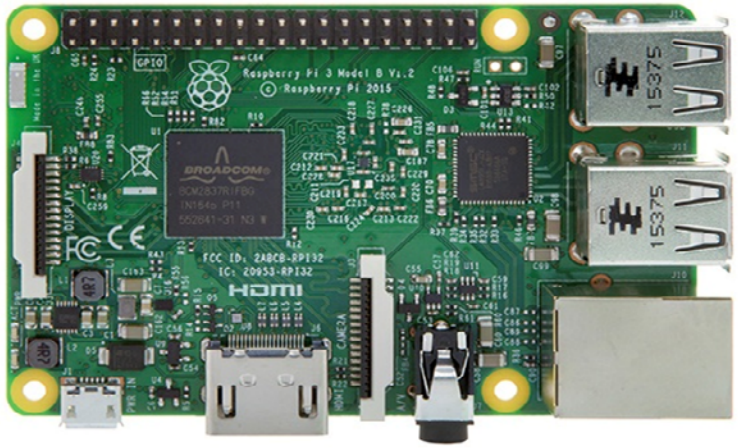
Motor 디바이스 제어

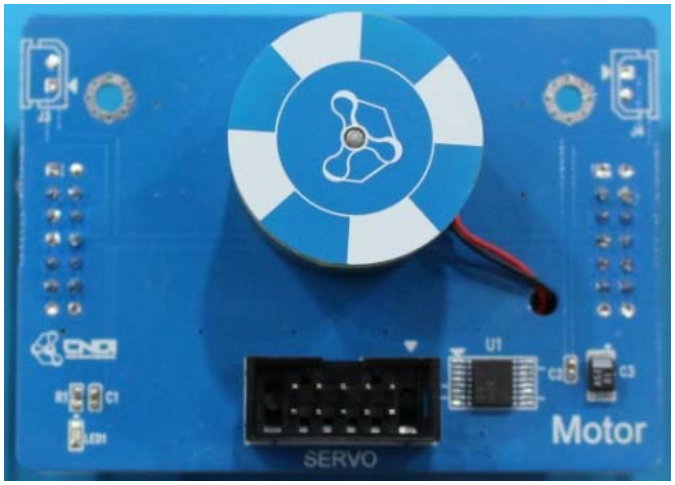
조 진 성
경희대학교 컴퓨터공학과
Mobile & Embedded System Lab.

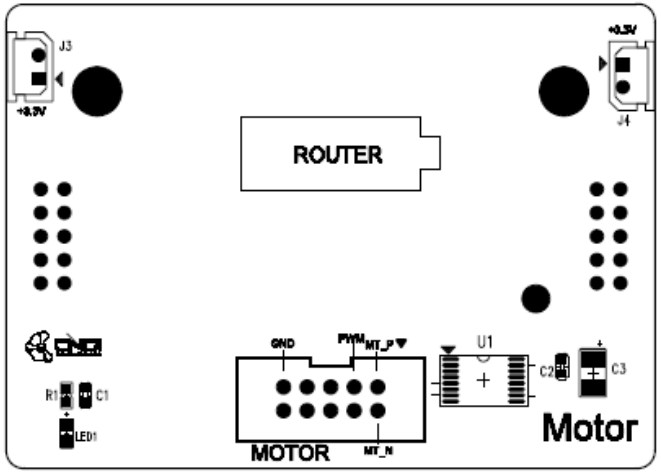
하드웨어 구성

❖ Raspberry Pi 3 Model B



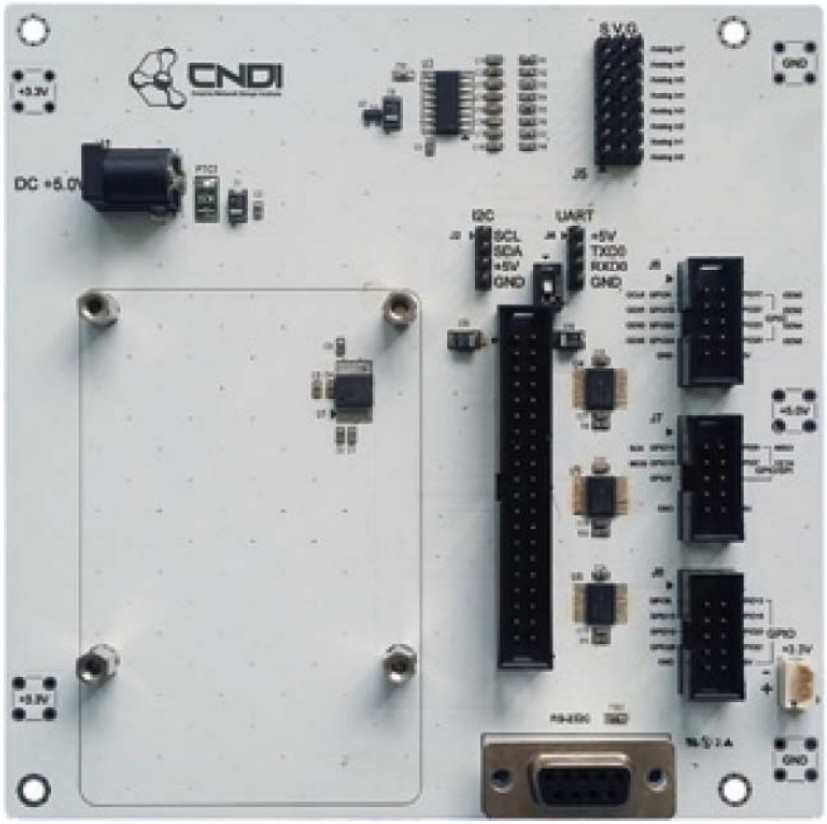
❖ Motor모듈

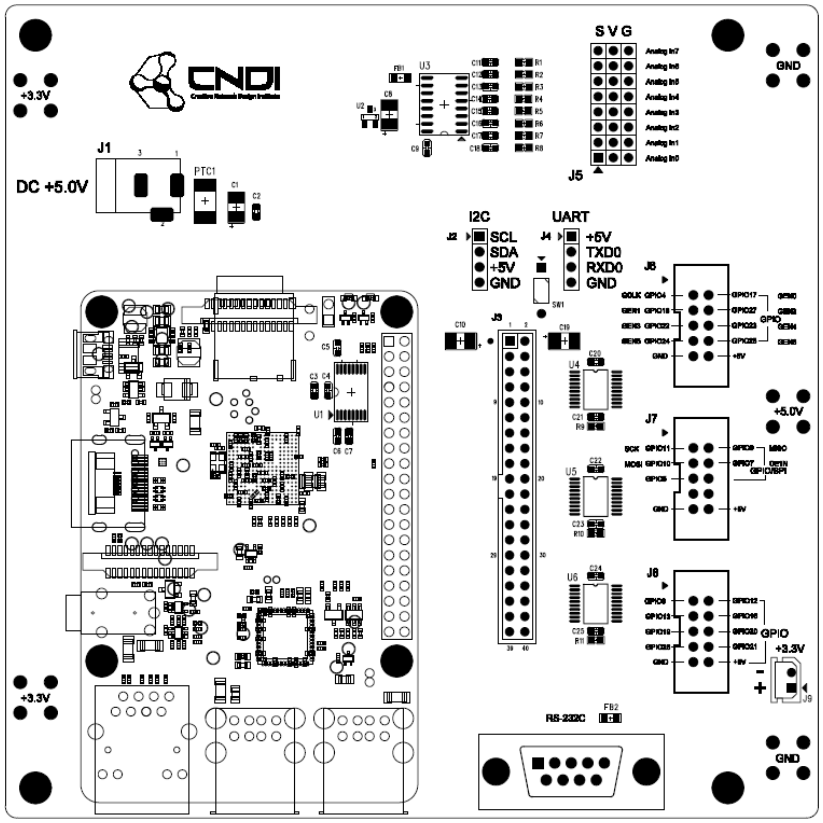




하드웨어 구성

❖ Raspberry Pi Adapter





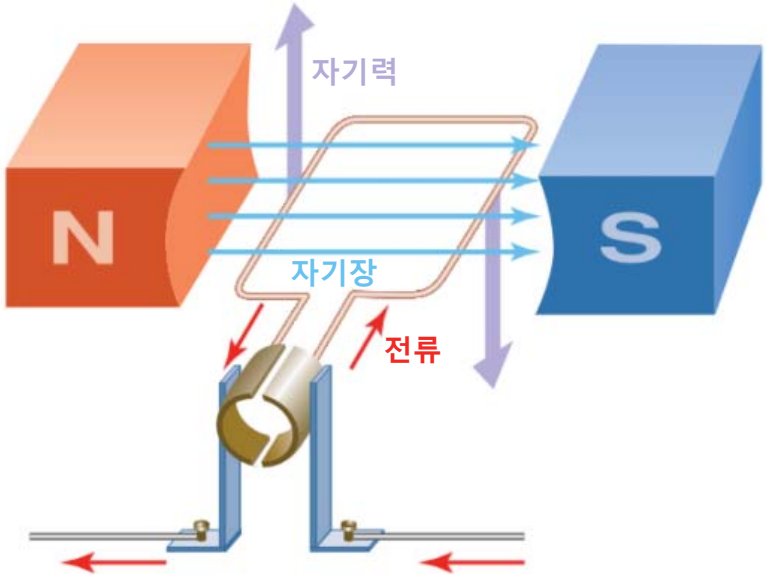
Motor

Motor 란

- 전력(전기적 에너지)을 이용하여 회전운동의 힘(기계적 에너지)을 얻는 기계

DC 모터의 구동 원리

- DC 모터(직류 전동기)는 건전지와 같은 직류 전원으로 동작하는 모터
- 가해주는 전압에 따라서 속도 변화
- 자석의 극이 같으면 서로 밀어내고, 다르면 끌어당기는 성질 이용
- 두 자석과 그 중간에 권선이 위치하고 권선에 전류가 흐르면 자기장이 형성
- 자기장 영역에 전류가 흐르고, 전류와 자기장이 수직이면 자기력이 생기는 원리 이용



Motor

❖ Motor 제어 방법

- 모터 제어를 위해 전원 공급
- 모터의 극성을 바꿔 줌으로써 회전 방향 변경
- 모터의 회로를 살펴보면 각 전극에 신호를 전달할 수 있는 핀 확인 가능
- PWM을 사용하여 모터 속도 제어 가능
- 모터에 전달하는 신호는 GPIO를 사용하며 모터와 연결된 특정 핀에 적절한 신호를 전달

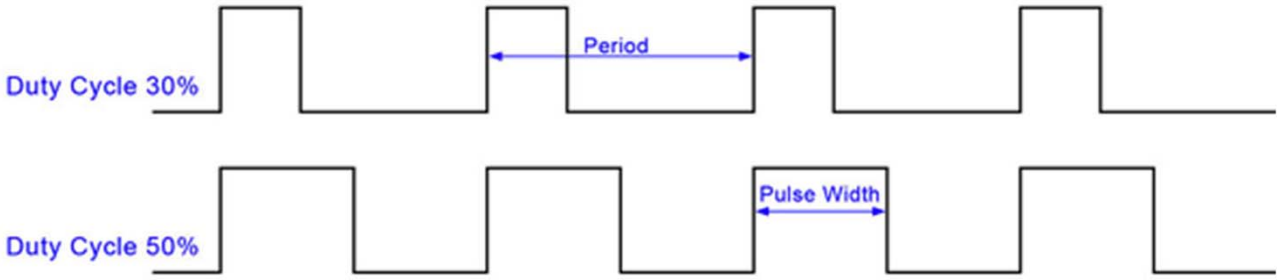
- PWM을 이용한 모터 제어
  - 모터의 방향과 회전 속도를 제어하며 동작하는 방법에 대하여 학습
  - 모터를 제어하는 방법으로 GPIO 사용
  - 라즈베리파이엔 아날로그 핀이 없기 때문에 *Software PWM library* 에서 제공하는 PWM 사용
  - PWM을 통해 Duty Cycle(or Duty Rate)을 30%, 80%로 설정하고, 모터의 회전속도를 제어
  - Motor 모듈 각 핀에 입력되는 신호에 따른 동작

동작	MT_P 핀	MT_N 핀
ON(우회전)	High	Low
ON(좌회전)	Low	High
OFF(정지)	Low	Low

PWM(Pulse Width Modulation)

PWM 이란

- 펄스 폭을 변조하여 디지털 신호를 아날로그 신호처럼 제어하는 방법
  - 아날로그(analog): 연속된 값을 가짐
  - 디지털(digital): 0과 1의 값만 가짐
- 제어 방법
  - 한 주기(Period) 안에서 신호가 1인 상태의 지속시간을 'Pulse Width' 라 함
  - 이 때, 0과 1의 지속 시간 비율을 'Duty Cycle(or Duty Rate)' 이라 함
  - Duty Cycle을 통해 PWM 제어
- Duty Cycle 계산 방법
  - $\text{Duty Cycle} = \frac{\text{Pulse Width}}{\text{한 주기의 값}} \text{ X } 100$
  - 계산 예
    - Pulse Width가 80일 때, Duty Cycle = $\frac{80}{100} \text{ X } 100 = 80\%$



PWM(Pulse Width Modulation)

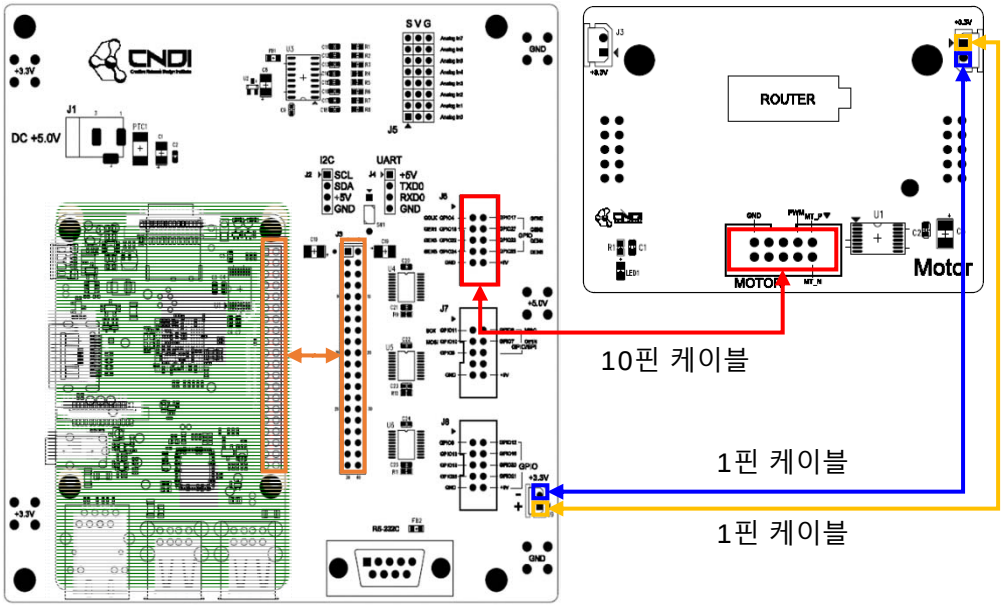
Software PWM Library

- Wiring Pi에 포함된 라이브러리로써 PWM 제어에 사용
  - softPwm.h 사용
- 관련 함수
  - int softPwmCreate(int pin, int initialValue, int pwmRange)
    - 소프트웨어로 제어하는 PWM 핀을 생성하는 함수
    - 라즈베리파이의 모든 GPIO 핀 이용 가능
    - 매개 변수
      - pin: PWM 핀으로 사용할 GPIO 핀
      - initialValue: 초기 Pulse width
      - pwmRange: 한 주기
    - 사용 예) softPwmCreate(4, 64, 255);
      - 한 주기가 255ms인 펄스를 25% Duty Cycle로 생성하고 4번 핀을 통해 제어함
      - Duty Cycle = $\frac{64}{255}$ X $100 = 25\%$
  - void softPwmWrite(int pin, int value)
    - 생성된 PWM 핀을 통해 1의 지속 시간을 변경하여 Duty Cycle 변경
    - 매개 변수
      - pin: 생성된 PWM 핀
      - value: 1의 지속 값

하드웨어 구성

❖ 실습 예제 (1), (2) 위한 하드웨어 구성

- 실습을 위한 Raspberry Pi 3와 LED 결선 방법
  - Raspberry Pi 3와 Raspberry Pi Adapter 보드를 20 x 2 케이블을 이용하여 연결
  - Raspberry Pi Adapter 모듈의 포트 J6(10핀)을 Motor 모듈의 포트 MOTOR(10핀)에 연결
  - Raspberry Pi Adapter 모듈의 핀 +(1핀)를 Motor 모듈의 포트 J4의 3.3V(+) (1핀)에 연결
  - Raspberry Pi Adapter 모듈의 핀 -(1핀)를 Motor 모듈의 포트 J4의 3.3V(+) (1핀) 아래 핀에 연결



실습 예제 (1)

❖ Motor 방향 제어하기

- 목표: Motor가 회전하는 방향을 주기적으로 변경
- 제어 프로그램 작성
  - lab2-5_1.c

```
#include <wiringPi.h> // GPIO Access Library 헤더파일 선언
#include <softPwm.h> // Software PWM library 헤더파일 선언

// Motor 핀 설정
#define MOTOR_MT_N_PIN    17
#define MOTOR_MT_P_PIN    4
// Motor 회전 방향 정의
#define LEFT_ROTATE       1
#define RIGHT_ROTATE      2

// Motor 정지 함수
void MotorStop() {
        softPwmWrite(MOTOR_MT_N_PIN, 0);
        softPwmWrite(MOTOR_MT_P_PIN, 0);
}

// Motor 회전 함수
void MotorControl(int rotate) {
        if (rotate == LEFT_ROTATE) {
                digitalWrite(MOTOR_MT_P_PIN, LOW);
                softPwmWrite(MOTOR_MT_N_PIN, 50);
        }
        else if (rotate == RIGHT_ROTATE) {
                digitalWrite(MOTOR_MT_N_PIN, LOW);
                softPwmWrite(MOTOR_MT_P_PIN, 50);
        }
}
```

```
int main(void)
{
        if (wiringPiSetupGpio() == -1)
                return 1;
        // Motor 핀 출력으로 설정
        pinMode(MOTOR_MT_N_PIN, OUTPUT);
        pinMode(MOTOR_MT_P_PIN, OUTPUT);

        // Motor 핀 PWM 제어 핀으로 설정
        // 주기: 100ms
        softPwmCreate(MOTOR_MT_N_PIN, 0, 100);
        softPwmCreate(MOTOR_MT_P_PIN, 0, 100);

        while (1)
        {
                MotorControl(LEFT_ROTATE); // Motor 왼쪽으로 회전
                delay(2000);
                MotorStop(); // Motor 정지
                delay(1000);

                MotorControl(RIGHT_ROTATE); // Motor 오른쪽으로 회전
                delay(2000);
                MotorStop(); // Motor 정지
                delay(1000);
        }
        return 0;
}
```

실습 예제 (1)

Motor 방향 제어하기

- 제어 프로그램 컴파일
  - gcc –o lab2-5_1 lab2-5_1.c –lwiringPi
- 제어 프로그램 실행
  - ./lab2-5_1
- 동작 화면

실습 예제 (2)

❖ Motor 속도 제어하기

- 목표: Motor의 속도를 주기적으로 변경
- 제어 프로그램 작성
  - lab2-5_2.c

```
#include <wiringPi.h> // GPIO Access Library 헤더파일 선언
#include <softPwm.h> // Software PWM library 헤더파일 선언

// Motor 핀 설정
#define MOTOR_MT_N_PIN    17
#define MOTOR_MT_P_PIN    4

// Motor 회전 방향 정의
#define LEFT_ROTATE       1
#define RIGHT_ROTATE      2

// Motor 정지 함수
void MotorStop()
{
        softPwmWrite(MOTOR_MT_N_PIN, 0);
        softPwmWrite(MOTOR_MT_P_PIN, 0);
}

// Motor 속도 조절 함수
void MotorControl(int speed)
{
        digitalWrite(MOTOR_MT_P_PIN, LOW);
        softPwmWrite(MOTOR_MT_N_PIN, speed);
}
```

```
int main(void)
{
        if (wiringPiSetupGpio() == -1)
                return 1;
        // Motor 핀 출력으로 설정
        pinMode(MOTOR_MT_N_PIN, OUTPUT);
        pinMode(MOTOR_MT_P_PIN, OUTPUT);

        // Motor 핀 PWM 제어 핀으로 설정
        // 주기: 100ms
        softPwmCreate(MOTOR_MT_N_PIN, 0, 100);
        softPwmCreate(MOTOR_MT_P_PIN, 0, 100);

        while (1) {
                MotorControl(25); // Duty Cycle 25% 동작
                delay(2000);
                MotorStop(); // Motor 정지
                delay(2000);

                MotorControl(50); // Duty Cycle 50% 동작
                delay(2000);
                MotorStop(); // Motor 정지
                delay(2000);

                MotorControl(75); // Duty Cycle 75% 동작
                delay(2000);
                MotorStop(); // Motor 정지
                delay(2000);

        }
        return 0;
}
```

실습 예제 (2)

Motor 속도 제어하기

- 제어 프로그램 컴파일
  - gcc –o lab2-5_2 lab2-5_2.c –lwiringPi
- 제어 프로그램 실행
  - ./lab2-5_2
- 동작 화면

실습 과제 (1)

Motor 속도 및 방향 동시 제어하기

- 실습 예제 (1), (2)를 참고하여 아래 조건에 맞게 동작
  - 초기 Pulse Width: 0 / 주기: 128ms
    - softPwmCreate() 를 이용
  - 유지 시간 / 회전 방향 / Duty Cycle
    - 2초 / 왼쪽 / 25%
    - 2초 / 왼쪽 / 50%
    - 2초 / 왼쪽 / 75%
    - 2초 / 왼쪽 / 100%
    - 2초 / 오른쪽 / 100%
    - 2초 / 오른쪽 / 75%
    - 2초 / 오른쪽 / 50%
    - 2초 / 오른쪽 / 25%
  - 유의 사항
    - 각 동작 간 유지 시간 이후 2초간 정지
    - 주기가 실습 예제와 달리 128ms 임을 주의

실습 과제 (2)

PWM을 이용하여 LED 제어하기

- LED 한 개를 PWM을 이용하여 다음 조건에 맞게 동작
  - 초기 Pulse Width: 0 / 주기: 128ms
    - softPwmCreate() 를 이용
    - Duty Cycle(밝기)이 0%에서 부터 100%까지 점점 밝아지게 변화
    - Duty Cycle(밝기)이 100%에서 부터 0%까지 점점 어두워지게 변화
    - 위 과정 무한 반복

Q & A

http://mesl.khu.ac.kr