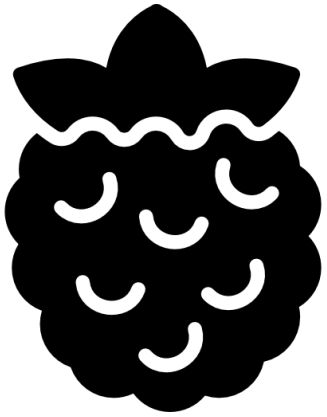
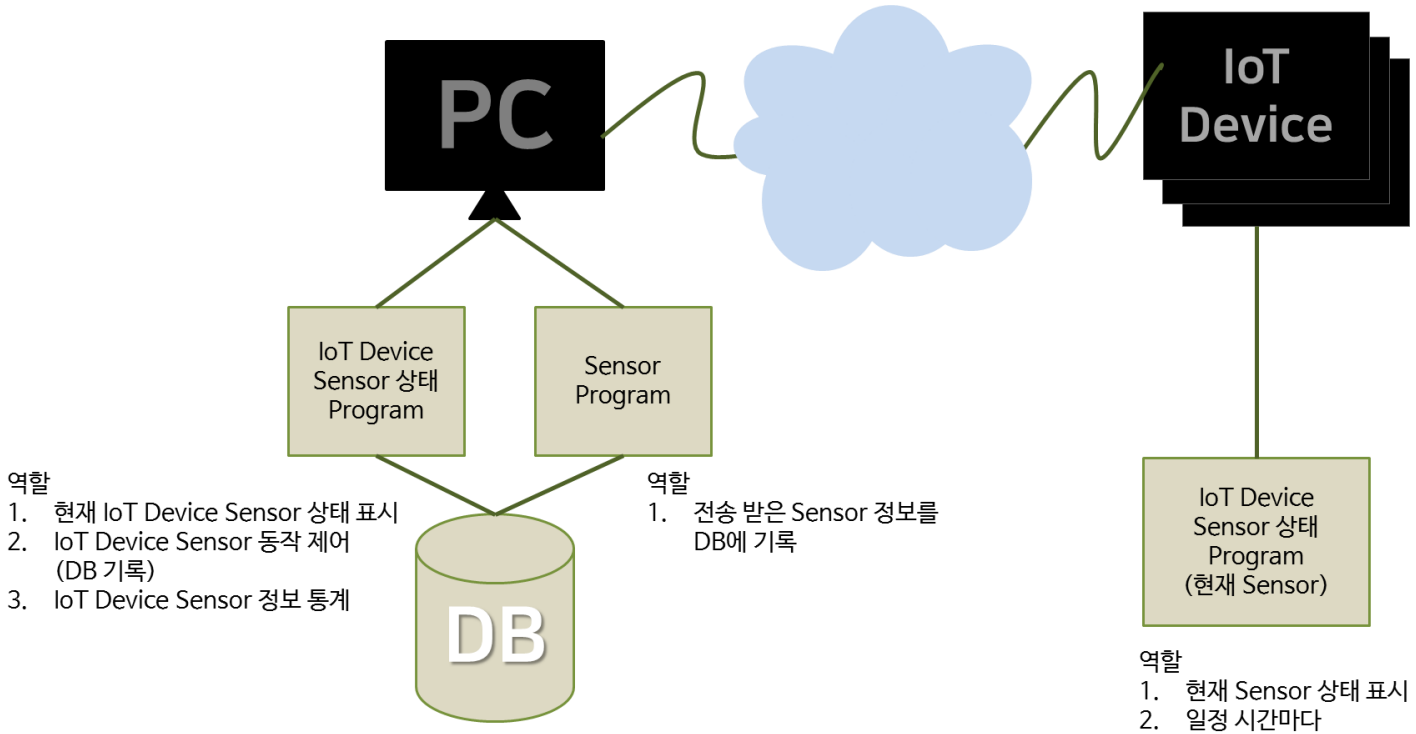


RaspberryPie를 이용한 IoT 환경 구축 Team Project

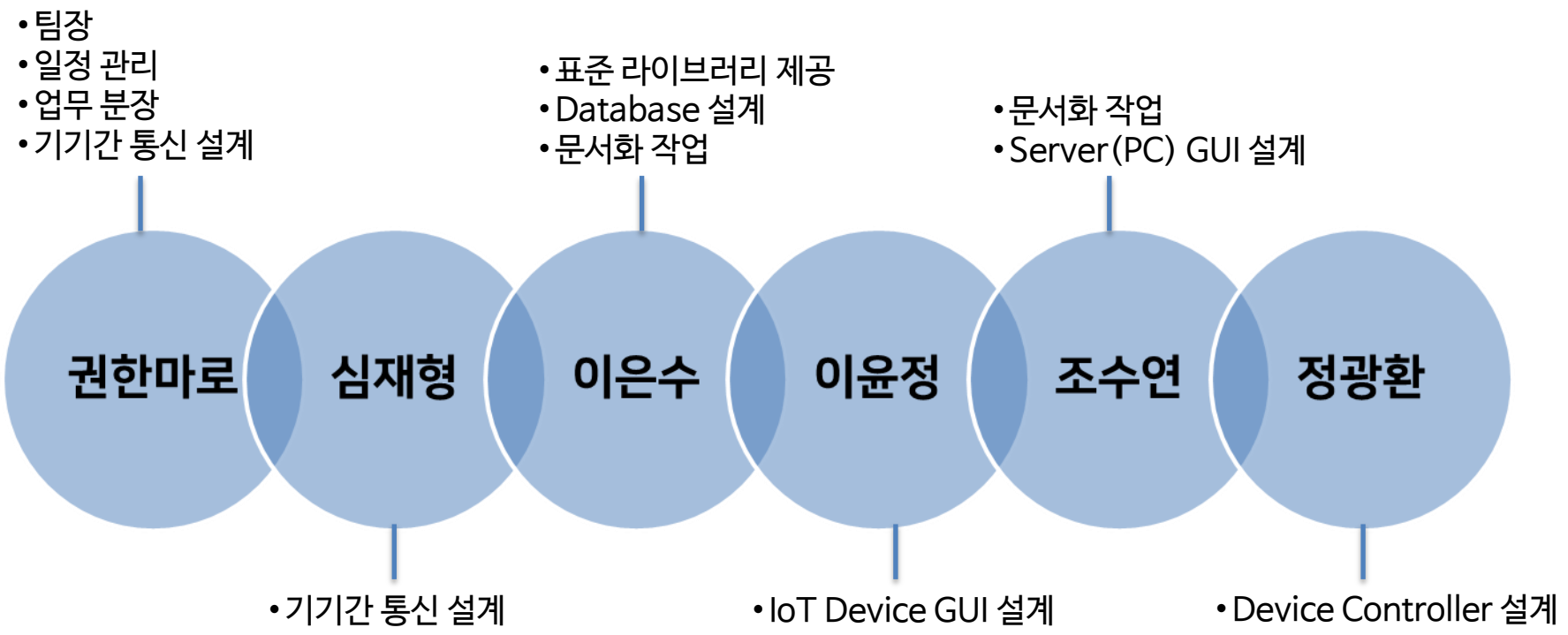
1조 권한마로 심재형 이윤정 이은수 정광환 조수연

개요

- PC – IoT Device 간 통신, 제어
- Device에서 전달 받은 Sensor Data를 PC Database에서 관리



담당


• 팀장
• 일정 관리
• 업무 분장
• 기기간 통신 설계
권한마로
• 기기간 통신 설계
심재형
• 표준 라이브러리 제공
• Database 설계
• 문서화 작업
이은수
• IoT Device GUI 설계
이윤정
• 문서화 작업
• Server(PC) GUI 설계
조수연
• Device Controller 설계
정광환

IoT Device Sensor

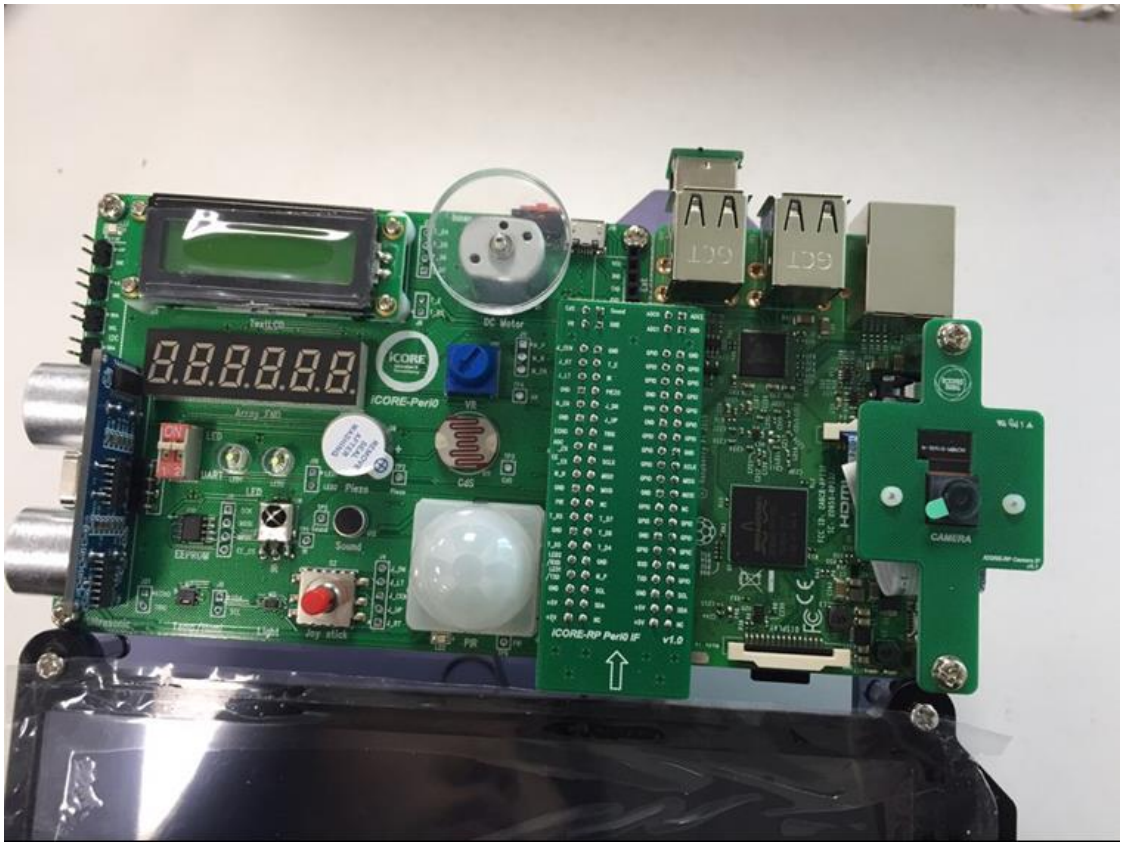
사용 Sensor 총 10개

GPIO – 7개

- LED
- Motor
- Jog Switch
- CLCD
- 초음파 거리센서
- Piezo
- PIR

I2C – 3개

- 조도 감지 센서
- 온습도 센서
- FND



Database schema

테이블 (10)	
CharacterLCD	CREATE TABLE "CharacterLCD" (`eventID` INTEGER, `time_stamp` TEXT, `line1` TEXT, `line2` TEXT, PRIMARY KEY(`eventID`))
DCMotor	CREATE TABLE "DCMotor" (`eventID` INTEGER, `time_stamp` TEXT, `state` TEXT, `direction` TEXT, `speed` INTEGER, PRIMARY KEY(`eventID`))
FND	CREATE TABLE "FND" (`eventID` INTEGER, `time_stamp` TEXT, `number` INTEGER, PRIMARY KEY(`eventID`))
JogSwitch	CREATE TABLE "JogSwitch" (`eventID` INTEGER, `time_stamp` TEXT, `direction` TEXT, PRIMARY KEY(`eventID`))
LED	CREATE TABLE "LED" (`eventID` INTEGER, `time_stamp` TEXT, `LED1` TEXT, `LED2` TEXT, `LED1Lux` INTEGER, `LED2Lux` INTEGER, PRIMARY KEY(`eventID`))
LightSensor	CREATE TABLE "LightSensor" (`eventID` INTEGER, `time_stamp` TEXT, `light_value` INTEGER, PRIMARY KEY(`eventID`))
PIR	CREATE TABLE "PIR" (`eventID` INTEGER, `time_stamp` TEXT, `state` TEXT, PRIMARY KEY(`eventID`))
Piezo	CREATE TABLE "Piezo" (`eventID` INTEGER, `time_stamp` TEXT, `state` TEXT, PRIMARY KEY(`eventID`))
TempHumid	CREATE TABLE "TempHumid" (`eventID` INTEGER, `time_stamp` TEXT, `temperature` REAL, `humidity` REAL, PRIMARY KEY(`eventID`))
Ultrasonic	CREATE TABLE "Ultrasonic" (`eventID` INTEGER, `time_stamp` TEXT, `distance` REAL, PRIMARY KEY(`eventID`))

- sensor 별 테이블 관리
- 각 테이블 별 Primary key는 고유한 정보인 시간 eventID 사용
- 시간 형식은 '2017-10-11 01:49:22' 의 형식 사용

```
>>> from time import localtime, strftime
>>> strftime("%Y-%m-%d %I:%M:%S", localtime() )
'2017-10-11 01:54:24'
```

- PC - Device간 통신 간격
  - 센서 별 이벤트 발생 시 데이터 전송(이벤트 발생 조건은 후반부 명시)

Device to PC Data type

테이블 (10)	
CharacterLCD	CREATE TABLE "CharacterLCD" (`eventID` INTEGER, `time_stamp` TEXT, `line1` TEXT, `line2` TEXT, PRIMARY KEY(`eventID`))
DCMotor	CREATE TABLE "DCMotor" (`eventID` INTEGER, `time_stamp` TEXT, `state` TEXT, `direction` TEXT, `speed` INTEGER, PRIMARY KEY(`eventID`))
FND	CREATE TABLE "FND" (`eventID` INTEGER, `time_stamp` TEXT, `number` INTEGER, PRIMARY KEY(`eventID`))
JogSwitch	CREATE TABLE "JogSwitch" (`eventID` INTEGER, `time_stamp` TEXT, `direction` TEXT, PRIMARY KEY(`eventID`))
LED	CREATE TABLE "LED" (`eventID` INTEGER, `time_stamp` TEXT, `LED1` TEXT, `LED2` TEXT, `LED1Lux` INTEGER, `LED2Lux` INTEGER, PRIMARY KEY(`eventID`))
LightSensor	CREATE TABLE "LightSensor" (`eventID` INTEGER, `time_stamp` TEXT, `light_value` INTEGER, PRIMARY KEY(`eventID`))
PIR	CREATE TABLE "PIR" (`eventID` INTEGER, `time_stamp` TEXT, `state` TEXT, PRIMARY KEY(`eventID`))
Piezo	CREATE TABLE "Piezo" (`eventID` INTEGER, `time_stamp` TEXT, `state` TEXT, PRIMARY KEY(`eventID`))
TempHumid	CREATE TABLE "TempHumid" (`eventID` INTEGER, `time_stamp` TEXT, `temperature` REAL, `humidity` REAL, PRIMARY KEY(`eventID`))
Ultrasonic	CREATE TABLE "Ultrasonic" (`eventID` INTEGER, `time_stamp` TEXT, `distance` REAL, PRIMARY KEY(`eventID`))

TEXT(문자열)

- time_stamp
- state(on/off)
- direction
  (cw/ccw, U/D/L/R/C)
- CLCD 출력 값

INTEGER(정수형)

- eventID
- LED1lux,LED2lux
- speed
- number(FND 출력 값)
- light_value

REAL(소수형)

- temperature
- humidity
- distance

Device to PC return information

분류	Sensor	공통 정보	개별정보	
GPIO	LED	현재 시간, 센서명, 상태(on/off)	LED1(on/off), LED2(on/off)	
	Motor		on/off,방향(cw, ccw), 속도(0~100)	
	Jog Switch		방향(up, down, left, right, center)	
	CLCD		출력한 문자열	
	UltraSonic		거리(cm)	
	Piezo		on/off	
	PIR		감지 여부 (O/X)	
I2C	Light Sensor		사용 가능 여부	조도 값(light_val)
	FND			출력한 숫자 데이터
	Temp/Humid			온도, 습도

Event triggers for each sensors

분 류	Sensor	event trigger
GPIO	LED	각 LED 별 작동 상태 변화(on/off) 및 조도 변화
	Motor	작동 상태(on/off), 방향(cw, ccw), 속도(duty) 변화
	Jog Switch	JogSwitch 작동
	CLCD	문자열 출력
	UltraSonic	10cm 이하로 접근
	Piezo	Piezo 작동
	PIR	적외선 포착
I2C	Light Sensor	너무 어둡거나 너무 밝을 시(기준 dark 〈 200lux, bright 〉 1000lux)
	FND	정수 출력
	Temp/Humid	온/습도 값 변화

PC to Device send information

분류	Sensor	공통정보	개별정보
GPIO	LED	현재 시간, 센서명	LED 1 on/off, 밝기(duty) Control
			LED 2 on/off, 밝기(duty) Control
	Motor		on/off, 방향(cw, ccw), 속도(duty) Control
	CLCD		textbox에 입력한 text 전송
	Piezo		on/off Control
I2C	FND		textbox에 입력한 text 전송

Client display

Label

LED LED1 : on/off, 밝기 / LED2 : on/ off, 밝기	**Piezo** On/ off
Motor on/off, 방향, 스피드(듀티비 0~100)	**PIR** 감지여부
JogSwitch 방향	**Light Sensor** 밝기 값
CLCD 입력한 텍스트	**FND** 입력한 숫자 값
UltraSonic 거리값	**Temp/Humid** 온/습도 값

Server display

Label | Button | textbox

LED

LED1: on/off ,밝기 | on/off | ▲ | ▼

LED2: on/off ,밝기 | on/off | ▲ | ▼

Piezo

On/ off

Motor

on/off, 방향, 스피드(듀티비 0~100)

Up | Down | on/off | cw/ccw

PIR

PIR : 감지여부

Jog Switch

Jog : 방향

Light Sensor

Light Sensor : 밝기 값

CLCD

입력한 텍스트 : 'hello'

텍스트를 입력하세요 | OK

FND

입력한 숫자 : '0000'

숫자를 입력하세요 | OK

UltraSonic

UltraSonic : 거리 값

Temp/Humid

Temp/Humid : 온/습도 값

Library

공통 라이브러리는 10개의 센서를 GPIO와 I2C로 나누어
각 센서 별 공통 기능을 묶어 Base class를 만든 후 상속해 설계하였다.

〈GPIO Base Class〉

```
class GPIO_Base:
    def __init__( self, pin ):
        GPIO.setmode( GPIO.BCM )
        GPIO.setwarnings( False )
        self.pin = pin
        self.pwm_start = 0

    def set_output( self, pin ):
        GPIO.setup( pin, GPIO.OUT )

    def set_input( self, pin ):
        GPIO.setup( pin, GPIO.IN )

    def set_PWM( self, pin, pwm_start):
        self.pwm_start = pwm_start
        self.PWM = GPIO.PWM( pin, self.pwm_start )
        return self.PWM
```

〈I2C Base Class〉

```
class I2C_Base:
    DELAY = 0.26

    def __init__( self, addr ):
        self.addr = addr
        self.bus = smbus.SMBus( 1 )
        self.data = [ 0, 0 ]
        self.val = 0

    def readData( self ):
        for i in range(2):
            self.data[i] = self.bus.read_byte( self.addr )
        self.val = ( self.data[0] << 8 ) | self.data[1]
        return self.val

    def writeData( self, contents ):
        self.bus.write_byte( self.addr, contents )
        time.sleep( self.DELAY )
```

Library

1.LED

```
led1_on()                    - LED1 on
led1_off()                   - LED1 off
led2_on()                    - LED2 on
led2_off()                   - LED2 off
leds_on()                    - LED1, LED2 on
leds_off()                   - LED1, LED2 off
led1_adjust('up'/'down')     - change LED1 PWM
led2_adjust('up'/'down')     - change LED2 PWM
```

2.DCMotor

```
cw()             - set clockwise
ccw()            - set counterclockwise
stop()           - stop motor
speed_up()       - speed up
speed_down()     - speed down
DCmotor_jog()    - up( speed up ), down( speed down )
                   left( clockwise ), right( counterclockwise )
                   center( stop )
```

3.JogSwitch

```
check_jog()   - return directions( 'up', 'down', 'left', 'right', 'center' )
```

4.CLCD

```
lcd_string( message )     - present message to CharacterLCD
```

5.Ultrasonic

```
check_distance()     - check distance and return state
                       (sense <= 10cm --> 'too_close')
                       (sense > 10cm  --> 'safe')
```

6.Piezo

```
play_sound( scale, time )     - play scale for time
-scale 'DO', 'RE', 'MI', 'FA', 'SOL', 'LA', 'SI', 'DO2'
lalala()                      - play song
```

7.PIR

```
check_move()      - check movement and return count
```

8.LightSensor

```
check_light()     - check light and return illumination
```

9.FND

```
printFND( place, number )     - print number at specific place
FNDprint( number )    - print number
```

10.TempHumid

```
check_temp()      - check temperature and return temperature
check_humid()     - check humidity and return humidity
```