

임베디드 시스템 소프트웨어

타겟 보드 및 개발환경 이해

2023.12

(주)다인시스

목 차

◆ INL (IoT Node Link)

ESP32-S3

ESP32-S3-DevKitC-1

ESP32-S3 개발 환경

Arduino Platform

범용 IoT Node Link (INL Series)

◆ 범용 IoT Node Link(INL Series)

- ESP32-S3와 Arduino Platform 기반의 산업용 제어 보드
- UART, RS232, RS485, GPIO와 LAN, WLAN, BLE 지원하는 IoT 기반 제어 및 상태 감시 목적의 Node Device로 사용

◆ 주요 특징

구분	INL-01	INL-02	INL-03	INL-04	INL-05	INL-06
MCU	ESP32-S3	ESP32-S3	ESP32-S3	ESP32-S3	ESP32-S3	ESP32-S3
Serial	UART, RS232	RS232,RS485	RS232,RS485	RS232,RS485,UART(4)	RS232,RS485,UART(4)	RS232,RS485,UART(4)
USB			지원	지원	지원	지원
IO	I2C(2),ADIN/GPIO(8)	I2C(1),ADIN/GPIO(1)	I2C(1),ADIN/GPIO(1)	I2C(1),ADIN/GPIO(4)	I2C(1),ADIN/GPIO(4)	I2C(1),ADIN/GPIO(4)
Digital In		5V (1), 24V(1)	5V (1), 24V(1)	5V (4), 24V(4)	5V (8), 24V(8)	5V (4), 24V(4)
Digital Out		Sink(1)	Sink(1)	Sink(4)	Sink(8)	Sink(4)
Relay Out				Relay(4)	Relay(2)	
RTC			지원	지원	지원	지원
통신	WLAN/BT/LAN	WLAN/BT	WLAN/BT/LAN	WLAN/BT/LAN	WLAN/BT/LAN	WLAN/BT LTE CatM.1 지원

범용 IoT Node Link 제품군 사양 비교

구분	INL-01	INL-02	INL-03	INL-04	INL-05	INL-06	설명
MCU	ESP32-S3	ESP32-S3	ESP32-S3	ESP32-S3	ESP32-S3	ESP32-S3	
Upload	1	1	1	1	1	1	ESP32 UART0, USB-C
RS232	1	1	1	1	1	1	ESP32 UART1
RS485		1	1	1	1	1	ESP32 UART2
UART	1(U2)			4	4	4	MAX14830 (I2C to UART)
USB			1	1	1	1	USB-C, OTG
I2C	2(HW,SW)	1(GPIO)	1(GPIO)	1(GPIO)	1(GPIO)	1(GPIO)	4P Wafer connector : VGG, GND, SDA, SCL
ADIN/GPIO	8(All)	1	1	4	4	4	ESP32 ADC, GPIO 3.3V
Digital In		1	1	4	8	4	DC Input 0V(Low), 24V(High, 20~28V)
Digital In		1	1	4	8	4	DC Input 0V(Low), 5V(High)
Digital Out		1	1	4	8	4	DC Sink 5~26 VDC, NPN TR OUTPUT (High->GND, Low->OFF)
Relay Out				4	2		10A 출력, (High->On, Low->Off)
RTC			1	1	1	1	DS1307(I2C)
LED	2	2	2	2	2	2	Status, Power
Button	2	2	2	2	2	2	Reset, Boot
Switch		8	8	8	8	8	Additional GPIO, PCF8574, 4 pin select switch : RS485 ID select
R45(LAN)	1		1	1	1		Wiznet W5500
WiFi	1	1	1	1	1	1	ESP32-S3-WROOM-1(내장안테나/-1U(외장안테나)
BLE	1	1	1	1	1	1	ESP32-S3-WROOM-1(내장안테나/-1U(외장안테나)
LTE CatM.1						1	
Power	12~24V	12~24V	12~24V	12~24V	12~24V	12~24V	Terminal Block 사용

재난안전용 IoT Node Link (INL Series)

◆ 재난안전용 IoT Node Link for Diaster(INL-D Series)

- ESP32-S3와 Arduino Platform 기반의 산업용 제어 보드
- 지진, 화재 감지, 긴급 버튼 지원

◆ 주요 특징

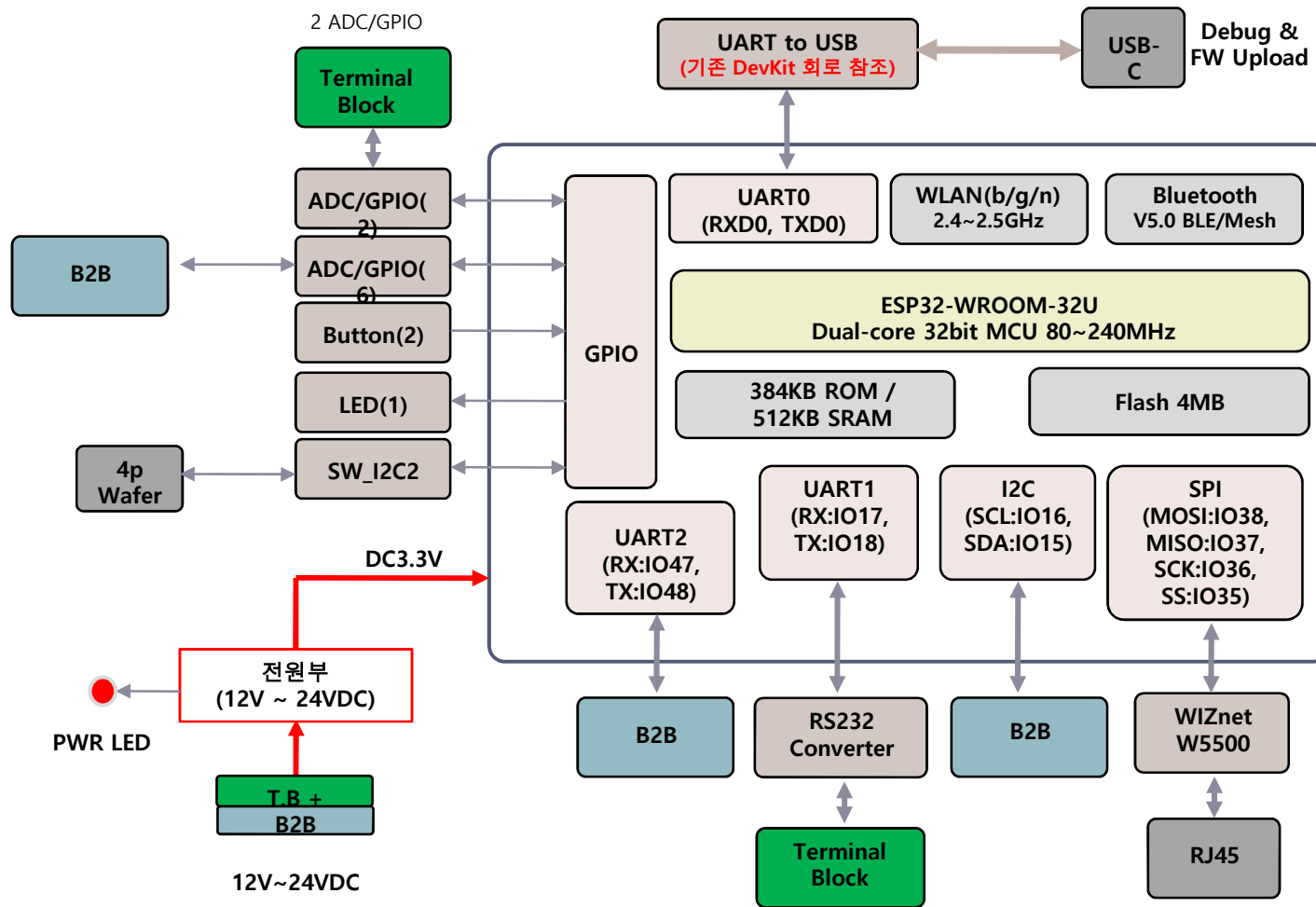
구 분	사 양	비 고
ESP32-S3	Xtensa dual-core, LX7 240MHz	
지진감지	6축 Gyro/Acc 센서, ICM-42670	
화재감지	가연성 가스 감지센서, MQ2	
	불꽃감지 센서, L-51POPT1D2	
	온습도센서, SHT40, Sensirion	
기타 센서	조도센서, VEML3235SL	생활감지
	동작감시, TMF8801	생활감지
긴급버튼	긴급호출	
알람	긴급 알람, Buzzer, HN9650B	소리 큰 Buzzer 적용
Power	Battery with 상시 전원	
케이스	플라스틱 케이스	

INL-01 사양

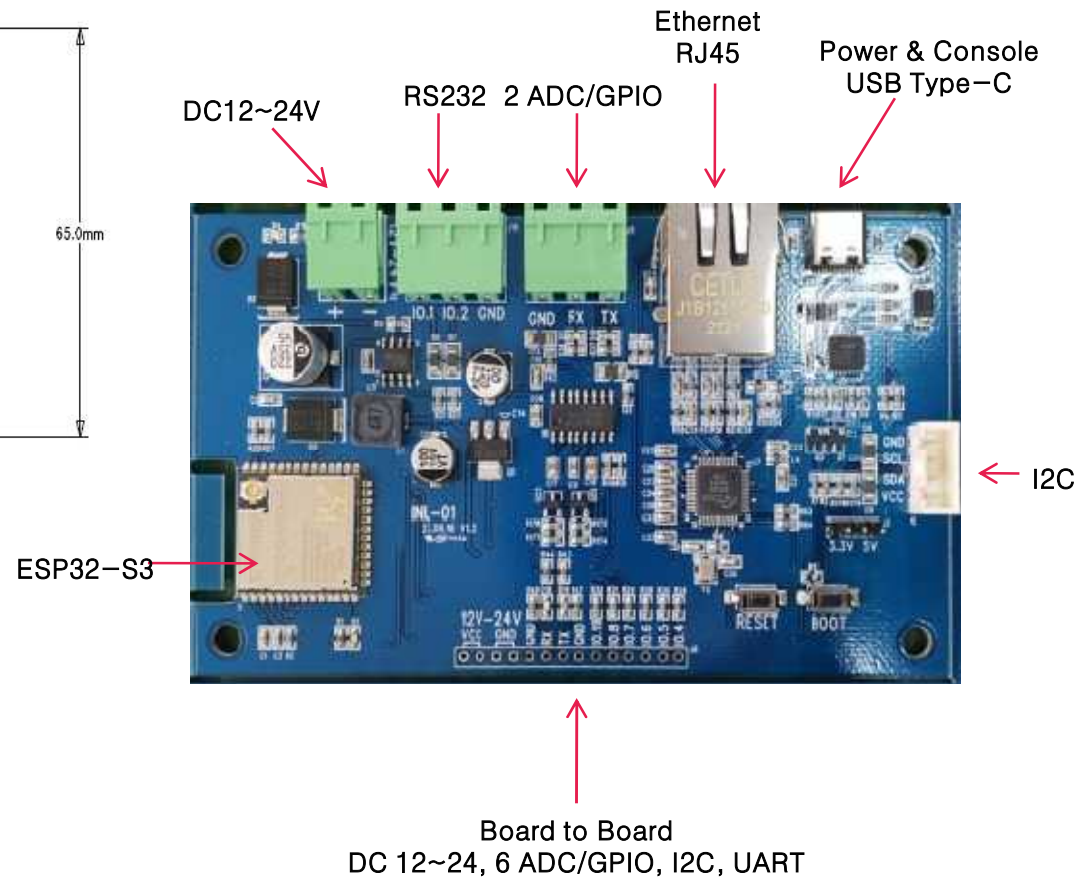
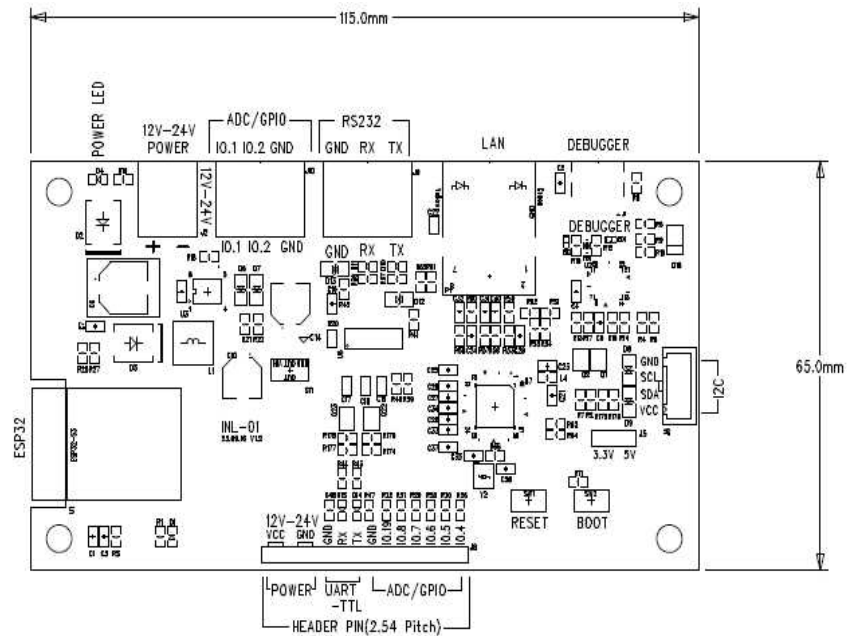
◆ INL-01 제품 사양 및 하드웨어 구조

구분	INL-01	설 명	Connector	비 고
MCU	ESP32-S3	ESP32-S3-WROOM-1(U)		내장안테나 / 외장안테나 모듈 지원
Upload	1	ESP32 UART0, USB-C		Debug & F/W upload
RS232	1	ESP32 UART1	T.B	Terminal Block 사용
UART	1	ESP32 UART2	B2B	B2B Connector
HW I2C	1	ESP32 I2C	B2B	HW I2C 사용 : B2B Connector
SW I2C	1	GPIO	Wafer	SW I2C_2, IO41, IO42 : , 4P Wafer Connector 적용
ADIN/GPIO	6	ESP32 ADC, GPIO 3.3V	B2B	IO1, IO2, IO3, IO4, O5, IO6 : B2B Connector
ADIN/GPIO	2	ESP32 ADC, GPIO 3.3V	T.B	IO7, IO8 : External Terminal Block
LED	2	Status, Power		Status LED GPIO46, Power LED는 Power에 연결
Button	2	Reset, Boot		Switch 및 Upload 포트에서 자동 reset 제어 용도로 사용
R45(LAN)	1	Wiznet W5500		SPI Interface 적용
WiFi	1	WLAN IEEEb/g/n, 2.4~2.5GHz		ESP32-S3-WROOM-1(내장안테나/-1U(외장안테나)
BLE	1	Bluetooth V5.0 BLE/Mesh		ESP32-S3-WROOM-1(내장안테나/-1U(외장안테나)
Power	12~24V	12~25V 전원 입력 가능하도록 설계	B2B / T.B	B2B Connector + External Terminal Block

INL-01 블록도



INL-01 PCB 구성

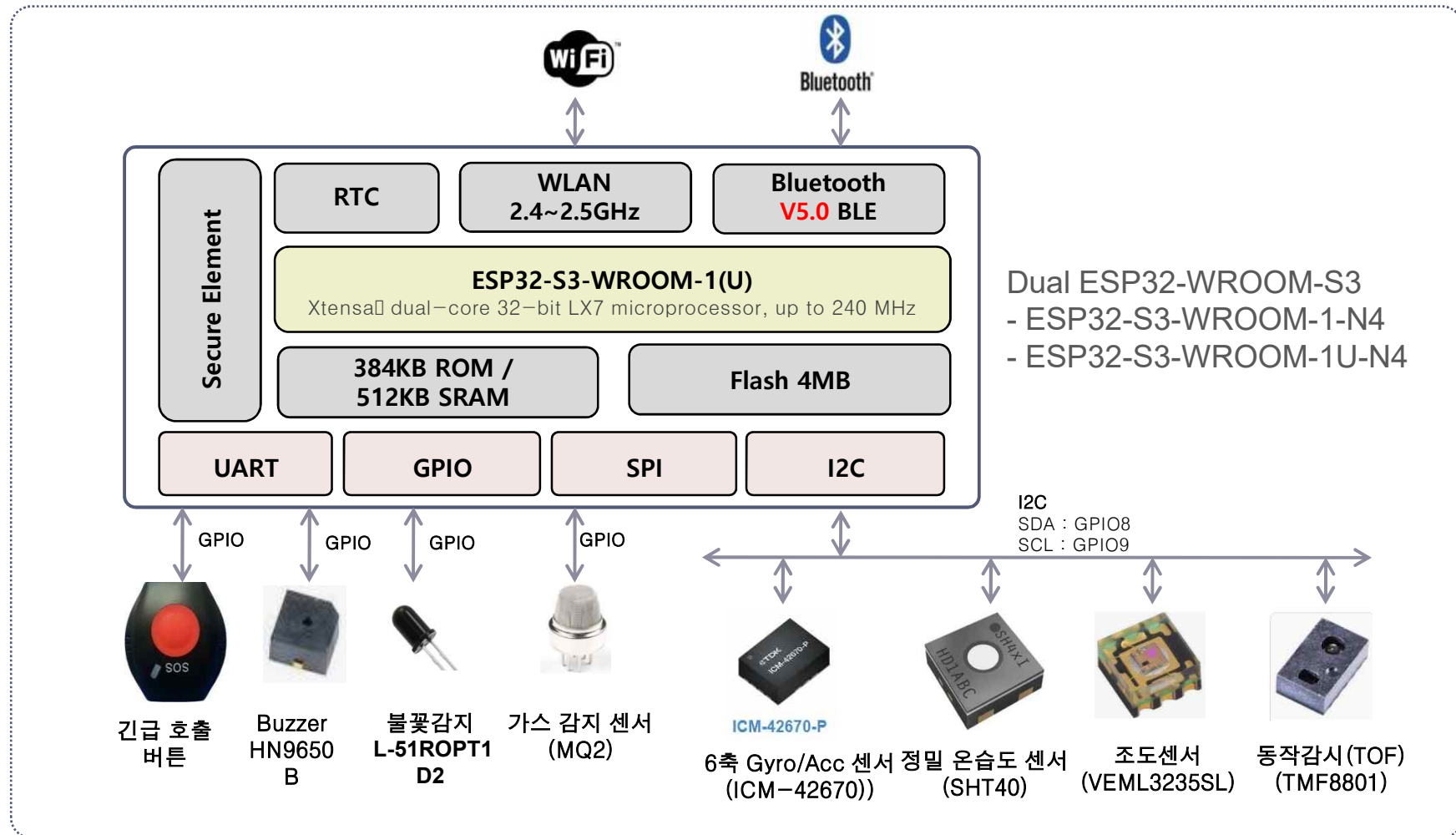


생활안전 대응 센서 노드 : INL-D01 제품 사양

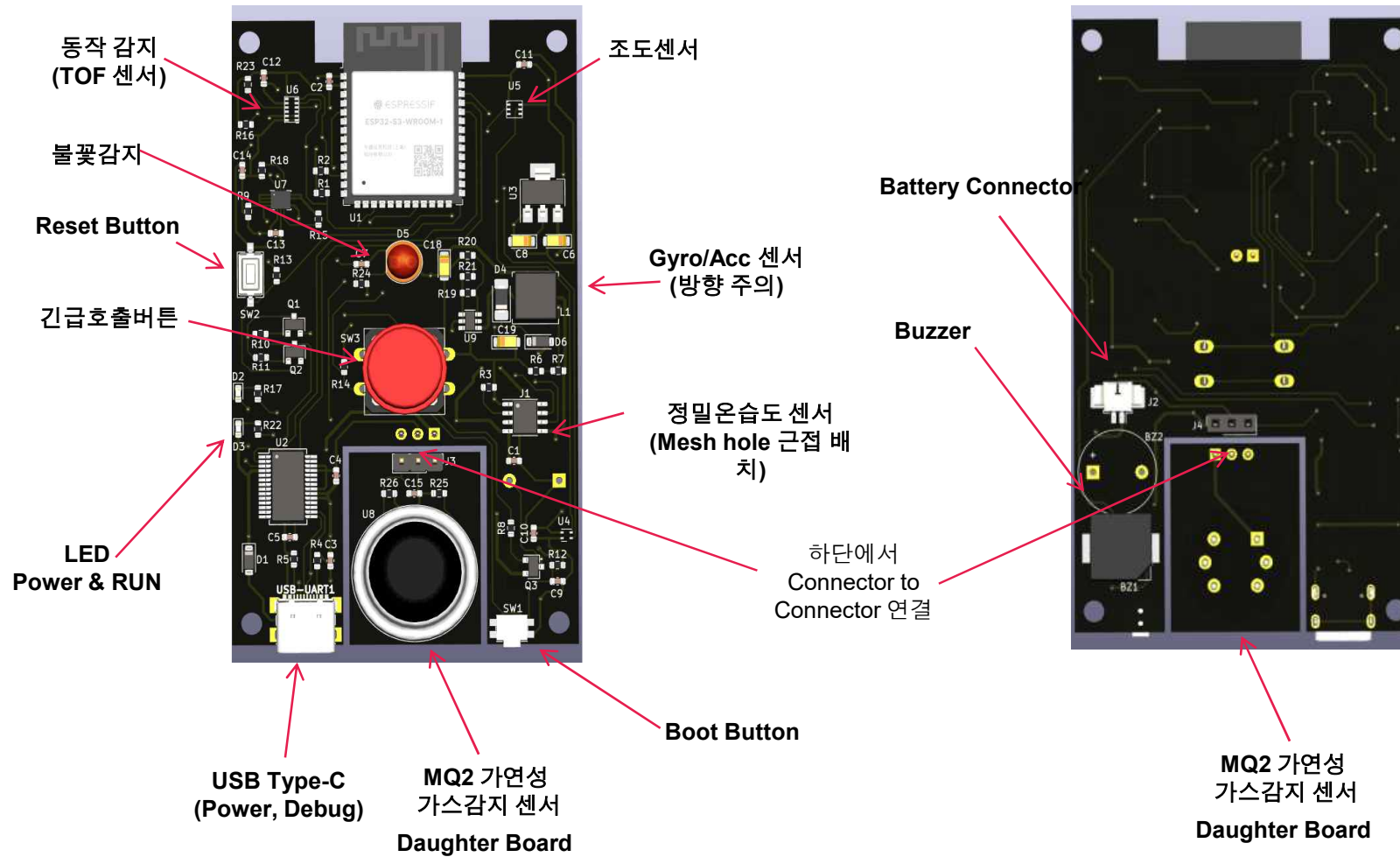
◆ INL-D01 제품 사양 및 하드웨어 구조

구분	INL-D01	ESP32-S3 IO	설 명	비 고	I2C Addr
MCU	ESP32-S3		ESP32-S3-WROOM-1(U)	내장안테나 / 외장안테나 모듈 지원	
Upload	1	TXD0, RXD0	ESP32-S3 UART0, USB-C	Debug & F/W upload	
HW I2C	1	IO8(SDA), IO9(SCL)	ESP32-S3 I2C	HW I2C 사용	
Gyro+Accel	1	IO8(SDA), IO9(SCL)		6축 Gyro/Acc 센서, ICM-42670	0x68
Temp+Hum.	1	IO8(SDA), IO9(SCL)		SHT40, 정밀 온습도 센서	0x44
Light	1	IO8(SDA), IO9(SCL)		조도센서, VEML3235SL	0x10
TOF	1	IO8(SDA), IO9(SCL)		동작 감지용, TMF8801	0x41
Gas	1	Analog		MQ2, Analog Read	
Flame	1	IO5	불꽃감지	GPIO, L-51POPT1D2	
BUZZER	1	IO7		GPIO, HN9650B	
BAT	1	IO2		Battery Present	
LED	2	X	Battery Charge, Standby	Battery Charge, Standby State 표시	
Button	1	EN	Reset	Reset 제어용	
Button	1	IO0	Boot	Boot 스위치, Factory Reset 용으로 사용 예정	
Button	1	IO6	Safe	긴급 호출 용도로 사용	
WiFi	1		WLAN IEEE b/g/n, 2.4~2.5GHz	ESP32-S3-WROOM-1(내장안테나/-1U(외장안테나)	
BLE	1		Bluetooth V5.0 BLE/Mesh	ESP32-S3-WROOM-1(내장안테나/-1U(외장안테나)	
Power	USB, Battery		USB 전원 or Battery		

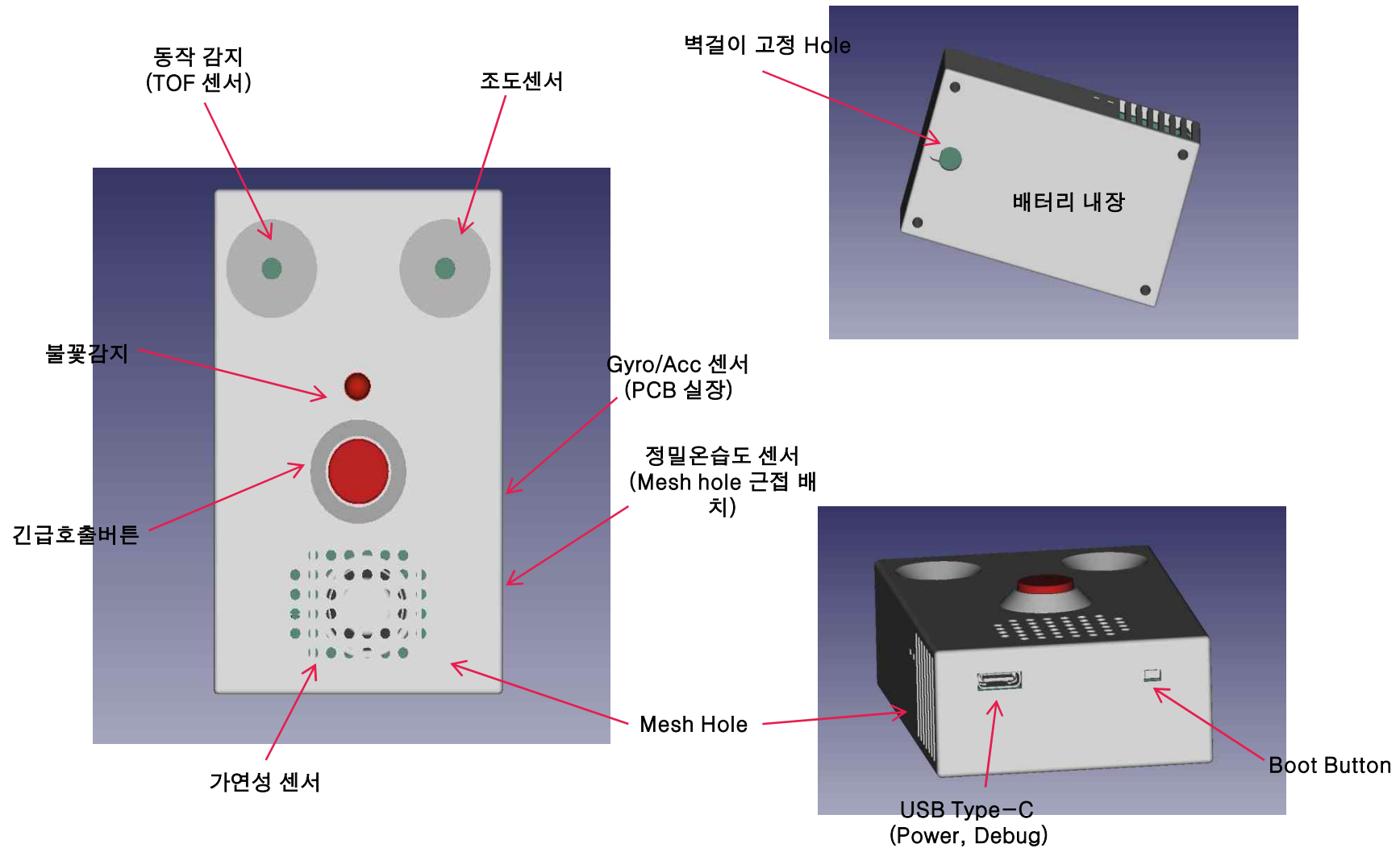
INL-D01 블록도



INL-D01 PCB 구성



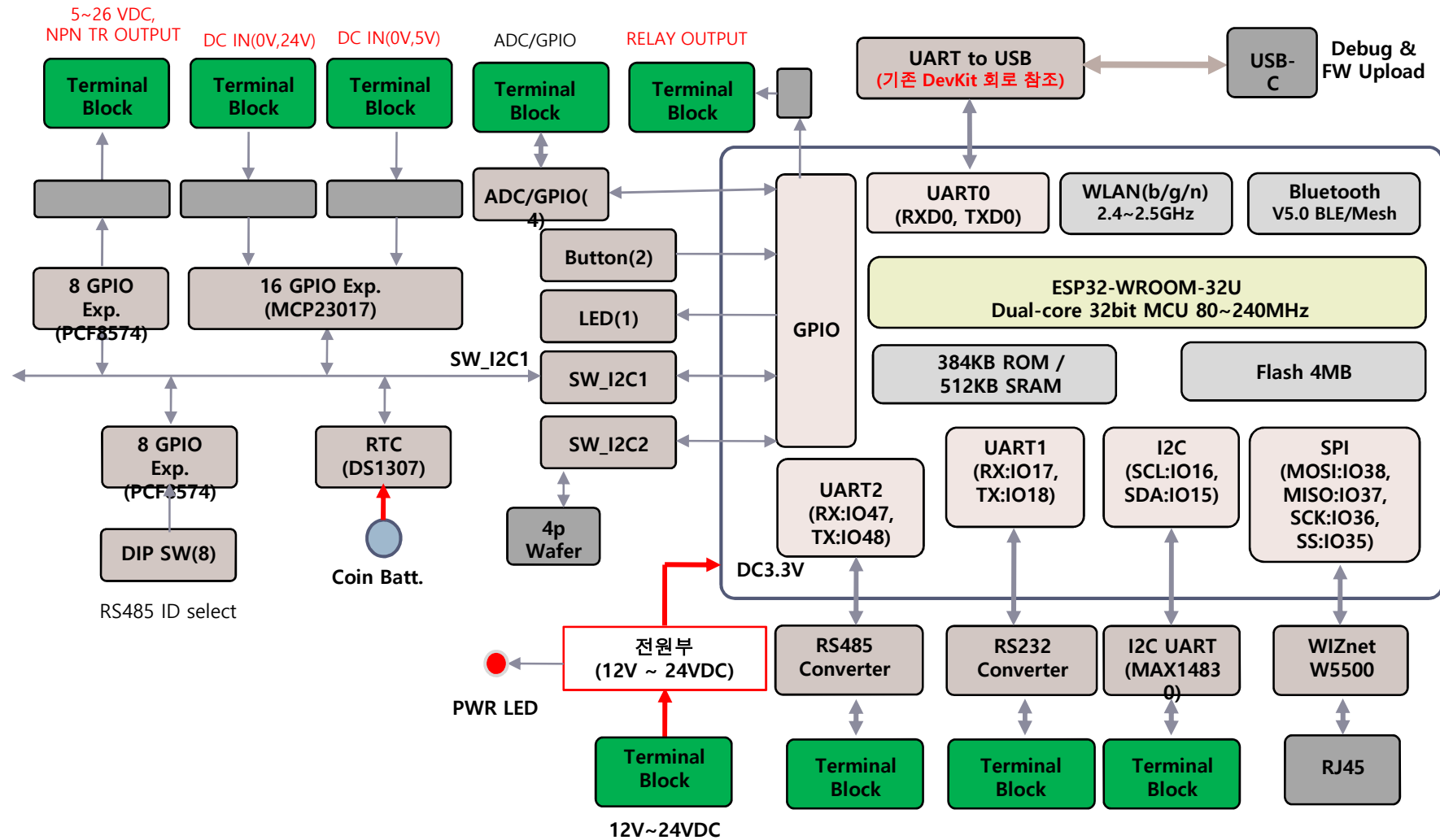
INL-01 외관



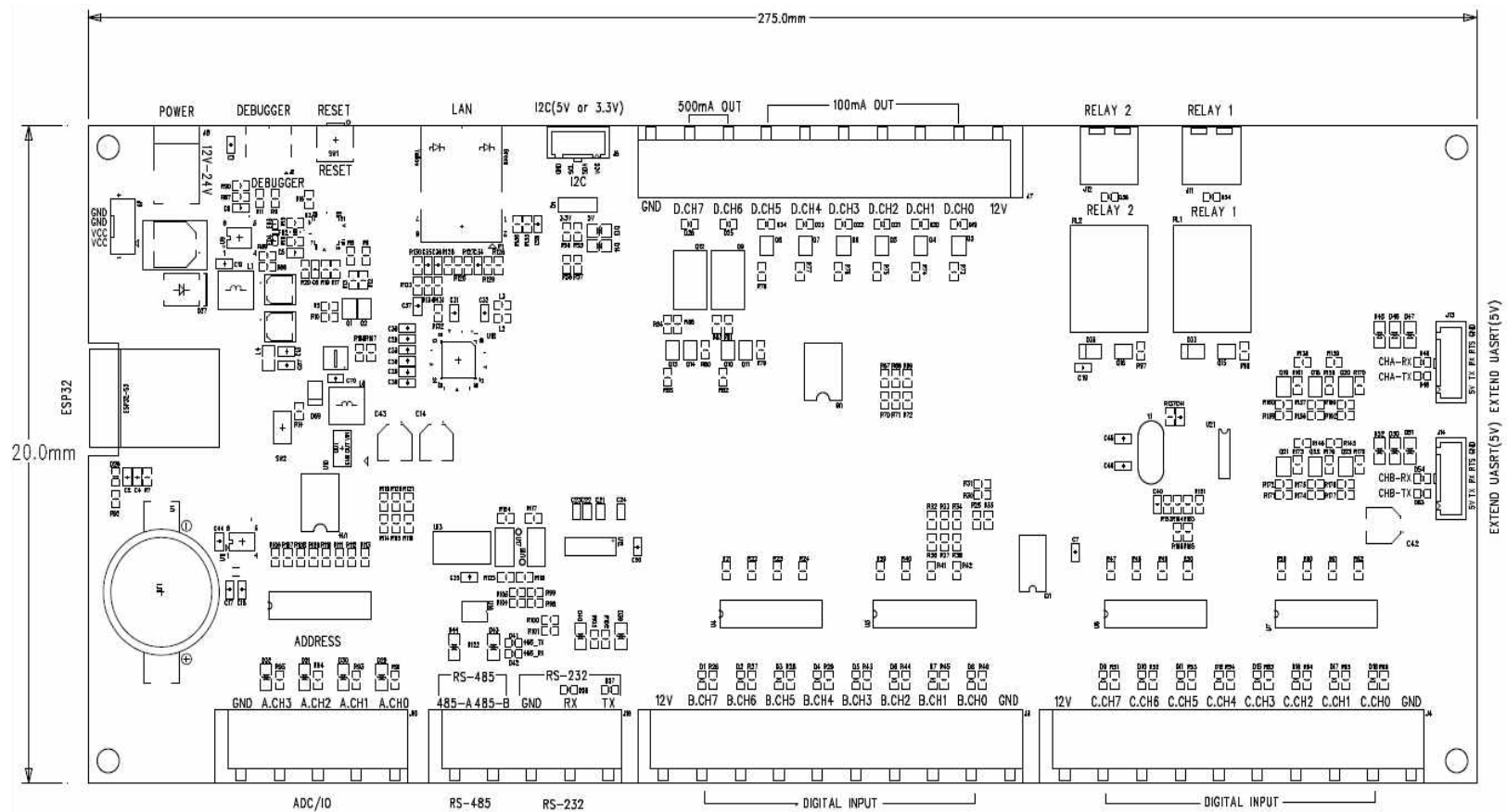
INL-05 사양

구분	INL-05	설 명	비 고
MCU	ESP32-S3	ESP32-S3-WROOM-1(U)	내장안테나 / 외장안테나 모듈 지원
Upload	1	ESP32 UART0, USB-C	Debug & F/W upload
RS232	1	ESP32 UART1	Terminal Block 사용
RS485	1	ESP32 UART2	Terminal Block 사용
UART	4	MAX14830 (I2C to UART)	HW I2C 사용, UART TTL 출력
USB	1	USB-C, OTG	
I2C	1(SW)	GPIO	SW I2C_2, IO41, IO42, 4P Wafer Connector 적용
ADIN/GPIO	4	ESP32 ADC, GPIO 3.3V	IO1, IO2, IO3, IO4
Digital In	8	DC Input 20~24 VDC	MCP23017 16 port , Software I2C 사용
Digital In	8	DC Input 0~5 VDC	DC 입력 범위는 Compile Tech. FA-DUINO 참조
Digital Out	8	DC Sink 5~26 VDC	PCF8574 I2C GPIO 확장, NPN TR OUTPUT , Compile Tech. FA-DUINO 참조
Relay Out	2	Relay 10A	ESP32 IO21,IO45, 10A 출력, Compile Tech. FA-DUINO 참조
RTC	1	DS1307(I2C)	Software I2C 사용
LED	2	Status, Power	Status LED GPIO46, Power LED는 Power에 연결
Button	2	Reset, Boot	Switch 및 Upload 포트에서 자동 reset 제어 용도로 사용
Switch	8	RS485 ID select switch	PCF8574 사용, 8 port GPIO 확장, Software I2C 사용
RJ45(LAN)	1	Wiznet W5500	SPI Interface 적용
WiFi	1	WLAN IEEE802.11b/g/n, 2.4~2.5GHz	ESP32-S3-WROOM-1(내장안테나/-1U(외장안테나)
BLE	1	Bluetooth V5.0 BLE/Mesh	ESP32-S3-WROOM-1(내장안테나/-1U(외장안테나)
Power	12~24V	12~25V 전원 입력 가능하도록 설계	Terminal Block 사용

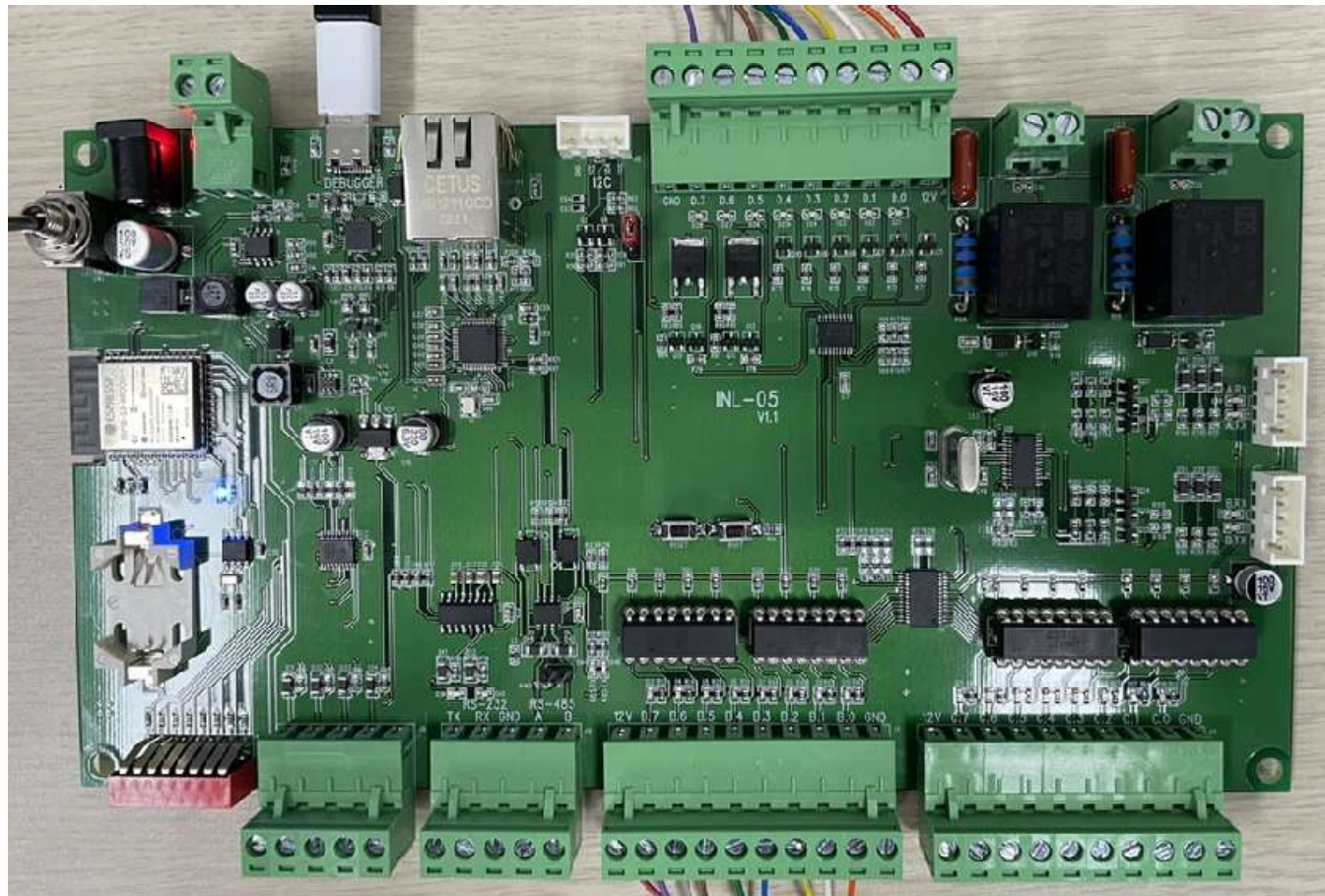
INL-D05 블록도



INL-05 PCB 구성



INL-05 PCBA



목 차

INL(IoT Node Link)

◆ **ESP32-S3**

ESP32-S3-DevKitC-1

ESP32-S3 개발 환경

Arduino Platform

ESP32-S3 SoC 이해

◆ ESP32 SoC 란?

- ESPRESSIF사에서 Xtensa Silicon IP를 도입하여 설계하여 판매하는 SoC 제품
- <https://www.espressif.com/en/products/socs/esp32-s3>

◆ ESP32 SoC 특징

- ESP32는 소형 IoT기기에 탑재하기 위한 무선 통신 컨트롤러 장치
- Wi-Fi와 Bluetooth가 통합된 SoC모듈
- 주로 모바일, 웨어러블 디바이스 및 IoT통신 제품에 탑재하기 위해 설계됨

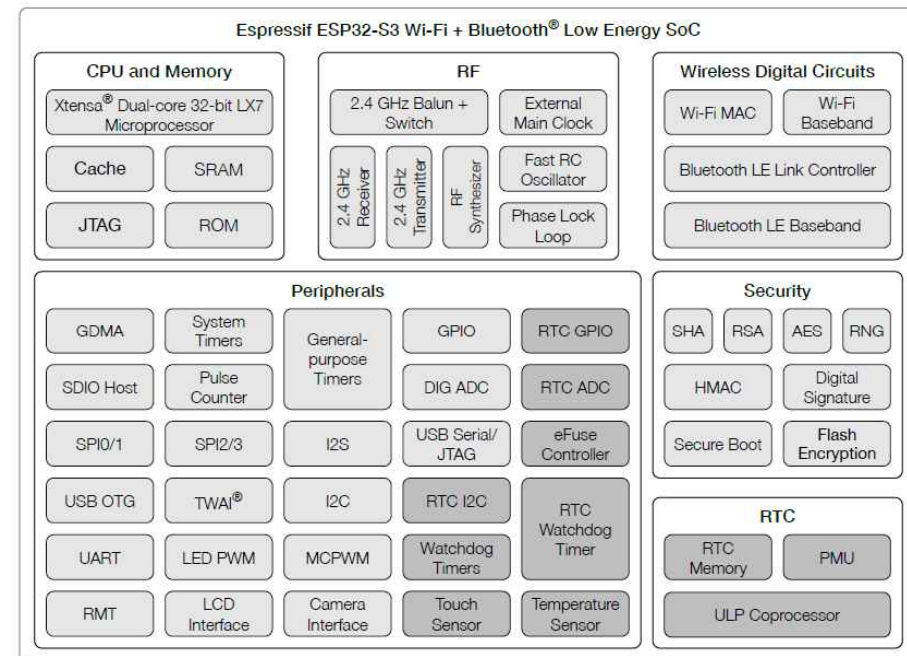
◆ ESP32-S3 HW spec.

- Designed for AIoT application
- Processor : Xtensa® dual-core 32-bit LX7 microprocessors, upto 240MHz
- Memory: 384 KB ROM, 512 KB SRAM, 16 KB SRAM in RTC
- 2.4 GHz Wi-Fi & Bluetooth 5 (BLE)
- Peripherals
 - ❖ 45 programmable GPIOs, SPI, I2S, I2C, PWM, RMT, ADC and UART, SD/MMC host and TWAI™.
 - ❖ 14 GPIOs can be configured as capacitive touch input for HMI applications
- Ultra-low-power(ULP) core
 - ❖ supports multiple low-power modes in a variety of such use-cases

ESP32-S3 SoC 구조

◆ 단일 칩으로 구성

- MCU-based system on a chip (SoC)
- 칩 내부에 CPU를 비롯한 네트워크 및 IO 디바이스 Integration
- Xtensa® dual-core 32-bit LX7 microprocessors
- 2.4 GHz Wi-Fi & Bluetooth 5 (BLE)



ESP32-S3 Block Diagram

ESP32-S3 : CPU & On-Chip Memory

◆ CPU(Processor)

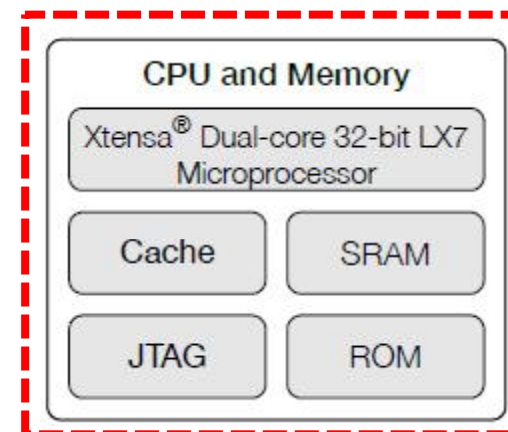
- Xtensa® dual-core 32-bit LX7 microprocessor, up to 240 MHz
- 1 core at 240 MHz: 613.86 CoreMark;
 - ❖ 2.56 CoreMark/MHz
- 2 cores at 240 MHz: 1181.60 CoreMark;
 - ❖ 4.92 CoreMark/MHz
- 128-bit data bus and SIMD commands

◆ On-Chip Memory

- 384 KB ROM
- 512 KB SRAM
- 16 KB SRAM in RTC

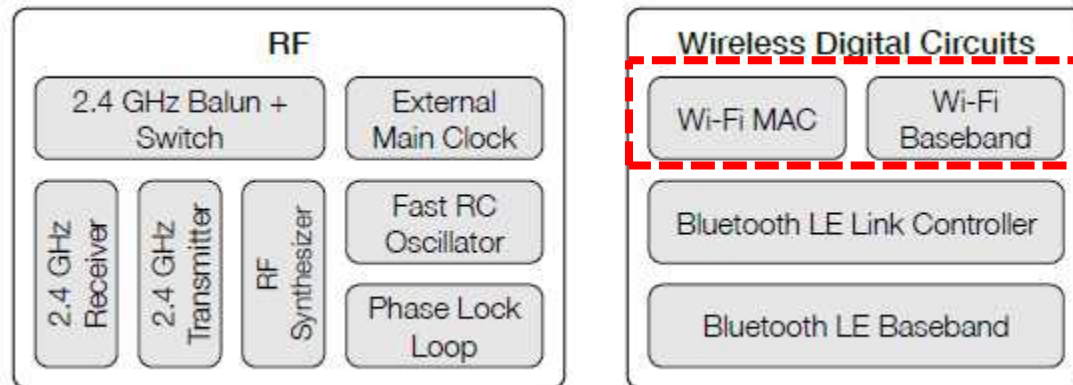
◆ External Flash and external RAM

- SPI, Dual SPI, Quad SPI, Octal SPI, QPI and OPI interfaces
- Flash controller with cache is supported
- Flash in-Circuit Programming (ICP) is supported



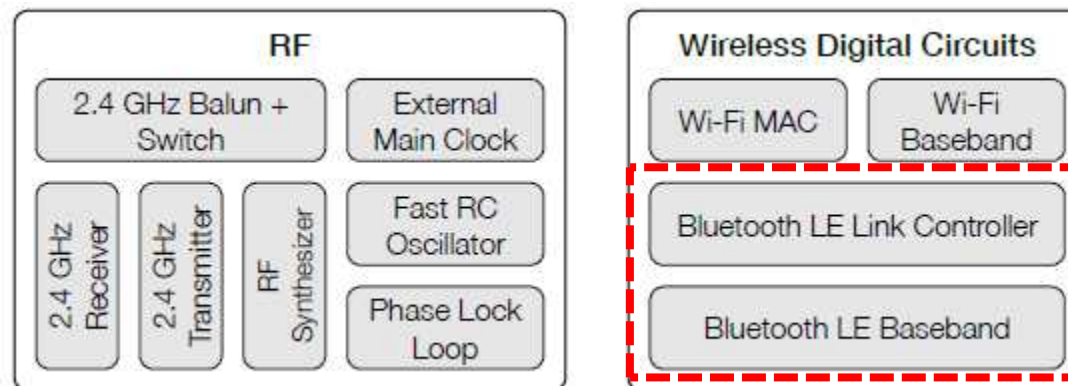
ESP32-S3 : WiFi

- ◆ IEEE 802.11 b/g/n-compliant
 - Supports 20 MHz, 40 MHz bandwidth in 2.4 GHz band
 - 1T1R mode with data rate up to 150 Mbps
- ◆ 4 × virtual Wi-Fi interfaces
 - Simultaneous support for Infrastructure BSS Station, SoftAP, or Station + SoftAP modes
- ◆ Wi-Fi Multimedia (WMM)
- ◆ TX/RX A-MPDU, TX/RX A-MSDU
- ◆ Fragmentation and defragmentation
- ◆ Automatic Beacon monitoring (hardware TSF)
- ◆ Antenna diversity
- ◆ 802.11mc FTM



ESP32-S3 : Bluetooth

- ◆ Bluetooth LE: Bluetooth 5, Bluetooth mesh
- ◆ High power mode (20 dBm)
- ◆ Speed: 125 Kbps, 500 Kbps, 1 Mbps, 2 Mbps
- ◆ Advertising extensions
- ◆ Multiple advertisement sets
- ◆ Channel selection algorithm #2
- ◆ Internal co-existence mechanism between Wi-Fi and Bluetooth to share the same antenna

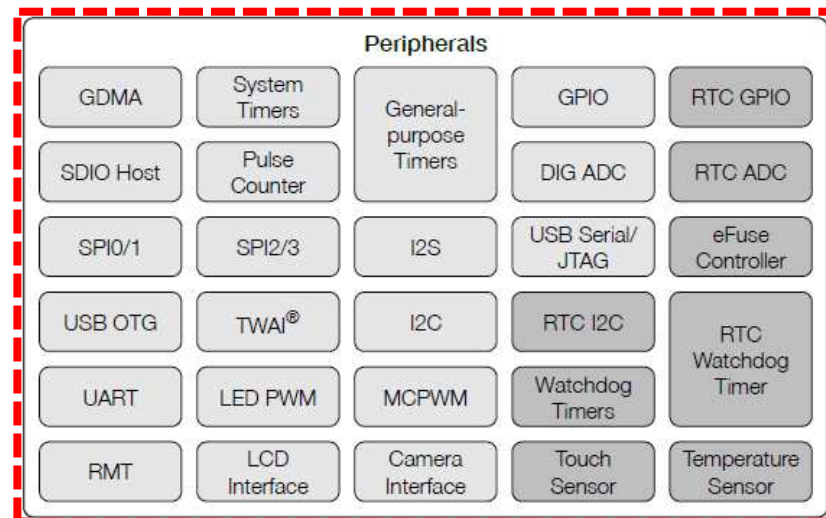


ESP32-S3 : Peripheral

◆ 45 × programmable GPIOs

◆ Digital interfaces:

- 4 × SPI
 - ❖ 8-bit ~16-bit parallel RGB
 - ❖ I8080 and MOTO6800)
- 1 × LCD interface
- 1 × DVP 8-bit ~16-bit camera interface
- 3 × UART
- 2 × I2C
- 2 × I2S
- 1 × RMT (TX/RX)
- 1 × pulse counter LED PWM controller, up to 8 channels
- 1 × full-speed USB OTG
- 1 × USB Serial/JTAG controller
- 2 × MCPWM
- 1 × SDIO host controller
- 1 × TWAI® controller, compatible with ISO 11898-1 (CAN Spec. 2.0)



◆ Analog interfaces:

- 2 × 12-bit SAR ADCs, up to 20 channels
- 1 × temperature sensor
- 14 × touch sensing I/Os

◆ Timers:

- 4 × 54-bit general-purpose timers
- 1 × 52-bit system timer
- 3 × watchdog timers

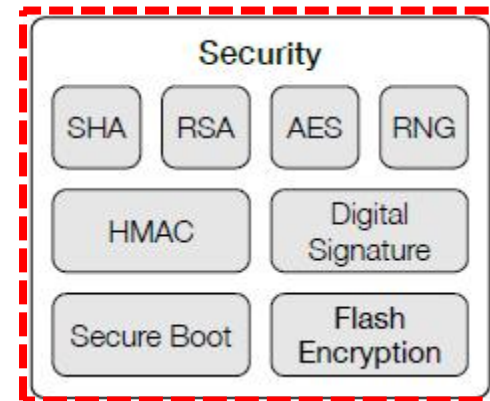
ESP32-S3 : Peripheral (GPIOs)

- ◆ ESP32-S3 GPIO 핀은 IO MUX 기능을 지원하여 F0~F4 서로 다른 기능 제공
 - IO MUX는 ESP32-S3의 각 핀을 GPIO 및 특정 Peripheral 인터페이스로 사용 가능
 - IO MUX 적용 예

Pin	IO MUX GPIO Name	IO MUX Function									
		F0		F1		F2		F3		F4	
		0	Type	1	Type	2	Type	3	Type	4	Type
13	GPIO8	GPIO8	I/O/T	GPIO8	I/O/T			SUBSPICS1	O/T		
14	GPIO9	GPIO9	I/O/T	GPIO9	I/O/T			SUBSPIHD	I1/O/T	FSPIHD	I1/O/T
15	GPIO10	GPIO10	I/O/T	GPIO10	I/O/T	FSPIIO4	I1/O/T	SUBSPICS0	O/T	FSPICS0	I1/O/T
16	GPIO11	GPIO11	I/O/T	GPIO11	I/O/T	FSPIIO5	I1/O/T	SUBSPID	I1/O/T	FSPID	I1/O/T
17	GPIO12	GPIO12	I/O/T	GPIO12	I/O/T	FSPIIO6	I1/O/T	SUBSPICLK	O/T	FSPICLK	I1/O/T
18	GPIO13	GPIO13	I/O/T	GPIO13	I/O/T	FSPIIO7	I1/O/T	SUBSPIQ	I1/O/T	FSPIQ	I1/O/T
19	GPIO14	GPIO14	I/O/T	GPIO14	I/O/T	FSPIIDQS	O/T	SUBSPIWP	I1/O/T	FSPIWP	I1/O/T

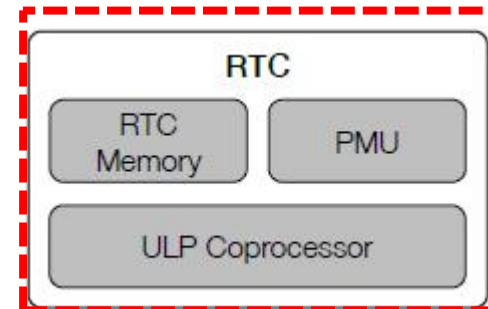
ESP32-S3 : Security

- ◆ Secure boot
- ◆ Flash encryption
- ◆ 4-Kbit OTP, up to 1792 bits for users
- ◆ Cryptographic hardware acceleration:
 - AES-128/256 (FIPS PUB 197)
 - Hash (FIPS PUB 180-4)
 - RSA
 - Random Number Generator (RNG)
 - HMAC
 - Digital signature




ESP32-S3 : RTC & Low Power Management

- ◆ RTC (Real Time Clock)
- ◆ PMU (Power Management Unit)
 - Low Power Management
 - Power Management Unit with five power modes
 - ❖ Active, Sleep mode 제어
- ◆ Ultra-Low-Power (ULP) coprocessors
 - ULP-RISC-V coprocessor
 - ULP-FSM coprocessor



ESP32-S3 기술 자료

◆ <https://www.espressif.com/en/support/download/all>

 **ESPRESSIF**

Products Solutions Support Ecosystem Company Join Us Contact Us

🔍 中文 Subscribe

Support > Download > All

All SDKs & Demos Apps Tools AT

Filter Clear

Product

- ☒ ESP32-S3
- ☐ ESP32-S2
- ☐ ESP32-C3
- ☐ ESP32
- ☐ ESP8266

Technology

- ☐ ESP-MESH
- ☐ ESP-BluFi
- ☐ ESP-Drone
- ☐ ESP-TOUCH
- ☐ ESP Provisioning
- ☐ Evaluation Kit

Found 13 results Expand all + Download selected

<input type="checkbox"/>	Title	Platform	Version	Release Date	Download
<input type="checkbox"/>	+ Flash Download Tools	Windows PC	V3.9.4	2023.02.21	↓
<input type="checkbox"/>	+ ESP-IDF (ESP32, ESP32-S, ESP32-C, ESP32-H)	RTOS SDK	V5.0.1	2023.02.15	↓
<input type="checkbox"/>	+ ESP SoftAP Provisioning for iOS	iOS	V2.1.0	2022.10.08	↗
<input type="checkbox"/>	+ ESP BLE Provisioning for iOS	iOS	V2.1.0	2022.10.08	↗
<input type="checkbox"/>	+ ESP SoftAP Provisioning for Android	Android	V2.0.13	2022.10.07	↗
<input type="checkbox"/>	+ ESP BLE Provisioning for Android	Android	V2.0.13	2022.10.07	↗
<input type="checkbox"/>	+ ESP RF Test Tool and Test Guide	ZIP	V2.8	2021.11.10	↓

ESP32-S3 모듈

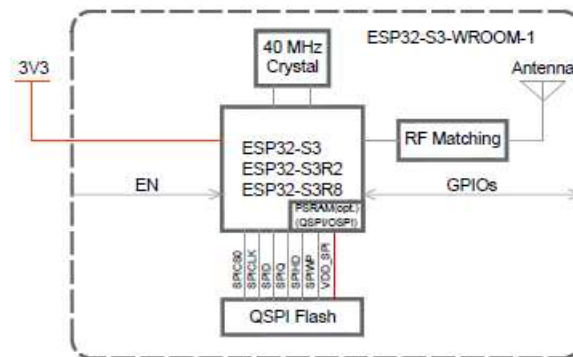
◆ ESP32-S3 SoC를 이용하여 사용자가 사용하기 쉽게 모듈화 설계

- ESP32-S3 SoC와 외부 연결이 용이하도록 설계
- WiFi/BT RF 특성 튜닝 된 안테나 제공

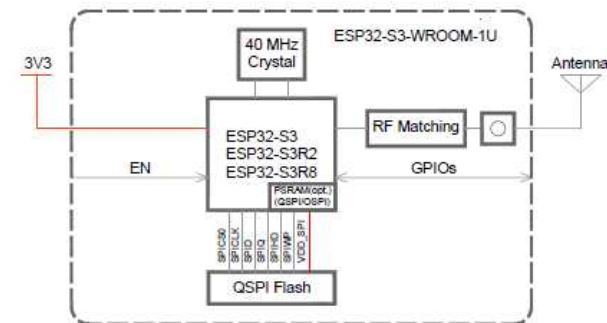


◆ ESP32-S3 WROOM

- 2.4 GHz WiFi(802.11 b/g/n) and Bluetooth® 5 (LE) module
- Built around ESP32S3 series of SoCs, Xtensa® dualcore 32bit LX7 microprocessor
- Flash up to 16 MB, PSRAM up to 8 MB
- 36 GPIOs, rich set of peripherals
- Onboard PCB antenna



ESP32-S3-WROOM-1 Block Diagram



ESP32-S3-WROOM-1U Block Diagram

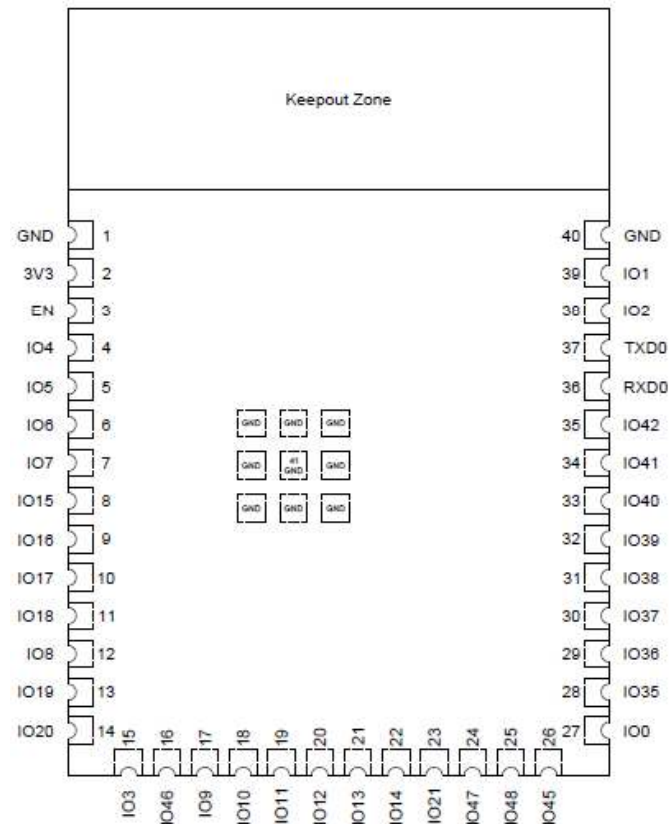
ESP32-S3-WROOM

- ◆ ESP32-S3를 이용한 임베디드 보드 개발시 대부분 ESP32-S3 모듈 사용
- ◆ ESP32-S3 모듈 사용시 장점
 - 즉시 제품화 가능하도록 대부분의 하드웨어 설계 완료 됨
 - ❖ 일부 필요한 센서 등 IO 장치만 추가로 설계하여 사용 가능
 - WiFi/BT 안테나 회로 구현
 - ❖ 별도로 안테나 성능을 높이기 위한 RF 설계 불필요
 - ❖ 제품화에 필요한 RF 인증시험 완료된 모듈 사용하여 제품화 용이
- ◆ ESP32-S3-WROOM-1 시리즈

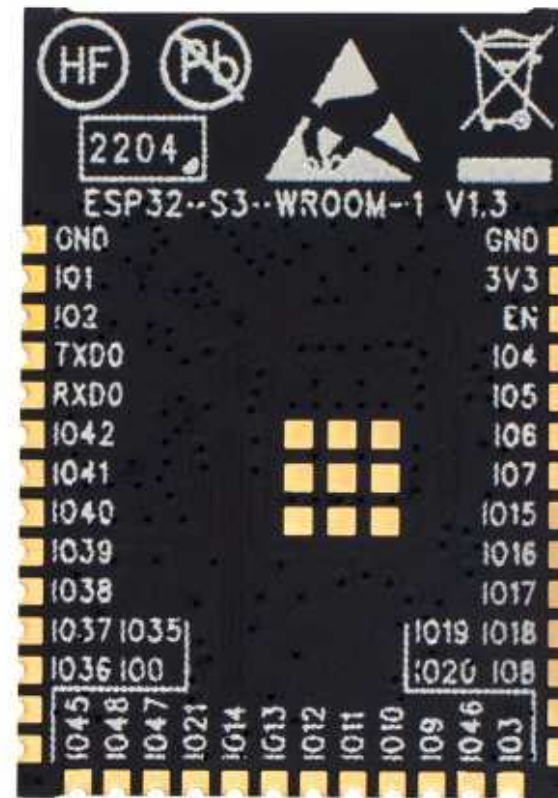
Ordering Code	Flash ²	PSRAM	Ambient Temp. ³ (°C)	Size ⁴ (mm)
ESP32-S3-WROOM-1-N4	4 MB (Quad SPI)	-	-40 ~ 85	18.0 x 25.5 x 3.1
ESP32-S3-WROOM-1-N8	8 MB (Quad SPI)	-	-40 ~ 85	
ESP32-S3-WROOM-1-N16	16 MB (Quad SPI)	-	-40 ~ 85	
ESP32-S3-WROOM-1-H4	4 MB (Quad SPI)	-	-40 ~ 105	
ESP32-S3-WROOM-1-N4R2	4 MB (Quad SPI)	2 MB (Quad SPI)	-40 ~ 85	
ESP32-S3-WROOM-1-N8R2	8 MB (Quad SPI)	2 MB (Quad SPI)	-40 ~ 85	
ESP32-S3-WROOM-1-N16R2	16 MB (Quad SPI)	2 MB (Quad SPI)	-40 ~ 85	
ESP32-S3-WROOM-1-N4R8	4 MB (Quad SPI)	8 MB (Octal SPI)	-40 ~ 65	
ESP32-S3-WROOM-1-N8R8	8 MB (Quad SPI)	8 MB (Octal SPI)	-40 ~ 65	
ESP32-S3-WROOM-1-N16R8	16 MB (Quad SPI)	8 MB (Octal SPI)	-40 ~ 65	

ESP32-S3-WROOM-1

- ◆ ESP32-S3-WROOM 모듈의 I/O interface는 제공된 datasheet을 통하여 확인 가능
 - [esp32-s3-wroom-1_wroom-1u_datasheet_en.pdf](#)



▲ ESP32-S3-WROOM-1 Pin Layout (Top View)



▲ ESP32-S3-WROOM-1 PCB (Bottom View)

ESP32-S3-WROOM : Pin Description (1)

Name	No.	Type	Function
GND	1	P	Ground
3V3	2	P	Power supply
EN	3	I	Module-enable signal. Active high.
IO4	4	I/O/T	RTC_GPIO4, GPIO4, TOUCH4, ADC1_CH3
IO5	5	I/O/T	RTC_GPIO5, GPIO5, TOUCH5, ADC1_CH4
IO6	6	I/O/T	RTC_GPIO6, GPIO6, TOUCH6, ADC1_CH5
IO7	7	I/O/T	RTC_GPIO7, GPIO7, TOUCH7, ADC1_CH6
IO15	8	I/O/T	RTC_GPIO15, GPIO15, U0RTS, ADC2_CH4, XTAL_32K_P
IO16	9	I/O/T	RTC_GPIO16, GPIO16, U0CTS, ADC2_CH5, XTAL_32K_N
IO17	10	I/O/T	RTC_GPIO17, GPIO17, U1TXD, ADC2_CH6
IO18	11	I/O/T	RTC_GPIO18, GPIO18, U1RXD, ADC2_CH7, CLK_OUT3
IO8	12	I/O/T	RTC_GPIO8, GPIO8, TOUCH8, ADC1_CH7, SUBSPICS1
IO19	13	I/O/T	RTC_GPIO19, GPIO19, U1RTS, ADC2_CH8, CLK_OUT2, USB_D
IO29	14	I/O/T	RTC_GPIO20, GPIO20, U1CTS, ADC2_CH9, CLK_OUT1, USB_D+
IO3	15	I/O/T	RTC_GPIO3, GPIO3, TOUCH3, ADC1_CH2
IO46	16	I/O/T	GPIO46
IO9	17	I/O/T	RTC_GPIO9, GPIO9, TOUCH9, ADC1_CH8, FSPIHD, SUBSPIHD
IO10	18	I/O/T	RTC_GPIO10, GPIO10, TOUCH10, ADC1_CH9, FSPICS0, FSPIIO4, SUBSPICS0
IO11	19	I/O/T	RTC_GPIO11, GPIO11, TOUCH11, ADC2_CH0, FSPID, FSPIIO5, SUBSPID
IO12	20	I/O/T	RTC_GPIO12, GPIO12, TOUCH12, ADC2_CH1, FSPICLK, FSPIIO6, SUBSPICLK

ESP32-S3-WROOM : Pin Description (2)

Name	No.	Type	Function
IO13	21	I/O/T	RTC_GPIO13, GPIO13, TOUCH13, ADC2_CH2, FSPIQ, FSPIIO7, SUBSPIQ
IO14	22	I/O/T	RTC_GPIO14, GPIO14, TOUCH14, ADC2_CH3, FSPIWP, FSPIDQS, SUBSPIWP
IO21	23	I/O/T	RTC_GPIO21, GPIO21
IO47	24	I/O/T	SPICLK_P_DIFF,GPIO47, SUBSPICLK_P_DIFF
IO48	25	I/O/T	SPICLK_N_DIFF,GPIO48, SUBSPICLK_N_DIFF
IO45	26	I/O/T	GPIO45
IO0	27	I/O/T	RTC_GPIO0, GPIO0
IO35	28	I/O/T	SPIIO6, GPIO35, FSPID, SUBSPID
IO36	29	I/O/T	SPIIO7, GPIO36, FSPICLK, SUBSPICLK
IO37	30	I/O/T	SPIDQS, GPIO37, FSPIQ, SUBSPIQ
IO38	31	I/O/T	GPIO38, FSPIWP, SUBSPIWP
IO39	32	I/O/T	MTCK, GPIO39, CLK_OUT3, SUBSPICS1
IO40	33	I/O/T	MTDO, GPIO40, CLK_OUT2
IO41	34	I/O/T	MTDI, GPIO41, CLK_OUT1
IO42	35	I/O/T	MTMS, GPIO42
RXD0	36	I/O/T	U0RXD, GPIO44, CLK_OUT2
TXD0	37	I/O/T	U0TXD, GPIO43, CLK_OUT1
IO2	38	I/O/T	RTC_GPIO2, GPIO2, TOUCH2, ADC1_CH1
IO1	39	I/O/T	RTC_GPIO1, GPIO1, TOUCH1, ADC1_CH0
GND	40	P	GND

ESP32 SoC 이해

◆ ESP32 SoC 란?

- ESPRESSIF사에서 Xtensa Silicon IP를 도입하여 설계하여 판매하는 SoC 제품
- <https://www.espressif.com/en/products/socs/esp32>

◆ ESP32 SoC 특징

- ESP32는 소형 IoT기기에 탑재하기 위한 무선 통신 컨트롤러 장치
- Wi-Fi와 Bluetooth가 통합된 SoC모듈
- 주로 모바일, 웨어러블 디바이스 및 IoT통신 제품에 탑재하기 위해 설계됨

◆ HW spec.

- Processor : Xtensa® single-/dual-core 32-bit LX6 microprocessor(s),
up to 600 MIPS (200 MIPS for ESP32-S0WD, 400 MIPS for ESP32-D2WD)
- Memory: 448 KB ROM, 520 KB SRAM, 16 KB SRAM in RTC
- 2.4 GHz Wi-Fi & Bluetooth
- Peripherals : Capacitive touch sensors, Hall sensor, SD card interface, Ethernet, high-speed SPI, UART, I2S and I2C
- Ultra-low-power

목 차

INL(IoT Node Link)

ESP32-S3

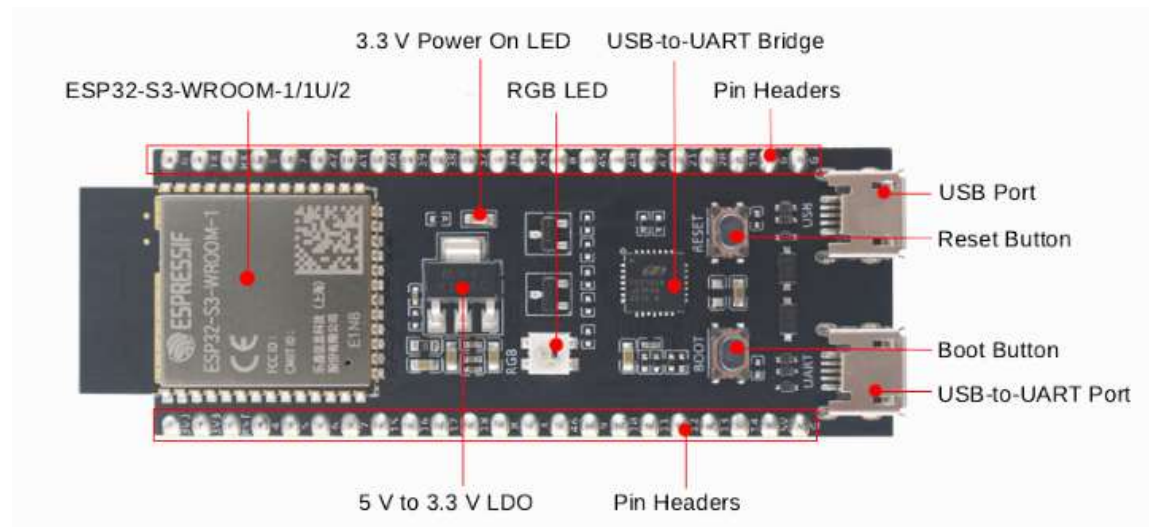
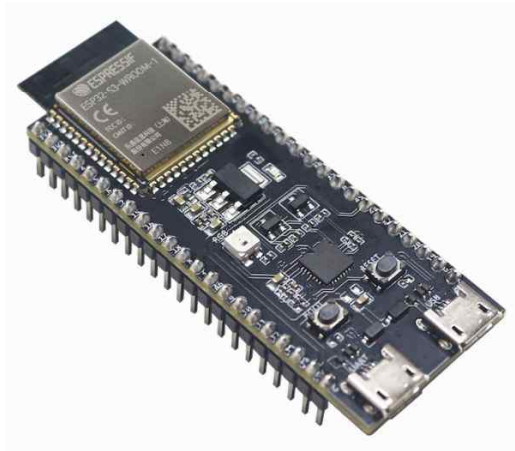
◆ **ESP32-S3-DevKitC-1**

ESP32-S3 개발 환경

Arduino Platform

ESP32-S3-DevKitC-1 Board - Overview

- ◆ ESP32-S3 프로세서가 탑재된 개발 보드
 - <https://docs.espressif.com/projects/esp-idf/en/latest/esp32s3/hw-reference/esp32s3/user-guide-devkitc-1.html>



ESP32-S3-DevKitC-1 주요 구성

◆ ESP32-S3-DevKitC-1 주요 컴포넌트 구성 (그림 참조)

Component	용도
ESP32-S3-WROOM-1/1U/2	ESP32-S3-WROOM-1, ESP32-S3-WROOM-1U, and ESP32-S3-WROOM-2 모듈
5 V to 3.3 V LDO	5V를 3.3V로 변환하는 Regulator
Pin Headers	All available GPIO pins (except for the SPI bus for flash)
USB-to-UART Port	A Micro-USB port used for 1) Power supply to the board 2) Flashing applications to the chip 3) Communication with the chip via the on-board USB-to-UART bridge.
Boot Button	Download button. 시리얼 포트로 펌웨어 다운로드 할 때 Boot 버튼을 누르고 Reset 누른다
Reset Button	시스템 재시작 버튼
USB Port	USB1.1 호환, ESP32-S3 full-speed USB OTG 인터페이스 1) Power supply to the board 2) Flashing applications to the chip 3) Communication with the chip using USB 1.1 protocols 4) JTAG debugging.
USB-to-UART Bridge	Single USB-to-UART bridge chip provides transfer rates up to 3 Mbps.
RGB LED	RGB LED, GPIO38 핀으로 제어
3.3 V Power On LED	USB 전원 연결 시 ON

37



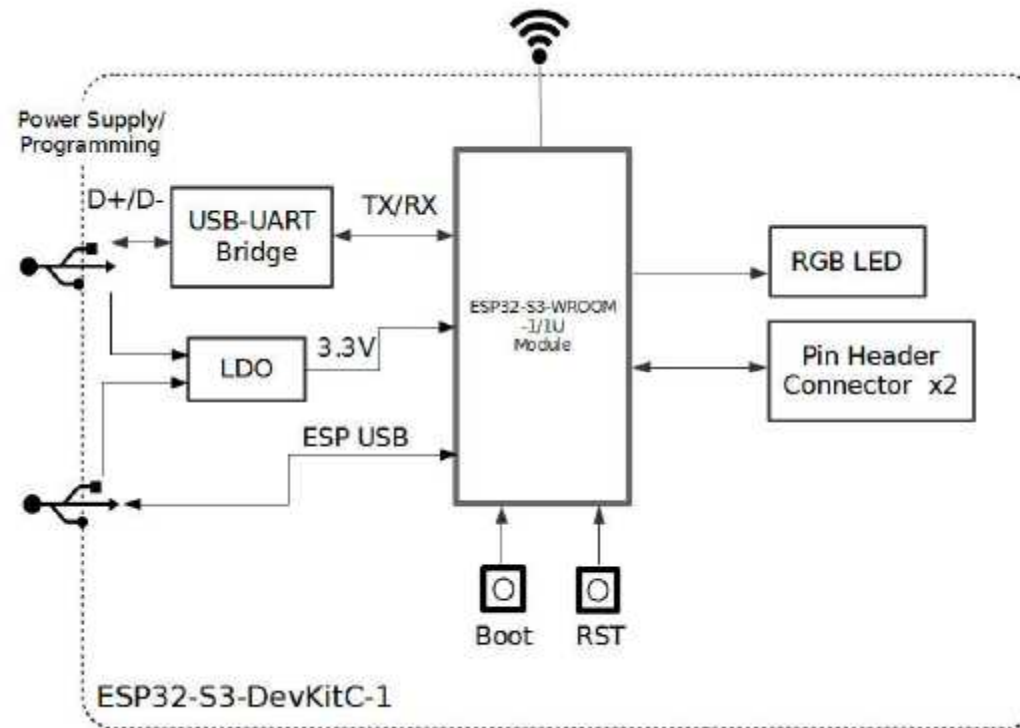
ESP32-S3-DevKitC-1 헤더 (J1)

Name	No.	Type	Function
3V3	1	P	3.3 V power supply
3V3	2	P	3.3 V power supply
RST	3	I	EN
4	4	I/O/T	RTC_GPIO4, GPIO4, TOUCH4, ADC1_CH3
5	5	I/O/T	RTC_GPIO5, GPIO5, TOUCH5, ADC1_CH4
6	6	I/O/T	RTC_GPIO6, GPIO6, TOUCH6, ADC1_CH5
7	7	I/O/T	RTC_GPIO7, GPIO7, TOUCH7, ADC1_CH6
15	8	I/O/T	RTC_GPIO15, GPIO15, U0RTS, ADC2_CH4, XTAL_32K_P
16	9	I/O/T	RTC_GPIO16, GPIO16, U0CTS, ADC2_CH5, XTAL_32K_N
17	10	I/O/T	RTC_GPIO17, GPIO17, U1TXD, ADC2_CH6
18	11	I/O/T	RTC_GPIO18, GPIO18, U1RXD, ADC2_CH7, CLK_OUT3
8	12	I/O/T	RTC_GPIO8, GPIO8, TOUCH8, ADC1_CH7, SUBSPICS1
3	13	I/O/T	RTC_GPIO3, GPIO3, TOUCH3, ADC1_CH2
46	14	I/O/T	GPIO46
9	15	I/O/T	RTC_GPIO9, GPIO9, TOUCH9, ADC1_CH8, FSPIHD, SUBSPIHD
10	16	I/O/T	RTC_GPIO10, GPIO10, TOUCH10, ADC1_CH9, FSPICS0, FSPIIO4, SUBSPICS0
11	17	I/O/T	RTC_GPIO11, GPIO11, TOUCH11, ADC2_CH0, FSPID, FSPIIO5, SUBSPID
12	18	I/O/T	RTC_GPIO12, GPIO12, TOUCH12, ADC2_CH1, FSPICLK, FSPIIO6, SUBSPICLK
13	19	I/O/T	RTC_GPIO13, GPIO13, TOUCH13, ADC2_CH2, FSPIQ, FSPIIO7, SUBSPIQ
14	20	I/O/T	RTC_GPIO14, GPIO14, TOUCH14, ADC2_CH3, FSPIWP, FSPIDQS, SUBSPIWP
5V	21	P	5 V power supply
G	22	G	Ground

ESP32-S3-DevKitC-1 헤더 (J3)

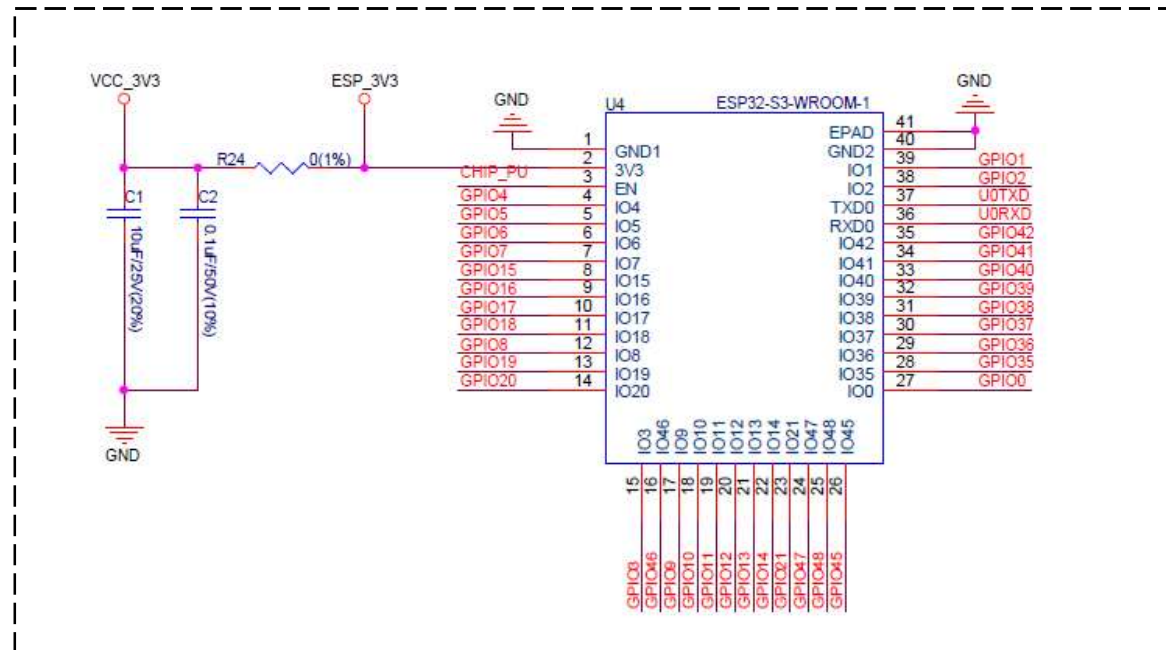
Name	No.	Type	Function
GND	1	G	Ground
TX	2	I/O/T	U0TXD, GPIO43, CLK_OUT1
RX	3	I/O/T	U0RXD, GPIO44, CLK_OUT2
1	4	I/O/T	RTC_GPIO1, GPIO1, TOUCH1, ADC1_CH0
2	5	I/O/T	RTC_GPIO2, GPIO2, TOUCH2, ADC1_CH1
42	6	I/O/T	MTMS, GPIO42
41	7	I/O/T	MTDI, GPIO41, CLK_OUT1
40	8	I/O/T	MTDO, GPIO40, CLK_OUT2
39	9	I/O/T	MTCK, GPIO39, CLK_OUT3, SUBSPICS1
38	10	I/O/T	GPIO38, FSPIWP, SUBSPIWP, RGBLED(V1.1)
37	11	I/O/T	SPIDQS, GPIO37, FSPIQ, SUBSPIQ
36	12	I/O/T	SPIIO7, GPIO36, FSPICK, SUBSPICK
35	13	I/O/T	SPIIO6, GPIO35, FSPID, SUBSPID
0	14	I/O/T	RTC_GPIO0, GPIO0
45	15	I/O/T	GPIO45
48	16	I/O/T	GPIO48, SPICK_N, SUBSPICK_N_DIFF, RGBLED(V1.0)
47	17	I/O/T	GPIO47, SPICK_P, SUBSPICK_P_DIFF
21	18	I/O/T	RTC_GPIO21, GPIO21
20	19	I/O/T	RTC_GPIO20, GPIO20, U1CTS, ADC2_CH9, CLK_OUT1, USB_D+
19	20	I/O/T	RTC_GPIO19, GPIO19, U1RTS, ADC2_CH8, CLK_OUT2, USB_D-
GND	21	G	Ground
GND	22	G	Ground

ESP32-S3-DevKitC-1 블록도



ESP32-S3-DevKitC-1 회로 : ESP32-S3-WROOM-1

- ◆ 제공된 schematic(회로도)을 이용하여 회로 설계 내용 확인
 - SCH_ESP32-S3-DevKitC-1_V1.1_20221130.pdf
- ◆ ESP32-S3-WROOM 모듈을 이용한 회로도 설계
 - 모듈의 I/O interface는 제공된 datasheet을 통하여 확인 가능
 - esp32-s3-wroom-1_wroom-1u_datasheet_en.pdf

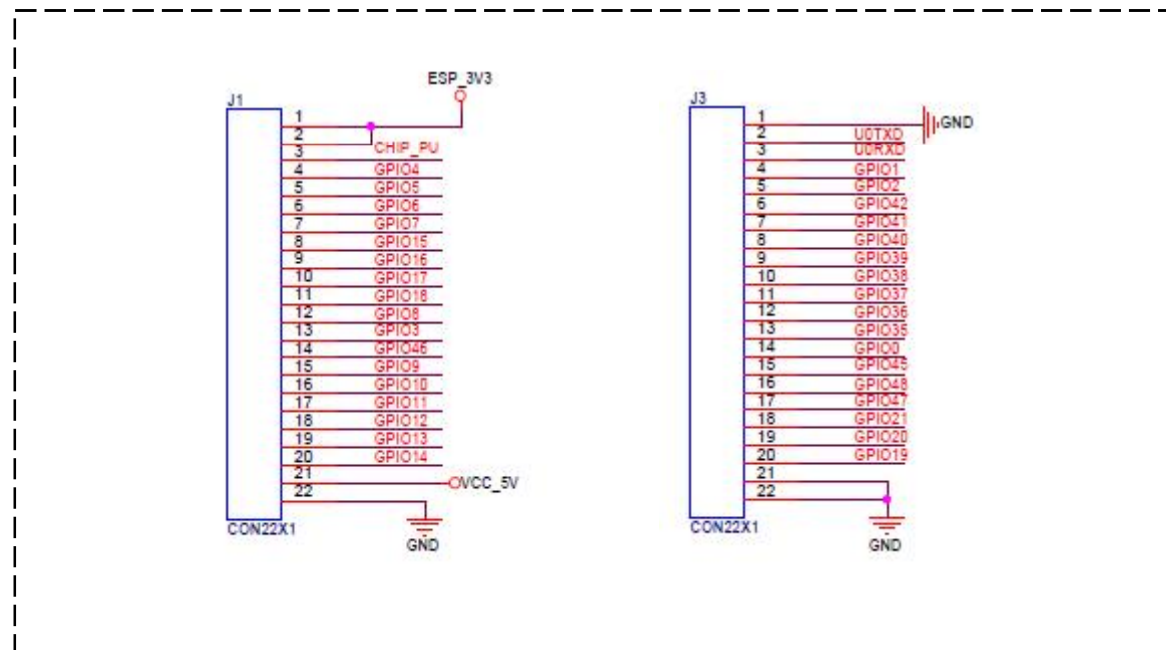


▲ ESP32-S3-WROOM-1 Schematics

ESP32-S3-DevKitC-1 회로 : 핀 헤더

◆ ESP32-S3-DevKitC1-1 외부 핀 헤더 주변 회로 설계

- ESP32-S3의 대부분 핀 지원
- ESP32-S3의 GPIO35, GPIO36, GPIO37 내부 Octal SPI flash/PSRAM 메모리 I/F 용도로 사용, 따라서 사용자는 사용 불가



▲ Pin header (J1, J3)

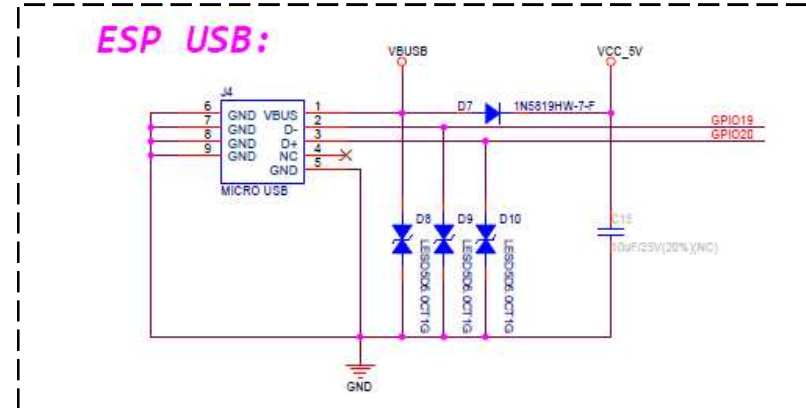
ESP32-S3-DevKitC-1 회로 : USB & UART

◆ ESP32-S3 USB I/F 지원

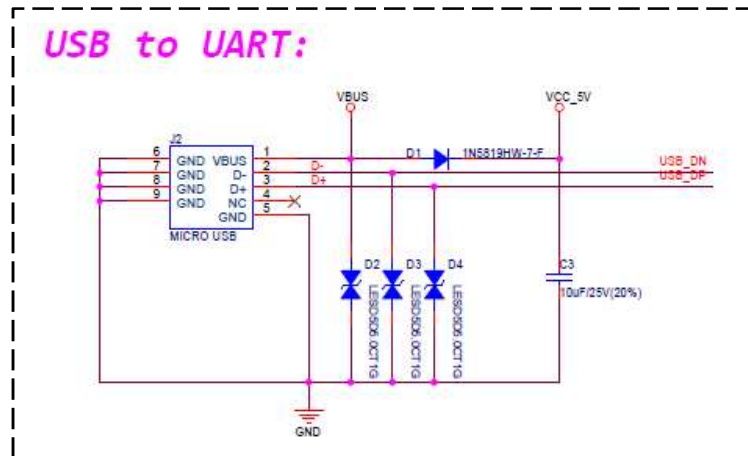
- ESP32-S3 USB OTG

◆ ESP32-S3 USB to UART 지원

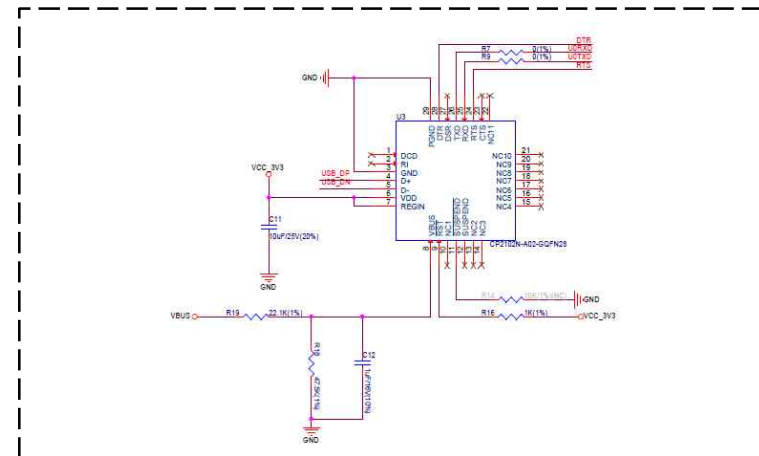
- USB to UART Bridge 적용(CP2102N)
- ESP32-S3의 UART0포트를 통해 데이터 신호 입출력
- RTS(request To Send)와 DTR(Data Terminal Ready)을 통한 하드웨어 통신제어



▲ USB OTG Interface



▲ USB to UART Interface



▲ USB to UART bridge IC schematic

ESP32-S3-DevKitC-1 회로 : Power

◆ Power

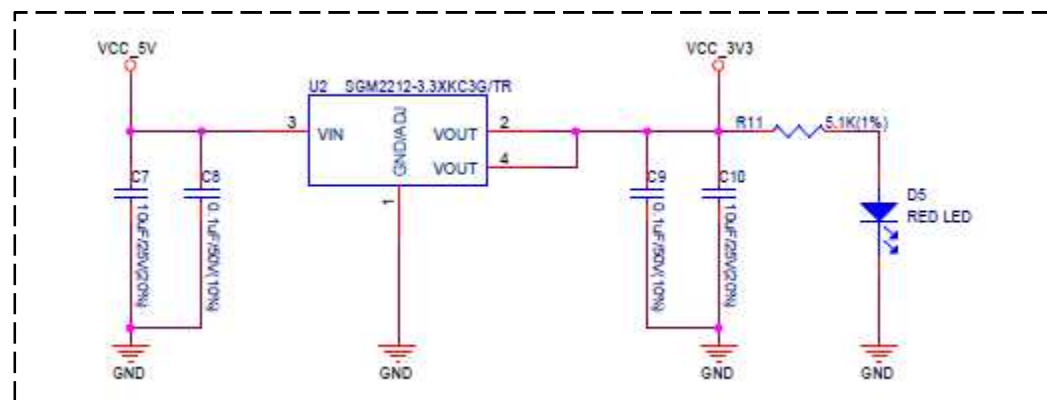
- 동작 전압범위는 2.3V~3.6V
- 단일 전원 공급 장치 사용할 경우 권장 전압은 3.3V, 500mA이상

◆ 전원 공급

- 2개의 USB장치 중 하나를 통해 공급이 가능
- USB-to-UART port
- USB Port

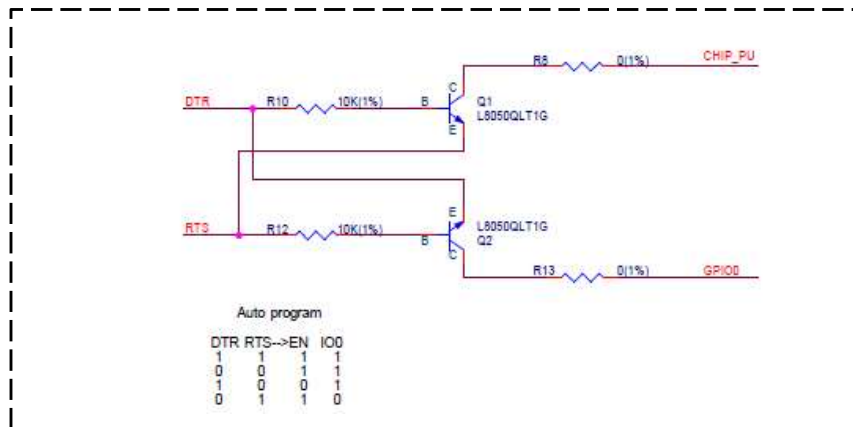
◆ 전원 공급 회로

- USB장치에서 공급받은 5V전원은 레귤레이터(Regulator)를 통해 3.3V 전원 생성
- 외부확장인터페이스(Connector Interface)의 경우 3.3V~5V전압의 입출력장치를 연결하기 위해 Power Enable된 5V전압을 확장인터페이스로 인가

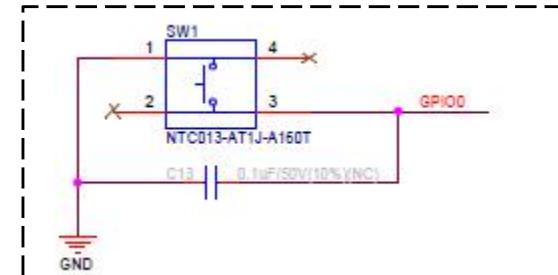


ESP32-S3-DevKitC-1 회로 : Switch & Button

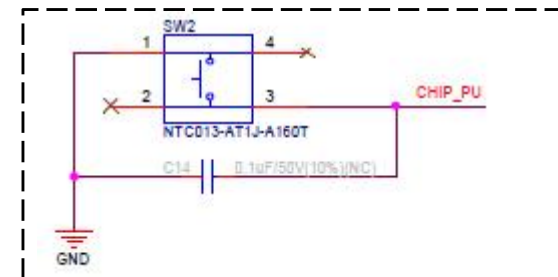
- ◆ 부트 버튼 (SW1)
 - GPIO0 에 연결
- ◆ 리셋 버튼 (SW2)
 - CHIP_PU 에 연결 (EN pin of ESP32-S3)
- ◆ Programming Mode
 - 시리얼 포트로 펌웨어 다운로드 할 때 **Boot** 버튼을 누르고 **Reset** 누른다
- ◆ Auto Program 지원
 - DTR/RTS 신호를 이용한 자동 프로그래밍 모드 지원



▲ Auto Programming mode 지원



▲ Boot 버튼 (SW1)

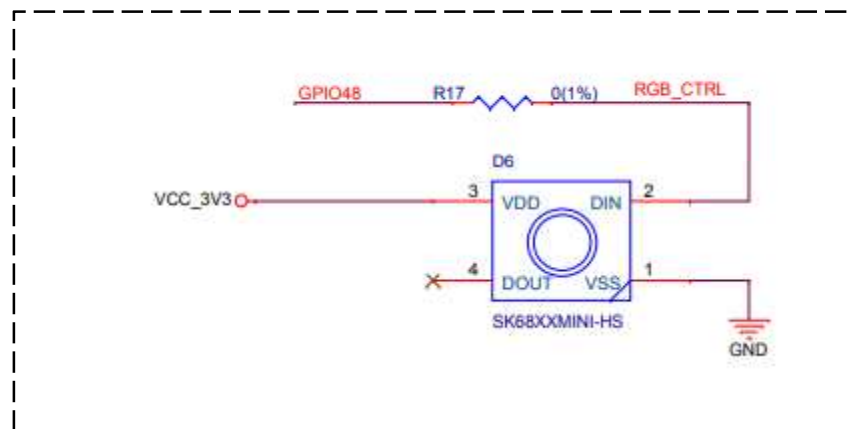


▲ Reset 버튼 (SW2)

ESP32-S3-DevKitC-1 회로 : RGB LED

◆ RGB LED on GPIO48

Name	No.	Type	Function
48	16	I/O/T	GPIO48, SPICLK_N, SUBSPICLK_N_DIFF



▲ RGB LED

Note

Both versions(V1.0 & V1.1) of ESP32-S3-DevKitC-1 are available on the market.
The main difference lies in that the RGB LED is connected to different pins.
(V1.0 connected to GPIO48, V1.1 connected to GPIO38)

목 차

INL(IoT Node Link)

ESP32-S3

ESP32-S3-DevKitC-1

◆ **ESP32-S3 개발 환경**

Arduino Platform

소프트웨어 개발 환경

◆ 교차 개발 환경 (Cross Development Platform)

- 소프트웨어를 개발하는 시스템(컴퓨터)과 실제 실행하는 시스템이 서로 다른 개발 환경

◆ 호스트(Host) 시스템과 타겟(Target) 시스템

- 호스트 시스템 : 개발환경을 가지고 소프트웨어를 개발하고 빌드하는 컴퓨터 (PC)
 - ❖ Windows, Linux, Mac OS 등 다양한 환경 구성 가능
- 타겟 시스템 : 개발하고자 하는 임베디드 시스템 (ESP32-S3-DevKitC-1 / INL)

◆ 교차 개발 환경 구성 요소

- 크로스 컴파일러 / 툴체인
- 디버거, 에디터 등
- 통합개발환경(IDE)



▲ Target : ESP32-S3-DevKitC-1
INL Serial Device

USB Cable
(전원 & Debug)

USB-A

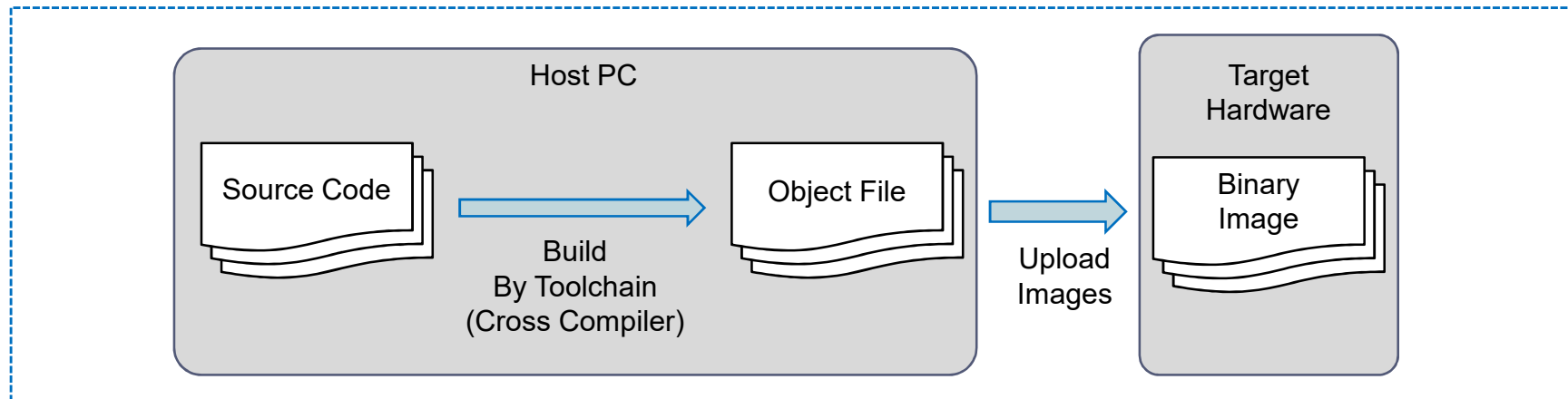
USB Micro



▲ Host : Desktop or Laptop

ESP32 개발 환경 설정

- ◆ 소스코드 편집기 : VSCode, Eclipse, VI 에디터
- ◆ ESP32 툴체인 : 어셈블러, 컴파일러, 링커, 디버거 등
 - SDK 형태로 툴체인을 포함하여 API, Library, 샘플 소스코드를 포함한 툴체인 사용
 - ❖ SDK : Software Development Kit
 - ❖ API : Application Programming Interface
 - ESP32-IDF, ESP32 Arduino, NodeMCU, MicroPython 등 다양함



- ◆ Reference
 - ESP32-IDF : <https://docs.espressif.com/projects/esp-idf/en/latest/esp32/get-started/index.html>
 - Arduino ESP32 : <https://docs.espressif.com/projects/arduino-esp32/en/latest/index.html>

목 차

INL(IoT Node Link)

ESP32-S3

ESP32-S3-DevKitC-1

ESP32-S3 개발 환경

◆ **Arduino Platform**



Arduino Platform

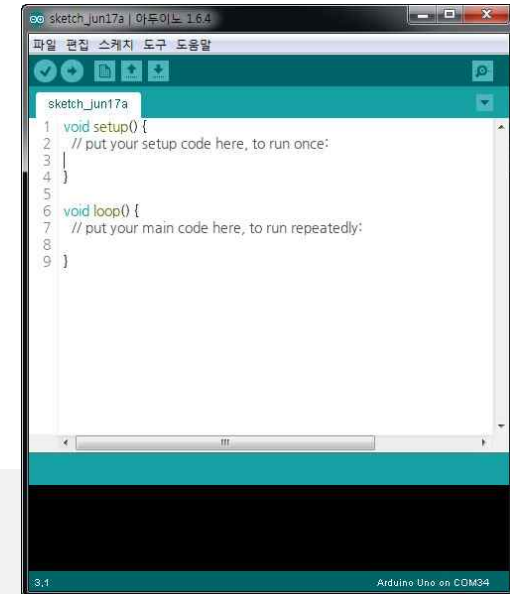
- ◆ Arduino is an open–source electronics platform based on easy–to–use hardware and software
- ◆ Basic structure
 - read inputs (sensor) turn it into an output (actuator)
- ◆ What to be done by board between sensing and actuating
 - a set of instructions to the microcontroller on the board
- ◆ Started from AVR, now extended to various microcontrollers (e.g., ARM Cortex M–series)
- ◆ Reference :
 - <https://www.arduino.cc>

Arduino Programming

- ◆ Based on C/C++
- ◆ Arduino IDE
 - <https://www.arduino.cc/en/software>
- ◆ Basic structure of Arduino Programming

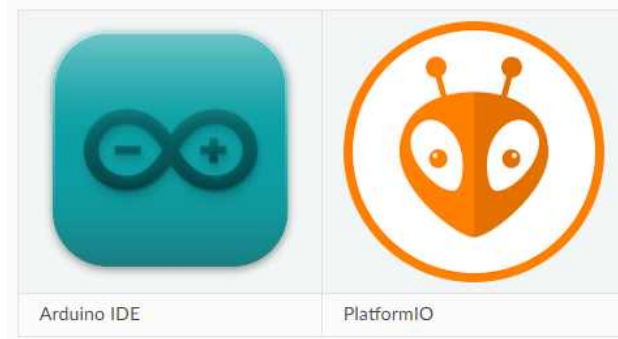
```
void setup() {  
    // put your setup code here, to run once:  
}
```

```
void loop() {  
    // put your main code here, to run repeatedly:  
}
```



Arduino ESP32

- ◆ Arduino API를 ESP32에서 사용 가능하도록 구성
- ◆ IDE 지원



- Arduino ESP32 IDE Install
 - ❖ <https://docs.espressif.com/projects/arduino-esp32/en/latest/installing.html>
 - ❖ Arduino IDE 설치 시 최신 버전을 설치하는 경우 ESP32용으로 정상동작 하지 않음 (1.8.xx 버전 사용 권장)
- ◆ Community
 - ESP32 Forum : <https://esp32.com>
 - ESP32 Arduino Forum : <https://esp32.com/viewforum.php?f=19>
 - ESP32 Hardware Forum : <https://esp32.com/viewforum.php?f=12>
 - Document : <https://docs.espressif.com/projects/arduino-esp32/en/latest/index.html>

Arduino API Reference

◆ <https://www.arduino.cc/reference/en/>

- Function, variable(variable & constant) 및 structure 정의
- 상세 내용은 reference site 참조

구분	함수 이름	설명
Digital	pinMode(pin, mode)	입력된 핀번호에 해당하는 핀의 동작(Input, Output)을 설정 한다.
	digitalWrite(pin, value)	입력된 핀번호에 해당하는 핀으로 디지털값(High, Low)값을 출력 한다.
	digitalRead(pin)	입력된 핀번호로부터 디지털 값(High, Low)을 읽는다.
Analog	analogRead(pin)	입력된 핀번호에 해당하는 핀으로부터 아날로그신호를 읽는다.
	analogWrite(pin, value)	입력된 핀번호에 해당하는 핀으로 다음 analogWrite()가 올 때까지 특정 듀티사이클을 가지는 PWM(Pulse Width Modulation 펄스 폭 변조)을 출력한다.
Serial	Serial.begin(speed)	시리얼통신의 속도를 설정한다.
	Serial.available()	시리얼포트로부터 시리얼데이터의 바이트크기를 읽어 반환 한다
	Serial.flush()	시리얼포트안에 존재하는 데이터를 비운다.
	Serial.println(val)	시리얼데이터를 아스키코드형식에 맞추어서 출력한다.
	Serial.read()	시리얼 데이터를 읽는다.
Time	delay()	지정된 밀리초 시간만큼 프로그램을 일시 중지한다.
	delayMicroseconds()	지정된 마이크로초 시간만큼 프로그램을 일시 중지한다.
	micros()	현재 프로그램이 시작된 이후의 경과시간을 마이크로초 단위로 반환한다.
	millis()	현재 프로그램이 시작된 이후의 경과시간을 밀리초 단위로 반환한다.

Arduino API Reference (cont'd)

구분	함수 이름	설명
Math	min(x, y)	입력된 두 값중 최소값을 반환 한다.
	max(x, y)	입력된 두 값중 최대값을 반환 한다.
	abs(x)	입력된 값의 절대값을 반환 한다.
	constrain(x, a, b)	첫번째 인자값을 두번째 인자값과 세번째 인자값의 사이값으로 제한한다.
	map()	첫번째 인자값을 지정된 범위로 선형사상하여 반환 한다.
	pow(base, exponent)	입력한 밑과 지수값으로 거듭제곱하여 값을 반환 한다.
	sqrt(x)	입력한 값을 루트씩워 계산하여 반환 한다.
Interrupt	interrupts()	nointerrupt()에 의해 금지된 인터럽트의 발생을 허용한다.
	noInterrupts()	인터럽트의 발생을 금지 시킨다
Bits and Bytes	lowByte(x)	입력된 값의 최하위 바이트를 추출 한다.
	highByte(x)	입력된 값의 최상위 바이트를 추출 한다.
	bitRead(x, n)	주어진 데이터의 n번째 비트를 읽어 반환 한다.
	bitWrite(x, n, b)	주어진 데이터의 n번째 비트에 b값(0또는1)을 입력한다.
	bitSet(x, n)	주어진 데이터의 n번째 비트를 1로 설정 한다.
	bitClear(x, n)	주어진 데이터의 n번째 비트를 0으로 설정 한다.
	bit(n)	지정된 비트 위치(n번째)에 해당하는 비트 값을 계산하여 반환 한다.

Arduino API Reference (cont'd)

Advanced IO	tone(pin, frequency)	입력된 핀번호로 입력된 주파수를 가지는 square wave를 생성한다.
	noTone(pin)	특정 핀에서 tone()함수에 의한 square wave 생성을 중지시킨다.
	shiftOut()	입력된 핀번호로 입력된 값을 출력 순서에 맞게 출력 한다
	shiftIn()	특정핀(dataPin)으로 부터 데이터를 입력받아 비트 순서에 맞게 정렬 후 바이트 단위로 반환 한다.
	pulseIn(pin, value)	특정핀으로부터 Pulse(또는 High or Low)를 읽어서 Pulse의 길이를 마이크로초단위로 반환 한다

통합 개발환경 다운로드 및 설치

◆ Arduino ESP32 IDE 다운로드 및 설치

- <https://www.arduino.cc/en/Main/Software> 링크 접속
- 아래 순서대로 클릭 시, 파일 다운로드
- 다운 받은 zip 파일 압축 해제

Legacy IDE (1.8.X)



Arduino IDE 1.8.19

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. This software can be used with any Arduino board.


Refer to the [Getting Started](#) page for Installation instructions.

SOURCE CODE

Active development of the Arduino software is [hosted by GitHub](#). See the instructions for [building the code](#). Latest release source code archives are available [here](#). The archives are PGP-signed so they can be verified using [this](#) gpg key.

DOWNLOAD OPTIONS

Windows Win 7 and newer
Windows ZIP file

Windows app Win 8.1 or 10 

Linux 32 bits
Linux 64 bits
Linux ARM 32 bits
Linux ARM 64 bits

Mac OS X 10.10 or newer

Release Notes
Checksums (sha512)

Previous Releases

Download the previous version of the current release, the classic 1.0.x, or old beta releases.

DOWNLOAD OPTIONS

[Previous Release 1.8.18](#)
[Arduino 1.0.x](#)
[Arduino 1.5.x beta](#)

Support the Arduino IDE

Since the release 1.x release in March 2015, the Arduino IDE has been downloaded **64,015,459** times — impressive! Help its development with a donation.

\$3

\$5

\$10


\$25

\$50

Other

JUST DOWNLOAD

CONTRIBUTE & DOWNLOAD

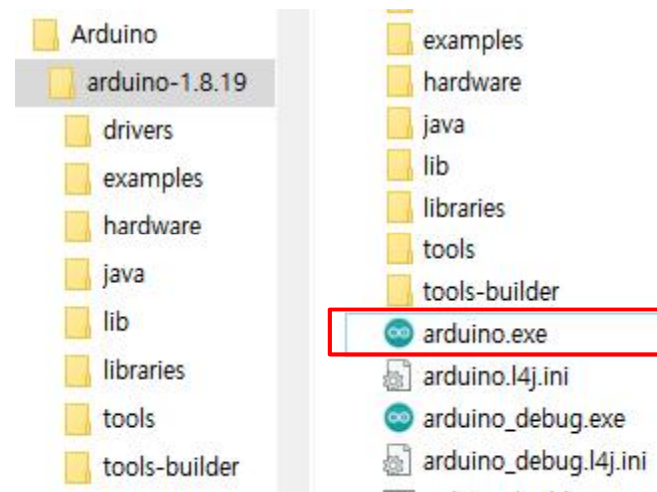


Learn more about [donating to Arduino](#).

통합 개발환경 다운로드 및 설치 (계속)

◆ Arduino 설치 및 실행

- 압축 해제하여 Arduino 설치
 - ❖ 설치 위치 : 실습PC 내 C:/Work 또는 D:/Work 디렉토리 사용
- Arduino 실행
 - ❖ 설치 디렉토리의 arduino.exe 실행
 - ❖ 실행 후 반드시 종료 (필수)



◆ 문서/Arduino 디렉토리 생성 확인

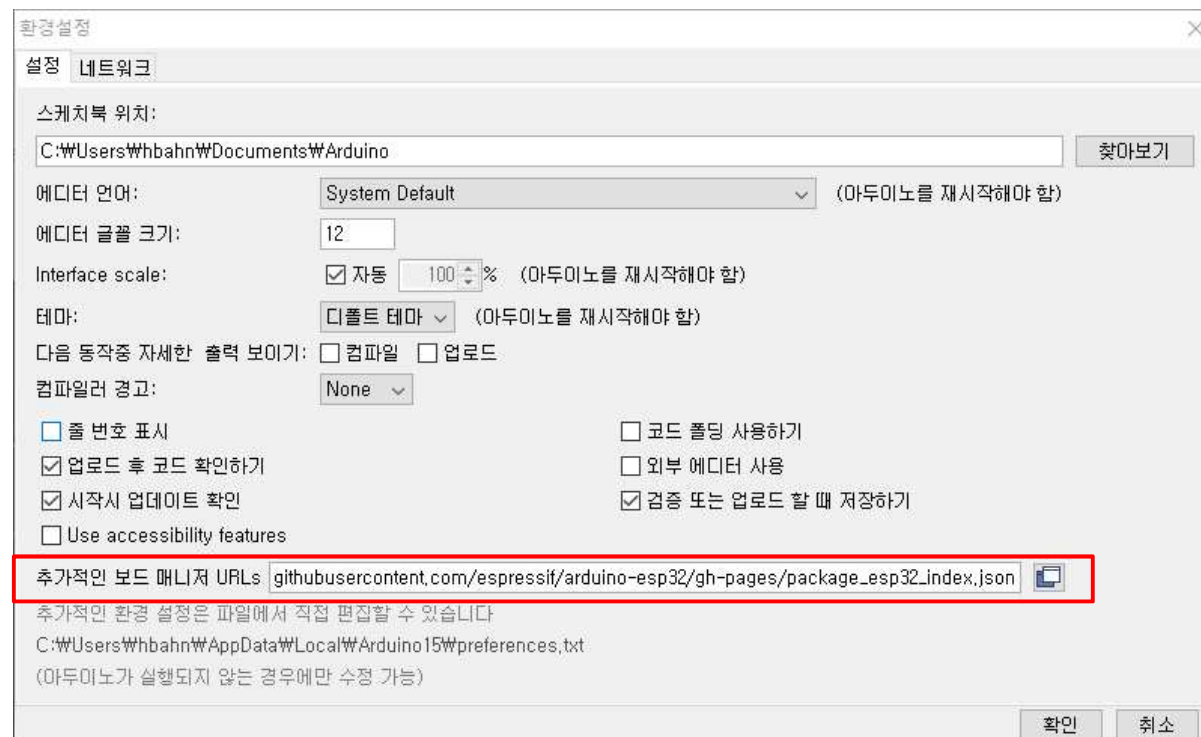
- 명령 실행 후 문서 디렉토리에 Arduino 디렉토리 생성 됨

하드웨어 패키지 설치

◆ Arduino IDE 환경에서 ESP32 SoC 사용에 필요한 추가적인 매니저 URL 설정

- 참조 : <https://docs.espressif.com/projects/arduino-esp32/en/latest/installing.html>
- 설치 경로 : Arduino IDE > 파일 > 환경설정 > 추가적인 보드 매니저 URLs
- 설정 : 위 경로의 Stable release link 복사

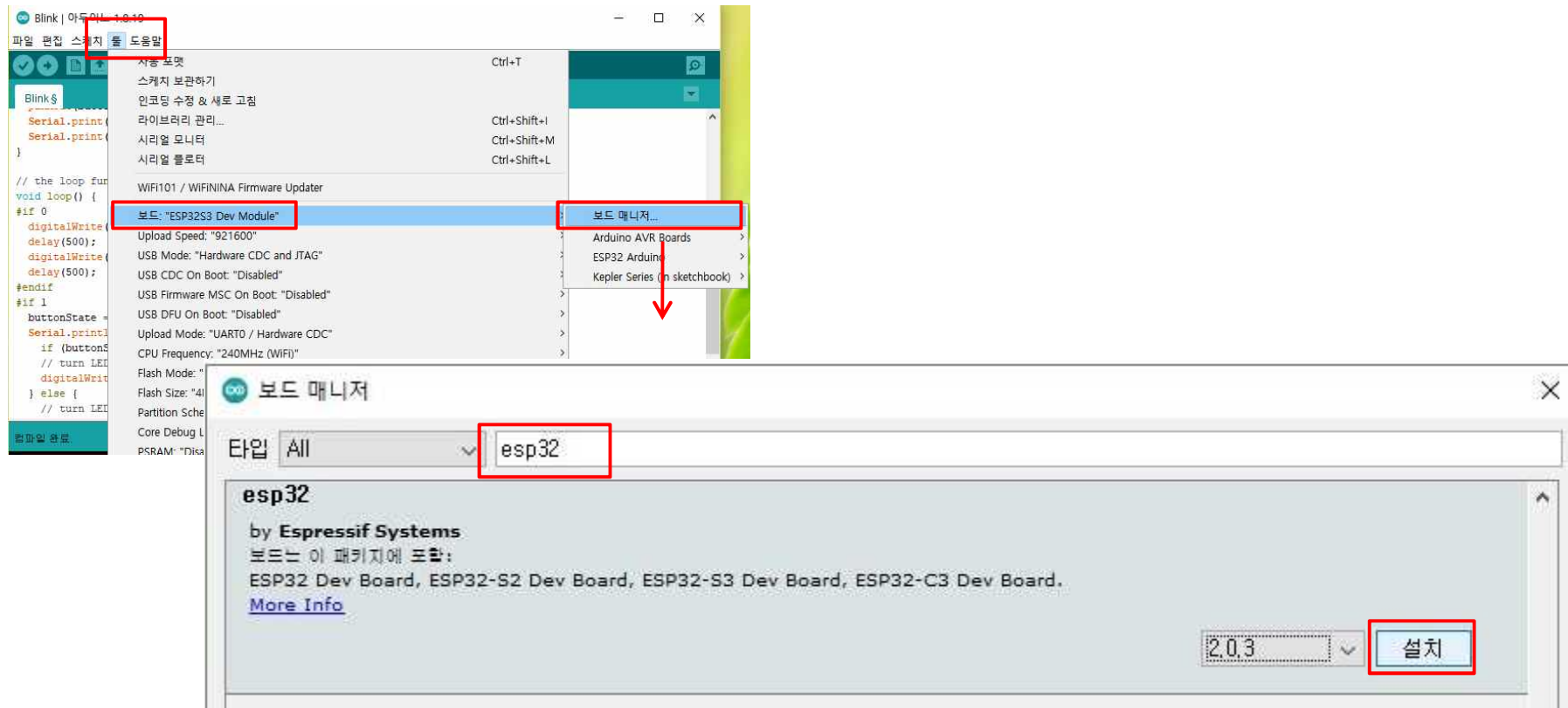
https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json



하드웨어 패키지 설치 (계속)

◆ Arduino IDE 환경에서 ESP32 SoC 사용에 필요한 보드 설정 추가

- 참조 : <https://docs.espressif.com/projects/arduino-esp32/en/latest/installing.html>
- 설치 경로 : Arduino IDE > 툴 > 보드 > 보드 매니저
- ESP32 보드를 지정한 후 설치



USB 드라이버 설치

◆ 드라이버 다운로드

- 아래 사이트 참조하여 드라이버 설치, 윈도우 자동 드라이버 설치 가능
- <https://www.silabs.com/developers/usb-to-uart-bridge-vcp-drivers?tab=downloads>

Software Downloads

Software (11)

Software • 11

CP210x Universal Windows Driver	v11.2.0 10/21/2022
CP210x VCP Mac OSX Driver	v6.0.2 10/27/2021
CP210x VCP Windows	v6.7 9/4/2020
CP210x Windows Drivers	v6.7.6 9/4/2020
CP210x Windows Drivers with Serial Enumerator	v6.7.6 9/4/2020

Show 6 more Software

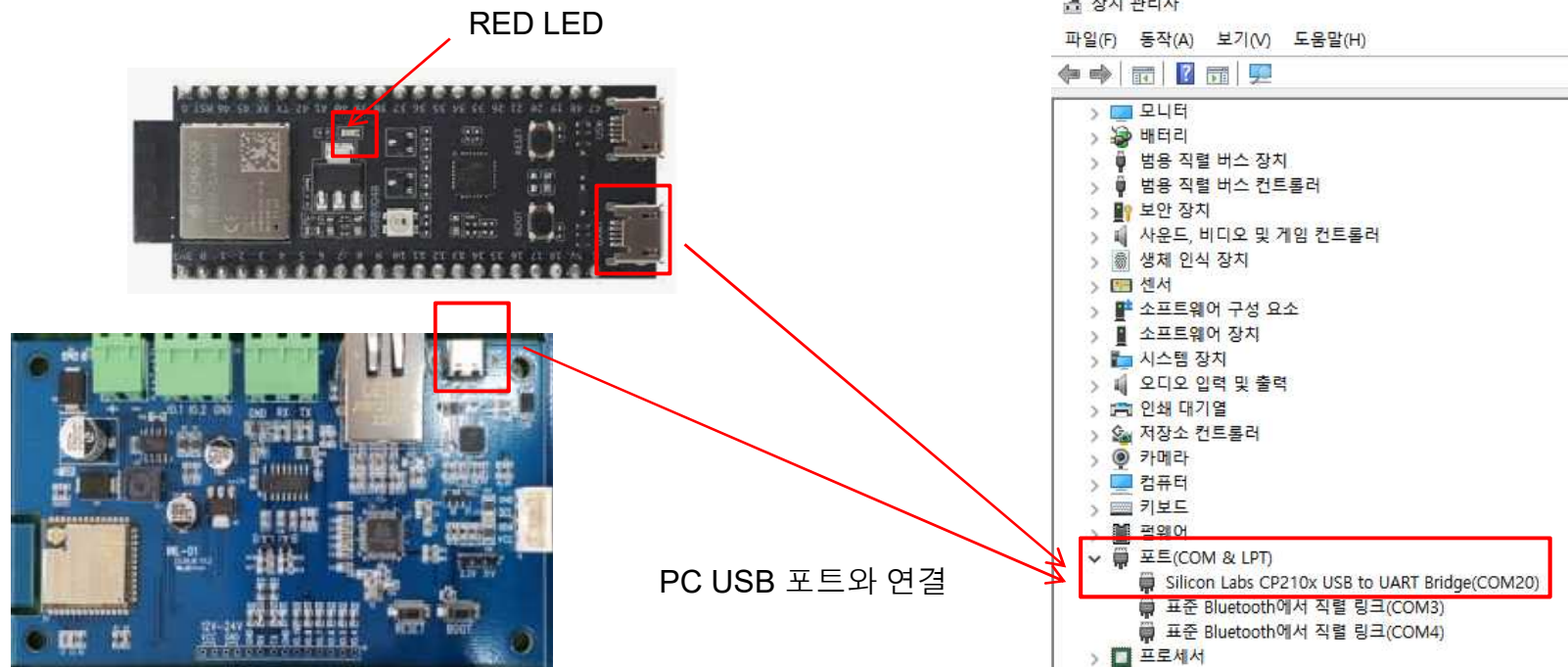
하드웨어 연결 및 동작 확인

◆ 타깃 보드와 PC 연결

- Micro USB Cable을 ESP32-S3-DevKitC-1 보드의 USB(UART로 표기된 부분)와 PC USB 포트 연결
- 하드웨어 연결이 완료되면 중앙 우측 RED LED ON

◆ 장치관리자에서 새로운 USB 장치가 검색되었는지 확인

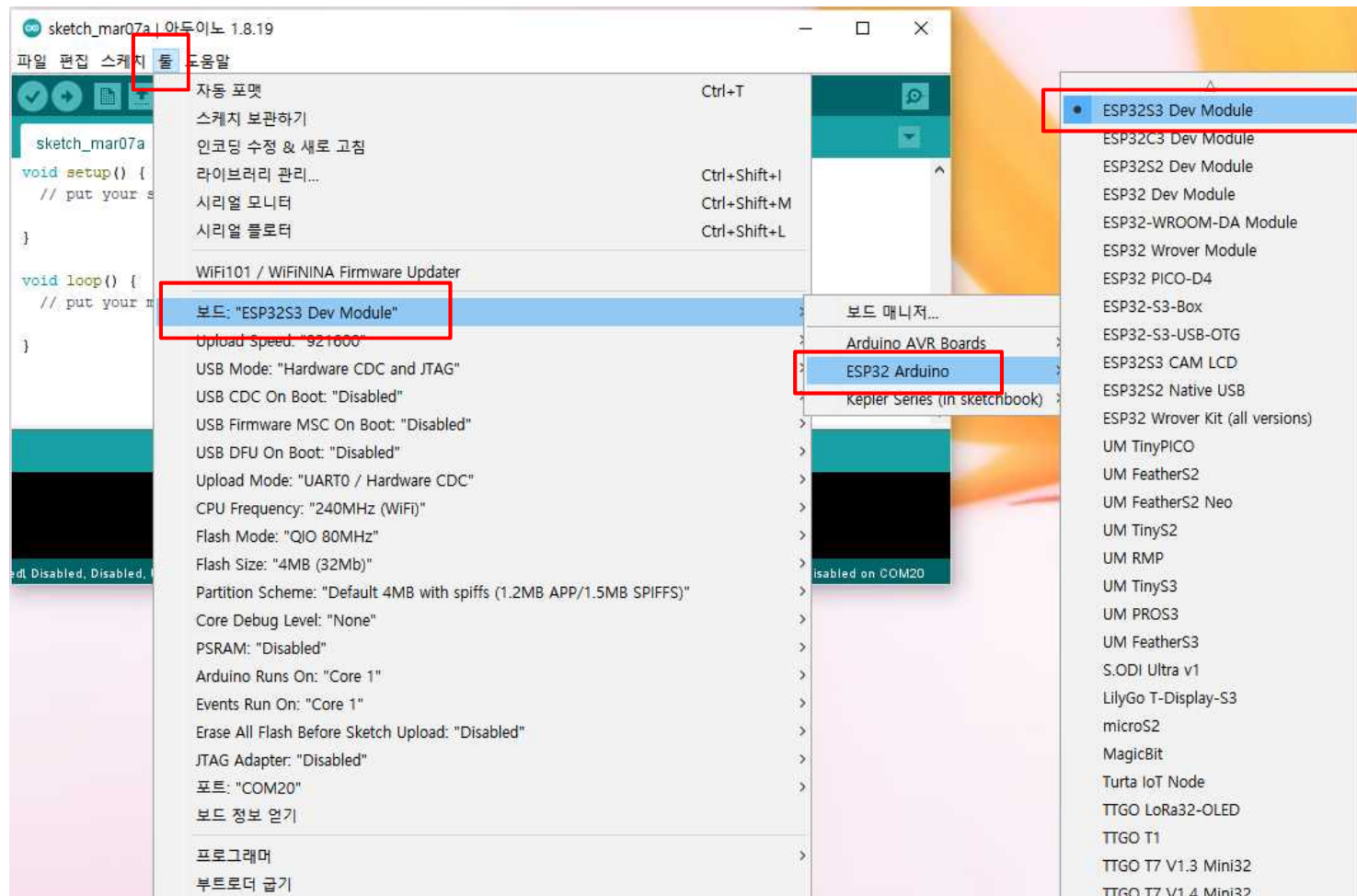
- 아래 그림은 드라이버 설치 된 후임. 드라이버 설치 전에는 비 정상



IDE 환경 설정 및 업로드

◆ Arduino를 실행하고 보드 선택

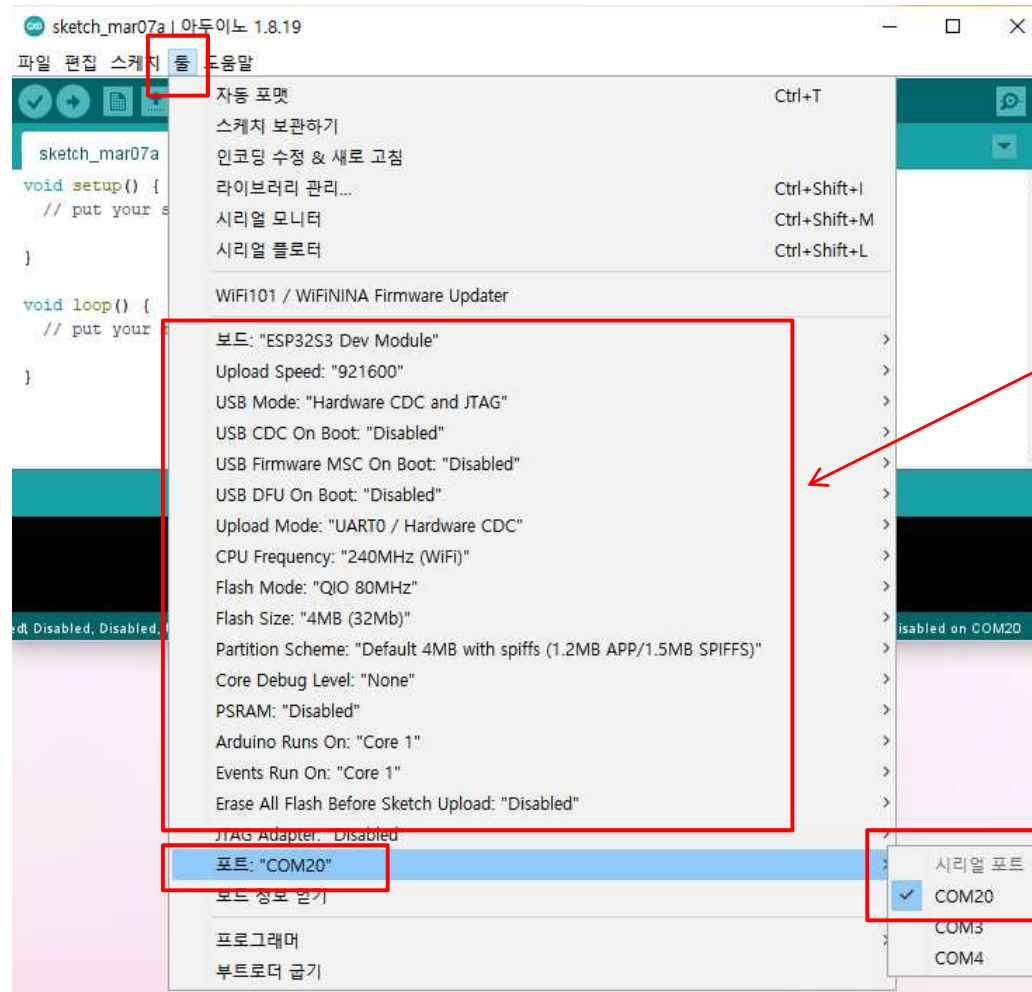
보드 선택 : 툴 > 보드 > **ESP32S3 Dev Module**



IDE 환경 설정 및 업로드 (계속)

◆ 시리얼 디버그 포트 설정

포트 선택 : 툴 > 포트 > **COMx**



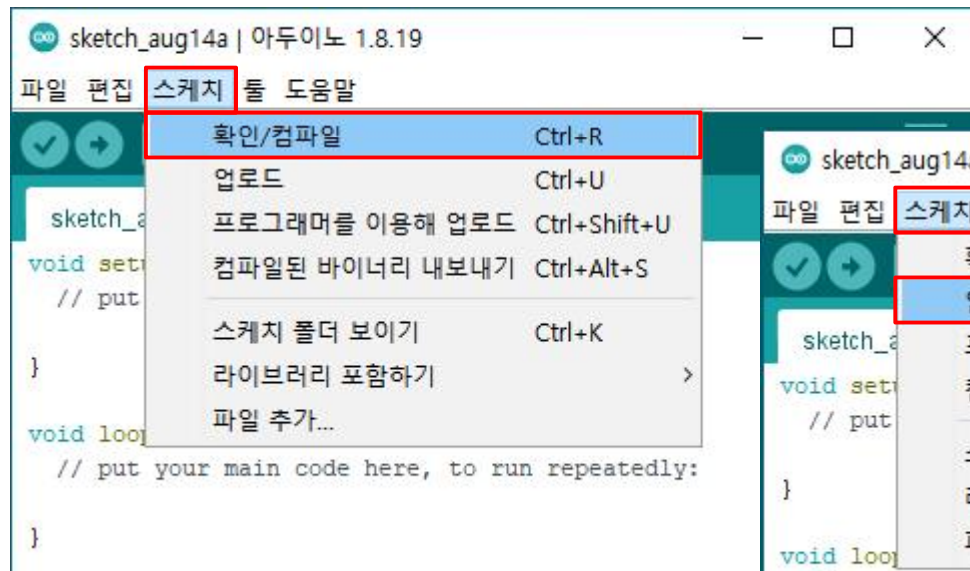
보드와 시리얼포트 설정이 완료되면
보드에 대한 정보 확인 가능

※ 주의 : 포트 확인이 안 되는 경우 드라이버 설치
및 USB에 정확하게 연결되어 있는지 확인 할 것.

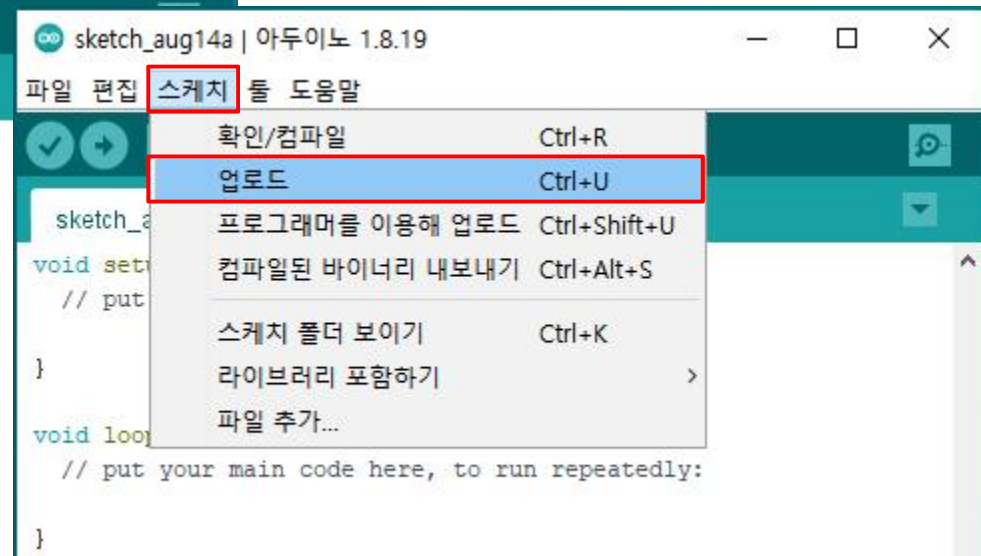
IDE 환경 설정 및 업로드 (계속)

- ◆ 소프트웨어 개발 후 컴파일
- ◆ 컴파일 후 타겟 보드에 업로드 및 실행

컴파일 : 스케치 > 확인 및 컴파일 (Ctrl+R)



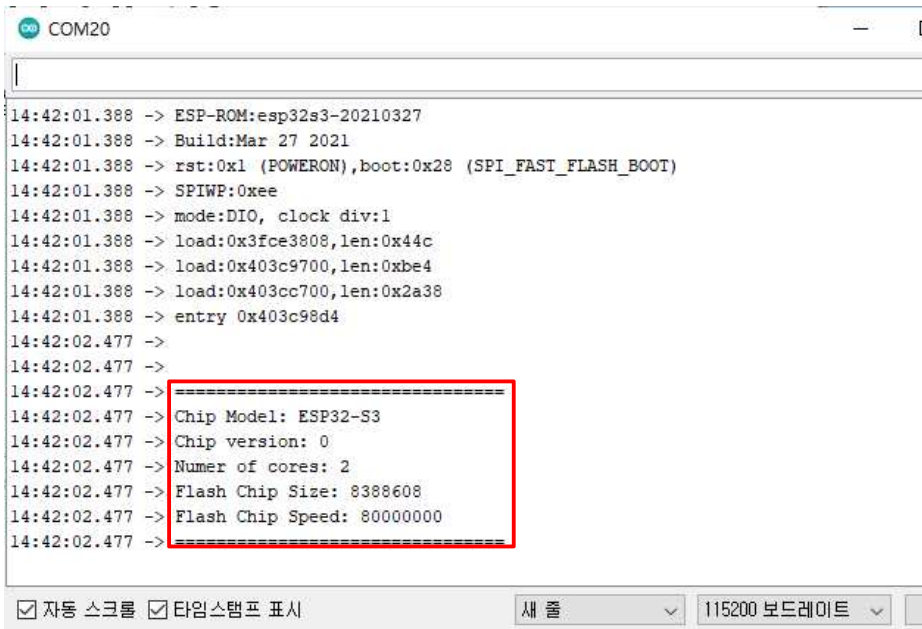
업로드 : 스케치 > 업로드 (Ctrl+U)



※ 참고 : 업로드 하면 자동으로 컴파일도 실행함

[실습] Hello Arduino on ESP32

◆ 컴파일 후 타겟 보드에 업로드 및 실행



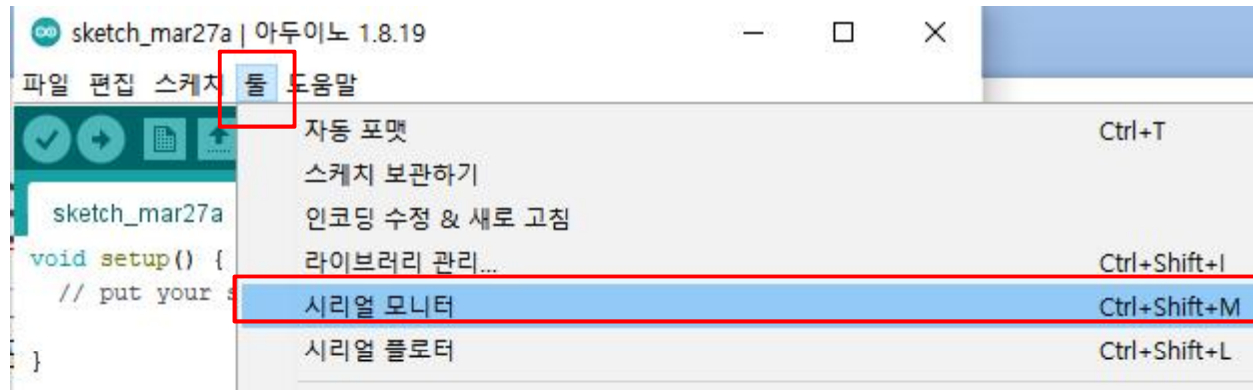
```
COM20
14:42:01.388 -> ESP-ROM:esp32s3-20210327
14:42:01.388 -> Build:Mar 27 2021
14:42:01.388 -> rst:0x1 (POWERON),boot:0x28 (SPI_FAST_FLASH_BOOT)
14:42:01.388 -> SPIWP:0xee
14:42:01.388 -> mode:DIO, clock div:1
14:42:01.388 -> load:0x3fce3808,len:0x44c
14:42:01.388 -> load:0x403c9700,len:0x4be4
14:42:01.388 -> load:0x403cc700,len:0x2a38
14:42:01.388 -> entry 0x403c98d4
14:42:02.477 ->
14:42:02.477 ->
14:42:02.477 -> =====
14:42:02.477 -> Chip Model: ESP32-S3
14:42:02.477 -> Chip version: 0
14:42:02.477 -> Numer of cores: 2
14:42:02.477 -> Flash Chip Size: 8388608
14:42:02.477 -> Flash Chip Speed: 80000000
14:42:02.477 -> =====
[ ] 자동 스크롤 [x] 타임스탬프 표시 새 줄 115200 보드레이트
```

```
void setup() {
  delay(500);
  Serial.begin(115200);
  delay(500);
  Serial.println("\n\n=====");
  Serial.print("Chip Model: ");
  Serial.println(ESP.getChipModel());
  Serial.print("Chip version: ");
  Serial.println(ESP.getChipRevision());
  Serial.print("Numer of cores: ");
  Serial.println(ESP.getChipCores());
  Serial.print("Flash Chip Size: ");
  Serial.println(ESP.getFlashChipSize());
  Serial.print("Flash Chip Speed: ");
  Serial.println(ESP.getFlashChipSpeed());
  Serial.println("=====");
}

void loop() {
  // put your main code here, to run repeatedly:
}
```

Arduino Serial.print (1)

- ◆ 시리얼(Serial) 포트로 다양한 형식의 데이터를 ASCII 문자로 출력
 - Serial 포트 출력은 Arduino IDE의 시리얼 모니터 출력으로 확인 가능



- ◆ 함수 Serial.print() `size_t Serial.print(val);`
`size_t Serial.print(val, format);`
- ◆ 함수 Serial.println() `size_t Serial.println(val);`
`size_t Serial.println(val, format);`
- val : 출력할 값
- format : 출력 형식(BIN,OCT,DEC,HEX) 및 소수점 자릿수
- 리턴 값 : 출력한 바이트 수
- Serial.println은 줄바꿈 포함
- 참고 : Serial.print() 사용 전에 Serial.begin() 함수를 이용하여 설정 필수

Arduino Serial.print (2)

◆ 함수 Serial.begin()

```
void Serial.begin(speed);  
void Serial.begin (speed, config);
```

- speed : long 형식의 bps (전송 속도)
- config : data, parity, stop 비트 설정
 - ❖ 예) SERIAL_8N1

◆ 사용 예

```
void setup() {  
  Serial.begin(115200);
```

```
  Serial.println(56);           // "56"  
  Serial.println(56, BIN);      // "111000"  
  Serial.println(56, OCT);      // "70"  
  Serial.println(56, DEC);      // "56"  
  Serial.println(56, HEX);      // "38"  
  Serial.println(56.7860, 0);    // "57" *반올림되어 출력됨  
  Serial.println(56.7860, 4);    // "56.7860"  
}
```

```
void setup() {  
  Serial.begin(115200);  
  
  Serial.println(56);           // "56"  
  Serial.println(56.7865);      // "56.79" *반올림되어 출력됨  
  Serial.println("H");          // "H"  
  Serial.println("Hello!!");    // "Hello!!"  
}
```

ESP32 하드웨어 패키지 지원 내용

C:\Users\[사용자]\AppData\Local\Arduino15\packages\esp32\hardware\esp32\2.0.7\cores

Peripheral	ESP32	ESP32-S2	ESP32-C3	ESP32-S3
ADC	Yes	Yes	Yes	Yes
Bluetooth	Yes	Not Supported	Not Supported	Not Supported
BLE	Yes	Not Supported	Yes	Yes
DAC	Yes	Yes	Not Supported	Not Supported
Ethernet	Yes	Not Supported	Not Supported	Not Supported
GPIO	Yes	Yes	Yes	Yes
Hall Sensor	Yes	Not Supported	Not Supported	Not Supported
I2C	Yes	Yes	Yes	Yes
I2S	Yes	Yes	Yes	Yes
LEDC	Yes	Yes	Yes	Yes
Motor PWM	No	Not Supported	Not Supported	Not Supported
Pulse Counter	No	No	No	No
RMT	Yes	Yes	Yes	Yes
SDIO	No	No	No	No
SDMMC	Yes	Not Supported	Not Supported	Yes
Timer	Yes	Yes	Yes	Yes
Temp. Sensor	Not Supported	Yes	Yes	Yes
Touch	Yes	Yes	Not Supported	Yes
TWAI	No	No	No	No
UART	Yes	Yes	Yes	Yes
USB	Not Supported	Yes	Yes	Yes
Wi-Fi	Yes	Yes	Yes	Yes

ESP32-S3 Pins for Arduino

C:\Users\[사용자]\AppData\Local\Arduino15\packages\esp32\hardware\esp32\2.0.7\variants\esp32s3

```
#ifndef Pins_Arduino_h
#define Pins_Arduino_h

#include <stdint.h>
#include "soc/soc_caps.h"

#define USB_VID 0x303a
#define USB_PID 0x1001

#define EXTERNAL_NUM_INTERRUPTS          46
#define NUM_DIGITAL_PINS                 48
#define NUM_ANALOG_INPUTS                20

// Some boards have too low voltage on this pin (board design bug)
// Use different pin with 3V and connect with 48
// and change this setup for the chosen pin (for example 38)
static const uint8_t LED_BUILTIN = SOC_GPIO_PIN_COUNT+48;
#define BUILTIN_LED              LED_BUILTIN // backward compatibility
#define LED_BUILTIN              LED_BUILTIN
#define RGB_BUILTIN              LED_BUILTIN
#define RGB_BRIGHTNESS          64

#define analogInputToDigitalPin(p)      (((p)<20)?(analogChannelToDigitalPin(p)):-1)
#define digitalPinToInterrupt(p)        (((p)<48)?(p):-1)
#define digitalPinHasPWM(p)             (p < 46)
```

```
static const uint8_t TX = 43;
static const uint8_t RX = 44;

static const uint8_t SDA = 8;
static const uint8_t SCL = 9;

static const uint8_t SS  = 10;
static const uint8_t MOSI = 11;
static const uint8_t MISO = 13;
static const uint8_t SCK = 12;

static const uint8_t A0 = 1;
static const uint8_t A1 = 2;
static const uint8_t A2 = 3;
static const uint8_t A3 = 4;
static const uint8_t A4 = 5;
static const uint8_t A5 = 6;
static const uint8_t A6 = 7;
static const uint8_t A7 = 8;
static const uint8_t A8 = 9;
static const uint8_t A9 = 10;
static const uint8_t A10 = 11;
static const uint8_t A11 = 12;
static const uint8_t A12 = 13;
```

```
static const uint8_t A13 = 14;
static const uint8_t A15 = 16;
static const uint8_t A16 = 17;
static const uint8_t A17 = 18;
static const uint8_t A18 = 19;
static const uint8_t A19 = 20;

static const uint8_t T1 = 1;
static const uint8_t T2 = 2;
static const uint8_t T3 = 3;
static const uint8_t T4 = 4;
static const uint8_t T5 = 5;
static const uint8_t T6 = 6;
static const uint8_t T7 = 7;
static const uint8_t T8 = 8;
static const uint8_t T9 = 9;
static const uint8_t T10 = 10;
static const uint8_t T11 = 11;
static const uint8_t T12 = 12;
static const uint8_t T13 = 13;
static const uint8_t T14 = 14;

#endif /* Pins_Arduino_h */
```

질의 응답