

August 23rd (mon).

CMPE240
Section 1.

CMPE240

HARRY LI.

Email: huagli@sjsw.edu

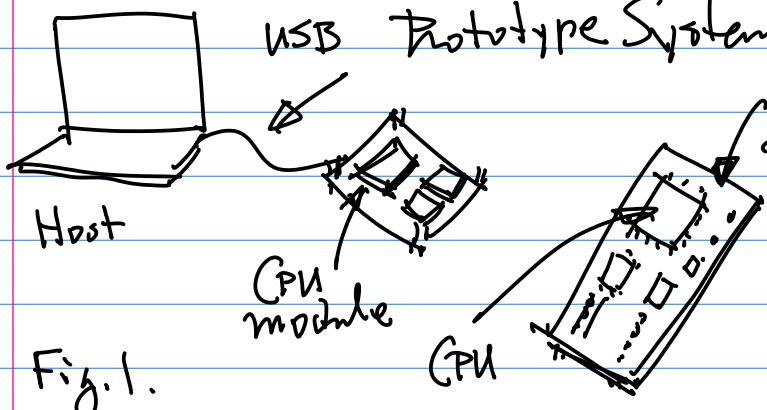
Text message (650) 400-1116

Office hours: M.W. 3:40-4:40 pm.

Advanced Microprocessor Systems

=

Prototype System
with A CPU module



GPU (Graphics Processing Unit), Array of Processors, Machine Learning, AI.

Autonomous Systems, Nvdia Jetson

Tx2.

Textbooks, References

1° NXP LPC1769 GPU Datasheet
80+ pages

Homework: Download pdf. Before

Next Monday, Aug. 30th.

2. LPC1769 Schematics
of the CPU module

3. Nvdia Jetson NA10
Datasheet on Tx2 (6 CPU + 256 GPU)
4 front pages. 5% Bonus.
(optional)

4. RISC-V Open Source
Architecture, A Super Set
of ARM, FPGA, Verilog,
SoC. + RTOS. (optional)

A proposal (One +5%
paragraph) By Sept. 1st
(Wed). Submit to my
Email;

Note: Buy LPC1769 GPU
module.

digi-key.com,
monster.com, etc.

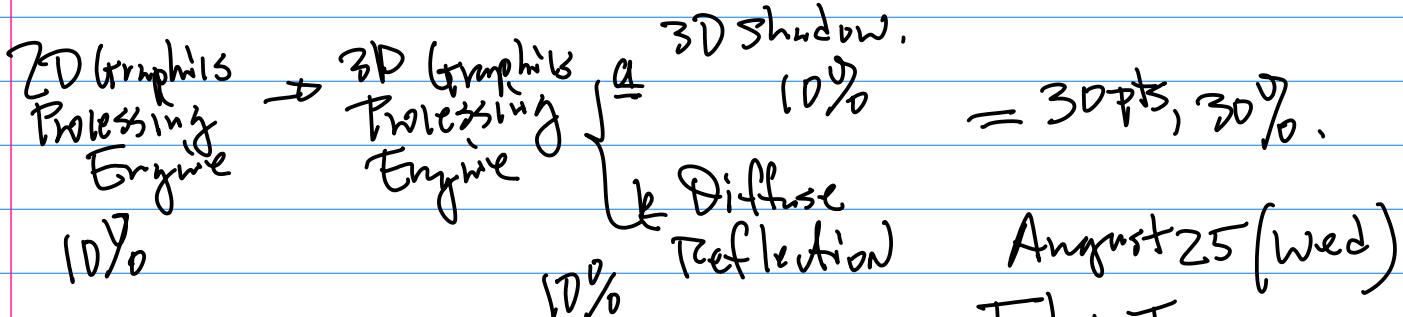
Grading Policy & Projects
2 projects
(phase I & II)

2D Graphics
Processing
Engine → 3D Graphics
Processing
Engine

→ 3D Graphics
Processing
Engine

CmPE240

2.



midterm: 30%, Final 40% (Comprehensive)

Option 1. (5%+) NVDIA NAND

- a. Likely Device Drivers, O.S. C/C++, Python.
- b. I/O Interface: "EdgeAI"

GPIOD, SPI.

Option 2. (5%+) RISC-V Target

SOC, FPGA Board,

Proposal (one paragraph), Submission
By Sept 1st (Wed) via Email.

Policy On Project Submission.

1° Form 3-4 person Team.

2° No Source Code/Design material

Can be Copied, All Course material has to be completed individually;

3° Late project, 10% per week;

Tool for
Flashing the
CPU module

August 25 (Wed)

Today's Topics:

1° Bill of material

Reference: github/finalisti
/Cmpe240/2018F

The B.D.M.

1. CPU module NXP LPC1769

3rd Party (DigitalArt), module
To Distributors

DigKey.com, Mouser.com
etc. Expecting Delays.
Lead Time over 8 weeks

Alternative { Re-use the previously
used module
Team (4 person)

Each person will need to have
his/her Board;

Option 1: NAND. @ 440+ pages

"firmware" Datasheet

= Jetpack 4.3 or Higher

→ (O.S. + Libs. + Packages)

Compendium

c Coding in Both user & Kernel Spaces. \rightarrow O.S. Distr.

Tool chain, Device Driver Debugging
2. Development;

Option 2. PISC-V, verilog, FPGAs.

2. Power Regulator ICs such as

7812, 7805 ... 1117

$\underbrace{\hspace{1cm}}$

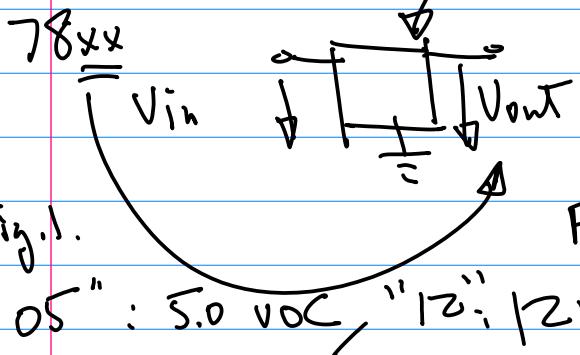


Fig. 1.

"05": 5.0 VDC, "12": 12 VDC

$$V_{in} \geq V_{out} + 1.5 \text{ VDC} \quad \dots \quad (1)$$

DC Voltage Source

a About 7805

1000 mW.

b 7.5 VDC

OR

$$9. VDC \quad (a) 1000 \text{ mW} + 500 \text{ mW} \\ = 1500 \text{ mW}$$

c Why Do we use it?
Current Rating.

\hookrightarrow Deploy the System.

3 "Glue" Components / Resistors a

c LEDs. (Red, Green) for Debugging purpose, for PWZ.
(GPIO), $I_{LED} = 4 \text{ mA}$

d Connectors.

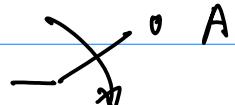
d J1 for PWZ Input R pin

d IN-Line pins.

d Breakable

to mount CPU module.

e Switch. S/W1: to toggle PWR.



f Wire for Wire Wrapping / Soldering

28-30 AWG

4. Color LCD Display module

a SPI (Serial Peripheral Interface)

b Software Graphics (Driver)
C/C++ Lib.

to Activate / Initialize LCD.

MCUXpresso (I.D.E.)

\downarrow
S.T. Lib.

5. "Other" thing.

PJ-45 Connector

Sept. 8 (W)

Topics: 1. "Hello, the World" program

J Hardware Implementation

NXP MC91Xpresso.

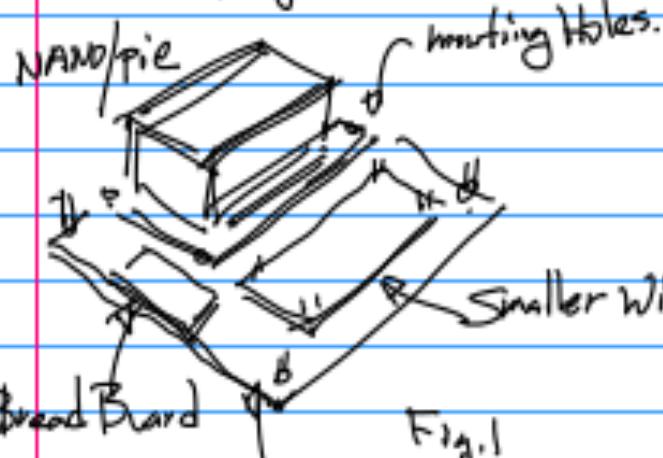
a. Installation of MCU

Xpresso.

b. github.com/hualili/CMPE240/2018/L7C1769 Patch, Import this patch
to your Xpresso.<https://github.com/hualili/CMPE240-Adv-Microprocessors/blob/master/1769%20patch.zip>Note: Wirewrapping Board with
"Stand-off's (Legs)

Homework: Next Show-and-tell

Wirewrapping Board;



"Top Plastic"

On the Board: a) Stand-offs.

b) Connector(s) for External JTAG

Prototyre Board Build Up

External Power CKT (Red LED should be

GPP Testing CKT

a. Wirewrapping
Board (L7C/NANO)
Pie

b. Stand-offs.

Pie

External Power CKT (Red LED should be

included)

Implementation/Design

of the CKT.

Architecture Aspects,

CPU Architecture, M. Map.

→ PVRIC(7805), with
Red LED

CPU Architecture :

1. 32-bit Architecture

CPU Architecture

a. ALU 32bit

Arithmetic/Logic
Unit.

b. Register File,

A Bank of Registers. 32 bits
GPRs

General Purpose Registers

Those Registers that can
participate Any meaningful

Arithmetic/Logic Operations. To Define/Determine the Behavior
of peripheral Special Purpose Registers.

SPIRs 32 bit

Naming Convention: Controllers.
6 letters

Common Design for SPIRs:

- 1° Central Register(s) per Each Peripheral Controller

CON



- 2° Data Register, DAT

- 3° Pull-up/Down (Electric Characteristics)

- C. Data Bus, Bi-Directional " 32 bits

Information Flowing Both Directions.

Address, "Uni-directional" from

CPU to the Outside. 32 bits

Notation: 32 bit Register

$\text{GPR}_X[31:0]$

LSB

$$2^{32} = 2 \cdot 2^{10} \cdot 2^{10} \cdot 2^{10} \dots (1)$$

$$2^{10} = 1K, 2^{20} = 2^{10} \cdot 2^{10} = 1 \dots (2) M$$

$$2^{30} = 1M \cdot 1K = 1 \text{ gig} \dots (3)$$

... (4)

$$2^{32} = 2 \cdot 2^{30} = 4 \text{ GB}$$

3. Memory Map.

$\text{GPR}_X[31]$

Fig. 2

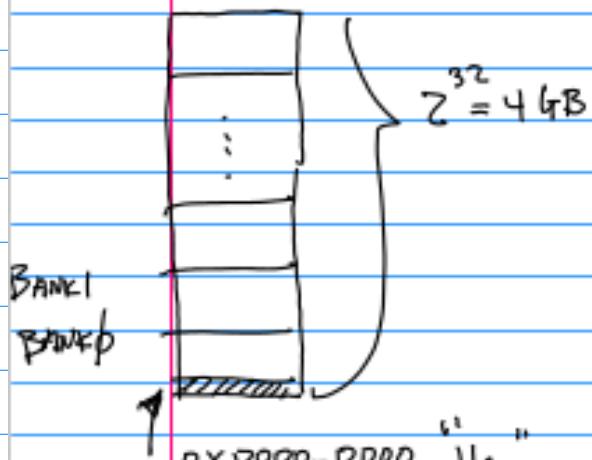
$\text{GPR}_X[0]$

For Address Bus, Addr[31:0] =

$a_31 a_{30} \dots a_1 a_0$

CMPE240

8



Define Starting Addr. of Each Bank:

a_31	a_{30}	a_{29}	a_{28}	
0	0	0	1	BANK0
0	0	1	1	BANK1
0	1	0	1	BANK2

⋮
BANK7

Fig. 3.

32 bits for the address
& 8 bits for this memory

Write the address for Each Bank.
"Starting" (32 bit)

a. PWR-up Address:

CPU will fetch the 1st
Executable from this memory
Location.

→ 0x0000-0000

for ARM

Note: for x86, the PWR-up

Address: 0xFFFF_FFF0

For BANK0 : 0x0000_0000

BANK1 : 0x2000_0000

.. 2 : 0x4000_0000

Example: CPU Datasheet pp. 13.

GPIO 0x2009_C000

b. BANKS.

$$2^{32}/8 = 2^{32}/2^3$$

$$= 2^{29} = 2^9 \cdot 2^{20} = 512\text{MB}$$

How many Bits Do we need to
uniquely define Each Bank?

3 bits → a_3, a_{30}, a_{29}

c. Collection of SPRs are
mapped to here, e.g.

Addr. for SPRs are
mapped to here

b Which memory Bank holds

this GPIO? BANK1
whose starting Address is

0x2009_C000

Sept 13 (mon)

1^o Today's Topics: Integrate Architecture Discussion with Software Development

IDE. Objectives: To

Write first C program for testing purpose

Example: Starting from CPU

Memory map \rightarrow 8 Banks
PP13

1st 256 kB
= Flash

0x0000_0000,

Rest of the Banks, such as
Mem. Controller, Peripheral
Controller

\downarrow
Peripheral controllers \rightarrow SSP1 (SPI)
on the Mem. map.
APB \neq APB
 \downarrow

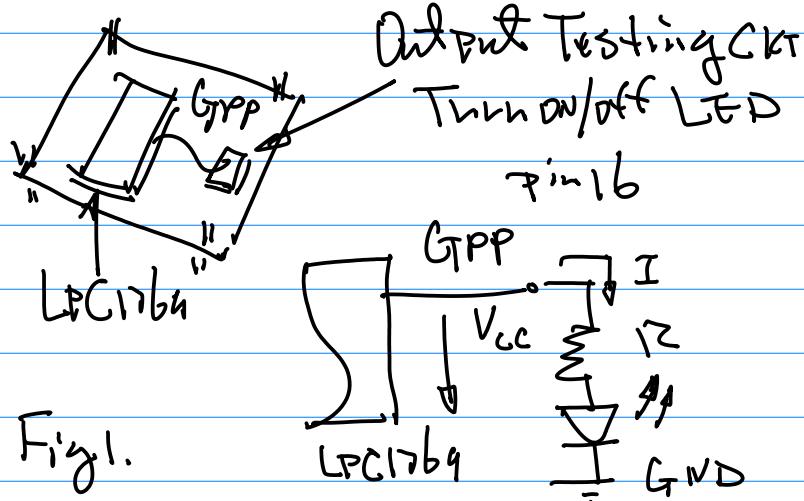
S.P.R.s. \leftarrow 0x4003_0000
"Con" 3 Letter
Size 4000

a Naming conversion Prefix Root + Postscript

"SPICON" \rightarrow "SPICON001" for Example \rightarrow C Compiler/C code
3 letters 3 letters 3 letter

= b Definition: Are those SPRs for the init & Config of a Peripheral Controller.

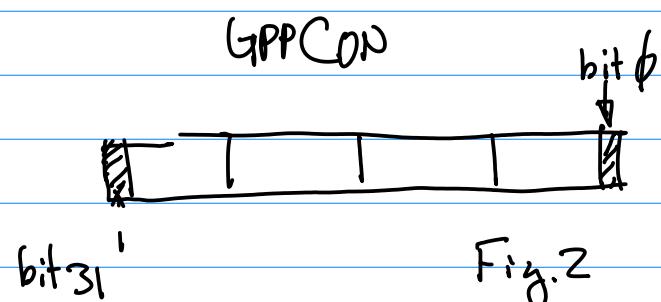
Example. GPP



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

$$I \approx 8 \text{ mA.} \rightarrow R \approx 2 \text{ k}\Omega \quad 300 \text{ }\mu\text{A}$$

$$V_{LED} \approx 1.8 \text{ VDC}$$



Where to find GPPCON on the
memory map? \rightarrow Addv.
of GPPCON is described on
CPU Datasheet.

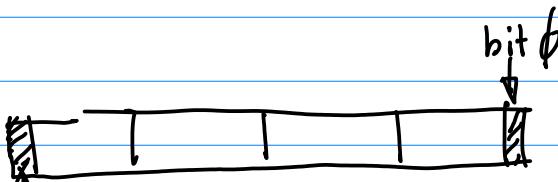
CmPE240

2^{32} Possible Combinations
of Init & Config. Feature

GPP (General Purpose Port)

32 pins, Define pin 1/b as output
pin.

We have to use the following
init & config pattern:



bit 31 0x F2ff_ff00

To make pin 1/b as an output.

First, "Port" the Architecture

Compiler to the target

#define GPPCON

then, Copy 0xF2ff_ff00 into
this memory location.

Homework: Show + Tell
By Next Week Installation of mCU
+ Import LTC17b9.

Sept 15 (W)

Architecture

Today's Topics: GPIO Design

Reference: 1° 2021F-105 ~ On
ID: UPI0

C-Code for Init & Config
is required

Example: Make GPP for
Input & Output
Testing.

Hardware

Software { NXP MCUXpresso
Import GPP Sample
"zip"

Design Step 1.

Identify / Select GPP / GPP-Pins
P_b, P_b3

(Connector → CPU → Selection
Data Sheet)

Step 2. Define P_b.2 Asent ,

P_b.3 As Input ,

Design the Hardware

Step 3. SPRs (Special
Purpose Registers) for

the GPP Peripheral
Controller

Connector → CPU
J2-21 → Pin → Data
Sheet
...
P_b.2
P_b.3
↓
SPRs

Note: SFRs commonly defined / utilized are

GPFCON

where $x = A, B, C, D, \dots$

GPFCON

GPF DAT (32 bits \rightarrow 32 pins)

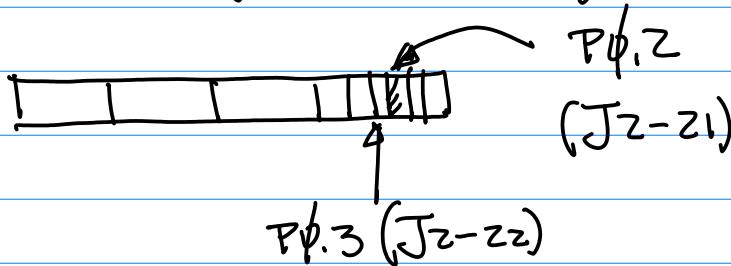
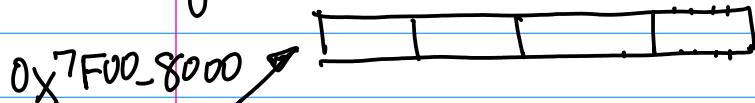


Fig. 1

Find Table on CPU Datasheet to define Pb.2 output, Pb.3 as input.

Example, Samsung Arren II Datasheet
pp312

Fig. 2



Question: Define Binary Pattern for
Find GPFCON to make
its pin 2 as an output?

Sept. 20 (mon) Topics: 1^o GPF

SFRs, IDE, Sample Code;

2^o 2D GE.

Example: git(class)

2021F-105-GPP...

i. Naming Convention in C Compiler

LPC_GPIO0->FIODIR

LPC_GPIO0->FIOSET

LPC_GPIO0->FIOCLR

↑
Target Peripheral Controller (Family)
↑
Special purpose Register

From the Example, qit, 2021F-105

From CPU datasheet, GPIOs are configured using the following registers:

1. Power: always enabled.
2. Pins: See Section 8.3 for GPIO pins and their modes.
3. Wake-up: GPIO ports 0 and 2 can be used for wake-up if needed, see (Section 4.8.8).
4. Interrupts: Enable GPIO interrupts in IO0/2IntEnR (Table 115) or IO0/2IntEnF (Table 117). Interrupts are enabled in the NVIC using the appropriate Interrupt Set Enable register.

Chapter 9, pp 129

PINSEL[5:4] for P0,2

PINSEL[5:4] = 00 for I2C

PINSEL[5:4] = 01 for UART
P0,2
Tx

pp 133 FIODIR Example,

P0,3 output, Find SPR?

Define bit values
for the output

Table Look up.

FIODIR

FIOSET, CPU Datasheet
Look up.

```
void GPIOinitOut(uint8_t portNum,
                  uint32_t pinNum)
```

```
{ if (portNum == 0)
```

```
{ LPC_GPIO0->FIODIR |= (1 <<
pinNum); }
```

```
else if (portNum == 1)
```

```
{ LPC_GPIO1->FIODIR |= (1 <<
pinNum); }
```



pin Num
Set to
"1"

$1 \ll \text{pinNum}$ // set direction
to pinNum

Logic Operation
"1" = "Otherwise"
"0" = "Or"
"&" = ?

Example: Set Pin

```
void setGPIO(uint8_t portNum,
             uint32_t pinNum)
{
    if (portNum == 0)
    {
        LPC_GPIO0->FIOSET = (1 << pinNum); // 1 as output
        printf("Pin 0.%d has been set.\n", pinNum);
    }
}
```

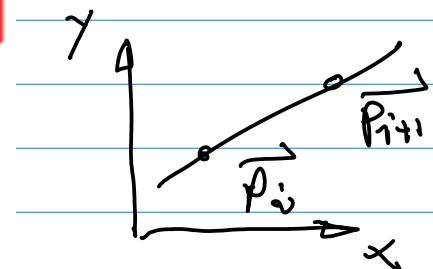
Turn ON LED
(Output '1')

$\vec{P}(x, y)$ Notation
 $\vec{P}(x, y) = (x, y)$
 $\vec{P}_i \rightarrow \vec{P}_i(x_i, y_i) \rightarrow$
 point(s),
 (x_i, y_i)
 Vertex, Vectors
 $\vec{P}_i = \vec{P}_i(x_i, y_i) = (x_i, y_i)$

Example: Clear the pin

```
void clearGPIO(uint8_t portNum, uint32_t pinNum)
{
    if (portNum == 0)
    {
        LPC_GPIO0->FIOCLR = (1 << pinNum);
        printf("Pin 0.%d has been cleared.\n", pinNum);
    }
}
```

Formulation for
a straight line



Now, 2D Vector Graphics

$\vec{P}(x, y)$ a point, vertex, a vector

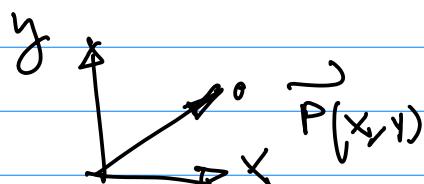
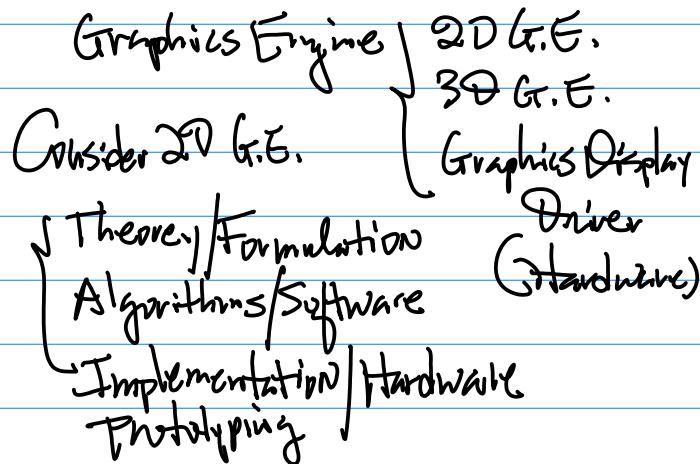


Fig.1

Sept. 22 (W)

Fig.2



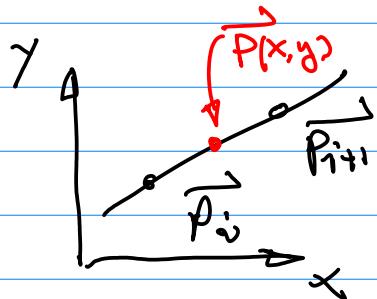


Fig. 2

Note: Need 2 points \vec{P}_i, \vec{P}_{i+1} to define a line

Let's define a direction vector

$$\vec{D}(x_d, y_d) = \vec{P}_{i+1} - \vec{P}_i$$

$$= \vec{P}_{i+1}(x_{i+1}, y_{i+1}) - \vec{P}_i(x_i, y_i) \quad \text{OR,}$$

... (1)

Example: Suppose given a starting pt. $\vec{P}_i(x_i, y_i) = (3, 4.5)$

$$\vec{P}_{i+1}(x_{i+1}, y_{i+1}) = (5.5, 6.3)$$

Find direction vector?

Sol. By Eqn(1), we have

$$\begin{aligned} \vec{D}(x_d, y_d) &= \vec{P}_{i+1} - \vec{P}_i \\ &= ((x_{i+1}, y_{i+1}) - (x_i, y_i)) \\ &= (x_{i+1} - x_i, y_{i+1} - y_i) \end{aligned}$$

Sub. the given condition

into the directional vector, we have

$$\begin{aligned} \vec{D} &= ((5.5 - 3), (6.3 - 4.5)) \\ &= (2.5, 1.8), \end{aligned}$$

In C/C++ Coding, we use the following equation, From Eqn(1), we have

$$\vec{D}(x_d, y_d) = (x_{i+1} - x_i, y_{i+1} - y_i) \quad \dots (1b)$$

$$\left. \begin{array}{l} x_d = x_{i+1} - x_i \\ y_d = y_{i+1} - y_i \end{array} \right\} \dots (1c)$$

$$\begin{aligned} \text{direction_x} &= x[i+1] - x[i]; \\ \text{direction_y} &= y[i+1] - y[i]; \end{aligned}$$

Let's briefly define a line

Need a pt \vec{P}_i , or \vec{P}_{i+1} ; and directional vector

$$\vec{P}(x, y) = \vec{P}_i + \lambda (\vec{P}_{i+1} - \vec{P}_i) \quad \dots (2)$$

↑ Starting pt ↑ scalar ↓ Directional Vector

Let

$x=0$, then $\vec{P}(x, y) = \vec{P}_i(x_i, y_i)$
Starting pt.

 $x=1$, then
$$\vec{P}(x, y) = \vec{P}_{i+1}(x_{i+1}, y_{i+1})$$

$0 < x < 1$, Any Point $\vec{P}(x, y)$ Between
 \vec{P}_i and \vec{P}_{i+1} .

$x > 1$ Any Pt. $\vec{P}(x, y)$ Beyond
 $\vec{P}_{i+1}(x_{i+1}, y_{i+1})$.

$x < 0$, Any Point Beneath
 $\vec{P}_i(x_i, y_i)$.

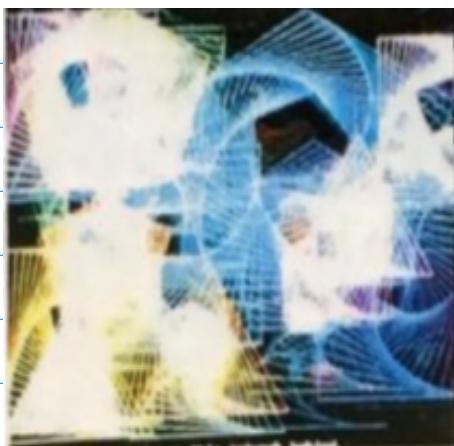


Fig 3a



Fig 3b

Screen Saver Design for LFC

2D G.E.

Rotating Squares And Trees.

Example: Design of Rotating Squares

Step 1. Define 4 vertices/pts

$$\vec{P}_i, i=0, 1, 2, 3$$

$$\vec{P}_0(x_0, y_0) = (b0, b0), \vec{P}_1(x_1, y_1) = (0, b0)$$

$$\vec{P}_2(x_2, y_2) = (10, 10), \vec{P}_3(x_3, y_3) = (b0, 10)$$

Based on the physical display device

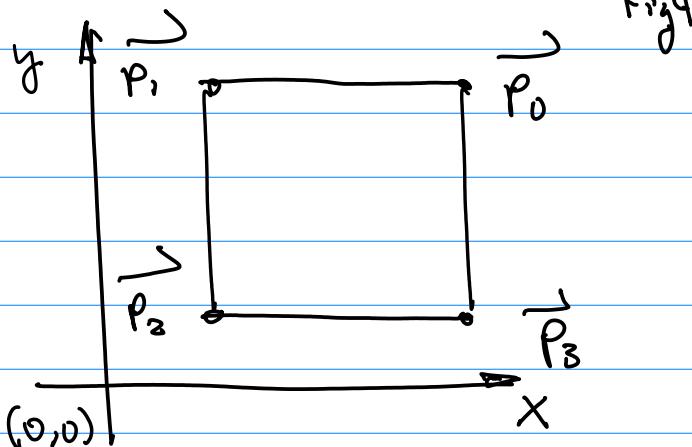


Fig 4

Note: Be sure to arrange \vec{P}_i in a
Counter Clockwise direction.
(for later 3D Hidden Line/Surface
Removal)

Step 2. Use

$$\vec{P}(x, y) = \vec{P}_i(x_i, y_i) + \lambda (\vec{P}_{i+1}(x_{i+1}, y_{i+1}) - \vec{P}_i(x_i, y_i)) \dots (1)$$

CmpE240

16

Prepric: LCD Soldering on
the mininwrapping Board,
Input Single line
Drawing Project.

Sept. 27 (Mon)

Homework, 2 pts. Due 1 week from Today

[Topics: 2D Screen Saver Design]

Requirements:

a. Build LCD Hardware

Interface;

b. Input Sample code from
github/finalili/CmpE240

2018S-1D-LCD-DrawLine.

Modify the code to Display 2D

Rotating Squares Using 2D
Vector equation;

Submission:

c. Project (Zip, Exported)
d. Screenphoto

Submission to CANVAS.

Announcement:

Office hours — The 3:40-4:40 pm.

Due to SJSL off-Campus
Program.

Example: Continued from pp 15.

Step 2. Use Vector Equation
to find 4 pts

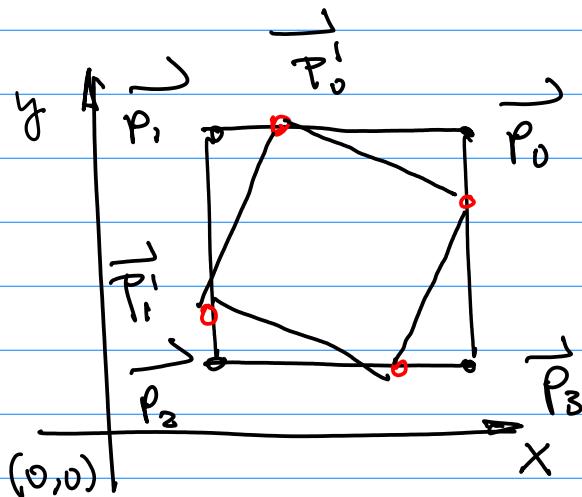


Fig. 1

Let $\lambda = 0.8$, for

line 1 (\vec{P}_1 and \vec{P}_1): Eqn(1), pp 15.

Calculate a point \vec{P}_0'

SuperScript: the
level of iteration;
for line 2, 3, and 4, we do the
same.

Line 2 ($\vec{P}_1 \& \vec{P}_2$), Line 3 ($\vec{P}_2 \& \vec{P}_3$)

Line 4 ($\vec{P}_3 \& \vec{P}_0$)

In Homework, level ≥ 10 .

Coding:

$$x = x_i + \lambda (x_{i+1} - x_i) \dots (1a)$$

$$y = y_i + \lambda (y_{i+1} - y_i) \dots (1b)$$

Hardware Implementation of LCD Interface.

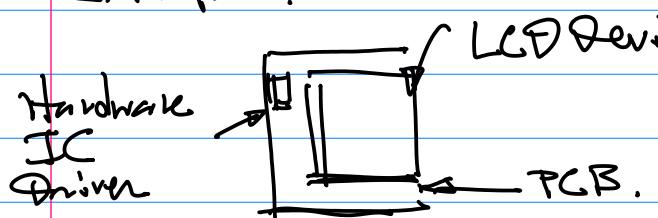


Fig.2

CPU



Host



Slave



IC Driver



LCD Display

Display module

- a. To Drive LCD Display
To Display a pixel } (x,y) Location
- b. To provide feedBack } I(x,y) Intensity, and color
and Interface to CPU module.

(SPI Interface)

To Establish Interface, SPI (Serial Peripheral Interface)

Hardware pins of SPI : 3+1.

Now, Consider the I/F to LCD module.

Ref: github.com/mahili/Cmpe240

Z018S-9-SPILCD...

- MOSI (Master Output
Slave Input)
- MISO (Master Input
Slave Output)
- SCK (SPI Clock)
- SSEL_x (SPI Enable)

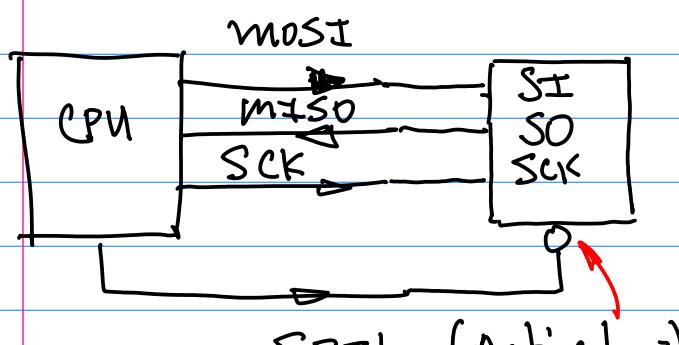


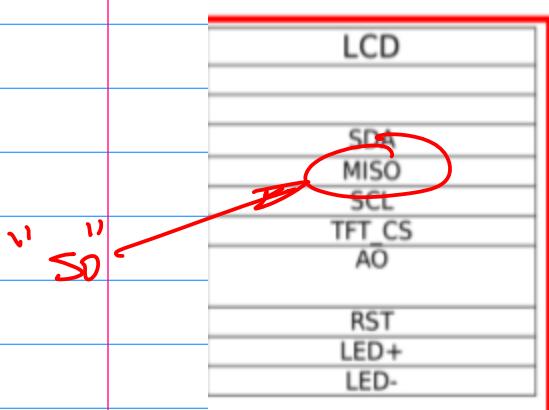
Fig.3.

Note: Mark the Direction of the Signal

Label	Pin
MOSI	P0.18
MISO	P0.17
SCK	P0.15
CS	P0.16
GPIO-DC (Data / command)	P0.21
GPIO-Reset	P0.22
3.3V	
GND	

a Identify all pins on CPU for SPI I/F;

b Identify all pins on LCD for SPI I/F, Correct matching the host/master and slave



Labels from the LCD Display.

SPI pins:

SDA, MOSI,
MISO
:

Type: Change MISO on LCD to SD,
etc.

Note: In addition to SPI interface,
Identify Command / Data Toggle
Control pin, the label should be

C/D Depending on the Signal Level, the Communication
From CPU to LCD is interpreted by LCD either as a Command
or Data.

Now, Software Part

[github/nihalili/Cmpe240](https://github.com/nihalili/Cmpe240)

2018S-10 - DrawLine

Example: Draw A Line Code

1. Color Definition. hex Digits

2 hexs for Each primitive

color

(red, green, blue)

Primitive Colors: r, g, b

2 hex Digits : min. 0

max: 255

Identify module @ Line 285

Parameters $(x_0, y_0), (x_1, y_1)$ and

Color

$\overrightarrow{P_0}$ or $\overrightarrow{P_1}$ or
 $\overrightarrow{P_i}$ or $\overrightarrow{P_{init}}$

Match to Expr.(w) & (l_b)

to Build n Square One
Line at time.

Sept. 29 (Wed)

Project 1. (10 pts) Due Oct. 18th

2 hex \Rightarrow 8 bit \Rightarrow $2^8 = 256$ Requirements:

2° Bit Arrangement for the primitive colors: $R G B = (2 \text{ hex}) (2 \text{ hex}) (2 \text{ hex})$ Board, Programs, Report

1° All work including prototype

CmpE440

However Team work is encouraged.

- 2° Implement Hardware LCD Display. $\stackrel{a}{=} \text{Rotation of sets of Squares}$, $\stackrel{b}{=} \text{Creates trees to forest}$; $\subseteq 3D$ World Coordinate System Visualization;

Submission:

- 1° Formal written Requirements

Rubrics will be posted on Line.

- 2° Submission on CANVAS.

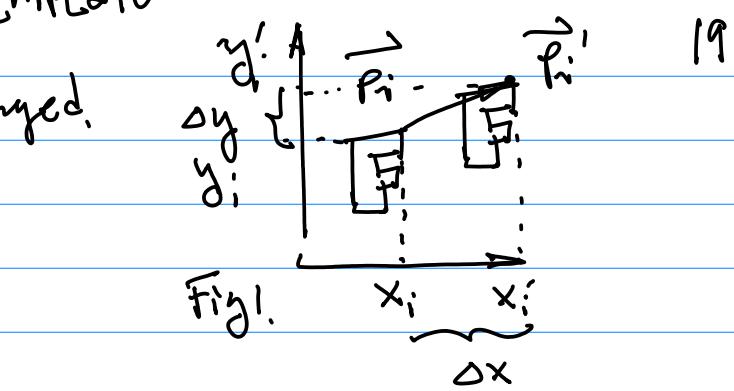
- 3° Source Code/Binary have to Exported Project, in zip.

- 4° Project Report (5 pages) in IEEE format;

- 5° 5 Seconds video

- $\stackrel{a}{=}$ Entire System Setting.
Frost + Prototype Board;
 $\stackrel{b}{=}$ Screen of the Animated Display; \subseteq Show the Prototype Board.

2D Transforms
Mathematical Formulation



Given 2D pattern $\{\vec{P}_i(x_i, y_i) | i=0, 1, \dots, N-1\}$
Establish Translation

Matrix T.

$\vec{P}_i(x_i, y_i)$ Before; $\vec{P}'_i(x'_i, y'_i)$ After

$$x'_i \stackrel{?}{=} x_i + \Delta x \quad \text{After} \quad \text{Before} \quad \dots (1)$$

Similarly

$$y'_i = y_i + \Delta y \quad \dots (2)$$

After Before

$$\begin{pmatrix} x'_i \\ y'_i \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & \Delta x \\ 0 & 1 & \Delta y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} \quad \dots (3)$$

Let's consider Rotation

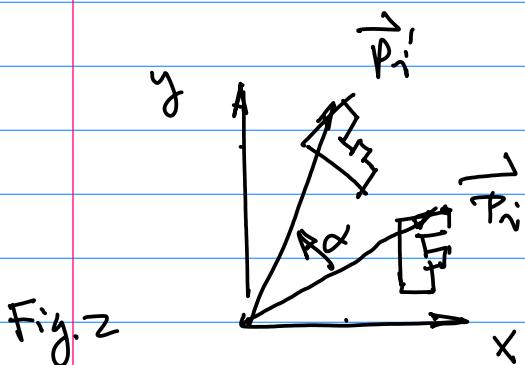


Fig. 2

Note: Counter Clockwise Rotation
"Positive" Angles

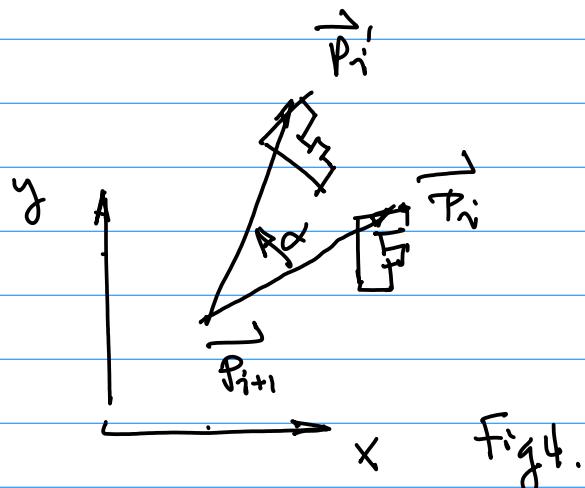


Fig. 4.

After

Before

$$\begin{pmatrix} x'_i \\ y'_i \\ 1 \end{pmatrix} = \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} \quad \dots (4)$$

Note: for Rotations in Fig 4, we will have to conduct

Pre-processing to Translate the reference point P_{i+1} to origin(0,0)

Then, Perform Rotation;

Finally, Post-processing. Translate the rotated Pattern Bank to its Original Location

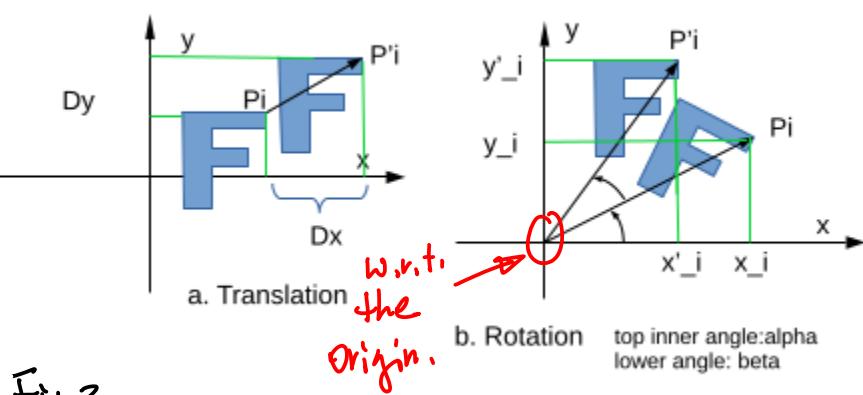


Fig. 3

From Eqn (4)

$$\begin{cases} x'_i = x_i \cos\alpha - y_i \sin\alpha \end{cases} \dots (5a)$$

$$\begin{cases} y'_i = x_i \sin\alpha + y_i \cos\alpha \end{cases} \dots (5b)$$

After

Before

$$\begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} = T^{-1} R T \begin{pmatrix} x'_i \\ y'_i \\ 1 \end{pmatrix} \quad \dots (b)$$

where

$$T^{-1} = \begin{pmatrix} 1 & 0 & -Dx \\ 0 & 1 & -Dy \\ 0 & 0 & 1 \end{pmatrix} \quad \dots (7)$$

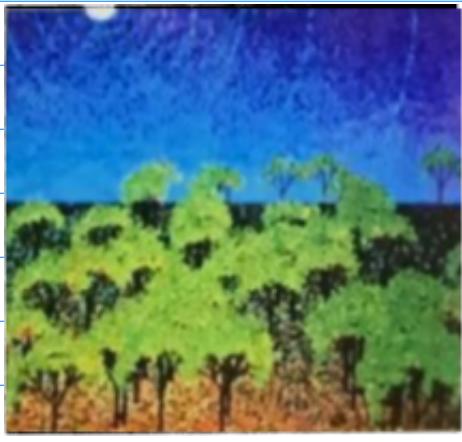


Fig. 5

Example: Use 2D Transforms to Create Trees shown Above

Step 1. Define Initial Points

Points to give a tree trunk.

$$\vec{P}_0(x_0, y_0), \vec{P}_1(x_1, y_1)$$

$$\vec{P}_0(x_0, y_0) = (10, 10), \vec{P}_1(x_1, y_1) = (10, 20)$$

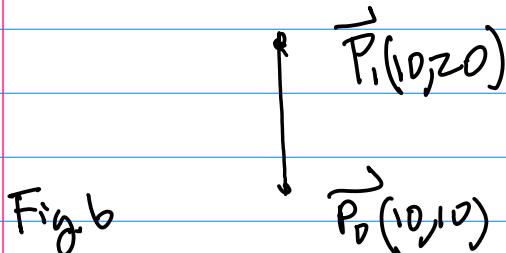


Fig. 6

Step 2. Use Vector Eqn to Create next level major Branch

$\vec{P}_1(x'_1, y'_1)$ as in Fig.

$$\vec{P}_1(x'_1, y'_1) = \vec{P}_0(x_0, y_0) + \lambda (\vec{P}_1(x_1, y_1) - \vec{P}_0(x_0, y_0))$$

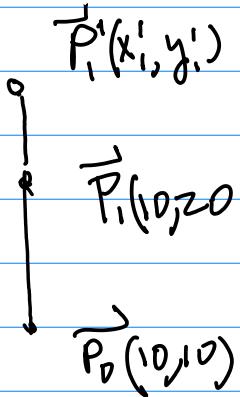


Fig. 7

make $\lambda = 0.8$.

Step 3. Rotation of \vec{P}_1' Counter clockwise to Create Left Branch.

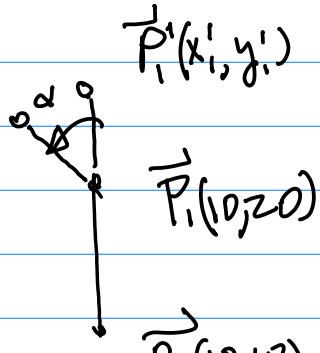


Fig. 8

Oct. 4 (Monday)

Topics : 1° 2D Example for trees
2° Virtual Display vs.
Physical Display, Implementation

Example: Continued from Step 3.
First, Preprocess

Computation

22

$$T = \begin{pmatrix} 1 & 0 & DX \\ 0 & 1 & DY \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & +D \\ 0 & 1 & -2D \\ 0 & 0 & 1 \end{pmatrix}$$

Now, Similarly,

Next Rotation,

$$R = \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad \alpha = 30^\circ.$$

Post-Processing:

$$T^{-1} = \begin{pmatrix} 1 & 0 & -DX \\ 0 & 1 & -DY \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 10 \\ 0 & 1 & 20 \\ 0 & 0 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & +D \\ 0 & 1 & -2D \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 10 \\ 36 \\ 1 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 10 \\ 0 & 1 & 20 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 16 \\ 1 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 10 \\ 0 & 1 & 20 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} -b\sin\alpha \\ b\cos\alpha \\ 1 \end{pmatrix}$$

From Eqn(s) (7):

$$\begin{pmatrix} 1 & 0 & 10 \\ 0 & 1 & 20 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & +D \\ 0 & 1 & -2D \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 10 \\ 20 \\ 1 \end{pmatrix} = \begin{pmatrix} -b\sin\alpha + 10 \\ b\cos\alpha + 20 \\ 1 \end{pmatrix}$$

New X
New Y

$$= \begin{pmatrix} 1 & 0 & 10 \\ 0 & 1 & 20 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

Summary: Put Together Eqn (b) to form the Rotation Algorithm.

$$T^{-1} R T =$$

$$= \begin{pmatrix} 1 & 0 & 10 \\ 0 & 1 & 20 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 10 \\ 20 \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & -DX \\ 0 & 1 & -DY \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & DX \\ 0 & 1 & DY \\ 0 & 0 & 1 \end{pmatrix}$$

Then for $\vec{P}_1(10, 36)$

$$= \begin{pmatrix} 1 & 0 - \Delta X & \cos\theta - \sin\theta & \Delta X \cos\theta - \Delta Y \sin\theta \\ 0 & 1 & -\Delta Y & \sin\theta \cos\theta & \Delta X \sin\theta + \Delta Y \cos\theta \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

Suppose LCD
Resolution is 120×100

$$= \begin{pmatrix} \cos\theta - \sin\theta & \Delta X \cos\theta - \Delta Y \sin\theta - \Delta X \\ \sin\theta \cos\theta & \Delta X \sin\theta + \Delta Y \cos\theta - \Delta Y \\ 0 & 0 \end{pmatrix}$$

Number of Pixels/Row
100: Rows.

Therefore

$$\begin{pmatrix} x'_i \\ y'_i \\ 1 \end{pmatrix} = \begin{pmatrix} \cos\theta - \sin\theta & \Delta X \cos\theta - \Delta Y \sin\theta - \Delta X \\ \sin\theta \cos\theta & \Delta X \sin\theta + \Delta Y \cos\theta - \Delta Y \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix}$$

x : Left To Right $0, 1, 2, \dots, m-1$

y : Top down $0, 1, 2, \dots, N-1$

Limitation:

1° No Negative Value
in the System.

2° Tied to the
physical Device
with Resolution $m \times N$.

3° Not Portable

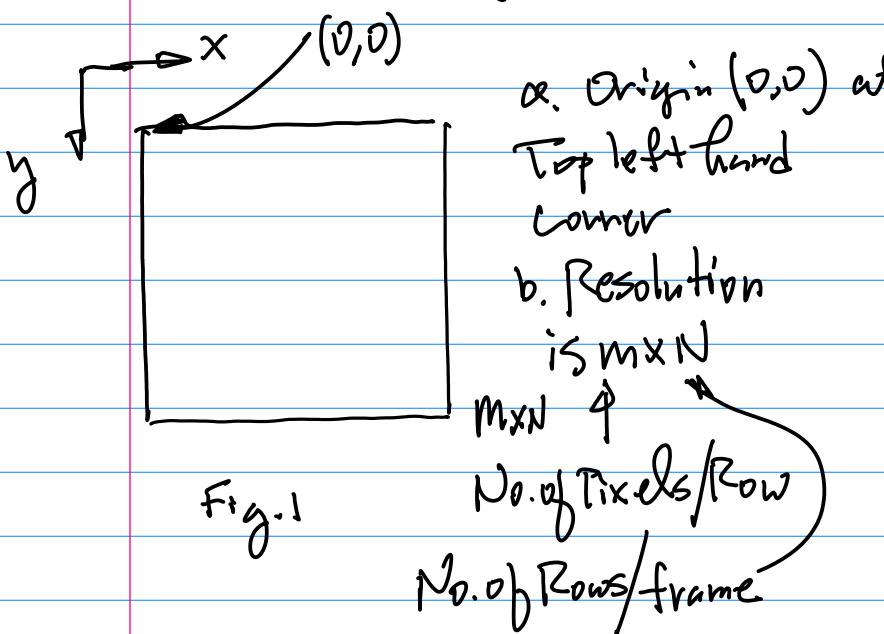
C/C++ Implementation

$$x'_i = \cos\theta \cdot x_i - \sin\theta \cdot y_i + \Delta X \cos\theta - \Delta Y \sin\theta - \Delta X$$

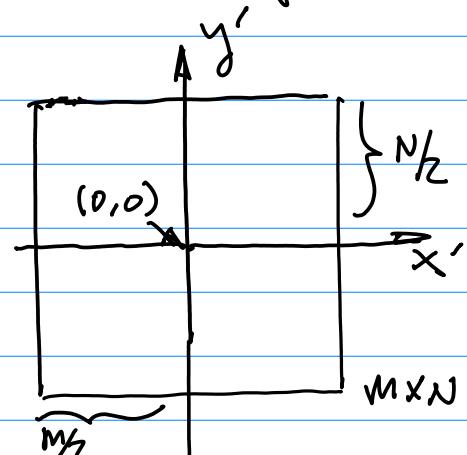
$$y'_i = \sin\theta \cdot x_i + \cos\theta \cdot y_i + \Delta X \sin\theta + \Delta Y \cos\theta - \Delta Y \quad \dots (b*)$$

Physical Display Coordinate

v.s. Virtual Display Coordinate



Now, Virtual Display coordinate



Set $(0,0)$ at the center of the display device.

Transform physical Display to

ComPE240

24

Virtual Display.

$$\begin{cases} x = x' + \frac{m}{2} & \dots (1) \\ y = -y' + \frac{n}{2} & \dots (2) \end{cases}$$

Note: Verify Eqn(1) & (2).

How to use Eqn(1) and (2).

Conduct Computation in Virtual Coordinate (x', y') ,

make sure to Scale the result in $x' \in [-\frac{m}{2}, \frac{m}{2}]$
in Total No. of Col.

$$y' \in [-\frac{N}{2}, \frac{N}{2}], N : \text{Total}$$

No. of Rows.

Then, use Eqn(1) & (2) map to your physical display.

Homework (Due 1 week Oct. 11, Monday) Visit and physical Transform.

1° Write C code to realize Eqn(1) & (2).

2° Prompt the user for input (x, y) value in

Virtual coordinate System,

Then you compute Eqn(1) & (2)
to find physical display
Coordinate, plot (Draw) 5×5

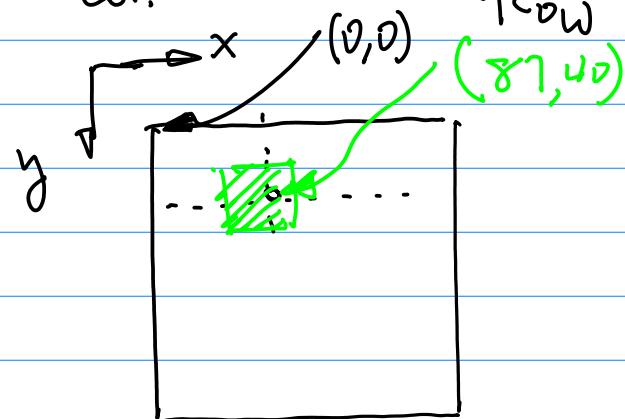
Patch with its center pixel
Equal to the Computation Result.

Example: Computation Result

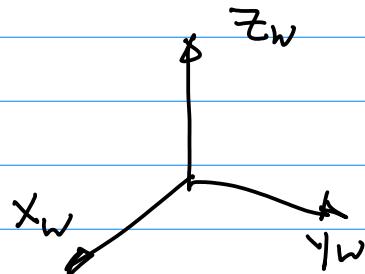
$$(x, y) = (87, 40)$$

Col.

Row



3D Graphics Processing Engine
Introduction: World Coordinate System



Outlook (Wed)

Fig. 1

Topics: 1° Transformation Pipeline

Note:

1° Affection Update

Posted on bit;

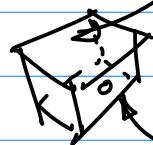
2° Handout on GPIO code,
please read it, understand
the code. It is required.

a "pin-Hole" model.

Diameter of the lens ("Hole") is
very small, $d \ll s$.

Enclose to form a virtual
camera.

Projection Plane
to form an image



pin-Hole

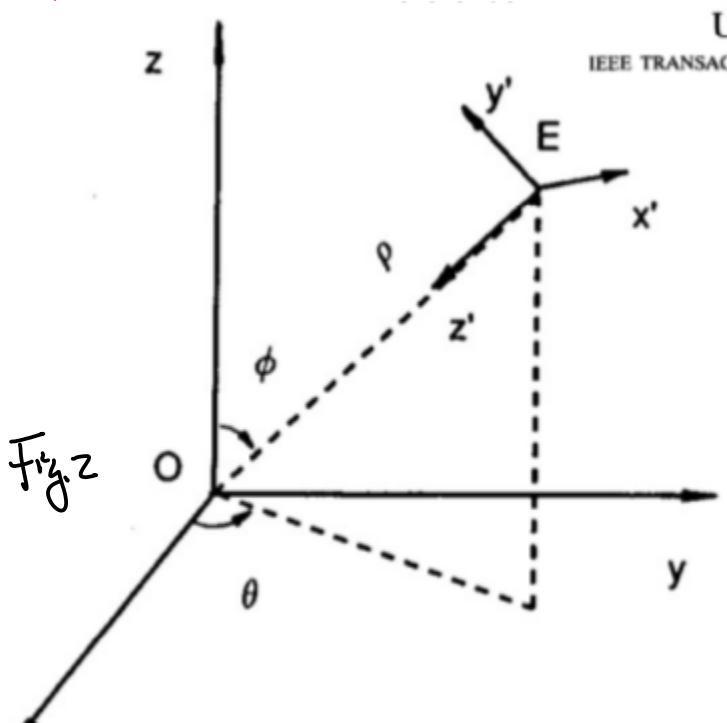
→ ED fixation direction of the
Virtual Camera

Note: This is just ONE formulation
Among other 3 Additional
Possible Formulations

1° Viewer Coordinate System
 $x_e-y_e-z_e$, Sub "e" for
"Eye" / Camera Location;

2° Left Hand System.

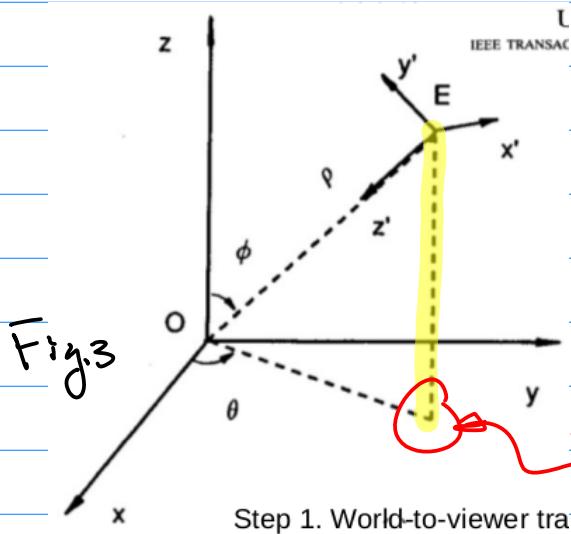
Relationship b/w $x_w-y_w-z_w$ and
 $x_e-y_e-z_e$ Systems to Allow
the Definition of Viewing 3D
Objects in a different
Perspective.



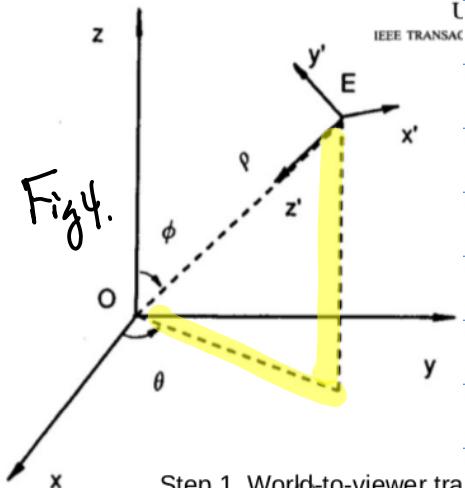
ComPE40

26

Example: Draw A Line, Passes $E(x_e, y_e, z_e)$
Perpendicular to $x_w-y_w-z_w$ Plane



Draw 2nd Line, Passes E' (on x_w-y_w)
to connect to the origin $(0,0,0)$ of
 $x_w-y_w-z_w$. as in Fig



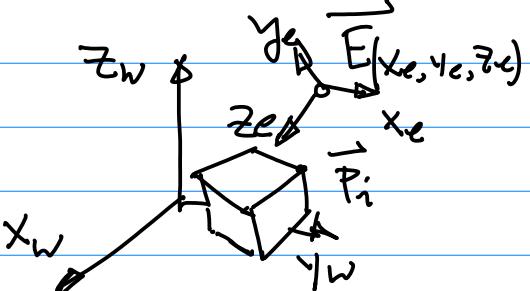
θ : (Theta) formed Between
Angle the 2nd Line
and x_w .

Angle ϕ (phi): Formed
Between \overrightarrow{OE} (Not Exactly
the \overrightarrow{EO}) and z_w

Distance ρ (rho): from $E(x_e, y_e, z_e)$
to the origin $(0,0,0)$:

$$\rho = \sqrt{x_e^2 + y_e^2 + z_e^2} \dots (1)$$

Example: A cube given in
the following figure:



$$T = \begin{bmatrix} -\sin \theta & \cos \theta & 0 & 0 \\ -\cos \phi \cos \theta & -\cos \phi \sin \theta & \sin \phi & 0 \\ -\sin \phi \cos \theta & -\sin \phi \cos \theta & -\cos \phi & \rho \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

\vec{P}_t defined in the $x_w-y_w-z_w$
 \vec{P}_i defined in the $x_e-y_e-z_e$... (2)

$$\begin{pmatrix} x'_i \\ y'_i \\ z'_i \\ 1 \end{pmatrix} = \begin{bmatrix} -\sin \theta & \cos \theta & 0 & 0 \\ -\cos \phi \cos \theta & -\cos \phi \sin \theta & \sin \phi & 0 \\ -\sin \phi \cos \theta & -\sin \phi \cos \theta & -\cos \phi & \rho \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x_i \\ y_i \\ z_i \\ 1 \end{pmatrix} \dots (z_b)$$

$\overrightarrow{P'_i}$ $\overrightarrow{P_i}$

After"in $x_w-y_w-z_w$ "

System Camera

Example: Given A Virtual
Camera E(200,200,200)

Find Transformation Matrix

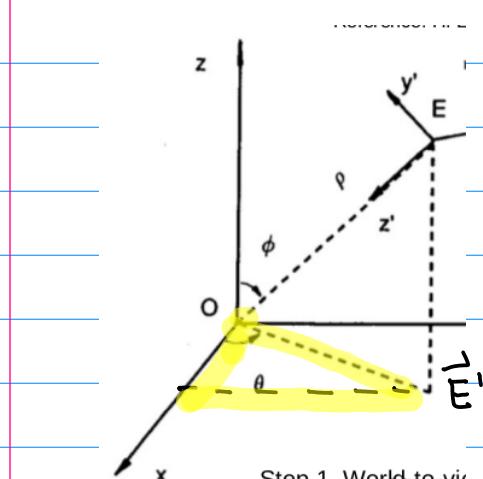
T (World-To-Viewer)

Transform Matrix.

Sol. For θ (theta) Angle

$$\cos \theta = \frac{x_e}{\sqrt{x_e^2 + y_e^2}} = \frac{200}{200\sqrt{2}} = \sqrt{2}/2 \dots (3)$$

(From Fig 4, pp. 2b)



Step 1. World-to-vi

Consider $\sin \phi$

Oct. 11 (monday)

Note: 1° 3 Handout material, C
Program, with 2021F-111x

a) gProg, C ; Architecture →

CPU Block Diagram →
SPRs → CPU Datasheet
Init & Config.

b) Drawing C :

Architecture → Write a
Pixel (location, color/
Intensity) · Vector Line≤ Sep. C ≤ SPI I/F. to Be
discussed;

Example: Suppose a given

Cub with $P_i(100, 100, 100)$,

And Virtual Camera $E(200, 200, 200)$

Find Viewer coordinates of

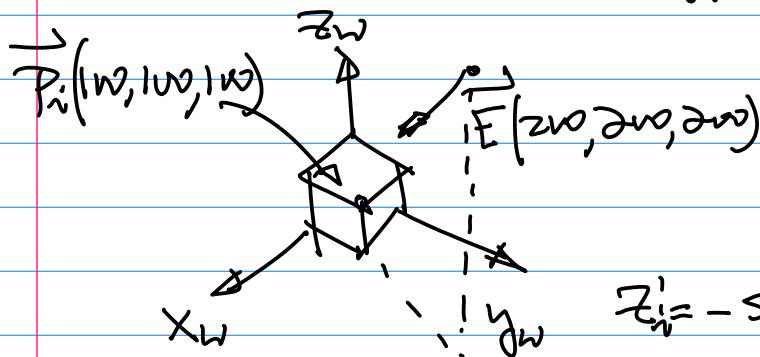
this cub. $P_i' = ?$

$$y_i' = -\frac{\sqrt{3}}{3} \cdot \frac{\sqrt{2}}{2} \cdot 100 - \frac{\sqrt{3}}{3} \cdot \frac{\sqrt{2}}{2} \cdot 100 + \frac{\sqrt{6}}{3} \cdot 100$$

$$\cos \phi = \frac{\sqrt{3}}{3}, \sin \phi = \frac{\sqrt{6}}{3}$$

$$= 100\sqrt{3}/3$$

$$y_i' = -\cos \phi \cos \theta x_i - \cos \phi \sin \theta y_i + \sin \phi z_i$$



$$z_i' = -\sin \phi \cos \theta x_i - \sin \phi \sin \theta y_i - \cos \phi z_i + p$$

From Eqn (2b) pp. 27

$$= -\frac{\sqrt{6}}{3} \cdot \frac{\sqrt{2}}{2} \cdot 100 - \frac{\sqrt{6}}{3} \cdot \frac{\sqrt{2}}{2} \cdot 100 - \frac{\sqrt{3}}{3} \cdot 100 + 200\sqrt{3}$$

$$\begin{pmatrix} x_i' \\ y_i' \\ z_i' \\ 1 \end{pmatrix} = \begin{bmatrix} -\sin \theta & \cos \theta & 0 & 0 \\ -\cos \phi \cos \theta & -\cos \phi \sin \theta & \sin \phi & 0 \\ -\sin \phi \cos \theta & -\sin \phi \cos \theta & -\cos \phi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x_i \\ y_i \\ z_i \\ 1 \end{pmatrix} = \begin{pmatrix} x_i' \\ y_i' \\ z_i' \\ 1 \end{pmatrix} = \left(-\frac{2\sqrt{3}}{3} - \frac{\sqrt{3}}{3}\right) 100 + 200\sqrt{3}$$

$$x_i' = -\sin \theta x_i + \cos \theta y_i = -100\sqrt{3} + 200\sqrt{3} = 100\sqrt{3}$$

$$\begin{cases} y_i' = -\cos \phi \cos \theta x_i - \cos \phi \sin \theta y_i + \sin \phi z_i \\ z_i' = -\sin \phi \cos \theta x_i - \sin \phi \sin \theta y_i - \cos \phi z_i + p \end{cases}$$

$$(x_i', y_i', z_i')$$

$$= (0, 100\sqrt{3}, 100\sqrt{3}),$$

C/C++ Coding Based on Eqn (1).

Optional Homework:

Write C code for Eqn (1).

From the given condition: $\sin \theta = \cos \theta = \frac{\sqrt{2}}{2}$, $x_i = y_i = z_i = 100$

$$x_i' = \frac{\sqrt{2}}{2} \cdot 100 + \frac{\sqrt{2}}{2} \cdot 100 = 0$$

Handout 2021F-11C SSP.C.
Regined (Homeworks, Test)

Handout
SSP.C

```
*****  
* $Id: ssp.c 5804 2010-12-04 00:32:12Z usb00423  
* Project: NXP LPC17xx SSP example  
*  
* Description:
```

Handout 2021F-11Z - ~ Review

2D Vector graphics. to Be posted
On Line. Question 1 & Question 2.

Homework: UV git hub / CANVAS
One week from Today 10/18. Submission
On CANVAS.

Note: Midterm Scheduled
Tentatively on the 1st week of
Nov. However, it will be finalized
Based on the progress in Projects
and Lecture, a Review (15-20
min. in Class), b One week Ahead
Notice.

Format of the lectures.

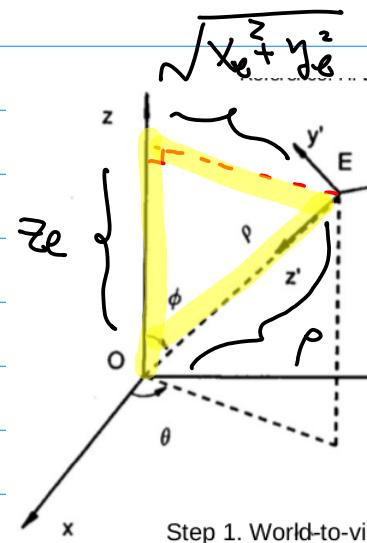
CPU Architectures \rightarrow CPU Dashed

Coding
Implementation

And Prototype Implementation

Today's Topics: Transformation

Pipeline.



Step 1. World-to-vit

Note Draw a line pass the
Vector \vec{E} location. make
sure a perpendicular to \vec{z}_w
b in parallel with the
line on X_w-Y_w plane

Example: Continue from previous
discussion.

$$\begin{aligned} \sin \phi &= \sqrt{x_e^2 + y_e^2} / \rho \\ &= \frac{\sqrt{x_e^2 + y_e^2}}{\sqrt{x_e^2 + y_e^2 + z_e^2}} \quad | \\ &\qquad\qquad\qquad x_e = y_e = z_e = 200 \end{aligned}$$

$$= \frac{200\sqrt{2}}{200\sqrt{3}} = \frac{\sqrt{2}\sqrt{3}}{3} = \sqrt{6}/3$$

$$\cos \phi = \frac{z_e}{\rho} = \frac{200}{200\sqrt{3}} = \sqrt{3}/3$$

2nd Step for Transformation

Pipeline \Rightarrow Perspective

Projection

Assume the point Before $\vec{P}_i(x'_i, y'_i, z'_i)$

After Perspective Project,

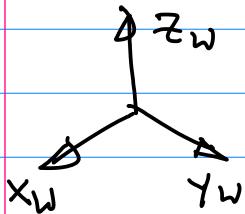
$\vec{P}''_i(x''_i, y''_i)$ (2D pt !)

$$\left\{ \begin{array}{l} x''_i = \frac{D}{z'_i} x'_i \quad \dots \underline{(za)} \\ y''_i = \frac{D}{z'_i} y'_i \quad \dots \underline{(zb)} \end{array} \right.$$

(x''_i, y''_i) on 2D LCD Display
With Depth Perception

D : focal length.

$$D \approx 20-30$$



Draw on 2D
Display.