

August 21 (Monday)

Organizational Meeting.

1. The "GreenShot" is posted on the github

Note: Bring your Laptop Computer to the class.

<https://github.com/hualili/CMPE244>

Course and Contact Information

Instructor:	Harry Li, Ph.D. Professor, Computer Engineering Department State University
Office Location:	Engineering Building 267A
Telephone:	(408) 924-4060 (650) 400-1126
Email:	hua.li@sjsu.edu
Class Days/Time:	Mondays and Wednesdays, 4:30 pm – 5:45 pm, Aug
Office Hours:	<u>Mondays and Wednesdays, 3:00 pm – 4:00 pm</u>
Classroom:	Engineering Building Room 295
Prerequisites:	CMPE 180A and CMPE 180D, classified standing, c Artificial Intelligence or Computer Engineering or S majors only.

2. Emphasis on POSIX O.S. Linux Open Source O.S. & Device Drivers Programming and Development. Scalability & Vertical Solution.

Course Description

Experiments dealing with advanced embedded software programming concepts, interfacing techniques, hardware organization, and software development using embedded systems. Individual projects.

3. Course Format: In-Person.

Hands-on Class. Prototype System

Option 1. NVIDIA Jetson Nano. (GPU (128)
4 GB Version GPU JetPack

Option 2. Broadcom Pi3B+, Pi4.

Option 3. RISC-V FPGA Dev. Board.
+ FreeRTOS

Selection Decision in 1 week

Option 4. NXP LPC1114 or
LPC1779, RTOS. NXP
Dev. Forum.

Has limited Processing power.
May Not meet the need for our Project

4. Textbook & References

Set I: Datasheet(s), CPU Datasheet, Developer Guide; Set II: NVIDIA Developer Forum. Set III: PPTs, Sample Code, Handouts in the Class github.

Course Materials

Instructor's teaching materials and online resources.

1. Professor's git: <https://github.com/hualili/CMPE244>
2. Jetson NANO Jetpack download <https://developer.nvidia.com/embedded/downloads>

Other Equipment / Material

1. Hardware Equipment: You may choose any one of the following options. For detailed selection information, I will cover it in the introduction session of the class. Option 1. Nvidia Jetson NANO Board with minimum 2 GB RAM; or Option 2. Pie 3B+, or Pie 4; Option 3: Nvidia Jetson Tx2 developer kit; or Option 4: LPC1769 CPU Module: https://www.mouser.com/NXP-Semiconductors/Embedded-Solutions/Engineering-Tools/Embedded-Processor-Development-Kits/Development-Boards-Kits-ARM/_/N-cxd2t?P=1z0jm4m&Keyword=LPC1769&FS=True&gclid=Cj0KCQjwqKuKBhCgARIsACf4XuHyN8WfqTQ24WGtoMdKd6n-kl7c-YNz-r1hTcPt0ErdZN62jrMQmgaAtXZEALw_wcB or Option 5: Samsung ARM11 developer platform.
2. Linux Host Machine (Ubuntu 18.04).

2021F-114-handout-gpi... Add files via upload

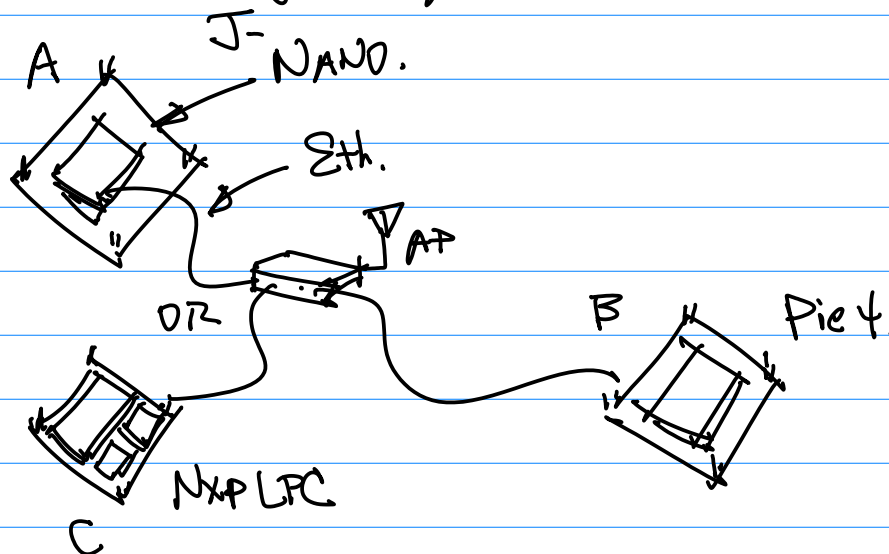
2021F-114b-pwm-nano... Add files via upload

yr. Semester ID

Naming Convention:

A & B
A & C

Note: Regarding The Selection of A Target Platform:



5. Grading Policy

Project Assignment (Two Projects) ^{Phased}
15% (pts) for the assigned projects.
15% for the Semester Long Project.

Assignments and projects:	30%
Midterm Exam:	30%
Final Exam:	40%
Total:	100%

August 23rd (Wed)

Introduction

Note: Rm 268

Ref: Datasheets.

C	A	D	B.
bcm	lpc	nvda	sam
Broadcom Pi Linux O.S.	NXP LPC1769 RTOS IP Stack Micro Web Server	Jetson NANO. JetPack O.S. Linux (Ubuntu) + Additional Packages.	Samsung ARM II

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

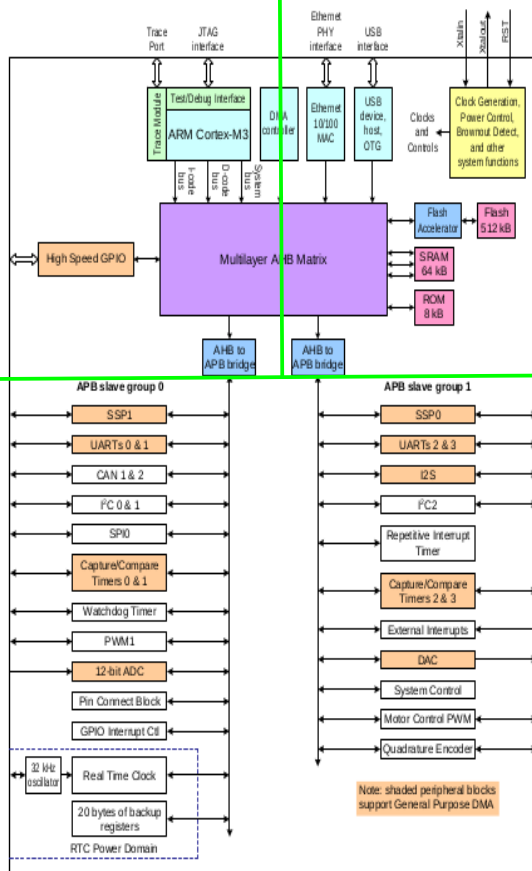


Fig 1. LPC1768 simplified block diagram

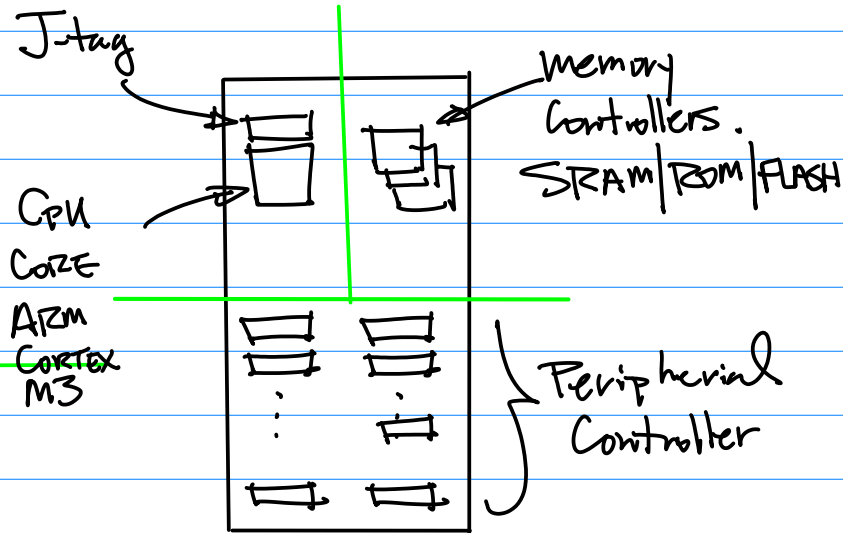


Fig. 1

Note 1: The GPU Block Diagram for LPC1769 is a Sample for the Rest of the target platforms, e.g. Pre3/4; Sam's ARM 11; NVIDIA Jetson NANO

Note 2:

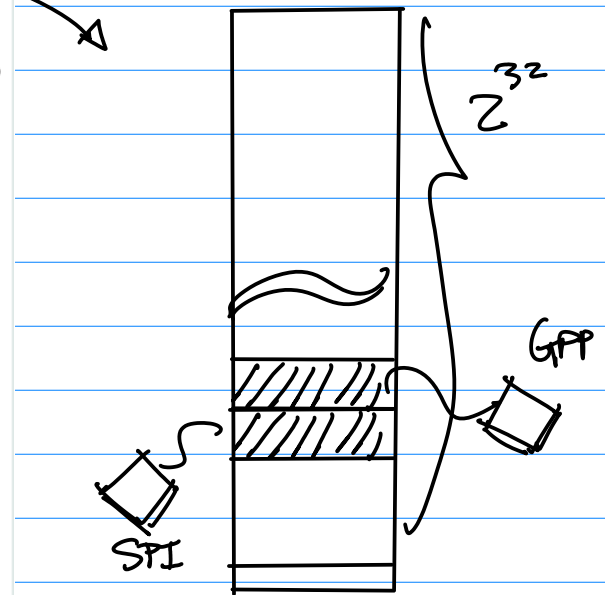
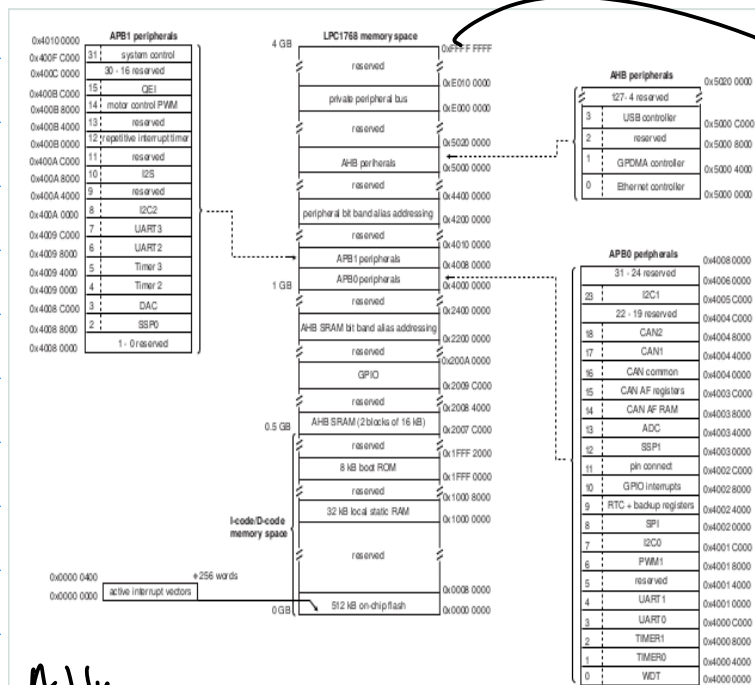


Fig. 2

0x0000-0000 PWR-up Addr.

Addr.
 $2^{32} = 2^{10} \cdot 2^{10} \cdot 2^{10} \cdot 2^2$
 1024 : : 4
 1K : :
 1Meg. :
 1G.
 4 G Addresses.

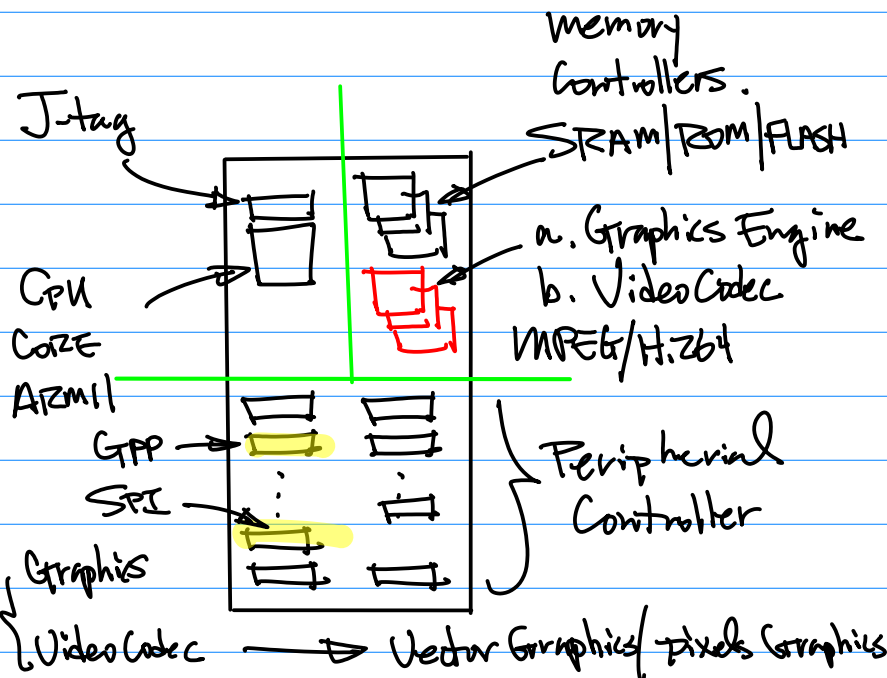
Example: "B", Sam's CPU
 ARM11

→ Datasheets.

2021F-105-#0-cpu-arm11-
 2018S-29-CPU_53C6410X.
 pdf

Locate the page with the top level
 Description of the CPU Architecture

Fig. 3.
 Differentiation



Vector Graphics / pixels Graphics

OMPE244
F2023

61

Example: Connection to (Embedded)
Software Architecture

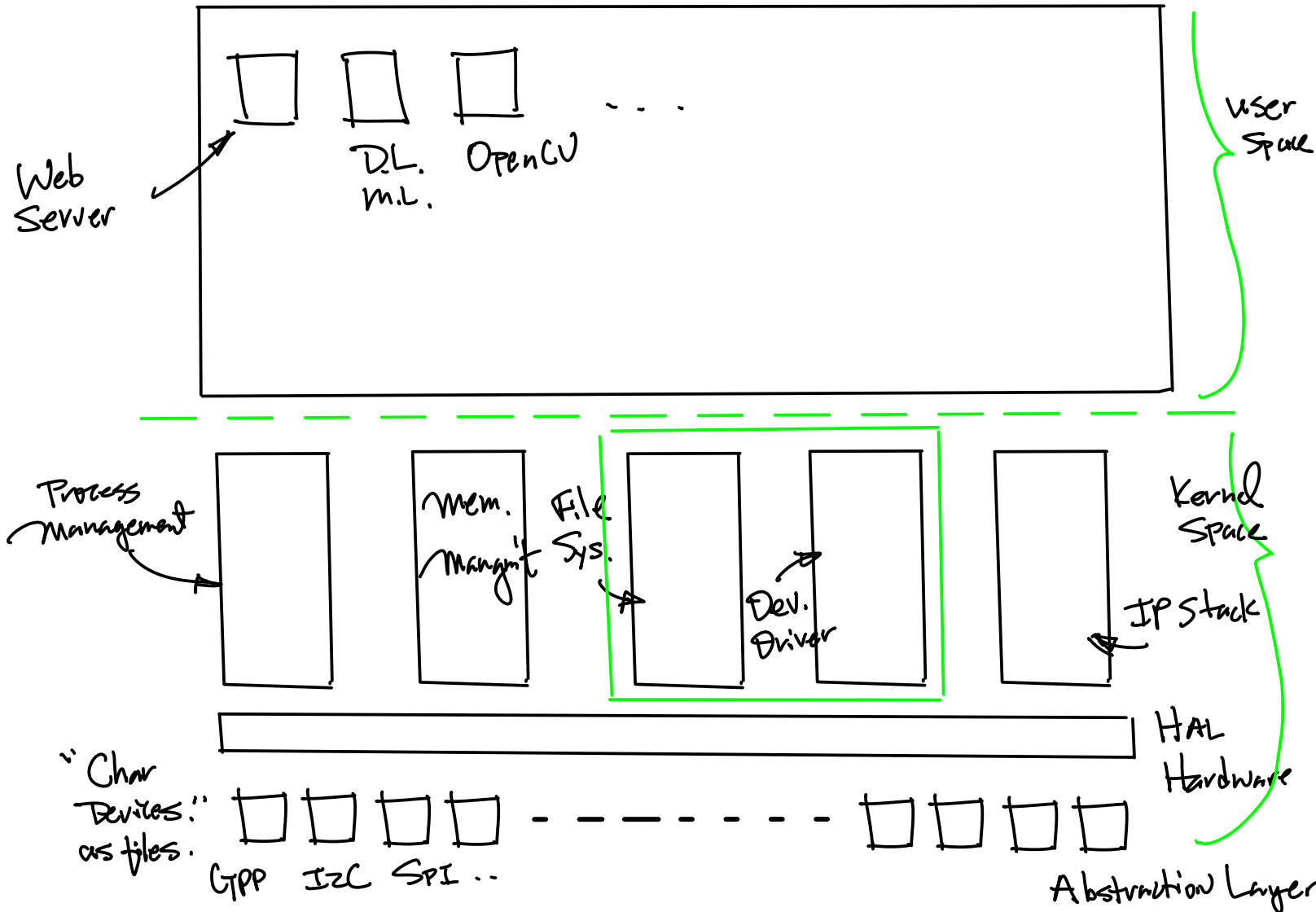


Fig.1.

Note: Data Size for 1080P
Image 0.2 720P

August 28 (Monday)

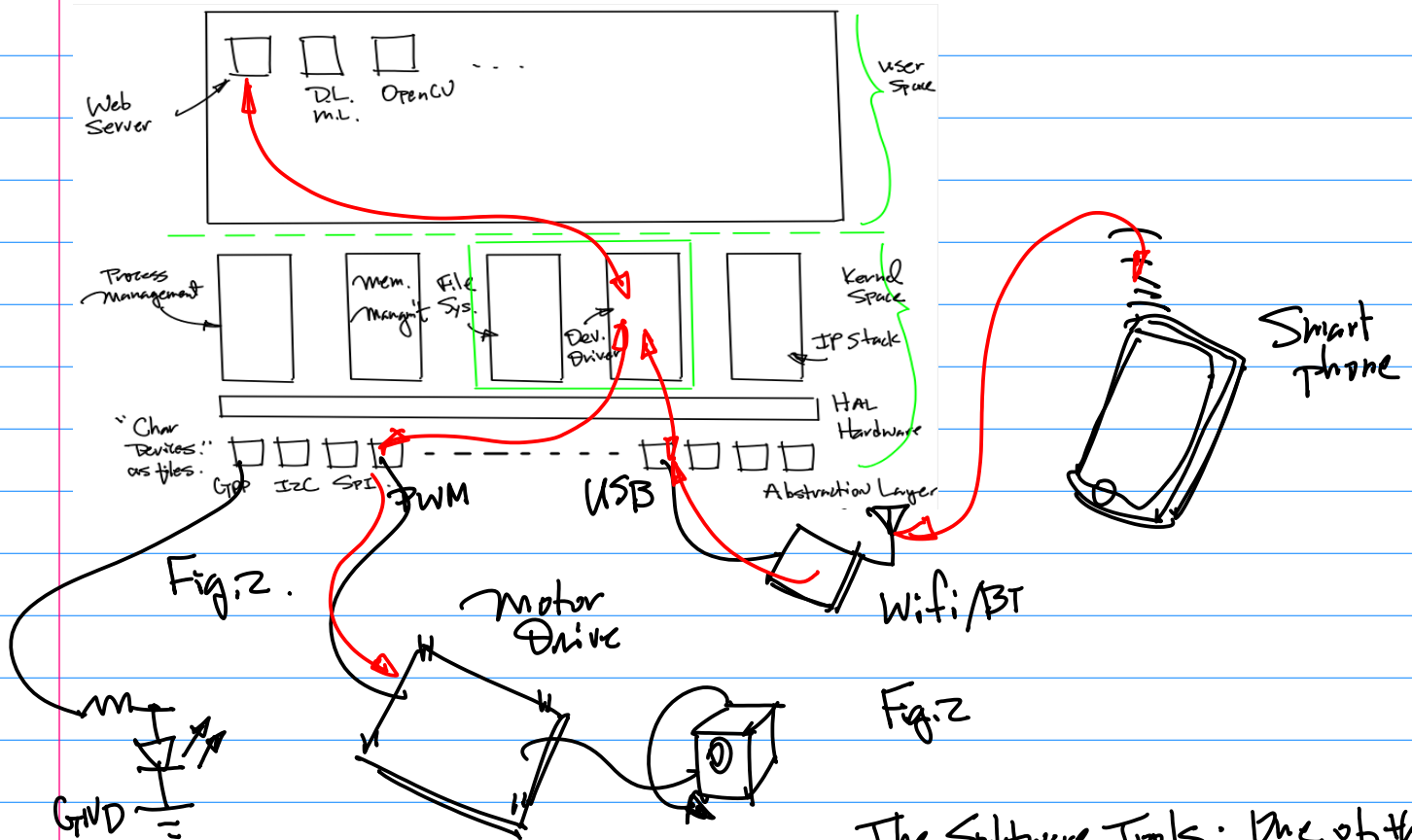
Note: 1st Brief Description on
the Scope of Semester-Long
Project.

Embedded Software; Kernel v.s.
Device Driver → APPS for iPhone/
Android phone

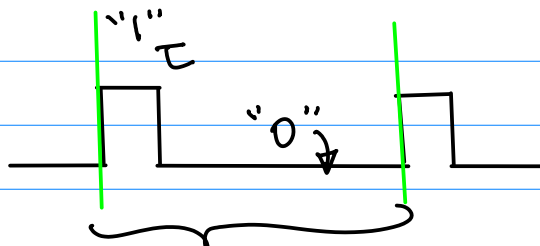
2nd CANVAS is up.
Honesty pledge

3rd Target platform → Minor upgrade
to Enable RTC By Adding On-Board
Battery

Example: Continuation of the Introduction/Embedded Software Architecture.



Note: PWM — Pulse Width Modulation.



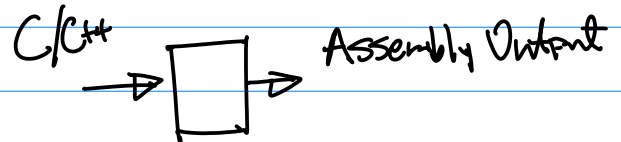
T One Period

$$\text{Duty Cycle} = \frac{\tau}{T} \dots (1)$$

$$f_{\text{PWM}} \dots (2)$$

Fig. 2

The Software Tools: One of them is open source gcc, or g++ Compiler.



Porting. Match to the CORE (ISA: Instruction Set Architecture) Device Drivers Customization.

Peripheral Controller
A Set of Special Purpose
Registers.

Most Likely this SPR has
the Addr in the Block.

151
0x2009-C000

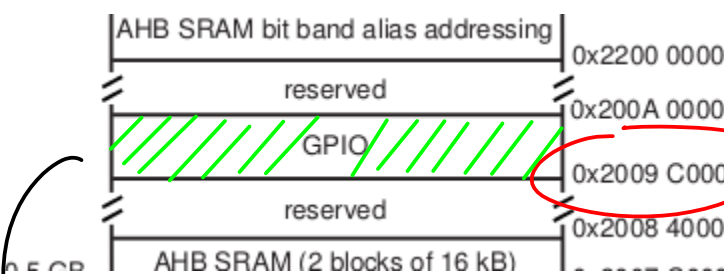
Identify A peripheral controller, GPP

Then, make a GPP as

- ① Output Port
- ② Turn on "LED"
- ③ Turn off the "LED"

See Fig. 3.

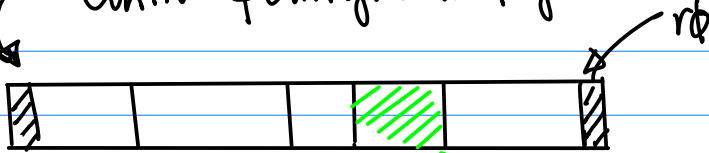
Fig. 3.



Memory is designated for
SPR's (Special Purpose Registers)

Go to 32Bit Control/Configuration
Register. To Define Init & Conf.
for GPP output.

Control & Configuration Register



For Selection of Port &
Pin(s).

4 G Possible Configurations Theoretically
($2^{32} = 2^7 \cdot 2^{10} \cdot 2^{10} \cdot 2^{10}$)

Coding (Software Aspect)

Write 32 bits unsigned Data
as Init & Config Pattern to
Select the GPP & the pin
as output.

0x2009-C000
GPPx CON

1K 1K 1K
1M

1 G.

It has its unique Address. (at
the multiple of 4).

Next. Naming Convention.

Guideline:

RTSC → UC Berkeley David
Patterson
Stanford, John. Hennessy.


```
#define GPDCN
GPDCAT
```

CMPE244
F2023

10)

Porting Porting
gcc/g++ → ARM → CORTEX
Porting ↓
Board

Sept 6 (Wed).

Note: 1st Target Board
Inspection:

Purpose: J41 Connector

RTC Battery
Ref: ON the github. 2021F-114 ~

2021F-114-gpio-nano-v2-hl-2021-10-20.pdf

Harry Li, Ph.D.

NVIDIA Jetson Nano J41 Header Pinout

<https://www.jetsonhacks.com/nvidia-jetson-nano-j41-header-pinout/>

Note: I2C and UART pins are connected to hardware and should not be reassigned. By default, all other pins (except power) are assigned as GPIO. Pins labeled with other functions are recommended functions if using a different device tree.

1. take Pin 1 Vcc (3.3V) and Pin 39 GND to test out LED, make sure you can light up a LED with 220 Ohm resistor in series.

Sysfs GPIO	Name	Pin	Pin	Name	Sysfs GPIO
	3.3 VDC Power	1	2	5.0 VDC Power	
	I2C_2_SDA I2C Bus 1	3	4	5.0 VDC Power	
	I2C_2_SCL I2C Bus 1	5	6	GND	
gpio216	AUDIO_MCLK	7	8	UART_2_TX /dev/ttyTHS1	
	GND	9	10	UART_2_RX /dev/ttyTHS1	
gpio50	UART_2_RTS	11	12	I2S_4_SCLK	gpio79
gpio14	SPI_2_SCK	13	14	GND	
gpio194	LCD_TE	15	16	SPI_2_CS1	gpio232
	3.3 VDC Power	17	18	SPI_2_CS0	gpio15
gpio16	SPI_1_MOSI	19	20	GND	
gpio17	SPI_1_MISO	21	22	SPI_2_MISO	gpio13
gpio18	SPI_1_SCK	23	24	SPI_1_CS0	gpio19
	GND	25	26	SPI_1_CS1	gpio20

Sysfs GPIO	Name	Pin	Pin	Name	Sysfs GPIO
	GND	25	26	SPI_1_CS1	gpio20
	I2C_1_SDA I2C Bus 0	27	28	I2C_1_SCL I2C Bus 0	
gpio149	CAM_AF_EN	29	30	GND	
gpio200	GPIO_P20	31	32	LCD_BL_PWM	gpio168
gpio38	GPIO_PE6	33	34	GND	
gpio76	I2S_4_LRCK	35	36	UART_2_CTS	gpio51
gpio12	SPI_2_MOSI	37	38	I2S_4_SDIN	gpio77
	GND	39	40	I2S_4_SDOUT	gpio78

Harry Li, Ph.D.

Note 1st Power Pins

GND: 6/9/25/39
V_{out} 3.3VDC/5VDC
Pin 1 Pin 2, 4.
V_{in}: J25 (5A or higher @ 5VDC)

Note 2nd GPIO

J41 Pins

J41-12

J41-40

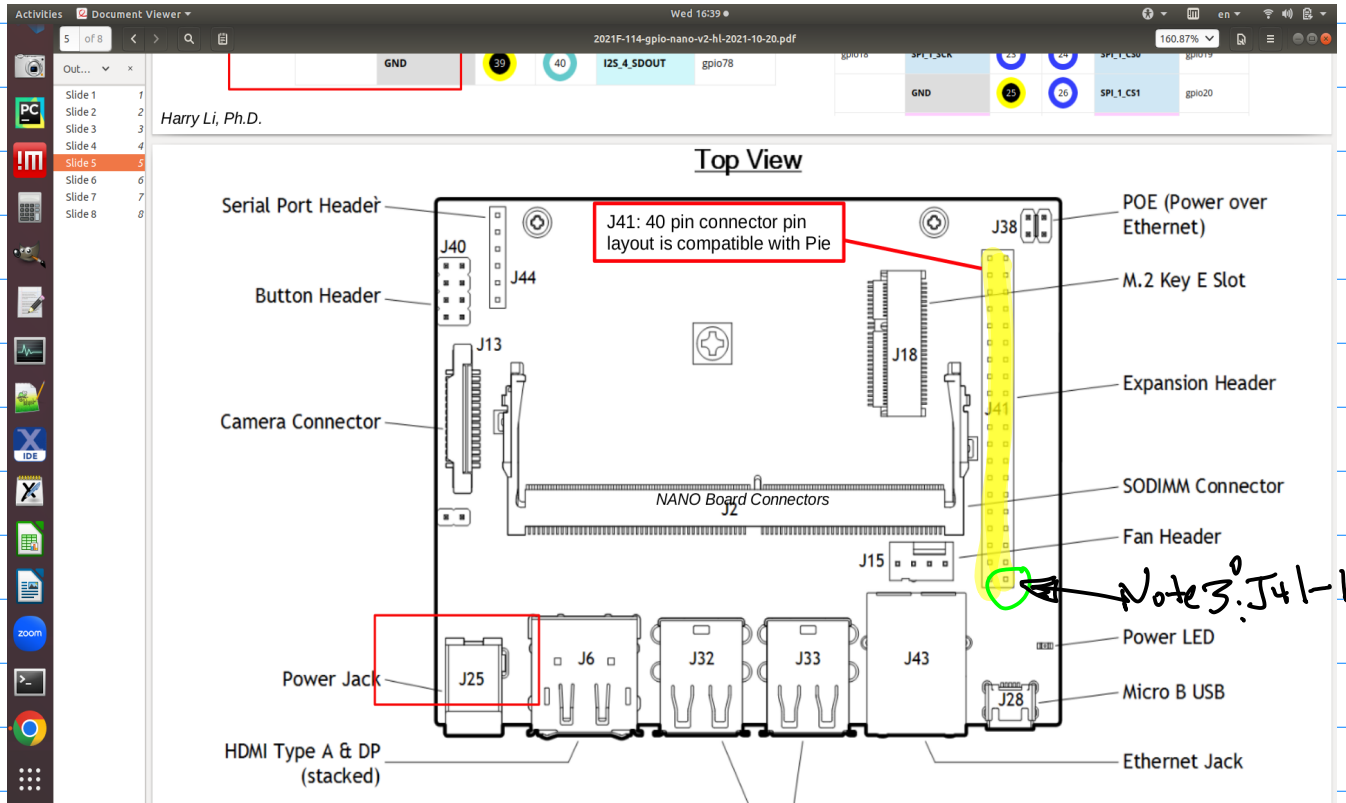
CPU Functionality

gpio79

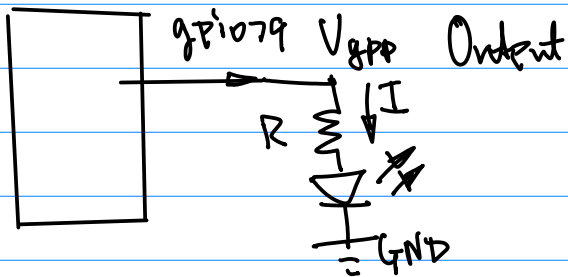
gpio78

CMPE258
F2023

11/



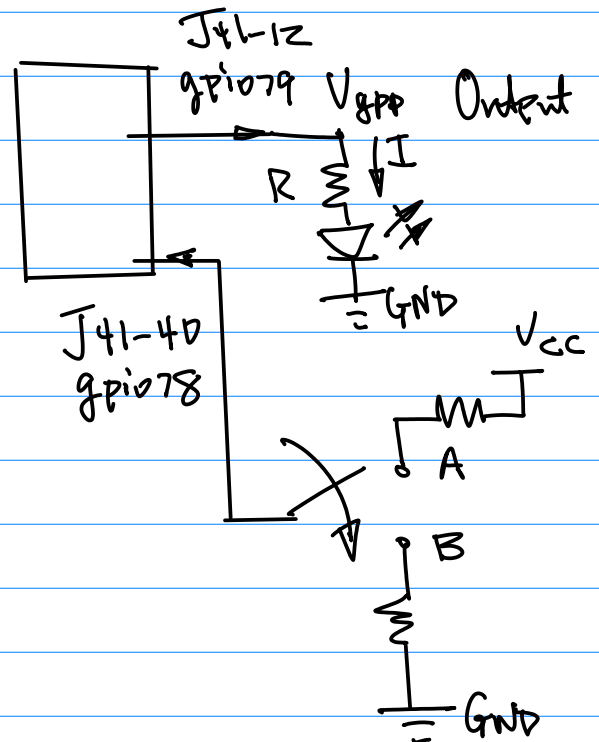
Note 4^o: GPIO Input/Output Testing CKT
Build the following Testing CKT.



Let $I = 4\text{mA}$, $V_{LED} \approx 1.8\text{V}$

$V_{gpp.H} = IR + V_{LED} \dots (1)$

W/o Resistor with Proper Selected LED.

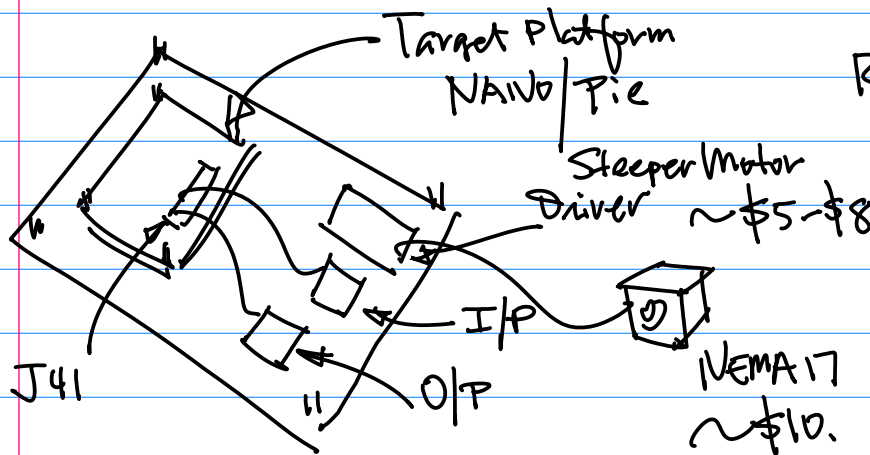


GPIO J41 Pins	CPU Functionality	Note
J41-12	gpio79	Output
J41-40	gpio78	Input

Code { User Space Code
Kernel Space Code

Take a Reference Design from ARM || Samsung CPU.

Ref: Sample code has been posted on the github.



```
harry@harry-laptop:/opt/FriendlyARM/min
2022s-104d-userSpace-gpio.c led led.c
harry@harry-laptop:/opt/FriendlyARM/min
```

Note: Form 2-person Team for A Semester Long Project.

Scoop: Hardware Layer
(Sensors/Actuators)
↓
"Security"
Device Driver/Kernel Space
↓
Process Management

Web Server

Smart phone APPs.

Check GPT 3.5 API + Python Interface

Example: Sample Code for GPP Device Driver

CMPE242-Embedded-Systems- / 2022S / 2022S-104d-userSpace-gpio.c

hualili Add files via upload

Code Blame 37 lines (30 loc) · 642 Bytes Code 55% faster with Gi

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <unistd.h>
4 #include <sys/ioctl.h>
5 #include <sys/types.h>
6 #include <sys/stat.h>
7 #include <fcntl.h>
```

Sept. 11 (Monday)

Note 1. Homework, Due in 1 1/2 Weeks.

To Be posted on Line today;

a. Bring up the target, Screen Capture with Personal Identifier.

b. GPIO Testing (Python + Hardware Circuit). (1) Input Testing CKT.

Output Testing CKT. (2) Coding:

Python.

Note 2: B.o.M. (Bill of Material)
for the class/Project

a. Motor (1) Stepper motor. NEMA. 17



Nema17
Stepper Motor
\$8.99
Amazon.com

(2) BLDC
Brushless DC
motor

c. Smartphone | iPhone

OR Android Phone

Swift Mac OS. as Development
platform.

Ubuntu Linux.
18.04.

NVDA JotPack (O.S.)

Note:

With gpio Testing
Ckt.

(Work-In-
Progress)

Target
Board

Prototype
Board.

NVDA Jetson Nano Bread Board/

OR Wirewrapping
Board.



350W Brushed
Electric Motor



10 Inch Hub
Motor 1000w, ...



48V 500W
Wheel Motor ...

b. Motor Drive Unit.



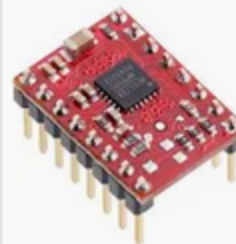
EASON
Stepper Motor
\$9.69
Amazon.com



SparkFun
Electronics ...
\$12.09
DigiKey



WWZMDiB
A4988 Stepper
\$7.99
Amazon.com



Pololu
Corporation
\$8.49
DigiKey



STEPPERONLINE
CNC Stepper Motor
Driver 1.0-4.2A
20-50VDC 1/128

Example: Continuation on Linux D.D.

on ARM-11. (Samsung). Note: UserSpace Code Samples, Kernel Space code

```
harry@harry-laptop:/opt/FriendlyARM/mini6410$ cd linux/
harry@harry-laptop:/opt/FriendlyARM/mini6410/linux$ ls
arm-qte-4.7.0      examples      u-boot-mini6410
arm-qt-extended-4.4.3  linux-2.6.38  x86-qte-4.6.1
arm-qtopia        rootfs_qtopia_qt4  x86-qt-extended-4.4.3
busybox-1.17.2    rootfs_qtopia_qt4-s  x86-qtopia
harry@harry-laptop:/opt/FriendlyARM/mini6410/linux$
```

Ref: Sample Code ON github.

```
harry@harry-laptop:/opt/FriendlyARM/mini6410/linux/examples/leds$ ls
2022s-104d-userSpace-gpio.c led led.c led.c~ Makefile Makefile~
harry@harry-laptop:/opt/FriendlyARM/mini6410/linux/examples/leds$
```

```
22
23 fd = open("/dev/leds0", 0);
24 if (fd < 0) {
25     fd = open("/dev/leds", 0);
26 }
27 if (fd < 0) {
28     perror("open device leds");
29     exit(1);
30 }
31
32 ioctl(fd, on, led_no);
33 close(fd);
34
```

Note 1: "Char" Device. open the Device just like a file.

Kernel (O.S. Image)
path to the Device.
The Device Driver can be either integrated as the whole kernel image OR module (Installed/Removed)

Note 2: for passing control parameter(s) to the Device for Control Action.

Note 3. Close it, when Done !

CMPE244

F2023

15/

Build
Kernel image using
"menuconfig"

Android.

→ NVIDIA, Broadcom, Smart phones
Embedded

```
harry@harry-laptop: /opt/FriendlyARM/mini6410/linux/linux-2.6.38
.config - Linux/arm 2.6.38 Kernel Configuration

Linux/arm 2.6.38 Kernel Configuration
Arrow keys navigate the menu. <Enter> selects submenus --->. Highlighted letters are hotkeys. Pressing
<Y> includes, <N> excludes, <M> modularizes features. Press <Esc><Esc> to exit, <?> for Help, </> for
Search. Legend: [*] built-in [ ] excluded <M> module < > module capable

General setup --->
[*] Enable loadable module support --->
-* Enable the block layer --->
  System Type --->
  Bus support --->
  Kernel Features --->
  Boot options --->
  CPU Power Management --->
  Floating point emulation --->
  Userspace binary formats --->
  Power management options --->
[*] Networking support --->
  Device Drivers --->
  File systems --->
  Kernel hacking --->
  Security options --->
-* Cryptographic API --->
  Library routines --->
  Load an Alternate Configuration File
  Save an Alternate Configuration File

<Select> < Exit > < Help >
```

```
.config - Linux/arm 2.6.38 Kernel Configuration

Device Drivers
Arrow keys navigate the menu. <Enter> selects submenus --->. H
<Y> includes, <N> excludes, <M> modularizes features. Press <Es
Search. Legend: [*] built-in [ ] excluded <M> module < > mod

^(-)
  SCSI device support --->
  < > Serial ATA and Parallel ATA drivers --->
  [ ] Multiple devices driver support (RAID and
  < > Generic Target Core Mod (TCM) and ConfigFS
  [*] Network device support --->
  [ ] ISDN support --->
  < > Telephony support --->
  Input device support --->
  Character devices --->
  <*> I2C support --->
  [ ] SPI support --->
  PPS support --->
  -* GPIO Support --->
  < > Dallas's 1-wire support --->
  < > Power supply class support --->
  <*> Hardware Monitoring support --->
```

CMPE244

F2023

161

Note 4. menuconfig controls how the Kernel Image is built.
Here, the "Char" Device Driver can be selected/De-selected.
b. Use "Space Bar" to toggle between 3 options.

```

Arrow keys navigate the menu. <Enter> selects submenus --->. Hlt
<Y> includes, <N> excludes, <M> modularizes features. Press <Esc>
Search. Legend: [*] built-in [ ] excluded <M> module < > modul

-* Virtual terminal
[ ] Support for binding and unbinding console
[ ] dev/kmem virtual device support
<M> LED Support for Mini6410 GPIO LEDs
<M> Harry 2021-2-3: I2C sensor module
<M> Harry: 2016-Feb-22, CMPE 242 Mini6410 module
<M> Harry: Mini6410 Test module
<M> Harry: Mini6410 PWM2 module
< > Buttons driver for FriendlyARM Mini6410 deve
< > Buzzer driver for FriendlyARM Mini6410 devel
< > ADC driver for FriendlyARM Mini6410 develop
[ ] Non-standard serial port support
< > GSM MUX line discipline support (EXPERIMENTA
Serial drivers --->
-* Unix98 PTY support
[ ] Support multiple instances of devpts
[*] Legacy (BSD) PTY support
(16) Maximum number of legacy PTY in use
[ ] ARM JTAG DCC console
< > TPM1 top-level message handler --->
  
```

(N|b|M|*)
↑
module
↑
Integrated
Kernel Image

Note 5. Folder for the "Char" D.D., The "gpio" (Leds) Device Driver

```

harry@harry-laptop:/opt/FriendlyARM/mini6410/linux/linux-2.6.38/drivers$ cd char
harry@harry-laptop:/opt/FriendlyARM/mini6410/linux/linux-2.6.38/drivers/char$ ls
20-20215-9-mini6410_pwmHarry.c  ip2  misc.c  rtc.c
2q  ipmi  misc.o  scc.l
agp  isicom.c  mmtimer.c  scx20
amiserial.c  installion.c  modules.built.in  ser_
apm-emulation.c  Kconfig  modules.order  ser_
applicom.c  lp.c  moxa.c  ser_
applicom.h  Makefile  moxa.h  ser_
bfin_jtag_comm.c  Makefile-backup  mspec.c  seri
bfin-otp.c  mbcs.c  nwave  snsc
briq_panel.c  mbcs.h  mxser.c  snsc
bsr.c  mem.c  mxser.h  snsc
built-in.o  mem.o  nozomi.c  sonyi
cd1865.h  mini6410_adc.c  nse_gpio.c  spec
cyclades.c  mini6410_adc.mod.c  nvram.c  spec
digi1.h  mini6410_adc.o  nwbutton.c  stal
digiFep1.h  mini6410_buttons.c  nwbutton.h  sxboi
digiPCI.h  mini6410_buttons.o  nwflash.c  sx.c
ds1302.c  mini6410_hello_module.c  pc8736x_gpio.c  sx.h
ds1620.c  mini6410_hello_module.ko  pcmcia  sxwi
dsp56k.c  mini6410_hello_module.mod.c  ppdev.c  sync
dtlk.c  mini6410_hello_module.mod.o  ps3flash.c  sync
efirtc.c  mini6410_hello_module.o  ramoops.c  sync
epca.c  mini6410_leds.c  random.c  tb02
epcaconfig.h  mini6410_leds.ko  random.o  tlcl
  
```

2025-10-4
(CMPE242)

Example: Kernel Space

Device Driver Code Requirements: Code Spec. Connect the Code to the CPU

Note1:

Datasheet

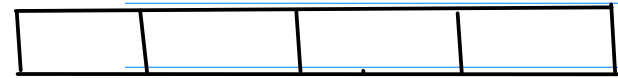
```

66 static int _init dev_init(void)
67 {
68     int ret;
69
70     {
71         unsigned tmp;
72         tmp = readl(S3C64XX_GPECON);
73         tmp = (tmp & ~(0xffffU<<16))|(0x1111U<<16);
74         writel(tmp, S3C64XX_GPECON);
75
76         tmp = readl(S3C64XX_GPEDAT);
77         tmp |= (0xF << 4);
78         writel(tmp, S3C64XX_GPEDAT);
79     }
80
81     ret = misc_register(&misc);
82
83     printk (DEVICE_NAME"\nHarry: PGE initialized\n");
84

```

Note2: Read long 32bit

Naming: ID — Peripheral Control (Manufacturer/Part) GPP



Memory map.

Ref: CPU Datasheet.

2021F-105-#0-cpu-arm11-2018S-29-CPU-S3C6410X.pdf

Note2: TP320.

a) GPE CON

b) Addr. / On the memory.

c) GPE DAT

d. Two Addition types:

Pull up down.

GPEPUD
GPECONSLP
GPEPUDSLP
Power Control

6410X_UM

GPIO

10.5.5 PORT E CONTROL REGISTERS

There are five control registers including GPECON, GPEDAT, GPEPUD, GPECONSLP and GPEPUDSLP in the Port E Control Registers.

Register	Address	R/W	Description	Reset Value
GPECON	0x7F008080	R/W	Port E Configuration Register	0x00
GPEDAT	0x7F008084	R/W	Port E Data Register	Undefined
GPEPUD	0x7F008088	R/W	Port E Pull-up/down Register	0x00000155
GPECONSLP	0x7F00808C	R/W	Port E Sleep mode Configuration Register	0x0
GPEPUDSLP	0x7F008090	R/W	Port E Sleep mode Pull-up/down Register	0x0

GPECON	Bit	Description	Initial State
GPE0	[3:0]	0000 = Input	0001 = Output
		0010 = PCM SCLK[1]	0011 = I2S CLK[1]
		0100 = AC97 BITCLK	0101 = Reserved
		0110 = Reserved	0111 = Reserved
GPE1	[7:4]	0000 = Input	0001 = Output
		0010 = PCM EXTCLK[1]	0011 = I2S CDCLK[1]
		0100 = AC97 RESETn	0101 = Reserved
		0110 = Reserved	0111 = Reserved

Note 3. From the D.T. Code, GPE1 is utilized for the I/O function
 $GPECON[7:4] = 0001$

5
GPIO
pins

GPDCON	Bit	Description		Initial State
GPE0	[3:0]	0000 = Input 0010 = PCM SCLK[1] 0100 = AC97 BITCLK 0110 = Reserved	0001 = Output 0011 = I2S CLK[1] 0101 = Reserved 0111 = Reserved	0000
GPE1	[7:4]	0000 = Input 0010 = PCM EXTCLK[1] 0100 = AC97 RESETn 0110 = Reserved	0001 = Output 0011 = I2S CDCLK[1] 0101 = Reserved 0111 = Reserved	0000
GPE2	[11:8]	0000 = Input 0010 = PCM FSYNC[1] 0100 = AC97 SYNC 0110 = Reserved	0001 = Output 0011 = I2S LRCLK[1] 0101 = Reserved 0111 = Reserved	0000
GPE3	[15:12]	0000 = Input 0010 = PCM SIN[1] 0100 = AC97 SDI 0110 = Reserved	0001 = Output 0011 = I2S DI[1] 0101 = Reserved 0111 = Reserved	0000
GPE4	[19:16]	0000 = Input 0010 = PCM SOUT[1] 0100 = AC97 SDO 0110 = Reserved	0001 = Output 0011 = I2S DO[1] 0101 = Reserved 0111 = Reserved	0000

How to

Question: Make GPE1 as an output pin,
 But keep the Rest unchanged?

Sept. 18 (Monday)

Note: 1st Homework Posted on the
 CANVAS.

2nd Homework, Due Sept. 27

(Wed), PWM Testing.

(1) Enable PWM Device
 Driver Via Driver mapping
 utility By NDA

(2) Test PWM Output By
 Changing f_{PWM} from 2 KHz
 to 50 Hz; By changing

Duty Cycle from 5% to 90%.

Then, Observe its output

Note: Output with 2222 npn Transistor
 to drive a LED. (see the
 details in the Class PPT on
 github). (O.S.)

Note: Prepare the Source Distribution
 Download, to build Kernel
 Image from the distribution.
 (1 ~ 1½ week)

Note: Ref. on Smartphone Apps.
 Android APPs Development.

CMPE244
F2023.

191

Note: Device Driver Sample Code

- 2023F-104-gpio-command-line-202...
- 2023F-105a-2022s-104d-userSpace...
- 2023F-105b-mini6410_leds.mod.c
- 2023F-106a-CMPE244-#190h-1d-A...

Apps.

Note: Install Android Studio
on your Laptop.

2023F-106a-CMPE244-#190h-1d-Android-Test-Environment-v2-YY-2023-9-18.pdf

Install Android Studio on Ubuntu 18.04 (9/15-8/3/2023)

nnn-n-Android-Development-HelloWorld-Calculator-v2-XW-2023-7-7.odt

1. Check Java version

```
nicole@nicole-G565:~$ java --version
openjdk 11.0.19 2023-04-18
OpenJDK Runtime Environment (build 11.0.19+7-post-Ubuntu-0ubuntu118.04.1)
OpenJDK 64-Bit Server VM (build 11.0.19+7-post-Ubuntu-0ubuntu118.04.1, mixed mode, sharing)
nicole@nicole-G565:~$
```

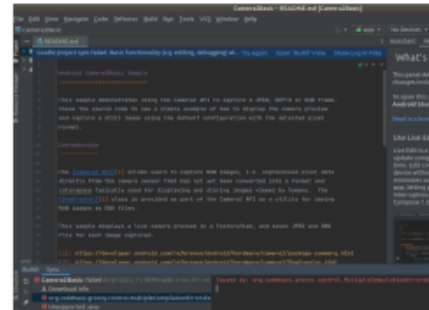
2. Install Android Studio using snap

```
$ snap find "android-studio"
$ sudo snap install android-studio --classic
```

```
nicole@nicole-G565:~$ snap find "android-studio"
Name          Version    Publisher    Notes    Summary
android-studio 2022.2.1.19 snapcrafters classic The IDE for Android
android-studio-canary 2022.1.1.7 snapcrafters classic The IDE for Android
(Canary build)
nicole@nicole-G565:~$ sudo snap install android-studio --classic
```

```
(base) harry@harrys-gpu-laptop:~$ sudo snap install android-studio --classic
[sudo] password for harry:
android-studio 2022.3.1.18 from Snapcrafters installed
```

3. Once installed, double click the icon to start the studio



Note: Team Project (1) Formation of
The Team ; (2) Selection of Smart; home
for the APP. (3) Stepper motor
Drive Board and Stepper motor.

Example: GPIO & PWM.

2023F-104

GPIO Command Line

<https://jetsonhacks.com/2019/06/07/jetson-nano-gpio/>

```
# Map GPIO Pin
$ gpio79 is pin 12 on the Jetson Nano
$ echo 79 > /sys/class/gpio/export
# Set Direction
$ echo out > /sys/class/gpio/gpio79/direction
# Bit Bangin'!
$ echo 1 > /sys/class/gpio/gpio79/value
$ echo 0 > /sys/class/gpio/gpio79/value
# Unmap GPIO Pin
$ echo 79 > /sys/class/gpio/unexport
# Query Status
$ cat /sys/kernel/debug/gpio
```

PWM

2021F-1146-~

Note: Establish Remote
Access to your target,
e.g. Laptop to Access
your target Board.

→ Next Monday, Show+Tell
in class