



# FEZ Tutorial

Rev.1.03

March 12, 2010

step-by-step guide

**Don't worry! It is...**



Brought to you by



with the



Technology

**Copyright © 2010 GHI Electronics, LLC**

[www.ghielectronics.com](http://www.ghielectronics.com)

## Table of Contents

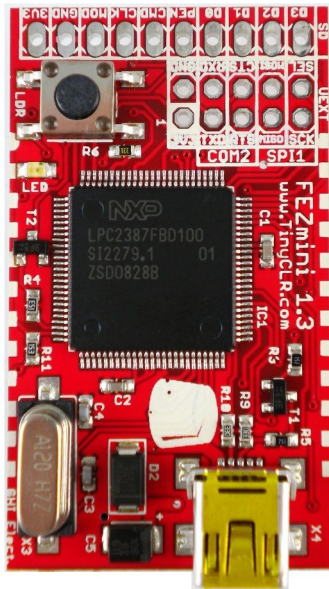
1.Introduction.....	3
The objective of this Guide.....	4
2.Getting Started.....	5
2.1.System Setup.....	5
2.2.The Emulator.....	6
Create a Project.....	6
Selecting Transport Interface.....	8
Executing (Deploying) the project.....	9
Debugging Breakpoints.....	11
2.3.Running on Hardware.....	12
MFDeploy can Ping!.....	12
Deploying to Hardware.....	13
Connect MFDeploy to show debugging messages.....	14
Checking USBizi firmware Version.....	15
2.4.Deleting The Deployed Project.....	15
Emergency user application code deletion.....	16
3.Assemblies with FEZ Project.....	17
3.1.What are Assemblies?.....	17
Standard Assemblies.....	17
FEZ (USBizi) assemblies.....	17
FEZ Platform Selector Assembly.....	18
3.2.How to add an assembly to Visual C# Project?.....	18
4.FEZ Hardware Components.....	21
Shields and Extensions.....	27
5.Firmware Update.....	29
5.1.Checking the Firmware Version.....	30
5.2.System Setup and Accessing the Boot Loader.....	32
5.3.Boot loader Commands.....	34
5.4.Simple Steps to update the firmware.....	34
6.What is next?.....	37

### Document Information

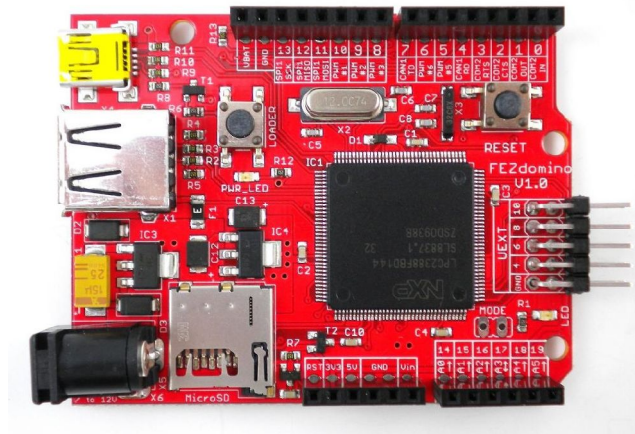
Information	Description
Abstract	This document provides a simple step-by-step tutorial for FEZ boards.

# 1. Introduction

FEZ Mini and FEZ Domino are board based on USBizi chipset running the Microsoft .NET Micro Framework at the core. FEZ is targeted for electronics beginners, hobbyists and even professionals. It is a solution for both hardware and software in one package for easy fast start with .NET Micro Framework and embedded systems.



FEZ Mini



FEZ Domino

With Micro framework, developers can write code much more efficiently using the C# programming language within the free Microsoft Visual C# Express.

Many libraries are available for FEZ hardware and software features including FAT file system, threading, UART, SPI, I2C, GPIO, PWM, ADC, DAC and much more.

Thanks to the USBizi at it's core, there are many advantages for using FEZ over Arduino, or any other similar products like BASIC STAMP:

1. Lowest cost at available features!
2. Runs Microsoft's .NET Micro Framework 4.0.
3. Uses Free Visual C# 2008 express.

4. Runtime debugging over USB or serial.
5. Program in modern managed language.
6. 32-bit ARM processor.
7. FAT file system for storage on SD cards and USB memory devices.
8. Easy upgrades to high end systems like ChipworkX or Embedded Master. Use same code and knowledge!
9. The FEZ core, USBizi, is widely used in commercial applications around the world.

### **The objective of this Guide**

This guide will help you in the first steps into the embedded devices world. From here, we will show you how to connect FEZ, make sure it is running, load a simple program, and explain how to use the FEZ components and their included C# drivers.

This guide only covers very basic points, for full details use the eBook.

## 2. Getting Started

### 2.1. System Setup

Before we try anything, we want to make sure the PC is setup with needed software. First download and install [Visual C# express 2008 with SP1](#).

Now, download and install [.NET Micro Framework 4.0 SDK](#) (not the porting kit).

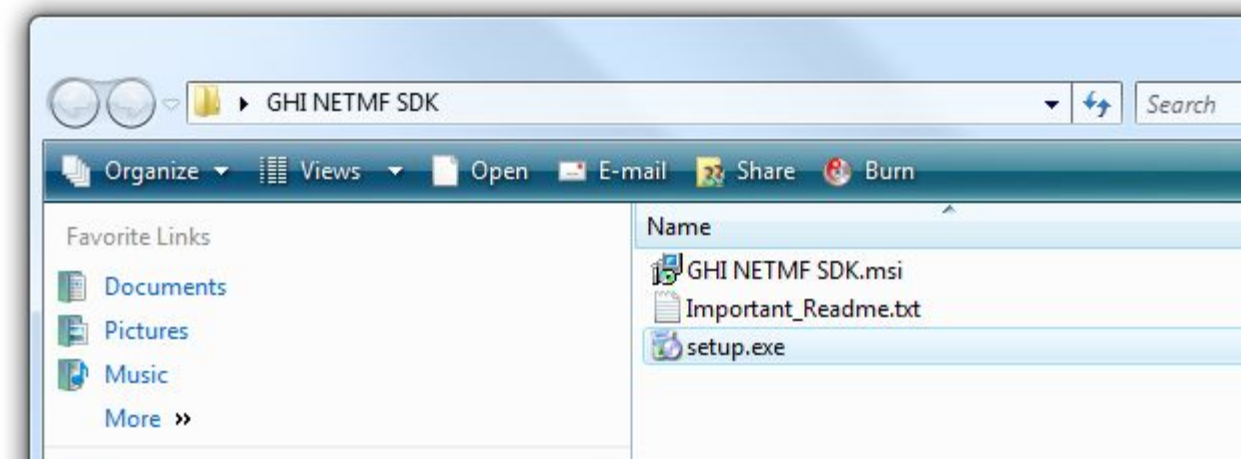
if link above didn't work, search for ".NET Micro Framework 4.0 SDK".

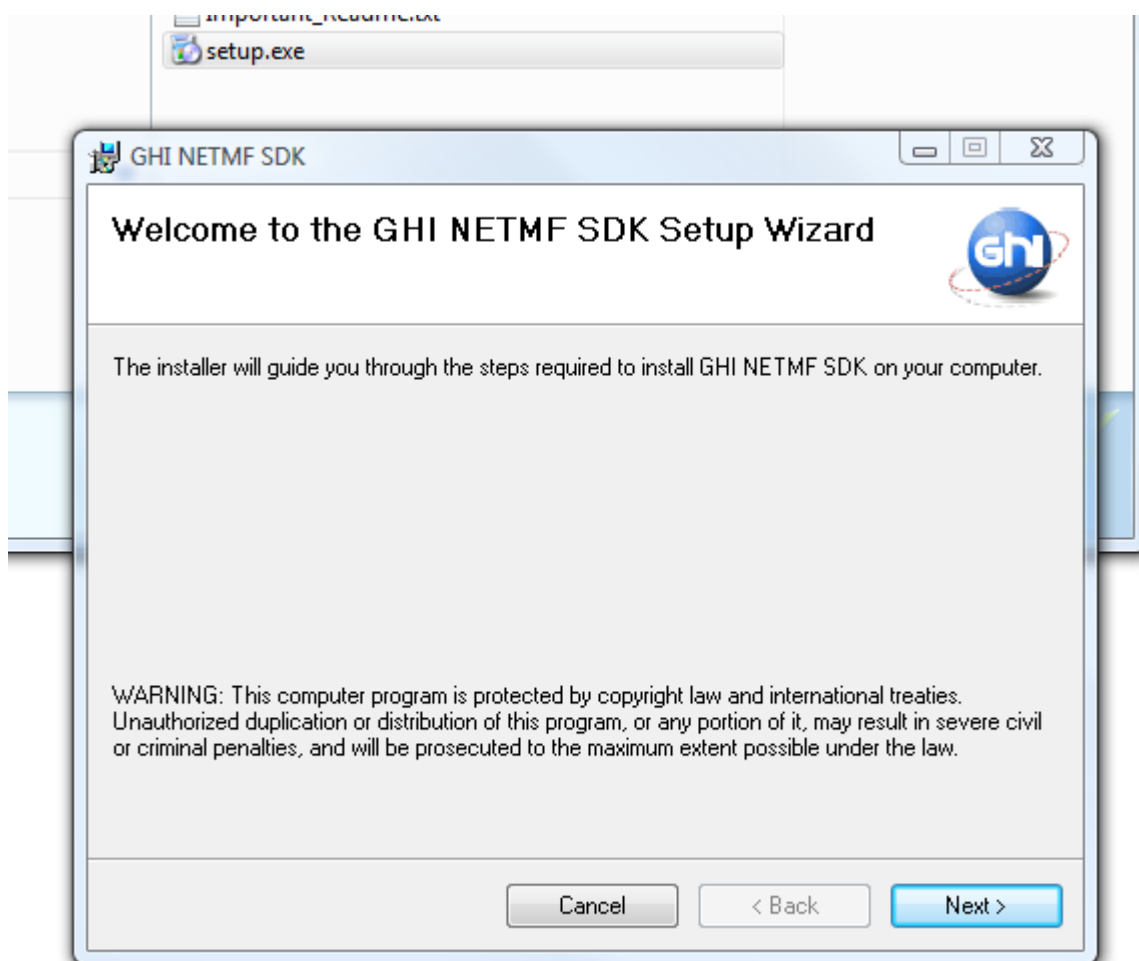
Finally, install the GHI NETMF SDK (Software Development Kit) from [www.TinyCLR.com](http://www.TinyCLR.com)

This is the direct link to the SDK

<http://www.ghielectronics.com/downloads/NETMF/GHI%20NETMF%20SDK.zip>

The SDK comes in a zip file, extract it and then run setup.exe to install the SDK.



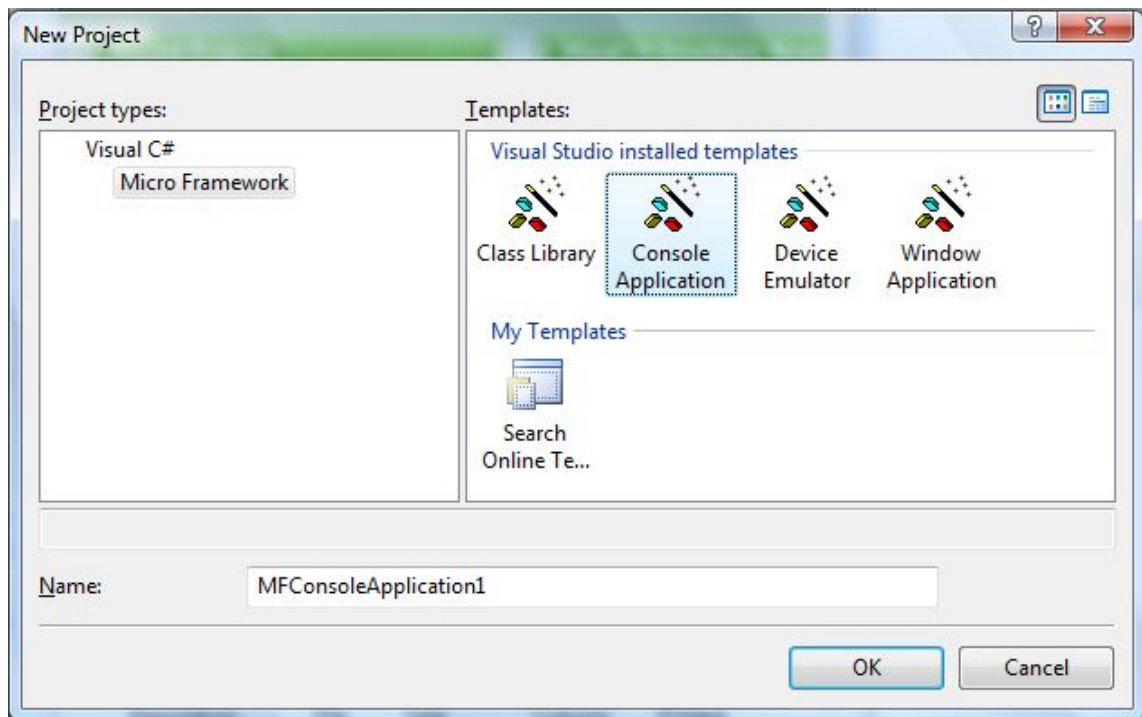


## 2.2. The Emulator

NETMF includes an emulator that allows running application right on the PC. For our first project, we will use the emulator to run a very simple application.

### Create a Project

Open Visual C# express and, from the menu, select file -> New Project. The wizard now should have "Micro Framework" option in the left menu. Click on it, and from the templates, select "Console Application"



Click the “OK” button and you will have a new project that is ready to run. The project has only one C# file, called Program.cs, which contains a few lines of code. The file is shown in “Solution Explorer” window. If this window is not showing then you can open it by clicking “View->Solution Explorer” from the menu.

```
using System;
using Microsoft.SPOT;

namespace MFConsoleApplication1
{
    public class Program
    {
        public static void Main()
        {
            Debug.Print(
                Resources.GetString(Resources.StringResources.String1));
        }
    }
}
```

For simplicity change the code to make it look like the listing below.

```
using System;
using Microsoft.SPOT;

public class Program
{
    public static void Main()
    {
        Debug.Print("Amazing!");
    }
}
```

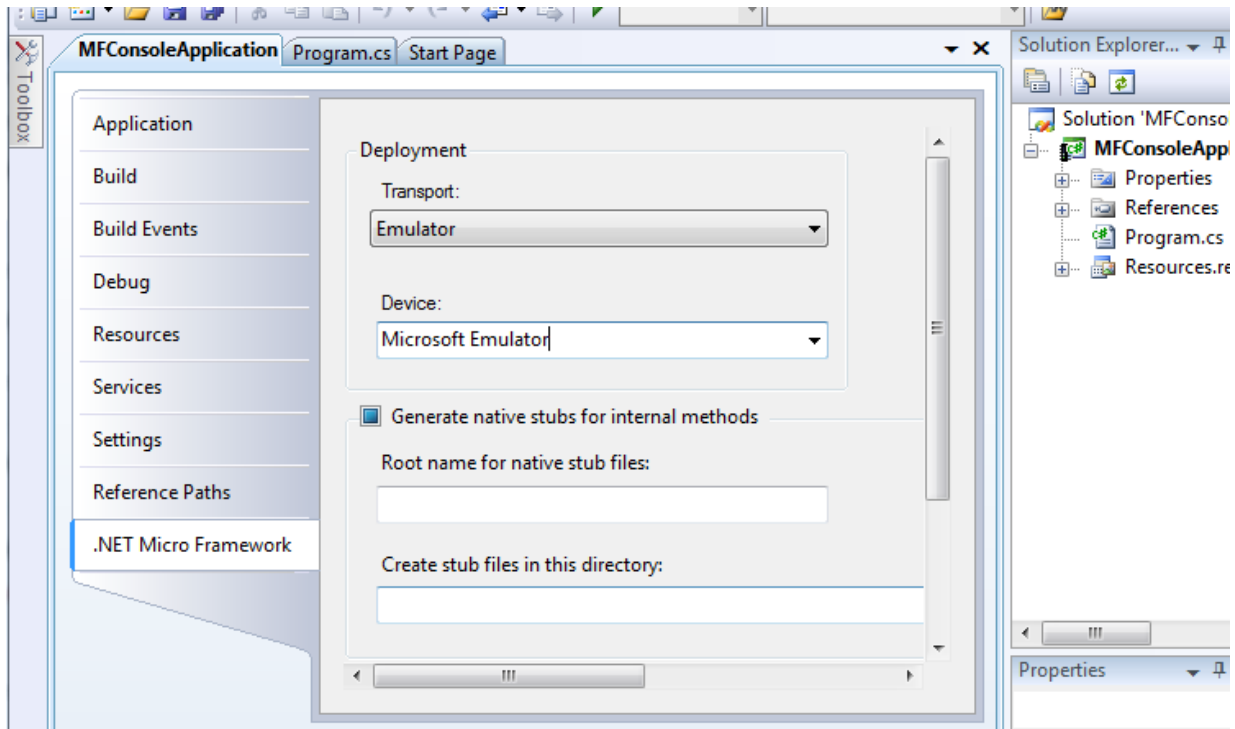
### Selecting Transport Interface

Do not worry if you do not understand the code. We will explain it later. For now, we want to run it on the emulator. Let us make sure you have everything setup properly. Click on "Project->Properties" from the menu. In the new showing window, we want to make sure we select the emulator. On the left side tabs, select ".NET Micro Framework" and make sure the window looks like the image below.

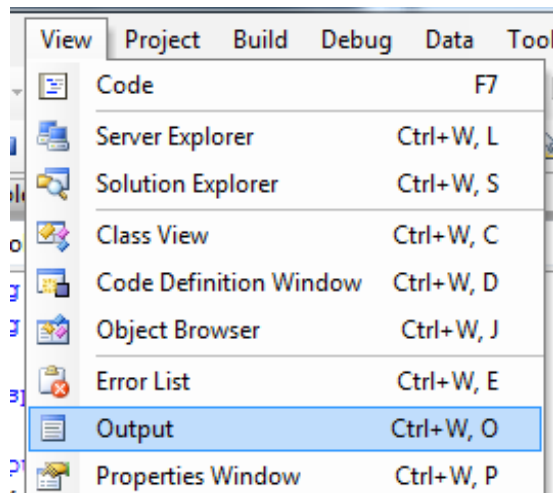
Transport: Emulator

Device: Microsoft Emulator





Make sure the output window is visible, click on “View->Output”



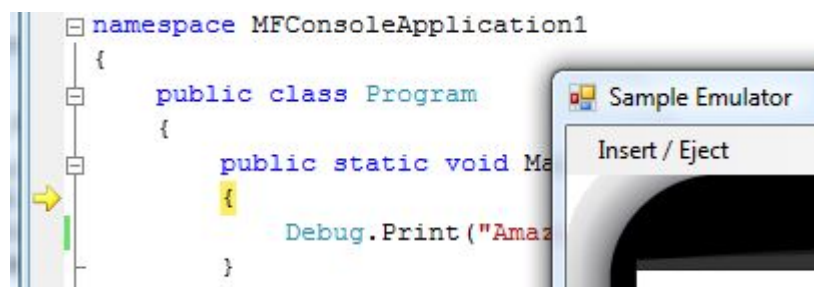
### Executing (Deploying) the project

Finally, we are ready to run our first application. Press F5 key on the computer. This is a

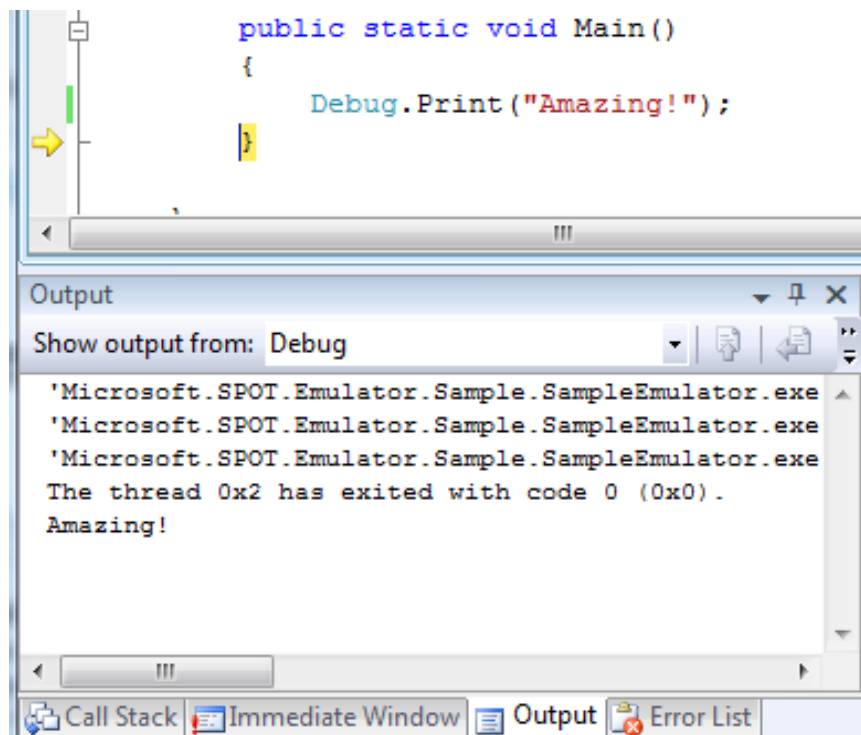
very useful shortcut and you will be using it a lot to run your applications. After you press F5, the application will be compiled and loaded on the emulator, and in couple seconds everything will stop! That is because our program had finished execution so fast that we didn't see much.

We want to “debug” the code now. Debugging means that you are able to step in the code and see what it is doing. This is one of the greatest values of NETMF.

This time use F11 instead of F5, this will “step” in the application instead of just running it. This will deploy the application on the emulator and stop at the very first line of the code. This is indicated by the yellow arrow.



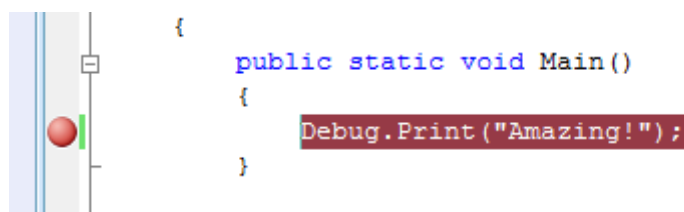
C# applications always start from a method called Main, and this is where the arrow had stopped. Press F11 again and the debugger will run the next line of code, which is the line you changed before. You probably have guessed it right, this line will print “Amazing!” to the debug window. The debug window is the output window on Visual C# express. Make sure Output window is visible like explained earlier and press F11 one more time. Once you step on that line, you will see the word Amazing! Showing in the output window.



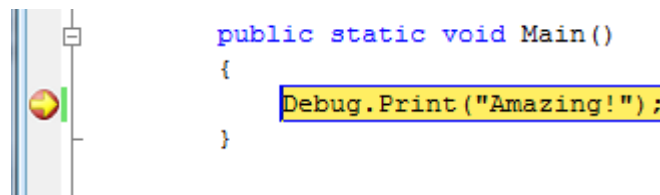
The Program that you've just deployed is called "Managed Code" in .NET Micro Framework Terminology.

## Debugging Breakpoints

Breakpoints are another useful feature when debugging code. While the application is running, the debugger checks if execution has reached a breakpoint. If so, the execution will pause. Click the bar right to the left of the line that prints "Amazing!". This will show a red dot which is the breakpoint.



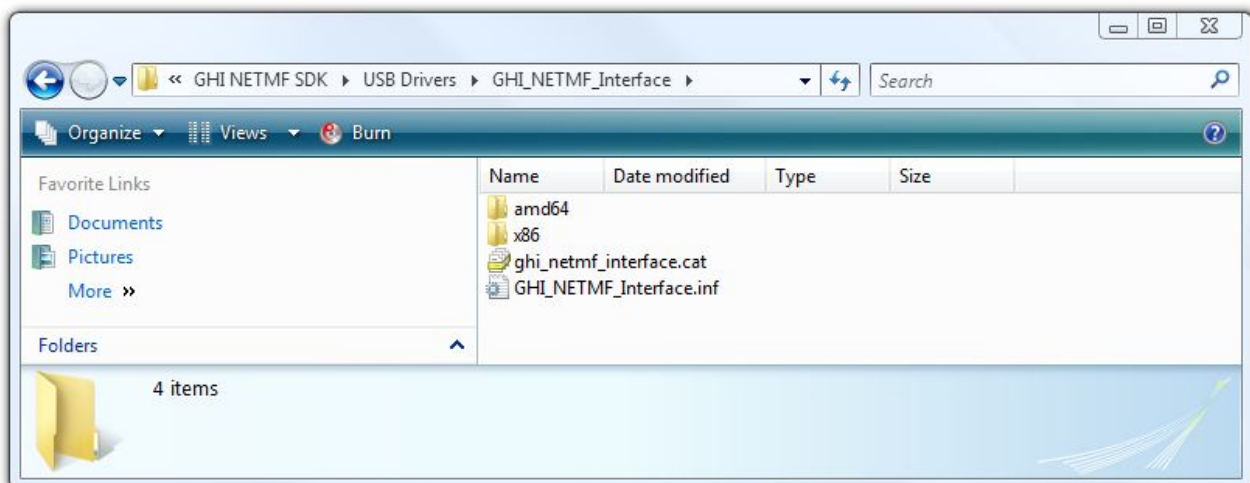
Now press F5 to run the software and when the application reaches the breakpoint the debugger will pause it as showing in the image below



Now, you can step in the code using F11 or continue execution using F5.

## 2.3. Running on Hardware

We are now ready to connect FEZ to the PC. Connect FEZ (Domino or Mini) using the USB cable. No power is needed as FEZ uses USB for power. If this is the first time you plug FEZ to your PC then windows will ask for drivers. Direct windows to the GHI NETMF interface driver available in GHI NETMF SDK under USB Drivers Folder.



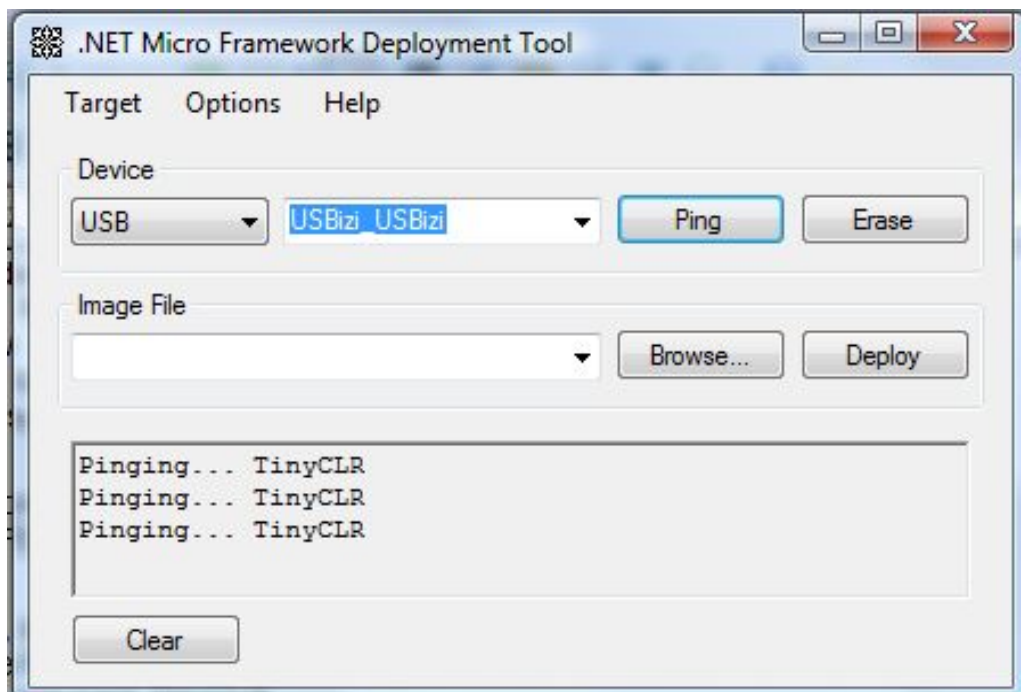
### MFDeploy can Ping!

Before we use the hardware, let us make sure it is properly connected. The NETMF (.NET Micro Framework) SDK comes with a software from Microsoft called MFDeploy. There are many good uses for MFDeploy but for now we only need it to “ping” the device. Basically, “ping” means MFDeploy will say “Hi” to the device and then checks if the device will respond with “Hi”. This is to make sure the device is connected properly and communication has no issues.

Open MFDeploy and connect FEZ using the included USB cable to your PC. If this is the first time you plug in FEZ, Windows will ask for drivers. Supply the driver from the SDK folder and wait until windows is finished.

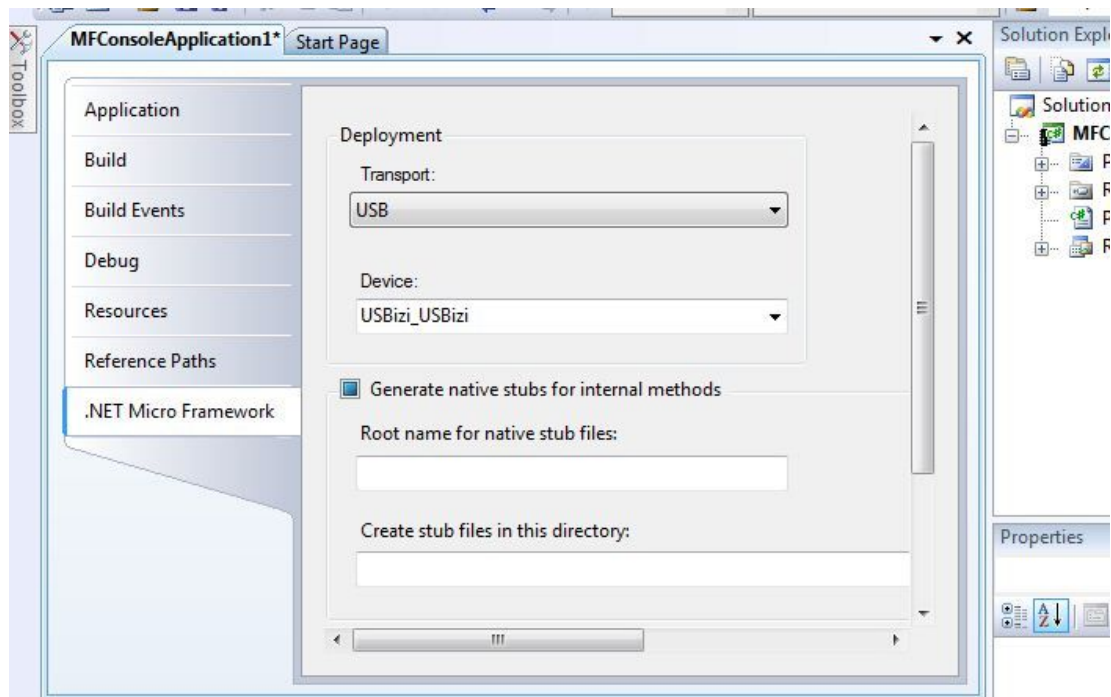
In the drop-down menu, select USB. You should see USBizi showing in the device list.

You will see USBizi because FEZ is based on USBizi chipset. Select USBizi and click the “Ping” button. You should now see back TinyCLR.



## Deploying to Hardware

Now that we checked the hardware is working using MFDeploy, we need to go back to Visual C# express. From the project properties, select USB for transport and USBizi for the device. Make sure your setup looks similar to the image below.



Allow a couple seconds to ensure the hardware has completed the boot up process then press F5, we will now send our simple application to FEZ and it will run right inside the real hardware. Switching from emulator to real hardware is that simple!

Try the steps we tried with the emulator, like setting breakpoints and using F11 to step in the code. Note that "Debug.Print" will still forward the debug messages from the hardware to the output window on Visual C# express.

You have to keep in mind that only one software can talk to USBizi interface at the same time. In another word, you can not Ping the hardware through MFDeploy if it is already connected through Visual Studio.

### Connect MFDeploy to show debugging messages

After Deploying the application through Visual Studio. the application still works on FEZ even if is Visual Studio was disconnected and it will run every time you reset or recycle power on FEZ.

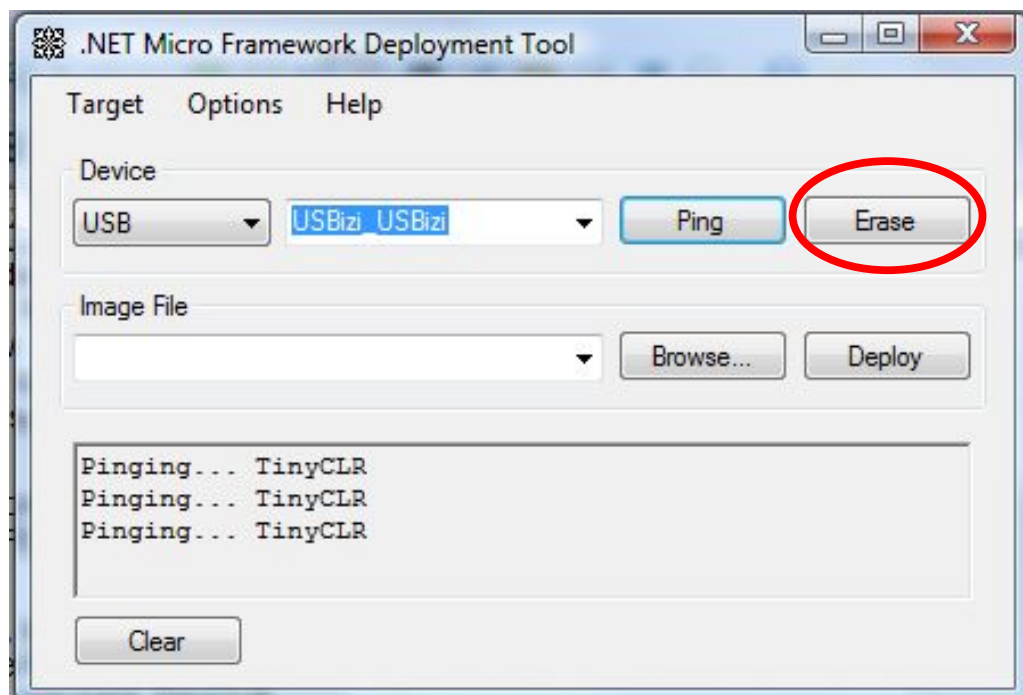
User can still view the boot up and debugging messages by connecting MFDeploy to USB. Simply after you connect USB cable and you can ping FEZ in MFDeploy, click Target->Connect.

After this you can even reset the board and see all the boot up sequence messages and any debugging messages including the strings printed out in the code using Debug.Print(). In our previous example,"Amazing!". When done, you can disconnect



response when pinging

Then click the erase button:



### Emergency user application code deletion

In some cases, user needs to delete the application because it is locking up the device or blocking the USB debugging interface. but Visual C# or MFDeploy tool does not help here and can not communicate with the device.

In this case you should access GHI boot loader and execute the delete command **D**. this command deletes only user application code. then reset or recycle power and FEZ is ready again to communicate with MFDeploy or Visual C#.

[Firmware Update section](#) includes some easy steps on how to access GHI boot loader and process commands.



## 3. Assemblies With FEZ Project

---

### 3.1. What are Assemblies?

Hardware/Software features are accessed through Managed code using C# Classes with their members such as methods, properties and variables.

On FEZ you may need Assemblies from 3 sources:

#### Standard Assemblies

.NET Micro Framework SDK includes pre-compiled assemblies that contain the managed code of various classes for hardware features such as Debug Class that you've just used in the example to print the string "Amazing!" using the Print Method. Users do not see the source code of these classes but can use these classes by simply adding the relevant assemblies to the Visual C# project.

.NET Micro Framework library examples:

- Microsoft.SPOT.Hardware
- Microsoft.SPOT.Native
- System.IO

Classes are described is in .NET Micro Framework SDK documentation which also includes information about the required assemblies.

#### FEZ (USBizi) assemblies

USBizi core-chip adds more functionality on top of the .NET Micro Framework. For example, hardware peripherals such as CAN, Analog converters, PWM,...etc are not directly supported in .NET Micro Framework. However, they can be easily used with FEZ boards.

The assembly provides access to hardware peripherals, storage solutions, USB connectivity, and many other features. The assembly files and their documentation are included with GHI NETMF SDK. These assembly files must be added to your Visual Studio project in order to be able to use the extra features.

USBizi (FEZ) uses the same powerful assemblies available for ChipworkX and EMX. This also means, you can switch from FEZ to USBizi to EMX to ChipworkX without the need to change any line of code!

FEZ (USBizi) library examples:

- GHIElectronics.NETMF.Hardware
- GHIElectronics.NETMF.System

### FEZ Platform Selector Assembly

Every FEZ board type has a unique library that the user must add in every project. This assembly has the required definitions that lets the user access the features much easier, specially with hardware “components drivers”.

FEZ Domino library: **FEZDomino\_GHIElectronics.NETMF.FEZ**

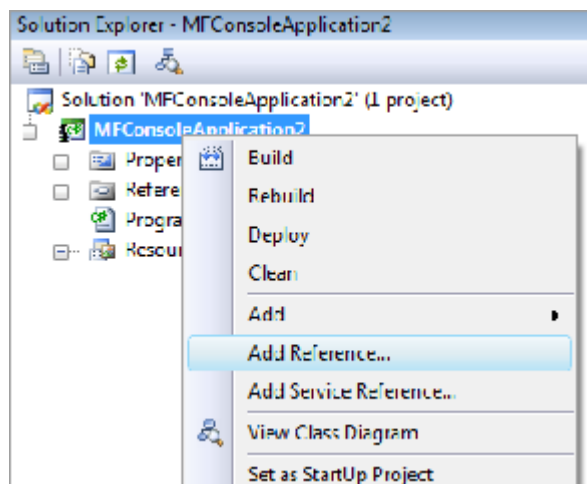
FEZ Mini library: **FEZMini\_GHIElectronics.NETMF.FEZ**

An example on how to use these libraries is explained later in the FEZ Hardware Components section more thoroughly.

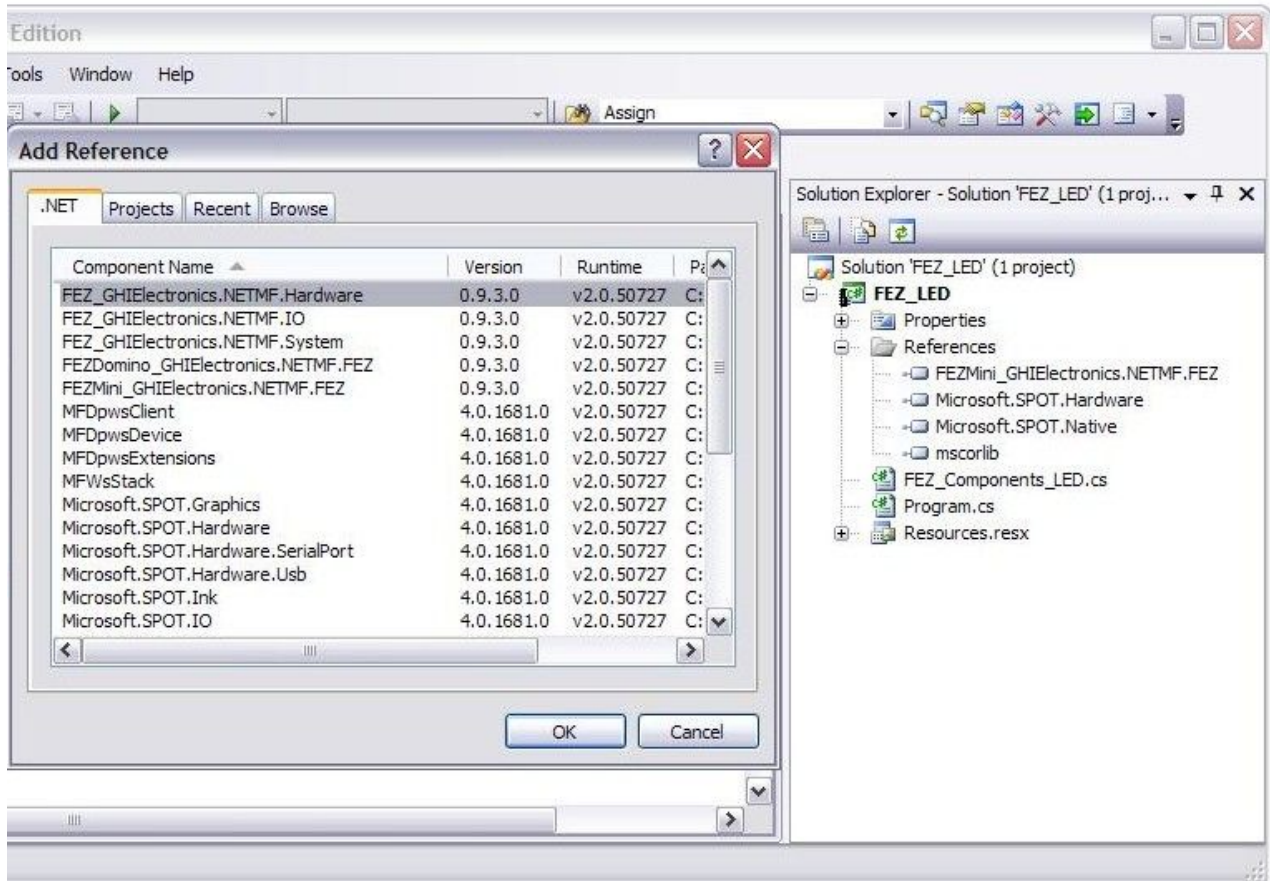
## 3.2. How to add an assembly to Visual C# Project?

Adding a library is pretty simple.

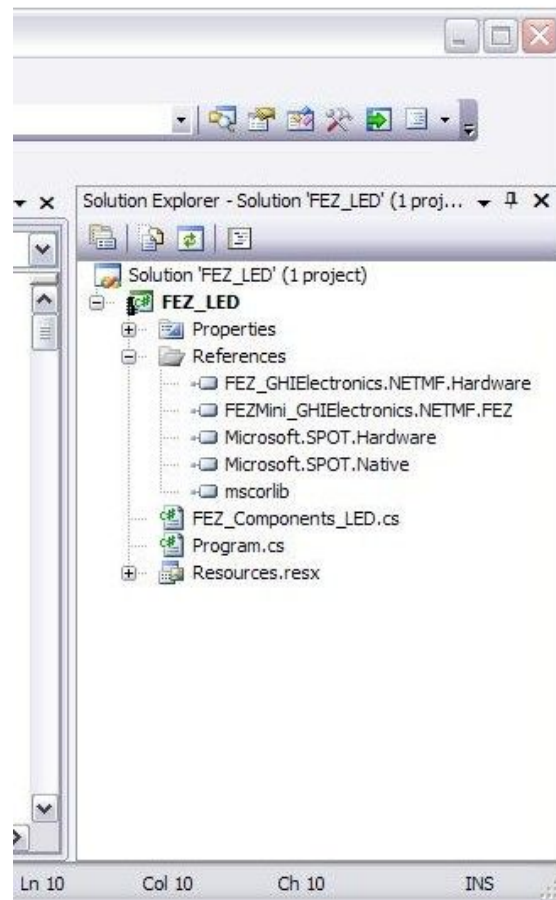
1. Go to the Menu and select “project --> Add Reference...”



2. Choose the library that you need then click OK.



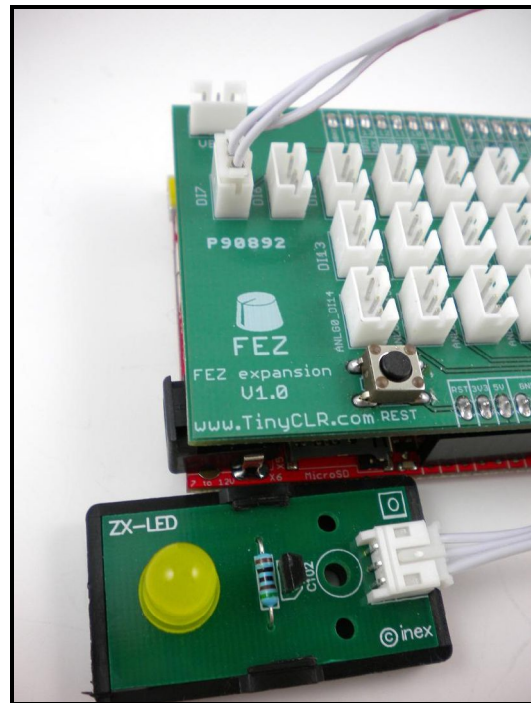
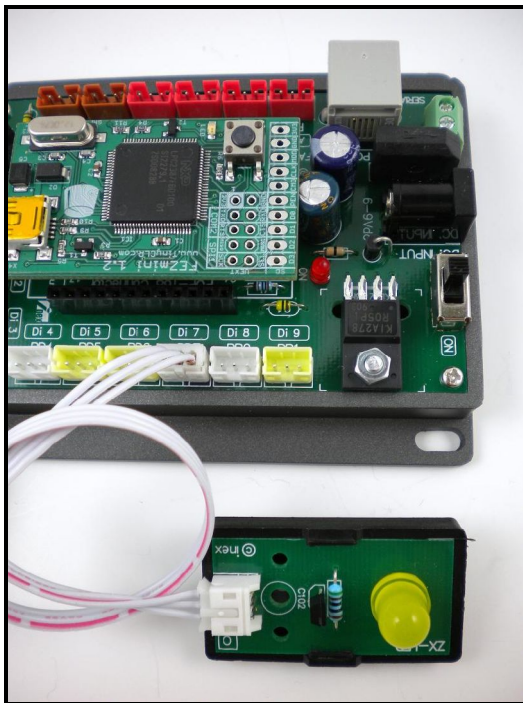
3. We can now see the new Assembly reference with the References in the solution explorer.



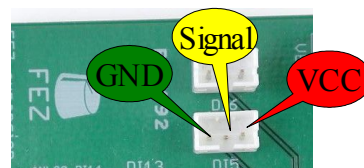
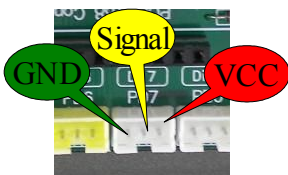
**Important note:** User must be aware that adding assemblies consumes RAM and Flash space, thus, the user is not expected to add unused assemblies or the project might fail to deploy due to the lack in resources. For beginners, it is okay to add them all if not sure what assemblies are needed.

## 4. FEZ Hardware Components

[www.TinyCLR.com](http://www.TinyCLR.com) website offers many components that can plug directly onto FEZ. For example, to have some light indicators, we will need the LED component. This component can plug into one of the connectors available on FEZ Mini's Starter Kit or Robot Kit and on FEZ Domino's Expansion shield. The images below shows how it is connected.



These JST female headers include 3 pins each. The middle pin is the signal which is connected directly to the relative pin on FEZ board, as marked on the header. For example, Di0, An2, ..etc.



That covered adding the hardware...very simply! Adding software is just as simple. But before, we need to make sure that we have added the appropriate libraries/assemblies to our project. Click on “Add assemblies” and add either

**FEZDomino\_GHIElectronics.NETMF.FEZ**

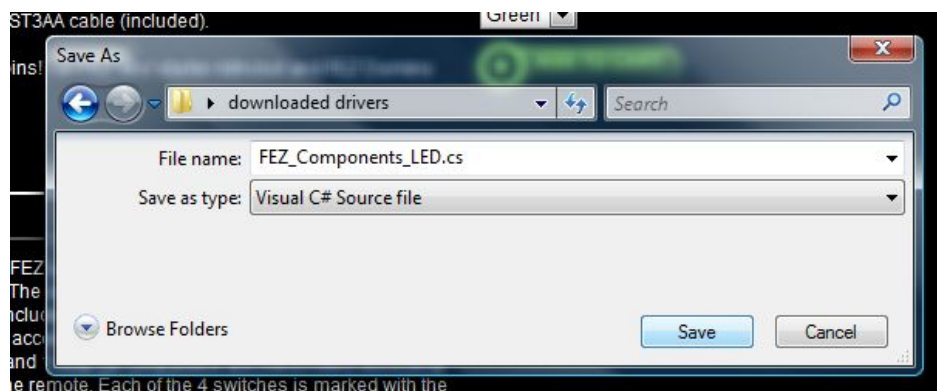
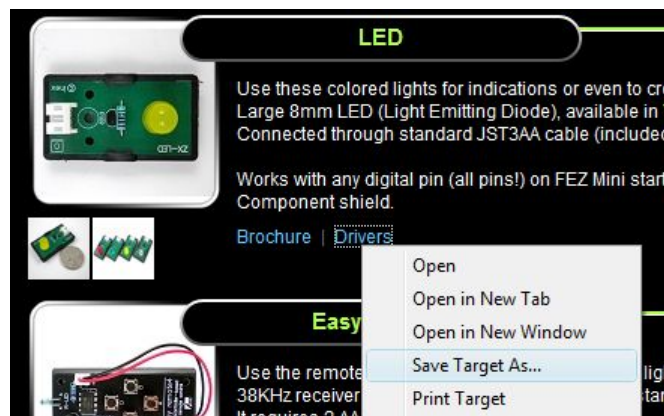
or

**FEZMini\_GHIElectronics.NETMF.FEZ**

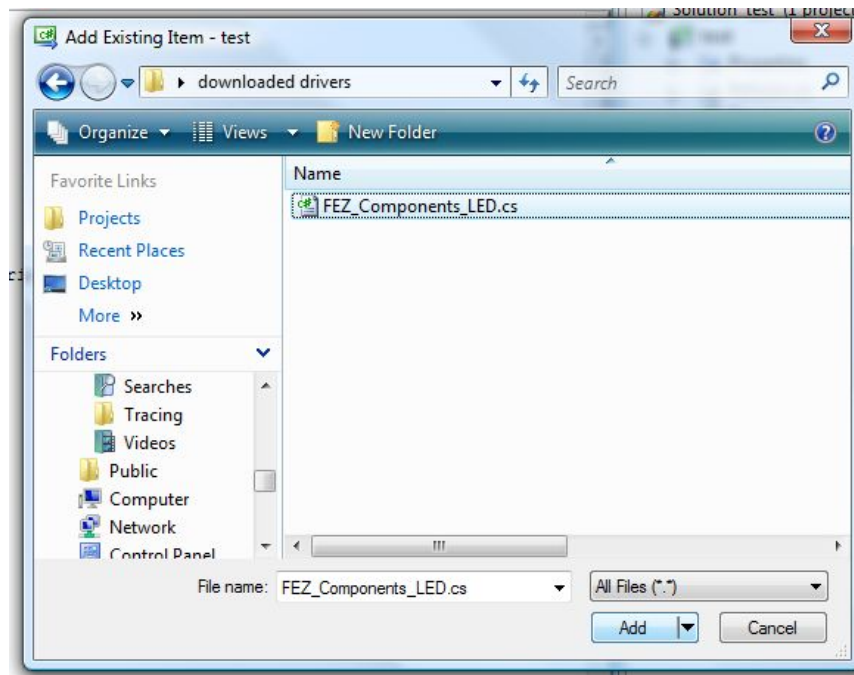
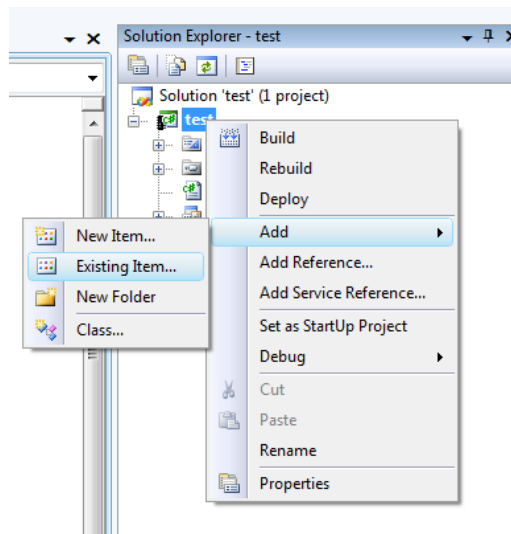
It is important that you add the appropriate one according to your FEZ. You may need to add other libraries according to the hardware used as mentioned in previous section such as **FEZ\_GHIElectronics.NETMF.Hardware**

Almost every component comes with a source-code C# driver. The same driver file will work with FEZ Domino and FEZ Mini.

To use the LED component, we will download the C# driver file



and include it in our project.

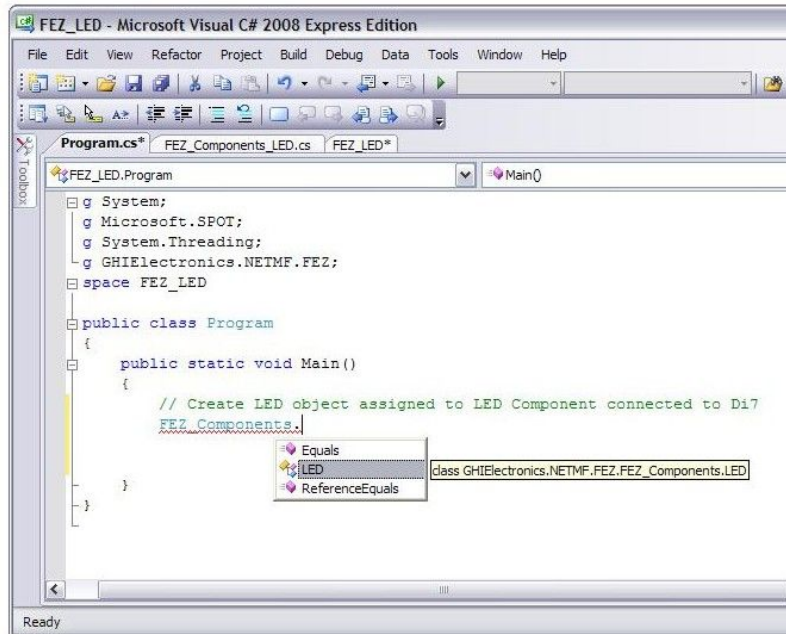


In our case the name of the file is **FEZ\_Components\_LED.cs** available under LED component link. Then in our project you will need to add

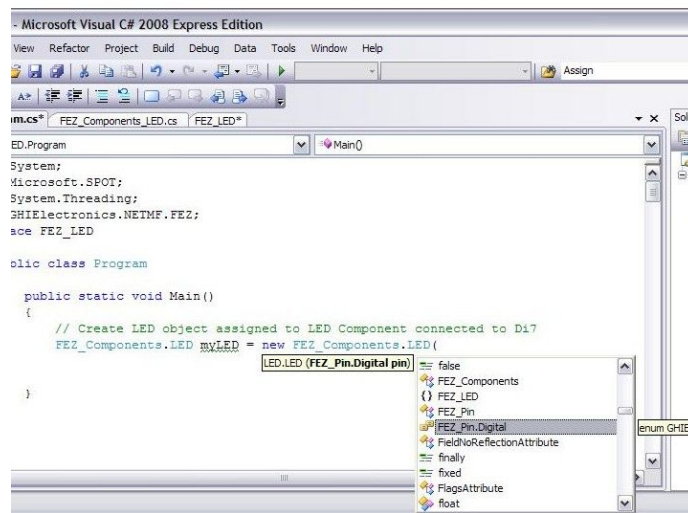
```
using GHIElectronics.NETMF.FEZ;
```

This will pull in everything we need.

Now in your code, if we type `FEZ.Components`, Visual Studio will automatically show us all the available components. Select LED, from there, you can initialize the object with what pin it is connected to as shown below:

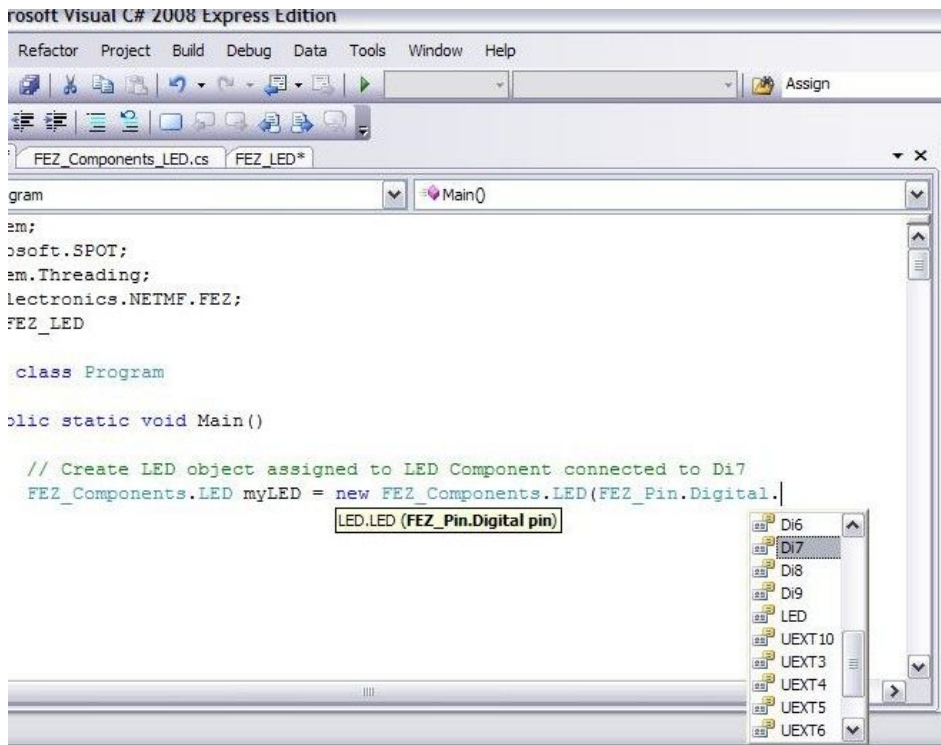


Create a new LED object and the constructor will automatically give you a list of available pins that can be used.

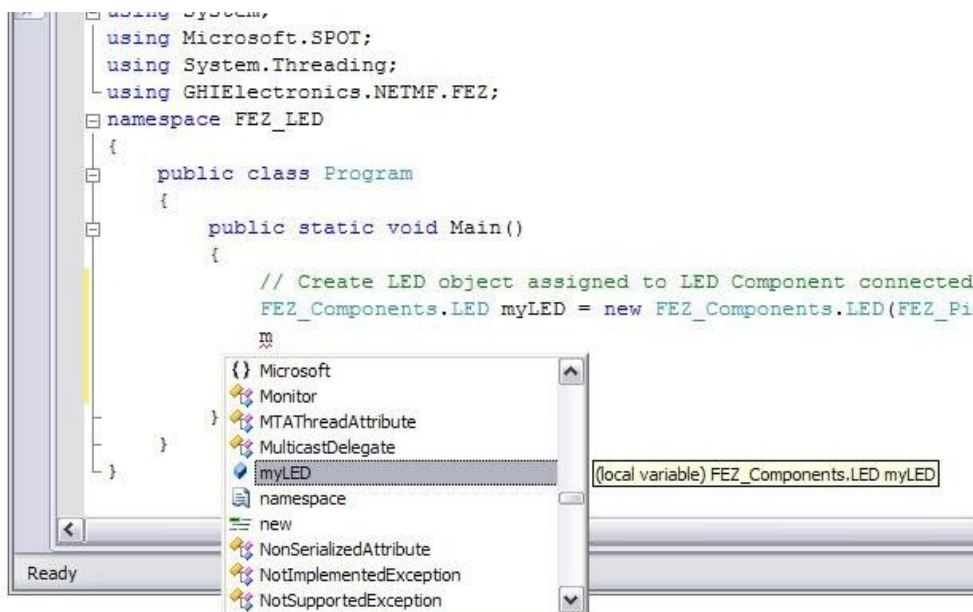


Let us assume that we have the LED connected to Di7






Now, we can use our LED object (myLED)

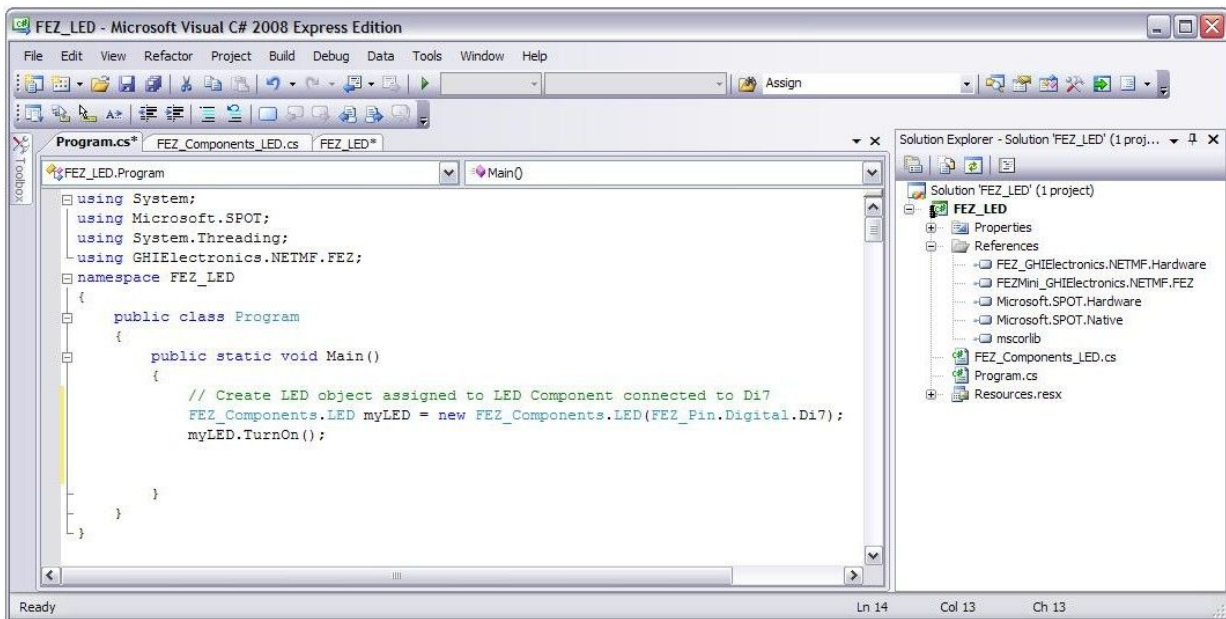


Very simply, Visual Studio will give us a list of available methods, like TurnOn.

```
public class Program
{
    public static void Main()
    {
        // Create LED object assigned to LED Component connected to Di7
        FEZ_Components.LED myLED = new FEZ_Components.LED(FEZ_Pin.Digital.Di7);
        myLED.
    }
}
```



Here is everything needed to create and LED object and then turn the LED on.



You can also find a brochure for every component that includes a brief description about the component with a simple starting-up code, using the mentioned C# driver. The following is a sample code quoted from LED component brochure:

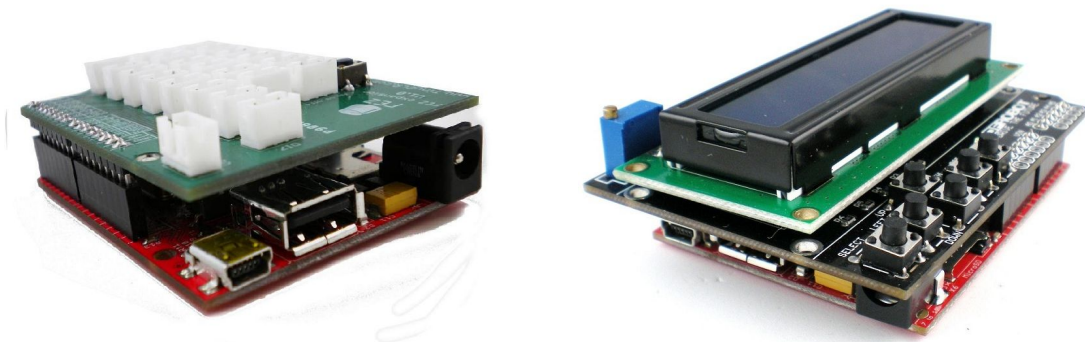
**Code snippet:**

```
using System;
using Microsoft.SPOT;
using System.Threading;
using GHIElectronics.NETMF.FEZ;
public class Program
{
    public static void Main()
    {
        // Create LED object assigned to the on board LED
        FEZ_Components.LED onBoardLED = new FEZ_Components.LED(FEZ_Pin.Digital.LED);
        // Turn the LED on
        onBoardLED.TurnOn();
        Thread.Sleep(1000);
        // Trun the LED off
        onBoardLED.ShutOff();
        Thread.Sleep(1000);
        //Blink the LED. on for 50 mSec and off for 200ms
        onBoardLED.StartBlinking(50, 200);
        Thread.Sleep(5000);
        onBoardLED.StopBlinking();
        //Dispose the LED object
        onBoardLED.Dispose();
    }
}
```

**Note:** Users can always ignore these drivers and develop their own but that requires a bit more knowledge about the hardware and software. The free eBook .NET Micro Framework Beginners Guide would give you this knowledge.

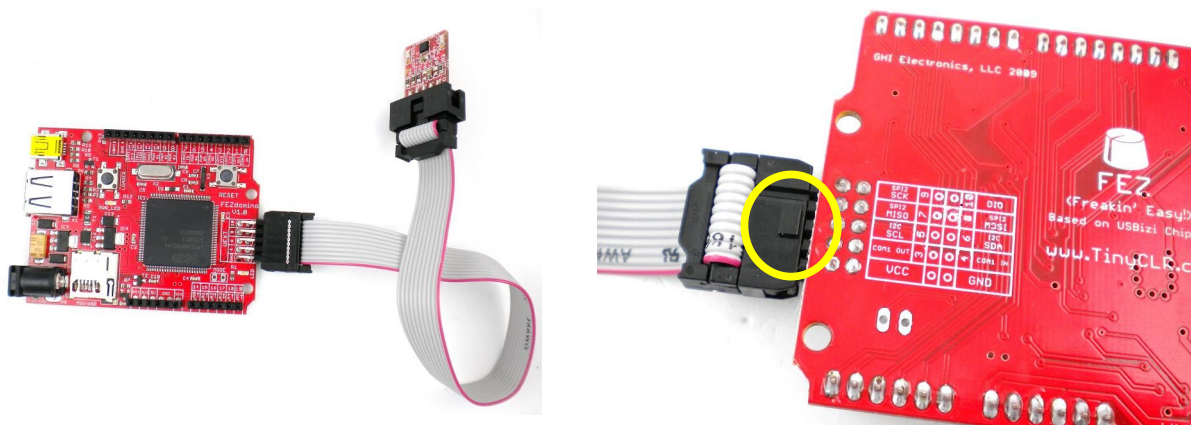
## Shields and Extensions

Shields are boards that plug in directly to FEZ Domino. An important shield is the component shield which let users connect the man components to Domino as shown in image below



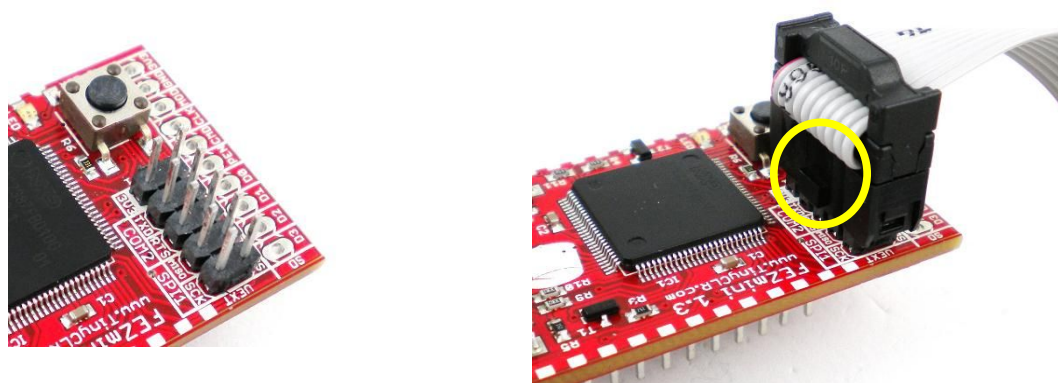
Extensions connect to UEXT connector. FEZ Domino come already with UEXT header so Using extensions is as simple as plug and play.

**Note:** The little squared on the cables connector should face downward with FEZ Domino, as you see in the next photo.



On the other hand, FEZ Mini has a placement for UEXT header but eh header is not placed on board by default. Users wanting to connect FEZ Mini to one of the extensions needs to solder a 0.1" header which require a little experience in soldering.

**Note:** The little squared on the cables connector should face toward inside with FEZ Mini, as you see in the next photo.



## 5. Firmware Update

Firmware is the program that manages an embedded device. Usually, an embedded device will have two programs stored internally, boot loader and firmware. When the device is powered up the boot loader (loader for short) will first run and initialize the system, then the loader checks for a valid firmware. If the firmware is available and is valid, the loader hands the execution to the firmware.

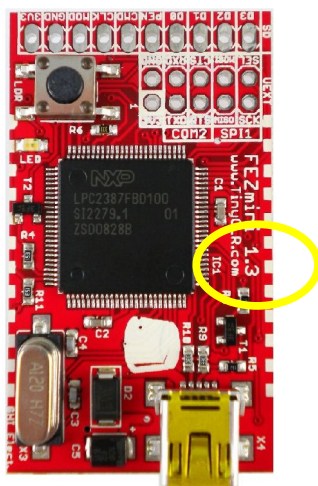
FEZ Mini and FEZ Domino boards are based on USBizi Chipsets.

For example, GHI Electronics may release a new USBizi (FEZ) firmware with additional features or bug fixes. You should be able to update the firmware on your device using a PC.

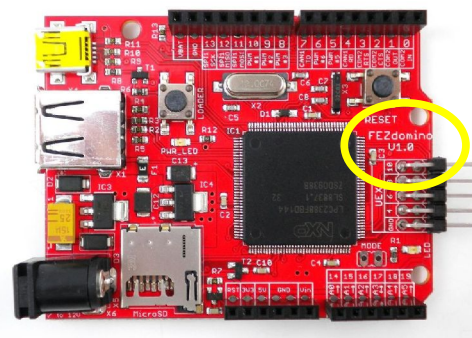
**Important note:** Always make sure the firmware loaded on the device matches the firmware version associated with the GHI NETMF SDK USBizi firmware. Ideally, whenever you install a new SDK on your PC, you will also update the firmware on your device and update the assemblies you have added to your application.

the next sections explains how to check the firmware version number and how to updating the firmware is explained here.

**Important Note:** The hardware version number printed on the hardware board is NOT related to firmware(software) version and should be ignored.



Ignore this version number

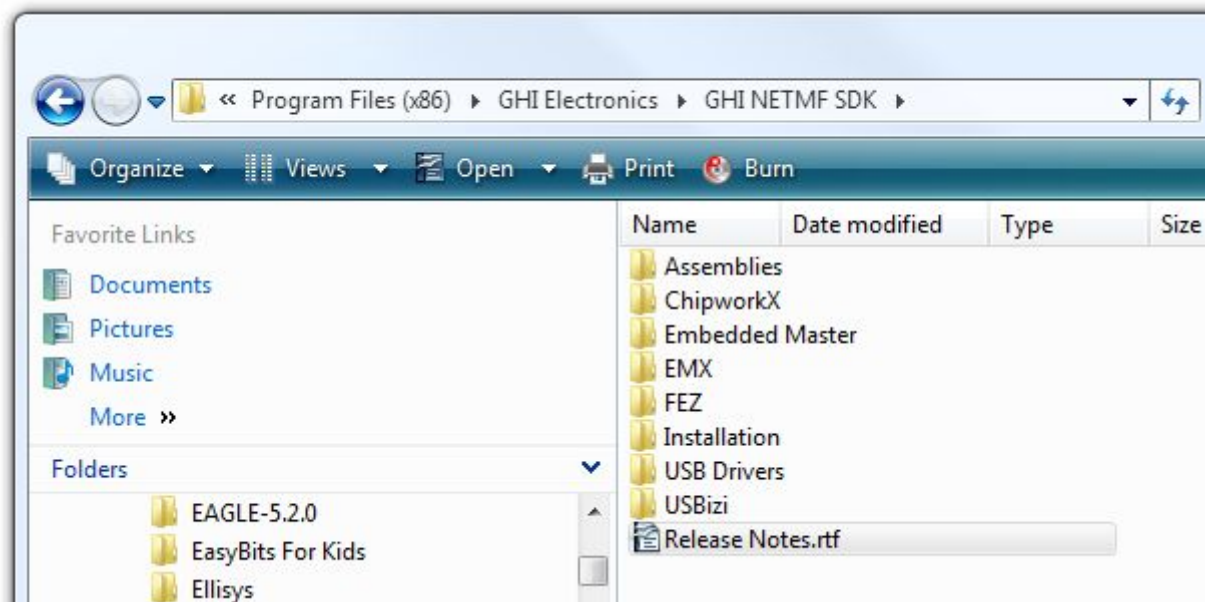


Ignore this version number

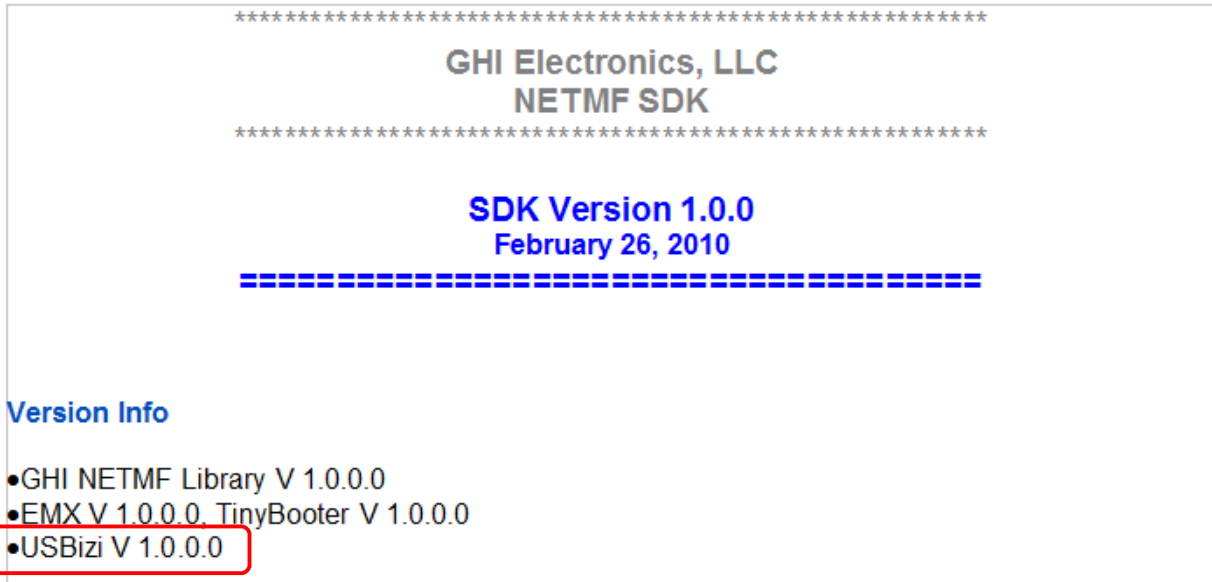
## 5.1. Checking the Firmware Version

Before updating the firmware, you can check the version number and make sure it needs to be updated or if you have the latest installed. Make sure you have the latest SDK and firmware installed.

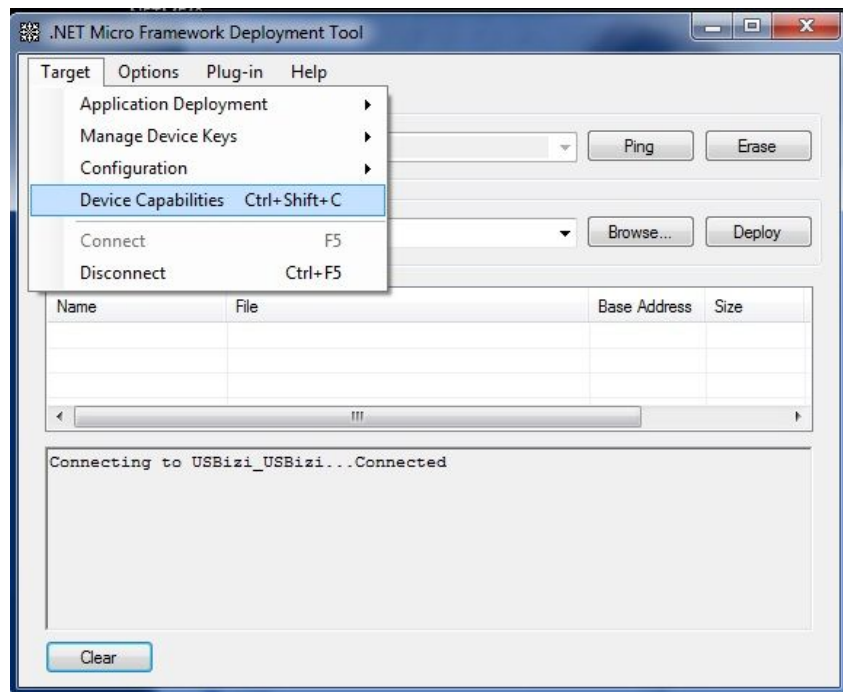
When go to [www.TinyCLR.com](http://www.TinyCLR.com), the current GHI NETMF SDK version is shown in the download link. You can download and install the SDK. When done, the version release notes and changes are shown automatically. These are also available in the SDK installation folder.



After you have the SDK installed, you can see the Release Notes file, for example USBizi firmware version number is 1.0.0.0 in this SDK, FEZ Mini and FEZ Domino are based on USBizi chipsets:



Next, let's verify USBizi firmware version number on your FEZ board. To do this, connect using MFDeploy over USB to FEZ as explained in previous section. Then, go to Target->Device Capabilities:



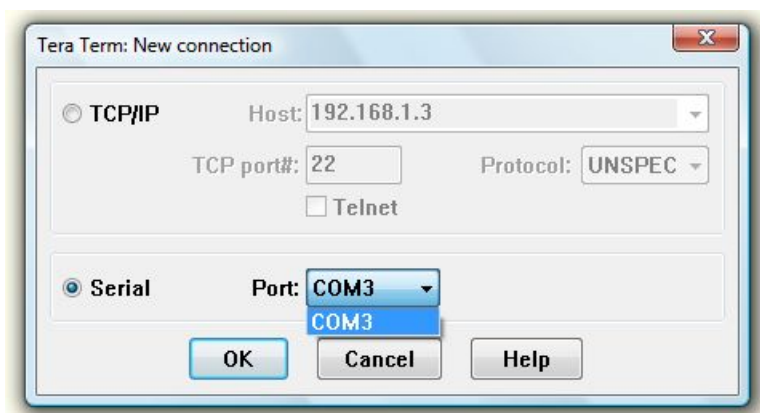
In the output messages, you will see the firmware release version under “SolutionReleaseInfo.solutionVersion” and in this case, it is 1.0.0.0, so we are OK and we don't need to update the firmware.

```
HalSystemInfo.moduleSerialNumber:
HalSystemInfo.systemSerialNumber:
ClrInfo.clrVersion:                4.0.1681.0
ClrInfo.clrVendorInfo:            Microsoft Copyright (C) Microsoft Corporation. All
rig
ClrInfo.targetFrameworkVersion:  4.0.1681.0
SolutionReleaseInfo.solutionVersion: 1.0.0.0
SolutionReleaseInfo.solutionVendorInfo: GHI Electronics, LLC
SoftwareVersion.BuildDate:        Feb 26 2010
SoftwareVersion.CompilerVersion:  310739
FloatingPoint:                    True
SourceLevelDebugging:             True
ThreadCreateEx:                  True
```

## 5.2. System Setup and Accessing the Boot Loader

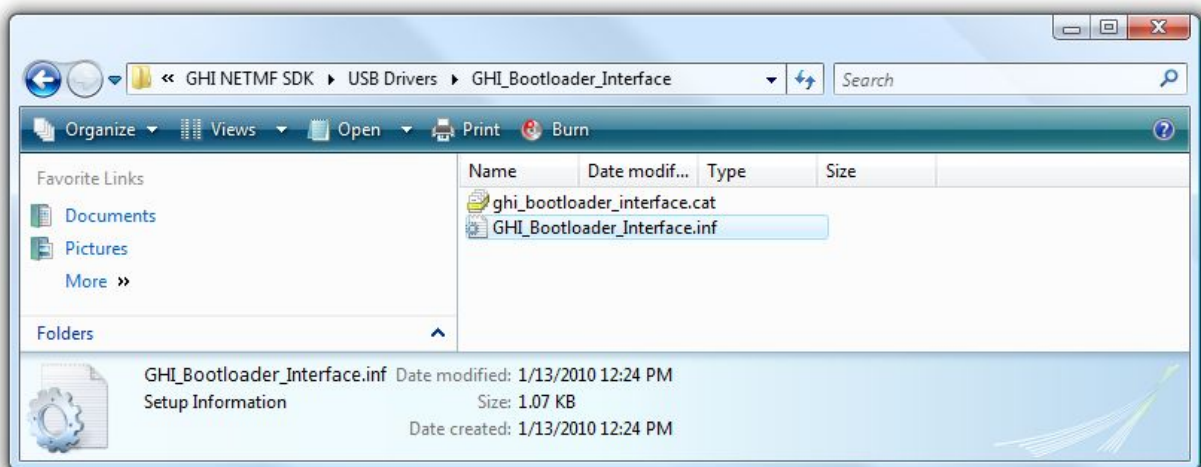
USBizi (FEZ) boot loader can be accessed through any terminal service software such as HyperTerminal or TeraTerm. Problems have been seen with some other terminals out there, so we highly recommend you download and install from GHI website: [Download TeraTerm](#). Other versions of TeraTerm have problems with XMODEM, so make sure you only use the one tested by GHI.

1. Install TeraTerm.
2. Disconnect FEZ (USBizi) from power and from your PC then open TeraTerm.
3. Select serial and click the arrow to drop down the list of COM ports. Note how many COM ports you have and write them down. On my system, I only have COM3.





4. Close TeraTerm for now.
5. Press an hold LOADER (LDR) button then connect the USB cable. If this is the first time, windows will ask for drivers. GHI boot loader is available in the GHI NETMF SDK \ USB Drivers folder. The boot loader driver is different from GHI NETMF Driver.
6. Release LOADER (LDR) button and lead Windows to GHI\_BootLoader\_Interface.inf to install the driver.



7. After windows is done installing the new drivers, open TeraTerm and observe what COM ports are now available. You should have a new COM port. This new COM port (serial port) is actually a USB connection between windows and USBizi loader. Windows applications do not know that this is a USB connection and treat it just like if it was a serial port.
8. Select the new COM port and clock "OK". Now press the "b" button on the keyboard. You should see back BL for every time you press "b". Now try to press "V" (upper case) to get back the loader version number. **Note**, your version number might be different! Also, this version is **NOT** related to GHI NETMF SDK or firmware.

```
BL
BL
BL
BL
1.01
BL
```

9. You are now successfully accessing the bootloader.

## 5.3. Boot loader Commands

The Boot Loader supports the following commands:

Command	Description
V	Get GHI Loader version number
E	Erase all Flash memory (except the boot loader)
X	Load new USBizi Firmware
N	Display serial number
R	Run Firmware
L	Load managed application
G	Read managed application
D	Delete managed application
P	Disable reading managed application

## 5.4. Simple Steps to update the firmware

First we want to make sure to start fresh before loading the new firmware.

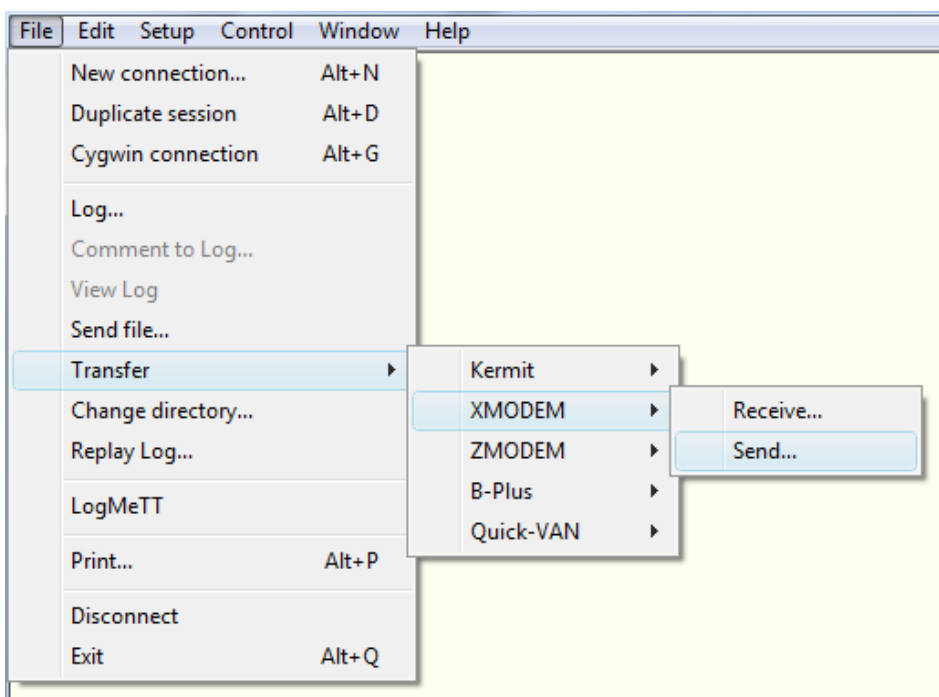
1. Access The boot loader using TeraTerm as explain earlier.
2. Erase Flash memory using E command then press Y to confirm, will take several seconds.
3. Loading new Firmware is simple but it requires a terminal that supports XMODEM file transfer. XMODEM has many versions, GHI Loader requires 1K transfers with 16-bit CRC error checking. Keep on using TeraTerm software.

Transfer is initiated using the X command. Once the X command is entered, GHI Loader will start sending back character C continuously. This C is an indicator for XMODEM that a device is waiting for data. After you see character C coming on the terminal window, you can now select XMODEM transfer and point the software to the firmware file from the GHI NETMF SDK "FEZ Domino & Mini.GHI".

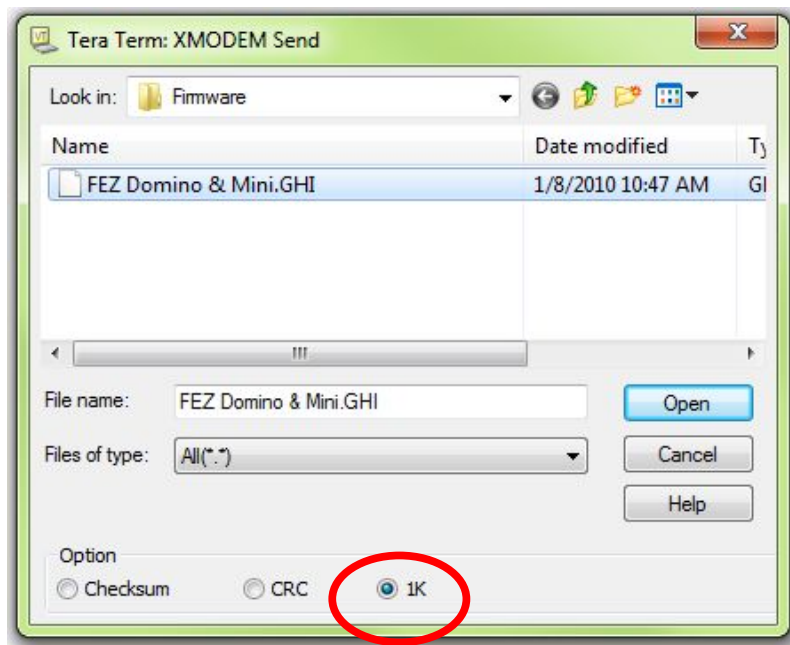
Entering X command:

```
BL
BL
BL
BL
1.01
BL
Start File Transfer
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
```

Select Xmodem transfer:



Select the firmware file from GHI NETMF SDK:



Updating the firmware takes very few seconds to load and when loading is done and the file is valid, the new firmware is executed automatically and you will not see "BL" again.



## 6. What Is Next?

This tutorial only makes sure FEZ is connected and you are able to use it. From this point you can use the free eBook for NETMF to learn all the great things you can do with FEZ.

This is a list of great resources that should help you in your next invention!

- The official FEZ website with all the fun  
<http://www.tinyclr.com/>
- GHI blog is always a good place to visit  
<http://tinyclr.blogspot.com/>
- The Micro Framework Project website is an excellent resource  
<http://www.microframeworkprojects.com/>
- A good and free eBook to continue learning about C# is available at  
<http://www.programmersheaven.com/2/CSharpBook>
- Jens Kuhner excellent book on .NET Micro Framework  
<http://www.apress.com/book/view/9781430223870>
- USB complete is an excellent book on USB  
<http://www.lvr.com/usbc.htm>
- Wikipedia is a good place for information about everything!  
[http://en.wikipedia.org/wiki/.NET\\_Micro\\_Framework](http://en.wikipedia.org/wiki/.NET_Micro_Framework)
- .NET Micro Framework's main page on Microsoft's website  
<http://www.microsoft.com/netmf>
- .NET Micro Framework Community  
<http://www.netmf.com/>