

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

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

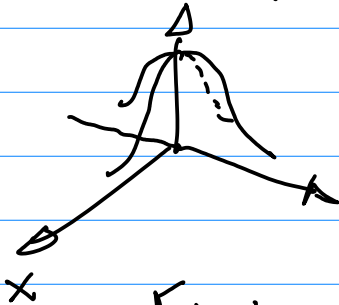
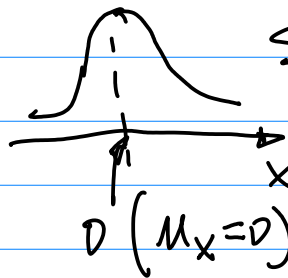


Fig 1.



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

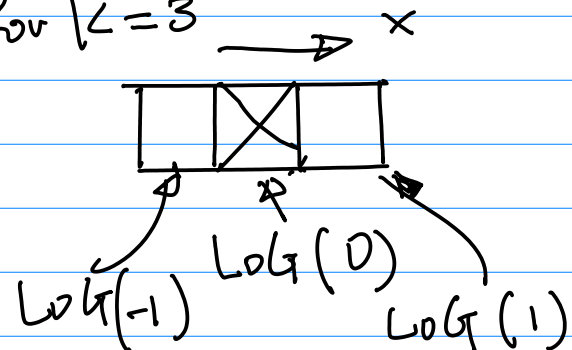
From Eqn (4), ppt from the github
2018S-15-lec6-V3 ...

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 Error $\nabla^2 G(x, 0)$.

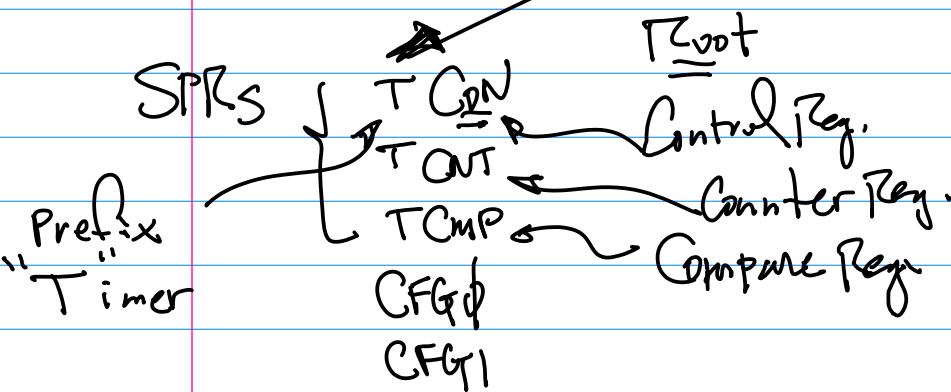
$\hat{=}$ Identify the Center Reference $\hat{=}$ from $\text{LoG}(x)$ (or $\nabla^2 G(x, 0)$). map it to the kernel

Solve for y.
 Driven Implementation. $\left\{ \begin{array}{l} f_{PWM} \\ D.C. \end{array} \right.$

2018S-10-0 ~
 PWM Driver

Add Duty Cycle Function to
 Device Driver.

Theoretical Aspect
 Implementation C Code.



Pwm Output Square Waveform

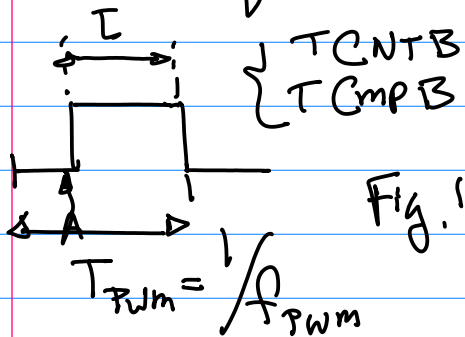


Fig. 1

$$f_{PWM} = \frac{CLK_P}{(Prescaler+1)(divider)} \dots (3)$$

PP1118, CPU Datasheet
 Prescaler: 8 bit, [0, 255]
 Divider: 1, 2, 4, 8, 16

Note CFG/Con are responsible
 for Setting Prescaler/Divider
 value.

$$f_{PWM} = \frac{50 \times 10^6}{(Pres+1) \cdot Div} \dots (*)$$

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

From CPU Datasheet CFG &
 Con.

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

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

$$(Pres+1) Div = 25 \times 10^3$$

Let Div = 16,
 Solve for Pres.

$$Pres+1 = \frac{25 \times 10^3}{16}$$

$$Pres = \frac{25 \times 10^3 - 16}{16} \approx 255$$

Iteration,

Change PCLK to 10MHz,
 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_P
 try CLK_P $\approx 2 \times 10^6$. 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. 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 ϕ , 1

Architecture \rightarrow Mem \rightarrow SPR
Code \leftarrow

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

Programming Requirements, No Program
However! ^{Writing} 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

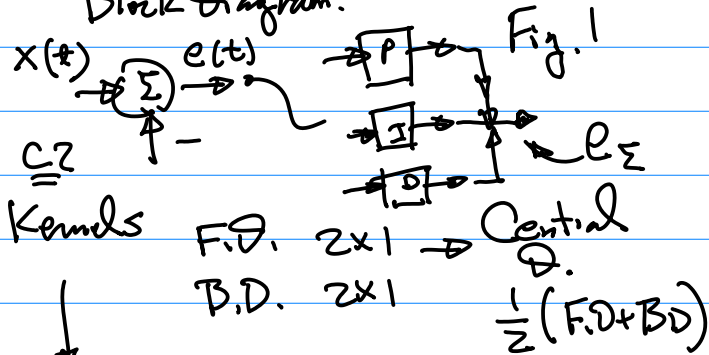
Resistance Value
Calculation

\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.

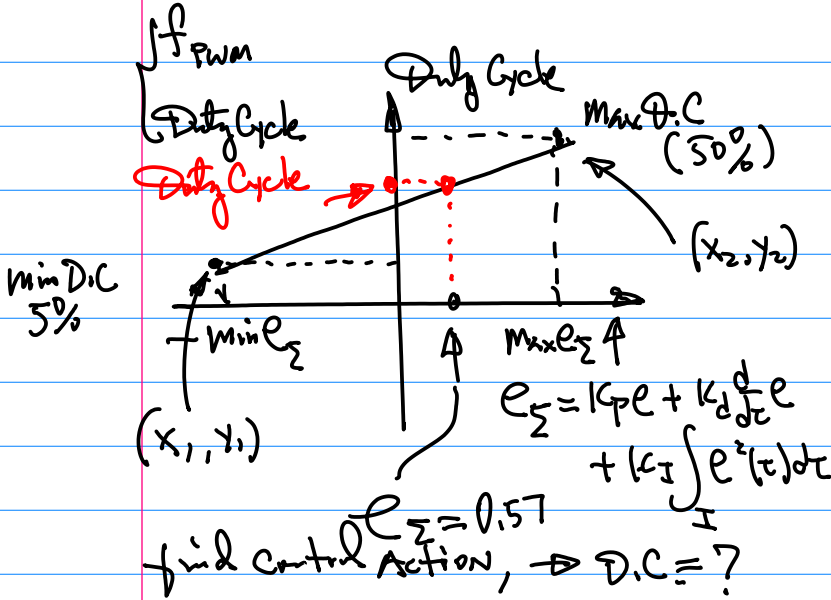
Note: Calculator IS Allowed.

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

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

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

Mapping to Control function PWM



To perform init & Config:

1^o 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
 $\frac{N'}{T}$ 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

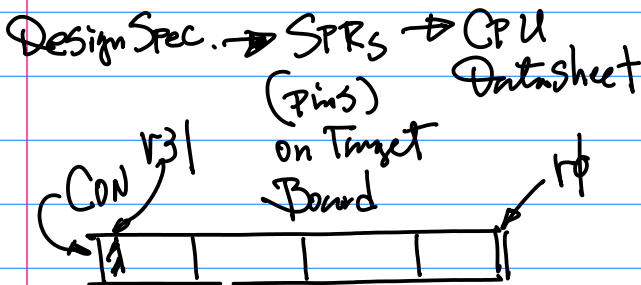
Architecture CPU Block Diagram \Rightarrow memory map. 2^{32} (4 GB) 8 Equal Bank

* SPRs. (Peripheral Controllers)

SPWM
GPP

GPX CDN
GPX DAT
TCNT B
TCMP B

$a_{31} a_{30} a_{29}$ 3 ADDR BITS



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

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

Script. Add your
 Device Driver

make menuconfig

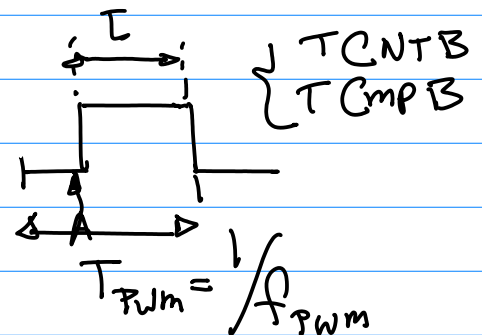
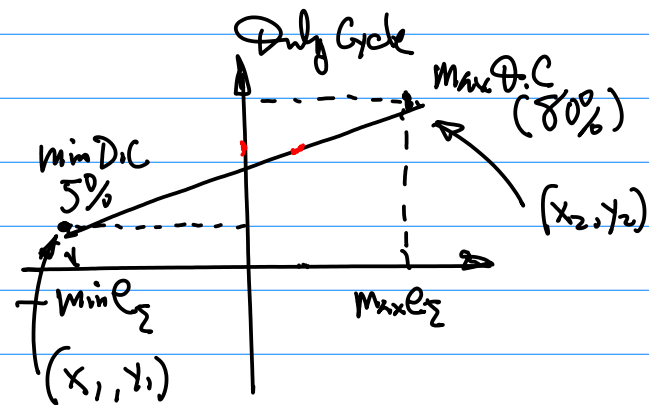
invoke your Change,
 Compile & Build
 (module Only for
 Simplicity Purpose)

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

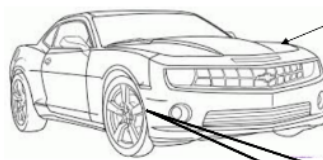
Note: Next Project use LSM303.

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

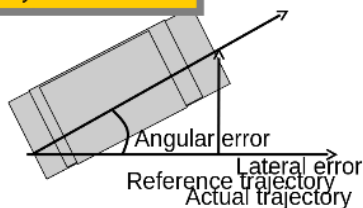
Sensors for Driving Direction and Turning Angle

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

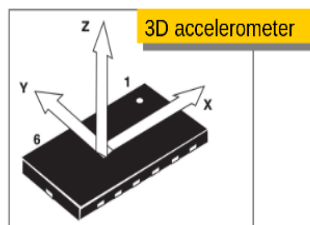
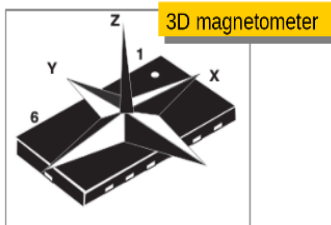
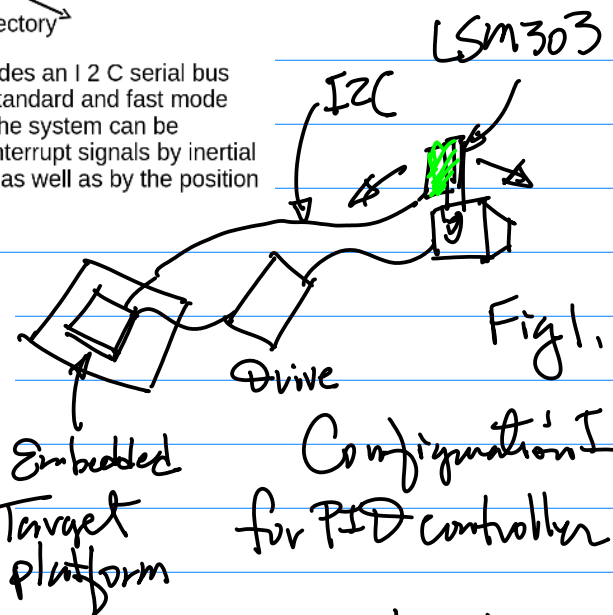


Table 9

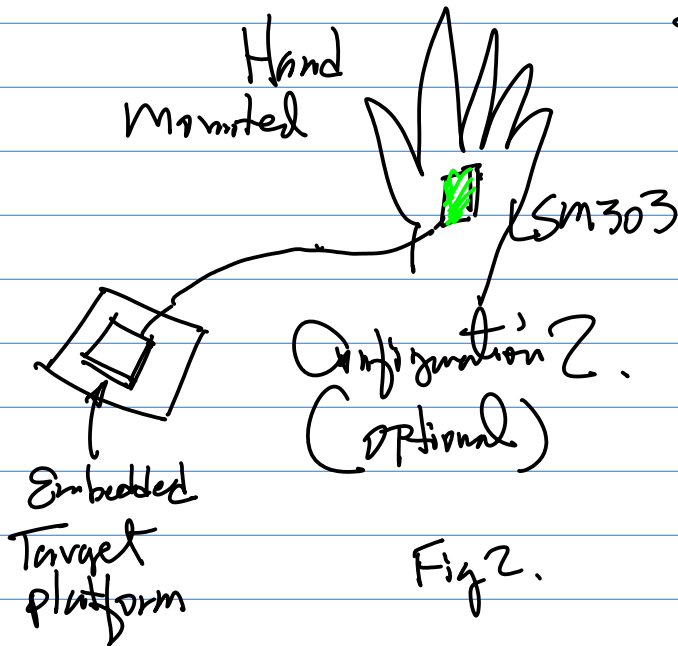
Pin name	Pin description
SCL	I2C serial clock (SCL)
SDA	I2C serial data (SDA)

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.

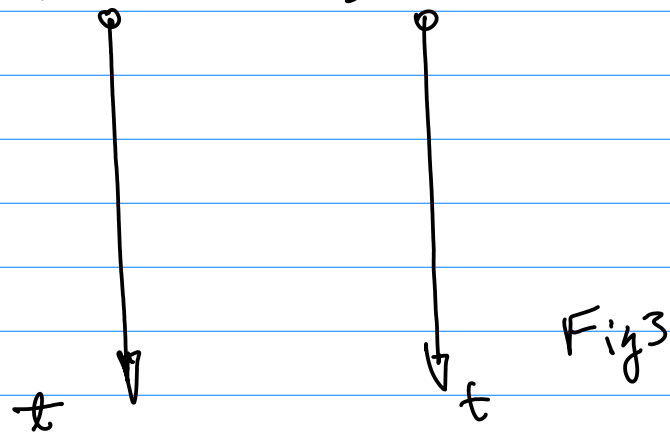


Homework: Implementation
I2C LSM303 Sensor
I/F. Due April 16 (Fri)



3. ^a 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.

^b To Describe "Hand-Shaking"

Three Small Steps.

Step 1. → Step 2 → Step 3

Host Slave "Ack" Data Command.

Transmission will start

to the Target via Address for Init & Config

1. Note: LSM303 for ST-micro

Sensor Supports { Acceleration

X-y-z axis

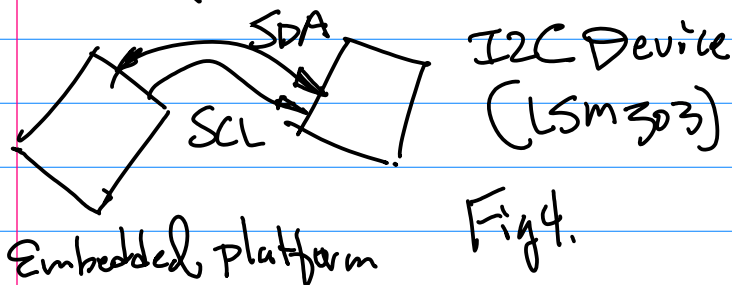
Magnetometer

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.

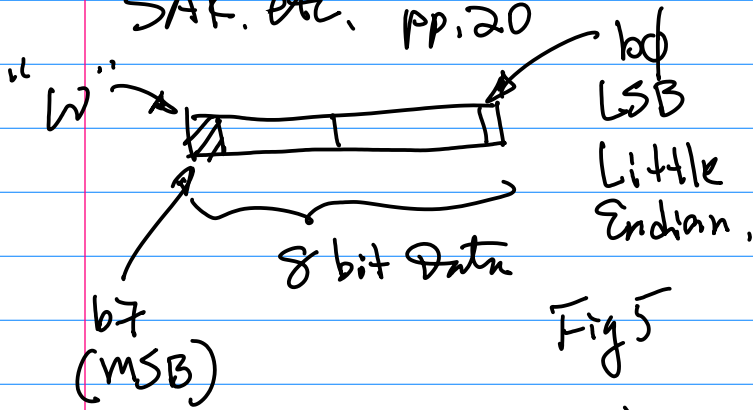
Notation:

A Frame

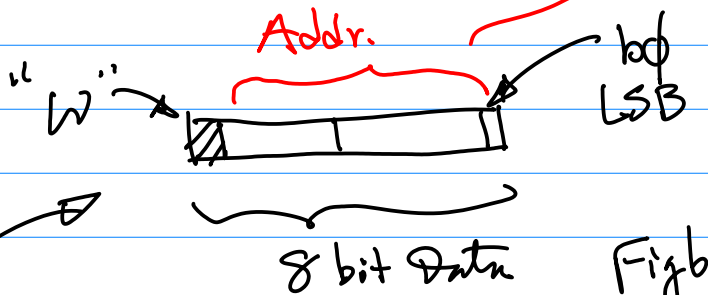
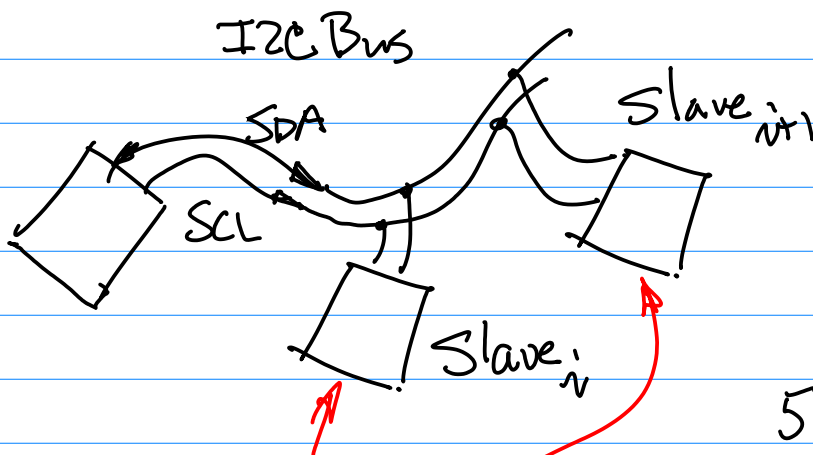
1st ST → DSP

"Start" "Stop"

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

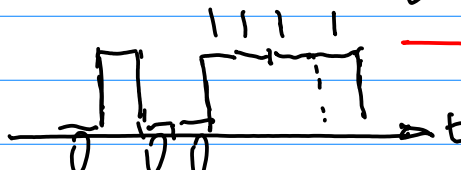


3. from pp. 20 (Datasheet)

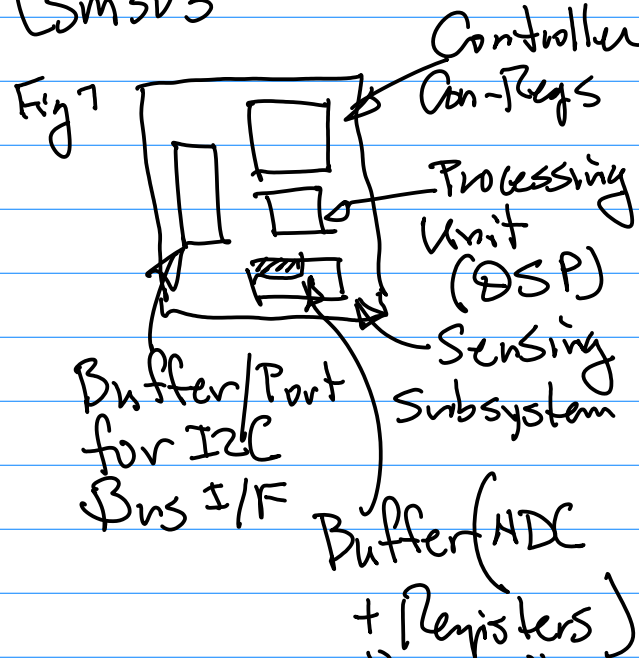


4. 1st Address (7 bits), 2nd Address (lower 7 bits),
 All from the host (Target platform)

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.

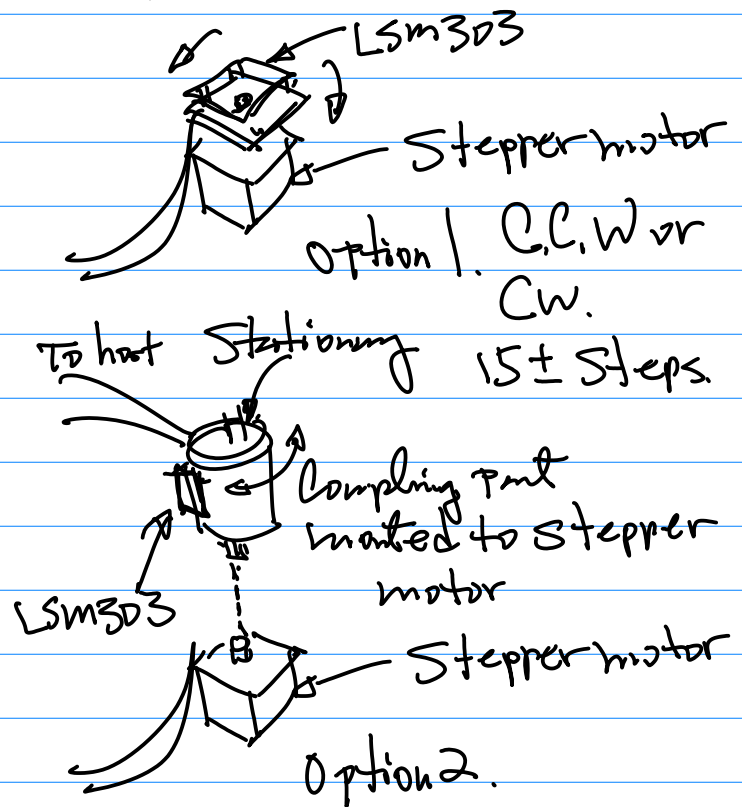
Repo: 20-20215-10-lsm

2° Submission

a Source Code, b Readme.txt

c photo(s) of your
 Implementation

3x photos, 1 for the entire
 System (with Laptop); 2nd for
 the Host Side, Expansion
 Connect is the focus;
 3rd for LSM303



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

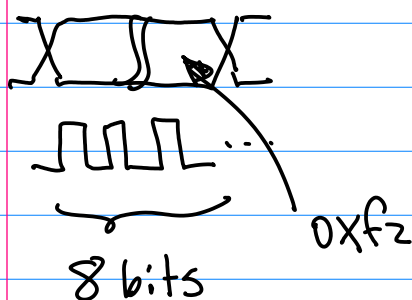
File Naming: first Name 4 Digits -
 242.zip

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



CTRL_REG1_A [7:0] 8 bits.

CTRL_REG1_A [7:4] = 0x7

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"

April 12 (Monday)

37

Homework Extension to April 19 (Monday)

Industrial Analog Sensor I/F Design

Example: NH₃ Analog Sensor (Ion Selective Electrode) Interface.

1. ADC (Analog to Digital Conversion Unit)

LTC1769 6x ADC

ARM11

ADC

Pic Stand Alone ADC
Jelsan NANO/Tx2

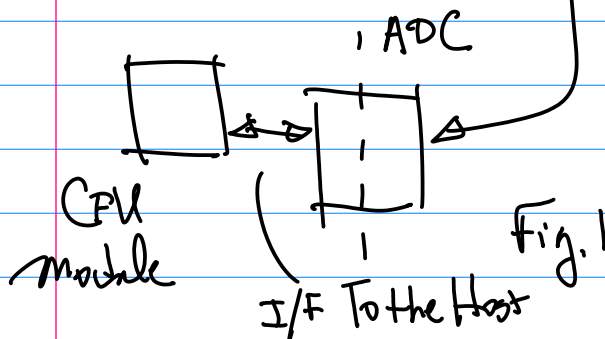
1° Sampling Rate
500KSPS

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

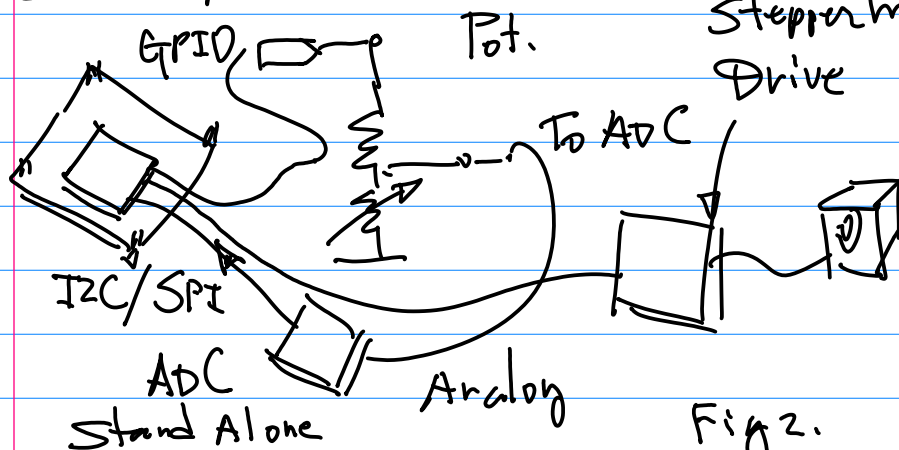
3° SPI/I2C
Faster $\leq 100\text{MHz}$

4. OP Amp's for preprocessing

Addition
Subtraction
Division
multiplication
Math. Operations for preprocessing



2. Prototype to Build



3. Analog Interface Design.

Start with Characteristic Curve

Linearization

OP Amp Preprocessing CKT

Optimized dynamic (Input) Range to ADC

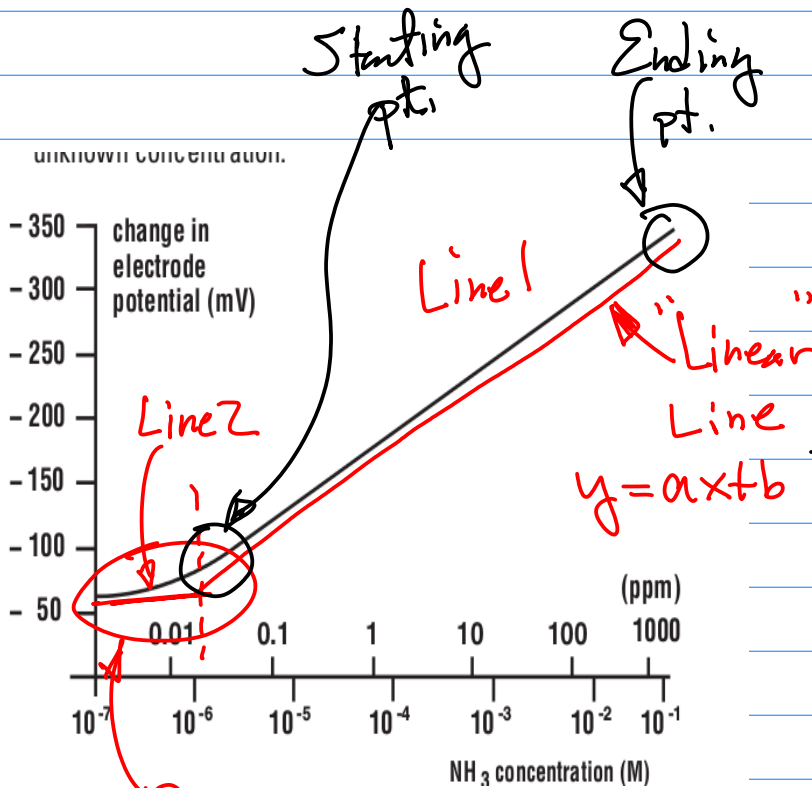
4. OP Amp's for preprocessing

Addition
Subtraction
Division
multiplication
Math. Operations for preprocessing

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

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.

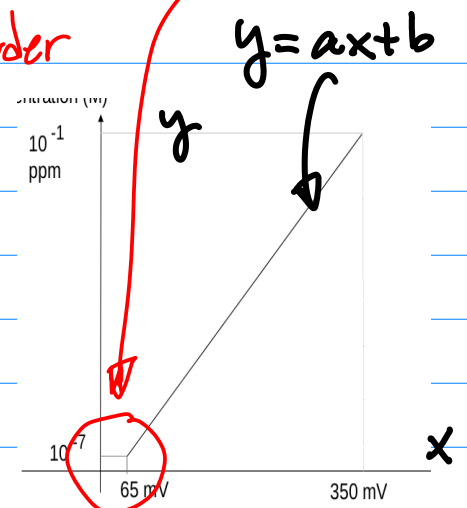
(V_0, C_0)

(V_N, C_N)

Voltage V.S. Concentration

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

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



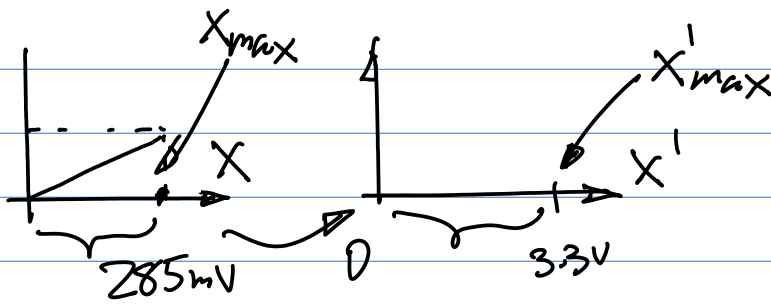
x_{New} is changed

By "Shifting"

$$ax \rightarrow a(x - \Delta x)$$

2.2 Dynamic Range

fits to ADC Dynamic Range



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

Gain
("Scaling Factor")

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

Review of OpAmps
 { Inverting Configuration
 { Non-Inverting Configuration

April 14 (Th).

OpAmps for Pre-processing Design

1. OpAmp {
 - a Very High Input Impedance
 - b Very Big Open-loop gain
 - c Very Small output Resistance

2. Using OpAmps As Basic

Building Blocks (B^3) for

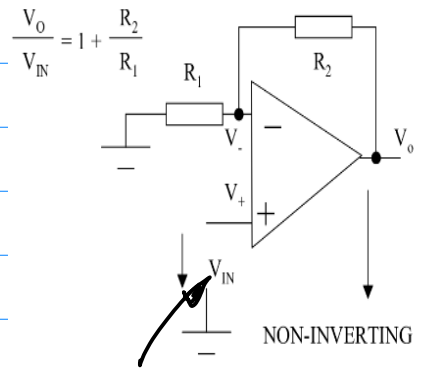
Arithmetic Computation {

- Add / Sub
- Multiplication /

Integral / Derivatives Division

3. Configurations {

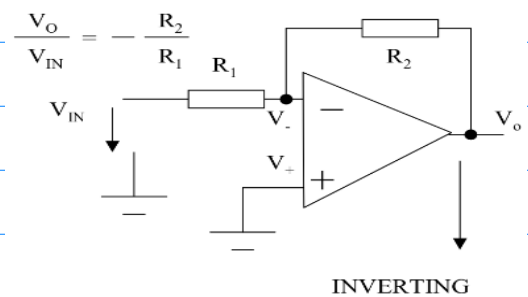
- Non-Inverting ~
- Inverting ~



a Input: Positive Polarity;
 b feedback CKT
 $V_{out} \rightarrow$ Input (V_{in})

Via R_f
 c Draw the CKT.

$$A = \frac{V_{out}}{V_{in}} = 1 + \frac{R_f}{R_1} \quad \dots (1)$$



a Input: V_{in}
 b feedback CKT
 $V_{out} \rightarrow V_{in}$ via R_f

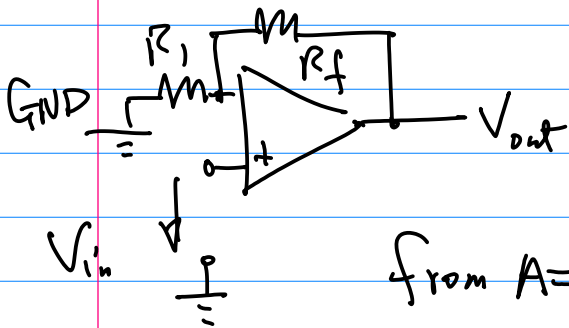
$$A = -\frac{R_2}{R_1} \quad \dots (2)$$

4. Use OpAmp CKTs for Math. Operations.

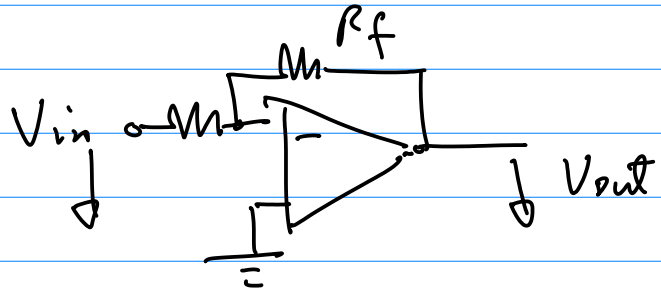
= Addition. $X_1 + X_2$

More than One Approach is possible, But Let's use Inverting Configuration

Non-Inverting Configuration

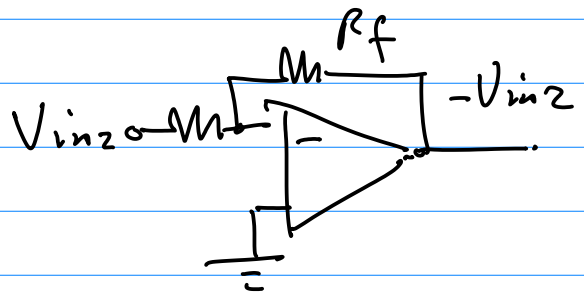


from $A = 1 + \frac{R_f}{R_1}$



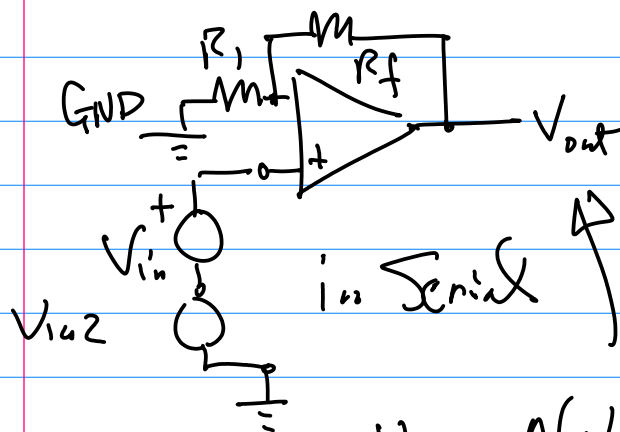
$$\frac{V_{out}}{V_{in}} = A = -\frac{R_f}{R_1}$$

Or, $A = \frac{V_{out}}{V_{in}}, V_{out} = A \cdot V_{in}$
 $\dots (3)$



$V_{in1} + V_{in2}$ Addition

Let input Circuit as follows



make $V_{out} = -V_{in2}$, Let
 $R_f = R_1 = 1 \text{ k}\Omega$

(741 opAmp, 384

Note: Not too Big Current, Quad-Pack)

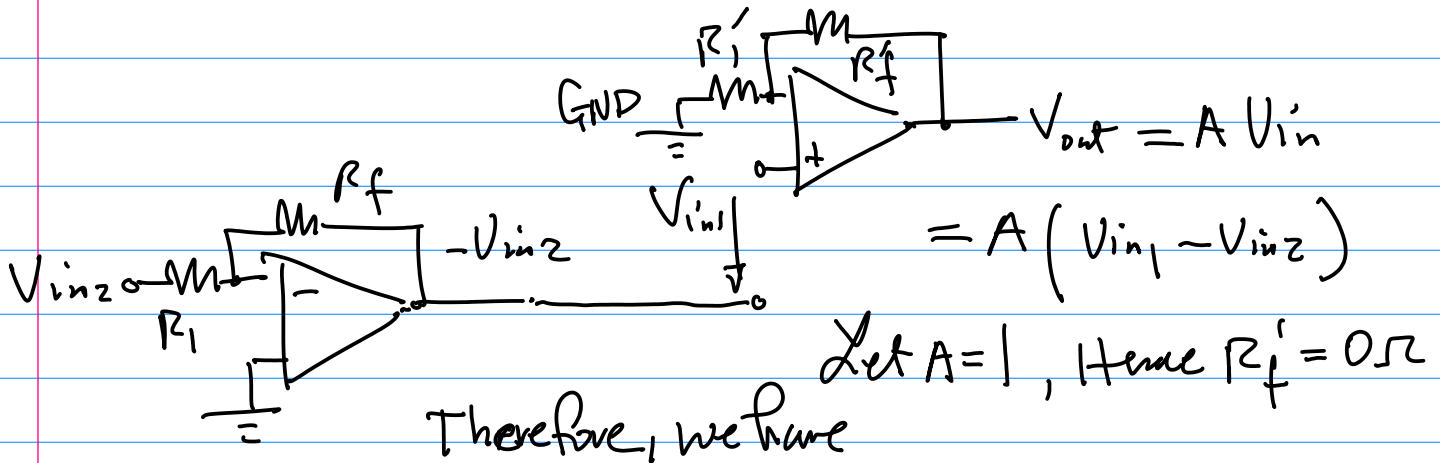
Power Consumption is going to be a problem.

Not too small, Noise will distort the Signal

Let $A=1, \rightarrow R_f = 0 \Omega$

= Subtraction $X_1 - X_2 (V_{in1} - V_{in2})$

Then, Combine it with Add CKT
 SO, we have $V_{in1} - V_{in2}$



Subtraction

$$V_{out} = V_{in1} - V_{in2}$$

≡ multiplication. $y = ax$
 $a > 1$

Use Non-Inverting
 Configuration

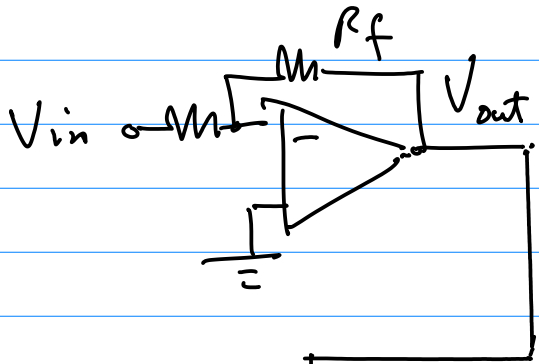
$$\frac{V_{out}}{V_{in}} = A \quad \left| \quad A = 1 + \frac{R_f}{R_1}\right.$$

$$V_{out} = A \cdot V_{in} = \left(1 + \frac{R_f}{R_1}\right) V_{in}$$

Note: For Linear System, the multiplication is done by multiplying a gain, But Not Another x (or V_{in}).

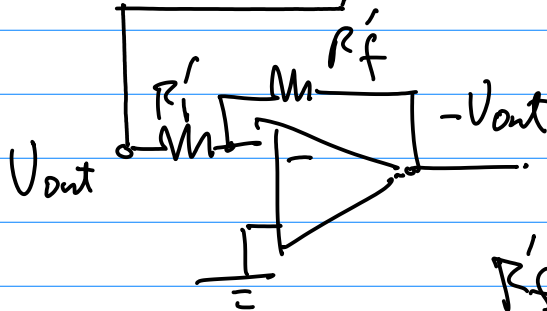
≡ Division (is a multiplication!)
 for the multiplier A less than 1.

Two Stage Inverting Configuration, 1st Stage does the division But with negative Sign, 2nd stage with gain = 1, But change to positive by 2nd negative.
 Example: 2 stage Inverting



$V_{out} = -\frac{R_f}{R_1} V_{in}$, where $\frac{R_f}{R_1}$ is a fractional Number for division, for example 0.32, $\frac{R_f}{R_1} = 0.32$, if $R_1 = 1k$

$$R_f = 320\Omega$$



to make gain = -1

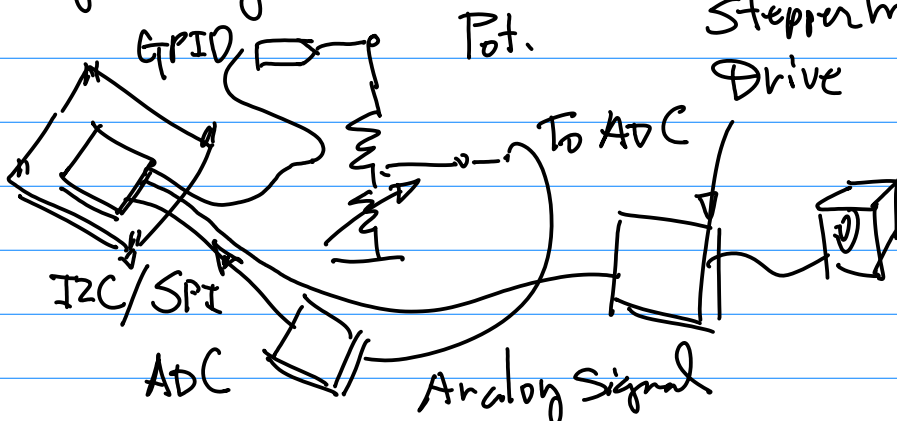
$\frac{R_f}{R_1} = 1$, Let $R_1 = 1k\Omega$ then solve for

$$R_f = R_1 = 1k\Omega.$$

Note: Analog Sensors have to meet the Output Current Requirement, e.g. $4 \sim 20mA$

April 19 (mon).

1st Preparation for the Coming Project. Fig 2. PP. 37.



Project Due May 9 (Sunday) 11:59 P.m.

April 19 (mon)

Design ON Preprocessing CKT.

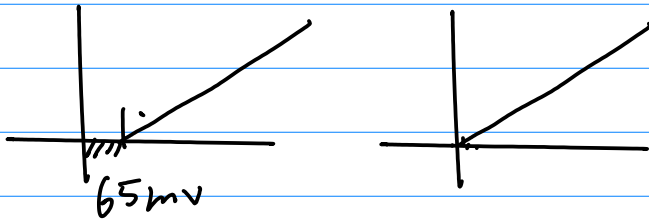
Example: See Fig. on github

Characteristic Curve.

Shifting \rightarrow Enlargement of the Dynamic Range.

a Shifting, Add/Sub.
b Dynamic Gain Range mult.

Shifting (Subtraction).



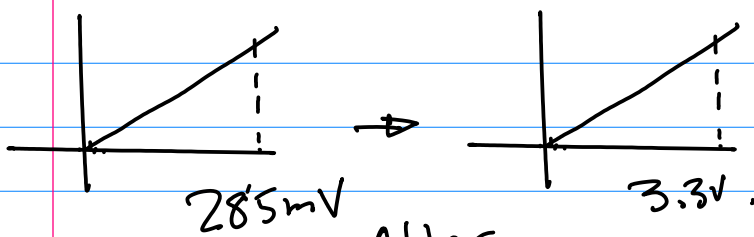
Inverting Configuration

Desired Shifting

$$\frac{V_D}{V_{in}} = - \frac{R_f}{R_1}$$
 Output from the Reference (External Voltage Source)

$$\frac{65 \times 10^{-3}}{5.0} = \frac{R_f}{R_1}, \text{ Let } R_f = 1 \text{ K}\Omega \dots (1)$$

Now, Dynamic Range Design



$$\text{Gain} = \frac{3.3 \text{ V}}{285 \text{ mV}} = \frac{3.3}{0.285} \approx 11.58 \dots (2)$$

After Before

Since Gain is positive, we have Non-inverting configuration

$$A = \left(1 + \frac{R_f}{R_1}\right), \text{ Substitute (2) into this Eqn.}$$

$11.58 = 1 + \frac{R_f}{R_1}$
 Let $R_1 = 1 \text{ K}\Omega$, find/solve for R_f . Hence $R_f = 10.58 \text{ K}\Omega$
 Then, Integrate OPamps Together to form Preprocessing CKT.

Note:

1° Analyze Sensor Characteristic Curve, Define Arithmetic/Math Operations, for Shifting and magnification (of the gain, to cover 3.3v Dynamic Range)

2° To be Able to use Inverting And/OR Non-inverting Configuration to Realize the design Requirements, e.g. Shifting, and magnification

Now, Let's consider the ADC

Design Scope:
 System Level for ADC
 Data Validation

D.F.T. (Discrete Fourier Transform) \rightarrow Power Spectrum

Nyquist Sampling Theorem.