Example: Continuation of the Introduction/Embedded Software Architecture.

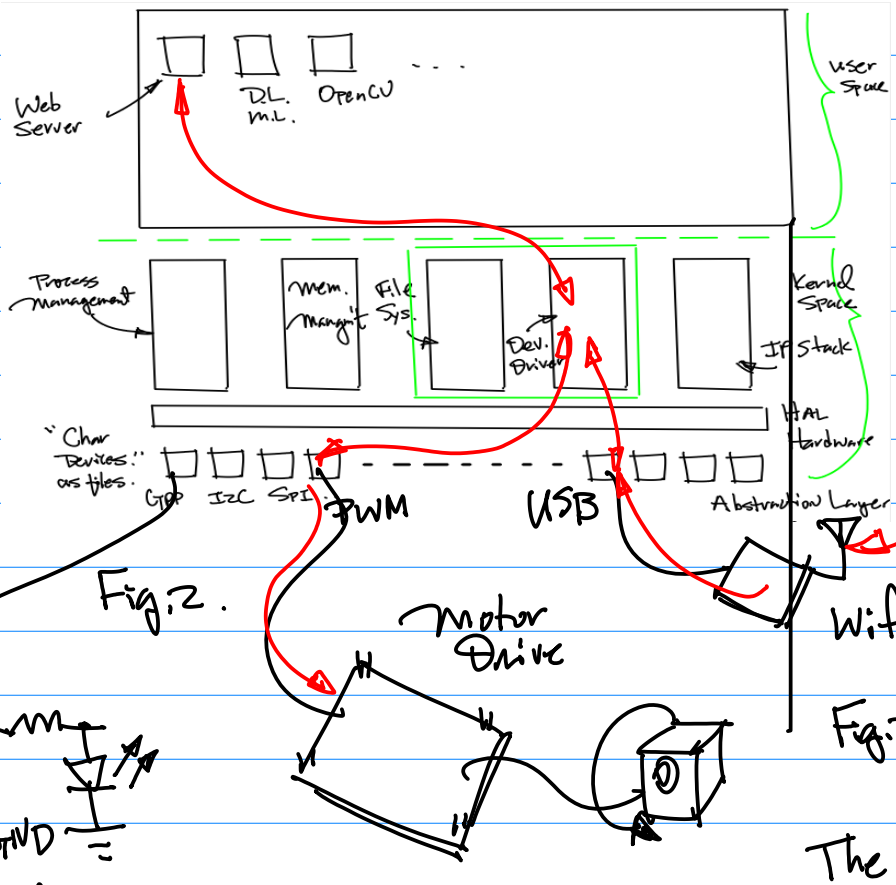
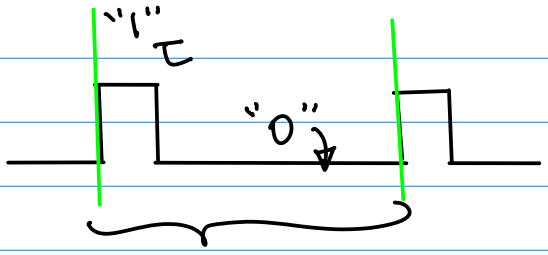


Fig.2.

Fig.2

Note: PWM — Pulse Width Modulation.

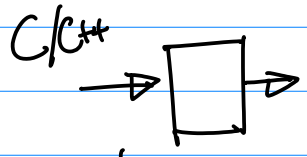


T One Period

$$\text{Duty Cycle} \triangleq \tau / T \quad \ldots (1)$$

$$f_{PWM} \quad \ldots (2)$$

The Software Tools: One of them is open source gcc, or g++ Compiler.



Porting.

- Match to the Core (ISA: Instruction Set Architecture)
- Device Drivers Customization.

Note1. Project Assignment on CANVAS

CMPE258
Semester Long Team Project On Embedded Software Systems
HL

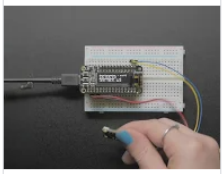
1. Design and implement a team project based on each 2-person team (or individual person). The project requirements are listed in details below. Note this project counts total 10 points and is due at the end of the semester and it requires a team presentation in the last week of the semester.

2. The technical requirements of the projects:
(2.1) Design and implement your team project which has to utilize embedded Linux OS on your chosen

Note2. This Coming Wednesday Presentation of your project Proposal, e.g., The Abstract.

3. One page project executive summary:
(3.1) Create one page executive summary of your project, with the following information
a. Title of the Project
b. List of each team members: First Name, Last Name, SID, Email Address, and Affiliation (such as Computer Engineering, Software Engineering, MS AI);
c. Team coordinator: Identify the team coordinator;
d. Abstract (up to one page):
Describe
(i) the objectives of the project ;
(ii) the technical challenges;
(iii) the proposed methodology to be employed;
(iv) the software tools and hardware platform;
(v) results and deliverable;
(vi) Experience gained and/or lessons learned.
e. table with list of the work contributed by each team member

Adafruit Industries LL…
$12.50
DigiKey
Get by 10/…



adafruit-Adafruit…
$12.50
Adafruit Indu…
30-day retur…

In-Class Presentation on Wednesday.

Homework. Nov. 1st (Wed) In-Class Demo.

1° Buy LSM303 sensor. (I2C Interface)

2° Enable PWM Driver on your target platform Then use Python, OR C/C++

To write an user program to read sensor information, and display the Sensor information on the Console.

3° In-class Demo Requirements.
Bring your prototype Board together with the sensor interface to the Class for Demo.
Ref1. from the github, First 25 slides.

CMPE244 / 2021F-116a3-i2c-v2-hl-2021-11-18.pdf

Ref 2. Command Line Commands for I2C Device Detection, and Python Sample Code.

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

Note: Midterm Exam is to be scheduled Once the I2C Sensor interface work is Covered and homework is done. Most likely in the 2nd ~ 3rd week of November.

Oct. 25 (Wed).
Note1. Quiz Next Monday.
Please Bring Your Laptop / Prototype System to the Class.

Scope: OS. Kernel. Build. Device Drivers, Kconfig.

Note2: Quiz (Nov. 6).
PWM Prototype Board + Motor Drive Board + Stepper motor.
→ Code with Calculation. f_{PWM}, Duty.

Note3. midterm Exam.
Nov. 13 (Monday).
① Architecture CPU Datasheet; Device Drivers (GPP, PWM, I2C) ②
③ Coding → Register Level (SFRs)
④ Hand calculation for Code Debugging.

Oct, 30 Moday
Note 1. The road map for the 2nd half of the class.
(1) Enable/design/deploy webserver solution by using GI (Gate Way Interaces), e.g., WSGI and CGI, the first one is the backend code (engine) written in Python, the 2nd one is CGI for C/C++ based backend solution.
(2) Design/develop/deploy ChatGPT API Version 3.5 to build technical/user support system.
(3) APP development for the Android platform.

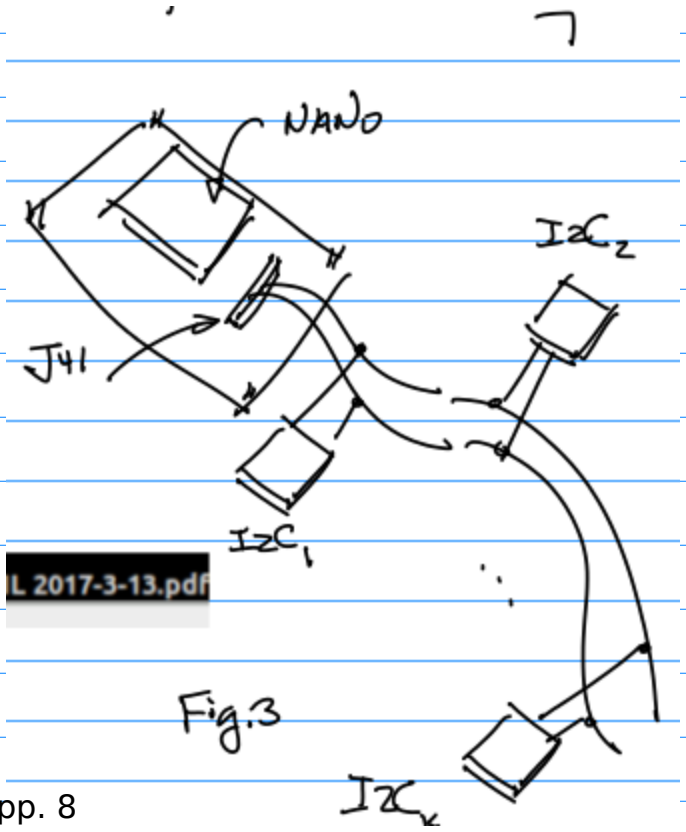
(4) Embedded software system will provide scalability and vertical integration. On hardwere side, we will add I2C sensor interface, LSM 303.

Note 2. Midterm exam is scheduled on Nov. 13 (Monday),
(1) One hour exam, one page formula is permitted. But close book and close notes.
(2) 15 minutes for prep for CANVAS submission.
Note: all the submission has to via CANVAS to be official, no late submission will be accepted. If the system is crashed or not responding to the subssion, please do screen capture to keep the record with time stamps and your ID.
(3) Scope of the midtem:
from the introducation to I2C SPR's init and configuration.
(4) format of the exam:
a. Combination of hand calculation and execution of your code.
b. you will need to bring your target platform to the exam.
c. The submitted homework, code can be reused.
d. Development platform, OS kernel compilation and build, which is a part of the subjects to be includeded.
kernel space debugging;
Kconf,
make menconfiguration;
e. Please bring your smart phone, you will need to take photos of your prototype board and the photo(s) of the actual implmentation, be sure to place SID card next to your board.
f. Scree captures may be needed, to capture the execution of your code with your ID.
(5) Prepare for the submission:
a. photos to be converted to pdf;
b. take photos of your hand calculation then convert the photos to pdf
c. combine all the pdf in a proper orcder into one pdf document.
Note: please use 1024x768 resolution as a reference when taking photos.

Example: On I2C protocole and I2C sensor interface.

Ref. 1.

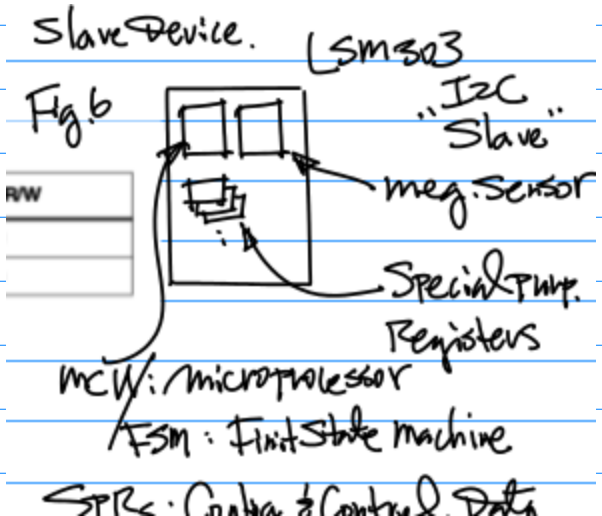
2022S-101-note-part2-cmpe242-2022-05-9.pdf



L 2017-3-13.pdf

Fig.3

pp. 8



Note: 1. Master and Slave of I2C devices;
2. Total number of slaves possible, theoretically, up to 7 bits, e.g., 2^7 = 128.
3. There is a MCU inside the slave for communication with this the master and execution of commands/instructions from the master, and perform init&config and other required task. 4. In case of LSM 303, there are 3 sensors in one package.

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

LGA-14 (3x5x1mm)

Description

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

Nov. 1st (Wed).
PWM Stepper motor Drive Demo. in-Class.
Note 1. Optional Feature:
Add Anaconda to Jetson NANO to Allow Better management of Development Environment and Packages.
Later in the Class Assignment, we may need this feature.

Example: Continuation of I2C Protocol.
1. Slave Addr. — Devive Datasheet
2. Additional Addr. for the master to Address to the SPRs (Special Purpose Registers) for the init & Config.
3. Information is then Read Back By the master. Which Can be realized By "Read" Command.

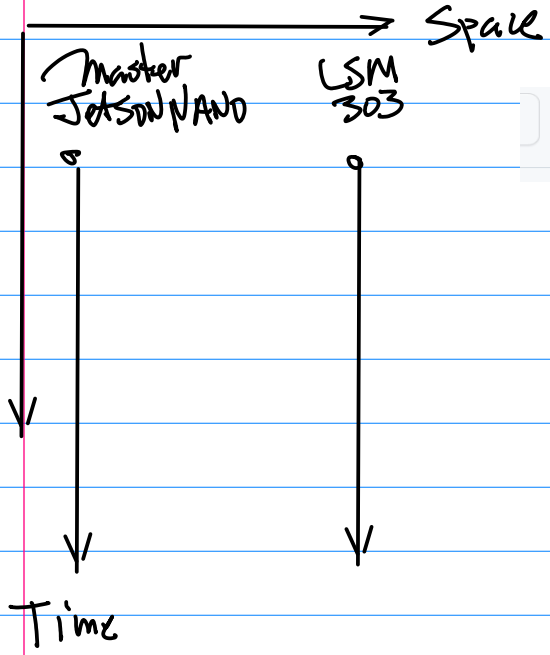
CMPE244

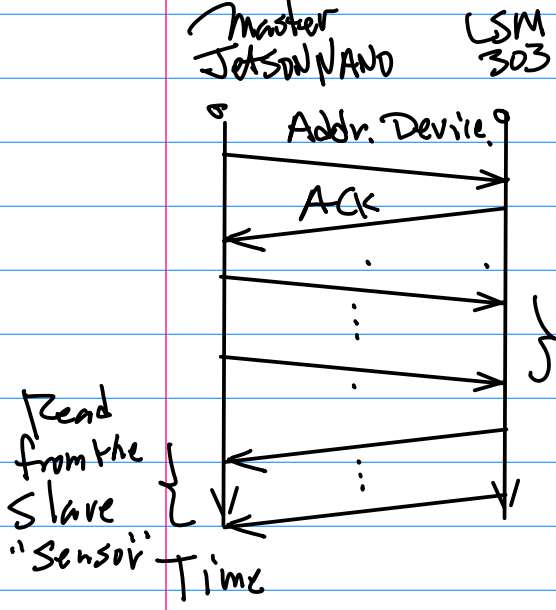
F2023

5D/

Space Time Diagram to Describe the I2C.

Spatial v.s. Temporal





Note: It's good Engineering Practice to Detect the Attached I2C Device(s).

Note: Read Datasheet to Complete the detailed Space-Time Diagram, then match up to your High Level Language Code.

Ref:

CMPE242-Embedded-Systems- / 2022S / 2022S-108b-AngularSensing-i2c-LSM303- final HL 2017-3-13.pdf

i2c

Nov.6 (Monday)

Note: Midterm Exam is scheduled on Next monday, Nov. 13th. Please Bring Your Prototype Board, And Blank Printer Paper.

Homework Nov. 19 (Sunday). a. 1° Integration of the target Board; b. motor Drive Board. (Note. The Board Has to Be able to Drive the Stepper motor per its Configuration, e.g. 1 Full Step (1.8 degree/step) OR A half step, etc. C. LSM303 Sensor to Calculate the motor angular Displacement;

2° Drive the motor 15 Degree Counter Clockwise Direction, then 15 Degree Clockwise Direction. Then Read the Angular information Back from your Sensor. (No further Calculation is Needed)

3° Submission ON CANVAS, And Bring your Prototype Board to the Class for Demo.

Note: For the team project, it is mendatory to Integrate ChatGPT into your Embedded Software System.

1. ChatGPT 3.5 API or higher;

Python Code.

Note: You may enable your Remote Access Software tool to connect to the Internet, in that case No USB wifi Needed.

2. USB wifi

Jetson NANO

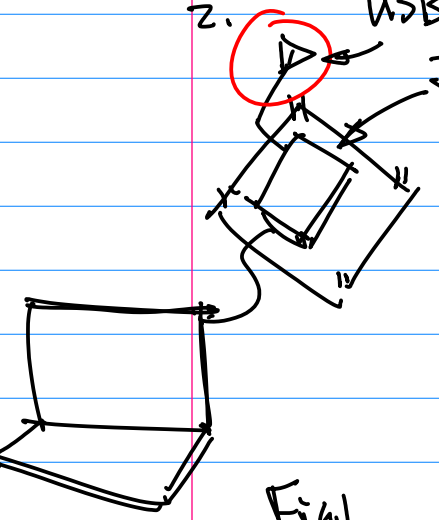


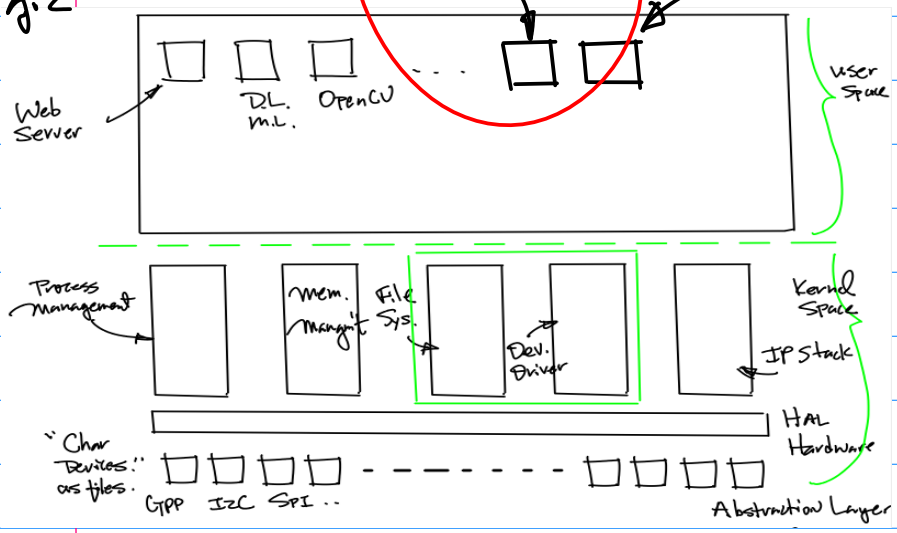
Fig1.

One of the program in the User Space.

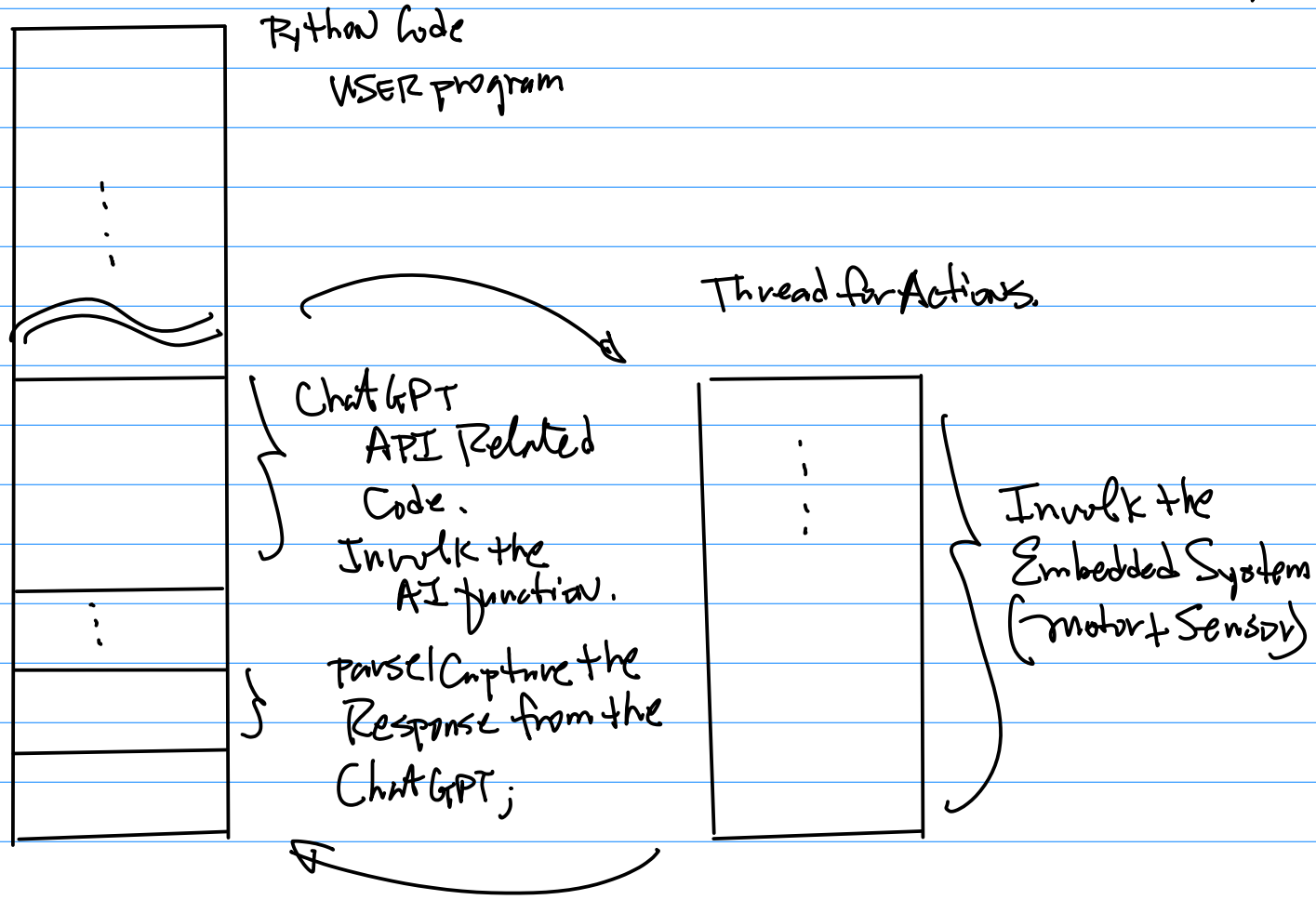
Python for ChatGPT API.

Web Server

Fig.2



Ref: github under OpenCV > DeepLearning2023S 20235-104 ~



Note: Both front End and Backend are Needed in this project.

Front End — UI (Browser)

Back End
1° Front End user input → Web Server → Back End code to Drive the motor.
2° Back End ~~Computation~~ → Server → On UI.
3° Back End Performs Some Computational Tasks. Such as Driving the motor

Get some data (From the Sensor)

Information flows One Direction

Information flows the Other Direction

Nov.8 (Wed).

Note1: Mandatory Requirement for the Semester Project

CMPE 244

Integration of ChatGPT API Functions to Your Embedded Software Systems

1. A mandatory requirement for the semester long project:
Integration of ChatGPT API functions to your project to realize user support function. T

2. he required ChatGPT API will be utilized to form a chatBot to answer user simple inquiry. It must be able to answer the user's questions regarding your system and to be able to start your proposed embedded system operations, such as activate a motor and receive sensor information feedback from your prototype board.

3. To get you started, see the PPT and sample code in the github here.
https://github.com/hualili/opencv/blob/master/deep-learning-2023s/2023F-104-%23190c-9a-chatGPT-API-YZ-HL-2023-9-7%20(copy).pdf

Note: Scope of the Implementation, especially Fine-Tuning the AI Enabled User/Technical Support feature is:

a. Fine-tuning Based On Q, &A ranges in 10~20 pieces of information.

b. Python ~~Or "Cython"~~ integration with your Embedded Software System, with Some features to Record/Remember the history of the interactions, then to start motor/get sensor input once the Number of interactions to Reach your preset value.

Review Session:

1) Bring your Prototype Board to the Class;

2) Make sure Motor, motor Drive Board work;
MotorDrive Board Output to Stepper motor (A+, A−, B+, B−)
↓
200 steps/360°
Board Inputs { PWM { f_{PWM}, DutyCycle; GPP }

3) Bring Blank Papers for Hand Calculation.

4) Screen Captures, photos to Be Converted to pdfs, then together with your hand written sheets in pdf to form One master Copy of pdf.

Format Requirements carries minor marks.

5) Three or Four Questions.

1 hour Exam. / 15 minutes To Additional Prepare the CANVAS submission.

In Case Screen Capture is needed, please be sure to provide your Name and SID.

If in case, CANVAS server is not Responsive, do screen Capture.

6) Scope of the Exam:
   a. Lecture Note
   b. Homework
   c. Prototype Board Demo.

7) Material to be prepared for Exam.
   Datasheet { LPC1769, ARM11, Jetson NANO OR Pi }
   Starting Guide.
   O.S. kernel config. Enviroment is Ready.
   C Sample code (ARM11. GPT, PWM) → github.

CMPE244
PART II
F2023

55/

Nov15 (Wed)

Road Map for the 2nd half of this class:

1° Apache Webserver. — CGI
Common Gateway Interface.
WS Gateway Interface.

2° ChatBot, Chat GPT API.
AI functionality to the
embedded System.

3° APPS
Smart phone
Android APPS
iOS APPS.

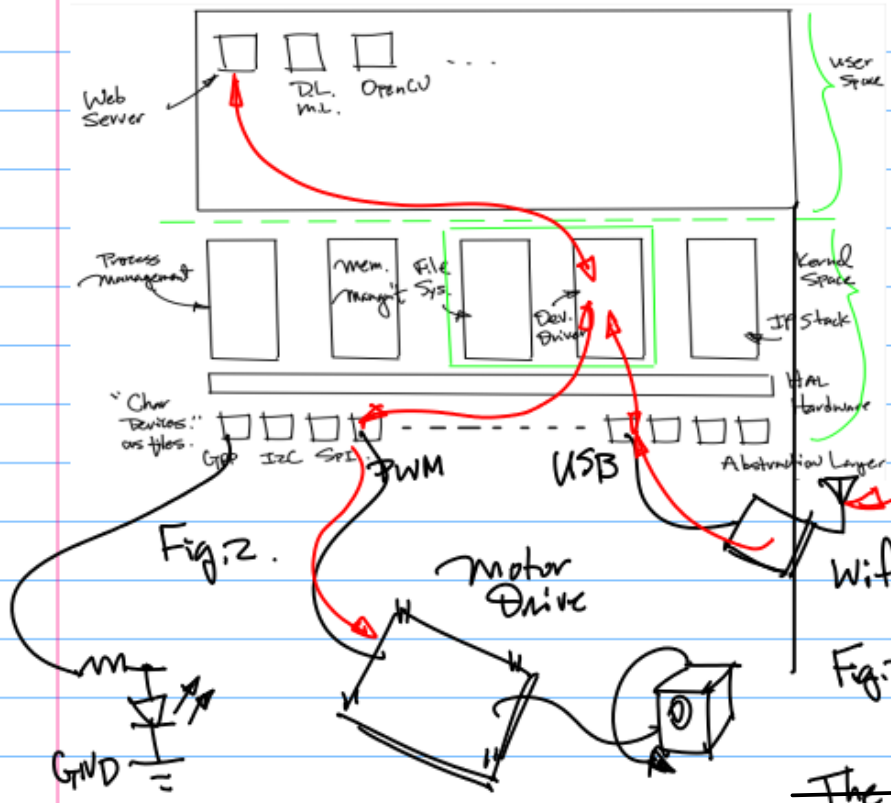


Fig.2.

Fig.2

Note: LSM303 I2C
Based on Sensor I/F.
Demo In-Class, Nov. 27
(Monday).

~~The Software Tools: Use of the~~

Group I Classes

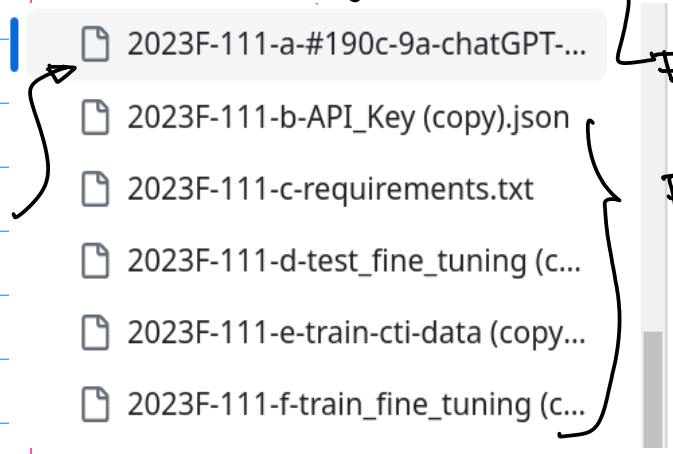
Group I classes are those classes which meet M, W, F, MTW, MWR, MTWF, MWRF, MTWRF, MW, WF, MWF, MF, TW, WR, MT, WS.

Regular Class Start Times	Final Examination Days	Final Examination Times
7:00 through 8:25 AM	Monday, December 11	7:15-9:30 AM
8:30 through 9:25 AM	Wednesday, December 13	7:15-9:30 AM
9:30 through 10:25 AM	Friday, December 8	7:15-9:30 AM
10:30 through 11:25 AM	Tuesday, December 12	9:45 AM-12:00 PM
11:30 AM through 12:25 PM	Thursday, December 14	9:45 AM-12:00 PM
12:30 through 1:25 PM	Monday, December 11	12:15-2:30 PM
1:30 through 2:25 PM	Wednesday, December 13	12:15-2:30 PM
2:30 through 3:25 PM	Friday, December 8	12:15-2:30 PM
3:30 through 4:25 PM*	Tuesday, December 12	2:45-5:00 PM
4:30* through 5:25 PM*	Thursday, December 14	2:45-5:00 PM

Final schedule

Consider the Implementation of OpenAI Based ChatBot Solution. ChatGPT . API 3.5

Ref: Grouping of the Refs.



Introduction "Hello".

Fine Tuning API

For Fine Tuning.

Note 1. Intro. with

Note 2: Install Conda ON your Target platform;

Note 3: Create Conda Enviroment for your ChatGPT To have the right version of Python and OpenAI package.

To Create conda Environment, you'll need *.yml file.

```
(base) harry@harrys-gpu-laptop:~/PycharmProjects/chatGPT$ tree -L 1
.
├── fine-tuningoutput.log
├── GPT_Fine_Tuning
├── GPT_Fine_Tuning (1).zip
├── GPT_Fine_Tuningoutput.log
├── hello-the-world
├── others
├── train-cti-data.jsonl
└── web-integration

4 directories, 4 files
```

Note 5. fold for integration of API for "fine-tuning".

Note 4. Corresponds to "Hello" Introduction of the first ChatGPT Code.

```
Terminal:  Local  +
 1 #-------------------------------------------
 2 # CTI One Corporation
 3 # for Chat-GPT
 4 # Version x0.1
 5 # Coded by: Youran Zheng, 2023-10-27
 6 # Create a Anaconda environment:
 7 # Open a terminal, then
 8 # $ conda env create -f chatgpt-2023-10-27.yml
 9 # Activate the Anaconda environment: $ conda activate chatgpt
10 #-------------------------------------------
11 name: chatgpt-2023-10-27
12
13 dependencies:
14   - python==3.7.1
15   - pip
16   - pip:
17     - openai==0.27.9
```

Note 6. To create the conda Environment.

file Name

Environment Name, does not have to match the file name. But it is good practice to have it matched.

Note 7. Create a .json file to keep/Enter your key

```
API_Key.json  chatgpt-2023-10-27.yml  ChatHistory.json  GPTWithHistorySaved.py
```

```
{
    "personalTestKey-2023-11-11": "sk-FGM
}
```

Nov. 20 (Monday).

Note 1. Team/Semester Long Project Presentation ON Dec. 6 (Wed), Last Day of the Instruction.

Mandatory Requirements:

1° Web Server & CGI

2° ChatGPT API integration.

3° Wifi:

(Note APP : Extra Bonus).

Note 2: In-Class Project Demo Requirements:

1° LSM303 Sensor (I2C) Interface

1° Mount the Sensor DV to motor.
So your System Can Read Sensor Output and display it ON the Web page ON your Smart phone/Laptop.

2° To be Able drive the motor 15° Counter Clockwise, 15° Clockwise.

3° This requirements will be Posted ON CANVAS.

Notes:

Webser Implementation Guideline:

Step 1. Installation of A Web Server

↓

Step 2. Test Parameter Passing in Both Direction.
(e.g. Brower → WebServer → BackEnd Program;
BackEnd Program → Web Server → Webpage Display ON your Browser)

Step 3. Integration of your Project into the Back End.

Example: I2C Sensor I/F.

Ref 1: Command Line Detect I2C Devices, And Communicate with an I2C.
2023S-I2C- from Cmpe242.

Test the Command Line Commands.

↓

Test Python Code

↓

Calculate Displacement Angle ±15°.

CmpE244
F2023

5a/

Python I2C Interfaces to LSM303

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.

Triple-axis Accelerometer+Magnetometer (Compass) Board - LSM303
Product ID: 1120 $14.95

https://www.adafruit.com/product/1120

```
1 import time
2 import board
3 import digitalio
4 import pwmio
5 import adafruit_lsm303_accel
6 import adafruit_lis2mdl
7 import lsm303

22 ### LSM303 SETUP BEGIN ###
23 i2c = board.I2C()
24 accel_out = adafruit_lsm303_accel.LSM303_Accel(i2c)
25 mag_out = adafruit_lis2mdl.LIS2MDL(i2c)
26 ### LSM303 SETUP END ###

28 ### TEXT DOCS SETUP BEGIN ###
29 file = open("Magnetometer_Output.txt", "w")
30 file.write("")
31 ### TEXT DOCS SETUP END ###

40     print("Angle %0.1f: " %angle, end='')
41     print("X=%0.2f Y=%0.2f Z=%0.2f" %mag_out.magnetic)
42     file = open("Magnetometer_Output.txt", "a")
43     file.write("Angle %0.1f: " %angle)
44     file.write("%0.2f %0.2f %0.2f\n"%mag_out.magnetic)
```

Example: from adafruit

```
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
```

Ref: 2022S-108 b. ~ github, CmpE242, folder: 2022S

Nov. 27 (monday).

1. Final Exam: Dec. 14 (Thu).
   2:45-5:00 pm
2. Demo in-Class, official Date on CANVAS.
   Optional Demo on this Wednesday, & official Date on Dec. 4.
3. Semester Long Project / Team Project Demo, presentation on Dec 4th, (Monday).

Note: Code Walk-Through during the presentation & Demo.
if Needed, some minor Number of presentations, may be scheduled on the 6th.

I2C Datasheets & Design.
Ref: 1. 2023S-102-i2C github 242-folder.

Command line Based Interface.

from Ref. 2. pp. 3

Note 1. Init & Config. Binary Init Pattern from the host to the "Slave" By Write op.

Note 2: ST (start by the host), SP (stop by the host)

I2C Handshaking LMS303

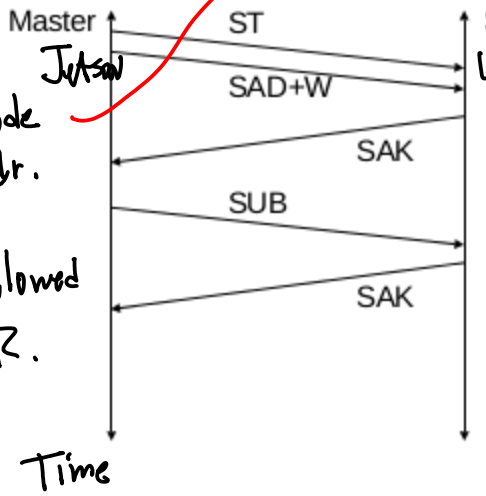
Table 11. Transfer when master is writing one byte to slave, pp 20, datasheet

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

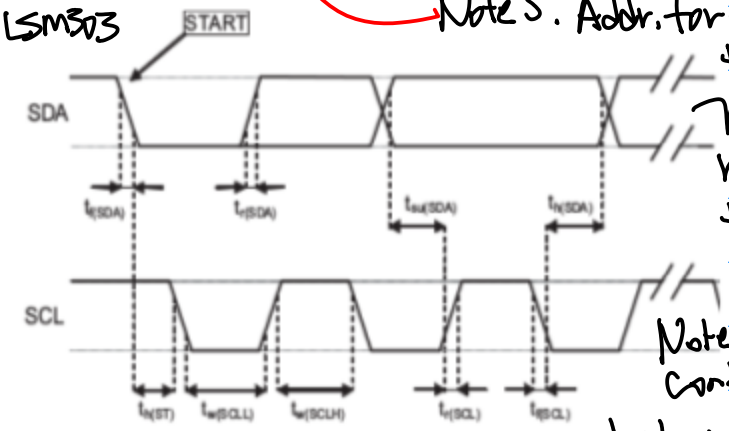
Note 4: SAK "Slave" Ack.

time.

Note 3: 7 bits node (slave) Addr. upto 128 "Slaves". followed by W or R.



Note 5. Addr. for the SPR that the master wants to talk to.



Note 6. Init & Config pattern to be written to the SPR. (Control/Config Register).

Figure 3. I 2 C slave timing diagram, pp 13 from datasheet

Harry Li, Ph.D. April 2015

TP:7. Note 7. SAD, SUB information { Datasheet / Command Line Prototype System

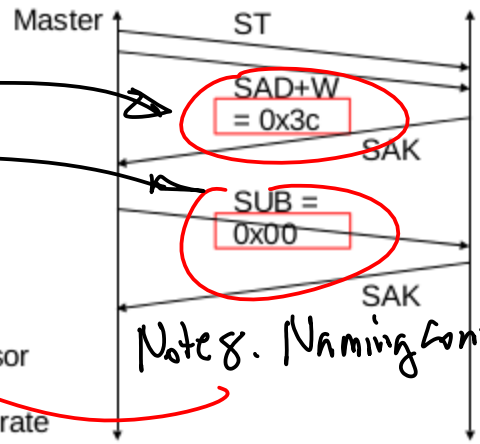
Steps for Init and Config (1)

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



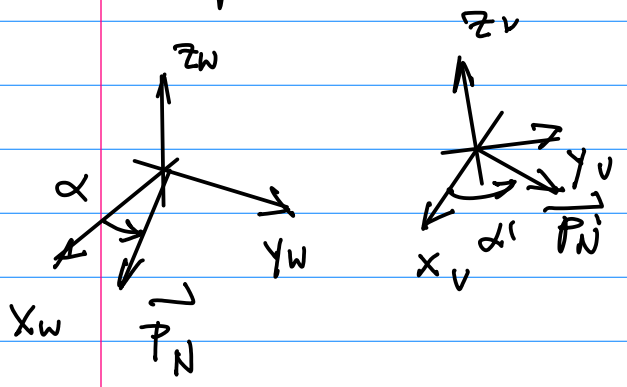
Note 8. Naming Convention

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

Note: fill in the Addr. Information Based on the Prototype System And Datasheet

Nov. 29 (Wed).

Discussion on LSM303 Data Interpretation.



$\vec{P}_N$ on X_W-Y_W, $\vec{P}_N'$ on X_V-Y_V

And X_W-Y_W & X_V-Y_V do not necessarily in Parallel. e.g.

$$\vec{n}_{Z_W} \neq \vec{n}_{Z_V} \quad \cdots (1)$$

$f(\alpha)$? $g(\alpha')$

map

$\alpha' \xrightarrow{\phi} \alpha \quad \cdots (2)$

But without further discussion of Eqn (2). find α' first

Note: To Test Rotation Angles, we have one option which is Based on Optic Encoder. But it lacks of the ability to define orientation indoor. Hence, Choose LSM303.

$(X_{m,i}, Y_{m,i}, Z_{m,i})$ at time step i $\cdots (3)$

And

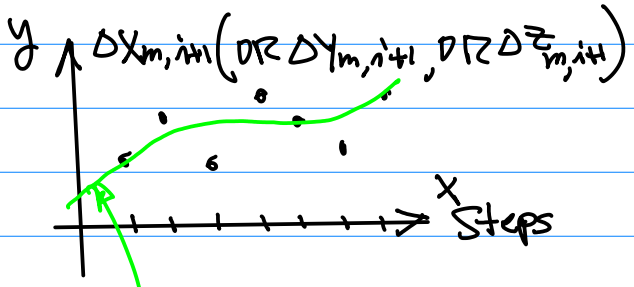
$(X_{m,i+1}, Y_{m,i+1}, Z_{m,i+1})$ at Step i+1 $\cdots (4)$

define

$$(\Delta X_{m,i+1}, \Delta Y_{m,i+1}, \Delta Z_{m,i+1}) =$$
$$(X_{m,i+1} - X_{m,i}, Y_{m,i+1} - Y_{m,i}, Z_{m,i+1} - Z_{m,i}) \quad \cdots (5)$$

Suppose (1) One second with Delta in Eqn (5) — Could be other Timeslice Depending LSM Control Register initialization such as 1 Hz, 10, 15 or 30 Hz.

(2) Number of pulses per the Unit time (1 second).



$y = f_1(x)$ ⇒ for X.

$$X_1 = f_1^{-1}(\Delta X_{m,i+1}) \quad \cdots (6)$$
$$X_2 = f_2^{-1}(\Delta Y_{m,i+1}) \quad \cdots (7)$$
$$X_3 = f_3^{-1}(\Delta Z_{m,i+1}) \quad \cdots (8)$$
$$X_\Sigma = \frac{1}{3}(X_1 + X_2 + X_3) \quad \cdots (9)$$

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

Note1. Control/Config Register is 8 bit

Note2. Address and Naming convention for these Registers from Datasheet.

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

Note3. Init & Config Binary Pattern.

Note4. Datasheet, Naming convention & Addr.

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

Note5. Additional Temperature Sensor Bit

Note6. Configuration Bits.

Table 71. CRA_REG_M description

TEMP _EN	Temperature sensor enable. 0: temperature sensor disabled (default), 1: temperature sensor enabled
DO2 to DO0	Data output rate bits. These bits set the rate at which data is written to all three data output registers (refer to *Table 72*). Default value: 100

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

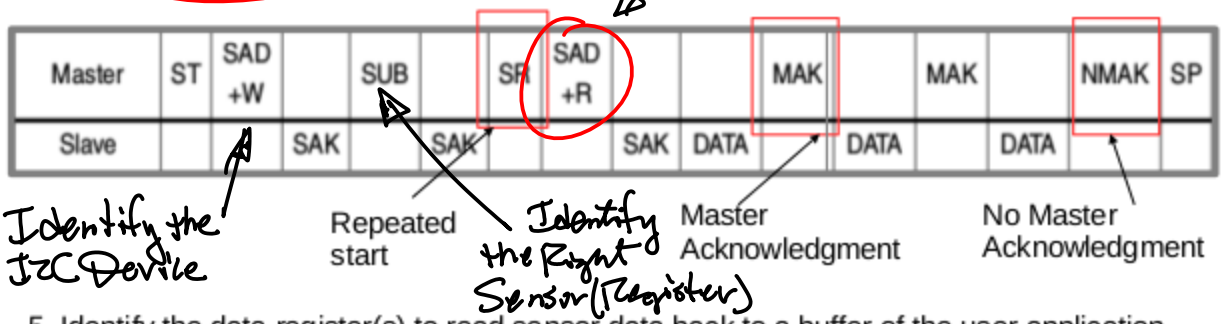
CmpE244
F2023

(63)

Note 1. "Read"

"Read" from here.

Read the Sensor Data



5. Identify the data register(s) to read sensor data back to a buffer of the user application program, from datasheet, pp 38

OUT_X_H_M (03), OUT_X_L_M (04h)
OUT_Z_H_M (05), OUT_Z_L_M (06h)
OUT_Y_H_M (07), OUT_Y_L_M (08h)

The data is in 2's complement form.

Therefore, to read the sensor date, we will have to assign the content at addresses 0x03, 0x04, …, 0x08 to 6 bytes buffers, the first 2 byte for X and the next 2 bytes for Z, and the final 2 bytes for Y.