A microcontroller is a microprocessor with inbuilt peripherals. A microcontroller can also be compared with a Swiss knife having multiple functionalities. The differences between a microprocessor and microcontroller are as follows:

<table>
<thead>
<tr>
<th>Micro-Processor</th>
<th>Micro-Controller</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intensive computation</td>
<td>Computation for particular functions</td>
</tr>
<tr>
<td>No inbuilt peripherals</td>
<td>CPU with inbuilt peripherals on same chip</td>
</tr>
<tr>
<td>Higher Clock speed</td>
<td>Slower clock speed</td>
</tr>
</tbody>
</table>

**Different types of Micro-controller chips**

1) Embedded (self contained) 8-bit micro controllers
2) 16 and 32 bit micro controllers
3) Digital Signal Processors
**Instruction set of an 8-bit micro controller:**

1) Data Transfer (MOV, LDA, etc)
2) Data Processing
3) Execution / Arithmetic and Logical
4) Processor Control

<table>
<thead>
<tr>
<th>CISC</th>
<th>RISC</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Complex Instruction Set Computer)</td>
<td>(Reduced Instruction set Computer)</td>
</tr>
<tr>
<td>More no of instructions</td>
<td>Less no of instructions</td>
</tr>
<tr>
<td>Programming is easy</td>
<td>Programming is difficult</td>
</tr>
<tr>
<td>Ex: 8085, 8086, Pentium etc</td>
<td>Ex: PIC, Power PC's</td>
</tr>
<tr>
<td>Debugging is easy</td>
<td>Debugging is difficult</td>
</tr>
</tbody>
</table>

**ARCHITECTURE**

**There are two basic architectures:**

1) Harvard Architecture
2) Princeton Architecture (Von Neumann Architecture)
**Princeton Architecture**

<table>
<thead>
<tr>
<th>Only one Memory interface is there.</th>
</tr>
</thead>
</table>

**Harvard Architecture**

<table>
<thead>
<tr>
<th>Two memory banks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prg store and data store</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Avg computation speed is low (no parallelism)</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Less no of control signals</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Avg computation speed is high</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>More no of control signal</th>
</tr>
</thead>
</table>

---

**Princeton (Von Neumann)**

Instruction:

Read a data byte from memory and store in the accumulator.

(1) If this instruction is executed in Princeton architecture then it is executed as follows.
The following cycles are required to execute the above instruction

Cycle 1: read instruction (Instruction Fetch cycle)
Cycle 2: Read Data out from memory and store in accumulator

Using Harvard Architecture

The execution above instruction will be done in the following cycles

**Cycle 1**: Complete the previous instructions and Read the present instruction.
(Note: this will not be done only for jump instruction as previous instruction. Because if jump is there, it flushes off the instructions int the stack).

**Cycle 2**: Execute, Read Data memory and store in Accumulator. Read next instruction.

Advantage of this architecture is that each instruction takes one instruction cycle.

*Example*: in PIC Microcontroller, of 4 MHz, it requires 4 clock cycles to execute one instruction cycle.
Hence, 1 million instructions could be executed in one second.
**Instruction Decoders:** There are two types of decoder architecture available for instruction decoding.

**Micro-coded Processor**

![Diagram of Micro-coded Processor]
Hard-coded /Hardware Processor

<table>
<thead>
<tr>
<th>Micro-Coded Processor</th>
<th>Hard-Coded /Hardware Process</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flexible, easy to revise. Because changing the code in the memory will change the decode.</td>
<td>Rigid</td>
</tr>
<tr>
<td>Easy to debug</td>
<td>Hard to debug</td>
</tr>
<tr>
<td>Generation of control signals is generated from the data stored in the micro-coded memory. So it's easy to change the decoding process by changing the data in the memory.</td>
<td>Sometimes it reduces the execution time. Since the control signals are generated by the combinational logic circuit, so to change decoding total circuit on chip is to be changed.</td>
</tr>
</tbody>
</table>
MEMORY: Volatile and Non-Volatile memory

Non-volatile memories-PROM,EPROM,EEPROM,Flash Memory
MASK ROM-(Programmed by the manufacturer)

Volatile - RAM,DRAM & SDRAM.

Non-volatile MEMORIES:

Mask ROM is programmed by manufacturer at the factory.
PROM is one time programmable.
EPROM- Program electrically and UV erasable.
EEPROM-Program & erased electrically. Here erasing operation requires so many clock cycles Erasing is done bit by bit.
Flash memory- Programming and erasing done electrically but the erasing operation is done in one clock pulse. It has different bus structure.

ROM: Basically this uses a MOS having different structure. PROM has more life when compare to Mask ROM.
Charge stored in between control and float gates.
RAM
Static RAM

The output is 0 until VDD is disconnected. It is self generated.

ROM
Initially all are in unprogrammed state.

Unprogrammed state means 1
Programmed state means 0

Program Counter Stack: This was LIFO (Last In First Out) technique
This is used for sub-routine calls.
PUSH- for storing data in stack.
POP- for reading data in stack.
Stack Memory is a part of RAM.
When a JUMP instruction is called then the data stored in the Program Counter (usually the address of next instruction) is (PUSHED) stored in the stack. When the Return instruction in subroutine is called then it POPs the data from the stack.

I/O Register Space:

How the register space is built in different Arch?

Princeton(Von-Neumann): In the arch, it has single memory bank. Usually these I/O registers are used after the ROM space.

(i) Memory mapped I/O register
(ii) Separate I/O register

Drawback in this is sometimes the address of I/O are used by the data memory courses flow in execution

Less Complex More complex

Using Harvard Architecture: There are so many ways to for I/O Register space.
(1) & (ii) are memory mapped I/O register space.
(iii) & (iv) are separate I/O register space.
(iii) & (iv) are complex
In (i) parallel execution of instruction and reading data is not done at the same time.
so (ii) is used for I/O register space in Harvard Arch.
Parallel Processing is not possible when the 1st method is selected.

Micro controller Clock:
One instruction cycle consists of a few clock cycles
To execute one instruction, we require a few instruction cycles.
In 8051, 12 clock cycles are require to execute on instruction cycle.

I/O Pins: Address bus, data bus, control bus, I/O, Serial Tx and RX.
In this data is written through the latch and the output is available all the time. Strobbing the Write Pin latches the data. Read Pin is enabled to read the data.

In the architecture shown below, two latches are used, one is for writing data to the latch and other is for outputting data through a buffer to the pin.

Data bus

The above architecture will be there for each line of the data bus. While accessing only a bit, then other lines are tri stated.
Interrupts:

Interrupt will be checked at the end of every instruction execution. Whenever an interrupt occurs, then the program execution of main program is stopped and calls the ISR (Interrupt subroutine). Before calling the ISR, it pushes the registers into the stack and calls the ISR routine. The subroutine of ISR is as follows:

Depending upon the priority of the interrupt the main program calls that particular subroutine. If the interrupt has less priority, the CPU completes the main program and attempts the interrupt.

**Vectored Interrupt:**

The interrupt addresses are fixed (Program Memory)

<table>
<thead>
<tr>
<th>Interrupt</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>000134</td>
</tr>
<tr>
<td>1</td>
<td>000BH</td>
</tr>
<tr>
<td>0</td>
<td>0003H</td>
</tr>
<tr>
<td>Reset</td>
<td>0000H</td>
</tr>
</tbody>
</table>
**Timer:**

- To determine the duration of an interval.
- To apply a delay

This is also used to measure the duration of a pulse, that it has high level. For example

\[ T_{ON} \text{Final count} - \text{Initial count} / f_{clock} \]
In this the counter starts from the FFFFH and it decreases from it. When the duration to be calculated then the counter decrements from FFFFH. If the duration is high then the counter goes to 0000H and starts from FFFFH hence the overflow goes high.
Assignment No.1
1. What are the differences between a microcontroller and a microprocessor
2. Why was the Princeton(Von Neumann) architecture accepted initially? What was the reason of popularity of Harvard architecture at a later stage?
3. Explain parallelism with Harvard architecture by an example.
4. What is a Mask ROM? Mention a few advantages of Mask ROM.
5. Explain the working of a ROM cell.
6. Explain the read write operation of a static RAM memory cell using CMOS transistors.
7. Why is it inconvenient to put I/O registers in program memory space in Harvard architecture? What is the convention followed in placing I/O registers in a typical microcontroller employing Harvard architecture?
8. Explain how simple I/O pins are realized with an 8051 microcontroller?
9. An I/O pin is to be designed for serial I/O operation. Give a block schematic showing the connections. Explain the philosophy.