

CMPE242

$$\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} \quad \dots (1)$$

March 10 (Wed)

$$G(x, y) = \frac{1}{\sqrt{2\pi} \sigma} e^{-\frac{(x^2 + y^2)}{2\sigma^2}} \quad \dots (2)$$

Note  $\mu_x = \mu_y = 0$ ,  $\sigma_x = \sigma_y = \sigma$

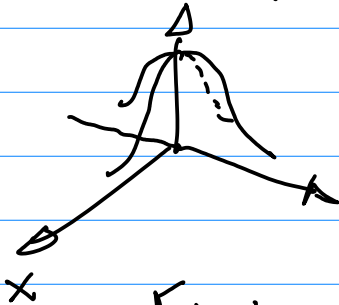
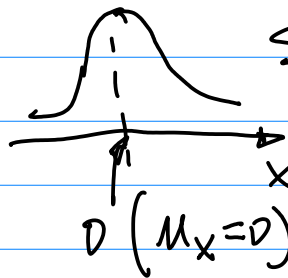


Fig 1.



Note:  $\text{LoG}(x)$  is NOT

Exactly the computation for derivatives, But we use it, for its Low pass nature, and 2nd order derivative.

Sol.

(i) "Mapping" to a kernel  
Build a kernel with "Odd" number of grids, elements

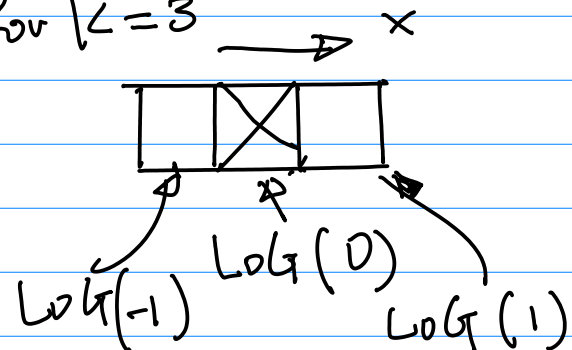
From Eqn (4), ppt from the github  
2018S-15-lec6-V3 ...

Let  $y=0$ , to have one indep. variable  $x$ .

$$\nabla^2 G(x, y) \Big|_{y=0} = \nabla^2 G(x, 0) \quad \dots (3)$$

$$= -\frac{1}{\sqrt{2\pi} \sigma^3} e^{-\frac{x^2}{2\sigma^2}} + \frac{x^2}{\sqrt{2\pi} \sigma^5} e^{-\frac{x^2}{2\sigma^2}}$$

$K \times 1$   
No. of elements One Row  
for  $K=3$



$\text{LoG}(x)$ , or  $\nabla^2 G(x, 0)$

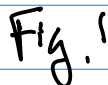
Example: (i) Use  $\text{LoG}(x)$  to Build a convolutional Kernel ( $z$ ) to Compute Derivatives of the Image  $\nabla^2 G(x, 0)$ .

$\hat{=}$  Identify the Center Reference  $\hat{=}$  from  $\text{LoG}(x)$  (or  $\nabla^2 G(x, 0)$ ). map it to the kernel

20185-10-0 ~  
PWM Driver

Theoretical Aspect  
Implementation C Code.

Power Antenna Square Waveform



TP1118, CPU Datasheet  
Prescaler: 8 bit, [0, 255]  
Divider: 1, 2, 4, 8, 16

$$f_{TWM} = \frac{50 \times 10^6}{(P_{rest} + 1) \cdot Div.} \dots (*)$$

If we need  $f_{\text{PWM}} = 2 \times 10^3$   
Find SPR, Set SPR. to Realize  
this frequency.)

From CPU Datasheet CFG of  
CON.

$$\underline{\underline{Sol}} \quad 2 \times 10^3 = \frac{50 \times 10^6}{(Pres+1) \cdot Div}$$

$$(Pres+1) \cdot Div = \frac{50 \times 10^6}{2 \times 10^3}$$

$$(P_{res} + 1) \text{ Div.} = 25 \times 10^3$$

Let Div = 16,  
Solve for Pres.

$$P_{rest1} = \frac{25 \times 10^3}{1}$$

$$P_{res} = \frac{25 \times 10^3 - 16}{16} \approx 1555$$

## Iteration,

Change PCLK to 10 MHz,  
then, we have

Pres =  $\frac{10 \times 10^6 - 16 \times 7 \times 10^3}{16 \times 2 \times 10^3}$  if it is still too big  
 therefore, then Low the CLK<sub>P</sub>  
 try CLK<sub>P</sub>  $\approx 2 \times 10^6$ . please verify it!  
 ARM11, Datasheet  
 { T CNT B "N" Counts  
 T Cmp B  $f_{PWM}$   
 20/85-10-  
 $f_{CLK}/f_{PWM} = N \dots (1)$

Note: SPR Responsible for  $f_{PWM}$

T CNT B  
 Timer Root Buffer

$f_{PWM} = 1 \times 10^3$  given.

$f$  master Clock peripheral

N Counts for CNT SPR.

$$f = f_{PWM} \cdot N \dots (4)$$

Given Target Unknown to be Calculated

Note: T Cmp B

Comparison Register

for Duty Cycle

2nd Counts Value for "Cmp"

Derived from Duty Cycle.

March 17 (Wed)

$f_{PWM}$  By Setting SPR's  
 Duty Cycle value

Define one period;  
 the CNT

Duty Cycle  $\rightarrow$  %  $\rightarrow$  Counts  
 Percentage  $\downarrow$   
 Cmp

GPFCON P.P. 322

GPFCON[29:28] = 10  $\rightarrow$  PWM

GPFCON[31:30] = 10  $\rightarrow$  PWM

#define S3C64XX\_ GPFCON

0x7...

"AND"  $\rightarrow$   $\sim (0x3U \ll 28)$   
 "Neg" "11"

$\rightarrow (0x2U \ll 28)$

"OR" "10" unsigned

Set 2 Bits  
 GPFCON[29:28] = 10

March 22nd (Mon)  
Review.

1° 3± Questions.

a Basic Concepts CPU Architecture  
Block Diagram,  
Memory Map, Peripheral Controllers

GPP (GPIO) } SPRS CDN  
PWM } DAT  
TCNT B  
TCMP B  
CFG  $\phi$ , 1

Architecture  $\rightarrow$  Mem  $\leftrightarrow$  SPR  
Code  $\leftarrow$

Kconf Script  $\leftrightarrow$  User Space programming  
Kernel Space Device Driver  
Define Compilation + Build Process.

Programming Requirements, No Program Code  
However! <sup>Writing</sup> Debug/Change the existing code is Needed;

b Design-Related Questions

SCH. Design, CKT for PWM

$P_{in}(s), f_{pwm}$   $\rightarrow$  GPIO  
Motor Drive  
 $P_{in}(s), \text{Labels}(s)$  Stepper motor I/F

GPP I/O Testing ("Hello, the World")

Input Testing CKT  
Output Testing CKT

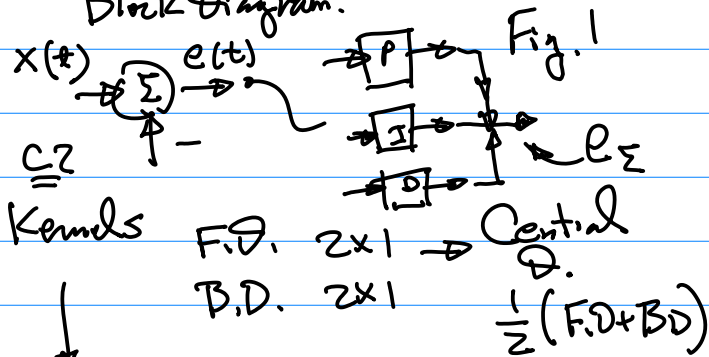
Resistance Value Calculation

$\subseteq$  Theoretical Aspects

c1. PID Controller Design

Basic Concepts

Block Diagram.



With Noise Reduction 3x1

Low Pass Filter:  $G(x)$  Gaussian.

$\downarrow$  2nd Order Derivation as in Computer Vision

$$\nabla^2 : \text{Laplacian } \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} \rightarrow \frac{1}{\partial^2} \frac{\partial^2}{\partial x^2}$$

$\log(x)$

Note: One page formula sheet is Allowed, However No verbal Description And/or Examples Allowed.

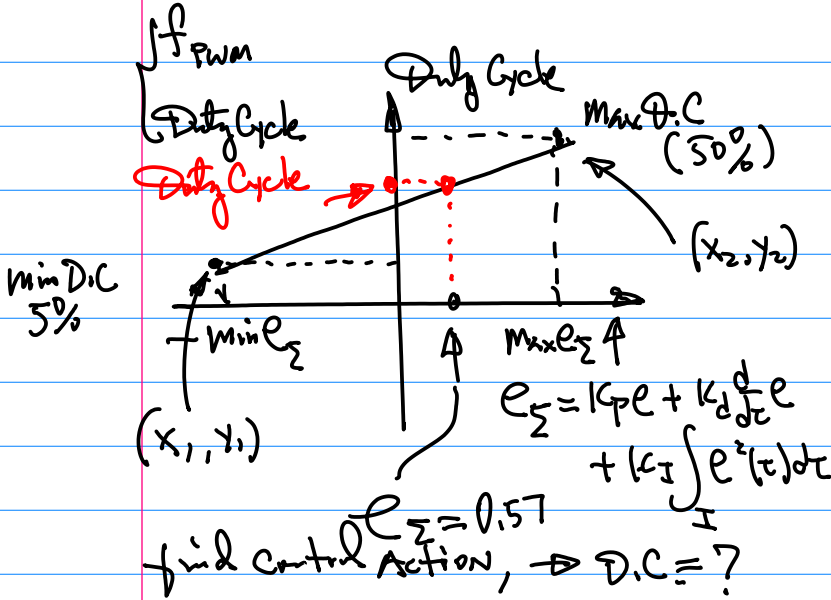
Note: Calculator IS Allowed.

Close Book, Close Notes  
Datsheets if needed will be provided;

Convolution with Kernel(s)  
Table of  $E(t)$ , find  $\frac{d}{dt} E(t)$  Convolution

$$\int K_I e^2(t) dt \approx \sum_{i=0}^I K_I e^2(t_i)$$

Mapping to Control function PWM



To perform init & Config:

1° Binary Pattern for SPR.

Read/modify user Application Programs / Kernel Space Device Driver Program.

C Code for this purpose.

Note: PWM Waveform, e.g. Duty Cycle Calculation.

$\frac{N}{N'}$  for CNT  
for CMP B

$$\frac{f_{CLK}}{f_{PWM}} = N$$

$$f_{PWM} = \frac{PCLK}{(Pres+1) * DVR}$$

$$\% (D.C) * N = N' \Rightarrow \text{CMP}$$

Implementation: Reference Platform ARM11

Architecture CPU Block Diagram  $\Rightarrow$  memory map.

232 (4 KB)

8 Equal Bank

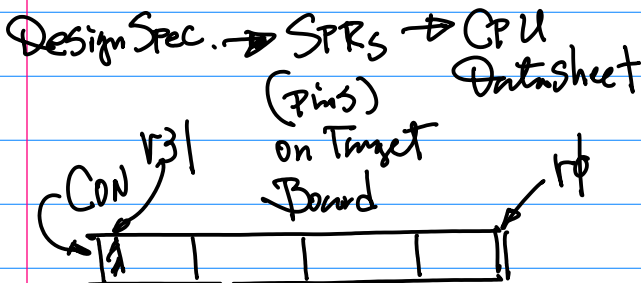
3 Address Bits

$a_{31} a_{30} a_{29}$

\* SPRs.  
(Peripheral Controllers)

SPWM  
GPP

GPX CDN ✓  
GPX DAT  
TCNT B ✓  
TCMP B



Pre-request  $\left\{ \begin{array}{l} 1^\circ \text{ O.S Source Distribution} \\ 2^\circ \text{ Tool Chain Distro.} \\ 3^\circ \text{ "Cross Comp" Datasheet.} \end{array} \right.$

Tool Chain Installed Running make menuconfig

Continued  $\rightarrow$   $\backslash$  Conf (at \drivers  
 $\sim$  \char)

Script. Add your  
 Device Driver

make menuconfig

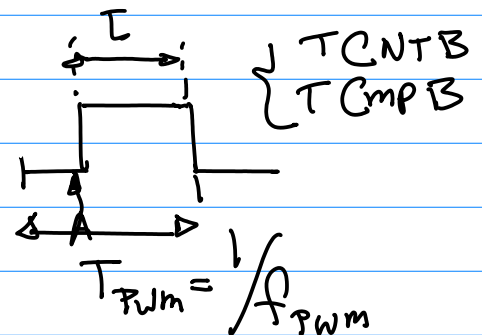
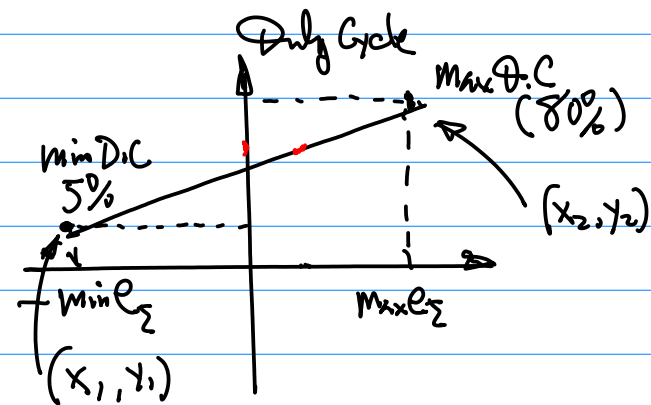
invoke your Change,  
 Compile & Build  
 (module Only for  
 Simplicity Purpose)

Object "KO".

Copy "usb"  $\rightarrow$  Upload  
 by "CP" Copy  
 Command to  
 your target

"insmod" mytest.ko (To make  
 it as a part of kernel Image)

Run your user application  
 Program (By Calling the module)





April 5th (Monday)

1. Midterm Graded, the key was posted online, github, search under folder 2014S, "Key"

2. 2nd half of this course.

I/IOT (Industrial IOT)

Sensors I/F { Digital Sensors - I2C I/F.  
Analog Sensors - ADC

FF.T to find / Characterize  
Analog Sensor Data  
(Nyquist Theorem)  
Validation w/ Sensor Data.

OpAmps to Build Processing CKT.

"SPICE" Simulation.

Example: LSM303

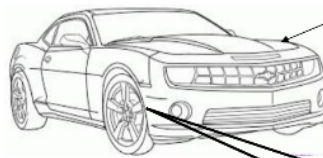
Note: Next Project use LSM303.

CMPE242-Embedded-Systems- / 2018S-16-AngularSensing-i2c-LSM303- final HL 2017-3-13.pdf

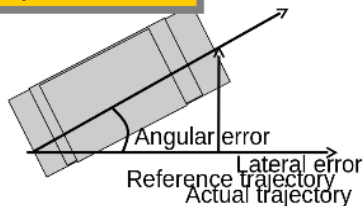
## Sensors for Driving Direction and Turning Angle

eCompass module:  
3D accelerometer and 3D magnetometer

Caution: Steering sensor input is not necessarily the real angle of the vehicle, "skipping" may occur



Use LSM303 or equivalent to sense the direction of the vehicle



The LSM303DLHC includes an I2C serial bus interface that supports standard and fast mode 100 kHz and 400 kHz. The system can be configured to generate interrupt signals by inertial wake-up/free-fall events as well as by the position of the device itself.

larry Li, Ph.D. April 2015

## 3D Accelerometer and 3D Magnetometer LMS303

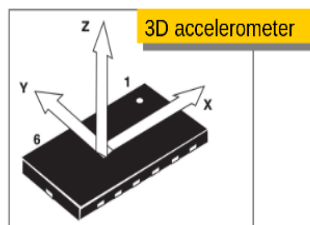
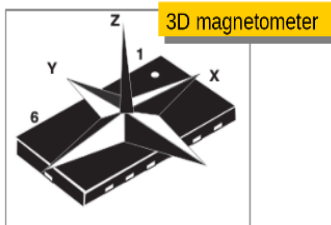
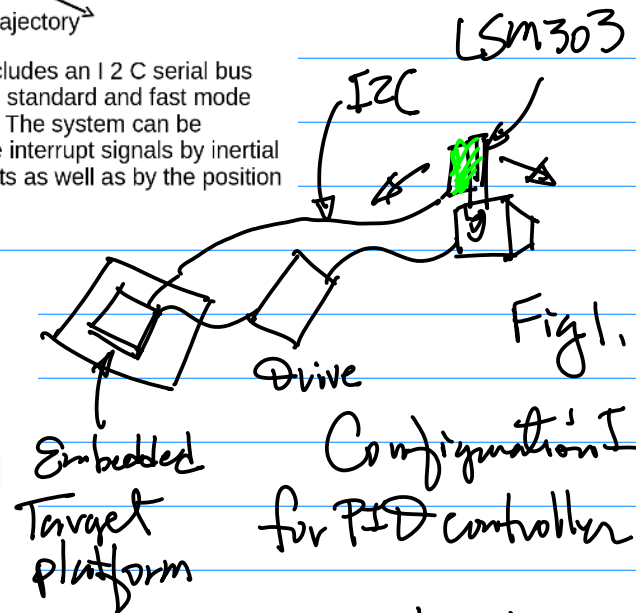


Table 9

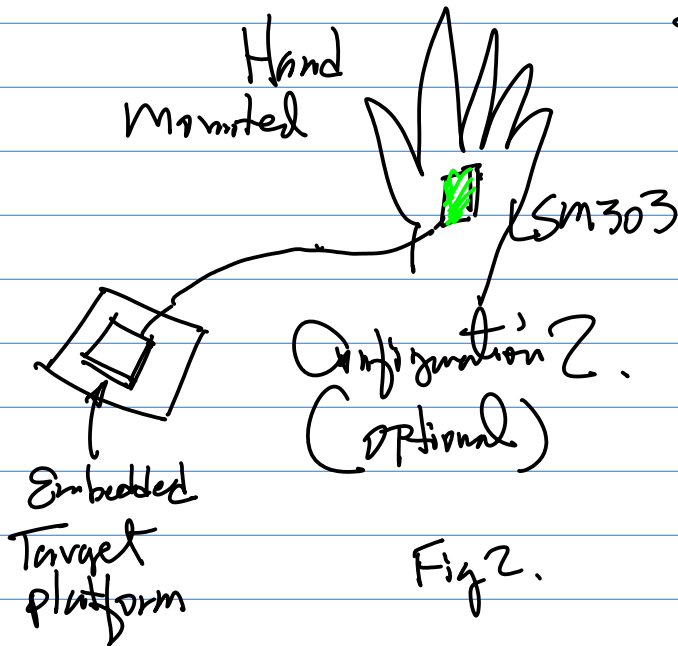
Pin name	Pin description
SCL	I <sup>2</sup> C serial clock (SCL)
SDA	I <sup>2</sup> C serial data (SDA)

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.

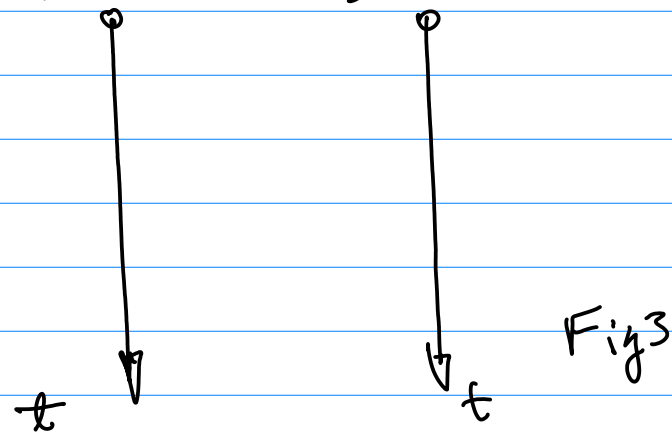


Homework: Implementation  
I2C LSM303 Sensor  
I/F. Due April 16 (Fri)



3. <sup>a</sup> Space-Time Diagram

Time → Space Embedded (Host) LSM303



Submission ON CANVAS

Objective (1) To be Able to Read Sensor Input,

(2) To be Able Config the Sensor.

<sup>b</sup> To Describe "Hand-Shaking"

Three Small Steps.

Step 1. → Step 2 → Step 3

Host Slave "Ack" Data Command.

Transmission will start via Address for Init & Config

1. Note: LSM303 for ST-micro

Sensor Supports { Acceleration

X-y-z axis

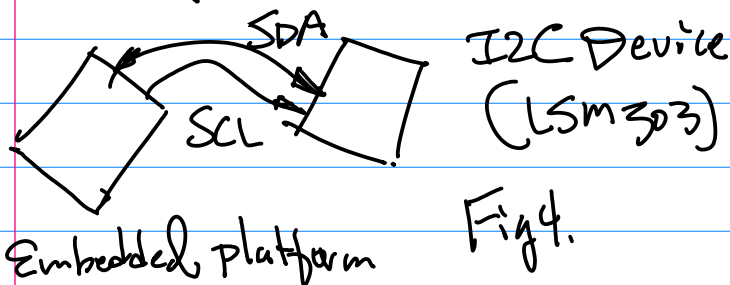
Magnetometer

Temperature

2. I2C

SDA (Serial Data) Bidirectional Data;

SCL (Serial Clock)



\* Be sure read Datasheet to map the Steps of the I/F to Space-Time Diagram.

3. Datasheet Table 9 & 11. TP20.

Notation:

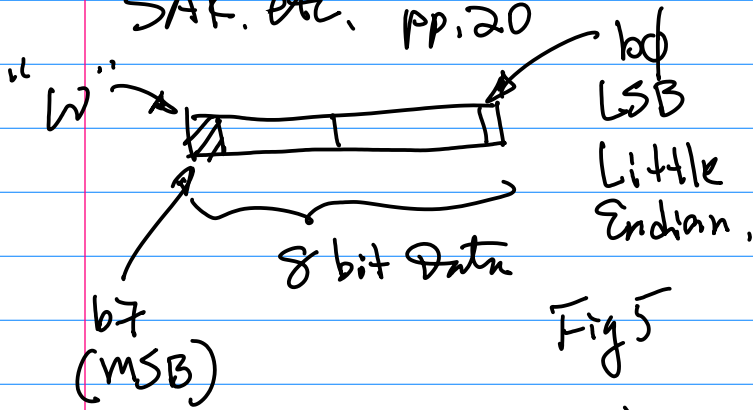
A Frame

1<sup>st</sup> ST → DSP

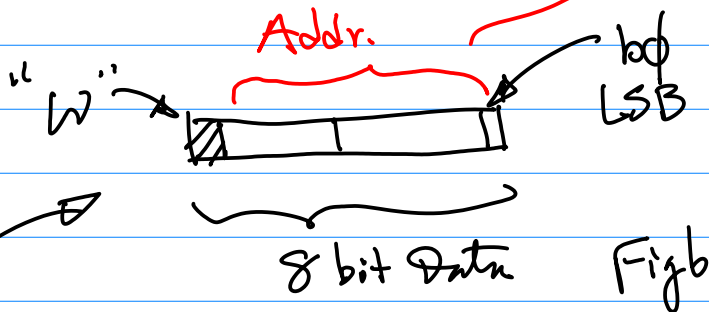
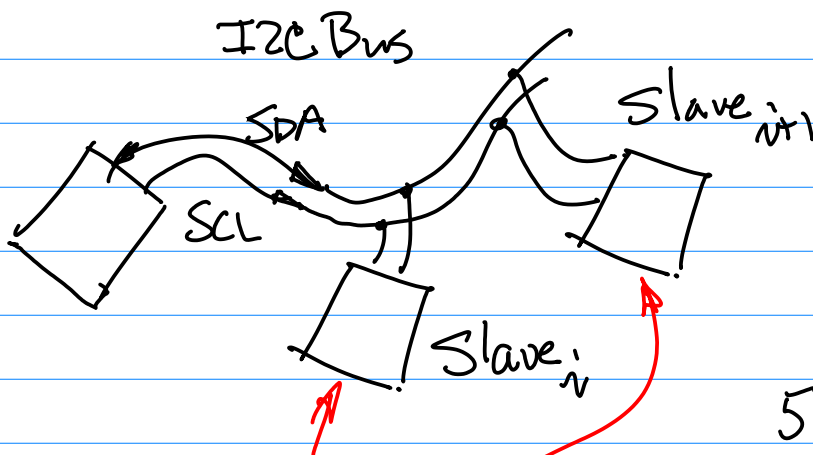
"Start" "Stop"



2. The Notations in Table 1  
 SADR, SADR+W, SNB, DATA,  
 SAK. etc. pp. 20

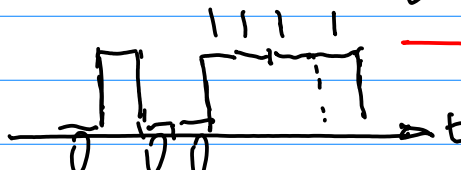


3. from pp. 20 (Datasheet)

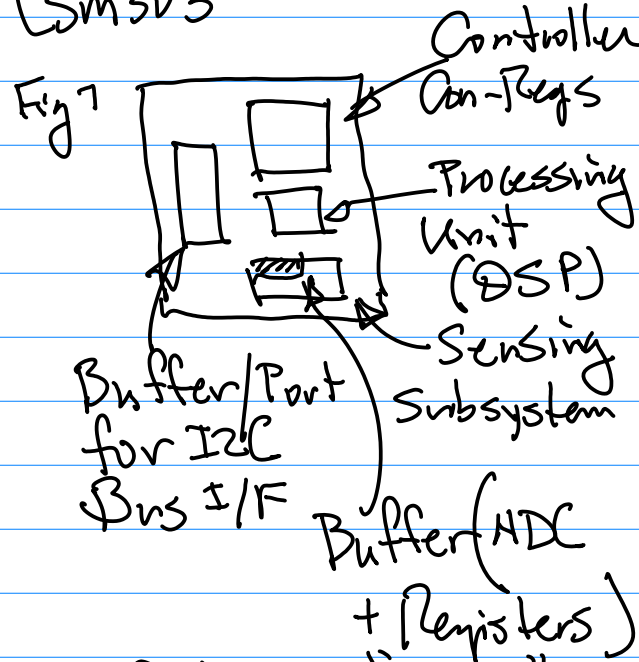


4. 1st Address (7 bits), 2nd Address (lower 7 bits),  
 All from the host (Target platform)

0xf2  
 1111 0010



Consider A Slave device  
 LSM303



2nd Address "SNB" is for Identifying the target inside the Slave Device.

5. 127 Devices possible (Theoretically) on I2C Bus, In Reality this has to be checked by "FAN-IN" or "FAN-OUT".

128 Internal Addresses  
 → Special Purpose Registers.

Most Significant Bit is transmitted first

Example: From Datasheet (LSM303)  
 PP.19 Table 11, 12, 13

Homework (1pt) Due A week  
 from Today, April 14, Due  
 On CANVAS

1° Build I2C Bus Interface  
 with your target platform  
 as a host, LSM303 Slave.  
 To be able:

a Hardware Implementation.  
 (e.g. mount LSM303  
 on the Stepper motor,  
 or mount it to your  
 hand)

b Read Acceleration Data  
 X, Y, Z., Display it on  
 your terminal.

c Read Magnetometer  
 Data and display it on  
 the terminal

Note: Sample code is posted  
 "as is" basis.

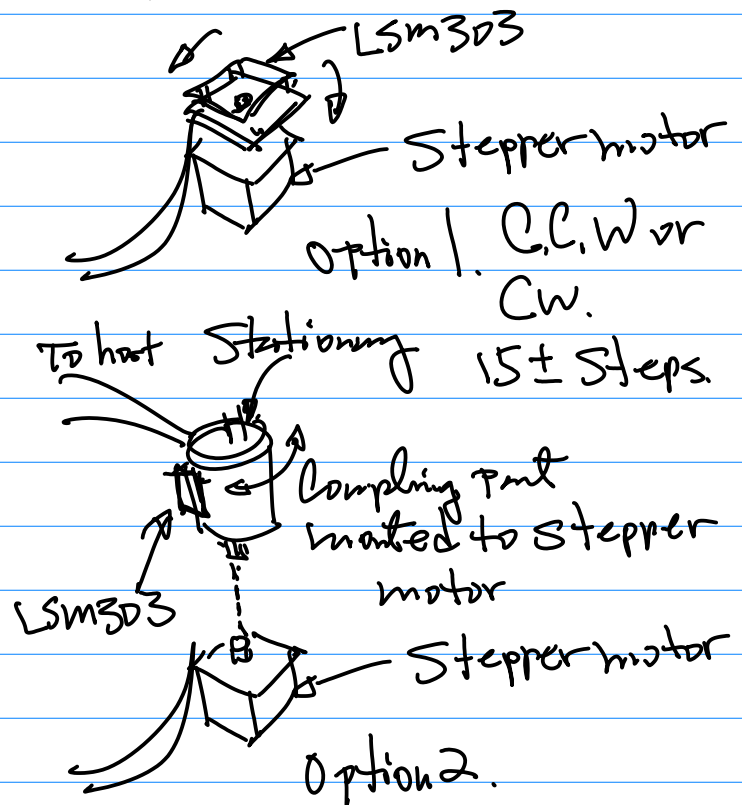
Repo: 20-20215-10-lsm

2° Submission

a Source Code, b Readme.txt

c photo(s) of your  
 Implementation

3x photos, 1 for the entire  
 System (with Laptop); 2<sup>nd</sup> for  
 the Host Side, Expansion  
 Connector is the focus;  
 3<sup>rd</sup> for LSM303



d 5 seconds Video Clip(s)  
 720P or 1080P (1920x1280)  
 Compressed, mp4, avi ?

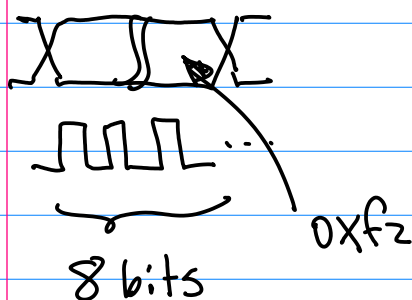
File Naming: first Name 4 Digits -  
 242.zip

Note: Table 11 & 12 (PP 19)

One Byte Writing → Multiple Bytes Writing  
Host Host

Table (Logical Behavior)

Timing (Waveform)



Now, the Address for the Sensor (S)  
and Addresses for Registers  
↓ Control Register — Init & Config  
↓ Data Register

SAD[1:0] Address + SAD[0] for W/R  
↓ = 1 for W  
↓ = 0 for R

Note: Use info from Table 14  
to fill in SAD+W, SAD+R  
in tables 11~13.

Note: Section 5.1.3 Magnetometer

Example: Table 18. Control Register A  
for Magnet



Tech Spec



Binary Pattern

(for 400Hz)

+ 1

Tech Spec:

- i. 400Hz Data Rate
- ii. X-Y axis.
- iii. Sensor Active (No Low Power)

CTRL\_REG1\_A[3] = 0

CTRL\_REG1\_A[2] = 0

CTRL\_REG1\_A[1] = 1

CTRL\_REG1\_A[0] = 1

0x3

Hence,

CTRL\_REG1\_A[7:0] =  
0x73 ✓

Section 7.1.8 Status  
Registers

Section 7.1.9 ~ 7.1.11

Data Registers.

2's Complement form

↓  
1's Complement  
By Negation  
"0" → "1"  
"1" → "0"

CTRL\_REG1\_A[7:0] 8 bits.

CTRL\_REG1\_A[7:4] = 0x7

April 12 (Monday)

37

Homework Extension to April 19 (Monday)

Industrial Analog Sensor I/F Design

Example: NH<sub>3</sub> Analog Sensor (Ion Selective Electrode) Interface.

1. ADC (Analog to Digital Conversion Unit)

LPC1769 6x ADC  
ARM11 ADC

Pic Stand Alone ADC  
Jelsoft NANO/Tx2

1° Sampling Rate  
500 KSPS

2° Quantization  
8 bits, 10 bits, 12 bits.

3° SPI/I2C  
Faster  $\leq 100 \text{ MHz}$

Start with Characteristic Curve

Linearization

OPAMP Preprocessing CKT

Optimized dynamic (Input) Range to ADC

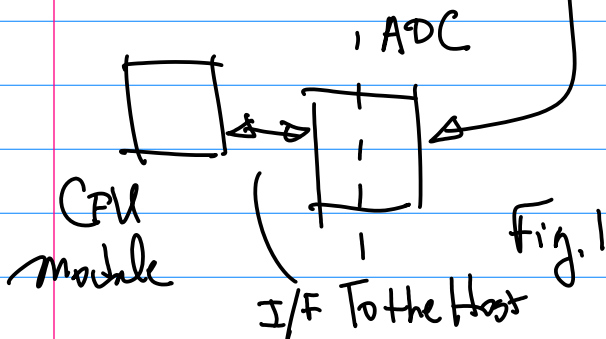
4. OPAMP's for preprocessing

Addition  
Subtraction  
Division  
multiplication

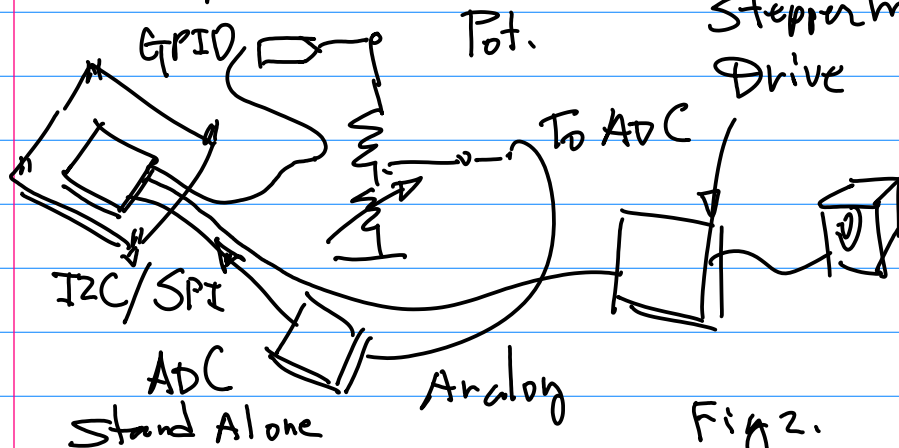
Math. Operations for preprocessing

Example: from Datasheet  
2018S-22-NH3  
TP13, Fig. 14.

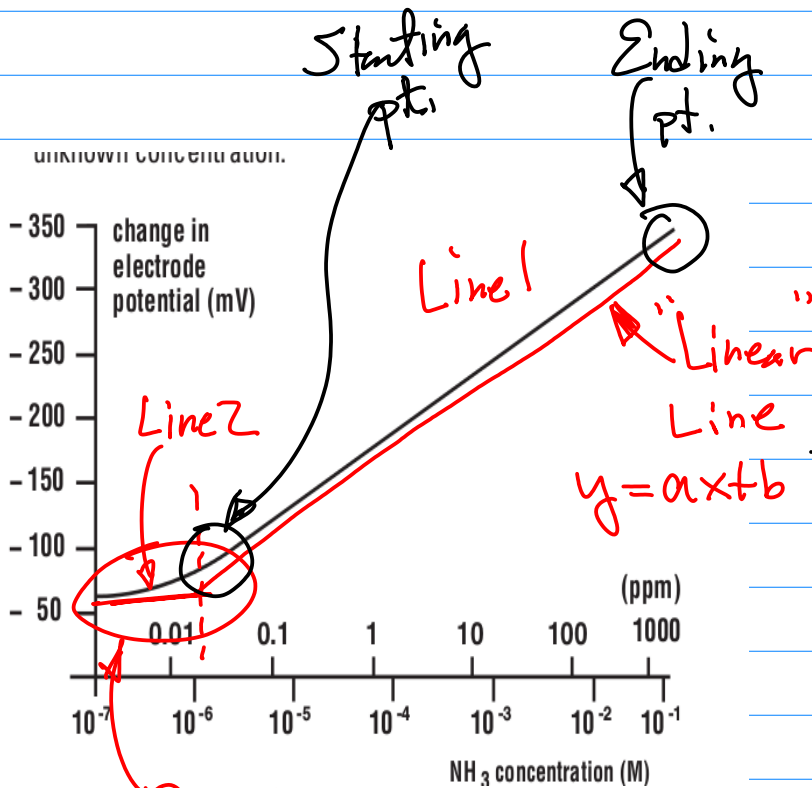
2. Prototype to Build



Stepper motor Drive



Step. Linearization Characteristic Curve



Step 2. Map the Dynamic Range of Sensor to the Dynamic Range of ADC Input.

2.1 "Shifting" OR Offset to move the Characteristic Curve to the origin.

Non-Linear.  $y = ax^2 + bx + C$  2nd order

Linearization: Line Equation(s) to replace Non-Linear Curve.

Simplification: Just Keep Line 1.

Note: Linearization By Identifying Starting and Ending points.

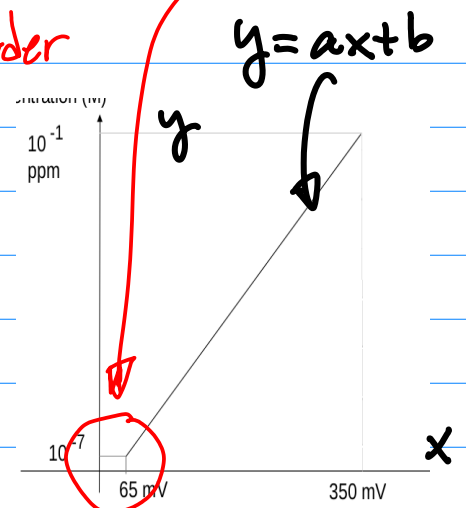
$(V_0, C_0)$

$(V_N, C_N)$

Voltage V.S. Concentration

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

$$(x_1, y_1) \rightarrow (V_0, C_0), (x_2, y_2) \rightarrow (V_N, C_N)$$

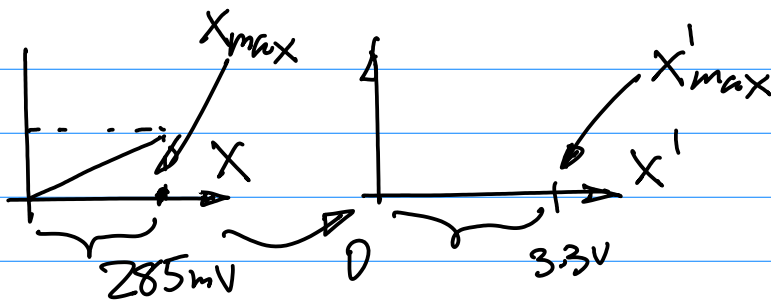


$x_{\text{New}}$  is changed

By "Shifting"  $ax \rightarrow a(x - \Delta x)$

2.2 Dynamic Range

fits to ADC Dynamic Range



$$\frac{X'_{max}}{X_{max}} = \frac{3.3V}{0.285V} \stackrel{\text{Let}}{=} A \quad \dots (2)$$

Gain  
("Scaling Factor")

$$X \cdot A = X' \quad \dots (3)$$

Review  
OpAmp { Inverting Configuration opAmps  
Non-Inverting Configuration