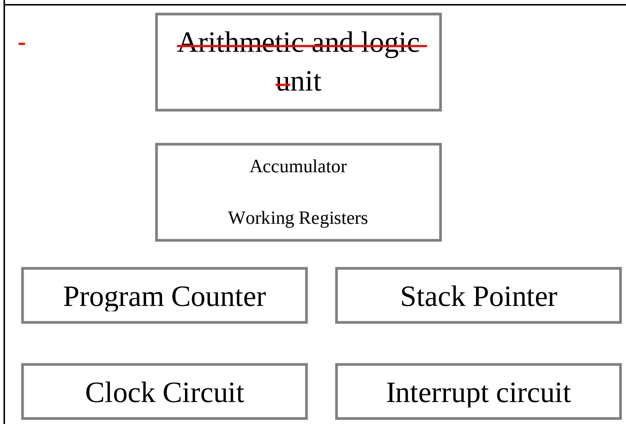
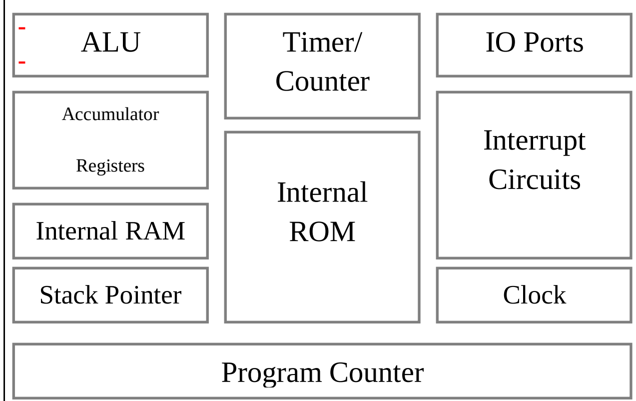
[redacted]

1.1 MICROPROCESSORS AND ~~MICROCONTROLLERS~~

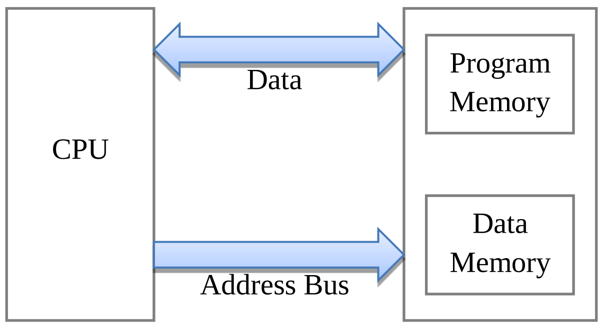
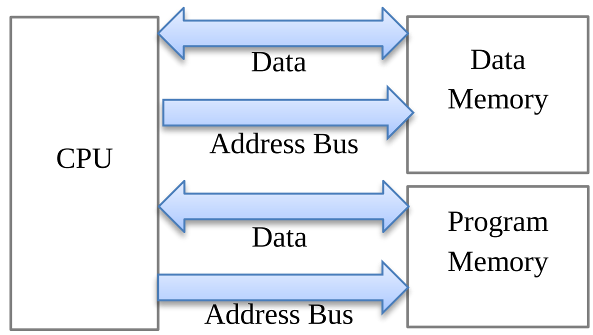
~~*Microprocessor*~~	~~*Microcontroller*~~
	
~~*Block diagram of microprocessor*~~	~~*Block diagram of microcontroller*~~
~~Microprocessor contains ALU, General purpose registers, stack pointer, program counter, clock timing circuit, interrupt circuit~~	~~Microcontroller contains the circuitry of microprocessor, 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. 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

1.2. 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 COMPUTER SOFTWARE

A set of instructions written in a specific sequence for the computer to solve a specific task is called a program and software is a collection of such programs.

The program stored in the computer memory in the form of binary numbers is called machine instructions. The *machine language* program is called *object code*.

An *assembly language* is a mnemonic representation of machine language. Machine language and assembly language are low level languages and are processor specific.

The assembly language program the programmer enters is called *source code*. The source code (assembly language) is translated to object code (machine language) using *assembler.*

Programs can be written in *high level languages* such as C, C++ etc. High level language will be converted to machine language using *compiler or interpreter*. Compiler reads the entire program and translate into the object code and then it is executed by the processor. Interpreter takes one statement of the high level language as input and translate it into object code and then executes.

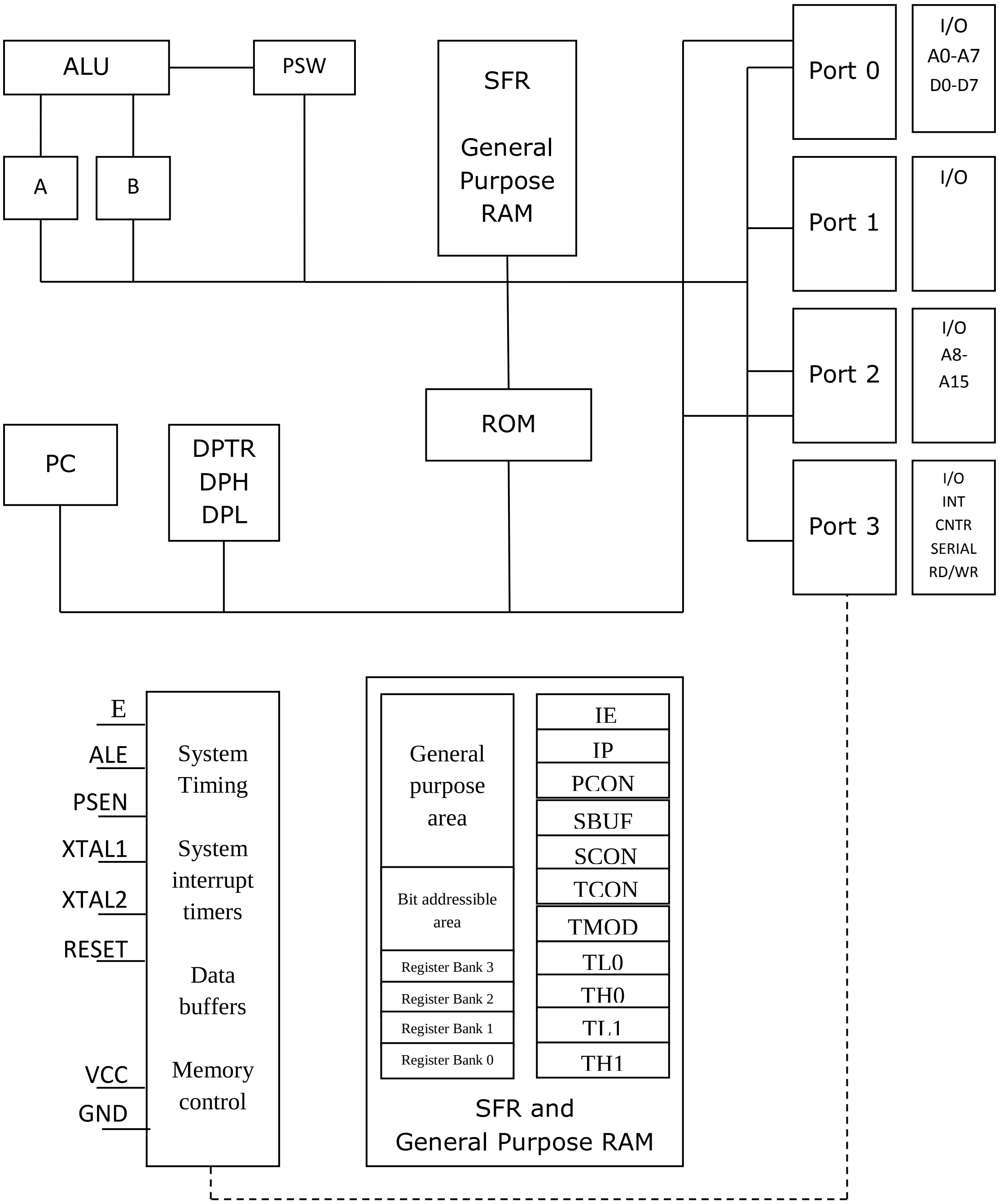
1.4 THE 8051 ARCHITECTURE

Introduction

Salient features of 8051 microcontroller are given below.

- Eight bit CPU
- On chip clock oscillator
- 4Kbytes 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.
- Sixteen bit Program counter (PC) and a data pointer (DPTR)
- 8 Bit Program Status Word (PSW)
- 8 Bit Stack Pointer
- Five vector interrupt structure (RESET 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.



- 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 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.

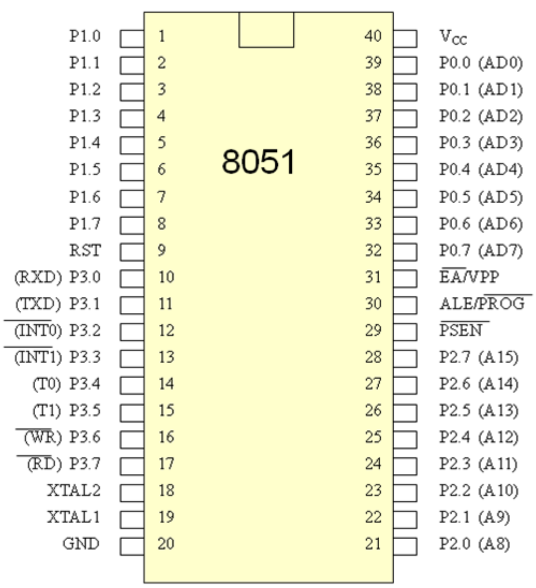
- PSW (Program Status Word). This is an 8 bit register which contains the arithmetic status of ALU and the bank select bits of register banks.

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) – it 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.

1.5 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 MEMORY ORGANIZATION

Internal RAM organization

R7	1F	BANK 3
R6	1E	
R5	1D	
R4	1C	
R3	1B	
R2	1A	
R1	19	
R0	18	
R7	17	BANK 2
R6	16	
R5	15	
R4	14	
R3	13	
R2	12	
R1	11	
R0	10	
R7	0F	BANK 1
R6	0E	
R5	0D	
R4	0C	
R3	0B	
R2	0A	
R1	09	
R0	08	
R7	07	BANK 0
R6	06	
R5	05	
R4	04	
R3	03	
R2	02	
R1	01	
R0	00	

Working Registers

2F	7F							78
2E	77							70
2D	6F							68
2C	67							60
2B	5F							58
2A	57							50
29	4F							48
28	47							40
27	3F							38
26	37							30
25	2F							28
24	27							20
23	1F							18
22	17							10
21	0F							08
20	07							00

Bit addressable memory

7F
7E
. . . .
32
31
30

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.

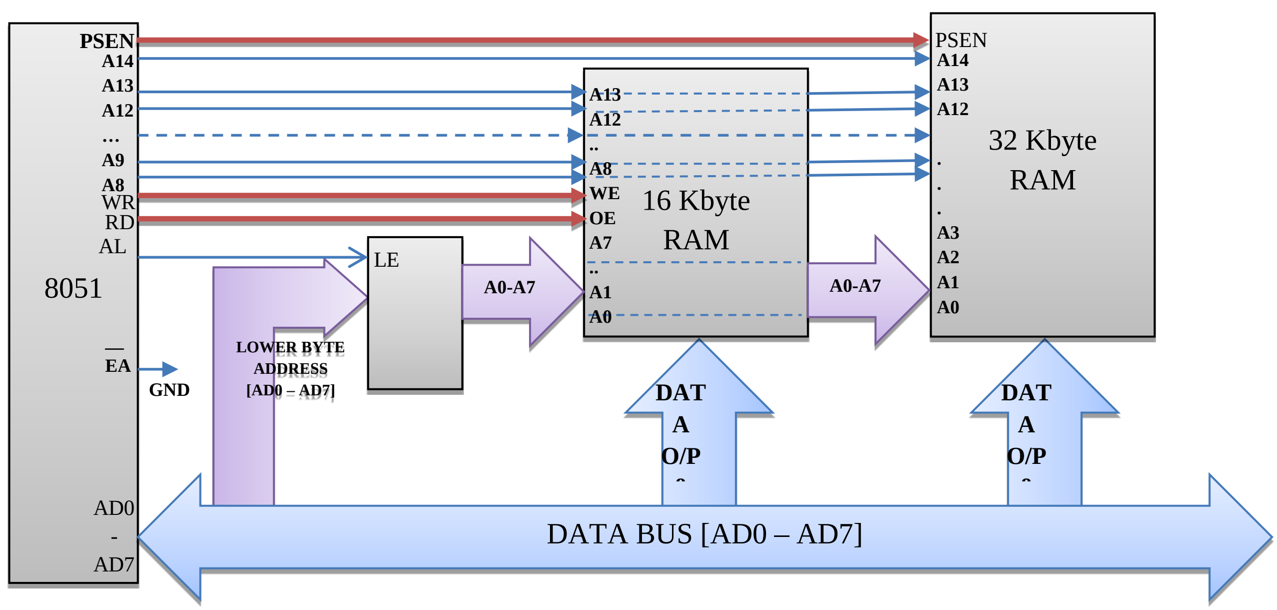
General Purpose RAM: 30h to 7Fh. Even if 80 bytes of Internal RAM memory are available for general-purpose data storage, user should take care while using the memory location from 00 -2Fh since these locations are also the default register space, stack space, and bit addressable space. It is a good practice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

1.7 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.