LSM Sensor Integration with ADC and Motor Drive Testing on Pie 4

Anirudh Ashrit
014535846
MS Electrical Engineering
San Jose State University, San Jose, CA 94303
E-mail: Anirudh.ashrit@sjsu.edu

Abstract

A design to measure the angle of rotation of a Stepper Motor connected to Motor Drive is built. A Potentiometer is connected to change the duty cycle of the PWM which further alters the rotation speed of the motor. An ADC is connected to convert the analog input obtained from the motor to further pass on to the LSM303 sensor, which is a combined functional device of Magnetometer and Accelerometer. The above requirements were implemented on a Pi4 Board.

Keywords: Stepper Motor, Motor Drive, LSM303, ADC, PWM, Duty Cycle, Pi4, Potentiometer.

1. Introduction

An embedded system is a combination of computer hardware and software, either fixed in capability or programmable, designed for a specific function or functions within a larger system. Industrial machines, agricultural and process industry devices, automobiles, medical equipment, cameras, household appliances, airplanes, vending machines, and toys, as well as mobile devices, are possible locations for an embedded system.

Embedded systems are computing systems, but they can range from having no user interface (UI) -- for example, on devices in which the system is designed to perform a single task -- to complex graphical user interfaces (GUIs), such as in mobile devices. User interfaces can include buttons, LEDs, touchscreen sensing and more. Some systems use remote user interfaces as well.

1.1 IoT

The Internet of Things, or IoT, refers to the billions of physical devices around the world that are now connected to the internet, all collecting and sharing data. Thanks to the arrival of super-cheap computer chips and the ubiquity of wireless networks, it's possible to turn anything, from something as small as a pill to something as big as an airplane, into a part of the IoT. Connecting up all these different objects and adding sensors to them adds a level of digital intelligence to devices that would be otherwise dumb, enabling them to communicate real-time data without involving a human being. The Internet of Things is making the fabric of the world around us smarter and more responsive, merging the digital and physical universes.

What is an example of an Internet of Things device?

Pretty much any physical object can be transformed into an IoT device if it can be connected to the internet to be controlled or communicate information.

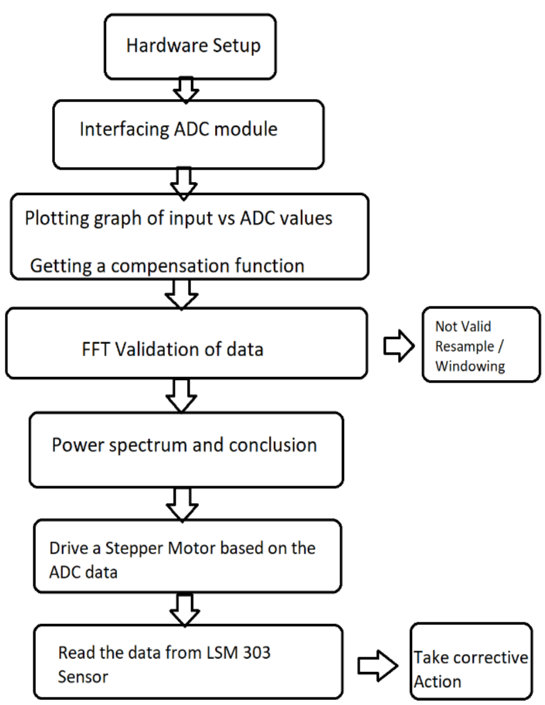
A lightbulb that can be switched on using a smartphone app is an IoT device, as is a motion sensor or a smart thermostat in your office or a connected streetlight. An IoT device could be as fluffy as a child's toy or as serious as a driverless truck. Some larger objects may themselves be filled with many smaller IoT components, such as a jet engine that's now filled with thousands of sensors collecting and transmitting data back to make sure it is operating efficiently. At an even bigger scale, smart cities projects are filling entire regions with sensors to help us understand and control the environment.

2. Methodology

The steps involved in the process of design and implementation are described in this section.

A Potentiometer is connected to Vin (3.3V) and Ground (GND) of the Pi Board, one of the three connections is attached to ADC. ADC is in turn connected to the Pi Board. As potentiometer is varied by the user, the ADC converts this analog input it receives from the potentiometer to a digital and passes it to the Pi. Only two out of 4 channels of the ADC are selected. The digital value obtained from the ADC is passed onto the PWM which will alter the duty cycle. The PWN pin and a direction pin are connected to the Stepper Motor driver which drives a motor. As the Potentiometer changes the speed of motor varies. This data is read by LSM303 sensor which then provides the user with the Acceleration and angle of rotation of the motor. The ADC and LSM303 share a common I2C bus (SDA, SCL) with the Pi Board.

Below a flowchart of the design flow can be found



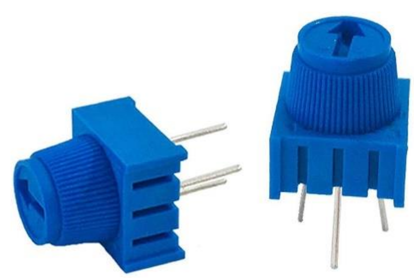
As can be seen from the flow chart, once the Pi is setup, ADC is interfaced and then checked for errors through a compensation function. If FFT validation of data fails, then process until here is repeated with a different sampling rate. The value is then given to Driver to drive the motor. Once the motor is driven, LSM303 reads and prints the data.

3. Implementation

Different Hardware components used in implementing the design are explained below.

3.1 Potentiometer

- Breadboard Trim Potentiometer with Knob
- Adjustable Range: 0-10K Ohm
- Rotation: Adjustable 270 degrees
- Body size: 0.375"x0.375"x0.19" (9.5x9.5x4.9mm)



3.2 ADC (Analog to Digital Converter)

Adafruit ADS1015 12-Bit ADC - 4 Channel with Programmable Gain Amplifier

For microcontrollers without an analog-to-digital converter or when you want a higher-precision ADC, the ADS1015 provides 12-bit precision at 3300 samples/second over I2C. The chip can be configured as 4 single-ended input channels, or two differential channels. As a nice bonus, it even includes a programmable gain amplifier, up to x16, to help boost up smaller single/differential signals to the full range. We like this ADC because it can run from 2V to 5V power/logic, can measure a large range of signals and its super easy to use. It is a great general purpose 12-bit converter.

The chip's small so it comes on a breakout board with ferrites to keep the AVDD and AGND quiet. Interfacing is done via I2C. The address can be changed to one of four options so you can have up to 4 ADS1015's connected on a single 2-wire I2C bus for 16 single ended inputs.

- Wide Supply Range: 2.0V to 5.5V
- Low Current Consumption: Continuous Mode: Only 150µA Single-Shot Mode: Auto Shut-Down
- Programmable Data Rate: 128SPS to 3.3kSPS
- Internal Low-Drift Voltage Reference
- Internal Oscillator
- Internal PGA
- I2c Interface: Pin-Selectable Addresses
- Four Single-Ended or Two Differential Inputs
- Programmable Comparator

3.3 LSM303

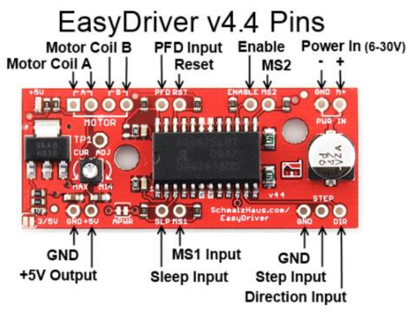
This compact sensor uses I2C to communicate and its extremely easy to use. Since it's a 3.3V max chip, we added circuitry to make it 5V-safe logic and power, for easy use with either 3 or 5V microcontrollers. Simply connect VCC to +3-5V and ground to ground. Then read data from the I2C clock and data pins. There is also a Data Ready and two Interrupt pins you can use

This board/chip uses I2C 7-bit addresses 0x19 & 0x1E

3.4 Stepper Motor Driver

This is a simple to use stepper motor driver, compatible with anything that can output a digital 0 to 5V pulse. The Driver requires a 6V to 30V supply to power the motor and can power any voltage of stepper motor. The Driver has an on-board voltage regulator for the digital interface that can be set to 5V or 3.3V. Connect a 4-wire stepper motor and a microcontroller and you've got precision motor control! Driver drives bi-polar motors, and motors wired as bi-polar. I.e. 4,6, or 8 wire stepper motors.

- MS1 and MS2 pins broken out to change microstepping resolution to full, half, quarter and eighth steps (defaults to eighth)
- Adjustable current control from 150mA/phase to 750mA/phase
- Power supply ranges from 7V to 30V. The higher the voltage, the higher the torque at high speeds



3.5 Stepper Motor

This hybrid bipolar stepping motor has a 1.8° step angle (200 steps/revolution). Each phase draws 1.7 A at 2.8 V, allowing for a holding torque of 3.7 kg-cm (51 oz-in). The motor has four color-coded wires terminated with bare leads: black and green connect to one coil; red and blue connect to the other. It can be controlled by a pair of suitable H-bridges.

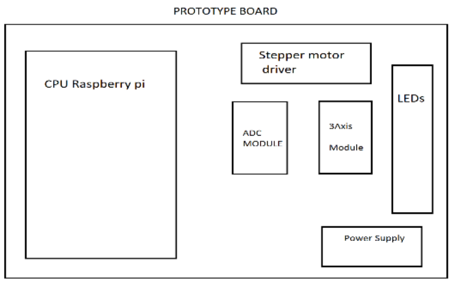
- Size: 42.3 mm square × 38 mm, not including the shaft (NEMA 17)
- Weight: 285 g (10 oz)
- Shaft diameter: 5 mm "D"
- Steps per revolution: 200
- Current rating: 1.68 A per coil
- Voltage rating: 2.8 V
- Resistance: 1.65 Ω per coil
- Holding torque: 3.7 kg-cm (51 oz-in)
- Inductance: 3.2 mH per coil
- Lead length: 30 cm (12″)
- Output shaft supported by two ball bearings

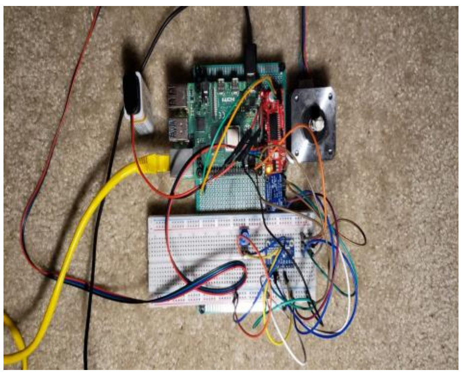
3.6 Pi 4 Board

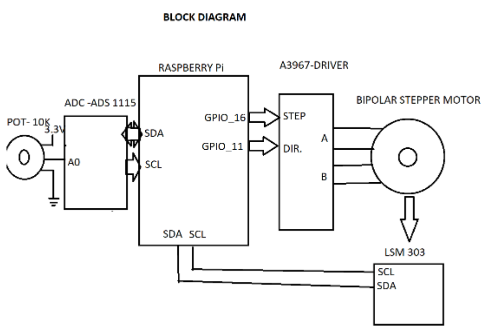
With the Pi 4 being faster, able to decode 4K video, benefiting from faster storage via USB 3.0, and faster network connections via true Gigabit Ethernet, the door is open to many new uses. It's also the first Pi that supports two screens at one -- up to dual 4K@30 displays -- a boon for creatives who want more desktop space.

- System-on-a-chip: Broadcom BCM2711
- CPU: Quad-core 1.5GHz Arm Cortex-A72 based processor
- GPU: VideoCore VI
- Memory: 1/2/4GB LPDDR4 RAM
- Connectivity: 802.11ac Wi-Fi / Bluetooth 5.0, Gigabit Ethernet
- Video and sound: 2 x micro-HDMI ports supporting 4K@60Hz displays via HDMI 2.0, MIPI DSI display port, MIPI CSI camera port, 4 pole stereo output and composite video port
- Ports: 2 x USB 3.0, 2 x USB 2.0
- Power: 5V/3A via USB-C, 5V via GPIO header
- Expandability: 40-pin GPIO header

Circuit Connection







4. Testing and Verification

The design was tested with inputs and was observed to be working properly. The output and code are provided in the following segments.

5. Conclusion

We learned how to integrate ADC output with the Raspberry pi and thus use the interpolation to plot the voltage vs output values and find the compensation function, then we learnt to use FFT as a tool to analyze the power spectrum and be able to analyze the Nyquist criteria.

We also by this lab got to learn the maximum and minimum frequency components and the importance of the total power.

6. Reference

[1] H. Li, “Author Guidelines for CMPE 146/242 Project Report”, Lecture Notes of CMPE 146/242, Computer Engineering Department, College of Engineering, San Jose State University, March 6, 2006, pp. 1.
[2] www.github.com/hualili
[3] https://www.zdnet.com/article/what-is-the-internet-of-things-everything-you-need-to-know-about-the-iot-right-now/
[4] https://opensource.com/resources/raspberry-pi
[5] https://internetofthingsagenda.techtarget.com/definition/embedded-system
[6] https://www.zdnet.com/article/what-is-the-raspberry-pi-4-everything-you-need-to-know-about-the-tiny-low-cost-computer/

7. Appendix

```
import time
import board
import busio
import adafruit_lsm303_accel
import adafruit_lsm303dlh_mag
import Adafruit_ADS1x15
import pigpio
import numpy as np
import math
from math import atan2, degrees


DIR = 20     # Direction GPIO Pin
STEP = 21    # Step GPIO Pin
CW = 1     # Clockwise Rotation
CCW = 0    # Counterclockwise Rotation
MS0=14     #Mode GPIO Pin
MS1=15     #Mode GPIO Pin

pi = pigpio.pi()

# Set up pins as an output
pi.set_mode(DIR, pigpio.OUTPUT)
```

```
pi.set_mode(STEP, pigpio.OUTPUT)
pi.write(DIR,CCW) #set clockwise or
anticlockwise

i2c = busio.I2C(board.SCL, board.SDA)
sensor =
adafruit_lsm303_accel.LSM303_Accel(i2c)
sensor1 =
adafruit_lsm303dlh_mag.LSM303DLH_Ma
g(i2c)
adc = Adafruit_ADS1x15.ADS1015()
GAIN = 1
voltage=[]

def vector_2_degrees(x, y):
   angle = degrees(atan2(y, x))
   if angle < 0:
      angle += 360
   return angle

def get_heading(_sensor):
   magnet_x, magnet_y, _ =
_sensor.magnetic
   return vector_2_degrees(magnet_x,
magnet_y)

def get_inclination(_sensor):
   x, y, z = _sensor.acceleration
   return vector_2_degrees(x, z),
vector_2_degrees(y, z)
MODE = (MS0,MS1)   # Microstep
Resolution GPIO Pins
RESOLUTION = {'Full': (0, 0),
           'Half': (1,0),
           '1/4': (0,1),
           '1/8': (1, 1)
             }
for i in range(2):
   pi.write(MODE[i],
RESOLUTION['Full'][i])
while True:
   for y in range(1648):
      print ('ADC channel 0 value:')
      print('| {0:>6} | {1:>6} | {2:>6} |
{3:>6} |'.format(*range(4)))
      print('-' * 37)
      values = [0]*8
      for i in range(4):
         values[i] = adc.read_adc(i,
gain=GAIN)
         if  i>0:
            values[i]=0
      print('| {0:>6} | {1:>6} | {2:>6} |
{3:>6} |'.format(*values))
      print('')
      time.sleep(0.5)
      v=(values[0]*3.3)/1647 #FOR
GAIN=1, 2047 adc counts=4.096v
      voltage.append(v)
      x= values[0]/1647; #1645=Maximum
ADC counts for 3.3v
      x=x*255;# 255=
      pi.set_PWM_dutycycle(STEP, x)
      pi.set_PWM_frequency(STEP, 500)
      acc_x, acc_y, acc_z =
sensor.acceleration
      mag_x, mag_y, mag_z =
sensor1.magnetic
      print('Acceleration
(m/s^2):({0:10.3f},{1:10.3f},{2:10.3f})'.for
mat(acc_x, acc_y, acc_z))
      angle_xz, angle_yz =
get_inclination(sensor)
      print("XZ angle = {:6.2f}deg   YZ
angle = {:6.2f}deg".format(angle_xz,
angle_yz))
      print('')
      time.sleep(1.0)
      print('Magnetometer
(gauss):({0:10.3f},{1:10.3f},{2:10.3f})'.for
mat(mag_x, mag_y, mag_z))
      print("heading: {:.2f}
degrees".format(get_heading(sensor1)))
      print('')
      time.sleep(1.0)
   f = np.fft.fft(voltage)
   power_spectrum = []
   entire_energy = 0
   highest_frequency_energy = 0
   for i in range(0,1648):#1647 adc
counts=3.3v

power_spectrum.append(math.sqrt((f.real[i]
* f.real[i])+(f.imag[i] * f.imag[i])))
      total_energy += power_spectrum[i]
      if(i==1023):#(N/2)-1

highest_frequency_energy=power_spectrum
[i]
   eta =
(highest_frequency_energy/(total_energy))*
100
```

```
    print("ADC Validation Efficiency:" +
str(eta) + "%")
    if eta>80 and eta <100:
        print("ADC data is valid")
    else:
        print("ADC data is not valid")
```

Output

From below, we can observe efficiency is 82.7% and hence is valid.

```
ADC channel 0 value:
|      0 |      1 |      2 |      3 |
-------------------------------------
|   1431 |      0 |      0 |      0 |

Acceleration (m/s^2):(     3.519,    -1.415,     8.452)
XZ angle =  67.40deg   YZ anglė =  99.50deg

Magnetometer (gauss):(    25.636,  -372.364,  -417.959)
heading: 62.94 degrees

ADC channel 0 value:
|      0 |      1 |      2 |      3 |
-------------------------------------
|   1647 |      0 |      0 |      0 |

Acceleration (m/s^2):(     0.841,     1.530,     8.835)
XZ angle =  84.56deg   YZ angle =  80.18deg

Magnetometer (gauss):(  -154.818,    58.364,  -417.959)
heading: 119.97 degrees

ADC channel 0 value:
|      0 |      1 |      2 |      3 |
-------------------------------------
|   1647 |      0 |      0 |      0 |

Acceleration (m/s^2):(     0.535,     2.065,     9.103)
XZ angle =  86.63deg   YZ angle =  77.22deg

Magnetometer (gauss):(    90.364,    74.818,  -417.959)
heading: 39.56 degrees

ADC Validation Efficiency:82.7%
ADC data is valid
```

Output Video Link:
https://youtu.be/uFNpcXSE0uI