MCU: Interrupts and Timers

Ganesh Pitchiah

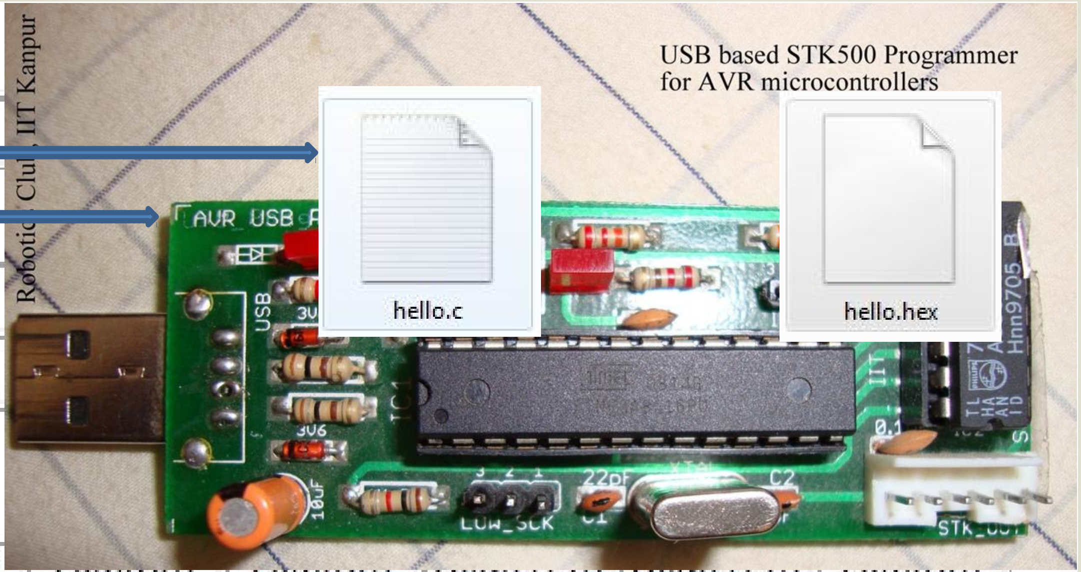
Electronics Club IIT Kanpur

What’s an MCU ?

Electronics Club IIT Kanpur

Important Points to Remember from Yesterday :

	PB7	PB6	PB5
Function	Output	Output	Input
DDRB	1	1	0



Value	High(+5V)	High(+5V)	Low(0V)	Low(0V)	Low(0V)	High(+5V)	High(+5V)	Low(0V)
PORTA	1	1	0	0	0	1	1	0

4. Program the MCU

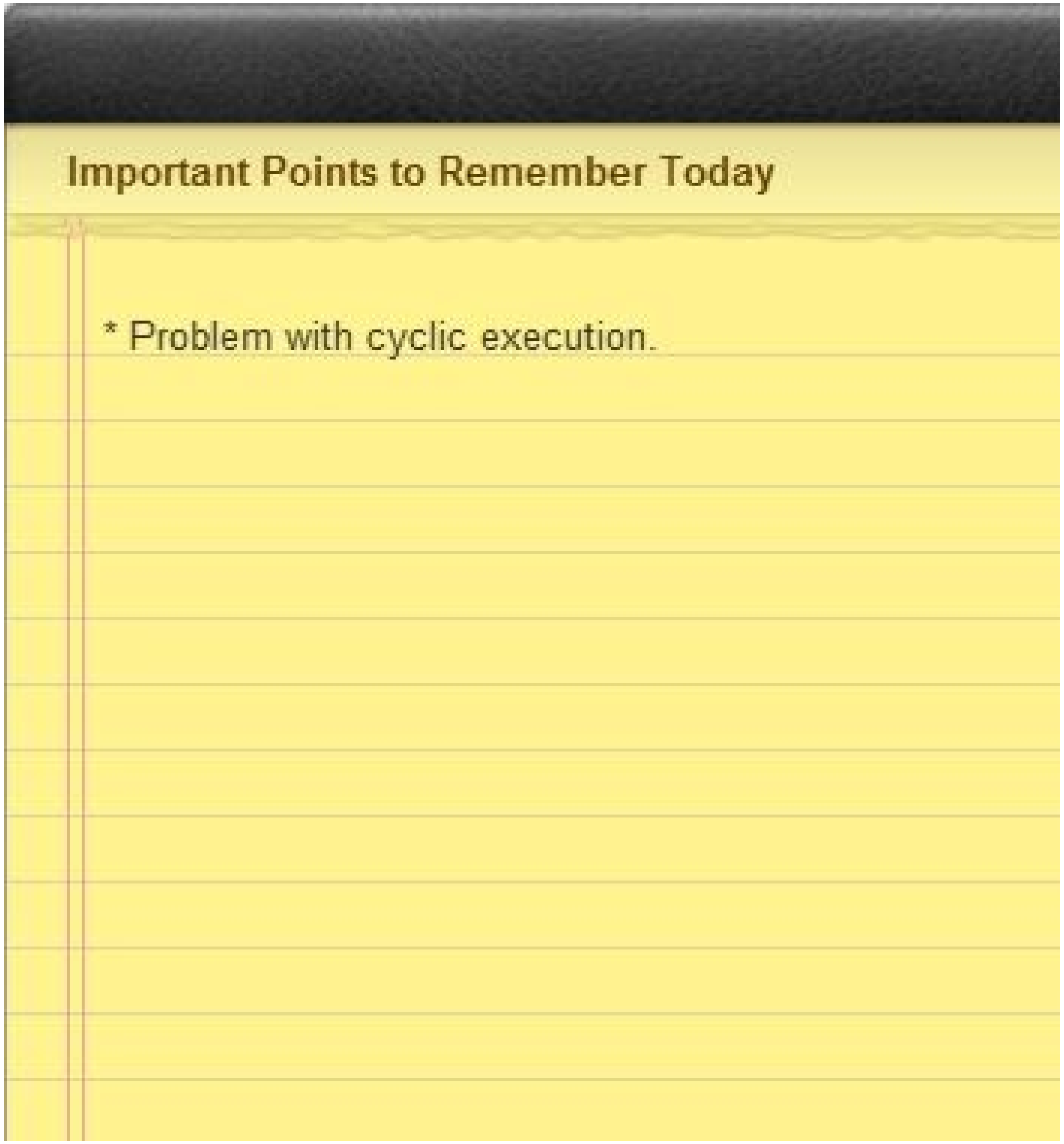
Electronics Club IIT Kanpur

Code for Switching LED

```
int a;               // Define variable a to store value of voltage
while(1)
{
a = PINA.0;  // read value at pin A.0 (make sure it is input)
If (a==1)      //  if voltage is 5V
PORTA.1=1;   // Light the LED
else
PORTA.1=0;  // Turn off the LED
}
```

The Problem

```
-------
    while(1){
    ---- -> Check value of a
    ---- -> Event ‘A’ :  a == 1
    ----
    ----
    ---- -> Event ‘B’ :  a == 0
    ----
    ----
    ----
    ---- -> Suppose event ‘A’ occurs here
 ----
}
```



Important Points to Remember Today

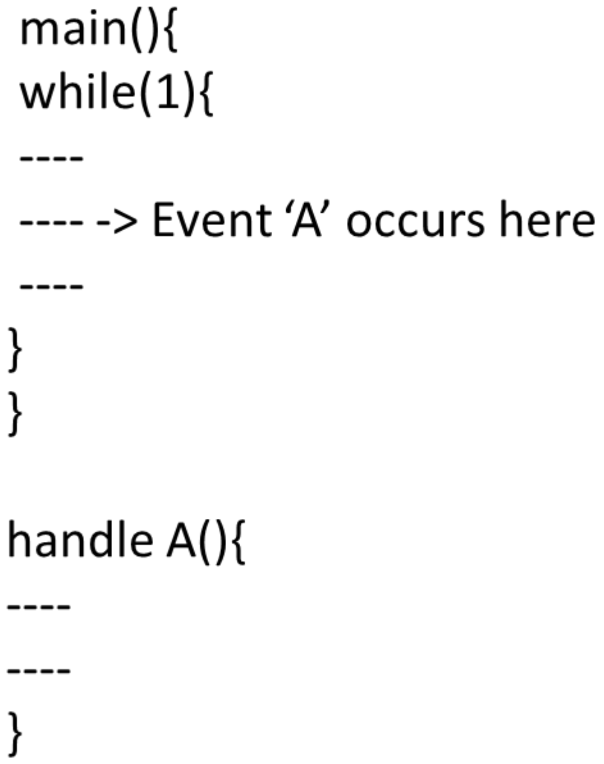
* Problem with cyclic execution.

Interrupt Means

- To cause or make a break in the continuity or uniformity of (a course, process, condition, etc.

The Problem & Solution

```
-------
    while(1){
    ---- -> Check value of a
    ---- -> Event ‘A’ :  a == 1
    ----
    ----
    ---- -> Event ‘B’ :  a == 0
    ----
    ----
    ----
    ---- -> Suppose event ‘A’ occurs here
 ----
}
```



```
 main(){
 while(1){
 ----
 ---- -> Event ‘A’ occurs here
 ----
}
}

handle A(){
----
----
}
```

Interrupts

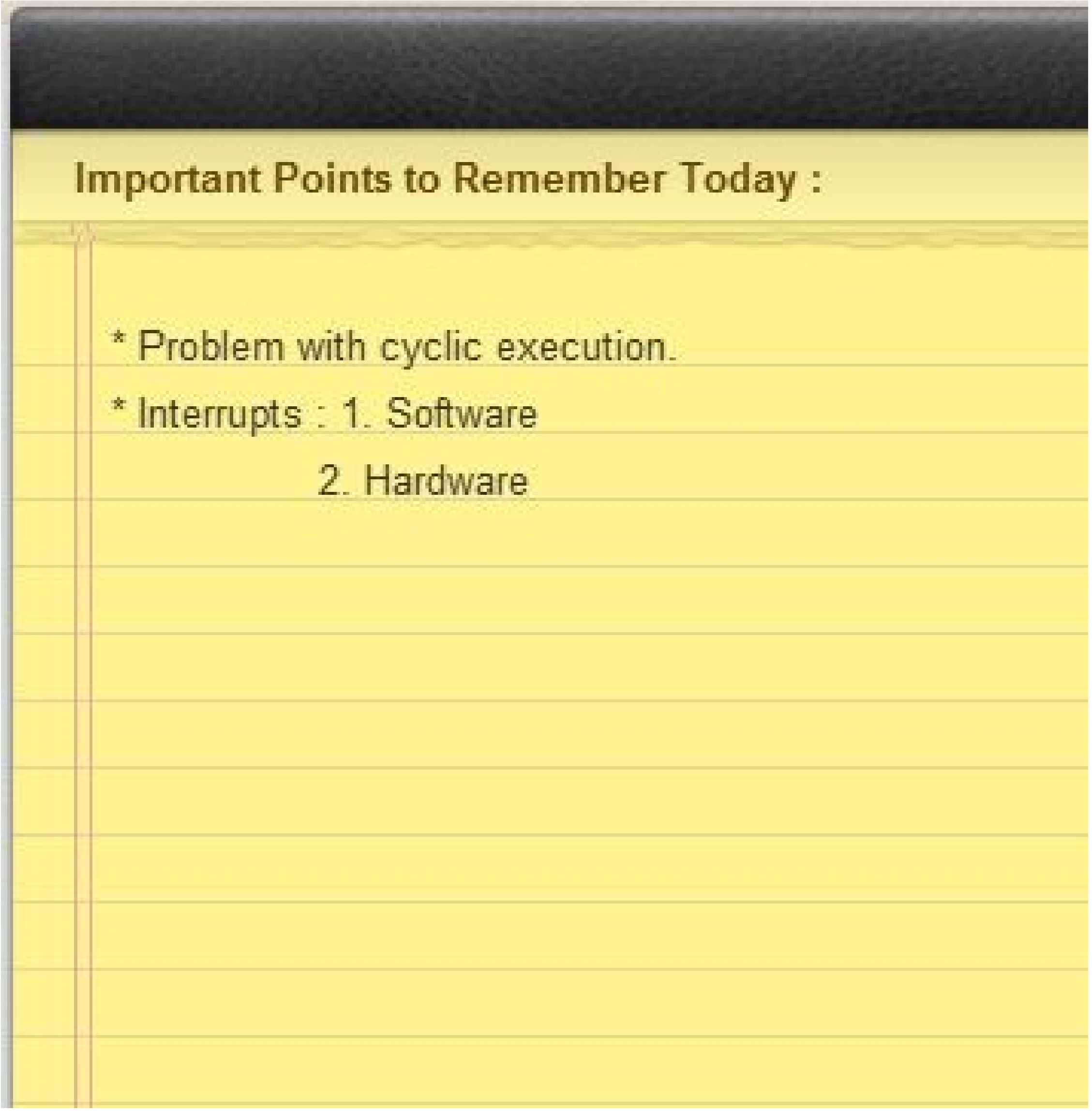
- Software Interrupt

```
while (1)
{
keep checking all events only
}
```

- Hardware Interrupt

Why Interrupts?

- Interrupts are special events that can “interrupt” the normal flow of a program.
- The processor stops the normal program, handles the interrupt, and then resumes its normal work.



Important Points to Remember Today :

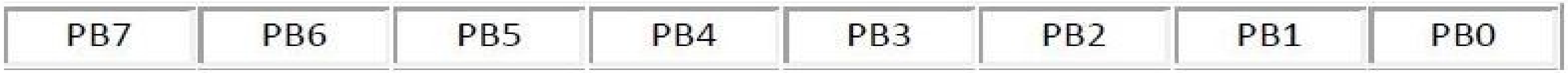
* Problem with cyclic execution.

* Interrupts : 1. Software

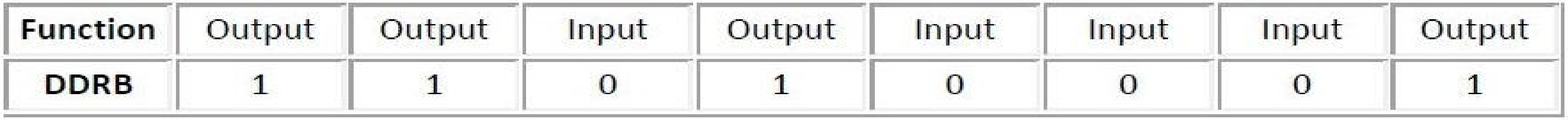
  2. Hardware

Registers

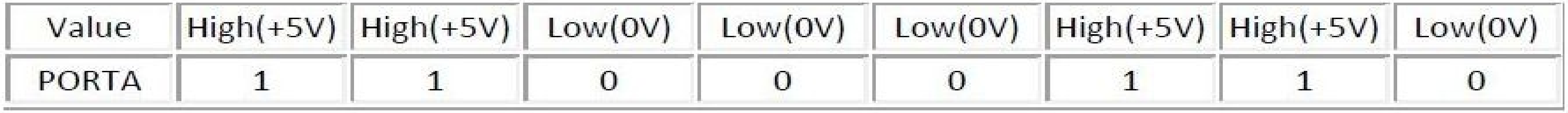
- Small amount of storage available in MCU/PC.
- Ex: A flip flop stores 1 bit of memory



PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0

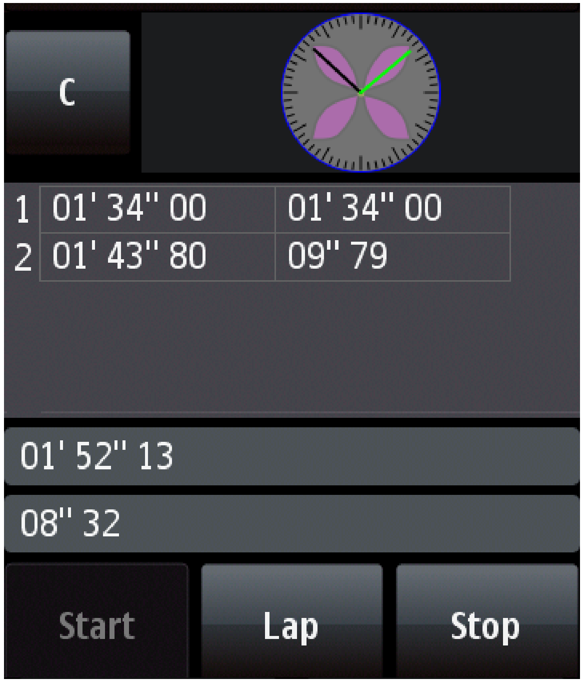


Function	Output	Output	Input	Output	Input	Input	Input	Output
DDRB	1	1	0	1	0	0	0	1



Value	High(+5V)	High(+5V)	Low(0V)	Low(0V)	Low(0V)	High(+5V)	High(+5V)	Low(0V)
PORTA	1	1	0	0	0	1	1	0

Timers



1	01' 34'' 00	01' 34'' 00
2	01' 43'' 80	09'' 79

01' 52'' 13

08'' 32

Timers

- A timer is a register.

7	6	5	4	3	2	1	0

- 255 -> Maximum value

  254

  .

  .

  .

  0 -> Starting value

Timers

- 8-bit Register and Starts with 0

1	1	1	1	1	1	1	1

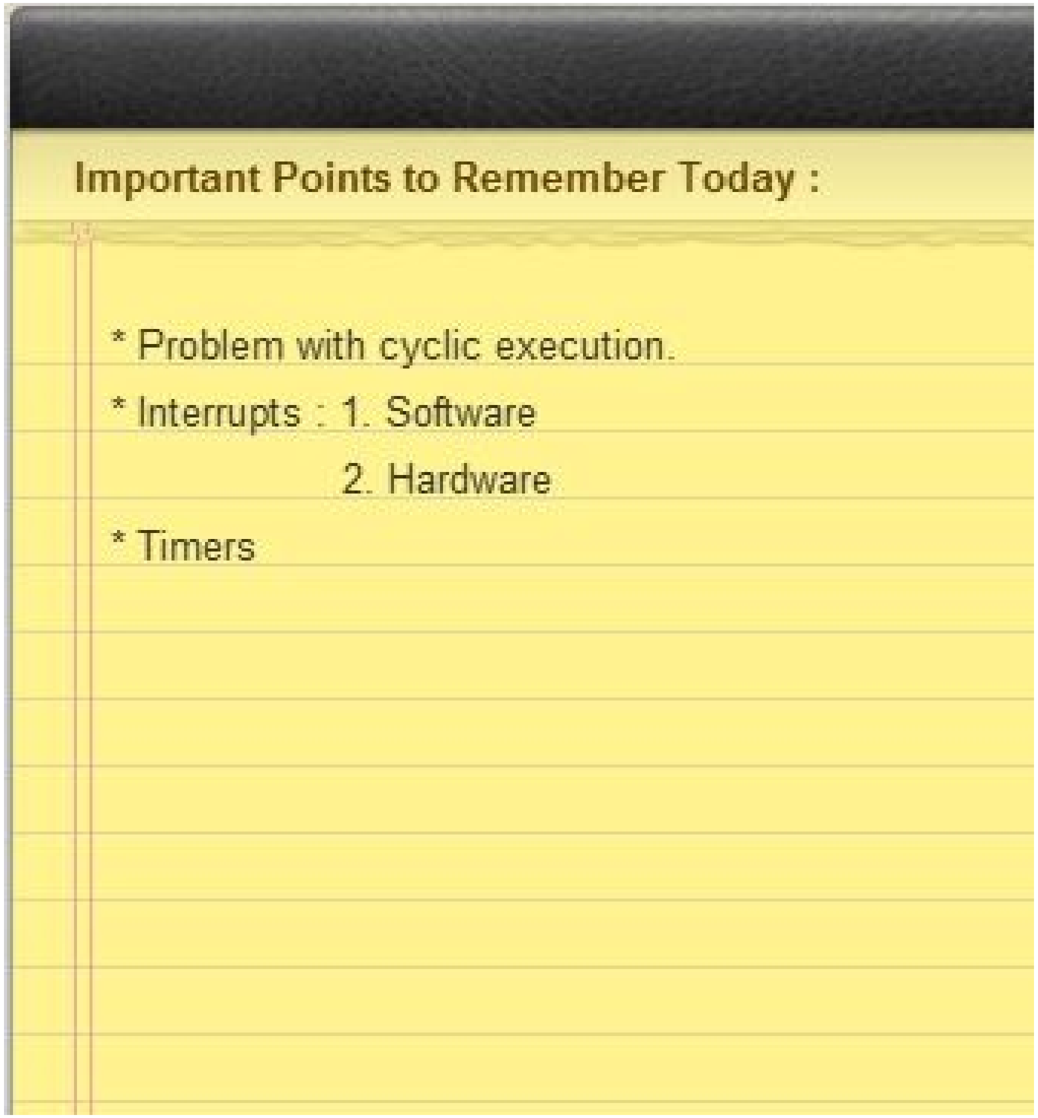
- Increase by 1,after each period.
- When the timer reaches its maximum value, in the next cycle, its value becomes 0 again and the process repeats itself.
- This process is **independent** of the CPU.

Simple statistics

- Maximum value of timer is ***n*** and clock period is ***t***, then:

  1. Timer period = t
  2. Timer cycle period = $(n+1)\times t$
  3. Frequency of timer (f) = $1/t$
  4. Frequency of timer cycle = $1/(n+1)\times t$

EC Electronics Club IIT Kanpur



Important Points to Remember Today :

* Problem with cyclic execution.

* Interrupts : 1. Software

2. Hardware

* Timers

Timers and Interrupts

- Timers can generate certain two interrupts:
  1. OVERFLOW interrupt and
  2. COMPARE MATCH interrupt.

OVERFLOW interrupt

- OVERFLOW is generated when a timer tries to exceed its maximum valu

0	0	0	0	0	0	0	0

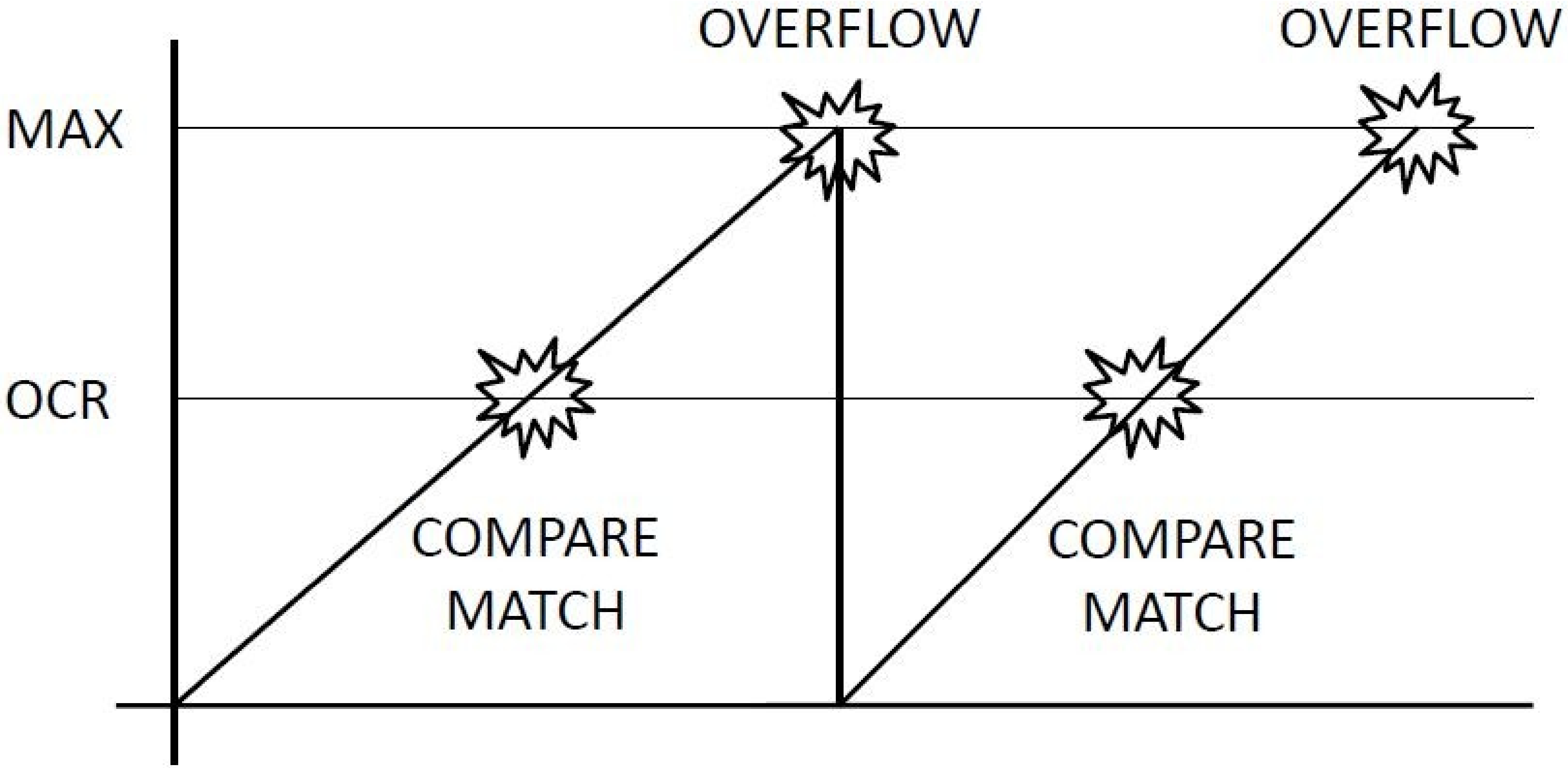
- The interrupt may or may not have a handler. In either case, the timer continues to run; remember: timers are **independent** of the CPU.

OVERFLOW statistics

- Suppose a timer of maximum value ***n*** has a time period ***t*** (also called as clock period).
    1. Timer cycle frequency = $1/(n+1)\times t$
    2. OVERFLOW interrupt frequency = $1/(n+1)\times t$
- If OVERFLOW interrupt is enabled, then an interrupt is generated in every cycle.

EC Electronics Club IIT Kanpur

OVERFLOW and COMPARE MATCH



EC Electronics Club IIT Kanpur

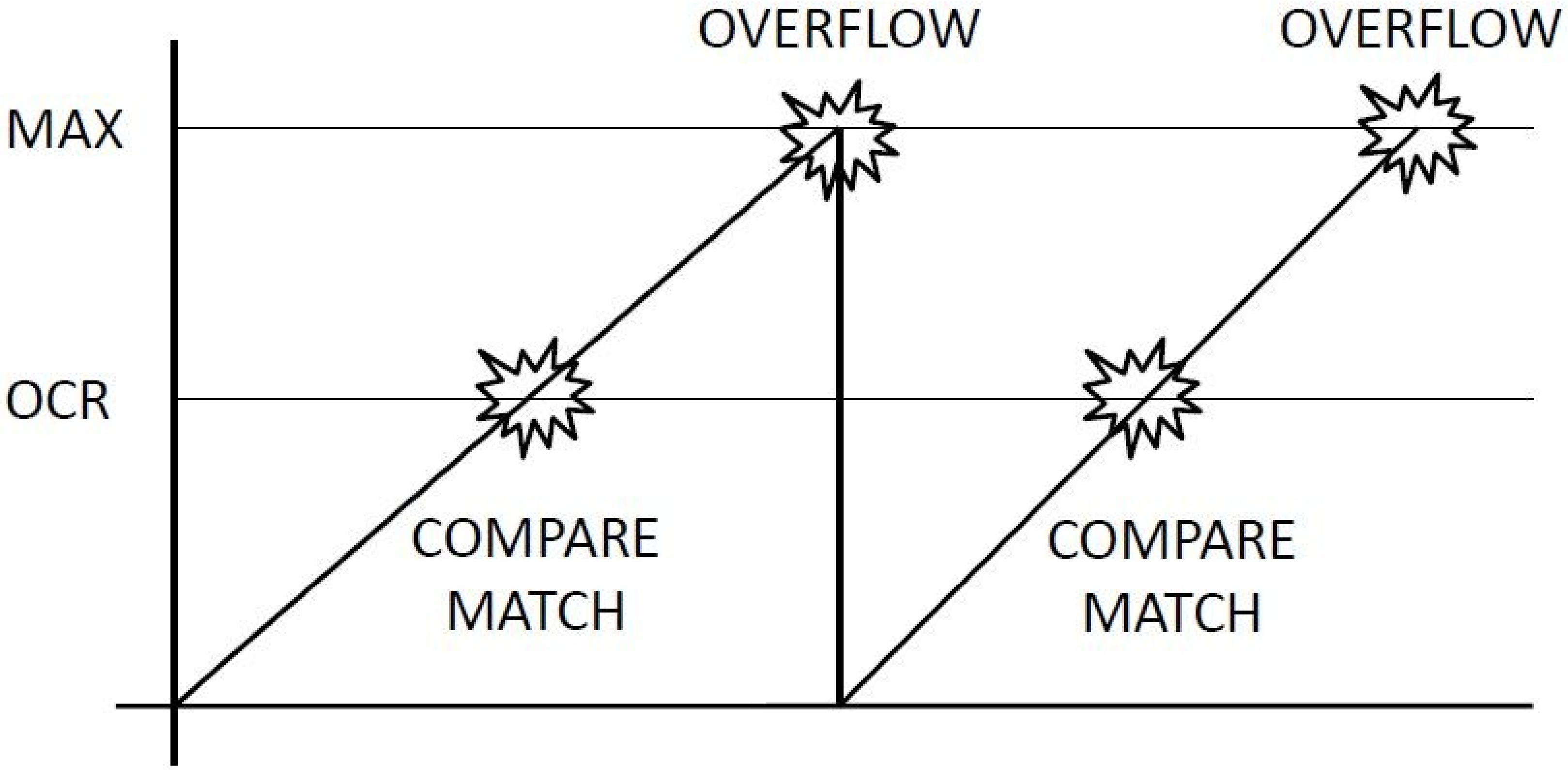
COMPARE MATCH interrupt

- There is a register called as **OCR** (Output Compare Register), whose value we can set.
- Before incrementing, the value of the timer is compared to **OCR**. If the two are equal, a COMPARE MATCH interrupt is generated.

COMPARE MATCH statistics

- Suppose a timer of maximum value ***n*** has a time period ***t*** (also called as clock period).
  1. Timer cycle frequency = $1/(n+1)\times t$
  2. COMPARE MATCH interrupt frequency
     = $1/(n+1)\times t$
- If COMPARE MATCH interrupt is enabled, then an interrupt is generated in every cycle.

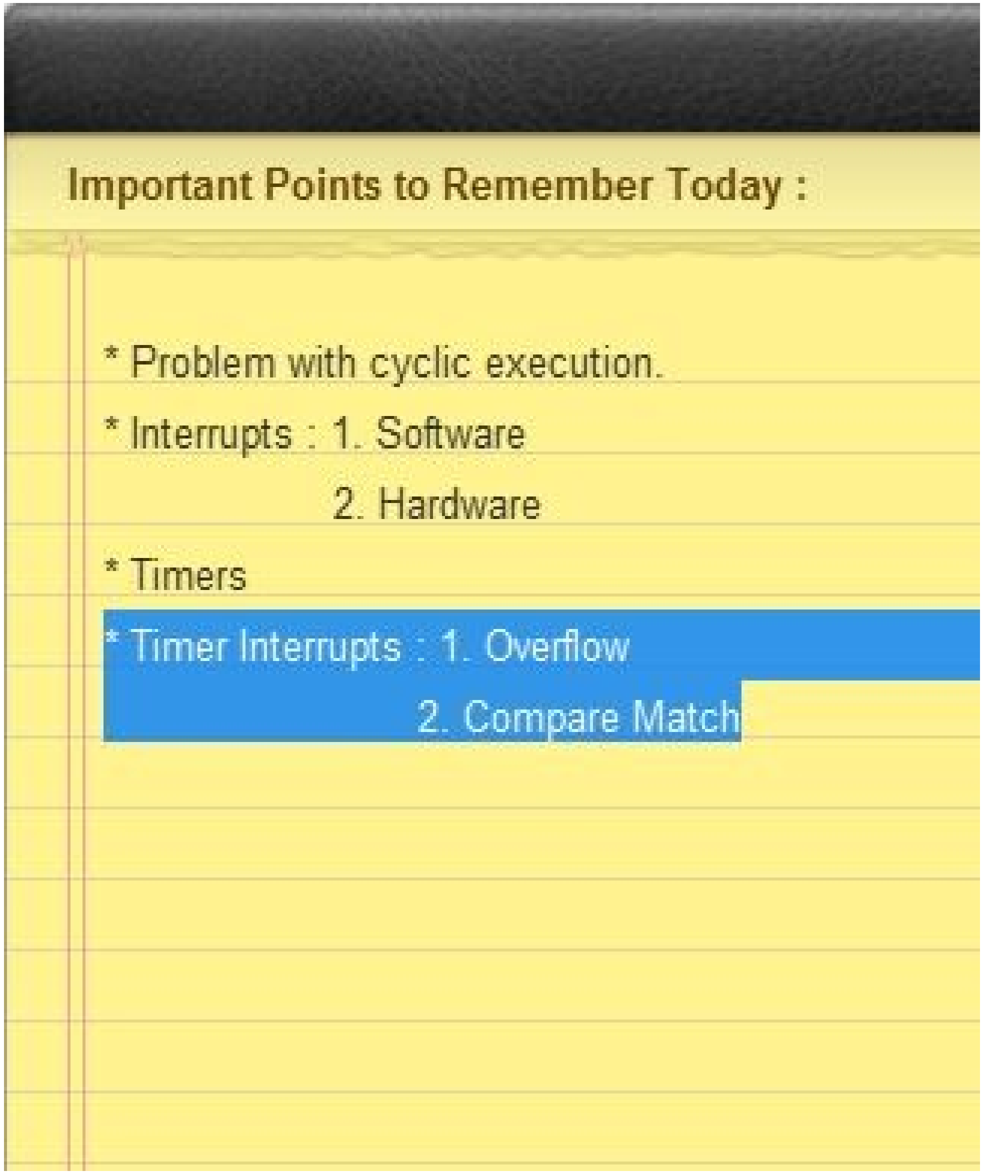
OVERFLOW and COMPARE MATCH



Electronics Club
IIT Kanpur

Summary of Timers

- A timer is not affected by interrupts: it generated interrupts, but it does not stop running because of them.
- Interrupts is how timers are useful. Sample applications: digital clock, periodic events (such as blinking LEDs quickly for POV globe), etc.



Important Points to Remember Today :

* Problem with cyclic execution.

* Interrupts : 1. Software
  2. Hardware

* Timers

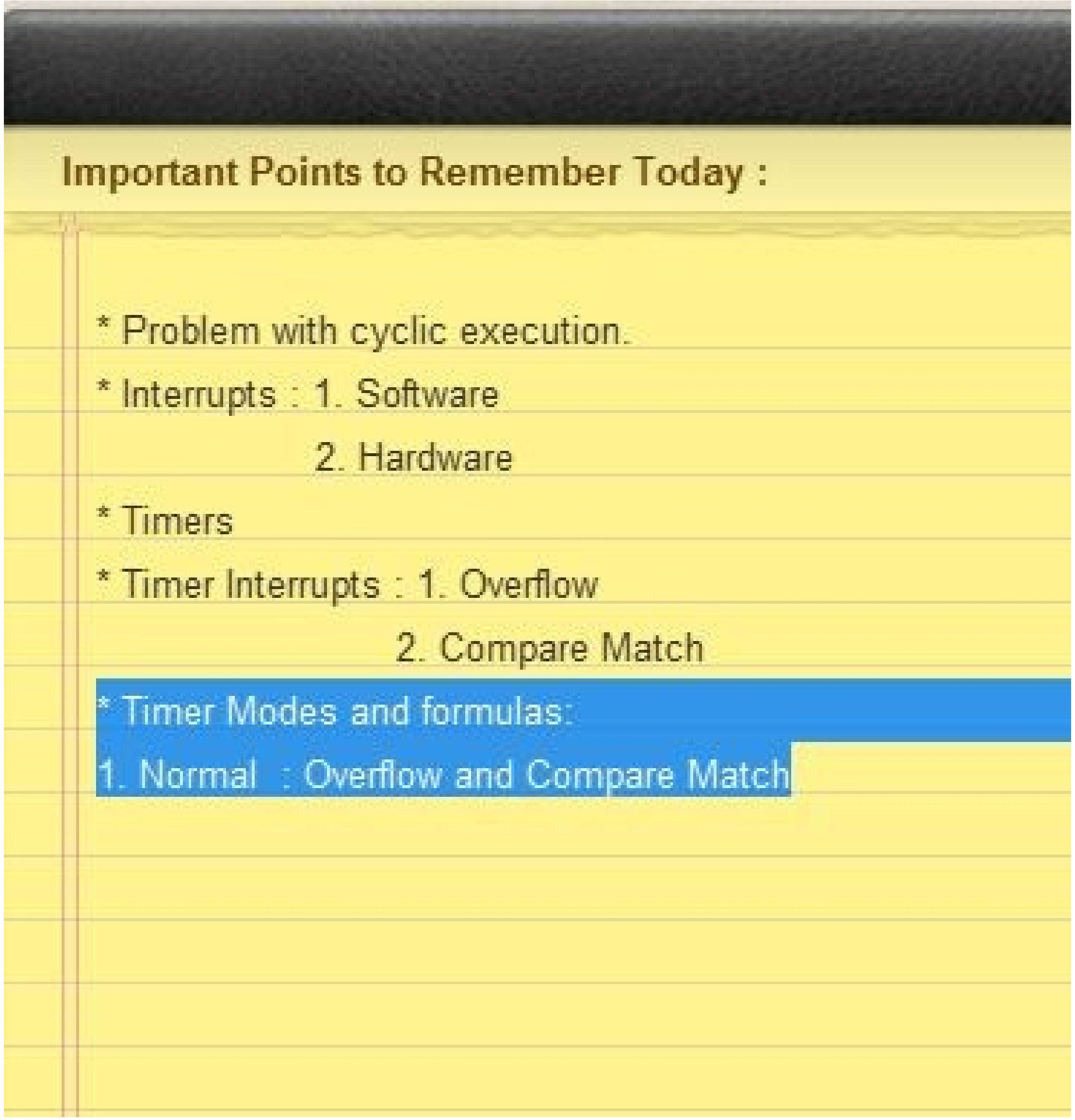
* Timer Interrupts : 1. Overflow
  2. Compare Match

Timer Modes

- A timer works in three modes: Normal, CTC and PWM.
- All three modes are again unaffected by interrupts, but all three modes can generate interrupts.
- The timer mode used so far in this presentation is normal mode.

Normal Mode

- Standard mode: Timer starts at 0, goes to maximum value and then resets itself.
- OVERFLOW and COMPARE MATCH interrupts generated as normal.



Important Points to Remember Today :

* Problem with cyclic execution.

* Interrupts : 1. Software
2. Hardware

* Timers

* Timer Interrupts : 1. Overflow
2. Compare Match

* Timer Modes and formulas:
1. Normal : Overflow and Compare Match

CTC (Clear Timer on Compare) Mode

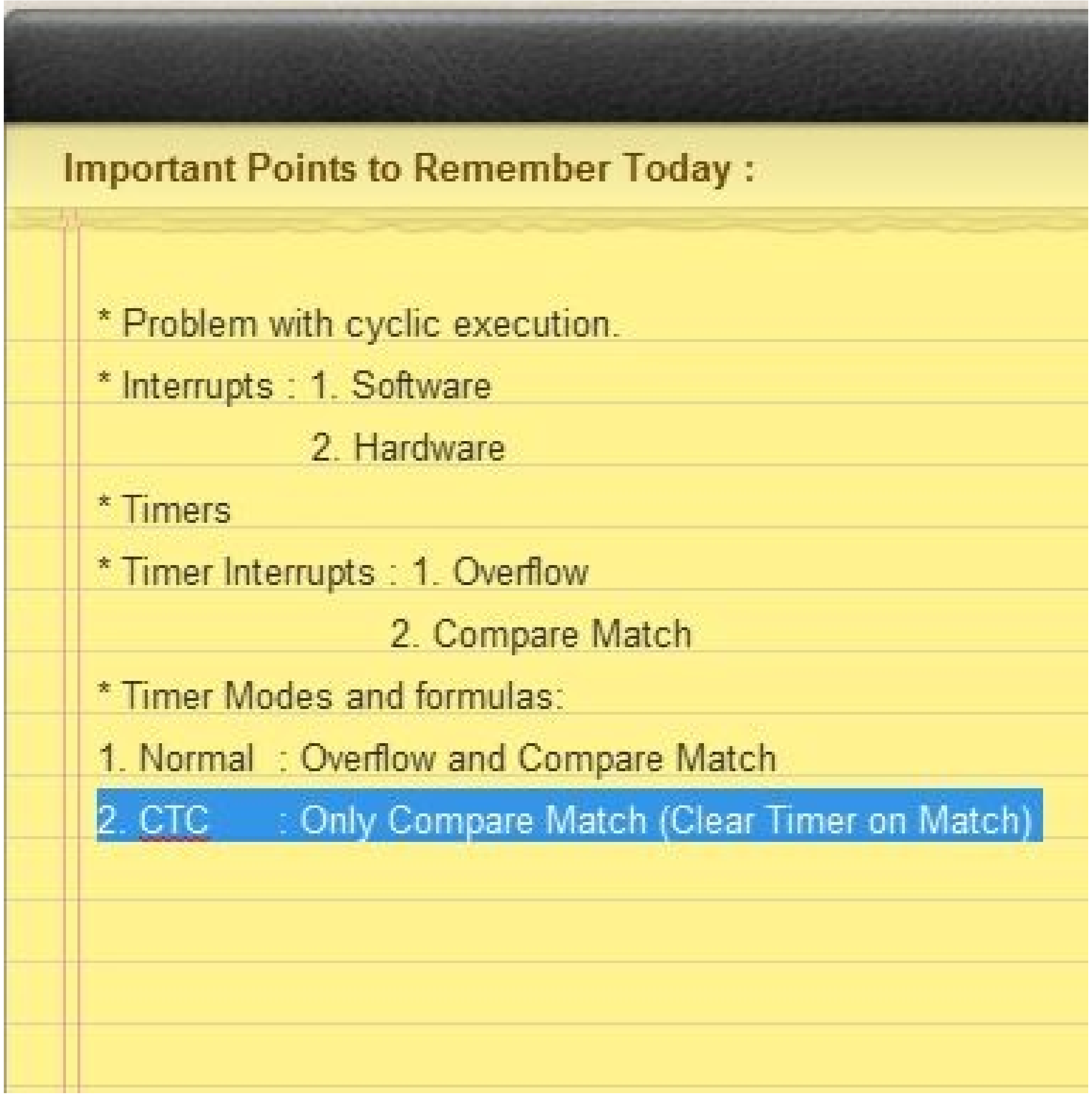
- Timer starts at 0 as usual, but instead of resetting after maximum value, it resets after reaching value specified in **OCR** register.

CTC mode statistics

- If clock time period is *t*:

  1. Timer cycle time period = $(OCR+1)\times t$
  2. Frequency = $1/(OCR+1)\times t$

- COMPARE MATCH interrupt will work normally, but OVERFLOW interrupt will not work (Why?).



Important Points to Remember Today :

* Problem with cyclic execution.

* Interrupts : 1. Software
                      2. Hardware

* Timers

* Timer Interrupts : 1. Overflow
                              2. Compare Match

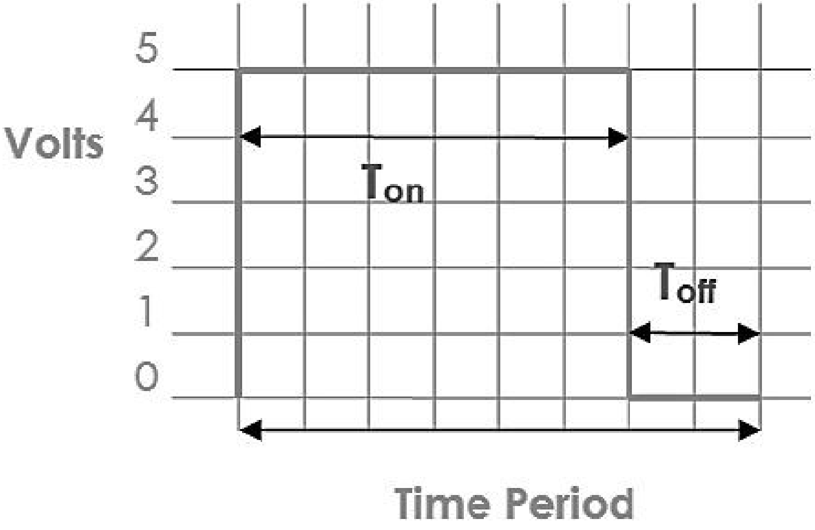
* Timer Modes and formulas:

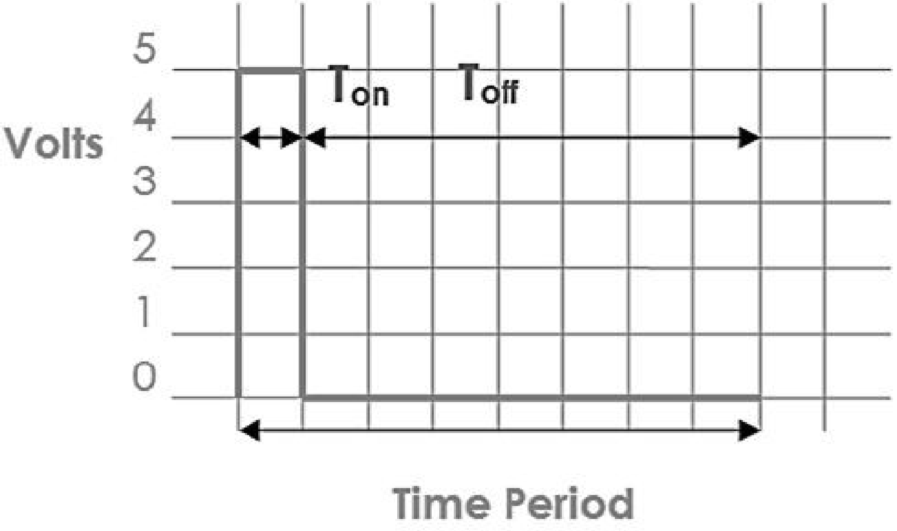
1. Normal : Overflow and Compare Match
2. CTC   : Only Compare Match (Clear Timer on Match)

PWM (Pulse Width Modulation) Mode

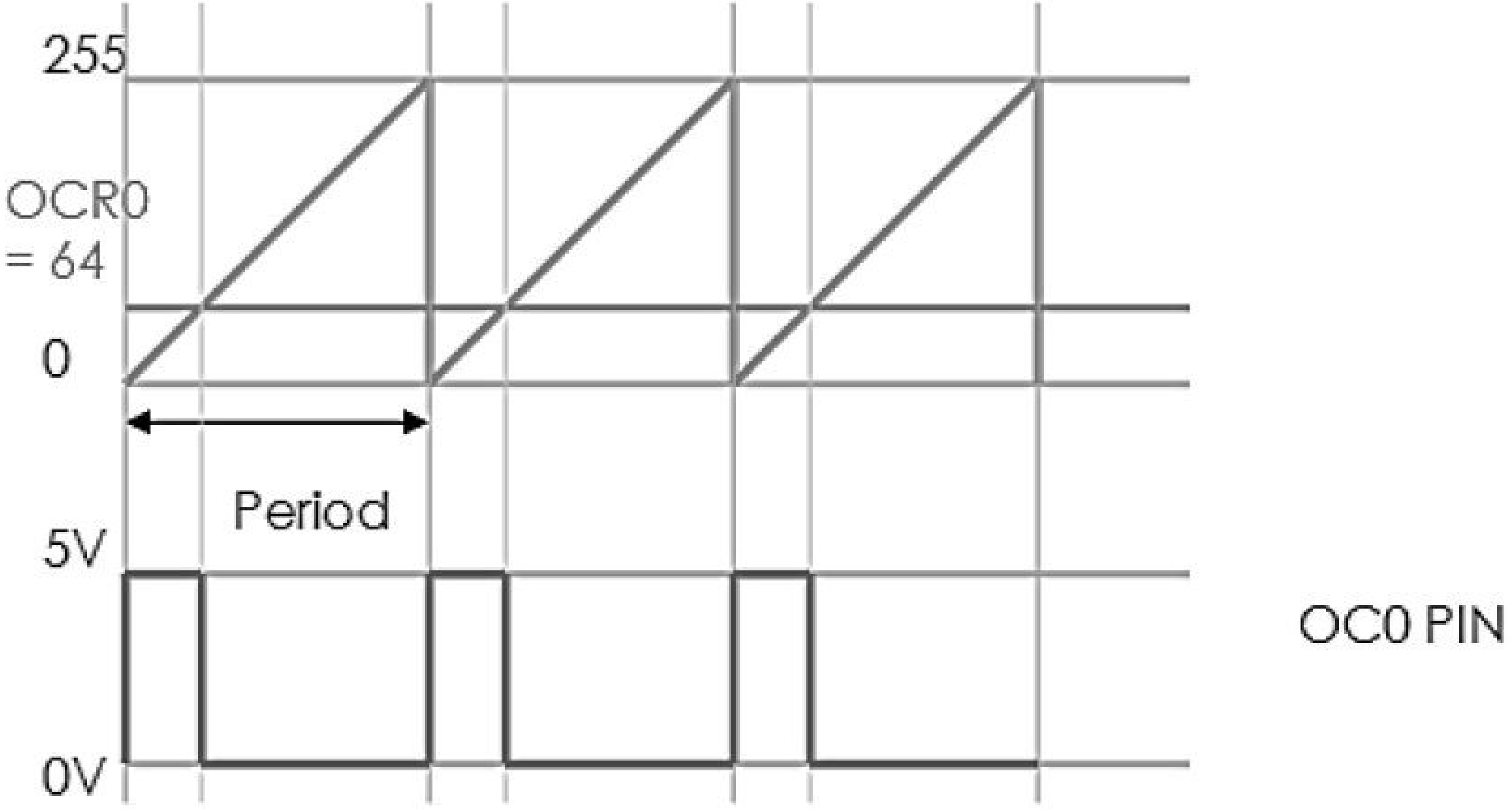
- Simple method of obtaining analog output of any value between 0 and 5V.
- Desired output is *x*% of 5V.
- If Ton = x% then average value is x% of 5V.

PWM(Pulse Width Modulation) mode





Electronics Club IIT Kanpur


255
OCR0
= 64
0
Period
5V
0V
OC0 PIN

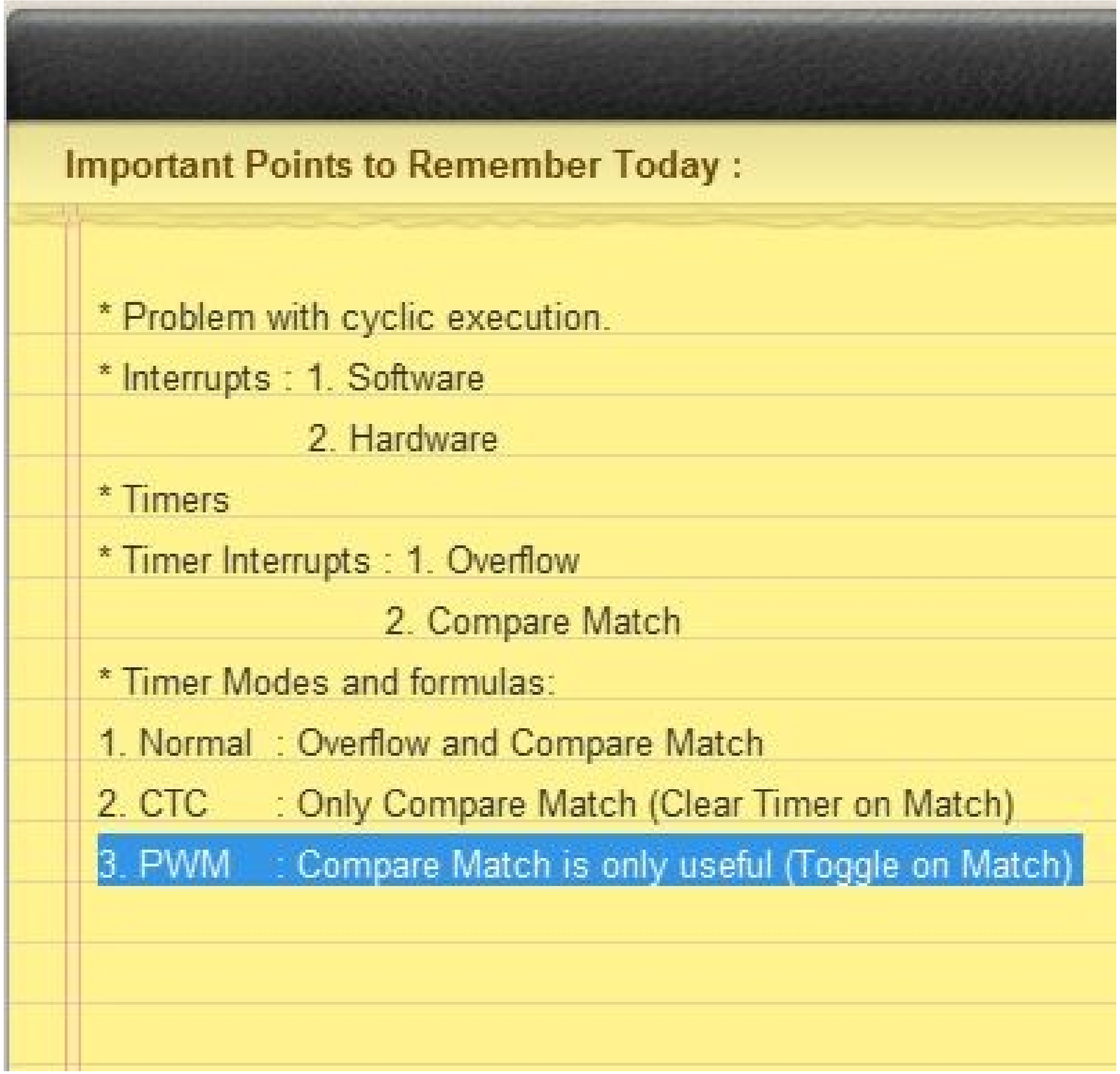
PWM statistics

- If clock time period is *t* and maximum timer value is *n*:

  1. Timer cycle time period $=(n+1)\times t$
  2. Frequency $=1/(n+1)\times t$
  3. Duty cycle $=[OCR/$
$(n+1)]\times 100\%$
  4. Output voltage $=[OCR/(n+1)]\times 5V$

- COMPARE MATCH interrupt and OVERFLOW interrupt will work properly.

Electronics
IIT Kanpur Club



Important Points to Remember Today :

* Problem with cyclic execution.

* Interrupts : 1. Software
                      2. Hardware

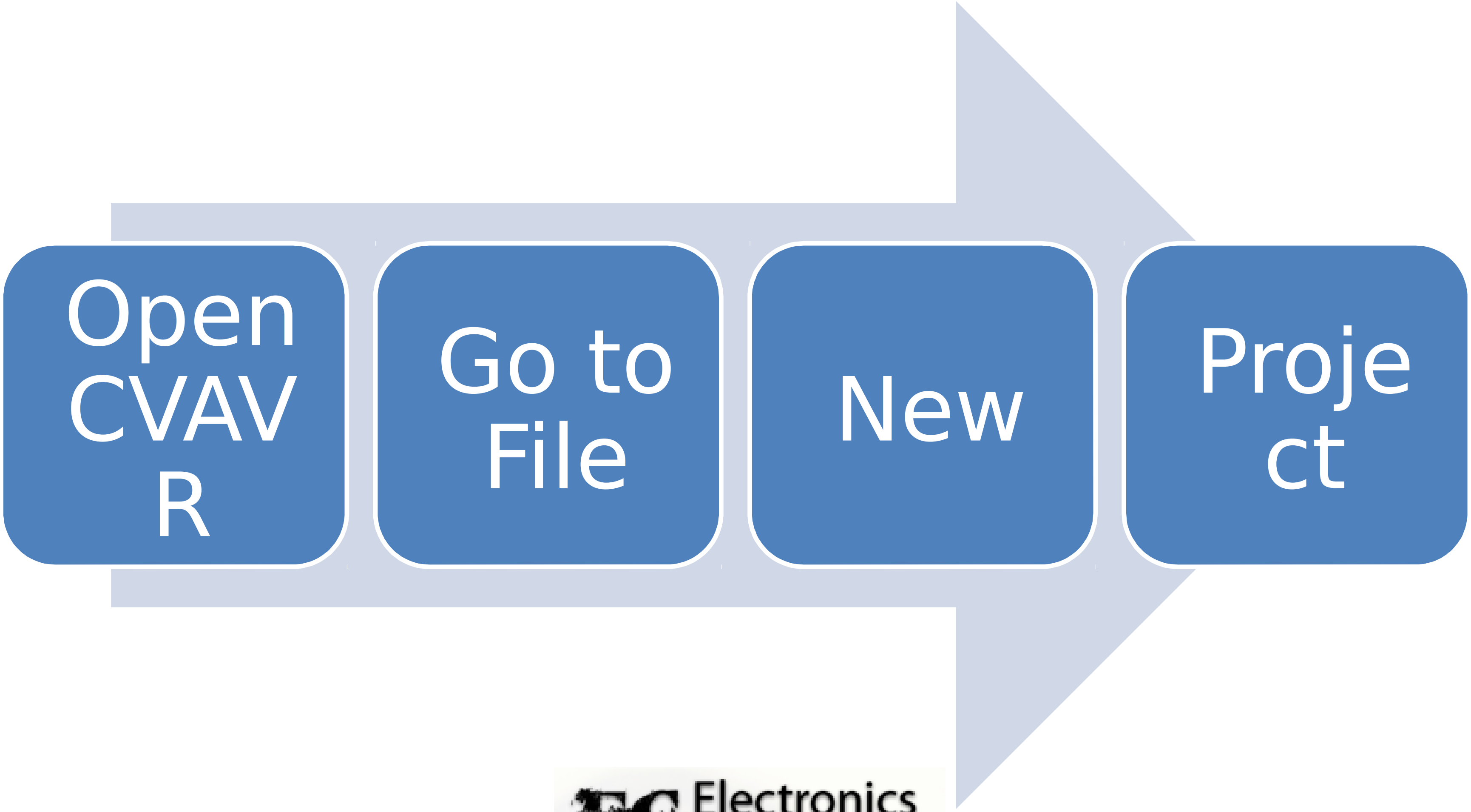
* Timers

* Timer Interrupts : 1. Overflow
                              2. Compare Match

* Timer Modes and formulas:

1. Normal : Overflow and Compare Match
2. CTC   : Only Compare Match (Clear Timer on Match)
3. PWM   : Compare Match is only useful (Toggle on Match)

Using CVAVR



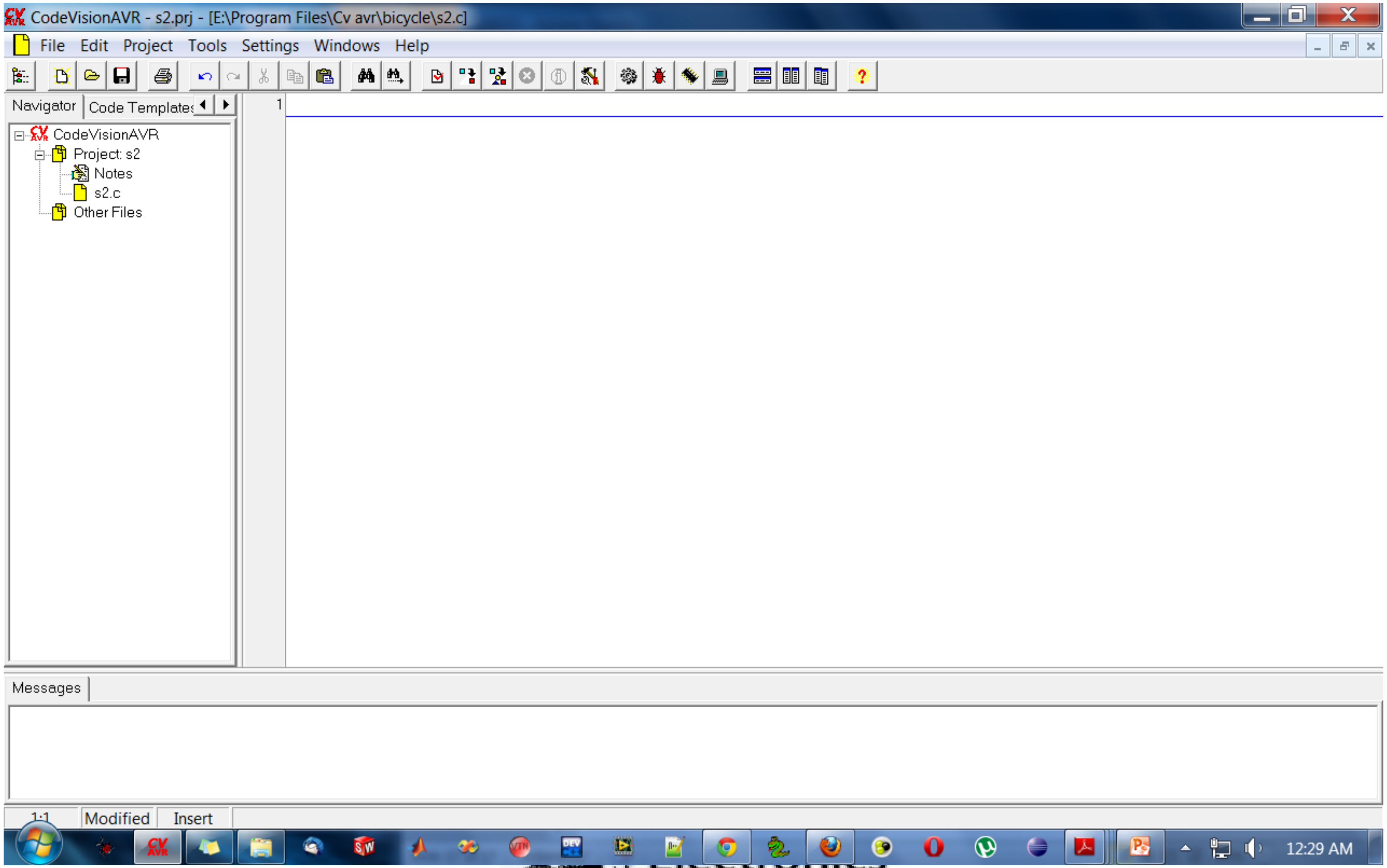
Open CVAVR

Go to File

New

Project

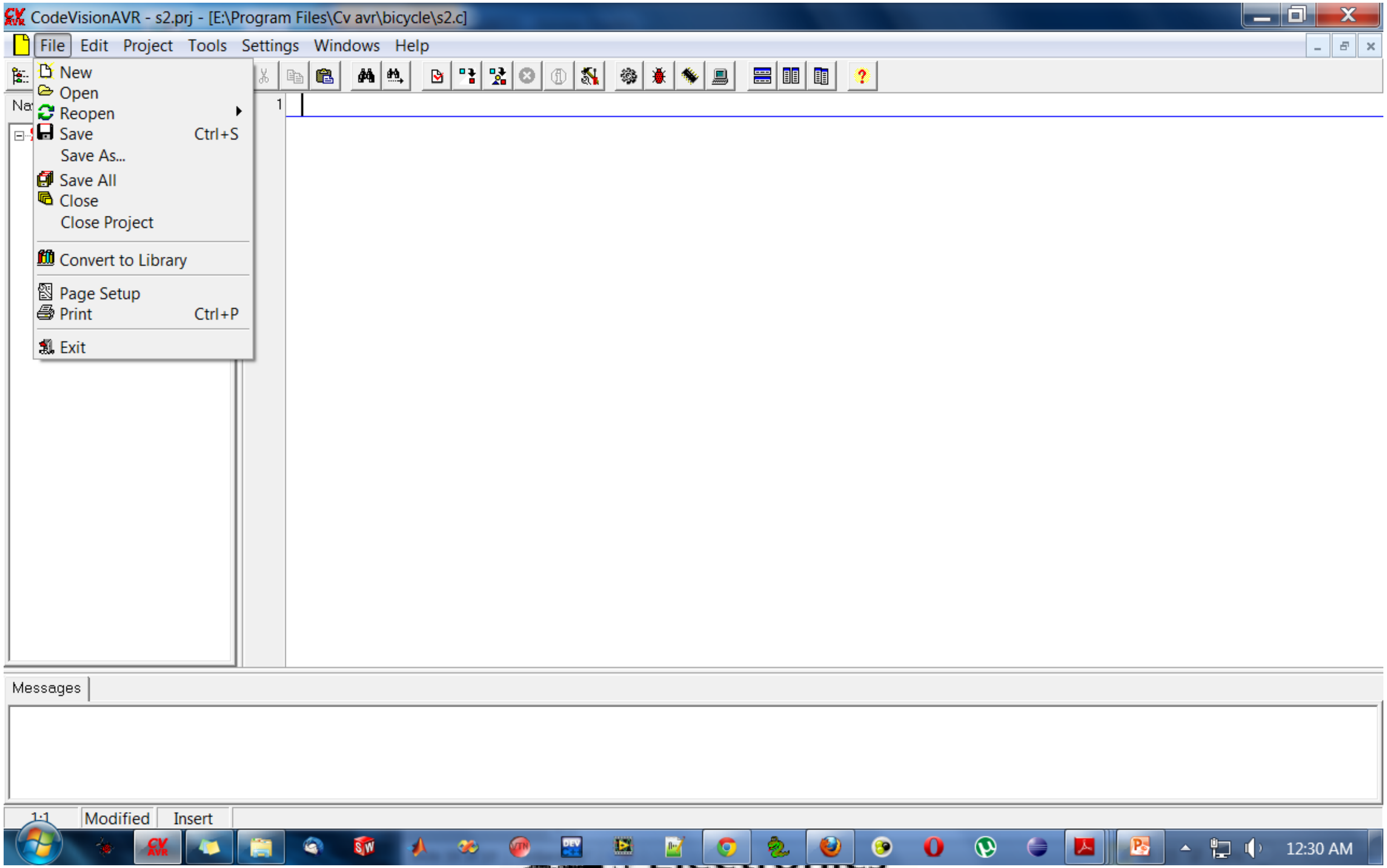
CVAVR Home



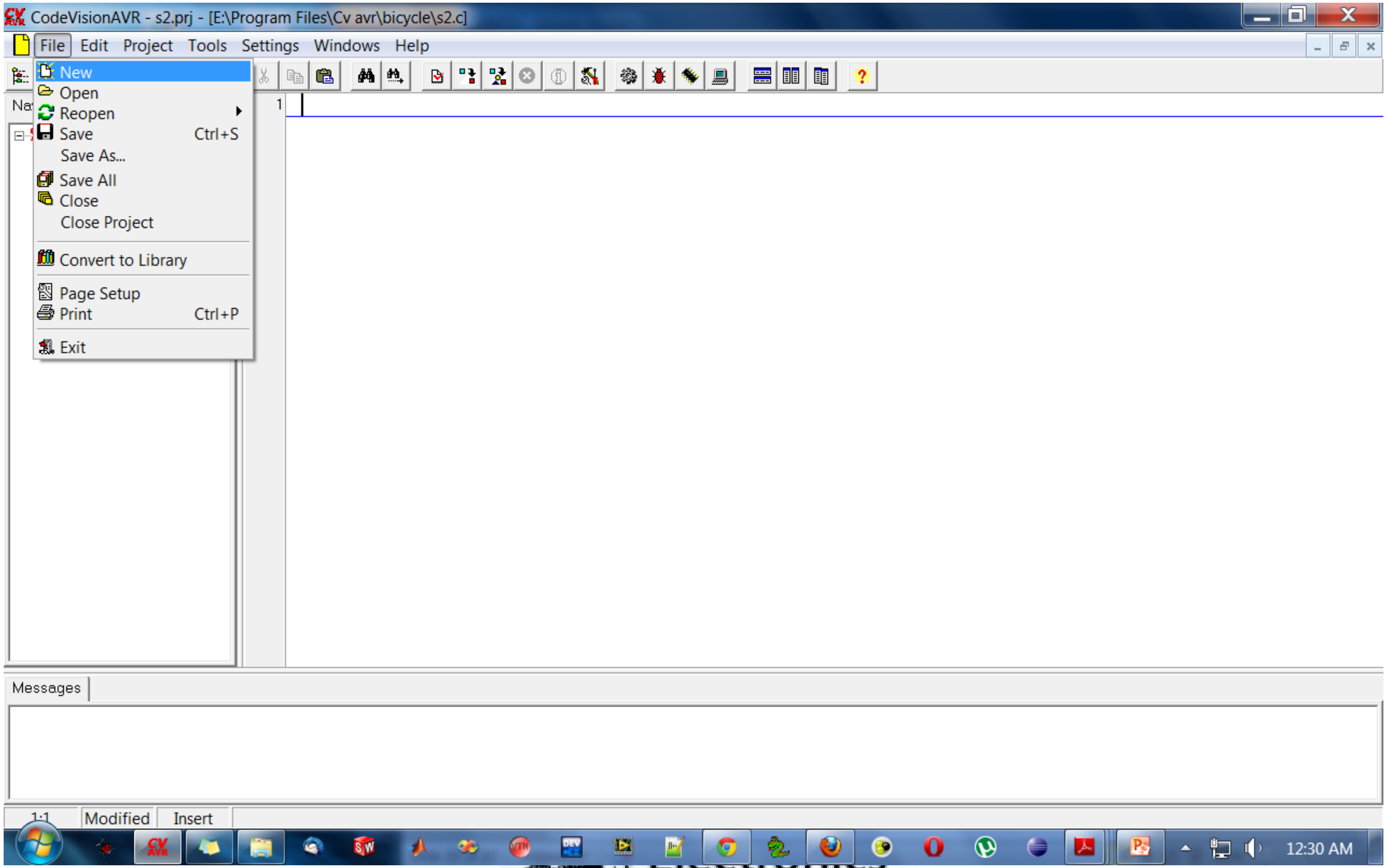
CodeVisionAVR - s2.prj - [E:\Program Files\Cv avr\bicycle\s2.c]
File Edit Project Tools Settings Windows Help
Navigator
Code Templates
CodeVisionAVR
Project: s2
Notes
s2.c
Other Files
1
Messages
1:1
Modified
Insert
12:29 AM

IIT Kanpur Club

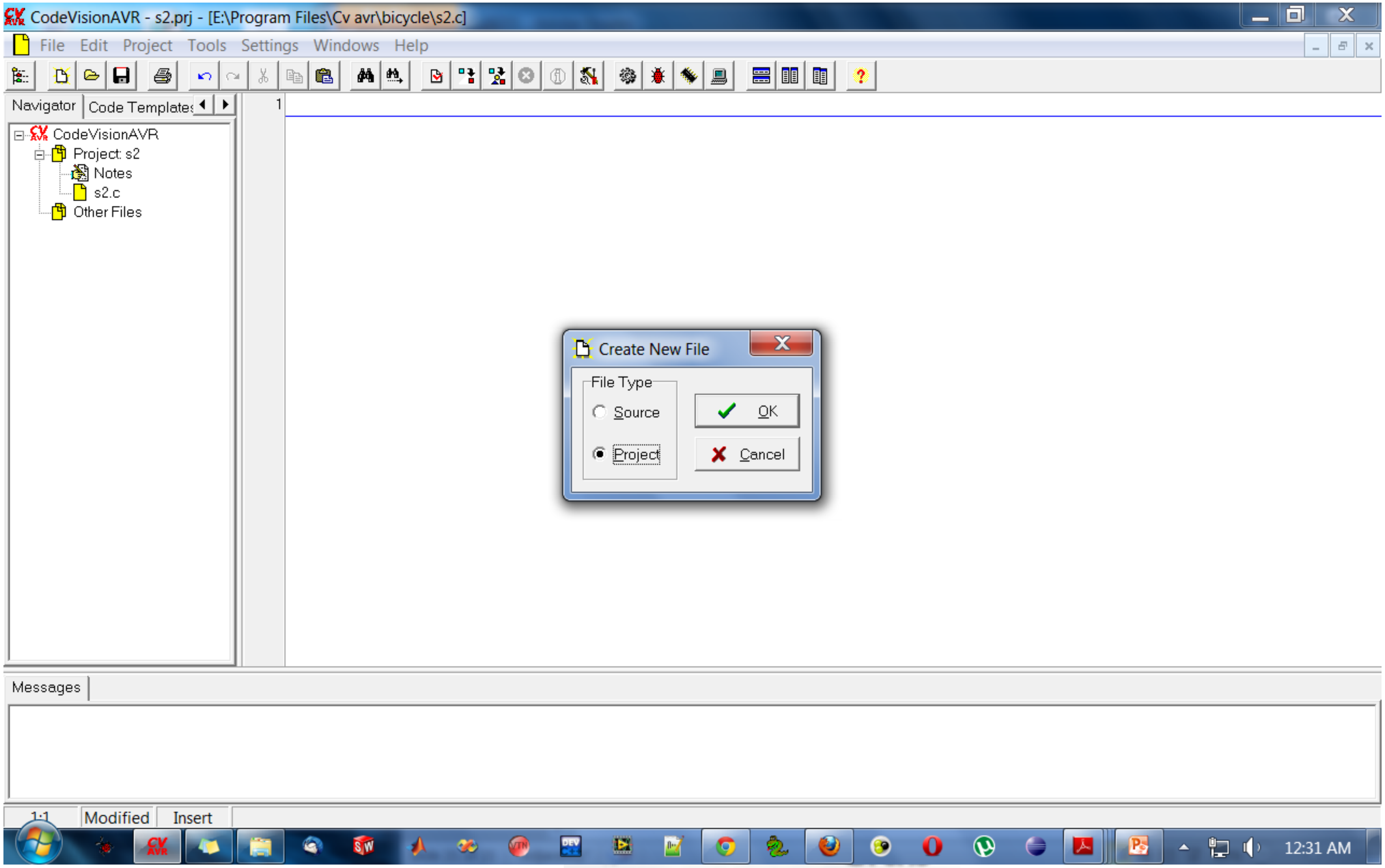
Go to File


CodeVisionAVR - s2.prj - [E:\Program Files\Cv avr\bicycle\s2.c]
File
Edit
Project
Tools
Settings
Windows
Help
New
Open
Reopen
Save Ctrl+S
Save As...
Save All
Close
Close Project
Convert to Library
Page Setup
Print Ctrl+P
Exit
Messages
1:1
Modified
Insert
12:30 AM

Choose New


CodeVisionAVR - s2.prj - [E:\Program Files\Cv avr\bicycle\s2.c]
File
Edit
Project
Tools
Settings
Windows
Help
New
Open
Reopen
Save Ctrl+S
Save As...
Save All
Close
Close Project
Convert to Library
Page Setup
Print Ctrl+P
Exit
Messages
1:1
Modified
Insert
12:30 AM

Choose Project



CodeVisionAVR - s2.prj - [E:\Program Files\Cv avr\bicycle\s2.c]
File Edit Project Tools Settings Windows Help
Navigator
Code Templates
CodeVisionAVR
Project: s2
Notes
s2.c
Other Files
Messages
1:1
Modified
Insert
12:31 AM

Create New File

File Type

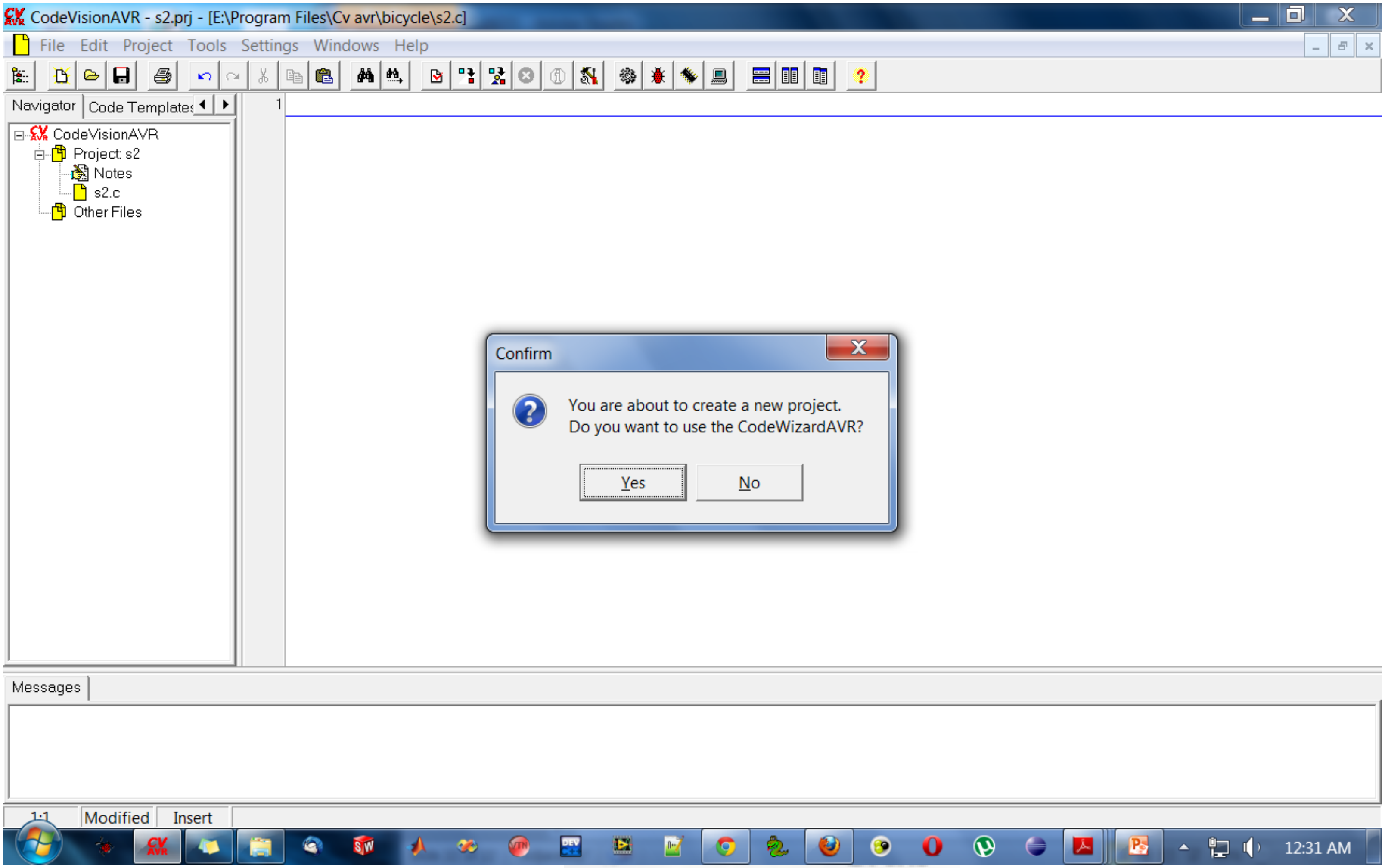
○ Source

● Project

OK

Cancel

Press Yes



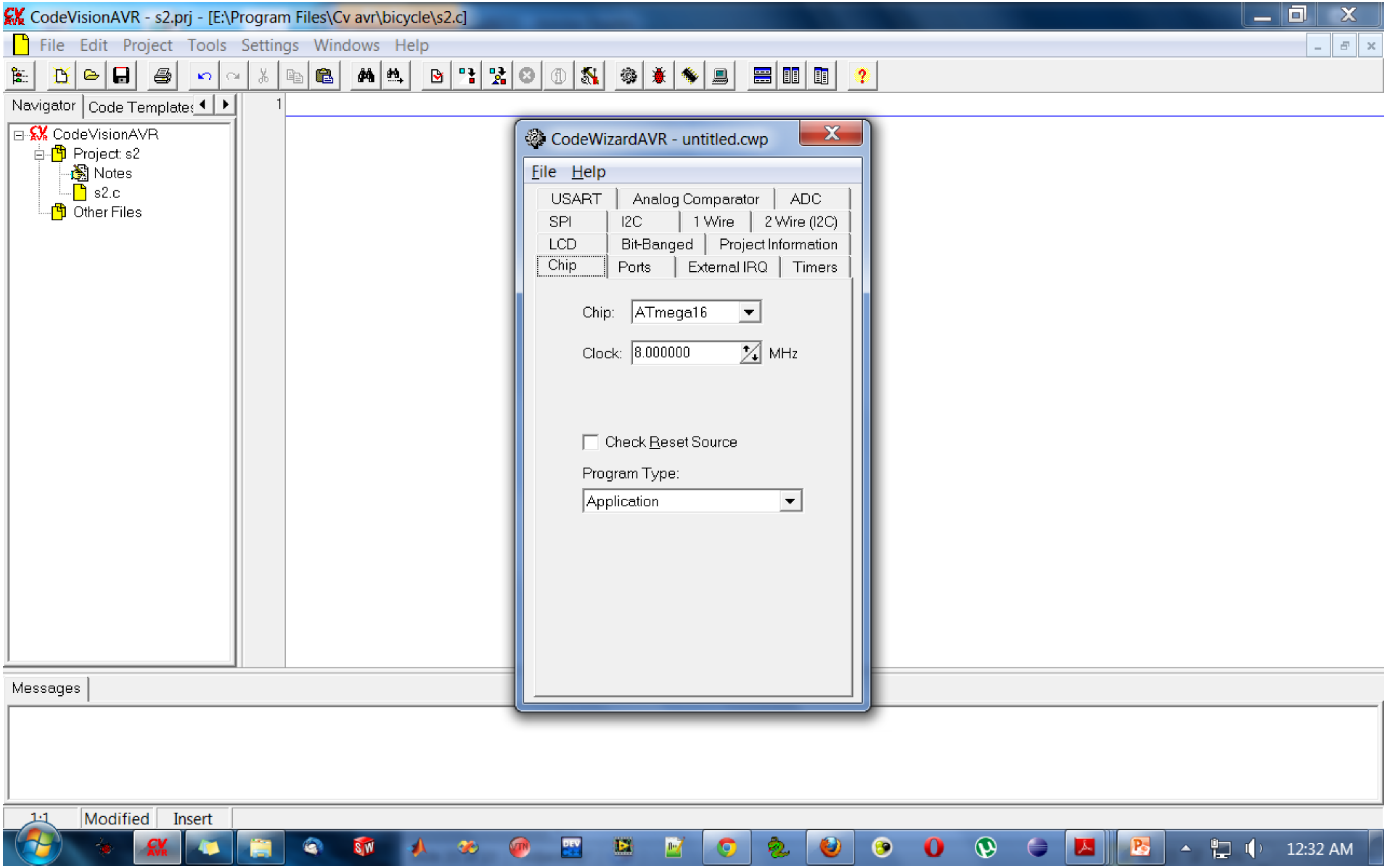
CodeVisionAVR - s2.prj - [E:\Program Files\Cv avr\bicycle\s2.c]
File Edit Project Tools Settings Windows Help
Navigator
Code Templates
CodeVisionAVR
Project: s2
Notes
s2.c
Other Files
Messages
1:1
Modified
Insert
12:31 AM

Confirm

You are about to create a new project.
Do you want to use the CodeWizardAVR?

Yes No

Select Chip



CodeVisionAVR - s2.prj - [E:\Program Files\Cv avr\bicycle\s2.c]

CodeWizardAVR - untitled.cwp

Chip: ATmega16

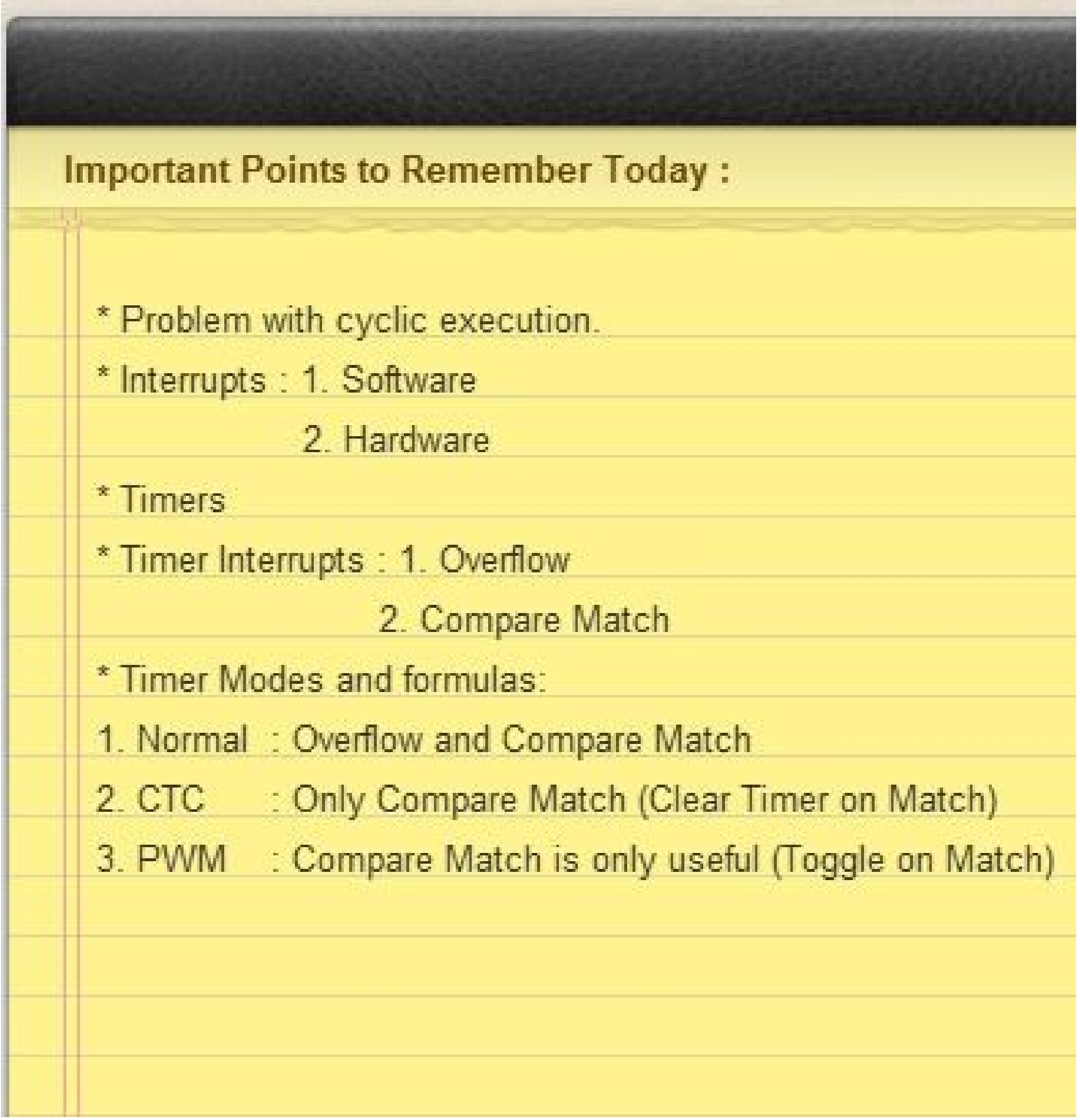
Clock: 8.000000 MHz

□ Check Reset Source

Program Type: Application

Using CVAVR

Demo



Important Points to Remember Today :

* Problem with cyclic execution.

* Interrupts : 1. Software
2. Hardware

* Timers

* Timer Interrupts : 1. Overflow
2. Compare Match

* Timer Modes and formulas:

1. Normal : Overflow and Compare Match
2. CTC : Only Compare Match (Clear Timer on Match)
3. PWM : Compare Match is only useful (Toggle on Match)

EC Electronics Club IIT Kanpur