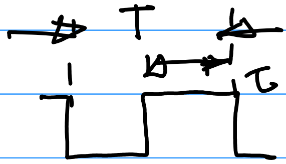
March 16 (Wed)

Topics: 1° PWM Architecture/Hardware Aspect — Waveforms, Timing Diagrams, SPRs
2° LSM303 Sensor I2C

Example: PWM Discussion.

f_{PWM}
Duty Cycle

One Period



Duty Cycle = τ / T ... (1).

Square Wave: D.C. = 50%.

Architectural Aspects:

2021F-105-#0-cpu-arm11-2018S-29-...

P1106

6410X_UM

32 PWM TIMER

This chapter describes the functions and usage of PV

C. Prescaler for f_{PWM} Config
8 bit

PP1107

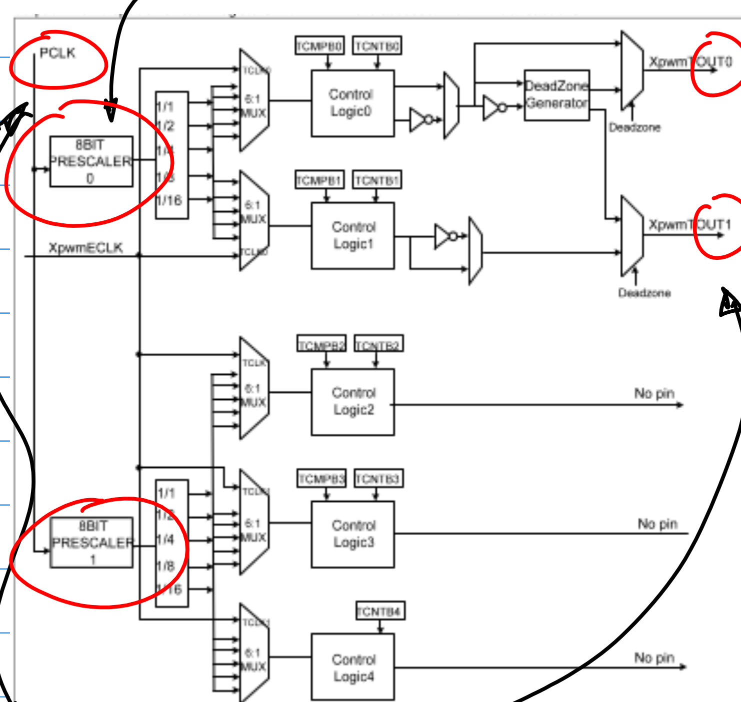
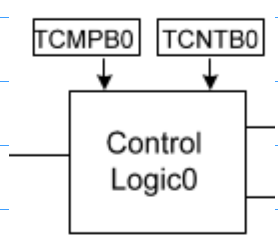


Figure 32-1. PWMTIMER Clock Tree Diagram

a. Input: PCLK peripheral clock.
1/2, 1/4, 1/8, ..., of the System Clock

b. Output (2 outputs)

d. Special Purpose Registers

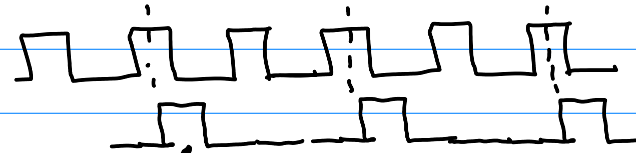


CONF (Configuration Register)
CNT (Control — "Count")
CMP (Comparision)

define f_{PWM} & Duty Cycle.

- 2022S-107e-pwm-waveform-v3-2018-3-4.jpg
- 2022S-107f-pwm-specialPurposeRegister-v3-2018-3-4.jpg
- 2022S-107g-pwm-calculation-v3-2018-3-4.pdf

Note: Background on Counters.



Count By 2

Expand this to A counter for Both integer Number & Fractional Number (in General) → f_{PWM} (for interger Only)

then, use Another Counter to get Duty Cycle.

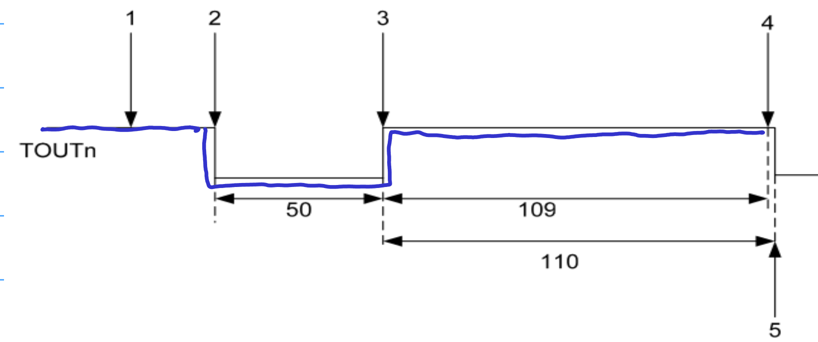
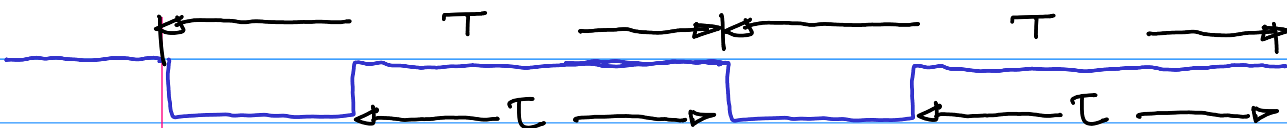


Figure 32-2. Simple Example of PWM Cycle Block Diagram



$T = 1/f_{PWM} \ldots (2)$, $f_{PWM} = \dfrac{PCLK}{(Prescaler+1)DIV} \ldots (3)$

Frequency = PCLK / ({prescaler value + 1}) / {divider value}

Defined By Special Purpose Register.

Suppose $PCLK = \frac{1}{4}(\text{System Clock}) = \frac{1}{4}(800 \times 10^6)$

$= 200 \times 10^6$

Design Guidelines for PWM:

① CNT (TCNTB0) Define f_{PWM}

Control Count

Timer

Buffer0 for PWM0

② CMP (TCMPB0) Defines Duty Cycle

Comparision

③ CONF Configuration Register. Defines f_{PWM}

Example: Suppose $PCLK = 500 \times 10^6$ (MHz), Suppose to Drive A Stepper Motor $f_{PWM} = 2\text{KHz} = 2 \times 10^3$

Find the Counts N for TCNTB0

$\dfrac{PCLK}{N} = f_{PWM} \ldots (4)$

$\frac{PCLK}{N} = f_{PWM}$, Substitute the design Requirements into it.

$$\frac{500\times10^6}{N} = 2\times10^3,$$

$$\therefore N = \frac{500\times10^6}{2\times10^3} = 250\times10^3$$

Verify if TCNTB0 Can hold up to that Number

pp1117

32.4 SPECIAL FUNCTION REGISTERS

32.4.1 REGISTER MAP

Register	Offset	R/W	Descrip
TCFG0	0x7F006000	R/W	Timer Configuration Registe two 8-bit Prescaler and Dea
TCFG1	0x7F006004	R/W	Timer Configuration Registe and DMA Mode Select Bit
TCON	0x7F006008	R/W	Timer Control Register
TCNTB0	0x7F00600C	R/W	Timer 0 Count Buffer Regis
TCMPB0	0x7F006010	R/W	Timer 0 Compare Buffer Re
TCNTO0	0x7F006014	R	Timer 0 Count Observation
TCNTB1	0x7F006018	R/W	Timer 1 Count Buffer Regis

2^{32}

32.4.1.4 TCNTB0 (Timer0 Counter Register)

Register	Offset	R/W	
TCNTB0	0x7F00600C	R/W	Timer 0 Count I

Conclusion: 5-Steps operation of PWM. Can be described as (1) Count By N with Eqn (4), pp3. And deposit N into TCNTB0; (2) Deposit Count M into TCMPB0, where $M = (D.C.) \times N$ ...(5)

(3) The Down Counting will decrement TCNTB0's count by 1 at a time, And a Comparison is made to TCMPB0, if matched, then trigger the Output to "1", Down Counting continues till the Count in TCNTB0 = 0. One period is reached. Then Repeat this process.

Feb.21. 21.

Example: Suppose CLK = 50 MHz. Design Implementation Technique to produce $f_{PWM} = 1000$ Hz, to Drive Stepper Motor Controller, in addition, Duty Cycle is 30%.

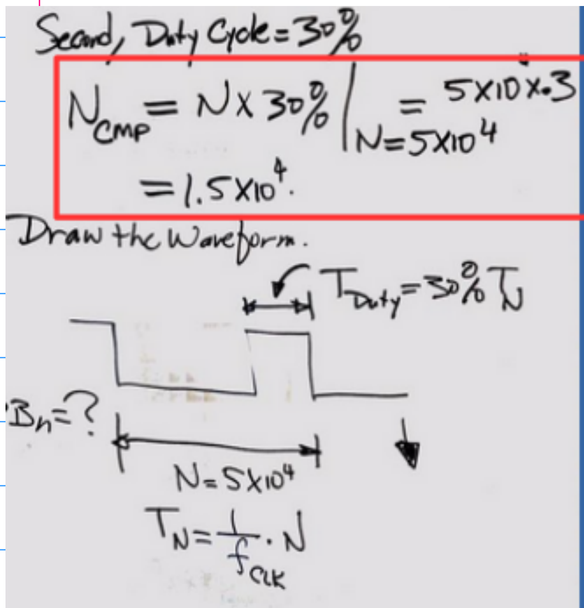
Find: (1) $TCNTB_n = ?$; (2) Find TCMP $TCMPB_n$

Sol First, find the Counts N Based on the given condition.

$$\frac{50\times10^6}{N} = f_{PWM}, \quad \ldots(1) \quad f_{PWM} = 1000$$

$$N = \frac{50\times10^6}{1\times10^3} = 5\times10^4 \to \text{Hex}$$

TCNTn



March21 (monday)

Midterm ON 23rd (Wed).

1 hr. Exam, + 15 min.

Review on Midterm.

3 Questions.

1. A question on Basic Concepts.

1. CPU Architecture
   32 Bit Architecture

a. Memory map. Banks,

b. GPP/IO Peripheral controller.

c. SPRs. Naming, functions.
   GPx CON, GPxDAT
   Tech. Spec ⇄ Binary Pattern

d. ARM11 Reference, Code
   User Space, Kernel Space

e. Target Platform, NAND,

Software Side { O.S. distribution. Toolchain, menu config.

f. SPRs in Driver Code.

GPIO user program.

```
fd = open("        ");
            Path/Driver
ioctl(             );
```

Kernel Space Program Sample

GPECON, etc → CPU Datasheet

g. GPIO Testing { I/P Testing, O/P testing.

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

PP.9 CKT → Pin Selection

gpio79 (pin12)
gpio78 (pin40)

2. Question(s) ON PID Controller Design.

{ Hardware, Software } → Motor Drive GPP/PWM

Motor Drive Pin Connection Requirements

SCH., Connectivity Table.

"Demo" Live Execution of your program.

i. Board is Ready. Take a photo of your Board During the exam.

ii. Stepper motor / motor Drive And the Prototype Board work together, Take a photo, Screen Capture of program execution.

Motor Operation, micro steps, Angular Displacement.

Target Platform. Hardware Configuration to access device driver, such as GPIO, PWM or I2C.

3. Theoretical Aspects of the PID Controller Design.

a. PID Block Diagram.

P~, PI~, PD, ... etc.

D(erivative Controller) Short future.

I(ntegration Controller) History

b. Computation. Forward Difference, Backward Difference, Central Difference. Kernels, Computation.

Integration Controller. History Back N Steps.

c. Sensor Interface Hardware I2C I/F.

LSM303 I2C Based Sensor.

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

Example: An Application of LSM303

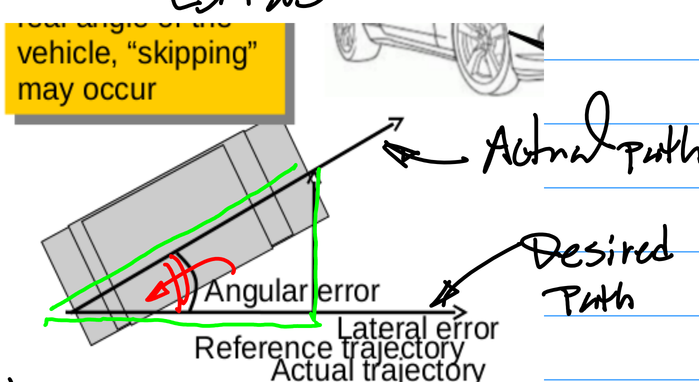


Fig 1.

α: from LSM303

$\vec{P_A}$: is measurable

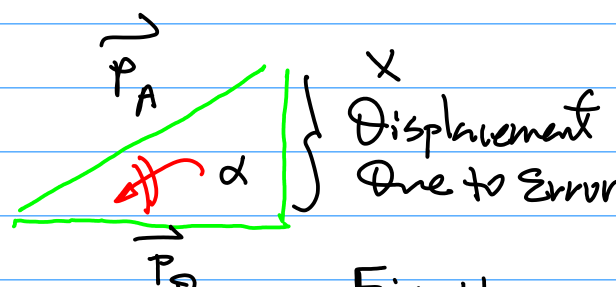


Fig. 1b

CmpE242 March 21, 22

Find $\|\vec{P_A}\|$ is defined by PWM Driven Steppermotor action.

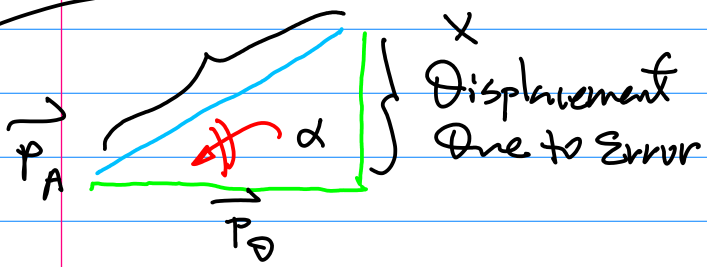


Fig. 2

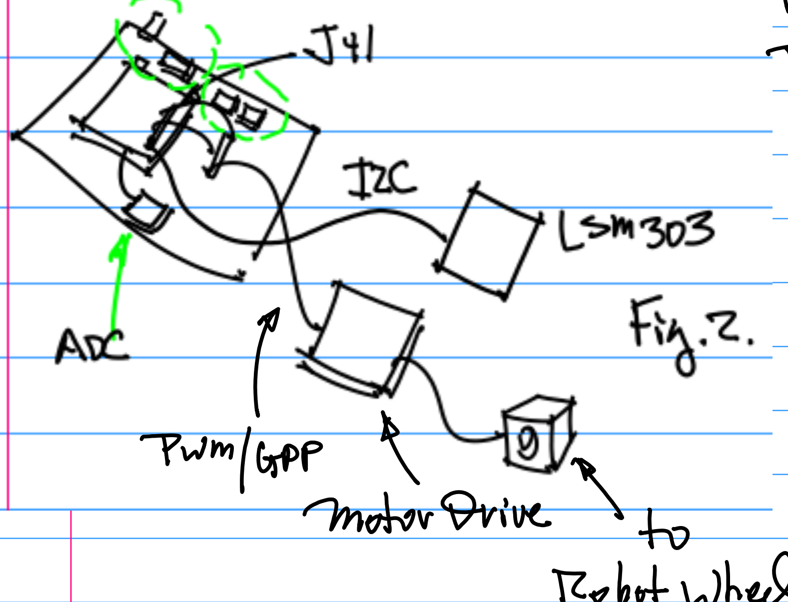
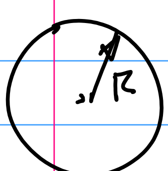


Fig. 2.



the wheel of the Robot has the dimension R = 100mm

Now, Let's take a look at the hard ware of the motor Combo.

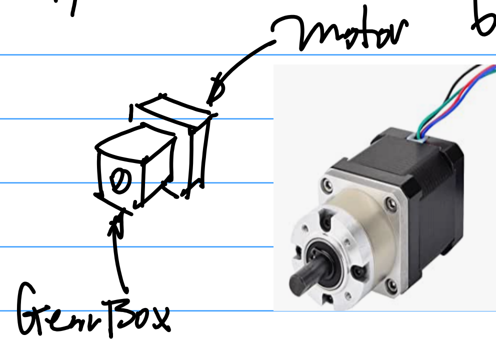


Fig 3.

STEPPERONLINE Nema 17 Geared Stepper Motor Gear Ratio 5:1 3D Printer Extruder Motor DIY CNC Robotics

Visit the STEPPERONLINE Store

25 ratings

Amazon's Choice in 3D Printer Motors by STEPPERONLINE

$40.05

Reduction Ratio, R_R

For the purpose of increasing the Torque.

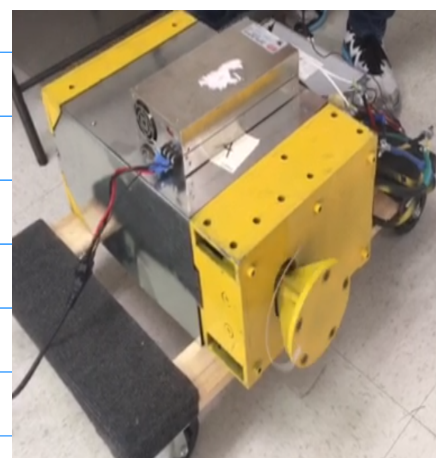


Fig. 4a, 4b With Reduction Ratio Gear Box.

CmpE242 March 21, 22

To find / Establish one-to-one mapping Between the actual path and PWM operation.

April 4 (monday)

Topics: 1° I2C Interface for LSM303 Sensor Integration.

Ref:

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

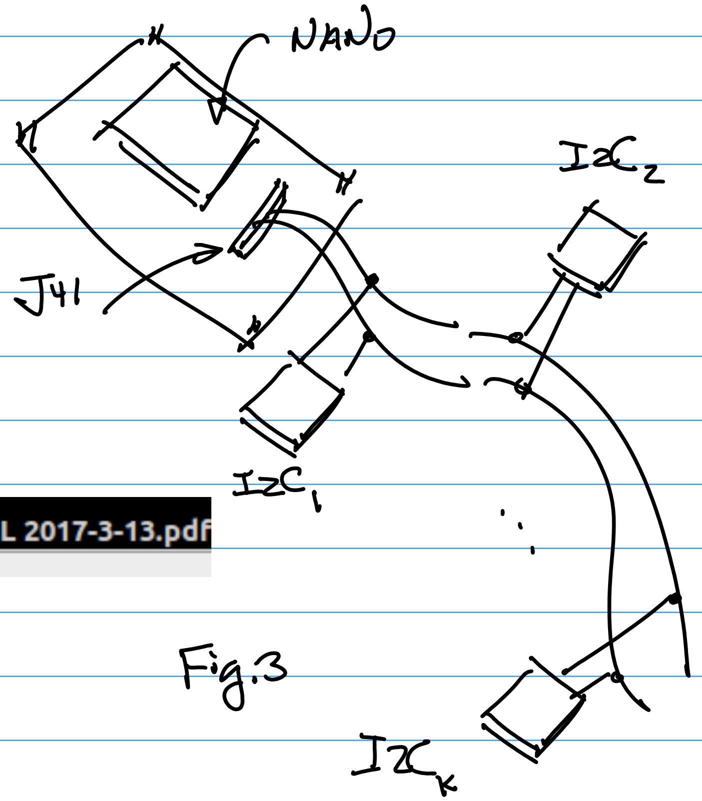


Fig. 3

I2C Hardware Features
- Protocol Definition
- Coding Implementation

Hardware Features:

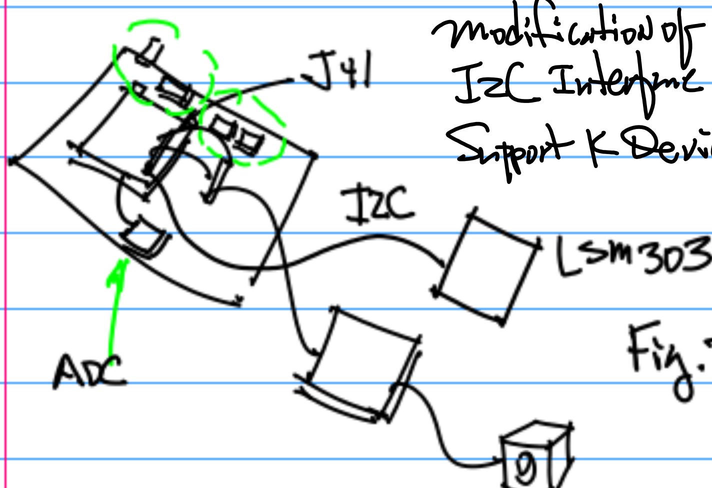
1. Pins
   - SDA: Bi-Directional Data Pin. (Signals in Both Direction)
   - SCK: Serial Clock.

   Speed: ~4 mbps, ≤ 10 mbps

Sub-Address: 7 bits, $2^7 = 128$ → K = 128 Devices, But in Real Engineering Design, "FAN-IN", "FAN-OUT" (e.g. Adequate Electric Current) have to be taken into Consideration.

Bus Communication
- Master Device
- Slave Device(s)

2. I2C Bus



modification of I2C Interface to Support K Devices

Fig. 2.

Ref:

2022S-108-LSM303DLHC.PDF

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.

Note

t_0 . t_1 t_2 t_3

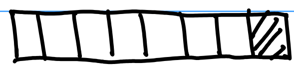
Table 11. Transfer when master is writing one byte to slave, pp 20, datasheet

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

0X3D Table16

0X3D Table17

Fig4.



7 bits for the Address "Slave"

Directional Bit:
- WRITE: 1
- READ: 0

Note: Space-Time Diagram.

→ Space

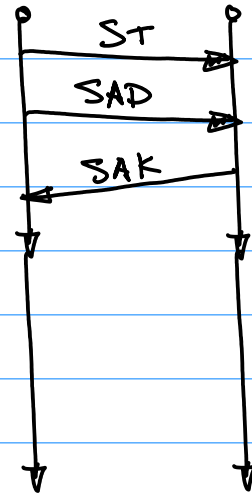
Spatial

Target (NAND) "Slave"

I2C Master LSM303

→ time

a. On SDA Line
b. from the master
c. 7 bits for SAD (Salve Addr)



Time

Fig.5

Note: 1° ST Start
2° SAD Slave Address
3° SAK Slave Ack.

And at t_3: SUB Sub-Slave-Addr. to Address the unit inside the Slave Device.

Question: SAD for LSM303

a.

b. SAD[7:1]

PP.20

Table 14. SAD+Read/Write patterns

Command	SAD[7:1]	R/W	SAD+R/W
Read	0011001	1	00110011 (33h)
Write	0011001	0	00110010 (32h)

0011001X

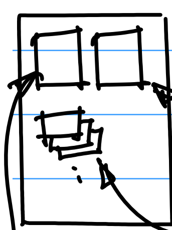
3

(1) 0011 Read : 3
(2) 0010 Write : 2

0X33 Read
0X32 Write

LSM303

Fig 6



"I2C "Slave"

meg. Sensor

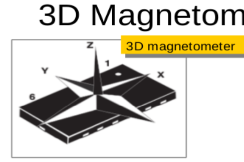
Special Purp. Registers

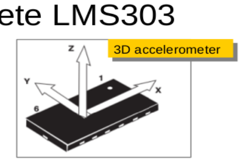
MCU: microprocessor
FSM: Finit State machine

SPRs: Config & Control, Data Sensing

Example: Find Sub-Addr.

3D Accelerometer and
3D Magnetomete LMS303





Note: Read Op is very often the 1st one from the master to get Manufacturer's ID & Device ID etc.

Ref: Slide 7. a. Sub-Addr. for magnetic Sensor

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

Control/Config Register's Address: 0X00 "SUB"

For magnetic Sensor.

Table 16. SAD

Command	SAD[6:0]	R/W	SAD+R/W
Read	0011110	1	00111101 (3Dh)
Write	0011110	0	00111100 (3Ch)

Table 17. Register address map (continued)

Name	Slave address	Type	Register address		Default
			Hex	Binary	
TIME_LATENCY_A	Table 14	rw	3C	011 1100	00000000
TIME_WINDOW_A	Table 14	rw	3D	011 1101	00000000
Reserved (do not modify)	Table 14		3E-3F	--	--
CRA_REG_M	Table 16	rw	00	00000000	0001000
CRB_REG_M	Table 16	rw	01	00000001	0010000
MR_REG_M	Table 16	rw	02	00000010	00000011
OUT_X_H_M	Table 16	r	03	00000011	output

Example: Tech Spec.
1° Read Angular Information / x-, y-, z-Acceleration
2° Sample Rate (of Read): 30Hz
Find 1° Special Purpose Register Responsible to perform Configuration.
2° Find Binary Pattern to initialize the SPR.

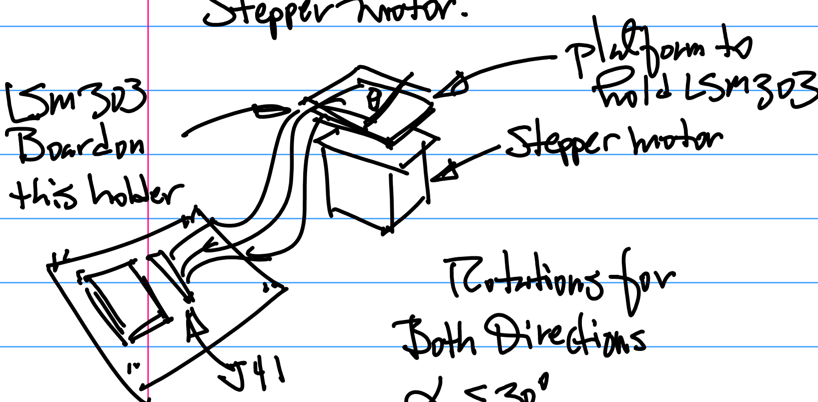
April 7. Wed

Topics: 1° LSM303 Interface Design.
Project Due April 18 befor Class.

Project Requirements:

1. Written Requirements + Rubrics Are to Be posted online, on CANVAS.

2. Integrate LSM303 Sensor to Stepper motor.



Rotations for Both Directions
$\alpha \leq 30°$

3. To Run Stepper motor program to Actuate motor Rotation, then Read LSM303 Data to find for each Actual the displacement Value.

4. Test 3 Configuration of the motor, e.g.
   a. Full Step. 1.8 Degree Angle per step.
   b. Half Step. 0.9 Degree
   c. 1/4 Step. 0.45 Degree

5. Form A table to List the Data Read from LSM303

PWM pulses	LSM303 Input

Note: Try to Run multiple pulses of the PWM, for Example, 5 ~ 10 pulses, when collecting Data.

6. Create Readme file, one page to Describe your implementation.
   a. Schematics
   b. Screen Capture of your code execution
   c. Generate 10 Sec. ~ 15 Sec. Video Clips to Show the working system.
   d. 2 photos. { One for the entire System. (Laptop, Target Platform, Stepper Motor, Sensor)
   One for the Sensor (LSM303) Interface Design

7. Includes All your Source code for testing/verification.

8. Put all of the Above into A Zip file, Submit the file to CANVAS

Note: please Bring your Implementation Board to the Class for Demo. Next Monday.

Example: Process/Steps to Interface to LSM303.

Step 1. Addr. of the Sensor. See Table 16 For mag. Sensor.

Note: Table 17, Special Purpose Registers { For Config/Control
For Data

CMPE242 April 7, 22

7.2 Magnetic field sensing register

7.2.1 CRA_REG_M (00h)

8 bit Register

a.

			0	0	0	0	0

Table 73

CRA_REG[7:5]

Table 73. CRA_REG register

GN2	GN1	GN0	0(1)	0(1)	0(1)	0(1)	0(1)

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

b. For Mag. Sensor

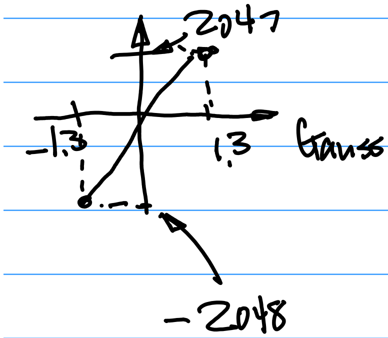
Smaller Range / more Sensitive

Table 75. Gain setting

GN2	GN1	GN0	Sensor input field range [Gauss]	Gain X, Y, and Z [LSB/Gauss]	Gain Z [LSB/Gauss]	Output range
0	0	1	±1.3	1100	980	0xF800–0x07FF (-2048–2047)
0	1	0	±1.9	855	760	
0	1	1	±2.5	670	600	
1	0	0	±4.0	450	400	
1	0	1	±4.7	400	355	
1	1	0	±5.6	330	295	
1	1	1	±8.1	230	205	

c. Out Range

$2^{11} = 1024 \times 2$
$= 2048$



Larger Range / Less Sensitive

GN2=0, GN1=0, GNϕ =1

Note: Start with either default Setting of the Gain, or Set the gain to GN2 GN1 GN0 = 100

Then, Control Register for Operation mode Selection

Only 2 Bits

4. identify the control register responsible for mode of operation, from datasheet, pp 37, table 76, 78 MR_REG_M (02h) ← Sub-Addr. for this Register

0(1)	0(1)	0(1)	0(1)	0(1)	0(1)	MD1	MD0

use Continuous mode

MD1	MD0	Mode
0	0	Continuous-conversion mode
0	1	Single-conversion mode

To Config/Access to Special Purpose Register(s), follow this "Write" Pattern:

SPR's Address: 0x01 for GAIN Setting, CRB_REG_M
0x02 for mode MR_REG_M,

SUB		DATA	
	SAK		

Binary Pattern

For Reading After the Configuration, we will need find SADHR (Sub Addr. for Read), Example is given in PPT, from Datasheet, Table 15, PP:20.

Coding Implementation: C/C++ Option
Python Option.

This Capability is to be implemented in this Class as a ~~project~~ Part of the

April 11 (Monday). ADC
Analog To Digital Conversion.
Industrial IOT Application.
Analog Sensor Interface

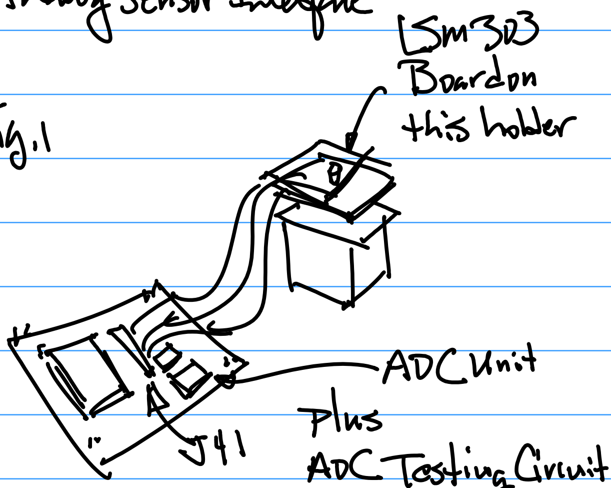


Fig.1

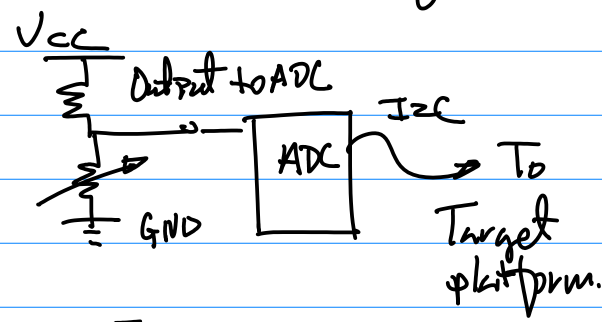
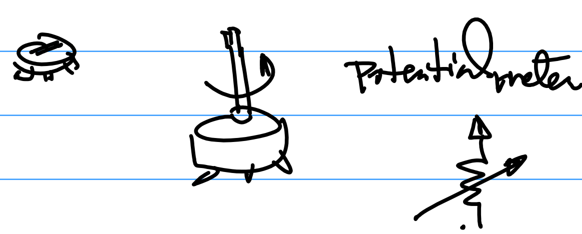


Fig.2.

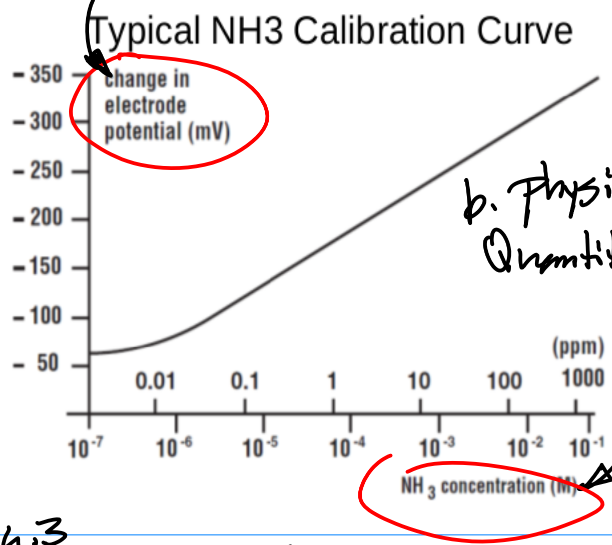
Example: Background of Analog to Digital Conversion.
Sensor (Analog) Input,
Analog Current



2018S-20-NH3- 4Design2018-1-16.pdf

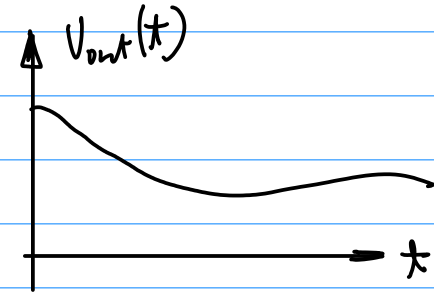
CmpE242 April 11, Monday, 22

Note: a. Analog Change Output



b. Physical Quantity

Fig. 3



1° Sampling ADC.

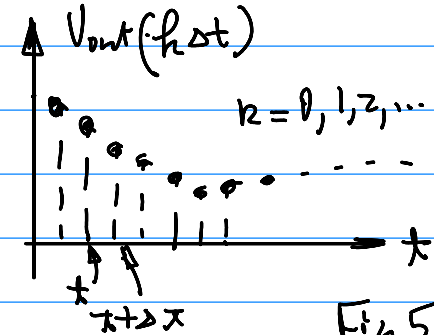
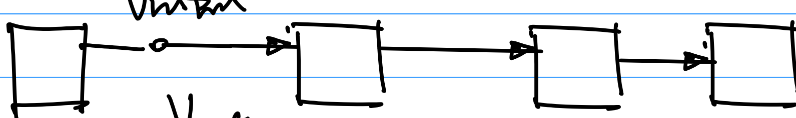


Fig. 5



2° Quantization

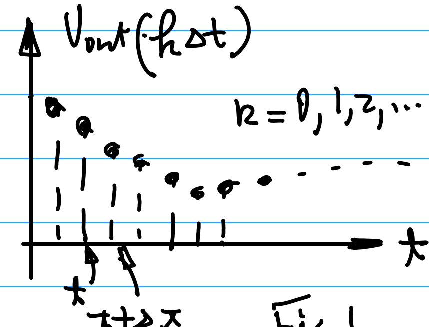
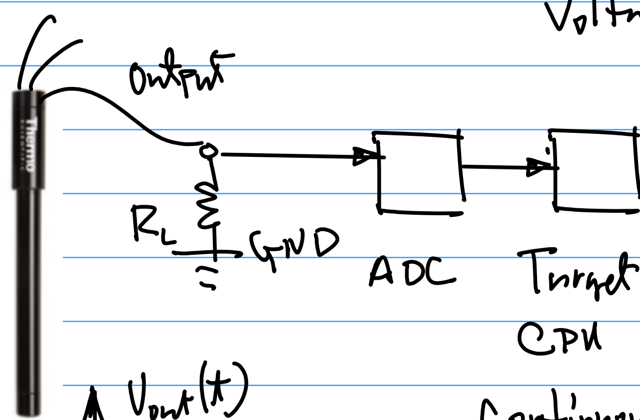


Fig. 6

Conversion: Current i to Voltage v.

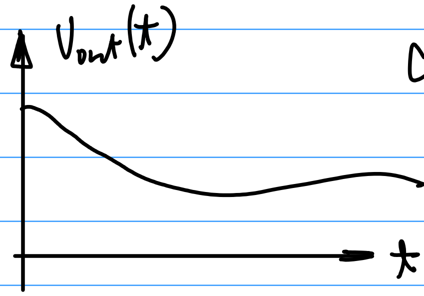


Note: the magnitude after the quantization becomes one of 2^x Possible Levels.

(10 bits ADC, $2^{10} = 1024$ Levels,

12 bit ADC, $2^{10} \cdot 2^2 = 4096$ Levels)

Fig 4.



Continuous from the Analog Sensor.

Performance of ADC:

CmpE242 April 11, Monday. 22

1. Sampling Rate:
(Common) 500KSPS
(Samples per Second)

Sampling Theorem (Nyquest ~)

$$f_{Sampling} \geq 2 f_{max} \quad \cdots (1)$$

The Sampling Speed has to be greater than or equal to Twice of the maximum Speed of a given Signal.

2. Measurement of ADC. No. of Quantization Level. 10 bits ($2^{10}=1024$), 8 bits ($2^8=256$) 12 bits ($2^{12}=4096$).

3. Linear Characteristic.

ADC output (10 bits)

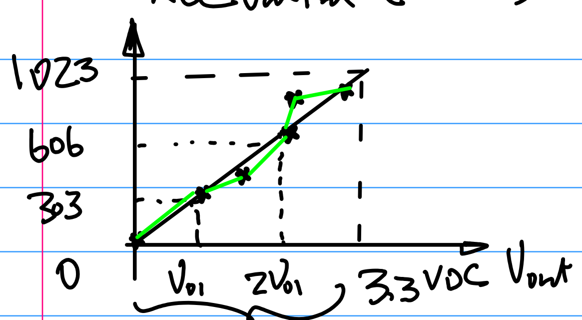


Fig. 7

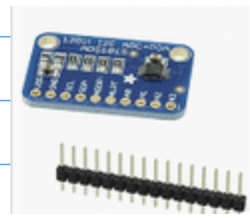
Fig. 7 ADC Input Range (Dynamic ~)

$$\frac{\Delta Output}{\Delta V_{out}} = k \text{ (Slop of the Line in Fig. 7).}$$

We have to measure "Linearity" of a given ADC.

Homework (After the project of LSM303)
Formal Due Date to Be Announced in the next lecture.

1° Identify & Purchase Your ADC module.



ADC with I2C Interface, ADS1115.

2° Generate Schematics.

Start with Symbols

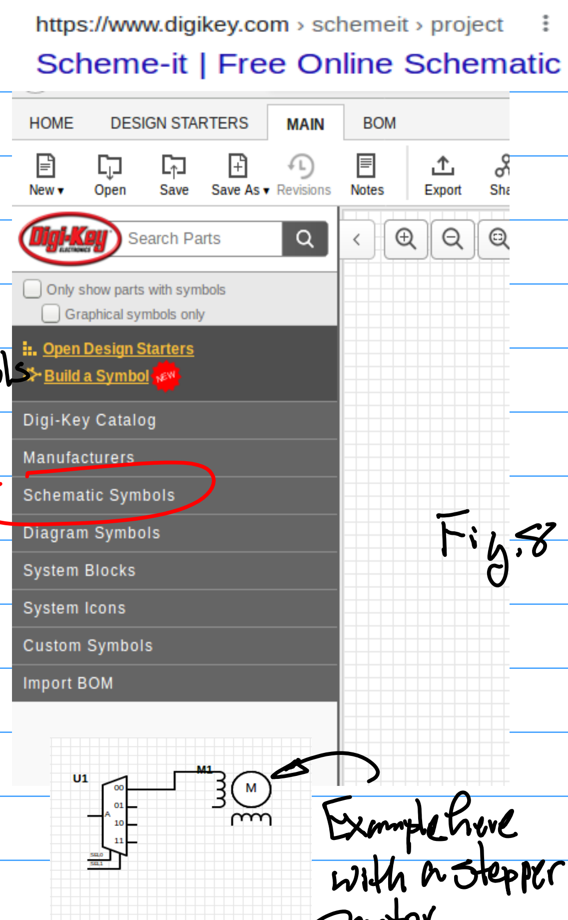


Fig. 8

Example here with a stepper motor

3. Use your Built Circuit Below, to collect 10 pairs of Data Points.

$(V_{01}, D_1), (V_{02}, D_2), \ldots (V_{010}, D_{10}),$

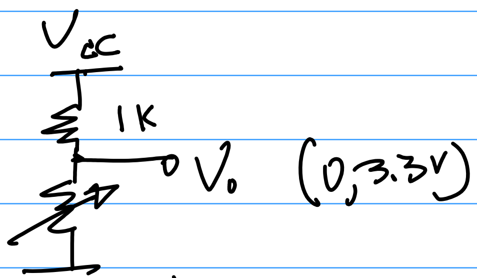


Fig. 9 4.7KΩ Potential meter / 470KΩ meter

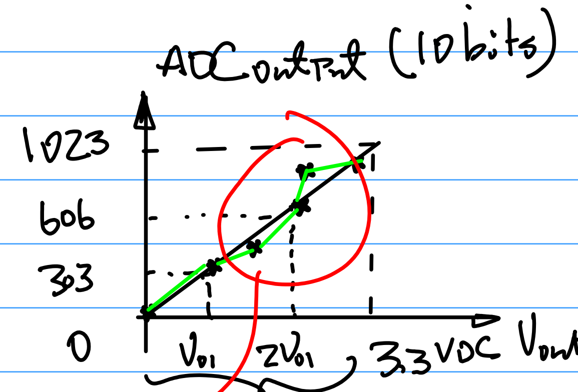
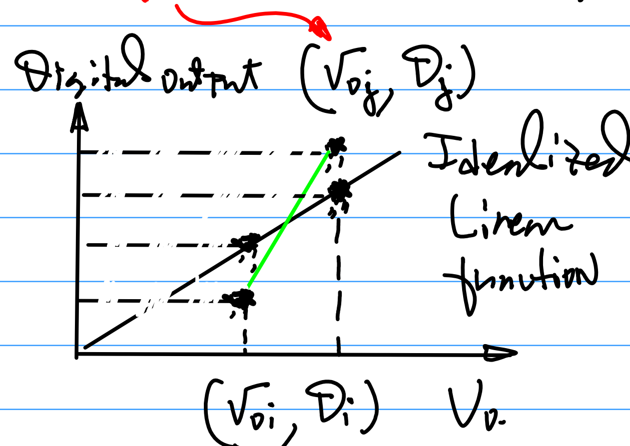


Fig. 11 ADC Input Range (Dynamic ~)

4. In order to do 3, you will have to have Python or C/C++ interface to your ADC.



5. plot your data, as in the Fig. here

Actual measurement

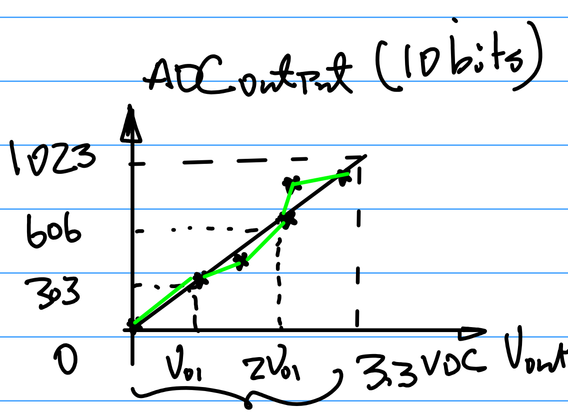


Fig. 10 ADC Input Range (Dynamic ~)

Develop A Correction function to correct Non-Linear Behavior By Developing A function.

6. Correction of Non-Linear Behavior

April 13.

Topics: 1° ADC Linear Compensation

Example: To Correct/Compensate the NonLinear Deviation (Green Line) to make it as Linear function (Black Line).

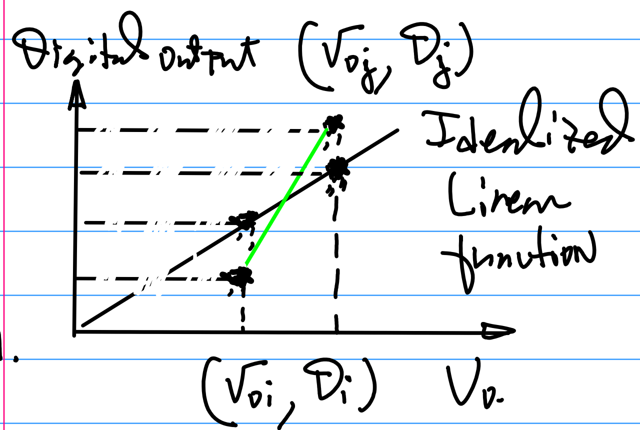


Fig 1.

Define $g(x,y)$ as the "green" Line.

$f(x,y)$ as the desired Linear Characteristic Line

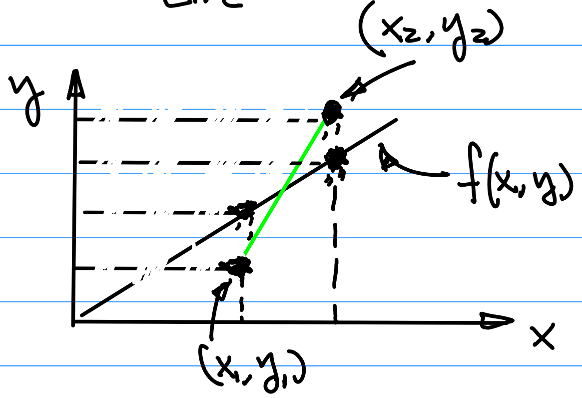


Fig 1-b

Objective: To Develop Post Processing Technique to Correct/Compensate the green Line (Actual Data).

Step 1. Desired Linear Characteristic Line.

Take A target form with An ADC

ADC has: 10 bits ADC

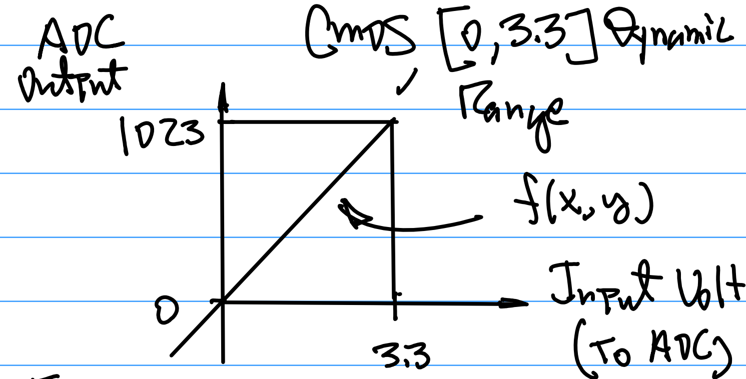


Fig. 2

$10 \text{ bits} = 2^{10} \text{ bits} \Rightarrow 1024 \text{ Levels}$

Since $y = ax + b$ ... (1)

where $b = 0$ B/C the Line passes the origin.

a is a slop.

$a = \dfrac{\Delta y}{\Delta x}$ | $\Delta y = 1024$, $\Delta x = 3.3$

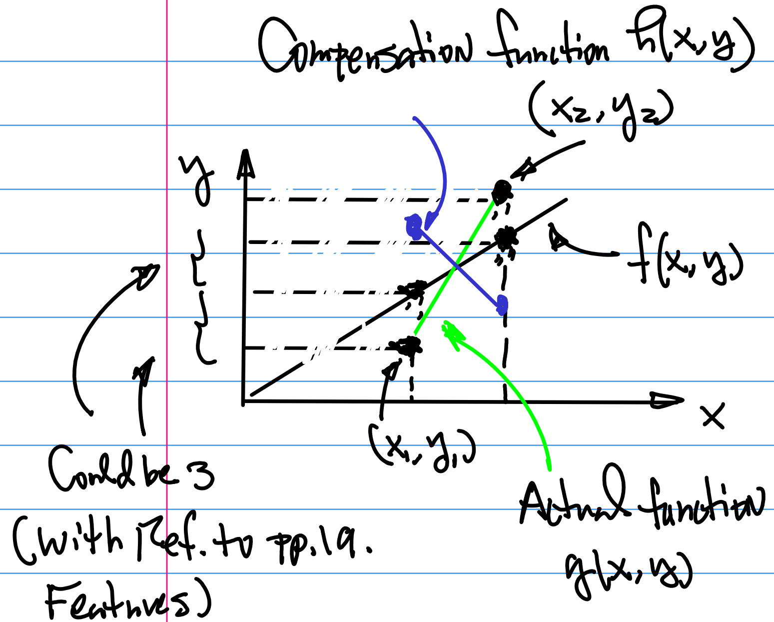
$= 1024/3.3 = 310.3$

Step 2.

$g(x,y) + h_{Comp}(x,y) = f(x,y)$ ... (2)

we have

$h_{Comp}(x,y) = f(x,y) - g(x,y)$ ... (2-b)



Step 3. g(x,y) from the Actual Data, we have from the Calibration Process.

From Fig. 1-b.

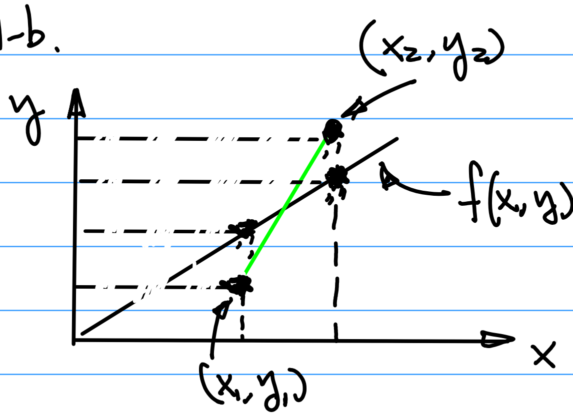


Fig. 1-b

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

$$y = ax+b.$$

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

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

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

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

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

Hence

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

where

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

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

From Step 2, Eqn (2-b), find the Compensation.

$$h(x,y) = f(x,y) - g(x,y)$$

where $f(x,y)$: $y = ax$

$$f(x,y) = y - ax \quad \ldots (5)$$

$$g(x,y): \; y = \cancel{a}^{\,b'} x + \cancel{b}^{\,c'}$$

$$g(x,y) = y - (a'x + b') \quad \ldots (6)$$

Therefore

$$h(x,y) = y - ax - [y - (\cancel{a'}^{\,b'}x + \cancel{b'}^{\,c'})]$$
$$= -ax + \cancel{a'}^{\,b'}x + \cancel{b'}^{\,c'}$$
$$= (\cancel{a'}^{\,b'} - a)x + \cancel{b'}^{\,c'}$$
$$= (b' - a)x + c' \quad \ldots(7)$$

Hardware, Architectural Aspects of ADC

2021F-105-#0-cpu-arm11-2018S-29…

github, in 244 folder

6410X_UM

pp1209

39 ADC & TOUCH SCRE

This chapter describes the functions and usage of ADC & Touch

39.1 OVERVIEW

Note: a. 10/12 bit ADC. for 10 bits [0, 1023], or for 12 bits [0, 4095]

b. CMOS, Dynamic Range [0, 3.3V]

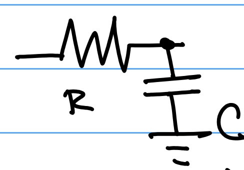
c. 8 ADCs

The 10-bit/12-bit CMOS ADC (Analog to Digital Converter) is a recycling type device with 8-channel analog inputs. It converts the analog input signal into 10-bit/12-bit binary digital codes at a maximum conversion rate of 1MSPS with 5MHz A/D converter clock. A/D converter operates with on-chip sample-and-hold function. The power down mode is supported.

Touch Screen Interface can control input pads (XP, XM, YP, and YM) to obtain X/Y-position on the external touch screen device. Touch Screen Interface contains three main blocks; these are touch screen pads control logic, ADC interface logic and interrupt generation logic.

d. $f_{Sampling,max} = 1 \times 10^6$ Hz, $f_{ADC} = 5 \times 10^6$ Hz

e. Sample-And-Hold



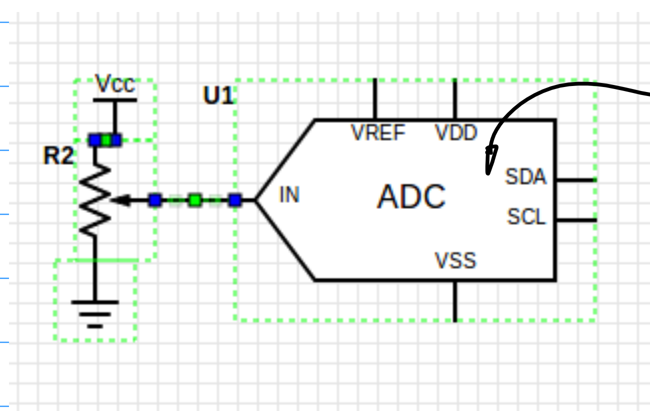
Inside ADC

Schematic Symbols

- Amplifiers
- Circuit Protection
- Clock Management
- Connectors
- **Converters**
  - Analog to Digital

15

- Resolution: 10-bit/12-bit
- Differential Nonlinearity: ± 2.0 LSB
- Integral Nonlinearity: ± 4.0 LSB
- Maximum Conversion Rate: 1MSPS
- Low Power Consumption
- Power Supply Voltage: 3.3V
- Analog Input Range: 0 ~ 3.3V
- On-chip sample-and-hold function
- Normal Conversion Mode
- Separate X/Y position conversion Mode
- Auto(Sequential) X/Y Position Conversion Mode
- Waiting for Interrupt Mode
- STOP mode wakeup source



Suppose for 10bits ADC Configuration

$b_9 b_8 \cdots b_1 b_0$

$6 \begin{cases} +3 \ (=11) \\ -3 \end{cases}$

max error Due to Non-Linear Behavior.

- Resolution: 10-bit/12-bit
- Differential Nonlinearity: ± 2.0 LSB
- Integral Nonlinearity: ± 4.0 LSB
- Maximum Conversion Rate: 1MSPS
- Low Power Consumption
- Power Supply Voltage: 3.3V
- Analog Input Range: 0 ~ 3.3V
- On-chip sample-and-hold function
- Normal Conversion Mode
- Separate X/Y position conversion Mode
- Auto(Sequential) X/Y Position Conversion Mode
- Waiting for Interrupt Mode
- STOP mode wakeup source

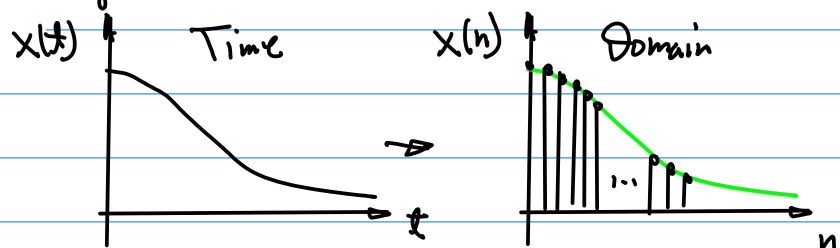
"Sampling"

Nyquest Theorem

$$f_{Sampling} \geq 2 f_{max} \quad \cdots (8)$$

Step 2. Frequency Characterization

Signal



Continuous Signal From a Sensor.

Discrete Signal (After ADC)

Freq. Domain Fourier Transform

Discrete Fourier Transform

$$X(f) \triangleq \int_{-\infty}^{+\infty} x(t) e^{-j2\pi f t} dt \quad \cdots (2)$$

April 18 (Monday).

Topics: 1. Data Validation.

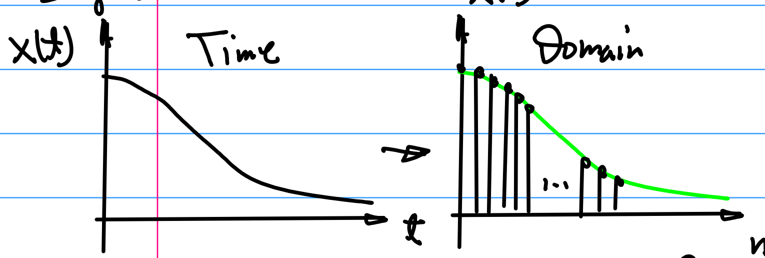
Background:

Sampling Theorem (Nyquest Theorem)

$$f_{Sampling} \geq 2 f_{max} \quad \cdots (1)$$

Sampling Freq. (ADC)

Signal Freq.



Continuous Signal From a Sensor.

Discrete Signal (After ADC)

Fig. 1

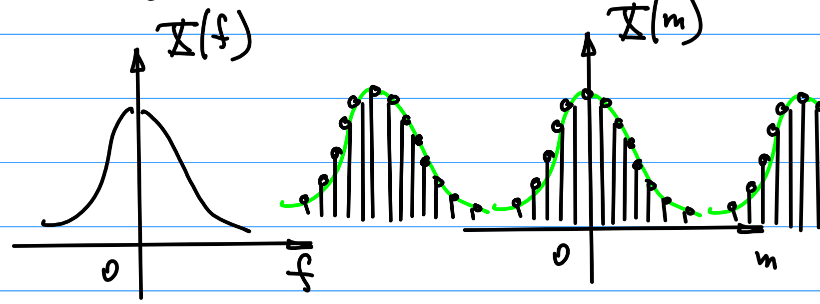


Fig. 2

Note: Discrete Signal's Fourier Transform is defined as D.F.T (Discrete Fourier Transform) it repeats itself (Periodic Function). And it could have "Alising" (Distortion).

Fig.3.

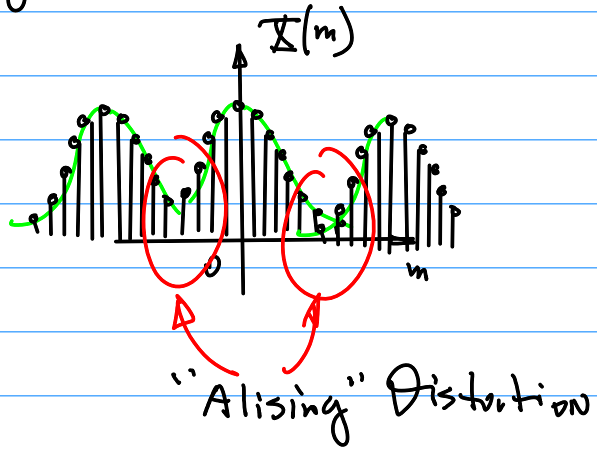
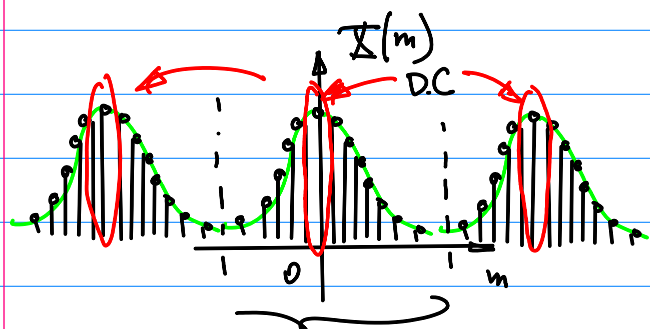


Fig.4. Analyzing the Distortion.
Analyze Sampling Rate



One Period of the Signal x(n) (After ADC/Digitization)

1° $x(n) = x(n+kN)$, $k = 0, 1, 2, \dots$ ...(3)

Time Index

Total Number of Points of One Period.

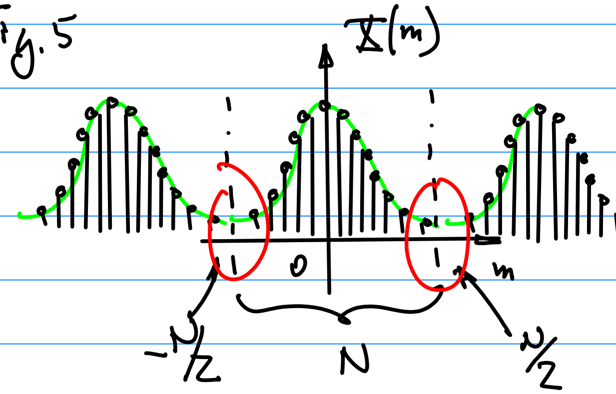
2° DC Component

$X(m)$: Discrete Fourier Transform.

$m=0$, $X(0)$ is D.C. Component

3° Period

Fig.5



$X(m) = X(m+kN)$ ... (4)

You can take Any N points (Consecutive Points)

Highest Frequency index

$\underbrace{\hspace{3em}}_{m}$

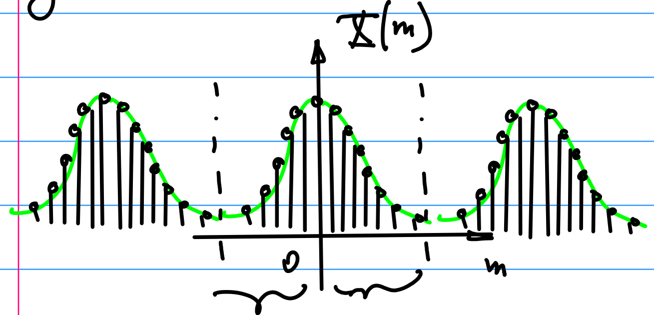
$N = 2^x$ ... (5) Even Number

Condition for Computed D.F.T, e.g. FFT (Fast Fourier Transform)

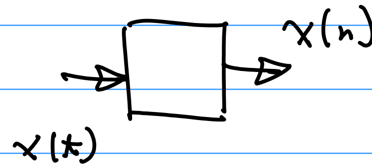
a. One Component (D.C) at $m=0$, $N-1$ Components Left.

b. $N-1$ has to be distributed

Fig. 6



ADC



highest Freq. index $m = \frac{N}{2} - 1$

Highest Frequency of A given Signal.

$f_{Sampling} \geq 2 f_{max}$.

Take $x(n)$ (ADC output)

Perform Discrete Fourier Transform.

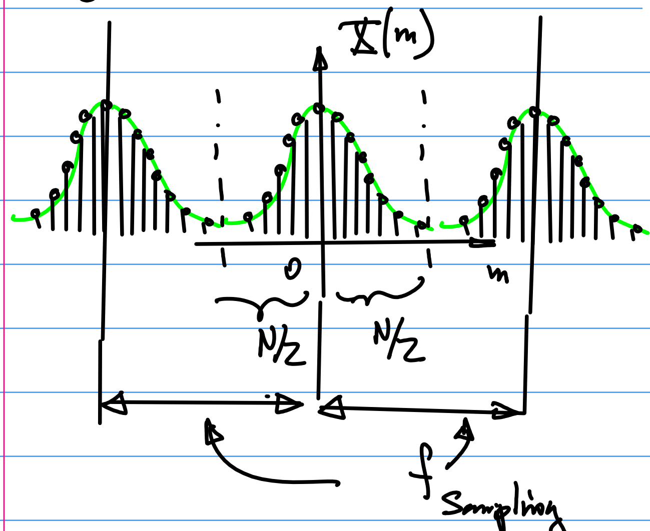
$$X(m) = \frac{1}{N} \sum_{n=0}^{N-1} x(n) e^{-j 2\pi \frac{mn}{N}} \quad \cdots (6)$$

Ref 1D DFT v2.pdf

C. One Period N for $X(m)$

D.F.T. is apart from the other period by $f_{Sampling}$

Fig. 7



$$\begin{bmatrix} X(0) \\ X(1) \\ \vdots \\ X(N-1) \end{bmatrix} = \frac{1}{N} \begin{bmatrix} W_N^0 & W_N^0 & W_N^0 & \cdots & W_N^0 \\ W_N^0 & W_N^1 & W_N^2 & \cdots & W_N^{N-1} \\ W_N^0 & W_N^2 & W_N^4 & \cdots & W_N^{2(N-1)} \\ \vdots & & & & \\ W_N^0 & W_N^{N-1} & W_N^{2(N-1)} & \cdots & W_N^{(N-1)(N-1)} \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ \vdots \\ x(N-1) \end{bmatrix} \quad \cdots (7)$$

Note: 1° Left hand Side

$$\underbrace{X(0), X(1), \cdots, X(N-1)}_{N \text{ Points}}$$

D.C. $X(0)$.

Define Discrete Fourier Transform

The Highest Freq. Component

$X(\frac{N}{2}-1)$. , The Highest Freq. Index $m = N/2 - 1$

2° $W_N^{mn} = e^{-j2\pi \frac{mn}{N}}$

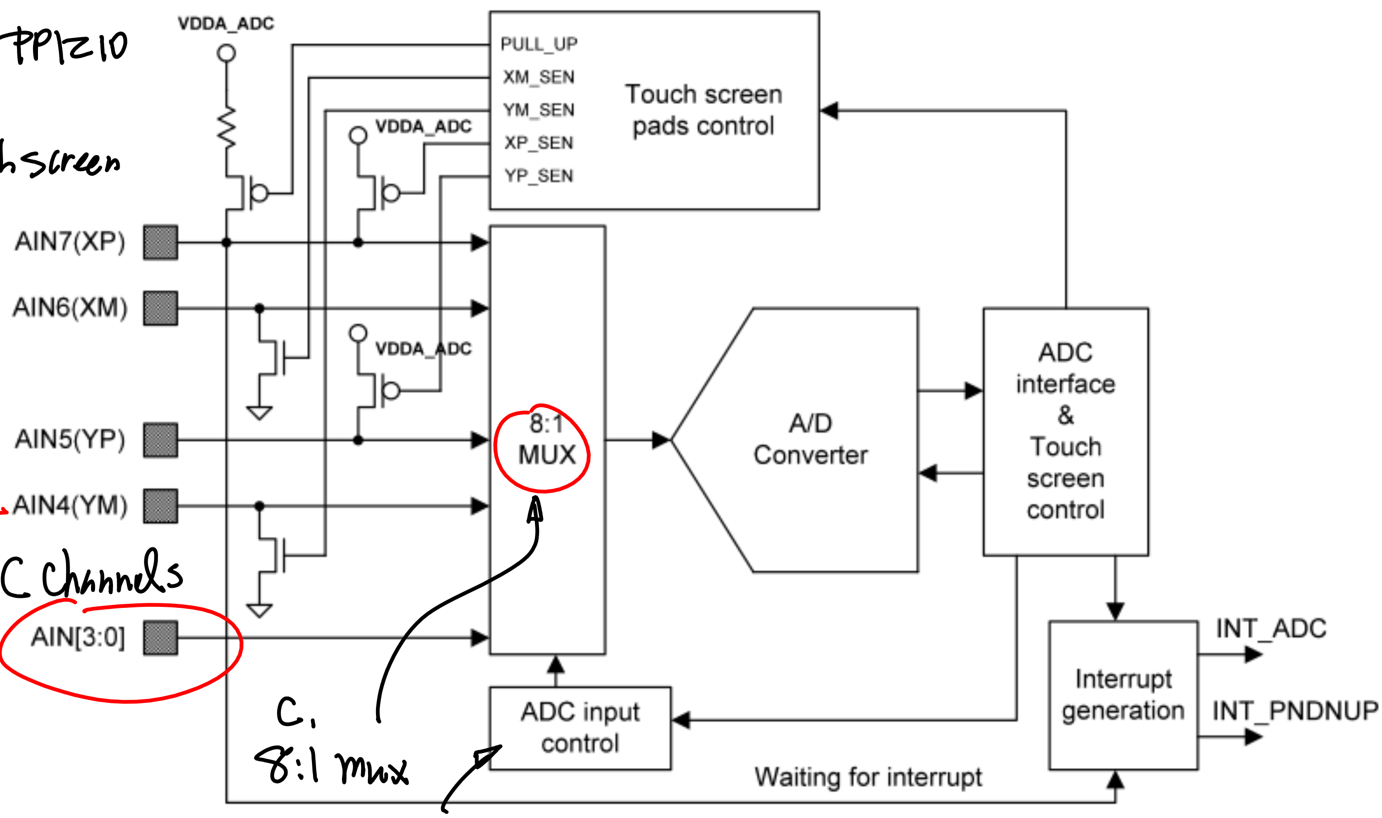
ADC Architecture, ARM-11

Ref:

hualili / CMPE244

2021F-105-#0-cpu-arm11-2018S-29...

PP1210



b. Touchscreen

a. ADC Channels

c. 8:1 mux

d. ADC Input (mux) control

$2^3 = 8$, 3 bits mux