# Micro Controller Basics

Prasad Mehendale

## 1 Why Micro controller?

Figure 1: Micro controller

**Why should we use and learn micro controllers?** people ask.

Human brain desires comfort. It tries to avoid doing the same things again and again. To avoid this boredom and to live comfortably, many (though not all) gadgets are designed. At the advent of electronics, precise control of such gadgets is possible.

Let us take an example. Washing clothes was (or is) a drudgery. So washing machines were born. Making them automatic has made us more comfortable. Precise control over them and making them automatic is now possible with micro controllers. This not only allows us to be lazy but saves resources like water, soap and electricity.

Micro controller executes instruction every micro second. So the whole function of a gadget is broken down in small parts. Each small part is done by a single instruction. This is how micro controller works.

To measure physical quantities, controllers are used. Now a days, ADCs (Analog to Digital Converters) are built inside the controller chips by default. To measure temperature for example, connect a temperature sensor to the ADC channel. ADC will convert the analog voltage to its digital (binary) equivalent. This binary number is further processed and displayed by micro controller.

Micro controllers are becoming cheap. I remember buying a chip (Intel 8051 with UV erasable eeprom) for Rs. 800 in 1992. Today we have somewhat better feature packed chips of the same architecture. They are available off the shelf for less than 40 Rs. This tells us that the cost of silicon brain is going down. Why not take advantage of it and also pass on this benefit to the final user ?

In this article, you will get a general idea of what micro controller is and how it works. The book will introduce to you some key concepts from the field of embedded systems. When you are taking any course at Techno e-School, this book will help you more because the programmes given here are directly relevant to the Techno e-School hardware kits.

Happy micro controlling !!

–**Prasad Mehendale**

# 2 Introduction to micro controller

Figure 2: Micro controller chip

## 2.1 Math behind $\mu$ controllers

A micro controller understands and processes only numbers. To be precise, it uses only the binary number system for its internal operation. In this section, we will know more about common number systems.

## 2.2 Decimal number system

We use this number system in our day-to-day life. Total ten symbols (0 to 9) are used to represent any number. Larger is the number value, more is the number width.

## 2.3 Binary number system

This system uses only two symbols to count numbers. 0 and 1 are those symbols. As we count further, width of the number increases. Each symbol is called a bit. More are the number of bits, larger number is represented. Decimal 99 has two symbols. Binary equivalent of decimal 99 is 110 0011. Notice that the same value uses more symbols in binary system.

## 2.4 Hexadecimal number system

**This system is useful in reading and manipulating binary numbers in groups of four**. To convert binary number to its hex equivalent, following procedure with an example is useful:

1. Consider an 8 bit binary number: 00010011
2. Divide the number in two groups of 4 bits (i.e. nibbles)- 0001 0011
3. Left nibble is called as the Most Significant Nibble (MSN)
4. Right nibble is called as the Least Significant Nibble (LSN)
5. Convert MSN to HEX number: 0001 $\rightarrow$ 8x0 + 4x0 + 2x0 + 1x1 =1
6. Convert LSN to HEX number: 0011 $\rightarrow$ 8x0 + 4x0 + 2x1 + 1x1 =3
7. Put the two HEX numbers again side by side. i.e. 13H (H stands for Hexadecimal)
8. So 00010011$\rightarrow$ 13H $\rightarrow$ 16x1 + 1x3 = 19 Decimal

Table 1: Comparing number systems

| **binary** | **decimal** | **hexadecimal** |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 1 | 1 |
| 10 | 2 | 2 |
| 11 | 3 | 3 |
| 100 | 4 | 4 |
| 101 | 5 | 5 |
| 110 | 6 | 6 |
| 111 | 7 | 7 |
| 1000 | 8 | 8 |
| 1001 | 9 | 9 |
| 1010 | 10 | A |
| 1011 | 11 | B |
| 1100 | 12 | C |
| 1101 | 13 | D |
| 1110 | 14 | E |
| 1111 | 15 | F |

### 2.5 Representing Hex numbers

**HEX numbers** use symbols A to F to represent numbers after 9. They are also a part of alpha-numeric character set.

**To confirm** that A to F symbols represent a number and not part of some text, a 0 is presides the number.

**For example** , 1100 0110 → 0C6H → 16x12 + 1x6 = 198 Decimal

*You can use scientific calculator to convert numbers from one system to the other.* The table 1 on page 4 compares the three number systems discussed earlier.

## 3 The sense of time

A programmer breaks down a big task in small chunks. Every little chunk is further divided in still tiny chunks. Every small part of the task are executed in **micro seconds** (or even in nano seconds) by the controller chip. This is done in a specific order.

Now the question is, if controller does the tasks in micro *seconds*, how does a micro controller understand time ? This question is very important. Here is the answer-

We humans, have the sense of time mainly due to sunrise and sunset. Accordingly we have adjusted our lives. The terms memory, now and then have meaning if we have the sense of time. Same is true for micro controllers! But how does a micro controller understand time?

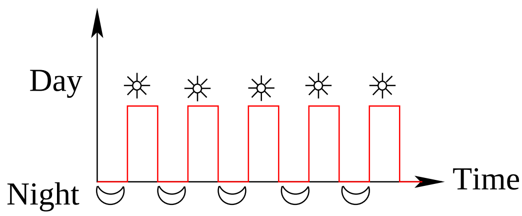


Figure 3: The Sun clock

Well, let us know more about the sunrise and the sunset. Our perception of sunrise and sunset is not simply about light and dark. *Sunrise and sunset is a repetitive process*. The day and the night together form a cycle of an oscillator. These cycles give us the sense of time. When you say **I read a book yesterday, it means, you read it one sun-cycle before the current sun-cycle.** Figure 3 makes this situation clear.

Now apply this principle to micro controllers. If micro controller is connected to an electrical oscillator of fixed frequency, the smallest action of a micro controller can be related to a cycle of the oscillator.

Every micro controller has an oscillator connected to it. (See fig. 4) Most of the times, it is a crystal oscillator. A peizo crystal oscillates with very high frequency (in KHz and MHz) and provides the micro controller the sense of time. Crystal oscillators produce very stable frequency. They are stable especially for a wide range of temperature.

High frequency oscillators have period in micro or nano seconds. Usually, micro controllers execute instructions in one or more oscillator cycles. Thus micro controllers work in synchronism with the oscillator pulses and have the

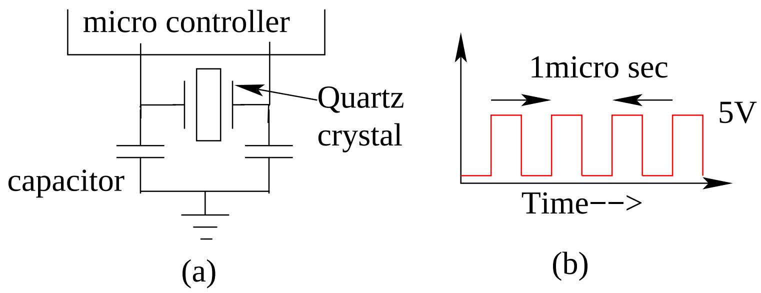


Figure 4: Oscillator clock

sense of time. The oscillator is like a clock to the micro controller which goes ahead micro second by micro second performing a small task per clock-tick.

# 4 The memory

Any intelligent digital system needs memory. However, only memory doesn't mean intelligence. Let us discuss how memory is implemented using electronics.

Everyone has seen an electric keyboard on a house wall. You will see that the keyboard has wonderful features.

1. We can put any of the keys to either ON or OFF state.
2. There is no third state possible.
3. Now once a key is made ON or OFF, the state doesnt change on its own.
4. This key continues to be ON till it is made OFF and vice versa.
5. *So this array of keys stores (memorises) a binary number at any given instance.*
6. In figure 5, if 1 means ON and 0 means OFF, the key array on the wall will represent the binary number say : 1 0 1 1
7. The left most key is most significant and the right most is least significant. (This is very similar to our normal decimal number system.)

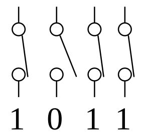


Figure 5: Simple mechanical memory

## 4.1 Flip-flop

In electronics, HI state (means 1) is represented by +5V and LO state (means 0) is represented by 0V.

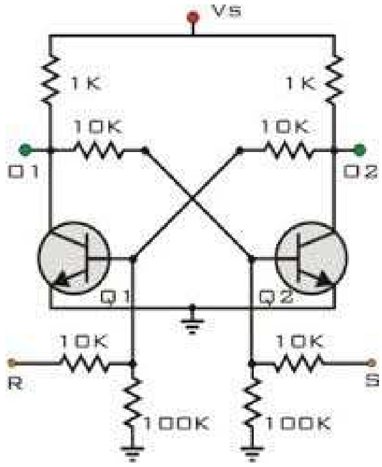


Figure 6: The flip flop

A transistorised circuit (see figure 6 on page 7) has the ability of changing and maintaining the output state. The circuit in figure 6 has following features:

1. Give a positive going pulse to terminal "S" (set) in figure 6. This makes output "O1" HI. It will remain HI (5V).
2. Give a positive going pulse to terminal "R" (reset) in figure 6. This makes output "O1" LO. It will remain LO (0V).
3. To change the state of output "O1", we have to trigger "R" or "S" terminal. In the meanwhile, the earlier state is "remembered".
4. At any instance, output "O2" has complementary state as that of "O1".
5. The circuit is stable in any of the two states at a given instance so it is also called as bi-stable multi-vibrator. It is also called as the **"flip-flop".**

6. Notice that this circuit works like a mechanical switch on the wall. We can also say that the normal ON-OFF switch is a mechanical flip-flop.

7. Also notice that the flip-flop can store either 0 (0-Volts) or 1 (5-Volts) at a time. It can't take any other third state.

## 4.2 Electronic registers

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| MSB | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | LSB |

Figure 7: An 8 bit register

Electronic registers are made of flip-flops. In figure 7, Each square represents a flip-flop. In this section, we will understand how (binary) numbers are stored in electronic registers:

- A digital intelligent system understands only binary numbers.
- Binary numbers are stored in registers which are like keyboard arrays.
- Registers are made of flip-flops. Flip flops are bistable multi-vibrators.
- They can be stable in either of the state HI or LO.
- This simply means that a binary number can be stored in an array of flip-flops easily.
- We can change its value using control terminals and leave the array as it is till we want to change it.
- Each flip flop can store (i.e. memorise) only one of the two possible values 0 or 1.
- This smallest unit of memory (flip flop) is called a bit.
- Usually flip-flops are arranged in groups of 4,8,16,32,64.
    1. Each flip flop represents a bit.
    2. The 4 bit array is known as a nibble.
    3. The 8 bit array is known as a byte.
    4. The 16 bit array is called as a word.
    5. The 32 bit array is called as the double word.

6. The 64 bit array is called as the quad word.
7. **A byte (8 bit array) can store any number between 0 to 255.**
8. **A word (16 bit array) can store any number between 0 to 65535.**

In micro controllers normally we will handle only a bit, a byte and at the most a word. Inside the micro controller there are thousands of 8 bit registers available in different forms. This makes the controller 8-bit controller. Which means, an 8 bit micro controller can handle an 8 bit number (0 to 255) at a time for any (arithmetic or logical) operation.

Now, everything is a number inside the micro controller.

- The flip flop array has an address (a number)
- Data that is stored in an array is a number
- An instruction used by the programmer is expressed in terms of numbers.
- So to deal with anything, it needs just to be a number.
- If the controller can process numbers, it can do anything.

The clock i.e. the crystal oscillator provides triggering pulses to the micro controller. Suppose at the $145^{th}$ pulse, control terminals of a flip-flop array (say a byte) set the array value to be 235 and at $200^{th}$ pulse the controller changes the number to 100. Then in the mean time, the number 235 is said to be memorised.

Anything is expressed as a number for a micro controller! So the system of storing numbers with reference to time, is called as the memory which can remember anything.