

August 23rd (mon).

CmpE240
Section 1.

CmpE240

HARRY LI.

E-mail: huagli@sjsw.edu

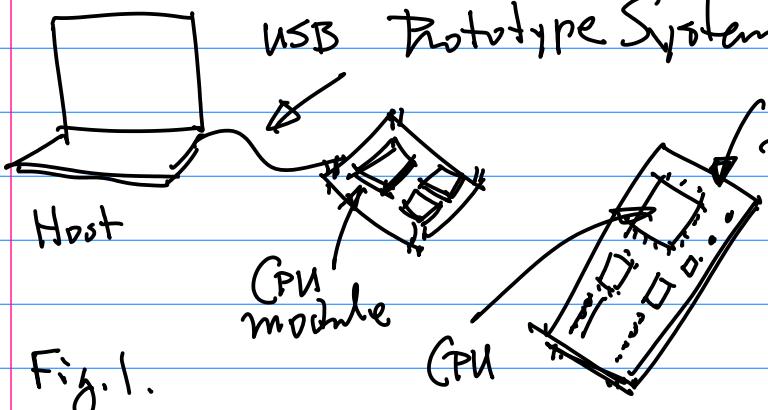
Text message (650) 400-1116

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

Advanced Microprocessor Systems

=

Prototype System
with A CPU module



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

Autonomous Systems, Nvdia Jetson

Tx2.

Textbooks, References

1° NXP LPC1769 GPU Datasheet
800+ pages

Homework: Down
Load pdf. Before

Next Monday, Aug. 30th.

2. LPC1769 Schematics
of the CPU module

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

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

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

Note: Buy LPC1769 GPU
module.

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

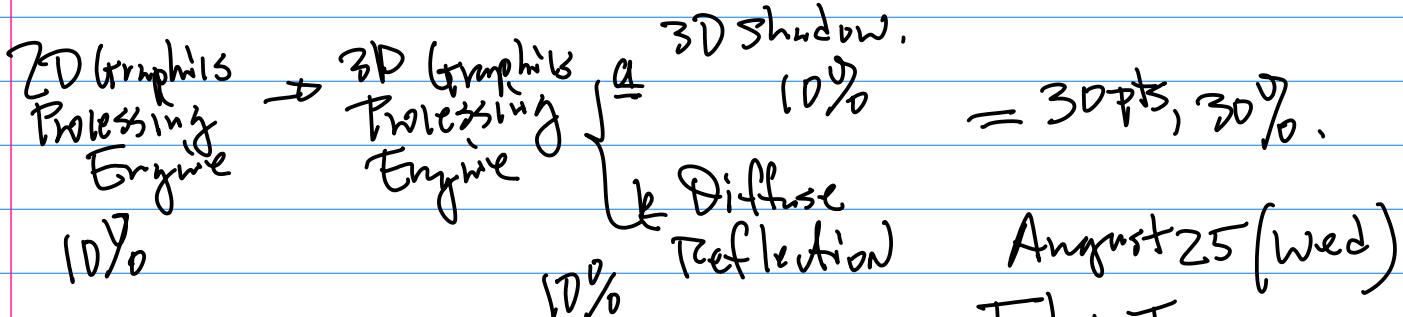
Grading Policy & Projects
2 projects
(phase I & II)

2D Graphics
Processing
Engine → 3D Graphics
Processing
Engine

→ 3D Graphics
Processing
Engine

CmPE240

2.



midterm: 30%, Final 40% (Comprehensive)

Option 1. (5%+) NVDIA NAND

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

GPIOD, SPI.

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

SOC, FPGA Board,

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

Policy On Project Submission.

1° Form 3-4 person Team.

2° No Source Code/Design material

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

3° Late project, 10% per week;

Tool for
Flashing the
CPU module

August 25 (Wed)

Today's Topics:

1° Bill of material

Reference: github/finalisti
/Cmpe240/2018F

The B.D.M.

1. CPU module NXP LPC1769

3rd Party (DigitalArt), module
To Distributors

DigKey.com, Mouser.com

etc. Expecting Delays.
Lead Time over 8 weeks

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

Each person will need to have
his/her Board;

Option 1: NAND. @ 440+ pages

"firmware" Datasheet

= Jetpack 4.3 or Higher

→ (O.S. + Libs. + Packages)

Compendium

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

Tool chain, Device Driver Debugging
2. Development;

Option 2. PISC-V, verilog, FPGAs.

2. Power Regulator ICs such as

7812, 7805 ... 1117

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

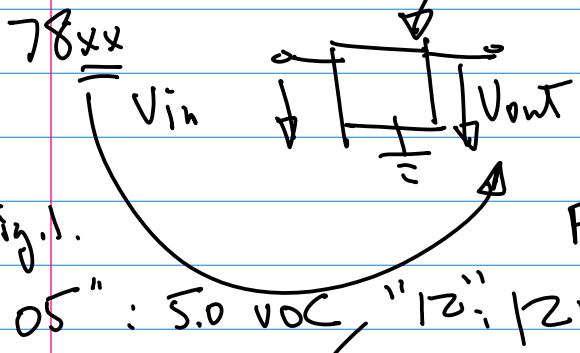


Fig. 1.

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

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

DC Voltage Source

a About 7805

1000 mW.

b 7.5 VDC

OR

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

c Why Do we use it?
Current Rating.

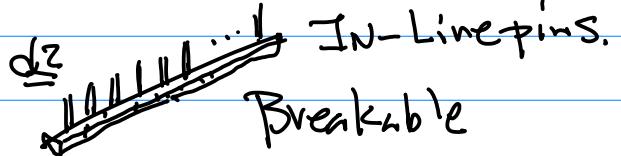
\hookrightarrow Deploy the System.

3 "Glue" Components / Resistors a

c LEDs. (Red, Green) for Debugging purpose, for PWZ.
(GPIO), $I_{LED} = 4 \sim 10 \text{ mA}$

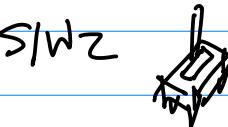
d Connectors.

d J1 for Power Input \Rightarrow pin

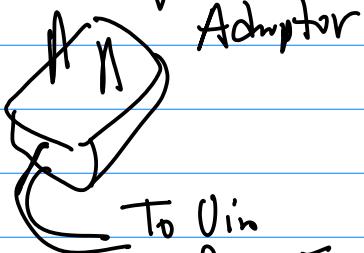


to mount CPU module.

e Switch. S/W1: to toggle PWR.



f Wire for Wire Wrapping / Soldering
28-30 AWG



4. Color LCD Display module

a SPI (Serial Peripheral Interface)

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

to Activate/Initialize LCD.

MCU Xpresso (J.D.E.)

\downarrow
S.T. Lib.

5. "Other" thing.

PJ-45 Connector

Sept. 8 (W)

Topics: 1. "Hello, the World" program

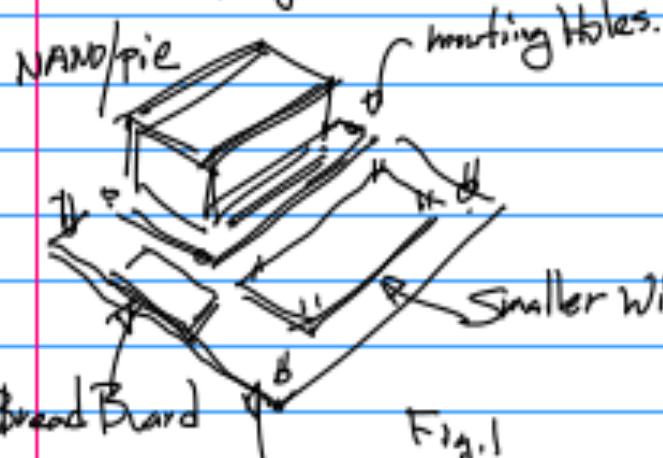
J Hardware Implementation

NXP MC91Xpresso.

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

Homework: Next Show-and-tell

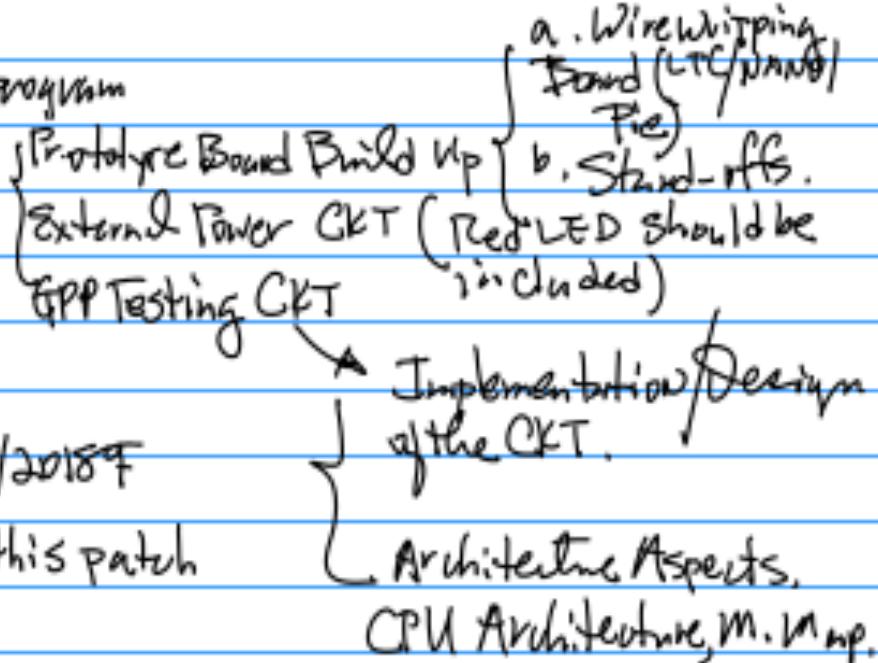
Wirewrapping Board;

Wirewrapping Board
"Carrier" Board

"Tap Plastic"

On the Board: a) Stand-offs.

b) Connector(s) for External JTAG



→ PVRIC(7805), with Red LED

CPU Architecture :

1. 32-bit Architecture

CPU Architecture

a. ALU 32bit

Arithmetic/Logic Unit.

b. Register File,

A Bank of Registers. 32 bits
GPRs

General Purpose Registers

Those Registers that can participate Any meaningful

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

SPIRs 32 bit

Naming Convention: Controllers.
6 letters

Common Design for SPIRs:

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

CON



- 2° Data Register, DAT

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

- C. Data Bus, Bi-Directional " 32 bits Information Flowing Both Directions.

Address, "Uni-directional" from CPU to the Outside. 32 bits

Notation: 32 bit Register

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

LSB

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

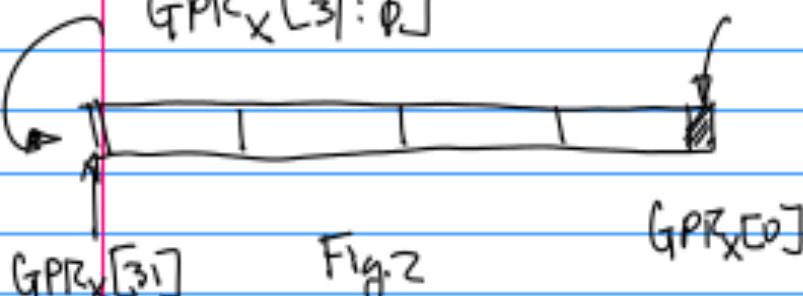
$$2^{10} = 1K, 2^{20} = 2^{10} \cdot 2^{10} = 1 \dots (2) M$$

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

... (4)

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

3. Memory Map.

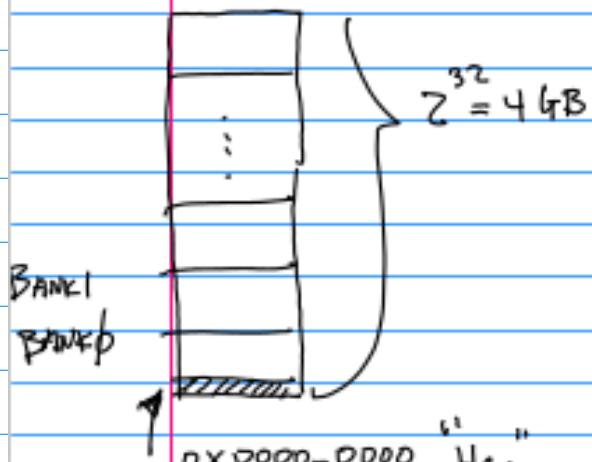


For Address Bus, Addr[31:0] =

$a_3 | a_{30} \dots a_1 | a_0$

CMPE240

8



Define Starting Addr. of Each Bank:

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

⋮
BANK7

Fig. 3.

32 bits for the address
& 8 bits for this memory

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

a. PWR-up Address:

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

→ 0x0000-0000

for ARM

Note: for x86, the PWR-up

Address: 0xFFFF_FFF0

For BANK0 : 0x0000-0000

BANK1 : 0x2000-0000

.. 2 : 0x4000-0000

Example: CPU Datasheet pp. 13.

GPIO 0x2009-C000

b. BANKS.

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

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

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

Addr. for SPRs are
mapped to here

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

3 bits → a_3, a_{30}, a_{29}

b. Which memory Bank holds

this GPIO? BANK1
whose starting Address is

0x2009-C000

Sept 13 (mon)

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

IDE. Objectives: To

Write first C program for testing purpose

Example: Starting from CPU

Memory map \rightarrow 8 Banks
PP13

1st 256 kB
= Flash

0x0000_0000,

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

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

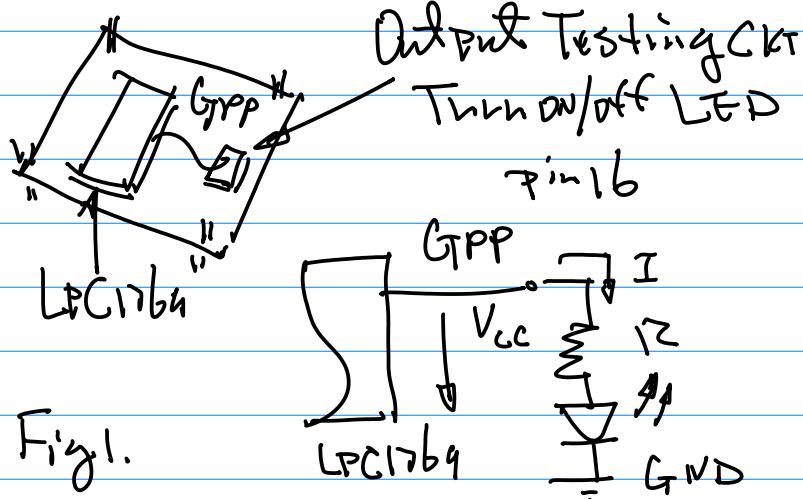
S.P.R.s. \leftarrow 0x4003_0000
"Con" 3 Letter
Size 4000

a Naming conversion Prefix Root + Postscript

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

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

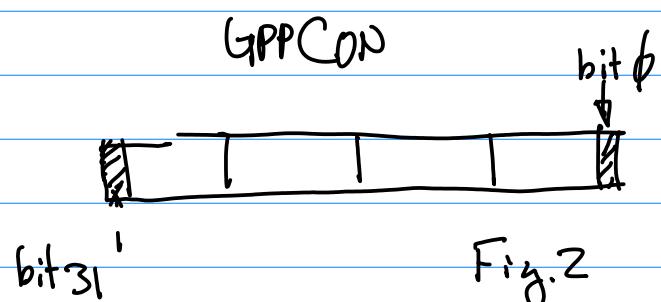
Example. GPP



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

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

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



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

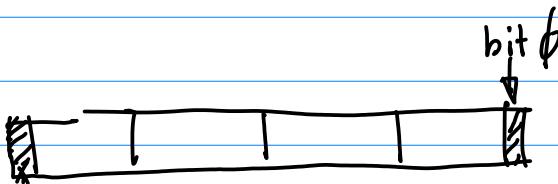
CmPE240

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

GPP (General Purpose Port)

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

We have to use the following
init & config pattern:



bit 31 0x F2ff_ff00

To make pin 1/b as an output.

First, "Port" the Architecture

Compiler to the target

#define GPPCON

then, Copy 0xF2ff_ff00 into
this memory location.

Homework: Show + Tell

By Next Week Installation of MCU

+ Import LTC17b9.

Sept 15 (W)

Architecture

Today's Topics: GPIO Design

Reference: 1° 2021F-105 ~ On
ID: UPI0

C-Code for Init & Config
is required

Example: Make GPP for
Input & Output
Testing.

Hardware

Software { NXP MCUXpresso
Import GPP Sample
"zip"

Design Step 1.

Identify / Select GPP / GPP-Pins
P_b.2, P_b.3

(Connector → CPU → Selection
Data Sheet)

Step 2. Define P_b.2 Asntent ,

P_b.3 As Input ,

Design the Hardware

Step 3. SPRs (Special
Purpose Registers) for

the GPP Peripheral
Controller

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

Note: SFRs commonly defined / utilized are

GPFCON

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

GPFCON

GPF DAT (32 bits \rightarrow 32 pins)

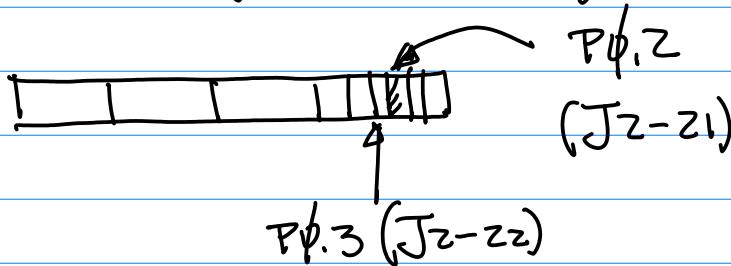
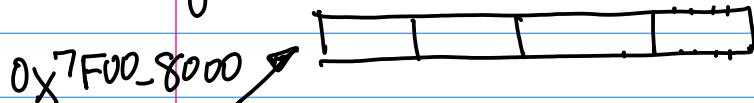


Fig. 1

Find Table on CPU Datasheet to
define P0.2 output,
P0.3 as input.

Example, Samsung Arren II Datasheet
pp312

Fig. 2



Question: Define Binary Pattern for
GPA CON to make
its pin 2 as an output?

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

SFRs, IDE, Sample Code;

2^o 2D GE.

Example: git (class)

2021F-105-GPP...

i. Naming Convention in C Compiler

LPC_GPIO0->FIODIR

LPC_GPIO0->FIOSET

LPC_GPIO0->FIOCLR

↑
Target Peripheral Controller
(Family)
↑
Special purpose
Register

From the Example, qit, 2021F-105

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

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

Chapter 9, pp 129

PINSEL[5:4] for P0,2

PINSEL[5:4] = 00 for I2C

PINSEL[5:4] = 01 for UART
Tx

pp 133 FIODIR Example,

P0,3 output, Find SPR?

Define bit values
for the output

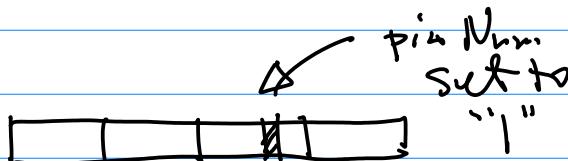
Table Look up.

FIODIR

FIOSET, CPU Datasheet
Look up.

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

```
{
    if (portNum == 0)
    {
        LPC_GPIO0->FIODIR |= (1 <<
            pinNum);
    }
    else if (portNum == 1)
    {
        LPC_GPIO1->FIODIR |= (1 <<
            pinNum);
    }
}
```



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

Logic Operation
 $\text{I} = \text{Btwise}$
 $\text{I} \&= ?$

Example: Set Pin

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

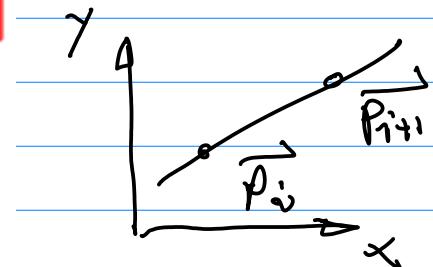
Turn ON LED
(Output '1')

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

Example: Clear the pin

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

Formulation for
a straight line



Now, 2D Vector Graphics

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

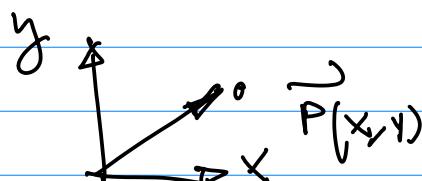
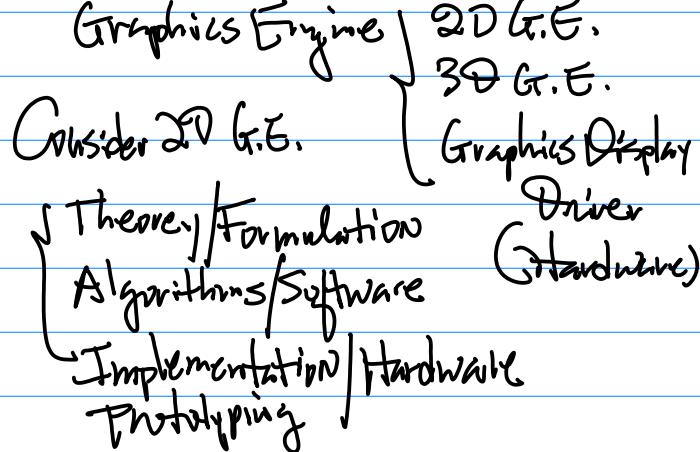


Fig.1

Sept. 22 (W)

Fig.2



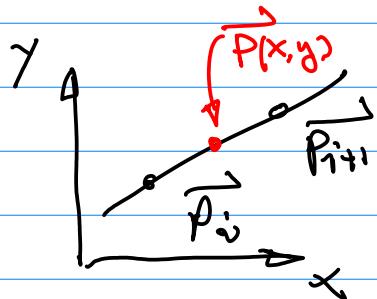


Fig. 2

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

Let's define a direction vector

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

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

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

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

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

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

Find direction vector?

Sol. By Eqn (i), we have

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

Sub. the given condition

$$\begin{cases} x_d = x_{i+1} - x_i \\ y_d = y_{i+1} - y_i \end{cases} \dots (1c)$$

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

Let's briefly define a line

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

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

↑ Starting pt ↑ scalar ↓ Directional Vector

Let

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

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

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

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

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

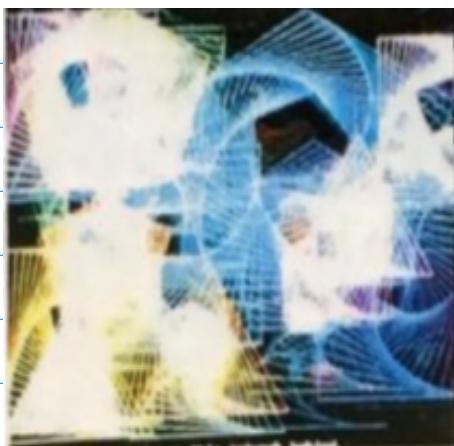


Fig 3a



Fig 3b

Screen Saver Design for LFC

2D G.E.

Rotating Squares And Trees.

Example: Design of Rotating Squares

Step 1. Define 4 vertices/pts

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

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

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

Based on the physical display device

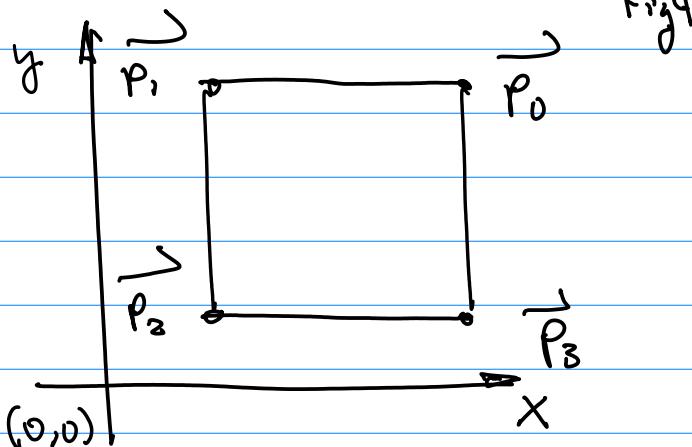


Fig 4

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

Step 2. Use

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

CmpE240

16

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

Sept. 27 (mon)

Homework, 2pts. Due 1 week from Today
Topics: 2D Screen Saver Design

Requirements:

a. Build LCD Hardware

Interface;

b. Input Sample code from
github/finalili/CmpE240

2018S-1D-LCD-DrawLine.

Modify the code to Display 2D

Rotating Squares Using 2D
Vector equation;

Submission:

c. Project (Zip, Exported)
d. Screenphoto

Submission to CANVAS.

Announcement:

Office hours — The 3:40-4:40 pm.

Due to SJSL off-Campus
Program.

Example: Continued from pp 15.

Step 2. Use Vector Equation
to find 4 pts

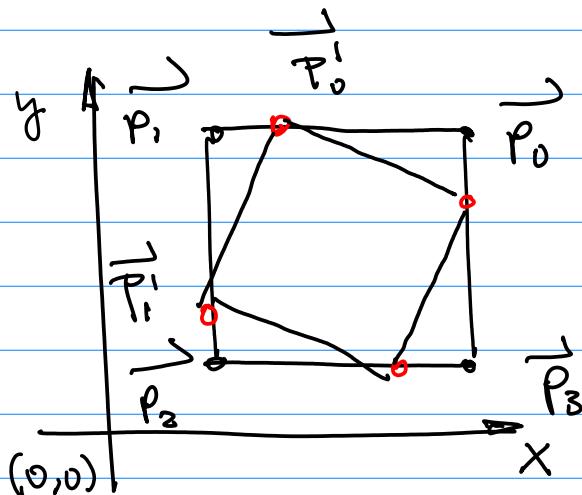


Fig. 1

Let $\lambda = 0.8$, for

line1 (\vec{P}_1 and \vec{P}_1'): Eqn(1), pp 15.

Calculate a point \vec{P}_0'

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

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

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

In Homework, level ≥ 10 .

Coding:

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

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

Hardware Implementation of LCD Interface.

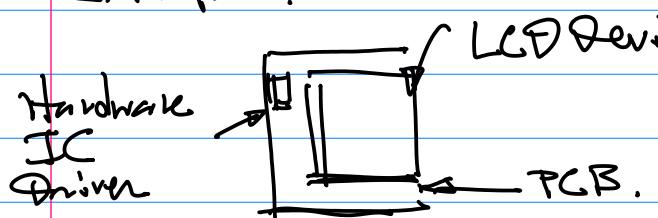


Fig.2



IC Driver

LCD Display

Display module

a. To Drive LCD Display
To Display a pixel } (x,y) Location

b. To provide feedBack { I(x,y) Intensity, and color
and Interface to CPU module.

(SPI Interface)

To Establish Interface, SPI (Serial Peripheral Interface)

Hardware pins of SPI : 3+1.

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

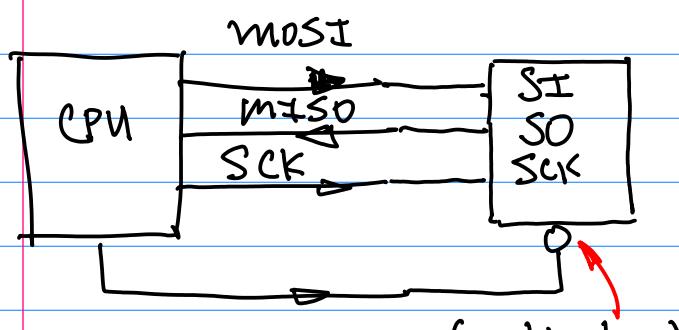


Fig.3.

Note: Mark the Direction of the Signal

Now, Consider the I/F to LCD module.

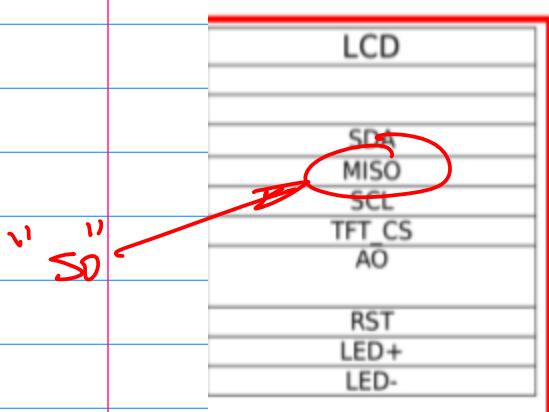
Ref: github.com/mahili/Cmpe240

Z018S-9-SPILCD...

LPC-1769	
Label	Pin
MOSI	P0.18
MISO	P0.17
SCK	P0.15
CS	P0.16
GPIO-DC (Data / command)	P0.21
GPIO-Reset	P0.22
3.3V	
GND	

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

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



Labels from the LCD Display.

SPI pins:

SDA, MOSI,
MISO
:

Type: Change MISO on LCD to SD,
etc.

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

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

Now, Software Part

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

2018S-10 - DrawLine

Example: Draw A Line Code

1. Color Definition. hex Digits

2 hexs for Each primitive

color

(red, green, blue)

Primitive Colors: r, g, b

2 hex Digits : min. 0

max: 255

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

2⁰ Bit Arrangement for the primitive

Colors: RGB = 2(hex)/2(hex)(2hex)

Identify module @ Line 285

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

Color

$\xrightarrow{P_0 \text{ or } P_i}$ $\xrightarrow{P_1 \text{ or } P_{i+1}}$

Match to Expr.(w) & (l_b)

to Build n Square One
Line at time.

Sept. 29 (Wed)

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

Before the Class.

Requirements:

1^o All work including prototype
Board, Programs, Report

CmpE440

However Team work is encouraged.

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

Submission:

- 1° Formal written Requirements

Rubrics will be posted on Line.

- 2° Submission on CANVAS.

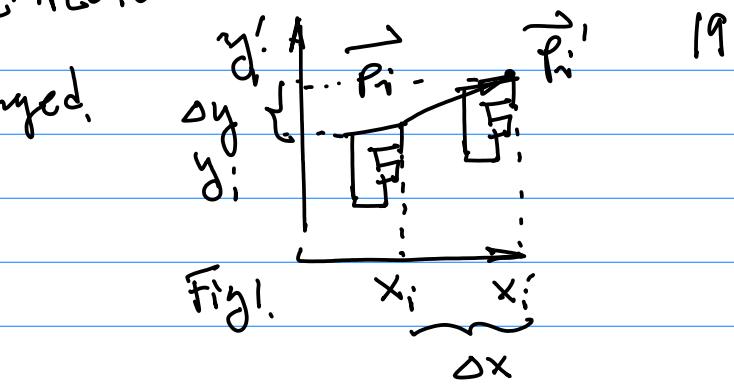
- 3° Source Code/Binary have to Exported Project, in zip.

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

- 5° 5 Seconds video

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

2D Transforms
Mathematical Formulation



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

Matrix T.

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

$$x'_i \stackrel{?}{=} x_i + \Delta x$$

After Before ... (1)

Similarly

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

After Before

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

Let's consider Rotation

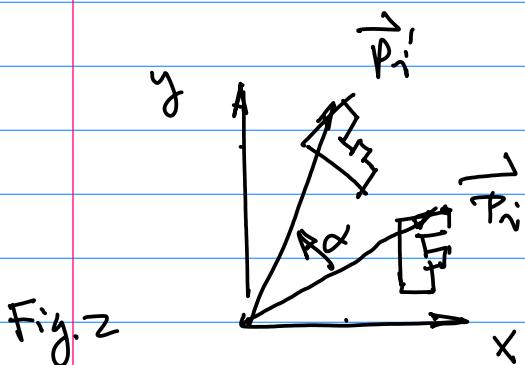


Fig. 2

Note: Counter Clockwise Rotation
"Positive" Angles

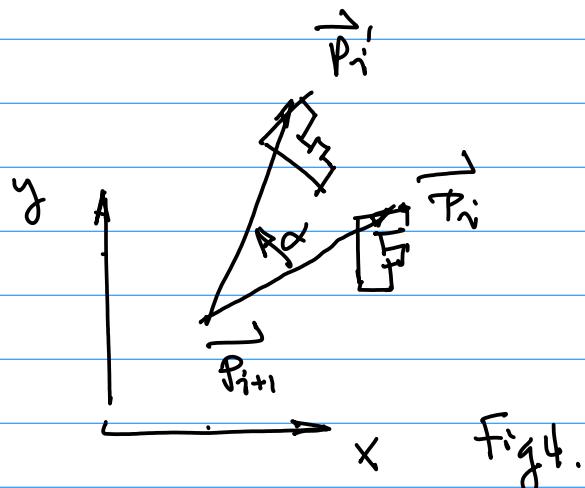


Fig. 4.

After

Before

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

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

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

Then, Perform Rotation;

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

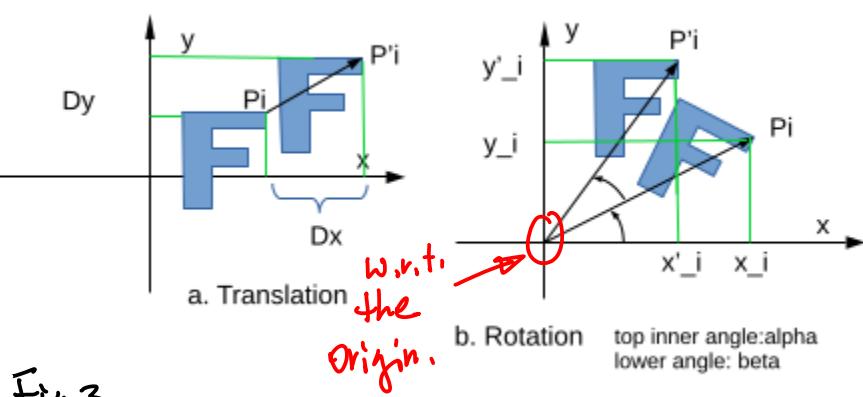


Fig. 3

From Eqn (4)

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

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

After

Before

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

where

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

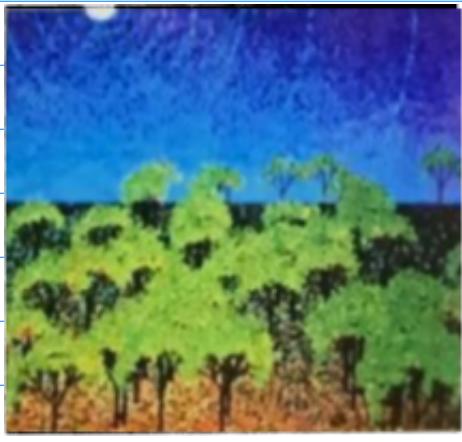


Fig. 5

Example: Use 2D Transforms to Create Trees shown Above

Step 1. Define Initial Points

Points to give a tree trunk.

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

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

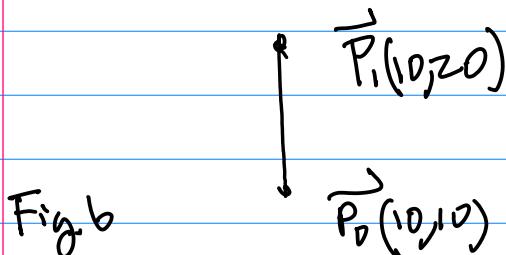


Fig. 6

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

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

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

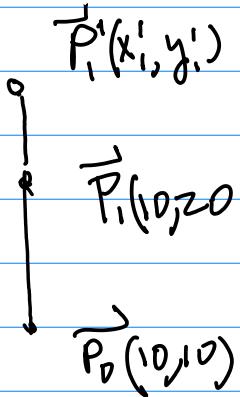


Fig. 7

make $\lambda = 0.8$.

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

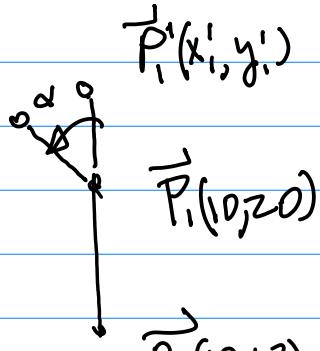


Fig. 8

Oct. 4 (Monday)

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

Example: Continued from Step 3.
First, Preprocess

Computation

22

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

Now, Similarly,

Next Rotation,

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

Post-Processing:

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

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

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

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

From Eqn(s) (7):

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

New X
New Y

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

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

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

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

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

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

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

Suppose LCD
Resolution is 120×100

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

Number of Pixels/Row
100: Rows.

Therefore

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

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

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

Limitation:

1° No Negative Value
in the System.

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

3° Not Portable

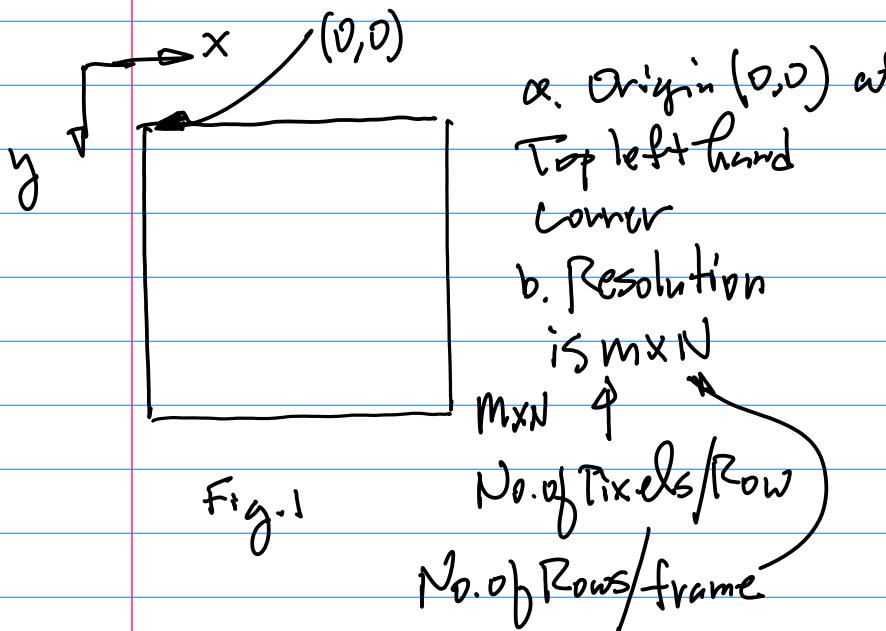
C/C++ Implementation

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

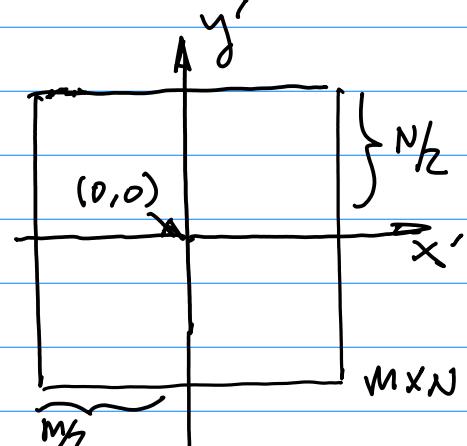
$$y'_i = \sin\theta \cdot x_i + \cos\theta \cdot y_i + \Delta X \sin\theta + \Delta Y \cos\theta - \Delta Y \quad \dots (b*)$$

Physical Display Coordinate

v.s. Virtual Display Coordinate



Now, Virtual Display coordinate



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

Transform physical Display to

Compe240

24

Virtual Display.

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

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

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

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

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

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

No. of Rows.

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

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

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

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

Virtual coordinate System,

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

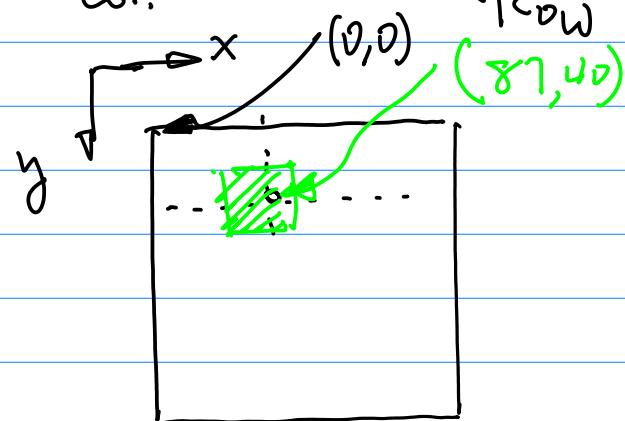
Patch with its center pixel
Equal to the Computation Result.

Example: Computation Result

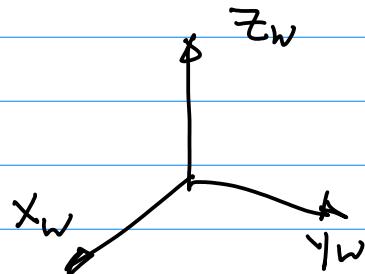
$$(x, y) = (87, 40)$$

Col.

Row



3D Graphics Processing Engine
Introduction: World Coordinate System



Outlook (Wed)

Fig. 1

Topics: 1° Transformation Pipeline

Note:

1° Affection Update

Posted on bit;

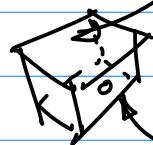
2° Handout on GPIO code,
please read it, understand
the code. It is required.

a "pin-Hole" model.

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

Enclose to form a virtual
camera.

Projection Plane
to form an image



pin-Hole

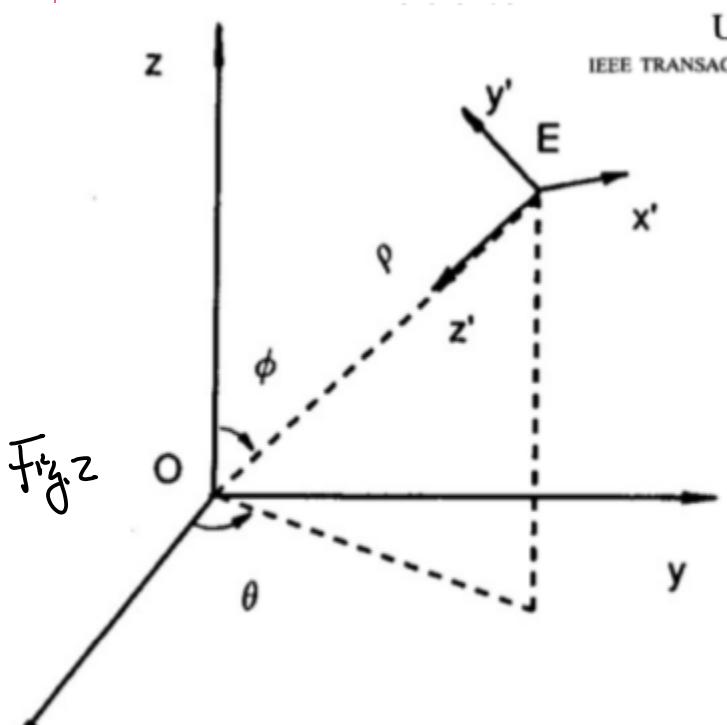
→ ED fixation direction of the
Virtual Camera

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

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

2° Left Hand System.

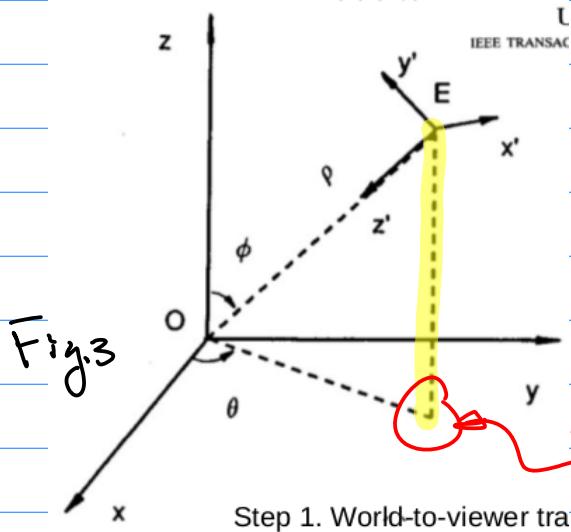
Relationship b/w $x_w-y_w-z_w$ and
 $x_e-y_e-z_e$ Systems to Allow
the Definition of Viewing 3D
Objects in a different
Perspective.



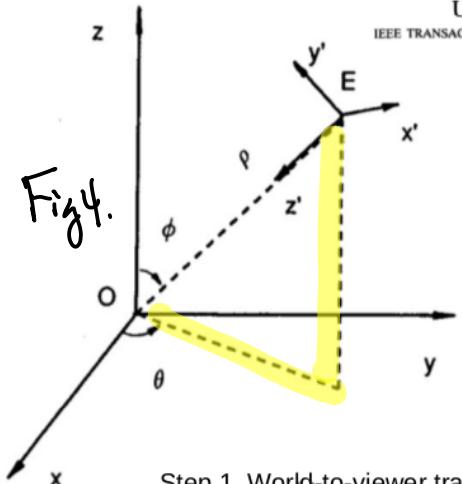
ComPE40

26

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



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



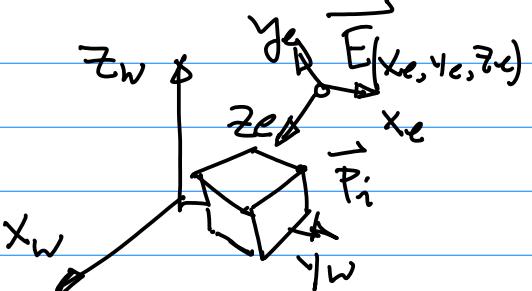
θ : (Theta) formed Between
Angle the 2nd Line
and x_w .

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

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

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

Example: A cube given in
the following figure:



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

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

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

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

After"in $x_w-y_w-z_w$ "

System Camera

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

Find Transformation Matrix

T (World-To-Viewer)

Transform Matrix.

Sol. For θ (theta) Angle

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

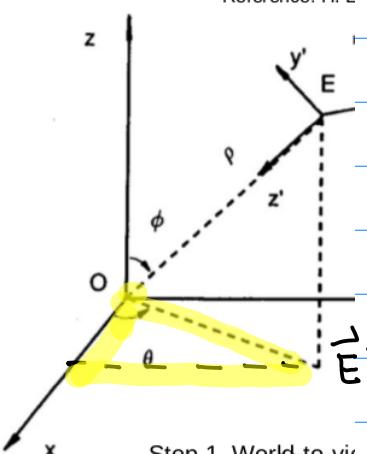
(From Fig 4, pp. 2b)

Draw A Line Passing

E' Perpendicular

to x_w to

form A Triangle



Step 1. World-to-vi

Consider $\sin \phi$

Oct. 11 (Monday)

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

a) gProg, C ; Architecture →

CPU Block Diagram →

SPRs → CPU Datasheet

Init & Config.

b) Drawing C :

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

Example: Suppose a given

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

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

Find Viewer coordinates of

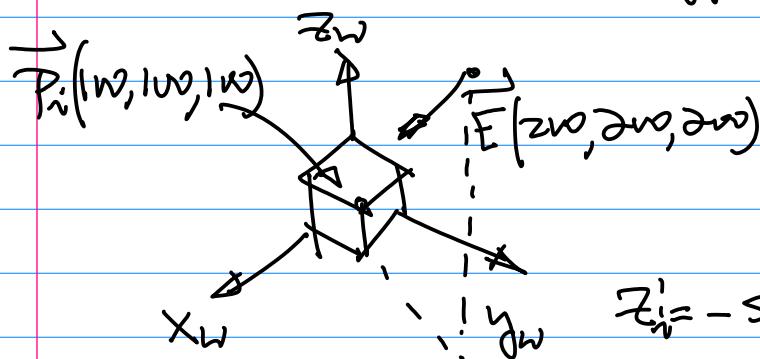
this cub. $P_i' = ?$

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

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

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

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



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

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

From Eqn (2b) pp. 27

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

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

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

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

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

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

Optional Homework:

Write C code for Eqn (1).

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

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

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

Handout
SSP.C

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

Handout 2021F-11Z - ~ Review

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

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

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

Format of the lectures.

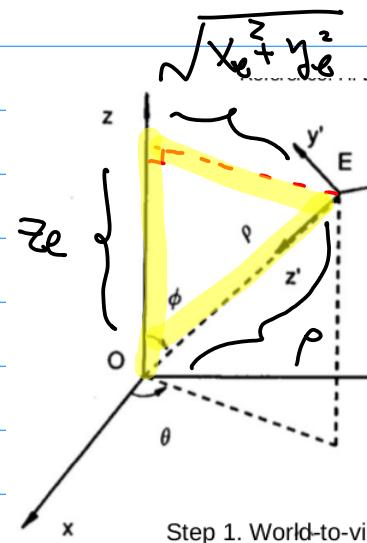
CPU Architectures \rightarrow CPU Dashed

Coding
Implementation

And Prototype Implementation

Today's Topics: Transformation

Pipeline.



Step 1. World-to-vit

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

Example: Continue from previous
discussion.

$$\begin{aligned} \sin \phi &= \sqrt{x_e^2 + y_e^2} / \rho \\ &= \frac{\sqrt{x_e^2 + y_e^2}}{\sqrt{x_e^2 + y_e^2 + z_e^2}} \quad | \\ &= \frac{2\sqrt{2}\sqrt{2}}{2\sqrt{2}\sqrt{3}} = \frac{\sqrt{2}\sqrt{3}}{3} = \sqrt{6}/3 \end{aligned}$$

$$\cos \phi = \frac{z_e}{\rho} = \frac{2\sqrt{2}}{2\sqrt{2}\sqrt{3}} = \sqrt{3}/3$$

2nd Step for Transformation

Pipeline \Rightarrow Perspective

Projection

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

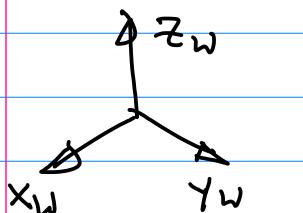
After Perspective Project,

$\vec{P}_i''(x_i'', y_i'')$ (2D pt !)

$$\begin{cases} x_i'' = \frac{D}{z'_i} x'_i & \dots \underline{(a)} \\ y_i'' = \frac{D}{z'_i} y'_i & \dots \underline{(b)} \end{cases}$$

(x_i'', y_i'') ON 2D LCD Display
With Depth Perception

D : focal length.



$D \approx 20-30$

Draw ON 2D
Display.

Tree Creation

Oct. 13 (Wed) Class Repo

Ref: <https://github.com/hanulili/CMPE240>

Architectural Aspects ON

Display Driver Design (Ref1,
Ref2) \rightarrow S.P.I. (Architecture

+ Software Coding, SPRs

(CPU Datasheet) \rightarrow Handout
on S.P.I. program.

Road Map Before the midterm

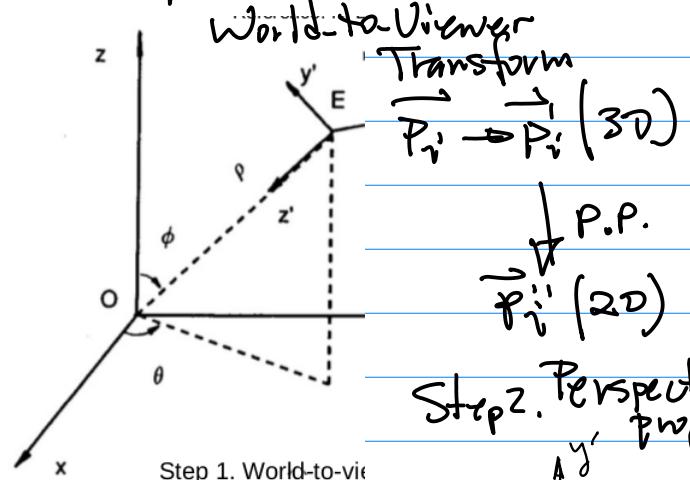
CPU Architecture \rightarrow GPGPU \rightarrow 2D

Display 2D G.E. \leftarrow Vector
Driver \leftarrow Graphics
Design Processing Engine

\uparrow Hardware \rightarrow S.P.I. I/F
Coding/
Programming

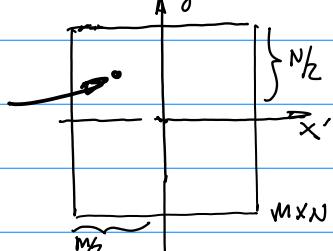
3D
Introduction

Example:



Step 2. Perspective
projection

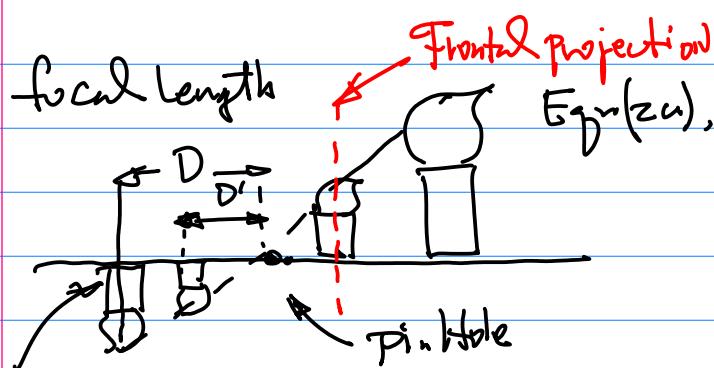
\vec{P}_i''
 (x_i'', y_i'')



2021F-113-LCD-TFT (ThinFilmTransistor).jpg

2021F-114-display-NEC-3P5-LCD-68775.pdf

2021S-100-accessible CMPE240-S21-v2-HarryLi.pdf



Back
projection
plane

Fig. 1.

Compute Perspective projection

Let choose $D=20$.

And

$$\begin{aligned} P'_i(x'_i, y'_i, z'_i) = \\ \left(0, 1w\frac{\sqrt{b}}{3}, 1w\frac{\sqrt{3}}{3}\right), \text{ PP2.7} \end{aligned}$$

Find Perspective Projection

From Eq(2a)

$$x''_i = \left(\frac{D}{z'_i}\right) x'_i = \left(\frac{20}{z'_i}\right) x'_i$$

$$= \left(\frac{20}{1w\frac{\sqrt{3}}{3}}\right) \cdot x'_i = \left(\frac{20}{1w\frac{\sqrt{3}}{3}}\right) \cdot 0 = 0$$

$$y''_i = \left(\frac{D}{z'_i}\right) y'_i = \left(\frac{20}{z'_i}\right) y'_i$$

$$= \left(\frac{20}{1w\frac{\sqrt{3}}{3}}\right) \cdot 1w\frac{\sqrt{b}}{2} \quad \text{In Virtual Display Coordinate}$$

Display coordinates
Transform to find
physical display value
for your target platform.

Design of Display Driver Hardware & Architecture

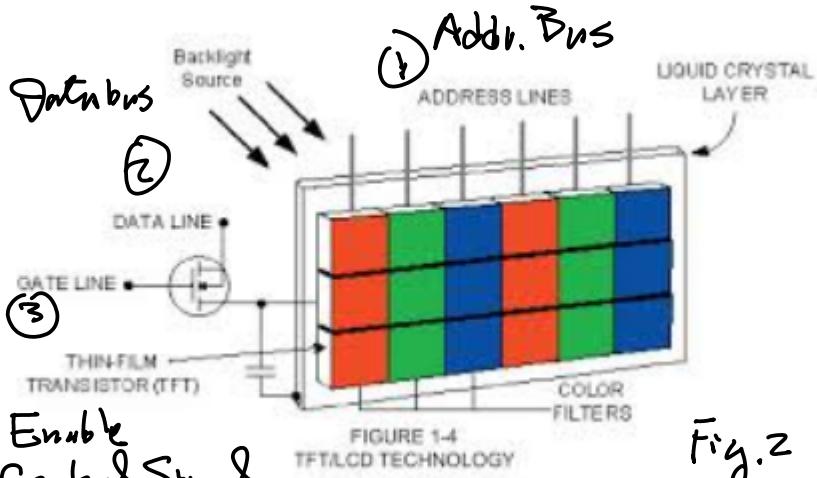
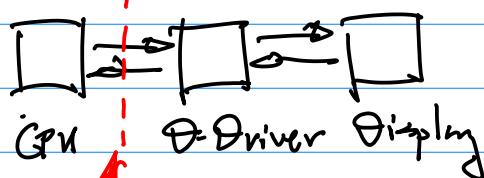
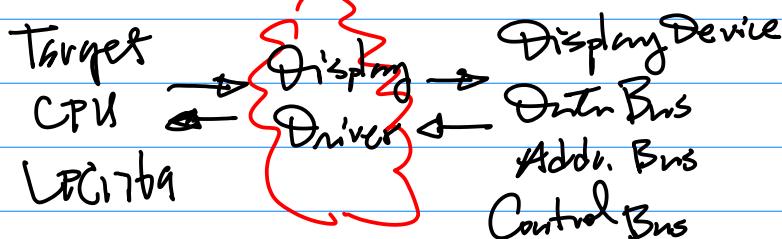


Fig. 2



Interface Part, SPI

Fig. 3

In Virtual Display Coordinate } Hardware "3+1"
Software

Step 3. For Implementation.

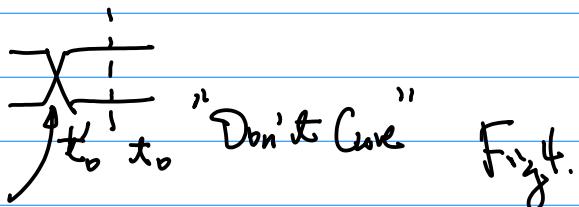
Use Virtual-To-Physical

"3+1" MOSI : Master Out Slave In
MISO : Master In Slave Out
SCK : Slave Out

SCK: Serial Clock.

SS_L: Enable

SPI Waveform (Protocol)



at t_0 time instant, "Changing State".

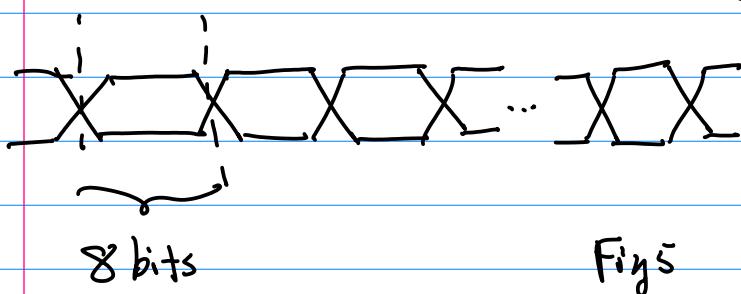
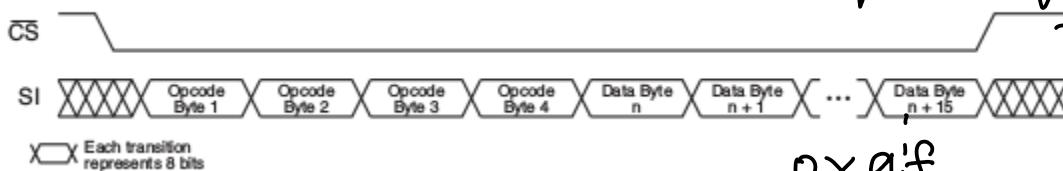


Figure 9-3. Program Sector Protection Register

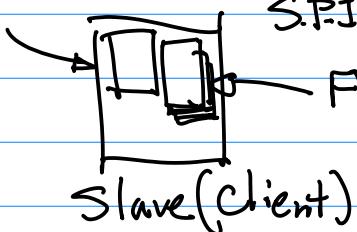


SPI protocol, 3 Fields

- { Opcode : Field 1, Instruction Field
- Address Field, Field 2
- Data (Payload), Field 3, Data Field

Controller (MCU)

Control Register



Depends on Application Need,
we could have just Instruction
Field for SPI Communication.
Or,

We could have complete 3
Fields, e.g. Opcode (Instruction)
Field, Addr. Field, and Data Field.

Use Logic Analyzer or DSC.
to Capture the Waveform for
Debugging.

Note: Each Field, for Example

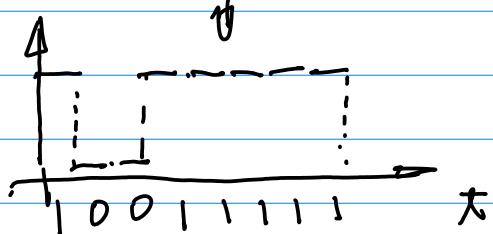
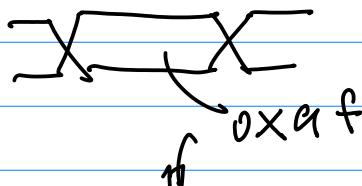
Opcode
0x9f

To ask for Manufacturer's ID &
Product ID.

0x9f

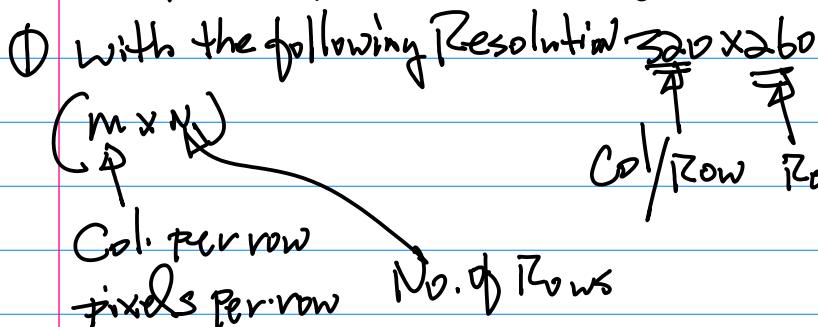
10011111

↓ Waveform



Suppose Display Device

Example: Suppose a Display Device



NEC NEC LCD Technologies, Ltd.

2. GENERAL SPECIFICATIONS

Display area	53.64 (H) × 71.52 (V) mm
Diagonal size of display	8.9cm (3.5 inches)
Drive system	a-Si TFT active matrix
Display color	262,144 colors
Pixel	240 (H) × 320 (V) pixels
Pixel arrangement	RGR (Red dot Green dot Blue)

Homework Requirements

- ① Read the source code then, from Schematic of the LPC1769 Design, find the proper value to perform init & Config. to make JZ-27 GPIO pin as an input, JZ-23 Output.

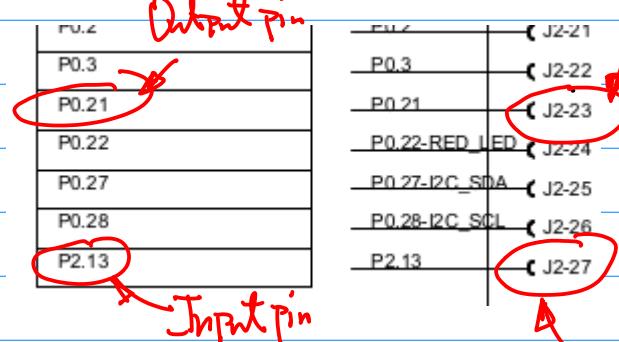
LPCXpresso1769_CD_revD(1).pdf

NEC 0.85 LCD 0077E-LK

② And display 30 FPS (Frame/second)

③ pixel depth (color) 24 Bits

V, G, B (red, green, blue) 8 bits each.



Find 1. Bit Rate for SPI Interface?

Submission.

1° Schematic shows GPIO Input & Output Pin

2° Output Testing Ckt (LED, Resistor, Resistor Value)

(Calculation)

3° Show Main program

4° Photo of (GPIO Testing / Output testing Result)

5° Submission to CANVAS.

```
//Initialize the port and pin as outputs.
void GPIOinitOut(uint8_t portNum, uint32_t pinNum)
{
    if (portNum == 0)
    {
        LPC_GPIO0->FIODIR |= (1 << pinNum);
    }
    else if (portNum == 1)
    {
        LPC_GPIO1->FIODIR |= (1 << pinNum);
    }
    else if (portNum == 2)
    {
        LPC_GPIO2->FIODIR |= (1 << pinNum);
    }
}
```

2021F-111-handout-gpio-dotC.pdf

Oct. 25th

CmRE24D

三

2. Midterm scheduled on Nov. 10th
= (Wed), Close Book, Close Notes.

Requirements: a On-Line, Video
has to be at all time, b Scan/Take
photos of your papers, Convert to
One pdf file. c Naming for the
midterm paper:

first - Last - 4 digits ID - mid - Comp 240.pdf

Submission to CANVAS with Deadline

Exams One Home Example.

Example: Find SPI Clock Rate (Continued from PP33).

2° Find SFR (Special Purpose)

Register) Responsible for Init's Config
to fulfill this task?

30 Define the init & Config Binary Pattern, Write/modify C-Code to

Realize this function.

Sol 1° find SPI clock Rate?

Δt is Related to the Frame Rate

Frame rate : 30 FPS (Frame per Second)

$$\Delta t = \frac{1}{f} = \frac{1}{30} = 33.3 \times 10^{-3} \text{ Sec.}$$

SPI protocol: 3 Fields \rightarrow Payload (3rd)

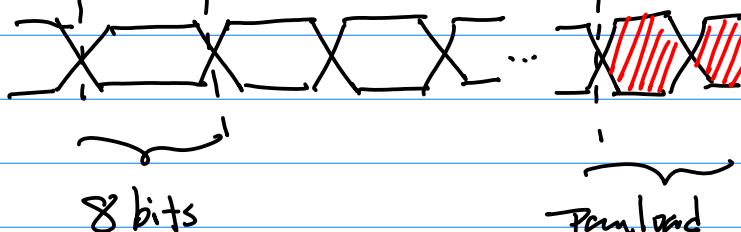


Fig. 1. Graphical Data Display

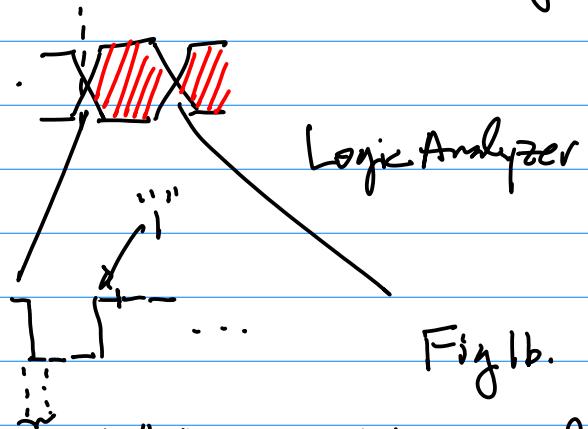


Fig 1b.

1 bit, bit "0" 1 bit transmitted per time needed to fulfill Requirements from 1° to 3°
So,

$$\begin{aligned}
 & \underbrace{320 \times 260 \times 30 \times 2^4}_{\text{Resolution FPS Color}} = 2^8 \cdot 2^8 \cdot 2^5 \cdot 2^5 \\
 & = 2^{10} \cdot 2^{10} = 2^{10} \cdot 2^{10} \cdot 2^6 \\
 & \approx 2^8 \times 2^8 \cdot 2^5 \\
 & = 2^6 \cdot \underbrace{1K}_{1 \text{ meg.}} \\
 & = 64 \text{ mbps}
 \end{aligned}$$

(Need to transfer bit mag. bits
Data to the Display Device)
from L+CL7ba

From Fig 1b, One bit per clock,
therefore SPI Clock rate :

$$f_{\text{SPI}} = f_{\text{SCLK}} \approx 64 \times 10^6 \quad \dots(1)$$

Note: If Resolution is Reduced
from 360x240 to 160x120 for example
then Bit Rate Roughly is Reduced to
its quart of the original.

$$f'_{\text{SCLK}} \approx f_{\text{SCLK}} / 4 = \frac{64}{4} \text{ MHz} = 16 \text{ MHz}$$

2° Find SPR Responsible, TP431, Datasheet

$$\text{SSPnCR}\phi[3:\phi] = 0111 \quad (\text{B})$$

$$\text{SSPnCR}\phi[5:4] = 00 \quad (\text{B}) \text{ for SPI.}$$

Next 2 Bits are Set to ϕ By
default.

$$\text{SSPnCR}\phi[6] = \text{SSPnCR}\phi[7] = 0$$

Serial Clock Rate

SCR 8 bits, 256 Different
Combinations.

the clock line.

Serial Clock Rate. The number of prescaler-output clocks per bit on the bus, minus one. Given that CPSDVSR is the prescale divider, and the APB clock PCLK clocks the prescaler, the bit frequency is $\text{PCLK} / (\text{CPSDVSR} \times [\text{SCR} + 1])$.

Reserved, user software should not write ones to reserved bits.
The value read from a reserved bit is not defined.

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

... (4)

Oct. 20 (Wed)

Note: mid-term Scheduled
on Nov. 10 (We).

Close Book / Close Notes

One page Formula Allowed

Example: CPU Datasheet, TP431.

Suppose CPU Operates 200MHz.

SPR Setting for PCLK is set to
deliver $\frac{1}{4}$ of System Clock, \rightarrow

$$\begin{aligned} \text{PCLK} &= \frac{\text{CLK}_s}{4} = \frac{200}{4} \times 10^6 \\ &= 50 \text{ MHz} \end{aligned}$$

Table 371: SSPn Control Register 0 (SSP0CR0 - address 0x4008 8000, SSP1CR0 - 0x4003 0000) bit description

① SuperSet for SPI

② Control Register, CR ϕ

③

SSP ϕ CR ϕ : 0x4008-8000
prefix Root

r ϕ

4
r31

Fig.2

SSP ϕ CR ϕ [3: ϕ] : DSS (Data Size Select), 8bits

$$f_{\text{SPI}} = 1 \text{ b} \text{mHz} \text{ (for } 1 \text{b} \text{ox } 120 \text{ Resolution)}$$

Find SCR settings to realize this SPI Clock (BitRate of 1bmbps)

So 1: from Eqn (4), pp35

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

Substitute the given conditions

$$f_{\text{SPI}} = 1 \text{b} \times 10^6, \text{ PCLK} = 50 \text{mHz} = 50 \times 10^6$$

$$1 \text{b} \times 10^6 = \frac{50 \times 10^6}{\text{CPSDVSR} (\text{SCR}+1)} \dots (1-b)$$

SCR field from SPI Control

Register CPSR [5:8], And CPSDVSR is a Special Purpose

Register [2, 254]

① pp433
②

Table 375: SSPn Clock Prescale Register (SSP0CPSR - address 0x4008 8010, SSP1CPSR - 0x4003 0010) bit description

Bit	Symbol	Description	Reset Value
7:0	CPSDVSR	This even value between 2 and 254, by which SSP_PCLK is divided to yield the prescaler output clock. Bit 0 always reads as 0.	0
31:8	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

Important: the SSPnCPSR value must be properly initialized or the SSP controller will not be able to transmit data correctly.

SSPnCPSR[7:0]

$$\text{from Eqn}(1-b): 1b = \frac{50}{\text{CPSDVSR} (\text{SCR}+1)}$$

Design By selecting a meaningful combination of CPSDVSR, and SCR
[2, 254] [0, 255]

Trial-by-Error, Iteration.

Let CPSDVSR = 128, Hence

$$1b = \frac{50}{128(\text{SCR}+1)}, \text{ OR}$$

$$1b = \frac{50}{64(\text{SCR}+1)}$$

$$1b = \frac{25}{32(\text{SCR}+1)}$$

$$\frac{1b \times 32}{25} = \frac{1}{\text{SCR}+1}, \text{ SCR}+1 = \frac{25}{1b \times 32}$$

If $\frac{25}{1b \times 32}$ falls in [0, 255]? Not

Exactly, so 2nd iteration; try

to Reduce CPSDVSR

try to make it equal

to 2

from Eqn(1-b)

$$1b = \frac{50}{2(\text{SCR}+1)}$$

$$\text{SCR}+1 = \frac{50}{2 \times 1b} = \frac{50}{32}$$

CMPEDTO

37

$$SCR = \frac{50 - 32}{32} = \frac{18}{32} = 0.56 \approx 1$$

Bit Rate of 1b/mbps $\rightarrow f_{SPI} = 1b \times 10^6 \text{ (MHz)}$
2. Supports Resolution 160x120, 30FPS,
Pixel depth is 24 bpp.

Compare $\frac{18}{32}$ to $\frac{25}{16 \times 32}$, this result is

better.

SCR ∈ [0, 255]

Check if iteration 3 is needed,
Since we have the CPSDVSR = 2
minimum, therefore no further
iterations.

$$\therefore SCR \approx 1$$

$$CRD[5:8] = 0x1 = \begin{pmatrix} 0 & 0 \\ \cancel{1} & \cancel{1} \end{pmatrix}$$

4 bits 4 Bits (Lower) Together
(Upper)

From CPU CRD.

$$CRD[3:16] = 0x0$$

PP431.

Hence, Control Register CRD has
to be set to the following binary
pattern

CRD[31:0] :

$$CRD[3:0] = 0111 = 0x7 \text{ Bits/Trans.}$$

$$CRD[5:4] = 00 \text{ for SPI}$$

$$CRD[7:6] = 00 \text{ By default}$$

$$CRD[15:8] = 0000; 0001 = 0x1$$

$$CRD[3:16] = 0 \dots 0 = 0x00$$

$$CRD[31:0] = 0x0000 - 0101 \\ = 0x0101 \\ = 0x101$$

Note: Technical Spec.

15:8 SCR	the clock line.
31:8 - 16 ¹⁶ Type	Serial Clock Rate. The number of prescaler-output clocks per bit on the bus, minus one. Given that CPSDVSR is the prescale divider, and the APB clock PCLK clocks the prescaler, the bit frequency is PCLK / (CPSDVSR × [SCR+1]).
	Reserved, user software should not write ones to reserved bits. NA The value read from a reserved bit is not defined.

6.2 SSPn Control Register 1 (SSP0CR1 - 0x4008 8004, SSP1CR1 -

Hence, we have the following Technical
Specification of this Design:

1. Support SPI Display Driver function with

Now, C/C++ Implementation
(Example from DrawLine.c)

(↑ ↓)

Binary Pattern
for Init & Config

CmpEx40

38

Required Init & Config By Control Registers CR0, CR1
SSPLCR0, SSPLCR1

```

218 *****
219 void SSP1Init( void )
220 {
221     uint8_t i, Dummy=Dummy;
222
223     /* Enable AHB clock to the SSP1. */
224     LPC_SC->PCONF |= (0x1<<10);
225
226     /* Further divider is needed on SSP1 clock. Using default divided by 8 */
227     LPC_SC->PCLKSEL0 |= (0x3<<20); // PCLKSEL0 for SSP1 Clock.
228
229     /* P0.6~0.9 as SSP1 */
230     LPC_PINCON->PINSEL0 &= ~(0x3<<12) | (0x3<<14) | (0x3<<16) | (0x3<<18));
231     LPC_PINCON->PINSEL0 |= ((0x2<<12) | (0x2<<14) | (0x2<<16) | (0x2<<18));
232
233     /*#if !USE_CS
234     LPC_PINCON->PINSEL0 &= ~(0x3<<12);
235     LPC_GPIO0->FIODIR |= (0x1<<6);
236     #endif
237
238     /* Set DSS data to 8-bit, Frame format SPI, CPOL = 0, CPHA = 0, and SCR is 15 */
239     LPC_SSP1->CR0 = 0x0707;
240
241     /* SSPCPSR clock prescale register, master mode, minimum divisor is 0x02 */
242

```

LPC_SC → PCONF SPR: peripheral controller timer

② $\frac{1}{2}$ bit control.

③ $\frac{1}{2}$ bit control.

④ $\frac{1}{2}$ bit control.

⑤ $\frac{1}{2}$ bit control.

$\approx \sim$ "Negation" $\rightarrow 11 \rightarrow \sim$ "of 11" $\rightarrow 00$

Clear 2 bits \leftarrow AND " & " \Rightarrow

Set 2 bits as 1D" ($= 0x2$)

Example: for PINSEL0, refer to CPU Datasheet

8.1 How to read this chapter

Table 75 shows the functions of the PINSEL registers in the LPC176x/5x

Table 75. Summary of PINSEL registers

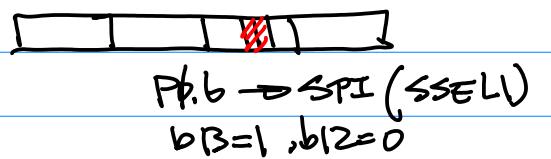
Register	Controls	Table
PINSEL0	P0[15:0]	Table 80
PINSEL1	P0 [31:16]	Table 81
PINSEL2	P1 [15:0] (Ethernet)	Table 82
PINSEL3	P1 [31:16]	Table 83
PINSEL4	P2 [15:0]	Table 84
PINSEL5	P2 [31:16]	not used
PINSEL6	P3 [15:0]	not used

TP117 ① PINSEL0 Select Define Multiplexed functions

Table 80. Pin function select register 0 (PINSEL0 - address 0x4002 C000) bit des

PINSEL0	Pin name	Function when 00	Function when 01	Function when 10	Function when 11
1:0	P0.0	GPIO Port 0.0	RD1	TXD3	SDA1
3:2	P0.1	GPIO Port 0.1	TD1	RXD3	SCL1
5:4	P0.2	GPIO Port 0.2	TXD0	AD0.7	Reserved
7:6	P0.3	GPIO Port 0.3	RXD0	AD0.6	Reserved
9:8	P0.4	GPIO Port 0.4	I2SRX_CLK	RD2	CAP2.0
11:10	P0.5	GPIO Port 0.5	I2SRX_WS	TD2	CAP2.1
13:12	P0.6	GPIO Port 0.6	I2SRX_SDA	SSEL1	MAT2.0
15:14	P0.7	GPIO Port 0.7	I2STX_CLK	SCK1	MAT2.1
17:16	P0.8	GPIO Port 0.8	I2STX_WS	MISO1	MAT2.2

Program line 230.
bit b12



Out: 25 (Monday).

Note: 1^o Prep. for mid-term Exam.

Nov. 10 (Wed). on Zoom.

(1) Video of the entire session.

Mandatory. In emergency,
Text to (BSO) 400-111b.

(2) One page formula is allowed.

Submit the formula sheet together
with exam papers By the end
of Exam. No verbal description,
No Block Diagrams allowed.

Put your First, Last Name with
4 Digits of Student ID. on the
Sheet.

Close Book, Close Note, Data sheet
if needed, it will be provided

together within the exam paper.

(3) Use Blank Printer Paper, No
Class Notepad is allowed.

1^o Put your First, Last Name
on the top right corner
on each page ; together with
4 Digits ID.

2^o Scan paper(s) and then

place them together, use software
to form/ generate One PDF
document.

<https://www.google.com/search?client=ubuntu&channel=fs&q=convert+multiple+jpegs+to+pdf&ie=utf-8&oe=utf-8>

Example :

<https://jpg2pdf.com> ::

JPG to PDF – Convert JPG Images

This free online JPG to PDF converter allows to c

3^o 5-10 minutes Extra time for preparing
CANVAS Submission.

4^o One PDF Document, then zip it . With
the Name as follows :

First Name - Last Name - 4 Digits - mid - Cmpe240 -
yy-mm-dd.zip.

Exercise on Wednesday. Bring a Blank paper
to Class, Ready for Scanning, Submission to
CANVAS.

Continue on S.P.I. program (SPR. GRP 4)
↓ Handout of the programs.
Init & Config.

Example: Program Line 238, pp.38

① SPI's Special Purpose Register, Control Registers

40

```

232 //#if !USE_CS
233 LPC_PINCON->PINSEL0 &= ~(0x3<<12);
234 LPC_GPIO0->FIODIR |= (0x1<<6); /* P0.6 defined as GPIO and Outputs */
235 //endif
236
237 /* Set DSS data to 8-bit, Frame format SPI, CPOL = 0, CPHA = 0, and SCR is 15 */
238 LPC_SSP1->CR0 = 0x0707;
239
240 /* SSPCPSR clock prescale register, master mode, minimum divisor is 0x02 */
241 LPC_SSP1->CPSR = 0x2;

```

② Naming Convention Revisit: $LPC_SSP1 \rightarrow CR0$

Product Family Peripherial Controller Pointer(Addr.)
 Peripheral Controller to CR0
 "1" Enumeration

more than one SSPx

③ Binary Pattern for SSP1 init & config.

Technical Specification (Verbalized OR Tabulation)

Binary Pattern

$$LPC_SSP1 \rightarrow CR0 = 0x0707; (=0x0000\overset{1}{|}0707)$$

Upper/blocks
Unused.
 $CR0[31:16]$

$\overset{00000111}{00000111}$

SCR

8 bit Target Data

$$f_{SPI} = \frac{PCLK}{CPSDIVSR (SCR+1)} \quad \text{Eqn(1), PP36.}$$

Line 241, Defines Prescaler Needed in f_{SPI} .

Example: Now, Consider the

Summary: SPI CR0 Init & Config. Done. Display Driver Design task:
 (SSPx)

Handout on the Class github.

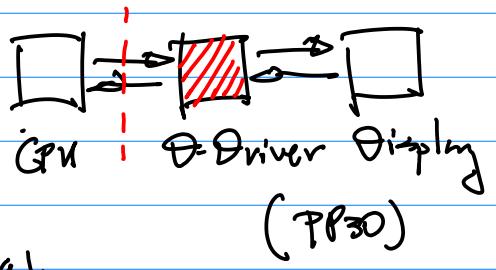


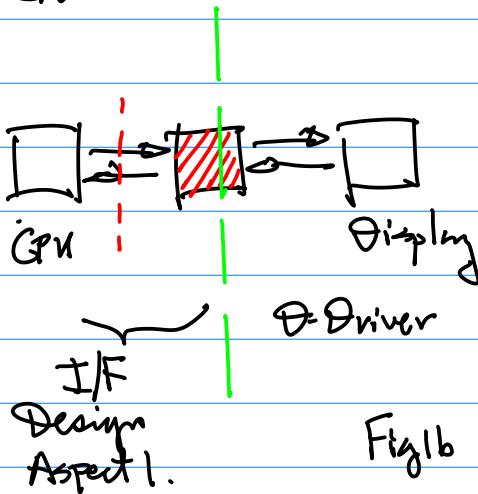
Fig1.

2021F-111-handout-gpio-dotC.pdf

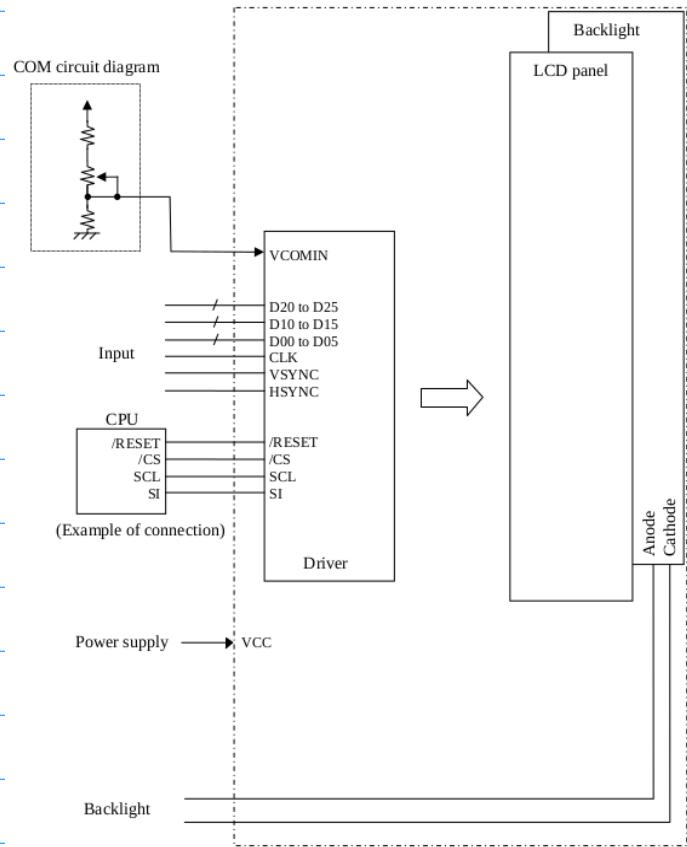
2021F-111b-handout-drawline-dotC.pdf

2021F-111c-handout-ssp-dotC.pdf

Design Aspect 1 : Interface Between the Device Driver and the CPU.



Design Aspect 1.



Finish this Design Based on SPI
Protocol "3+1"

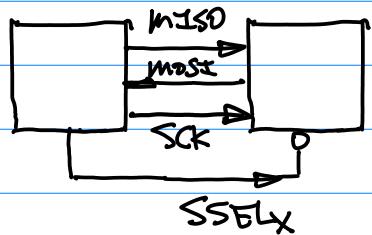


Fig. 1c

The 2nd Aspect of the Driver Design is Related Display Panel.

[2021F-114-display-NEC-3P5-LCD-68775.pdf](#)

Note:

- 1° Display panel consists of the following pins (e.g., Subsystems)
 - 1.1° Data Bus (Bi-directional) 16 bits to 32 bits
 - 1.2° Address Bus (optional)
 - Timing Signals to serve the purpose of Address Bus.
 - Frame-Sync.
 - H-Sync. (Horizontal)
 - Data-Sync.

NEC NEC LCD Technologies, Ltd.

TFT COLOR LCD MODULE

NL2432HC22-40J

8.9cm (3.5 Type)
QVGA

DATA SHEET
DOD-PP-0129 (3rd edition)

[2021F-115b-homework-review1-cmpe240...](#)