# AVR Assembly Programming in MPLAB X

## Step by Step Tutorial

Sepehr Naimi

BIHE University

3/18/2024

## Contents

## Dedication

The tutorial is dedicated to the memory of Prof. Muhammad Ali Mazidi who wrote valuable books about hardware and embedded systems and tried his best to serve human beings.

## Introduction

This tutorial will teach you how to write, compile, and trace a simple Assembly program for AVR microcontrollers using MPLAB X.

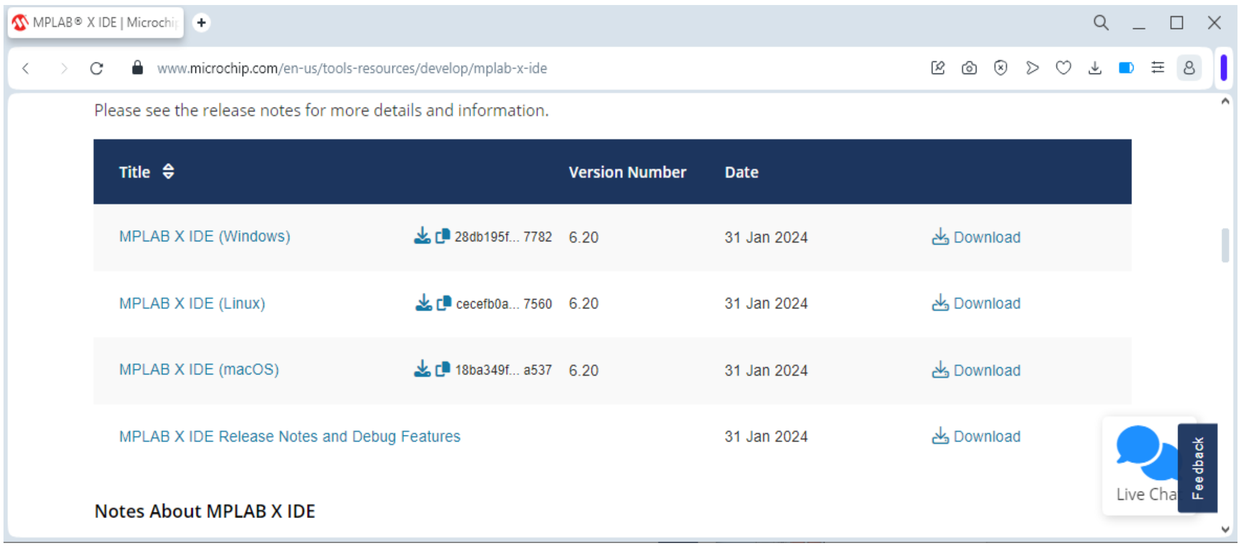
## Downloading and Installing MPLAB X IDE and AVR Assembler

1. Download AVR Assembler from the following link:

   https://nicerland.com/eduFiles/AVR/Software/avrasm2.zip

2. Unzip the downloaded zip file and place ***avrasm2.exe*** in "**C:\**" or anywhere you prefer.
3. Microchip provides the *MPLAB X* IDE for Windows, Linux, and Mac OS for free. Download the proper MPLAB X from the Microchip website:

   https://www.microchip.com/en-us/tools-resources/develop/mplab-x-ide

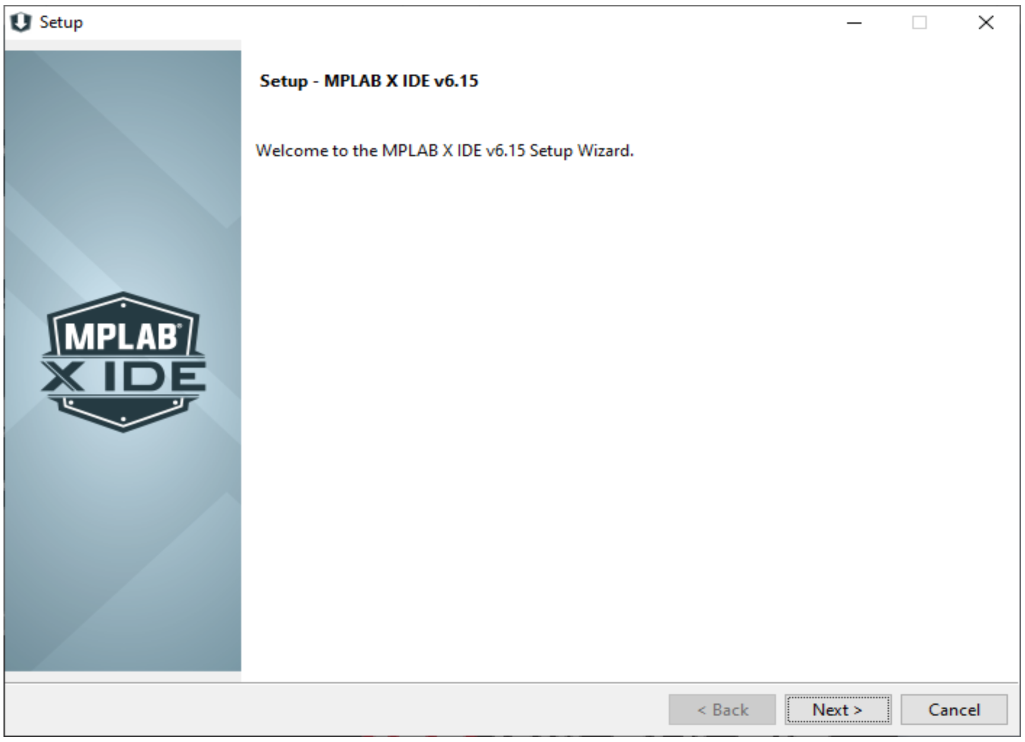


Please see the release notes for more details and information.

| Title | | Version Number | Date | |
|---|---|---|---|---|
| MPLAB X IDE (Windows) | 28db195f... 7782 | 6.20 | 31 Jan 2024 | Download |
| MPLAB X IDE (Linux) | cecefb0a... 7560 | 6.20 | 31 Jan 2024 | Download |
| MPLAB X IDE (macOS) | 18ba349f... a537 | 6.20 | 31 Jan 2024 | Download |
| MPLAB X IDE Release Notes and Debug Features | | | 31 Jan 2024 | Download |

Notes About MPLAB X IDE

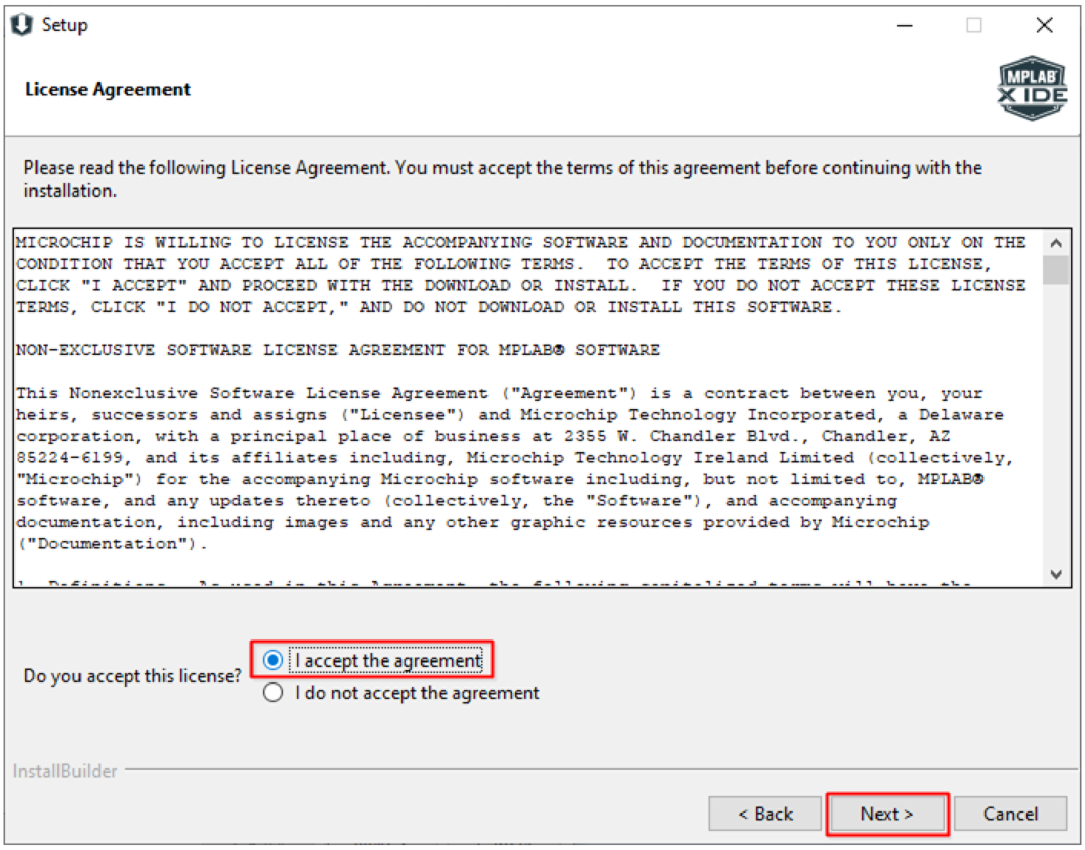
4. Run the downloaded program to install the MPLAB X IDE. Installing the MPLAB X is straight forward.
   A. The first page is just a welcome page. Press ***Next***.



**Setup - MPLAB X IDE v6.15**

Welcome to the MPLAB X IDE v6.15 Setup Wizard.

B. The second page is an agreement. Choose "***I accept the agreement***" and press ***Next***.



Please read the following License Agreement. You must accept the terms of this agreement before continuing with the installation.

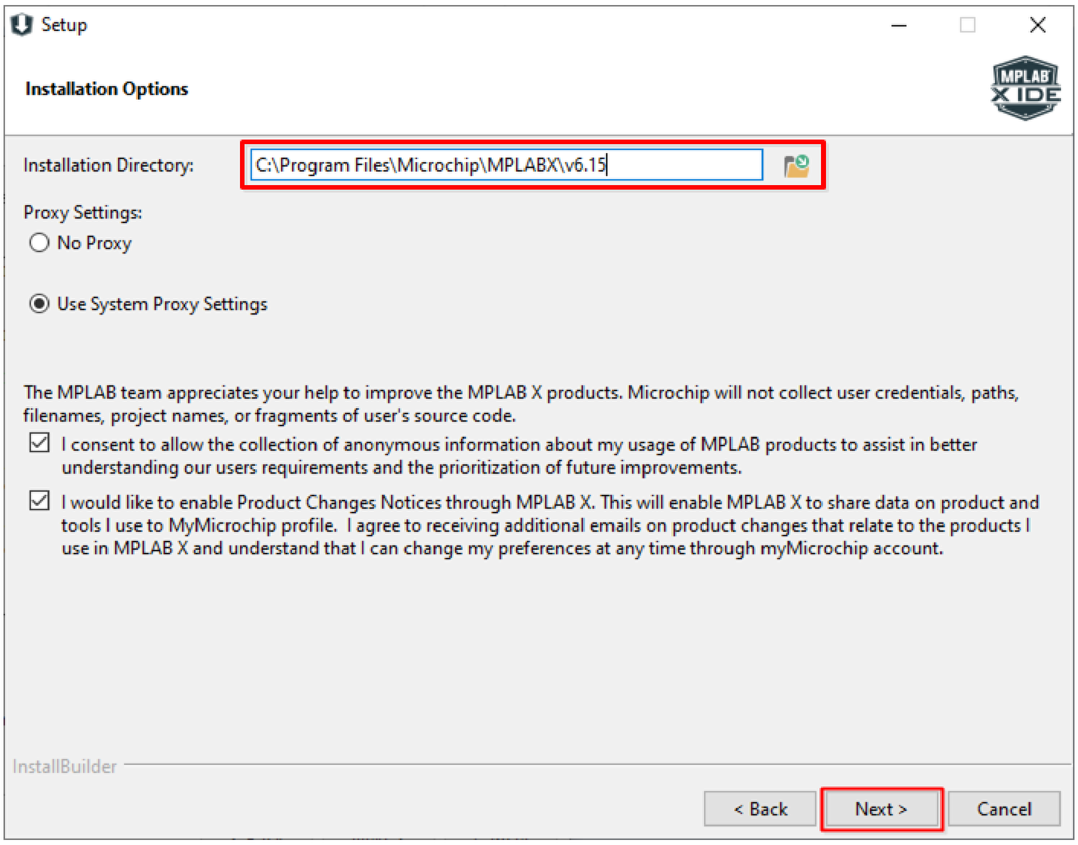
```
MICROCHIP IS WILLING TO LICENSE THE ACCOMPANYING SOFTWARE AND DOCUMENTATION TO YOU ONLY ON THE
CONDITION THAT YOU ACCEPT ALL OF THE FOLLOWING TERMS.  TO ACCEPT THE TERMS OF THIS LICENSE,
CLICK "I ACCEPT" AND PROCEED WITH THE DOWNLOAD OR INSTALL.  IF YOU DO NOT ACCEPT THESE LICENSE
TERMS, CLICK "I DO NOT ACCEPT," AND DO NOT DOWNLOAD OR INSTALL THIS SOFTWARE.

NON-EXCLUSIVE SOFTWARE LICENSE AGREEMENT FOR MPLAB® SOFTWARE

This Nonexclusive Software License Agreement ("Agreement") is a contract between you, your
heirs, successors and assigns ("Licensee") and Microchip Technology Incorporated, a Delaware
corporation, with a principal place of business at 2355 W. Chandler Blvd., Chandler, AZ
85224-6199, and its affiliates including, Microchip Technology Ireland Limited (collectively,
"Microchip") for the accompanying Microchip software including, but not limited to, MPLAB®
software, and any updates thereto (collectively, the "Software"), and accompanying
documentation, including images and any other graphic resources provided by Microchip
("Documentation").
```

Do you accept this license? [✓ I accept the agreement] I do not accept the agreement

C. In the third page, you can choose the path for installing the ***MPLAB X***. Press ***Next***.



Installation Directory: C:\Program Files\Microchip\MPLABX\v6.15

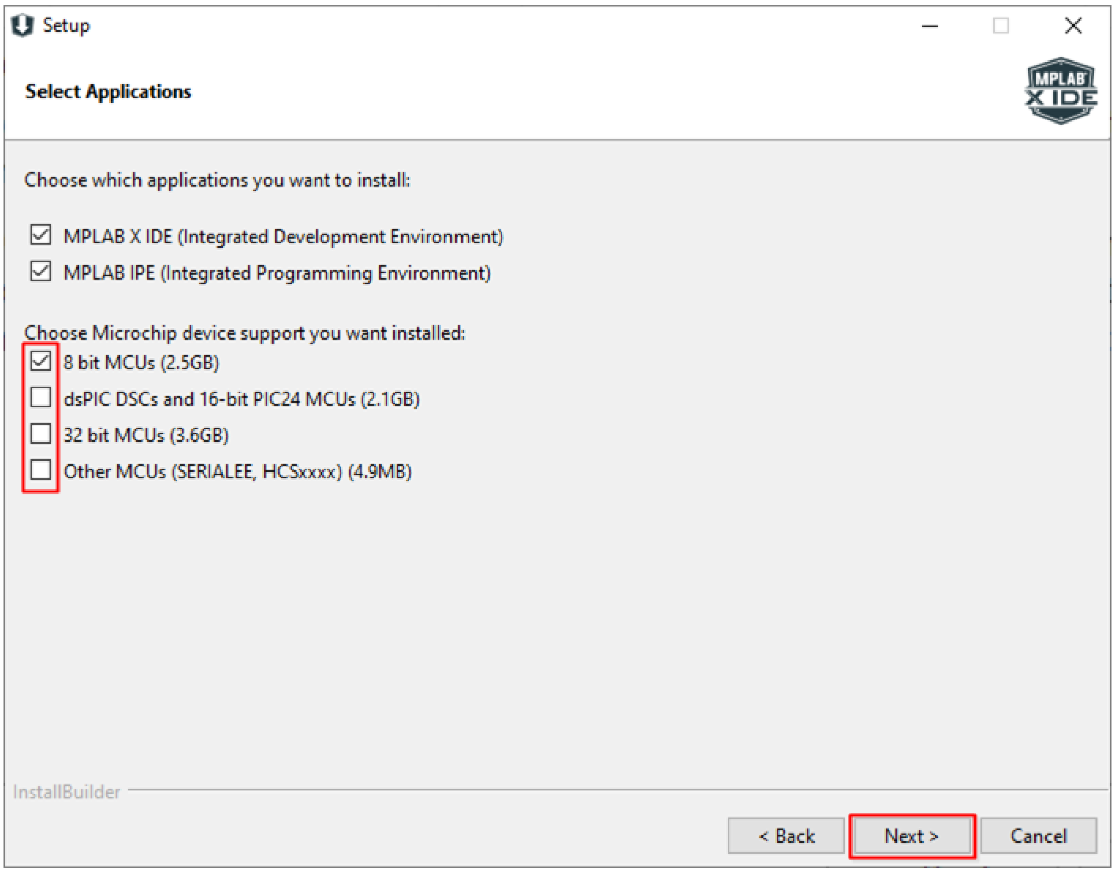
Proxy Settings:
No Proxy [✓ Use System Proxy Settings]

The MPLAB team appreciates your help to improve the MPLAB X products. Microchip will not collect user credentials, paths, filenames, project names, or fragments of user's source code.

☑ I consent to allow the collection of anonymous information about my usage of MPLAB products to assist in better understanding our users requirements and the prioritization of future improvements.

☑ I would like to enable Product Changes Notices through MPLAB X. This will enable MPLAB X to share data on product and tools I use to MyMicrochip profile. I agree to receiving additional emails on product changes that relate to the products I use in MPLAB X and understand that I can change my preferences at any time through myMicrochip account.

D. In the page, you can choose the chips you want to use. To install the IDE faster, uncheck the MCUs except the ***8-bit MCUs***.


Setup

**Select Applications**

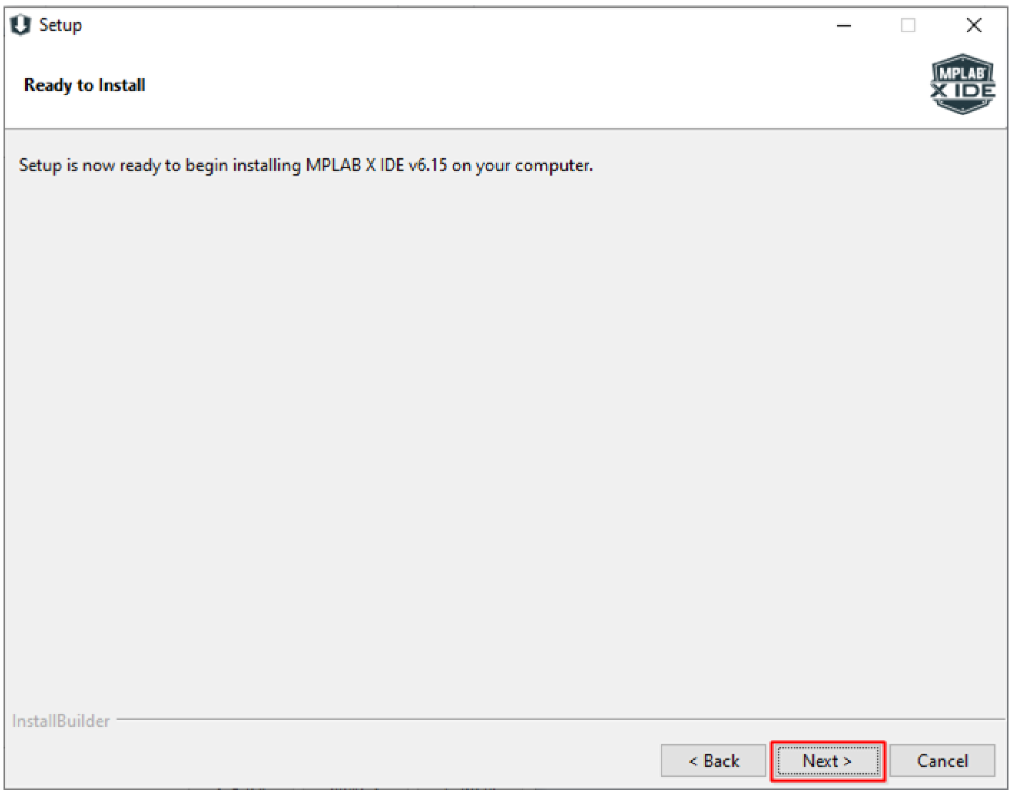
Choose which applications you want to install:

- [✓] MPLAB X IDE (Integrated Development Environment)
- [✓] MPLAB IPE (Integrated Programming Environment)

Choose Microchip device support you want installed:

- [✓] 8 bit MCUs (2.5GB)
- [ ] dsPIC DSCs and 16-bit PIC24 MCUs (2.1GB)
- [ ] 32 bit MCUs (3.6GB)
- [ ] Other MCUs (SERIALEE, HCSxxxx) (4.9MB)

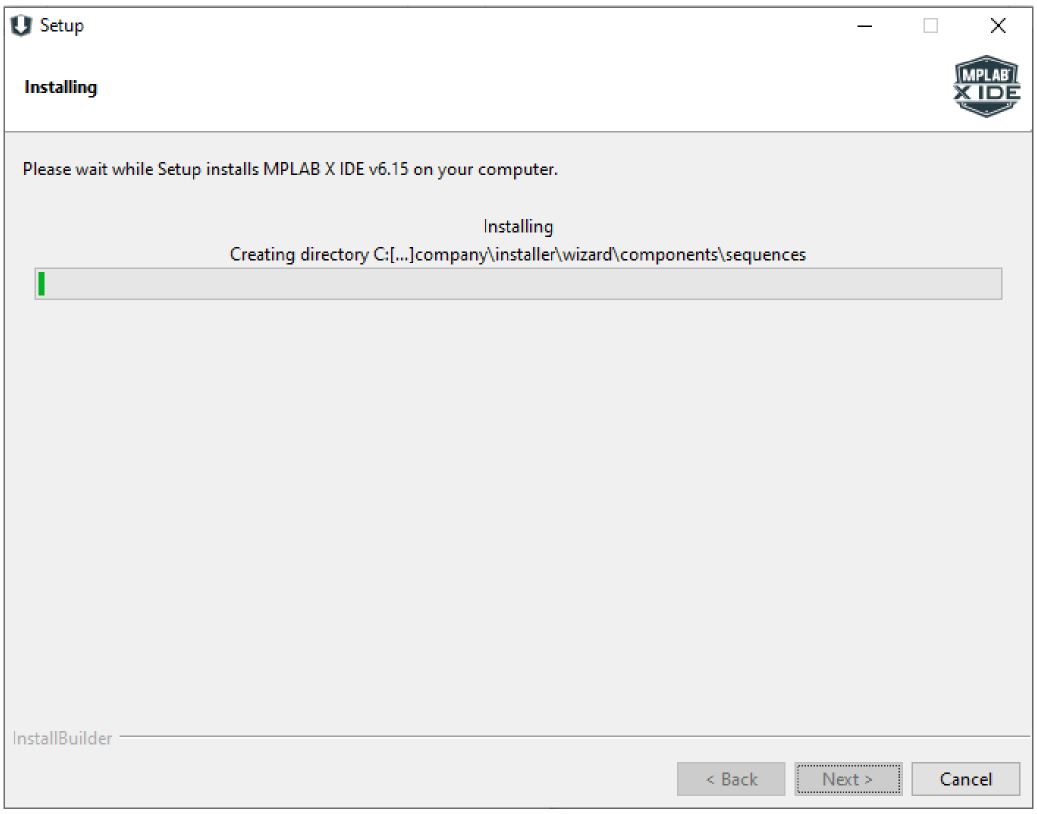
E. Press ***Next***.


Setup

**Ready to Install**

Setup is now ready to begin installing MPLAB X IDE v6.15 on your computer.

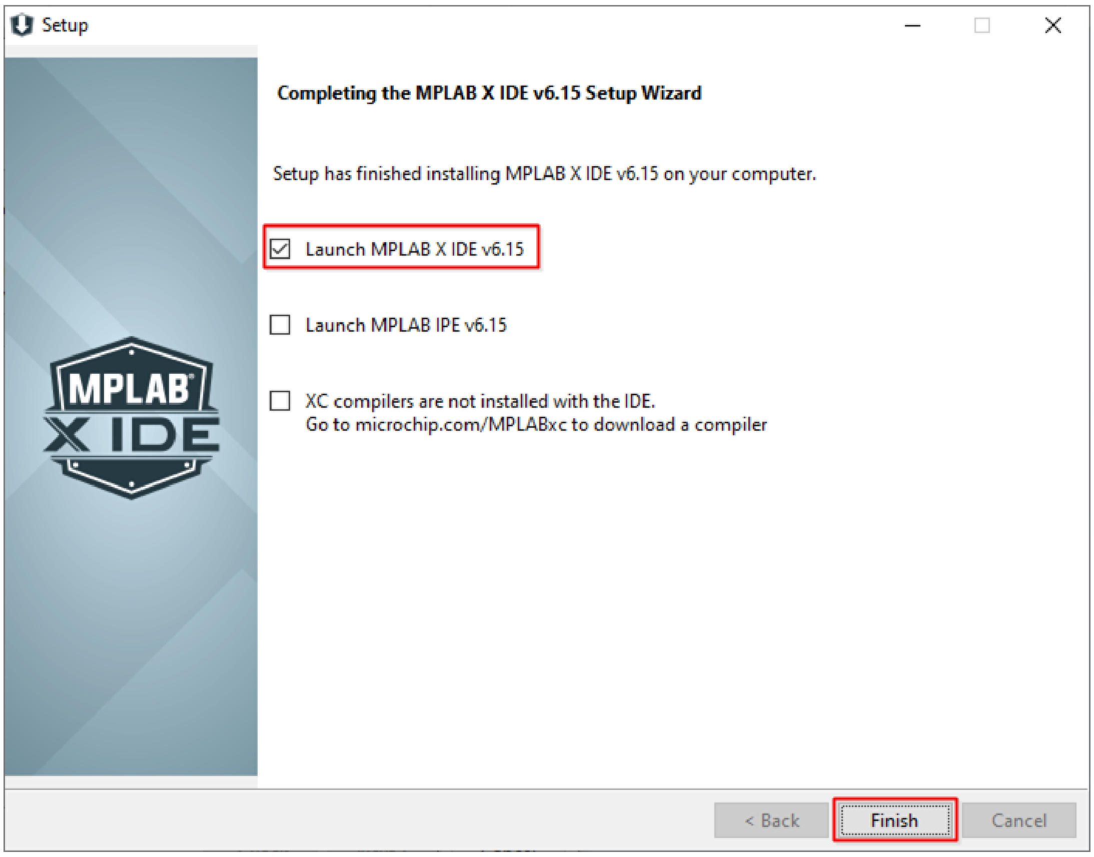
F. Wait for a few minutes until the IDE installs.



Please wait while Setup installs MPLAB X IDE v6.15 on your computer.

Installing
Creating directory C:[...]company\installer\wizard\components\sequences

G. Now, the installation is completed. Press ***Finish*** to close the installer and the MPLAB X IDE opens.

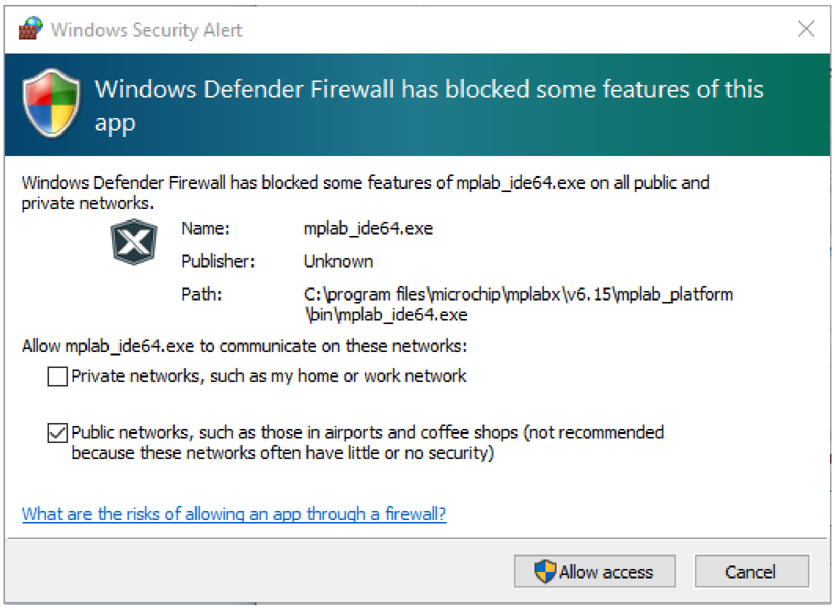


**Completing the MPLAB X IDE v6.15 Setup Wizard**

Setup has finished installing MPLAB X IDE v6.15 on your computer.

- [✓] Launch MPLAB X IDE v6.15
- [ ] Launch MPLAB IPE v6.15
- [ ] XC compilers are not installed with the IDE.
  Go to microchip.com/MPLABxc to download a compiler

5. Windows Firewall might ask if MPLAB can access to the network or not. Choose ***Allow access***.



Windows Security Alert

Windows Defender Firewall has blocked some features of this app

Windows Defender Firewall has blocked some features of mplab_ide64.exe on all public and private networks.

| | |
|---|---|
| Name: | mplab_ide64.exe |
| Publisher: | Unknown |
| Path: | C:\program files\microchip\mplabx\v6.15\mplab_platform \bin\mplab_ide64.exe |

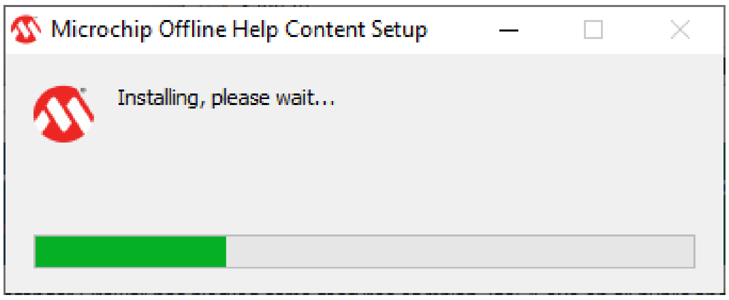
Allow mplab_ide64.exe to communicate on these networks:

☐ Private networks, such as my home or work network

☑ Public networks, such as those in airports and coffee shops (not recommended because these networks often have little or no security)

What are the risks of allowing an app through a firewall?

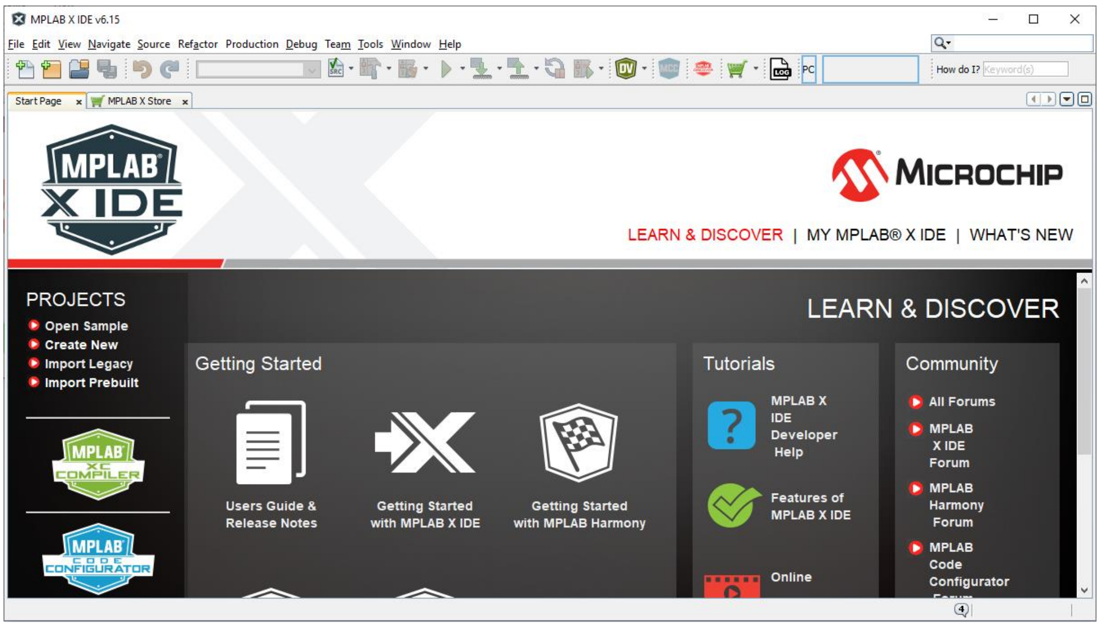
6. Microchip automatically installs Microchip Offline Help. Wait a few seconds until the install finishes.



Microchip Offline Help Content Setup

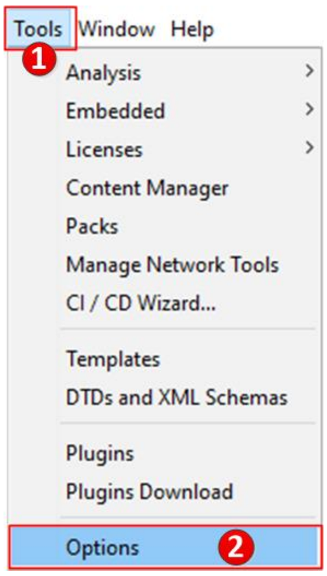
Installing, please wait...

7. Now, the MPLAB X IDE opens.

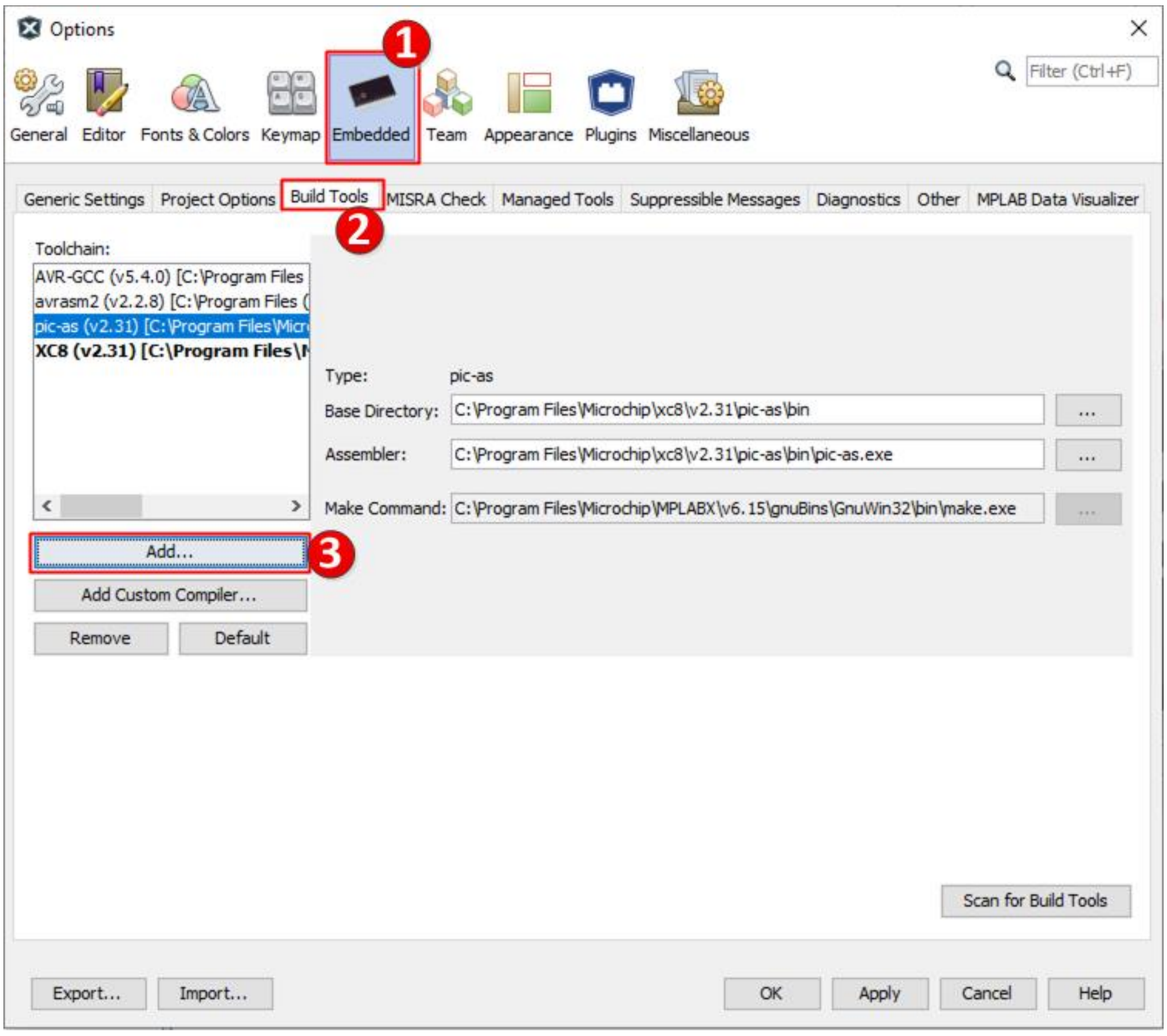


## Adding AVR Assembler to MPLAB toolchain

8. Go to the ***Tools*** menu and choose ***Options***.



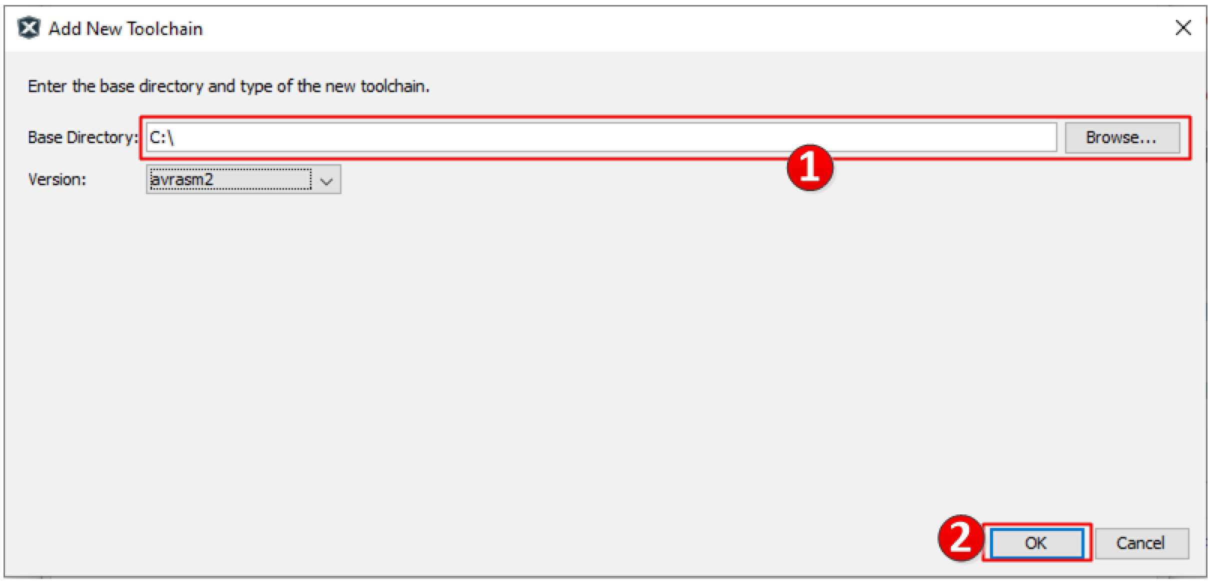
9. Click on ***Embedded***, choose ***Build Tools***, and click on the ***Add…*** button.



Toolchain:
AVR-GCC (v5.4.0) [C:\Program Files
avrasm2 (v2.2.8) [C:\Program Files (
pic-as (v2.31) [C:\Program Files\Micr
**XC8 (v2.31) [C:\Program Files\M**

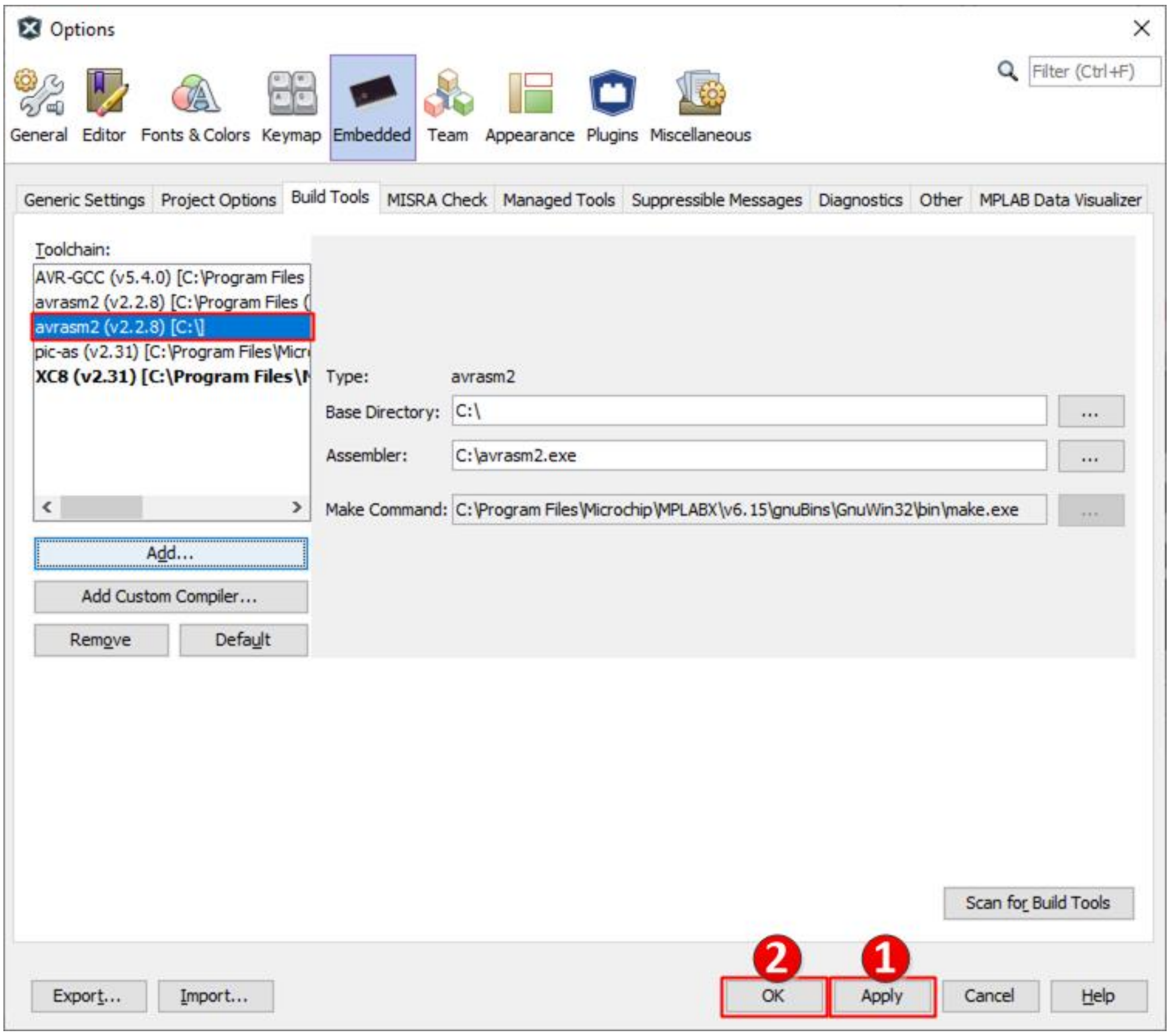
Type: pic-as
Base Directory: C:\Program Files\Microchip\xc8\v2.31\pic-as\bin
Assembler: C:\Program Files\Microchip\xc8\v2.31\pic-as\bin\pic-as.exe
Make Command: C:\Program Files\Microchip\MPLABX\v6.15\gnuBins\GnuWin32\bin\make.exe

10. Type ***C:\*** if you placed ***avrasm2.exe*** in the root of C. (or choose the path for avrasm2.exe by clicking on the ***Browse*** button.) Then, press ***OK***.



Base Directory: C:\
Version: avrasm2

11. ***avrasm2*** should be added to the toolchain list. Press ***Apply*** and then ***OK***.

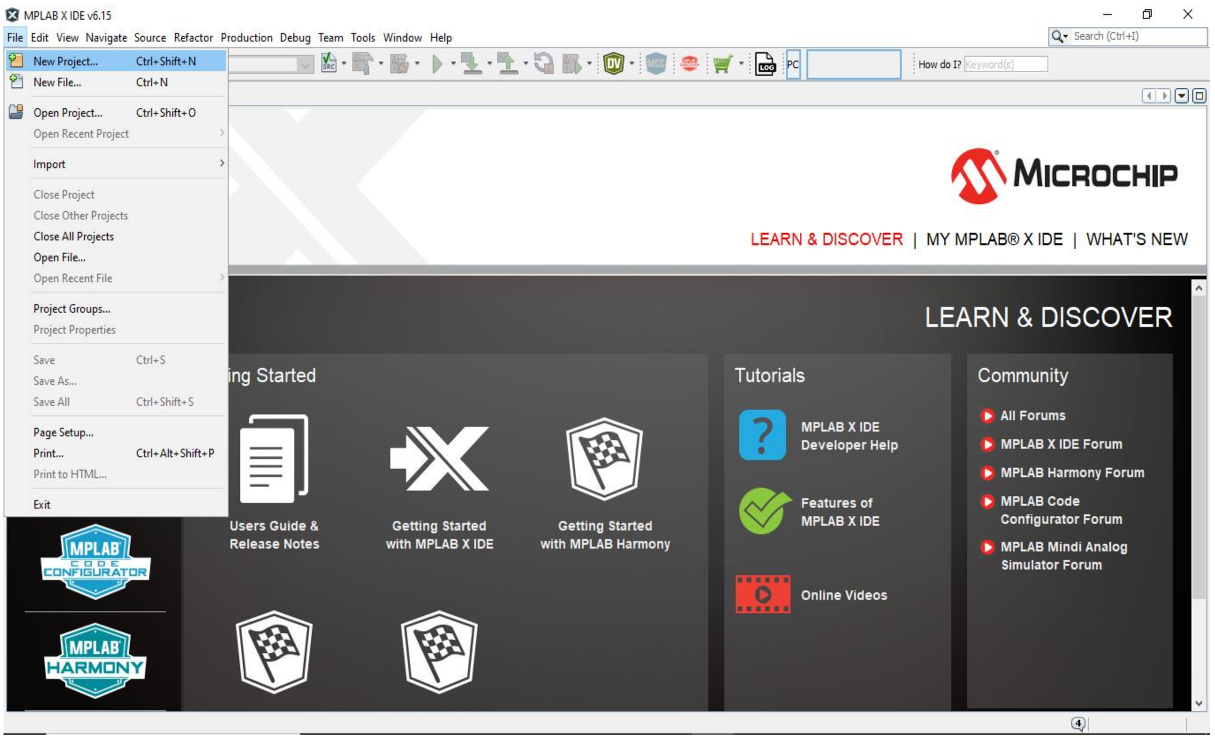


Toolchain:
AVR-GCC (v5.4.0) [C:\Program Files
avrasm2 (v2.2.8) [C:\Program Files (
avrasm2 (v2.2.8) [C:\]
pic-as (v2.31) [C:\Program Files\Micr
**XC8 (v2.31) [C:\Program Files\N**

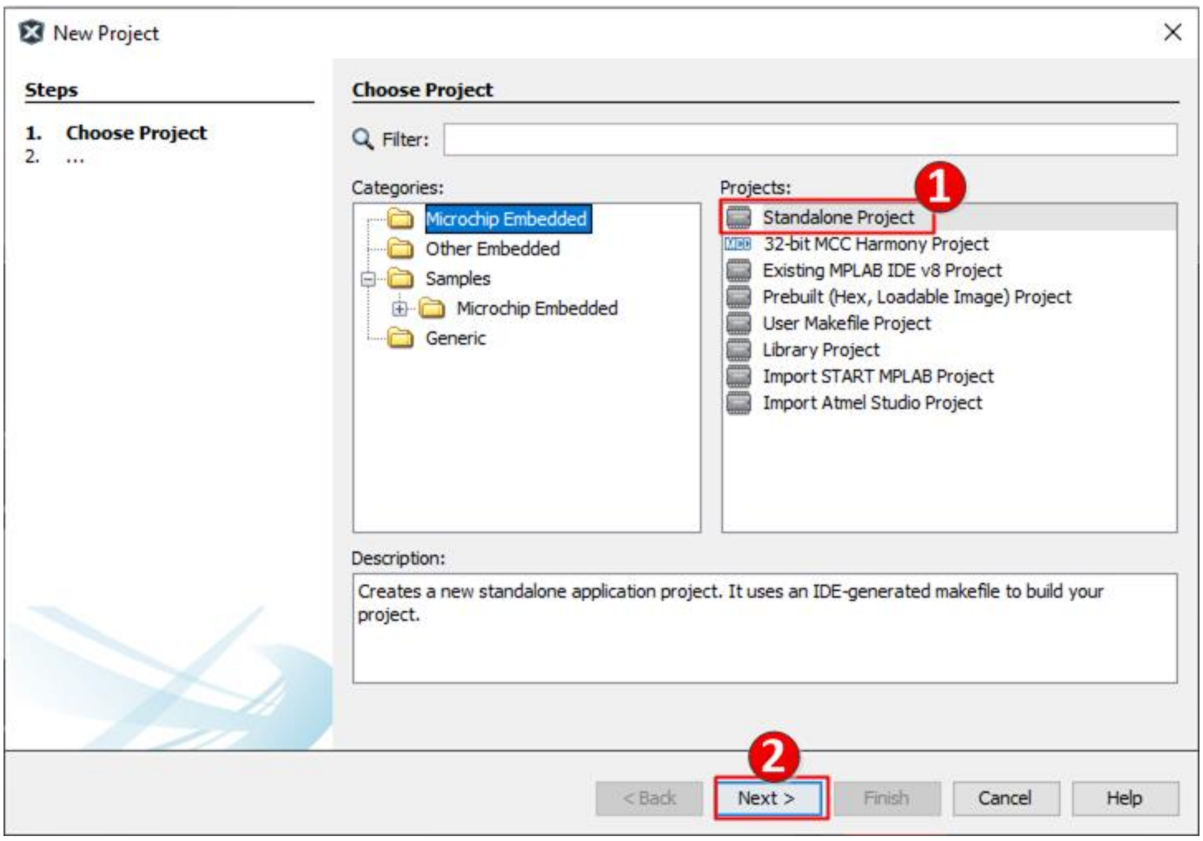
Type: avrasm2
Base Directory: C:\
Assembler: C:\avrasm2.exe
Make Command: C:\Program Files\Microchip\MPLABX\v6.15\gnuBins\GnuWin32\bin\make.exe

## Creating the first project

1. Go to the ***File*** menu and choose ***New Project***.

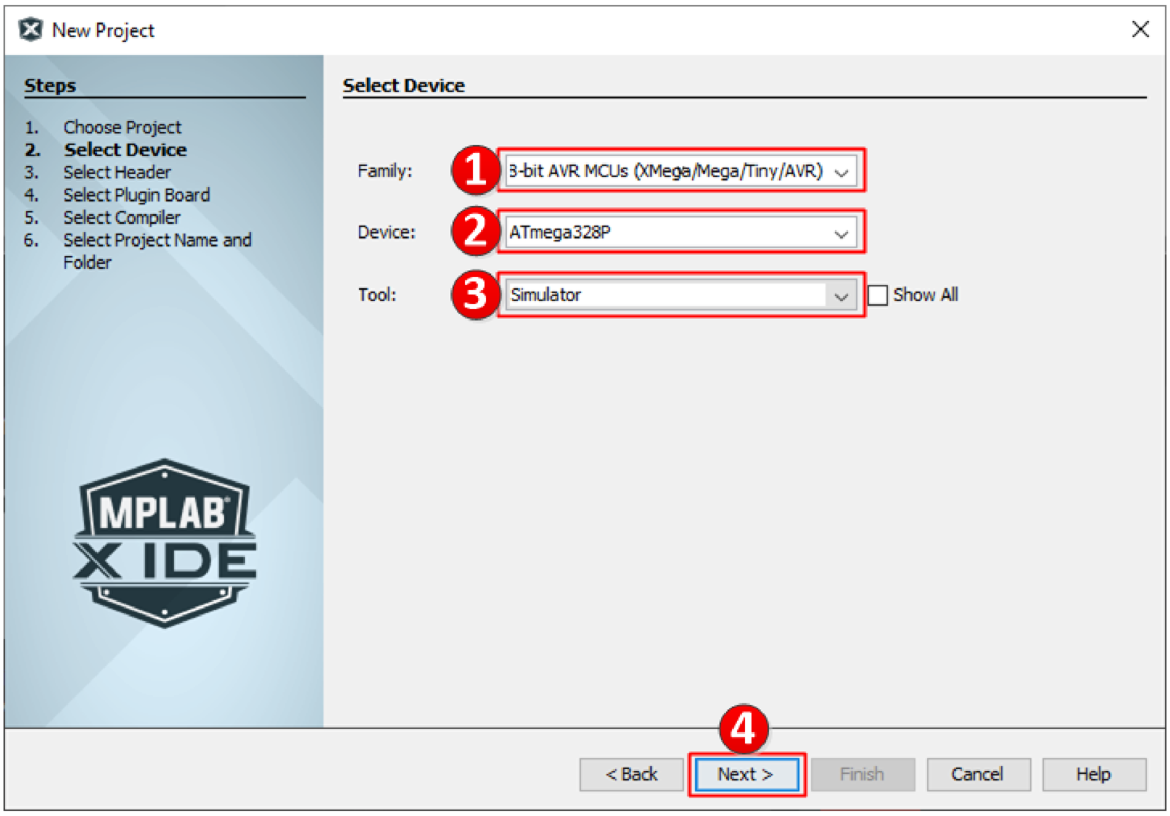


2. In the opened dialog, choose ***Standalone Project*** and press ***Next***.



Creates a new standalone application project. It uses an IDE-generated makefile to build your project.

3. In the dialog:
    a. Select ***8-bit AVR MCUs*** as the device ***family***.
    b. Choose ***ATmega328P*** (or any other Chip you want to use)
    c. Select ***Simulator***.
    d. Press ***Next***.

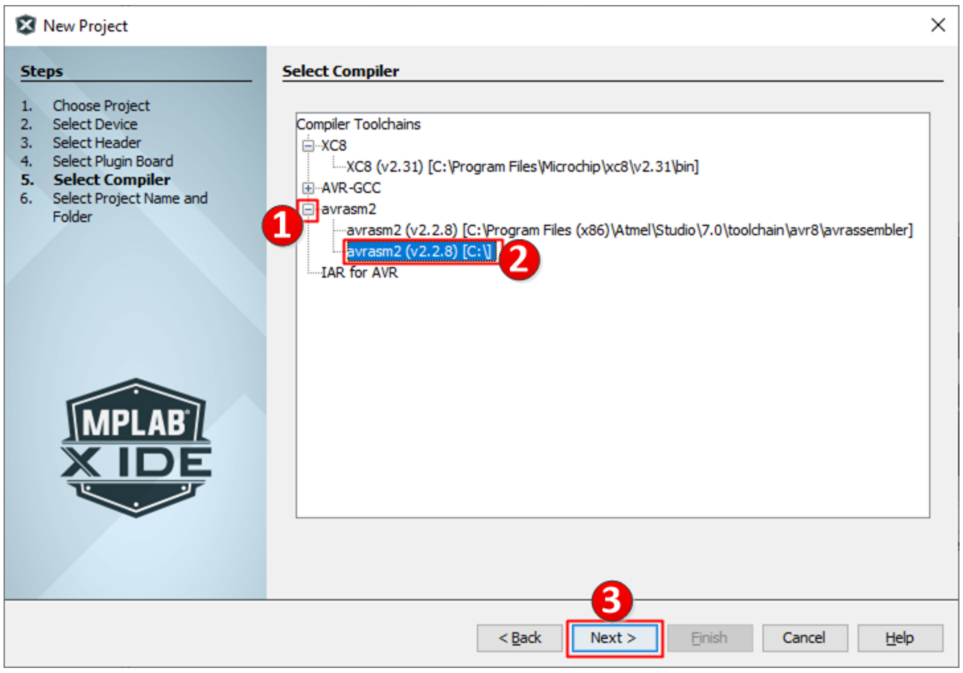


Family: 8-bit AVR MCUs (XMega/Mega/Tiny/AVR)

Device: ATmega328P

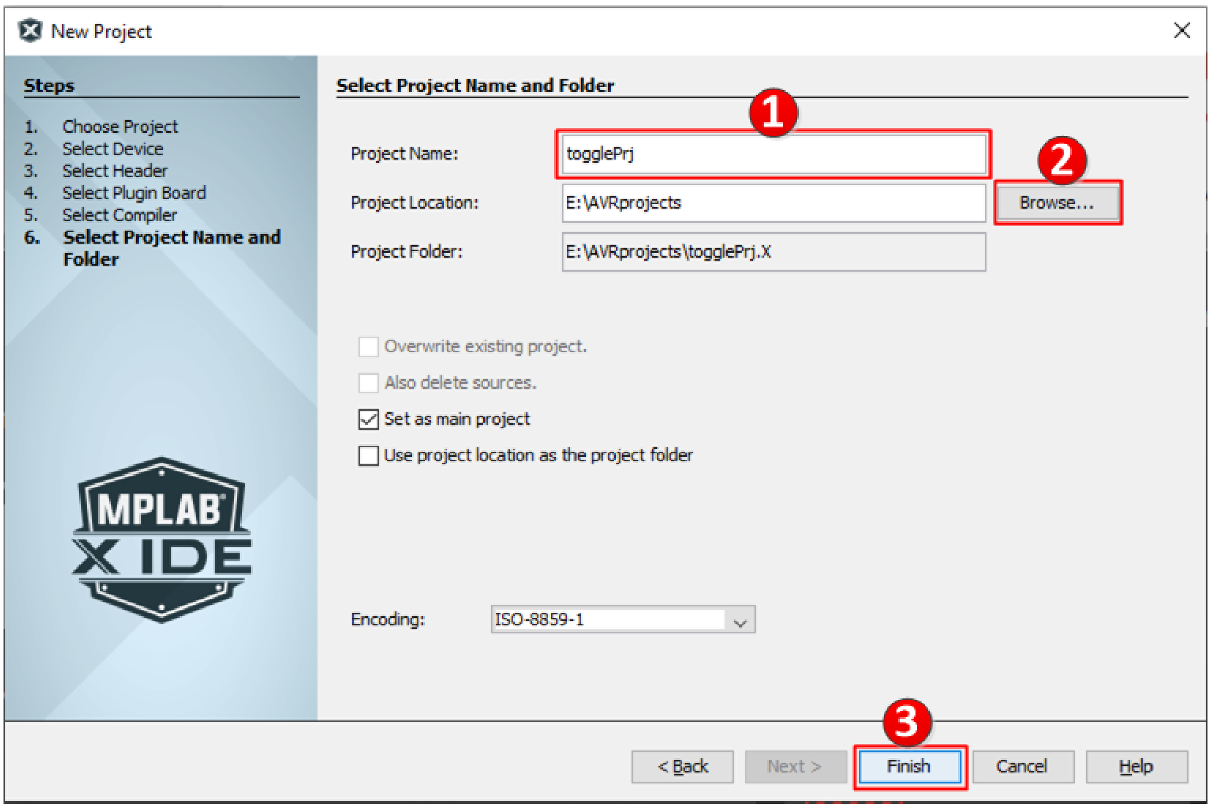
Tool: Simulator

4. Click on the **+** next to ***avrasm2*** to expand and choose ***avrasm2*** as the compiler. Then, press ***Next***.



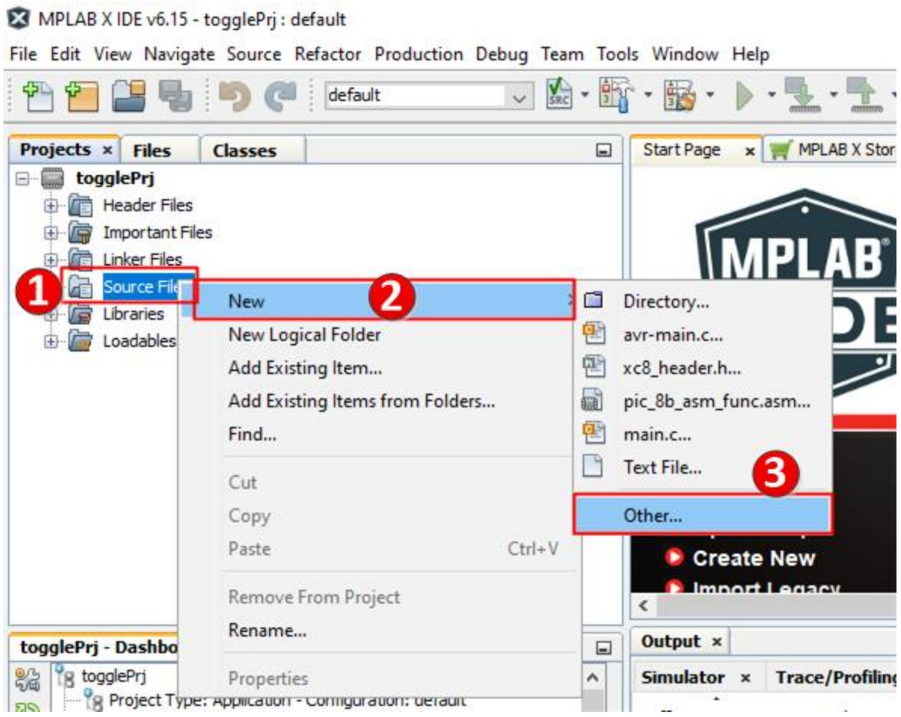
```
Compiler Toolchains
⊟ XC8
   XC8 (v2.31) [C:\Program Files\Microchip\xc8\v2.31\bin]
⊞ AVR-GCC
⊟ avrasm2
   avrasm2 (v2.2.8) [C:\Program Files (x86)\Atmel\Studio\7.0\toolchain\avr8\avrassembler]
   avrasm2 (v2.2.8) [C:\]
IAR for AVR
```

5. Type ***togglePrj*** as the ***Project Name***. Choose the location where the project is being saved, by clicking on the ***Browse*** button. Then, click on the ***Finish*** button. The project is created after a few seconds.

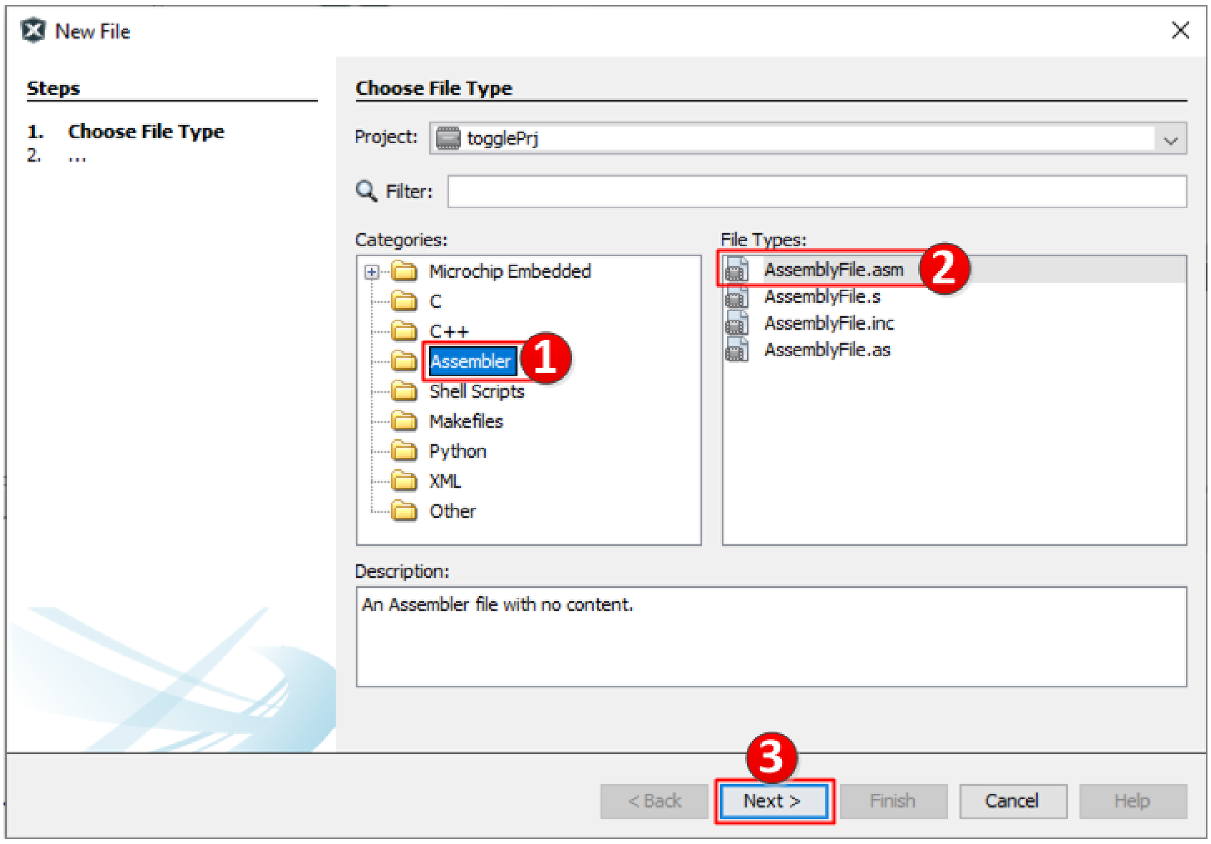


Project Name: togglePrj
Project Location: E:\AVRprojects
Project Folder: E:\AVRprojects\togglePrj.X
[✓] Set as main project
[ ] Use project location as the project folder
Encoding: ISO-8859-1

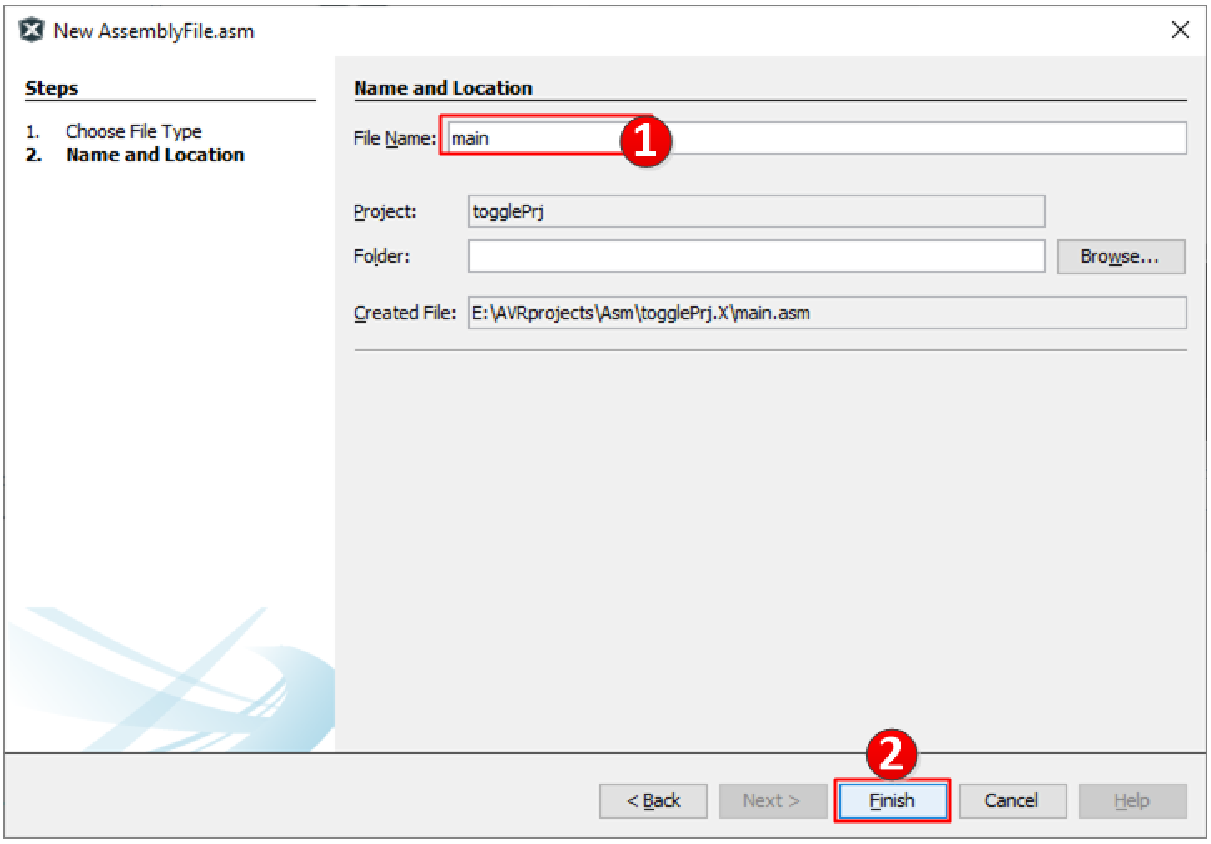
6. To add an assembly file to the project:
   a. Right click on ***Source Files***. Choose ***New*** and then ***Other***.



b. Click on ***Assembler*** and choose ***AssemblyFile.asm***. Then, click ***Next***.



c. Type a name, e.g. ***main***, as the ***File Name***. Then, press ***Finish***.

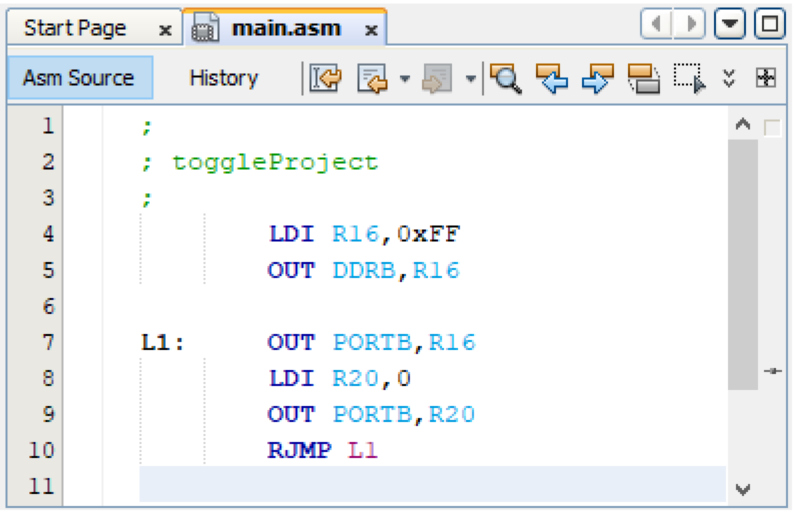


## Writing the first Assembly program

7. Type the following program.

```
;
; toggleProject
;
        LDI  R16,0xFF
        OUT  DDRB,R16

L1:     OUT  PORTB,R16
        LDI  R20,0
        OUT  PORTB,R20
        RJMP  L1
```

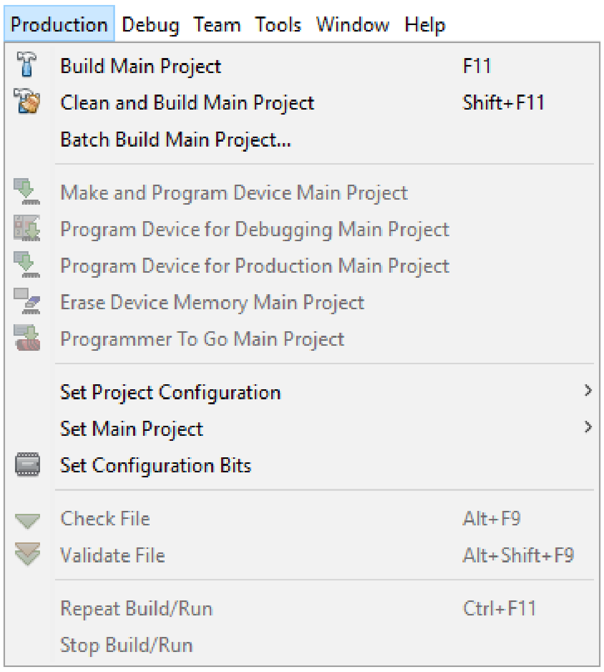


```
;
; toggleProject
;
        LDI  R16,0xFF
        OUT  DDRB,R16

L1:     OUT  PORTB,R16
        LDI  R20,0
        OUT  PORTB,R20
        RJMP  L1
```

## Building

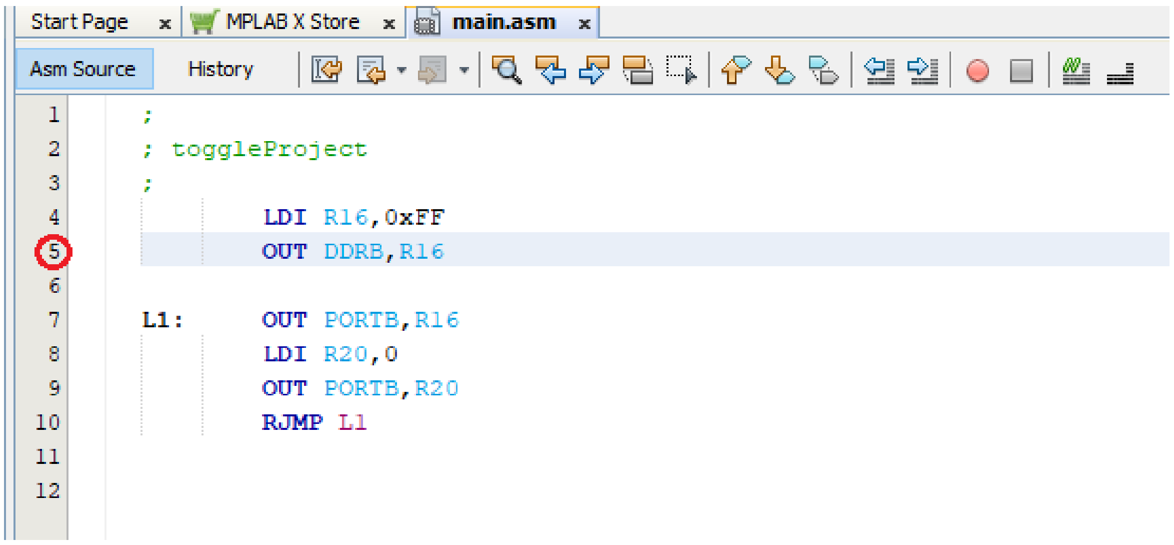
8. Press ***F11*** to assemble, or choose ***Build Main Project*** from the ***Production*** menu. The results of assembling the program are shown in the ***Output*** window.



Production
Debug
Team
Tools
Window
Help
Build Main Project F11
Clean and Build Main Project Shift+F11
Batch Build Main Project...
Make and Program Device Main Project
Program Device for Debugging Main Project
Program Device for Production Main Project
Erase Device Memory Main Project
Programmer To Go Main Project
Set Project Configuration
Set Main Project
Set Configuration Bits
Check File Alt+F9
Validate File Alt+Shift+F9
Repeat Build/Run Ctrl+F11
Stop Build/Run

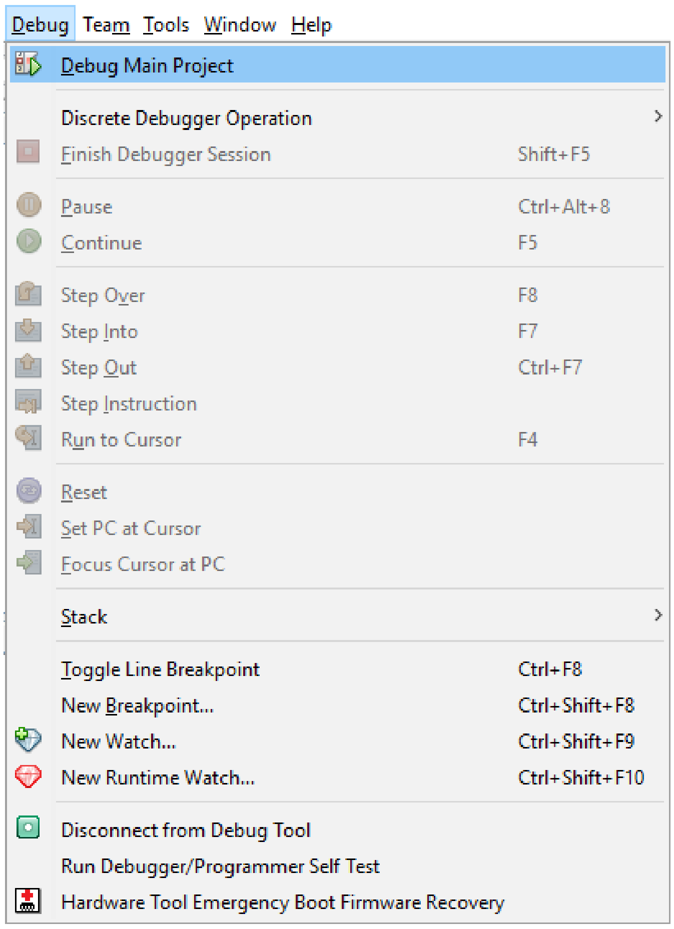
## Debugging

9. To add a breakpoint, click on the line number next to the "`OUT DDRB,R16`" (number 5 in the figure). A red square appears on the left side of the "`OUT DDRB,R16`" instruction and the line becomes red.

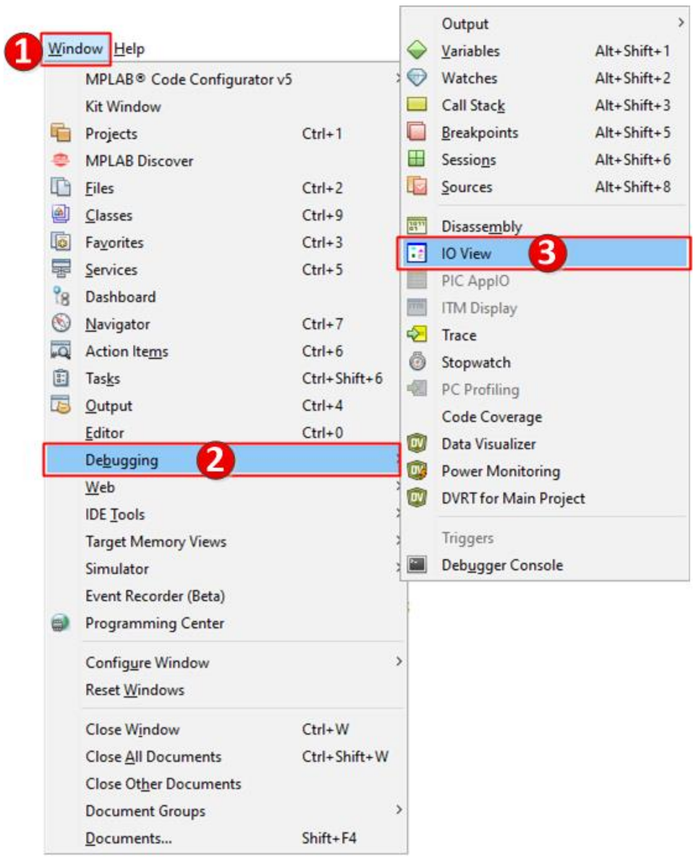


```
1   ;
2   ; toggleProject
3   ;
4           LDI R16,0xFF
5           OUT DDRB,R16
6
7   L1:     OUT PORTB,R16
8           LDI R20,0
9           OUT PORTB,R20
10          RJMP L1
11
12
```

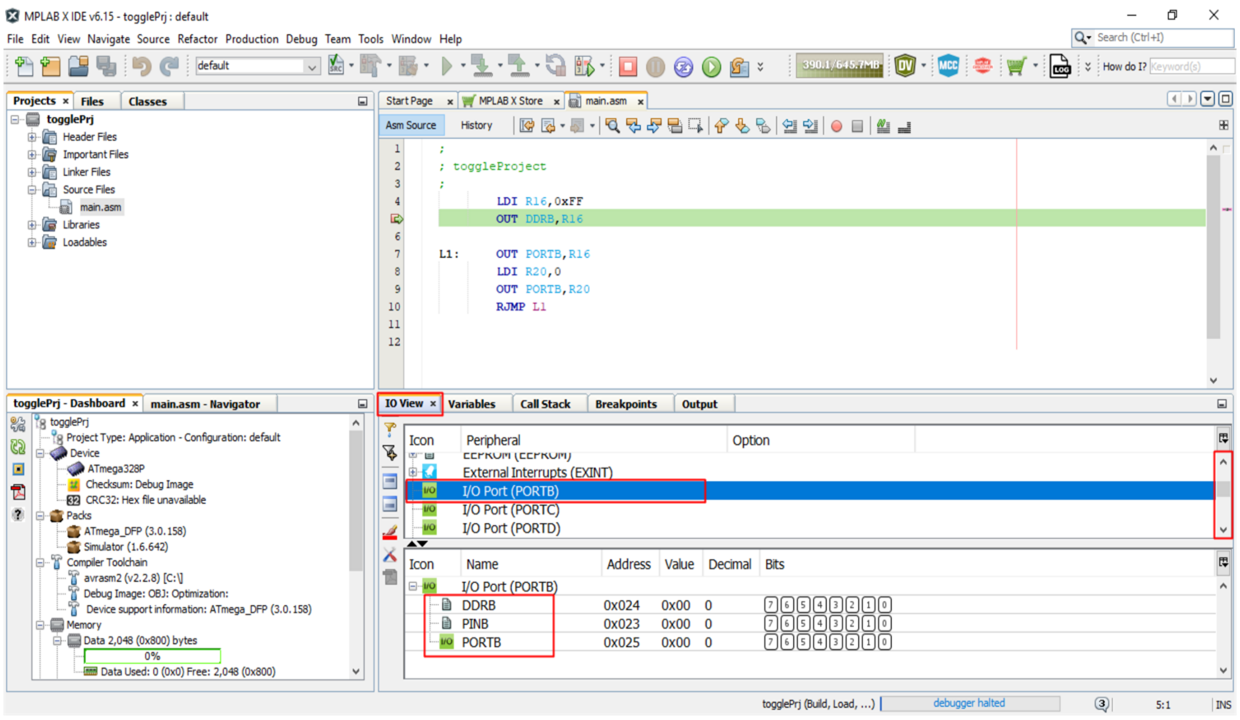
10. To start debugging choose ***Debug Main Project*** from the ***Debug*** menu. Now a green cursor is on the "`OUT DDRB,R16`" instruction of the program and the IDE is ready to debug the program.


Debug
Team
Tools
Window
Help
Debug Main Project
Discrete Debugger Operation
Finish Debugger Session Shift+F5
Pause Ctrl+Alt+8
Continue F5
Step Over F8
Step Into F7
Step Out Ctrl+F7
Step Instruction
Run to Cursor F4
Reset
Set PC at Cursor
Focus Cursor at PC
Stack
Toggle Line Breakpoint Ctrl+F8
New Breakpoint... Ctrl+Shift+F8
New Watch... Ctrl+Shift+F9
New Runtime Watch... Ctrl+Shift+F10
Disconnect from Debug Tool
Run Debugger/Programmer Self Test
Hardware Tool Emergency Boot Firmware Recovery

11. To monitor the peripherals, including the I/O ports, in the ***Window*** menu, choose ***Debugging*** and then click on ***IO View***.



12. To monitor the values of DDRB and PORTB, in the ***IO View*** tab scroll down using the scroll bar and click on ***I/O Port (PORTB)***. The values of the related registers (DDRB, PINB, and PORTB) will be shown below.

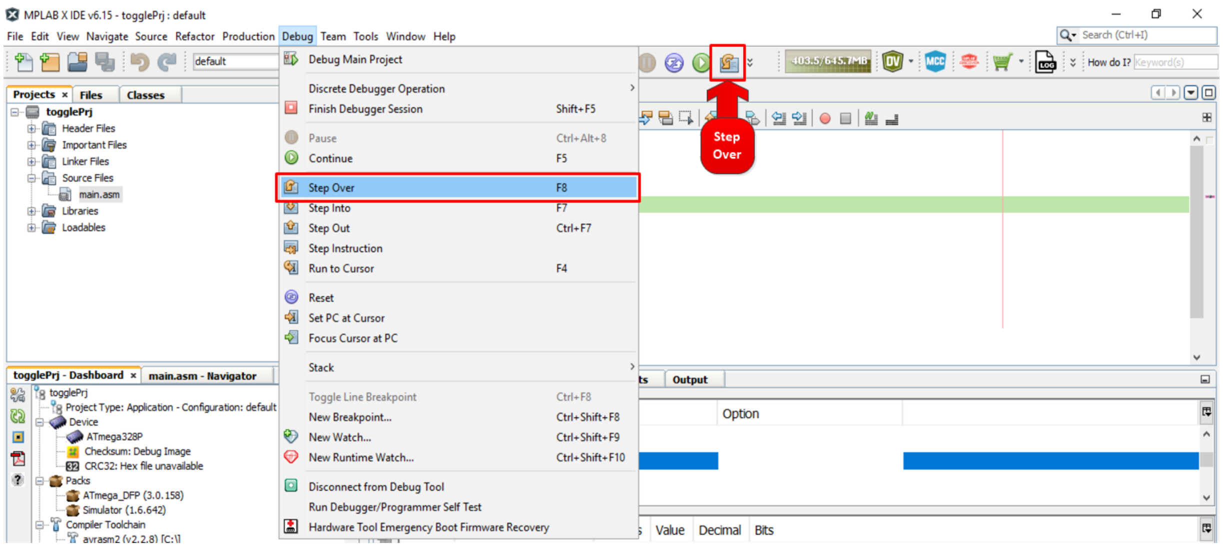


```
;
; toggleProject
;
        LDI R16,0xFF
        OUT DDRB,R16

L1:     OUT PORTB,R16
        LDI R20,0
        OUT PORTB,R20
        RJMP L1
```

| Icon | Name | Address | Value | Decimal | Bits |
|---|---|---|---|---|---|
| | I/O Port (PORTB) | | | | |
| | DDRB | 0x024 | 0x00 | 0 | 7 6 5 4 3 2 1 0 |
| | PINB | 0x023 | 0x00 | 0 | 7 6 5 4 3 2 1 0 |
| | PORTB | 0x025 | 0x00 | 0 | 7 6 5 4 3 2 1 0 |

13. To execute the instructions line by line, press ***F8*** or click on the ***Step Over*** icon or click on ***Step Over*** from the ***Debug*** menu.


Debug Main Project
Discrete Debugger Operation
Finish Debugger Session Shift+F5
Pause Ctrl+Alt+8
Continue F5
Step Over F8
Step Into F7
Step Out Ctrl+F7
Step Instruction
Run to Cursor F4
Reset
Set PC at Cursor
Focus Cursor at PC
Stack
Toggle Line Breakpoint Ctrl+F8
New Breakpoint... Ctrl+Shift+F8
New Watch... Ctrl+Shift+F9
New Runtime Watch... Ctrl+Shift+F10
Disconnect from Debug Tool
Run Debugger/Programmer Self Test
Hardware Tool Emergency Boot Firmware Recovery
Step Over

**Step Into vs. Step Over**

Both ***F8 (Step over)*** and ***F7 (Step into)*** execute one instruction and go to the next instruction. But they work differently when the cursor is on a function call. If the cursor is on the function call, ***Step into*** goes into the first instruction of the function, but ***Step Over*** executes the whole function and goes to the next instruction.

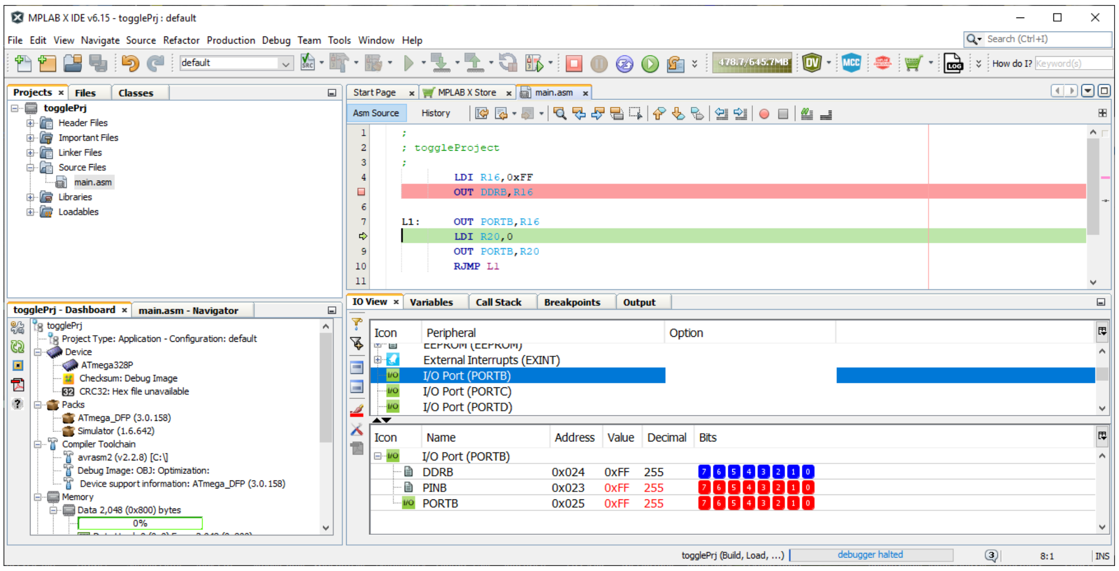
**Step Out**

If the execution is in a function, you can execute the function to the end by pressing the ***Step Out***.

**Run to Cursor**

You can put the cursor on an instruction and then press the Run to Cursor button. In the case, the program runs until it reaches the instruction which the cursor is on it.

14. Press ***F8 (Step Over)*** a few times and see the PORTB register changes in the ***IO View***.



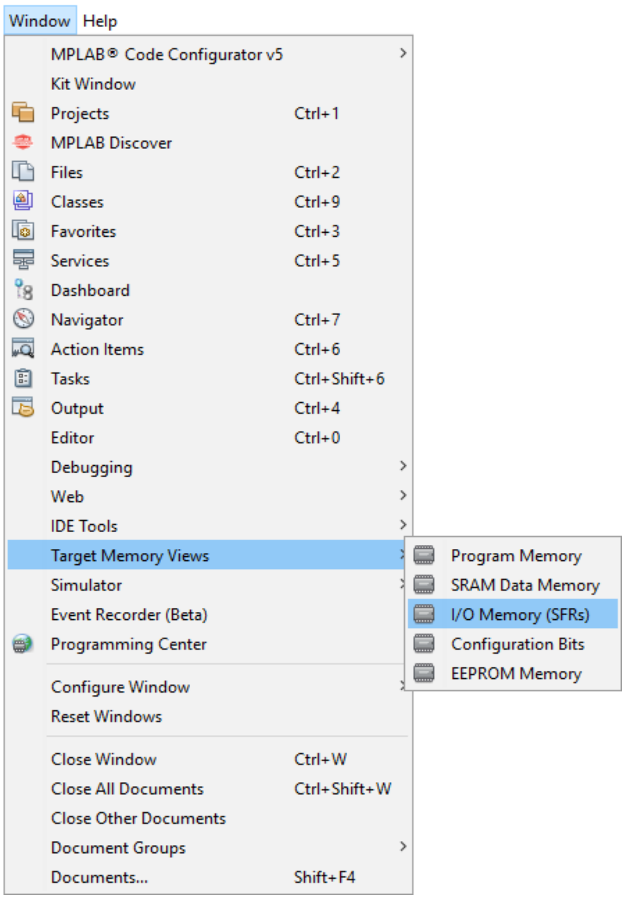
```
;
; toggleProject
;
        LDI  R16,0xFF
        OUT  DDRB,R16

L1:     OUT  PORTB,R16
        LDI  R20,0
        OUT  PORTB,R20
        RJMP L1
```

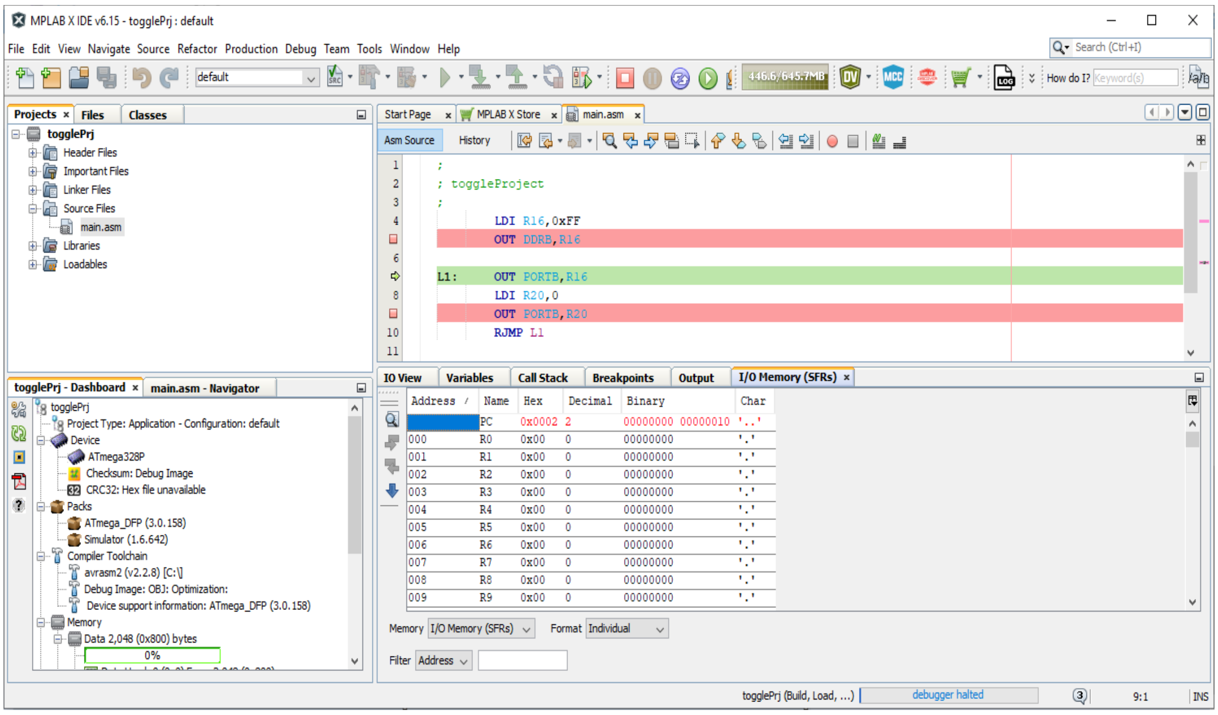
| Icon | Name | Address | Value | Decimal | Bits |
|---|---|---|---|---|---|
| | I/O Port (PORTB) | | | | |
| | DDRB | 0x024 | 0xFF | 255 | 7 6 5 4 3 2 1 0 |
| | PINB | 0x023 | 0xFF | 255 | 7 6 5 4 3 2 1 0 |
| | PORTB | 0x025 | 0xFF | 255 | 7 6 5 4 3 2 1 0 |

## Monitoring the CPU Registers

15. To monitor the CPU registers, including R0-R31 and the Program Counter, choose ***Target Memory Views*** from the ***Window*** menu and click on ***I/O Memory (SFRs)***.



16. In the I/O Memory (SFRs) tab, scroll down to see the value of R16.



```
;
; toggleProject
;
        LDI R16,0xFF
        OUT DDRB,R16

L1:     OUT PORTB,R16
        LDI R20,0
        OUT PORTB,R20
        RJMP L1
```

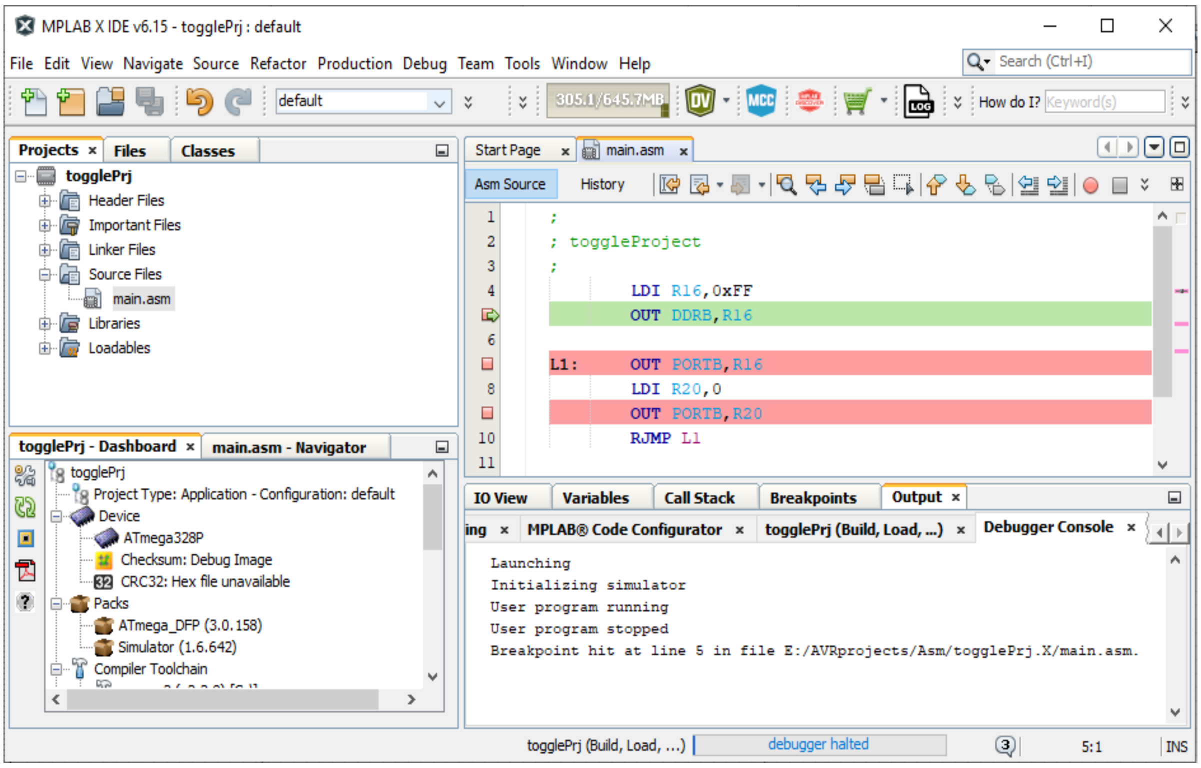
```
togglePrj
  Project Type: Application - Configuration: default
  Device
    ATmega328P
    Checksum: Debug Image
    CRC32: Hex file unavailable
  Packs
    ATmega_DFP (3.0.158)
    Simulator (1.6.642)
  Compiler Toolchain
    avrasm2 (v2.2.8) [C:\]
    Debug Image: OBJ: Optimization:
    Device support information: ATmega_DFP (3.0.158)
  Memory
    Data 2,048 (0x800) bytes
      0%
```

| Address | Name | Hex | Decimal | Binary | Char |
|---|---|---|---|---|---|
|  | PC | 0x0002 | 2 | 00000000 00000010 | '..' |
| 000 | R0 | 0x00 | 0 | 00000000 | '.' |
| 001 | R1 | 0x00 | 0 | 00000000 | '.' |
| 002 | R2 | 0x00 | 0 | 00000000 | '.' |
| 003 | R3 | 0x00 | 0 | 00000000 | '.' |
| 004 | R4 | 0x00 | 0 | 00000000 | '.' |
| 005 | R5 | 0x00 | 0 | 00000000 | '.' |
| 006 | R6 | 0x00 | 0 | 00000000 | '.' |
| 007 | R7 | 0x00 | 0 | 00000000 | '.' |
| 008 | R8 | 0x00 | 0 | 00000000 | '.' |
| 009 | R9 | 0x00 | 0 | 00000000 | '.' |

Memory: I/O Memory (SFRs) Format: Individual
Filter: Address

## Using Breakpoints

17. If you want to debug a portion of a program, add a breakpoint to the beginning of this part of the code and press the run button. The IDE runs the program and when it reaches the breakpoint, it stops running and the green cursor is shown on the breakpoint line. Below, you see the steps in detail.
    a. click on the line number of "`OUT PORTB,R16`" instruction.
    b. click on the line number of "`OUT PORTB,R20`" instruction.
    c. Press ***F5*** or click on ***Continue*** from the ***Debug*** menu. The IDE runs the program until it reaches the breakpoint. Now, you can continue debugging from the breakpoint using the ***Step into*** and ***Step over*** buttons. Press F5 again to run to next breakpoint.
    d. Using ***Finish Debugger Session*** from the ***Debug*** menu, you can stop debugging whenever you want.



```
;
; toggleProject
;
        LDI  R16,0xFF
        OUT  DDRB,R16

L1:     OUT  PORTB,R16
        LDI  R20,0
        OUT  PORTB,R20
        RJMP L1
```

```
Launching
Initializing simulator
User program running
User program stopped
Breakpoint hit at line 5 in file E:/AVRprojects/Asm/togglePrj.X/main.asm.
```