

CMPE240 Spring 22

Jan 26 (Wed)

Today's Topics: Introduction

& Organizational meeting.

Harry Li, Ph.D.

Green Sheet: On-Line from github

<https://github.com/hualili/CMPE240-Adv-Microprocessors>

hualili / CMPE240-Adv-Microprocessors Public

Code Issues 1 Pull requests Actions Projects Wiki

master 1 branch 0 tags

hualili Add files via upload

2018F Add files via upload

1769 patch.zip Add files via upload

2022S-112-project-Lecture-Reflection-Handouts.pdf

2022S-100-accessible_CMPE240-HarryLi.pdf CMSIS_CORE_LPC17xx.tar.gz

Naming convention: Year + Semester + ID + Name of the Doc.

E-mail: hua.li@sjsu.edu

Text message: (650) 400-1116

Office Hours: Mondays & Wednesdays
4:30 - 5:30 PM.

Office hours Zoom link: Join Zoom Meeting

[https://us04web.zoom.us/j/9841607683?
pwd=UIA3aEk1TnV4bjNLQk5CQkw0dDk4UT09](https://us04web.zoom.us/j/9841607683?pwd=UIA3aEk1TnV4bjNLQk5CQkw0dDk4UT09)

Meeting ID: 984 160 7683 Passcode:
121092

TextBooks + Ref

1.

No text Book, But NXP CPU Datasheet
is utilized as a Base Line Ref

2. SCH Design of the CPU module.

FR5 (Rev.D.)

Prototype Board: Each person will
Build his/her Prototype System.

Team work is encouraged, form 4 people
team for this Class. However all the
work has to be done individually.

Grading: Midterm 30%

Format: Written Exam But

Need to have prototype ready to execute
programs, And get photos of your
Prototype Board.

Final Exam: 40%, Similar format as
the midterm.

Homework / Projects Counts Another 30%.

Written Announcement on WhiteBoard
(Lecture Notes) And SJSU CANVAS.

Late Project submission will have 1 pt.

Penalty per each Lecture Day.

Introduction.

1. Bill of material for LFC17ba
Prototype Board Design.

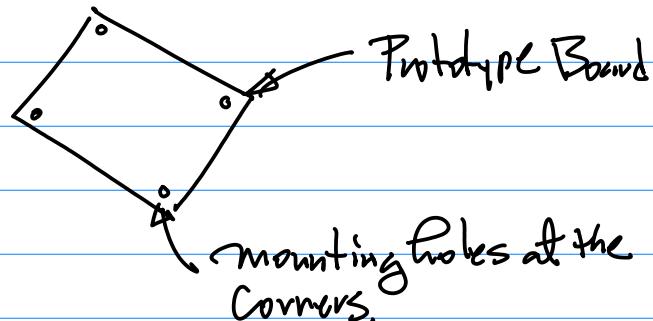
Jan 31 (Monday)

Today's Topic: Introduction.

Prototype System.

Task 1. Form 4 Person Team

Work Together throughout the entire Semester. By this Wednesday.



Stand-offs: (Legs) for the Board.

Prototype System.

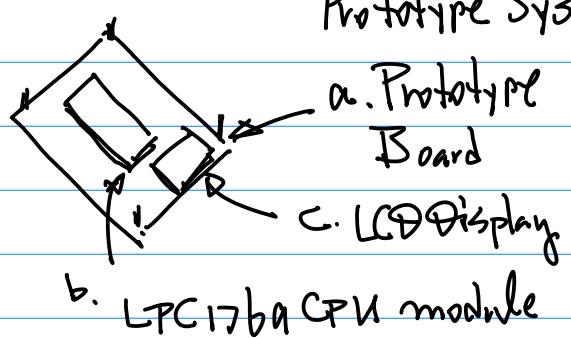


Fig. 1

Note: a. Adequate Size

Not Too big, to host LPC1769

module & LCD Display module, plus "Glue" logic.

"Glue Logic" GPIO Circuit as a part of the "Glue"

Ethernet Connector, RJ45, in the future for possible networking Applications,

TCP/IP, micro-Web Server.

4" x 3" Or Similar Size.

16 cm x 11 1/2 cm.



Fig. 2 CPU
Digi-Key, or mouser Electronics

CPU: Cortex M3

c. LCD Display module

SPI (Serial Peripheral Interface)

Software Driver function(s) are provided/Accesible

- LPC 1769
 - Color TFT LCD display
- Resolution : 128x160,
Pixel Depth: 18-bit (262,144) colors
- Controller: ST7735
Interface: SPI interface

LCD Pins

LPC1769 Pins

From the class github

LCD module

- 2021F-113-LCD-TFT (ThinFilmTransistor).jpg
- 2021F-114-display-NEC-3P5-LCD-68775.pdf

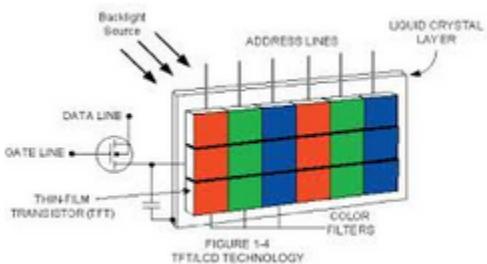


Fig.3

Components for:

Power Unit Design Need

GPIO (General Purpose I/O)

LPC1769 is powered with 3 options.

Option 1: USB Cable Connection to provide power from the host (PC) to LPC1769 CPU module.

for Debugging / Testing Purpose But not for Deployment

Option 2: External Power to the prototype System. To Allow you Deployment of the Prototype System. → Mandatory, By

the last project, each prototype will have to be deployed with External Power.

7.5V ~ 9V DC @
1500 mA ~ 4000 mA

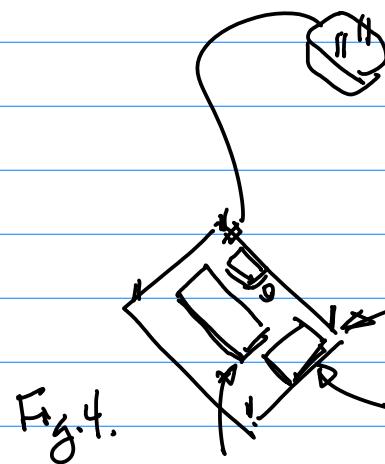


Fig.4.

Prototype System.

a. Prototype Board

c. LCD Display

b. LPC1769 CPU module

Build Option 2 Power Unit Circuit is required Now, Before the first Project.

Components for the Power Unit

① Wall-mount Adapter.

Spec. 7.5 ~ 9VDC

1500 mA ~ 4000 mA.

Or Battery Pack.



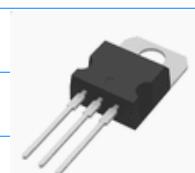
Fig.5

② Red LED indicator, to Show Power is on/off.

$V_{LED} \approx 1.2 \text{ VDC}$; $8 \text{~} 10 \text{ mA}$

③ Power Regulation IC 7805, or 1117.

Fig 6



STMicroelect
L7805CV ...

\$0.63
Digi-Key

Note: 78xx family
7812, 7805 etc.

- 3-Terminal Regulators
 - Output Current up to 1.5 A
 - Internal Thermal-Overload Protection
- KC (TO-220) PACKAGE (TOP VIEW)

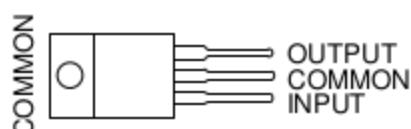
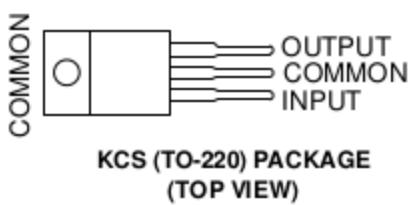


Fig. 9.

Lecture Notes is 2022S-101n
in the class github.

2022S-101-accessible_CMPE240-HarryLi.pdf

2022S-101-Notes-2022-01-26.pdf

μ A7800 SERIES POSITIVE-VOLTAGE REGULATORS

SLVS056J – MAY 1976 – REVISED MAY 2003

(4) Right Angle (plng) Connector



Kobiconn
151-7620E-E DC
\$1.14
Mouser Electronics

(5) Assorted Resistors (A few hundred ohms to a few Mega Ohms).

(6) Compensating Caps.

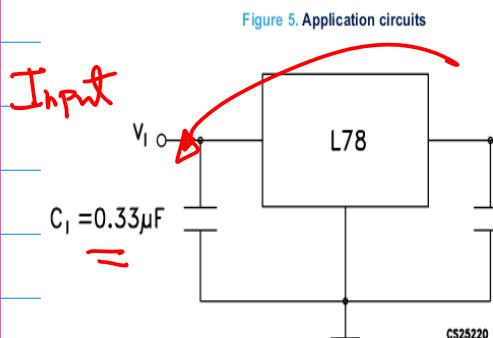


Fig. 7

Components for GPIO

Use GPIO "Hello, the world"
for Debugging | Input { "0"s
| Output { "1"s
} "0"s
} "1"s

Feb and (Wed)

Today's Topics:

1° Bill of Material

2° CPU Datasheet, Architecture

options for the target platform.

(LTC1769)

Note for STmicro
the Caps Required are
Z.

the Caps Required are

Z.

a. NVIDIA (Nvidia) Jetson NANO

Caution: Device Driver Programming
in U.S. Kernel Space.

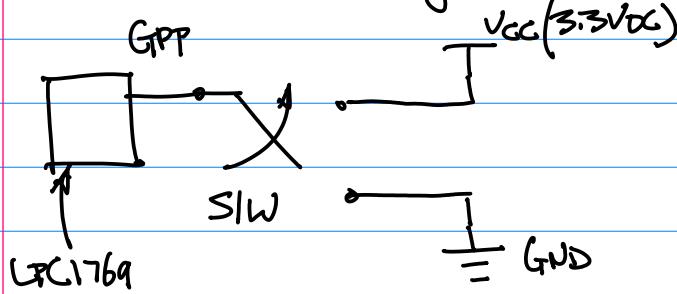
5% Bonus: Implementation at
Registers/Hardware Level, LCD

Fig. 8

LCD has to the Same SPI I/F
for NAB & LPC1769.

GPP I/O Output Testing.

Example: GPIO I/O Testing Circuit



GPP: General Purpose Port,
Same as GPIO.

Fig 1.

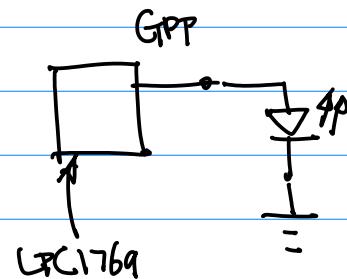
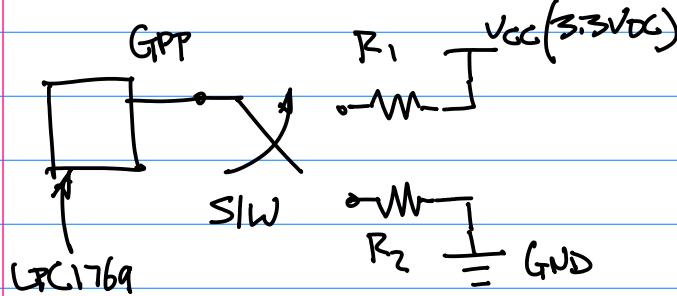
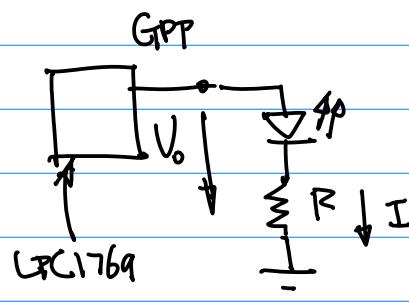


Fig 2



Better Design.

Fig 4.

Consider Resistor Value Calculation

$$V_{CC} = 3.3VDC, V_{out} = 3.3VDC \\ (\text{GPP})$$

Select Resistor Value to Regulate
the Amount of Current $\sim 10mA$

$$\text{Hence, } R_{1,2} = \frac{V_{CC}}{10mA} = \frac{3.3}{10 \times 10^{-3}} \\ = 3.3 \times 10^2 \Omega.$$

Calculation of the Resistor

$$V_o = V_{LED} + IR$$

... (1)

$$V_{LED} \approx 1.2V, I = 10mA, V_o = 3.3V \\ (\text{CMOS})$$

Substitute the

above Conditions into Eqn(1),

$$3.3 = 1.2 + R \times 10 \times 10^{-3}$$

$$R = \frac{2.1}{10^{-3}} = 2.1 \times 10^2 = 210 \Omega$$

Note: please use right size of the
prototype wire
 $26 \sim 30 \text{ AWG}$.



Striveday™ 26 AWG 100ft
amazon.com



Wire - 200m 30AWG Blue

CMPED40 Spring22

6

Note: Optional — Right Angle

RJ-45 Connector: (8 pos)
(Female)



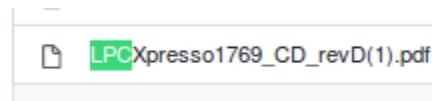
Modular Jack
8P8C PCB ...

\$0.69
PEconnecto...

1. Exercise: Bring your Prototype Board together w/LPC1769 CPU module to the next Class.

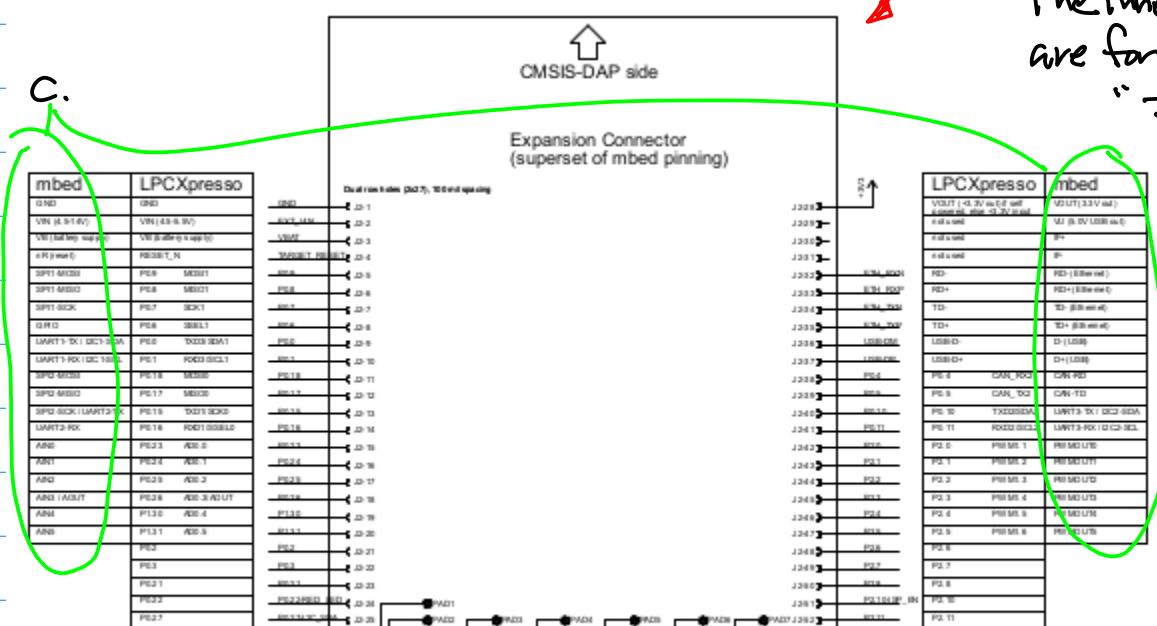
2. Form 4-Person Team, have a Coordinator, And report your team formation By next Class.

Example: LPC1769 CPU module Schematics



a. CPU module and Schematics in b.
c. LPC1768 mbed.

b. The inner tables are for 1769.
"I/O Rich"

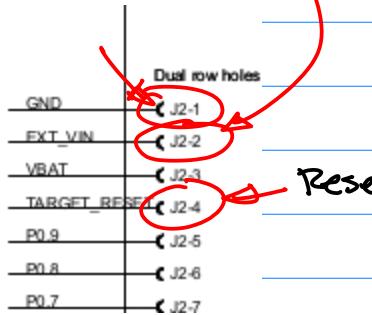


CMP-E240 Spring 22

Exercise: How many GPIO pins?
(for LPC1767 module)

Example: Power Input to LPC1769

LPCXpresso	
GND	
VIN (4.5-5.5V)	
VBAT	
RESET_N	circled
P0.9	MOSI1
P0.8	MISO1
P0.7	SCK1



Note: Enumeration of the Connector
pins → the first pin is
marked as "1".

Note: physical mapping of the pin
to Actual module ;

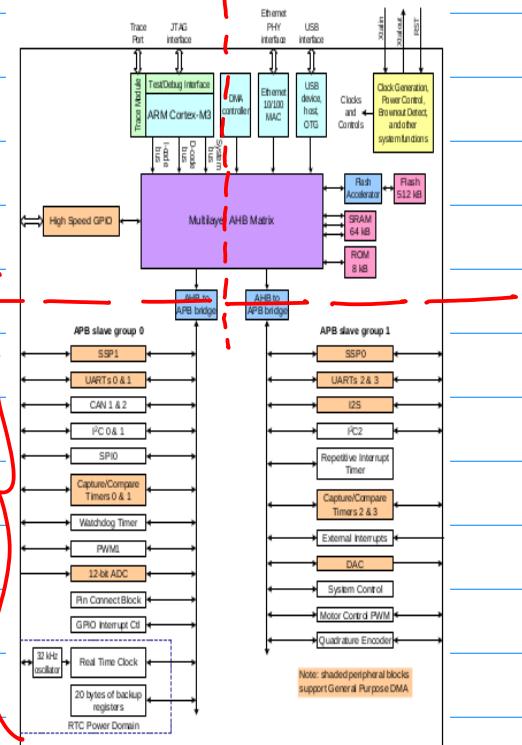
Note: Reset pin — provide
Access to this pin in your
Prototype design.

CPU Datasheet from github / ~lcmpe244

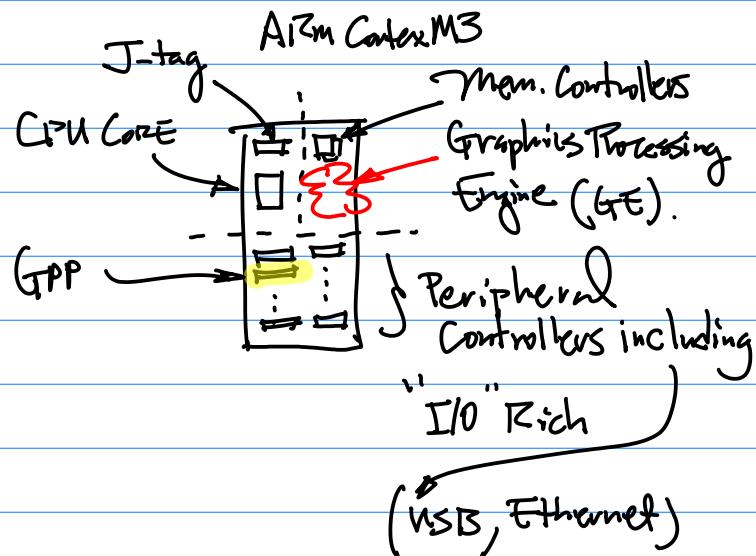
2021F-107-lpc-cpu-UM10360.pdf

Example: Build "Hello, the world" for
the prototype system.

Step 1. CPU Architecture, PP.9 ≠ 12



Peripheral
Controllers



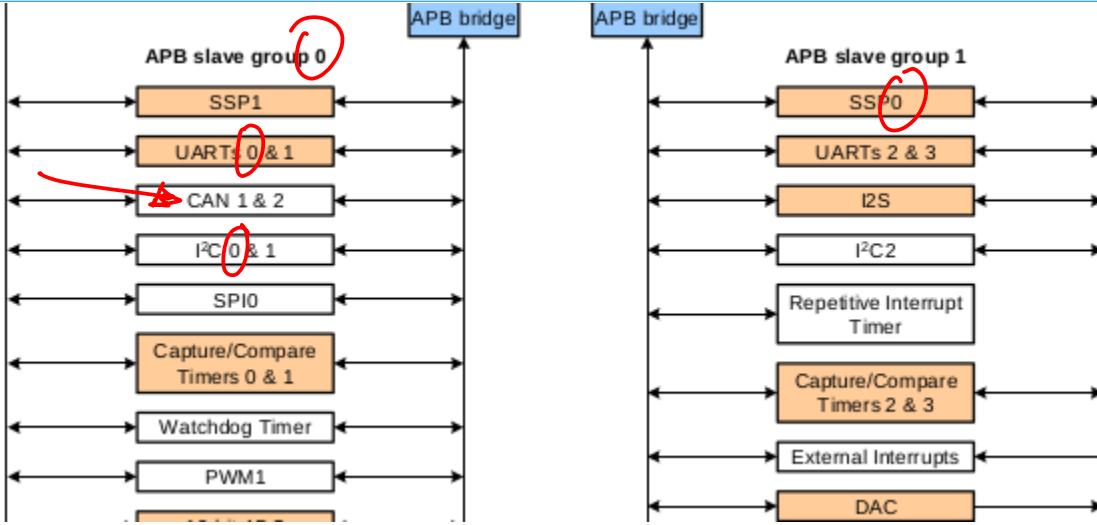
Feb 7 (Monday).

Today's Topics 1° CPU Datasheet
+ Schematics; 2° Hardware

Design for the Prototype System.

Select GPP (General Purpose Port) to Build
GPIO I/O to Realize "Hello, the world"
function. (with Input Testing / Output Testing
Circuit)

Note: Peripheral Controllers

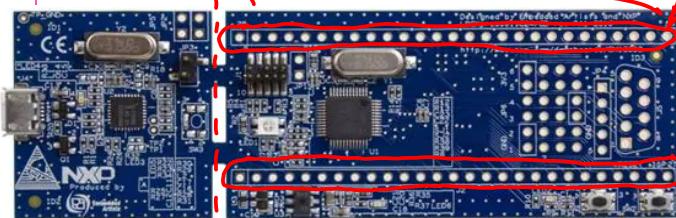


Step 2: Map the GPP to its physical Board, e.g., CPU module.
Hence, we need schematics.

from the Class github.

a. from the schematics,
identify the first
pin, e.g., J2-1

Do the same of your physical
CPU module. J2 connector
CPU module



Interface to the host

LPCXpresso

GND	
VIN (4.5-5.5V)	
VB (battery supply)	
RESET_N	
P0.9	MOSI1
P0.8	MISO1
P0.7	SCK1
P0.6	SSEL1
P0.0	TXD3/SDA1
P0.1	RXD3/SCL1
P0.18	MOSI0
P0.17	MISO0
P0.15	TXD1/SCK0
P0.16	RXD1/SSEL0
P0.23	AD0.0
P0.24	AD0.1
P0.25	AD0.2
P0.26	AD0.3/AOUT
P1.30	AD0.4
P1.31	AD0.5
P0.2	
P0.3	
P0.21	
P0.22	
P0.27	
P0.28	
P2.13	

Dual row holes (2x27), 1:

GND	J2-1
EXT_VIN	J2-2
VBAT	J2-3
TARGET_RESET	J2-4
P0.9	J2-5
P0.8	J2-6
P0.7	J2-7
P0.6	J2-8
P0.0	J2-9
P0.1	J2-10
P0.18	J2-11
P0.17	J2-12
P0.15	J2-13
P0.16	J2-14
P0.23	J2-15
P0.24	J2-16
P0.25	J2-17
P0.26	J2-18
P1.30	J2-19
P1.31	J2-20
P0.2	J2-21
P0.3	J2-22
P0.21	J2-23
P0.22	J2-24
P0.27	J2-25
P0.28	J2-26
P2.13	J2-27

CMPED40 Spring 22

9

Step 3. Select 2 GPIO pins

One for Input Testing Circuit, P0.2 Input, P0.3 Output Table 1.

The other for Output Testing Circuit.

Connectivity & Pin Assignment Table
3 col. format

P0.26	AD0.3/AOUT
P1.30	AD0.4
P1.31	AD0.5
P0.2	
P0.3	
P0.21	
P0.22	
P0.27	
P0.28	
P2.13	

Connector Pin

CPU/Connector Pin	Description	Notes
P0.2/J2-21	GPIO Input	
P0.3/J2-22	GPIO Output	

GPIO pins, CPU Naming.

CPU Datasheet for General Purpose Port

TP117

pin. For other functions, the direction is controlled automatically.

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

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

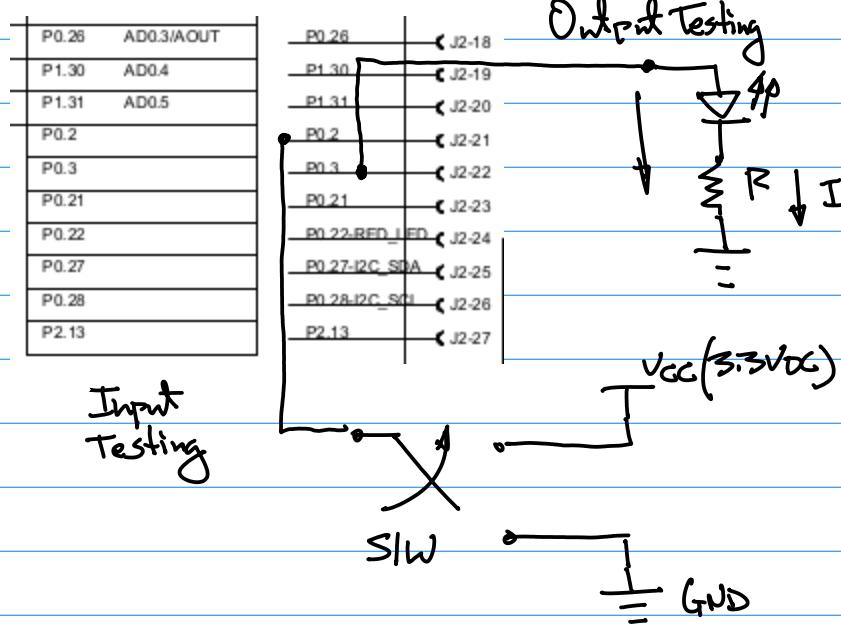
PINSEL0 Special Purpose Register

$$\text{PINSEL0}[5:4] = \begin{cases} 00 & \text{GPIO} \\ 01 & \text{TX} \\ 10 & \text{ADC} \\ 11 & \text{Not used.} \end{cases}$$

2 bits @ Bit 4 & 5

CmPE240 Spring 22

10



PP14.

Framework: The A week from today
(Feb 14).

1° Draw Schematic Design for
GPIO Testing.

2° Take a photo of your prototype
Board. To Capture the work-in-progress

3° Combine 1° & 2° into One Pdf,
then Zip it. Submit your
Work online to SJSU CANVAS.

Consider GPU Architecture Discussion.

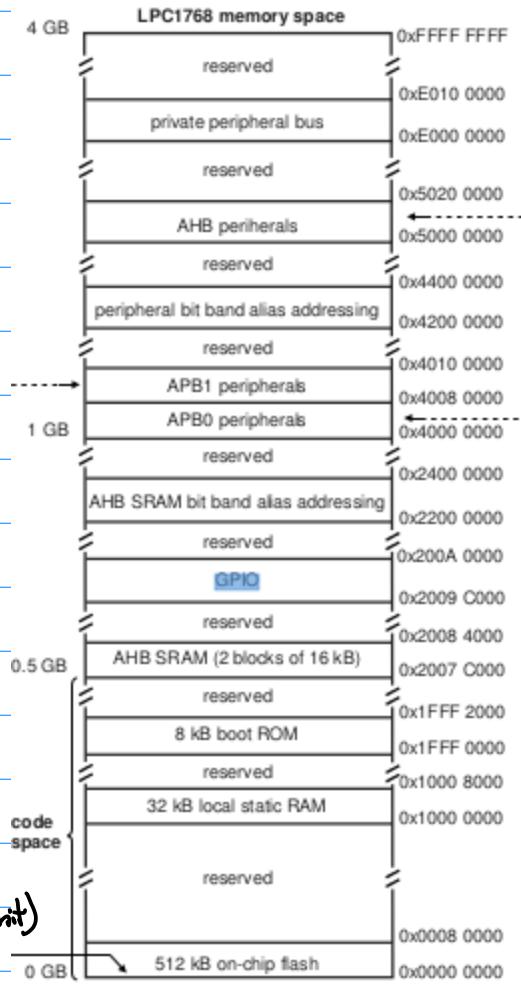
Example: GPU Block Diagram →

Memory Map.

1. 32 Bit Architecture

ALU (Arithmetic Logic Unit)
R.F. (Register files),
Bus System, Data ~, Addr. ~

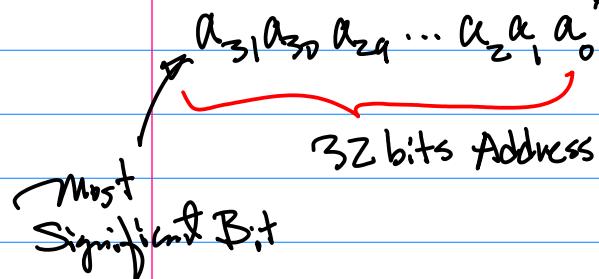
Bank of Registers, 32 bit
32 bit



Cmpe240 Spring 22

11

2. Addr. Bus 32 Bit.



Least Significant bit (Little Endian)

"Little Endian" v.s. "Big Endian"

Step 2

To set up the IDE Development Environment to match your target platform.

Feb 9 (Wed) Today's Topics :

- 1° CPU Architecture ;
- 2° MCU Xpresso, IDE, GPIO Testing Program.

Example: Compilation & Build

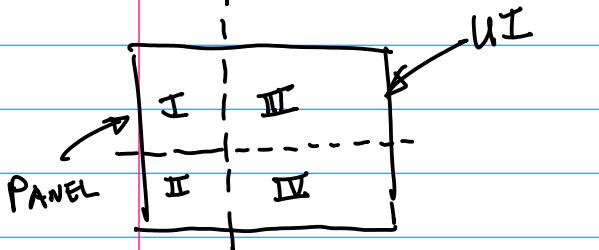
First Firmware Program for GPIO I/O Testing.

Note: Visit NXP website, Sign up to become a Developer.

Download MCU Xpresso.

Version 11.3.0

Start your JDE XPRESSO.



Project Panel (I):

Import LPC1769 patch from Z4D Class github.

1769 patch.zip

on11.10.Guidelines for Devs

Step 3

2018S-11-GPIO-2015-1-30.zip

Import GPIO Project (Sample code) for your Homework.

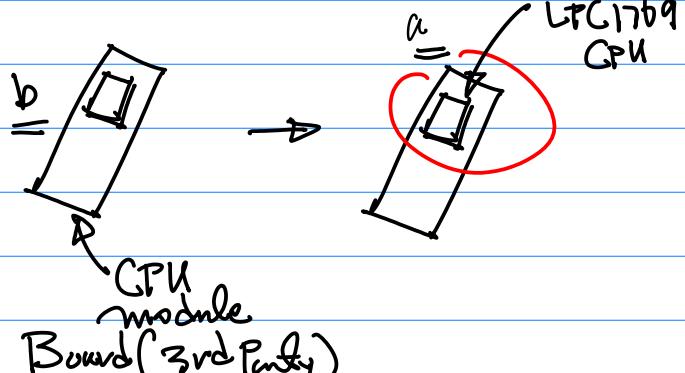
Step 4. Build & Execute GPIO Sample Project

Highlight GPIO Project at the Project Panel → Move to

Quickstart Panel, click on "Build" to Build it.

Step 2

To set up the IDE Development Environment to match your target platform.



Go to "Quickstart" → Panel to Import LPC1769 Patch, And projects

Note: There are several projects, pre-tested for your references purpose, including C/C++, SRI, etc.

Note: If not the first time, then Click on "Clean" to Clean it.

Step5 Flash/Writes to FLASH

Memory, then executing the Program Step by Step (or Execute the entire program).

Note: For Debugging, you will need to make sure the JDE Setting is for "Semi-Host", e.g. your host Laptop will be allowed communication to the Board During Debugging process. However, when doing deployment, choose "host" settings instead in the Board Configuration.

Homework: (Due 1½ weeks from Today)

Feb21, Monday 11:59 pm.

1. Build A Prototype System for GPIO Testing, Interrupt Testing.

2. Provide Schematic Drawing.
Use the tool of your preference.

3. Take a photo showing LED on And Entire System (Prototype + Host with USB Link)

4. Provide modify C code as a Separate file.

5. Put Everything into One pdf Except C code.

6. Zip the Submission, use Naming convention

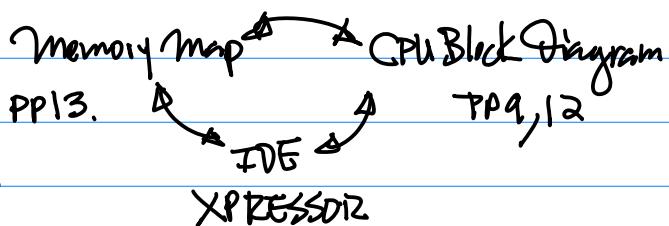
7. First Name - Last Name - SID - GPIO.ZIP
Submit to CANVAS.

Feb14 (Monday)

Today's Topics: 1° CPU Architecture

Base Reference: CPU Datasheet
from the class github/fkualili/cmpe240

Z02/F-17 ~



Examples: Continuation of the Theoretical Discussion

1. CPU Core, Cortex M3.

ARM CPU Belongs RISC.

Reduced Instruction Set Computer

General Design Guidelines

{ Uniformity

Regularity

Orthogonality

2. 32 Bit Architecture

Bus System (Address, Data, Control)
32 bits
R.F. (Register File)
32 bits

T.C.F.
BANK of Registers
32 bits

General Purpose

Registers : Those Registers that can Participate Any Meaningful Arithmetic/Logic Operations

Special Purpose

Registers : With Special Dedicated Functions, such as init/Config of Peripheral Controller.

ALU, 32 bits

(Arithmetic/Logic Unit).

3. Memory Map.

Question : How many individual memory address are there for 32 Bit Architecture?

$$2^{32} = 2^2 \cdot 2^{10} \cdot 2^{10} \cdot 2^{10} = 4 \text{ G}$$

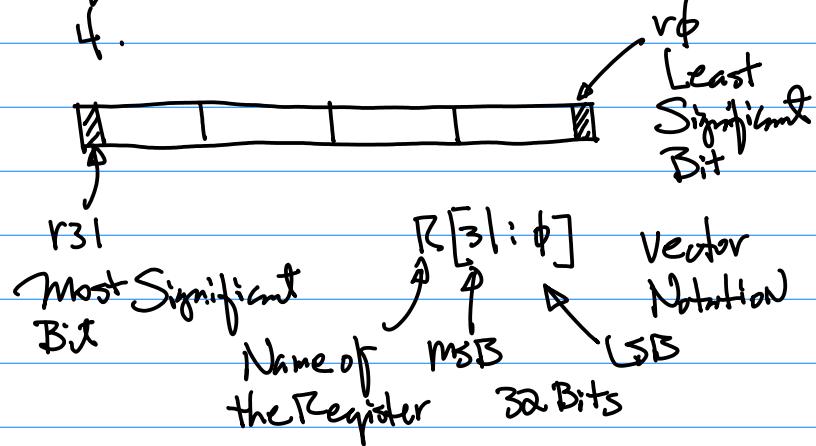
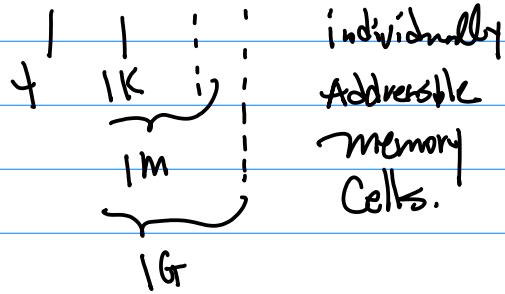


Fig. 1

4. Byte Addressable Machine

Those Machines whose Smallest memory cell with an unique address is a Single Byte.

1. Byte Addressable Machine

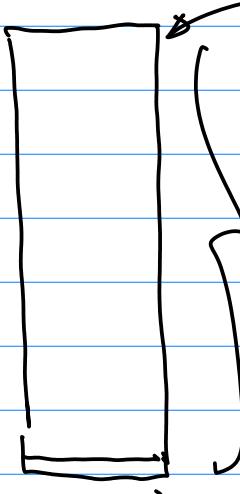
Question : What is the unit for the memory cells mentioned

Above in Question 1? A Byte.

Therefore, 32 bit Architecture \rightarrow 4 G Bytes

0x0000-0000

1st mem. cell



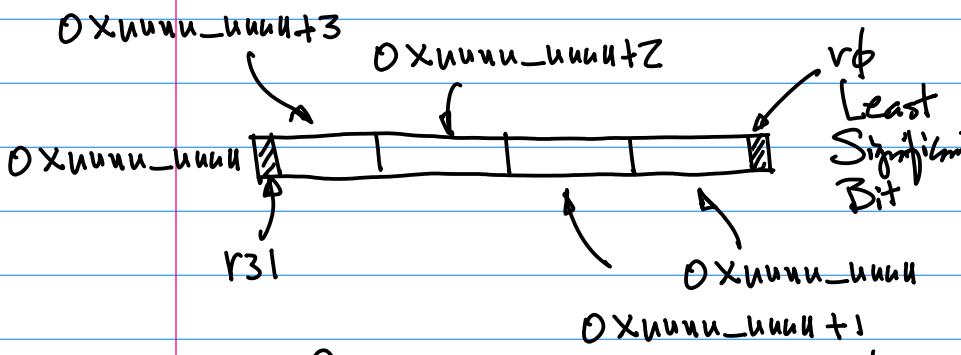
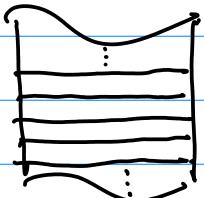
$$2^{32} = 4 \text{ G (Bytes)}$$

Fig. 2

5. Power-up Address : ~ when the CPU is Powered up, it fetches the 1st executable

1st Executable instruction from this memory location.

Now, For A Special purpose Register illustrated Below, it occupies 4 Bytes in A memory Block



A Special Purpose Register covers 4 Address, Suppose the smallest Addr. is x , then, we have

$x+1, x+2, x+3$ 3 Additional Addr. for the Special Purpose Register

b. Divide memory map into 8 Equal Banks.

$$2^{32} / 8 = 2^{32} / 2^3 = 2^{29} = 2^9 \cdot 2^{20} = \\ 512M (\text{Byte}) \quad 1M$$

the size for Each Memory Bank.

Feb 16 (Wed)

Topics: 1° CPU Architecture Discussion
2° Prep. for the Project of Building
3D G.P.U (Graphics Processing)

Example: Continuation of our Previous Discussion.

1. Eight Equal Banks

a. 1st Bank: BANK0
Last Bank: BANK7

b. BANK0 holds the power up Address \rightarrow Boot Process
 \downarrow
Boot Sequence

c. a_3, a_2, a_1 (3 Consecutive Bits to define each Mem. Bank)

Question: which 3 bits from the Addr. Bus Does one need to uniquely define Each Mem. Bank?

3 most Significant Bits $\rightarrow a_3, a_2, a_1$

$a_3, a_2, a_1, \phi, \phi, \phi, \phi, \dots, \phi, \phi, \phi$
 $\underbrace{\qquad\qquad\qquad}_{\text{Don't Cares}}$

d. Defining Starting Addr. for Each Mem. Bank

$a_{31} \ a_{30} \ a_{29} \ | \ a_{28} \ a_{27} \ \dots \ a_1 \ a_0$

$0 \ 0 \ 0 \ | \ 0 \ 0 \ \dots \ 0 \ 0$

$0 \ 0 \ 1 \ | \ 0$

$0 \ 1 \ 0 \ | \ 0$

\vdots

$1 \ 1 \ 1 \ | \ 0 \ 0 \ \dots \ 0 \ 0$

BANK0 (1st)

BANK1 (2nd)

BANK2 (3rd)

Starting Address

0x0000_0000

0x2000_0000

0x4000_0000

BANK7 (8th)

0x8000_0000

Example: Find memory Bank which holds (a) SPI Port
(b) GPP Port?

Sol: (Option 1). From GPU Datasheet,
Memory map.

GPID Block Starting Addr.
0x2009_C000 ;

Option 2: (more in-depth information)

GPP Peripheral controller \rightarrow

Special Purpose Registers \rightarrow Configuration
DR Control Register
 \rightarrow To Config And Init

Addr. of the Register

\leftarrow
GPP Peripheral Controller

FIDIR (Fast I/O Direction defining

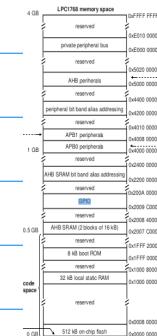
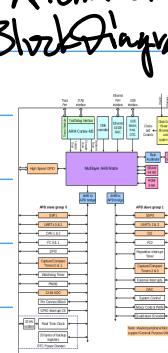
Register, e.g.: Define Input/Output
Direction)

2nd Memory Bank starting Addr: 0x2000_0000

3rd " " " " " : 0x4000_0000

Therefore GPP port is within Bank 1 (2nd Bank)

GPU Architecture Block Diagram \rightarrow memory map



IDE Xpresso

Fig. 1

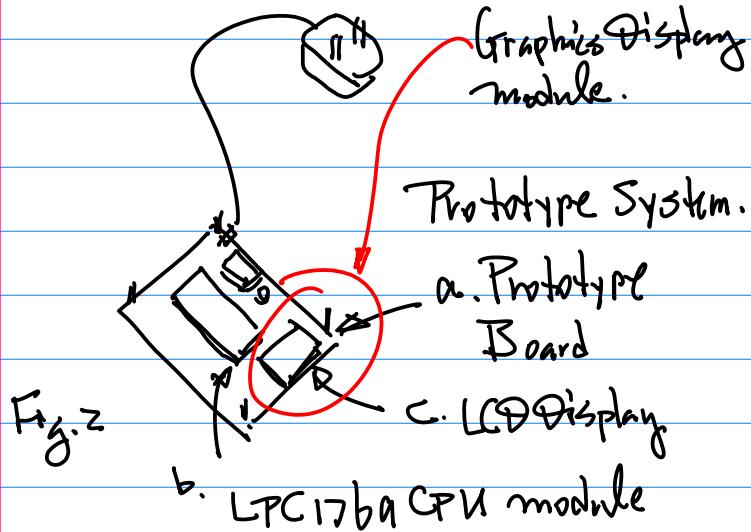
Consider 2D Graphics Engine Design.

Phase I: Design & Realize
Display Function By Hardware-Software Co-Design.

Phase II: 2D Vector Graphics Processing Engine.

Phase I Design / Implement

Graphics Display module



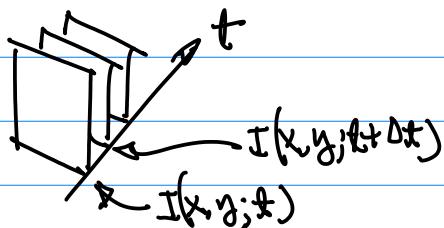
1024x768 Comparable to TZOP.
Color Image 24 bits for Each pixel
"pixel Depth"
bpp (Bits per pixel)
30 FPS (Frames per Second)
To find Bit Rate (Total Bits per Second)
 $1024 \times 768 \times 24 \times 30$

(PP.3)

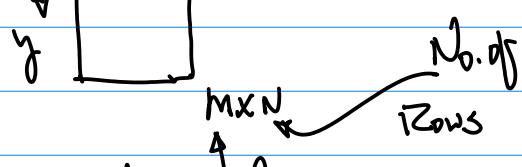
Design Guideline:

1° SPI Interface (Serial Peripheral Interface) to provide Adequate Bit Rate. $50 \text{ mbps} \approx 100 \text{ Mbps}$

Example: Suppose we have to display a video/graphics,
(Pixel graphics)



Resolution: $M \times N$ (0,0) Fig. 3



TZOP, 1080p High Resolution

Feb 21 Monday

Note: 1° Next Lecture, Feb 23rd,
Show & Tell, Inspection of your Prototype Board (One person on Board).
2° Form 4 Person Team, Update the Status.

Example: Continued from the previous Example.

$$1024 \times 768 \times 24 \times 30 = 2^x ?$$

$$1024 \times 768 \times 24 \times 30$$

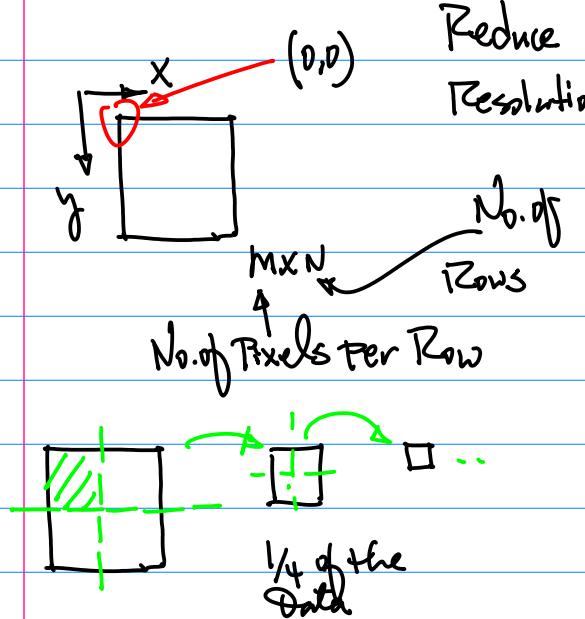
$$\vdots \quad \vdots \quad \vdots \quad \vdots$$

$$2^{10} \quad 2^{10} \quad 2^5 \quad 2^5 =$$

$$2^x = 2^{10} = 1024$$

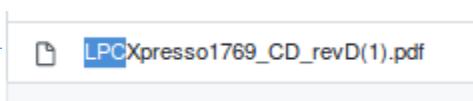
$$2^x = 2^9 = 512 \quad 768$$

$$= 1K \cdot 1K \quad 768 = 1Gbps$$



For Pixel Graphics, with Vector Graphics, we can further Reduce the Amount of Data, So LPC1769 as a platform can be utilized to Build Graphics Processing Engine.

Example: Start SPI I/F Hardware Discussion First.



LPCXpresso	
GND	
VIN (4.5-5.5V)	
VB (battery supply)	
RESET_N	
P0.9	MOSI1
P0.8	MISO1
P0.7	SCK1
P0.6	SSEL1
P0.0	TxD3/SDA1
P0.1	RxD3/SCL1
P0.18	MOSI0
P0.17	MISO0
P0.15	TxD1/SCK0
P0.16	RxD1/SSEL0
P0.23	AD0.0

1. SPI Hardware Interface requires 4 pins, "3+1" pins
Names:
 Output pin MISO Master Output
 Input pin MOSI Slave Input
 Clock pin SCK ~ Input ~ DP
 plus "Enable" pin SSEL_X
 move than ONE SPI.

Table 1. SPI pin Assignment

Name	Pin CPU/Connector	Note
MOSI	P0.9 / J2-5	
MISO	P0.8 / J2-6	
SCK	P0.7 / J2-7	
SSEL	P0.6 / J2-8	

Example: Design SPI I/F to Color LCD module.

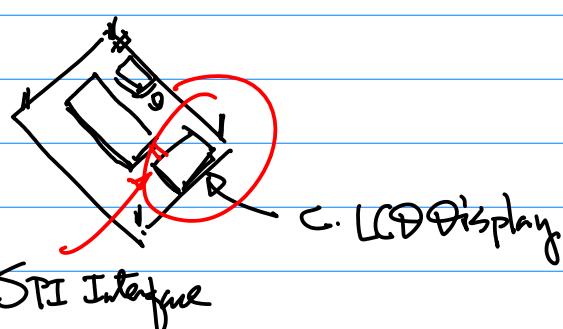


Fig 1.

Identify the Right Suitable LCD module.

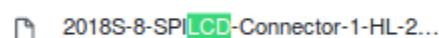
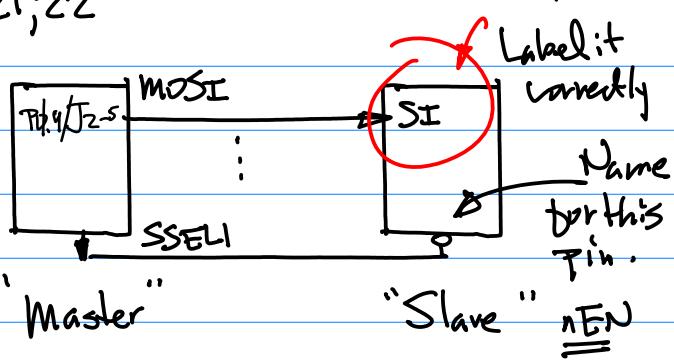
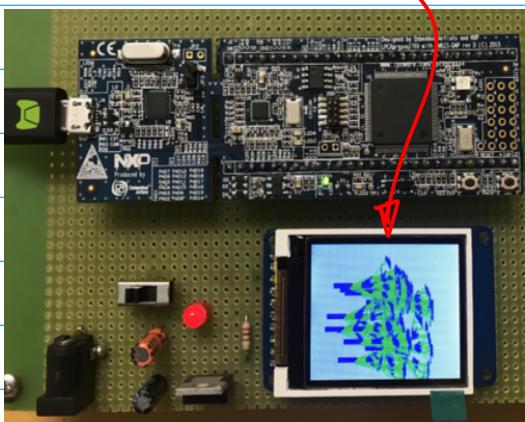


Fig 2

CmPE240

Feb 21, 22

LCD Sample module



Pin Connectivity Table for Schematic Design.

2018S-9-SPILCD-Connector-1-HL-2..

LPC-1769		LCD
Label	Pin	
MOSI	P0.18	SDA
MISO	P0.17	MISO
SCK	P0.15	SCL
CS	P0.16	TFT CS
GPIO-DC (Data / command)	P0.21	AO
GPIO-Reset	P0.22	RST
3.3V		LED+
GND		LED-

Reference: CTI One Corporation

Note :

Bit Rate for 30 FPS, and

18bit Pixel Depth

- LPC 1769

- Color TFT LCD display

Resolution: 128x160

Pixel Depth: 18-bit (262 144) colors

Controller: ST7735

Interface: SPI interface

Note 1. Defining which Device is "master",
which Device is a "Slave".

In our case, CPU (LPC1769) is a master

Name	Pin CPU/Connector	Note
SI	MOSI P0.18 / J2-5	SI → P0.18 / J2-5
	MISO P0.17 / J2-6	SI → P0.17 / J2-6
	SCK P0.15 / J2-7	SI → P0.15 / J2-7
	SSEL P0.21 / J2-8	SI → P0.21 / J2-8

Annotations around the table:

- A large curved arrow labeled "Serial Input for LCD" points from the "SI" row to the "P0.18 / J2-5" column.
- A callout bubble points to the "P0.18 / J2-5" entry with the text "provide Pin Number of the LCD module".
- A callout bubble points to the "P0.18 / J2-5" entry with the text "Pin Number of the CPU module".
- A green circle highlights the "SI" label in the first row.
- A green circle highlights the "P0.18" label in the first row.
- A green circle highlights the "J2-5" label in the first row.

Interface Signals on LCD (General Guidelines)

SPI Signals
 Control Backlight
 Toggle Data/Command

Name	Pin CPU/Connector	LCD Pin Label	LCD Pin No. (Connector)
MOSI	P0.9 / J2-5		
MISO	P0.8 / J2-6		
SCK	P0.7 / J2-7		
SSEL	P0.6 / J2-8		

Feb 23rd (Wed)

Today's Topics : 1° SPI LCD
Hardware Interface ; 2°

2D Graphics Engine

Example: LCD Interface

SPI Signals — Signal from the master
 Control Backlight — GPIO from the CPU
 Toggle Data/Command — GPIO from the CPU

Note : To prepare for 2D Graphics Engine Design, shown Below

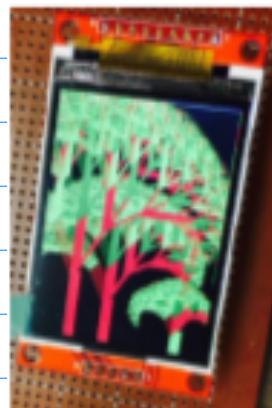


Fig.1.

Exercise : Build Connectivity Table
for the CPU & SPI LCD
Interface.

- By Next Monday, Feb. 28.
- One col. to cover all CPU pins for SPI, for Backlight, for Command/Data, Enable Signal (to enable the LCD Display module).
- One col. to match each CPU pin, to define the pins connected to the CPU module.

2D Vector Graphics : Screen Saver to generate trees ; And



Fig.2



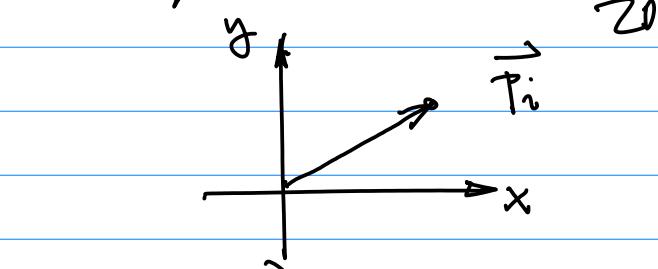
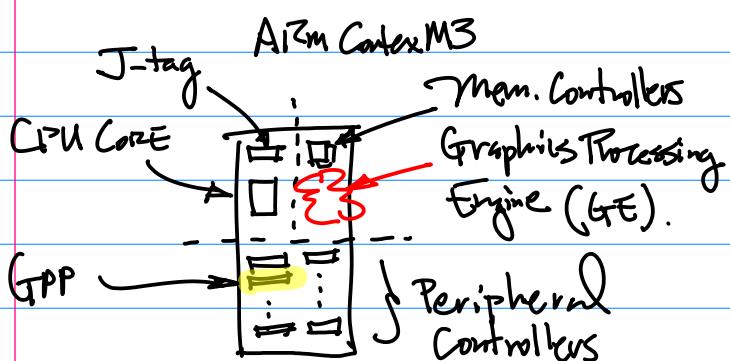
Fig.3

Cmpe240

Feb 23rd, 22

Note: Please have SPI LCD
 Interfacing Design Implementation
 (Hardware) Ready By Next Monday
 Feb 28th.

Example: 2D Graphics Engines:



2D Vector \vec{P}_i on X-Y Coordinate System
 as point, to draw this vector as
 pointed line (Direction, & Magnitude)
 A line from the origin to the point.

Notations

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

Vector Representation of A Line

Graphics Engine

- 1. Display Driver
- 2. 2D GE
- 3. 3D GE
- 4. Video Codec

2D GE. Vector Graphics

- Primitives, 1. :
- 2. 2D Transforms
- 3. D.D.A.

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

where λ is scalar

... (1)

Feb 28 (Monday)

Theoretical Background On 2D Vector Graphics.

Note:
 Vector Graphics: Graphics By Vectors
 Pixel Graphics: Images Videos

Define A 2D Vector

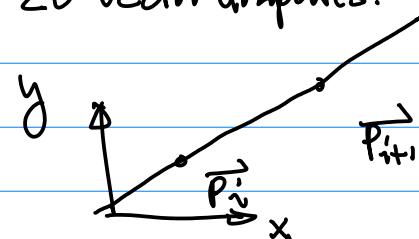


Fig 1.

\vec{P}_i Starting Point; \vec{P}_{int} Ending Point

First, Continuation of Prototype Board Demo/Inspection.

2D Vector Graphics.

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

Direction Vector.

Starting point, A fixed point

Note: 1. Define A Directional Vector

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

Example: Suppose we have

$$\vec{P}_i(x_i, y_i) = (2, 4), \vec{P}_{i+1}(x_{i+1}, y_{i+1}) =$$

(5, 10). Find A Directional vector $\vec{J}(x, y)$?

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

From the given condition, we have

$$\begin{aligned} (x_{i+1} - x_i, y_{i+1} - y_i) &= (5 - 2, 10 - 4) \\ &= (3, 6) \end{aligned}$$

Note 2. Eqn(1),

a fixed point (starting pt) + λ : directional vector

Scalar, Scaling

$$-\infty < \lambda < +\infty$$

Question 1: In order to find \vec{P}, \vec{P}_i from Eqn(1), what is $\lambda = ?$

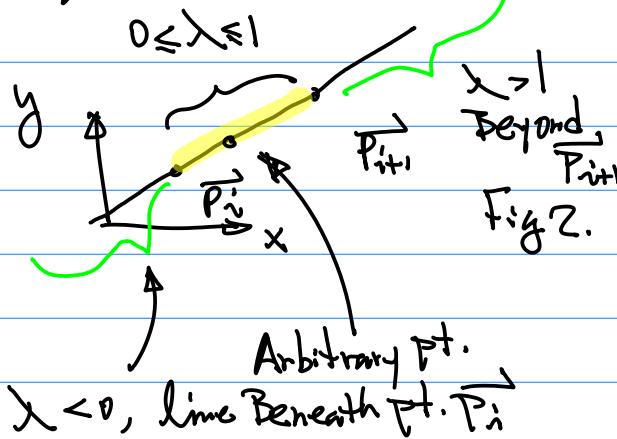
$$\lambda = 0, \vec{P} = \vec{P}_i$$

Question 2: To find \vec{P}_{i+1} from Eqn(1), what is $\lambda = ?$

$$\lambda = 1, \vec{P} = \vec{P}_{i+1}$$

$$\begin{aligned} \text{Verify: } \vec{P}(x, y) &= \vec{P}_i + \lambda (\vec{P}_{i+1} - \vec{P}_i) \\ &= \vec{P}_i + 1 \cdot (\vec{P}_{i+1} - \vec{P}_i) = \vec{P}_i + \vec{P}_{i+1} - \vec{P}_i \\ &= \vec{P}_{i+1} \end{aligned}$$

Question 3: To find Any arbitrary Point from Eqn(1) Between \vec{P}_i and \vec{P}_{i+1} , what is $\lambda = ?$



Conclusion:

Conclusion: For Eqn(1), when

$$\lambda=0, \vec{P} = \vec{P}_i \text{ Starting pt}$$

$$\lambda=1, \vec{P} = \vec{P}_{i+1} \text{ Ending pt}$$

$$0 < \lambda < 1, \text{ Any pts Between}$$

$$\vec{P}_i \text{ & } \vec{P}_{i+1}$$

$$\lambda < 0 \text{ Beneath } \vec{P}_i$$

$$\lambda > 1 \text{ Beyond } \vec{P}_{i+1}$$

Design A Screen Saver, Rotating Squares



Fig.4

Example: Design A Rotating Set of Squares By Eqn(1).

Step1. Define A set of 4 Points of a Square.

$$\vec{P}_0(x_0, y_0) = (25, 5)$$

$$\vec{P}_1(x_1, y_1) = (25, 25)$$

$$\vec{P}_2(x_2, y_2) = (5, 25), \vec{P}_3(x_3, y_3) = (5, 5)$$

$$\{\vec{P}_i(x_i, y_i) | i=0, 1, \dots, 3\} \dots (z)$$

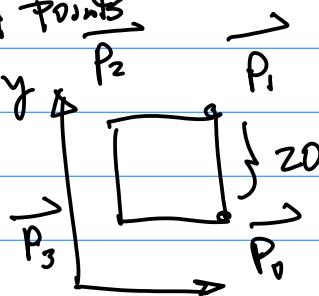


Fig.5

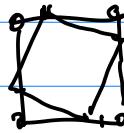


Fig.6

Marchand (wed)

Project I Announcement: Due March 19 Saturday 11:59 pm DN CANVAS.

Requirements

1. Design & Implement 2D Vector Graphics Engine By implementing Screen Savers a, Rotating Squares, b Trees. as shown below.



a.



b.

Fig.5

2. Implementation has to be individual with his/her own Board; No Code/Program is allowed to share. Work/Implement independently.

3. Submission: (1) Exported project as zip file; (2) provide photos of Screen Capture for Both a & b;

(3) 5 Second Video Clip for tree creation, e.g. Display of the tree(s).

4. Written Requirements in addition to this announcement will be Posted on Line (Both on github, and on CANVAS).

5. Submission of the Project:

- ① Exported Project with Source code;
- ② Photos Required from 1-3;
- ③ Video Clip (5 seconds)

Zip together, Submit it to SJSU CANVAS.

Note: Prototype Board Demo and Inspection in Class (Online Zoom Class) is mandatory, will be continued Next Monday. Bring your Board to the class.

Example: Continued from Rotating Step 2. Square Design.

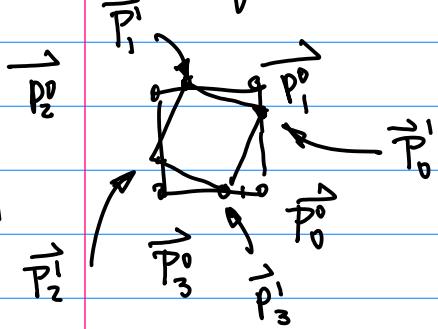


Fig. 1

If we continue this process with

a new set of Vertices $\{\vec{P}_0^1, \vec{P}_1^1, \dots, \vec{P}_3^1\}$ to form a rotated Square as in Fig 1. we can form a new rotated Square just like we did for level 1.

A New set of Vertices:

$$\{\vec{P}_0^2, \vec{P}_1^2, \vec{P}_2^2, \vec{P}_3^2\}$$

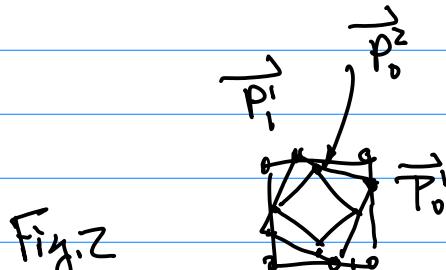


Fig. 2

Counter clockwise

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

Question: What is $\lambda = ?$ For the rotation in Fig 2
 $\lambda = 0.8$

Question: To generate opposite rotation direction, what is $\lambda = ?$ $\lambda = 0.2$

Generally, the formula for the above

Rotating Squares:

From Eqn (1),

$$\text{e.g. } \vec{P}_i^{i+1} = \vec{P}_i^i + \lambda(\vec{P}_{i+1}^i - \vec{P}_i^i) \quad \dots (2)$$

$$\vec{P}_i^{i+1}(x_i^{i+1}, y_i^{i+1}) = \vec{P}_i^i(x_i^i, y_i^i) + \lambda(\vec{P}_{i+1}^i(x_i^i, y_i^i) - \vec{P}_i^i(x_i^i, y_i^i)) \quad \dots (2b)$$

Consider Hard Calculation &

C/C++ Implementation.

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



Or

$$\begin{aligned} (x_i^{j+1}, y_i^{j+1}) &= (x_i^j, y_i^j) + \lambda ((x_{i+1}^j, y_{i+1}^j) - (x_i^j, y_i^j)) \\ &= (x_i^j, y_i^j) + \lambda (x_{i+1}^j - x_i^j, y_{i+1}^j - y_i^j) \end{aligned}$$

... (3)

OR Equation for X Component,

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

Similarly,

$$y_i^{j+1} = y_i^j + \lambda (y_{i+1}^j - y_i^j)$$

Therefore,

$$\begin{cases} x_i^{j+1} = x_i^j + \lambda (x_{i+1}^j - x_i^j) & \dots (3a) \\ y_i^{j+1} = y_i^j + \lambda (y_{i+1}^j - y_i^j) & \dots (3b) \end{cases}$$

In C-Code for Eqn(3a)

$$x[j+1][i] = x[j][i] + \text{lambda} * (x[j][i+1] - x[j][i]);$$

$$y[j+1][i] = y[j][i] + \text{lambda} * (y[j][i+1] - y[j][i]);$$

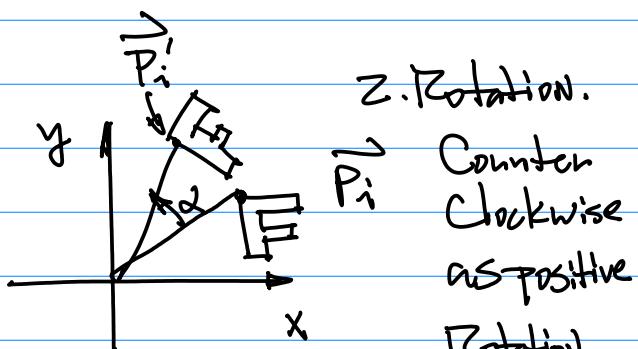
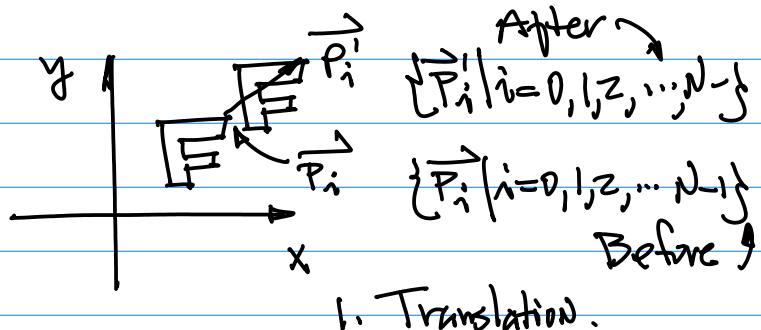
Note: Rotation is defined w.r.t.
the origin of the x-y 2D Coordinate
System



Background Blue Background Brown A tree

Blue Brown

Math. Formulation, 2D Transforms



Z. Rotation.
Pi Counter
Clockwise
as positive
Rotation
 $\alpha > 0$

Example: Develop A matrix formula to define translation.

Goal: To Establish Math. Relationship

Between 2 Sets of Vectors,
e.g.

Before $\{\vec{P}_i | i=0,1,2,\dots N-1\}$

And

After $\{\vec{P}'_i | i=0,1,2,\dots N-1\}$

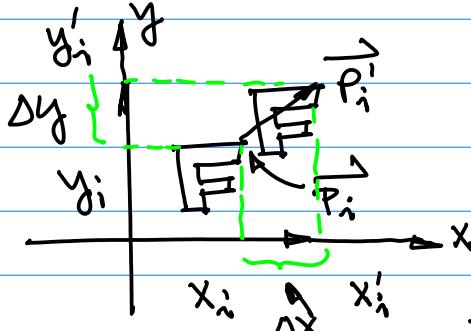


Fig. 1

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

$$x'_i = ? \quad x_i + \text{Displacement}(\Delta x)$$

(After) (Before) ... (1)

$-\infty < \Delta x < +\infty$, hence

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

For y, we have a similar relationship

$$y'_i = y_i + \text{displacement}(\Delta y)$$

(After) (Before)

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

$$-\infty < \Delta y < +\infty$$

\vec{P}'_i After

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

Dummy Dimension

Now, Consider Rotation.

2D Transforms. — Translations

Before $\{\vec{P}_i | i=0,1,2,\dots N-1\}$ And

After $\{\vec{P}'_i | i=0,1,2,\dots N-1\}$

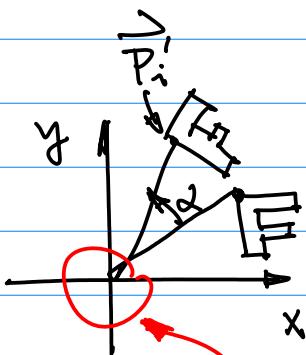
$$\begin{pmatrix} x'_i \\ y'_i \\ 1 \end{pmatrix} = \begin{pmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} \quad \dots (4)$$

CMPE240 March 7, 22

26

Ref:

- 2018F-111-lec-2D-vectors-v3-2018-9-19.pdf
- 2018F-112-lec4-2D-vectors-v3-2018-10-1.pdf



Note:

1° Angle α defined with Reference to the origin $(0,0)$ of the $x-y$ Coordinate System;

2° $\alpha > 0$ for Counter Clockwise Rotation.

$\alpha < 0$ for Clockwise Rotation.

Fig.2

Given Two Vectors, \vec{P}_i , and \vec{P}_{i+1}

1. Find \vec{P}'_{i+1} By Translation
2. Find \vec{P}''_{i+1} After Rotating \vec{P}'_{i+1} By

$\alpha = 30^\circ$ Counter Clockwise Direction.

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

Which Allows us to find \vec{P}'_{i+1} in Fig.3a

Consider Rotation to find \vec{P}''_{i+1} in Fig.3b.

Question: Can Eqn(4) be Applied directly to find \vec{P}''_{i+1} ?

$$\begin{pmatrix} x''_{i+1} \\ y''_{i+1} \\ 1 \end{pmatrix} ? \begin{pmatrix} \cos \alpha & -\sin \alpha & 0 \\ \sin \alpha & \cos \alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x'_{i+1} \\ y'_{i+1} \\ 1 \end{pmatrix}$$

Consider Design An Algorithm Based
2D Vector Graphics & 2D Transforms.

Example: Find Rotation Point $\vec{P}'_{i+1}(x'_i, y'_i)$
for the following operation.

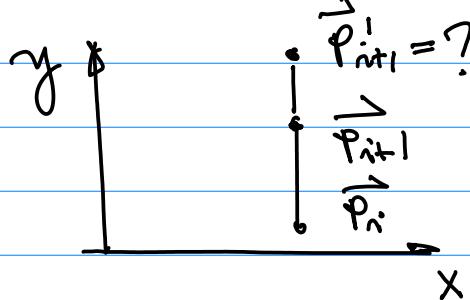


Fig.3a

Angle α w.r.t $(0,0)$
We can not directly use the
above matrix for Rotation in

Fig.3b.

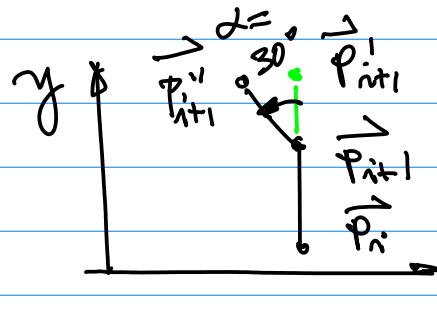


Fig.3b.

Question: Since Rotation Matrix in Eqn(4) can not be directly applied to the solution for Fig.3b, find a way to find a solution for this rotation?

March 9 (Wed)

Midterm March 13rd
(Wed) on-line Room.

Prep. for the midterm.

1. Midterm Exercise on Monday March 14.

Bring 2 Printer Papers to the Class. for Midterm Exercise
Put your Name: ^aFirst-Last-Name
to the top right corner

- b. 4 Digit SID
- c. CmpE240 Mid

2. Have Video ON During Entire Exercise Session.

Video is Required During the Exam.

3. Prototype System Ready to Run. MCUXpresso Ready

4. No text messaging, E-mail are not allowed.

5. Have Screen Capture

Software tool is ready. to
Capture the execution of
your program;

6. Smartphone to take photos of your prototype system.

7. Use online tool to convert photos to pdf.

8. Merge all pdf files in the order into one pdf document.

9. Naming of the pdf:

FirstName-LastName-SID-
CmpE240-mid.pdf
↓ Zip it.

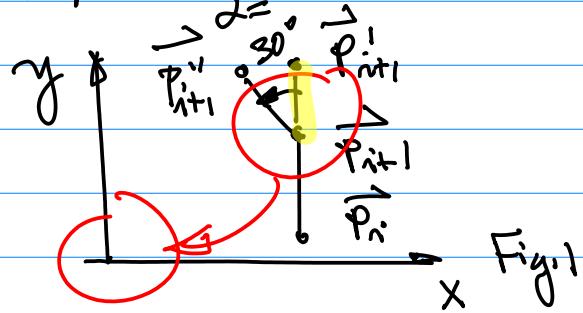
10. Submit the zipped file to SJSU Canvas

Note: Images/photos are in the range of 1~2MB. (~4MB)

Ref:

2018S-10-LCD-DrawLine.zip

Example: Continued.



Step 1. Preprocessing.

move the pattern in such a way
So \vec{P}_{i+1} to overlap with
the origin (0,0) By translation

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

to Be Rewritten
as

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

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

$$T' = \begin{pmatrix} 1 & 0 & -\Delta x \\ 0 & 1 & -\Delta y \\ 0 & 0 & 1 \end{pmatrix} \dots (2)$$

All together, we have
(Composition of 2D Transformations)

$$T_\Sigma = T' T T$$

$$R = \begin{pmatrix} \cos \alpha & -\sin \alpha & 0 \\ \sin \alpha & \cos \alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \dots (3)$$

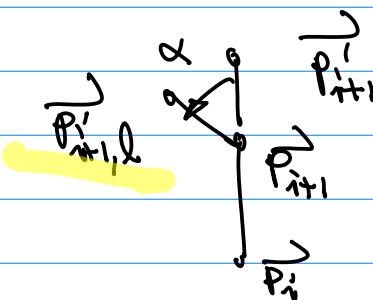
where $\alpha = 30^\circ$

$$\begin{pmatrix} x'_{i+1,l} \\ y'_{i+1,l} \\ 1 \end{pmatrix} = T_\Sigma \begin{pmatrix} x'_{i+1} \\ y'_{i+1} \\ 1 \end{pmatrix} \dots (4)$$

Rotate \vec{P}'_{i+1} after translation.

$$T_\Sigma =$$

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



Evaluate the Above Matrices,
By multiplying them Out, you
will have One equation for x , and
One equation for y .

Step 3 Post-Processing.
(Under Pre-processing)

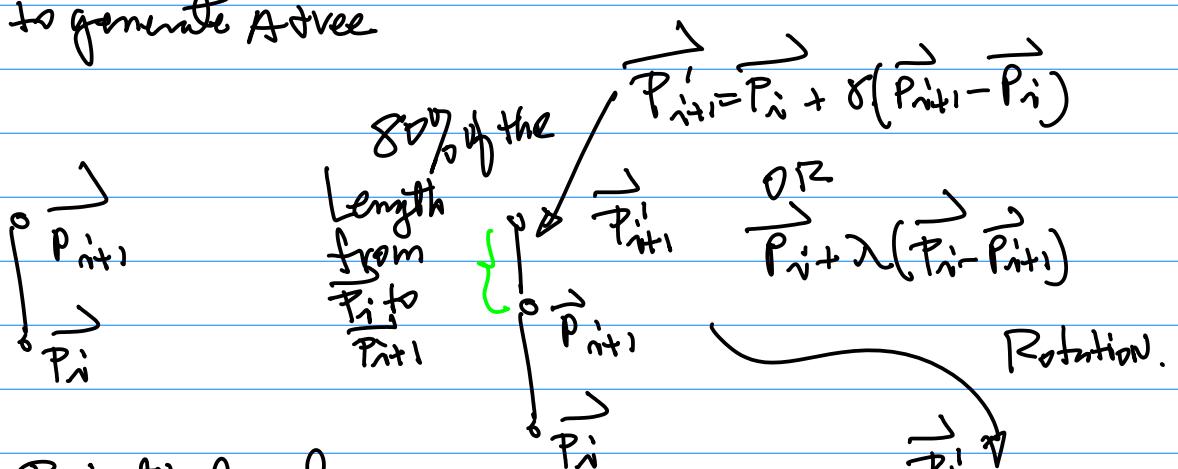
$$\begin{aligned}
 & \left(\begin{array}{ccc} 1 & 0 & -\Delta X \\ 0 & 1 & -\Delta Y \\ 0 & 0 & 1 \end{array} \right) \left(\begin{array}{ccc} \cos \alpha & -\sin \alpha & 0 \\ \sin \alpha & \cos \alpha & 0 \\ 0 & 0 & 1 \end{array} \right) \left(\begin{array}{ccc} 1 & 0 & \Delta X \\ 0 & 1 & \Delta Y \\ 0 & 0 & 1 \end{array} \right) \\
 & = \left(\begin{array}{ccc} 1 & 0 & -\Delta X \\ 0 & 1 & -\Delta Y \\ 0 & 0 & 1 \end{array} \right) \left(\begin{array}{ccc} \cos \alpha & -\sin \alpha & \cos \alpha \Delta X - \sin \alpha \Delta Y \\ \sin \alpha & \cos \alpha & \sin \alpha \Delta X + \cos \alpha \Delta Y \\ 0 & 0 & 1 \end{array} \right) \\
 & = \left(\begin{array}{ccc} \cos \alpha & -\sin \alpha & \cos \alpha \Delta X - \sin \alpha \Delta Y - \Delta X \\ \sin \alpha & \cos \alpha & \sin \alpha \Delta X + \cos \alpha \Delta Y - \Delta Y \\ 0 & 0 & 1 \end{array} \right)
 \end{aligned}$$

Therefore

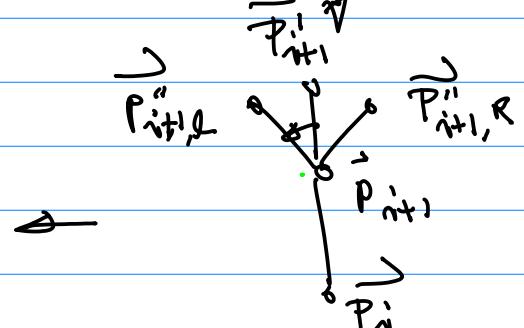
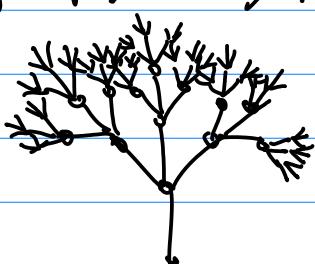
$$x'_{i+1,l} = \cos \alpha x'_{i+1} - \sin \alpha y'_{i+1} + \cos \alpha \Delta X - \sin \alpha \Delta Y - \Delta X$$

$$y'_{i+1,l} = \sin \alpha x'_{i+1} + \cos \alpha y'_{i+1} + \sin \alpha \Delta X + \cos \alpha \Delta Y - \Delta Y$$

The steps to generate A tree



Note: In Project 1, level ≥ 7



CmPE420 March 9 (wed), 22

30

Consider CPU Architecture.

Datasheet: SSP from the github

244

2021F-107-[pc-cpu-UM10360.pdf](#)

2021F-107-hwch-#1 DCY Version 1.2A0

Prefix + Con + Postscript

Power
Peripheral Control Register

17.1 Basic configuration

The SPI is configured using the following registers:

1. Power: In the PCONP register ([Table 46](#)), set bit PCSPI.
Remark: On reset, the SPI is enabled (PCSPI = 1).
2. Clock: In the PCLKSEL0 register ([Table 40](#)), set bit PCLK_SPI. In master mode, the clock must be an even number greater than or equal to 8 (see [Section 17.7.4](#)).

Note:

Special Purpose Registers →

Special functions, most commonly used important SPRs include Configuration/Control Registers(s)

Which defines / dictates the Behavior of Peripheral controller.

NXP Semiconductors

$$\& CR\phi = 0x4008_8000$$

Note: Address in a memory

Bank, find it
0x4000_0000 → Bank
3rd

d. Notation

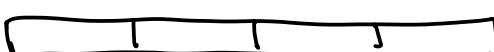
CR ϕ [3:0]

a. SSP: A Super set of SPI

UM10360

Chapter 18: LPC176x/8x SSP0/1

18.6 Register description



c. Addr.

The register addresses of the SSP controllers addresses are shown in [Table 370](#).

Table 370. SSP Register Map

Generic Name	Description	Access	Reset Value	SSPn Register Name & Address
CR0	Control Register 0. Selects the serial clock rate, bus type, and data size.	R/W	0	SSP0CR0 - 0x4008_8000 SSP1CR0 - 0x4003_0000
CR1	Control Register 1. Selects master/slave and other modes.	R/W	0	SSP0CR1 - 0x4008_8004 SSP1CR1 - 0x4003_0004
DR	Data Register. Writes fill the transmit FIFO, and reads empty the receive FIFO.	R/W	0	SSP0DR - 0x4008_8008 SSP1DR - 0x4003_0008
SR	Status Register	RO		SSP0SR - 0x4008_800C SSP1SR - 0x4003_000C
CPSR	Clock Prescale Register	R/W	0	SSP0CPSR - 0x4008_8010 SSP1CPSR - 0x4003_0010
IMSC	Interrupt Mask Set and Clear Register	R/W	0	SSP0IMSC - 0x4008_8014 SSP1IMSC - 0x4003_0014

b. Control Registers

c. Control Registers

e. CLK, Bus, Data size

f. Master/Slave

CmPE240 March 9 (Wed), 22

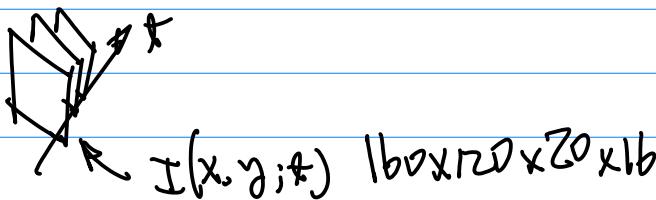
→ 1

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

Bit	Symbol	Value	Description
3:0	DSS	Data Size Select. This field controls the number of transferred in each frame. Values 0000-0010 are not recommended and should not be used.	
0011		4-bit transfer	

Objective : Define
 ① Clock Rate,
 ② Datysize, and SPI I/F
 ③

Tech Spec. → Init & Config.
for CR0.



Find Bit Rate, Define CR0 for
init & config.

March 11. (Monday)

Example: SPI CR0 for SPI I/F b. Init

Table 40. Peripheral Clock Selection register 0 (PCLKSEL0 - address 0x4003 0004) bit description

Bit	Symbol	Description
1:0	PCLK_WDT	Peripheral clock selection for WDT.
3:2	PCLK_TIMER0	Peripheral clock selection for TIMER0.
5:4	PCLK_TIMER1	Peripheral clock selection for TIMER1.
7:6	PCLK_UART0	Peripheral clock selection for UART0.
9:8	PCLK_UART1	Peripheral clock selection for UART1.
11:10	-	Reserved.
13:12	PCLK_PWM1	Peripheral clock selection for PWM1.
15:14	PCLK_I2C0	Peripheral clock selection for I2C0.
17:16	PCLK_SPI	Peripheral clock selection for SPI.
19:18	-	Reserved.
21:20	PCLK_SSP1	Peripheral clock selection for SSP1.
21:22	PCLK_DAC	Peripheral clock selection for DAC.
21:24	PCLK_ADC	Peripheral clock selection for ADC.

Ref: from github

2021F-107-lpc-cpu-UM10360.pdf

a. Project Structure

b. Init Function (ssp.c)

```

212 /**
213 ** Descriptions:
214 ** parameters:
215 ** Returned value:
216 */
217 ****
218 ****
219 void SSP1Init( void )
220 {
221     uint8_t i, Dummy=Dummy;
222
223     /* Enable AHB clock to the SSP1. */
224     LPC_SC->PCLKSEL0 |= (0x1<<10);
225
226     /* Further divider is needed on SSP1 clock. Using default divided by 8 */
227     /* P0.6-0.9 as SSP1 */
228     /* P0.6-0.9 as SSP1 */
229     /* Convert this Hex to Binary
230     /* 0001<<10 shift, CPU Bitsht
231     /* Negation → 0 for these Bits
232     /* #if !USE_CS
233     /* 0011<<20
234     /* 10(B)
235     /* 0
236     /* Set DSS data to 8-bit, Frame format SPI, CPOL = 0, CPHA = 0, and SCR is 15 */
237

```

c. Naming Convention

LPC_SC → SPI /

d. Project Explorer

e. Source Code

f. $10(B) \rightarrow 0x2$, Bit location in PINSEL0

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

PINSEL0	Pin name	Function when 00	Function when 01	Function when 10	Function when 11	Res value
1:0	P0.0	GPIO Port 0.0	RD1	TXD3	SDA1	00
3:2	P0.1	GPIO Port 0.1	TD1	RXD3	SCL1	00
5:4	P0.2	GPIO Port 0.2	TXD0	AD0.7	Reserved	00
7:6	P0.3	GPIO Port 0.3	RXD0	AD0.6	Reserved	00
9:8	P0.4	GPIO Port 0.4	I2SRX_CLK	RD2	CAP2.0	00
11:10	P0.5	GPIO Port 0.5	I2SRX_WS	TD2	CAP2.1	00
13:12	P0.6	GPIO Port 0.6	I2SRX_SDA	SSEL1	MAT2.0	00
15:14	P0.7	GPIO Port 0.7	I2STX_CLK	SCK1	MAT2.1	00
17:16	P0.8	GPIO Port 0.8	I2STX_WS	MISO1	MAT2.2	00
19:18	P0.9	GPIO Port 0.9	I2STX_SDA	MOSI1	MAT2.3	00
21:20	P0.10	GPIO Port 0.10	TXD2	SDA2	MAT3.0	00
23:22	P0.11	GPIO Port 0.11	RXD2	SCL2	MAT3.1	00
29:24	-	Reserved	Reserved	Reserved	Reserved	0

$$0x11011 = 0x707$$

00001011100010111
1 b bits.

$$CR0[15:0] = 0x707 \\ (= 0x0707)$$

Example: Based on the code find tech. Spec.?

Note: 1. CR0 Control/Configuration Register
 $LPC_SSP1 \rightarrow CR0$

Family P-Controller , Addr. of $LPC_SSP1 \rightarrow CR0$: 17xx.h

```

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 /* SSPCSR clock prescale register, master mode, minimum divisor is 0x02 */
241 LPC_SSP1->CPSR = 0x2;
242
243 for ( i = 0; i < FIFO_SIZE; i++ )
244 {
245   Dummy = LPC_SSP1->DR;      /* clear the RxFIFO */
246 }
247
248 /* Enable the SSP Interrupt */
249 NVIC_EnableIRQ(SSP1_IRQn);
250
251 /* Device select as master, SSP Enabled */
252 #if LOOPBACK_MODE
253   LPC_SSP1->CR1 = SSPCR1_LBM | SSPCR1_SSE;
254 #else
255 #if SSP_SLAVE
256   /* Slave mode */
257   if ( LPC_SSP1->CR1 & SSPCR1_SSE )
258   {
259     /* The slave bit can't be set until SSE bit is zero. */
260     LPC_SSP1->CR1 &= ~SSPCR1_SSE;
261   }
262   LPC_SSP1->CR1 = SSPCR1_MS;      /* Enable slave bit first */
263   LPC_SSP1->CR1 |= SSPCR1_SSE; /* Enable SSP */
264 #else
265   /* Master mode */
266   LPC_SSP1->CR1 = SSPCR1_SSE;
267 #endif
268 #endif
269 /* Set SSPINMS registers to enable interrupts */
270 /* enable all error related interrupts */

```

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

Bit	Symbol	Value	Description
3:0	DSS	0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111	Data Size Select. This field controls transferred in each frame. Values 0C and 0D should not be used. 4-bit transfer 5-bit transfer 6-bit transfer 7-bit transfer 8-bit transfer 9-bit transfer 10-bit transfer 11-bit transfer 12-bit transfer 13-bit transfer 14-bit transfer 15-bit transfer 16-bit transfer
5:4	FRF	00 01 10 11	Frame Format. SPI ✓ TI Microwire This combination is not supported a
6	CPOL		Clock Out Polarity. This bit is only u:

$$f_{\text{SPI}} = \frac{\text{PCLK}}{\text{CPSIVSR}(S(z+1))} \dots (1)$$

\uparrow \nwarrow
 $[z, z54]$ $[z, z55]$