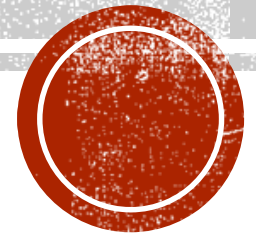
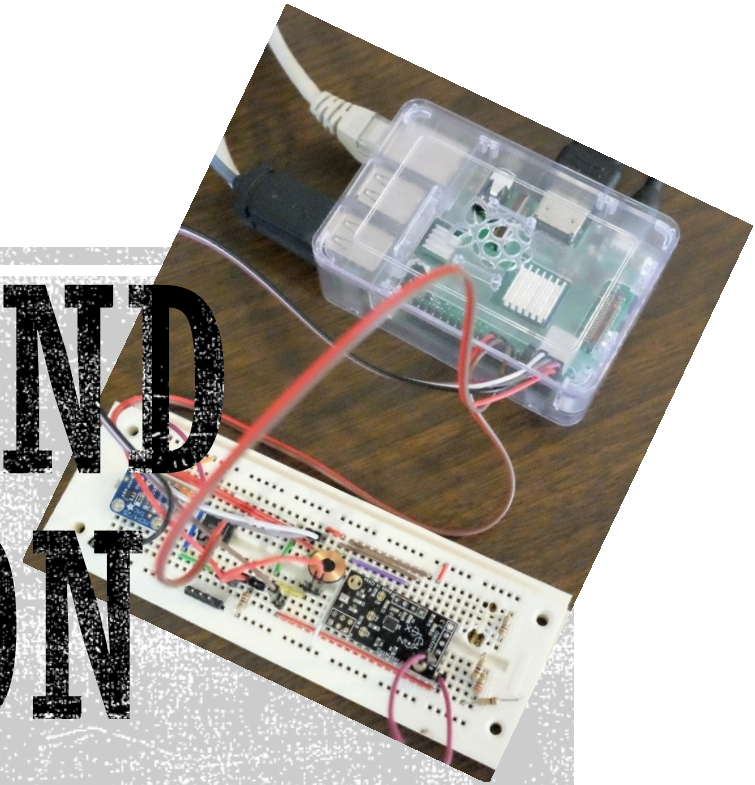


MAGNETOMETERS AND ACCELEROMETERS ON RASPBERRY PI



Comparison of *Raspbian* to *Windows 10 IoT Core* operating systems
and development tools

By Clarke Bowers



ABOUT THE PRESENTER

- Clarke D. Bowers
- Clarke Bowers Consulting, LLC
- <http://www.cbsoftwareengineering.com/>
- mailto: clarke@cbsoftwareengineering.com
- 35 years of industry experience
- He has architected and developed embedded systems, desktop applications, enterprise data warehouses, web sites and web services; cloud bases and locally hosted solutions
- He holds six patents



You can find this presentation
and all the examples at:
https://1drv.ms/f/s!Ar3pO7_GhJY9hOtmaUYC20lCv4Tl-g



AGENDA

Talk

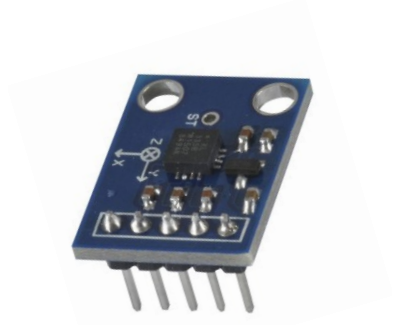
- Raspberry Pi hardware overview
- Raspbian Linux
- Windows 10 Core
- Others operating systems
- Installation, boot and shutdown
- Demonstration
 - Bobble head Santa
 - 3D Compass

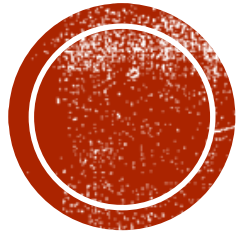
Lab 1

- Eclipse IDE Python & C++
- I²C Libraries
- Code examples of
 - Barometer
 - Magnetometer

Lab 2

- Visual Studio C# & .NET framework
- GPIO & Interrupts
- Cross platform development
- Code examples of
 - Magnetometer
 - Accelerometer





TALK



March 21

HARDWARE COMPARISON



Raspberry Pi 3 B+

- 2018
- Broadcom BCM2837B0, ARMv8 1.4GHz, 4-core 64-bit
- No GPU
- 1GB of RAM
- Micro SD slot (no storage included)
- No display
- 4 x USB 2.0
- Full-size HDMI
- Gigabit Ethernet, 802.11.b/g/n/ac Wi-Fi, Bluetooth 4.2
- 40 pin GPIO
- Not included: power supply, batteries, power switch, keyboard, mouse, heat sinks, case
- \$35



Panasonic CF-C1 Laptop

- 2010
- Intel Core i5 M520 2.4GHz, 2-core 64-bit
- Integrated 500 MHz GPU
- 2GB of RAM
- 250GB hard drive
- 12.1-inch 1280x800-pixel multi-touch-capable display
- 3 x USB 2.0
- DB15 VGA
- Gigabit Ethernet, 802.11n Wi-Fi, Bluetooth 2.1
- RJ11 Dialup Modem
- Included: power supply, batteries, power switch, keyboard, trackpad, fan
- \$2,000



IOT APPLICATION #1: TIDAL PROBE

- Mount Raspberry Pi, GPS, sensors and batteries on buoy
- Measure location, linear acceleration, angular acceleration, temperature and barometric pressure
- Transmit to cloud via G5 cellular



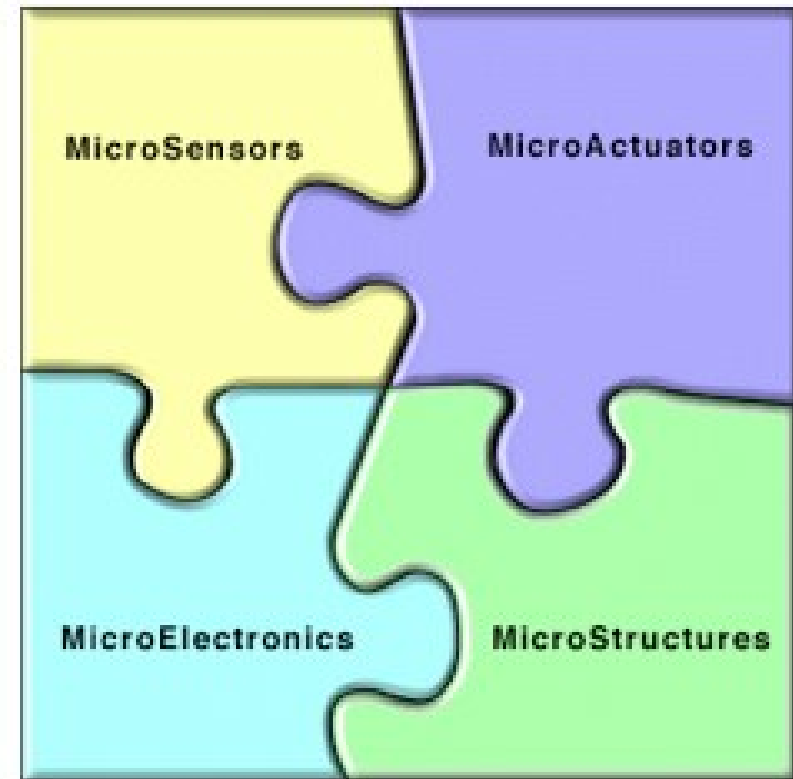
Application for harbor pilots. It tracks the cargo ships location and pulls latest buoy data from cloud for nearby buoys. Displays on map the microclimate marine conditions.



MEMS

- Micro-Electro-Mechanical Systems, or MEMS, is miniaturized mechanical and electro-mechanical elements that are made using the techniques of microfabrication.
- The critical physical dimensions of MEMS devices can vary from well below one micron on the lower end of the dimensional spectrum, all the way to several millimeters.
- The types of MEMS devices can vary from relatively simple structures having no moving elements, to extremely complex electromechanical systems with multiple moving elements under the control of integrated microelectronics.
- At least some elements having some sort of mechanical functionality whether or not these elements can move.

Components of MEMS

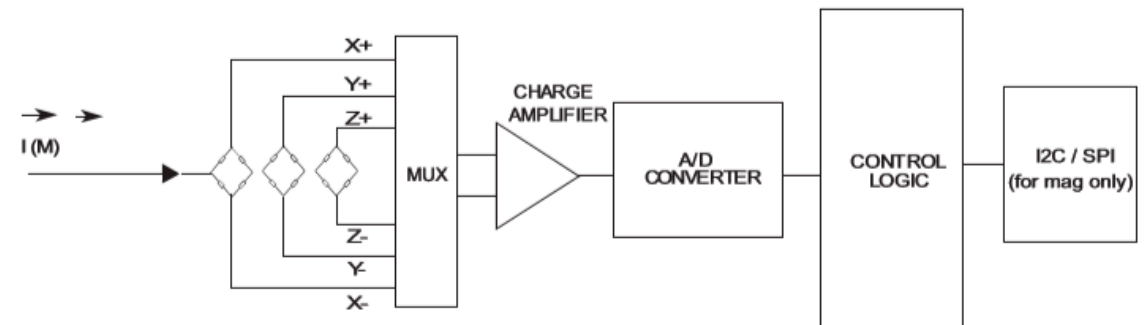


MIXED SIGNAL

- Mixed-signal integrated circuit is any integrated circuit that has both analog circuits and digital circuits on a single semiconductor die.
- Many new or improved applications are possible because firmware running on DSPs (Digital Signal Processing) or micros with analog or mixed-signal silicon analog front ends.
- Many traditionally analog systems now have a firmware component.

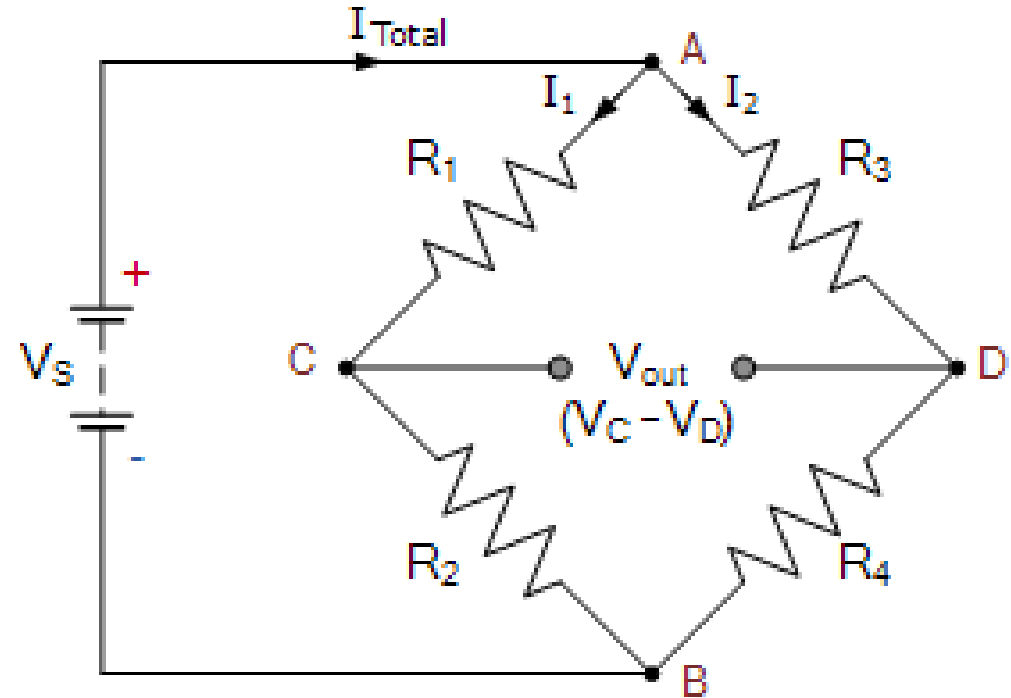
- Our sensors
 - I2C Bus is digital
 - Sensor is analog
 - ADC (Analog to Digital Converter) bridges from one to the other

Figure 9. Magnetometer block diagram



WHEATSTONE BRIDGE

- Developed by Charles Wheatstone
- Used to measure very low values of resistances down in the milli-Ohms range.
- Two series-parallel arrangements of resistances connected between a voltage supply terminal and ground producing zero voltage difference between the two parallel branches when balanced.
- Two input terminals and two output terminals consisting of four resistors configured in a diamond-like arrangement as shown.



All our sensors are some type of wheatstone bridge. With three fixed resistors and one resistor which varies linearly with the property of interest.



SENSOR

Magnetometer	Temperature	Gyroscope	Acceleration	Pressure
Magnetoresistance material	Thermal resistance material	MEMS metal fragment	MEMS metal fragment	MEMS metal bar
Magnetic moment	Heat	Angular rate	Linear moment	Barometric Pressure
microGauss	degrees centigrade	Degrees per second	Micro-g (gravities)	hectopascals

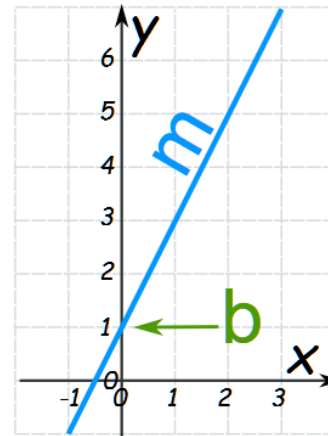


LINEAR SYSTEM



- The wheatstone bridge creates a linear response to the measured property
- Slope is the unit of measure per bit
 - E.g. 0.2 degrees centigrade per bit
- Intercept is the offset from zero
 - Environmental
 - MEMS design
 - Calibration can remove offset

Slope-Intercept Form

The most common form is the [slope-intercept equation of a straight line](#) :



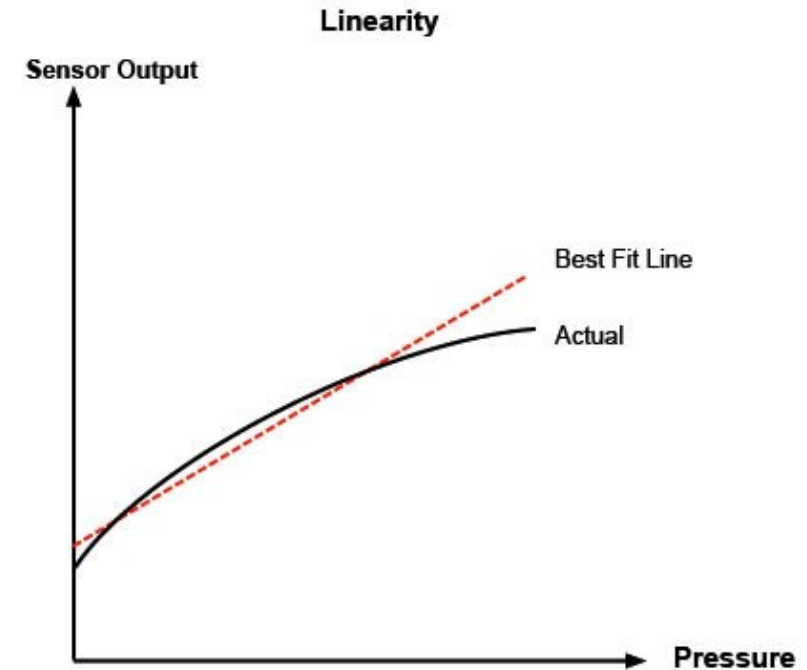
$$y = mx + b$$

 Slope (or Gradient)  Y Intercept



SOURCES OF ERROR

- Chip manufacture
 - The traces on the IC are never perfect
 - Laser trim (removes error)
 - PROM (stores correction)
 - Just specified (live with it)
- Temperature error source
 - Fixed resistors in the bridge will vary some
 - Variable resistor may drift with temperature
- Temperature error correction
 - On die temperature sensor automatically corrects
 - On die temperature provides value & your code corrects
 - Spec tells error & you must measure

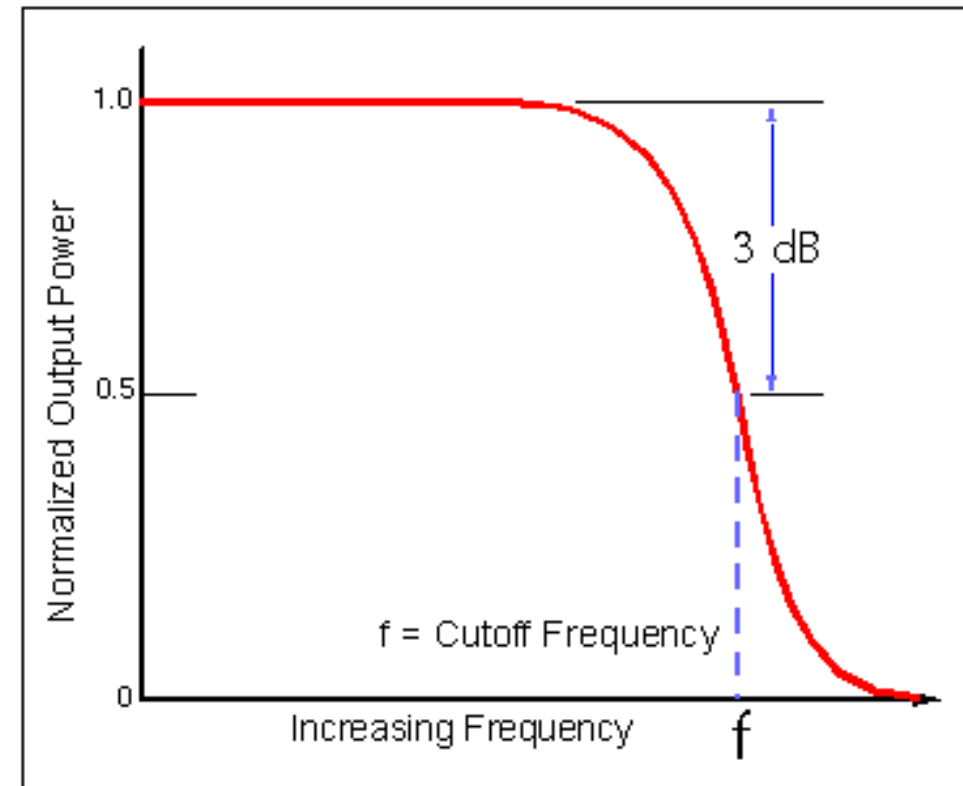


- Nonlinearity
 - Some material are not perfect straight lines over the range for the part
 - Better manufactures specify the guaranteed non-linearity. In other word worst case for it.



FILTERING

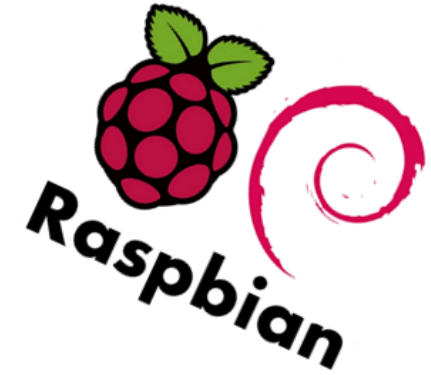
- Low Pass (LPF)
 - Remove high frequency noise and signals
- High Pass (HPF)
 - Remove low frequency noise and signals
- Band Pass (BPF)
 - Remove high and low frequencies
 - Infinite Impulse Response (IIR) filter
 - Finite Impulse Response (FIR) filters
 - narrow transition bands



low pass filter



RASPBIAN LINUX



- Unix -> Linux -> Debian -> Ubuntu -> Raspbian
 - Release 9.4 (stretch)
 - Open source, <https://www.raspbian.org/>
- Raspberry Pi organization has three operating systems versions: NOOBS, NOOBS Lite, Raspbian, Raspbian with Desktop. <https://www.raspberrypi.org/downloads/>
 - NOOBS, New Out Of the Box Software
 - Easy operating system installer contains Raspbian and LibreELEC.
 - Alternative operating systems links: Ubuntu Mate, Snappy Ubuntu Core, Libreelec, Pinet, OsMc, Risc Os
 - which are then downloaded from the internet and installed.
 - Windows 10 IoT Core is shown as an option on NOOBS, but will not install!
- How to install: <https://www.raspberrypi.org/documentation/installation/installing-images/windows.md>
 - SD Card Formatter: https://www.sdcard.org/downloads/formatter_4/
 - Use Win32DiskImager to make a bootable SD card <https://sourceforge.net/projects/win32diskimager/>
 - Default user login: pi, password: raspberry
- Beware Raspbian from Raspberrypi.org now includes cripple-ware



DEBIAN TOOLS INSTALL

- `sudo apt-get install packageName`
 - `X-man` is a manual
 - `X-dev` are developer tools
 - `libxxx` are static link libraries
- `sudo apt update & upgrade`
- Raspbian requires ARM EABI armhf from the raspberrypi.org repositories. Do not load from Debian

Package Name	Tool
gpio, libgpio-dev	GPIO
python-all	Python
i2c-tools, libi2c-dev	I2C tools & libs
eclipse, eclipse-cdt	IDE, C++
python-smbus	I2C Python
node, nodejs	Node



BOOTING & SHUTDOWN

Windows 10 Core

- Green light blinks a couple times during boot and forever after shutdown
- Boots in more than a minute

Raspbian

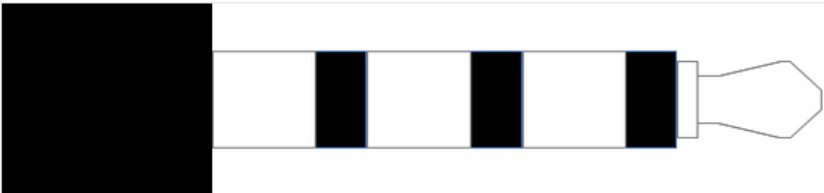
- Green light blinks when accessing SD card
- Boots in ten seconds
- Quick power off – Alt-SysRq-I (system request kill all processes) then Alt-SysRq-O (system request power-Off)

Boots on power connection. There is no power switch or reset button. Do not plug a transform into The 1/8" coaxial connector (that is audio & video out). Plug USB power adapter into micro-USB (not the standard USB-A connectors)

COMPOSITE VIDEO & AUDIO

- Do not plug a transform into the 1/8" coaxial connector (that is audio & video out)
- 3.5mm jack is **FOUR** pole audio and composite video (not a 3 pole for ear buds). Ring 2 must be ground!
- Raspberry Pi Display Resolution if monitor is disconnected
- The Raspberry Pi may not maintain Display Resolution if monitor is disconnected. The EDID of the monitor is used to set the resolution of the system when one is connected. When disconnected, the Raspberry Pi firmware defaults to what is in the config.txt in the root of the SD card.
- `tvservice -s`
 - <https://www.raspberrypi-spy.co.uk/2014/07/raspberrypi-model-b-3-5mm-audiovideo-jack/>
 - <https://bhavyanshu.me/tutorials/force-raspberrypi-output-to-composite-video-instead-of-hdmi/03/03/2014/>

Raspberry Pi Model B+ 3.5mm Audio/Video Socket

Device	Sleeve	Ring 2	Ring 1	Tip	OK?
					
	4	3	2	1	
Model B+	Video	Ground	Right	Left	✓
Apple	Video	Ground	Right	Left	✓
Zune	Video	Ground	Right	Left	✓
Camcorders	Right	Ground	Video	Left	⚠
MP3 Players	Ground	Video	Right	Left	✗

www.raspberrypi-spy.co.uk



WINDOWS 10 CORE

Additional Insider Preview downloads

RaspberryPi 3B+ Technical Preview Build 17661



- Windowsondevices.com (redirects to <https://developer.microsoft.com/en-us/windows/iot>)
- Long Term Servicing Branch, 10 year support but you must register your application every six months.
 - New release every six months
- Windows IoT Core is a version of Windows 10 that is optimized for smaller devices with or without a display that run on both ARM and x86/x64 devices. The Windows IoT Core documentation provides information on connecting, managing, updating, and securing your devices, and more.
- The release for the Raspberry Pi 3B+ (the downloadable ISO can be found [here](#)) is a technical preview and there is currently no timeline for a release version. For a better evaluation experience and for any commercial products, please use the Raspberry Pi 3B or other devices with supported Intel, Qualcomm or NXP SoCs. From <<https://docs.microsoft.com/en-us/windows/iot-core/release-notes/currentcommercial>>
- OS Install Location (<https://www.microsoft.com/en-us/software-download/windowsiot>)
- Cost \$0.00



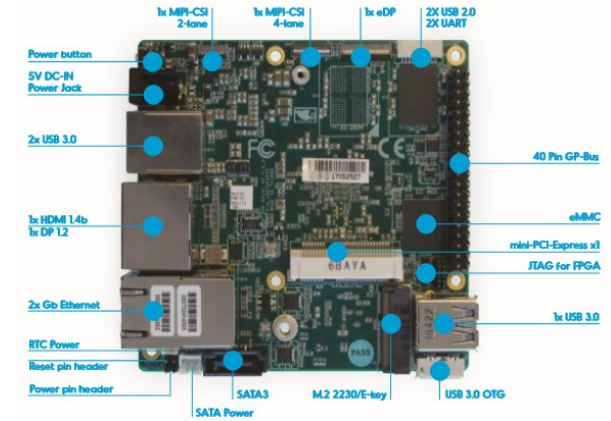
RASPBIAN HARDWARE



Name	Processor	Cost	Comments
<u>APC</u>	VIA ARM Cortex-A9	\$49 to \$59	APC is a fundamental redesign of the PC based on the Neo-ITX form factor
<u>Raspberry Pi 2 & 3</u>	Broadcom ARM	\$35	All versions are supported including the 3 B+ (the only version currently for sale). The Broadcom Wi-Fi chip is supported.
<u>Raspberry Pi Zero</u>	Broadcom ARM	\$5 - \$10	<u>Supported both Wi-Fi and Bluetooth versions</u>



W10 IOT CORE HARDWARE



Name	Processor	Cost	Comments
<u>UP²</u>	Intel Pentium, Atom, Celeron & Apollo	\$150 to \$400	
<u>DragonBoard 410c</u>	Qualcomm Snapdragon 400	\$75	Product of Arrow Electronics
<u>MinnowBoard Turbo</u>	Intel Atom	\$150 to \$200	Open source hardware
<u>Raspberry Pi 2 & 3</u>	Broadcom ARM	\$35	3 B+ is only preview without Wi-Fi chip support. Windows 10 prefers a TPM chip & GPU. Pi has neither. Pi 2 is discontinued.
Raspberry Pi Zero	Broadcom ARM	\$5 – \$10	<u>Not supported by Windows 10</u>



WINDOWS DEVICE PORTAL

IOTCoreDefaultApplication	Foreground	<input checked="" type="radio"/>	Running	Actions ▾
IoTUAPOOBE	Foreground	<input type="radio"/>	Stopped	Actions ▾
Barometer	Background	<input type="checkbox"/>	Stopped	Actions ▾
IoTOnboardingTask	Background	<input type="checkbox"/>	Stopped	Actions ▾

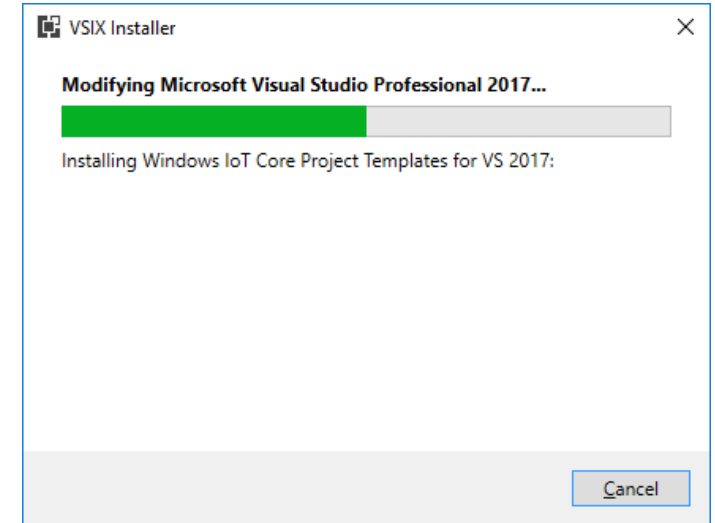
- Remote administration and development for Windows 10 IoT Core
- Ip4 address port 8080
- List installed applications
- Start, Stop or set to auto-start application



WINDOWS TOOLS INSTALL

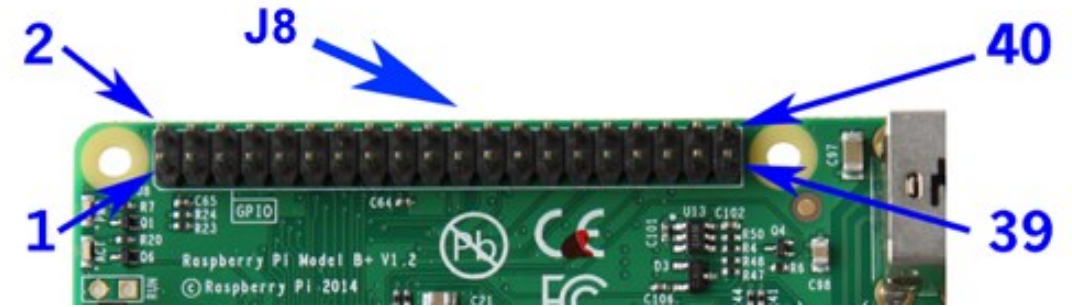
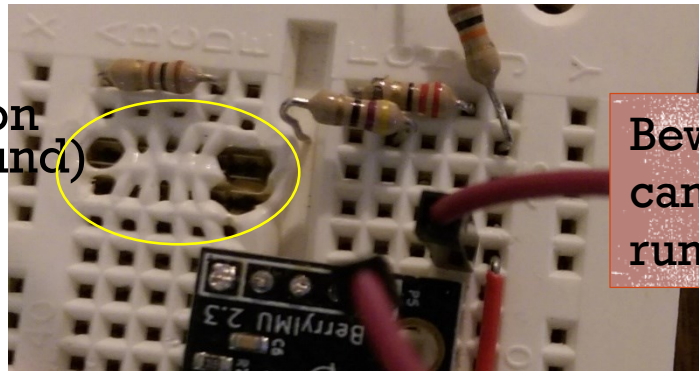
When the credentials dialog opens, make sure you use yourrpi3machinename\Administrator for the user name. The default password is p@ssw0rd. You need to connect with the computer name in front of the username. The zero can easily be mistaken as an 'O' so calling that part out.

- VS2017 Community Edition
- Install Windows IoT Core Project Templates for VS 2017 (VS Extensions & Updates)
- Install latest Windows development SDK (MSDN downloads)
- Windows 10 IoT Dashboard
- Join the Windows Insider Program (<https://developer.microsoft.com/en-us/windows/iot/Downloads>)



GENERAL PURPOSE I/O

- All services based on Broadcom SoC capabilities
- J8 40 pin connector 0.1" centers on Raspberry Pi
(<http://pi4j.com/pins/model-3b-plus-rev1.html>)
- 3.3 & 5.0 VDC and ground
- **I²C (Inter-Integrated Circuit) Bus**
 - Single master
 - 126 slave devices
 - kHz speeds
- Other: UART (RS232), Pulse-Code Modulation (a.k.a. I²S, Inter-IC Sound)



- SPI (Serial Peripheral Interface) Bus
 - Multiple masters
 - Multiple slaves (independent or cooperative slaves)
 - Limited by TTL signal levels
 - DMA (Direct Memory Access)
 - MHz speeds
- **GPIO (27 pins)**
 - Write (set) high or low
 - Read signal level (high or low)
 - Interrupt on transition (either way)

Beware, hardware is not transactional. You cannot roll it back. Do not leave new code running unattended!

OZZMAKER BERRYIMU

- Quick start, <http://ozzmaker.com/berryimu-quick-start-guide/>
- Inertial measurement unit (IMU), ST μ LSM9DS1, [3D digital linear acceleration sensor](#), a [3D digital angular rate sensor](#), and a [3D digital magnetic sensor](#)
- Bosch BMP280 digital pressure sensor, <https://ae-bst.resource.bosch.com/media/tech/media/datasheets/BST-BMP280-DS001-19.pdf>



- I2C bus
 - 400 KHz, fast mode operation
- 6-pin connector for top of PI J8
- Side four pin jumper to PI J8
- All five interrupt pins available on bottom



DEMO #1

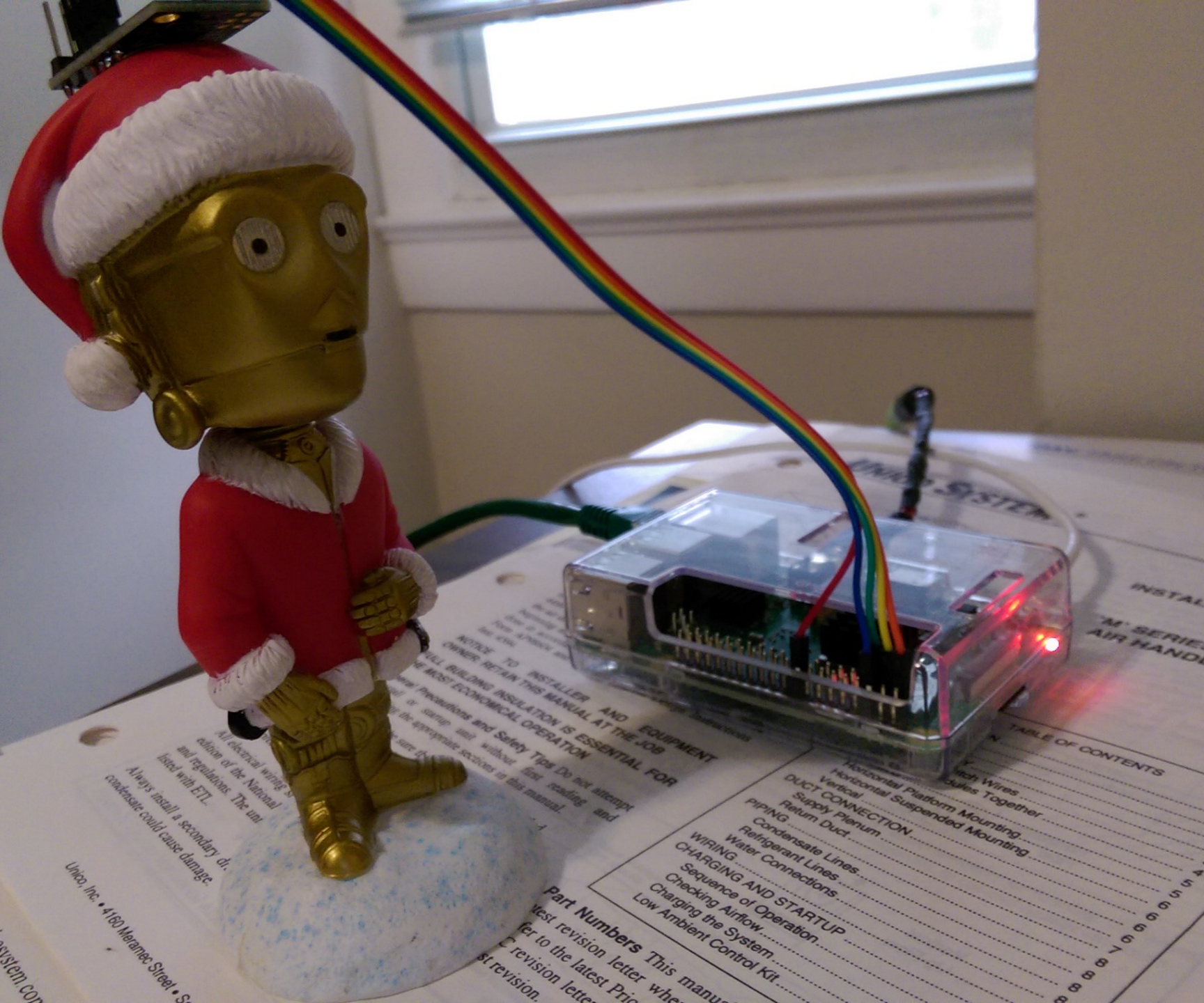
Barometric pressure and tempature



DEMO #2

3D Compass

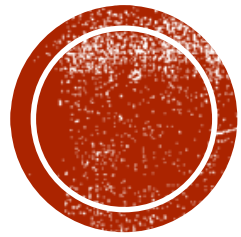




DEMO #3

Bobble head Santa plays music
Tempo is based on acceleration
Lab #2





LAB 1



Raspbian & Eclipse

RASPBIAN I2C

- Raspbian does not install with I2C enabled
- Follow these steps to enable:
<http://ozzmaker.com/i2c/>
- Use `i2cdetect` to read all the slave addresses
 - 0x6A for the gyroscope and accelerometer
 - 0x1C for the magnetometer
 - 0x77 for the pressure sensor

```
pi@raspberrypi2:~ $ sudo i2cdetect -y 1
```

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
00:				--	--	--	--	--	--	--	--	--	--	--	--	--
10:	--	--	--	--	--	--	--	--	--	--	--	--	1c	--	--	--
20:	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
30:	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
40:	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
50:	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
60:	--	--	--	--	--	--	--	--	--	--	6a	--	--	--	--	--
70:	--	--	--	--	--	--	--	77								



BAROMETRIC SENSOR



- Bosch BMP280 digital pressure sensor, <https://ae-bst.resource.bosch.com/media/tech/media/datasheets/BST-BMP280-DS001-19.pdf>
- BerryIMU breakout board, <http://ozzmaker.com/berryimu-quick-start-guide/>
- I²C slave device on address 0x77
- ID register 0xD0 contains the chip identification number chip_id[7:0], 0x58
- Pressure range 300 to 1100 hPa (equiv. to +9000...-500 m above/below sea level)
 - One hectopascal = 0.02953 inches of mercury
- Package 8-pin LGA metal-lid
Footprint : 2.0 × 2.5 mm, height: 0.95 mm
- Temperature coefficient offset 1.5 Pa/K, equiv. to 12.6 cm/K (25 ... 40°C @900hPa)
- Digital interfaces I²C (up to 3.4 MHz)
SPI (3 and 4 wire, up to 10 MHz)



PYTHON INITIALIZE I2C BUS

- Create a file .py
- Open the I2C bus
 - System Management Bus (smbus) is derivative of I²C for PC power
 - <https://pypi.org/project/smbus2/>
- Read from bus for the slave address 0x77
 - The product id register, 0xD0
 - It should be 0x58 (88 decimal)



```
import smbus
# Get I2C bus
bus = smbus.SMBus(1)
# BMP280 address, 0x77
# Read data back from 0xD0,
# One byte
b1 = bus.read_i2c_block_data(0x77, 0xD0, 1)
print(b1)
```



READ PRESSURE

- Add to .py file
- Set Configuration and Control Measurement registers
- Read 19-bits of pressure data
- Convert to inches or mercury
 - **Why is the value off by a little?**
 - Trim and Temperature



```
#Select Configuration register, 0xF5(245)
# 0 - I2C, # 2,3,4 - IIR filter, # 5,6,7 - tstandby, Stand_by time = 0.5 ms
bus.write_byte_data(0x77, 0xF5, 0x08)

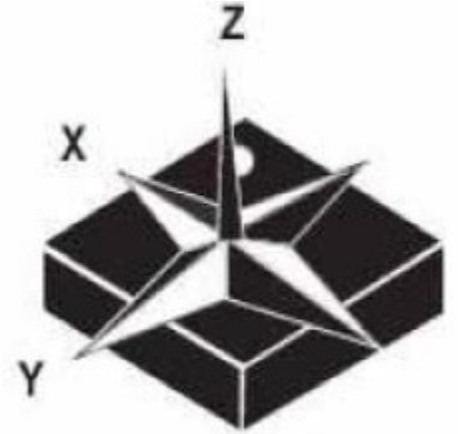
#Select Control measurement register, 0xF4(244)
# 0,1 - Power Mode, Forced mode, # 2,3,4 - Oversample pressure x8
# 5,6,7 - Oversample temp x1
bus.write_byte_data(0x77, 0xF4, 0x32)
time.sleep(0.5)

# Read data back from 0xF7(247), 8 bytes
# Pressure MSB, Pressure LSB, Pressure xLSB
data = bus.read_i2c_block_data(0x77, 0xF7, 3)
print(data);
```



MAGNETOMETER

- Same ST μ LSM9DS1, 3D digital linear acceleration sensor, a 3D digital angular rate sensor, and a 3D digital magnetic sensor
- Same BerryIMU breakout board, <http://ozzmaker.com/berryimu/>
- I²C slave device on address 0x1C
- WHO_AM_I_M register 0x0F contains 0x3D



- I2C bus
 - 400 KHz, fast mode operation
- Magnetic field full scale of $\pm 4/\pm 8/\pm 12/\pm 16$ gauss
- Magnetic sensitivity, FS = ± 4 gauss
0.14 mgauss/ LSB



C++ INITIALIZE I2C BUS

- Create a file .c
 - Include i2c-dev.h header file
- Open the I2C bus
 - As a file from devices
 - Use bus number from i2cdetect
 - Select slave device with I/O control
- Read from bus for the slave address 0x1C
 - The who am I register, 0x0F
 - It should be 0x3D
- Compile
 - gcc -o magnetometer magnetometer.c

```
#include "linux/i2c-dev.h"

const int MAG_ADDRESS = 0x1C;
const int ProductId = 0x0F;
const int WhoAmIResponse = 0x3D;

//the I2C bus file handle for I/O
int file;

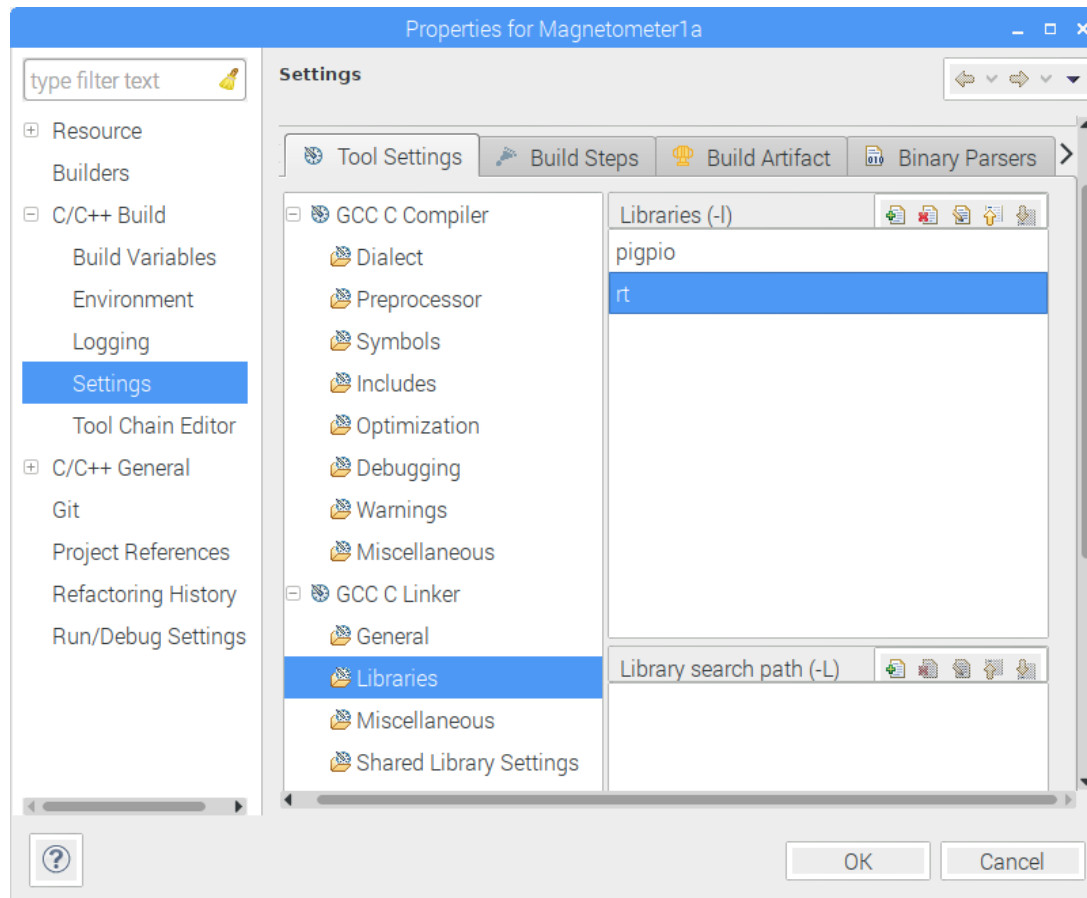
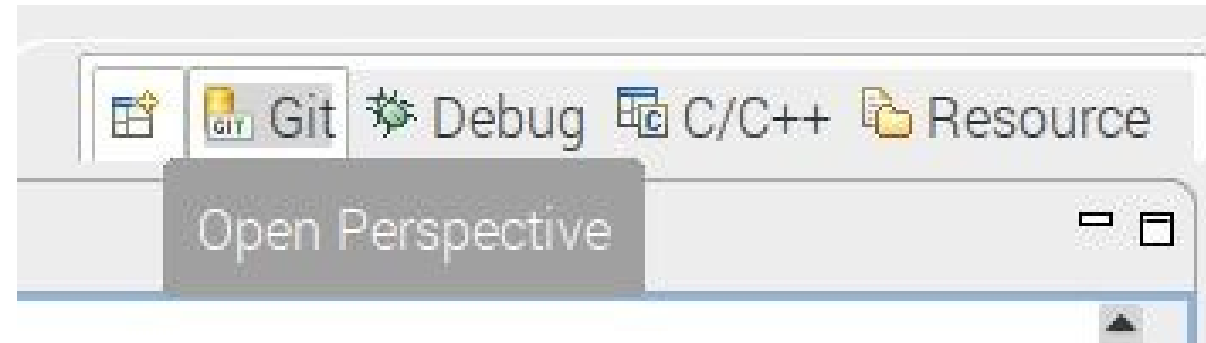
//open the bus device driver
char filename[20];
sprintf(filename, "/dev/i2c-%d", 1);
file = open(filename, O_RDWR);

// I/O control to set the slave address on the bus file
// in effect until file is closed or selection changed.
ioctl(file, I2C_SLAVE, MAG_ADDRESS);

//Detect if LSM9DS1 is connected
int response = i2c_smbus_read_byte_data(file, ProductId);
if (response == WhoAmIResponse) printf("LSM9DS1 detected");
```



ECLIPSE

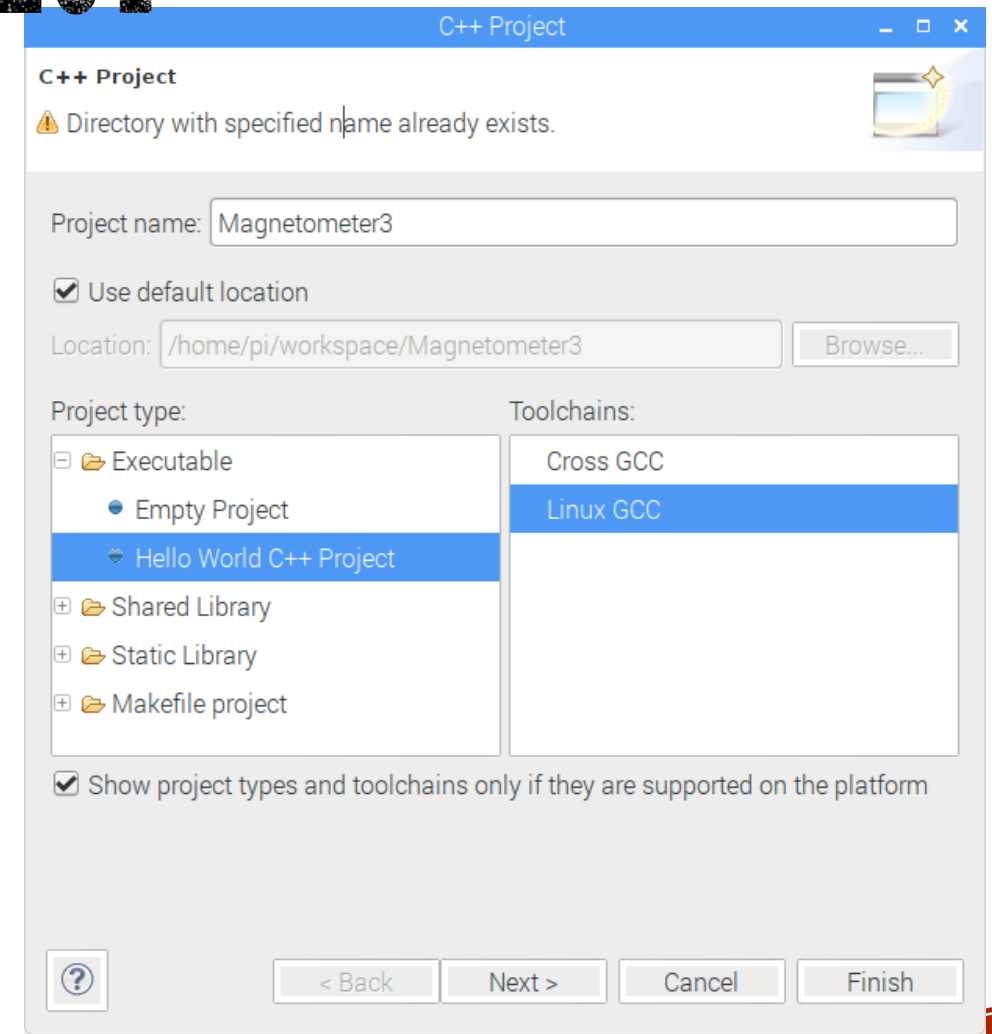


- Open Perspectives
 - Debug and C/C++



CREATE NEW C++ PROJECT

- File -> New -> C++ Project
- Name it Magnetometer1c
- Choose “Hello World” project type and Linux GCC toolchain
- Finish
- Delete the file in the src folder
- Download all the files from https://1drv.ms/f/s!Ar3pO7_GhJY9hOttE8IIFGsVOuS5VA
- Place the files in the src folder and refresh the project



Starting LSM9DS1

Detected Magnetometer.

Magnetometer Initialized...

x: -31749 y: -1007 z: -1028

x: -28165 y: -1007 z: -1028

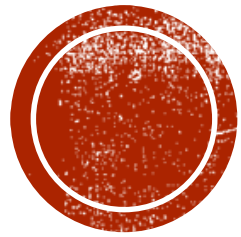
x: -30725 y: -1008 z: -1028

CHALLENGE #1

Slow down the output to one reading per second.

Move the magnet and determine the orientation of x, y and z axis





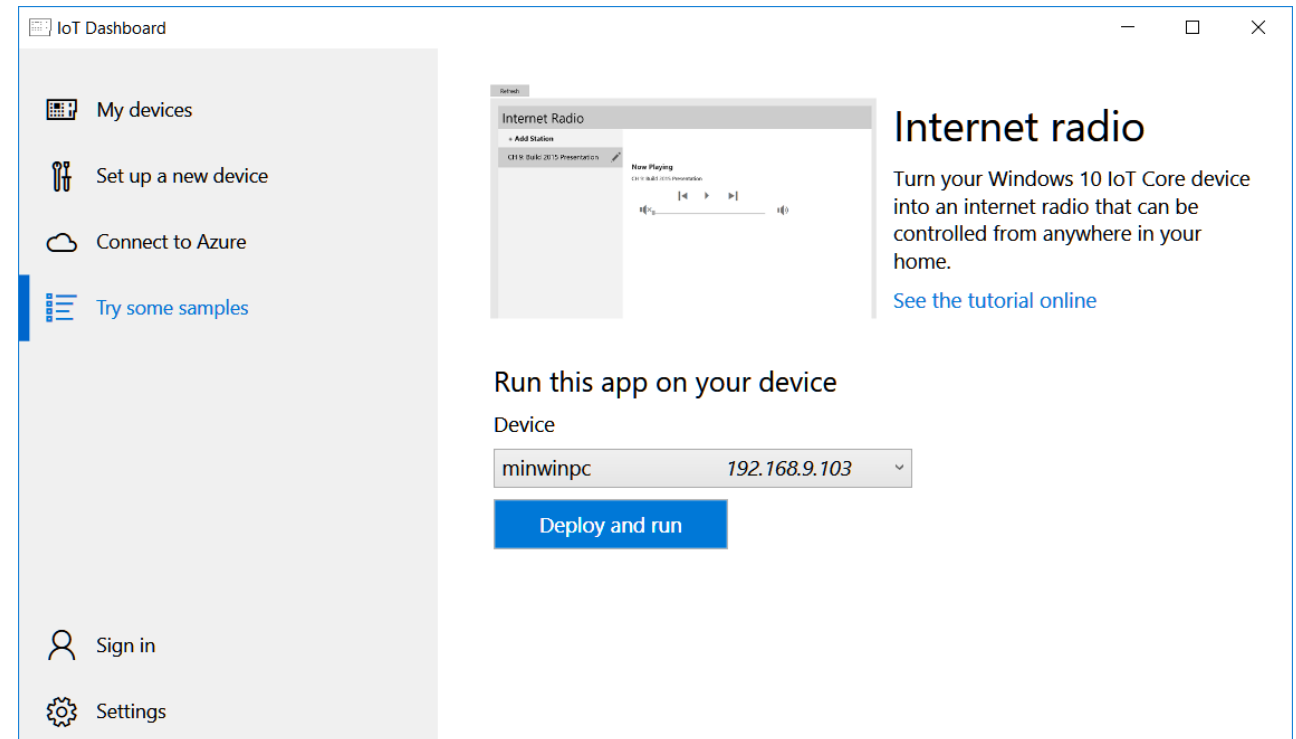
LAB 2



Windows 10 IoT Core & Visual Studio

APPLICATION DEPLOYMENT

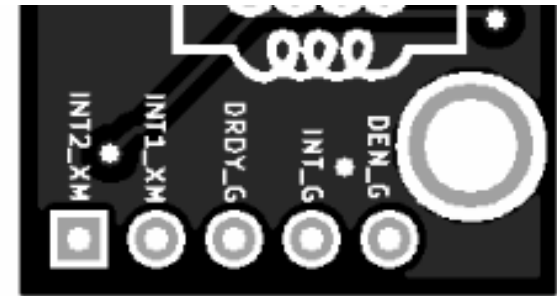
- Open IoT dashboard
- Find your device on the subnet
- Deploy and run the internet radio sample application
- Ensure you can hear channel 9



INTERRUPT CONNECTIONS

Broadcom	J8 Connector	Ozzmaker
GPIO 23	16	DRDY_M
GPIO 24	18	INT1_A_G

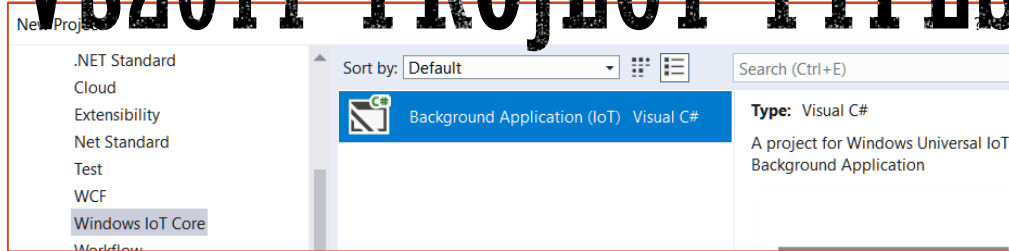
DRDY_M is only required for the magnetometer3 application. We will not try this application in class.



Your board is a different revision & layout

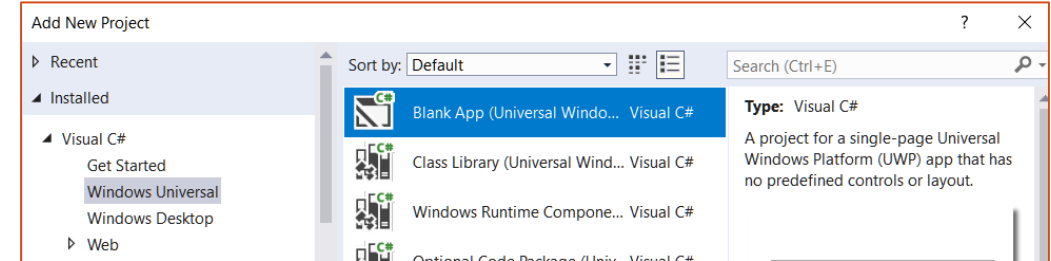


VS2017 PROJECT TYPES



Background Application

- UWP but no UI
- `class StartupTask : IBackgroundTask`
- Package Deployment
- Declaration: Background Tasks
- Assembly executed by `svchost.exe`



Foreground Application

- UWP must be foreground to run
- `partial class App : Application`
- Package Deployment
- Declaration: *none*
- XAML

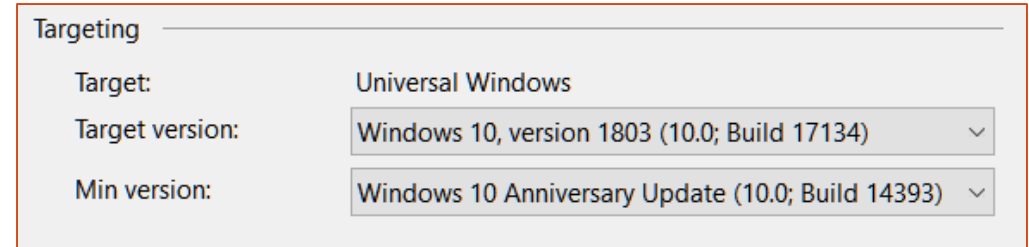
Choose Package->Capability->"Low Level" to use .NET GPIO libraries. Choose "Low Level Device" if you intend to write your own device driver instead. It allows IOCTLs. You may need to hand edit the declarations into the package XML.



VS PROJECT SETTINGS

- Must target at least Anniversary edition to work and v1803 for full functionality
- Compile with ARM instructions
- Advanced Build Settings
 - C# 7.3 compiler has some new key words and improved code generation
 - Arithmetic checks because you will like have some math

This is cross platform development

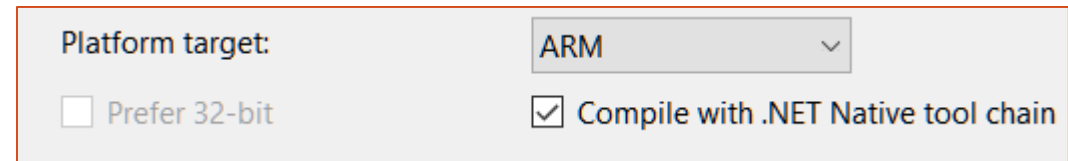


Targeting

Target: Universal Windows

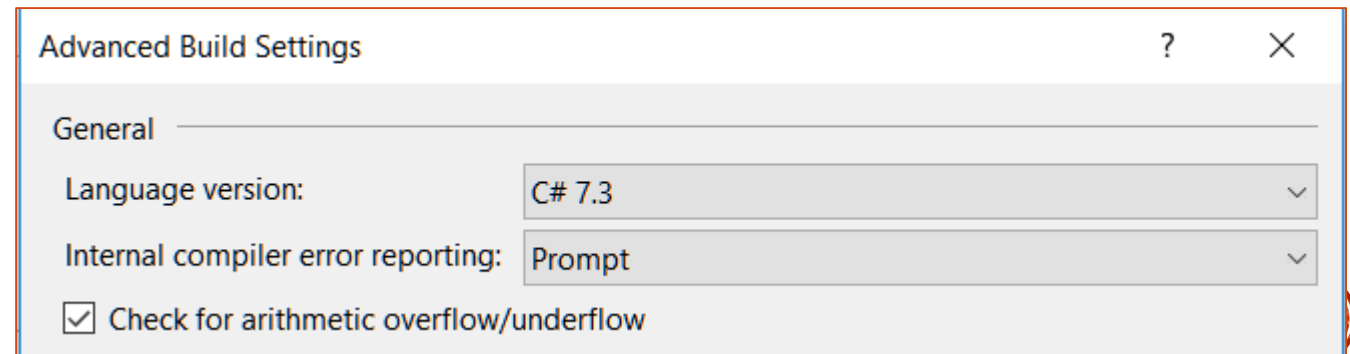
Target version: Windows 10, version 1803 (10.0; Build 17134) ▼

Min version: Windows 10 Anniversary Update (10.0; Build 14393) ▼



Platform target: ARM ▼

☐ Prefer 32-bit ☒ Compile with .NET Native tool chain



Advanced Build Settings ? X

General

Language version: C# 7.3 ▼

Internal compiler error reporting: Prompt ▼

☒ Check for arithmetic overflow/underflow

VS DEBUGGING

- Use project Deploy options to side-load code
 - Read the package file and deploys any missing sub-packages
- Use Remote Machine debugging (not Device debugging)
- Find button will search local subnet for Raspberry Pi's.
- Debugging & Deployment requires: wired VS & Pi (no Wi-Fi)

Start options

Target device: Remote Machine

Remote machine: Wave2 Find...

Authentication Mode: Universal (Unencrypted Protocol)

Remote Connections

Searching...

Manual Configuration

Address: Wave2

Authentication Mode: Universal (Unencrypted Protocol)

Select

Auto Detected

minwinpc -- 192.168.9.156



INITIALIZE I2C BUS

- Create a new Background Application IoT in VS2017 (<https://docs.microsoft.com/en-us/windows/iot-core/develop-your-app/backgroundapplications>)
- Add project reference to *Windows IoT Extensions for the UWP*
- Find the I2C bus
- Open the I2cDevice for the slave address
 - 400 kHz bus speed

```
// Get a selector string that will return all I2C controllers on the
system
string allIc2Controllers = I2cDevice.GetDeviceSelector();

// Find the I2C bus controller device with our selector string
var discoveredI2cDevices =
    await DeviceInformation.FindAllAsync(allIc2Controllers);
var discoveredI2cDevice = discoveredI2cDevices[0].Id;

// Create I2cDevices with our selected bus controller and I2C settings
var i2cConnectionSettings = new I2cConnectionSettings(slaveAddress)
{
    BusSpeed = I2cBusSpeed.FastMode, // Enable 400kHz I2C bus speed
};

// Create I2cDevices with our selected bus controller and I2C settings
var i2cDevice = await I2cDevice.FromIdAsync(
    discoveredI2cDevice, i2cConnectionSettings);
```



UWP & EVENTS

- Create Blank UWP Application
- Add reference to *Windows IoT Extensions for the UWP*
- Select package capability *Low Level*
- Program a GPIO pin for input
 - Fire an event when signal goes high
 - Event thread taken from thread pool
 - Better performance than polling
 - May need to write device driver if time critical



```
//fire an event when the data ready pin goes high
```

```
drdyMPin = EnableInterruptMonitor(DRDY_M,  
    InterruptDrdyM,  
    GpioPinDriveMode.InputPullDown);
```

```
while (ReadData().HasValue) ;
```

```
/// <summary>
```

```
/// Receive the interrupt event that
```

```
/// data is ready
```

```
/// </summary>
```

```
/// <param name="sender">the pin</param>
```

```
/// <param name="args">the event</param>
```

```
protected void InterruptDrdyM(GpioPin sender,  
    GpioPinValueChangedEventArgs args) {
```

```
    if (drdyMPin.Read() == GpioPinValue.High)  
    {
```

```
        var newReading = ReadData();
```

```
        NewReading?.Invoke(this, newReading);
```

```
    } }
```

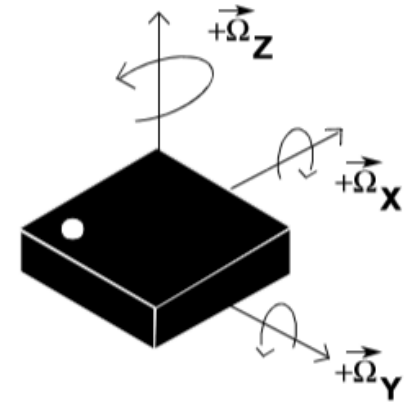


IOT APPLICATION #2: BOBBLE HEAD SANTA

- Mount Raspberry Pi on a bobble head Santa.
 - Place an inertial measurement unit in Santa's head.
 - Connect a speaker to the Pi
- Use gyroscope & accelerometer to determine when the head is bobbing.
- Play a Jingle Bells with the tempo based on the angular velocity and linear acceleration.
 - Replace with MP3 of your choice

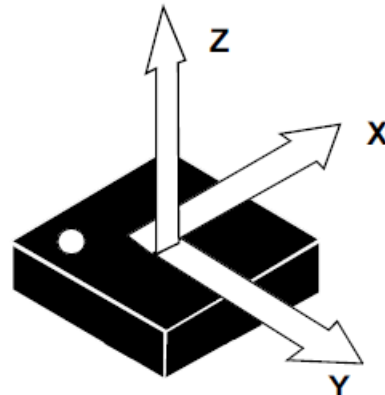


GYROSCOPE & ACCELEROMETER



- ST μ LSM9DS1, 3D digital linear acceleration sensor, a 3D digital angular rate sensor, and a 3D digital magnetic sensor
- Same BerryIMU breakout board, <http://ozzmaker.com/berryimu/>
- I²C slave device on address 0x6A
- WHO_AM_I register 0x0F contains 0x68

- I2C bus
 - 400 KHz, fast mode operation
- Linear acceleration full scale of $\pm 2g/\pm 4g/\pm 8/\pm 16 g$
- Angular rate of $\pm 245/\pm 500/\pm 2000$ degrees per second.



STATUS REGISTER

- Data ready pin transitioning to high causes event to fire
- Read status register
- Determine what data is read
- Read it
- Fire event for gyroscope data
- Fire different event for accelerometer data

```
var status = (StatusRegister)ReadByteFrom(STATUS_REG);

if (status.HasFlag(StatusRegister.GDA))
{
    var gyroReading = GetGyroscopeReadings();
    NewGyroscopeReading?.Invoke(this, gyroReading);
}

if (status.HasFlag(StatusRegister.XDA))
{
    var accelerometerReading =
        GetAccelerometerReadings();
    NewAccelerometerReading?.Invoke(this,
        accelerometerReading);
}

if (status.HasFlag(StatusRegister.TDA))
    DiscardTemperatureData();
```



MEDIA PLAYER

- Play an mp3 file distributed with the package
- Set as the source to the player
- Alter the playback session's rate based on head motion



```
var mediaPlayer =  
    new MediaPlayer();  
var audioPath = new Uri(  
    @"ms-appx:///JingleBells.mp3");  
var source = MediaSource.  
    CreateFromUri(audioPath);  
mediaPlayer.Source = source;  
mediaPlayer.Play();  
mediaPlayer.PlaybackSession.  
    PlaybackRate = value;
```



BONUS SLIDES



If time allows

GCC LINUX PI

- Compiler the settings required for most GNU tools are as follows:
 - `march=armv6`
 - `mfpv=vfp`
 - `mfloat-abi=hard`
- Produce code for armv6 specific instructions
 - vector floating point instructions



RASPBIAN GPIO & EVENTS



- Linux GPIO statement
 - GPIO interfaces in the kernel
- Universal File System
 - /sys/class/gpio
 - echo 23 > /sys/class/gpio/export
 - ls -lh /sys/class/gpio/gpio23
 - hardware pin can turn on interrupts by writing your desired setting into the edge file
 - Another example
- wiringPi
 - #include <wiringPi.h>
 - Arduino clone
 - Functions
 - Requires Superuser
 - Assumes ownership of all Pi hardware: SPI, IC2, GPIO, UART, etc...
- Pigpio Library
 - Requires Superuser
 - Polling, not interrupt based, uses timers
 - Several versions including a daemon



RASPBIAN REMOTE ACCESS

- SSL is turned off as installed
 - `sudo service ssh start`
 - `sudo systemctl enable ssh`
 - Then SSH will work (no GUI, of course)
- A pre-installed vnc-server
 - Configuration utility enables
 - Requires a purchased commercial license to use

