Microcontroller, Relays & Programmable Logic Controllers (PLCs)

What is a Microcontroller?

- Mini-Computer
  - *Microprocessor*
    - *The Brains*
    - *Arithmetic Logic Unit (ALU)*
    - *Control Unit*

- *Program/ Data Storage*
- *Peripherals (Input/Output)*
- *Low-Cost*

•A microcontroller is a kind of miniature computer that found in all kinds of gizmos

•Generally speaking, if a device has buttons and a digital display, chances are it also has a programmable microcontroller brain.

Microcontroller

Microcontrollers are 'single chip' computers specifically designed to:

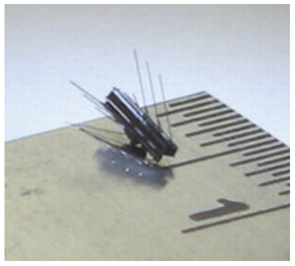
- Read input devices, such as buttons and sensors
- Process data or information
- Control output devices, such as lights, displays, motors and speakers

•First used in 1975(Intel 8048)

- **Microcontrollers are placed in devices, or embedded, for operation and control (embedded control).**

• The introduction of EEPROM in 1993, allowed microcontrollers to be electrically erased

• The same year, Atmel introduced the first microcontroller using Flash memory.

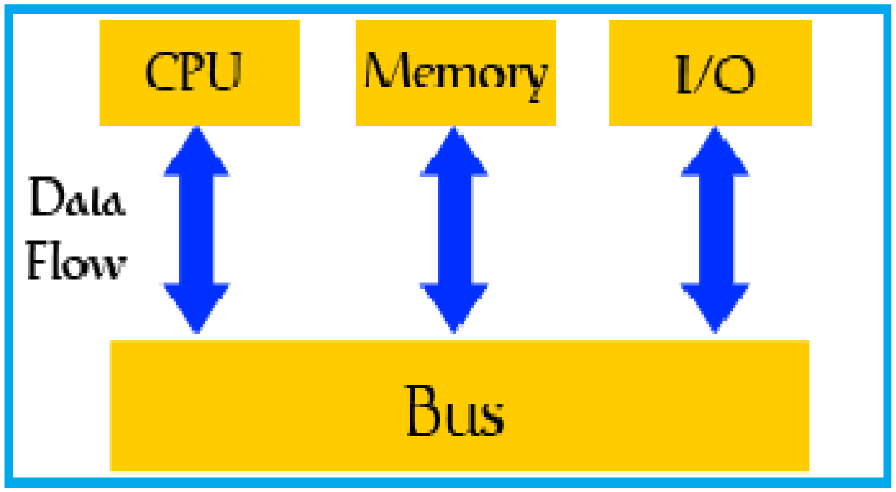


Microcontrollers Architecture

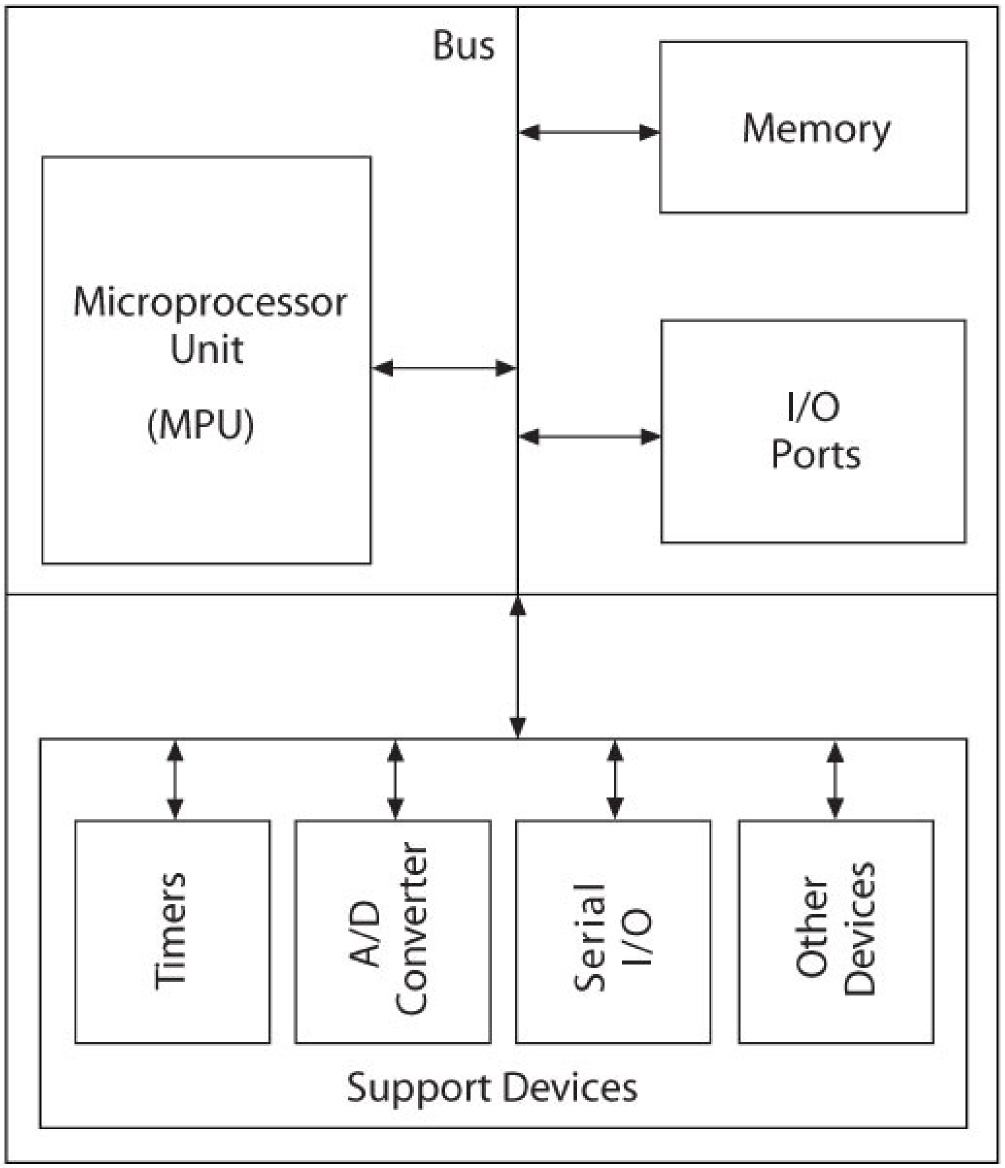
Microcontroller incorporates features of Microprocessor (CPU, ALU, Registers)

Additional features:

- RAM,ROM,I\O ports, counter etc.
- Control the operation of machine
- using fixed program stored in Rom

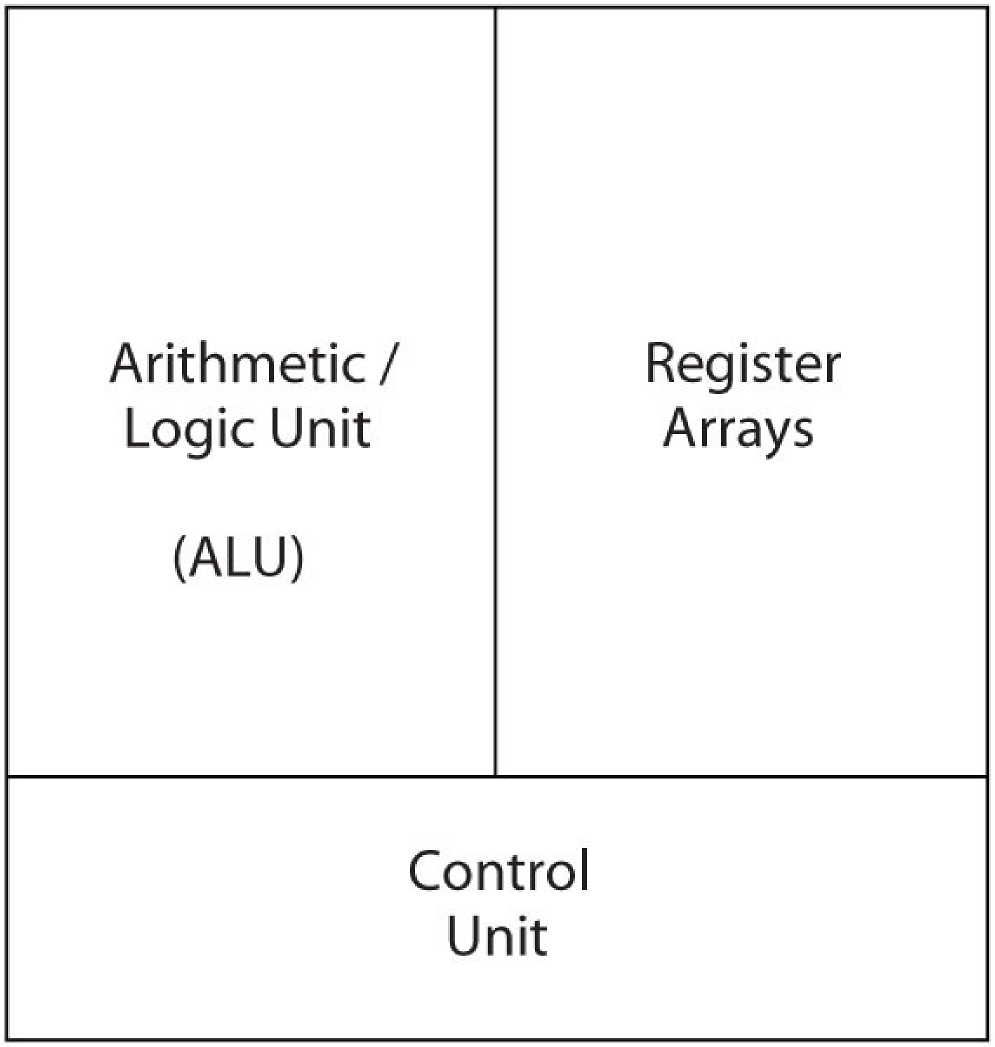


- **The CPU core**
- **Memory (both ROM and RAM)**
- **Some parallel digital I/O**



Basic microcontroller architecture

Microprocessor (MPU)



MPU (CPU)

- Read instructions
- Process binary data

Memory: Storage Device

- Addresses
- Registers

Major Categories

- Read/Write Memory (R/W)
- Read-only-Memory (ROM)

Input/Output (I/O)

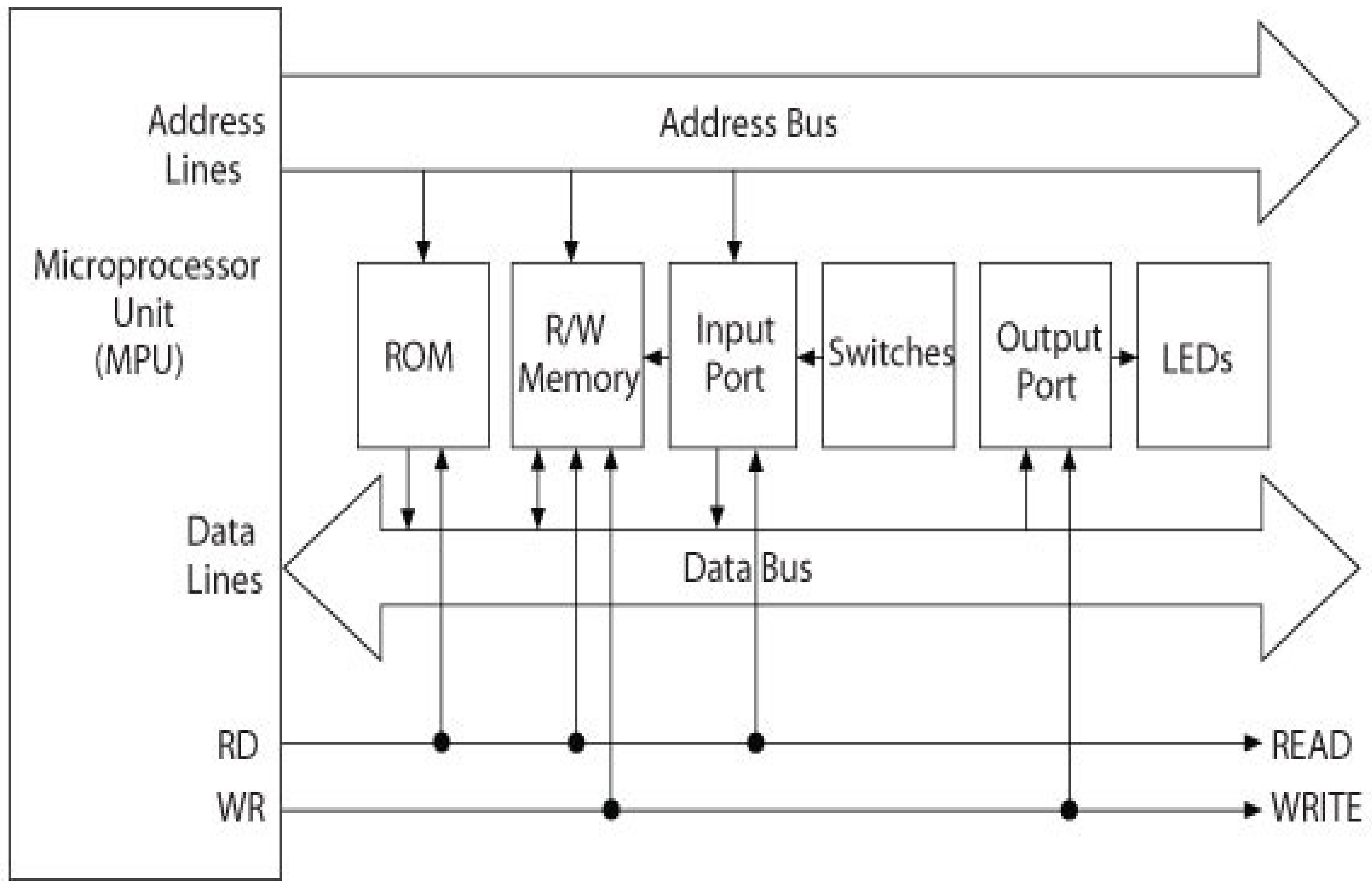
Input Devices

- Switches and Keypads
- Provide binary information to the MPU

Output devices

- LEDs and LCDs
- Receive binary information from the MPU

Microprocessor-Based Systems



Microprocessor vs. Microcontroller

A microprocessor is the "brain" of a computer system

- Generally referred to as the central processing unit (CPU), the microprocessor by itself is practically useless
- To be useful, one must have means of communicating with it using input and output devices
- One must also add memory (ROM and RAM) so that the system can be programmed.

A microcontroller is a computer chip designed for control-oriented applications

•Unlike ordinary microprocessors, microcontrollers have built-in features that make them operate almost independent of additional circuitry

•This is possible because microcontrollers contain things like

- **•memory (ROM, EPROM, RAM, etc)**
- **•input and output ports and timers**
- **•serial and parallel communication capability**
- **•analog-to-digital converters**

•A microcontroller is a specialized form of microprocessor

•Designed to be self-sufficient and cost-effective

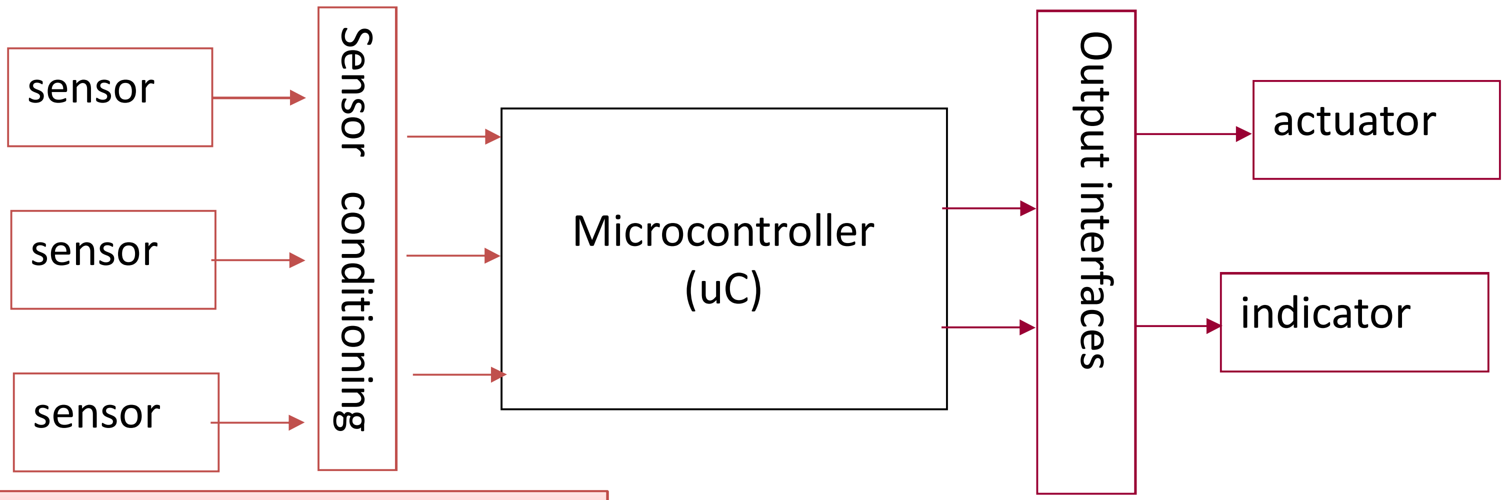
•Microprocessor is typically designed to be general purpose

Embedded System

•A computer system designed to perform one or a few dedicated functions often with real-time computing constraints.

• Controlled by one or more main processing cores that is typically either a microcontroller or a digital signal processor.

Embedded System (General Block Diagram)



Implementation of Embedded System

•Microcontroller : Independent System consisting of all components

•PLC (Programmable Logic Control) : Microcontroller or Microprocessor is the core Used in industrial projects

Block diagram of Microcontroller

•Memory : RAM, ROM, Store data and code

•CPU: Mathematical and logical operation, Memory units are called Register

•BUS: Group of 8,16 or more wires
–Three type, address bus, data bus and control bus

•Input-output unit : port A, port B, port C … …
–Input, output and bidirectional ports

•Serial communication

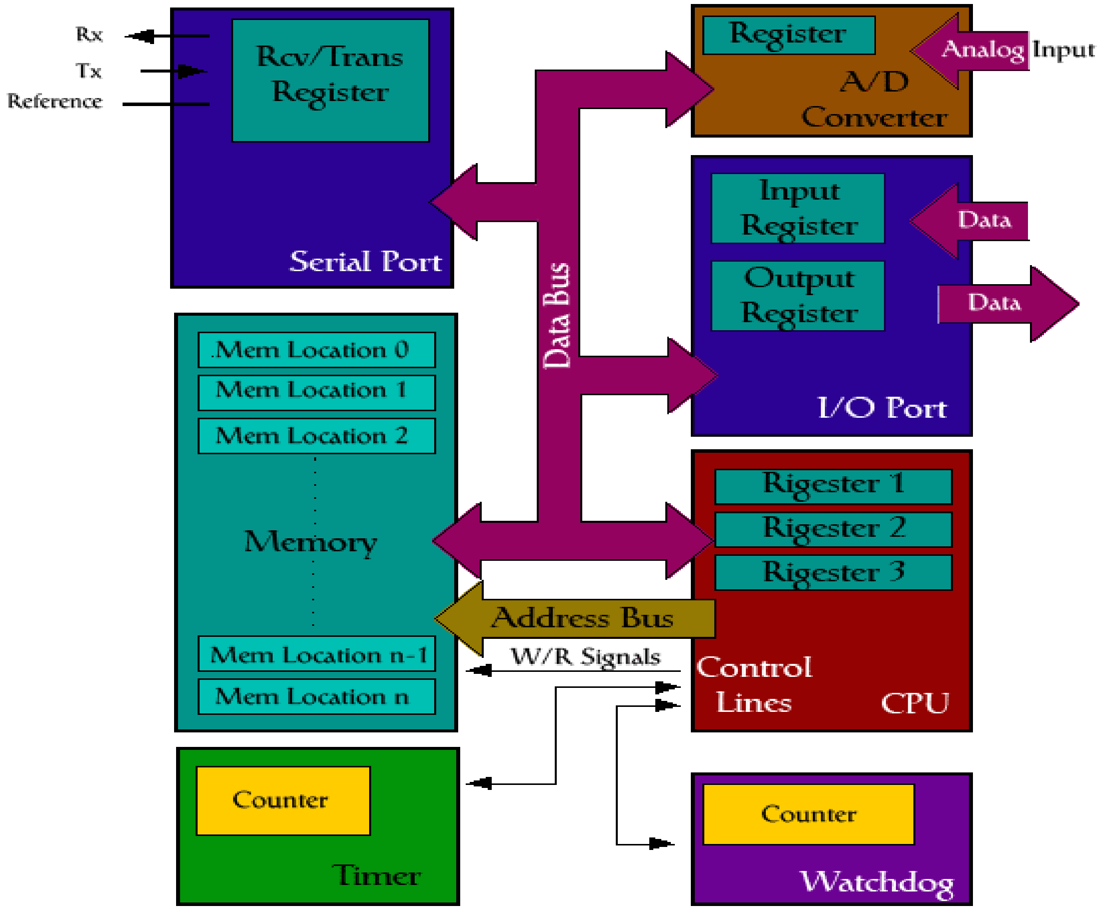
•Timer unit

•Watchdog: Automatic reset to prevent stall

•Analog to Digital Converter (ADC)

•Writing a program which is executed while the microcontroller runs.

•Can be written in assembly, C or Basic



Microcontroller Programming

• Assembly :Takes up least amount of spaces and Best result in terms of speed

•C :Easier to write and understand & Slower than Assembly

•Basic: Nearest a man's way of reasoning

•Microcontroller understands only hex code

What is 8051 Standard?

•Microcontroller manufacturers every couple of days a new chip with a higher operating frequency, more memory and upgraded A/D converters appeared on the market.

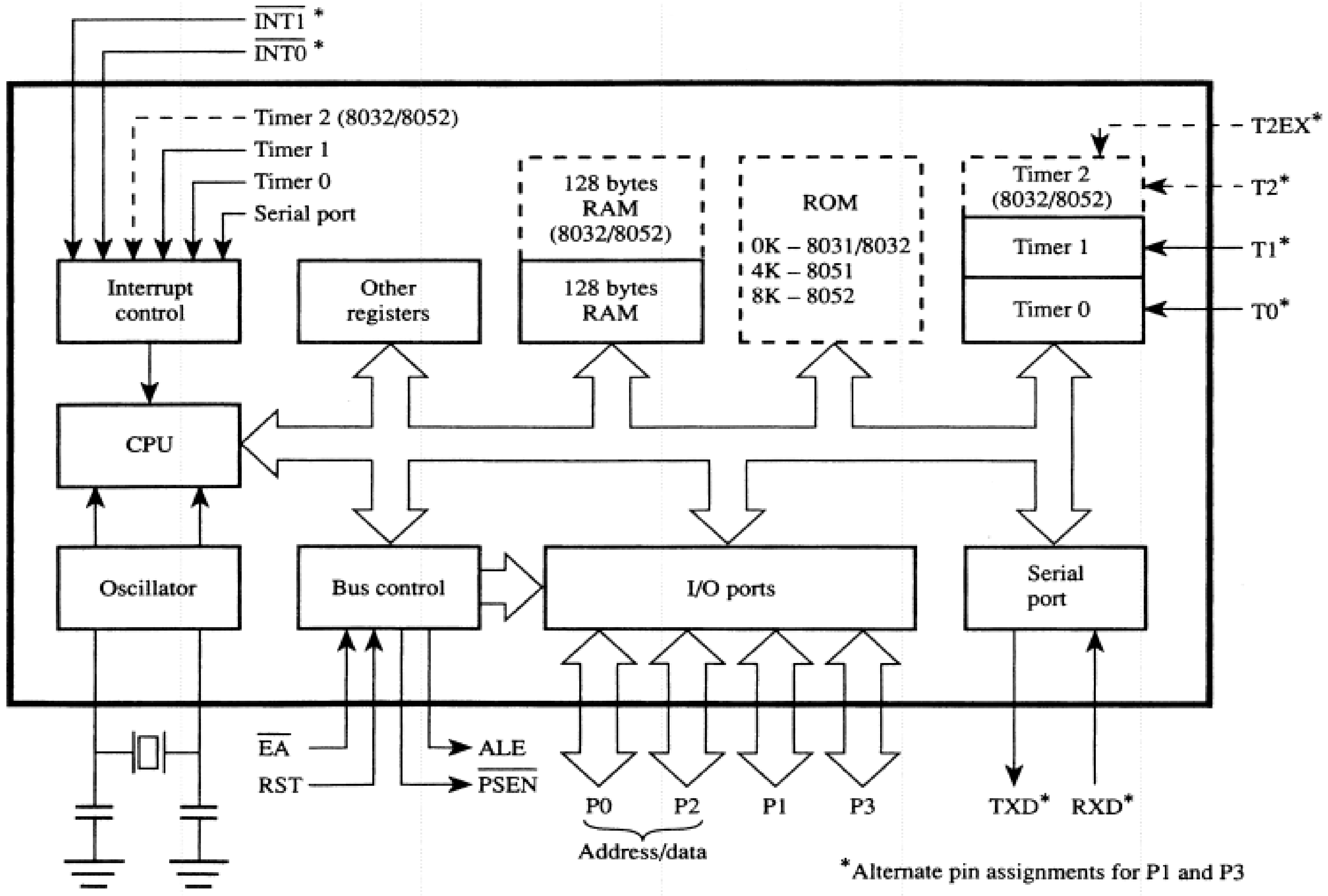
•However, most of them had the same or at least very similar architecture known in the world of microcontrollers as “8051 compatible”. What is all this about?

•The whole story has its beginnings in the far 80s when Intel launched the first series of microcontrollers called the MCS 051.

•Even though these microcontrollers had quite modest features in comparison to the new ones, they conquered the world very soon and became a standard for what nowadays is called the microcontroller.

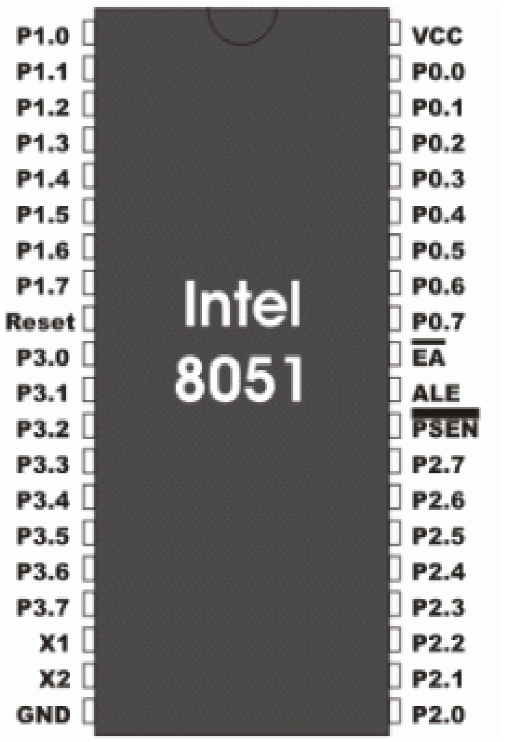
•This is the reason for having a great number of various microcontrollers which basically are solely upgraded versions of the 8051 family. What makes this microcontroller so special and universal so that almost all manufacturers all over the world manufacture it today under different name?

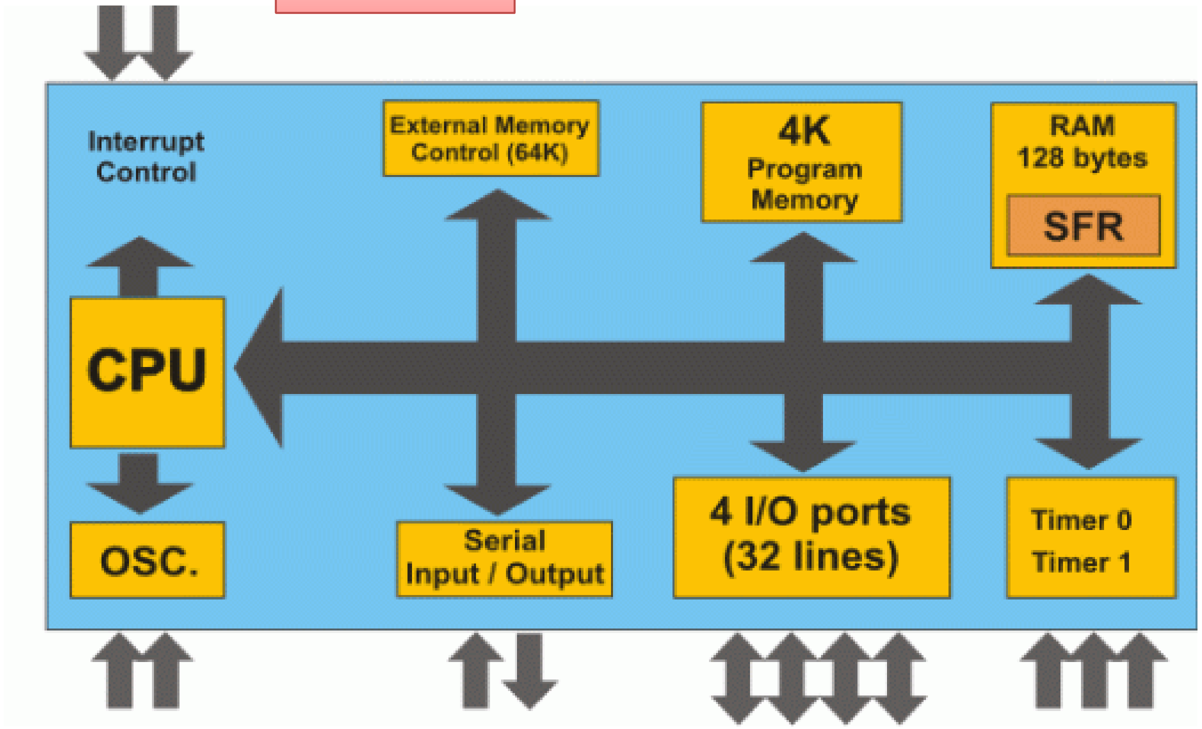
8051 Microcontroller



*Alternate pin assignments for P1 and P3

Cont…





Pins 1-8: Port 1 Each of these pins can be configured as an input or an output.
Pin 9: RS A logic one on this pin disables the microcontroller and clears the contents of most registers. In other words, the positive voltage on this pin resets the microcontroller. By applying logic zero to this pin, the program starts execution from the beginning.
Pins10-17: Port 3 Similar to port 1, each of these pins can serve as general input or output. Besides, all of them have alternative functions:
Pin 10: RXD Serial asynchronous communication input or Serial synchronous communication output.
Pin 11: TXD Serial asynchronous communication output or Serial synchronous communication clock output.
Pin 12: INT0 Interrupt 0 input.
Pin 13: INT1 Interrupt 1 input.
Pin 14: T0 Counter 0 clock input.

Cont……

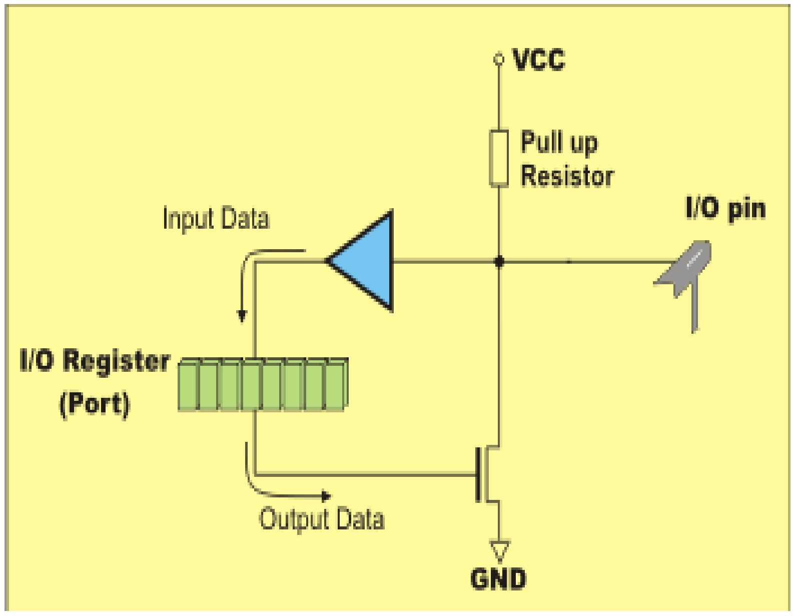
Pin 15: T1 Counter 1 clock input.
Pin 16: WR Write to external (additional) RAM.
Pin 17: RD Read from external RAM.
Pin 18, 19: X2, X1 Internal oscillator input and output. A quartz crystal which specifies operating frequency is usually connected to these pins. Instead of it, miniature ceramics resonators can also be used for frequency stability. Later versions of microcontrollers operate at a frequency of 0 Hz up to over 50 Hz.
Pin 20: GND Ground.
Pin 21-28: Port 2 If there is no intention to use external memory then these port pins are configured as general inputs/outputs. In case external memory is used, the higher address byte, i.e. addresses A8-A15 will appear on this port. Even though memory with capacity of 64Kb is not used, which means that not all eight port bits are used for its addressing, the rest of them are not available as inputs/outputs.
Pin 29: PSEN If external ROM is used for storing program then a logic zero (0) appears on it every time the microcontroller reads a byte from memory.
Pin 30: ALE Prior to reading from external memory, the microcontroller puts the lower address byte (A0-A7) on P0 and activates the ALE output. After receiving signal from the ALE pin, the external register (usually 74HCT373 or 74HCT375 add-on chip) memorizes the state of P0 and uses it as a memory chip address. Immediately after that, the ALU pin is returned its previous logic state and P0 is now used as a Data Bus. As seen, port data multiplexing is performed by means of only one additional (and cheap) integrated circuit. In other words, this port is used for both data and address transmission.
Pin 31: EA By applying logic zero to this pin, P2 and P3 are used for data and address transmission with no regard to whether there is internal memory or not. It means that even there is a program written to the microcontroller, it will not be executed. Instead, the program written to external ROM will be executed. By applying logic one to the EA pin, the microcontroller will use both memories, first internal then external (if exists).
Pin 32-39: Port 0 Similar to P2, if external memory is not used, these pins can be used as general inputs/outputs. Otherwise, P0 is configured as address output (A0-A7) when the ALE pin is driven high (1) or as data output (Data Bus) when the ALE pin is driven low (0).
Pin 40: VCC +5V power supply.

Input / Output Ports (I/O Ports)

•All 8051 microcontrollers have 4 I/O ports each comprising 8 bits which can be configured as inputs or outputs. Accordingly, in total of 32 input/output pins enabling the microcontroller to be connected to peripheral devices are available for use.

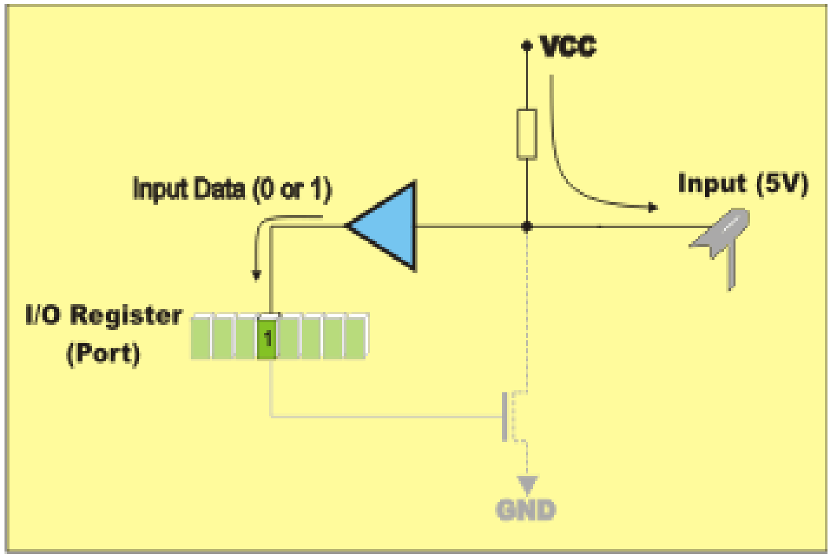
•Logic state (voltage) of any pin can be changed or read at any moment. A logic zero (0) and logic one (1) are not equal. A logic one (0) represents a short circuit to ground. Such a pin acts as an output.

•A logic one (1) is "loosely" connected to the power supply voltage over a resistor of high resistance. Since this voltage can be easily "reduced" by an external signal, such a pin acts as an input.



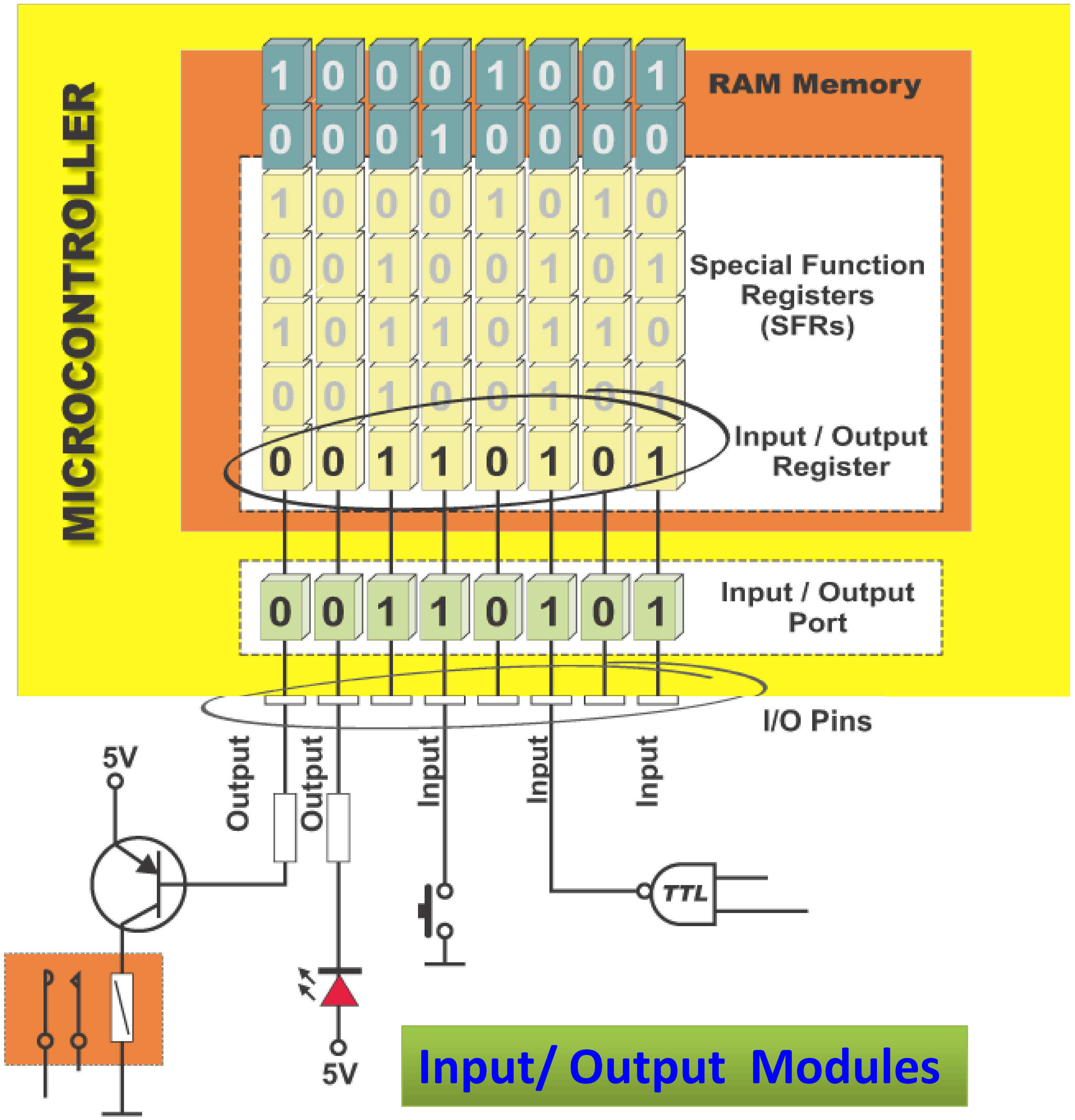
Output pin

A logic zero (0) is applied to a bit of the P register. The output FE transistor is turned on, thus connecting the appropriate pin to ground.



Input pin

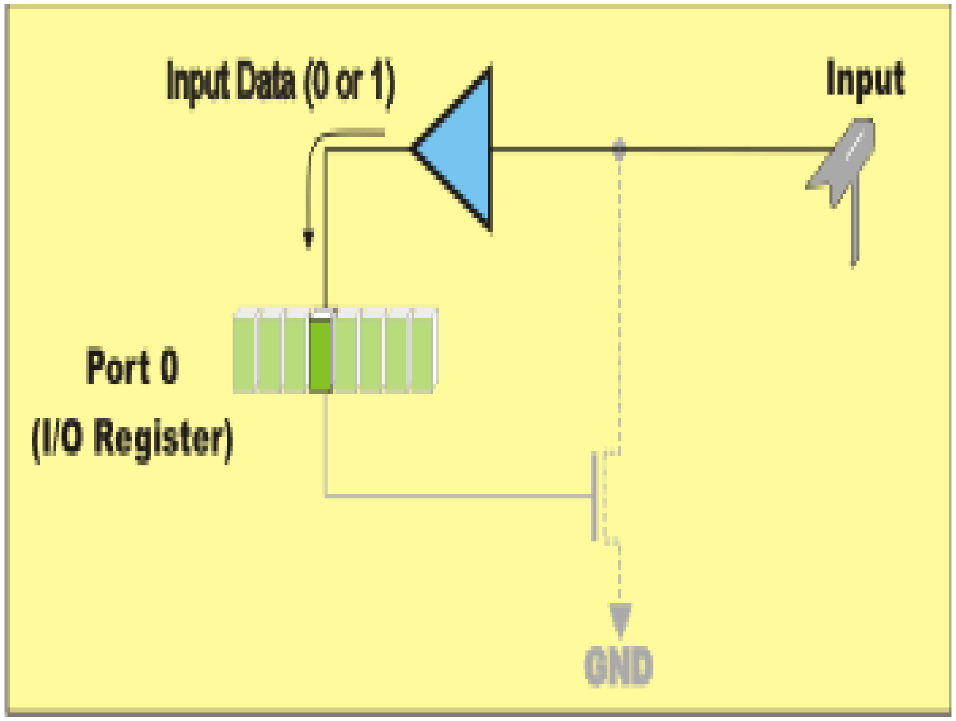
A logic one (1) is applied to a bit of the P register. The output FE transistor is turned off and the appropriate pin remains connected to the power supply voltage over a pull-up resistor of high resistance.

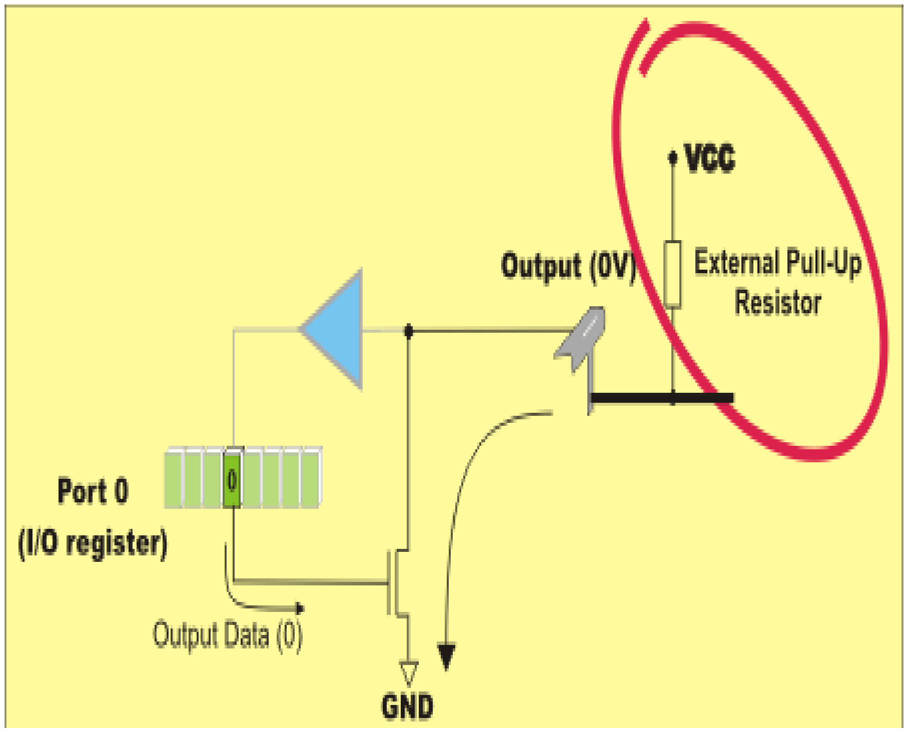


Input/ Output Modules

Port 0

Only in case P0 is used for addressing external memory, the microcontroller will provide internal power supply source in order to supply its pins with logic one. There is no need to add external pull-up resistors.





if they shall be used as outputs with high voltage level (5V), then P0 should be avoided because its pins do not have pull-up resistors, thus giving low logic level only. When using other ports, one should have in mind that pull-up resistors have a relatively high resistance, so that their pins can give a current of several hundreds microamperes only.

Port 1

P1 is a true I/O port, because it doesn't have any alternative functions as is the case with P0, but can be cofigured as general I/O only. It has a pull-up resistor built-in and is completely compatible with TTL circuits.

Port 2

P2 acts similarly to P0 when external memory is used. Pins of this port occupy addresses intended for external memory chip. This time it is about the higher address byte with addresses A8-A15. When no memory is added, this port can be used as a general input/output port showing features similar to P1.

Port 3

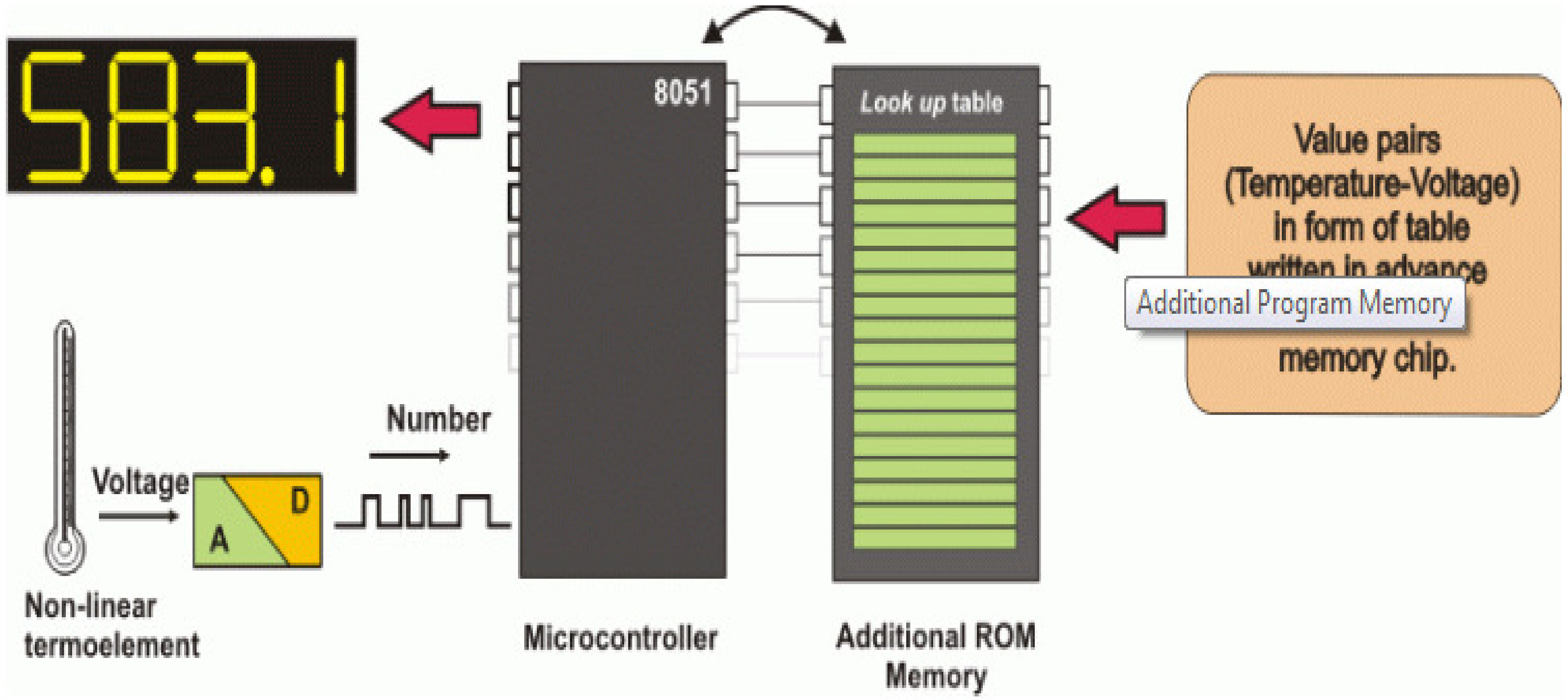
All port pins can be used as general I/O, but they also have an alternative function. In order to use these alternative functions, a logic one (1) must be applied to appropriate bit of the P3 register. In tems of hardware, this port is similar to P0, with the difference that its pins have a pull-up resistor built-in.

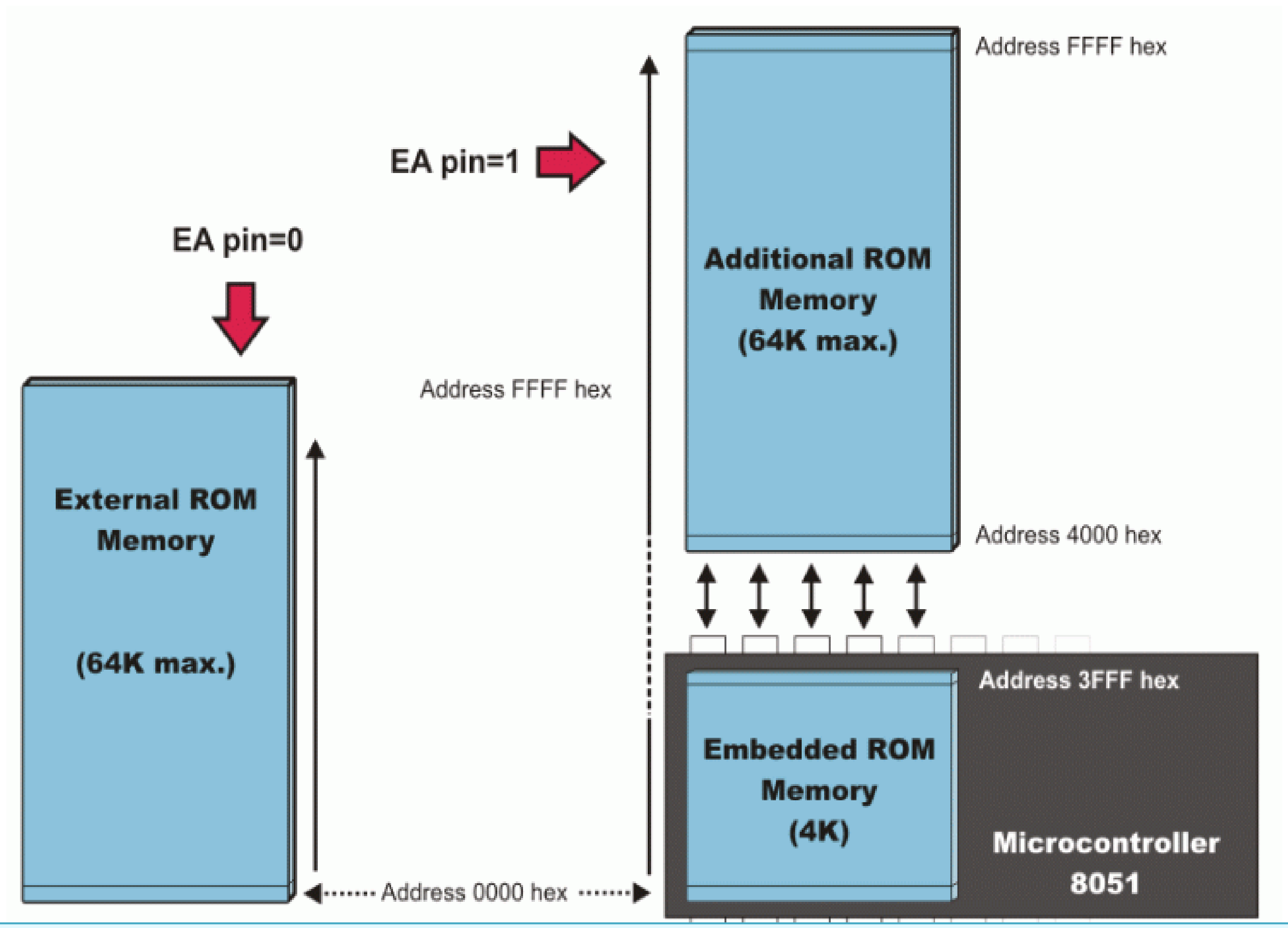
Memory Organization 8051

- Program Memory (ROM)
- Data Memory (RAM)

- ROM is used to permanently save the program being executed,
- RAM is used for temporarily storing data and intermediate results created and used during the operation of the microcontroller.

- 8051 microcontroller family in general at most a few Kb of ROM and 128 or 256 bytes of RAM is used.





EA=0 *In this case, the microcontroller completely ignores internal program memory and executes only the program stored in external memory.*

EA=1 *In this case, the microcontroller executes first the program from built-in ROM, then the program stored in external memory.*

In both cases, P0 and P2 are not available for use since being used for data and address transmission. Besides, the ALE and PSEN pins are also used.

Data Memory

•Data Memory (RAM) is used for temporarily storing data and intermediate results created and used during the operation of the microcontroller and includes many registers such as hardware counters, timers, input/output ports, serial data buffers etc.

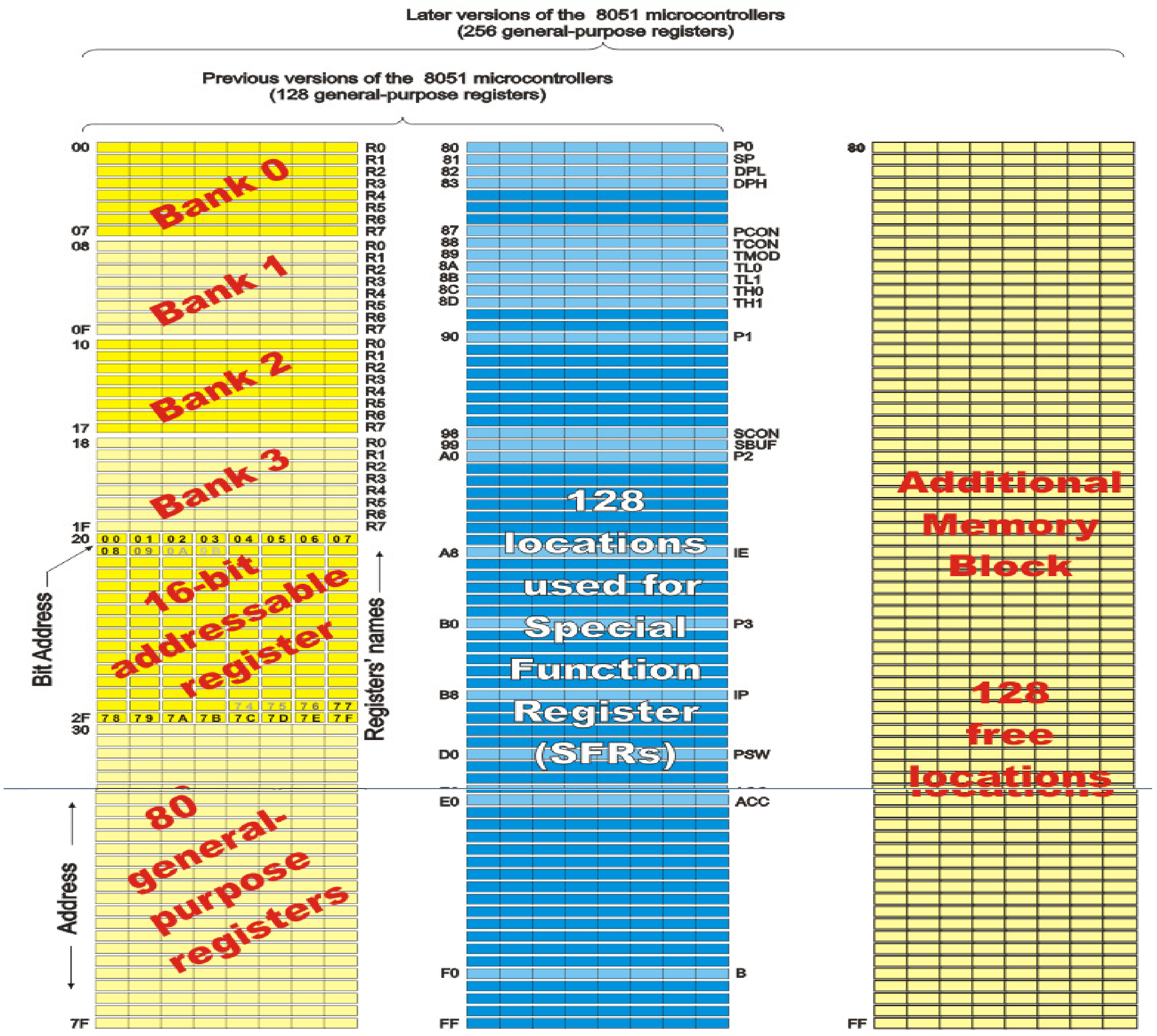
•However, the first 256 memory locations (addresses 0-FFh) are the heart of memory common to all the models belonging to the 8051 family. Locations available to the user occupy memory space with addresses 0-7Fh, i.e. first 128 registers.

•*This part of Data Memory (RAM) is divided in several blocks.*

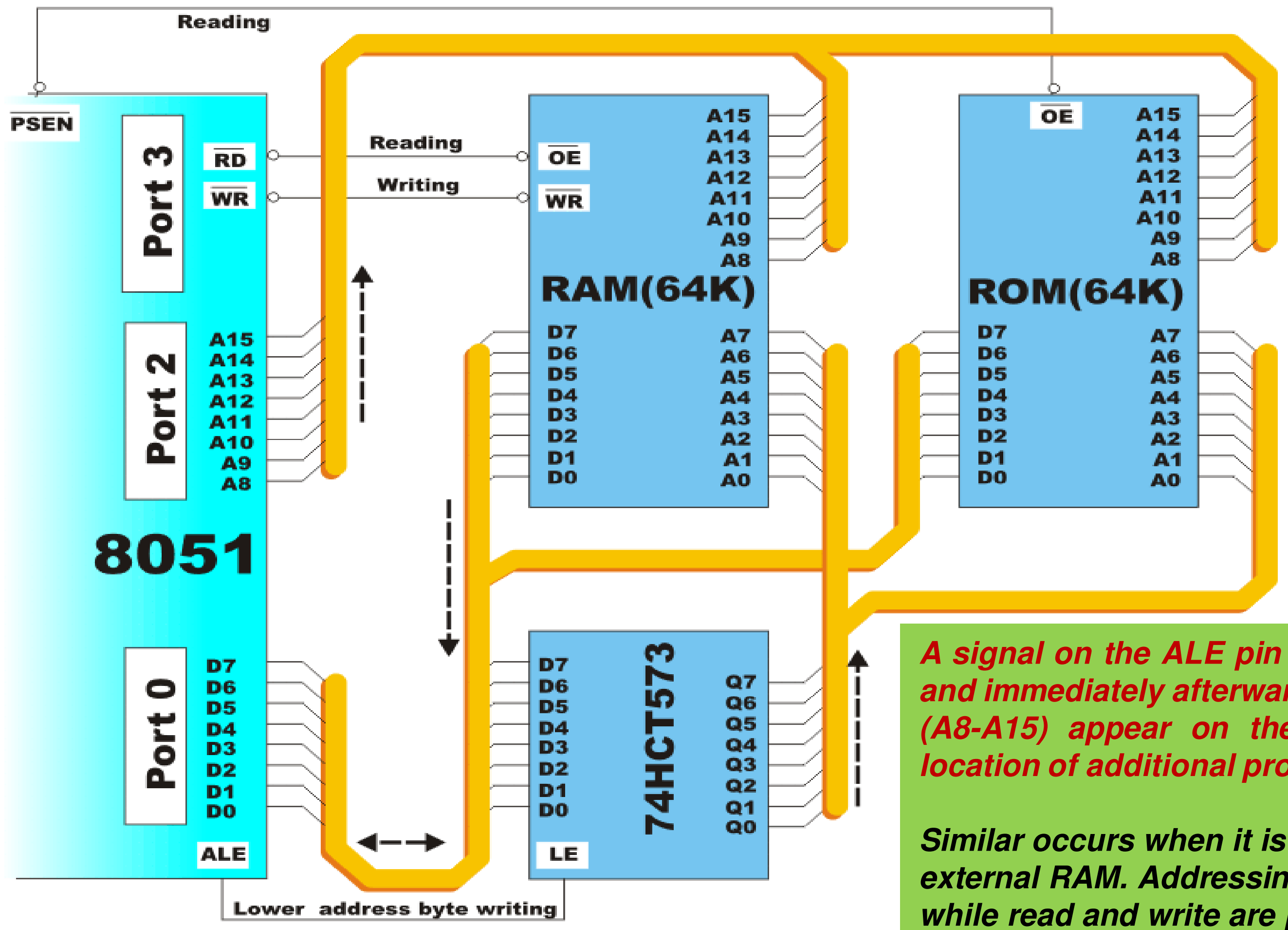
a)The first block consists of 4 banks each including 8 registers denoted by R0-R7. Prior to accessing any of these registers, it is necessary to select the bank containing it.

b)The next memory block (address 20h-2Fh) is bit- addressable, which means that each bit has its own address (0-7Fh). Since there are 16 such registers, this block contains in total of 128 bits with separate addresses (address of bit 0 of the 20h byte is 0, while address of bit 7 of the 2Fh byte is 7Fh).

c)The third group of registers occupy addresses 2Fh-7Fh, i.e. 80 locations, and does not have any special functions or features.


Later versions of the 8051 microcontrollers
(256 general-purpose registers)
Previous versions of the 8051 microcontrollers
(128 general-purpose registers)
00
07
08
0F
10
17
18
1F
20
2F
30
7F
R0
R1
R2
R3
R4
R5
R6
R7
Bank 0
Bank 1
Bank 2
Bank 3
00 01 02 03 04 05 06 07
08 09 0A 0B
74 75 76 77
78 79 7A 7B 7C 7D 7E 7F
16-bit addressable register
Bit Address
Registers' names
80 general-purpose registers
Address
80 P0
81 SP
82 DPL
83 DPH
87 PCON
88 TCON
89 TMOD
8A TL0
8B TL1
8C TH0
8D TH1
90 P1
98 SCON
99 SBUF
A0 P2
A8 IE
B0 P3
B8 IP
D0 PSW
E0 ACC
F0 B
FF
128 locations used for Special Function Register (SFRs)
80
FF
Additional Memory Block
128 free locations

Memory expansion for 8051 Microcontroller



When the program during execution encounters an instruction which resides in external memory (ROM), the microcontroller will activate its control output ALE and set the first 8 bits of address (A0-A7) on P0. Port P0 pins are configured as inputs, the PSEN pin is activated and the microcontroller reads from memory chip.

A signal on the ALE pin latches the IC circuit 74HCT573 and immediately afterwards 8 higher bits of address (A8-A15) appear on the port. In this way, a desired location of additional program memory is addressed.

Similar occurs when it is necessary to read location from external RAM. Addressing is performed in the same way, while read and write are performed via signals appearing on the control outputs RD (is short for read) or WR (is short for write).

In case memory (RAM or ROM) built in the microcontroller is not sufficient, it is possible to add two external memory chips with capacity of 64Kb each. P2 and P3 I/O ports are used for their addressing and data transmission.

Addressing

Two ways of addressing are used for all 8051 microcontrollers the processor processes data as per program instructions depending on which part of memory should be accessed.

a) Direct Addressing

b) Indirect Addressing

Direct Addressing

On direct addressing, the address of memory location containing data to be read is specified in instruction.

For example: Since the address is only one byte in size (the largest number is 255), only the first 255 locations of RAM can be accessed this way. The first half of RAM is available for use, while another half is reserved for SFRs.

```
MOV A,33h; Means: move a number from address 33 hex. to accumulator
```

Indirect Addressing: On indirect addressing, a register containing the address of another register is specified in instruction. Data to be used in the program is stored in the letter register.

For example: Indirect addressing is only used for accessing RAM locations available for use from additional memory block (128 locations of RAM) (never for accessing SFRs).Simply put, when the program encounters instruction including "@" sign and if the specified address is higher than 128 (7F hex.), the processor knows that indirect addressing is used and skips memory space reserved for SFRs.

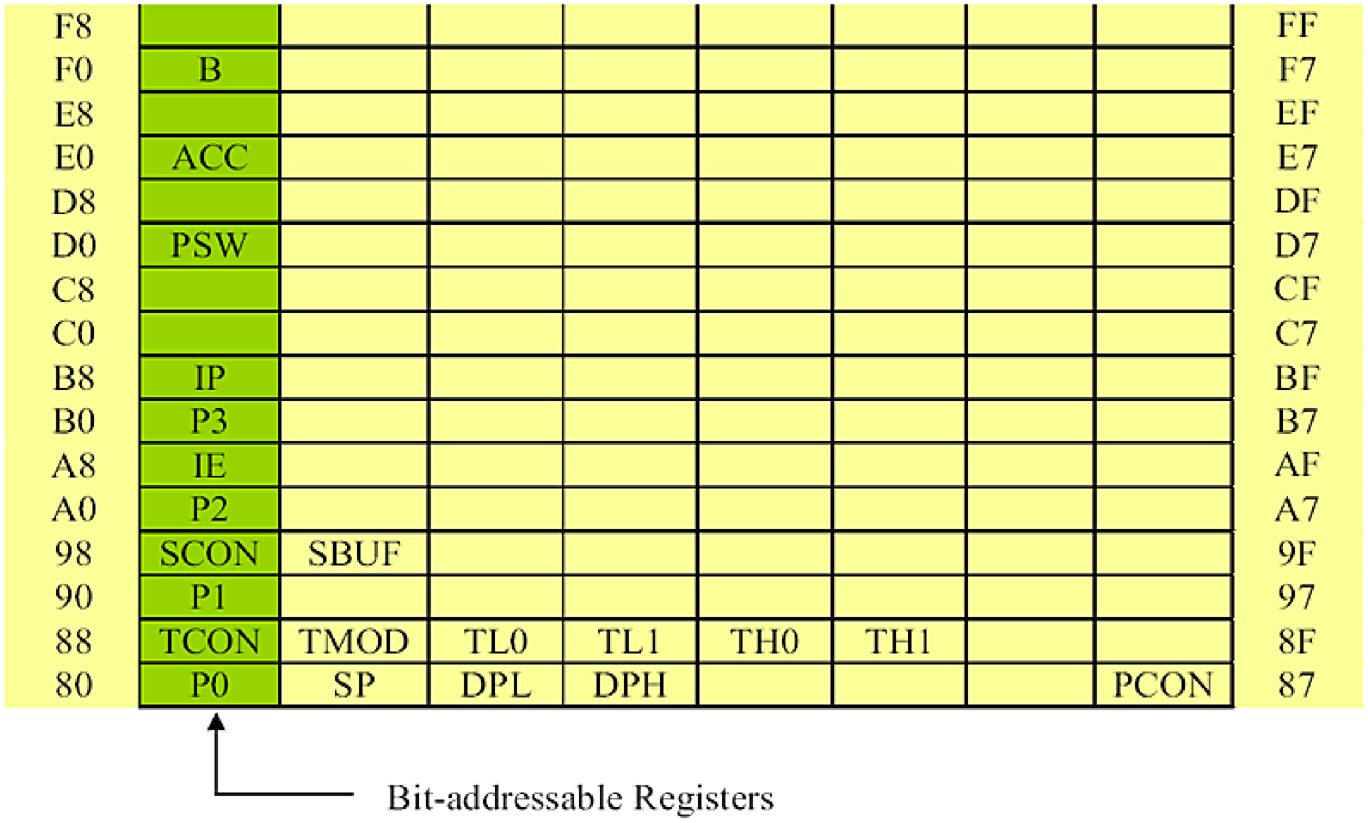
```
MOV A,@R0; Means: Store the value from the register whose address is in
the R0 register into accumulator
```

Special Function Registers (SFRs)

•Special Function Registers (SFRs) are used for running and monitoring the operation of the microcontroller.

•Each of these registers as well as each bit they include, has its name, address in the scope of RAM and precisely defined purpose such as timer control, interrupt control, serial communication control etc.

•Even though there are 128 memory locations intended to be occupied by them. Rest of locations are intentionally left unoccupied in order to enable the manufacturers to further develop microcontrollers keeping them compatible with the previous versions.



F8									FF
F0	B								F7
E8									EF
E0	ACC								E7
D8									DF
D0	PSW								D7
C8									CF
C0									C7
B8	IP								BF
B0	P3								B7
A8	IE								AF
A0	P2								A7
98	SCON	SBUF							9F
90	P1								97
88	TCON	TMOD	TL0	TL1	TH0	TH1			8F
80	P0	SP	DPL	DPH				PCON	87

Bit-addressable Registers

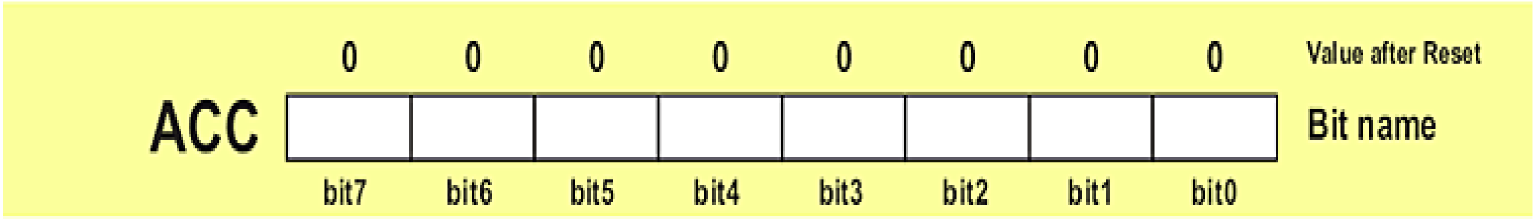
Accumulator Register (ACC)

•A register is a general-purpose register used for storing intermediate results obtained during operation.

•Prior to executing an instruction upon any number or operand it is necessary to store it in the accumulator first.

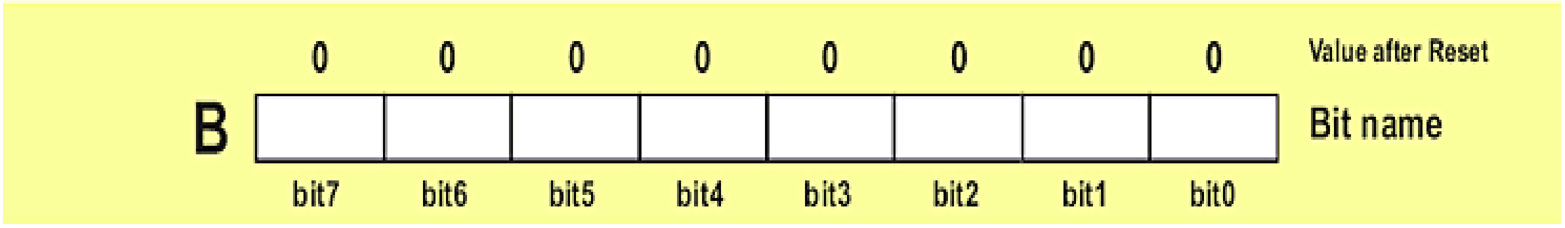
•All results obtained from arithmetical operations performed by the ALU are stored in the accumulator.

•Data to be moved from one register to another must go through the accumulator. In other words, the A register is the most commonly used register and it is impossible to imagine a microcontroller without it. More than half instructions used by the 8051 microcontroller use somehow the accumulator.



B Register

Multiplication and division can be performed only upon numbers stored in the A and B registers. All other instructions in the program can use this register as a spare accumulator (A).



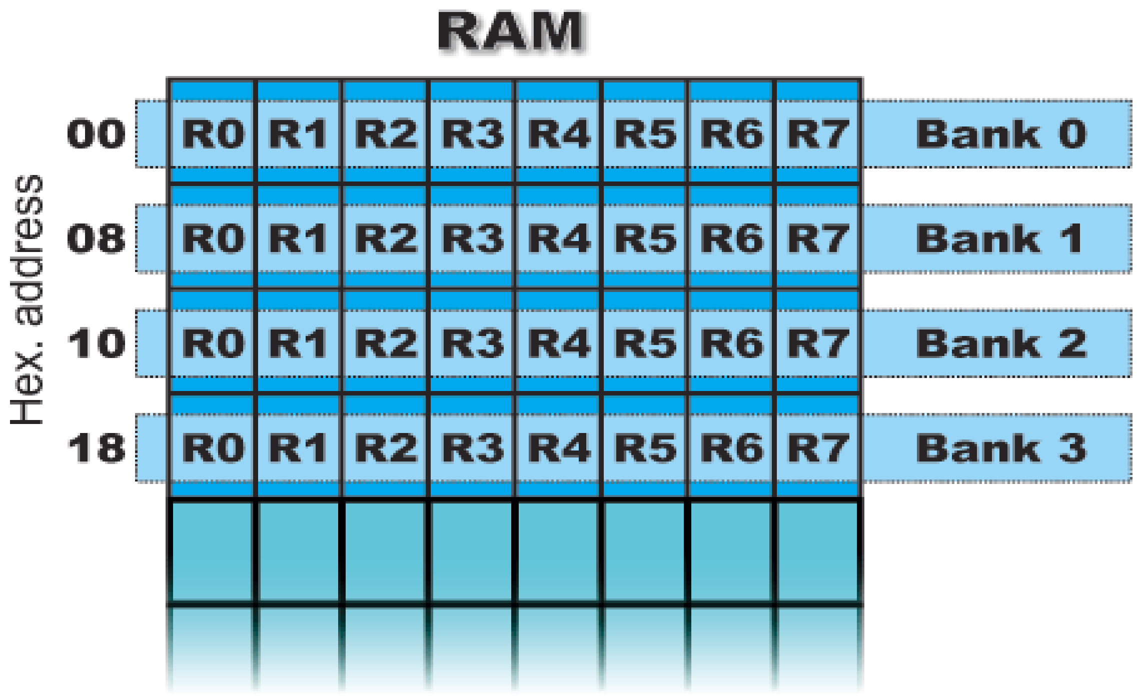
R Registers (R0-R7)

•This is a common name for 8 general-purpose registers (R0, R1, R2 ...R7).

•Even though they are not true SFRs, they deserve to be discussed here because of their purpose.

•They occupy 4 banks within RAM.

•Similar to the accumulator, they are used for temporary storing variables and intermediate results during operation. Which one of these banks is to be active depends on two bits of the program status word (PSW) Register. Active bank is a bank the registers of which are currently used.



The following example best illustrates the purpose of these registers.

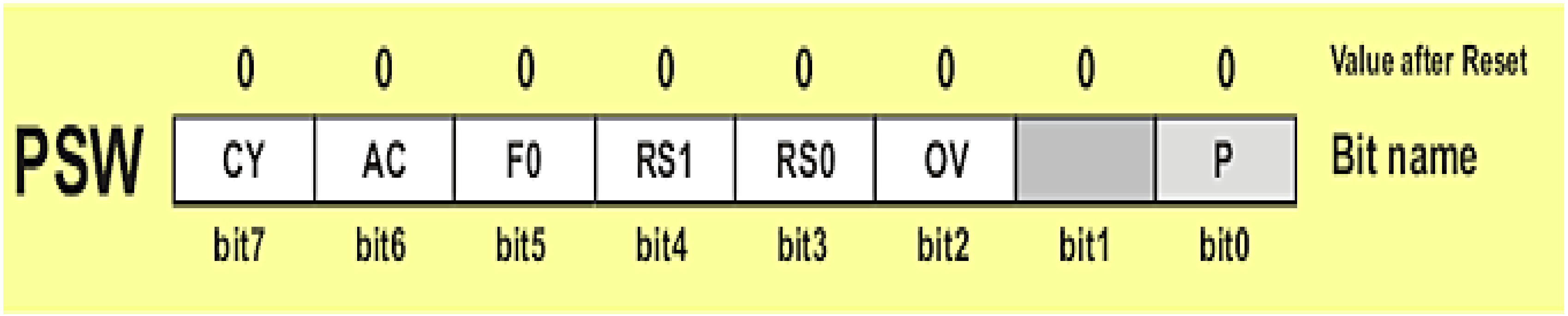
Example

Suppose it is necessary to perform some arithmetical operations upon numbers previously stored in the R registers: (R1+R2) - (R3+R4). Obviously, a register for temporary storing results of addition is needed. This is how it looks in the program:

```
MOV A,R3; Means: move number from R3 into accumulator

ADD A,R4; Means: add number from R4 to accumulator (result remains in accumulator)

MOV R5,A; Means: temporarily move the result from accumulator into R5

MOV A,R1; Means: move number from R1 to accumulator

ADD A,R2; Means: add number from R2 to accumulator

SUBB A,R5; Means: subtract number from R5 (there are R3+R4)
```

Program Status Word (PSW) Register

- **PSW register is one of the most important SFRs.**
- ***It contains several status bits that reflect the current state of the CPU. Besides, this register contains Carry bit (CY), Auxiliary Carry (AC), two register bank select bits (RS1 & RS0), overflow flag (OV), parity bit and user-definable status flag.***
- **P - Parity bit.** **If a number stored in the accumulator is even then this bit will be automatically set (1), otherwise it will be cleared (0). It is mainly used during data transmit and receive via serial communication.**
- **Bit 1. This bit is intended to be used in the future versions of microcontrollers.**
- **OV Overflow** **occurs when the result of an arithmetical operation is larger than 255 and cannot be stored in one register. Overflow condition causes the OV bit to be set (1). Otherwise, it will be cleared (0).**
- **RS0, RS1 - Register bank select bits.** **These two bits are used to select one of four register banks of RAM. By setting and clearing these bits, registers R0-R7 are stored in one of four banks of RAM.**
- **F0 - Flag 0.** **This is a general-purpose bit available for use.**
- **AC - Auxiliary Carry Flag** **is used for BCD operations only.**
- **CY - Carry Flag** **is the (ninth) auxiliary bit used for all arithmetical operations and shift instructions.**



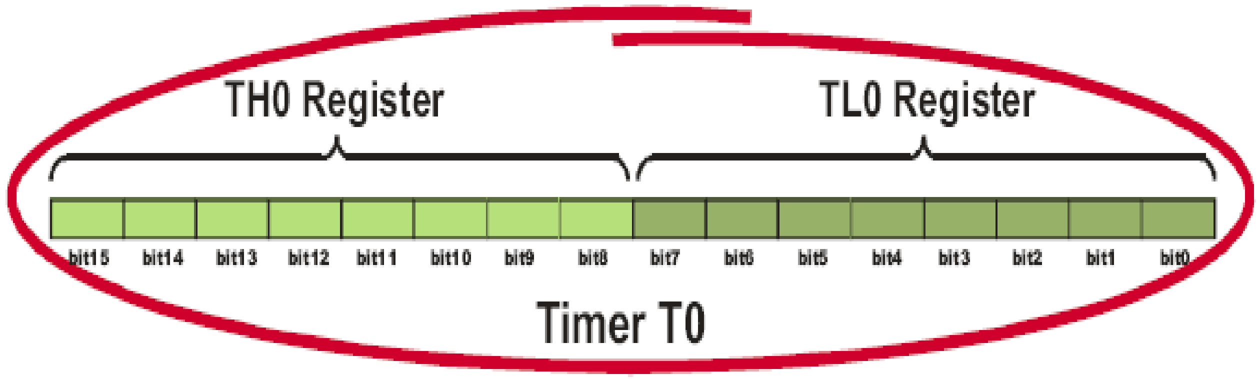
Counters and Timers

•The 8051 microcontroller has 2 timers/counters called T0 and T1.

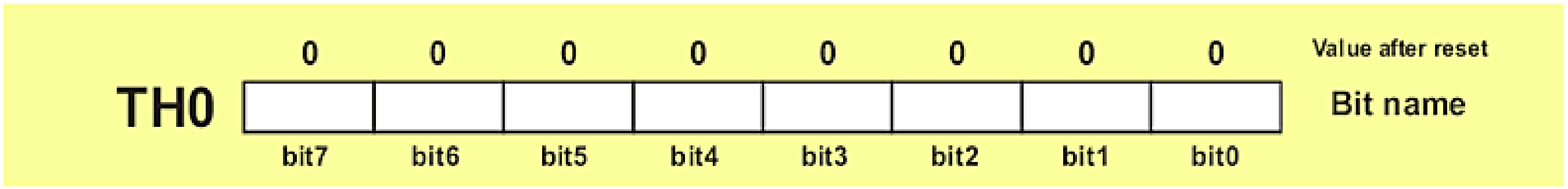
•As their names suggest, their main purpose is to measure time and count external events. Besides, they can be used for generating clock pulses to be used in serial communication, so called Baud Rate.

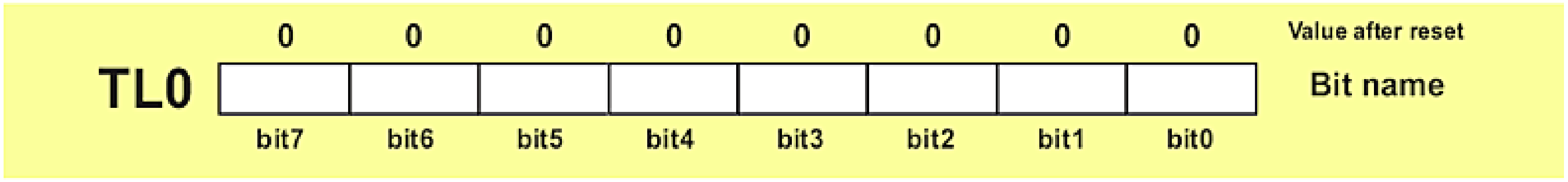
Timer T0

As seen in figure below, the timer T0 consists of two registers – TH0 and TL0 representing a low and a high byte of one 16-digit binary number.



Accordingly, if the content of the timer T0 is equal to 0 (T0=0) then both registers it consists of will contain 0. If the timer contains for example number 1000 (decimal), then the TH0 register (high byte) will contain the number 3, while the TL0 register (low byte) will contain decimal number 232.



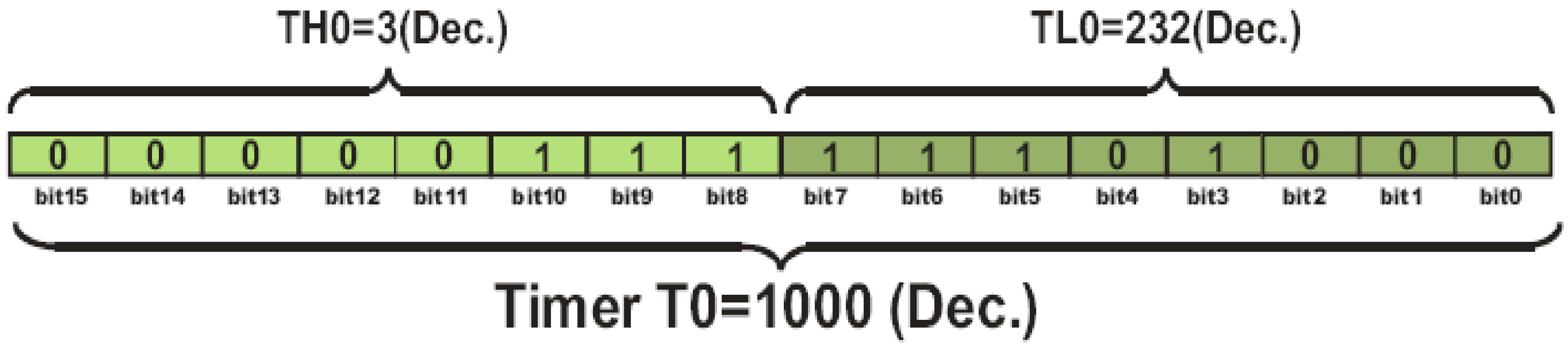


Formula used to calculate values in these two registers is very simple:

TH0 × 256 + TL0 = T

Matching the previous example it would be as follows:

3 × 256 + 232 = 1000



TH0=3(Dec.)
TL0=232(Dec.)
0 0 0 0 0 1 1 1 1 1 1 0 1 0 0 0
bit15 bit14 bit13 bit12 bit11 bit10 bit9 bit8 bit7 bit6 bit5 bit4 bit3 bit2 bit1 bit0
Timer T0=1000 (Dec.)

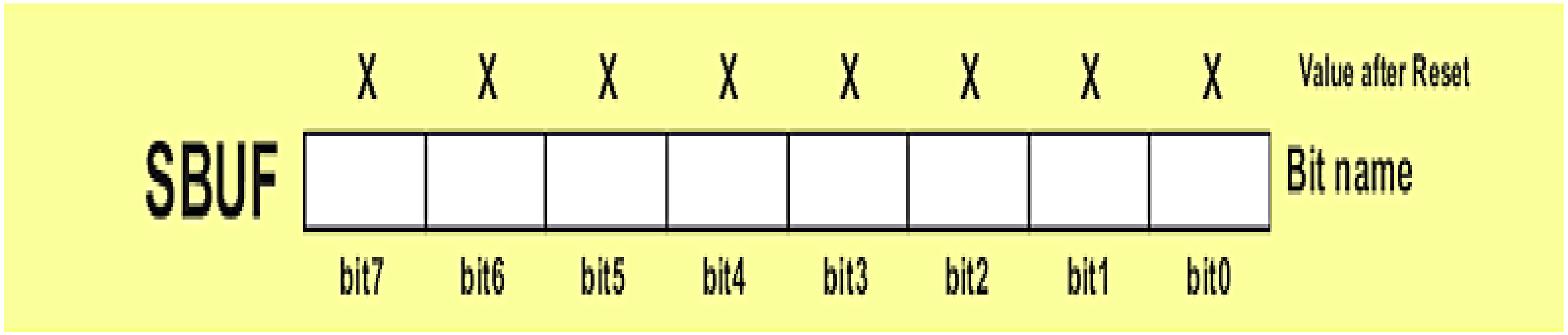
Since the timer T0 is virtually 16-bit register, the largest value it can store is 65 535. In case of exceeding this value, the timer will be automatically cleared and counting starts from 0. This condition is called an overflow. Two registers TMOD and TCON are closely connected to this timer and control its operation.

UART (Universal Asynchronous Receiver and Transmitter)

•One of the microcontroller features making it so powerful is an integrated UART, better known as a serial port.

•It is a full-duplex port, thus being able to transmit and receive data simultaneously and at different baud rates/pulse rate.

• When using UART, all the programmer has to do is to simply select serial port mode and baud rate. When it's done, serial data transmit is nothing but writing to the SBUF register, while data receive represents reading the same register.

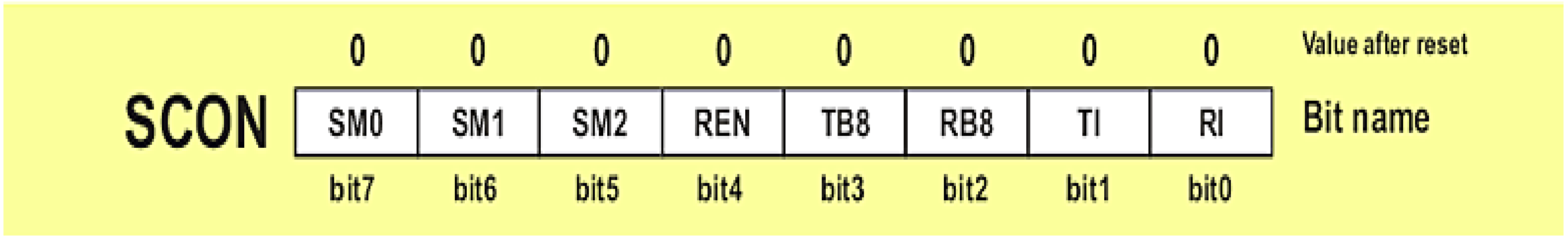


•Serial port must be configured prior to being used.

•In other words, it is necessary to determine how many bits is contained in one serial “word”, baud rate and synchronization clock source.

•The whole process is in control of the bits of the SCON register (Serial Control).

Serial Port Control (SCON) Register



	0	0	0	0	0	0	0	0	Value after reset
SCON	SM0	SM1	SM2	REN	TB8	RB8	TI	RI	Bit name
	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	

SM0 - ***Serial port mode bit 0 is used for serial port mode selection.***

SM1 - ***Serial port mode bit 1.***

SM2 - ***Serial port mode 2 bit, also known as multiprocessor communication enable bit.***

REN - ***Reception Enable bit enables serial reception when set. When cleared, serial reception is disabled.***

TB8 - ***Transmitter bit 8. Since all registers are 8-bit wide, this bit solves the problem of transmitting the 9th bit in modes 2 and 3. It is set to transmit a logic 1 in the 9th bit.***

RB8 - ***Receiver bit 8 or the 9th bit received in modes 2 and 3. Cleared by hardware if 9th bit received is a logic 0. Set by hardware if 9th bit received is a logic 1.***

TI - ***Transmit Interrupt flag is automatically set at the moment the last bit of one byte is sent. It's a signal to the processor that the line is available for a new byte transmit. It must be cleared from within the software.***

RI - ***Receive Interrupt flag is automatically set upon one byte receive. It signals that byte is received and should be read quickly prior to being replaced by a new data. This bit is also cleared from within the software.***

Microcontroller Interrupts

- There are five interrupt sources for the 8051, which means that they can recognize 5 different events that can interrupt regular program execution.

- Each interrupt can be enabled or disabled by setting bits of the IE register. Likewise, the whole interrupt system can be disabled by clearing the EA bit of the same register. Refer to figure below.

- Now, external interrupts- INT0 and INT1. If the IT0 and IT1 bits of the TCON register are set, an interrupt will be generated on high to low transition, i.e. on the falling pulse edge (only in that moment). If these bits are cleared, an interrupt will be continuously executed as far as the pins are held low.

	0	X	0	0	0	0	0	0	Value after Reset
IE	EA		ET2	ES	ET1	EX1	ET0	EX0	Bit name
	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	

EA - global interrupt enable/disable:

0 - disables all interrupt requests.
1 - enables all individual interrupt requests.

ES - enables or disables serial interrupt:

0 - UART system cannot generate an interrupt.
1 - UART system enables an interrupt.

ET1 - bit enables or disables Timer 1 interrupt:

0 - Timer 1 cannot generate an interrupt.
1 - Timer 1 enables an interrupt.

EX1 - *bit enables or disables external 1 interrupt:*

- **0 - change of the pin INT0 logic state cannot generate an interrupt.**
- **1 - enables an external interrupt on the pin INT0 state change.**

ET0 - *bit enables or disables timer 0 interrupt:*

- **0 - Timer 0 cannot generate an interrupt.**
- **1 - enables timer 0 interrupt.**

EX0 - *bit enables or disables external 0 interrupt:*

- **0 - change of the INT1 pin logic state cannot generate an interrupt.**
- **1 - enables an external interrupt on the pin INT1 state change.**

IP Register (Interrupt Priority)

The IP register bits specify the priority level of each interrupt (high or low priority).

	X	X	0	0	0	0	0	0	Value after Reset
IP			PT2	PS	PT1	PX1	PT0	PX0	Bit name
	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	

PS - Serial Port Interrupt priority bit
- Priority 0
- Priority 1

PT1 - Timer 1 interrupt priority
- Priority 0
- Priority 1

PX1 - External Interrupt INT1 priority
- Priority 0
- Priority 1

PT0 - Timer 0 Interrupt Priority
- Priority 0
- Priority 1

PX0 - External Interrupt INT0 Priority
- Priority 0
- Priority 1

Microcontroller Power Consumption Control

•Generally speaking, the microcontroller is inactive for the most part and just waits for some external signal in order to takes its role in a show.

•In extreme cases, the only solution is to set the whole electronics in sleep mode in order to minimize consumption.

• A typical example is a TV remote controller: it can be out of use for months but when used again it takes less than a second to send a command to TV receiver.

•Actually, there are two power-saving modes of operation: *Idle* and *Power Down.*

Idle mode

•Upon the IDL bit of the PCON register is set, the microcontroller turns off the greatest power consumer- CPU unit while peripheral units such as serial port, timers and interrupt system continue operating normally consuming 6.5mA.

• In Idle mode, the state of all registers and I/O ports remains unchanged.

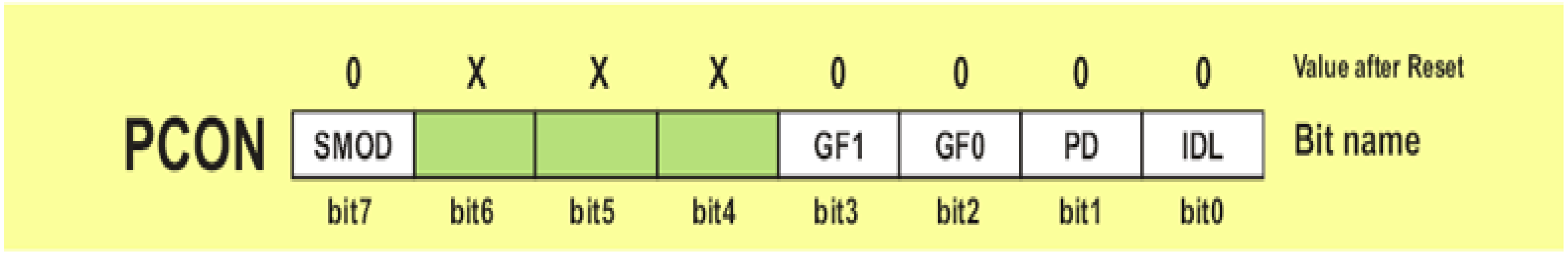
•In order to exit the Idle mode and make the microcontroller operate normally, it is necessary to enable and execute any interrupt or reset. It will cause the IDL bit to be automatically cleared and the program resumes operation from instruction having set the IDL bit. It is recommended that first three instructions to execute now are NOP instructions. They don't perform any operation but provide some time for the microcontroller to stabilize and prevents undesired changes on the I/O ports.

Power Down mode

•**By setting the PD bit of the PCON register from within the program, the microcontroller is set to Power down mode, thus turning off its internal oscillator and reduces power consumption enormously.**

•**The microcontroller can operate using only 2V power supply in power- down mode, while a total power consumption is less than 40uA. The only way to get the microcontroller back to normal mode is by reset.**

•**While the microcontroller is in Power Down mode, the state of all SFR registers and I/O ports remains unchanged. By setting it back into the normal mode, the contents of the SFR register is lost, but the content of internal RAM is saved. Reset signal must be long enough, approximately 10mS, to enable stable operation of the quartz oscillator.**



The purpose of the Register PCON bits is:

•SMOD Baud rate is twice as much higher by setting this bit.
•GF1 General-purpose bit (available for use).
•GF1 General-purpose bit (available for use).
•GF0 General-purpose bit (available for use).
•PD By setting this bit the microcontroller enters the *Power Down* mode.
•IDL By setting this bit the microcontroller enters the *Idle* mode.

Programming a microcontroller

•In order to transfer a “hex code” to the microcontroller, it is necessary to provide a cable for serial communication and a special device, called programmer, with software. There are several ways to do it.

•A large number of programs and electronic circuits having this purpose can be found on the Internet.

Do as follows:

a)open hex code document, set a few parameters and click the icon for compiling.

b)After a while, a sequence of zeros and ones will be programmed into the microcontroller through the serial connection cable and programmer hardware.

•What's left is to place the programmed chip into the target device. In the event that it is necessary to make some changes in the program, the previous procedure may be repeated an unlimited number of times.

Writing program in assembly language, (simulator tool), compiling to machine code
Copy program to MCU
Operating of MCU in real device
Copying program to a microcontroller

Applications Microcontroller

- It is used in any smart system
- Suitable for light weight electronic devices
- Microcontrollers are vastly used in robotics
- Are used to control the robots
- Can be automated or outside controlled
- Limited intelligence
- Used in microwaves to automobiles
- Electronic locks
- Calculating devices
- Automatic ticketing system
- And many more