		Ethernet cable, fiber optic cable splicing, fiber optic cable mechanical splices, insulation, gauge, current capacity, flexibility etc. used in various electronics products, different input output sockets (15 hrs) 208. Identify suitable connectors, solder/crimp /terminate & test the cable sets. (10 hrs) 209. Check the continuity as per the marking on the connector for preparing the cable set. (10 hrs) 210. Identify and select various connectors and cables inside the CPU cabinet of PC. (10 hrs) 211. Identify the suitable connector and cable to connect a computer with a network switch and prepare a cross over cable to connect two network computers. (5 hrs)	connector & their terminations to the cables. Male / Female type DB connectors. Ethernet 10 Base cross over cables and pin out assignments, UTP and STP, SCTP, TPC, coxoial, types of fibre optical Cables and Cable trays. Different types of connectors Servo 0.1" connectors, FTP, RCA,BNC,HDMI Audio/video connectors like XLR, RCA (phono), 6.3 mm PHONO, 3.5 / 2.5 mm PHONO, BANTAM, SPEAKON, DIN, mini DIN, RF connectors, USB, Fire wire, SATA Connectors, VGA, DVI connectors, MIDI and RJ45,RJ11 etc.
63-65	• Assemble and test a commercial AM/ FM receiver and evaluate performance.	Communication electronics 212. Modulate and Demodulate various signals using AM and FM on the trainer kit and observe waveforms (10 hrs) 213. Construct and test IC based AM Receiver (10 hrs) 214. Construct and test IC based FM transmitter (10 hrs) 215. Construct and test IC based AM transmitter and test the transmitter power. Calculate the modulation index. (10 hrs) 216. Dismantle the given FM receiver set and identify different stages (AM section, audio amplifier section etc) (10 hrs) 217. Modulate two signals using AM kit draw the way from and calculate percent (%) of modulation. (10 hrs) 218. Modulate and Demodulate a signal using PAM, PPM, PWM Techniques (15 hrs)	Radio Wave Propagation – principle, fading. Need for Modulation, types of modulation and demodulation. Fundamentals of Antenna, various parameters, types of Antennas & application. Introduction to AM, FM & PM, SSB-SC & DSB-SC. Block diagram of AM and FM transmitter. FM Generation & Detection. Digital modulation and demodulation techniques, sampling, quantization & encoding. Concept of multiplexing and de multiplexing of AM/ FM/ PAM/ PPM /PWM signals. A simple block diagram approach to be adopted for explaining the above mod/ demod. techniques
66-68	• Test, service and troubleshoot the various components of different domestic/ industrial programmable systems.	Microcontroller (8051) 219. Identify various ICs & their functions on the given Microcontroller Kit. (5 hrs) 220. Identify the address range of RAM & ROM. (5 hrs) 221. Measure the crystal frequency, connect it to the controller. (5 hrs)	Introduction Microprocessor & 8051Microcontroller, architecture, pin details & the bus system. Function of different ICs used in the Microcontroller Kit. Differentiate microcontroller with microprocessor. Interfacing of memory to the

		222.Identify the port pins of the controller & configure the ports for Input & Output operation. (7 hrs) 223.Use 8051 microcontroller, connect 8 LED to the port, blink the LED with a switch. (10 hrs) 224. Perform the initialization, load & turn on a LED with delay using Timer. (8 hrs) 225. Perform the use of a Timer as an Event counter to count external events. (10 hrs) 226. Demonstrate entering of simple programs, execute & monitor the results. (10 hrs) 227. Perform with 8051 microcontroller assembling language program, check the reading of an input port and sending the received bytes to the output port of the microcontroller, used switches and LCD for the input and output. (15 hrs)	microcontroller. Internal hardware resources of microcontroller. I/O port pin configuration. Different variants of 8051 & their resources. Register banks & their functioning. SFRs & their configuration for different applications. Comparative study of 8051 with 8052. Introduction to PIC Architecture.
69-71	• Execute the operation of different process sensors, identify, wire & test various sensors of different industrial processes by selecting appropriate test instruments.	Sensors, Transducers and Applications 228. Identify sensors used in process industries such as RTDs, Temperature ICs, Thermocouples, proximity switches (inductive, capacitive and photo electric), load cells, strain gauge. LVDT PT 100 (platinum resistance sensor), water level sensor, thermostat float switch, float valve by their appearance (15 hrs) 229. Measure temperature of a lit fire using a Thermocouple and record the readings referring to data chart. (15 hrs) 230.Measure temperature of a lit fire using RTD and record the readings referring to data chart (15 hrs.) 231. Measure the DC voltage of a LVDT (15 hrs) 232. Detect different objectives using capacitive, inductive and photoelectric proximity sensors (15 hrs)	Basics of passive and active transducers. Role, selection and characteristics. Sensor voltage and current formats. Thermistors / Thermocouples - Basic principle, salient features, operating range, composition, advantages and disadvantages. Strain gauges/ Load cell – principle, gauge factor, types of strain gauges. Inductive/ capacitive transducers - Principle of operation, advantages and disadvantages. Principle of operation of LVDT, advantages and disadvantages. Proximity sensors – applications, working principles of eddy current, capacitive and inductive proximity sensors

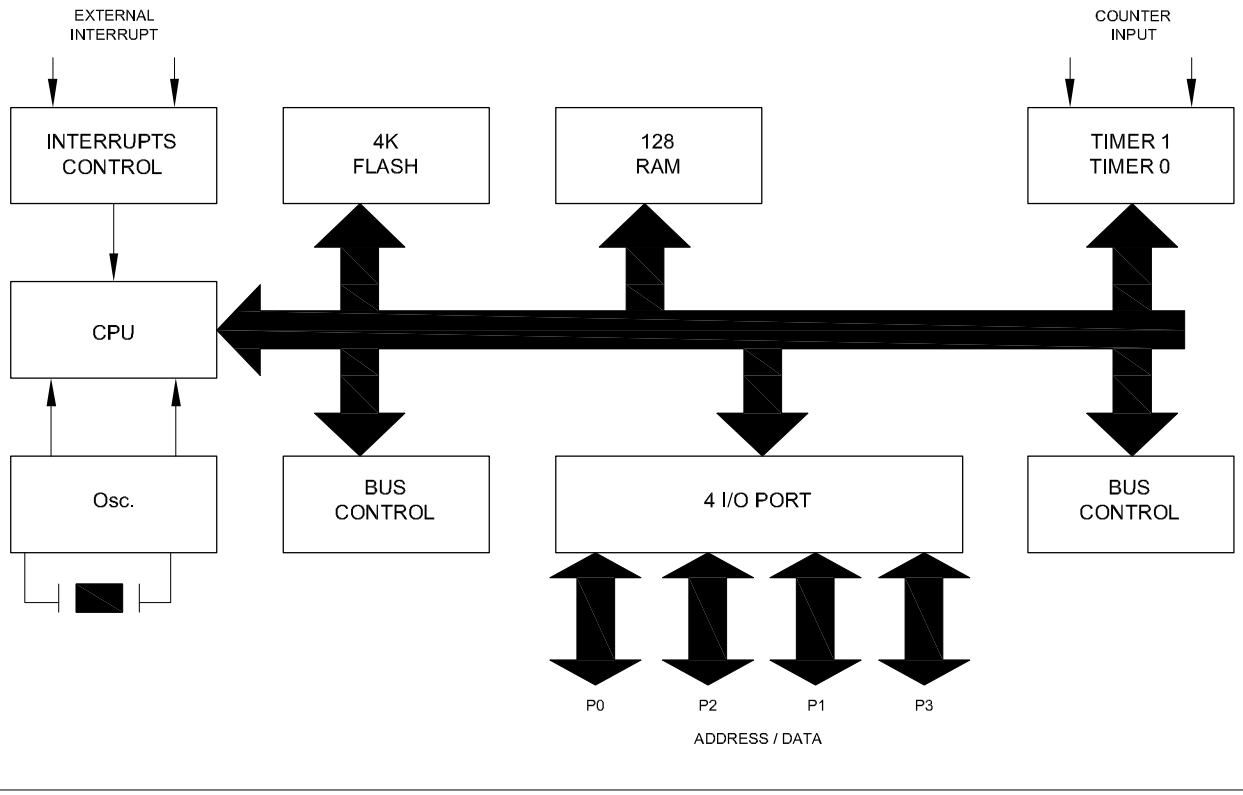
Architecture of 8051

Objectives : At the end of this lesson you shall be able to
- **understand the architecture of 8051 microcontroller**
- **differetiate between microprocessor and microcontroller**
- **observe advantages of microcontroller.**

Microcontroller

The main reason for the development of microcontroller is to overcome the drawback of the microprocessor. Even though microprocessors are powerful devices, they require external chips like RAM, ROM input/output ports and other components in order to design a complete working system. This made it economically difficult to develop computerized consumer applicances on a large scale as the system cost is very high. Microcontrollers are the devices that actually fit the profile “Computer - on - a chip” as it consists of a main processing unit or processor along with some other components that are necessary to make it a complete computer. The components that are present on a typical microcontroller IC are CPU, memory, input / ouput ports and timers. The block diagram of a microcontroller is shown below in fig.1.

Fig 1



Microcontrollers are basically used in embedded systems. Microcontrollers can be classified based on bus width, memory structure and instruction set. Bus width indicates a the size of the data bus.

Microcontrollers can be classified as 8-bit, 16- bit or 32-bit based on the bus width. Higher bus widths often result in better performance. Microcontrollers can be divided into two types based on their memory structures ; Embedded memory and external memory. In case of embedded memory microcontrollers, the required data and program memory is embedded into the IC. Whereas external memory microcontrollers do not have program memory embedded on them and require an external chip for the same. Now a day, all microcontrollers are embedded memory microcontrollers. The classification based on instruction set is similar to that of a microprocessor. They can be either CISC (complex instruction set computer) or RISC (Reduced instruction set computer. Majority of microcontrollers follow CISC architecture with over 80 instructions. Microcontrollers can also be divided based on their computer architecture into von neumann and harvard.

Functions of different ICs used in the microcontroller kit

1 EPROM : 27256 (32k x 8 EPROM)

The micro -51 EB LCD has a standard EPROM configuration of 32KB. The address for the monitor EPROM is 0000-3FFF. EPROM expansion is C000-FFFF.

2 RAM : 61256 (256K x 16 BIT SRAM)

The micro - 51 EB LCD has 32 KB of read /write program / data memory using one 61256 whose addres is from 4000 to BFFFF. The micro - 51 EB LCD has one more 32KB of read/write data memory using one 61256 whose address is 0000-3FFF and C000-FEFF.

3 Parallel I/O interface : 8255 PPI (Programmable pheripheral interface)

Intel 8255 programmble pheripheral interface 24 programmbable I/O lines configured as three 8 bit ports direct bit set / reset capability. Three modes of operation namely basic I/O, strobed I/O and bidirectional bus.

4 RS485 Drivers and RS232 drivers : ICL 232 (RS232) and 74LBC184D (RS485)

8051 is used for serial communication with associated diver for interface immunity and overcoming attenuation.

5 Address Latch : (74LS273)

It is used to latch the address (A0-A7) from AD lines (AD0-AD7). The latch stores the number output by the 8051 from the databus. So that the LED can be lit with any 8 bit binary number.

6 Data bus buffer : (72LS244)

It connect 8 bit of input data to I/O pheripheral devices.

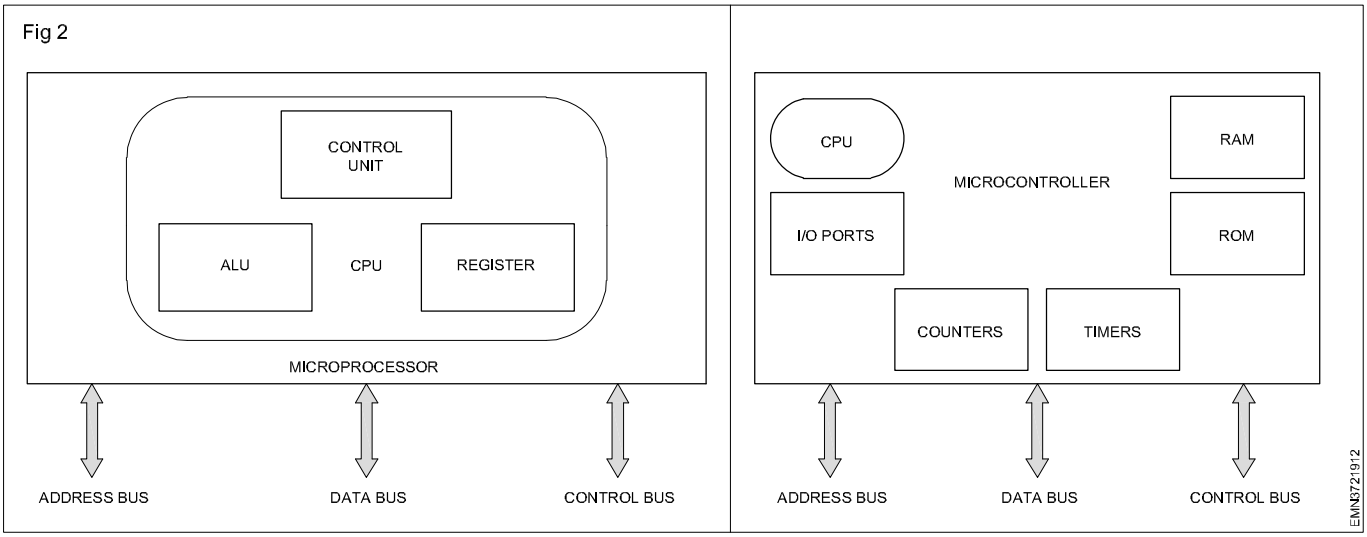
7 LCD interface and LCD module : (IC74174)

The LCD is diaplay is driven by both address latch and data bus buffer.

The following table shows some of the difference between microprocessors and microcontrollers.

Micropocessor	Microcontroller
Microprocessor assimilates the function of a central processing unit (CPU) on to a single integrated circuit (IC)	Microcontroller can be considered as a small computer which has a processor and some other components order to make it a micro computer chip.
Microprocessors are mainly used in designing general purpose systems from small to large and complex systems like super computers	Microcontrollers are used in automatically controlled devices
Microprocessors are basic components of personal computers	Microcontrollers are generally used in embedded system .
A microprocessor based system can perform numerous tasks	A microcontroller based system can perform single of very few tasks .
The main task of microprocessor is to perform the instruction cycle repeatedly. This includes fetch, decode and execute.	In addition to performing the tasks of fetch, decode and execute, a microcontroller also controls its environment based on the output of the instruction cycle.
In order to build or design a system (Computer, a microprocessor has to be connected externally to some other components like memory (RAM and ROM) and input output ports	The IC of a microcontroller has memory (both RAM, ROM) integrated on it along with some other components like I/O devices and timers
The overall cost of a sytem built using a microprocessor is high. This is because of the requirement of external components.	Cost of a system built using a microcontroller is less, all the components are readily available.
Generally power consumption and dissipation is high because of the external devices. Hence it requires external cooling system.	Power consumption is less
The clock frequency is very high usually in the order of Giga Hertz.	Clock frequency is less usually in the order of Mega Hertz.

Fig 2



Advantages of microcontroller : As the microcontroller is having on - chip I/O ports, timer/counters, code and data memory (limited) which reduces the program exeuction time. (Fig.2)

Comparative study of 8051 and 8052.

Although the 8051 microcontroller was introduced first with 8 bit capacity the enhanced version of 8051 been developed later which is called 8052 micro controller. The 8052 microcontroller is technically designed with certain additional features. The common features of these microcontrollers are given in Table - 1 and the diffrences are given in Table - 2 below:

Table - 2

Diffrences between 8051 and 8052

PARAMETERS	8051	8052
Internal RAM (DATA Memory)	128byte	256 Bye
Internal ROM (Code Memory)	4Kb	8Kb
Timer/Counter	2	3
No.of interrupt	5	6

Table - 1

Commom features of 8051 and 8052

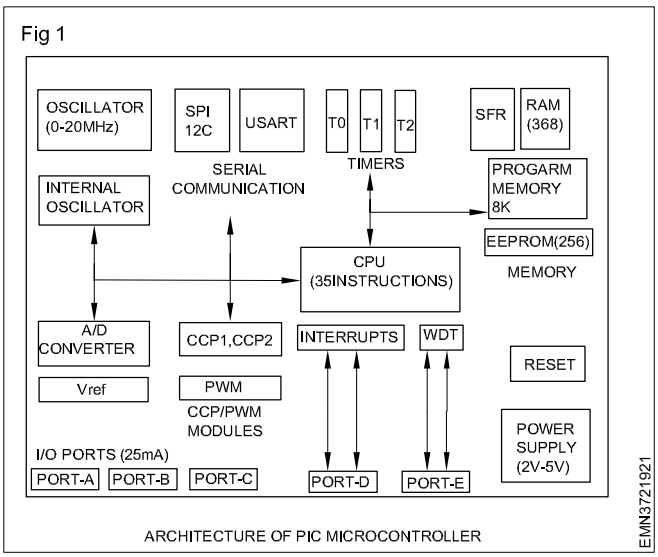
Sl.No.	Parameter	8051	8052
1.	Clocks instruction cycle	12	12
2.	UARTs/serial ports	1	1
3.	Maximum program size without external logic	64K	64K
4.	Maximum PIO port pins	32	32
5.	Size	8Bit	8Bit

Peripheral Interface Controller

Peripheral Interface Controller (PIC) is the world's smallest and very fast microcontroller that can be programmed to execute a vast range of tasks compare with other controllers. These programming and the simulated process of this microcontroller can be done by a circuit-wizard software. PIC microcontroller is an IC and its architecture comprises of CPU, RAM, ROM, timers, counters and protocols like SPI, UART, CAN which are used for interfacing with other peripherals. Microcontrollers are used for industrial purpose also the advantages of using this microcontroller includes low power consumption, high performance, supports hardware and software tools such as simulators, compilers, and debuggers.

The block diagram of PIC microcontroller architecture is shown in (Fig-1) comprises of central processing unit (CPU), I/O ports, A/D converter, memory organization, timers/counters, serial communication, interrupts, oscillator and CCP module that are discussed in given below:

Fig 1



ARCHITECTURE OF PIC MICROCONTROLLER

Architecture of PIC Microcontroller

Central Processing Unit (CPU)

PIC microcontroller's CPU is not different like other microcontroller CPU, which includes the ALU, controller unit, the memory unit, and accumulator. ALU is mainly used for arithmetic and logical operations. The memory unit is used to store the commands after processing. The control unit is used to control the internal & external peripherals, and the accumulator is used to store the final results and further processes.

Memory Organization

The memory module of the PIC microcontroller architecture consists of Random Access Memory (RAM) Read Only Memory (ROM) and STACK.

Memory Organization

Random Access Memory (RAM)

(RAM) The Random access memory is used to store the information temporarily in its registers. It is categorized into two banks, each bank has so many registers. The RAM registers are categorized into two types, namely Special Function Registers (SFR) General Purpose Registers (GPR).

General Purpose Registers (GPR)

As the name implies, These registers are used for general purpose only. For instance, if we want to multiply any two numbers by using this microcontroller. Usually, registers are used for multiplying and storing in other registers. So, GPR registers don't have any superior function,- CPU can simply access the data in the registers.

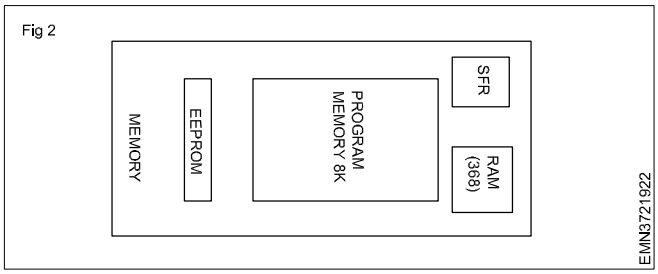
Special Function Registers (SFR)

As the name implies, SFRs are used only for special purposes. These registers work based on the function assigned to them, and these registers cannot work as a normal register. For instance, if you cannot use the STATUS register for storing the information, SFRs are used for viewing the status of the program. So, the user cannot change the SFR's function; the function is given by the manufacturer at the time of built-up.

Memory Organization

The memory organization of Peripheral Interface Controller (PIC) is shown in Fig 2 which includes the following:

Fig 2



- Read Only Memory (ROM)
- Electrically Erasable Programmable Read Only Memory (EEPROM)
- Flash Memory
- Stack

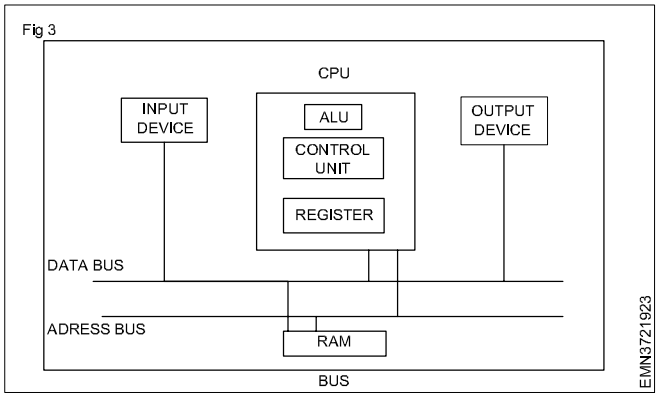
Input Output (I/O) Ports

The PIC microcontroller consists of 5-ports, namely Port-A, Port-B, Port-C, Port-D and Port-E.

BUS

BUS is used to transfer and receive the data from one peripheral to another as shown in Fig 3. It is categorized into two types as data bus and address bus.The Data Bus is used to transfer or receive the data.

Fig 3

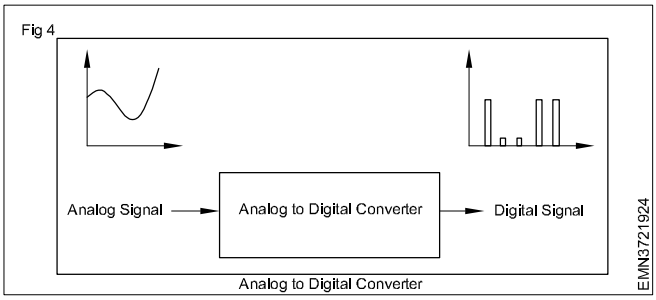


The address bus is used to transfer the memory address from the peripherals to the Central Processing Unit(CPU).Input /Output pins are used to interface the exterior peripherals; both the UART and USART are serial communication protocols, used to interface with serial devices such as GPS, GSM, IR, Bluetooth, etc.

Analog to Digital (A/D) Converter

A/D converter is shown in Fig 4 . It is used to convert analog voltage values to digital voltage values. An A/D module in PIC Microcontroller Controller comprises of 5-inputs for 28-pin devices and 8-inputs for 40-pin devices. The operation of the A/D converter is controlled by special registers like ADCON0 & ADCON1. The upper and lower bits of the converter are stored in registers like ADRESH and ADRESL. In this process, it needs 5V of an analog reference voltage.

Fig 4



Timer/Counters

PIC microcontroller has four-timer/counters wherein the one 8-bit timer and the remaining timers have the choice to select 8 or 16-bit mode. Timers are used for generating accuracy actions, for example, creating specific time delays between two operations.

Interrupts

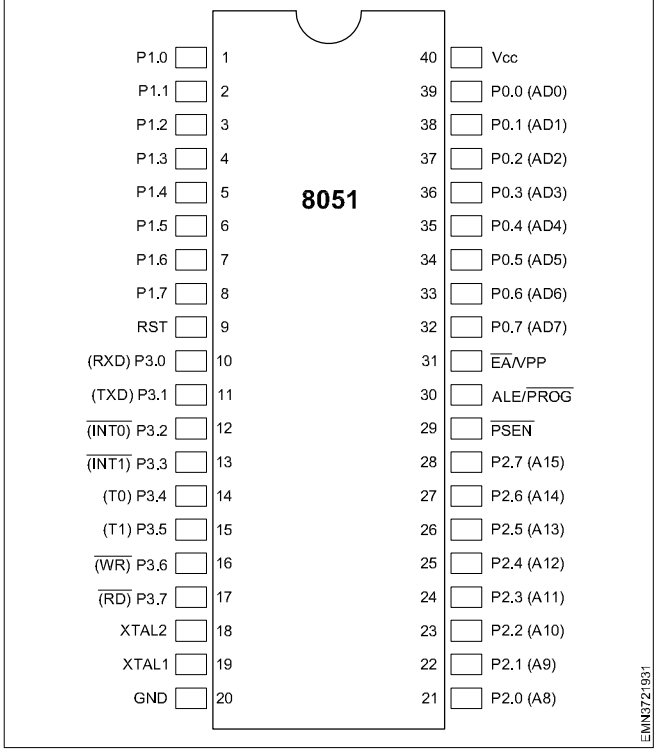
PIC microcontroller consists of 20 internal and 3-external interrupt sources which are allied with different peripherals like USART, ADC, Timers, and so on.

Pin details of 8051, Internal data memory, SFR and on-chip features

Objectives : At the end of this lesson you shall be able to
- **pin diagram of 8051**
- **data memory and special function registers**
- **utilization of on - chip resources such as ADC.**

The pin diagram of 8051: 8051 is a 40 pin microcontroller with I/O ports (Ref.Fig.1)

Fig 1



There are 4 ports in 8051 IC (Port 0, Port 1, Port 2 and Port 3) 32 pins are function as I/O port lines and 24 of these lines are dual purpose (P0, P1, P3). Each can operate as I/O, or as a control line or part of the address or data bus. Eight lines in each port can be used in interfacing to parallel devices like printers, DAC etc., or each line the port can be used in interfacing to single bit devices like LED's, switches, transistors, solenoid, motors and loudspeakers.

PORT 0 (32-39 Pins)

It is a dual purpose port (P0.0-P0.7). For simple design it is used as I/O ports. For larger design with external memory, it is used as multiplexed address and data bus (AD0-AD7)

PORT 1(1-8 Pins)

It is a dedicated I/O port (P1.0-P1.7). It is used only to inferface with the external devices.

PORT 2 (21-28 Pins)

It is a dual purpose port (P2.0-P2.7). It is used as I/O port or higher byte of address bus (A8-A15).

PORT 3 (10-17 Pins)

It is a dual purpose port (P3.0- P3.7) It is used as I/O port, or used to special features of 8051 (Table 1)

Table 1

BIT	Name	BIT Address	Alternate function
P 3.0	RXD	B0H	Receive data for serial port
P 3.1	TXD	B1H	Transmit data for serial port
P 3.2	INT0	B2H	External interrupt 0
P 3.3	INT1	B3H	External interrupt 1
P 3.4	T0	B4H	Timer /counter 0 external input
P 3.5	T1	B5H	Timer/counter 1 external input
P 3.6	WR	B6H	External data memory write stroke
P 3.7	RD	B7H	External data memory read stroke
P 1.0	T2	90H	Timer/ Counter 2 external input
P 1.1	T2EX	91 H	Timer /Counter 2 capacture / reload

RST (9 Pin No)

It is a master reset input pin. It should be kept high to start - up 8051.

On- chip oscillator (18-19 Pins)

It is a crystal oscillator with stabilizing capacitor connected to pin number 18 and 19. The normal crystal frequency is 12 MHz.

Power connection (20,40 Pins)

The 8051 operates at +5V DC. Pin No. 40 is Vcc. Pin No. 20 is Vss (GND)

PSEN (Program store enable) (29 Pin No)

PSEN is an output and control signal to enable the external memory.

ALE (Address Latch Enable) (30 Pin No)

ALE is an ouput signal to control demultiplexing the address and data bus. ALE signal oscillates at 2 MHz.

EA (External access) (31 Pin No)

It is an input signal is generally kept high (+5VDC) or low (GND). If EA is high 8051 executes program from internal ROM. If EA is low it executes program from external memory.

Internal data memory

128 Bytes of internal data memory is divided in to two parts, Part I is RAM (00- 7FH) Part II is special function registers (SFR) (80-FFH) Fig. 2

RAM

i) Register Bank (00H-1FH) 4 banks (Bank 0,1,2,3) Each bank consisting of 8 register (R0-R7)

ii) Bit addressable RAM (20 H-2FH)

iii) General purpose RAM (30 H-7FH)

i. Register banks

The bottom 32 locations of internal memory contain the register banks. The 8051 instruction set supports of 8 registers R0 through R7, and by default these registers are addresses 00H-07H.

ii. Bit addressable RAM

There are 128 general - purpose bit addressable locations at byte addresses 20 H through 2FH (8 bits /byte X 16 bytes = 128 bits). These addresses are accessed as bytes or as bits, depending on the instruction.

For example, to set bit 67H, the following instruction could be used.

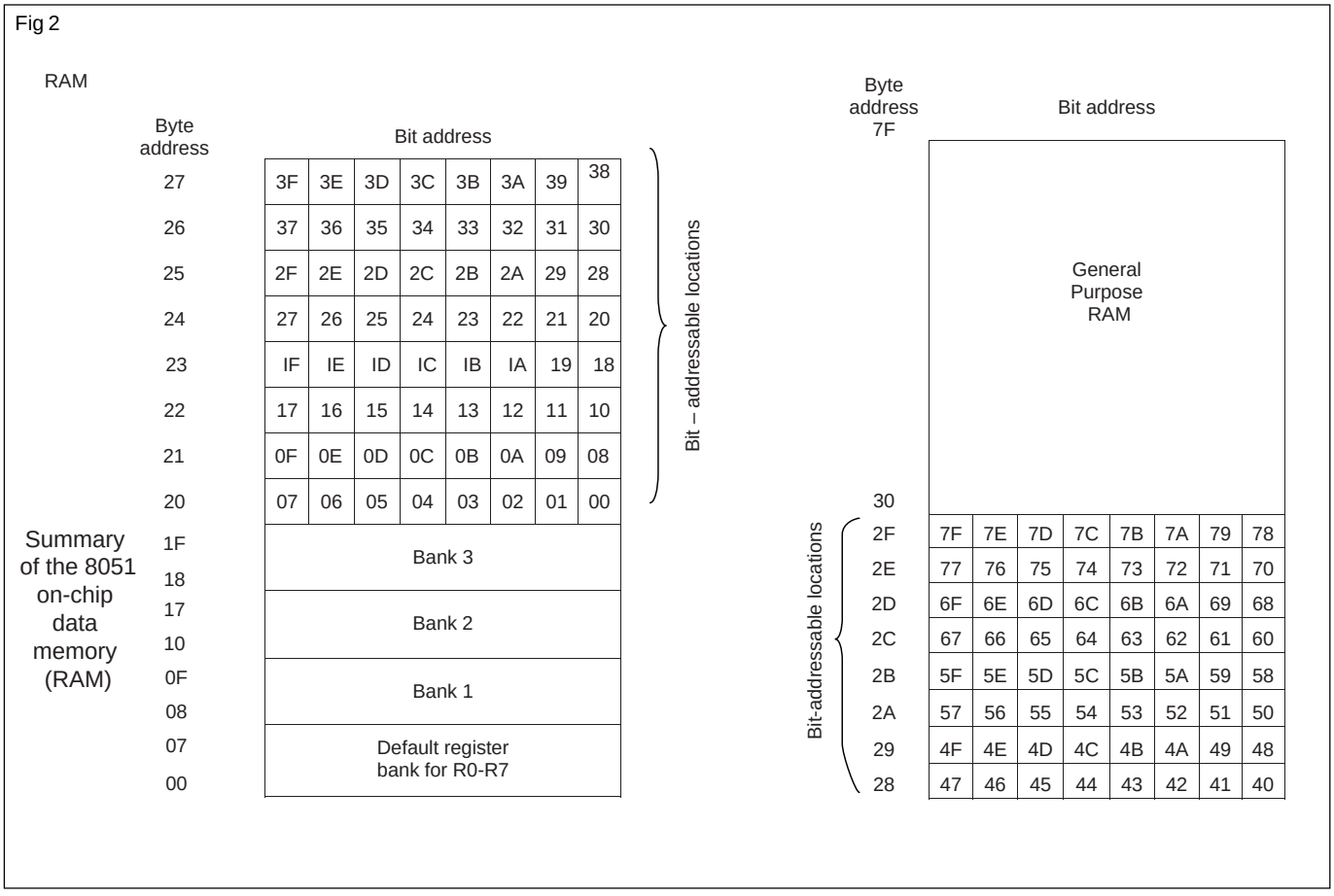
Set B 67H

Note that “Bit address 67H” is the most significant bit at “byte address 2CH”.

iii. General purpose RAM

General purpose RAM consisting of address location(30H-7FH) which is byte addressable available to programmer to store data /programs.

Fig 2



21 SFRs are available, out of 21, 11 are bit addressable and 10 are byte addressable

P0 (80 H) : Port 0, bit addressable

SP (81 H) : Stack pointer

DPL (82 H) : Data pointer low byte

DPH (83 H) : Data pointer high byte

PCON (87H) : Power control register

TCON (88H) : Timer control register, bit addressable

TMOD (89H) : Timer mode register

TL0 (8AH) : Timer 0 low byte

TL1 (8BH) : Timer 1 low byte

TH0 (8CH) : Timer 0 high byte

TH1 (8DH) : Timer 1 high byte

P1 (90H) : Port 1, bit addressable

SCON (98H) : Serial port control register, bit addressable

SBUF (99H) : Serial data buffer, byte addressable

P2 (A0H) : Port 2, bit addressable

IE (A8H) : Interrupt enable, bit addressable

P3 (B0H) :Port 3, bit addressable

IP (B8H) : Interrupt priority, bit addressable

PSW (D0H) : Program status word, bit addressable

ACC (E0H) : Accumulator, bit addressable

On-chip features of 8051 philips microcontroller

The derivative of 8051 philips microcontroller is most powerful 8 bit microcontroller. It has got an 8 bit CPU optimized for control application. 64 K program memory space, 64K data memory space, 4K bytes of on - chip program memory. 128 bytes of on - chip data memory, 32 bi-directional and individually addressable I/O lines, two 16 bit timer / counters, one full duplex serial port and 6 source /5- vector interruput with two priority level on - chip.

Utilization of on - chip resources such as ADC

The PCF 8591 is a single - chip, single - supply low - power 8 bit CMOS data acquisition, device with four analog inputs, one analog output and a serial I2C - bus interface. The functions of the device include analog input multiplexing, on- chip track and hold function, 8 bit analog-to-digital conversion and an 8- bit digital - to analog conversion. The maximum conversion rate is given by the maximum speed of the I2C- bus.

Features and benefits

- Single power supply
- Operating supply voltage 2.5V to 6.0V
- Low standby current
- Serial input and output via I2C- bus
- I2C address selection by 3 hardware address pins
- Max sampling ratte given by I2C- bus speed
- 4 Analog inputs configurable as single ended or differential inputs
- Auto- incremented channel selection
- Analog voltage range from VSS to VDD
- On - chip track and hold circuit
- 8-bit successive approximation A/D conversion
- Multiplying DAC with one analog output.

Applications

Supply monitoring

Reference setting

Instruction set of 8051, arithmatic and logical function

Objectives : At the end of this lesson you shall be able to
- **write program for adding, subtracting, multiplying and dividing two 8 bit numbers**
- **write program for logical and, or function for two 8 bit numbers.**

Assembly software for 8051

Here some simple assembly language programs for 8051 microcontroller are given to understand the operation of different instructions and to understand the logic behind particular program.

MOVC A, @ A+DPTR ; A ← ext_code_mem (A+DPTR)

MOVC A, @ A+PC ; A ← ext_code_mem (A+PC)

8051 Instruction set

The 8051, 8-bit microcontroller family instruction set includes 111 instructions, 49 of which are single - byte, 45 two - byte and 17 three - byte instruction. The instruction opcode format consists of a function mnemonic followed by a destination & source operand field. The instruction set is divided into four functional groups.

- Data transfer
- Arithmetic
- Logic
- Control transfer

i. Data transfer instructions

Data transfer operations are divided into three classes:

- General - purpose
- Accumulator - specific
- Address - object

None of these operations affects the PSW flag settings except a POP or MOV directly to the PSW.

Examples

MOV A, # 45 - Immediate addressing mode

MOV A, R1 - Register addressing mode

MOV 45h, A - Direct addressing mode

MOV @ R1, 32 h - Indirect addressing mode

ii. Arithmetic instructions

The MCS - 51 family microcontrollers have four basic mathematical operations. Only 8- bit operations using unsigned arithmetic are supported directly. The overflow flag, however, permits the addition and subtraction operation to serve for both unsigned and signed binary integers. Arithmetic can also be performed directly on packed BCD representations.

Examples

ADD A, #84 - Immediate addressing mode

SUBB A, R2 - Register addressing mode

ADD 73h, a - Direct addressing mode

ADDC @R1, 25h - Indirect addressing mode

iii. Logic instructions

The MCS - 51 family microncontrollers perform basic logic operations on both bit and byte operands.

Bit level (single operand) operations

In 8051 internal RAM and SFRs can be addressed by the address of each bit within a byte. This bit addressing is very convenient when we wish to alter a single bit of a byte. The ability to operate on individual bits creates the need for the area of RAM that contains data addresses that hold a single bit. The bit addresses are numbered from 00H to 7FH to represent the 128d bit addresses that exist from byte addresses 20H to 2FH.

CLR sets a or any directly addressable bit to zero (0)

SETB sets and directly bit - addressable bit to one (1).

CPL is used to complement the contents of the A register without affecting any flag, or any directly addressable bit location.

RL, RLC, RR, RRC, SWAP are the five operations that can be performed on A. RL, rotate left, RR, rotate right, RLC, rotate left through carry, RRC, rotate right through carry and SWAP, rotate left four. Four RLC and RRC and CY flag become equal to the last bit rotated out. SWAP rotates A left four places to exchange bits 3 through 0 with bits 7 through 4.

Byte level (two operand) operations

ANL performs bits wise logical AND of two operands (for both bit and byte operands) and returns the result to the location of the first operand.

ORL performs bit wise logical OR of two source operands (for both bit and byte operand) and returns the result to the location of the first operand.

XRL performs logical exclusive OR two source operands (byte operands) and returns the result to the location of the first operand.

Example

ANL A, #45h - Immediate addressing mode

ORL A, R2 - Register addressing mode

XRL 52h, A - Direct addressing mode

ANL @R3, 65h - Indirect addressing mode

iv. Control transfer instructions

There are three classes of control transfer operations: unconditonal calls, returns, jumps, conditional jumps, and interrupts. All control transfer operations, some upon a specific condition, cause the program execution to continue a non - sequential location in program memory.

Example

CJNE A, #22H, loop - Immediate addressing mode

DJNZ R1, loop - Register addressing mode

DJNZ 30H, loop - direct addressing mode

JMP @A + DPTR - Indirect addressing mode

Notes on data addressing modes

Rn- Working register R0-R7

Direct - 128 Internal RAM locations, any I/O port, control or status register

@Ri - Indirect internal or external RAM location addressed by register R0 or R1

#data - 8-bit constant included in instruction

#data 16- 16- bit constant included as bytes 2 and 3 of instruction.

bit - 128 software flags, any bit - addressable I/O pin, control or status bit

A - accumulator

Notes on program addressing modes

addr16- Destination address for LCALL and LJMP may be anywhere within the 64-kbyte program memory address space.

addr11- Destination address for ACALL and AJMP will be within the same 2- kbyte page of program memory as the first byte of the following instruction.

rel - SJMP and all conditional jumps include an 8-bit offset byte. Range is +127/- 128 byts relative to the first byte of the following instruction.

Instruction set summary

Mnemonic		**Description**	**Byte**	**Cycle**
Arithmetic operations				
ADD	A, Rn	Add register to accumulator	1	1
ADD	A, direct	Add direct byte to accumulator	2	1
ADD	A @Ri	Add indirect RAM to accumulator	1	1
ADD	A, # data	Add immediate data to accumulator	2	1
ADDC	A, Rn	Add register to accumulator with carry flag	1	1
ADDC	A, direct	Add direct byte to A with carry flag	2	1
ADDC	A, @ Ri	Add indirect RAM to A with carry flag	1	1
ADDC	A, # data	Add immediate data to A with carry flag	2	1
SUBB	A, Rn	Subtract register from A with borrow	1	1
SUBB	A, direct	Subtract direct byte from A with borrow	2	1
SUBB	A, @Ri	Subtract indirect RAM from A with borrow	1	1
SUBB	A, #data	Subtract immediate data from A with borrow	2	1
INC	A	Increment accumulator	1	1
INC	Rn	Increment register	1	1
INC	direct	Increment direct byte	2	1
DEC	@Ri	Increment indirect RAM	1	1
DEC	A	Decrement accumulator	1	1
DEC	Rn	Decrement register	1	1
DEC	direct	Decrement direct byte	2	1

DEC	@Ri	Decrement indirect RAM	1	1
INC	DPTR	Increment data pointer	1	2
MUL	AB	Multiply A and B	1	4
DIV	AB	Divide A by B	1	4
DA	A	Decimal adjust accumulator	1	1

Logic Operations		**Description**	**Byte**	**Cycle**
ANL	A, Rn	AND register to accumulator	1	1
ANL	A, direct	AND direct by the to accumulator	2	1
ANL	A, @Ri	AND indirect RAM to accumulator	1	1
ANL	A, @data	AND immediate data to accumulator	2	1
ANL	direct, A	AND accumulator to direct byte	2	1
ANL	direct, #data	AND immediate data to direct byte	3	2
ORL	A, Rn	OR register to accumulator	1	1
ORL	A, direct	OR direct byte to accumulator	2	1
ORL	A, @Ri	OR indirect RAM to accumulator	1	1
ORL	A, #data	OR immediate data to accumulator	2	1
ORL	direct, A	OR accumulator to direct byte	2	1
ORL	direct, #data	OR immediate data to direct byte	3	2
XRL	A, Rn	Exclusive OR register to accumulator	1	1
XRL	A, direct	Exclusive OR direct byte to accumulator	2	1
XRL	A, @Ri	Exclusive OR indirect RAM to accumulator	1	1
XRL	A, #data	Exclusive OR immediate data to accumulator	2	1
XRL	direct, A	Exclusive OR accumulator to direct byte	2	1
XRL	direct, #data	Exclusive OR immediate data to direct byte	3	2
CLR	A	Clear accumulator	1	1
CPL	A	Complement accumulator	1	1
RL	A	Rotate accumulator left	1	1
RLC	A	Rotate accumulator left through carry	1	1
RR	A	Rotate accumulator right	1	1
RRC	A	Rotate accumulator right through carry	1	1
SWAP	A	Swap nibbles within the accumulator	1	1

Data transfer		Description	Byte	Cycle
MOV	A, Rn	Move register to accumulator	1	1
MOV	A, direct	Move direct byte to accumulator	2	1
MOV	A, @Ri	Move indirect RAM to accumulator	1	1
MOV	A, #data	Move immediate data to accumulator	2	1
MOV	Rn, A	Move accumulator to register	1	1
MOV	Rn, direct	Move direct byte to register	2	2
MOV	Rn, #data	Move immediate data to register	2	1
MOV	direct, A	Move accumulator to direct byte	2	1
MOV	direct, Rn	Move register to direct byte	2	2
MOV	direct, direct	Move direct byte to direct byte	3	2
MOV	direct, @Ri	Move indirect RAM to direct byte	2	2
MOV	direct, #data	Move immediate data to direct byte	3	2
MOV	@Ri, A	Move accumulator to indirect RAM	1	1
MOV	@Ri, direct	Move direct byte to indirect RAM	2	2
MOV	@Ri, #data	Move immediate data to indirect RAM	2	1
MOV	DPTR, #data 16	Load data pointer with a 16 - bit constant	3	2
MOV	A, @A+DPTR	Move code byte relative to DPTR to accumulator	1	2
MOVC	A, @A, +PC	Move code byte relative to PC to accumulator	1	2
MOVX	A, @Ri	Move external RAM (8-bit addr.) to A	1	2
MOVX	A, @DPTR, A	Move A to external RAM (16-bit addr.)	1	2
PUSH	direct	Push direct byte onto stack	2	2
XCH	A, Rn	Exchange register with accumulator	1	1
XCH	A, direct	Exchange direct byte with accumulator	2	1
XCH	a, @Ri	Exchange indirect RAM with accumulator	1	1
XCHD	A, @Ri	Exchange low- order nibble indir. RAM with A	1	1

Boolean variable manipulation

Mnemonic		Description	Byte	Cycle
CLR	C	Clear carry flag	1	1
CLR	bit	Clear direct bit	2	1
SETB	C	Set carry flag	1	1
SETB	bit	Set direct bit	2	1
CPL	C	Complement carry flag	1	1
CPL	bit	Complement direct bit	2	1

Mnemonic		Description	Byte	Cycle
ANL	C, bit	AND direct bit to carry flag	2	2
ANL	C, /bit	AND complement of direct bit to carry	2	2
ORL	C, bit	OR direct bit to carry flag	2	2
ORL	C, /bit	OR complement of direct bit to carry	2	2
MOV	c, bit	Move direct bit to carry flag	2	1
MOV	bit, C	Move cary flag to direct bit	2	2

Program and Machine control

Mnemonic		Description	Byte	Cycle
ACALL	addr16	Absolute subroutine call	3	2
LCALL	addr16	Long subroutine call	3	2
RET	-	Return from subroutine	1	2
RETI	-	Return from interrupt	1	2
AJMP	addr16	Absolute jump	3	2
LJMP	addr16	Long jump	3	2
SJMP	rel	Short jump (relative addr.)	3	2
JMP	@A + DPTR	Jump inidrect relative to the DPTR	1	2
JZ	rel	Jump if accumulator is zero	2	2
JNZ	rel	Jump if carry flag is not zero	2	2
JC	rel	Jump if carry flag is set	2	2
JNC	rel	Jump if carry flag is not set	2	2
JB	bit, rel	Jump if direct bit is set	3	2
JNB	bit, rel	Jump if direct bit is set	3	2
JBC	bit, rel	Jump if direct bit is set and clear bit	3	2
CJNE	A, direct, rel	Compare direct byte to A and jump if not equal	3	2
CJNE	A, #data, rel	Compare immediate to A and jump if not equal	3	2
CJNE	Rn, #data, rel	Compare immed. to reg. and jump if not equal	3	2
CJNE	@Ri, #data, rel	Compare immed. to ind. and jump if not equal	3	2
DJNZ	Rn, rel	Decrement register and jump if not zero	2	2
DJNZ	direct, rel	Decrement direct byte and jump if not zero	3	2
NOP		No operation	1	1

Program 1: 16 - bit addition

Objective

To perform 16-bit additoin of the two 16-bit data using immediate addressing and store the result in memory.

Theory :

As there is only one 16- bit register in 8051, 16-bit addition is performed by using ADDC instruction twice, i.e adding LSD first and MSD next.

Example

The program is to add the 16-bit data 1234 with the data 5678 and store the result at the locations 4150 and 4151 using immediate addressing.

Result : (4150) = AC (LSB); (4151) = 68 (MSB)

DATAL1 = 34; DATAL2 = 78

DATAM1 = 12; DATAM2 = 56

DATAM1 - MSD OF DATA1,

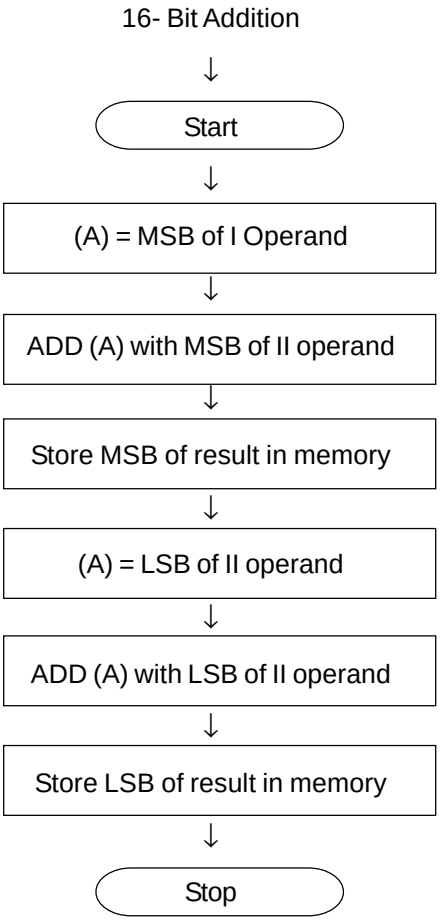
DATAM2 - MSD OF DATA2,

DATA1 - LSD OF DATA1,

DATA2 - LSD OF DATA 2,

Flowchart:

Program for 16 bit addition (refer manual)



Program :

```
CLR     C
MOV     A, #DATAL1
ADDC    A, #DATAL2
MOV     DPTR, # 4150
MOVX    @DPTR, A
INC     DPTR
MOV     A, #DATAM1
ADDC    A, #DATAM2
MOVX    @DPTR, A
HLT : SJMP  HLT
```

Object codes

Memory addreses	Object codes	Mnemonics
4100	C3	CLR C
4101	74	MOV A,#DATAL1
4102	34	
4103	34	ADDC. A, #DATAL2
4104	78	
4105	90	MOV DPTR, #4150
4106	41	
4107	50	
4108	F0	MOVX @DPTR, A
4109	A3	INC DPTR
410A	74	MOV A, #DATAM1
410B	12	
410C	34	ADDC A, #DATAM1
410D	56	
410E	F0	MOVX @DPTR,A
410F	80	
4110	FE	HERE, SJMP HERE

Program 2 - 8 bit subtraction

Objective

To perform subtraction of two 8-bit data using immediate addressing and store the result in memory.

Theory

Using the accumulator, subtraction is performed and the result is stored. Immediate addressing is employed. The SUBB instruction writes the result in the accumulator.

Example

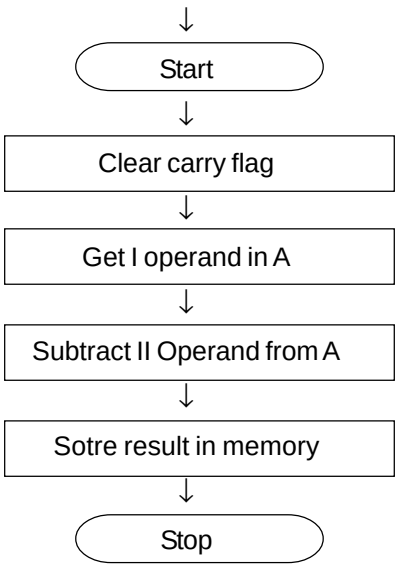
Sample data : DATA1 = 20

DATA 2 = 10

Result : (4500) = 10

Flow chart

8 - bit subtraction




```
CLR        C
MOV        A, #DATA1
SUBB       A, #DATA2
MOV        DPTR, #4500
MOVX       @DPTR, A
Here : SJMP Here
```

Object codes

Memory addreses	Object codes	Mnemonics
4100	C3	CLR C
4101	74	MOV A,#DATAL1
4102	20	
4103	94	SUBB A, #DATAL2
4104	10	
4105	90	MOV DPTR, #4500
4106	45	
4107	00	
4108	F0	MOVX @DPTR, A
4109	80	Here: SJMP here
410A	FE	

Procedure

i Enter the op codes and data in the trainer

ii Execute the program and verify for results

iii Change data and see that correct results are obtained.

Exercises

i. Subtract the contents of location 4500 from the contents of location 4501 and store the result at location 4600.

Sample data : (4500) = 56

(4501) = 6A

Result : (4600) = 14

ii. Perform the same subtraction using two's complement addition.

Program 3 - 8 bit multiplication

Objective

To obtain the product of two 8-bit data using immediate addressing and store the result in memory.

Theory

The 8051 has a "MUL" instruction unlike many other 8-bit processors. MUL instruction multiplies the unsigned eight - bit integers in A and B. The low - order byte of the product is left in A and the high- order byte in B. If the product is > 255, the overflow flag is set. Otherwise it is cleared. The carry flag is always cleared.

Example

Let us multiply the contents of registers A and B and store the 16-bit result at locations 4500 and 4501.

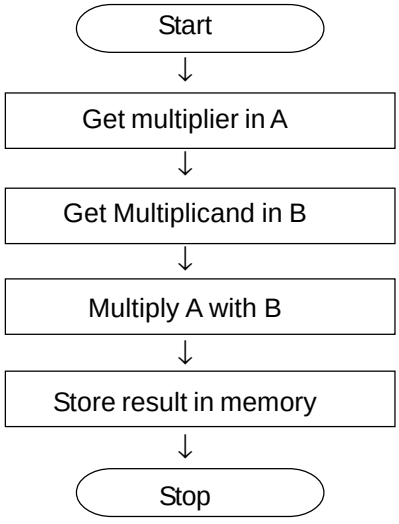
Sample data : DATA 1= 0A

DATA2 = 88

(4500) = 50 (LSB)

(4501) = 05 (MSB)

8-bit Multiplication



Program

```
MOV        A, #DATA1
MOV        B, #DATA2
MUL        AB
MOV        DPTR, #4500
MOVX       @DPTR, A
```

INC DPTR

MOV A, B

MOVX @DPTR, A

Here : SJMP Here

Object codes

Memory address	Object codes	Mnemonics
4100	74	MOV A,#DATAL1
4101	0A	
4102	75	MOV A,#DATAL2
4103	F0	
4104	88	
4105	A4	MUL AB
4106	90	MOV DPTR, # 4500
4107	45	
4108	00	
4109	F0	MOVX @DPTR, A
410A	A3	INC DPTR
410B	E5	MOV A, B
410C	F0	
410D	F0	MOVX, @DPTR, A
410E	80	Here : SJMP here
410F	FE	

Procedure

i. Enter the above opcode from 4100

ii. Execute the program; see that the result is stored correctly.

iii. Change data and check if the results are correct each time.

Exercises

i. Obtain the square of a number stored in memory

Sample : (4500) = 0A

Result : (4600) = 64

ii. Obtain the fourth power of 08 using MUL instruction and store the result in memory.

Result : (4500) = 10 (MSB)

(4501) = 00 (LSB)

iii. Do a decimal multi - byte addition in 32-bit and store the result in memory.

Data : (4500) = 04 - Count

(4551) = 99 - First number

(4552) = 99

(4553) = 99

(4554) = 99

(4561) = 99 - Second number

(4562) = 99

(4563) = 99

(4564) = 99

Result : (4570) = 98

(4571) = 99

(4572) = 99

(4573) = 99

(4574) = 01

Program 4 - 8 bit division

Objective

To divide an 8-bit number by another 8-bit number and store the quotient and remainder in memory.

Theory

The 8051 has a “DIV” instruction unlike many other 8-bit processors. DIV instruction divides the unsigned eight - bit integer in A by unsigned 8-bit integer in register B. The accumulator receives the integer part of the quotient and register B receives the integer remainder. The carry and flags will be cleared.

Example

Let the divisor and dividend be in registers B and A respectively.

Data : DATA 1 = 65 - Dividend

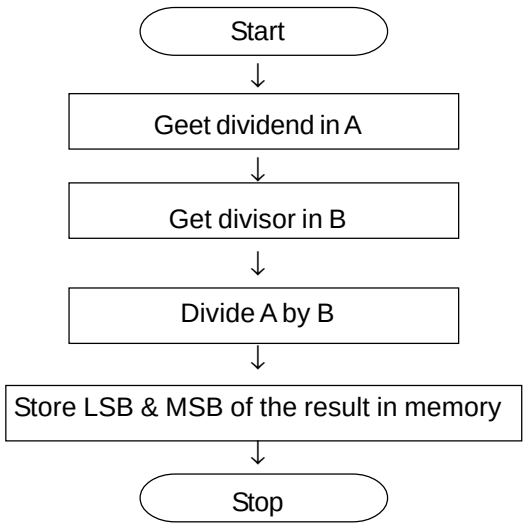
DATA2 = 08 - Divisor

Result : (4500) = 0C - Quotient

(4501) = 05= Remainder

Flow Chart

8 bit by 8- bit division



Program

```
MOV   A, #DATA1 ; Load Acc. with dividend
MOV   B # DATA2 ; Load Reg. B with divisor
DIV   AB
MOV   DPTR, # 4500
MOVX  @DPTR, A ; Store quotient at 4500
INC   DPTR
MOV   A, B ; Store remainder at 4501
MOVX  @DPTR, A
HLT : SJMP HLT
```

Object codes

Memory address	Object codes	Mnemonics
4100	74	MOV A,#DATAL1
4101	65	
4102	75	MOV B,#DATAL2
4103	F0	
4104	08	
4105	84	DIV AB
4106	90	MOV DPTR, # 4500
4107	45	
4108	00	
4109	F0	MOVX @DPTR, A
410A	A3	INC DPTR
410B	E5	MOV A, B
410C	F0	
410D	F0	MOVX, @DPTR, A
410E	80	Here : SJMP here
410F	0E	
4110	41	

Procedure

i. Enter the opcodes and the data in the trainer

ii. Execute the program and check for results

iii. Change data and check for the corresponding results.

Discussion

One's complement is nothing but the logical operation 'NOT' In this example, the CPL instruction has been employed . Since the two's complement of a number is its one's complement +1, by INC instruction has been employed. It can also be performed by adding 1 to one's complement number by using ADD instruction.

Exercise

i) Obtain the one's and two's complement of the data 77 and store it in memory.

Result : One's complement (4500) = 88

Two's complement (4501) = 89

Program 5

Objective

Perform OR function of an 8 bit number

Theory

Setting bits can be done by ORing that particular bit by1. The following program explains how to set a particualr bit in an 8-bit number by using the ORL instruction of 8051

Example

In the following program, the contents of the accumulator is ORed with an immediate data accordingly to set the required bits.

Sample data : DATA1 = 2F

DATA2 = 45

Result : (4500) = 6F

Program

```
MOV       A, # DATA 1
ORL       A, # DATA 2
MOV       DPTR, # 4500
MOVX      @DPTR, A
Here : SJMP   Here
```

Object codes

Memory address	Object codes	Mnemonics
4100	74	MOV A,#DATA1
4101	2F	
4102	44	ORL A,#DATA 2
4103	45	
4104	90	MOV DPTR, #4500
4105	45	
4106	00	
4107	F0	MOVX @DPTR, A
4108	80	Here; SJMP here

Procedure

i. Enter the opcodes and execute the program

ii. Check whether the corresponding bits are set accordingly.

Discussion

The ORL instruction can be used to set a particular bit in the command or control registers of peripherals interfaced with the processor and can also be used to determine if the status read from peripherals is as expected.

Exercises

i. Set alternate bits in an 8-bit number, starting from bit 0 and store the result at location 4200 in memory.

ii. Store successive powers of 2 from 0 to 7 in consecutive memory locations starting from 4300.

Result : (4300) = 01

(4301) = 02

(4302) = 04

(4303) = 08

(4304) = 10

(4305) = 20

(4306) = 40

(4307) = 80

Program 6

Objective

To perform AND function of an 8 bit number.

Theory

The ANL instruction of 8051 can be used to reset bits. AND ing with zero prodcues a cleared bit. ANDing with one does not change the status of the bit.

Example

To mask bits 0 and 7, the 8- bit data has to be ANDed with 7E, which is 01111110 in binary.

Sample data : DATA 1 = 87

DATA 2 = 7E

Result : (4500) = 06

Program

```
MOV        A, #DATA 1
ANL        A, # DATA 2
MOV        DPTR, # 4500
MOVX       @DPTR, A
Here : SJMP Here
```

Object codes

Memory address	Object codes	Mnemnics
4100	74	MOV A,#DATA1
4101	87	
4102	54	ANL A, #DATA 2
4103	7E	
4104	90	MOV DPTR, #4500
4105	45	
4106	00	
4107	F0	MOVX @DPTR, A
4108	80	Here; SJMP here
4109	08	
410A	41	

Procedure

i. Enter the opcodes from 4100 and execute the program.

ii. Check whether the result is 06 at 4500

Discussion

The ANL instruction can be used to check whether a particular status is reached in the peripheral device just like the ORL instruction. The other logical instruction available in the instruction set of 8051 is the XRL (Exclusive -OR). The CLR (clear operand) instruction is also a logical instruction which can be used to initialize registers in counter operations.

Timer on the microcontroller kit

Objectives : At the end of this lesson you shall be able to
- **Explain the function of timer is 8051**
- **design a delay program using timer in microcontroller kit.**

The 8051 microcontroller has two independent 16 bit up counting timers named timer 0 and timer 1 and this article is about generating time delays using the 8051 timers. Generating delay using pure software loops have been already discussed here but such delays are poor in accuracy and cannot be used in sensitive applications. Delay usng timer is the most accurate and surely the best method.

A timer can be generalized as a multi - bit counter which increments / decrements itself on receiving a clock signal and produces an interrupt signal up on roll over. When the counter is running on the processor's clock, it is called a "Timer", which counts a predefiend number of processor clock pulses and generates a programmable delay. When the counter is running on an external clock source (may be periodic or aperiodic external signal) it is called a "counter" itself and it can be used for counting external events.

In 8051, the oscillator output is divided by 12 using a divide by 12 network and then fed to the timer as the clock signal. That means for an 8051 running at 12 MHz, the timer clock input will be 1 MHz. That means the timer advances once in every 1μS and the maximum time delay possible using a single 8051 timer is (2^{16}) x (1μS) = 65536μS. Delays longer than this can be implemented by writing up a basic delay program using timer and then looping it for a required number of time. We will see all these in detail in next following sections.

Designing a delay program using 8051 timers

While designing delay programs in 8051, calculating the initial value that has to be loaded in to TH and TL registers forms a very important thing. Let us see how it is done.

Assume the processor is clocked by a 12MHz crystal.

That means, the timer clock input will be 12MHz/12=1MHz

That means, the time taken for the timer to make one increment = 1/1MHz= 1μs.

For a time delay of "X" μS the timer has to make "X" increments.

2^{16}= 65536 is the maximim number of counts possible for a 16 bit timers.

Let TH be the value that has to be loaded to TH registed and TL be the value that has to be loaded to TL register.

Then, THTL = Hexadecimal equivalent of (65536-X) where (65536-X) is considered in decimal.

Example

Let the required delay be 1000 μS =(ie; 1mS)

That means X = 1000

65536- X = 65536 - 1000 = 64536

645536 is considered in decimal and converting it to hexadecimal gives FC18

That menas THTL = FC18

Therefore TH=FC and TL = 18

Program for generating 1mS delay using 8051 timer

The program shown below can be used for generating 1mS delay and it is written as a subroutine so that you can call it anywhere in the program. Also you can put this in a loop for creating longer time delays (multiples of 1ms). Here timer 0 of 8051 is used and it is operating in MODE1 (16 bit timer).

```
Delay : MOV TMOD< #0000001B/ Sets timer 0 to MODE1 (16 bit timer). Timer 1 is not used
MOV TH0, #0FCH // TH0 register with FCH
MOV TL0, #018H // Loads TL0 register with 18H
SETB TR0 // Starts the timer 0
Here : JNB TF0, Here// Loops here until TF is set (ie ; until rool over)
CLR TR0 // Stops timer 0
CLR TF0 // Clear TF0 flag
RET
```

The above delay routine can be looped twice in order to get a 2mS delay and it is shown in the program below.

```
Main : MOV R6, #2D
LOOP : ACALL DELAY
DJNZ R6, LOOP
SJMP Main
Delay : MOV TMOD, #00000001B
MOV TH0, #0FCH
MOV TL0, #018H
SETB TR0
Here : JNB TF0, here
CLR TR0
```

CLR TF0

RET

Few points to remember while using timers

- Once timer flag (TF) is set,the programmer must clear it before it can be set agian
- The timer does not stop after the timer flag is set. The programmer must clear the TR bit in order to stop the timer.
- Once the timer overflows, the programmer must reload the initial start values to the TH and TL registers to begin counting up from.
- We can configure the desired timer to create an interrupt when the TF flag is set.
- IF interrupt is not used, then we have to check the timer flag (TF) is set using some conditional branching instruction.
- Maximum delay possible using a single 8051 timer is 65536μS and minimum is 1 μS provided that you are using a 12MHz crystal for clocking the microcontroller.

Application of 8051 (motor, traffic control)

Objectives : At the end of this lesson you shall be able to
- **explain the application of 8051 microcontroller**
- **design the circuit to control of DC motor using 8051.**

Application of 8051 microcontroller

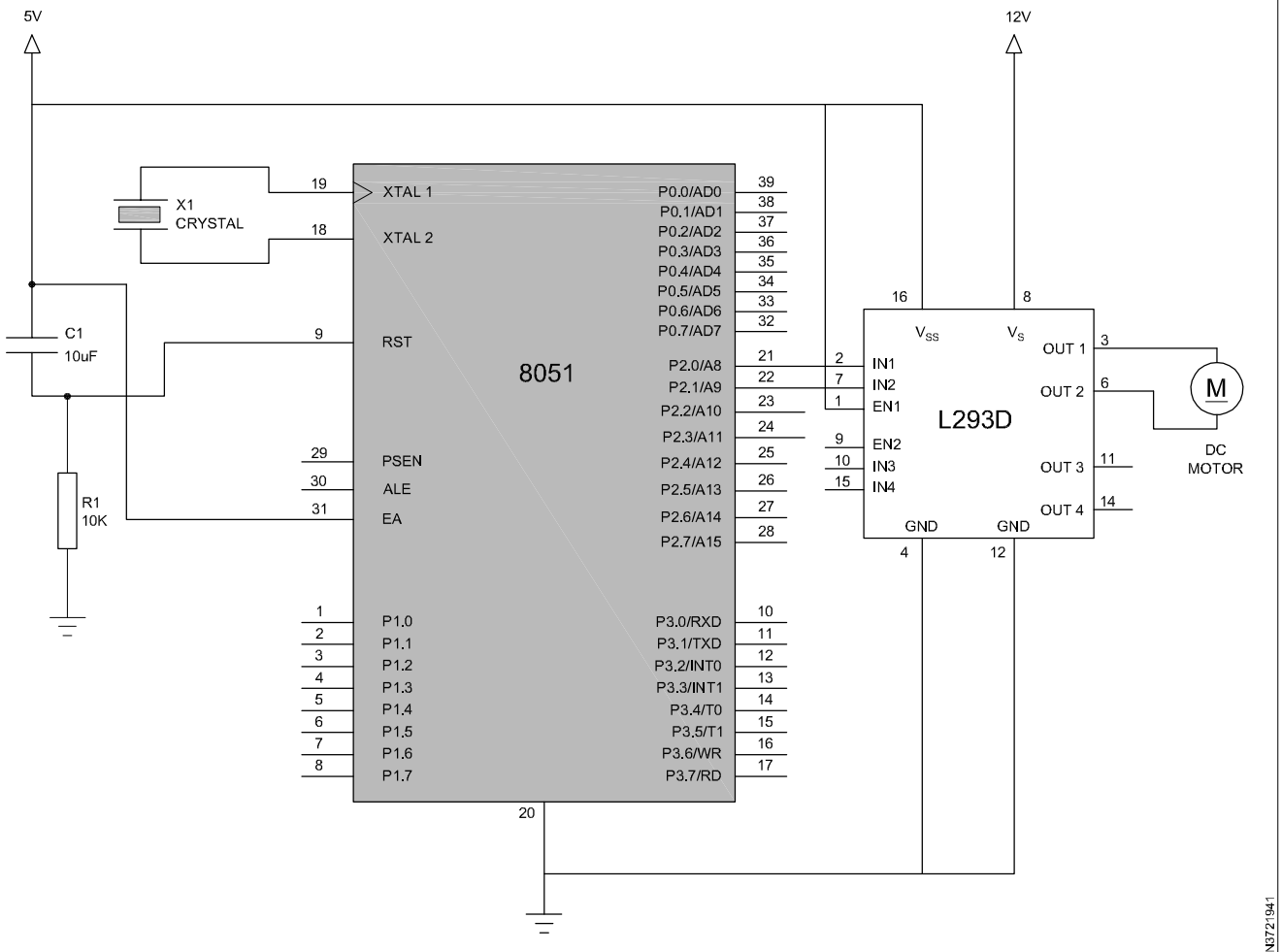
A microcontroller is a versatile chip which can be used in various fields starting from simple consumer electronics to high end medical, automobile and defence application also. So now a day the microcontrollers are found in every walk of life.

Interfacing DC motor to 8051 using L293D

A DC motor runs in response to the applied DC current. It prodcues torque by using both electric and magnetic field.

The DC motor has rotor, stator, field magnet brushes, shaft, commutator etc., The DC motor required large currents of the order of 400 mA for its rotation. But this much amount of current cannot be generated by the ports of the microcontroller. So if it is direclty connect the DC motor to the ports of the conroller it may draw high current for its operation from the port and hence the microcontroller may be damaged. So we use a driving circuit along with opto isolator provides an additional protection to the microcontroller from large currents. (fig)1

Fig 1



Assembly lanuage program to control DC motor using 8051

ORG	0000H		Remarks
Main	Set B	P1.2	
	MOV	P1, # 00000001B	Motor runs in clockwise
	ACALL	Delay	
	MOV	P1, #00000010B	Motor runs in anticlockwise
	ACALL	Delay	
	SJMP	Main	Motor rotates continuously in clockwise for some time and anticlockwise for some time
Delay	MOV	R4, # FFH	Load R4 register with FF
	MOV	R3, #FFH	Load R3 register with FF
LOOP1	DJNZ	R3, LOOP1	Decrement R3 until it is zero
LOOP2	DJNZ	R4, LOOP2	Decrement R4 until it is zero
	RET		Return to the main program

Traffic light control

Traffic lights, which may also be known as stop lights, traffic lamps, traffic signals, signal lights, robots or semaphore, are signaling devices positioned at road intersections, pedestrian crossings and other locations to control competing flows of traffic.

Interfacing traffic light with 8051

The traffic light controller section consists of 12 Nos. point LEDS are arranged by 4 lanes in PS/8051 trainer kit. Each lane has go (green), listen (yellow) and stop (red) LED is being placed (Refer fig.2).

About the colors of traffic light control

Traffic lights alternate the right of way of road users by displaying lgihts of a standard color (red, yellow/amber, and green), using a universal color code (and a precise sequence to enable comprehension by those who are color blind). In the typical sequence of colored lights.

Illumination of the green light allows traffic to proceed in the direction denoted.

Illumination of the yellow/ amber light denoting, if safe to do so, prepare to stop short of the intersection, and

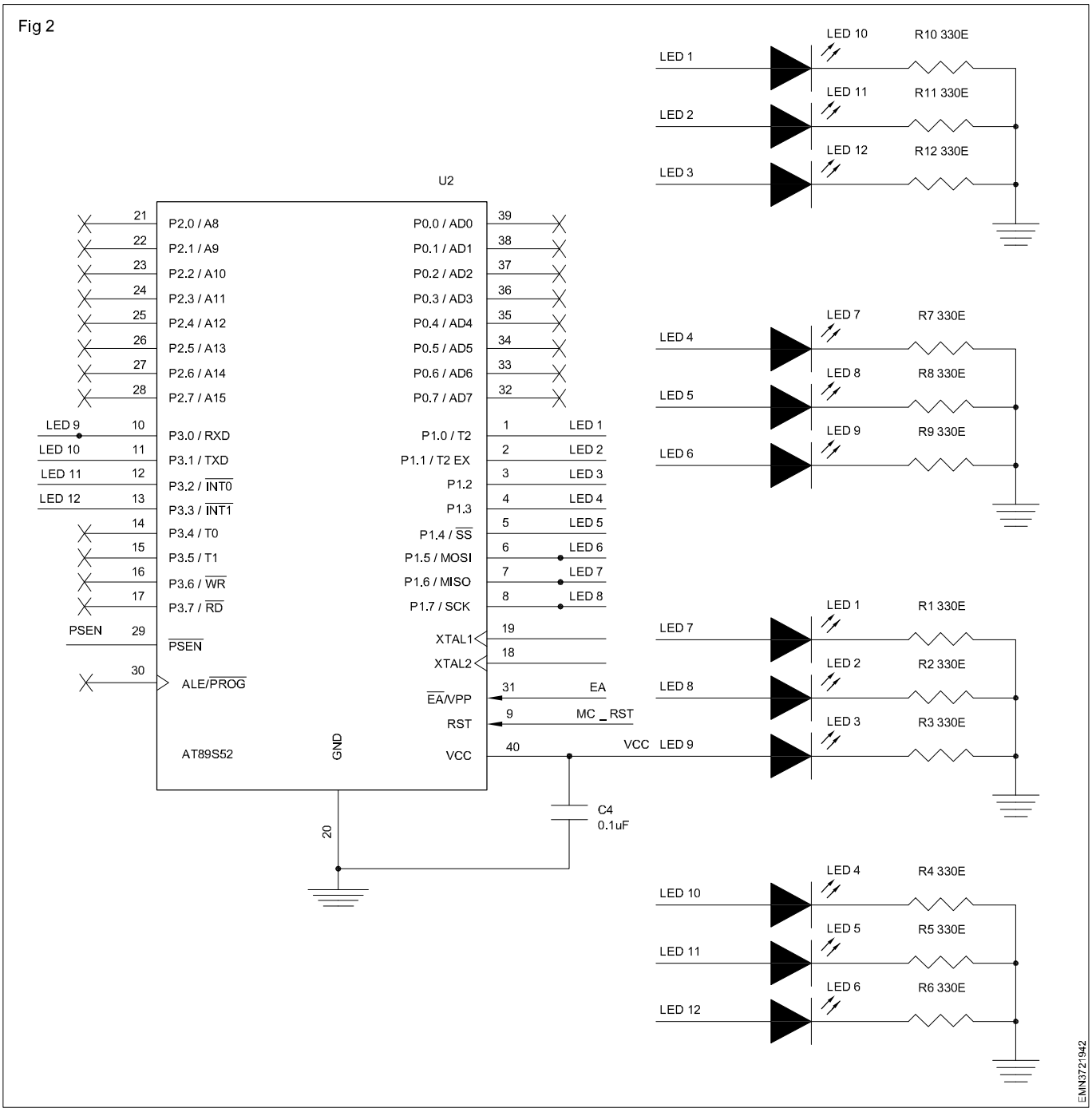
Illumination of the red signal prohibits any traffic from proceeding.

Usually, the red light contains some orange in its hue, and the green light contains some blue, for the benefit of people with red - green color blindness, and “green” lights in many areas are in fact blue lenses on a yellow light (which together appear green)

PIN assignment with 8051

LAN direction	8051 lines	Modules
South	P1.0 P1.1 P1.2	Go Listen Stop
East	P1.3 P1.4 P1.5	Go Listen Stop
North	P1.6 P1.7 P3.0	Go Listen Stop
West	P3.1 P3.2 P3.3	Go Listen Stop
PWR	13-16 17,19 18, 20	NC Vcc GND

Fig 2



Assembly program to interface traffic light

Opcode **Mnemonics**

Title :

Program to inferface

Traffic light with 8051

CNTL PORT : 4003

PORT A : 4000

PORT B : 4001

Memory Address	Opcode	Mnemonics
8500	90 85 45	Start : MOV DPTR, # TRE
8503	7A 0C	MOV R2, #0C
8505	E0	MOVX @DPTR, A
8506	C0 83	PUSH DPH
8508	C0 83	PUSH DPL
850A	90 40 03	MOV DPTR, #CNTL PORT
850D	F0	MOVX @DPTR, A
850E	D0 82	POP DPL
8510	D0 82	POP DPL
8512	A3	INC DPTR
8513	E0	LOOP 1: MOVX @DPTR, A
8514	C0 83	PUSH DPH
8516	C0 82	PUSH DPL
8518	90 40 00	MOV DPTR, # PORTA
851B	F0	MOVX @ DPTR, A
851C	D0 82	POP DPL
851E	D0 83	POP DPH
8520	A3	INC DPTR
8521	E0	MOVX @DPTR, A
8522	C0 83	PUSH DPH
8524	C0 82	PUSH DPL

Memory Address	Opcode	Mnemonics
8526	90 40 01	MOV DPTR, #PORT B
8529	F0	MOVX @DPTR, A
852A	12 85 36	LCALL DELAY
852D	D0 82	POP DPL
852F	D083	POP DPH
8531	A3	INC DPTR
8532	DA DF	DJNZ R2, LOOP 1
8534	80 CA	SJMP START
8536	7F 10	DELAY: MOV R7, # 10H
8538	7D FF	LOOP P3, MOV R6, # 0FFH
853C	00	LOOP2 : NOP
853D	00	NOP
853E	DE FC	DJNZ R6, LOOP2
8540	DD F8	DJNZ R5, LOOP3
8542	DF F4	DJNZ R7, LOOP4
8544	22	RET

TRE : 8545

8545 21H, 09H, 10H, 00H (South way)

8549 0CH, 09H, 80J, 00H (East way)

854D 64H, 08H, 00H, 04H (North way)

8551 24H, 03H, 02H, 00H (West way)

8555 End

Note : The schematics sections given is, traffic light connected to port 1 and port 3 the sample program is given based on 8255