Features

- 8 Bit Microcomputer with 8051 architecture
- Fully static design
- 64KB internal Flash,
- 8KB internal RAM
- In – Application Programming (IAP)
- In – System Programming (ISP)
- Low Standby Current At Full Supply Voltage
- 0-33 MHz Operation (12 clock mode)
- 4 8-Bit Bidirectional Ports
- Boolean Processor
- 44 pin PLCC Package
- Built In Power Management
- Three 16 bit timer/counters
- Full Duplex Serial Channel With Hardware Address Decode
- Multiple Source, Four Level Interrupt Capability
- Programmable Counter Array
- Watchdog Timer For Greater System Reliability
- Dual Data Pointers
- Program Security Features
- 6 – clock / 12 clock select
- TWI Serial Interface

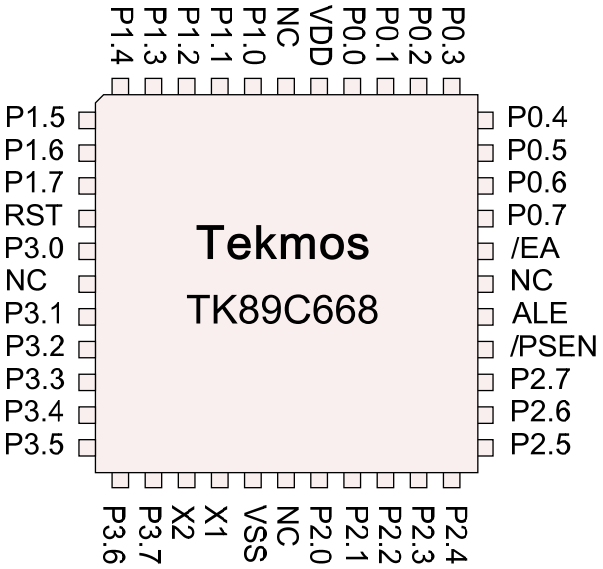
General Description

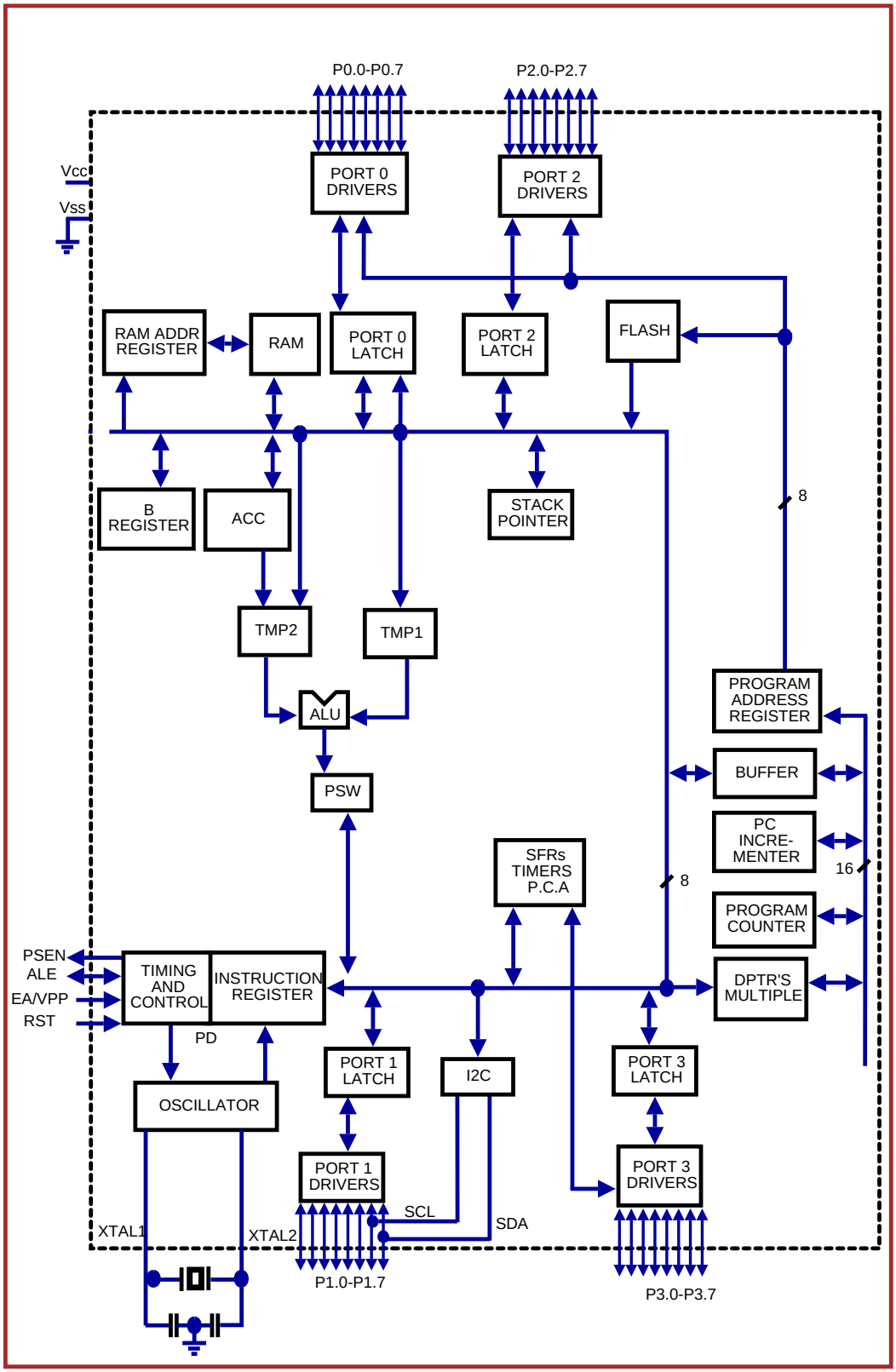
The TK89C668 is based on the 8051- microcontroller architecture. With 64Kx8 of Flash memory and 8Kx8 of internal RAM, these parts are a pin-for-pin replacement for existing NXP P89C668 microcontroller.

The TK89C668 contains four 8-bit bidirectional parallel ports, two external interrupt sources, three timer/counters, a serial port with a hardware interrupt capability and a frame error detect flag, power management, a programmable counter array (PCA), TWI Serial Interface, and enhanced data security features. These peripherals are supported by a multiple source, four level interrupt capability. The core processor contains 256 bytes of scratchpad RAM and 64KB of program storage. The core also contains 7936 bytes of Extended RAM.

The TWI (Two Wire Interface) is fully implemented in hardware. This allows byte level transfers, resulting in simpler software overhead.

Figure 1 shows the basic TK89C668 configuration.





TK89C668 Block Diagram - Figure 1

Pinout Description

PLCC			Name	Description
1			NC	
2			P1.0 + T2	Port 1. Bit 0 + Timer 2
3			P1.1 + T2EX	Port 1. Bit 1 + Timer 2 Exchange
4			P1.2 + ECI	Port 1. Bit 2 + PCA External Clock Input
5			P1.3 + CEX0	Port 1. Bit 3 + PCA Capture / Compare External I/O for PCA 0
6			P1.4 + CEX1	Port 1. Bit 4 + PCA Capture / Compare External I/O for PCA 1
7			P1.5 + CEX2	Port 1. Bit 5 + PCA Capture / Compare External I/O for PCA 2
8			P1.6 + SCL	Port 1. Bit 5 + SCL for TWI
9			P1.7 + SDA	Port 1. Bit 5 + SDA for TWI
10			RST	Reset
11			P3.0 + RxD	Port 3, Bit 0 + Serial Port Receive Data
12			NC	
13			P3.1 + TxD	Port 3, Bit 1 + Serial Port Transmit Data
14			P3.2 + /INT0	Port 3, Bit 2 + External Interrupt 0
15			P3.3 + /INT1	Port 3, Bit 3 + External Interrupt 1
16			P3.4 + T0 + CEX3	Port 3, Bit 4 + Timer 0 + PCA Capture / Compare External I/O for PCA3
17			P3.5 + T1 + CEX4	Port 3, Bit 5 + Timer 1 + PCA Capture / Compare External I/O for PCA4
18			P3.6 + /WR	Port 3, Bit 6 + Data Write
19			P3.7 + /RD	Port 3, Bit 7 + Data Read
20			XTAL2	Clock / Crystal Oscillator Output
21			XTAL1	Clock / Crystal Oscillator Input
22			VSS	Ground
23			NC	
24			P2.0 + A8	Port 2, Bit 8 + Address Bus Bit 8
25			P2.1 + A9	Port 2, Bit 9 + Address Bus Bit 9
26			P2.2 + A10	Port 2, Bit 10 + Address Bus Bit 10
27			P2.3 + A11	Port 2, Bit 11 + Address Bus Bit 11
28			P2.4 + A12	Port 2, Bit 12 + Address Bus Bit 12
29			P2.5 + A13	Port 2, Bit 13 + Address Bus Bit 13
30			P2.6 + A14	Port 2, Bit 14 + Address Bus Bit 14
31			P2.7 + A15	Port 2, Bit 15 + Address Bus Bit 15
32			/PSEN	Program Store Enable
33			ALE	Address Latch Enable
34			NC	
35			/EA	External Address
36			P0.7 + AD7	Port 0, Bit 7 + Address / Data Bus Bit 7
37			P0.6 + AD6	Port 0, Bit 6 + Address / Data Bus Bit 6
38			P0.5 + AD5	Port 0, Bit 5 + Address / Data Bus Bit 5
39			P0.4 + AD4	Port 0, Bit 4 + Address / Data Bus Bit 4
40			P0.3 + AD3	Port 0, Bit 3 + Address / Data Bus Bit 3
41			P0.2 + AD2	Port 0, Bit 2 + Address / Data Bus Bit 2
42			P0.1 + AD1	Port 0, Bit 1 + Address / Data Bus Bit 1
43			P0.0 + AD0	Port 0, Bit 0 + Address / Data Bus Bit 0
44			VDD	Positive Supply

Pin Descriptions

Port 0
P00 – P07

With **EA** high, these pins are connected to Port 0.

The pins are open-drain is a general purpose parallel port.

Port 1
P10 – P17

The pins are open-drain is a general purpose parallel port.

Port 2
P20 – P27

With **EA** high, these pins are connected to Port 2.

The pins are open-drain is a general purpose parallel port.

Port 3
P30 – P307

The pins are open-drain is a general purpose parallel port.

There are six control signals associated with the TK8051. These are XTAL1, XTAL2, CLOCK, RESET, /EA, ALE and /PSEN.

XTAL1, XTAL2

The TK89C668 has a crystal oscillator connected to the XTAL1 and XTAL2 pins.

An external clock may be used by connecting it directly to the XTAL1 pin. When an external clock is used, the XTAT2 pin should be left unconnected.

Reset

The external RESET signal is sampled at S5P2. It must be held high for at least two machine cycles while the oscillator is running to take effect.

A Schmitt trigger should be used in the reset line when an external RC network is used for reset. The reset logic also has a special glitch removal circuit that ignores most glitches on the reset line.

During reset, the ports are initialized to FFH, the stack pointer to 07H, PCON with the exception of bit 4 to 00H, and all of the other SFR registers except SBUF to 00H. SBUF is not reset.

Also during reset, ALE and /PSEN become inputs that are held high with an internal pull-up. These two pins serve to enable various alternate modes.

ALE/PSEN/EA/P2.7/P2.6	Function: (When RESET is removed)
01XXX	Emulation Mode
10111	Flash Boot Loader
11XXX	Normal Mode

Emulation Mode

The TK89C668 will remain in reset if reset is removed from the part while ALE is held low and /PSEN is held high. This mode is used to support the use of in-circuit emulators, since all of the TK89C668 output pins are in a high impedance state during reset.

Architecture

The TK89C668 consists of a core controller surrounded by four general purpose I/O ports, up to 256 bytes of scratchpad RAM, 7936 byes of extended RAM, three timer counters, a serial port, a programmable counter array, a watchdog timer, an TWI Serial Interface controller, an enhanced interrupt controller, and various control registers. The processor supports 111 different opcodes, and references both a 64Kb program address space and a 64Kb data storage space.

One of the distinguishing features of the architecture is the Special Function Register (SFR) address space, into which all of the registers, peripherals, and scratchpad RAM are mapped. Many of the instructions operate on an SFR address rather than a specific register. This greatly increases the power of the instruction set.

Address Space

Program Storage Space

The TK89C668 processor operates out of three separate address spaces. The first of these is the

program space. This read-only address space consists of 64K bytes and is used to store the program code. This address space may be either internally access 64Kbytes of Flash ROM, or access data program space in external memory. This is selectable with the EA pin. When 0, external memory is used for program storage, otherwise internal Flash ROM is used. The program space is accessed by both opcode fetches and the MOVC instructions.

The TK89C668 contains up to 64KB of Flash memory.

Data Space

The second address space is referred to as the data space or extended memory (XRAM) and consists of 64K bytes that can be both read and write memory.

If the EXTRAM bit is set in the AUXR register, all of the extended memory is external to the TK89C668. If EXTRAM is cleared, the first 7936 bytes are internally accessed, with the balance external to the device.

While the data space is separate from the program space, these spaces can be combined in hardware by AND-ing the /PSEN and /RD signals to produce a composite read signal. This will allow the processor to execute code out of RAM. The data space is accessed by the MOVX instructions.

SFR Space

The third address space is referred to as the Special Function Register (SFR) space and consists of 384 bytes located internal to the TK89C668. These 384 bytes are mapped into 256 address locations.

The lower 128 bytes of the SFR space consist of scratchpad RAM. While any of these addresses may be used by the programmer, a number of them have special uses. The lower 32 bytes are organized into four 8-byte banks. The bank select bits in the PSW register select one of these banks to be used as an operand by the instruction set. Registers 0 to 7 in the bank are referenced by the register direct opcodes. Registers 0 and 1 may also contain an address that is referenced by the register indirect opcodes.

Registers 20H to 2FH are bit addressable by the Boolean instructions. Register 20H, bit 0 has the Boolean address 00H. Register 2FH, bit 7 has the Boolean address 7FH.

The high level Boolean addresses (80H - FFH) are mapped into the upper 128 bytes of the SFR address space. The upper 5 bits of the Boolean address are combined with three lower bits of 0 to specify a register address. The lower three bits of the Boolean address specify a bit within the designated register.

The upper 128 bytes of the SFR address space contain both 128 bytes of general purpose scratchpad RAM and the SFR registers themselves. They are separated by the addressing means used to access them. Direct addressing accesses the SFR registers. Register indirect addressing will access the scratchpad RAM. Stack pointer operations are considered register indirect addressing. The SFR address space is accessed by all instructions except the MOVC and MOVX instructions.

SFR Registers Bit Maps

Name	Description	SFR Addr.	Reset Value	Bit Functions D7	D6	D5	D4	D3	D2	D1	D0
ACC	Accumulator	E0h									
AUXR	Aux. Register	8Eh	00h	x	x	x	x	x	x	EXTRAM	A0
AUXR1	Aux. Register 1	A2h	00h	x	x	x		GF2	0		DPS
B	B Register	F0h									
CCAP0H	PCA Module 0 Capture High	FAh									
CCAP1H	PCA Module 1 Capture High	FBh									
CCAP2H	PCA Module 2 Capture High	FCh									
CCAP3H	PCA Module 3 Capture High	FDh									
CCAP4H	PCA Module 4 Capture High	FEh									
CCAP0L	PCA Module 0 Capture Low	EAh									
CCAP1L	PCA Module 1 Capture Low	EBh									
CCAP2L	PCA Module 2 Capture Low	ECh									
CCAP3L	PCA Module 3 Capture Low	EDh									
CCAP4L	PCA Module 4 Capture Low	EEh									
CCAPM0	PCA Module 0 Mode	C2h	00h	x	ECMO0	CAPP0	CAPN0	MAT_0	TOG_0	PWM_0	ECCF0
CCAPM1	PCA Module 1 Mode	C3h	00h	x	ECMO1	CAPP1	CAPN1	MAT_1	TOG_1	PWM_1	ECCF1
CCAPM2	PCA Module 2 Mode	C4h	00h	x	ECMO0	CAPP2	CAPN2	MAT_2	TOG_2	PWM_2	ECCF2
CCAPM3	PCA Module 3 Mode	C5h	00h	x	ECMO3	CAPP3	CAPN3	MAT_3	TOG_3	PWM_3	ECCF3
CCAPM4	PCA Module 4 Mode	C6h	00h	x	ECMO4	CAPP4	CAPN4	MAT_4	TOG_4	PWM_4	ECCF4
CCON	PCA Counter Control	C0h	00h	CF	CR	x	CCF4	CCF3	CCF2	CCF1	CCF0
CH	PCA Counter High	F9h									
CL	PCA Counter Low	E9h									
CMOD	PCA Counter Mode	C1h	00h	CIDL	WDTE	x	x	x	CPS1	CPS0	ECF
DPH	Data Pointer High	83h									
DPL	Data Pointer Low	82h									
IEN0	Interrupt Enable 0	A8h		EA	EC	ES1	ES0	ET1	EX1	ET0	EX0
IEN1	Interrupt Enable 1	E8h		x	x	x	x	x	x	x	ET2
IP	Interrupt Priority	B8h		PT2	PPC	PS1	PS0	PT1	PX1	PT0	PX0
IPH	Interrupt Priority High	B7h		PT2H	PPCH	PS1H	PS0H	PT1H	PX1H	PT0H	PX0H
P0	Parallel Port 0	80h		P0.7	P0.6	P0.5	P0.4	P0.3	P0.2	P0.1	P0.0
P1	Parallel Port 1	90h		P1.7	P1.6	P1.5	P1.4	P1.3	P1.2	P1.1	P1.0
P2	Parallel Port 2	A0h		P2.7	P2.6	P2.5	P2.4	P2.3	P2.2	P2.1	P2.0
P3	Parallel Port 3	B0h		P3.7	P3.6	P3.5	P3.4	P3.3	P3.2	P3.1	P3.0
PCON	Power Control	87h		SMOD1	SMOD0	BOF	POF	GF1	GF0	PD	IDL

Name	Description	SFR Addr.	Reset Value	Bit Functions D7	D6	D5	D4	D3	D2	D1	D0
PSW	Processor Status Word	D0h		CY	AC	F0	RS1	RS0	OV	F1	P
RCAP2H	Timer 2 Capture High	CBh									
RCAP2L	Timer 2 Capture Low	CAh									
S0CON	Serial Control	98h	00h	SM0/FE	SM1	SM2	REN	TB8	TB8	TI	RI
S0BUF	Serial Data Buffer	99h	XXh								
S1ADR	Serial 1 Address	DBh	00h	S1A7	S1A6	S1A5	S1A4	S1A3	S1A2	S1A0	GC
S1DAT	Serial 1 Data	DAh	00h								
S1CON	Serial 1 Control	D8h	00h	CR2	ENS1	STA	STO	SI	AA	CR1	CR0
S1STA	Serial 1 Status	D9h	F8h	SC4	SC3	SC2	SC1	SC0	0	0	0
SADDR	Serial Address	A9h									
SADEN	Serial Address Enable	B9h									
SP	Stack Pointer	81h									
TCON	Timer 0 / 1 Control	88h	00h	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
T2CON	Timer 2 Control	C8h	00h	TF2	EXF2	RCK	TCK	EXEN2	TR2	C/T2	CP/RL2
T2MOD	Timer 2 Mode Control	C9h	x0h	x	x	EN2				T2OE	DCEN
TH0	Timer 0 High	8Ch									
TH1	Timer 1 High	8Dh									
TH2	Timer 2 High	CDh									
TL0	Timer 0 Low	8Ah									
TL1	Timer 1 Low	8Bh									
TL2	Timer 2 Low	CCh									
TMOD	Timer 0 / 1 Mode	89h	00h	GATE	C/T	M1	M0	GATE	C/T	M1	M0
WDTRST	Watchdog Timer Reset	A6h	00h								

SFR Address Space

	0 / 8	1 / 9	2 / A	3 / B	4 / C	5 / D	6 / E	7 / F	
F8h		CH	CCAP0H	CCAP1H	CCAP2H	CCAP3H	CCAP4H		FFh
F0h	B								F7h
E8h	IEN1	CL	CCAP0L	CCAP1L	CCAP2L	CCAP3L	CCAP4L		EFh
E0h	ACC								E7h
D8h	S1CON	S1STA	S1DAT	S1ADR					DFh
D0h	PSW								D7h
C8h	T2CON	T2MOD	RCAP2L	RCAP2H	TL2	TH2			CFh
C0h	CCON	CMOD	CCAPM0	CCAPM1	CCAPM2	CCAPM3	CCAPM4		C7h
B8h	IP								BFh
B0h	P3	SADEN						IPH	B7h
A8h	IEN0	SADDR							AFh
A0h	P2		AUXR1				WDTRST		A7h
98h	S0CON	S0BUF							9Fh
90h	P1								97h
88h	TCON	TMOD	TL0	TL1	TH0	TH1	AUXR		8Fh
80h	P0	SP	DPL	DPH				PCON	87h
	0 / 8	1 / 9	2 / A	3 / B	4 / C	5 / D	6 / E	7 / F	

SFR Registers

The architecture of a processor is defined by its registers. In addition to the 256 general purpose scratchpad registers, the TK89C668 has more than 60 special purpose registers in the SFR address space.

CPU Registers

Accumulator (ACC or A)

The accumulator provides the implied operand for many of the opcodes. The accumulator has additional hardware associated with it to detect the zero condition, even parity, and to allow it to be shifted when multiply or divide instructions are being executed.

Processor Status Word (PSW)

The PSW register contains the flags resulting from ALU operations, status bits, and control bits. Here are the bit definitions:

Processor Status Word (PSW)		
Bit	Name	Function
7	CY	Carry Flag
6	AC	Auxiliary Carry. Used in BCD operations.
5	F0	Flag 0. This is a user definable flag.
4	RS1	Register Select 1. Used with RS0 to select the working register bank.
3	RS0	Register Select 0.
2	OV	Overflow Flag
1	F1	Flag 1. This is another user definable flag.
0	P	Parity. This is the even parity of the accumulator contents.

B Register

The B register is used by both the multiply and divide instructions. Prior to the instruction it holds one of the operands, and after the instruction it will hold either the upper 8 bits of the multiply result or the remainder from the divide operation. At other times the B register may be used as a general purpose register.

Stack Pointer (SP)

The stack pointer contains an indirect address that is used to store or recall data associated with the PUSH, POP, CALL, RET, and RETI instructions and with interrupt servicing. The SP register is initialized by reset to a value of 07H. It is incremented prior to storing data, which means that the stack will begin at location 08H.

Data Pointer

The data pointer register is a 16 bit register organized as two 8 bit registers, DPL and DPH. It holds a 16 bit address that is used by the MOVX and MOVC instructions to access the program and data memories.

Auxiliary Register (AUXR)

The PCON register controls the power management functions. It also contains two control bits for the serial port. The PCON register is discussed in section 5.7.

AUXR (8Eh)		
Bit	**Name**	**Function**
7	-	Unused
6	-	Unused
5	-	Unused
4	-	Unused
3	-	Unused
2	-	Unused
1	EXTRAM	All XRAM is external
0	A0	Disable ALE output

AO – When AO = 0, ALE is generated at the internal frequency (6 or 12 clocks) with a 1/3 duty cycle. When A0 = 1, ALE is only active during off-chip memory accesses.

EXTRAM – When EXTRAM = 0, accesses with MOVX are directed to the internal XRAM (up to 8Kbytes). When EXTRAM = 1, all MOVX instructions are to external memory.

Auxiliary Register 1 (AUXR1)

The AUXR1 register controls the dual data pointer flag. IT also contains a general purpose flag.

AUXR1 (A2h)		
Bit	**Name**	**Function**
7	-	Unused
6	-	Unused
5	ENBOOT	Enable Boot Sector
4	-	Unused
3	GF2	General purpose flag
2	0	Always 0
1	-	Unused
0	DPS	Data Pointer Select

DPS – Selects between DPTR0 (0) and DPTR1 as the active Data Pointer.
GF2 – General Purpose user defined flag. Note that bit 2 is a zero allowing the DPS bit to be toggled with an INC AUXR1 instruction.
ENBOOT – Determines if BOOTROM is enabled. This bit is automatically set on the falling edge of RESET when PSEN = 0, ALE = 1, and EA = 1.

I/O Ports

The TK89C668 has 4 8-bit bidirectional I/O ports. These ports are mapped into the SFR address space and may be directly operated on by instructions. The port bits are configured as open-drain bidirectional pins. Writing a 1 to the port allows that bit to be configured as an input.

Port 0

During external RAM and ROM operations, Port 0 serves as a multiplexed address/data bus. During this time, the pins have push-pull capability. The ALE signal serves as a strobe for external logic to demultiplex the Port 0 address from the data. Port 0 is set to FFH during external operations to avoid conflict with incoming data.

Port 0 is located at address 80H.

Port 1

This port also provides the T2 and T2EX inputs for Timer 2 in the 8052 and 8051FB configurations and the ECI and CEXn inputs for the PCA in the 8051FB.

Port 1		
Pin	Name	Alternate Function
P1.0	T2	Timer 2 clock
P1.1	EXF2	Timer 2 gate
P1.2	ECI	PCA Clock
P1.3	CEX0	PCA 0 I/O
P1.4	CEX1	PCA 1 I/O
P1.5	CEX2	PCA 2 I/O
P1.6	SCL	TWI SCL
P1.7	SDA	TWI SDA

Port 1 is located at address 90H.

Port 2

Port 2 provides the upper byte for the address during external ROM and RAM operations.

Port 2 is located at address A0H.

Port 3

Port 3 pins support the serial port, Timer 0, Timer 1, the external interrupt functions, and the read and write strobes for the external data space.

Port 3		
Pin	Name	Alternate Function
P3.0	RXD	Serial Receive Data
P3.1	TXD	Serial Transmit Data
P3.2	/INT0	External Interrupt 0
P3.3	/INT1	External Interrupt 1
P3.4	T0/CEX3	Timer 0 Input/ PCA 3 I/O
P3.5	T1/CEX4	Timer 1 Input/ PCA 4 I/O
P3.6	/WR	Data Write Strobe
P3.7	/RD	Data Read Strobe

Port 3 is located at address B0H.

Read-Modify-Write Instructions

Since each port pin is configured as an open-drain, it is possible for the contents of the port latch to differ from the value at the pin. Instructions that read the port reference the value at the pin. An exception to this is a set of instructions that perform a read-modify-write function. These instructions read the contents of the port latch, modify it, and write it back out. The read-modify-write instructions are:

ANL	JBC
CLR B	MOV B, C
CPL	ORL
DEC	SET B
DJNZ	XRL
INC	

Precharge

Each pin in Port 0 is configured as a bidirectional open-drain when it is used as a port. The pins become push-pull during the multiplexed address/data operation.

The other three ports have an internal pull-up resistor on each pin. In addition, whenever a port makes a 0 to a 1 transition, a precharge occurs for two crystal clock cycles. This precharge improves the rise times of the port pins. The precharge also occurs during the alternate port functions.

Port 2 pins also become push-pull outputs during external memory cycles.

Serial Port

The TK89C668 contains a full duplex serial port. This port is receive buffered and can be operated in any of 4 different modes. The serial port also has the ability to analyze a received word to determine if it is data or an address, and then selectively generate an interrupt depending on which address has been received.

Mode 0

The serial port acts as a synchronous shift register in Mode 0. Data is shifted into and out of the RXD pin. The shift clock appears on the TXD pin.

A write to the SBUF register begins the transmission. Data is shifted out LSB first. Data bits are shifted out once per machine cycle, which is Fosc/12.

Setting the bits REN = 1 and RI = 0 in the SCON register begins a receive cycle.

Mode 1

While in Mode 1, the serial port acts as a UART with 1 start bit, 8 data bits, and 1 stop bit. The baud rate may be different for both the transmit and receive. The baud rate is selected from either the Timer 1 overflow rate or the Timer 2 overflow rate.

The receive stop bit is stored in RB8.

Mode 2

Mode 2 causes the serial port to be a UART with 1 Start Bit, 8 data bits, 1 control bit, and 1 stop bit. The baud rate is either Fosc/32 or Fosc/64, depending on the SMOD bit. The control bit may be used for parity, or it can be used in multiprocessor communications. This is described in the address recognition section.

Mode 3

Mode 3 is the same as mode 2, except that the baud rate is derived from either the Timer 1 or the Timer 2 overflows.

Frame Errors

A frame error is defined as a missing stop bit. Should a frame error occur, the FE bit is set. No other action is taken. The reception of a good frame following a bad one will reset the FE bit.

To read the FE bit, the SCON0 bit in PCON (Bit 6) must be set. This replaces the SM0 bit in SCON (Bit 7) with the FE bit.

Transmission

Transmission is initiated by writing a byte to the SBUF register. The transmitter is not buffered, so caution must be exercised to avoid overwriting the transmitter. The software should wait until the TI bit has been set before writing the next byte.

Interrupts

A serial port interrupt request is generated whenever the RI or the TI bits have been set. The serial port hardware sets the TI bit after a transmit is finished, and it sets the RI bit after a word has been received.

It is up to the programmer to determine which source caused the serial port interrupt, and to clear the bits during the interrupt service.

Address Recognition

The serial port has some special hardware to assist in multiprocessor communications. If the SM2 bit is set, and the serial port is operating in either modes 2 or 3, then the receive interrupt may be suppressed depending on the received bit 8 and the data contents.

If RB8 is a 0, then the byte is assumed to be a data byte, and no interrupt is generated.

If RB8 is a 1, then the received byte is considered to be an address. An interrupt will be generated if the address matches the address in the SADDR register, as modified by the SADEN register. The receive address is compared to the SADDR register on a bit by bit basis. The results of this comparison are enabled by the corresponding bit in the SADEN register. A bit match occurs if either the bits are the same and the enable bit is set, or the enable bit is not set. If a bit match occurs for all 8 bits, then a receive interrupt is generated.

The serial port also supports a broadcast address that is formed by the logical OR of the SADDR and SADEN registers, with 0's being defined as don't cares. A receive interrupt will be generated if the address matches the broadcast address.

Serial Port Control Registers

The serial port is controlled by the SCON register and two of the bits in the PCON register.

S0CON (98h)		
Bit	**Name**	**Function**
7	SM0/FE	Serial Mode Bit 0 or FE Frame Error
6	SM1	Serial Mode Bit 1
5	SM2	Serial Mode Bit 2
4	REN	Receiver Enable
3	TB8	Transmit Bit 8
2	TB8	Receive Bit 8
1	TI	Transmit Interrupt
0	RI	Receive Interrupt

Serial Modes		
SM0 / 1	**Mode**	**Baud Source**
0 0	0	Fosc/12
0 1	1	Variable
1 0	2	Fosc/32, Fosc/64
1 1	3	Variable

PCON (87h)		
Bit	**Name**	**Function**
7	SMOD	Doubles Timer 1 Baud Rate
6	SMOD0	Enables FE Bit in SCON

Serial Port Data Registers

The following registers are data registers associated with the serial port.

S0BUF Register - SFR Address = 99H
SADDR Register - SFR Address = A9H
SADEN Register - SFR Address = B9H

TWI Serial Interface

The TWI Serial Interface is a two-wire communication protocol. Since it is a multi-master interface, continuous arbitration is performed by each transmitting device, by comparing expected output data with actual data being driven onto the Open-Drain outputs.

The interface uses two open-drain signals, SCL and SDA. SCL contains the clock information and SDA the serial data being transferred.

The TWI bus has the following features:

- Two wire synchronized bus transfer between multiple masters and slave.
- Continuous arbitration between transmitting masters and slaves.
- Synchronization and clock speed is controlled allowing variable bit rates on each device.
- Clock may be suspended by holding SCL low during data transmission thus throttling the speed of devices.

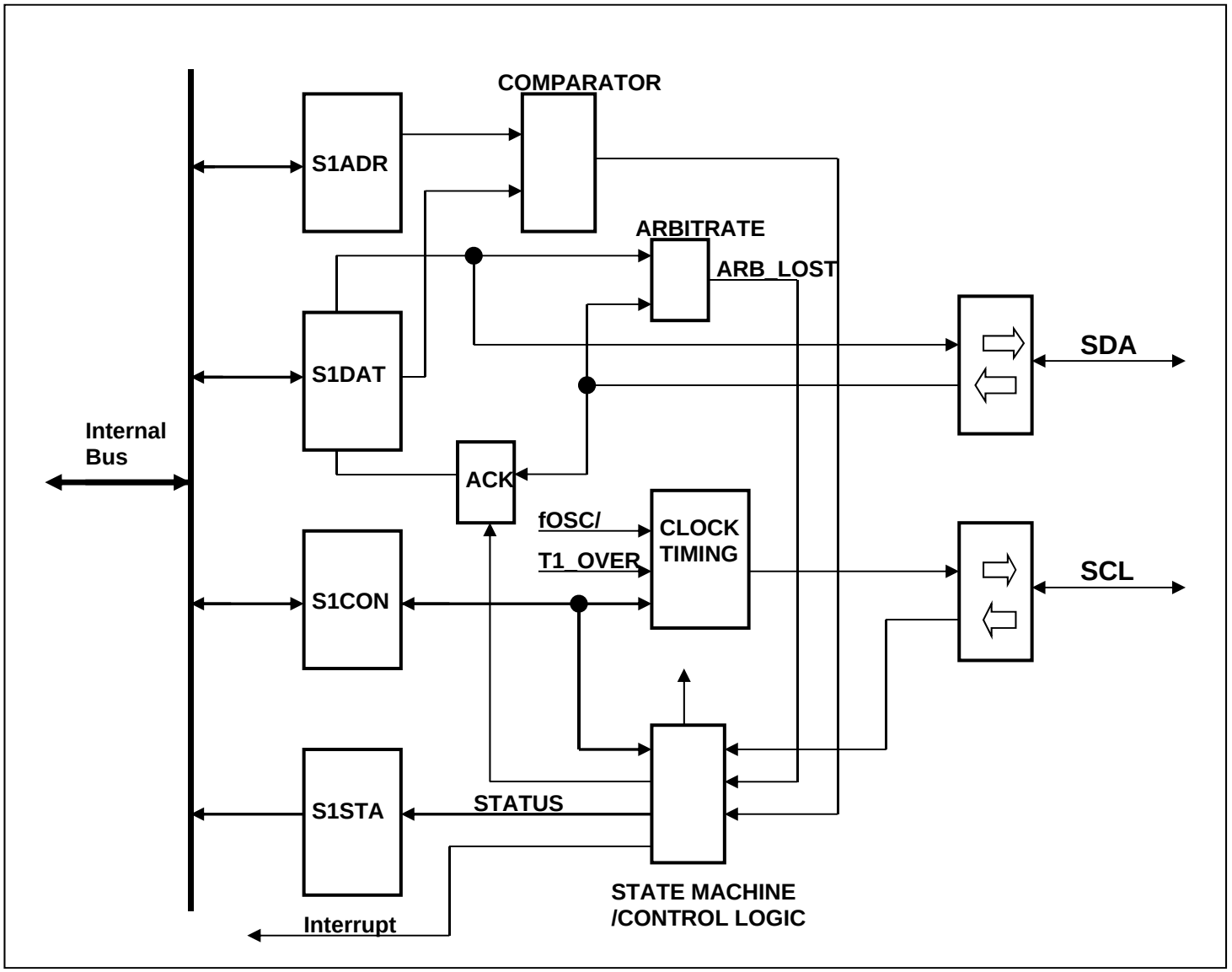


Fig TWI-1 - TWI Serial Interface Logic Block

TWI Interface Control Register

The processor is interfaced with the TWI logic through four SFR registers.

S1DAT data register – SFR = DAh
S1ADR address register – SFR = DBh
S1STA Status Register – SFR = D9h
S1CON Control Register – SFR = D8h

S1DAT Register contains the data to be transmitted or data that was received over the TWI interface.

S1ADR Register contains seven bits of the Address that the TWI logic will respond as a slave. If the LSB is set, the TWI will respond to General Call address.

S1STA Register contains five bits of status information about the TWI controller. The three LSBs are not used and always 0.

S1CON contains the Control bits for the TWI Serial Interface.

S1CON (D8h)		
Bit	**Name**	**Function**
7	CR2	Clock Rate 2
6	ENS1	Enable TWI Serial Port
5	STA	TWI Start
4	STO	TWI Stop
3	SI	TWI Interrupt
2	AA	Assert Acknowledge Flag
1	CR1	Clock Rate 1
0	CR0	Clock Rate 0

CR2-0 – Clock Rate bits:
Those CR2-0 values, that when combined with the fOSC frequency yields a Clock Rate greater than 100KHz, are invalid and should not be used.

When CR2-0 = 0, TH1 is the reload value in Timer 1 when used in Mode 2.

TWI Serial Clock Rates		
CR2-0	**6 Clock Mode f_{osc} divided by**	**12 Clock Mode f_{osc} divided by**
7	128	256
6	112	224
5	96	192
4	80	160
3	480	960
2	60	120
1	30	60
0	48x(CR0 – TH1)	96x(256- TH1)

Clock Rate bits are ignored in Slave Mode as the Master only generates the clock for a data transfer.

ENS1 – Enable TWI Serial Port
When ENS1 is 0, the TWI Interface is disabled and the P1.6 and P1.7 bit may be used as open drain I/O ports.

When ENS1 is 1, the TWI interface is enabled and the P1.6 and P1.7 port bits should be set to 1.

STA – TWI Start Bit
If the TWI bus is free, the STA flag being set will cause a Master bus transaction to begin by generating a START Condition on the Bus. If the bus is not free, it will wait for a STOP condition before generating the Start Condition.

A second STA flag being set during a Master bus transaction will generate a repeated START condition on the bus.

STO – TWI Stop Bit
When STO is set during a Master bus transaction, a STOP condition is generated on the TWI bus. Reception of the STOP condition then resets the STO flag.

In Slave Mode, setting the STO bit will cause a recovery from an error condition but no STOP Condition is transmitted on the bus.

If the STA and STO bits are both set, a STOP condition is transmitted on the bus, then a START condition is transmitted.

SI – TWI Serial Interface Interrupt
An TWI Interrupt is requested when the SI bit is set (assuming it is enabled in the IEN0 register). SI is set by hardware when one of the TWI controller states is active (other than IDLE).

SI must be reset by software.

AA – Assert Acknowledge Bit
If AA bit is set, an ACK will be transmitted when:

- A data byte is received while in Slave Receiver Mode.
- A data byte is received while in the Master Receiver Mode.
- A General Call Address is received when the GC bit is set.
- The controller's own slave address is received.

If the AA bit is cleared, a NACK will be transmitted when:

- Data has been received when in the Master Receiver Mode.
- Data has been received when in the Addressed Slave Receiver mode.

TWI Data Transfer

Two types of data transmission are performed on the TWI interface

- Data transmission from master transmitter to a slave receiver, or
- Data transmission from a slave transmitter to a master receiver.

In the first instance, the master transmitter first broadcasts a slave address byte. The receiving slave device acknowledges at the end of the first transfer. The master transmitter then follows with data bytes and is acknowledged by the slave after each byte is sent.

In the second instance, the master transmitter first broadcasts the slave address byte, and the transmitting slave acknowledges at the end of the first transfer. Data is then transmitted from the responding Slave device to the Master, and the receiving Master then acknowledges each transferred byte. A "Not Acknowledge" bit is transferred on the last byte in order to tell the Slave to stop transmitting.

Operational Modes

Master Transmitter:

The controller first transmits a Start Condition followed by a byte containing 7 bits of address and a 0 in the RW position, thus setting up a Write Transmission to the Slave Address. The Slave device transmits a 0 in the ACK bit position to acknowledge that the address has been recognized.

Following the Slave Address cycle, multiple bytes may be transmitted to the addressed slave each being ACKed until a STOP condition terminates the transmission.

Master Receiver:

The controller first transmits a Start Condition followed by a byte containing 7 bits of address and a "1" in the RW position thus setting up a "READ" cycle to the addressed Slave device. The Slave device transmits a 0 in the ACK bit position to acknowledge that the address has been received.
Following the Slave Address cycle, multiple bytes of data is transmitted on the SDA signal by the Slave device while the Master generates the clock rate with the SCL signal. Each byte is ACKed by the Master device until a STOP condition terminates the transmission.

Slave Receiver:

The controller enters Slave Receiver mode when it recognizes the receipt of its address in the Address cycle posted by a Master device and that a 0 has been recognized in the RW bit position. The Slave acknowledges with an ACK (0). After each following byte is received, and ACK is transmitted until a STOP or Repeated START condition is recognized.

Slave Transmitter:

The controller enters Slave Transmitter mode when it recognizes the receipt of its address with a 1 in the RW bit position. The controller acknowledges with an ACK. Bytes of data is delivered by the Slave device over the SDA signal in response to the Master generating a Clock on the SCL bit. Each byte transmitted by the Slave is ACKed by the Master device until a STOP or Repeated START condition is recognized.

On the last cycle, a "Not Acknowledge" (N) terminates the transfer.

Detailed Operation

Master Transmitter Mode

The Master Transmitter Mode is used to send a number of bytes of data to a Slave device on the TWI bus.

Before Master Transmitter Mode is entered, ENS1 bit (TWI Enable) must be set and the CR[2:0] selects the clock rate.

The Master Transmitter Mode is entered when the CPU sets the STA bit. When the TWI bus is free, a START condition is generated. When the START condition is recognized on the TWI bus, a SI interrupt is generated with a status code (S1STA) of 08h. This status code is used by the Interrupt service routine to load the S1DAT register with the Slave Address and direction bit (SLA+W). The SI bit is then cleared by the Interrupt service routine.

The Slave Address and direction bit are transmitted over the TWI bus and on the ninth clock the acknowledge bit is received. A number of status codes may be generated depending on the ACK bit and whether arbitration was lost. See Figure TWI-2 for a detailed illustration of the state machine Status Codes for the Master Transmitter Mode.

If the acknowledge bit is received as a 0 (ACK), the S1DAT register may be loaded again with the Data Byte to be transmitted, or the transfer may be terminated on a NO ACK (acknowledge bit = 1) or loss of arbitration

Multiple bytes may be transmitted in this manner until all the bytes have been transmitted, a NO ACK is received or loss of arbitration occurs.

A “Loss of Arbitration” occurs when the data bit being transmitted does not match the data bit present on the TWI bus during a Master Transmit operation. A “Loss of Arbitration” during the SLA+RW cycle indicates that a Second Master device is simultaneously trying to use the TWI bus. The Master Device recognizing the lost arbitration will cease data transfer and release the TWI bus.

If more data is to be transmitted to another Slave Device, a repeated START condition may be transmitted thus maintaining control of the TWI bus.

Master Receiver Mode

The Master Receiver Mode is used to receive a number of bytes of data from a Slave device on the TWI bus.

The TWI bus is enabled as in the Master Transmitter Mode generating a START condition. When the START condition is recognized on the bus, the SI interrupt is generated with a status code (S1STA) of 08h. The Interrupt service routine will respond by loading the S1DAT with the slave address and direction bit (SLA+R).

The (SLA+R) bits are transmitted over the TWI bus and on the ninth clock the acknowledge bit is received. A number of status codes may be generated depending on the ACK bit and whether arbitration was lost.

If the acknowledge bit is received as a 0 (ACK), a data receive cycle is begun with data being received over the TWI bus and shifted into the S1DAT register. On the ninth clock, an acknowledge bit is transmitted over the TWI to the Slave device to denote that the data was received by the Master.

The SI interrupt is asserted with the status code present in the S1STA register. The data is then taken from the S1DAT register and the SI bit is cleared. The appropriate action for each code is detailed in Figure TWI-3.

Multiple bytes of data may be received in this manner until the transaction is terminated with a NO ACK or a STOP condition.

Slave Receiver Mode

The Slave Receiver Mode is used to receive a number of bytes of data from a Master Transmitter device on the TWI bus. To respond to the Master Transmitter, the Slave must be initialize with its own Slave Address loaded into the S1ADR register and/or have the GC bit set.

If the upper seven bits of the Address cycle match the address in the S1ADR register, the Slave Receiver will respond with an acknowledgement. Also, if the GC bit is set, the Slave Receiver will respond if the Address cycle is a 00H.

When the S1ADR and S1CON registers are initialized, the Slave Receiver waits until it is addressed with its own Slave Address (or GC address) and the write bit set. The Slave Receiver responds with an acknowledge bit (ACK) and generates a SI interrupt with the corresponding correct status on the S1STA register.

The SI interrupt invokes an interrupt service routine that provides the appropriate action for the status code. See Figure TWI-4 for a detailed illustration of the state machine Status Codes for the Slave Receiver Mode.

The data is then received on subsequent data cycles and each data byte is acknowledged with an ACK. The last data byte able to be received will be acknowledged with a NO ACK.

Slave Transmitter Mode

In Slave Transmitter Mode, a number of data bytes are transmitted to a Master Receive. The data transfer is initialized as in the Slave Receiver Mode. The controller waits until it recognizes its own Slave Address or a General Call address with the RW direction bit set to 1. The Slave Transmitter will respond with an ACK during the acknowledge bit position. The SI interrupt will be set, and the

appropriate status code will be presented to the S1STA register. See Figure TWI-5 for a detailed illustration of the state machine Status Codes for the Slave Transmitter Mode

The status code vectors the CPU to an interrupt service routine that loads the data to be transmitted into the S1DAT register and clears the SI interrupt bit. On the subsequent data cycle, the data is transmitted serially from the S1DAT register, and the acknowledge bit is received from the Master device to terminate the data cycle.

Multiple bytes may be transmitted in this manner. If the AA bit is reset during the transfer, the last byte of data is transferred to the Master device and the SI interrupt is presented with a status code of "C8". The Slave Transmitter ceases to respond to continued byte transfers and the Master Receiver will receive "all ones" on these transfers. While the AA bit is reset, the Slave does not respond to any activity on the TWI again until the AA bit is set.

Miscellaneous

Bus Error

Bus Error state (00h) indicates that a START or STOP condition has occurred in an illegal position in the byte frame. To clear a Bus Error, the STO flag must be set and the SI bit cleared. The controller will enter the IDLE state, clear the STO flag, and cause the SDA and SCL lines to be released.

Simultaneous Repeated START Condition from Two Masters

It is possible that two masters may take the bus and not have a loss of arbitration as they are sending the same address and data information.

If the other master should generate a Repeated START condition before its own Repeated START has been transmitted, the controller will release the bus and no interrupt is generated.

If the other master should generate a STOP condition, the controller will issue a normal START condition and a retry of the entire serial transfer can begin.

Forced Access to TWI BUS

If the TWI bus controller should be hung up due to another uncontrolled device causing incorrect operation, it is possible to clear the controller by setting the STO flag while the STA flag is still set.

The controller acts as if a STOP condition has been received and is able to issue a START condition. The controller then clears the STO flag.

TWI Bus Hung-Up by a Low Level on SCL or SDA

The TWI Bus becomes hung-up if the SDA or SCL signals are pulled low by an uncontrolled device. If the SCL line is pulled low, no further operation can occur and no external hardware can fix the problem.

If the SDA is stuck low, the problem may be resolved by sending additional clock pulses to the offending device over the SCL signal. The controller attempts to send two additional clock pulses until the SDA is released. The normal START condition is transmitted after the SDA has recovered.

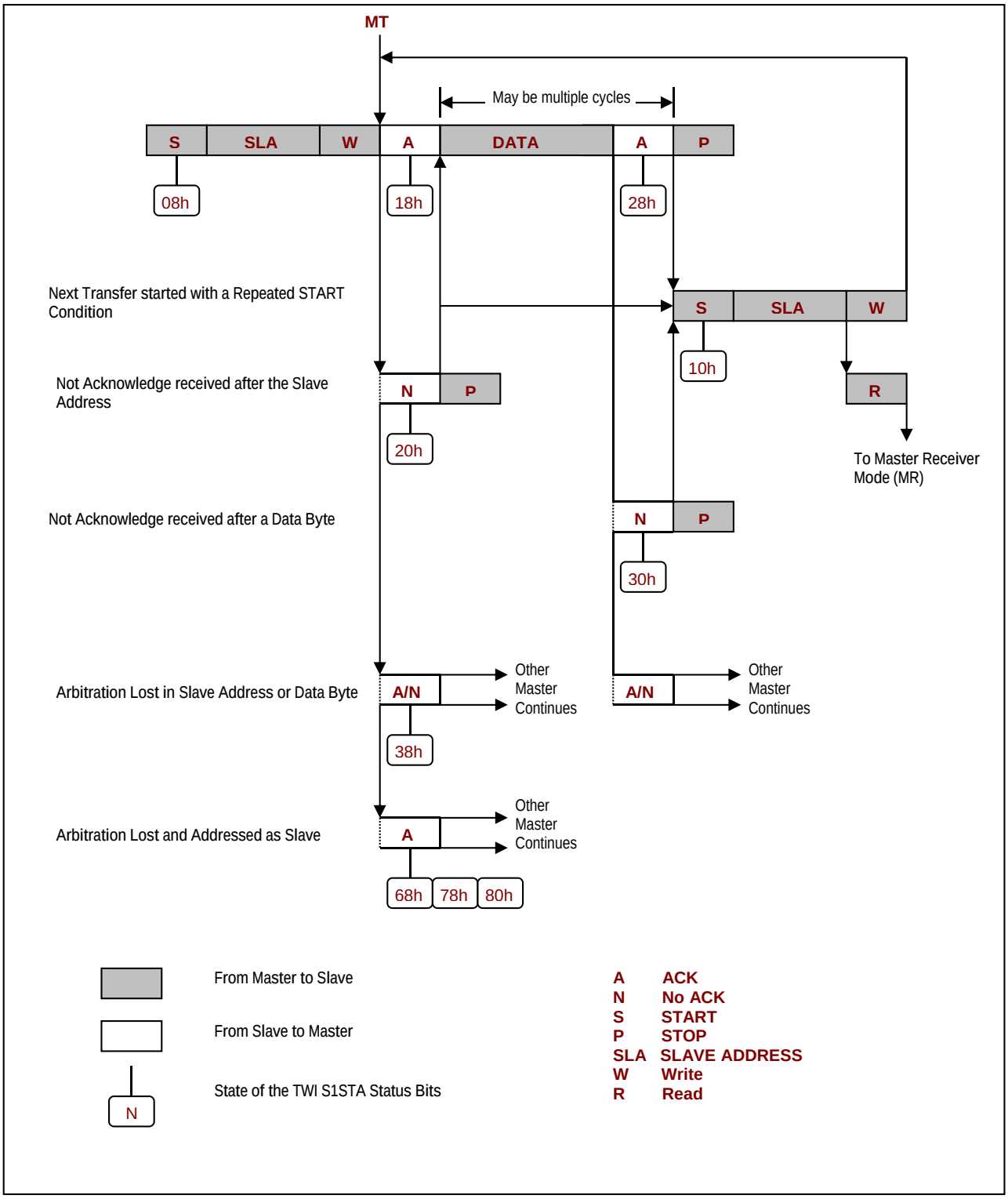


Figure TWI-2 - States of the Master Transmitter Mode

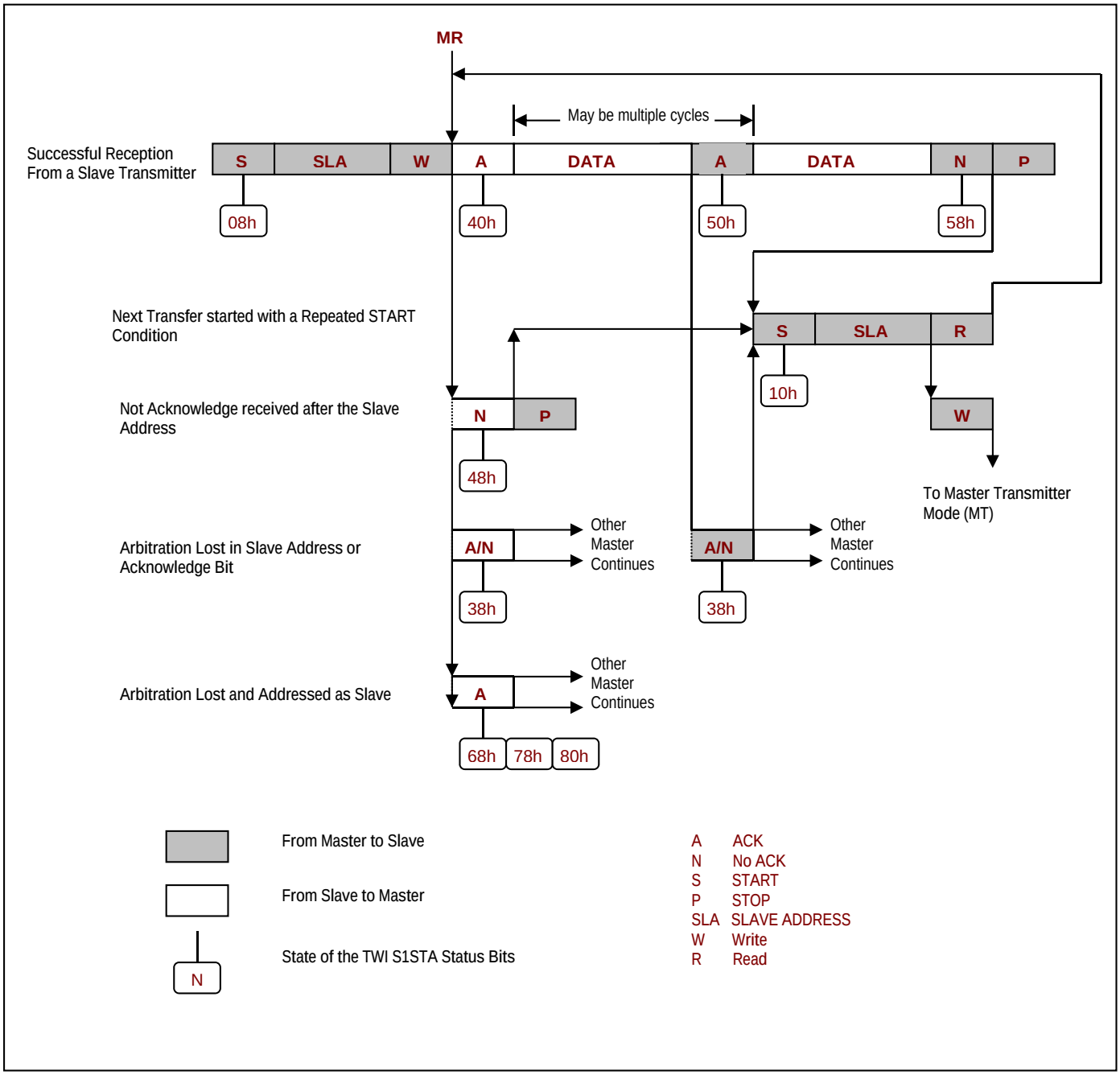


Figure TWI-3 - States of the Master Receiver

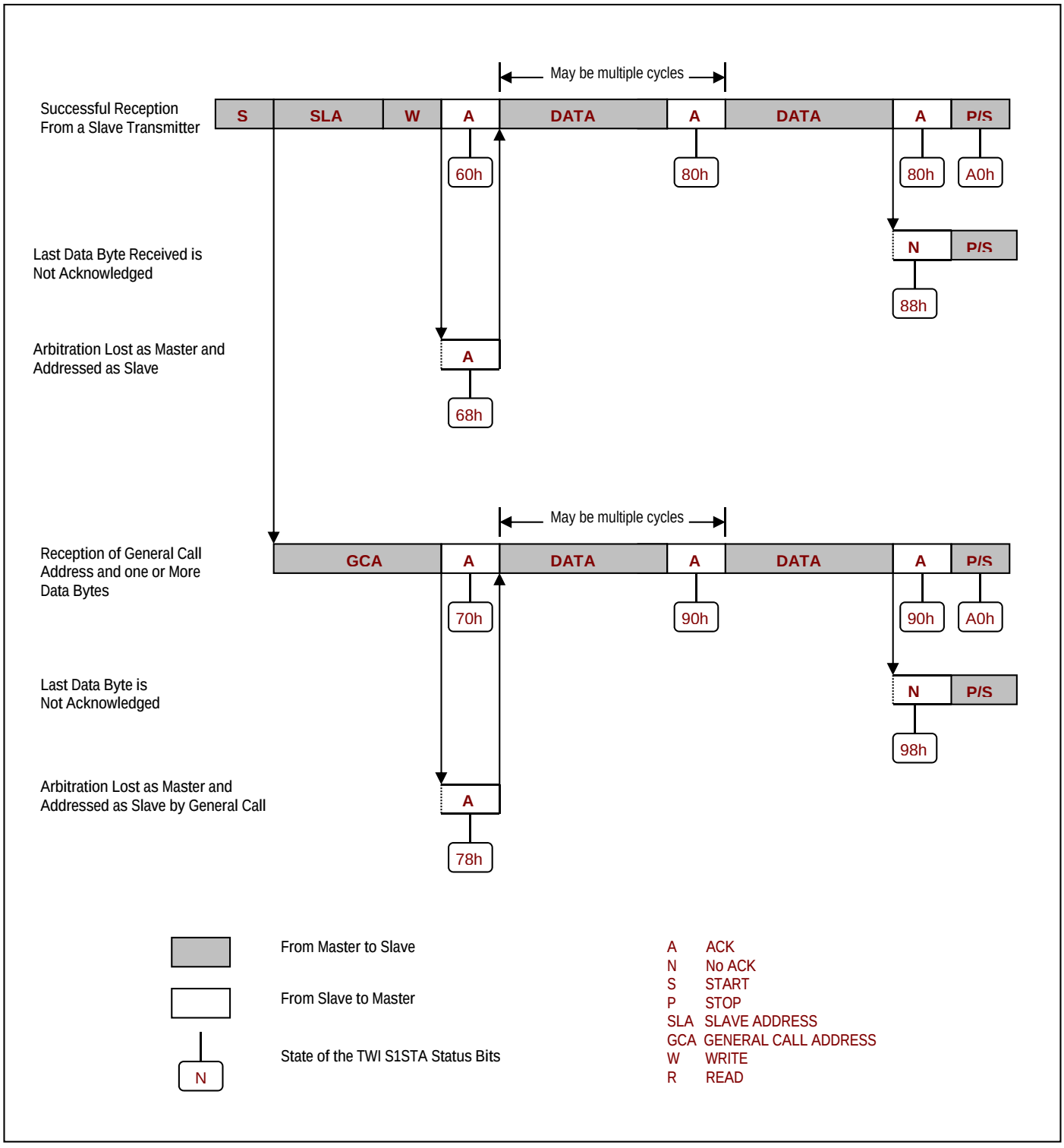


Figure TWI-4 - States of the Slave Receiver Mode

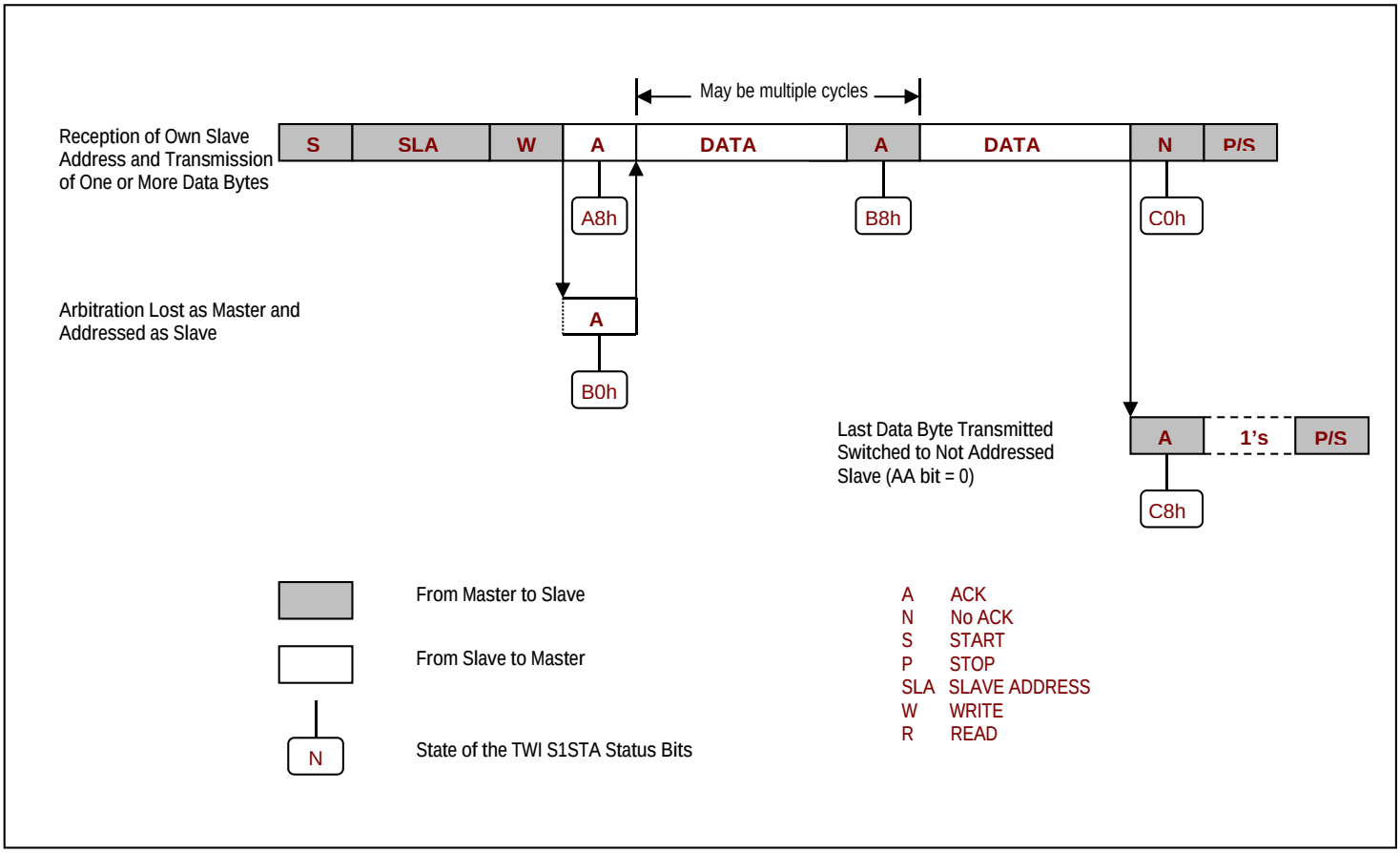


Figure TWI-5 - States of the Slave Transmitter Mode

Table TWI-1. Status Codes and Definitions of TWI Interface Controller

State Code	Status of TWI Bus Interface	Application Software Response					Next Action Taken by Hardware
		To/From S1DAT	To S1CON				
			STA	STO	SI	AA	
Master Transmitter Mode							
08h	Start Condition has been transmitted	Load SLA+W	X	0	0	X	SLA will be transmitted, ACK bit will be received
10h	Repeated Start Condition has been transmitted	Load SLA+W Load SLA+R	X X	0 0	0 0	X X	As above SLA will be transmitted; Mode will be switched to Master Receive
18h	SLA+W has been transmitted; ACK has been received	Load Data Byte or No S1DAT action or No S1DAT action or No S1DAT action	0 1 0 1	0 0 1 1	0 0 0 0	X X X X	Data Byte will be transmitted, ACK will be received Repeated START will be transmitted STOP Condition will be transmitted, STO flag will be cleared STOP Condition followed by a START Condition will be transmitted, STO flag will be cleared
20h	SLA+W has been transmitted; NO ACK has been received	Load Data Byte or No S1DAT action or No S1DAT action or No S1DAT action	0 1 0 1	0 0 1 1	0 0 0 0	X X X X	Data Byte will be transmitted, ACK will be received Repeated START will be transmitted STOP Condition will be transmitted, STO flag will be cleared STOP Condition followed by a START Condition will be transmitted, STO flag will be cleared
28h	Data Byte in S1DAT has been transmitted; ACK has been received	Load Data Byte or No S1DAT action or No S1DAT action or No S1DAT action	0 1 0 1	0 0 1 1	0 0 0 0	X X X X	Data Byte will be transmitted, ACK will be received Repeated START will be transmitted STOP Condition will be transmitted, STO flag will be cleared STOP Condition followed by a START Condition will be transmitted, STO flag will be cleared
30h	Data Byte in S1DAT has been transmitted; NO ACK has been received	Load Data Byte or No S1DAT action or No S1DAT action or	0 1 0	0 0 1	0 0 0	X X X	Data Byte will be transmitted, ACK will be received Repeated START will be transmitted

		No S1DAT action	1	1	0	X	STOP Condition will be transmitted, STO flag will be cleared STOP Condition followed by a START Condition will be transmitted, STO flag will be cleared
38h	Arbitration lost in SLA + R/W or Data Bytes	No S1DAT action or No S1DAT action	0 1	0 0	0 0	X X	TWI bus will be released, not addressed Slave Mode will be entered START condition will be transmitted when bus becomes free
Master Receiver Mode							
08h	START condition has been transmitted	Load SLA+R	X	0	0	X	SLA+R will be transmitted, ACK bit will be received
10h	Repeated START condition has been transmitted	Load SLA+R or Load SLA+W	X X	0 0	0 0	X X	SLA+R will be transmitted Master Transmitter mode will be entered
38h	Arbitration Lost in NO ACK bit	No S1DAT action or No S1DAT action	0 1	0 0	0 0	X X	TWI bus will be released, Slave Mode will be entered START condition will be transmitted when bus becomes free
40h	SLA+R has been transmitted, ACK has been received	No S1DAT action or No S1DAT action	0 0	0 0	0 0	0 1	Data byte will be received; NO ACK bit will be returned Data byte will be received; ACK bit will be returned
48h	SLA+R has been transmitted, NO ACK has been received	No S1DAT action or No S1DAT action or No S1DAT action	1 0 1	0 1 1	0 0 0	X X X	Repeated START condition will be transmitted STOP Condition will be transmitted, STO flag will be cleared STOP Condition followed by a START Condition will be transmitted, STO flag will be cleared
50h	Data Byte has been received, ACK has been returned	Read Data Byte or Read Data Byte	0 0	0 0	0 0	0 1	Data byte will be received; NO ACK bit will be returned Data byte will be received; ACK bit will be returned
58h	Data Byte has been received, NO ACK has been returned	Read Data Byte or Read Data Byte or Read Data Byte	1 0 1	0 1 1	0 0 0	X X X	Repeated START condition will be transmitted STOP Condition will be transmitted, STO flag will be cleared STOP Condition followed by a START Condition will be transmitted, STO flag will be cleared

Slave Receiver Mode							
60h	Own SLA+W has been received, ACK has been returned	No S1DAT action or	X	0	0	0	Data byte will be received, NO ACK will be returned
		No S1DAT action	X	0	0	1	Data byte will be received, ACK will be returned
68h	Arbitration Lost in SLA+R/W as master; Own SLA+W has been received, ACK returned	No S1DAT action or	X	0	0	0	Data byte will be received, NO ACK will be returned
		No S1DAT action	X	0	0	1	Data byte will be received, 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, NO ACK will be returned
		No S1DAT action	X	0	0	1	Data byte will be received, ACK will be returned
78h	Arbitration Lost in SLA+R/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, NO ACK will be returned
		No S1DAT action	X	0	0	1	Data byte will be received, ACK will be returned
80h	Previously address with own slave address, Data has been received, ACK has been returned	Read Data Byte or	X	0	0	0	Data byte will be received, NO ACK will be returned
		Read Data Byte	X	0	0	1	Data byte will be received, ACK will be returned
88h	Previously address with own slave address, Data has been received, NO ACK has been returned	Read Data Byte or	0	0	0	0	Switched to not addressed Slave Mode; No recognition of own SLA or General Call Address
		Read Data Byte or	0	0	0	1	Switched to not addressed Slave Mode, own SLA will be recognized, General Call address recognized if GC = 1
		Read Data Byte or	1	0	0	0	Switched to not addressed Slave Mode, no recognition of own SLA or General Call address. START condition will be transmitted when bus becomes free
		Read Data Byte	1	0	0	1	Switched to no addressed Slave Mode, Own SLA will be recognized, General call address will be recognized if GC = 1. A START condition will be transmitted when the bus becomes free.
90h	Previously address with General Call, Data byte has been received, NO ACK has been returned	Read Data Byte or	X	0	0	0	Data byte will be received, NO ACK will be returned
		Read Data Byte	X	0	0	1	Data byte will be received, ACK will be returned
98h	Previously address with General Call, Data has been	Read Data Byte or	0	0	0	0	Switched to not addressed Slave Mode; No recognition of own SLA or General Call Address

	received, NO ACK has been returned	Read Data Byte or	0	0	0	1	Switched to not addressed Slave Mode, own SLA will be recognized, General Call address recognized if GC = 1
		Read Data Byte or	1	0	0	0	Switched to not addressed Slave Mode, No recognition of own SLA or General Call address. START condition will be transmitted when bus becomes free
		Read Data Byte	1	0	0	1	Switched to not addressed Slave Mode, Own SLA will be recognized, General call address will be recognized if GC = 1. A START condition will be transmitted when the bus becomes free.
A0h	A STOP condition or repeated START condition has been received while still addressed as Slave Receive or Slave Transmitter Mode	No S1DAT action or	0	0	0	0	Switched to not addressed Slave Mode; No recognition of own SLA or General Call Address
		No S1DAT action or	0	0	0	1	Switched to not addressed Slave Mode, own SLA will be recognized, General Call address recognized if GC = 1
		No S1DAT action or	1	0	0	0	Switched to not addressed Slave Mode, No recognition of own SLA or General Call address. START condition will be transmitted when bus becomes free
		No S1DAT action	1	0	0	1	Switched to not addressed Slave Mode, Own SLA will be recognized, General call address will be recognized if GC = 1. A START condition will be transmitted when the bus becomes free.
Slave Transmitter Mode							
A8h	Own SLA+R has been received, ACK has been returned	Load Data Byte or	X	0	0	0	Last data byte will be transmitted and ACK bit will be received
		Load Data Byte	X	0	0	1	Data byte will be transmitted; ACK will be received.
B0h	Arbitration lost in SLA+R/W as master, Own SLA+R has been received, ACK has been returned	Load Data Byte or	X	0	0	0	Last data byte will be transmitted and ACK bit will be received
		Load Data Byte	X	0	0	1	Data byte will be transmitted; ACK 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 will be received.
C0h	Data byte in S1DAT has been transmitted,	No S1DAT action or	0	0	0	0	Switched to not addressed Slave Mode; No recognition of

	NO ACK has been received	No S1DAT action or	0	0	0	1	own SLA or General Call Address Switched to not addressed Slave Mode, own SLA will be recognized, General Call address recognized if GC = 1
		No S1DAT action or	1	0	0	0	Switched to not addressed Slave Mode, no recognition of own SLA or General Call address. START condition will be transmitted when bus becomes free
		No S1DAT action	1	0	0	1	Switched to not addressed Slave Mode, Own SLA will be recognized, General call address will be recognized if GC = 1. A START condition will be transmitted when the bus becomes free.
C8h	Data byte in S1DAT has been transmitted (AA = 0), NO ACK has been received	No S1DAT action or	0	0	0	0	Switched to not addressed Slave Mode; No recognition of own SLA or General Call Address
		No S1DAT action or	0	0	0	1	Switched to not addressed Slave Mode, own SLA will be recognized, General Call address recognized if GC = 1
		No S1DAT action or	1	0	0	0	Switched to not addressed Slave Mode, no recognition of own SLA or General Call address. START condition will be transmitted when bus becomes free
		No S1DAT action	1	0	0	1	Switched to not addressed Slave Mode, Own SLA will be recognized, General call address will be recognized if GC = 1. A START condition will be transmitted when the bus becomes free.
Miscellaneous States							
F8h	IDLE State	No S1DAT action	X	X	X	X	Wait or proceed to next transfer
00h	Bus Error during Master or selected Slave Modes, due to an invalid START or STOP condition. Also an undefined state.	No S1DAT activity	0	1	0	X	Only internal HW is affected in Master or addressed Slave Modes. Bus is released, STO is cleared

Timer 0 and Timer 1

Timer 0 and Timer 1 are two independent timer/counters, but both are configured by the TMOD register and controlled by the TCON register. These registers allow the programmer to select and control the clock source and to operate either timer in any of 4 different operating modes. Since the TCON register address is 88H, it may be operated upon on a bit by bit basis with the microcontroller Boolean instructions.

Clock Control

There is considerable flexibility in the control of the clock for either timer. The C/T bit in the TMOD register selects the source of the clock. Setting it to a one selects the external Tn pin as a clock source, while clearing it to a zero selects the internal clock.

After the source has been selected, it is further controlled by the TRn bit of the TCON register. Clearing this bit blocks the clock and prevents operation. Setting the bit enables the clock.

The clock can also be controlled by the GATEn bit in the TMOD register. Setting this bit allows the external interrupt pin to gate the clock. A 1 on the external interrupt pin enables the clock.

The clock controls act as an enable on the clock generator. This prevents spurious clocks from being generated when the controls are switched.

Clock Speed

When operating off the internal clock source, the timers will be incremented once every machine cycle. Since there are twelve external clocks per machine cycle, the maximum count frequency is Fosc/12.

External pins are sampled once per machine cycle. If the external clock source is selected, it will take two machine cycles to create a complete clock waveform, thus making the maximum external clock frequency Fosc/24. While there are no limitations on the external clock duty cycle, care must be taken to insure that the clock signal has been sampled correctly during the S5P2 time. This is easily accomplished by maintaining the external clock in a given state for a minimum of one machine cycle.

Operating Modes

The timer/counters may be operated in one of four modes. The mode is selected by the two mode bits in the TMOD register. Timer 0 and Timer 1 behave identically in the first three modes.

Mode 0

In Mode 0, the timer operates as a 13 bit counter. The THn register serves as the upper 8 bits, and the TLn register provides the lower 5 bits. The upper three bits of the TLn register will increment, but they have no effect on the counter operation. As the counter rolls over from all ones to all zeros, it will set the TFn flag in the TCON register.

Mode 1

In Mode 1, the timer operates as a 16 bit counter. As in Mode 0, when the counter rolls over from all ones to all zeros, the TFn flag is set.

Mode 2

In Mode 2, the TLn register acts as an 8 bit counter with an automatic reload from the contents of the THn register when the counter rolls over. The TFn flag is set at the reload time.

Mode 3

In Mode 3, Timer 0 functions as two separate 8 bit counters. TL0 is controlled by the Timer 0 control bits. TH0 is controlled by the Timer 1 control bits. When Timer 0 is in Mode 3, Timer 1 may still be operated as a timer or used as a baud rate generator. However, it is not able to set the TF1 bit. Placing Timer 1 in Mode 3 has the same effect as clearing the TR1 bit, which turns Timer 1 off.

Interrupts

The setting of the TFn bit in the TCON register triggers an interrupt request. The TFn bit is reset by hardware when the requested interrupt is acknowledged.

The TCON register also controls the external interrupts. These functions are explained in the interrupt section.

Baud Rate Generation

The overflow of Timer 1 is directly connected to the Serial port as a possible baud rate clock source. This is useful when Timer 0 is being operated in Mode 3 and none of the controls for Timer 1 are available.

The Timer Data Registers

Each timer consists of a lower register (TLn) and an upper register (THn). These registers are treated as any other SFR register and may be read from or written to at any time. These registers are unbuffered and represent the current count value.
The data registers are mapped into the SFR address space at the following locations:

Register	Address
TL0	8AH
TL1	8BH
TH0	8CH
TH1	8DH

The Timer Control Registers

Timer 0 and Timer 1 are controlled by the TMOD and TCON registers. As with the data registers, the control registers may be treated as any other SFR register, and may be read from or written to at any time.

The TMOD register serves as a mode select register. The upper nibble controls Timer 1 and the lower nibble controls Timer 0.

The TCON register controls the enables for each timer, the timer interrupt bits, the edge selects for external interrupts and the external interrupt bits. Interrupt bits generate the interrupt and may be set by software as well as hardware. Here are the bit assignments for the TMOD and TCON registers.

TMOD (89H)		
Bit	**Name**	**Function**
7	GATE1	Timer 1 clock gate enable
6	C/T1	Counter/Timer 1 clock source select
5	M11	Mode select bit 1 for Timer 1
4	M10	Mode select bit 0 for Timer 1
3	GATE0	Timer 0 clock gate enable
2	C/T0	Counter/Timer 0 clock source select
1	M01	Mode select bit 1 for Timer 0
0	M00	Mode select bit 0 for Timer 0

TCON (88H)		
Bit	**Name**	**Function**
7	TF1	Timer 1 interrupt bit
6	TR1	Timer 1 run enable bit
5	TF0	Timer 0 interrupt bit
4	TR0	Timer 0 run enable bit
3	IE1	Interrupt 1
2	IT1	Interrupt 1 edge select
1	IE0	Interrupt 0
0	IT0	Interrupt 0 edge select

Timer 2

Timer 2 is a sixteen bit up/down counter which is configured by the T2MOD register and controlled by the T2CON register. Timer 2 is equipped with a capture/reload capability. As with the Timer 0 and Timer 1 counters, there exists considerable flexibility in selecting and controlling the clock and in defining the operating mode.

Clock Control

The clock source for Timer 2 may be selected from either the external T2 pin (C/T2=1) or the crystal oscillator (C/T2=0). The clock is then enabled when TR2 is a one, and disabled when TR2 is a zero.

Clock Speed

The external clock is sampled once each machine cycle. This puts the maximum speed of the internal clock at Fosc/24. The designer must insure that the external clock signal is valid during the S5P2 sampling time.

During the capture and auto-reload modes, the timer is clocked once per machine cycle, which represents a clock frequency of Fosc/12. During the baud mode, the timer is clocked once per internal clock, which represents a clock frequency of Fosc/2.

Operating Modes

Timer 2 has four modes of operation. These are the capture mode, an auto-reload mode with DCEN=0, an auto-reload mode with an up/down count capability, and a baud mode.

Capture Mode

The capture mode is enabled by setting the CP/RL2 bit in the T2CON register to a 1. In the capture mode, Timer 2 serves as a 16 bit up counter. When the counter rolls over from all 1's to all 0's, the TF2 bit is set, which will generate an interrupt request.

If the EXEN2 bit is set, then a negative transition of the T2EX pin will cause the value in the TL2 and TH2 registers to be captured by the RCAP2L and RCAP2H registers. This action also causes the EXF2 bit in T2CON to be set, which will also generate an interrupt.

Auto-reload Mode, Counting Up

The auto-reload mode as an up counter is enabled by clearing the CP/RL2 bit in T2CON and clearing the DCEN bit in T2MOD. In this mode, Timer 2 is a 16 bit up counter. When the counter rolls over from all 1's, a reload is generated that causes the contents of the RCAP2L and RCAP2H registers to be loaded into the TL2 and TH2 registers. The reload action also sets the TF2 bit.

If the EXEN2 bit is set, then a negative transition of the T2EX pin will also cause a reload. This action also sets the EXF2 bit in T2CON.

Auto-reload Mode, Counting Up/Down

Timer 2 will be in an auto-reload mode as an up/down counter if the CP/RL2 bit in T2CON is cleared and the DCEN bit in T2MOD is set. In this mode, Timer 2 is an up/down counter whose direction is controlled by the T2EX PIN. A one on this pin causes the counter to count up.

An overflow while counting up will cause the counter to be reloaded with the contents of the capture registers. The next down count following the case where the contents of Timer 2 equal the capture registers will load an 0FFFFH into Timer 2. In either event a reload will set the TF2 bit. A reload will also toggle the EXF2 bit. However, the EXF2 bit cannot generate an interrupt request while in this mode.

Baud Rate Mode

The baud rate mode is enabled by setting either the RCLK or TCLK bits in the T2CON register. While in the baud rate mode, Timer 2 is a 16 bit counter with auto reload when the count rolls over from all 1's. However, rolling over does not set the TF2 bit.

If the EXEN2 bit is set, then a negative transition of the T2EX pin will set the EXF2 bit in T2CON and cause an interrupt request.

Interrupts

Interrupt requests for Timer 2 are generated from the TF2 bit and the EXF2 bit, depending on the operating mode. Unlike Timers 0 and 1, these bits are not reset by the interrupt and must be cleared by software. The Timer 2 interrupt service routine is located at address 002BH.

Timer 2 Data Registers

There are 4 data registers associated with Timer 2. TL2 and TH2 are the lower and upper bytes of Timer 2. RCAP2L and RCAP2H are the lower and upper capture and reload registers for the timer. The data registers are mapped into the SFR address space at the following locations.

Register	Address
RCAP2L	CAH
RCAP2H	CBH
TL2	CCH
TH2	CDH

The timer clocks are skewed with respect to the processor cycles, so they can be read or written to at any time without problem. However, when Timer 2 is operated in the baud mode and using the divide-by-two clock, the timer contents will be changing at high speed. This will interfere with their being correctly operated upon by read-modify-write instructions.

Timer 2 Control Registers

The T2MOD and T2CON registers control Timer 2. As with the data registers, these registers may be written to and read from at any time without conflict with the timer operation.

T2MOD (C9h)		
Bit	**Name**	**Function**
7		Unused
6		Unused
5		Unused
4		Unused
3		Unused
2		Unused
1	T2OE	Timer 2 Output Enable
0	DCEN	Down Count Enable. A 1 enables the Up/Down capability in the auto-reload mode

T2CON (C8h)		
Bit	**Name**	**Function**
7	TF2	Timer 2 Flag
6	EXF2	Timer 2 External Flag
5	RCLK	Receive clock enable
4	TCLK	Transmit clock enable
3	EXEN2	External Enable
2	TR2	Timer 2 Run Enable
1	C/T2	Counter/Timer select
0	CP/RL2	Capture/Reload select

Programmable Counter Array (PCA)

The TK89C668 contains a programmable counter array. This array consists of a 16 bit counter, five compare modules, and control logic. The counter may be programmed to operate off a choice of four different sources. Each compare module may in turn be programmed to operate in one of five different modes. The operating modes may generate interrupts through a dedicated interrupt channel.

Figure PCA-1 shows the general organization of the PCA.

Control Registers

The PCA uses seven control registers. Each of the five modules has its own control register, which are called the CCAPMn registers. There are two additional registers, the CCON and CMOD registers, which control the 16 bit counter and provide overall control and mode functions. Here are the control registers and their bit assignments.

CMOD (C1h)		
Bit	**Name**	**Function**
7	CIDL	Counter Idle Control
6	WDTE	Watchdog Timer Enable
5		
4		
3		
2	CPS1	Count Pulse Select 1
1	CPS0	Count Pulse Select 0
0	ECF	Enable Counter Overflow Interrupt

CCON (C0h)		
Bit	**Name**	**Function**
7	CF	Ctr. Overflow Flag
6	CR	Counter Run Enable
5		Not Used
4	CCF4	Mod 4 Interrupt Flag
3	CCF3	Mod 3 Interrupt Flag
2	CCF2	Mod 2 Interrupt Flag
1	CCF1	Mod 1 Interrupt Flag
0	CCF0	Mod 0 Interrupt Flag

CCAPMn (C2H-C6H)		
Bit	**Name**	**Function**
7		Not used
6	ECOMn	Enable Comparator
5	CAPPn	Capture Positive
4	CAPNn	Capture Negative
3	MATn	Match
2	TOGn	Toggle
1	PWMn	Pulse Width Modulation
0	ECCFn	Enable CCF Interrupt

Data Registers

All of the counters and compare modules are accessible as SFR addresses.

Register	Address	Function
CL	E9H	Lower Counter
CH	F9H	Upper Counter
CCAP0L	EAH	Lower Module 0
CCAP0H	FAH	Upper Counter 0
CCAP1L	EBH	Lower Module 1
CCAP1H	FBH	Upper Counter 1
CCAP2L	ECH	Lower Module 2
CCAP2H	FCH	Upper Counter 2
CCAP3L	EDH	Lower Module 3
CCAP3H	FDH	Upper Counter 3
CCAP4L	EEH	Lower Module 4
CCAP4H	FEH	Upper Counter 4

Sixteen Bit Counter

The PCA is built around a central 16 bit counter. This counter provides the time information to each of the 5 compare modules. The lower 8 bits are in the CL register and the upper 8 bits are in the CH register. These registers may be read and written to at any time. As with any running counter, care should be taken when writing to the CL and CH registers to prevent false decodes.

The clock source may be selected from one of four different sources by the CPS1 and CPS0 bits in the CMOD register. A 00 selects the internal machine cycle clock, which is Fosc/12. A 01 selects one half of the internal clock, which is Fosc/4. A 10 selects Timer 0 overflow as the clock. A 11 selects an external clock from the P12 pin. The maximum rate for this clock will be FOSC/8. Unlike other SFR pins which are sampled every Fosc/12, this pin is sampled at a FOSC/4 rate.

The clock for the counter is gated by the CR bit in the CCON register. The clock may also be gated by the processor being in the idle mode and the CIDL bit being set.

When the counter overflows, it sets the CF bit in the CCON register. Depending on the status of the ECF bit in the CMOD register, this overflow can trigger an interrupt request.

Figure PCA-2 shows the counter clock and interrupt circuit.

Capture Modules

A capture module consists of a 16 bit compare register, a 16 bit comparator, and a control register. A match out of the comparator sets the CCFn flag, which in turn can be used to generate an interrupt. A match can also be brought out of the chip through one of the CEXn pins.

Capture Mode

Bits 4 or 5 of the CCAPMn register, with bits 1, 2, 3, and 6 = 0 enable a capture mode. In this mode, a rising (Bit 5) or falling (Bit 4) edge on the CEXn pin creates a capture signal. The capture signal transfers the contents of the 16 bit counter into the CCAPnH and CCAPnL registers.

The capture signal will set the CCFn bit in the CCON register. Enabling the ECCFn bit will allow the CCFn bit to trigger an interrupt.

Figure PCA-3 shows the configuration of the capture mode.

Software Timer Mode

A 16 bit software timer with an optional output may be created by clearing bits 1, 4, and 5, and controlling the ECOMn (Bit 6), MATn (Bit 3), TOGn (Bit 2), and ECCFn (Bit 0).

Setting the ECOMn bit enables the 16 bit comparator. When the count equals the value in the CCAPn registers, an equal signal is generated. Setting the MATn bit will allow this signal to set the CCFn bit in the CCON register. And setting the ECCFn bit will allow the CCFn bit to generate an interrupt.

The TOGn bit enables the equal signal to clock a toggle flip-flop, whose output is visible on the CEXn pin.

Writing to the CCAPnL register clears the ECOMn bit, thus disabling the comparator. Writing to the CCAPnH register sets the ECOMn bit, re-enabling the comparator. This is done to allow the programmer to make changes to the CCAPn register without causing output glitches.

Figure PCA-4 shows the configuration of the software timer mode.

Pulse Width Modulation (PWM) Mode

Clearing bits 0, 2, 3, 4, and 5 of the CCAPMn register and setting the PWMn and ECOMn bits enable the PWM mode. In this mode, the CEXn pin is a 1 when the CL count is greater than or equal to the value in the CCAPn registers, and CEXn is zero when the CL count is less than the CCAPnL value. When the CL register rolls over, the value in the CCAPnH register is transferred into the CCAPnL register. This allows for the programmer to change the value for the PWM signal without causing a glitch.

Programming the CCAPnH register with numbers ranging from 00H to FFH causes the output duty cycle to range from 100 percent to 0.4 percent.

Figure PCA-5 shows the configuration of the PWM mode.

Watchdog Timer Mode

Setting bits 1, 4, and 5 to zero and bit 3 to a one in the CCON4 register and setting the WDTE bit in the CMOD register enables the watchdog timer mode in module 4. The timer is controlled through the ECOM4 bit. When an equal between the counter bus and the contents of the CCAP4 register is detected, the watchdog timer resets the processor. To prevent the watchdog timer from resetting the processor, it is the responsibility of the programmer to insure that the equal is never generated. This is easily done by changing the values in the CCAP4 registers or by changing the values in the CH/CL registers. Changing the WDTE bit is not recommended, since the processor could conceivably fail while the timer was disabled.

Figure PCA-6 shows the configuration of the watchdog timer mode.

PCA Interrupts

The PCA has six interrupt sources. Each module has one, and the counter overflow is the sixth. These interrupt requests are OR'ed together to create the PCA interrupt request. The interrupt routine must

determine the source of the interrupt and clear it before returning from the interrupt.

PCA Block Diagram

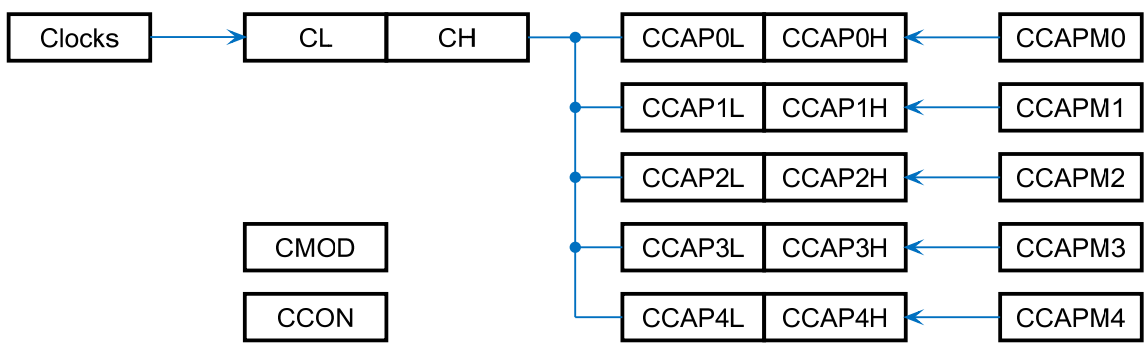


Figure PCA-1

PCA Clock Input Circuit

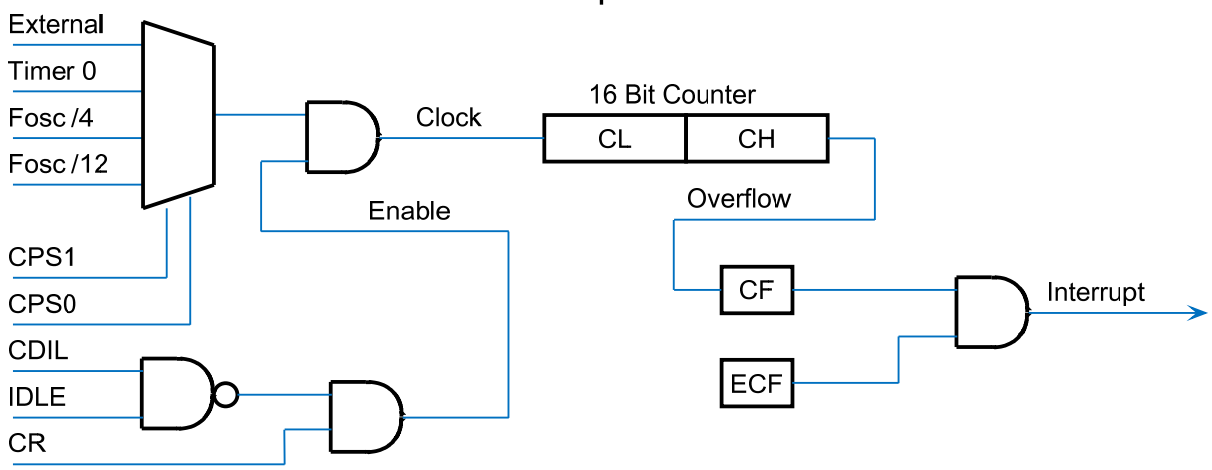


Figure PCA-2

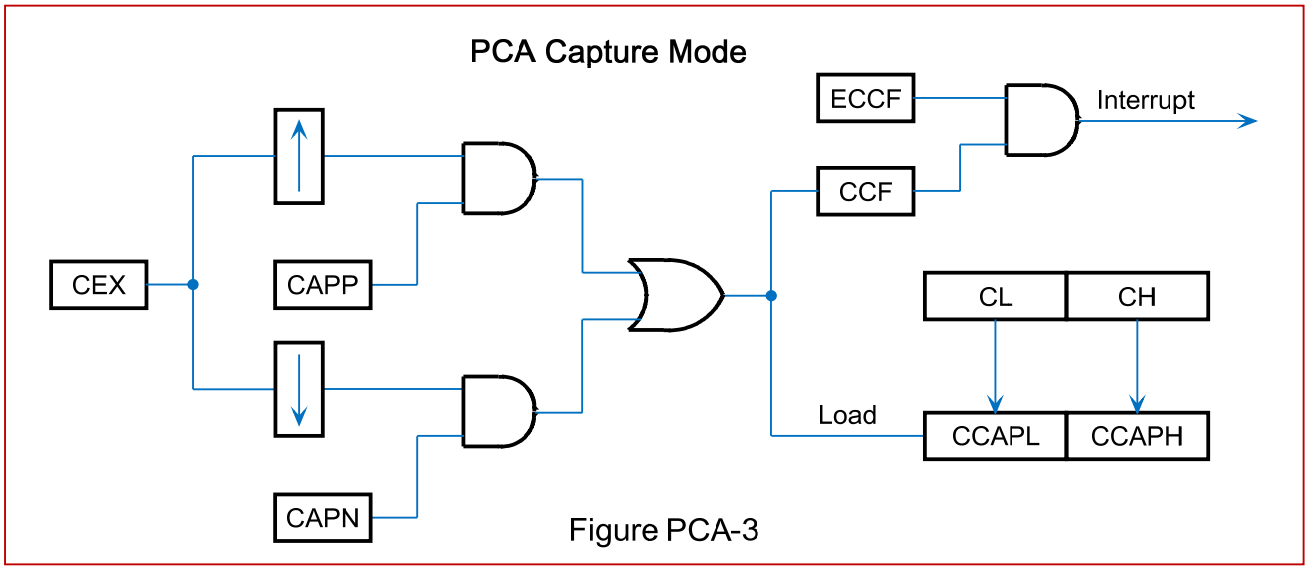


Figure PCA-3

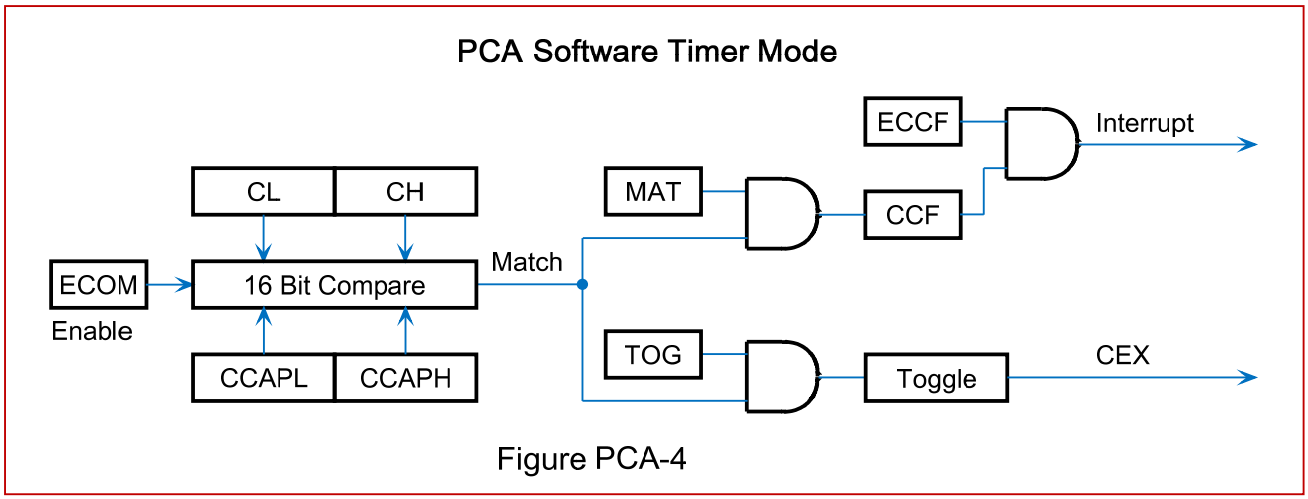


Figure PCA-4

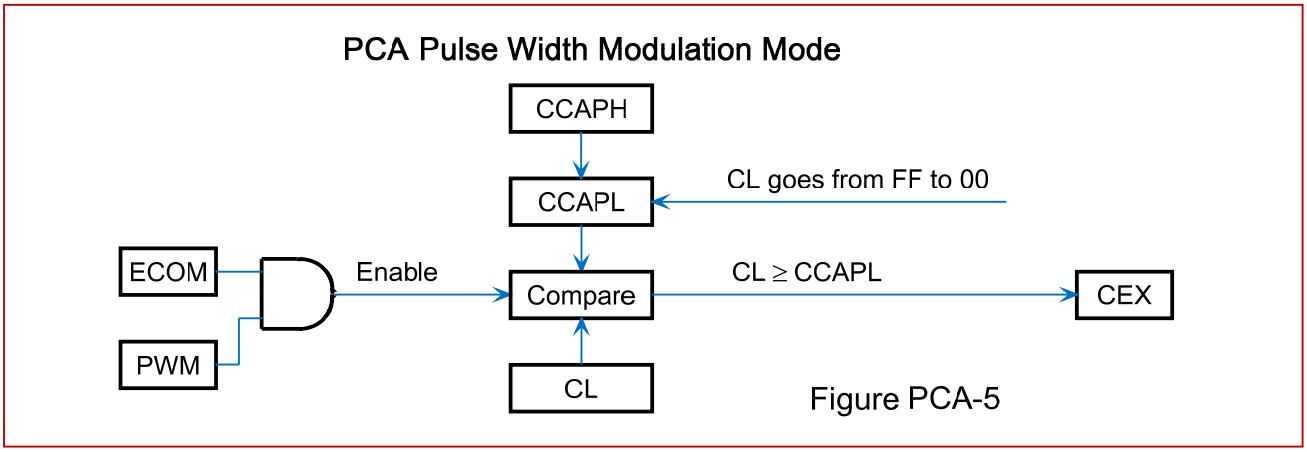


Figure PCA-5

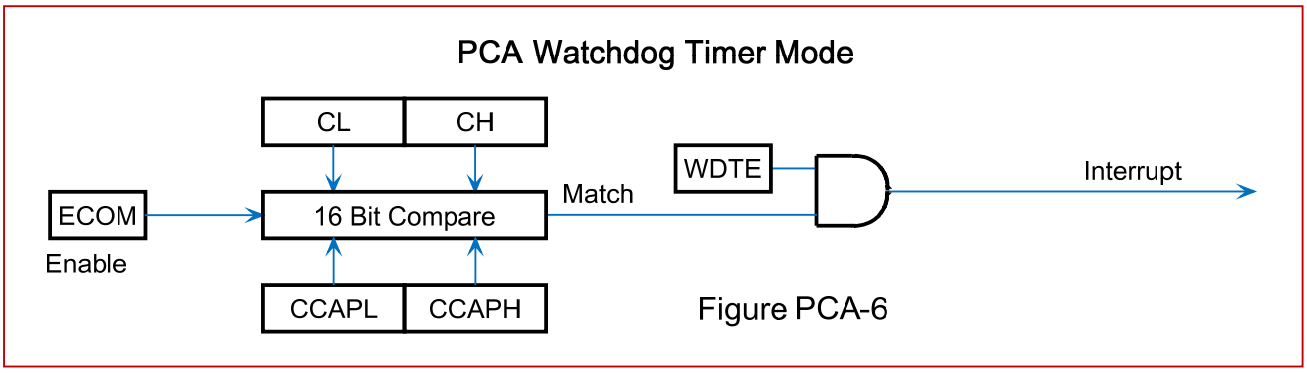


Figure PCA-6

Interrupt Controller

The TK89C668 has eight interrupt channels. These channels may be individually or collectively enabled, and each may be assigned one of four priority levels. The interrupt channels are dedicated to specific functions within the TK89C668 and are controlled by bits in the control registers.

An interrupt may be generated by software by setting the appropriate control bits.

External Interrupts

The TK89C668 has two external interrupts, which are called /INT0 and /INT1. The IT0 and IT1 bits in the TCON register determine if these interrupts are level triggered (ITn=0) or falling edge triggered (ITn=1). The interrupt bit will be reset by the interrupt hardware if it was an edge triggered interrupt. It is the programmer's responsibility to insure that the external event that generates a level triggered interrupt is satisfied and that the input is removed before the end of the interrupt service routine.

Timer 0, 1 Interrupts

Overflows in Timer 0 and 1 set the TFn bits in the TCON register. They in turn generate the interrupt. As with the external interrupts, the interrupt hardware will reset the interrupt bit during the interrupt routine.

Serial Port Interrupts

A serial port interrupt will be generated if either the RI or the TI bits in the SCON register have been set. It is the programmer's responsibility to determine whether the transmitter or the receiver generated the interrupt. Also, the interrupt bits must be cleared by software.

Timer 2 Interrupts

Both the EXF2 and the TF2 bit in the T2CON register will generate a Timer 2 interrupt. As with the serial port, the programmer has the responsibility to determine who caused the interrupt and to reset the interrupt bits.

The EXF2 bit will not generate an interrupt if the timer is in the auto-reload mode and DCEN is 1.

PCA Interrupts

Either the CF counter overflow bit or any of the five CCFn bits can generate a PCA interrupt request

Brown Out Interrupts

A Brown-Out interrupt will be generated if the BOF bit in the PCON register is set.

Interrupt Enables

Each interrupt channel may be individually enabled by setting the appropriate bit in the interrupt enable (IE) register. The EA (Enable All) bit must also be set to enable an interrupt.

IEN0 (A8h)		
Bit	**Name**	**Function**
7	EA	Enable All
6	EC	Enable PCA
5	ES1	Enable TWI
4	ES0	Enable Serial
3	ET1	Enable Timer 1
2	EX1	Enable External 1
1	ET0	Enable Timer 0
0	EX0	Enable External 0

IEN1 (E8h)		
Bit	**Name**	**Function**
7		
6		
5		
4		
3		
2		
1		
0	ET2	Enable Timer 2

Interrupt Priorities

Each interrupt channel is assigned a bit in the interrupt priority (IP) register. Setting the bit increases the interrupt priority level. Within the priority level, there exists a priority sequence. The priority level takes precedence over the priority sequence. These are listed along with the vector locations below.

IP (B8h)		
Bit	**Name**	**Function**
7	PT2	Timer 2
6	PPC	PCA Low Priority
5	PS1	TWI
4	PS0	Serial
3	PT1	Timer 1
2	PX1	External 1
1	PT0	Timer 0
0	PX0	External 0

IPH (B7h)		
Bit	**Name**	**Function**
7	PT2H	Timer 2
6	PPCH	PCA Low Priority
5	PS1H	TWI
4	PS0H	Serial
3	PT1H	Timer 1
2	PX1H	External 1
1	PT0H	Timer 0
0	PX0H	External 0

Interrupt Vector Locations		
Address	**Priority**	**Source**
0003h	1	IE0
000Bh	3	IT0
0013h	4	IE1
001Bh	5	TF1
0023h	6	RI + TI
002Bh	2	TWI
0033h	8	CF + CCFn
003Bh	7	TF2 + EXF2

Interrupt Response

An interrupt begins with the setting of the appropriate bit in a control register. The interrupt hardware compares the bit with the enables and priorities to determine if an interrupt request is warranted and which interrupt should be requested. This logic provides either a request for a priority 1 or a priority 0 interrupt. This logic also prepares a vector address for the interrupt service routine.

If the processor is at the end of an instruction, and if the current instruction is not a RETI, and the current instruction does not involve a write to either the IP or IE registers, and the processor is not currently servicing an interrupt of equal or higher priority, then the interrupt request will be granted, and the processor will execute the interrupt service routine. Under normal operation, the program counter is incremented immediately after an opcode fetch. This action is inhibited by the interrupt, and the program counter is frozen at the current value. The interrupt service routine then pushes the program counter onto the stack, sets an internal interrupt status bit, clears the upper byte of the program counter, and loads the lower byte with the interrupt vector. Overall, the interrupt service routine behaves as a subroutine call.

Interrupt Return

The RETI instruction should be used for a return from a subroutine. This instruction is the same as a RET instruction, except that it clears the internal interrupt status bit, and thus enables future interrupts.

Power Management

The TK89C668 provides for the two standard power management modes and for the POR status bit. The ASIC nature of the design also allows the superior method of simply stopping the clock. This method was not available in the original 8051 design due to the use of dynamic logic.

PCON (98H)		
Bit	**Name**	**Function**
7	SMOD1	Serial Port Mode 1
6	SMOD0	Serial Port Mode 0
5	-	Unused
4	POF	Power On Flag
3	GF1	General Purpose Flag 1
2	GF0	General Purpose Flag 0
1	IDLE	Idle Mode
0	PD	Power Down

SMOD1

SMOD1 is a mode control bit for the serial port.

SMOD0

SMOD0 is another mode control bit for the serial port.

Power On Flag

Bit 4 of the PCON register contains a power-on-flag. This bit is set when VDD has been applied to the part. If it is subsequently reset by software, it can be used to determine if a reset is a warm boot or a cold boot.

GF1

GF1 is a general purpose flag bit that cna be used in customer applications.

GF0

GF0 is a general purpose flag bit that cna be used in customer applications.

Idle Mode

The idle mode is entered by setting the IDLE bit in the PCON register. In the idle mode, the internal clock to the processor is stopped. The peripherals and the interrupt logic continue to be clocked. The processor will leave idle when either an interrupt or a reset occurs.

Power Down Mode

When the PD bit of the PCON register is set, the processor enters the power down mode. During this mode all of the clocks, including the oscillator are stopped. The only way to exit power down mode is with a reset.

Stopping The Clock

The clock for the TK89C668 may be stopped externally at any time and in any state. It may then be resumed at any time without interference with the processor operation. The only consideration is that the external clock stopping logic should not cause glitches on the clock input.

Expanded Data RAM Addressing

The TK89c668 has four segments of data memory:

Lower 128 bytes of local RAM, Upper 128 bytes of local RAM, and 7936Bytes of Expanded RAM (XRAM).

The lower 128 bytes can be accessed by either direct or indirect addressing. Since the space is shared with the SFR registers, the upper 128 bytes may be accessed by indirect addressing only.

The 7936Bytes of Expanded RAM are indirectly accessed with the DPTR or Ri registers by the MOVX instruction with the EXTRAM bit cleared.

A MOVX instruction above the maximum XRAM address will generate an indirect access to external memory with the P3.6(WR_) and P3.7(RD_) signals at the address pointed to by the DPTR register.

With the EXTRAM bit set to 1, MOVX @Ri and MOVX @DPTR operate the same as the standard 8051. All MOVX accesses are to external memory with the low order address byte appearing on the P0 port and the high order address byte on the P2 port.

Hardware Watchdog Timer

The Watch Dog Timer (WDT) is intended as method to recover from situations in which the software becomes unstable. The WDT is a 14-bit counter that is accessed through a SFR address. The WDT is disabled at reset and is enabled when the CPU writes a 1EH and E1H sequence to the WDTRST address (0A6H).

After the WDT is enabled, the CPU needs to write the same sequence as above in order to clear the timer before an overflow occurs. The 14-bit counter will overflow when the count reaches (3FFFh) and reset the device by generating a pulse on the RESET pin of 98-clock cycles width (196 in 12 clock mode).

Flash ROM Memory

The TK89C668 contains a 64Kbyte Flash ROM memory that is used to provide program or data storage for the user software. It is organized as 5 separate blocks of memory. The first two blocks are 8Kbytes, and the last three blocks are 16Kbytes in size.

The Flash ROM may be written or erased using two methods, In-Application Programming or In-System Programming. The Flash ROM ISP and IAP interfaces have the following features available:

- Chip Erase – features full chip erase.
- Block Erase – features ability to erase selected blocks.
- Security bits – features four levels of program security.
- Program Data Byte – features individual byte programming.

Boot ROM

When the TK89C668 programs its own Flash memory, it performs this function using an internal 2Kbyte Boot ROM. The Boot ROM overlays the program memory space at F800h-FFFFh when it is enabled.

Power-On Reset Code Execution

The TK89C668 contains two special registers located in Flash memory, the STATUS BYTE and BOOT VECTOR. At the falling edge of reset, if the contents of the STATUS BYTE are non-zero, the content of the BOOT VECTOR is used as the upper byte of the execution address (low byte = 00h) for a custom boot loader written by the user. If the STATUS BYTE is 00h, power-up execution begins at location 0000h.

Hardware Activation of Boot Loader

The boot loader may also be activated by setting PSEN low, ALE high, EA high, P2.7 and P2.6 High at the falling edge of RESET. This will cause immediate execution of the boot loader code at the location set in the BOOT VECTOR (factory default = 0FCh).

Security Bits

The Security Bits prevent the contents of the Flash ROM from being read. The Security bits are located in Flash and will provide multiple levels of protection of the user's code.

Security Bits		
Level	**Bits**	**Description**
1	000	MOVC instructions executed from external program memory are disabled from fetching code bytes from internal memory
2	100	Same as Level 1 plus block erase is disabled. Erase or programming of Status Byte or Boot Vector is disabled.
3	110	Same as Level 2, plus verify of code memory is disabled
4	111	Same as Level 3, plus external execution is disabled

In-System Programming (ISP)

In-System Programming feature allows the user to program the internal Flash ROM through the Serial Port with a minimum of hardware resources. The Boot ROM contains an embedded application that performs this task.

The ISP function uses four signals to drive the TK89C668 device to program the Flash ROM: RxD, TxD, VSS and VDD.

NOTE: No signal should provide a voltage greater than VDD on any pin, as it will damage the device.

See Figure ISP-1 for a diagram of In-System Programming with a minimum of pins.

Using In-System Programming

The ISP feature contains an autobaud rate calculation that allows it to be used on a wide range of baud rates and oscillator frequencies. The user ISP program must first send an uppercase U character (55h) to the device in order to establish the baud rate.

Once the baud rate has been determined, the ISP firmware on the device will only receive Intel Hex records. A Hex record is detailed below:

:NNAAAARRDD….DDCC<crlf>

NN - represents the number of data bytes in the record.
AAAA – represents address of the first byte in the record.
RR – Indicates the record type
DD - Indicates a data byte in Hex.
CC – Denotes the checksum of the record
<crlf> - end of record (Carriage Return, Line Feed)

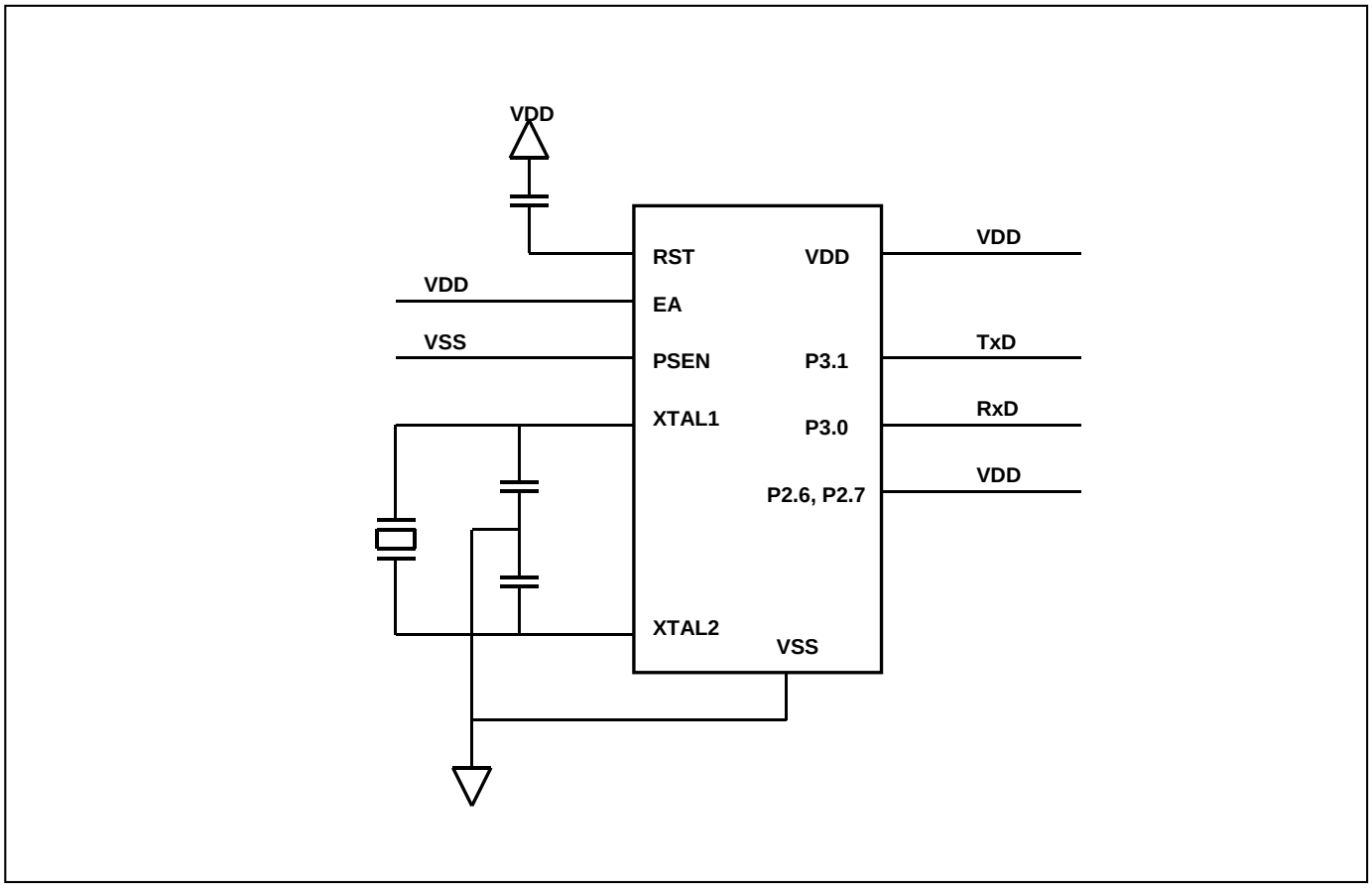


Figure ISP-1 - In-System Programming with a Minimum of Pins

The record types are summarized in Table 1.

As the record is received, a checksum is calculated on the data. If this checksum does not match the checksum in the record, an "X" (58h) is sent out of the serial port indicating an invalid checksum. If the checksums match, the command is executed and a "." (2Eh) is transmitted indicating a successful reception of the record. In some cases, data will be transmitted for displaying the contents of the Flash Memory.

In the case of a data record (record type 00), the "." character will not be sent until all of the bytes in the record have been programmed successfully. An "X" indicates that the checksum failed and an "R" (52h) indicates that one of the bytes did not program properly.

Table ISP-1. Intel Hex Records used by In-System Programming (ISP)

Record Type	ISP Command / Data Function
00	**Program Data** :nnaaaa00dd….ddcc **Where:** nn = number of bytes (hex) in record aaaa = memory address of first byte in record 00 = record type dd….dd = data bytes cc = checksum **Example:** :10020000120270745575B0A07B5AE5905403B40384
01	**End of File (EOF), no operation** :xxxxxx01cc **Where:** xxxxxx = required but value is a 'don't care' 01 = record type cc = checksum **Example:** :00000001FF
02	**Program Data** :01xxxx02ddcc **Where:** 01 = number of bytes (hex) in record xxxx = required but value is a 'don't care' 02 = record type dd = data bytes cc = checksum **Example:** :0100000210ED

Table ISP-1 (cont.). Intel Hex Records used by In-System Programming (ISP)

Record Type	ISP Command / Data Function
03	**Miscellaneous Write Functions** :nnxxxx03ffssddcc **Where:** nn = number of bytes (hex) in record xxxxxx = required but value is a 'don't care' 03 = record type ff = subfunction code ss = selection code dd = data input cc = checksum **Subfunction Code = 01 (Erase Block)** ff = 01 ss = block code 00 - block 0, 0000h to 1FFFh 20 - block 1, 2000h to 3FFFh 40 - block 2, 4000h to 7FFFh 80 - block 3, 8000h to BFFFh C0 - block 4, C000h to FFFFh **Example:** :0200000301807A Erase Block 3 **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 (No Flash Write) 01 - program security bit 2 (No Flash Read) 02 - program security bit 3 (No external memory access) **Example:** :020000030502F4 program security bit 3 **Subfunction Code = 06 (Program Status Byte or Boot Vector)** ff = 06 ss = 00 program status byte 01 program boot vector **Example:** :03000003060001F3 Program status byte with 01h **Subfunction code = 07 (Full Chip Erase)** ff = 07 ss = don't care dd = don't care **Example:** :0100000307F5

Table ISP-1 (cont.). Intel Hex Records used by In-System Programming (ISP)

Record Type	ISP Command / Data Function
04	**Display Device Data or Blank Check – Record type 04 causes the contents of the Flash to be sent out of the serial port in a formatted manner. The output consists of an address and 16 bytes of data.** **Format of Record Type 04** :05xxxx04sssseeeeffcc **Where:** 05 = number of bytes (hex) in record xxxx = don't'care 04 = record type ssss = starting address eeee = ending address ff = subfunction 00 = display data 01 = blank check cc = checksum **Example:** :0500000420003FFF0099 display data between 2000h – 3FFFh
05	**Miscellaneous Read Functions** :02xxxx05ffsscc **Where:** 02 = number of bytes (hex) in record xxxx = don't'care 05 = record type ffss = subfunction an d selection code x0000 = read signature byte – manufacturer id (15h) x0001 = read signature byte – device id #1 (C2h) x0003 = read signature byte – device id #2 ( ) x0700 = read security bits x0701 = read status byte x0702 = read boot vector cc = checksum **Example:** :020000050000F9 read signature byte – device id #1
06	**Direct Load of Baud Rate** :02xxxx06hhllcc **Where:** 02 = number of bytes (hex) in record xxxx = required but value is a 'don't care' 06 = record type hh = High Byte of Timer 2 ll = Low byte of Timer 2 cc = checksum **Example:** :02000006F50003

In-Application Programming (IAP)

The user code may make use of In-Application programming to permit selective erasing and programming of the TK89C668 Flash ROM. This is accomplished by performing a CALL to location FFF0h with the ENBOOT bit set. Four registers must first be initialized with the correct parameters for the function being performed.

See Table IAP-1 for a summary of the IAP calls.

The user may specify that the Watchdog Timer be fed by setting the MSB of the R1 parameter register before making the IAP call. Requesting that the Watchdog be fed should only be performed if the WDT is previously enabled, since the process of feeding the Watchdog will start the WDT if it was not previously working.

Table IAP-1. Intel Hex Records used by In-Application Programming (IAP)

IAP Function	Parameters

Program Data Byte	**Input Parameters:**
	R0 = oscillator frequency (nearest integer)
	R1 = 02h
	R1 = 82h (WDT feed)
	DPTR = Address of byte to program
	ACC = byte to program
	Return Parameter:
	ACC = 00h if pass, not 00h if fail
	Example Routine:
	WRDATA: MOV AUXR1, #20H ; set ENBOOT bit
	MOV R0, #11H ; set oscillator frequency
	MOV R1, #02h ;set to program data function
	MOV A, MyData ; set data to write
	MOV DPTR, Address ; specify address to write to
	CALL IAP ; execute IAP call
	RET

Erase Block	**Input Parameters:** R0 = oscillator frequency (nearest integer) R0 = 00h (Quick Erase) R1 = 01h R1 = 81h (WDT feed) DPH = block code as shown below: 00h - Block 0, 0000h to 1FFFh 20h - Block 1, 2000h to 3FFFh 40h - Block 2, 4000h to 7FFFh 80h - Block 3, 8000h to BFFFh C0h - Block 4, C000h to CFFFh DPL = 00h **Return Parameter:** None
Erase Boot Vector and Status Byte	**Input Parameters:** R0 = oscillator frequency (nearest integer) R1 = 04h R1 = 84h (WDT feed) DPH = 00h DPL = don't care **Return Parameter:** None

Table IAP-1 (cont). Intel Hex Records used by In-Application Programming (IAP)

IAP Function	Parameters
Program Security Bits	**Input Parameters:** R0 = oscillator frequency (nearest integer) R1 = 05h R1 = 85h (WDT feed) DPH = 00h DPL = 00h – Security bit #1 (No Flash Write) 01h – Security bit #2 (No Flash Read) 02h – Security bit #3 (No External memory access) **Return Parameter:** None
Program Status Byte	**Input Parameters:** R0 = oscillator frequency (nearest integer) R1 = 06h R1 = 86h (WDT feed) DPH = 00h DPL = 00h – program status byte ACC = status byte **Return Parameter:** ACC = 00h if pass, not 00h if fail
Program Boot Vector	**Input Parameters:** R0 = oscillator frequency (nearest integer) R1 = 06h R1 = 86h (WDT feed) DPH = 00h DPL = 01h – program boot vector ACC = boot vector **Return Parameter:** ACC = 00h if pass, not 00h if fail
Read Device Data	**Input Parameters:** R1 = 03h R1 = 83h (WDT feed) DPTR = address of byte to read **Return Parameter:** ACC = value of byte read
Read Manufacturer ID	**Input Parameters:** R0 = oscillator frequency (nearest integer) R1 = 00h R1 = 80h (WDT feed) DPH = 00h DPL = 00h – manufacturer ID **Return Parameter:** ACC = value of byte read

Table IAP-1 (cont). Intel Hex Records used by In-Application Programming (IAP)

IAP Function	Parameters
Read Device ID #1	**Input Parameters:** R0 = oscillator frequency (nearest integer) R1 = 00h R1 = 80h (WDT feed) DPH = 00h DPL = 01h – device ID #1 **Return Parameter:** ACC = value of byte read
Read Device ID #2	**Input Parameters:** R0 = oscillator frequency (nearest integer) R1 = 00h R1 = 80h (WDT feed) DPH = 00h DPL = 02h – device ID #2 **Return Parameter:** ACC = value of byte read
Read Security Bits	**Input Parameters:** R0 = oscillator frequency (nearest integer) R1 = 07h R1 = 87h (WDT feed) DPH = 00h DPL = 00h – security bits **Return Parameter:** ACC = value of byte read
Read Status Byte	**Input Parameters:** R0 = oscillator frequency (nearest integer) R1 = 07h R1 = 87h (WDT feed) DPH = 00h DPL = 01h – status byte **Return Parameter:** ACC = value of byte read
Read Boot Vector	**Input Parameters:** R0 = oscillator frequency (nearest integer) R1 = 07h R1 = 87h (WDT feed) DPH = 00h DPL = 02h – boot vector **Return Parameter:** ACC = value of byte read

Instruction Set

The TK89C668 has 255 different instructions. Most are addressing variations of 35 basic instruction groups. These addressing modes provide efficient and fast data transfer between various registers, memory, and peripherals.

- Implied - The operand is implied in the opcode.
- Immediate - The operand follows the opcode.
- Register Direct - The operand is contained in a register specified in the opcode.
- Register Indirect - The operand's address is in a register specified in the opcode.
- Direct - The operand's address follows the opcode.
- Absolute - The destination address is specified in the opcode.
- Absolute Page - 11 bits of the destination address are specified in the opcode.
- Relative - The opcode specifies an offset to the present address.
- Bit Direct - The operand's bit address follows the opcode.

Many of the opcodes use bits within the opcode to identify the operand. The following abbreviations are used as "hex" characters in these cases.

```
R = 1rrr
I = 011i
Z = aaa1
Y = aaa0
```

In these definitions, rrr signifies a register from R0 to R7, i signifies either R0 or R1, and aaa are address bits A10, A9, A8.

The specific implementation of an instruction is shown as:

```
64/3/2
```

which means that the machine code is 064H, there are three bytes, and it takes 2 cycles to execute.

Absolute Call

ACALL addr11

This instruction performs a subroutine call to an address within the same 2K block of memory as the first byte of the instruction immediately following the ACALL instruction. The opcode is followed by the lower 8 bits of the subroutine address.

```
Z1/2/2
```

Add

ADD A,<Source>

This instruction adds the operand to the accumulator and stores the result in the accumulator. The C, AC, and OV flags are affected.

```
2R/1/1      A + Rn > A
25/2/1      A + D > A
2I/1/1      A + @Ri > A
24/2/1      A + #N > A
```

Add With Carry

ADDC A, <Source>

This instruction adds the operand and the carry to the accumulator, and stores the result in the accumulator. The C, AC, and OV flags are affected.

```
3R/1/1      A + Rn + C > A
35/2/1      A + D + C > A
3I/1/1      A + @Ri + C > A
34/2/1      A + #N + C > A
```

Absolute Jump

AJMP addr11

This instruction performs an absolute jump within the same 2K block as the first byte of the instruction immediately following the AJMP. The byte following the opcode contains the lower 8 address bits of the jump destination.

```
Y1/2/2
```

And

ANL <Destination>,<Source>

This instruction performs a logical AND between the source and the destination, and then stores the results in the destination.

```
5R/1/1      A and Rn > A
55/2/1      A and D > A
5I/1/1      A and @Ri > A
54/2/1      A and #N > A
52/2/1      D and A > D
53/3/2      D and #N > D
```

```
82/2/2      C and Bit > C
B0/2/2      C and /Bit > C
```

Compare And Jump If Not Equal

CJNE <Dest.>, <Source>, <Offset>

This instruction subtracts the source from the destination, and performs a relative branch if the difference is not equal to zero. This instruction affects the carry flag.

```
B5/3/2      If A <> D, jump
B4/3/2      If A <> #N, jump
BR/3/2      If A <> Rn, jump
BI/3/2      If A <> @Ri, jump
```

Clear

CLR <Source>

This instruction clears the source.

```
E4/1/1      0 > A
C3/1/1      0 > C
C2/2/1      0 > Bit
```

Complement

CPL <Source>

This instruction complements the source.

```
F4/1/1      /A > A
B3/1/1      /C > C
B2/2/1      /Bit > Bit
```

Decimal Adjust On The Accumulator

DA A

Perform a decimal adjust on the accumulator. This instruction will add 6 to the lower nibble of the accumulator if the AC flag is set or if it contains a number greater than 9. It will also add a 6 to the upper nibble if the C flag is set, or if the nibble contains a number greater than 9, or if the lower nibble contains a number greater than 9 and the upper nibble contains a 9. The Carry flag is affected.

```
D4/1/1      A + (66) > A
```

Decrement

DEC <Destination>

The decrement instruction subtracts one from the destination. No flags are affected.

```
14/1/1      A - 1 > A
1R/1/1      Rn - 1 > Rn
15/2/1      D - 1 > D
1I/1/1      @Ri - 1 > @Ri
```

Divide

DIV AB

This instruction divides A by B. The quotient goes into A, and the remainder goes into B. The Carry flag is cleared. The overflow flag will be set if B contains 0. Otherwise, the overflow flag will be cleared.

```
84/1/4      A / B > A, B
```

Decrement And Jump I f Not Zero

DJNZ <Source>, <Offset>

This instruction decrements the source byte. A conditional branch is made if the result is not zero. No flags are affected.

```
DR/2/2      Rn - 1 > Rn,
            PC + 2 > PC

            If Rn <> 0 then
            PC + Rel > PC

D5/3/2      D - 1 > D,
            PC + 2 > PC

            If D <> 0 then
            PC + Rel > PC
```

Increment

INC <Source>

The increment instruction adds 1 to the source. No flags are affected.

```
04/1/1      A + 1 > A
0R/1/1      Rn + 1 > Rn
05/2/1      D + 1 > D
0I/1/1      @Ri + 1 > @Ri
A3/1/2      DPTR + 1 > DPTR
```

Conditional Branches

Jxx <Source>, <Offset>

There are seven conditional branches. If the condition is met, the offset is added to the program counter as an 8 bit signed number. No flags are affected.

```
40/2/2      JC, Jump if C = 1
50/2/2      JNC, Jump if C = 0
20/3/2      JB, Jump if Bit = 1
30/3/2      JNB, Jump if Bit = 0
60/2/2      JZ, Jump if A = 0
70/2/2      JNZ, Jump if A <> 0
10/3/2      JBC, Jump if Bit = 1
            and clear bit
```

Jump Indirect

JMP @A+DPTR

This instruction adds the accumulator to the data pointer and loads the result into the program counter. No flags are affected.

```
73/1/2      A + DPTR > PC
```

Long Call

LCALL addr16

This instruction performs a subroutine call to the 16 bit address. The current program counter is saved on the stack. No flags are affected.

```
12/3/2      Push PC on stack,
New address > PC
```

Long Jump

LJMP addr16

This instruction jumps to the 16 bit address. No flags are affected.

```
02/3/2  Address > PC
```

Move

MOV <Destination>, <Source>

This instruction moves the contents of the source to the contents of the destination. No flags are affected.

```
ER/1/1      Rn > A
E5/2/1      D > A
EI/1/1      @Ri > A
74/2/1      #N > A
FR/1/1      A > Rn
AR/2/2      D > Rn
7R/2/1      #N > Rn
F5/2/1      A > D
8R/2/2      Rn > D
85/3/2      D > D
8I/2/2      @Ri > D
75/3/2      #N > D
FI/1/1      A > @Ri
AI/2/2      D > @Ri
7I/2/1      #N > D
A2/2/1      Bit > C
92/2/2      C > Bit
90/3/2      #N > DPTR
```

Move Code

MOVC <Destination>, <Source>

These instructions calculate an address in the program space and move the address contents to the accumulator. No flags are affected.

```
93/1/2      (A + DPTR) > A
83/1/2      (A + PC) > A
```

Move External Data

MOVX <Destination>, <Source>

These instructions read or write the contents of the accumulator into the data space. No flags are affected.

```
E0/1/2      @DPTR > A
E2/1/2      @R0 > A
E3/1/2      @R1 > A
F0/1/2      A > @DPTR
F2/1/2      A > @R0
F3/1/2      A > @R1
```

Multiply

MUL AB

This instruction performs an unsigned multiply between the contents of the accumulator and the B register. The sixteen bit result is stored in the B register and the Accumulator. The Carry flag is cleared. The overflow flag is set if the product is greater than 255.

```
A4/1/4      A * B > B,A
```

No Operation

NOP

A NOP does nothing. The undefined opcode A5 acts as a two cycle NOP. No flags are affected.

```
00/1/1
A5/2/2
```

OR

ORL <Destination>, <Source>

This instruction does a logical OR between the source and destination..

```
4R/1/1      Rn or A > A
45/2/1      D or A > A
4I/1/1      @Ri or A > A
44/2/1      #N or A > A
42/2/1      D or A > D
43/3/2      #N or D > D
72/2/2      Bit or C > C
A0/2/2      /Bit or C > C
```

Pop Operand From Stack

POP <Destination>

This opcode moves a variable from the stack to the destination. The stack pointer is then decremented. No flags are affected.

```
D0/2/2      Pop D
```

Push Operand On Stack

PUSH <Source>

This opcode increments the stack pointer and then moves a byte from the source to the stack. No flags are affected.

```
C0/2/2      Push D
```

Return

RET

This opcode restores the program counter from the stack and decrements the stack pointer by 2. No flags are affected.

```
22/1/2      Stack > PC, SP-2>SP
```

Return From Interrupt

RETI

This opcode restores the program counter from the stack and decrements the stack pointer by 2. It also clears the interrupt status bit. No flags are affected.

```
32/1/2      Stack > PC, SP-2>SP
```

Rotate Accumulator

Rxx A

These instructions rotate the accumulator 1 bit either left or right, and either around or through the carry flag. No other flags are affected.

```
23/1/1      RL A - Rotate A
            left.
33/1/1      RLC A - Rotate A
            left through carry.
03/1/1      RR A - Rotate A
            right.
13/1/1      RRC A - Rotate A
            right through carry.
```

Set Bit

SETB <Source>

This instruction sets a specified bit. No flags are affected.

```
D3/1/1      1 > C
D2/2/1      1 > Bit
```

Short Jump

SJMP <Offset>

This opcode is an unconditional branch. The offset is treated as a signed 8 bit number that is added to the program counter. No flags are affected.

```
80/2/2      PC + Offset > PC
```

Subtract With Borrow

SUBB <Destination>, <Source>

This opcode subtracts the source and the carry from the accumulator, and stores the results in the accumulator. The C, AC and OV flags are affected.

```
9R/1/1      A - C - Rn > A
95/2/1      A - C - D > A
```

```
9I/1/1      A - C - @Ri > A
94/2/1      A - C - #N > A
```

Swap Nibbles In Accumulator

SWAP A

This instruction swaps the nibbles in the accumulator. No flags are affected.

```
C4/1/1      A(0-3) <-> A(4-7)
```

Exchange Accumulator With Operand

XCH A, <Source>

This instruction exchanges the accumulator with operand. No flags are affected.

```
CR/1/1      A <-> Rn
C5/2/1      A <-> D
CI/1/1      A <-> @Ri
```

Exchange Digit Within Accumulator With Operand

XCHD A, <Source>

This instruction exchanges a nibble between the accumulator and the operand. No flags are affected.

```
DI/1/1      A(0-3) <-> @Ri(0-3)
```

Exclusive Or

XRL <Destination>, <Source>

This instruction performs an exclusive-or between the source and destination. No flags are affected.

```
6R/1/1      A xor Rn > A
65/2/1      A xor D > A
6I/1/1      A xor @Ri > A
64/2/1      A xor #N > A
62/2/1      D xor A > D
63/3/2      D xor #N > D
```

Op-Code Map

Upper 4 Bits	Lower 4 Bits																
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
0	NOP	AJMP	LJMP	RR A	INC A	INC D	INC @R0	INC @R1	INC R0	INC R1	INC R2	INC R3	INC R4	INC R5	INC R6	INC R7	0
1	JBC	ACALL	LCALL	RRC A	DEC A	DEC D	DEC @R0	DEC @R1	DEC R0	DEC R1	DEC R2	DEC R3	DEC R4	DEC R5	DEC R6	DEC R7	1
2	JB	AJMP	RET	RL A	ADD A, N	ADD A, D	ADD @R0	ADD @R1	ADD A, R0	ADD A, R1	ADD A, R2	ADD A, R3	ADD A, R4	ADD A, R5	ADD A, R6	ADD A, R7	2
3	JNB	ACALL	RETI	RLC A	ADDC A, N	ADDC A, D	ADDC @R0	ADDC @R1	ADDC A, R0	ADDC A, R1	ADDC A, R2	ADDC A, R3	ADDC A, R4	ADDC A, R5	ADDC A, R6	ADDC A, R7	3
4	JC	AJMP	ORD, A	ORL D,#N	ORL A, N	ORL A, D	ORL @R0	ORL @R1	ORL A, R0	ORL A, R1	ORL A, R2	ORL A, R3	ORL A, R4	ORL A, R5	ORL A, R6	ORL A, R7	4
5	JNC	ACALL	ANLD, A	ANL D,#N	ANL A, N	ANL A, D	ANL @R0	ANL @R1	ANL A, R0	ANL A, R1	ANL A, R2	ANL A, R3	ANL A, R4	ANL A, R5	ANL A, R6	ANL A, R7	5
6	JZ	AJMP	XRLD, A	XRL D,#N	XRL A, N	XRL A, D	XRL @R0	XRL @R1	XRL A, R0	XRL A, R1	XRL A, R2	XRL A, R3	XRL A, R4	XRL A, R5	XRL A, R6	XRL A, R7	6
7	JNZ	ACALL	ORC,Bit	JMP @A+DP	MOV A,N	MOV D,N	MOV @R0,N	MOV @R1,N	MOV R0,N	MOV R1,N	MOV R2,N	MOV R3,N	MOV R4,N	MOV R5,N	MOV R6,N	MOV R7,N	7
8	SJMP	AJMP	ANLC.Bit	MOVC A+PC	DIV A,B	MOV D,D	MOV D,@RO	MOV D,@R1	MOV R0,D	MOV R1,D	MOV R2,D	MOV R3,D	MOV R4,D	MOV R5,D	MOV R6,D	MOV R7,D	8
9	MOV DP,N	ACALL	MOV Bit,C	MOVC A+DP	SUBB A, N	SUBB A, D	SUBB @R0	SUBB @R1	SUBB A, R0	SUBB A, R1	SUBB A, R2	SUBB A, R3	SUBB A, R4	SUBB A, R5	SUBB A, R6	SUBB A, R7	9
A	ORC,/Bit	AJMP	MOV C,Bit	INC DP	MUL A,B	NOP	MOV @R0,D	MOV @R1,D	MOV R0,D	MOV R1,D	MOV R2,D	MOV R3,D	MOV R4,D	MOV R5,D	MOV R6,D	MOV R7,D	A
B	ANL C,/Bit	ACALL	CPL BIT	CPL C	CJNE A, N	CJNE A, D	CJNE @R0	CJNE @R1	CJNE R0,N	CJNE R1,N	CJNE R2,N	CJNE R3,N	CJNE R4,N	CJNE R5,N	CJNE R6,N	CJNE R7,N	B
C	PUSH	AJMP	CLR Bit	CLR C	SWAP A	XCH A, D	XCH @R0	XCH @R1	XCH A, R0	XCH A, R1	XCH A, R2	XCH A, R3	XCH A, R4	XCH A, R5	XCH A, R6	XCH A, R7	C
D	POP	ACALL	SETB D	SETB C	DAA	DJNZ D	XCHD @R0	XCHD @R1	DJNZ R0	DJNZ R1	DJNZ R2	DJNZ R3	DJNZ R4	DJNZ R5	DJNZ R6	DJNZ R7	D
E	MOVX A,@DP	AJMP	MOVX A,@R0	MOVX A,@R1	CLR A	MOV A, D	MOV A,@R0	MOV A,@R1	MOV A, R0	MOV A, R1	MOV A, R2	MOV A, R3	MOV A, R4	MOV A, R5	MOV A, R6	MOV A, R7	E
F	MOVX @DP,A	ACALL	MOVX @R0,A	MOVX @R1,A	CPL A	MOV D,A	MOV @R0,A	MOV @R1,A	MOV R0,A	MOV R1,A	MOV R2,A	MOV R3,A	MOV R4,A	MOV R5,A	MOV R6,A	MOV R7,A	F
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	

Electrical Specifications

Maximum Ratings

Characteristics		Symbol	Min	Max	Unit
Supply Voltage		Vdd	-0.5	5.5	V
Input Voltage		Vin	Vss – 0.3	Vdd + 0.3	V
Current Drain per Pin		I_{OL}		15	mA
Operating Temperature Range	Commercial Industrial	Tac Tai	0 -40	70 85	°C °C
Storage Temperature range		Tstg	-55	+150	°C

DC Electrical Specifications (Vdd = 5.0 V +/- 10%, Vss = 0 V, Ta = 0°C to +70°C)

Characteristics	Condition	Symbol	Min	Max	Unit
Input high level (Ports 0, 1, 2, 3, /EA)		V_{IH}	2.0	Vdd	V
Input high level (X1, RST)		V_{IH1}	2.0	Vdd	V
Input low level		V_{IL}	0.0	0.8	V
Output high level	Ioh = 2 mA	V_{OH}	2.4	Vdd	V
Output high level	Ioh = 4 mA	V_{OH1}	2.4	Vdd	V
Output low level	Iol = 2 mA	V_{OL}	0	0.4	V
Output low level	Iol = 4 mA	V_{OL1}	0	0.4	V
Input current (Ports 1, 2, 3)	V_{IN} = 0.4 V	I_{LI}	-10	10	uA
Logical 1 to 0 transition current (Ports 1, 2, 3)	Itl	I_L	-10	10	uA
Input Leakage current (Port 0)		I_L	-10	10	uA
Supply current, Active mode,	X1 = 12 MHz	I_{CCA}		20	mA
Internal Reset Pull-Down Resistor	Vin = 0V	R_{RST}	-10	10	uA
Pin Capacitance	Cio	C_{IO}		10	uA

Notes:

1. AC measurements made with a 50 pF load, at a 50% supply voltage level.
2. Float delay measured with a 3.3K resistor tied to opposite supply, measured after a +/- 0.2V change in voltage level.

AC Electrical Specifications (Vdd = 3.3 V +/- 10%, Vss = 0 V, Ta = 0ºC to +70ºC)

Characteristics	Symbol	Min	Max	Unit
Oscillator frequency	**1/t**clcl	0	33	MHz
ALE pulse width	**t**lhll	2**t**clcl -40		ns
Address valid to ALE low	**t**avll	**t**clcl -25		ns
Address Hold after ALE low	**t**llax	**t**clcl -25		ns
ALE low to valid instruction in	**t**lliv		4**t**clcl -65	ns
ALE low to PSEN low	**t**llpl	**t**clcl -25		ns
PSEN pulse width	**t**plph	3**t**clcl -45		ns
PSEN low to valid instruction in	**t**pliv		3**t**clcl –60	ns
Input instruction hold after PSEN	**t**pxix	0		ns
Input instruction float after PSEN	**t**pxiz		**t**clcl -25	ns
Address to valid instruction in	**t**aviv		5**t**clcl -80	ns
PSEN low to address float	**t**plaz		10	ns
Data Memory				
RD pulse width	**t**rlrh	6**t**clcl -100		ns
WR pulse width	**t**wlwh	6**t**clcl -100		ns
RD low to valid data in	**t**rldv		5**t**clcl -90	ns
Data hold after RD	**t**rhdx	0		ns
Data float after RD	**t**rhdz		2**t**clcl -28	ns
ALE low to data valid	**t**lldv		8**t**clcl -150	ns
Address to valid data in	**t**avdv		9**t**clcl -165	ns
ALE low to RD or WR low	**t**llwl	3**t**clcl -50	3**t**clcl -50	ns
Address valid to WR low or RD low	**t**avwl	4**t**clcl -75		ns
Data Valid to WR transition	**t**qvwx	**t**clcl -30		ns
Data hold after WR	**t**whqx	**t**clcl -25		ns
Data valid to WR high	**t**qvwh	7**t**clcl -130		ns
RD low to address float	**t**rlaz			ns
RD or WR high to ALE high	**t**whlh	**t**clcl -25		ns

Notes:

3. AC measurements made with a 50 pF load, at a 50% supply voltage level.
4. Float delay measured with a 3.3K resistor tied to opposite supply, measured after a +/- 0.2V change in voltage level.

Timing Diagrams

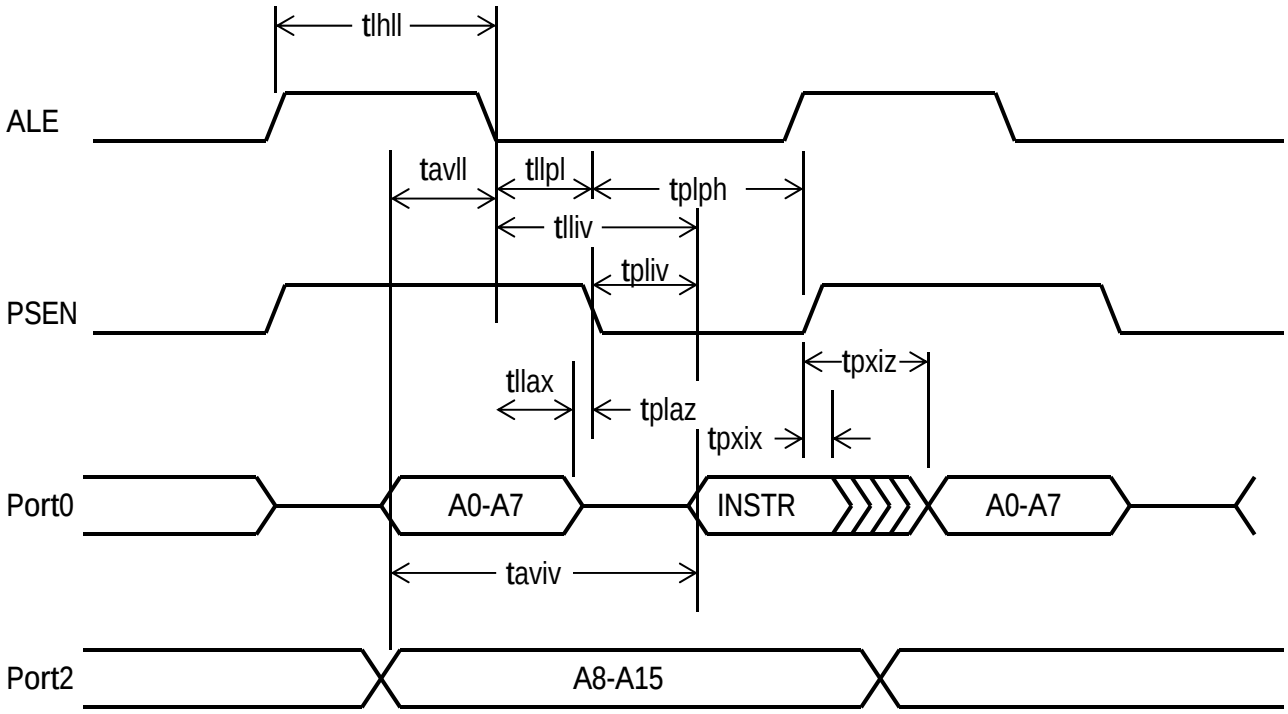


Figure Time-1 – External Program Memory Read Cycle

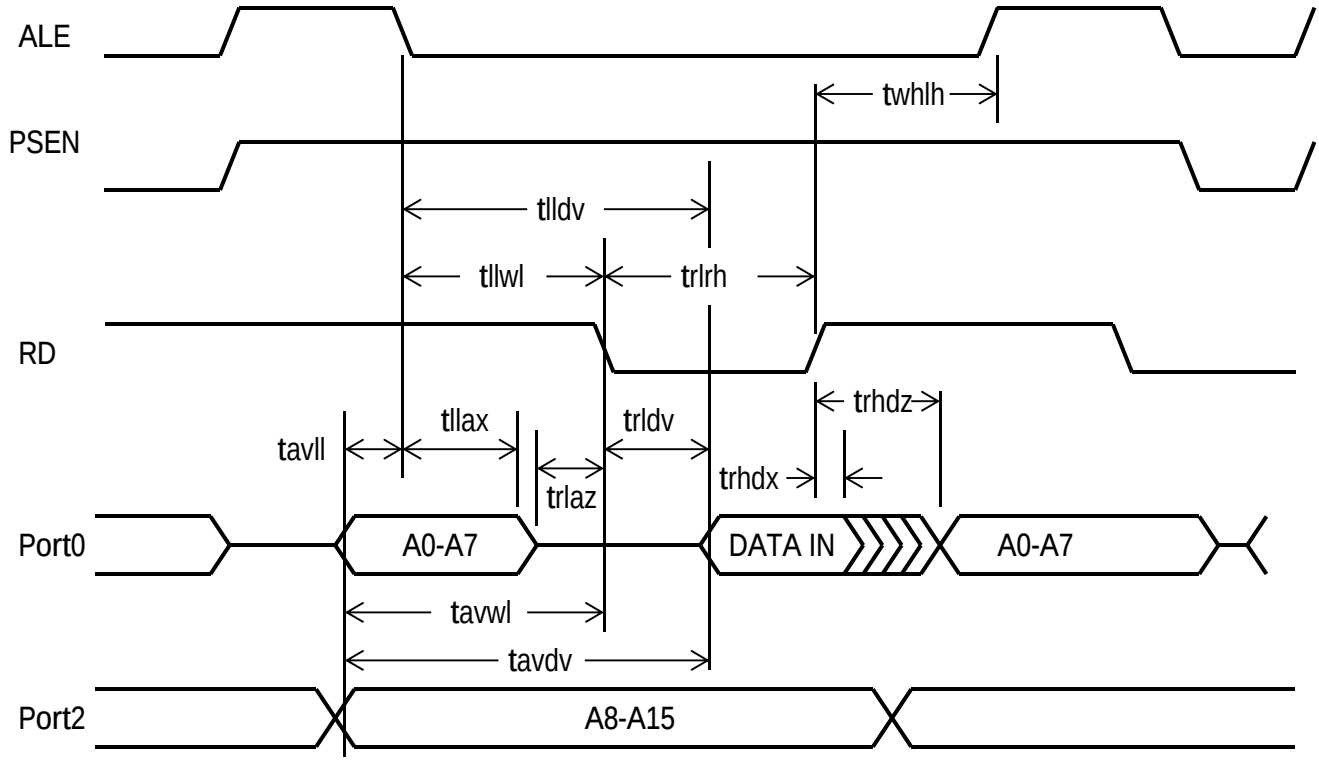


Figure Time-2 – External Program Memory Read Cycle

Errata

TK89C668 – Rev A – Code 7602

1. The IAP / ISP routines clear the watchdog bit. Using the IAP / ISP while the watchdog is active can result in a watchdog timeout.

2. The overflow on Timer 0 is not recognized by the PCA (CPS1, CPS0 = 10).

3. In the TWI hardware, while AA is reset, the TWI **will** respond to a General Call Address. It normally should not respond. Users should either not use GC addressing **or** make sure GC is not disabled with AA reset.

4. The upper frequency in 12X clock mode is 20 MHz instead of 33 MHz. The upper frequency for the 6X clock mode is still 20 MHz.

5. When in power down, the leading edge of interrupt triggers a release from power down. It should be the trailing edge.

6. The flash may fail to initialize the part if the initial reset pulse is asserted before Vdd reaches operating levels. The flash initialization is occurring at the beginning of reset. If Vdd is still below the flash operating level, incorrect data is read from the flash, resulting in incorrect values in the X6 and security bits.

7. Any RMW instructions which read the I2C port read the pin value instead of the latch value. This can cause the I2C bus to hang if a “0” is read from the bus during this operation.

8. Any AND, OR, or XOR instruction which is directly acting on a SFR register may inadvertently corrupt a bit that changes during the instruction. This has been observed to occur in the SCON register when a receive interrupt is cleared while processing a transmit interrupt.

9. The JBC instruction may clear a bit that was set during the instruction execution, without taking the jump. This is of particular concern with the interrupt bits.

10. Switching from Mode 1 to Mode 3 in the UART may cause the UART to hang-up.

11. If bit 8 is a 1, then RB8 is correctly set. However, if bit 8 is a zero, we do not clear RB8. A software workaround is to have the program clear RB8 after it has been tested and found to be a 1.

TK89C668 – Rev B – Code 7748 – Date code 1701 and later

Revision B has no known errata.

Ordering Information

Here is the explanation for the TK89C668 ordering codes.

Device	Package	Temperature	Replaces
TK89C668AI	PLCC 44	0 C – 70 C	P89C668HBA
TK89C668AI	PLCC 44	-40 C – 85 C	P89C668HFA

Contact Information

The TK89C668 series may be ordered directly from Tekmos

Tekmos, Inc.
7901 E. Riverside Drive
Building 2, Suite 150
Austin, TX 78744

512 342-9871 phone
512 342-9873 fax
Sales@Tekmos.Com
www.Tekmos.com

Revision History

Date	Revision	Description
12/20/07	0.1	Internal Review
3/18/08	1.0	Initial Release
8/4/08	1.1	Add errata 5
9/2/09	1.2	Add errata
1/30/13	1.3	Correct errors in interrupt vector addresses, add errata 11
5/2/17	1.4	Change part number, update errata, change address