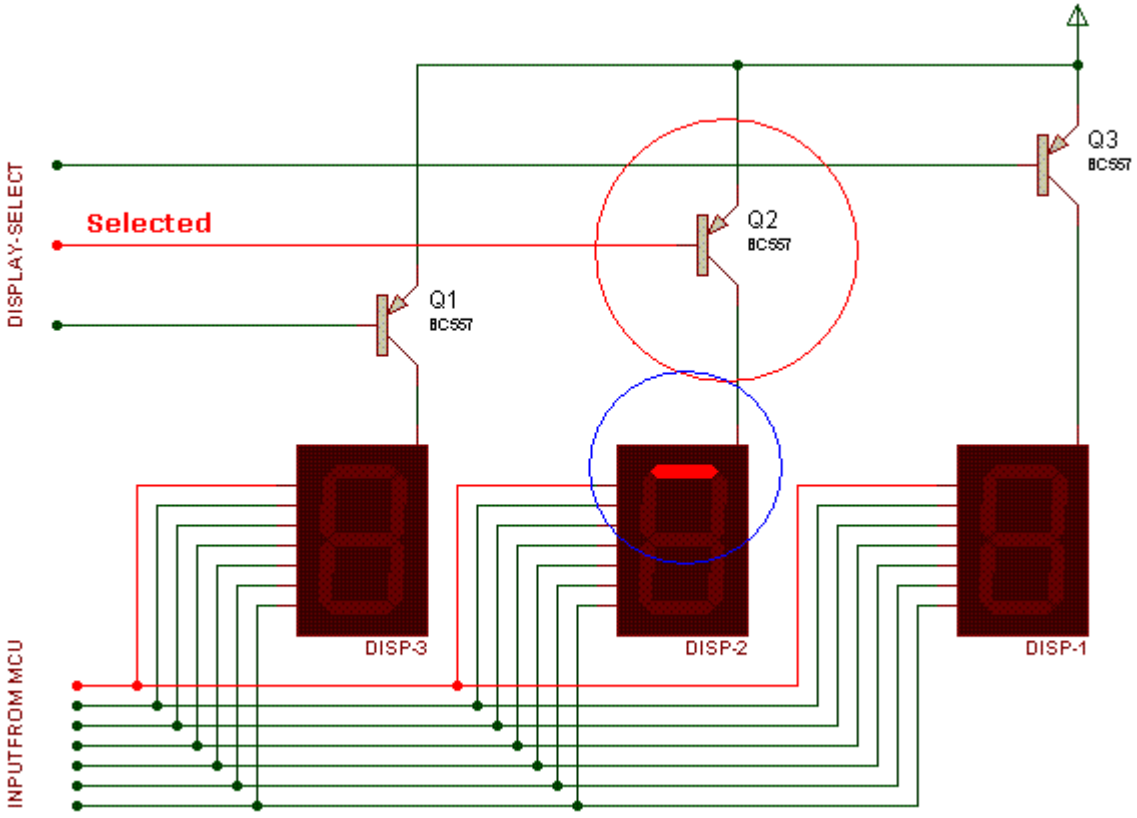
Multiplexed Seven Segment Displays.

AVR Tutorial Series

We have discussed the basics of seven segment displays on our tutorial "Using Seven Segment Displays with AVR MCUs". So you should be familiar with them. In this tutorial we will discuss about multiplexing of seven segment displays. Multiplexing is required when we want to interface 3 or 4 or even more such displays with MCUs since it we go for normal way it will require lots of IO port. So the smart way is multiplexing. Multiplexing achieved by tricking our eyes. Only one display is active at a time but we see all of them active.

For multiplexing all the displays are connected in parallel such that if you activate any segment, say 'a' the 'a' segment of all displays glows up. But the trick is that we can switch on and off the "common" line of the displays under MCU control. So if we wish to light up the 'a' segment of display 2 we simply switch on display 2 first by applying proper level at the base of its driving transistor as shown in figure.



The DISPLAY 2 selected.

If we like to display the digit say "123" on three displays first we select disp-3 by applying a "low" level at the base of transistor Q1 and output the code of required digit at the data input terminals. Since display 3 is selected "1" is displayed in disp-3 then we wait for some time and select disp-2 and output code of digit "2". This "2" will be shown in disp-2. Then we select display-1 and output code for digit "3". And now you can guess it will be displayed in display-1. This process is shown in animation below.

If we repeat this step fast enough (not a problem with MCUs !) what we get due to persistence of vision is that we see the number "123" displayed "still" in the display as if all the displays are active simultaneously.

Programming

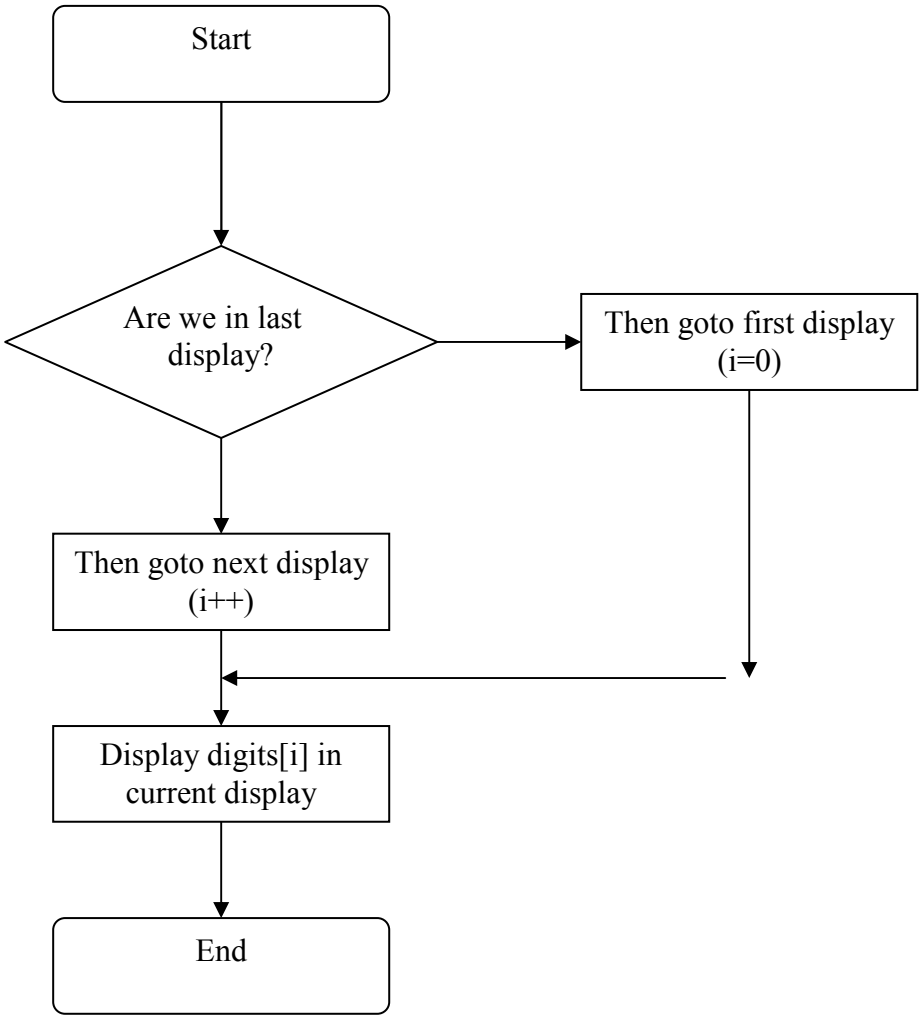
The actual programming will be covered in next tutorial. In this part we will simply see the steps involved.
We will use the TIMER0 to refresh our displays. To learn about timer please see the previous tutorial. The timer will interrupt the CPU at predefined time interval an the CPU will switch to next display and display a digit there.
An array variable

```
uint8_t digits[3];
```

will hold the digits for the three displays what ever we need to display on the displays we store them there. So to display "911" on the display we store.

```
digits[0]=1;
digits[1]=1;
digits[2]=9;
```

The ISR of TIMER0 Overflow is as follows



We set the TIMER0 prescaler as 256 so the sytem clock i.e. 16MHz is divided by 256 and the timer increment its value at 16000000/256 = 62500Hz i.e. 62.5KHz
And timer overflows when its value changes from 255 to 0 so the frequency of Overflow is 62.5 KHz/ 256 = 244.140625 Hz
As we switch display in each ISR the display of switching is 244.14 Hz since 3 switching are needed to completely draw a three digit number so the frequency of display update is 244.14/3= 81.38 Hz

So we draw the complete digit approximately 81 times per second !!!
This is too fast for our eyes to catch up so we see all three digits lit simultaneously.

So friends this was the theory behind one of the most used techniques in MCU design world. You must have seen these Seven Segments used in weighing machines,PCOs and even currency counters in banks they all use this technique of Multiplexing the displays. Many microcontroller projects published in net of magazines uses multiplexed seven segment displays and this article will help you understand them.

I said it all, now its your turn so all of you get ready for some typing because I need your comment on this post, simply say anything you think good or bad. While I write the next tutorial I will be heartily waiting for your comments.

Whats next

We will write code for this in AVR-GCC.