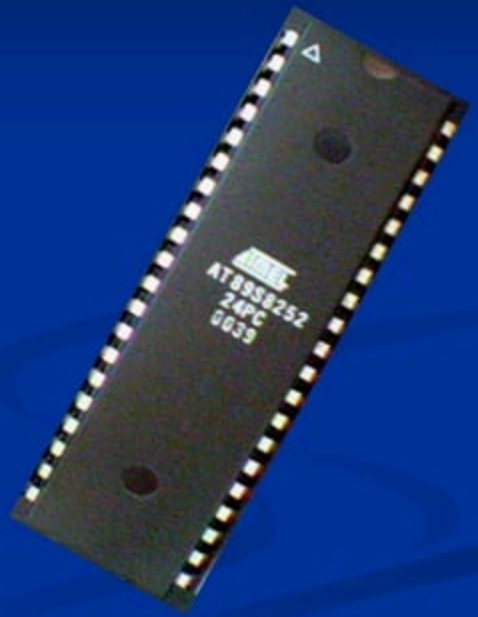
Overview

- Introduction
- Block Diagram and Pin Description of the 8051
- Registers
- Memory mapping in 8051
- Stack in the 8051
- I/O Port Programming
- Timers
- Interrupts &
- Applications



Why do we need to learn Microcontrollers ?

- Its not an exaggeration if I say that, today there is no electronic gadget on the earth which is designed without a Microcontroller. Ex: communication devices, digital entertainment, portable devices etc…

Not believable ??? See the next slide

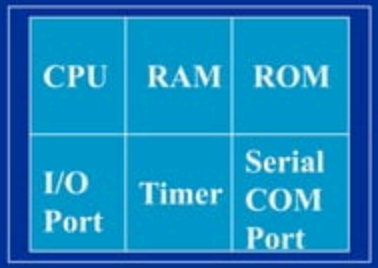
- Personal information products: Cell phone, pager, watch, pocket recorder, calculator
- Laptop components: mouse, keyboard, modem, fax card, sound card, battery charger
- Home appliances: door lock, alarm clock, thermostat, air conditioner, TV remote, VCR, small refrigerator, exercise equipment, washer/dryer, microwave oven
- Industrial equipment: Temperature/pressure controllers, Counters, timers, RPM Controllers
- Toys: video games, cars, dolls, etc.

So, A good designer should always know what type of controller he/she is using ,their architecture, advantages , disadvantages , ways to reduce production costs and product reliability etc.

O.K ????

Then What is a Microcontroller ?

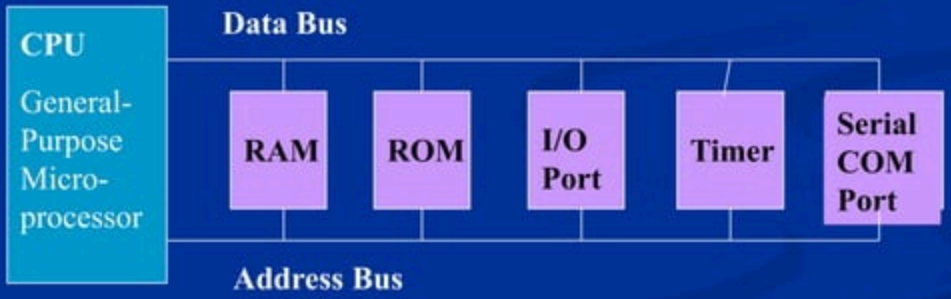
- A smaller computer
- On-chip RAM, ROM, I/O ports...
- Example：Motorola's 6811, Intel's 8051, Zilog's Z8 and PIC 16X



How is it different from a Microprocessor ??

- General-purpose microprocessor

  CPU for Computers
  No RAM, ROM, I/O on CPU chip itself
  Example：Intel's x86, Motorola's 680x0



Microprocessor vs. Microcontroller

Microprocessor

- CPU is stand-alone, RAM, ROM, I/O, timer are separate
- designer can decide on the amount of ROM, RAM and I/O ports.
- expansive
- versatility
- general-purpose

Microcontroller

- CPU, RAM, ROM, I/O and timer are all on a single chip
- fix amount of on-chip ROM, RAM, I/O ports
- Highly bit addressable
- for applications in which cost, power and space are critical
- single-purpose

EVOLUTION

Flashback !!!!

In the year 1976, Motorola created a Microprocessor chip called 6801 which replaced its brother 6800 with certain add-on chips to make a computer. This paved the way for the new revolution in the history of chip design and gave birth to a new entity called

MICROCONTROLLER.

The INTEL bagged the credit of producing the first Microcontroller 8048 with a CPU and 1K bytes of EPROM, 64 Bytes of RAM an 8-Bit Timer and 27 I/O pins in 1976.

contd

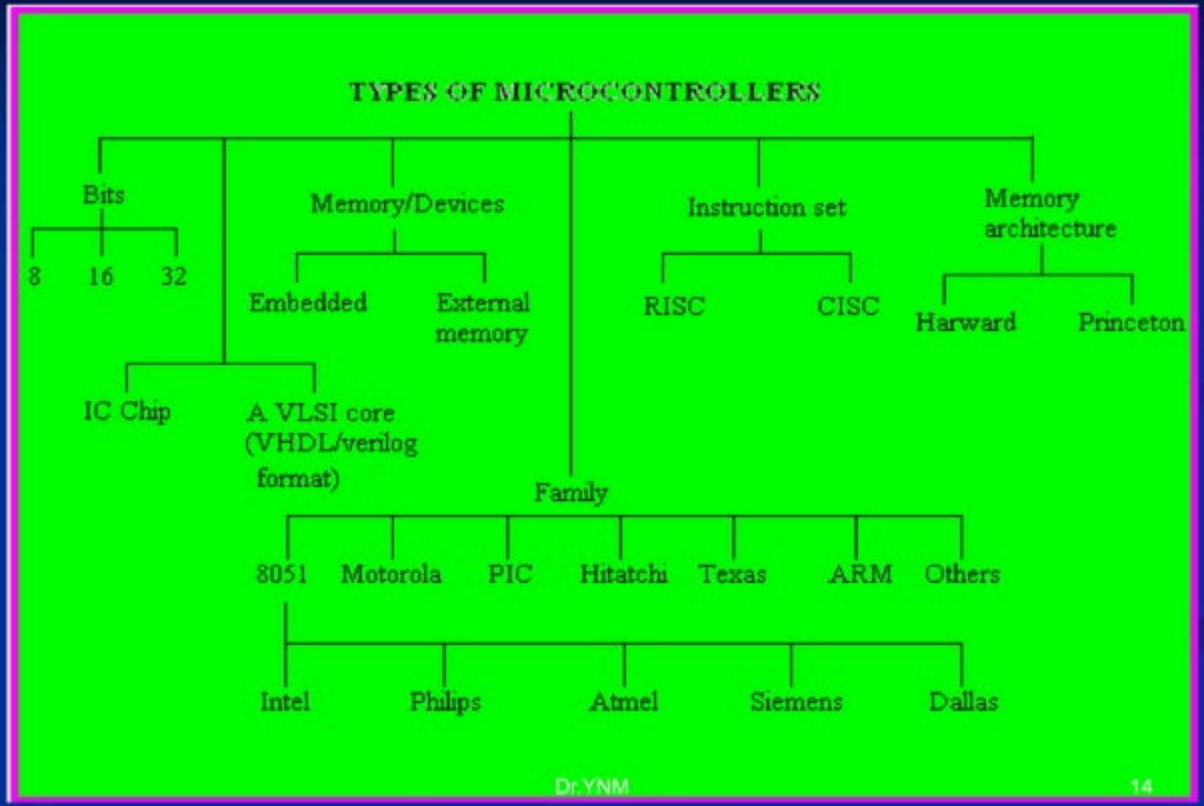
- Then followed the most popular controller 8051 in the year 1980 with 4K bytes of ROM,128 Bytes of RAM , a serial port, two 16-bit Timers , and 32 I/O pins.
- The 8051 family has many additions and improvements over the years and remains a most soughtafter tool for todays circuit designers.
- The same INTEL introduced a 16 bit controller 8096 in the year 1982

- Later INTEL introduced 80c196 series of 16-bit microcontrollers for mainly industrial applications
- Microchip, another company has introduced a microcontroller PIC 16C64 an 8-bit in the year 1985.
- 32-bit microcontrollers have been developed by IBM and Motorola-MPC 505 is a 32-bit RISC controller of Motorola
- The 403 GA is a 32 -bit RISC embedded controller of IBM

ARM Controllers

- In recent times ARM company (Advanced Risc machines) has developed and introduced 32 bit controllers which are highend application devices,especially communication devices like mobiles , ipods etc..(Refer www.arm.com)

Types of Microcontrollers



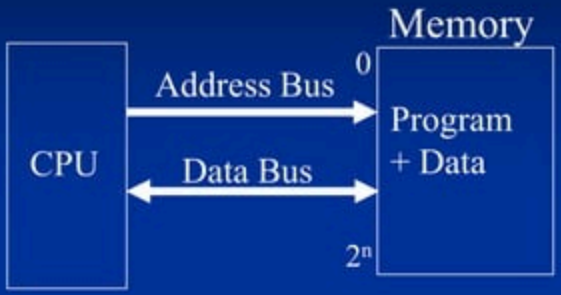
Microcontrollers from different manufacturers

- Atmel
- ARM
- Intel
  - 8-bit
    - 8XC42
    - MCS48
    - **MCS51**
    - 8xC251
  - 16-bit
    - MCS96
    - MXS296
- National Semiconductor
  - COP8
- Microchip
  - 12-bit instruction PIC
  - 14-bit instruction PIC
    - PIC16F84
  - 16-bit instruction PIC
- NEC
- Motorola
  - 8-bit
    - 68HC05
    - 68HC08
    - 68HC11
  - 16-bit
    - 68HC12
    - 68HC16
  - 32-bit
    - 683xx
- Texas Instruments
  - TMS370
  - MSP430
- Zilog
  - Z8
  - Z86E02

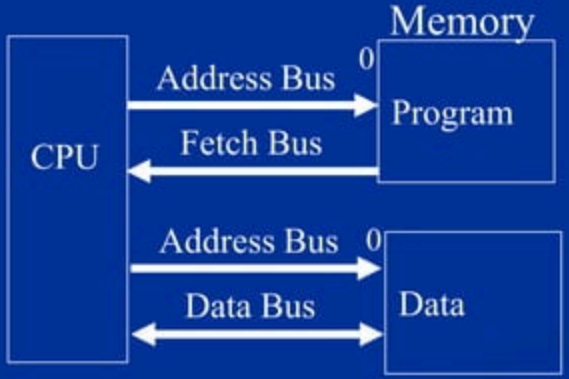
MCS-51 "Family" of Microcontollers

Feature	8031	8051	8052	8751
ROM	NO	4kB	8kB	4kB UV Eprom
RAM (Bytes)	128	128	256	128
TIMERS	2	2	3	2
I/O PINS	32	32	32	32
SERIAL PORTS	1	1	1	1
INTERRUPT SOURCES	6	6	8	6

Microcontroller Architectures



Von Neumann Architecture

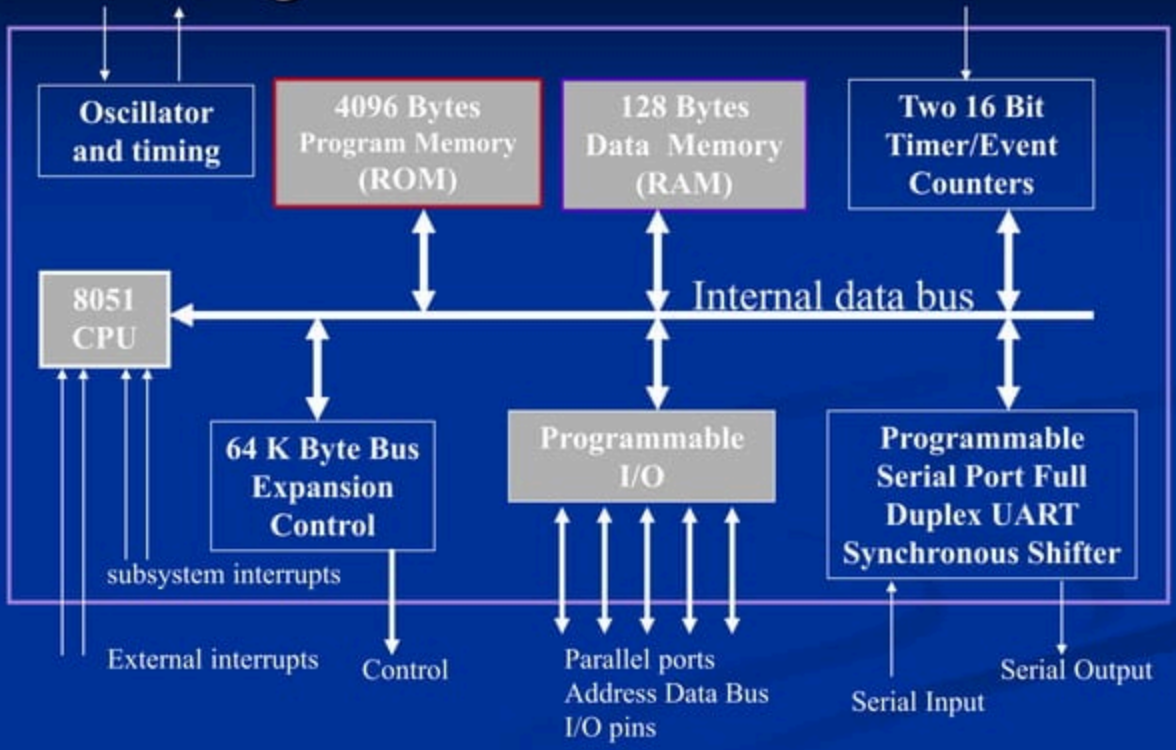


Harvard Architecture

Important Features of 8051

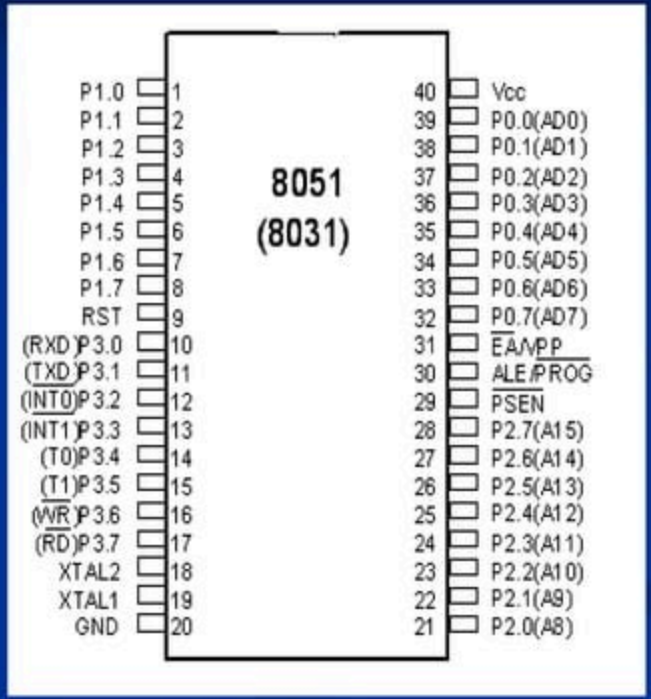
- 4K bytes ROM
- 128 bytes RAM
- Four 8-bit I/O ports
- Two 16-bit timers
- Serial interface
- 64K external code memory space
- 64K data memory space

"Original" 8051 Microcontroller

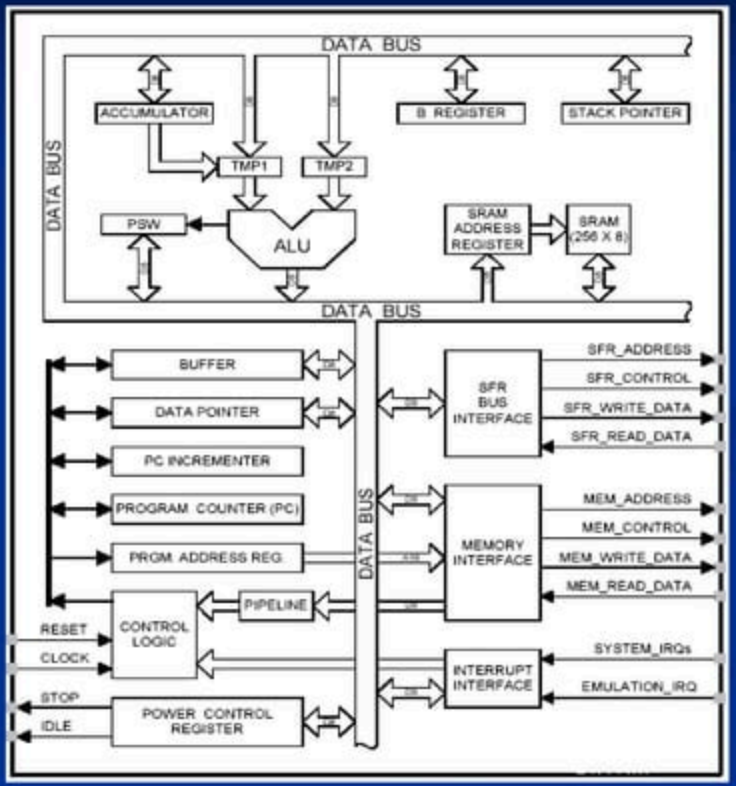


Pin Description of the 8051

- The 8051 is a 40 pin device, but out of these 40 pins, 32 are used for I/O.
- 24 of these are dual purpose, i.e. they can operate as I/O or a control line or as part of address or date bus.



8051 CPU Registers



A (8-bit Accumulator)

B (8-bit register for Mul &Div)

PSW (8-bit Program Status Word)

SP (8-bit Stack Pointer)

PC (16-bit Program Counter)

DPTR (16-bit Data Pointer)

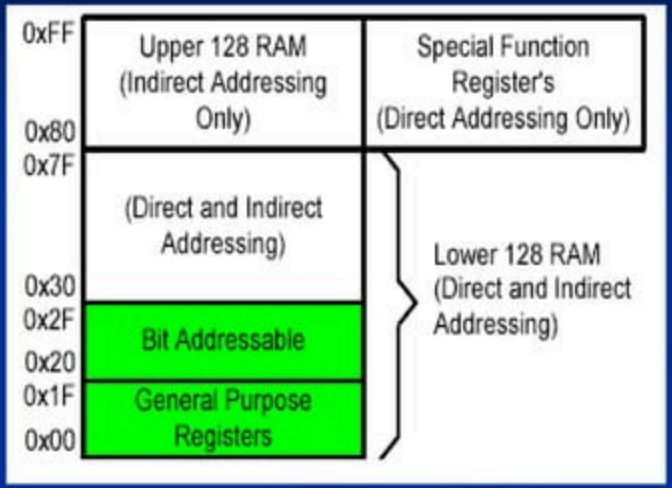
Special Function Registers

DATA registers

CONTROL registers

- Timers
- Serial ports
- Interrupt system
- Analog to Digital converter
- Digital to Analog converter

etc..



Addresses 80h – FFh

Direct Addressing is used to access SFRs

List of Registers
(*Denotes the SFRs)

ACC *	Accumulator	0E0H
B *	B Register	0F0H
PSW *	Program Status Word	0D0H
SP	Stack Pointer	81H
DPTR	Data Pointer (2 Bytes)	
DPL	Low Byte	82H
DPH	High Byte	83H
P0 *	Port 0	80H
P1 *	Port 1	90H
P2 *	Port 2	0A0H
P3 *	Port 3	0B0H

Contd…

IP *	Interrupt Priority Control	0B8H
IE *	Interrupt Enable Control	0A8H
TMOD	Timer/Counter Mode Control	89H
TCON *	Timer/Counter Control	88H
TH0	Timer/Counter 0 High Byte	8CH
TL0	Timer/Counter 0 Low Byte	8AH
TH1	Timer/Counter 1 High Byte	8DH
TL1	Timer/Counter 1 Low Byte	8BH

PSW REGISTER

The PSW register contains several status bits that reflect the current state of the CPU.

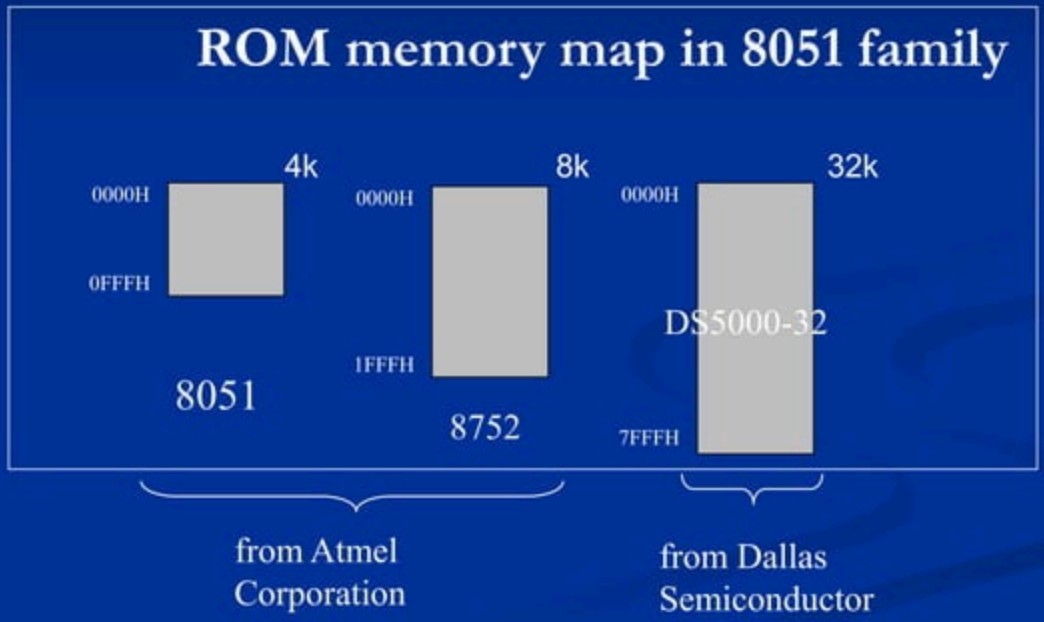
CY	AC	F0	RS1	RS0	OV	—	P

CY	PSW.7	Carry Flag
AC	PSW.6	Auxiliary Carry Flag
FO	PSW.5	Flag 0 available to the user for general purpose
RS1	PSW.4	Register Bank selector bit 1
RS0	PSW.3	Register Bank selector bit 0
OV	PSW.2	Overflow Flag
—	PSW.1	User definable flag
P	PSW.0	Parity flag. Set/cleared by hardware each instruction cycle to indicate an odd/even umber of '1' bits in the accumulator.

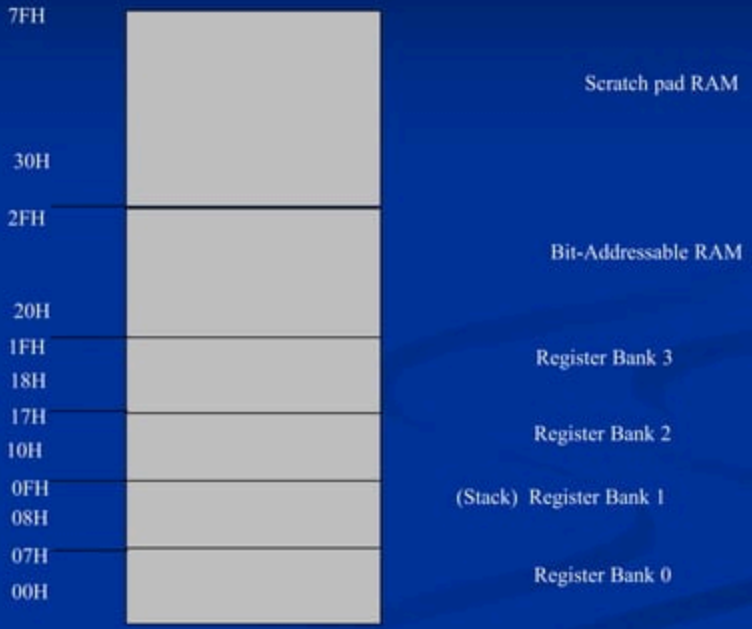
With RS1 and RS0 bits we can select the corresponding register bank.

RS1	RS0	Register Bank	Address
0	0	0	00H-07H
0	1	1	08H-0FH
1	0	2	10H-17H
1	1	3	18H-1FH

Memory mapping in 8051



RAM memory space allocation in the 8051



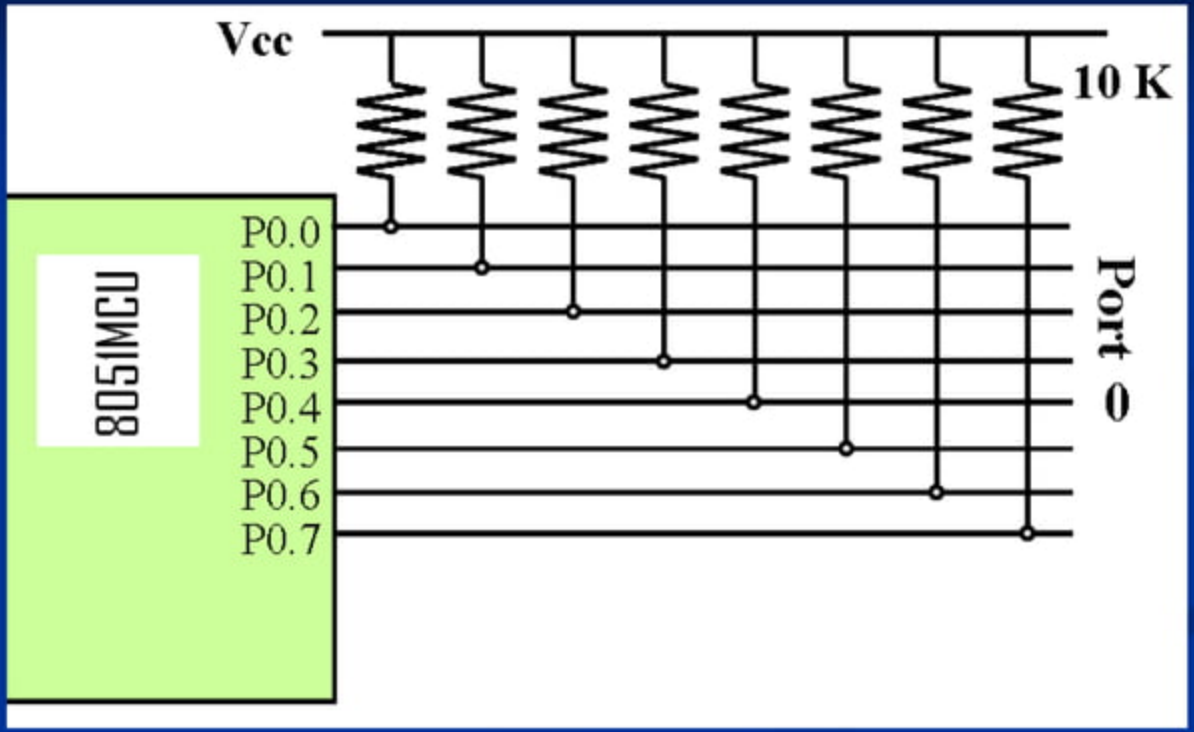
PORTS OF 8051

- 8051 has 4 Ports. Port 0, Port1, Port2 , Port3
- Port 0 is a dual purpose port, it is located from pin 32 to pin 39 (8 pins). To use this port as both input/output ports each pin must be connected externally to a 10 k ohm pull-up resistor.This is because Port 0 is an open drain.

Simple ex:

```
BACK    MOV A, #22
        MOV P0 ,A
        ACALL DELAY
        CPL A
        SJMP BACK
```

Port 0 with Pull-Up Resistors



Ports..

- Port 1 is a dedicated I/O port from pin 1 to pin 8.Upon reset it is configured as outport. It is generally used for interfacing to external device thus if you need to connect to switches or LEDs, you could make use of these 8 pins,but it doesn't need any pull-up resistors as it is having internally
- Like port 0, port 2 is a dual-purpose port.(Pins 21 through 28) It can be used for general I/O or as the high byte of the address bus for designs with external code memory.Like P1 ,Port2 also doesn't require any pull-up resistors

Ports contd…

- Port 3 is also dual purpose but designers generally avoid using this port unnecessarily for I/O because the pins have alternate functions which are related to special features of the 8051. Indiscriminate use of these pins may interfere with the normal operation of the 8051.
- However, for a programmer, it is the same to program P0, P1, P2 and P3.
- All the ports upon RESET are configured as **output**. To use any of the ports as an input port,it must be set(Programmed)

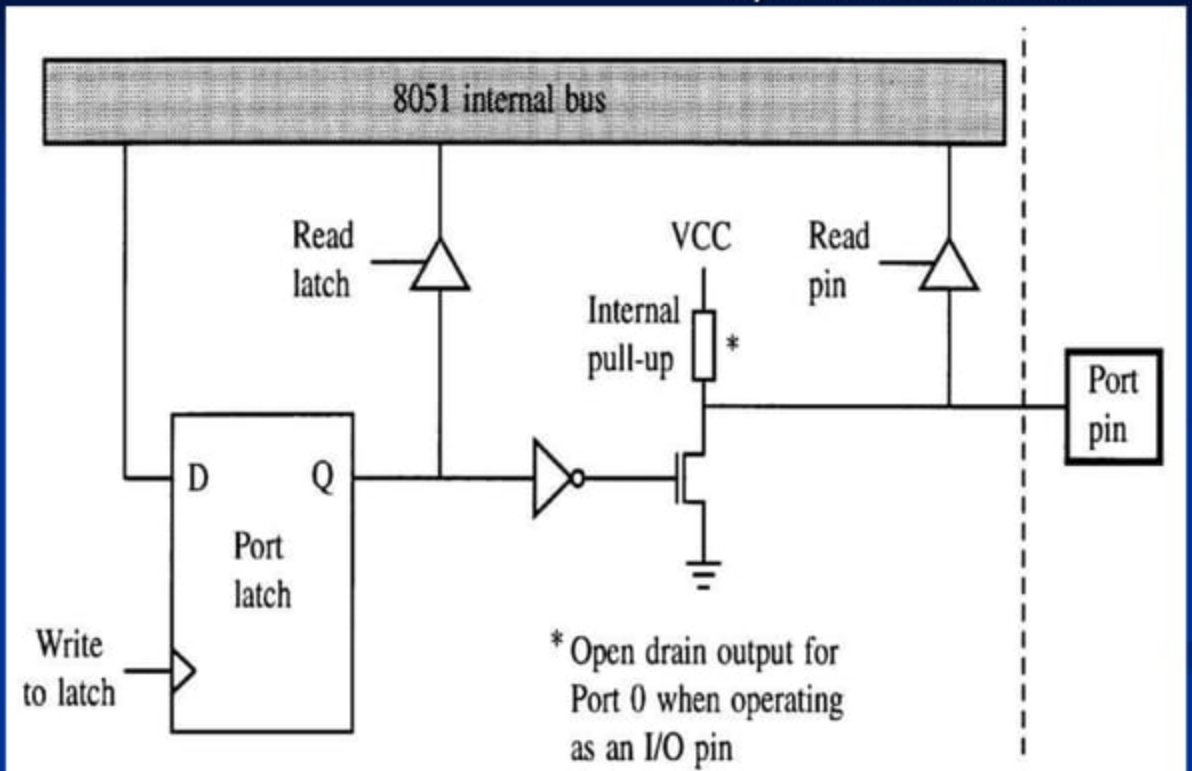
Alternate functions of P3

BIT	NAME	BIT ADDRESS	ALTERNATE FUNCTION
P3.0	RXD	B0H	Receive data for serial port
P3.1	TXD	B1H	Transmit data for serial port
P3.2	$\overline{\text{INT0}}$	B2H	External interrupt 0
P3.3	$\overline{\text{INT1}}$	B3H	External interrupt 1
P3.4	T0	B4H	Timer/counter 0 external input
P3.5	T1	B5H	Timer/counter 1 external input
P3.6	$\overline{\text{WR}}$	B6H	External data memory write strobe
P3.7	$\overline{\text{RD}}$	B7H	External data memory read strobe

I/O Port structure

- The internal circuitry for the I/O port is shown in the next slide
- If you want to read in from a pin, you must first give a logic '1' to the port latch to turn off the FET otherwise the data read in will always be logic '0'.
- When you write to the port you are actually writing to the latch e.g. a logic 0 given to the latch will be inverted and turn on the FET which cause the port pin to be connected to Gnd (logic 0).

I/O Port contd...



Timers /Counters

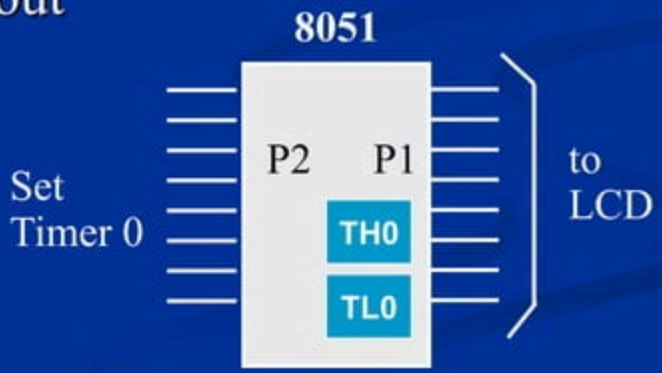
- The 8051 has 2 timers/counters:
  - Timer/Counter 0
  - Timer/Counter 1

They can be used as

1. The **Timer** :Uused as a time delay generator.
   - The clock source is the **internal** crystal frequency of the 8051.
2. An event **counter.**
   - **External input** from input pin to count the number of events on registers.
   - These clock pulses cold represent the number of people passing through an entrance, or the number of wheel rotations, or any other event that can be converted to pulses.

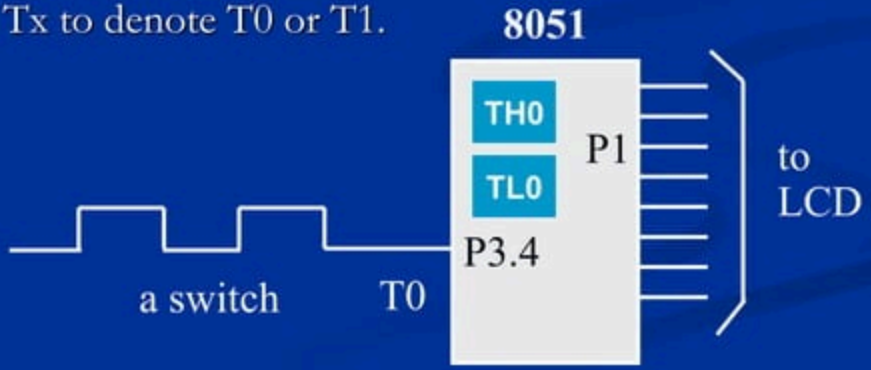
Timer

- Set the initial value of registers
- Start the timer and then the 8051 counts up.
- Input from internal system clock (machine cycle)
- When the registers equal to 0 and the 8051 sets a bit to denote time out



Counter

- Count the number of events
  - Show the number of events on registers
  - External input from T0 input pin (P3.4) for Counter 0
  - External input from T1 input pin (P3.5) for Counter 1
  - **External input** from Tx input pin.
  - We use Tx to denote T0 or T1.



Registers Used in Timer/Counter

- 8051 has two 16-bit Timer registers ,Timer 0 & Timer 1.
- As 8051 has 8-bit architecture , each Timer register is treated as two 8-bit registers namely

  TH0, TL0, TH1, TL1.
- One 8-bit mode register -TMOD.
- One 8-bit control register-TCON.

TMOD Register

- Both Timer 0 &Timer 1 use the same Mode register TMOD.
- It is an-8-bit register .The lower 4-bits are meant for Timer 0 &the uooer 4-bits are meant for Timer 1
- It is not bit addressible.
- It is used similar to any other register of 8051 . For ex:

```
MOV TMOD,#21H
```

Gate

- Every timer has a mean of starting and stopping.
  - GATE=0
    - **Internal** control
    - The start and stop of the timer are controlled by way of **software.**
    - Set/clear the TR for start/stop timer.

```
SETB TR0
CLR  TR0
```

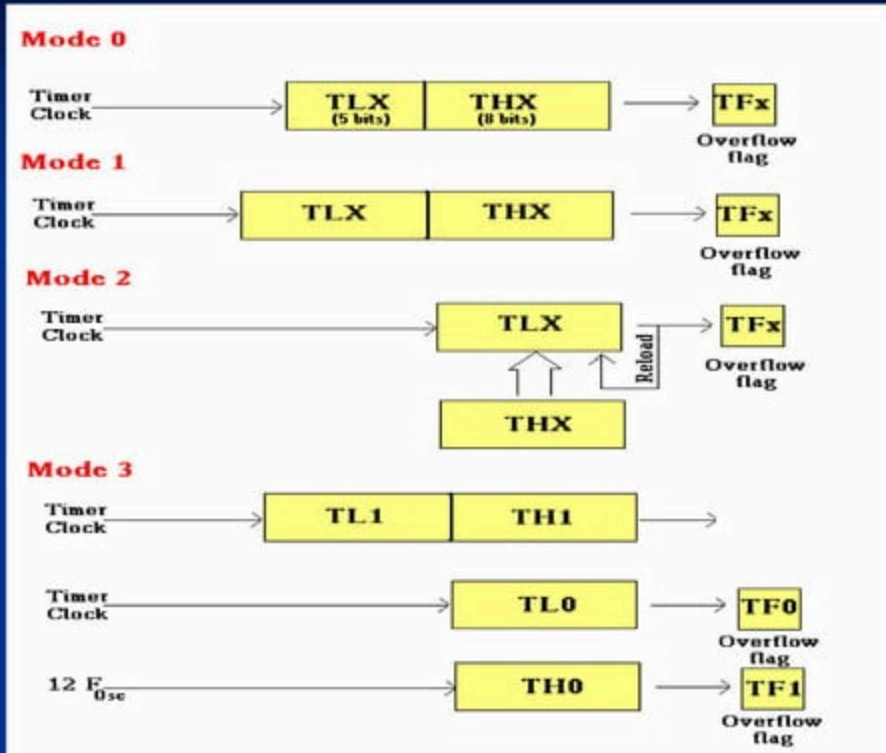
  - GATE=1
    - **External** control
    - The hardware way of starting and stopping the timer by **software** and **an external source.**
    - Timer/counter is enabled only while the INT pin is high and the TR control pin is set (TR).

C/T : Timer or counter selected cleared for timer operation (input from internal system clock). Set for counter operation (input from Tx input pin).

M1,M0 : Used for mode selection.Because the Timers of 8051 can be set in 4-different modes.

M1	M0	Mode	Operation
0	0	**0**	**13-bit timer** mode 8-bit THx + 5-bit TLx (x= 0 or 1)
0	1	1	**16-bit timer** mode 8-bit THx + 8-bit TLx
1	0	**2**	**8-bit auto reload** 8-bit auto reload timer/counter; THx holds a value which is to be reloaded into TLx each time it overflows.
1	1	**3**	Split timer mode

TMod contd…


Mode 0
Timer Clock
TLX (5 bits)
THX (8 bits)
TFx
Overflow flag
Mode 1
Timer Clock
TLX
THX
TFx
Overflow flag
Mode 2
Timer Clock
TLX
TFx
Overflow flag
Reload
THX
Mode 3
Timer Clock
TL1
TH1
Timer Clock
TL0
TF0
Overflow flag
12 F Osc
TH0
TF1
Overflow flag

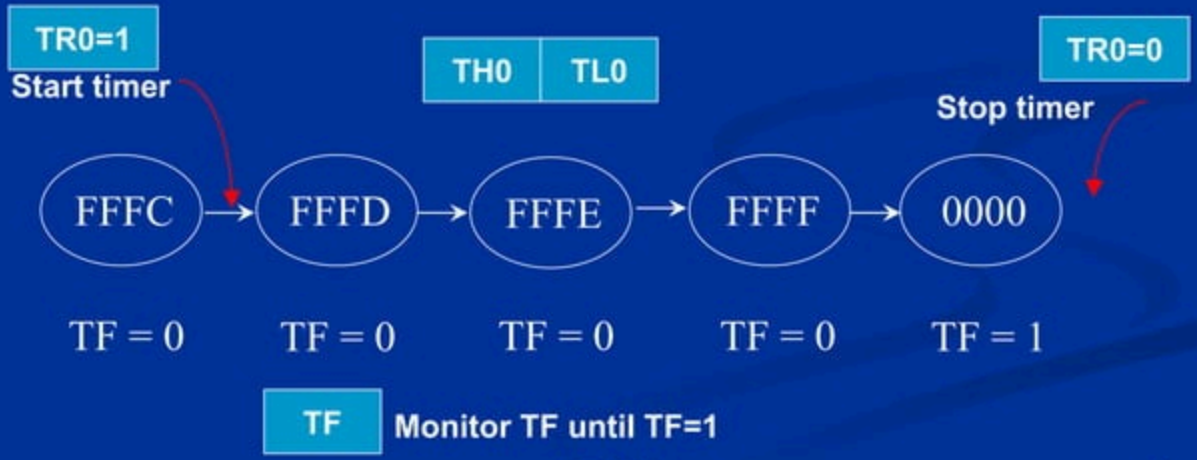
Let us understand the working of Timer Mode 1

- For this , let us consider timer 0 as an example.
- **16-bit** timer (TH0 and TL0)
- TH0-TL0 is incremented continuously when TR0 is set to 1. And the 8051 stops to increment TH0-TL0 when TR0 is cleared.
- The timer works with the internal system clock. In other words, the timer counts up each machine cycle.
- When the timer (TH0-TL0) reaches its maximum of FFFFH, it rolls over to 0000, and TF0 is raised.
- Programmer should check TF0 and stop the timer 0.

Steps of Mode 1

1. Choose mode 1 timer 0
   - `MOV TMOD,#01H`
2. Set the original value to TH0 and TL0.
   - `MOV TH0,#FFH`
   - `MOV TL0,#FCH`
3. You better to clear the TF: TF0=0.
   - `CLR TF0`
4. Start the timer.
   - `SETB TR0`

5. The 8051 starts to count up by incrementing the TH0-TL0.
   - **TH0-TL0= FFFCH,FFFDH,FFFEH,FFFFH,0000H**



Mode 1 contd…

6. When TH0-TL0 rolls over from FFFFH to 0000, the 8051 set TF0=1.

```
TH0-TL0= FFFE H, FFFF H, 0000 H (Now
   TF0=1)
```

7. Keep monitoring the timer flag (TF) to see if it is raised.

```
AGAIN:  JNB TF0, AGAIN
```

8. Clear TR0 to stop the process.

```
CLR TR0
```

9. Clear the TF flag for the next round.

```
CLR TF0
```

TCON Register

TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

- Timer control register TMOD is a 8-bit register which is bit addressable and in which Upper nibble is for timer/counter, lower nibble is for interrupts

Tcon contd…

- **TR** (Timer run control bit)
  - TR0 for Timer/counter 0; TR1 for Timer/counter 1.
  - TR is set by programmer to turn timer/counter on/off.
    - TR=0 : off (stop)
    - TR=1 : on (start)

- **TF** (timer flag, control flag)
  - **TF0 for timer/counter 0; TF1 for timer/counter 1.**
  - **TF is like a carry. Originally, TF=0. When TH-TL roll over to 0000 from FFFFH, the TF is set to 1.**
    - TF=0 : not reach
    - TF=1: reach
    - If we enable interrupt, TF=1 will trigger ISR.

Equivalent Instructions for the Timer Control Register

For timer 0

SETB TR0	=	SETB TCON.4
CLR TR0	=	CLR TCON.4
SETB TF0	=	SETB TCON.5
CLR TF0	=	CLR TCON.5

For timer 1

SETB TR1	=	SETB TCON.6
CLR TR1	=	CLR TCON.6
SETB TF1	=	SETB TCON.7
CLR TF1	=	CLR TCON.7

TCON: Timer/Counter Control Register

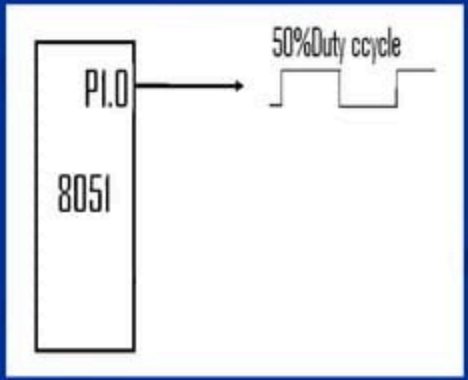
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

Simple applications using ports &Timers

- Using a port ,by a simple program you can generate a Square wave of any duty cycle.

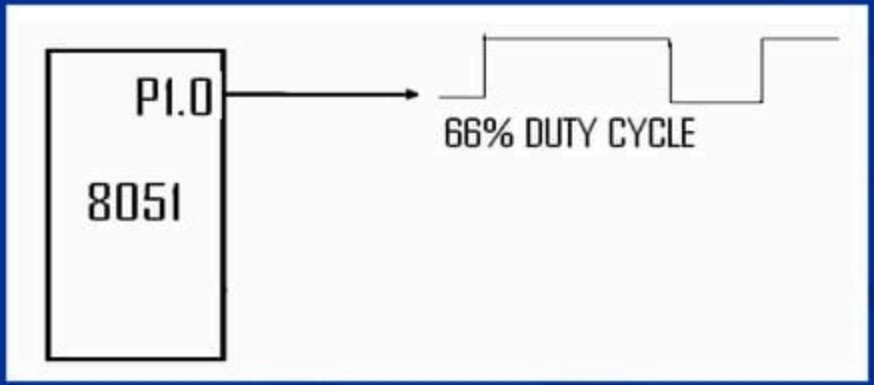
```
HERE : SETB   P1.0     (Make bit of Port 0 High)
       LCALL DELAY
       CLR P1.0
       LCALL DELAY
       SJMP  HERE   : Keep doing it
```

Here same delay is used for both
High & low



Square-wave of 66%duty cycle.

```
HERE : SETB   P1.0    ( Make bit of Port 0 High)
       LCALL DELAY
       LCALL DELAY
       CLR   P1.0
       LCALL DELAY
       SJMP HERE   : Keep doing it
```

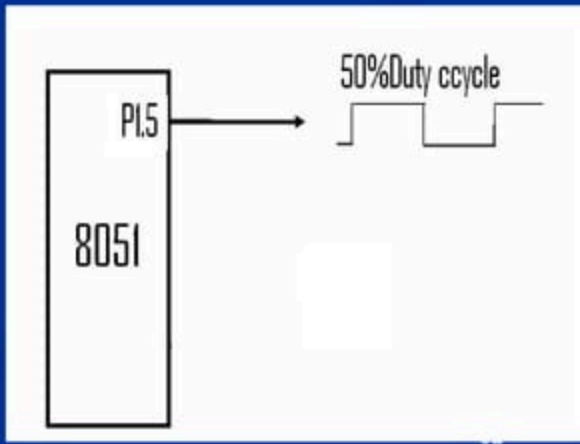


Square-wave generation using Timer

- Square wave of 50% duty on P1.5
- Timer 0 is used

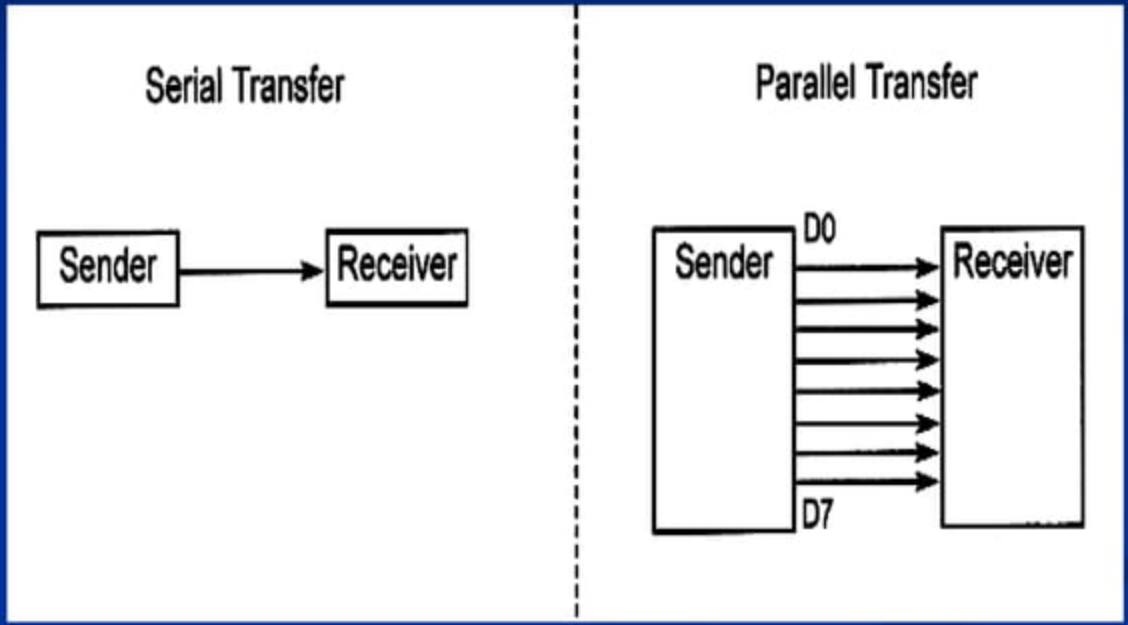
```
; each loop is a half clock
        MOV TMOD,#01   ;Timer 0,mode 1(16-bit)
HERE: MOV TL0,#0F2H  ;Timer value = FFF2H
        MOV TH0,#0FFH
        CPL P1.5
        ACALL DELAY
        SJMP HERE

DELAY:
        SETB TR0      ;start the timer 0
AGAIN: JNB TF0,AGAIN
        CLR TR0       ;stop timer 0
        CLR TF0       ;clear timer 0 flag
        RET
```

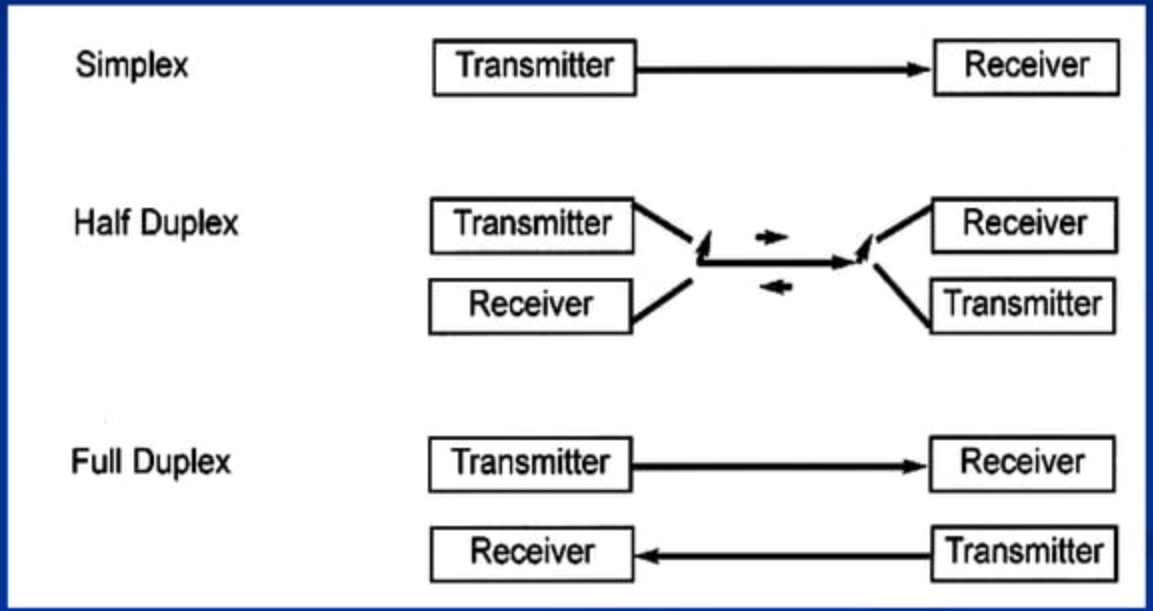


8051- SERIAL COMMUNICATION

Basics of serial communication



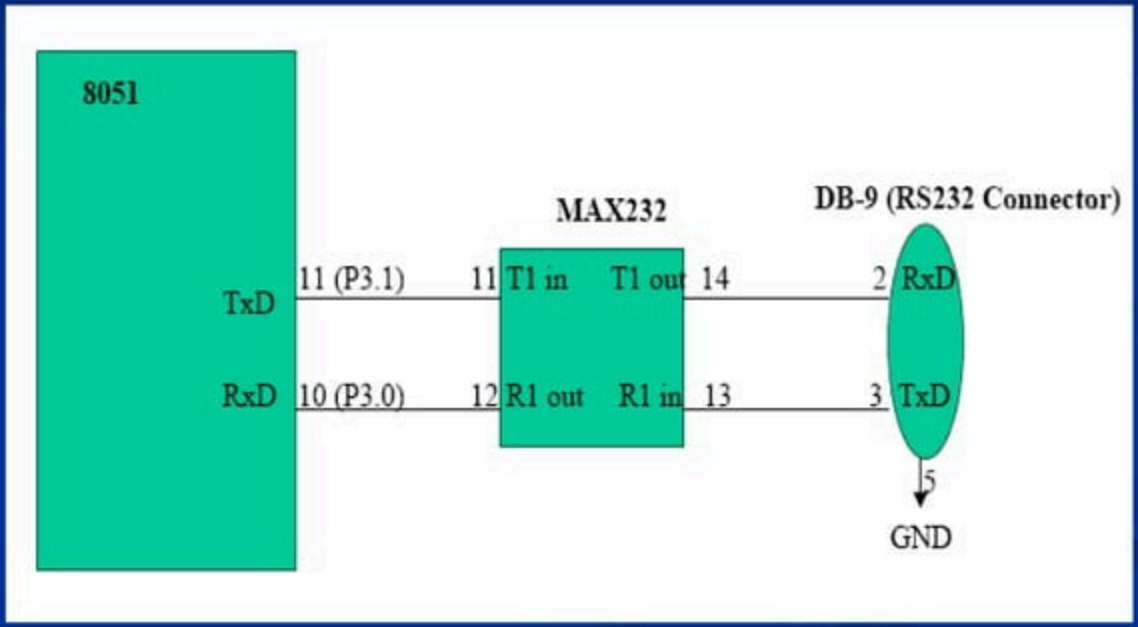
Types of Serial communications



RxD and TxD pins in the 8051

- The 8051 has two pins for transferring and receiving data by serial communication. These two pins are part of the Port3(P3.0 &P3.1)
- These pins are TTL compatible and hence they require a line driver to make them RS232 compatible
- Max232 chip is one such line driver in use.
- Serial communication is controlled by an 8-bit register called SCON register,it is a bit addressable register.

Interfacing to PC



SCON (Serial control) register

SM0	SM1	SM2	REN	TB8	RB8	TI	RI

SM0	SCON.7	Serial port mode specifier
SM1	SCON.6	Serial port mode specifier
SM2	SCON.5	Used for multiprocessor communication. (Make it 0)
REN	SCON.4	Set/cleared by software to enable/disable reception.
TB8	SCON.3	Not widely used.
RB8	SCON.2	Not widely used.
TI	SCON.1	Transmit interrupt flag. Set by hardware at the beginning of the stop bit in mode 1. Must be cleared by software.
RI	SCON.0	Receive interrupt flag. Set by hardware halfway through the stop bit time in mode 1. Must be cleared by software.

SM0 , SM1

- These two bits of SCON register determine the framing of data by specifying the number of bits per character and start bit and stop bits. There are 4 serial modes.

SM0	SM1	
0	0	Serial Mode 0
0	1	Serial Mode 1, 8 bit data, 1 stop bit, 1 start bit
1	0	Serial Mode 2
1	1	Serial Mode 3

REN, TI, RI

- REN (Receive Enable) also referred as SCON.4. When it is high,it allows the 8051 to receive data on the RxD pin. So to receive and transfer data REN must be set to 1.When REN=0,the receiver is disabled. This is achieved as below

```
SETB  SCON.4
&     CLR   SCON.4
```

contd…

- TI (Transmit interrupt) is the D1 bit of SCON register. When 8051 finishes the transfer of 8-bit character, it raises the TI flag to indicate that it is ready to transfer another byte. The TI bit is raised at the beginning of the stop bit.

- RI (Receive interrupt) is the D0 bit of the SCON register. When the 8051 receives data serially ,via RxD, it gets rid of the start and stop bits and places the byte in the SBUF register.Then it raises the RI flag bit to indicate that a byte has been received and should be picked up before it is lost.RI is raised halfway through the stop bit.

8051 Interrupts

- An *interrupt* is an external or internal event that disturbs the microcontroller to inform it that a device needs its service.

  A Microcontroller can serve various devices.
- There are two ways to do that:
  - interrupts &
  - polling.
- The program which is associated with the interrupt is called the ***interrupt service routine*** (ISR) or ***interrupt handler***.

Steps in executing an interrupt

- Upon receiving the interrupt signal the Microcontroller , finish current instruction and saves the PC on stack.
- Jumps to a fixed location in memory depending on type of interrupt
- Starts to execute the interrupt service routine until RETI (return from interrupt)
- Upon executing the RETI the microcontroller returns to the place where it was interrupted. Get pop PC from stack

Interrupt Sources

- Original 8051 has 6 sources of interrupts
  - Reset
  - Timer 0 overflow
  - Timer 1 overflow
  - External Interrupt 0
  - External Interrupt 1
  - Serial Port events (buffer full, buffer empty, etc)
- Enhanced version has 22 sources
  - More timers, programmable counter array, ADC, more external interrupts, another serial port (UART)

Interrupt Vectors

- Each interrupt has a specific place in code memory where program execution (interrupt service routine) begins.
- **External Interrupt 0:** **0003h**
- **Timer 0 overflow:** **000Bh**
- **External Interrupt 1:** **0013h**
- **Timer 1 overflow:** **001Bh**
- **Serial :** **0023h**
- **Timer 2 overflow**(8052+) **002bh**

Interrupt Enable Register

- Upon reset all Interrupts are disabled & do not respond to the Microcontroller
- These interrupts must be enabled by software in order for the Microcontroller to respond to them.
- This is done by an 8-bit register called Interrupt Enable Register (IE).

EA	—	ET2	ES	ET1	EX1	ET0	EX0

- EA : **Global enable/disable.**
- **---** : **Undefined.**
- ET2 : Enable Timer 2 interrupt.
- ES : Enable Serial port interrupt.
- ET1 : Enable Timer 1 interrupt.
- EX1 :Enable External 1 interrupt.
- ET0 : Enable Timer 0 interrupt.
- EX0 : Enable External 0 interrupt.

Enabling and disabling an interrupt

- By bit operation
- Recommended in the middle of program

```
SETB   EA      SETB   IE.7    ;Enable All
SETB   ET0     SETB   IE.1    ;Enable Timer0 ovrf
SETB   ET1     SETB   IE.3    ;Enable Timer1 ovrf
SETB   EX0     SETB   IE.0    ;Enable INT0
SETB   EX1     SETB   IE.2    ;Enable INT1
SETB   ES      SETB   IE.4    ;Enable Serial port
```

- By Mov instruction
- Recommended in the first of program

```
MOV IE, #10010110B
```

Interrupt Priorities

- What if two interrupt sources interrupt at the same time?
- The interrupt with the highest PRIORITY gets serviced first.
- All interrupts have a power on default priority order.
  1. External interrupt 0 (INT0)
  2. Timer interrupt0 (TF0)
  3. External interrupt 1 (INT1)
  4. Timer interrupt1 (TF1)
  5. Serial communication (RI+TI)
- Priority can also be set to "high" or "low" by IP reg.

Interrupt Priorities (IP) Register

---	---	PT2	PS	PT1	PX1	PT0	PX0

IP.7: reserved

IP.6: reserved

IP.5: Timer 2 interrupt priority bit (8052 only)

IP.4: Serial port interrupt priority bit

IP.3: Timer 1 interrupt priority bit

IP.2: External interrupt 1 priority bit

IP.1: Timer 0 interrupt priority bit

IP.0: External interrupt 0 priority bit

Interrupt Priorities Example

---	---	PT2	PS	PT1	PX1	PT0	PX0

- `MOV IP , #00000100B` or `SETB IP.2` gives priority order
  1. Int1
  2. Int0
  3. Timer0
  4. Timer1
  5. Serial
- `MOV IP , #00001100B` gives priority order
  1. Int1
  2. Timer1
  3. Int0
  4. Timer0
  5. Serial

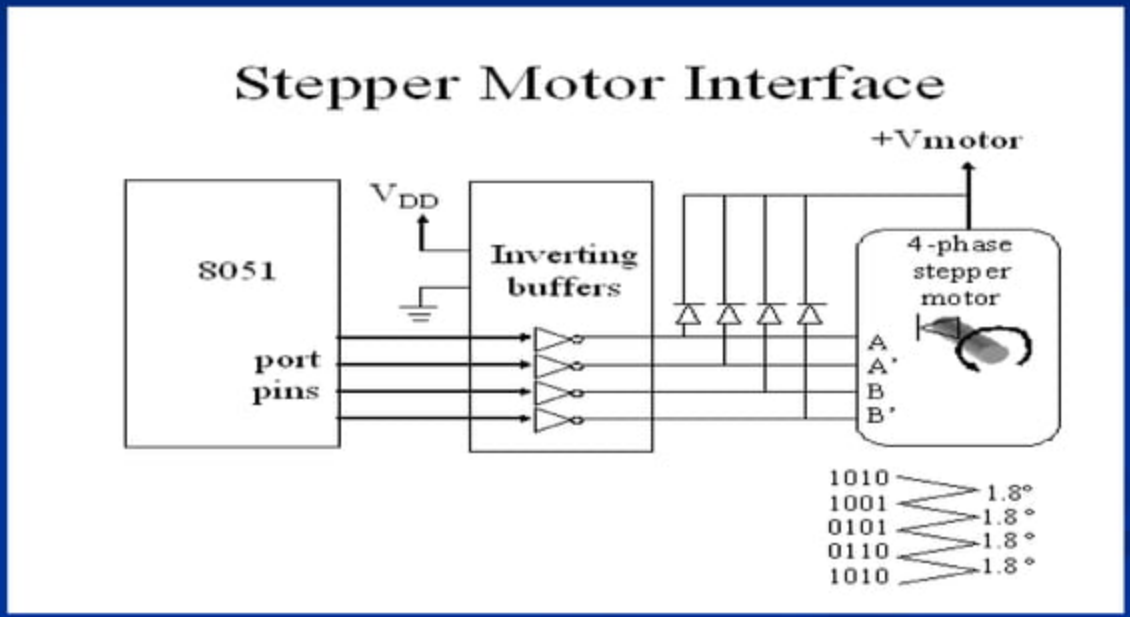
Interrupt inside an interrupt

---	---	PT2	PS	PT1	PX1	PT0	PX0

- A high-priority interrupt can interrupt a low-priority interrupy
- All interrupt are latched internally
- Low-priority interrupt wait until 8051 has finished servicing the high-priority interrupt

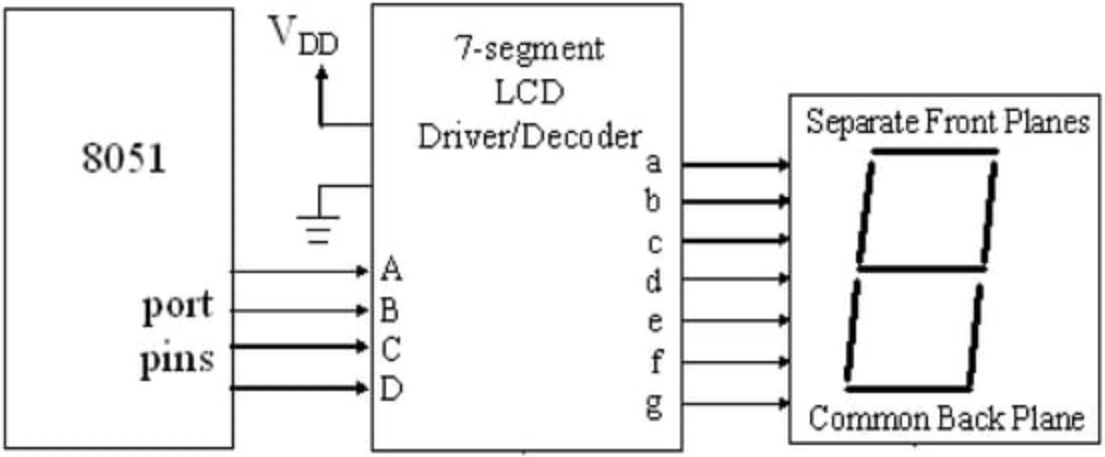
Applications of Microcontrollers

Simple Interfacing Examples

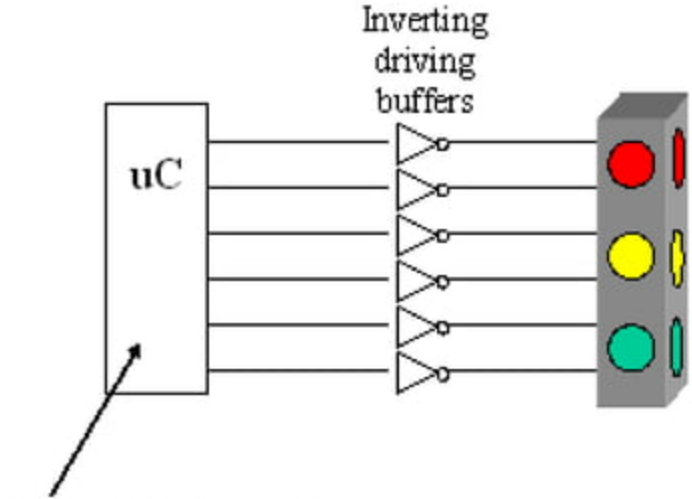


Seven segment Interfacing

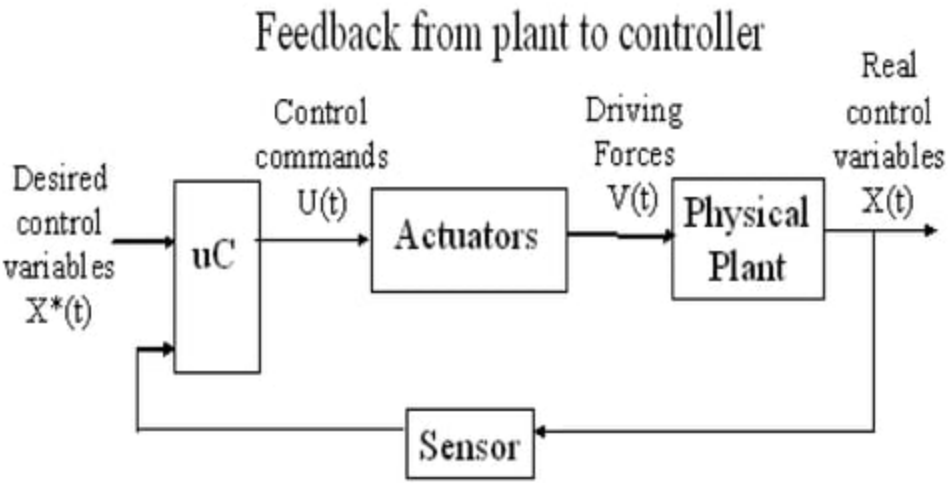
Simple parallel interface – similar to LED:



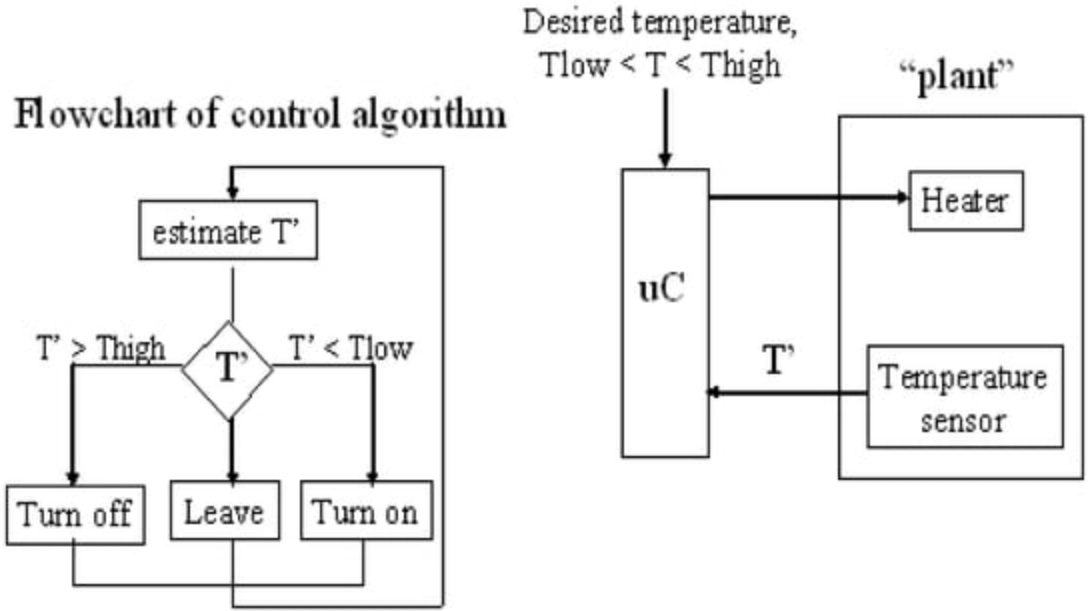
Traffic light controller



Closed-loop Control



Closed loop control system- Temperature control example



Every Day Examples of Devices that use Microcontrollers

- A typical home in the Western world is likely to have only one or two general-purpose microprocessors but somewhere between one and two dozen microcontrollers.
- They can be found in almost any electrical device, washing machines, microwave ovens, telephones etc.

Microcontroller Application Examples

Point of Sale

- P.O.S.
- Vending Machine

Telecom

- PABX
- Bridges

Appliances

- High-end appliance
- User interface

Measurement

- Testing equipment
- Medical equipment

USB

- USB key
- Smart card reader
- High end peripherals

Automation

- PLC
- User interface

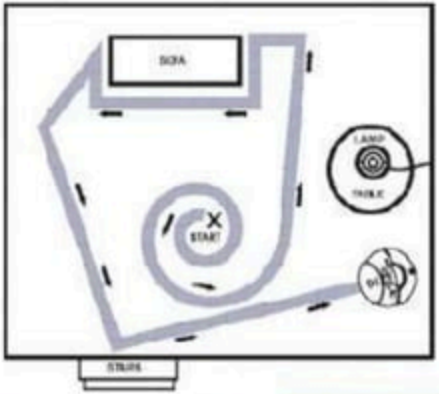
Examples of Research Robots that use Microcontrollers

Lawn Mower Robots

Robotic Lawn Mower

Prototype by the Robot Shop

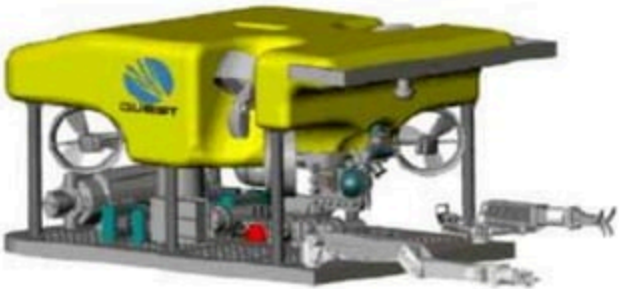
Commercial Vacuum Cleaning Robots



High Tech and Aerospace Microcontroller Examples

High-tech and Aerospace Microcontroller Examples

Ecological data collection by EME Systems (left), undersea research by Harbor Branch Institute (center), and JP Aerospace test launch (right)



Educational Robots

Educational Robots

Parallax Boe-Bot™ (left)
First Competition Robot (right)

Novel Design Applications

Microcontroller Emphasis on Small Portable Computing applications

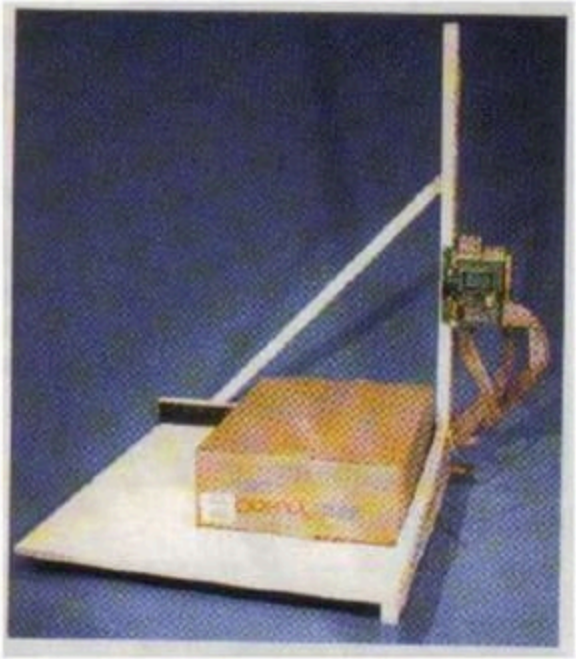
- Multifunctional Bicycle Computer
- Uses MAXQ2000 Microcomputer
- Speedometer
- Clock
- Thermometer
- Humidity sensor
- Heart rate
- Stop watch

Recent Wonders

- Hand Speak
- Enhance communication abilities of deaf people
- M16/62 Microcontroller converts American Sign Language (ASL) movements into alphanumeric characters that are displayed on LCD

Recent Wonders contd .....

- "Weasure"
- Clever system designed to not only weigh, but also measure packages for shipment
- With touch of a button the weight and dimensions are shown on a display and uploaded to a PC through serial port



Recent Wonders contd .....

- "Pet Inspect"
- M16C/62P Microcontroller-based system to monitor every faucet of your pet's life
- Data logger and Wireless communications system monitors activity level,pressure(altitude), ambient and core body temperature,natural and artificial light intensity, and proximity

Books that have helped me to understand the Microcontrollers & embedded systems

1) Barr, Michael, Programming embedded sytems in C and C++ - OReilly Publ.
2) Raj Kamal, Embedded systems, TMG
3) Mazidi and Mazidi, The 8051 microcontroller and embedded sytems - Pearson education.
4) Peatman,J.B. Design with microcontrollers and microcomputers, McGraw Hil

5) Sewart. J.W. The 8051 Microcontroller Hardware, Software and Interfacing – Prentice Hall

6) Ayala Kenneth, The 8051 Microntroller – Architecture, Programming and Applications – Delmar Publ.

7) Ajay Deshmukh, Microcontrollers – TATA McGraw Hill

8) Rajkamal, Microcontrollers - Architecture, Programming – Pearson Publ.

9) Myke Predko, Programming the 8051 Microcontroller – McGraw Hill

10) Michael J. Pont, Embedded C - Addison Wesely Publ.

Useful websites

8052 tutorial information by Vault Information Services:
http://www.8052.com

Intel's site for 8051 based products:
http://developer.intel.com/design/mcs51/

Philips' site for 8051 based products:
http://www-us.semiconductors.philips.com/microcontrol/

Infineon (formerly Siemens) site for 8051 based products:
http://www.infineon.com/products/micro/micro.htm

Keil development tools:
http://www.keil.com/home.htm

Information on Analog Devices ADuC812 (8051/8052 compatible processor):
www.analog.com/microconverter

Useful websites contd…

1. http://www.eg3.com
2. http://www.ARM.MCU.com
3. http://www.mcjournal.com
4. http://www.iar.com
5. http://http://www.embedded.com
6. http://www.powersoftsystems.com

Epilogue

- The woods are lovely, dark and deep,
  But I have promises to keep,
  And miles to go before I sleep,
  And miles to go before I sleep.
  ---- Robert Frost

- *GOOD LUCK!*