



# AI with IoT

Lecture Introduction

# 최신 기술 동향

## 4차 산업혁명 주요 기술 (Key Enablers)



5세대 무선 통신

사물인터넷  
(Internet of Things)



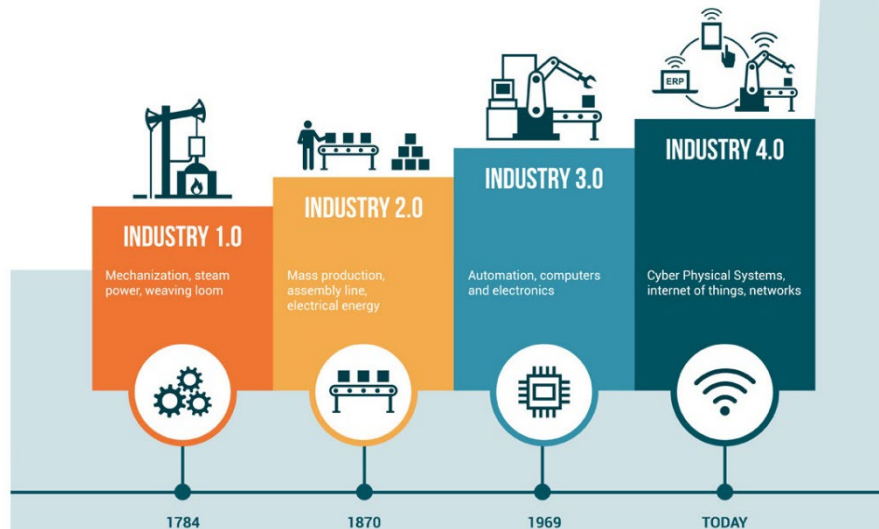
빅 데이터

인공지능  
(Artificial Intelligence)



## INDUSTRIAL REVOLUTION

TRANSFORMING INDUSTRIES AND INNOVATION



# 기술 간 융합



**“Driving Smart Home Product Value Through A.I Applications”**

- CIO Business Technology Solutions

개인 맞춤형 자동화

예측 기반 환경 유지

지능형 기기 연동

# 실습 내용 개요



## 실습 컨텐츠

### MODLINK 기반 IoT 센서 장치 설계/구현 실습

- Arduino 기반 모듈화 장치 사용 [Vitcon Smart Farm Kit]
- IoT 시스템에서의 End-device 구현 실습
- End-device 간 상호 연동 사례 구현

### 라즈베리 파이 기반 스마트 홈-AI 구현 실습

- 라즈베리 파이를 이용한 스마트 홈 컨트롤러 구현 실습 [mCube IoT-AI Kit]
- IoT 기반 스마트 홈 제어를 위한 인공지능 API 활용 실습
- AI 기반 음성 인식 (Google Assistant) 및 Vision 인식 (Open CV – face recognition) 실습

IoT End-device 구현

IoT 시스템 구현

AI API 실습

IoT-AI 통합 제어





# AI with IoT

## PART I. MODLINK BASIC

# IoT 스마트 팜 KIT



## 스마트 팜 개요

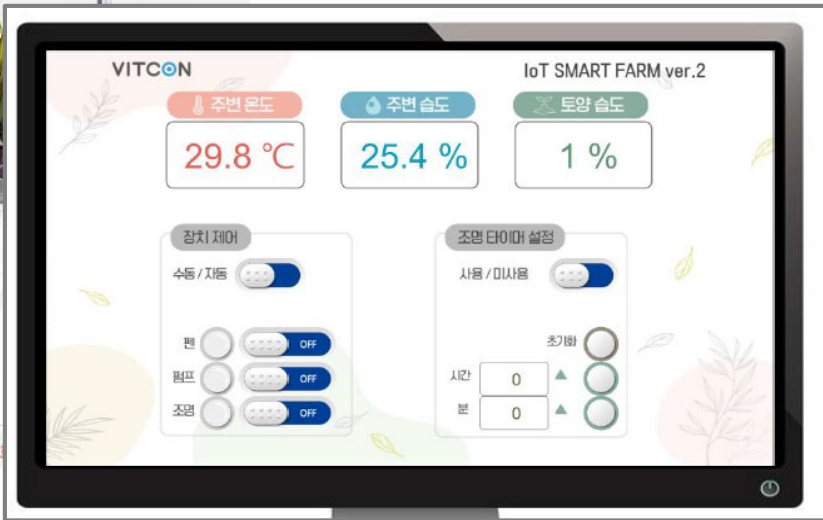
스마트 팜 내부 온·습도와 토양의 습도를 측정하여 그 값을 OLED와 PC/스마트 폰으로 모니터링이 가능합니다.

식물 성장을 위한 LED조명, DC펌프, DC팬을 자동제어 또는 스마트 폰을 사용하여 원격 제어 할 수 있습니다.



## 주요 기능

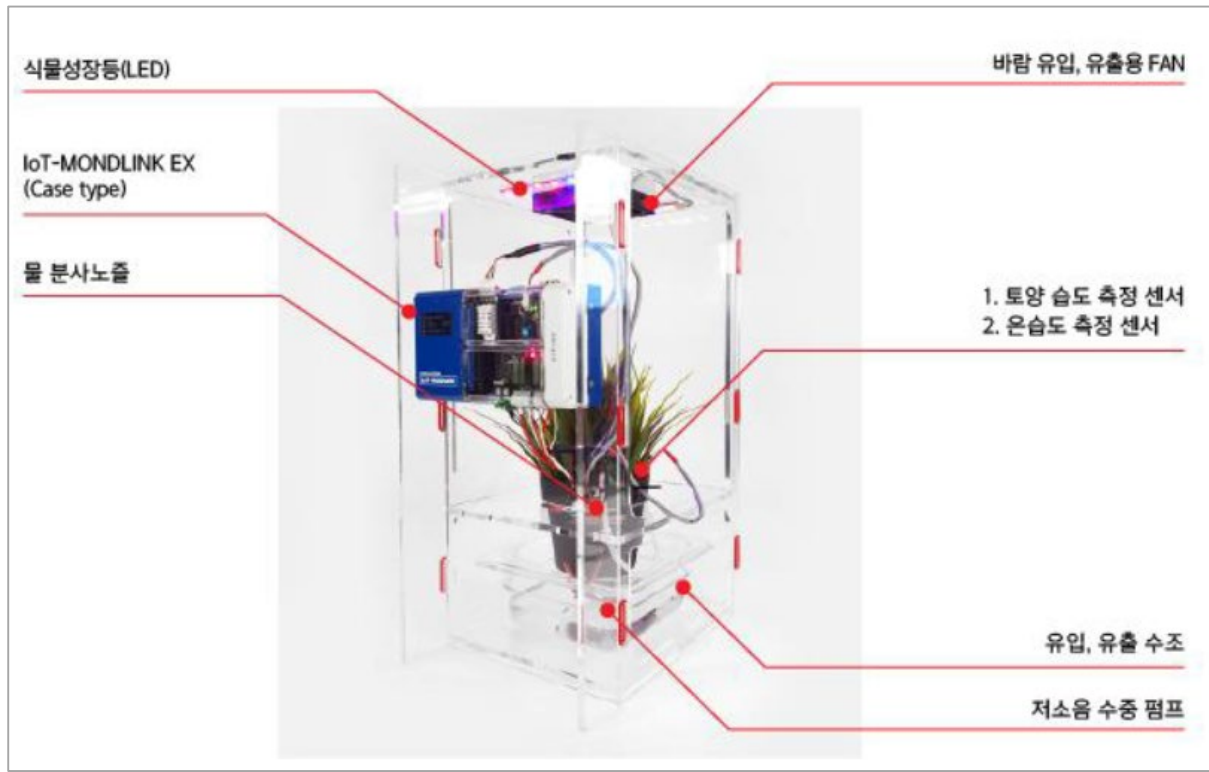
- 식물 성장을 위한 청색/적색 LED 조명
- USB-LINK를 통한 코드 업로드
- DC FAN을 통한 환기, 온도 조절
- LED 조명 제어를 위한 타이머



# IoT 스마트 팜 KIT



## 스마트 팜 KIT 구성

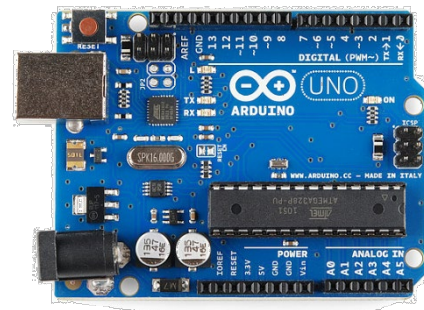


# 아두이노란?

## Arduino

오픈소스 기반 단일보드(Single-board) 마이크로컨트롤러(MCU)

- 하드웨어, 소프트웨어 모두 오픈소스 기반
- 누구나 쉽게 제작, 수정, 배포가 가능
- 시리얼 통신(Serial Communication)을 사용해 PC와 비트 단위로 데이터를 주고 받음
- 배선이 복잡한 특성: 동작 실패 가능성이 높음
- 회로 설계가 필요한 특성: 코딩보다 배선과 테스트에 많은 시간을 소요





# 모드링크란?

## MODLINK

C언어로 간편하게 사용할 수 있는 아두이노 기반 조립식 컨트롤러

- 원하는 기능의 링크 모듈을 활용
- 레고처럼 조립하여 쉽고 빠르게 하드웨어 구성  
: 배선이나 회로 설계에 시간이 필요하지 않는 장점
- 제조사(VITCON) 서버를 활용  
: IoT 구현 및 스마트폰을 사용한 원격 제어와 모니터링 가능



COMPLEX!



SIMPLE

쉽고 간편한 모듈형 컨트롤러



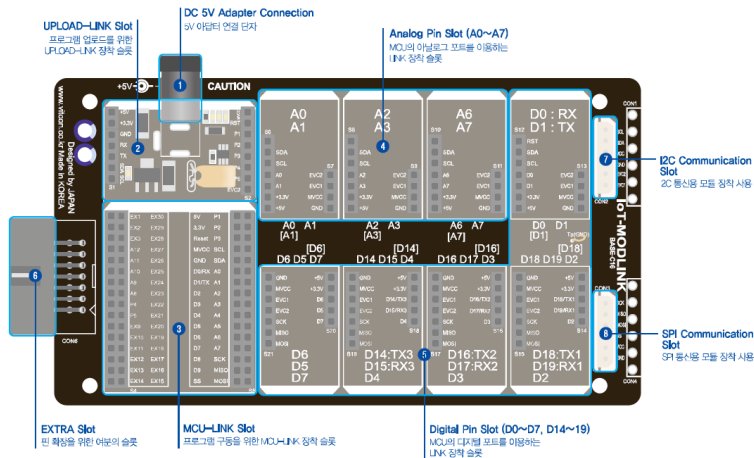
IoT SMART FARM KIT

# 모드링크 구성 개요

## MODLINK 키트 구성

임베디드 시스템에서 사용할 수 있는 여러 장치들을 모듈화한 개발 보드

- 복잡한 회로 구성이 필요하지 않음
- 모듈화된 장치를 결합하여 실습 및 시운전이 가능한 보드
- Arduino를 기반으로 제작



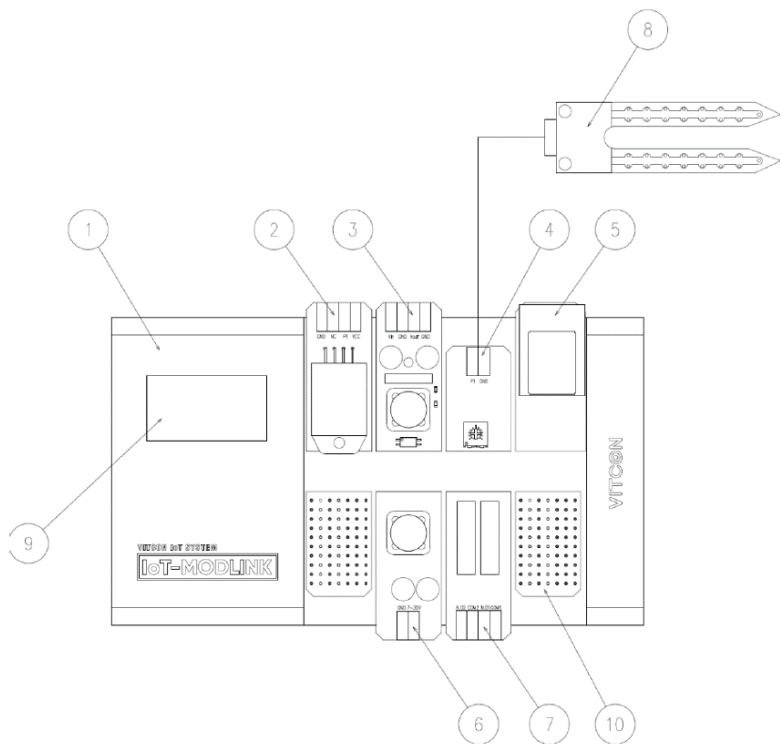
## BASE-C16 보드

- IoT-MODLINK의 메인보드
- UPLOADER-LINK 포함 (프로그램 업로드 목적)
- MEGA-LINK 장착 가능 (Arduino MEGA 내장)
- 측면 단자는 확장용

# 모드링크 구성 개요



## MODLINK 모듈 개요



### 1) IoT-MODLINK-EX

- BASE-C16 보드 기반 IoT 모드링크

### 2) DHT22-LINK

- 온/습도 측정 모듈 링크
- 일반 DHT센서보다 높은 정확도

### 3) EVC-ADJ-LINK

- PWM 제어로 출력 전압 조절
- 3.3 ~ 12 VDC까지 조절

### 4) MOISTURE DETECT-LINK

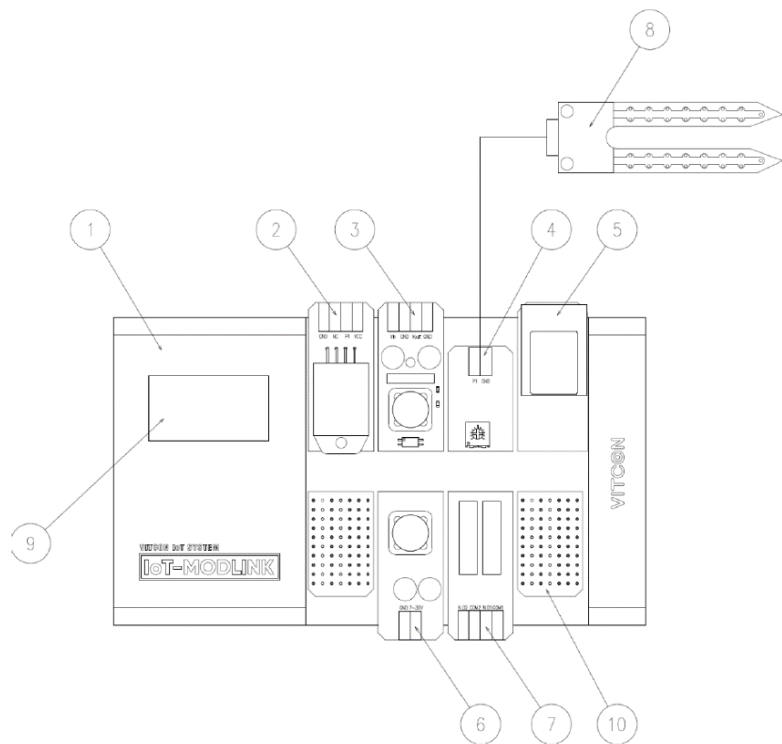
- 수분센서와 함께 사용
- 센서로부터 아날로그/디지털 값 수신

### 5) WiFi-LINK

- 와이파이 무선통신 모듈 링크

# 모드링크 구성 개요

## MODLINK 모듈 개요



### 6) EVC-LINK

- BASE 보드에 외부전원 공급
- 외부전원 이용 모듈과 함께 사용

### 7) RELAY-LINK

- 릴레이가 포함된 모듈 링크
- 250V/5A까지 제어 가능
- 접점의 ON/OFF 제어 용도

### 8) SOIL-LINK

- 토양의 습도 측정
- MOISTURE DETECT-LINK와 사용

### 9) OLED

- 0.96인치 OLED 디스플레이

### 10) PROTO-LINK (x2)

- 직접 회로 구성으로 링크 제작



# 모드링크 개발 환경 구축



## Arduino IDE 설치

### 아두이노 스케치란

- C언어 기반 프로그램 작성 및 컴파일/업로드
- 모드링크 동작을 위한 프로그램을 C언어 코딩을 위한 프로그램
- [www.arduino.cc](http://www.arduino.cc) 접속 후 소프트웨어 – 다운로드 (사용하고 있는 OS 환경 선택)





# 모드링크 개발 환경 구축



## Arduino IDE

**# 톨**  
보드의 종류와 보드가 연결된 포트 설정 가능

**# 컴파일(확인)**  
작성한 코드의 이상 유무 확인  
사람의 언어로 작성한 소스코드를  
컴퓨터 언어로 변환

**# 업로드(실행)**  
코드 확인 후 이상이 없을 시  
모드링크에 코드를 보내는 기능

**# 시리얼 모니터**  
컴퓨터와 모드링크 간에 시리얼 통신을 통해  
모드링크의 상태를 모니터로 확인 가능

**# 코드 입력 창**  
명령어를 입력하여 원하는 프로그램을  
만들기 위한 스크린

**# 콘솔**  
프로그램 상태 확인

**# 시리얼 포트**

**# 보드**

# 모드링크 개발 환경 구축

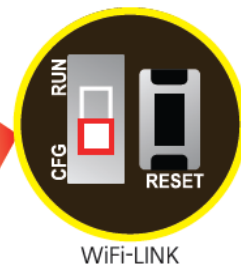
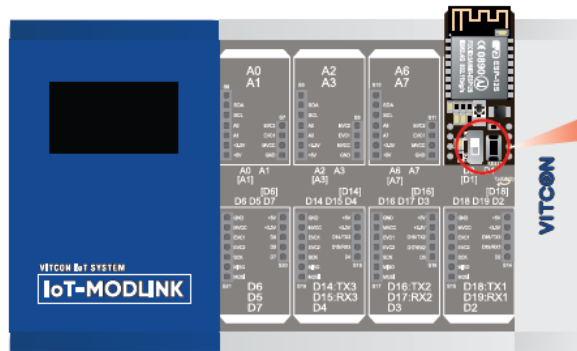


## MODLINK 스위치 선택

업로드 전, 케이스 스위치가 UPLOAD 방향인지 확인

업로드 전, WiFi-LINK의 스위치가 CFG로 되어 있는지 확인

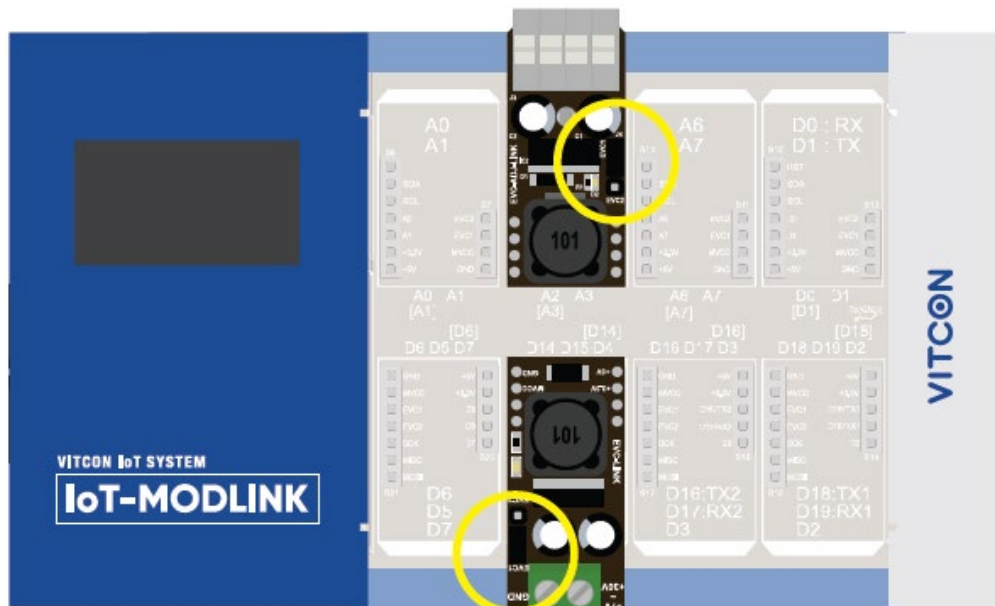
- WiFi-LINK 모듈의 사용법은 2일차 강의에서 진행



## A top-down view of a person's hands working on a laptop. The person has red-painted fingernails and is wearing a light blue shirt. They are typing on the laptop keyboard. To the left of the laptop, there are several sheets of paper, some with text and diagrams. Above the papers are four pens: orange, yellow, pink, and blue. A small white cup of coffee with a spoon is on a saucer to the right of the laptop. The background is a dark, textured surface.

## MODLINK 링크 점퍼캡 위치 확인

EVC-ADJ-LINK와 EVC-LINK의 점퍼캡이 EVC1 방향으로 꽂혀 있도록 조정



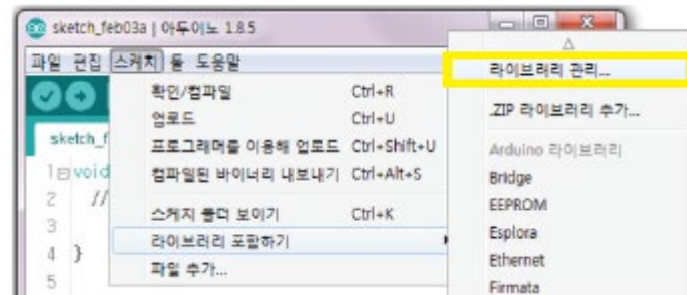
# 모드링크 개발 환경 구축



## Arduino 라이브러리 설치하기

Arduino IDE에서 '스케치' - '라이브러리 포함하기' - '라이브러리 관리' 선택

- 라이브러리 매니저에서 다음의 항목을 설치
- VitconCommon by vitcon
- VitconIoT by vitcon
- DHT sensor library by Adafruit
- U8g2 by oliver



### 별도 라이브러리 포함하기

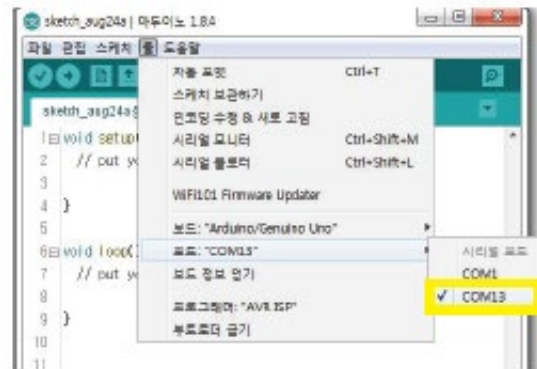
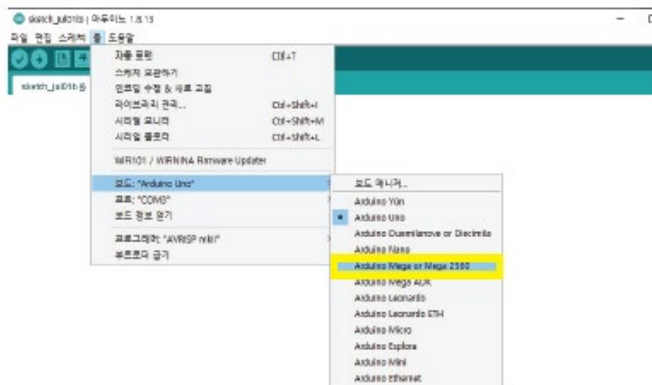
- '파일' - '환경설정' 에서 아두이노 사용자 라이브러리 경로 확인
- 사용자 라이브러리 경로에 SoftPWM 폴더와 Adafruit\_Sensor-master 폴더 복사 [별도 제공]
- Github.com/monetIoT/IoT 접속 후, Arduino - libraries 내 파일 다운로드

# 모드링크 개발 환경 구축



## MODLINK – PC 연결

1. 준비된 MODLINK를 USB업로드 케이블로 PC와 연결
2. [툴] - [보드] - [Arduino/Genuino MEGA/MEGA 2560] 선택
3. [툴] - [포트] - [COM X] 선택 (COM 1제외)





# 아두이노 프로그램 작성



## 아두이노 코드의 구조

```
int var = 1;
```

Define Variables

Declare the variables to be used in the program

```
void setup()  
{
```

Initialization

Setup function is run once, when the microcontroller boots up or resets.

```
void loop()  
{
```

Eternal loop

After setup function the processor moves to run code inside the loop function.

# 모드링크 개발 환경 테스트



## 아두이노 스케치 작성

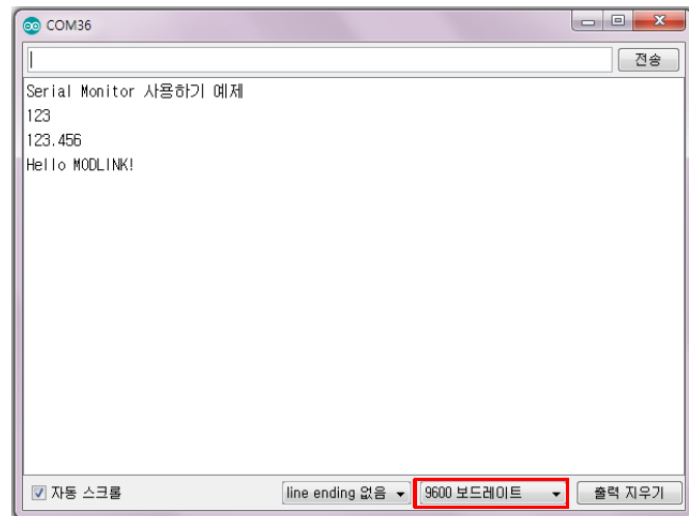
```
int a = 123;
float b = 123.456;
String c = "Hello MODLINK!";

void setup() {
    Serial.begin(9600);

    Serial.println("Serial Monitor 사용하기 예제");
    Serial.println(a);
    Serial.println(b);
    Serial.println(c);
}

void loop() {
}
```

### < 업로드 후 결과 확인 >



# Example I - DHT22



## 전처리문

작성한 프로그램(코드)는 MODLINK가 이해하도록 기계어로 변환

- 기계어로 변환되는 과정을 컴파일(Compile)이라 함
- 전처리문은 코드가 컴파일 되기 전에 먼저 처리되는 문장을 의미

- 파일 처리를 위한 전처리문

`#include "DHT.h"` : 해당 코드파일 위치에 DHT.h 파일을 포함

- 형태 정의를 위한 전처리문

`#define DHTPIN A1` : 컴파일 하기 전 코드에 DHTPIN이라고 작성한 부분을 모두 A1으로 치환

## 온/습도 측정 준비



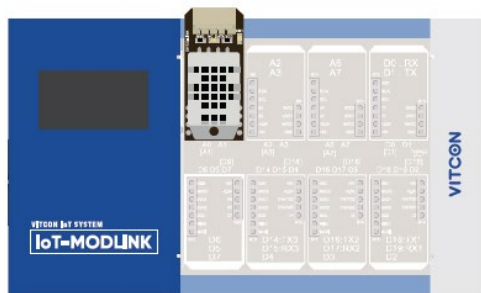
IoT-MODLINK-EX



DHT22-LINK



USB 업로드 케이블(USB 2.0)



# Example I - DHT22



## DHT22 온/습도 측정 센서

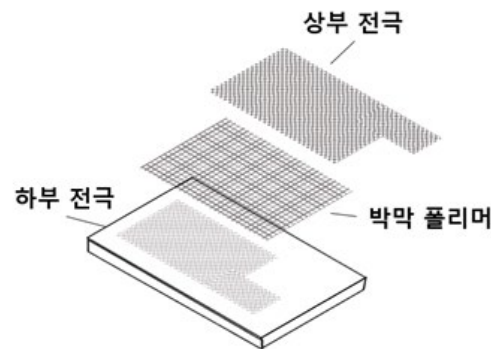
### 온도 측정 원리

- 써미스터(Thermistor) 소자 활용
- 온도에 따라 물질의 저항 값이 변하는 소재 특성 이용
- 저항 값의 변화를 감지하여 온도 계산

### 습도 측정 원리

- 상부 전극과 하부 전극 사이의 저항 변화 측정
- 박막 폴리머 양쪽 표면에 전극이 부착된 얇은 판이 존재
- 얇은 판이 공기 중의 수분을 흡수하면, 두 전극의 전도도에 변화가 발생
- 양 전극에 흐르는 미세 전류를 습도로 역산

DHT22 센서



# Example I - DHT22



## 온/습도 측정을 위한 스케치 작성

```
#include "DHT.h"

#define DHTPIN A1    // Digital pin connected to the DHT sensor
#define DHTTYPE DHT22 // DHT 22 (AM2302), AM2321

// Initialize DHT sensor.
DHT dht(DHTPIN, DHTTYPE);

uint32_t DataCaptureDelay = 2000; //ms
uint32_t StartTime = 0;

void setup() {
  Serial.begin(9600);
  dht.begin();
  StartTime = millis();
}
```



# Example I - DHT22



## 온/습도 측정을 위한 스케치 작성

```
void loop() {  
  if ((millis() - StartTime) > DataCaptureDelay) { //2초 간격으로 실행  
    float h = dht.readHumidity();  
    // Read temperature as Celsius (the default)  
    float t = dht.readTemperature();  
  
    // Check if any reads failed and exit early (to try again).  
    if (isnan(h) || isnan(t)) {  
      Serial.println(F("Failed to read from DHT sensor!"));  
      return;  
    }  
  
    Serial.print(F("Humidity: "));  
    Serial.print(h);  
    Serial.print(F("% Temperature: "));  
    Serial.print(t);  
    Serial.println(F("°C "));  
  
    StartTime = millis();  
  }  
}
```

# Example I - DHT22



## millis() 함수

- 타이머 사용을 위한 시간 함수
- millis() 함수의 실행 결과는 아두이노가 작동되고 지난 시간을 밀리초(ms) 단위로 알려줌

## millis() vs. delay()

- delay() 함수 실행 시, 지정한 시간 동안 아두이노가 아무 동작을 하지 못함 (아두이노가 정지!)
- millis()는 흐르는 시간만 계속 확인 - **아두이노가 계속 동작 상태!**
- 여러 개의 모듈을 동시에 타이머 조작할 때에는 millis()가 유리
- delay()로 여러 모듈의 타이머를 조절하기에는 매우 어려움

# Example II - OLED



## 128x64 I2C OLED 모듈

I2C 인터페이스를 사용하는 OLED 모듈

- SSD1306 드라이버 IC로 디스플레이 제어
- 128x64 해상도: x축 128칸, y축 64칸 표시 가능
- u8g2 라이브러리를 사용하여 텍스트 및 그래픽 표시가 가능  
(u8x8 라이브러리 사용 시, 텍스트만 출력 가능)
- 디스플레이 렌더링 시, MCU 메모리 사용



## OLED 모듈 사용 방법

- u8g2 객체 선언 및 시작
- 렌더링 버퍼 비운 후, 사용할 폰트 선언
- OLED 상 x축/y축 좌표 지정 후, 텍스트 drawing
- 또는, 커서 위치 조정 후 변수 print

# Example II - OLED



## 측정한 온/습도 정보를 OLED 화면으로 출력

```
#include <U8g2lib.h>
#include "DHT.h"
U8G2_SH1106_128X64_NONAME_F_HW_I2C u8g2(U8G2_R0, /* reset=*/ U8X8_PIN_NONE);

#define DHTPIN A1      // Digital pin connected to the DHT sensor
#define DHTTYPE DHT22 // DHT 22 (AM2302), AM2321

// Initialize DHT sensor.
DHT dht(DHTPIN, DHTTYPE);

uint32_t DataCaptureDelay = 3000; //ms
uint32_t DataCapture_ST = 0; //Start Time

float Temp;
float Humi;

void setup() {
  dht.begin();
  u8g2.begin();

  DataCapture_ST = millis();
}
```

# Example II - OLED



## 측정한 온/습도 정보를 OLED 화면으로 출력

```
void loop() {  
  if ((millis() - DataCapture_ST) > DataCaptureDelay) { //3초 간격으로 실행  
    Humi = dht.readHumidity();  
    // Read temperature as Celsius (the default)  
    Temp = dht.readTemperature();  
  
    // Check if any reads failed and exit early (to try again).  
    if (isnan(Humi) || isnan(Temp)) {  
      Serial.println(F("Failed to read from DHT sensor!"));  
      return;  
    }  
    OLEDdraw(); //OLED에 데이터 출력  
    DataCapture_ST = millis();  
  }  
}
```



# Example II - OLED



## 측정한 온/습도 정보를 OLED 화면으로 출력

```
void OLEDdraw() {  
    u8g2.clearBuffer();  
  
    u8g2.setFont(u8g2_font_ncenB08_te); //폰트 지정  
    u8g2.drawStr(1, 15, " SMART FARM"); //x축좌표, y축좌표, 문자열  
  
    u8g2.drawStr(15, 36, "Temp.");  
    u8g2.setCursor(85, 36);  
    u8g2.print(Temp);  
    u8g2.drawStr(114, 36, "Wxb0" );  
    u8g2.drawStr(119, 36, "C");  
  
    u8g2.drawStr(15, 47, "Humidity");  
    u8g2.setCursor(85, 47); u8g2.print(Humi);  
    u8g2.drawStr(116, 47, "%");  
  
    u8g2.sendBuffer();  
}
```

# Example III - SOIL LINK



## 토양 습도 측정 센서

DHT22 내 습도 측정 센서의 원리와 동일

## 토양 습도 측정 준비



IoT-MODLINK-EX



DHT22-LINK



USB 업로드 케이블(USB 2.0)



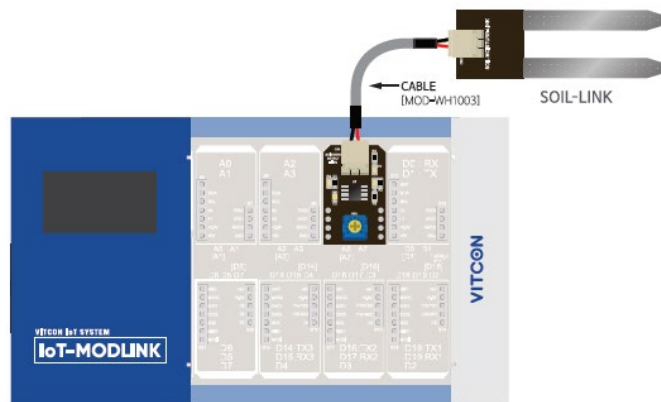
MOISTURE DETECT-LINK



SOIL-LINK



케이블 MOD-WH1003



# Example III - SOIL LINK



## pinMode() 함수

- MODLINK-EX에는 19개의 디지털 핀, 6개의 아날로그 핀이 존재
- 해당 핀에 연결된 LINK가 입력으로 사용할 것인지 출력으로 사용할 것인지 명시해야 함
- pinMode(pin, mode)의 형태  
: pin 자리에 사용할 핀 번호, mode에 INPUT 또는 OUTPUT 설정

## map() 함수

- map() 함수는 하나의 범위에서 다른 범위로 숫자를 다시 매핑해주는 함수
- 보통 아날로그 데이터를 디지털 데이터 값에 맞게 변환할 때 사용
- Map(Rawdata, fromLow, fromHigh, toLow, toHigh)의 형태



< map(analog, 0, 1023, 0, 4) 사용 예시 >

# Example III - SOIL LINK



## 토양 습도 시리얼 모니터 확인

```
#define SOILHUMI A6

int Soilhumi = 0;

void setup() {
  Serial.begin(9600);
  pinMode(SOILHUMI, INPUT);
}

void loop() {
  //습도가 높아질수록 숫자가 커지도록 변환

  //토양 습도 단계를 0~1023에서 0~100으로 변환
  Soilhumi = map(analogRead(SOILHUMI), 0, 1023, 100, 0);

  Serial.print("현재 토양 습도 : ");
  Serial.println(Soilhumi);
  delay(500); //0.5초 딜레이
}
```



# Exercise I - LINK 혼합 사용

## 3초 간격으로 주변 온/습도 및 토양 습도 측정 후 OLED 표시

- 1) 3초마다 DHT22와 SOIL-LINK의 데이터 측정  
: delay() 함수 대신 millis()함수를 사용하여 측정
- 2) 측정된 각각의 데이터를 OLED에서 한번에 표시  
: DHT22-OLED 예제에서 SOIL-LINK 데이터 추가 표시

hint -1) OLED(15, 58)위치에 Soil Humi. 문자열 draw  
hint -2) OLED(85, 58)위치에 커서 변경 후, 측정 값 print  
hint -3) OLED(116, 58)위치에 단위(%) 문자 draw

# Example IV - DC FAN



## DC FAN

### DC FAN 작동 원리

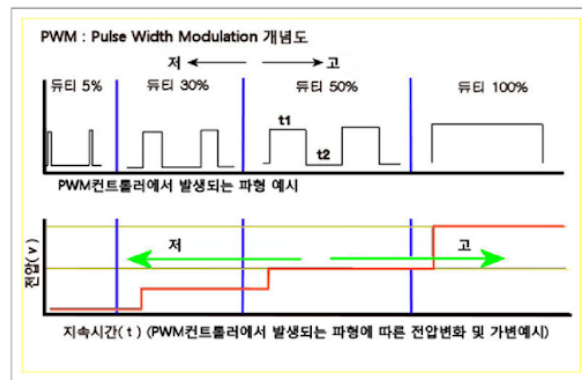
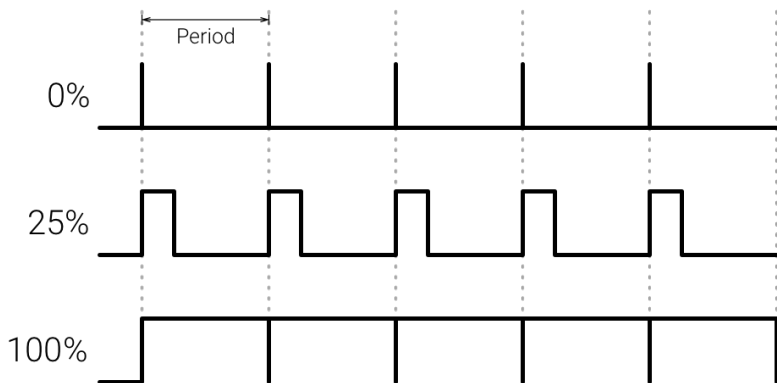
- 전하의 방향과 극성이 같은 직류(Direct Current, DC) 전압으로 속도를 제어

### PWM 전압 제어 방식

- Pulse Width Modulation의 약자
- Duty Cycle: 파동이 1회 생성될 때의 ON 되는 시간과 OFF 되는 시간의 비율



3.5inch DC FAN



# Example IV - DC FAN



## PWM 제어

### EVC-ADJ-LINK

- MODLINK의 EVC1, EVC2 전원 또는 외부에서 7~30 V DC 전원을 공급받음
- PWM을 입력 신호로 하여 외부 단자로 일정한 전압을 출력하는 모듈
- SoftPWM 라이브러리 사용  
: 디지털 핀에 PWM 신호를 출력
- SoftPWM 라이브러리를 사용하지 않는다면,  
analogWrite() 함수를 사용하여 지정된 핀에만 dutycycle 값을 출력



EVC-ADJ-LINK

입력전압	PWM값	출력전압	DUTY
24VDC	255	12.17VDC	100%
24VDC	230	11.66VDC	90%
24VDC	204	9.32VDC	80%
24VDC	179	6.37VDC	70%
24VDC	155(min)	3.11VDC	60%

입력전압	PWM값	출력전압	DUTY
12VDC	255	11.67VDC	100%
12VDC	230	11.52VDC	90%
12VDC	204	9.36VDC	80%
12VDC	179	6.38VDC	70%
12VDC	155(min)	3.07VDC	60%



# Example IV - DC FAN



## 주의사항 - DC 전원 잭 극성

EVC-ADJ-LINK와 DC 잭을 사용할 시, 점선 무늬가 있는 쪽이 -(GND)에 해당



# Example IV – DC FAN



## 주의사항 – 꽃음형 커넥터 사용 방법

EVC-ADJ-LINK와 전원 공유 연결 시 사용  
- 빵판(Breadboard)처럼 4칸이 이어져 있음



# Example IV - DC FAN



## DC FAN 제어 준비



# Example IV – DC FAN



## 2초마다 DC FAN ON/OFF

```
#include <SoftPWM.h>

SOFTPWM_DEFINE_CHANNEL(A3); //SoftPWM으로 사용할 핀 설정

void setup() {
  SoftPWM.begin(490); //PWM frequency 설정
}

void loop() {
  SoftPWM.set(100); //duty rate 100%, DC FAN 켜짐
  delay(2000); //2초 딜레이
  SoftPWM.set(0); //duty rate 0%, DC FAN 꺼짐
  delay(2000); //2초 딜레이
}
```

# Example IV – DC FAN



## DC FAN PWM 제어

```
#include <SoftPWM.h>
SOFTPWM_DEFINE_CHANNEL(A3); //SoftPWM으로 사용할 핀 설정

void setup() {
  SoftPWM.begin(490); //PWM frequency 설정
}

void loop() {
  //최소 duty rate : 65%
  for (int i = 65; i < 100; i++)
  {
    //팬 속도가 점점 빨라짐
    SoftPWM.set(i);
    delay(100);
  }

  //최대 duty rate : 100%
  for (int i = 100; i > 65; i--)
  {
    //팬 속도가 점점 느려짐
    SoftPWM.set( i);
    delay(100);
  }
}
```



# Exercise II - LINK 혼합 사용 2

## 주변 온도 조건에 따른 DC FAN 제어

- 1) DHT22의 온도 측정치에 따라 DC FAN ON/OFF
- 2) 30도 이상 측정 시, DC FAN ON(PWM: 100)
- 3) 25도 이하 측정 시, DC FAN OFF(PWM: 0)
- 4) 중간 온도 범위 ( $25 < \text{temp.} < 30$ )에서 DC FAN PWM 65으로 ON
- 5) 현재 온도와 DC FAN PWM 제어 상태를 OLED에 표시

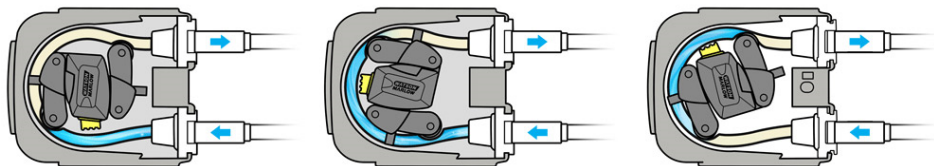
# Example V - DC PUMP



## DC PUMP

### DC PUMP 작동 원리

- DC 전압이 인가될 시, 모터가 작동
- 모터가 회전하면서 배수 방향으로 물을 흘려 보냄



- 릴레이(Relay) LINK와 연결하여 ON/OFF 제어



DC PUMP 내부 구조



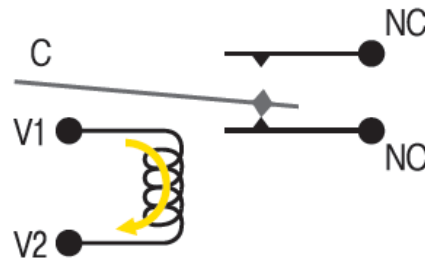
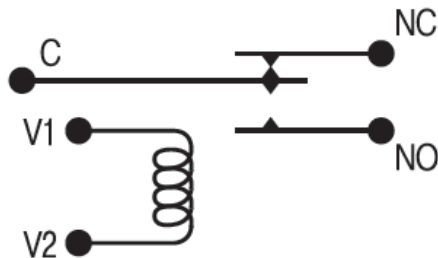
# Example V - DC PUMP



## Relay

### 전자석을 이용한 전기적 스위치

- 스위치의 일종이나, 손으로 스위치를 누르지 않음
- 전자석의 원리로 단절된 부분을 이어주는 방식

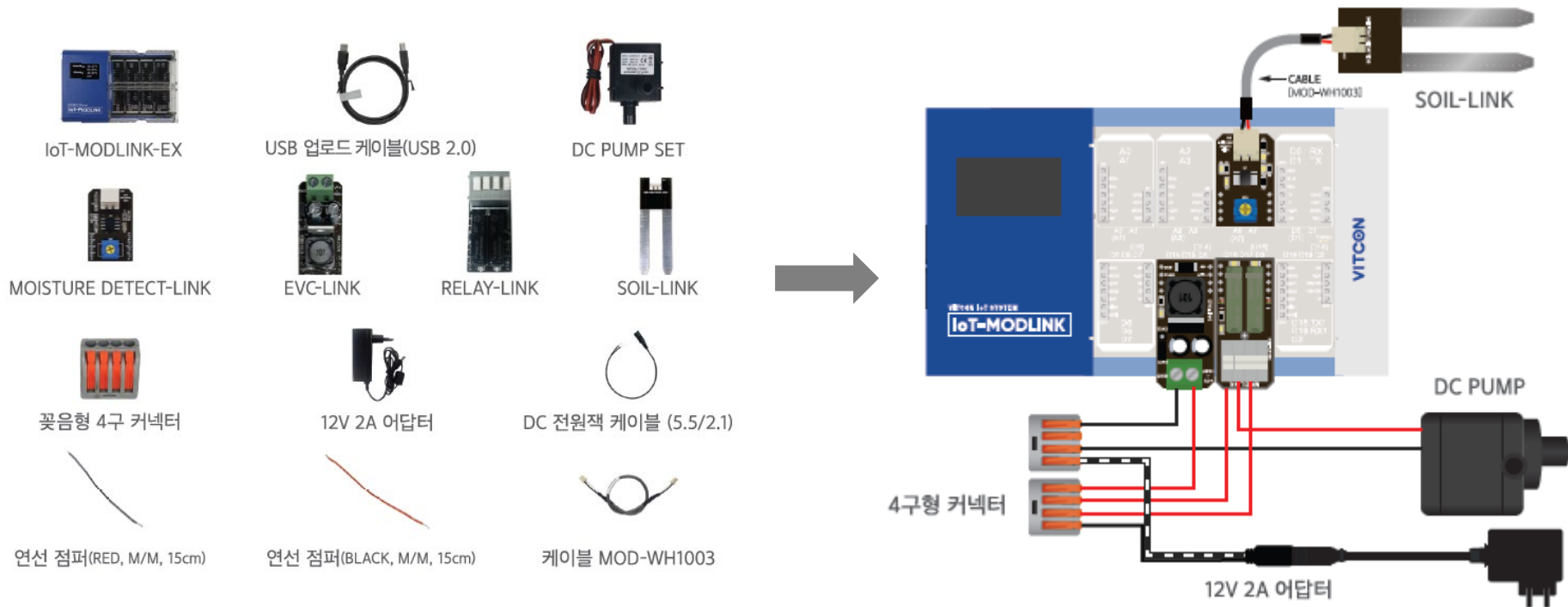


- 릴레이를 동작시키는 회로와 실제 스위치 부분이 전기적으로 분리
- 릴레이의 스위치가 되는 부분에 기기를 연결하면, 릴레이 동작 전압과 다른 전압의 기기를 쉽게 작동 가능
- 릴레이 동작 회로에 전압이 발생 - 전자석이 동작 - 릴레이 스위치 핀이 옮겨짐
- 스위치 접점이 붙을 때, "딸깍" 하는 소리가 발생

# Example V - DC PUMP



## DC PUMP 제어 준비



# Example V - DC PUMP



## DC PUMP ON/OFF

```
#define PUMP 16 //D16 핀을 DC PUMP와 연결된 릴레이 핀으로 지정

void setup() {
  pinMode(PUMP, OUTPUT); //D16핀을 출력 모드로 지정
}

void loop() {
  //DC PUMP와 연결된 RELAY 접점 ON
  digitalWrite(PUMP, HIGH);
  delay(2000); //2초 딜레이
  //DC PUMP와 연결된 RELAY 접점 OFF
  digitalWrite(PUMP, LOW);
  delay(2000); //2초 딜레이
}
```

**DC 펌프는 수중용입니다. 수중 이 외 동작 시 고장이 발생할 수 있습니다!**

물이 든 종이컵 안에 넣고 사용 권장



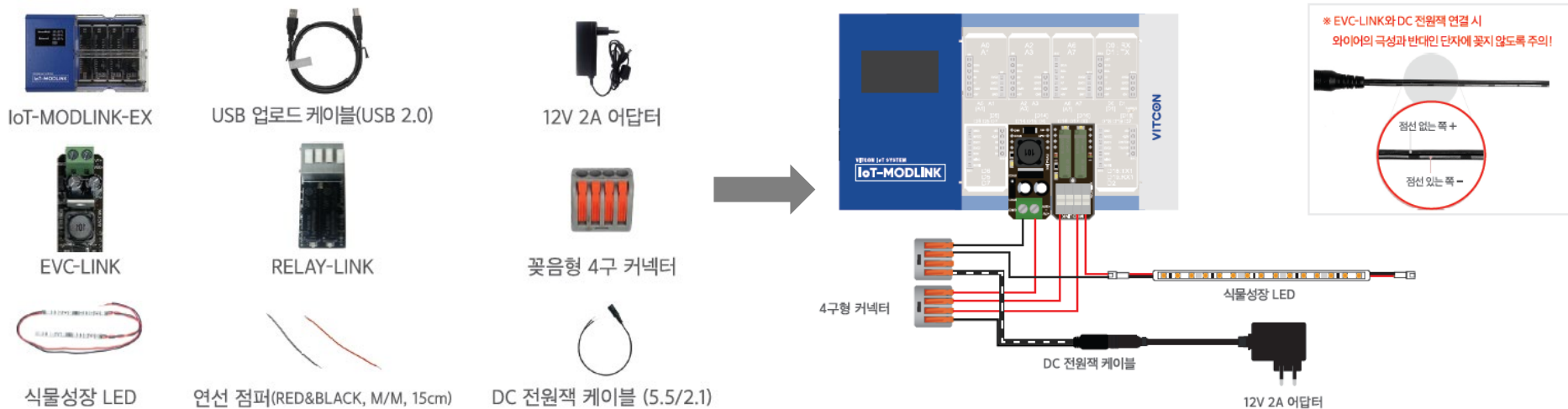
# Exercise III - LINK 혼합 사용 3

## 토양 습도 조건에 따른 DC PUMP 제어

- 1) SOIL-LINK에서 토양 습도 측정
- 2) 토양 습도가 30 이하일 때 DC PUMP 동작
- 3) 토양 습도가 60 이상일 때 DC PUMP OFF

# Example VI - LED

## LED 제어 준비



# Example VI - LED



## 시간 간격 조절에 따른 LED ON/OFF

```
#define LAMP 17 //D17 핀을 LED 조명과 연결된 릴레이 핀으로 지정

uint32_t TimeCompare;
uint32_t StartTime = 0;

//LAMP가 ON/OFF하게 될 시간 간격 설정(최소 단위 1분)
uint32_t TimeSum; //총 시간(ms 단위)
int Hour = 0; //시간
int Minute = 1; //분

void setup() {
    pinMode(LAMP, OUTPUT); //D3핀을 출력 모드로 지정
    TimeSum = (uint32_t)(Hour * 60 + Minute) * 60 * 1000; //Interval 시간단위, ms단위로 변환
    StartTime = millis();
}

void loop() {
    //현재 시간에서 setup함수가 끝난 시점의 시간을 빼고 Interval 시간 단위로 나눈다.
    TimeCompare = (millis() - StartTime) / TimeSum;
    if (TimeCompare % 2 == 0) { //TimeCompare값이 2의 배수일 때
        digitalWrite(LAMP, LOW);
    }
    else if (TimeCompare % 2 == 1) { //TimeCompare값이 2의 배수가 아닐 때
        digitalWrite(LAMP, HIGH);
    }
}
```



# Exercise IV – 센서, 액츄에이터 통합

## 센서 및 액츄에이터 통합 – 기본 IoT 시스템 구현

- 1) 온/습도(DHT22)는 2초마다 측정
- 2) 토양 습도(SOIL-LINK)는 3초마다 측정
- 3) FAN / PUMP는 주변 온습도 환경에 따라 자동 동작  
: FAN ON(29도 이상) / FAN OFF (20도 이하) / FAN PWM 65 (사이 범위)  
: PUMP ON (토양 습도 30~60% 사이)
- 4) 센서 및 액츄에이터 상태 OLED 표시  
: DHT22 및 SOIL-LINK 측정 정보  
: FAN/PUMP/LED 동작 상태  
: LED 자동 ON/OFF 시간 간격