Small Computer Software Embedded Breakout
EE/CS 356 Project 6 2016

For project 6 you will be using the .NET Gadgeteer. The .NET Gadgeteer is a rapid development platform that is a standard maintained by Microsoft for standardizing the connections between mainboards and modules. You will be using the FEZ Cerberus mainboard for development as well as several of the Gadgeteer modules.

You will be using the platform to design a handheld embedded breakout game.

History

Breakout is an arcade game developed and published by Atari, It was conceptualized by Nolan Bushnell and Steve Bristow, influenced by the 1972 Atari arcade game Pong, and built by Steve Wozniak aided by Steve Jobs. The game was ported to multiple platforms and upgraded to video games such as Super Breakout. In addition, Breakout was the basis and inspiration for books, video games, film, and the Apple II personal computer.

Game Play

Breakout begins with eight rows of bricks, with each two rows a different color. The color order from the bottom up is yellow, green, orange and red. Using a single ball, the player must knock down as many bricks as possible by using the walls and/or the paddle below to ricochet the ball against the bricks and eliminate them. If the player's paddle misses the ball's rebound, he or she loses a turn. The player has three turns to try to clear two screens of bricks. Yellow bricks earn one point each, green bricks earn three points, orange bricks earn five points and the top-level red bricks score seven points each. The paddle shrinks to one-half its size after the ball has broken through the red row and hit the upper wall. Ball speed increases at specific intervals: after four hits, after twelve hits, and after making contact with the orange and red rows.

Minimum Requirements

For project 6 you must implement a basic version of the popular game breakout. Your game must allow two players, keep score, and announce a winner at the end of the game. Your project DOES NOT need to include the blocks at the top of the screen so to meet the minimum requirement for the project the score will be based on the number of times the player can hit back the ball. In this aspect the game is much more like PONG. Each player should be given 3

turns the player with the highest score at the end of the game should be called out as the winner.

Additions

Breakout bars at the top of the screen will be considered an addition

Increasing the “speed” of the ball will be considered an addition

Changing Paddle size (ON PURPOSE) will be considered an addition

…

Installing:

The NETMF Gadgeteer package does not yet support Visual Studio 2013 so you will need to install 2013 if you have not already. 2013 has already been installed in KC136 and I believe 139.

Installation instruction: https://www.ghielectronics.com/support/netmf

Once you have VS2013 installed download and install .Net Micro Framework 4.3

unzip and install MicroFrameworkSDK.MSIand netmfvs2013.vsix

Download and Install Microsoft .NET Gadgeteer Core

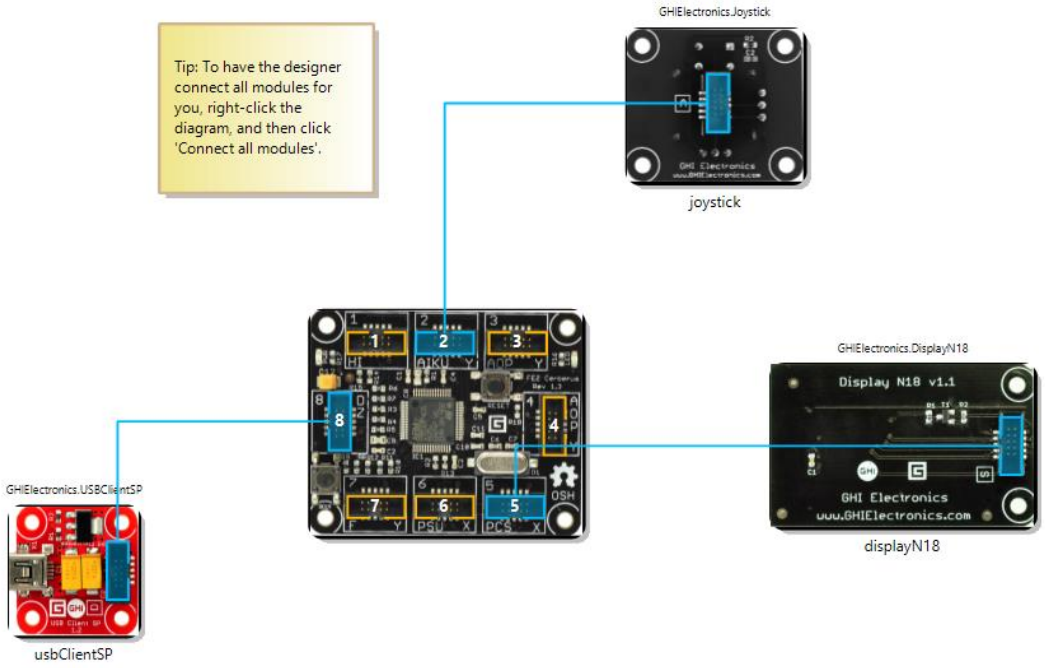
Download and Install GHI Electronics NETMF SDK 2015 R1

You should then have your development environment setup for the .net Gadgeteer.

From there follow this tutorial to setup your project

.Net Gadgeteer for Beginners

Setup of Gadgeteer, you will be using the N18 Display, Joystick, and USB Client SP for this project. We will use the usbClientSP for power so no external power supply will be needed.



Drawing on the LCD:

The screen can be written to by drawing a bmp onto it in a specific location. However, there is a catch the amount of memory in the Cerberus does not allow you to create a bmp file the size of the screen so you must draw and update only the portions of the screen as needed. Example:

```
Bitmap Scorebitmap = new Bitmap(120, 12);
```

This creates a bitmap that is 120 pixels wide by 12 pixels high.

The bitmap can be drawn on just as you have in past projects.

Example:

```
string sscore = "    " + score + "                          " + lives;
```

Draw a rectangle and fille with color white:

```
Scorebitmap.DrawRectangle(Color.White, 1, 0, 0, 120, 12, 0, 0, Color.White, 0, 0,
Color.White, 120, 12, 0xFF);
```

Draw Black text on the white rectangle:

```
Scorebitmap.DrawText(sscore, Resources.GetFont(Resources.FontResources.small),
Color.Black, 0, 0);
```

Finally Draw the bmp on the display, at location (0,0)

```
displayN18.Draw(Scorebitmap,0,0);
```

To allow the game to play you will need to create a Timer that will be used to run the game remember this is C# so it is event driven programming.

Example. This code will create a timer that times out every 33ms note I have added an event handler (timer_TimeOut) that will be ran each time the timer runs out. You can think of this as your while (true) loop.

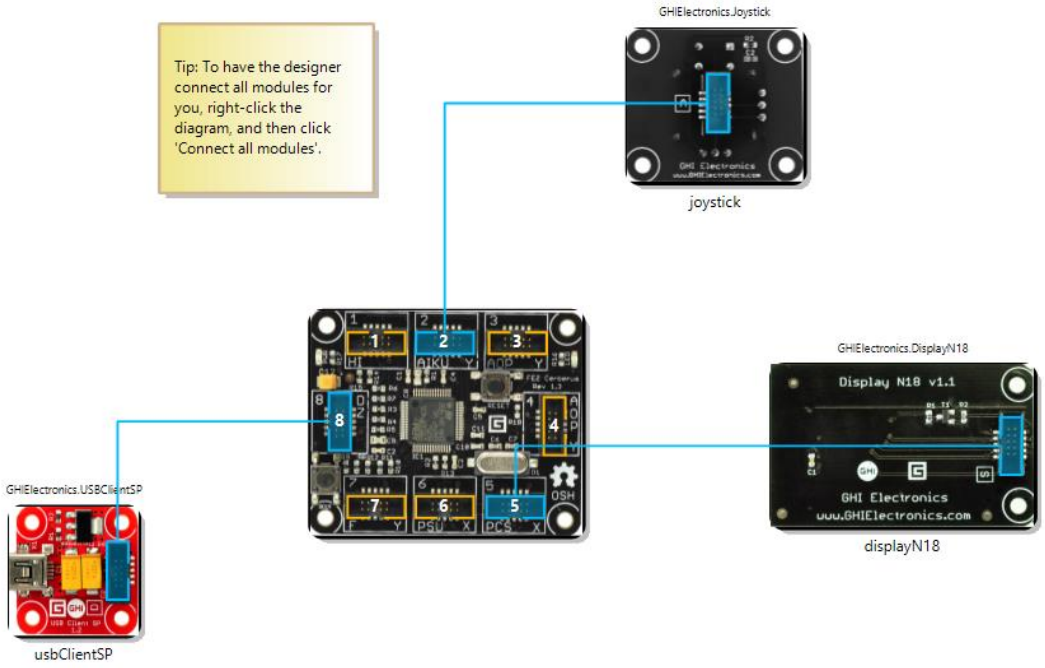
```
GT.Timer timer = new GT.Timer(33); // every second (1000ms)
timer.Tick += new GT.Timer.TickEventHandler(timer_TimeOut);
timer.Start();
```

Once you have added the Joystick module to your design it will be available to you as an object called joystick. The position of the joystick can be read in Cartesian coordinates using:

```
 double xpos = joystick.GetPosition().X;
 double ypos = joystick.GetPosition().Y;
```

Good Luck!!

Setup of Gadgeteer, you will be usingthe N18 Display, Joystick, and USB Client SP for this project. We will use the usbClientSP for power so no external power supply will be needed.



Drawing on the LCD:

The screen can be written to by drawing a bmp onto it in a specific location. However, there is a catch the amount of memory in the Cerberus does not allow you to create a bmp file the size of the screen so you must draw and update only the portions of the screen as needed. Example:

```
Bitmap Scorebitmap = new Bitmap(120, 12);
```

This creates a bitmap that is 120 pixels wide by 12 pixels high.

The bitmap can be drawn on just as you have in past projects.

Example:

```
string sscore = "    " + score + "                          " + lives;
```

Draw a rectangle and fille with color white:

```
Scorebitmap.DrawRectangle(Color.White, 1, 0, 0, 120, 12, 0, 0, Color.White, 0, 0,
Color.White, 120, 12, 0xFF);
```

Draw Black text on the white rectangle:

```
Scorebitmap.DrawText(sscore, Resources.GetFont(Resources.FontResources.small),
Color.Black, 0, 0);
```

Finally Draw the bmp on the display, at location (0,0)

```
displayN18.Draw(Scorebitmap,0,0);
```

To allow the game to play you will need to create a Timer that will be used to run the game remember this is C# so it is event driven programming.

Example. This code will create a timer that times out every 33ms note I have added an event handler (timer_TimeOut) that will be ran each time the timer runs out. You can think of this as your while (true) loop.

```
GT.Timer timer = new GT.Timer(33); // every second (1000ms)
timer.Tick += new GT.Timer.TickEventHandler(timer_TimeOut);
timer.Start();
```

Once you have added the Joystick module to your design it will be available to you as an object called joystick. The position of the joystick can be read in Cartesian coordinates using:

```
 double xpos = joystick.GetPosition().X;
 double ypos = joystick.GetPosition().Y;
```

Good Luck!!