

라즈베리파이 포팅

LED SHIFT,SERIAL

목차

1)라즈베리파이

2)준비물

3)포팅

4)환경설정

5)GPIO

6)스트림

7)버퍼

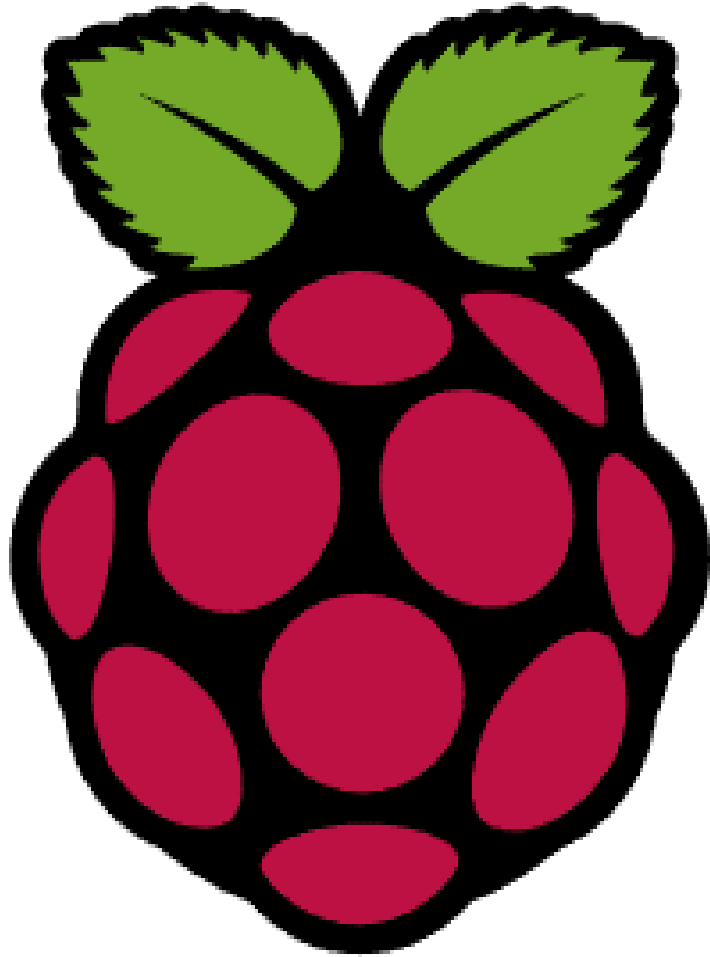
8)SERIAL통신

9)헤더파일

10)회로도

11)구동영상

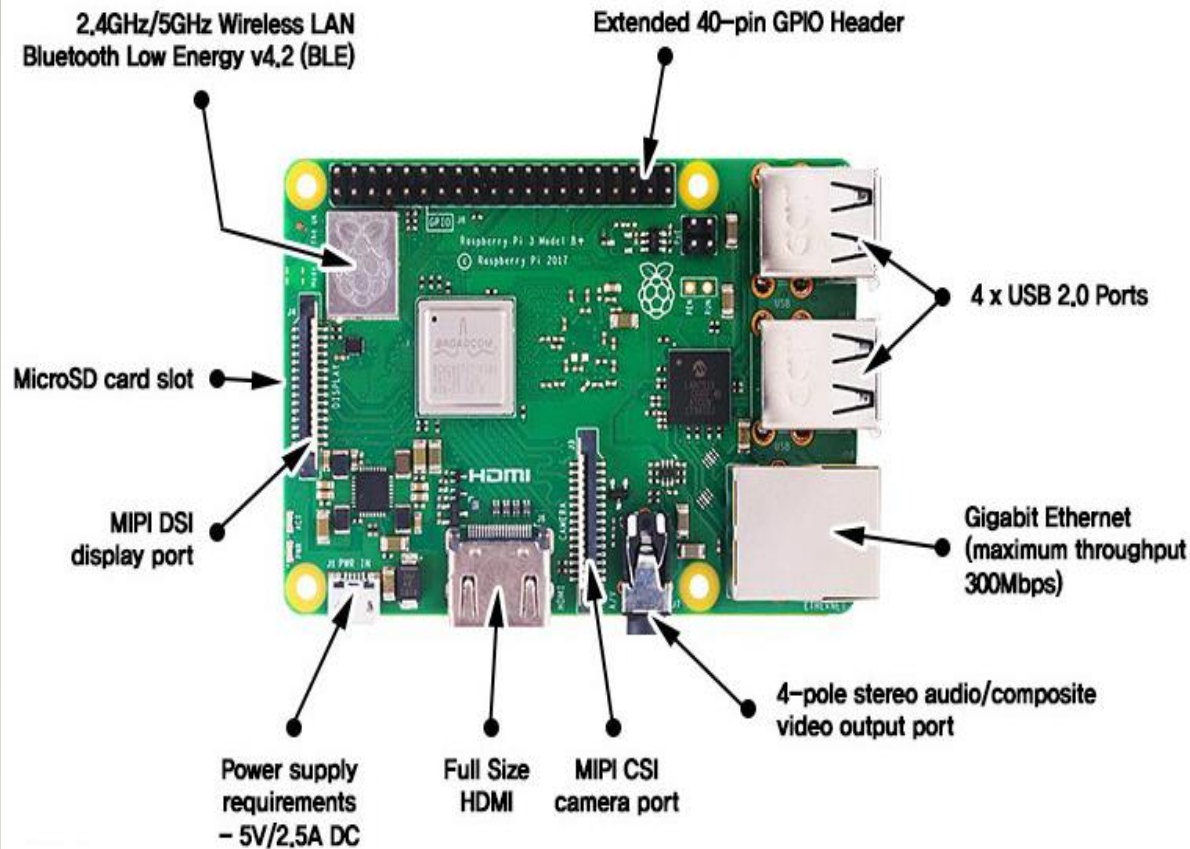
라즈베리파이



라즈베리파이는 영국의
라즈베리 파이 재단이 기초
컴퓨터 과학의 교육을 위해
개발한 싱글보드형 컴퓨터

즉, 마이크로 프로세서 메모리
입출력 연결단자 등 하나의
회로로 구성한 초소형 컴퓨터

라즈베리파이



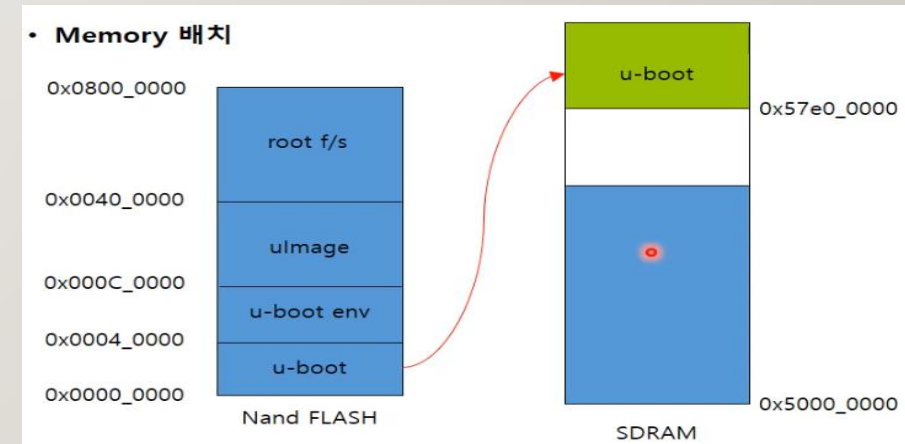
Model	Raspberry Pi 3B+
Soc	Broadcom BCM2837B0, Cortex-A53
CPU	64-bit SoC@1.4GHz, Quadro Core
Memory	1GB LPDDR2 SDRAM
Wifi	Zero W Antenna 2.4GHz and 5GHz IEEE 802.11 b/g/n/ac Wireless LAN
Bluetooth	Bluetooth 4.2, BLE
Ethernet	300Mbps

준비물

라즈베리파이
HDMI 케이블 for Raspberry Pi 3B+, 모니터
마우스
키보드
5핀 케이블 3A
SD 카드
LAN 케이블
GPIO핀 보드, 브레드 보드

포팅

- 컴퓨터가 실행 가능한 프로그램이 원래 설계된 바와 다른 컴퓨팅 환경에서 동작할 수 있도록 하는 과정을 가르킨다.
- CPU, 메모리, 기타 장치가 탑재된 시스템에는 포팅 과정을 통해 os를 사용한다.
 1. 플래시 메모리에 이미지파일을 플래싱한다.(운영체제설치)
 2. 시스템을 부팅하면 부트로더에 의해서 플래시 메모리에 저장되어 있는 시스템파일이 모두 RAM으로 불러와 저장되고 OS를 이용할 수 있게 된다.



포팅

라즈베리파이 공식 홈페이지에서 OS 다운로드

Install Raspberry Pi OS using Raspberry Pi Imager

Raspberry Pi Imager is the quick and easy way to install Raspberry Pi OS and other operating systems to a microSD card, ready to use with your Raspberry Pi. [Watch our 45-second video](#) to learn how to install an operating system using Raspberry Pi Imager.

Download and install Raspberry Pi Imager to a computer with an SD card reader. Put the SD card you'll use with your Raspberry Pi into the reader and run Raspberry Pi Imager.

[Download for Windows](#)

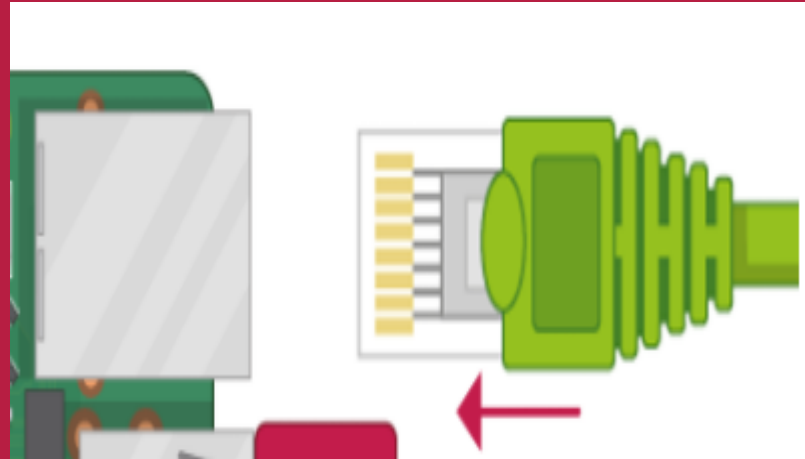
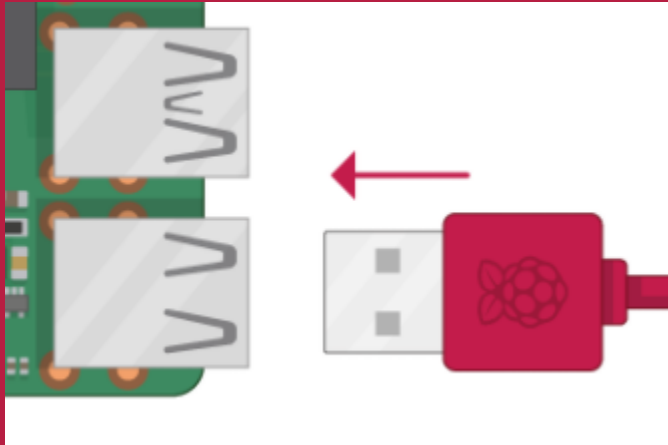
<https://www.raspberrypi.com/software/>

라즈비안 이외에 윈도우10,
안드로이드 등 리눅스 커널을
기반으로 제작된 여러가지 OS
포팅가능

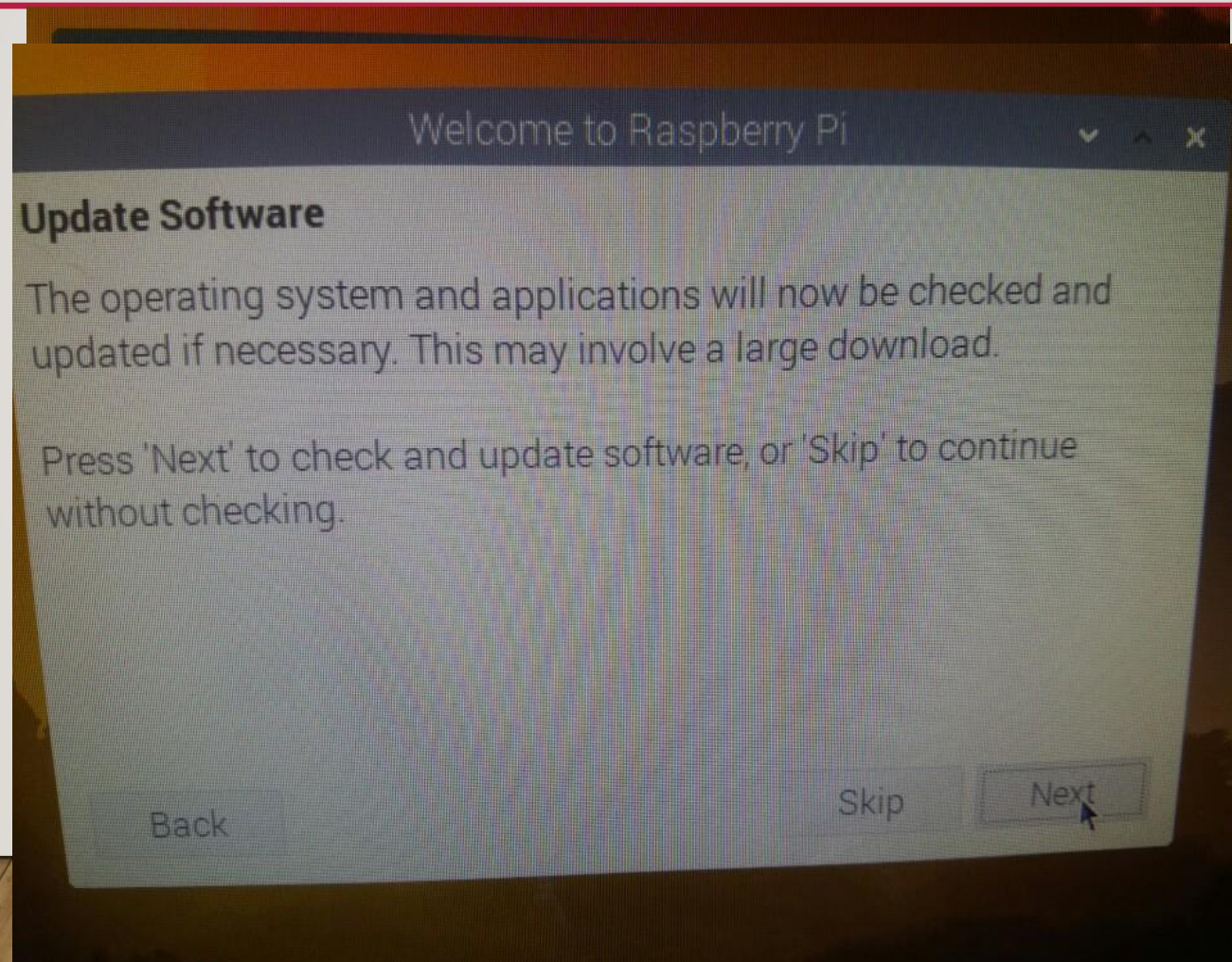
환경설정



- LAN 케이블 연결

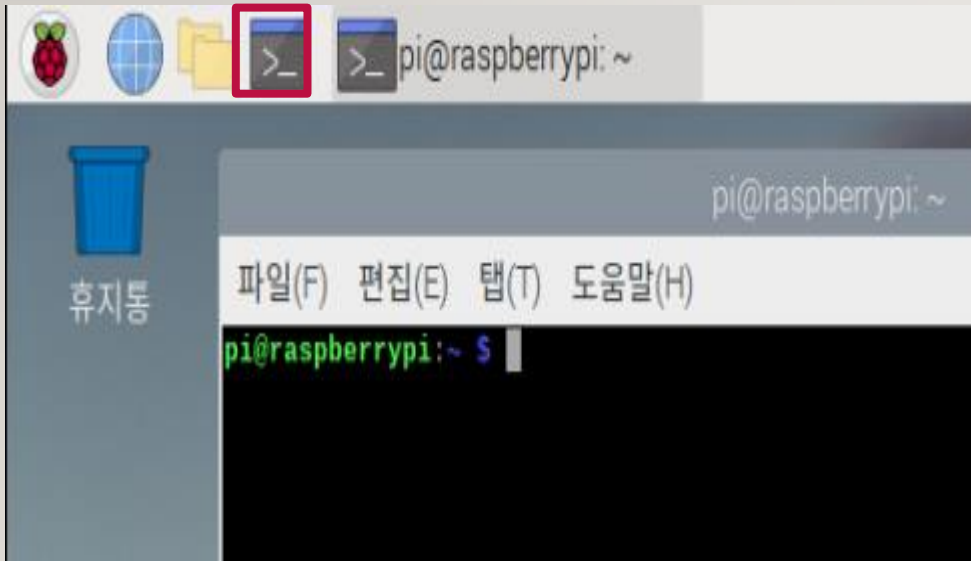


환경 설정



환경 설정

- 터미널 실행



❖ `sudo apt-get update` 패키지 목록을 업데이트

❖ `sudo apt-get upgrade` 패키지 버전 최신화

❖ `sudo apt-get install ibus-hangul` 한글 입력기 설치

❖ `sudo apt-get install fonts-unfonts-core` 한글 폰트 설치

❖ `sudo im-config -n ibus` ibus를 기본 입력기로 설정

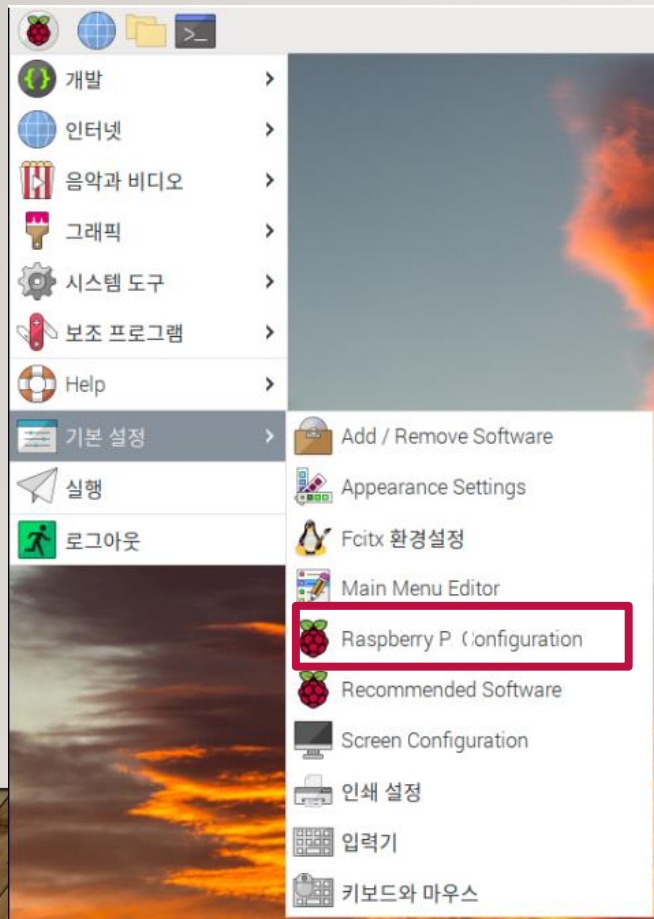
❖ Reboot 재부팅

- 명령어 모음 <https://blog.naver.com/icbanq/221677433460>

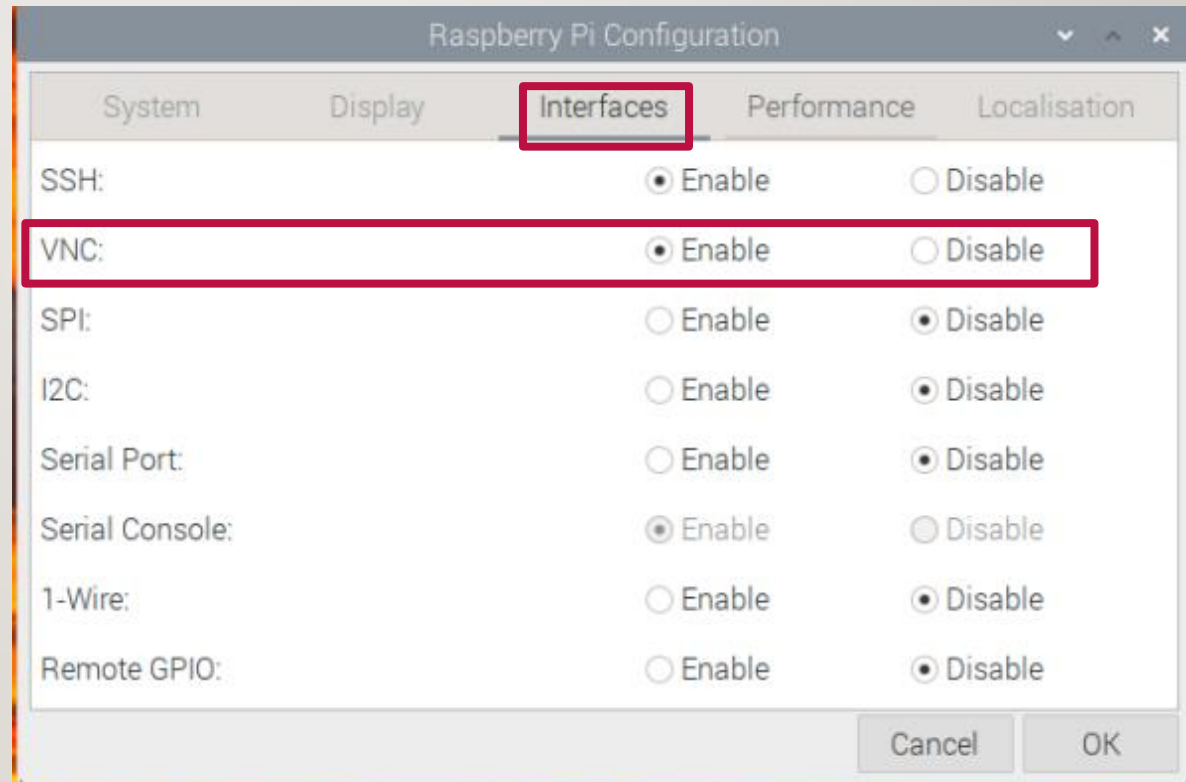
환경설정

❖ VNC : GUI화면을 원격으로 접속.

• VNC 설정



• VNC 허용



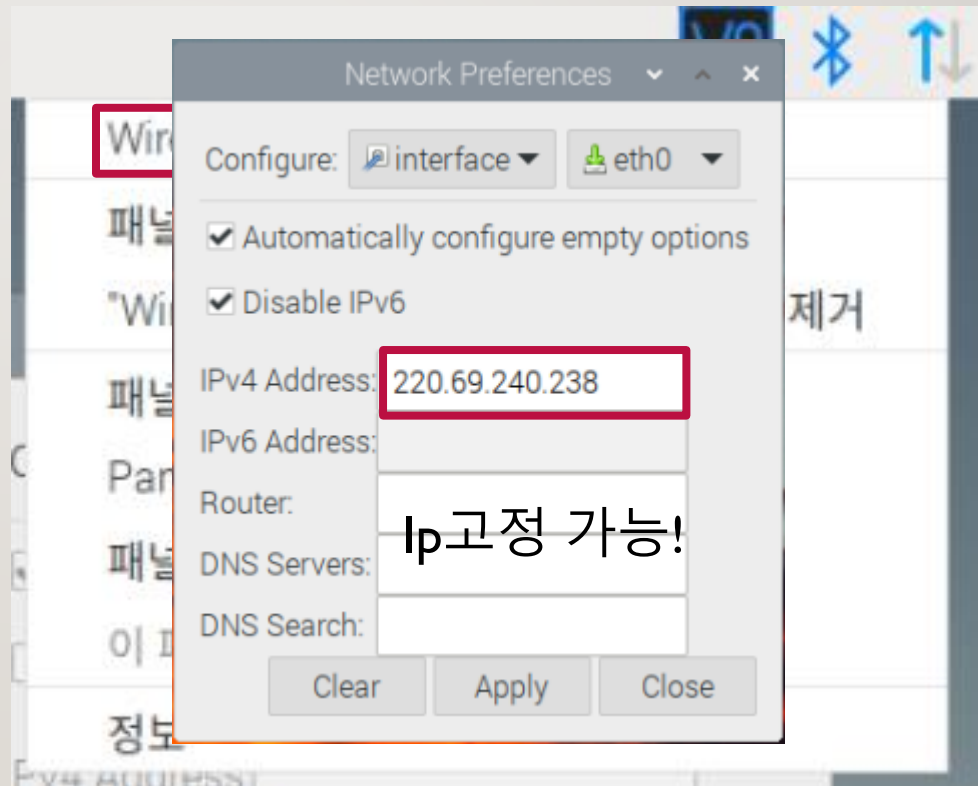
환경설정

❖ ifconfig 유,무선 접속상태 확인

- ip확인

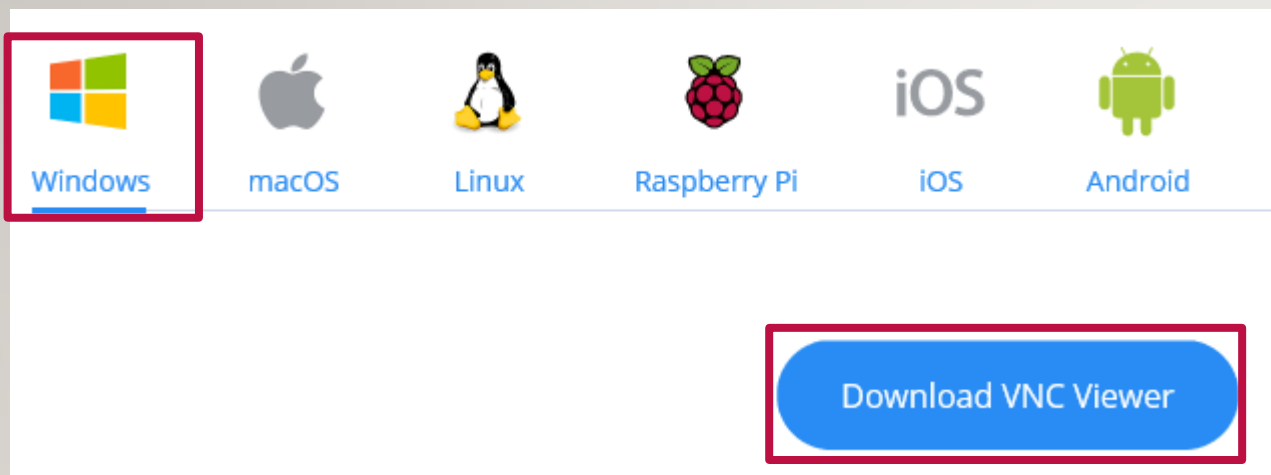
```
pi@raspberrypi:~ $ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 220.69.240.238 netmask 255.255.255.0 broadcast 220.69.240.255
    inet6 fe80::ba27:ebff:fe2d:b688 prefixlen 64 scopeid 0x20<link>
    ether b8:27:eb:2d:b6:88 txqueuelen 1000 (Ethernet)
    RX packets 176840 bytes 23421105 (22.3 MiB)
    RX errors 0 dropped 10 overruns 0 frame 0
    TX packets 48126 bytes 13400796 (12.7 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 41 bytes 5190 (5.0 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 41 bytes 5190 (5.0 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```



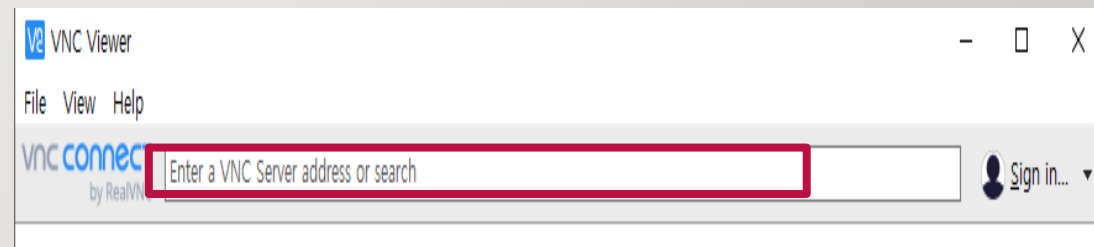
환경 설정

- VNC Viewer 설치



❖ 윈도우용으로 다운로드

- 실행 후 ip 입력

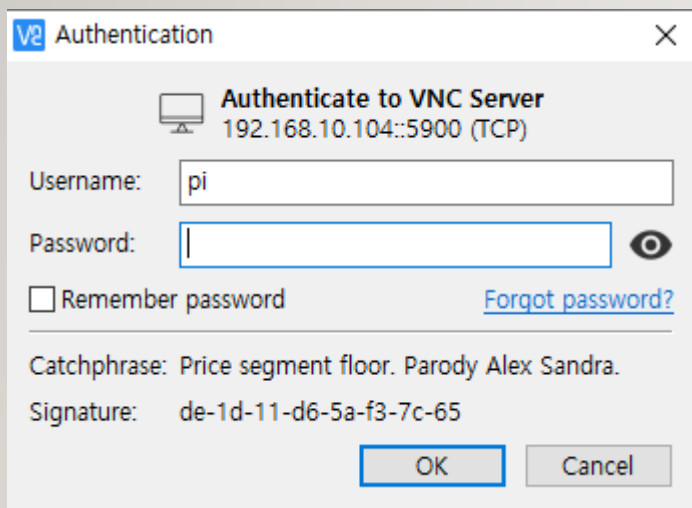


❖ 터미널에서 확인한 ip 입력

- VNC Viewer 다운로드 링크 <https://www.realvnc.com/en/connect/download/viewer/>

환경 설정

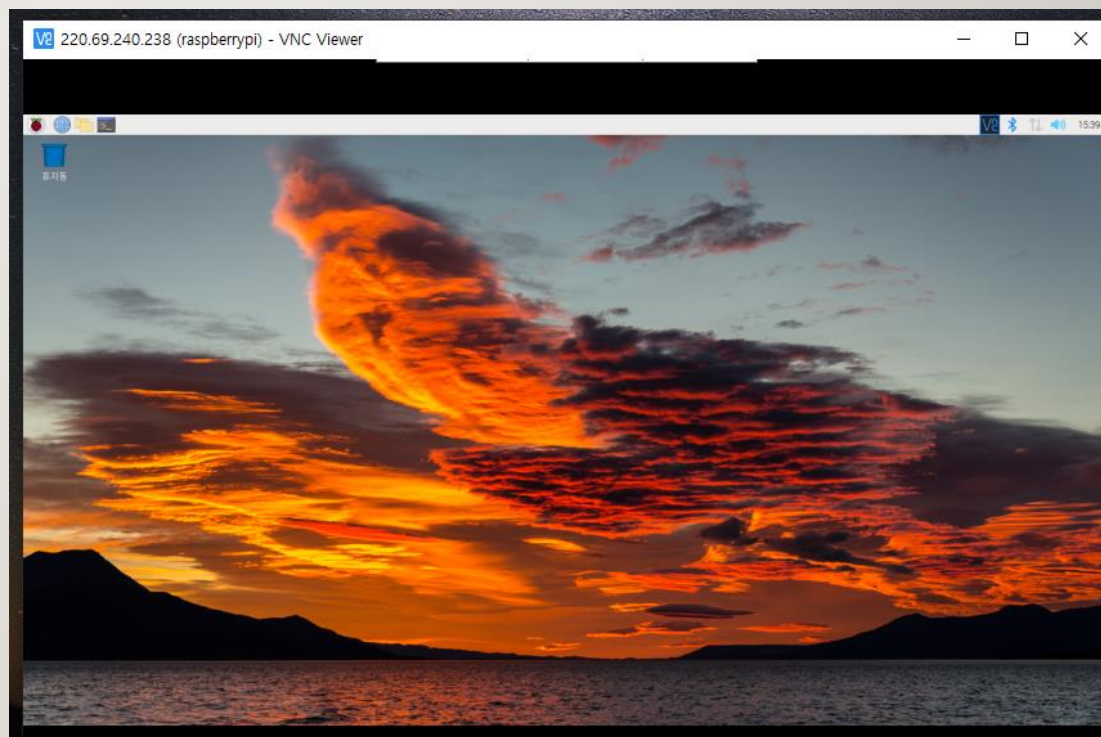
- 로그인



❖ 초기 아이디는 pi이며, 초기 비밀번호는 raspberry이다.

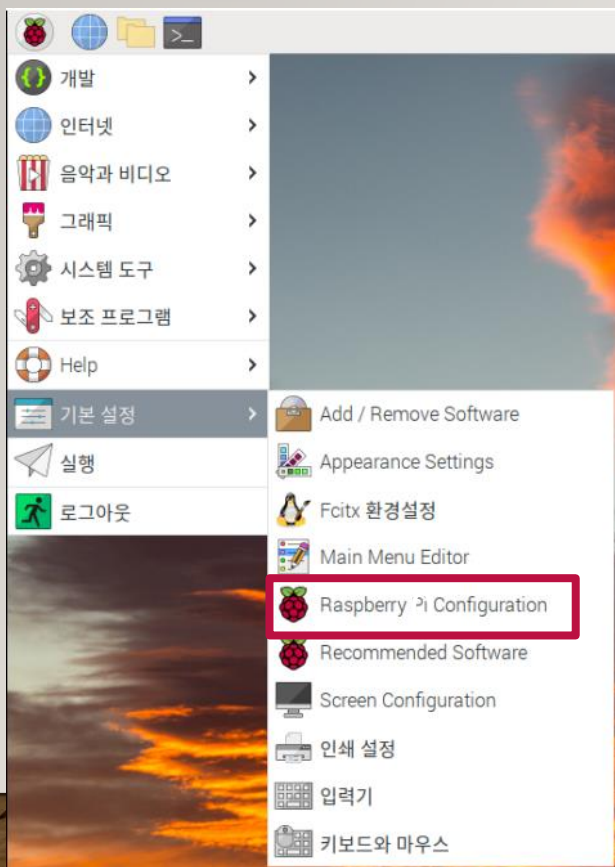
❖ 아까 비밀번호를 설정했다면 설정한 비밀번호 입력

- 컴퓨터로 원격제어 가능

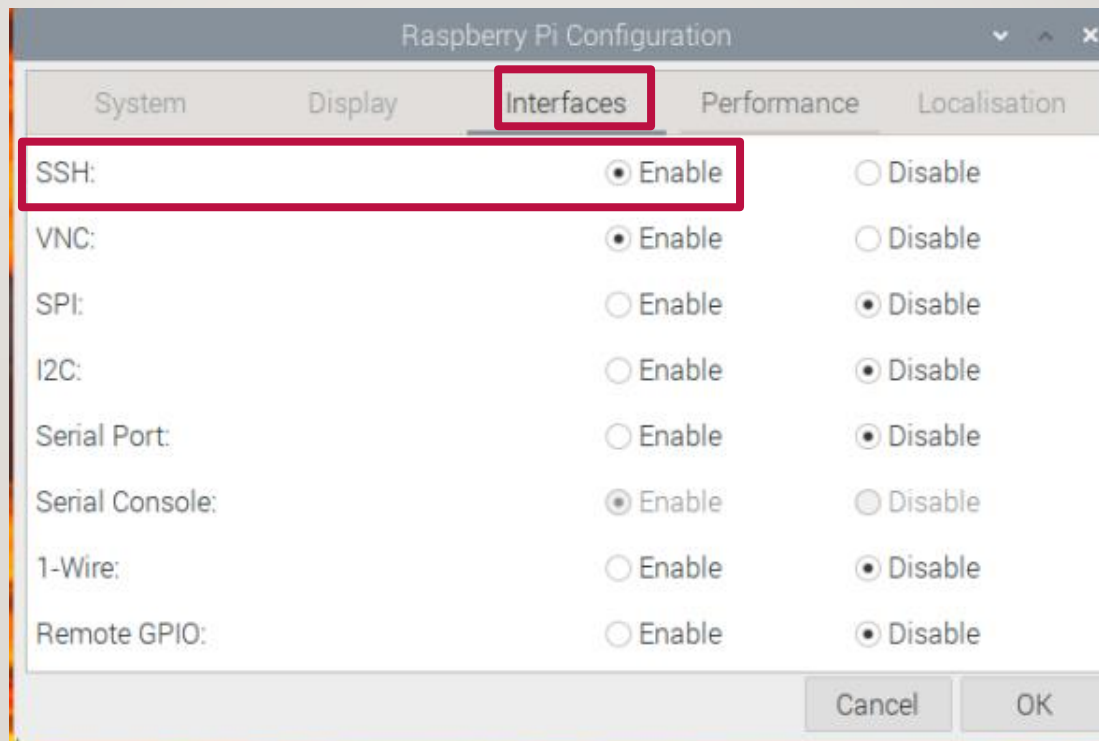


환경설정

- SSH 설정








- SSH 허용

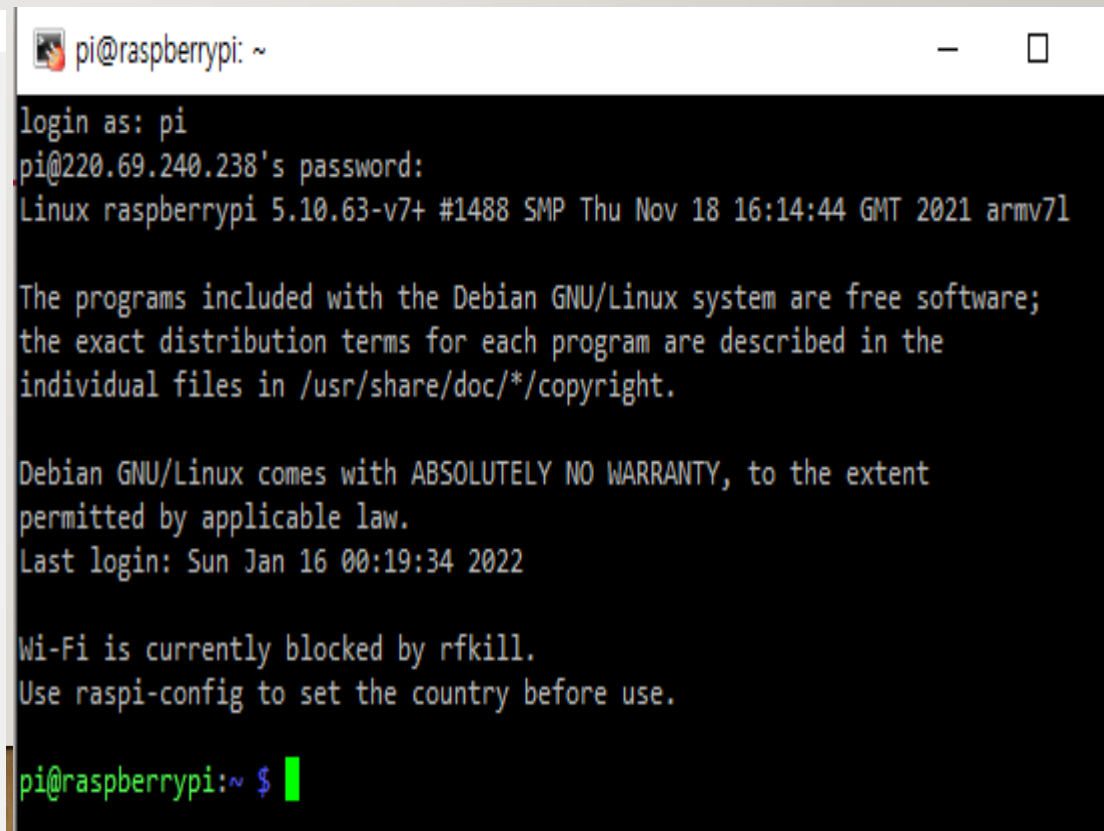
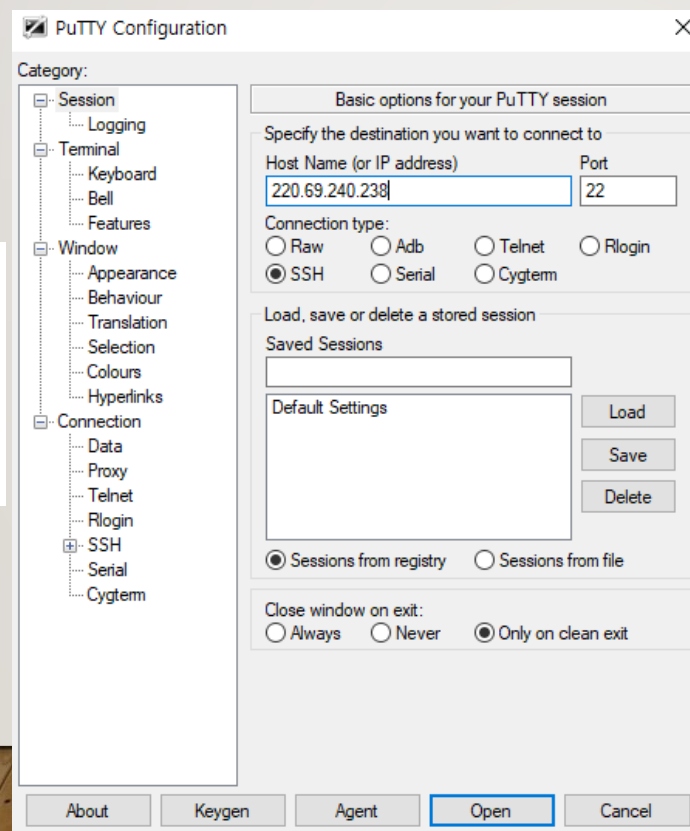


❖ SSH는 Secure Shell Protocol, 즉 네트워크 프로토콜 중 하나로 컴퓨터와 컴퓨터가 인터넷과 같은 Public Network를 통해 서로 통신을 할 때 보안적으로 안전하게 통신을 하기 위해 사용하는 프로토콜입니다.

환경 설정

- 프로그램 실행
- 확인한 ip 입력
- 로그인 후 원격으로 터미널 사용가능

	plink	2016-07-02 오후 9:25
	pscp	2016-07-02 오후 9:26
	psftp	2016-07-02 오후 9:26
	putty	2016-07-02 오후 9:26
	uninstall	2022-01-15 오전 1:28



환경설정

wiringPi 라이브러리

터미널 창에 순서대로

1. sudo apt-get update
 2. sudo apt-get upgrade
 3. sudo apt-get install
 4. git clone http://github.com/adafruit/Adafruit_BMP085_Library
 5. ls
 6. cd wiringpi
 7. ./build
- 설치 및 확인
1. gpio -v
 2. gpio readall

```
pi@raspberrypi:~ $ gpio readall
```

Pi 3B+											
BCM	wPi	Name	Mode	V	Physical	V	Mode	Name	wPi	BCM	
		3.3v			1	2		5v			
2	8	SDA.1	IN	1	3	4		5v			
3	9	SCL.1	IN	1	5	6		0v			
4	7	GPIO. 7	IN	1	7	8	0	IN	TxD	15	14
		0v			9	10	1	IN	RxD	16	15
17	0	GPIO. 0	IN	1	11	12	1	IN	GPIO. 1	1	18
27	2	GPIO. 2	IN	0	13	14		0v			
22	3	GPIO. 3	IN	0	15	16	0	IN	GPIO. 4	4	23
		3.3v			17	18	0	IN	GPIO. 5	5	24
10	12	MOSI	IN	0	19	20		0v			
9	13	MISO	IN	0	21	22	0	IN	GPIO. 6	6	25
11	14	SCLK	IN	0	23	24	1	IN	CE0	10	8
		0v			25	26	1	IN	CE1	11	7
0	30	SDA.0	IN	1	27	28	1	IN	SCL.0	31	1
5	21	GPIO.21	IN	1	29	30		0v			
6	22	GPIO.22	OUT	0	31	32	0	IN	GPIO.26	26	12
13	23	GPIO.23	OUT	1	33	34		0v			
19	24	GPIO.24	OUT	1	35	36	0	IN	GPIO.27	27	16
26	25	GPIO.25	OUT	1	37	38	0	IN	GPIO.28	28	20
		0v			39	40	0	IN	GPIO.29	29	21
BCM	wPi	Name	Mode	V	Physical	V	Mode	Name	wPi	BCM	
Pi 3B+											

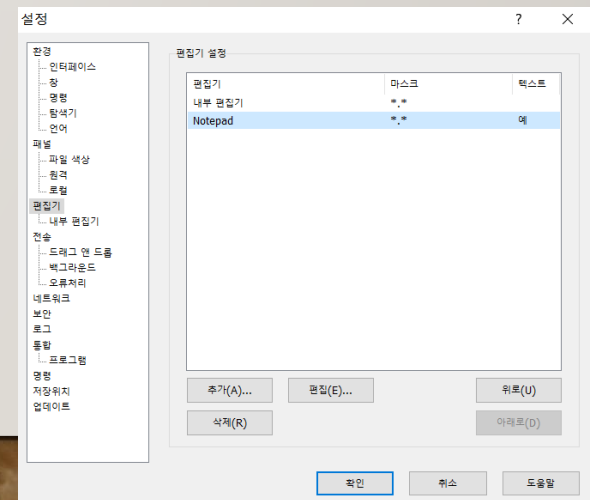
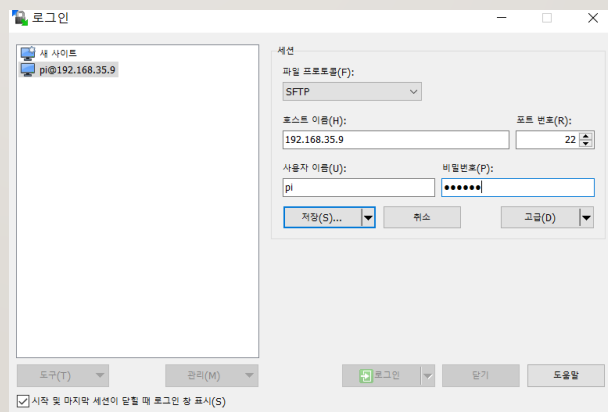
환경설정

- Win scp설치

- 원격으로 파일 수정 다운로드 업로드하는 프로그램
- <https://winscp.net/eng/download.php>에서 설치

- 설치 후 설정

1. Ip 주소
2. 사용자이름
3. 비밀번호 입력
4. 옵션에 설정 편집기 선택
5. 내부편집기 에서 다른 편집기로 변경



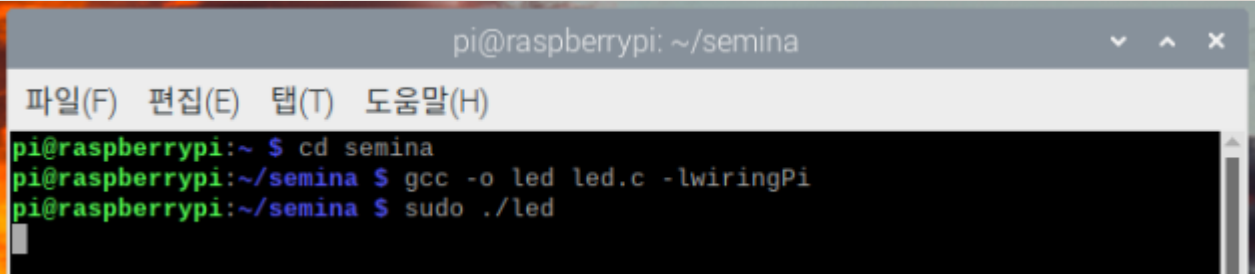
환경설정

1. Winscp에서 새로운 폴더를 만들고 그 폴더에 작성한 코드를 넣고
터미널 창에서

1. Cd 폴더명

2. gcc -o 파일명 파일명.c -lwiringPi

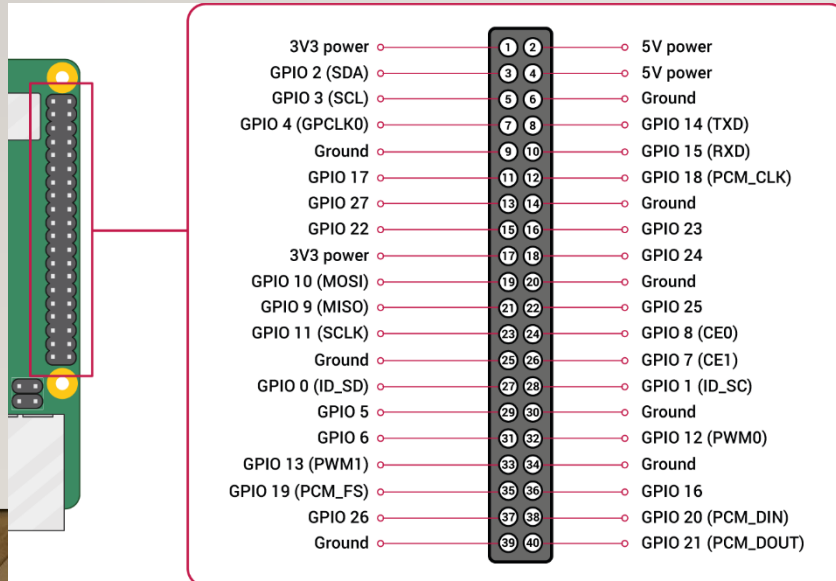
3. sudo ./파일명

A terminal window titled 'pi@raspberrypi: ~/semina' with standard window controls. The menu bar shows '파일(F) 편집(E) 탭(T) 도움말(H)'. The terminal text shows the following commands and prompts:

```
pi@raspberrypi:~ $ cd semina
pi@raspberrypi:~/semina $ gcc -o led led.c -lwiringPi
pi@raspberrypi:~/semina $ sudo ./led
```

GPIO

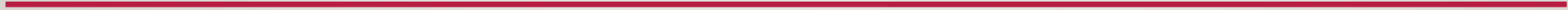
- general-purpose input/output
- 사용자에게 의해 제어될 수 있는, 집적 회로나 전기 회로 기판의 디지털 신호 핀



```
pi@raspberrypi:~ $ gpio readall
```

BCM	wPi	Name	Mode	V	Physical	V	Mode	Name	wPi	BCM
		3.3v			1	2		5v		
2	8	SDA.1	IN	1	3	4		5v		
3	9	SCL.1	IN	1	5	6		0v		
4	7	GPIO. 7	IN	1	7	8	0	IN	TxD	15 14
		0v			9	10	1	IN	RxD	16 15
17	0	GPIO. 0	IN	1	11	12	1	IN	GPIO. 1	1 18
27	2	GPIO. 2	IN	0	13	14		0v		
22	3	GPIO. 3	IN	0	15	16	0	IN	GPIO. 4	4 23
		3.3v			17	18	0	IN	GPIO. 5	5 24
10	12	MOSI	IN	0	19	20		0v		
9	13	MISO	IN	0	21	22	0	IN	GPIO. 6	6 25
11	14	SCLK	IN	0	23	24	1	IN	CE0	10 8
		0v			25	26	1	IN	CE1	11 7
0	30	SDA.0	IN	1	27	28	1	IN	SCL.0	31 1
5	21	GPIO.21	IN	1	29	30		0v		
6	22	GPIO.22	OUT	0	31	32	0	IN	GPIO.26	26 12
13	23	GPIO.23	OUT	1	33	34		0v		
19	24	GPIO.24	OUT	1	35	36	0	IN	GPIO.27	27 16
26	25	GPIO.25	OUT	1	37	38	0	IN	GPIO.28	28 20
		0v			39	40	0	IN	GPIO.29	29 21

SERIAL

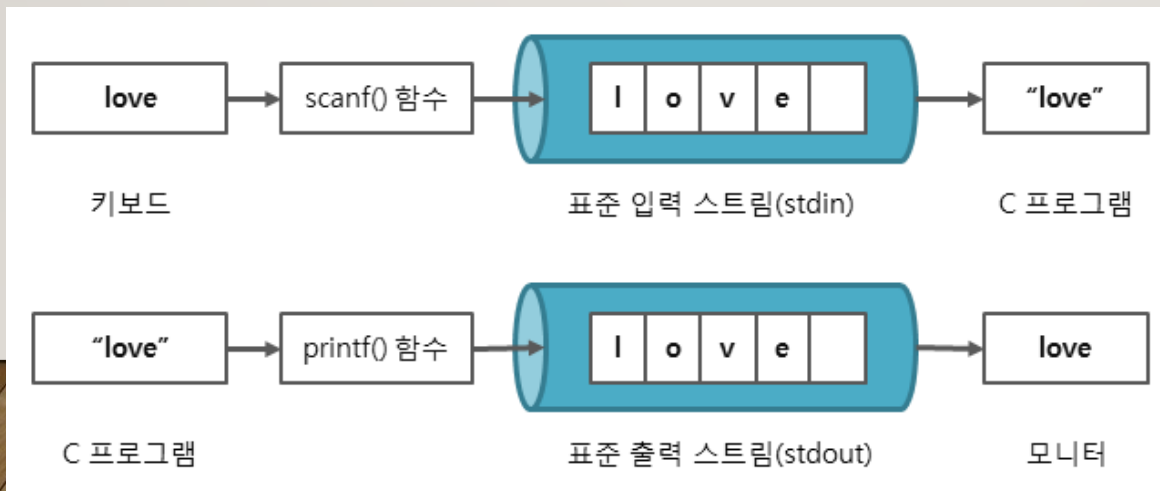


스트림

- 표준 스트림은 유닉스 계열의 운영체제에서 컴퓨터 프로그램과 외부장치 사이에 미리 연결된 입출력 통로.
- C 실행 라이브러리의 `<stdio.h>` 헤더파일에 정의되어 있다.
- 파일 디스크립터 0, 1, 2로 표현하기도 한다.
- 단방향 통신만 가능하다.
- 순차적 통신을 한다.

스트림

- 표준 스트림은 3개가 있다.
 1. Stdin(Standard input, STDIN, 0) : 입력을 위한 스트림
 2. Stdout(Standard Output, STDOUT, 1) : 출력을 위한 스트림
 3. Stderr(Standard Error, STDERR, 2) : 오류 메시지를 출력하기 위한 스트림
- 데이터의 입출력은 시스템 콘솔의 키보드와 모니터를 통해 일어난다.

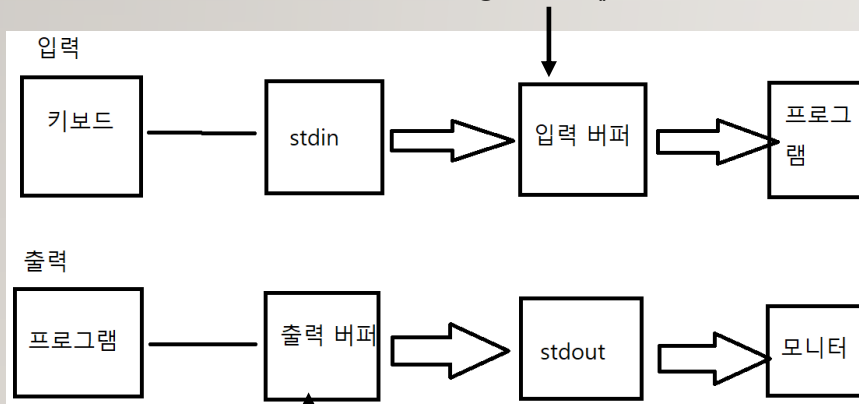


버퍼

`fflush(stdin);` 함수는 윈도우에서 정상작동하지만

리눅스에서는 오류가 발생한다.

입력 버퍼를 지우기 위해서 `getchar();`를 사용한다.



오류를 방지하고자 `fflush(stdout);` 함수로 출력 버퍼 삭제하고(비우고) 출력한다.

- 스트림은 내부에 버퍼라는 임시 메모리 공간을 가지고 있습니다.
- 버퍼를 이용하면 입력과 출력을 좀 더 효율적으로 처리할 수 있게 됩니다.
 1. 문자를 하나씩 전달하는 것이 아닌 묶어서 한 번에 전달하므로, 전송 시간이 적게 걸려 성능이 향상됩니다.
 2. 사용자가 문자를 잘못 입력했을 경우 수정을 할 수가 있습니다.

wiringPi 헤더파일

`int serialOpen (char *device, int baud) ;`

이 시리얼 오픈 함수는 시리얼 포트를 열고 초기화 하며, 속도(buad rate)를 설정하는 함수이다.

Device는 "/dev/ttyAMA0"로 오픈 할 시리얼 포트를 지정하며, baud는 통신 속도를 지정하면 된다.

리턴 값으로는 파일 디스크립터나 에러가 있을 경우 -1을 넘겨 준다.

`void serialClose (int fd) ;`

오픈했던 시리얼 포트를 닫는 함수이다.

`void serialFlush (int fd) ;`

수신된 모든 데이터를 삭제하는 함수이다.

`void serialPutchar (int fd, unsigned char c) ;`

지정된 파일 디스크립터에 의해 식별되는 시리얼 장치에 단일 바이트를 전송하는 함수이다.(데이터 전송)

`void serialPuts (int fd, char *s) ;`

지정된 파일 디스크립터에 의해 식별되는 시리얼 장치에 NUL 종료 문자열을 전송하는 함수이다.

`void serialPrintf (int fd, char *message, ...) ;`

시리얼 장치에서 printf함수를 에뮬레이트하는 함수이다.

`int serialDataAvail (int fd) ;`

읽을 수 있는 문자의 수를 돌려주거나, 오류상태에서는 -1을 돌려주는 함수이다.

Rx에 데이터 들어오는지 확인하는 함수

`int serialGetchar (int fd) ;`

이 함수는 시리얼 포트에서 한 바이트를 받아서 리턴 해 주는 함수이다.

Fd는 시리얼 포트를 오픈시에 리턴된 파일 디스크립터 함수이다.

STDIO.H 헤더파일

- `getchar()`, `fgetc()`, `getc()` 문자단위로 입력하는 함수
- `putchar()`, `fputc()`, `putc()` 문자단위로 출력하는 함수
- `gets()`, `fgets()` 문자열을 입력하는 함수
- `puts()`, `fputs()` 문자열을 출력하는 함수
- `scanf()`, `fscanf()` 자료형에 따라 자료를 입력하는 함수
- `printf()`, `fprintf()` 자료형에 따라 자료를 출력하는 함수

SERIAL 통신

- 라즈베리파이3 B모델은 블루투스 4.1을 지원하는데 UART와 같은 포트를 사용하기 때문에 블루투스를 사용중지 하여야 한다.

```
pi@raspberrypi:~ $ sudo nano /boot/config.txt
```

```
enable_uart=1  
# disable bluetooth  
dtoverlay=pi3-disable-bt
```

- Sudo nano /boot/config.txt 편집기로 오픈

마지막줄에 추가해준다.

- Enable_uart=1
- Dtoverlay=pi3-disable-bt

SERIAL 통신

```
pi@raspberrypi: ~  
pi@raspberrypi:~ $ sudo raspi-config
```

```
Raspberry Pi Software Configuration Tool (raspi-config)  
1 System Options      Configure system settings  
2 Display Options     Configure display settings  
3 Interface Options   Configure connections to peripherals  
4 Performance Options Configure performance settings
```

```
P4 SPI      Enable/disable automatic loading of SPI kernel module  
P5 I2C      Enable/disable automatic loading of I2C kernel module  
P6 Serial Port Enable/disable shell messages on the serial connection  
P7 1-Wire   Enable/disable one-wire interface  
P8 Remote GPIO Enable/disable remote access to GPIO pins
```

```
pi@raspberrypi:~ $ sudo nano /boot/cmdline.txt
```

```
console=serial0,115200
```

- Sudo raspi-config 설정창 열기
- 인터페이스 옵션 -> 시리얼 포트 enable
- 기본 통신속도(115200) 삭제
- Console=serial0, 115200 삭제

SERIAL 통신

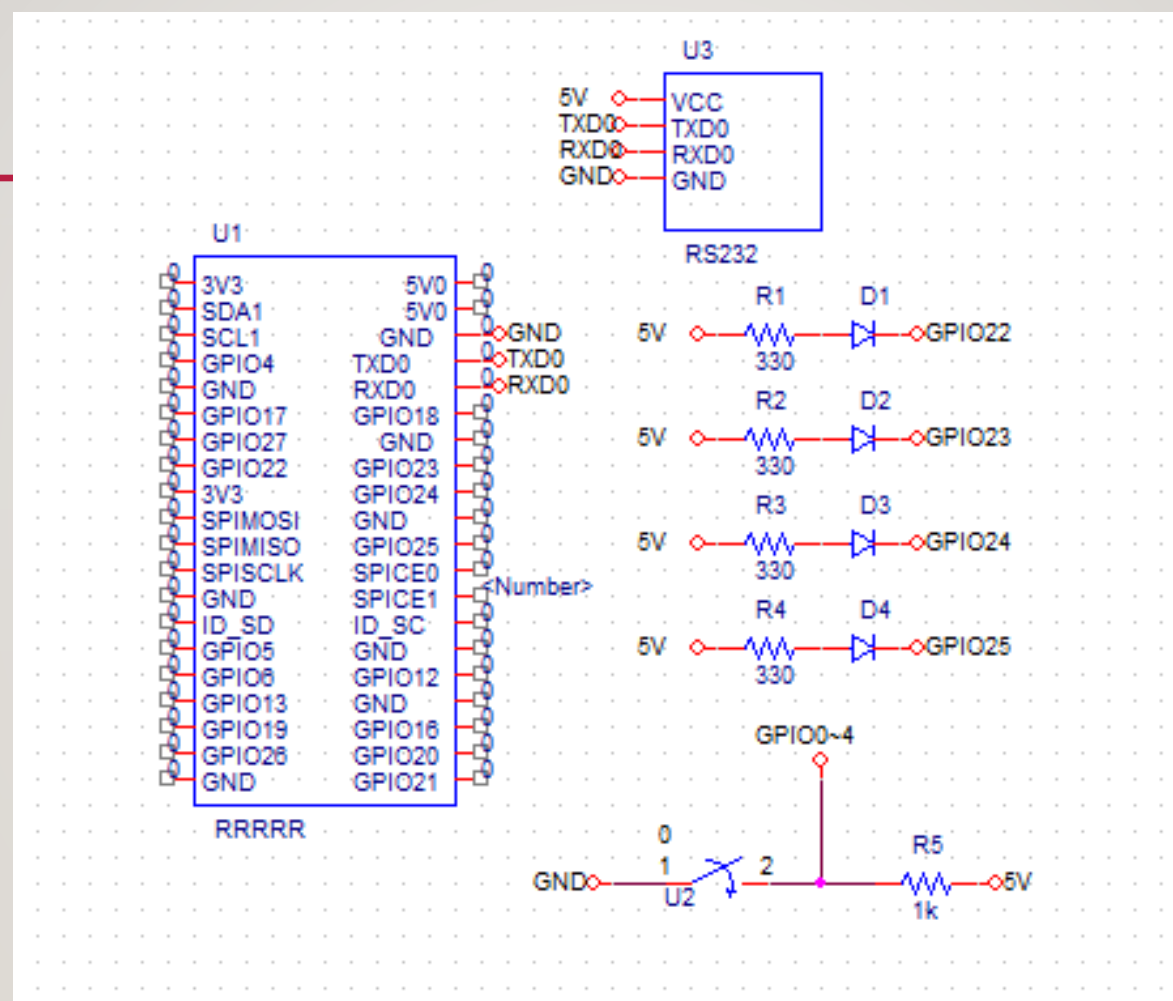
```
pi@raspberrypi:~ $ ls -l /dev
```

```
serial0 -> ttyAMA0  
serial1 -> ttyS0
```

```
stderr -> /proc/self/fd/2  
stdin -> /proc/self/fd/0  
stdout -> /proc/self/fd/1
```

- `ls -l /dev` 시리얼 설정 상태 확인
- 블루투스(ttyAMA0)
- 시리얼(ttyS0)
- 스트림 및 파일 디스크립터 확인

회로도



문제

led 개수 자율 (4개)

sw1 한 번 누를 때 마다 옆으로 한 칸씩 점등

sw2 자유 패턴 점등 (0 2 1 3 0 2 1 3 순으로 진행!)

sw3 serial 1~n 를 입력시 그 led 점등

sw4 serial 통신으로 라즈베리와 컴퓨터 통신

구동 영상

```
ser.c: In function 'main':  
ser.c:67:53: warning: implicit declaration of  
'perror'? [-Wimplicit-function-declaration]  
67 |     fprintf(stdout, "Unable to start wiringPi  
|  
|  
@raspberrypi:~/semina $ gcc -oledser ledser.c -  
ledser.c: In function 'main':  
ledser.c:67:53: warning: implicit declaration of  
'perror'? [-Wimplicit-function-declaration]  
67 |     fprintf(stdout, "Unable to start wiringPi  
|  
|  
@raspberrypi:~/semina $ sudo ./ledser  
[A^][A^C  
@raspberrypi:~/semina $ gcc -oledser ledser.c -l  
ledser.c: In function 'main':  
ledser.c:67:53: warning: implicit declaration of fu  
n 'perror'? [-Wimplicit-function-declaration]  
67 |     fprintf(stdout, "Unable to start wiringPi  
|  
|  
@raspberrypi:~/semina $ sudo ./ledser
```

