Virtual Joystick Control of Finch Robot

Prof. David R. Loker, Pennsylvania State University, Erie

David R. Loker received the M.S.E.E. degree from Syracuse University in 1986. In 1984, he joined General Electric (GE) Company, AESD, as a design engineer. In 1988, he joined the faculty at Penn State Erie, The Behrend College. In 2007, he became the Chair of the Electrical and Computer Engineering Technology Program. His research interests include wireless sensor networks, data acquisition systems, and communications systems.

Mr. Stephen A. Strom, Penn State Behrend

Stephen Strom joined the faculty of Penn State Erie, The Behrend College, and the School of Engineering in Fall 2010. He is a lecturer in the Electrical and Computer Engineering department and holds a B.S. in electrical engineering from Carnegie Mellon University. Steve comes to Penn State Behrend with more than thirty years experience in designing and programming embedded systems and has multiple patents for both hardware designs and software algorithms.

Joystick Control of Finch Robot

Abstract

Junior-level students in the Electrical and Computer Engineering Technology program complete a 3-credit Measurements & Instrumentation course. There are three main sections of the course: (1) Programming applications using LabVIEW, (2) Data acquisition, sensors, and signal conditioning, and (3) Design of measurement systems. Weekly laboratory activities mirror the lecture materials.

Part of the requirements in the course includes an end-of-semester team design project where one possible option is to design and implement software application for the Finch Robot. Students are provided LabVIEW SubVIs for all of the robot's low-level functions (audio buzzer, tri-color LED, left/right motor control, light sensors, obstacle detectors, temperature sensor, and tri-axis accelerometer values) as well as the corresponding DLL files to run the SubVIs. The objectives for the project are to utilize their LabVIEW programming skills to design a joystick control for the speed and direction of the robot, display the pitch and roll of the robot, and audibly alert the user of the presence of an obstacle in front of the robot.

This paper provides a detailed listing of the engineering requirements for the project. An example of student work is provided, along with a project assessment. Recommendations are included to help ensure student success on the project.

Introduction to the Measurements and Instrumentation Course

This is a required junior-level course for Electrical and Computer Engineering Technology students. The purpose of the course is several-fold:

- Learn principles of LabVIEW programming.
- Use LabVIEW to design software for programming PC-based data acquisition (DAQ) systems
- Understand various sensors and design signal conditioning circuits to interface the sensors to DAQ systems
- Integrate all of these components into the design of measurement systems

This course is lab intensive and utilizes LabVIEW with a data acquisition (DAQ) device as a primary vehicle for the design of measurement systems[1-3]. The course is 3 credits and consists of 2 hours of lecture and 2 hours of lab per week. The lecture content of the course is divided into three areas: Programming applications using LabVIEW (5 weeks), Data acquisition, sensors, and signal conditioning (4 weeks), and Design of measurement systems (7 weeks). LabVIEW is a graphical programming environment that allows a developer access to a wide variety of I/O and sensor interfaces, perform mathematical analysis, and link all of these operations to custom designed "control panels" or user interfaces.

The lab content of the course is designed to reinforce concepts discussed during lecture. Each lab is considered a project since it lists a series of engineering requirements and depending upon the scope of the project, requires either 2 or 3 weeks to complete. Each project is completed by a student team that consists of no more than 2 students (some students prefer to work by themselves), where students pick their team members at the beginning of the semester.

For nearly all of the projects, students are expected to work outside of the scheduled lab time in order to complete the objectives. Grading for the project consists of 60% based on meeting all of the engineering requirements, 30% based on the content of the lab report, and 10% based on spelling, grammar, and writing style. There is a 5% reduction for late lab report submittals. A listing of the projects for the course is shown below.

• Lab 1: Software-defined Calculator Project	(2 weeks)
• Lab 2: Thermocouple Project	(2 weeks)
• Lab 3: Waveform Generator Project	(2 weeks)
• Lab 4: Digital Voltmeter Project	(2 weeks)
• Lab 5: Digital Multimeter Project	(2 weeks)
• Lab 6: Temperature Control System Project	(2 weeks)

Also, a team final project is required for the course. Three weeks of scheduled lab time are provided at the end of the semester for students to work on their project. One possible option for the final project is to design and implement a software application for the Finch Robot. The purpose of this paper is to describe the details about the Finch Robot project.

Project Definition

For the Finch Robot project, students were given the task of designing a "Joystick" control for the speed and direction of a Finch Robot[4]. The resulting LabVIEW program used a (USB) serial link to the robot to send commands to the device as well as poll the unit and read back its sensor information.

This project fits within the course outcomes as it requires the students to:

- Design a measurement system using LabVIEW
- Incorporate simultaneous operations all within the program
- Perform software signal conditioning (digital averaging of the accelerometer values)
- Take a complex problem, break it into smaller operations, implement each component, and combine all operations together into an overall program

Engineering Requirements for the Project

The overall project objectives are to:

- Use LabVIEW for the design of a state machine to control the motion of the robot
- Design a joystick control to move the robot forward, back, left and right
- Play an audible beep on the robot's speaker when an obstacle is detected
- Display the pitch and roll of the robot on the screen

The students were given detailed engineering requirements for the project as shown below.

User Interface Requirements:

- Joystick control
- Indicators for displaying pitch and roll of the robot
- Audible alert for warning the user of the presence of an obstacle in front of the robot

Functional Requirements:

- Speed of the robot controlled by the joystick
- Direction of the robot controlled by the joystick

Documentation:

- VI online description
- Title information is shown on front panel
- Appropriate comments are provided on block diagram

Deliverables:

- Soft copy of lab report
- Soft copies of VI and all SubVIs

Finch Robot

The Finch Robot is a device that is tethered to the PC via a USB cable. Both power and control come across the cable. The robot has two independent drive motors that power the left/right wheels and can be run at varying speeds in both forward and reverse.

The robot also has two (photocell) light sensors, one temperature sensor, two forward IR obstacle sensors, a 3-axis accelerometer, an audio buzzer and a tri-color LED.

When plugged into a PC, its USB interface registers itself as a human interface device (HID) similar to that of a keyboard or mouse. By using HID packets, the PC can send and receive information to the robot. The low-level code consists of a dynamically linked library and a series of LabVIEW SubVIs, as shown in Figure 1, that allow the user to interface to each of the sensors as well as to the motor.

The SubVIs can be grouped into four categories: Initialization/Exit, Motion, Audio/Visual, and Sensors. Each of the SubVIs' categories, descriptions, and inputs/outputs are listed in Table 1.

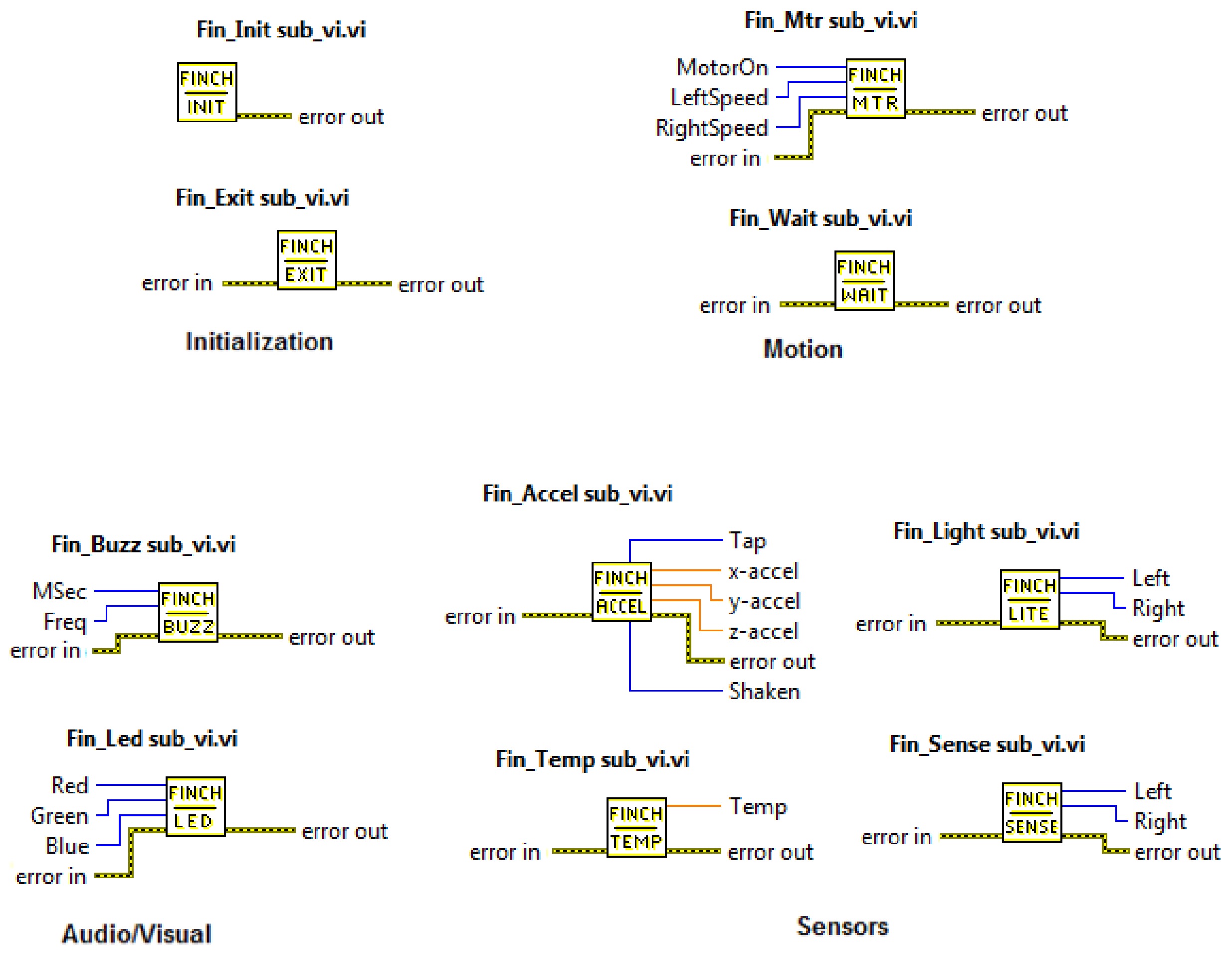

Figure 1. Finch LabVIEW SubVIs.

Category	SubVI	Description
Initialization/Exit	Fin_Init	Initialize the interface to the Finch Robot
	Fin_Exit	Terminate the interface to the Finch Robot and close the connection to the USB port
Motion	Fin_Mtr	Enable/disable the left/right motor speeds. Inputs are integers for: • Time on in tenths of a second • Left/right motor speed (values range from -255 to +255, where -255 is full speed backward, 0 is stop, and +255 is full speed forward
	Fin_Wait	Wait for the Finch Robot to stop. This is used in conjunction with the Fin_Mtr SubVI to wait for the robot to come to a complete stop.
Audio/Visual	Fin_Buzz	Enable/disable the audio buzzer. Inputs are integers for: • Time for the buzzer to remain on in mS • Frequency of the buzzer in Hz (use 0 to turn buzzer off)
	Fin_Led	Enables/disables the tri-color LED. Brightness of each color (red, green, and blue) is controlled by an integer ranging from 0 to 255, where 0 is off and 255 is full brightness.
Sensors	Fin_Accel	Reads the tri-axis accelerometer values. Outputs are: • 3 floating point numbers representing the x, y, and z axis acceleration values • 2 integers representing Boolean values (1/0) indicating whether the Finch Robot has been tapped or shaken
	Fin_Light	Reads the left/right light sensors. Each output is an integer ranging from 0 to 255, where 0 indicates no light (completely dark)
	Fin_Sense	Reads the left/right IR sensors. Each output is an integer representing a Boolean value (1/0) indicating the presence of an obstacle
	Fin_Temp	Reads the temperature sensor on the Finch Robot. Output is a floating point number representing temperature in degrees Celsius.

Table 1. SubVI Descriptions.

The Finch Robot LabVIEW SubVIs represent the basic operations of the robot and must be included into a larger LabVIEW program to perform full control and display operations. A typical LabVIEW program, shown in Figure 2, has multiple SubVIs chained together using the error in and error out terminals. Thus, as one SubVI completes its operation, the next one begins.

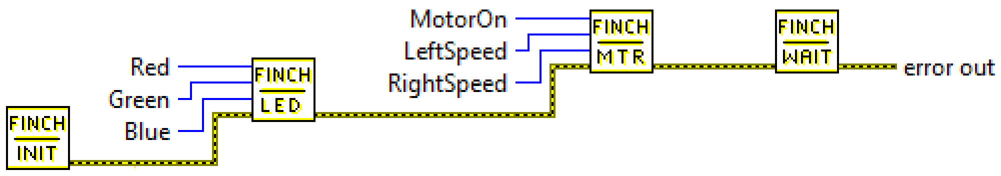

Figure 2. Linking One Finch Robot SubVI to Another.

Project Implementation

The project includes several programming and design aspects. The first step was to separate the project into smaller operations, where each operation was addressed via its own LabVIEW SubVIs. The students came to the conclusion that they needed the following operations:

- Initialization operation
- Joystick controller operation
- Motor speed operation
- Obstacle detector operation
- Accelerometer operation

Since a state machine was required, each state was assigned to represent one of the above operations. Each state would have its set of controls and indicators, and students needed to determine which SubVIs to use in each state. After completing each state, students determined how to merge the states together to design the full program. Lastly, testing was performed on the completed VI.

LabVIEW Block Diagram

Each state of the state machine is described below, along with its function. Screen shots of each state were obtained from a student's LabVIEW program.

Initialization State

The initialization state, shown in Figure 3, is a "one-time" call to the SubVIs that connects the program to the Finch Robot and joystick. While the Finch-Init SubVI is required, setting the motor speed to zero is optional. Concurrent with this, the program is connecting to the joystick and setting the initial values for the SubVIs in the next state.

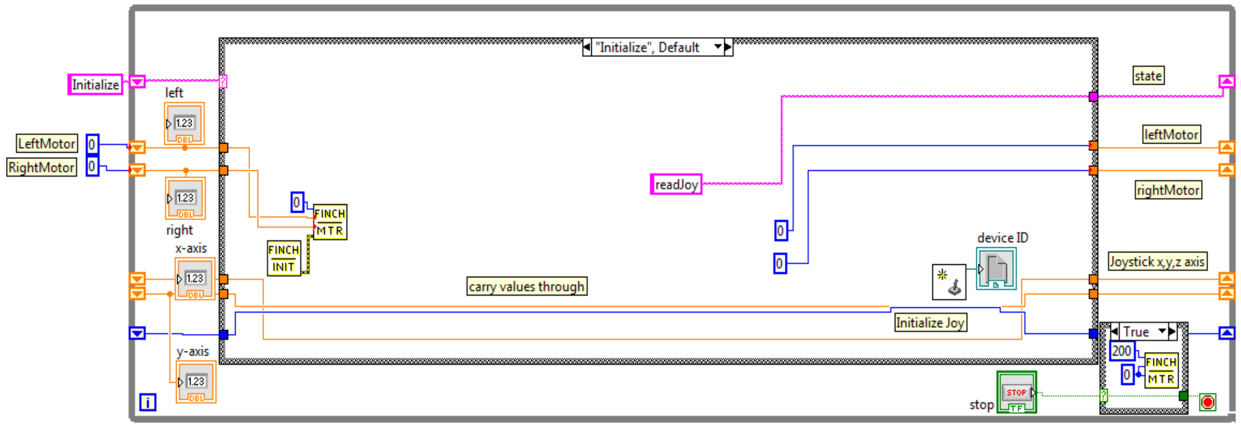

Figure 3. LabVIEW Initialization State.

The joystick control moved the robot forward, back, left and right and at varying speeds. This portion required multiple iterations in that the students had to solve several problems:

- Read in the position of the joystick using the X- and Y-axis from the controller and convert its input range to the values expected by the Finch Robot SubVIs.

- The Finch Robot can accept any input speed from -255 to +255. However, some speed settings are invalid as the robot does not have enough "traction" to implement a full speed turn. As a result, it was necessary to limit the speed settings that were sent to the robot.

- When the joystick is pushed completely forward, it returns a value of +32767, when pulled completely back, it returns a value of -32768, and when released, it reads 0. The students found that when the joystick was released, the X/Y values did not go to exactly zero. Thus, it was necessary to add a "deadband" zone around the zero point.

- If an obstacle was detected, the program needed to halt the motion of the robot. Thus, both the set motor speed and the obstacle detection operations had to work with each other.

The solution was implemented via two states: read joystick and set motor speed.

Read Joystick State and Set Motor Speed State

The joystick returns its position as a range of values from +32767 to -32768. This will get converted to a motor speed of +255 to -255 (which is in the range of the Finch motor SubVI). In addition, the X/Y values from the joystick are used to determine if the robot is to move straight or turn. In Figure 4, several examples are shown to illustrate the joystick axis values which are read and converted to left/right motor speed values.

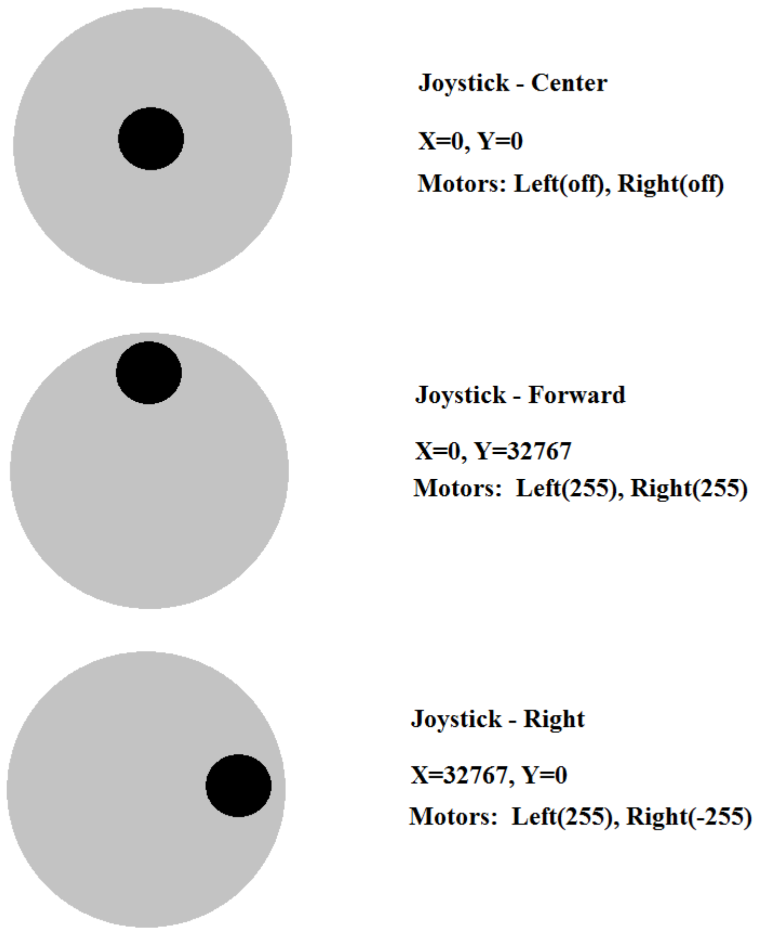
Joystick - Center
X=0, Y=0
Motors: Left(off), Right(off)
Joystick - Forward
X=0, Y=32767
Motors: Left(255), Right(255)
Joystick - Right
X=32767, Y=0
Motors: Left(255), Right(-255)

Figure 4. Joystick Axis Values Converted to Motor Speeds.

Once calculated, the motor speed settings are then sent to the detect state where they can be passed onto the motor or blocked (if an obstacle is detected). In Figures 5 and 6, the joystick data is converted/scaled, checked against a deadband, and limited to the range expected by the Finch Robot.

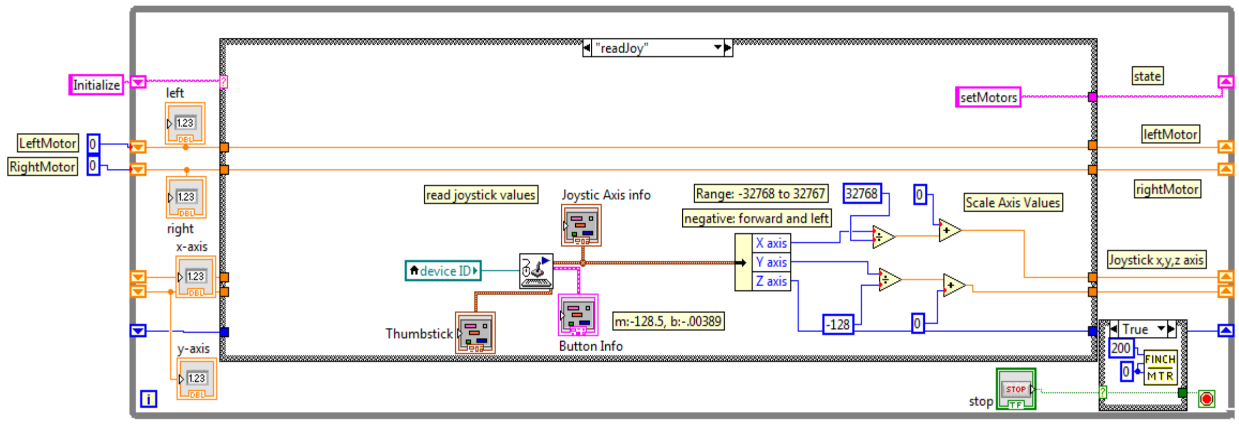

Figure 5. LabVIEW Read Joystick State.

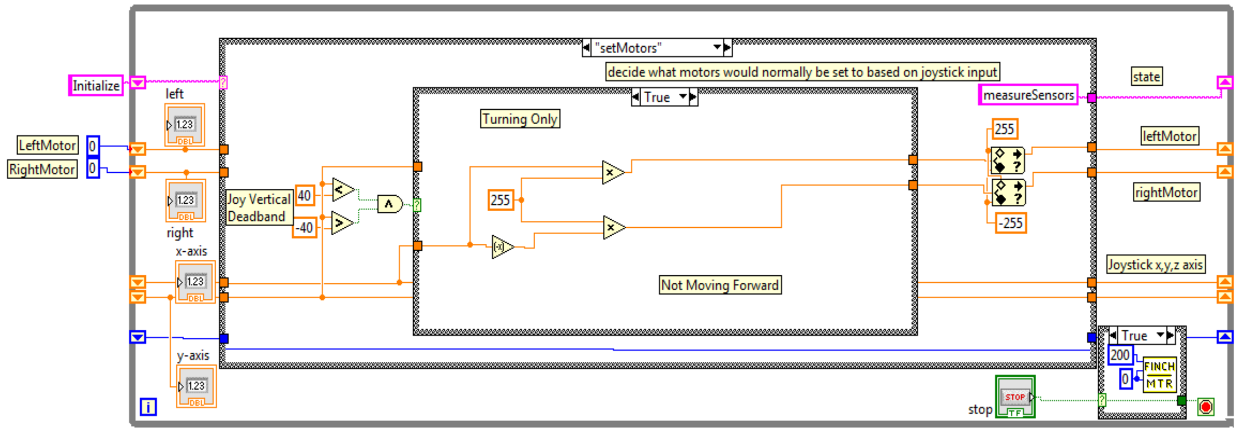

Figure 6. LabVIEW Set Motor Speed State.

Detect State

When an obstacle is detected, an audible beep on the robot’s speaker is played. This integrated the forward IR sensors as well as the motor speed settings from the previous state and either enabled the motion of the robot, or stopped the robot and played an audible tone. Once the obstacle is removed, the tone is disabled and the robot is allowed to move.

This state needed to "or" the values from the two forward obstacle sensors and set a flag to True/False (indicating if an object is present). Depending on this flag, it would send the motor speed calculated in the previous state to the robot or send the motor a speed setting of zero (to stop the robot). This is shown in Figure 7.

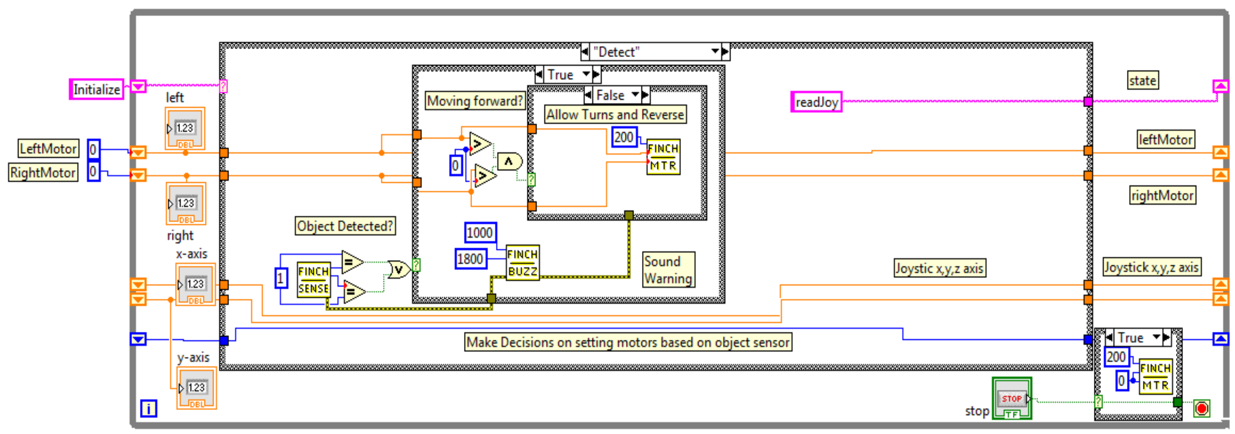

Figure 7. LabVIEW Detect State.

Measure Sensors State

While reading the X, Y and Z-axis acceleration sensor data is straight forward, it was necessary to perform averaging on the data. This was done by determining a simple average over the last 20 readings. The “conditioned” result was then converted from “gravity” to the actual tilt of the robot in both pitch (front-to-back) and roll (side-to-side) using simple arithmetic. This is illustrated in Figure 8. Both data values were indicated on the front panel.

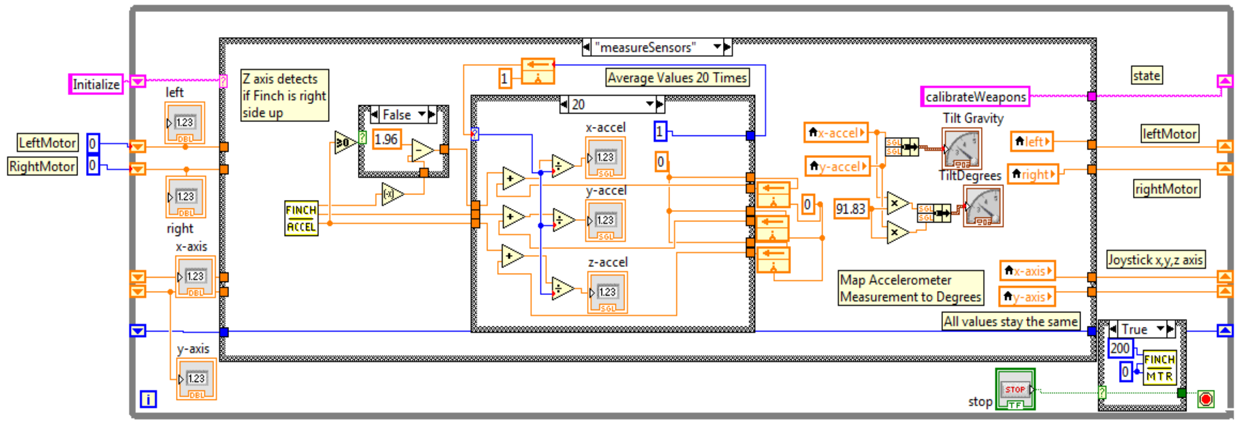

Figure 8. LabVIEW Measure Sensors State.

LabVIEW Front Panel

The front panel was the user interface that displayed the controls and indicators. The main components were:

- Joystick X/Y axis values
- Motor power left/right speed settings
- Pitch and roll values in both “degrees” and “gravity” (blue and red indicators on each gauge)

An example of the screen shot of a student’s front panel is shown in Figure 9. Additionally, there are several other fields as the students began to include additional features. The joystick controller has several finger buttons and knobs, and these were used to light up the tri-color Led as well as play a series of tones from the robot’s speaker.

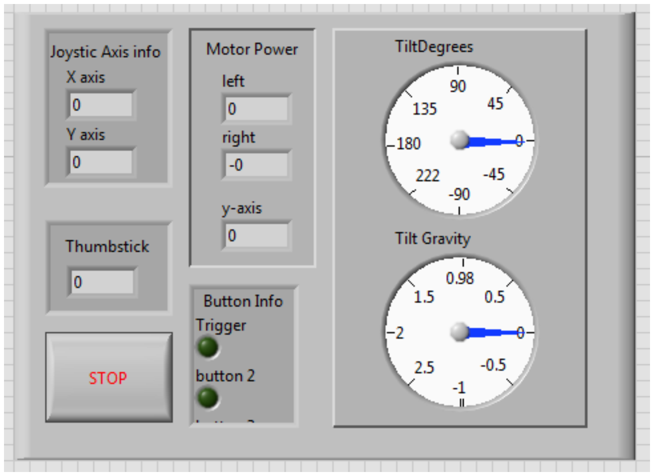
Joystic Axis info
X axis 0
Y axis 0
Thumbstick 0
Motor Power
left 0
right -0
y-axis 0

Figure 9. LabVIEW Front Panel.

Assessment by the Students

Assessment data is provided from students completing the project during the fall 2013 semester, since this was the first time the project was offered. The two student teams that completed the project were asked to assess their work. Their responses were:

- The project was reasonable for the course, since the design for the project was based on the materials learned during the semester.
- It was reasonable for a final project, since it took approximately 8 hours to complete and it incorporated a number of programming features.
- It was an interesting and challenging project, since there were multiple solutions to the problem.
- It allows for “bonus” features, such as using the buzzer and tri-color Led for extra operations. A motivated student could easily create additional engineering requirements for the project.

There were also some suggestions to improve the project:

- Find a wireless controlled robot to eliminate the need for a tether to operate.
- The project did not use a DAQ device (had internal sensors), so it did not involve a hardware design element.
- Find a robot that has better position and speed control. The Finch Robot had very little weight and minimal wheel friction, and as a result, it was hard to get it to drive in a straight direction.

Conclusions /Recommendations

Both student teams successfully completed the project. However, it was interesting to see how the two student teams approached their designs and where they focused their development. One team identified the user interface as the section that required the most attention, while the other team focused on sensors and motion. Both teams had trouble in converting gravity readings to degrees and determining pitch from a 3-axis accelerometer.

One observation is that the team that worked mainly on the user interface spent very little time on the sensor and signal conditioning. In addition, they received less satisfaction from the project than the team working on integrating the sensors with the motion and control.

One goal in defining a final lab project is to empower students to determine creative solutions and encourage them to go beyond the listed requirements. This occurred in one of the teams as they implemented a series of audio/visual effects using some of the buttons on the joystick. This can be encouraged by allowing students to select one of several possible options for the final project, and to provide extra credit for students who want to add additional requirements to the project.

Additionally, it was important for student success on the project to have them break the project into a series of smaller operations. Once these were implemented, it was easier for them to combine the operations together to design the complete program.

Acknowledgement

We would like to thank Luke Elliott, a junior in our program, who completed the Finch Robot project and provided the LabVIEW program from which the screen shots were obtained.

Bibliography

1. Bishop, Robert H., *Learning with LabVIEW 2009*, Pearson Education, 2010.
2. Travis, Jeffrey and Jim Kring, *LabVIEW for Everyone*, 3rd Edition, Pearson Education, 2007.
3. Essick, John, *Hands-On Introduction to LabVIEW for Scientists and Engineers*, 2nd Edition, Oxford University Press, 2013.
4. Web Site http://www.finchrobot.com/