Microcontrollers Notes for B.E. IV Sem ECE/ETE Students

Visvesvaraya Technological University (VTU), Belagavi
by
Saneesh Cleatus Thundiyil
Associate Professor,
Department of Electronics and Communication Engineering,
BMS Institute of Technology and Management
Bengaluru – 64

SYLLABUS

MODULE I

8051 Microcontroller: Microprocessor Vs Microcontroller, Embedded Systems, Embedded Microcontrollers, 8051 Architecture- Registers, Pin diagram, I/O ports functions, Internal Memory organization. External Memory (ROM & RAM) interfacing.

MODULE II

8051 Instruction Set: Addressing Modes, Data Transfer instructions, Arithmetic instructions, Logical instructions, Branch instructions, Bit manipulation instructions. Simple Assembly language program examples (without loops) to use these instructions.

MODULE III

8051 Stack, I/O Port Interfacing and Programming: 8051 Stack, Stack and Subroutine instructions. Assembly language program examples on subroutine and involving loops. Interfacing simple switch and LED to I/O ports to switch on/off LED with respect to switch status.

MODULE IV

8051 Timers and Serial Port: 8051 Timers and Counters – Operation and Assembly language programming to generate a pulse using Mode-1 and a square wave using Mode-2 on a port pin. 8051 Serial Communication- Basics of Serial Data Communication, RS-232 standard, 9 pin RS232 signals, Simple Serial Port programming in Assembly and C to transmit a message and to receive data serially.

MODULE V

8051 Interrupts and Interfacing Applications: 8051 Interrupts. 8051 Assembly language programming to generate an external interrupt using a switch, 8051 C programming to generate a square waveform on a port pin using a Timer interrupt. Interfacing 8051 to ADC-0804, DAC, LCD and Stepper motor and their 8051 Assembly language interfacing programming.

Text Books:

1. "The 8051 Microcontroller and Embedded Systems – using assembly and C", Muhammad Ali Mazidi and Janice Gillespie Mazidi and Rollin D. McKinlay; PHI, 2006 / Pearson, 2006.
2. "The 8051 Microcontroller", Kenneth J. Ayala, 3rd Edition, Thomson/Cengage Learning.

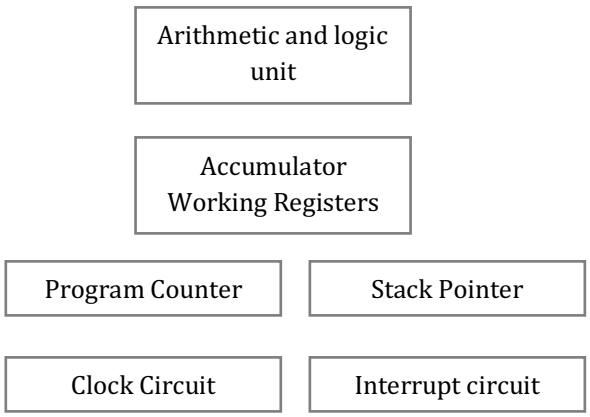
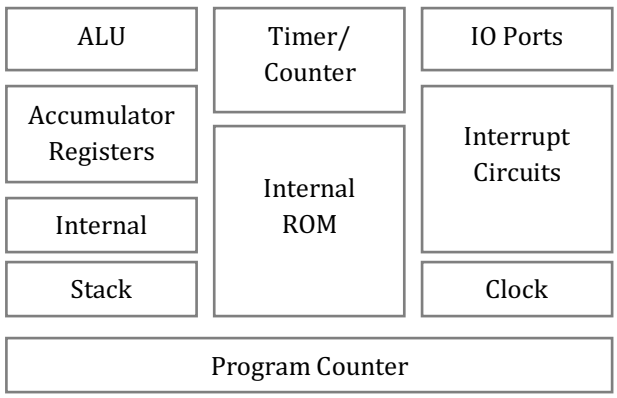
Reference Books:

1. "The 8051 Microcontroller Based Embedded Systems", Manish K Patel, McGraw Hill, 2014, ISBN: 978-93-329-0125-4.
2. "Microcontrollers: Architecture, Programming, Interfacing and System Design", Raj Kamal, Pearson Education, 2005.

MODULE - 1

Syllabus: 8051 Microcontroller: Microprocessor Vs Microcontroller, Embedded Systems, Embedded Microcontrollers, 8051 Architecture- Registers, Pin diagram, I/O ports functions, Internal Memory organization. External Memory (ROM & RAM) interfacing.

1.1 MICROPROCESSORS vs MICROCONTROLLERS

Microprocessor	Microcontroller
	
Block diagram of microprocessor	*Block diagram of microcontroller*
Microprocessor contains ALU, General purpose registers, stack pointer, program counter, clock timing circuit, interrupt circuit	Microcontroller contains the circuitry of microprocessor, and in addition it has built in ROM, RAM, I/O Devices, Timers/Counters etc.
It has many instructions to move data between memory and CPU	It has few instructions to move data between memory and CPU
Few bit handling instruction	It has many bit handling instructions
Less number of pins are multifunctional	More number of pins are multifunctional
Single memory map for data and code (program)	Separate memory map for data and code (program)
Access time for memory and IO are more	Less access time for built in memory and IO.
Microprocessor based system requires additional hardware	It requires less additional hardwares
More flexible in the design point of view	Less flexible since the additional circuits which is residing inside the microcontroller is fixed for a particular microcontroller
Large number of instructions with flexible addressing modes	Limited number of instructions with few addressing modes

1.2 EMBEDDED SYSTEMS AND EMBEDDED MICROCONTROLLERS.

An embedded system uses a microprocessor (or microcontroller) to do one task. There is only one application software that is typically burned into ROM. An embedded system can be an independent system or it can be a part of a large system. For example, a fire alarm is an embedded system; it will sense smoke and activates the alarm/sprinkler.

An embedded system has hardware, application software and sometimes Real Time Operating system (RTOS) that supervises the application software. A small-scale embedded system may not have RTOS. Hence, we can define an embedded system as a Microcontroller based, software driven, reliable, real-time control system.

Characteristics of an Embedded System

- ***Single-functioned* –** An embedded system usually performs a specialized operation and does the same repeatedly. For example: A fire alarm only detects smokes and activates sound/sprinkler
- ***Tightly constrained* –** Embedded system has tight design metrics. It must be of a size to fit on a single chip, must perform fast enough to process data in real time and consume minimum power to extend battery life.
- ***Reactive and Real time* –** Many embedded systems must continually react to changes in the system's environment and must compute certain results in real time without any delay.
- ***Memory* –** It must have a memory, as its software usually embeds in ROM.
- ***Connected* –** It must have connected peripherals to connect input and output devices.
- ***HW-SW systems* –** Software is used for more features and flexibility. Hardware is used for performance and security.

Embedded controller is the heart of an embedded system. It is the basic unit that takes inputs and produces an output after processing the data. For an embedded system designer, it is necessary to have the knowledge of both microprocessors and microcontrollers.

An embedded controller has two essential units − Program Flow Control Unit (CU), Execution Unit (EU), The CU includes a fetch unit for fetching instructions from the memory. The EU has circuits that implement the instructions pertaining to data transfer operation and data conversion from one form to another. The EU includes the Arithmetic and Logical Unit (ALU) and also the circuits that execute instructions for a program control task such as interrupt, or jump to another set of instructions.

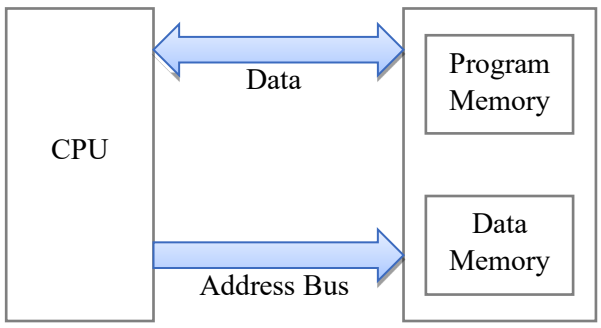
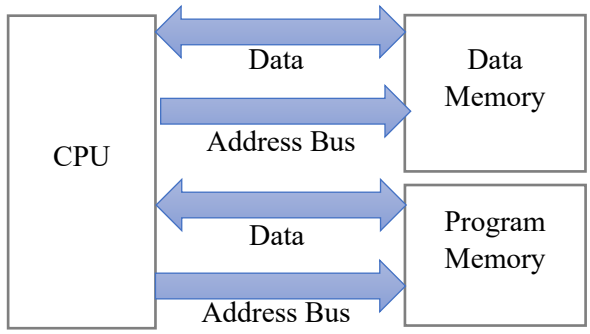
Microprocessor, microcontroller, DSP processor etc are different types of embedded controllers.

RISC AND CISC CPU ARCHITECTURES

Microcontrollers with small instruction set are called reduced instruction set computer (RISC) machines and those with complex instruction set are called complex instruction set computer (CISC). Intel 8051 is an example of CISC machine whereas microchip PIC 18F87X is an example of RISC machine.

RISC	***CISC***
Instruction takes one or two cycles	*Instruction takes multiple cycles*
Only load/store instructions are used to access memory	*In additions to load and store instructions, memory access is possible with other instructions also.*
Instructions executed by hardware	*Instructions executed by the micro program*
Fixed format instruction	*Variable format instructions*
Few addressing modes	*Many addressing modes*
Few instructions	*Complex instruction set*
Most of the have multiple register banks	*Single register bank*
Highly pipelined	*Less pipelined*
Complexity is in the compiler	*Complexity in the microprogram*

HARVARD & VON- NEUMANN CPU ARCHITECTURE

Von-Neumann (Princeton architecture)	Harvard architecture
	
Von-Neumann (Princeton architecture)	Harvard architecture
It uses single memory space for both instructions and data.	It has separate program memory and data memory
It is not possible to fetch instruction code and data	Instruction code and data can be fetched simultaneously
Execution of instruction takes more machine cycle	Execution of instruction takes less machine cycle
Uses CISC architecture	Uses RISC architecture
Instruction pre-fetching is a main feature	Instruction parallelism is a main feature
Also known as control flow or control driven computers	Also known as data flow or data driven computers
Simplifies the chip design because of single memory space	Chip design is complex due to separate memory space
Eg. 8085, 8086, MC6800	Eg. General purpose microcontrollers, special DSP chips etc.

1.3 8051 ARCHITECTURE

Salient features of 8051 microcontroller are given below.

- Eight bit CPU
- On chip clock oscillator
- 4 Kbytes of internal program memory (code memory) [ROM]
- 128 bytes of internal data memory [RAM]
- 64 Kbytes of external program memory address space.
- 64 Kbytes of external data memory address space.
- 32 bi directional I/O lines (can be used as four 8 bit ports or 32 individually addressable I/O lines)
- Two 16 Bit Timer/Counter :T0, T1
- Full Duplex serial data receiver/transmitter
- Four Register banks with 8 registers in each bank.
- 16 bit Program counter (PC) and a data pointer (DPTR)
- 8 Bit Program Status Word (PSW)
- 8 Bit Stack Pointer
- 5 vector interrupt structure (When RESET is not considered as an interrupt)
- 8051 CPU consists of 8 bit ALU with associated registers like accumulator 'A' , B register, PSW, SP, 16 bit program counter, stack pointer.
- ALU can perform arithmetic and logic functions on 8 bit variables.
- 8051 has 128 bytes of internal RAM which is divided into
    - Working registers [00 – 1F]
    - Bit addressable memory area [20 – 2F]
    - General purpose memory area (Scratch pad memory) [30-7F]

THE 8051 ARCHITECTURE.

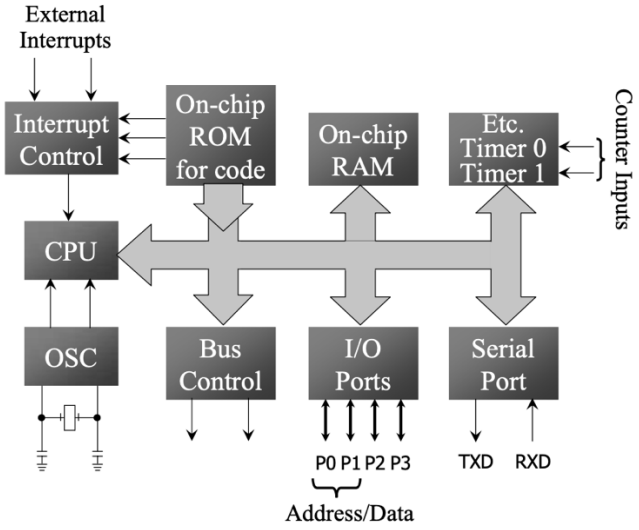


Figure 1.a – 8051 Simplified Architecture

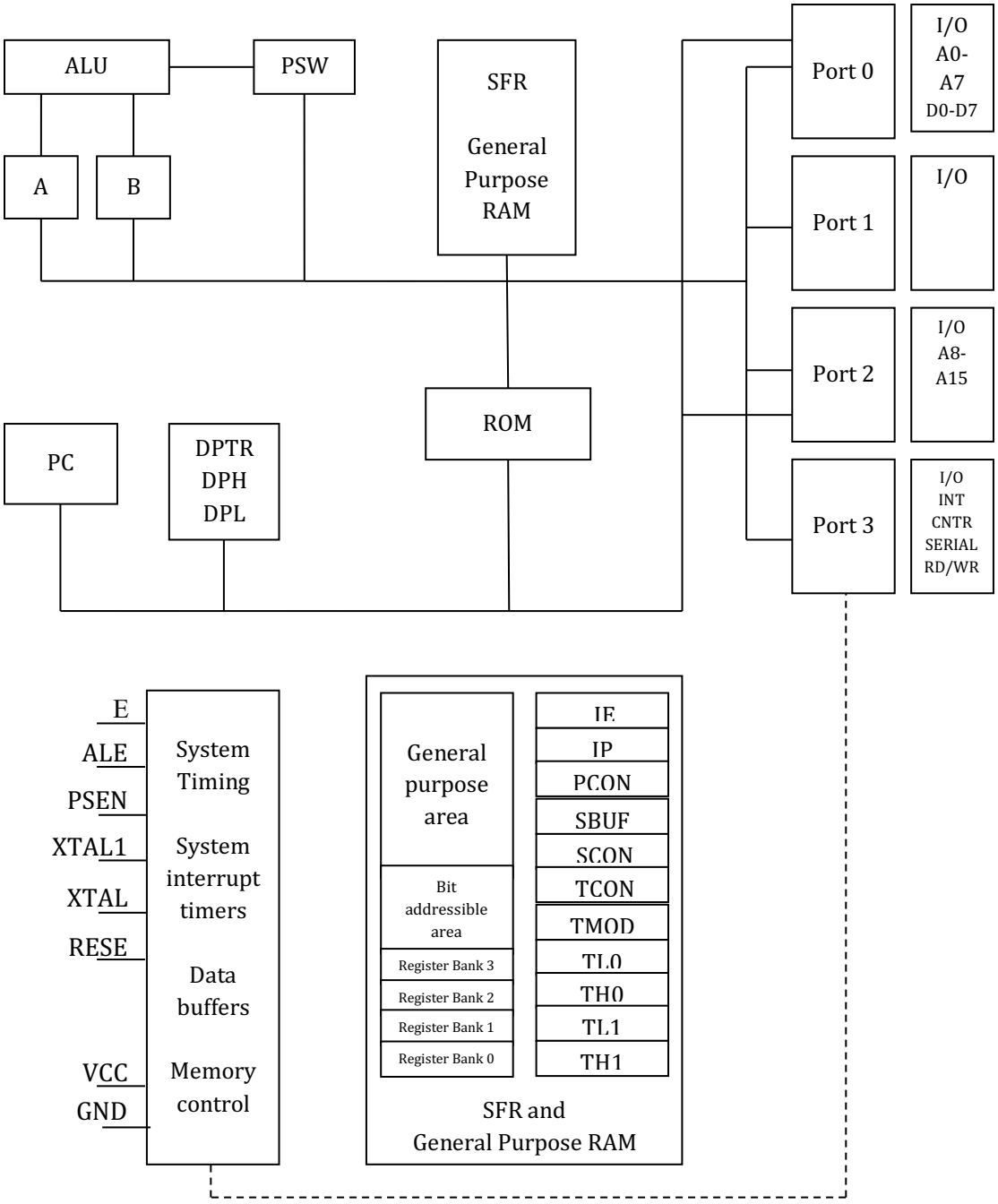


Figure 1.b – 8051 Architecture (Detailed)

- 8051 has **4 K Bytes of internal ROM**. The address space is from 0000 to 0FFFh. If the program size is more than 4 K Bytes 8051 will fetch the code automatically from external memory.
- **Accumulator** is an 8 bit registr widely used for all arithmetic and logical operations.
- **PSW (Program Status Word).** This is an 8 bit register which contains the arithmetic status of ALU and the bank select bits of register banks.
- **Stack Pointer (SP)** – it contains the address of the data item on the top of the stack.

- **Data Pointer (DPTR)** –This is a 16 bit register which is used to furnish address information for internal and external program memory and for external data memory.
- Program Counter (PC) – 16 bit PC contains the address of next instruction to be executed.

1.4 REGISTERS

Registers are used in the microcontroller CPU to store information on temporarily basis. This information could be the data to be processed, or an address pointing to the data which is to be fetched etc. Since 8051 is an 8 bit microcontroller, the data which can be handled by 8051 is of 8 bit and hence the registers are generally 8 bit registers. If there is a need to handling 16 bit data, the data is split into two 8 bit data.

The most widely used registers of the 8051 are

- **Accumulator** is an 8 bit register widely used for all arithmetic and logical operations. Accumulator is also used to transfer data between external memory. B register is used along with Accumulator for multiplication and division. A and B registers together is also called MATH registers.
- **PSW (Program Status Word).** This is an 8 bit register which contains the arithmetic status of ALU and the bank select bits of register banks.

CY	AC	F0	RS1	RS0	OV	-	P

CY - carry flag
AC - auxiliary carry flag
F0 - available to the user for general purpose
RS1,RS0 - register bank select bits
OV - overflow
P - parity

- **Stack Pointer (SP)** – This is an 8bit register which contains the address of the data item on the top of the stack. Stack may reside anywhere on the internal RAM. On reset, SP is initialized to 07 so that the default stack will start from address 08 onwards.
- **Data Pointer (DPTR)** – DPH (Data pointer higher byte), DPL (Data pointer lower byte). This is a 16 bit register which is used to furnish address information for internal and external program memory and for external data memory.
- **Program Counter (PC)** – 16 bit PC contains the address of next instruction to be executed. On reset PC will set to 0000. After fetching every instruction PC will increment by one.
- **General purpose registers - R0-R7**
  The 8051 has 4 registers banks. The B0, B1, B2, and B3 stand for banks and each bank contains eight general purpose registers ranging from 'R0' to 'R7'. A register is a storage element that can be store bits of information.
- **Special Function Register**
  Apart from above mentioned register, 8051 has special function registers which is required for various units of 8051 such as timers, serial communication, interrupt etc. TCON (Timer Control), TMOD (Timer Mode), IP (Interrupt Prioity), IE (Interrupt Enable), SCON (Serial Control), SBUF (Serial Buffer) etc., are few of such special function registers.

1.5 PIN DIAGRAM

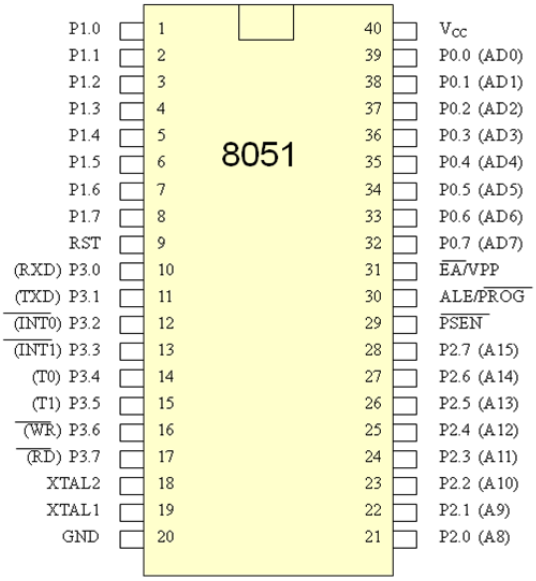


Figure 2 – 8051 Pin Diagram

Pinout Description

Pins 1-8	PORT 1. Each of these pins can be configured as an input or an output.
Pin 9	RESET. 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.External Interrupt 0 input
Pin 13	INT1. External Interrupt 1 input
Pin 14	T0. Counter 0 clock input
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	XTAL2, XTAL1. Internal oscillator input and output. A quartz crystal which specifies operating frequency is usually connected to these pins.

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 latch latches the state of P0 and uses it as a memory chip address. Immediately after that, the ALE pin is returned its previous logic state and P0 is now used as a Data Bus.
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.

1.6 IO PORTS FUNCTIONS

8051 microcontrollers have four Input/Output (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. The ports may be configured as either input port or output port. Whether it is to be configured as an input or an output, depends on its logic state. In order to configure a microcontroller pin as an input, it is necessary to apply a logic zero (0) to appropriate I/O port bit. In this case, voltage level on appropriate pin will be 0. Similarly, in order to configure a microcontroller pin as an input, it is necessary to apply a logic one (1) to appropriate port. In this case, voltage level on appropriate pin will be 5V

PORT 0: The P0 port is characterized by two functions. If external memory is used then the lower address byte (addresses A0-A7) is applied on it. Otherwise, all bits of this port are configured as inputs/outputs. Port 1

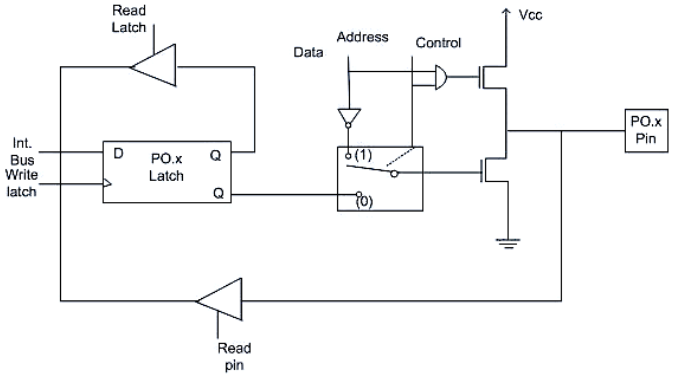
PORT 1: P1 is a true I/O port, because it doesn't have any alternative functions as is the case with P0, but can be cofigured as general I/O only. It has a pull-up resistor built-in and is completely compatible with TTL circuits.

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 of port 3 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 terms of hardware, this port is similar to P0, with the difference that its pins have a pull-up resistor built-in.

PORT STRUCTURE

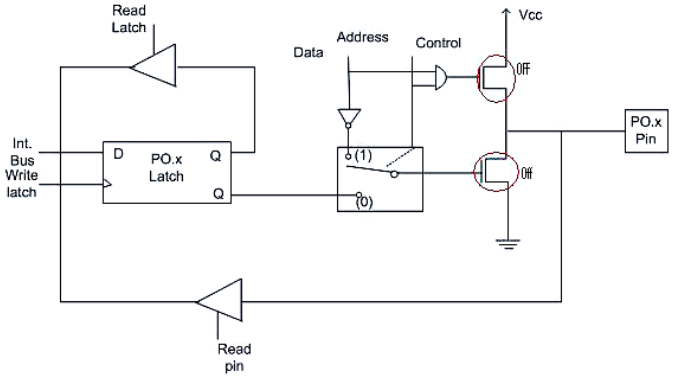
The structure of a Port-0 pin is shown in figure. It has 8 pins (P0.0-P0.7).



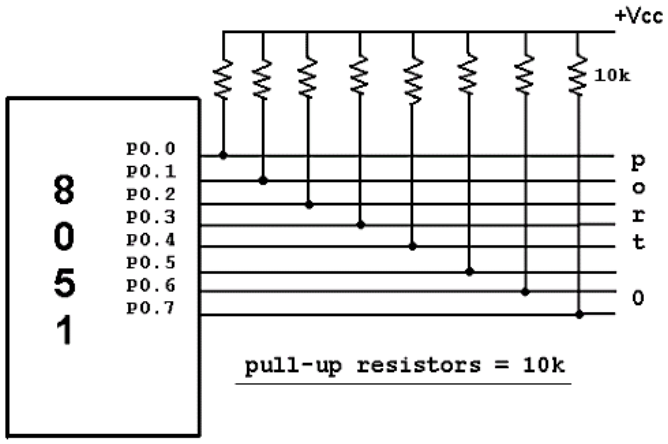
Port-0 can be used as a normal bidirectional I/O port or it can be used for address/data interfacing for accessing external memory. When control is '1', the port is used for address/data interfacing. When the control is '0', the port can be used as a bidirectional I/O port.

PORT 0 as an Input Port.

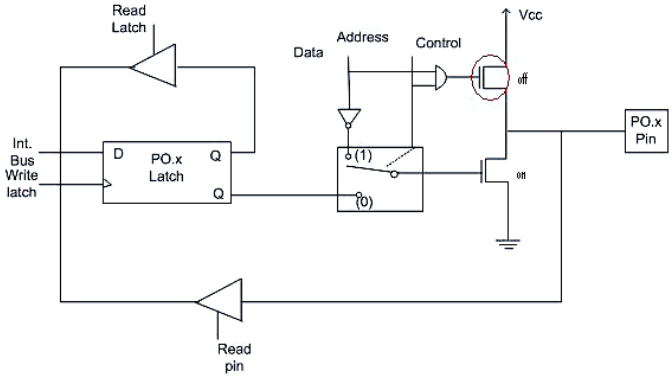
Let us assume that control is '0'. When the port is used as an input port, '1' is written to the latch. In this situation both the output MOSFETs are 'off'. Hence the output pin have floats hence whatever data written on pin is directly read by read pin.



PORT 0 as an Output Port: Suppose we want to write 1 on pin of Port 0, a '1' written to the latch which turns 'off' the lower FET while due to '0' control signal upper FET also turns off as shown in fig. above. Here we wants logic '1' on pin but we getting floating value so to convert that floating value into logic '1' we need to connect the pull up resistor parallel to upper FET . This is the reason why we needed to connect pull up resistor to port 0 when we want to initialize port 0 as an output port .



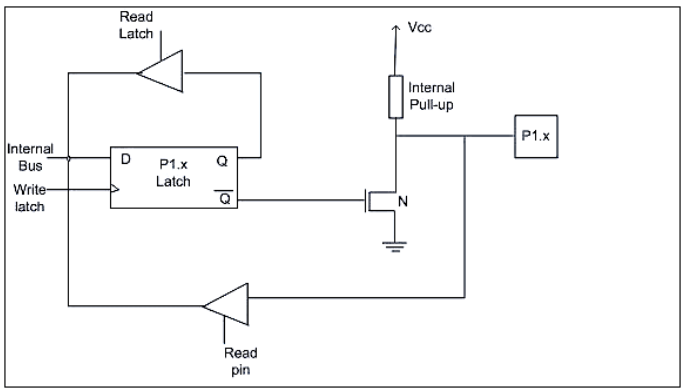
If we want to write '0' on pin of port 0 , when '0' is written to the latch, the pin is pulled down by the lower FET. Hence the output becomes zero.



When the control is '1', address/data bus controls the output driver FETs. If the address/data bus (internal) is '0', the upper FET is 'off' and the lower FET is 'on'. The output becomes '0'. If the address/data bus is '1', the upper FET is 'on' and the lower FET is 'off'. Hence the output is '1'. Hence for normal address/data interfacing (for external memory access) no pull-up resistors are required. Port-0 latch is written to with 1's when used for external memory access.

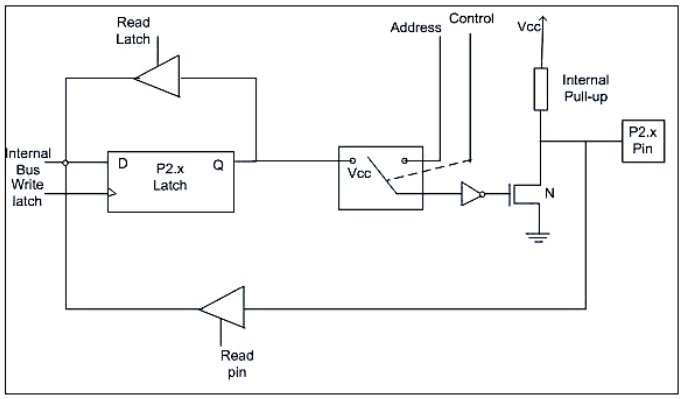
The structure of a port-1 pin is shown in fig below. It has 8 pins (P1.1-P1.7) .

Port-1 dedicated only for I/O interfacing. When used as output port, not needed to connect additional pull-up resistor like port 0. It has provided internally pull-up resistor as shown in fig. below. The pin is pulled up or down through internal pull-up when we want to initialize as an output port. To use port-1 as input port, '1' has to be written to the latch. In this input mode when '1' is written to the pin by the external device then it read fine. But when '0' is written to the pin by the external device then the external source must sink current due to internal pull-up. If the external device is not able to sink the current the pin voltage may rise, leading to a possible wrong reading.

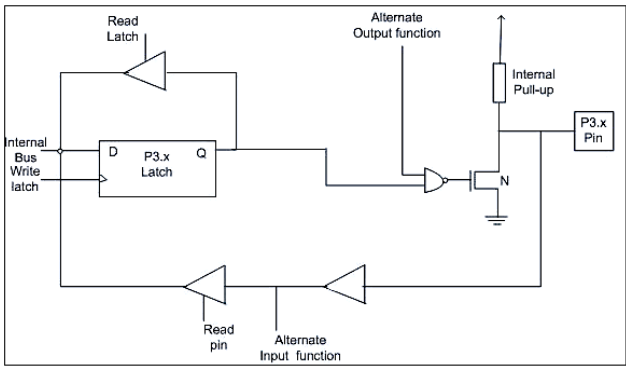


The structure of a port-2 pin is shown in fig. below. It has 8-pins (P2.0-P2.7) .

Port-2 can be used for higher external address byte or a normal input/output port. The I/O operation is similar to Port-1. Port-2 latch remains stable when Port-2 pin are used for external memory access. Here again due to internal pull-up there is limited current driving capability.



The internal structure of a port-3 pin is shown in fig below. It has 8 pins (P3.0-P3.7)

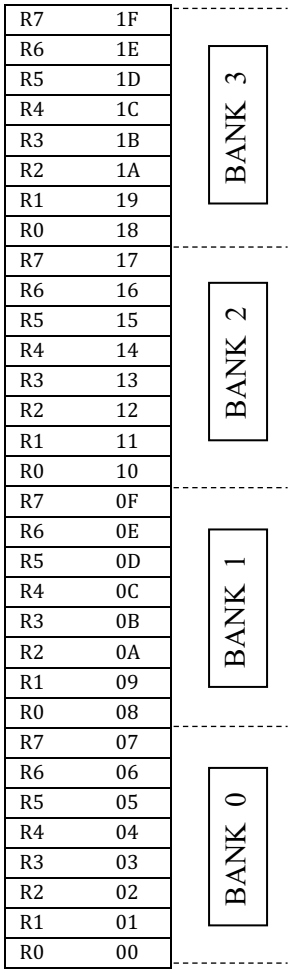


Port-3 is having alternate functions to each pin,

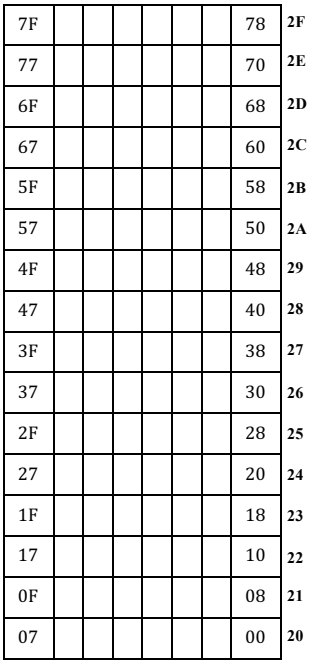
P 3.0	**RxD**
P 3.1	**TxD**
P 3.2	**$\overline{\text{INT0}}$**
P 3.3	**$\overline{\text{INT1}}$**
P 3.4	**T0**
P 3.5	**T1**
P 3.6	**$\overline{\text{WR}}$**
P 3.7	**$\overline{\text{RD}}$**

1.7 INTERNAL MEMORY ORGANIZATION

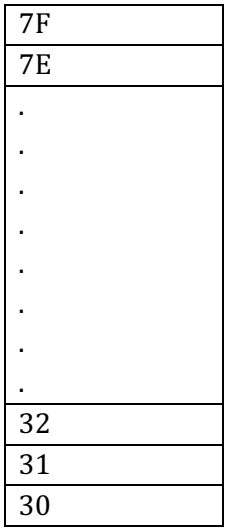
Internal RAM organization



Working Registers



Bit addressable memory



General purpose memory

Register Banks: 00h to 1Fh. The 8051 uses 8 general-purpose registers R0 through R7 (R0, R1, R2, R3, R4, R5, R6, and R7). There are four such register banks. Selection of register bank can be done through RS1,RS0 bits of PSW. On reset, the default Register Bank 0 will be selected.

Bit Addressable RAM: 20h to 2Fh . The 8051 supports a special feature which allows access to bit variables. This is where individual memory bits in Internal RAM can be set or cleared. In all there are 128 bits numbered 00h to 7Fh. Being bit variables any one variable can have a value 0 or 1. A bit variable can be set with a command such as SETB and cleared with a command such as CLR.
Example instructions are:
SETB 25h ; sets the bit 25h (becomes 1)
CLR 25h ; clears bit 25h (becomes 0)
Note, bit 25h is actually bit 5 of Internal RAM location 24h.
The Bit Addressable area of the RAM is just 16 bytes of Internal RAM located between 20h and 2Fh.

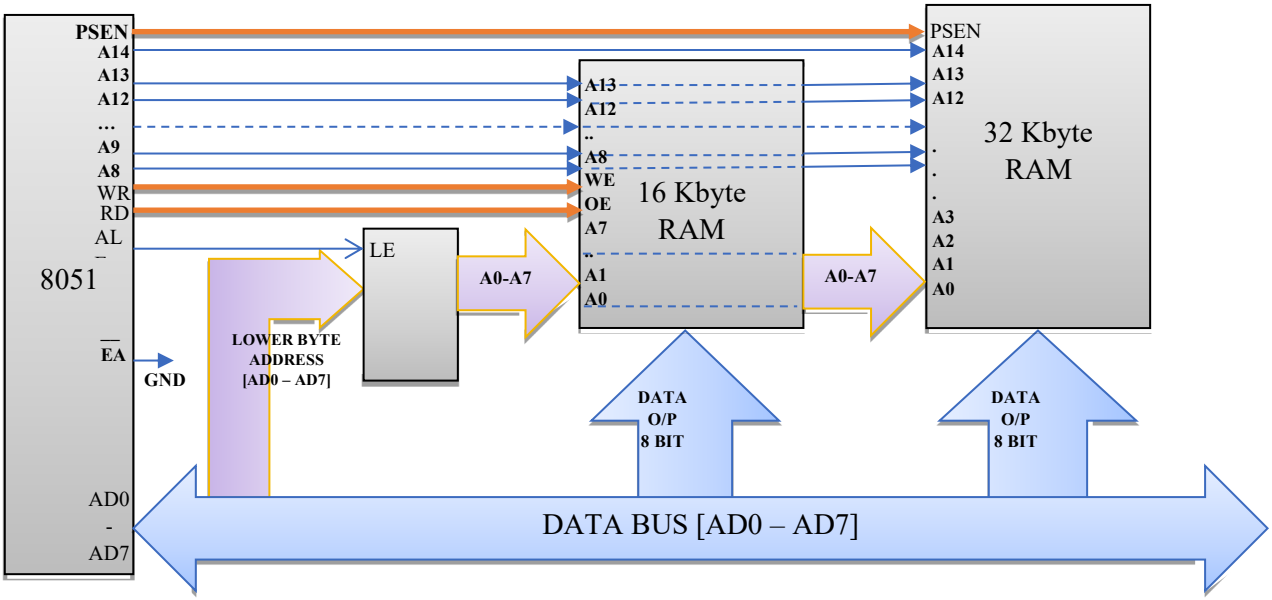
General Purpose RAM: 30h to 7Fh. Even if 80 bytes of Internal RAM memory are available for general-purpose data storage, user should take care while using the memory location from 00 -2Fh since these locations are also the default register space, stack space, and bit addressable space. It is a good practice to use general purpose memory from 30 – 7Fh. The general purpose RAM can be accessed using direct or indirect addressing modes.

1.8 EXTERNAL MEMORY INTERFACING

Eg. Interfacing of 16 K Byte of RAM and 32 K Byte of EPROM to 8051

Number of address lines required for 16 Kbyte memory is 14 lines and that of 32Kbytes of memory is 15 lines.

The connections of external memory is shown below.



The lower order address and data bus are multiplexed. De-multiplexing is done by the latch. Initially the address will appear in the bus and this latched at the output of latch using ALE signal. The output of the latch is directly connected to the lower byte address lines of the memory. Later data will be available in this bus. Still the latch output is address it self. The higher byte of address bus is directly connected to the memory. The number of lines connected depends on the memory size.

The RD and WR (both active low) signals are connected to RAM for reading and writing the data.

PSEN of microcontroller is connected to the output enable of the ROM to read the data from the memory.

EA (active low) pin is always grounded if we use only external memory. Otherwise, once the program size exceeds internal memory the microcontroller will automatically switch to external memory.