

Embedded Systems

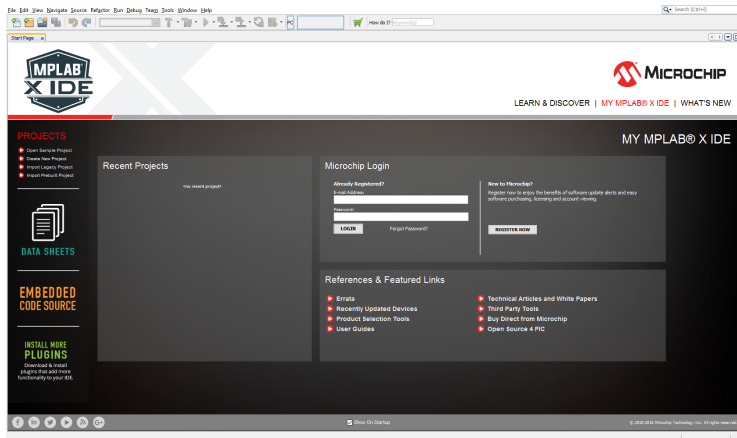
Appendix A2 - Tools for Clicker2 Board

Enrico Simetti

ISME - Interuniversity Research Center on Integrated Systems for Marine Environment
DIBRIS - Department of Computer Science, Bioengineering, Robotics and System Engineering
University of Genova, Italy
enrico.simetti@unige.it

- ▶ **MPLAB X IDE:**
<http://www.microchip.com/mplab/mplab-x-ide>
- ▶ **XC16 Compiler:**
<http://www.microchip.com/mplab/compilers>
- ▶ **MikroE mikroBootloader:**
<https://www.mikroe.com/mikrobootloader>

Creating a new Project I



Creating a new Project II

Steps

1. Choose Project
2. ...

Choose Project

Filter:

Categories:

- Microchip Embedded
 - Other Embedded
 - Samples

Projects:


- Standalone Project
- Existing MPLAB IDE v8 Project
- Prebuilt (Hex, Loadable Image) Project
- User Makefile Project
- Library Project

Description:

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

< Back **Next >** Finish Cancel Help

Creating a new Project III



New Project

Steps

1. Choose Project
- 2. Select Device**
3. Select Header
4. Select Plugin Board
5. Select Compiler
6. Select Project Name and Folder

Select Device

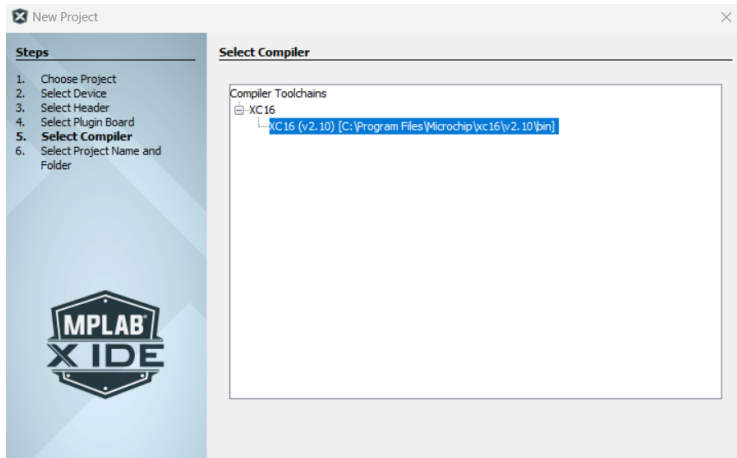
Family: 16-bit DSCs (dsPIC33) ▾

Device: dsPIC33EP512MU810 ▾

Tool: No Tool ▾ ☐ Show All

< Back **Next >** Finish Cancel Help


Creating a new Project IV



Creating a new Project V

Steps

1. Choose Project
2. Select Device
3. Select Header
4. Select Tool
5. Select Plugin Board
6. Select Compiler
- 7. Select Project Name and Folder**



Select Project Name and Folder

Project Name:

Project Location:

Project Folder:

☐ Overwrite existing project.

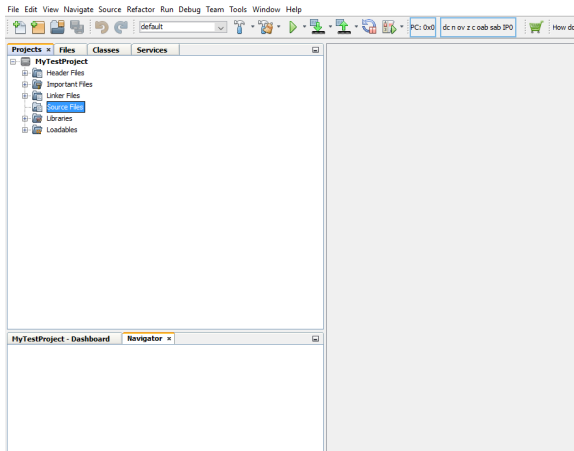
☐ Also delete sources.

☒ Set as main project

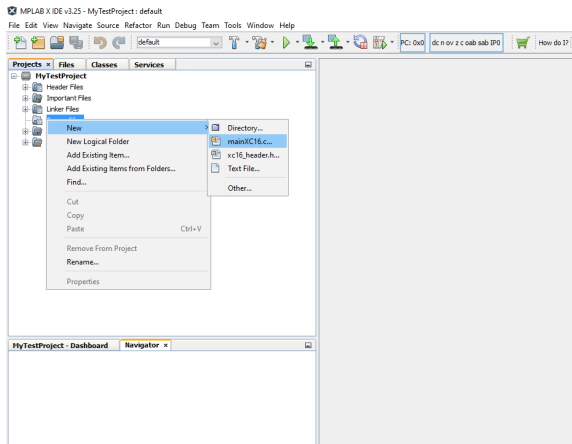
☐ Use project location as the project folder

Encoding:

Creating a new Project VI



Creating a new Project VII



Creating a new Project VIII

Steps

1. Choose File Type
2. **Name and Location**

Name and Location

File Name:

Extension:

☐ Set this Extension as Default

Project:

Folder:

[Browse...](#)

Created File:

< Back

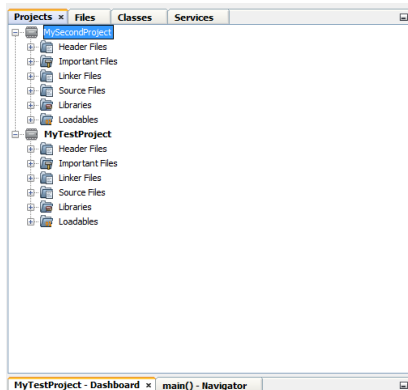
Next >

Finish

Cancel

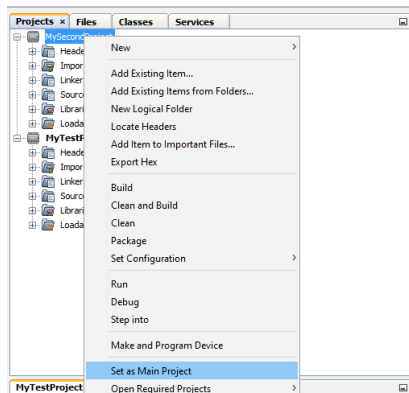
Help

Now the IDE is set to compile “MyTestProject”



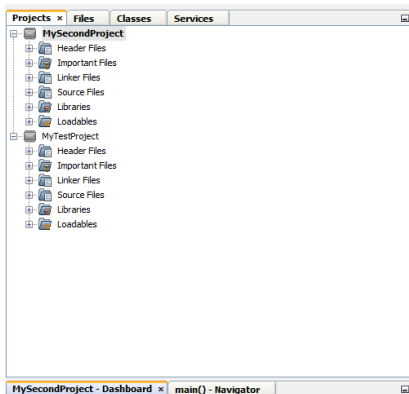
Multiple Projects II

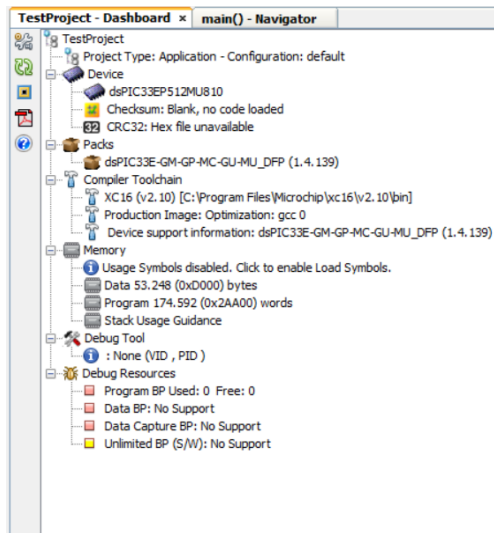
If I want to compile “MySecondProject”, right click and select “Set as Main Project”



Multiple Projects III

Now “MySecondProject” is written in **bold** indicating that it is the main project





Configuration Bits I

Window → Target Memory Views → Configuration bits

Address	Name	Value	Field	Option	Category
F80004	FGS	0003	-	-	-
		1	GWRP	OFF	General Segment Write-Protect bit
		1	GSS	OFF	General Segment Code-Protect bit
		0	GSSK	OFF	General Segment Key bits
F80006	FOSCSEL	0087	-	-	-
		7	FNOSC	FRCDIVN	Initial Oscillator Source Selection Bits
		1	IESO	ON	Two-speed Oscillator Start-up Enable bit
F80008	FOSC	00E7	-	-	-
		3	POSCMD	NONE	Primary Oscillator Mode Select bits
		1	OSCIOFNC	OFF	OSC2 Pin Function bit
		1	IOL1WAY	ON	Peripheral pin select configuration
		3	FCKSM	CSDCMD	Clock Switching Mode bits
F8000A	FWDI	00FF	-	-	-
		F	WDTPRST	PS32768	Watchdog Timer Postscaler Bits
		1	WDTPRE	PR128	Watchdog Timer Prescaler bit
		1	PLLKEN	ON	PLL Lock Wait Enable bit
		1	WINDIS	OFF	Watchdog Timer Window Enable bit
		1	FWDTEN	ON	Watchdog Timer Enable bit
F8000C	FPOR	003F	-	-	-
		7	FPWRT	PWR128	Power-on Reset Timer Value Select bits
		1	BOREN	ON	Brown-out Reset (BOR) Detection Enable bit
		1	ALTI2C1	OFF	Alternate I2C pins for I2C1
		1	ALTI2C2	OFF	Alternate I2C pins for I2C2
F8000E	FICD	00D7	-	-	-
		3	ICS	PGD1	ICD Communication Channel Select bits
		1	RSTPRI	PF	Reset Target Vector Select bit
		0	JTAGEN	OFF	JTAG Enable bit
F80010	FAS	0003	-	-	-
		1	AWRP	OFF	Auxiliary Segment Write-protect bit
		1	APL	OFF	Auxiliary Segment Code-protect bit
		0	APLK	OFF	Auxiliary Segment Key bits

Disable the watchdog!

Set Oscillator source = Primary Oscillator

Set Primary oscillator = XT

Oscillator Source:

- ▶ if set to Internal Fast RC: $F_{osc} = 8 \text{ MHz}$ approximately
- ▶ if set to Primary Oscillator: then the Primary Oscillator Mode option determines F_{osc}

Primary Oscillator Mode:

- ▶ Only considered if Oscillator Source = Primary Oscillator
- ▶ can be used to select the XT source (external) with PPL options
- ▶ can be used to select the FRC with PLL options

With Primary Oscillator = XT, $F_{osc} = 8 \text{ MHz}$, directly from the Oscillator X1 on the Clicker2 board

After setting the bits, press “Generate source code to output” and then cut and paste to the main.c file

Example configuration bits

```
// FGS
#pragma config GWRP = OFF           // General Segment Write-Protect bit (General Segment may be written)
#pragma config GSS = OFF            // General Segment Code-Protect bit (General Segment Code protect is disabled)
#pragma config GSSK = OFF           // General Segment Key bits (General Segment Write Protection and Code Protection is Disabled)

// FOSCSEL (Primary Oscillator, 8MHz quartz on pin OSC1)
#pragma config FNOOSC = PRIPLL
#pragma config IESO = ON

// FOSC (Primary)
#pragma config POSCMD = XT // 3 to 10 MHz
#pragma config OSCIOFNC = OFF
#pragma config IOL1WAY = ON
#pragma config FCKSM = CSDCMD

// FWDT
#pragma config WDTPRST = PS32768    // Watchdog Timer Postscaler Bits (1:32,768)
#pragma config WDTPRE = PR128       // Watchdog Timer Prescaler bit (1:128)
#pragma config PLLKEN = ON           // PLL Lock Wait Enable bit (Clock switch to PLL source will wait until the PLL lock signal is
valid.)
#pragma config WINDIS = OFF          // Watchdog Timer Window Enable bit (Watchdog Timer in Non-Window mode)
#pragma config FWDTEN = OFF          // Watchdog Timer Enable bit (Watchdog timer enabled/disabled by user software)
```

Example configuration bits II

```
// FPOR
#pragma config FFWRT = PWR128 // Power-on Reset Timer Value Select bits (128ms)
#pragma config BOREN = ON // Brown-out Reset (BOR) Detection Enable bit (BOR is enabled)
#pragma config ALTI2C1 = OFF // Alternate I2C pins for I2C1 (SDA1/SCK1 pins are selected as the I/O pins for I2C1)
#pragma config ALTI2C2 = OFF // Alternate I2C pins for I2C2 (SDA2/SCK2 pins are selected as the I/O pins for I2C2)

// FICD
#pragma config ICS = PGD1 // ICD Communication Channel Select bits (Communicate on PGEC1 and PGED1)
#pragma config RSTPRI = PF // Reset Target Vector Select bit (Device will obtain reset instruction from Primary flash)
#pragma config JTAGEN = OFF // JTAG Enable bit (JTAG is disabled)

// FAS
#pragma config AWRP = OFF // Auxiliary Segment Write-protect bit (Auxiliary program memory is not write-protected)
#pragma config APL = OFF // Auxiliary Segment Code-protect bit (Aux Flash Code protect is disabled)
#pragma config APLK = OFF // Auxiliary Segment Key bits (Aux Flash Write Protection and Code Protection is Disabled)

#include <xc.h>

int main(void) {
    // all analog pins disabled
    ANSELA = ANSELB = ANSELC = ANSELD = ANSELE = ANSELG = 0x0000;

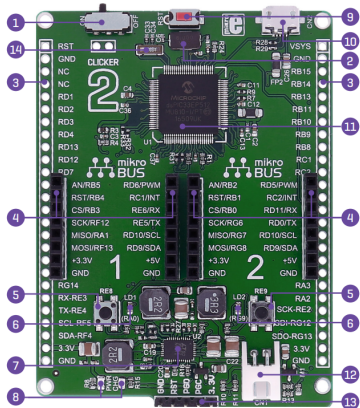
    return 0;
}
```

If you don't need to use analog pins, Be sure to configure them properly.
Setting ANSELx to 0 guarantees that they are setup as digital

mikroE Clicker 2 board for dspic33

Key features

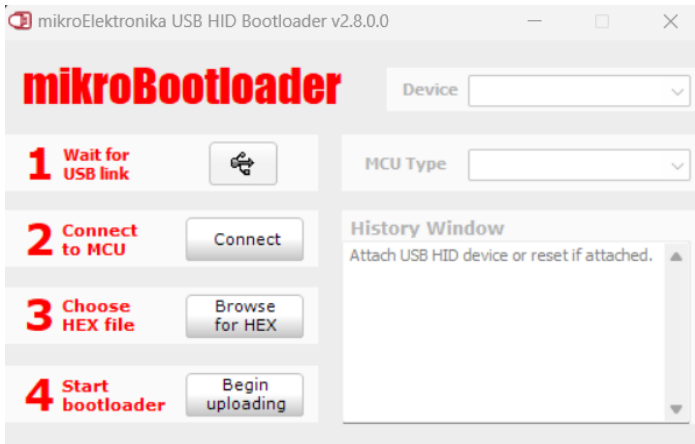
- 1 ON/OFF switch
- 2 8 MHz crystal oscillator
- 3 two 1x26 connection pads
- 4 mikroBUS™ sockets 1 and 2
- 5 Pushbuttons
- 6 Additional LEDs
- 7 LTC3586 USB power manager IC
- 8 Power and Charge indication LEDs
- 9 RESET button
- 10 Micro USB connector
- 11 dsPIC33EP512MU810 MCU
- 12 Li-Polymer battery connector
- 13 mikroProg programmer connector
- 14 32.768 KHz crystal oscillator



How to program? (Windows)

The Clicker2 board contains a *bootloader*, a light program stored permanently in memory that allows to flash firmware without a specific programmer. The mikroE *mikroBootloader* software is required to download the firmware on the board.

It's only available on Windows platforms.



How to program? (Windows)

To write your program on the board use the following steps:

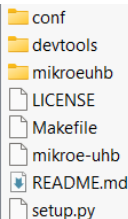
- ▶ Connect the board to your PC with the microUSB cable
- ▶ Press the RESET button (the red one) on the board
- ▶ the icon besides *Wait for the USB link* should become red. You have approx. 5 seconds to press the connect button on the interface. If you don't press it within 5 seconds the board will run the previously flashed firmware
- ▶ Once you're connected, select *Browse for HEX* and look for the following path:
C:/Users/yourusername/MPLABXProjects/projectname.X/dist/default/production/projectname.X.production.hex. You generate this file every time you build your project in MPLAB.
- ▶ Click on *Begin uploading* and wait until the firmware is written on the board. After 5 seconds the board will execute your code

How to program? (Ubuntu)

To write your program on the board from a Ubuntu distro you must install an additional tool:

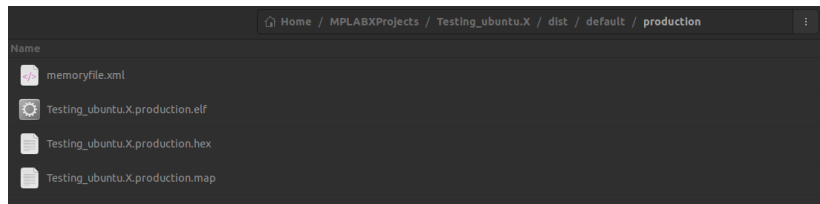
<https://github.com/thotypous/mikroe-uhb>

Clone the *master* repository from the github link, the folder content will be the following:



Open a terminal in the main folder and run *make* and then *sudo make install* commands. Note that **you must have a Python installation on your system**. If you encounter issues in building the library, run *sudo pip install "setuptools<58.0.0"*

To build your project on Ubuntu just follow the same steps shown in the Windows tutorial, then open a terminal in the folder containing the *.hex* file.



How to program? (Ubuntu)

Connect the board to the PC and run *sudo lsusb*: you need this step to know how the board is identified.

The output should look like follows:

```
sudo lsusb
Bus 004 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 003 Device 002: ID 0c45:6a1b Microdia Integrated_Webcam_FHD
Bus 003 Device 004: ID 046d:c03e Logitech, Inc. Premium Optical Wheel Mouse (M-BT58)
Bus 003 Device 003: ID 8087:0033 Intel Corp.
Bus 003 Device 016: ID 2dbc:0001 Mikroelektronika USB HID Bootloader
Bus 003 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

You can notice that the Bootloader stored on the board is recognized as *Mikroelektronika USB HID Bootloader*. If that record does not appear in the listing, press the reset button on the board and re-run the *lsusb* command within 5 seconds from the botton toggle.

Take note of the ID values in the listing, in this case *2dbc and 0001*.

How to program? (Ubuntu)

Now you can run the flashing command, remember that this is just an example and the *vendor id* and *product id* could be different on your machine.

```
francesco@francesco-Vostro-7620:~/MPLABXProjects/Testing_ubuntu.X/dist/default/production$ sudo mikroe-uhb -v --vend
or=0x2dbc --product=0x0001 Testing_ubuntu.X.production.hex
[sudo] password for francesco:
DEBUG:mikroeuhb.hid.linux:opening device vendor=2dbc, product=1
INFO:mikroeuhb.hid.linux:USB device 2dbc:0001 plugged
INFO:mikroeuhb.hid.linux:USB ID matches the expected one
DEBUG:mikroeuhb.device:send cmd: stx, cmd=INFO, addr=0x00000000, counter=0x0000
DEBUG:mikroeuhb.device:recv data: 3c010b0008000000408000300000c040080010500001306000040050007436c69636b6572203220666f
72206473504943333000000000000000000000549526cd
ERROR:mikroeuhb.bootinfo:Field 51 not recognized -- aborting parsing
McUType: 'DSPIC33'
EraseBlock: 0xc00
WriteBlock: 0x180
BootRev: 0x1300
BootStart: 0x54000
DevDsc: b'Clicker 2 for dsPIC3'
McUSize: 0x80400

DEBUG:mikroeuhb.device:send cmd: stx, cmd=BOOT, addr=0x00000000, counter=0x0000
DEBUG:mikroeuhb.device:recv cmd: stx, cmd=BOOT, addr=0x0008000b, counter=0x0400
DEBUG:mikroeuhb.device:send cmd: stx, cmd=SYNC, addr=0x00000000, counter=0x0000
DEBUG:mikroeuhb.device:recv cmd: stx, cmd=SYNC, addr=0x0008000b, counter=0x0400
DEBUG:mikroeuhb.devkit:reset code before fix: 000204000000
DEBUG:mikroeuhb.devkit:reset code after fix: 004004050000
DEBUG:mikroeuhb.devkit:transfer to device starting
```

At this point the terminal will hang in loop until the on-board bootloader is detected, showing an *opening device* message. Just press again the reset button and the flashing process will begin. The output should be similar to the following screen.