.NET Gadgeteer Intro

Distributed Embedded Systems Lab

Daniel Stojcsics, PhD
stojcsics.daniel@nik.uni-obuda.hu

2015 autumn

Overview

1. What is .NET Gadgeteer?
2. FEZ Spider Starter Kit
3. Usage of the Starter Kit
4. First Gadgeteer application
5. Pictures from the app

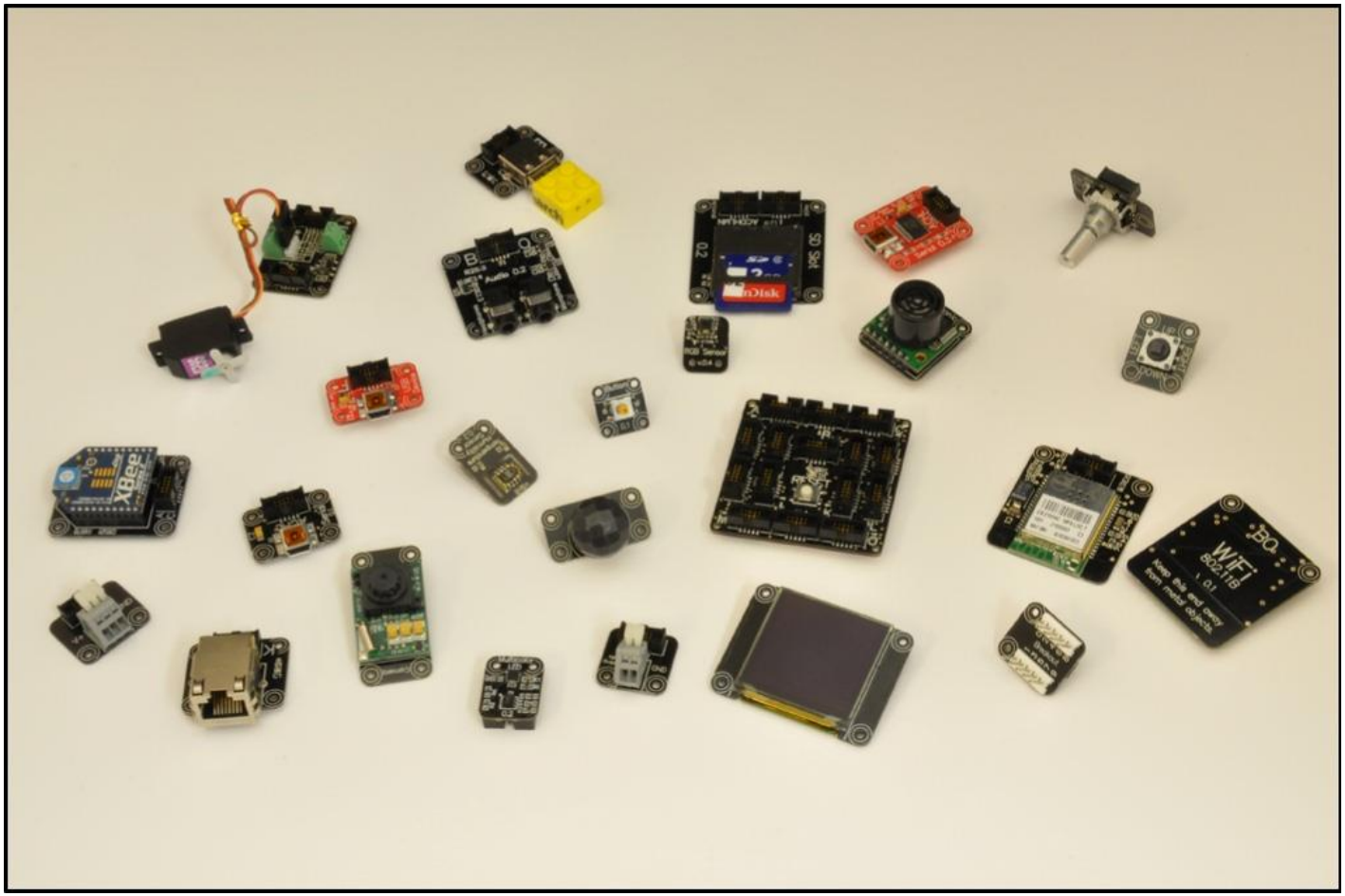
1. What is .NET Gadgeteer?

- Microsoft .NET Gadgeteer is a rapid prototyping platform for small electronic gadgets and embedded hardware devices.

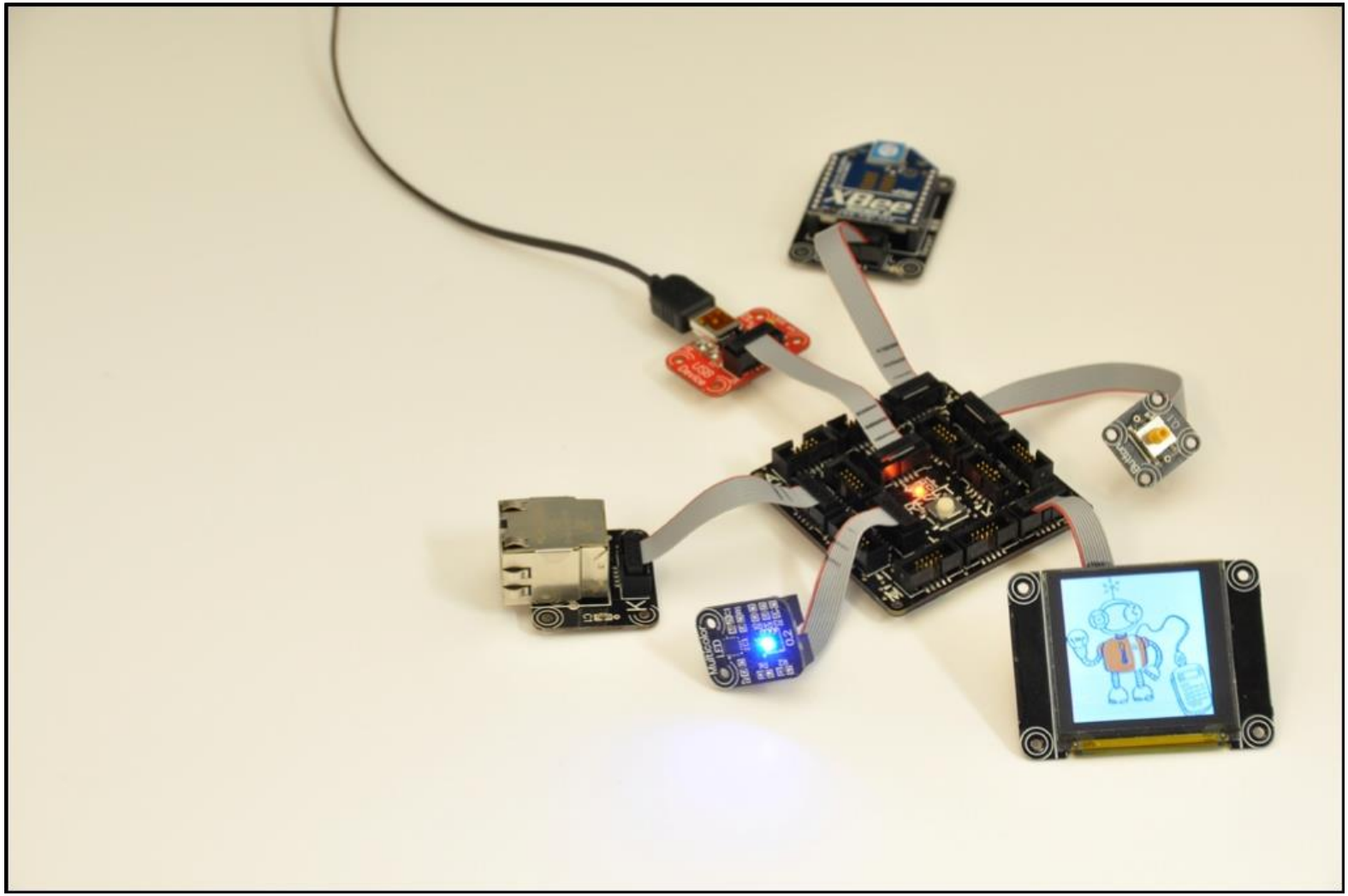
- It combines the advantages of object-oriented programming, solderless assembly of electronics using a kit of hardware modules, and quick physical enclosure fabrication using computer-aided design.

- The platform is built on the .NET Micro Framework, which allows small devices to be programmed in the C# language and make use of Visual Studio's programming and debugging tools.

1. What is .NET Gadgeteer?



1. What is .NET Gadgeteer?



.NET Gadgeteer dependencies

Hardware

- Main board (eg.: Fez Spider, Fez Hydra, Sytech NANO)
- USBClient module („RED module”) -> Power suppy & debug interface
- Misc. modules (eg.: sensors, displays, LEDs, input devices, connectivity etc.)

The **FEZ Spider Starter Kit** includes all of these ☺

Software

- Microsoft Visual Studio 2010+ (or the free Visual C# 2010 Express)
- **.NET Micro Framework SDK**
- Manufacturer dependent drivers

.NET Micro Framework

- A memory footprint of about 300 KB; for comparison, the next smallest .NET implementation, the .NET Compact Framework running on Windows CE, needs about 12 MB

- Can run directly "on the metal" without an operating system; running on an OS is also possible

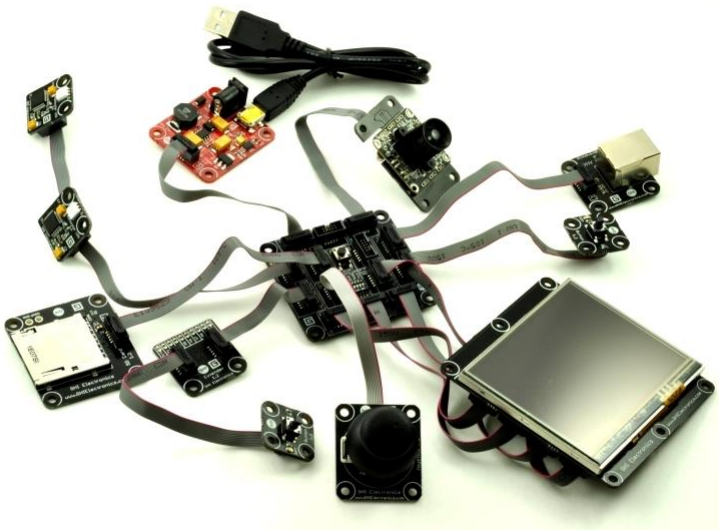
- Supports common embedded peripherals and interconnects, including flash memory, EEPROM, GPIO, I^2C, SPI, Serial port, USB

- Optimized for energy-efficiency in battery-powered devices

2. FEZ Spider Starter Kit (1)

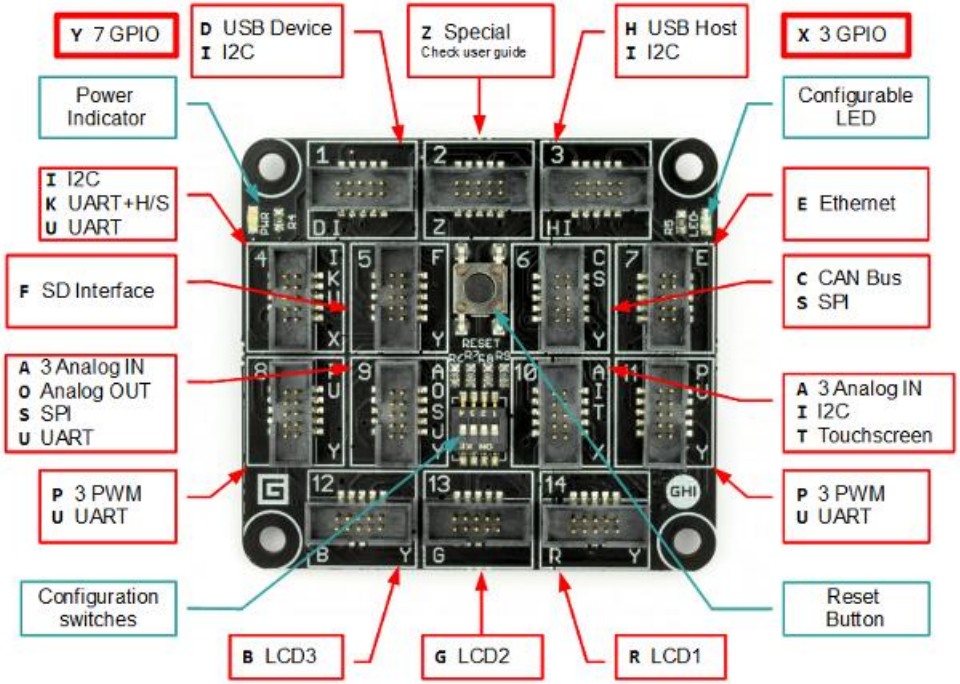
The kit includes:

- FEZ Spider Mainboard
- Display TE35 Module (3.5" with touchscreen)
- USB Client DP Module (with USB cable)
- Camera Module
- 2x Multicolor LED Module (DaisyLink)
- 2x Button Module
- Ethernet J11D Module
- SD Card Module
- USB Host Module
- Extender Module
- Joystick Module



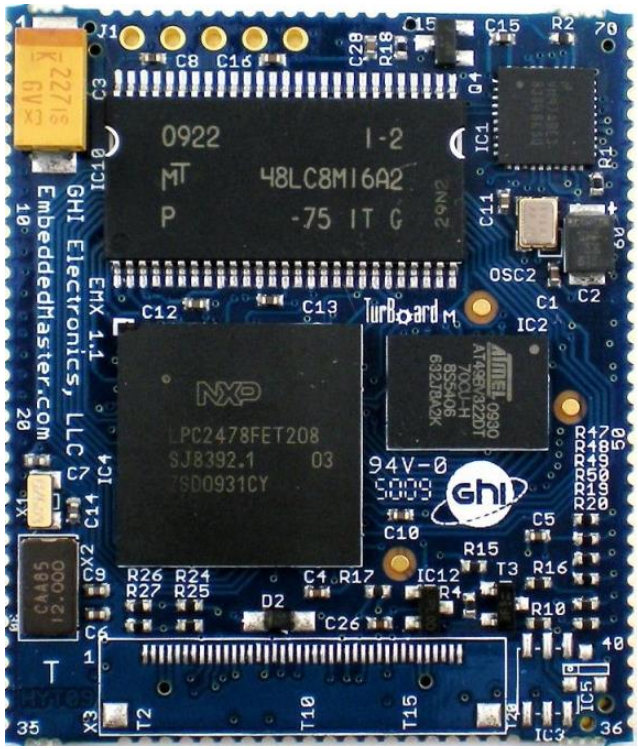
FEZ Spider main board

- 14 .NET Gadgeteer compatible sockets that include these types: X, Y, A, C, D, E, F, H, I, K, O, P, S, T, U, R, G, B and Z.
- Configurable on-board LED
- Configuration switches.
- **EMX module:**



EMX module (1)

- EMX module contains all the basic hardwares (ARM processor, flash, RAM, ethernet, etc) on an System on Module (SoM) platform.



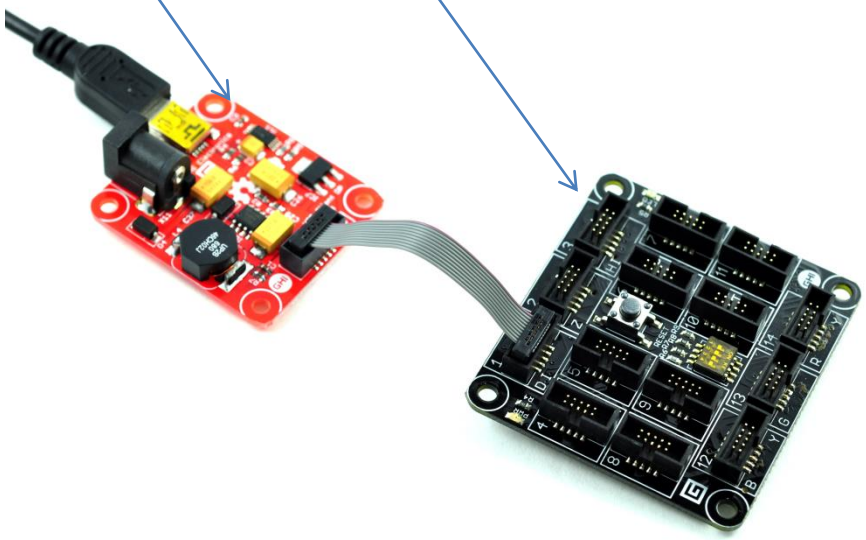
EMX module (2)

Specs:

- 72MHz 32-bit ARM7 processor
- 4.5 MB Flash
- 16 MB RAM
- LCD controller
- Full TCP/IP Stack with SSL, HTTP, TCP, UDP, DHCP
- Ethernet, WiFi driver and PPP (GPRS/ 3G modems) and DPWS
- USB host
- USB Device with specialized libraries to emulate devices like thumb-drive, virtual COM (CDC), mouse, keyboard
- 76 GPIO Pin
- 2 SPI (8/16bit)
- I2C
- 4 UART
- 2 CAN Channels
- 7 10-bit Analog Inputs
- 10-bit Analog Output (capable of WAV audio playback)
- 4-bit SD/MMC Memory card interface
- 6 PWM
- OneWire interface (available on any IO)
- Built-in Real Time Clock (RTC) with the suitable crystal
- Processor register access
- OutputCompare for generating waveforms with high accuracy
- RLP allowing users to load native code (C/Assembly) for real-time requirements
- Extended double-precision math class
- FAT File System
- Cryptography (AES and XTEA)
- Low power and hibernate support
- In-field update (from SD, network or other)

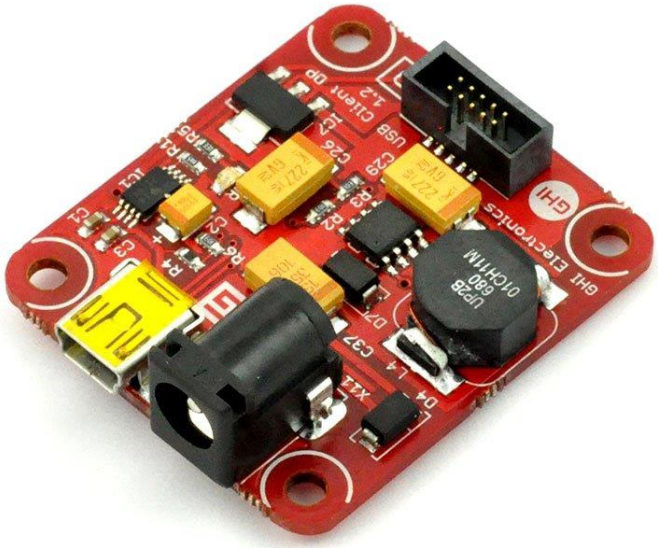
Usage of the mainboard

- FEZ Spider main board contains 14 slot for connecting the modules
- **„Reset**" button ☺
- USB Client Dual Power module gives the necessary power from USB or DC power supply



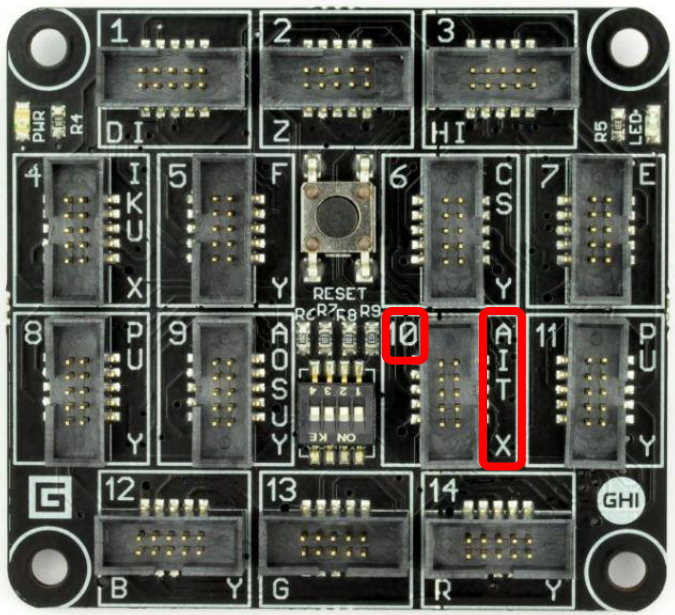
„Red module” – Power module

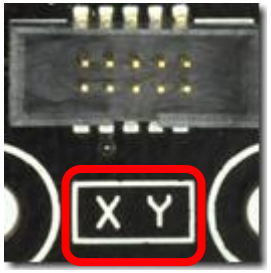
It has only one slot: „D”
It can only paired the other „D” slot on the main board!
(pairing only the same letter slots!)



Slot marks on the panels and the sensors (1)

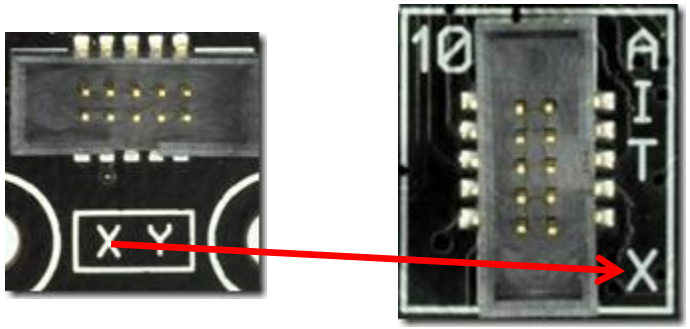
- There are slot marks on the main board and on the sensors
  - Slot no. (1-14)
  - Letters for corresponding sockets





Slot marks on the panels and the sensors (2)

- XY marks on the sensor
- On the main board it can paired with X or Y socket (e.g. slot no. 10)



- X and Y are GPIO!
- Connect only the same letter slots!!!!

Socket Types Table (Version 16)

Slot r

- XY
- On

- Cor

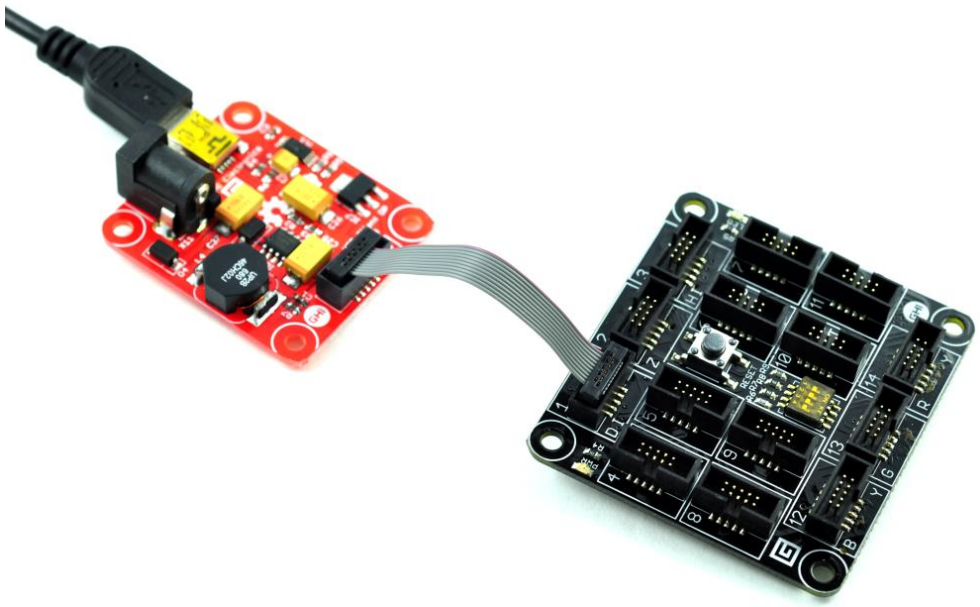
TYPE	LETTER	PIN 1	PIN 2	PIN 3	PIN 4	PIN 5	PIN 6	PIN 7	PIN 8	PIN 9	PIN 10
3 GPIO	X	+3.3V	+5V	GPIO!	GPIO	GPIO	[UN]	[UN]	[UN]	[UN]	GND
7 GPIO	Y	+3.3V	+5V	GPIO!	GPIO	GPIO	GPIO	GPIO	GPIO	GPIO	GND
Analog In	A	+3.3V	+5V	AIN (G!)	AIN (G)	AIN	GPIO	[UN]	[UN]	[UN]	GND
CAN	C	+3.3V	+5V	GPIO!	TD (G)	RD (G)	GPIO	[UN]	[UN]	[UN]	GND
USB Device	D	+3.3V	+5V	GPIO!	D-	D+	GPIO	GPIO	[UN]	[UN]	GND
Ethernet	E	+3.3V	+5V	[UN]	LED1 (OPT)	LED2 (OPT)	TX D-	TX D+	RX D-	RX D+	GND
SD Card	F	+3.3V	+5V	GPIO!	DAT0	DAT1	CMD	DAT2	DAT3	CLK	GND
USB Host	H	+3.3V	+5V	GPIO!	D-	D+	[UN]	[UN]	[UN]	[UN]	GND
I2C	I	+3.3V	+5V	GPIO!	[UN]	[UN]	GPIO	[UN]	SDA	SCL	GND
UART+ Handshaking	K	+3.3V	+5V	GPIO!	TX (G)	RX (G)	RTS	CTS	[UN]	[UN]	GND
Analog Out	O	+3.3V	+5V	GPIO!	GPIO	AOUT	[UN]	[UN]	[UN]	[UN]	GND
PWM	P	+3.3V	+5V	GPIO!	[UN]	[UN]	GPIO	PWM (G)	PWM (G)	PWM	GND
SPI	S	+3.3V	+5V	GPIO!	GPIO	GPIO	CS	MOSI	MISO	SCK	GND
Touch	T	+3.3V	+5V	[UN]	YU	XL	YD	XR	[UN]	[UN]	GND
UART	U	+3.3V	+5V	GPIO!	TX (G)	RX (G)	GPIO	[UN]	[UN]	[UN]	GND
LCD 1	R	+3.3V	+5V	LCD R0	LCD R1	LCD R2	LCD R3	LCD R4	LCD VSYNC	LCD HSYNC	GND
LCD 2	G	+3.3V	+5V	LCD G0	LCD G1	LCD G2	LCD G3	LCD G4	LCD G5	BACK-LIGHT	GND
LCD 3	B	+3.3V	+5V	LCD B0	LCD B1	LCD B2	LCD B3	LCD B4	LCD EN	LCD CLK	GND
Manufacturer	Z	+3.3V	+5V	[MS]	[MS]	[MS]	[MS]	[MS]	[MS]	[MS]	GND

The application

Let’s make an app which is blinking led after a button pressed.

Preparation

- Connect the „red module” with the FEZ Spider main board slot no. 1!
  D slot is only used for USB
- The remaining modules can be connected after this step!



Warning!!!
The modules can be connected only in POWER OFF STATE!!!

Connecting the components

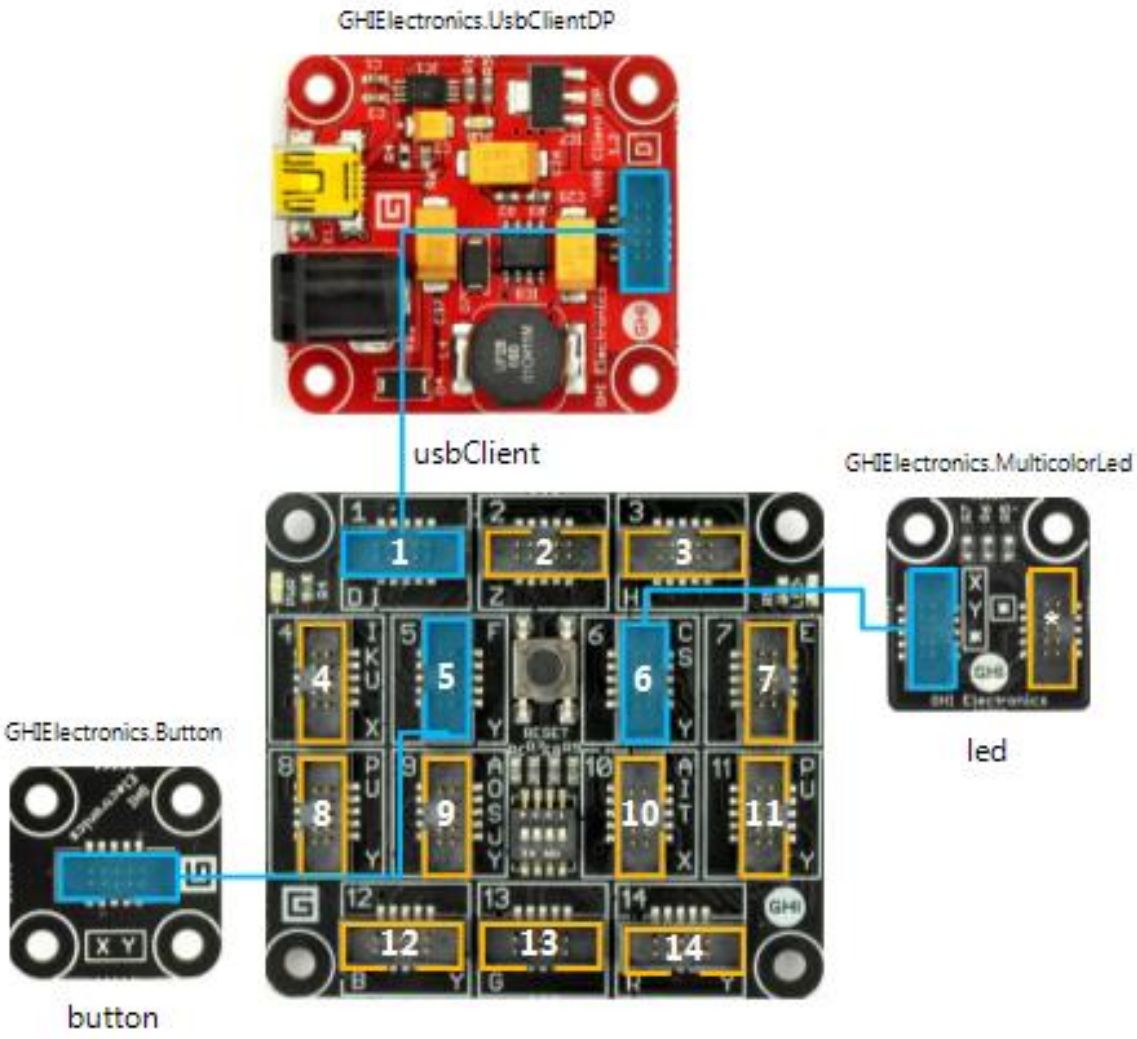
We need the following modules:

- Multicolor LED

- Button

Connecting the components

Connect the modules like this:

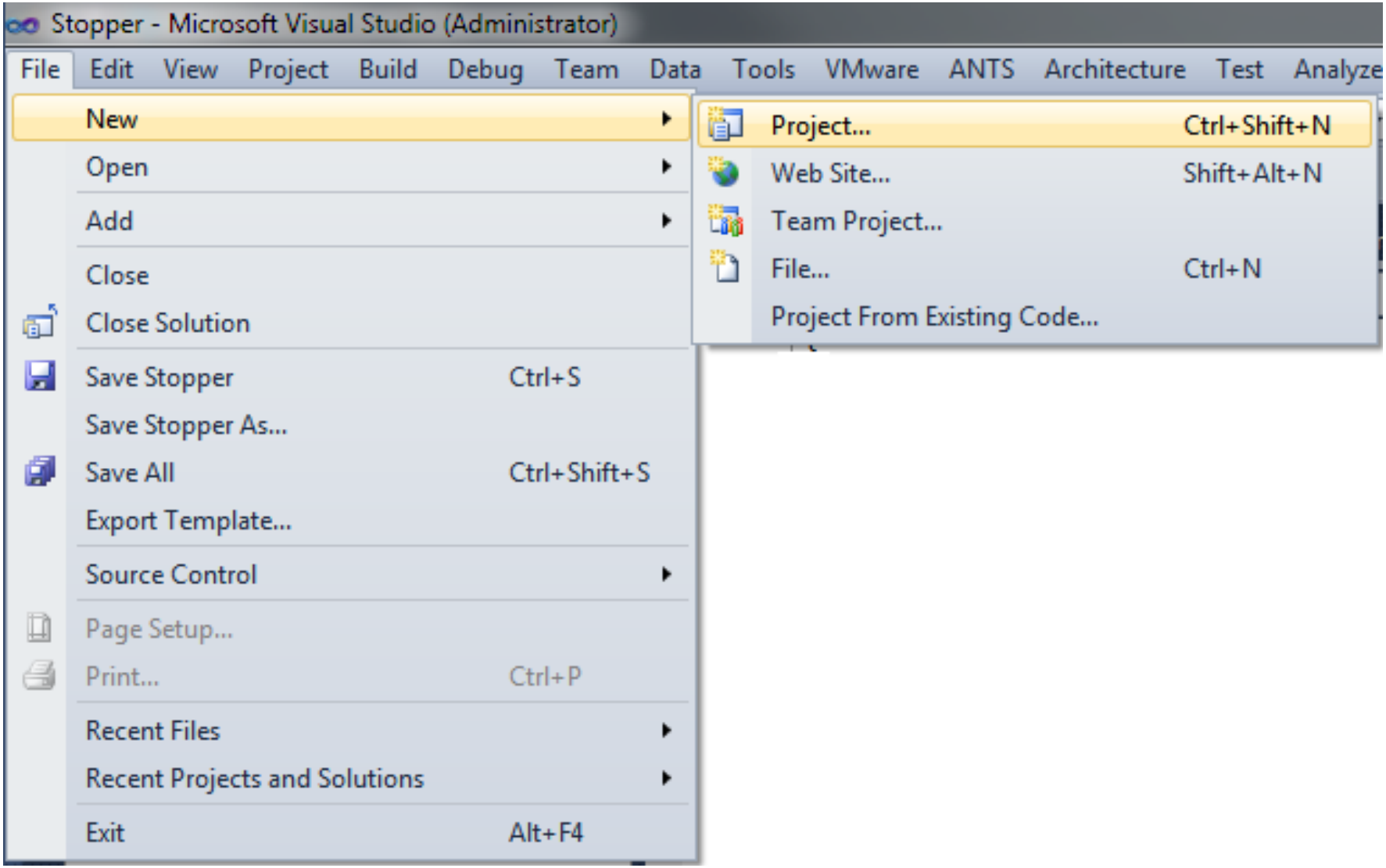


Connecting the Starter Kit with PC

- Connect the USB cable to the red module
- Connect the USB cable other end to PC
- On power the main board boots up, than starts the (previous) program on it
- Via Visual Studio the new program can be deployed

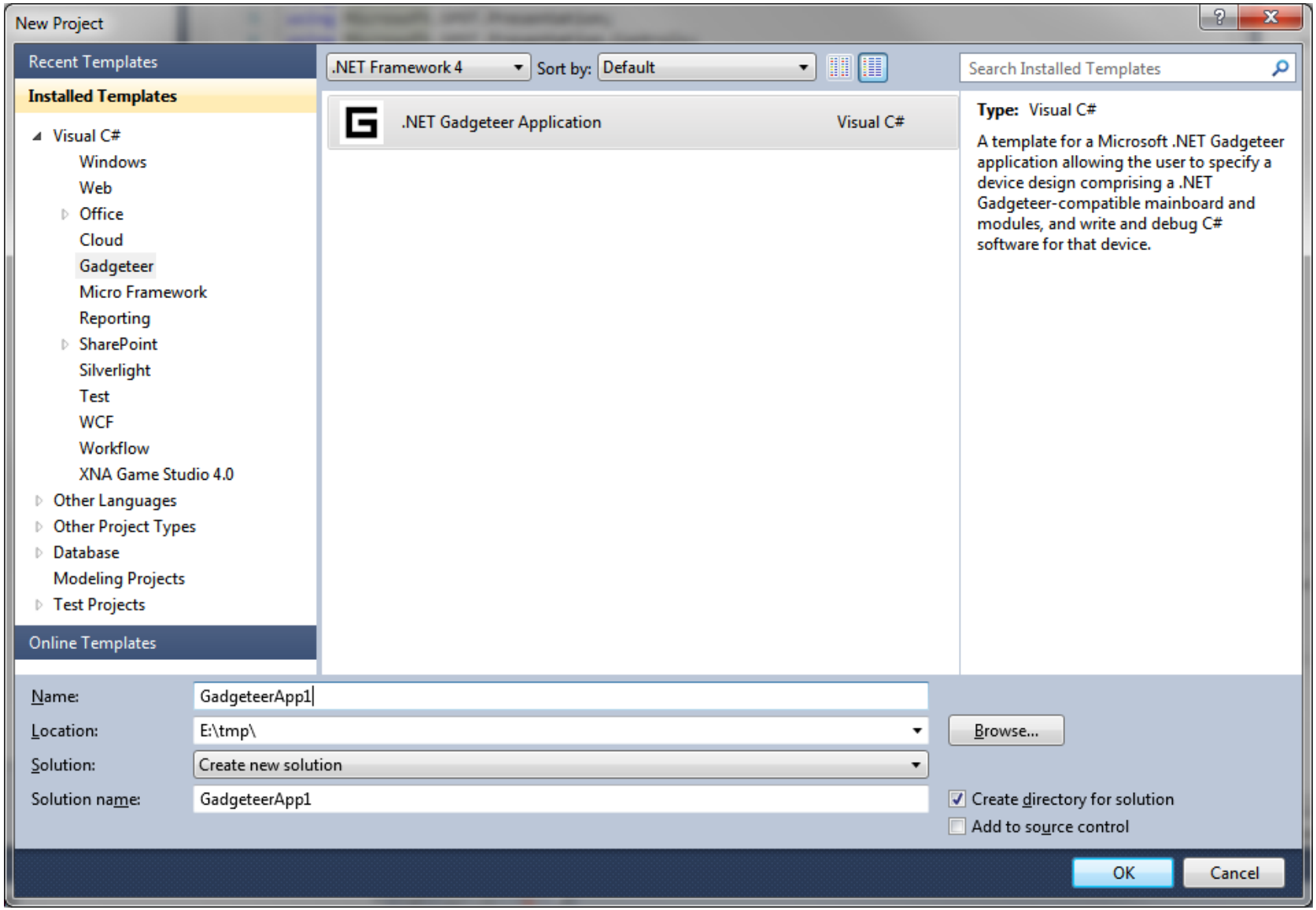
4. First application (6)

Open Visual Studio 2010
File → New → Project…

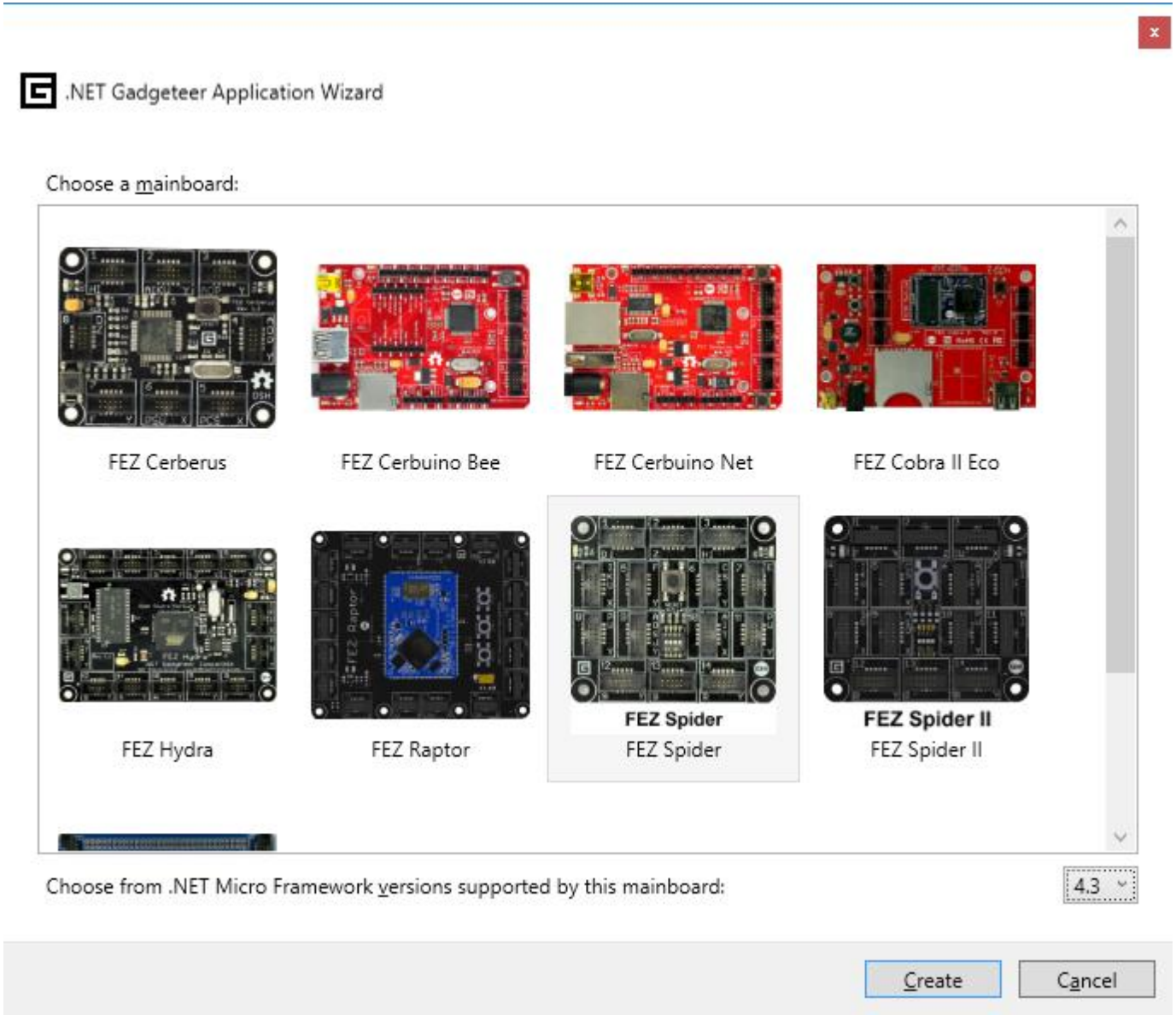


4. First application (7)

Pop up window: Visual C#/Gadgeteer/.NET Gadgeteer Application.
Name the project and declare a location (e.g. C:/student/)

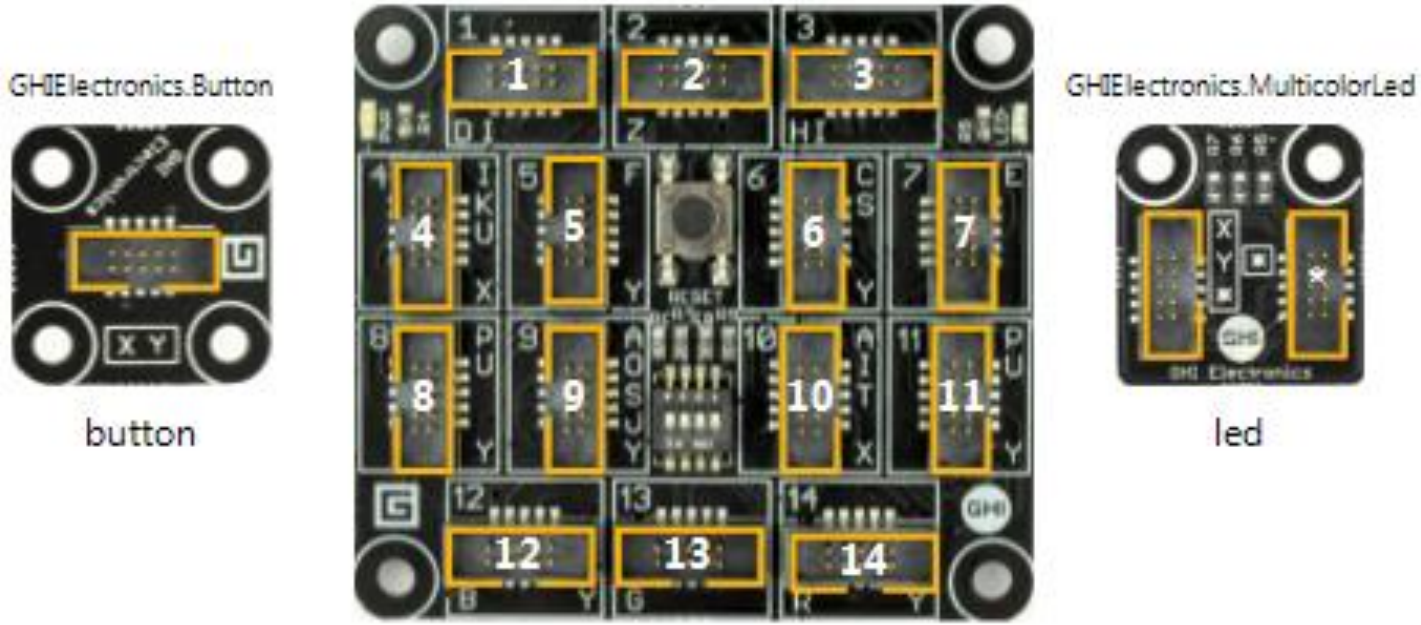


The Starter kit contains a FEZ Spider main board!



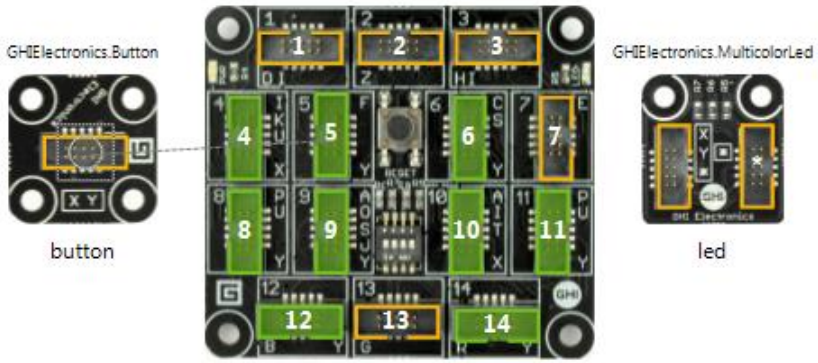
Connecting the modules

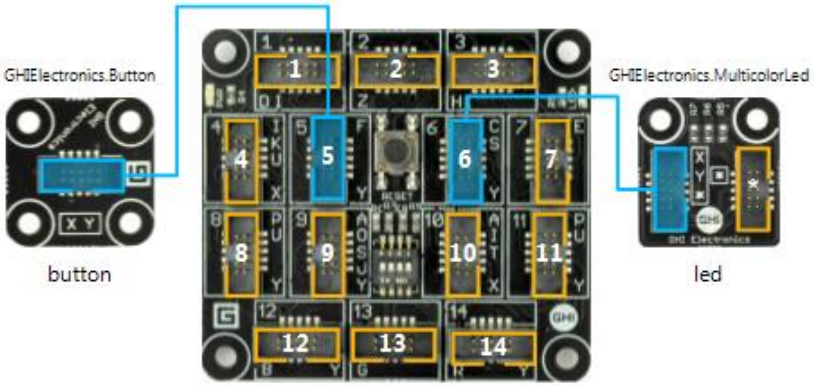
- Drag and drop the Button and the Multicolor led modules from the toolbox to the designer
- Connect the modules in the designer same as in the „real life” ☺



Connecting the modules

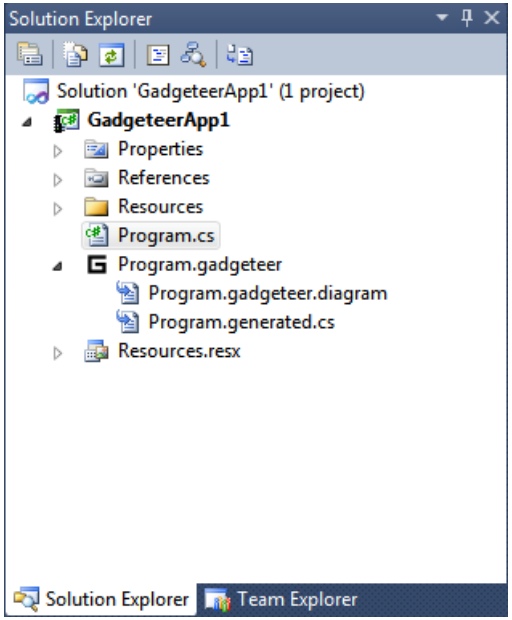
- Select the socket on the Button (yellow part
- Select the suggested green destination slot (we used slot no.5 in for the hardware)
- Completed connection is a blue line
- Now, the Visual Studio knows how our hardware looks like, it can help to make the coding part… ☺





Coding…

Duble click on Program.cs in the Solution Explorer!



Coding…

A Program.cs source file is created using a template
It contains a Program Class and a ProgramStarted() method. It runs first after deployment (Start Debugging)

Program.gadgeteer | Program.cs

Stopper.Program | ProgramStarted()

```
1  using System;
2  using System.Collections;
3  using System.Threading;
4  using Microsoft.SPOT;
5  using Microsoft.SPOT.Presentation;
6  using Microsoft.SPOT.Presentation.Controls;
7  using Microsoft.SPOT.Presentation.Media;
8  using Microsoft.SPOT.Touch;
9
10 using Gadgeteer.Networking;
11 using GT = Gadgeteer;
12 using GTM = Gadgeteer.Modules;
13
14 namespace Stopper
15 {
16     public partial class Program
17     {
18         // This method is run when the mainboard is powered up or reset.
19         void ProgramStarted()
20         {
21             /*******************************************************************************
22             Modules added in the Program.gadgeteer designer view are used by typing
23             their name followed by a period, e.g.  button.  or  camera.
24
25             Many modules generate useful events. Type +=<tab><tab> to add a handler to an event, e.g.:
26                 button.ButtonPressed +=<tab><tab>
27
28             If you want to do something periodically, use a GT.Timer and handle its Tick event, e.g.:
29                 GT.Timer timer = new GT.Timer(1000); // every second (1000ms)
30                 timer.Tick +=<tab><tab>
31                 timer.Start();
32             *******************************************************************************/
33
34
35             // Use Debug.Print to show messages in Visual Studio's "Output" window during debugging.
36             Debug.Print("Program Started");
37         }
38     }
39 }
```

100 %

Coding…

Complete ProgramStarted() method with the following lines:

```
void ProgramStarted()
{
    Debug.Print("Program Started");
    button.ButtonPressed += button_ButtonPressed

}
```

HINT: Press TAB button twice here ☺

- To use the button we have to subscribe to the *ButtonPressed* event using a *button_ButtonPressed* method as a parameter

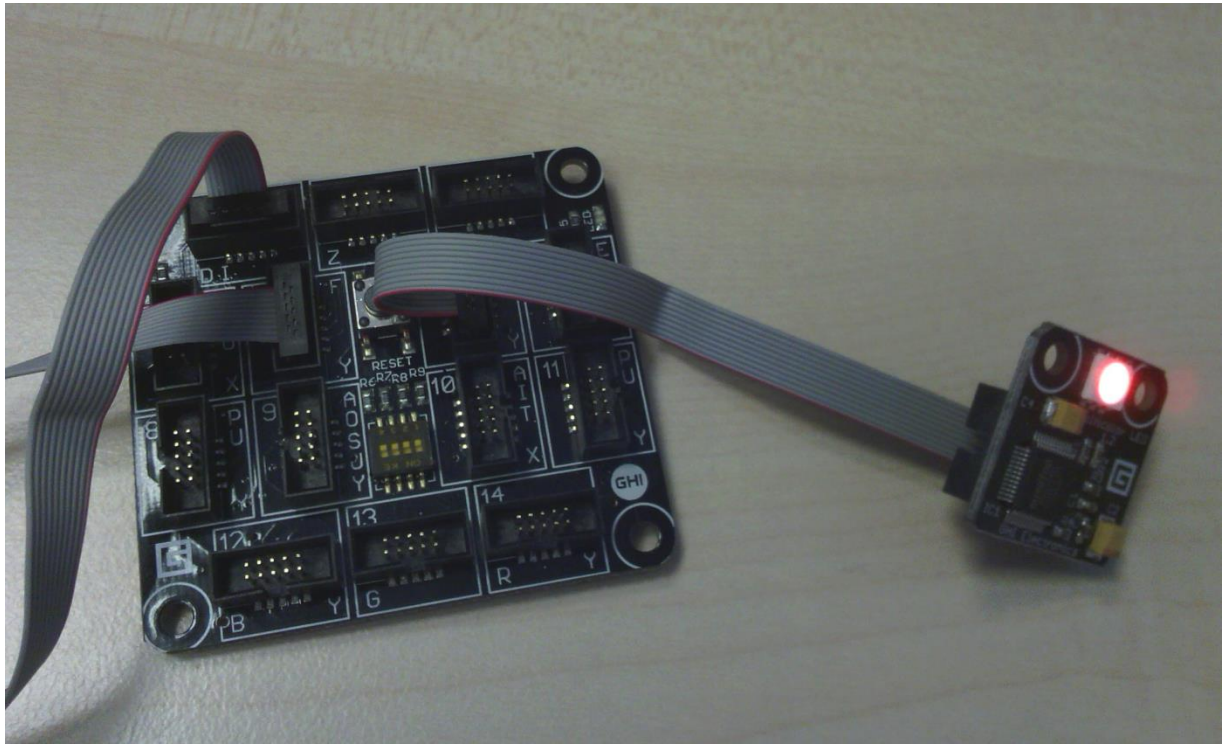
Coding…

button_ButtonPressed method

```
void button_ButtonPressed(Button sender, Button.ButtonState state)
{
   led.BlinkRepeatedly(GT.Color.Red);
}
```

- After the button press this method runs. *Sender* contains the reference for the corresponding button, and *State* the button actual state.
- The LED turns on, and blinks repeatedly with RED color (GT.Color.Red).

The first application:



Exercise #1

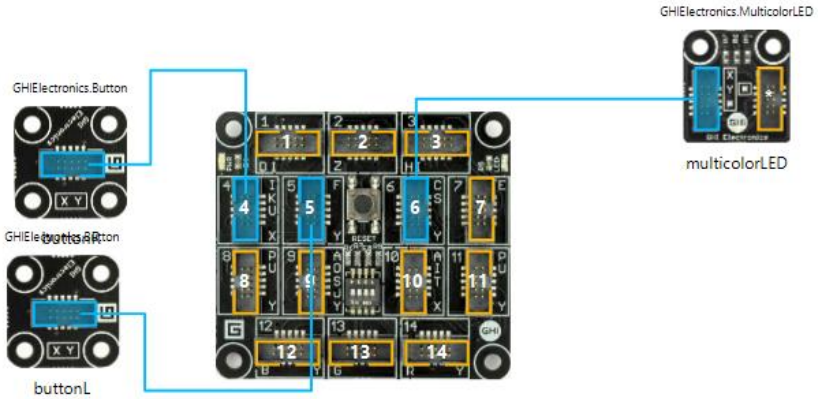
Create an application with 2 buttons and 1 multicolor led

- Buttons:
  - ButtonLeft
  - ButtonRight
- If Left button is pressed, blink one with **blue** color
- If Right button is pressed, blink one with **green** color

- Hints:

```
public class MulticolorLed : Module.DaisyLinkModule
        {…
        public void BlinkOnce(Color color);
        …}
```

Exercise #1




```
void ProgramStarted()
{
    Debug.Print("Program Started");
    buttonL.ButtonPressed += buttonL_ButtonPressed;
    buttonR.ButtonPressed += buttonR_ButtonPressed;
}

void buttonR_ButtonPressed(Button sender, Button.ButtonState state)
{
    multicolorLED.BlinkOnce(GT.Color.Green);
}

void buttonL_ButtonPressed(Button sender, Button.ButtonState state)
{
    multicolorLED.BlinkOnce(GT.Color.Blue);
}
```

Exercise #2

Modify the previous app to the following:

- If LeftButton is actually pressed, turn on its own led (simple red led)
- When the button released, turn it off!

- Hint:

```
public class Button : Module
        {…
        public void ToggleLED();
        public void TurnLEDOff();
        public void TurnLEDOn();
        …}
```

Exercise #2

```
void ProgramStarted()
{
    Debug.Print("Program Started");
    buttonL.ButtonPressed += buttonL_ButtonPressed;
    buttonR.ButtonPressed += buttonR_ButtonPressed;
    buttonL.ButtonReleased += buttonL_ButtonReleased;
    buttonR.ButtonReleased += buttonR_ButtonReleased;
}

void buttonR_ButtonPressed(Button sender, Button.ButtonState state)
{
    multicolorLED.BlinkOnce(GT.Color.Green);
    buttonR.TurnLedOn();
}

void buttonL_ButtonPressed(Button sender, Button.ButtonState state)
{
    multicolorLED.BlinkOnce(GT.Color.Blue);
    buttonL.TurnLedOn();
}

void buttonL_ButtonReleased(Button sender, Button.ButtonState state)
{
    buttonL.TurnLedOff();
}

void buttonR_ButtonReleased(Button sender, Button.ButtonState state)
{
    buttonR.TurnLedOff();
}
```

Exercise #3

- RightButton pressed: start a timer, and turn on a multicolor led in every second with a random color
- LeftButton pressed: stop the timer, turn off the led

- Hint:

```
public class MulticolorLed : Module.DaisyLinkModule
        {…
        SetBlueIntensity(int intensity);  //0-255
        SetRedIntensity(int intensity);   //0-255
        SetGreenIntensity(int intensity); //0-255
        …}
public class Random
        {…
        public virtual int Next(int maxValue);
        …}
```

```
GT.Timer timer = new GT.Timer(1000); // every second (1000ms)
timer.Tick +=<tab><tab> // (Timer.TickEventHandler)
timer.Start();
```

```
public partial class Program
{
    Random r;
    GT.Timer timer;
    void ProgramStarted()
    {
        Debug.Print("Program Started");
        buttonL.ButtonPressed += buttonL_ButtonPressed;
        buttonR.ButtonPressed += buttonR_ButtonPressed;
        timer = new GT.Timer(1000);
        timer.Tick += timer_Tick;
        r = new Random();
    }

    void timer_Tick(GT.Timer timer)
    {
        multicolorLED.SetBlueIntensity(r.Next(255));
        multicolorLED.SetGreenIntensity(r.Next(255));
        multicolorLED.SetRedIntensity(r.Next(255));
    }

    void buttonR_ButtonPressed(Button sender, Button.ButtonState state)
    {
        timer.Start();
    }

    void buttonL_ButtonPressed(Button sender, Button.ButtonState state)
    {
        timer.Stop();
        multicolorLED.TurnOff();
    }
}
```

Exercise #4

Create a „police light bar”

- Red and Blue flashing (2 multicolor led)