

Tinker board

IoT를 위한 최선의 엣지 단말기



MAKER SPACE
G·CAMP

Contents

- Debian 기본 설정
- GPIO LED 제어
- GPIO SERVO MOTOR 제어
- Camera 제어

Debian Tinker 기본 설정(Wi-Fi)



1. Tinker Board 에서 Wi-Fi 모듈과 안테나 연결
(main만 연결해도 지장 없음)

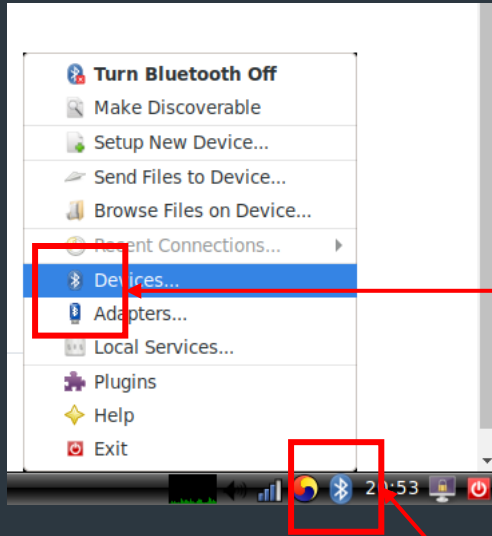
Debian Tinker 기본 설정(Wi-Fi)



3. 자신이 원하는 Wi-Fi 선택 및 암호 입력 후 연결 상태 확인

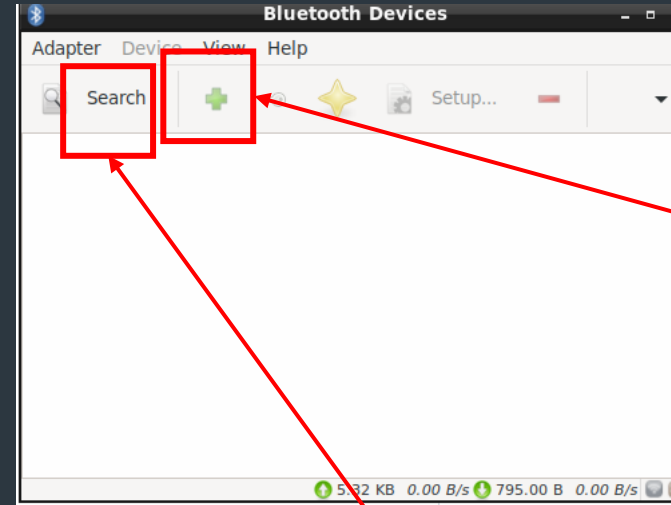
2. Wi-Fi 아이콘 더블 클릭
(이더넷 연결 시 별도의 설정 필요 없음)

Debian Tinker 기본 설정(blueetooth)



2. Device 아이콘 더블 클릭

1. bluetooth 아이콘 더블 클릭



4. +아이콘 클릭 후 자신의 기기와 pairing

3. Search 아이콘 더블 클릭 후 자신이 찾는 디바이스 찾기

Debian Tinker 기본 설정(Git)

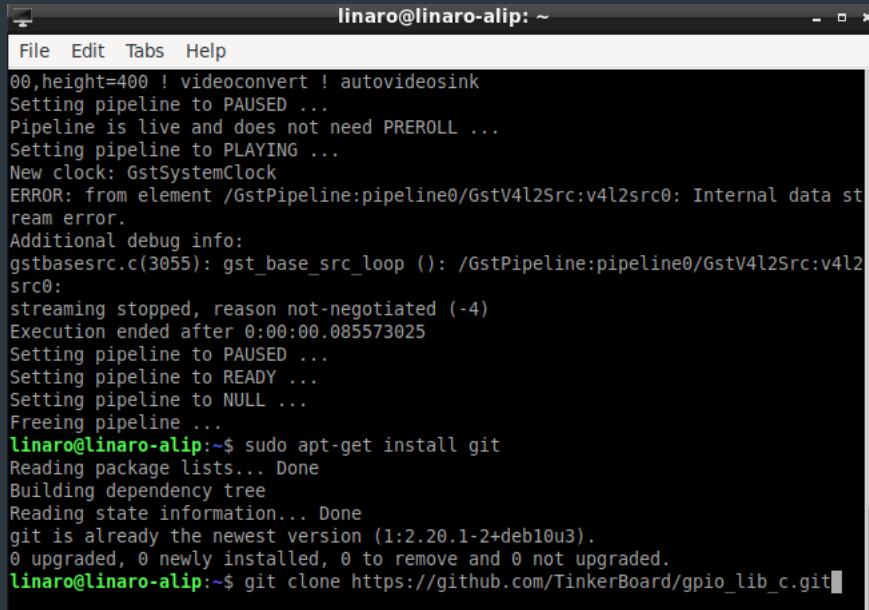
- GPIO를 사용 하고 GIT에서 Clone을 가져오기 위해 Git 설치

1. 하단 Bar에서 System Tools - LXTerminal 클릭
2. Sudo apt-get install git 입력 후 엔터

```
linaro@linaro-alip: ~  
File Edit Tabs Help  
Setting pipeline to PAUSED ...  
Setting pipeline to READY ...  
Setting pipeline to NULL ...  
Freeing pipeline ...  
linaro@linaro-alip:~$  
linaro@linaro-alip:~$ gst-launch-1.0 v4l2src! videoconvert ! video/x-raw,width=5  
90,height=400 ! videoconvert ! autovideosink  
Setting pipeline to PAUSED ...  
Pipeline is live and does not need PREROLL ...  
Setting pipeline to PLAYING ...  
New clock: GstSystemClock  
ERROR: from element /GstPipeline:pipeline0/GstV4l2Src:v4l2src0: Internal data st  
ream error.  
Additional debug info:  
gstbasesrc.c(3055): gst_base_src_loop (): /GstPipeline:pipeline0/GstV4l2Src:v4l2  
src0:  
streaming stopped, reason not-negotiated (-4)  
Execution ended after 0:00:00.085573025  
Setting pipeline to PAUSED ...  
Setting pipeline to READY ...  
Setting pipeline to NULL ...  
Freeing pipeline ...  
linaro@linaro-alip:~$ sudo apt-get install git
```

Debian Tinker 기본 설정(Git Clone)

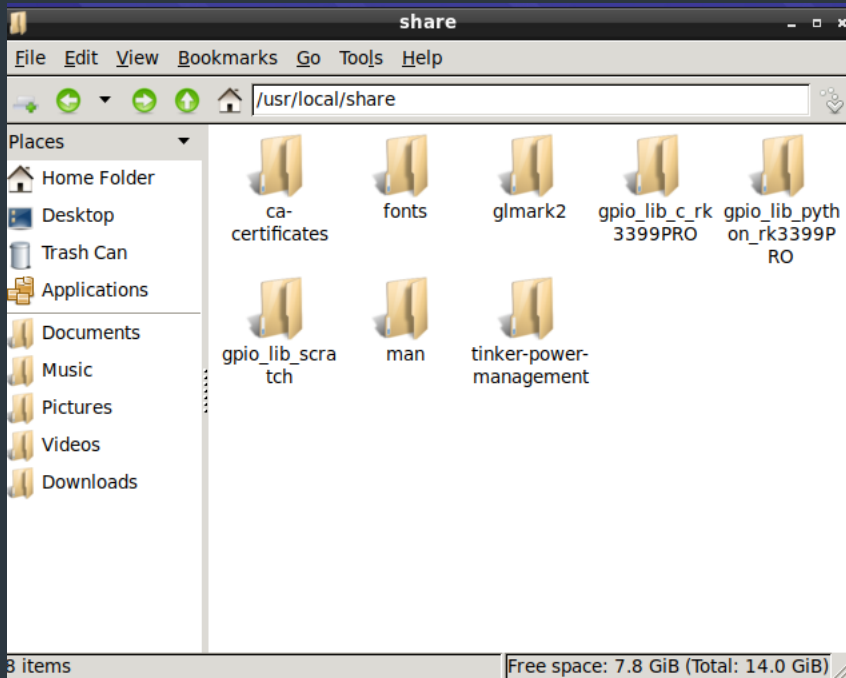
1. Sudo apt-get install git 로 Git 설치 후 진행 가능
2. git clone https://github.com/TinkerBoard/gpio_lib_python.git 입력 후 엔터



```
linaro@linaro-alip: ~  
File Edit Tabs Help  
00,height=400 ! videoconvert ! autovideosink  
Setting pipeline to PAUSED ...  
Pipeline is live and does not need PREROLL ...  
Setting pipeline to PLAYING ...  
New clock: GstSystemClock  
ERROR: from element /GstPipeline:pipeline0/GstV4l2Src:v4l2src0: Internal data stream error.  
Additional debug info:  
gstbasesrc.c(3055): gst_base_src_loop (): /GstPipeline:pipeline0/GstV4l2Src:v4l2src0:  
streaming stopped, reason not-negotiated (-4)  
Execution ended after 0:00:00.085573025  
Setting pipeline to PAUSED ...  
Setting pipeline to READY ...  
Setting pipeline to NULL ...  
Freeing pipeline ...  
linaro@linaro-alip:~$ sudo apt-get install git  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
git is already the newest version (1:2.20.1-2+deb10u3).  
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.  
linaro@linaro-alip:~$ git clone https://github.com/TinkerBoard/gpio_lib_c.git
```

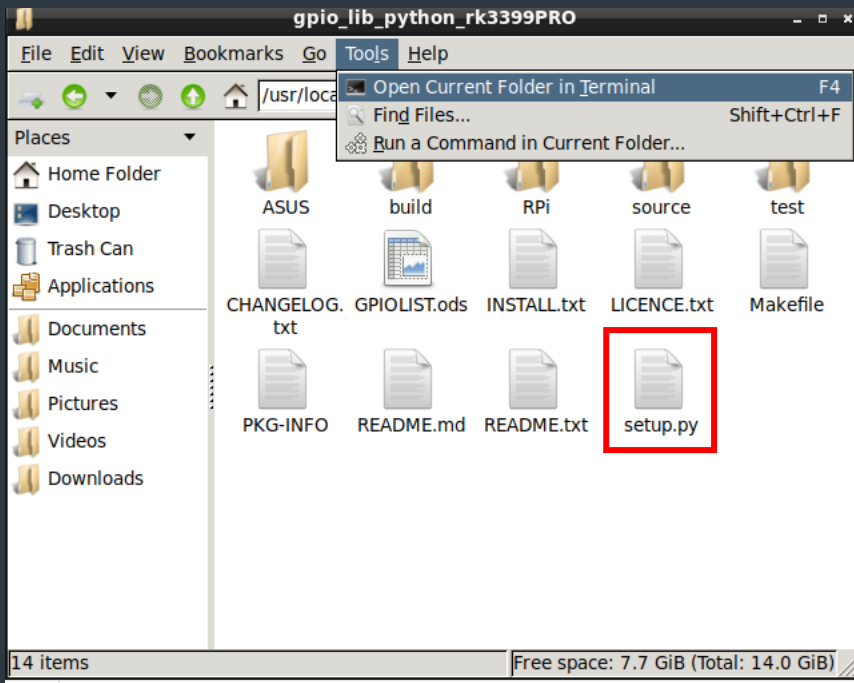
Debian Tinker 기본 설정(Git Clone)

3. 정상적으로 설치가 완료되면 /usr/local/share 경로에 아래 이미지와 같은 하위 폴더가 생성됨



Debian Tinker 기본 설정(Git Clone build)

1. 경로 지정의 편의를 위해 /usr/local/share/gpio_lib_python_rk3399pro 폴더 진입
2. Setup.py 파일 확인
3. Tools-Open Current Folder in Terminal 클릭



Debian Tinker 기본 설정(Git Clone build)

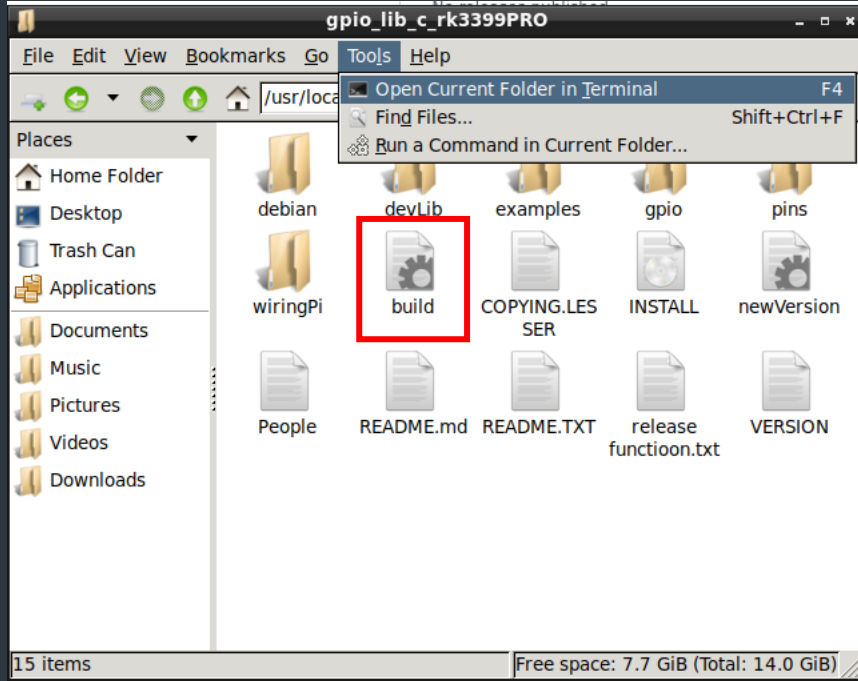
1. Terminal이 gpio_lib_python_rk3399pro로 디렉토리가 설정 되었는지 확인
2. sudo python3 setup.py install 입력

```
linaro@linaro-alip: /usr...lib_python_rk3399PRO - □ ▸
or: setup.py --help [cmd1 cmd2 ...]
or: setup.py --help-commands
or: setup.py cmd --help

error: no commands supplied
linaro@linaro-alip:/usr/local/share/gpio_lib_python_rk3399PRO$ sudo python3 set
p.py install
running install
running build
running build_py
running build_ext
running install_lib
running install_egg_info
Removing /usr/local/lib/python3.7/dist-packages/ASUS.GPIO-0.1.egg-info
Writing /usr/local/lib/python3.7/dist-packages/ASUS.GPIO-0.1.egg-info
running install
running build
running build_py
running build_ext
running install_lib
running install_egg_info
Removing /usr/local/lib/python3.7/dist-packages/RPi.GPIO-0.1.egg-info
Writing /usr/local/lib/python3.7/dist-packages/RPi.GPIO-0.1.egg-info
linaro@linaro-alip:/usr/local/share/gpio_lib_python_rk3399PRO$
```

Debian에서 Tinker board pin map 확인하기

1. 경로 지정의 편의를 위해 /usr/local/share/gpio_lib_c_rk3399pro 폴더 진입
2. build 파일 확인
3. Tools-Open Current Folder in Terminal 클릭



Debian에서 Tinker board pin map 확인하기

1. Terminal이 gpio_lib_c_rk3399pro로 디렉토리가 설정 되었는지 확인
2. sudo ./build 입력

```
linaro@linaro-alip: /usr.../gpio_lib_c_rk3399PRO - □ ×  
  
ldconfig: file /usr/lib/mali/libmali-midgard-t86x-r18p0-x11.so is truncated  
ldconfig: file /usr/lib/mali/libmali.so.1 is truncated  
ldconfig: file /usr/lib/mali/libmali.so.1.9.0 is truncated  
ldconfig: file /usr/lib/libmali.so.1 is truncated  
  
GPIO Utility  
make: Nothing to be done for 'all'.  
[Install]  
  
All Done.  
  
NOTE: To compile programs with wiringPi, you need to add:  
-lwiringPi  
to your compile line(s) To use the Gertboard, MaxDetect, etc,  
code (the devLib), you need to also add:  
-lwiringPiDev  
to your compile line(s).  
  
linaro@linaro-alip:/usr/local/share/gpio_lib_c_rk3399PRO$ sudo ./build
```

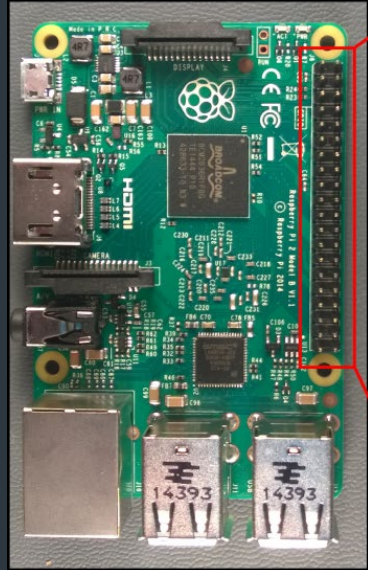
Debian에서 Tinker board pin map 확인하기

1. Sudo ./build 후 gpio readall 입력
2. 현재 확인되는 핀의 종류, 명칭, Gpio wPi 확인 가능

```
linaro@linaro-alip: /usr.../gpio_lib_c_rk3399PRO - □ x
| 73 | 8 | 3.3v | IN | 1 | 1 | 2 | | | 5v | | |
| 74 | 9 | GPIO2B2 | IN | 1 | 5 | 6 | | | 5v | | |
| 89 | 7 | GPIO2D1 | IN | 1 | 7 | 8 | 1 | SERL | TxD,0 | 15 | 81 |
| | | 0v | | | 9 | 10 | 1 | SERL | RxD,0 | 16 | 80 |
| 83 | 0 | RTSN,0 | SERL | 1 | 11 | 12 | 0 | IN | GPIO3D0 | 1 | 120 |
| 85 | 2 | GPIO2C5 | IN | 0 | 13 | 14 | | | 0v | | |
| 84 | 3 | GPIO2C4 | IN | 0 | 15 | 16 | 0 | IN | GPIO2C6 | 4 | 86 |
| | | 3.3v | | | 17 | 18 | 0 | IN | GPIO2C7 | 5 | 87 |
| 40 | 12 | GPIO1B0 | IN | 0 | 19 | 20 | | | 0v | | |
| 39 | 13 | GPIO1A7 | IN | 0 | 21 | 22 | 0 | IN | GPIO3D4 | 6 | 124 |
| 41 | 14 | GPIO1B1 | IN | 1 | 23 | 24 | 1 | IN | GPIO1B2 | 10 | 42 |
| | | 0v | | | 25 | 26 | 0 | IN | GPIO0A6 | 11 | 6 |
| 71 | 30 | GPIO2A7 | IN | 1 | 27 | 28 | 1 | IN | GPIO2B0 | 31 | 72 |
| 126 | 21 | GPIO3D6 | IN | 0 | 29 | 30 | | | 0v | | |
| 125 | 22 | GPIO3D5 | IN | 0 | 31 | 32 | 0 | IN | GPIO4C2 | 26 | 146 |
| 150 | 23 | GPIO4C6 | IN | 0 | 33 | 34 | | | 0v | | |
| 121 | 24 | GPIO3D1 | IN | 0 | 35 | 36 | 1 | IN | GPIO2C2 | 27 | 82 |
| 149 | 25 | GPIO4C5 | IN | 0 | 37 | 38 | 0 | IN | GPIO3D3 | 28 | 123 |
| | | 0v | | | 39 | 40 | 0 | IN | GPIO3D7 | 29 | 127 |
+-----+-----+-----+-----+-----+-----+-----+-----+
| CPU | wPi | Name | Mode | V | Physical | V | Mode | Name | wPi | CPU |
+-----+-----+-----+-----+-----+-----+-----+-----+
| | | | | | Tinker | | | | | |
linaro@linaro-alip: /usr/local/share/gpio_lib_c_rk3399PRO$ gpio readall
```

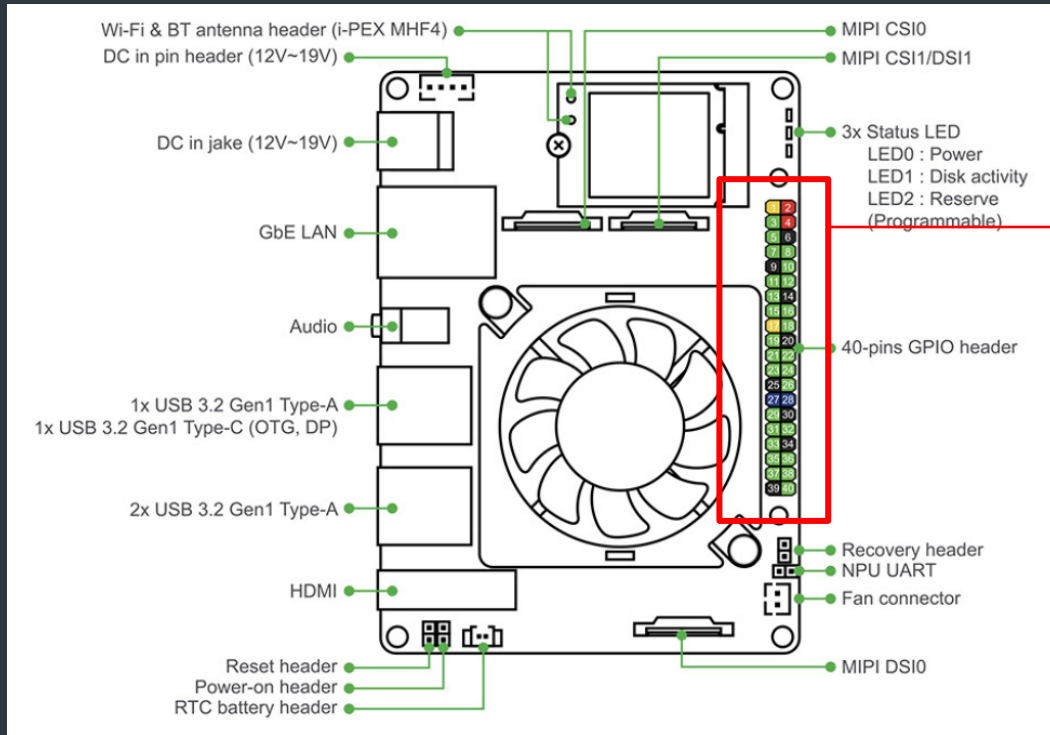
GPIO

- GPIO (general-purpose input/output) : 다용도 입출력이 가능한 입력과 출력을 포함한 동작이 사용자에게 의해 제어될 수 있는, 집적 회로나 전기 회로 기판의 디지털 신호 핀.
- 대표적인 raspberry pi GPIO PIN MAP



Alternate Function					Alternate Function
	3.3V PWR	1		2	5V PWR
I2C1 SDA	GPIO 2	3		4	5V PWR
I2C1 SCL	GPIO 3	5		6	GND
	GPIO 4	7		8	UART0 TX
	GND	9		10	UART0 RX
	GPIO 17	11		12	GPIO 18
	GPIO 27	13		14	GND
	GPIO 22	15		16	GPIO 23
	3.3V PWR	17		18	GPIO 24
SPI0 MOSI	GPIO 10	19		20	GND
SPI0 MISO	GPIO 9	21		22	GPIO 25
SPI0 SCLK	GPIO 11	23		24	GPIO 8
	GND	25		26	GPIO 7
	Reserved	27		28	Reserved
	GPIO 5	29		30	GND
	GPIO 6	31		32	GPIO 12
	GPIO 13	33		34	GND
SPI1 MISO	GPIO 19	35		36	GPIO 16
	GPIO 26	37		38	GPIO 20
	GND	39		40	GPIO 21

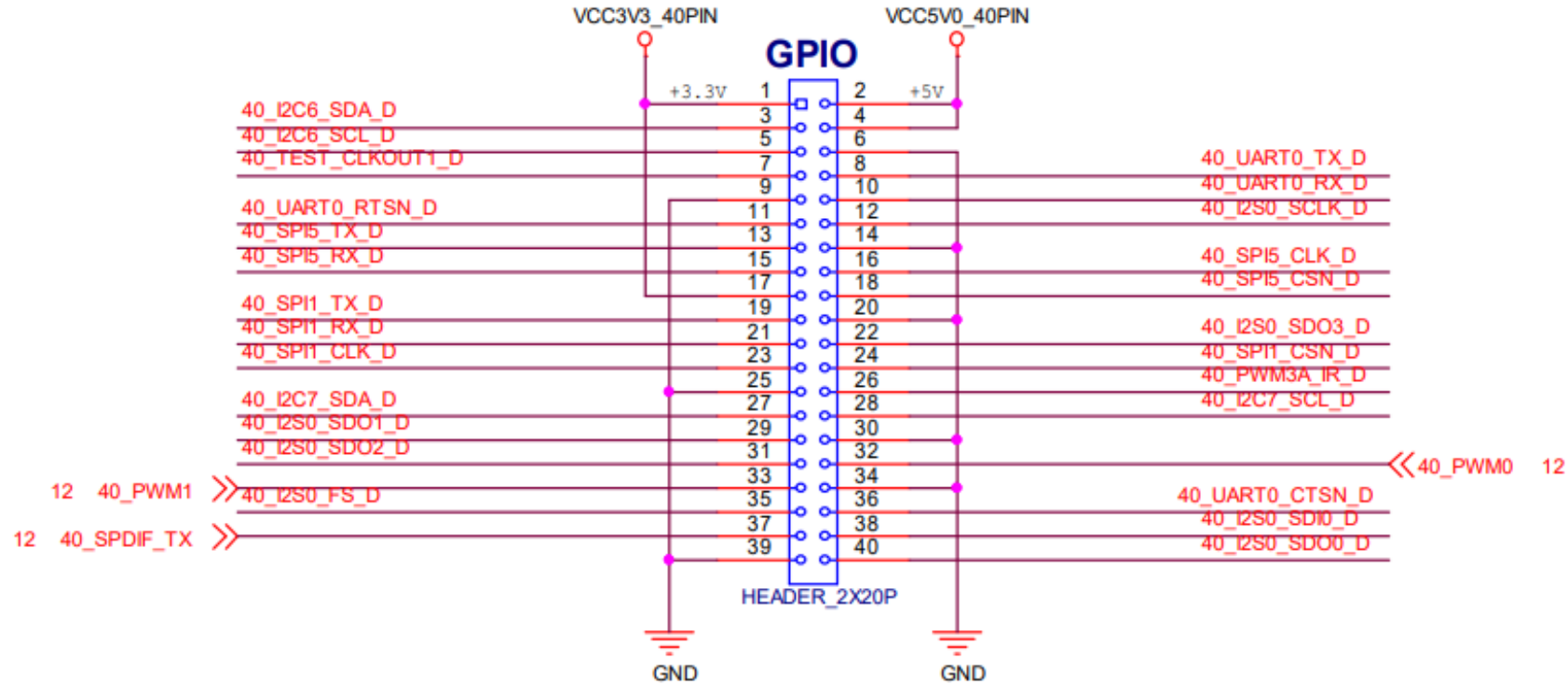
GPIO PIN MAP



ASUS Tinker Edge R 핀 구성 요소

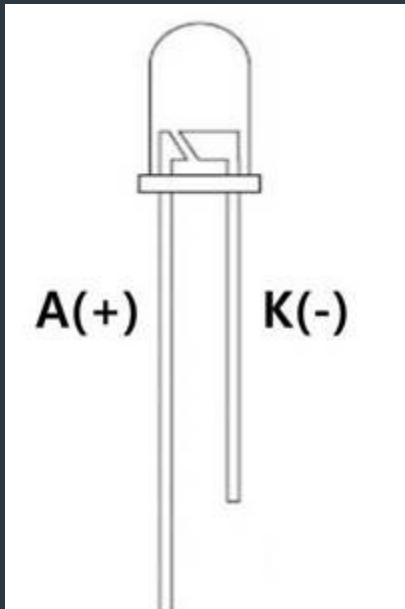
- GPIO (28pin) 지원
- SPI BUS (2pin) 지원
- I2C bus (2pin) 지원
- UART (2pin) 지원
- PMW (3pin) 지원
- PDIF TX(1pin) 지원
- 5V and 3.3V power 2pin씩 지원
- 8pin GND

GPIO PIN MAP schematic



GPIO를 통해 LED(발광다이오드) 제어

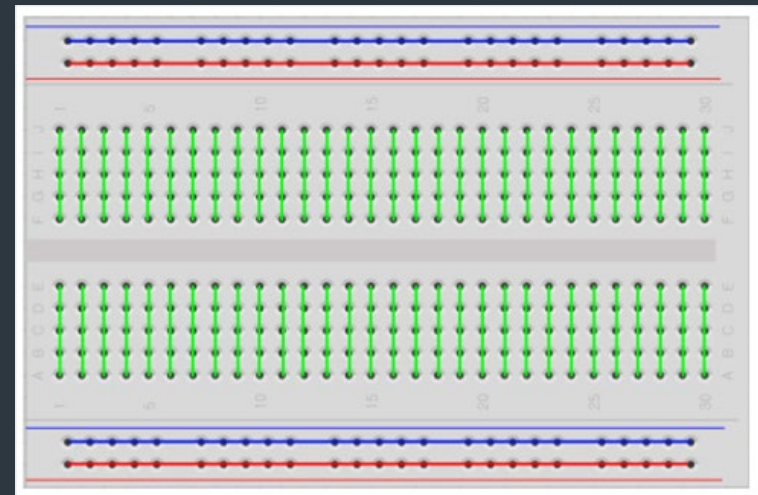
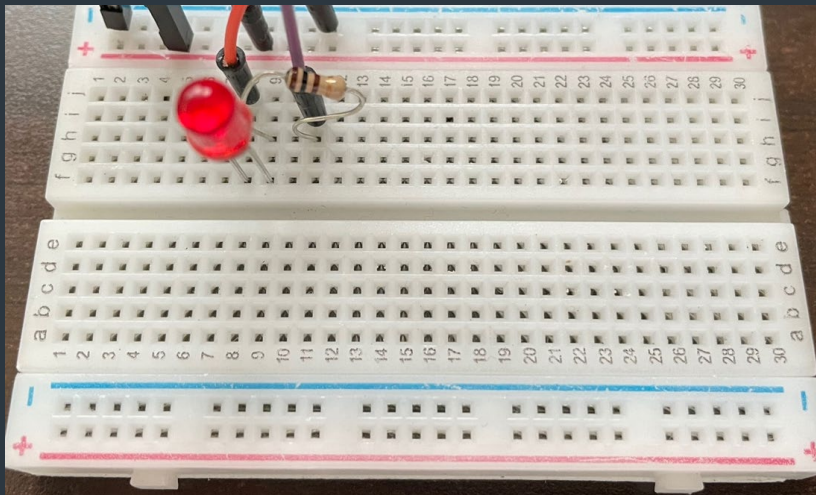
- 발광 다이오드는 아래와 같이 긴 pin이 +(Anode) 짧은 pin이 -(cathode) 이다.
- 일반적인 발광다이오드의 구동 전압 1.8V~2.0V 소모 전류는 20~30mA 이다.
- GPIO은 3.3V로 구동된다. 그리하여 저항을 사용해 구동 전압을 1.8~2.0V로 낮춰야 한다.(그렇지 않으면 과전압으로 다이오드가 망가질 수 있다.)



GPIO를 통해 LED(발광다이오드) 제어

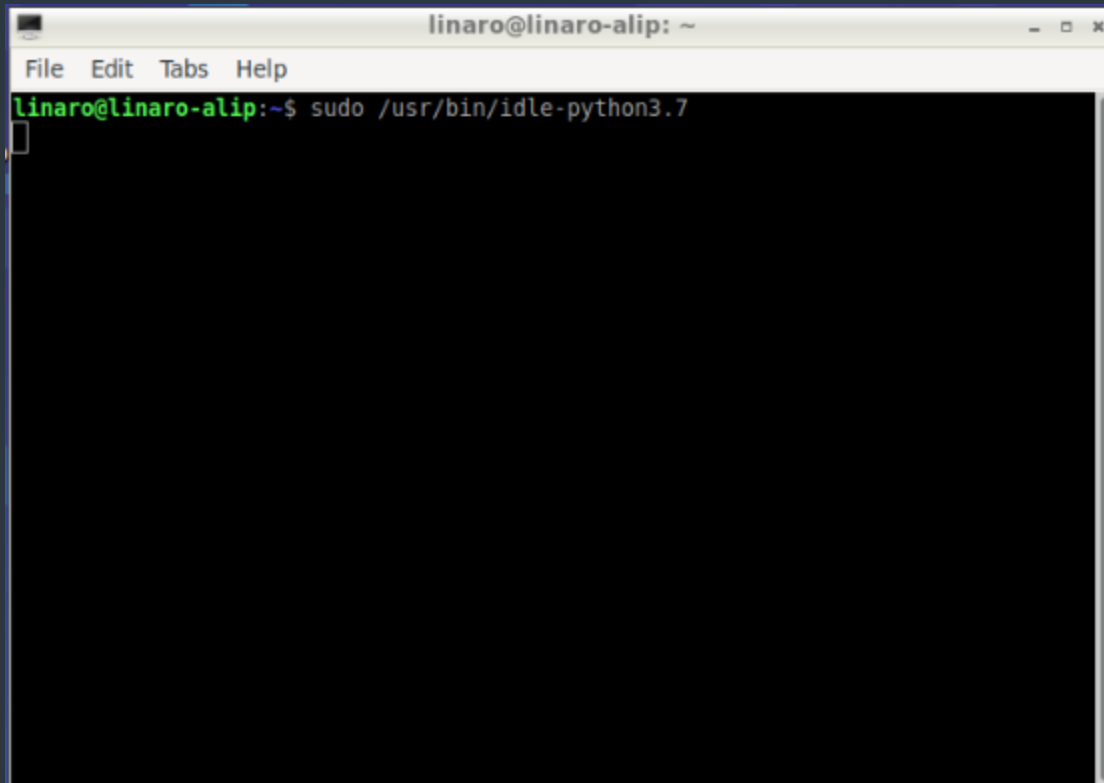
- 옴의 법칙($V=IR$) 을 사용하여, 작동 전압을 위해 사용될 저항치 계산
- Bread board는 아래와 같이 wiring 되어 있음
- $V/I=R$ $(3.3V-(1.8\sim2.0V))/20mA = 65\sim75\Omega$ 의 저항을 사용하여 회로에 저항을 직렬로 구성

(혹은 직 병렬회로로 저항을 구성하여 해당 수치의 저항을 만들 수 있다.)



GPIO를 통해 LED(발광다이오드) 제어

- 관리자 권한으로 Gpio를 제어하기 위하여 Terminal 에서 python 실행
- Sudo /usr/bin/idle-python3.7



```
linaro@linaro-alip: ~  
File Edit Tabs Help  
linaro@linaro-alip:~$ sudo /usr/bin/idle-python3.7
```

GPIO를 통해 LED(발광다이오드) 제어

- ASUS에서 제공한 모듈을 이용하기 위하여

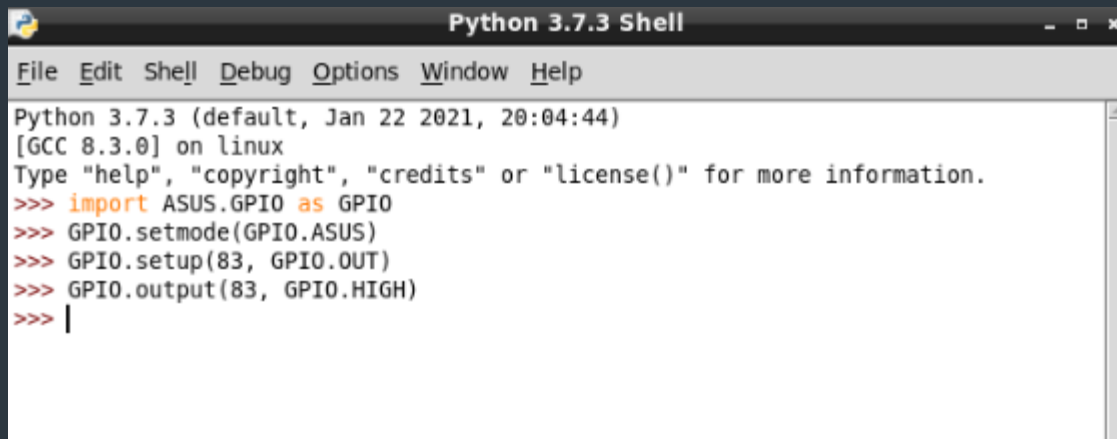
```
import ASUS.GPIO as GPIO
```

```
GPIO.setmode(GPIO.ASUS)
```

```
GPIO.setup(83, GPIO.OUT) ## 다른 Gpio핀도 사용 가능하다.
```

```
GPIO.output(83, GPIO.HIGH) ## 단 전원 핀은 출력 조절이 불가능 하다
```

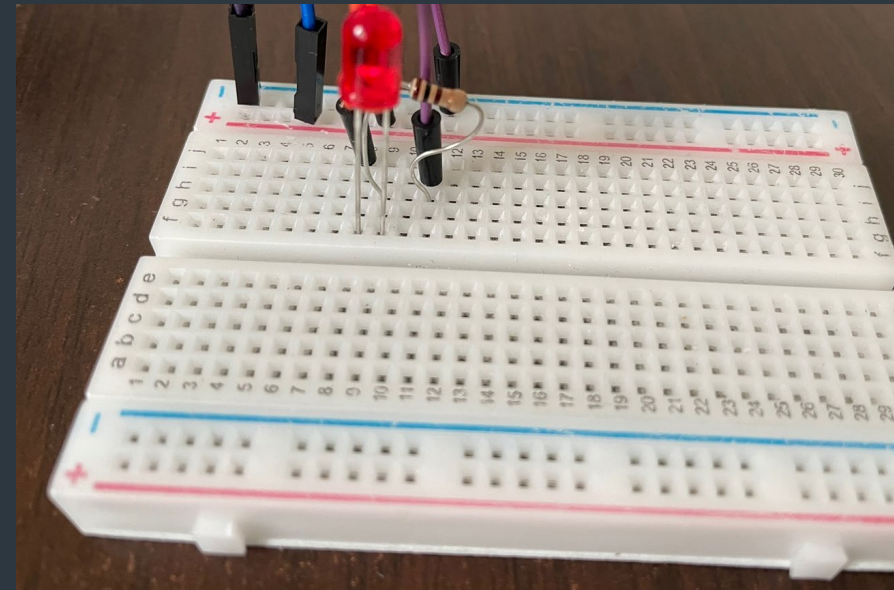
입력 (주석은 입력하지 않아도 된다.)

A screenshot of a Python 3.7.3 Shell window. The window has a title bar that says "Python 3.7.3 Shell" and a menu bar with "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main area shows the following text: "Python 3.7.3 (default, Jan 22 2021, 20:04:44)", "[GCC 8.3.0] on linux", "Type 'help', 'copyright', 'credits' or 'license()' for more information.", and then four lines of Python code being executed: ">>> import ASUS.GPIO as GPIO", ">>> GPIO.setmode(GPIO.ASUS)", ">>> GPIO.setup(83, GPIO.OUT)", and ">>> GPIO.output(83, GPIO.HIGH)". The prompt ">>>" is shown at the end of each line, and a cursor is visible at the end of the last line.

GPIO를 통해 LED(발광다이오드) 제어

- 전에 설명한 gpio readall에서 CPU 83번 Gpio가 IN->OUT으로 변경된 모습
- 다이오드 또한 불 빛이 들어온다.
(계산 저항치 보다 높은 저항과 VCC pin과 다르게 Gpio 전류를 20mA 이상 받기 힘들기 때문에 빛의 세기가 약하다)

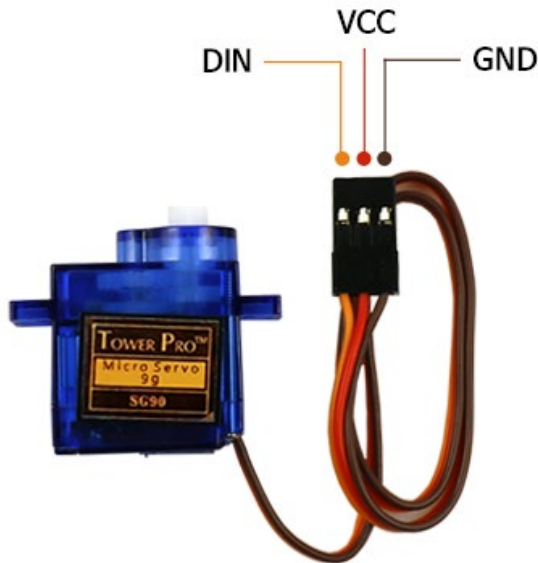
```
linaro@linaro-alip: /usr.../gpio_lib_c_rk3399PRO - x
linaro@linaro-alip:/usr/local/share/gpio_lib_c_rk3399PRO$ gpio readall
-----Tinker-----
| CPU | uPi | Name | Mode | V | Physical | V | Mode | Name | uPi | CPU |
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 73 | 8 | 3.3v | | | 1 | 2 | | | 5v | | |
| 74 | 9 | GPIO2B1 | IN | 1 | 3 | 4 | | | 5v | | |
| 89 | 7 | GPIO2B2 | IN | 1 | 5 | 6 | | | 0v | | |
| 83 | 0 | GPIO2D1 | IN | 0 | 7 | 8 | 1 | SERL | TxD,0 | 15 | 81 |
| 85 | 2 | GPIO2D2 | IN | 0 | 9 | 10 | 1 | SERL | RxD,0 | 16 | 80 |
| 84 | 3 | GPIO2C3 | OUT | 1 | 11 | 12 | 0 | IN | GPIO3D0 | 1 | 120 |
| 85 | 2 | GPIO2C5 | IN | 0 | 13 | 14 | 0 | IN | 0v | | |
| 84 | 3 | GPIO2C4 | IN | 0 | 15 | 16 | 0 | IN | GPIO2C6 | 4 | 86 |
| 40 | 12 | 3.3v | | | 17 | 18 | 0 | IN | GPIO2C7 | 5 | 87 |
| 39 | 13 | GPIO1B0 | IN | 0 | 19 | 20 | 0 | IN | 0v | | |
| 41 | 14 | GPIO1A7 | IN | 0 | 21 | 22 | 0 | IN | GPIO3D4 | 6 | 124 |
| 41 | 14 | GPIO1B1 | IN | 1 | 23 | 24 | 1 | IN | GPIO1B2 | 10 | 42 |
| 71 | 30 | 0v | | | 25 | 26 | 0 | IN | GPIO0A6 | 11 | 6 |
| 126 | 21 | GPIO2A7 | IN | 1 | 27 | 28 | 1 | IN | GPIO2B0 | 31 | 72 |
| 125 | 22 | GPIO3D6 | IN | 0 | 29 | 30 | 0 | IN | 0v | | |
| 150 | 23 | GPIO3D5 | IN | 0 | 31 | 32 | 0 | IN | GPIO4C2 | 26 | 146 |
| 121 | 24 | GPIO4C6 | IN | 0 | 33 | 34 | 0 | IN | 0v | | |
| 149 | 25 | GPIO3D1 | IN | 0 | 35 | 36 | 1 | IN | GPIO2C2 | 27 | 82 |
| 149 | 25 | GPIO4C5 | IN | 0 | 37 | 38 | 0 | IN | GPIO3D3 | 28 | 123 |
| | | 0v | | | 39 | 40 | 0 | IN | GPIO3D7 | 29 | 127 |
```



GPIO를 통해 Servo motor제어

- 사용할 Servo motor는 SG90 모델이다. 아래는 Servo motor의 핀 맵
- 동작 전압이 4.8~7.2V이기 때문에 5V VCC와 연결 DIN은 gpio와 연결한다.

 PIN MAP
핀 배치도



핀 구성	VCC / GND / SIG
회전 각도	0 ~ 180도
제어 범위	각도 및 각속도 제어 가능
동작 전압	4.8 ~ 7.2V
소비 전류	0.2 ~ 0.7A
무게	9g
토크	1.8kg*cm
크기	22.2 * 11.8 * 31mm
특성	· 아두이노 또는 MCU의 PWM (Pulse Width Modulation)제어를 통해 구동하는 서보모터 입니다. · 16칸 2라인

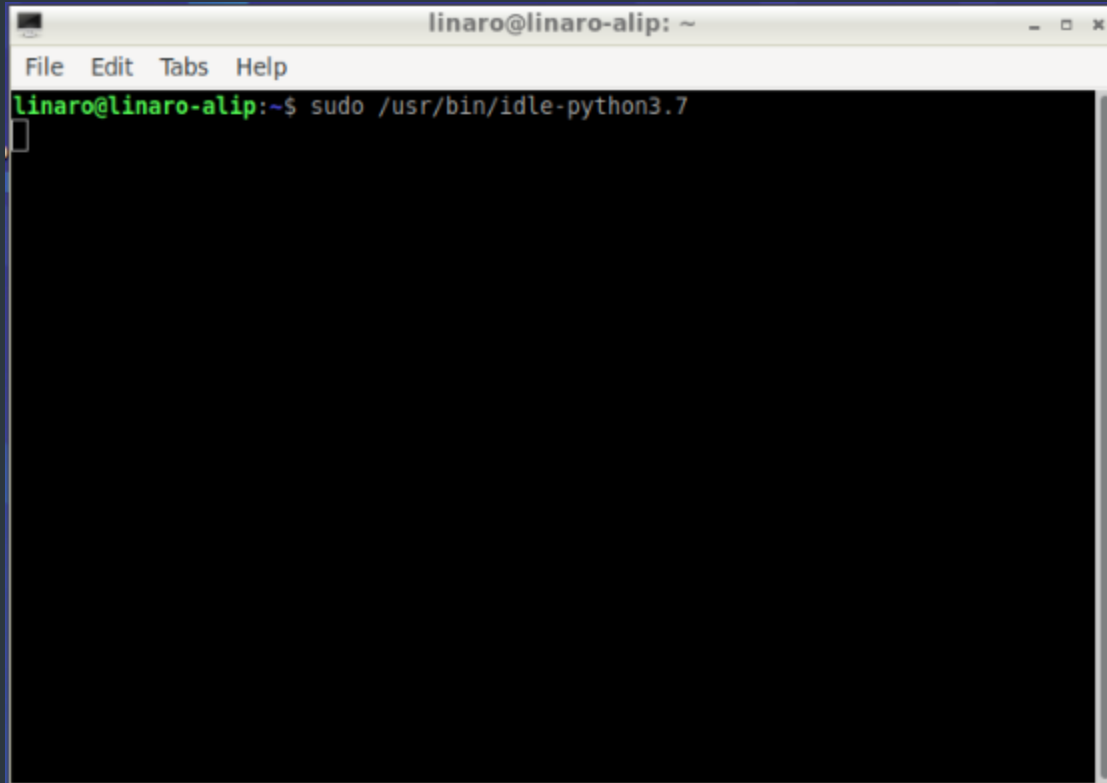
GPIO를 통해 Servo motor제어

- SG90과 연결하기 위하여 Jumper cable Female-male 로 Gpio와 SG90을 연결 한 모습
- 연결은 VCC->5V VCC, DIN->GPIO, 83 GND->GND로 연결



GPIO를 통해 Servo motor제어

- 관리자 권한으로 Gpio를 제어하기 위하여 Terminal 에서 python 실행
- Sudo /usr/bin/idle-python3.7



```
linaro@linaro-alip: ~  
File Edit Tabs Help  
linaro@linaro-alip:~$ sudo /usr/bin/idle-python3.7
```


GPIO를 통해 Servo motor제어

- ASUS에서 제공한 모듈과 myservo라는 변수를 gpio 핀으로 지정하여 83핀 gpio를 이용해 pwm 제어로 구동 되는 모습이다.

```
Python 3.7.3 Shell
File Edit Shell Debug Options Window Help
Python 3.7.3 (default, Jan 22 2021, 20:04:44)
[GCC 8.3.0] on linux
Type "help", "copyright", "credits" or "license()" for more information.
>>> import ASUS.GPIO as GPIO
>>> import time
>>> GPIO.setmode(GPIO.ASUS)
>>> GPIO.setwarnings(False)
>>> myservo = 83
>>> GPIO.setup(myservo,GPIO.OUT)
>>> pwm = GPIO.PWM(myservo,50) # 50hz yani 20mslik periyod
>>> pwm.start(2.5) # 0 derece
>>> |
```

```
import ASUS.GPIO as GPIO
import time
GPIO.setmode(GPIO.ASUS)
GPIO.setwarnings(False)
# 사용하는 gpio setup시 error 출력x
myservo = 83
##Gpio pin
GPIO.setup(myservo,GPIO.OUT)
pwm = GPIO.PWM(myservo,50) # 50hz 마다 pulse
pwm.start(2.5)
# 0 derece (x,5) x는 (2~12까지 고정이며 비례하여
# 0~180도 까지 변경 가능
```