Week 4 – Sensors I

Advanced Mechatronics System Design – MANU2451

Dr Chow Yin LAI

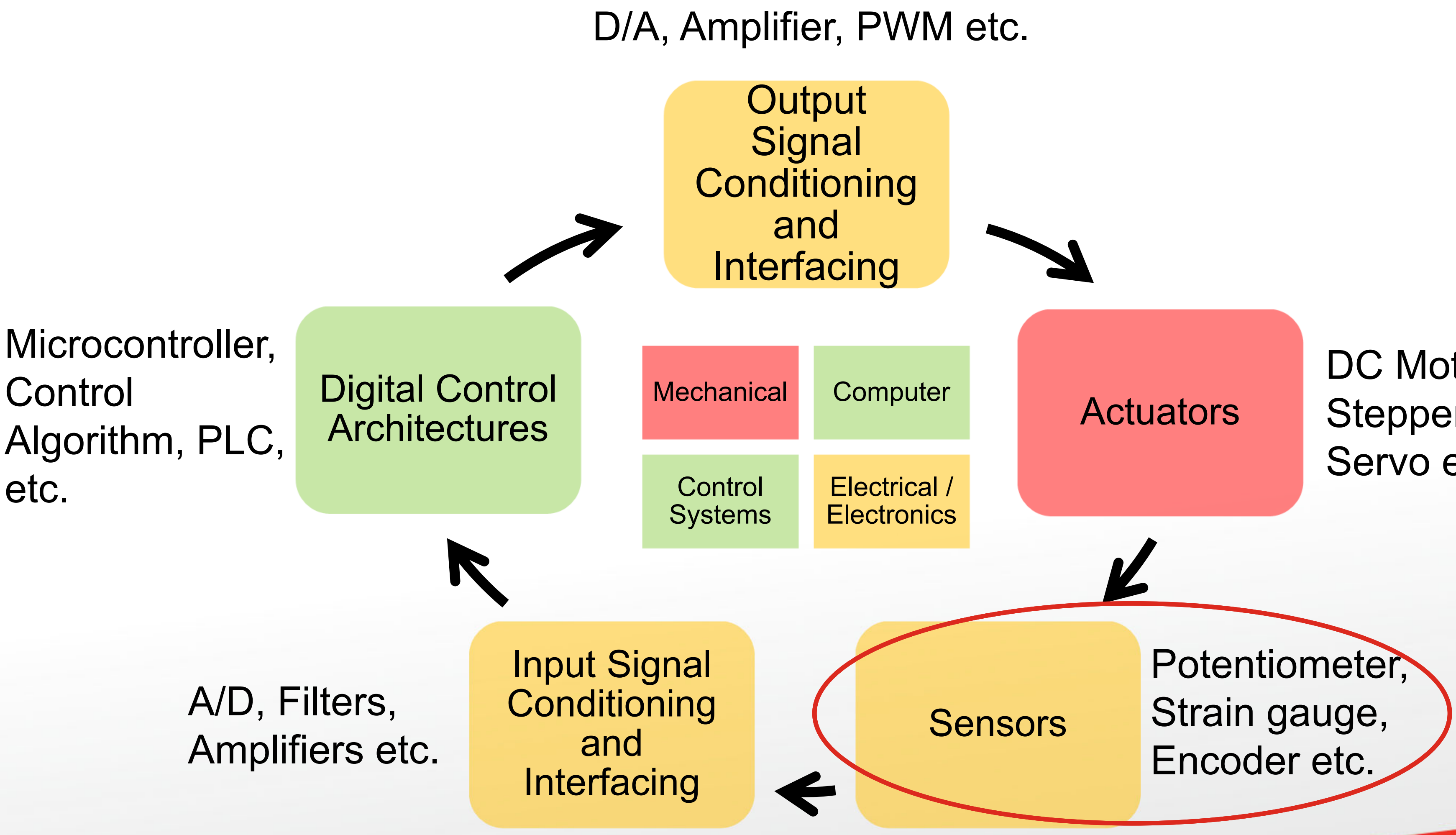
Edited by Dr Milan Simic
School of Engineering
RMIT University, Victoria, Australia
Email: milan.simic@rmit.edu.au

RMIT UNIVERSITY

New Teaching Schedule

Week		Class Activity Before	Lecture	Class Activity During or After
1			Introduction to the Course / Introduction to LabVIEW	LabVIEW Programming
2			Introduction to LabVIEW / Data Acquisition	LabVIEW Programming
3			Gripper / Introduction to Solidworks / Safety	Gripper Design
4			Sensors I	myRIO Programming for Sensor Signal Reading / Gripper Design
5			Sensors II	myRIO Programming for Sensor Signal Reading
6			Actuators I	LabVIEW Tutorial
7		LabVIEW Assessment.	DC Motors I	Matlab Simulink Simulation
8		Design report submission	DC Motors II	Matlab Simulink Simulation / myRIO Programming for Control
9			Actuators II	Matlab Simulink Simulation Gripper CAD
10			Modeling and System Identification	Matlab Simulink Simulation / Gripper simulation testing
11			Artificial Intelligence I	Matlab Simulation / Finalize Gripper
12		Gripper Simulation / Submission of Report	Artificial Intelligent II	Revision

Mechatronics System Components



Mechatronics System Components

Sensors:
Encoder at
each joint

Input signal interfacing

Robot controller:

- Generate desired motion trajectory
- Calculate current end-effector position based on angular position (kinematics)
- Calculate desired angular position for desired end-effector position and trajectory (inverse kinematics)
- Control algorithm
- Safety, collision detection etc.

Industrial Robots
https://commons.wikimedia.org/wiki/File:Float_Glass_Unloading.jpg

Output signal interfacing

Actuators:
Geared motor
at each joint

Content

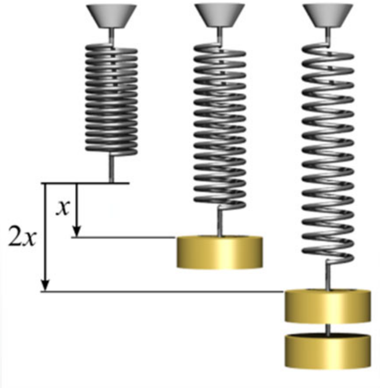
- Introduction
- Touch Sensors and Switches
    - Measure using myRIO
- Proximity Sensing
    - Measure using myRIO
- Position Measurement
    - Measure using myRIO
- Attachment: Writing FPGA Codes

Content

- Introduction
- Touch Sensors and Switches
    - Measure using myRIO
- Proximity Sensing
    - Measure using myRIO
- Position Measurement
    - Measure using myRIO
- Attachment: Writing FPGA Codes

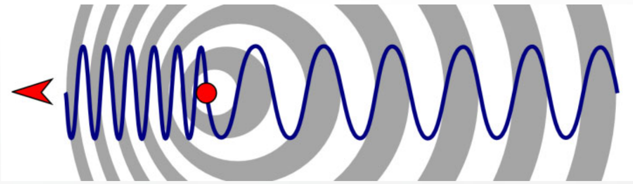
Introduction

- Sensor detects the magnitude of physical parameter, and transforms it into a signal (usually voltage) that can be processed by the (mechatronics) system.
- Transducer is the active element of the sensor.
- Sensors and transducers make use of physics or chemistry principal to measure a certain quantity. E.g.
  - Hooke's law, for stress measurement using strain gauge
  - Doppler effect, for speed measurement



Hooke's Law
https://en.wikipedia.org/wiki/Hooke%27s_law#/media/File:Hookes-law-springs.png

Doppler Effect Radar Gun
https://commons.wikimedia.org/wiki/File:Radarvelocidade20022007.jpg

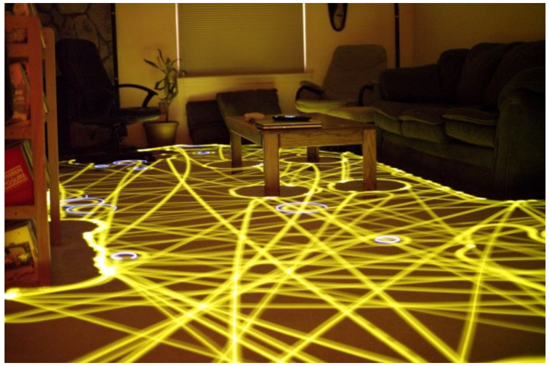


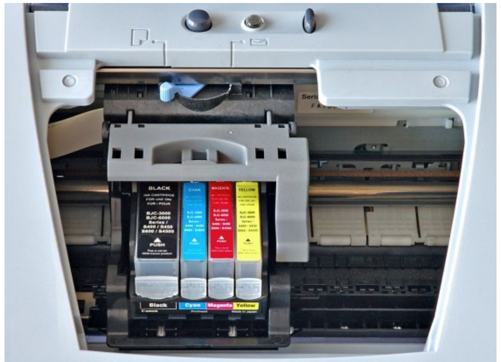
Doppler Effect
https://commons.wikimedia.org/wiki/File:Doppler_effect_diagrammatic.png

Introduction

- Sensors are a crucial part of mechatronics devices.
  - For logic / decision making. E.g.
    - Has the robotic vacuum cleaner hit a wall? → change path

Path of a Robotic Cleaner
https://upload.wikimedia.org/wikipedia/commons/7/77/Roomba_time-lapse.jpg





Inkjet Printer
https://commons.wikimedia.org/wiki/File:Canon_S520_ink_jet_printer_-_opened.jpg

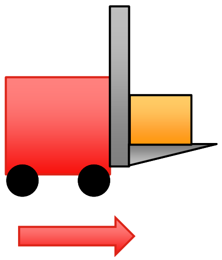
    - Has the gripper firmly gripped the object? → Next robot command
  - For precision control. E.g.
    - Gripper must grip an egg with a force less then 30N
    - Head of inkjet printer moves to position 20.2mm to dispense a dot of red ink, then move to 20.25mm to dispense a dot of blue ink

Content

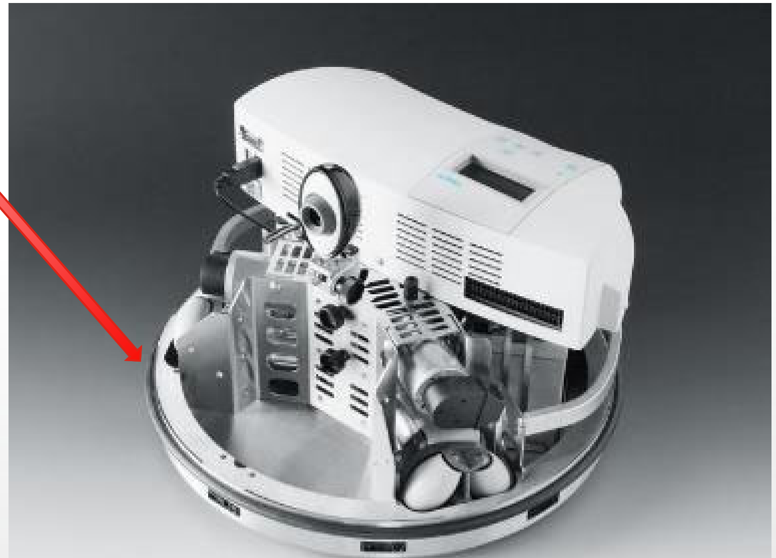
- Introduction
- Touch Sensors and Switches
    - Measure using myRIO
- Proximity Sensing
    - Measure using myRIO
- Position Measurement
    - Measure using myRIO
- Attachment: Writing FPGA Codes

Touch Sensors and Switches

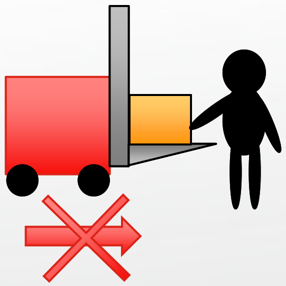
- Consists of an element that changes its state or its analog signal, when it touches an object.
- Usage examples:
  - Automated guided vehicles stops immediately after bumping into human
  - Robotic vacuum cleaner changes path after hitting the wall
  - Bumps sensor *(rubber around the body)*



Festo Robotino includes
Bumps sensors,
Infrared distance sensors,
and a color VGA camera



https://www.ros.org/news/robots/



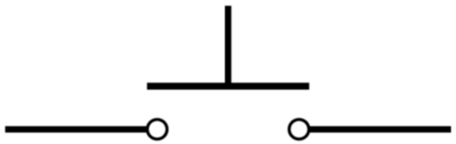
Robotino View software FREE to install:
https://www.festo-didactic.com/int-en/services/robotino/programming/robotino-view/?fbid=aW50LmVuLjU1Ny4xNy4zNC4xNDI2

RMIT UNIVERSITY

Touch Sensors and Switches

Switches

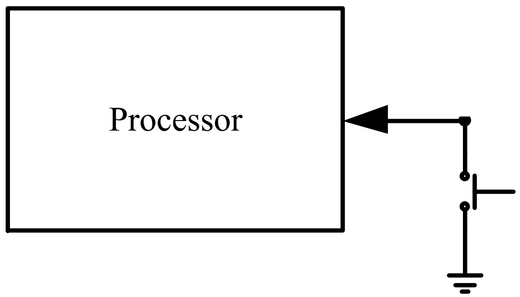
- Normally Open (NO) or Normally Closed (NC)


NO pushbutton


NC pushbutton

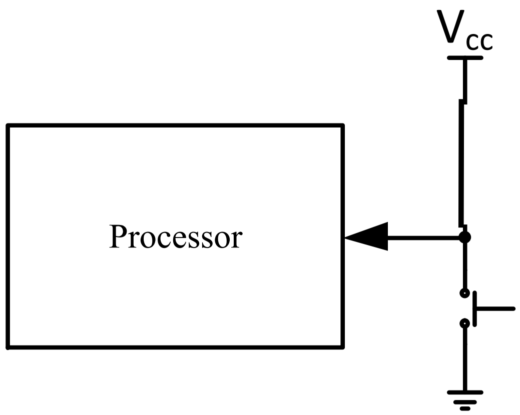
Aside: Pull-up Resistors

- What is the voltage level when:
  - Switch is closed?
    - 0V or logic 0
  - Switch is open?
    - Floating! Susceptible to noise → Small amount of noise can make the state change.



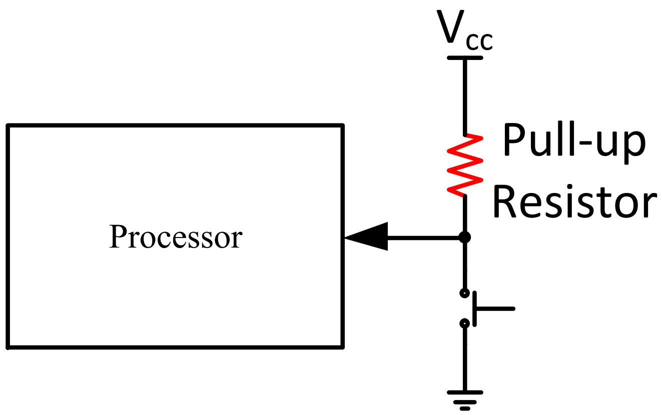
Aside: Pull-up Resistors

- What is the voltage level when:
  - Switch is closed?
    - Short circuit, dangerous!
    - Note: the processor has high input impedance (resistance) thus current “prefers” to go to ground.
  - Switch is open?
    - 5V or logic 1



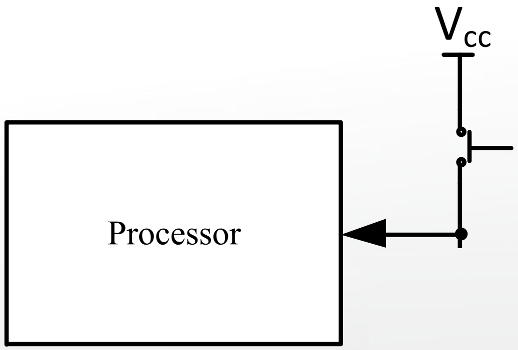
Aside: Pull-up Resistors

- Pull-up resistors:
  - Switch is closed:
    - The input to the processor is logical 0.
  - Switch is open:
    - The pull-up resistor allows the input to the processor to become logical 1.



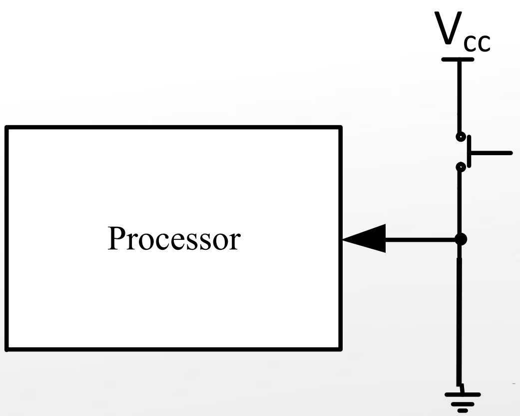
Aside: Pull-down Resistors

- What is the voltage level when:
  - Switch is closed?
    - 5V or logic 1
  - Switch is open?
    - Floating!



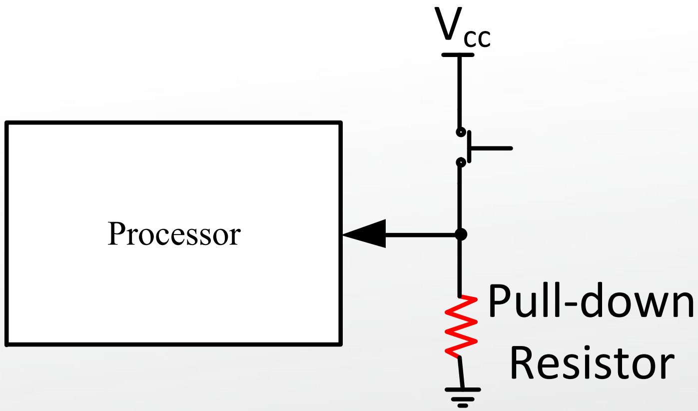
Aside: Pull-down Resistors

- What is the voltage level when:
  - Switch is closed?
    - Short circuit, dangerous!
  - Switch is open?
    - 0V or logic 0



Aside: Pull-down Resistors

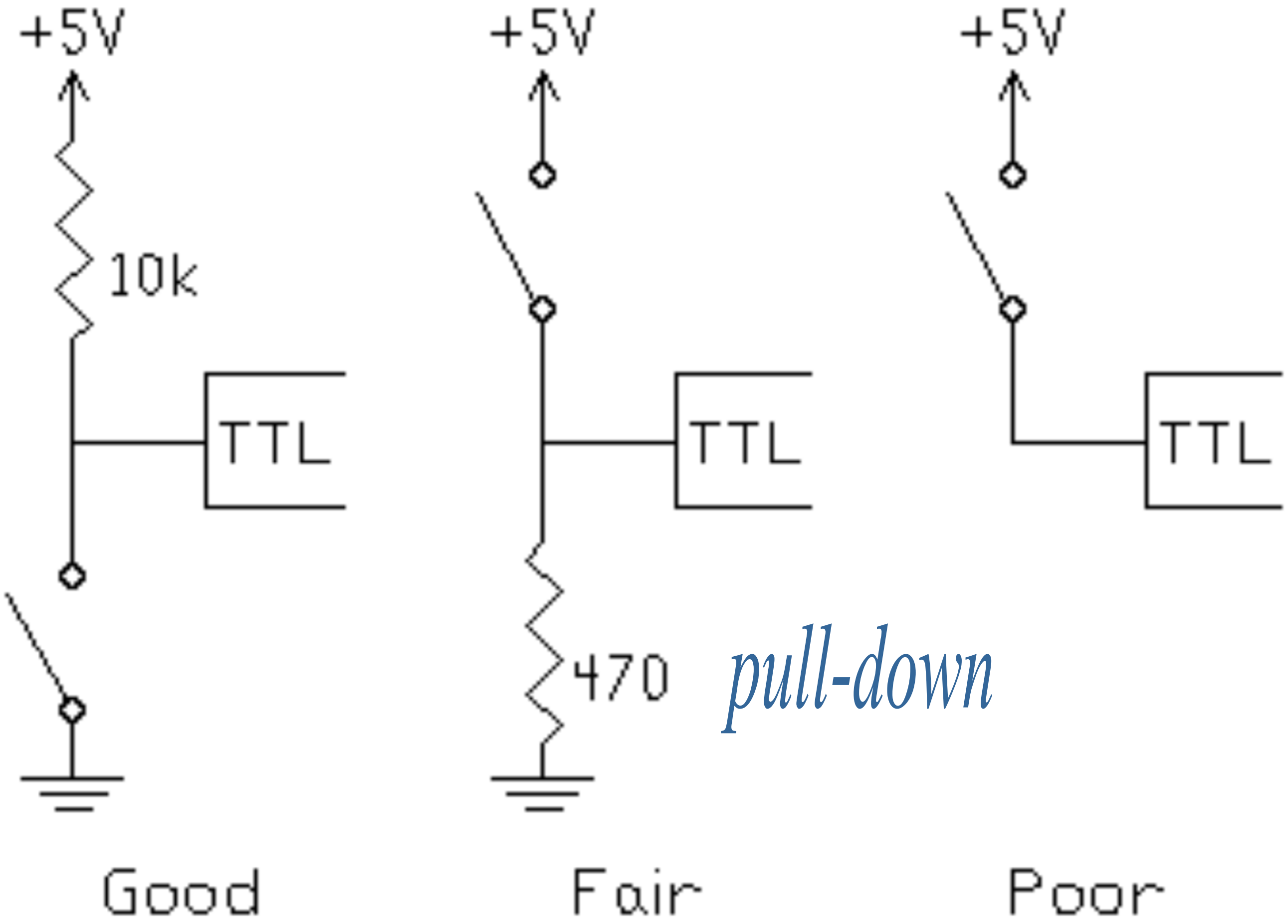
- Pull-down resistor:
  - Switch is closed:
    - The input to the processor is logical 1.
  - Switch is open:
    - The pull-down resistor allows the input to the processor to become logical 0.



Ports Interfacing - Input

Example

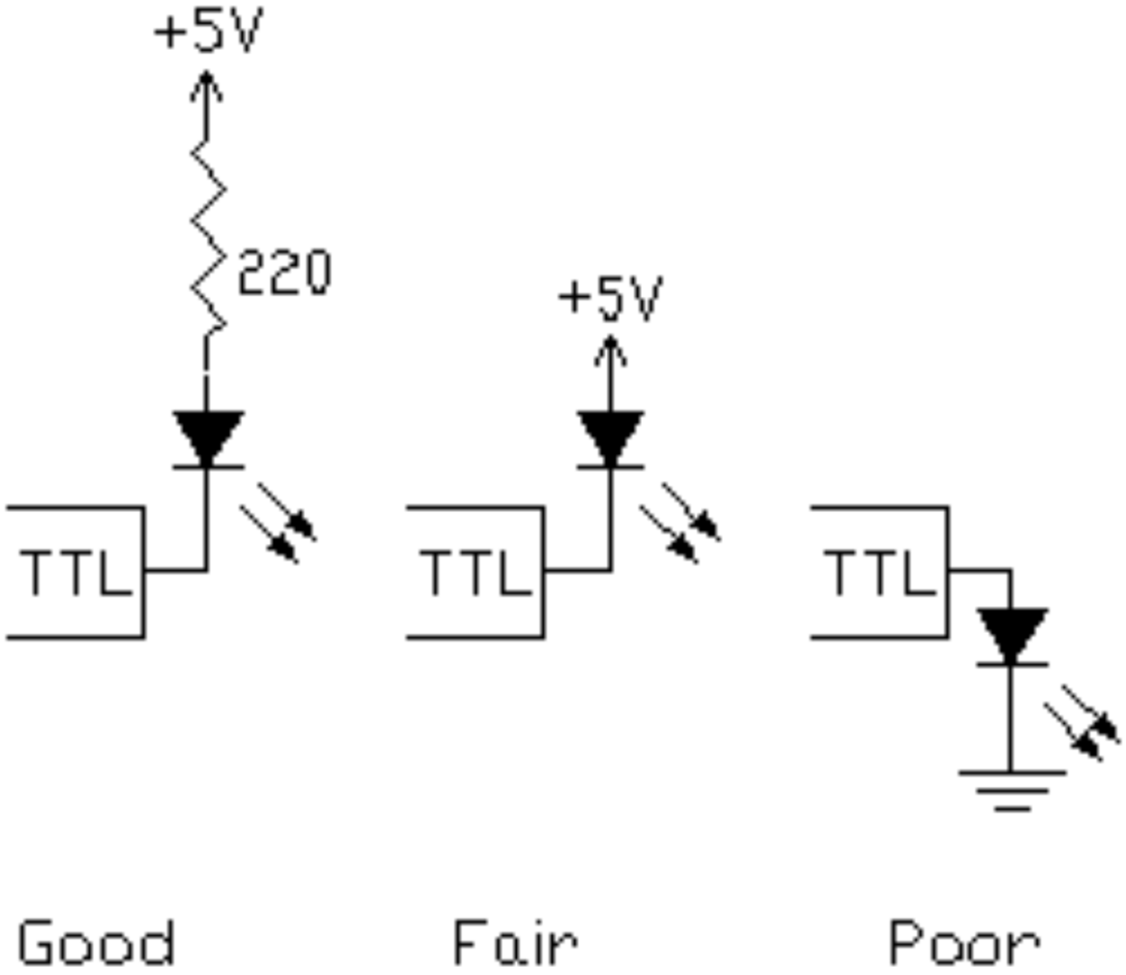
pull-up



pull-down

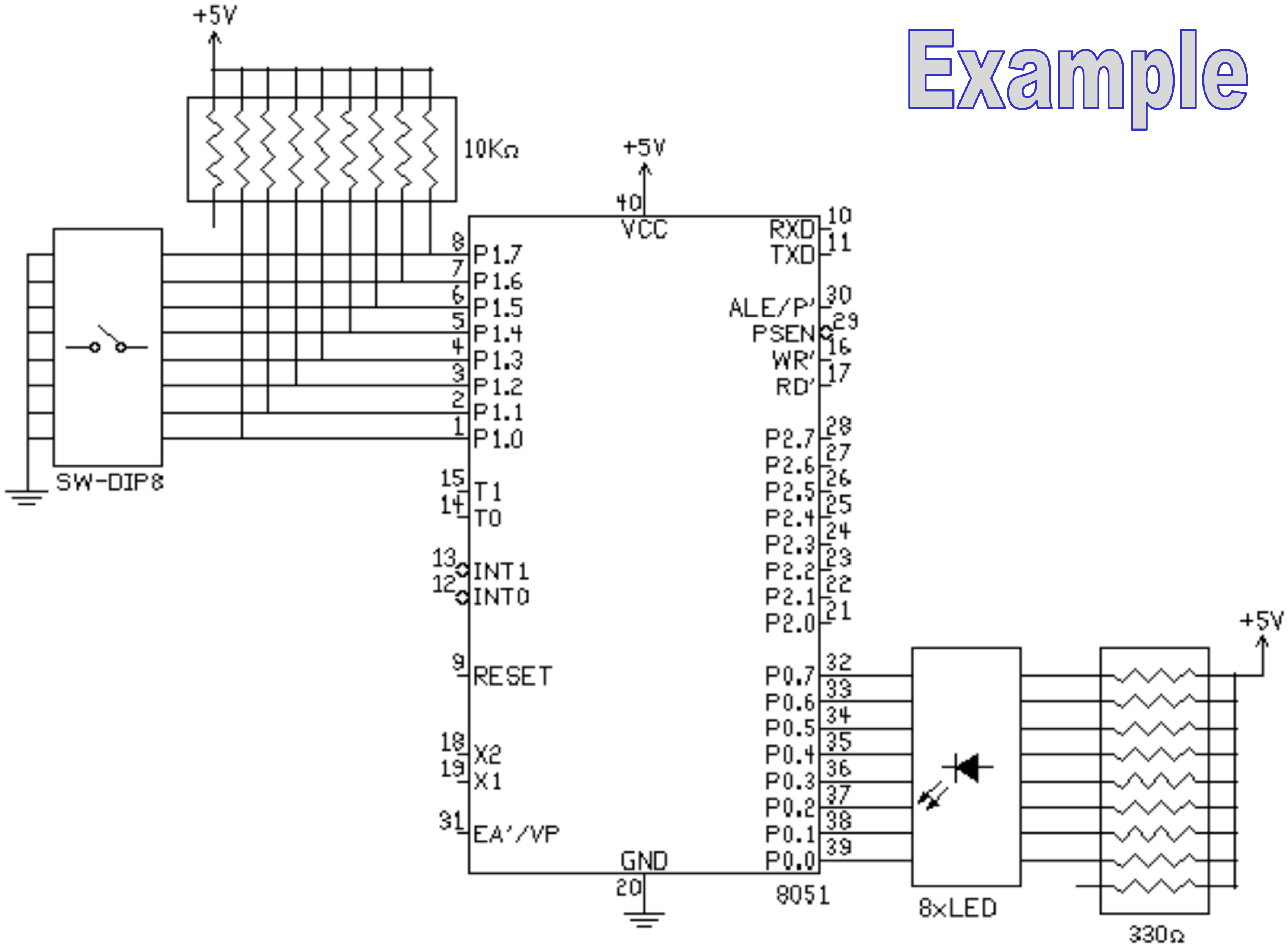
Ports Interfacing - Output

Example



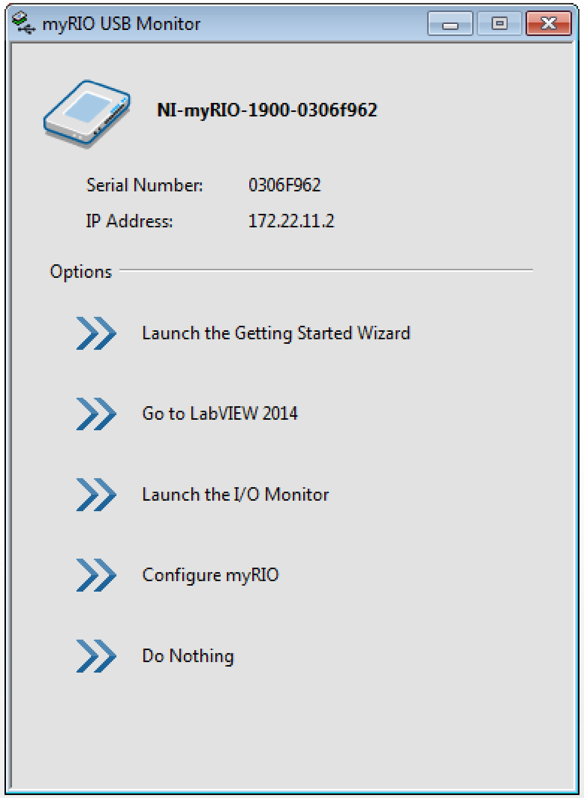
8051 Interfacing

Example



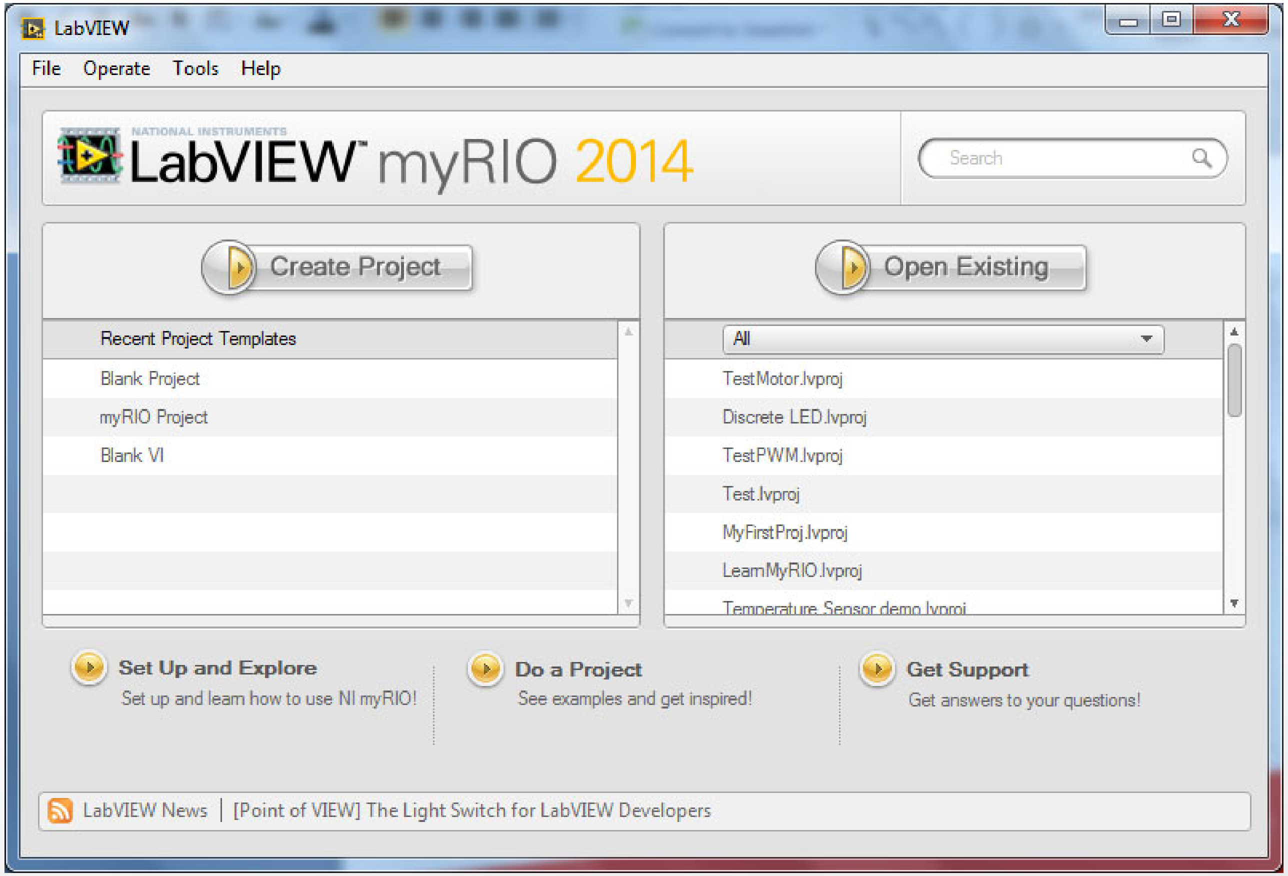
Use myRIO to read in Touch Sensors

- In this section, we will create a VI to read in the touch sensor signal, which is a digital input.
  - Every time a rising edge is detected, a counter will increment by 1.
- Connect myRIO to the PC via a USB cable.
- PC will automatically detect myRIO and the following dialogue will pop up:
  - Choose “Go to LabVIEW 201x”.



Use myRIO to read in Touch Sensors

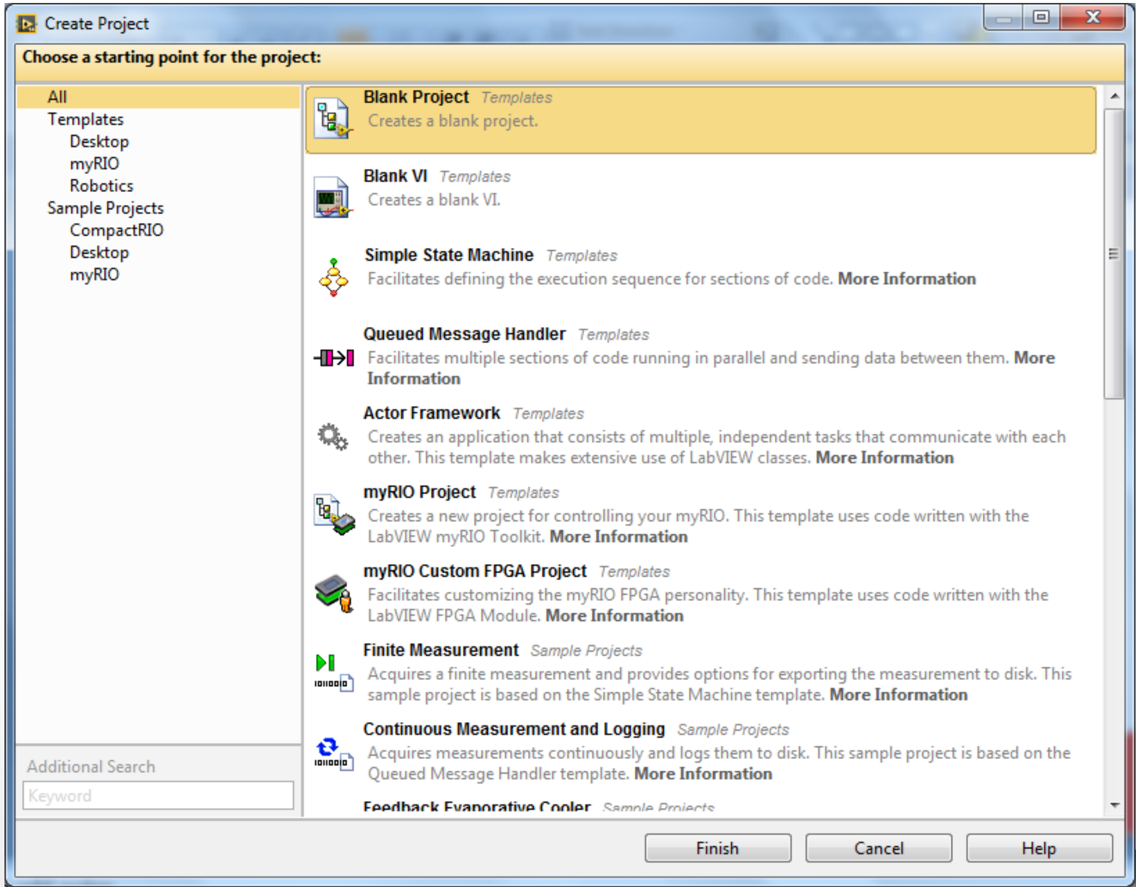
- LabVIEW will be started.



- Click on “Create Project”

Using Templates

- There are templates for blank project, myRIO project and myRIO custom FPGA projects.

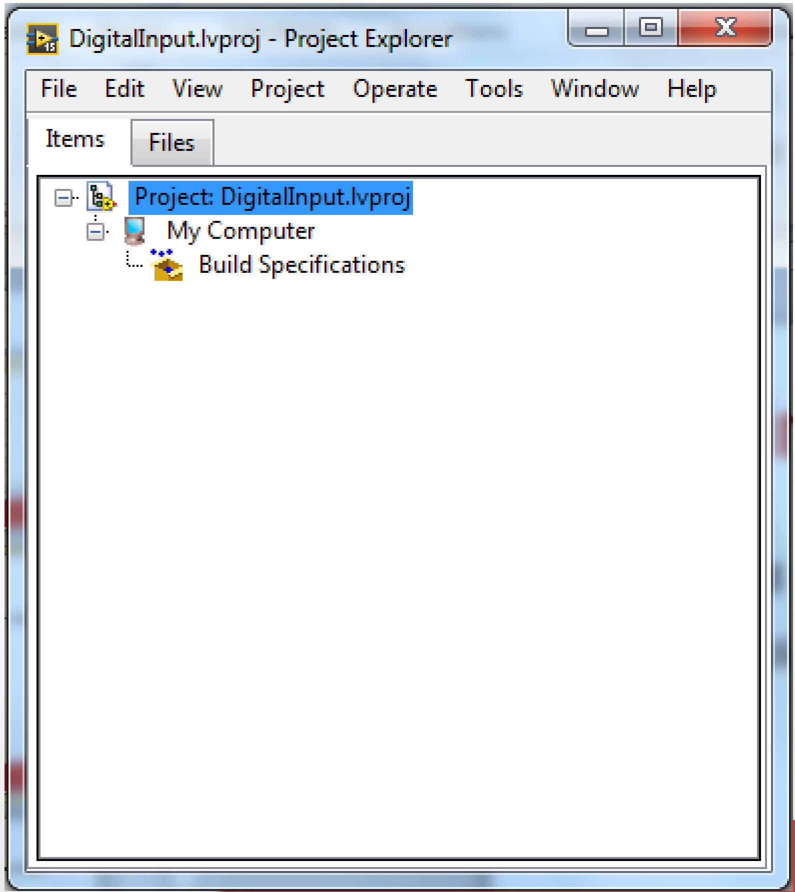


- Choose blank project.

RMIT UNIVERSITY

Setting up the Project

- You will see the project window.
- Right click on “Project: Untitled Project 1” → Save as → “DigitalInput”.



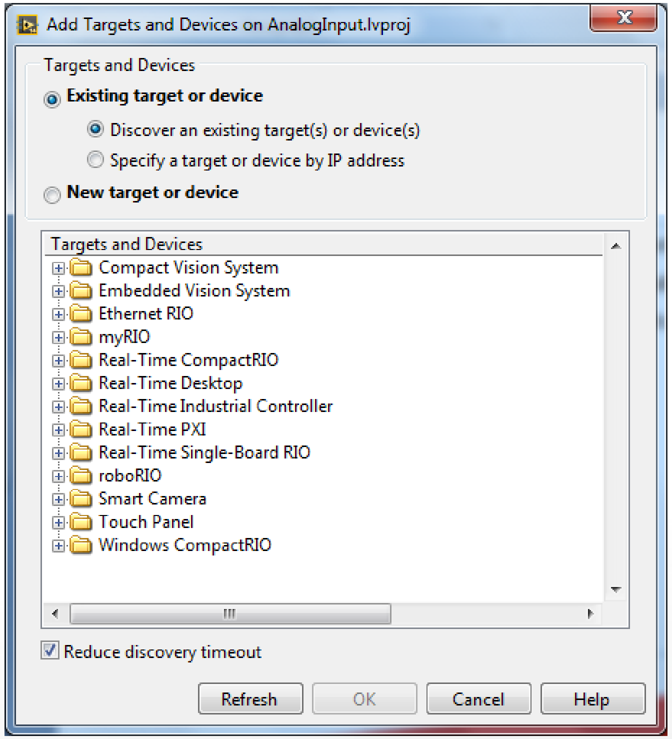
myRIO Project Hierarchy

- In a myRIO project, the program can run from three different “locations”:
  - PC (host):
    - Suggested: For non-time-critical tasks such as complex UI to visualize data.
  - Real-time processor:
    - Suggested: For time-critical tasks which needs to run in real-time, i.e. cannot be interrupted by lower priority tasks (anti-virus, UI etc.)
    - Also for data logging, file management.
  - FPGA:
    - Suggested: For tasks which are time-critical and has to be very fast, e.g. Accessing IO, PWM, reading encoder pulses, emergency stop.
- In this course, we will program everything in Real-Time processor.
- Notes on programming in FPGA will also be given but these are only for self-study. (Please refer to attachment).

RMIT UNIVERSITY

Setting up the Project

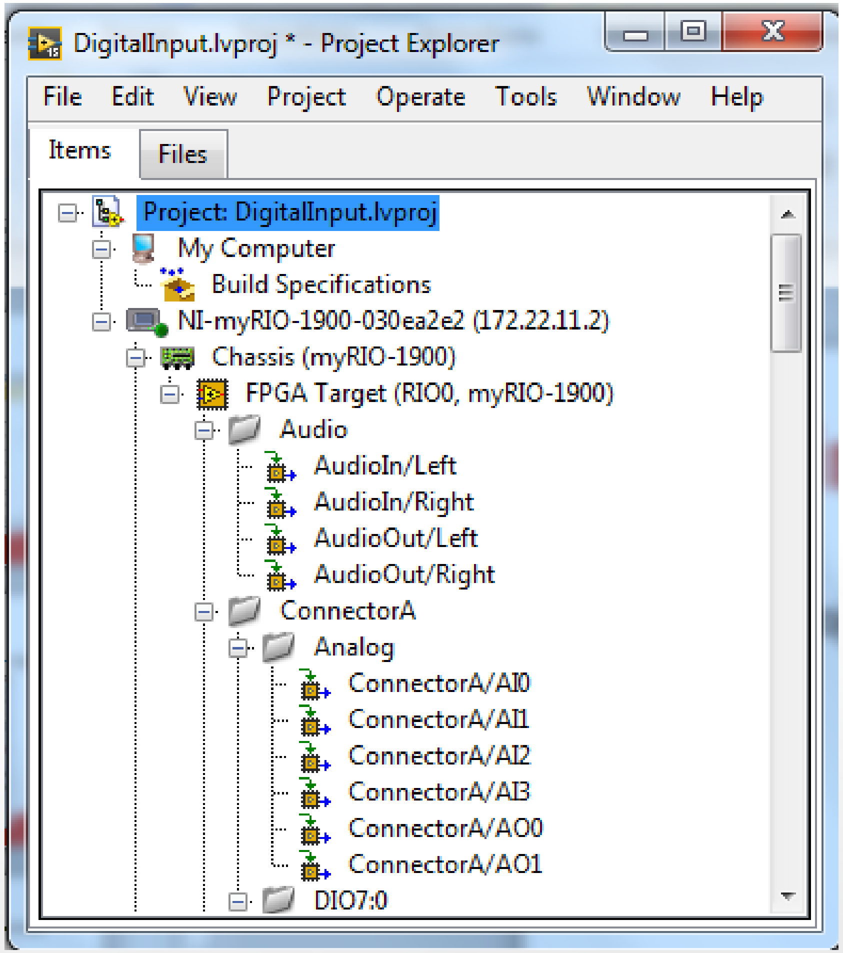
- Right click on Project DigitalInput → New → Targets and Devices



- Existing target or device:
  - Discover an existing target or device.
  - Or specify a target or device by IP address. (myRIO's IP is 172.22.11.2)
- Click on the "+" sign before myRIO.

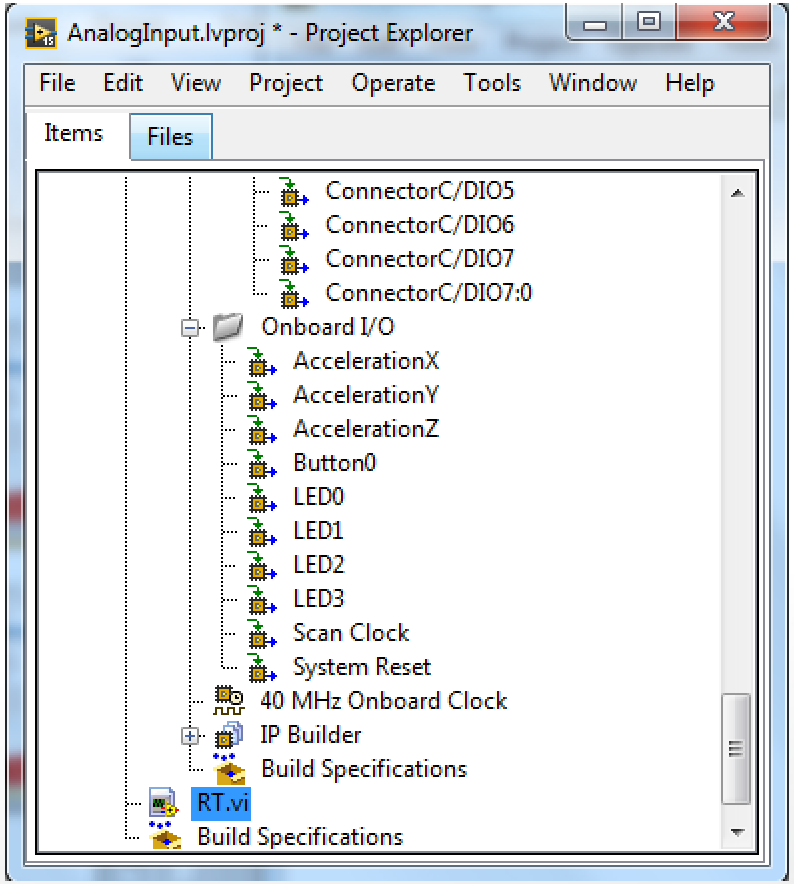
Setting up the Project

- The project tree now looks like this:



Setting up the Project

- Right click on “NI-myRIO-1900…” → New → VI
- A VI will open.
- Save it as RT.vi.
- Now we are ready to program the VI for reading digital inputs.

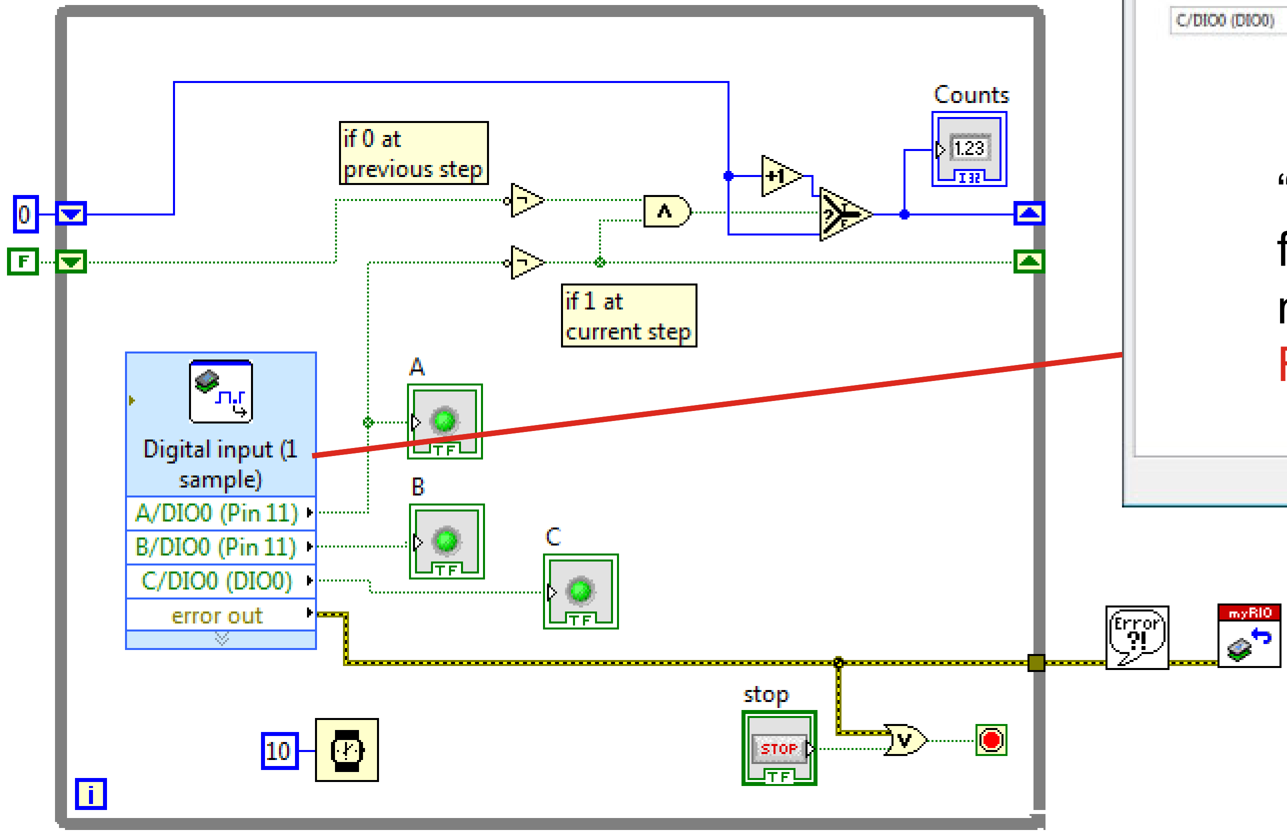


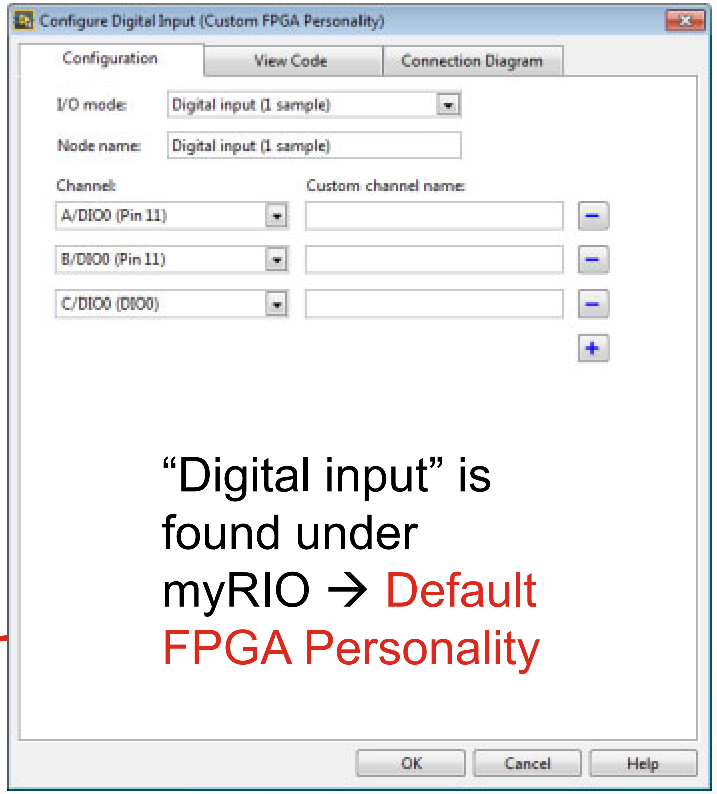
Digital Input

- Note:
  - On MXP connectors (A&B), the digital IO's have pull-up resistors.
    - Logic 1 if external switch is open
    - Logic 0 if external switch is closed
  - On MSP connector (C), the digital IO's have pull-down resistors.
    - Logic 0 if external switch is open
    - Logic 1 if external switch is closed
- In our VI, we will also show this characteristics of the digital inputs.

Digital Input

- We will create the following VI:

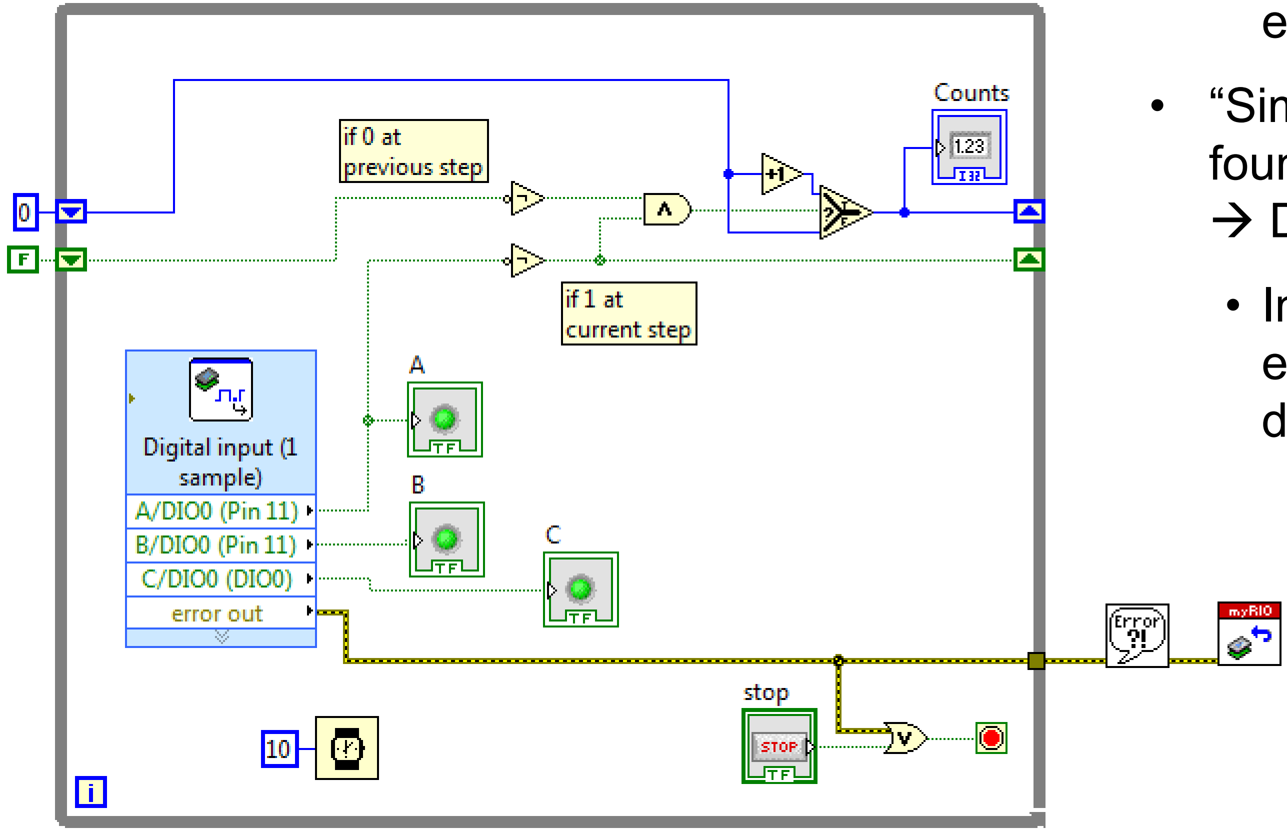




"Digital input" is found under myRIO → Default FPGA Personality

Digital Input

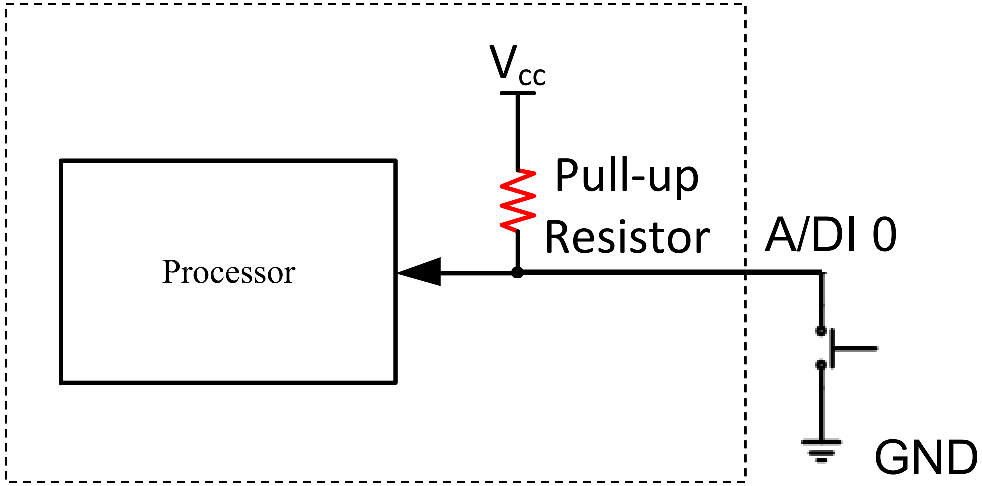
- We will create the following VI:



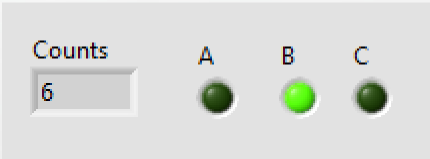
- “Reset myRIO” is found under myRIO → Device Manager.
  - It is to reset myRIO at the end of the application.
- “Simple Error Handler” is found under Programming → Dialog & User Interface.
  - Indicates whether an error occurs and description of the error.

Digital Input

- The circuit is as follows:



- Because B and C are not connected to any external signal, B shows logic 1 and C shows logic 0, due to the internal pull-up (B) and pull-down (C) resistors.
- A will react to us switching the lever up and down, and counts will increase every time we switch from 0 to 1.



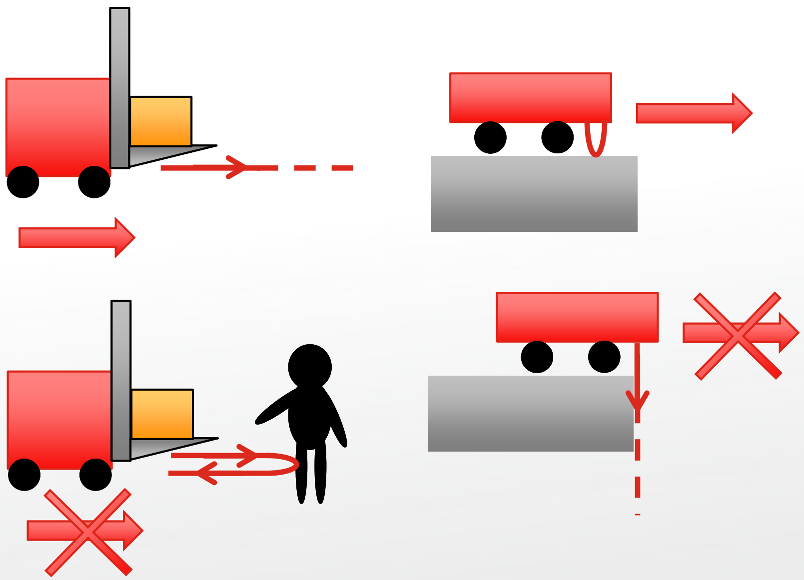
(red not connected)

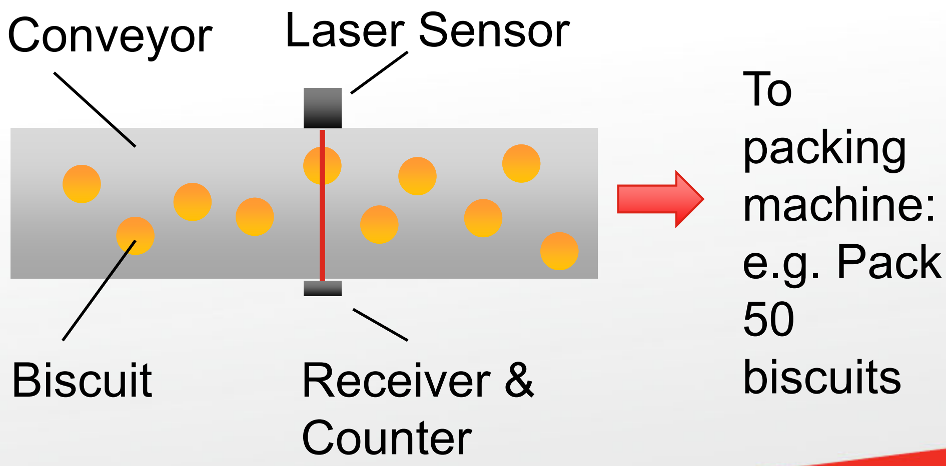
Content

- Introduction
- Touch Sensors and Switches
    - Measure using myRIO
- Proximity Sensing
    - Measure using myRIO
- Position Measurement
    - Measure using myRIO
- Attachment: Writing FPGA Codes

Proximity Sensors

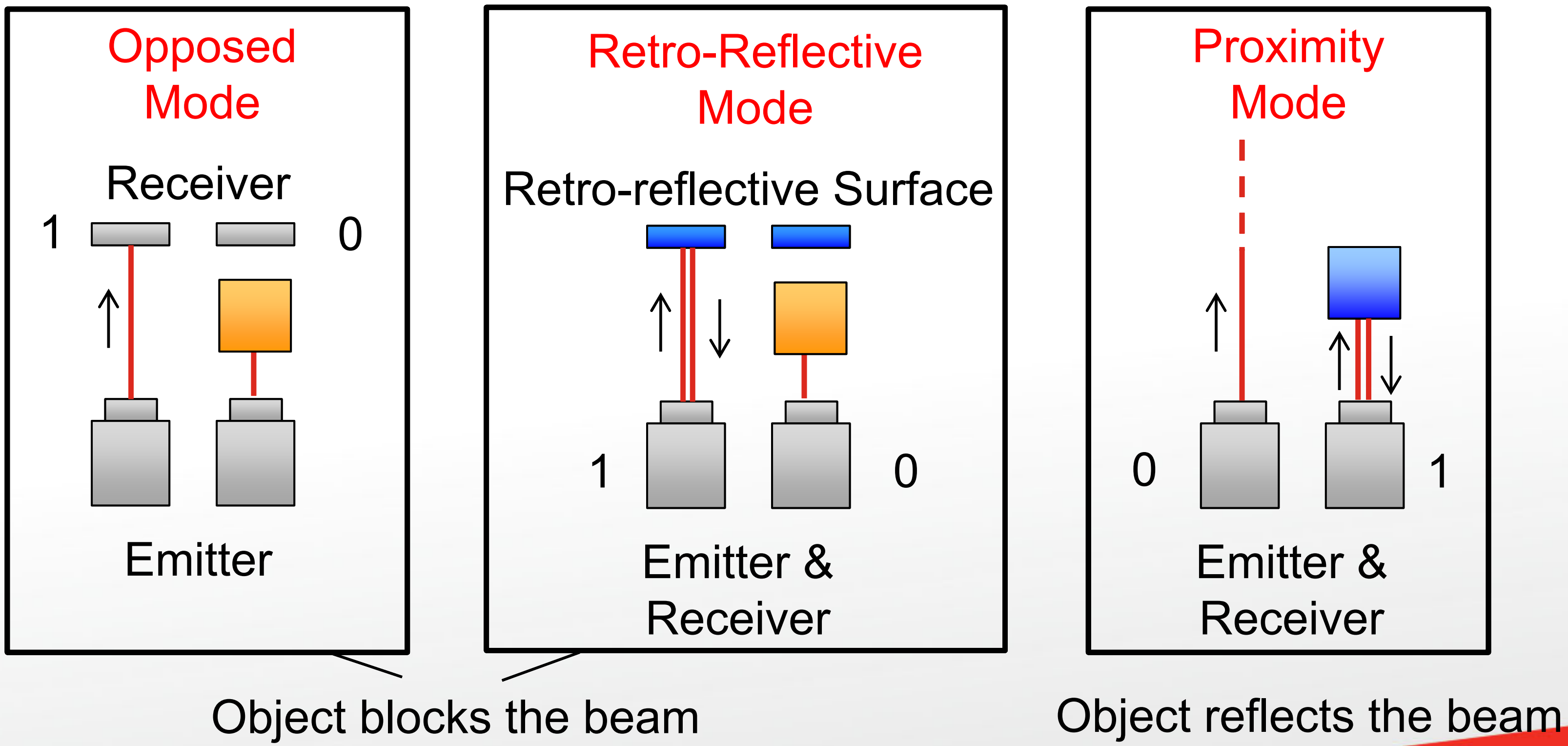
- Consists of an element that changes its state or its analog signal, when it is close to an object (but not actually touching).
- Usage examples:
  - Automated forklift avoids bumping into human
  - Wheeled robot avoids falling down a cliff
  - Counting moving objects on a conveyor belt
  - Detects object and command robot to pick it up





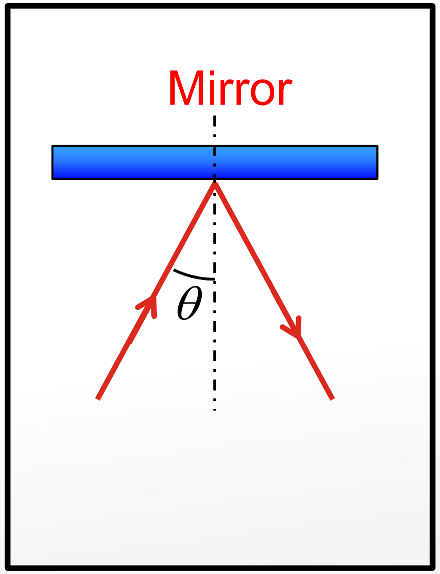
Proximity Sensors (1)

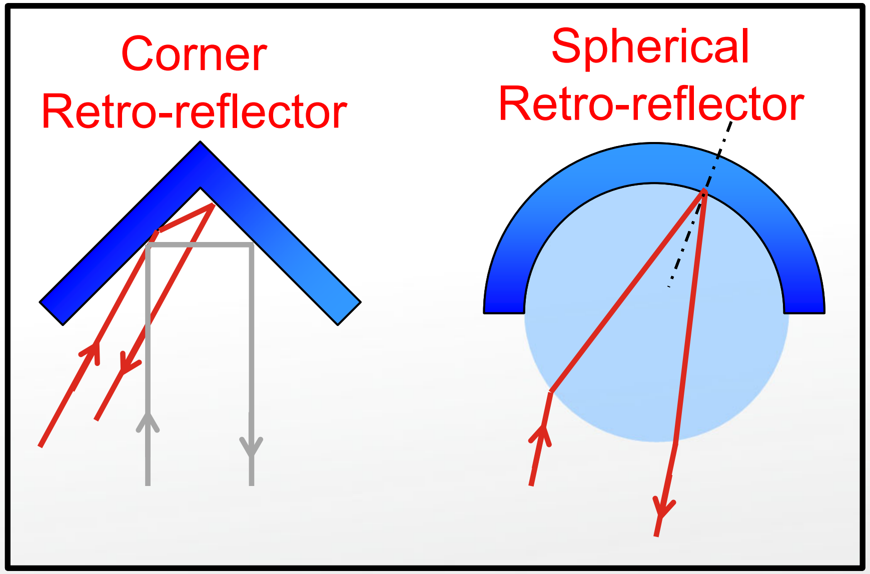
- Photoemitter-detector pair
  - Emitter: Laser, LED
  - Detector: Phototransistor, Photodiode
  - Different configurations:



What is Retroreflector

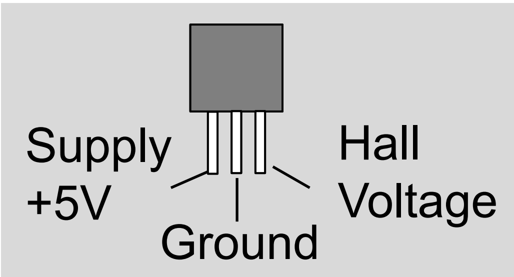
- A device or surface that reflects light back to its source with a minimum of scattering.
- Light is reflected back parallel to the direction of light's source, without a need of zero angle of incidence (unlike mirror).

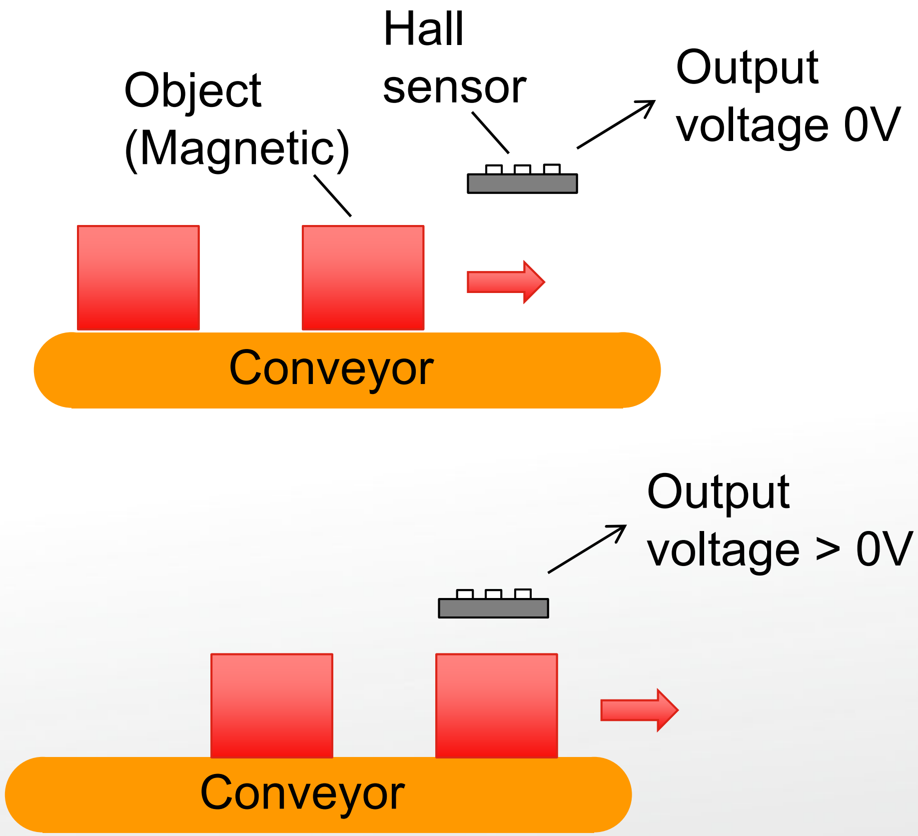




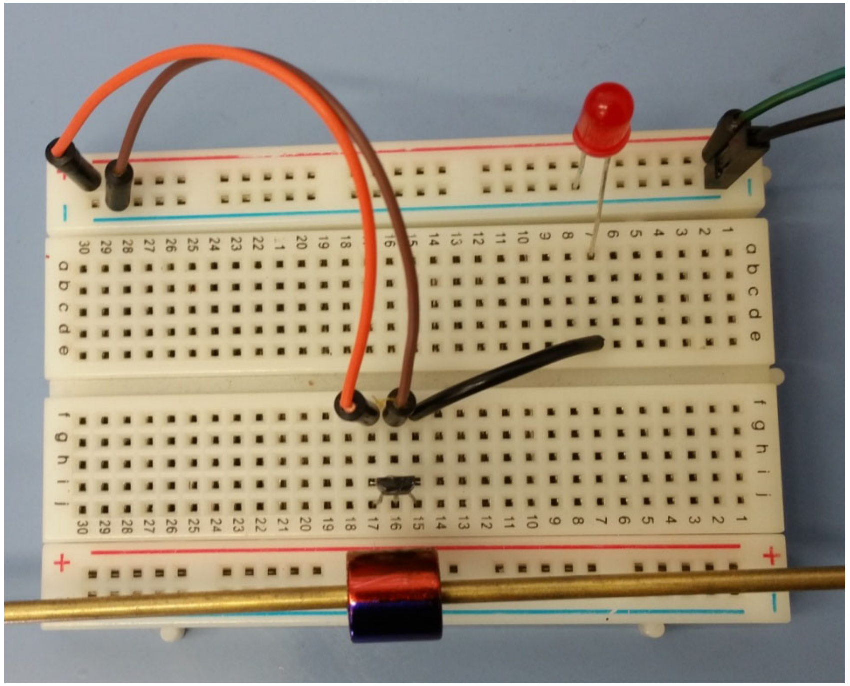
Proximity Sensors (2)

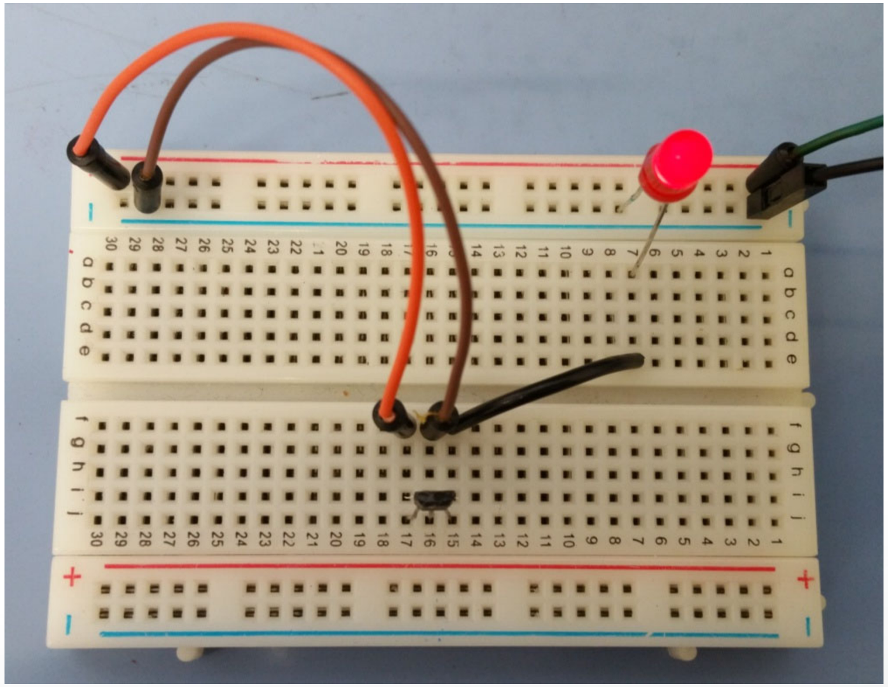
- Magnetic: Hall effect sensor
  - If magnetic object is in the proximity of the sensor, “Hall Voltage” will be produced.



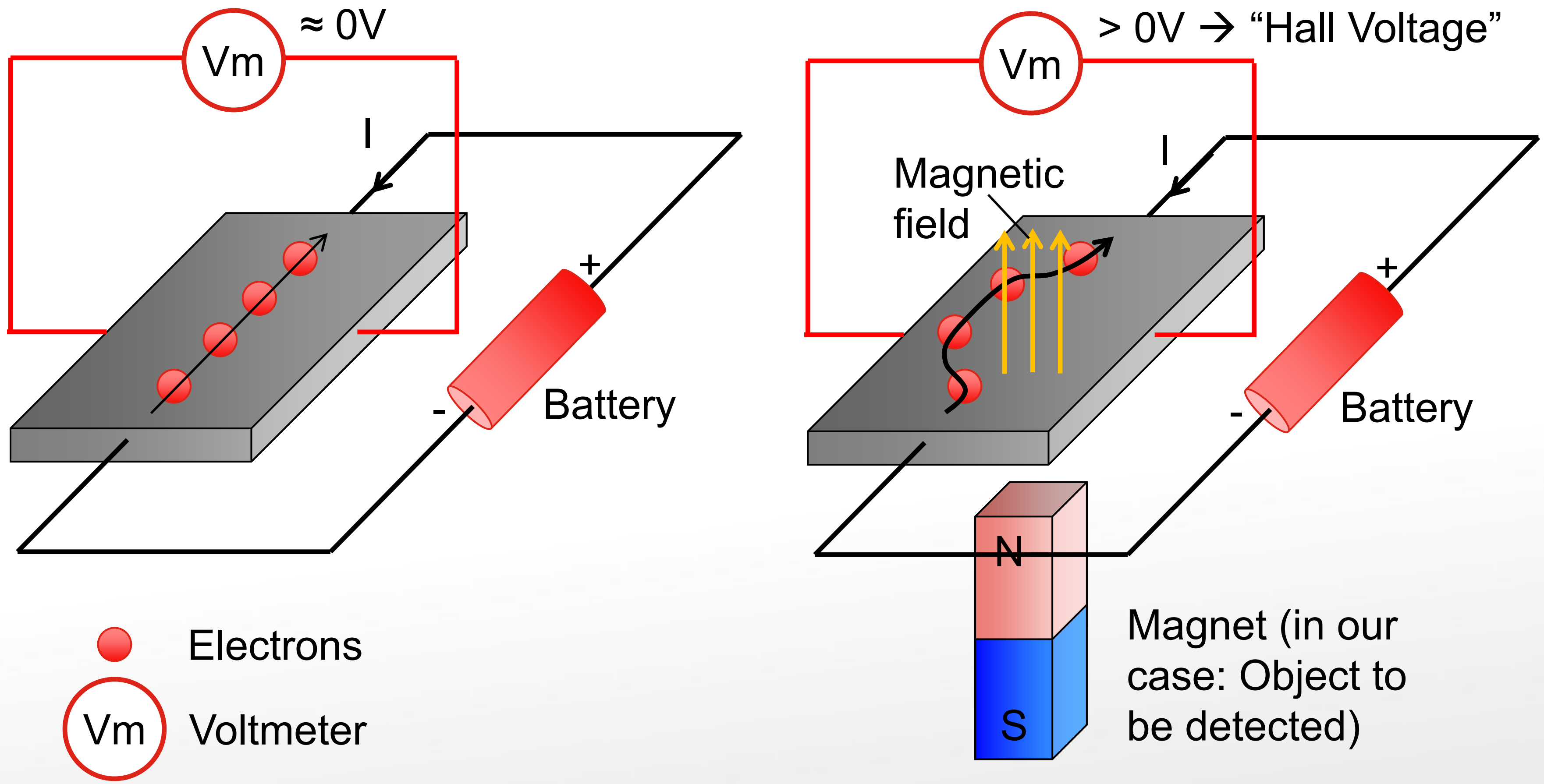


What is Hall Effect



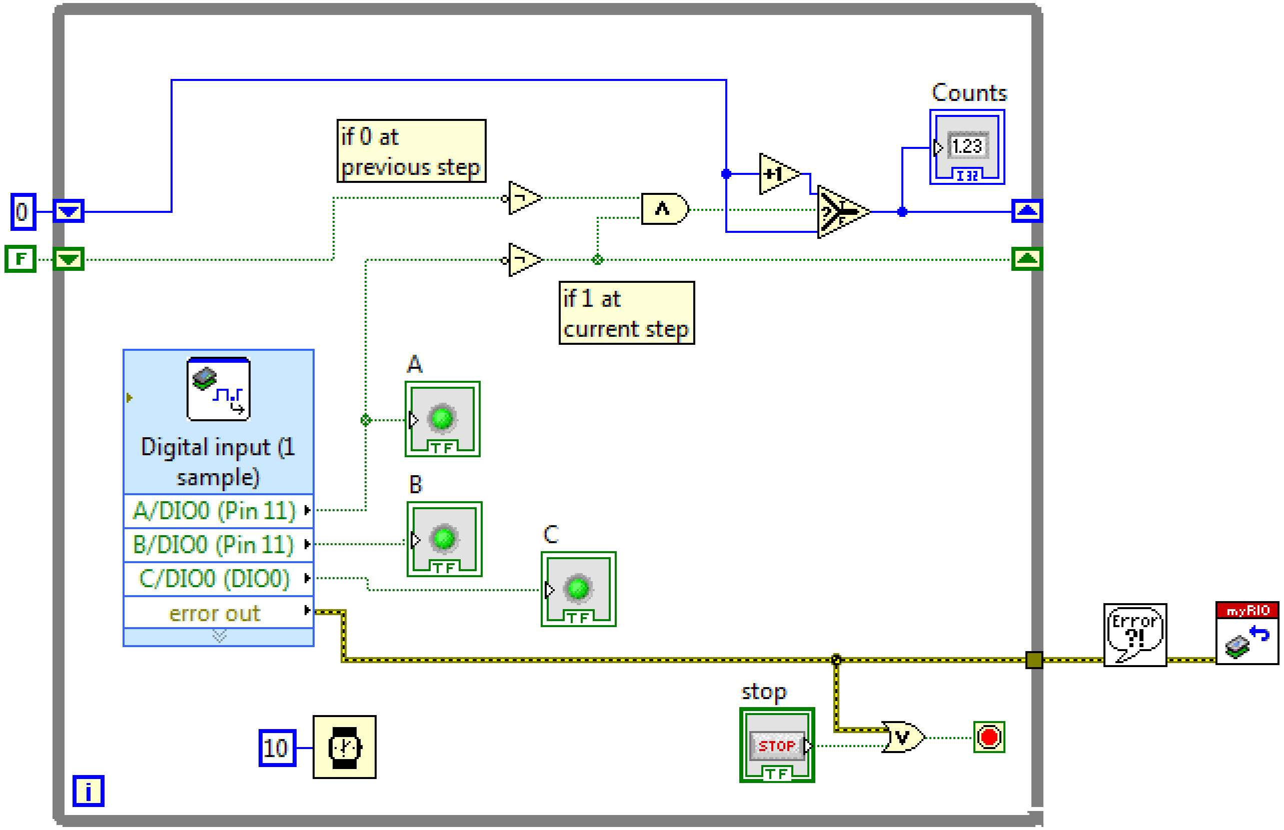


What is Hall Effect



Use myRIO to read Proximity Sensor

- Proximity Sensors are mostly digital sensors.
- They output logical 1 or 0 depending on whether object is detected.
- Therefore, we can use exactly the same code as in the previous section, for reading proximity sensors.



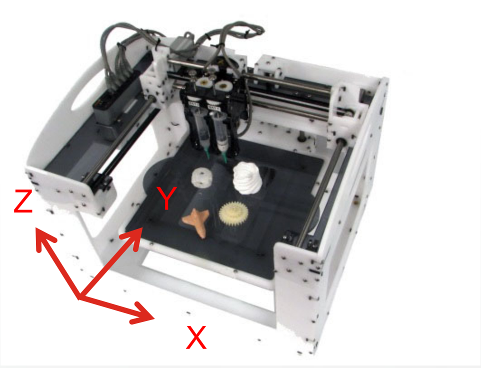
RMIT UNIVERSITY

Content

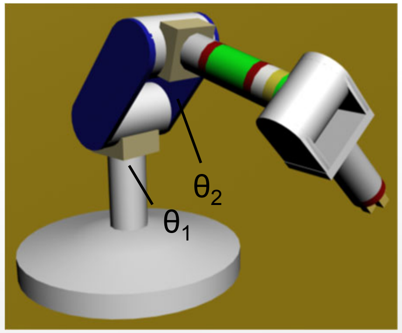
- Introduction
- Touch Sensors and Switches
    - Measure using myRIO
- Proximity Sensing
    - Measure using myRIO
- Position Measurement
    - Measure using myRIO
- Attachment: Writing FPGA Codes

Position Sensors

- As the name suggests, position sensors are for position or distance measurements.
- Can be absolute position or relative displacement.
- The measurement axis can be linear or angular.
- Usage examples:
  - The 3D printer head must move to x = 20mm, y = 30mm and z = 15mm.
  - The robot arm must move to $\theta_1 = 40^{\circ}$, $\theta_2 = 25^{\circ}$, …
  - The CNC machine tool must move from x = 100mm to x = 200mm.



3D Printer
https://en.wikipedia.org/wiki/File:Fab@Home_Model_2_3D_printer.jpg



Articulated Arm
https://commons.wikimedia.org/wiki/File:Robot_arm_model_1.png

Milling Machine
https://commons.wikimedia.org/wiki/File:Makino-S33-MachiningCenter-example.jpg

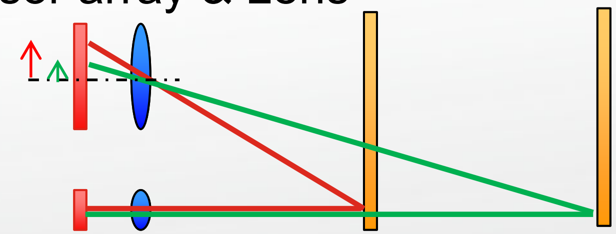
RMIT UNIVERSITY

Position Sensors (1)

- IR Range Finder

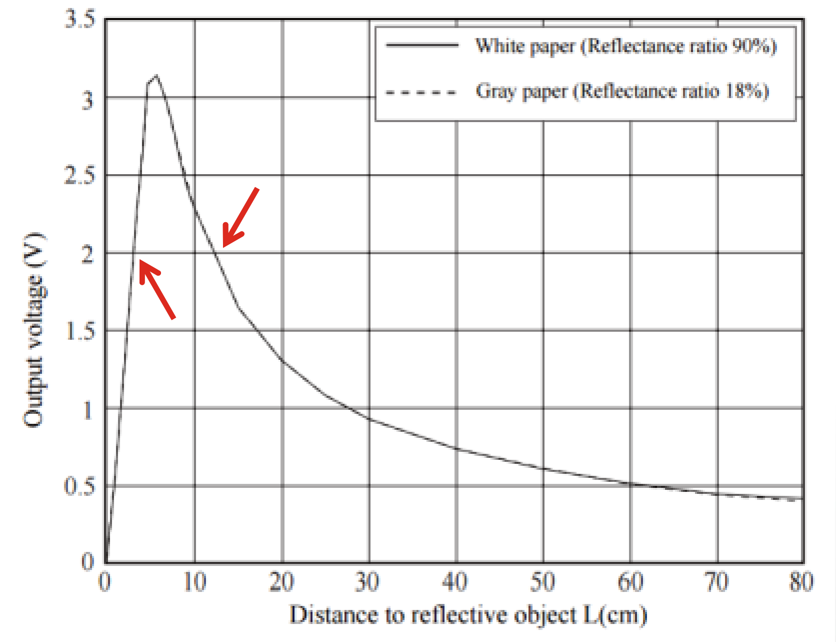
IR Range Finder
https://commons.wikimedia.org/wiki/File:Sharp_GP2Y0A21YK_IR_proximity_sensor_cropped.jpg

- The LED is pulse modulated at several tens of kilohertz.
- The infrared sensor is tuned to have the same frequency.
    - Relatively insensitive to ambience lightings.
- Working principle:



Analog Input – IR Range Sensor

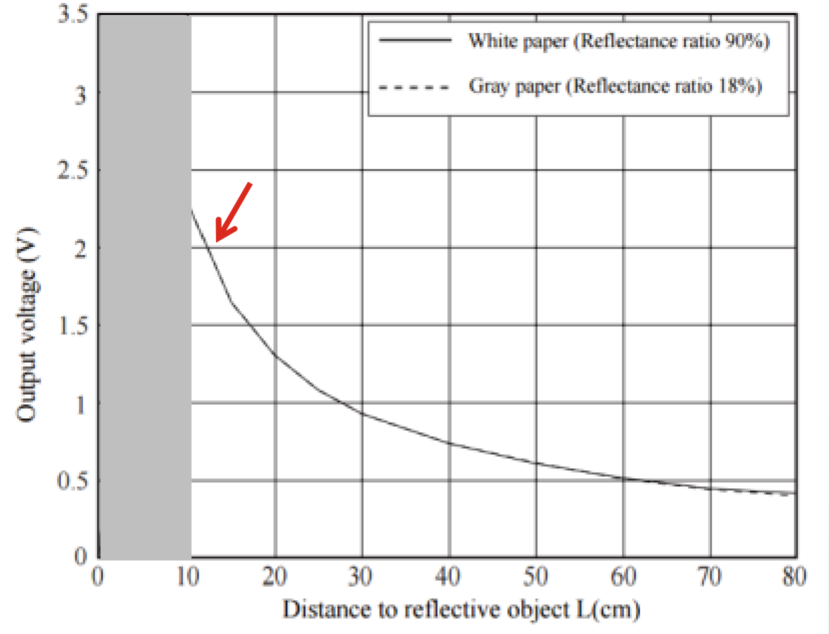
- The response of the IR Range Sensor is typically as follows (x-scale depending on model):



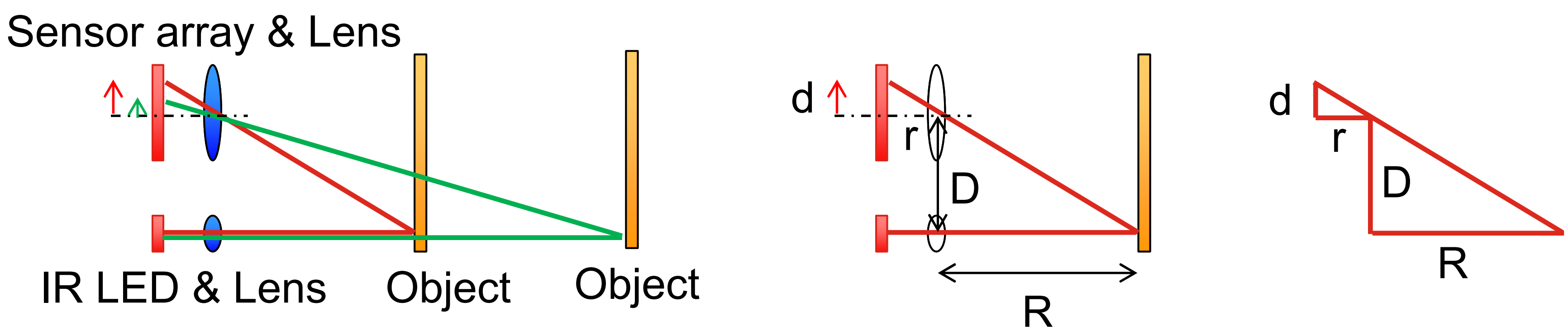
- Do you see any problem?
- If you get a reading of 2V, is the distance about 3cm or 13cm?

Analog Input – IR Range Sensor

- We need to constrain mechanically / physically that the sensor is never nearer than10cm to the surface (depends on model), in order to get one clear answer.



IR Range Finder - Calculations



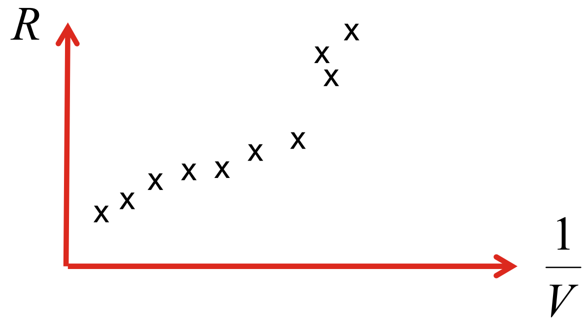
- Similar triangles: $\frac{d}{D} = \frac{r}{R}$ $\longrightarrow$ $R = \frac{rD}{d}$
- d is sensed by sensor array as $d = kV$ where k is a constant and V is voltage.
- Therefore: $R = \frac{rD}{kV} = \frac{C}{V}$ where $C = \frac{rD}{k}$.

IR Range Finder - Calibration

- Aim: To find the constant C in

$$R = \frac{rD}{kV} = \frac{C}{V}$$

- Take multiple measurements of R (known, e.g. using ruler) and V.
- Plot R against V^{-1}.

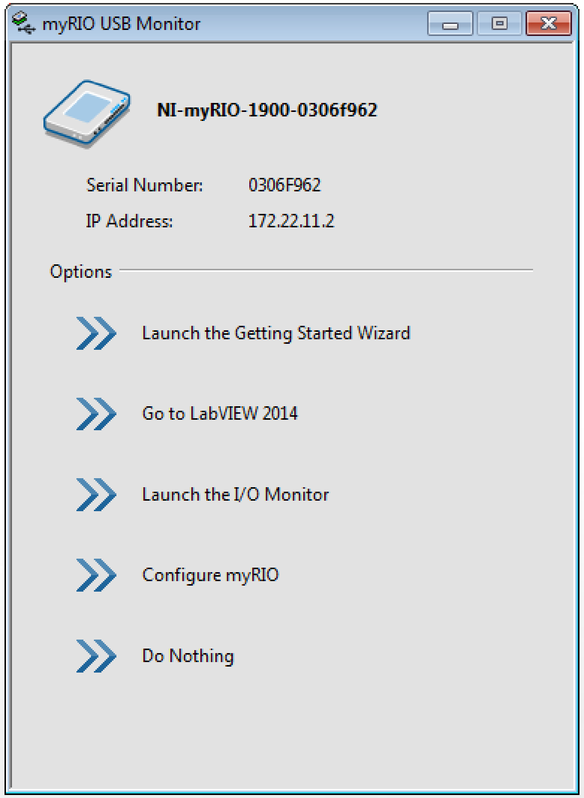


- Find linear region and get best fit line (e.g. least squares).

$$R = C\left(\frac{1}{V}\right) + b$$

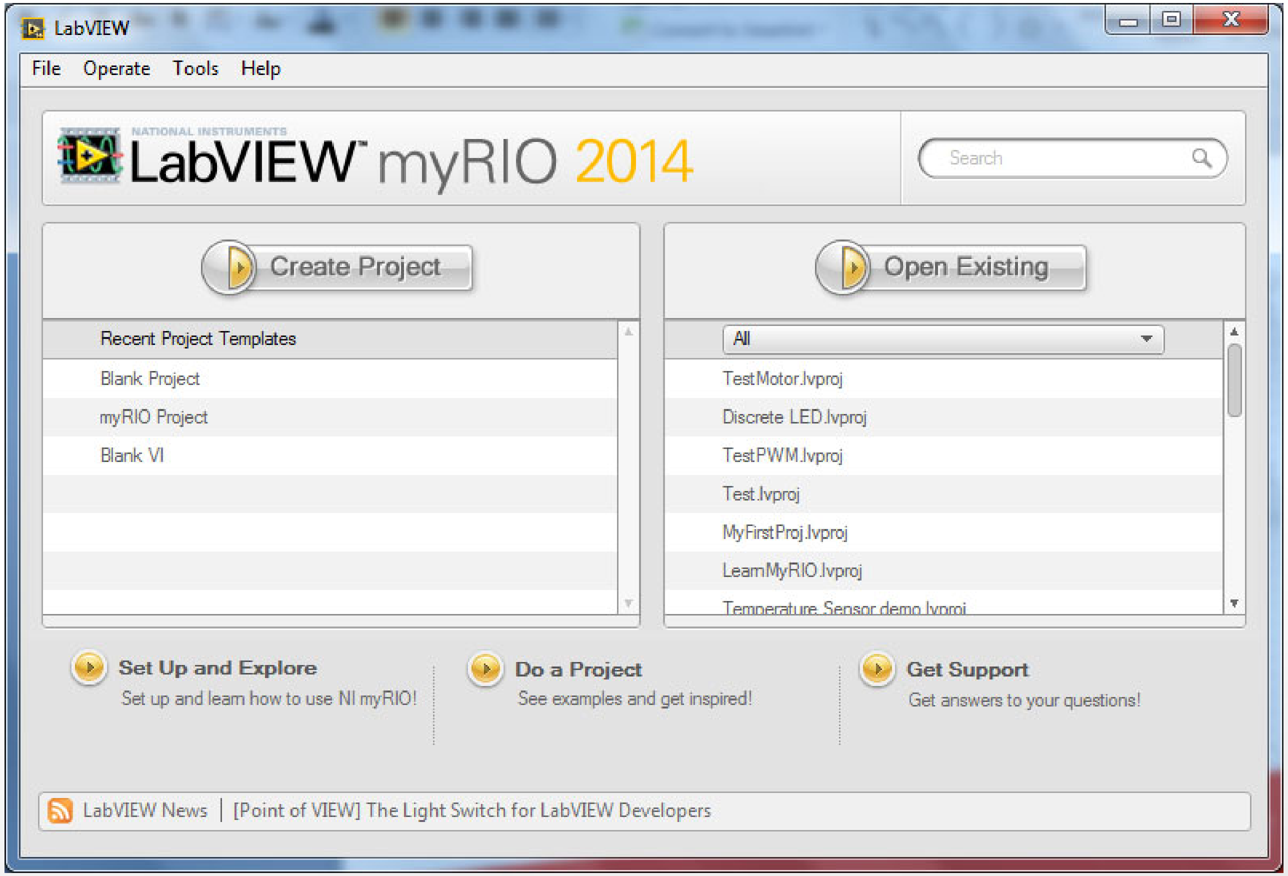
Use myRIO to read Range Sensor

- Now let's set up a project in LabVIEW to read in and calibrate the range sensor signals.
- As practice, let's start all over again.
- Unplug and plug-in myRIO device again.
- On the pop-up menu, choose "Go to LabVIEW 201x".



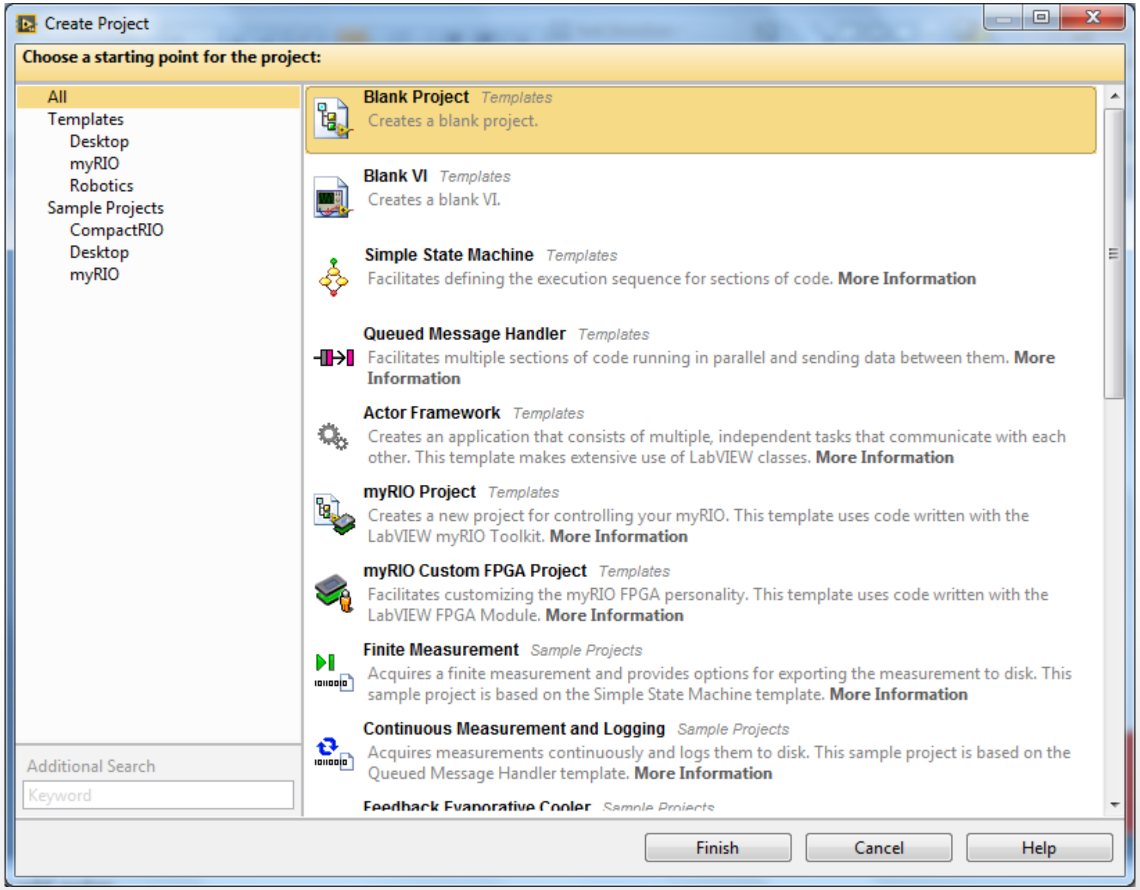
Setting up the Project

- Click on “Create Project”



Setting up the Project

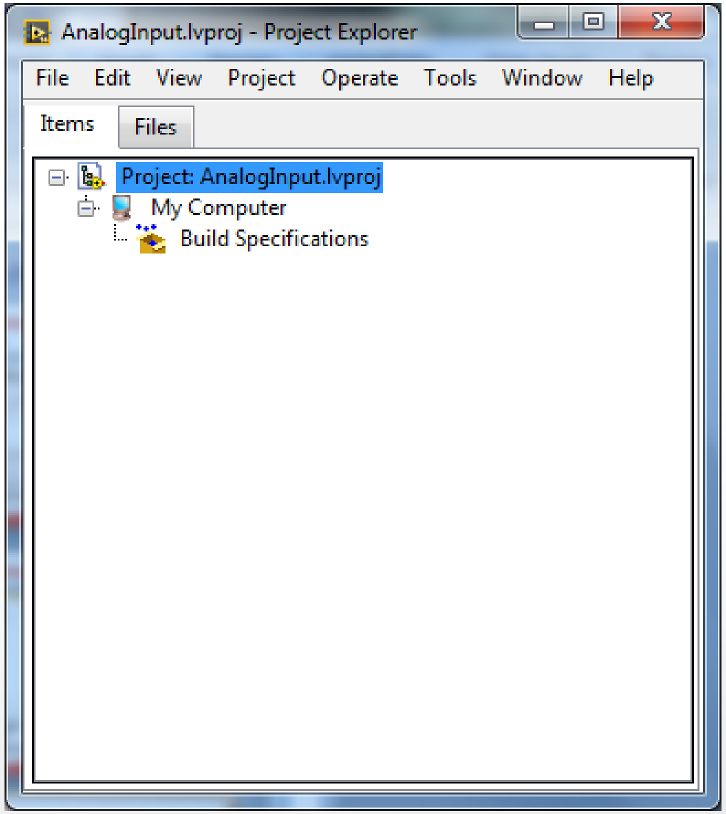
- Choose blank project.



RMIT UNIVERSITY

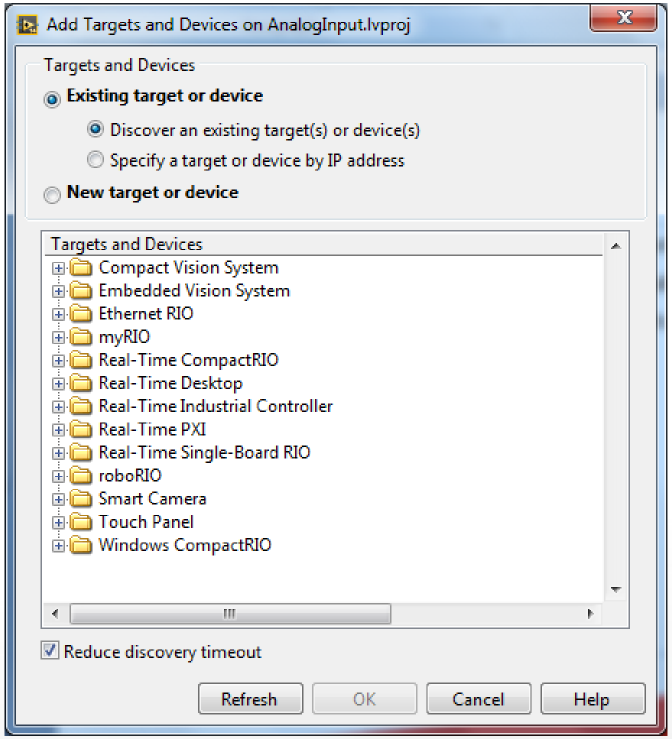
Setting up the Project

- You will see the project window.
- Right click on "Project: Untitled Project 1" → Save as → "AnalogInput".



Setting up the Project

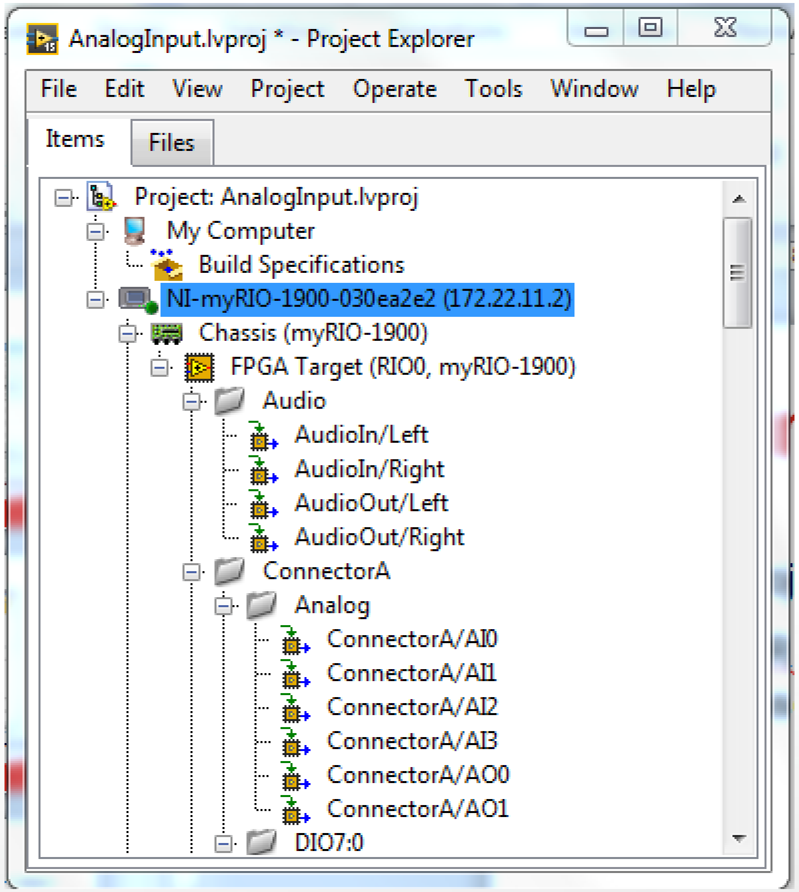
- Right click on Project AnalogInput → New → Targets and Devices



- Existing target or device:
  - Discover an existing target or device.
  - Or specify a target or device by IP address. (myRIO's IP is 172.22.11.2)
- Click the "+" sign before myRIO.

Setting up the Project

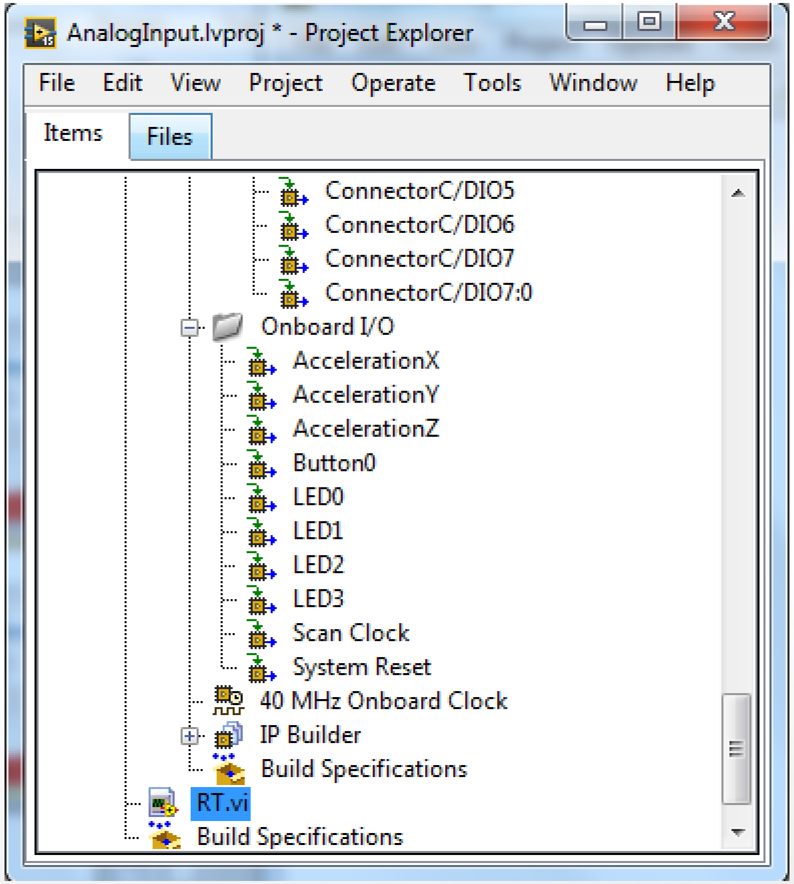
- The project tree now looks like this:



RMIT UNIVERSITY

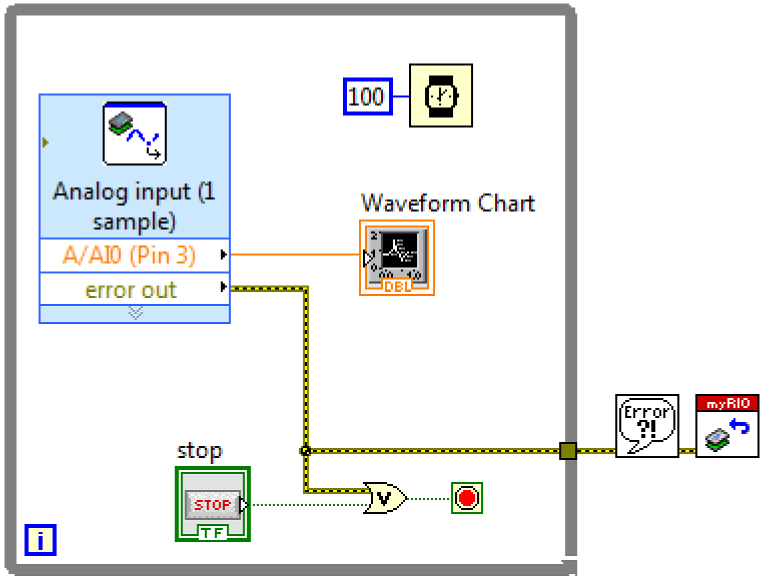
Setting up the Project

- Right click on “NI-myRIO-1900…” → New → VI
- A VI will open.
- Save it as RT.vi.
- Now we are ready to program the VI for reading analog inputs.



Reading & Calibrating Range Sensor

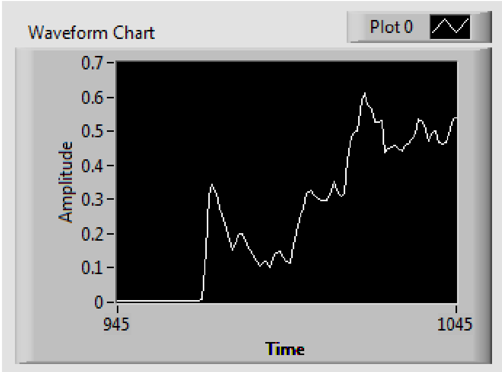
- Next, create the following VI in RT.vi:



- Connect the IR Range Sensor to the myRIO board as follows:
  - Red → 5V
  - Black → Ground
  - White → AI 0

Reading & Calibrating Range Sensor

- Place your palm over the range sensor and move it closer or further, and you should be able to see the values in the waveform chart varying.

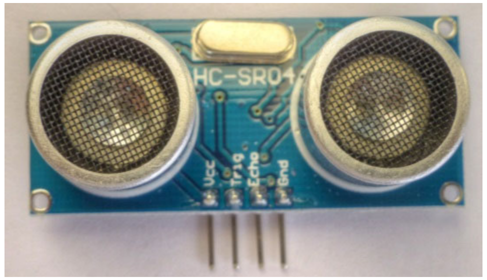


- Now, please take 15 minutes to calibrate your range sensor.

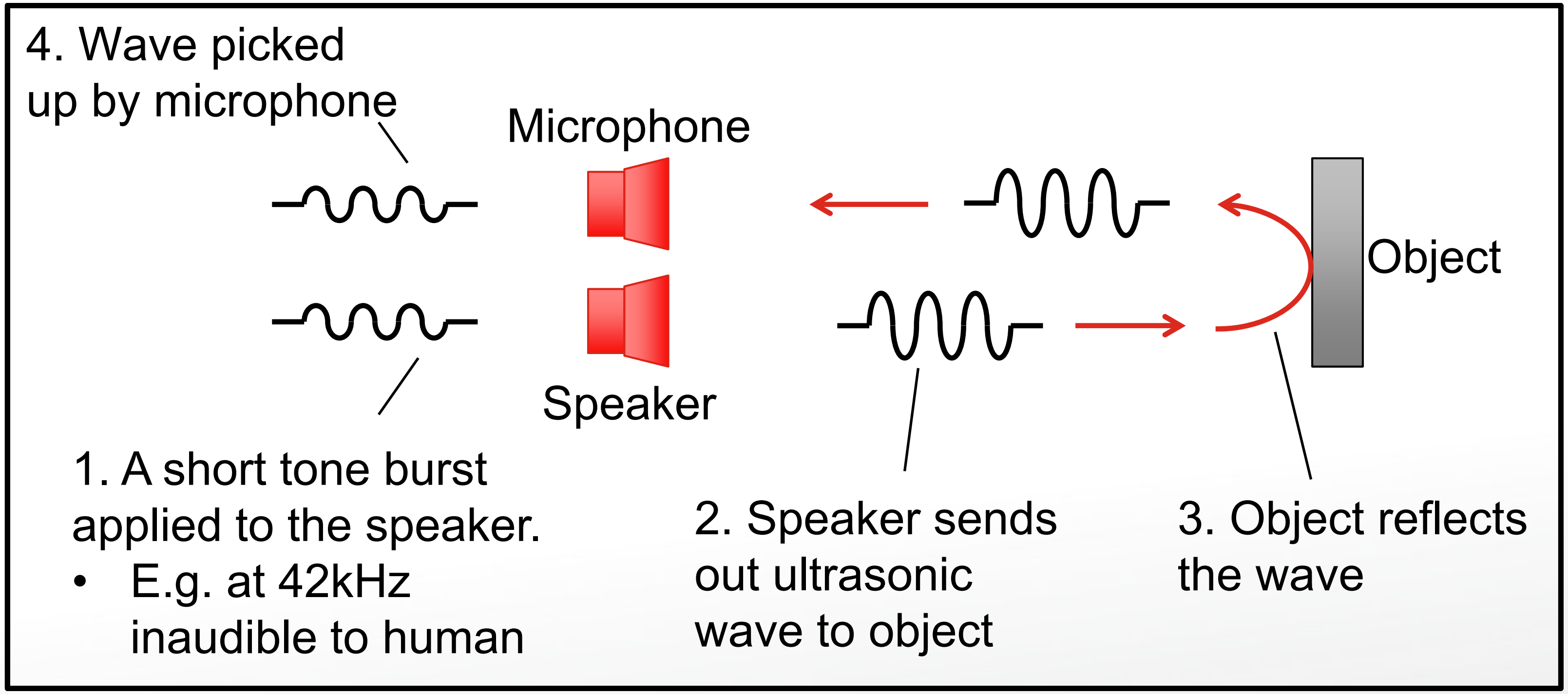
$$R = C\left(\frac{1}{V}\right) + b$$

Position Sensors (2)

- Sonic Range Finder



Sonar Sensor
https://commons.wikimedia.org/wiki/File:HC_SR04_Ultrasonic_sensor_1480322_3_4_HDR_Enhancer.jpg



- Distance is calculated based on time of flight.

$$R = v \times \frac{t_{tof}}{2}$$

V = speed of sound in air (345m/s at 22.5°C)

$$t_{tof} = t_{send} + t_{receive}$$

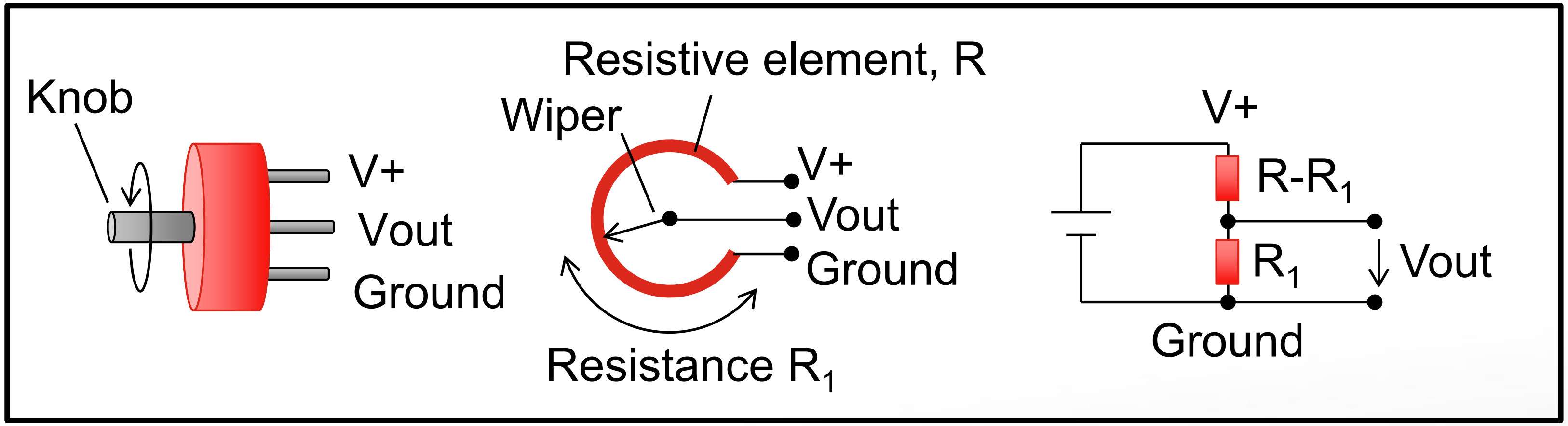
RMIT UNIVERSITY

Position Sensors (3)

Potentiometer

https://en.wikipedia.org/wiki/Potentiometer#/media/File:Potentiometer.jpg

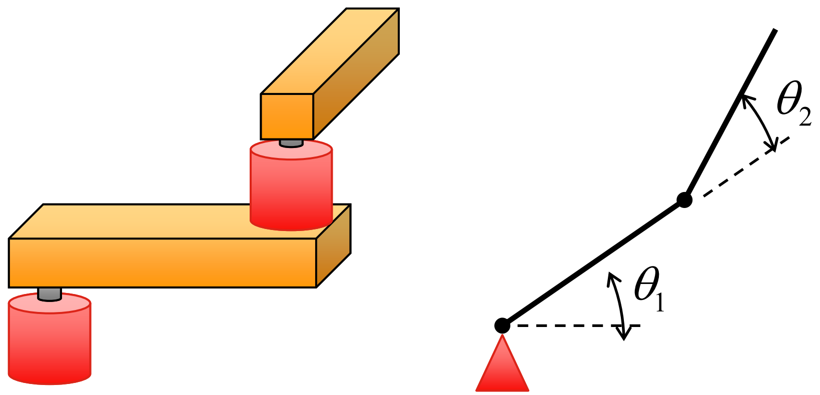
- Potentiometer
  - Variable resistance device.
  - Can be used for measurement of angular position.



  - The contact point between "wiper" and resistive element changes when the knob is turned.
  - The resistance between wiper and ground, R_1, changes in proportion to the angular displacement.
  - Output voltage: $V_{out} = \frac{R_1}{R} V_+$ proportional to angular displacement.

Usage & Limitation of Potentiometer

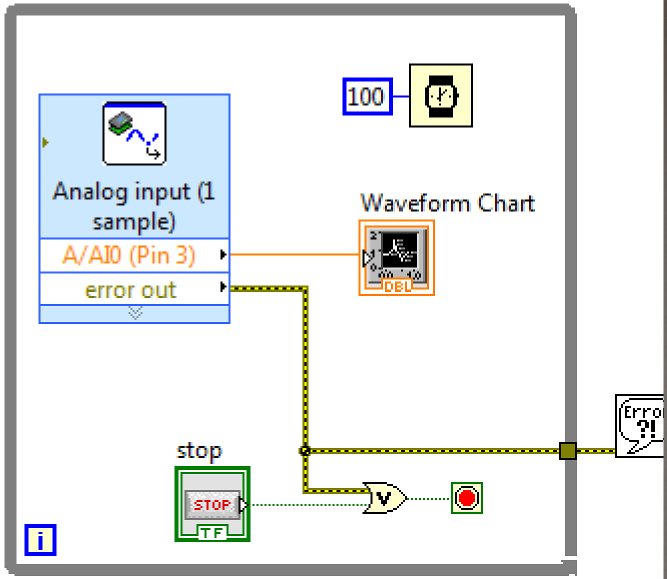
- Potentiometer works fine for angular position measurement. E.g. angular position of robotic arms.

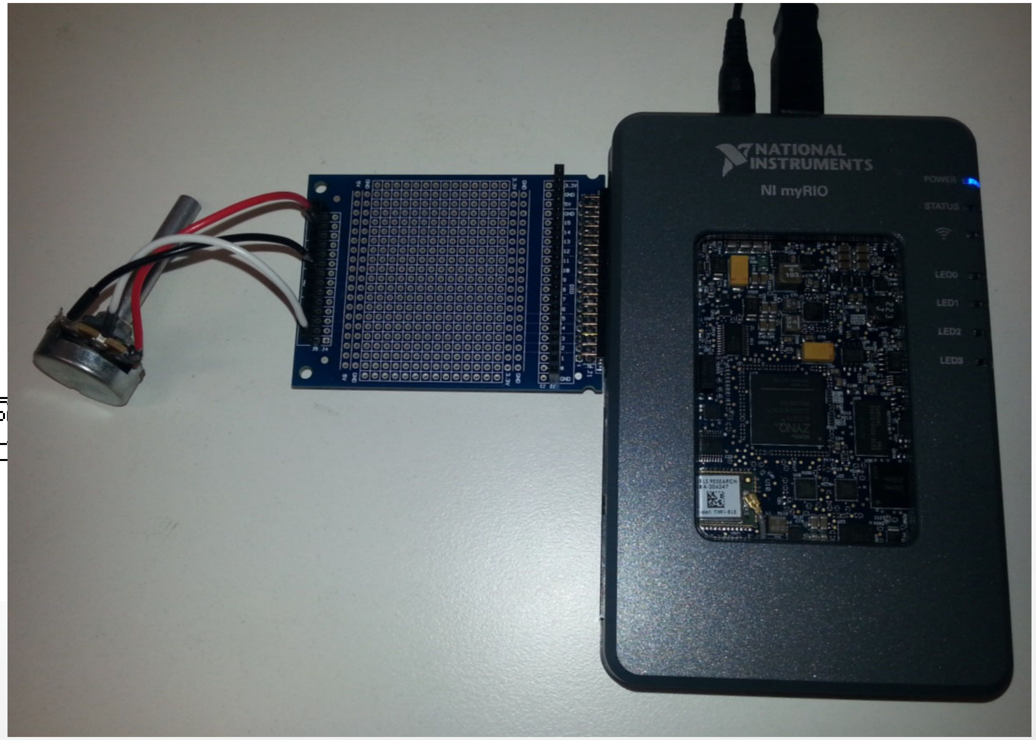


- Disadvantage: Limited measurement range, e.g. -180° to 180° only.
    - Thus not suitable for measurement of angle of a continuously rotating motor.

Use myRIO to read Potentiometer

- As potentiometer is also an analog device, the code for reading in potentiometer is exactly the same as that of range sensor.

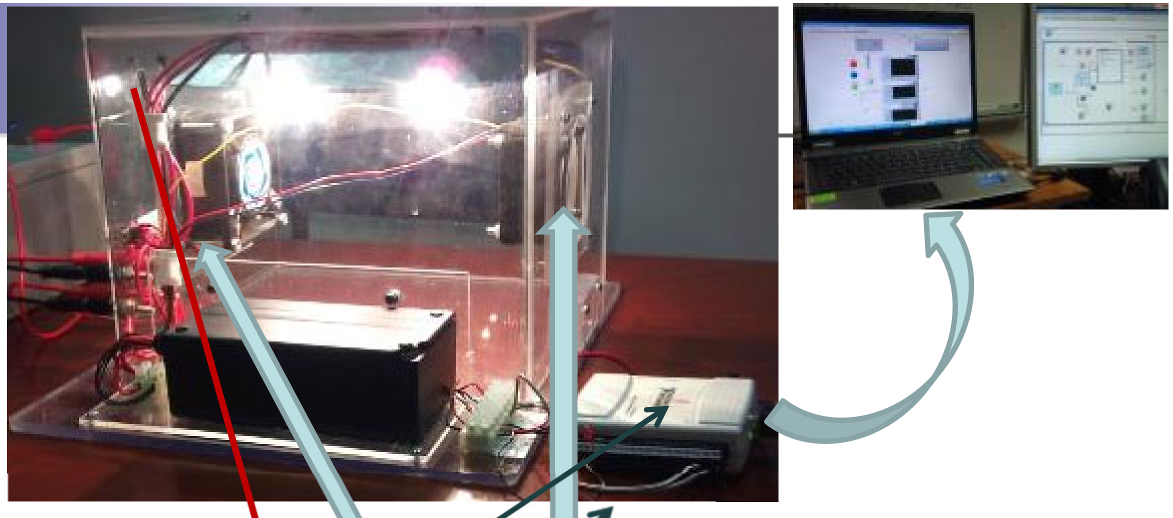




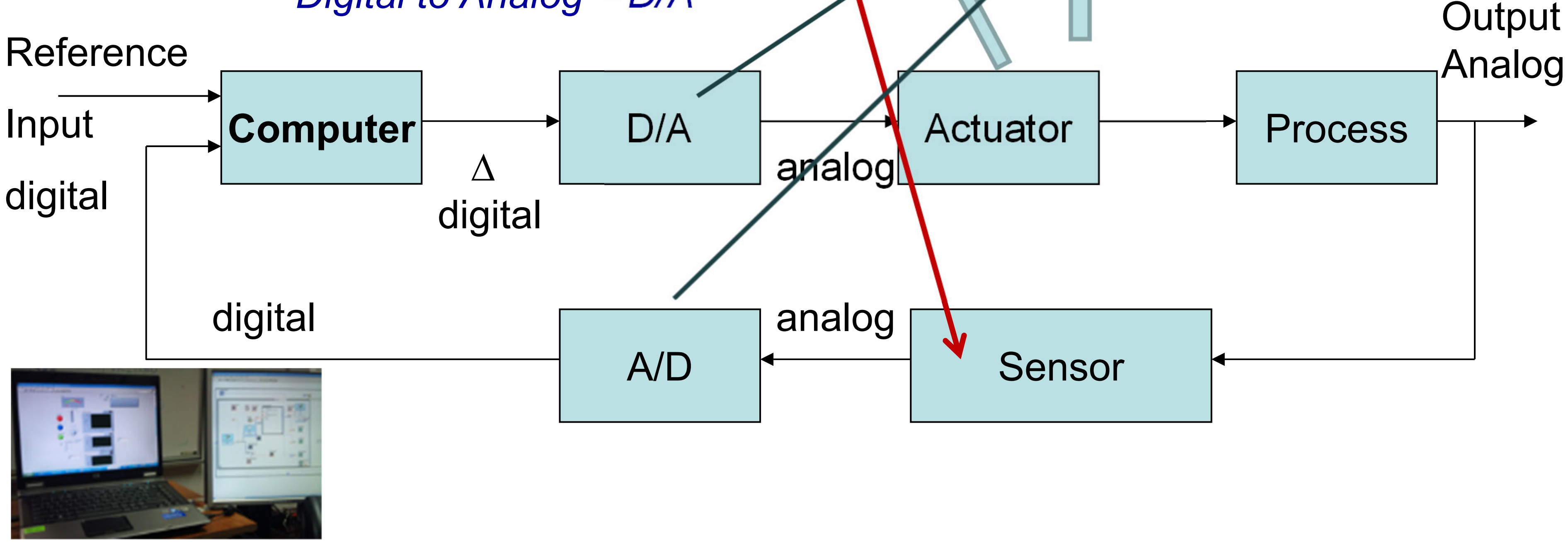
- Connect the Potentiometer to myRIO board as follows:
  - Red → 5V
  - Black → Ground
  - White → AI 0

Digital Control Systems

- Computer Control,
- Data Acquisition (DAQ)
- Signal Conversion:
  - *Analog to Digital – A/D,*
  - *Digital to Analog – D/A*

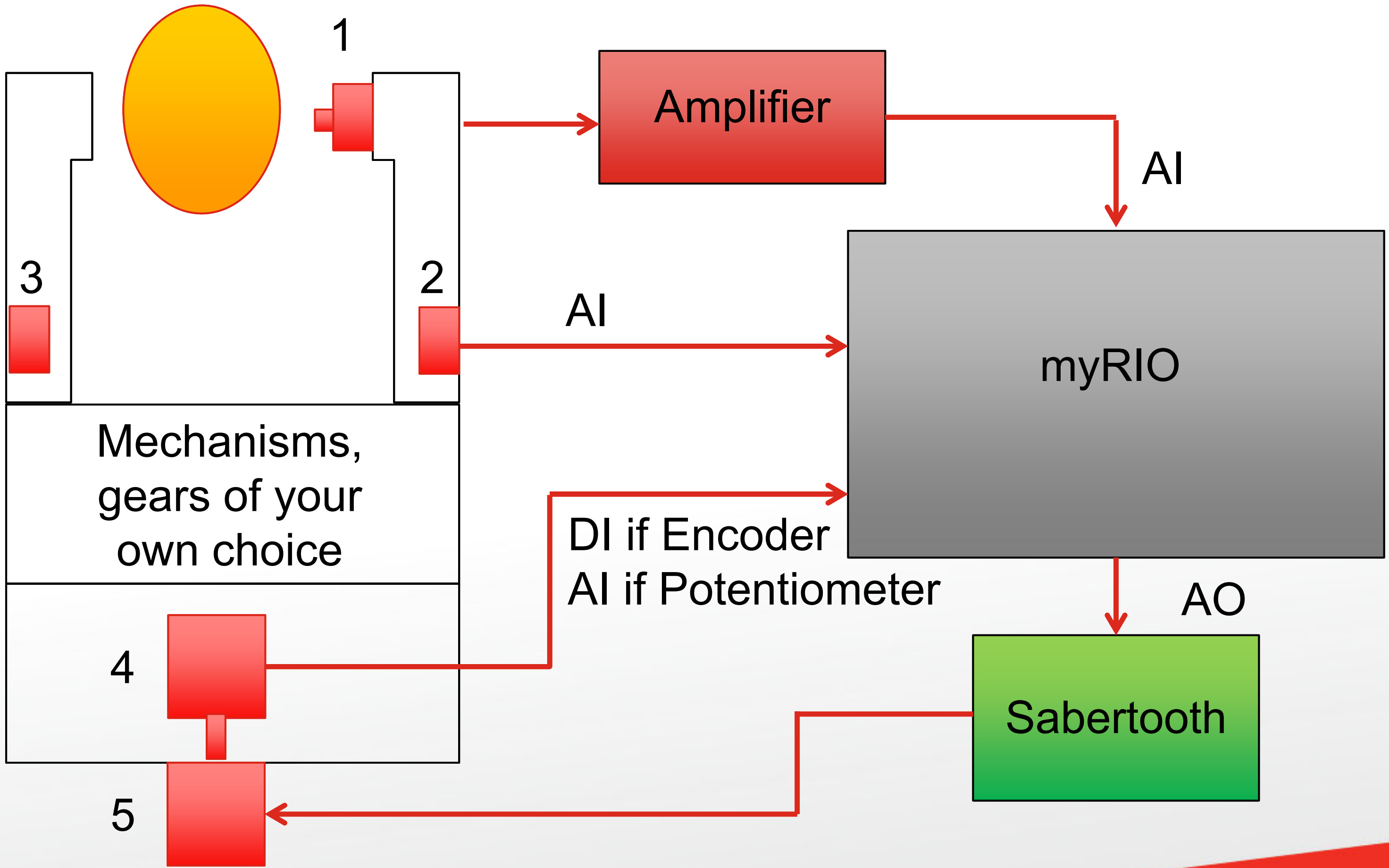


Temperature Control

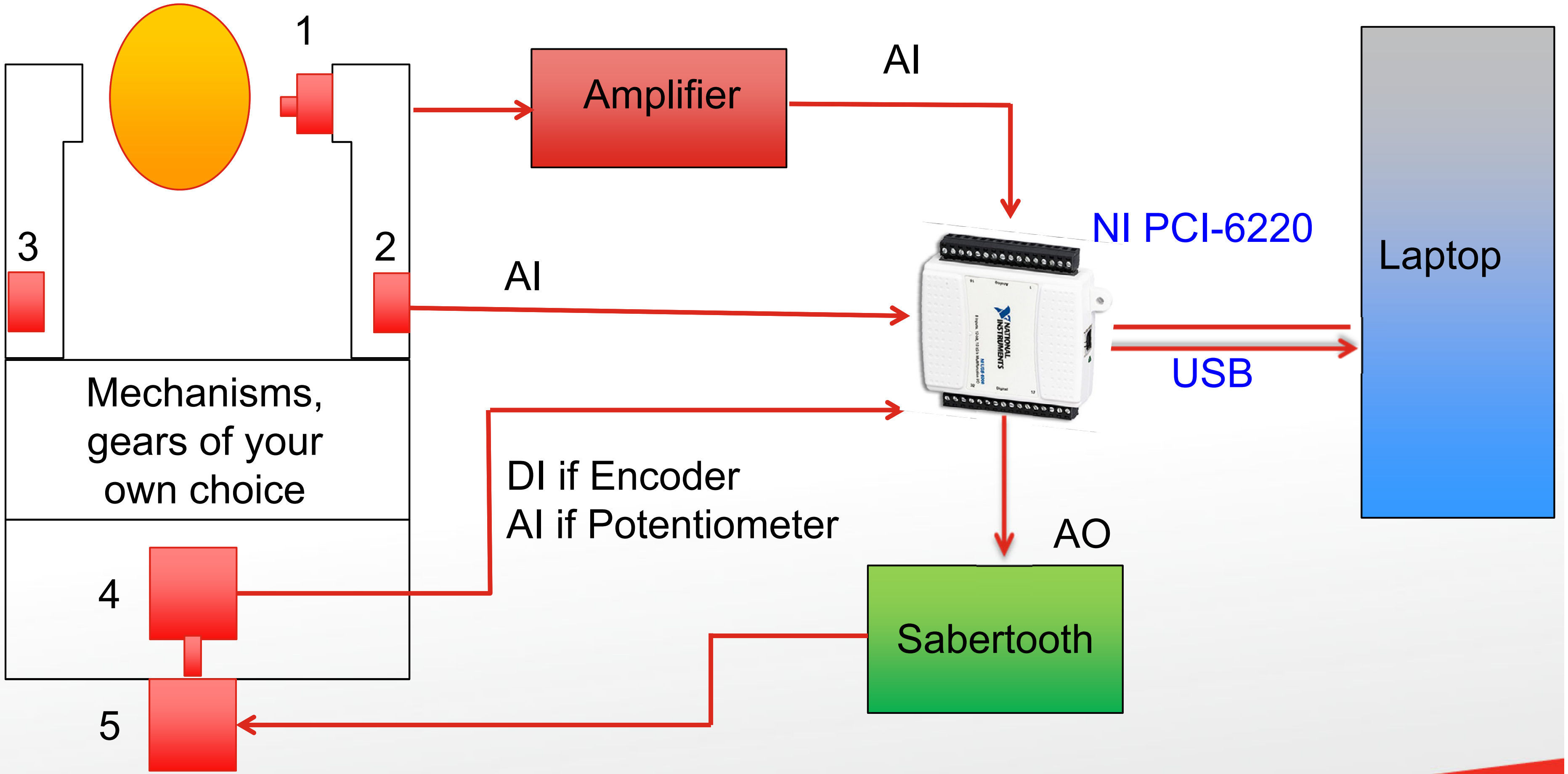


Gripper Project with myRIO

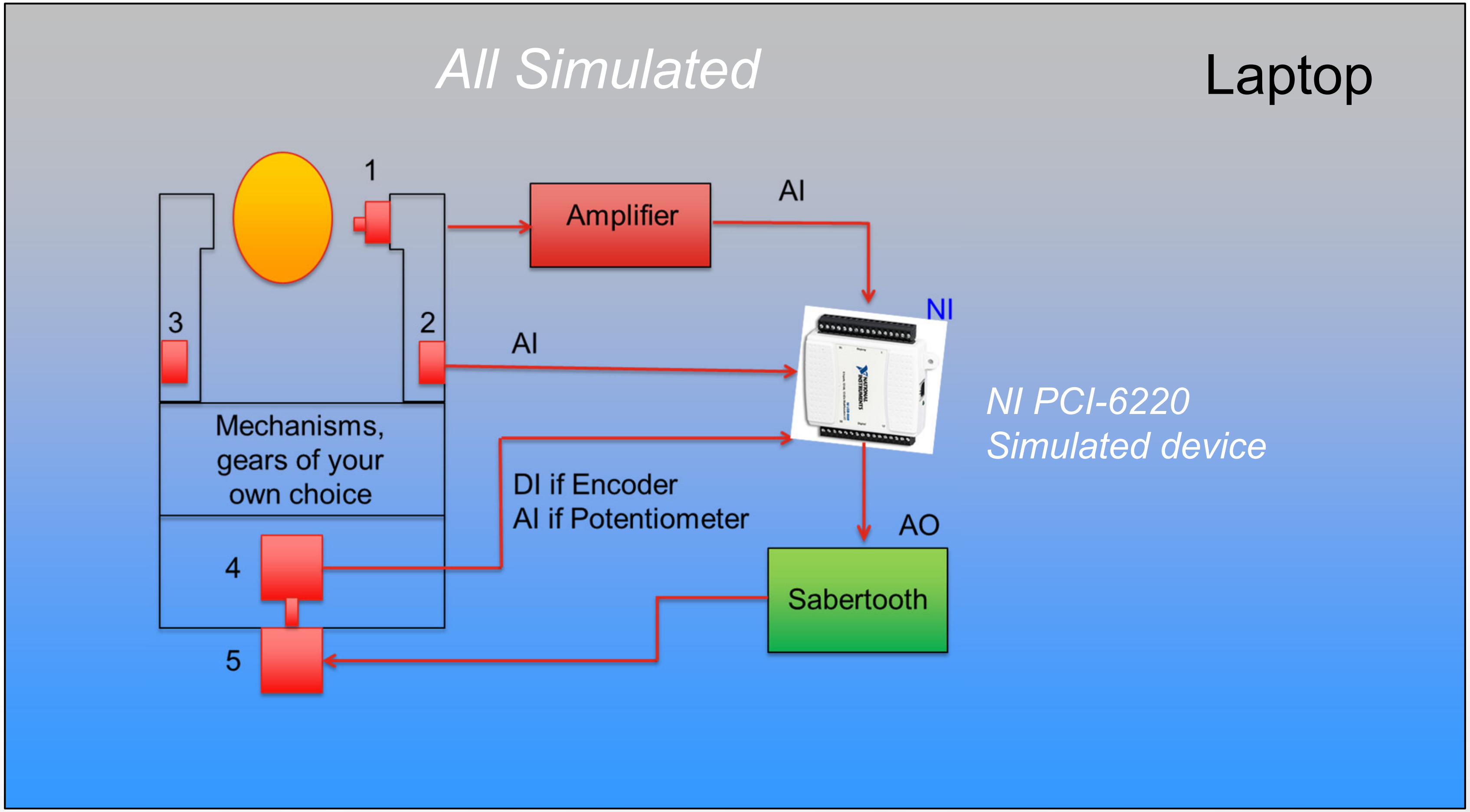
- Connections:



Gripper Project with NI USB DAQ



Gripper Simulated



RMIT UNIVERSITY

Content

- Introduction
- Touch Sensors and Switches
    - Measure using myRIO
- Proximity Sensing
    - Measure using myRIO
- Position Measurement
    - Measure using myRIO
- Attachment: Writing FPGA Codes

RMIT UNIVERSITY

Attachment: Writing FPGA Codes for myRIO

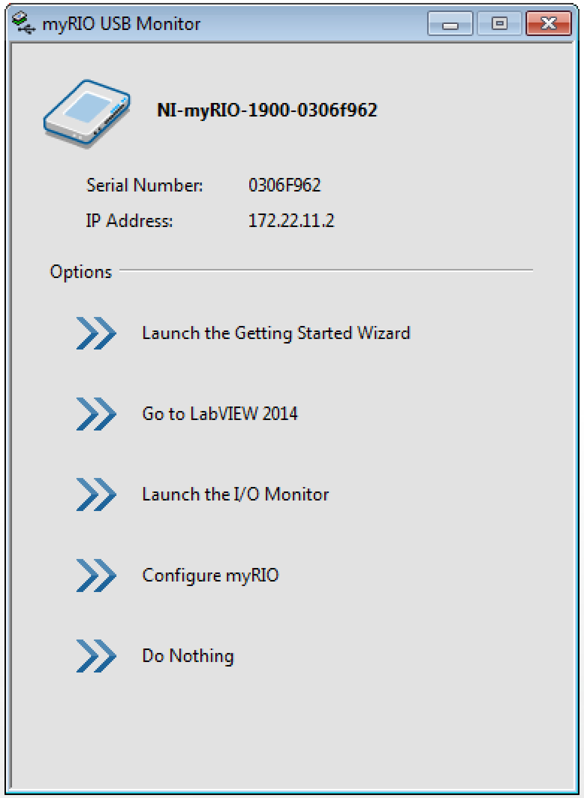
- We do not have time to cover this during our lectorial hours.
- However, I strongly encourage you to try this at your free time.
- The steps-by-steps instruction are quite clear.
- You are also welcome to ask me questions if you face difficulties.

Writing FPGA Codes for myRIO

- We already saw that myRIO has some readily-available FPGA "Personality", which allows you to read in and send out signals.
- Nevertheless, it is worthwhile to learn how to write the FPGA codes by ourselves, for a few reasons:
  - You can create codes for your own intended applications.
  - Other RIO devices such as compact RIO might not have the default FPGA personality.
    - Learning how to code FPGA will prepare you for your future use, in case you have to use these devices.

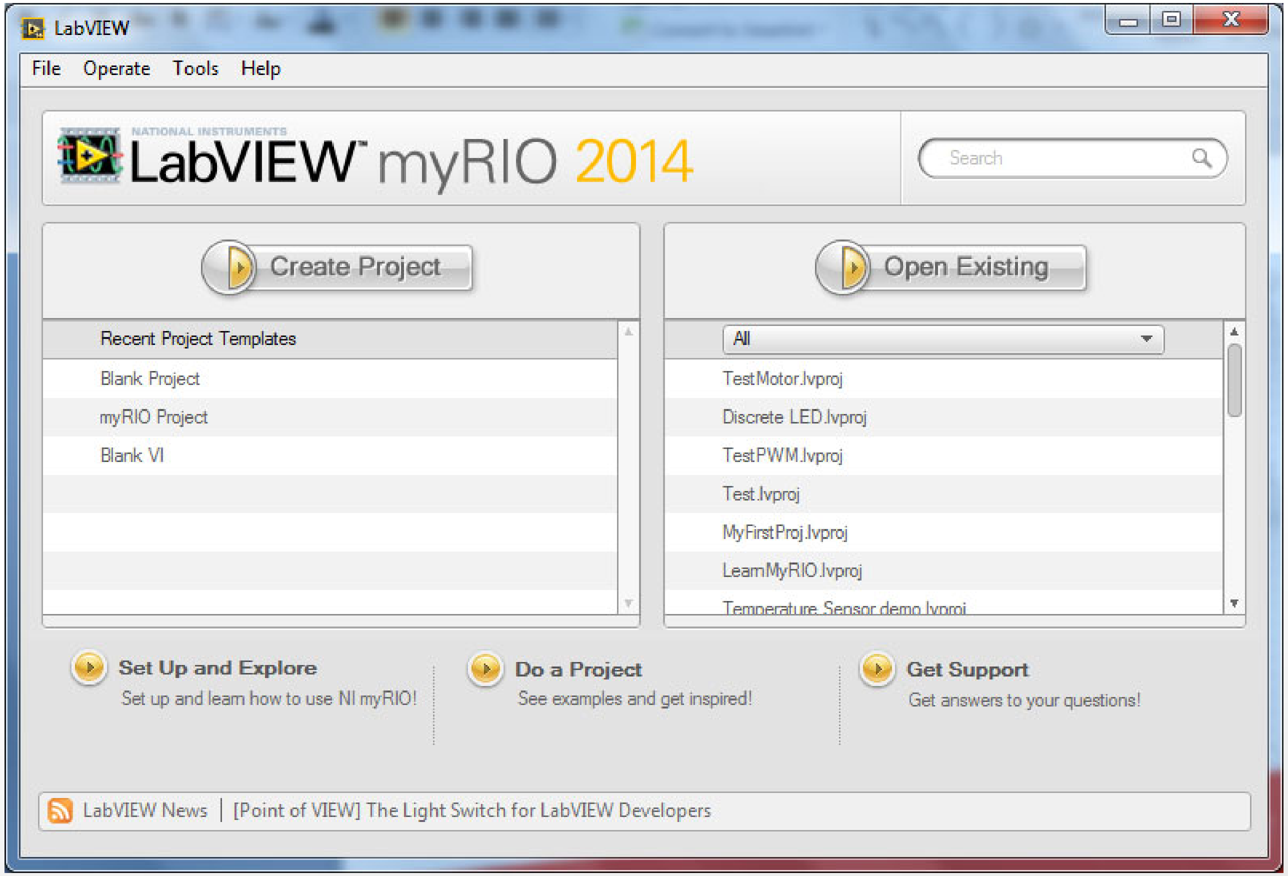
Setting up FPGA Project

- Setting up an FPGA project is similar to setting up an RT project, only with one more additional step.
- Let's start from scratch.
- Unplug and plug-in myRIO device again.
- On the pop-up menu, choose "Go to LabVIEW 201x".



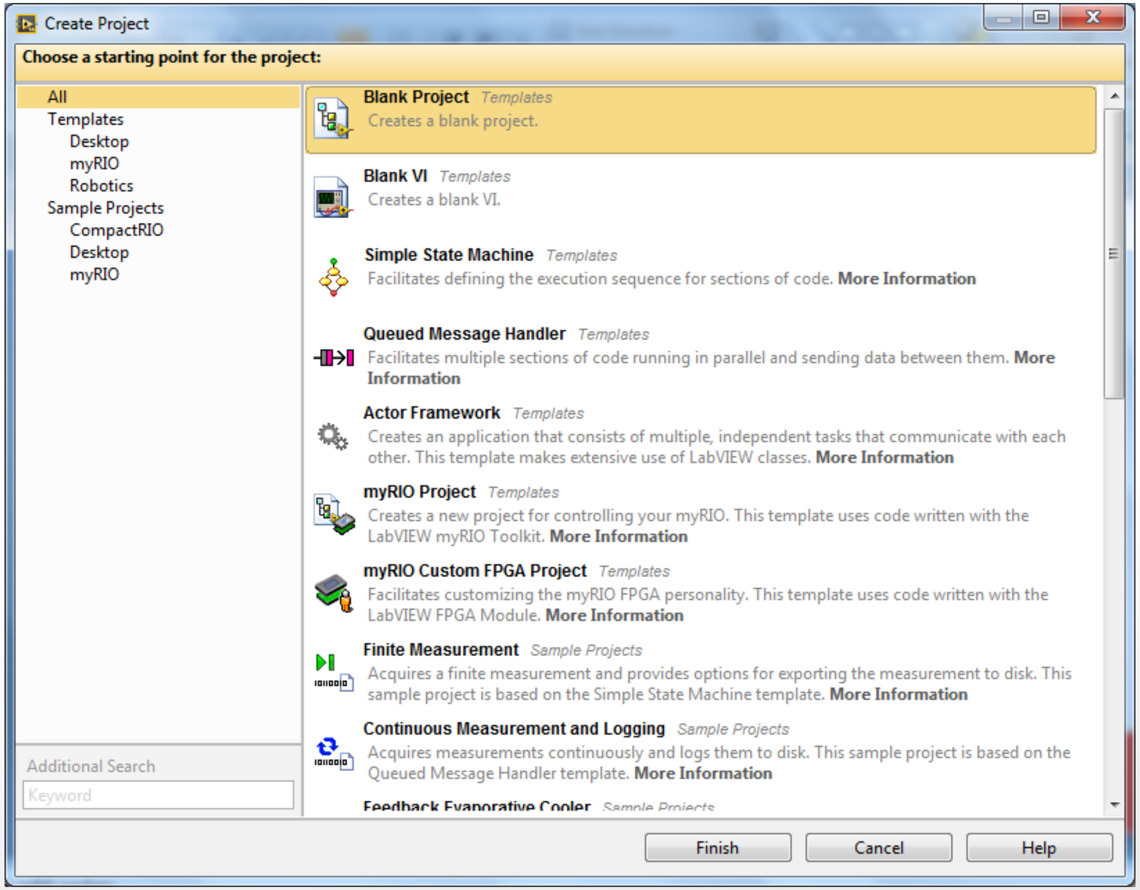
Setting up FPGA Project

- Click on “Create Project”



Setting up FPGA Project

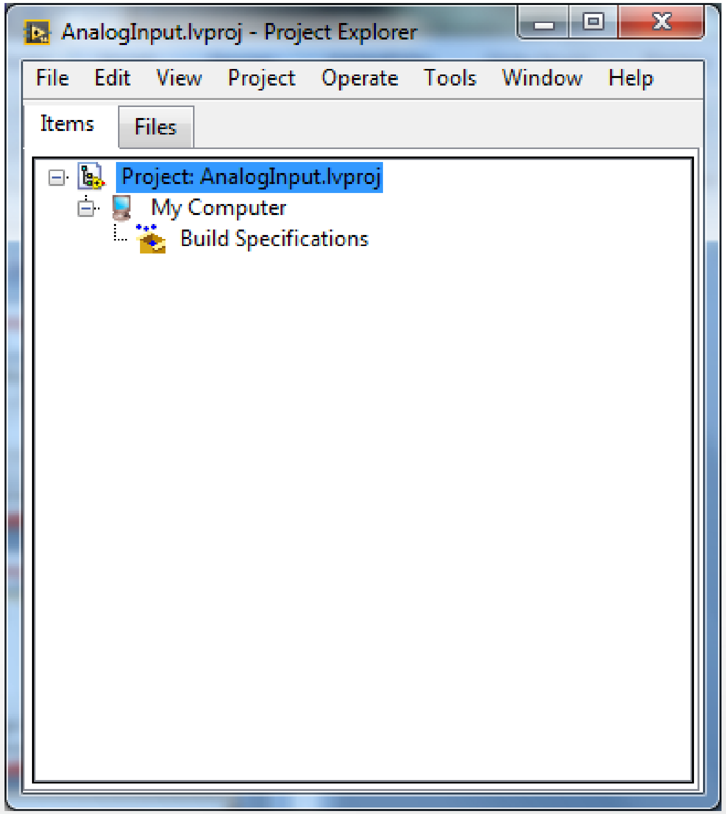
- Choose blank project.



RMIT UNIVERSITY

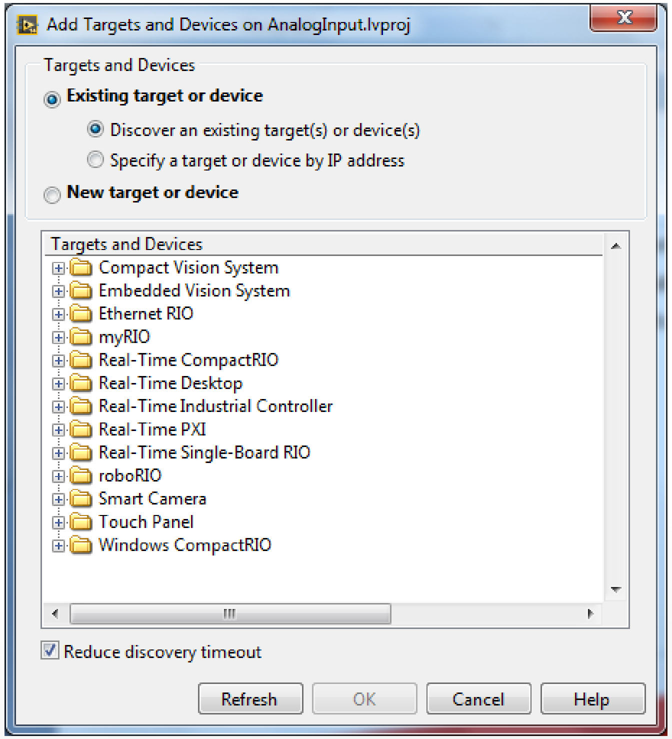
Setting up FPGA Project

- You will see the project window.
- Right click on “Project: Untitled Project 1” → Save as → “AnalogInput”.



Setting up FPGA Project

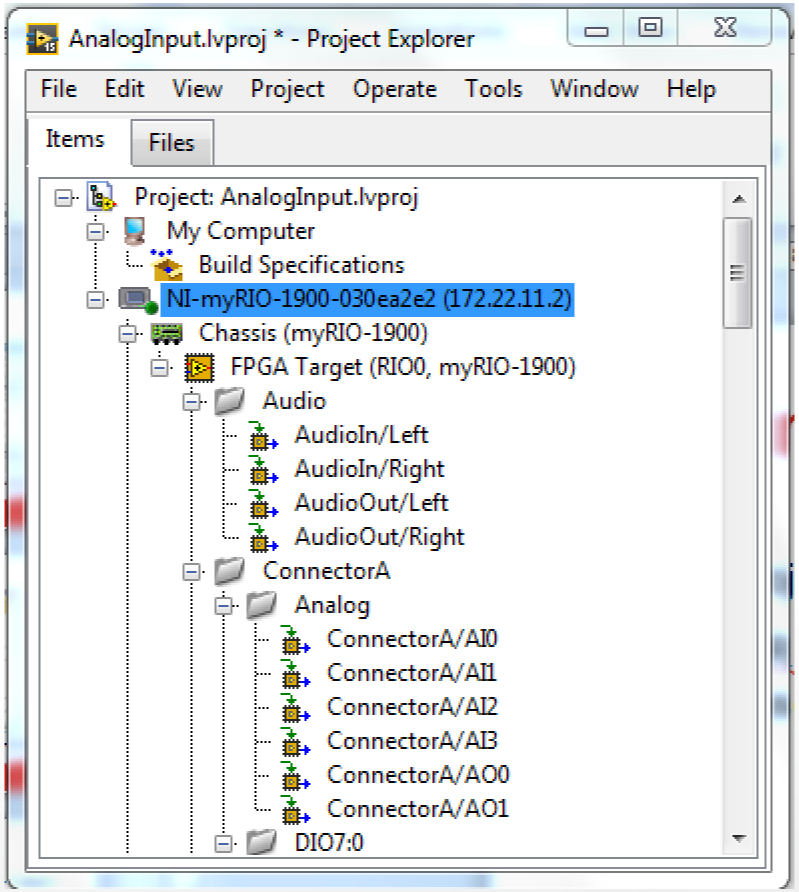
- Right click on Project AnalogInput → New → Targets and Devices



- Existing target or device:
  - Discover an existing target or device.
  - Or specify a target or device by IP address. (myRIO's IP is 172.22.11.2)
- Click the "+" sign before myRIO.

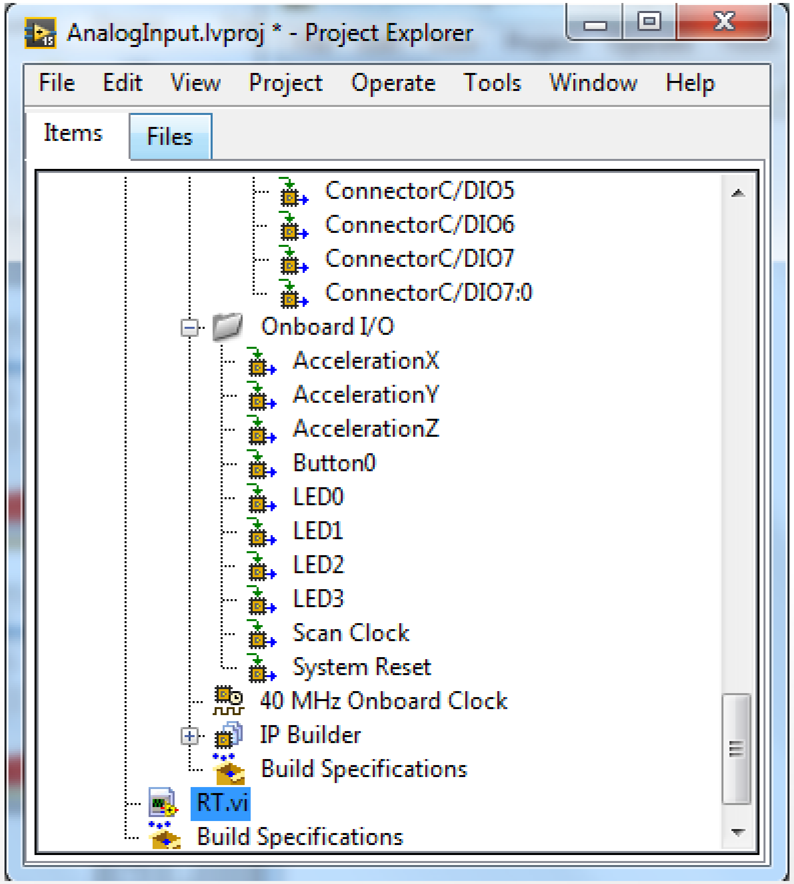
Setting up FPGA Project

- The project tree now looks like this:



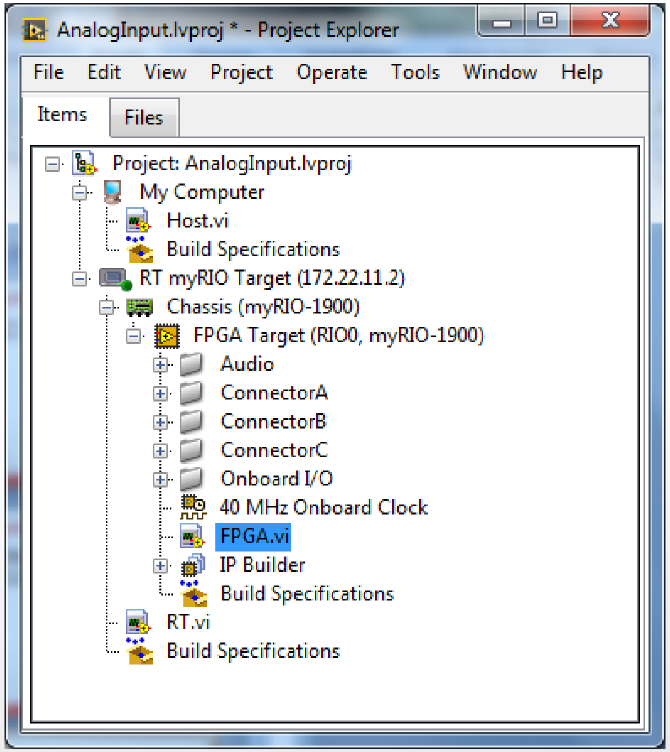
Setting up FPGA Project

- Right click on "NI-myRIO-1900…" → New → VI
- A VI will open.
- Save it as RT.vi.



Setting up FPGA Project

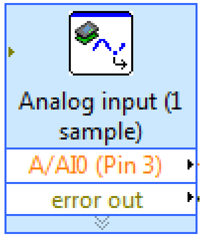
- Now, one more step to have a program running from FPGA.
- Right click on “FPGA Target” → new VI → Save as “FPGA.vi”
- The project tree now looks like this:



Project With/out FPGA

- The difference between projects with or without FPGA is as follows:

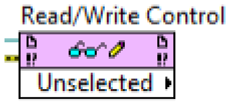
- Without FPGA:
- In RT.vi, we use sub-vi's called "Default FPGA Personality".



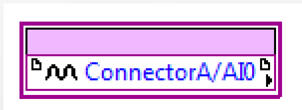
- You do not have FPGA.vi, but the sub-vi above is already pre-programmed in background using FPGA.

FPGA Code not accessible

- With FPGA:
- In RT.vi, we use "FPGA Interface → Read Write Control" as bridge to the FPGA level.



- In FPGA.vi, we write the codes and then use "FPGA IO" as bridge to the RT level.

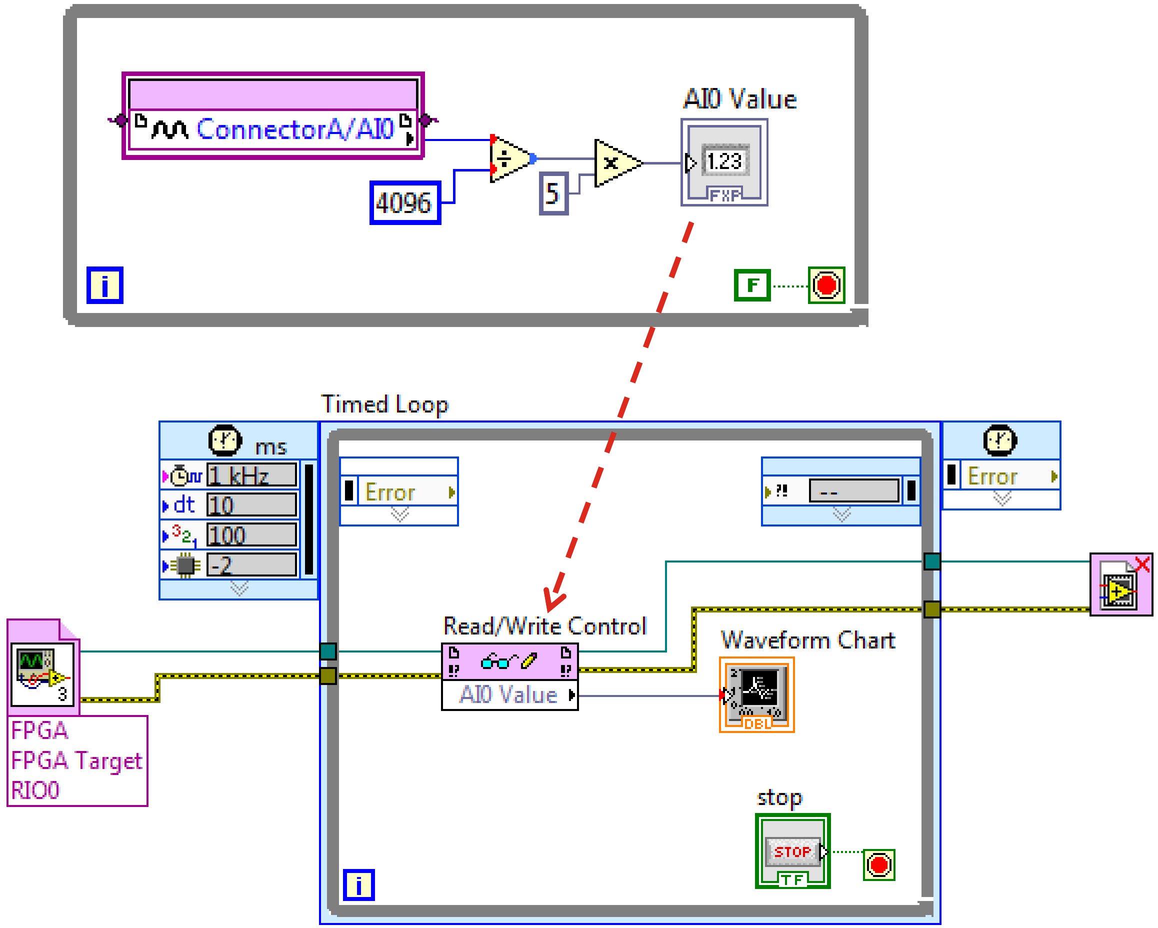


Project With/out FPGA

- We will go through the details later, but here is a feel of what was mentioned in the last slide:

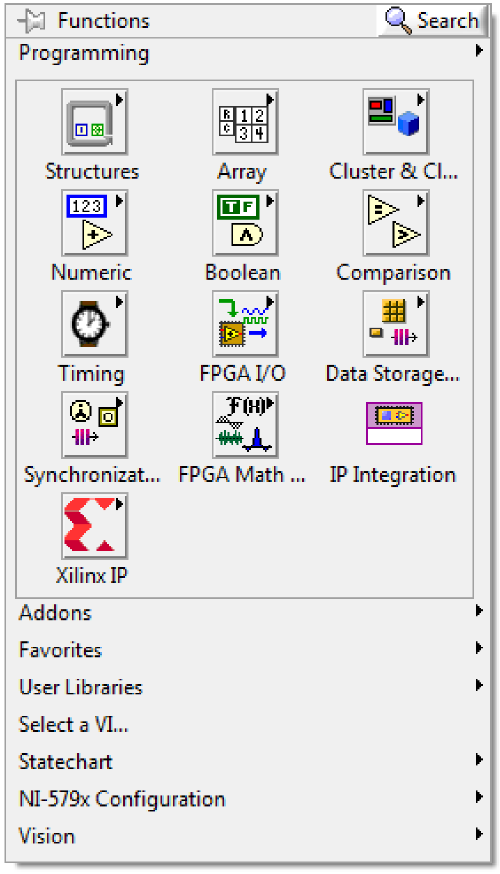
FPGA.vi – Signal from outside world is read through AI0, and FPGA code is written to process the signal.

RT.vi – The calculated "AI0 Value" in FPGA.vi is linked to RT.vi through Read/Write Control, for further use (e.g. display)



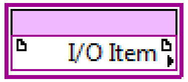
Analog Input - FPGA

- Now let's write the source code in FPGA to read in analog signals.
- Open up FPGA.vi.
- When you pop-up on the block diagram, you will see less functions in some of the categories as compared to RT and Host.
- However, there are also some additional functions such as FPGA Math & Analysis.

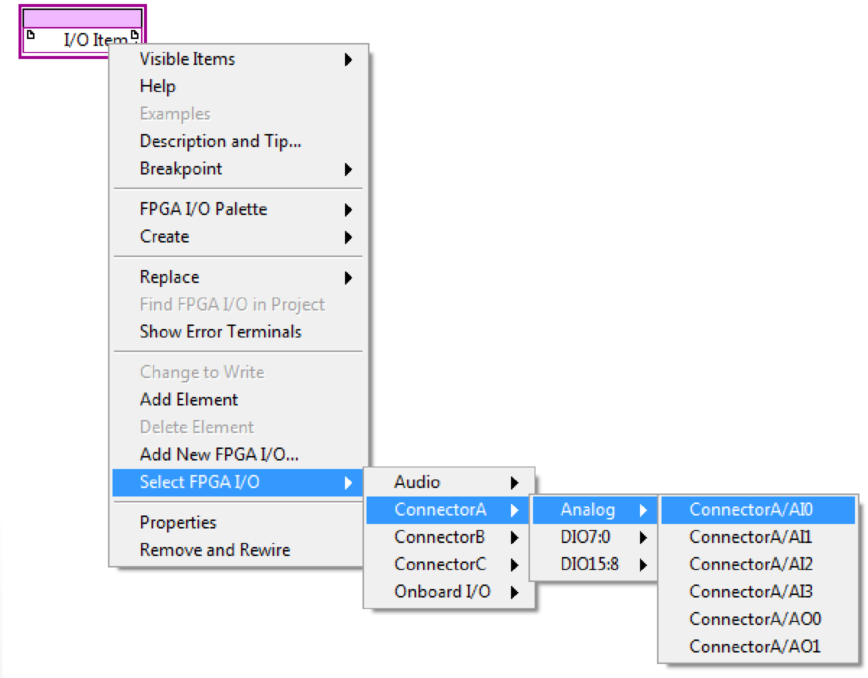


RMIT UNIVERSITY

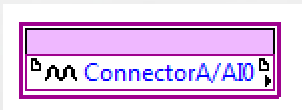
Analog Input - FPGA

- Select FPGA I/O → I/O Node. 

- Pop-up and look for ConnectorA/AI0.

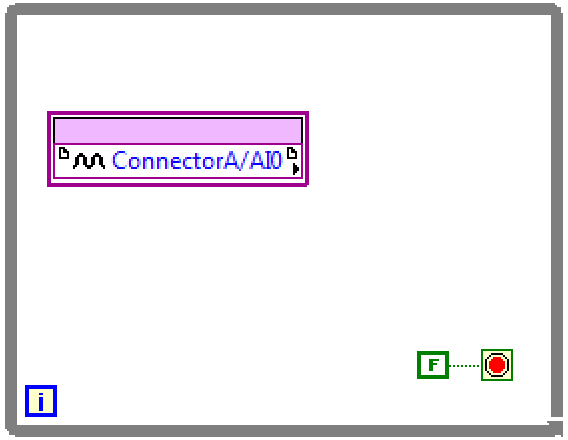


- The button will change to:



Analog Input - FPGA

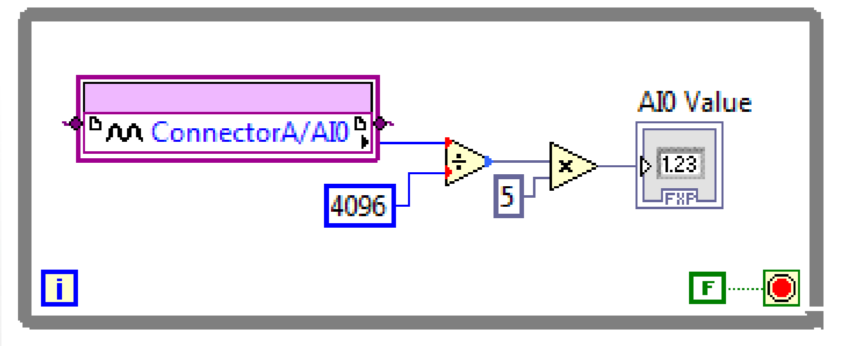
- To make it run continuously, use the while loop.



- For FPGA, we wire a constant False to the stop condition because we want the hardware to run forever.

Analog Input - FPGA

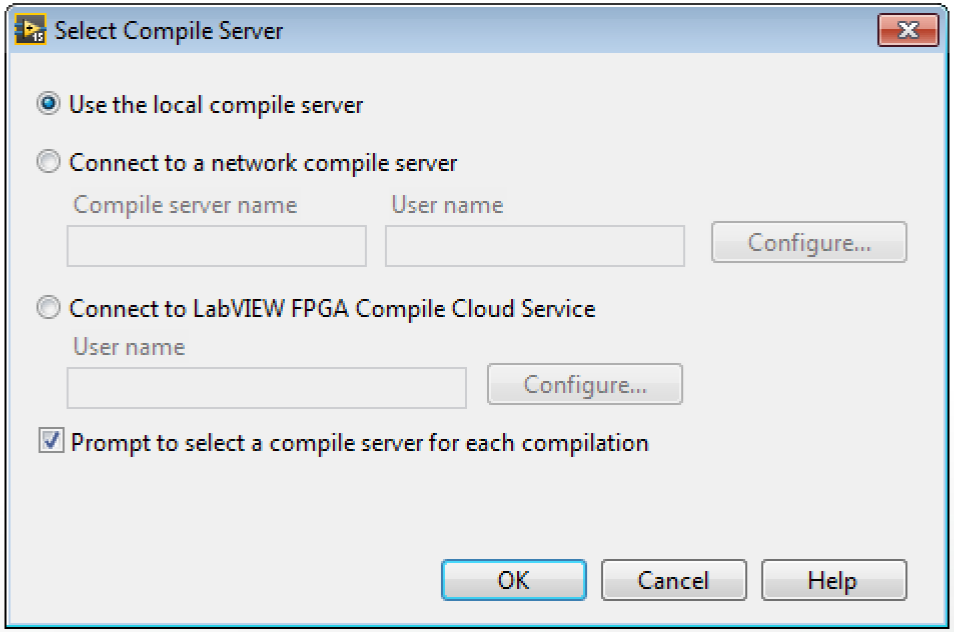
- Now, the Analog Digital Converter of MXP (A&B) has a 12-bit resolution between 0 and 5V, i.e. each step is 5V / 4096 = 1.221mV.
  - Value of 0 = 0V & Value of 4095 = 4.999V.
- Whereas the Analog Digital Converter of MSP (C) has a 12-bit resolution between -10V and +10V, i.e. each step is 20V / 4096 = 4.883mV.
  - Value of -2048 = -10V & Value of 2047 = 9.995V.
- In the example, we need to divide the value read in through the AI0 by 4096, and then multiply by 5.



- Configure AI0 Value and the output of the division to be FXP (Floating Point).
  - FPGA cannot handle DBL (Double).

Analog Input – Compile FPGA

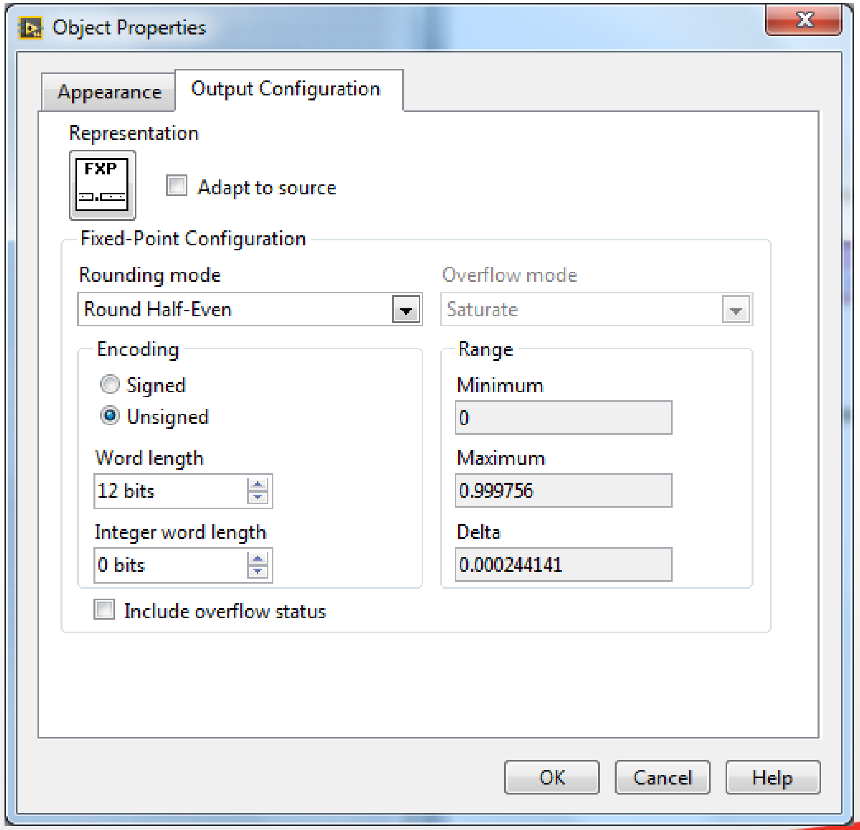
- Now click the white arrow to compile FPGA.
- You will get the following dialogue:



- Select “Use the local compile server” then click OK.
- It will take some time to complete the compilation.
  - About 10 mins for this code.

A Note on Fixed Point

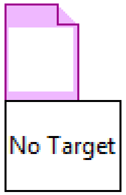
- Number that has a fixed number of digits (before and) after the decimal point.
- For complicated VI's, you may need to manually reduce the word length and integer word length so that there is enough memory to run the code.



- E.g. After division by 4096, we only expect a value of 0 to 1, and the precision will only be 1.221mv/5.
- Change the values, including "unsigned", as shown.

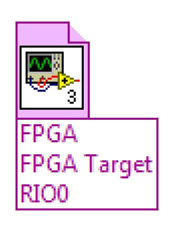
Analog Input - RT

- Now that the FPGA code is done, we want to access the data in RT.
- Open RT.vi
- Select FPGA Interface → Open FPGA VI Reference



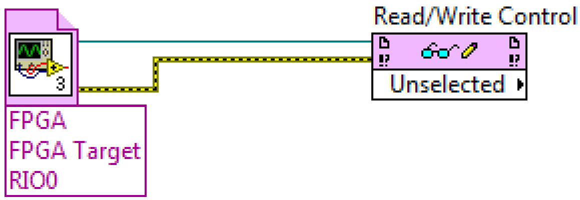
  - We use this item to "tell" RT.vi that it needs to access to signals created in "FPGA.vi"

- Drag and Drop "FPGA.vi" from project tree to the item above:



Analog Input - RT

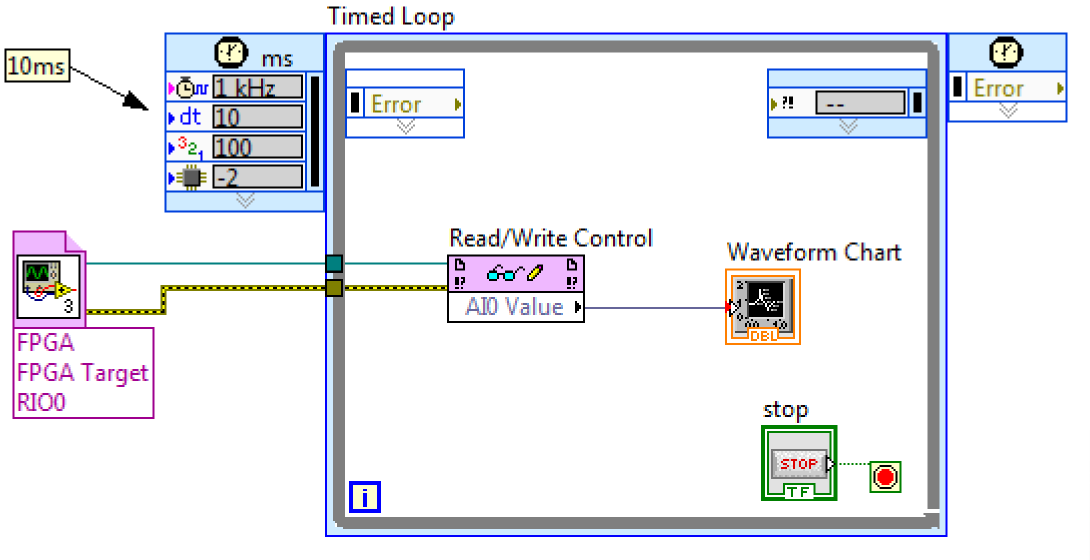
- Next bring in FPGA Interface → Read Write Control, and wire as shown:



- When you click on the right arrow on "Unselected", you will see that you have the selection "AI0 Value".
  - This is the signal which we have in FPGA.vi!

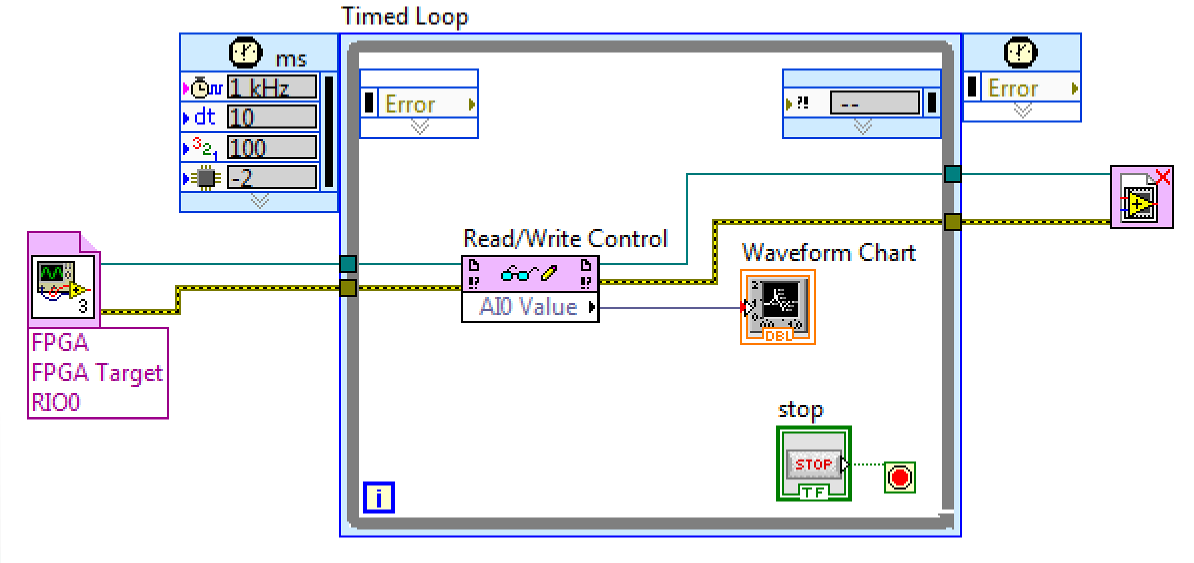
Analog Input - RT

- Continue building the VI as shown below:



Analog Input - RT

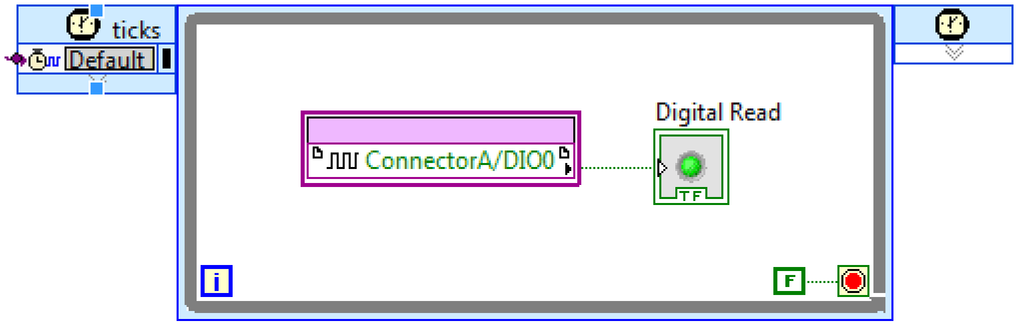
- Finally, we need to close the reference to FPGA.
- Select FPGA Interface → Close FPGA VI Reference.
- Complete the VI as shown:



- Run the VI and you will be able to see the measured signal.

Digital Input - FPGA

- Please setup your project again to include RT.vi and FPGA.vi.
  - Project name “DigitalInputFPGA”.
- Open up FPGA.vi.
- Create the following VI:

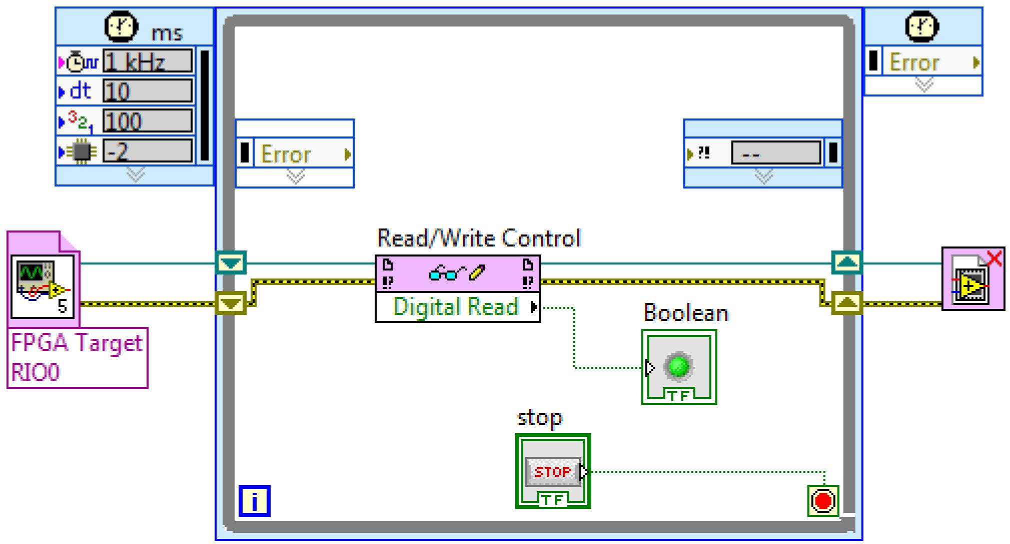


- Note: The timed loop in FPGA is also known as the “single cycle timed loop”.
  - It runs at 40MHz, and guarantees that everything within the loop is complete within one tick.
  - For the AnalogInputFPGA project, we need to stick to While loop, because Analog Digital Converter of the Analog Inputs cannot process the signal within 25ns.
- Compile the FPGA code.

RMIT UNIVERSITY

Digital Input - RT

- Create the VI as shown:



- Run the VI and you will be able to see the measured digital signal.

Thank you,
Questions