



## School of Information Technology

---

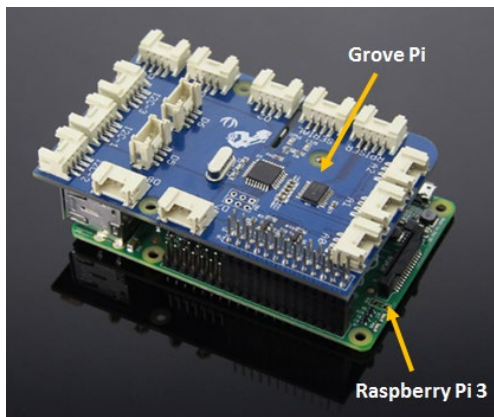
Course	:	Diploma in Infocomm & Security (ITDF12)
Module	:	Sensor Technologies and Project (ITP272)

---

Raspberry Pi Practical : Programming Raspberry Pi

### Objectives:

- Familiarise with Raspberry Pi Kit.
- Learn how to verify connectivity of the Raspberry Pi
- Learn how to create and deploy a simple Raspberry Pi Program

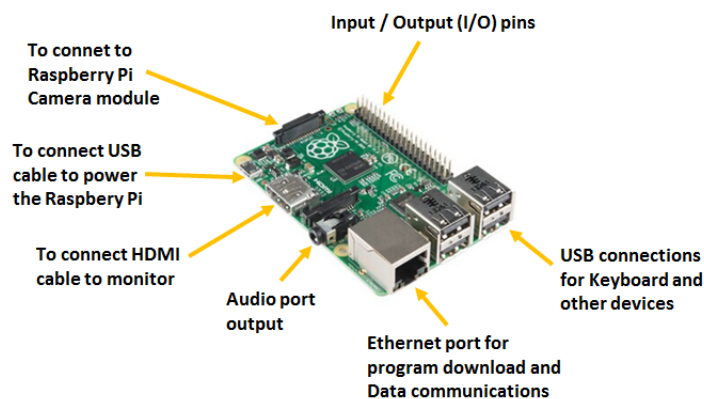


The Lab Kit Set is made up of 2 items.

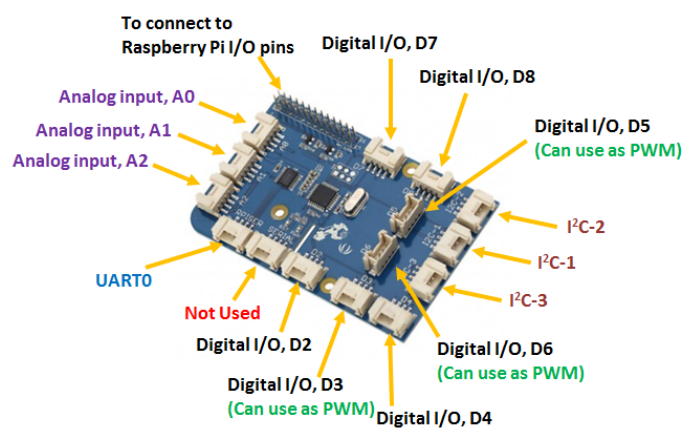
- Raspberry Pi 3 (Version 3 of Raspberry Pi)
- Grove Pi+

The Raspberry Pi 3 is a small single board computer that we deploy our application to communicate with the sensors. The Grove Pi+ is a processor which provide direct connection to Grove Sensors. It helps to convert raw sensor values coming from the Grove sensors to suitable format to pass to the Raspberry Pi 3. It provides a library to help raspberry pi application interface with the Grove Sensors connected to it.

### Raspberry Pi Input Output interfaces

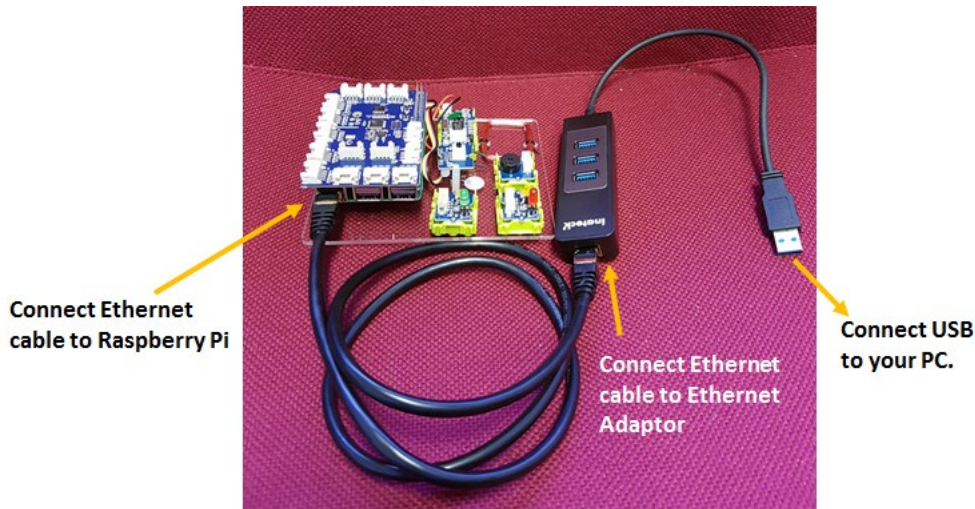


### Grove Pi+ Input Output interfaces



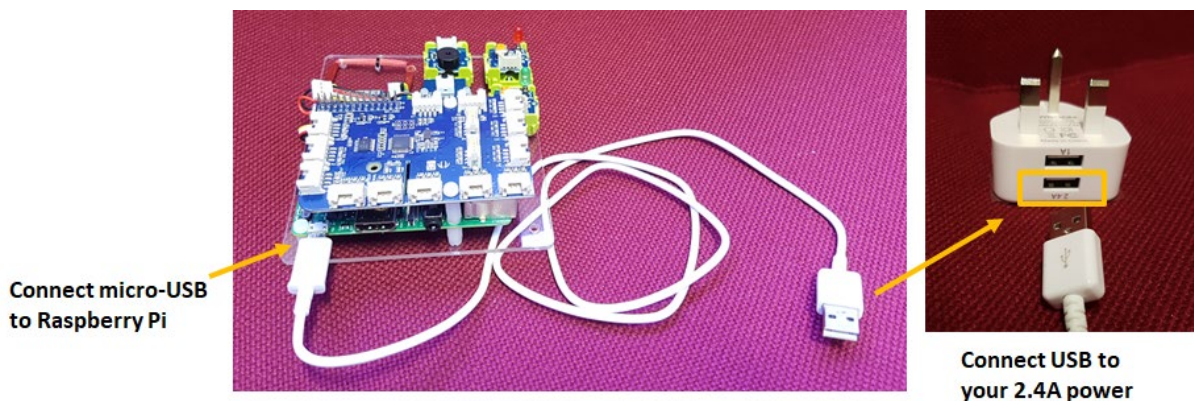
### Exercise 1: Verify Raspberry Pi connection

1. Connect the USB Ethernet Adaptor to the Raspberry pi and your PC as shown below.



This is to provide Ethernet connection between Raspberry Pi and your PC. The Ethernet connection is used to download and run your application to on the Raspberry Pi. It is also used for the Raspberry Pi to send data to your PC Windows Application (You'll learn in one of the labs to create a Windows application to communicate with Raspberry Pi)

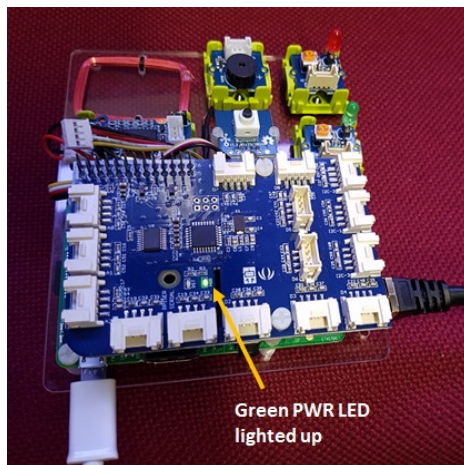
2. Connect the USB cable to power up the Raspberry Pi as shown below.



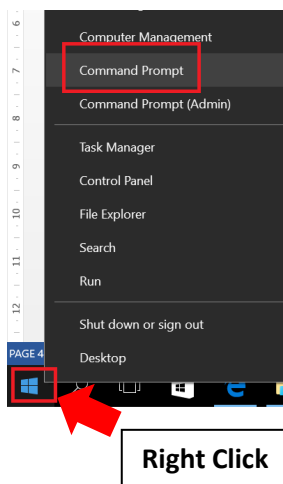
Connect the USB to the **2.4A** port of the power adaptor so that it has enough current to work. You may sometimes need to re-power a few times before the Raspberry Pi can work correctly. The next step can help you verify whether you have connection between your Raspberry Pi and PC. Always verify the connection is stable before doing your development.

## ITP272 Sensor Technologies and Project

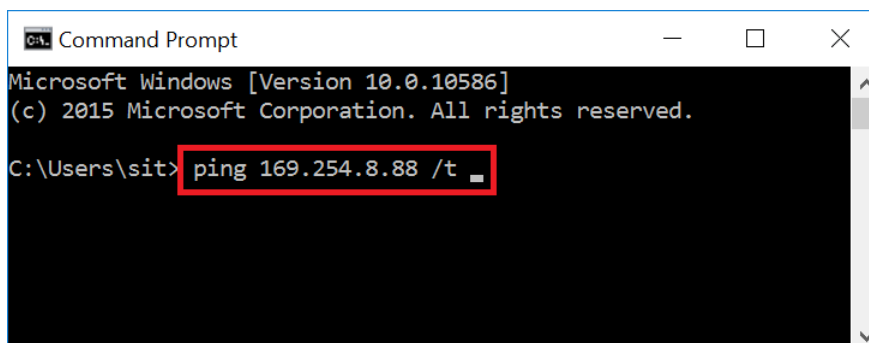
- After you've connected up your PC to Ethernet Adaptor and Raspberry Pi, it takes a while (approximately less than 2 minutes) before the Raspberry Pi and Ethernet Adaptor gets powered up and initialised. You should make the below 2 observations if everything is setup successfully.



- Right Click** on the Windows icon and Open up your Command Prompt



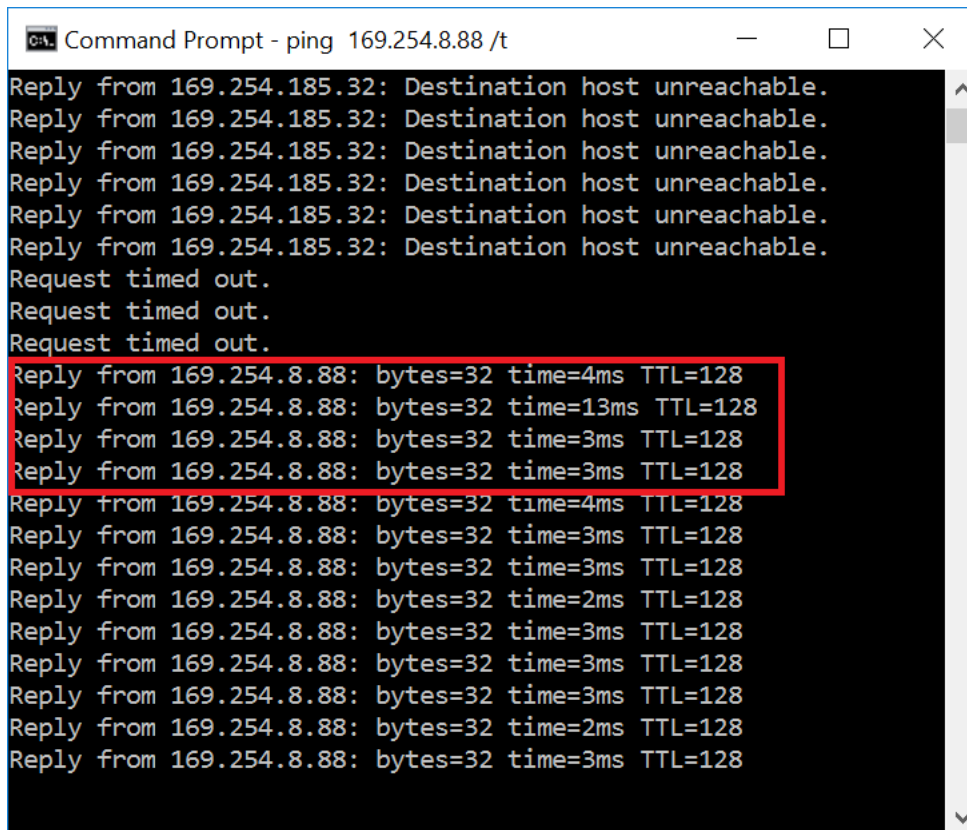
- Issue a ping command "ping 169.254.8.88 /t" as shown below



This command is to verify that the PC is able to communicate with the Raspberry Pi.

6. Verify that there is reply from the Raspberry Pi as shown below. You should verify there is consistent replies for at least 1 min to ensure connection is stable for development.

When Raspberry Pi and Ethernet is first starting up, there will be no reply for 1 to 2 minutes. After the Raspberry Pi and Ethernet has started up successfully, you should receive a reply.



```
Command Prompt - ping 169.254.8.88 /t
Reply from 169.254.185.32: Destination host unreachable.
Reply from 169.254.185.32: Destination host unreachable.
Reply from 169.254.185.32: Destination host unreachable.
Reply from 169.254.185.32: Destination host unreachable.
Reply from 169.254.185.32: Destination host unreachable.
Reply from 169.254.185.32: Destination host unreachable.
Request timed out.
Request timed out.
Request timed out.
Reply from 169.254.8.88: bytes=32 time=4ms TTL=128
Reply from 169.254.8.88: bytes=32 time=13ms TTL=128
Reply from 169.254.8.88: bytes=32 time=3ms TTL=128
Reply from 169.254.8.88: bytes=32 time=3ms TTL=128
Reply from 169.254.8.88: bytes=32 time=4ms TTL=128
Reply from 169.254.8.88: bytes=32 time=3ms TTL=128
Reply from 169.254.8.88: bytes=32 time=3ms TTL=128
Reply from 169.254.8.88: bytes=32 time=2ms TTL=128
Reply from 169.254.8.88: bytes=32 time=3ms TTL=128
Reply from 169.254.8.88: bytes=32 time=3ms TTL=128
Reply from 169.254.8.88: bytes=32 time=3ms TTL=128
Reply from 169.254.8.88: bytes=32 time=2ms TTL=128
Reply from 169.254.8.88: bytes=32 time=3ms TTL=128
```

If you do not see a reply, you can try to remove and re-connect the power to both Ethernet and Raspberry Pi.

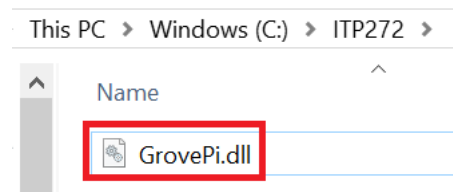
The following are some possible causes for unsuccessful connection

- Contact problem (Loose or not fully connected)
- USB Port not enough power or Faulty (try use separate USB port for Ethernet and Raspberry Pi. You may also try to switch to use other USB port as some USB ports may be able to supply more power)
- Cable faulty
- Ethernet or Raspberry Pi faulty
- Raspberry Pi corrupted

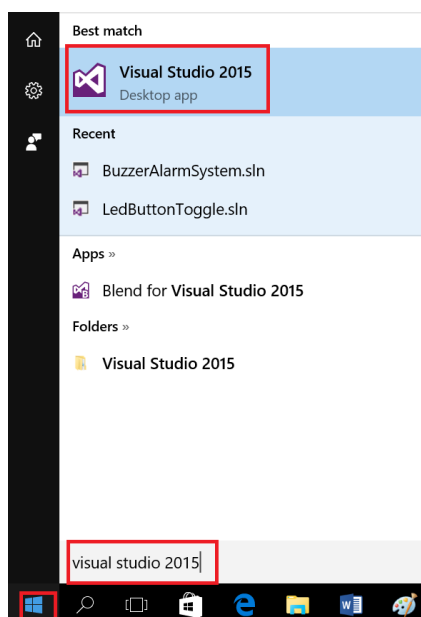
7. Once you have a stable connection with the Raspberry Pi, you're ready to develop application to download and run on the Raspberry Pi

### Exercise 2: Create Hello World Application on Raspberry Pi

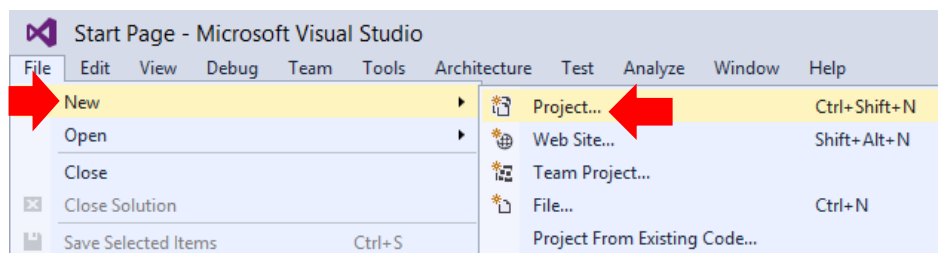
1. It is advisable to have a fixed folder to store all your Lab exercises. In our Lab practical, we shall assume a folder “ITP272” has been created in C:\. We will create our Visual studio project solution in this folder “C:\ITP272”. You can create the folder in your desktop if you like as long as you remember and specify it correctly throughout the Lab practical. Please create all solutions on the PC rather than in your Thumb drive. You can copy over to your Thumb drive later.
2. Download the “GrovePi.dll” from Blackboard and put it in the “ITP272” folder. This is the Grove Pi Library used to communicate with any Grove Sensor connected to the Grove Pi.



3. Click on Windows button, Type “Visual Studio 2015” and select the Vsual Studio 2015.



4. Click on “File – New – Project...”

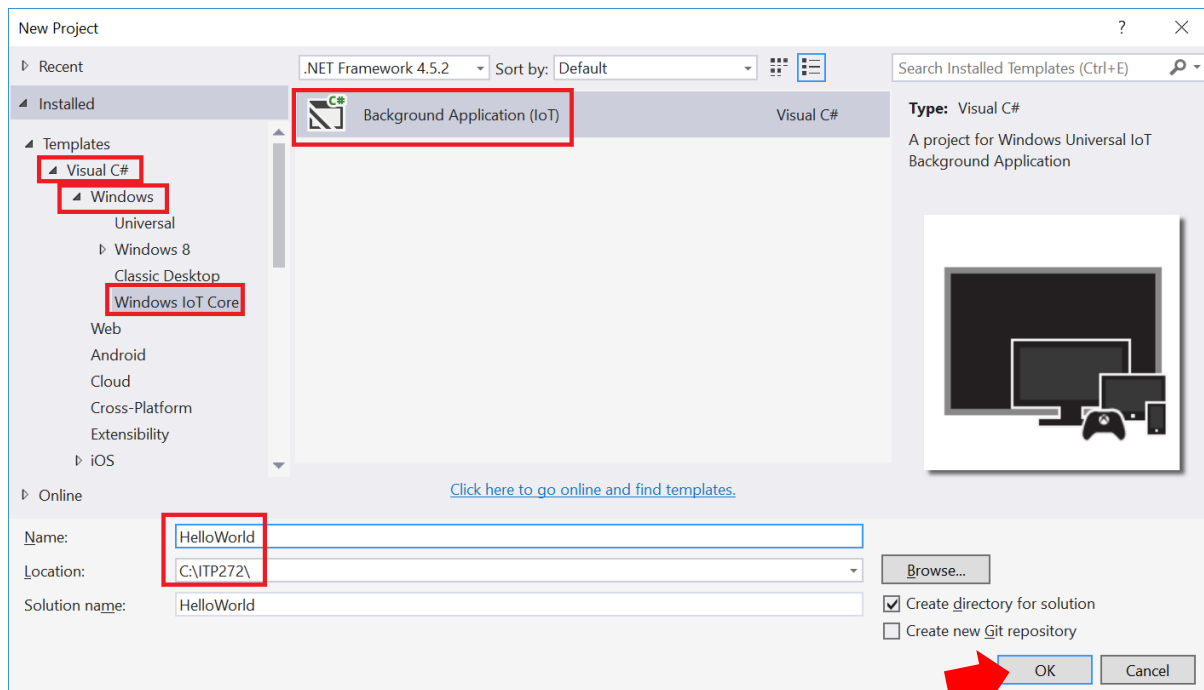




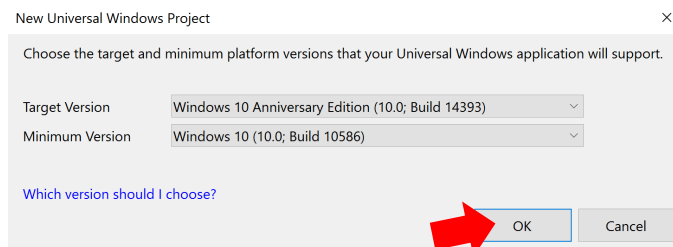
5. From the Installed Templates, select

**Visual C# – Windows - Windows IoT Core – Background Application (IoT)**

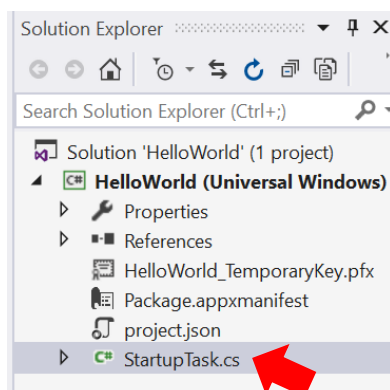
Name your project as “HelloWorld” and save it in your ITP272 folder and Click OK.



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



7. At the solution explorer, check that you have StartupTask.cs. This is the startup program file. Double Click on Startup Task.cs.



8. You should see the method

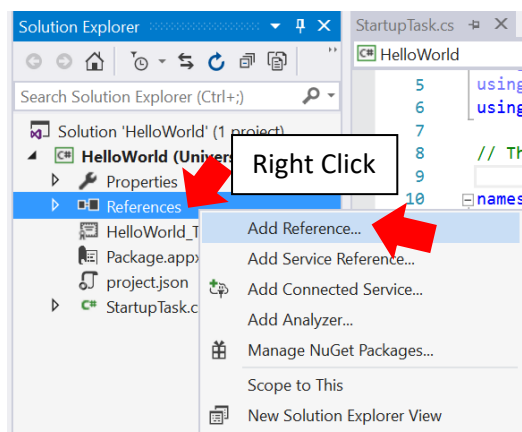
```
public void Run(IBackgroundTaskInstance taskInstance)

namespace HelloWorld
{
    0 references
    public sealed class StartupTask : IBackgroundTask
    {
        0 references
        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
            //
        }
    }
}
```

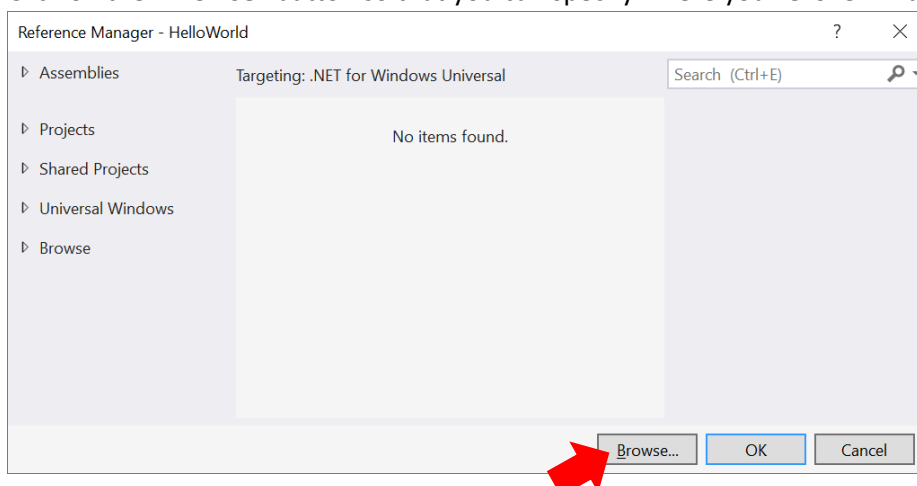
Your program starts running from this **Run** method.

9. We're now going to add the GrovePi library to your workspace.  
(Although we're not using the Library yet in this HelloWorld application, the steps are included here as you'll definitely need it in most of your other Practical).

To add the library, you need to add Reference. **Right Click** on **References** on Solution Explorer and Click on **Add Reference...**



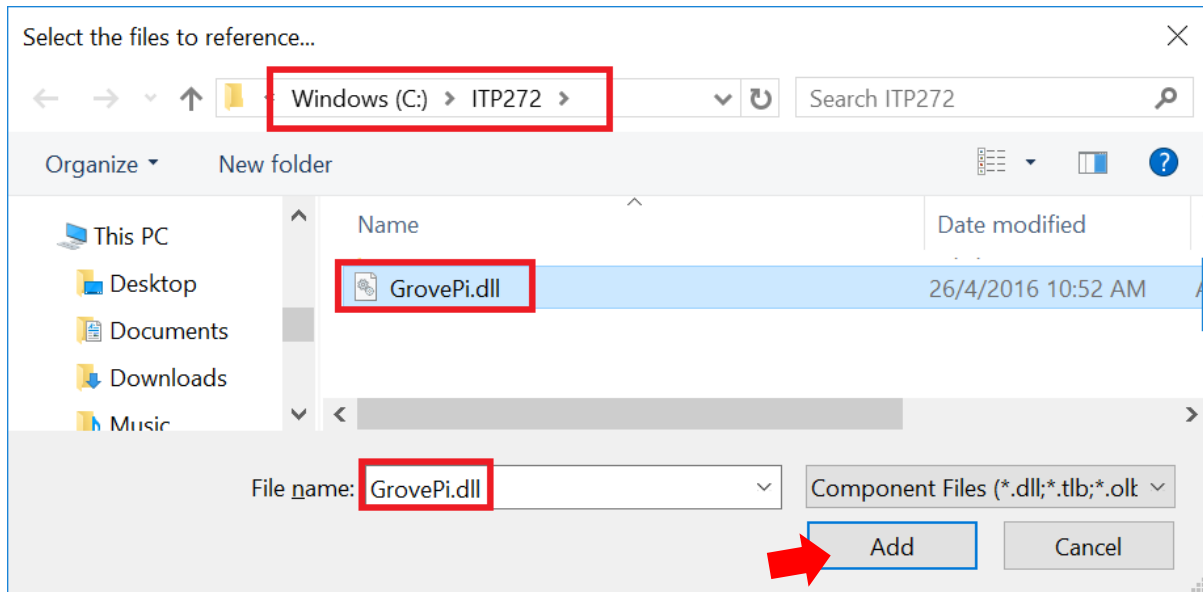
10. Click on the “Browse” button so that you can specify where your GrovePi Library is



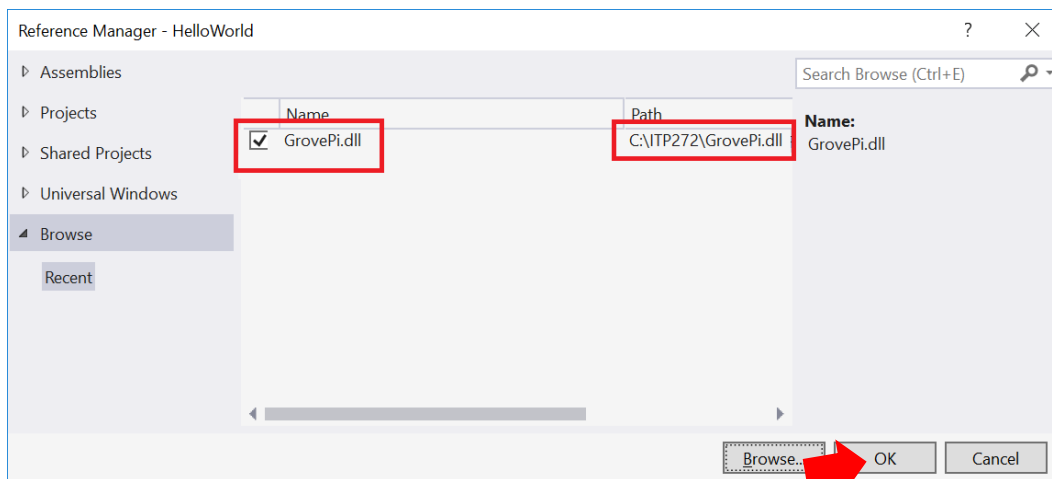


## ITP272 Sensor Technologies and Project

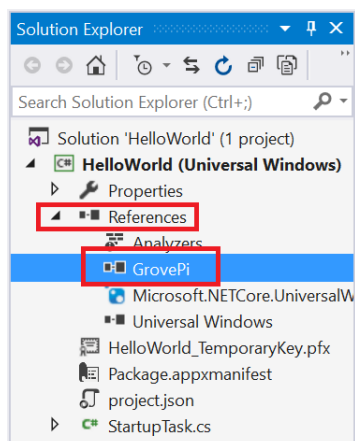
11. Navigate to the folder where you've downloaded your GrovePi.dll (In this practical, it is in C:\ITP272), select the "**GrovePi.dll**" and click on the "**Add**" button.



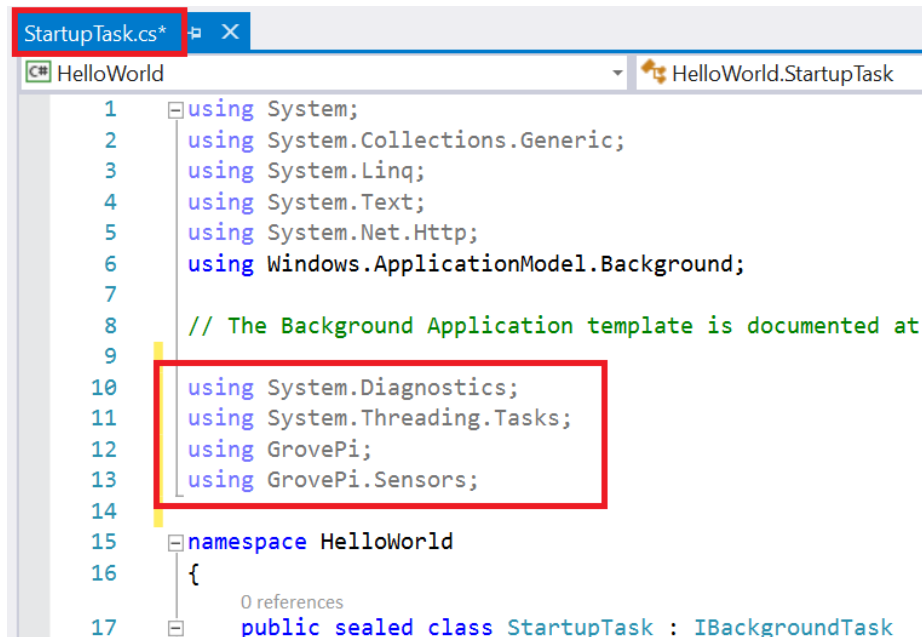
12. Verify that the "**GrovePi.dll**" is selected and click on the "**OK**" button.



13. Under "**References**" in the solution Explorer, you should see your "**GrovePi**" library



14. Go to your “StartupTask.cs” and type in the following codes above your namespace to include required libraries. It is fine to be in gray initially since you have not used them in the codes



```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Net.Http;
6 using Windows.ApplicationModel.Background;
7
8 // The Background Application template is documented at
9
10 using System.Diagnostics;
11 using System.Threading.Tasks;
12 using GrovePi;
13 using GrovePi.Sensors;
14
15 namespace HelloWorld
16 {
17     0 references
18     public sealed class StartupTask : IBackgroundTask
```

15. Add the codes (See below) to create a method “Sleep”. This method will be used extensively in the program to wait for a specified number of time in milliseconds. For example Sleep(300); will cause the program to wait for 300ms before executing the next program statement.

```
namespace HelloWorld
{
    0 references
    public sealed class StartupTask : IBackgroundTask
    {
        private void Sleep(int NoOfMs)
        {
            Task.Delay(NoOfMs).Wait();
        }
        0 references
        public void Run(IBackgroundTaskInstance taskInstance)
        {
```

16. A Raspberry Pi application needs to be running indefinitely (forever) checking on the sensors and deciding on what to do. So we need to have a while loop that is always running non-stop. So in our HelloWorld Application, we're going to create this while loop that keeps printing a Debug Message on the Output screen every 300 milliseconds.

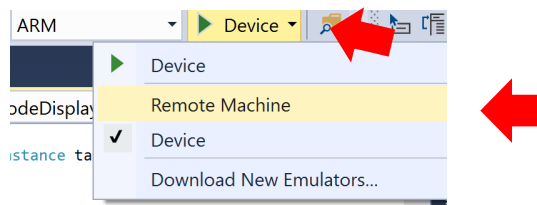
Add the codes below

```
public sealed class StartupTask : IBackgroundTask
{
    1 reference
    private void Sleep(int NoOfMs)
    {
        Task.Delay(NoOfMs).Wait();
    }

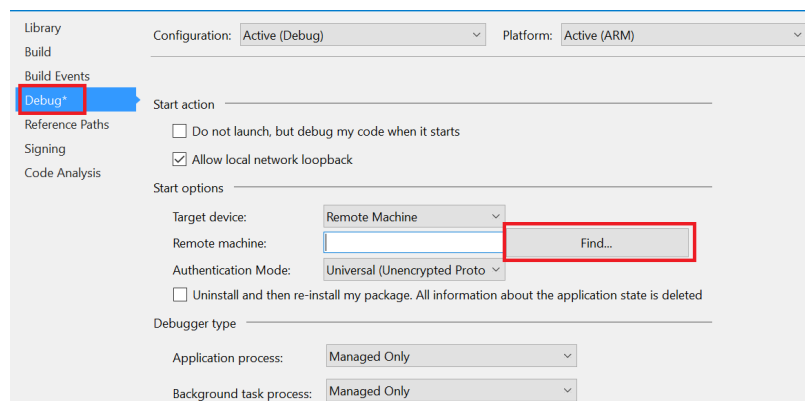
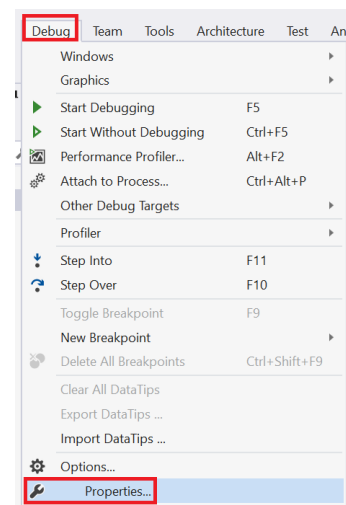
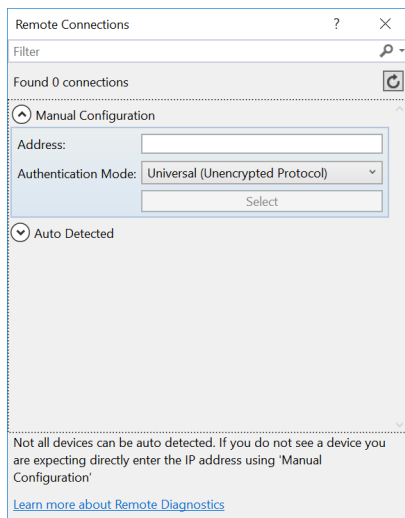
    0 references
    public void Run(IBackgroundTaskInstance taskInstance)
    {
        //
        // TODO: Insert code to perform background work
        //
        // If you start any asynchronous methods here, prevent the task
        // from closing prematurely by using BackgroundTaskDeferral as
  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

        while(true)
        {
            Sleep(300);
            Debug.WriteLine("Hello World");
        }
    }
}
```

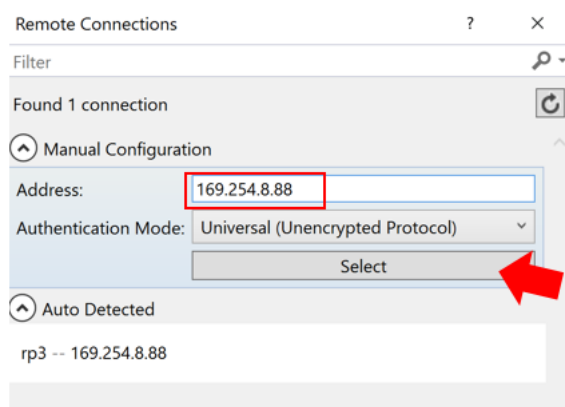
17. In order to deploy the program to your hardware, click on the Drop Down button beside the Device and Select Remote Machine.



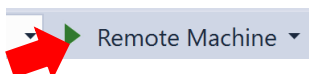
18. You should see this screen below. If you do not see the screen, do this



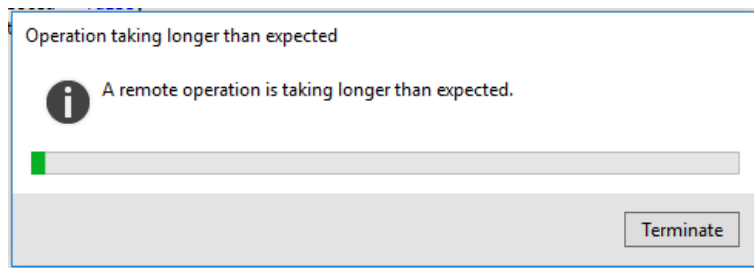
19. Key in the address “169.254.8.88” manually as shown below. Click on “Select” button after that.



20. Click on “Remote Machine” to deploy and run the Program on Raspberry Pi.



21. 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.



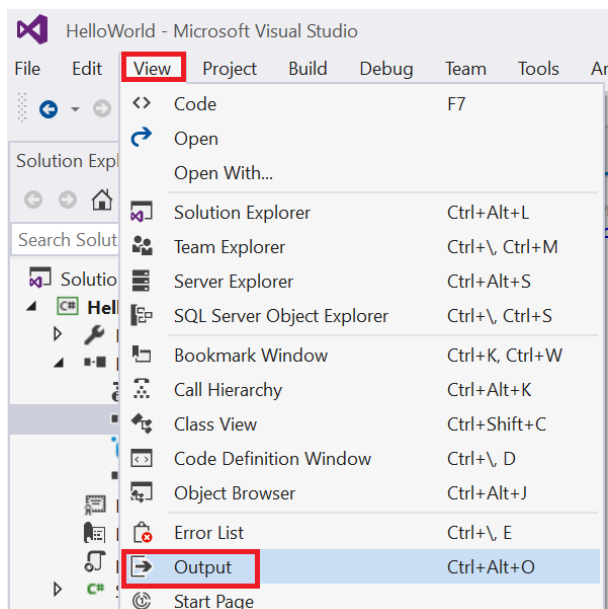
The status of the deployment can be found at the bottom of the window. If the processing bar is still showing, that means the program is still deploying and you have to wait.



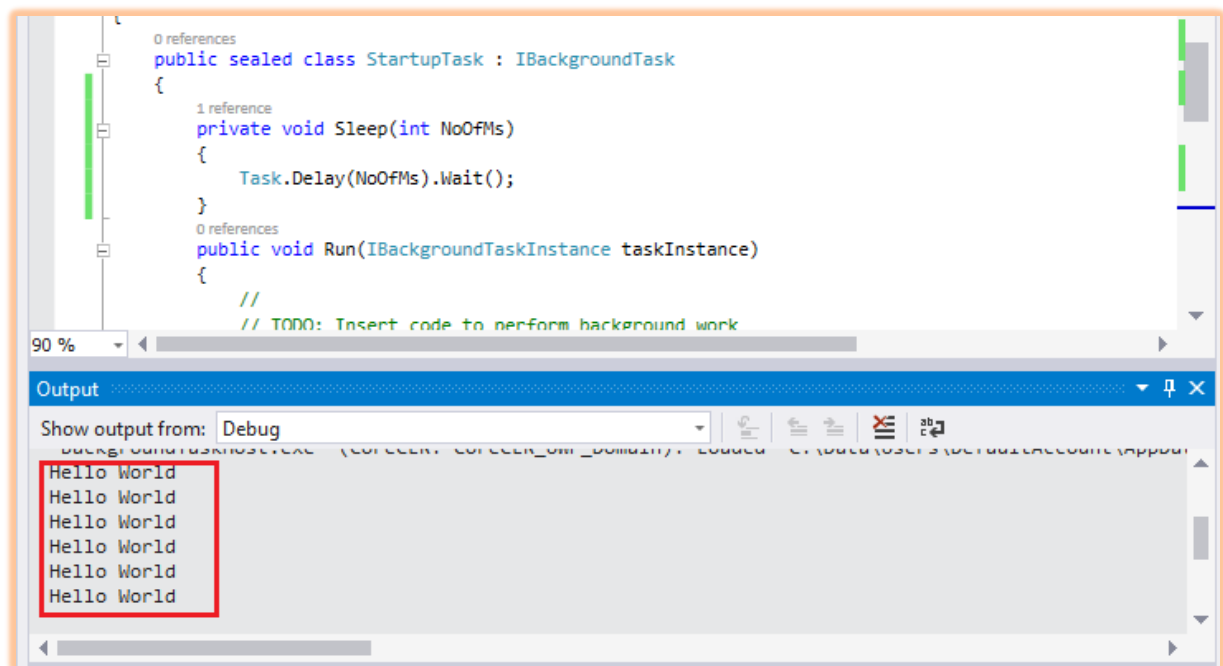
If the program is taking too long to deploy, terminate and do it again. The status will change to “Ready” state when program has been downloaded to Raspberry Pi successfully.



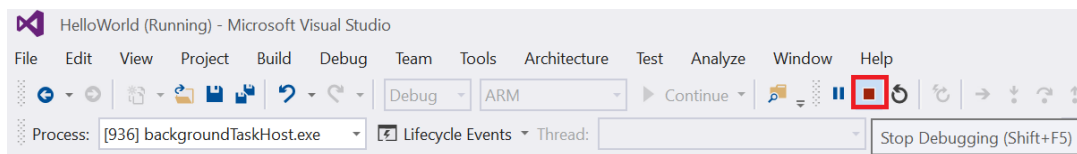
22. Select from the menu bar “**View – Output**” to show the Output window.



23. Verify the “Hello World” message is shown on the “Output” Window.



24. After verifying, you can click on “Stop Debugging” button to stop the application.



You have gone through a complete cycle from connecting up the Raspberry Pi hardware, setting up the development environment and deploying a simple program to run on the Raspberry Pi. The remaining Lab Practical may omit some of these steps to keep the lab short. You can always refer back to the steps here during development.

**==End of Practical==**