CMPE240 Spring 22

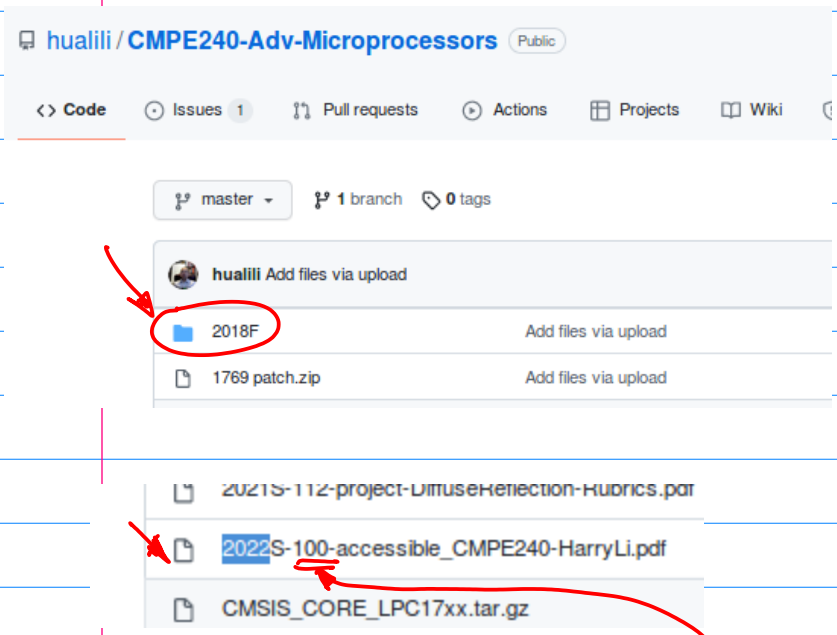
Jan26 (Wed)

Today's Topics: Introduction & Organizational meeting.

Harry Li, Ph.D.

Greensheet: On-Line from github

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



Naming convention: yr + 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 p.m.

Office hours Zoom link: Join Zoom Meeting
https://us04web.zoom.us/j/9841607683?pwd=UlA3aEk1TnV4bjNLQk5CQkw0dDk4UT09
Meeting ID: 984 160 7683 Passcode: 121092

Textbooks + Ref

1. No textbook, but NXP CPU Datasheet is utilized as a Base Line Ref
2. SCH Design of the CPU module.

TPS (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 White Board (Lecture Notes) And SJSU CANVAS.

Late Project submission will have 1 pt. Penalty per each Lecture Day.

Introduction.

1. Bill of material for LPC1769 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.

Prototype System.

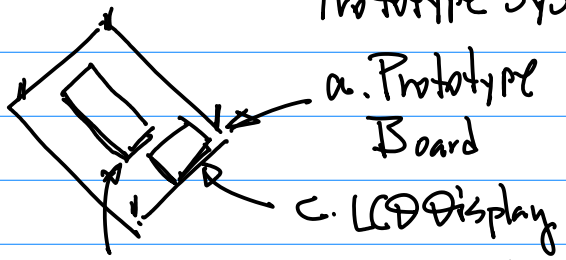


Fig. 1

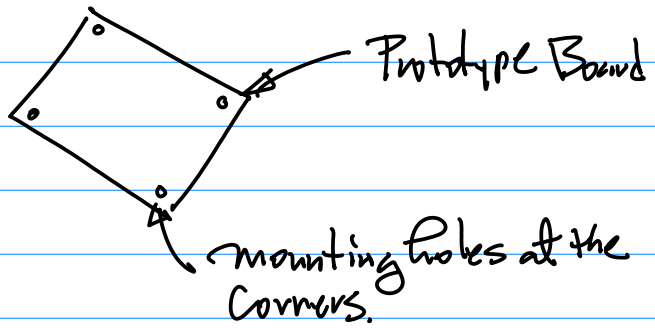
Note: a. Adequate Size Not Too big, to host LPC1769 module & LCD Display module, plus "Glue" Logic.

∴ "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 11½ cm.



Stand-offs: (Legs) for the Board.

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/Accessible

- 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

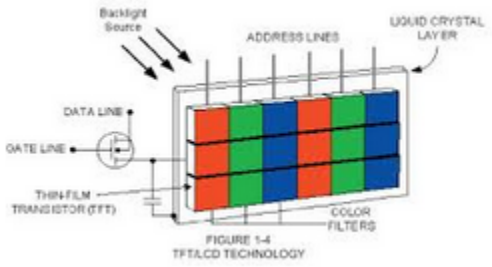


Fig 3

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 LPC 1769 GPU 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 deployed with External power.

7.5V ~ 9VDC @ 1500 mA ~ 4000 mA

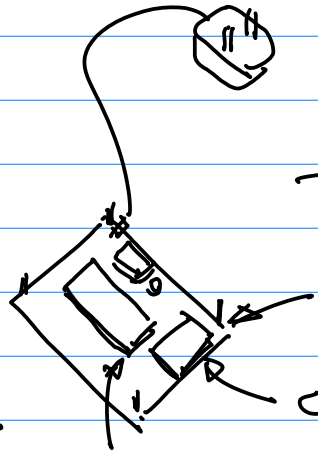


Fig. 4.

Prototype System.

a. Prototype Board

b. LPC1769 GPU module

c. LCD Display

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

Components for the Power Unit

① Wall-mount Apator.

Spec. 7.5 ~ 9VDC

1500 mA ~ 4000 mA.

Or Battery Pack.

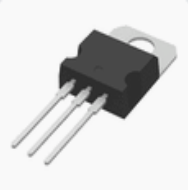
Fig. 5

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

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

③ Power Regulator IC 7805, or 1117.

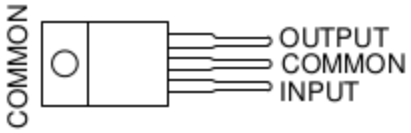
Fig6



STMicroelectr
L7805CV ...
$0.63
Digi-Key

- 3-Terminal Regulators
- Output Current up to 1.5 A
- Internal Thermal-Overload Protection

KC (TO-220) PACKAGE
(TOP VIEW)



KCS (TO-220) PACKAGE
(TOP VIEW)

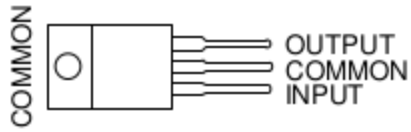
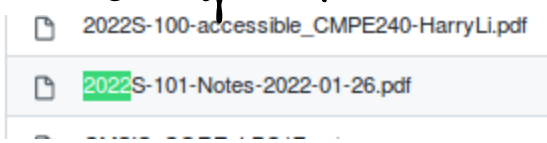


Fig.9.

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



Note: 78xx family
7812, 7805 etc.

µA7800 SERIES
POSITIVE-VOLTAGE REGULATORS

SLVS056J – MAY 1976 – REVISED MAY 2003

Components for GPIO
Use GPI "Hello, the world"
for Debugging { Input { "0"s "1"s
Output { "0"s "1"s

(4) Right Angle (plug) connector

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

Fig.7

Feb 2nd (Wed)
Today's Topics:
1° Bill of Material
2° CPU Datasheet, Architecture
(options for the target platform.
(LPC176q)

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

(6) Compensating Caps.

Note for ST mi lm the Caps Required are 2.

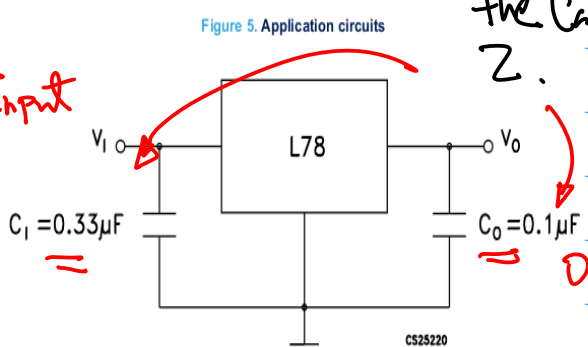


Fig.8

a. NVDA (Nvidia) Jetson NANO
Caution: Device Driver Programming in U.S. Kernel Space.
5% Bonus. Implementation at Registers/Hardware Level, LCD

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

Example: GPIO I/O Testing Circuit

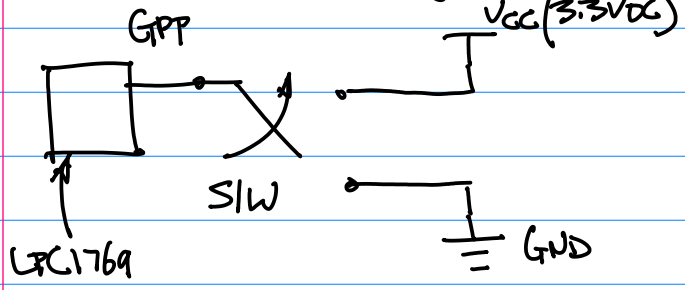
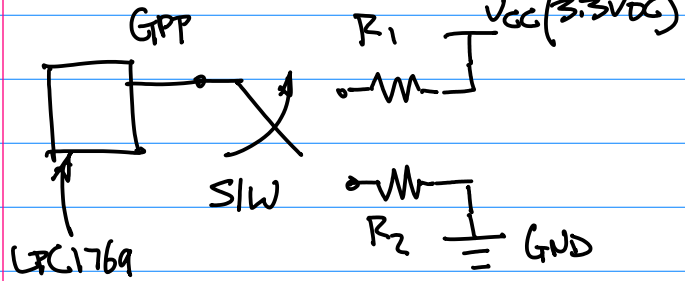


Fig1.

GPP: General Purpose Port, Same as GPIO.



Better Design.

Fig1b.

Consider Resistor Value Calculation

$V_{cc} = 3.3VDC$, $V_{out} = 3.3VDC$ (GPP)

Select Resistor Value to Regulate the Amount of Current ~10mA

Hence, $R_{1,2} = V_{cc}/10mA = 3.3/10\times10^{-3}$

$= 3.3\times10^{2}\ \Omega$.

GPP I/O Output Testing.

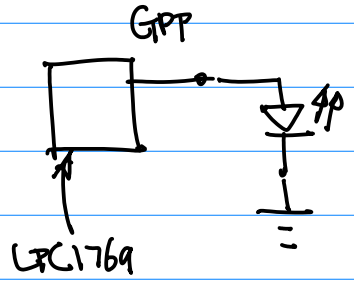
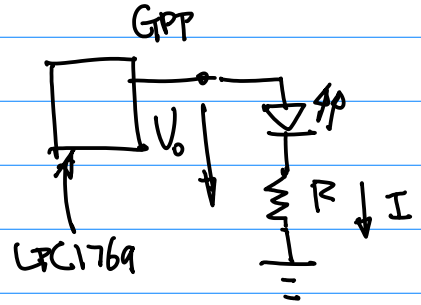


Fig.2



Calculation of the Resistor

$V_D = V_{LED} + IR$ ... (1)

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

Substitute the above conditions into Eqn(1),

$3.3 = 1.2 + R\times10\times10^{-3}$

$R = 2.1/10^{-2} = 2.1\times10^{2} = 210\ \Omega$

Note: please use right size of the prototype wire 26~30 AWG.

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

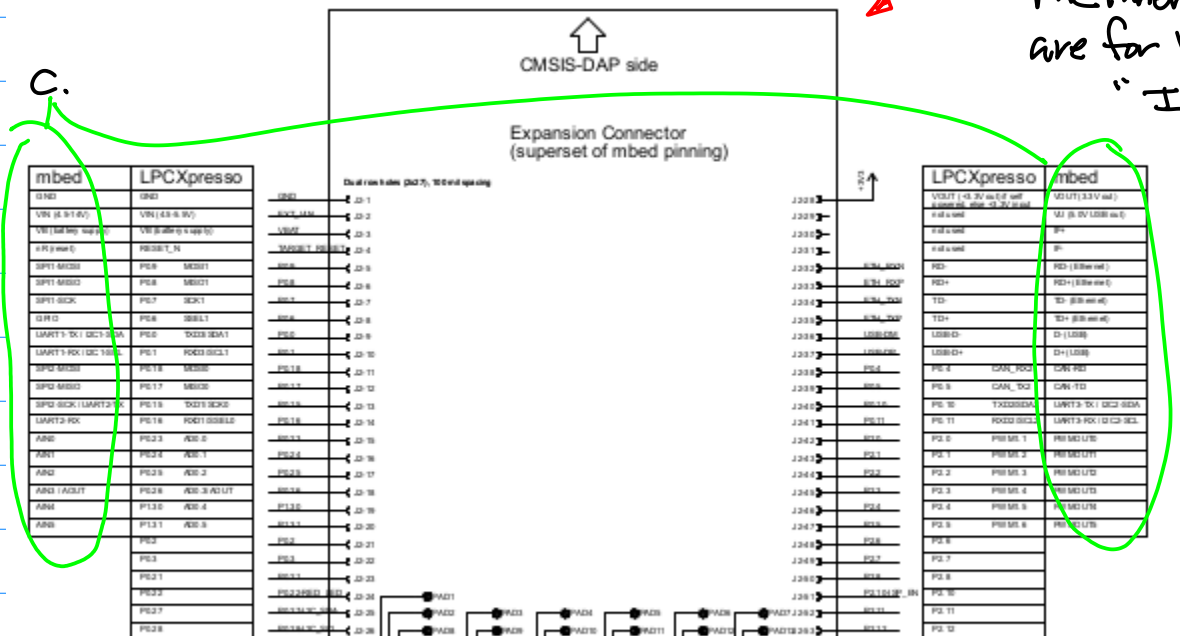
LPCXpresso1769_CD_revD(1).pdf

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

b.

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

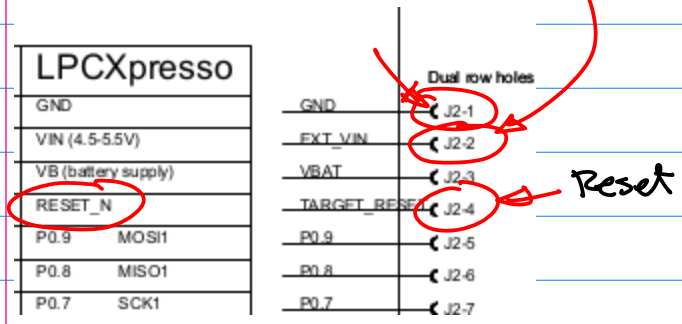
c.



CmpE240 Spring22

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

Example: Power Input to LPC1769



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

Note: physical mapping of the pin to Actual module;

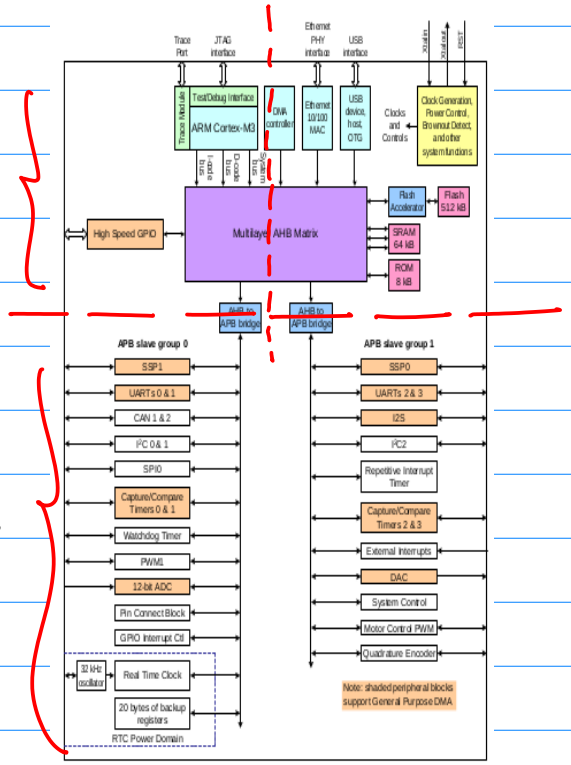
Note: Reset pin — provide Access to this pin in your Prototype design.

CPU Datasheet from github / ~lcmpe244

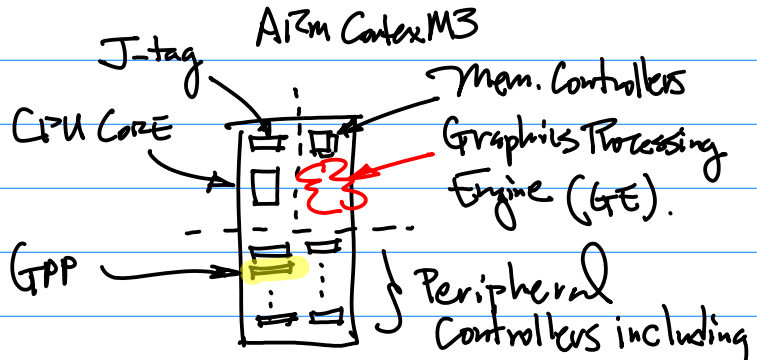
2021F-107-lpc-cpu-UM10360.pdf

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

Step 1. CPU Architecture, pp.9 & 12



Peripheral Controllers



Feb7 (Monday).

Today's Topics 1° CPU Datasheet & Schematics; 2° Hardware Design for the Prototype System.

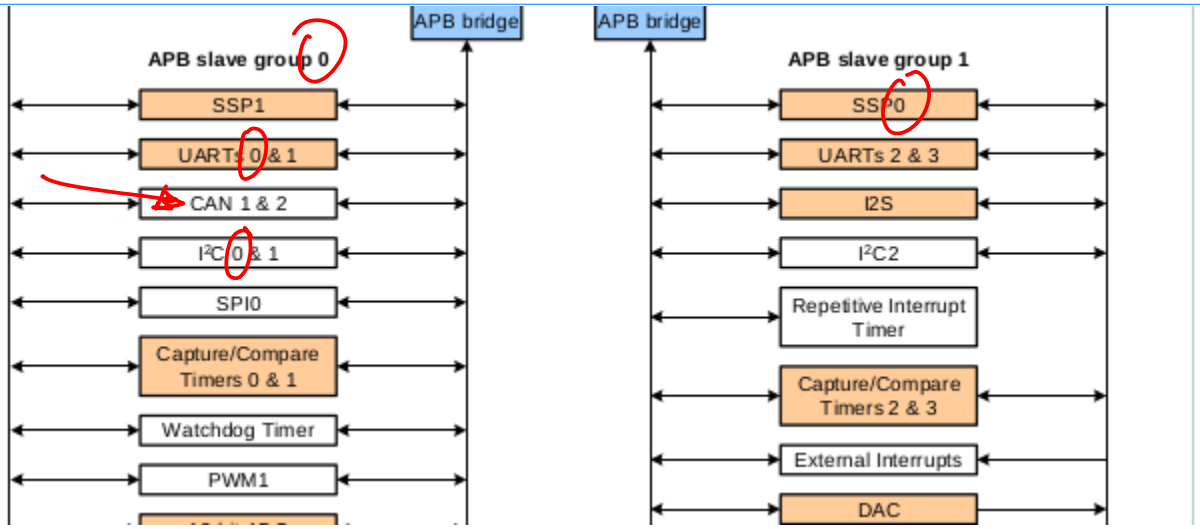
Select GPP (General Purpose Pin) to Build GPIO I/O to Realize "Hello, the world" function. (with Input Testing / Output Testing Circuit)

UM10360

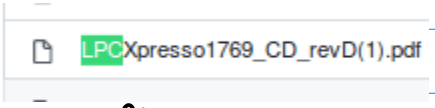
LPC176x/5x User manual

Rev. 3.1 — 2 April 2014

Note: Peripheral Controllers



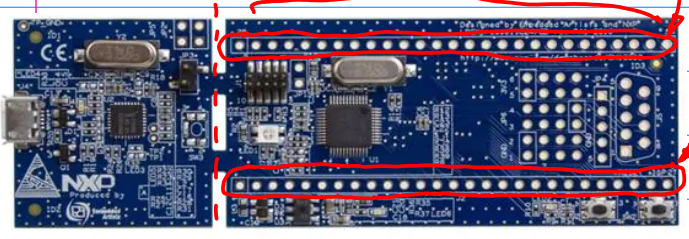
Step 2. Map the GPP to its physical Board, e.g., CPU module. Hence, we need schematics.



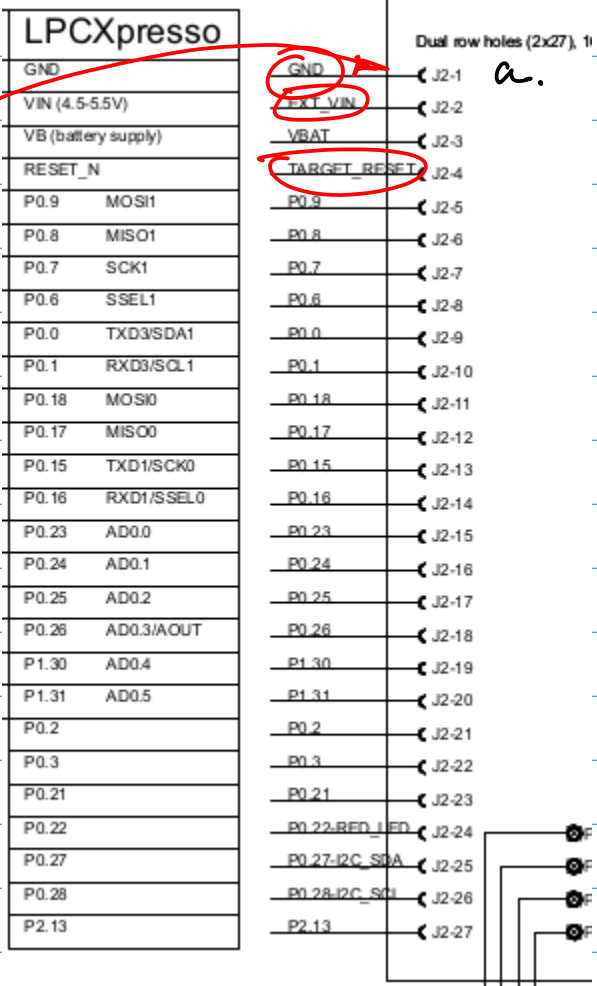
from the Class github.

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

Do the Same of your physical CPU module. J2 connector



Interface to the host

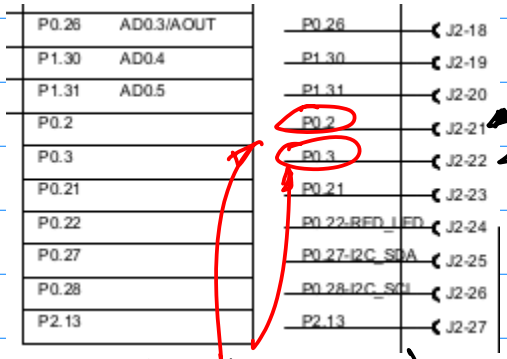


a.

Step3. Select 2 GPIO pins

One for Input Testing Circuit, P0.2 Input, P0.3 Output
The other for Output Testing Circuit.

Table 1. Connectivity & Pin Assignment Table
3 col. format



Connector Pin

GPP Pins, CPU Naming.

CPU/Connector Pin	Description	Note
P0.2/J2-21	GPP Input	
P0.3/J2-22	GPP Output	

CPU Datasheet for General Purpose Port 0
pp117

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

a b. c. d. e.

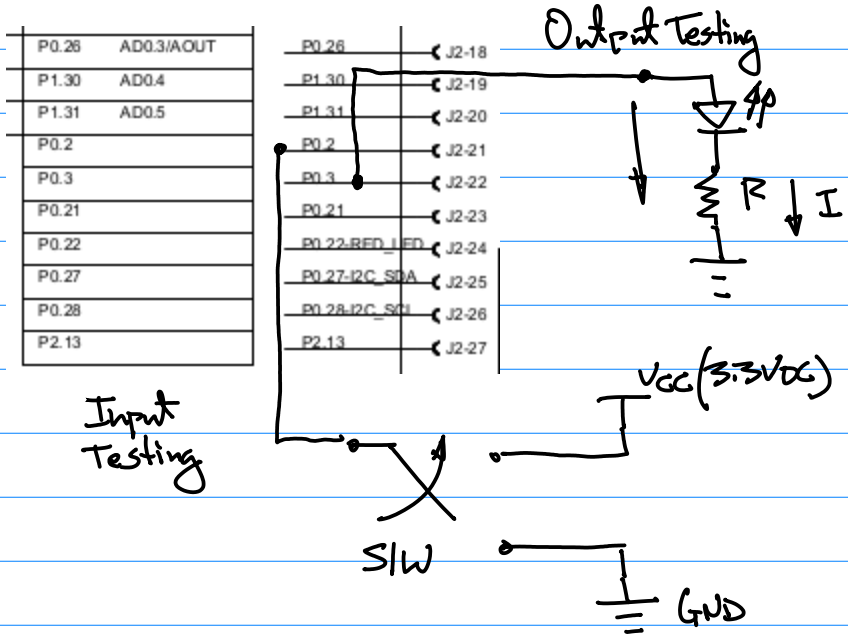
Table 80. Pin function select register 0 (PINSEL0 - address 0x4002 C000) bit description

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[1]	GPIO Port 0.4	I2SRX_CLK	RD2	CAP2.0	00
11:10	P0.5[1]	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

PINSEL0[5:4] = 2 bits @ Bit 4 & 5

- 00 GPIO
- 01 TX
- 10 ADC
- 11 Not used.



TP14.

Homework: Due A week from today (Feb14).

1° Draw Schematics Design for GPP I/O 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 on Line to SJSU CANVAS.

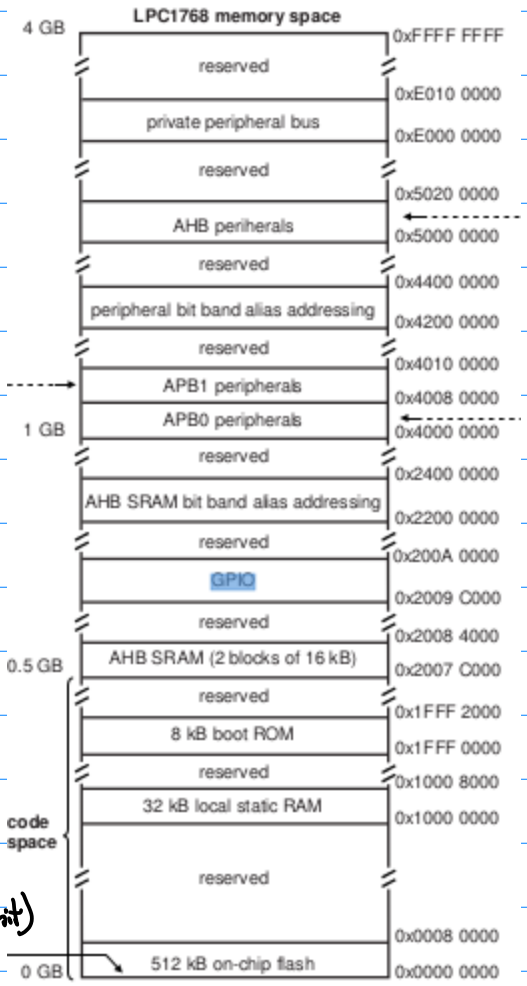
Consider CPU Architecture Discussion.

Example: CPU 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

2. Addr. Bus 32 Bit. ← Least Significant bit (Little Endian)

$a_{31} a_{30} a_{29} \cdots a_2 a_1 a_0$

32 bits Address

Most Significant Bit

"Little Endian" V.S. Big Endian

Feb 9 (Wed) Today's Topics:

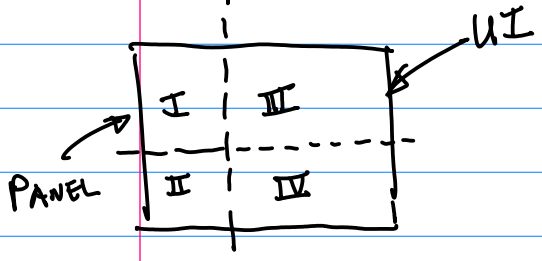
1° CPU Architecture;

2° MCU Xpresso, IDE, GPIO Testing Program.

Example: Compilation & Build First Firmware Program for GPP I/O Testing.

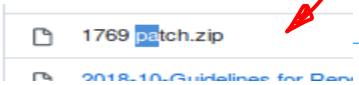
Note: Visit NXP website, Sign up to Become a Developer. Download MCU Xpresso. Version 11.3.0

Step1 = Start your IDE Xpresso.



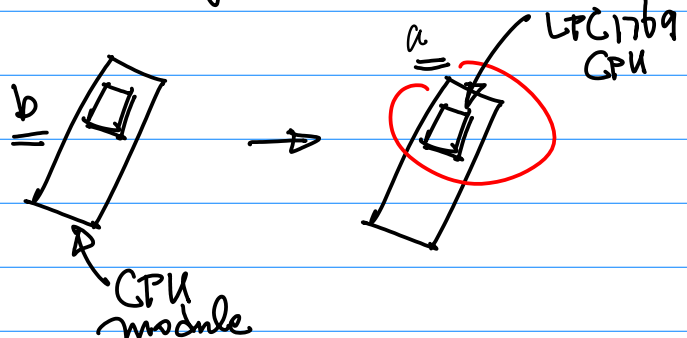
Project Panel (I):

Import LPC17ba patch from 240 Class github.



Step2 =

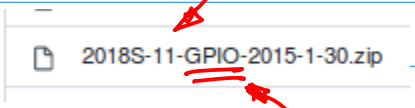
To set up the IDE Development Environment to match your target platform.



Go to "Quickstart" Panel to Import LPC17ba Patch, And projects

Note: There are several projects, pre-tested for your references purpose including GPP, SPI, etc.

Step3 =



Import GPIO Project (Sample code) for your Homework.

Step4. Build & Execute GPIO Sample Project

Highlight GPIO project at the project Panel → Move to Quickstart Panel, click on "Build" to Build it.

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

Step5 Flash/Write 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 IDE 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 1 1/2 Week's from Today)
Feb21, Monday 11:59pm.

1. Build A Prototype System for GPIO Testing, Input/Output Testing.
2. Provide Schematics 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
   FirstName-LastName-SJSU-GPIO.zip
7. Submit to CANVAS.

Feb14 (Monday)

Today's Topics: 1° CPU Architecture
Base Reference: CPU Datasheet from the class github/hualili/Cmpe244 2021F-107 ~

Memory map pp13. ↔ CPU Block Diagram pp9,12
↕
IDE XPRESSO12

Example: 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 (Addr. ~ / Data ~ 32 bits)
   - R.F. (Register File) 32 bits

R.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 Peripherial Controller.

ALU, 32 bits
(Arithmatic/Logic Unit).

3. Memory Map.

Question 1: How many individual memory Address are there for 32 Bit Architecture?

$2^{32} = 2^2 \cdot 2^{10} \cdot 2^{10} \cdot 2^{10}$ = 4G individually Addressable memory Cells.

4 | 1K | 1M | 1G

4. Byte Addressable Machine

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

"Byte Addressable Machine"

Question 2: What is the unit for the memory Cells mentioned Above in Question 1? A Byte.

Therefore, 32 bit Architecture → 4 GiByte

Question 3: Consider A Special Purpose Register, which has 4 Bytes (Because of 32 Bit Architecture), How many possible Addresses for Each Special Purpose Register to Cover?

4.

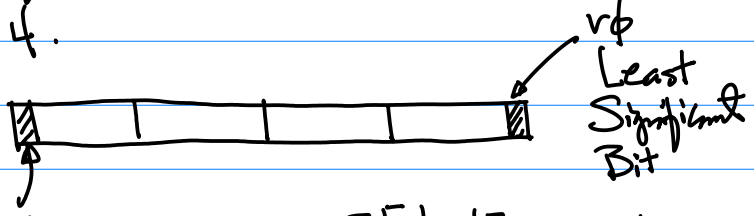


Fig. 1

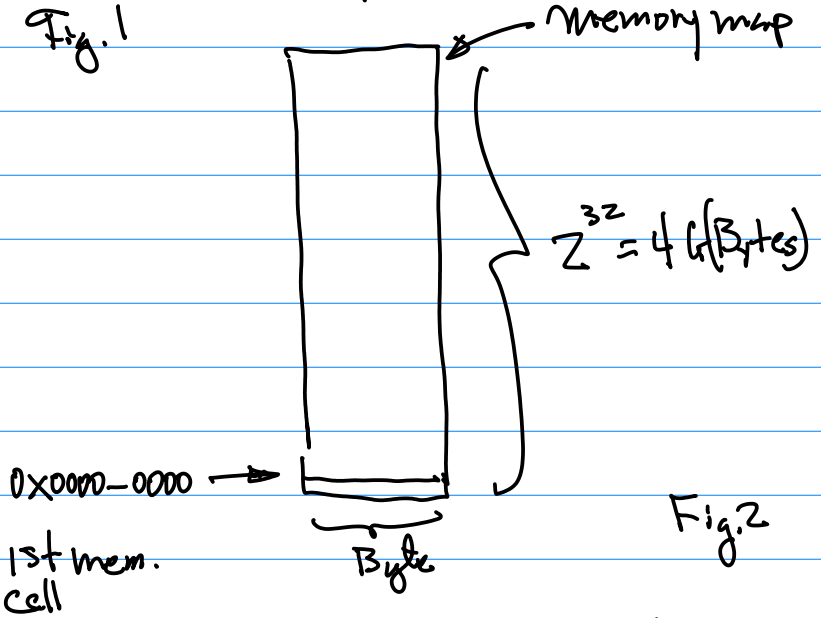
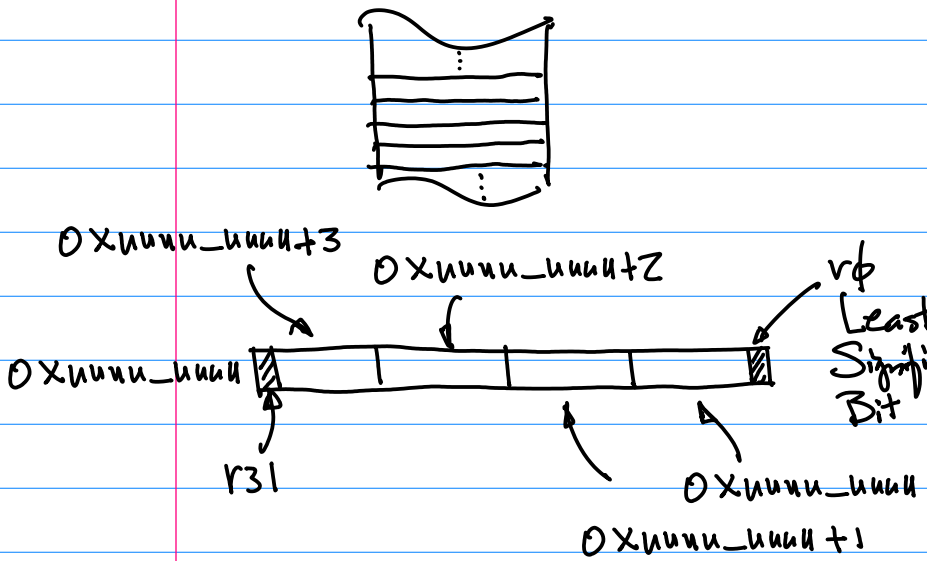


Fig. 2

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

6. Divide memory map into 8 Equal Banks.

$$2^{32}/8 = 2^{32}/2^3 = 2^{29} = 2^9 \cdot 2^{20} =$$

512m (Byte) 1m

the size for Each Memory Bank.

Feb16 (Wed)

Topics: 1° CPU Architecture Discussion

2° Prep. for the project of Building 2D GPE (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 → Boot process → Boot Sequence

      0X0000_0000

   c. $a_{31}a_{30}a_{29}$ (3 Consecutive Bits to define each mem. Bank)

      Question: which 3 bits from the Addr. Bus Do we need to uniquely define Each mem. Bank?

      3 most Significant Bits → $a_{31}a_{30}a_{29}$

      $a_{31}a_{30}a_{29}\phi\phi\phi\phi\phi$ ..... $1\phi\phi\phi\phi$

      Don't Cares

   d. Defining Starting Addr. for Each mem Bank

a_{31} a_{30} a_{29}	a_{28} a_{27} ... a_1 a_0		Starting Address
0 0 0	0 0 ... 0 0	BANK0 (1st)	0X0000_0000
0 0 1	0	BANK1 (2nd)	0X2000_0000
0 1 0	0	BANK2 (3rd)	0X4000_0000
⋮		⋮	
1 1 1	0 0 ... 0 0	BANK7 (8th)	0xE000_0000

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

(b) GPP port 7

Sol: Option1. From GPU Datasheet, ⇒ memory map.

GPIO Block Starting Addr.

0X2009_C000 ;

Option2: (more In-depth information)

GPP Peripheral controller →

Special Purpose Registers → Configuration OR Control Register : To Config And init GPP Peripheral Controller ← Addr. of the Register

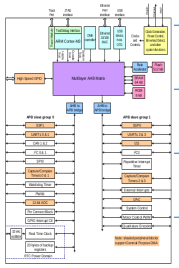
FIODIR (Fast I/O Direction defining Register, e.g.; Define Input/Output Direction)

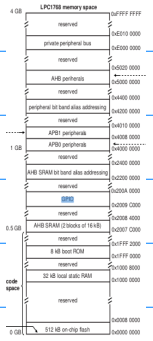
2nd memory Bank starting addr: 0X2000_0000

3rd " " " " : 0X4000_0000

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

GPU Architecture Block Diagram ⇔ 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

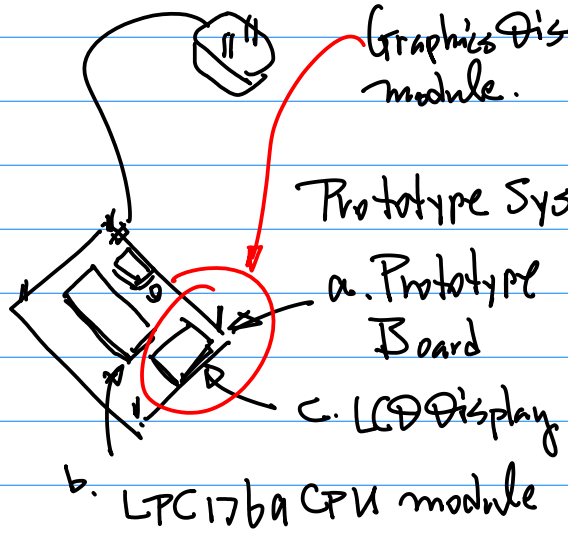


Fig.2

(pp.3)

Design Guideline:

1° SPI Interface (Serial Peripheral Interface) to provide Adequate Bit Rate. 50 mbps ~ 100 Mbps

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

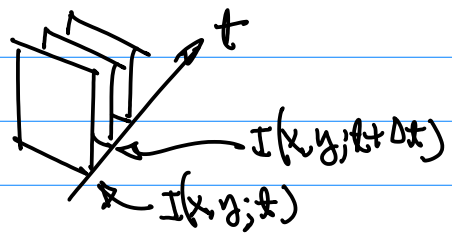
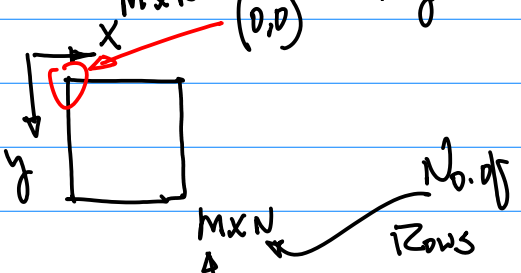


Fig.3

Resolution: M x N



720p, 1080p High Resolution

1024x768 Comparable to 720p.

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)

1024x768x24x30

Feb21 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.

1024x768x24x30

$= 2^x$?

1024x768x24x30

2^{10} 2^{10} 2^5 2^5 =

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

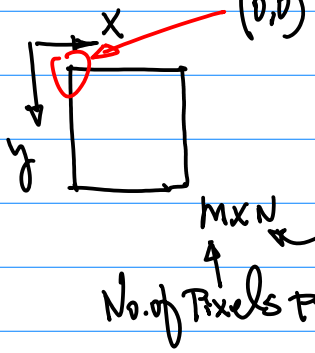
$2^x = 2^9 = 512$

768

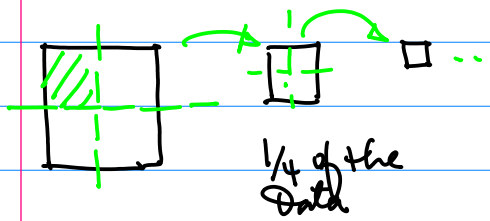
2^{10} 2^{10} 2^5 2^5

$= \underbrace{1K \cdot 1K}_{1M} \underbrace{1K}_{1K} = 1$ Gbps

CmpE240 Feb21,22



Reduce Resolution



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.

LPCXpresso1769_CD_revD(1).pdf

1. SPI Hardware Interface requires 4 pins "3+1" pins

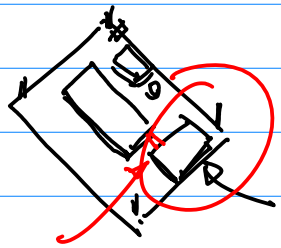
	Name	
Output pin	MOSI	Master Output Slave Input
Input pin	MISO	~Input~OP
Clock pin	SCK	

plus "Enable" pin SSELx ← more than One SPI.

Table1. 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.



C. LCD Display

SPI Interface

Fig.2

Identify the Right Suitable LCD module.

2018S-8-SPILCD-Connector-1-HL-2...

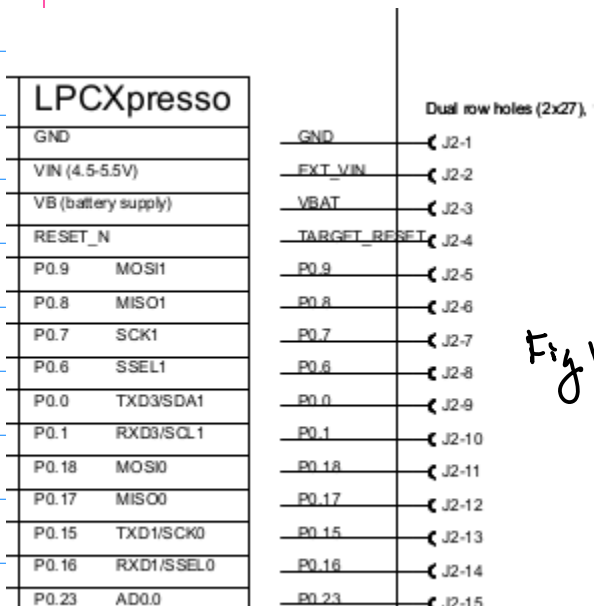
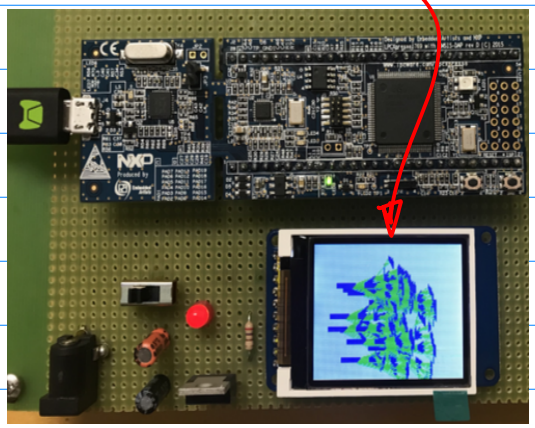
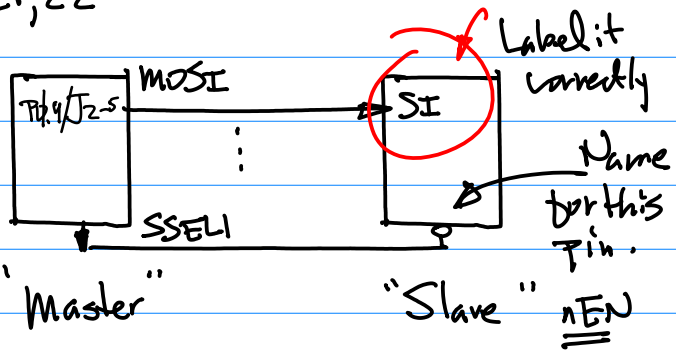


Fig.1.

CmpE240 Feb21,22

LCD Sample module





Note:
Pin Number, Label of the pin, direction of the Signal Flow, if logical Low Active, then "circle" has to be provided as well.

Pin Connectivity Table for Schematic Design.

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

The Base to Build Connectivity Table for LCD module

LPC-1769		LCD
Label	Pin	
MOSI	P 0.18	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

Exercise: Conectivity Table for LCD, use the following Table format

Note:

- LPC 1769
- Color TFT LCD display

Resolution : 128x160,
Pixel Depth: 18-bit (262,144) colors
Controller: ST7735
Interface: SPI interface

Bit Rate for 30 FPS, and 18-bit Pixel Depth

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

Serial Input for LCD

Provide Pin Number of the LCD module

Pin Name, Pin Number of the CPU module

Note 1. Defining which Device is "master", which Device is a "slave".
In our case, CPU (LPC1769) is a master

Interface Signals on LCD (General Guideline)

- 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 SI
- Control Backlight — GPIO from the CPU
- Toggle Data/Command — GPIO from the CPU

Exercise: Build Connectivity Table for the CPU & SPILCD Interface.

a. By Next Monday, Feb. 28.

b. One col. to cover all CPU pins for SPI, for Backlight, for Command/data, Enable signal (to enable the LCD Display module).

c. One col. to match each CPU pin, to define the pins connected to the CPU module.

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

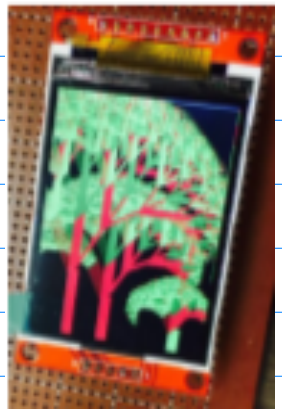


Fig 1.

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

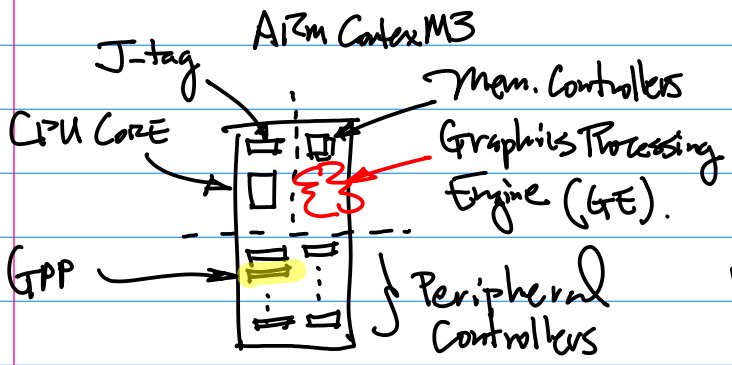
Fig 2

Fig 3

CmpE240 Feb 23rd, 22

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

Example: 2D Graphics Engines:



Graphics Engine
1. Display Driver
2. 2D GE
3. 3D GE
4. Video Codec

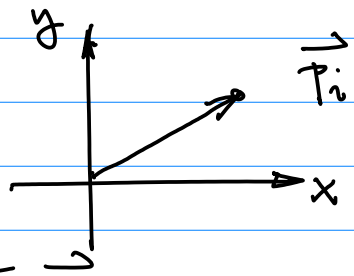
2D G.E.
Vector Graphics Primitives
: 2D Transforms
: D.D.A.

Theoretical Background On 2D Vector Graphics.

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

Define A 2D Vector

2D



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

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

where λ is scalar

Feb 28 (Monday)

First, Continuation of Prototype Board Demo/Inspection.

2D Vector Graphics.

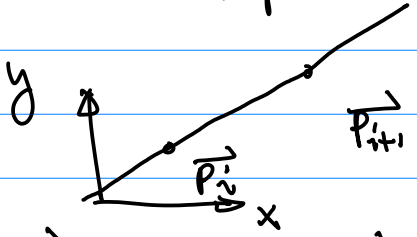


Fig 1.

$\vec{P_i}$ Starting point; $\vec{P_{i+1}}$ Ending point

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

Direction Vector.

Starting point, A fixed point

Note: 1. Define A Directional Vector

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

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 $\vec{d}(x,y)$?

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

From the given condition, we have

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

Note 2. Eqn(1),

a fixed point (Starting pt) + λ · directional Vector

Scalar, scaling

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

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

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

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

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

Verify: $\vec{P}(x,y) = \vec{P}_i + \lambda(\vec{P}_{i+1} - \vec{P}_i)\Big|_{\lambda=1}$

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

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

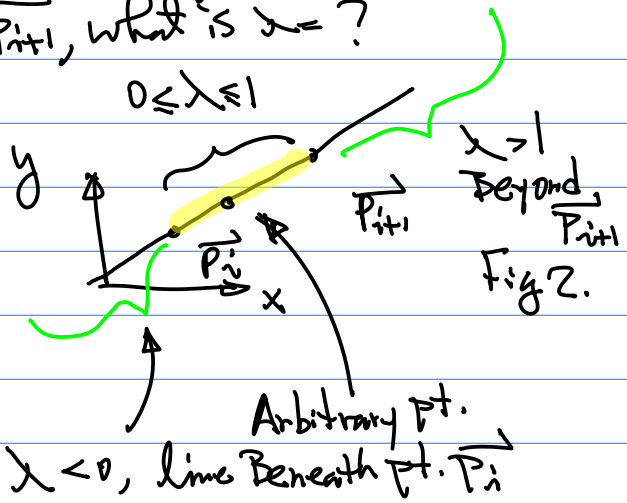


Fig 2.

Conclusion:

Conclusion: For Eqn(1), when

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

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

$0<\lambda<1$, Any pts Between $\vec{P_i}$ & $\vec{P_{i+1}}$

$\lambda<0$ Beneath $\vec{P_i}$

$\lambda>1$ 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.

$\vec{P_0}(x_0,y_0)=(25,5)$

$\vec{P_1}(x_1,y_1)=(25,25)$

$\vec{P_2}(x_2,y_2)=(5,25)$, $\vec{P_3}(x_3,y_3)=(5,5)$

$\{\vec{P_i}(x_i,y_i) \mid i=0,1,\ldots,3\}$ ...(2)

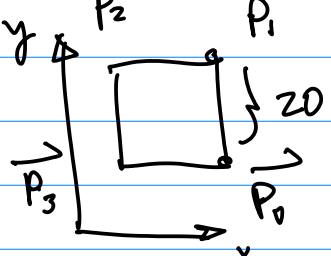


Fig.5.

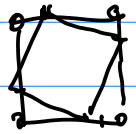


Fig.6

March 4 (Wed.)

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

Requirements

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

a.

b.

Fig5

2. Implementation has to be individual with his/her own Board; No Code/Program is allowed to shared. 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 Replied from 1-3;
③ Video Clip (5 seconds)

Zip together, Submit it to SJSU CANVAS.

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

Example: Continued from Rotating Square Design.

Step 2.

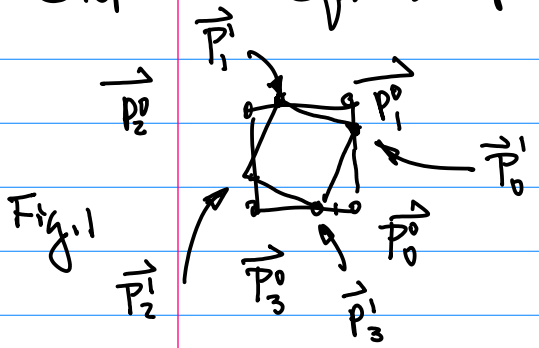


Fig.1

If we continue this process with a new set of Vertices $\{\vec{P}_0^1, \vec{P}_1^1, \cdots, \vec{P}_3^1\}$ 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:

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

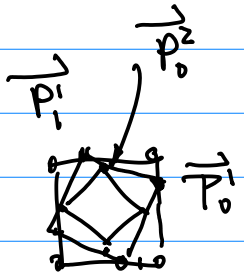


Fig.2

Counter clockwise Rotation

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

Question: what is λ=? For the Rotation in Fig2

$\lambda = 0.8$

Question: To generate Opposite Rotation direction, what is λ=? $\lambda = 0.2$

Generally, the formula for the above Rotating Squares:

From Eqn (1),

$$\vec{P}_i^{j+1} = \vec{P}_i^j + \lambda(\vec{P}_{i+1}^j - \vec{P}_i^j) \quad \cdots (2)$$

e.g.

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

$$\cdots (2b)$$

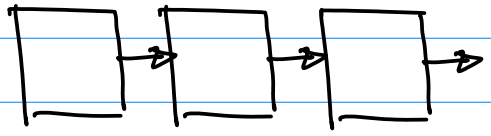
Consider Hand Calculation &
C/C++ Implementation.

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

Or

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

$$= \left(x_i^{j}, y_i^{j}\right) + \lambda\left(x_{i+1}^{j} - x_i^{j},\ y_{i+1}^{j} - y_i^{j}\right) \quad \cdots (3)$$



Background Blue | Background Brown | A tree

Or Equation for X-Component,

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

Similarly,

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

Therefore,

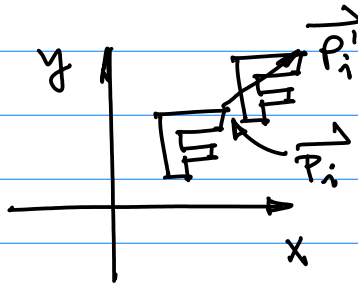
$$\begin{cases} x_i^{j+1} = x_i^{j} + \lambda\left(x_{i+1}^{j} - x_i^{j}\right) & \cdots (3a) \\ y_i^{j+1} = y_i^{j} + \lambda\left(y_{i+1}^{j} - y_i^{j}\right) & \cdots (3b) \end{cases}$$

In C-code for Eqn(3a)

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

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

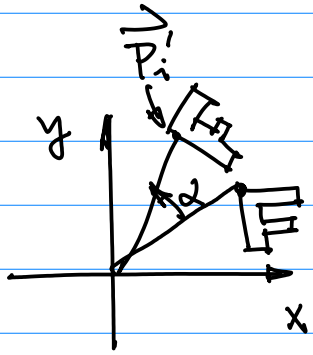
Math. Formulation, 2D Transforms



After → $\{\vec{P}_i' \mid i = 0, 1, 2, \cdots, N-1\}$

$\{\vec{P}_i \mid i = 0, 1, 2, \cdots N-1\}$ ← Before

1. Translation.



2. Rotation.

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 $\{\vec{P}_i \mid i=0,1,2,\ldots N-1\}$

And

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

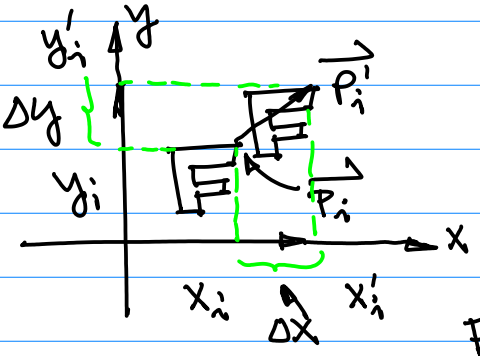


Fig. 1

"Before–After" From

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

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

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

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

From $x'_i = x_i + \Delta x$ ... (1)

For y, we have a similar Relationship

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

(After) (Before)

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

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

$\vec{P}'_i$ After

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

$\vec{P}$ — Dummy Dimension

Now, Consider Rotation.

$$\begin{pmatrix} x'_i \\ y'_i \\ 1 \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}_{3\times 3} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} \quad \ldots (4)$$

March 7th (Monday)

1. Continued Prototype Board Show & Tell, Inspection
2. Project 1. Written Requirements

/ 2018F / 2022S-103-project-2D-GE-2022-3-7.pdf

Due March 19 (Sat. 11:59pm)

Midterm March 23rd (Wed) on-Line Zoom.

2D Transforms. — Translations

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

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

Ref:

2018F-111-lec-2D-vectors-v3-2018-9-19.pdf

2018F-112-lec4-2D-vectors-v3-2018-10-1.pdf

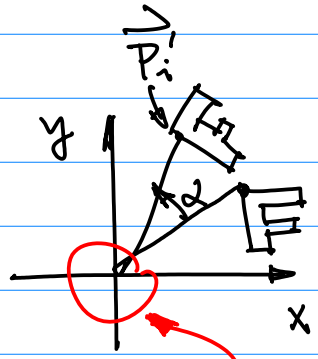


Fig.2

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.

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

Example: Find Rotation Point $\vec{P}_i'(x_i', y_i')$ for the following Operation.

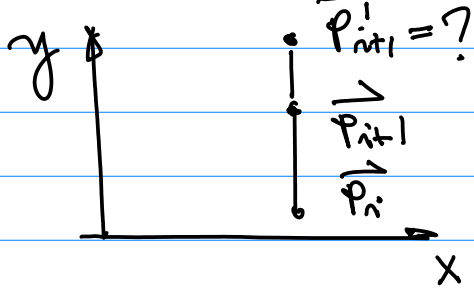


Fig.3a

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 $\alpha = 30°$ Counter Clockwise Direction.

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

Which Allows us to find $\vec{P}'_i$ in Fig.3a

Consider Rotation to find $\vec{P}''_i$ in Fig.3b.

Question: Can Eqn(4) be Applied directly to find $\vec{P}''_i$?

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

Angle α w.r.t (0,0) we Can not directly use the above matrix for Rotation in Fig.3b.

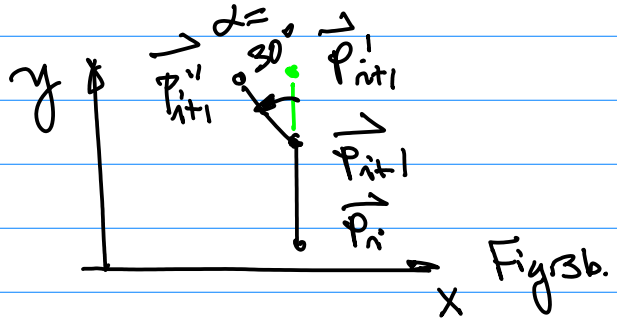


Fig.3b

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 23rd (Wed.) on-Line Zoom.

Prep. for the midterm.

1. Midterm Exercise on Monday March 14.

Bring 2 Blank Printer Papers to the Class. for Midterm Exercise

Put your Name: a First-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-mails 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 on-Line 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

b. Zip it.

10. Submit the Zipped file to SJSU CANVAS

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

Ref

2018S-10-LCD-DrawLine.zip

Example: Continued.

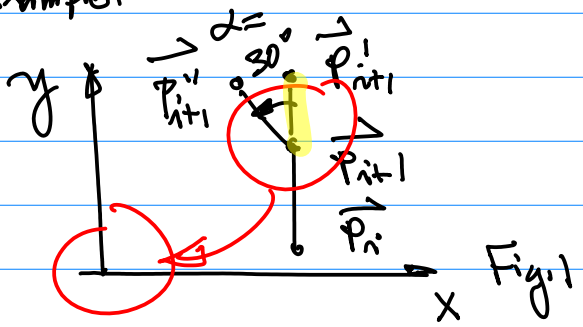


Fig.1

Step 1. Preprocessing.

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

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

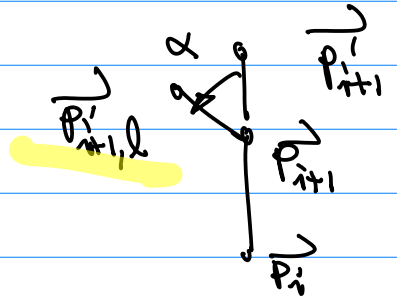
where $\Delta x = -x_{i+1}$, $\Delta y = -y_{i+1}$

Step 2. Rotation.

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

where $\alpha = 30°$

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



Step 3 Post-Processing.
(Undo Pre-Processing)

$$\begin{pmatrix} 1 & 0 & \Delta x \\ 0 & 1 & \Delta y \\ 0 & 0 & 1 \end{pmatrix}^{-1}$$ to Be Rewritten as

$$T' = \begin{pmatrix} 1 & 0 & -\Delta x \\ 0 & 1 & -\Delta y \\ 0 & 0 & 1 \end{pmatrix} \quad \cdots (3)$$

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

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

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

C/C++ :

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

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

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

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

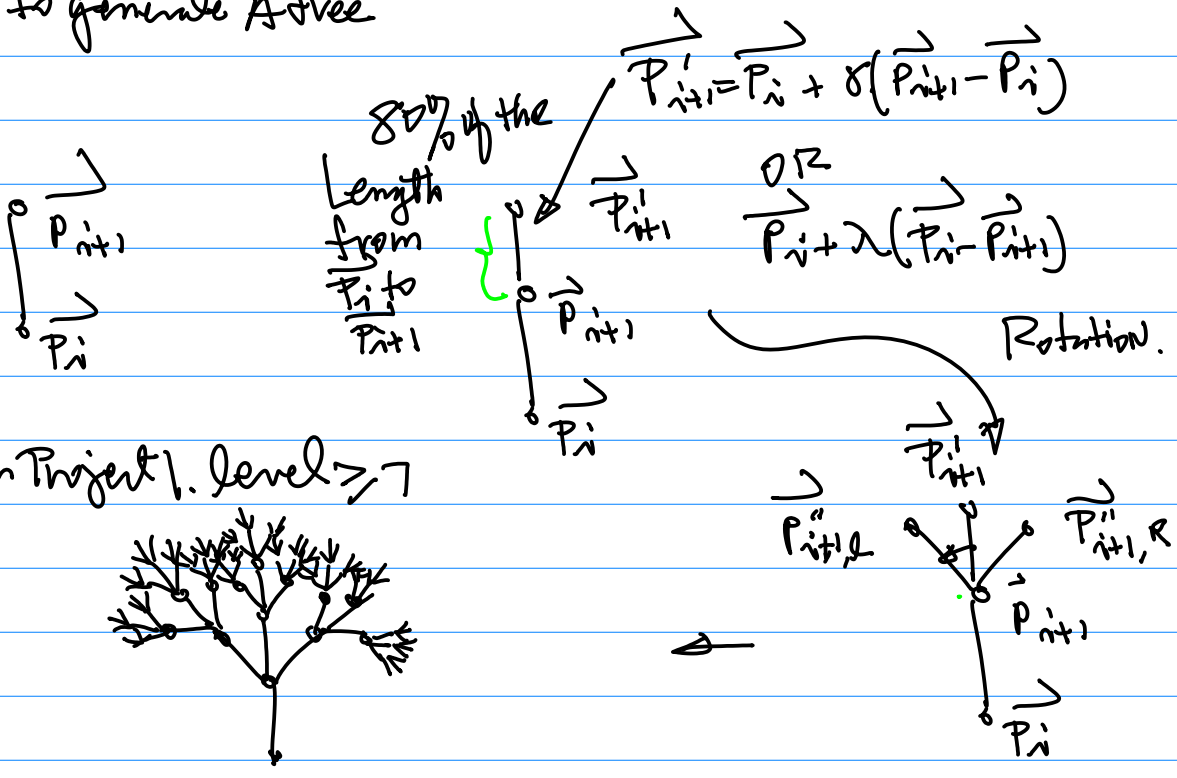
$$= \begin{pmatrix} \cos\alpha & -\sin\alpha & \cos\alpha\Delta x - \sin\alpha\Delta y - \Delta x \\ \sin\alpha & \cos\alpha & \sin\alpha\Delta x + \cos\alpha\Delta y - \Delta y \\ 0 & 0 & 1 \end{pmatrix}$$

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

Consider GPIO Architecture.

Datasheet. SSP from the github

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

Prefix + CoN + Postscript Peripheral Control Register

Power

Root: Control/Config

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 Register(s) Which Defines/Dictates the Behavior of Peripheral Controller.

Note: Address in A memory Bank, find it
0X4000_0000 → 3rd Bank

d. Notation
CR0[3:0]

a. SSP : A Super set of SPI

&CR0 = 0x4008_8000

18.6 Register description

The register addresses of the SSP controllers addresses are shown in Table 370.

c. Addr.

b.

Table 370. SSP Register Map

Generic Name	Description	Access	Reset Value[1]	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

Control Register0 →

Control Register1 →

e. CLK, Bus, Data size

f. Master/Slave

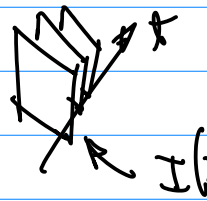
CmpE240 March 9 (Wed), 22

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 t transferred in each frame. Values 0000-0010 are no and should not be used.
		0011	4-bit transfer

Objective : Define ① Clock Rate, ② Datasize, And ③ SPI I/F

Tech. Spec. → Init & Config. for CRP.



I(x, y, t) 160x120x20x16

Find Bit Rate, Define CRP. for init & Config.

March 14. (Monday)

Example: SPI CRP for SPI I/F

Table 40. Peripheral Clock Selection register 0 (PCLKSEL0 - addres description

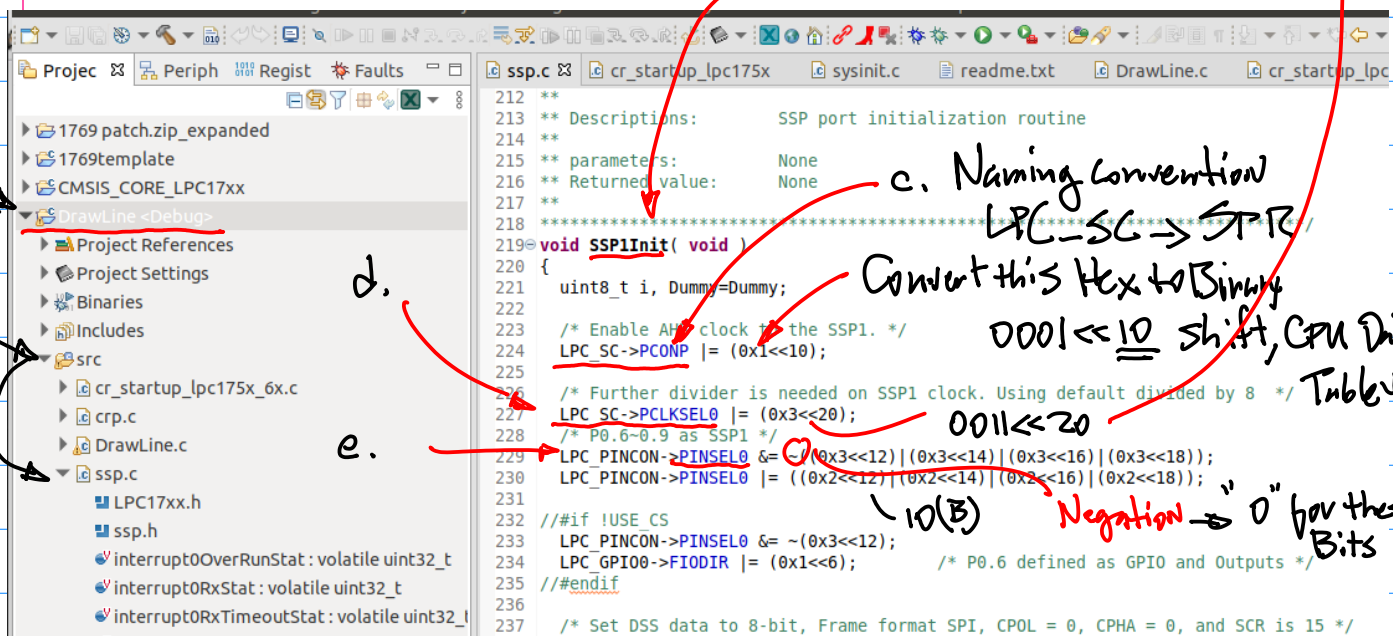
Bit	Symbol	Description
1:0	PCLK_WDT	Peripheral clock selection for WDT.
3:2	PCLK_TIMER0	Peripheral clock selection for TIMER0.
5:4	PCLK_TIMER1	Peripheral clock selection for TIMER1.
7:6	PCLK_UART0	Peripheral clock selection for UART0.
9:8	PCLK_UART1	Peripheral clock selection for UART1.
11:10	-	Reserved.
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.
23:22	PCLK_DAC	Peripheral clock selection for DAC.
25:24	PCLK_ADC	Peripheral clock selection for ADC

Ref: from github

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

b. Init

a



c. Naming convention LPC_SC -> STR

Convert this Hex to Binary

0001<<10 shift, CPU Datasht Tab46.

0011<<20

10(B)

Negation → "0" for these Bits

d.

e.

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[1]	GPIO Port 0.4	I2SRX_CLK	RD2	CAP2.0	00
11:10	P0.5[1]	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

f. 10(B) → 0x2, Bit Location in PINSEL0

0x0707 = 0x707

0000 0111 0000 0111

16 bits.

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

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

Note: 1. CR0 Control/Configuration Register

LPC_SSP1 -> CR0

Family P-Controller, Addr. of LPC_SSP1->CR0 : 17xx.h.

Init & Config.

```
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;
242
243   for ( i = 0; i < FIFOSIZE; 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 */
```

		the clock line.
15:8	SCR	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]).

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

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

✓ 0x7

CR0[15:8]

Serial Clock Rate:

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

PCLK: Peripheral Clock from CPU. → Peripheral Clock is defined By S.P.R. with System Clock.

1/2, 1/4, 1/8, ... of the System Clock.

CPSDVSR: Diviser. [2,254]

SCR: Serial Clock Rate R0[15:8]

$2^8 = 256$ (8 bits) ← SCR ∈ [0,255]

Example: Suppose PCLK = 50MHz

(1) f_{SPI} minimum clock Rate;

(2) f_{SPI} maximum clock Rate?

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

[2,254] [0,255]

Sol. From Eqn (1).

$$f_{SPI,min} = \frac{PCLK}{CPSDVSR\ SCR+1)} = \frac{50\times10^6}{CPSDVSR(SCR+1)} = \frac{50\times10^6}{254(255+1)} \simeq 769Hz$$

$$= \frac{50\times10^6}{254\times256} = \frac{50\times10^6}{65024}$$

March 16 (Wed) Topics:

1. SPI + SPR + Coding,
2. Review for the Midterm

Example: 0x SPI CR0

Tech. Spec → Binary Pattern → CONFIG.

Continued from Table 371.

CR[3:0] = 0111 (B) = 0x7 → 8 bit Packets