

Aug 21 (Monday).

Organizational meeting.

1. github.

<https://github.com/hualili/CMPE240-Adv-Microprocessors/tree/master/2018F>

2.

Course and Contact Information

Instructor(s): Harry Li

Office Location: Engineering Building, Room 267A

Telephone: (650) 400-1116

Email: hua.li@sjsu.edu

Office Hours: M.W. 3:00-4:00 pm

IN PERSON.

Class Days/Time: Mondays, Wednesdays, 1:30-2:45 pm

Classroom: Engineering Building, Room 331

Prerequisites: CmpE 180D for non CMPE or non EE undergraduates. Documentation of having satisfied the class prerequisite requirements will be required for students who have been dropped from the class.

3. Emphasis on the Advanced Nature of the Microprocessor Systems. → Embedded Nature, ARM CPU. → GPU: graphics Processing Unit

Architecture of a computing system including system bus, memory subsystems and peripherals. Uni-directional and bidirectional bus architectures, SRAM and FLASH memories and their interfaces with the system bus. Design of Graphics Processing Engines, interrupt controller, transmitter receiver, timers, display adapter, and other system peripherals and bus interfaces.

→ Engine for Deep learning, AI etc.

4. Hands-on.

Datasheets. → Spec. → Hardware

Prototype

Board

With Color

LCD

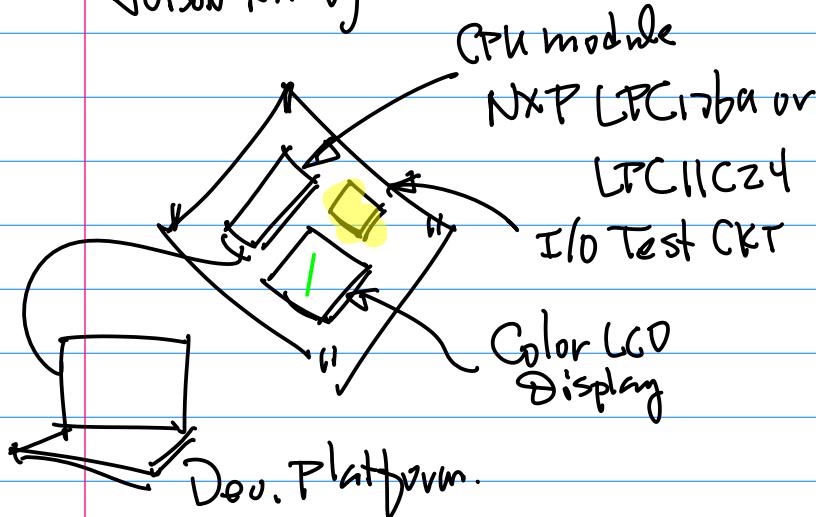
Display

2D
Graphics
Engine



3D Graphics Engine

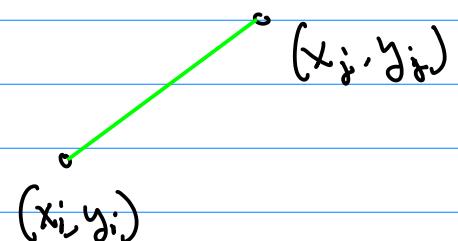
Benchmarking (with Ref. to NVDA
Jetson Nano)



Line Drawing Sample Code.

(x_i, y_i) Starting pt.

(x_j, y_j) Ending pt.



Action Item (Homework, No Submission)

About 22,800,000 results (0.59 seconds)



NXP Semiconductors

<https://www.nxp.com>

NXP Semiconductors: Automotive, IoT & Industrial Solutions

NXP is a global semiconductor company creating solutions that enable secure connections for a smarter world.

Results from nxp.com



Note: Homework/Projects are ~~to be~~
ON CANVAS, with Written
Requirements. These are the
material to be Submitted.

5. PPTs, Lecture Notes (White Board Notes), Datasheet(s), are posted on the github.

Textbook

- NXP LPC17xx datasheets;
- LPC1768/1769 CPU Module schematics;
- Dave Jaggar, ARM Architectural Reference Manual, Prentice Hall, ISBN 0-13-736299-4;

- Reference: ARM11 data sheets and on-line web materials on line <https://github.com/hualili/>, or at the SJSU CANVAS provided copyright permitted;
- (Optional) Nvidia Jetson NANO datasheet and user menu (online from Nvidia developer website);
- (Optional) RISC-V tutorial (the link to be given in the lecture) and FPGA verilog implementation guide (the link to be given in the lecture).

Note: 1° Initial Sample Projects, ~ A Dozen Sample Projects.

<input type="checkbox"/> 2018F	Add files via upload
<input type="checkbox"/> 1769 patch.zip	Add files via upload

GPP (General Purpose Port)

Target CPU

NXP LPC11C24

LPC1769

The code was for LPC1769, But newer samples for GPP, Graphics Display for LPC11C24 were developed and posted on the github.

Next Level of the Sample Code

<input type="checkbox"/> 2018S-10-LCD-DrawLi...	Add files via upload
---	----------------------

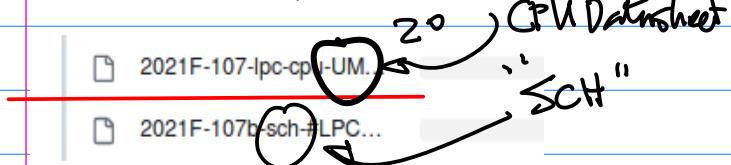
↓ The Lower Layer

Sample Code for
2D & 3D Engine Design.

PPT material in pdf.

will be used in the class.

Datasheet. Note: CPU Datasheet is in CMPE244 folder



Grading Information

Quiz, Homework, Projects	30%
Midterm Examination	30%
Final Examination	40%

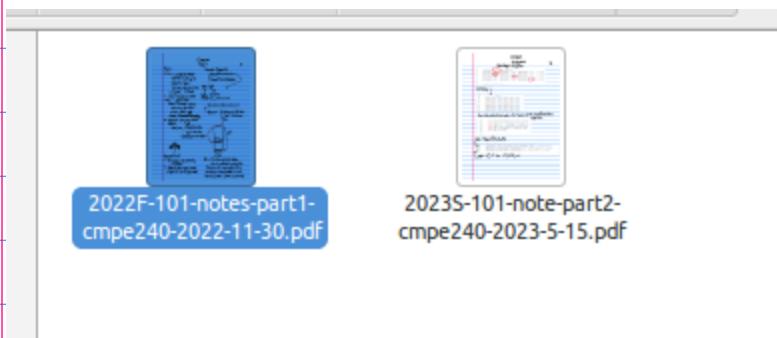
August 23rd (Wed)

Announcement:

1° Lab Space Rm 268

2° CANVAS to be up by
this week.

Introduction.



Example: Architecture of LPC11b9
(LPC11C24)

Can be purchased from
dig-Key.com or
Mouser Electronics



NXP Semiconductors

<https://www.nxp.com/general-purpose-mcus/lpc11...>

Scalable Entry Level 32-bit Microcontroller (MCU) based ...

The LPC11Cxx MCU family is designed for 8/16-bit micro-controller operations,



CMPE240
Fall 2023

51

RESTORE SESSION

https://www.digikey.com/en/products/detail/nxp-usa-inc/0/OM13093UL

DigiKey All Products Enter keyword or part #

Product Index > Development Boards, Kits, Programmers > Evaluation Boards > Embedded MCU, DSP Evaluation Boards > NXP USA Inc. OM13093UL

OM13093UL

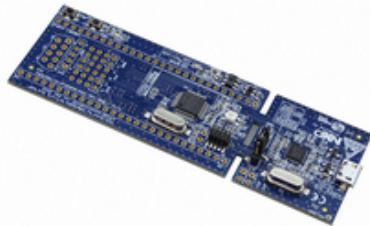


Image shown is a representation only. Exact specifications should be obtained from the product data sheet.

Digi-Key Part Number	568-14402-ND
Manufacturer	NXP USA Inc.
Manufacturer Product Number	OM13093UL
Description	LPCXPRESSO LPC11C24 EVAL BRD
Manufacturer Standard Lead Time	16 Weeks
Detailed Description	LPC11C24 LPCXpresso™ LPC11C00 ARM® Cortex®-N Evaluation Board
Customer Reference	Customer Reference

Note: Please Start the Purchasing Process.
Note: CPU Datasheet.



2018

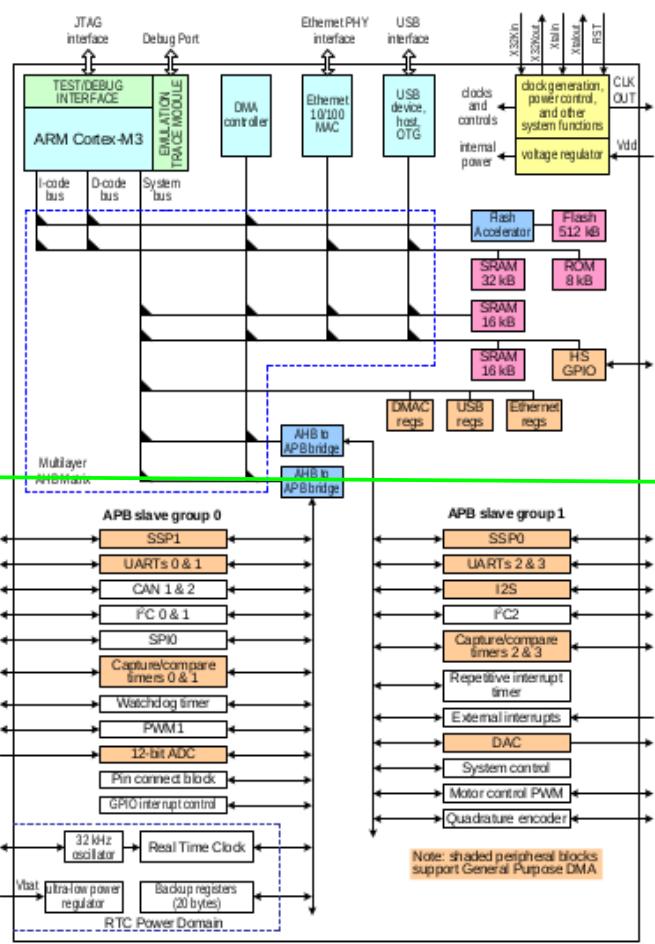


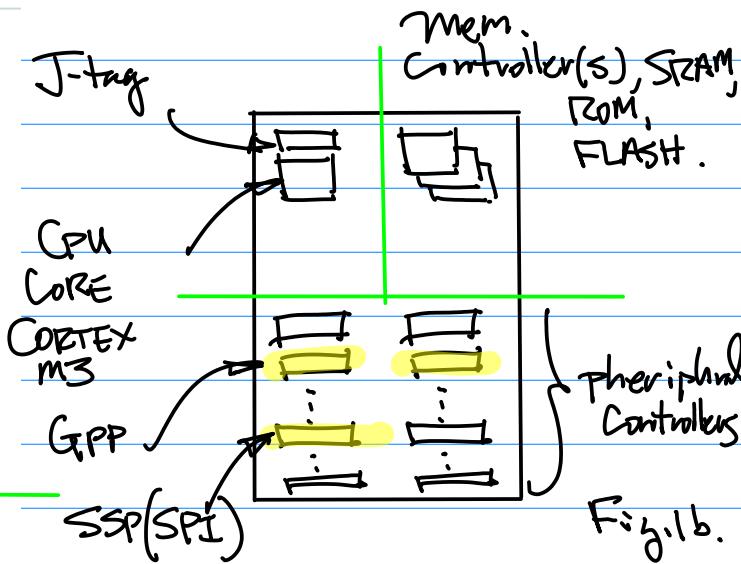
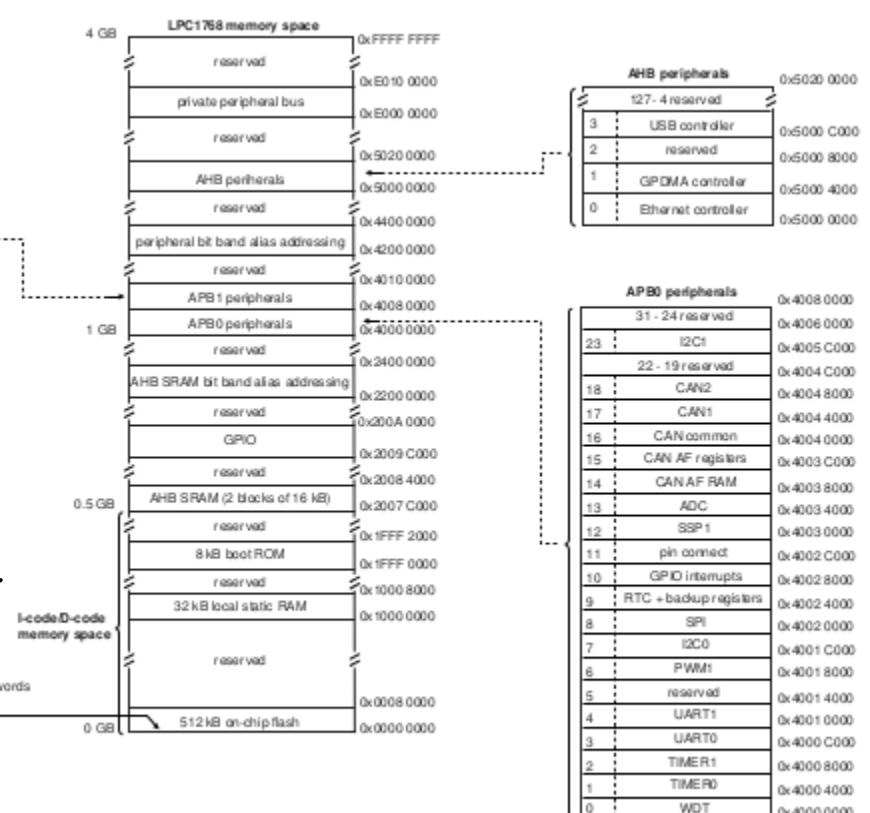
Fig. 1.c

Memory Map.

APB1 peripherals	
0x40010000	31 : system control
0x400FC000	30 - 16 reserved
0x400C0000	15 : QEI
0x400B C000	14 : motor control PWM
0x400B 8000	13 : reserved
0x400B 4000	12 : repetitive interrupt timer
0x400B 0000	11 : reserved
0x400A C000	10 : I2S
0x400A 8000	9 : reserved
0x400A 4000	8 : I2C2
0x400A 0000	7 : UART3
0x4009 C000	6 : UART2
0x4009 8000	5 : Timer 3
0x4009 4000	4 : Timer 2
0x4009 0000	3 : DAC
0x4008 C000	2 : SSP0
0x4008 8000	1 - 0 reserved

Note: "17xx.h"
Porting/Mapping
in C/C++ Code.

0x0000 0400
active interrupt vectors
0x0000 0000 + 256 words



GPP/GPIO : General Purpose Port
or General Purpose I/O

S.P.I. (Serial Peripheral Interface)

Note: One of the GPPs supports Ex INT. (External Interrupt).

Note: For the memory map discussion:

1° RISC : Reduced Instruction

Set Computer.
ARM.
MIPS (Golden Rules:
Uniformity,
Regularity,
Orthogonality)

3° Byte Addressable
Machine.

A smallest memory cell
with an unique address
is a Single Byte.

40

2° 32bit RISC Processor \rightarrow 32bit

Architecture

32bit Addr. Bus.

32bit Data Bus.

32bit R.F. (RegisterFile)

{ GPRs (General Purpose
Registers) 32 bits.
SPRs (Special Purpose
Registers)

3 Cols.

32bit memory map.

$$2^{32} = 2^{10} \cdot 2^{10} \cdot 2^{10} \cdot 2^2$$

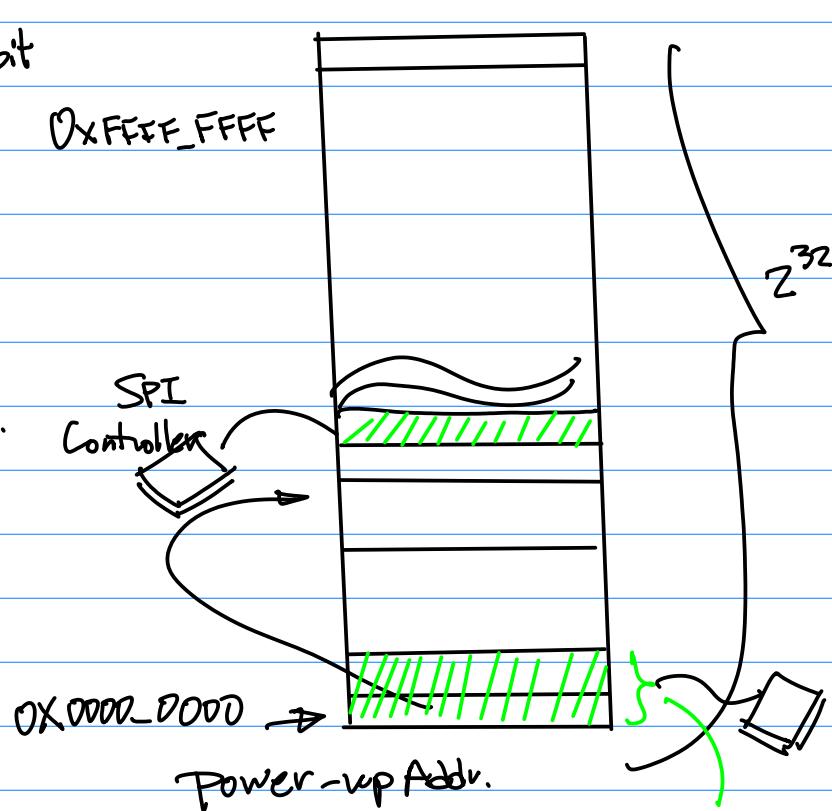
$1K$
(1024)

\sim

$1M$
($1K \times 1K$)

\sim

$1G$



August 28 (Monday).

Note: 1° CANVAS is up.

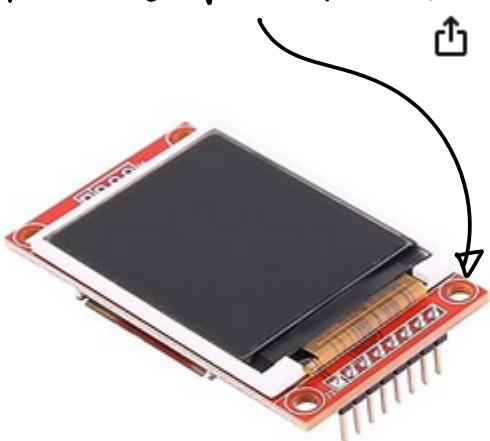
2° CPU module \nparallel LCD module

ST7735 Controller
SPI - Interface

4G? Byte!

cs > Computers & Accessories > Tablet Replacement Parts > LCD Displays

Note: 8-pins D₂ to 10 pins module are OK for the Implementation



1.8 inch SPI TFT LCD Display

Module for ST7735 128x160

51/AVR/STM32/ARM 8/16 bit

Visit the Walfront Store

4.0 ★★★★☆ 42 ratings

Note: ST7725 or
ST7735.

\$10⁹⁹

With Amazon Business, you would have saved \$85.08 in the last year. [Create a free account](#) and save up to 5% today.

Brand Walfront

Personal All in One

computer
design type

Operating Linux

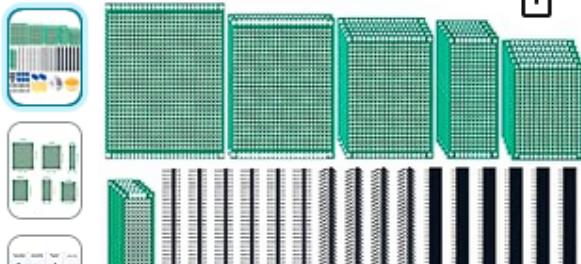
Roll over image to zoom in

3° Bill of Material (BoM) for
this Class:

- 1° CPU Module
- 2° LCD Module
- 3° Wire Wrapping Board.

4" x 3" Through-Holes with metal plating (Just to Cover the Holes, Not the Entire Board)

OR your choice



Miuzei PCB Board Prototype Kit for Electronic Projects, Circuit Solder Double-Side Board with 40 Pin 2.54 mm Male to Female Headers Connector, 2P&3P Screw

Example: Memory Map.

Divide the mem. map into

8 Equal Banks.

			a ₃₁ a ₃₀ a ₂₉ Starting Addr.
BANK 0	First	000	0000:0000:...:0000 → 0x0
BANK 1	2nd	001	0010: ... → 0x2000_0000
BANK 2	3rd	010	0100: .. → 0x4000_0000
:	:	:	
BANK 7	8th	111	
			LSB
		a ₃₁ a ₃₀ ...	a ₂ a ₁ a ₀
		MSB	32 bits Addr. Bus

Note: Important, to Be used
in the Design Process.

Note: "LittleEndian" Convention

Choose a₃₁a₃₀a₂₉ to Identify the
Memory Bank.

a₃₁a₃₀a₂₉ : 0000 : ... : 0000
Lowest Add.

a₃₁a₃₀a₂₉ : 1111 : ... : 1111
Highest Add.

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

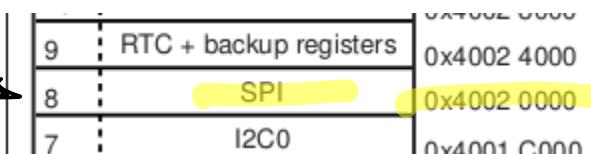
512 | meg.

Example: Identify One of the SPI
Peripheral Controllers By
mem. map.

Memory Bank with a
Starting Addr.

0x4000_0000 →
3rd BANK (BANK 0, BANK 1
BANK 2)

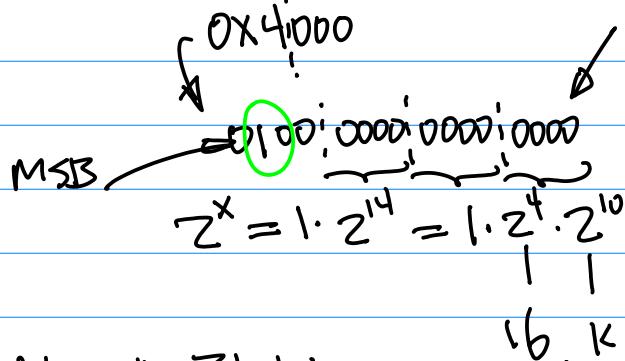
Find A SPI Block



SPI Peripheral Controller
is Located at 0x4002_0000

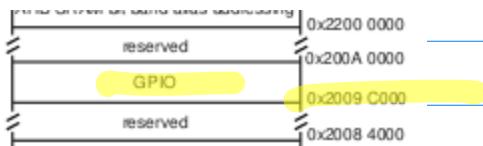
Question: How Big is the memory
Block for SPI Controller?

$0x4002_4000$
 $- 0x4002_0000$



Note: This Block of memory is employed for a set of SPR's (Special Purpose Registers) to perform SPI function.

Example: GPP (General Purpose Port)



GPP (Peripheral Controller)

Mem. map location & its Block size

SPRs (Special Purpose Registers)

Responsible for Init & Config.

&

Control Register.



32 bit SPR.

Naming Convention "3+3" for All if Possible Discussion.

LSB

Prefix 3 letters + Root 3 letters

August 30 (Wed)

Note: 1^o CANVAS has been updated with Homework One ON Friday, Opt. Honesty Pledge Signed Form to Be Submitted ON CANVAS.

2^o Bill of Material.

Ref: ON the github of CMPE240
2018S-2 - ... Bill-of-Material

3^o Homework (0 pt) Due Sept. 10 (Sun)

Installation of NXP MCUXpresso.

Submission: Screen Capture that Shows the MCUXpresso + Personal Identifier.

Ref: github. → PPT for MCUXpresso Configuration.

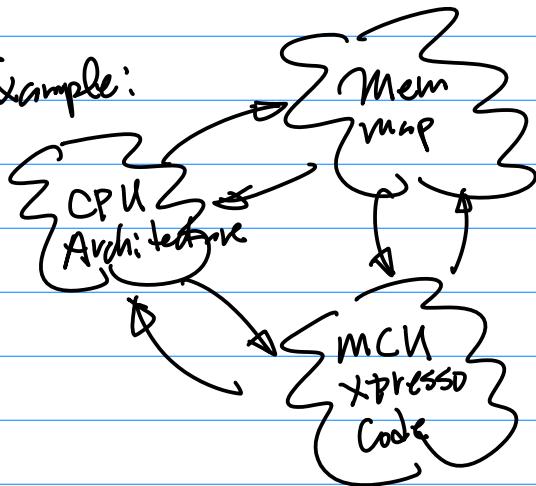
{ NXP Developer Forum, pdf

Note:

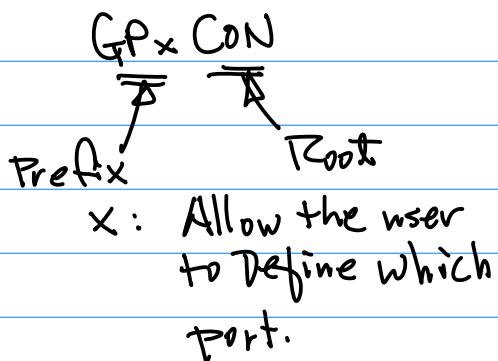
gcc/g++ → Porting to Compiler the Target CPU, NXP Board
Open Source ARM CPU Core LPC family
Cortex M3

4^o Please bring the CPU module to the class for inspection & discussion.

Example:

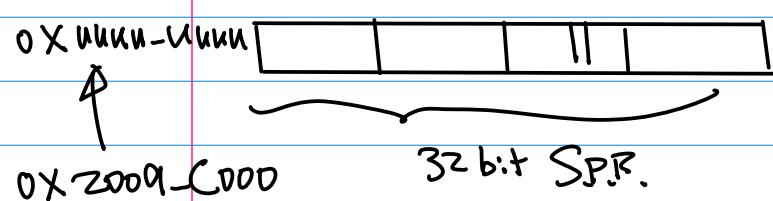


Design of GPP



Note: Multiple I/O pins possible for each GPx

GPACON



GPAPortPin5
as an Output pin → Need to
Identify the
Bit(s) in GPACON
for the selection
Purpose.

Sept. 6 (Wed)

Example: Inspection of LPC11C24 OR
LPC1769.

Purpose: Identify Pin 1 on the module.

Match it up to Schematic of the Board module.

Ref: On the github LPC11C24 AND LPC1769.

Note: 1° Physical pin assignment,
e.g. Pin 1, Pin 2, ..., etc.

2° Nameings of the pins

- a. Connector Related
- b. CPLD Datasheet Related (C/C++ IDE, Code)
- c. Functionality Related.

Use All of the above in your Connectivity Table

3° Power Pins

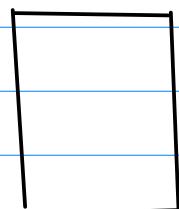
- a. V_{IN}, V_{out} pins
- b. GND pins

Common GND

4° SPI (Serial Peripheral Interface).

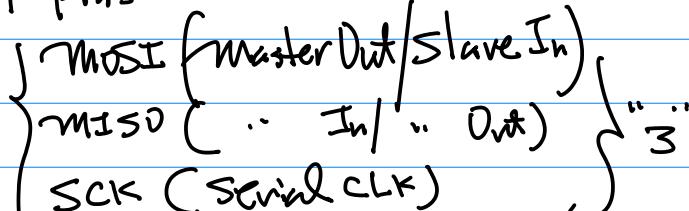


Master



"Slave"

"3+1" pins



EN/nEN Enable Active high
0V Active Low.

J6 or J2, 40 pins
connector.

Header Connector → male

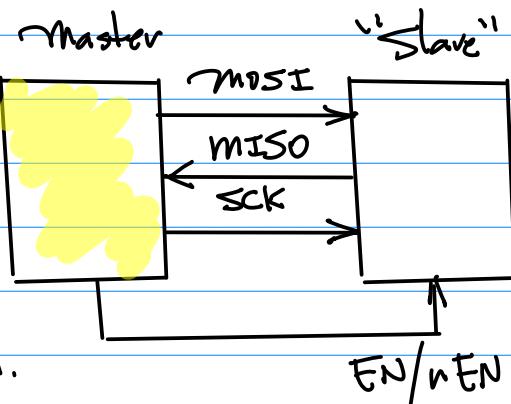


Fig.1.

Connector Header Through Hole 16 position 0.100" (2.54mm)

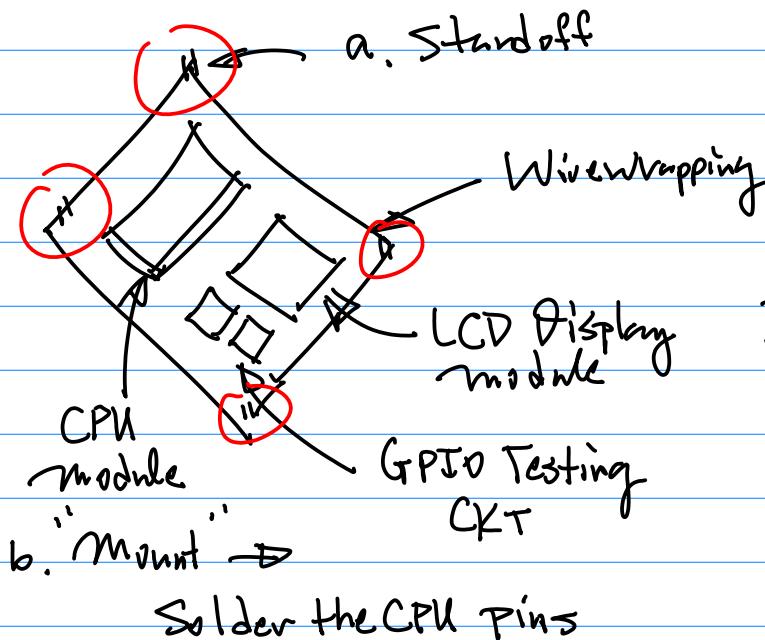
Soldering the top left corner pins
(2~3 pins), And the bottom pin(s)
(1~2 pins).

c. Soldering the LCD module.

Note: 1° Bring your Prototype
Board with the CPU
Mounted On the Board.

Example: SPR for SSPd.

Ref: [github/fivalili/Cmpe240](https://github.com/fivalili/Cmpe240)



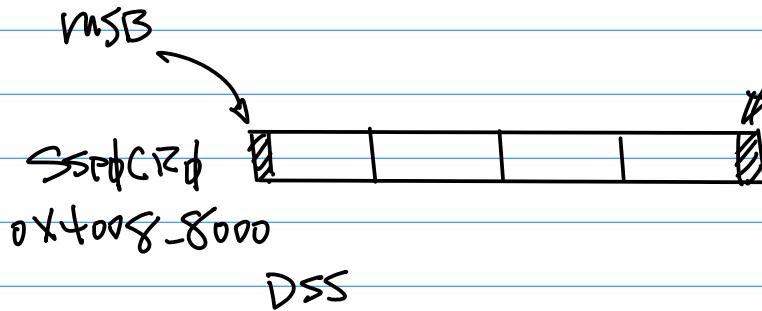
PP131. a. 32bit SPR.

b. Name: SSPdCRd

for the Peripheral controllerd.
(multiple controllers)

Prefix + Root





$SSPdCRd[3:0] = 0111$ for 8 bits Transfer

$SSPdCRd[5:4] = 00$ for SPI.

$SSPdCRd[6] = 0$, $SSPdCRd[7] = 0$ By default

$SSPdCRd[5:8]$ SCR . Serial Clock Rate

CR0 → 8 bit $\Rightarrow 2^8 = 256 \Rightarrow [0, 255]$

SysCLK (System Clock)

CPU Clock.

Range to work with
to define Serial Clock.

PCLK (peripheral clock)

$\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \dots$

SCK.

SCR Controls SPI Clock.

$$f_{SPI} = \frac{PCLK}{(PSDIVSR * (SCR + 1))}$$

... (1)

from a SPR

$[2, 255]$

$[0, 255]$

?
254

Homework Due 1 week
from Today. Sept. 20 (11:59 pm)

a) Build / Mount
CPU module and SPI
LCD Display module
On the Prototype Board,
Solder them on the
Board.)

b) Take a photo of your
Prototype System, and
Submit the photo with
Caption on it, with
your SID, Name.
Submission on
CANVAS.

* Bring your Prototype
Board to the class.

Example: Suppose we define

① PCLK = 20 MHz;

② $f_{SPI} = 5\text{ KHz}$

(e.g. $\rightarrow 5\text{ Kbps}$)

③ Design By Assigning
SCR to Realize the

Bit Rate Requirement.

Sept. 13 (Wed)

Note 1. Inspection of the prototype
Board (Work-In-Progress)

Sol: From Eqn (1), Pg 431.

Hence, Our Design provides the following value for the Required f_{SPI} .

$$f_{SPI} = \frac{PCLK}{CPSDVSR * (SCR+1)} \dots (1)$$

from the given Condition, we have

$$CPSDVSR = 32 \text{ and}$$

$$SCR = 124$$

$$5 \times 10^3 = \frac{20 \times 10^6}{CPSDVSR * (SCR+1)}$$

Design By Iteration.

First, Let $CPSDVSR = 4$
Solve for SCR

$$CPSDVSR * (SCR+1) = \frac{20 \times 10^6}{4 \times 10^3}$$

$$CPSDVSR = 4$$

$$SCR+1 = \frac{4 \times 10^3}{4}$$

$$SCR = 1 \times 10^3 - 1 = 999 > 255$$

So, the Next Iteration of the Design

Let $CPSDVSR = 32$

from Eqn (1), we have

$$CPSDVSR * (SCR+1) = 4 \times 10^3$$

$$CPSDVSR = 32$$

$$SCR+1 = \frac{4 \times 10^3}{32}, SCR = \frac{4 \times 10^3}{32} - 1 = 124 < 255$$

CmPE24D

F2023

IS

C-Code / MCUXpresso.

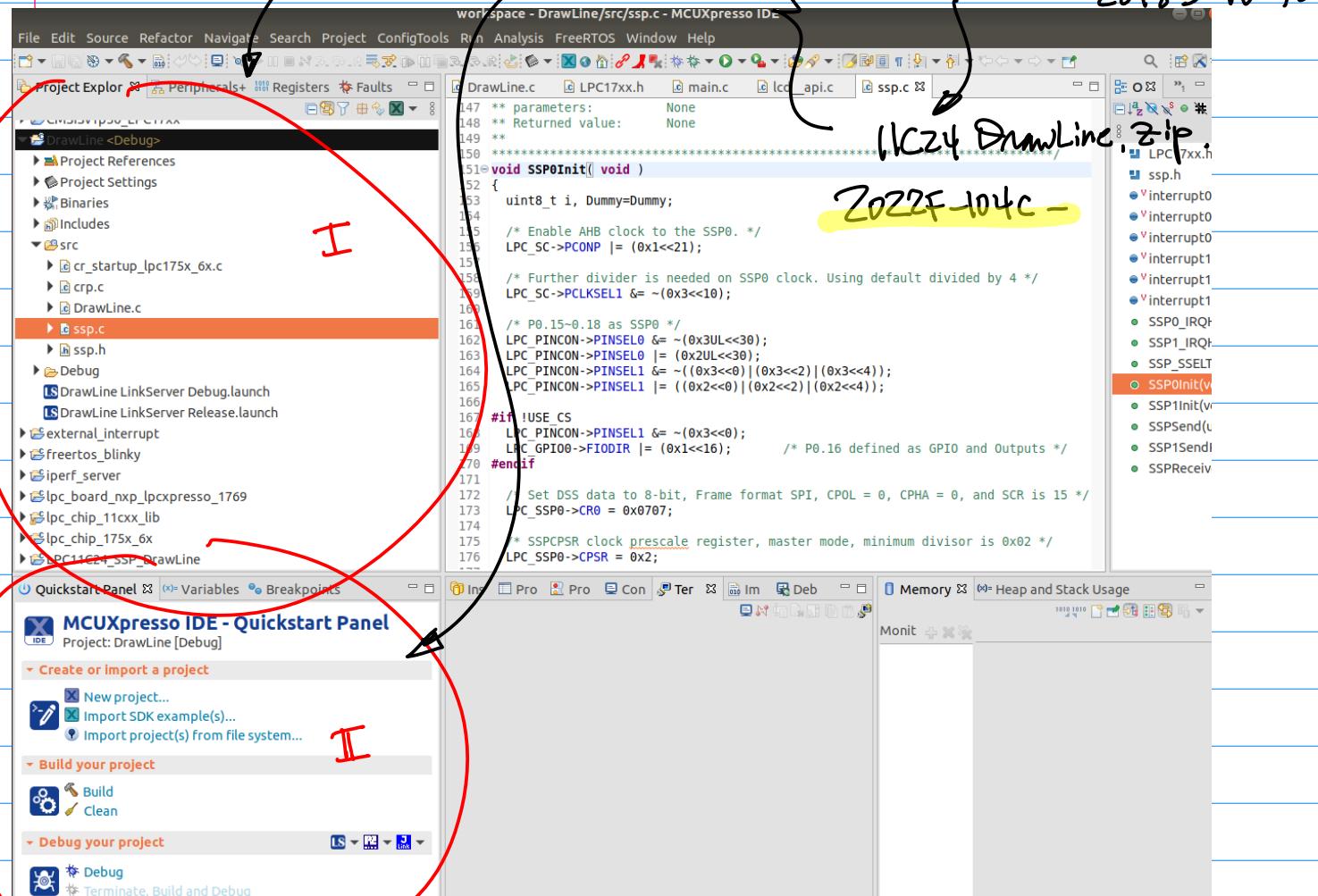
Note1. Projects Imported into the
MCUXpresso .(See I)

Note2: Import 17ba.zip.(from the
class git) By using II.

(GPIO project
as Ref.)

Notes:

DrawLine.zip. 2018S-10-n



Step1. Config the → Step2 → Step3. → Step4

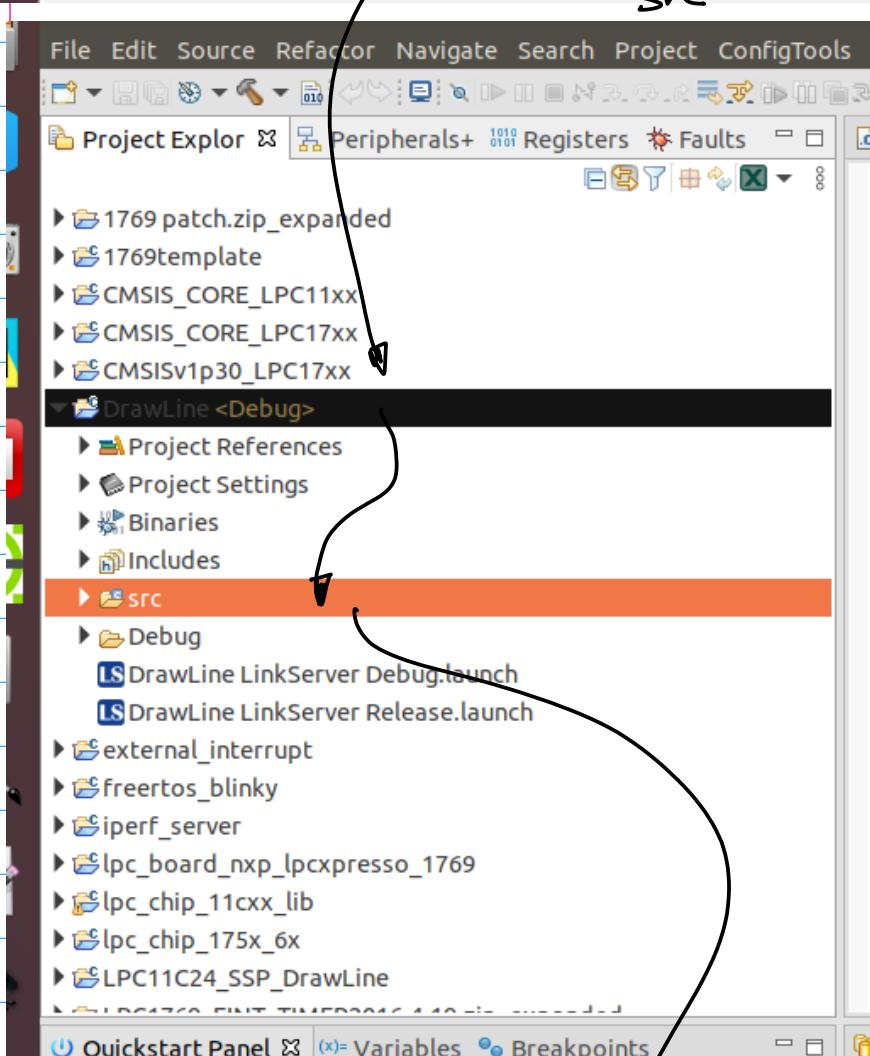
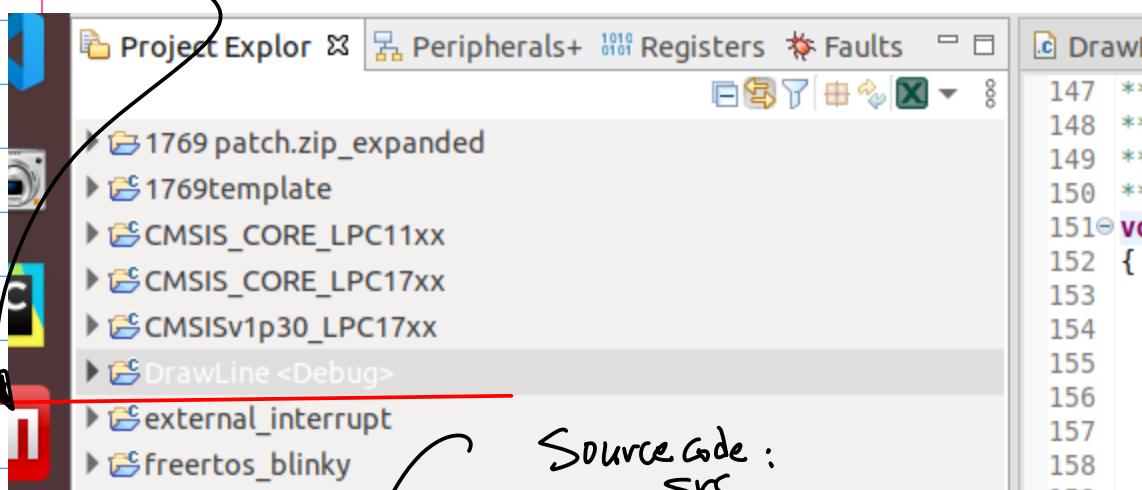
MCUXpresso. Import Import Import Import
CPU LPC17ba. Drawline Drawline Drawline (x_i, y_i)
LBoard Patch. Project(17ba) Project Project (x_i, y_i)
Ref: Class PPT "zip". (x_i, y_i)
OR NXP

CMPE24D

F2023

16

Note 7. Sample Project for I7bg. A Starting Point.



This is from SSP.C

```

148  ** Returned value:      None
149  **
150 ****
151 void SSP0Init( void )
152 {
153     uint8_t i, Dummy=Dummy;
154
155     /* Enable AHB clock to the SSP0. */
156     LPC_SC->PCONP |= (0x1<<21);
157
158     /* Further divider is needed on SSP0 clock. Using default divided by 4 */
159     LPC_SC->PCLKSEL1 &= ~(0x3<<10);
160
161     /* P0.15~0.18 as SSP0 */
162     LPC_PINCON->PINSEL0 &= ~(0x3UL<<30);
163     LPC_PINCON->PINSEL0 |= (0x2UL<<30);
164     LPC_PINCON->PINSEL1 &= ~((0x3<<0)|(0x3<<2)|(0x3<<4));
165     LPC_PINCON->PINSEL1 |= ((0x2<<0)|(0x2<<2)|(0x2<<4));
166
167     #if !USE_CS
168         LPC_PINCON->PINSEL1 &= ~(0x3<<0);
169         LPC_GPIO0->FIODIR |= (0x1<<16); /* P0.16 defined as GPIO and Outputs */
170     #endif
171
172     /* Set DSS data to 8-bit, Frame format SPI, CPOL = 0, CPHA = 0, and SCR is 15 */
173     LPC_SSP0->CR0 = 0x0707;
174
175     /* SSPCPSR clock prescale register, master mode, minimum divisor is 0x02 */
176     LPC_SSP0->CPSR = 0x2;

```

SPR. Init²
Config

Note: please Read this code!

Note: Naming: (Product) + (Peripheral Family) + (Controller) + (SPR)

Note: Code \rightarrow Tech. Spec.

$0x0707 \rightarrow$ Datasheet.
0000;0111;0000;0111

Sept.18 (Monday).

Please Check CANVAS
for the Homework (Prototype Board).

Example: LCD Display Pin Connectivity.

Ref: [github/finalili/Cmpe240/](https://github.com/finalili/Cmpe240/)
2022f-103f - ~

Note:

Toggle Between the Commands and Data

LPC11C24 Connectivity Table t

Fig.1

HL (2022-9-30) corrected this typo by replacing LPC 1769 to LPC11C24

LPC11C24	Description	LCD
1. J6-28	3VOUT	VCC
2. J6-1	GND	GND
3. J6-8	SSEL0	TFT_CS
4. J6-14	RST	RESET
5. J6-13	D/C	AO
6. J6-5	MOSI0	SDA
7. J6-7	SCK0	SCK
8. J6-28	3VOUT	LED

a) 8 bit mode
b) SPI Interface
c) Clock Setting is default.
d) SCR (Eqn.-1)
 $f_{SPI} = \frac{PCLK}{DVSF(SCR+1)}$
e) SCR = 7.



Brand: All in One
Personal computer design type: All in One
Operating System: Linux

find f_{SPI} , Hz

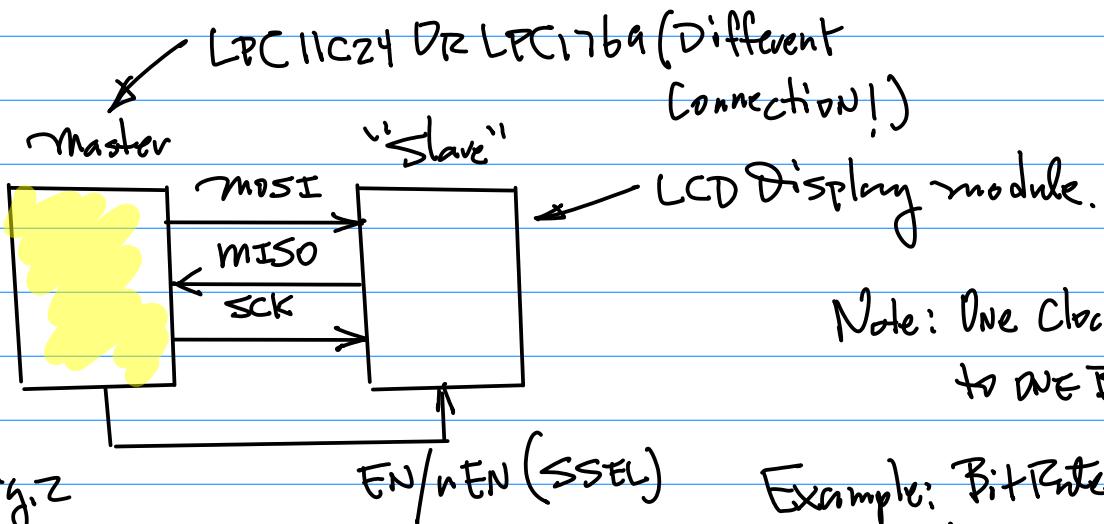


Fig.2

EN/nEN (SSel)

Example: BitRate Calculation.

From PP.12

Note: Use this table as a reference

Build a Complete Connectivity table.

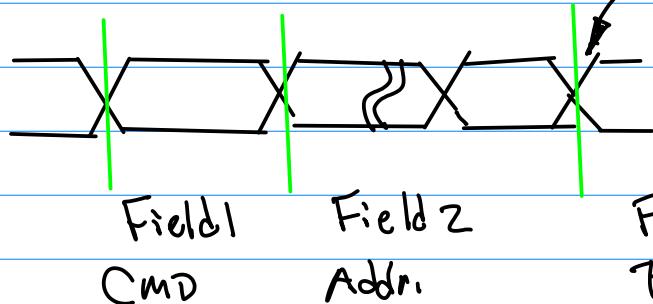
- a) CPU pins;
- b) Connector pins;
- c) Function Name (MOSI, etc).

Example: Continuation on `SSP1init()`
(line 151).

Note: SPI Data Communication.

Waveform Captured By Logic Analyzer, from MOSI pin.

Fig.3



Timing Info.
No. of Pixels per Row.
No. of Rows.

Assuming the Resolution of the LCD
is $M \times N$ (160×120).

Frame Rate (FPS) 30

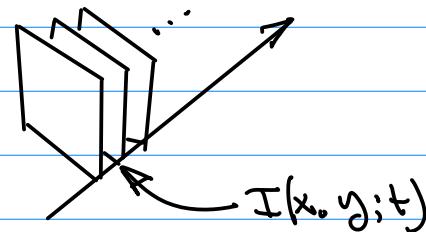
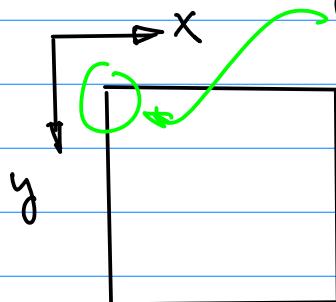


Fig.4

Given Graphics Display Resolution
 $M \times N$.
Physical coordinate
(0,0)



$M \times N$
Col. Row

Find the Bit Rate for SPI Interface.

$$160 \times 120 \times 24 \times 30 = \begin{pmatrix} 1 \text{ Second} \\ \text{Total Bits} \end{pmatrix}$$

Total No. of pixels per frame Bit/Pixel FPS pixel Depth.

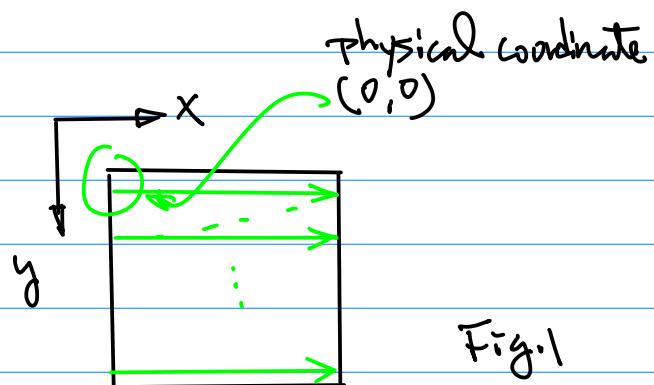


Fig.1

$$160 \times 120 \times 24 \times 30 \stackrel{Z^x}{=}$$

$$\begin{array}{cccc} | & | & | & | \\ Z^7 & Z^7 & Z^5 & Z^5 \end{array} = Z^{14} \cdot Z^{10} = Z^4 \cdot Z^10 \cdot Z^{10} =$$

160

$\frac{1}{16}$

$$\begin{array}{cc} | & | \\ 16 & 1K \end{array}$$

1 meg

$$\therefore 1b \text{ Mbps}$$

Therefore, the SPI Bit Rate is adequate.

Progressive Scanning
2° Virtual Coordinate System.

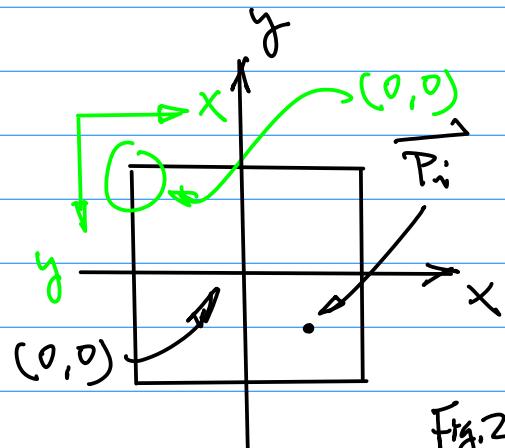


Fig.2

Sept. 20 (Wed).

Example: Discussion on 2D G.E. Design.

Definitions and Notations.

1° Physical Coordinate System.

3° A Picture element, e.g., a pixel, is denoted as

$$\overrightarrow{P_i(x_i, y_i)} \rightarrow \overrightarrow{P_i} \rightarrow (x_i, y_i)$$

... (1)

4° A Line Segment
Starting point $\overrightarrow{P_i(x_i, y_i)}$

Ending pt : $\vec{P}_{i+1}(x_{i+1}, y_{i+1})$

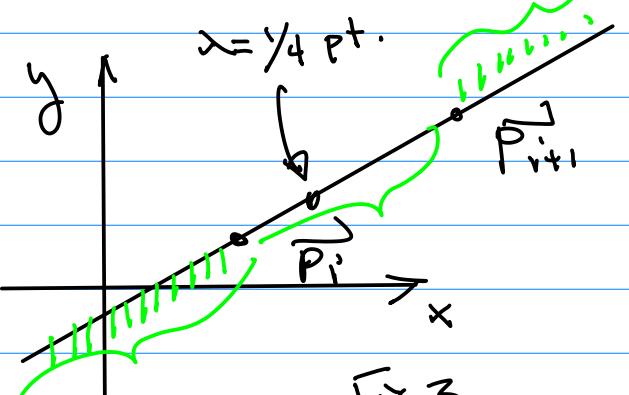
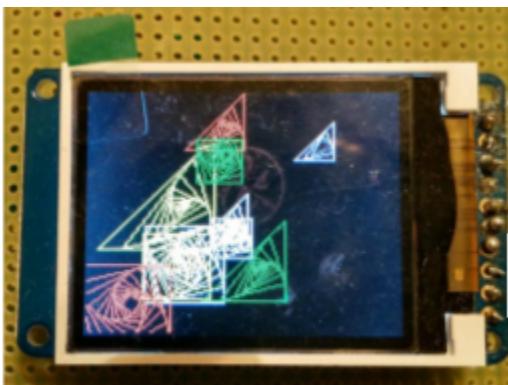


Fig.3



Let's Define the Line in Fig.3.

First, Define a Directional Vector.

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

$$= \vec{P}_{i+1} - \vec{P}_i$$

$$= (x_{i+1}, y_{i+1}) - (x_i, y_i)$$

$$= (x_{i+1} - x_i, y_{i+1} - y_i)$$

... (2)

Line Equation

$$\vec{P}(x, y) = \vec{P}_i(x_i, y_i) + \lambda \vec{d}(x, y)$$

$$= \vec{P}_i(x_i, y_i) + \lambda (\vec{P}_{i+1}(x_{i+1}, y_{i+1}) - \vec{P}_i(x_i, y_i)) \quad \dots (3)$$



When $\lambda = 0$, $\vec{P}(x, y) = \vec{P}_i(x_i, y_i)$ starting pt.

" $\lambda = 1$, $\vec{P}(x, y) = \vec{P}_{i+1}(x_{i+1}, y_{i+1})$ Ending pt.

" $0 \leq \lambda \leq 1$, or $\lambda \in [0, 1]$, Line Segment Between \vec{P}_i and \vec{P}_{i+1}

When $\lambda > 1$, $\vec{P}(x, y)$ points

Beyond $\vec{P}_{i+1}(x_{i+1}, y_{i+1})$

When $\lambda < 0$, $\vec{P}(x, y)$ points

Beneath $\vec{P}_i(x_i, y_i)$

Example: Suppose a starting point $\vec{P}_i(z, 3)$, and ending point $\vec{P}_{i+1}(7, 9)$.

Find:

- 1) Directional Vector $\vec{J}(x, y)$
- 2) Find the Line Equation $\vec{P}(x, y)$.
- 3) Find λ that defines $\frac{1}{4}$ of the distance from the pt. $P_i(x_i, y_i)$.

Sol:

$$\begin{aligned} \text{1) From Egn (2), } \\ \vec{J}(x, y) &= \vec{P}_{i+1}(x_{i+1}, y_{i+1}) - \vec{P}_i(x_i, y_i) \\ &= \vec{P}_{i+1}(7, 9) - \vec{P}(z, 3) \\ &= (7-z, 9-3) \\ &= (5, 6) \end{aligned}$$

C/C++ Code

$$\text{direc.x}[i] = P^t[i+1].x - P^t[i].x$$

2) Line Egn

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

\Rightarrow
 $= \vec{P}(z, 3) + \lambda [\vec{P}(7, 9) - \vec{P}(z, 3)]$
 $= (z, 3) + \lambda (7-z, 9-3)$
 $= (z, 3) + \lambda (5, 6)$

(3) Since $\frac{1}{4}$ of the distance from P_i , so let $\lambda = \frac{1}{4}$

Find that $\frac{1}{4}$ pt.

from

$$\begin{aligned} \vec{P}(x, y) &= (z, 3) + \lambda (5, 6) \Big|_{\lambda=\frac{1}{4}} \\ &= (z, 3) + \frac{1}{4} * (5, 6) \\ &= \left(\frac{8}{4} + \frac{5}{4}, \frac{12}{4} + \frac{6}{4}\right) \\ &= \frac{1}{4}(13, 18) \end{aligned}$$

Sept 25 (Monday)

Example: Prep. for Project / Design





Step 1. Design/Defining A set of
4 vectors/pts/Vertices

$$\{\vec{P}_i(x_i, y_i) \mid i=1, 2, \dots, 4\}$$

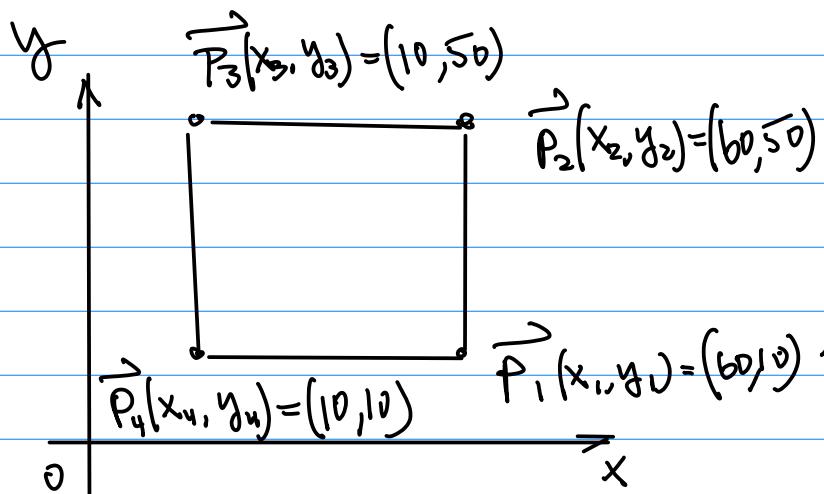
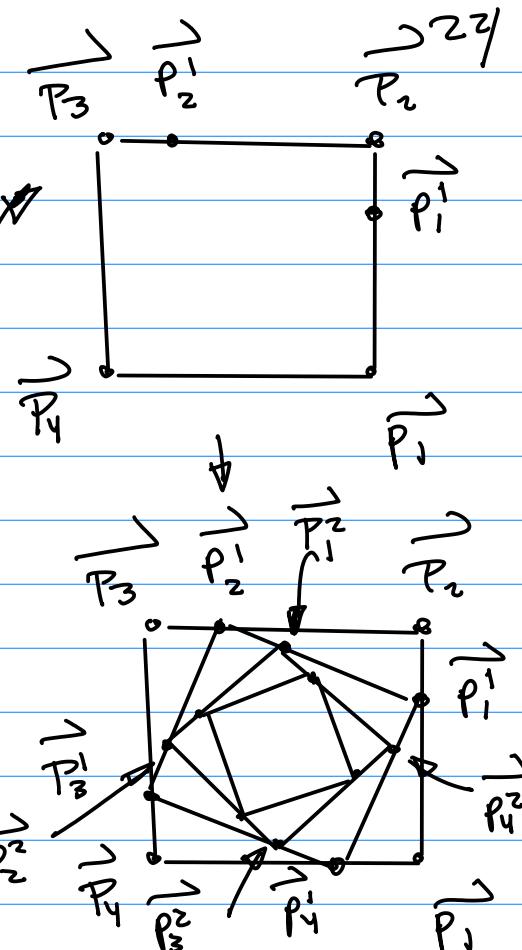
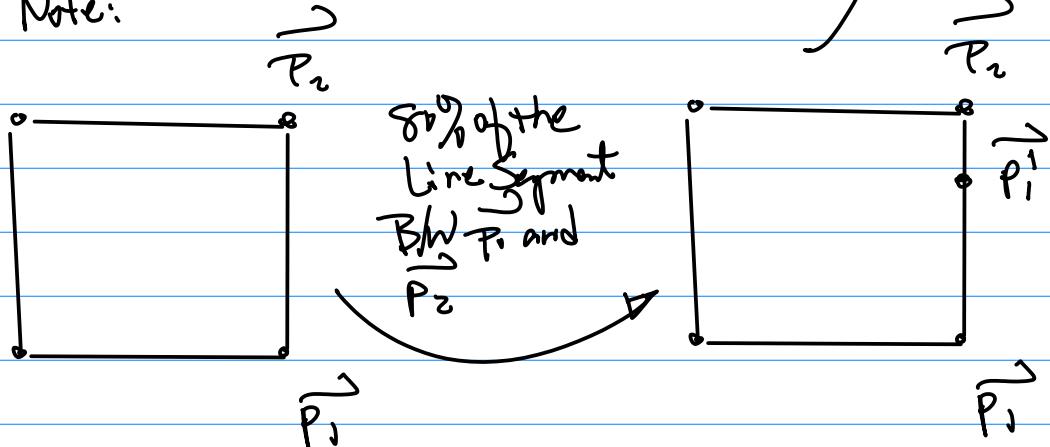


Fig.1 Vertical Coordinate System

Note: \vec{P}_i , for $i=1, 2, \dots, N$, is arranged in a Counter Clockwise direction. As a Convention.

(For \rightarrow Hidden Line/Hidden Surface Removal).

Note:



Note: 80% pts produces a Square of Counter Clockwise Rotation!!

From Eqn (3) on pp. 20,

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

Let's $\lambda = 0.8$ for the above design.

for $i=1, 2, 3, 4$, $i+1 = 5$?

"Mod" operator
so, $i+1 \mod 5 = 1$

Consider Adding Superscript to define the levels.

$$\vec{P}_i^{j+1}(x_i^j, y_i^j) =$$

$$\vec{P}_i^j(x_i^j, y_i^j) + \lambda (\vec{P}_{i+1}^j(x_{i+1}^j, y_{i+1}^j) - \vec{P}_i^j(x_i^j, y_i^j))$$

Note: For the project, make $\dots(1)$
 $j \geq 10$

Coding Aspects:

$$x_i^{j+1} = x_i^j + \lambda (x_{i+1}^j - x_i^j)$$

... (za)

Color (r, g, b).

Intensity, 8 bit.

Width of the line

Style of the line

Sept. 27 (Wed).

Note: Project 1 assignment will be posted Today. Due 2 weeks.

Continuation on Project Screen Saver Design.

$$x[i+1][j] = x[i][j] + \text{lambda} *$$

$$(x[j][i+1] - x[i][j]);$$

Example: Code Sample, GitHub, drawline

Graphic Controller's Driver

Program, 2D tri ity

graphics function.

a.

i. Draw a Single pixel @ (x_i, y_i)

b. Color, r, g, b (red, green, blue)

c. Intensity 8 bit for each color.

$$(r(x, y), g(x, y), b(x, y))$$

ii. Draw A Line :

OpenGL

Starting pt.
(x_p, y_p),

(x_g, y_g). Ending pt.

Color (r, g, b).

Intensity, 8 bit.

Width of the line

Style of the line

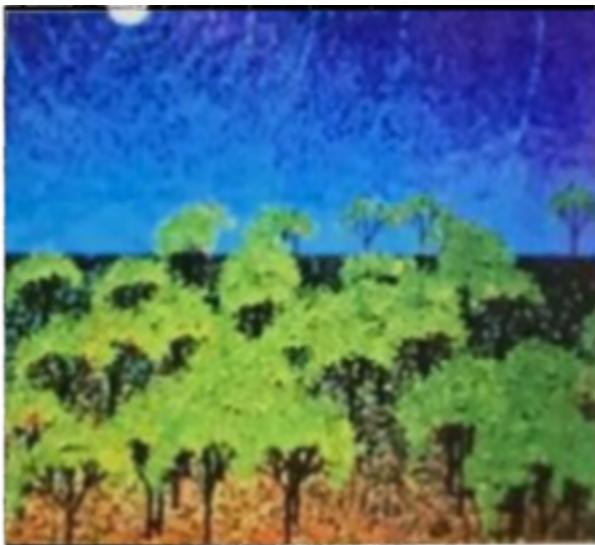


Fig.1

Construct / Generate the
Screen Saver.

Then, Create Branches.

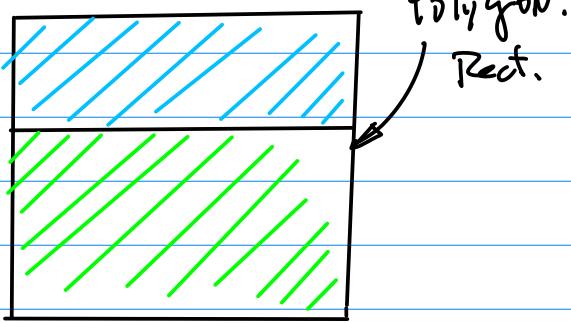


Fig.2

A Single Tree.

Build a trunk first.
With a starting point
 $\vec{P}_i(x_i, y_i)$ and ending
point $\vec{P}_{i+1}(x_{i+1}, y_{i+1})$.

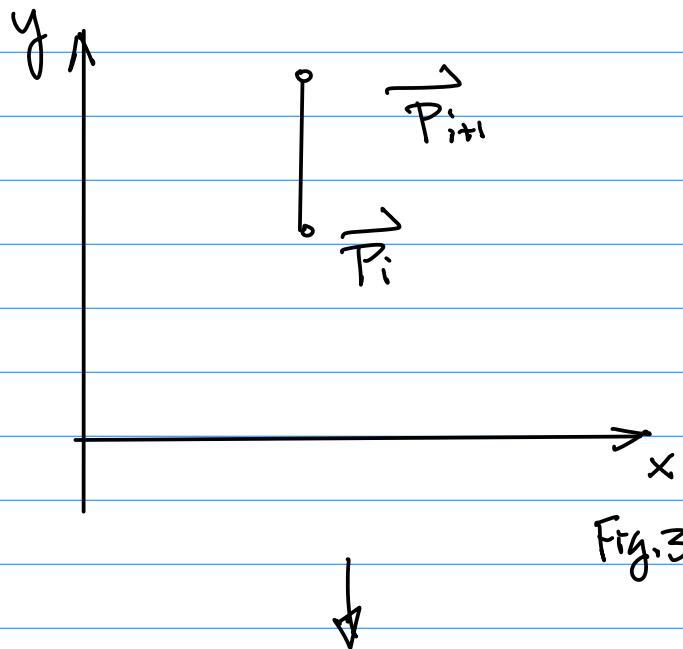


Fig.3

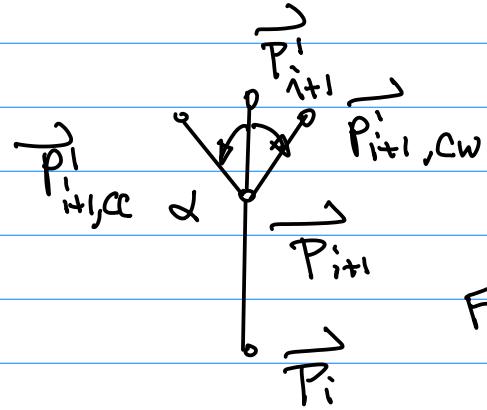


Fig.4

Duplicate the main Branch with $x=0.8$
(80% Reduction of its Length)

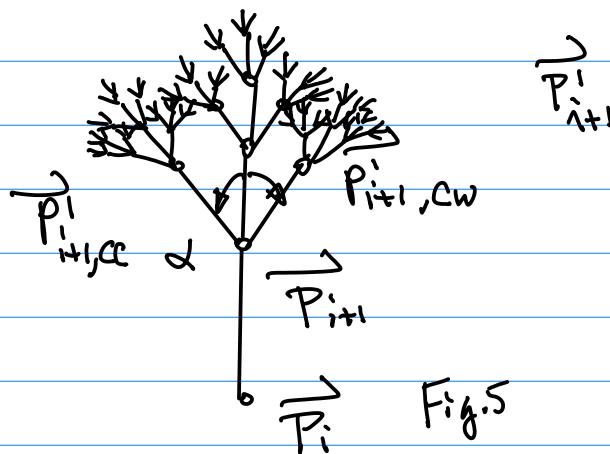
$$\vec{P} = \vec{P}_i + \lambda (\vec{P}_{i+1} - \vec{P}_i) \text{ for Example.}$$

$$\vec{P} = \vec{P}_{i+1} + \mu (\vec{P}_{i+1} - \vec{P}_i) \text{ OR}$$

Rotation Counter Clockwise to form
the side branch, denoted as $\vec{P}_{i+1,CC}$

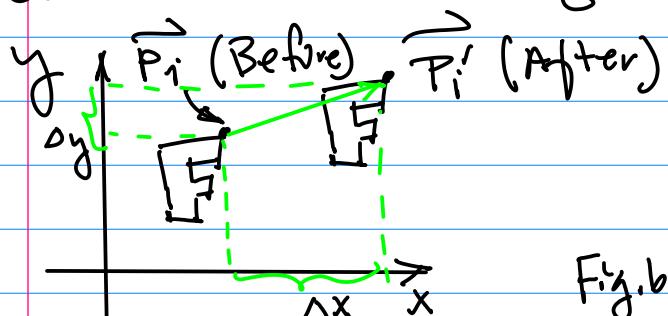
Rotation Clockwise to form
the 2nd branch (Symmetric)
denote it as $\vec{P}_{i+1,CW}^i$

Next, Continue the process
to grow the level for each
and every 2nd Points



Notations : Use the Ending points!
Subscript, then increment the
Level, e.g., the SuperScript
by 1.

Consider 2D Transformations. { Translation }



$$F: \left\{ \vec{P}_i(x_i, y_i) \mid i=1, 2, \dots, N \right\}$$

Translation.

After

$$\vec{P}_i'(x'_i, y'_i)$$

Before

$$\vec{P}_i(x_i, y_i)$$

$$\underline{x'_i} = x_i + \Delta x \dots (1a)$$

Similarly, for y:

$$\underline{y'_i} = y_i + \Delta y \dots (1b)$$

Introduce a col. vector for \vec{P}_i and \vec{P}_{i+1} . With added 1 Dummy Dimension.

So

$$\begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} \text{ for } \vec{P}_i(x_i, y_i)$$

$$\begin{pmatrix} x_{i+1} \\ y_{i+1} \\ 1 \end{pmatrix} \text{ for } \vec{P}_{i+1}(x_{i+1}, y_{i+1}).$$

Hence for $\vec{P}_i'(x'_i, y'_i)$:

$$\begin{pmatrix} x'_i \\ y'_i \\ 1 \end{pmatrix}$$

Therefore,

$$\begin{pmatrix} x'_i \\ y'_i \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & \Delta x \\ 0 & 1 & \Delta y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix}$$

... (1c)

Rotation:

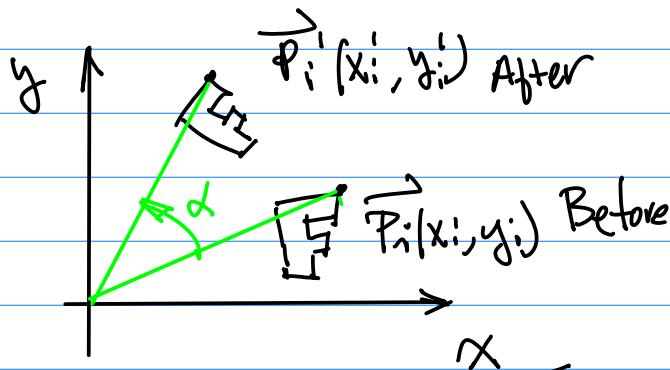


Fig. 7

α : CounterClockwise $\rightarrow \alpha > 0$

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

... (2)

Example: To perform Rotation
in Fig. 4 in the tree Design.

Let's Translate P_{ini} to $(0,0)$.
the origin

Step 1. Translation

$$\begin{pmatrix} x'_{ini} \\ y'_{ini} \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & DX \\ 0 & 1 & DY \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_{ini} \\ y_{ini} \\ 1 \end{pmatrix}$$

After Before

where $DX = -x_{ini}$

$DY = -y_{ini}$

Rotation

$$\begin{pmatrix} x'_{ini} \\ y'_{ini} \\ 1 \end{pmatrix} = \begin{pmatrix} \cos \alpha & -\sin \alpha & 0 \\ \sin \alpha & \cos \alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix}$$

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

$$= R \cdot T \begin{pmatrix} x_{ini} \\ y_{ini} \\ 1 \end{pmatrix}$$

Oct, 2nd (Monday).

Note 1. Quiz is scheduled

ON Next Monday :

1° Prototype Board
With CPU/LCD
mounted and Ready
to go.

2° Run Drawline Code;

3° Take 2 photos.

a). Entire System with
your Laptop System;

b) Screen Capture of
DrawLine Result.

4° Please Bring A Blank
Printer Paper, Answer

the questions (z), then take a photo using your Smartphone, send the photo to your Laptop, then Convert the photo to pdf format.

5) Submission on CANVAS

Zip file.

- { 1. Photo (png, jpeg)
of the System.
- 2. Photo of the Drawline
Result;
- 3. pdf Page

FirstName_LastName_SID(4 Digits)-Z40.zip

Note 2: Project 1 is Due 2 weeks.

Oct 15 (Sun), 11:59 pm

Requirements to Be Posted
on CANVAS.

Example: Continuation from

$$\begin{array}{c} R \cdot T \\ \hline \text{---} \\ \text{Rotation} \end{array} \quad \dots (1)$$

We need Postprocessing.
PostProcessing: To Undo the Preprocessing.

$$T^{-1} \cdot T = I$$

Inverse

\hookrightarrow physical meaning of T^{-1}
for Undoing Translation.

a) Translation Matrix.

by making $\Delta x \rightarrow -\Delta x$

b) Similarly for the y. Change
 $\Delta y \rightarrow -\Delta y$

$$T^{-1} = \begin{pmatrix} 1 & 0 & -\Delta x \\ 0 & 1 & -\Delta y \\ 0 & 0 & 1 \end{pmatrix} \quad \dots (2)$$

$$\begin{pmatrix} x'_i \\ y'_i \\ 1 \end{pmatrix} = T^{-1} R T \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix}$$

After Before
 ... (3)

$$\begin{aligned}
 T^{-1} R T &= \begin{pmatrix} 1 & 0 & -\Delta x \\ 0 & 1 & -\Delta y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & \Delta x \\ 0 & 1 & \Delta y \\ 0 & 0 & 1 \end{pmatrix} \\
 &= \begin{pmatrix} 1 & 0 & -\Delta x \\ 0 & 1 & -\Delta y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\theta & -\sin\theta & \cos\theta \cdot \Delta x - \sin\theta \cdot \Delta y \\ \sin\theta & \cos\theta & \sin\theta \cdot \Delta x + \cos\theta \cdot \Delta y \\ 0 & 0 & 1 \end{pmatrix} \\
 &= \begin{pmatrix} \cos\theta & -\sin\theta & \cos\theta \cdot \Delta x - \sin\theta \cdot \Delta y - \Delta x \\ \sin\theta & \cos\theta & \sin\theta \cdot \Delta x + \cos\theta \cdot \Delta y - \Delta y \\ 0 & 0 & 1 \end{pmatrix} \dots (8).
 \end{aligned}$$

Therefore, we have

$$\left\{
 \begin{array}{l}
 x_i'' = \cos\theta \cdot x_i - \sin\theta \cdot y_i + \cos\theta \cdot \Delta x - \sin\theta \cdot \Delta y - \Delta x \quad \dots (4a) \\
 y_i'' = \sin\theta \cdot x_i + \cos\theta \cdot y_i + \sin\theta \cdot \Delta x + \cos\theta \cdot \Delta y - \Delta y \quad \dots (4b)
 \end{array}
 \right.$$

Example: Drawing Code.

Emphasis on the Graphics Library
functions, e.g.

primitive graphics Routines.

Note: Resolution

```

30 uint8_t dest_addr[SSP_BUFSIZE];
31
32
33 #define ST7735_TFTWIDTH 127
34 #define ST7735_TFTHEIGHT 159
35
36 #define ST7735_CASET 0x2A
37 #define ST7735_RASET 0x2B

```

Col.
Row.

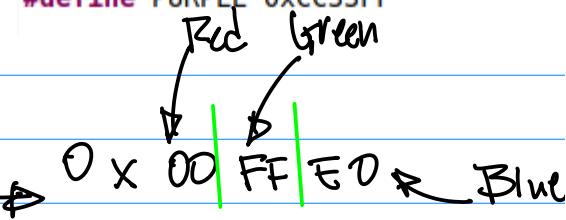
COMP724D

F2023

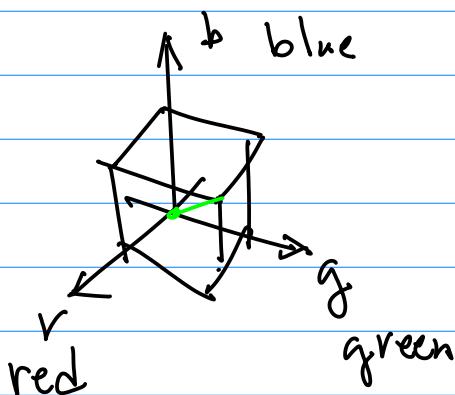
ZG1

Note: Color Space & Color Definition.

```
47  
48 #define LIGHTBLUE 0x00FFE0  
49 #define GREEN 0x00FF00  
50 #define DARKBLUE 0x000033  
51 #define BLACK 0x000000  
52 #define BLUE 0x0007FF  
53 #define RED 0xFF0000  
54 #define MAGENTA 0x00F81F  
55 #define WHITE 0xFFFFFFFF  
56 #define PURPLE 0xCC33FF  
57
```



R,g,b primitive color is used to define colors



r-g-b color Space

(0,0,0) Black

(1,1,1) White

from (0,0,0) to (1,1,1) on the line, we have gray color.

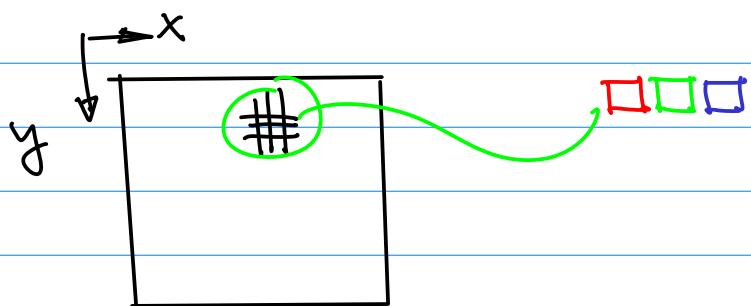
Brightest Red (1,0,0)

.. Green (0,1,0)

.. Blue (0,0,1)

Note:

```
123  
124 void write888(uint32_t color, uint32_t repeat)  
125  
126 {  
127     uint8_t red, green, blue;  
128     int i;  
129     red = (color >> 16);  
130     green = (color >> 8) & 0xFF;  
131     blue = color & 0xFF;  
132     for (i = 0; i < repeat; i++) {  
133         writedata(red);  
134         writedata(green);  
135         writedata(blue);  
136     }
```



Oct.4 (Wed).

Example: Continuation on the ST7735 Graphics Lib.

```

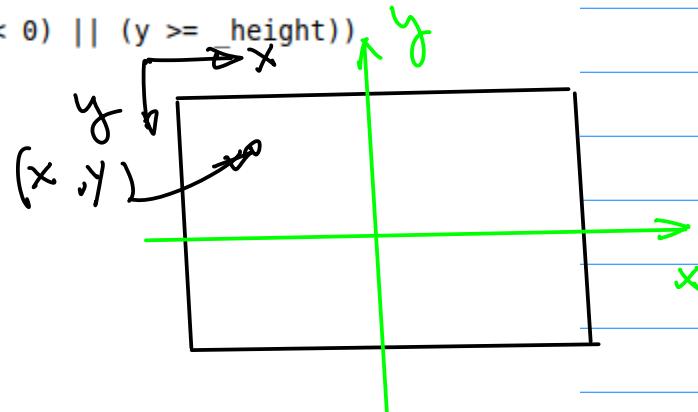
1 /v
171 void fillrect(int16_t x0, int16_t y0, int16_t x1, int16_t y1, uint32_t color)
172 {
173     int16_t i;
174     int16_t width, height;
175     width = x1-x0+1;
176     height = y1-y0+1;
177     setAddrWindow(x0,y0,x1,y1);
178     writecommand(ST7735_RAMWR);
179     write888(color,width*height);
180 }
181
182
183
184
185
186
187
188
189
190

```

```

250
251 void drawPixel(int16_t x, int16_t y, uint32_t color)
252 {
253     if ((x < 0) || (x >= _width) || (y < 0) || (y >= height))
254         return;
255     setAddrWindow(x, y, x + 1, y + 1);
256     writecommand(ST7735_RAMWR);
257     write888(color, 1);
258 }
259
260
261
262
263
264
265
266

```



Consider the Mapping from
a physical Display (Coordinate)
to a Virtual Display.

Motivations: Portable code
Development, Simplicity.



2022F-101-notes-part1-
cmpe240-2022-11-30.pdf

After ? Before

$$x_p = \frac{M}{2} + x_v$$

Therefore

$$x_p = x_v + \frac{M}{2}$$

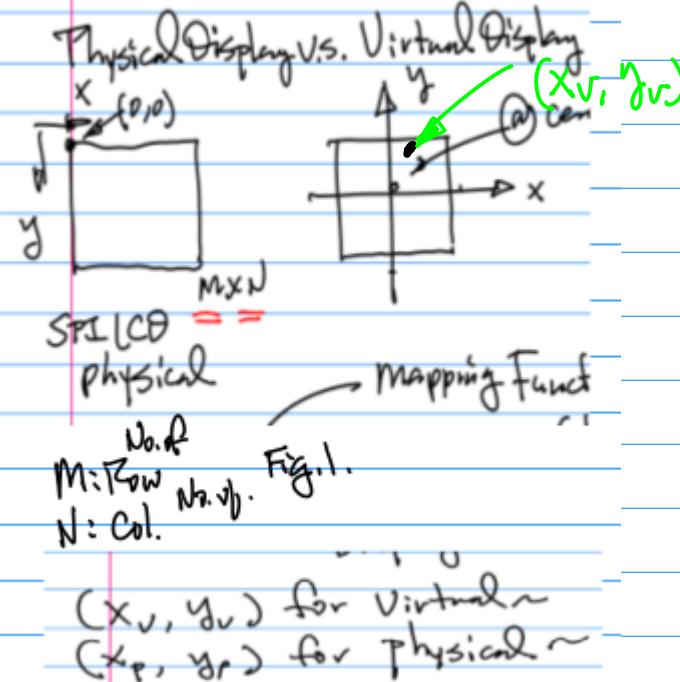
Eqn(7) from pp.23 Ref. is verified. for y

After Before

$$y_{p} = \frac{N}{2} - y_v$$

Total Number
of Rows.

Pointing to
the opposite
direction



Note: The mapping from Virtual display to the physical display is given in the following equations

$$x_p = x_v + \frac{M}{2} \dots (7)$$

$$y_p = -y_v + \frac{N}{2} \dots (8)$$

Consider X Comp. "Before" (e.g. in the virtual display) and "After" (e.g. in the physical display)

Introduction to 3D G.E.
(Graphics Engine Design).

Oct 9th (Monday)

Ref:

2022F-101-notes-part1-cmpe240-2022-11-30.pdf

Example: World Coordinate System.
V.S.

Viewer Coordinate System.
(For the Virtual Camera).

Note: 1° Notation for the world
Coordinate System. $x_w y_w z_w$
Subscript for the
World.

2° Right-Hand System.

Check Vector Cross Product,

$$\vec{x_w} \times \vec{y_w} = \vec{z_w}$$

3° Define A Virtual Camera

location @ $\vec{E}(x_e, y_e, z_e)$

4° Always Point the Camera towards
to the Origin of the $x_w-y_w-z_w$
for Simplicity.

See Fig. 4 on pp. 28 from the
Ref.

We can find the distance from

$\vec{E}(x_e, y_e, z_e)$ to the Origin

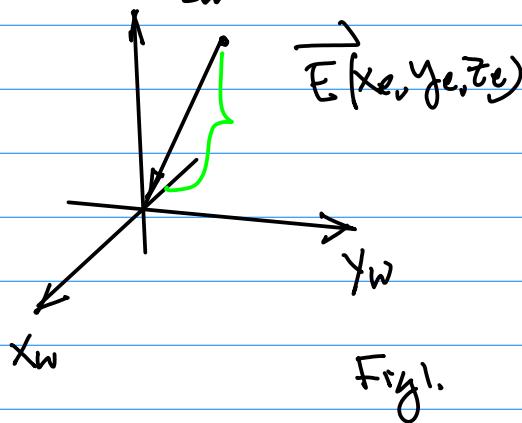


Fig. 1.

$$\text{Distance (magnitude)} = \sqrt{x_e^2 + y_e^2 + z_e^2} \quad \dots (1)$$

5° Draw A Vector Passing through \vec{E}
Perpendicular to x_w-y_w plane.

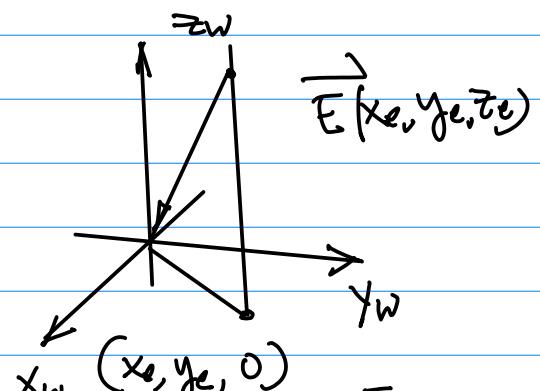


Fig. 2

Draw a Line from the Intersection
Point to the Origin of the World
Coordinate System as in Fig. 2.

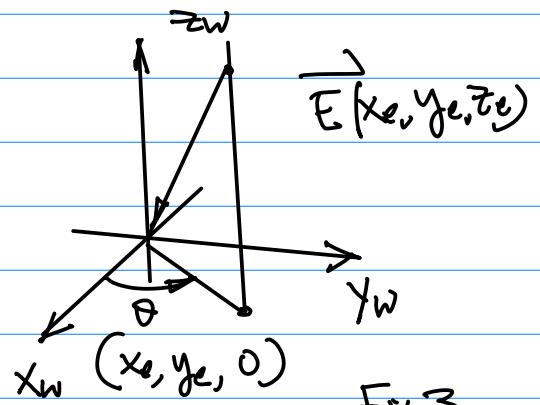
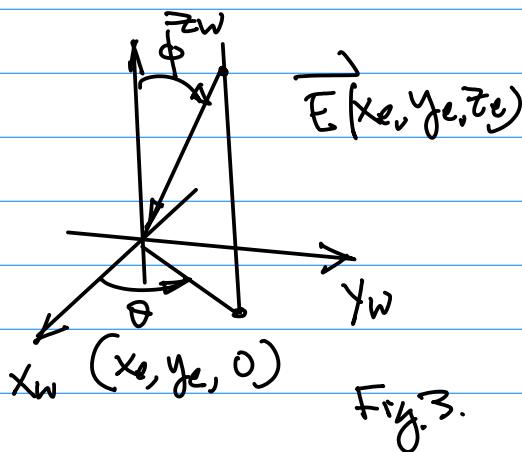


Fig. 3.

6° Define An Angle w.r.t. $(0,0,0)$
and Positive x_w -axis,
Counter Clockwise Rotation,

Define 2nd Angle. ϕ



2nd Angle ϕ : W.r.t Z_w -axis

Counter Clockwise : Positive.

$\phi > 0$ if it is counter
Clockwise

Oct. 11th (Wed).

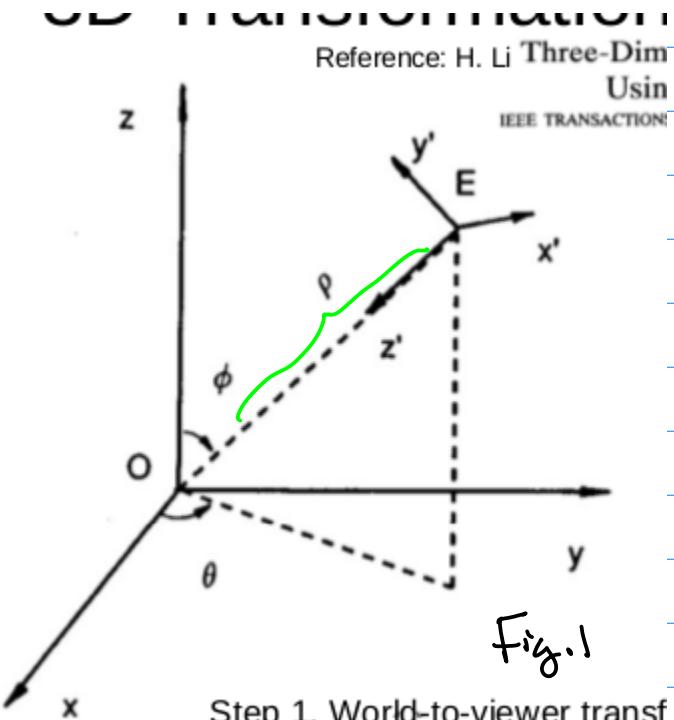
Continuation on offsetting w.r.t

Viewer Coordinate System.

Transformation Pipe.

Ref: Lect. github.

2023S-101-note-Part1. Note, Transformation Pipeline



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

- 2018F-114-lec-3D-vectors-v3-2018-10-26.pdf
- 2018F-114-lec5-3D-Shade-v3-2018-10-29.pdf
- 2018F-114-lec5-3D-Shade-v5-2018-11-4.pdf
- 2018F-115-lab-DiffuseReflection-Rubrics.txt
- 2018F-116-11diffuse20181114.cpp
- 2018F-117-12dda.cpp
- 2018F-118-13diffuseInterpolation20181127.cpp

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

... (z)

before (in the world Coordinate)

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

x_e-y_e-z_e: Left Hand System.

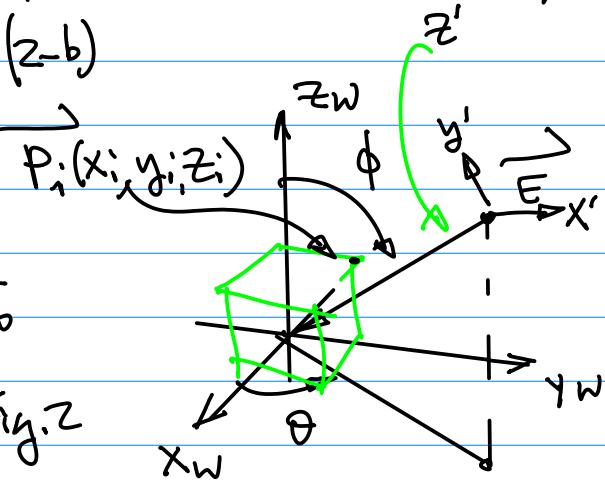
After (in the Viewer coordinate)

... (z-b)

$$\begin{pmatrix} - & & & \\ - & - & & \\ - & - & - & \end{pmatrix} \quad 4 \times 4$$

Side of the Cube = 100

Fig.2



$$\begin{pmatrix} \sin \theta & \cos \theta & 0 & 0 \\ \cos \theta & \sin \theta & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \quad 4 \times 4$$

Given A Cube with One of its Vertices is $\vec{P}_i(x_i, y_i, z_i) = (0, 100, 100)$

Now, using Eqn (z-b), we can find

$$\begin{pmatrix} \cos \phi & \sin \phi & \sin \theta & 0 \\ \sin \phi & \cos \phi & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad 4 \times 4$$

!!

$$\begin{pmatrix} x'_i \\ y'_i \\ z'_i \\ 1 \end{pmatrix} \text{ after the } x_w-y_w-z_w \rightarrow x_e-y_e-z_e$$

Assume $E(x_e, y_e, z_e) = (200, 200, 200)$ First, find $\sin \theta =$

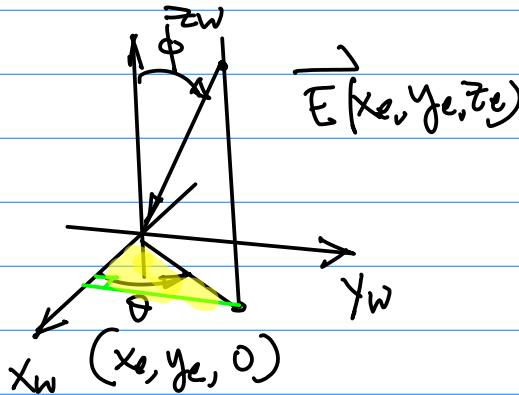


Fig.3.

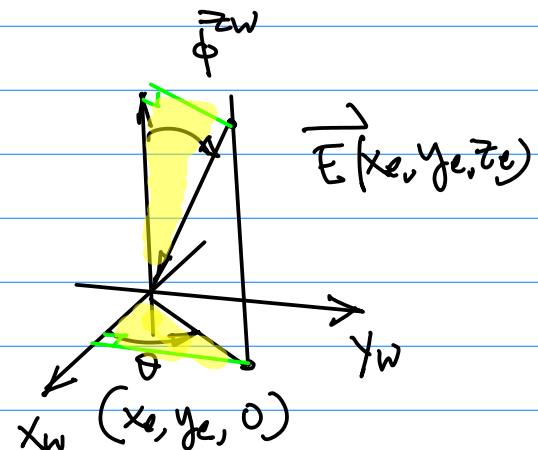


Fig.4

$$\sin \theta = \frac{\text{Green Line}}{\Delta} \quad \text{where } \Delta = \sqrt{x_e^2 + y_e^2}$$

$$\sin \theta = \frac{y_e}{\sqrt{x_e^2 + y_e^2}} \quad \dots (3)$$

Now, at the Same Triangle,

$$\cos \theta = \frac{x_e}{\sqrt{x_e^2 + y_e^2}} \quad \dots (4)$$

From the given condition,

$$\vec{E}(x_e, y_e, z_e) = (200, 200, 200)$$

then,

$$\begin{aligned} \sin \theta &= \frac{200}{\sqrt{200^2 + 200^2}} = \frac{200}{200\sqrt{2}} = \\ &= \frac{1}{\sqrt{2}} = \frac{\sqrt{2}}{2} \end{aligned}$$

$$\sin \phi = \frac{\text{Green Line}}{\rho}$$

$$\text{where Green Line} = \sqrt{x_e^2 + y_e^2}$$

$$\rho = \sqrt{x_e^2 + y_e^2 + z_e^2}$$

$$\begin{aligned} \sin \phi &= \frac{\sqrt{x_e^2 + y_e^2}}{\rho} \\ &= \frac{\sqrt{x_e^2 + y_e^2}}{\sqrt{x_e^2 + y_e^2 + z_e^2}} \quad \dots (4-b) \end{aligned}$$

$$\begin{aligned} &= \frac{\sqrt{200^2 + 200^2}}{\sqrt{200^2 + 200^2 + 200^2}} = \frac{\sqrt{2}}{\sqrt{3}} \\ &= \frac{\sqrt{2} \cdot \sqrt{3}}{3} = \frac{\sqrt{6}}{3} \end{aligned}$$

Therefore, the mapping matrix is

$$\cos \phi = \frac{z_e}{\sqrt{x_e^2 + y_e^2 + z_e^2}} \quad \dots (5)$$

$$= \frac{200}{200\sqrt{3}} = \frac{1}{\sqrt{3}} = \frac{\sqrt{3}}{3}$$

Similarly,

$$\cos \theta = \frac{x_e}{\sqrt{x_e^2 + y_e^2}} = \frac{\sqrt{2}}{2}$$

$$\begin{bmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 & 0 \\ \frac{\sqrt{2}\sqrt{3}}{2\sqrt{3}} & \frac{\sqrt{2}\sqrt{3}}{2\sqrt{3}} & \frac{\sqrt{6}}{3} & 0 \\ \frac{\sqrt{2}\sqrt{6}}{2\sqrt{3}} & \frac{\sqrt{2}\sqrt{6}}{2\sqrt{3}} & \frac{\sqrt{3}}{3} & 20\sqrt{3} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Therefore, the New Pt After

Transformation from $x_w-y_w-z_w$
 after
 $(x_e-y_e-z_e)$ to $x_e-y_e-z_e$ is:

$$\begin{bmatrix} x'_e \\ y'_e \\ z'_e \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 & 0 \\ \frac{\sqrt{2}\sqrt{3}}{2\sqrt{3}} & \frac{\sqrt{2}\sqrt{3}}{2\sqrt{3}} & \frac{\sqrt{6}}{3} & 0 \\ \frac{\sqrt{2}\sqrt{6}}{2\sqrt{3}} & \frac{\sqrt{2}\sqrt{6}}{2\sqrt{3}} & \frac{\sqrt{3}}{3} & 20\sqrt{3} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 100 \\ 110 \\ 1 \end{bmatrix}$$

Before
 (In $x_w-y_w-z_w$ World)

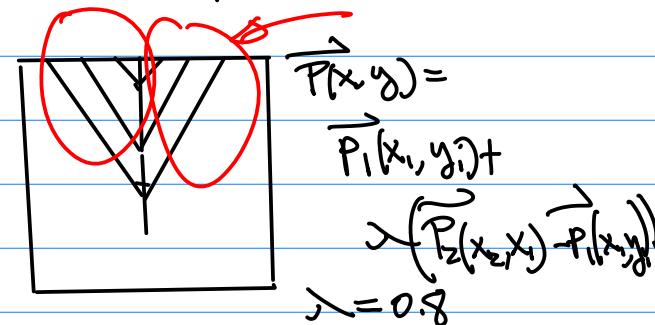
Oct. 16 (Mon).

- Note 1. Quiz update,
- Naming convention to be followed;
 - Personal ID Required.
 - Quiz Exam Cannot be Submitted Late.

Note 2. Form 2 Person Team.

Work Together, Debugging,
 Designing Idea. But All work,
 Coding, Pictures, Reports have
 to be individual / Independently.

Q3A.



2^{o} $X[i][j]$

Index for Superscript for the different level.

Example, Continuation.

Find the pt. in the Xe-Ye-Ze,
e.g. After the World to Viewer.

$$\vec{P}_i^{j+1}(x_i^{j+1}, y_i^{j+1}) = \vec{P}_i^j(x_i^j, y_i^j) +$$

$$\lambda \left(\vec{P}_{i+1}^j(x_{i+1}^j, y_{i+1}^j) - \vec{P}_i^j(x_i^j, y_i^j) \right)$$

... (1)

$$x_i^j = \frac{\sqrt{2}}{2}x_0 + \frac{\sqrt{2}}{2}x_1 + 0x_2 + 0x_3 \\ = 50\sqrt{2}$$

3^{o} $T^{-1}RT \dots (2)$

$$T = \begin{pmatrix} \alpha x & 0 & 0 \\ 0 & \alpha y & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 & 0 \\ \frac{\sqrt{2}\sqrt{3}}{2} & \frac{\sqrt{2}\sqrt{3}}{2} & \frac{\sqrt{6}}{3} & 0 \\ \frac{\sqrt{2}\sqrt{6}}{2} & \frac{\sqrt{2}\sqrt{6}}{2} & \frac{\sqrt{3}}{3} & 20\sqrt{3} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 100 \\ 110 \\ 1 \end{bmatrix}$$

from Eqn (1).

$$\vec{P}_i^{j+1}(x_i^{j+1}, y_i^{j+1})$$

$$y_i^j = \frac{\sqrt{2}\sqrt{3}}{2}x_0 + \frac{\sqrt{2}\sqrt{3}}{2}x_1 + \frac{\sqrt{6}}{3}x_2 + 0x_3$$

$$= 50\sqrt{6} + \frac{10}{3}\sqrt{6} = \left(50 + \frac{10}{3}\right)\sqrt{6}.$$

4^{o} Color: We would like

to have more colors

than the graphic colors.

C-Code defined

a) r, g, b primitive color.

8 bit Each.

b) Total Possible colors.

Hence it's $2^4 = 2^4 \cdot 2^{20}$

= 16 M.

$$z_i^j = 0 + \frac{50\sqrt{2}\sqrt{3}}{3} + \frac{10\sqrt{3}}{3} + 200\sqrt{3}$$

$$= \frac{50\sqrt{6}}{3} + \frac{10\sqrt{6}}{3} + 200\sqrt{3}.$$

C-Code Implementation.

$$X_{\text{Prim}}[i] = -\sin(\Theta) * x[i] +$$

$$\cos(\Theta) * y[i];$$

$$Y_{\text{Prim}}[i] = -\cos(\Theta) * \cos(\Phi) * x[i]$$

$$-\sin(\Theta) * \cos(\Phi) * y[i]$$

$$+\sin(\Phi) * z[i];$$

$$\begin{aligned} z_{\text{prim}}[i] = & -\cos(\theta) * \sin(\phi) * x[i] - \cos(\theta) * \sin(\phi) * y[i] \\ & - \cos(\phi) * z[i] + \rho; \end{aligned}$$

Coding Example on the github.



- 2018F-117-12dda.cpp
- 2018F-118-13diffuseInterpolation20181127.cpp
- 2018S-17-Lab-report-rubrics.txt
- 2019F-100-accessible_CMPE240-F19HarryLi.pdf

Note 1. $\sin()$; $\cos()$ Computation. See Egn(3), (4), (4-b), and (5), on PP.35.

```
127     //sin and cosine computation for world-to-viewer
128     float sPheta = Ye / sqrt(pow(Xe, 2) + pow(Ye, 2));
129     float cPheta = Xe / sqrt(pow(Xe, 2) + pow(Ye, 2));
130     float sPhi = sqrt(pow(Xe, 2) + pow(Ye, 2)) / Rho;
131     float cPhi = Ze / Rho;
```

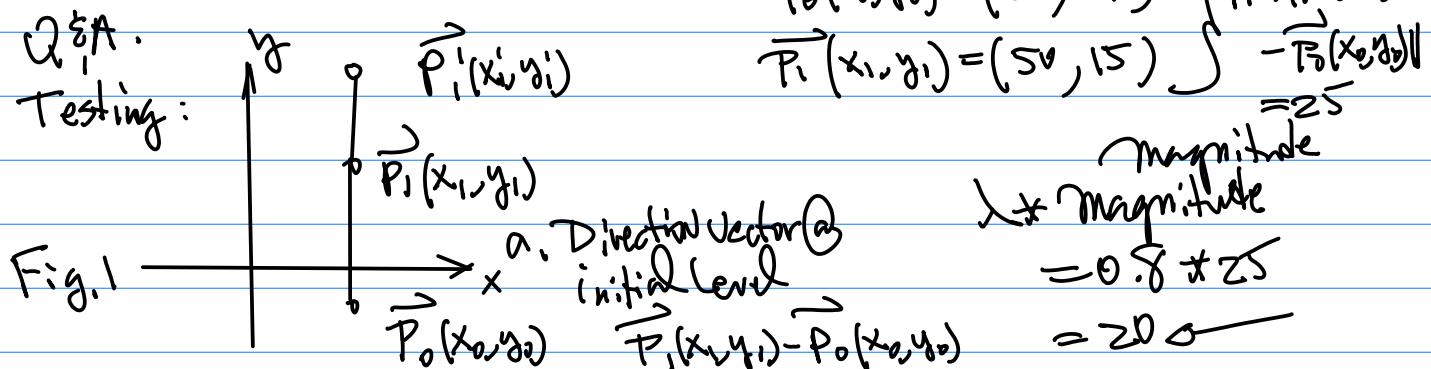
Note 2. World-Z-Viewer Transformation, $x_{\text{prim}}[i]$ in Our Discussion.

```
240     /*-----Transformation pipeline-----*/
241     for(int i = 0; i <= UpperBD; i++)
242     {
243         /*-----world to viewer-----*/
244         viewer.X[i] = -sPheta * world.X[i] + cPheta * world.Y[i];
245         viewer.Y[i] = -cPheta * cPhi * world.X[i]
246             - cPhi * sPheta * world.Y[i]
247             + sPhi * world.Z[i];
248         viewer.Z[i] = -sPhi * cPheta * world.X[i]
249             - sPhi * cPheta * world.Y[i]
250             - cPheta * world.Z[i] + Rho;
251     }
```

Oct.18(wed).

Q&A.

Testing:



$$\vec{P}_1^i(x_1^i, y_1^i) = (50, 15+20) \\ = (50, 35)$$

$$T = \begin{pmatrix} \Delta x & 0 & 0 \\ 0 & \Delta y & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$\vec{P}_1^i(x_1^i, y_1^i) = \vec{P}_1^0(x_1^0, y_1^0) + \lambda (\vec{P}_1^0(x_1^0, y_1^0) - \vec{P}_0^0(x_0^0, y_0^0))$$

$$\Delta x = -x_1 \\ \Delta y = -y_1$$

$$= (50, 15) + 0.8 * \left((50, 15) - (50, 10) \right) \\ = (50, 15) + 0.8 * (0, 5) \\ = (50, 15) + (0, 20) \\ = (50, 35)$$

$$T^{-1} = \begin{pmatrix} -\Delta x & 0 & 0 \\ 0 & -\Delta y & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$-\Delta x = x_1, -\Delta y = y_1$$

Example: Continuation of Transformation

Pipeline. Project 3D Objects.

in $x-e-y-e-z-e$ to,

Reference: H. Li Three-Dim
Using IEEE TRANSACTION

2D Display Screen
By Perspective Projection

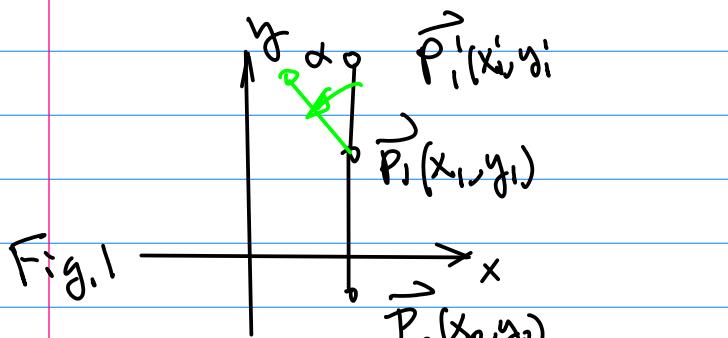
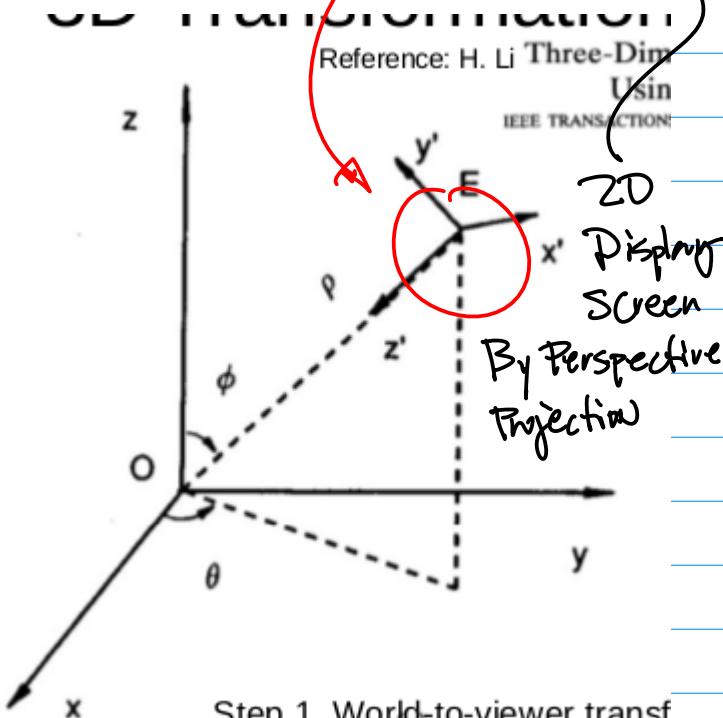


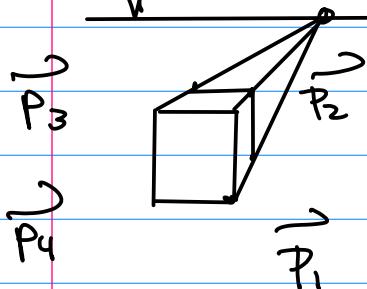
Fig.1

Note: 1° Rotation is w.r.t.
 $\vec{P}_1(x_1, y_1)$ in the
Project Design. However

$R_{3 \times 3}$ is defined w.r.t.
the origin $(0,0)$ of
x-y plane.

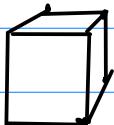


Background : ① Horizon
② Vanishing Point



③ Draw Lines
to Connect
Each Vertex
to the V.P.

④ Draw a Parallel plane
to the polygon, Then
Connect the intersection
points.



2D Display of
the Object with
depth Information.

Ref: Perspective Projection
2nd step of
the transformation pipeline

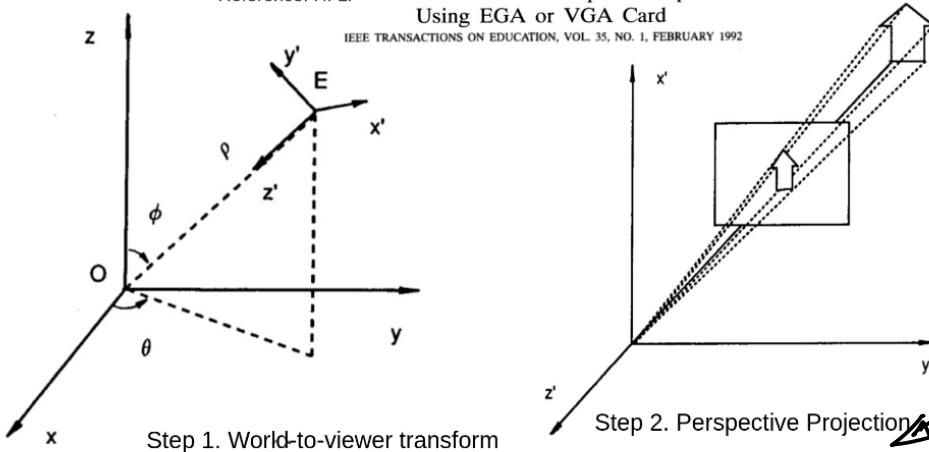
CMPE240-Adv-Microprocessors / 2014F / 2018F-114-lec-3D-vectors-v3-2018-10-26.pdf

hualili Add files via upload

449 KB Code 55% faster with GitHub Copilot

3D Transformation Pipeline Technique

Reference: H. Li Three-Dimensional Computer Graphics
Using EGA or VGA Card
IEEE TRANSACTIONS ON EDUCATION, VOL. 35, NO. 1, FEBRUARY 1992

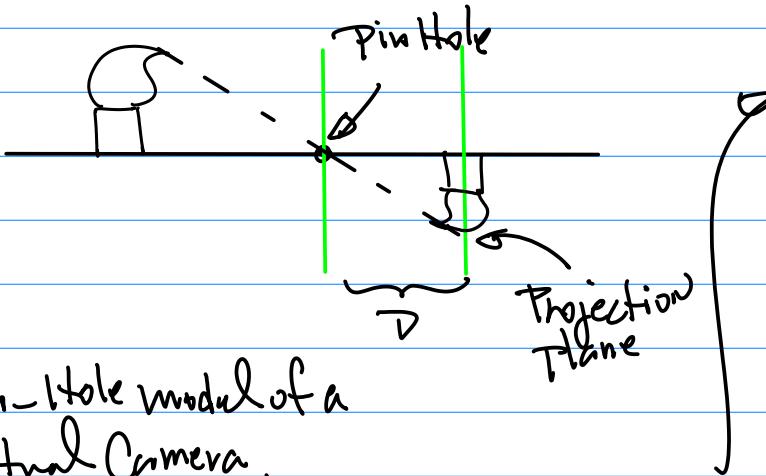


$$\mathbf{T} = \begin{bmatrix} -\sin \theta & \cos \theta & 0 & 0 \\ -\cos \phi \cos \theta & -\cos \phi \sin \theta & \sin \phi & 0 \\ -\sin \phi \cos \theta & -\sin \phi \sin \theta & -\cos \phi & \rho \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$x_p = x_e \left(\frac{D}{z_e} \right)$$

$$y_p = y_e \left(\frac{D}{z_e} \right)$$

... (1)
... (2)



Transformation Pipeline.
 ① World to Viewer.
 ② Perspective Projection
 $\text{Eqn.}(1) \not\equiv (2)$.

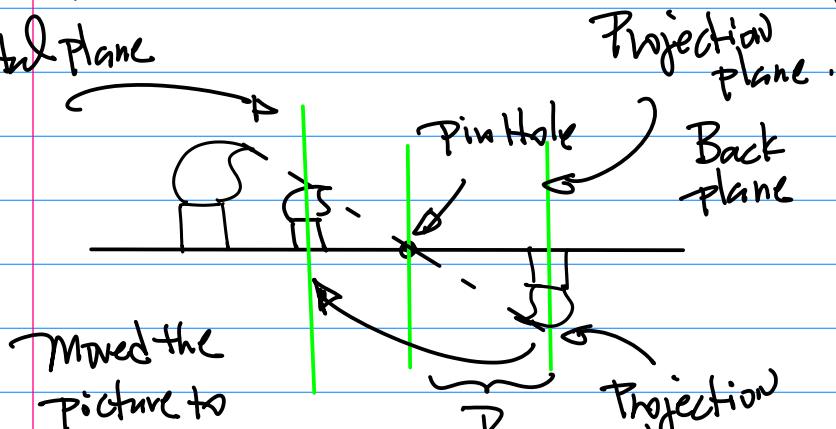
Note 1° $D \uparrow$, Picture Size \uparrow .

$D \downarrow$, Picture Size \downarrow

Matches Eqn (1) $\not\equiv$ (2).

Note 2°

frontal plane



Requirements.

Design & Draw World Coordinate

System $x_w-y_w-z_w$.

for x_w :

$$\vec{P}_0(x_0, y_0, z_0) = (0, 0, 0)$$

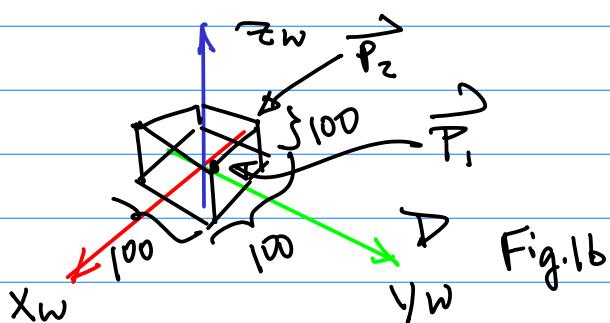
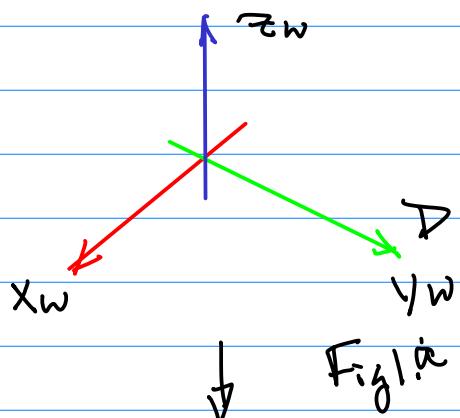
$$\vec{P}_1(x_1, y_1, z_1) = (50, 0, 0)$$

Draw Line to plot (x'_0, y'_0) to (x'_1, y'_1) After Perspective Projection

x_w

y_w

Oct. 23rd (Monday)

Example: Hand Calculation of
3D Transformation Pipeline.

$$\text{Option1: } \vec{E}(x_e, y_e, z_e) = (200, 200, 200)$$

$$\text{Option2: } \vec{E}(x_e, y_e, z_e) = (0, 0, 200)$$

Sol:
1. Data Points, Vertices of
the given Cube:

$$\begin{aligned} & \{\vec{P}_i(x_i, y_i, z_i) \mid i=1, 2, \dots, 8\} \\ &= \{\vec{P}_1(x_1, y_1, z_1), \vec{P}_2(x_2, y_2, z_2), \dots \\ & \quad \dots \vec{P}_8(x_8, y_8, z_8)\} \end{aligned}$$

Where

$$\vec{P}_1(x_1, y_1, z_1) = (50, 50, 100)$$

Note:

Pick the pts in A Counter Clockwise
fashion when viewing the object from
the Outside.

Removal of

Hidden Lines

Hidden Surfaces



$$\vec{P}_2(x_2, y_2, z_2) = (50, 50, 100) \text{ etc.}$$

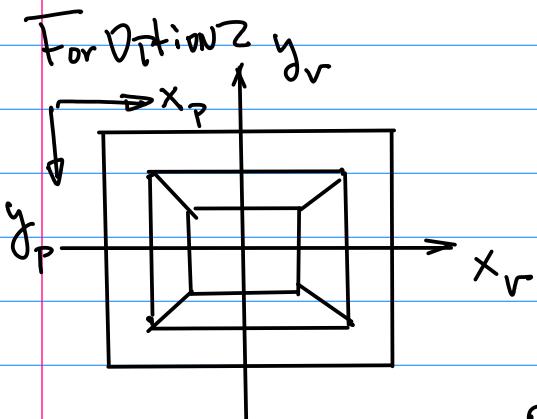


Fig. 2

2. First, Try Option1, then test option2

$$\vec{E}(x_e, y_e, z_e) = (200, 200, 200).$$

Eqn. for the $x_w - y_w - z_w$ to $x_v - y_v - z_v$

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

... (1)

Find $\cos\theta$, $\sin\theta$ and
 $\cos\phi$, $\sin\phi$.

(See Eqn(3)-(5), DN PP.35)

$$\text{distance } p = \sqrt{x_v^2 + y_v^2 + z_v^2}$$

$$\left\{ \begin{array}{l} x_p' = x_v' \frac{D'}{z_v'} \quad (za) \\ y_p' = y_v' \frac{D'}{z_v'} \quad (zb) \end{array} \right.$$

3. 2nd Step of the Transformation
 Pipeline:

$$\left\{ \begin{array}{l} x_p = x_v \frac{D}{z_v} \quad (za) \\ y_p = y_v \frac{D}{z_v} \quad (zb) \end{array} \right.$$

Assume $D=20$

Substitute it into Eqn(za) & (zb).

4. Mapping (x_p, y_p) to
 physical display.)

(x'', y'') to keep it from
 "Physical Display Notation"

Homework: Due A Week (Sunday,

29th.) Draw Color Coded

Part 1: $x_w - y_w - z_w$ (R-G-B)
 World Coordinate

System, And Display
 it On Your Prototype
 System.

Part 2. Use Transformation
 Pipeline Technique to
 Build Option 1 3D Cube

Now, Consider a New Subject.
 Ref: [github](#).

2021F-01b ~ Note

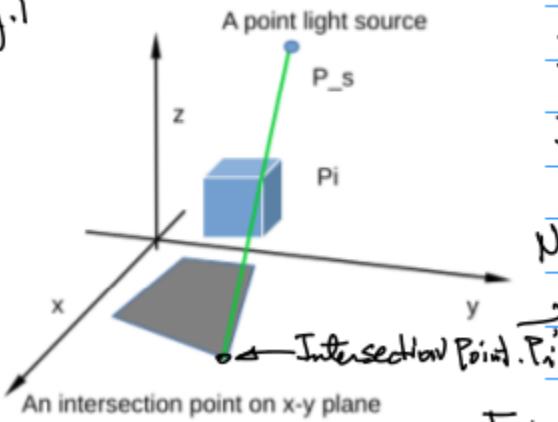
pp 51.

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 & 0 \\ \frac{\sqrt{2}\sqrt{3}}{2} & \frac{\sqrt{2}\sqrt{3}}{2} & \frac{\sqrt{6}}{3} & 0 \\ \frac{\sqrt{2}\sqrt{6}}{2} & \frac{\sqrt{2}\sqrt{6}}{2} & \frac{\sqrt{3}}{3} & 20\sqrt{3} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 50 \\ 50 \\ 100 \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} x_v \\ y_v \\ z_v \end{pmatrix}$$

Substitute it into
 Eqn(za) & (zb)

Fig.1



Example: Ref: On the github.

[2021F-101-notes-cmpe240-2021-11-8.pdf](#)

Background:

$\left\{ \begin{array}{l} 1. \vec{r} \text{ Ray Equation} \\ 2. \text{plane Equation} \end{array} \right.$

PP#6. Ray Equation

$$\vec{r}(x_r, y_r, z_r) =$$

$$\vec{P}_i(x_i, y_i, z_i) + \lambda (\vec{P}_s(x_s, y_s, z_s) -$$

$$\vec{P}_i(x_i, y_i, z_i))$$

$\dots (z)$

Note: \vec{P}_s or \vec{P}_i can be utilized for the Ray Equation (with different λ). But for the uniformity purpose, let's use \vec{P}_i .

To be posted
on the github.

Oct.25 (Wed)

Note 1. Canvas Posting of the Homework Due On Oct.29 (Sunday).

By the End of the day today.

Note 2. Midterm is Scheduled

After Shadow Computation.

Nov. 13 (Monday).

Note 3. Please Bring Your Prototype Systems to the Class for Discussion and Inspection (Homework on)

$X_w-Y_w-Z_w$ Drawing & Cube Wireframe display).

Example: Plane Equation.
from the Lecture Notes.

Ray Equation: PP#6.

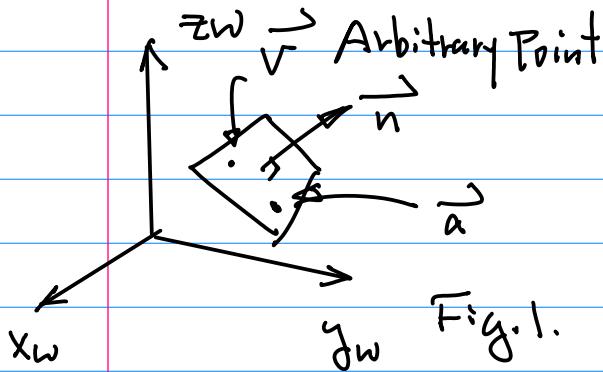
$$\vec{R} = \vec{P}_i + \lambda (\vec{P}_s - \vec{P}_i) \dots (3)$$

The Intersection Point on X_w-Y_w plane, \vec{P}_i' is a common pt shared by Ray Eqn(3) and plane eqn(z).

$$\vec{R} = \vec{P}_i + \lambda (\vec{P}_s - \vec{P}_i) \dots (4-1)$$

$$[\vec{n} \cdot (\vec{v} - \vec{a})] = 0 \dots (4-2)$$

Formulation.



\vec{n} : Normal Vector

To define A plane

- a. Need a Normal Vector $\vec{n}(n_x, n_y, n_z)$ for its Orientation;
- b. Need a Known Point ON the plane to define its Location. \vec{a}

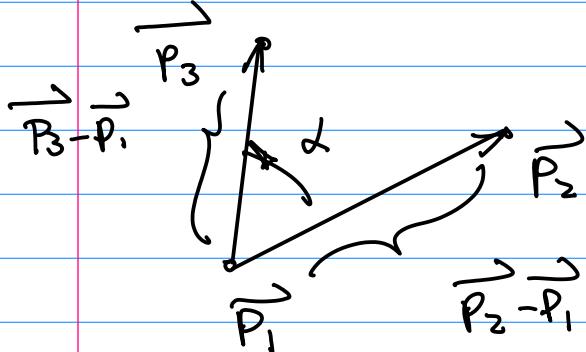
$$\vec{n}(n_x, n_y, n_z) = (n_x, n_y, n_z) \quad \dots (1)$$

And a known point \vec{a}

$$\vec{v} - \vec{a} \quad \dots (2) \text{ A line}$$

$$\vec{n} \cdot (\vec{v} - \vec{a}) = 0 \quad \dots (2-b)$$

Note: Vector Dot Product.



$$(\vec{P}_2 - \vec{P}_i) \cdot (\vec{P}_3 - \vec{P}_i)$$

$$= \|\vec{P}_2 - \vec{P}_i\| \|\vec{P}_3 - \vec{P}_i\|$$

Cos α

... (3)

Now, Let's find the Intersection point

From Eqn (4-2),

$$\vec{n} \cdot (\vec{v} - \vec{a}) = 0 \quad \dots (4)$$

$\vec{v} = \vec{P}_i + \lambda (\vec{P}_3 - \vec{P}_i)$

$$\vec{n} \cdot (\vec{P}_i + \lambda (\vec{P}_3 - \vec{P}_i) - \vec{a}) = 0$$

$$\vec{n} \cdot (\vec{P}_i + \lambda (\vec{P}_3 - \vec{P}_i) - \vec{a}) = 0$$

Now,

$$\vec{n} \cdot \vec{P}_i + \lambda \cdot \vec{n} \cdot (\vec{P}_3 - \vec{P}_i) - \vec{n} \cdot \vec{a} = 0$$

From Fig. 1 ON PP. 44.

Let's make $\vec{a} = (0, 0, 0)$ on

$x_w y_w$.

$$\vec{n} \cdot \vec{P}_i + \lambda \vec{n} \cdot (\vec{P}_3 - \vec{P}_i) = 0$$

$$\lambda \vec{n} \cdot (\vec{P}_3 - \vec{P}_i) = - \vec{n} \cdot \vec{P}_i$$

$$\lambda = - \frac{\vec{n} \cdot \vec{P}_i}{\vec{n} \cdot (\vec{P}_3 - \vec{P}_i)} \quad \dots (5)$$

Now, Calculation

$$\begin{aligned} \lambda &= -\frac{\vec{n} \cdot \vec{P}_i}{\vec{n} \cdot (\vec{P}_s - \vec{P}_i)} \\ &= -\frac{(n_x, n_y, n_z) \cdot (x_i, y_i, z_i)}{(n_x, n_y, n_z) \cdot (x_s - x_i, y_s - y_i, z_s - z_i)} \\ &= -\frac{n_x x_i + n_y y_i + n_z z_i}{n_x (x_s - x_i) + n_y (y_s - y_i) + n_z (z_s - z_i)} \quad (5-b) \end{aligned}$$

Now, we can use (5-b) for

C/C++ Coding.

Sample code on the class github.

pp.38 on this Note.

Oct. 30 (Monday)

Note 1. Midterm exam is scheduled on Nov. 13 (Monday).

1. One hour exam, you may one page formula sheet is allowed.

Close book and close notes.

2. Format of the Exam:

(1) hand calculation, with CPU datasheet, SCH pdf of the CPU module; Either

LPC1769 (Core) and/or 11C24;

(2) MCU Xpresso ready with the prototype board, to be able to run the code and take screen captures of the execution, and photos of your prototype board, including the display of the LCD.

3. Submission of the work:

(1) 15 minutes for the prep for the CANVAS submission, which include:

a. screen captures to be converted to pdf;

b. photos:

- b. Photos of the prototype system and LCD display, to be converted to pdf;
- c. to integrate all the pdf into one pdf file.

Note: Please use adequate resolution, for example, 1024x768 or similar resolution.

4. Review session next week.

5. Scope of the midterm: from the introduction till 3D shadow calculation and coding.

Example: To build a shadow on x_w-y_w , by finding 4 intersection points.

Sol:

First, let us find the normal vector $\mathbf{N} = (n_x, n_y, n_z) = (0, 0, 1) \dots (1)$

Suppose we have a cube with side length = 100, and then

$P_i(x_i, y_i, z_i) = (50, 50, 100)$ from Fig. 1 on pp. 42.

Now make the cube float above the x_w-y_w plane by 10 units. Hence we have

$P_i = (50, 50, 110)$

Now, define a light source location $P_s(x_s, y_s, z_s) = (-50, 0, 250)$

Now, find lambda.

From equation (5-b), we step by step calcualtion, first,

$$n_x x_i + n_y y_i + n_z z_i = 0 + 0 + z_i \dots (2)$$

and

$$\begin{aligned} n_x (x_s - x_i) + n_y (y_s - y_i) &+ n_z (z_s - z_i) \\ &= 0 + 0 + 1 * (z_s - z_i) \\ &= z_s - z_i = 250 - 110 = 140 \end{aligned}$$

$$\begin{aligned}\lambda &= -z_i / (z_s - z_i) \\ &= -110 / 140\end{aligned}$$

Substitute the finding into the ray equation, (4-1) on pp. 44

$$\begin{aligned}R &= P_i + \lambda * (P_s - P_i) \\ &= (50, 50, 110) + \lambda * (-50 - 50, 0 - 50, 250 - 110) \\ &= (50, 50, 110) + \lambda * (-100, -50, 140) \\ &= (50, 50, 110) - 110/140 * (-100, -50, 140) \\ &= (50 - 110/140 * (-100), \\ &\quad 50 - 110/140 * (-50), \\ &\quad 110 - 110/140 * 140)\end{aligned}$$

Note the Z is equal to 0.
And it should be since it is on
the x_w - y_w plane.

Nov. 1st. (Wed).

Example: Reading the
Sample Code.

Ref.

2018F-118-13 diffuse ~

Note1: Define the Virtual Camera Location, $\vec{E}(x_e, y_e, z_e)$

```
34 float Xe = 200.0f, Ye = 200.0f, Ze = 250.0f; //virtual camera location
35 float Rho = sqrt(pow(Xe, 2) + pow(Ye, 2) + pow(Ze, 2));
36 float D_focal = 100.0f;
```

Note2: Define "struct" for the points
in x_w - y_w - z_w , x_e - y_e - z_e , and
Perspective projection

```

39     typedef struct {
40         float X[UpperBD], Y[UpperBD], Z[UpperBD];
41     } pworld;
42
43     typedef struct {
44         float X[UpperBD], Y[UpperBD], Z[UpperBD];
45     } pviewer;
46
47     typedef struct{
48         float X[UpperBD], Y[UpperBD];
49     } pperspective;

```

Note 3. Define Data points $P_i(x_i, y_i, z_i)$ for $x_w-y_w-z_w$.
 Origin $(0,0,0)$.

```

70 //define the x-y-z world coordinate
71 world.X[0] = 0.0;    world.Y[0] = 0.0;    world.Z[0] = 0.0;    // o
72 world.X[1] = 50.0;   world.Y[1] = 0.0;    world.Z[1] = 0.0;    // x
73 world.X[2] = 0.0;    world.Y[2] = 50.0;   world.Z[2] = 0.0;    // y
74 world.X[3] = 0.0;    world.Y[3] = 0.0;    world.Z[3] = 50.0;   // y
75 Reference purpose for the Decoration / Texture Mapping in 3D.
76 //define projection plane
77 world.X[4] = 60.0;   world.Y[4] = -50.0;  world.Z[4] = 0.0; //p4
78 world.X[5] = 60.0;   world.Y[5] = 50.0;   world.Z[5] = 0.0; //p5
79
80 world.X[6] = 60.0;   world.Y[6] = 50.0;   world.Z[6] = 100.0; //p6
81 world.X[7] = 60.0;   world.Y[7] = -50.0;  world.Z[7] = 100.0; //p
82

```

Note 3. The Sin, Cos and ρ for the 4×4 matrix, in the transformation pipeline

```

127     //sin and cosine computation for world-to-viewer
128     float sPheta = Ye / sqrt(pow(Xe, 2) + pow(Ye, 2));
129     float cPheta = Xe / sqrt(pow(Xe, 2) + pow(Ye, 2));
130     float sPhi = sqrt(pow(Xe, 2) + pow(Ye, 2)) / Rho;
131     float cPhi = Ze / Rho;

```

Note 4. Compute λ for the Ray Equation,

$$\vec{R} = \vec{P}_i(x_i, y_i, z_i) + \lambda (\vec{P}_s(x_s, y_s, z_s) - \vec{P}_i(x_i, y_i, z_i))$$

where

$$\lambda = -\frac{\vec{n} \cdot \vec{P}_i}{\vec{n} \cdot (\vec{P}_s - \vec{P}_i)}$$

```

139     //----- lambda for Intersection pt on xw-yw plane-----*
140     float temp = (world.X[47] * (world.X[46] - world.X[45]))
141             +(world.Y[47] * (world.Y[46] - world.Y[45]))
142             +(world.Z[47] * (world.Z[46] - world.Z[45]));
143     float lambda = temp / ((world.X[47] * (world.X[45] - world.X[7]))
144                             +(world.Y[47] * (world.Y[45] - world.Y[7]))
145                             +(world.Z[47] * (world.Z[45] - world.Z[7])));
146     float lambda_2 = temp / ((world.X[47] * (world.X[45] - world.X[6]))
147                             +(world.Y[47] * (world.Y[45] - world.Y[6]))
148                             +(world.Z[47] * (world.Z[45] - world.Z[6])));

```

Note 5. Based on \sin, \cos, ρ value Computed in the previous lines of code, Now, Compute the 4×4 matrix for the 1st Step of the Transformation Pipeline

$x_w - y_w - z_w \rightarrow$ (mapping) $x_e - y_e - z_e$.

```

240     //----- Transformation pipeline-----*
241     for(int i = 0; i <= UpperBD; i++)
242     {
243         //-----world to viewer-----*
244         viewer.X[i] = -sPheta * world.X[i] + cPheta * world.Y[i];
245         viewer.Y[i] = -cPheta * cPhi * world.X[i]
246             - cPhi * sPheta * world.Y[i]
247             + sPhi * world.Z[i];
248         viewer.Z[i] = -sPhi * cPheta * world.X[i]
249             - sPhi * cPheta * world.Y[i]
250             -cPheta * world.Z[i] + Rho;

```

x_e

y_e

z_e

Note b. Now Perspective Projection.

$$\begin{cases} x_p = x_e \frac{D}{z_e} \\ y_p = y_e \frac{D}{z_e} \end{cases}$$

Note a primitive Clipping feature is implemented here.

```

252 //-----perspective projection-----
253 perspective.X[i] = D_focal * viewer.X[i] / viewer.Z[i];
254 perspective.Y[i] = D_focal * viewer.Y[i] / viewer.Z[i];
255 if (perspective.X[i] > xMax) xMax = perspective.X[i];
256 if (perspective.X[i] < xMin) xMin = perspective.X[i];
257 if (perspective.Y[i] > yMax) yMax = perspective.Y[i];
258 if (perspective.Y[i] < yMin) yMin = perspective.Y[i];
...

```

Homework (On Shadow Computing).

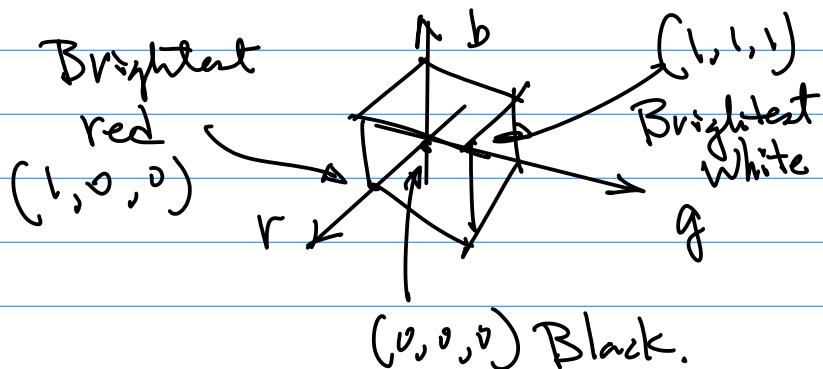
1. Implement Shadow Computation Algorithm.
By plotting A 4-Point Shadow. Based on your previous Homework.

2. Modify the Cube in the previous homework By elevating the Cube 15 units.

3. Color for the shadow, use dark blue.

r, g, b Settings:

- a) 8 bits for Each Color.
- b) Define r, g, b Combination to form Dark Blue.



255 for Normalized 1
0 for Normalized 0.



Due ON Sunday Nov.12 (11:59 p.m)