CmpE242
Spring2023

1/

Jan 25th(Wed). 2023
Organizational meeting.
1° Class Syllabus, "Greensheet".

Organizational meeting. Note: 1° In-person Class, M.W. 3:00-4:15pm.; 2° Office Hours, Please make a good use of the O.H. But it will expire by the end of the Semester, By the end of the class. 3°.

San José State University
Computer Engineering Department
CMPE 242 Embedded Hardware Systems, Section 1, S2023 Lab Room/Facility: Rm 268. Access form.

Course and Contact Information

Instructor:	Hua Harry Li, Ph.D.
Office Location:	Engineering Building, Rm 267A
Telephone:	(650) 400-1116 Text Message Only
Email:	hua.li@sjsu.edu
Office Hours:	Mondays and Wednesdays 4:30 – 5:30 pm Zoom link for the Office Hours Join Zoom Meeting https://us04web.zoom.us/j/9841607683?pwd=UlA3aEk1TnV4bjNLQk5CQkw0dDk4UT09 Meeting ID: 984 160 7683 Passcode: 121092
Class Days/Time:	Monday and Wednesday 3:00 – 4:15 pm
Classroom:	Engineering Build Room 325
Prerequisites:	CMPE 180A and 180D, classified standing, or instructor consent

Note: 4°. Nature of the Class

Target platform selection:
- Hardware: NVDA Jetson NANO 2gb/4gb. Broadcom R.pie 3b+, Pie 4.
- Software: 1° Kernel Source Dist. JetPack; 2° Device Development SPI, I2C, PWM, 3° Python, C/C++

Course Format

Technology Intensive, Hybrid, and Online Courses (Required if applicable)

This course requires use of computer/laptop, special microprocessor/ARM hardware for system prototyping, Python and/or C/C++ compiler for software programming. Students must have to participate in classroom activities and after class homework and projects assignment.

Faculty Web Page and MYSJSU Messaging (Optional)

Copies of the course reference materials such as datasheets, project references etc. can be found on line at https://github.com/hualili/CMPE242-Embedded-Systems- and/or SJSU CANVAS. Office hours Zoom link (during the Pandemic): Join Zoom Meeting https://us04web.zoom.us/j/9841607683?pwd=UlA3aEk1TnV4bjNLQk5CQkw0dDk4UT09 Meeting ID: 984 160 7683 Passcode: 121092

5° Github

6°. Homework/projects submission on CANVAS.

Course Description (Required)

Advanced topics dealing with microprocessor and microcontroller hardware and firmware including processor architecture, advanced memory and I/O systems design, multilevel bus architecture, interrupt systems. Design project. Prerequisites: CMPE 180A and 180D, classified standing, or instructor consent.

Course Learning Outcomes (CLO) (Required)

hualili / CMPE242-Embedded-Systems- Publi

Course Learning Objectives (CLO):

CMPE242
Spring2023

2/

Course Description/Nature: Hands-on, Sound Theoretical Background, Coverage of Theory. Note: Sensors. LSM303

3D Printer, CNC machines.

Motors { Stepper motors. NEMA17
3phase BLDC motor.
Automobile window Wipper motor

Robotics.

Required Texts/Readings (Required) Note: Datasheets.

Textbook

1. S3C6410 RISC Processor datasheets, Samsung Electronics https://github.com/hualili/CMPE244/blob/main/2021F-105-%230-cpu-arm11-2018S-29-CPU_S3C6410X.pdf and Development Board schematics https://github.com/hualili/CMPE244/blob/main/2021F-105b-%232018S-29-SCH-Tiny6410SDK-1111-PCB.pdf
2. Nvidia Jetson NANO datasheets.
   (a) Jetson Nano development kit document https://github.com/hualili/CMPE244/blob/main/2021F-108-%231NVIDIA_Jetson_Nano_Developer_Kit_User_Guide.pdf
   (b) Jetson NANO System-on-Module https://github.com/hualili/CMPE244/blob/main/2021F-108b-%23JetsonNano_DataSheet.pdf
   (c) Optional (not used) SoC Park CPU reference https://github.com/hualili/CMPE244/blob/main/2021F-106-tx2-%23Parker_TRM_DP07821001p.pdf
3. Broadcom Raspberry Pie CPU datasheets, BCM2835 CPU https://github.com/hualili/CMPE244/blob/main/2021F-104-%230-cpu-pie-BCM2835-ARM-Peripherals.pdf and https://github.com/hualili/CMPE244/blob/main/2021F-104d-simplifiedCPU-datasheet-%23rpi_DATA_CM_1p0.pdf

Other Readings

1. Professor Li's PPT, handout materials, lecture notes on line https://github.com/hualili/CMPE242-Embedded-Systems-

Ref: on github, Lecture Notes.

2022S-101-notes-cmpe242-3-14.pdf

CmpE242
Spring2023

Grading Information (Required)

Midterm Examination	30%
Homework and Projects	30%
Final Examination	40%

The examination grades are given based on the written answer in
grades are given based on the work submitted, prototype system
programming source code. The detailed rubrics for each homew
assignment is given, check online both CANVAS and https://git
project will be given to students for each submission with multip
learning. Rubrics examples for project 1 submission, for exampl
software implementation counts 40%, report counts 20%, so the

Determination of Grades

Exams: In-Person, In-Class.
Submission To CANVAS.
Need Laptop & Prototype System in the exam.

Alternative1: Broadcom. Raspberry Pie3, 3B+, 4.
Quad ARM.

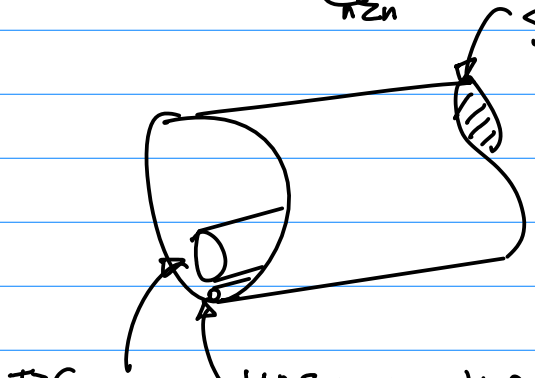
Jan29 (Monday).
1. Homework. 0 pt. Honesty pledge.
Due this Wednesday, ON CANVAS.

Discussion: I/O I/F for An Embedded System:
1° UART (Serial Communication) Tx/Rx/GND
MiniCom/Putty ~1200kbps Slow!
2° SPI. ~100mbps
3° I2C {SDA SCK
4° PWM
5° CAN
6° ADC

"3+1" {MOSI MISO SCK nEn

Ref from the github
2022S-101-notes-cmpe242-3-14.pdf

Example: Selection of Target platform.
Build Selection matrix Below.
1° Architectural Aspects.
X86; ARM, MIPS
CISC : RISC MIPS 1982
for Server
2. User Basis, Marketshare.
3. OS kernel Aspect: Linux/Unix.
4. Forward Looking → GPU →
GPGPU (general purpose) →
AI/ML.
for Example: Jetson NANO
Quad CPU: ARM.
GPU (128 GPUs)

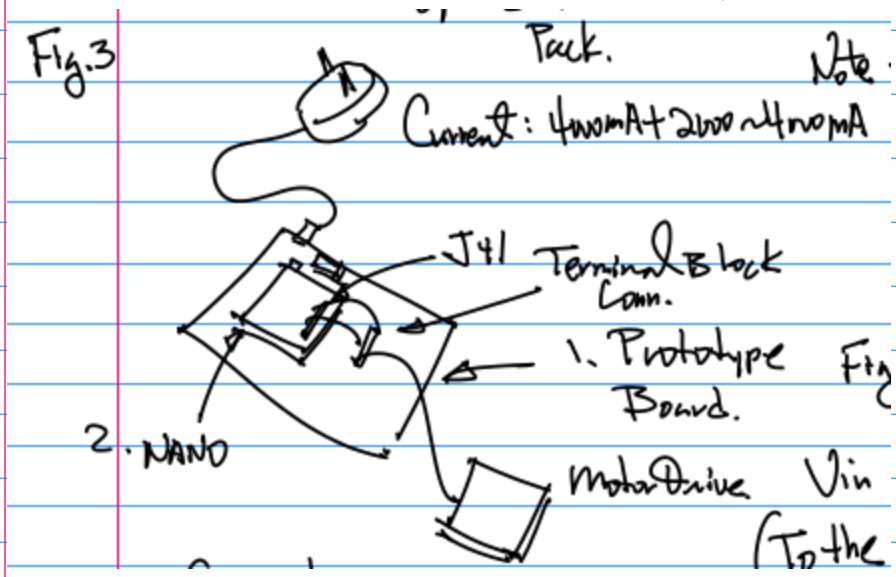


NANO, ~$140
4gb.
Check 2gb Towards the end of Life.

Homework Preparation.

1° Build A prototype Board. Ref. pp.3. Fig.3

Fig.3



Note: 1° prototype Board. Dimension:

Feb 1. (Wednesday).

Note: 1. Target Board Selection By today. Bring your Target Together with the Prototype Board to the Class a week from today.

Example: To prepare the first Homework.

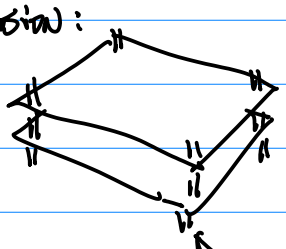
Note: 1° First Homework: "Hello, the World" prototype System.

Python (Pycharm IDE) program to flip LED. Turn ON/OFF LED.

2° Prototype Board.

Order on Line (Amazon) or Local store, Anchor Electronics. (Santa Clara.)

Dimension:



a. Stand-offs Required

b. Connectors Encouraged/Required;

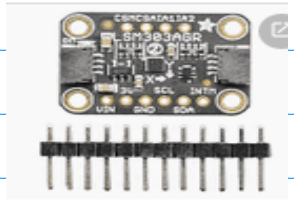
10pcs Upgraded Tiny Whoop JST-PH 2.0 Male and Female Connector Cable for Battery JJRC H36 H67 Blade Inductrix E010 E013

C. Bread Board for Quick Prototyping.

d. Right gauge of the prototyping wires, #28 or higher (etc. 1 or 2 steps. Ref: 10mA ~ upto 40~50mA)

3. LED Assted, Red, Yellow. etc.
Resistors.
Capacitors. 4.7nF (1~2PCS)

4° LSM303 (I2C, SPI)



Adafruit Industries
LLC 4413

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

5° I2C Approximity Sensor.

Note: I2C Mux for Multi-I2C Devices.

6° Design for the 1st Homework.

- Hardware Design
- Software Design
  - I/P Testing
    - I.P. High
    - I.P. Low
  - O/P Testing
    - Op. High → LED ON
    - Op. Low → OFF

Note: Toggle SW to generate "High" or "Low" Input to the GPIO of the NANO

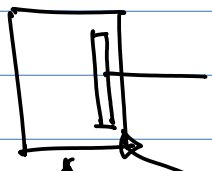
Pin Assignment Table

Pin	Description	Note	
J41-1	3V3		
J41-2	5V		
J41-39	GND		
J41-12	GPIO	gpio79	Output
J41-40	GPIO	gpio78	Input

element14 Community

Jetson Nano Dev-Board Expansion Header

Alt Function	Linux(BCM)	Board Label		Board Label	Linux(BCM)
DAP4_DOUT	78(21)	D21	40 39	GND	
DAP4_DIN	77(20)	D20	38 37	D26	12(26)
UART2_CTS	51(16)	D16	36 35	D19	76(19)
		GND	34 33	D13	38(13)
LCD_BL_PWM	168(12)	D12	32 31	D6	200(6)
		GND	30 29	D5	149(5)
		D1/ID_SC	28 27	D0/ID_SD	
SPI1_CS1	20(7)	D7	26 25	GND	
SPI1_CS0	19(8)	D8	24 23	D11	18(11)
SPI2_MISO	13(25)	D25	22 21	D9	17(9)
		GND	20 19	D10	16(10)
SPI2_CS0	15(24)	D24	18 17	3.3V	
SPI2_CS1	232(23)	D23	16 15	D22	194(22)
		GND	14 13	D27	14(27)
DAP4_SCLK	79(18)	D18	12 11	D17	50(17)
		RXD/D15	10 9	GND	
		TXD/D14	8 7	D4	216(4)



Target GPU NANO

J41, 40pins connector/2 Rows. (Longest Connector)

Identify 3 pins
- GND
- Vcc
- GPIO

Good Ref. Source

https://jetsonhacks.com › nvidia-jetso

NVIDIA Jetson Nano J41 H

Feb 6 (Monday).
Todays Topics: Design of
Prototype Board to Bring up the
target platform (NAND).

Ref: 1° Github

CMPE242-Embedded-Systems- / 2022S / 2022S-103-SDcard-source-distribution-tool-chain-menuconfigu-2021-10-8.pdf

2° Github, Lecture Notes

/stems- / 2022S / 2022S-101-notes-cmpe242-3-14.pdf

Note: Bring your target platform
to the Class for inspection
on Wednesday.

a. Target platform.
b. Work-in-progress.

Note: Try to use ON-Line tool.

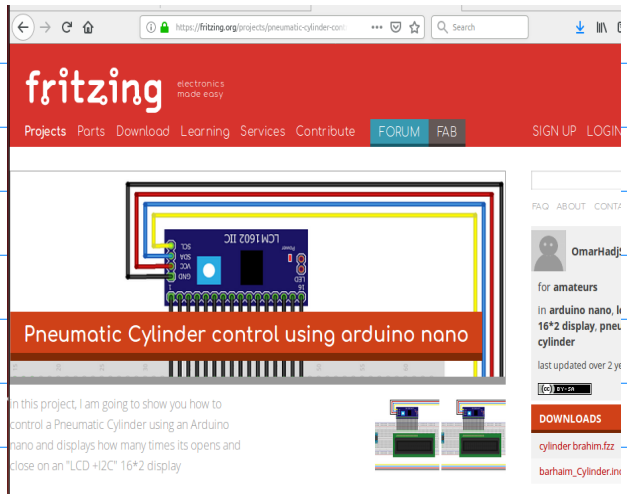
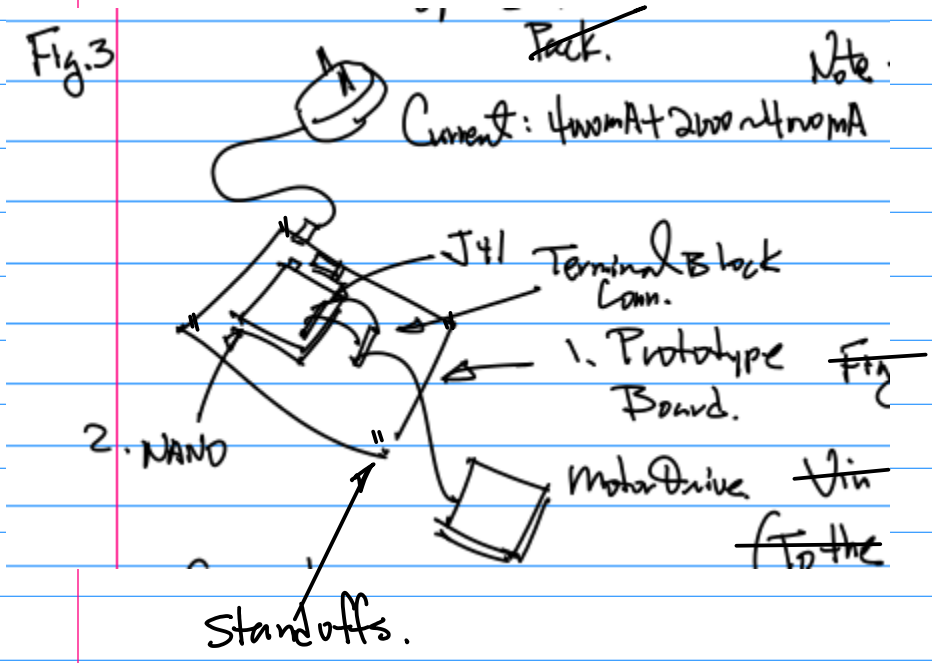
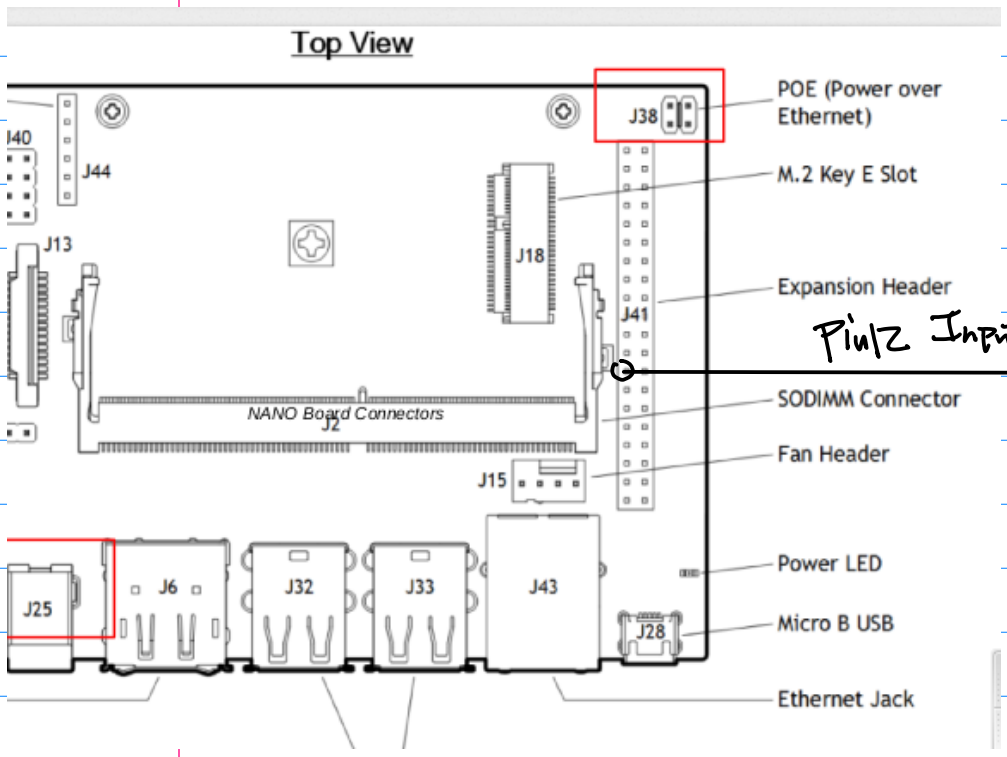
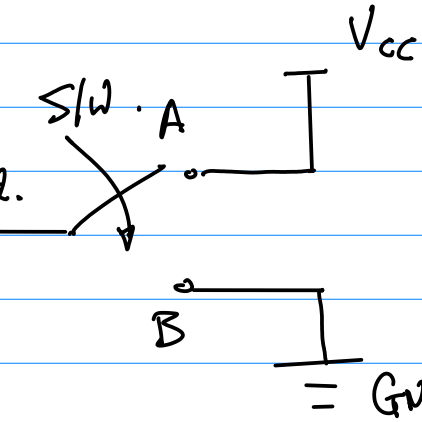


Fig.3



Example: Continuation ON GPIO I/F
Design.
Design for the Input Testing.





↓ modify the Design to Add R_1

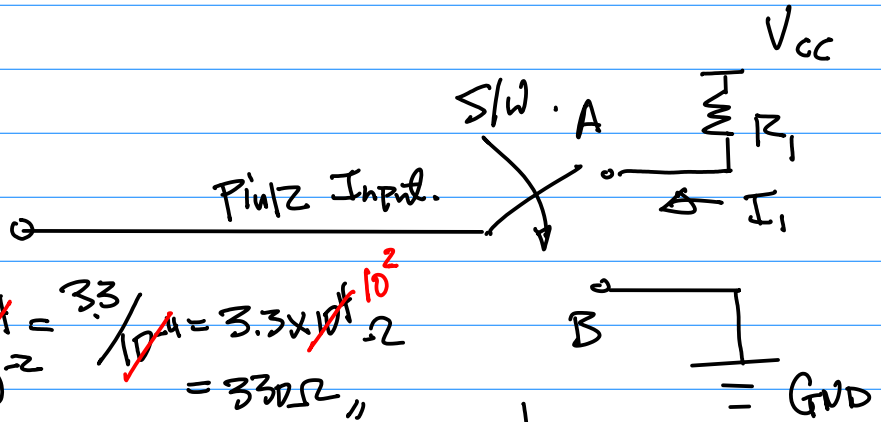
Let $I_1 = 10\,mA$, find R_1.

$V_{CC} - R_1 I_1 = V_{in}$

where $I_1 = 10\,mA$, $V_{in} = 0V$.

$V_{CC} - R_1 \times 10 \times 10^{-3} = 0$, $R_1 = V_{CC}/10^{-2} = 3.3/10^{-2} = 3.3 \times 10^{2}\,\Omega$

$= 330\Omega$ //



↓ modify the Design By Adding R_2

Next, update the Design to add R_2 to Regulate I_2.

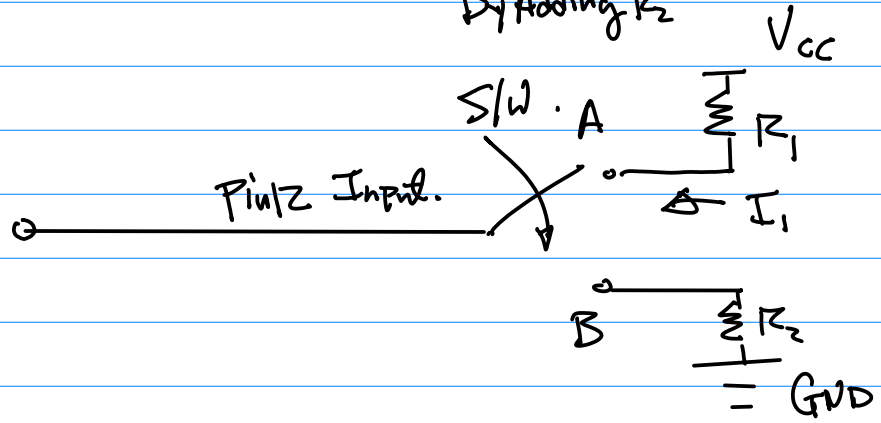
Make $I_2 = 10 \times 10^{-3}$ A

Assum $V_{12} = V_{CC} = 3.3V$.

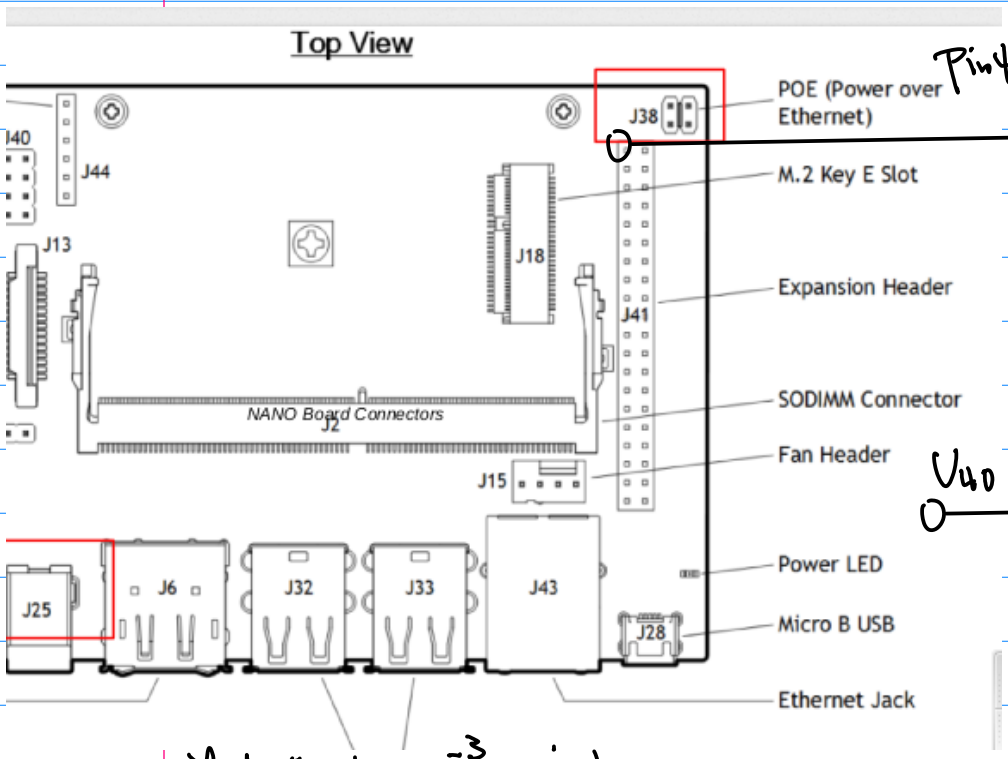
$V_{12} - I_2 R_2 = 0$, hence

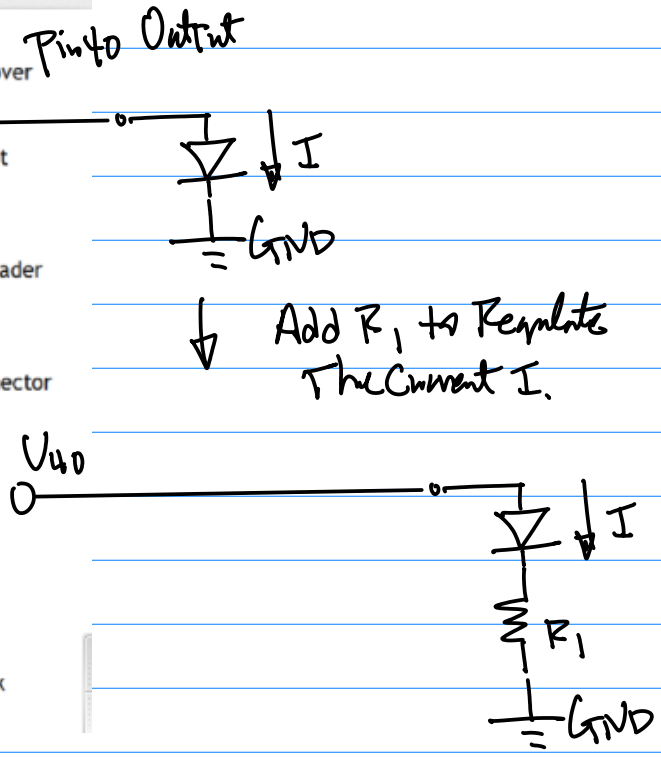
$V_{CC} = I_2 R_2 \Big|_{I_2 = 10 \times 10^{-3}}$

$\therefore R_2 = V_{CC}/I_2 = 3.3/10^{-2} = 330\Omega$.



Output Testing Circuit.





Let $I \simeq 10 \times 10^{-3}$. find R_1.

$V_{40} - I R_1 = 0$, $R_1 = V_{40}/I \Big|_{V_{40}=V_{CC}=3.3V,\ I=10\times10^{-3}} = 3.3/10^{-2} = 330\Omega$.

2022S-103-SDcard-source-distribution-tool-chain-menuconfigu-2021-10-8.pdf

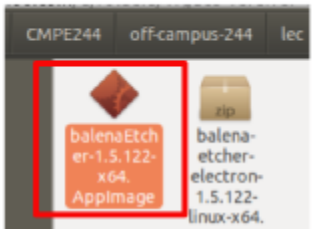
Software Design:

Step 1. Prepare microSD Card, 16GB or 32GB (for Additional App.). Down Load Pre-Compiled/Built Kernel Image from the Nvidia Site

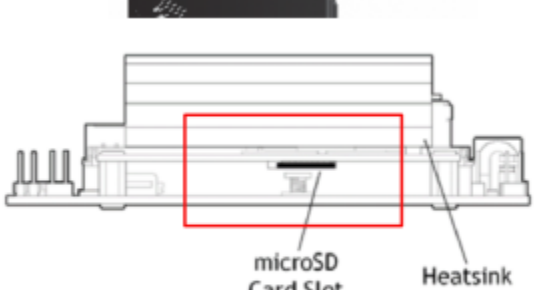
https://developer.nvidia.com/embedded/learn/get-started-jetson-nano-devkit#write

Step2. Down Load the software "Etcher" to your host machine in order to write the Kernel image to your micro SD Card.

(2.1) for Linux host, Download, install, and launch Etcher.
https://www.balena.io/etcher/



Step3. Take the MicroSD, insert it to your target platform.



the power up the NANO Board, Config the Board By following the prompt.

Note: It is recommended to use 4 Amps Power Adaptor.

Step4. Init & Config. for GPIO Driver.

Ref.

2022S-104-gpio-systemLevel-and-c-#2021F-114-gpio-nano-v3-hl-2021-10-20.pdf

P.2. Note: a. Website, Ref. Sources.

Pi and NANO are pin to pin compatible
Jetson Nano GPIO - JetsonHackshttps://www.jetsonhacks.com› ... › GPIO/I2C
Jun 7, 2019 — As you may have heard, the GPIO pin layout on the Jetson Nano is compatible with the 40 pin layout of a Raspberry Pi (RPi).

b. 40 pin Connector is Compatible With Raspberry Pie.

c. By Default, the Kernel Image (O.S.) has already configured GPIO Driver, So, use Command Line to Turn ON/OFF LED As follows.

```
$echo 79 > /sys/class/gpio/export
$ echo out > /sys/class/gpio/gpio79/direction
$echo 1 >  /sys/class/gpio/gpio79/value
$echo 0 >  /sys/class/gpio/gpio79/value
$echo 79 /sys/class/gpio/unexport
$cat /sys /kernel /debug/gpio
```

Connect to the GPIO Driver.
Config the GPIO As An Output
Set output="1"
Set Output="0"
Release GPIO

Homework: GPIO Testing. Due A week from Today. Feb.
Bring the Board to Class for Demo.

Feb 8 (Wed). Wednesday
Homework. Due Feb 15th. (GPIO Homework)

1° Written Requirements for the Homework will be posted on CANVAS.

2° Submission ON CANVAS. No E-mail Submission.

3° Target platform with Prototype Board

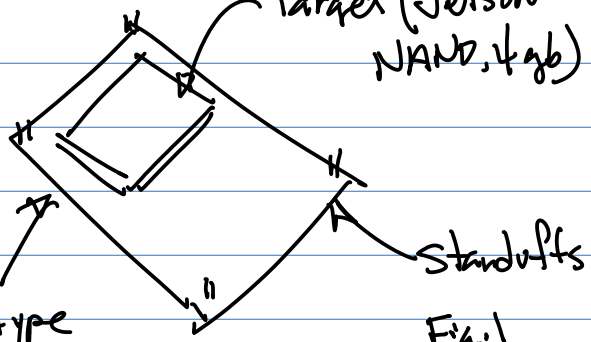
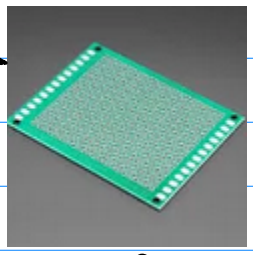


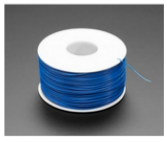
Fig.1



Note: 1° Bread Board for this Homework is OK, But for the Rest of the Homework, We must use Prototype Board.

2° Standoffs.

3° Wire for prototyping should be in the Range of 28AWG ~ 32 AWG



"Wire Wrap" Thin Prototyping & Rep
Wire - 200m 30AWG Blue
PRODUCT ID: 1446
Add to Cart
$7.50
In stock

Fig.2

4° Input/Output Testing Circuit has to be included, please take photos of

(a) Entire System.

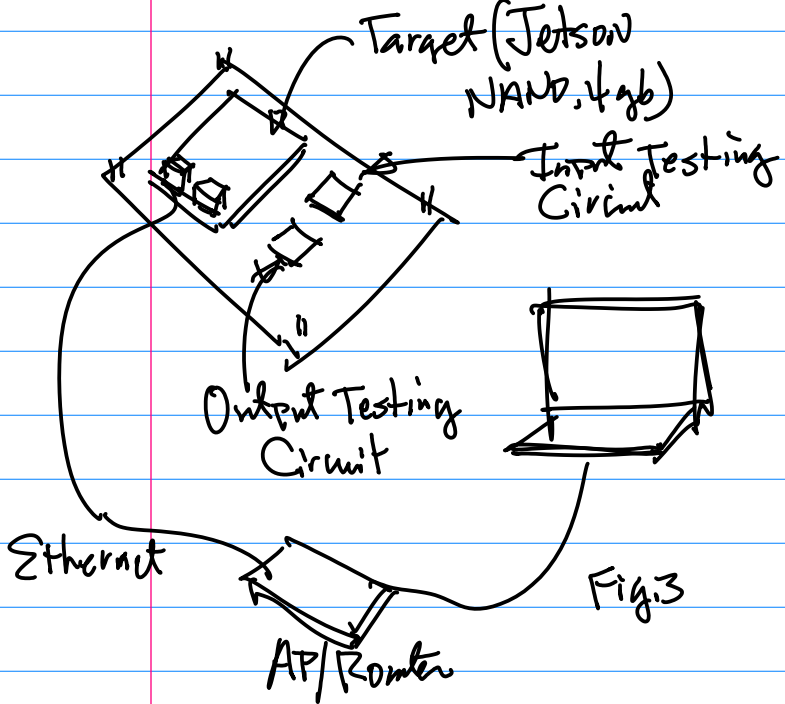


Fig.3

(b) I/P, O/P circuit of the Board. "Closed-In" View.

5° Readme.file ~1 page.

6° Screen Capture of the Program Execution & Result.

Make sure the screen Captures have your personal identifier.

7° Source code Listing.

a. "Template" Name of the program:
Coded by:
Date:
Version: { Release / Debug

Purpose:

Copyright:

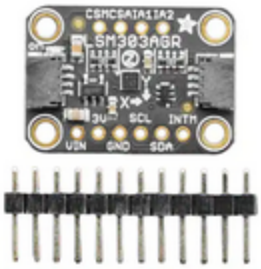
Note:

8° Short Video Clips, Not exceeding 15~30 Sec.

Options. { github for the class.
for your work.
Video Clip(s). Youtube.

Bill of material Needed for the Coming homework and lectures.

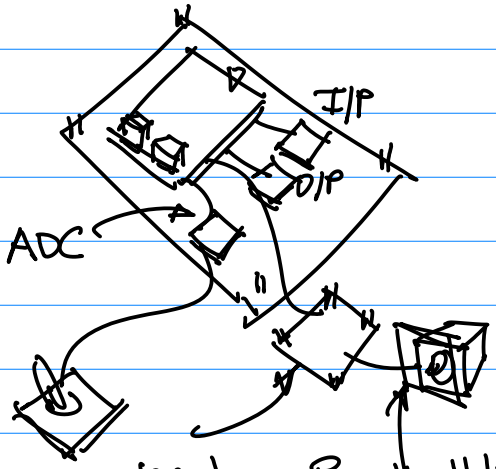
1. LSM303 Sensor $12.95 from digikey.



Please Bring it to the class Next Monday.

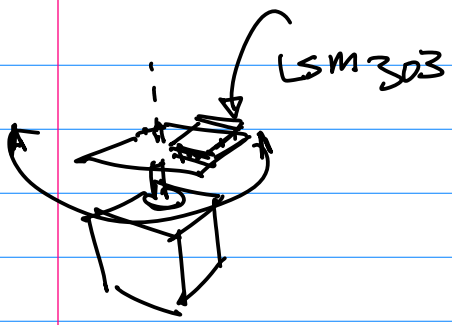
Fig.4

Fig.5



CmpE242
Spring2023

11/



2. Motor.

Option1. Stepper motor
NEMA17

Adafruit Industries LLC
324 Bipolar Stepper
Motor Hybrid Frame Size
17 200 Step 350mA

$14.00
Digi-Key

4 Axis Nema23 Stepper
Motor 270oz-in 76mm
3A Dual Shaft+TB6560
MD430 Driver CNC

$178.00
Amazon.com
Free shipping

Option2. Motor for E.V.

BLDC (Brushless D.C. motor)
$50 ~ $200

for Scooters, eBike, or2

Too Big for the Class, But Similar Controllers.

Option3. Motors from Automobile Industry, from Amazon. $35 ~ $200

Note: please form 2 person Team By Next week.

CMPE242
Spring2023

Human Control Interface Devices. Handlebar Controller for Scooters.

$12.

Ads · Shop adc i2c pie

Datasheet

adafruit-ADS1015 12-Bit ADC - 4

$9.95

Adafruit Industries

Python Reference Code is Available as well.

Wireless Game Console Controller.

~$199.

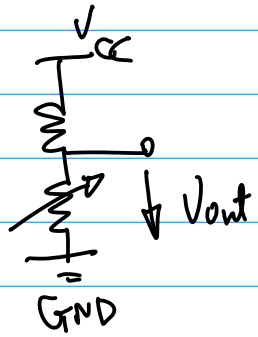
DualSense Edge wireless controller -

Feb 13 (Monday).

Note: Homework Due Feb 15 (11:59pm). TPM. Submission on CANVAS. Inspection.

Example: I2C Based Sensor Interface LSM303

Or: Potential meter.



Bourns Inc. PDB181-

$1.47

Digi-Key

CmpE242
Spring2023

13/

LSM303DLHC

Ultra compact high performance e-compass 3D accelerometer and 3D magnetometer module

Preliminary data

Features

- 3 magnetic field channels and 3 acceleration channels
- From ±1.3 to ±8.1 gauss magnetic field full-scale
- ±2g/±4g/±8g/±16g selectable full-scale
- 16 bit data output
- I^2C serial interface
- Analog supply voltage 2.16 V to 3.6 V
- Power-down mode/ low-power mode
- 2 independent programmable interrupt generators for free-fall and motion detection
- Embedded temperature sensor
- Embedded FIFO
- 6D/4D orientation detection
- ECOPACK® RoHS and "Green" compliant

LGA-14 (3x5x1mm)

Description

The LSM303DLHC is a system-in-package featuring a 3D digital linear acceleration sensor and a 3D digital magnetic sensor.

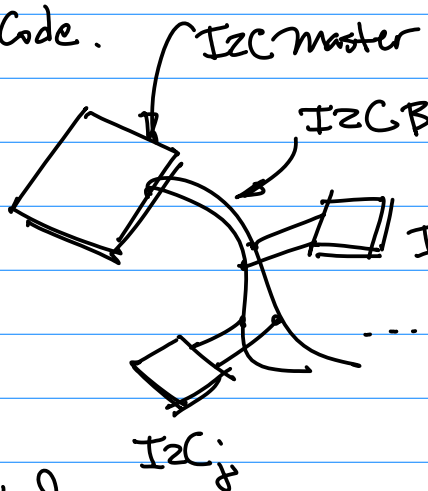
LSM303DLHC has linear acceleration full-scales of ±2*g* / ±4*g* / ±8*g* / ±16*g* and a magnetic field full-scale of ±1.3 / ±1.9 / ±2.5 / ±4.0 / ±4.7 / ±5.6 / ±8.1 gauss. All full-scales available are fully selectable by the user.

Interface Design:

- Hardware Design.
- Software Design.
  - I2C Protocol
  - Coding.
    - Command Line Based Code
    - Python Code.

Target platform:

Jetson NANO. J41 connector has I2C pins.



Target platform.

Fig. 1

2022S-108-LSM303DLHC.PDF Ad

2022S-108b-AngularSensing-i2c-LSM303- f... Ad

Consider I2C Hardware Interface Design.

I2C
- SDA: Serial Data. Bi-directional 2v8 mbps
- sck: Serial clock. Output from the master

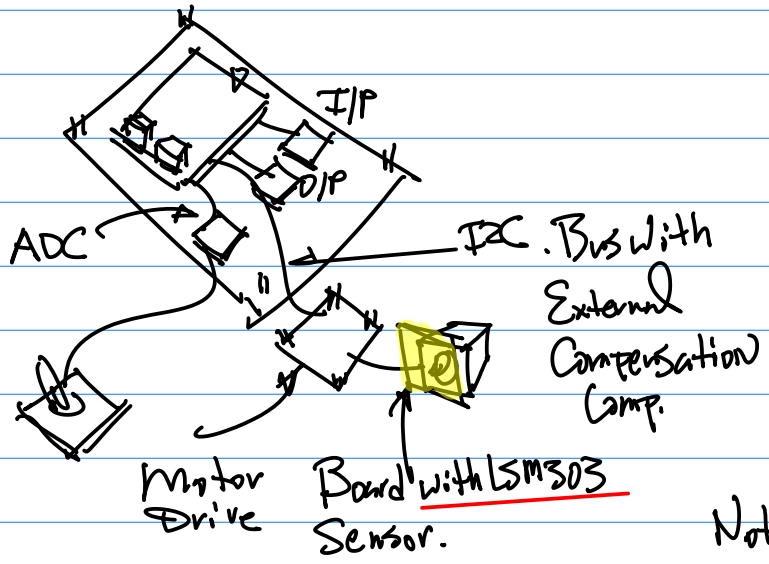
Note: A Typical I2C "Slave Address" takes 7 bits, → $2^7 = 128$ Device Address. → Very often, I2C master Can Only Drive to a few Devices, Such as 4 Devices.

Ref:

2022S-108c-i2c-jetson-nano-2022-04-06 1... Add

Describes the I2C Command Line Testing.

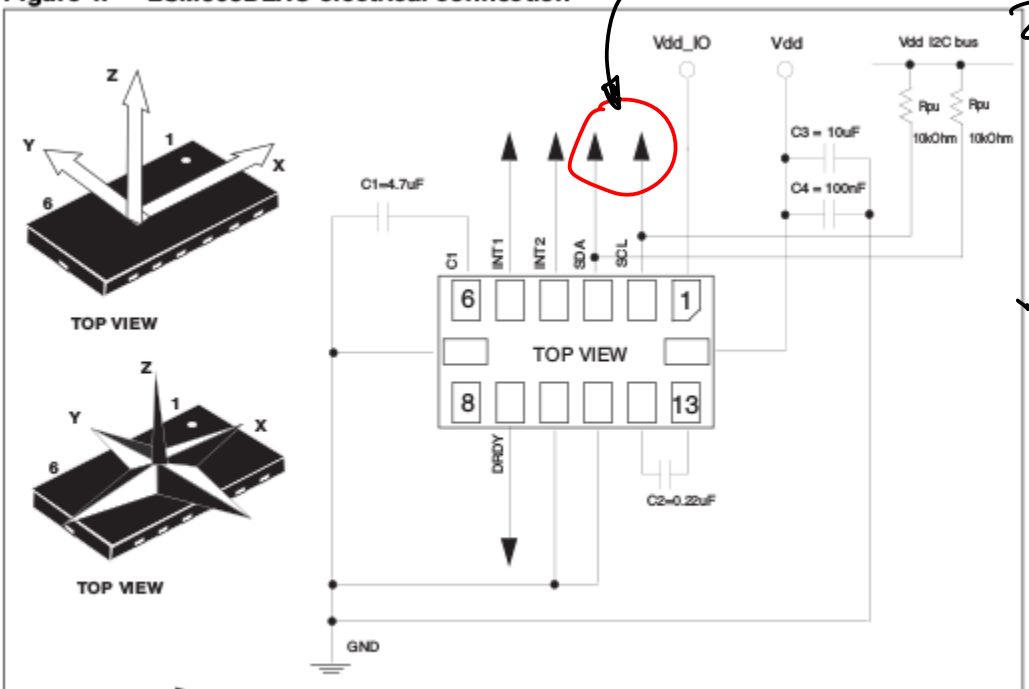
From Fig. 1, Build LSM303 I/F. as follows.



Note: 1° SDA, SCL, I2C Bus. with 10KΩ Pull up Resistors.

2° External Caps. 100nF (C4), 10nF (C3) to form Low Pass filter to Remove "high Freq" Noise. And C_1 (4.7nF), C_2 (0.22nF).

Figure 4. LSM303DLHC electrical connection



Note: 1° Python Code. Works for NVDA Jetson NANO.

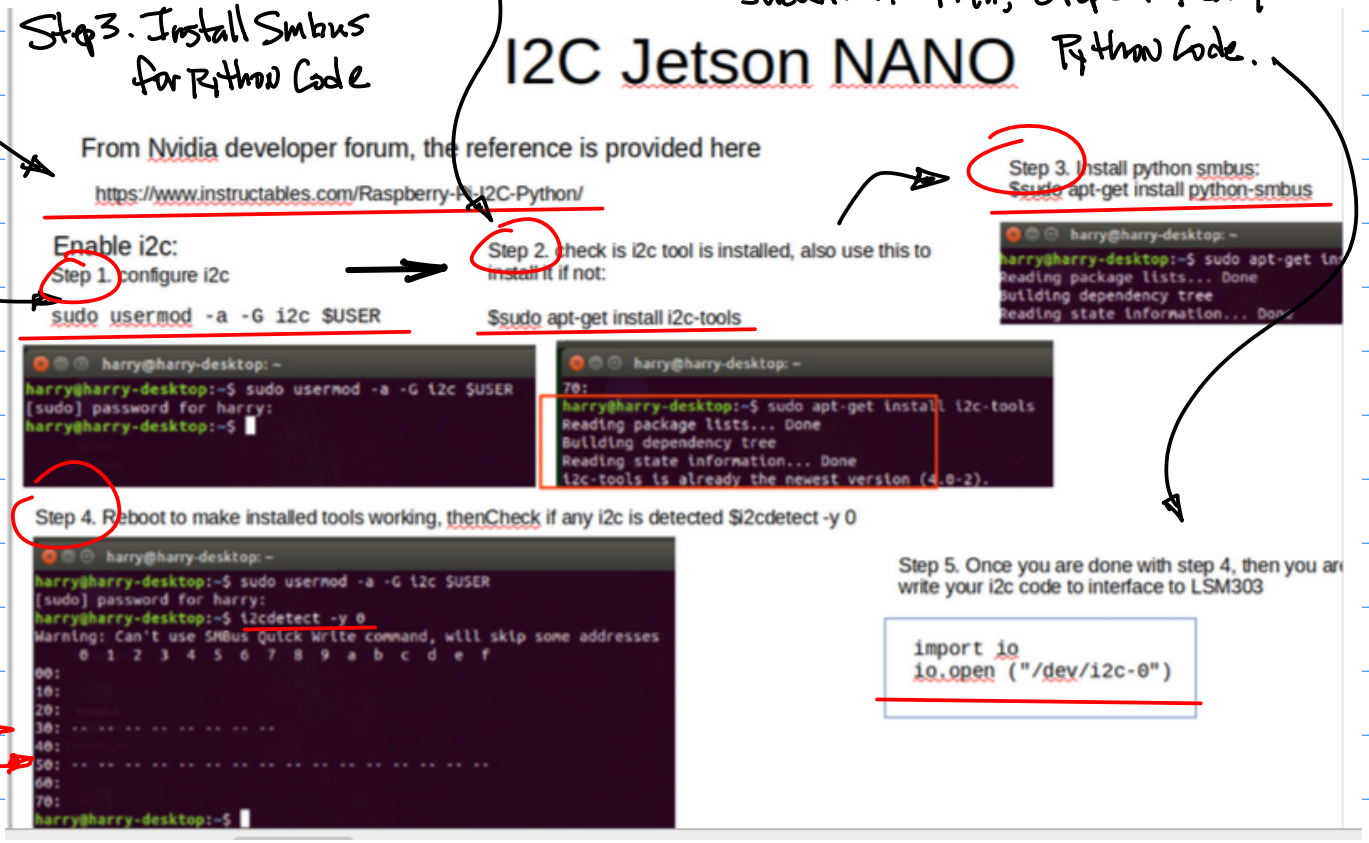
2° Command Line Testing.

Step1. Config. I2C

Step2. Install I2C Tools.

Step3. Install Smbus for Python Code

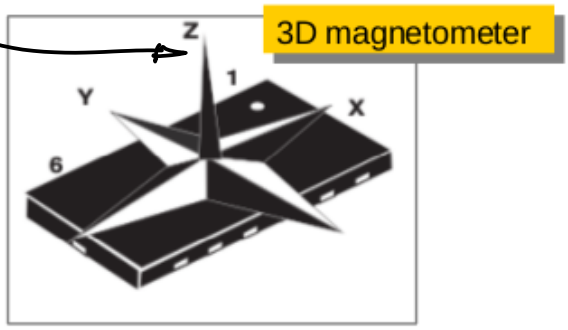
Step4. Check if the installation is succeful. Then, Step5. Ready for Python Code.



Example: I2C protocol / LSM303

Note: 1. Magnetometer. With Reference to the North Pole. & Accelerometer X-, y-, z-Axis.

3D Accelerometer and 3D Magnetomete LMS303



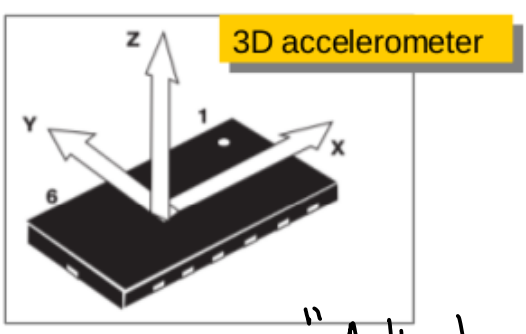


Table 9

Pin name	Pin description
SCL	I^2C serial clock (SCL)
SDA	I^2C serial data (SDA)

Reference: Table 9, pp 19, from LSM303 datasheet

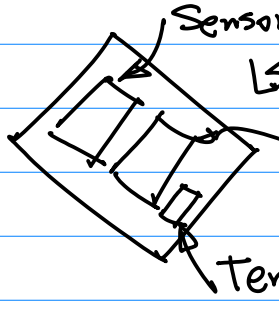
I2C Interface

(1) The transaction started through a START (ST) signal, defined as a high-to-low on the data line while the SCL line is held high.
(2) After ST, the next byte contains the slave address (the first 7 bit), bit 8 for if the master is receiving or transmitting data.
(3) When an address sent, each device compares the first seven bits after ST. If they match, the device is addressed.

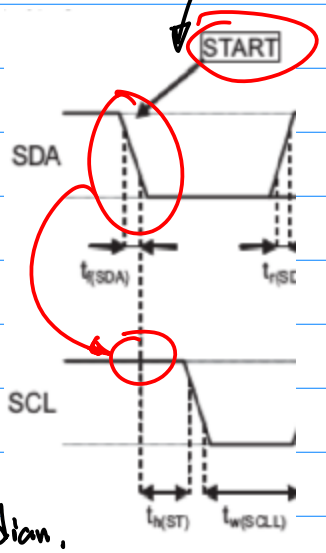
"Active Low"

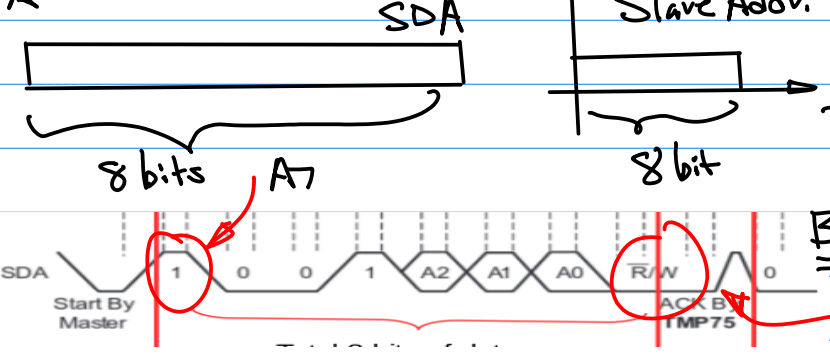
Harry Li, Ph.D. April 2015

Each Sensing Unit has its own I2C Address. "Slave" Address.



Note: 2.





Big Endian.

MS Bit (Most Significant)

Feb15 (Wed)
Homework:
1° Build I2C Sensor Interface to the prototype System.
Sensor LSM303 with I2C Interface.
2° Using Your target platform to show I2C Interface is Supported, with Command line function.
$ i2cdetect.
To Demonstrate the NANO is properly configured with i2ctools installed, And Sm_bus package for Python programming is imported.
3° Hardware Implementation to connect LSM303 Sensor to the NANO.
Using connectors is recommended;
4° Write A python code to read Sensor Output:
(a. Megnetometer,
b. acceleration.)
Print the Sensor input on the Console.
then, make your program Work with gpio Code.
when gpio input = "1", Read/Print Megnetometer Information;
when gpio = "0", Your Program Will read/Print Acceleration information.
5° Submit Source code
6°. Create readme (1 page) file.
7°. photos.
a. Entire System Setup.
b. Closed-In View of the NANO & Sensor interface (Pins/connectors Visible for Inspection).
c. Screen Capture(s) with personal identifier To show the program execution & Data.
8° All the files into one Zip document.
(Readme, photos, Schematics make one pdf file. C/python code Stand-Alone is OK.)
Video Clips optional
9° Submission on CANVAS.

Example: Continuation of I2C Sensor Interface.

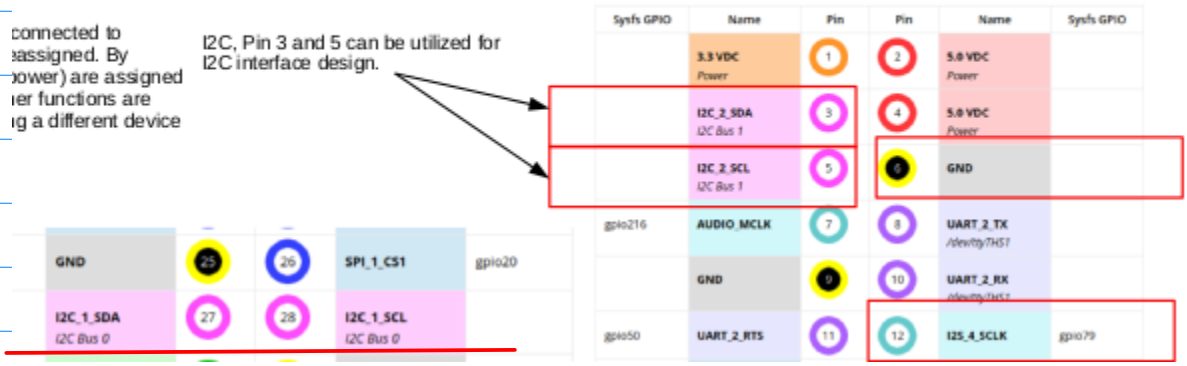
Background: Hardware Design Since I2C Sensor module is Ready

Step 1. Identify the I2C pins on NANO. I2C ports.
Ref: 24 github.

2021F-116a3-i2c-v2-hl-2021-11-18.pdf

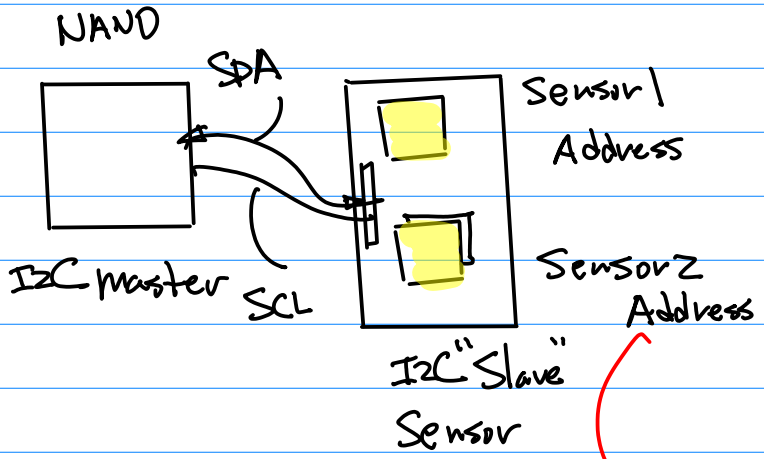
I2C on Jetson Nano J41 Header

https://www.jetsonhacks.com/nvidia-jetson-nano-j41-header-pinout/



Connectivity table

J41-27	SDA
J41-28	SCL
J41-3	SDA
J41-5	SCL.



Step2. Setup & Verify the I2C Drivers By using Command Line, PP15.

Step3. Design methodology.

PP25

Table 14. SAD+Read/Write patterns

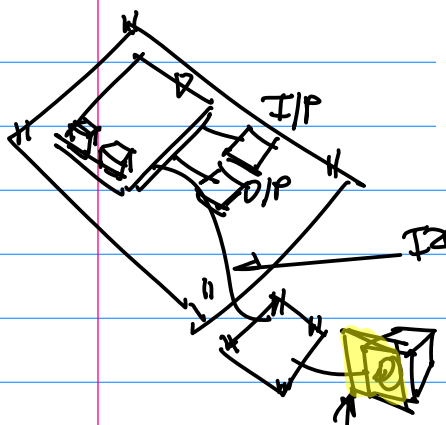
Command	SAD[6:1]	SAD[0] = SA0	R/W	SAD+R/W
Read	001100	0	1	00110001 (31h)
Write	001100	0	0	00110000 (30h)
Read	001100	1	1	00110011 (33h)
Write	001100	1	0	00110010 (32h)

7.1.2 Linear acceleration digital interface

For linear acceleration, the default (factory) 7-bit slave address is 001100xb. T

For Acceleration Sensor. (sensor2)

0011 ; 00xx { 0011 ; 0000 0x30 Write
0011 ; 0001 0x31 Read



I2C Bus with External Compensation Comp.

Board with LSM303 Sensor.

For magnetic sensor, the default (factory) 7-bit slave address is 0011110b (0x3C) for write operations, or 00111101b (0x3D) for read operations.

"Slave" Addr.

Sensor1. Mag.		0X3C	Write
		0X3D	Read
Sensor2. Accel		0X30	Write
		0X31	Read

for the 1st of the Dual Acceleration Sensors.

Next. for Writing the init & configuration pattern to the Sensor unit

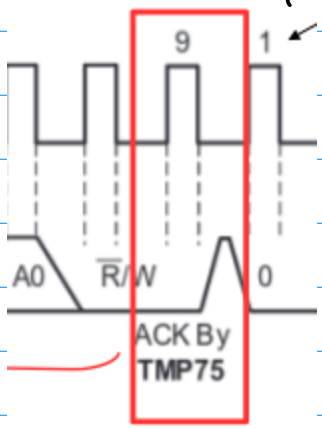
Note: Start — 7 bits Slave Address — W: Write = "1" — Additional Addr. for Special Purpose Register(s). Such as CONTROL Configuration Registers. — Stop.

Table 11. Transfer when master is writing one byte to slave, pp 20, datasheet

Master	ST	SAD + W		SUB		DATA		SP
Slave			SAK		SAK		SAK	

R: Read = "0"

Slave Acknowledgement hardware Level, Response



The SAK slave acknowledges

8 Register mapping

The tables given below provide a listing of the 8-bit registers embedded in the related addresses:

Table 17. Register address map

Name	Slave address	Type	Register address		Default
			Hex	Binary	
Reserved (do not modify)			00 - 1F		
CTRL_REG1_A	TAB.13	rw	20	010 0000	00000111
CTRL_REG2_A	TAB.13	rw	21	010 0001	00000000
CTRL_REG3_A	TAB.13	rw	22	010 0010	00000000
CTRL_REG4_A	TAB.13	rw	23	010 0011	00000000

CMPE242

Spring2023

Feb 20 (Monday):

Prepare for the Homework on LSM303 Interface. Hardwist Due.

1° Hardware Design. To build Jetson NANO (J40, pin27, Pin28) For I2C Interface to LSM303.

2° Use command Line Commands to Verify the I2C Driver is Mapped/Deployed to your Kernel (O.S.) Image. See P.P.T. for Class Additional reference.

3° Use Command Line Instructions to perform Configuration of the Sensor, And Read the Sensor Output. See class P.P.T. for Additional Reference.

4° Use the Sample code (Python Code) from adafruit.com. Modify the Code to perform Sensor init & Config, and Read Sensor Data.

5°. Readme file. (1~3 pages)

6° Source Code.

7° photos of the System Setup. & Execution of the Code.

8° Short video clip (~15 sec.) to Demo the Success of the implementation.

9° Integrate all files (Readme, photos) into one pdf Document, Zip it together with the Source Code, and video clip.

Submission on CANVAS.

Naming:

Cmpe242_LSM303 HW_ First_LastName_SID(4Digits). Zip.

Example: Continuation of I2C Sensor Interface.

Step1. Identification of Jetson NANO pins.

J40-27, J40-28 for I2C

Connect the Hardware Sensor.

↓

Step2. Verify Config. Device Driver (I2C) in the OS. Kernel Space. Command Line Instructions. If Not Pre-config'd, then we will Run the python code from Nvidia for Device Driver Configuration.

↓

Step3. Command Line Instruction to Config the I2C Sensor, And to Read the Sensor Data.

In order to perform that task at Step3. We Need to

- Understand I2C protocol;
- I2C Sensor I/F with its Datasheet.

↓

Step4. Put All Command Line Instructions from Step2+3 into ONE Python Program.

Perform Init & Config. to Read Data from LSM303.

Slave Address.

(Sensor Mag.) 0X3C Write

↓

Sub Control Register

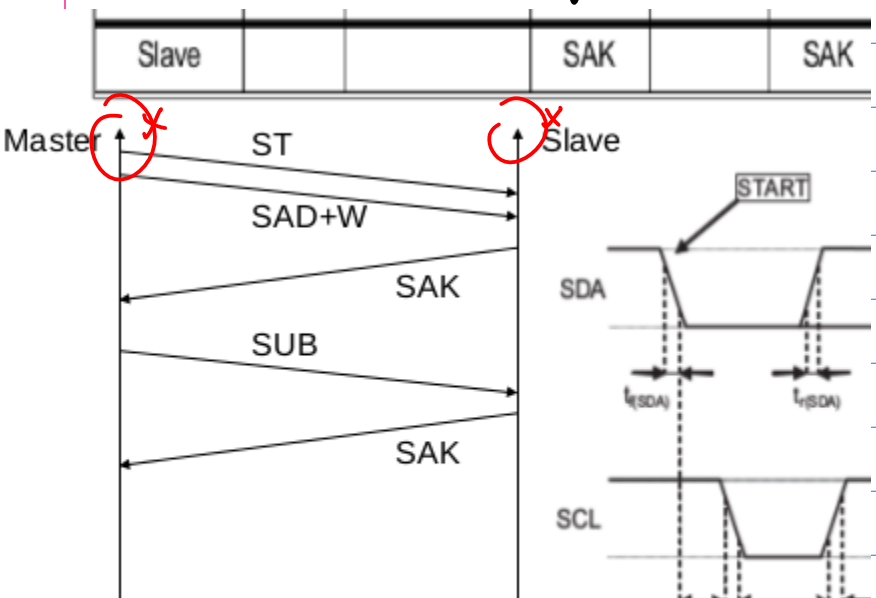
2022S-108b-AngularSensing-i2c-LSM303- final HL 2017-3-13.pdf

↑ PPT. from the github

LSM303 Datasheet

2022S-108-LSM303DLHC.PDF

Now, the Discussion on Table 11. Space-Time Diagram. Required.



CMPE242
Spring2023

22/

Task: To Read Angular Info from the Mag. Sensor
No Slower than 10 times a Second.

Pg. 36.

7.2 Magnetic field sensing register description

7.2.1 CRA_REG_M (00h)

Table 70. CRA_REG_M register

TEMP_EN	0(1)	0(1)	DO2	DO1	DO0	0(1)	0(1)

1. This bit must be set to '0' for correct working of the device

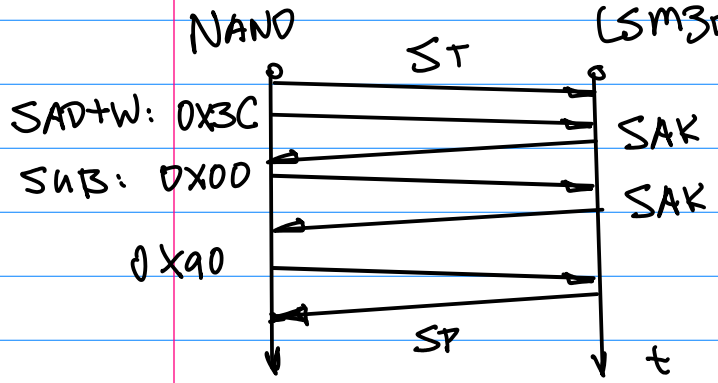
Note: CRA_REG_M
1° = Control Register A for Magnetic Field
2° 8 Bit Reg.
CRA_REG_M[1:0] = 00;
= 0 ✓

Table 72. Data rate configurations

DO2	DO1	DO0	Minimum data output rate (Hz)
0	0	0	0.75
0	0	1	1.5
0	1	0	3.0
0	1	1	7.5
1	0	0	15
1	0	1	30

CRA_REG_M[4:2] = 0x4

then, enable the Temparature Sensing By setting
it to 1. Hence. CRA_REG_M[7:0] = 1001,0000 = 0x90;



Ref. Confirms our discussion.

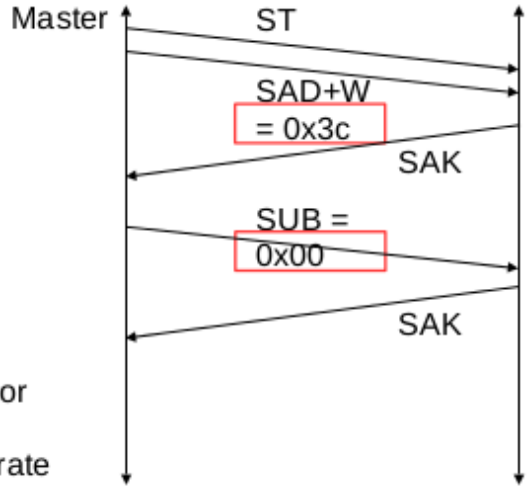
pp.7.

Steps for Init and Config (1)

Master	ST	SAD + W		SUB		DATA		DATA		SP
Slave			SAK		SAK		SAK		SAK	

1. Perform init and config by identify the i2c device (device address, from datasheet 0x3c, pp. 21)

For magnetic sensors the default (factory) 7-bit slave address is 0011110xb. The x bit is 0 for read and 1 for write



2. identify control register(s) for the right sensor block with the sub-address to set data rate
(1) CRA_REG_M register (0x00) to set data rate

TEMP_EN	$0^{(1)}$	$0^{(1)}$	DO2	DO1	DO0	$0^{(1)}$	$0^{(1)}$

Example; 3.0 Hz data rate, so 1 0 0 0 1 0 0 0 → 0x88, so write 0x88 to 0x00 location (CRA_REG_M)
DO2-DO1-DO0 = 0 1 0

Harry Li, Ph.D. Mar 2017

Now, Read Data. pp.10 PPT.

Read the Sensor Data

Master	ST	SAD +W		SUB		SR	SAD +R			MAK		MAK		NMAK	SP
Slave			SAK		SAK			SAK	DATA		DATA		DATA		

Repeated start

Master Acknowledgment

No Master Acknowledgment

Feb 22 (Wed)

Ref: from the class github/Cmpe242.

2023S-102-i2c-command-line-2023S-104-i2c-v2-jetson-nano-2023-02-8.pdf

Example: Continuation of I2C Interface Design.

Given Tech. Spec. → Binary Pattern for Init & Config.

Coding for I2C Device Interface.

1° O.S. Kernel Related Tool (Driver).

2° Python Code.

find if the O.S. Kernel Image is equipped with Device Driver; if Not, then what to do to bring the proper driver to install;

↓

Use Command line instruction(s) to interface the i2c Device(s).

↓

Python Code implementation

Note: Good Ref. Sources.

From Nvidia developer forum, the referen

https://www.instructables.com/Raspberry-Pi-I2C-Pyt

Enable i2c:

Step 1. configure i2c

```
sudo usermod -a -G i2c $USER
```

```
harry@harry-desktop: ~
harry@harry-desktop:~$ sudo usermod -a -G i2c $USER
[sudo] password for harry:
harry@harry-desktop:~$
```

Step 2. check is i2c tool is installed, also use this to install it if not:

$sudo apt-get install i2c-tools

```
harry@harry-desktop: ~
$USER
70:
harry@harry-desktop:~$ sudo apt-get install
Reading package lists... Done
Building dependency tree
Reading state information... Done
```

For Python Coding Purpose, install Python-smbus

Step 3. Install python smbus:
$sudo apt-get install python-smbus

```
harry@harry-desktop: ~
harry@harry-desktop:~$ sudo apt-get install py
Reading package lists... Done
Building dependency tree
Reading state information... Done
```

Step 2
install i

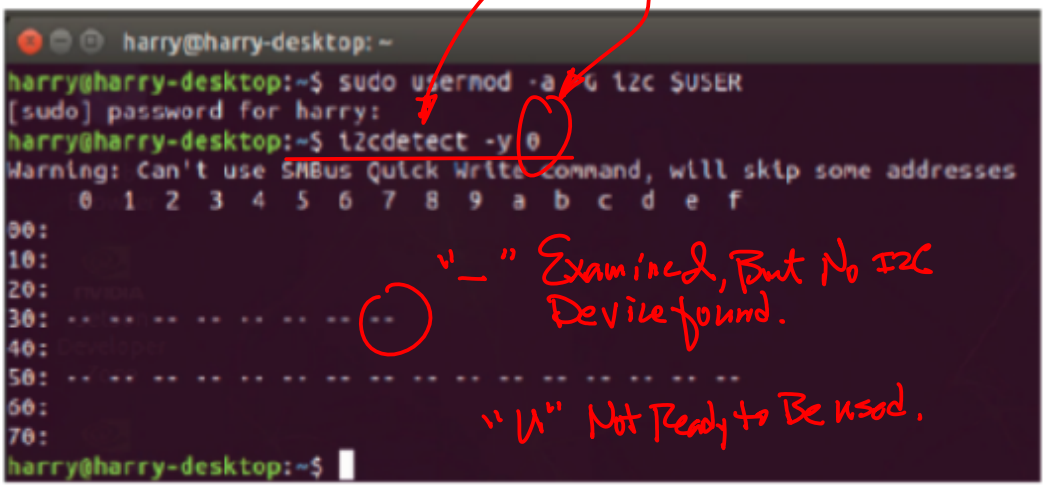
$sudo

Cmpe242
Spring2023

To Detect i2c Devices

PWM ϕ

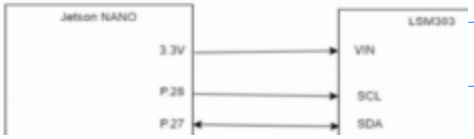
Step 4. Reboot to make installed tools working, thenCheck if any i2c is detected $i2cdetect -y 0



Step 5. Once you are done with step 4, then you are ready to write your i2c code to interface to LSM303

```
import io
io.open ("/dev/i2c-0")
```

Pin 27 (SDA), 28 (SCL) from Billy La



BaseLine Reference:

https://learn.adafruit.com/lsm303-accelerometer-slash-compass-breakout/python-circuitpython

Python & CircuitPython for LSM303 sensor with CircuitPython and the Adafruit CircuitPython LSM303 Accelerometer
https://github.com/adafruit/Adafruit_CircuitPython_LSM303_Accel
Adafruit CircuitPython LIS2MDL or
Adafruit CircuitPython LSM303DLH Magnetometer libraries
https://github.com/adafruit/Adafruit_CircuitPython_LSM303DLH_Mag
These libraries allow you to easily write Python code that reads the accelerometer and magnetometer values from the sensor.

CmpE242
Spring 2023

-6/

```
import time
import board
import adafruit_lsm303dlh_mag
i2c = board.I2C()  # uses board.SCL and board.SDA
sensor = adafruit_lsm303dlh_mag.LSM303DLH_Mag(i2c)
while True:
    mag_x, mag_y, mag_z = sensor.magnetic
    print('Magnetometer (gauss): ({0:10.3f}, {1:10.3f}, {2:10.3f})'.format(mag_x, mag_y, mag_z))
    print('')
    time.sleep(1.0)
```

Using Command Line instruction for debugging.

2. $i2cget #for reading/writing testing

```
harry@harry-desktop:~$ i2cget
Usage: i2cget [-f] [-y] I2CBUS CHIP-ADDRESS [DATA-ADDRESS [MODE]]
  I2CBUS is an integer or an I2C bus name
  ADDRESS is an integer (0x03 - 0x77)
  MODE is one of:
    b (read byte data, default)
    w (read word data)
    c (write byte/read byte)
    Append p for SMBus PEC
harry@harry-desktop:~$
```

Device Addr.

```
nvidia@nvidia-desktop:/sys/class/i2c-dev$ i2cset 0 0x19 0x20
WARNING! This program can confuse your I2C bus, cause data l
I will write to device file /dev/i2c-0, chip address 0x19, d
0x20, data 0x7f, mode byte.
Continue? [Y/n] y
nvidia@nvidia-desktop:/sys/class/i2c-dev$ i2cget -y 0 0x19 0
0xd7
nvidia@nvidia-desktop:/sys/class/i2c-dev$ i2cget -y 0 0x19 0
```

Note: Read Operation, pp.19

Table 13. Transfer when master is receiving (reading) one byte of data from slave:

Master	ST	SAD + W		SUB		SR	SAD + R			NMAK	SP
Slave			SAK		SAK			SAK	DATA		

(1)

Data are transmitted in byte format (DATA). Each data transfer contains 8 bits. The number of bytes transferred per transfer is unlimited. Data is transferred with the most significant bit (MSB) first. If a receiver can't receive another complete byte of data until it has performed some other function, it can hold the clock line SCL LOW to force the transmitter into a wait state. Data transfer only continues when the receiver is ready for another byte and releases the data line. If a slave receiver doesn't acknowledge the slave address (i.e. it is not able to receive because it is performing some real-time function) the data line must be left HIGH by the slave. The master can then abort the transfer. A LOW to HIGH transition on the SDA line while the SCL line is HIGH is defined as a STOP condition. Each data transfer must be terminated by the generation of a STOP (SP) condition.

Feb27 (Monday).

Note: 1° Homework on I2C LSM303 Sensor Interface.

Consider Motor/Actuator Design & Implementation
- Stepper Motor
  - BLDC (phase A, B, and C)

Ref: 1° Definitions/Introduction of Stepper Motor and Stepper Motor Drive

2022S-105b-StepperMotor-Control-part1-2018-2-13.pdf

2° Motor Drive (Professional Grade) Example.

2022S-105b-StepperMotor-Control-part1-2018-2-13.pdf

3° Human Hand Control with P.O.T. for Debugging

2022S-105c-motor-drive-manual-updated-hl-2022-2-23.pdf

Example: Characterization of A Stepper Motor. pp.2.

Note2: The 4-wires of the motor are not directly connected to the CPU of the target platform.

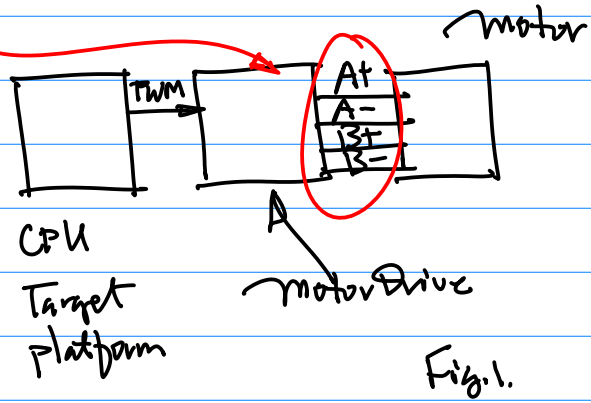


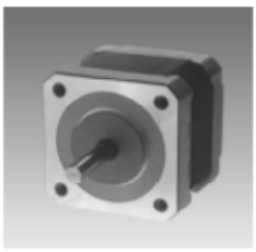
Fig.1.

3° Behavior of A Stepper Motor
a. Full Step Actuation of A motor.
200 Steps for 360° Revolution.
1 Full Step (FS) → 1.8 Degree Angular Displacement.

Note: PWM of A Target platform.

Note1°: 4 wires { A+, A−; B+, B−

Standard Stepper Motor



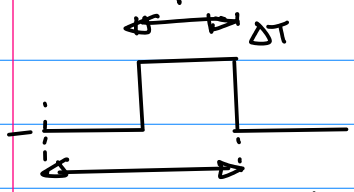
Stepper Motors

Note:
1. One revolution (360 degree) = 200 steps;
2. 1.8 degree per step;
3. Four wire stepper motor can provide

A full step:	1.8 degree;
A half step:	1.8/2 degree;
A quarter step:	1.8/4 degree;
One eighth step:	1.8/8 degree.

4. Stepper drive circuit provide 4 wires output to connect to the 4-wire (A1, A2, B1, B2) stepper motor, by modulation of the base band signal of each wire, the above step size can be achieved.

PWM: Pulse Width Modulation.



η = Duty Cycle = $\Delta T / T$ ... (1)

The control of f_{PWM}, and Duty cycle η are done by Coding of the Special purpose Registers.

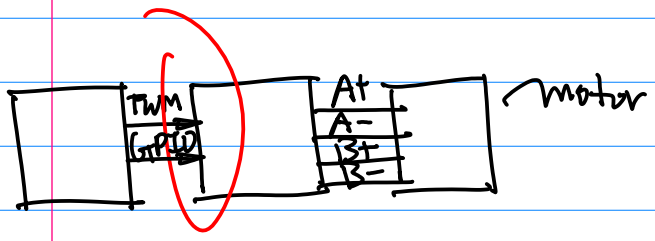
f_{PWM}: Common Range 50 Hz ~ 1 KHz.

b. Half Step (HS)
Angular Displacement
0.9 Degree/Step

c. Quarter Step (QS): 0.45 Degree/Step

d. 1/8 Step: 0.225 Degree/Step.

Consider the inputs to A Stepper motor Drive.



CPU Target platform

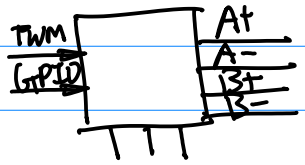
Motor Drive

Fig. 2.

CPU	Drive Board	Note
PWM (Output)	Input	Controls the speed.
GPIO (Output)	Input	Direction (C.W. or. C.C.W.)

Note: pins/connectors Information To Be Added to Complete this table.

Suppose we would like to have 5 Different Step Options for the Drive Board, Since $2^3 = 8$, $2^2 = 4$. Hence, we will Need 3 pins Added to the Driver Board.



Configuration Pins

CF2	CF1	CF0	
0	0	0	F.S.
0	0	1	H.S.
0	1	0	1/4 Step
0	1	1	1/8 Step
⋮			⋮

Use 3 Additional GPP pins for the upgrade of Step Configuration.

CmpE242
Spring2023

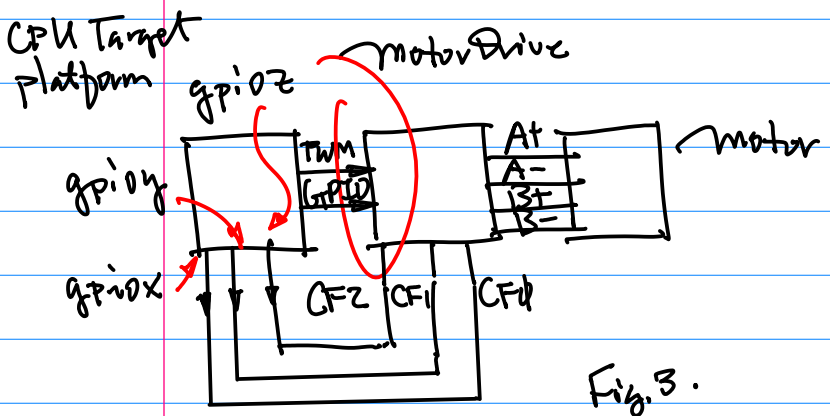


Fig. 3.

March 1st (Wed)
Note!: Homework is coming for Design & Implement Stepper motor Control with PWM.

2° Bring your LSM303 Homework to the class for Inspection & Demo.

motorDriveBoard.
1° Speed of the Actuation, f_{pwm} from the target Board.

10Pcs A4988 StepStick ...
$14.29
Amazon.com

HiLetgo 2pcs A3967
$13.49
Amazon.com

Current ~1000mA to 1500mA.
Speed $\leq f_{pwm} \simeq 1$ kHz.
Not Recommended.

Note: It is OK to use L298N [29] As Entry Level Drive Board

4 PACK L298N Motor
$11.59
Amazon.com

Next Level higher, Recommended

Microstepping Stepper Drive: 2.2A per phase, 2-phase output, 12-24 VDC, 400 to 12800 steps ...
$27.50
AutomationDirect...
Get it by 3/6

Now, Discussion ON P.I.D. Controller Design.

Proportional
Integral
Derivative

Ref. 2022S-107a-PID-v5-2019-4-25.pdf

Example: Robotics (Cart) Application.
Task: To Drive the Robot from Point A to Point B.
$\vec{P}_A(x_A, y_A)$, $\vec{P}_B(x_B, y_B)$.

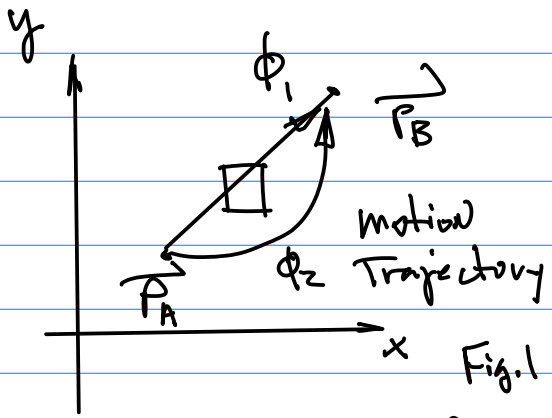


Fig.1

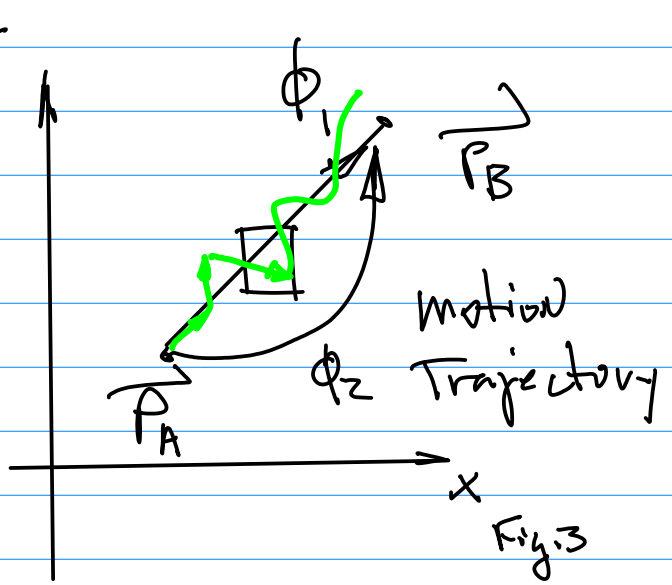


Fig.3

ϕ_1, ϕ_2 are the trajectories from Existing Technology, such as D.L. (Deep Learning), D.R.L (Deep Reinforcement Learning)

Input: ① PWM + ② GPIO

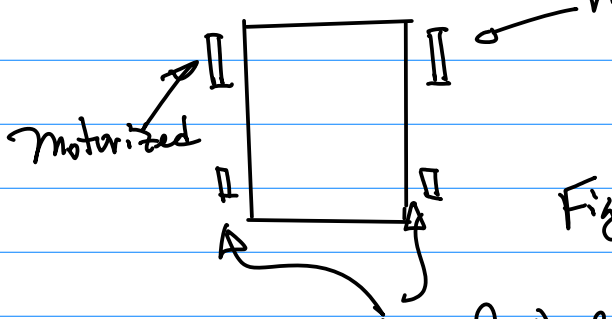


Fig.4

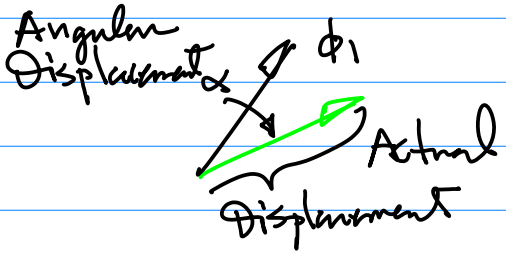


Fig.5

Output Displacement; Angular Displacement

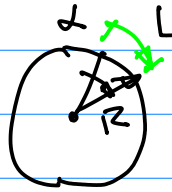
Start with open loop, where plant H(s) can be a stepper motor for example, so we have

$Y(s) = H(s)\,X(s)$ ... (1)

Fig.2

f_{PWM} → Displacement

f_{PWM} → Configuration of the motor Drive, F.S. H.S. 1/4S etc. → physical Dimension of the motorized wheel.



L: Distance travel per one PWM pulse

α if Full Step:

360/200 for one pulse

from a PWM.

Based on this, we can find travel Distance of the wheel, physical displacement

Add Sensors to measure the System Performance.

Open Loop System:

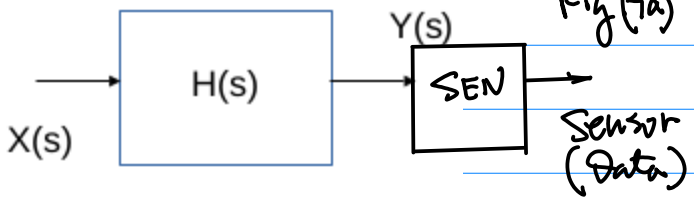


Fig (7a)

With Sensor Data, we can measure the performance of the "Open Loop" System.

Use the sensor data to control/adjust the input with the objective of minimizing the error.

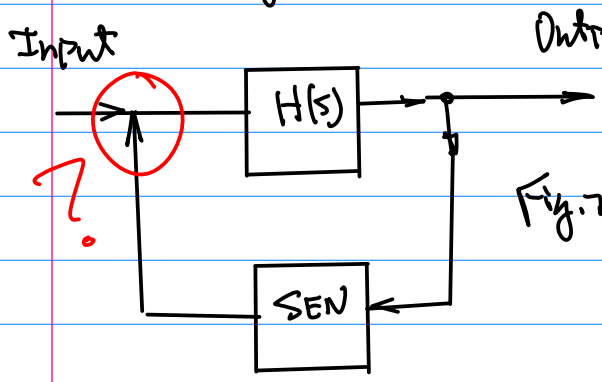
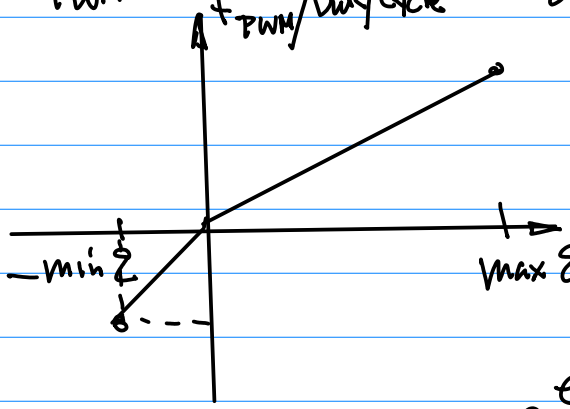


Fig. 7b.

PWM



Assume Duty Cycle 0% ~ 95%

f_{PWM} = 500 Hz.

GPIO controls Direction.

error (Angular Error)

Fig. 8

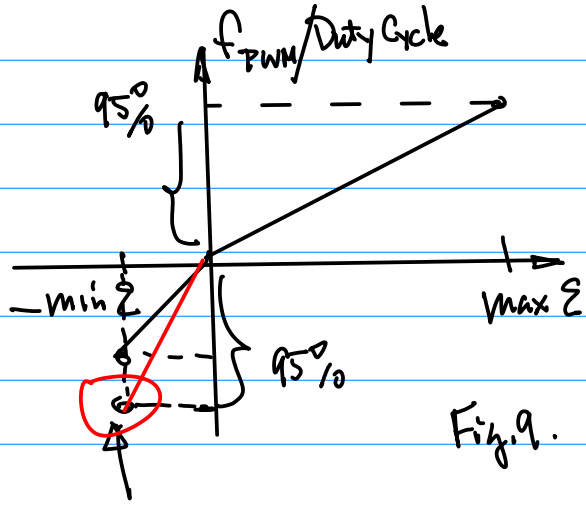


Fig. 9.

March 6 (Monday).

Note: 1° Homework, Due A week from the Wednesday (15th).

Brief Description of the Homework; use PWM to control the motor of your Choice, Motor can be a stepper motor, or other type of motor.

Example: Design methodology for P.I.D. Close Loop Control

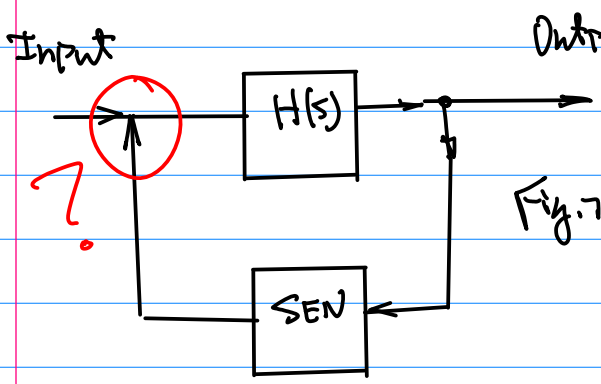


Fig. 7b. pp31.

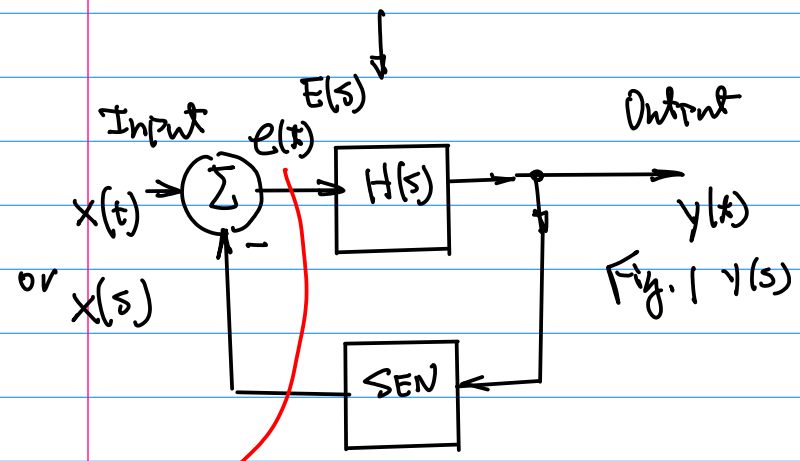


Fig. 1

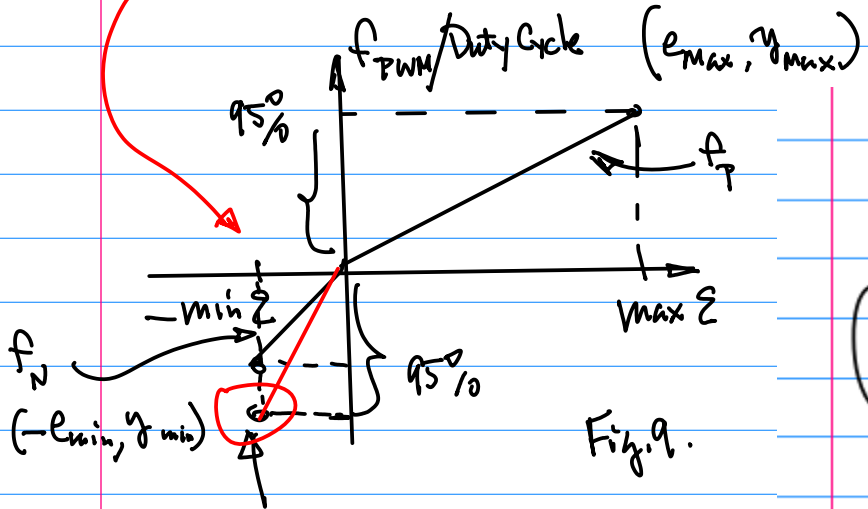


Fig. 9.

Design Objective: To find a Control Action to reduce the error, and eventually to minimize the error overtime.

Given: 1° Characteristic Curve of the Error Distribution. f_P, and f_N.

or $f_P(e)$, $f_N(e)$

2° Based on the feedback Loop, we would like to Correct the error(s) by Control Action(s).

Find $f_P(e)$ and $f_N(e)$ by using the following Technique.

$$\frac{y_2 - y_1}{x_2 - x_1} = \frac{y - y_1}{x - x_1} \quad \cdots (1)$$

(x_1, y_1), (x_2, y_2) are the pairs of the Line.

Note: Take this formulation.

$$y - y_1 = \frac{y_2 - y_1}{x_2 - x_1}(x - x_1)$$

2022S-101-note-part2-cmpe242-2022-05-9.pdf

pp17.

$$\frac{x - x_2}{y - y_2} = \frac{x_1 - x_2}{y_1 - y_2} = \frac{x_2 - x_1}{y_2 - y_1} \quad \cdots (3)$$

$$y = ax + b.$$

$$x - x_2 = \frac{x_1 - x_2}{y_1 - y_2}(y - y_2)$$

$$\frac{y_1 - y_2}{x_1 - x_2}(x - x_2) = y - y_2$$

From the Ref. (Lecture Notes, 2022S, pp.17)

$$y = \underbrace{\frac{y_1 - y_2}{x_1 - x_2}}_{b} x - \underbrace{\frac{y_1 - y_2}{x_1 - x_2} x_2 + y_2}_{c}$$

$ay + bx + c = 0$

Hence

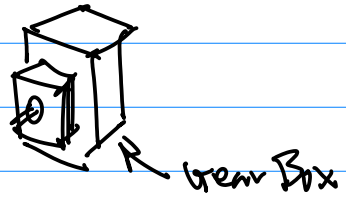
$$y = bx + c \quad \ldots (4)$$

where

$$b = \frac{y_2 - y_1}{x_2 - x_1} \quad \ldots (4\text{-}b)$$

$$c = -\frac{y_2 - y_1}{x_2 - x_1} x_2 + y_2 \quad \ldots (4\text{-}c)$$

Given the motor Configuration With OR Without Gear Box.



- Nema 8 Stepper Motor Bipolar w/ 64:1 Planetary Gearbox
- Nominal Voltage: 6 V
- Stall Current: 0.6 A
- Stall Torque: 127.5 oz.in
- Shaft Diameter: 6 mm

The **Nema 8 Stepper Motor Bipolar w/ 64:1 Planetary Gearbox** is an 8 stepper motor with 38 mm body and 0.6 A rated current, integrated a planetary gearbox of 64:1 gear ratio. It's a good solution to applications with limited space but need low speed and/or high torque.

Example for A quick Calculation:

Suppose $e = 109.75$, find the Characteristic Equation (4) And Corresponding PWM Control Signal.

Assuming. $f_{PWM} = 500$ Hz

Duty Cycle Range is illustrated in Fig.9. pp31.

$e_{max} = 1000$; $e_{min} = -800$ from Experiments.

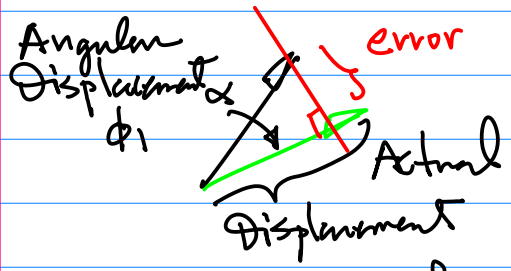


Fig.5 from pp.30.

Given the wheel Dimension R, (10", 12", etc).

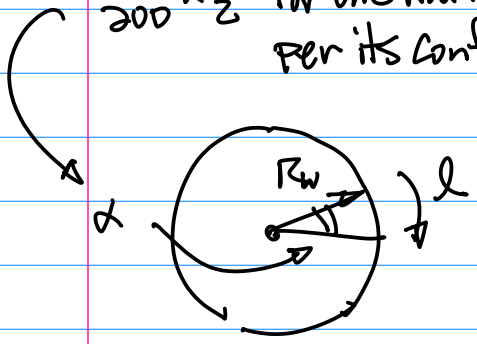
Find the error Displacement distance. for X No. of Pulses.

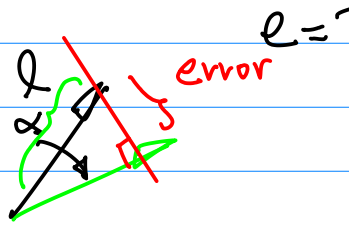
Suppose, $f_{PWM} = 500$ Hz, Duty = 50%, H.S.

given f_{PWM}, Duty Cycle → Given Stepper (F.S, H.S, etc) motor Config.

Angle.

$\frac{360}{200} \times \frac{1}{2}$ for the Half Step per its configuration.





Now, with 64:1 GearBox Reduction.

$\alpha' = \frac{1}{64}\alpha$, Reduction Ratio $= R_r$

$\alpha' = \frac{1}{r}\alpha$ ... (5).

To find physical displacement

$\because \alpha = \frac{l}{r}$ ... (6)

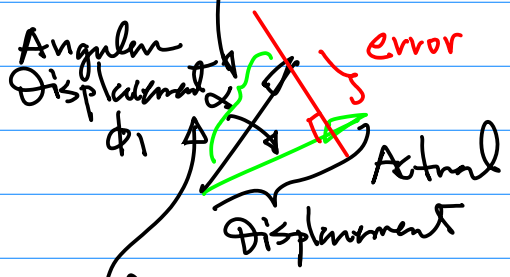
$l = \alpha \cdot r$

OR,

displacement $= \alpha' \cdot r_w$ ... (7)

"Wheel"

$dist = \alpha' \cdot r_w$ from Eqn (7)



Angle is from LSM303 (Deviation from the Desired Trajectory)

Therefore, the error.

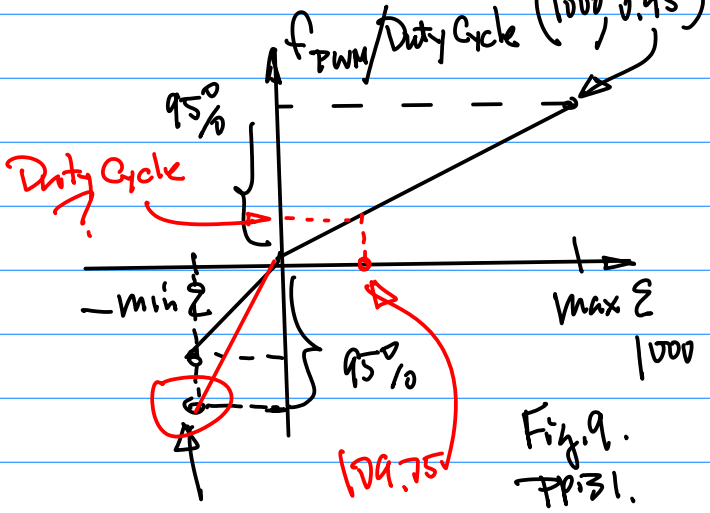


Fig.9. PPI31.

from Eqn(4)

$y = bx + c$

Slope from the Characteristic Equation.

Offset, from the given condition.

Where $x = 109.75$

Duty Cycle for Correction.