

CMPE242
Spring 2023

1/

Jan 25th (Wed), 2023

Organizational Meeting.

1^o Class Syllabus, "Greensheet". Make a good use of the O.H. But it will expire by the end of the Semester, By the end of

San José State University the class. 3^o.

Computer Engineering Department

CMPE 242 Embedded Hardware Systems, Section 1, S2023 Lab

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?](https://us04web.zoom.us/j/9841607683?pwd=UIA3aEk1TnV4bjNLQk5CQkw0dDk4UT09)

pwd=UIA3aEk1TnV4bjNLQk5CQkw0dDk4UT09 Meeting ID: 984 160 7683

Passcode: 121092

Room/Facility: Rm 268.

Access Form.

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^o. Nature of the Class

Hardware : Target Platform Selection:

NDA Jetson Nano

2gb

4gb

Broadcom

R-pi 3 b+,

Pi 4.

Software: 1^o Kernel Source Dist.

JetPack;

2^o Device Development SPI,

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.

I2C, PWM, 3^o Python,
C/C++

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?](https://us04web.zoom.us/j/9841607683?pwd=UIA3aEk1TnV4bjNLQk5CQkw0dDk4UT09)

pwd=UIA3aEk1TnV4bjNLQk5CQkw0dDk4UT09 Meeting ID: 984 160 7683 Passcode: 121092

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.

5^o GitHub

6^o. Homework/projects submission
on CANVAS.

Course Learning Outcomes (CLO) (Required)

[hualili / CMPE242-Embedded-Systems-](#)

Course Learning Objectives (CLO):

Publi

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

3D printer, CNC machines.



Motors

Stepper motors. NEMA 17

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 Pi 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
Spring 2023

Grading Information (Required)

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

The examination grades are given based on the written answer in exams; In-Person, In-Class. grades are given based on the work submitted, prototype system programming source code. The detailed rubrics for each homework 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 example software implementation counts 40%, report counts 20%, so the

Exams: In-Person, In-Class.

Submission To CANVAS.

Need Laptop & Prototype System in the Exam.

Alternative 1: Broadcom Raspberry Pi 3, 3B+, 4.

Determination of Grades

Jan 29 (Monday).

1. Homework, 0 pt. Honesty pledge.
Due this Wednesday, ON CANVAS.

Ref from the github

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

Example: Selection of Target platform.

Build Selection Matrix Below.

1. Architectural Aspects.



2. User Basis, Market Share.

3. OS Kernel Aspect: Linux / Unix.

4. Forward Looking: \rightarrow GPU \rightarrow

GP GPU (general Purpose) \rightarrow
AI/ML.

for Example: Jetson Nano

↓ Quad CPU: ARM.

↓ GPU (128 GPUs)

Quad ARM.

Discussion: I/O I/F for An Embedded System:

1° UART (Serial Communication)
Tx/Rx/GND

2° SPI. ~10Mbps Mini Com/Putty

~1Mbps Slow!

3° I2C { SDA SCK

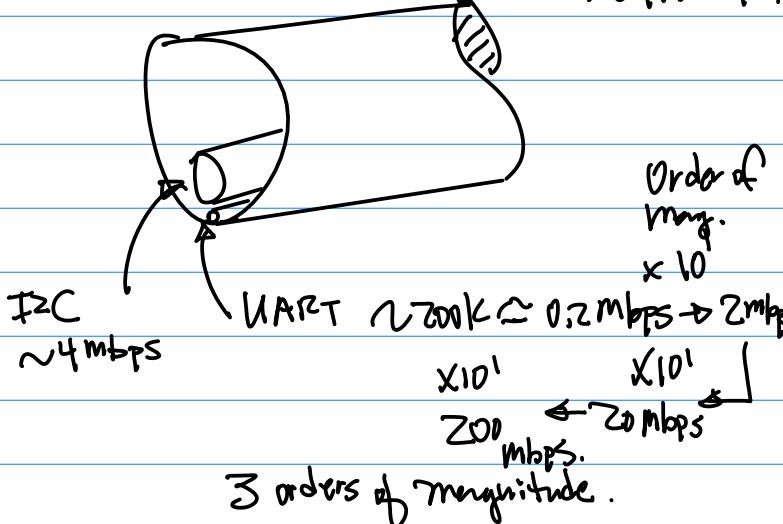
4° PWM

5° CAN

6° ADC

"3+1" MOSI
MISO
SCK
EN

SPI Bit Rate
~10Mbps.



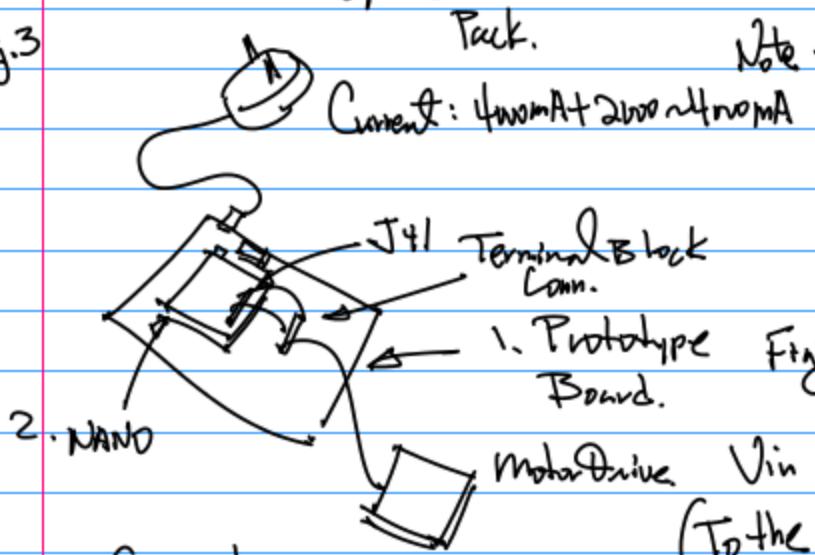
NAND, ~\$140
4gb.

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.

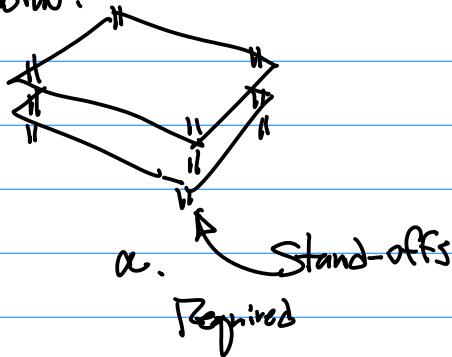
Note: 1° First Homework: "Hello, the
World" prototype System.

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

2° Prototype Board.

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

Dimension:



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. BreadBoard for Quick Prototyping.

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

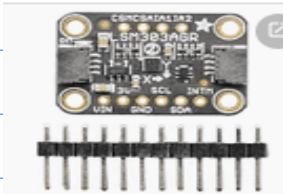
Up to 40mA

3. LED Assorted, Red, Yellow, etc.

Resistors.

Capacitors. 4.7nF (1nF to 10nF)

4^o LSM303 (I2C, SPI)



[Adafruit Industries](#)
[LLC 4413](#)

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

5^o I2C Approximity Sensor.

Note: I2C Mux for Multi-I2C Devices.

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

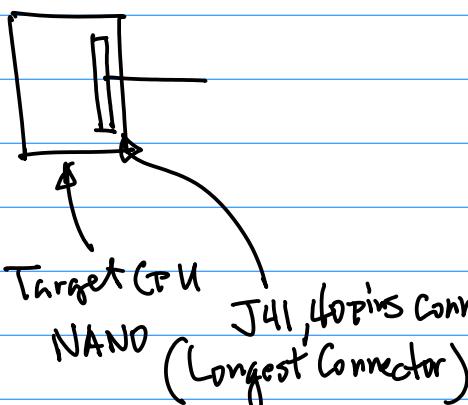
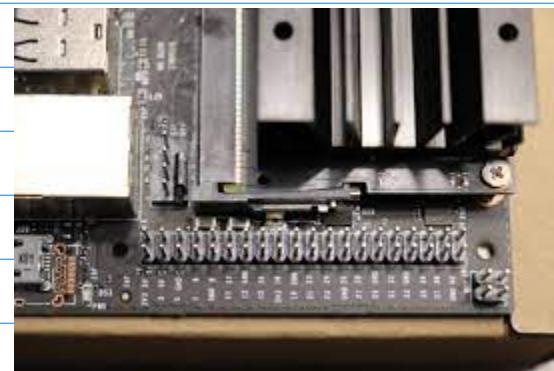


element14 Community



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
UART2_CTS	51(16)	D16	36 35	D19
		GND	34 33	D13
LCD_BL_PWM	168(12)	D12	32 31	D6
		GND	30 29	D5
		D1/D0_SC	28 27	DQ/D_S0
SP11_CS1	20(7)	D7	26 25	GND
SP11_CS0	19(8)	D8	24 23	D11
SP12_MISO	13(25)	D25	22 21	D9
		GND	20 19	D10
SP12_CS0	15(24)	D24	18 17	3.3V
SP12_CS1	232(23)	D23	16 15	D22
		GND	14 13	D27
DAP4_SCLK	79(18)	D18	12 11	D17
		RXD/D15	10 9	GND
		TXD/D14	8 7	D4
		GND	6 5	SCUD



Identify 3 pins
GND
Vcc
GPIO

J41_40 pins connector for Tens. (Ground Ref. Source)
<https://jetsonhacks.com/nvidia-jetson/>

NVIDIA Jetson Nano J41 F

Feb 6 (Monday).

Today's Topics: Design of

Prototype Board to Bring up the target platform (NANO).

Ref: 1^o Github

[CMPE242-Embedded-Systems- / 2022S / 2022S-103-SDcard-source-distribution-tool-chain-menuconfigu-2021-10-8.pdf](#)

2^o 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.

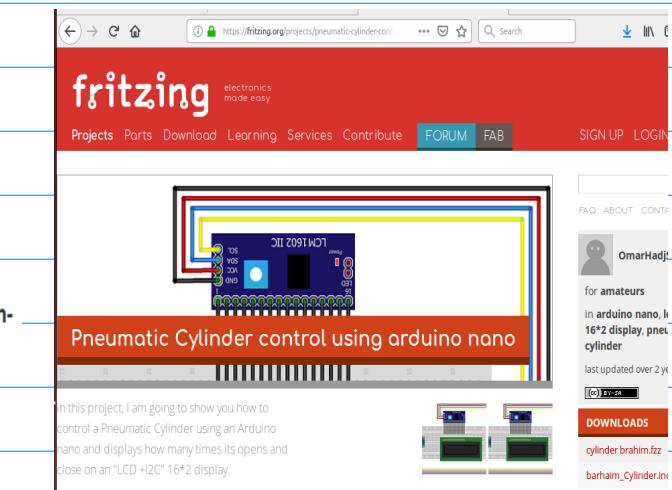
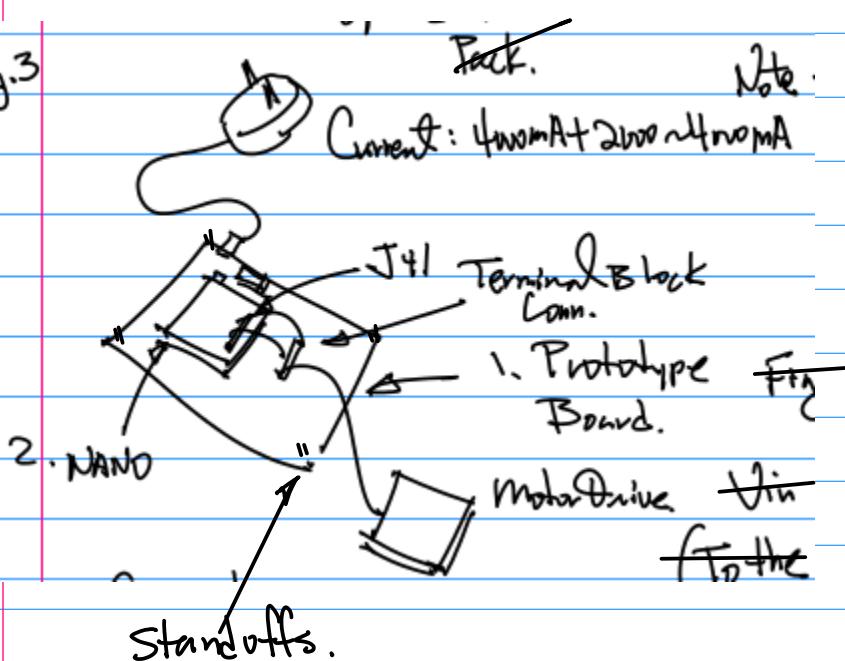


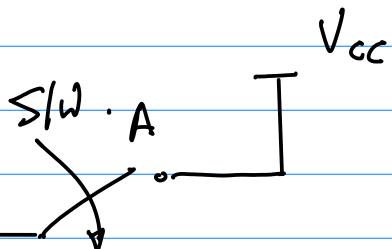
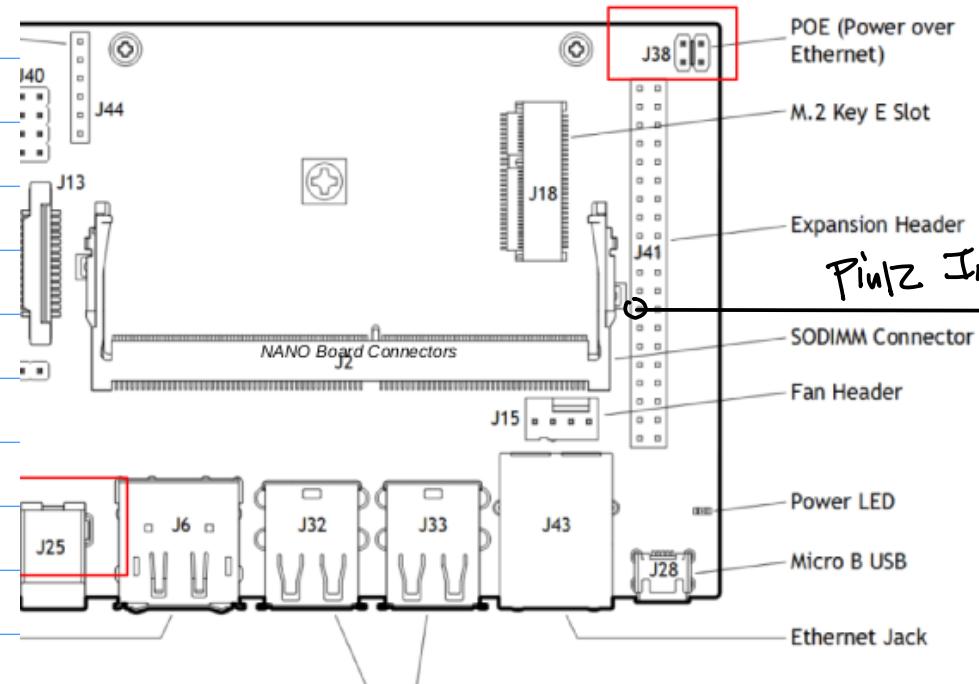
Fig.3



Example: Continuation ON GPIO I/F Design.

Design for the Input Testing.

Top View



↓ modify the Design
to Add R_1

Let $I_1 = 10 \text{ mA}$, find R_1 .

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

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

$$V_{CC} - R_1 \times 10 \times 10^{-3} = 0, R_1 = \frac{V_{CC}}{I_1} = \frac{3.3}{10^{-2}} = 330 \Omega$$

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

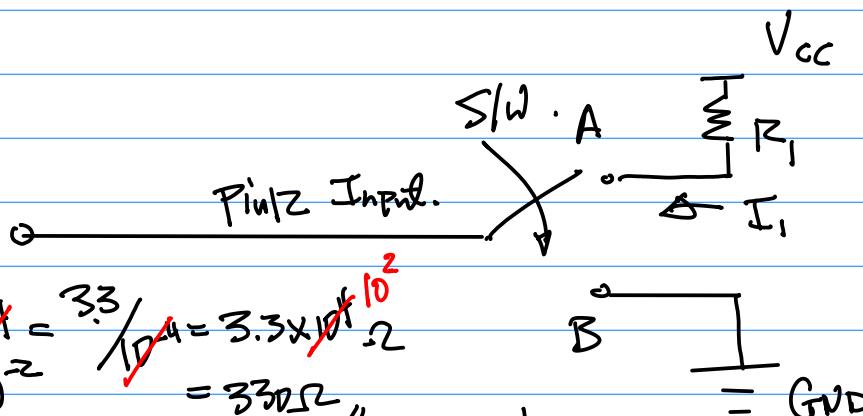
$$\text{make } I_2 = 10 \times 10^{-3} \text{ A}$$

Assume $V_{in} = V_{CC} = 3.3 \text{ V}$.

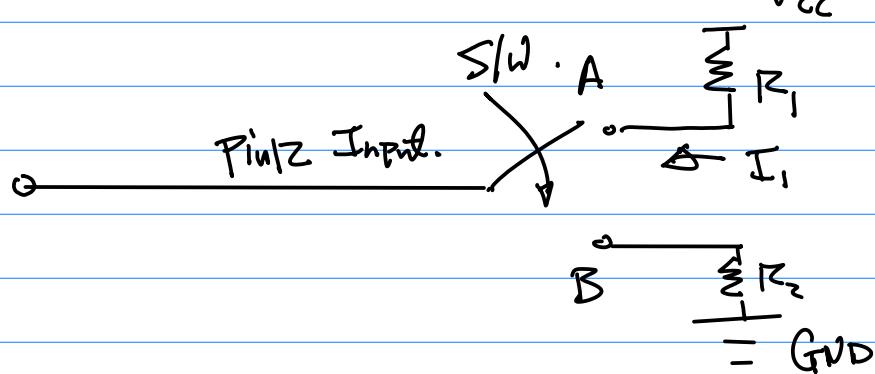
$$V_{in} - I_2 R_2 = 0, \text{ hence}$$

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

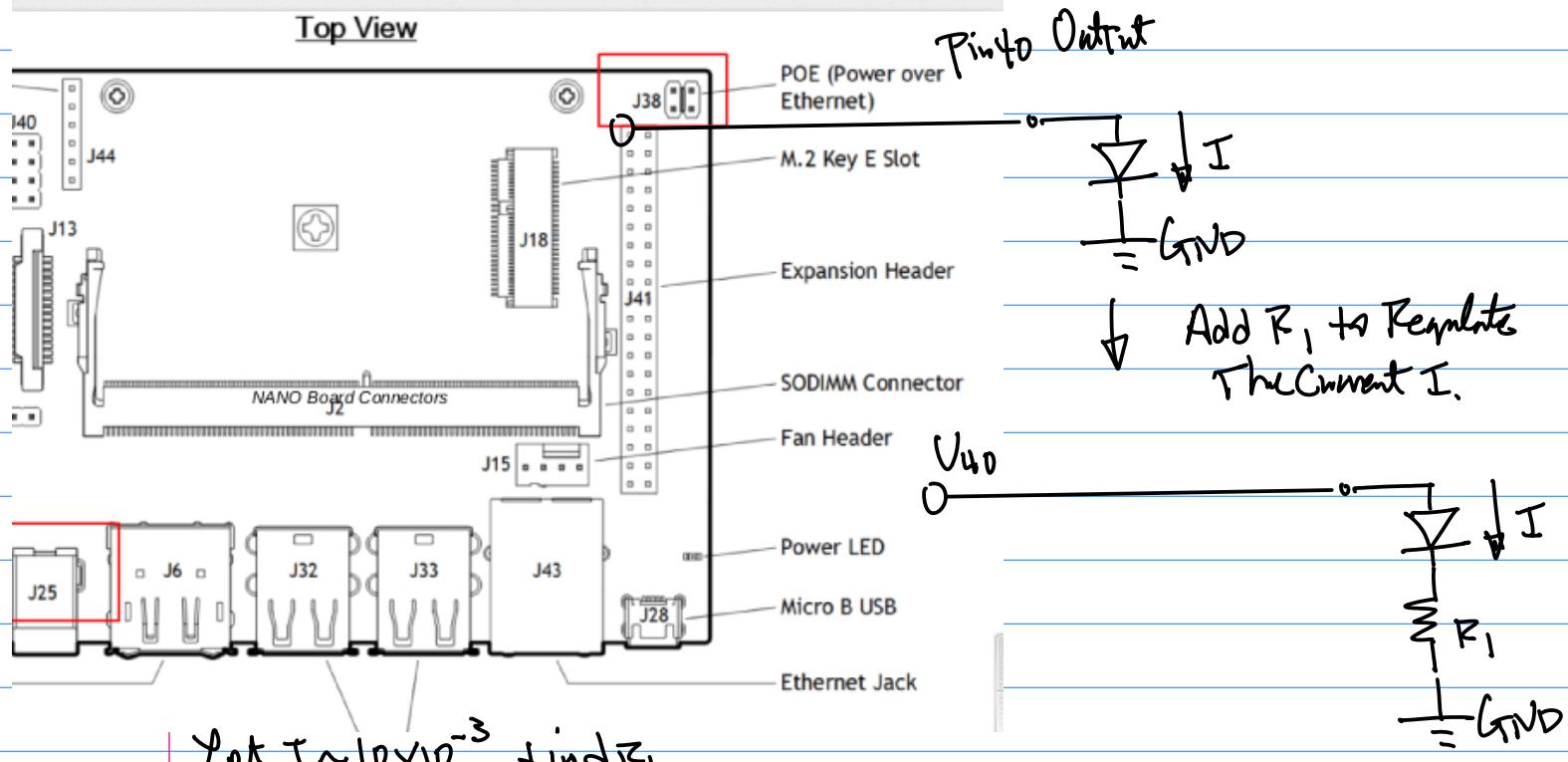
$$\therefore R_2 = \frac{V_{CC}}{I_2} = \frac{3.3}{10^{-2}} = 330 \Omega.$$



↓ modify the Design
By adding R_2



Output Testing Circuit.



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

$$V_{4D} - IR_1 = 0, \quad R_1 = V_{4D}/I$$

$$V_{4D} = V_{CC} = 3.3V.$$

$$I = 10 \times 10^{-3}$$

$$= 3.3 / 10^2 = 330\Omega.$$

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

Software Design:

Step 1. Prepare microSD Card, 16GB

or 32GB (for additional APP.).

Download pre-compiled Built Kernel Image from the Nvidia Site

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

Step 2. Download 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/>



Step 3. Take the MicroSD, insert it to your target platform.



the power up the NAND Board,
Config the Board By Following the
prompt.

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

Step 4. Init & Config. for GPIO
Driver.

Ref.

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

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

Pi and NANO are pin to pin compatible

[Jetson Nano GPIO - JetsonHacks](https://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. Up pin Connector is Compatible
With Raspberry Pie.

c. By Default, the Kernel Image (OS.)

Has already Configured GPIO
Driver, So, use Command Line

to Turn On/Off LED As follows.

\$echo 79 > /sys/class/gpio/export

Connect to the GPIO
Driver.

\$ echo out > /sys/class/gpio/gpio79/direction

Config the
GPIO As An
Output

\$echo 1 > /sys/class/gpio/gpio79/value

Set Output = 1

\$echo 0 > /sys/class/gpio/gpio79/value

Set Output = 0

\$echo 79 /sys/class/gpio/unexport

Release GPIO

\$cat /sys/kernel/debug/gpio

Homework : GPIO Testing. Due A
Week from Today.

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

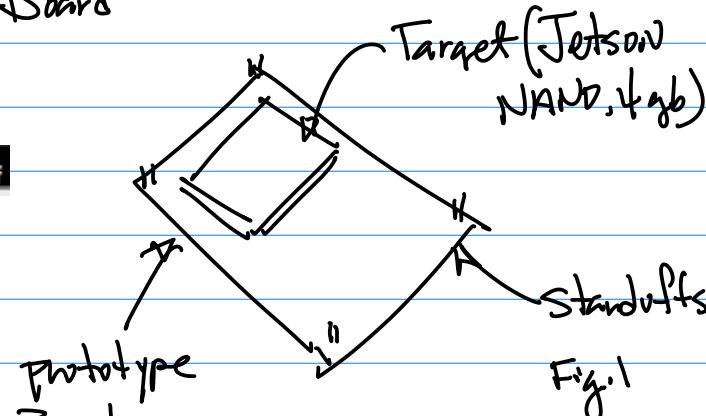
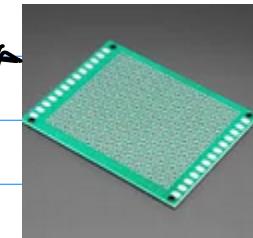


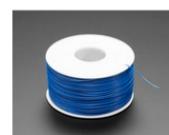
Fig.1



1° Note: 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 &
Wire - 200m 30AWG Blue
PRODUCT ID: 1446

Add to Cart

\$7.50
In stock

Fig.2

CMP-E242

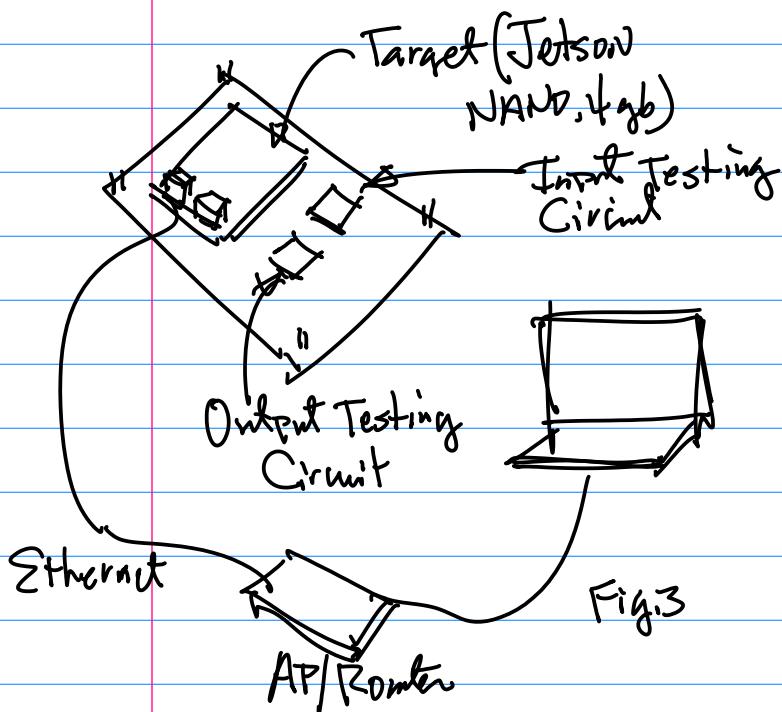
Spring 2023

(1).

4° Input/Output Testing Circuit

Has to be included,
please take photos of

(a) Entire System.



(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° Some code Listing.

a. "Template" Name of the program:

Coded by:

Date: Release
Version: 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
digikay.

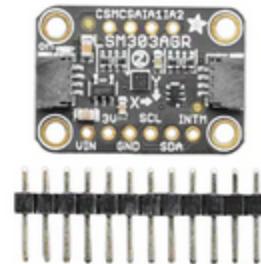
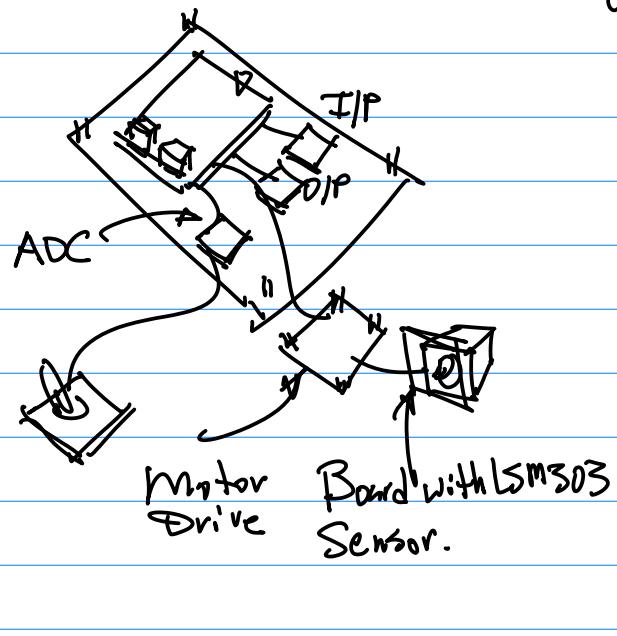


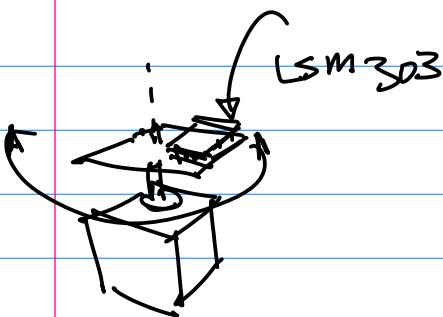
Fig.4

Fig.5



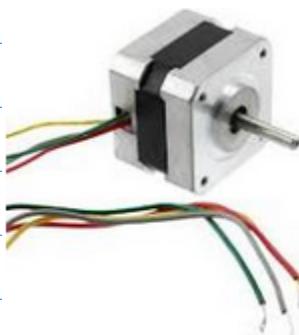
CMPE242
Spring 2023

11/



2. Motor.

Option 1. Stepper motor
NEMA 17



[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](#)

Option 2. Motor for E.V.

BLDC (Brushless D.C. motor)
for Scooters, eBike, or \$50 ~\$200



Option 3. Motors from Automobile Industry, from Amazon. \$35 ~\$200



Note: please form 2 person
Team by next week.



Human Control Interface

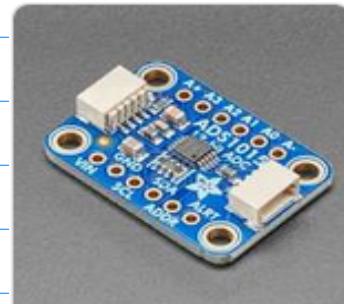
Devices: HandBar Controller for Scooters.



\$12.

[Ads · Shop ada 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 -

OR: Potential meter.



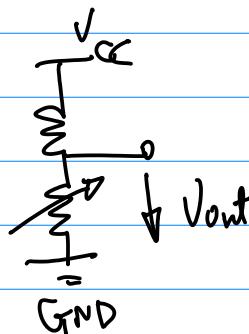
Bourns Inc.
PDB181-

\$1.47
Digi-Key

Feb 13 (Monday).

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

Example: I2C Based Sensor Interface
LM303





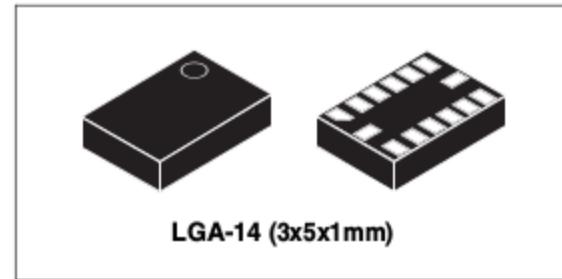
LSM303DLHC

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
- $\pm 2g/\pm 4g/\pm 8g/\pm 16g$ selectable full-scale
- 16 bit data output
- I²C 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



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 $\pm 2g/\pm 4g/\pm 8g/\pm 16g$ and a magnetic field full-scale of $\pm 1.3/\pm 1.9/\pm 2.5/\pm 4.0/\pm 4.7/\pm 5.6/\pm 8.1$ 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.

Target platform:

Jetson Nano. J41 Connector

has I²C pins.

2022S-108-LSM303DLHC.PDF

Ad

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

Ad

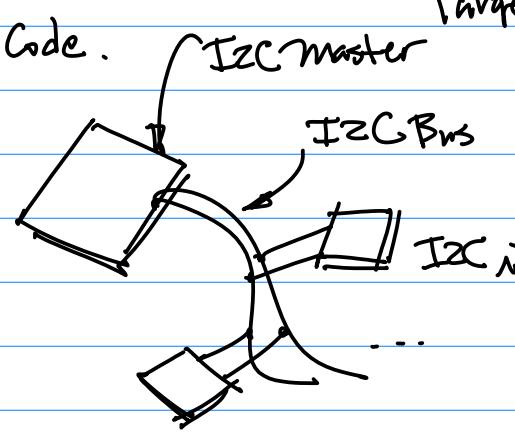


Fig. 1

Consider I²C Hardware Interface Design.

I²C { SDA : Serial Data, Bi-directional
2n8 Mbps

SCK : Serial Clock. Output from the Master

Note: A Typical I₂C "Slave Address"

takes 7 bits, $\rightarrow 2^7 = 128$

Device Address. \rightarrow Very Often,

I₂C Master Can Only Drive to
a few Devices, Such as 4 Devices.

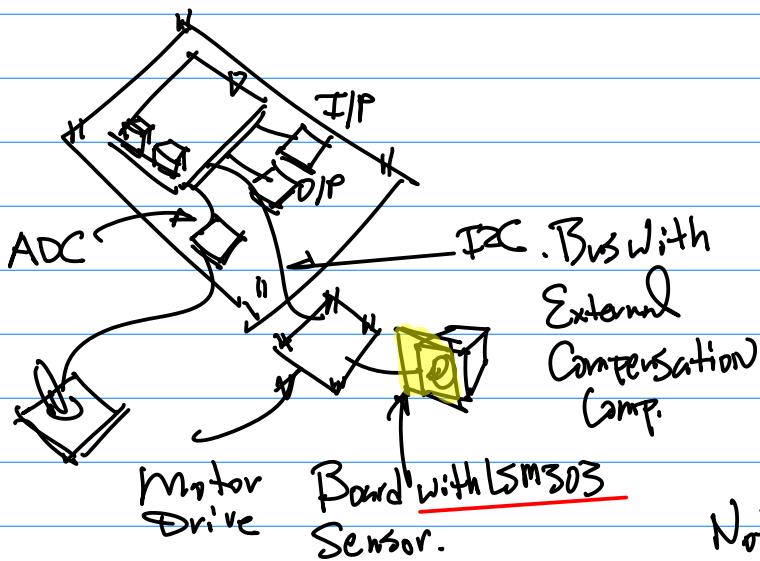
Ref:



2022S-108c-I2C-jetson-nano-2022-04-06 1... Add

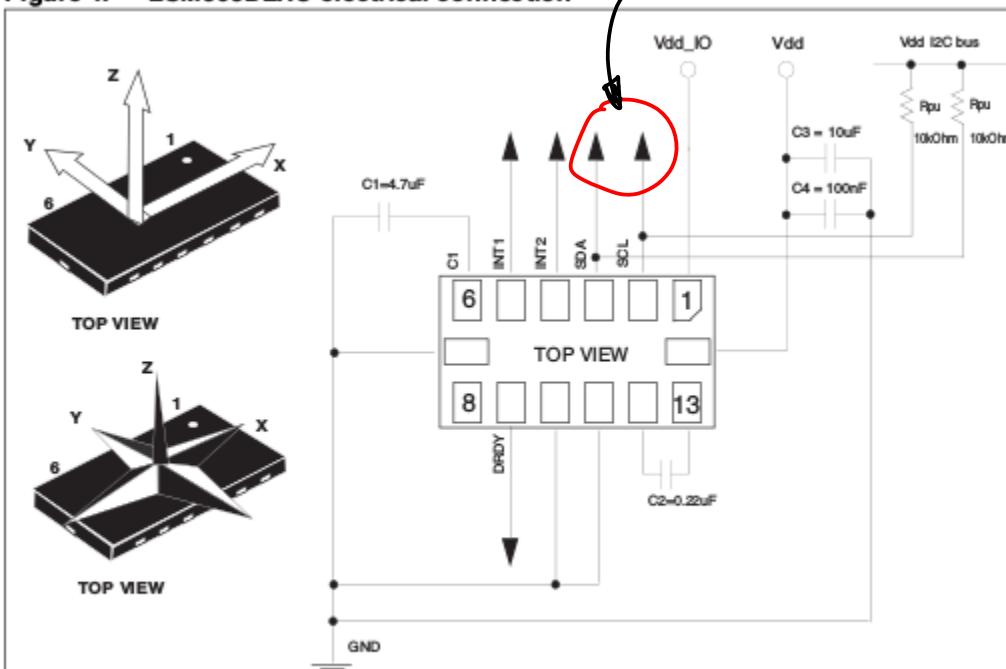
Describes the I₂C Command Line
Testing.

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



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

Figure 4. LSM303DLHC electrical connection



2° External Caps.
100nF (C4),
10uF (C3)
to form Low Pass
filter to Remove
"high freq." Noise.
And C1 (4.7uF),
C2 (0.22uF).

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

2° Command Line Testing.

Step 1. Config. I2C

Step 2. Install I2C Tools

Step 3. Install SMBus for Python Code

From Nvidia developer forum, the reference is provided here

<https://www.instructables.com/Raspberry-Pi-I2C-Python/>

Enable i2c:

Step 1. configure i2c

`sudo usermod -a -G i2c $USER`

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

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

`sudo apt-get install i2c-tools`

```
harry@harry-desktop:~$ sudo apt-get install i2c-tools
Reading package lists... Done
Building dependency tree
Reading state information... Done
i2c-tools is already the newest version (4.0-2).
```

Step 4. Reboot to make installed tools working, then check if any i2c is detected `i2cdetect -y 0`

```
harry@harry-desktop:~$ sudo usermod -a -G i2c $USER
[sudo] password for harry:
harry@harry-desktop:~$ i2cdetect -y 0
Warning: Can't use SMBus Quick Write command, will skip some addresses
 0:  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
10: 
11: 
12: 
13: 
14: 
15: 
16: 
17: 
18: 
19: 
20: 
21: 
22: 
23: 
24: 
25: 
26: 
27: 
28: 
29: 
30: 
31: 
32: 
33: 
34: 
35: 
36: 
37: 
38: 
39: 
40: 
41: 
42: 
43: 
44: 
45: 
46: 
47: 
48: 
49: 
50: 
51: 
52: 
53: 
54: 
55: 
56: 
57: 
58: 
59: 
60: 
61: 
62: 
63: 
64: 
65: 
66: 
67: 
68: 
69: 
70: 
71: 
72: 
73: 
74: 
75: 
76: 
77: 
78: 
79: 
80: 
81: 
82: 
83: 
84: 
85: 
86: 
87: 
88: 
89: 
90: 
91: 
92: 
93: 
94: 
95: 
96: 
97: 
98: 
99: 
100: 
101: 
102: 
103: 
104: 
105: 
106: 
107: 
108: 
109: 
110: 
111: 
112: 
113: 
114: 
115: 
116: 
117: 
118: 
119: 
120: 
121: 
122: 
123: 
124: 
125: 
126: 
127: 
128: 
129: 
130: 
131: 
132: 
133: 
134: 
135: 
136: 
137: 
138: 
139: 
140: 
141: 
142: 
143: 
144: 
145: 
146: 
147: 
148: 
149: 
150: 
151: 
152: 
153: 
154: 
155: 
156: 
157: 
158: 
159: 
160: 
161: 
162: 
163: 
164: 
165: 
166: 
167: 
168: 
169: 
170: 
171: 
172: 
173: 
174: 
175: 
176: 
177: 
178: 
179: 
180: 
181: 
182: 
183: 
184: 
185: 
186: 
187: 
188: 
189: 
190: 
191: 
192: 
193: 
194: 
195: 
196: 
197: 
198: 
199: 
200: 
201: 
202: 
203: 
204: 
205: 
206: 
207: 
208: 
209: 
210: 
211: 
212: 
213: 
214: 
215: 
216: 
217: 
218: 
219: 
220: 
221: 
222: 
223: 
224: 
225: 
226: 
227: 
228: 
229: 
230: 
231: 
232: 
233: 
234: 
235: 
236: 
237: 
238: 
239: 
240: 
241: 
242: 
243: 
244: 
245: 
246: 
247: 
248: 
249: 
250: 
251: 
252: 
253: 
254: 
255: 
256: 
257: 
258: 
259: 
260: 
261: 
262: 
263: 
264: 
265: 
266: 
267: 
268: 
269: 
270: 
271: 
272: 
273: 
274: 
275: 
276: 
277: 
278: 
279: 
280: 
281: 
282: 
283: 
284: 
285: 
286: 
287: 
288: 
289: 
290: 
291: 
292: 
293: 
294: 
295: 
296: 
297: 
298: 
299: 
300: 
301: 
302: 
303: 
304: 
305: 
306: 
307: 
308: 
309: 
310: 
311: 
312: 
313: 
314: 
315: 
316: 
317: 
318: 
319: 
320: 
321: 
322: 
323: 
324: 
325: 
326: 
327: 
328: 
329: 
330: 
331: 
332: 
333: 
334: 
335: 
336: 
337: 
338: 
339: 
340: 
341: 
342: 
343: 
344: 
345: 
346: 
347: 
348: 
349: 
350: 
351: 
352: 
353: 
354: 
355: 
356: 
357: 
358: 
359: 
360: 
361: 
362: 
363: 
364: 
365: 
366: 
367: 
368: 
369: 
370: 
371: 
372: 
373: 
374: 
375: 
376: 
377: 
378: 
379: 
380: 
381: 
382: 
383: 
384: 
385: 
386: 
387: 
388: 
389: 
390: 
391: 
392: 
393: 
394: 
395: 
396: 
397: 
398: 
399: 
400: 
401: 
402: 
403: 
404: 
405: 
406: 
407: 
408: 
409: 
410: 
411: 
412: 
413: 
414: 
415: 
416: 
417: 
418: 
419: 
420: 
421: 
422: 
423: 
424: 
425: 
426: 
427: 
428: 
429: 
430: 
431: 
432: 
433: 
434: 
435: 
436: 
437: 
438: 
439: 
440: 
441: 
442: 
443: 
444: 
445: 
446: 
447: 
448: 
449: 
450: 
451: 
452: 
453: 
454: 
455: 
456: 
457: 
458: 
459: 
460: 
461: 
462: 
463: 
464: 
465: 
466: 
467: 
468: 
469: 
470: 
471: 
472: 
473: 
474: 
475: 
476: 
477: 
478: 
479: 
480: 
481: 
482: 
483: 
484: 
485: 
486: 
487: 
488: 
489: 
490: 
491: 
492: 
493: 
494: 
495: 
496: 
497: 
498: 
499: 
500: 
501: 
502: 
503: 
504: 
505: 
506: 
507: 
508: 
509: 
510: 
511: 
512: 
513: 
514: 
515: 
516: 
517: 
518: 
519: 
520: 
521: 
522: 
523: 
524: 
525: 
526: 
527: 
528: 
529: 
530: 
531: 
532: 
533: 
534: 
535: 
536: 
537: 
538: 
539: 
540: 
541: 
542: 
543: 
544: 
545: 
546: 
547: 
548: 
549: 
550: 
551: 
552: 
553: 
554: 
555: 
556: 
557: 
558: 
559: 
560: 
561: 
562: 
563: 
564: 
565: 
566: 
567: 
568: 
569: 
570: 
571: 
572: 
573: 
574: 
575: 
576: 
577: 
578: 
579: 
580: 
581: 
582: 
583: 
584: 
585: 
586: 
587: 
588: 
589: 
590: 
591: 
592: 
593: 
594: 
595: 
596: 
597: 
598: 
599: 
600: 
601: 
602: 
603: 
604: 
605: 
606: 
607: 
608: 
609: 
610: 
611: 
612: 
613: 
614: 
615: 
616: 
617: 
618: 
619: 
620: 
621: 
622: 
623: 
624: 
625: 
626: 
627: 
628: 
629: 
630: 
631: 
632: 
633: 
634: 
635: 
636: 
637: 
638: 
639: 
640: 
641: 
642: 
643: 
644: 
645: 
646: 
647: 
648: 
649: 
650: 
651: 
652: 
653: 
654: 
655: 
656: 
657: 
658: 
659: 
660: 
661: 
662: 
663: 
664: 
665: 
666: 
667: 
668: 
669: 
670: 
671: 
672: 
673: 
674: 
675: 
676: 
677: 
678: 
679: 
680: 
681: 
682: 
683: 
684: 
685: 
686: 
687: 
688: 
689: 
690: 
691: 
692: 
693: 
694: 
695: 
696: 
697: 
698: 
699: 
700: 
701: 
702: 
703: 
704: 
705: 
706: 
707: 
708: 
709: 
710: 
711: 
712: 
713: 
714: 
715: 
716: 
717: 
718: 
719: 
720: 
721: 
722: 
723: 
724: 
725: 
726: 
727: 
728: 
729: 
730: 
731: 
732: 
733: 
734: 
735: 
736: 
737: 
738: 
739: 
740: 
741: 
742: 
743: 
744: 
745: 
746: 
747: 
748: 
749: 
750: 
751: 
752: 
753: 
754: 
755: 
756: 
757: 
758: 
759: 
760: 
761: 
762: 
763: 
764: 
765: 
766: 
767: 
768: 
769: 
770: 
771: 
772: 
773: 
774: 
775: 
776: 
777: 
778: 
779: 
779: 
780: 
781: 
782: 
783: 
784: 
785: 
786: 
787: 
788: 
789: 
790: 
791: 
792: 
793: 
794: 
795: 
796: 
797: 
798: 
799: 
800: 
801: 
802: 
803: 
804: 
805: 
806: 
807: 
808: 
809: 
810: 
811: 
812: 
813: 
814: 
815: 
816: 
817: 
818: 
819: 
820: 
821: 
822: 
823: 
824: 
825: 
826: 
827: 
828: 
829: 
830: 
831: 
832: 
833: 
834: 
835: 
836: 
837: 
838: 
839: 
840: 
841: 
842: 
843: 
844: 
845: 
846: 
847: 
848: 
849: 
850: 
851: 
852: 
853: 
854: 
855: 
856: 
857: 
858: 
859: 
860: 
861: 
862: 
863: 
864: 
865: 
866: 
867: 
868: 
869: 
870: 
871: 
872: 
873: 
874: 
875: 
876: 
877: 
878: 
879: 
880: 
881: 
882: 
883: 
884: 
885: 
886: 
887: 
888: 
889: 
889: 
890: 
891: 
892: 
893: 
894: 
895: 
896: 
897: 
898: 
899: 
900: 
901: 
902: 
903: 
904: 
905: 
906: 
907: 
908: 
909: 
910: 
911: 
912: 
913: 
914: 
915: 
916: 
917: 
918: 
919: 
920: 
921: 
922: 
923: 
924: 
925: 
926: 
927: 
928: 
929: 
930: 
931: 
932: 
933: 
934: 
935: 
936: 
937: 
938: 
939: 
940: 
941: 
942: 
943: 
944: 
945: 
946: 
947: 
948: 
949: 
950: 
951: 
952: 
953: 
954: 
955: 
956: 
957: 
958: 
959: 
960: 
961: 
962: 
963: 
964: 
965: 
966: 
967: 
968: 
969: 
970: 
971: 
972: 
973: 
974: 
975: 
976: 
977: 
978: 
979: 
980: 
981: 
982: 
983: 
984: 
985: 
986: 
987: 
988: 
989: 
990: 
991: 
992: 
993: 
994: 
995: 
996: 
997: 
998: 
999: 
1000: 
1001: 
1002: 
1003: 
1004: 
1005: 
1006: 
1007: 
1008: 
1009: 
1010: 
1011: 
1012: 
1013: 
1014: 
1015: 
1016: 
1017: 
1018: 
1019: 
1020: 
1021: 
1022: 
1023: 
1024: 
1025: 
1026: 
1027: 
1028: 
1029: 
1030: 
1031: 
1032: 
1033: 
1034: 
1035: 
1036: 
1037: 
1038: 
1039: 
1040: 
1041: 
1042: 
1043: 
1044: 
1045: 
1046: 
1047: 
1048: 
1049: 
1050: 
1051: 
1052: 
1053: 
1054: 
1055: 
1056: 
1057: 
1058: 
1059: 
1060: 
1061: 
1062: 
1063: 
1064: 
1065: 
1066: 
1067: 
1068: 
1069: 
1070: 
1071: 
1072: 
1073: 
1074: 
1075: 
1076: 
1077: 
1078: 
1079: 
1080: 
1081: 
1082: 
1083: 
1084: 
1085: 
1086: 
1087: 
1088: 
1089: 
1090: 
1091: 
1092: 
1093: 
1094: 
1095: 
1096: 
1097: 
1098: 
1099: 
1100: 
1101: 
1102: 
1103: 
1104: 
1105: 
1106: 
1107: 
1108: 
1109: 
1110: 
1111: 
1112: 
1113: 
1114: 
1115: 
1116: 
1117: 
1118: 
1119: 
1120: 
1121: 
1122: 
1123: 
1124: 
1125: 
1126: 
1127: 
1128: 
1129: 
1130: 
1131: 
1132: 
1133: 
1134: 
1135: 
1136: 
1137: 
1138: 
1139: 
1140: 
1141: 
1142: 
1143: 
1144: 
1145: 
1146: 
1147: 
1148: 
1149: 
1150: 
1151: 
1152: 
1153: 
1154: 
1155: 
1156: 
1157: 
1158: 
1159: 
1160: 
1161: 
1162: 
1163: 
1164: 
1165: 
1166: 
1167: 
1168: 
1169: 
1170: 
1171: 
1172: 
1173: 
1174: 
1175: 
1176: 
1177: 
1178: 
1179: 
1180: 
1181: 
1182: 
1183: 
1184: 
1185: 
1186: 
1187: 
1188: 
1189: 
1190: 
1191: 
1192: 
1193: 
1194: 
1195: 
1196: 
1197: 
1198: 
1199: 
1200: 
1201: 
1202: 
1203: 
1204: 
1205: 
1206: 
1207: 
1208: 
1209: 
1210: 
1211: 
1212: 
1213: 
1214: 
1215: 
1216: 
1217: 
1218: 
1219: 
1220: 
1221: 
1222: 
1223: 
1224: 
1225: 
1226: 
1227: 
1228: 
1229: 
1230: 
1231: 
1232: 
1233: 
1234: 
1235: 
1236: 
1237: 
1238: 
1239: 
1240: 
1241: 
1242: 
1243: 
1244: 
1245: 
1246: 
1247: 
1248: 
1249: 
1250: 
1251: 
1252: 
1253: 
1254: 
1255: 
1256: 
1257: 
1258: 
1259: 
1260: 
1261: 
1262: 
1263: 
1264: 
1265: 
1266: 
1267: 
1268: 
1269: 
1270: 
1271: 
1272: 
1273: 
1274: 
1275: 
1276: 
1277: 
1278: 
1279: 
1280: 
1281: 
1282: 
1283: 
1284: 
1285: 
1286: 
1287: 
1288: 
1289: 
1290: 
1291: 
1292: 
1293: 
1294: 
1295: 
1296: 
1297: 
1298: 
1299: 
1300: 
1301: 
1302: 
1303: 
1304: 
1305: 
1306: 
1307: 
1308: 
1309: 
1310: 
1311: 
1312: 
1313: 
1314: 
1315: 
1316: 
1317: 
1318: 
1319: 
1320: 
1321: 
1322: 
1323: 
1324: 
1325: 
1326: 
1327: 
1328: 
1329: 
1330: 
1331: 
1332: 
1333: 
1334: 
1335: 
1336: 
1337: 
1338: 
1339: 
13310: 
13311: 
13312: 
13313: 
13314: 
13315: 
13316: 
13317: 
13318: 
13319: 
13320: 
13321: 
13322: 
13323: 
13324: 
13325: 
13326: 
13327: 
13328: 
13329: 
13330: 
13331: 
13332: 
13333: 
13334: 
13335: 
13336: 
13337: 
13338: 
13339: 
13340: 
13341: 
13342: 
13343: 
13344: 
13345: 
13346: 
13347: 
13348: 
13349: 
13350: 
13351: 
13352: 
13353: 
13354: 
13355: 
13356: 
13357: 
13358: 
13359: 
13360: 
13361: 
13362: 
13363: 
13364: 
13365: 
13366: 
13367: 
13368: 
13369: 
13370: 
13371: 
13372: 
13373: 
13374: 
13375: 
13376: 
13377: 
13378: 
13379: 
13380: 
13381: 
13382: 
13383: 
13384: 
13385: 
13386: 
13387: 
13388: 
13389: 
13390: 
13391: 
13392: 
13393: 
13394: 
13395: 
13396: 
13397: 
13398: 
13399: 
133100: 
133101: 
133102: 
133103: 
133104: 
133105: 
133106: 
133107: 
133108: 
133109: 
133110: 
133111: 
133112: 
133113: 
133114: 
133115: 
133116: 
133117: 
133118: 
133119: 
133120: 
133121: 
133122: 
133123: 
133124: 
133125: 
133126: 
133127: 
133128: 
133129: 
133130: 
133131: 
133132: 
133133: 
133134: 
133135: 
133136: 
133137: 
133138: 
133139: 
133140: 
133141: 
133142: 
133143: 
133144: 
133145: 
133146: 
133147: 
133148: 
133149: 
133150: 
133151: 
133152: 
133153: 
133154: 
133155: 
133156: 
133157: 
133158: 
133159: 
133160: 
133161: 
133162: 
133163: 
133164: 
133165: 
133166: 
133167: 
133168: 
133169: 
133170: 
133171: 
133172: 
133173: 
133174: 
133175: 
133176: 
133177: 
133178: 
133179: 
133180: 
133181: 
133182: 
133183: 
133184: 
133185: 
133186: 
133187: 
133188: 
133189: 
133190: 
133191: 
133192: 
133193: 
133194: 
133195: 
133196: 
133197: 
133198: 
133199: 
133200: 
133201: 
133202: 
133203: 
133204: 
133205: 
133206: 
133207: 
133208: 
133209: 
133210: 
133211: 
133212: 
133213: 
133214: 
133215: 
133216: 
133217: 
133218: 
133219: 
133220: 
133221: 
133222: 
133223: 
133224: 
133225: 
133226: 
133227: 
133228: 
133229: 
133230: 
133231: 
133232: 
133233: 
133234: 
133235: 
133236: 
133237: 
133238: 
133239: 
133240: 
133241: 
133242: 
133243: 
133244: 
133245: 
133246: 
133247: 
133248: 
133249: 
133250: 
133251: 
133252: 
133253: 
133254: 
133255: 
133256: 
133257: 
133258: 
133259: 
133260: 
133261: 
133262: 
133263: 
133264: 
133265: 
133266: 
133267: 
133268: 
133269: 
133270: 
133271: 
133272: 
133273: 
133274: 
133275: 
133276: 
133277: 
133278: 
133279: 
133280: 
133281: 
133282: 
133283: 
133284: 
133285: 
133286: 
133287: 
133288: 
133289: 
133290: 
133291: 
133292: 
133293: 
133294: 
133295: 
133296: 
133297: 
133298: 
133299: 
133300: 
133301: 
133302: 
133303: 
133304: 
133305: 
133306: 
133307: 
133308: 
133309: 
133310: 
133311: 
133312: 
133313: 
133314: 
133315: 
133316: 
133317: 
133318: 
133319: 
133320: 
133321: 
133322: 
133323: 
133324: 
133325: 
133326: 
133327: 
133328: 
133329: 
133330: 
133331: 
133332: 
133333: 
133334: 
133335: 
133336: 
133337: 
133338: 
133339: 
133340: 
133341: 
133342: 
133343: 
133344: 
133345: 
133346: 
133347: 
133348: 
133349: 
133350: 
133351: 
133352: 
133353: 
133354: 
133355: 
133356: 
133357: 
133358: 
133359: 
133360: 
133361: 
133362: 
133363: 
133364: 
133365: 
133366: 
133367: 
133368: 
133369: 
133370: 
133371: 
133372: 
133373: 
133374: 
133375: 
133376: 
133377: 
133378: 
133379: 
133380: 
133381: 
133382: 
133383: 
133384: 
133385: 
133386: 
133387: 
133388: 
133389: 
133390: 
133391: 
133392: 
133393: 
133394: 
133395: 
133396: 
133397: 
133398: 
133399: 
133400: 
133401: 
133402: 
133403: 
133404: 
133405: 
133406: 
133407: 
133408: 
133409: 
133410: 
133411: 
133412: 
133413: 
133414: 
133415: 
133416: 
133417: 
133418: 
133419: 
133420: 
133421: 
133422: 
133423: 
133424: 
133425: 
133426: 
133427: 
133428: 
133429: 
133430: 
133431: 
133432: 
133433: 
133434: 
133435: 
133436: 
133437: 
133438: 
133439: 
133440: 
133441: 
133442: 
133443: 
133444: 
133445: 
133446: 
133447: 
133448: 
133449: 
133450: 
133451: 
133452
```

Note: 1. Magnetometer.

With Reference to the North Pole. & Accelerometer X, Y, Z-Axis.

3D Accelerometer and 3D Magnetometer LMS303

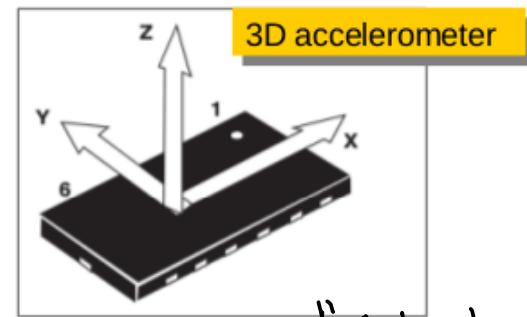
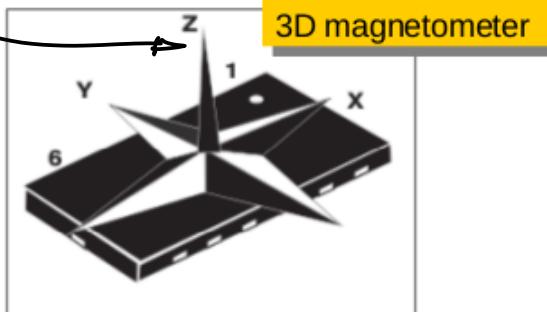


Table 9

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

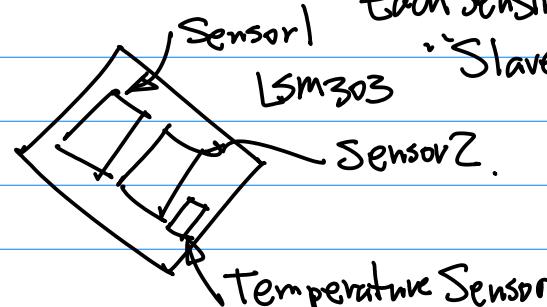
Reference: Table 9, pp 19, from
LMS303 datasheet

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"

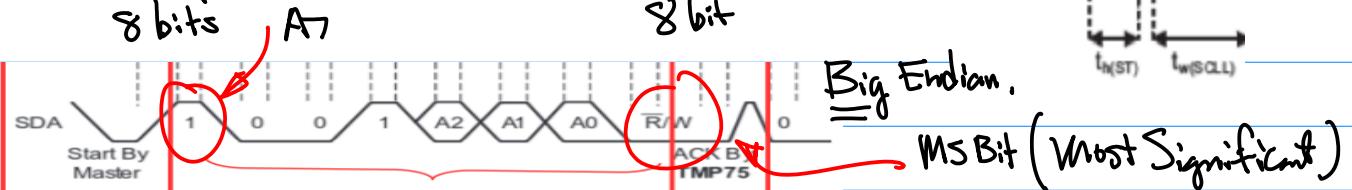
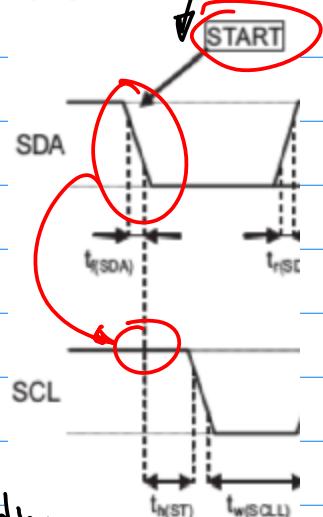
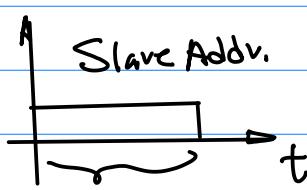
Harry Li, Ph.D. April 2015



Each Sensing Unit has its own I²C Address.

"Slave" Address.

Note: 2.



Big Endian.

MS Bit (Most Significant)

Feb 15 (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.
\$ icdetct.

To Demonstrate the NAND is properly configured with i2ctools installed, And sm-bus package for python programming is imported.

3° Hardware Implementation to Connect LSM303 Sensor to the NAND.

Using connectors is recommended;

4° Write A python Code to read Sensor Output:

(a. Magnetometer,
b. acceleration.)

Print the Sensor input on the Console.

then, make your program work with gpio Code.

when gpio input = "1", Read/Print

Magnetometer 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 NAND & 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: Step1.

Hardware Design → Identify the I²C pins on NAND.
Since I²C Sensor module is ready

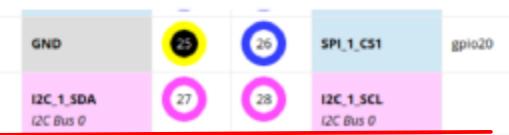
I²C parts
Ref: ZY4 github.

I2C on Jetson Nano J41 Header

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

connected to assigned. By power) are assigned for functions are in a different device

I2C, Pin 3 and 5 can be utilized for I2C interface design.



Sysfs GPIO	Name	Pin	Pin	Name	Sysfs GPIO
	3.3 VDC Power	1	2	5.0 VDC Power	
	I2C_2_SDA I2C Bus 1	3	4	5.0 VDC Power	
	I2C_2_SCL I2C Bus 1	5		GND	
gpio216	AUDIO_MCLK	7	8	UART_2_TX /dev/ttyHS1	
	GND	9	10	UART_2_RX /dev/ttyHS2	
gpio50	UART_2_RTS	11	12	I2S_4_SCLK	gpio79

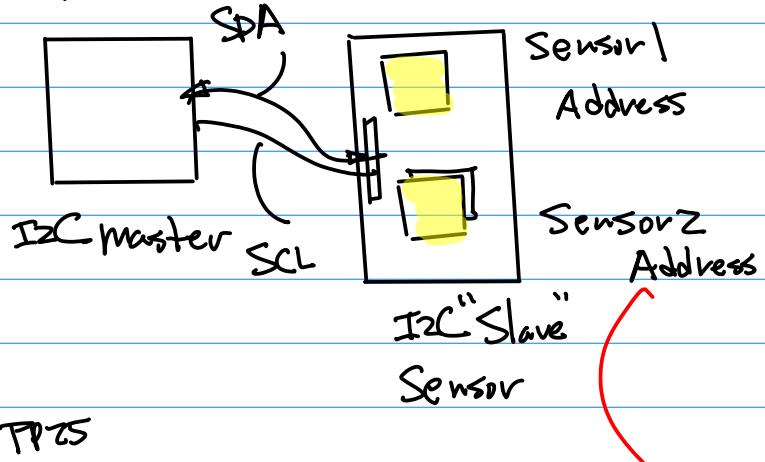
Connectivity table

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

Step 2. Set up & Verify the I2C Drivers By Using Command Line, pp15.

Step 3. Design methodology.

NAND



PP25

Table 14. SAD+Read/Write patterns

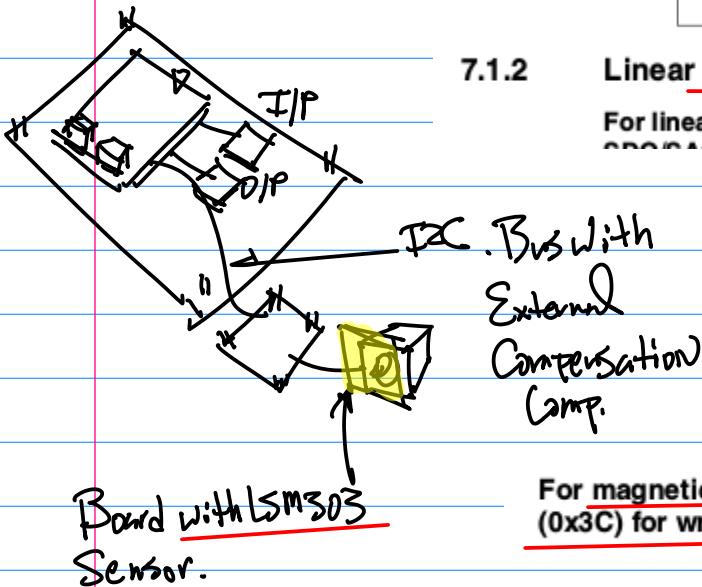
Command	SAD[6:1]	SAD[0] = SA0	R/W	SAD+R/W
Read	001100	0	1	00110001 (31h)
Write	001100	0	0	00110000 (30h)
Read	001100	1	1	00110011 (33h)
Write	001100	1	0	00110010 (32h)

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



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	Meg.	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"

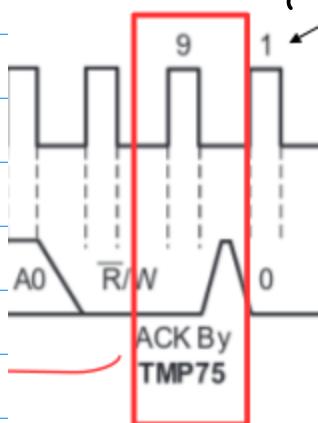
Additional Addr. for Special purpose Registers(s). Such as CONTROL Configuration Registers. Stop.

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

Cmpe422

Spring 2023

20

Feb 20 (Monday) :

Prepare for the Homework on

LSM303 Interface. March 1st 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
(OS.) Image. See P.P.T. for

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)

b° 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:

Cmpe422_LSM303_HW-

First_LastName_SID(4Digit).
Zip.

Example: Continuation of I2C Sensor
Interface.

Step 1. Identification
of Jetson
NANO pins.

J40-27,

J40-28

for I2C

Connect the Hardware
Sensor.

↓
Step 2. Verify Config Devile 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 Devile Driver
Configuration.

↓

Step 3. Command Line Instruction to Config the I₂C Sensor, And to Read the Sensor Data.

In order to perform that task at Step 3. We Need to

Understand I₂C protocol ;

I₂C Sensor I/F with its Datasheet.

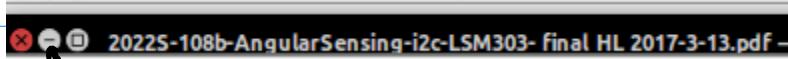
Perform Init & Config to Read Data from LSM303.

Slave Address.

(Sensor)
(Magnet) 0x3C Write

↓
SUB Control Register

↓
Step 4. Put All Command Line Instructions from Step 2+3 into One Python Program.

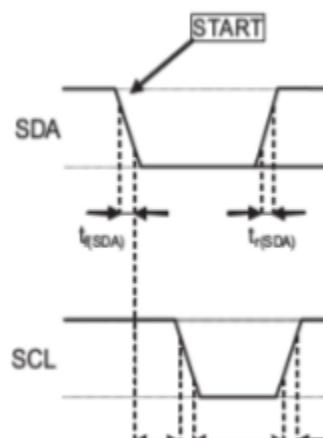
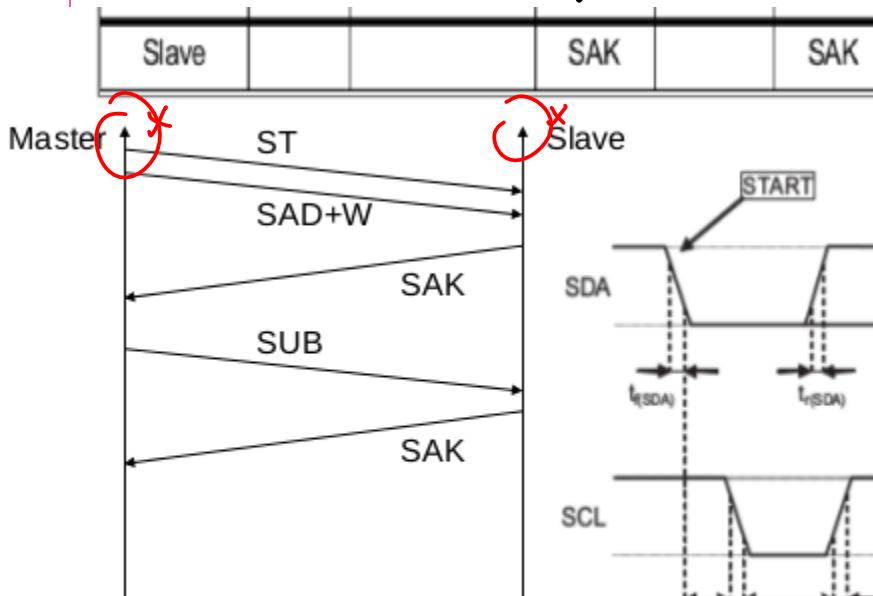


PPT. from the github

LSM303 Datasheet

2022S-108-LSM303DLHC.PDF -

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



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

P.I. 3b.

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
1:0 =

Control Register A for Magnetic Field

2³ 8 Bit Reg.

$$\text{CRA_REG_M}[1:0] = 0\phi; \\ = \phi \quad \checkmark$$

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

$$\text{CRA_REG_M}[4:2] = 0x4$$

then, enable the Temperature Sensing by setting

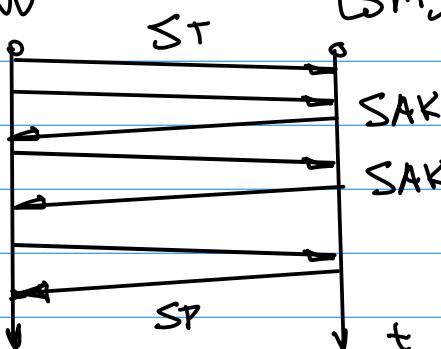
it to 1. Hence. $\text{CRA_REG_M}[7:6] = 1001,0000 = 0x90$;

NAND (Sm303(Mag)).

SAD+W: 0x3C

SUB: 0x00

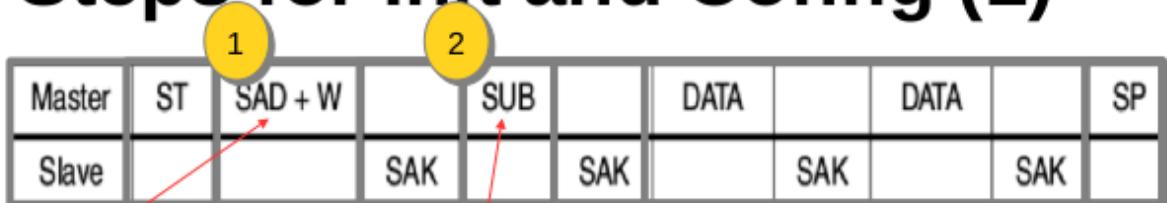
0x90



Ref. Confirms our discussion.

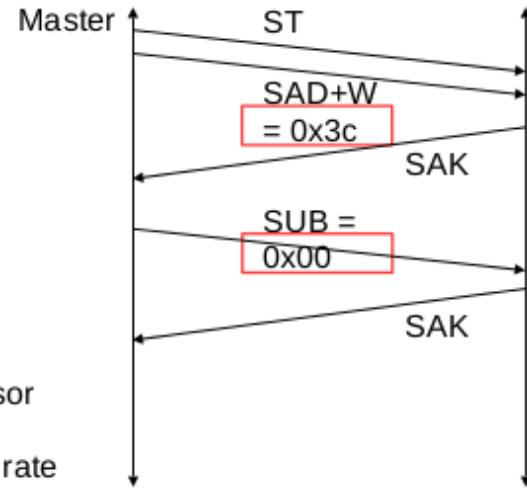
PP.7.

Steps for Init and Config (1)



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

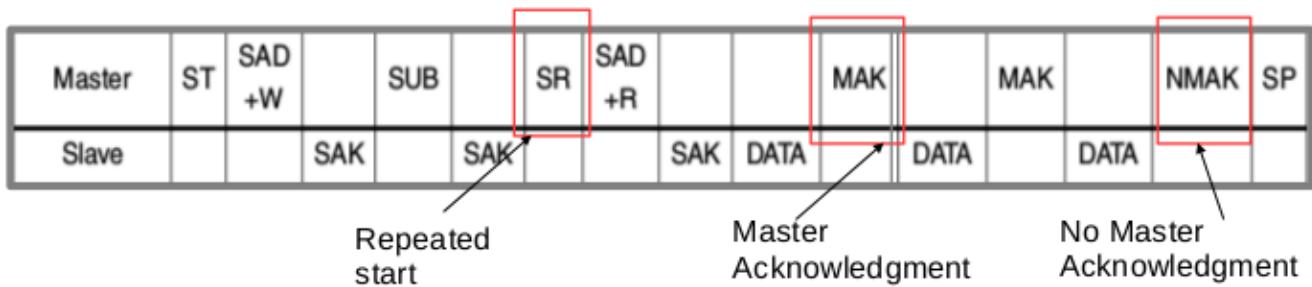


Example; 3.0 Hz data rate, so $1\ 0\ 0\ 0\ 1\ 0\ 0\ 0$ → 0x88, so write 0x88 to 0x00 location (CRA_REG_M)

Harry Li, Ph.D. Mar 2017

Now, Read Data. PP.10 P.P.T.

Read the Sensor Data



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).



Note: Good Ref.

Python Code implementation Sources .

From Nvidia developer forum, the referen

<https://www.instructables.com/Raspberry-Pi-I2C-Python/>

Enable i2c:

Step 1. configure i2c

sudo usermod -a -G i2c \$USER

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:~$ sudo apt-get install i2c-tools
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:~$ sudo apt-get install python-smbus
Reading package lists... Done
Building dependency tree
Reading state information... Done
```

To Detect i2c Devices

PWM φ

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

```
harry@harry-desktop:~$ sudo usermod -aG i2c $USER
[sudo] password for harry:
harry@harry-desktop:~$ i2cdetect -y 0
Warning: Can't use SMBus Quick Write command, will skip some addresses
  0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          ...
10:          ...
20:          ...
30:          ...
40:          ...
50:          ...
60:          ...
70:          ...
harry@harry-desktop:~$
```

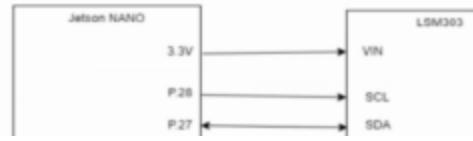
"U" Not Ready to Be used.

"U" Examined, But No I2C Device found.

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 L:



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](https://github.com/adafruit/Adafruit_CircuitPython_LIS2MDL)

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.

```

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:~$ 

```

```

nvidia@nvidia-desktop:/sys/class/i2c-dev$ i2cset 0 0x19 0x20
WARNING! This program can confuse your I2C bus, cause data lo
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 0x
0xd7
nvidia@nvidia-desktop:/sys/class/i2c-dev$ i2cget -v 0 0x19 0x

```

Note: I2c 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		NACK	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.

Feb 27 (Monday)

Note: 1° Homework on I2C LSM303 Sensor Interface.

Consider Motor Activation Design & Implementation

Stepper motor

BLDC (Phase A, B, and C)

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

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

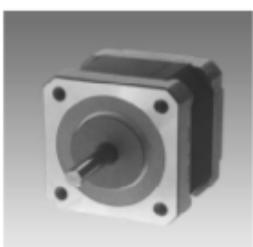
2° Motor Drive (Professional Grade) Example.

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

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

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

Example: Characterization of A Stepper Motor. T.P.Z.



Stepper Motors

Note 1°: 4 wires { A+, A-, B+, B-

Standard Stepper Motor

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.

Note 2: The 4-wires of the motor are not directly connected to the CPU of the target platform.

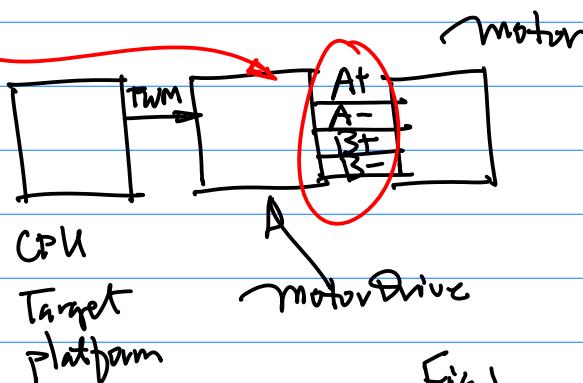


Fig.1.

3° Behavior of A Stepper Motor

a. Full Step Activation of A motor.

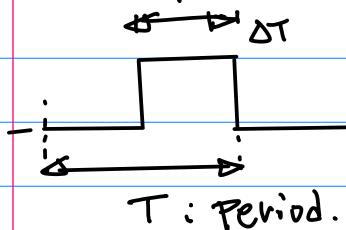
200 Steps for 360° Revolution.
1 Full Step (FS) \rightarrow 1.8 Degree Angular Displacement.

Note: PWM of A Target platform.



Witbot NEMA17
Stepper Motor ...
\$9.59
Amazon.com

PWM: Pulse width Modulation.



$$\eta = \text{Duty Cycle} = \frac{\Delta T}{T} \dots (1)$$

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

f_{PWM} : Common Range 500 Hz ~ 1 kHz.

b. Half Step (HS)

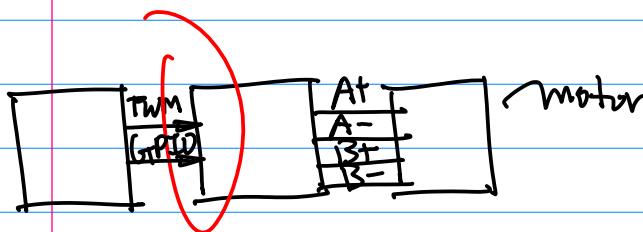
Angular Displacement

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

A
motordrive

Fig. 2.

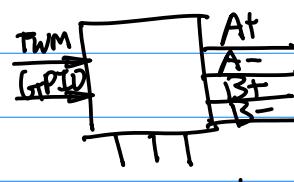
CPU	DriveBoard	Note
PWM (Output)	Input	Controls the speed.
GPID (Output)	Input	Direction (C.W. or C.C.W.)

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

Suppose we would like to have 5 Different Step Options for the DriveBoard, Since

$2^3 = 8$, $2^2 = 4$. Hence, we will

Need 3 pins added to the Driver Board.

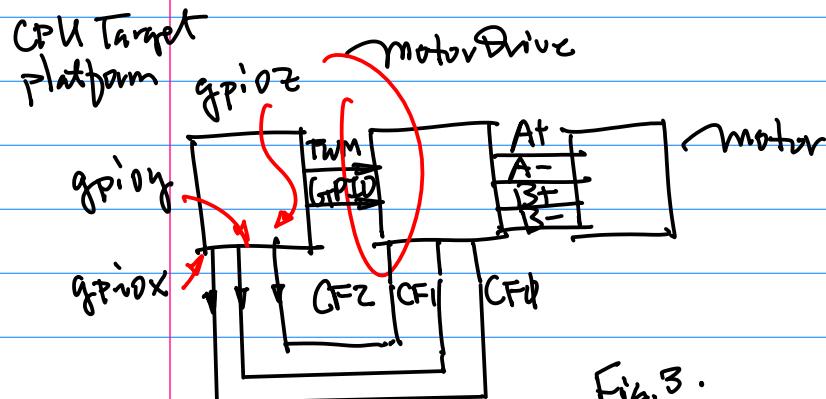


CF2 CF1 CF0

Configuration Pins

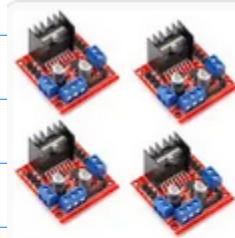
CF2	CF1	CF0	
0	0	0	F.S.
0	0	1	H.S.
0	1	0	1/4 Step
0	1	1	1/8 Step
⋮	⋮	⋮	⋮

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



Note: It is OK to use L298N

As Entry
Level Drive
Board



4 PACK
L298N Motor
\$11.59
Amazon.com

March 1st (Wed)

Note! Homework is coming for
Design & Implement Stepper
motor Control with PWM.

Next Level Higher, Recommended

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

Motor Drive Board.

1^o Speed of the Activation,
f_{PWM} from the target Board.



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



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



HiLetgo 2pcs
A3967
\$13.49
Amazon.com

Current ~1000mA to 1500mA.
Speed $\leq f_{PWM} \approx 1\text{ kHz}$.

Not Recommended.

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.

 $\vec{P}_A(x_A, y_A), \vec{P}_B(x_B, y_B)$.

y

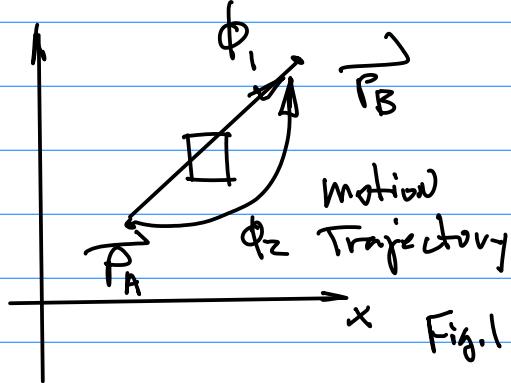


Fig.1

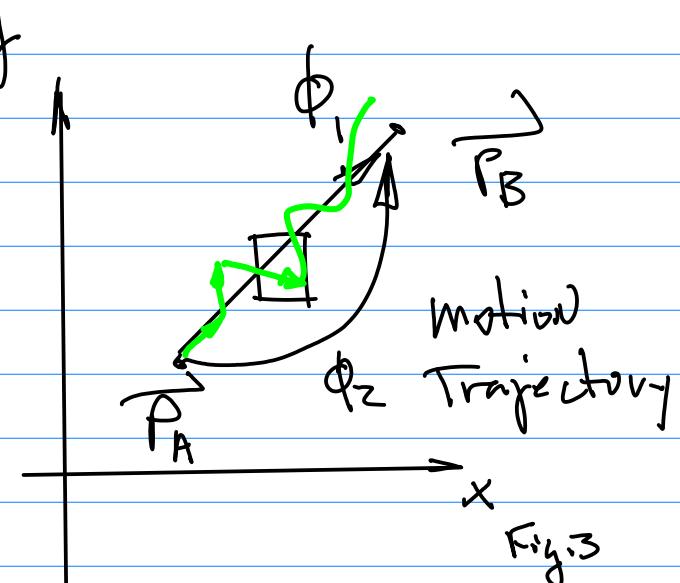


Fig.3

motorized

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

Input: ① PWM + GPIO

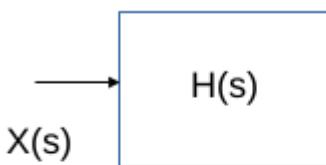


Fig.2

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

$$Y(s) = H(s) X(s) \dots (1)$$

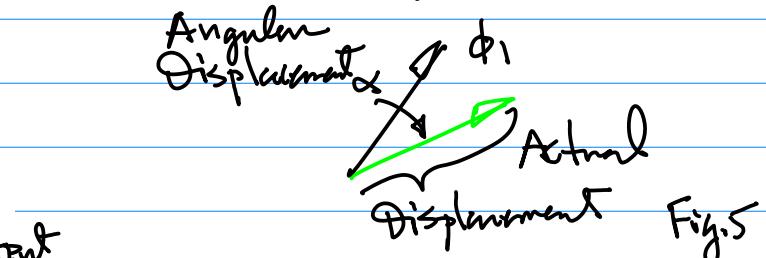
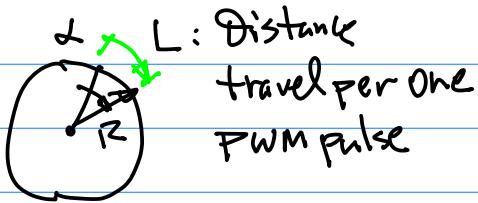


Fig.5

$f_{PWM} \rightarrow$ Displacement

$f_{PWM} \rightarrow$ Configuration of the motor \rightarrow Physical Dimension Drive, F.S. H.S. $\frac{1}{4}S$ etc.

of the motorized wheel.



α if Full Step:

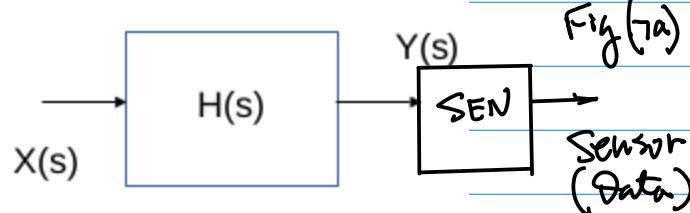
$$\frac{360^\circ}{2\pi} \text{ for one pulse}$$

from a PWM.

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

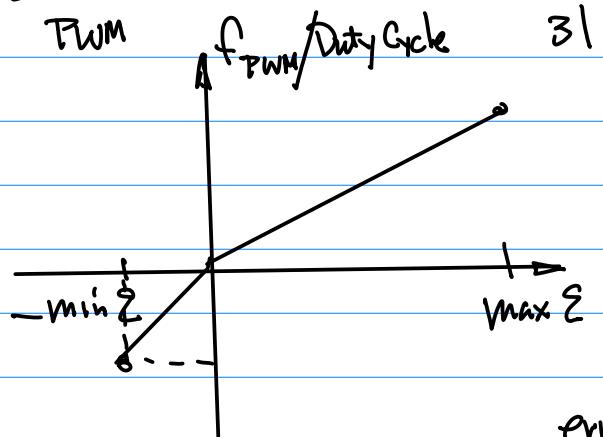
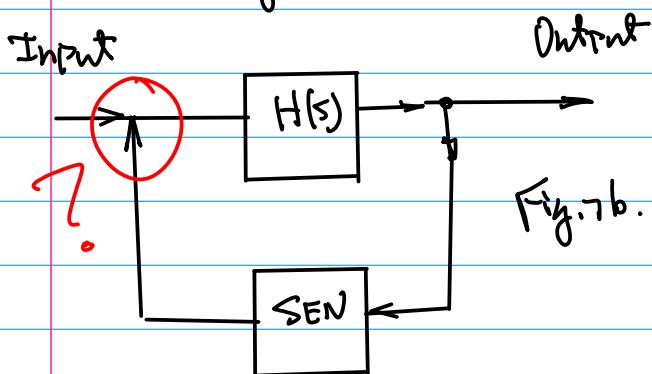
Add Sensors to measure the System Performance.

Open Loop System



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.



Assume Duty Cycle 0% ~ 95%

$$f_{PWM} = 500 \text{ Hz.}$$

GPIO controls Direction.

Fig. 8

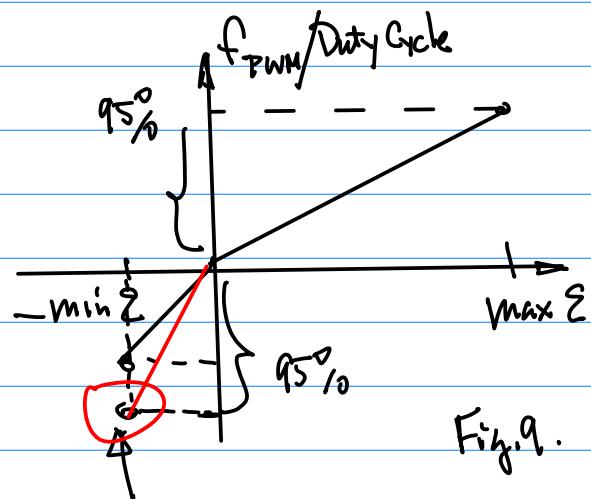


Fig. 9.

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

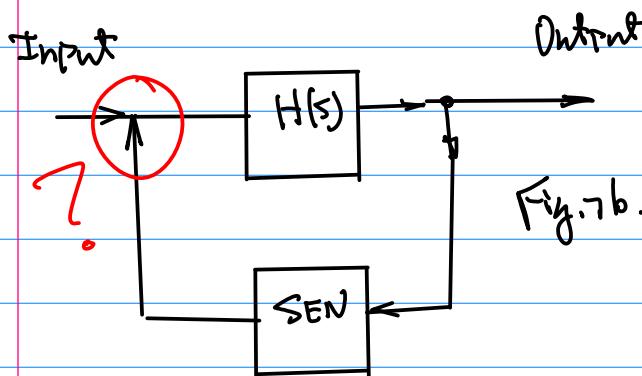


Fig. 7b. PP31.

2^o 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.

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

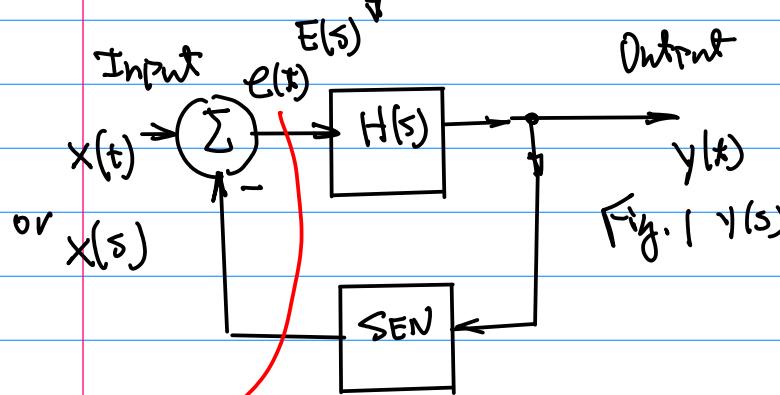


Fig. 1 y(s)

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

Note: Take this formulation.

$$y - y_1 = \frac{y_2 - y_1}{x_2 - x_1} (x - x_1) \dots (2)$$

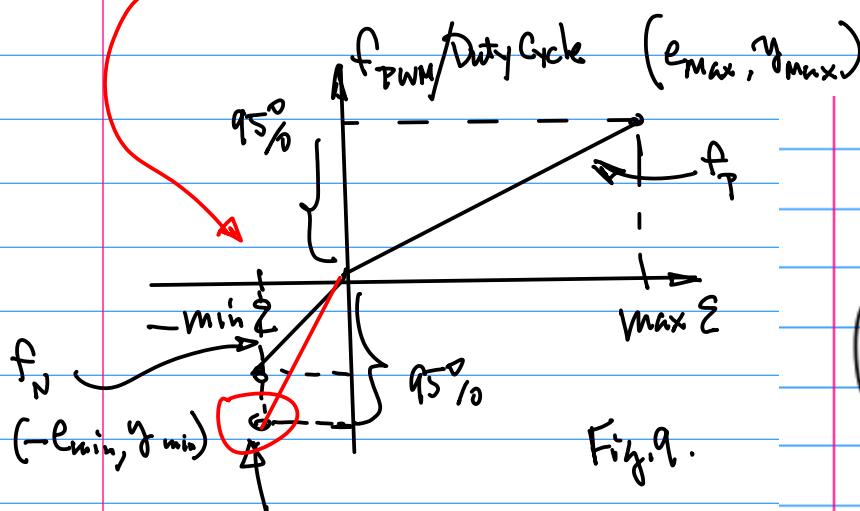


Fig. 9.

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

Given : 1^o Characteristic Curve of the Error Distribution.
 f_p , and f_n .

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

PP17.

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

$$y = ax + b.$$

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

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

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

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

b c

hence

$$\underline{y = bx + c} \quad \dots (4)$$

where

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

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

$$ay + bx + c = 0$$

Given the motor Configuration
With or without GearBox.



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.

$\ell_{max} = 1000$; $\ell_{min} = -800$ from

Experiments.

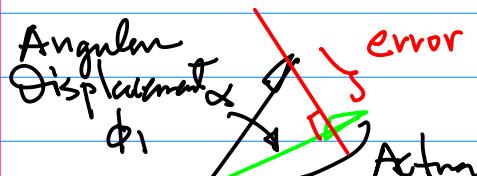


Fig.5
from pp.30.

Given f_{PWM} , Duty Cycle \rightarrow Given Stepper (F.S., H.S., etc.)
motor config.

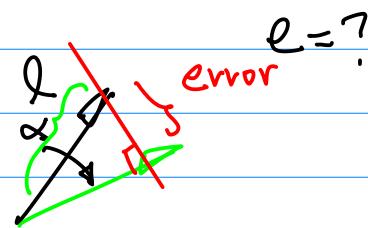
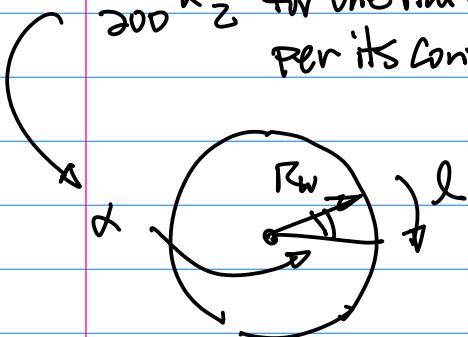
Given the wheel Dimension
 R , (10", 12", etc).

Find the error/Displacement
distance. for \times No. of
Pulses.

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

Angle.

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



Now, with 1:1 GearBox Reduction.

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

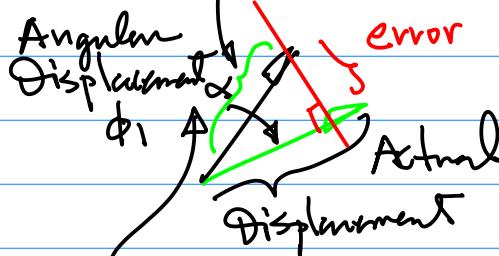
$$\alpha' = \frac{1}{R} \alpha \quad \dots (5)$$

To find physical displacement

$$l = \alpha \cdot R \quad \dots (b)$$

$$\text{OR, Displacement} = \alpha' \cdot R_w \quad \dots (7)$$

$$\text{dist} = \alpha' \cdot R_w \quad \text{from Eqn(7)}$$



Angle is from Lsm303
(Deviation from the Desired Trajectory)
Therefore, the error.

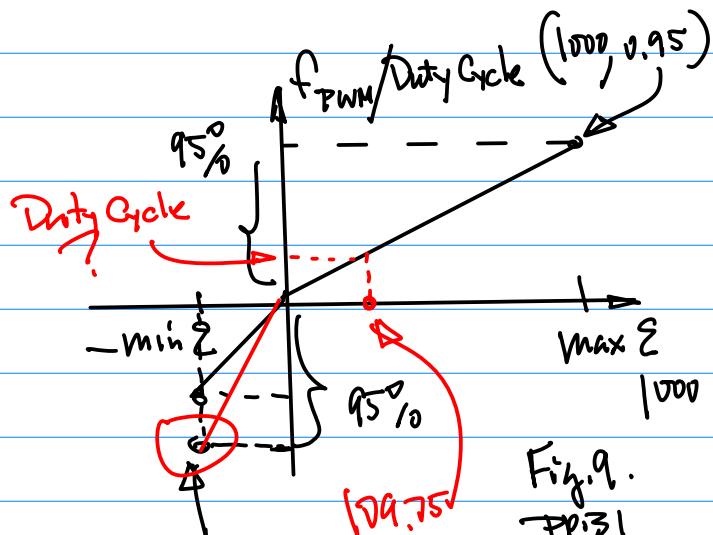


Fig. 9.
pp.31.

from Eqn(4)

$$y_f = bx + c$$

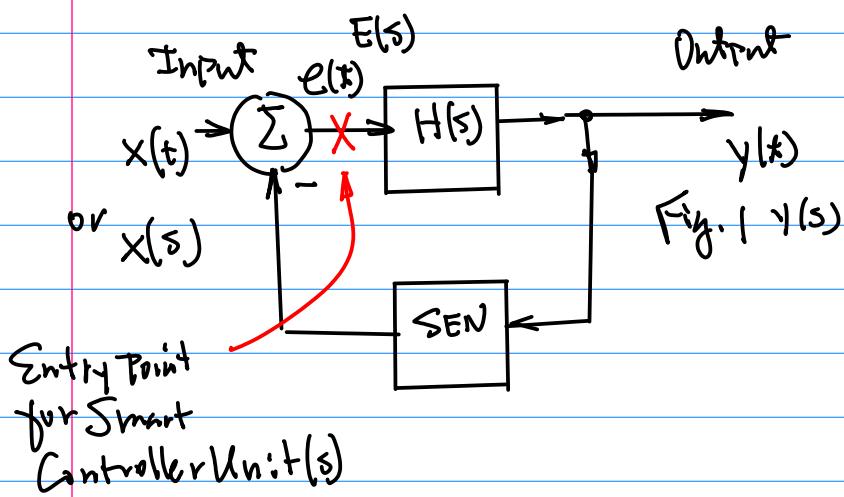
Slope from
the Characteristic
Equation.

$$\text{Where } x = 109.75$$

Duty Cycle for Correction.

March 8th (Wed).

Midterm Exam scheduled on the 22nd
(Wed). Please Bring the Prototype
Board to the Exam.



Observe the Responses of the System, Note to Reduce | Eliminate the errors(s) in the past (e.g. Accumulative Error), we Can Build integral Controller, Such as the one below.

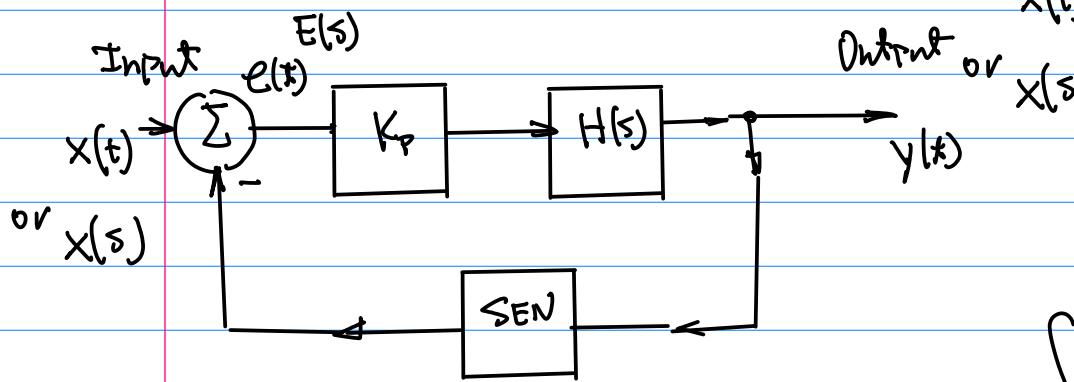


Fig.2

K_p : Gain for Proportional control.

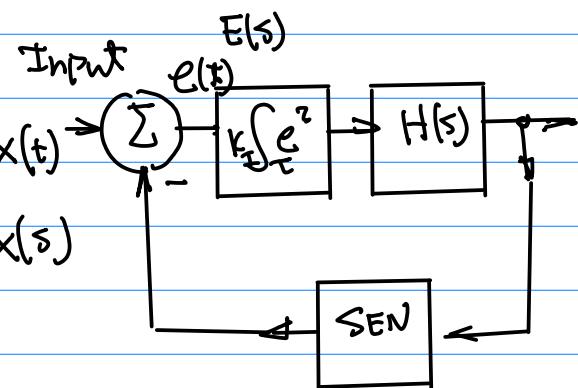


Fig.4

$$\int_{\frac{T}{2}}^T e(t) dt = \dots (1)$$

T: Time Interval

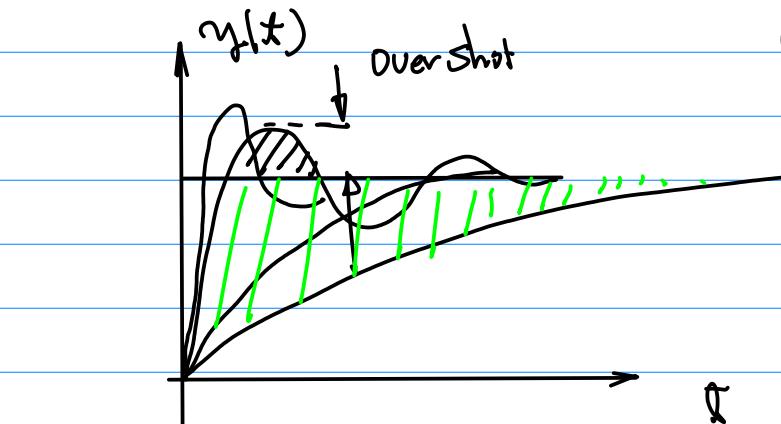


Fig.3

Question:
the Positive Error Area of the Response Curve From Fig. 1,?

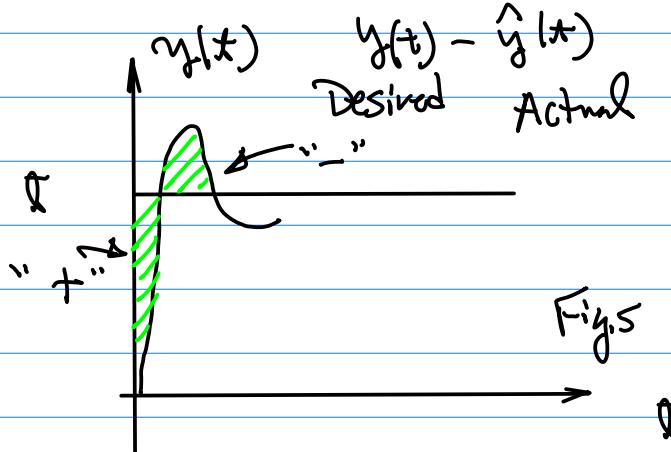


Fig.5

Integration of the error may have the cancellation due to the positive areas v.s. negative areas. \rightarrow try Absolute Error.

$$\text{Error: } f(e) = |e|$$

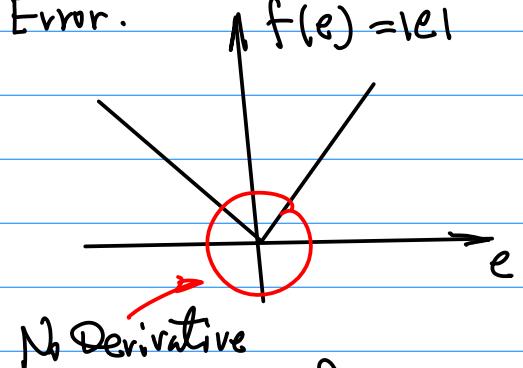


Fig.6

No Derivative

Therefore, signed error is adopted, hence, Eqn(1) becomes.

$$K_I \int_{t_0}^t e^2(\tau) d\tau \quad \dots (2)$$

For the discrete time :

$$K_I \sum_{i=-T}^0 e^2(i) \quad \dots (2-b)$$

Design a controller that can predict the error in the future.
Derivative Control.

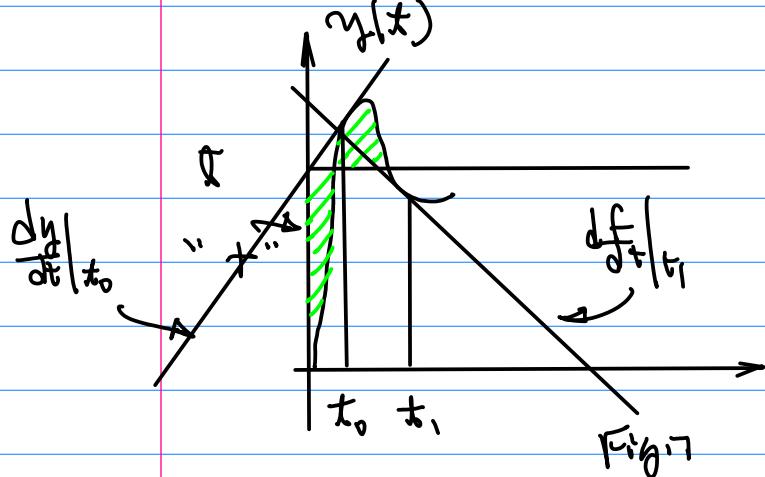


Fig.7

at $t=t_0$ $\frac{df}{dt}|_{t_0} > 0$, "Slop is up"

Error is increasing a short step ahead of t_0

if $\frac{df}{dt}|_{t_1} < 0$, "Slop is down", negative, then the error is decreasing. a short step ahead of t_1 .

Hence, the derivative controller is defined as follows,

$$K_d \frac{de(t)}{dt} \quad \dots (3)$$

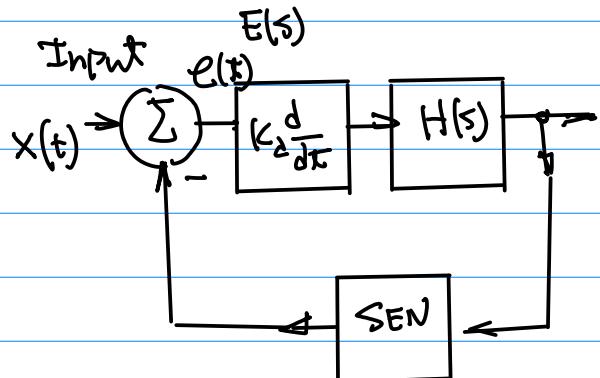


Fig.8

Numerical Computation of $\frac{df(t)}{dt}$

for

$$\frac{df(t)}{dt} = \lim_{\Delta t \rightarrow 0} \frac{f(t+\Delta t) - f(t)}{\Delta t} \quad \dots (4)$$

$$\approx \frac{f(t+1 \cdot \Delta t) - f(t)}{1 \cdot \Delta t} \quad \dots (4-b)$$

then, knowing Δt , then simplify the notation, we have

$$\approx f(t+\tau) - f(t) \dots (4-c)$$

for the error function, we have

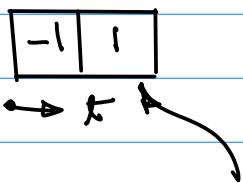
$$f(t) = e(t),$$

so

$$e(t+1) - e(t) \dots (5)$$

$$= 1 \cdot e(t+1) + (-1) \cdot e(t)$$

Design a kernel



... (5-b)

March 13 (Monday).

1° please check the CANVAS for the homework, Due 1 week from today.
Bonus for Enabling PID Control of the Motor Drive.

Test: Constant Speed even when the load is changed;
No/Minimum Vibration (Torque Control feature Enabled).

2° In Class Demo Next Monday.

Example: Computation of Integral Control.

Given Error Data in Table 1.

So] the Control Block Diagram, Fig. 4.
PP.35.

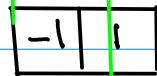
Then, from Equation in PP. 36.

Example: Suppose we have the error given in the following Table, find $\frac{de}{dt}$ the derivative of the error.

Table 1.

t	0	1	2	3	4
e(t)	0.75	0.93	1.1	0.9	0.7

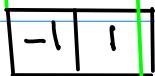
Step 1



$$-1 \times 0.75 + 1 \times 0.93$$

$$= 0.18$$

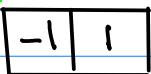
Step 2



$$-1 \times 0.93 + 1 \times 1.1$$

$$= 0.08$$

Step 3



$$-1 \times 1.1 + 1 \times 0.9 = -0.2$$

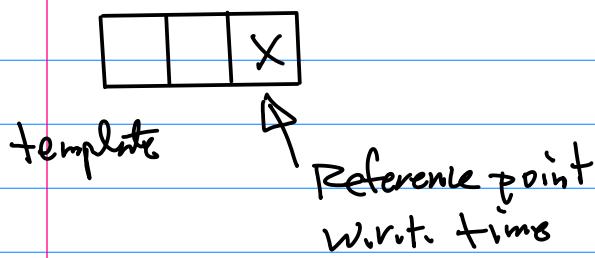
$$K_I \int_{t_0}^t e^z(\tau) d\tau \dots (z)$$

For the discrete Time :

$$K_I \sum_{i=-T}^0 e^z(i) \dots (z-b)$$

Time "T": steps.

Suppose for 3 steps T. from Eqn (z-b)
for the Reference Point (e.g., the Current Time t) : is equal to 0.



Now, we can put proportional, derivative, and integral controller together.
(see. the Class github)

Table 1.

Ref. 20225-107a-PID-v5-2019-4-25.pdf

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

t	0	1	2	3	4
$e(t)$	0.75	0.93	1.1	0.9	0.7

for ref. $t=0$



$$K_I \sum_{i=-I}^0 e^z(i) = K_I \left(e^z(0) + e^z(-1) + e^z(-2) \right)$$

$$= K_I \left(0.75 + 0 + 0 \right) = K_I * 0.75$$

for Ref t=1

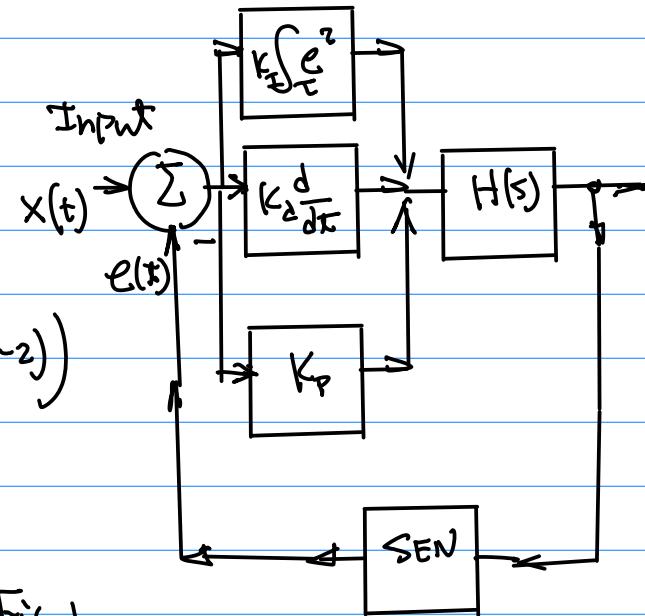


Fig. 1.

$$e_{\Sigma} = e_{kp} + e_{KI} + e_{Kd}$$

$$K_I \sum_{i=-T}^0 e^z(i) = K_I (e^z(1) + e^z(1-1) + e^z(1-2)) = K_P + K_I \int_{-T}^0 e^z(t) dt + K_d \frac{de}{dt} \dots (z)$$

$$= K_I \left(0.93^2 + 0.75^2 + 0^2 \right) = K_I \left(0.93^2 + 0.75^2 \right)$$

Consider Embedded ARM Architecture.

$f_{v,v} \mathcal{R}_{ef} t=z$

$$K_I \left(e^z(z) + e^z(z_1) + e^z(z_2) \right)$$

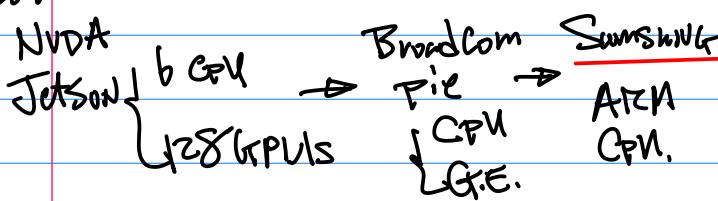
$$= \zeta_1(e^z) + e^{(1)} + e^v = \dots$$

To generalize this formula

Upper Bound of
Steps =
 \sqrt{f}

$$K_I \sum_{\tau=0}^{\infty} e^{(t_{ref} - \tau)} = K_I \sum_{\tau=0}^n e^{(t_{ref} - \tau)} \dots \text{... (1)}$$

Score:



Exercise: Check / Zend Samsung GPU
Datasheet, git hub.

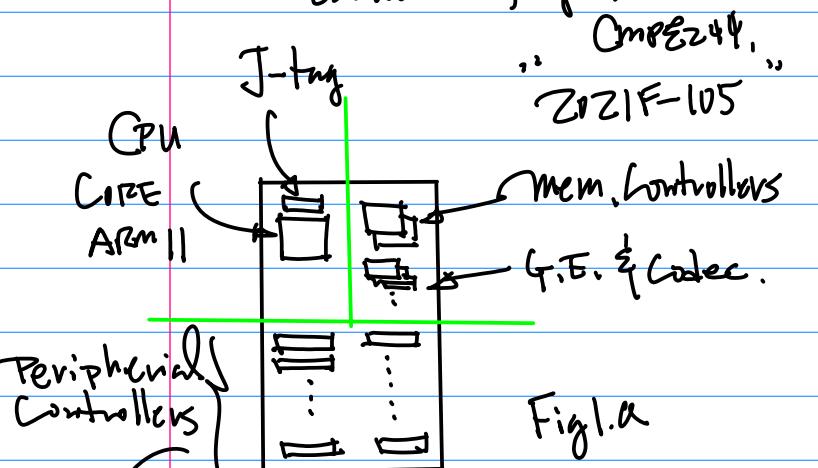


Fig. 1.a

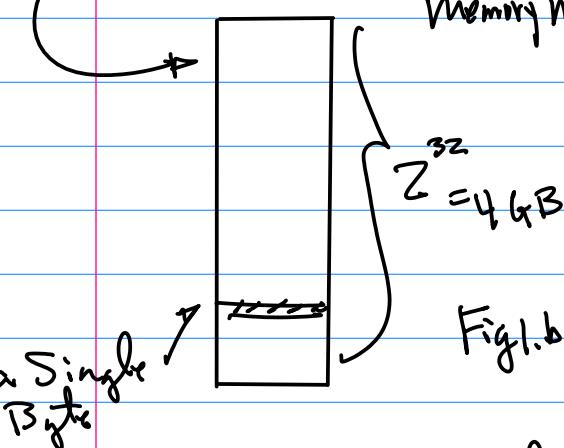


Fig. 1.b

Question: Map Any Peripheral Controller(s) to the memory?
Starting / Ending Addr. of the peripheral controller from GPU Datasheet.

GPIO as an example: Versatile, Scalable.

Register Bank | File | General Purpose Reg. 32bit | Special Purpose Reg. 32bit

Special Purpose Registers.

Control / Configuration Register

Naming Convention for RISC.

3 Letters 3 Letters 3 Letter
Prefix Root Transcript

GPX CON
Datasheet

10.4 REGISTER DESCRIPTION

10.4.1 MEMORY MAP

Register	Address	R/W	
GPACON	0x7F008000	R/W	Port A Config
GPADAT	0x7F008004	R/W	Port A Data R

Note 1° Addr. (ptr.) of the S.P.R.

GPACON[31:0]

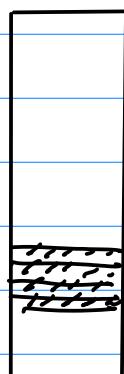
v3

10

0x7F00-8000

GPU Datasheet, pp308'

Fig. Z



Let's Design GPIO Testing.

With Pin2 as Output
Pin1 as Input.

Find the Binary pattern for it.

CMPE242

Spring 2023

40

PP312

GPAPUDSLP	0x7F008010	R/W	Port A S
GPACON	Bit		
GPA0	[3:0]	0000 = Input 0010 = UART RXD[0] 0100 = Reserved 0110 = Reserved	
GPA1	[7:4]	0000 = Input 0010 = UART TXD[0] 0100 = Reserved 0110 = Reserved	
GPA2	[11:8]	0000 = Input 0010 = UART CTSn[0] 0100 = Reserved 0110 = Reserved	
GPA3	[15:12]	0000 = Input 0010 = UART RTSn[0] 0100 = Reserved 0110 = Reserved	
GPA4	[19:16]	00 = Input 10 = UART RXD[1] 0100 = Reserved 0110 = Reserved	
GPA5	[23:20]	0000 = Input 0010 = UART TXD[1] 0100 = Reserved 0110 = Reserved	
GPA6	[27:24]	0000 = Input 0010 = UART CTSn[1] 0100 = Reserved 0110 = Reserved	
GPA7	[31:28]	0000 = Input 0010 = UART RTSn[1] 0100 = Reserved 0110 = Reserved	

March 15 (Wed) Example: Continuation
of the S.P.R. Discussion.

Example: From Fig.2, PP39

r3 |

→ █ | █ | █ | █ | █ | █ | █ | █ |

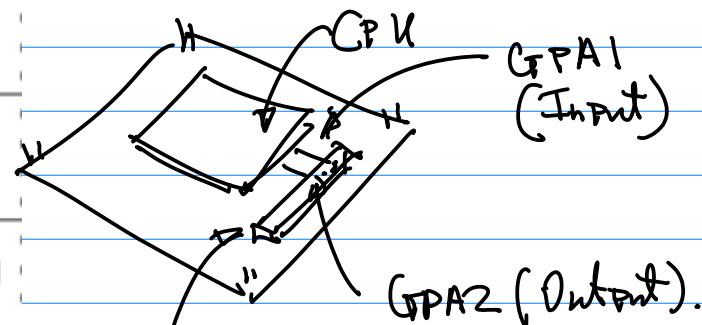
0x7F008000

CPU Datasheet, PP308

Suppose Design GPIO Testing

Circuit. GPA1 Input,

GPA2 Output.



Connector.

Design A Init Pattern for this job.

From CPU Datasheet,

r3 |

rf

█ | | | | | |

0x7F008000

CPU Datasheet, PP308

GPACON[7:4] = 0000 = 0x0 (Input)

GPACON[11:8] = 0001 = 0x1 (Output)

So,
GPACON[11:4] = 0001; 0000
= 0x10

rf Suppose Define A ptr for GPA

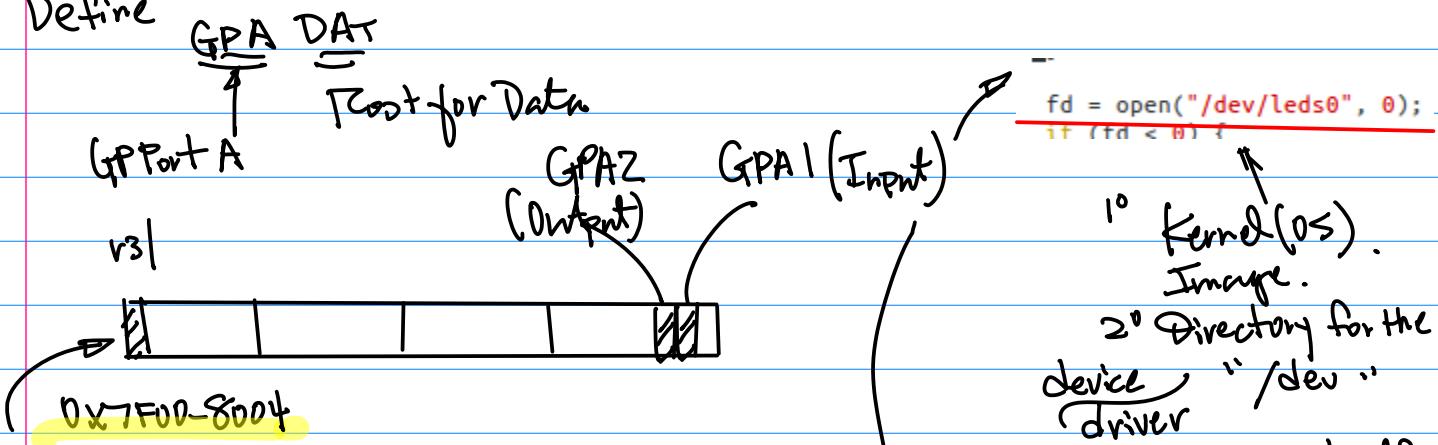
First, To clear [Prep.

S-ARM11 → GPACon & = ~0x11 << 4

Then to Set

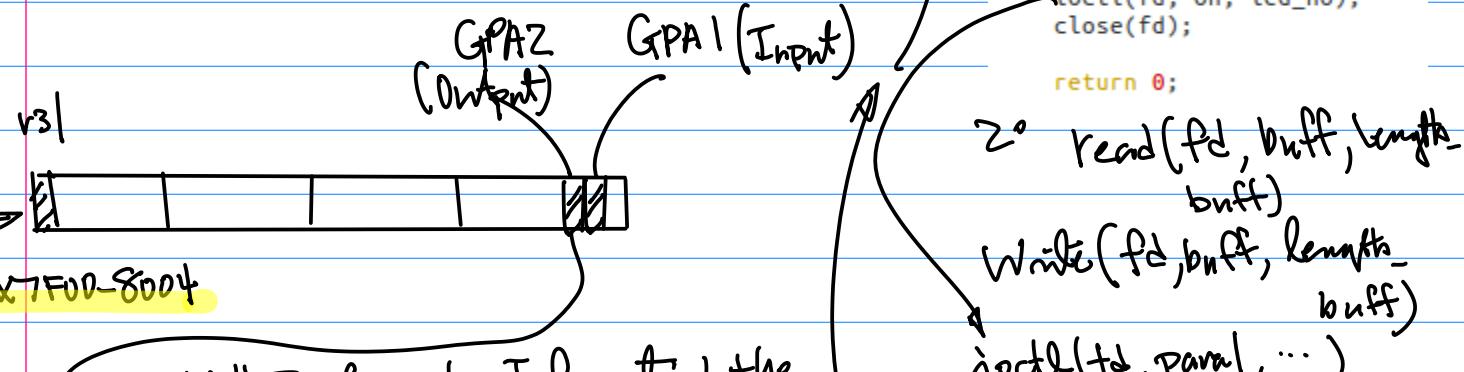
S-ARM11 → GPACon |= 0x10 << 4

Define



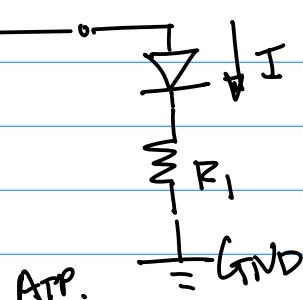
10.4.1 MEMORY MAP

TP 308

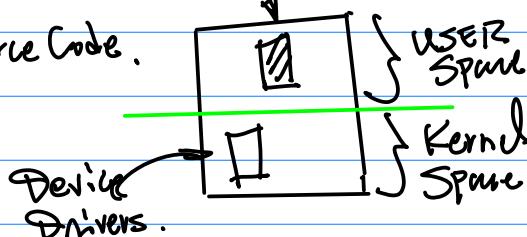


With Pin Connector Information, the Circuit Can be built

Similarly, the Input Circuit Can be implemented.



Example: Source Code.



```
fd = open("/dev/leds0", 0);
if (fd < 0) {
```

1° Kernel(OS).
Image.

2° Directory for the
device "/dev"
(driver)

3° Driver itself
"/leds0"

Note: To access to the Device
"read, write, ioctl" 3
functions.

```
ioctl(fd, on, led_no);
close(fd);
```

```
return 0;
```

2° read(fd, buff, length-
buff)

Write(fd, buff, length-
buff)

ioctl(fd, para1, ...)

Value to
Be Passed

```
static long sbc2440_leds_ioctl(struct file *filp, unsigned int cmd, unsigned long arg)
```

```
[  
    switch(cmd) {  
        unsigned tmp;  
    case A:
```

Required.

```
static int __init dev_init(void)
```

```
{  
    int ret;
```

```
    {  
        unsigned tmp;  
        tmp = readl(S3C64XX_GPECON);  
        tmp = (tmp & ~(0xffffU<<16))|(0x1111U<<16);  
        writel(tmp, S3C64XX_GPECON);  
  
        tmp = readl(S3C64XX_GPEDAT);  
        tmp |= 0xF << 4;  
        writel(tmp, S3C64XX_GPEDAT);
```

Note: GPEDAT Write Back to GPEDAT

Note: 1. Read long from GPECON.

32bit

PP320

		0100 = AC97 SDI 0110 = Reserved	0101 = Reserved 0111 = Reserved	
GPE4	[19:16]	0000 = Input 0010 = PCM SOUT[1] 0100 = AC97 SDO 0110 = Reserved	0001 = Output 0011 = I2S DO[1] 0101 = Reserved 0111 = Reserved	0000

Ref on PWM

20225-107c-pwm-config-v4-hl-2022-3-3.pdf

\$ls sys/class/pwm

```
harry-nano@harry-desktop-nano: /sys/class/  
harry-nano@harry-desktop-nano: /sys/class$ ls /sys/class/pwm  
pwmchip0 pwmchip4  
harry-nano@harry-desktop-nano: /sys/class$
```