

Jan 27, 21
Welcome to

CMPE242 Harry Li

Embedded Hardware Systems

1. GreenSheet github/huawili/cmpe242

Email: hua.li@sjtu.edu

(650) 400-1116 Text message

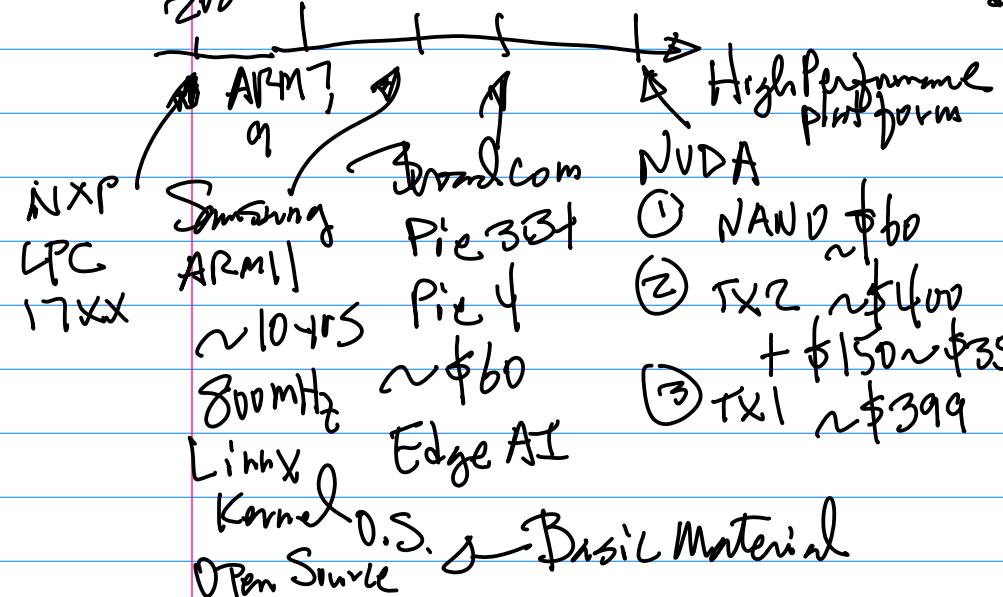
2. Pre-requist Requirements ISO A4

Course Description

Hands-ON.

Target Development Platform

ZED



Scope of the Course

Device Driver

Dep Development

C/C++, Python

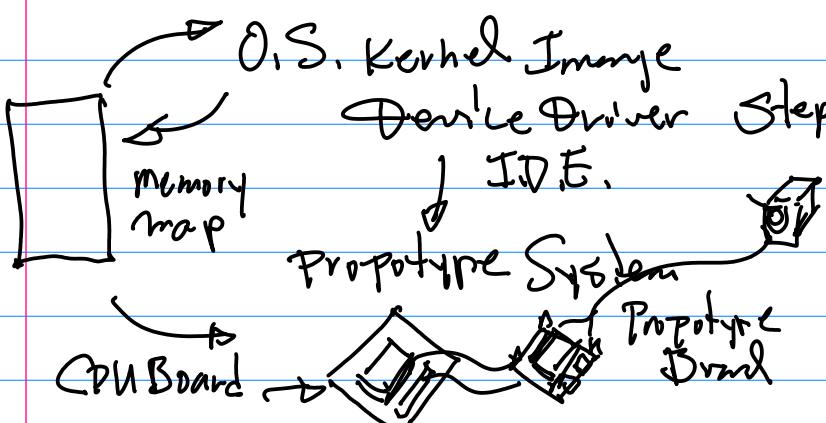
O.S. Kernel

① Assignments (No)

② Collaboration / Submission

of your Class Work

Action 1. github/huawili/cmpe242



2. Datasheet

Samsung ARM11 Datasheet

NXP i.MX6 Datasheet

Architecture PXPiPC

Before: 3 Labs/Projects ✓

Device Driver
Sensor Lsm
303 → FeedBack
Loop

I/F to Target P.I.D
Board Controller

Human Readable
file.

Integration
Optional subjects
(HDL)

1. RISC-V Project - FP (A)

Device Driver & O.S. Kernel
Development Image

2. ROS (Robotics

Operating System)

platform visualization
Tool.

* Grading Policy

3-3-4

Mid Projects Final

Action 3. Target Platform Selection

a) Unix-like OS. b) Edge AI Computing (Scalability) \Rightarrow GPU

Office hours M.W. 4:30-5:30pm
ON ZOOM.

Baseline Software

Linux Distribution
Optimized for Embedded platform
Device Drivers.

Example: Datasheet

LPC1769 ARM Cortex M3

CnPE242 Feb. 2021

Today's Topics: 1° System Architecture Review, CPU Datasheet; 2° Target platform

Samsung ARM-11

NVDA --- Pix

System Architecture

NANO
TX1, TX2
NVDA CPU/GPU platform.
Broadcom, Pix 3/4 G.E. (Graphics Engine)
Samsung ARM-11 (9, 7)

CPU Datasheet, LPC1769

2018S-3-UM10360, FP. 9

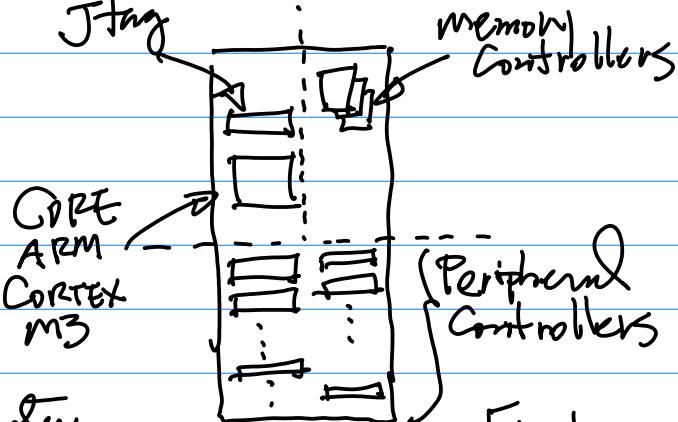


Fig 1.

* Optimization Not Only on the Hardware
But On the Compiler Design, and
System Software Design.

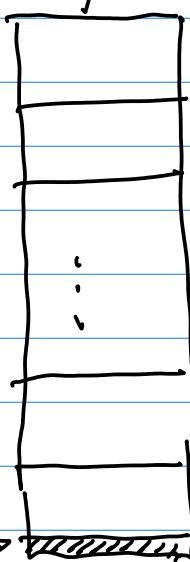
Fig 1. CPU Architecture
ARM Cortex M3MIPS, ARM (most widely adopted)
RISCNote: To Be Able to Draw/
Design CPU Architecture

ARM Architecture — Common Core

Either this OR Your Target
platform (Mba + ARM11)
Base Line

Base Line Hardware (Datasheet); ARM11

Memory map.



$$2^{32} = 4 \text{ GB (B)}$$

0x0000-
0000

Fig. 2

① 32 Bit RISC Architecture

$$2^{32} = 2^0 \cdot 2^0 \cdot 2^0 \cdot 2^2$$

$$= \underbrace{1K \cdot 1K \cdot 1K}_{1M} \cdot 2^2$$

$$= 4 \text{ GB} \quad \text{(Byte)}$$

② Byte Addressable Machine

whose minimum memory cell with an unique address is a single byte

③ Memory Banks, 8 Banks

Size of Each Bank: 4 GB / 8

$$= 2^{32} / 2^3 = 2^{32-3} = 2^{29} = 2^9 \cdot 2^{20}$$

$$= 512 \text{ MB}$$

Starting Address of Each Bank

Question: How many Bits needed to define the Starting Address of Each Bank? 3 bits, $2^3 = 8$

3 bits Needed

$a_{31} a_{30} a_{29} : a_{28} \dots a_1 a_0$

Little
Indian

ARM CPU can be configured at Boot Stage as either "Little Indian" or "Big Indian".

Find Starting Address for Bank:

$$a_{29} = a_{30} = a_{31} = 0$$

a_{28} has to be added, to form a hex

0x0000-0000

2nd Banks Starting Address

$$a_{31} a_{30} a_{29} : a_{28}$$

$$\underbrace{0 \ 0 \ 1}_{\text{001}} : 0$$

0x2000-0000

3rd Banks Starting Address

$$a_{31} a_{30} a_{29} : a_{28}$$

$$\underbrace{0 \ 1 \ 0}_{\text{010}} : 0$$

0x4000-0000

Now, Consider target Board
Conditions to qualify the Selection

Plus: Many examples
on Device Drivers (I2C,
PWM, SPI, UART, ...)

- (1) ARM Based ; (2) Linux-Like O.S. (Bootloader Tool 4/5)
- (3) Establish Linux Kernel eco-System
- Developer Base (~ millions) { Some Distro.
Tool Chain }
- (4) Technology Innovators / Leaders.
- (5) External Expansion Connectors

Feb 3rd (Wed) CMPE242

Note: 1° Submission of Honest Pledge Form (Signed), CANVAS,
By Sat 11:59 pm; EE242 submission
to e-mail;

Today's Topics: 1° CPU Architecture
2° Target Board Selection - Bill of Material

Ref: github

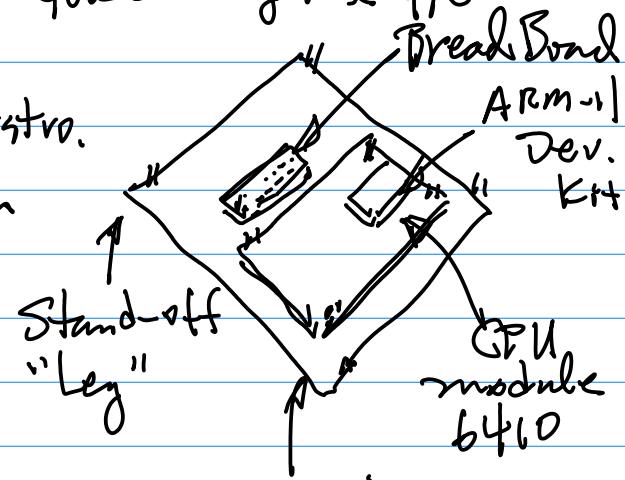
1-ZeroS-left1-hardware board...

ARM-1 | ① Coupled w/ Linux Open Source
Distro { Kernel Sources
② ARM tool Chain
Datasheet Baseline Reference
Requirements, Exams }

DrawBack: Lack of the Ability to

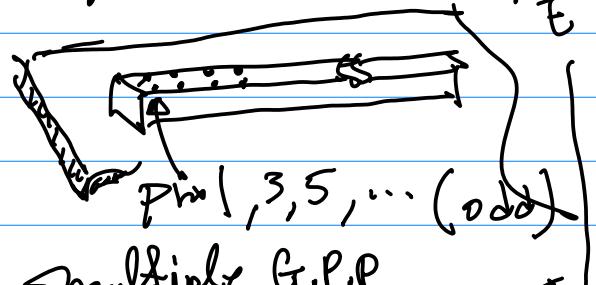
Handle Edge AI ; (G.E.) Tiny b410

Kit from FriendlyARM.com, ~\$590



Note: Bread Limitation -
Run High Speed
Cannot ~20 MHz

CNN.5 GPEx → General
Connector I Purpose (out)
Pin Number physical location



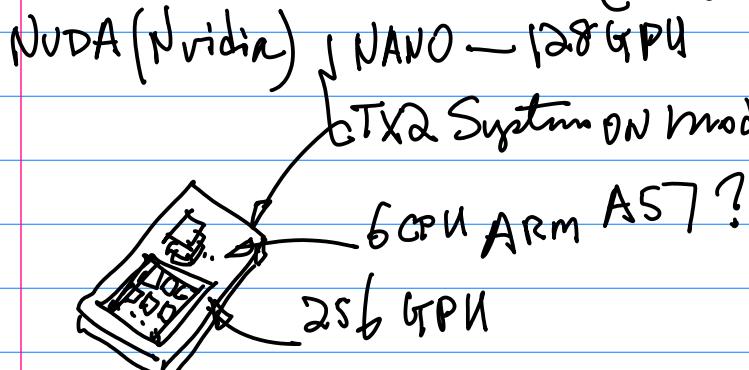
GPEx : GPE3

pin of the Port

Option (Pie Board STB+, 4)

\$75 Pie-4, 8GB mem.

ComPEasy 2



Note: 1° NANO Expansion Connector, Yes
(Limited I/O Function)
Compare to ARM11 → SPI ✓, I2C ?, PWM

2° Kernel Source Distribution, Yes
To Become a developer, to Sign up
→ \$59

Option (NVDA TX2 System-on-Module)

1° Expensive! \$299 + Carrier Board
Edge on AI \$150 - Best

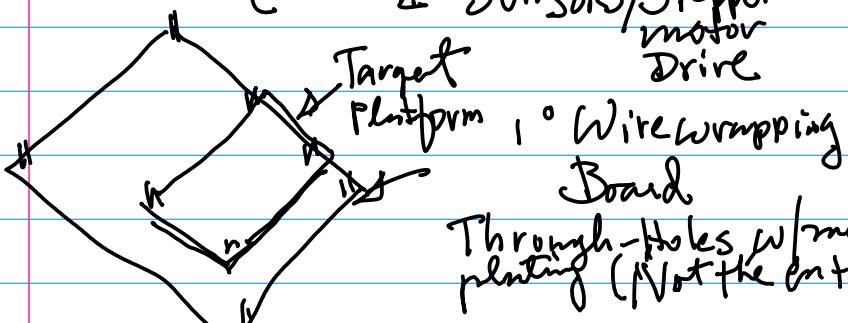
If this is selected, then Z → 4 person
to Share the Cost

Note: 1° 4-Person Team By Next Week;

2° Homework/Project has to
Individual, Each person has
your own Board ;

Bill of materials:

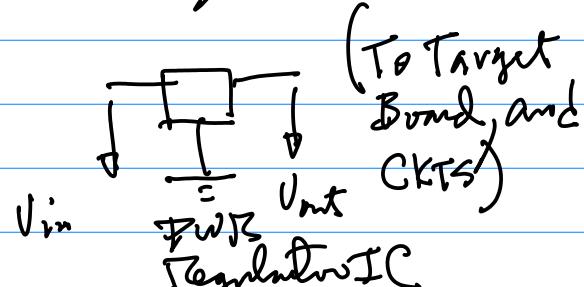
Phase I & II { Phase I - HW1 - "Hello,..."
" II Sensors/Stepper motor



2° 4 Stand-offs ("Legs")

3° Components to Build

PWR CKT & CLK for
I/O Testing ("Hello, the
world").



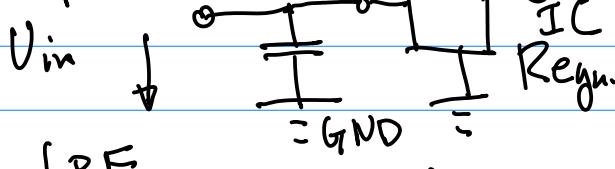
78xx 7805 7812
5VDC 12VDC

Red LED, 4 ~ 10 mA

Resistors. $V_{CC} = 5.0$

$$R = \frac{V}{I} = \frac{5}{4 \times 10^{-3}} = 1.25 \text{ k}\Omega$$

Cap for your External Power Input Clk.



$$C = 4.7 \text{ nF} \quad T = RC \Rightarrow$$

3 dB

4° Toggle s/w

Feb 8, (Monday) CMPE242
Harry Li

Today's Topics: 1^o Bill of Material

2^o Prepare for the 1st Assignment

"Hello, the world". 3^o CPU Architecture

Bill of Material

- Phase I: Target platform
"Hello, the world"
- Phase II: Sensors / Drive / Stepper motor / OP Amps

1. Target:
 - (1) Pic3B+, Pic4
 - (2) NANO, TX2 (Edge AI)
 - (3) ARM11 — Baseline Reference platform

2. Prototype Board, 6" x 4"

POWER CKT : PWR Regulator IC

Wallmount Adapter (M7805) 2VDC dropoff Note: you may need higher voltage source to Stepper motor Drive.
3.
V_{in} C ——————|————— V_{out}
—————|————— GND To CPU
Target

Fig. 5VDC

Connections { External PWR
Internal PWR

4. "Glow" Logic { Asorted Resistors
200 ~ 5KΩ 10KΩ
Caps 7805 PWR Input Node

Switch (Toggle Switch)

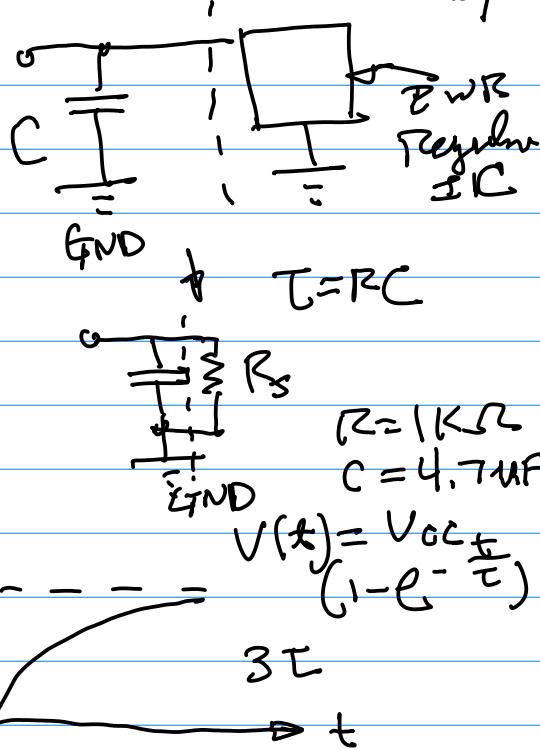


To Control PWR Source;

(1) PWR Distribution Node

LPF (Low Pass Filter)

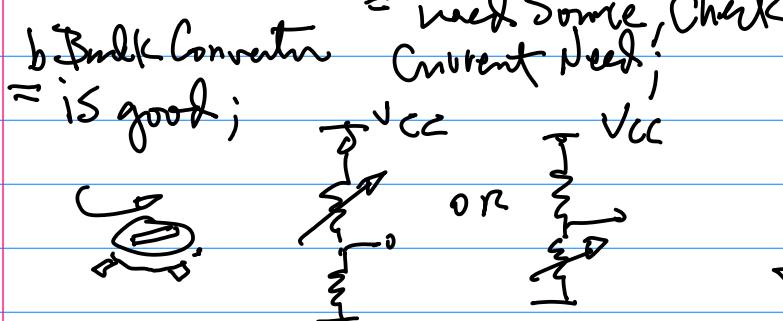
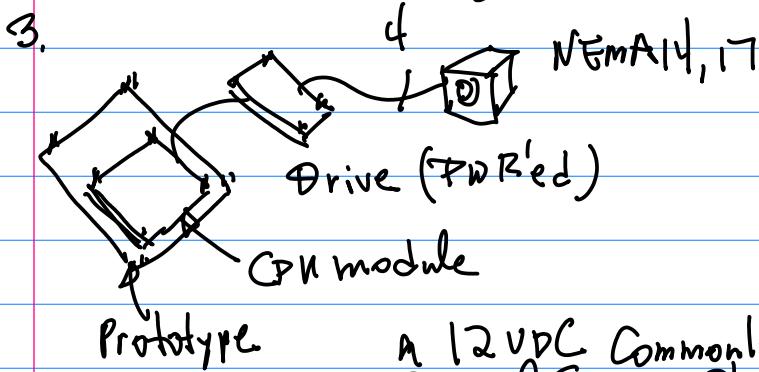
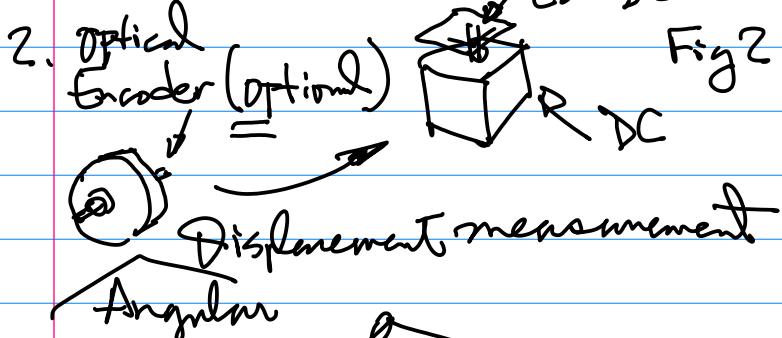
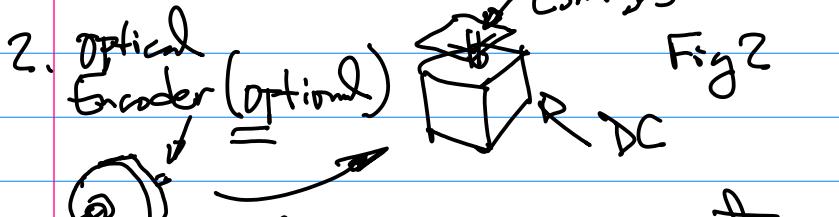
To Build I/O Testing



Phase II (Bill of Material):

Ref: github:

1. LSm303 (I2C) → Autonomous Robot



Consider Building "Hello, the world"
Test CKT

Objectives
1. Bring up the target Board
and Print "your Name,
Last 4 Digits Student ID"

2. Test GPIO Interface

2.1 Send Logic "1" to Turn ON On-Board LED, then flash @ 1 Hz Frequency

2.2 Send Logic "0" via gpio port to turn off On-Board LED;

2.3 Read input via GPIO port, if the input is "1", then Send Command to Turn ON On-Board LED

2.4. Read Inputs via GPIO Port, if the input is "0" then, Send Command to Turn off the On-Board LED.

Discussion on CPU Architecture
then on this Assignment.

Action 1: Form 4 person

Team, send me email
First, Last Name, SID

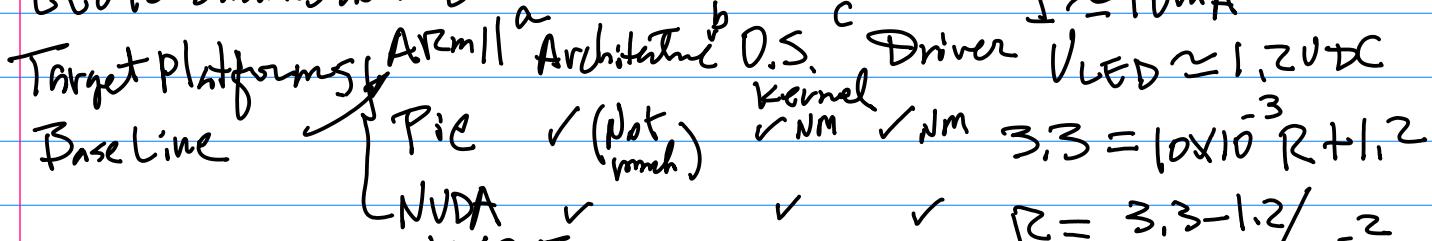
Feb 10.

Homework 1 Due Feb 24 (W) 11:59 pm.

$$V_{GPP} = IR + V_{LED} \dots (1)$$

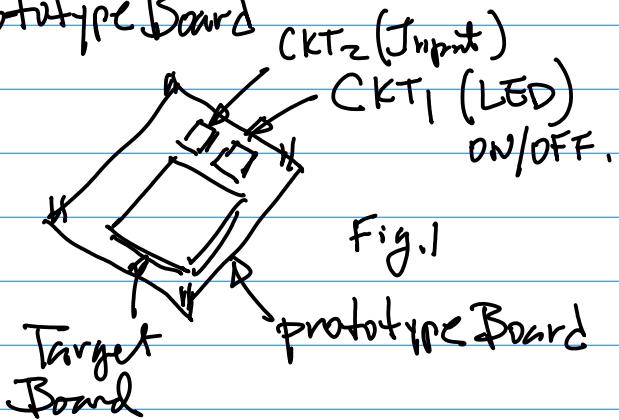
CmpE242 CANVAS

EE242 Submission to E-mail:



1° Target platform Built on

Prototype Board



Note: Find the pin(s) from Target platform.
 Pic3 GPIO14 — Pin 8

Consider Input Testing CKT

Input Pin { Read "1", PWR
 GPP } { Read "0", GND
 with toggle switch

Ref: I-tec-Hardware Board —

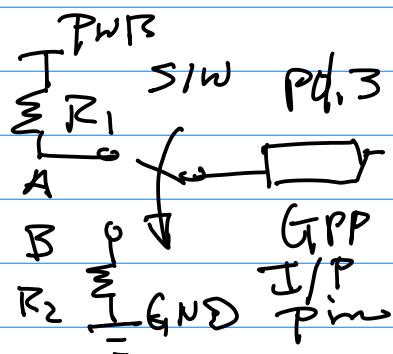
print "Hello, the world", Drive LED ON/OFF

C/C++ or Python

Has to be the LED
ON your prototype

2° GPIO Testing
 GPP Output
 CKT₁

{ "1" Turn on LED
 "0" Turn off LED }

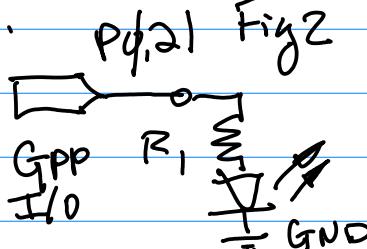


Design:

Identify the GPIO pins for Input
and Output Testing CKT.

$$R_1: 3.3 / 10 \text{ mA} = R_1$$

$$3.3 / 10 \times 10^{-3} = 330 \Omega$$

Pd.3 GPP Input pin
 Pd.21 GPP Output Pin

$$R_2: 3.3 (\text{from GPP}) / 10 \times 10^{-3} = 330 \Omega$$

9/
0.5.
I/O
function

System (Architecture) & Software Design. Boot Loader

3° Source code, Binary \rightarrow Zip.

4° One page Report IEEE paper
format (Template is given on-line)
github

5° Form 4-Person Team Submit

First, Last Name, 4 Digits SID

E-mail Contact Information

Indicate Coordination of the Group

By Thursday.

Note: All work has to be individual

6° Short Video Clip (5-10 seconds)

Shows the Prototype Board and
Screen Capture.

Feb 15, Monday

Topics: GPP I/O, Device
Driver.

github: Z-2020S-Lec2 ...

=
Example: 1) CPU Broadcom BCM2835

2) PWR1 (SP3), 2, 4 (5VDC)

3) GPIO for Homework \rightarrow Choose
pin those not

marked w/ other
function

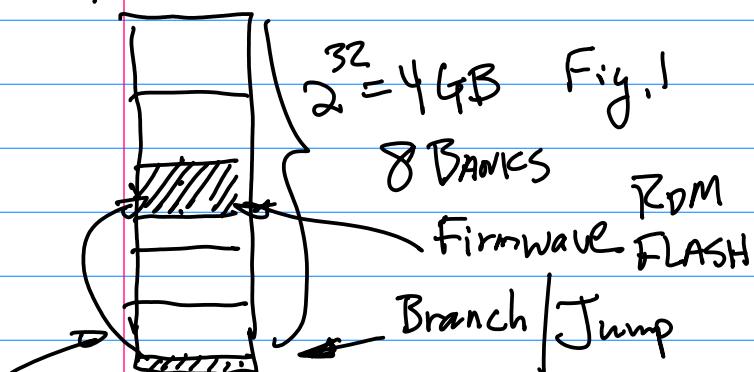
Concept of Device Driver

↳ Architecture + memory map

↳ Software + Kernel (OS)

↳ Device Drivers

Architecture

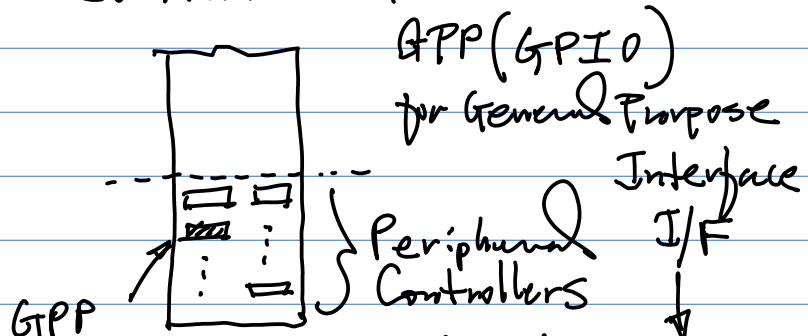


BANKs } (PP2.Fig4)

0x0000_0000 PWR-up Address

Addr. when CPU is powered up, it
will fetch the 1st Instruction
from this addr.

CPU Architecture



Note:

1° SPRs are 32 bits.

2° SPR's Formations into 3 Categories

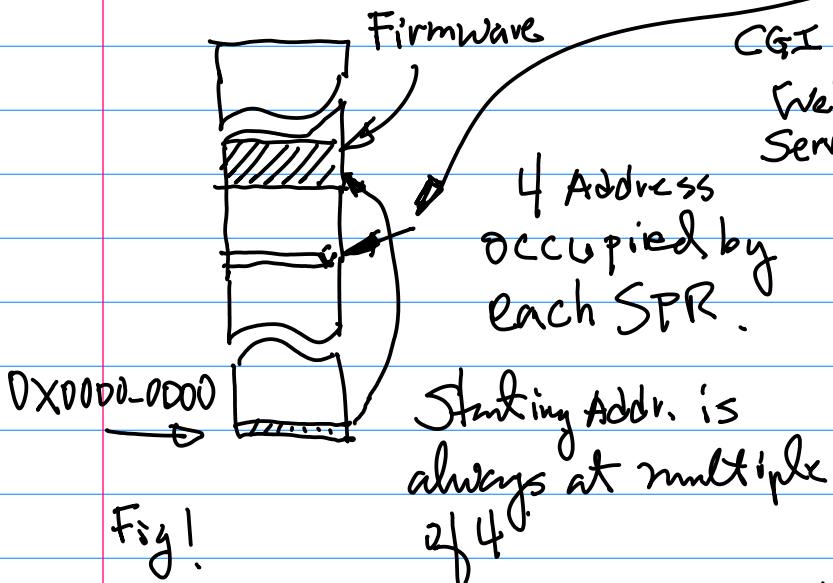
- 1° Control Function, Init & Config
- 2° Data
- 3° Pull Up/Down

3° Map 32bit SPR onto memory



32bit SPR

which is mapped to the memory location



1° PWR up Address: ~ when CPU is powered up, it will fetch the 1st Instruction from this location ~ /CMPE242/2018S-29-CPU (ARM11 CPU Datasheet)

Feb 17 (W)

Ref: CPU Datasheets

1° github ~ 2018S-29

2° Boot up Sequence

Firmware: { Boot Loader
ROM/FLASH } I/F

3° OS. Image is Being Loaded

Then user space program can be executed, And the device(s) can be accessed via Device Driver in

Kernel space, open V_TF Space

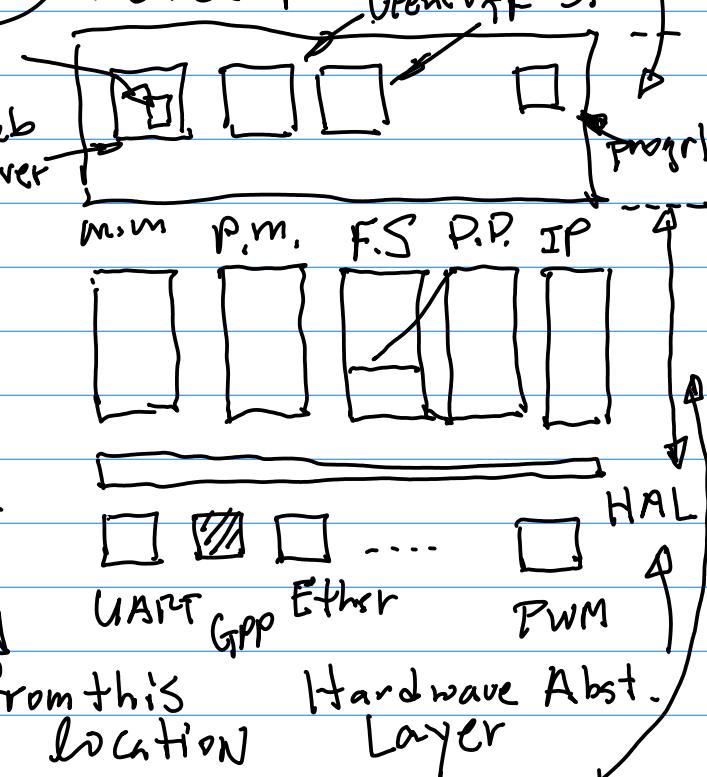


Fig 2. Kernel Space

Example: Prog1.C → USER Space

UserSpace \rightarrow Open("dev");
 $fd =$

"dev"

\downarrow
 Path Device Driver
 location in
 the Kernel Image
 \downarrow
 read(fd,
 Buffer, Size for GPP Input
 Testing)

GPxCON

$$2^{32} = 4 \text{ G}$$

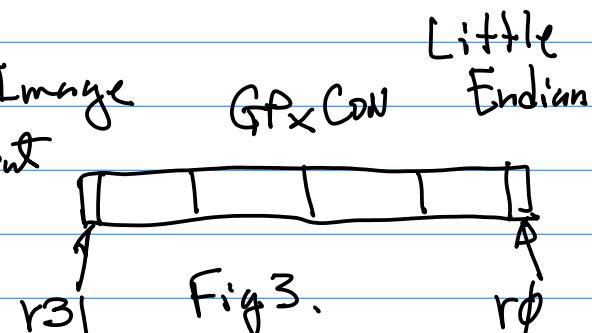


Fig 3.

Test GPP Output GPx
 pin 8 as an output

pin

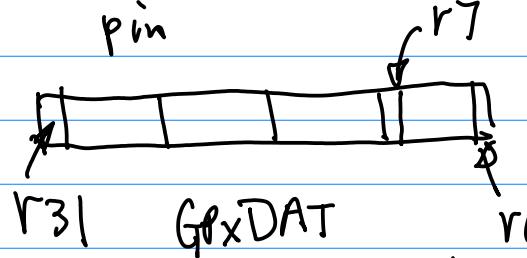


Fig 4

4. Kernel Space:

OS : manage ~ Resources &
 Policy

DeviceDriver(s) : A collection of
 program(s), manage/manipulate

Special Purpose Registers, to be
 able to utilize peripheral controller(s)

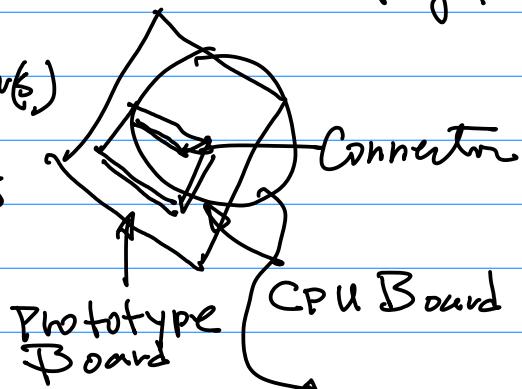
Init & Config of Special Purpose Registers

Example: Naming Convention for
 SPRs.

1° For Control Register

Prefix + Root + Postscript
 (3) (3) (3)

GPx CON



Make
 Pin 8
 as an
 output

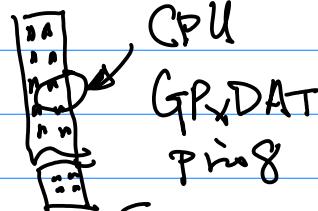
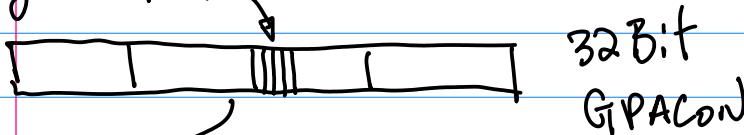


Fig 5
 Set GPxCON
 pin 8 to 1

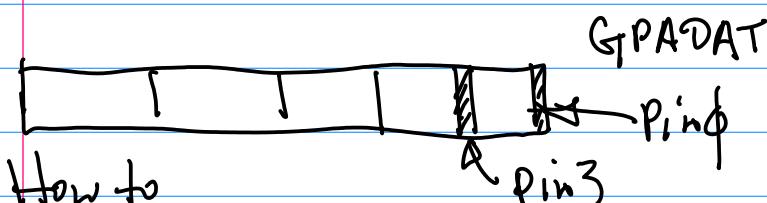
Question: How many control functions

can one Control Register realize? $GPxCON[7] = 1$

$\text{GPACON}[3:0] = 0000$ Input
 Fig 1. $15:12 \downarrow 0001$ Output



Question: Define Pin3 GPP Input.
 $(\text{Pin}0, 1, 2, 3 \dots)$



How to Perform Init & Config?

First GPACON \Rightarrow GPA3 GPACON[15:12]

\downarrow
 $\text{GPACON}[15:12] = 0000$ for Input

$\text{GPACON}[15:12] = 0001$ for Output

Suppose that is task (Init & Config)

Question: Find Binary Patterns to Define GPACON?

0x0000_1000 ✓

Move/Copy this Binary Pattern

to 0x7FOO_8000

Note: 1° All CONFDAT Registers for GPA ~ GPR

2° To Be Able to Define Input/Output Based on Table Look up.

3° To Be Able to Generate Binary Pattern for Init & Config.

Action 2: Read CPU Details about
 ARM11 (Samsung)
 TX2 (Nvidia)
 Pix3/4 (Broadcom)

Note: Programming Kernel
 Sample Drivers.

4° make menuconfig
 5° KConf Script

Note: To Be Able to Write
 Simple Script.

MODULE NAME
(1) Define 3 options to bind
(2) followed by "...."
Verbage
CPU - manufacturer ID + Device ID
depends on
help
Text Info

Example of C Code Driver.

`printf()`

Kernel Space

Cmpe242

From Fig.3 (pp 11) $GPxCON[7] = 1$ Kernel Space Driver Development
Hands-On Experience.

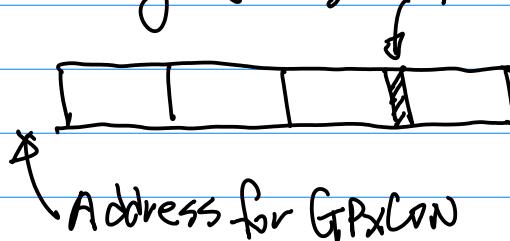


Fig.6

Suppose Addr. = 0X4008_0000

Find the BANK which holds
this SPR, addr, GPx Controller

Note: GPxCON occupies 4 Bytes

Action 1
Kernel Source + Tool Chain
Distribution + ARM
250mB + plus
Document CPU

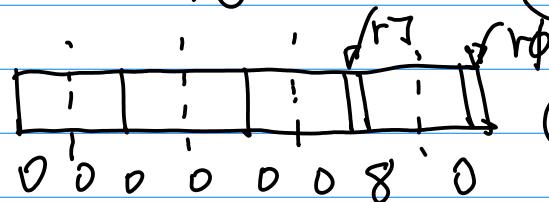
Action 2
Perform Init & Configuration
Define the Behavior of
of Peripheral SPRs
Controller.

Example: Samsung CPU ARM-11
2018S-Z9 - ~ (Datasheet)
Chapter 10, GPA-GPQ Table
Look up

Section 10.4.1.

GPACON's Address

Init & Config Pattern (Binary)

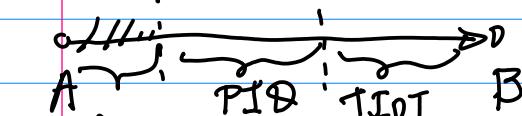


$GPxCON[7] = 1$

0x80 → Define

Homework ON CANVAS & github

Feb 22nd (Monday)



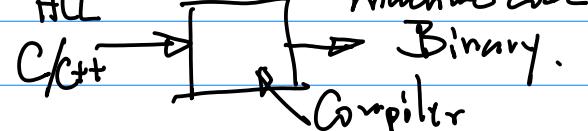
Drivers. Background → Architecture + Mem.map

Architecture → IDE → Implementation
Kernel DD

GPP I/O Testing

Section 10.5.1

within GPACON[3:0]



CMPE242

Feb 24 (Wed)

Example: ARM ToolChain Based
Environment → ARM11

Linux O.S. Kernel Target

Source → Image →
Compiled & Built Source Code
@ Kernel
Device Drivers. $\text{tmp} = \text{readl}(S3C64XX_$ $\text{GPKDAT});$
GPK PortK
DATA Register.

ID

*.h

posting to make
O.S. to Target CPU

#define S3C64XX_GPK

(1)

O.S. Source
Distribution
ARM11
NVDA TX2
Pre-Built
Compiled + BuiltDAT 0Xnnnn-nnnn
→ Example $\text{tmp} \&= ~(1 \ll (4 + \text{arg}));$

User Space Example

mask
folder (Program
I/F to Device Drivers) " & " Bitwise
AND

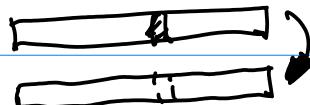
" ~ " Negation → 0

AND

" & " Bitwise AND

Char & Drivers

locate Drivers for the Target CPU "minibutton..."

Note: 1° Required to Be Able to "hello.c" Mask → masking SPRX
write a Simple Driver Test Code. Logic Operation
② Bit wise level2° init 2 modules
exit

Devices is installed

Example



3° Printk(" "); → involved

Kernel Space

Check the Source Code (to Be posted
On the github)

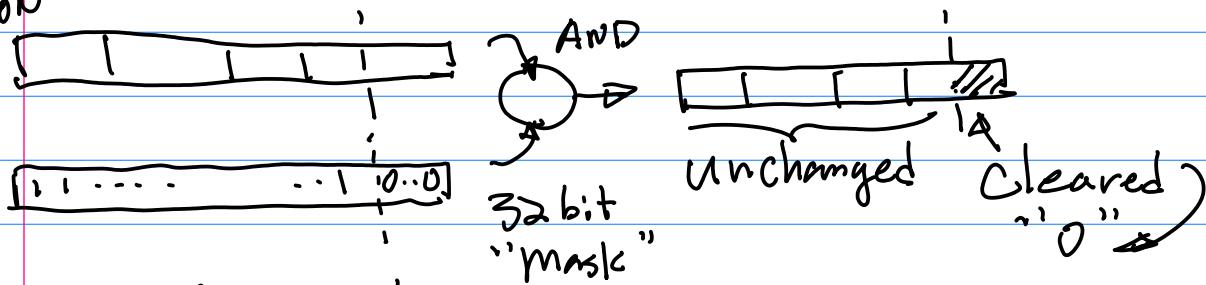
* ioctl.h . #define DEVICE_NAME "leds0"

GPKCON
GPKCON[3:0]

to init GPKφ

Clear GPKCON[3:0]

GPKCON



CPU Datasheets, ARmI

Section 10.5.5, pp. 320

- (1) SPRs { CON ✓
DAT ✓
PUL }

Powermanagement

(2) $0x7F008080 \rightarrow$ GPECON

(3) GPECON : Total 5 pins GRPP I/F

GPE3 GPECON[15:12]

Once Driver Done \rightarrow make menuconfig

Load Kernel Image
to the target
Board

Done, Pie3B+/
NAND
TX2

module

Built { "m"
"*"
"v" id }

Action 1: { (1) Download
NVDA TX O.S.
Distribution
(2) Download
ToolChain
(3) Download
Document }

NVDA
Broadcom
Samsung
ARM Tool Chain

DR, Load Driver module

Copy Driver module *.ko to SD

Mount it (mount)

Jonathan <β> Prototype

Akhil

PID Controller {
= P.I.D. ~
= PWM Stepper
motor
= L5M303 I/F}

Target platform. (O.S. Running)
Linux

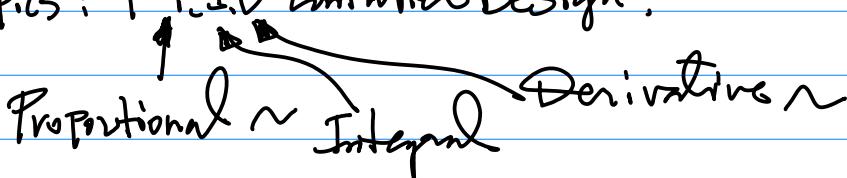
insmod : Install module

Ref: github : 2018S-12 - ...PID

2018S-14 - ...Stepper Motor

Comparison of the Actual output with the Desired Output

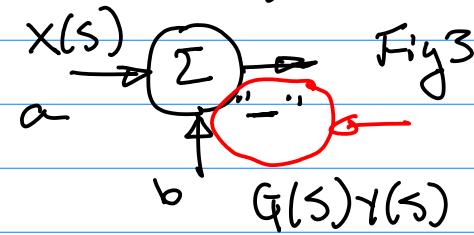
Topics : 1^o PID Controller Design.



Note :

Input
a - b

Fig 3



Consider Design CNC machine

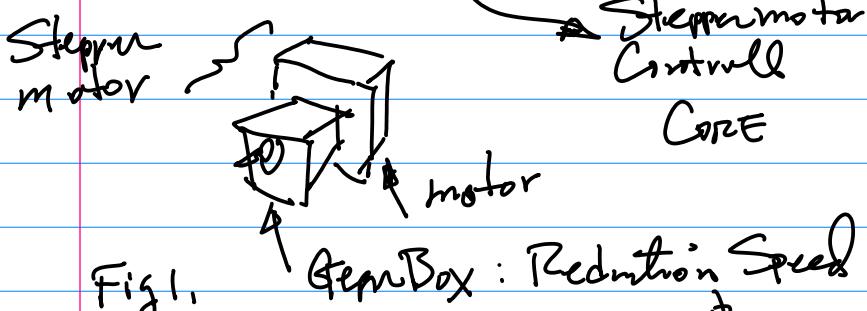
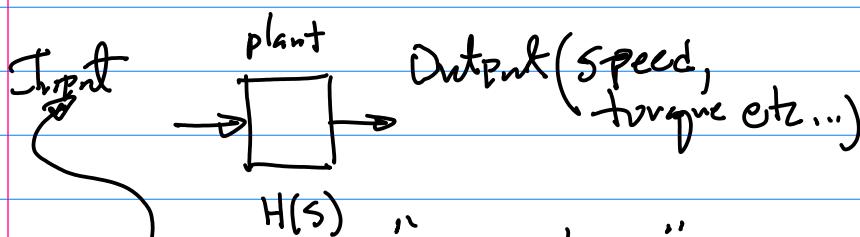


Fig 1. Gear Box : Reduction Speed
Gain Torques.

Comparison
{ a - b ✓ Difference
a/b Error

Negative Feed Back Loop



Note 2^o

Performance

Enhancement

- ✓ P: Proportional Controller
- ✓ I: Integral
- ✓ D: Derivative
- Frequency Domain

$$E(s) = \Sigma(s) - G(s)Y(s)$$

... (1)

$e(t), e(\tau)$ Time Domain

Control Action Proportional to the error, in Time Domain

Note^o Stepper Motor Drives Are of the Equipped with P.I.D. Control System.
functions.

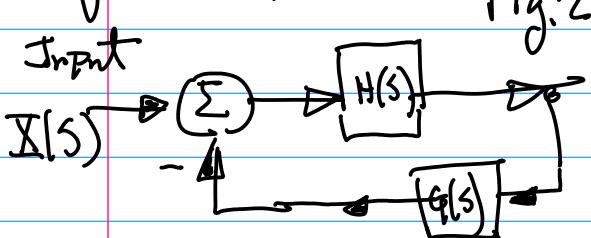
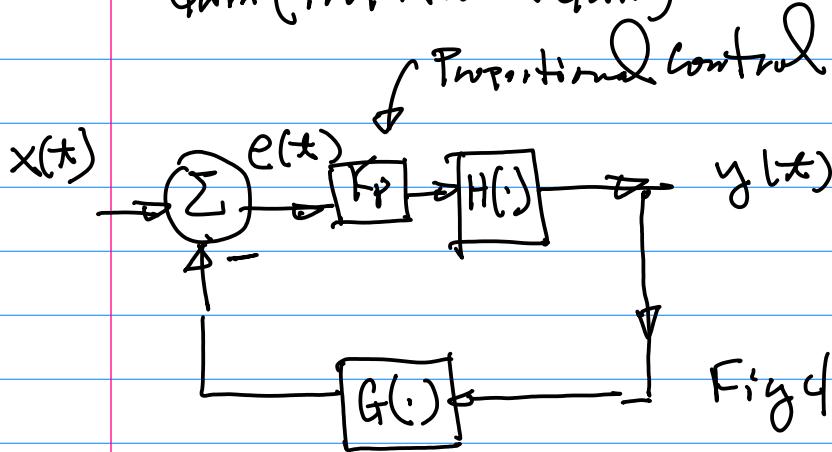


Fig 2

Output
 $Y(s)$
Sensor(s)/Transducer

$K_p e(t) \dots (z)$ (at this moment) To Solve Integration
 ↑ Gain (Proportional Gain) \rightarrow Compensation Due, \rightarrow

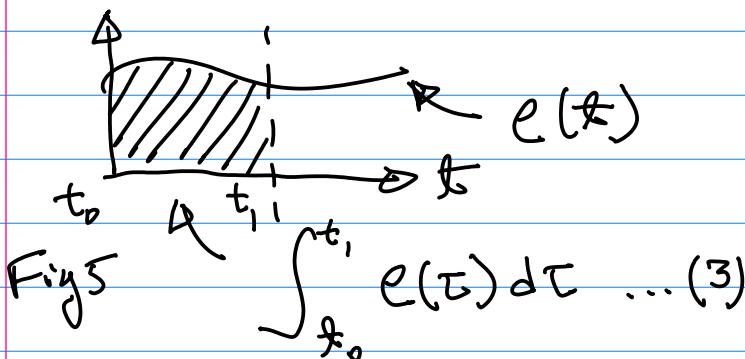


Let's ignore the error $e(t)$, e.g. $e^2(t)$

Note: Don't use Absolute Value of the error.

Fig 5. ~~|e|~~ No Derivative

Now, Integral Controller



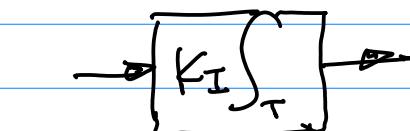
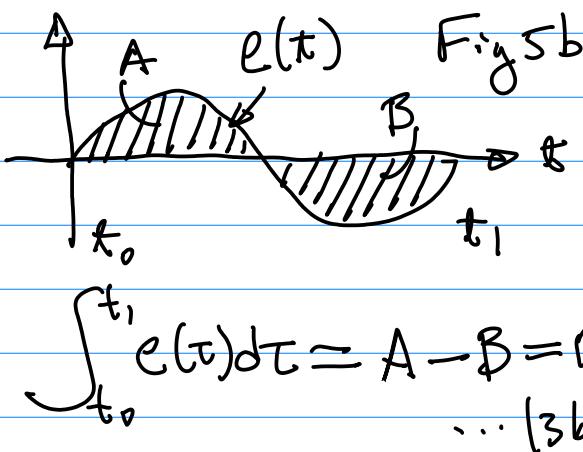
Therefore

$$\int_T e^2(\tau) d\tau \dots (3c)$$

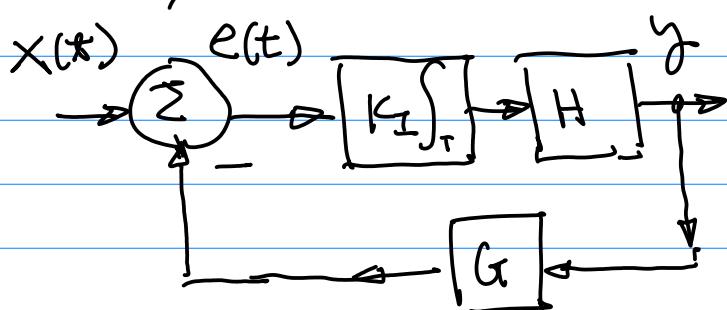
Integral Controller

$$K_I \int_T e^2(\tau) d\tau \dots (4)$$

taking into the consideration of Controller's Accumulative Performance



Hence,



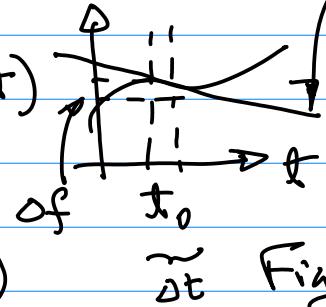
Consider Derivatives

Background: $f(t)$

$$\lim_{\Delta t \rightarrow 0} \frac{\Delta f}{\Delta t} = \frac{df}{dt}$$

Changing Rate

... (5)



Derivative

P: Proportional

I: Integral (History
Accumulative)

D: Derivative (Pre-
dictive)

- Action
- 1. Stepper motor Drive
motor Ready
 - 2 GPP + PWM Driver
(Software)

$$\frac{df}{dt} \Big|_{t_0} > 0 \rightarrow f(t+t_0) \uparrow$$

$$\frac{df}{dt} \Big|_{t_0} = 0 \rightarrow \text{unchanged}$$

$$\frac{df}{dt} \Big|_{t_0} < 0 \rightarrow f(t+t_0) \downarrow$$

March 3rd (Wed)

Let $e(t) = f(t)$

$$\frac{d}{dt} e(t) = e'(t) \dots (b)$$

Build derivative controller

$$K_d \frac{d}{dt} e(t) \dots (bb)$$

Topics: 1° PID Controller
with Each Subsystem (3)
Combined together.

2° C/C++, Python Implementation

Convolution

3° Motor Drive

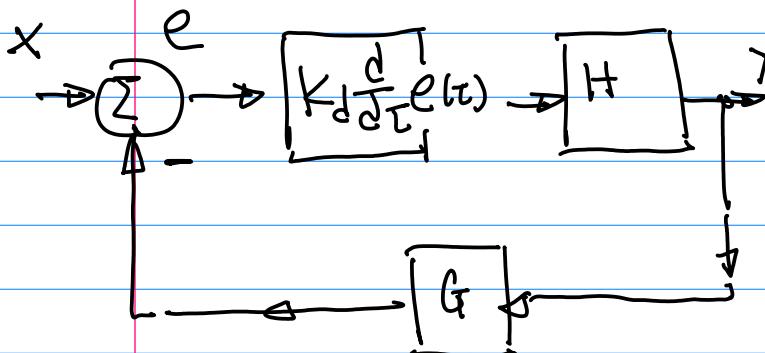


Fig 7b.

Predictive

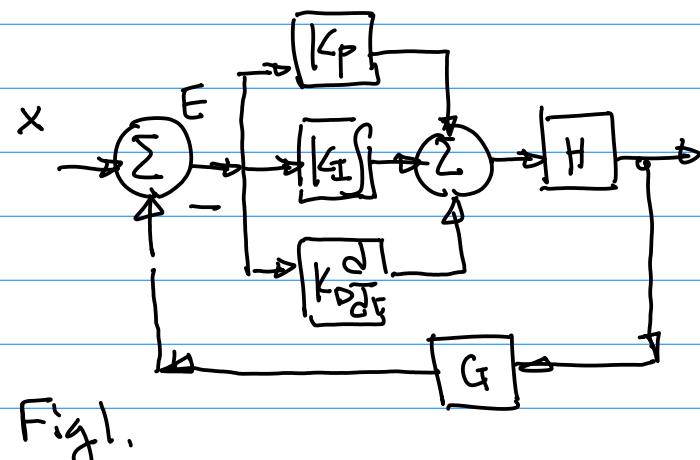


Fig 1.

2018S-12-Lecb-PID

{ Transfer function
Characteristic Equation }

Full Step \rightarrow ? Displacement

(How much can this wheel travel)

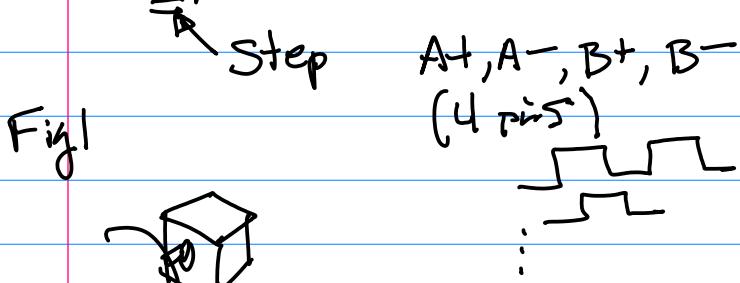
GearBox: 68:1

Example: Implementation Technique
P.I.D controller.

Note: Motor Drive, NEMA 14 or 17

2018S-14- Stepper Controller

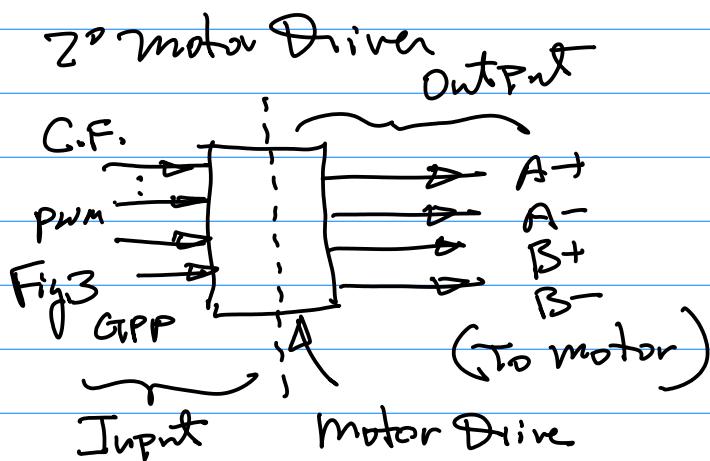
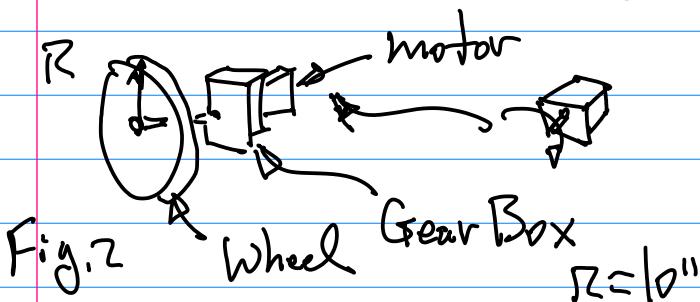
1° Stepper motor



$a = \frac{360}{200} (\text{step})$, 20 steps

$b = \text{One step } 1.8 \text{ Degree}$
 Half step 0.9 Degree
 $\frac{1}{4} \text{ } 0.45 \text{ Degree}$
 $\frac{1}{8} \text{ } 0.225 \text{ Degree}$

Example: Robotics/Self Driving.



Configuration: $K=2 \text{ or } 3$
 $\text{pins } K (2^K)$,
 $Z^2=4$
 $Z^3=8$

PWM: Pulse Width Modulation

Target Platform: Pi e / NANO / TX2
 TIGSC-V Hardware: Pins

Software: Device Driver
 Controls the speed of the motor

 $f_{\text{PWM}} \rightarrow \text{Change Speed};$

$f_{\text{PWM}} = 1 \text{ kHz}$, Change Only
 Cycle \rightarrow Speed

Note: CAD Design Tool

Eagle Linux & Windows

ORCAD (Cadence) \rightarrow Eval Version
Free 30 Days
Link

Schematics \rightarrow
Registed in Project Report

Action 1: Build motor Drive
and Connect to NEMA 14 or
Equivalent, Try to Drive it.

Please bring your Prototype Board
to the class for Show & Tell.

Example: P.I.D. Implementation.

1. Error Signal $e(t)$, From Fig. 1 P.P.17

Find Error of Lsm303 Direction
Optical (self Driving
Encoder Robot)
Displacement/
Wheel travelled

Suppose the error is denoted as

$e(t) \rightarrow e(k)$ K index for

Continuous Discrete time
System System

Sampling process

Suppose the set of error data is tabulated in the following Table:

Time Index	Error $e(k)$	Notes
K_0	0.1	
K_1	1.1	
K_2	1.5	
K_3	0.5	
K_4	-0.1	
K_5	-2	
:	:	