CMPE240
Fall 2023

1/

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 undergradua
documentation of having satisfied the class prerequisite requireme
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 Deeplearning, AI etc.

4. Hands-on.

Datasheets. → Spec. → Hardware Prototype Board With Color LCD Display ← 2D Graphics Engine

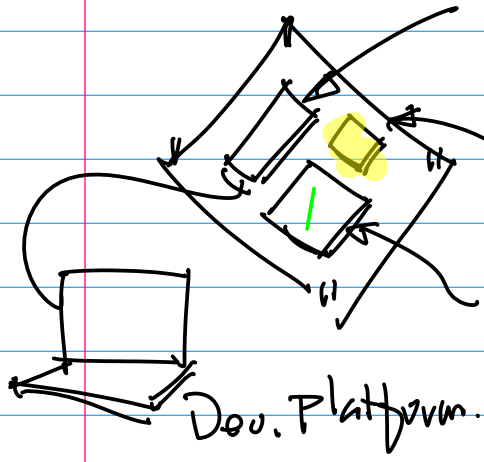
↓

CmpE240 Fall 2023

3D Graphics Engine

↓

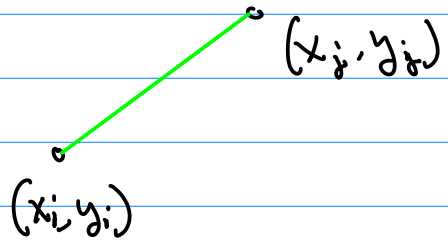
Benchmarking (with Ref. to NVDA Jetson NANO)



Live 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~~ posted on CANVAS, with Written Requirements. Those are the material to be Submitted.

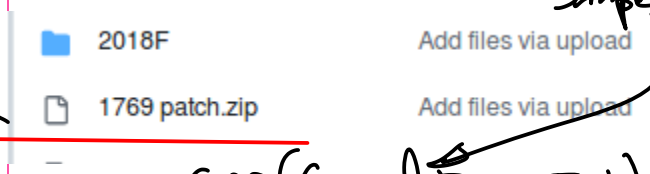
5. PPTs, Lecture Notes (White Board Notes), Datasheet(s), are posted on the github.

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.

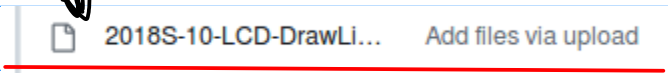


GPP (General Purpos Port)

Target CPU
NXP LPC11C24
LPC1769

The Code was for LPC17ba, But Newer Samples for GPP, Graphics Display for LPC11C24 were developed and posted on the github.

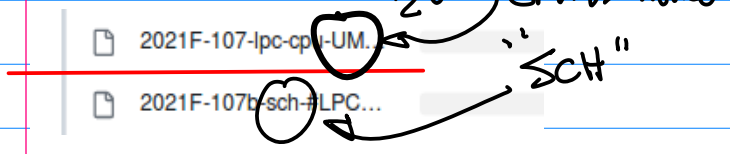
Next Level of the Sample Code



The Lower After Sample code for 2D & 3D Engine Design.

PPT Material in pdf.
will be used in the Class.

Datasheet. Note: 1° CPU Datasheet is in CMPE244 folder



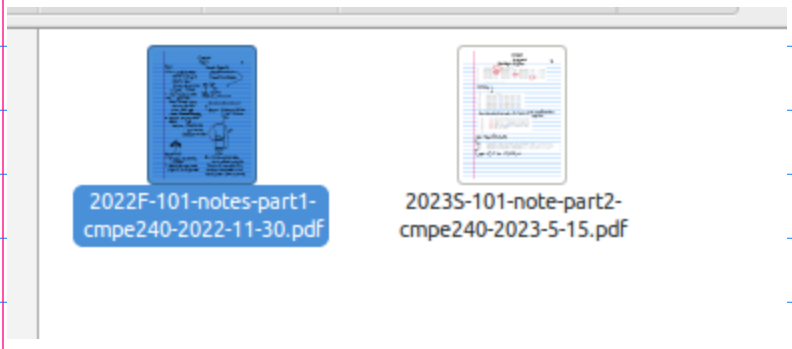
Grading Information

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

August 23rd (Wed)

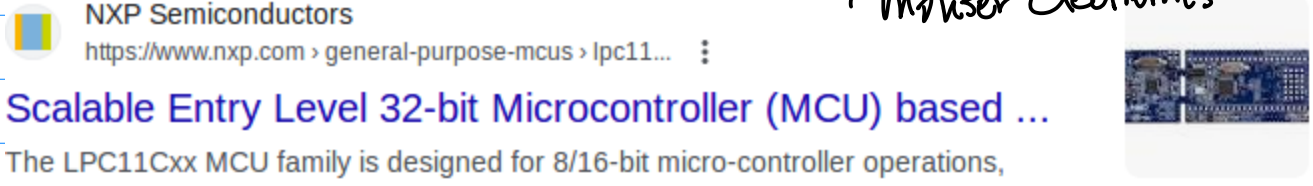
Announcement:
1° Lab Space BM268
2° CANVAS to be up by this week.

Introduction.



Example: Architecture of LPC1769 (LPC11C24)

Can be purchased from digi-key.com or Mouser electronics



CmpE 240
Fall 2023

5/

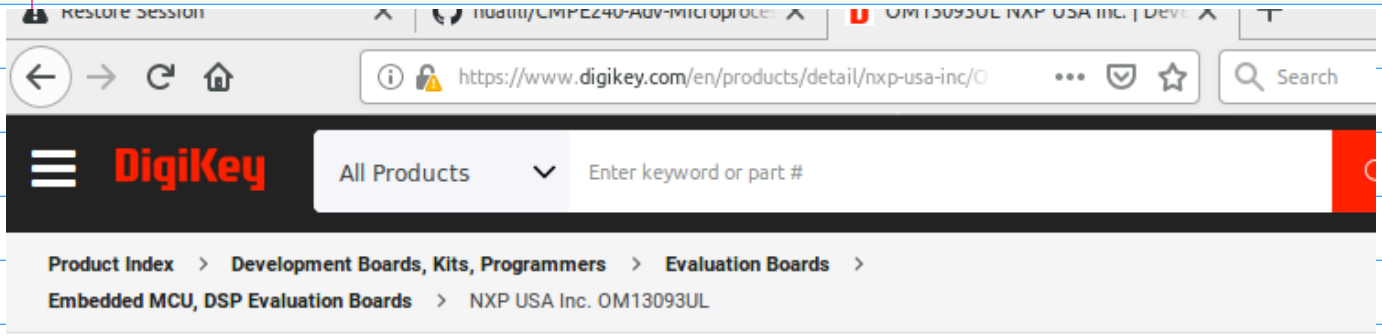


Image shown is a representation only. Exact specifications should be obtained from the product data sheet.

OM13093UL

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®-M Evaluation Board
Customer Reference	Customer Reference

Note: Please start the Purchasing Process.

Note: CPU Datasheet.

2011

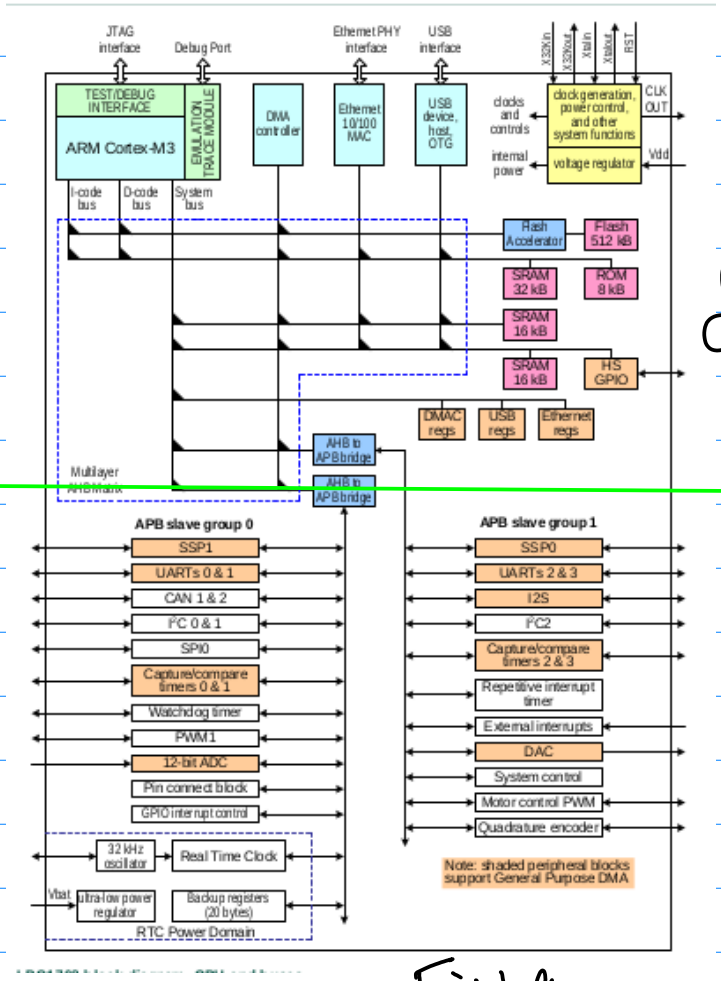


Fig 1.a

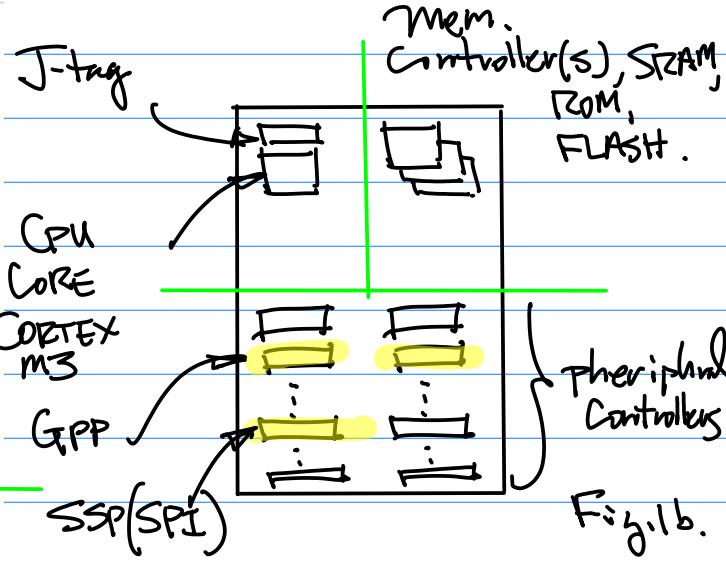


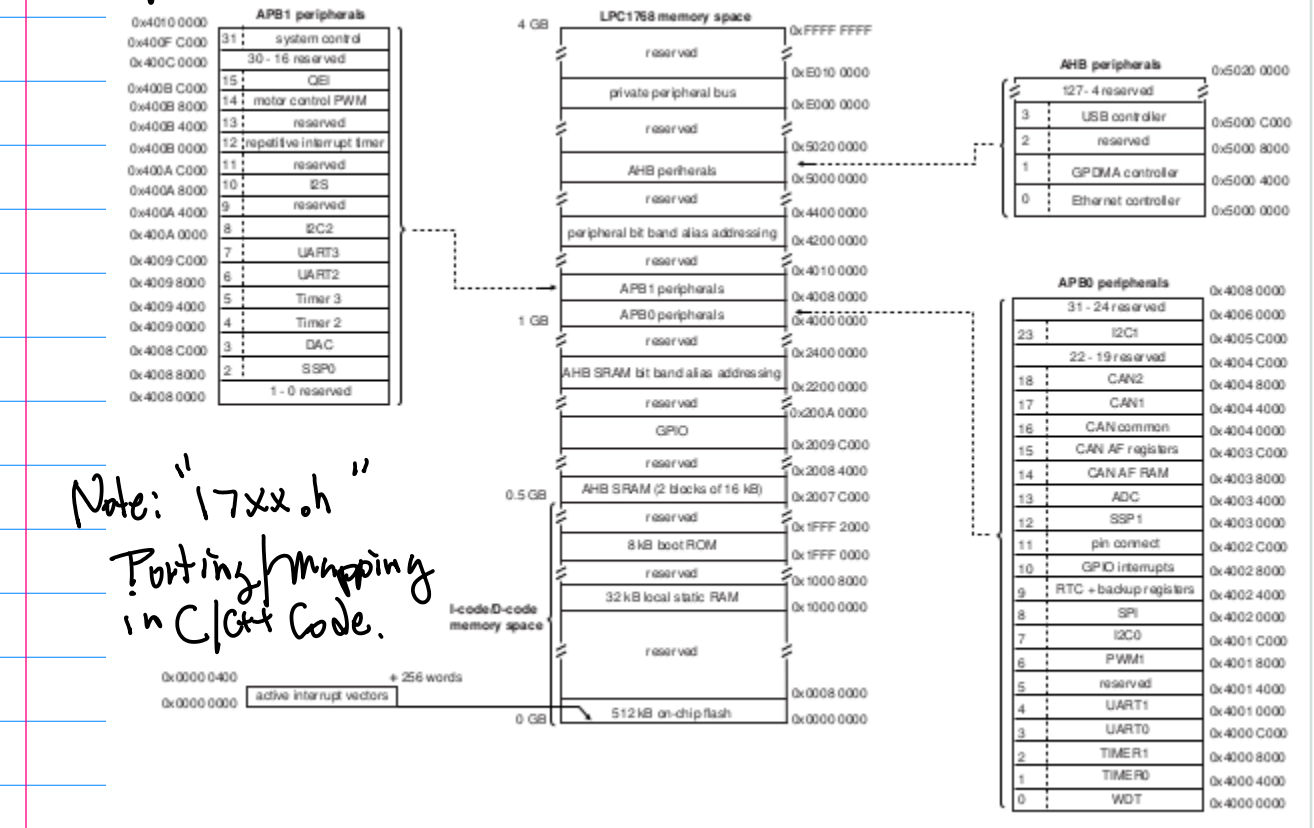
Fig 1.b

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).

Memory Map.



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

CmpE240
F2023

Note: For the memory map Discussion:

1° RISC: Reduced Instruction Set Computer.
- ARM.
- MIPS
  - Golden Rules:
    - Uniformity,
    - Regularity,
    - Orthogonality)

2° 32Bit RISC Processor → 32bit Architecture
- 32bit Addr. Bus.
- 32bit Data Bus.
- 32bit R.F. (Register File)
  - GPRs (General-Purpose Registers) 32bits.
  - SPRs (Special Purpose Registers)
    - 3 Cats.

↓

32bit memory map.

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

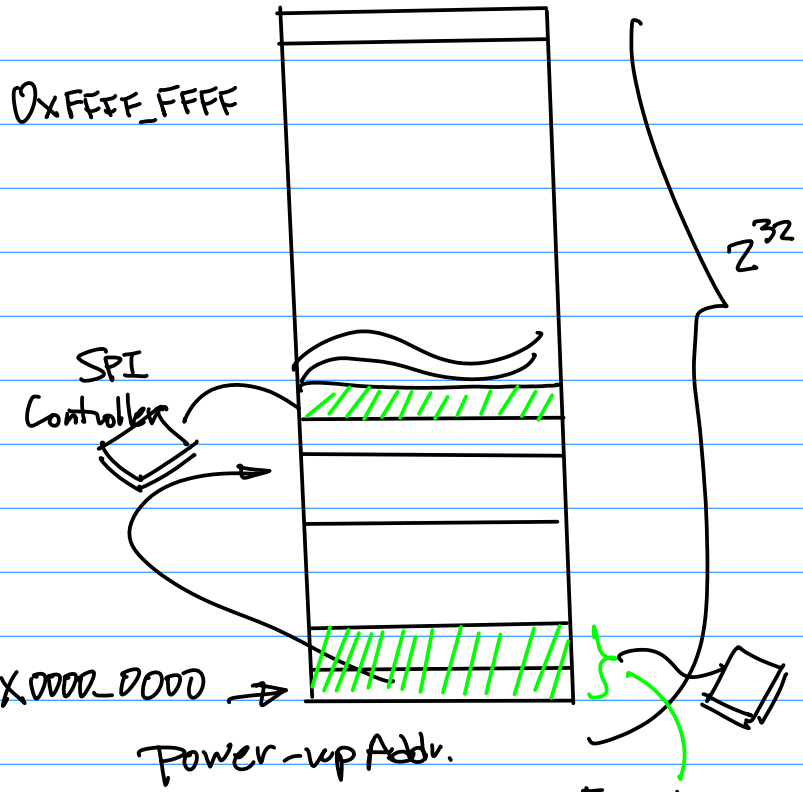
1K (1024) ~ 1M (1K x 1K) ~ 1G

4G? Byte!

3° Byte Addressable Machine.

A smallest memory cell with an Unique Address is a Single Byte.

4°



August 28 (Monday).

Note: 1° CANVAS is up.

2° CPU module & LCD module

ST7735 Controller

SPI – Interface

https://www.amazon.com/Display-Module-ST7735-128x160-STM32/dp/B07BFV69DZ

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

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

Roll over image to zoom in

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.

$10.99

With **Amazon Business**, you would have **saved $85.08** in the last year. Create a free account and **save up to 5%** today.

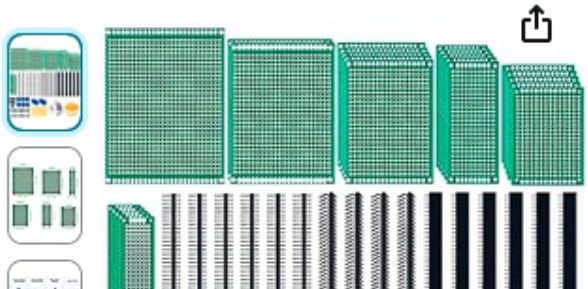
Brand	Walfront
Personal computer design type	All in One
Operating	Linux

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

1°. CPU module
2°. LCD module
3°. Wire Wrapping Board.

4"x3" 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.

Divid 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	

$a_{31}a_{30}\cdots$ $a_{29}a_0$ ← LSB

MSB

32 bits Addr. Bus

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

Memory Bank with a Starting Addr. 0x4000_0000 → 3rd BANK (BANK0, BANK1, BANK2)

Find A SPI Block

9	RTC + backup registers	0x4002 4000
8	SPI	0x4002 0000
7	I2C0	0x4001 C000

SPI Peripheral Controller is Located at 0x4002_0000

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

Note: "Little Endian" Convention

Choose $a_{31}a_{30}a_{29}$ to Identify the memory Bank.

$a_{31}a_{30}a_{29}0;0000;\ldots;0000$ Lowest Addr.

$a_{31}a_{30}a_{29}1;1111;\ldots;1111$ Highest Addr.

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

512 | meg.

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

0x4002_4000
− 0x4002_0000
0x4000

LSB

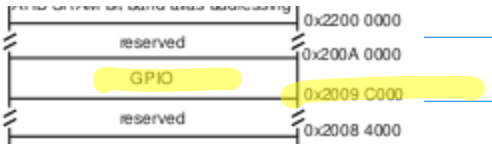
MSB 0100:0000:0000:0000

$2^x = 1 \cdot 2^{14} = 1 \cdot 2^4 \cdot 2^{10}$

16 . K

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

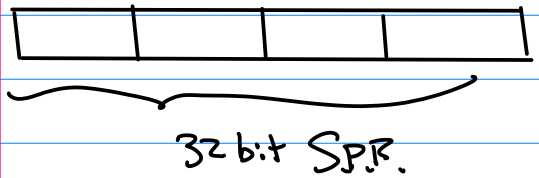
↓

SPRs (Special Purpose Registers)

Responsible for Init & Config.

↓

Control Register.



32 bit SPR.

Naming Convention "3+3" for All if Possible

Prefix + Root 3 Letters
3 Letters

August 30 (Wed)

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

2° Bill of Material.

Ref: ON the github of CmpE240 20185-2_... Bill_of_Material

3° Homework (0 pt) Due Sept. 10 (Sun)

Installation of NXP MCU Expresso.

Submission: Screen Capture that Shows the MCU Expresso. + Personal Identifier.

Ref: github. → PPT for MCUXpresso Configuration.
{ NXP Developer Forum, pdf

Note:

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

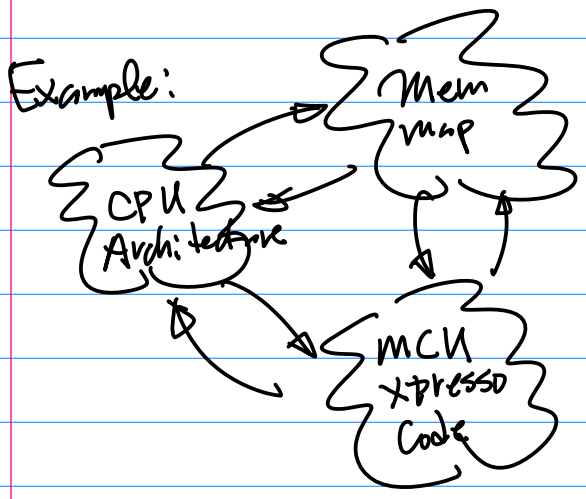
Open Source

4° Please bring the CPU module to the class for inspection & Discussion.

CmpE240
F2023

11/

Example:



Design of GPP

GPx CON

Prefix — Root

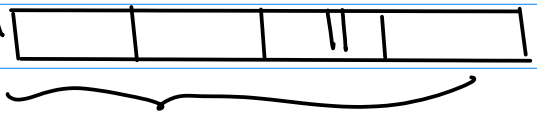
x: Allow the user to Define which port.

Note: Multiple I/O pins possible for Each GPx

GPACON

0X uuuu-uuuu

0X2009_C000



32 bit SFR.

GPA Port Pin 5 as an Output pin → Need to Identify the Bit(s) in GPACON for the selection Purpose.

Sept. 6 (Wed)

Example: Inspection of LPC1114 OR LPC1769.

Purpose: Identify Pin 1 ON the module. Match it up to Sch of the Board module

Ref: On the github LPC1114 AND LPC1769.

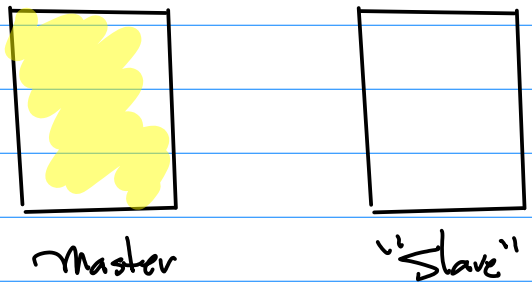
Note: 1° Physical pin assignment, e.g. pin 1, pin 2, ..., etc.

2° Namings of the pins
- a. Connector Related
- b. CPU 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).



CmpE240

F2023

12/

"3+1" pins

- MOSI (Master Out/Slave In)
- MISO (" In/" Out)
- SCK (Serial CLK)

} "3"

- EN/nEN Enable Active high or Active Low.

J6 or J2, 40 pins Connector.

Header Connector → male

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

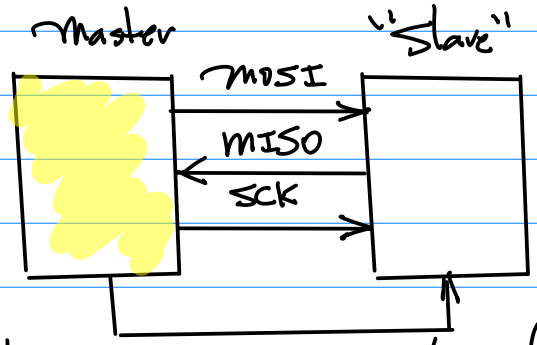


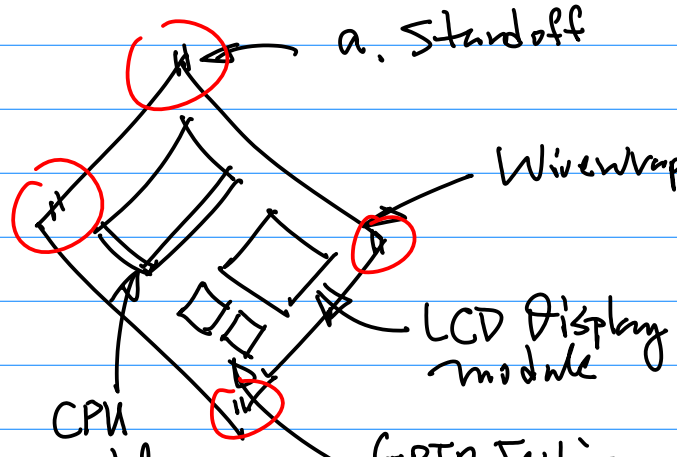
Fig.1.

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

~~c. Soldering the LCD module.~~

Sept. 11 (Monday).

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



b. "Mount" → Solder the CPU pins

Example: SPR for SSP0.

Ref: github/hvedrih/Cmpe244~2021F-107 or

PP[3:1]. a. 32bit SPR.

b. Name: SSP0CR0

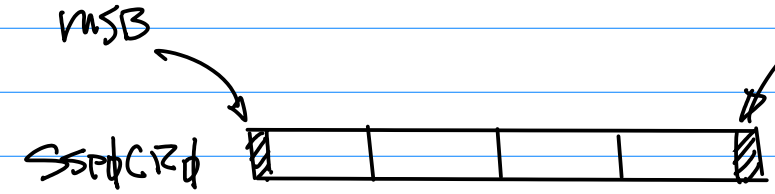
for the peripheral controller0. (multiple controllers)

Prefix + Rest

CmpE240

Fall 2023

1 pt. 13/



SSP0CR0

0x4008_8000

DSS

SSP0CR0[3:0] = 0111 for 8 bits Transfe

SSP0CR0[5:4] = 00 for S.P.I.

SSP0CR0[6] = 0, SSP0CR0[7] = 0 By default

SSP0CR0[15:8], SCR. Serial clock Rate

CR0

8 bit → 2^8 = 256 → [0, 255]

SysCLK (System Clock)
CPU Clock.

PCLK (peripheral clock)
1/2, 1/4, 1/8, ...

SCR Controls S.P.I. Clock.

Range to work with to define Serial clock. SCK.

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

from a SPR

[2, 255]

254 ?

[0, 255]

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 KHz (e.g. → 5 Kbps)

(3) Design By Assigning SCR to Realize the Bit Rate Requirement.

Sept. 13 (Wed)

[Note], Inspection of the prototype Board (Work-In-Progress)

Sol: From Eqn (1), pp 431.

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

from the given condition, we have

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

Design By Iteration.

First, Let CPSDVSR = 4
Solve for SCR

$$CPSDVSR * (SCR+1) \Big|_{CPSDVSR=4} = \frac{20 \times 10^6}{5 \times 10^3}$$

$$SCR+1 = 4 \times 10^3 / 4$$

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

So, the Next Iteration of the Design

Let CPSDVSR = 32
from Eqn (1), we have

$$CPSDVSR * (SCR+1) \Big|_{CPSDVSR=32} = 4 \times 10^3$$

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

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

CPSDVSR = 32 and
SCR = 124
⇓
124 in Binary format.

CmpE240

F2023

15

C-Code / MCU Xpresso.

Note1. Projects Imported into the MCU Xpresso. (see I)

Note2: Import 1769.zip. (from the Class git) By using II. (GPIO project as Ref.)

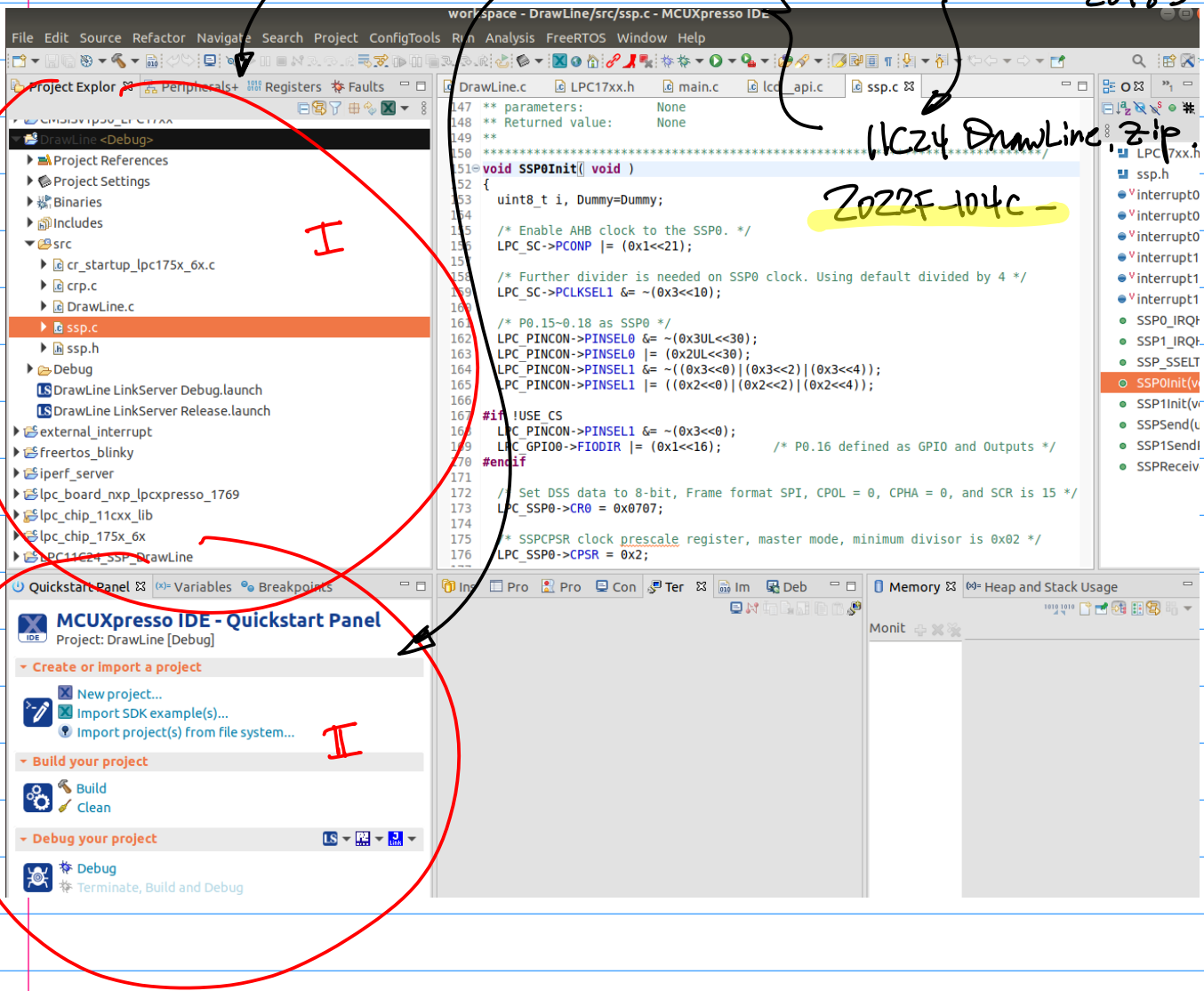
Note3:

DrawLine.zip.

LPC24 DrawLine.zip.

2018S-10-n

2022F-104c



Step1. Config the MCU Xpresso.

CPU

Board

Ref: Class PPT OR NXP

Step2: Import LPC1769. patch. "zip".

Step3. Import DrawLine Project (1769)

Step4 Import DrawLine Project (LPC24)

(x_{i+1}, y_{i+1})

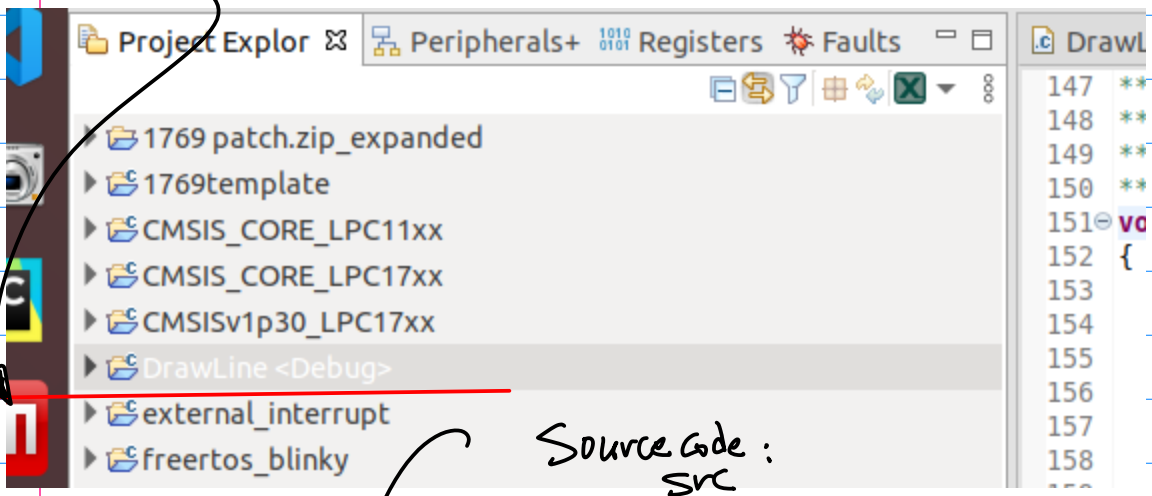
(x_i, y_i)

CMPE240

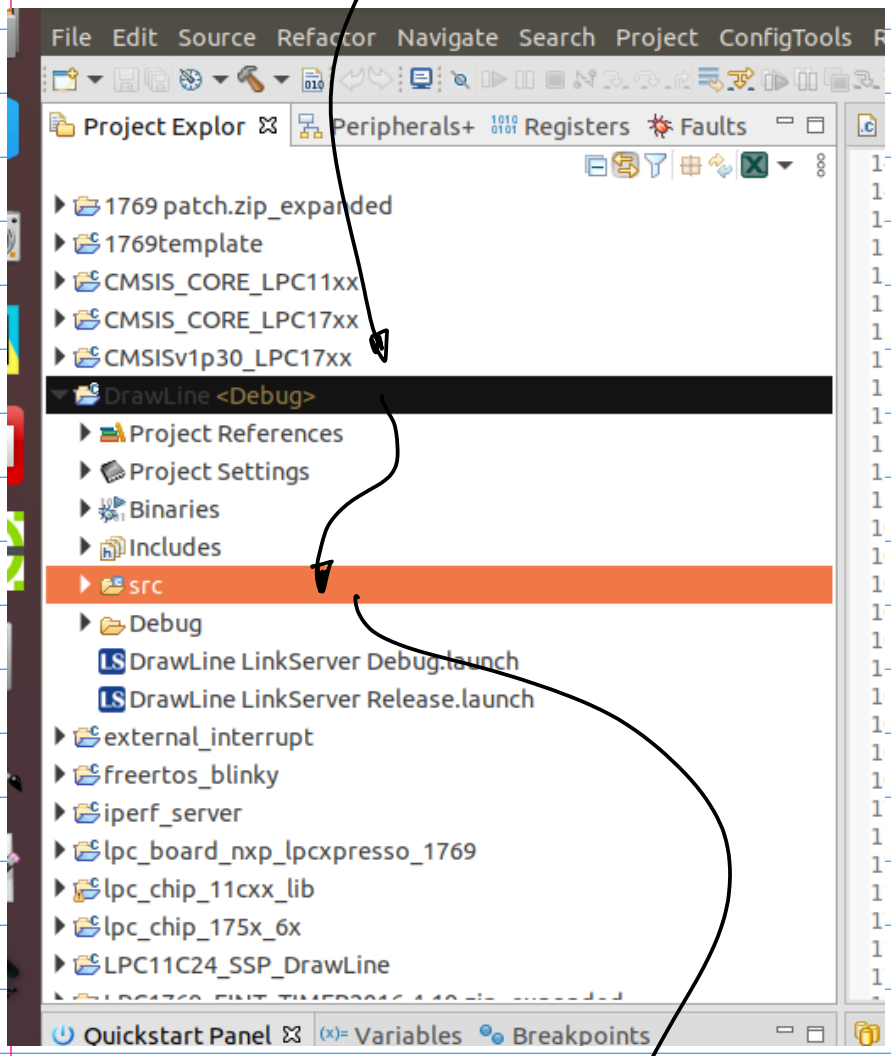
F2023

16/

Note1. Sample Project for 1769. A Starting Point.



Source code : src



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;
```

Note: please Read this code!

SPI. Init & Config

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

Note: Code ⇄ Tech. Spec.

0x0707

0000 0111 0000 0111 → Datasheet.

Sept. 18 (Monday).
Please Check CANVAS for the Homework (Prototype Board).

Example: LCD Display Pin Connectivity.
Ref: github/hualili/Cmpe240/2022f-103f-~

Fig. 1

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

Table 5. Connectivity Table of LPC11C24 and LCD

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	A0
6. J6-5	MOSI0	SDA
7. J6-7	SCK0	SCK
8. J6-28	3VOUT	LED

LPC11C24 Connectivity Table t

a) 8 bit mode
b) SPI Interface
c) Clock setting is default.
d) SCR (Equ.-1)

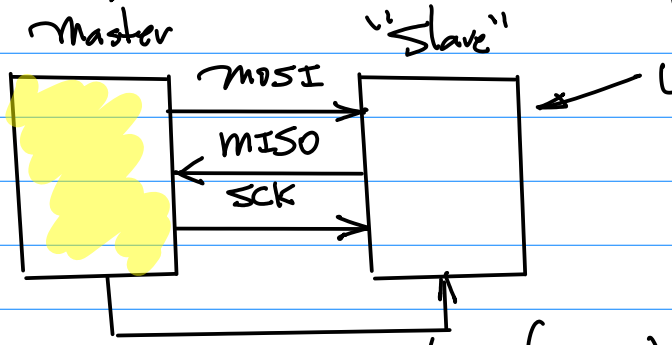
$$f_{SPI} = \frac{PCLK}{DVSR(SCR+1)}$$

SCR=7.

find f_{SPI}, Hz

Note: Toggle Between the Commands and Data

LPC11C24 OR LPC1769 (Different Connection!)



LCD Display module.

Fig.2

Note: One Clock Corresponds to ONE Bit.

Example: Bit Rate 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( ) (line151).

Note: SPI Data Communication.
Waveform Captured By Logic Analyer. from MOSI pin.

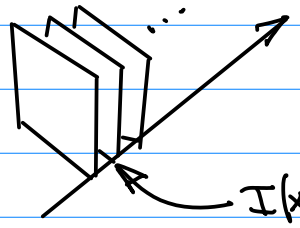
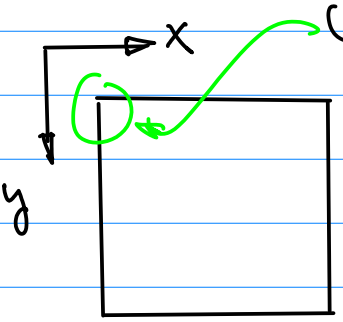


Fig.4

Given Graphics Display Resolution M x N.

Physical coordinate (0,0)



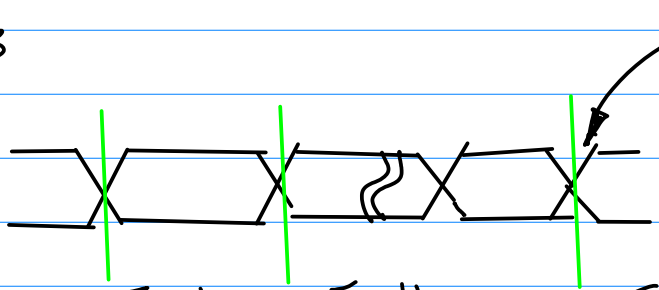
M x N

Col. Row

No. of Pixels per Row. No. of Rows.

Assuming the Resolution of the LCD is M x N (160 x 120).
Frame Rate (FPS) 30

Fig.3



Find the Bit Rate for SPI Interface.

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

- 160 × 120: Total No. of the pixels per frame
- 24: Bit/pixel (pixel Depth)
- 30: FPS

$\overset{2^x}{=}$

$160 \times 120 \times 24 \times 30$

$2^7 \quad 2^7 \quad 2^5 \quad 2^5 = 2^{14} \cdot 2^{10} = 2^4 \cdot 2^{10} \cdot 2^{10} =$

160 → $2^7 = 128$

16 → $2^8 = 256$

1K → 1meg

$\therefore$ 16Mbps

Therefore, the SPI Bit Rate is adequate.

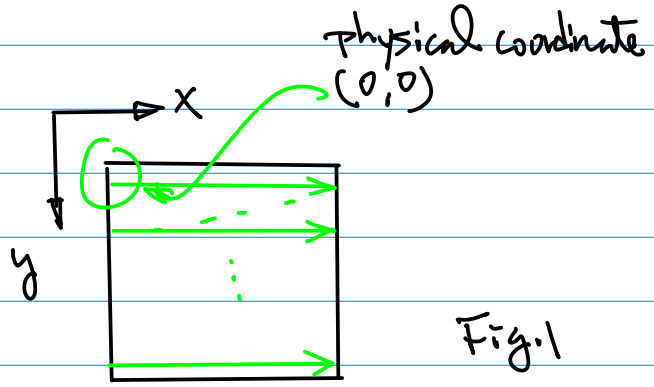


Fig.1

M x N

Col. Row

Progressive Scaning

2° Virtual Coordinate System.

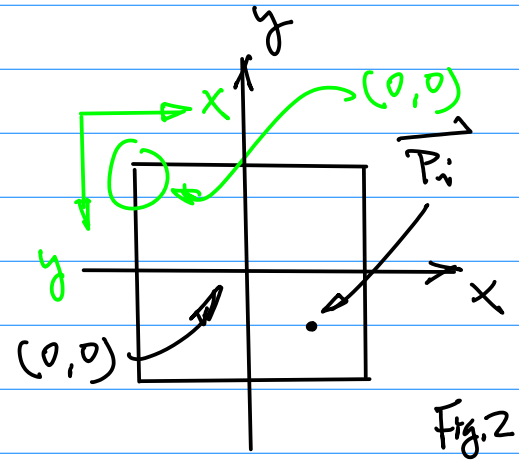


Fig.2

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

$\lambda = 1/4$ pt.

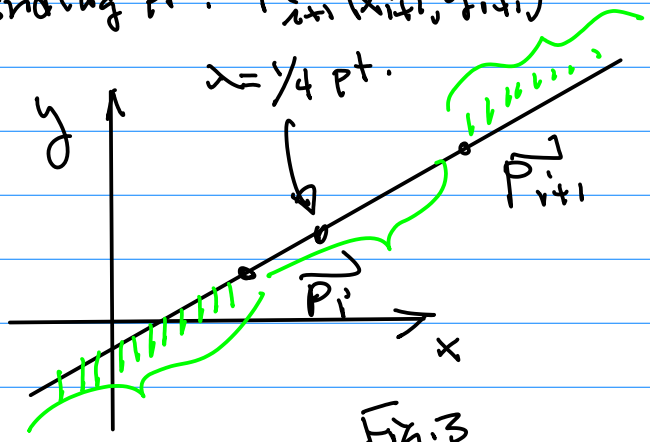


Fig.3

Let's Define the Line in Fig.3.

First, Define a Directional Vector.

$$\vec{d}(x,y) = \vec{P}_{i+1}(x_{i+1}, y_{i+1}) - \vec{P}_i(x_i, y_i)$$
$$= \vec{P}_{i+1} - \vec{P}_i$$
$$= (x_{i+1}, y_{i+1}) - (x_i, y_i)$$
$$= (x_{i+1} - x_i, y_{i+1} - y_i) \quad \cdots (2)$$

Line Equation

$$\vec{P}(x,y) = \vec{P}_i(x_i, y_i) + \lambda \vec{d}(x,y)$$
$$= \vec{P}_i(x_i, y_i) + \lambda\left(\vec{P}_{i+1}(x_{i+1}, y_{i+1}) - \vec{P}_i(x_i, y_i)\right) \quad \cdots (3)$$

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.

" $0 \le \lambda \le 1$, 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 $\vec{P}_i(2,3)$, and ending point $\vec{P}_{i+1}(7,9)$.

Find:
1) Directional vector $\vec{d}(x,y)$
2) Find the Line Equation $\vec{P}(x,y)$.
3) Find λ that defines 1/4 of the distance from the pt. $\vec{P}_i(x_i, y_i)$.

Sol:
(1) From Eqn (2),

$$\vec{d}(x,y) = \vec{P}_{i+1}(x_{i+1}, y_{i+1}) - \vec{P}_i(x_i, y_i)$$
$$= \vec{P}_{i+1}(7,9) - \vec{P}(2,3)$$
$$= (7-2, 9-3)$$
$$= (5,6)$$

C/C++ Code

```
direc.X[i] = pt[i+1].x - pt[i].x
```

(2) Line Eqn

$$\vec{P}(x,y) = \vec{P}_i(x_i, y_i) + \lambda\left[\vec{P}_{i+1}(x_{i+1}, y_{i+1}) - \vec{P}_i(x_i, y_i)\right]$$
$$= \vec{P}(2,3) + \lambda\left[\vec{P}(7,9) - \vec{P}(2,3)\right]$$
$$= (2,3) + \lambda(7-2, 9-3)$$
$$= (2,3) + \lambda(5,6)$$

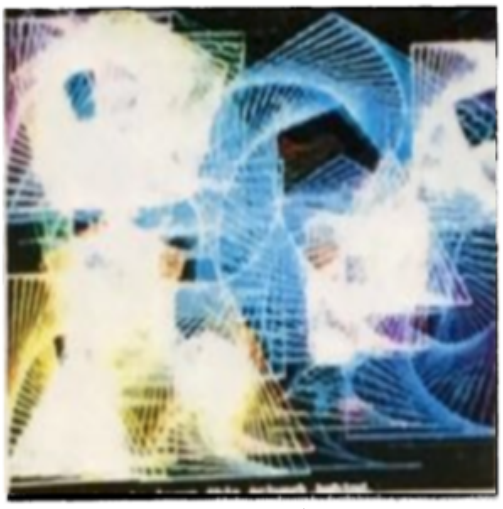
(3) Since 1/4 of the distance from $\vec{P}_i$, so Let $\lambda = 1/4$

Find that 1/4 pt. from

$$\vec{P}(x,y) = (2,3) + \lambda(5,6)\Big|_{\lambda = 1/4}$$
$$= (2,3) + \frac{1}{4} * (5,6)$$
$$= \left(\frac{8}{4} + \frac{5}{4}, \frac{12}{4} + \frac{6}{4}\right)$$
$$= \frac{1}{4}(13, 18)$$

Sept 25 (Monday)

Example: Prep. for Project 1 Design



CmpE240
F2023

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

$$\{\vec{P_i}(x_i, y_i) \mid i = 1, 2, \cdots, 4\}$$

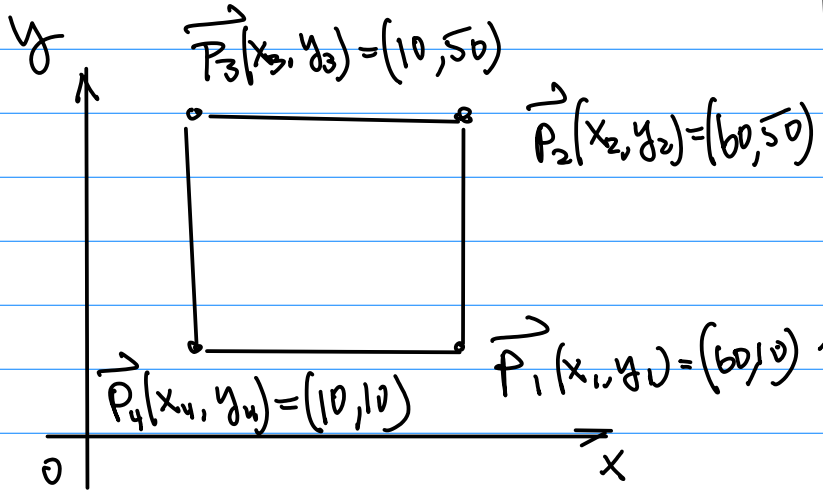
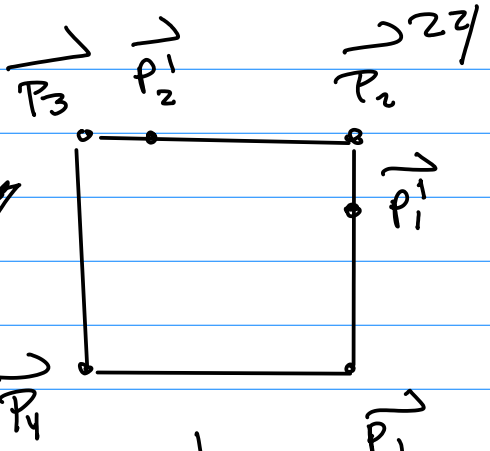
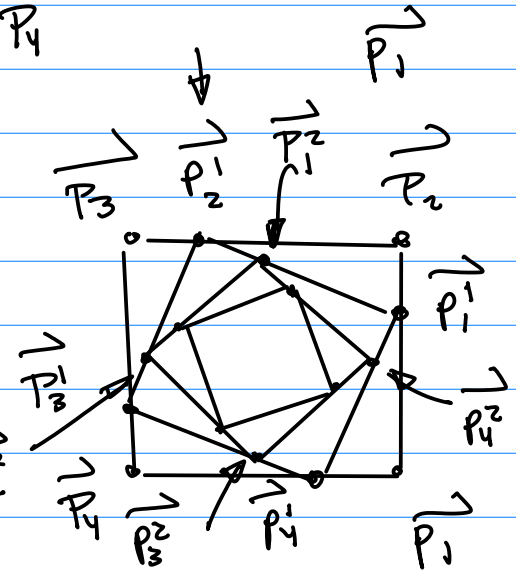


Fig. 1 Virtual Coordinate System

Note: $\vec{P_i}$, for $i = 1, 2, \cdots, N$, is arranged in a Counter Clockwise direction. As a Convention. (For 3D Hidden Line/Hidden Surface Removal).

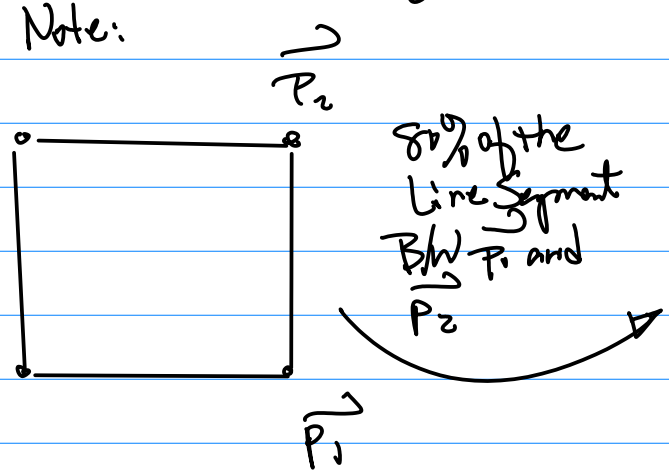




Note: 80% pts produces a Square of ~~Counter~~ Clockwise Rotation !!

From Eqn (3) on pp. 20,

Note:



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

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

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

"mod" operator

so, $(i+1) \bmod (5) = 1$

Consider Adding Superscript to define the levels.

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

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

Note: For the Project, make $j \geq 10$

Coding Aspects:

$$x_i^{j+1} = x_i^j + \lambda\left(x_{i+1}^j - x_i^j\right) \quad \cdots (2a)$$

Similarly,

$$y_i^{j+1} = y_i^j + \lambda\left(y_{i+1}^j - y_i^j\right) \quad \cdots (2b)$$

```
x[j+1][i] = x[j][i] + lamda*
            (x[j][i+1] - x[j][i]);
```

Example: CodeSample, github, drawline

Graphic Controllers Driver Program, 2D-trinity 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_q, y_q). Ending pt.
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.

Fig.1

Construct/Generate the Screen Saver.

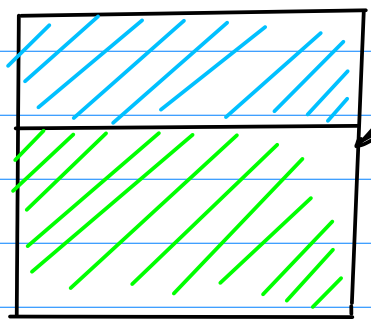


Fig.2

A Single Tree.

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

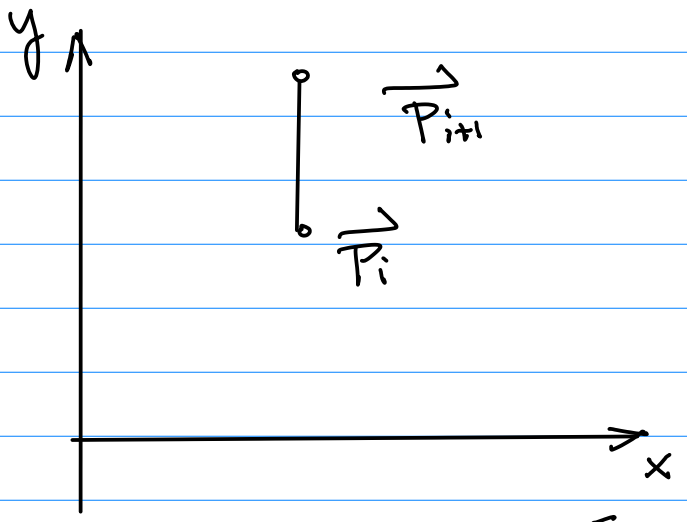


Fig.3

Then, Create Branches.

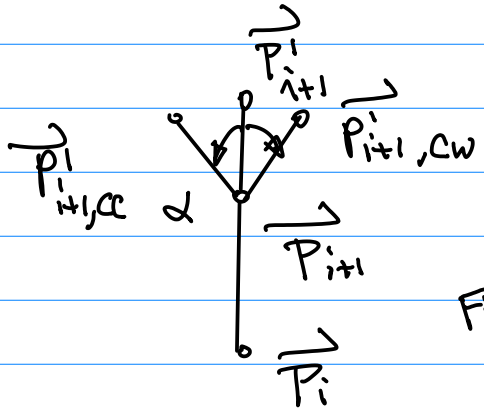


Fig.4

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

$$\vec{P} = \vec{P_i} + \lambda(\vec{P_{i+1}} - \vec{P_i})$$ for Example.

OR

$$\vec{P} = \vec{P_{i+1}} + M(\vec{P_{i+1}} - \vec{P_i})$$

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 $\vec{P}'_{i+1,CW}$

Next, Continue the process to grow the level for each and every end points

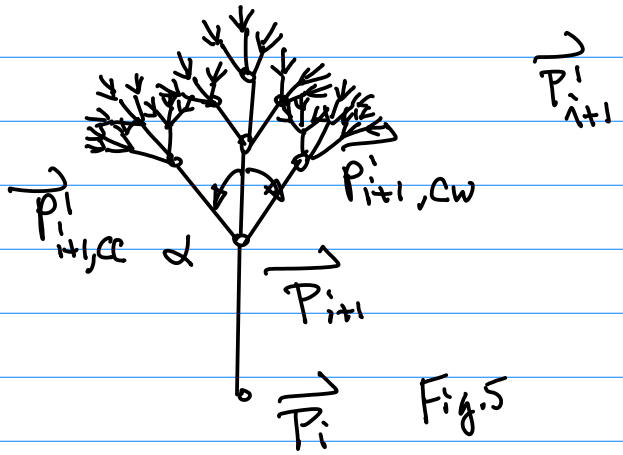


Fig.5

Notations: Use the Ending points' subscript, then increment the Level, e.g., the superscript by 1.

Consider 2D Transformations. { Translation

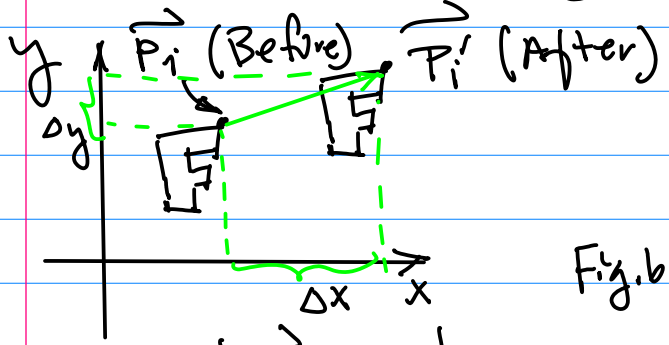


Fig.6

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

Translation.

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

$$x'_i = x_i + \Delta x \quad \cdots (1a)$$

Similarly, for y:

$$y'_i = y_i + \Delta y \quad \cdots (1b)$$

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,

$$\begin{pmatrix} x'_i \\ y'_i \\ 1 \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 0 & \Delta x \\ 0 & 1 & \Delta y \\ 0 & 0 & 1 \end{pmatrix}}_{3\times 3} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} \quad \cdots (1c)$$

Rotation:

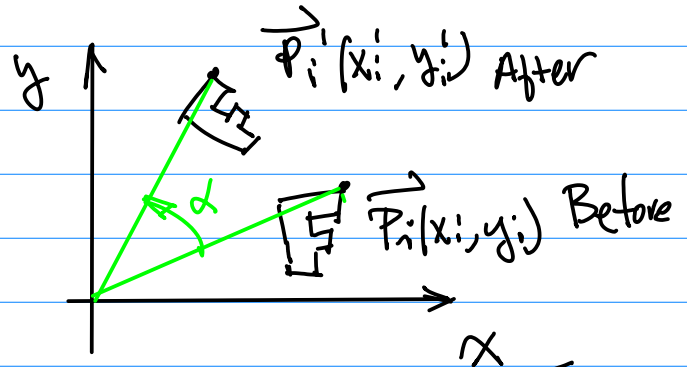


Fig.7

α: CounterClockwise → α > 0

$$\begin{pmatrix} x_i' \\ y_i' \\ 1 \end{pmatrix} = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} \quad \cdots (2)$$

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

Let's Translate $\vec{P}_{i+1}$ to (0,0). (the origin)

Step1. Translation

$$\begin{pmatrix} x_{i+1}' \\ y_{i+1}' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & \Delta x \\ 0 & 1 & \Delta y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_{i+1} \\ y_{i+1} \\ 1 \end{pmatrix}$$

After — Before

where $\Delta x = -x_{i+1}$

$\Delta y = -y_{i+1}$

Rotation

$$\begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_{i+1}' \\ y_{i+1}' \\ 1 \end{pmatrix}$$

$$= \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & \Delta x \\ 0 & 1 & \Delta y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_{i+1} \\ y_{i+1} \\ 1 \end{pmatrix}$$

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

Oct, 2nd (Monday).

Note1. 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 (2), 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(4Digits)_240.zip

Note 2: Project 1 is Due 2 weeks.
Oct 15 (Sun), 11:59 pm
Requirements to Be Posted on CANVAS.

Example: Continuation from

$$R_{3\times3} \cdot T_{3\times3} \quad \dots (1)$$

Rotation — Preprocessing

We need Postprocessing.

PostProcessing: To Undo the Preprocessing.

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

Inverse

physical meaning of T^{-1} for undoing Translation.

a) Translation matrix.
by making Δx to $-\Delta x$

b) Similly for the y. Change Δy to $-\Delta y$

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

$$\begin{pmatrix} x_i'' \\ y_i'' \\ 1 \end{pmatrix} = T^{-1} R T \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} \quad \dots (3)$$

After — Before

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

$$= \begin{pmatrix} 1 & 0 & -\Delta x \\ 0 & 1 & -\Delta y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\alpha & -\sin\alpha & \cos\alpha\cdot\Delta x - \sin\alpha\Delta y \\ \sin\alpha & \cos\alpha & \sin\alpha\cdot\Delta x + \cos\alpha\cdot\Delta y \\ 0 & 0 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} \cos\alpha & -\sin\alpha & \cos\alpha\cdot\Delta x - \sin\alpha\cdot\Delta y - \Delta x \\ \sin\alpha & \cos\alpha & \sin\alpha\cdot\Delta x + \cos\alpha\cdot\Delta y - \Delta y \\ 0 & 0 & 1 \end{pmatrix} \quad \cdots (8).$$

Therefore, we have

$$\begin{cases} x_i'' = \cos\alpha\cdot x_i - \sin\alpha\cdot y_i + \cos\alpha\cdot\Delta x - \sin\alpha\cdot\Delta y - \Delta x & \cdots (4a) \\ y_i'' = \sin\alpha\cdot x_i + \cos\alpha\cdot y_i + \sin\alpha\cdot\Delta x + \cos\alpha\cdot\Delta y - \Delta y & \cdots (4b) \end{cases}$$

Example: Drawline Code.
Emphase on the Graphics Library functions, e.g.

Primitive graphics Routines.

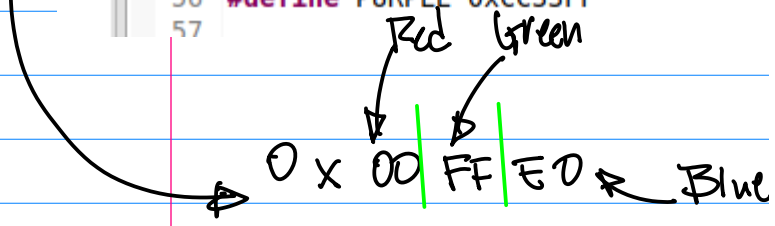
Note: Resolution

```
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
```

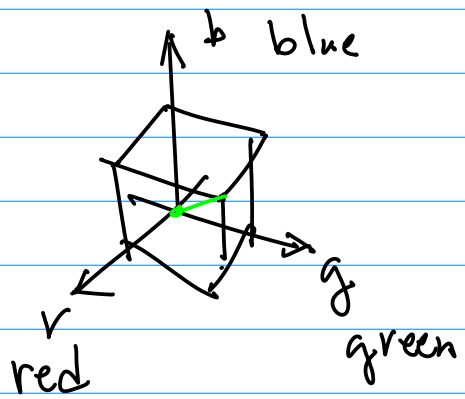
Col.
Row.

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 0xFFFFFF
56 #define PURPLE 0xCC33FF
57
```



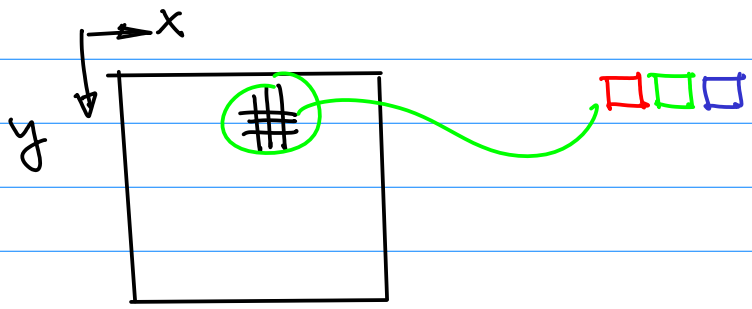
r,g,b primitive Color is used to define colors



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
128  uint8_t red, green, blue;
129
130  int i;
131
132  red = (color >> 16);
133
134  green = (color >> 8) & 0xFF;
135
136  blue = color & 0xFF;
137
138  for (i = 0; i< repeat; i++) {
139
140   writedata(red);
141
142   writedata(green);
143
144   writedata(blue);
```

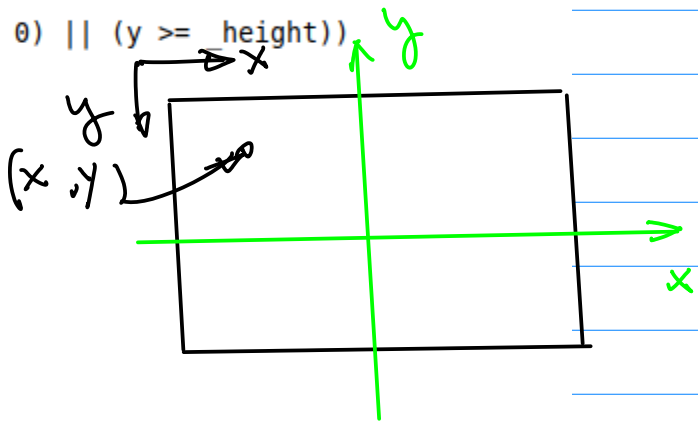


Oct.4 (Wed).

Example: Continuation on the ST7735 Graphics Lib.

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

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



Consider the mapping from a physical Display (Coordinates) to a Virtual Display.

Motivations: Portable code Development, Simplicity.

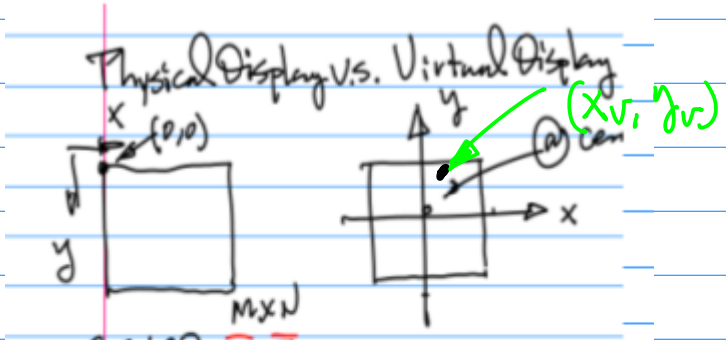
CMPE240

F2023

3/1

Ref → 2022F-101-notes-part1-cmpe240-2022-11-30.pdf pp.23

Physical Display v.s. Virtual Display



SPILCD physical

M: Row No. of
N: Col. No. of

Mapping Funct Fig.1.

(x_v, y_v) for virtual ~

(x_p, y_p) for physical ~

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

$$\begin{cases} x_p = x_v + M/2 & \cdots (7) \\ y_p = -y_v + N/2 & \cdots (8) \end{cases}$$

Consider x comp. "Before" (e.g. in the virtual display) and "After" (e.g. in the physical Display)

After ? Before

$$x_p = M/2 + x_v$$

Therefore

$$x_p = x_v + M/2$$

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

After Before

$$y_p = N/2 - y_v$$

$N/2$: Total Number of Rows.

$-y_v$: Pointing to the opposite direction

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).

pp.27.

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.
$\vec{x_w i} \times \vec{y_w j} = \vec{z_w k}$

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

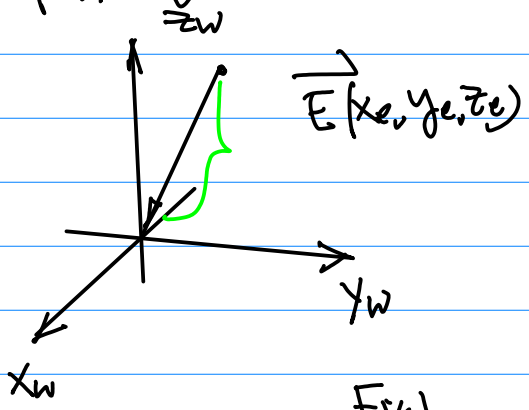


Fig1.

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

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

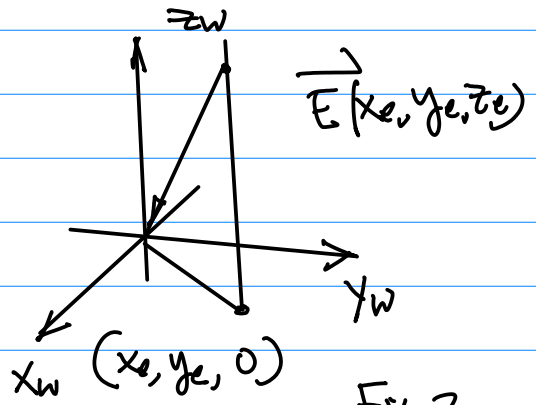


Fig.2

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

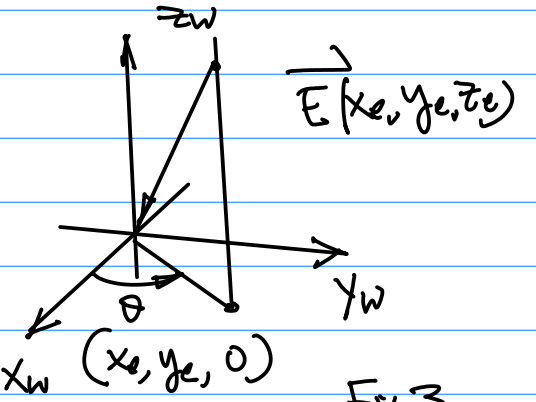


Fig.3.

6°. Define An Angle w.r.t. (0,0,0) and positive x_w-axis, Counter Clockwise Rotation,
Define 2nd Angle. ϕ

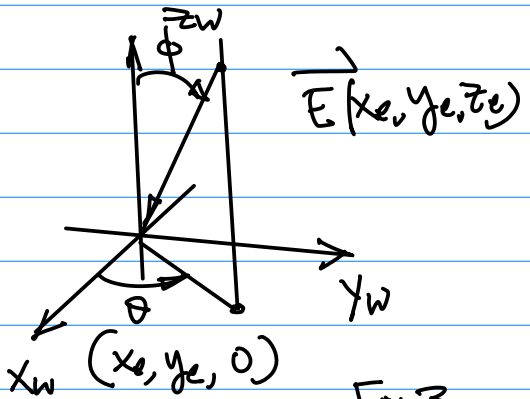


Fig. 3.

2nd Angle ϕ: W.r.t Z_w-axis
Counter Clockwise : Positive.
$\phi \geq 0$ if it is counter Clockwise

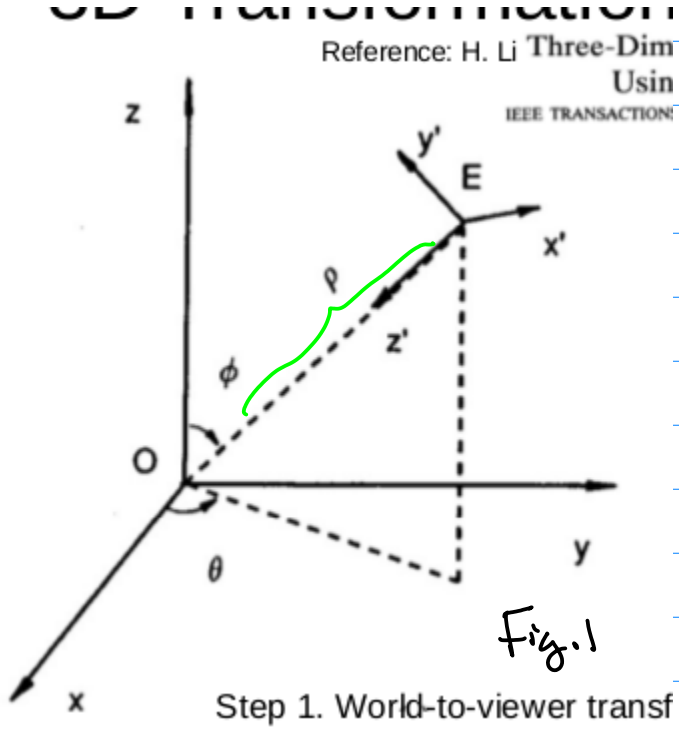
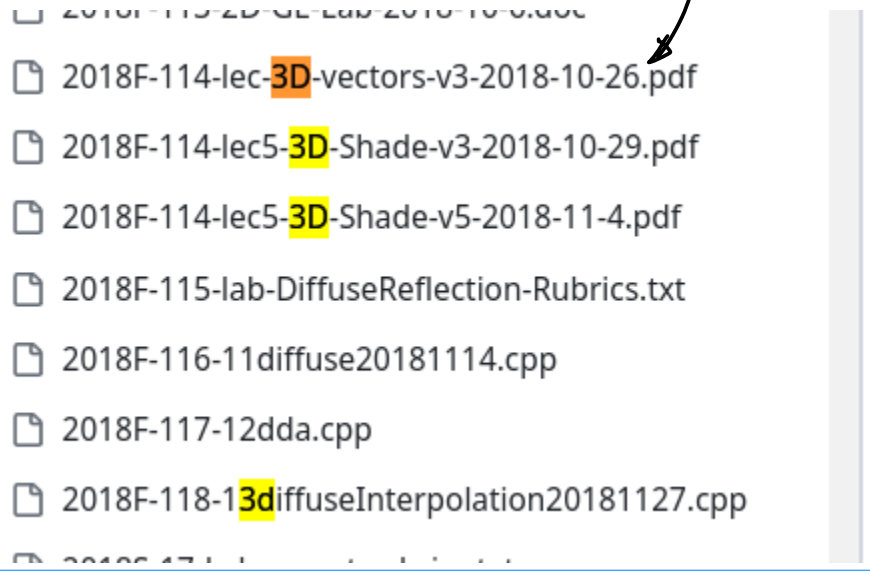


Fig. 1

Step 1. World-to-viewer transf

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

Oct. 11th (Wed).
Continuation of setting up Viewer Coordinate System.
Transformation Pipe.
Ref: Lect. github.
2023S-101-note-Part1. Note: Transformation pipeline



$$\mathbf{T} = \begin{bmatrix} -\sin\theta & \cos\theta & 0 & 0 \\ -\cos\phi\cos\theta & -\cos\phi\sin\theta & \sin\phi & 0 \\ -\sin\phi\cos\theta & -\sin\phi\cos\theta & -\cos\phi & \rho \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \ldots(2)$$

$$\begin{pmatrix} x_i' \\ y_i' \\ z_i' \\ 1 \end{pmatrix} = \begin{bmatrix} -\sin\theta & \cos\theta & 0 & 0 \\ -\cos\phi\cos\theta & -\cos\phi\sin\theta & \sin\phi & 0 \\ -\sin\phi\cos\theta & -\sin\phi\cos\theta & -\cos\phi & \rho \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x_i \\ y_i \\ z_i \\ 1 \end{pmatrix} \quad \ldots(2\text{-}b)$$

After (in the Viewer Coordinate)

before (in the World Coordinate)

x_e-y_e-z_e : Left Hand System.

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

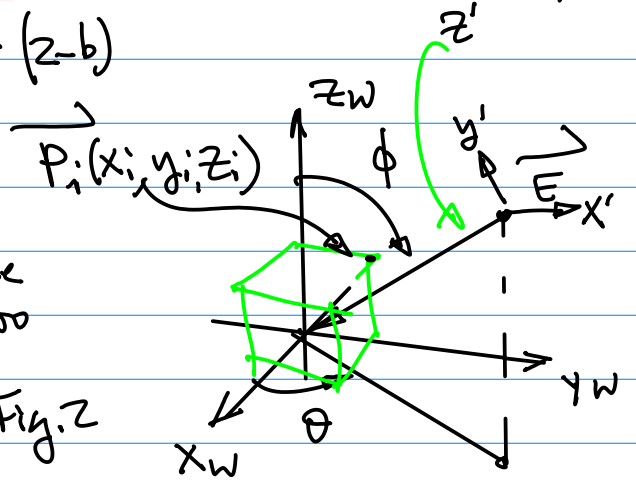


Fig. 2

$\begin{pmatrix} - & & \\ - & - & \\ - & - & - \end{pmatrix}_{4\times4}$

Side of the Cube = 100

$\begin{pmatrix} \sin\theta & \cos\theta & \\ \cos\theta & \sin\theta & \\ \cos\theta & \cos\theta & \end{pmatrix}_{4\times4}$

$\begin{pmatrix} \cos\phi & \cos\phi & \sin\phi & \\ \sin\phi & \sin\phi & \cos\phi & \rho \end{pmatrix}_{4\times4}$!!

Given A cube with one of its Vertices is $\overrightarrow{P_i(x_i, y_i, z_i)} = (0, 100, 110)$

Now, using Eqn (2-b), we can find $\begin{pmatrix} x_i' \\ y_i' \\ z_i' \\ 1 \end{pmatrix}$ after the x_w-y_w-z_w to x_e-y_e-z_e

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

First, find $\sin\theta$ =

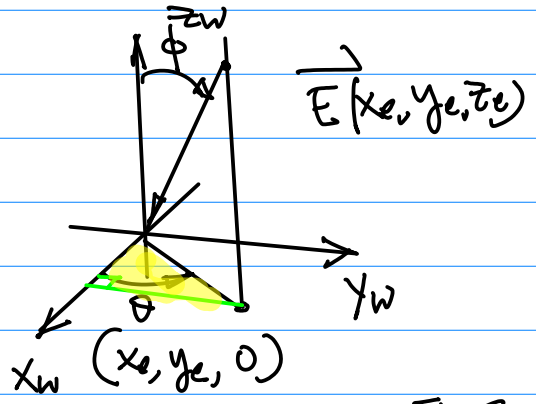


Fig.3.

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

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

Now, at the Same Triangle,

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

From the given condition,

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

Hence, $\sin\theta = \frac{200}{\sqrt{200^2+200^2}} = \frac{200}{200\sqrt{2}} =$

$$= \frac{1}{\sqrt{2}} = \sqrt{2}/2$$

Similarly,

$$\cos\theta = \frac{200}{\sqrt{200^2+200^2}} = \sqrt{2}/2$$

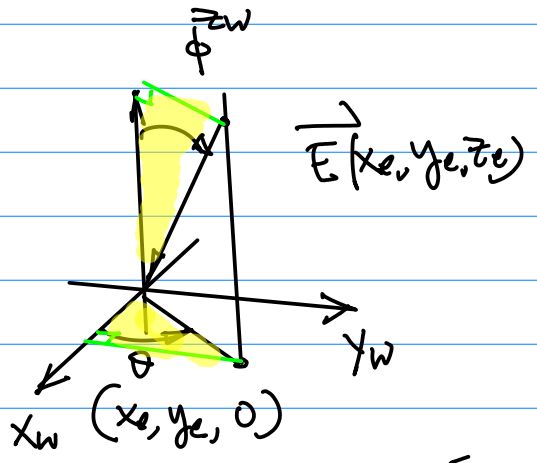


Fig.4

$$\sin\phi = \frac{\text{Green Line}}{\Delta'}$$

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

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

$$\sin\phi = \frac{\sqrt{x_e^2+y_e^2}}{\rho} = \frac{\sqrt{x_e^2+y_e^2}}{\sqrt{x_e^2+y_e^2+z_e^2}} \quad \cdots (4\text{-b})$$

$$= \frac{\sqrt{200^2+200^2}}{\sqrt{200^2+200^2+200^2}} = \frac{\sqrt{2}}{\sqrt{3}}$$

$$= \sqrt{2}\cdot\sqrt{3}/3 = \sqrt{6}/3$$

Therefore, the mapping matrix is

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

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

$$\begin{bmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 & 0 \\ \frac{\sqrt{2}}{2}\frac{\sqrt{3}}{3} & \frac{\sqrt{2}}{2}\frac{\sqrt{3}}{3} & \frac{\sqrt{6}}{3} & 0 \\ \frac{\sqrt{2}}{2}\frac{\sqrt{6}}{3} & \frac{\sqrt{2}}{2}\frac{\sqrt{6}}{3} & \frac{\sqrt{3}}{3} & 200\sqrt{3} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Therefore, the New pt After Transformation from X_w-Y_w-Z_w to X_e-Y_e-Z_e is:

after (X_e-Y_e-Z_e)

$$\begin{bmatrix} X' \\ Y' \\ Z' \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 & 0 \\ \frac{\sqrt{2}}{2}\frac{\sqrt{3}}{3} & \frac{\sqrt{2}}{2}\frac{\sqrt{3}}{3} & \frac{\sqrt{6}}{3} & 0 \\ \frac{\sqrt{2}}{2}\frac{\sqrt{6}}{3} & \frac{\sqrt{2}}{2}\frac{\sqrt{6}}{3} & \frac{\sqrt{3}}{3} & 200\sqrt{3} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 100 \\ 110 \\ 1 \end{bmatrix}$$

Before (In X_w-Y_w-Z_w World)

Oct. 16 (Mon).

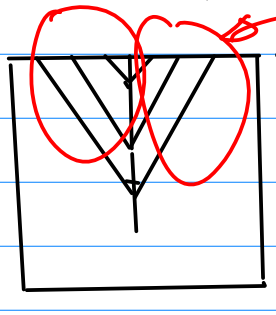
Note 1. Quiz update,
- a. Naming convention to be followed;
- b. Personal ID required.
- c. Quiz & Exam cannot be submitted Late.

Note 2. Form 2 Person Team.

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

Q&A.

1.



$\vec{P}(x,y) = \vec{P}_1(x_1,y_1) + \lambda(\vec{P}_2(x_2,y_2) - \vec{P}_1(x_1,y_1))$

$\lambda = 0.8$

2° X[i][j]

Index for Superscript for the different level.

Index for Subscript.

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

3° $T^{-1}RT \quad \cdots (2)$

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

from Eqn (1).

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

4° Color: We would like to have more colours than the Graphic colours. (C-Code defined)

a) r, g, b primitive color. 8 bit Each.

b) Total Possible colors. hence it's $2^{24} = 2^4 \cdot 2^{20}$

$= 16\,M.$

Example. Continuation.

Find the pt. in the x_e-y_e-z_e, e.g. After the world to Viewer.

$$\begin{pmatrix} x_i' \\ y_i' \\ z_i' \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 & 0 \\ \frac{\sqrt{2}}{2}\frac{\sqrt{3}}{3} & \frac{\sqrt{2}}{2}\frac{\sqrt{3}}{3} & \frac{\sqrt{6}}{3} & 0 \\ \frac{\sqrt{2}}{2}\frac{\sqrt{6}}{3} & \frac{\sqrt{2}}{2}\frac{\sqrt{6}}{3} & \frac{\sqrt{3}}{3} & 200\sqrt{3} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 100 \\ 110 \\ 1 \end{pmatrix}$$

$$x_i' = \frac{\sqrt{2}}{2}\times 0 + \frac{\sqrt{2}}{2}\times 100 + 0\times 110 + 0\times 1$$

$$= 50\sqrt{2}$$

$$y_i' = \frac{\sqrt{2}}{2}\frac{\sqrt{3}}{3}\times 0 + \frac{\sqrt{2}}{2}\frac{\sqrt{3}}{3}\times 100 + \frac{\sqrt{6}}{3}\times 110 + 0\times 1$$

$$= 50\sqrt{6} + \frac{110}{3}\sqrt{6} = \left(50 + \frac{110}{3}\right)\sqrt{6}.$$

$$z_i' = 0 + \frac{50}{3}\sqrt{2}\sqrt{3} + \frac{110}{3}\sqrt{3} + 200\sqrt{3}$$

$$= \frac{50}{3}\sqrt{6} + \frac{110}{3}\sqrt{3} + 200\sqrt{3}.$$

C-Code Implementation.

```
X_Prim[i] = -sin(Theta) * X[i] +
            cos(Theta) * y[i] ;
Y_Prim[i] = -cos(Theta) * cos(phi) * X[i]
            - sin(Theta) * cos(phi) * y[i]
            + sin(phi) * z[i] ;
```

```
Z_prim[i] = -Cos(Theta)*sin(phi)*x[i] - Cos(Theta)*Sin(phi)*y[i]
            -Cos(phi)*z[i] + rho;
```

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 Eqn (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-2-Viewer Transformation, X_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:

b. $\lambda = 0.8$ Reduction By 20%

$\vec{P_0}(x_0, y_0) = (50, -10)$

$\vec{P_1}(x_1, y_1) = (50, 15)$

$\| \vec{P_1}(x_1, y_1) - \vec{P_0}(x_0, y_0) \| = 25$ magnitude

$\lambda *$ magnitude $= 0.8 * 25 = 20$

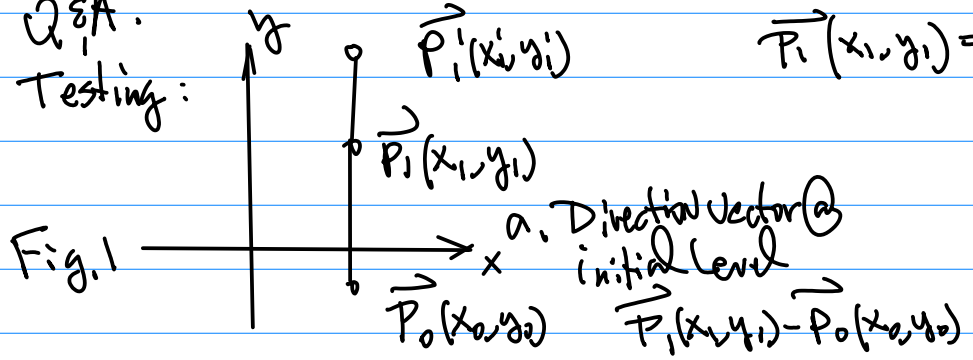


Fig. 1

a. Direction Vector @ initial Level

$\vec{P_1}(x_1, y_1) - \vec{P_0}(x_0, y_0)$

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

$= (50, 35)$

$\vec{P}_1'(x_1', y_1') = \vec{P}_1^0(x_1^0, y_1^0) + \lambda(\vec{P}_0^0(x_1^0, y_1^0) - P_0^0(x_0^0, y_0^0))$

$= (50,15) + 0.8 * ((50, \cancel{-10}\ 15) - (50, \cancel{15}\ -10))$

$= (50,15) + 0.8 * (0, 25)$

$= (50,15) + (0, 20)$

$= (50, 35)$

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

$\Delta x = -x_1$

$\Delta y = -y_1$

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

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

Example: Continuation of Transformation Pipeline. Project 3D objects. in xe-ye-ze to

Consider Pre-Processing/Post Processing.

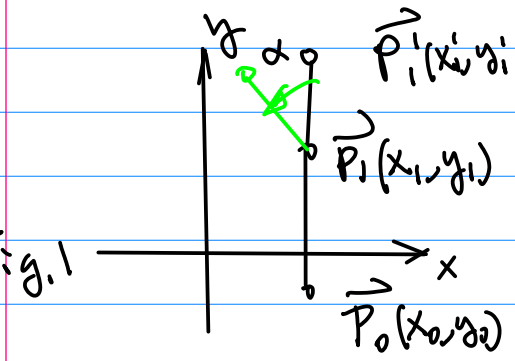
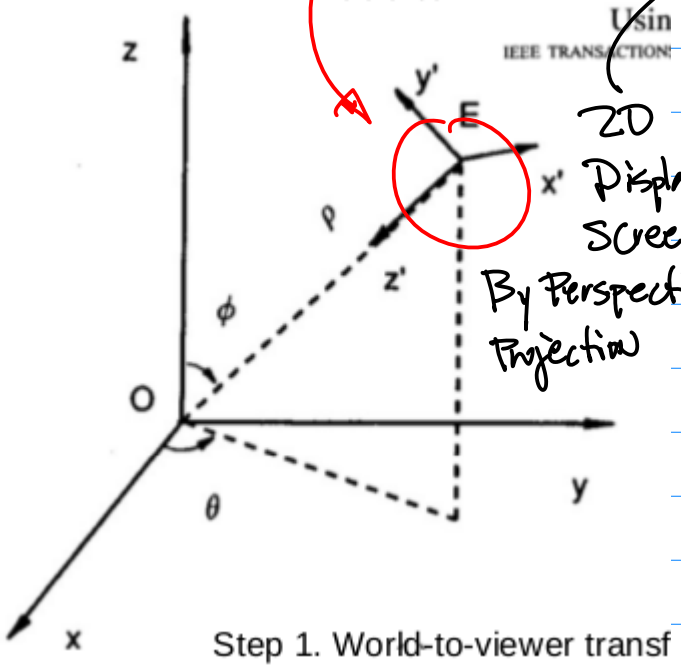


Fig.1

Note: 1° Rotation is w.r.t. $\vec{P}_1(x_1, y_1)$ in the Project Design. However $R_{3\times3}$ is defined w.r.t. the origin (0,0) of x-y plane.

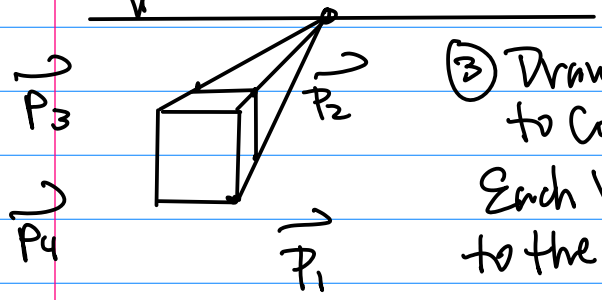


2D Display Screen By Perspective Projection

CMPE240

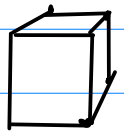
Fall 2023

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

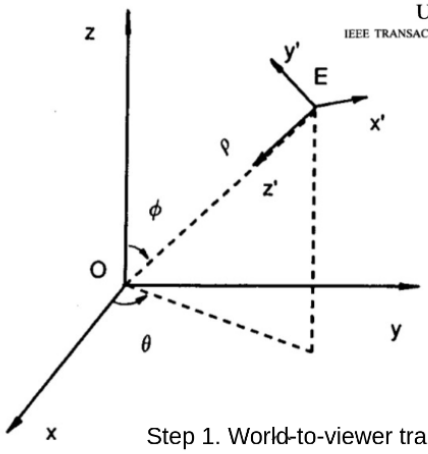
CMPE240-Adv-Microprocessors / 2018F / 2018F-114-lec-3D-vectors-v3-2018-10-26.pdf

hualili Add files via upload

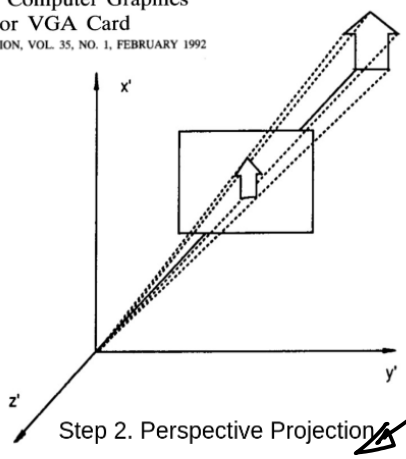
449 KB Code 55% faster with GitHub Copilot

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



Step 1. World-to-viewer transform

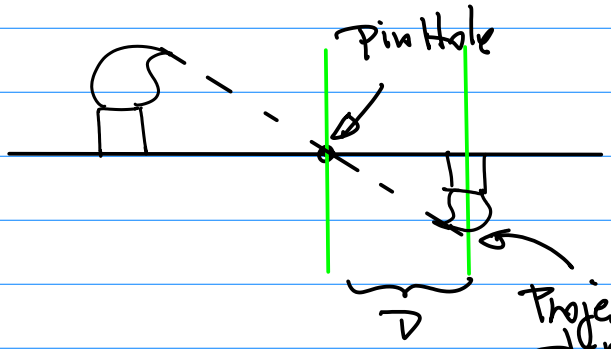


Step 2. Perspective Projection

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

$$x_p = x_e\left(\frac{D}{z_e}\right) \quad \text{...(1)}$$

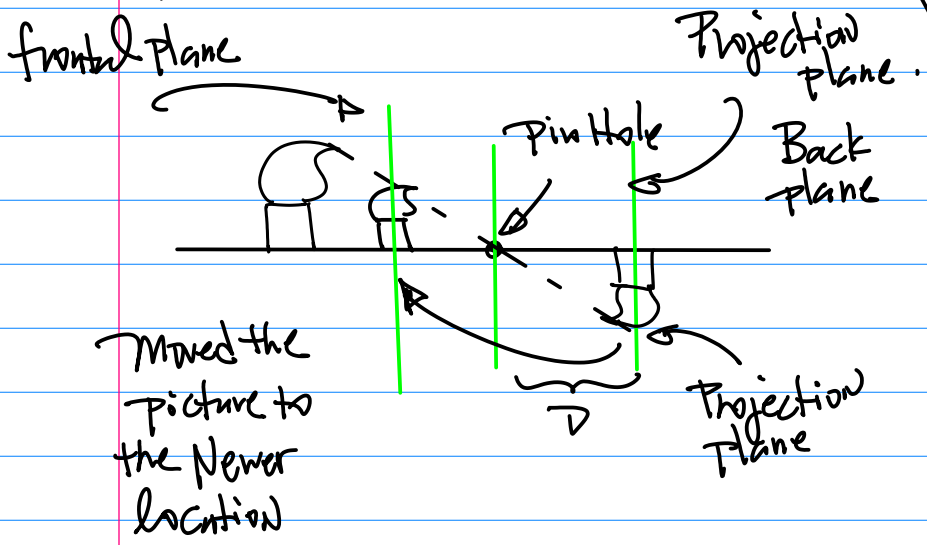
$$y_p = y_e\left(\frac{D}{z_e}\right) \quad \text{...(2)}$$



Pin-Hole model of a Virtual Camera.

Note 1° D↑, Picture Size↑.
D↓, Picture Size↓
matches Eqn (1) & (2).

Note 2°

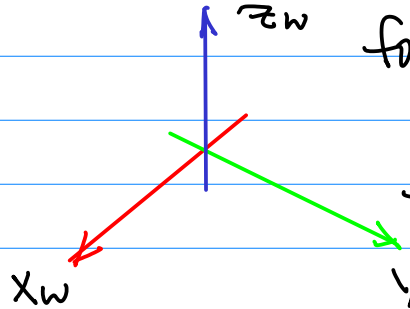


Moved the picture to the Newer location

Transformation pipeline.
(1) World to Viewer.
(2) Perspective Projection Eqn (1) & (2).

Requirements.

Design & Draw World Coordinate System X_w-Y_w-Z_w.



for X_w: $\vec{P_0}(x_0, y_0, z_0) = (0,0,0)$
$\vec{P_1}(x_1, y_1, z_1) = (50,0,0)$

DrawLine to plot (x_0', y_0') to (x_1', y_1') After Perspective Projection

Oct. 23rd (Monday)

Example: Hand Calculation of 3D Transformation Pipeline.

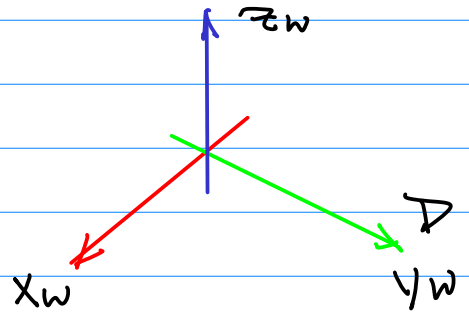


Fig 1a

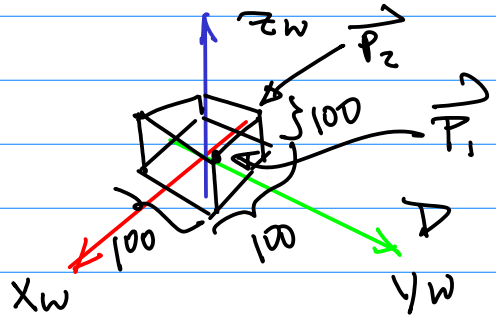


Fig. 1b

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

option2: $\vec{E}(x_e, y_e, z_e) = (0, 0, 200)$

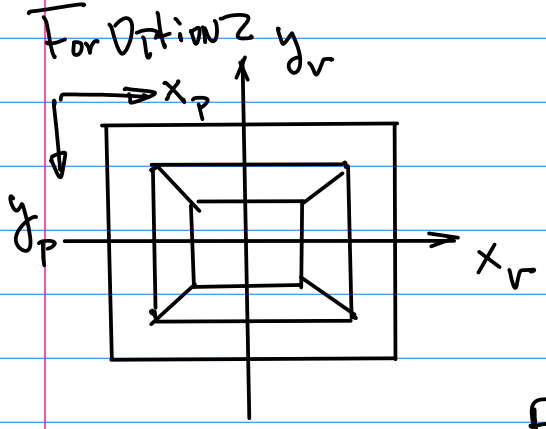


Fig. 2

Sol:

1. Data points, Vertices of the given Cube:

$$\{\vec{P_i}(x_i, y_i, z_i) \mid i = 1, 2, \ldots, 8\}$$

$$= \{\vec{P_1}(x_1, y_1, z_1), \vec{P_2}(x_2, y_2, z_2), \ldots$$

$$\ldots \vec{P_8}(x_8, y_8, z_8)\}$$

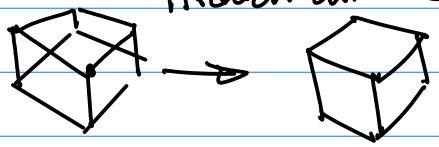
where

$$\vec{P_1}(x_1, y_1, z_1) = (50, 50, 100)$$

Note:

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

Removal of Hidden Lines Hidden Surfaces



$$\vec{P_2}(x_2, y_2, z_2) = (50, 50, 100) \text{ etc.}$$

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

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

Find $\cos\theta$, $\sin\theta$ and $\cos\phi$, $\sin\phi$.
(see Eqn(3)–(5), on pp.35)

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

3. 2nd step of the Transformation pipeline:

$$\begin{cases} x_p = x_v \dfrac{D}{z_r} & (2a) \\ y_p = y_v \dfrac{D}{z_r} & (2b) \end{cases}$$

Assume $\theta = 20$
Substitute it into Eqn(2a) & (2b).

$$\begin{pmatrix} x_i' \\ y_i' \\ z_i' \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 & 0 \\ \frac{\sqrt{2}}{2}\frac{\sqrt{3}}{3} & \frac{\sqrt{2}}{2}\frac{\sqrt{3}}{3} & \frac{\sqrt{6}}{3} & 0 \\ \frac{\sqrt{2}}{2}\frac{\sqrt{6}}{3} & \frac{\sqrt{2}}{2}\frac{\sqrt{6}}{3} & \frac{\sqrt{3}}{3} & 20\sqrt{3} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \sqrt{50} \\ 50 \\ 100 \\ 1 \end{pmatrix}$$

$\begin{pmatrix} x_v \\ y_v \\ z_v \end{pmatrix}$ Substitute it into Eqn(2a) & (2b)

$$\begin{cases} x_p = x_v \dfrac{D}{z_r} & (2a) \\ y_p = y_v \dfrac{D}{z_r} & (2b) \end{cases}$$

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

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

Homework: One A week (Sunday, 29th.)
Part 1: X_w-Y_w-Z_w (R-G-B) Draw Color Coded 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-101b ~ Note
pp 51.

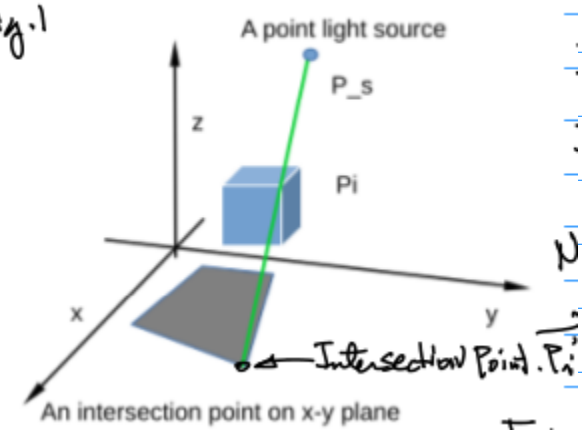
Oct.25 (Wed)

Note1. Canvas Posting of the Homework Due On Oct.29 (Sunday) By the End of the day today.

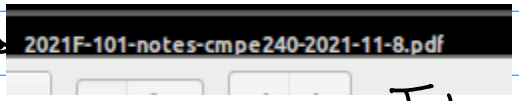
Note2. Midterm is scheduled After Shadow Computation. Nov. 13 (Monday).

Note3. Please Bring Your Prototype System to the Class for Discussion and Inspection (Homework on Xw-Yw-Zw Drawing & Cube Wire-frame display).

Fig.1



Example: Ref: On the github.



To be posted on the github.

Background:
1° Ray Equation
2° plane Equation

Pp46. Ray Equation

$$\vec{r}(x_r, y_r, z_r) = \vec{P_i}(x_i, y_i, z_i) + \lambda(\vec{P_s}(x_s, y_s, z_s) - \vec{P_i}(x_i, y_i, z_i)) \quad \cdots (2)$$

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 $\vec{P_i}$.

Example: Plane Equation. from the Lecture Notes.

Ray Equation: Pp.46.

$$\vec{R} = \vec{P_i} + \lambda(\vec{P_s} - \vec{P_i}) \quad \cdots (3)$$

The Intersection Point on X_w-Y_w plane, $\vec{P_i}'$ is a common pt shared by Ray Eqn (3) and plane eqn (2).

$$\begin{cases} \vec{R} = \vec{P_i} + \lambda(\vec{P_s} - \vec{P_i}) & \cdots (4\text{-}1) \\ \vec{n} \cdot (\vec{r} - \vec{a}) = 0 & \cdots (4\text{-}2) \end{cases}$$

Formulation.

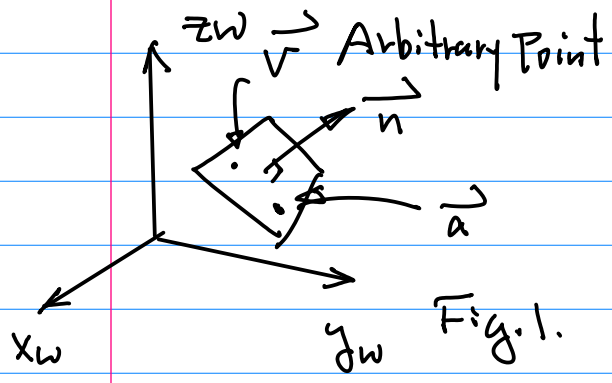


Fig. 1.

$\vec{n}$: 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. $\vec{a}$

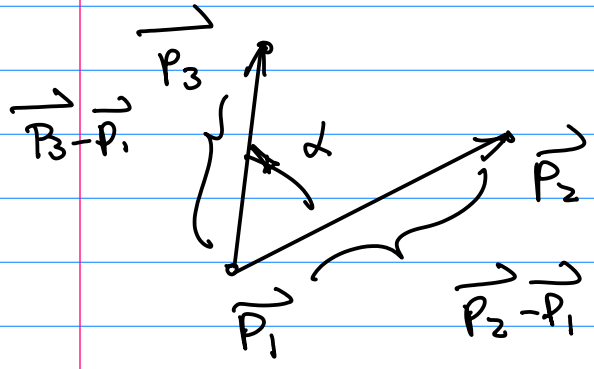
$$\vec{n}(n_x, n_y, n_z) = (n_x, n_y, n_z) \quad \cdots (1)$$

And a known point $\vec{a}$

$\vec{v} - \vec{a}$ ... (2) A line

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

Note : Vector Dot Product.



$$(\vec{P_2} - \vec{P_1}) \cdot (\vec{P_3} - \vec{P_1})$$

$$= \|\vec{P_2} - \vec{P_1}\| \, \|\vec{P_3} - \vec{P_1}\| \cos\alpha \quad \cdots (3)$$

Now, Let's find the Intersection Point

From Eqn(4.2),

$$\vec{n} \cdot (\vec{v} - \vec{a}) \Big|_{\vec{v} = \vec{R}} = 0 \quad \cdots (4)$$

$$\vec{n} \cdot (\vec{R} - \vec{a}) = 0$$

$$\vec{R} = \vec{P_1} + \lambda(\vec{P_5} - \vec{P_1})$$

$$\vec{n} \cdot (\vec{P_1} + \lambda(\vec{P_5} - \vec{P_1}) - \vec{a}) = 0$$

Now,

$$\vec{n} \cdot \vec{P_1} + \lambda \cdot \vec{n} \cdot (\vec{P_5} - \vec{P_1}) - \vec{n} \cdot \vec{a} = 0$$

From Fig.1 ON PP.44.

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

$$\vec{n} \cdot \vec{P_1} + \lambda \vec{n} \cdot (\vec{P_5} - \vec{P_1}) = 0$$

$$\lambda \vec{n} \cdot (\vec{P_5} - \vec{P_1}) = -\vec{n} \cdot \vec{P_1}$$

$$\lambda = -\frac{\vec{n} \cdot \vec{P_1}}{\vec{n} \cdot (\vec{P_5} - \vec{P_1})} \quad \cdots (5)$$

Now, Calculation

$$\lambda = -\frac{\vec{n}\cdot\vec{P_i}}{\vec{n}\cdot(\vec{P_s}-\vec{P_i})}$$

$$= -\frac{(n_x,n_y,n_z)\cdot(x_i,y_i,z_i)}{(n_x,n_y,n_z)\cdot(x_s-x_i,\ y_s-y_i,\ z_s-z_i)}$$

$$= -\frac{n_x x_i + n_y y_i + n_z z_i}{n_x(x_s-x_i)+n_y(y_s-y_i)+n_z(z_s-z_i)} \quad (5\text{-}b)$$

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
N = (nx,ny,nz) = (0, 0, 1) ... (1)

Suppose we have a cube with side length = 100, and then
P_i(x_i,y_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,

nx*x_i + ny * y_i + nz * z_i
= 0 + 0 + z_i ... (2)

and

nx * (x_s - x_i) + ny * (y_s - y_i)
+ nz * (z_s - z_i)
= 0 + 0 + 1 * (z_s - z_i)
= z_s-z_i = 250 - 110 = 140

lambda = - z_i / (z_s - z_i)
= - 110 / 140

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

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),
50 - 110/140 * (-50),
110 - 110/140 * 140)

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-118-13diffuse~

Note1. Define the Virtual Camera Location, E(xe,ye,ze)

```
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 Xw-Yw-Zw, Xe-Ye-Ze, 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;
```

Note3. Define Data points $P_i(x_i, y_i, z_i)$ for x_w-y_w-z_w.
Origin (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
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
```

Reference purpose for the Decoration / Texture Mapping in 3D.

Note3. The Sin, Cos and ρ for the 4x4 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,

$$\vec{R} = \vec{P_i}(x_i, y_i, z_i) + \lambda\left(\vec{P_s}(x_s, y_s, z_s) - \vec{P_i}(x_i, y_i, z_i)\right)$$

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 4x4 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 Xe          viewer.X[i] = -sPheta * world.X[i] + cPheta * world.Y[i];
245 ye          viewer.Y[i] = -cPheta * cPhi * world.X[i]
246             - cPhi * sPheta * world.Y[i]
247             + sPhi * world.Z[i];
248 ze          viewer.Z[i] = -sPhi * cPheta * world.X[i]
249             - sPhi * cPheta * world.Y[i]
250             -cPheta * world.Z[i] + Rho;
```

Note6. 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];
```

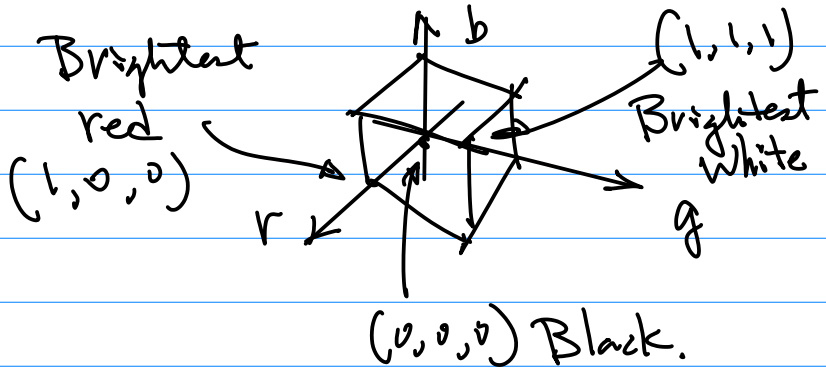
Homework (ON Shadow Computing).

1. Implement Shadow Computation Algorithm. By plotting A 4-Point Shadow. Based on your previous Homework.
2. Modify the Cube in the previous Home work By elevating the Cube 15 units.
3. Color for the shadow, use dark blue.

r, g, b settings:

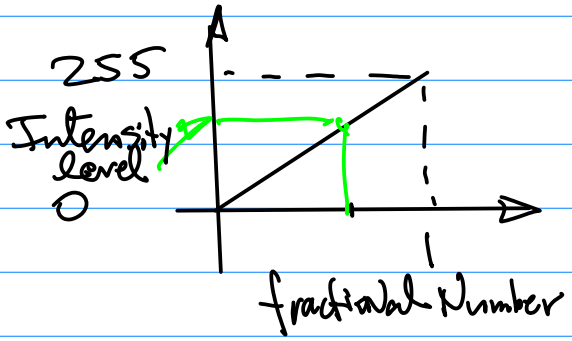
a) 8 bits for Each color.

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)

Nov. 6th (Monday).

Note 1. Midterm Exam is scheduled on Next Monday.

Review Session in-class is scheduled on Wednesday.

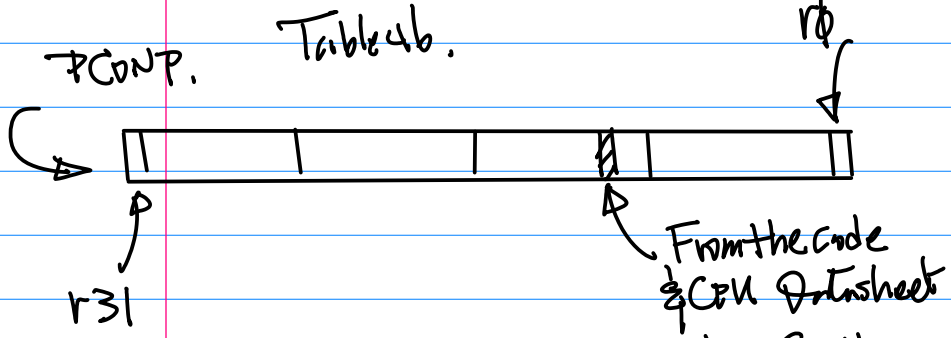
Today's Topics:

1° SSPInit Code.

2° Hardware Design for LCD/CPU Interface Design.

Example: Line 210-246.

Line 224 Naming Convention
LPC_SC → PCONP
Section 4.8.5. pp61
CPU Datasheet.
Table 46.



Line 227.
LPC_SC → PCLKSEL

Line 229
Line 230
LPC_SC → PINSEL0.

pp.58, Table 42.
Section 4.7.3. pp57.

Note: Draw a 32Bit SPR illustration.
Connect to CPU Datasheet (Tables for the SPR).
Memory Bank holding this SPR.
Code Implementation.

Note: The Sequence to Init & Config SPRs for SSP Interface. (SPI)

PCONP → PCLKSEL → PINSEL0

Line 233/234 CS (Chip select: e.g. Select/Enable my LCD Display module)

Line 238
SSP1 CLK.
pp.431. Table 371

Use Case Leads to Design Requirements, for Example, 2D G.E. Design. Frame Rate, Resolution of the Display → Carry out the Design By Using CPU Datasheet, and formula

$$f_{pwm} = \frac{PCLK}{CPSDVSR * (SCR+1)}.$$

then, Coding.

Homework: Due Nov. 14th (Sunday).

1° Requirements:

a) Based on the Homework of Drawing A wireframe Cube in X_w-Y_w-Z_w, Add a point light source, such as $\vec{P_S}(x_s, y_s, z_s) = (-5, 50, 250)$,

b) Use the vertices from the Top Surface of the Cube to generate 4 Ray Equations, then Compute the intersection Points on X_w-Y_w plane

Note: Computation is Carried out Before the Transformation pipeline, e.g., in X_w-Y_w-Z_w Coordinate.

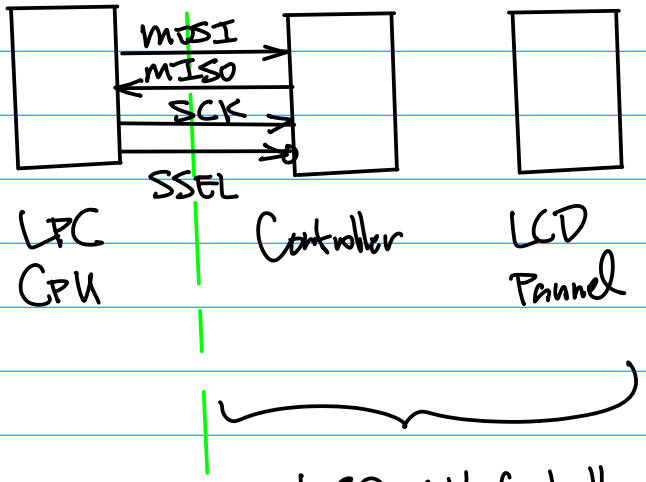
c) Draw the shadow first Before Drawing the Cube.

Note: Design/Select A proper dark color for the shadow.

Note: please work/Discuss this homework with your team. But coding has to Be individually, No code can be shared.

Note: Please Bring Your Board for show & Tell, Demo ON Monday, Nov. 20 (Monday).

Consider Hardware Design for LCD Controller Interface.



LCD with Controller Build In.

Note: please provide Clear indication of the Signal Flow By Drawing Arrow on Each Signal line.

Also, place a circle "O" for Active Low Signal.