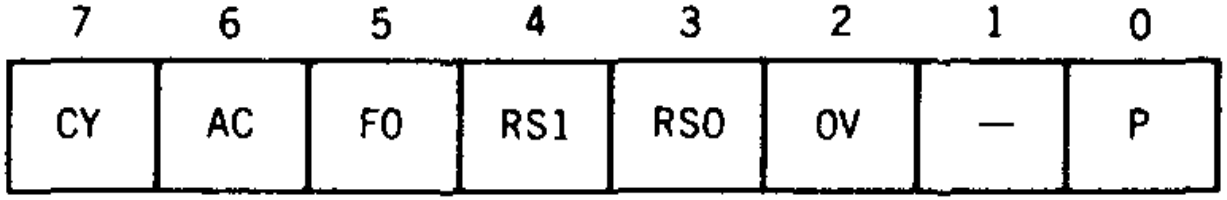
Microcontroller Performance

To calculate the time any particular instruction will take to be executed, find the number of cycles, *C*, from the list in Appendix A. The time to execute that instruction is then found by multiplying *C* by 12 and dividing the product by the crystal frequency:

$$\text{Tinst} = \frac{C \times 12\text{d}}{\text{crystal frequency}}$$

For example, if the crystal frequency is 16 megahertz, then the time to execute an ADD A, R1 one-cycle instruction is .75 microseconds. A 12 megahertz crystal yields the convenient time of one microsecond per cycle. An 11.0592 megahertz crystal, while seemingly an odd value, yields a cycle frequency of 921.6 kilohertz, which can be divided evenly by the standard communication baud rates of 19200, 9600, 4800, 2400, 1200, and 300 hertz.

PSW Program Status Word Register



THE PROGRAM STATUS WORD (PSW) SPECIAL FUNCTION REGISTER

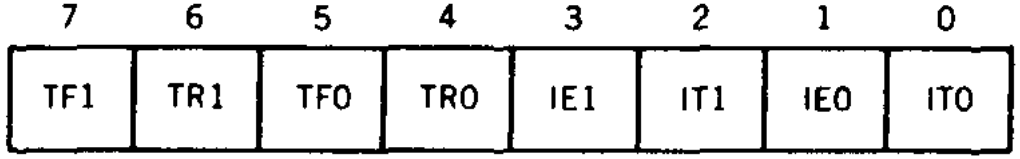
Bit	Symbol	Function
7	CY	Carry flag; used in arithmetic, JUMP, ROTATE, and BOOLEAN instructions
6	AC	Auxilliary carry flag; used for BCD arithmetic
5	F0	User flag 0
4	RS1	Register bank select bit 1
3	RS0	Register bank select bit 0

RS1	RS0	
0	0	Select register bank 0
0	1	Select register bank 1
1	0	Select register bank 2
1	1	Select register bank 3

2	OV	Overflow flag; used in arithmetic instructions
1	—	Reserved for future use
0	P	Parity flag; shows parity of register A: 1 = Odd Parity

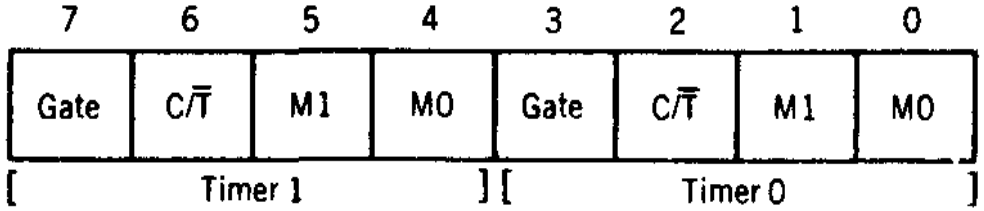
Bit addressable as PSW.0 to PSW.7

FIGURE 2.10 TCON and TMOD Function Registers



THE TIMER CONTROL (TCON) SPECIAL FUNCTION REGISTER

Bit	Symbol	Function
7	TF1	Timer 1 Overflow flag. Set when timer rolls from all ones to zero. Cleared when processor vectors to execute interrupt service routine located at program address 001Bh.
6	TR1	Timer 1 run control bit. Set to 1 by program to enable timer to count; cleared to 0 by program to halt timer. Does not reset timer.
5	TF0	Timer 0 Overflow flag. Set when timer rolls from all ones to zero. Cleared when processor vectors to execute interrupt service routine located at program address 000Bh.
4	TR0	Timer 0 run control bit. Set to 1 by program to enable timer to count; cleared to 0 by program to halt timer. Does not reset timer.
3	IE1	External interrupt 1 edge flag. Set to 1 when a high to low edge signal is received on port 3 pin 3.3 ($\overline{INT1}$). Cleared when processor vectors to interrupt service routine located at program address 0013h. Not related to timer operations.
2	IT1	External interrupt 1 signal type control bit. Set to 1 by program to enable external interrupt 1 to be triggered by a falling edge signal. Set to 0 by program to enable a low level signal on external interrupt 1 to generate an interrupt.
1	IE0	External interrupt 0 edge flag. Set to 1 when a high to low edge signal is received on port 3 pin 3.2 ($\overline{INT0}$). Cleared when processor vectors to interrupt service routine located at program address 0003h. Not related to timer operations.

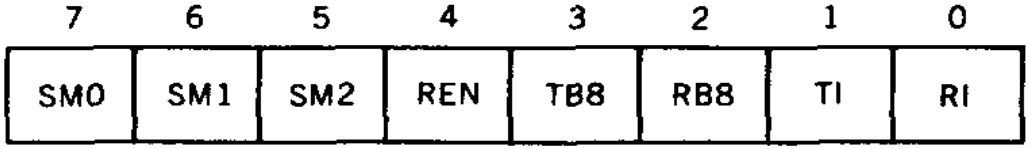


THE TIMER MODE CONTROL (TMOD) SPECIAL FUNCTION REGISTER

Bit	Symbol	Function
7/3	Gate	OR gate enable bit which controls RUN/STOP of timer 1/0. Set to 1 by program to enable timer to run if bit TR1/0 in TCON is set and signal on external interrupt $\overline{INT1}$/0 pin is high. Cleared to 0 by program to enable timer to run if bit TR1/0 in TCON is set.
6/2	C/$\overline{T}$	Set to 1 by program to make timer 1/0 act as a counter by counting pulses from external input pins 3.5 (T1) or 3.4 (T0). Cleared to 0 by program to make timer act as a timer by counting internal frequency.
5/1	M1	Timer/counter operating mode select bit 1. Set/cleared by program to select mode.
4/0	M0	Timer/counter operating mode select bit 0. Set/cleared by program to select mode.

M1	M0	Mode
0	0	0
0	1	1
1	0	2
1	1	3

TMOD is not bit addressable

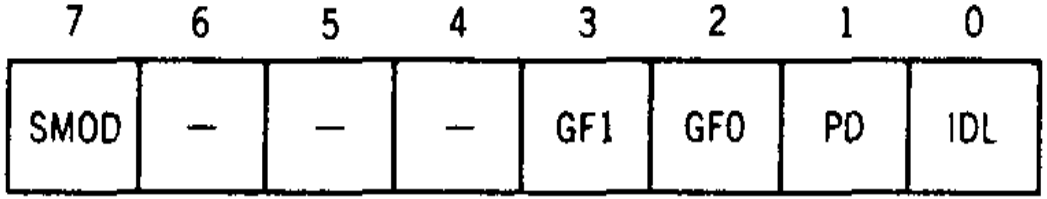


THE SERIAL PORT CONTROL (SCON) SPECIAL FUNCTION REGISTER

Bit	Symbol	Function
7	SM0	Serial port mode bit 0. Set/cleared by program to select mode.
6	SM1	Serial port mode bit 1. Set/cleared by program to select mode.

SM0	SM1	Mode	Description
0	0	0	Shift register; baud = f/12
0	1	1	8-bit UART; baud = variable
1	0	2	9-bit UART; baud = f/32 or f/64
1	1	3	9-bit UART; baud = variable

5	SM2	Multiprocessor communications bit. Set/cleared by program to enable multiprocessor communications in modes 2 and 3. When set to 1 an interrupt is generated if bit 9 of the received data is a 1; no interrupt is generated if bit 9 is a 0. If set to 1 for mode 1, no interrupt will be generated unless a valid stop bit is received. Clear to 0 if mode 0 is in use.
4	REN	Receive enable bit. Set to 1 to enable reception; cleared to 0 to dissable reception.
3	TB8	Transmitted bit 8. Set/cleared by program in modes 2 and 3.
2	RB8	Received bit 8. Bit 8 of received data in modes 2 and 3; stop bit in mode 1. Not used in mode 0.
1	TI	Transmit interrupt flag. Set to one at the end of bit 7 time in mode 0, and at the beginning of the stop bit for other modes. Must be cleared by the program.
0	RI	Receive interrupt flag. Set to one at the end of bit 7 time in mode 0, and halfway through the stop bit for other modes. Must be cleared by the program.



THE POWER MODE CONTROL (PCON) SPECIAL FUNCTION REGISTER

Bit	Symbol	Function
7	SMOD	Serial baud rate modify bit. Set to 1 by program to double baud rate using timer 1 for modes 1, 2, and 3. Cleared to 0 by program to use timer 1 baud rate.
6-4	—	Not implemented.
3	GF1	General purpose user flag bit 1. Set/cleared by program.
2	GF0	General purpose user flag bit 0. Set/cleared by program.
1	PD	Power down bit. Set to 1 by program to enter power down configuration for CHMOS processors.
0	IDL	Idle mode bit. Set to 1 by program to enter idle mode configuration for CHMOS processors. PCON is not bit addressable.

What is a performance metric?

- Count
  - Of how many times an event occurs
- Duration
  - Of a time interval
- Size
  - Of some parameter
- A value derived from these fundamental measurements

Performance Measures

CPU Speed is measured as the Number of basic operations it can perform per unit time.

Ex. Typical basic operation of fixed point addition of the content of two registers R1 and R2 is:

R1:=R1+R2

Such operations are timed by a regular stream of signals(ticks or beats) issued by System clock.

Speed of Clock = Frequency (f) measured in Millions of ticks per second (MHz).

Clock Cycle or clock period (T_{clock}) = time required to execute the operation =1/f microseconds.

Ex. If Clock speed is 250 MHz can perform one basic operation in the clock period (T_{clock}) = 1/250=.004 μs.

NOTE: Operations such as division or floating point numbers requires more than one clock cycle to complete the execution.

CPU's processing of an instruction involves several steps each of which requires at least one clock cycle:

1. Fetch the instruction from main memory M.
2. Decode the instruction's opcode.
3. Load from M any operands needed unless they are already in CPU registers.
4. Execute the instruction via register to register operation.
5. Store the result in M.

$T = N / IPS$

T= total program execution time

N= Actual no. of instruction executed.

IPS= Average number of instruction executed per second

$CPI = (f * 10^6) / IPS$

CPI= Average no of cycles per instruction.

f = CPU's clock frequency(MHz).

Hence $T = N * CPI / (f * 10^6)$

MIPS(Millions of instruction executed per second)$= IPS * 10^6$ or

$= f / CPI$.

Equation T=N*CPI/ (f * 10^6) shows that the three separate factors

Software, Architecture, and Hardware jointly determines the

performance of CPU .

Software: Efficiency with which the programs are written and compiled

Into object code influences N, the no. of instruction executed. Reducing

N reduces the overall execution time T.

Architecture: The Efficiency with which individual instructions are

Processed directly affects CPI. Reducing CPI reduces T.

Hardware: Speed of processor determines f , the clock frequency.

Increasing f reduces T.

Note: *CISC processor's aim is to reduce N at the expence of CPI whereas*

RISC processors aim to reduce CPI at expence of N.

MFLOPS

- Better definition of “distance traveled”
- 1 unit of computation (~ distance) ≡ 1 floating-point operation
- *Millions of floating-point ops per second*
- MFLOPS = f / (T_e * 1000000)
  - f = number of floating-point instructions
  - T_e = execution time