

HOW TO PROGRAM MCU?

Rudra Pratap Suman

Flash Back (Takneek/Techkriti)

- ▶ Line Following Robots
- ▶ Wireless keyboards
- ▶ Wireless Gamepad
- ▶ Tachometer
- ▶ They were made using Microcontrollers

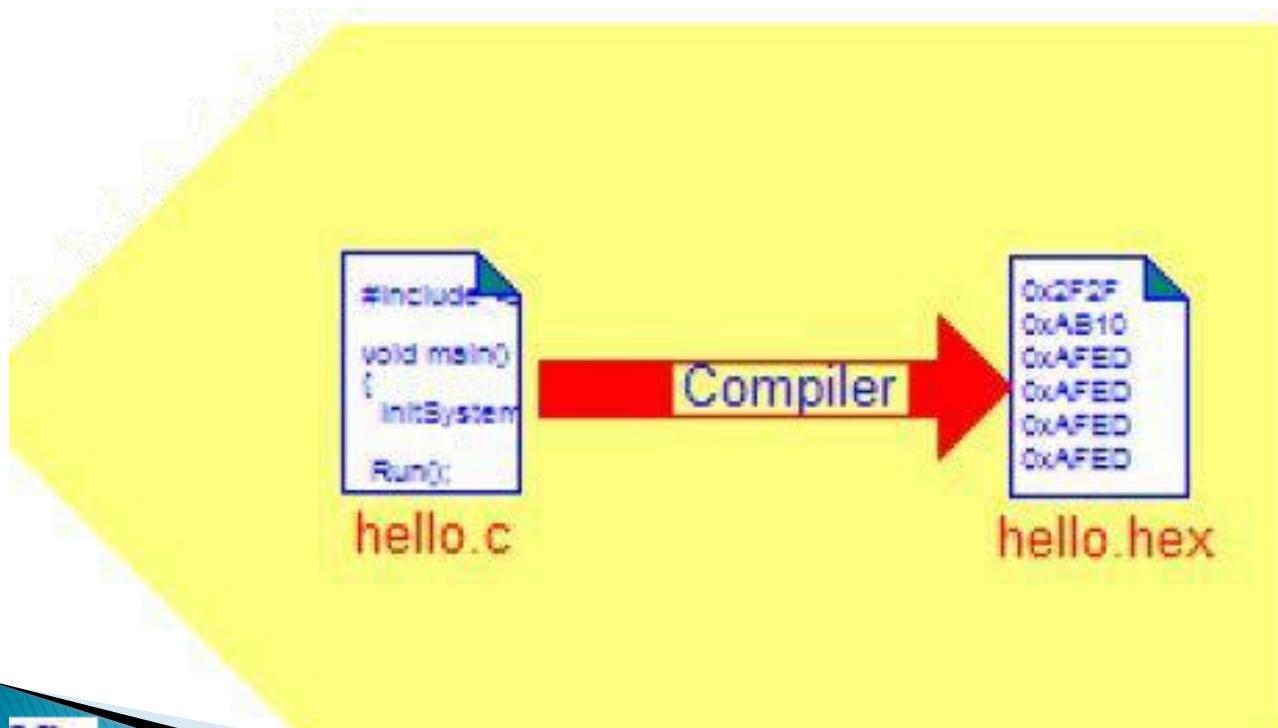
Tools Required -> CAVR



PC Running IDE for
entering, editing and compiling
source program.

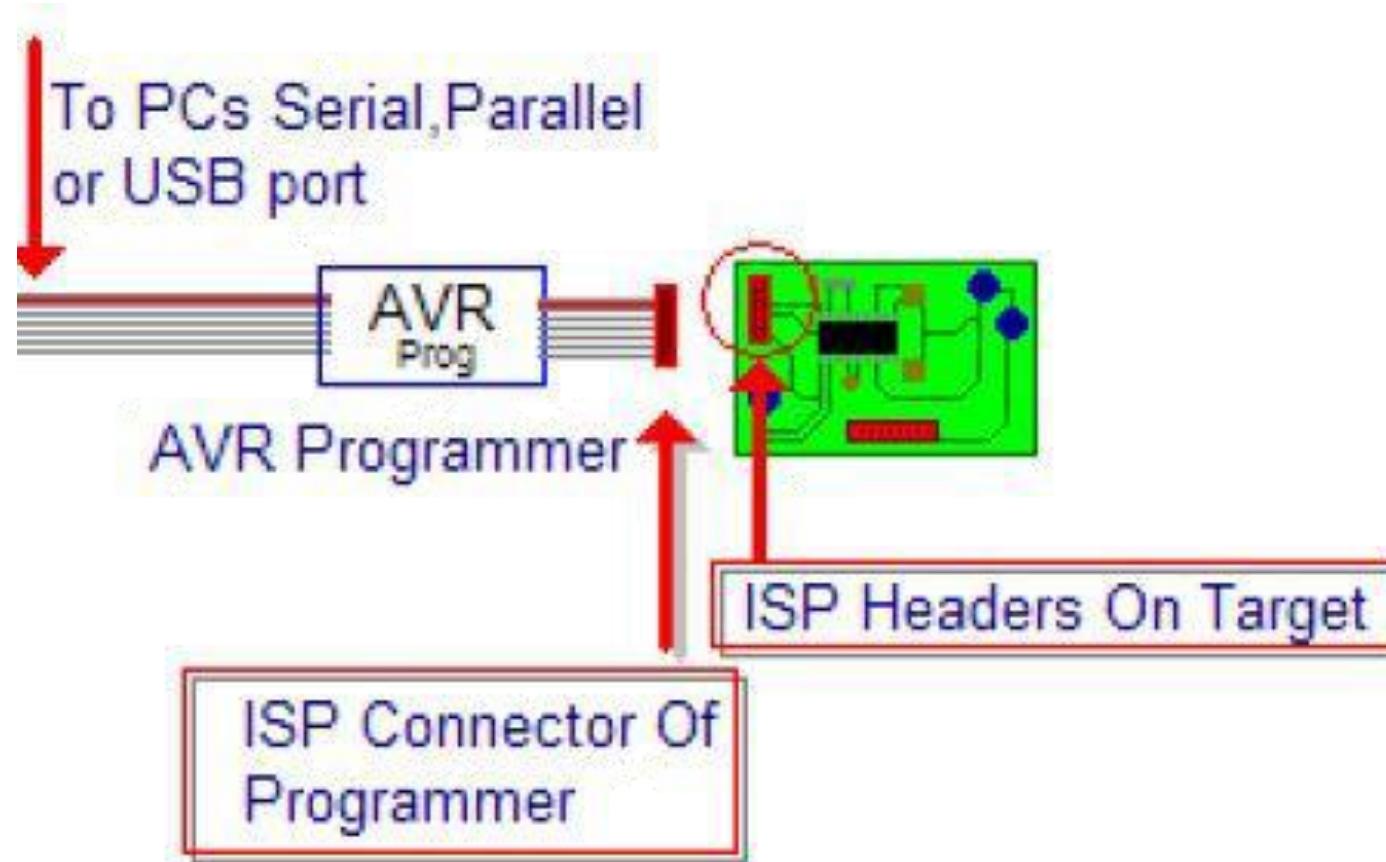
Compiler -> CVAVR

- The code is written in C language so we need to convert it into the format that Atmega understands



10

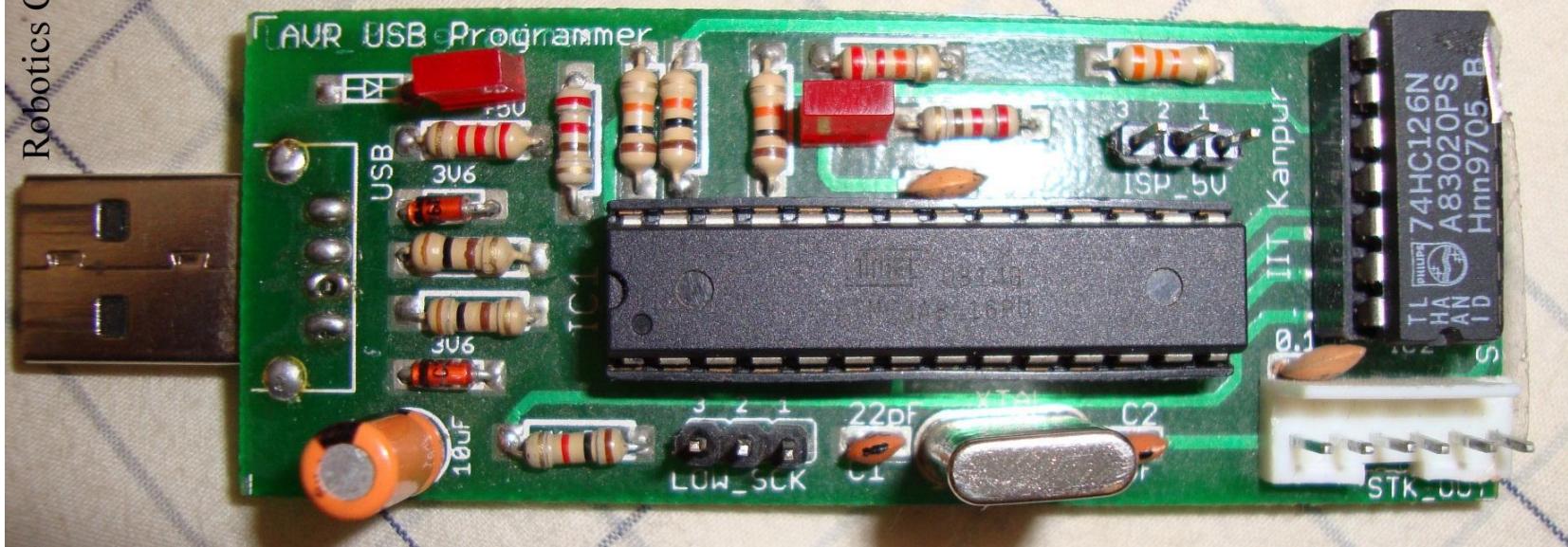
Transfer code to Atmega AVR Studio



Avr Programmer

Robotics Club, IIT Kanpur

USB based STK500 Programmer
for AVR microcontrollers



- So we need two softwares overall
 - a) CVAVR -> Editor and Compiler
 - b) Avr Studio -> Transfer Code to Atmega

The ATmega16

- ▶ 40 pin IC.
- ▶ 32 pins for I/O.
- ▶ 8 pins reserved.
- ▶ I/O pins divided into 4 groups of 8 pins, called ports.
- ▶ Ports labeled as A, B, C and D.

(XCK/T0)	PB0	1	40	PA0 (ADC0)
(T1)	PB1	2	39	PA1 (ADC1)
(INT2/AIN0)	PB2	3	38	PA2 (ADC2)
(OC0/AIN1)	PB3	4	37	PA3 (ADC3)
(SS)	PB4	5	36	PA4 (ADC4)
(MOSI)	PB5	6	35	PA5 (ADC5)
(MISO)	PB6	7	34	PA6 (ADC6)
(SCK)	PB7	8	33	PA7 (ADC7)
	RESET	9	32	AREF
	VCC	10	31	GND
	GND	11	30	AVCC
	XTAL2	12	29	PC7 (TOSC2)
	XTAL1	13	28	PC6 (TOSC1)
(RXD)	PD0	14	27	PC5 (TDI)
(TXD)	PD1	15	26	PC4 (TDO)
(INT0)	PD2	16	25	PC3 (TMS)
(INT1)	PD3	17	24	PC2 (TCK)
(OC1B)	PD4	18	23	PC1 (SDA)
(OC1A)	PD5	19	22	PC0 (SCL)
(ICP1)	PD6	20	21	PD7 (OC2)

Basics of C language

- ▶ If else block

- ▶ If(condition)

```
{
```

```
...
```

```
...
```

```
}
```

```
else
```

```
{
```

```
...
```

```
...
```

```
...
```

```
}
```

While & For

- ▶ While (conditon)

```
{
```

```
...
```

```
...
```

```
}
```

- ▶ for(initialisation; condition; increment)

```
{
```

```
...
```

```
...
```

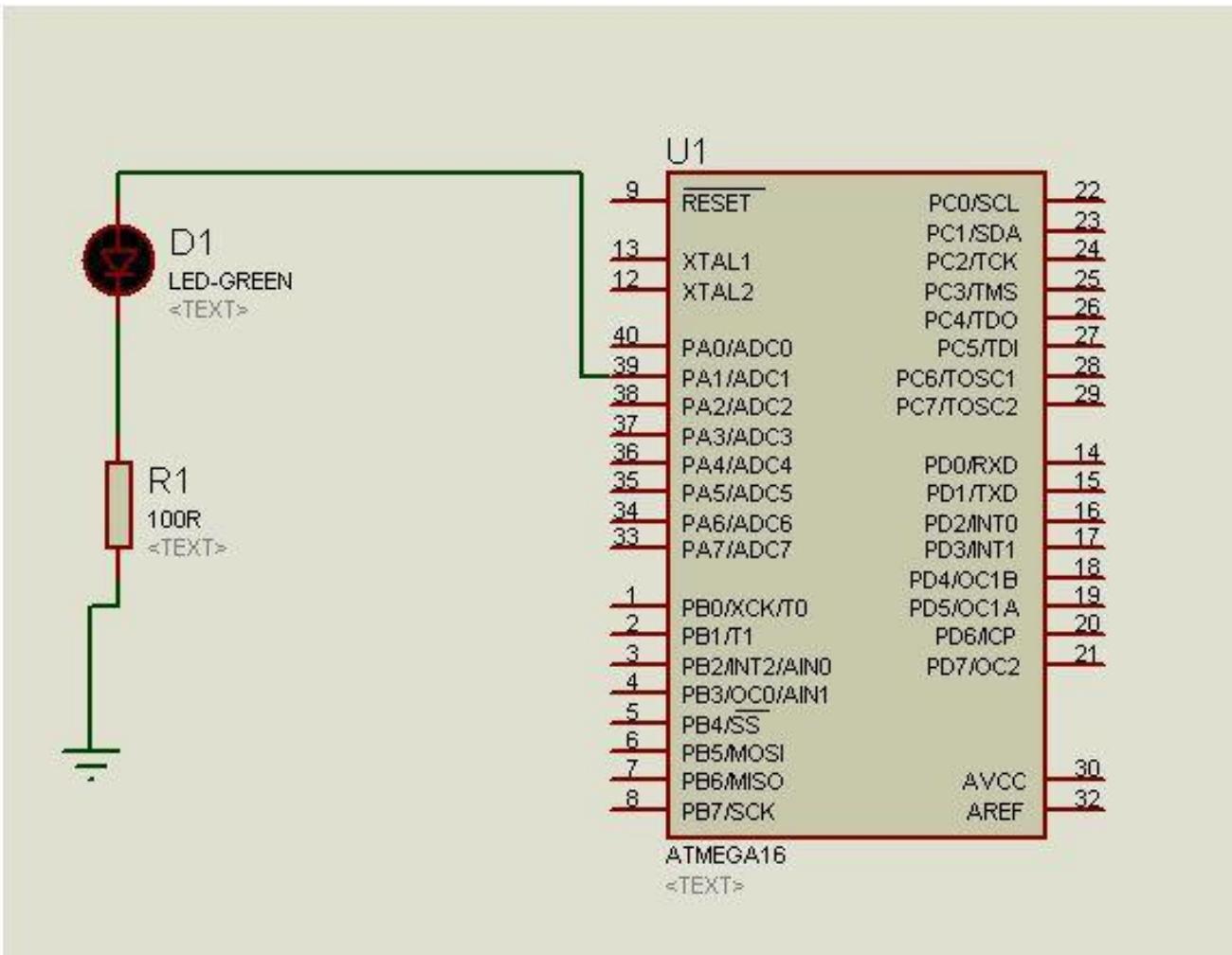
```
}
```

Some C operators

- ▶ | is bitwise OR. Eg. $10100111 | 11000101 = 11100111$
- ▶ & is bitwise AND. Eg. $10100111 \& 11000101 = 10000101$
- ▶ ~ is bitwise NOT. Eg. $\sim 10100110 = 01011001$
- ▶ << is shift left. >> is shift right.

Lets Begin by blinking a simple LED

Circuit Diagram



Getting Started with CAVR

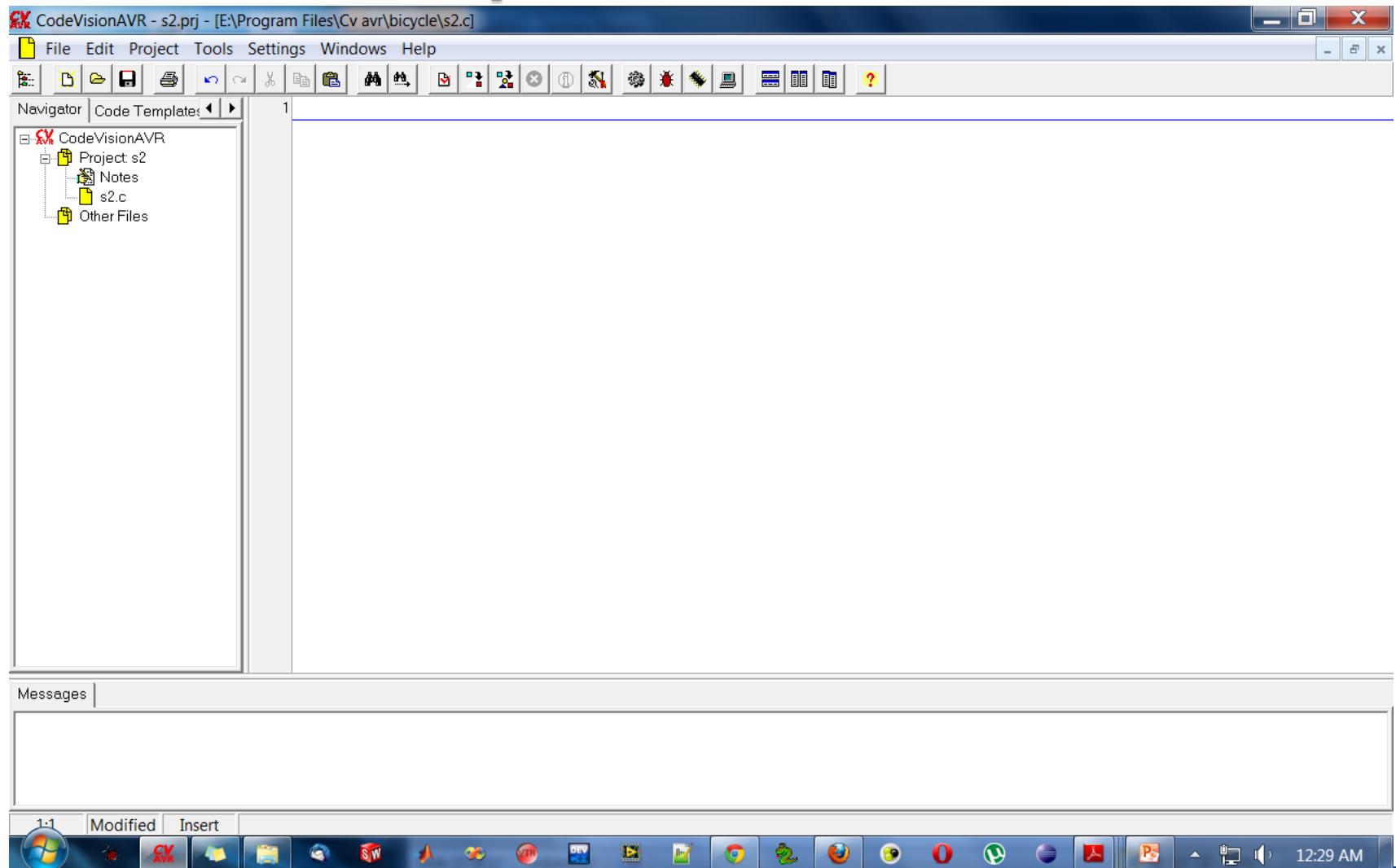
Open
CVAVR

Go to
File

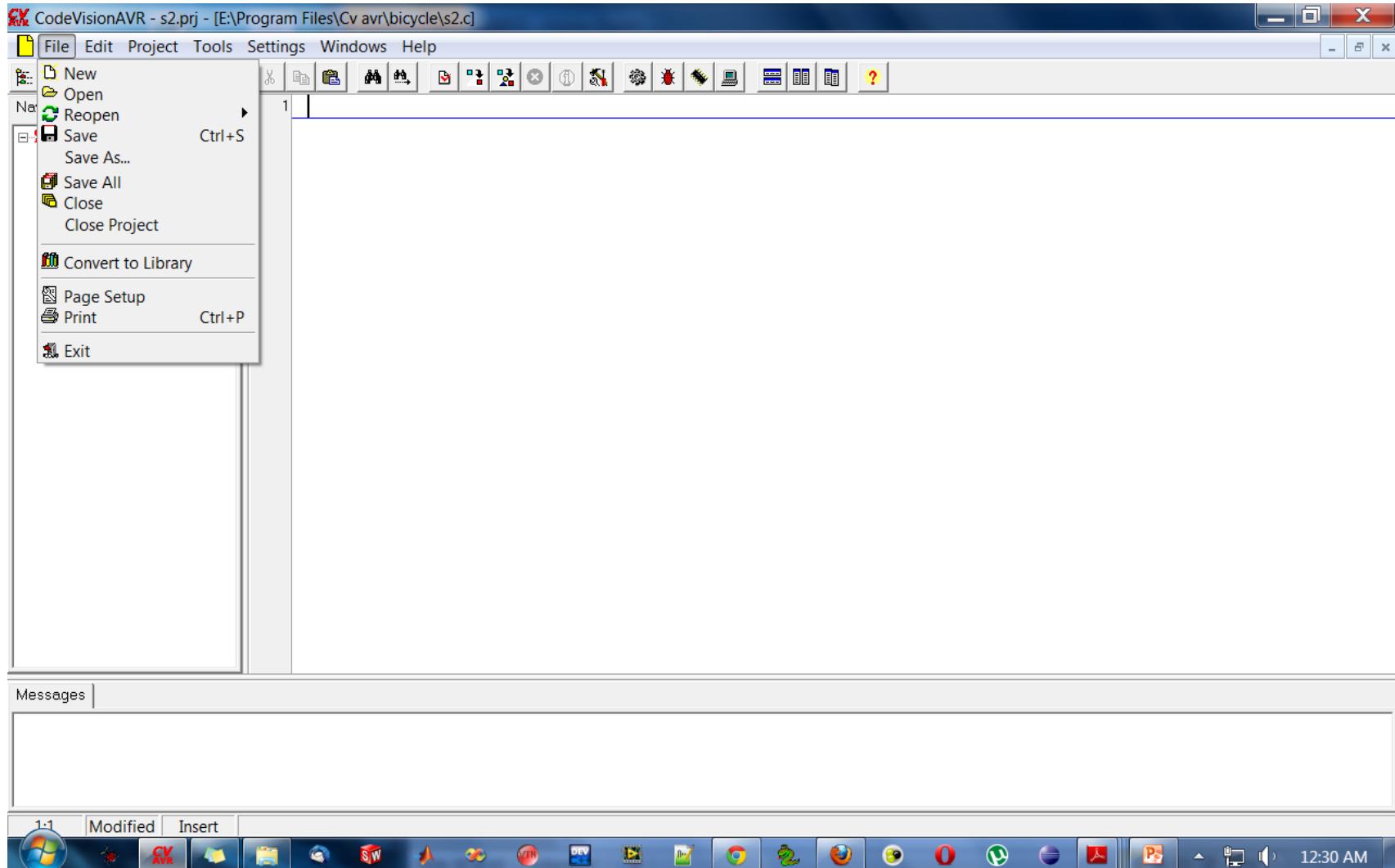
New

Project

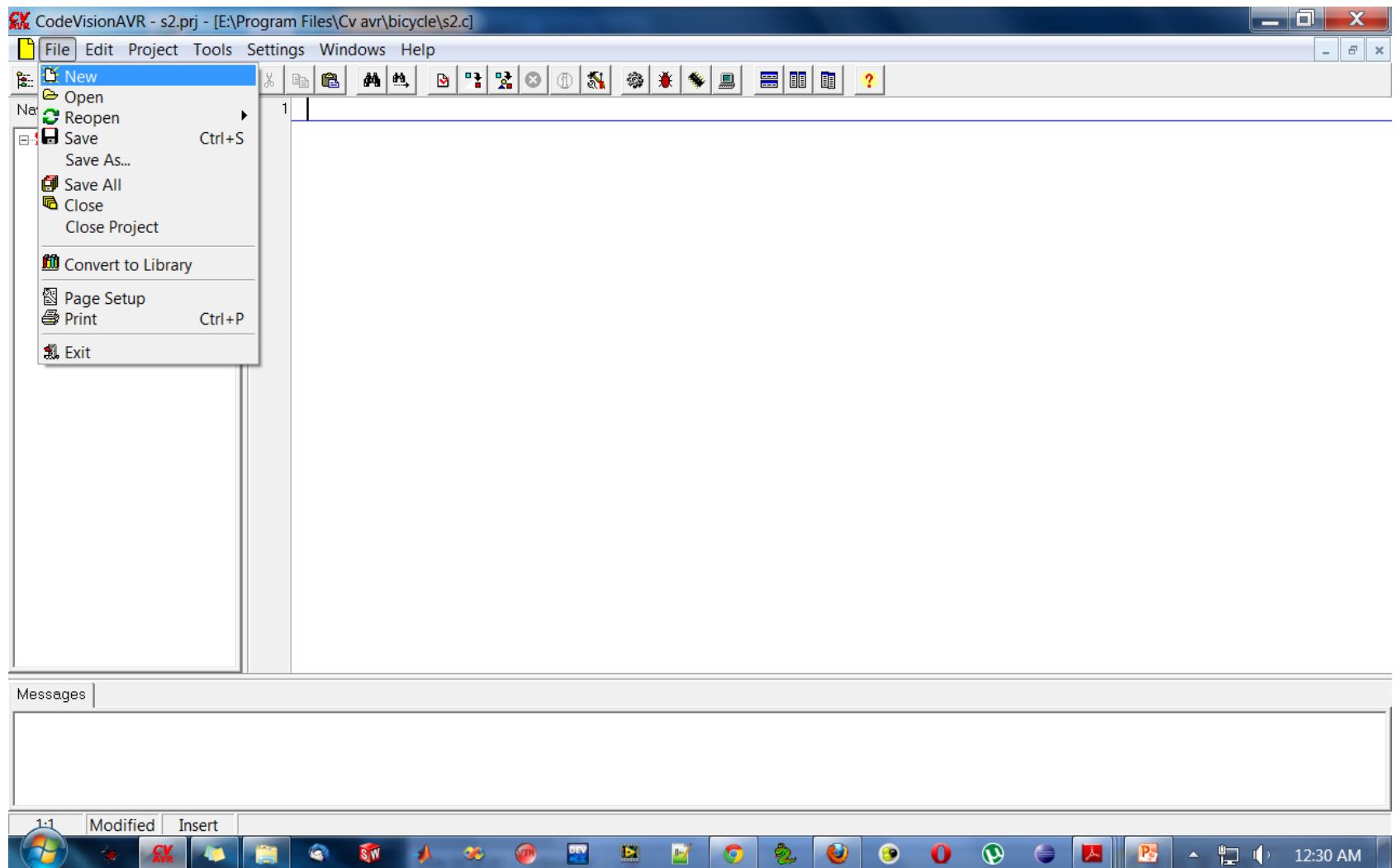
Open CVAVR



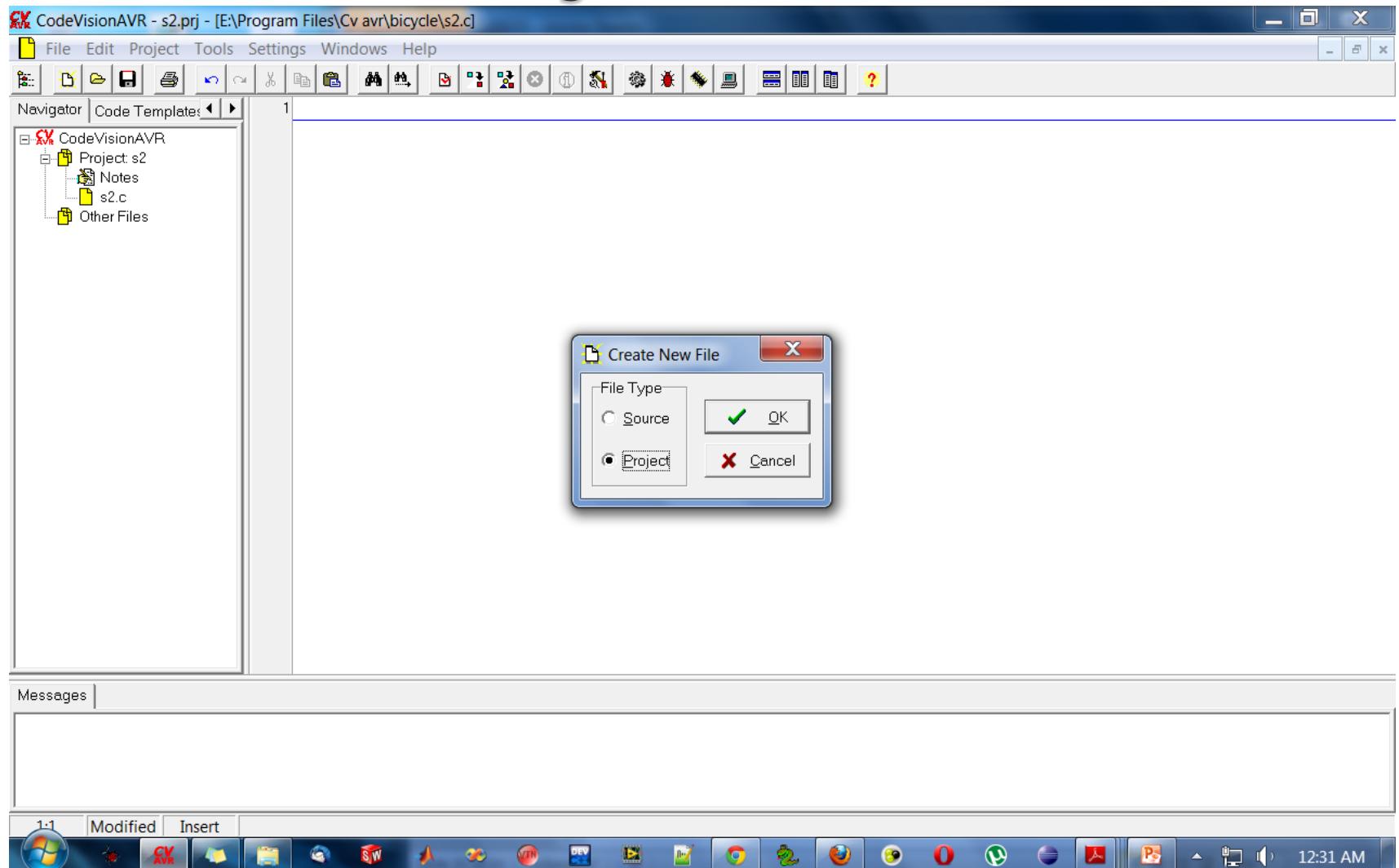
Go to File



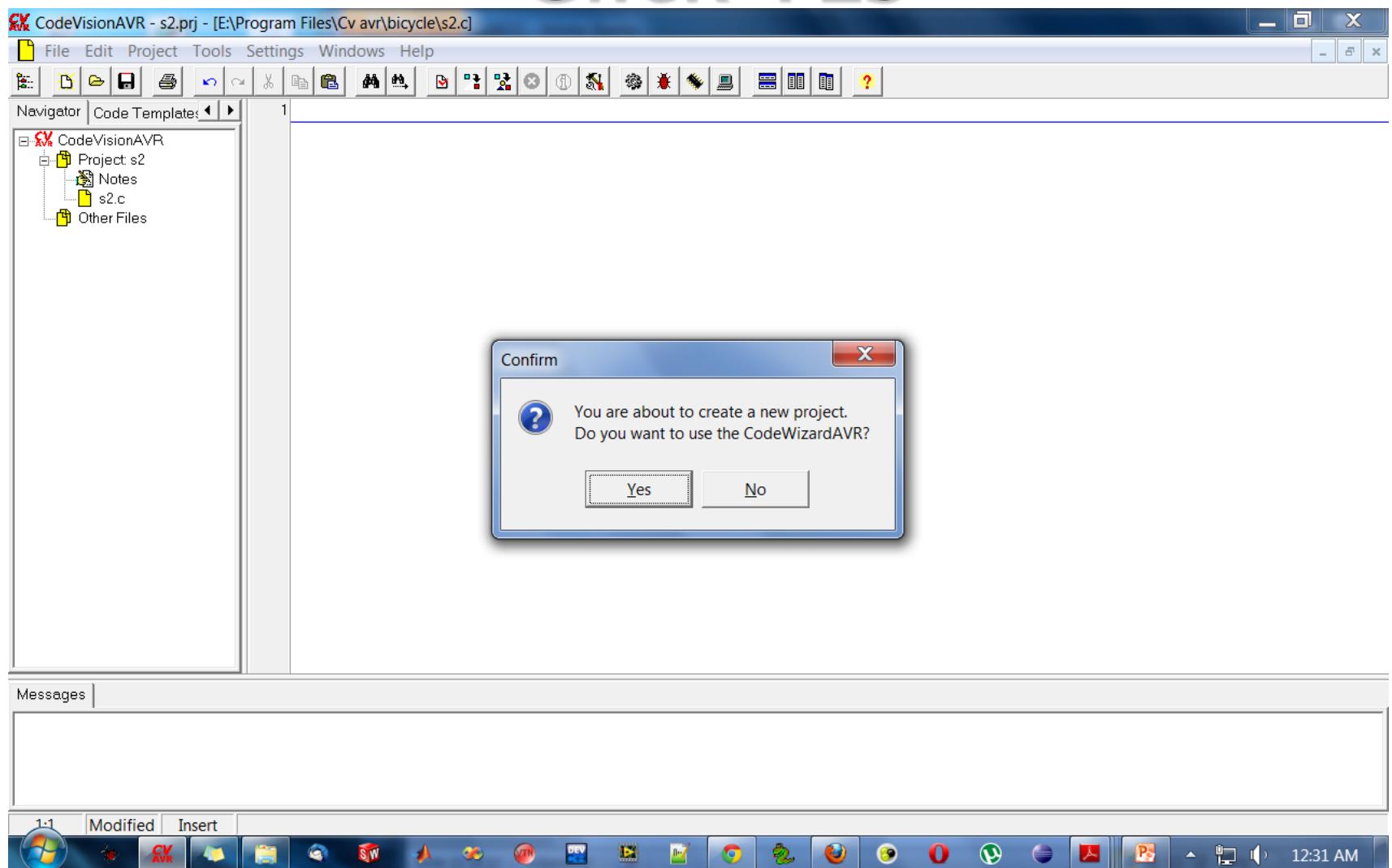
Click on New



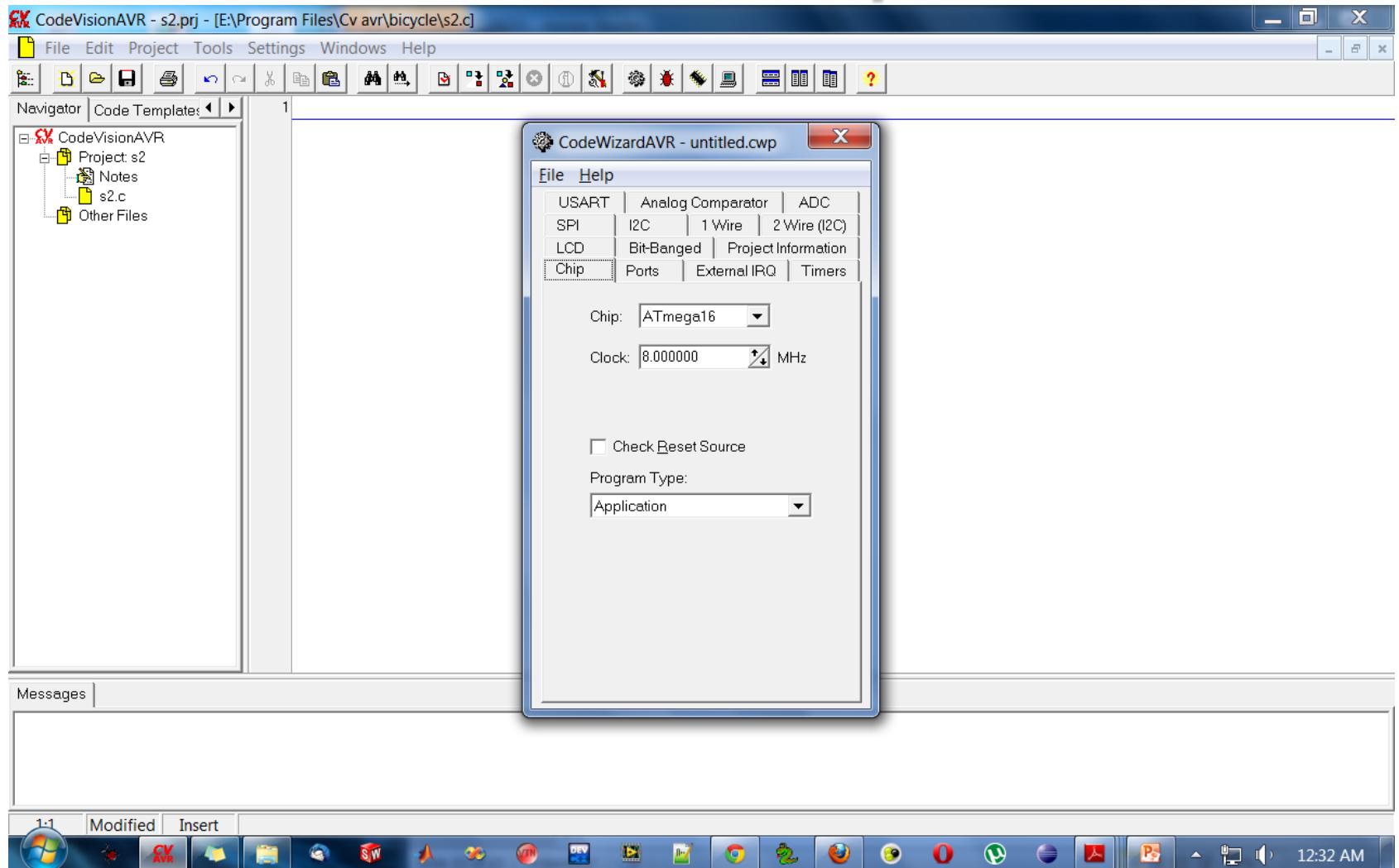
Select Project- > Click OK



Click YES



Select Chip

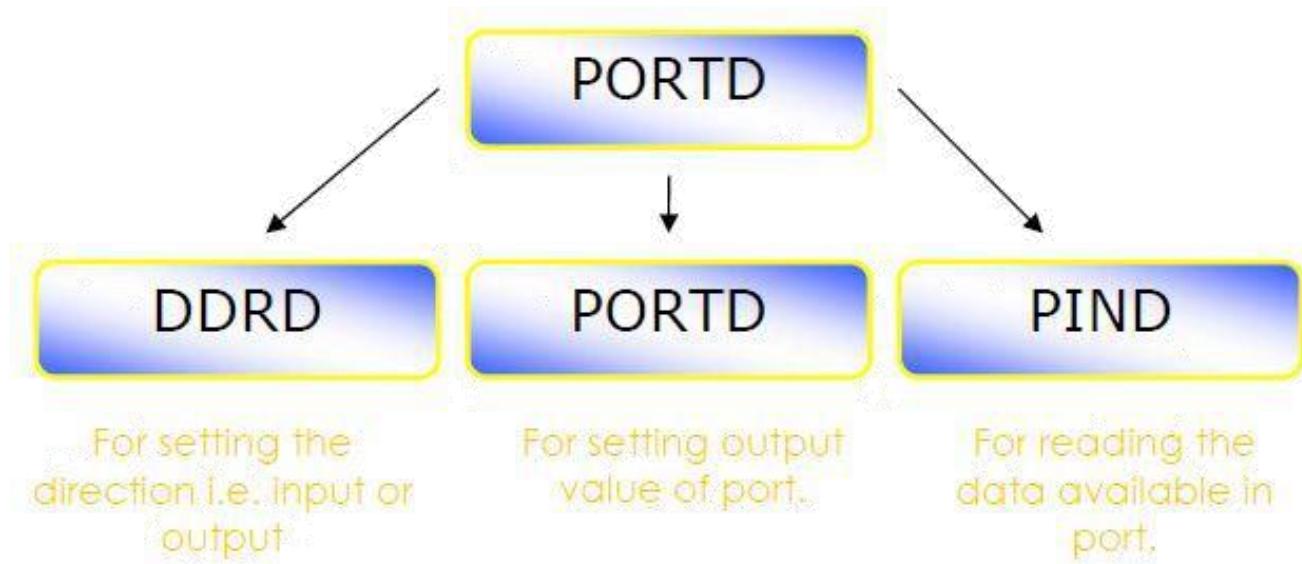


Introduction to I/O

- ▶ Atmega has total of 40 pins out of which 32 pins can be used as Input or Output
- ▶ These 32 pins are divided into 4 groups of 8 pins PORTA, PORTB , PORTC , PORTD

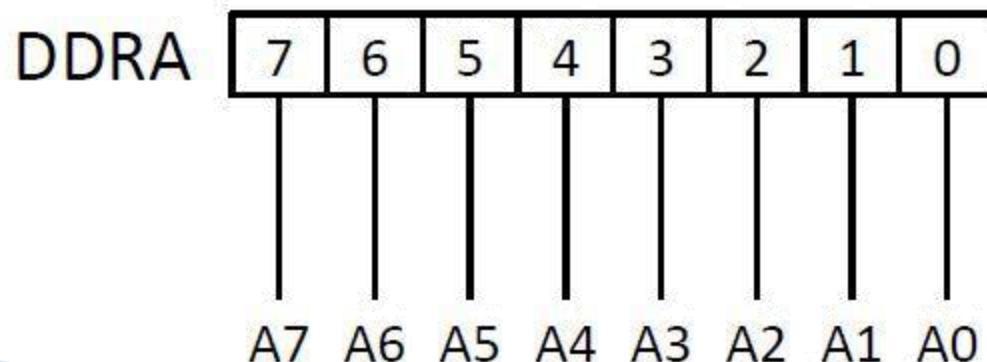
Accessing digital IO in C

Each PORT in AVR has three related Registers.



Data Direction register (DDR)

- ▶ This sets direction for all pins (32)
- ▶ Direction for these pins can be Input or Output
- ▶ To blink an LED we need to set pin as “OUTPUT” but “HOW“ ?
- ▶ `DDRA = 0b00000001 ;`
- ▶ `DDRA = 0x01 ;`
- ▶ 1 Stands for Output & 0 stands for Input



What Next ?

- ▶ We have set the Pin as Output
- ▶ What else do we need to light the LED ??
- ▶ Supply of 5 Volts !!! This is given by PORT Register

PORT Register

- ▶ Only after you have set the Pin to Output you can control them through this Register
- ▶ It is a 8 bit register . It corresponds to the pin in same manner as that of DDR Register
- ▶ Used to set output value (0 or 1) only if the corresponding Pin has been set as output by DDR Register
- ▶ PORTA= 0b 00000001; or
- ▶ PORTA= 0x01 ;
- ▶ 1 stands for 5V
- ▶ 0 stands for 0V



Setting I/O

Go to Ports

The screenshot shows two windows of the CodeVisionAVR IDE. The left window is the main editor for the file s2.c, displaying C code for an AVR microcontroller. The right window is the CodeWizardAVR interface, specifically the 'Ports' tab under the 'Chip' category, which allows configuring the digital pins of the AVR chip.

s2.c File Content:

```
198 SFIOR=0x00;
199
200 // LCD module i
201 lcd_init(16);
202
203 // Global enable
204 #asm("sei")
205     lcd_clear();
206     lcd_puts("Hello, World!");
207     delay_ms(1000);
208     lcd_clear();
209     while (1)
210     {
211
212
213
214
215
216
217
218     // Place your code here
219
220 }
221
222 }
```

CodeWizardAVR - Ports Configuration:

Port	Data Direction	Pullup/Output Value
Bit 0	In	T Bit 0
Bit 1	In	T Bit 1
Bit 2	In	T Bit 2
Bit 3	In	T Bit 3
Bit 4	In	T Bit 4
Bit 5	In	T Bit 5
Bit 6	In	T Bit 6
Bit 7	In	T Bit 7

- Click on In to make that pin Output
- Can do so for all four ports

The screenshot shows the CodeVisionAVR IDE interface. On the left is the Navigator pane, which displays the project structure: CodeVisionAVR > Project: s2 > s2.c. The main area is a code editor with the following C code:

```

198 SFIOR=0x00;
199
200 // LCD module initia
201 lcd_init(16);
202
203 // Global enable int
204 #asm("sei")
205     lcd_clear();
206     lcd_putsf("re
207     delay_ms(1000)
208     lcd_clear();
209 while (1)
210 {
211
212
213
214
215
216
217
218     // Place your
219
220 }
221
222

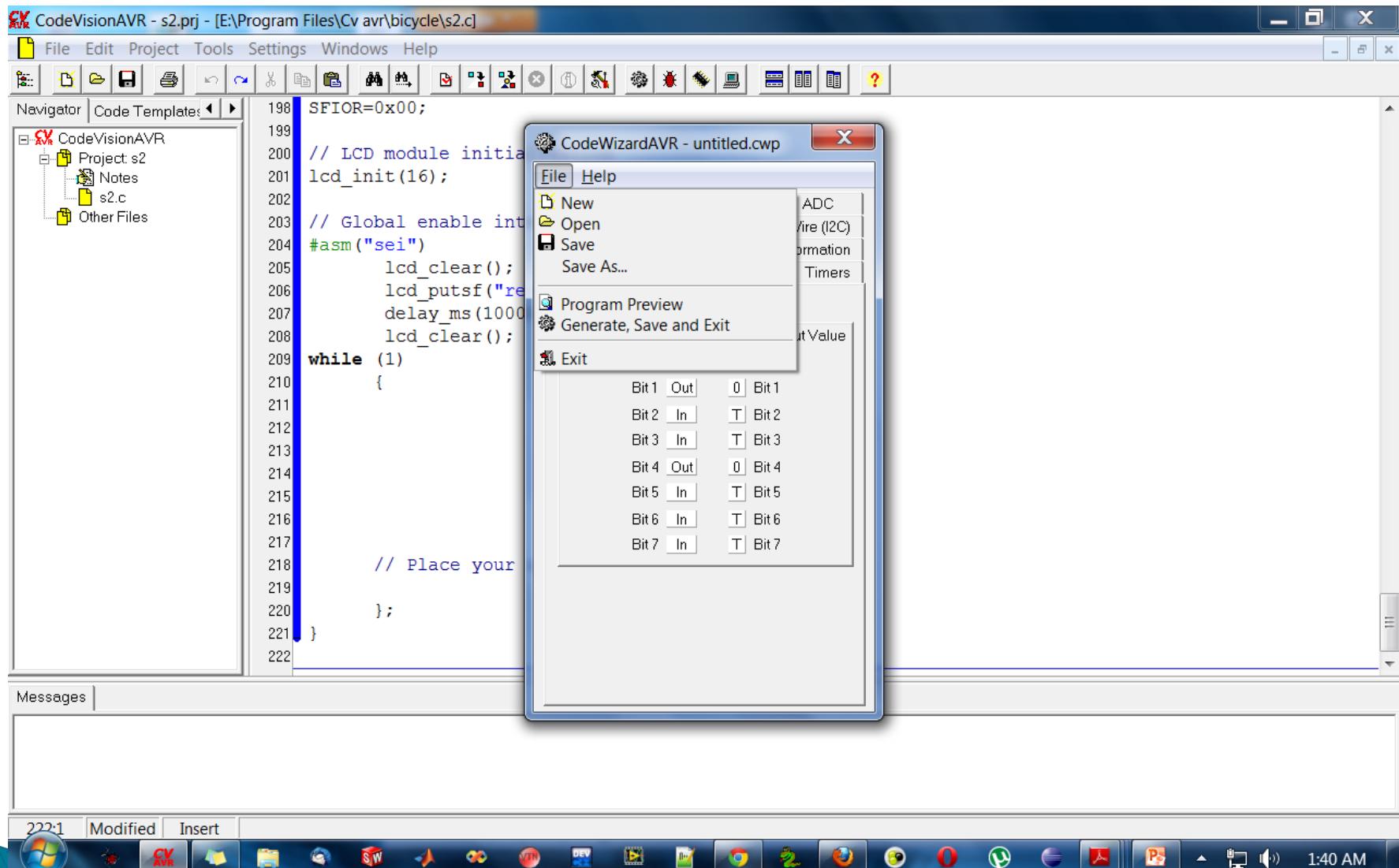
```

A blue vertical bar highlights the line "Bit 0 In". A modal dialog box titled "CodeWizardAVR - untitled.cwp" is open in the center. It has tabs for USART, Analog Comparator, ADC, SPI, I2C, 1 Wire, 2 Wire (I2C), LCD, Bit-Banged, Project Information, Chip, Ports, External IRQ, and Timers. The "Ports" tab is selected. Below it are tabs for Port A, Port B, Port C, and Port D. The table shows the current port settings:

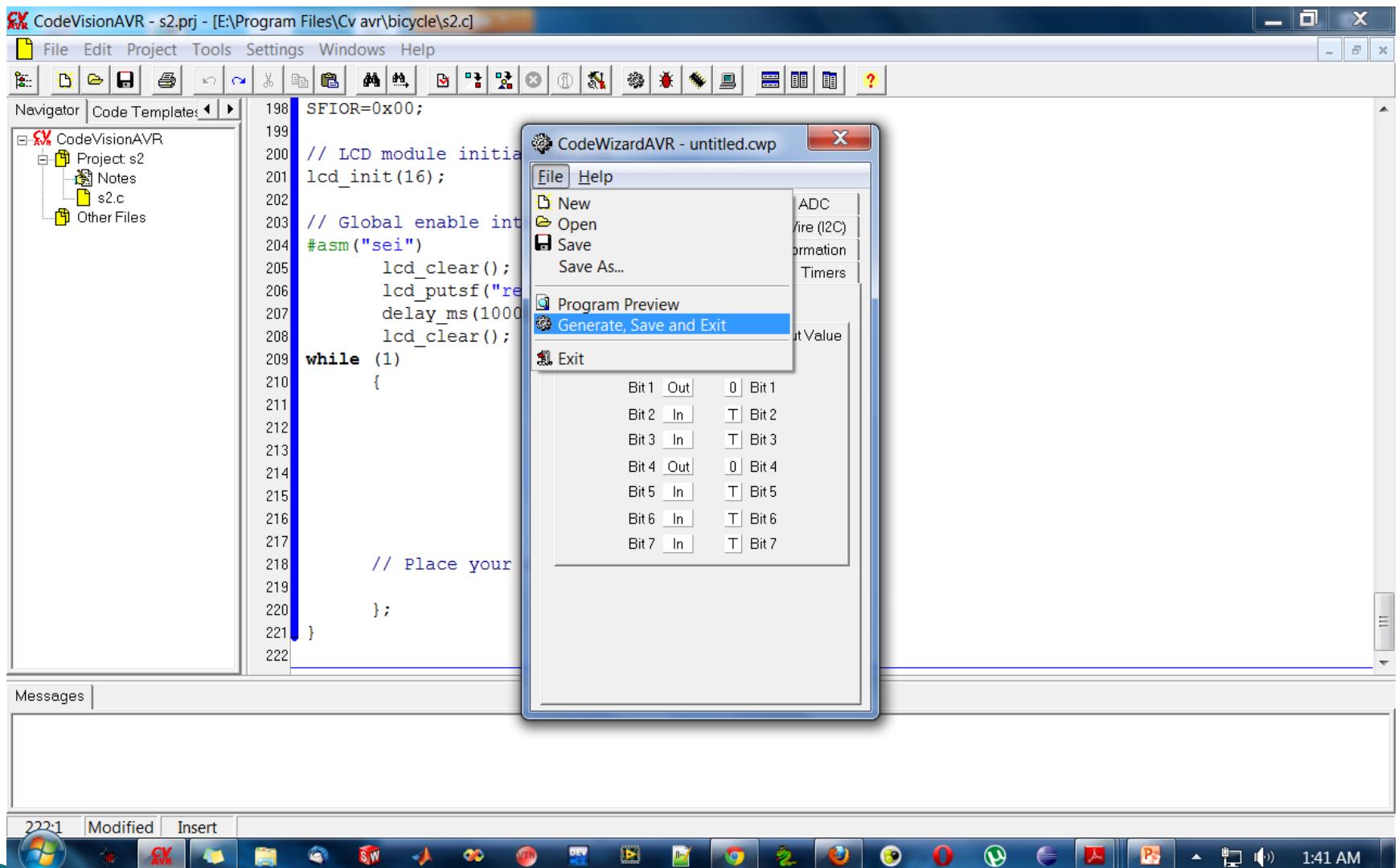
	Data Direction	Pullup/Output Value
Bit 0	In	T Bit 0
Bit 1	Out	0 Bit 1
Bit 2	In	T Bit 2
Bit 3	In	T Bit 3
Bit 4	Out	0 Bit 4
Bit 5	In	T Bit 5
Bit 6	In	T Bit 6
Bit 7	In	T Bit 7

The status bar at the bottom shows "222:1 Modified Insert".

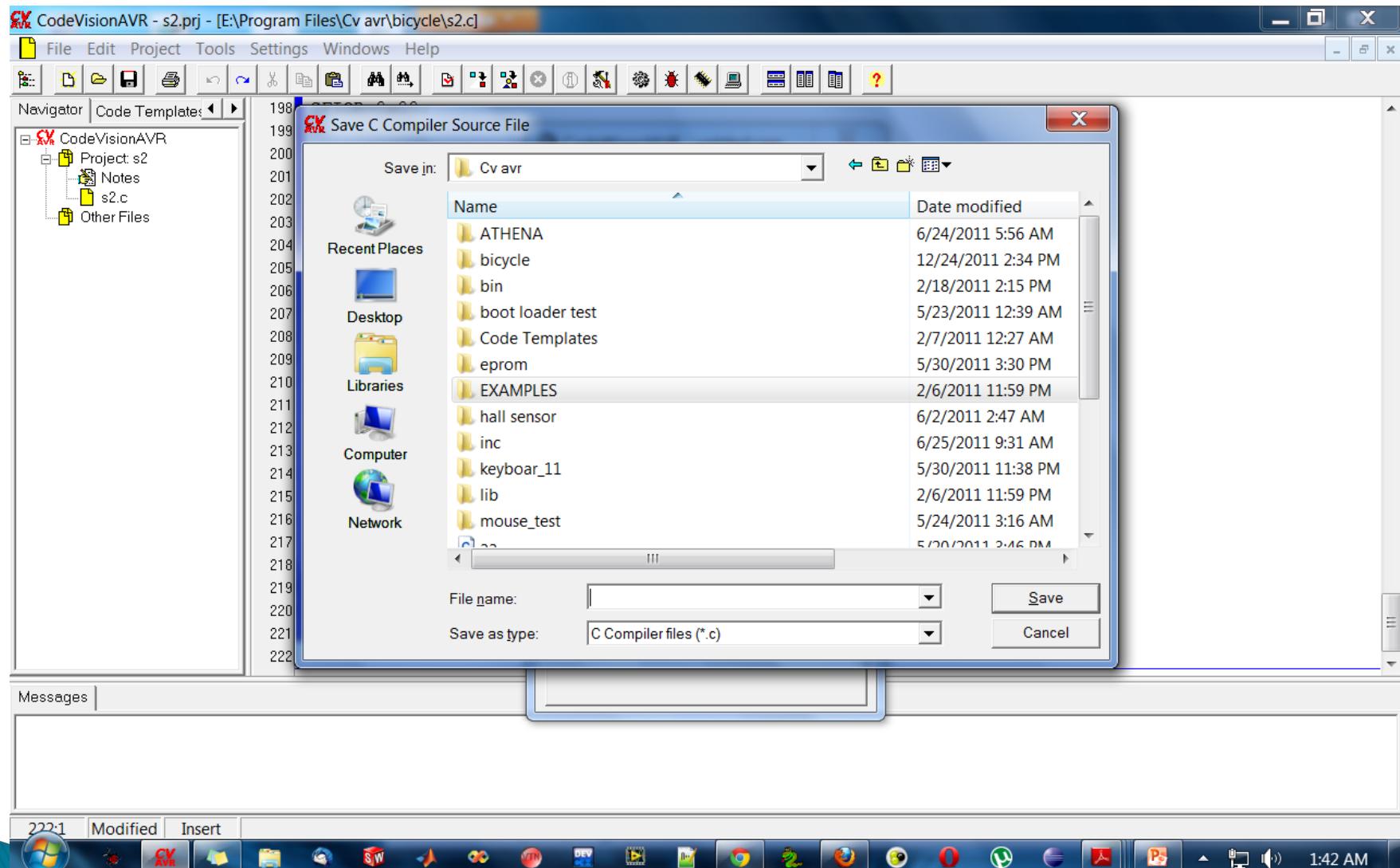
Click on File



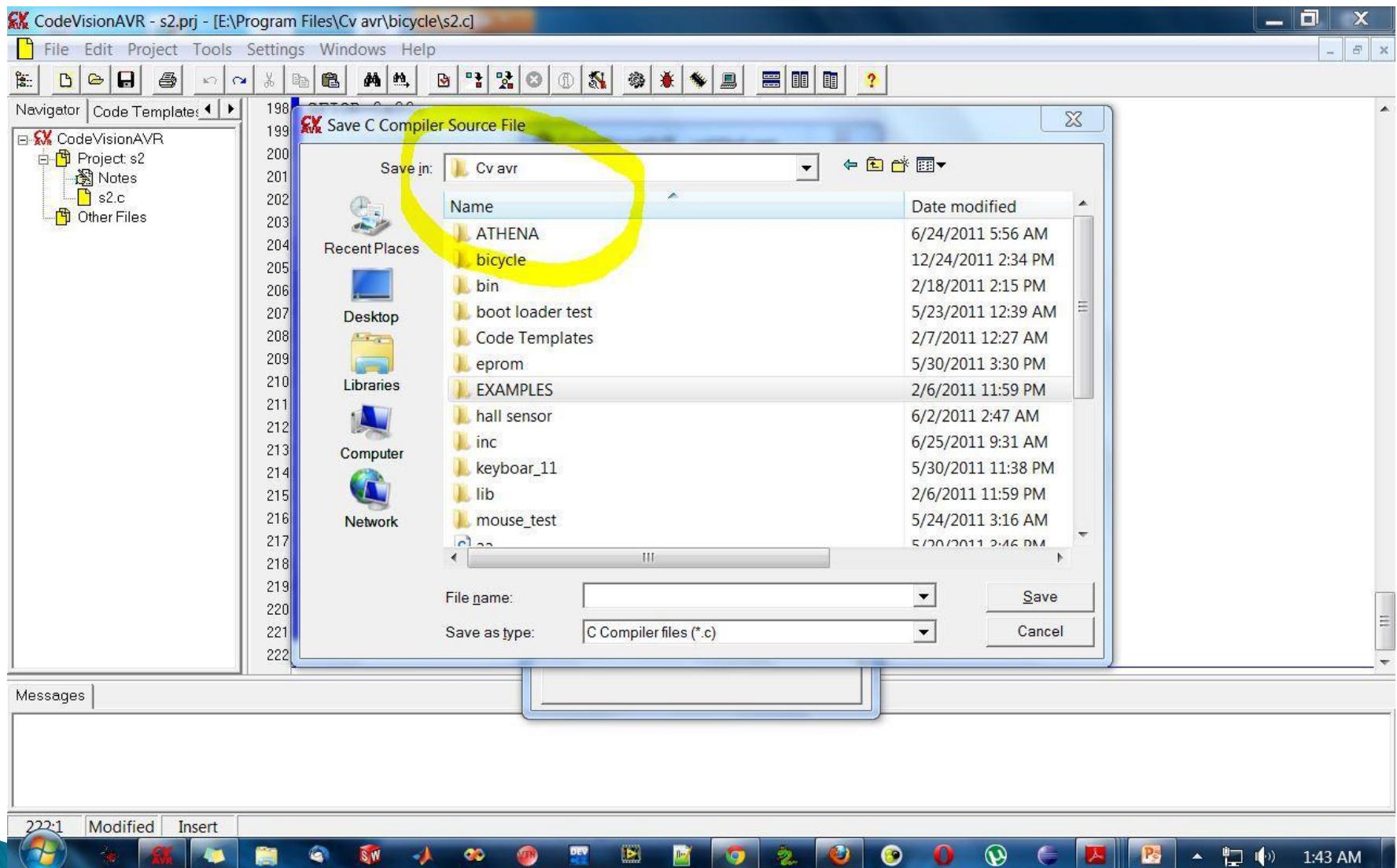
Generate Save and Exit



Enter name (3 times)

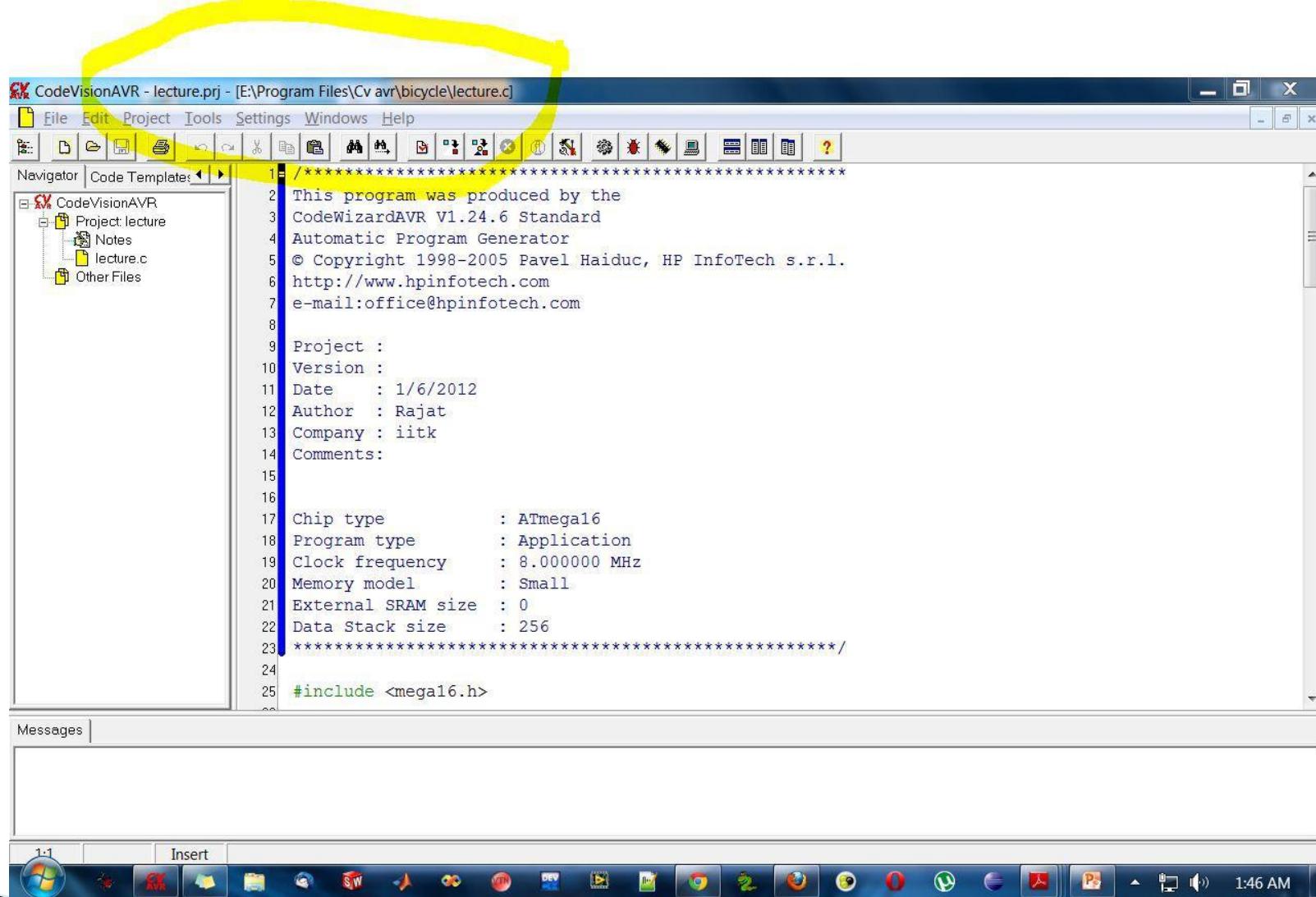


Where is the code stored ?



Then Click Save

Name of Project & Location



The screenshot shows the CodeVisionAVR software interface. The title bar reads "CodeVisionAVR - lecture.prj - [E:\Program Files\CV avr\bicycle\lecture.c]". The menu bar includes File, Edit, Project, Tools, Settings, Windows, and Help. A toolbar with various icons is located above the main window. On the left is a Navigator pane showing a project structure: "CodeVisionAVR" expanded, "Project: lecture" selected, containing "Notes", "lecture.c", and "Other Files". The main code editor window displays the following C code:

```
1 //*****
2 This program was produced by the
3 CodeWizardAVR V1.24.6 Standard
4 Automatic Program Generator
5 © Copyright 1998-2005 Pavel Haiduc, HP InfoTech s.r.l.
6 http://www.hpinfotech.com
7 e-mail:office@hpinfotech.com
8
9 Project :
10 Version :
11 Date : 1/6/2012
12 Author : Rajat
13 Company : iitk
14 Comments:
15
16
17 Chip type      : ATmega16
18 Program type   : Application
19 Clock frequency : 8.000000 MHz
20 Memory model   : Small
21 External SRAM size : 0
22 Data Stack size : 256
23 *****/
24
25 #include <mega16.h>
```

The status bar at the bottom shows "1-1", "Insert", and the system tray with various icons. The date and time are shown as "1:46 AM".

Writing the Code

- ▶ NOTE : We write our code in While block

- ▶ While (1)

{

```
PORTA.1=1; // sets the Pin to 5 volts
```

```
PORTA.1=0; // sets the Pin to 0 volts
```

}

- ▶ This makes the LED to blink but we cannot see blinking !!!

- ▶ This is because Atmega runs at a frequency of 8000000 Hz
- ▶ We need to introduce delay so as to see blinking
- ▶ Use header file delay.h
- ▶ Function to be used • delay_ms(time in millis);
- ▶ While (1) {
delay_ms(1000);
PORTA.1=1;
delay_ms(1000);
PORTA.1=0;
}

- ▶ This is because Atmega runs at a frequency of 8000000 Hz
- ▶ We need to introduce delay so as to see blinking
- ▶ Use header file delay.h
- ▶ Function to be used • delay_ms(time in millis);
- ▶ While (1) {
delay_ms(1000);
PORTA.1=1;
delay_ms(1000);
PORTA.1=0;
}

How to compile

- ▶ Code is written in C language but Atmega understands Hex file
- ▶ so we need to convert the C file to Hex file

Compiling

CodeVisionAVR - lecture.prj - [E:\Program Files\CV avr\bicycle\lecture.c]

File Edit Project Tools Settings Windows Help

Navigator | Code Templates < >

CodeVisionAVR
Project: lecture
Notes
lecture.c
Other Files

```
1 //*****  
2 This program was compiled by the  
3 CodeWizardAVR V1.24.6 Standard  
4 Automatic Program Generator  
5 © Copyright 1998-2005 Pavel Haiduc, HP InfoTech s.r.l.  
6 http://www.hpinfotech.com  
7 e-mail:office@hpinfotech.com  
8  
9 Project :  
10 Version :  
11 Date : 1/6/2012  
12 Author : Rajat  
13 Company : iitk  
14 Comments:  
15  
16  
17 Chip type : ATmega16  
18 Program type : Application  
19 Clock frequency : 8.000000 MHz  
20 Memory model : Small  
21 External SRAM size : 0  
22 Data Stack size : 256  
23 *****/  
24  
25 #include <mega16.h>  
26
```

Messages

2:18 Insert AVR SW DEV DVFS Firefox 12:37 PM

Make the Project

The screenshot shows the CodeVisionAVR IDE interface. The title bar reads "CodeVisionAVR - lecture.prj - [E:\Program Files\CV avr\bicycle\lecture.c]". The menu bar includes File, Edit, Project, Tools, Settings, Windows, and Help. The toolbar contains various icons for file operations like Open, Save, Print, and Build. A large arrow points from the text "Make the project" to the toolbar icon for building the project. The left sidebar shows the project structure: "CodeVisionAVR" with "Project: lecture" expanded, containing "Notes" and "lecture.c", and "Other Files". The main code editor window displays the "lecture.c" file with the following content:

```
1: ****
2: This program was produced by
3: CodeWizardAVR v1.24.6 Standard
4: Automatic Program Generator
5: © Copyright 1998-2005 Pavel Haiduc, HP InfoTech s.r.l.
6: http://www.hpinfotech.com
7: e-mail:office@hpinfotech.com
8:
9: Project :
10: Version :
11: Date : 1/6/2012
12: Author : Rajat
13: Company : iitk
14: Comments:
15:
16:
17: Chip type : ATmega16
18: Program type : Application
19: Clock frequency : 8.000000 MHz
20: Memory model : Small
21: External SRAM size : 0
22: Data Stack size : 256
23: ****
24:
25: #include <mega16.h>
```

The status bar at the bottom shows "7-4" and "Insert" mode, along with standard Windows system icons.

Check for errors

CodeVisionAVR - lecture.pj - [E:\Program Files\CV avr\bicycle\lecture.c]

File Edit Project Tools Settings Windows Help

Navigator Code Template: ▲ ▼

CodeVisionAVR
Project: lecture
Notes
lecture.c
Included Files
Global Variables
Functions
Other Files

```
103 // INT1: Off
104 // INT2: Off
105 MCUCR=0x00;
106 MCUCSR=0x00;
107
108 // Timer(s) /Counter(s)
109 TIMSK=0x00;
110
111 // Analog Comparator i
112 // Analog Comparator:
113 // Analog Comparator I
114 ACSR=0x80;
115 SFIOR=0x00;
116
117 while (1)
118 {
119     delay_ms(1000);
120     PORTA.1=1;
121     delay_ms(1000);
122     PORTA.1=0;
123     // Place your co
124
125 }
126
127
```

Information

Compiler | Assembler

Chip: ATmega16
Program type: Application
Memory model: Small
Optimize for: Size
(s)printf features: int, width
(s)scanf features: int, width
Promote char to int: No
char is unsigned: Yes
8 bit enums: Yes
Enhanced core instructions: On
Automatic register allocation: On

247 line(s) compiled
No errors
No warnings

Bit variables size: 0 byte(s)

Data Stack area: 60h to 15Fh
Data Stack size: 256 byte(s)
Estimated Data Stack usage: 0 byte(s)

Global variables size: 0 byte(s)

Hardware Stack area: 160h to 45Fh
Hardware Stack size: 768 byte(s)

Heap size: 0 byte(s)

EEPROM usage: 0 byte(s) (0.0% of EEPROM)
Program size: 150 words (1.8% of FLASH)

OK

Messages

122-16 Insert

12:47 PM

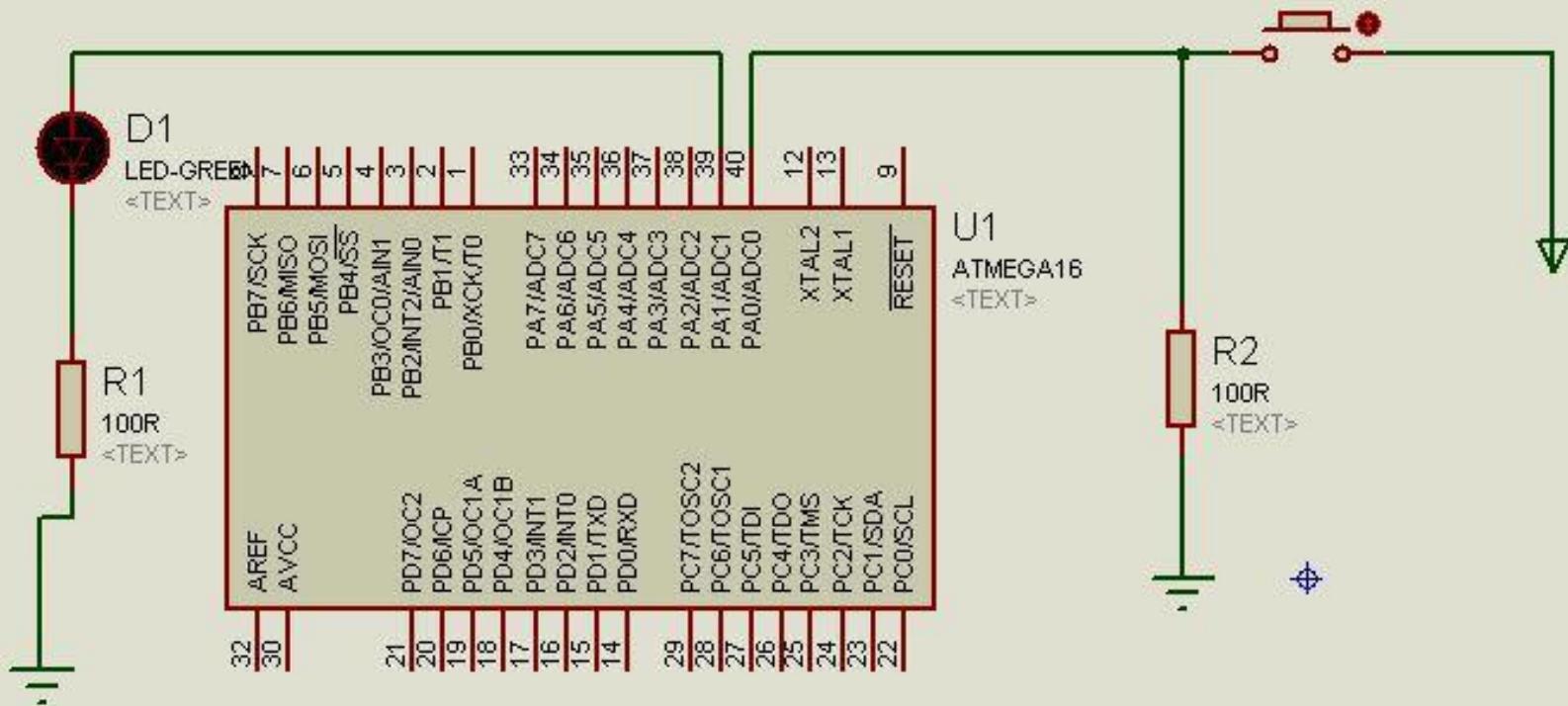
Hex File

- ▶ You can find the Hex file in Bin folder or the EXE folder of the directory where You installed CVAVR
- ▶ So we Have our Code ready
- ▶ Feed this code to Atmega using Programmer
- ▶ Lets see the code in action

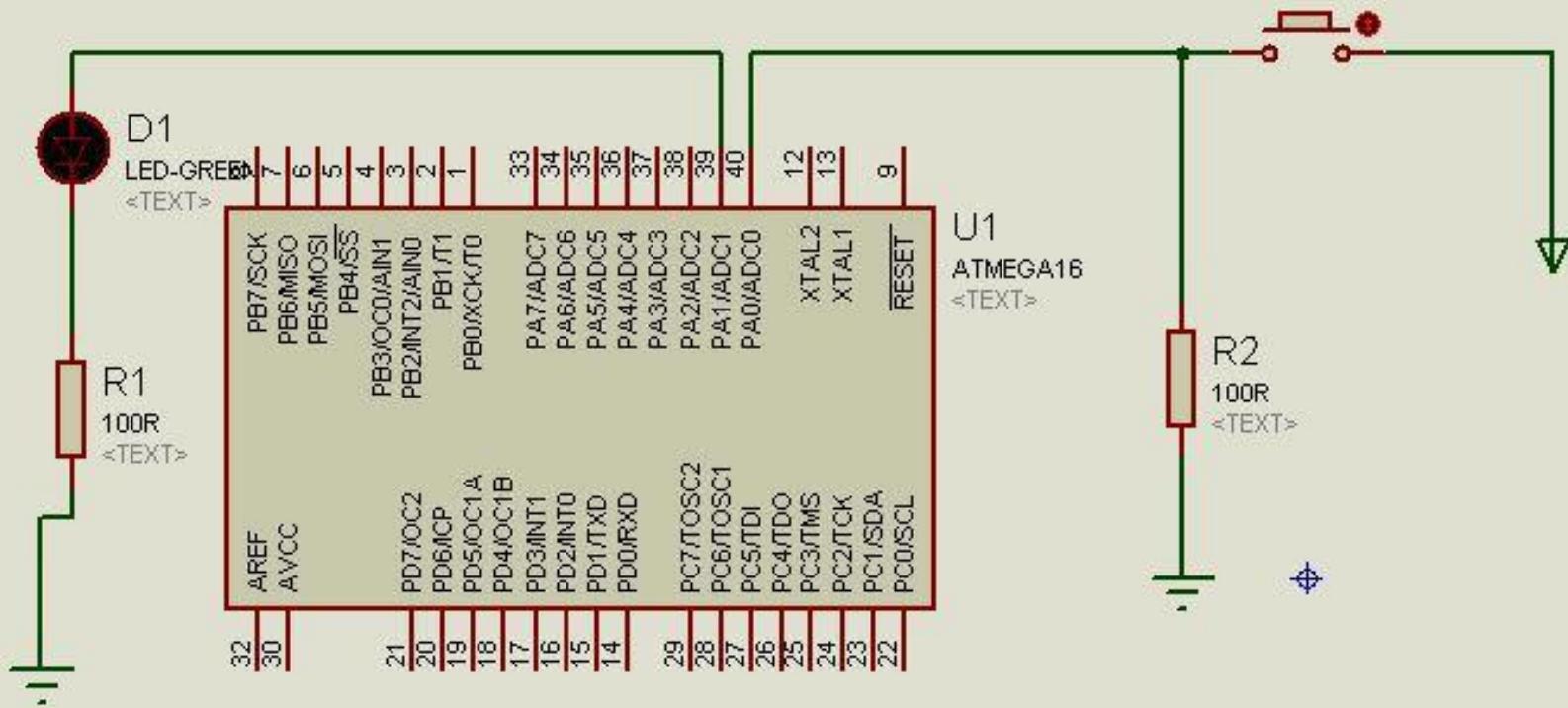
Lets add an Input

- ▶ Most Common Input • Button
- ▶ Since we have already made A0 as Input we connect a button to that pin
- ▶ If button is pressed light the LED else turn it off
- ▶ First draw the Circuit Diagram

Circuit Diagram



Circuit Diagram



PIN Register

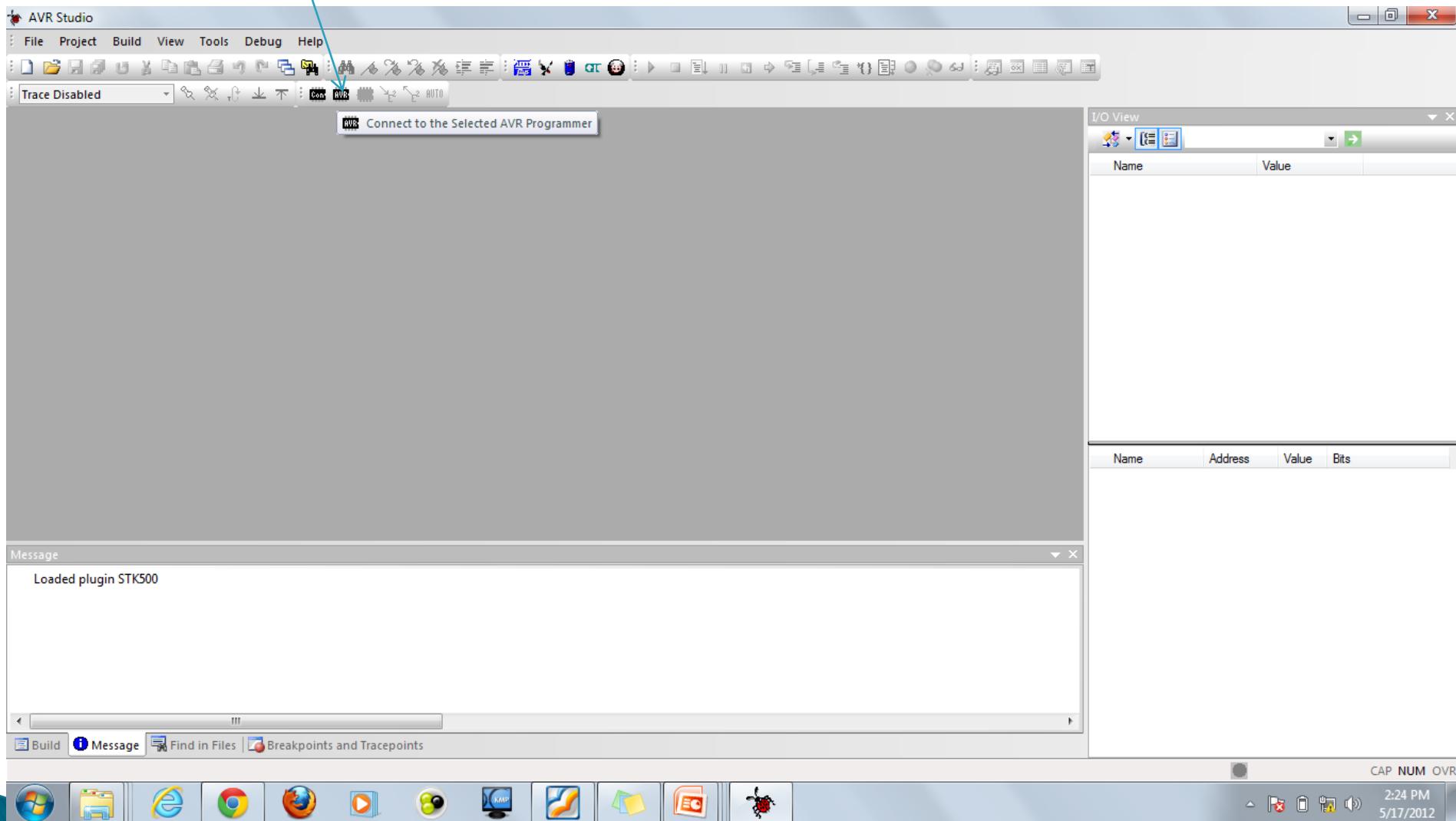
- ▶ It is a 8 bit register . It corresponds to the pin in same manner as that of DDR Register
- ▶ It is used to read voltage at a pin
- ▶ To be used only after the pin has been set as input by DDR register

Using Pin Register

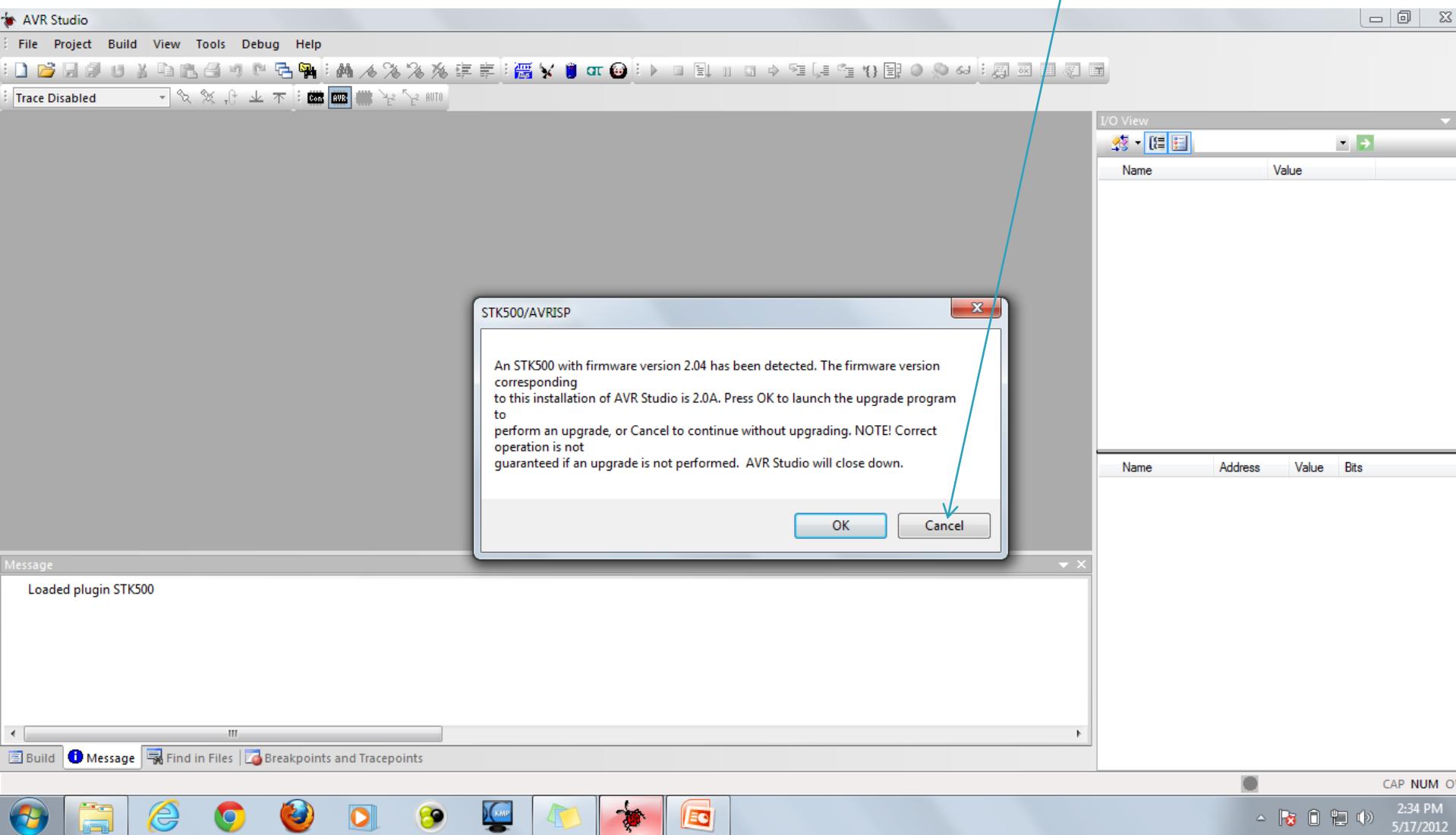
- ▶ int a; // Define the variable a to store the value of voltage
- ▶ a=PINA.0; // read value at pin A.0 (make sure it is input)
- ▶ If (a==1) // if voltage is 5V {
- ▶ PORTA.1=1; // Light the LED
- ▶ }
- ▶ else
- ▶ {
- ▶ PORTA.1=0; // Turn off the LED
- ▶ }

AVR Studio4

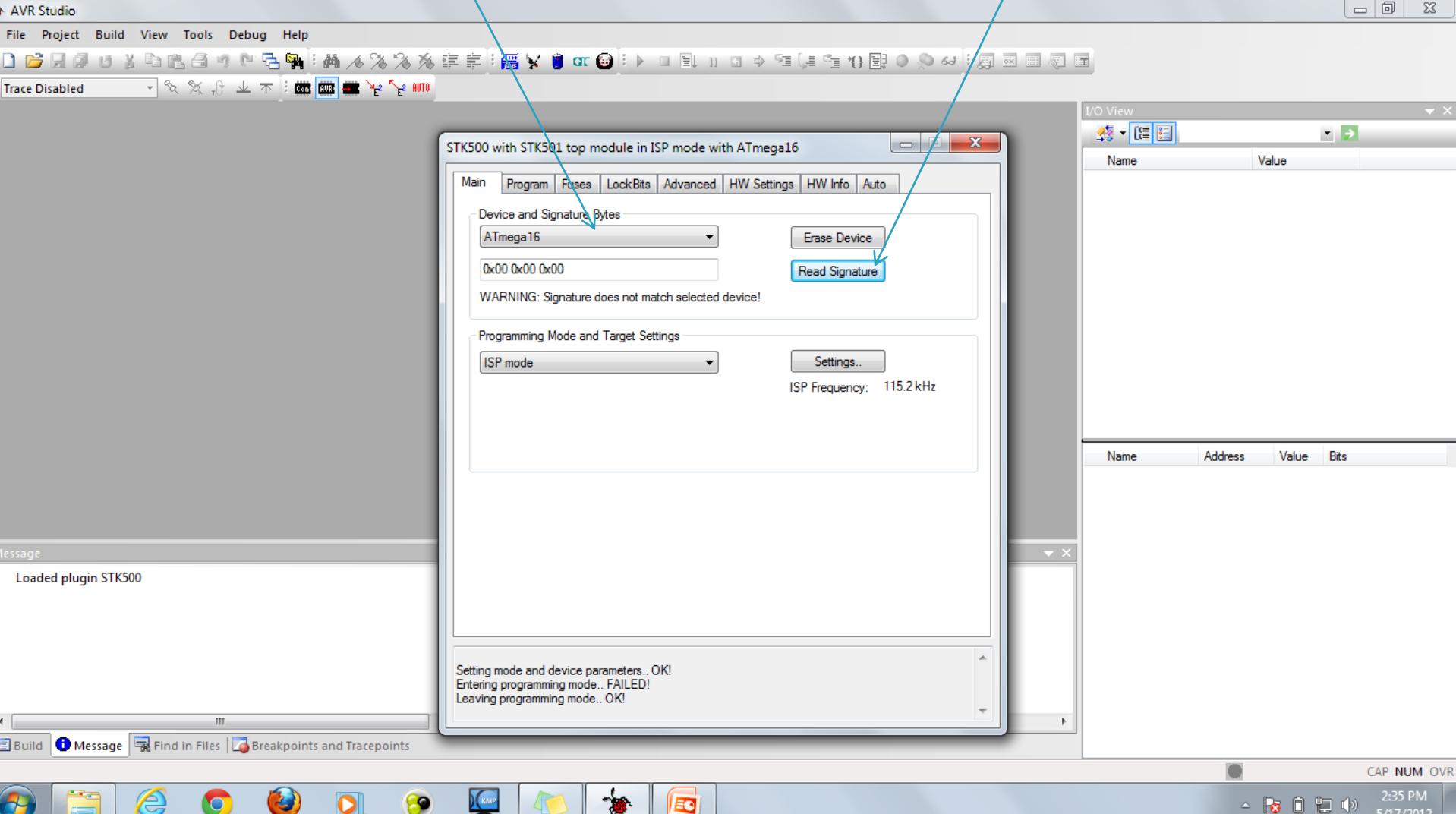
Click on this icon



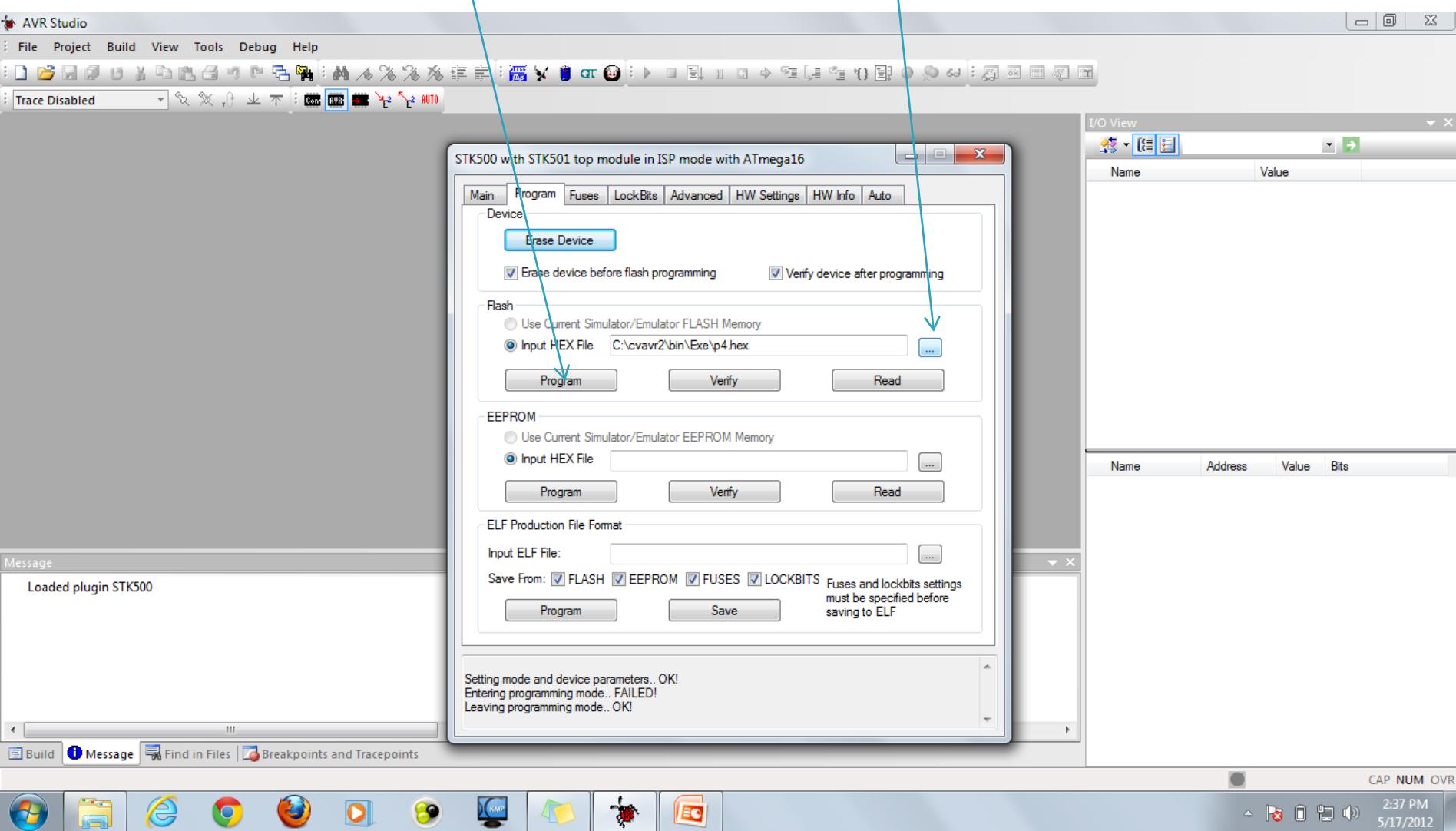
Click Cancel



Select Device and Read signature



Select Hex file and then click program





Difference from Other

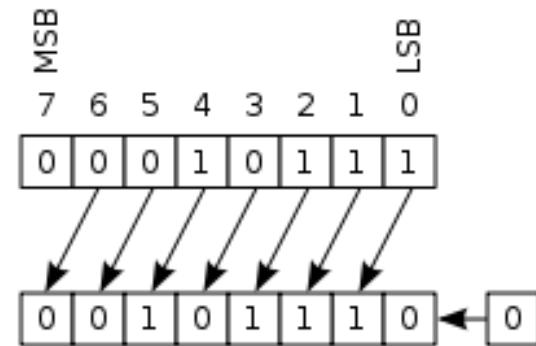
- ▶ You can't write PORTA.1 in this
- ▶ User friendly
- ▶ Easy interface
- ▶ Can add external library

How to Access PortA.1

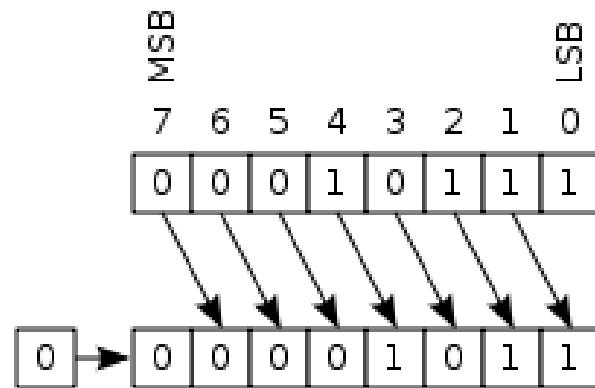
- ▶ Using simple bit wise C operation
- ▶ For PINA.1
- ▶ $\text{Value_PINA1} = (\text{PINA} \& 0b00000010) >> 1$
- ▶ For PORTA.1
- ▶ $\text{PORTA} = \text{PORTA} | 0b00000010 \ // \ \text{PORTA.1} = 1$
- ▶ $\text{PORTA} = \text{PORTA} \& 0b11111101 \ // \ \text{PORTA.1} = 0$

How to Access PortA.x

- ▶ Left shift operator
- ▶ <<



- ▶ Right shift operator
- ▶ >>

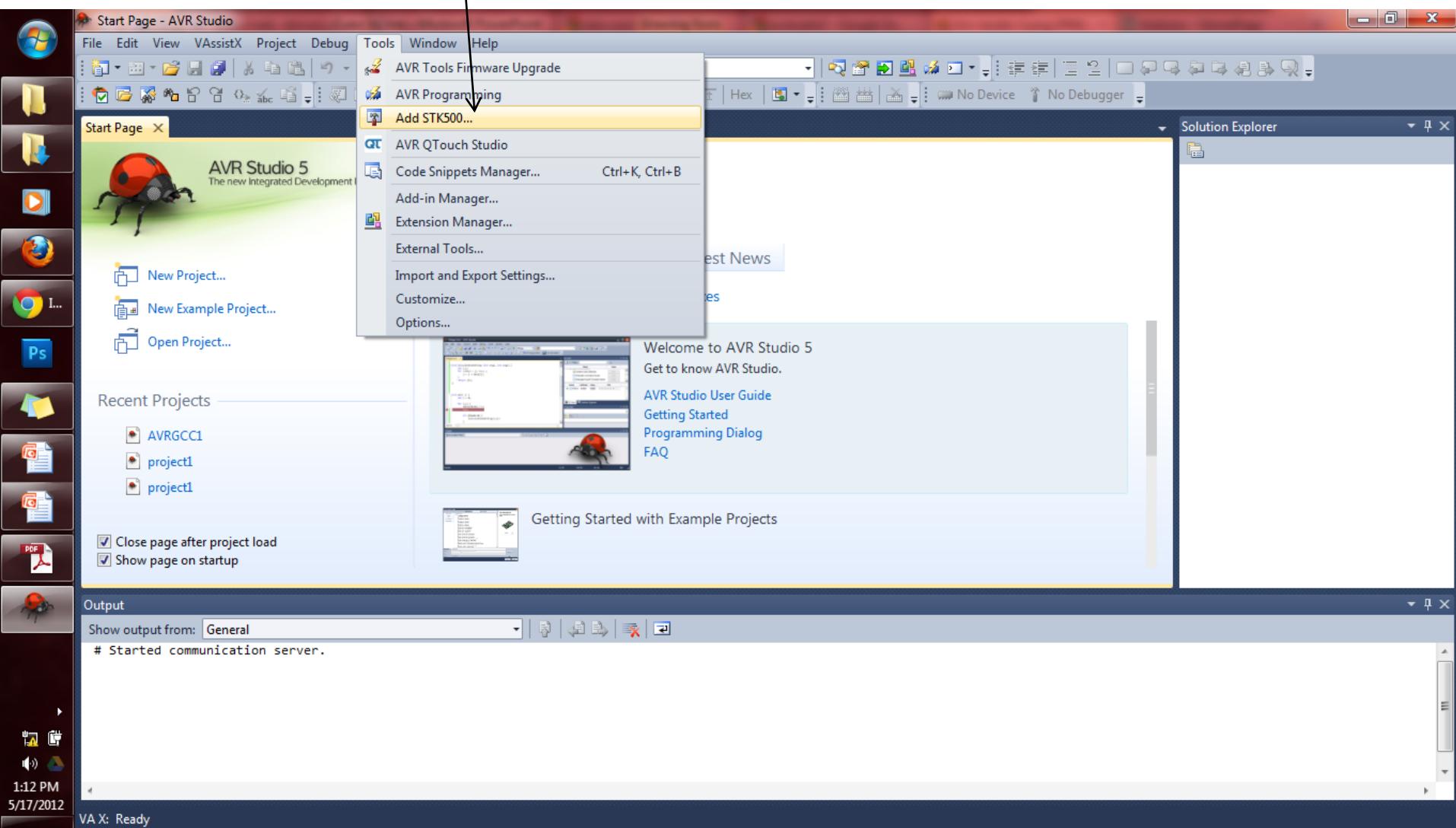


How to Access PortA.x

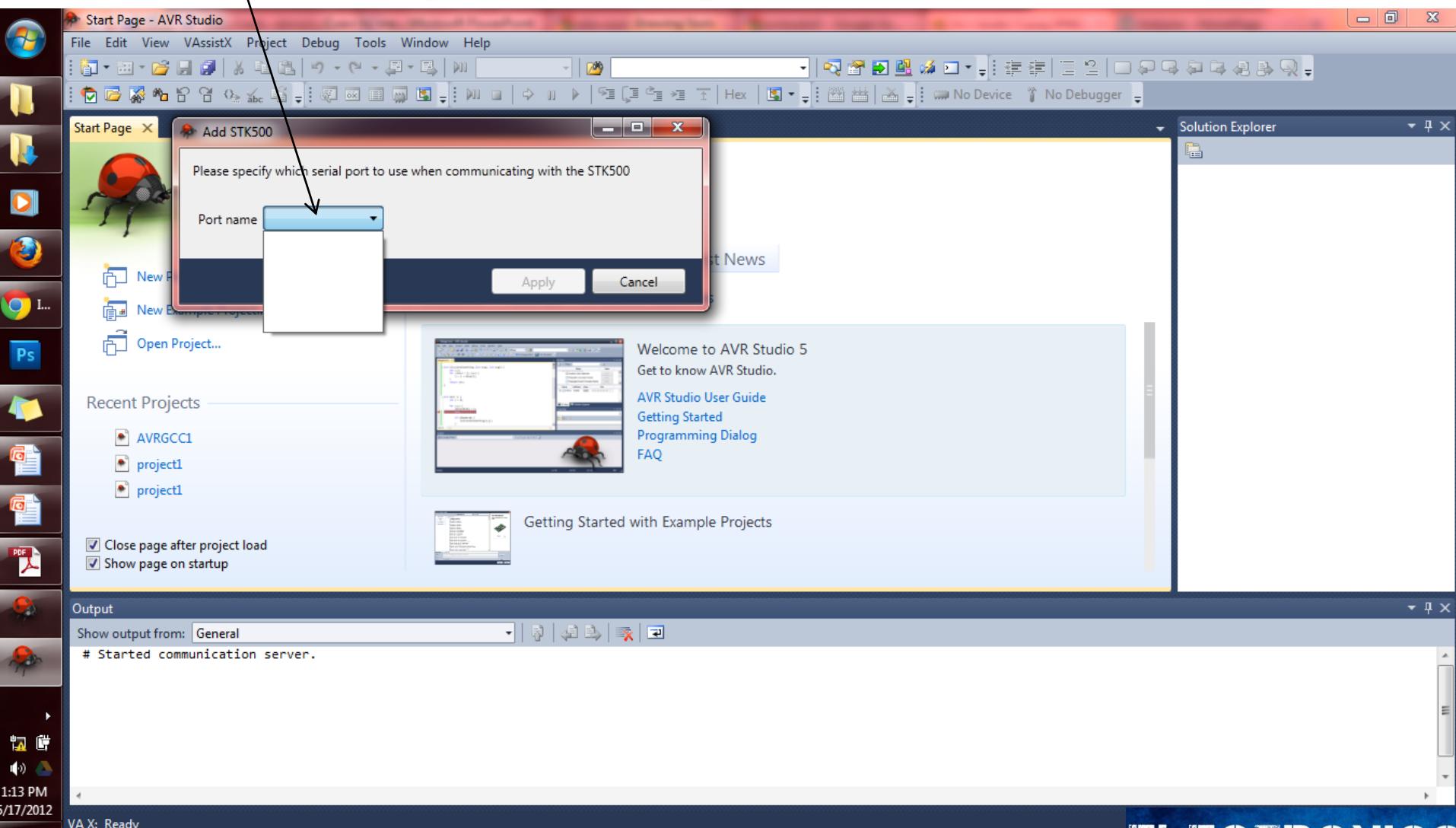
- ▶ For PINA.x
- ▶ Value_PINAx=(PINA & (1<<x))>>x
- ▶ For PORTA.1
- ▶ PORTA=PORTA | (1<<x) // to set PORTA.1=1
- ▶ PORTA=PORTA & (~1<<x)) // set
PORTA.1=0

Programming

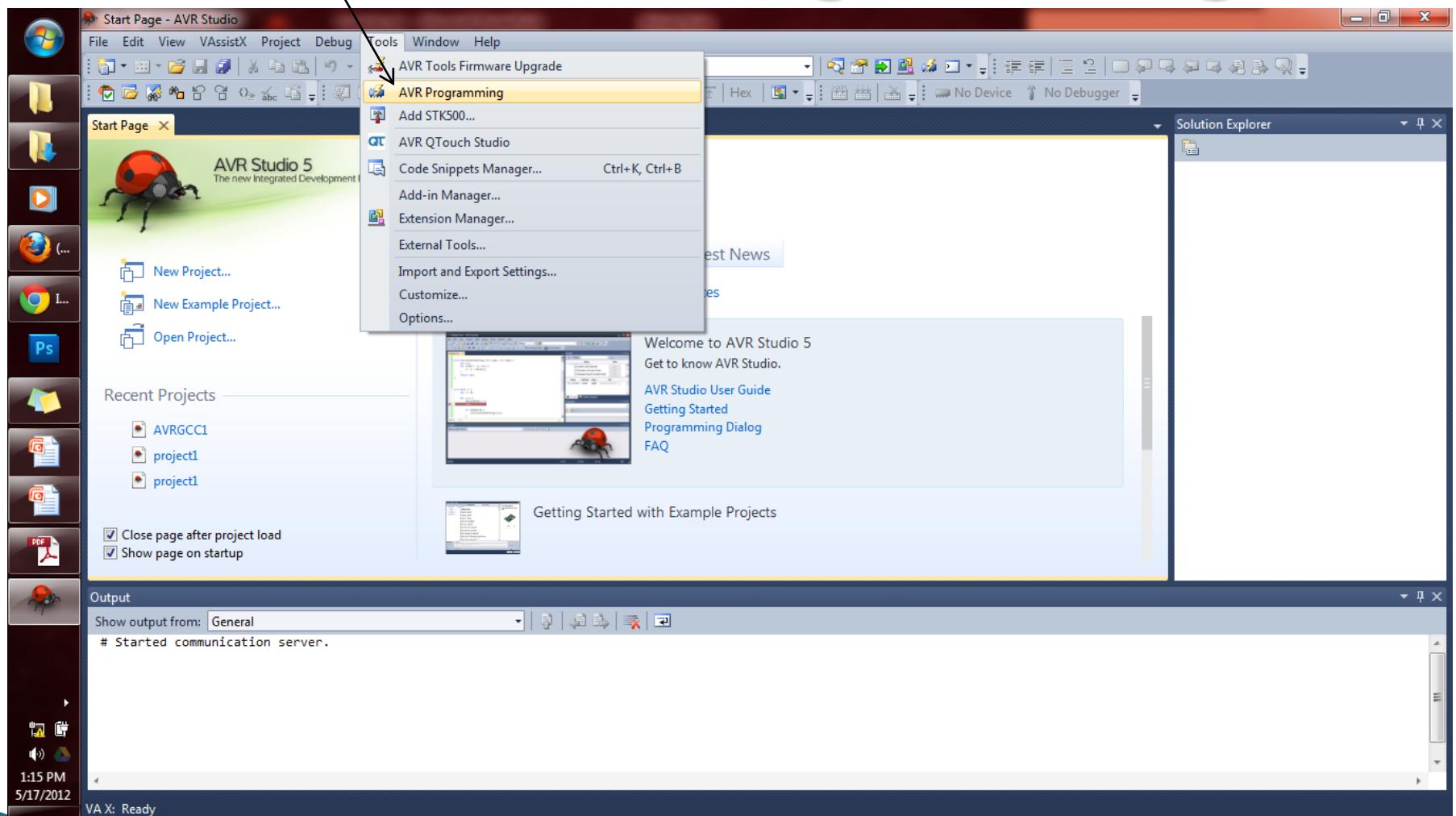
Add Stk500



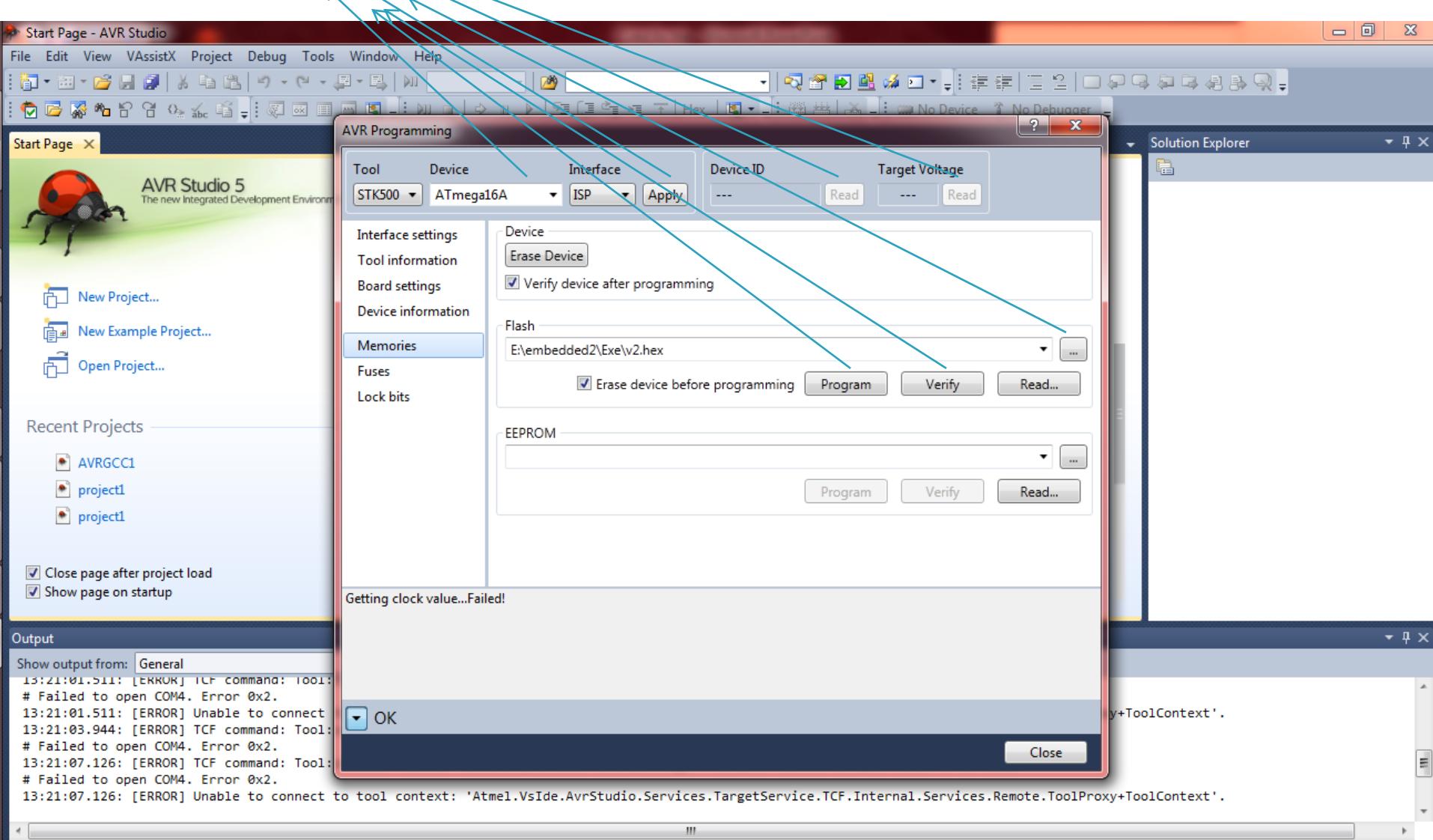
Select Com Port



Select AVR Programming



▶ Select Device → Click Apply → Read Device ID → Read target Voltage → Choose Hex File → Then Program



Thank you

