EE 675: Microprocessor Applications in Power Electronics - Assignment 4

Jayesh Choudhary 170070038

1 BUFFST_x

We initialize a variable with the starting address of RAM block-x which we keep on incrementing once we store a value at that address. We then move the address to auxiliary register. We initialize another variable count to keep track of the 4th instant.

We see if the block is full or not by comparing the current address with last address of the block + 1. If the block is full, we exit from the sub-routine. Otherwise, we see if the variable count is 3 or not. If it is 3, we jump to another sub-routine where we put the value of AH in the address taken from auxiliary register and then increment the auxiliary register. We also reset count to 0. We then store this incremented address to the variable which we use in upcoming iterations. If count is not equal to three then we increment the count by 1 and exit from the sub-routine.

2 TIMER ROUTINE

Timer initialization is taken from ttest.asm file and only the timer interrupt subroutine is modified. Here we perform the numerical integration to get the values for Simple harmonic Motion. We call the BUFFST_x subroutine to store the value of x and y, and then we update their values by using practical numerical integration to get rid of the decaying problem.

The equations involved in the numerical integration are as following:

$$x_{n+1} = x_n + (\omega T).y_n \tag{1}$$

$$y_{n+1} = y_n - (\omega T).x_{n+1} \tag{2}$$

Here we need to take care of the corner cases or the cases where overflow can happen. This might cause sudden jumps from 0x7fff to 0x8001 or 1 to -1 or vice-versa. We use conditional jumps and overflow flag to take care of corner cases.

2.1 Challenges

Figuring out the correct set of conditions for overflow was difficult because initially to keep the number of variables less, I was performing addition and subtraction using a single variable having value ωT. But due to some precision errors, after 3-4 cycles, sudden jump from 1 to -1 was still there. I tried various nested conditional branching to take care of this but was not successful. Finally, to get rid of this, another variable was initialized carrying the value of $-\omega T$ and after that only 2 conditional branching were sufficient to take care of the overflow.

2.2 Screenshot

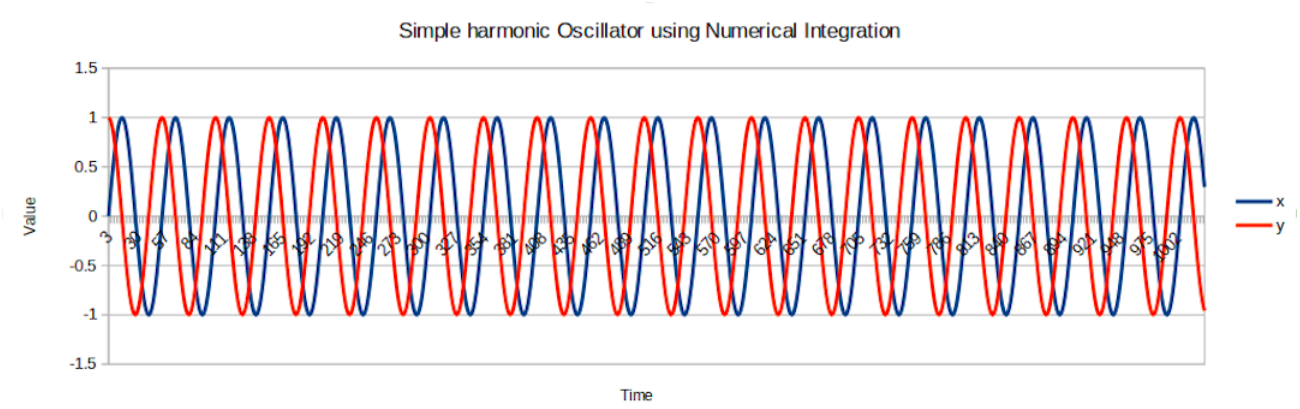

Figure 1: Screenshot from the chart generated in Libre-Office

As we can see in the figure, perfect sinusoidal waves are obtained. Here we have taken 1024 starting values stored in the RAM blocks. To extract the values, sq command was used instead of si command as si command gives hex values whereas sq gives decimal values and it was thus much easier to plot in Libre-Office.

The txt files for x and y used to generate the chart are also stored in a sub-directory named testing_4 in the home directory in the home server. The bin file along with the asm files, cmd files and Makefile used during simulation are also there.