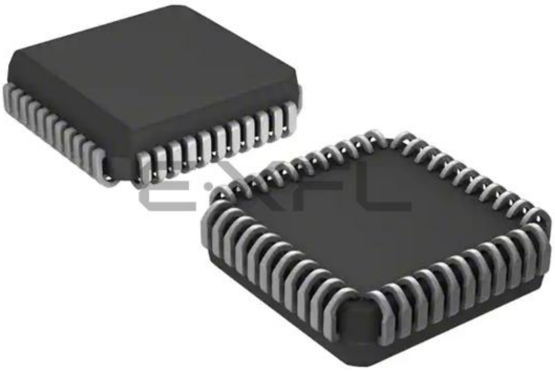


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Obsolete
Core Processor	8051
Core Size	8-Bit
Speed	33MHz
Connectivity	I²C, UART/USART
Peripherals	POR, PWM, WDT
Number of I/O	32
Program Memory Size	32KB (32K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	4.5V ~ 5.5V
Data Converters	-
Oscillator Type	Internal
Operating Temperature	0°C ~ 70°C (TA)
Mounting Type	Surface Mount
Package / Case	44-LCC (J-Lead)
Supplier Device Package	44-PLCC (16.59x16.59)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/p89c662hba-00-512

BLOCK DIAGRAM (CPU-ORIENTED)

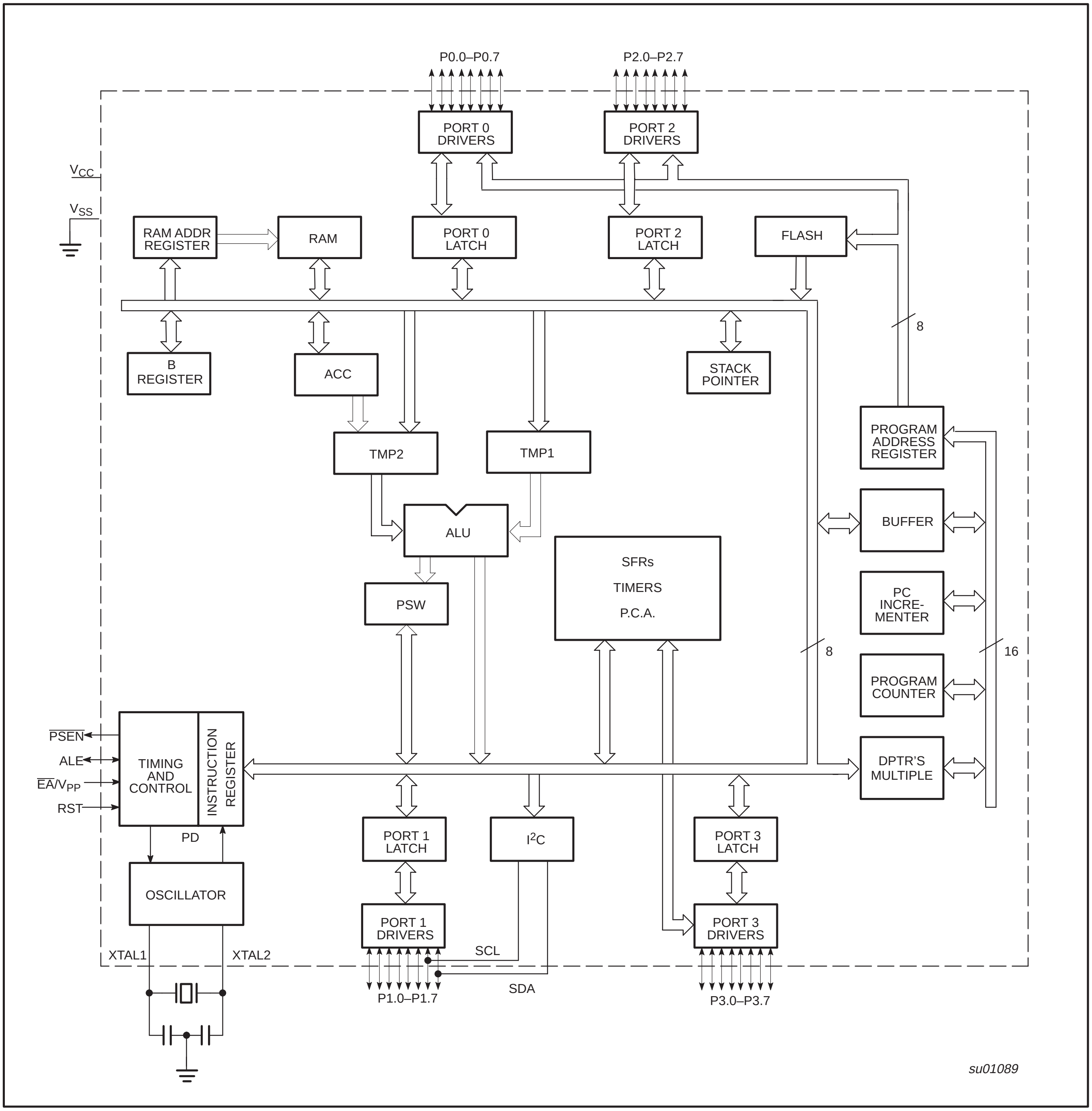


Table 1. Special Function Registers

SYMBOL	DESCRIPTION	DIRECT ADDRESS	BIT ADDRESS, SYMBOL, OR ALTERNATIVE PORT FUNCTION								RESET VALUE
			MSB							LSB	
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.

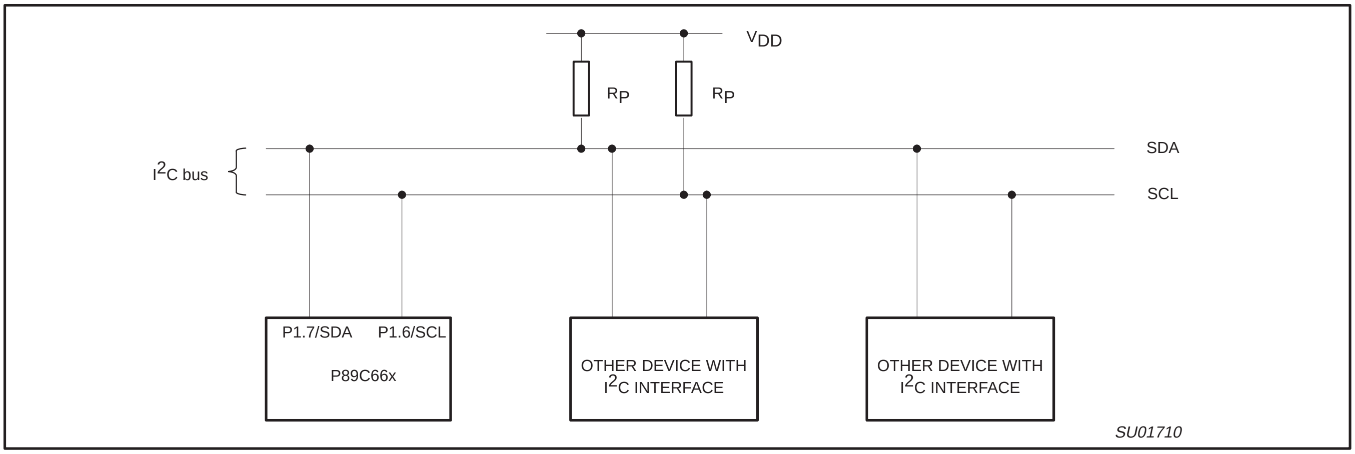

VDD
RP
RP
I2C bus
SDA
SCL
P1.7/SDA
P1.6/SCL
P89C66x
OTHER DEVICE WITH I2C INTERFACE
OTHER DEVICE WITH I2C INTERFACE
SU01710

Figure 1. Typical I^{2}C Bus Configuration

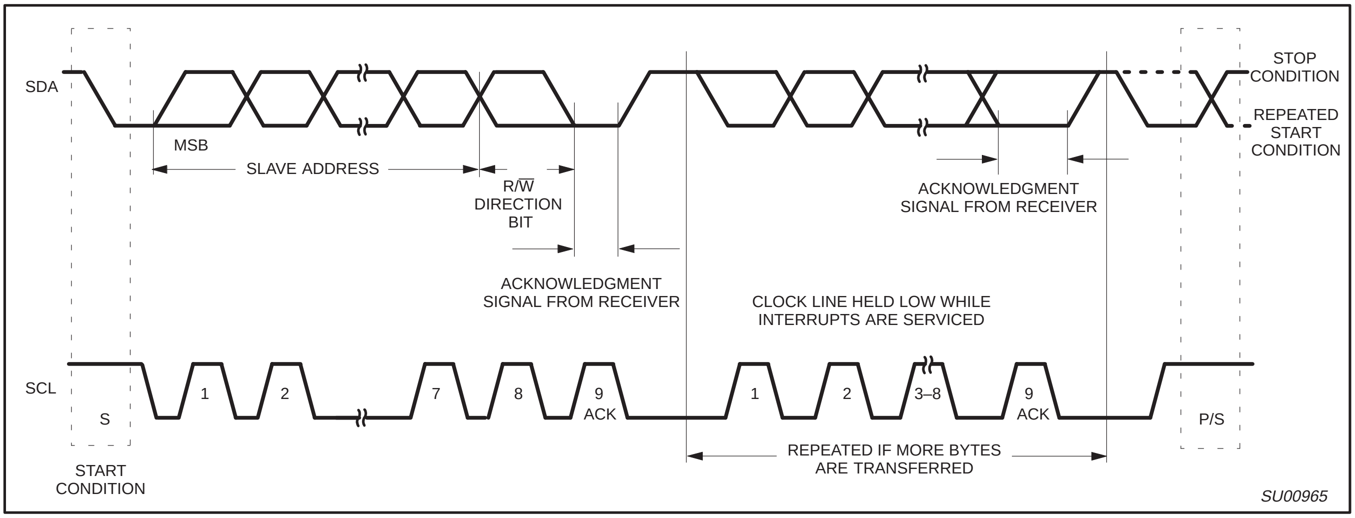

SDA
MSB
SLAVE ADDRESS
R/W DIRECTION BIT
ACKNOWLEDGMENT SIGNAL FROM RECEIVER
CLOCK LINE HELD LOW WHILE INTERRUPTS ARE SERVICED
ACKNOWLEDGMENT SIGNAL FROM RECEIVER
STOP CONDITION
REPEATED START CONDITION
SCL
S
1
2
7
8
9
ACK
1
2
3–8
9
ACK
P/S
START CONDITION
REPEATED IF MORE BYTES ARE TRANSFERRED
SU00965

Figure 2. Data Transfer on the I^{2}C Bus

SIO1 Implementation and Operation

Figure 3 shows how the on-chip I^{2}C bus interface is implemented, and the following text describes the individual blocks.

Input Filters and Output Stages

The input filters have I^{2}C 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 I^{2}C bus and V_{DD} is switched off, the I^{2}C 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

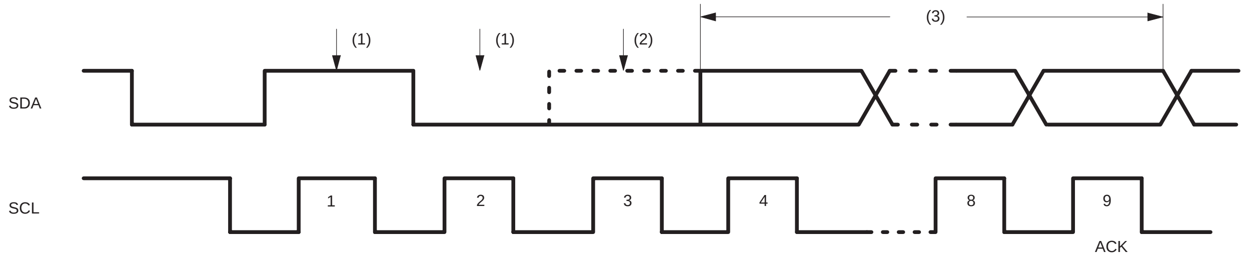
This 8-bit special function register contains a byte of serial data to be transmitted or a byte which has just been received. Data in S1DAT is always shifted from right to left; the first bit to be transmitted is the MSB (bit 7) and, after a byte has been received, the first bit of received data is located 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.

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^{2}C 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.

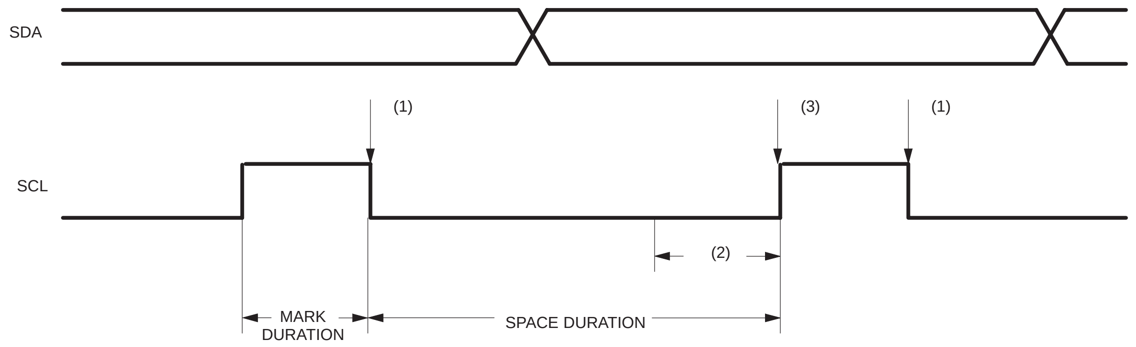
A slave may stretch the space duration to slow down the bus master. The space duration may also be stretched for handshaking purposes. This can be done after each bit or after a complete byte transfer. SIO1 will stretch the SCL space duration after a byte has been transmitted or received and the acknowledge 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

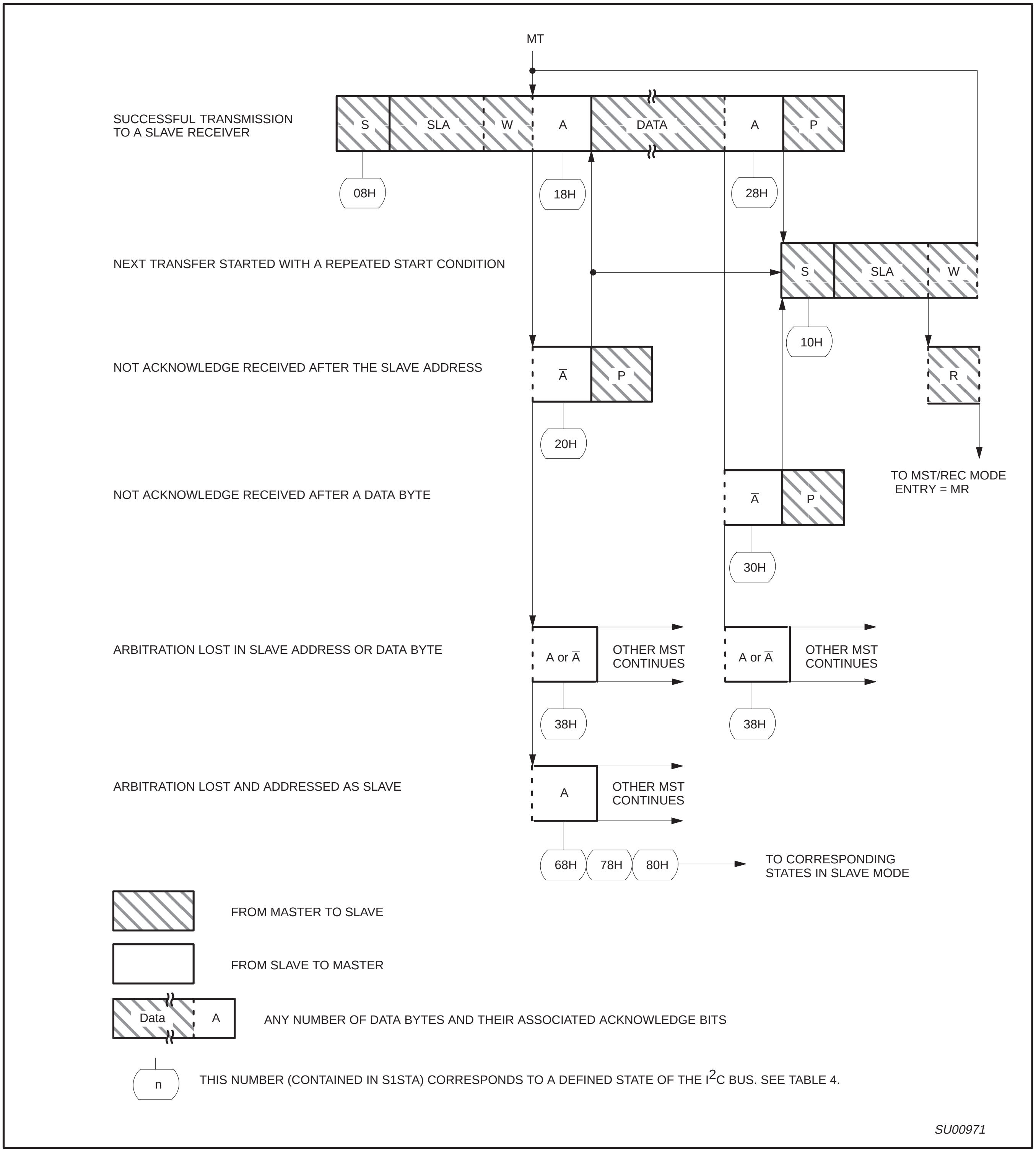

MT
SUCCESSFUL TRANSMISSION
TO A SLAVE RECEIVER
S
SLA
W
A
DATA
A
P
08H
18H
28H
NEXT TRANSFER STARTED WITH A REPEATED START CONDITION
S
SLA
W
10H
NOT ACKNOWLEDGE RECEIVED AFTER THE SLAVE ADDRESS
$\overline{A}$
P
20H
R
TO MST/REC MODE
ENTRY = MR
NOT ACKNOWLEDGE RECEIVED AFTER A DATA BYTE
$\overline{A}$
P
30H
ARBITRATION LOST IN SLAVE ADDRESS OR DATA BYTE
A or $\overline{A}$
OTHER MST
CONTINUES
A or $\overline{A}$
OTHER MST
CONTINUES
38H
38H
ARBITRATION LOST AND ADDRESSED AS SLAVE
A
OTHER MST
CONTINUES
68H
78H
80H
TO CORRESPONDING
STATES IN SLAVE MODE
FROM MASTER TO SLAVE
FROM SLAVE TO MASTER
Data
A
ANY NUMBER OF DATA BYTES AND THEIR ASSOCIATED ACKNOWLEDGE BITS
n
THIS NUMBER (CONTAINED IN S1STA) CORRESPONDS TO A DEFINED STATE OF THE I2C BUS. SEE TABLE 4.
SU00971

Figure 8. Format and States in the Master Transmitter mode

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.

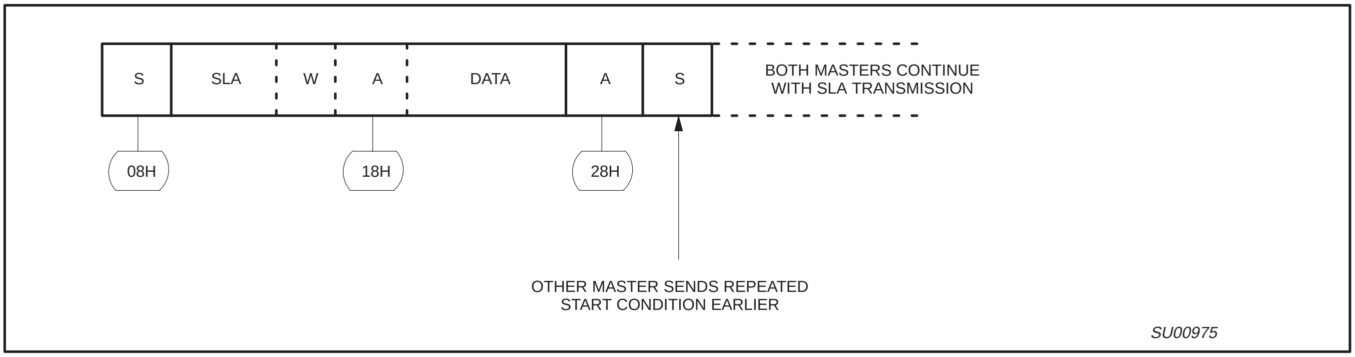


Figure 12. Simultaneous Repeated START Conditions from 2 Masters

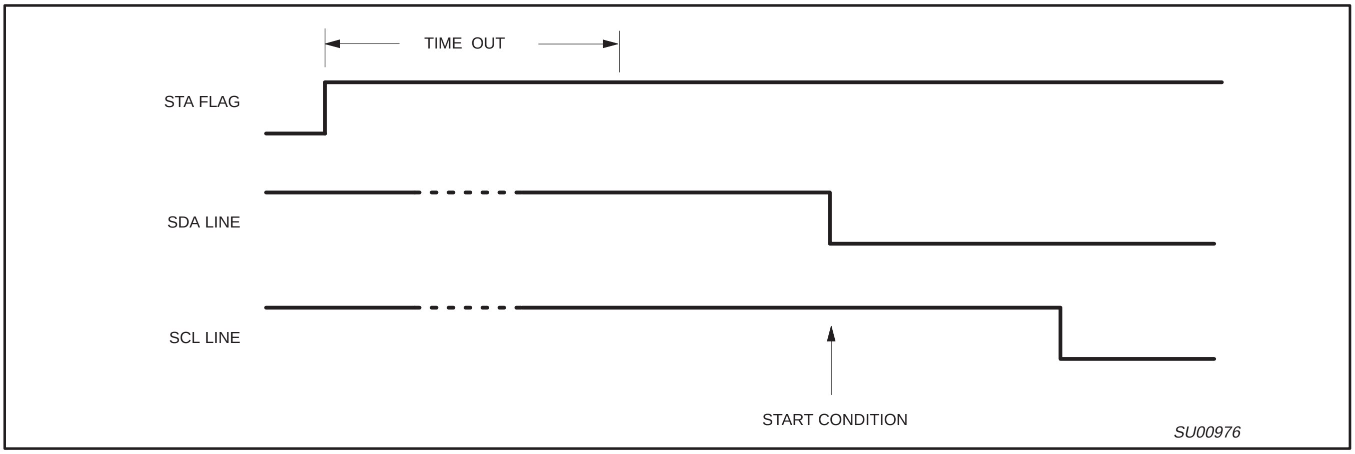


Figure 13. Forced Access to a Busy I^{2}C Bus

I^{2}C Bus Obstructed by a Low Level on SCL or SDA

An I^{2}C 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 I^{2}C 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.

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

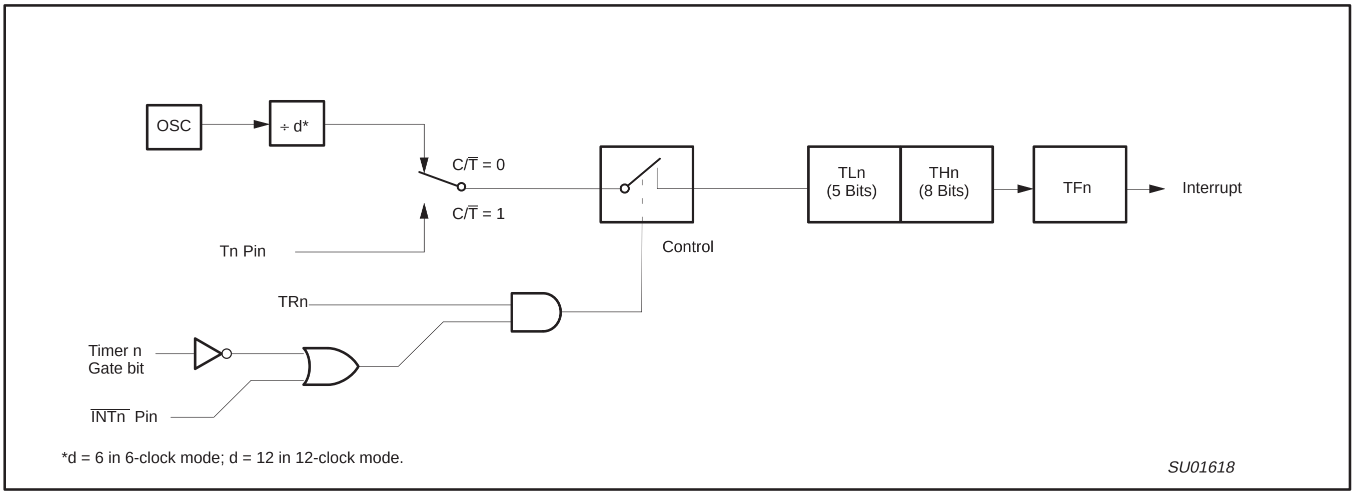
7	6	5	4	3	2	1	0
GATE	$C/\overline{T}$	M1	M0	GATE	$C/\overline{T}$	M1	M0
TIMER 1				TIMER 0			

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



*d = 6 in 6-clock mode; d = 12 in 12-clock mode.

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

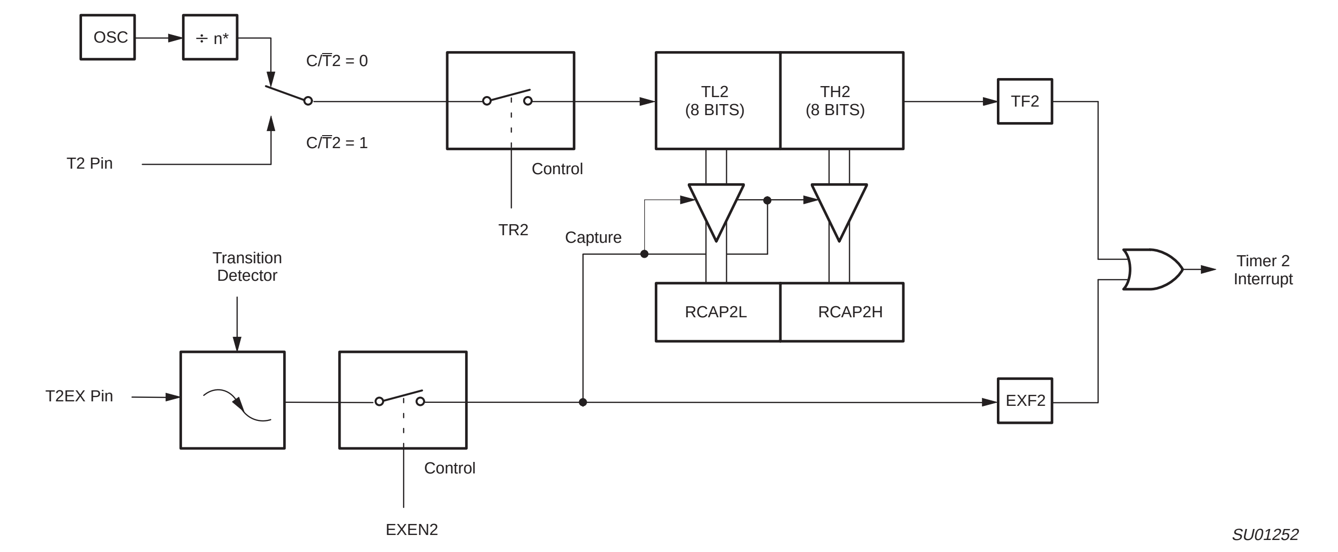
BIT	SYMBOL	FUNCTION
TCON.7	TF1	Timer 1 overflow flag. Set by hardware on Timer/Counter overflow. Cleared by hardware when processor vectors to interrupt routine, or clearing the bit in software.
TCON.6	TR1	Timer 1 Run control bit. Set/cleared by software to turn Timer/Counter on/off.
TCON.5	TF0	Timer 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

Table 9. Timer 2 Operating Modes

RCLK + TCLK	CP/$\overline{\text{RL2}}$	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

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

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

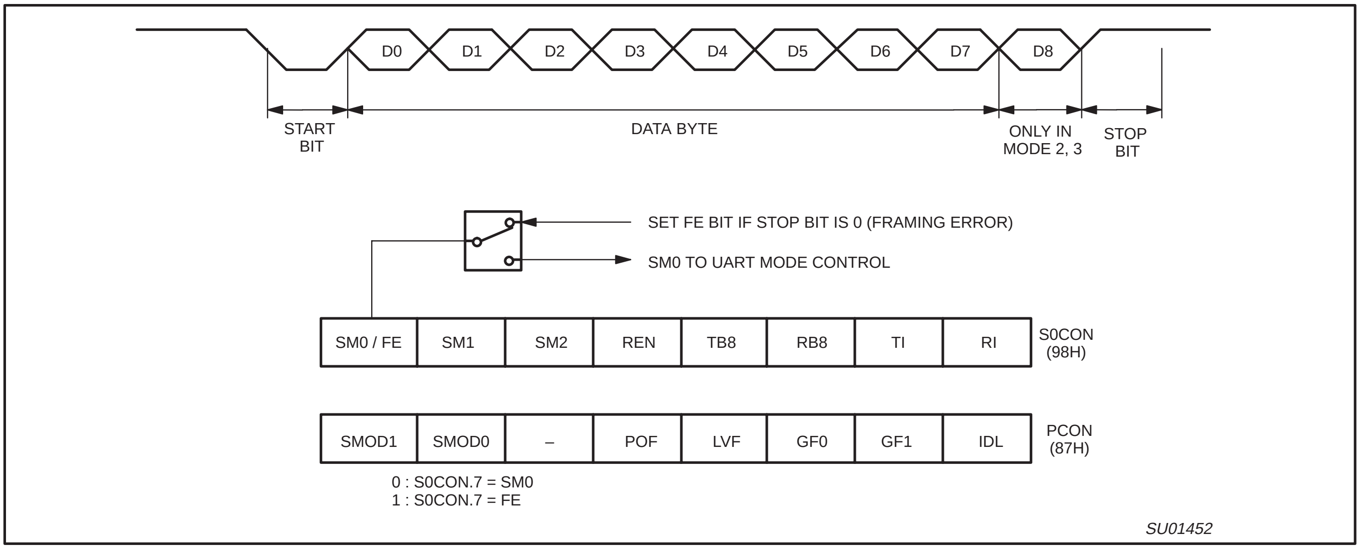


Figure 33. UART Framing Error Detection

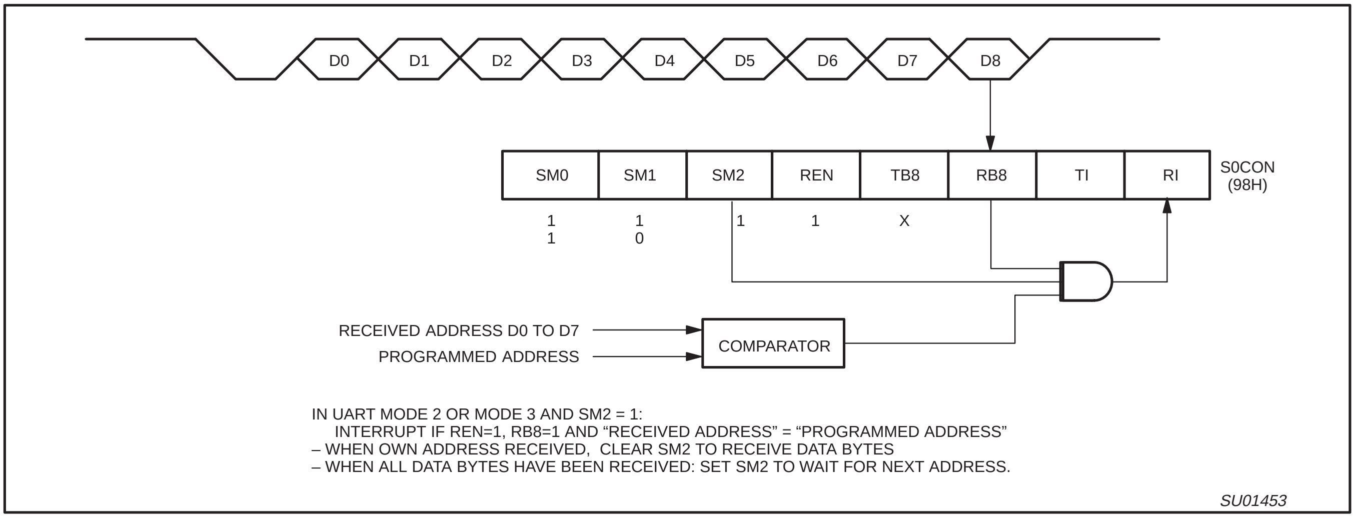

D0
D1
D2
D3
D4
D5
D6
D7
D8
SM0
SM1
SM2
REN
TB8
RB8
TI
RI
S0CON (98H)
1
1
1
0
1
1
X
RECEIVED ADDRESS D0 TO D7
PROGRAMMED ADDRESS
COMPARATOR
IN UART MODE 2 OR MODE 3 AND SM2 = 1:
INTERRUPT IF REN=1, RB8=1 AND “RECEIVED ADDRESS” = “PROGRAMMED ADDRESS”
– WHEN OWN ADDRESS RECEIVED, CLEAR SM2 TO RECEIVE DATA BYTES
– WHEN ALL DATA BYTES HAVE BEEN RECEIVED: SET SM2 TO WAIT FOR NEXT ADDRESS.
SU01453

Figure 34. UART Multiprocessor Communication, Automatic Address Recognition

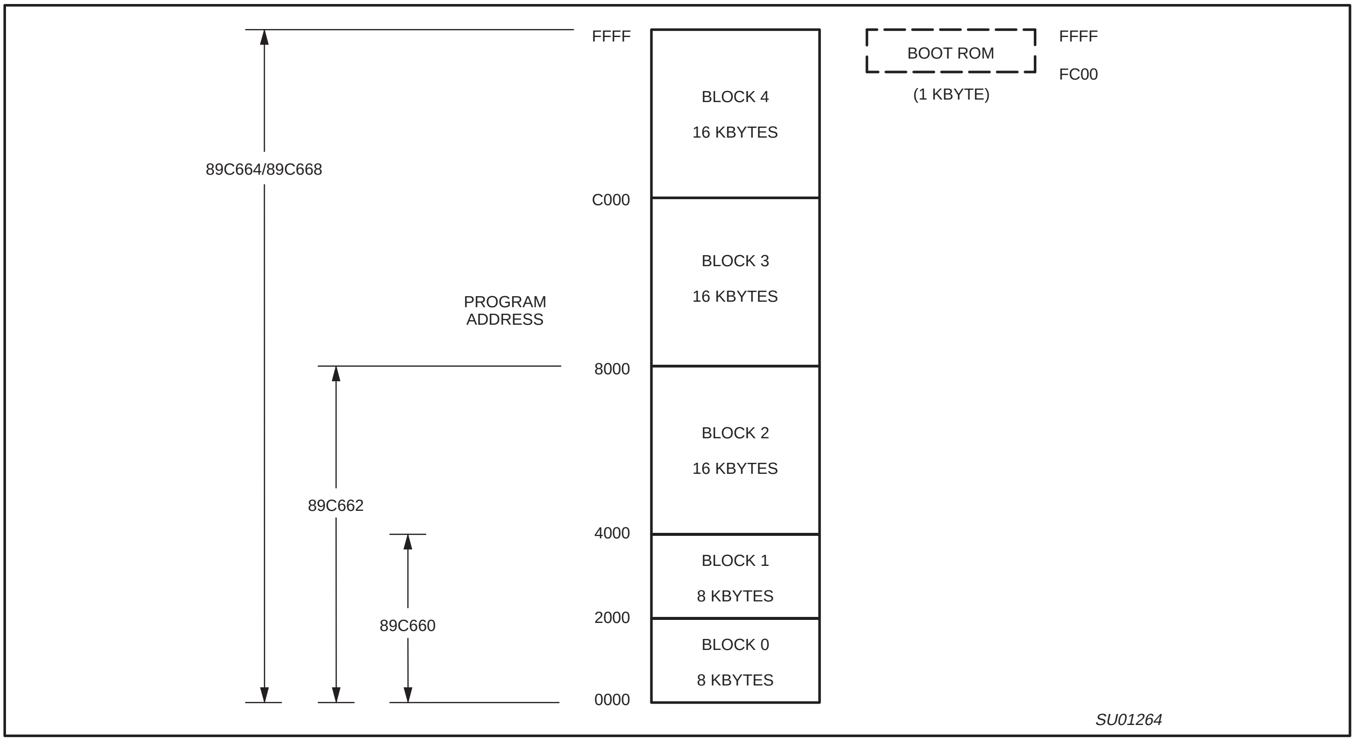


Figure 55. Flash Memory Configurations

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.

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`

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

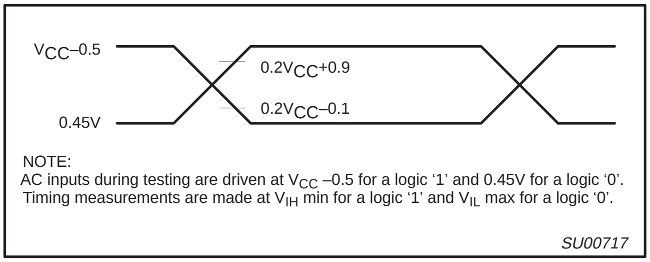
AC ELECTRICAL CHARACTERISTICS (12 CLOCK MODE)

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 $\overline{\text{PSEN}}$ low	t_{CLCL}–25	–	5	–	ns
t_{PLPH}	57	$\overline{\text{PSEN}}$ pulse width	$3t_{CLCL}$–45	–	45	–	ns
t_{PLIV}	57	$\overline{\text{PSEN}}$ low to valid instruction in	–	$3t_{CLCL}$–60	–	30	ns
t_{PXIX}	57	Input instruction hold after $\overline{\text{PSEN}}$	0	–	0	–	ns
t_{PXIZ}	57	Input instruction float after $\overline{\text{PSEN}}$	–	t_{CLCL}–25	–	5	ns
t_{AVIV}	57	Address to valid instruction in	–	$5t_{CLCL}$–80	–	70	ns
t_{PLAZ}	57	$\overline{\text{PSEN}}$ low to address float	–	10	–	10	ns
Data Memory							
t_{RLRH}	58, 59	$\overline{\text{RD}}$ pulse width	$6t_{CLCL}$–100	–	82	–	ns
t_{WLWH}	58, 59	$\overline{\text{WR}}$ pulse width	$6t_{CLCL}$–100	–	82	–	ns
t_{RLDV}	58, 59	$\overline{\text{RD}}$ low to valid data in	–	$5t_{CLCL}$–90	–	60	ns
t_{RHDX}	58, 59	Data hold after $\overline{\text{RD}}$	0	–	0	–	ns
t_{RHDZ}	58, 59	Data float after $\overline{\text{RD}}$	–	$2t_{CLCL}$–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 $\overline{\text{RD}}$ or $\overline{\text{WR}}$ low	$3t_{CLCL}$–50	$3t_{CLCL}$+50	40	140	ns
t_{AVWL}	58, 59	Address valid to $\overline{\text{WR}}$ low or $\overline{\text{RD}}$ low	$4t_{CLCL}$–75	–	45	–	ns
t_{QVWX}	58, 59	Data valid to $\overline{\text{WR}}$ transition	t_{CLCL}–30	–	0	–	ns
t_{WHQX}	58, 59	Data hold after $\overline{\text{WR}}$	t_{CLCL}–25	–	5	–	ns
t_{QVWH}	59	Data valid to $\overline{\text{WR}}$ high	$7t_{CLCL}$–130	–	80	–	ns
t_{RLAZ}	58, 59	$\overline{\text{RD}}$ low to address float	–	0	–	0	ns
t_{WHLH}	58, 59	$\overline{\text{RD}}$ or $\overline{\text{WR}}$ 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:

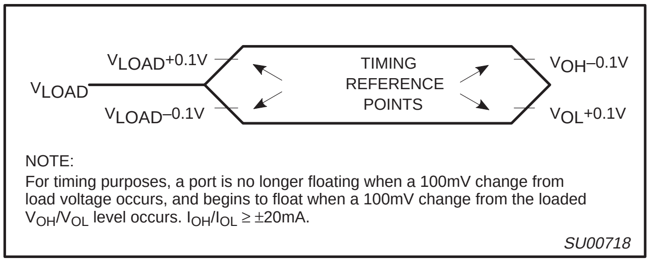
1. Parameters are valid over operating temperature range unless otherwise specified.
2. Load capacitance for port 0, ALE, and $\overline{\text{PSEN}}$ = 100 pF, load capacitance for all other outputs = 80 pF.
3. Interfacing the microcontroller to devices with float times up to 45 ns is permitted. This 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.



NOTE:
AC inputs during testing are driven at V_{CC} –0.5 for a logic ‘1’ and 0.45V for a logic ‘0’. Timing measurements are made at V_{IH} min for a logic ‘1’ and V_{IL} max for a logic ‘0’.

SU00717

Figure 62. AC Testing Input/Output



NOTE:
For timing purposes, a port is no longer floating when a 100mV change from load voltage occurs, and begins to float when a 100mV change from the loaded V_{OH}/V_{OL} level occurs. I_{OH}/I_{OL} ≥ ±20mA.

SU00718

Figure 63. Float Waveform

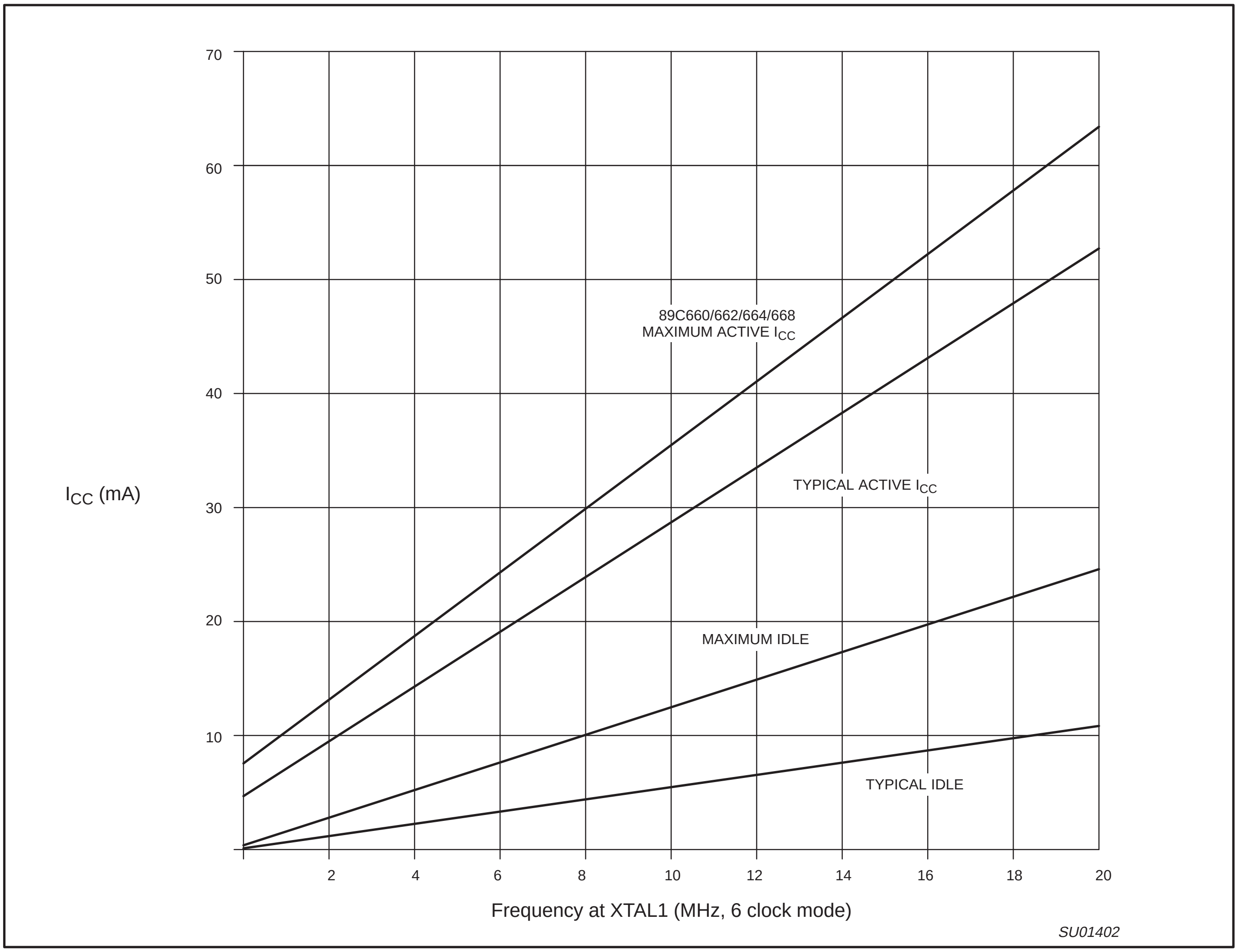
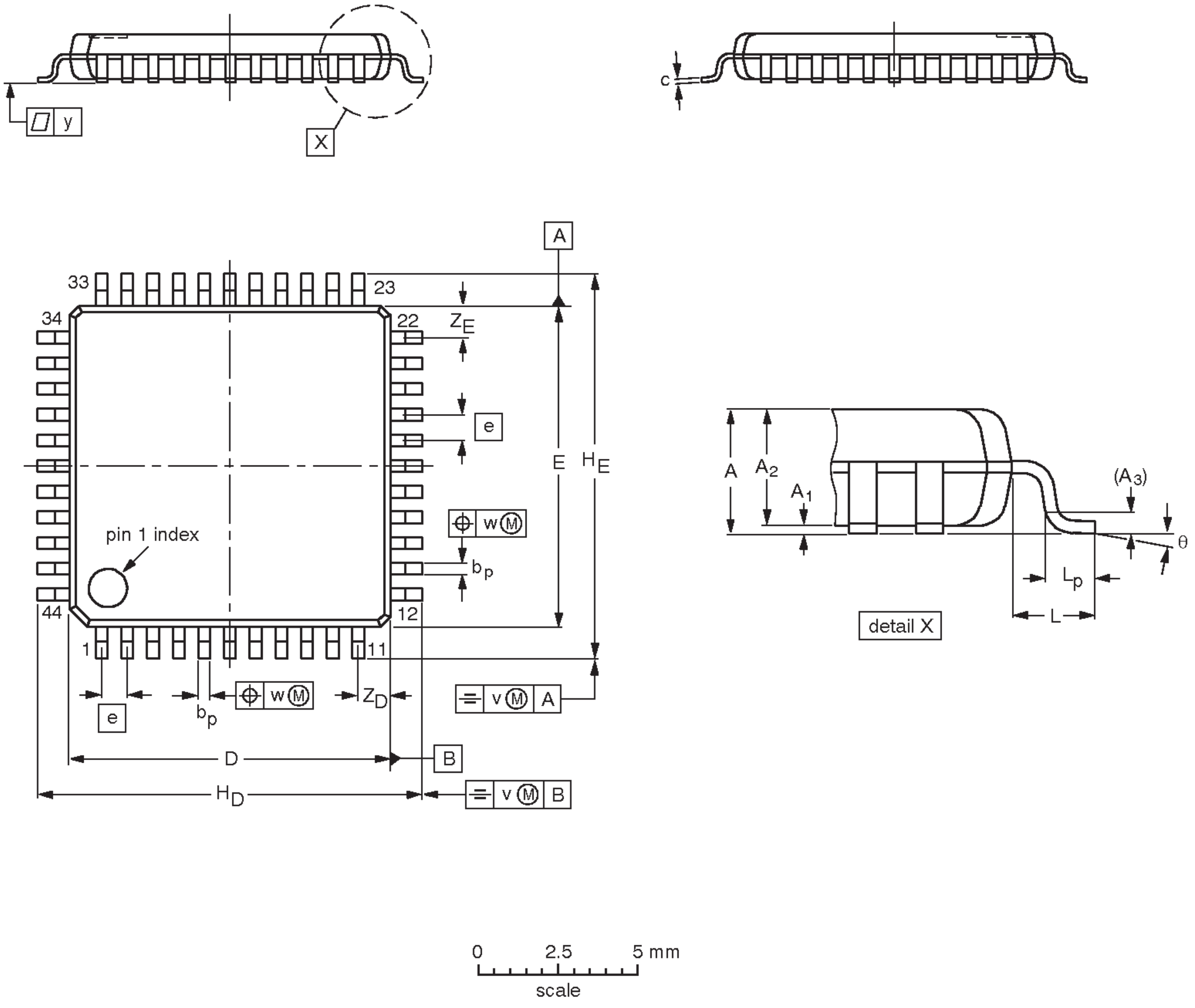


Figure 64. I_{CC} vs. FREQ
Valid only within frequency specifications of the device under test

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