Nanyang Polytechnic

School of Information Technology

Course	:	Diploma in Infocomm & Security (ITDF12)
Module	:	Sensor Technologies and Project (ITP272)

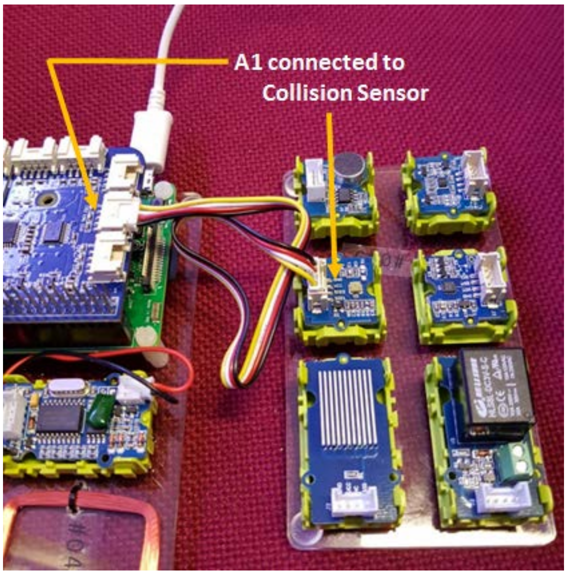
Raspberry Pi Practical : Programming Raspberry Pi with Grove Collision (Vibration) Sensor

Objectives:

- Learn how to interface with Analog Input to get the Collision Sensor readings
- Learn how to create program to interface with the Collision Sensor.

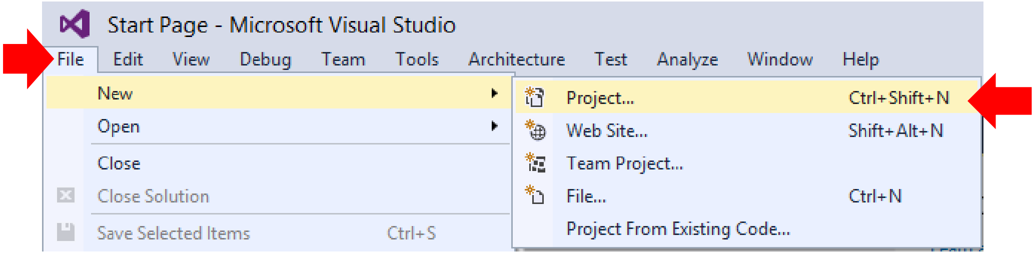
Exercise 1: Creating CollisionDetect

1. Today we will be creating a program that will read data from a collision sensor.
2. We shall use Analog port **A1** of the GrovePi for the Collision (Vibration) Sensor. Ensure that you have port **A1** connected to the **Collision Sensor**.

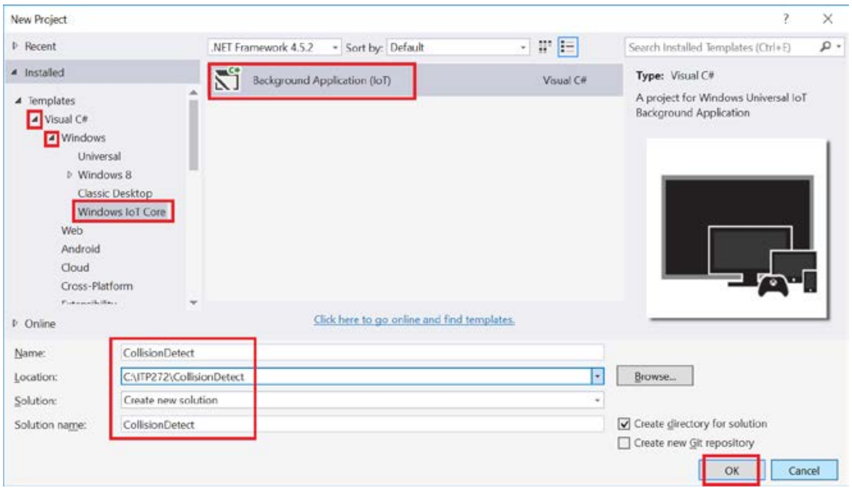


Collision Sensor

3. Start Visual Studio 2015, Click on File – New - Project.



4. From the Installed Templates, select Visual C# – Windows IoT Core – Background Application. Name your project as **CollisionDetect**, save it in you ITP272 folder and Click OK.



Type: Visual C#

A project for Windows Universal IoT Background Application

Name: CollisionDetect

Location: C:\ITP272\CollisionDetect

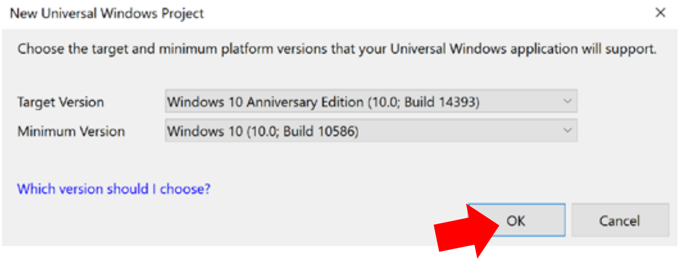
Solution: Create new solution

Solution name: CollisionDetect

☑ Create directory for solution

□ Create new Git repository

5. You should see this pop-up. Click OK.



New Universal Windows Project

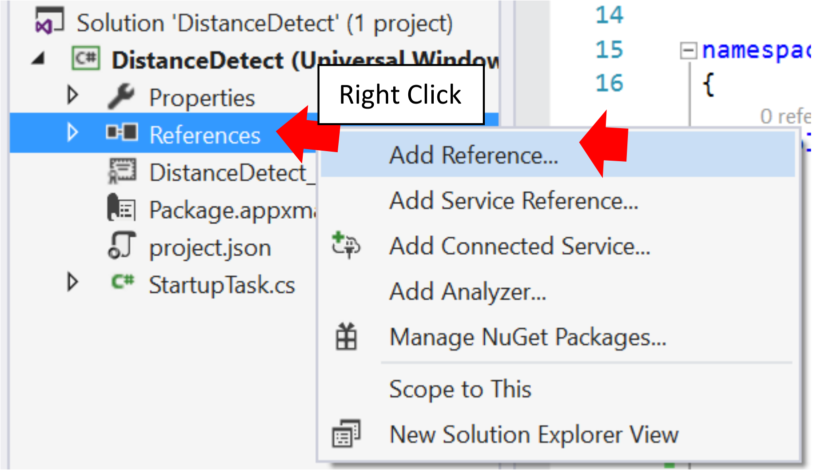
Choose the target and minimum platform versions that your Universal Windows application will support.

Target Version: Windows 10 Anniversary Edition (10.0; Build 14393)

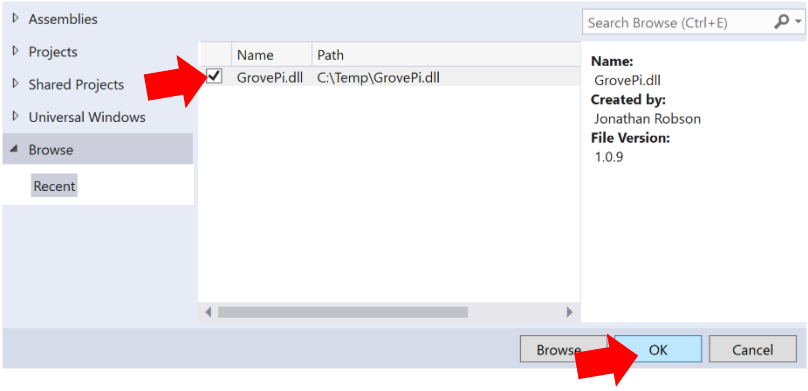
Minimum Version: Windows 10 (10.0; Build 10586)

6. At the solution explorer on the right side of the screen, check that you have StartupTask.cs. This is the startup program file. Double Click on Startup Task.cs.

7. Add Reference for the GrovePi. Right Click on References on Solution Explorer and Click on Add Reference.



8. Download and select GrovePi.dll as usual. It should appear in your Recent if you've used it before. Check on it and Click OK.



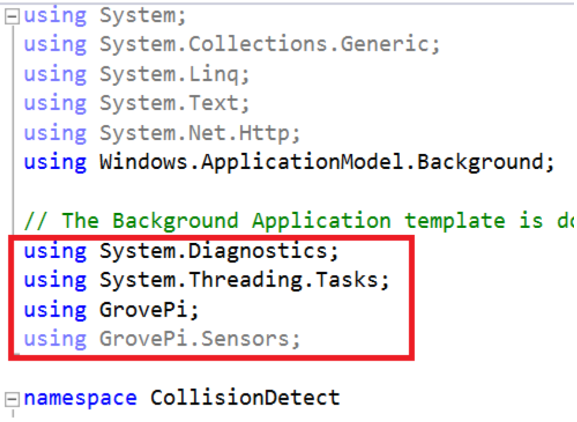
	Name	Path
☑	GrovePi.dll	C:\Temp\GrovePi.dll

Name: GrovePi.dll

Created by: Jonathan Robson

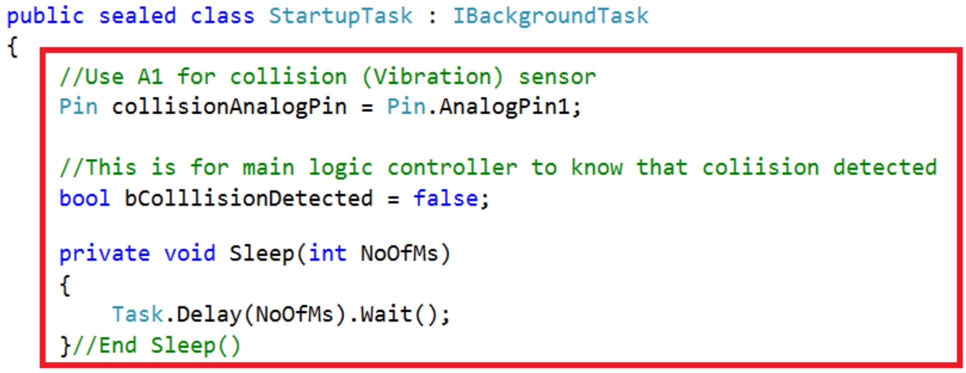
File Version: 1.0.9

9. Add in the following namespace codes



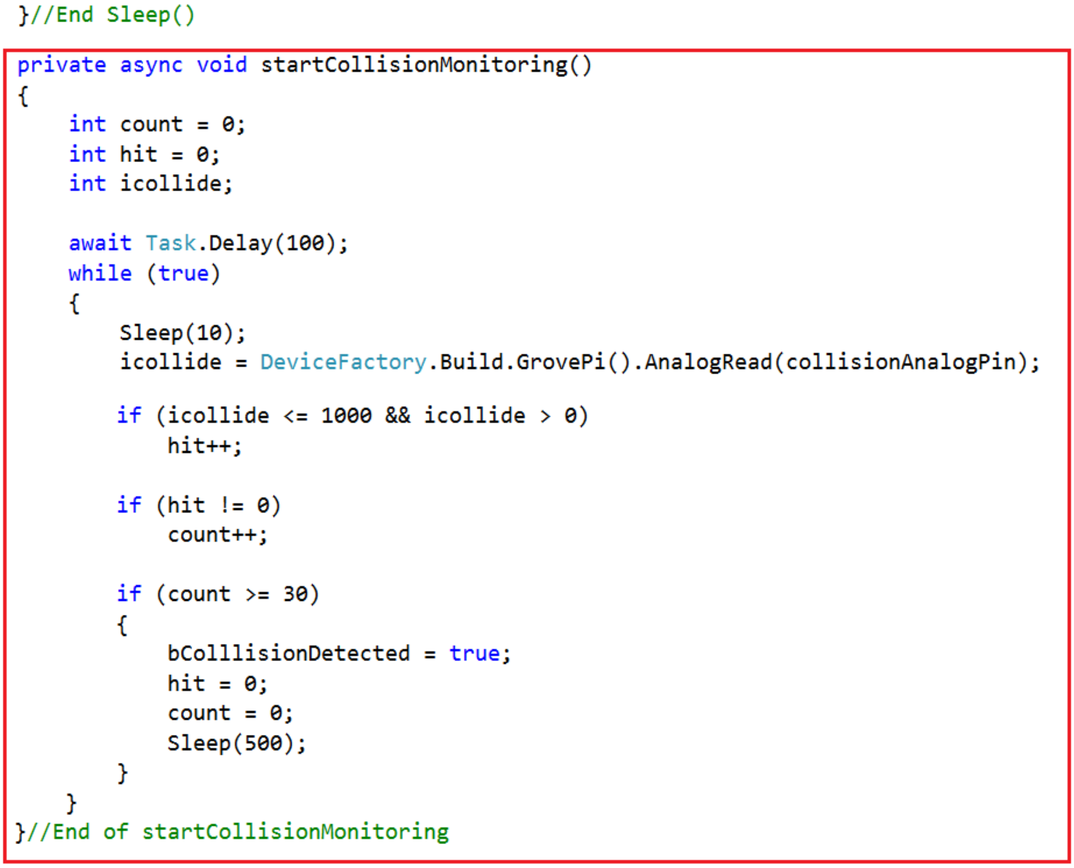
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Net.Http;
using Windows.ApplicationModel.Background;

// The Background Application template is d
using System.Diagnostics;
using System.Threading.Tasks;
using GrovePi;
using GrovePi.Sensors;

namespace CollisionDetect
```

10. Add in these codes in the StartupTask.cs



```
public sealed class StartupTask : IBackgroundTask
{
    //Use A1 for collision (Vibration) sensor
    Pin collisionAnalogPin = Pin.AnalogPin1;

    //This is for main logic controller to know that coliision detected
    bool bColllisionDetected = false;

    private void Sleep(int NoOfMs)
    {
        Task.Delay(NoOfMs).Wait();
    }//End Sleep()
```

11. Add in the codes below the Sleep(). This is to create Collision monitoring methods

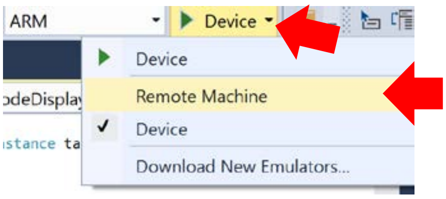


```
}//End Sleep()

private async void startCollisionMonitoring()
{
    int count = 0;
    int hit = 0;
    int icollide;

    await Task.Delay(100);
    while (true)
    {
        Sleep(10);
        icollide = DeviceFactory.Build.GrovePi().AnalogRead(collisionAnalogPin);

        if (icollide <= 1000 && icollide > 0)
            hit++;

        if (hit != 0)
            count++;

        if (count >= 30)
        {
            bColllisionDetected = true;
            hit = 0;
            count = 0;
            Sleep(500);
        }
    }
}//End of startCollisionMonitoring
```

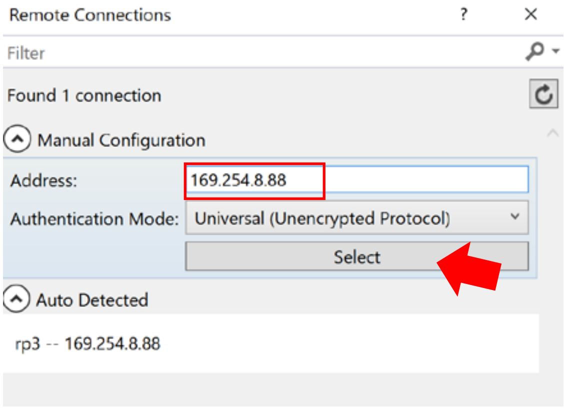
12. Finally, add the codes in the **Run()** method.

```
public void Run(IBackgroundTaskInstance taskInstance)
{
    //
    // TODO: Insert code to perform background work
    //
    // If you start any asynchronous methods here, prevent the task
    // from closing prematurely by using BackgroundTaskDeferral as
    // described in http://aka.ms/backgroundtaskdeferral
    //

    startCollisionMonitoring();
    int no = 0;

    Debug.WriteLine("Start Collision Detect!");
    while (true)
    {
        Sleep(300);
        if (bColllisionDetected == true)
        {
            Debug.WriteLine("collision detected " + no++);

            //must remember to clear after process
            bColllisionDetected = false;
        }
    }
}
```

13. In order to deploy the program to your hardware, click on the Drop Down button beside the Device and Select Remote Machine.



14. Key in the address manually like shown below. Click on **Select** after that.

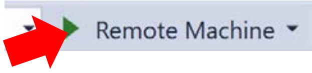


Address: 169.254.8.88

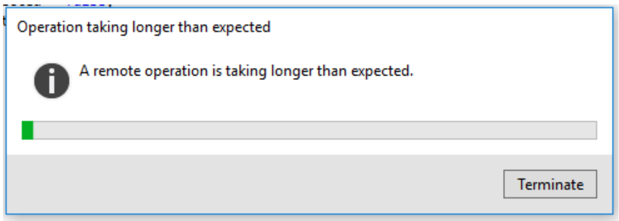
Authentication Mode: Universal (Unencrypted Protocol)

rp3 -- 169.254.8.88

15. Click on Remote Machine to Run the Program.

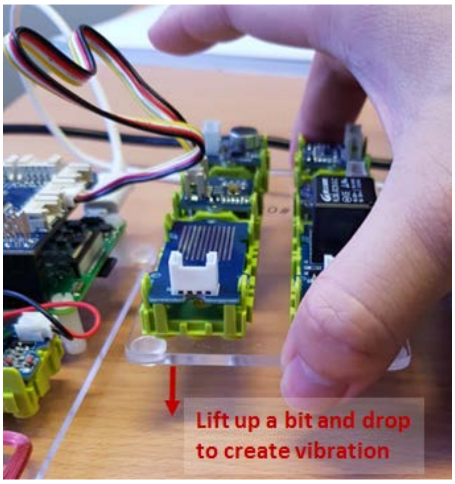


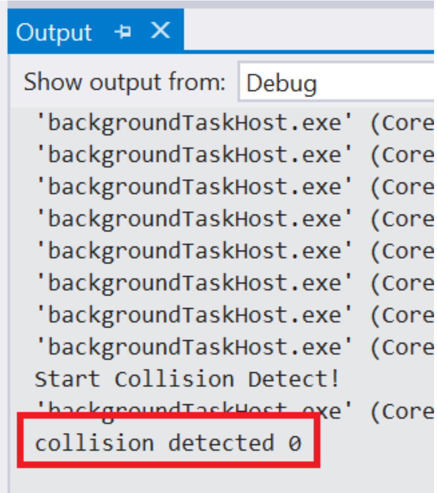
16. You may see the following warning but it is fine. Every time when you deploy a project for the first time on the hardware, it will take a long time to deploy. Just wait for a while and it should be deployed successfully.


Operation taking longer than expected

A remote operation is taking longer than expected.

17. Upon successful deployment, try to lift your sensor board up a little bit and drop it down onto the table to create a vibration. You should see the vibration detected and reported by the sensor.





```
'backgroundTaskHost.exe' (Core
'backgroundTaskHost.exe' (Core
'backgroundTaskHost.exe' (Core
'backgroundTaskHost.exe' (Core
'backgroundTaskHost.exe' (Core
'backgroundTaskHost.exe' (Core
'backgroundTaskHost.exe' (Core
'backgroundTaskHost.exe' (Core
Start Collision Detect!
'backgroundTaskHost.exe' (Core
collision detected 0
```

You've just learnt how to use the Collision (Vibration) sensor to detect vibration. Please understand the operation of the sensor and the codes and determine whether it is suitable for your project.

==End of Practical==