (TMOD.3).

Mode 1

Mode 1 is the same as Mode 0, except that the Timer register is being run with all 16 bits.

Mode 2

Mode 2 configures the Timer register as an 8-bit Counter (TLn) with automatic reload, as shown in Figure 18. Overflow from TLn not only sets TFn, but also reloads TLn with the contents of THn, which is preset by software. The reload leaves THn unchanged.

Mode 2 operation is the same for Timer 0 as for Timer 1.

Mode 3

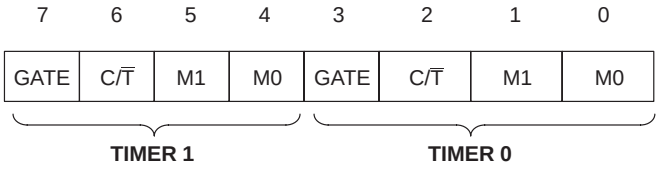
Timer 1 in Mode 3 simply holds its count. The effect is the same as setting TR1 = 0.

Timer 0 in Mode 3 establishes TL0 and TH0 as two separate counters. The logic for Mode 3 on Timer 0 is shown in Figure 19. TL0 uses the Timer 0 control bits: C/$\overline{T}$, GATE, TR0, and TF0 as well as pin $\overline{INT0}$. 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 requiring an extra 8-bit timer on the counter. Putting Timer 0 in Mode 3 allows an 80C51 to have three Timer/Counters. When Timer 0 is in Mode 3, Timer 1 can be turned on and off by switching it out of and into its own Mode 3, or can still be used by the serial port as a baud rate generator, or in fact, in any application not requiring an interrupt.

TMOD Address = 89H Reset Value = 00H

Not Bit Addressable



BIT	SYMBOL	FUNCTION
TMOD.3/ TMOD.7	GATE	Gating control when set. Timer/Counter "n" is enabled only while "$\overline{INTn}$" pin is high and "TRn" control pin is set. when cleared Timer "n" is enabled whenever "TRn" control bit is set.
TMOD.2/ TMOD.6	C/$\overline{T}$	Timer or Counter Selector cleared for Timer operation (input from internal system clock.) Set for Counter operation (input from "Tn" input pin).

M1	M0	OPERATING
0	0	8048 Timer: "TLn" serves as 5-bit prescaler.
0	1	16-bit Timer/Counter: "THn" and "TLn" are cascaded; there is no prescaler.
1	0	8-bit auto-reload Timer/Counter: "THn" holds a value which is to be reloaded into "TLn" each time it overflows.
1	1	(Timer 0) TL0 is an 8-bit Timer/Counter controlled by the standard Timer 0 control bits. TH0 is an 8-bit timer only controlled by Timer 1 control bits.
1	1	(Timer 1) Timer/Counter 1 stopped.

SU01580

Figure 15. Timer/Counter 0/1 Mode Control (TMOD) Register

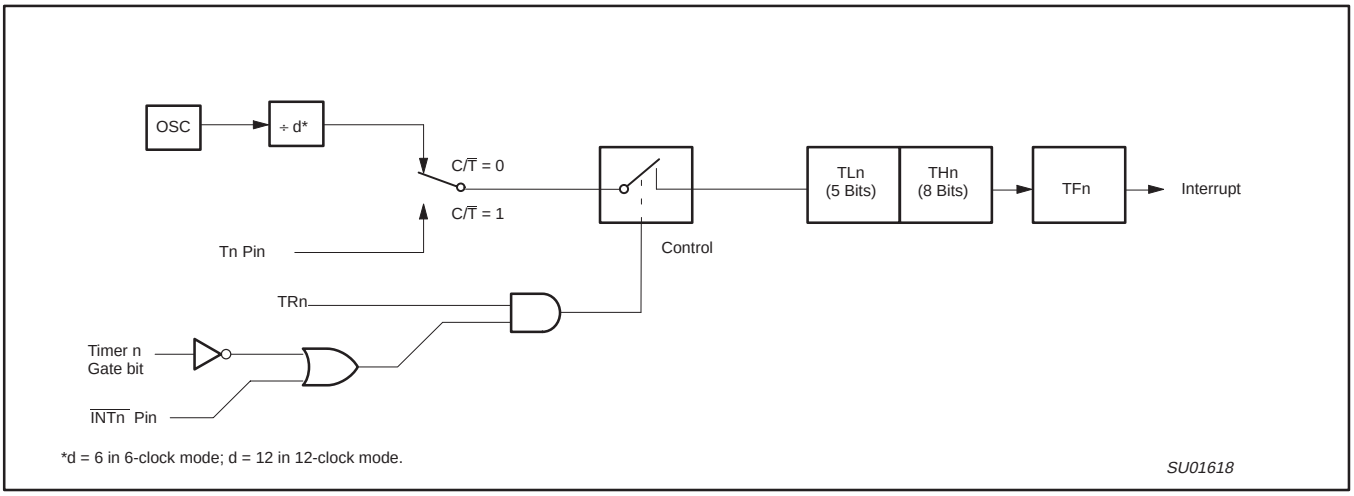


Figure 16. Timer/Counter 0/1 Mode 0: 13-Bit Timer/Counter

TCON Address = 88H Reset Value = 00H

Bit Addressable

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

BIT	SYMBOL	FUNCTION
TCON.7	TF1	Timer 1 overflow flag. Set by hardware on Timer/Counter overflow. Cleared by hardware when processor vectors to interrupt routine, or clearing the bit in software.
TCON.6	TR1	Timer 1 Run control bit. Set/cleared by software to turn Timer/Counter on/off.
TCON.5	TF0	Timer 0 overflow flag. Set by hardware on Timer/Counter overflow. Cleared by hardware when processor vectors to interrupt routine, or by clearing the bit in software.
TCON.4	TR0	Timer 0 Run control bit. Set/cleared by software to turn Timer/Counter on/off.
TCON.3	IE1	Interrupt 1 Edge flag. Set by hardware when external interrupt edge detected. Cleared when interrupt processed.
TCON.2	IT1	Interrupt 1 type control bit. Set/cleared by software to specify falling edge/low level triggered external interrupts.
TCON.1	IE0	Interrupt 0 Edge flag. Set by hardware when external interrupt edge detected. Cleared when interrupt processed.
TCON.0	IT0	Interrupt 0 Type control bit. Set/cleared by software to specify falling edge/low level triggered external interrupts.

SU01516

Figure 17. Timer/Counter 0/1 Control (TCON) Register

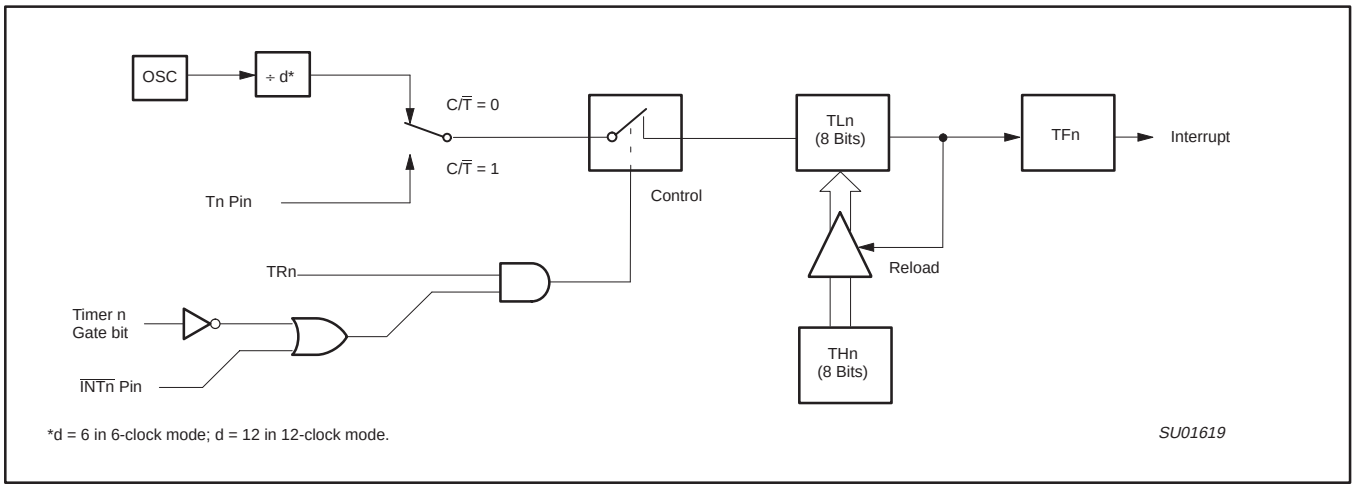


Figure 18. Timer/Counter 0/1 Mode 2: 8-Bit Auto-Reload

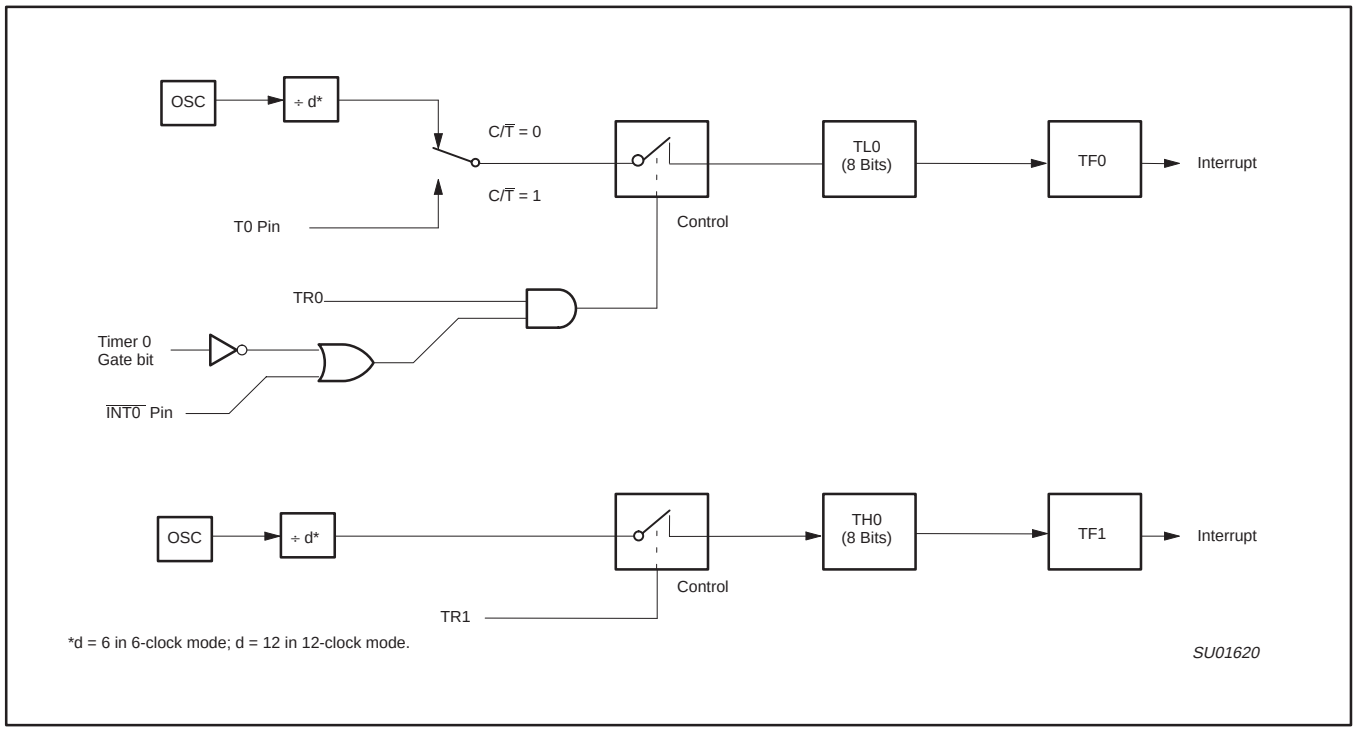


Figure 19. Timer/Counter 0 Mode 3: Two 8-Bit Counters

TIMER 2 OPERATION

Timer 2

Timer 2 is a 16-bit Timer/Counter which can operate as either an event timer or an event counter, as selected by C/$\overline{T}$2 in the special function register T2CON (see Figure 20). Timer 2 has three operating modes:

– Capture Mode
– Auto-Reload Mode (up or down counting)
– Baud Rate Generator Mode (see Table 10)

Capture Mode

In the capture mode there are two options which are selected by bit EXEN2 in T2CON. If EXEN2 = 0, then Timer 2 is a 16-bit timer or counter (as selected by C/$\overline{T}$2 in T2CON) which, upon overflowing sets bit TF2, the Timer 2 overflow bit. This bit can be used to generate an interrupt (by enabling the Timer 2 interrupt bit in the IE register). If EXEN2 = 1, Timer 2 operates as described above, with the added feature that a 1-to-0 transition at external input pin T2EX causes the current value in the Timer 2 registers, TL2 and TH2, to be captured into registers RCAP2L and RCAP2H. In addition, the transition at T2EX causes bit EXF2 in T2CON to be set. If Timer 2 interrupt has been enabled, EXF2 will generate an interrupt (which vectors to the same location as Timer 2 overflow interrupt). The Timer 2 interrupt service routine can interrogate TF2 and EXF2 to determine which event caused the interrupt. The capture mode is illustrated in Figure 21 (There is no reload value for TL2 and TH2 in this mode). Even when a capture event occurs from T2EX, the counter keeps on counting T2EX pin transitions or osc/6 pulses (osc/12 in 12 clock mode).

Auto-Reload Mode (Up or Down Counter)

In the 16-bit auto-reload mode, Timer 2 can be configured as either a timer or counter (C/$\overline{T}$2 in T2CON), then programmed to count up or down. The counting direction is determined by bit DCEN (Down Counter Enable) which is located in the T2MOD register (see Figure 22). When reset is applied (DCEN = 0), Timer 2 defaults to counting up. If DCEN bit is set, Timer 2 can count up or down depending on the value of the T2EX pin.

Figure 23 shows Timer 2 which will count up automatically since DCEN = 0. In this mode there are two options selected by bit EXEN2 in T2CON register. If EXEN2 = 0, then Timer 2 counts up to 0FFFFH and sets the TF2 (Overflow Flag) bit upon overflow. This causes the Timer 2 registers to be reloaded with the 16-bit value in RCAP2L and RCAP2H. The values in RCAP2L and RCAP2H are preset by software.

If EXEN2 = 1, then a 16-bit reload can be triggered either by an overflow or by a 1-to-0 transition at input pin T2EX. This transition also sets the EXF2 bit. The Timer 2 interrupt, if enabled, can be generated when either TF2 or EXF2 are 1.

In Figure 24 DCEN = 1 which enables Timer 2 to count up or down. This mode allows pin T2EX to control the direction of count. When a logic 1 is applied at pin T2EX Timer 2 will count up. Timer 2 will overflow at 0FFFFH and set the TF2 flag, which can then generate an interrupt, if the interrupt is enabled. This timer overflow also causes the 16-bit value in RCAP2L and RCAP2H to be reloaded into the timer registers TL2 and TH2.

When a logic 0 is applied at pin T2EX this causes Timer 2 to count down. The timer will underflow when TL2 and TH2 become equal to the value stored in RCAP2L and RCAP2H. Timer 2 underflow sets the TF2 flag and causes 0FFFFH to be reloaded into the timer registers TL2 and TH2.

The external flag EXF2 toggles when Timer 2 underflows or overflows. This EXF2 bit can be used as a 17th bit of resolution if needed. The EXF2 flag does not generate an interrupt in this mode of operation.

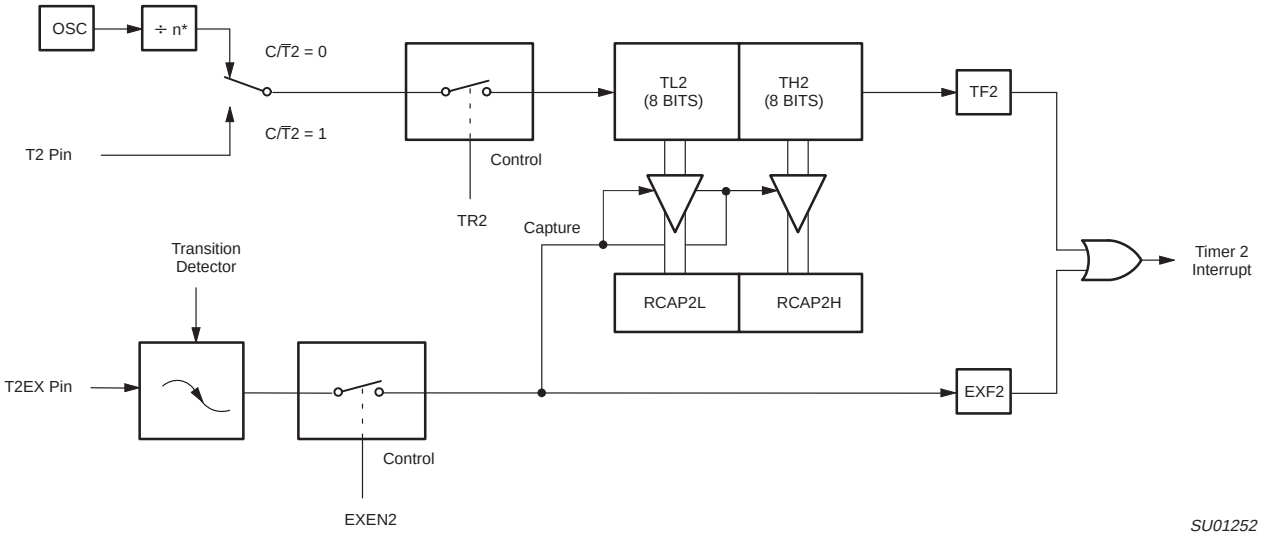
Symbol	Position	Name and Significance
TF2	T2CON.7	Timer 2 overflow flag set by a Timer 2 overflow and must be cleared by software. TF2 will not be set when either RCLK or TCLK = 1.
EXF2	T2CON.6	Timer 2 external flag set when either a capture or reload is caused by a negative transition on T2EX and EXEN2 = 1. When Timer 2 interrupt is enabled, EXF2 = 1 will cause the CPU to vector to the Timer 2 interrupt routine. EXF2 must be cleared by software. EXF2 does not cause an interrupt in up/down counter mode (DCEN = 1).
RCLK	T2CON.5	Receive clock flag. When set, causes the serial port to use Timer 2 overflow pulses for its receive clock in modes 1 and 3. RCLK = 0 causes Timer 1 overflow to be used for the receive clock.
TCLK	T2CON.4	Transmit clock flag. When set, causes the serial port to use Timer 2 overflow pulses for its transmit clock in modes 1 and 3. TCLK = 0 causes Timer 1 overflows to be used for the transmit clock.
EXEN2	T2CON.3	Timer 2 external enable flag. When set, allows a capture or reload to occur as a result of a negative transition on T2EX if Timer 2 is not being used to clock the serial port. EXEN2 = 0 causes Timer 2 to ignore events at T2EX.
TR2	T2CON.2	Start/stop control for Timer 2. A logic 1 starts the timer.
C/$\overline{T}$2	T2CON.1	Timer or counter select. (Timer 2) 0 = Internal timer (OSC/6 in 6-clock mode or OSC/12 in 12-clock mode) 1 = External event counter (falling edge triggered).
CP/$\overline{RL}$2	T2CON.0	Capture/Reload flag. When set, captures will occur on negative transitions at T2EX if EXEN2 = 1. When cleared, auto-reloads will occur either with Timer 2 overflows or negative transitions at T2EX when EXEN2 = 1. When either RCLK = 1 or TCLK = 1, this bit is ignored and the timer is forced to auto-reload on Timer 2 overflow.

SU01251

Figure 20. Timer/Counter 2 (T2CON) Control Register

Table 9. Timer 2 Operating Modes

RCLK + TCLK	CP/$\overline{RL2}$	TR2	MODE
0	0	1	16-bit Auto-reload
0	1	1	16-bit Capture
1	X	1	Baud rate generator
X	X	0	(off)



* n = 6 in 6-clock mode, or 12 in 12-clock mode.

Figure 21. Timer 2 in Capture Mode

T2MOD Address = 0C9H Reset Value = XXXX XX00B

Not Bit Addressable

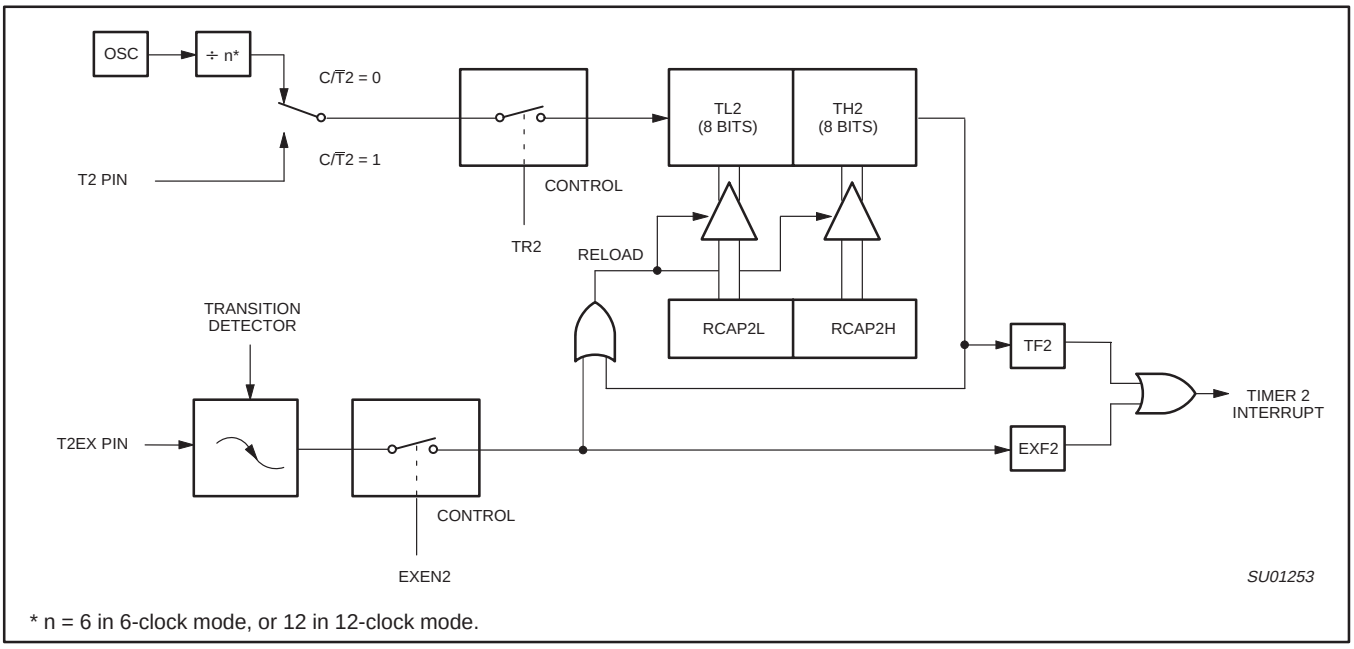
—	—	—	—	—	—	T2OE	DCEN
Bit 7	6	5	4	3	2	1	0

Symbol	Function
—	Not implemented, reserved for future use.*
T2OE	Timer 2 Output Enable bit.
DCEN	Down Count Enable bit. When set, this allows Timer 2 to be configured as an up/down counter depending on the value of the T2EX pin.

* User software should not write 1s to reserved bits. These bits may be used in future 8051 family products to invoke new features. In that case, the reset or inactive value of the new bit will be 0, and its active value will be 1. The value read from a reserved bit is indeterminate.

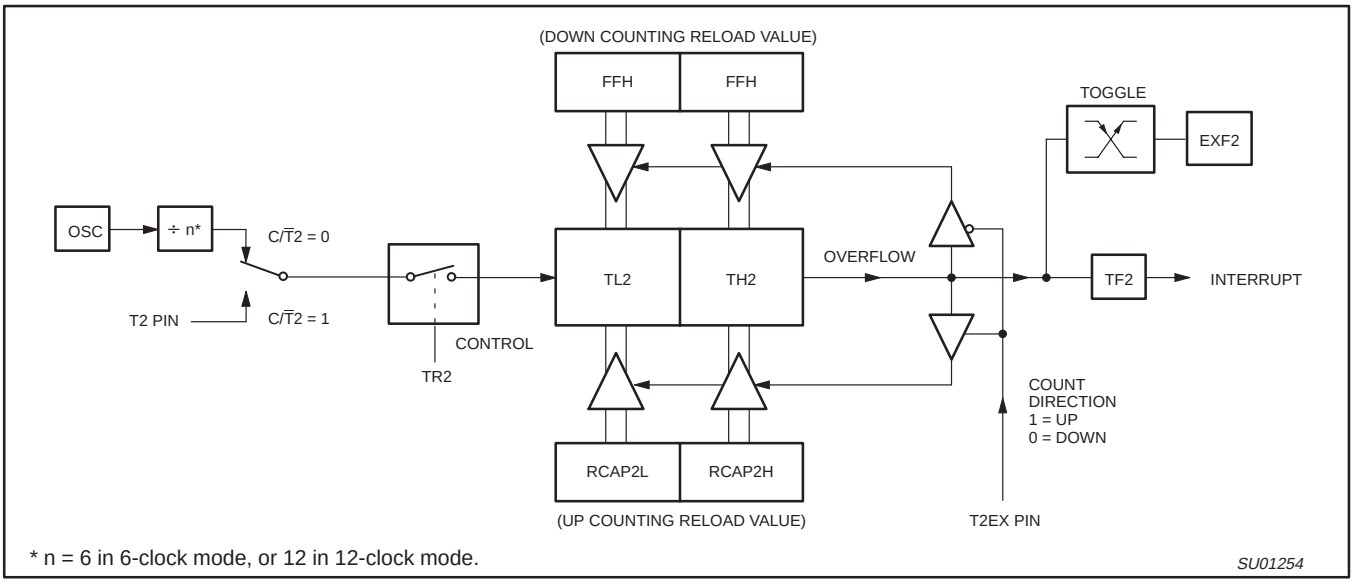
SU01714

Figure 22. Timer 2 Mode (T2MOD) Control Register



* n = 6 in 6-clock mode, or 12 in 12-clock mode.

Figure 23. Timer 2 in Auto-Reload Mode (DCEN = 0)



* n = 6 in 6-clock mode, or 12 in 12-clock mode.

Figure 24. Timer 2 Auto Reload Mode (DCEN = 1)

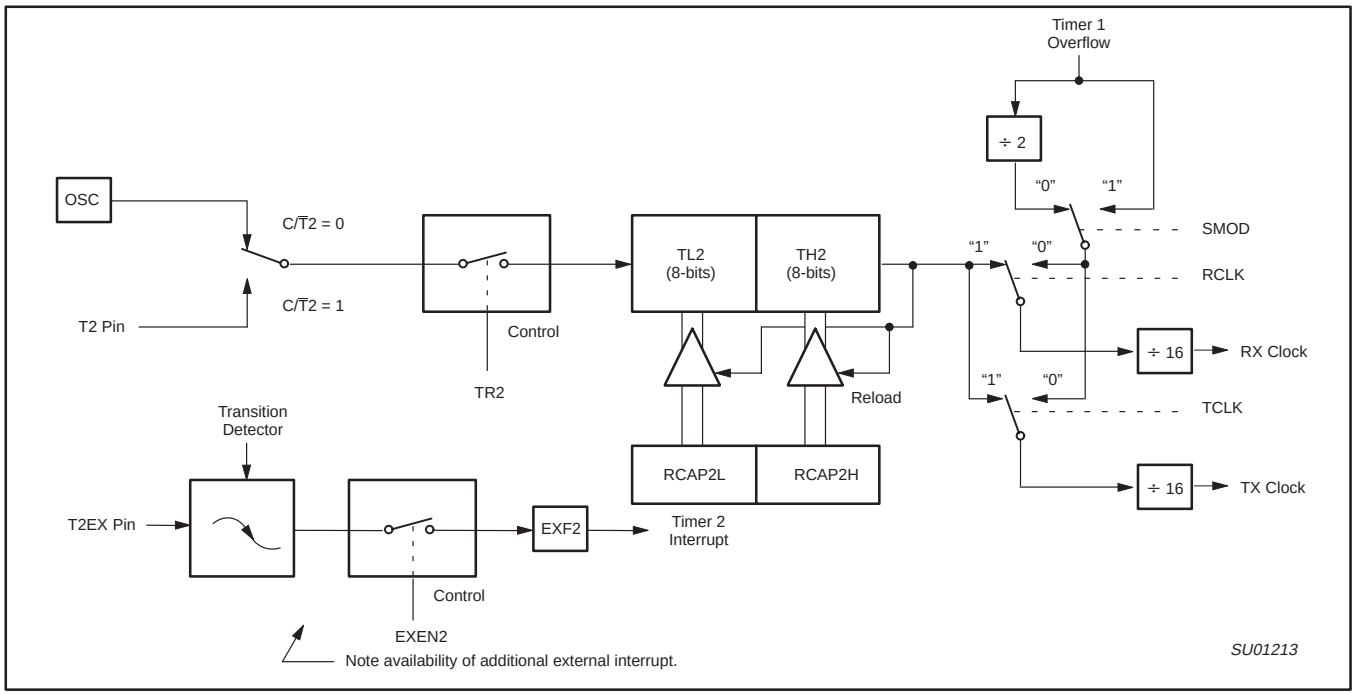


Figure 25. Timer 2 in Baud Rate Generator Mode

Table 10. Timer 2 Commonly Used Generated Baud Rates

Baud Rate		Osc Freq	Timer 2	
12 clock mode	6 clock mode		RCAP2H	RCAP2L
375 k	750 k	12 MHz	FF	FF
9.6 k	19.2 k	12 MHz	FF	D9
2.8 k	5.6 k	12 MHz	FF	B2
2.4 k	4.8 k	12 MHz	FF	64
1.2 k	2.4 k	12 MHz	FE	C8
300	600	12 MHz	FB	1E
110	220	12 MHz	F2	AF
300	600	6 MHz	FD	8F
110	220	6 MHz	F9	57

Baud Rate Generator Mode

Bits TCLK and/or RCLK in T2CON (see Figure 20) allow the serial port transmit and receive baud rates to be derived from either Timer 1 or Timer 2. When TCLK = 0, Timer 1 is used as the serial port transmit baud rate generator. When TCLK = 1, Timer 2 is used as the serial port transmit baud rate generator. RCLK has the same effect for the serial port receive baud rate. With these two bits, the serial port can have different receive and transmit baud rates – one generated by Timer 1, the other by Timer 2.

Table 10 shows commonly used baud rates and how they can be obtained from Timer 2.

Figure 25 shows Timer 2 in baud rate generation mode. The baud rate generation mode is similar to the auto-reload mode, in that a rollover in TH2 causes the Timer 2 registers to be reloaded with the 16-bit value in registers RCAP2H and RCAP2L, which are preset by software.

The baud rates in modes 1 and 3 are determined by Timer 2's overflow rate given below:

$$\text{Modes 1 and 3 Baud Rates} = \frac{\text{Timer 2 Overflow Rate}}{16}$$

The timer can be configured for either "timer" or "counter" operation. In many applications, it is configured for "timer" operation (C/$\overline{T}$2 = 0). Timer operation is different for Timer 2 when it is being used as a baud rate generator.

Usually, as a timer, it would increment every machine cycle (i.e., $^1/_6$ the oscillator frequency in 6 clock mode, $^1/_{12}$ the oscillator frequency in 12 clock mode). As a baud rate generator, it increments at the oscillator frequency in 6 clock mode (f_{OSC}/2 in 12 clock mode). Thus the baud rate formula is as follows:

$$\text{Modes 1 and 3 Baud Rates} = \frac{\text{Oscillator Frequency}}{[\,n^* \times [65536 - (\text{RCAP2H, RCAP2L})]]}$$

* n = 16 in 6 clock mode
32 in 12 clock mode

Where: (RCAP2H, RCAP2L) = The content of RCAP2H and RCAP2L taken as a 16-bit unsigned integer.

The Timer 2 as a baud rate generator mode shown in Figure 25, is valid only if RCLK and/or TCLK = 1 in T2CON register. Note that a rollover in TH2 does not set TF2, and will not generate an interrupt. Thus, Timer 2 interrupt does not have to be disabled when Timer 2 is in the baud rate generator mode. Also if the EXEN2 (T2 external enable flag) is set, a 1-to-0 transition in T2EX (Timer/counter 2 trigger input) will set EXF2 (T2 external flag) but will not cause a reload from (RCAP2H, RCAP2L) to (TH2,TL2). Therefore when Timer 2 is in use as a baud rate generator, T2EX can be used as an additional external interrupt, if needed.

When Timer 2 is in the baud rate generator mode, one should not try to read or write TH2 and TL2. As a baud rate generator, Timer 2 is incremented every state time ($f_{OSC}/2$) or asynchronously from pin T2; under these conditions, a read or write of TH2 or TL2 may not be accurate. The RCAP2 registers may be read, but should not be written to, because a write might overlap a reload, and cause write and/or reload errors. The timer should be turned off (clear TR2) before accessing Timer 2 or RCAP2 registers.

Summary Of Baud Rate Equations: Timer 2 is in baud rate generating mode. If Timer 2 is being clocked through pin T2(P1.0) the baud rate is:

$$\text{Baud Rate} = \frac{\text{Timer 2 Overflow Rate}}{16}$$

If Timer 2 is being clocked internally, the baud rate is:

$$\text{Baud Rate} = \frac{f_{OSC}}{[\,n^{*} \times [65536 - (\text{RCAP2H, RCAP2L})]]}$$

* n = 16 in 6 clock mode
32 in 12 clock mode

Where f_{OSC} = Oscillator Frequency

To obtain the reload value for RCAP2H and RCAP2L, the above equation can be rewritten as:

$$\text{RCAP2H, RCAP2L} = 65536 - \left(\frac{f_{OSC}}{n^{*} \times \text{Baud Rate}}\right)$$

Timer/Counter 2 Set-up
Except for the baud rate generator mode, the values given for T2CON do not include the setting of the TR2 bit. Therefore, bit TR2 must be set, separately, to turn the timer on. See Table 11 for set-up of Timer 2 as a timer. Also see Table 12 for set-up of Timer 2 as a counter.

Table 11. Timer 2 as a Timer

MODE	T2CON	
	INTERNAL CONTROL (Note 1)	EXTERNAL CONTROL (Note 2)
16-bit Auto-Reload	00H	08H
16-bit Capture	01H	09H
Baud rate generator receive and transmit same baud rate	34H	36H
Receive only	24H	26H
Transmit only	14H	16H

Table 12. Timer 2 as a Counter

MODE	TMOD	
	INTERNAL CONTROL (Note 1)	EXTERNAL CONTROL (Note 2)
16-bit	02H	0AH
Auto-Reload	03H	0BH

NOTES:
1. Capture/reload occurs only on timer/counter overflow.
2. Capture/reload occurs on timer/counter overflow and a 1-to-0 transition on T2EX (P1.1) pin except when Timer 2 is used in the baud rate generator mode.

FULL-DUPLEX ENHANCED UART

Standard UART operation

A full-duplex serial port can transmit and receive simultaneously. It is also receive-buffered, meaning it can commence reception of a second byte before a previously received byte has been read from the register. (However, if the first byte still hasn't been read by the time reception of the second byte is complete, one of the bytes will be lost.) The serial port receive and transmit registers are both accessed at Special Function Register SBUF. Writing to SBUF loads the transmit register, and reading SBUF accesses a physically separate receive register.

The serial port can operate in 4 modes:

Mode 0: Serial data enters and exits through RxD. TxD outputs the shift clock. 8 bits are transmitted/received (LSB first). The baud rate is fixed at 1/12 the oscillator frequency in 12-clock mode or 1/6 the oscillator frequency in 6-clock mode.

Mode 1: 10 bits are transmitted (through TxD) or received (through RxD): a start bit (0), 8 data bits (LSB first), and a stop bit (1). On receive, the stop bit goes into RB8 in Special Function Register SCON. The baud rate is variable.

Mode 2: 11 bits are transmitted (through TxD) or received (through RxD): a start bit (0), 8 data bits (LSB first), a programmable 9th data bit, and a stop bit (1). On Transmit, 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. On receive, the 9th data bit goes into RB8 in Special Function Register SCON, while the stop bit is ignored. The baud rate is programmable to either 1/32 or 1/64 the oscillator frequency in 12-clock mode or 1/16 or 1/32 the oscillator frequency in 6-clock mode.

Mode 3: 11 bits are transmitted (through TxD) or received (through RxD): a start bit (0), 8 data bits (LSB first), a programmable 9th data bit, and a stop bit (1). In fact, Mode 3 is the same as Mode 2 in all respects except baud rate. The baud rate in Mode 3 is variable.

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.

Multiprocessor Communications

Modes 2 and 3 have a special provision for multiprocessor communications. In these modes, 9 data bits are received. The 9th bit goes into RB8. Then comes a stop bit. The port 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. A 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 will be coming. The slaves that weren't being addressed leave their SM2s set and go on about their business, ignoring the coming data bytes.

SM2 has no effect in Mode 0. In Mode 1, it can be used to check the validity of the stop bit. In a Mode 1 reception, if SM2 = 1, the receive interrupt will not be activated unless a valid stop bit is received.

Serial Port Control Register

The serial port control and status register is the Special Function Register SCON, shown in Figure 26. This register contains not only the mode selection bits, but also the 9th data bit for transmit and receive (TB8 and RB8), and the serial port interrupt bits (TI and RI).

Baud Rates

The baud rate in Mode 0 is fixed: Mode 0 Baud Rate = Oscillator Frequency / 12 (12-clock mode) or / 6 (6-clock mode). The baud rate in Mode 2 depends on the value of bit SMOD in Special Function Register PCON. If SMOD = 0 (which is the value on reset), and the port pins in 12-clock mode, the baud rate is 1/64 the oscillator frequency. If SMOD = 1, the baud rate is 1/32 the oscillator frequency. In 6-clock mode, the baud rate is 1/32 or 1/16 the oscillator frequency, respectively.

Mode 2 Baud Rate =

$$\frac{2^{SMOD}}{n} \times (\text{Oscillator Frequency})$$

Where:

n = 64 in 12-clock mode, 32 in 6-clock mode

The baud rates in Modes 1 and 3 are determined by the Timer 1 or Timer 2 overflow rate.

Using Timer 1 to Generate Baud Rates

When Timer 1 is used as the baud rate generator (T2CON.5 = 0, T2CON.4 = 0), the baud rates in Modes 1 and 3 are determined by the Timer 1 overflow rate and the value of SMOD as follows:

Mode 1, 3 Baud Rate =

$$\frac{2^{SMOD}}{n} \times (\text{Timer 1 Overflow Rate})$$

Where:

n = 32 in 12-clock mode, 16 in 6-clock mode

The Timer 1 interrupt should be disabled in this application. The Timer itself can be configured for either "timer" or "counter" operation, and in any of its 3 running modes. In the most typical applications, it is configured for "timer" operation, in the auto-reload mode (high nibble of TMOD = 0010B). In that case the baud rate is given by the formula:

Mode 1, 3 Baud Rate =

$$\frac{2^{SMOD}}{n} \times \frac{\text{Oscillator Frequency}}{12 \times [256-(TH1)]}$$

Where:

n = 32 in 12-clock mode, 16 in 6-clock mode

One can achieve very low baud rates with Timer 1 by leaving the Timer 1 interrupt enabled, and configuring the Timer to run as a 16-bit timer (high nibble of TMOD = 0001B), and using the Timer 1 interrupt to do a 16-bit software reload. Figure 27 lists various commonly used baud rates and how they can be obtained from Timer 1.

SCON Address = 98H Reset Value = 00H

Bit Addressable

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

Where SM0, SM1 specify the serial port mode, as follows:

SM0	SM1	Mode	Description	Baud Rate
0	0	0	shift register	$f_{OSC}/12$ (12-clock mode) or $f_{OSC}/6$ (6-clock mode)
0	1	1	8-bit UART	variable
1	0	2	9-bit UART	$f_{OSC}/64$ or $f_{OSC}/32$ (12-clock mode) or $f_{OSC}/32$ or $f_{OSC}/16$ (6-clock mode)
1	1	3	9-bit UART	variable

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 1, if SM2=1 then RI will not be activated if a valid stop bit was not received. In Mode 0, SM2 should be 0.

REN Enables serial reception. Set by software to enable reception. Clear by software to disable reception.

TB8 The 9th data bit that will be transmitted in Modes 2 and 3. Set or clear by software as desired.

RB8 In Modes 2 and 3, is the 9th data bit that was received. In Mode 1, it SM2=0, RB8 is the stop bit that was received. In Mode 0, RB8 is not used.

TI Transmit interrupt flag. Set by hardware at the end of the 8th bit time in Mode 0, or at the beginning of the stop bit in the other modes, in any serial transmission. Must be cleared by software.

RI Receive interrupt flag. Set by hardware at the end of the 8th bit time in Mode 0, or halfway through the stop bit time in the other modes, in any serial reception (except see SM2). Must be cleared by software.

SU01626

Figure 26. Serial Port Control (SCON) Register

Baud Rate			f_{OSC}	SMOD	Timer 1		
Mode	12-clock mode	6-clock mode			C/$\overline{T}$	Mode	Reload Value
Mode 0 Max	1.67 MHz	3.34 MHz	20 MHz	X	X	X	X
Mode 2 Max	625 k	1250 k	20 MHz	1	X	X	X
Mode 1, 3 Max	104.2 k	208.4 k	20 MHz	1	0	2	FFH
Mode 1, 3	19.2 k	38.4 k	11.059 MHz	1	0	2	FDH
	9.6 k	19.2 k	11.059 MHz	0	0	2	FDH
	4.8 k	9.6 k	11.059 MHz	0	0	2	FAH
	2.4 k	4.8 k	11.059 MHz	0	0	2	F4H
	1.2 k	2.4 k	11.059 MHz	0	0	2	E8H
	137.5	275	11.986 MHz	0	0	2	1DH
	110	220	6 MHz	0	0	2	72H
	110	220	12 MHz	0	0	1	FEEBH

Figure 27. Timer 1 Generated Commonly Used Baud Rates

More About Mode 0

Serial data enters and exits through RxD. TxD outputs the shift clock. Eight data bits are transmitted/received (LSB first). The baud rate is fixed at 1/12 the oscillator frequency (12-clock mode) or 1/6 the oscillator frequency (6-clock mode).

Figure 28 shows a simplified functional diagram of the serial port in Mode 0, and associated timing.

Transmission is initiated by any instruction that uses SBUF as a destination register. The “write to SBUF” signal at S6P2 also loads a 1 into the 9th position of the transmit shift register and tells the TX Control block to commence a transmission. The internal timing is such that one full machine cycle will elapse between “write to SBUF” and activation of SEND.

SEND enables the output of the shift register to the alternate output function line of P3.0 and also enable SHIFT CLOCK to the alternate output function line of P3.1. SHIFT CLOCK is low during S3, S4, and S5 of every machine cycle, and high during S6, S1, and S2. At S6P2 of every machine cycle in which SEND is active, the contents of the transmit shift are shifted to the right one position.

As data bits shift out to the right, zeros come in from the left. When the MSB of the data byte is at the output position of the shift register, then the 1 that was initially loaded into the 9th position, is just to the left of the MSB, and all positions to the left of that contain zeros. This condition flags the TX Control block to do one last shift and then deactivate SEND and set T1. Both of these actions occur at S1P1 of the 10th machine cycle after “write to SBUF.”

Reception is initiated by the condition REN = 1 and R1 = 0. At S6P2 of the next machine cycle, the RX Control unit writes the bits 11111110 to the receive shift register, and activates RECEIVE in the next clock phase.

RECEIVE enable SHIFT CLOCK to the alternate output function line of P3.1. SHIFT CLOCK makes transitions at S3P1 and S6P1 of every machine cycle. At S6P2 of every machine cycle in which RECEIVE is active, the contents of the receive shift register are

shifted to the left by one position. The value that comes in, from the right, is the value that was sampled at the P3.0 pin at S5P2 of the same machine cycle.

As data bits come in from the right, 1s shift out to the left. When the 0 that was initially loaded into the rightmost position arrives at the leftmost position in the shift register, it flags the RX Control block to do one last shift and load SBUF. At S1P1 of the 10th machine cycle, after the write to SCON that cleared RI, RECEIVE is cleared as RI is set.

More About Mode 1

Ten bits are transmitted (through TxD), or received (through RxD): a start bit (0), 8 data bits (LSB first), and a stop bit (1). On receive, the stop bit goes into RB8 in SCON. In the 80C51 the baud rate is determined by the Timer 1 or Timer 2 overflow rate.

Figure 29 shows a simplified functional diagram of the serial port in Mode 1, and associated timings for transmit receive.

Transmission is initiated by any instruction that uses SBUF as a destination register. The “write to SBUF” signal also loads a 1 into the 9th bit position of the transmit shift register and flags the TX Control unit that a transmission is requested. Transmission actually commences at S1P1 of the machine cycle following the next rollover in the divide-by-16 counter. (Thus, the bit times are synchronized to the divide-by-16 counter, not to the “write to SBUF” signal.)

The transmission begins with activation of SEND which puts the start bit at TxD. One bit time later, DATA is activated, which enables the output bit of the transmit shift register to TxD. The first shift pulse occurs one bit time after that.

As data bits shift out to the right, zeros are clocked in from the left. When the MSB of the data byte is at the output position of the shift register, the 1 that was initially loaded into the 9th position is just to the left of the MSB, and all positions to the left of that contain zeros. This condition flags the TX Control unit to do one last shift and then deactivate SEND and set TI. This occurs at the 10th divide-by-16 rollover after “write to SBUF.”

Reception is initiated by a detected 1-to-0 transition at RxD. For this purpose RxD is sampled at a rate of 16 times whatever baud rate has been established. When a transition is detected, the divide-by-16 counter is immediately reset, and 1FFH is written into the input shift register. Resetting the divide-by-16 counter aligns its rollovers with the boundaries of the incoming bit times.

The 16 states of the counter divide each bit time into 16ths. At the 7th, 8th, and 9th counter states of each bit time, 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 unit goes back to looking for another 1-to-0 transition. This is to provide 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.

As data bits come in from the right, 1s shift out to the left. When the start bit arrives at the leftmost position in the shift register (which in mode 1 is a 9-bit register), it flags the RX Control block to do one last shift, load SBUF and RB8, and set RI. 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:

1. R1 = 0, and
2. Either SM2 = 0, or the received stop bit = 1.

If either of these two conditions is not met, the received frame is irretrievably lost. If both conditions are met, the stop bit goes into RB8, the 8 data bits go into SBUF, and RI is activated. At this time, whether the above conditions are met or not, the unit goes back to looking for a 1-to-0 transition in RxD.

More About Modes 2 and 3

Eleven bits are transmitted (through TxD), or received (through RxD): a start bit (0), 8 data bits (LSB first), a programmable 9th data bit, and a stop bit (1). On transmit, the 9th data bit (TB8) can be assigned the value of 0 or 1. On receive, the 9th data bit goes into RB8 in SCON. The baud rate is programmable to either 1/32 or 1/64 (12-clock mode), or 1/16 or 1/32 (6-clock mode) of the oscillator frequency in Mode 2. Mode 3 may have a variable baud rate generated from Timer 1 or Timer 2.

Figures 30 and 31 show a functional diagram of the serial port in Modes 2 and 3. The receive portion is exactly the same as in Mode 1. The transmit portion differs from Mode 1 only in the 9th bit of the transmit shift register.

Transmission is initiated by any instruction that uses SBUF as a destination register. The “write to SBUF” signal also loads TB8 into the 9th bit position of the transmit shift register and flags the TX Control unit that a transmission is requested. Transmission commences at S1P1 of the machine cycle following the next rollover in the divide-by-16 counter (thus, the bit times are synchronized to the divide-by-16 counter, not to the “write to SBUF” signal).

The transmission begins with activation of SEND, which puts the start bit at TxD. One bit time later, DATA is activated, which enables the output bit of the transmit shift register to TxD. The first shift pulse occurs one bit time after that. The first shift clocks a 1 (the stop bit) into the 9th bit position of the shift register. Thereafter, only zeros are clocked in. Thus, as data bits shift out to the right, zeros are clocked in from the left. When TB8 is at the output position of the shift register, then the stop bit is just to the left of TB8, and all positions to the left of that contain zeros. This condition flags the TX Control unit to do one last shift and then deactivate SEND and set TI. This occurs at the 11th divide-by-16 rollover after “write to SBUF.”

Reception is initiated by a detected 1-to-0 transition at RxD. For this purpose RxD is sampled at a rate of 16 times whatever baud rate has been established. When a transition is detected, the divide-by-16 counter is immediately reset, and 1FFH is written to the input shift register.

At the 7th, 8th, and 9th counter states of each bit time, 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. If the value accepted during the first bit time is not 0, the receive circuits are reset and the unit goes back to looking for another 1-to-0 transition. If the start bit proves valid, it is shifted into the input shift register, and reception of the rest of the frame will proceed.

As data bits come in from the right, 1s shift out to the left. When the start bit arrives at the leftmost position in the shift register (which in Modes 2 and 3 is a 9-bit register), it flags the RX Control block to do one last shift, load SBUF and RB8, and set RI.

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:

1. RI = 0, and
2. Either SM2 = 0, or the received 9th data bit = 1.

If either of these conditions is not met, the received frame is irretrievably 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. One bit time later, whether the above conditions were met or not, the unit goes back to looking for a 1-to-0 transition at the RxD input.

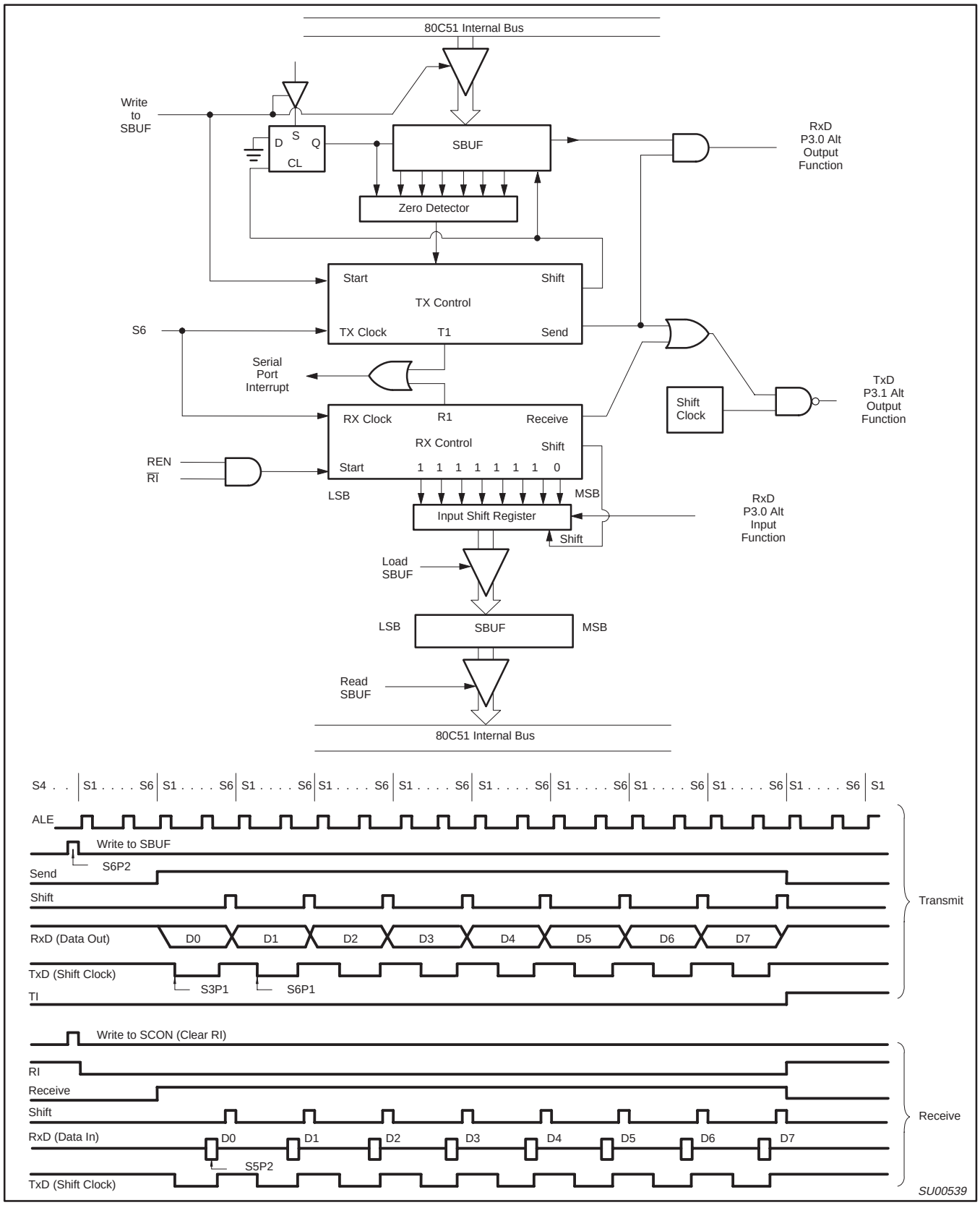


Figure 28. Serial Port Mode 0

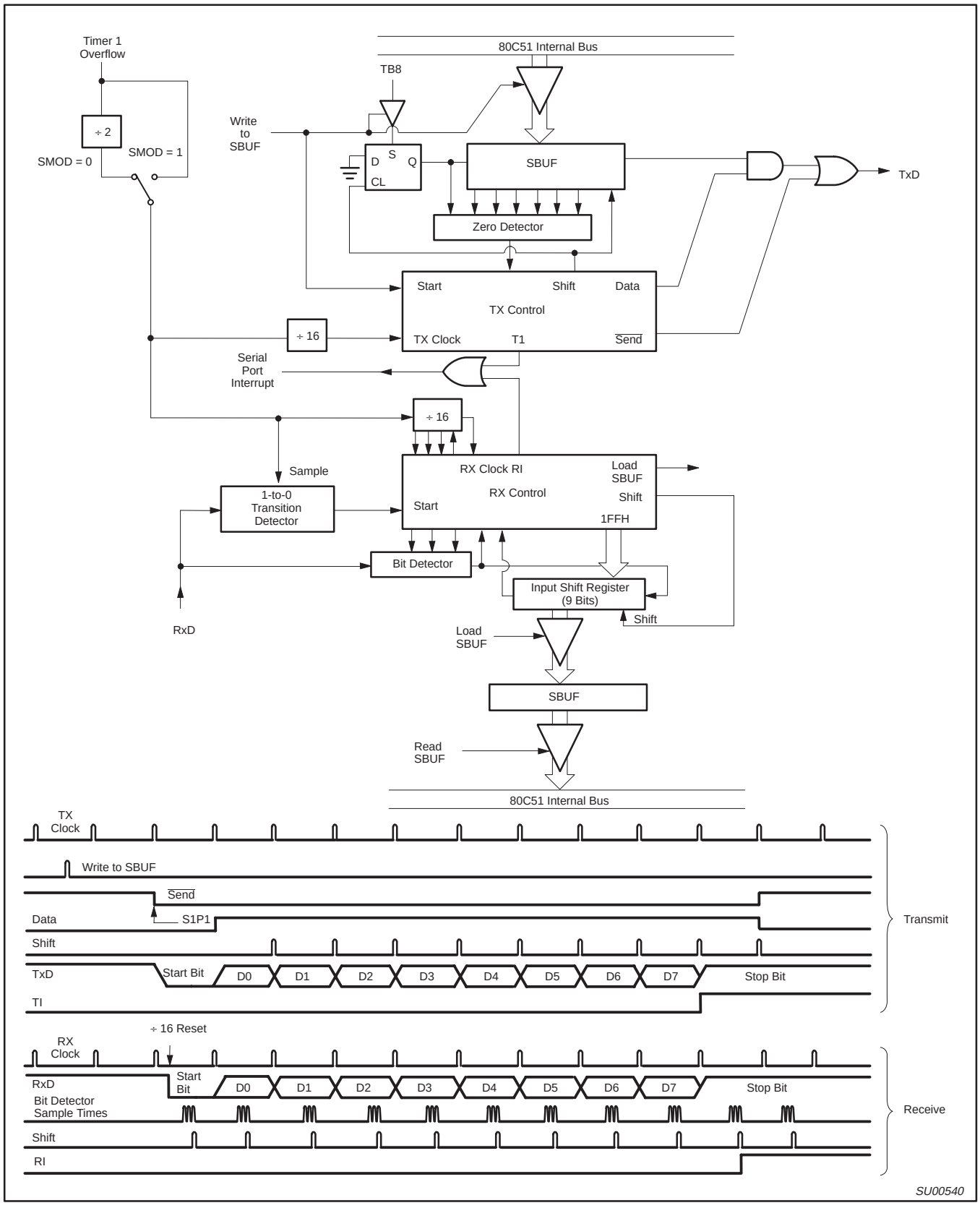


Figure 29. Serial Port Mode 1

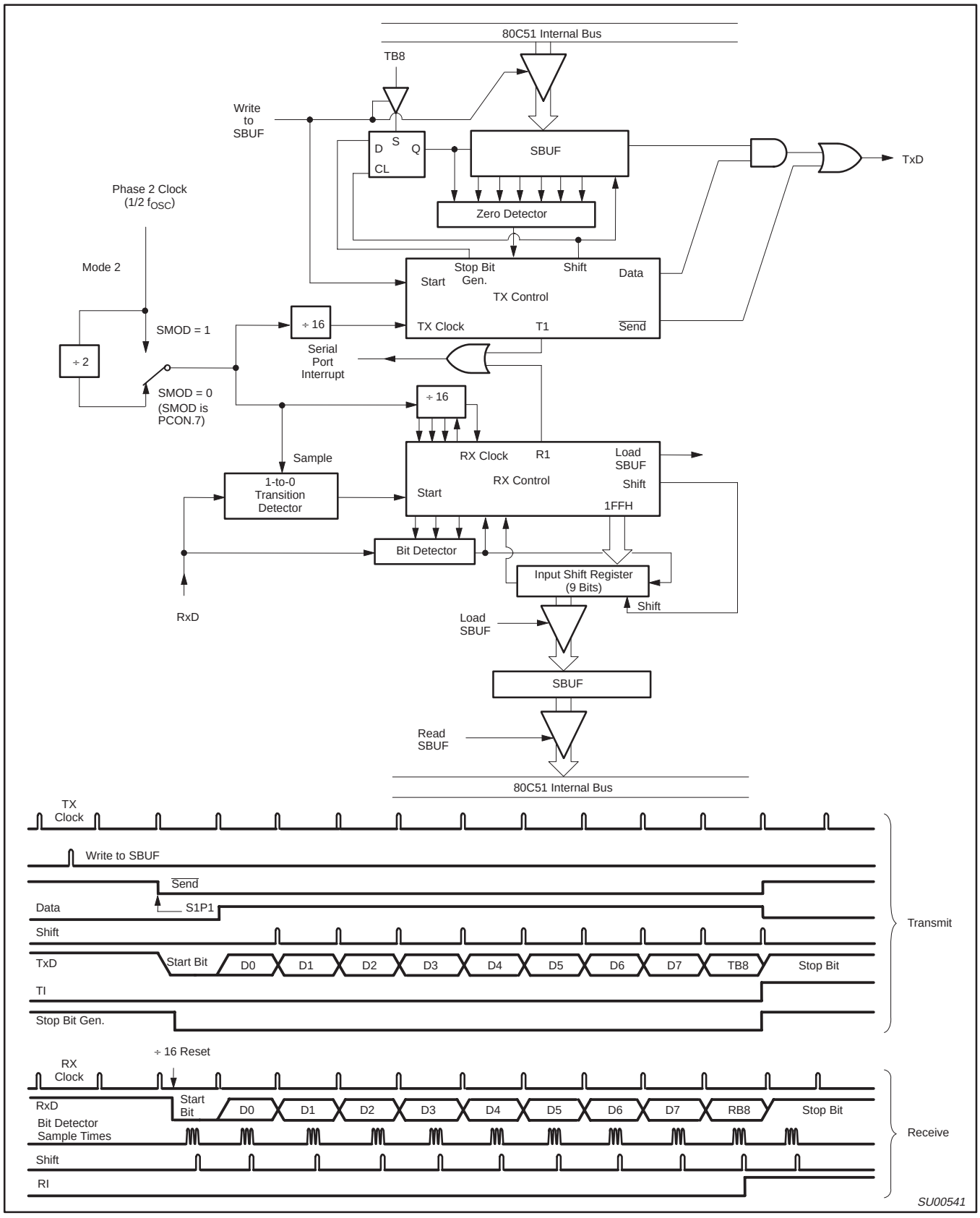


Figure 30. Serial Port Mode 2

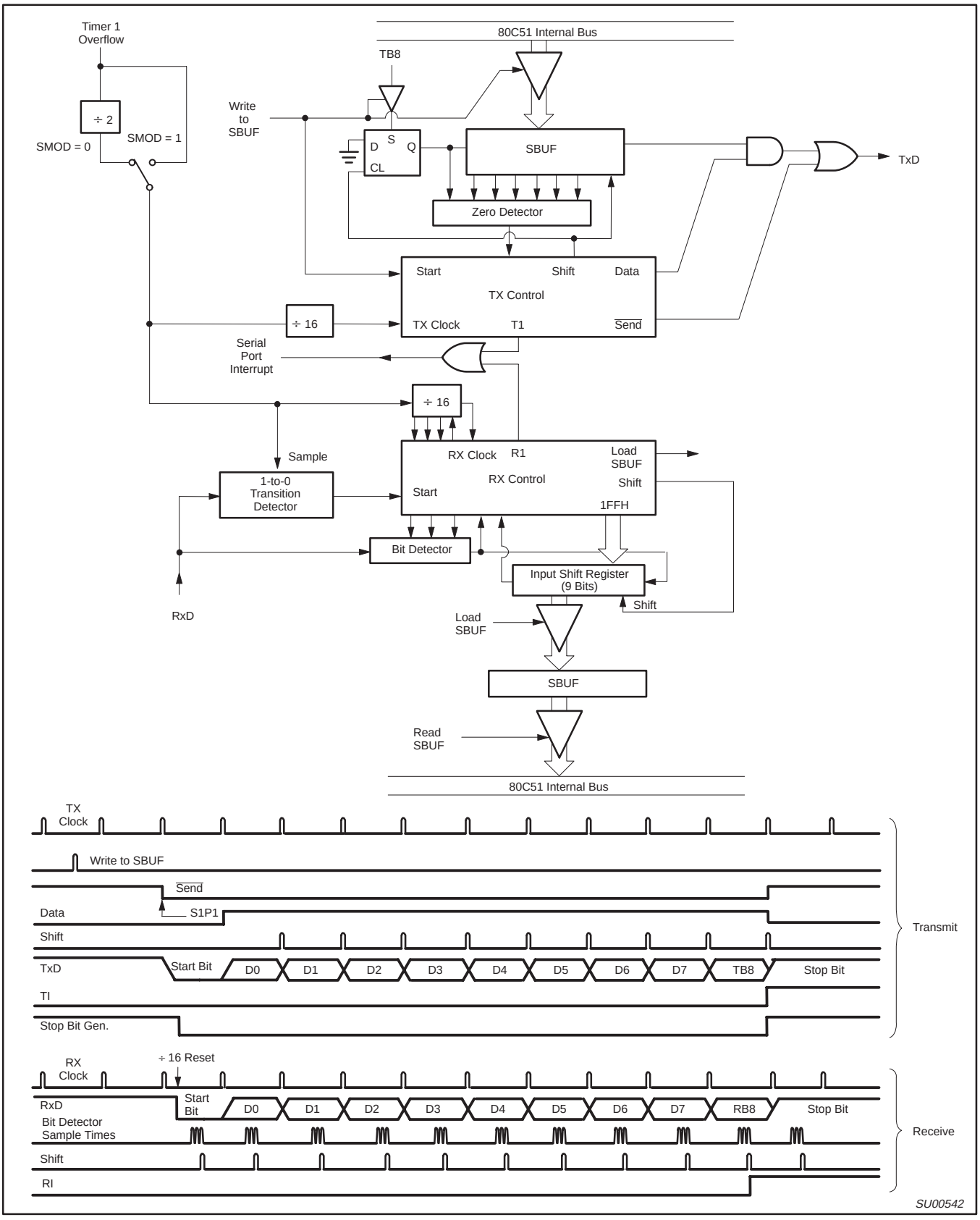


Figure 31. Serial Port Mode 3

Enhanced UART

In addition to the standard operation, the UART can perform framing error detect by looking for missing stop bits, and automatic address recognition. The UART also fully supports multiprocessor communication as does the standard 80C51 UART.

When used for framing error detect, the UART looks for missing stop bits in the communication. A missing bit will set the FE bit in the S0CON register. The FE bit shares the S0CON.7 bit with SM0, and the function of S0CON.7 is determined by PCON.6 (SMOD0) (see Figure 32). If SMOD0 is set then S0CON.7 functions as FE. S0CON.7 functions as SM0 when SMOD0 is cleared. When used as FE, S0CON.7 can only be cleared by software (refer to Figure 33).

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 S0CON. 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. Automatic address recognition is shown in Figure 34.

The 8-bit mode is called Mode 1. In this mode, the RI flag will be set if SM2 is enabled and the information received has a valid stop bit following the 8 address bits, and the information is either a Given or Broadcast address.

Mode 0 is the Shift Register mode and SM2 is ignored.

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:

Slave 0	SADDR	=	1100 0000
	SADEN	=	1111 1101
	Given	=	1100 00X0
Slave 1	SADDR	=	1100 0000
	SADEN	=	1111 1110
	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:

Slave 0	SADDR	=	1100 0000
	SADEN	=	1111 1001
	Given	=	1100 0XX0
Slave 1	SADDR	=	1110 0000
	SADEN	=	1111 1010
	Given	=	1110 0X0X
Slave 2	SADDR	=	1110 0000
	SADEN	=	1111 1100
	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 trended as don’t-cares. In most cases, interpreting the don’t-cares as ones, the broadcast address will be FF hexadecimal.

Upon reset, SADDR (SFR address 0A9H) and SADEN (SFR address 0B9H) are leaded 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 80C51 type UART drivers which do not make use of this feature.

S0CON Address = 98H Reset Value = 0000 0000B

Bit Addressable

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

(SMOD0 = 0/1)*

Symbol	Function
FE	Framing Error bit. This bit is set by the receiver when an invalid stop bit is detected. The FE bit is not cleared by valid frames but should be cleared by software. The SMOD0 bit must be set to enable access to the FE bit.
SM0	Serial Port Mode Bit 0, (SMOD0 must = 0 to access bit SM0)
SM1	Serial Port Mode Bit 1

SM0	SM1	Mode	Description	Baud Rate**
0	0	0	shift register	f_{OSC}/6 (6 clock mode) or f_{OSC}/12 (12 clock mode)
0	1	1	8-bit UART	variable
1	0	2	9-bit UART	f_{OSC}/32 or f_{OSC}/16 (6 clock mode) or f_{OSC}/64 or f_{OSC}/32 (12 clock mode)
1	1	3	9-bit UART	variable

SM2	Enables the Automatic Address Recognition feature in Modes 2 or 3. If SM2 = 1 then RI will not be set unless the received 9th data bit (RB8) is 1, indicating an address, and the received byte is a Given or Broadcast Address. In Mode 1, if SM2 = 1 then RI will not be activated unless a valid stop bit was received, and the received byte is a Given or Broadcast Address. In Mode 0, SM2 should be 0.
REN	Enables serial reception. Set by software to enable reception. Clear by software to disable reception.
TB8	The 9th data bit that will be transmitted in Modes 2 and 3. Set or clear by software as desired.
RB8	In modes 2 and 3, the 9th data bit that was received. In Mode 1, if SM2 = 0, RB8 is the stop bit that was received. In Mode 0, RB8 is not used.
TI	Transmit interrupt flag. Set by hardware at the end of the 8th bit time in Mode 0, or at the beginning of the stop bit in the other modes, in any serial transmission. Must be cleared by software.
RI	Receive interrupt flag. Set by hardware at the end of the 8th bit time in Mode 0, or halfway through the stop bit time in the other modes, in any serial reception (except see SM2). Must be cleared by software.

NOTE:
*SMOD0 is located at PCON6.
**f_{OSC} = oscillator frequency

SU01451

Figure 32. S0CON: Serial Port Control Register

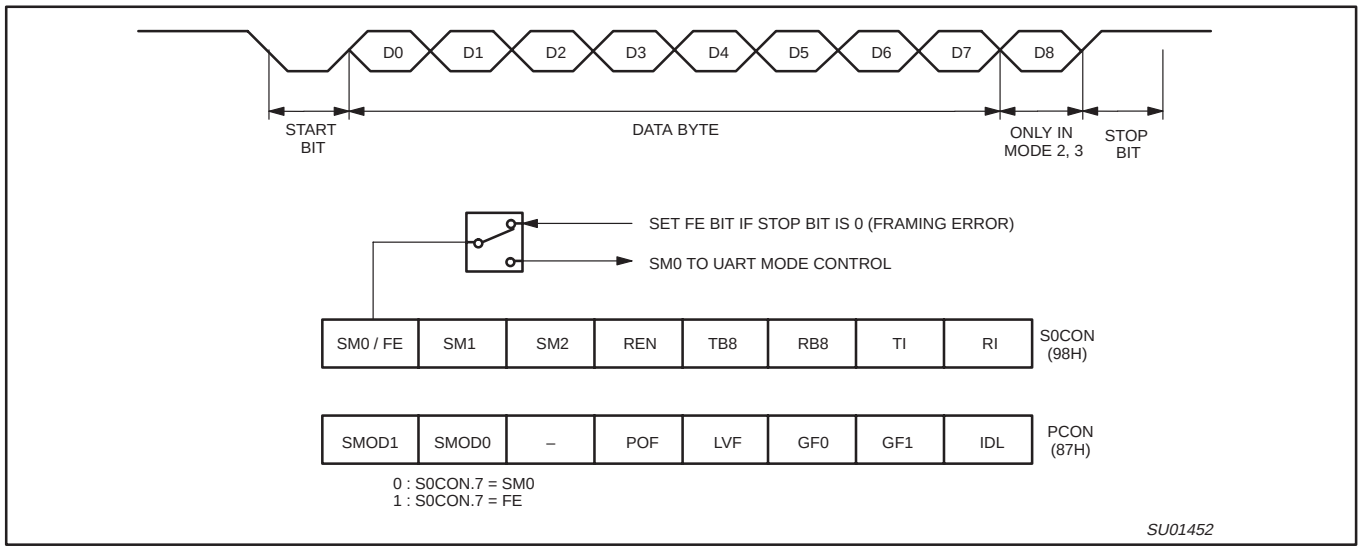


Figure 33. UART Framing Error Detection

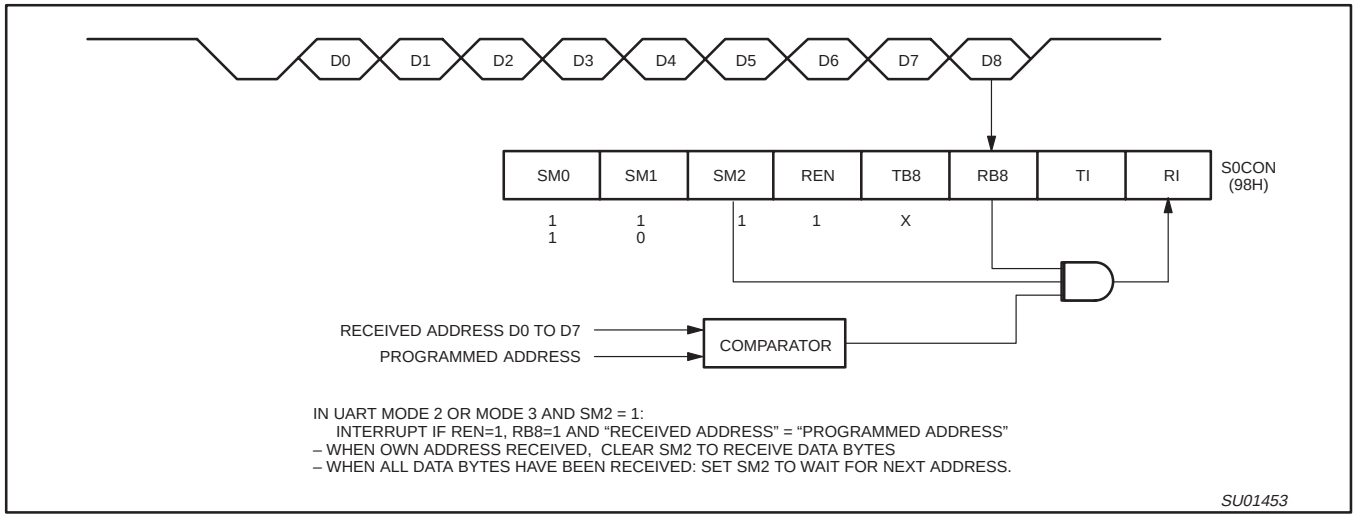


Figure 34. UART Multiprocessor Communication, Automatic Address Recognition

Interrupt Priority Structure

The P89C660/662/664/668 has an 8 source four-level interrupt structure (see Table 13).

There are 4 SFRs associated with the four-level interrupt. They are the IE, IP, IEN1, and IPH (see Figures 35, 36, 37, and 38). The IPH (Interrupt Priority High) register makes the four-level interrupt structure possible. The IPH is located at SFR address B7H. The structure of the IPH register and a description of its bits is shown in Figure 37.

The function of the IPH SFR, when combined with the IP SFR, determines the priority of each interrupt. The priority of each interrupt is determined as shown in the following table:

PRIORITY BITS		INTERRUPT PRIORITY LEVEL
IPH.x	IP.x	
0	0	Level 0 (lowest priority)
0	1	Level 1
1	0	Level 2
1	1	Level 3 (highest priority)

The priority scheme for servicing the interrupts is the same as that for the 80C51, except that there are four interrupt levels rather than two (as on the 80C51). An interrupt will be serviced as long as an interrupt of equal or higher priority is not already being serviced. If an interrupt of equal or higher level priority is being serviced, the new interrupt will wait until it is finished before being serviced. If a lower priority level interrupt is being serviced, it will be stopped and the new interrupt serviced. When the new interrupt is finished, the lower priority level interrupt that was stopped will be completed.

Table 13. Interrupt Table

SOURCE	POLLING PRIORITY	REQUEST BITS	HARDWARE CLEAR?	VECTOR ADDRESS
X0	1	IE0	N (L)[1] Y (T)[2]	03H
SI01 (I^2C)	2	—	N	2BH
T0	3	TP0	Y	0BH
X1	4	IE1	N (L) Y (T)	13H
T1	5	TF1	Y	1BH
SP	6	RI, TI	N	23H
T2	7	TF2, EXF2	N	3BH
PCA	8	CF, CCFn n = 0–4	N	33H

NOTES:
1. L = Level activated
2. T = Transition activated

	7	6	5	4	3	2	1	0
IEN0 (0A8H)	EA	EC	ES1	ES0	ET1	EX1	ET0	EX0

Enable Bit = 1 enables the interrupt.
Enable Bit = 0 disables it.

BIT	SYMBOL	FUNCTION
IEN0.7	EA	Global disable bit. If EA = 0, all interrupts are disabled. If EA = 1, each interrupt can be individually enabled or disabled by setting or clearing its enable bit.
IEN0.6	EC	PCA interrupt enable bit
IEN0.5	ES1	I^2C interrupt enable bit.
IEN0.4	ES0	Serial Port interrupt enable bit.
IEN0.3	ET1	Timer 1 interrupt enable bit.
IEN0.2	EX1	External interrupt 1 enable bit.
IEN0.1	ET0	Timer 0 interrupt enable bit.
IEN0.0	EX0	External interrupt 0 enable bit.

SU01454

Figure 35. IE Registers

	7	6	5	4	3	2	1	0
IP (0B8H)	PT2	PPC	PS1	PS0	PT1	PX1	PT0	PX0

Priority Bit = 1 assigns high priority
Priority Bit = 0 assigns low priority

BIT	SYMBOL	FUNCTION
IP.7	PT2	Timer 2 interrupt priority bit.
IP.6	PPC	PCA interrupt priority bit
IP.5	PS1	Serial I/O1 (I^2C) interrupt priority bit.
IP.4	PS0	Serial Port interrupt priority bit.
IP.3	PT1	Timer 1 interrupt priority bit.
IP.2	PX1	External interrupt 1 priority bit.
IP.1	PT0	Timer 0 interrupt priority bit.
IP.0	PX0	External interrupt 0 priority bit.

SU01455

Figure 36. IP Registers

	7	6	5	4	3	2	1	0
IPH (B7H)	PT2H	PPCH	PS1H	PS0H	PT1H	PX1H	PT0H	PX0H

Priority Bit = 1 assigns higher priority
Priority Bit = 0 assigns lower priority

BIT	SYMBOL	FUNCTION
IPH.7	PT2H	Timer 2 interrupt priority bit high.
IPH.6	PPCH	PCA interrupt priority bit
IPH.5	PS1H	Serial I/O (I^2C) interrupt priority bit high.
IPH.4	PS0H	Serial Port interrupt priority bit high.
IPH.3	PT1H	Timer 1 interrupt priority bit high.
IPH.2	PX1H	External interrupt 1 priority bit high.
IPH.1	PT0H	Timer 0 interrupt priority bit high.
IPH.0	PX0H	External interrupt 0 priority bit high.

SU01456

Figure 37. IPH Registers

	7	6	5	4	3	2	1	0
IEN1 (E8H)	—	—	—	—	—	—	—	ET2

Enable Bit = 1 enables the interrupt
Enable Bit = 0 disables the interrupt

BIT	SYMBOL	FUNCTION
IEN1.7	—	
IEN1.6	—	
IEN1.5	—	
IEN1.4	—	
IEN1.3	—	
IEN1.2	—	
IEN1.1	—	
IEN1.0	ET2	Timer 2 interrupt enable bit.

SU01095

Figure 38. IEN1 Registers

Reduced EMI Mode

The AO bit (AUXR.0) in the AUXR register when set disables the ALE output.

Reduced EMI Mode

AUXR (8EH)

7	6	5	4	3	2	1	0
–	–	–	–	–	–	EXTRAM	AO

AUXR.1	EXTRAM	(See more detailed description in
AUXR.0	AO	Figure 53.)

Dual DPTR

The dual DPTR structure (see Figure 39) is a way by which the chip will specify the address of an external data memory location. There are two 16-bit DPTR registers that address the external memory, and a single bit called DPS (AUXR1.0), that allows the program code to switch between them.

- New Register Name: AUXR1#
- SFR Address: A2H
- Reset Value: xxxxx0x0B

AUXR1 (A2H)

7	6	5	4	3	2	1	0
–	–	ENBOOT	–	GF2	0	–	DPS

Where:

DPS (AUXR1.0), enables switching between DPTR0 and DPTR1.

Select Reg	DPS
DPTR0	0
DPTR1	1

The DPS bit status should be saved by software when switching between DPTR0 and DPTR1.

The GF2 bit is a general purpose user-defined flag. Note that bit 2 is not writable and is always read as a zero. This allows the DPS bit to be quickly toggled simply by executing an INC AUXR1 instruction without affecting the GF2 bit.

The ENBOOT bit determines whether the BOOTROM is enabled or disabled. This bit will automatically be set if the status byte is non zero during reset or $\overline{PSEN}$ is pulled low, ALE floats high, and EA > V_{IH} on the falling edge of reset. Otherwise, this bit will be cleared during reset.

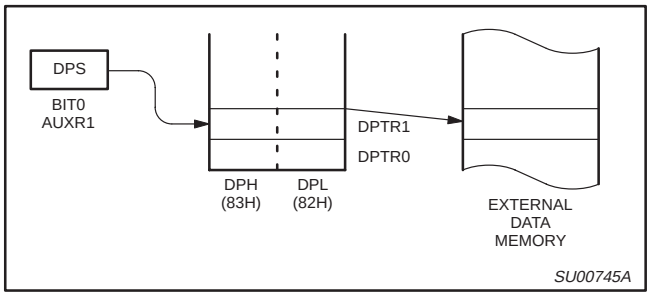


Figure 39.

DPTR Instructions

The instructions, that refer to DPTR, refer to the data pointer that is currently selected by the DPS bit (AUXR1.0). The six instructions that use the DPTR are as follows:

INC DPTR	Increments the data pointer by 1
MOV DPTR, #data16	Loads the DPTR with a 16-bit constant
MOV A, @ A+DPTR	Move code byte relative to DPTR to ACC
MOVX A, @ DPTR	Move external RAM (16-bit address) to ACC
MOVX @ DPTR , A	Move ACC to external RAM (16-bit address)
JMP @ A + DPTR	Jump indirect relative to DPTR

The data pointer can be accessed on a byte-by-byte basis by specifying the low or high byte in an instruction which accesses the SFRs. See application note AN458 for more details.

Programmable Counter Array (PCA)

The Programmable Counter Array available on the 89C66x is a special 16-bit Timer that has five 16-bit capture/compare modules associated with it. Each of the modules can be programmed to operate in one of four modes: rising and/or falling edge capture, software timer, high-speed output, or pulse width modulator. Each module has a pin associated with it in port 1. Module 0 is connected to P1.3(CEX0), module 1 to P1.4(CEX1), etc. The basic PCA configuration is shown in Figure 40.

The PCA timer is a common time base for all five modules and can be programmed to run at: 1/6 the oscillator frequency, 1/2 the oscillator frequency, the Timer 0 overflow, or the input on the ECI pin (P1.2). The timer count source is determined from the CPS1 and CPS0 bits in the CMOD SFR as follows (see Figure 43):

CPS1	CPS0	PCA Timer Count Source
0	0	1/6 oscillator frequency (6 clock mode); 1/12 oscillator frequency (12 clock mode)
0	1	1/2 oscillator frequency (6 clock mode); 1/4 oscillator frequency (12 clock mode)
1	0	Timer 0 overflow
1	1	External Input at ECI pin

In the CMOD SFR, there are three additional bits associated with the PCA. They are CIDL which allows the PCA to stop during idle mode, WDTE which enables or disables the watchdog function on module 4, and ECF which, when set, causes an interrupt and the PCA overflow flag CF (in the CCON SFR) to be set when the PCA timer overflows. These functions are shown in Figure 41.

The watchdog timer function is implemented in module 4 (see Figure 50).

The CCON SFR contains the run control bit for the PCA, and the flags for the PCA timer (CF) and each module (refer to Figure 44). To run the PCA the CR bit (CCON.6) must be set by software. The PCA is shut off by clearing this bit. The CF bit (CCON.7) is set when the PCA counter overflows and an interrupt will be generated if the ECF bit in the CMOD register is set, The CF bit can only be cleared by software. Bits 0 through 4 of the CCON register are the flags for the modules (bit 0 for module 0, bit 1 for module 1, etc.) and are set by hardware when either a match or a capture occurs. These flags also can only be cleared by software. The PCA interrupt system is shown in Figure 42.

Each module in the PCA has a special function register associated with it. These registers are: CCAPM0 for module 0, CCAPM1 for module 1, etc. (see Figure 45). The registers contain the bits that control the mode that each module will operate in. The ECCF bit (CCAPMn.0 where n = 0, 1, 2, 3, or 4 depending on the module) enables the CCF flag in the CCON SFR to generate an interrupt when a match or compare occurs in the associated module. PWM (CCAPMn.1) enables the pulse width modulation mode. The TOG bit (CCAPMn.2), when set, causes the CEX output associated with the module to toggle when there is a match between the PCA counter and the module's capture/compare register. The match bit MAT (CCAPMn.3), when set, will cause the CCFn bit in the CCON register to be set when there is a match between the PCA counter and the module's capture/compare register.

The next two bits CAPN (CCAPMn.4) and CAPP (CCAPMn.5) determine the edge that a capture input will be active on. The CAPN bit enables the negative edge, and the CAPP bit enables the positive edge. If both bits are set, both edges will be enabled and a capture will occur for either transition. The last bit ECOM (CCAPMn.6), when set, enables the comparator function. Figure 46 shows the CCAPMn settings for the various PCA functions.

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

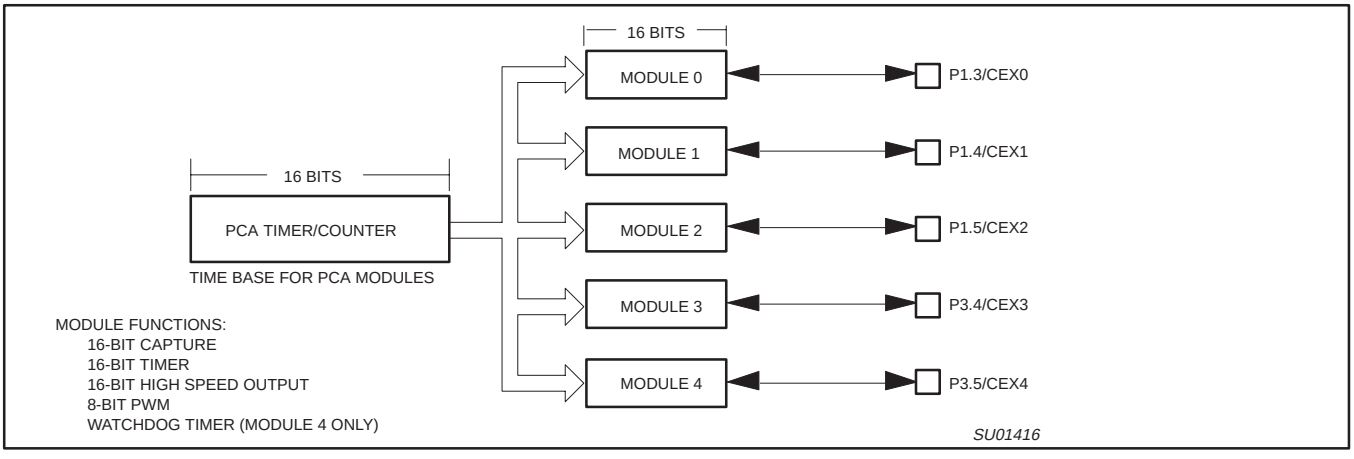


Figure 40. Programmable Counter Array (PCA)

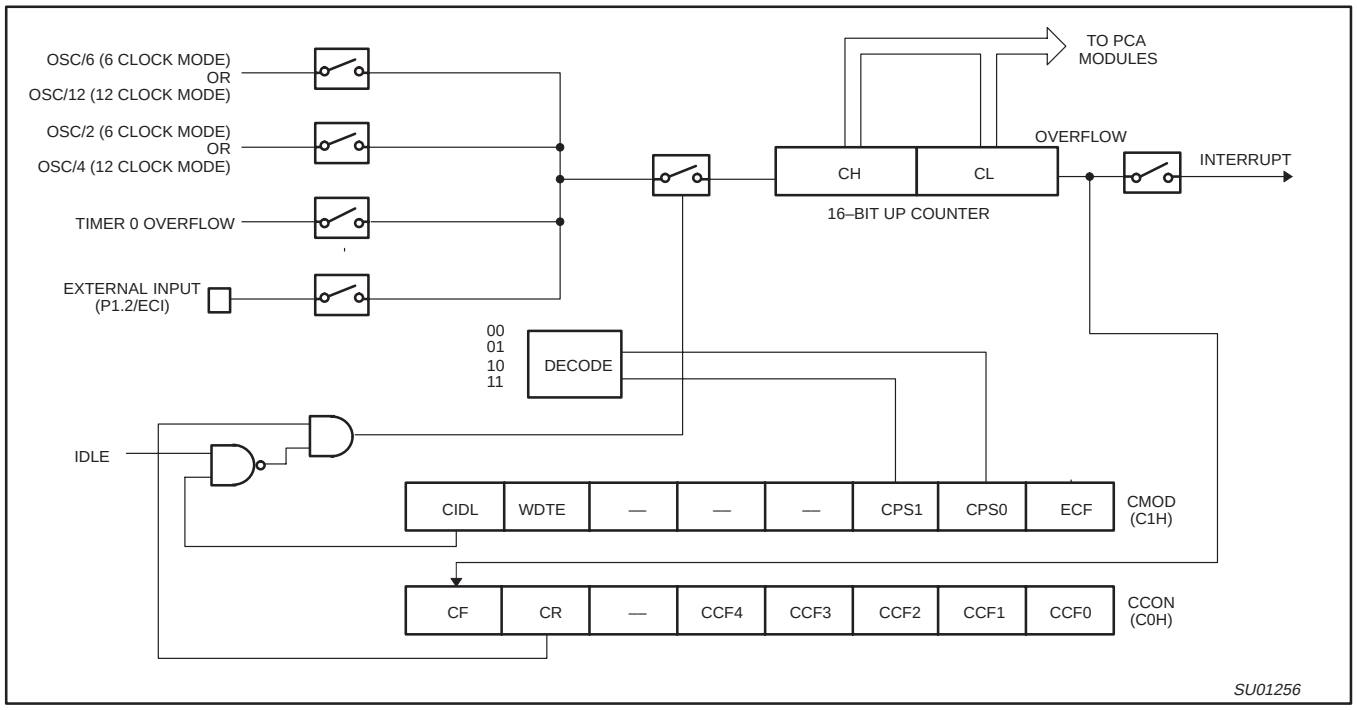


Figure 41. PCA Timer/Counter

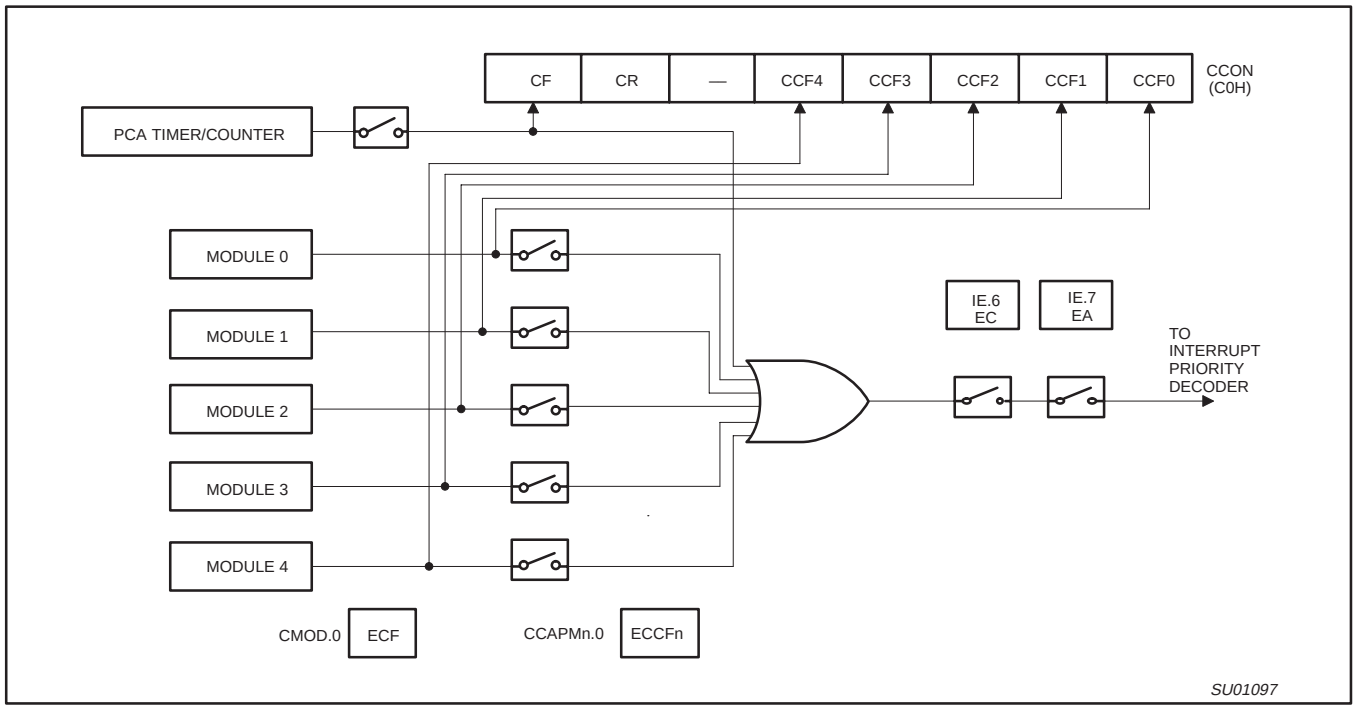


Figure 42. PCA Interrupt System

CMOD Address = C1H Reset Value = 00XX X000B

	CIDL	WDTE	–	–	–	CPS1	CPS0	ECF
Bit:	7	6	5	4	3	2	1	0

Symbol	Function
CIDL	Counter Idle control: CIDL = 0 programs the PCA Counter to continue functioning during idle Mode. CIDL = 1 programs it to be gated off during idle.
WDTE	Watchdog Timer Enable: WDTE = 0 disables Watchdog Timer function on PCA Module 4. WDTE = 1 enables it.
–	Not implemented, reserved for future use.*
CPS1	PCA Count Pulse Select bit 1.
CPS0	PCA Count Pulse Select bit 0.

CPS1	CPS0	Selected PCA Input**	
0	0	0	Internal clock, f_{OSC}/6 in 6 clock mode (f_{OSC}/12 in 12 clock mode)
0	1	1	Internal clock, f_{OSC}/2 in 6 clock mode (f_{OSC}/4 in 12 clock mode)
1	0	2	Timer 0 overflow
1	1	3	External clock at ECI/P1.2 pin (max. rate = f_{OSC}/4 in 6 clock mode, f_{OCS}/8 in 12 clock mode)

ECF	PCA Enable Counter Overflow interrupt: ECF = 1 enables CF bit in CCON to generate an interrupt. ECF = 0 disables that function of CF.

NOTE:
* User software should not write 1s to reserved bits. These bits may be used in future 8051 family products to invoke new features. In that case, the reset or inactive value of the new bit will be 0, and its active value will be 1. The value read from a reserved bit is indeterminate.
** f_{OSC} = oscillator frequency

SU01257

Figure 43. CMOD: PCA Counter Mode Register

CCON Address = 0C0H Reset Value = 00X0 0000B

Bit Addressable

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

Symbol	Function
CF	PCA Counter Overflow flag. Set by hardware when the counter rolls over. CF flags an interrupt if bit ECF in CMOD is set. CF may be set by either hardware or software but can only be cleared by software.
CR	PCA Counter Run control bit. Set by software to turn the PCA counter on. Must be cleared by software to turn the PCA counter off.
–	Not implemented, reserved for future use*.
CCF4	PCA Module 4 interrupt flag. Set by hardware when a match or capture occurs. Must be cleared by software.
CCF3	PCA Module 3 interrupt flag. Set by hardware when a match or capture occurs. Must be cleared by software.
CCF2	PCA Module 2 interrupt flag. Set by hardware when a match or capture occurs. Must be cleared by software.
CCF1	PCA Module 1 interrupt flag. Set by hardware when a match or capture occurs. Must be cleared by software.
CCF0	PCA Module 0 interrupt flag. Set by hardware when a match or capture occurs. Must be cleared by software.

NOTE:
* User software should not write 1s to reserved bits. These bits may be used in future 8051 family products to invoke new features. In that case, the reset or inactive value of the new bit will be 0, and its active value will be 1. The value read from a reserved bit is indeterminate.

SU01099

Figure 44. CCON: PCA Counter Control Register

CCAPMn Address CCAPM0 0C2H
CCAPM1 0C3H
CCAPM2 0C4H
CCAPM3 0C5H
CCAPM4 0C6H

Reset Value = X000 0000B

Not Bit Addressable

	–	ECOMn	CAPPn	CAPNn	MATn	TOGn	PWMn	ECCFn
Bit:	7	6	5	4	3	2	1	0

Symbol	Function
–	Not implemented, reserved for future use*.
ECOMn	Enable Comparator. ECOMn = 1 enables the comparator function.
CAPPn	Capture Positive, CAPPn = 1 enables positive edge capture.
CAPNn	Capture Negative, CAPNn = 1 enables negative edge capture.
MATn	Match. When MATn = 1, a match of the PCA counter with this module's compare/capture register causes the CCFn bit in CCON to be set, flagging an interrupt.
TOGn	Toggle. When TOGn = 1, a match of the PCA counter with this module's compare/capture register causes the CEXn pin to toggle.
PWMn	Pulse Width Modulation Mode. PWMn = 1 enables the CEXn pin to be used as a pulse width modulated output.
ECCFn	Enable CCF interrupt. Enables compare/capture flag CCFn in the CCON register to generate an interrupt.

NOTE:
*User software should not write 1s to reserved bits. These bits may be used in future 8051 family products to invoke new features. In that case, the reset or inactive value of the new bit will be 0, and its active value will be 1. The value read from a reserved bit is indeterminate.

SU01100

Figure 45. CCAPMn: PCA Modules Compare/Capture Registers

–	ECOMn	CAPPn	CAPNn	MATn	TOGn	PWMn	ECCFn	MODULE FUNCTION
X	0	0	0	0	0	0	0	No operation
X	X	1	0	0	0	0	X	16-bit capture by a positive-edge trigger on CEXn
X	X	0	1	0	0	0	X	16-bit capture by a negative trigger on CEXn
X	X	1	1	0	0	0	X	16-bit capture by a transition on CEXn
X	1	0	0	1	0	0	X	16-bit Software Timer
X	1	0	0	1	1	0	X	16-bit High Speed Output
X	1	0	0	0	0	1	0	8-bit PWM
X	1	0	0	1	X	0	X	Watchdog Timer

Figure 46. PCA Module Modes (CCAPMn Register)

PCA Capture Mode
To use one of the PCA modules in the capture mode, either one or both of the CCAPM bits CAPN and CAPP for that module must be set. The external CEX input for the module (on port 1) is sampled for a transition. When a valid transition occurs, the PCA hardware loads the value of the PCA counter registers (CH and CL) into the module's capture registers (CCAPnL and CCAPnH). If the CCFn bit for the module in the CCON SFR and the ECCFn bit in the CCAPMn SFR are set, then an interrupt will be generated (refer to Figure 47).

16-bit Software Timer Mode
The PCA modules can be used as software timers by setting both the ECOM and MAT bits in the modules CCAPMn register. The PCA timer will be compared to the module's capture registers, and when a match occurs, an interrupt will occur if the CCFn (CCON SFR) and the ECCFn (CCAPMn SFR) bits for the module are both set (see Figure 48).

High Speed Output Mode
In this mode, the CEX output (on port 1) associated with the PCA module will toggle each time a match occurs between the PCA counter and the module's capture registers. To activate this mode, the TOG, MAT, and ECOM bits in the module's CCAPMn SFR must be set (see Figure 49).

Pulse Width Modulator Mode
All of the PCA modules can be used as PWM outputs. Figure 50 shows the PWM function. The frequency of the output depends on the source for the PCA timer. All of the modules will have the same frequency of output because they all share the PCA timer. The duty cycle of each module is independently variable by using the module's capture register CCAPLn. When the value of the PCA CL SFR is less than the value in the module's CCAPLn SFR, the output will be low. When it is equal to or greater than, the output will be high. When CL overflows from FF to 00, CCAPLn is reloaded with the value in CCAPHn. This allows PWM update without glitches. The PWM and ECOM bits in the module's CCAPMn register must be set to enable the PWM mode.

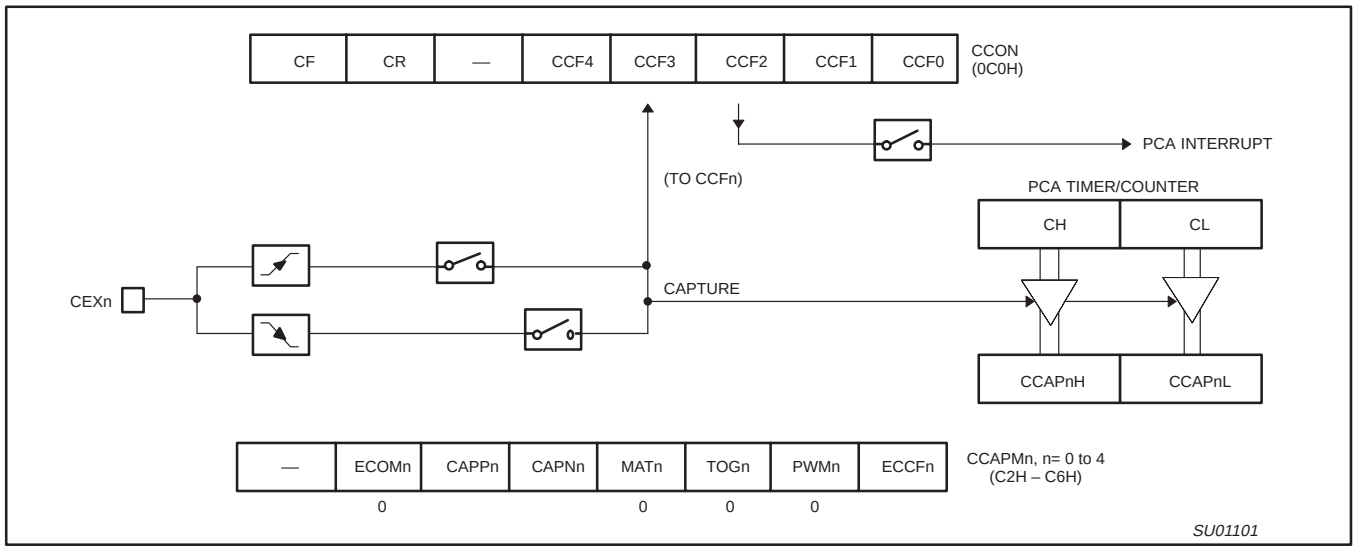


Figure 47. PCA Capture Mode

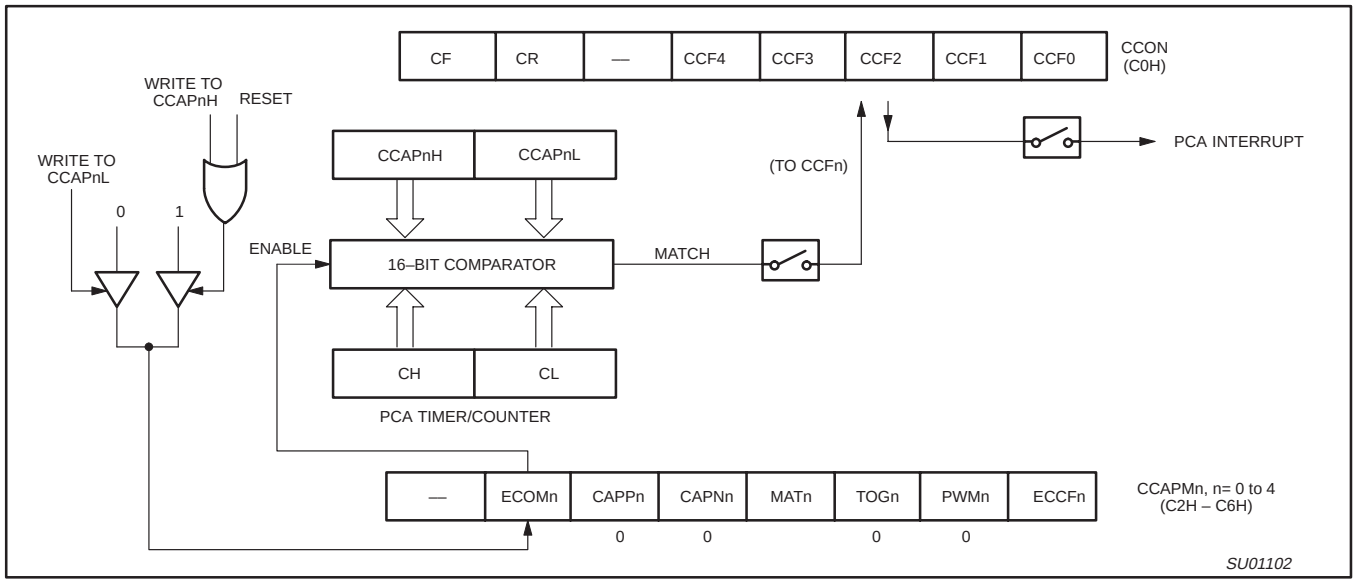


Figure 48. PCA Compare Mode

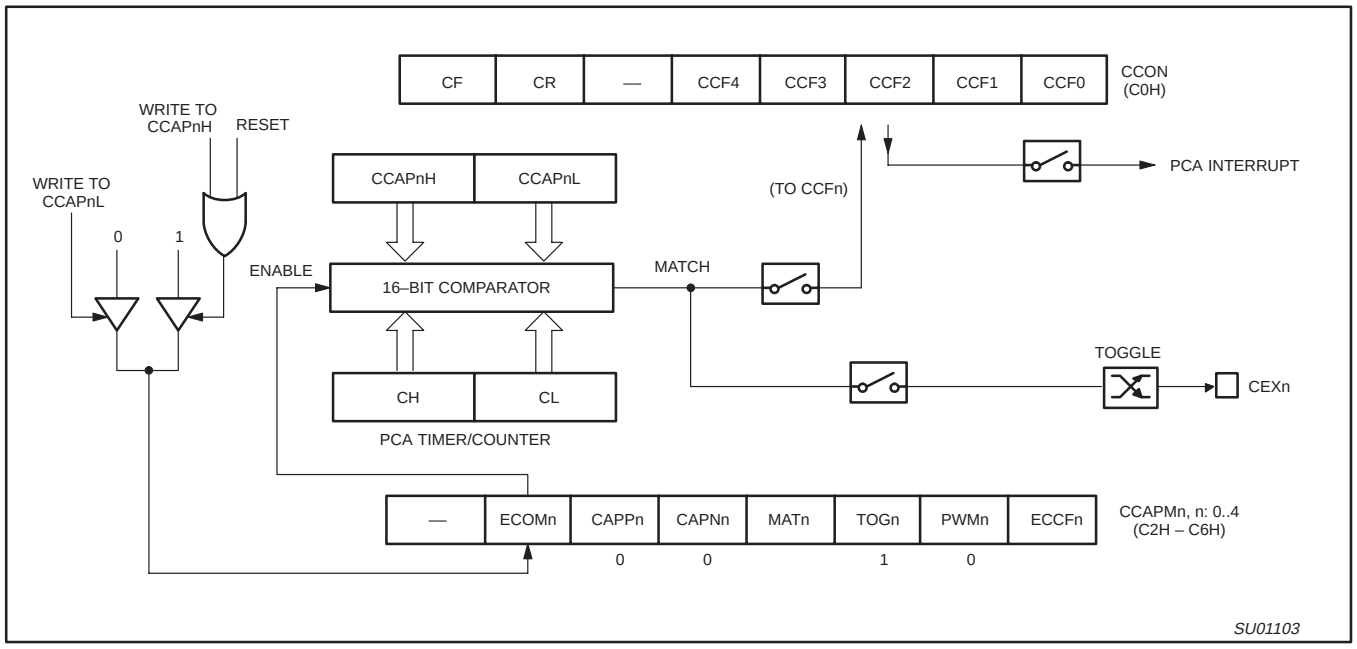


Figure 49. PCA High Speed Output Mode

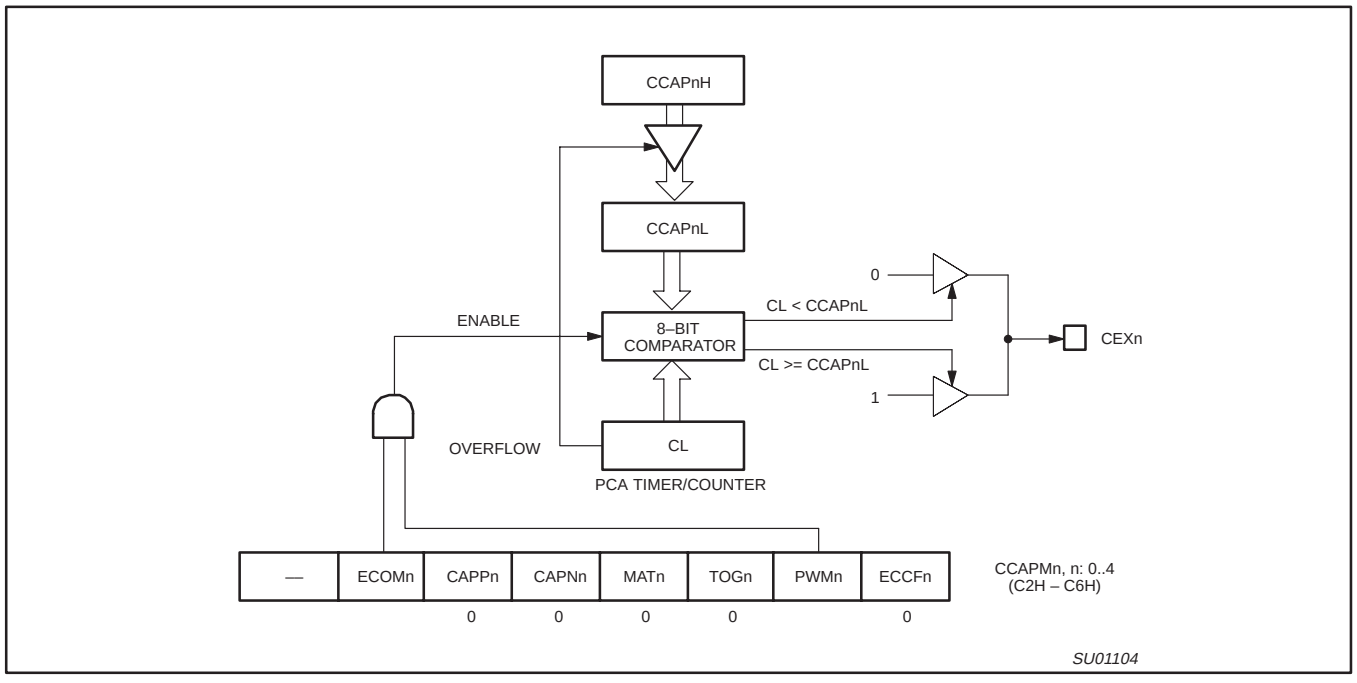


Figure 50. PCA PWM Mode

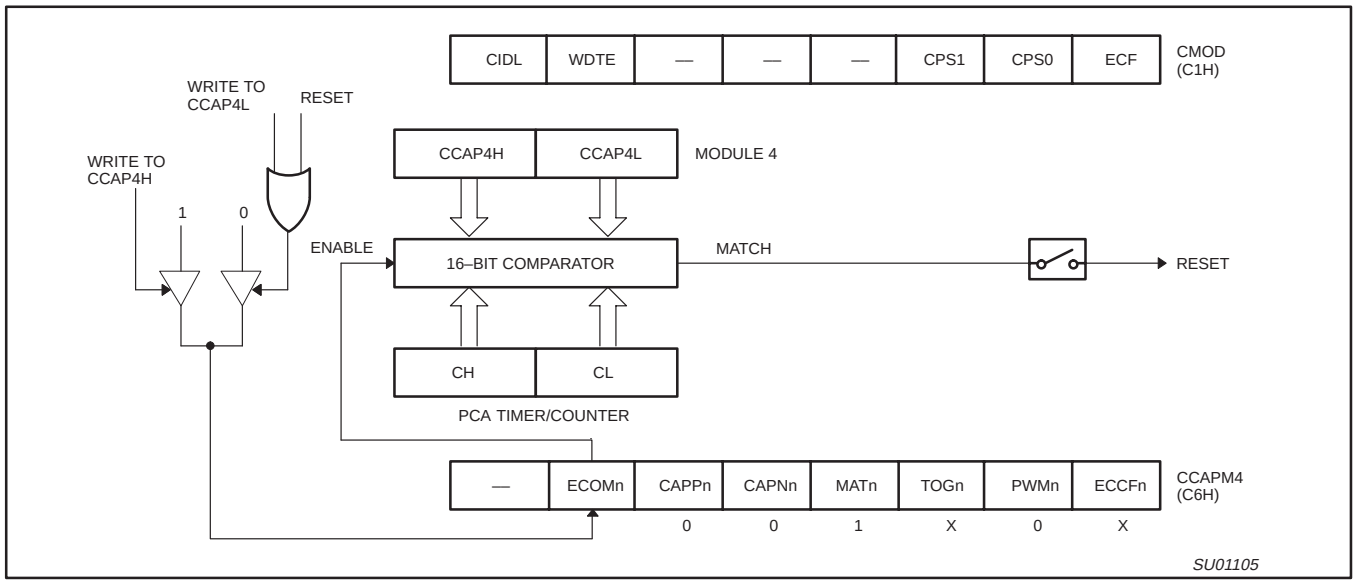


Figure 51. PCA Watchdog Timer m(Module 4 only)

PCA Watchdog Timer

An on-board watchdog timer is available with the PCA to improve the reliability of the system without increasing chip count. Watchdog timers are useful for systems that are susceptible to noise, power glitches, or electrostatic discharge. Module 4 is the only PCA module that can be programmed as a watchdog. However, this module can still be used for other modes if the watchdog is not needed.

Figure 51 shows a diagram of how the watchdog works. The user pre-loads a 16-bit value in the compare registers. Just like the other compare modes, this 16-bit value is compared to the PCA timer value. If a match is allowed to occur, an internal reset will be generated. This will not cause the RST pin to be driven high.

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

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

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

Figure 52 shows the code for initializing the watchdog timer. Module 4 can be configured in either compare mode, and the WDTE bit in CMOD must also be set. The user's software must periodically change (CCAP4H,CCAP4L) to keep a match from occurring with the PCA timer (CH,CL). This code is given in the WATCHDOG routine in Figure 52.

This routine should not be part of an interrupt service routine, because if the program counter goes astray and gets stuck in an infinite loop, interrupts will still be serviced and the watchdog will keep getting reset. Thus, the purpose of the watchdog would be defeated. Instead, call this subroutine from the main program within 2^{16} count of the PCA timer.

```
INIT_WATCHDOG:
   MOV CCAPM4, #4CH         ; Module 4 in compare mode
   MOV CCAP4L, #0FFH        ; Write to low byte first
   MOV CCAP4H, #0FFH        ; Before PCA timer counts up to
                            ; FFFF Hex, these compare values
                            ; must be changed
   ORL CMOD, #40H           ; Set the WDTE bit to enable the
                            ; watchdog timer without changing
                            ; the other bits in CMOD
;
;*******************************************************************
;
; Main program goes here, but CALL WATCHDOG periodically.
;
;*******************************************************************
;
WATCHDOG:
   CLR EA                   ; Hold off interrupts
   MOV CCAP4L, #00          ; Next compare value is within
   MOV CCAP4H, CH           ; 255 counts of the current PCA
   SETB EA                  ; timer value
   RET
```

Figure 52. PCA Watchdog Timer Initialization Code

Expanded Data RAM Addressing

The P89C660/662/664/668 has internal data memory that is mapped into four separate segments: the lower 128 bytes of RAM, upper 128 bytes of RAM, 128 bytes Special Function Register (SFR), and 256 bytes expanded RAM (ERAM) (256 bytes for the '660; 768 bytes for the '662; 1792 bytes for the '664; 7936 bytes for the '668).

The four segments are:

1. The Lower 128 bytes of RAM (addresses 00H to 7FH) are directly and indirectly addressable.
2. The Upper 128 bytes of RAM (addresses 80H to FFH) are indirectly addressable only.
3. The Special Function Registers, SFRs, (addresses 80H to FFH) are directly addressable only.
4. The 256/768/1792/7936-bytes expanded RAM (ERAM, 00H – XFFH/2FFH/6FFH/1FFFH) are indirectly accessed by move external instruction, MOVX, and with the EXTRAM bit cleared, see Figure 53.

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

When an instruction accesses an internal location above address 7FH, the CPU knows whether the access is to the upper 128 bytes of data RAM, or to SFR space by the addressing mode used in the instruction. Instructions that use direct addressing, access SFR space. For example:

MOV 0A0H,A

accesses the SFR at location 0A0H (which is P2). Instructions that use indirect addressing, access the Upper 128 bytes of data RAM. For example:

MOV @R0,A

where R0 contains 0A0H, accesses the data byte at address 0A0H, rather than P2 (whose address is 0A0H).

The ERAM can be accessed by indirect addressing, with EXTRAM bit cleared and MOVX instructions. This part of memory is physically located on-chip, logically occupies the first 256 bytes (660), 768 (662), 1792 (664), 7936 (668) of external data memory.

With EXTRAM = 0, the ERAM is indirectly addressed, using the MOVX instruction in combination with any of the registers R0, R1 of the selected bank or DPTR. An access to ERAM will not affect ports P0, P3.6 (WR#) and P3.7 (RD#). P2 SFR is in output state during external addressing. For example, with EXTRAM = 0,

MOVX @R0,A

where R0 contains 0A0H, access the ERAM at address 0A0H rather than external memory. An access to external data memory locations higher than the ERAM will be performed with the MOVX DPTR instructions in the same way as in the standard 80C51 (with P0 and P2 as data/address bus, and P3.6 and P3.7 as write and read timing signals. Refer to Figure 54).

With EXTRAM = 1, MOVX @Ri and MOVX @DPTR will be similar to the standard 80C51. MOVX @ Ri will provide an 8-bit address multiplexed with data on Port 0 and any output port pins can be used to output higher order address bits. This is to provide the external paging capability. MOVX @DPTR will generate a 16-bit address. Port 2 outputs the high-order eight address bits (the contents of DPH) while Port 0 multiplexes the low-order eight address bits (the contents of DPL) with data. MOVX @Ri and MOVX @DPTR will generate either read or write signals on P3.6 ($\overline{WR}$) and P3.7 ($\overline{RD}$).

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

AUXR Address = 8EH Reset Value = xxxx xx10B

Not Bit Addressable

	—	—	—	—	—	—	**EXTRAM**	**AO**
Bit:	7	6	5	4	3	2	1	0

Symbol	Function
AO	Disable/Enable ALE
	AO **Operating Mode**
	0 ALE is emitted at a constant rate of $^1/_3$ the oscillator frequency (6 clock mode; $^1/_6$ f_{OSC} in 12 clock mode)
	1 ALE is active only during off-chip memory access.
EXTRAM	Internal/External RAM access using MOVX @Ri/@DPTR
	EXTRAM **Operating Mode**
	0 Internal ERAM access using MOVX @Ri/@DPTR
	1 External data memory access.
—	Not implemented, reserved for future use*.

NOTE:
*User software should not write 1s to reserved bits. These bits may be used in future 8051 family products to invoke new features. In that case, the reset or inactive value of the new bit will be 0, and its active value will be 1. The value read from a reserved bit is indeterminate.

SU01711

Figure 53. AUXR: Auxiliary Register

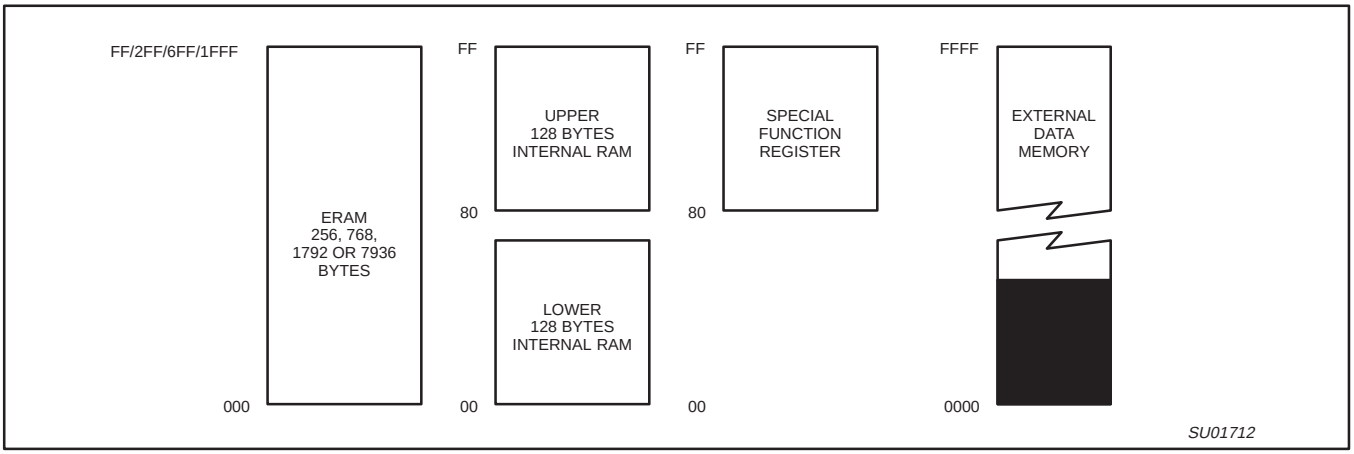


Figure 54. Internal and External Data Memory Address Space with EXTRAM = 0

Hardware WatchDog Timer (One-Time Enabled with Reset-Out for P89C660/662/664/668)

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

Using the WDT

To enable the WDT, user must write 01EH and 0E1H in sequence to the WDTRST (SFR location 0A6H). When WDT is enabled, the user needs to service it by writing 01EH and 0E1H to WDTRST to avoid WDT overflow. The 14-bit counter overflows when it reaches 16383 (3FFFH) and this will reset the device. When WDT is enabled, it will increment every machine cycle while the oscillator is running. This means the user must reset the WDT at least every 16383 machine cycles. To reset the WDT, the user must write 01EH and 0E1H to WDTRST. WDTRST is a write only register. The WDT counter cannot be read or written. When the WDT overflows, it will generate an output RESET pulse at the RST pin. The RESET pulse duration is $98 \times T_{OSC}$ (6 clock mode; 196 in 12 clock mode), where $T_{OSC} = 1/f_{OSC}$. To make the best use of the WDT, it should be serviced in those sections of code that will periodically be executed within the time required to prevent a WDT reset.

FLASH EPROM MEMORY

GENERAL DESCRIPTION

The P89C660/662/664/668 Flash memory augments EPROM functionality with in-circuit electrical erasure and programming. The Flash can be read and written as bytes. The Chip Erase operation will erase the entire program memory. The Block Erase function can erase any Flash byte block. In-System Programming and standard parallel programming are both available. On-chip erase and write timing generation contribute to a user-friendly programming interface.

The P89C660/662/664/668 Flash reliably stores memory contents even after 10,000 erase and program cycles. The cell is designed to optimize the erase and programming mechanisms. In addition, the combination of advanced tunnel oxide processing and low internal electric fields for erase and programming operations, produces reliable cycling. The P89C660/662/664/668 uses a +5 V V_{PP} supply to perform the Program/Erase algorithms.

FEATURES – IN-SYSTEM PROGRAMMING (ISP) AND IN-APPLICATION PROGRAMMING (IAP)

- Flash EPROM internal program memory with Block Erase.
- Internal 1 kbyte fixed boot ROM, containing low-level in-system programming routines and a default serial loader. User program can call these routines to perform In-Application Programming (IAP). The Boot ROM can be turned off to provide access to the full 64 kbyte of Flash memory.
- Boot vector allows user provided Flash loader code to reside anywhere in the Flash memory space. This configuration provides flexibility to the user.
- Default loader in Boot ROM allows programming via the serial port without the need for a user provided loader.
- Up to 64 kbytes of external program memory if the internal program memory is disabled ($\overline{EA}$ = 0).
- Programming and erase voltage +5 V (+12 V tolerant).
- Read/Programming/Erase using ISP/IAP:
  - Byte Programming (20 μs).
  - Typical quick erase times:
    Block Erase (8 kbytes or 16 kbytes) in 10 seconds.
    Full Erase (64 kbytes) in 20 seconds.
- In-System Programming.
- Programmable security for the code in the Flash.
- 10,000 minimum erase/program cycles for each byte.
- 10-year minimum data retention.

CAPABILITIES OF THE PHILIPS 89C51 FLASH-BASED MICROCONTROLLERS

Flash organization

The P89C660/662/664/668 contains 16KB/32KB/64KB of Flash program memory. This memory is organized as 5 separate blocks. The first two blocks are 8 kbytes in size, filling the program memory space from address 0 through 3FFF hex. The final three blocks are 16 kbytes in size and occupy addresses from 4000 through FFFF hex.

Figure 55 depicts the Flash memory configurations.

Flash Programming and Erasure

There are three methods of erasing or programming of the Flash memory that may be used. First, the Flash may be programmed or erased in the end-user application by calling low-level routines through a common entry point in the Boot ROM. The end-user application, though, must be executing code from a different block than the block that is being erased or programmed. Second, the on-chip ISP boot loader may be invoked. This ISP boot loader will, in turn, call low-level routines through the same common entry point in the Boot ROM that can be used by the end-user application. Third, the Flash may be programmed or erased using the parallel method by using a commercially available EPROM programmer. The parallel programming method used by these devices is similar to that used by EPROM 87C51, but it is not identical, and the commercially available programmer will need to have support for these devices.

Boot ROM

When the microcontroller programs its own Flash memory, all of the low level details are handled by code that is permanently contained in a 1 kbyte “Boot ROM” that is separate from the Flash memory. A user program simply calls the common entry point with appropriate parameters in the Boot ROM to accomplish the desired operation. Boot ROM operations include things like: erase block, program byte, verify byte, program security lock bit, etc. The Boot ROM overlays the program memory space at the top of the address space from FC00 to FFFF hex, when it is enabled. The Boot ROM may be turned off so that the upper 1 kbytes of Flash program memory are accessible for execution.

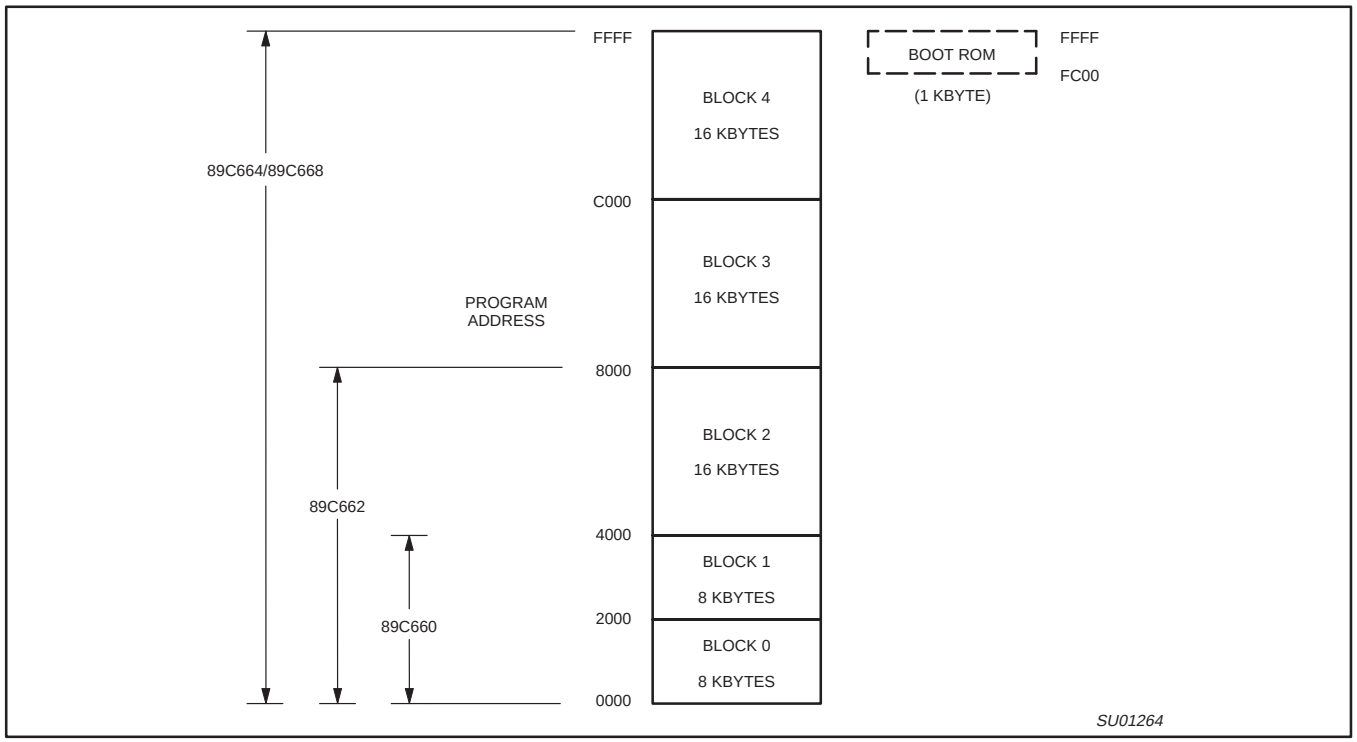


Figure 55. Flash Memory Configurations

Power-On Reset Code Execution

The P89C660/662/664/668 contains two special Flash registers: the BOOT VECTOR and the STATUS BYTE. At the falling edge of reset, the P89C660/662/664/668 examines the contents of the Status Byte. If the Status Byte 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 Status Byte is set to a value other than zero, 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 setting is 0FCH, corresponds to the address 0FC00H for the factory masked-ROM ISP boot loader. A custom boot loader can be written with the Boot Vector set to the custom boot loader.

> **NOTE:** When erasing the Status Byte or Boot Vector, both bytes are erased at the same time. It is necessary to reprogram the Boot Vector after erasing and updating the Status Byte.

Hardware Activation of the Boot Loader

The boot loader can also be executed by holding PSEN LOW, P2.7, P2.6 high, $\overline{EA}$ greater than V_{IH} (such as +5 V), and ALE HIGH (or not connected) at the falling edge of RESET. This is the same effect as having a non-zero status byte. This allows an application to be built that will normally execute the end user's code but can be manually forced into ISP operation.

If the factory default setting for the Boot Vector (0FCH) is changed, it will no longer point to the ISP masked-ROM boot loader code. If this happens, the only possible way to change the contents of the Boot Vector is through the parallel 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 Status Byte.

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.

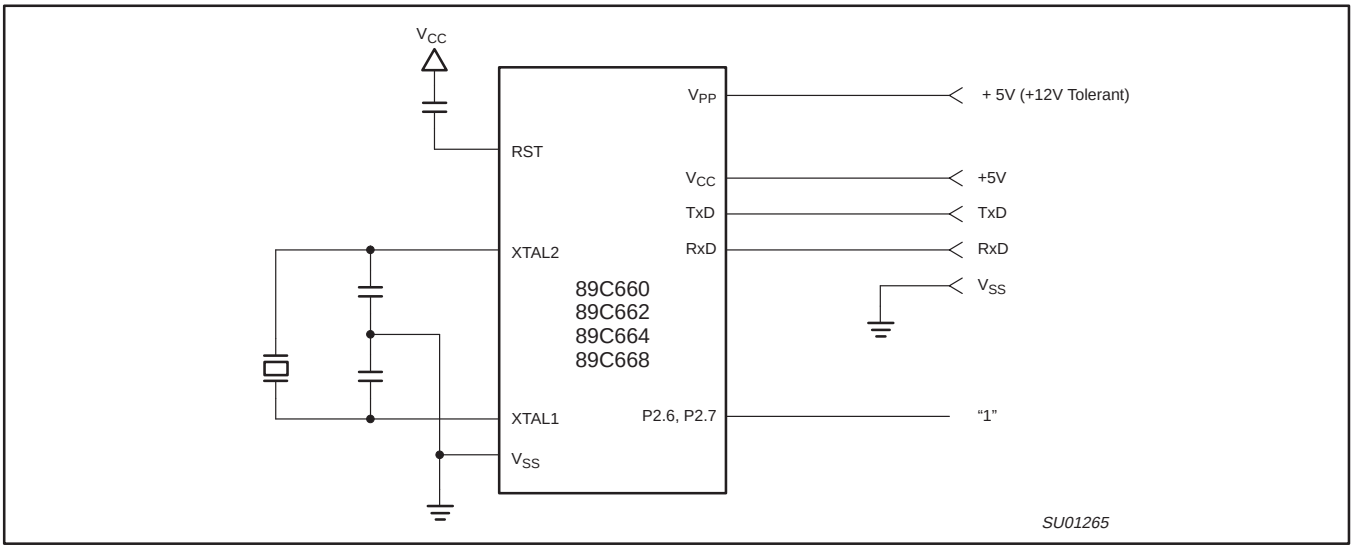


Figure 56. In-System Programming with a Minimum of Pins

In-System Programming (ISP)

The In-System Programming (ISP) is performed without removing the microcontroller from the system. The In-System Programming (ISP) facility consists of a series of internal hardware resources coupled with internal firmware to facilitate remote programming of the P89C660/662/664/668 through the serial port. This firmware is provided by Philips and embedded within each P89C660/662/664/668 device.

The Philips In-System Programming (ISP) 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: TxD, RxD, V_{SS}, V_{CC}, and V_{PP} (see Figure 56). Only a small connector needs to be available to interface your application to an external circuit in order to use this feature. The V_{PP} supply should be adequately decoupled and V_{PP} not allowed to exceed datasheet limits.

Free ISP software is available on the Philips web site: "WinISP"

1. Direct your browser to the following page: **http://www.semiconductors.philips.com/products/standard/microcontrollers/download/80c51/flash/**
2. Download "WinISP.exe"
3. Execute WinISP.exe to install the software

Free ISP software is also available from the Embedded Systems Academy: "FlashMagic"

1. Direct your browser to the following page: **http://www.esacademy.com/software/flashmagic/**
2. Download Flashmagic
3. Execute "flashmagic.exe" to install the software

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 P89C660/662/664/668 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 P89C660/662/664/668 will accept up to 16 (10H) 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 16 (decimal). ISP commands are summarized in Table 14.

As a record is received by the P89C660/662/664/668, 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 P89C660/662/664/668 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 (displaying the contents of the internal program memory is an exceptions).

In the case of a Data Record (record type 00), an additional check is made. A "." character will NOT be sent unless the record checksum matched the calculated checksum and all of the bytes in the record were successfully programmed. For a data record, an "X" indicates that the checksum failed to match, and an "R" indicates that one of the bytes did not properly program. It is necessary to send a type 02 record (specify oscillator frequency) to the P89C660/662/664/668 before programming data.

The ISP facility was designed so that specific crystal frequencies were not required in order to generate baud rates or time the programming pulses. The user thus needs to provide the P89C660/662/664/668 with information required to generate the proper timing. Record type 02 is provided for this purpose.

Table 14. Intel-Hex Records Used by In-System Programming

RECORD TYPE	COMMAND/DATA FUNCTION

00	Program Data
	`:nnaaaa00dd....ddcc`
	Where:
	`Nn = number of bytes (hex) in record`
	`Aaaa = memory address of first byte in record`
	`dd....dd = data bytes`
	`cc = checksum`
	Example:
	`:10008000AF5F67F0602703E0322CFA92007780C3FD`

01	End of File (EOF), no operation
	`:xxxxxx01cc`
	Where:
	`xxxxxx = required field, but value is a “don’t care”`
	`cc = checksum`
	Example:
	`:00000001FF`

02	Specify Oscillator Frequency
	`:01xxxx02ddcc`
	Where:
	`xxxx = required field, but value is a “don’t care”`
	`dd = integer oscillator frequency rounded down to nearest MHz`
	`cc = checksum`
	Example:
	`:0100000210ED (dd = 10h = 16, used for 16.0–16.9 MHz)`

RECORD TYPE	COMMAND/DATA FUNCTION

03	Miscellaneous Write Functions
	`:nnxxxx03ffssddcc`
	Where:
	`nn = number of bytes (hex) in record`
	`xxxx = required field, but value is a “don’t care”`
	`03 = Write Function`
	`ff = subfunction code`
	`ss = selection code`
	`dd = data input (as needed)`
	`cc = checksum`
	
	Subfunction Code = 01 (Erase Blocks)
	`ff = 01`
	`ss = block code as shown below:`
	`block 0, 0k to 8k, 00H`
	`block 1, 8k to 16k, 20H`
	`block 2, 16k to 32k, 40H`
	`block 3, 32k to 48k, 80H`
	`block 4, 48k to 64k, C0H`
	Example:
	`:0200000301C03C erase block 4`
	
	Subfunction Code = 04 (Erase Boot Vector and Status Byte)
	`ff = 04`
	`ss = don’t care`
	Example:
	`:020000030400F7 erase boot vector and status byte`
	
	Subfunction Code = 05 (Program Security Bits)
	`ff = 05`
	`ss = 00 program security bit 1 (inhibit writing to Flash)`
	`01 program security bit 2 (inhibit Flash verify)`
	`02 program security bit 3 (disable external memory)`
	Example:
	`:020000030501F5 program security bit 2`
	
	Subfunction Code = 06 (Program Status Byte or Boot Vector)
	`ff = 06`
	`ss = 00 program status byte`
	`01 program boot vector`
	Example:
	`:030000030601FCF7 program boot vector with 0FCH`
	
	Subfunction Code = 07 (Full Chip Erase)
	Erases all blocks, security bits, and sets status and boot vector to default values
	`ff = 07`
	`ss = don’t care`
	`dd = don’t care`
	Example:
	`:0100000307F5 full chip erase`

04	Display Device Data or Blank Check – Record type 04 causes the contents of the entire Flash array to be sent out the serial port in a formatted display. This display consists of an address and the contents of 16 bytes starting with that address. No display of the device contents will occur if security bit 2 has been programmed. Data to the serial port is initiated by the reception of any character and terminated by the reception of any character.
	
	General Format of Function 04
	`:05xxxx04sssseeeeffcc`
	Where:
	`05 = number of bytes (hex) in record`
	`xxxx = required field, but value is a “don’t care”`
	`04 = “Display Device Data or Blank Check” function code`
	`ssss = starting address`
	`eeee = ending address`
	`ff = subfunction`
	`00 = display data`
	`01 = blank check`
	`cc = checksum`
	Example:
	`:0500000440004FFF0069 display 4000–4FFF`

RECORD TYPE	COMMAND/DATA FUNCTION

05	Miscellaneous Read Functions
	
	General Format of Function 05
	`:02xxxx05ffsscc`
	Where:
	`02 = number of bytes (hex) in record`
	`xxxx = required field, but value is a “don’t care”`
	`05 = “Miscellaneous Read” function code`
	`ffss = subfunction and selection code`
	`0000 = read signature byte – manufacturer id (15H)`
	`0001 = read signature byte – device id # 1 (C2H)`
	`0002 = read signature byte – device id # 2`
	
	`0700 = read security bits`
	`0701 = read status byte`
	`0702 = read boot vector`
	`cc = checksum`
	Example:
	`:020000050001F8 read signature byte – device id # 1`

06	Direct Load of Baud Rate
	
	General Format of Function 06
	`:02xxxx06hhllcc`
	Where:
	`02 = number of bytes (hex) in record`
	`xxxx = required field, but value is a “don’t care”`
	`06 = ”Direct Load of Baud Rate” function code`
	`hh = high byte of Timer 2`
	`ll = low byte of Timer 2`
	`cc = checksum`
	Example:
	`:02000006F500F3`

In Application Programming Method

Several In Application Programming (IAP) calls are available for use by an application program to permit selective erasing and programming of Flash sectors. 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 FFF0H. The oscillator frequency is an integer number rounded down to the nearest megahertz. For example, set R0 to 11 for 11.0592 MHz. Results are returned in the registers. The IAP calls are shown in Table 15.

Using the Watchdog Timer (WDT)

The 89C66x devices support the use of the WDT in IAP. The user specifies that the WDT is to be fed by setting the most significant bit of the function parameter passed in R1 prior to calling PGM_MTP. The WDT function is only supported for Block Erase when using the Quick Block Erase. The Quick Block Erase is specified by performing a Block Erase with register R0 = 0. Requesting a WDT feed during IAP should only be performed in applications that use the WDT since the process of feeding the WDT will start the WDT if the WDT was not working.

Table 15. IAP calls

IAP CALL	PARAMETER

PROGRAM DATA BYTE	Input Parameters:
	R0 = osc freq (integer)
	R1 = 02h
	R1 = 82h (WDT feed, Rx2 & 66x only)
	DPTR = address of byte to program
	ACC = byte to program
	Return Parameter
	ACC = 00 if pass, !00 if fail
	Sample routine:
	;***** Program Device Data (DData) *****
	;***** ACC holds data to write
	;***** DPTR holds address of byte to write *****
	;***** Returns with ACC = 00h if successful, else ACC NEQ 00h
	WRData:
	MOV AUXR1,#20H ;set the ENBOOT bit
	MOV R0, #11 ;FOSC
	MOV R1,#02H ;program data function
	MOV A,Mydata ;data to write
	MOV DPTR,Address ;specify address of byte to read
	CALL PGM_MTP ;execute the function
	RET

ERASE BLOCK	Input Parameters:
	R0 = osc freq (integer)
	R0 = 0 (QUICK ERASE, Rx2 & 66x only)
	R1 = 01h
	R1 = 81h (WDT feed, Rx2 & 66x only; can only be used with Quick Erase)
	DPH = block code as shown below:
	block 0, 0k to 8k, 00H
	block 1, 8k to 16k, 20H
	block 2, 16k to 32k, 40H
	block 3, 32k to 48k, 80H
	block 4, 48k to 64k, C0H
	
	DPL = 00h
	Return Parameter
	none
	Sample routine:
	;***** Erase Code Memory Block *****
	;***** DPH (7:5) indicates which of the 5 blocks to erase
	;***** DPTR values for the blocks are:
	; 0000h = block 0
	; 2000h = block 1
	; 4000h = block 2
	; 8000h = block 3
	; C000h = block 4
	
	ERSBLK:
	MOV AUXR1,#20H ;set the ENBOOT bit
	MOV R0, #11 ;FOSC
	MOV R1,#01H ;erase block
	MOV DPTR,#BLk_NUM ;specify which block
	CALL PGM_MTP ;execute the function
	RET

IAP CALL	PARAMETER

ERASE BOOT VECTOR & STATUS BYTE	Input Parameters:
	R0 = osc freq (integer)
	R1 = 04h
	R1 = 84h (WDT feed, Rx2 & 66x only)
	DPH = 00h
	DPL = don't care
	Return Parameter
	none
	Sample routine:
	;***** Erase Boot Vector (BV) & Status Byte (SB) *****
	;***** Note: This command erases BOTH the SB & BV
	
	ERSBBV;
	MOV AUXR1,#20H ;set the ENBOOT bit
	MOV R0, #11 ;FOSC
	MOV R1,#04H ;erase status byte & boot vector
	MOV DPH,#00h ;we don't care about DPL
	CALL PGM_MTP ;execute the function
	RET

PROGRAM SECURITY BIT	Input Parameters:
	R0 = osc freq (integer)
	R1 = 05h
	R1 = 85h (WDT feed, Rx2 & 66x only)
	DPH = 00h
	DPL = 00h – security bit # 1 (inhibit writing to Flash)
	01h – security bit # 2 (inhibit Flash verify)
	02h – security bit # 3 (disable external memory)
	Return Parameter
	none
	Sample routines:
	;***** Program Security Bit1 *****
	;***** DPTR indicates security bit to program *****
	
	WRSB1:
	MOV AUXR1,#20H ;set the ENBOOT bit
	MOV R0,#11 ;FOSC
	MOV R1,#05H ;program security bit function
	MOV DPTR,#0000h ;specify security bit 1
	CALL PGM_MTP ;execute the function
	RET
	;***** Program Security Bit2 *****
	;***** DPTR indicates security bit to program *****
	
	WRSB2:
	MOV AUXR1,#20H ;set the ENBOOT bit
	MOV R0,#11 ;FOSC
	MOV R1,#05H ;program security bit function
	MOV DPTR,#0001h ;specify security bit 2
	CALL PGM_MTP ;execute the function
	RET
	;***** Program Security Bit3 *****
	;***** DPTR indicates security bit to program *****
	
	WRSB3:
	MOV AUXR1,#20H ;set the ENBOOT bit
	MOV R0,#11 ;FOSC
	MOV R1,#05H ;program security bit function
	MOV DPTR,#0002h ;specify security bit 3
	CALL PGM_MTP ;execute the function
	RET

IAP CALL	PARAMETER

PROGRAM STATUS BYTE	Input Parameters:
	R0 = osc freq (integer)
	R1 = 06h
	R1 = 86h (WDT feed, Rx2, 66x only)
	DPH = 00h
	DPL = 00h – program status byte
	ACC = status byte
	Return Parameter
	ACC = 00 if pass; not 00 if fails
	Sample routine:
	;***** Program Status Byte (SB) *****
	;***** DPTR indicates program status byte *****
	;***** ACC holds new value of Status Byte to program *****
	
	WRSB:
	MOV AUXR1,#20H ;set the ENBOOT bit
	MOV R0,#11 ;FOSC
	MOV R1,#06H ;program status byte or boot vector
	MOV DPTR,#0000h ;specify status byte
	MOV A,NEW_SB ;
	CALL PGM_MTP ;execute the function
	RET

PROGRAM BOOT VECTOR	Input Parameters:
	R0 = osc freq (integer)
	R1 = 06h
	R1 = 86h (WDT feed, Rx2 & 66x only)
	DPH = 00h
	DPL = 01h – program boot vector
	ACC = boot vector
	Return Parameter
	ACC = 00 if pass; not 00 if fails
	Sample routine:
	;***** Program Boot Vector (BV) *****
	;***** DPTR indicates program boot vector *****
	;***** ACC holds new value of boot vector to program *****
	
	WRBV:
	MOV AUXR1,#20H ;set the ENBOOT bit
	MOV R0,#11 ;FOSC
	MOV R1,#06H ;program status byte or boot vector
	MOV DPTR,#0001h ;specify boot vector
	MOV A,NEW_SB ;new value for the boot vector
	CALL PGM_MTP ;execute the function
	RET

READ DEVICE DATA	Input Parameters:
	R1 = 03h
	R1 = 83h (WDT feed, Rx2 & 66x only)
	DPTR = address of byte to read
	Return Parameter
	ACC = value of byte read
	Sample routine:
	;*****reads the Device Data (DData) *****
	;***** DData returned in ACC *****
	;***** DPTR holds address of byte to read *****
	RDData:
	MOV AUXR1,#20H ;set the ENBOOT bit
	MOV R0,#11 ;FOSC
	MOV R1,#03H ;read data function
	MOV DPTR,Address ;specify address of byte to read
	CALL PGM_MTP ;execute the function
	RET

IAP CALL	PARAMETER

READ MANUFACTURER ID	Input Parameters:
	R0 = osc freq (integer)
	R1 = 00h
	R1 = 80h (WDT feed, Rx2 & 66x only)
	DPH = 00h
	DPL = 00h (manufacturer ID)
	Return Parameter
	ACC = value of byte read
	Sample routine:
	;*****reads the Manufacturer ID (MID) *****
	;***** MID returned in ACC (should be 15h for Philips)
	RDMID:
	MOV AUXR1,#20H ;set the ENBOOT bit
	MOV R0,#11 ;FOSC
	MOV R1,#00H ;read misc function
	MOV DPTR,#0000H ;specify MID
	CALL PGM_MTP ;execute the function
	RET

READ DEVICE ID # 1	Input Parameters:
	R0 = osc freq (integer)
	R1 = 00h
	R1 = 80h (WDT feed, Rx2 & 66x only)
	DPH = 00h
	DPL = 01h (device ID # 1)
	Return Parameter
	ACC = value of byte read
	Sample routine:
	;*****reads the Device ID 1 (DID1) *****
	;***** DID1 returned in ACC
	RDDID1:
	MOV AUXR1,#20H ;set the ENBOOT bit
	MOV R0,#11 ;FOSC
	MOV R1,#00H ;read misc function
	MOV DPTR,#0001H ;specify device id 1
	CALL PGM_MTP ;execute the function
	RET

READ DEVICE ID # 2	Input Parameters:
	R0 = osc freq (integer)
	R1 = 00h
	R1 = 80h (WDT feed, Rx2 & 66x only)
	DPH = 00h
	DPL = 02h (device ID # 2)
	Return Parameter
	ACC = value of byte read
	Sample routine:
	;*****reads the Device ID 2 (DID2) *****
	;***** DID2 returned in ACC
	RDDID2:
	MOV AUXR1,#20H ;set the ENBOOT bit
	MOV R0,#11 ;FOSC
	MOV R1,#00H ;read misc function
	MOV DPTR,#0002H ;specify device id 2
	CALL PGM_MTP ;execute the function
	RET

IAP CALL	PARAMETER

READ SECURITY BITS	Input Parameters:
	R0 = osc freq (integer)
	R1 = 07h
	R1 = 87h (WDT feed, Rx2 & 66x only)
	DPH = 00h
	DPL = 00h (security bits)
	Return Parameter
	ACC = value of byte read
	Sample routine:
	;*****reads the Security Bits (SBits) *****
	;***** SBits returned in ACC (2:0)
	RDSBits:
	MOV AUXR1,#20H ;set the ENBOOT bit
	MOV R0,#11 ;FOSC
	MOV R1,#07H ;read misc function
	MOV DPTR,#0000H ;specify security bits
	CALL PGM_MTP ;execute the function
	RET

READ STATUS BYTE	Input Parameters:
	R0 = osc freq (integer)
	R1 = 07h
	R1 = 87h (WDT feed, Rx2 & 66x only)
	DPH = 00h
	DPL = 01h (status byte)
	Return Parameter
	ACC = value of byte read
	Sample routine:
	;*****reads the Status Byte (SB) *****
	;***** SB returned in ACC
	RDSB:
	MOV AUXR1,#20H ;set the ENBOOT bit
	MOV R0,#11 ;FOSC
	MOV R1,#07H ;read misc function
	MOV DPTR,#0001H ;specify status byte
	CALL PGM_MTP ;execute the function
	RET

READ BOOT VECTOR	Input Parameters:
	R0 = osc freq (integer)
	R1 = 07h
	R1 = 87h (WDT feed, Rx2 & 66x only)
	DPH = 00h
	DPL = 02h (boot vector)
	Return Parameter
	ACC = value of byte read
	Sample routine:
	;*****reads the Boot Vector (BV) *****
	;***** BV returned in ACC
	RDBV:
	MOV AUXR1,#20H ;set the ENBOOT bit
	MOV R0,#11 ;FOSC
	MOV R1,#07H ;read misc function
	MOV DPTR,#0002H ;specify boot vector
	CALL PGM_MTP ;execute the function
	RET

Security

The security feature protects against software piracy and prevents the contents of the Flash from being read. The Security Lock bits are located in Flash. The P89C660/662/664/668 has 3 programmable security lock bits that will provide different levels of protection for the on-chip code and data (see Table 16).

Table 16.

SECURITY LOCK BITS[1]				PROTECTION DESCRIPTION
Level	LB1	LB2	LB3	
1	0	0	0	MOVC instructions executed from external program memory are disabled from fetching code bytes from internal memory.
2	1	0	0	Same as level 1, plus block erase is disabled. Erase or programming of the status byte or boot vector is disabled.
3	1	1	0	Same as level 2, plus verify of code memory is disabled.
4	1	1	1	Same as level 3, plus external execution is disabled.

NOTE:
1. Security bits are independent of each other. Full-chip erase may be performed regardless of the state of the security bits.
2. Any other combination of lockbits is undefined.
3. Setting LBx doesn't prevent programming of unprogrammed bits.

ABSOLUTE MAXIMUM RATINGS[1, 2, 3]

PARAMETER	RATING	UNIT
Operating temperature under bias	0 to +70 or –40 to +85	°C
Storage temperature range	–65 to +150	°C
Voltage on $\overline{EA}/V_{PP}$ pin to V_{SS}	0 to +13.0	V
Voltage on any other pin to V_{SS}	–0.5 to +6.5	V
Maximum I_{OL} per I/O pin	15	mA
Power dissipation (based on package heat transfer limitations, not device power consumption)	1.5	W

NOTES:
1. Stresses above those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any conditions other than those described in the AC and DC Electrical Characteristics section of this specification is not implied.
2. This product includes circuitry specifically designed for the protection of its internal devices from the damaging effects of excessive static charge. Nonetheless, it is suggested that conventional precautions be taken to avoid applying greater than the rated maximum.
3. Parameters are valid over operating temperature range unless otherwise specified. All voltages are with respect to V_{SS} unless otherwise noted.

DC ELECTRICAL CHARACTERISTICS

T_{amb} = 0 °C to +70 °C, 5 V ± 10% or –40 °C to +85 °C; 5V ±5%; V_{SS} = 0 V

SYMBOL	PARAMETER	TEST CONDITIONS	LIMITS			UNIT
			MIN	TYP[1]	MAX	
V_{IL}	Input low voltage	4.5 V < V_{CC} < 5.5 V	–0.5		0.2 V_{CC}–0.1	V
V_{IL2}	Input low voltage to P1.6/SCL, P1.7/SDA[11]		–0.5		0.3V_{DD}	V
V_{IH}	Input high voltage (ports 0, 1, 2, 3, $\overline{EA}$)		0.2V_{CC}+0.9		V_{CC}+0.5	V
V_{IH1}	Input high voltage, XTAL1, RST		0.7V_{CC}		V_{CC}+0.5	V
V_{IH2}	Input high voltage, P1.6/SCL, P1.7/SDA[11]		0.7V_{DD}		6.0	V
V_{OL}	Output low voltage, ports 1, 2, 3[8]	V_{CC} = 4.5 V I_{OL} = 1.6 mA[2]	–		0.4	V
V_{OL1}	Output low voltage, port 0, ALE, $\overline{PSEN}$ [7, 8]	V_{CC} = 4.5 V I_{OL} = 3.2 mA[2]	–		0.45	V
V_{OL2}	Output low voltage, P1.6/SCL, P1.7/SDA	I_{OL} = 3.0 mA	–		0.4	V
V_{OH}	Output high voltage, ports 1, 2, 3 [3]	V_{CC} = 4.5 V I_{OH} = –30 µA	V_{CC} – 0.7		–	V
V_{OH1}	Output high voltage (port 0 in external bus mode), ALE[9], $\overline{PSEN}$[3]	V_{CC} = 4.5 V I_{OH} = –3.2 mA	V_{CC} – 0.7		–	V
I_{IL}	Logical 0 input current, ports 1, 2, 3	V_{IN} = 0.4 V	–1		–75	µA
I_{TL}	Logical 1-to-0 transition current, ports 1, 2, 3[6]	V_{IN} = 2.0 V See Note 4	–		–650	µA
I_{LI}	Input leakage current, port 0	0.45 < V_{IN} < V_{CC} – 0.3	–		±10	µA
I_{L2}	Input leakage current, P1.6/SCL, P1.7/SDA	0V < VI < 6 V 0V < V_{DD} < 5.5 V	–		10	µA
I_{CC}	Power supply current (see Figure 64): Active mode (see Note 5) Idle mode (see Note 5) Power-Down mode or clock stopped (see Figure 71 for conditions) Programming and erase mode	See Note 5 T_{amb} = 0 °C to 70 °C T_{amb} = –40 °C to +85 °C f_{osc} = 20 MHz		 20 60	 100 125 	 µA µA mA
R_{RST}	Internal reset pull-down resistor		40		225	kΩ
C_{IO}	Pin capacitance[10] (except $\overline{EA}$)		–		15	pF

NOTES:

1. Typical ratings are not guaranteed. The values listed are at room temperature, 5 V.
2. Capacitive loading on ports 0 and 2 may cause spurious noise to be superimposed on the V_{OL}s of ALE and ports 1 and 3. The noise is due to external bus capacitance discharging into the port 0 and port 2 pins when these pins make 1-to-0 transitions during bus operations. In the worst cases (capacitive loading > 100 pF), the noise pulse on the ALE pin may exceed 0.8 V. In such cases, it may be desirable to qualify ALE with a Schmitt Trigger, or use an address latch with a Schmitt Trigger STROBE input. I_{OL} can exceed these conditions provided that no single output sinks more than 5mA and no more than two outputs exceed the test conditions.
3. Capacitive loading on ports 0 and 2 may cause the V_{OH} on ALE and $\overline{PSEN}$ to momentarily fall below the V_{CC}–0.7 specification when the address bits are stabilizing.
4. Pins of ports 1, 2 and 3 source a transition current when they are being externally driven from 1 to 0. The transition current reaches its maximum value when V_{IN} is approximately 2 V.
5. See Figures 68 through 71 for I_{CC} test conditions and Figure 64 for I_{CC} vs Freq.
   Active mode: $I_{CC(MAX)}$ = (2.8 × FREQ. + 8.0)mA for all devices, in 6 clock mode; (1.4 × FREQ. + 8.0)mA in 12 clock mode.
   Idle mode: $I_{CC(MAX)}$ = (1.2 × FREQ. +1.0)mA in 6 clock mode; (0.6 × FREQ. +1.0)mA in 12 clock mode.
6. This value applies to T_{amb} = 0 °C to +70 °C.
7. Load capacitance for port 0, ALE, and $\overline{PSEN}$ = 100 pF, load capacitance for all other outputs = 80 pF.
8. Under steady state (non-transient) conditions, I_{OL} must be externally limited as follows:
   Maximum I_{OL} per port pin: 15 mA (*NOTE: This is 85 °C specification.)
   Maximum I_{OL} per 8-bit port: 26 mA
   Maximum total I_{OL} for all outputs: 71 mA
   If I_{OL} exceeds the test condition, V_{OL} may exceed the related specification. Pins are not guaranteed to sink current greater than the listed test conditions.
9. ALE is tested to V_{OH1}, except when ALE is off then V_{OH} is the voltage specification.
10. Pin capacitance is characterized but not tested. Pin capacitance is less than 25 pF. Pin capacitance of ceramic package is less than 15 pF (except $\overline{EA}$ is 25 pF).
11. The input threshold voltage of P1.6 and P1.7 (SIO1) meets the I^2C specification, so an input voltage below 1.5 V will be recognized as a logic 0 while an input voltage above 3.0 V will be recognized as a logic 1.

AC ELECTRICAL CHARACTERISTICS (6 CLOCK MODE)

T_{amb} = 0 °C to +70 °C, V_{CC} = 5 V ± 10% or –40 °C to +85 °C, V_{CC} = 5 V ±5%, V_{SS} = 0 V[1, 2, 3]

SYMBOL	FIGURE	PARAMETER	VARIABLE CLOCK[4]		20 MHz CLOCK[4]		UNIT
			MIN	MAX	MIN	MAX	
$1/t_{CLCL}$	57	Oscillator frequency	0	20	–	–	MHz
t_{LHLL}	57	ALE pulse width	t_{CLCL}–40	–	10	–	ns
t_{AVLL}	57	Address valid to ALE low	$0.5t_{CLCL}$–20	–	5	–	ns
t_{LLAX}	57	Address hold after ALE low	$0.5t_{CLCL}$–20	–	5	–	ns
t_{LLIV}	57	ALE low to valid instruction in	–	$2t_{CLCL}$–65	–	35	ns
t_{LLPL}	57	ALE low to $\overline{PSEN}$ low	$0.5t_{CLCL}$–20	–	5	–	ns
t_{PLPH}	57	$\overline{PSEN}$ pulse width	$1.5t_{CLCL}$–45	–	30	–	ns
t_{PLIV}	57	$\overline{PSEN}$ low to valid instruction in	–	$1.5t_{CLCL}$–60	–	15	ns
t_{PXIX}	57	Input instruction hold after $\overline{PSEN}$	0	–	0	–	ns
t_{PXIZ}	57	Input instruction float after $\overline{PSEN}$	–	$0.5t_{CLCL}$–20	–	5	ns
t_{AVIV}	57	Address to valid instruction in	–	$2.5t_{CLCL}$–80	–	45	ns
t_{PLAZ}	57	$\overline{PSEN}$ low to address float	–	10	–	10	ns
Data Memory							
t_{RLRH}	58, 59	$\overline{RD}$ pulse width	$3t_{CLCL}$–100	–	50	–	ns
t_{WLWH}	58, 59	$\overline{WR}$ pulse width	$3t_{CLCL}$–100	–	50	–	ns
t_{RLDV}	58, 59	$\overline{RD}$ low to valid data in	–	$2.5t_{CLCL}$–90	–	35	ns
t_{RHDX}	58, 59	Data hold after $\overline{RD}$	0	–	0	–	ns
t_{RHDZ}	58, 59	Data float after $\overline{RD}$	–	t_{CLCL}–20	–	5	ns
t_{LLDV}	58, 59	ALE low to valid data in	–	$4t_{CLCL}$–150	–	50	ns
t_{AVDV}	58, 59	Address to valid data in	–	$4.5t_{CLCL}$–165	–	60	ns
t_{LLWL}	58, 59	ALE low to $\overline{RD}$ or $\overline{WR}$ low	$1.5t_{CLCL}$–50	$1.5t_{CLCL}$+50	25	125	ns
t_{AVWL}	58, 59	Address valid to $\overline{WR}$ low or $\overline{RD}$ low	$2t_{CLCL}$–75	–	25	–	ns
t_{QVWX}	58, 59	Data valid to $\overline{WR}$ transition	$0.5t_{CLCL}$–25	–	0	–	ns
t_{WHQX}	58, 59	Data hold after $\overline{WR}$	$0.5t_{CLCL}$–20	–	5	–	ns
t_{QVWH}	59	Data valid to $\overline{WR}$ high	$3.5t_{CLCL}$–130	–	45	–	ns
t_{RLAZ}	58, 59	$\overline{RD}$ low to address float	–	0	–	0	ns
t_{WHLH}	58, 59	$\overline{RD}$ or $\overline{WR}$ high to ALE high	$0.5t_{CLCL}$–20	$0.5t_{CLCL}$+20	5	45	ns
External Clock							
t_{CHCX}	61	High time	20	t_{CLCL}–t_{CLCX}	–	–	ns
t_{CLCX}	61	Low time	20	t_{CLCL}–t_{CHCX}	–	–	ns
t_{CLCH}	61	Rise time	–	5	–	–	ns
t_{CHCL}	61	Fall time	–	5	–	–	ns
Shift Register							
t_{XLXL}	60	Serial port clock cycle time	$6t_{CLCL}$	–	300	–	ns
t_{QVXH}	60	Output data setup to clock rising edge	$5t_{CLCL}$–133	–	117	–	ns
t_{XHQX}	60	Output data hold after clock rising edge	t_{CLCL}–30	–	20	–	ns
t_{XHDX}	60	Input data hold after clock rising edge	0	–	0	–	ns
t_{XHDV}	60	Clock rising edge to input data valid	–	$5t_{CLCL}$–133	–	117	ns

NOTES:
1. Parameters are valid over operating temperature range unless otherwise specified.
2. Load capacitance for port 0, ALE, and $\overline{PSEN}$ = 100 pF, load capacitance for all other outputs = 80 pF.
3. Interfacing the microcontroller to devices with float times up to 45ns is permitted. This limited bus contention will not cause damage to Port 0 drivers.
4. Parts are tested to 2 MHz, but are guaranteed to operate down to 0 Hz.

AC ELECTRICAL CHARACTERISTICS (6 CLOCK MODE) (Continued)

T_{amb} = 0 °C to +70 °C, V_{CC} = 5 V ± 10% or –40 °C to +85 °C, V_{CC} = 5 V ± 5%, V_{SS} = 0 V[1, 2]

SYMBOL	PARAMETER	INPUT	OUTPUT
I^2C Interface			
$t_{HD;STA}$	START condition hold time	≥ 7 t_{CLCL}	> 4.0 µs [4]
t_{LOW}	SCL low time	≥ 8 t_{CLCL}	> 4.7 µs [46]
t_{HIGH}	SCL high time	≥ 7 t_{CLCL}	> 4.0 µs [4]
t_{RC}	SCL rise time	≤ 1 µs	– [5]
t_{FC}	SCL fall time	≤ 0.3 µs	< 0.3 µs [6]
$t_{SU;DAT1}$	Data set-up time	≥ 250 ns	> 10 t_{CLCL} – t_{RD}
$t_{SU;DAT2}$	SDA set-up time (before rep. START cond.)	≥ 250 ns	> 1 µs [4]
$t_{SU;DAT3}$	SDA set-up time (before STOP cond.)	≥ 250 ns	> 4 t_{CLCL}
$t_{HD;DAT}$	Data hold time	≥ 0 ns	> 4 t_{CLCL} – t_{FC}
$t_{SU;STA}$	Repeated START set-up time	≥ 7 t_{CLCL} [4]	> 4.7 µs [4]
$t_{SU;STO}$	STOP condition set-up time	≥ 7 t_{CLCL} [4]	> 4.0 µs [4]
t_{BUF}	Bus free time	≥ 7 t_{CLCL} [4]	> 4.7 µs [4]
t_{RD}	SDA rise time	≤ 1 µs[7]	– [5]
t_{FD}	SDA fall time	≤ 300 ns[7]	< 0.3 µs [6]

NOTES:
1. Parameters are valid over operating temperature range and voltage range unless otherwise specified.
2. Load capacitance for port 0, ALE, and $\overline{PSEN}$ = 100 pF, load capacitance for all other outputs = 80 pF.
3. These values are characterized but not 100% production tested.
4. At 100 kbit/s. At other bit rates this value is inversely proportional to the bit-rate of 100 kbit/s.
5. Determined by the external bus-line capacitance and the external bus-line pull-resistor, this must be < 1 µs.
6. Spikes on the SDA and SCL lines with a duration of less than 3 t_{CLCL} will be filtered out. Maximum capacitance on bus-lines SDA and SCL = 400 pF.
7. $t_{CLCL} = 1/f_{OSC}$ = one oscillator clock period at pin XTAL1.

AC ELECTRICAL CHARACTERISTICS (12 CLOCK MODE)

T_{amb} = 0 °C to +70 °C, V_{CC} = 5 V ± 10%, or –40 °C to +85 °C, V_{CC} = 5 V ±5%, V_{SS} = 0V[1, 2, 3]

SYMBOL	FIGURE	PARAMETER	VARIABLE CLOCK[4]		33 MHz CLOCK[4]		UNIT
			MIN	MAX	MIN	MAX	
$1/t_{CLCL}$	57	Oscillator frequency	0	33	–	–	MHz
t_{LHLL}	57	ALE pulse width	$2t_{CLCL}$–40	–	21	–	ns
t_{AVLL}	57	Address valid to ALE low	t_{CLCL}–25	–	5	–	ns
t_{LLAX}	57	Address hold after ALE low	t_{CLCL}–25	–	5	–	ns
t_{LLIV}	57	ALE low to valid instruction in	–	$4t_{CLCL}$–65	–	55	ns
t_{LLPL}	57	ALE low to $\overline{PSEN}$ low	t_{CLCL}–25	–	5	–	ns
t_{PLPH}	57	$\overline{PSEN}$ pulse width	$3t_{CLCL}$–45	–	45	–	ns
t_{PLIV}	57	$\overline{PSEN}$ low to valid instruction in	–	$3t_{CLCL}$–60	–	30	ns
t_{PXIX}	57	Input instruction hold after $\overline{PSEN}$	0	–	0	–	ns
t_{PXIZ}	57	Input instruction float after $\overline{PSEN}$	–	t_{CLCL}–25	–	5	ns
t_{AVIV}	57	Address to valid instruction in	–	$5t_{CLCL}$–80	–	70	ns
t_{PLAZ}	57	$\overline{PSEN}$ low to address float	–	10	–	10	ns
Data Memory							
t_{RLRH}	58, 59	$\overline{RD}$ pulse width	$6t_{CLCL}$–100	–	82	–	ns
t_{WLWH}	58, 59	$\overline{WR}$ pulse width	$6t_{CLCL}$–100	–	82	–	ns
t_{RLDV}	58, 59	$\overline{RD}$ low to valid data in	–	$5t_{CLCL}$–90	–	60	ns
t_{RHDX}	58, 59	Data hold after $\overline{RD}$	0	–	0	–	ns
t_{RHDZ}	58, 59	Data float after $\overline{RD}$	–	$2t_{CLCL}$–28	–	32	ns
t_{LLDV}	58, 59	ALE low to valid data in	–	$8t_{CLCL}$–150	–	90	ns
t_{AVDV}	58, 59	Address to valid data in	–	$9t_{CLCL}$–165	–	105	ns
t_{LLWL}	58, 59	ALE low to $\overline{RD}$ or $\overline{WR}$ low	$3t_{CLCL}$–50	$3t_{CLCL}$+50	40	140	ns
t_{AVWL}	58, 59	Address valid to $\overline{WR}$ low or $\overline{RD}$ low	$4t_{CLCL}$–75	–	45	–	ns
t_{QVWX}	58, 59	Data valid to $\overline{WR}$ transition	t_{CLCL}–30	–	0	–	ns
t_{WHQX}	58, 59	Data hold after $\overline{WR}$	t_{CLCL}–25	–	5	–	ns
t_{QVWH}	59	Data valid to $\overline{WR}$ high	$7t_{CLCL}$–130	–	80	–	ns
t_{RLAZ}	58, 59	$\overline{RD}$ low to address float	–	0	–	0	ns
t_{WHLH}	58, 59	$\overline{RD}$ or $\overline{WR}$ high to ALE high	t_{CLCL}–25	t_{CLCL}+25	5	55	ns
External Clock							
t_{CHCX}	61	High time	17	t_{CLCL}–t_{CLCX}	–	–	ns
t_{CLCX}	61	Low time	17	t_{CLCL}–t_{CHCX}	–	–	ns
t_{CLCH}	61	Rise time	–	5	–	–	ns
t_{CHCL}	61	Fall time	–	5	–	–	ns
Shift Register							
t_{XLXL}	60	Serial port clock cycle time	$12t_{CLCL}$	–	360	–	ns
t_{QVXH}	60	Output data setup to clock rising edge	$10t_{CLCL}$–133	–	167	–	ns
t_{XHQX}	60	Output data hold after clock rising edge	$2t_{CLCL}$–80	–	50	–	ns
t_{XHDX}	60	Input data hold after clock rising edge	0	–	0	–	ns
t_{XHDV}	60	Clock rising edge to input data valid	–	$10t_{CLCL}$–133	–	167	ns

NOTES:
1. Parameters are valid over operating temperature range unless otherwise specified.
2. Load capacitance for port 0, ALE, and $\overline{PSEN}$ = 100 pF, load capacitance for all other outputs = 80 pF.
3. Interfacing the microcontroller to devices with float times up to 45 ns is permitted. This limited bus contention will not cause damage to Port 0 drivers.
4. Parts are tested to 3.5 MHz, but guaranteed to operate down to 0 Hz.

AC ELECTRICAL CHARACTERISTICS (12 CLOCK MODE) (Continued)

T_{amb} = 0°C to +70°C, V_{CC} = 5 V ± 10%, or –40°C to +85°C, V_{CC} = 5 V ± 5%, V_{SS} = 0 V[1, 2]

SYMBOL	PARAMETER	INPUT	OUTPUT
I^2C Interface			
$t_{HD;STA}$	START condition hold time	≥ 14 t_{CLCL}	> 4.0 µs [4]
t_{LOW}	SCL low time	≥ 16 t_{CLCL}	> 4.7 µs [4]
t_{HIGH}	SCL high time	≥ 14 t_{CLCL}	> 4.0 µs [4]
t_{RC}	SCL rise time	≤ 1 µs	– [5]
t_{FC}	SCL fall time	≤ 0.3 µs	< 0.3 µs [6]
$t_{SU;DAT1}$	Data set-up time	≥ 250 ns	> 20 t_{CLCL} – t_{RD}
$t_{SU;DAT2}$	SDA set-up time (before rep. START cond.)	≥ 250 ns	> 1 µs [4]
$t_{SU;DAT3}$	SDA set-up time (before STOP cond.)	≥ 250 ns	> 8 t_{CLCL}
$t_{HD;DAT}$	Data hold time	≥ 0 ns	> 8 t_{CLCL} – t_{FC}
$t_{SU;STA}$	Repeated START set-up time	≥ 14 t_{CLCL} [4]	> 4.7 µs [4]
$t_{SU;STO}$	STOP condition set-up time	≥ 14 t_{CLCL} [4]	> 4.0 µs [4]
t_{BUF}	Bus free time	≥ 14 t_{CLCL} [4]	> 4.7 µs [4]
t_{RD}	SDA rise time	≤ 1 µs[7]	– [5]
t_{FD}	SDA fall time	≤ 300 ns[7]	< 0.3 µs [6]

NOTES:
1. Parameters are valid over operating temperature range and voltage range unless otherwise specified.
2. Load capacitance for port 0, ALE, and $\overline{PSEN}$ = 100 pF, load capacitance for all other outputs = 80 pF.
3. These values are characterized but not 100% production tested.
4. At 100 kbit/s. At other bit rates this value is inversely proportional to the bit-rate of 100 kbit/s.
5. Determined by the external bus-line capacitance and the external bus-line pull-resistor, this must be < 1 µs.
6. Spikes on the SDA and SCL lines with a duration of less than 3 t_{CLCL} will be filtered out. Maximum capacitance on bus-lines SDA and SCL = 400 pF.
7. $t_{CLCL} = 1/f_{OSC}$ = one oscillator clock period at pin XTAL1. For 63 ns < t_{CLCL} < 285 ns (16 MHz > f_{OSC} > 3.5 MHz) the I^2C interface meets the I^2C-bus specification for bit-rates up to 100 kbit/s.

EXPLANATION OF THE AC SYMBOLS

Each timing symbol has five characters. The first character is always 't' (= time). The other characters, depending on their positions, indicate the name of a signal or the logical status of that signal. The designations are:

A – Address
C – Clock
D – Input data
H – Logic level high
I – Instruction (program memory contents)
L – Logic level low, or ALE
P – $\overline{\text{PSEN}}$
Q – Output data
R – $\overline{\text{RD}}$ signal
t – Time
V – Valid
W – $\overline{\text{WR}}$ signal
X – No longer a valid logic level
Z – Float

Examples: t_{AVLL} = Time for address valid to ALE low.
t_{LLPL} = Time for ALE low to $\overline{\text{PSEN}}$ low.

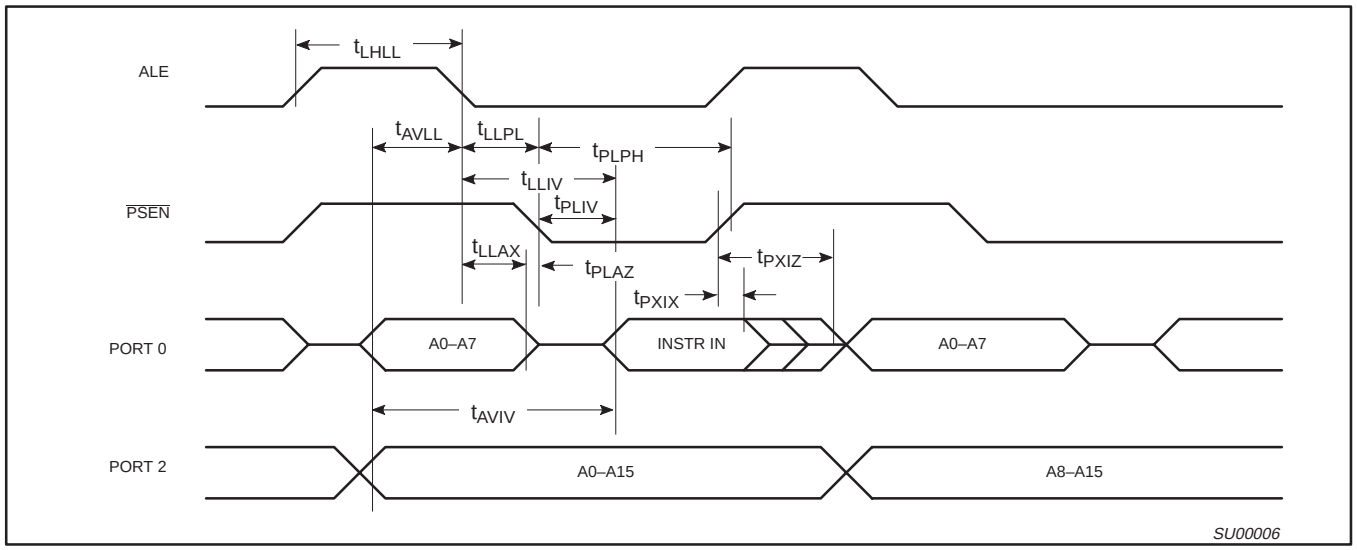


Figure 57. External Program Memory Read Cycle

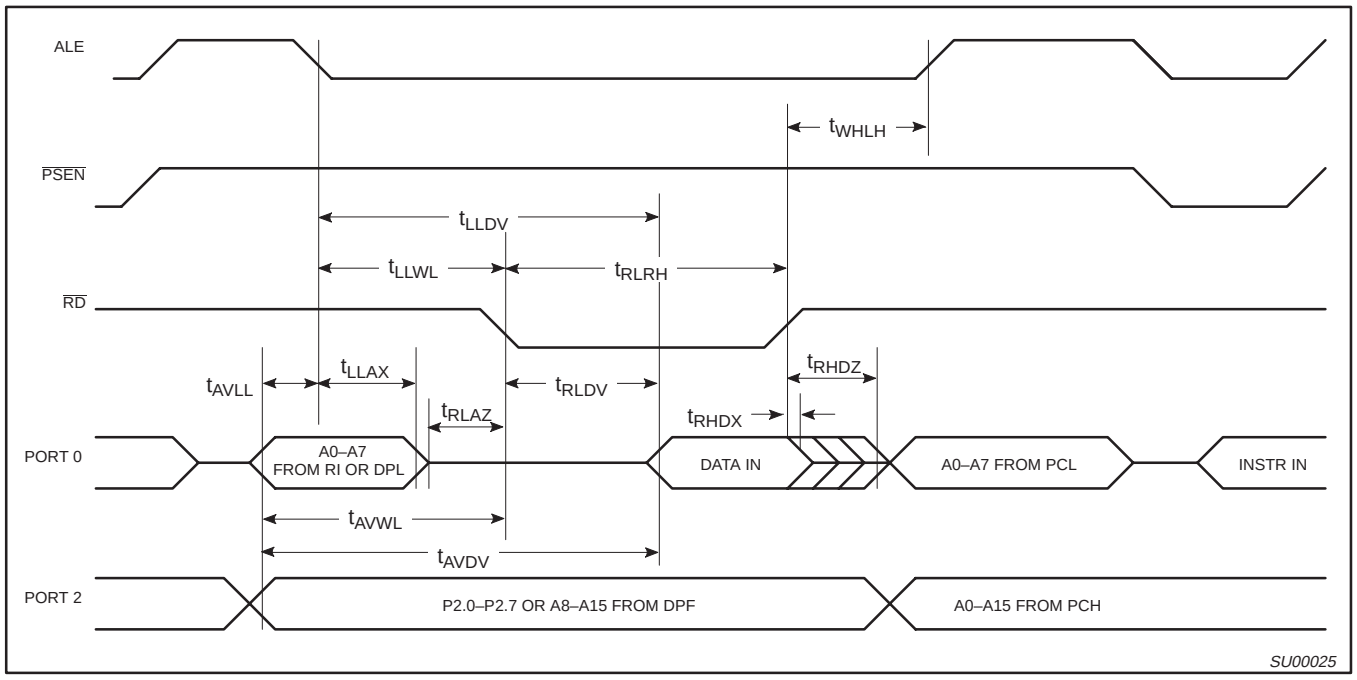


Figure 58. External Data Memory Read Cycle

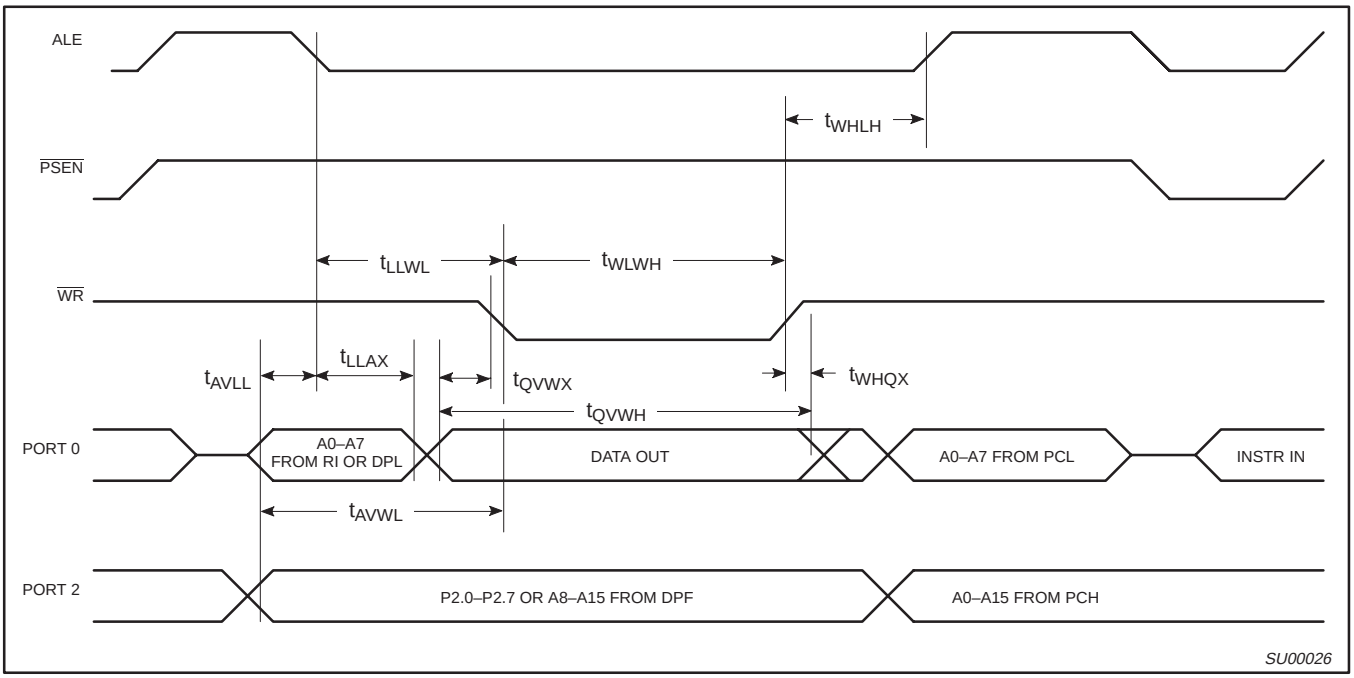


Figure 59. External Data Memory Write Cycle

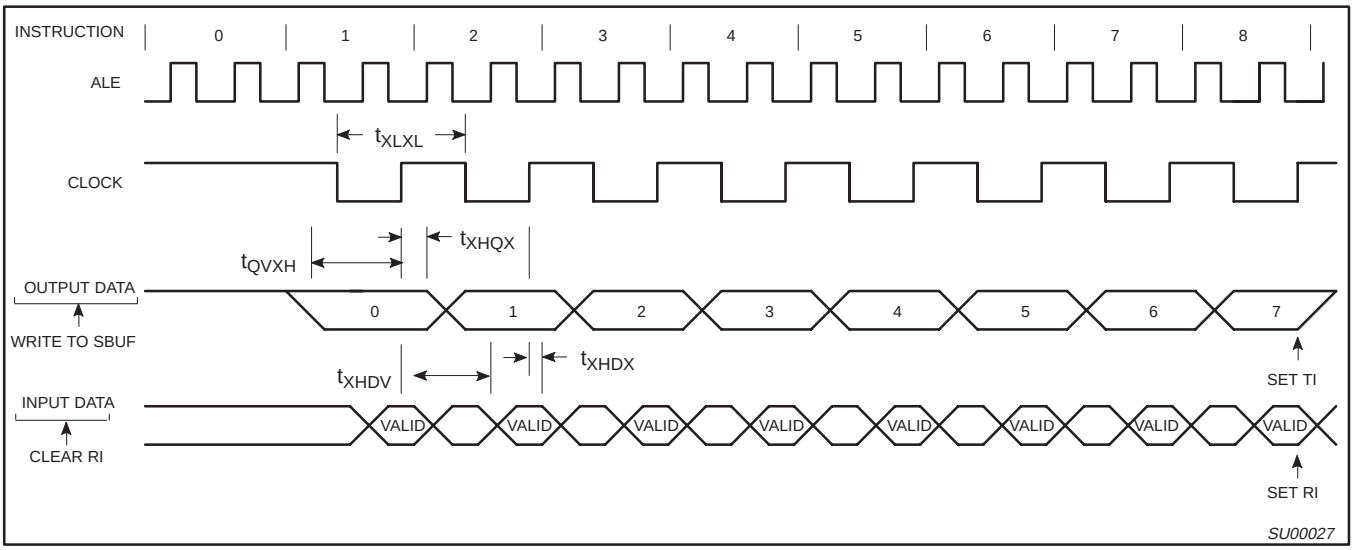


Figure 60. Shift Register Mode Timing

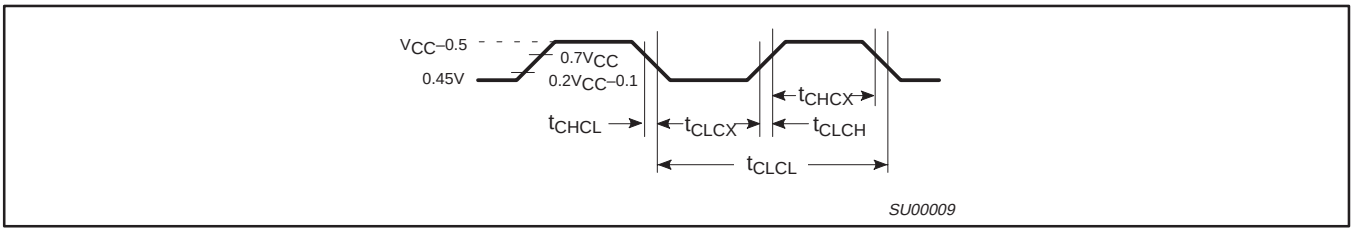


Figure 61. External Clock Drive

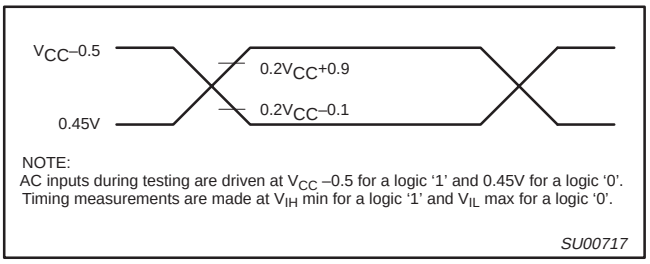


Figure 62. AC Testing Input/Output

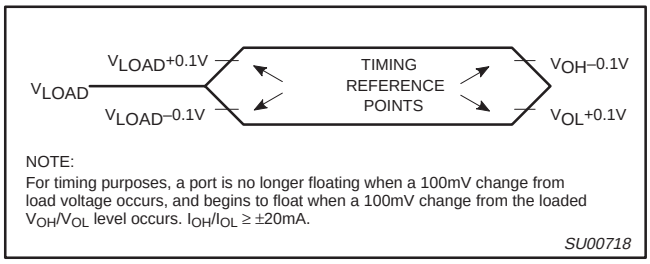


Figure 63. Float Waveform

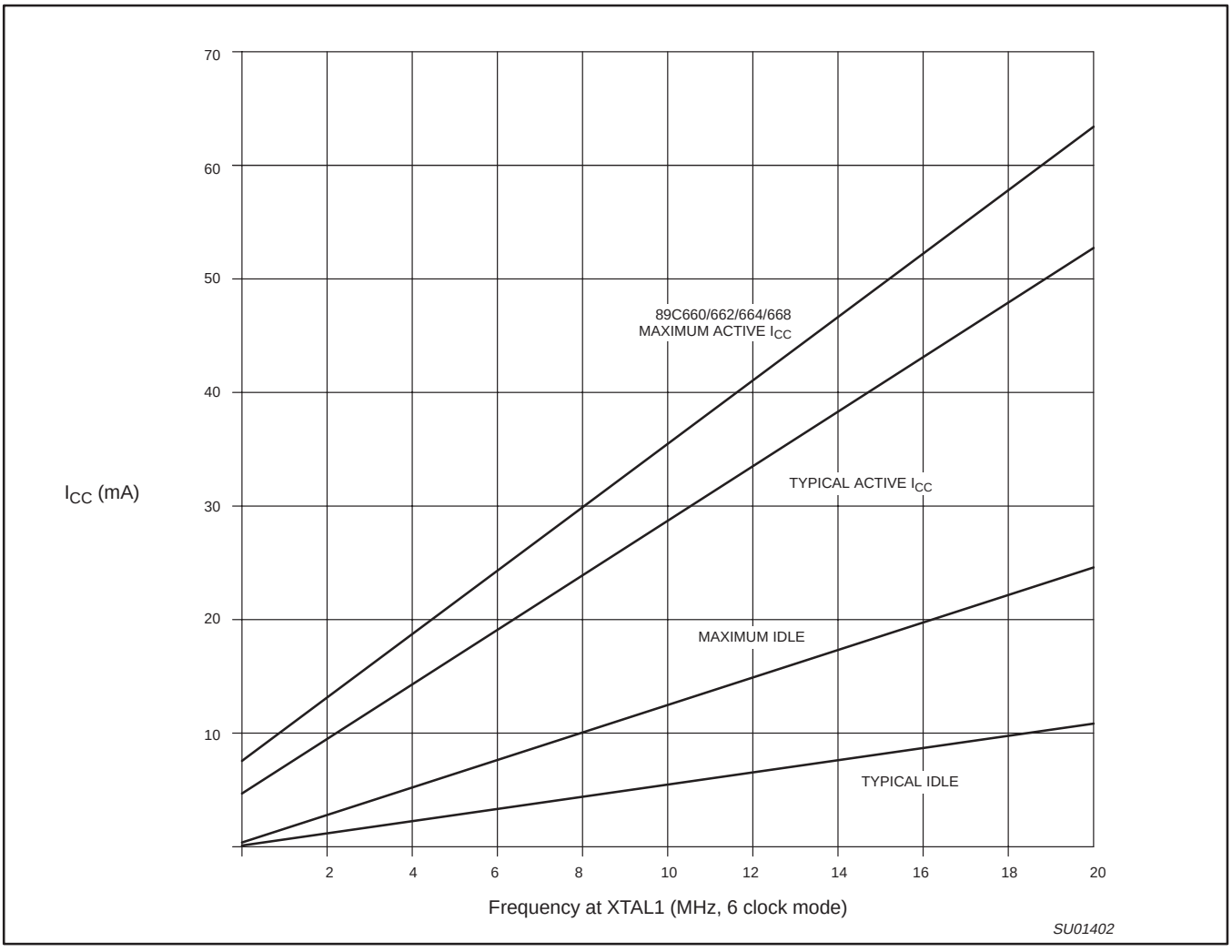


Figure 64. I_{CC} vs. FREQ
Valid only within frequency specifications of the device under test

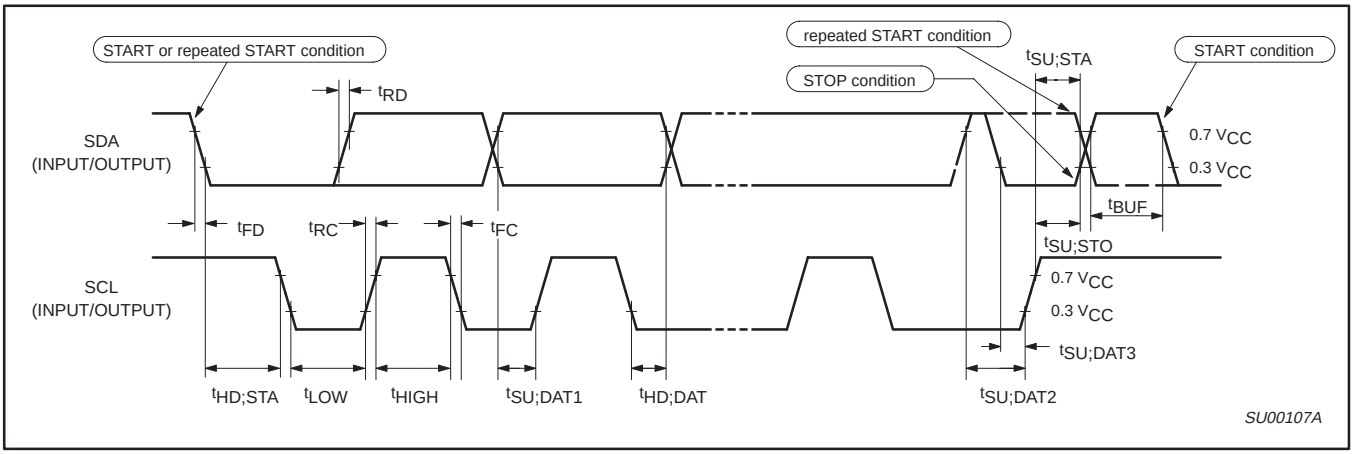


Figure 65. Timing SI01 (I^2C) Interface

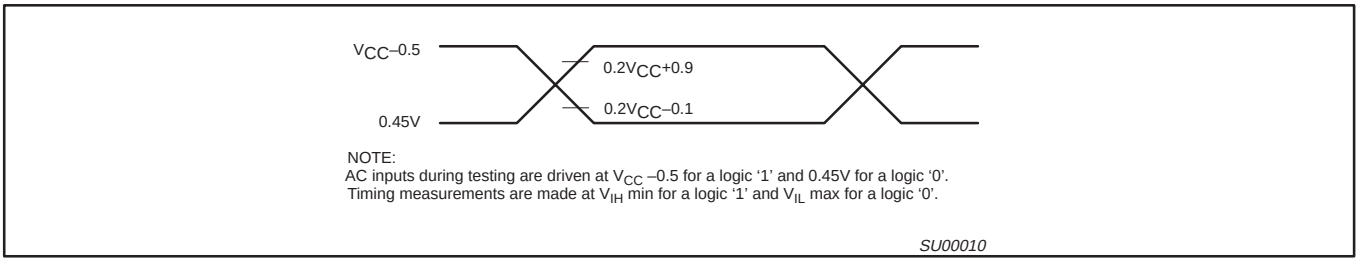


Figure 66. AC Testing Input/Output

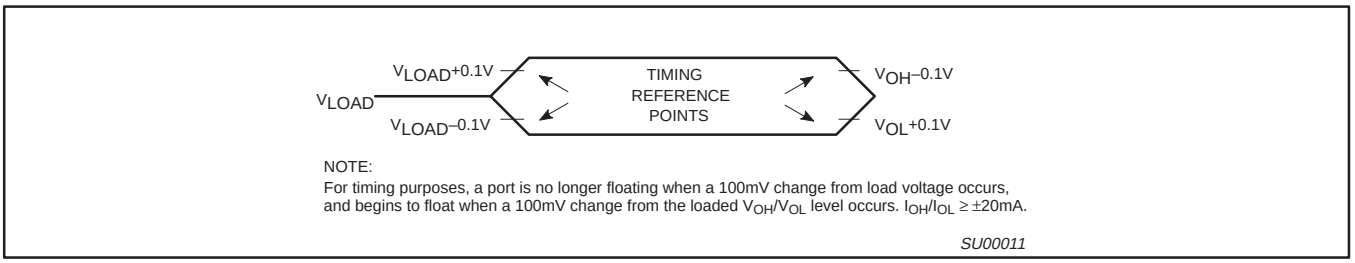


Figure 67. Float Waveform

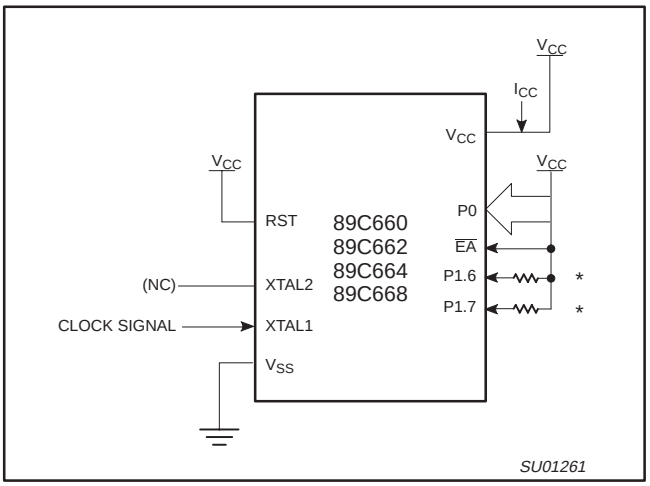


Figure 68. I_{CC} Test Condition, Active Mode. All other pins are disconnected

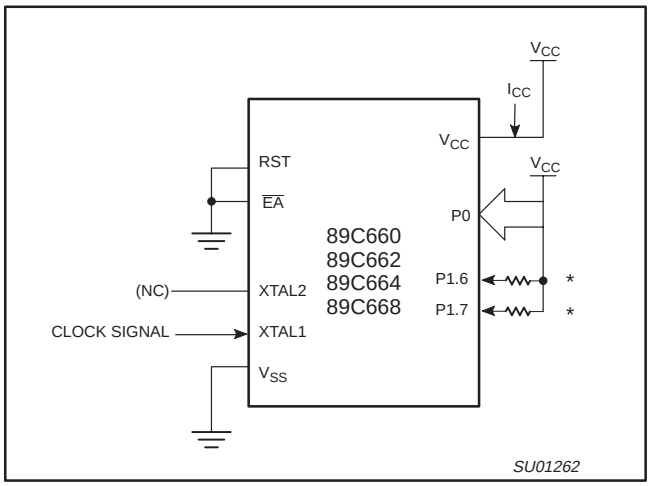


Figure 69. I_{CC} Test Condition, Idle Mode. All other pins are disconnected

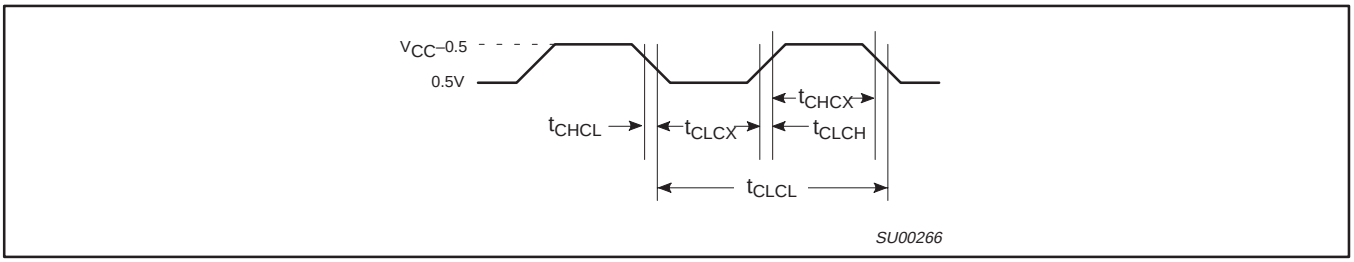


Figure 70. Clock Signal Waveform for I_{CC} Tests in Active and Idle Modes. $t_{CLCL} = t_{CHCL} = 10$ ns

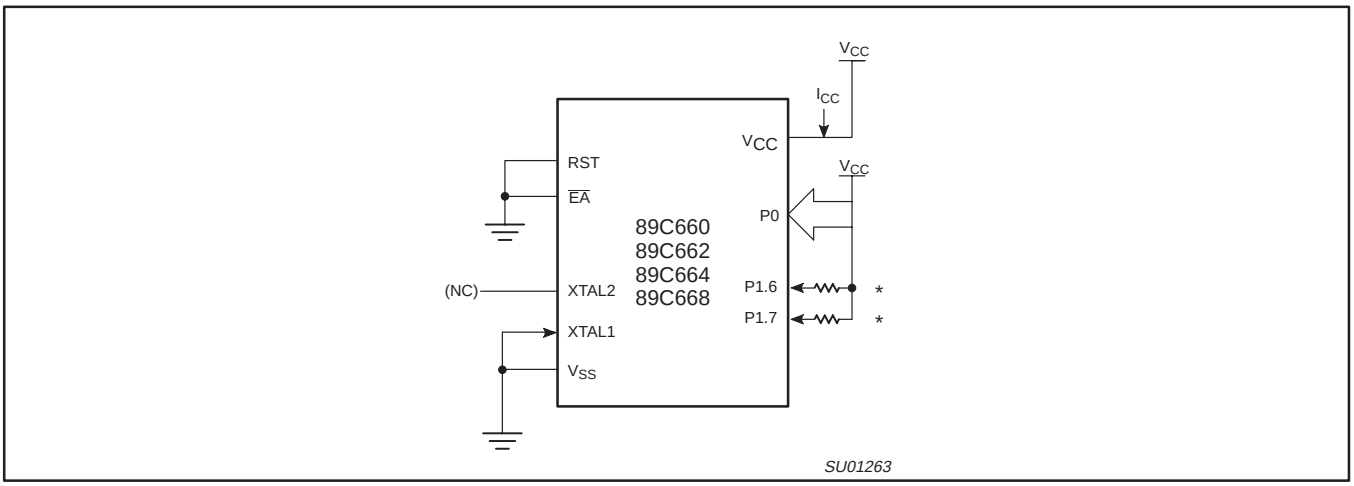
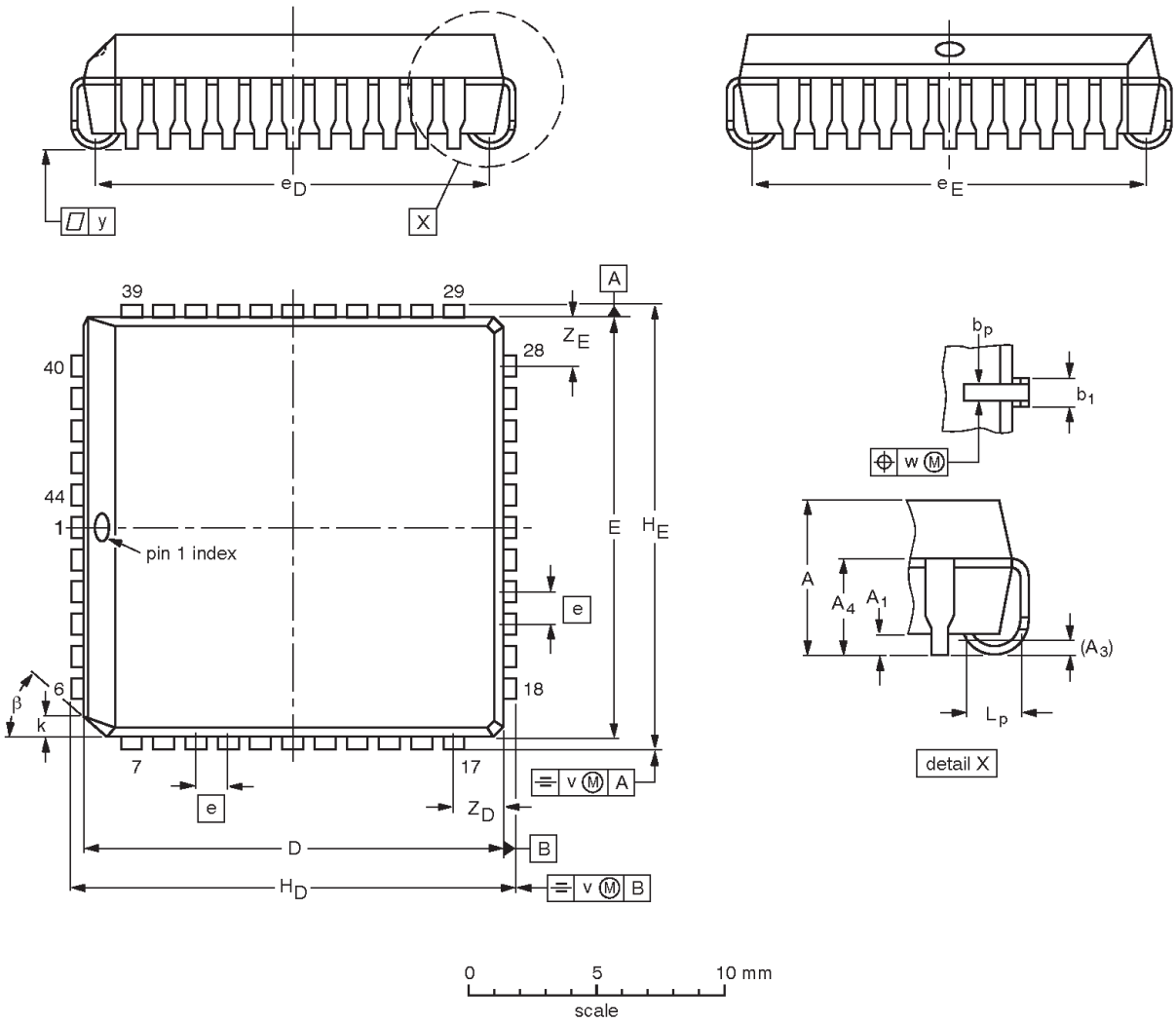


Figure 71. I_{CC} Test Condition, Power-Down mode. All other pins are disconnected; V_{CC} = 2V to 5.5V

NOTE:

* Ports 1.6 and 1.7 should be connected to V_{CC} through resistors of sufficiently high value such that the sink current into these pins does not exceed the I_{OL1} specification.

PLCC44: plastic leaded chip carrier; 44 leads SOT187-2



DIMENSIONS (mm dimensions are derived from the original inch dimensions)

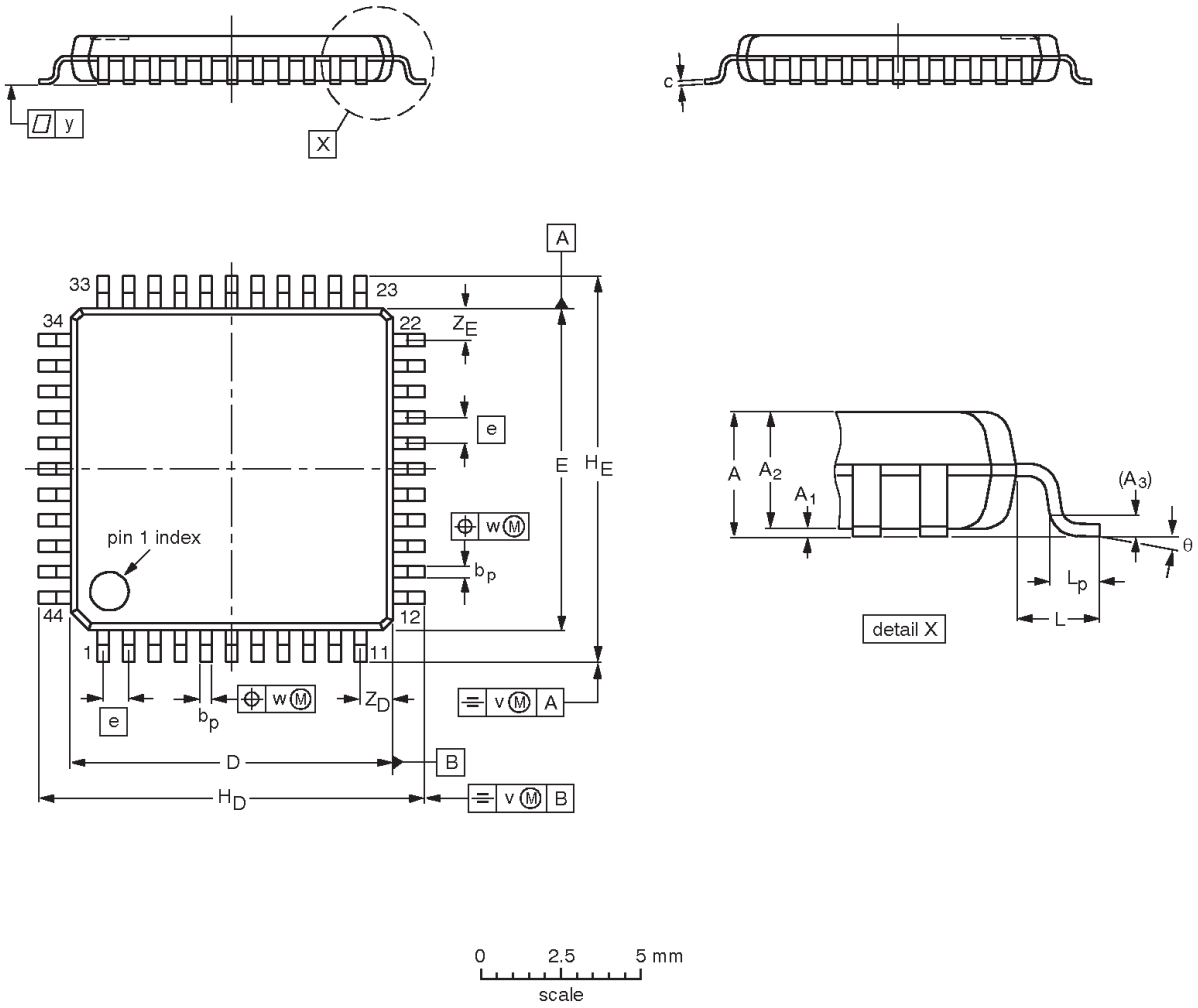
UNIT	A	A_1 min.	A_3	A_4 max.	b_p	b_1	$D^{(1)}$	$E^{(1)}$	e	e_D	e_E	H_D	H_E	k	L_p	v	w	y	$Z_D^{(1)}$ max.	$Z_E^{(1)}$ max.	β
mm	4.57 4.19	0.51	0.25	3.05	0.53 0.33	0.81 0.66	16.66 16.51	16.66 16.51	1.27	16.00 14.99	16.00 14.99	17.65 17.40	17.65 17.40	1.22 1.07	1.44 1.02	0.18	0.18	0.1	2.16	2.16	45°
inches	0.180 0.165	0.02	0.01	0.12	0.021 0.013	0.032 0.026	0.656 0.650	0.656 0.650	0.05	0.63 0.59	0.63 0.59	0.695 0.685	0.695 0.685	0.048 0.042	0.057 0.040	0.007	0.007	0.004	0.085	0.085	

Note

1. Plastic or metal protrusions of 0.25 mm (0.01 inch) maximum per side are not included.

OUTLINE VERSION	REFERENCES				EUROPEAN PROJECTION	ISSUE DATE
	IEC	JEDEC	JEITA			
SOT187-2	112E10	MS-018	EDR-7319			~~99-12-27~~ 01-11-14

LQFP44: plastic low profile quad flat package; 44 leads; body 10 x 10 x 1.4 mm **SOT389-1**



DIMENSIONS (mm are the original dimensions)

UNIT	A max.	A_1	A_2	A_3	b_p	c	$D^{(1)}$	$E^{(1)}$	e	H_D	H_E	L	L_p	v	w	y	$Z_D^{(1)}$	$Z_E^{(1)}$	θ
mm	1.6	0.15 0.05	1.45 1.35	0.25	0.45 0.30	0.20 0.12	10.1 9.9	10.1 9.9	0.8	12.15 11.85	12.15 11.85	1	0.75 0.45	0.2	0.2	0.1	1.14 0.85	1.14 0.85	7^o 0^o

Note

1. Plastic or metal protrusions of 0.25 mm maximum per side are not included.

OUTLINE VERSION	REFERENCES				EUROPEAN PROJECTION	ISSUE DATE
	IEC	JEDEC	JEITA			
SOT389-1	136E08	MS-026				~~00-01-19~~ 02-06-07

REVISION HISTORY

Rev	Date	Description
_4	20021028	**Product data (9397 750 10403); replaces P89C660/P89C662/P89C664 of 2001 Jul 19 (9397 750 08584) and P89C668 of 2001 Jul 27 (9397 750 08651)** Engineering Change Notice 853–2392 29118 (date: 20021028) Modifications: ● Integrated 89C668 in 89C66x datasheet ● Added more description on I^2C, Timer 0 and Timer 1, and Enhanced UART ● P2.6 must be high to activate the boot loader by hardware (ISP section).

Data sheet status

Level	Data sheet status [1]	Product status [2] [3]	Definitions
I	Objective data	Development	This data sheet contains data from the objective specification for product development. Philips Semiconductors reserves the right to change the specification in any manner without notice.
II	Preliminary data	Qualification	This data sheet contains data from the preliminary specification. Supplementary data will be published at a later date. Philips Semiconductors reserves the right to change the specification without notice, in order to improve the design and supply the best possible product.
III	Product data	Production	This data sheet contains data from the product specification. Philips Semiconductors reserves the right to make changes at any time in order to improve the design, manufacturing and supply. Relevant changes will be communicated via a Customer Product/Process Change Notification (CPCN).

[1] Please consult the most recently issued data sheet before initiating or completing a design.

[2] The product status of the device(s) described in this data sheet may have changed since this data sheet was published. The latest information is available on the Internet at URL http://www.semiconductors.philips.com.

[3] For data sheets describing multiple type numbers, the highest-level product status determines the data sheet status.

Definitions

Short-form specification — The data in a short-form specification is extracted from a full data sheet with the same type number and title. For detailed information see the relevant data sheet or data handbook.

Limiting values definition — Limiting values given are in accordance with the Absolute Maximum Rating System (IEC 60134). Stress above one or more of the limiting values may cause permanent damage to the device. These are stress ratings only and operation of the device at these or at any other conditions above those given in the Characteristics sections of the specification is not implied. Exposure to limiting values for extended periods may affect device reliability.

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.

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 license 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.

Contact information

For additional information please visit
http://www.semiconductors.philips.com. Fax: **+31 40 27 24825**

For sales offices addresses send e-mail to:
sales.addresses@www.semiconductors.philips.com.

© Koninklijke Philips Electronics N.V. 2002
All rights reserved. Printed in U.S.A.

Date of release: 10-02

Document order number: 9397 750 10403

Let's make things better.

Philips Semiconductors

PHILIPS

Looking for pricing, stock, or lifecycle information?

Click below to explore more details on WIN SOURCE:

- View P89C662HBA/00,512 on WIN SOURCE
- NXP / Nexperia Information

Optimize Your Supply Chain with WIN SOURCE Solutions

- Global Sourcing Solution
- Obsolete Management
- Cost Control Management
- Shortage Management
- Alternative Solution
- Excess Inventory Management