Technical Document

ELICIT-R1 USER GUIDE ver.1.00

RF Module for Industrial IoT

Feature List

1. Operation Condition

- Supply voltage: 1.8 ~ 3.8V
- Operating current (Tx): 87mA @ +20dBm
- Operating current (Rx): 8.2mA
- Deep-Sleep current: < 1μA
- Operating temperature: -40℃ to +85℃

2. Interface

- 2x UART
- 8x GPIO
- 2x Button Dedicated Pin
- 2x Status LED Dedicated Pin
- 1x I2C
- 1x 12 bit ADC, 1x 12bit DAC

3. General Specification

- Form factor: Through-hole(1.27mm), surface-mount
- Size: 20x27x3(mm)
- Antenna Connector: U.FL

4.Wireless Characteristics

- Operating frequency: KOR USN 940.1 ~ 944.3MHz
- Operating RF channel: 19
- Maximum Tx power: +20dBm
- LBT(CSMA/CA)
- 2 PHY mode

High Data rate (HDR) mode	
Modulation	MSK
RF Bit rate	80kbps
Channel coding	N/A
Rx sensitivity	-105dBm
Long Range (LR) mode	
Modulation	2FSK
RF Bit rate	20kbps
Channel coding	Convolutional coding(K=7)
Rx sensitivity	-110dBm

5. Certification

KC (R-R-ELQ-ELICIT-R1)

Revision History

Author	Description of Changes	Date
ELIOT	Initial Draft	Mar. 2022
ELIOT	Button, Event 설명 추가, 양산 제품으로 사진 변경, 인증 정보 추가	Sep. 2022
ELIOT	Command 변경, Linked event 추가	Mar. 2023

Disclaimer

엘리엇시스템 주식회사
https://www.eliotsystem.com/

Contents

1 Technical Specification

1.1 General Operating Conditions

Parameter	Min	Typ	Max	Unit
Operating temperature	-40	-	85	℃
Supply voltage (VCC)	1.8	3.3	3.8	V
Operating current (Tx), +20dBm[#1,2]	85	87	90	mA
Operating current (Tx), +14dBm[#1,2]	40	45	50	mA
Operating current (Tx), +10dBm[#1,2]	33	35	40	mA
Operating current (Rx), Listen[#1,2]	7.1	8.2	9.1	mA
Stand-by current (Sleep) [#1,2]	5	-	6	μA
Stand-by current (Deep-Sleep) [#1,2]	0.5	-	1	μA
Voltage on IO(GPIO, AUX, I2C,UART)	-0.3	VCC	VCC+0.3	V
Input Low voltage of GPIO	-	-	0.3*VCC	V
Input High voltage of GPIO	0.7*VCC	-	-	V
Input voltage of ADC	-0.3		VCC	V

#1 VCC = 3.3V

#2 PHY mode = High data rate mode (MSK 80kbps)

1.2 Absolute Maximum Ratings

Parameter	Min	Typ	Max	Unit
Storage temperature	-50	-	+150	℃
Voltage on supply pin (VCC)	-0.3	-	3.8	V
Voltage on GPIO pin	-0.3	-	VCC+0.3	V
Voltage on RESETn pin	-0.3	-	3.8	V
Absolute voltage on RF connector	-0.3	-	1.2	V
Current per GPIO pin (Sink)	-	-	50	mA
Current per GPIO pin (Source)	-	-	50	mA
Current for all GPIO pins (Sink)	-	-	200	mA
Current for all GPIO pins (Source)	-	-	200	mA

1.3 RF Characteristics

Parameter	Min	Typ	Max	Unit
RF Frequency[#1]	940.1	-	944.3	MHz
Maximum Tx Power	17.7	20	21.4	dBm
Minimum Tx Power	-0.8	0	0.2	dBm
Adjustable RF Power Step	-	0.1	-	dBm
Spurious emissions1 (718~915MHz)[#2]			-78	dBm
Spurious emissions2 (949.3~962MHz)[#2]			-85	dBm
Spurious emissions3 (over 1GHz)[#2]			-41	dBm

#1 간섭회피 또는 간섭경감기술로 940.1~944.3MHz 주파수 대역의 전파를 사용하는 USN 용 무선설비 기준

#2 RF CH 2 ~ 20, TX Power 10dBm

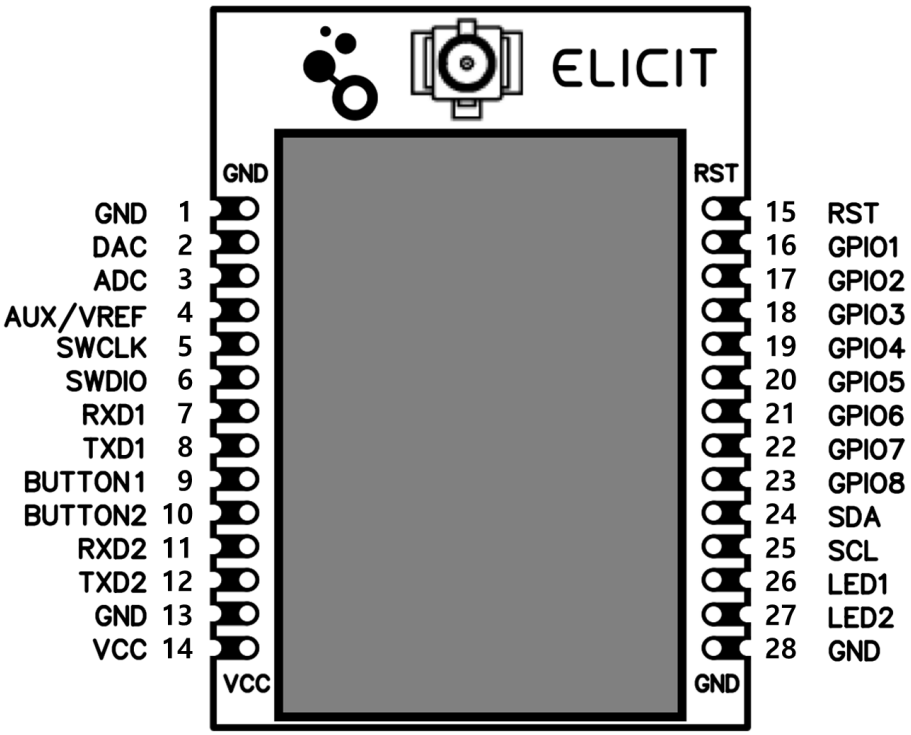
1.4 PHY Mode 0 – Long range mode

Parameter		Unit
Modulation	2FSK	-
Deviation	10	kHz
RF bit rate	20	kbps
Symbol (Data) rate	10	kbps
Channel spacing	200	kHz
Forward error correction (FEC)	Convolutional Code (Constraint k = 7)	-
DSSS (Direct sequence spread spectrum)	NA	-
Rx sensitivity	-110	dBm

1.5 PHY Mode 1 – High data rate mode

Parameter		Unit
Modulation	MSK	-
Deviation	20	kHz
RF bit rate	80	kbps
Symbol (Data) rate	80	kbps
Channel spacing	200	kHz
Channel coding	N/A	-
DSSS (Direct sequence spread spectrum)	N/A	-
Rx sensitivity	-105	dBm

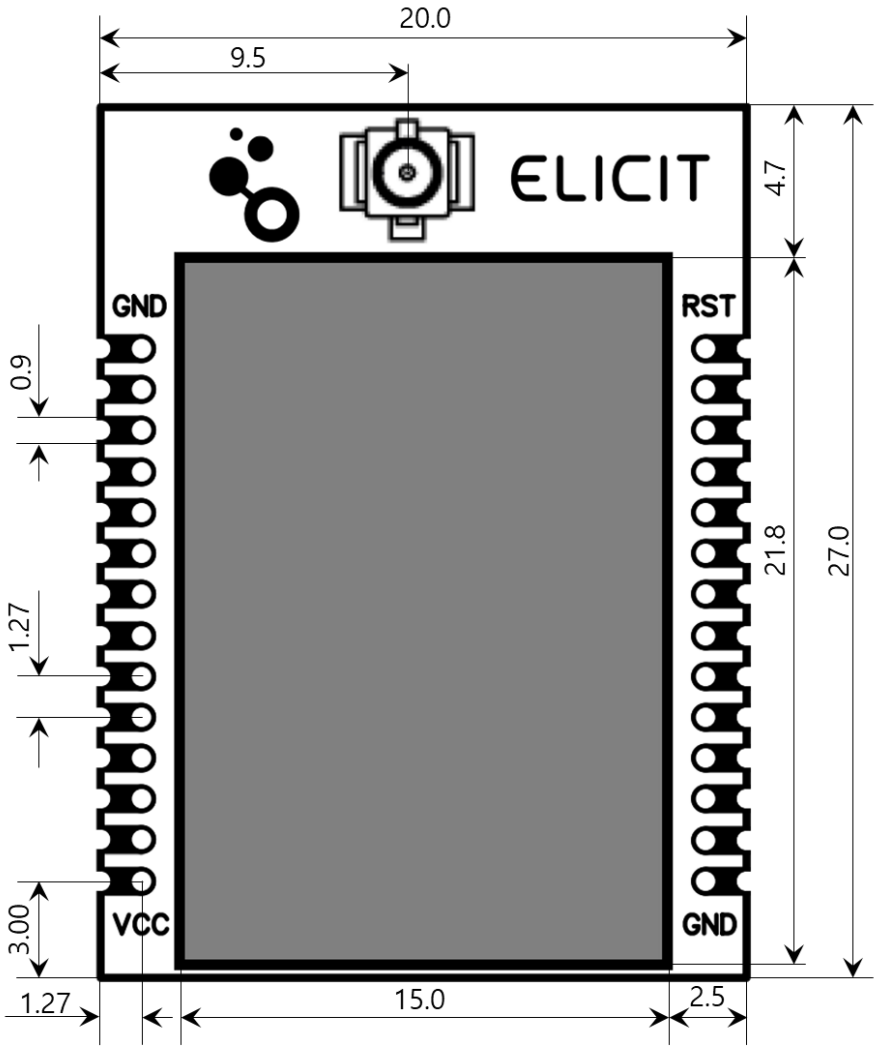
1.6 HW Pin Description

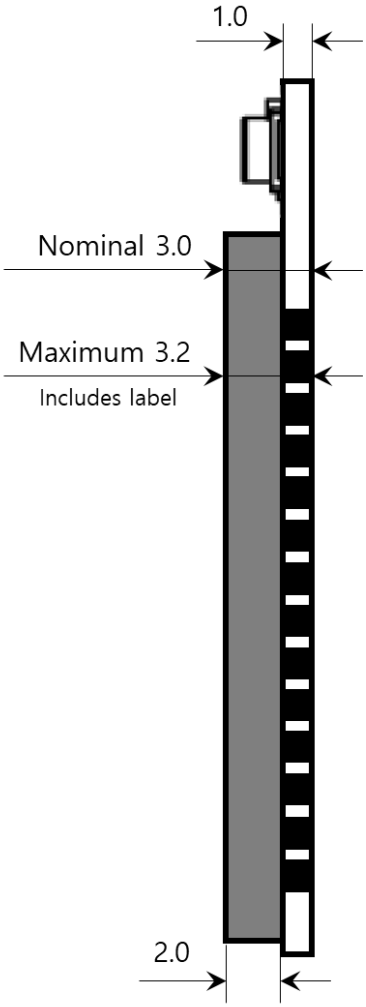


PIN NO.	Name	Description
1	GND	Ground
2	DAC	Digital Analog Converter Output
3	ADC	Analog Digital Converter Input
4	AUX/VREF	Auxiliary Input, could wake up from normal sleep
5	SWCLK	SWD CLOCK
6	SWDIO	SWD DATA
7	RXD1	UART1 RX, for host interface (AT command)
8	TXD1	UART1 TX, for host interface (AT command)
9	BUTTON1	External Button Input1, wakes up from Deep/normal sleep
10	BUTTON2	External Button Input2, wakes up from normal sleep
11	RXD2	UART2 RX, for user device
12	TXD2	UART2 TX, for user device
13	GND	Ground
14	VCC	Power supply (1.8 ~ 3.8V). default 3.3V
15	RST	Reset input, active low, internal pulled up.
16	GPIO1	GPIO (DI/DO, wakes up from Deep-sleep)
17	GPIO2	GPIO (DI/DO)
18	GPIO3	GPIO (DI/DO, wakes up from Deep-sleep)
19	GPIO4	GPIO (DI/DO, DI linked event)
20	GPIO5	GPIO (DI/DO, DI linked event)
21	GPIO6	GPIO (DI/DO, DI linked event)
22	GPIO7	GPIO (DI/DO, DI linked event, wakes up from normal sleep)

23	GPIO8	GPIO (DI/DO, DI linked event, wakes up from normal sleep)
24	SDA	I2C DATA
25	SCL	I2C CLOCK
26	LED1	Status LED1(Push-pull/Open drain output)
27	LED2	Status LED2(Push-pull/Open drain output)
28	GND	Ground

1.7 ELICIT R1 Dimension





단위: mm

PCB 두께: 1mm

Shield can 높이: 2mm

Through hall diameter: 0.6mm

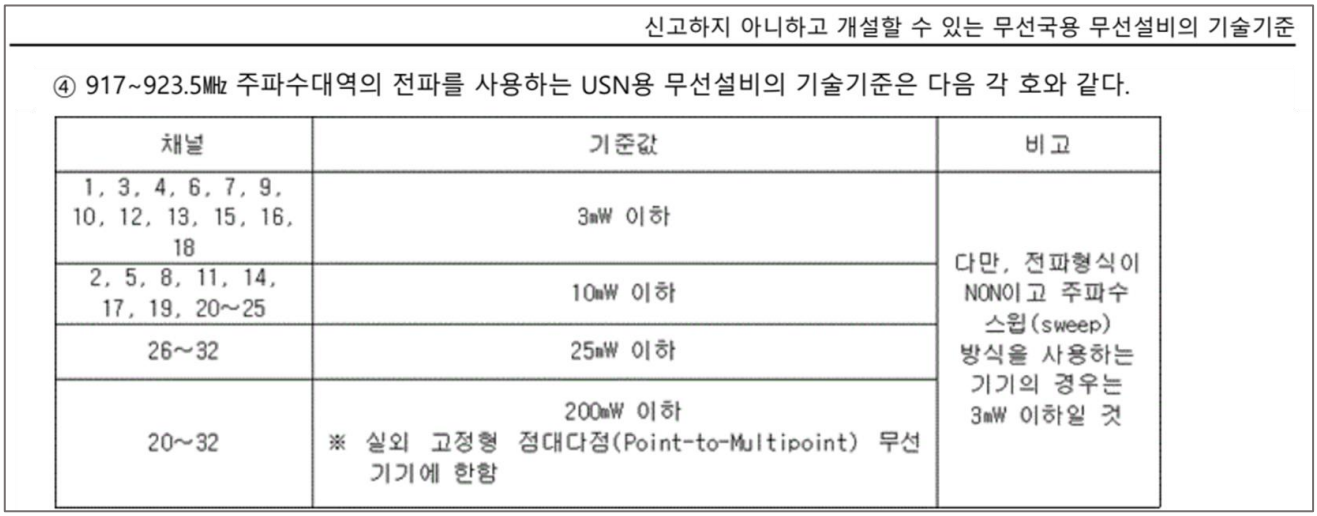
2 Elicit

Elicit 은 엘리엇시스템 주식회사(ELIOT SYSTEM Inc.)에서 자체 개발한 IoT 를 위한 Proprietary wireless network protocol 이자, 무선 모뎀 제품명이다. Elicit protocol 은 저전력으로 동작하며, 보안과 안정성, 확장성이 높은 무선 네트워크를 위해 개발되었으며, 무선 네트워크에 대한 전문적인 지식을 습득하지 않아도 현장 엔지니어들이 빠르게 무선 네트워크를 구축하고 운영할 수 있도록 사용자 인터페이스를 제공한다.

Elicit	무선 IoT 네트워크를 위한 Proprietary Protocol Stack
ELICIT-R1	Elicit protocol 이 탑재된 940MHz 대역의 무선 모뎀

2.1 IoT 와 ELICIT-R1

ELICIT-R1 은 산업용 사물인터넷(Industrial internet of things, IIoT) 무선 네트워크를 위해 설계, 개발되었다. 다양한 산업용 설비, 장비와 쉽게 연결할 수 있도록 8 개의 GPIO, 2 개의 UART(MODBUS 지원), I2C, 12bit ADC/DAC 를 가지고 있으며, Input 상태와 Timer 이벤트 등과 연동하여 사용자가 임의의 동작을 할 수 있도록 설정 가능하여 사용자가 원하는 IoT 서비스를 별도의 프로그래밍 없이 빠르게 제공할 수 있다. 사용하는 주파수 대역은 국내에 신규로 할당된 940.1 ~ 944.3MHz 대역이다.



신고하지 아니하고 개설할 수 있는 무선국용 무선설비의 기술기준

④ 917~923.5㎒ 주파수대역의 전파를 사용하는 USN용 무선설비의 기술기준은 다음 각 호와 같다.

채널	기준값	비고
1, 3, 4, 6, 7, 9, 10, 12, 13, 15, 16, 18	3㎽ 이하	다만, 전파형식이 NON이고 주파수 스윕(sweep) 방식을 사용하는 기기의 경우는 3㎽ 이하일 것
2, 5, 8, 11, 14, 17, 19, 20~25	10㎽ 이하	
26~32	25㎽ 이하	
20~32	200㎽ 이하 ※ 실외 고정형 점대다점(Point-to-Multipoint) 무선기기에 한함	

Figure 1 917 ~ 923.5MHz 대역의 기술기준(과학기술정보통신부)

LoRa, Wi-SUN 등의 기존 LPWA Sub-Giga 기술들과 RF ID 가 함께 사용하는 917MHz 대역은 산업계와 민간 영역에서 IoT 수요가 지속적으로 늘어나면서 가용 채널의 부족과 디바이스의 출력 제한[1] (25mW) 등의 한계로 인해 정부에서는 940.1 ~ 944.3MHz 대역에서 최대 200mW 까지 출력 가능한 21 개 채널을 USN 용도로 신규 지정하였다(Figure 2 940MHz 대역의 기술 기준 - 출력기준(과학기술정보통신부)).

ELICIT-R1 의 최대 출력은 100mW(20dBm)으로, 기존 917MHz 대역에서 디바이스의 출력 제한인 25mW 대비 4 배 높은 출력으로 보다 넓은 통신 범위를 가진다. 그리고 데이터 Hopping 기능을 통해 무선 음영지역 없이 무선 IoT 네트워크를 구성할 수 있다.

[1] 917MHz 대역에서 허용하는 200mW 출력은 실외 고정형 점대다점 기기로, 기지국 장비등이 이에 해당하며, 일반 무선 디바이스는 최대 25mW 로 제한된다.

신고하지 아니하고 개설할 수 있는 무선국용 무선설비의 기술기준

⑥ 간섭회피 또는 간섭경감기술로 940.1~944.3㎒ 주파수대역의 전파를 사용하는 USN용 무선설비의 기술기준은 다음 각 호와 같다.

4. 안테나절대이득을 포함한 복사전력은 200㎽ 이하일 것

Figure 2 940MHz 대역의 기술 기준 - 출력기준(과학기술정보통신부)

ELICIT-R1 은 저전력 센서 네트워크 구성을 위해 2 가지의 Sleep mode 를 지원한다. Deep sleep 모드에서는 소비전류가 1μA 이하로 떨어져 배터리로 운영하는 IoT 서비스를 구현하는데 적합하다.

산업용 IoT 네트워크를 위한 ELICIT-R1

- USN 용도로 신규 지정된 940MHz 대역 사용, 혼잡하지 않은 21 개의 무선 채널 운영
- 최대 출력 100mW(20dBm), 보다 넓은 통신 범위 제공
- 독자 개발한 IIoT 를 위한 무선 네트워크 스택(Elicit)을 적용
- 다양한 IO Interface 제공, 입력 이벤트, Timer 이벤트와 연동하여 사용자 지정 명령 실행 가능
- 초 저전력(Deep sleep mode: < 1μA)

2.2 Elicit Protocol Stack

Elicit protocol 의 Radio frame 구조는 아래 그림과 같다.

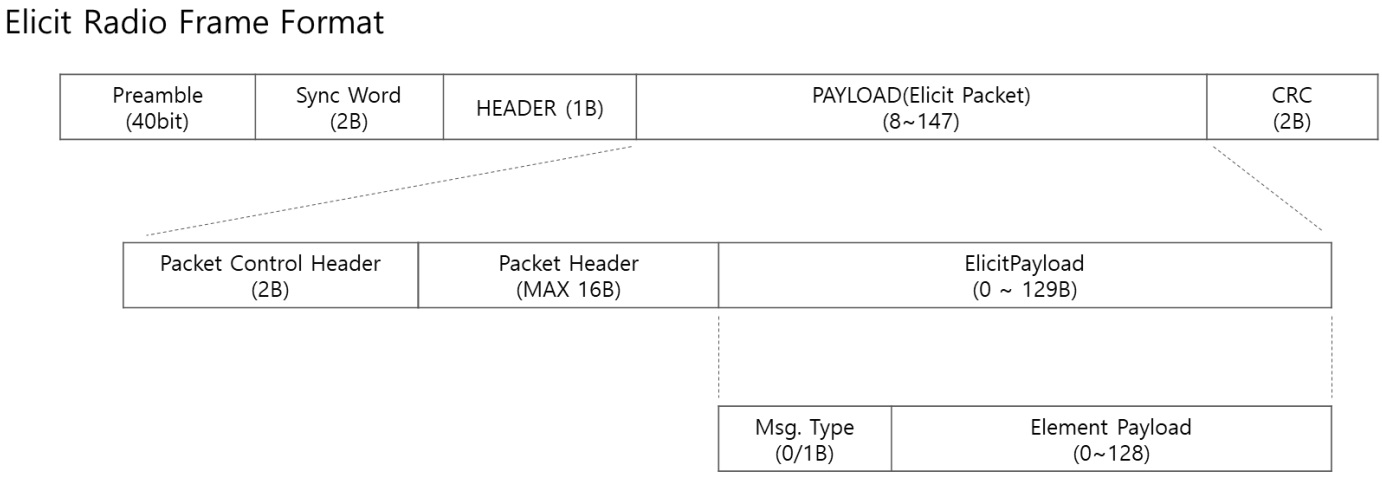


Figure 3 Elicit Radio frame format

Elicit 에서 무선 패킷(packet)은 40bit 의 preamble 과 2byte 의 Sync word 를 제외하고 최대 150byte 로 구성된다. HEADER, PAYLOAD, CRC 로 크게 구분되고, PAYLOAD 는 다시 Packet Control Header, Packet Header, Elicit Payload 로 구성된다. 그리고 PAYLOAD 는 보안을 위해 독자 개발한 Elicit Cipher 를 사용하여 매 패킷마다 서로 다르게 암호화한다. 사용자(User)의 데이터는 Element Payload 에 실리며, 한 번에 전송할 수 있는 최대 크기는 128byte 이다.

Elicit Packet 을 송신할 때 CSMA/CA[2]를 사용하여 다수의 디바이스들이 무선 채널을 원활히 공유하여 사용할 수 있도록 한다. 채널 확인은 Energy detection CCA(clear channel assessment)를 사용하며, Threshold 는 -75dBm 이다. 즉 송신 전 채널의 수신신호 세기(RSSI, Received signal strength indicator)가 -75dBm 이하일 때 송신하게 된다.

무선 전송의 신뢰도 향상을 위해 Elicit 에서는 상대방의 주소를 명시하여 전송할 때는 항상 ACK[3]를 요청하여 전송확인을 한다. 송신 후, 40ms 이내에 ACK 가 오지 않으면 최대 2 회 재송신을 하고, 최종 3 회 송신 후에도 ACK 를 수신하지 못하면 해당 패킷은 송신 실패로 처리한다. 다수의 디바이스에게 동시에 전송(broadcasting)하는 경우에는 ACK 를 요청하지 않는다.

신뢰성 높은 네트워크를 위한 Elicit 프로토콜

- 무선 패킷을 암호화하는 Elicit Cipher 는 동일 데이터를 전송해도 매 패킷마다 다르게 변조
- 사용자가 한 번에 전송할 수 있는 최대 데이터 크기는 128byte
- CSMA/CA 방식을 사용하여 전송하고, ACK 를 통해 수신 확인

2.2.1 Tree Topology

Elicit protocol 은 AP(Access point)를 중심으로 Node 들이 나무가지 형식으로 연결된 Tree 구조(Tree topology)를 기본으로 개발되었다. 계층적으로 상호 연결된 각 디바이스(AP 와 Node)들은 네트워크 내에서 서로 구분할 수 있는 주소(short address, SHORT ADDR)를 할당 받고, 이를 통해 정보를 교환한다.

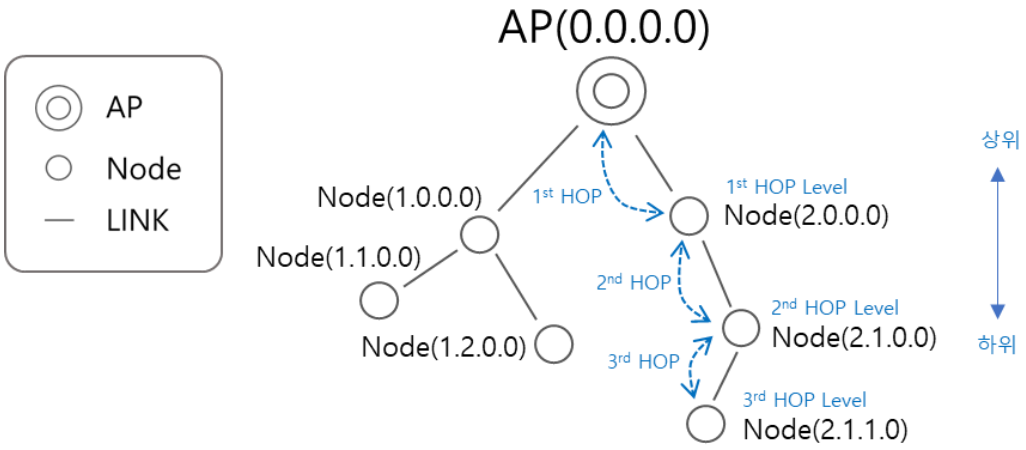


Figure 4 Elicit Tree Topology

최종 목적지까지 정보를 전송하는 경로는 연결된 가지를 통해서 이루어 진다. AP 는 모든 가지의 중심으로 네트워크 내의 모든 Node 와 통신이 가능하다. 위 [Figure 4 Elicit Tree Topology]에서 AP 는 직접 연결된 Node(1.0.0.0), Node(2.0.0.0)와 통신이 가능하고, Node(1.0.0.0)은 AP, Node(1.1.0.0), Node(1.2.0.0)과 통신이

[2] CSMA(Carrier Sense Multiple Access)/CA(Collison avoidance): 반송파 감지 다중 접속/충돌 회피 기술, 송신하기 전에 반송파(주파수 채널)를 누군가 사용 중인지 확인후 사용 중이면 임의의 시간 동안 기다린 후(Back-off)에 다시 확인하고, 사용 중이 아닐 때 송신하는 방법.

[3] ACK(Acknowledgement, 확인응답): 수신 측에서 에러 없이 정상적으로 수신했을 때 송신 측으로 응답하는 것.

가능하다. 직접 연결되지 않은 AP 와 Node(2.1.0.0) 사이의 통신은 중간의 Node(2.0.0.0)을 경유하여 이루어 진다. 이렇게 정보를 직접 전달하지 않고 다른 Node 를 경유하는 단위를 Hop 이라고 한다.

AP 는 Hop level 0 이며, AP 와 첫 번째 Hop 으로 연결된 Node 들을 1st Hop level 이라고 하고, 다시 1st Hop level 에서 연결된 두 번째 Hop 으로 연결된 Node 들을 2nd Hop level 이라고 한다. Elicit 에서는 최대 4th Hop level 까지 지원한다.

Tree 네트워크 상의 임의의 Node 를 기준으로 AP 쪽으로 연결된 Node 를 상위 Node 라고 하고, AP 에서 Hop 단계를 더 많이 거치는 Node 를 하위 노드라고 한다. 모든 Node 들의 최상위 Node 는 AP 이다. Tree 구조에서 AP 와 Node 는 자신을 중심으로 형성된 가지의 모든 하위 Node 들과 통신을 할 수 있으며, Node 는 자신의 상위 가지에 있는 Node 및 AP 와 통신할 수 있다.

Elicit 프로토콜 스택의 네트워크 구조

- 최대 4 단계의 Hop 을 지원하는 Tree 구조의 프로토콜
- AP 는 모든 Node 의 정보를 수집할 수 있음.
- Node 는 자신이 속한 가지의 모든 Node 와 연결 가능
- 다른 가지에 있는 Node 로의 전송은 동일 레벨까지 Hopping 한 후 전달
- AP 와 Node 의 하드웨어 차이는 없으며, 사용자 설정으로 결정

2.3 Elicit Network

AP 를 중심으로 구성된 단일 네트워크를 PAN(Personal area network)이라고 한다. 개별 AP 는 각각 개별 PAN 을 구성하고, PAN 은 AP 를 공유할 수 없다. AP 는 자신의 고유한 ID(64bit, LONG ADDR, LADDR)를 이용하여 PAN ID(16bit)를 생성한다. 다른 네트워크(인터넷)에 접속할 수 있는 다른 장비나 컴퓨터를 Host 로 AP 와 연결하여 Elicit Network 의 게이트웨이로 사용할 수 있다.

Node 가 PAN 에 가입하는 것을 Join 이라고 정의하고, PAN 에 속하지 않은 상태를 Alone, PAN 에서 벗어나는 것을 Exit 라고 정의한다. Alone 상태일때의 PAN ID 는 FFFF 이다.

Join 은 상위 level 에서 하위 level 을 확장하는 방식으로 구성한다. Node 는 Join 하면서 PAN 의 정보(PAN ID, RF CH, Data rate, Addressing mode 등)를 받아서 설정하고, SHORT ADDR 을 할당 받아서 PAN 의 구성원이 된다.

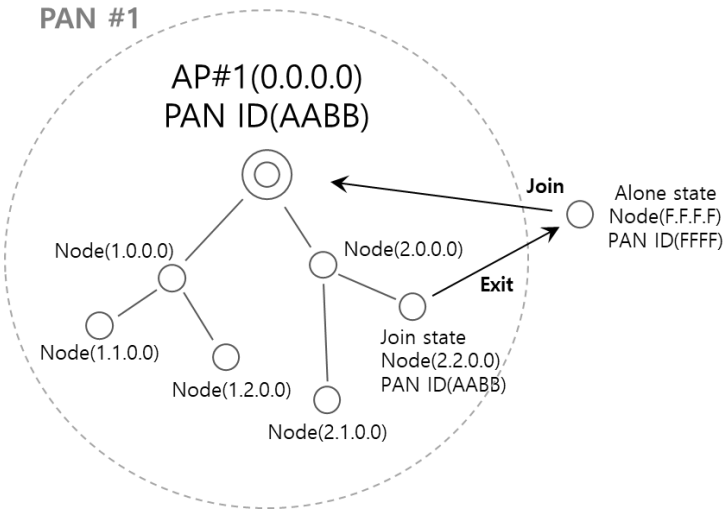


Figure 5 Elicit PAN

16bit 길이의 SHORT ADDR 는 PAN 내부에서 서로를 구분할 수 있는 주소이고 다른 PAN 에 속한 Device 와는 SHORT ADDR 을 사용하여 통신할 수 없다. Alone 상태일 때의 SHORT ADDR 은 FFFF 이다.

LONG ADDR 을 상호 알고 있는 디바이스들은 JOIN 여부와 상관없이 서로 직접 통신이 가능하다. Elicit Network 는 상호 통신이 가능한 모든 디바이스들의 통합된 네트워크를 의미한다.

Elicit Network

- 모든 디바이스는 자신의 고유 ID(64bit LONG ADDR, LADDR)를 가지고 있음
- Elicit PAN: 하나의 AP 를 중심으로 구성한 네트워크
- AP 는 LADDR 을 이용하여 16bit 의 PAN ID 를 생성한다.
- PAN 내부에서는 16bit SHORT ADDR(SADDR)을 사용한다. AP 의 SADDR 은 항상 0000 으로 고정
- PAN 에 가입하는 것을 Join, PAN 에서 나가는 것을 Exit 라고 정의
- PAN 에 속하지 않은 상태는 Alone(Alone state)
- Alone 상태의 PAN ID 는 0xFFFF, SHORT ADDR 은 FFFF 이다.
- PAN ID, SHORT ADDR, 고유 ID(Serial Number)는 모두 16 진수(Hexadecimal)로 표기
- 디바이스간 LONG ADDR 을 알고 있으면 상호 직접 통신 가능(AT+SEND)
- Elicit Network: Elicit protocol 을 통해 서로 통신할 수 있는 모든 디바이스 네트워크

AT Command				
CMD(Ref.)	**Description**	**Set/Get**	**Parameters/Usage**	**Default/Initial**
AT+ROLE	ROLE	Set/Get	0: Nothing, 1: Node, 2: AP	0
AT+PANID	PAN ID	Set/Get	16bit PAN ID	FFFF
AT+ADDR	SHORT ADDR	Set/Get	16bit SHORT ADDR	FFFF
AT+JOINSTS	Join status	Get	(0) Alone, (1) Join	0

2.4 Elicit Routing

Elicit 은 고정된 주소 길이(16bit)를 Hopping mode 마다 할당할 수 있는 주소 범위를 구분하여 최대 4 Hop 까지 지원되는 4 가지의 Hopping mode 를 제공한다. 이를 통해 다양한 IoT 서비스 환경에 유연하게 적용할 수 있고, 간단한 구조를 가지고 있어 주소 자체가 Tree network 에서 Routing table 의 역할을 한다.

SHORT ADDR 를 표기할 때 Hop level 마다 점(dot)을 찍어 level 을 구분한다. 단일 Hop level 만 있는 Hopping mode1 은 점이 없이 (0001)와 같이 표기하고, 2Hop 을 지원하는 Hopping mode2 의 경우 (01.00), Hopping mode3 은 (01.0.0), Hopping mode4 는 (1.0.0.0)와 같은 형식으로 표기한다.

SHORT ADDR 을 할당할 때 상위 level 주소와 자기 주소는 유지하고, 하위 주소만을 신규로 할당한다. 이렇게 주소를 할당하기 때문에 주소만 보면 해당 주소가 Tree 구조에서 어디에 위치한 것인지 쉽게 파악이 가능하다.

Elicit protocol 에서는 동일한 Level 의 Node 들은 서로 통신이 가능한 범위에 위치한다고 가정하고 메시지를 송신한다. 다른 가지에 있는 Node 에게 정보를 전달할 때는 동일 Hop level 일 경우에는 직접 전달하고, level 차이가 나면 자신의 가지에서 목적지 Node 의 level 과 동일한 level 까지 경유(hopping)한 후에 전달한다. 만약 상대방 level 이 자신이 속해 있는 가지의 최하위 Node 보다 낮은 level 이면 메시지는 Hopping 을 통해 전달할 수 없다. 하지만 통신이 가능한 거리에 위치한다면 직접 전달이 가능하다.

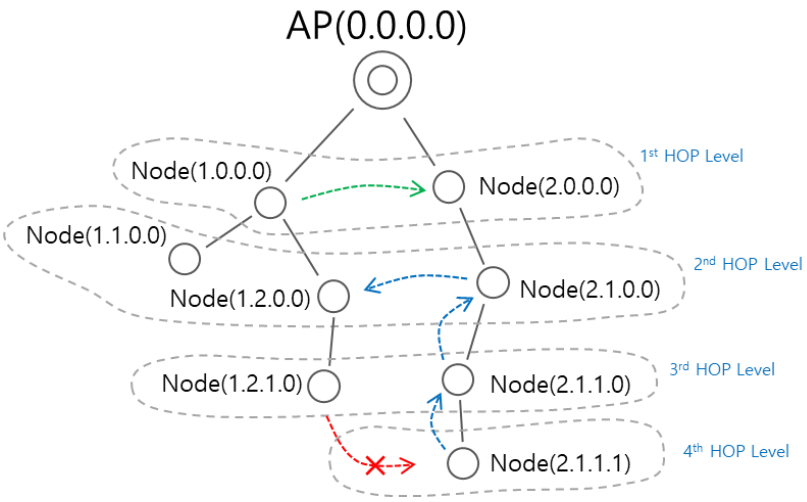


Figure 6 Packet Hopping

[Figure 6 Packet Hopping]에서 왼쪽 가지의 Node(1.0.0.0)에서 동일한 level 인 오른쪽 가지에 있는 Node(2.0.0.0)으로 메시지의 전달은 AP 를 경유하지 않고 직접 전달한다(녹색 화살표). 오른쪽 가지의 최하위 Node(2.1.1.1)에서 왼쪽 가지의 2 번째 level 에 위치한 Node(1.2.0.0)으로 메시지를 보낼 때, 우선 자신의 가지에서 상대방 Node 의

level 까지 hopping 한 후에 전달(파란색 화살표)한다. 하지만 왼쪽 가지의 3 번째 level 에 있는 Node(1.2.1.0)에서 오른쪽 가지의 4 번재 level 의 Node(2.1.1.1)로 메시지를 보낼 때는, 왼쪽 가지에는 목적지와 동일한 4 번째 level 의 하위 Node 가 없어서 Hopping 으로 전달할 수 없다(붉은색 화살표).

2.4.1 Broadcast

개별 Device 에 부여된 주소를 통해 특정한 단일 Device 와 통신하는 것을 Unicast 라고 한다. 동시에 다수의 Device 에게 전송하는 것을 Multicast 라고 하고, 모든 대상을 상대로 전송하는 것을 Broadcast 라고 한다. Elicit 에서 할당된 SHORT ADDR 로 전송하면 Unicast 로 동작하고, 자신의 하위 level 의 주소 범위를 F 로 채우면 해당 level 에 있는 모든 Node 에게 Broadcast(이하 BC)한다. Node(01.00)에서 (01.FF)로 전송하면 Node(01.00)하위의 모든 Node 에게 전달된다.

Broadcast 메시지는 다수의 Device 가 동시에 수신하므로, BC 메시지를 수신한 Device 는 ACK 를 송신하지 않는다.

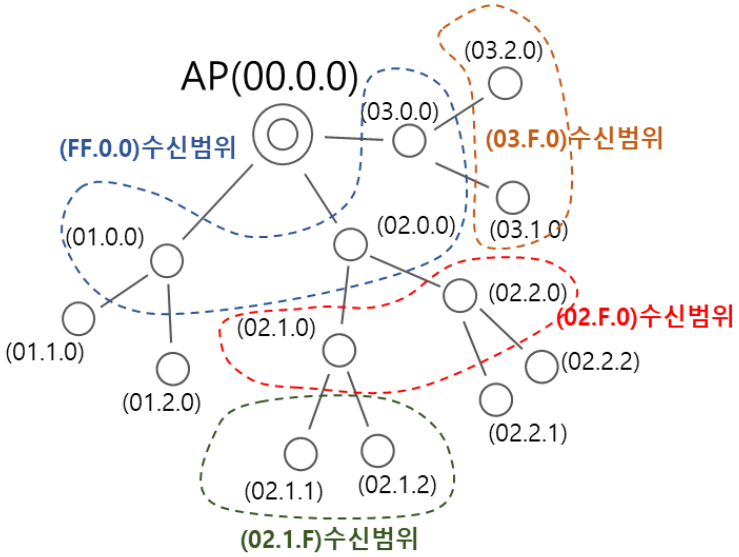
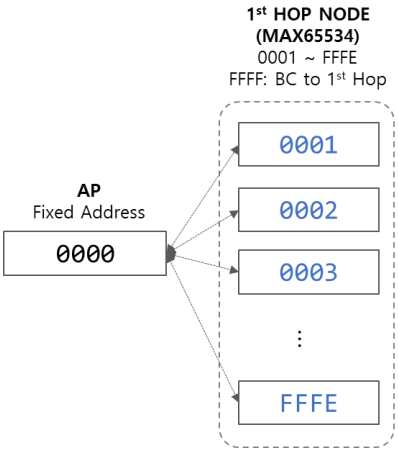


Figure 7 Broadcast 수신 범위

2.4.2 Hopping Mode 1

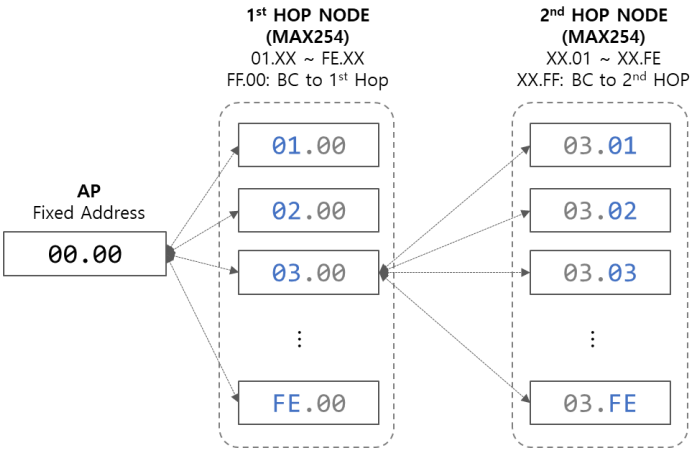
- AP 를 중심으로 모든 Node 가 연결된 Star 구조, 1Hop 만 지원.
- AP 에서 모든 Node 의 SHORT ADDR 을 할당
- 최대 65534 개의 Address 할당 가능
- Broadcast(BC) address 는 (FFFF)



2.4.3 Hopping Mode 2

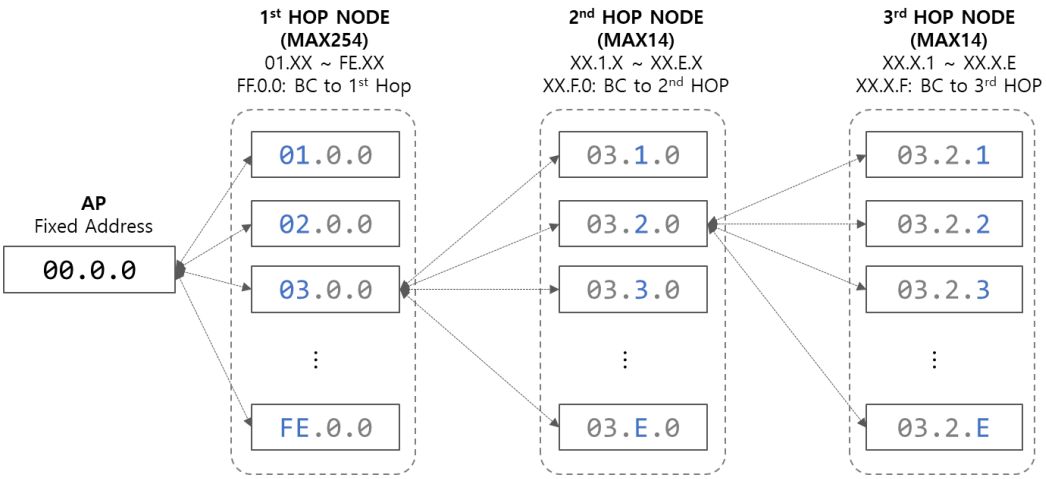
- Tree topology 2Hop 지원, 각 Hop 당 최대 254 개 주소 할당 가능
- 1st Hop 주소 범위: (01.00) ~ (FE.00), 1st Hop 에 대한 Broadcast 주소: (FF.00)
- 2nd Hop 주소 범위: (XX 주.01) ~ (XX.FE), 2nd Hop 에 대한 Broadcast 주소: (XX.FF)
- 1st Hop Broadcast ADDR: (FF.00)
- Broadcast ADDR to 2nd Hop Nodes: (XX.FF)

주) XX: 임의의 1Hop 주소



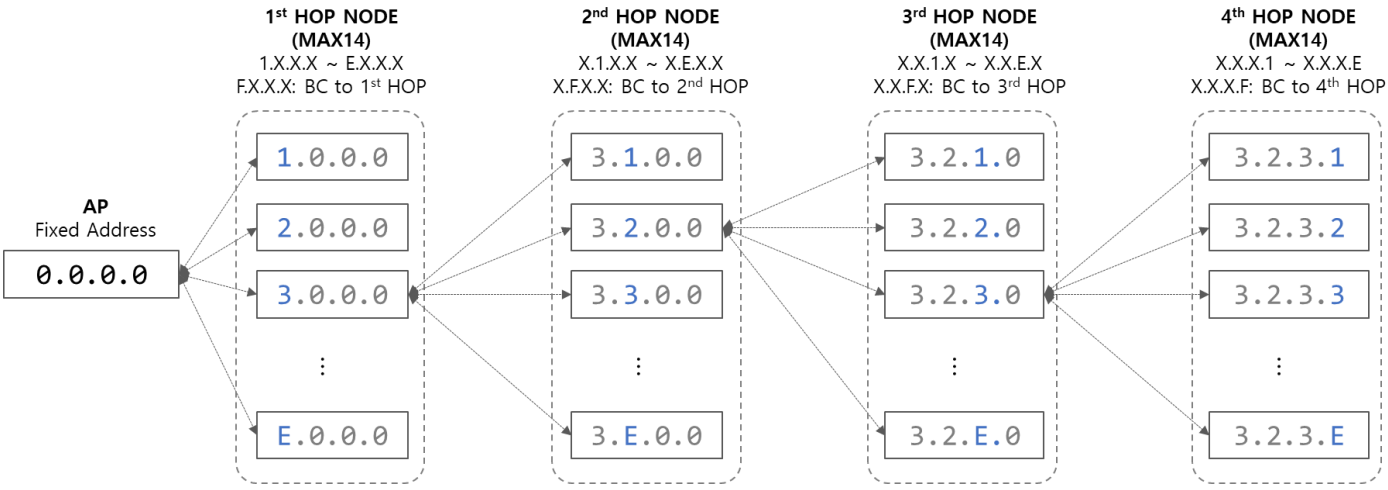
2.4.4 Hopping Mode 3

- Tree topology 3Hop 지원, 1st Hop 254 개, 2nd 와 3rd Hop 은 각각 14 개씩 주소 할당 가능
- 1st Hop 주소 범위: (01.0.0) ~ (FE.0.0), 1st Hop 에 대한 Broadcast 주소: (FF.00)
- 2nd Hop 주소 범위: (XX.1.0) ~ (XX.E.0), 2nd Hop 에 대한 Broadcast 주소: (XX.F.0)
- 3rd Hop 주소 범위: (XX.X.1) ~ (XX.X.E), 3rd Hop 에 대한 Broadcast 주소: (XX.X.F)



2.4.5 Hopping Mode 4

- Tree topology 4Hop 지원, 모든 Hop 은 각각 14 개씩 주소 할당 가능
- 1st Hop 주소 범위: (1.0.0.0) ~ (E.0.0.0), 1st Hop 에 대한 Broadcast 주소: (F.0.0.0)
- 2nd Hop 주소 범위: (X.1.0.0) ~ (X.E.0.0), 2nd Hop 에 대한 Broadcast 주소: (X.F.0.0)
- 3rd Hop 주소 범위: (X.X.1.0) ~ (X.X.E.0), 3rd Hop 에 대한 Broadcast 주소: (X.X.F.0)
- 4th Hop 주소 범위: (X.X.X.1) ~ (X.X.X.E), 4th Hop 에 대한 Broadcast 주소: (X.X.X.F)



2.5 Network Status

Elicit network 상태는 크게 Alone 과 Join 으로 구분할 수 있다. Alone 상태에서 Join 하기 위해서는 우선 접속가능 상태(Joinable state)로 변경된 후 Join 과정이 성공적으로 이루어진 후 Join state 가 된다.

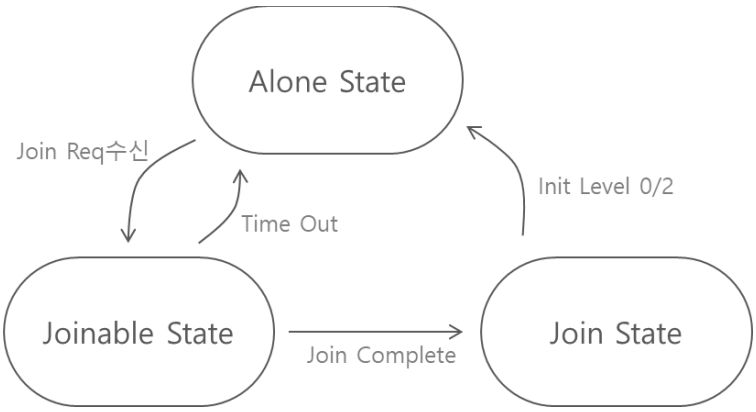


Figure 8 Network State Transition

2.5.1 Alone State

- 출고 후 기본 상태, Elicit Network 에 속하지 않은 상태
- PAN ID: 0xFFFF, SHORT ADDR: FFFF
- Alone 상태에서 상위 디바이스로부터 Join Request 메시지를 수신하면 Joinable 상태로 변경
- Join 상태에서 Alone 상태로 변경은 초기화 명령(AT+FINIT)을 사용

2.5.2 Joinable State

- Alone 상태에서 AP 로부터 Join Request 메시지를 수신하면 Joinable 상태로 변경
- Join Request 에는 PAN ID, RF CH, Data rate 등의 정보가 포함됨
- RF CH, Data rate 변경 후 2 초마다 LED 1 과 2 가 동시에 2 회씩 점멸하여 현재 상태를 표시
- 사용자는 1 분이내에 버튼을 이용하거나, AT Command 를 입력하여 Join 절차 수행
- Joinable 상태에서 1 분 이내 위 Join 절차를 수행하지 않으면 다시 Alone 상태로 변경됨

표 1 Join 방법

버튼 Join 방법

1. 버튼 1 을 5 초이상 누르고 있으면 LED 가 1 초마다 동시 점멸로 변경됨
2. 이 때 버튼을 떼면 자동으로 JOINRSP 메시지를 전송
3. 메시지 전송 후 LED 점멸은 중지되고, Join 절차 수행

AT Command 방법

- AT+JOINRSP FFFF 명령 전송

2.5.3 Join State

- 상위 디바이스로부터 SHORT ADDR 을 할당 받고, JOIN COMPLETE 메시지를 전송
- JOIN COMPLETE 전송 확인 후, 네트워크 설정을 변경한 다음 JOIN 상태로 변경
- 네트워크 설정: Hopping Mode, PAN ID, SHORT ADDR 변경
- Joinable 상태에서 1 분 이내 위 절차가 수행되지 않으면 다시 Alone 상태로 변경됨

AT Command				
CMD(Ref.)	**Description**	**Set/Get**	**Parameters**	**Default/Initial**
AT+FINIT	Initialize the parameters	Set	[INIT LEVEL: 0/1/2]	-

AT+RFCH	RF Channel number	Set/Get	[RF CH NUM: 1 ~21]	15 (943.0MHz)
AT+PHY	PHY Mode	Set/Get	[PHY MODE: 0/1]	1
AT+HOPMODE	Hopping Mode	Set/Get	[HOPPING MODE: 1/2/3/4]	1

2.5.4 Assign address

상위 디바이스에서 하위 레벨로 Join 하는 디바이스의 SHORT ADDR 을 할당한다. 주소 범위는 Hopping mode 에 따라 자신의 하위 레벨에 해당하는 범위에서 할당한다.

주소 관리는 명령[AT+ASADDR]로 할 수 있다. 이 값은 초기값이 0 이며, 신규 할당할 때 마다 1 씩 자동으로 증가시킨다. ASADDR 의 값은 사용자가 직접 변경 가능하고, 이를 이용하여 할당하려는 주소를 조정할 수 있는데, 주소가 겹치지 않도록 각별히 주의하여야 한다.

2.6 Network Configuration

ELICIT-R1 은 최대 출력이 20dBm(100mW)로, 매우 넓은 통신 범위를 가지고 있고 장비의 초기 설정은 모두 동일하다. 따라서 디바이스를 네트워크에 Join 시킬 때 다른 사용자 소유의 디바이스도 접속을 위한 메시지를 수신할 수가 있다. 이러한 원치 않는 접속을 차단하기 위해서 Elicit network 를 구성할 디바이스는 반드시 사용자가 버튼을 누르거나 AT Command 를 입력할 수 있어야 한다.

Elicit network 를 구성하는 과정은 상위 Node 에서 하위 Node 를 Network 에 Join 시키는 과정이다. 즉 AP 에서부터 1st level Node 를 Join 시키고 Join 된 Node 는 자신의 하위에 새로 Node 를 Join 시킬 수 있다. 즉, Elicit PAN 은 AP 의 network 파라메터 설정대로 구성한다. 따라서 PAN 을 구성하기 전에 AP 는 Data rate, RF Channel(이하 RFCH), Hopping mode 등을 미리 설정해야 한다.

[**Figure 9 Join Sequence Diagram**]은 AP 와 Host 가 UART1 을 통해 연결되고, 신규로 Node 를 Elicit network 에 Join 시키는 과정을 나타낸 흐름도이다.

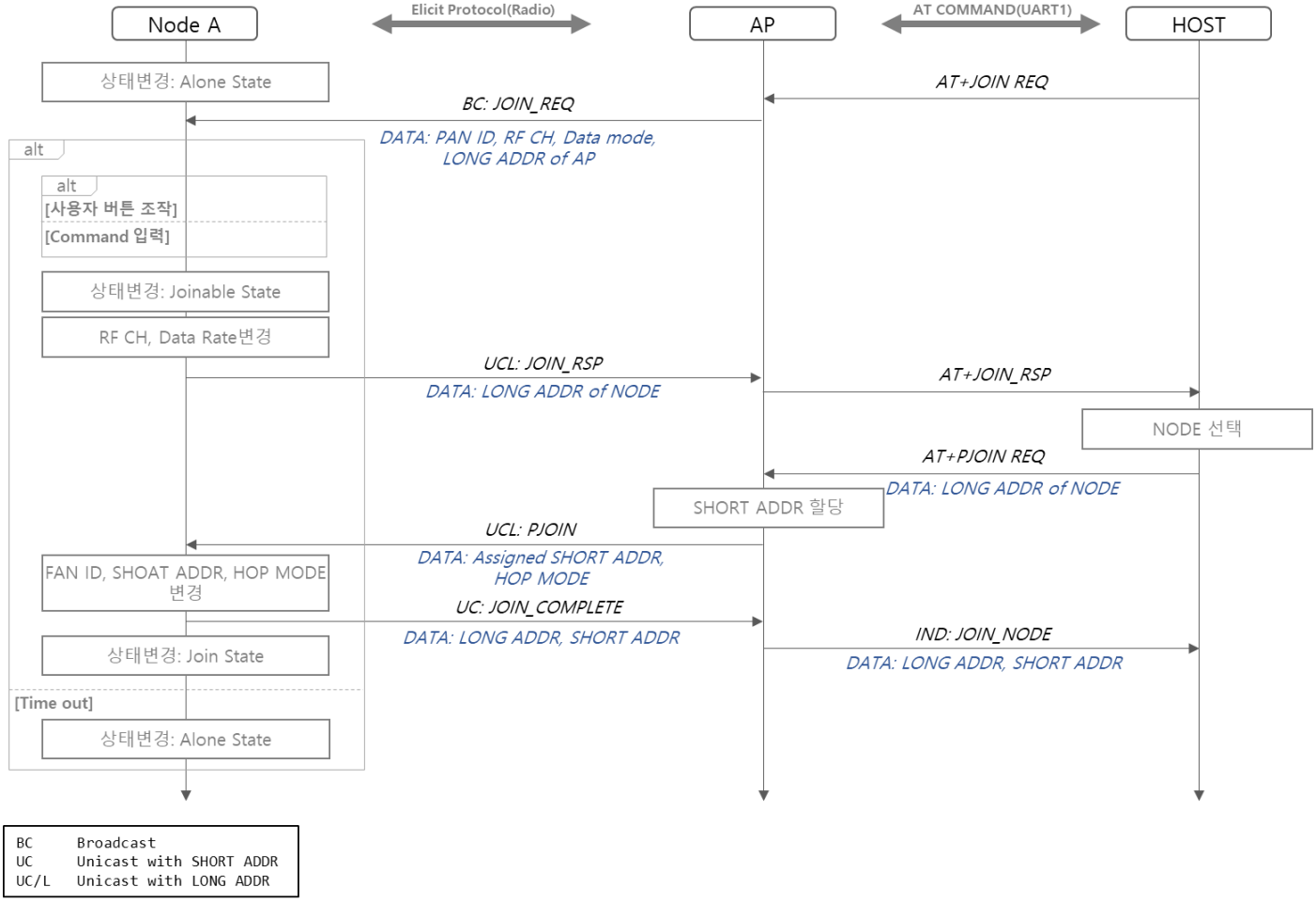


Figure 9 Join Sequence Diagram

Elicit 네트워크를 구성하는 절차의 상세한 순서와 내용을 다음 [표 2 Elicit Network 구성 절차]와 같다.
AT Command 설정과 사용 방법은 [AT command]을 참조

표 2 Elicit Network 구성 절차

단계	내용
1	**네트워크를 구성할 ELICIT-R1 디바이스 준비** - Elicit 네트워크 구성을 위해 최소 2 개 이상의 디바이스를 준비하고 Host 등과 연결 - AP 로 설정할 디바이스는 반드시 UART1 을 통한 AT Command 인터페이스를 가지고 있어야 한다 - 각 ELICIT-R1 디바이스를 초기화[9.1 디바이스 초기화]
2	**AP 지정 및 네트워크 파라메터 설정** - AP 로 운영할 ELICIT-R1 디바이스의 UART1 과 Host(PC, Gateway 등)를 연결 - Host 의 Baud rate 를 115200bps 로 설정 - ELICIT-R1 을 AP 로 설정(현재 SHORT ADDR 은 FFFF) **AT+ROLE=FFFF AP** - AP 로 설정한 후에는 SHORT ADDR 이 0000(ZERO)으로 자동 변경됨 - 무선 channel 상태 스캔(아래 명령은 모든 channel 을 각각 2 초간 scan) **AT+SCAN=0 0 2** - 무선 channel 설정(채널 10 번으로 설정할 경우)

	AT+RFCH=0 10 - Hop mode 2 로 변경(Hopping mode 2) **AT+HOPMODE=0 2**
3	**AP 에서 Join Request(JOINREQ) 메시지를 송출** - 모든 channel 에 있는 ELICIT R1 에게 송신 **AT+JOINREQ=0** - 지정된 channel 에 있는 ELICIT R1 에게 송신(아래 예제는 채널 10 으로 송신) **AT+JOINREQ=0 10**
4	**Node 에서 JOINREQ 메시지에 대해 응답** - Node 에서 JOINREQ 를 수신하면 [Joinable State]로 변경됨 - LED 1, 2 가 2 초마다 동시에 2 회씩 짧게 점멸 - 버튼이나 AT Command 중 하나를 사용하여 응답 - Joinable 상태에서 1 분동안 응답하지 않으면 다시 Alone 상태로 변경됨 - 응답 이후 LED 점멸 중지됨 **[버튼 응답 방법]** - Host 없이 디바이스를 구성하였을 때 버튼을 사용하여 Join 과정을 수행할 수 있음 - 버튼 1 을 5 초 이상 누르고 있으면, 1 초 주기로 LED1,2 가 동시 점멸로 변경 - 이 때 버튼을 떼면 자동으로 RF CH, Data rate 를 변경하고, JOINRSP message 를 전송 **[AT Command 로 직접 메시지 전송 방법]** - 사용자가 직접 물리적인 버튼을 누르지 않고 Host(PC, MCU, SBC 등)를 통해 자동으로 Join 가능 - AT Command JOINRSP message 를 전송(실제 전송할 때 자동으로 LONG ADDR 로 전송) **AT+JOINRSP=FFFF**
5	**AP 에서 JOINRSP 확인 후 해당 Node 선택** - Node 가 전송한 JOINRSP(Node 의 LONG ADDR 정보 포함)를 AP 에서 확인 - Alone 상태의 Node 는 모두 응답할 수 있으므로 다수의 JOINRSP 를 수신할 수 있음 - 수신한 JOINRSP 중에서 등록하고자 하는 Node 의 LONG ADDR 을 확인 후 등록허가 메시지 전송 Ex) LONG ADDR 이 0012345600abcd12 일 때 **AT+PJOIN=0 0012345600abcd12** - PJOIN 메시지에는 자동으로 할당한 Node 의 새로운 SHORT ADDR 을 포함하고 있음
6	**네트워크 구성 확인** - Node 는 PJOIN 메시지 수신 후 자동으로 SHORT ADDR 등의 정보를 변경하고, JOIN 상태로 변경 - Node 는 자동으로 네트워크 등록 완료 메시지인 JOIN CONFIRM 을 AP 로 전송 - AP 는 IND 형식으로 JOIN CONFIRM 수신(IND MSG TYPE 4: JOIN CONFIRM) AT+IND=[ADDR] [RSSI] [BATT] [MSG TYPE] [NEW JOINED ADDR] [NEW JOINED LONG ADDR] Ex) JOIN 한 Node 의 SHORT ADDR 이 0001 이고, LONG ADDR 이 0012345600abcd12 일 때 **AT+IND=0001 -60 100 4 0001 0012345600abcd12** AP 에 연결된 Host 는 JOIN CONFIRM IND 메시지를 이용하여 신규 Node 를 관리할 수 있다

AT Command				
CMD(Ref.)	**Description**	**Set/Get**	**Parameters**	**Default/Initial**
AT+JOINREQ	Request to Join	Set	[channel: 1 ~ 21]	
AT+JOINRSP	Response to Join	Set	[SHORT ADDR]	

AT+PJOIN	Hopping Mode	Set/Get		
AT+IND	Indication of Event	Ind	[EVENT TYPE]	
AT+ASADDR	Count of Assigned Addr	Set/Get	[ASSIGNED ADDR COUNT]	0

3 ELICIT R1 Radio 설정

3.1 RF Channel

ELICIT-R1 은 USN 용으로 할당된 940.1 ~ 944.3MHz 대역을 사용한다. 이 대역에서 사용하는 중심주파수는 [표 3 940MHz 대역의 중심주파수(Channel list)] 와 같이 총 21 개의 채널을 사용할 수 있다. 각 채널 당 점유주파수 대역폭은 200kHz 이다. 디바이스간 서로 송수신하기 위해서는 동일한 채널을 사용해야 한다.

표 3 940MHz 대역의 중심주파수(Channel list)

channel	주파수(MHz)	channel	주파수(MHz)	channel	주파수(MHz)
1	940.2	8	941.6	15	943.0
2	940.4	9	941.8	16	943.2
3	940.6	10	942.0	17	943.4
4	940.8	11	942.2	18	943.6
5	941.0	12	942.4	19	943.8
6	941.2	13	942.6	20	944.0
7	941.4	14	942.8	21	944.2

3.2 TX power

ELICIT-R1 의 최대 송신 출력은 약 20dBm(17.7 ~ 21.4dBm)이다. 출력설정은 AT Command 로 0dBm ~ 21.0dBm 까지 0.1dBm 단위로 설정 가능하다. 출력이 높을수록 전력소모도 높다.

3.3 PHY mode

ELICIT-R1 은 두 가지의 PHY 를 지원한다. 명령[AT+PHY]으로 Long range mode 와 High data rate mode 를 설정할 수 있다. 기본값은 High data rate mode 이다.

Long range mode 는 좀 더 높은 수신감도를 위해 RF bit rate 를 20kbps 로 낮추고, FEC(Forward error correction)를 적용하여 안정적으로 통신이 가능하도록 하였다. FEC 는 원데이터보다 많은 정보를 전송(2 배)하기 때문에 실제 사용자의 데이터 전송속도는 10kbps 이다.

High data rate mode 는 200kHz 씩 할당된 RF channel 을 최대한 효율적으로 사용하기 위해 MSK(Minimum shift keying)을 적용하고, 전송속도는 80kbps 이다. ELICIT-R1 을 운영하면서 전력소모가 가장 많을 때가 RF 전송할 때이다. 그래서 전송 속도가 높으면 동일한 데이터라도 더 짧은 시간동안 전송하기 때문에 전력소모가 더 적다.

표 4 PHY mode

PHY mode	Name	Modulation	RF Bit rate	FEC
L	Long rage mode	FSK	20kbps	Convolutional code(K=7)
H	High data rate mode	MSK	80kbps	None

3.4 Channel Scan

ELICIT-R1 은 무선 채널의 상태를 확인할 수 있도록 다음과 같이 RF channel scan 기능을 제공한다.

- 일정 시간(time unit: second) 동안에 무선 채널을 scan 하여 RSSI 값을 read

- Min/Max/Avg RSSI 값을 출력
- Scan 을 진행 중일때에는 data 송수신 제한
- 모든 channel scan 을 실행하면, 각 channel 의 RSSI 값들과 Max RSSI 가 가장 좋은 channel 을 출력

AT Command				
CMD	**Description**	**Set/Get**	**Parameters**	**Default/Initial**
AT+SCAN	Scan RF channels	Set	[channel: 1~21] [period: 1~60]	
AT+TXPWR	Tx Power	Set/Get	[Tx Power in 0.1dBm]	210

4 ELICIT R1 IO

ELICIT-R1 은 8 개의 GPIO, I2C, ADC, DAC, Button 전용 DI 2 개, LED 전용 DO 2 개와 보조 DI(AUX) 1 개를 가지고 있다. 이를 활용하여 다양한 IoT Device 를 빠르게 개발할 수 있도록 각 IO 마다 사전 정의된 기능들이 탑재되어 있다.

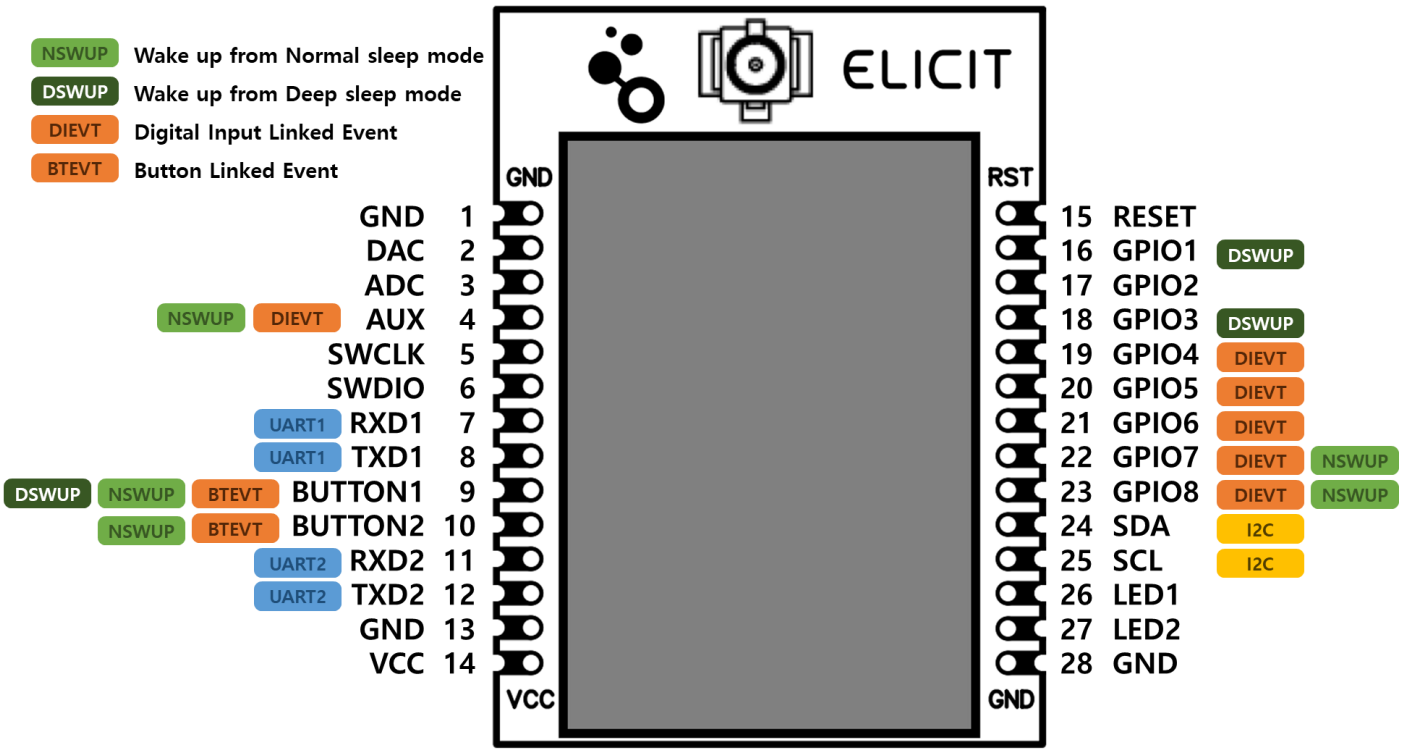


Figure 10 ELICIT-R1 IO

GPIO(Pad 16 ~ 23)는 모두 Input 이나 Output 으로 설정이 가능하다. GPIO4~GPIO8 은 입력 모드(Input mode)에서 입력 상태의 변화(High → Low, Low → High, Both)를 감지하여 이벤트를 발생시킬 수 있도록 설정할 수 있다. AUX(Pad 4)는 입력 전용으로 GPIO 의 Input mode 와 동일하게 사용할 수 있다.

BUTTON1,2(Pad 9,10)는 입력 전용 IO 로, 외부에 버튼을 연결하여 다양한 기능을 수행할 수 있다.

LED1,2(Pad 26,27)은 출력 전용 IO 로, 송/수신 상태나 내부 상태를 표시한다. 사용자가 직접 제어할 수는 없다.

SDA(Pad 24), SCL(Pad 25)는 I2C 전용으로, 내부 10kΩ으로 full-up 되어 있다. SWCLK, SWDIO(Pad 5,6)은 ELICIT-R1 의 MCU 디버깅용 Pad 이다. 사용자는 활용 불가능 하다.

RESET(Pad 15)은 최소 100ns 이상 Low(0.3 x Vcc) 상태를 유지하면 시스템이 Reset 된다. RESET Pad 는 내부에서 Pull-up 되어 있어서 외부에 별도의 전압을 걸지 않고 사용한다.

[Figure 10]에서 **[NSWUP]** 과 **[DSWUP]** 표시는 각각 Normal sleep 이나 Deep sleep 상태에서 외부 입력을 통해 깨어날 수 있는 IO 를 나타낸다. Button 은 사용자가 별도로 설정하지 않아도 누르면 Sleep 상태에서 깨어날 수 있도록 기본 설정되어 있고, 다른 IO 들은 해당 Sleep mode 에서 깨어나기 위해서는 별도로 설정해야 한다.

4.1 GPIO

ELICIT-R1 은 GPIO(General purpose input output) 8 개를 제공한다(Pad 16~23). GPIO 의 내부 구조는 [Figure 11]과 같다. GPIO 의 In/Out, Pull-up/down, Event 등의 설정은 명령[AT+IO]을 사용하여 설정한다.

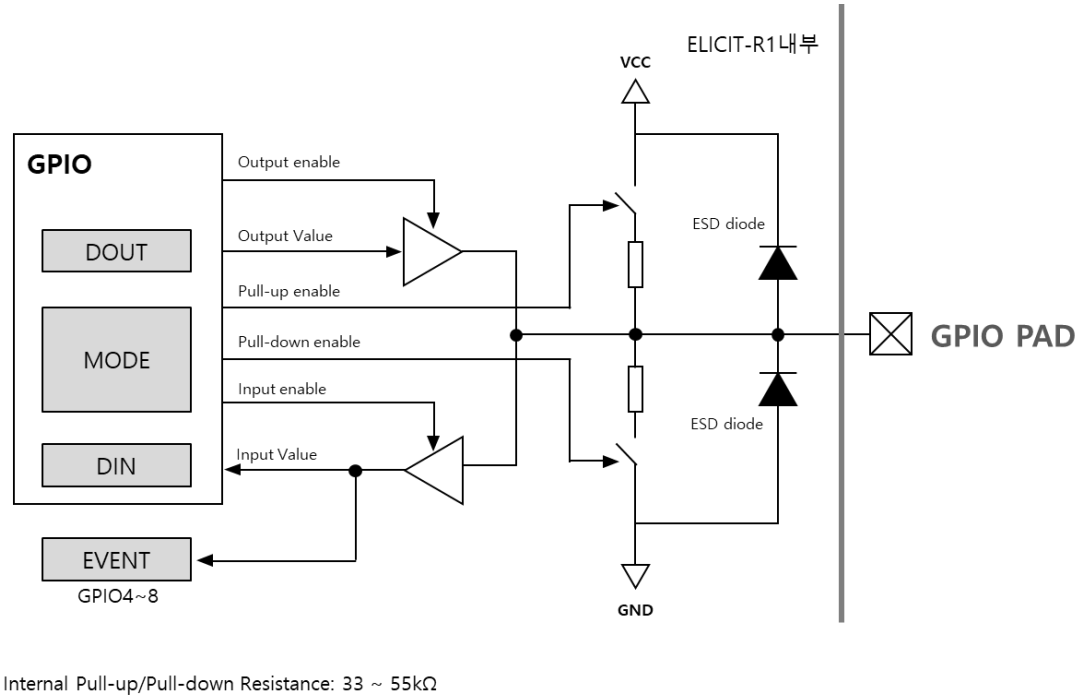


Figure 11 GPIO 의 내부 구조

4.2 DI

- 모든 GPIO 는 Digital Input 으로 설정 가능
- 각 DI 마다 개별적으로 입력모드 Pull-up, Pull-down, no pull 설정 가능
- 내부 Full up/down 저항: 33~55kΩ
- 명령[AT+DI]을 이용하여 해당 GPIO 의 DIN 값을 읽을 수 있음
- DIN 값은 logic high(PAD 전압 > 0.7*VCC)일 때 1 이고, logic low(PAD 전압 < 0.3* VCC)일 때 0 이다.
- GPIO 1, 3 은 Deep sleep 상태에서 깨어날 수 있도록 설정할 수 있음
- GPIO 7, 8 은 Normal sleep 상태에서 깨어날 수 있도록 설정할 수 있음
- GPIO 4~8 은 입력 상태의 변화(Positive edge/Negative edge/Both edge)에 대해 Event 설정 가능
- Positive edge event(HIGH)는 DIN 의 값이 logic low 에서 high 로 변할 때 발생.
- Negative edge event(LOW)는 DIN 의 값이 logic high 에서 low 로 변할 때 발생.
- Both edge event(BOTH)는 DIN 의 값이 변할 때 발생한다.
- 설정한 Event 가 발생하면, Event 메시지를 [AT+IND] 형식으로 출력하여 상태 변화 알림
- 명령[AT+REPORT]를 이용하여 발생한 Event 메시지를 자동으로 AP 에게 전달할 수 있다.
- Event 와 연동하여 사용자가 지정한 AT command 를 수행 가능: Linked Event
- DI event 를 사용할 경우에는, switching noise(chattering)를 방지할 수 있도록 외부 회로를 구성해야 함

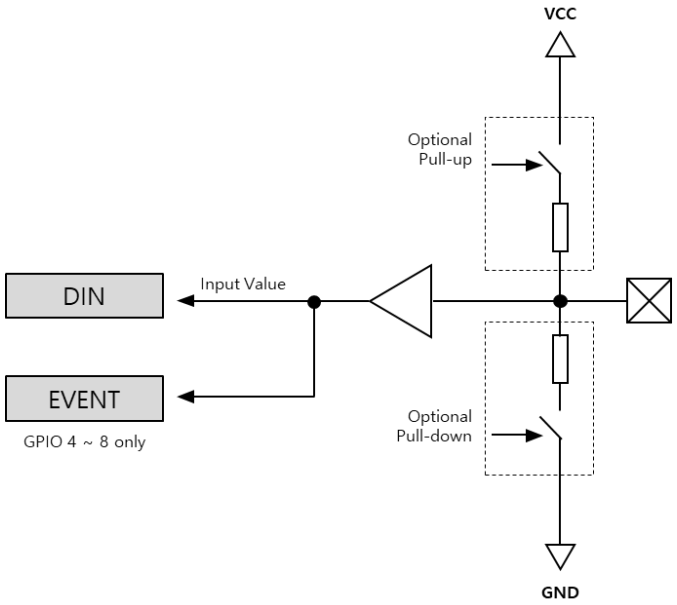


Figure 12 Digital Input 구조

4.3 DO

- 모든 GPIO 는 Digital Output 으로 사용 가능
- 각 DO 마다 개별적으로 출력 모드를 Push-pull, Open-drain, Open-source 중 하나로 설정 가능
- Open-drain 은 옵션으로 Pull up, Open-source 는 옵션으로 Pull down 설정 가능
- 내부 Full up/down 저항: 33~55kΩ
- 명령[AT+DO]을 이용하여 DOUT 값을 읽고, 쓸 수 있음
- GPIO Pad 의 출력은 DOUT 값과 출력 모드에 의해 결정
- 각 DO 마다 개별적으로 DOUT 값을 원하는 주기(time unit: 100ms)로 Toggle 가능

4.3.1 Push-pull out

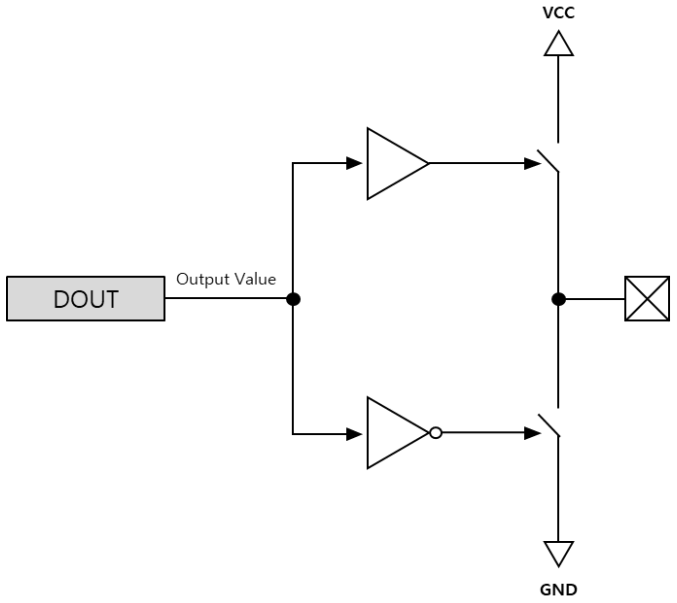


Figure 13 Digital Output(Push-Pull) 구조

- DOUT 값이 1 이면, Pad 는 Vcc 와 연결(GND 는 끊어짐)
- DOUT 값이 0 이면, Pad 는 GND 와 연결(Vcc 는 끊어짐)
- Push-pull 은 내부 Pull up/down 이 없기 때문에, 외부 회로를 구성할 때 단락(Short)되지 않도록 주의
- 단일 GPIO Pad 의 최대 출력 전류는 50mA 이고, 모든 출력의 합이 200mA 를 넘지 않도록 해야함

표 5 Push-pull out table

DOUT	PAD STATUS
0	GND
1	Vcc

4.3.2 Open-drain out

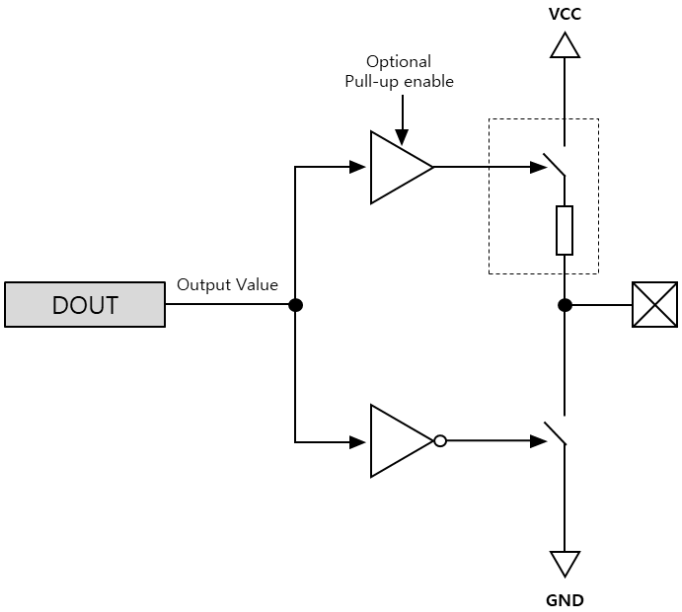


Figure 14 Digital Output(Open-drain) 구조

- DOUT 값이 1 이면, Pad 는 Floating 상태가 됨
- DOUT 값이 0 이면, Pad 는 GND 와 연결
- Open-drain + Pull up 설정에서 DOUT 값이 1 이면, Pad 는 Vcc 와 Pull up 연결
- Open-drain + Pull up 설정에서 DOUT 값이 0 이면, Pad 는 GND 와 연결(Pull up disabled)

표 6 Open-drain out table

DOUT	Open-drain	Open-drain with pull-up
0	GND	GND
1	FLOAT (Any State)	Internal pull-up

4.3.3 Open-source out

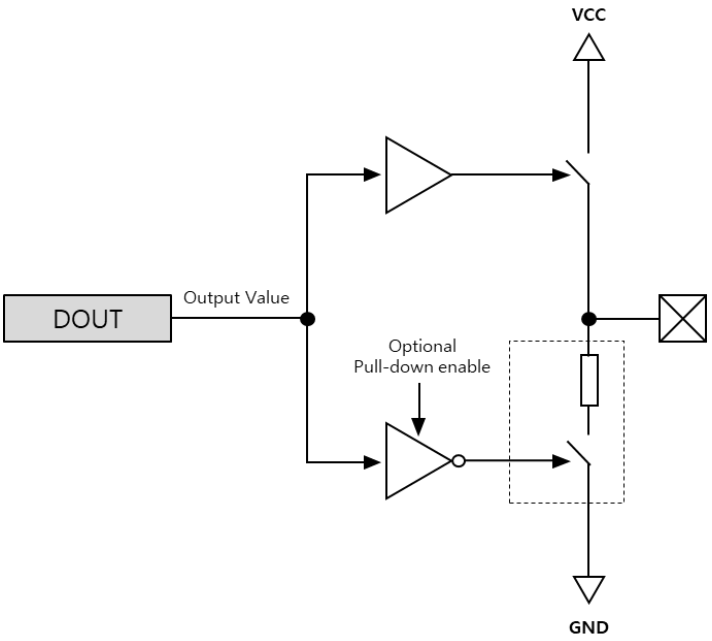


Figure 15 Digital Output(Open-Source) 구조

- DOUT 값이 1 이면, Pad 는 Vcc 와 연결
- DOUT 값이 0 이면, Pad 는 Floating 상태가 됨
- Open-source + Pull down 설정에서 DOUT 값이 1 이면, Pad 는 Vcc 와 연결(Pull-down disabled)
- Open-source + Pull down 설정에서 DOUT 값이 0 이면, Pad 는 GND 에 Pull-down 연결

표 7 Open-source out table

DOUT	Open-source	Open-source with pull-down
0	FLOAT (Any State)	Internal pull-down
1	Vcc	Vcc

4.4 AUX Input

- AUX(Pad4)는 보조 디지털 입력(Auxiliary digital input)
- GPIO 의 DI 설정과 동일하게 입력 모드와 Event 설정 가능
- Normal Sleep mode 에서 깨어날 수 있도록 설정 가능
- DI 와 동일하게 Linked event 설정
- DI event 를 사용할 경우에는, switching noise(chattering)를 방지할 수 있도록 외부 회로를 구성해야 함

AT Command			
CMD	**Description**	**Parameters**	**Notes**
AT+IO	GPIO Config	<IO> <MODE> <TRIGGER>	AUX 포함
AT+DI	Digital Input	<GPIO>	GPIO 만 읽음
AT+AUX	AUX Input		AUX 만 읽음
AT+DO	Digital Output	<GPIO> <DOUT> <TGPERIOD>	

4.5 RESET

- RESET Pad 를 외부에서 최소 100ns 이상 logic low(0.3*Vcc)로 떨어뜨리면 Reset 을 수행한다
- RESET Pad 는 내부에서 Pull-up 되어 있음

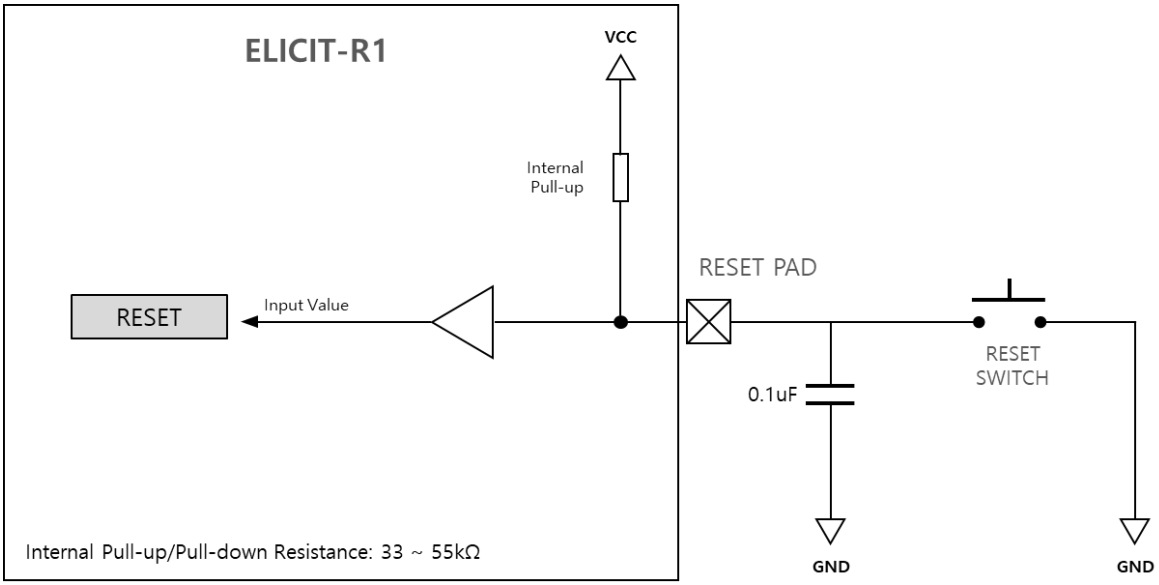


Figure 16 외부 Reset Switch 회로 구성

4.6 ADC

- 12bit ADC, 76.9ksps with x32 oversampling
- ADC(Pad3)와 전원(Vcc, Pad14)의 전압을 측정
- 내부의 1.21V 기준 전압을 사용하여 별도의 외부 기준 전압이 필요 없음
- 측정 범위: 0 ~ Vcc(1.8 ~ 3.8V), Vcc 를 넘는 전압은 측정할 수 없음
- Timer event 를 통해 주기적으로 ADC value read 가능
- 명령[AT+ADC]을 이용하여 ADC 측정을 하고, [AT+IND] 형식으로 결과(단위: mV)를 반환

4.7 DAC

- DAC(Pad2)는 mV 단위로 설정 가능한 전압 출력 전용 Pad
- 내부의 기준 전압을 사용하여 별도의 외부 기준 전압이 필요 없음
- 설정 범위: 0 ~ Vcc(1.8 ~ 3.8V) → Vcc 를 넘는 전압은 설정할 수 없음
- 명령 [AT+VDAC]을 이용하여 DAC 출력 값 설정

4.8 Button

- Button1(Pad9)과 Button2(Pad10)은 버튼 전용으로 할당된 IO, 일반 GPIO 로 사용할 수 없음
- Host 연결 없이 버튼으로 Join, 초기화, 사용자 지정 명령을 수행할 수 있음
- 입력모드로 Pull-up, Pull-down 설정 가능(내부 Full up/down 저항: 33~55kΩ)
- 버튼 눌림, 뗌 상태의 입력 값은 Pull-up/down 에 따라 다름[표 8 Button 모드 별 입력 값]

- Pull-up/down 설정에 따라 내부에서 자동으로 버튼 눌림, 뗌 상태 변환
- Deep sleep mode 에서 Button1 을 누르면 깨어남
- Normal sleep mode 에서 Button1 이나 Button2 를 누르면 깨어남
- **[중요]** 버튼은 Tactile switch(택트 스위치)와 같이 눌렀을 때 접점이 붙는 방식을 사용해야 함
- **[중요]** 반드시 Chattering 방지 회로(debounce)를 외부에 구성하여 사용해야 함[Figure 17][Figure 18]

표 8 Button 모드 별 입력 값

BUTTON INPUT VALUE	Pull-up	Pull-down
누름 (Press)	0	1
뗌 (Release)	1	0

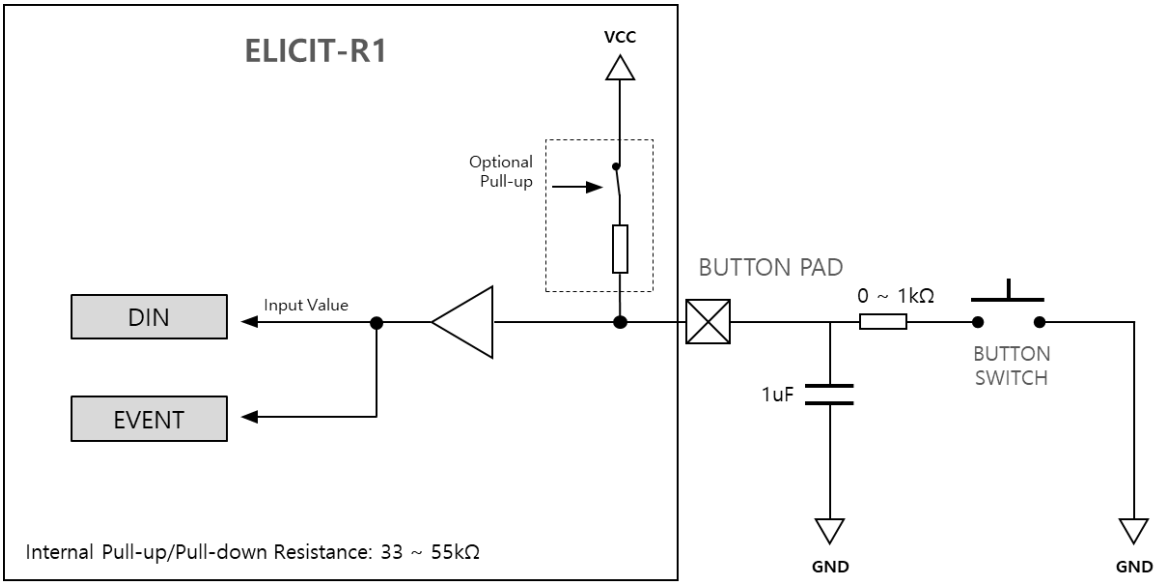


Figure 17 Button 회로(Pull-up, Debounce circuit)

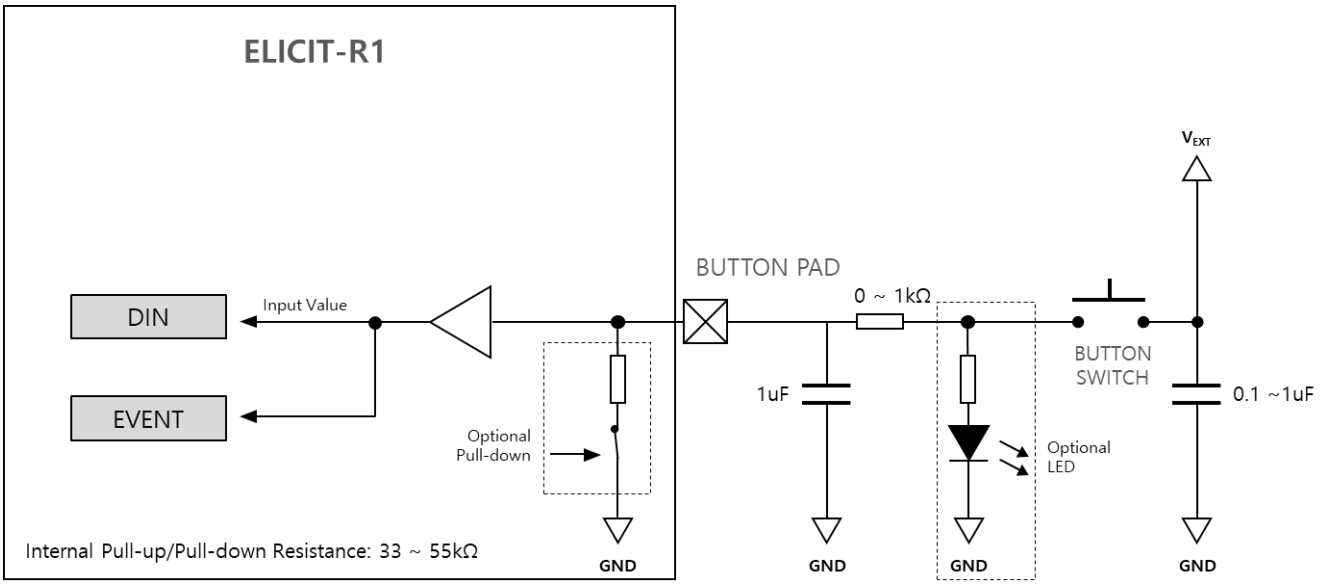


Figure 18 Button 회로(Pull-down, Debounce circuit)

4.8.1 버튼으로 Join

- Joinable 상태의 Node[2.5.2 Joinable State]는 Host 연결 없이 사용자가 직접 버튼으로 Join 가능
- Join by Button 절차
  1. AP 에서 JOINREQ 명령 송신
  2. Join 하려는 Node 에서 JOINREQ 를 수신하면 Joinable 상태로 변경됨
     (2 초마다 LED1, 2 가 동시에 짧게 2 회 점멸)
  3. 버튼 1 을 5 초 이상 누르고 있으면 LED1, 2 가 1 초 주기로 동시에 점멸 상태로 변함
  4. 이 때 버튼을 떼면 Join Response 메시지가 자동으로 전송된다[표 1 Join 방법].
  5. 사용자가 AP 에서 해당 Node 의 LONG ADDR 을 선택[AT+PJOIN]하여 Join 을 완료

4.8.2 버튼으로 초기화(Initialize)

- ELICIT-R1 은 Host 연결 없이 사용자가 직접 버튼으로 초기화를 진행할 수 있음
- 버튼 초기화 모드 진입 절차
  1. button1, button2 를 동시에 누른 상태에서 reset 실행
  2. reset 이후 3 초 이내에 누르고 있던 button1, button2 를 동시에 해제
  3. LED1,2 가 같이 켜지면 초기화 모드 진입
  4. 초기화 모드 진입 후 5 초 이내에 초기화를 수행하지 않으면 초기화 모드 해제

버튼초기화 방법 (초기화 모드 진입 후 5 초 이내 수행)
- 버튼 1 을 5 초간 누름: 네트워크 초기화(Init level 0)
- 버튼 2 를 5 초간 누름: IO, Event 초기화(Init level 1)
- 버튼 1 과 2 를 5 초간 누름: 모든 설정 초기화(Init level 2)

4.8.3 버튼으로 사용자 명령 수행(Button Linked Event)

- ELICIT-R1 은 Host 연결 없이 사용자가 직접 버튼으로 저장된 명령을 수행할 수 있음
- 버튼 1, 2 마다 각각 1 개의 AT Command 를 저장할 수 있음[AT+BTEVT]
- 버튼이 눌릴 때 수행

AT Command				
CMD	**Description**	**Set/Get**	**Parameters**	**Default/Initial**
AT+AUX	AUX Config	Set/Get	[Mode] [PULL] [EVENT]	
AT+BTN	Button Config	Get	[Mode] [EVENT]	

4.8.4 버튼으로 Sleep mode 해제

Sleep again mode 로 진입한 디바이스는 Wake up event 로 깨어나도 설정된 명령만 수행하고 다시 잠들기 때문에 메시지 수신도 불가능하고 사용자가 추가적인 명령할 수도 없다. 이런 경우에 버튼을 이용하여 Sleep again mode 를 해제할 수 있다. 해제한 이후에는 Normal 상태로 동작한다. 다시 계속해서 잠들기 위해서는 Sleep again 옵션을 활성화하여 [AT+SLEEP]명령을 수행해야 한다.

버튼 Sleep again mode 해제 방법(Normal/Deep sleep 공통)
- 버튼 2 를 누른 상태에서 버튼 1 을 1 초간 누름

4.9 LED

LED1(Pad26)과 LED2(Pad27)은 현재 상태를 표시하기 위한 LED 전용 출력 패드이다. 다른 용도로 사용할 수 없다.

표 9 내부 상태별 LED 점등

상태	상태설명	LED
무선 전송(Tx)	무선으로 데이터를 전송 중인 상태	LED1 점등
무선 수신(Rx)	무선으로 데이터를 수신 중인 상태	LED2 점등
버튼 초기화 모드	초기화 설정 모드로 진입한 상태	LED1,2 동시 점등
Joinable State	네트워크 Join 요청(Join Req) 메시지를 수신한 후 접속 가능한 상태(Joinable state)	LED 1, 2 가 2 초 주기로 동시에 2 회씩 점멸
Join Rsp 전송	Joinable state 에서 버튼 1 을 5 초이상 눌러서 Join Rsp 메시지를 보낼 준비가 된 상태	LED1, 2 가 1 초 주기로 동시에 점멸

LED pad 는 Open-drain 으로 고정되어 있다. 사용자는 이를 감안하여 LED 를 다음 회로를 참고하여 연결(Cathode 를 Pad 쪽으로 결선)한다.

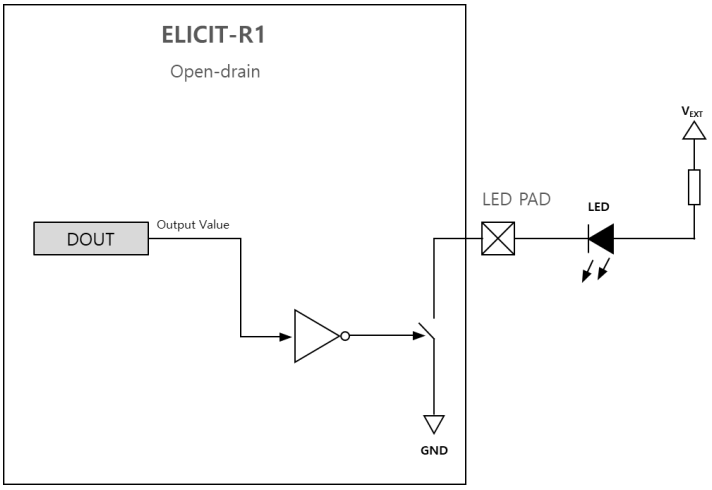


Figure 19 LED 연결(Push-pull, Open-source)

5 Serial Data Interface

Elicit R1 은 Serial data interface 로 2 개의 UART 전용 포트(UART1, UART2)와 1 개의 I2C 전용 포트를 가지고 있다. SWD(Serial wire debug)용의 SWCLK, SWDIO 는 디버깅 전용으로 사용자 지원은 하지 않는다.

IO 전압은 다른 IO 와 같이 VCC(1.8~3.8V)에 연동된다.

5.1 UART

UART1 은 사용자나 Host device 전용으로 AT command 를 입력할 수 있다. UART1 은 기본으로 Local Echo(사용자의 입력을 그대로 출력) 기능이 해제되어 있다. 그래서 UART 터미널(PuTTY, Tera term 등)을 연결했을 때 입력한 문자가 표시되지 않는다. 입력한 문자를 표기하려면 명령[AT+ECHO]을 이용하여 Echo 기능을 활성화해야 한다.

UART2 는 사용자 디바이스를 연결하여 다양한 IoT 서비스를 수행할 수 있다.

UART1/2 로 입력한 데이터는 설정에 따라 자신의 다른 UART 포트로 출력할 수 있을 뿐만 아니라, Elicit 네트워크 내의 다른 Node 의 UART 로 출력할 수 있다.

UART1/2 의 기본 입력 버퍼 크기는 128byte 이다. 따라서 이보다 더 큰 데이터는 나누어서 전송해야 한다.

UART

· 지원 속도(Baud rate)	9600/19200/38400/57600/115200 bps
· 입력 버퍼 크기	128 Byte
· UART1	User console, Host Device, AT command
· UART2	User device, Modbus device

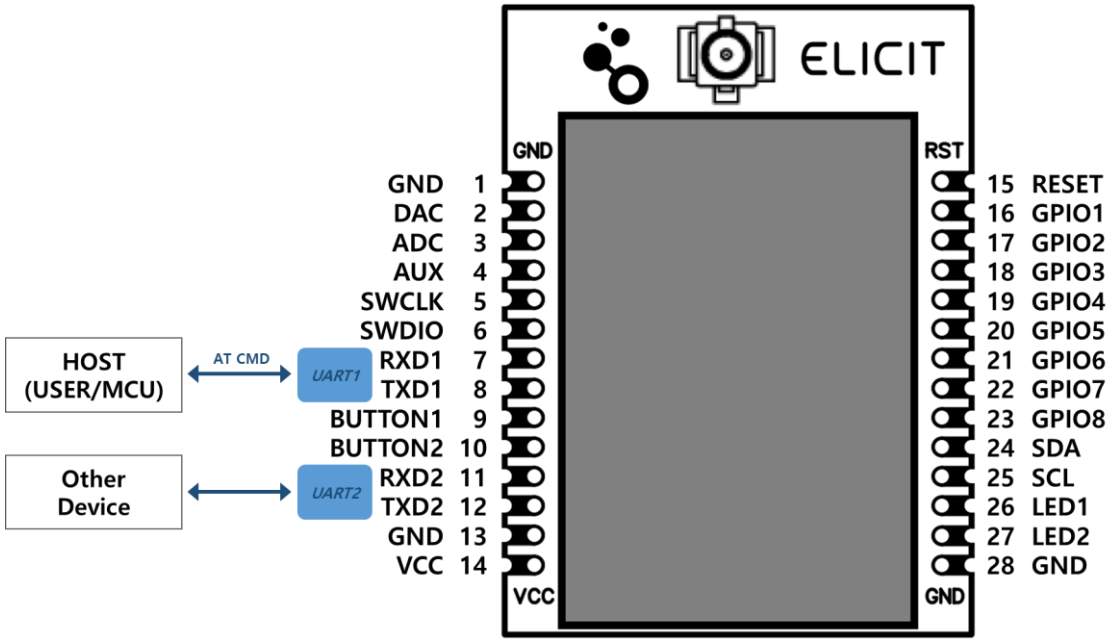


Figure 20 UART 연결

5.2 User data 전송

사용자의 데이터를 전달하는 방법은 AT command 를 이용하거나, UART port 에 입력되는 데이터를 자동으로 전송하는 방법의 두 가지가 있다.

5.2.1 SEND 명령

[AT+SEND] 명령을 이용하여 사용자 데이터를 목적지(최종 수신지), 출력할 UART port 와 형식을 지정하여 전송할 수 있다.

목적지 지정은 16bit SADDR 뿐만 아니라 64bit LONG ADDR(LADDR)을 사용할 수 있는데, AT 명령 중 유일하게 LADDR 을 사용할 수 있다. ALONE 상태의 디바이스라도 LADDR 을 알고 있으면 직접 메시지를 전달할 수 있다. 이때 PHY mode 와 RF CH 이 서로 일치해야 한다.

바이너리 데이터(binary data)를 목적지에서 출력하려고 할 때, 전송하려는 데이터는 16 진수(Hexadecimal)로 표현하여 입력한다. 16 진수 포맷은 접두어 '0x'을 생략할 수 있고, 0~9, A(a)~F(f)의 문자만으로 표현한다. 데이터 구분을 위해 공백을 추가할 때는 전체 데이터를 큰따옴표(")로 묵어서 전송한다. 데이터의 입력 단위는 최소 1byte, 최대 4byte 까지 입력한다.

출력형식으로 'Binary+CRC16' 옵션을 선택할 수 있다. 입력한 데이터에 대해 Modbus 에서 사용하고 있는 CRC16($X^{16}+X^{15}+X^2+1$)을 계산하여 Binary 데이터에 덧붙여서 출력한다. 이를 이용하여 Modbus RTU 를 지원하는 다양한 장비를 연결할 때 유용하게 사용할 수 있다.

사용자 데이터 전송 방법 ([AT+SEND] 명령)

- AT+SEND=<DADDR> <PORT> <FORMAT> <"USER DATA">
- DADDR 전송할 디바이스 선택, LADDR 사용가능
- PORT 출력할 UART 포트 선택
- FORMAT 출력 형식 선택, 문자열(String)이나 바이너리(Binary) 형식 등
- USER DATA 전달할 사용자 데이터, 최대

Hexadecimal 데이터 입력 형식

- 1byte 씩 입력 "A0 B1 12 34", "0xa0 0xb1 0x12 0x34"
- 2byte 씩 입력 "A0B1 1234", "0xa0b1 0x1234"
- 4byte 씩 입력 "A0B11234 A0B2AAFF", "0xa0b11234 0xa0b2aaff"

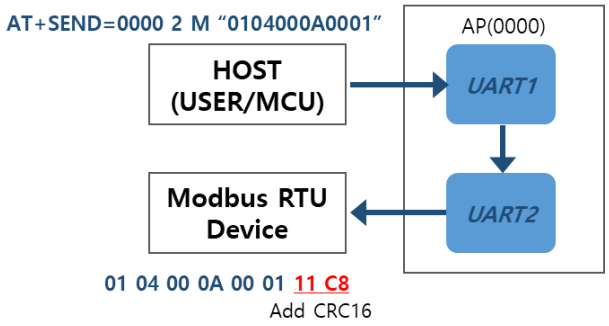


Figure 21 Send data to local UART2, Binary+CRC16 출력

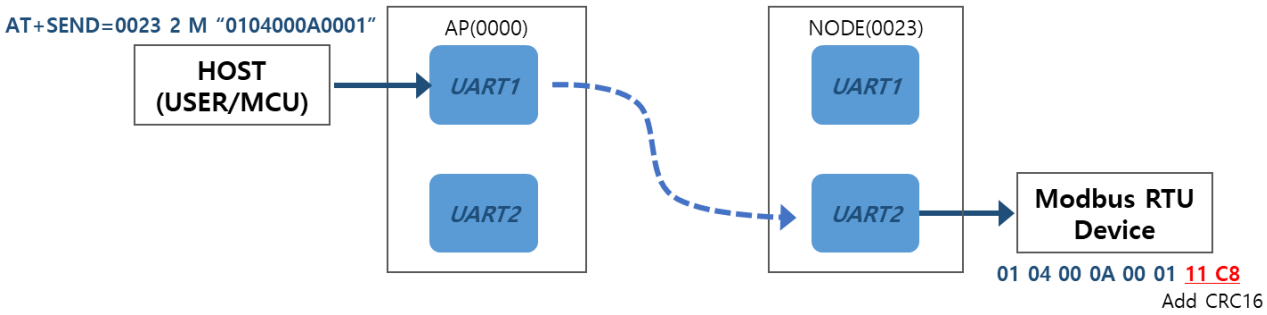


Figure 22 Send data to Other device, Binary+CRC16 출력

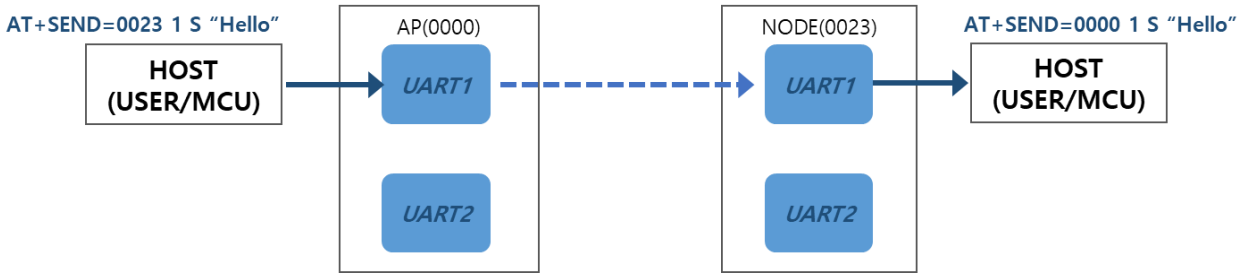


Figure 23 Send data to Other device, String 출력

5.2.2 UART1 자동 전송 모드

[AT+U1SND]명령을 이용하여 UART1 으로 입력되는 모든 데이터를 자동으로 원하는 디바이스의 지정된 포트로 전송하는 모드를 설정할 수 있다. 이 모드가 설정되면 UART1 으로 일반 AT 명령을 입력해도 수행하지 않고, 입력된 데이터를 무조건 지정된 디바이스로 전송한다. 이 모드를 해제하려면 연속으로 "AT#"을 입력하거나, 다른 디바이스에서 전송 중지 명령을 보내야 한다.

전송은 데이터 입력이 2ms 이상 중단되거나, 연속으로 입력된 크기가 128Byte 이상이 되면 자동으로 입력된 데이터를 전송한다.

자동 전송 모드에서 데이터의 출력은 문자열(String) 형식이나, 16 진수(hexadecimal) 형식, 바이너리(Binary) 형식, 입력 데이터 그대로 출력(Pass-through)하는 형식 등으로 지정할 수 있다.

UART1 자동 전송 모드([AT+U1SND] 명령)

- 입력된 모든 데이터를 지정된 디바이스의 지정 포트, 지정한 형식으로 자동으로 출력
- 출력포트를 자기 자신의 UART1 으로 설정할 수 없음
- +SEND 명령과 달리 LADDR 을 지원하지 않음
- AT 명령도 수행하지 않고 전송하기 때문에 명령을 입력하기 위해서는 모드를 해제해야 한다.
- Binary 출력할 때 입력 데이터는 Hexadecimal 형식의 문자열로 입력한다.

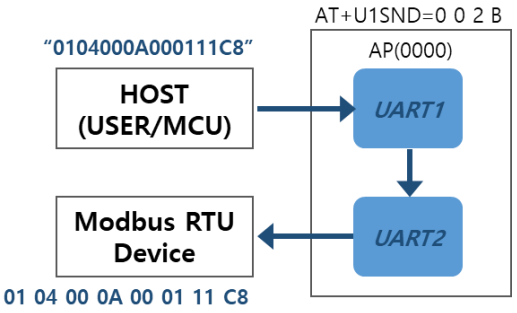


Figure 24 UART1 자동 전달 모드(UART1 to UART2, Binary)

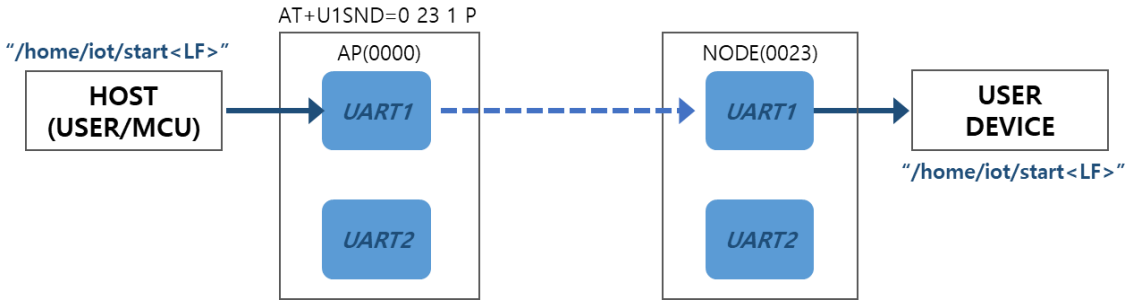


Figure 25 UART1 자동 전달 모드 (Pass-through to UART1)

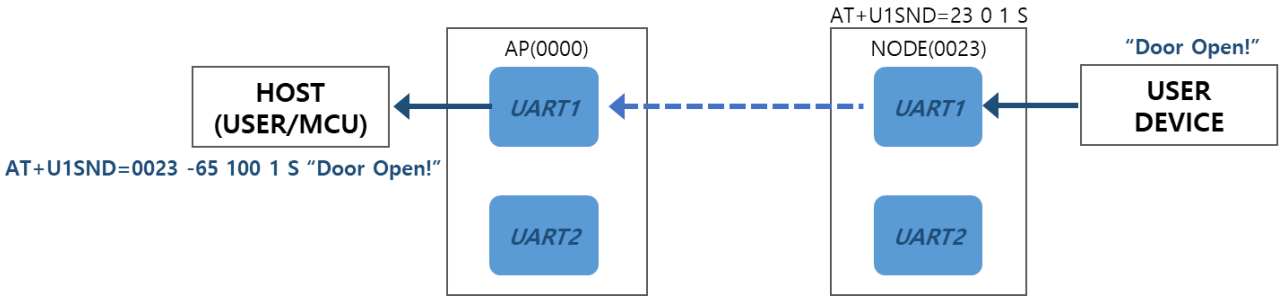


Figure 26 UART1 자동 전달 모드 (Send String data UART1)

5.2.3 UART2 자동 전송 모드

UART1 자동 전송 모드와 같이, [AT+U2SND]명령을 이용하여 UART2 로 입력되는 모든 데이터를 원하는 디바이스로 자동 전송하여 지정된 포트로 출력할 수 있다.

전송 시점은 UART2 로 데이터 입력이 2ms 이상 중단되거나, 연속으로 입력된 크기가 128Byte 이상이 되면 자동으로 입력된 데이터를 전송한다.

자동 전송 모드에서 데이터의 출력은 문자열(String) 형식이나, 16 진수(hexadecimal) 형식, 바이너리(Binary) 형식, 입력 데이터 그대로 출력(Pass-through)하는 형식 등으로 지정할 수 있다.

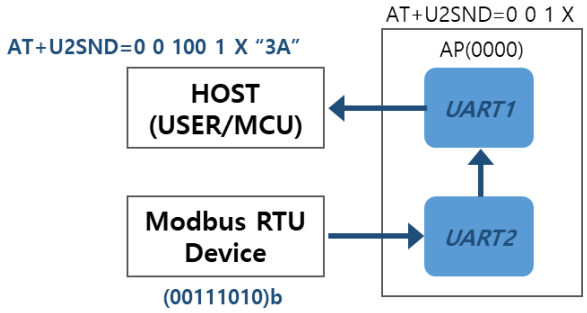


Figure 27 UART2 자동 전달 모드(UART2 to UART1, Hexadecimal)

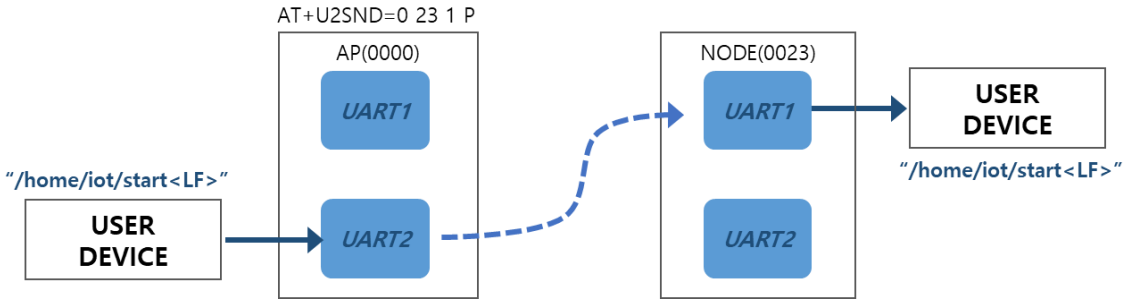


Figure 28 UART2 자동 전달 모드 (Pass-through to UART1)

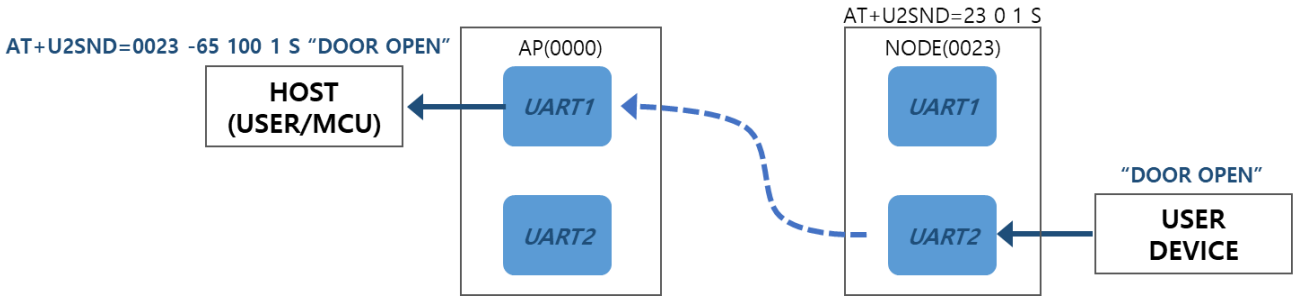


Figure 29 UART2 자동 전달 모드 (Send String data UART1)

ELICIT UART

- UART1 은 사용자 명령(AT command) 전용으로 사용
- UART2 는 다른 장비와 연결용으로 사용

Send User message

- AT 명령으로 사용자 메시지를 원하는 디바이스로 포트와 형식을 지정하여 전송 가능
- AT+SEND 명령은 유일하게 64bit LADDR 을 사용하여 직접 메시지를 전달할 수 있음

UART 자동 전송

- UART1 을 자동 전송 모드로 설정하면 AT 명령을 입력해도 수행하지 않음
- UART1 의 자동 전송 모드 해제: 연속으로 AT#입력(문자 사이에 시간 공백 없이 입력)

5.2.4 Example: 무선 UART 콘솔

장비의 UART 콘솔을 무선으로 연결하는 예제

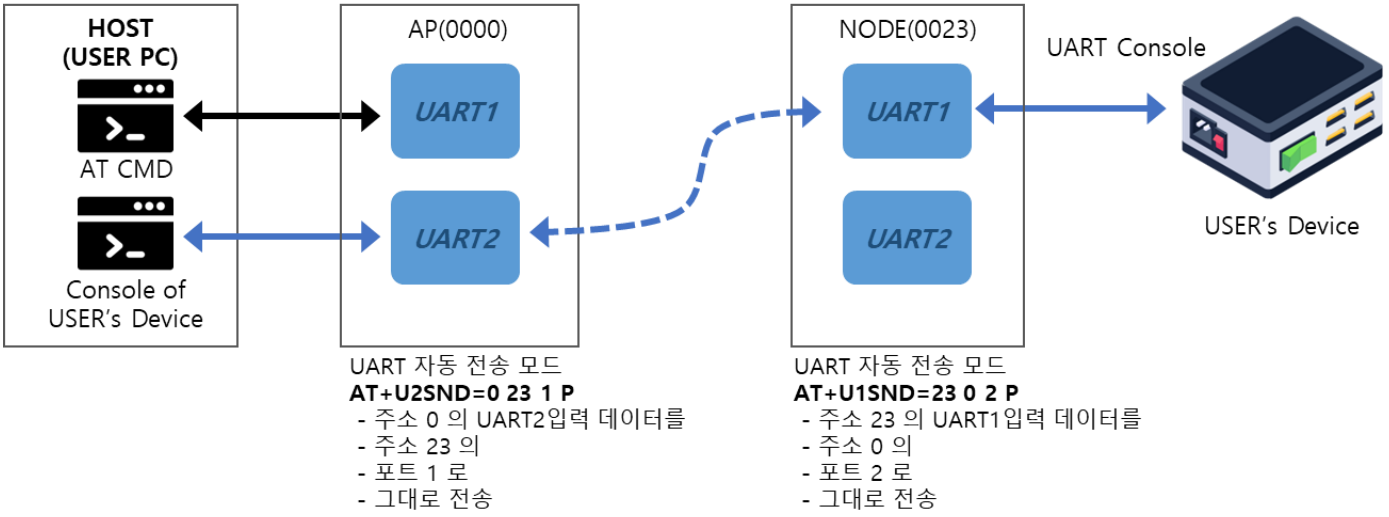


Figure 30 Example: UART Bypass

무선 UART 콘솔 구성 예제

- 원격으로 연결하려는 사용자 장비의 콘솔(Baud rate 38400)을 NODE(0023)의 UART1 과 연결
- AP(0000)에는 Host 용 PC 의 UART1 과 2 를 모두 연결
- AP 의 UART1 과 연결된 터미널은 명령 입력용으로 사용
- AP 의 UART2 와 연결된 터미널은 장비의 원격 콘솔로 사용
- 콘솔의 baud rate 보다 무선 속도가 높아야 원활한 통신이 가능
- AP 의 UART2 baud rate 설정(38400): AT+BAUD=0 1 2
- AP 의 UART2 자동 전달 모드 설정: AT+U2SND=0 23 1 P
- AP 에서 NODE 를 무선으로 설정(AP 의 UART1 과 연결된 터미널 사용)
- NODE(0023)의 UART1 baud rate 설정: AT+BAUD=23 1 2
- NODE(0023)의 UART1 자동 전달 모드 설정: AT+U1SND=23 0 2 P
- 설정 이후에 AP 의 UART2 터미널과 사용자 장비가 직접 연결된 것과 같이 사용 가능

5.2.5 Example: 무선 MODBUS 장비

UART2 에 Modbus 장비를 연결하여 사용하는 예제

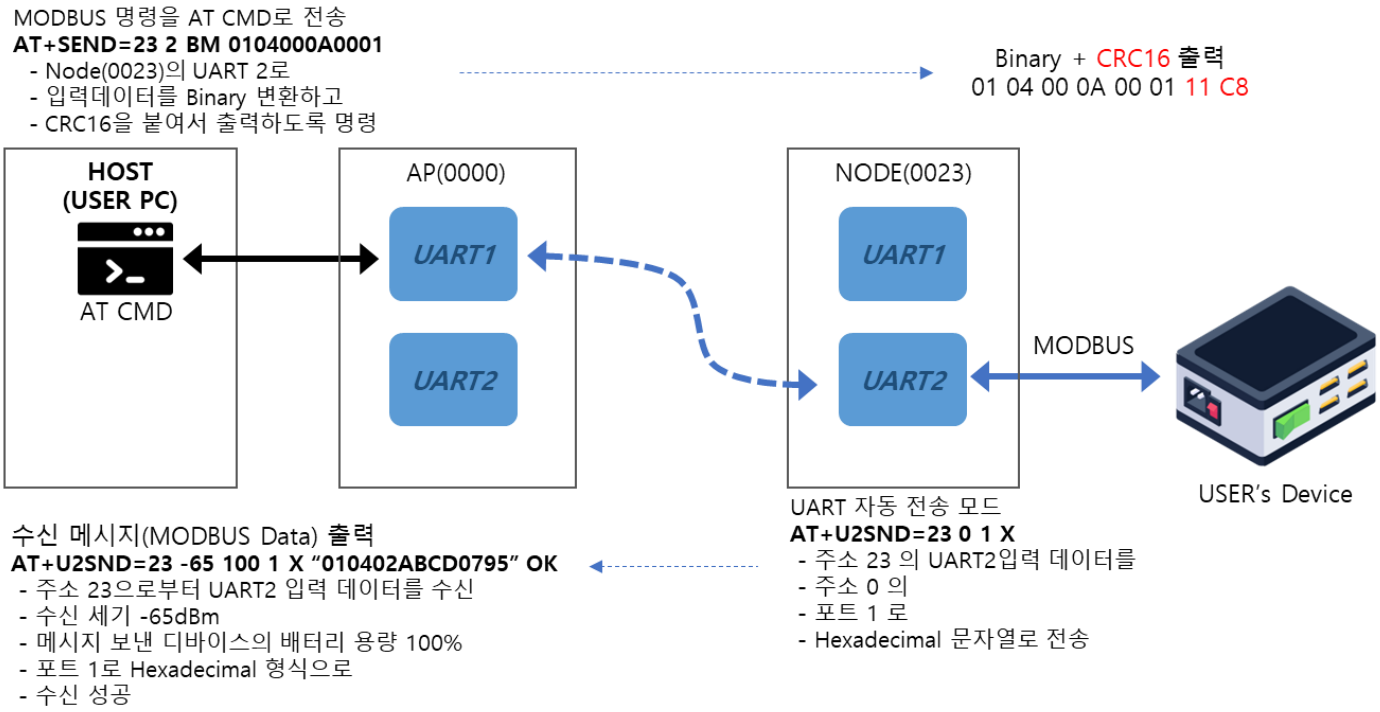


Figure 31 Example: MODBUS 장비 연결

Modbus 장비 연결 구성 예제

- NODE(0023)의 UART2 에 Modbus RTU 를 지원하는 장비 연결
- Modbus 장비와 통신은 AP 의 UART1 터미널에서 AT command 로 수행
- NODE(0023)의 UART2 baud rate 설정(예 19200): AT+BAUD=23 2 3
- NODE(0023)의 UART2 자동 전송 모드 설정: AT+U2SND=23 0 1 X
- AP 에서 장비(국번:01, Function code 4,)주소 10(A)에서 1 개 레지스터 읽기 명령을 전송
  - AT+SEND=23 2 BM 0104000A0001
  - NODE(0023)의 UART2 에서 출력될 때 MODBUS 용 CRC 가 자동을 추가된다
  - Modbus 장비에서 읽기 명령에 대한 응답으로 (010402ABCD0795)h 전송
- AP 에서 수신한 Modbus 장비의 응답 데이터
  - AT+U2SND=23 -65 100 1 X "010402ABCD0795" OK

5.3 I2C

다양한 IoT 장비를 연결할 수 있도록 I2C 통신 기능을 제공한다. Master 모드만 지원하며, 명령[AT+I2C]을 이용하여 쉽게 통신을 할 수 있다. I2C 읽기 명령을 전송한 다음, 수신한 응답 데이터는 [AT+IND]형식으로 반환된다.

I2C 버스의 Master mode 지원

- Standard mode(100 kHz 클럭 속도) 지원
- SCL, SDA 는 내부에서 10kΩ으로 Vcc 에 Pull-up 연결됨
- I2C Write 명령으로 전송할 수 있는 최대 데이터는 4byte
- I2C Read 명령으로 읽을 수 있는 최대 데이터는 16byte
- I2C Read 명령으로 읽은 데이터는 [AT+IND]으로 반환

6 Sleep mode

저전력 운영을 위해 ELICIT-R1 은 전력소모 레벨에 따라 Normal 과 Deep, 두 가지의 Sleep mode 를 제공한다. Sleep mode 에 진입한 상태에서는 RF block 과 Serial data interface(UART, I2C) block 의 전원을 차단하기 때문에 무선 송/수신과 UART, I2C 통신이 불가능하다. AP 는 항상 AT command 수신과 무선 수신 대기 상태이어야 하기 때문에 Sleep mode 로 들어가지 않는다.

표 10 Sleep mode

Sleep mode	Normal Sleep	Deep Sleep
Current consumption	5~6uA	MAX 1uA (Typically 0.6uA)
Wake up event	Sleep Timer, AUX, Button1/2, GPIO7/8	Sleep Timer, Button1, GPIO1/3
Wake up point	Sleep point	Reboot
Wake up time	MAX 800usec	MAX 250msec

일정 시간동안 Sleep mode 로 들어갔다 깨어날 수 있는 Sleep timer 를 Normal/Deep sleep 에서 공통으로 사용할 수 있다. Sleep time event(STEVT)는 Sleep timer 의 Time out 에 의해 깨어났을 때 최대 4 개의 지정된 명령을 수행하는 기능이다.

명령[AT+SLEEP]을 통해 Normal sleep 이나 Deep sleep mode 로 진입할 수 있고, Sleep timer 와 Sleep 에서 깨어났을 때 다시 잠들 것(sleep again option)인지 여부를 설정할 수 있다. Sleep again 이 적용되면 외부 입력이나 Sleep Timer 에 의해 깨어난 후, 설정된 명령이 있으면 이를 수행하고 다시 잠들게 된다.

Sleep timer

- Sleep timer 의 설정 주기는 초(second) 단위이고, 0 으로 설정하면 Timer 는 동작하지 않는다.
- Sleep timer 는 최대 2,073,600 초(24 일)까지 주기를 설정할 수 있다.

6.1 Normal sleep

Normal sleep mode 를 시작할 때는 수행 중인 동작을 멈추고 Sleep 상태로 들어간다. 그리고 깨어날 때는 이전에 Sleep 상태로 들어가기 전 멈춘 시점에서 깨어난다. 만약 전송 대기 중(pending)인 메시지가 있을 때 Sleep 상태가 되면, 깨어난 이후에 전송하게 된다.

Button1 과 Button2 를 누르면 별다른 설정 없이도 Normal sleep 에서 깨어날 수 있다. Button 과 연결된 명령(Button linked event)이 있으면 이를 항상 수행한다. AUX 와 GPIO7, GPIO8 은 IO 설정 명령[AT+IO]으로 Event trigger 를 설정하면, Normal sleep mode 에서 DI event 가 발생했을 때 깨어날 수 있다.

Normal Sleep mode

- Normal sleep 은 메모리 내용을 유지하면서 정지된 상태, 깨어나면 정지된 시점에서 시작
- 깨어나서 다시 시작하기까지 약 800μsec 소요됨
- Sleep timer 와 IO 에 의해 깨어날 수 있는 Event 와 명령을 연결할 수 있다

6.2 Deep sleep

Deep sleep 은 Sleep timer 와 외부 입력(Button1, GPIO1, GPIO3)에 의해서 깨어날 수 있다. 전력 소모를 최소화하기 위해 대부분의 기능을 정지시키고, 깨어날 때는 시스템을 Reset 하여 처음부터 시작한다. 이 때, ELICIT-R1 은 어떤 Event 에 의해 Reset 이 되었는지 확인할 수 있다. Deep sleep 진입 전후, 깨어나서 Reset(Reboot)이 완료될 때까지 GPIO 의 상태는 계속 유지된다.

명령[AT+DSWUP]을 이용하여 Deep sleep mode 에서 깨어날 수 있는 외부입력을 설정할 수 있고, 깨어나서 수행할 AT command 지정은 [AT+DSEVT]명령을 사용한다. 깨어날 수 있는 외부입력이 하나라도 설정되어 있어야 Deep sleep mode 로 진입할 수 있고, 설정된 외부 입력이 없으면 Deep sleep mode 로 들어가지 않는다.

Button1 은 눌렀을 때 Deep sleep 에서 깨어나도록 기본 설정되어 있다. Button2 를 누른 상태에서 Button1 을 누르면 다시 잠들기 기능(Sleep again)이 해제된다.

Deep Sleep 에서 깨어났을 때 RESET Indication 메시지

- button 1 AT+RESET=0100 0 100 1000
- GPIO1 AT+RESET=0100 0 100 1001
- GPIO3 AT+RESET=0100 0 100 1003

6.3 다시 잠들기(Back to sleep again)

SLEEP 명령의 옵션인 AGAIN 을 1(Enable)로 설정하면 Sleep 상태(Normal/Deep 공통)에서 깨어났을 때, Event 와 연결된 명령을 수행하고 다시 잠들게 된다.

Normal sleep 상태에서 Sleep timer 나 GPIO7/8 event 로 깨어나면 연결된 명령(Linked Event)을 수행하고 바로 Sleep mode 로 진입한다. Sleep timer 가 설정된 상태에서 Button 으로 깨어났을 경우에도 해당 Button 에 연결된 명령을 수행하고 다시 Sleep mode 로 진입한다.

Normal sleep 이나 Deep sleep 상태에서 깨어났을 때는 연결된 명령을 수행하고, 다시 잠든다. Button2 를 누른 상태에서 Button1 을 1 초 이상 누르면 다시 잠들기 기능(AGAIN)이 해제(disabled)된다.

AGAIN 을 0(Disable)으로 설정하면, Event 에 의해 깨어난 후, 계속 Normal mode 로 동작한다.

7 Event

ELICIT-R1 에서 Event 는 외부 입력이나, 내부 Timer event 를 감지했을 때 생성하는 메시지이다. 생성된 event 는 Indication message 형태로 사용자에게 UART1 이나 네트워크를 통해 전달할 수 있다. 그리고 event 와 명령을 연결하여 상황에 맞는 다양한 동작을 수행할 수 있다.

표 11 Event 의 종류

Event	Name	Description
Wake up by IO(Deep sleep)	DSEVT	Deep sleep 상태에서 외부 입력에 의해 깨어났을 때 발생
Wake up by IO(Normal sleep)	NSEVT	Normal sleep 상태에서 외부 입력에 의해 깨어났을 때 발생
Wake up by Sleep timer	STEVT	Sleep 상태에서 sleep timer 에 의해 깨어났을 때 발생
Periodic Timer	PTEVT	Normal mode 에서 지정한 주기마다 발생
GPIO(DI)/AUX Digital Input	DIEVT	GPIO(DI)/AUX 입력 상태가 변했을 때 발생
Button Press/Release	BTEVT	Button 을 누르거나 떼었을 때 발생
I2C Read	I2C	I2C 읽기 명령에 대한 응답을 수신했을 때 발생
ADC Read	ADC	ADC 읽기 명령에 대한 응답을 수신했을 때 발생
UART Receive	UART	UART 로 데이터를 수신했을 때 발생

7.1 Indication message

IND 메시지는 event 가 발생했을 때나 내부의 동작 상태 변화(Join Complete, Network Exit)를 알리는 용도로 사용한다. Host 는 IND message 를 통해 IO 의 변화나 동작 상태를 알 수 있다.

표 12 IND Message 를 생성하는 Event

IND Source	Description
DIEVT	GPIO(DI)/AUX Event 가 발생할 경우 IND 생성
RESET	Power on, RESET, Wake up from Deep sleep 등 Reset 이후 발생
I2C Event	I2C 읽기 명령을 완료할 경우 IND 생성
ADC Event	ADC 읽기 명령을 완료할 경우 IND 생성
JOIN COMPLETE	Elicit network 에 JOIN 을 완료할 때 IND 생성
ELICIT EXIT	Elicit network 에서 EXIT 했을 경우 IND 생성

7.2 Report to AP

Node 에서 발생한 IND message 를 자동으로 AP 로 전송하는 기능으로, 명령[AT+REPORT]을 통해 IND 자동 전송 모드를 설정할 수 있다.

8 Linked Event

ELICIT-R1 의 주요 기능인 Linked Event 는 Event 와 사용자 지정 명령을 연결하여 해당 Event 가 발생했을 때 자동으로 지정된 명령을 수행하는 것이다. Event 마다 명령을 저장할 수 있는 공간(Slot)에 차이가 있다.

표 13 Linked Event 의 종류

Event name	Event	Slot	비고
PTEVT	Periodic Timer event	8	Slot 마다 별도의 주기와 명령 연결
STEVT	Sleep timer event	4	깨어났을 때 최대 4 개 명령 순차 수행
DIEVT	DI event (GPIO/AUX)	8	Slot 마다 별도의 DI event 와 명령 연결
BTEVT	Button event	4	BUTTON 마다 별도 명령 연결
DSEVT	Wake up from deep sleep	4	깨운 GPIO 와 명령을 연결
NSEVT	Wake up from normal sleep	4	깨운 GPIO 와 명령을 연결

8.1 Periodic timer linked event (PTEVT)

설정한 주기마다 사용자가 지정한 명령을 수행하는 기능으로 Normal mode 에서 동작한다. 최대 8 slot 까지 지원한다. Timer event 주기는 1 초 단위로 최대 2,073,600 초(24 일)까지 설정이 가능하다. 주기를 0 으로 설정한 Slot 은 수행하지 않는다.

Timer event 는 주기와 함께 offset 도 설정할 수 있어서 동일한 주기를 갖지만 실행하는 시점은 각기 다른 Event 를 설정할 수 있다. 내부 RTC 의 시간(초로 환산한 시간)에 Offset 을 더한 값을 주기로 모듈러 연산을 하여 Timer event 를 실행하는 시점을 결정한다.

Timer Event 발생 시점(t): (RTC + OFFSET) % PERIOD 가 0 이 되는 시간

8.2 Sleep timer linked event (STEVT)

Normal/Deep sleep mode 에서 Timer 를 동작 시키고 Time out 이 발생했을 때 수행할 명령을 4 slot 까지 지정할 수 있다. Sleep 에 진입하기위한 명령[AT+SLEEP]에 Timer 주기를 옵션으로 지정한다. 주기는 1 초 단위로 최대 2,073,600 초(24 일)까지 설정이 가능하다. 깨어났을 때 Slot 에 저장된 명령을 Slot 순서대로 모두 수행한다. 개별 명령은 10ms 간격으로 실행한다.

8.3 DI Linked event (DIEVT)

DI Event 에 대해 Linked event 를 설정할 수 있다. 개별 DI Event 마다 다른 사용자 명령을 할당할 수 있다. GPIO 중 GPIO4 ~ GPIO8 만 Linked Event 를 지원한다. 명령[AT+DIEVT]을 이용하여 설정한다.

8.4 Button Linked event (BTEVT)

Button1 과 Button2 에 대해 각각 Linked event 를 설정할 수 있다. 명령[AT+BTEVT]으로 설정된 Event 는 Button 이 눌릴 때 지정된 명령을 수행한다.

8.5 Wake up from deep sleep linked event (DSEVT)

Deep sleep 은 외부 입력 중 Button1 과 GPIO1, GPIO3 으로 깨어날 수 있다. 최대 4 개의 slot 에 대해 각각 IO 와 수행 명령을 지정할 수 있다.

8.6 Wake up from normal sleep linked event (NSEVT)

Normal sleep 은 외부 입력 중 Button1/2 와 GPIO7, GPIO8, AUX 입력 event 로 깨어날 수 있다. 최대 4 개의 slot 에 대해 각각 IO 와 수행 명령을 지정할 수 있다.

9 기타 부가 기능

9.1 디바이스 초기화

ELICIT-R1 에 저장된 파라메터와 설정 값들의 초기화는 명령[AT+FINIT]이나 버튼으로 할 수 있다. 초기화는 3 가지 레벨로 구성된다.

Init level 0 (Network Initialize)
- 가입(Join)된 Elicit network 정보 초기화
- PAN ID, SADDR, JOINSTS, RFCH 등 초기화
- GPIO, AUX, UART, Event 정보 유지

Init level 1 (IO and Event Initialize)
- GPIO, AUX, UART 등의 설정 초기화
- Timer event, Sleep timer 주기, Sleep timer event 초기화
- Linked event 초기화
- Elicit network 정보 유지

Init level 2 (Factory Initialize)
- 모든 설정 초기화(Init level 0 + level 1)

9.2 펌웨어 업데이트

사용자는 UART1 을 이용하여 ELICIT-R1 의 펌웨어를 다운로드 할 수 있다. [AT+BOOT]명령으로 Bootloader 모드로 진입한 다음, 다운로드 메뉴를 선택할 수 있다. 다운로드는 XMODEM(128 Byte CRC) 프로토콜을 사용한다.

XMODEM 을 이용한 펌웨어 업데이트
- UART1 을 XMODEM 을 지원하는 터미널과 연결
- 명령[AT+BOOT]을 입력하여 Bootloader mode 로 재부팅
- Bootloader menu 다운로드(1), 재부팅(2) 중 다운로드 메뉴 선택(1 을 입력)
- 3 초 이내 재부팅(2)을 선택하지 않으면 자동으로 다운로드 모드(XMODEM Receive)로 변경
- XMODEM 프로토콜로 펌웨어 다운로드
- 다운로드가 완료되면 자동으로 업데이트후에 재 부팅
- 다운로드 모드에서 60 초 이내에 XMODEM 다운로드를 시작하지 않으면 재 부팅

9.3 PER(Packet error rate) test

두 디바이스 간 시험용 패킷을 주고받으며 무선 환경을 시험해 볼 수 있다. 명령[AT+PER]을 사용하여 시험을 진행할 상대방 디바이스, 전송 횟수, 시험데이터 크기, 패킷 전송 사이 간격(msec 단위) 등의 옵션을 설정할 수 있다. PER 시험을 시작하면, 송수신측 모두 진행 상황이 Indication 메시지로 RSSI, 송/수신 성공 카운트 등이 출력된다.

PER 시험
- Tx count (Max 65535), data length (Max 100), Tx interval(50 ~ 3000 ms)을 설정하여 시험한다.
- Test 완료 후 PER, Avg RSSI, Max RSSI, Min RSSI 등의 정보를 표시한다.

10 AT command

10.1 AT Command 전용 UART1

ELICIT-R1 은 UART1 을 통해 사용자 명령을 수행할 수 있다. UART2 는 명령어 입력을 지원하지 않는다. UART1 의 기본 설정은 (표 14)과 같다. UART 의 속도 설정은 명령[AT+BAUD]을 이용하여 변경할 수 있다.

표 14 UART1 의 기본 설정

Parameter	Value
속도(Baud rate)	9600/19200/38400/57600/115200 bps (Default 115200)
데이터비트(Data bit)	8bit
패리티비트(Parity bit)	None
종료비트(Stop bit)	1bit
흐름제어(Flow control)	None

10.2 표기 방식

Elicit 명령어는 다음의 표기법을 사용한다. 괄호와 슬래시 같은 기호는 실제로 입력하지 않는다.

파라메터 간 구분은 공백문자(space, ASCII code 32(0x20))로 한다. 파라메터로 숫자를 입력할 때 '0x'를 붙이면 16 진수로 인식하고, 그렇지 않으면 10 진수로 인식한다. '0x'를 붙이지 않아도 'a(A)~f(F)'를 포함한 숫자는 자동으로 16 진수로 인식한다.

주소(SHORT ADDR/LONG ADDR)은 항상 "0x"를 제외한 Hexadecimal 로 표기한다. 주소를 표기할 때 '0'으로 앞자리 채우기가 허용된다(000A, 00A, A 모두 동일한 주소로 인식)

CR 복귀문자(Carriage return character, ASCII code 13(0xD)), 명령어의 끝을 나타낸다.

LF 개행문자(Line feed return character, ASCII code 10(0xA)), 명령어의 끝을 나타낸다.

<...> 홑화살괄호(꺾쇠, angle brackets)는 생략할 수 없는 파라메터를 표현한다.

[...] 대괄호(square brackets)는 생략할 수 있는 옵션 파라메터를 표현한다.

(...) 소괄호(parentheses)는 명령어 내에서 명령어를 구분하기 위해 사용한다.

/ 슬래시(slash)는 여러 파라메터 중 하나를 선택할 때 파라메터 구분용으로 사용한다.

'C' 작은 따옴표로 표기하는 것은 하나의 문자(ASCII code)를 나타낸다.

"string" 큰 따옴표는 문자열(string)을 나타낸다. 문자열은 공백을 포함할 수 있다.

SADDR 16bit 네트워크 주소(SHORT ADDR)

LADDR 64bit 주소, 고유 ID(LONG ADDR)

OD 발신지(Origin device), 메시지를 최초 송신한 디바이스.

OADDR Address of origin device, OD 의 주소. Hopping mode 구분용 점('.' Comma)은 생략한다.

DD 목적지(Destination device), 메시지의 최종 목적지 디바이스.

DADDR Address of destination device, DD 의 주소. Hopping mode 구분용 점은 생략한다.

HD Hopping device, DD 로 전송하는 경로에서 메시지를 전달하는 디바이스.

T> 명령 수행 예제에서 터미널에 명령을 입력(전송)하는 것을 나타냄. 실제 입력하지 않음.

R< 명령 수행 예제에서 터미널로 출력된 내용을 나타냄. 실제로 출력되지 않음.

10.3 AT 명령어 형식

Elicit 에서 사용하는 AT 명령의 기본 입력 형식은 다음과 같다.

AT+<COMMAND>[<'='/'?'>] [<DADDR>] [<PARAMERTERS> …] <CR/LF/CRLF>

AT+ 모든 명령은 "AT+"나 "at+"로 시작한다

COMMAND 명령어는 항상 대문자로 입력한다.

=/? 해당 명령의 종류를 명시하는 구분자(separator).

'=': 파라메터 쓰기(Set), 기능실행(Execute), 명령/데이터를 다른 디바이스로 전달(Send)하는 명령.

'='다음에는 반드시 해당 명령을 전달할 디바이스나 보낸 디바이스의 주소가 있어야 한다.

'?': 파라메터 읽기(Get), 다른 디바이스에게 데이터를 문의(Query)하는 명령.

DADDR 해당 명령을 전송할 최종 목적지 주소, 자기 주소를 사용할 수 있다.

자기 정보를 읽는(Get) 경우, 생략할 수 있다.

PARAMETERS 명령마다 파라메터의 종류와 개수가 개별적으로 정의되어 있다.

CR/LF AT 명령의 종료를 나타내는 문자, 둘 중의 하나만 입력되어도 되고, 둘 다 입력되어도 된다.

표 15 명령의 종류

종류구분자	종류	내용
=	**Set**	파라메터/설정 값을 쓰기(변경)
	Execute	정의된 기능을 실행
	Send	다른 디바이스로 명령/데이터를 전달
	Receive/Indication	내부/외부로부터 수신한 데이터/메시지/이벤트, 알림
?	**Get**	파라메터/설정 값을 읽기
	Query	다른 디바이스에게 파라메터/데이터를 문의

[AT command 형식 예제]

종류 구분자, 목적지 주소 없이 실행

T>AT+HELP<LF>

T>AT+RESET<CR>

목적지 주소 없는 형식(자신의 파라메터 읽기)

T>AT+ADDR?<CRLF>

T>AT+RFCH?<LF>

목적지 주소 + 파라메터 형식(파라메터 쓰기/전달, '='다음에 명령을 전달할 주소를 쓴다)

T>AT+RFCH=0023 10<LF>

메시지 수신('='다음에 메시지를 보낸 디바이스의 주소가 출력된다)

R<AT+SEND=0021 -80 100 "Hello world" OK<LF>

명령어안에서 명령어 표기(소괄호 사용)

T>AT+DIEVT=0100 4 8 1 (AT+SEND=0 1 SL "Door Open!")<CR>

10.4 AT 명령 응답 형식

AT 명령은 사용자(User/Host)가 입력하고, Elicit 은 그에 대한 결과를 UART1 을 통해 출력하는데, 이를 명령에 대한 응답(response)라고 한다.

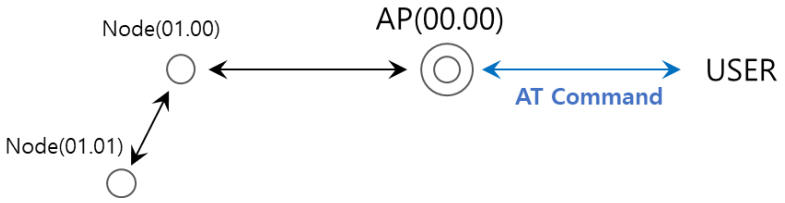
AT 명령에 대한 응답의 기본 형식은 다음과 같다.

AT+<COMMAND>=<DADDR> <RSSI> <BATT> [<DATA> …] <OK/FAIL> [FAIL REASON]<LF>

AT+	모든 명령에 대한 응답은 "AT+"로 시작한다.
COMMAND	수행한 명령어.
=	응답은 항상 등호('=')를 구분자로 사용한다.
DADDR	해당 명령을 수행한 디바이스의 주소, 입력한 명령의 DADDR 과 같다.
RSSI	응답 메시지에 대한 수신신호 세기(Received signal strength indication, Unit: dBm). DASSR 이 자신의 주소이면, 외부에서 수신한 것이 아니기 때문에 값은 0 으로 반환한다. DASSR 이 다른 디바이스이면 전송 경로 상의 첫 번째 디바이스로부터 수신한 신호의 세기.
BATT	해당 응답을 전달한 디바이스의 Vcc 전압 상태, 3V 이상이면 100[%], 2V 이하에서 1[%]로 표기.
DATA	읽은 파라메터(Get)나 요청한 상태에 대한 값, 명령마다 종류와 가짓수가 다르다.
OK/FAIL	해당 명령이 정상적으로 수행되면 OK, 실패하면 FAIL 로 표기.
FAIL REASON	해당 명령이 실패했을 때 실패의 원인/이유.
LF	AT 명령은 항상 LF 로 종료한다.

10.5 AT 명령 사용 예제

Hopping Mode2(00.00)로 구성된 Elicit network 의 AP(00.00)에서 UART1 을 통해 수행한 명령 예제



10.5.1 Get command

[AP 자신의 RF CH 읽기]

T>AT+RFCH?0<LF>

RFCH	RF Channel 값을 읽고(Get), 설정(set)하는 명령
?	Get/Query command 구분자
0	DADDR, 목적지 주소가 발신지 주소와 같으면 생략 가능(Ex: T>AT+RFCH?<LF>). ADDR 입력은 “0000”이나 “0” 모두 가능
LF	명령 입력 종료, <CR>도 사용 가능, <CR><LF>동시 사용도 가능

[명령에 대한 응답]

R<AT+RFCH=0000 0 100 15 OK<LF>

=	명령에 대한 응답 구분자
0000	DADDR, 목적지 주소 AP(00.00), 주소표기에는 Hopping mode 구분 점(‘.’)을 생략한다.
0	RSSI, 자신이 응답할 때는 항상 0[dBm]으로 고정
100	BATT 용량, 목적지의 전원 전압을 응답 전원 전압이 3V 이상일 때 100[%]로 표기
15	RFCH, 현재 설정된 무선 채널 번호는 15
OK	명령이 성공했음을 나타냄
LF	응답의 종료는 나타내는 문자

10.5.2 Querying command

[AP(00.00)에서 Node(01.00)의 GPIO#4 DI 상태 문의]

Node(01.00)의 해당 GPIO 는 사전에 DI 로 설정되어 있어야 한다.

T>AT+DI?0100 4<LF>

DI	DI Pin Pad 의 현재 상태(DIN 값)을 요청하는 명령
?	Get/Query command 구분자
0100	DADDR, 목적지 주소
4	GPIO Number, GPIO#4

[명령에 대한 응답]

R<AT+DI=0100 -75 80 4 1 OK<LF>

0100	DADDR, 목적지 주소, Hopping mode 를 구분하는 점은 생략한다.
-75	RSSI, OD 로 메시지를 전달한 디바이스(01.00)의 신호세기, -75[dBm].
80	목적지 Node(01.00)의 BATT 용량, 2.8V 일 때 80%.
4	GPIO Number, GPIO#4
1	DIN Value, 목적지 Node(01.00) GPIO#4 의 현재 (DIN)값.

10.5.3 Send set command

[AP(00.00)에서 Node(01.01)의 GPIO#7 DOUT 값을 1 로 쓰기]

T>AT+DO=0101 7 1<LF>

DO	DO Pin Pad 의 출력(DOUT 값)을 설정/요청하는 명령
=	Set/Execute/Send command 구분자
0101	DADDR, 목적지 주소
7	GPIO Number, GPIO#7
1	DOUT value, 해당 GPIO DOUT 을 1 로 설정

[메시지 전송에 실패했을 경우 응답]

R<AT+DO=0101 0 100 FAIL 8<LF>

0101	DADDR, 목적지 Node 주소
0	RSSI, 메시지 전송을 실패한 경우, RSSI 는 0[dBm]으로 반환.
100	BATT 용량, 메시지 전송 실패로, 자신(OD)의 BATT 용량 반환, 100%.
FAIL	명령 수행 실패
8	FAIL REASON, 8: TX_NO_ACK 송신을 하였으나, ACK 를 수신하지 못함. 5: TX_CHANNEL_BUSY RF CH 이 혼잡하여 송신을 못함(CSMA/CCA Fail)

[메시지 전송에 성공했을 경우 응답]

R<AT+DO=0101 -75 100 7 1 OK<LF>

0101	DADDR, 목적지 Node 주소
-75	RSSI, 응답 메시지를 수신한 신호 세기, -75[dBm] OD 로 메시지를 전달한 디바이스(01.00)의 신호세기
100	BATT 용량, OD 로 메시지를 전달한 디바이스(01.00)의 BATT 용량, 100%
7	GPIO Number, GPIO#7
1	DOUT value, 해당 GPIO DOUT 을 1 로 설정
OK	발신지(AP)에서 명령 전송이 성공(ACK 수신확인)했을 경우, OK 로 응답 목적지(01.01)로 가는 경로에 있는 첫 번째 디바이스(01.00)로 메시지가 전달된 것을 의미. 목적지까지 전송이 성공했음을 의미하지는 않음. 설정 확인을 위해서는 목적지 디바이스의 값을 읽어(Get) 봐야 함.

10.5.4 Send user data command

[Node(01.00)의 UART1 으로 메시지 보내기(Send message)]

T>AT+SEND=0100 1 S “Hello world”<LF>

0100	DADDR, 목적지 Node 주소
1	DD(Destination device)의 UART 포트 번호
S	DD 의 출력 형식: String

[메시지 전송에 성공했을 경우 AP 의 응답]

R<AT+SEND=0100 -75 100 OK<LF>

0100	DADDR, 목적지 Node 의 주소
-75	RSSI, 메시지 전송 경로 상의 첫 번째 디바이스가 송신한 ACK 의 신호 세기, -75[dBm]

```
        100     BATT 용량, 메시지 전송 경로 상의 첫 번째 디바이스의 BATT 용량, 100%
        OK      발신지(00.00)에서 명령 전송이 성공(ACK 수신확인)했을 경우, OK 로 응답

[목적지 Node(01.00)에서 수신하여, 목적지의 UART1 으로 출력하는 응답]
R<AT+SEND=0000 -80 100 "Hello world" OK<LF>
        0000    OADDR, 메시지를 보낸 발신지의 주소(AP)
        -80     RSSI, 메시지 전송 경로 상의 마지막 디바이스가 송신한 메시지의 신호 세기, -80[dBm]
                여기서는 AP(00.00)이 전달한 메시지에 대한 수신신호 세기.
        100     BATT 용량, 발신지(AP)의 BATT 용량, 100%
        OK      정상적으로 메시지를 수신했을 경우, OK 로 응답
```

10.6 AT+HELP

AT command list 를 출력
실행(Execute)만 지원, 명령 구분자('?'/'='), DADDR 없이 명령 단독 실행

```
[Execute format]
AT+HELP<CR/LF/CRLF>

[Execute Response]
R< ------------------------------------------------------------------------------
R< Get:  AT+<CMD>?<[DADDR]>
R< Set:  AT+<CMD>=<DADDR> <PARAMS> [<PARAMS> …]
R<------------------------------------------------------------------------------
R<HELP        - print all commands List
R<SN          - print serial number. Usage: AT+SN?<ADDR>
...
```

10.7 AT+VER

ELICIT-R1 의 버전 출력
읽기/문의(Get/Query) 지원

```
[Get format]
AT+VER?[DADDR]<CR/LF/CRLF>
        DADDR           목적지 주소, 생략하면 자기 정보를 반환

[Get Response]
AT+VER=<DADDR> <RSSI> <BATT> <HW VER> <SW VER> <PRDCT TIME> <LOT NUM> OK<LF>
        DADDR           목적지 주소
        RSSI            수신신호세기[dBm]
        BATT            배터리용량[%]
        HW VER          HW Revision version, 1: 2022.08
        SW VER          SW version, 100: 1.00
        PRDCT TIME      Production time, 제품 검사 일시("2022/10/31 14:0:0")
        LOT NUM         Production LOT, 제품 생산 Lot, "L22Q301" : Lot '22, 3 분기 1st 생산분
```

```
[Example]
T>AT+VER?<CR>
R<AT+VER=FFFF 0 100 1 100 “2022/10/31 14:0:0” “L22Q301” OK<CR>
```

10.8 AT+SN

ELICIT-R1 의 고유 ID(LONG ADDR) 출력
읽기(Get)만 지원, 다른 디바이스에게 문의(Query) 불가
16bit DADDR 지정 없음

```
[Get format]
AT+SN?<CR/LF/CRLF>

[Get Response]
AT+SN=<SADDR> <RSSI> <BATT> <LADDR> OK<LF>
        SADDR           자신의 SADDR
        RSSI            수신신호세기[dBm]
        BATT            배터리용량[%]
        LADDR           LONG ADDR

[Example]
T>AT+SN?<CR>
R<AT+SN=FFFF 0 100 1234567890ABCDEF OK<CR>
```

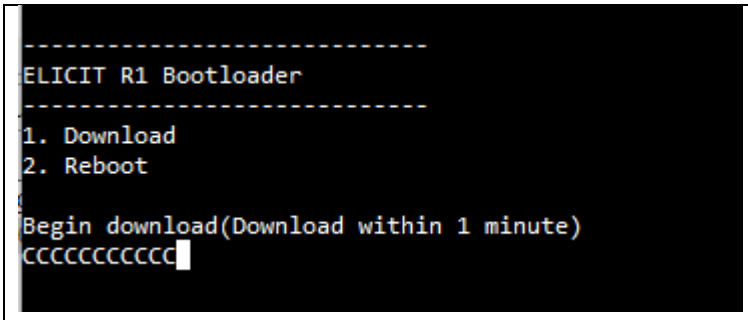
10.9 AT+BOOT

Bootloader mode 로 재부팅
실행(Execute)만 지원, 명령 구분자('?'/'='), DADDR 없이 명령 단독 실행

```
[Execute format]
AT+BOOT<CR/LF/CRLF>

[Execute Response format]
부트로더(bootloader) 모드로 재부팅(Reboot to bootloader mode)
펌웨어 다운로드(1)나 재부팅(2) 중 선택 가능
3 초 이내 재부팅(2)을 선택하지 않으면 자동으로 다운로드 대기(XMODEM Receive) 상태로 변경
60 초 이내 XMODEM 다운로드를 개시하지 않으면 자동으로 재부팅

[Execute Example]
T>AT+BOOT<CR>
```



10.10 AT+RESET

Reset ELICIT-R1
실행/전달/알림(Execute/Send/Indication) 지원

[Execute/Send format]
AT+RESET[=DADDR]<CR/LF/CRLF>

[Execute/Send Response]
AT+RESET=<DADDR> <RSSI> <BATT> OK<LF>
DADDR 목적지 SADDR
RSSI 수신신호세기[dBm]
BATT 배터리용량[%]

[Execute Example]
T>AT+RESET<CR>
R<AT+RESET=0000 0 100 OK<CR>
R< ... Booting messages ...

[Send Example]
#AP(0)에서 Node(0100)에게 Reset 명령 전달
T>AT+RESET=0100<CR>
#Reset 으로 Booting 하면서 Reset 의 원인을 IND 메시지로 알림
R<AT+IND=0100 0 100 RESET 40 OK<CR>

[Reset Indication format]
AT+IND=<DADDR> <RSSI> <BATT> <EVENT:"RESET"> <REASON> <CR/LF/CRLF>
REASON Reset 원인
0x1: Power on Reset
0x2: Pin Reset (RESET Pad)
0x3: Power on Reset + Pin Reset
0x4: Wake up from Deep sleep
0x8: Watchdog 0
0x10: Watchdog 1
0x40: Reset command(AT+RESET)
0x80: VCC Fail(Digital Block)
0x100: VCC Fail(Digital LE Block)
0x200: VCC Fail(Digital DEC Block)
0x400: VCC Fail(Analog Block)

```
                        0x800:  VCC Fail(IO Block)
                        0x1000: Wake up from Deep sleep by BUTTON1
                        0x1001: Wake up from Deep sleep by GPIO 1
                        0x1003: Wake up from Deep sleep by GPIO 3
```

10.11 AT+FINIT

Initialize parameters

실행(Execute) 지원, 쓰기/전달/읽기/문의(Set/Send/Get/Query)불가

```
[Execute format]
AT+FINIT=<DADDR> <INITLEVEL><CR/LF/CRLF>
        INITLEVEL       Initialize level<0/1/2>
                        0       Network Initialize(PANID, SADDR, JOINSTS, RFCH, PHY)
                                IND(EXIT) 알림 메시지를 자동 생성한다
                        1: IO/EVENT Initialize(IO, Timer, Sleep, LINK EVENT)
                                IND(INIT) 알림 메시지를 자동 생성한다
                        2: Initialize all(Factory Initialize)
                                IND(INIT) 과 IND(EXIT) 알림 메시지를 자동 생성한다

[Execute Response]
AT+FINIT=<SADDR> <RSSI> <BATT> OK<LF>
        SADDR           자신의 SADDR
        RSSI            수신신호세기[dBm]
        BATT            배터리용량[%]

[Example]
T>AT+FINIT=0100 2<CR>
R<AT+FINIT=FFFF 0 100 OK<CR>
```

10.12 AT+ADDR

Parameter: Short Address(SADDR), Elicit network address

읽기/쓰기(Get/Set) 지원, 다른 디바이스로 전달/문의(Send/Query) 불가

Role 이 AP 인 디바이스는 변경 불가

```
[Get format]
AT+ADDR?<CR/LF/CRLF>

[Get Response]
AT+ADDR=<SADDR> <RSSI> <BATT> <SADDR> OK<LF>
        SADDR           자신의 SADDR
        RSSI            수신신호세기[dBm]
        BATT            배터리용량[%]
        SADDR           자신의 SADDR

[Set format]
```

```
AT+ADDR=<SADDR><CR>/<LF>
        SADDR           변경하려는 SHORT ADDR

[Set Response]
AT+ADDR=<SADDR> <RSSI> <BATT> <SADDR> OK<LF>
        SADDR           변경된 자신의 SADDR
        RSSI            수신신호세기[dBm]
        BATT            배터리용량[%]
        SADDR           변경된 자신의 SADDR

[Example]
T>AT+ADDR?<CR>
R<AT+ADDR=FFFF 0 100 FFFF OK<CR>        현재 SADDR 은 FFFF

T>AT+ADDR=1234<CR>
R>AT+ADDR=1234 0 100 1234 OK<CR>        SADDR 을 1234 로 변경
```

10.13 AT+PANID

```
Parameter: PAN ID
읽기(Get) 지원, 다른 디바이스로 전달/문의(Send/Query) 불가
Role 이 AP 인 디바이스는 변경 불가
```

```
[Get format]
AT+PANID?<CR/LF/CRLF>

[Get Response]
AT+PANID=<SADDR> <RSSI> <BATT> <PANID> OK<LF>
        SADDR           자신의 SADDR
        RSSI            수신신호세기[dBm]
        BATT            배터리용량[%]
        PANID           PAN ID, ALONE 상태에서는 FFFF 반환

[Example]
T>AT+PANID?<CR>
R<AT+PANID=FFFF 0 100 FFFF OK<CR>       Alone 상태의 SADDR 과 PANID 는 FFFF
```

10.14 AT+HOPMODE

```
Hopping mode
쓰기/전달/읽기/문의(Set/Send/Get/Query)지원
```

```
[Set/Send format]
AT+HOPMODE=<DADDR> <HOP><CR/LF/CRLF>
        HOP             1/2/3/4,        Hopping mode

[Set/Send Response format]
```

```
AT+HOPMODE=<DADDR> <RSSI> <BATT> <HOP> OK<LF>

[Get/Query format]
AT+HOPMODE?[DADDR]<CR/LF/CRLF>

[Get/Query Response format]
AT+HOPMODE=<DADDR> <RSSI> <BATT> <HOP> OK<LF>
        HOP             1/2/3/4,        Hopping mode

[Get Example]
T>AT+HOPMODE?<CR>
R<AT+HOPMODE=0000 0 100 2 OK                    #Hopping mode 2(00.00)

[Set Example]
T>AT+HOPMODE=0000 1<CR>                         #Set Hopping mode to 1
R<AT+HOPMODE=0000 0 100 1 OK<CR>
```

10.15 AT+ROLE

```
Device role
쓰기/전달/읽기/문의(Set/Send/Get/Query)지원
[Set/Send format]
AT+ROLE=<DADDR> <ROLE><CR/LF/CRLF>
        ROLE            Device role
                        NA: Nothing, 공장초기화 이후 상태
                        NODE: Node
                        AP: AP

[Set/Send Response format]
AT+ROLE=<DADDR> <RSSI> <BATT> <ROLE> OK<LF>

[Get/Query format]
AT+ROLE?[DADDR]<CR/LF/CRLF>

[Get/Query Response format]
AT+ROLE=<DADDR> <RSSI> <BATT> <ROLE> OK<LF>
        ROLE            Device role NA/NODE/AP

[Get Example]
T>AT+ROLE?<CR>
R<AT+ROLE=0000 0 100 AP OK              #Role AP

[Set Example]
T>AT+ROLE=FFFF AP<CR>                   #Set Role as AP
R<AT+ROLE=0000 0 100 AP OK<CR>          #Role 을 AP 로 설정하면 자동으로 주소가 0000 으로 바뀐다.
```

10.16 AT+OPMODE

```
Operation mode
읽기/문의(Get/Query)지원
```

```
[Get/Query format]
AT+OPMODE?[DADDR]<CR/LF/CRLF>

[Get/Query Response format]
AT+ROLE=<DADDR> <RSSI> <BATT> <OPMODE> OK<LF>
        OPMODE          Operation mode
                        0: Init mode(Alone state)
                        1: Normal mode(Join state)
                        2: Sleep mode
                        4: Joinable mode
[Get Example]
T>AT+OPMODE?<CR>
R<AT+OPMODE=0000 0 100 1 OK                     #Normal mode
```

10.17 AT+PHY

```
Config PHY mode
쓰기/전달/읽기/문의(Set/Send/Get/Query)지원
```

```
[Set/Send format]
AT+PHY=<DADDR> <PHY><CR/LF/CRLF>
        PHY             PHY mode 0/1
                        L: Long range mode, FSK
                        H: High data rate mode, MSK 80kbps


[Set/Send Response format]
AT+PHY=<DADDR> <RSSI> <BATT> <PHY> OK<LF>

[Get/Query format]
AT+PHY?[DADDR]<CR/LF/CRLF>

[Get/Query Response format]
AT+PHY=<DADDR> <RSSI> <BATT> <PHY> OK<LF>

[Get Example]
T>AT+PHY?<CR>
R<AT+PHY=0000 0 100 H OK

[Set Example]
T>AT+PHY=0000 L<CR>                     #Set PHY to Long range mode
R<AT+PHY=0000 0 100 L OK<CR>
```

10.18 AT+JOINSTS

Network Status Parameter
읽기/문의(Get/Query) 지원, 쓰기/전달(Set/Send) 불가

```
[Get/Query format]
AT+JOINSTS?[DADDR]<CR/LF/CRLF>

[Get Response]
AT+JOINSTS=<DADDR> <RSSI> <BATT> <JOINSTS> OK<LF>
        DADDR           목적지 주소
        RSSI            수신신호세기[dBm]
        BATT            배터리용량[%]
        JOINSTS         Network Status,
                        A       Alone
                        J       Join

[Example]
T>AT+JOINSTS?<CR>
R<AT+ JOINSTS =FFFF 0 100 A OK<CR>              Alone 상태의 SADDR 과 PANID 는 FFFF
```

10.19 AT+PARAMS

기본 Parameters 정보: RFCH, TXPWR, PHY, JOINSTS, PANID, HOPMODE
읽기/다른 디바이스로 문의(Get/Query) 지원

```
[Get/Query format]
AT+PARAMS?[DADDR]<CR/LF/CRLF>

[Get/Query Response]
AT+PARAMS=<DADDR> <RSSI> <BATT> <RFCH> <TXPWR> <PHY> <JOINSTS> <PANID> <HOPMODE> OK<LF>
        DADDR           목적지 주소
        RSSI            수신신호세기[dBm]
        BATT            배터리용량[%]
        RFCH            RF Channel number(1 ~ 21)
        TXPWR           Tx Power, 출력 파워(0 ~ 210, 0.1dBm 단위, 210 = 21.0 dBm)
        PHY             PHY MODE <0/1>
                        L: Long range mode (2FSK, 10kbps)
                        H: High data rate mode (MSK, 80kbps)
        JOINSTS         Network Status, A: Alone, J: Join
        PANID           PAN ID, ALONE 상태에서는 FFFF 반환
        HOPMODE         Hopping Mode, 1:(0000), 2:(00.00), 3:(00.0.0), 4:(0.0.0.0)

[Example]
T>AT+PARAMS?0100<CR>
R<AT+PARAMS=0100 -65 100 15 210 H J E1C1 2 OK<CR>
        0100            SADDR
        -65             RSSI, -65dBm
        100             BATT, 100%
```

```
        15              RFCH #15
        210             TXPWR 21.0dBm
        H               PHY mode: High data rate mode(MSK, 80kbps)
        J               Join status: Join state(1)
        E1C1            PAN ID: E1C1
        2               Hopping mode 2(00.00)
```

10.20 AT+JOINREQ

JOIN 요청 메시지
상위 디바이스에서 자신의 하위 Node 를 네트워크에 가입시키기 위해 송신하는 Broadcasting 메시지
Alone 상태의 디바이스가 이 메시지를 수신하면 Joinable 상태로 상태를 변경한다.
실행/전달(Execute/Send)지원

[Execute/Send format]
AT+JOINREQ=<DADDR> [<RFCH>]<CR/LF/CRLF>
        RFCH   JOINREQ 메시지를 송신하려는 RF CH
                        생략할 경우, 모든 RF CH 에 대해 송신한다.

[Execute/Send Response format]
AT+JOINREQ=<DADDR> <RSSI> <BATT> OK<LF>

[Execute/Send Example]
모든 RF CH 로 메시지 송신
T>AT+JOINREQ=0<LF>
R<AT+JOINREQ=0000 0 100 OK<LF>

10.21 AT+JOINRSP

JOIN 응답 메시지
JOINREQ 를 받은 디바이스에서 네트워크 가입을 위해 JOINREQ 를 송신한 디바이스로 보내는 응답 메시지
Alone 상태에서 JOINREQ 를 받고 1 분 이내에 응답해야 한다.
실행(Execute)지원

[Execute format]
AT+JOINRSP=<DADDR><CR/LF/CRLF>
        DADDR   FFFF, 네트워크 Join 이전이기에 항상 FFFF 로 송신한다.

[Execute Response format]
AT+JOINRSP=<DADDR> <RSSI> <BATT> OK<LF>

[Execute/Send Example]
T>AT+JOINRSP=FFFF<LF>
R<AT+JOINRSP=FFFF 0 100 OK<LF>

10.22 AT+PJOIN

JOIN 허가(permit) 메시지
JOINREQ 에 대한 응답인 JOINRSP 를 송신한 디바이스를 네트워크에 가입(Join)시키는 메시지
수신한 JOINRSP 에 포함된 네트워크 가입을 원하는 디바이스의 Long Address 를 이용한다.
JOINRSP 를 받고 1 분 이내에 응답해야 한다.
실행(Execute)지원

[Execute format]
AT+PJOIN=<DADDR> <LADDR><CR/LF/CRLF>
LADDR Join 을 시키려는 디바이스의 Long Address

[Execute Response format]
AT+PJOIN=<DADDR> <RSSI> <BATT> OK<LF>

[Execute/Send Example]
R<AT+JOINRSP=FFFF 0012ABCD12345678 OK<LF> #수신한 JOINRSP 에 포함된 LADDR 을 확인한다.
T>AT+PJOIN=0 0012ABCD12345678<LF>
R<AT+PJOIN=0 0 100 OK<LF>

10.23 AT+EXIT

Elicit 네트워크에서 등록 해제
AP 에서 특정 Node 의 연결을 해제할 때 사용한다.
이 명령을 수신한 Node 는 IND(EXIT)메시지를 AP 로 전송하고 자신의 네트워크 정보를 초기화한다.
실행/전달(Execute/Send)지원

[Execute format]
AT+EXIT=<DADDR> <CR/LF/CRLF>
DADDR 네트워크 등록을 해제하려는 디바이스의 주소

[Execute Response format]
AT+EXIT=<DADDR> <RSSI> <BATT> OK<LF>

[Execute/Send Example]
T>AT+EXIT=0100<LF>
R<AT+EXIT=0 0 100 OK<LF>
R<AT+IND=0100 0 100 5 OK<LF> # Node 0100 이 네트워크에서 Exit

10.24 AT+ASADDR

하위 Node 를 등록한 카운트
신규 Node 를 등록(Join)시킬 때 사용하는 카운트, 이 값으로 주소를 할당한다.
쓰기/전달/읽기/문의(Set/Send/Get/Query)지원

[Set/Send format]
AT+ASADDR=<DADDR> <ASCNT><CR/LF/CRLF>
ASCNT 0~ADD_MAX, Associate counter, ADD_MAX 는 Hopping mode 에 따라 다름

[Set/Send Response format]

```
AT+ASADDR=<DADDR> <RSSI> <BATT> <ASCNT> OK<LF>

[Get/Query format]
AT+ASADDR?[DADDR]<CR/LF/CRLF>

[Get/Query Response format]
AT+ASADDR=<DADDR> <RSSI> <BATT> <ASCNT> OK<LF>

[Get Example]
T>AT+ASADDR?<LF>
R<AT+ASADDR=0000 0 100 8 OK<LF>          #지금까지 할당한 Node 의 개수는 8(첫 번째 하위 노드만 해당)

[Set Example]
T>AT+ASADDR=0000 7<LF>                   #Count 를 7 로 변경, 다음 신규 Node 를 할당할 때 주소는 8
R<AT+ASADDR=0000 0 100 7 OK<CR>
```

10.25 AT+ECHO

```
UART1 local echo back mode
공장초기화(Init level2) 이후에는 Disable 된다.
쓰기/전달/읽기/문의(Set/Send/Get/Query)지원
```

```
[Set/Send format]
AT+ECHO=<DADDR> <ENABLE><CR/LF/CRLF>
        ENABLE          Enable/disable the local Echo mode
                        0: Disable
                        1: Enable

[Set/Send Response format]
AT+ECHO=<DADDR> <RSSI> <BATT> <EN> OK<LF>

[Get/Query format]
AT+ECHO?[DADDR]<CR/LF/CRLF>

[Get/Query Response format]
AT+ECHO=<DADDR> <RSSI> <BATT> <EN> OK<LF>

[Get Example]
T>AT+ECHO?<CR>
R<AT+ECHO=0000 0 100 1 OK

[Set Example]
T>AT+ECHO=0000 1<CR>                     #Enable UART local Echo mode
R<AT+ECHO=0000 0 100 1 OK<CR>
```

10.26 AT+BAUD

Configure UART baud rate
쓰기/전달/읽기/문의(Set/Send/Get/Query)지원

[Set/Send format]
AT+BAUD=<DADDR> <PORT> <BAUD><CR/LF/CRLF>
PORT 1/2, UART Port
1: UART1
2: UART2
BOUD 0~4, Baud rate (bps)
0: 115200
1: 57600
2: 38400
3: 19200
4: 9600

[Set/Send Response format]
AT+BAUD=<DADDR> <RSSI> <BATT> <PORT> <BAUD> OK<LF>

[Get/Query format]
AT+BAUD?[DADDR]<CR/LF/CRLF>

[Get/Query Response format]
AT+BAUD=<DADDR> <RSSI> <BATT> <U1_BOUD> <U2_BOUD>OK<LF>
U1_BAUD 0~4, Baud rate of UART1
U2_BAUD 0~4, Baud rate of UART2

[Get Example]
T>AT+BAUD?<CR>
R<AT+BAUD=0000 0 100 0 4 OK #UART1: 115200, UART2: 9600

[Set Example]
T>AT+BAUD=0000 2 2<CR>
R<AT+BAUD=0000 0 100 2 2 OK<CR> #Set UART2 baud rate to 38400

10.27 AT+SEND

사용자 데이터(User data)를 지정된 주소의 지정된 port 로 전송하여 지정된 형식으로 출력
목적지 주소는 16bit SADDR 과 64bit LADDR 모두 가능
전달(Send) 지원

[Send format]
AT+SEND=<DADDR> <PORT> <FORMAT> <"USER DATA"><CR/LF/CRLF>
DADDR User data 를 전달할 목적지 주소, SADDR/LADDR 모두 가능
PORT 목적지에서 User data 를 출력할 UART port<1/2>
1: UART1
2: UART2
FORMAT User data 의 출력 형식<S/SC/SL/SCL/B/BM>
S: 문자열 출력(String format), User data 를 그대로 출력한다.
User data 에 공백이 포함될 경우 큰따옴표(“)를 앞뒤로 덧붙인다.

```
                                출력 포트가 UART1 인 경우: AT+SEND 응답으로 출력
                                출력 포트가 UART2 인 경우: 문자열만 출력
                        SC:     출력 포트가 UART2 인 경우 문자열 + CR 출력
                        SL:     출력 포트가 UART2 인 경우 문자열 + LF 출력
                        SCL:    출력 포트가 UART2 인 경우 문자열 + CR + LF 출력
                        B:      Hexadecimal 로 입력된 User data 를 binary 로 출력(Binary format)
                        BM:     Binary 출력 + CRC16(Modbus RTU 용)
                                User data 를 binary 로 변환한 후, 이에 대한 CRC16 을 덧붙여 출력
        ”USER DATA”     전송할 User data, 공백이 포함된 문자열은 큰따옴표(“)를 붙인다.
                        FORMAT 이 B/M 일 경우, 데이터를 hexadecimal 로 공백 없이 입력한다.

[Send Response]
AT+SEND=<OADDR> <RSSI> <BATT> <OK/FAIL><LF>
        OADDR           응답한 발신지 주소
        RSSI            수신신호세기[dBm]
        BATT            배터리용량[%]

[Received device]
SEND 명령을 수신한 디바이스의 출력 형식
AT+SEND=<OADDR> <RSSI> <BATT> <”STRING”> OK<LF>
        OADDR           발신지, 메시지를 보낸 디바이스의 주소
        RSSI            수신신호세기[dBm]
        BATT            배터리용량[%]
        STRING          수신한 문자열 메시지

[Example 1] NODE(0100)의 UART1 으로 문자메시지 전송
T>AT+SEND=0100 1 S “Hello world”<LF>
        0100            DADDR, 목적지 Node 주소
        1               DD(Destination device)의 UART 포트 번호
        S               DD 의 출력 형식: String
R<AT+SEND=0100 -75 100 OK<LF>
        0100            DADDR, 목적지 Node 의 주소
        -75             RSSI, 메시지 전송 후 수신한 ACK 의 신호 세기, -75[dBm]
        100             BATT 용량, ACK 를 보낸 디바이스의 BATT 용량, 100%
        OK              발신지(00.00)에서 명령 전송이 성공(ACK 수신확인)했을 경우, OK 로 응답

[목적지의 UART1 출력 메시지]
R<AT+SEND=0000 -80 100 “Hello world” OK<LF>
        0000            OADDR, 메시지를 보낸 발신지의 주소(AP)
        -80             RSSI, 수신한 메시지의 신호 세기, -80[dBm]
        100             BATT 용량, 메시지를 송신한 디바이스의 BATT 용량, 100%
        OK              정상적으로 메시지를 수신했을 경우, OK 로 응답

[Example 2] NODE(0100)의 UART2 로 Modbus 명령 전송
T>AT+SEND=0100 2 BM 0104000A0001<LF>
        0100            DADDR, 목적지 Node 주소
        2               DD(Destination device)의 UART 포트 번호
        BM              DD 의 출력 형식: Binary + CRC16
```

```
R<AT+SEND=0100 -75 100 OK<LF>

[목적지의 UART2 출력 데이터]
R<(01 04 00 0A 00 01 11 C8)h
        Binary 로 출력, 수신한 데이터에 대한 CRC16 을 추가전송(C811)h
```

10.28 AT+U1SND

UART1 으로 입력되는 모든 데이터를 지정한 디바이스로 자동 전송하는 모드 설정
데이터 입력이 2ms 이상 중단되거나, 연속으로 입력된 크기가 128Byte 이상이 될 때 전송
UART1 자동 전송 모드가 설정된 이후에는 AT 명령을 입력해도 수행하지 않음.
자동 전송 모드 해제는 연속으로 "AT#" 입력하거나, 다른 디바이스에서 <FORMAT> 값을 0 으로 명령 전송
자기 자신의 UART1 으로 출력 설정은 되지 않음
쓰기/전달/읽기/문의(Set/Send/Get/Query)지원

```
[Set/Send format]
AT+U1SND=<DADDR> <U1DADDR> <PORT> <FORMAT><CR/LF/CRLF>
        DADDR           U1SND 명령을 수행할 목적지 주소
        U1DADDR         UART1 로 입력되는 데이터를 자동으로 전달할 목적지 주소
        PORT            데이터를 출력할 UART port<1/2>
                        1:      UART1, 다른 디바이스로 전송 가능(U1DADDR 이 DADDR 과 달라야 함)
                        2:      UART2
        FORMAT          User data 의 출력 형식<0/P/H/B/S>
                        0:      UART1 입력 데이터 자동 전송 중지
                        P:      입력 데이터를 그대로 출력(Pass-through)
                                데이터 변환 없이 그대로 출력(UART1 에서 AT 명령 형식 적용 안함)
                                입력이 문자열이 아닌 Binary data 이면 출력도 Binary 로 출력
                        X:      입력 데이터를 Hexadecimal 문자열로 변환하여 출력
                                UART1 은 AT+U1SND 형식으로 출력, UART2 은 Hexadecimal 만 출력
                                입력("Hello")
                                        UART1 출력(AT+U1SND=0000 0 100 1 X "48656C6C6F")
                                        UART2 출력("48656C6C6F")
                                입력(01100101b)
                                        UART1 출력: AT+U1SND=0000 0 100 1 X "65"
                                        UART2 출력: "65"
                        B:      입력 데이터를 Binary 로 변환하여 출력
                                입력 데이터는 Hexadecimal 문자열로 입력해야 함
                                UART1, UART2 동일하게 Binary 출력
                                입력("0A35")
                                        UART1 출력(0000101000110101)b
                                        UART2 출력(0000101000110101)b
                        S:      입력 데이터를 문자열로 출력(UART1 은 제어문자#3 출력 안함)
                                UART1 은 AT+U1SND 형식으로 출력, UART2 은 문자열 출력
                                입력("Hello")
                                        UART1 출력: AT+U1SND=0000 0 100 1 S "Hello"
                                        UART2 출력: "Hello"
                                입력(01100101b)
```

UART1 출력: AT+U1SND=0000 0 100 1 S “e”
UART2 출력: “e”

[Set/Send Response format]
AT+U1SND=<DADDR> <RSSI> <BATT> <OK/FAIL><LF>
DADDR 목적지 주소
RSSI 수신신호세기[dBm]
BATT 배터리용량[%]

[목적지의 UART1 출력 메시지 format]
AT+U1SND=<OADDR> <RSSI> <BATT> <PORT:1> <FORMAT> <DATA> OK<LF>
OADDR 발신지 주소
RSSI 수신신호세기[dBm]
BATT 메시지를 보낸 디바이스의 배터리용량[%]
PORT 데이터를 출력할 UART port<1>
FORMAT User data 의 출력 형식<0/P/H/S>
DATA 수신한 User data

[Get/Query format]
AT+U1SND?[DADDR]<CR/LF/CRLF>

[Get/Query Response]
AT+U1SND=<OADDR> <RSSI> <BATT> <U1DADDR> <PORT> <FORMAT> OK<LF>
OADDR 응답을 보낸 디바이스 주소
RSSI 수신신호세기[dBm]
BATT 배터리용량[%]
U1DADDR UART1 으로 입력되는 데이터를 자동으로 전달할 목적지 주소
PORT 데이터를 출력할 UART port<1/2>
FORMAT User data 의 출력 형식<0/P/H/S>

[Example]
NODE(0100)의 UART1 에 Modbus 장비를 연결하고,
입력되는 모든 데이터를 AP(0000)로 전송하여 hexadecimal 형식으로 UART Port1 으로 출력하도록 설정
T>AT+U1SND=0100 0 1 X<LF>
0100 DADDR, 명령을 수행할 디바이스 주소(NODE 0100)
0 UART1 데이터 송신 목적지 주소(AP)
1 UART 포트 번호
X 데이터 출력 형식(Hexadecimal)

AP 에서 수신한 NODE(0100)의 UART1 데이터
R<AT+U1SND=0100 -75 100 “010402ABCD0795” OK<LF>
0100 OADDR, 데이터를 보낸 발신지 주소
-75 RSSI, 무선 메시지 수신 신호 세기, -75[dBm]
100 BATT 용량, 메시지를 보낸 디바이스의 BATT 용량, 100%
U1DATA NODE(0100)의 UART2 로 입력된 데이터는 (010402ABCD0795h)

#3 ASCII code(0 ~ 31, 127)

10.29 AT+U2SND

UART2 로 입력되는 모든 데이터를 지정한 디바이스로 자동 전송하는 모드 설정
데이터 입력이 2ms 이상 중단되거나, 연속으로 입력된 크기가 128Byte 이상이 될 때 전송
자기 자신의 UART2 로 출력 설정은 되지 않음
쓰기/전달/읽기/문의(Set/Send/Get/Query)지원

[Set/Send format]
AT+U2SND=<DADDR> <U2DADDR> <PORT> <FORMAT><CR/LF/CRLF>

DADDR		U2SND 명령을 수행할 목적지 주소
U2DADDR		UART2 로 입력되는 데이터를 자동으로 전달할 목적지 주소
PORT		데이터를 출력할 UART port<1/2>
	1:	UART1
	2:	UART2, 다른 디바이스로 전송 가능(U2DADDR 이 DADDR 과 달라야 함)
FORMAT		User data 의 출력 형식<0/P/H/B/S>
	0:	UART2 입력 데이터 자동 전송 중지
	P:	입력 데이터를 그대로 출력(Pass-through)
		데이터 변환 없이 그대로 출력(UART1 에서 AT 명령 형식 적용 안함)
		입력이 문자열이 아닌 Binary data 이면 출력도 Binary 로 출력
	X:	입력 데이터를 Hexadecimal 문자열로 변환하여 출력
		UART1 은 AT+U2SND 형식으로 출력, UART2 은 Hexadecimal 만 출력
		입력(“Hello”)
		UART1 출력(AT+U2SND=0000 0 100 1 X “48656C6C6F”)
		UART2 출력(“48656C6C6F”)
		입력(01100101b)
		UART1 출력: AT+U2SND=0000 0 100 1 X “65”
		UART2 출력: “65”
	B:	입력 데이터를 Binary 로 변환하여 출력
		입력 데이터는 Hexadecimal 문자열로 입력해야 함
		UART1, UART2 동일하게 Binary 출력
		입력(“0A35”)
		UART1 출력(0000101000110101)b
		UART2 출력(0000101000110101)b
	S:	입력 데이터를 문자열로 출력(UART1 으로 제어문자는 출력 안함)
		UART1 은 AT+U2SND 형식으로 출력, UART2 은 문자열 출력
		입력(“Hello<CR>”)
		UART1 출력: AT+U2SND=0000 0 100 1 S “Hello”
		UART2 출력: “Hello<CR>”
		입력(01100101b)
		UART1 출력: AT+U2SND=0000 0 100 1 S “e”
		UART2 출력: “e”

[Set/Send Response format]
AT+U2SND=<DADDR> <RSSI> <BATT> <OK/FAIL><LF>

DADDR	목적지 주소
RSSI	수신신호세기[dBm]
BATT	배터리용량[%]

[목적지의 UART1 출력 메시지 format]

AT+U2SND=<OADDR> <RSSI> <BATT> <PORT:1> <FORMAT> < DATA> OK<LF>
OADDR 발신지 주소
RSSI 수신신호세기[dBm]
BATT 메시지를 보낸 디바이스의 배터리용량[%]
PORT 데이터를 출력할 UART port<1>
FORMAT User data 의 출력 형식<0/P/H/S>
DATA 수신한 USER DATA

[Get/Query format]
AT+U2SND?[DADDR]<CR/LF/CRLF>

[Get/Query Response format]
AT+U2SND=<OADDR> <RSSI> <BATT> <U2DADDR> <PORT> <FORMAT> OK<LF>
OADDR 응답을 보낸 디바이스 주소
RSSI 수신신호세기[dBm]
BATT 배터리용량[%]
U2DADDR UART2 로 입력되는 데이터를 자동으로 전달할 목적지 주소
PORT 데이터를 출력할 UART port<1/2>
FORMAT User data 의 출력 형식<0/P/H/S>

[Example]
NODE(0100)의 UART2 에 IoT 장비의 UART 콘솔을 연결하고,
입력되는 모든 데이터를 AP(0000)로 전송하여 String 형식으로 UART Port2 로 출력하도록 설정
T>AT+U2SND=0100 0 2 S<LF>
0100 DADDR, 명령을 수행할 디바이스 주소
0 데이터를 보낼 목적지 주소(AP)
2 UART 포트 번호
S 데이터 출력 형식(String)

NODE(0100)의 UART2 에 연결된 IoT 장비가 “Door Open!<CR>” 메시지를 출력했을 때
AP 에서 수신한 NODE(0100)의 데이터 출력 – UART2 로 출력
R<“Door Open!<CR>”

10.30 AT+RFCH

Parameter: RF channel
쓰기/전달/읽기/문의(Set/Send/Get/Query)지원

[Set/Send format]
AT+RFCH=<DADDR> <RFCH><CR/LF/CRLF>
RFCH RF Channel number, 1 ~ 21

[Set/Send Response]
AT+RFCH=<DADDR> <RSSI> <BATT> <RFCH> OK<LF>
DADDR 목적지 SADDR
RSSI 수신신호세기[dBm]
BATT 배터리용량[%]
RFCH RF Channel number, 1 ~ 21

```
[Get/Query format]
AT+RFCH?[DADDR]<CR/LF/CRLF>

[Get/Query Response]
AT+RFCH=<OADDR> <RSSI> <BATT> <RFCH> OK<LF>
        OADDR           응답을 보낸 디바이스 주소
        RSSI            수신신호세기[dBm]
        BATT            배터리용량[%]
        RFCH            RF Channel number, 1 ~ 21

[Get Example]
T>AT+RFCH?<CR>
R<AT+RFCH=0000 0 100 15 OK<CR>
        15              RFCH #15

[Set Example]
T>AT+RFCH=0 3<CR>
R<AT+RFCH=0000 0 100 3 OK<CR>
```

10.31 AT+TXPWR

```
Parameter: RF Transmit Power
쓰기/전달/읽기/문의(Set/Send/Get/Query)지원
[Set/Send format]
AT+TXPWR=<DADDR> <PWR><CR/LF/CRLF>
        PWR             RF Tx Power in 0.1 dBm, 200 means 20.0dBm, 0 ~ 210

[Set/Send Response]
AT+TXPWR=<DADDR> <RSSI> <BATT> <PWR> OK<LF>
        DADDR           목적지 SADDR
        RSSI            수신신호세기[dBm]
        BATT            배터리용량[%]
        PWR             RF Tx Power in 0.1 dBm, 200 means 20.0dBm, 0 ~ 210

[Get/Query format]
AT+TXPWR?<DADDR><CR/LF/CRLF>

[Get/Query Response]
AT+TXPWR=<OADDR> <RSSI> <BATT> <PWR> OK<LF>
        OADDR           응답을 보낸 디바이스 주소
        RSSI            수신신호세기[dBm]
        BATT            배터리용량[%]
        PWR             RF Tx Power in 0.1 dBm, 200 means 20.0dBm

[Get Example]
T>AT+TXPWR?<CR>
```

```
R<AT+TXPWR=0000 0 100 210 OK<CR>
        210             Tx Power is 21.0dBm

[Set Example]
T>AT+TXPWR=0 100<CR>
R<AT+TXPWR=0000 0 100 10 OK<CR>
```

10.32 AT+TONE

```
Transmit test carrier(for RF test)
실행/읽기(Execute/Get)지원, 읽기는 현재 TONE test option 값을 반환
Test 목적으로 RF 신호를 끊임없이 출력한다. RF channel 을 점유하기 때문에 Test 용도로만 사용해야 한다.
[Execute format]
AT+TONE=<DADDR> <OPTION><CR/LF/CRLF>
        DADDR           목적지 주소는 자기 SADDR 만 허용
        OPTION          0: Stop
                        1: Transmit un-modulated carrier
                        2: Transmit modulated carrier(random data)

[Execute Response]
AT+TONE=<DADDR> <RSSI> <BATT> OK<LF>
        DADDR           DADDR
        RSSI            수신신호세기[dBm]
        BATT            배터리용량[%]

[Get format]
AT+TONE?[DADDR]<CR/LF/CRLF>

[Get/Query Response]
AT+TXPWR=<DADDR> <RSSI> <BATT> <OPTION> OK<LF>
        DADDR           목적지 주소는 자기 SADDR 만 허용
        RSSI            수신신호세기[dBm]
        BATT            배터리용량[%]
        OPTION          0: Stop, 1: Un-modulated carrier, 2: Modulated carrier(random data)

[Get Example]
T>AT+TONE?<CR>
R<AT+TONE=0000 0 100 0 OK<CR>
        0       Stop Tx tone

[Execute Example]
T>AT+TONE=0 1<CR>
R<AT+TONE=0000 0 100 OK<CR>
```

10.33 AT+BTN

```
Configure Button mode
Button1, Button2 공통 적용
쓰기/전달/읽기/문의(Set/Send/Get/Query)지원
```

```
[Set/Send format]
AT+BTN=<DADDR> <MODE><CR/LF/CRLF>
        MODE            Button Mode
                        11      Input with Pull-up
                        12      Input with Pull-down

[Set/Send Response]
AT+BTN=<DADDR> <RSSI> <BATT> <MODE> OK<LF>

[Get/Query format]
AT+BTN?[DADDR]<CR/LF/CRLF>

[Get/Query Response]
AT+BTN=<OADDR> <RSSI> <BATT> <MODE> OK<LF>

[Get Example]
T>AT+BTN?<CR>
R<AT+BTN=0000 0 100 11 OK<CR>           # Button mode is Input with pull-up

[Set Example]
T>AT+BTN=0 12<CR>
R<AT+BTN=0000 0 100 12 OK<CR>
```

10.34 AT+LED

```
Configure LED mode
LED1, LED2 공통 적용
쓰기/전달/읽기/문의(Set/Send/Get/Query)지원
```

```
[Set/Send format]
AT+LED=<DADDR> <MODE><CR/LF/CRLF>
        MODE            LED Mode
                        0       LED Off
                        20      Push-pull output
                        30      Open-drain output
                        40      Open-source output

[Set/Send Response]
AT+LED=<DADDR> <RSSI> <BATT> <MODE> OK<LF>

[Get/Query format]
AT+LED?[DADDR]<CR/LF/CRLF>

[Get/Query Response]
AT+LED=<OADDR> <RSSI> <BATT> <MODE> OK<LF>
```

```
[Get Example]
T>AT+LED?<CR>
R<AT+LED=0000 0 100 20 OK<CR>          # LED mode is push-pull out

[Set Example]
T>AT+LED=0 30<CR>                      #Configure LED mode to open-drain
R<AT+LED=0000 0 100 30 OK<CR>
```

10.35 AT+IO

Configure IO (GPIO and AUX)
AUX 는 Input(pull-up/down)만 설정 가능, 출력 설정 불가능.
GPIO1~8 모두 입력/출력 설정 가능, 입력 이벤트 설정은 GPIO4~8, AUX 만 가능
쓰기/전달/읽기/문의(Set/Send/Get/Query)지원

[Set/Send format]
GPIO4~8, AUX 는 Digital 입력에 대한 Event trigger 를 설정할 수 있다.
GPIO1,2,3 은 Event trigger 를 설정할 수 없다.

```
AT+IO=<DADDR> <IO> <MODE> <TRIGGER><CR/LF/CRLF>
        IO              IO number
                        1 ~ 8   GPIO1 ~ GPIO8
                        9       AUX
        MODE            IO Mode
                        0       Input/Output Disable
                        10      Input
                        11      Input with Pull-up
                        12      Input with Pull-down
                        20      Push-pull output
                        30      Open-drain output
                        31      Open-drain output with Pull-up
                        40      Open-source output
                        41      Open-source output with Pull-down
        TRIGGER         Event trigger
                        0       Disable Event
                        1       Positive edge
                        2       Negative edge
                        3       Both edge

[Set/Send Response format]
AT+IO=<DADDR> <RSSI> <BATT> <IO> <MODE> <TRIGGER>OK<LF>

[Get/Query format]
AT+IO?[<DADDR> <IO>]<CR/LF/CRLF>

[Get/Query Response format]
```

<IO>를 생략하면 IO1 ~ 9 까지 모두 출력하고, IO 번호를 지정하면 지정한 IO 에 대한 결과만 출력한다.

```
AT+IO=<OADDR> <RSSI> <BATT> <IO> <MODE> <TRIGGER> OK<LF>
```

```
[Get Example]
T>AT+IO?<CR>
T>AT+IO=0000 0 100 1 11 0 OK            #GPIO#1: Input with pull-up, disable event
T>AT+IO=0000 0 100 2 12 0 OK            #GPIO#2: Input with pull-down, disable event
T>AT+IO=0000 0 100 3 00 0 OK            #GPIO#3: IO disabled, disable event
T>AT+IO=0000 0 100 4 20 0 OK            #GPIO#4: Push-pull output, disable event
T>AT+IO=0000 0 100 5 30 0 OK            #GPIO#5: Open-drain output, disable event
T>AT+IO=0000 0 100 6 31 0 OK            #GPIO#6: Open-drain output with pull-up, disable event
T>AT+IO=0000 0 100 7 40 0 OK            #GPIO#7: Open-source output, disable event
T>AT+IO=0000 0 100 8 11 2 OK            #GPIO#8: Input with pull-up, Negative edge triggered event
T>AT+IO=0000 0 100 9 11 1 OK            #AUX: Input with pull-up, Positive edge triggered event

[Set Example]
T>AT+IO=0 8 0 0<CR>                     #GPIO#8: IO disabled, disable event
R<AT+IO=0000 0 100 8 0 0 OK<CR>
```

10.36 AT+AUX

Read DI value of AUX
읽기/문의(Get/Query)지원

```
[Get/Query format]
AT+AUX?[<DADDR>]<CR/LF/CRLF>

[Get/Query Response format]
AT+AUX=<OADDR> <RSSI> <BATT> <DIN>OK<LF>
        DIN             DIN value of AUX
                        0       Logic low(DIN < 0.3*VCC)
                        1       Logic high(DIN > 0.7*VCC)
                        2       This GPIO is not an input

[Get Example]
T>AT+AUX?<CR>
T>AT+AUX=0000 0 100 1 OK        # DI value of AUX is 1
```

10.37 AT+DI

Read DI value of GPIO
읽기/문의(Get/Query)지원

```
[Get/Query format]
AT+DI?[<DADDR> <GPIO>]<CR/LF/CRLF>
        GPIO            GPIO Number, 1 ~ 8

[Get/Query Response format]
<GPIO>를 생략하면 GPIO1 ~ 8 까지 모두 출력하고, GPIO 값을 지정하면 지정한 GPIO 에 대한 결과만 출력한다.
AT+DI=<OADDR> <RSSI> <BATT> [<DIN1> <DIN2> <DIN3> <DIN4> <DIN5> <DIN6> <DIN7> <DIN8>] OK<LF>
```

```
        DINx            DIN value(0/1/2) of GPIO Number x
                        0       Logic low(DIN < 0.3*VCC)
                        1       Logic high(DIN > 0.7*VCC)
                        2       This GPIO is not an input

[Get Example]
T>AT+DI?<CR>
T>AT+DI=0000 0 100 0 0 1 1 2 2 2 2 OK   #GPIO#1,2: 0, GPIO#3,4: 1, GPIO#5,6,7,8: Not a DI

T>AT+DI?0 3<CR>
T>AT+DI=0000 0 100 1 OK                 #GPIO#3: 1
```

10.38 AT+DO

```
Set and Get DO value of GPIO
쓰기/전달/읽기/문의(Set/Send/Get/Query)지원
[Set/Send format]
AT+DO=<DADDR> <GPIO> <DOUT> [<TGPERIOD>]<CR/LF/CRLF>
        GPIO            GPIO Number, 1 ~ 8
        DOUT            0/1     DOUT value, Available in Output mode
        TGPERIOD        0~6000  Toggle Period in 100ms, 1=100ms, 6000=600000ms=600sec=10min
                        0       Toggle disable

[Set/Send Response format]
AT+DO=<DADDR> <RSSI> <BATT> <GPIO> <DOUT> <TGPERIOD> OK<LF>

[Get/Query format]
AT+DO?[<DADDR> <GPIO>]<CR/LF/CRLF>

[Get/Query Response format]
<GPIO>를 생략하면 GPIO1 ~ 8 까지 모두 출력하고, GPIO 값을 지정하면 지정한 GPIO 에 대한 결과만 출력한다.
AT+DO=<OADDR> <RSSI> <BATT> <IO> <DOUT> <TGPERIOD> OK<LF>
        DOUTx           DOUT value(0/1/2) of GPIO Number x
                        0       Logic low
                        1       Logic high
                        2       This IO is not an output

[Get Example]
T>AT+DO?0 1<LF>
R<AT+DO=0000 0 100 1 1 0 OK     #GPIO#1 DOUT value is 1, Toggle disabled
T>AT+DO?<LF>
R<AT+DO=0000 0 100 1 1 0 OK     #GPIO#1 DOUT value is 1, Toggle disabled
R<AT+DO=0000 0 100 2 1 0 OK     #GPIO#2 DOUT value is 1, Toggle disabled
R<AT+DO=0000 0 100 3 2 0 OK     #GPIO#3 is not a Output mode, Toggle disabled
R<AT+DO=0000 0 100 4 0 0 OK     #GPIO#4 DOUT value is 0, Toggle disabled
R<AT+DO=0000 0 100 5 1 3 OK     #GPIO#5 DOUT value is 1, Toggle DOUT every 300ms
R<AT+DO=0000 0 100 6 1 3 OK     #GPIO#6 DOUT value is 1, Toggle DOUT every 300ms
```

```
R<AT+DO=0000 0 100 7 2 0 OK     #GPIO#7 is not a Output mode, Toggle disabled
R<AT+DO=0000 0 100 8 2 0 OK     #GPIO#8 is not a Output mode, Toggle disabled

[Set Example]
T>AT+DO=0 6 0 0<LF>             #GPIO#6 DOUT value is 1, Toggle disabled
R<AT+DO=0000 0 100 6 0 0 OK
```

10.39 AT+VCC

```
Read ADC and VCC value
읽기/문의(Get/Query)지원
[Get/Query format]
AT+VCC?[DADDR]<CR/LF/CRLF>

[Get/Query Response format]
AT+VCC=<OADDR> <RSSI> <BATT> <VCC> OK<LF>
        VCC             VCC voltage in mV

[Get/Query Example]
T>AT+VCC?0100<CR>
R<AT+VCC=0100 -65 100 6 3319 OK         #VCC: 3319mV
```

10.40 AT+ADC

```
Read ADC value
읽기/문의(Get/Query)지원
[Get/Query format]
AT+ADC?[DADDR]<CR/LF/CRLF>

ADC 결과는 IND 형식으로 반환한다.
[Indication of Get/Query command]
AT+IND=<OADDR> <RSSI> <BATT> <PARAMS : ADC> <ADC> OK<LF>
        ADC             ADC value(unit: mV)

[Get/Query Example]
T>AT+ADC?0100<CR>
R<AT+IND=0100 -65 100 ADC 1823 OK               #ADC: 1823mV
```

10.41 AT+VDAC

```
Set and Get the VDAC value
쓰기/전달/읽기/문의(Set/Send/Get/Query)지원
[Set/Send format]
AT+VDAC=<DADDR> <VDAC><CR/LF/CRLF>
```

```
        VDAC            VDAC value in mV, Do not exceed VCC

[Set/Send Response format]
AT+VDAC=<DADDR> <RSSI> <BATT> OK<LF>

[Get/Query format]
AT+VDAC?[DADDR]<CR/LF/CRLF>

[Get/Query Response format]
AT+VDAC=<OADDR> <RSSI> <BATT> <VDAC> OK<LF>
        VDAC            VDAC value in mV

[Get Example]
T>AT+VDAC?<CR>
R<AT+VDAC=0000 0 100 1234 OK            #Current VDAC value: 1234mV

[Set Example]
T>AT+VDAC=0100 1823<CR>                 #Set VDAC to 1823mV
R<AT+VDAC=0100 -70 100 OK<CR>
```

10.42 AT+I2C

```
Read and write the I2C device(Master only)
하나의 명령으로 Read 는 최대 16byte, Write 는 최대 4byte 까지 가능
I2C 의 Read 결과는 IND(Indication)으로 반환한다.
쓰기/전달/읽기/문의(Set/Send/Get/Query)지원
```

```
[Set/Send format – I2C Write]
AT+I2C=<DADDR> <I2C_ADDR> <TYPE> <REG> <LEN> [<DATA1> <DATA2> <DATA3> <DATA4>] <CR/LF/CRLF>
        I2C_ADDR        Address of I2C device(slave address, 7bit)
        TYPE            read/write flag(write : 1 read : 2)
        REG             Register address(sub address, 8bit)
        LEN             1~4     Data length to write
        DATAX           Data to write

[Set/Send Response format]
AT+I2C=<DADDR> <RSSI> <BATT> OK<LF>

[Get/Query format – I2C Read]
AT+I2C=<DADDR> <I2C_ADDR>> <TYPE> <REG> <LEN> <CR/LF/CRLF>
        I2C_ADDR        Address of I2C device(slave address, 7bit)
        TYPE            read/write flag(write : 1 read : 2)
        REG             Register address(sub address, 8bit)
        LEN             1~16    Data length to read

[Get/Query Response format]
AT+IND=<OADDR> <RSSI> <BATT> <I2C> [<DATA1> … <DATA16>] OK<LF>
```

```
[Get Example]
T>AT+I2C=0100 0x77 2 0x18 2<CR>         #Device Address 0x77, register address 0x18, read 2bytes
R<AT+IND=0100 0 100 I2C 12 34 OK                #Indication – I2C event, data: 12, 34

[Set Example]
T>AT+I2C=0100 0x77 1 0x10 1 0x3F <CR>   #Device Address 0x77, register 0x10, Write 1 byte data(0x3F)
R<AT+I2C=0100 -70 100 OK<CR>            #Success - I2C write
```

10.43 AT+TIME

RTC Time
쓰기/전달/읽기/문의(Set/Send/Get/Query)지원

```
[Set/Send format]
AT+TIME=<DADDR> <EPOCH><CR/LF/CRLF>
        EPOCH           Unix time

[Set/Send Response format]
AT+TIME=<DADDR> <RSSI> <BATT> OK<LF>

[Get/Query format]
AT+TIME?[DADDR]<CR/LF/CRLF>


[Get/Query Response format]
AT+TIME=<OADDR> <RSSI> <BATT> <"DATE"> <EPOCH> OK<LF>
        DATE            YYYY-MM-DD H:M:S 형식의 문자열
        EPOCH           Unix time

[Get Example]
T>AT+TIME?<CR>
R<AT+TIME=0000 0 100 "2023-02-01 20:05:10" 1675249510 OK

[Set Example]
T>AT+TIME=0100 1675249510<CR>                   #Set Time to "2023-02-01 20:05:10"
R<AT+TIME=0100 -70 100 OK<CR>
```

10.44 AT+XTAL

Internal Crystal tuning – Do not change the value
쓰기/전달/읽기/문의(Set/Send/Get/Query)지원

```
[Set/Send format]
※ 내부 Crystal 의 주파수를 미세 조정할 수 있는 명령, 임의로 변경하면 성능에 상당한 영향을 줄 수 있음.
AT+XTAL=<DADDR> <TUNE> <DELTA><CR/LF/CRLF>
        TUNE            X-tal Tune value
        DELTA           Delta of TUNE
```

```
[Set/Send Response format]
AT+XTAL=<DADDR> <RSSI> <BATT> <TUNE> <DELTA> OK<LF>

[Get/Query format]
AT+XTAL?[DADDR]<CR/LF/CRLF>

[Get/Query Response format]
AT+XTAL=<OADDR> <RSSI> <BATT> <TUNE> <DELTA> OK<LF>

[Get Example]
T>AT+XTAL?<LF>
R<AT+XTAL=0000 0 100 70 40 OK<LF>

[Set Example]
T>AT+XTAL=0000 70 40<LF>                          #Do not change
R<AT+XTAL=0000 0 100 70 40 OK<LF>
```

10.45 AT+DIEVT

Digital Input linked event, 8 개 slot 마다 개별 IO 지정 및 명령 설정
동일한 IO 에 대해 복수의 명령 설정 가능
쓰기/전달/읽기/문의(Set/Send/Get/Query)지원

```
[Set/Send format]
AT+DIEVT=<DADDR> <SLOT> <IO> <ACTIVATE> <(CMD)><CR/LF/CRLF>
        SLOT            1~8     Slot index, AT command 를 저장할 수 있는 공간
        IO              4~8     GPIO4 ~ GPIO8
                        9       AUX
        ACTIVATE        0/1     Slot index 에 저장된 AT command 의 실행 여부
                        0       Disable
                        1       Enable
        CMD             Deep sleep 에서 깨어난 후 수행할 AT command

[Set/Send Response]
AT+DIEVT=<DADDR> <RSSI> <BATT> <SLOT> <IO> <ACTIVATE> <(CMD)>OK<LF>

[Get/Query format]
AT+DIEVT?[<DADDR> <SLOT>]<CR/LF/CRLF>
     SLOT               1~8, Slot index

[Get/Query Response]
GET/QUERY 명령에서 SLOT Option 을 생략하면 Slot index 를 1 ~ 8 까지 모두 출력
AT+DIEVT=<OADDR> <RSSI> <BATT> <SLOT> <IO> <ACTIVATE> <(CMD)> OK<LF>

[Get Example]
T>AT+DIEVT?100 1<LF>
R<AT+DIEVT=0100 0 100 1 4 1 (SEND=0 1 S “Door Open”) OK<LF>
```

```
T>AT+DIEVT?100<LF>
R<AT+DIEVT=0100 0 100 1 4 1 (SEND=0 1 S “Door Open”) OK<LF>
R<AT+DIEVT=0100 0 100 2 4 1 (DO=100 1 1 0) OK<LF>
R<AT+DIEVT=0100 0 100 3 5 1 (SEND=0 1 S “Door Close”) OK<LF>
R<AT+DIEVT=0100 0 100 4 5 1 (DO=100 1 0 0) OK<LF>
R<AT+DIEVT=0100 0 100 5 6 1 (SEND=0 1 S “Window Open”) OK<LF>
R<AT+DIEVT=0100 0 100 6 6 1 (DO=100 2 1 0) OK<LF>
R<AT+DIEVT=0100 0 100 7 7 1 (SEND=0 1 S “Window Close”) OK<LF>
R<AT+DIEVT=0100 0 100 8 7 1 (DO=100 2 0 0) OK<LF>

[Set Example]
T>AT+DIEVT=100 4 3 1 (DO=100 2 0)<LF>
R<AT+DIEVT=0100 0 100 4 3 1 (DO=100 2 0)OK<LF>
```

10.46 AT+BTEVT

```
Button linked event, 4 개 slot 마다 개별 명령 설정
동일한 Button 에 대해 복수의 명령 설정 가능
쓰기/전달/읽기/문의(Set/Send/Get/Query)지원
```

```
[Set/Send format]
AT+BTEVT=<DADDR> <SLOT> <BTN> <ACTIVATE> <(CMD)><CR/LF/CRLF>
        SLOT            1~4, Slot index, AT command 를 저장할 수 있는 공간
        IO              1: Button1, 2: Button2
        ACTIVATE        0/1, Slot index 에 저장된 AT command 의 실행 여부
                        0: Disable, 1: Enable
        CMD             Button 이 눌렸을 때 수행할 AT command

[Set/Send Response]
AT+BTEVT=<DADDR> <RSSI> <BATT> <SLOT> <BTN> <ACTIVATE> <(CMD)>OK<LF>

[Get/Query format]
AT+BTEVT?[<DADDR> <SLOT>]<CR/LF/CRLF>
     SLOT               1~4, Slot index

[Get/Query Response]
GET/QUERY 명령에서 SLOT Option 을 생략하면 Slot index 를 1 ~ 4 까지 모두 출력
AT+BTEVT=<OADDR> <RSSI> <BATT> <SLOT> <BTN> <ACTIVATE> <(CMD)> OK<LF>

[Get Example]
T>AT+BTEVT?100 1<LF>
R<AT+BTEVT=0100 0 100 1 1 1 (SEND=0 1 S “Button1 Pressed”) OK<LF>

T>AT+BTEVT?100<LF>
R<AT+BTEVT=0100 0 100 1 1 1 (SEND=0 1 S “Button1 Pressed”) OK<LF>
R<AT+BTEVT=0100 0 100 2 1 1 (DO=100 3 1 0) OK<LF>
R<AT+BTEVT=0100 0 100 3 2 1 (SEND=0 1 S “Button2 Pressed”) OK<LF>
R<AT+BTEVT=0100 0 100 4 2 1 (DO=100 3 0 0) OK<LF>
```

```
[Set Example]
T>AT+BTEVT=100 3 2 0 <LF>                        # De-activate Button2 event stored in slot3
R<AT+BTEVT=0100 0 100 3 2 0 () OK<LF>
```

10.47 AT+PTEVT

Periodic timer event, 8 개 slot 마다 개별 Timer 및 명령 설정
쓰기/전달/읽기/문의(Set/Send/Get/Query)지원

```
[Set/Send format]
AT+PTEVT=<DADDR> <SLOT> <PERIOD> <OFFSET> <(CMD)><CR/LF/CRLF>
        SLOT            1~8, Slot index, AT command 를 저장할 수 있는 공간
        PERIOD          0~2,073,600 [sec], Event period, 최대 24 일, 0 인 Slot 은 수행하지 않음
        OFFSET          0~2,073,600 [sec], Event 수행 Offset, 내부 RTC 의 초를 기준으로 설정
                        Event 발생 시점은 (RTC + OFFSET) % PERIOD 로 계산된다
        CMD             PERIOD 마다 수행할 AT command

[Set/Send Response]
AT+PTEVT=<DADDR> <RSSI> <BATT> OK<LF>

[Get/Query format]
AT+PTEVT?[<DADDR> <SLOT>]<CR/LF/CRLF>
        SLOT            1~8, Slot index

[Get/Query Response]
GET/QUERY 명령에서 Slot index Option 을 생략하면 Slot index 를 1 ~ 8 까지 모두 출력
AT+PTEVT=<OADDR> <RSSI> <BATT> <SLOT> <PERIOD> <(CMD)> OK<LF>

[Get Example]
T>AT+PTEVT?0 1<LF>
R<AT+PTEVT=0000 0 100 1 10 0 (ADC?100) OK<LF>

T>AT+PTEVT?<LF>
R<AT+PTEVT=0000 0 100 1 10 0 (ADC?100) OK<LF>
R<AT+PTEVT=0000 0 100 2 30 2 (SEND=0100 2 BM 0104000A0001) OK<LF>
R<AT+PTEVT=0000 0 100 3 600 0 (DO=0 3 1) OK<LF>
R<AT+PTEVT=0000 0 100 4 600 60 (DO=0 3 0) OK<LF>
R<AT+PTEVT=0000 0 100 5 600 120 (DO=0 3 1) OK<LF>
R<AT+PTEVT=0000 0 100 6 600 180 (DO=0 3 0) OK<LF>
R<AT+PTEVT=0000 0 100 7 0 () OK<LF>
R<AT+PTEVT=0000 0 100 8 0 () OK<LF>

[Set Example]
T>AT+PTEVT=0 7 30 3 (SEND=100 1 S “READ MODBUS”)<LF>
R<AT+PTEVT=0000 0 100 7 30 3 (SEND=100 1 S “READ MODBUS”) OK<LF>
```

10.48 AT+STEVT

Sleep timer event, Normal/Deep sleep 공통 적용
sleep 에서 깨어나서 slot 에 저장된 모든 명령 순차 실행, 최대 4 개의 slot 지원
쓰기/전달/읽기/문의(Set/Send/Get/Query)지원

[Set/Send format]
AT+STEVT=<DADDR> <SLOT> <(CMD)><CR/LF/CRLF>
SLOT 1~4, Slot index, AT command 를 저장할 수 있는 공간
CMD Sleep 에서 깨어나서 수행할 AT command

[Set/Send Response]
AT+STEVT=<DADDR> <RSSI> <BATT> <SLOT> <(CMD)> OK<LF>

[Get/Query format]
AT+STEVT?[<DADDR> <SLOT>]<CR/LF/CRLF>
SLOT 1~4, Slot index

[Get/Query Response]
GET/QUERY 명령에서 SLOT Option 을 생략하면 Slot index 를 1 ~ 4 까지 모두 출력
AT+STEVT=<OADDR> <RSSI> <BATT> <SLOT> <(CMD)> OK<LF>

[Get Example]
T>AT+STEVT?100 1<LF>
R<AT+STEVT=0100 0 100 1 (ADC?100) OK<LF>

T>AT+STEVT?100<LF>
R<AT+STEVT=0100 0 100 1 (ADC?100) OK<LF>
R<AT+STEVT=0100 0 100 2 (DO=100 3 1) OK<LF>
R<AT+STEVT=0100 0 100 3 (SEND=0 2 S “SLEEP WAKE”) OK<LF>
R<AT+STEVT=0100 0 100 4 () OK<LF>

[Set Example]
T>AT+STEVT=100 4 (DO=100 3 0)<LF>
R<AT+STEVT=0100 0 100 4 (DO=100 3 0)OK<LF>

10.49 AT+NSEVT

Normal sleep wake up event, trigger IO 에 사용자 명령을 지정
쓰기/전달/읽기/문의(Set/Send/Get/Query)지원

[Set/Send format]
AT+NSEVT=<DADDR> <SLOT> <IO> <ACTIVATE> <(CMD)><CR/LF/CRLF>
SLOT 1~4, Slot index, AT command 를 저장할 수 있는 공간
IO 7/8/9/11/12, 7: GPIO7, 8: GPIO8, 9: AUX~~, 11: Button1, 12: Button2~~
ACTIVATE 0/1, Slot index 에 저장된 AT command 의 실행 여부
0: Disable, 1: Enable
CMD Deep sleep 에서 깨어난 후 수행할 AT command

```
[Set/Send Response]
AT+NSEVT=<DADDR> <RSSI> <BATT> <SLOT> <IO> <ACTIVATE> <(CMD)>OK<LF>

[Get/Query format]
AT+NSEVT?[<DADDR> <SLOT>]<CR/LF/CRLF>
     SLOT               1~4, Slot index

[Get/Query Response]
GET/QUERY 명령에서 SLOT Option 을 생략하면 Slot index 를 1 ~ 4 까지 모두 출력
AT+NSEVT=<OADDR> <RSSI> <BATT> <SLOT> <IO> <ACTIVATE> <(CMD)> OK<LF>

[Get Example]
T>AT+NSEVT?100 1<LF>
R<AT+NSEVT=0100 0 100 1 1 1 (SEND=0 1 S “Door Open”) OK<LF>

T>AT+NSEVT?100<LF>
R<AT+NSEVT=0100 0 100 1 1 1 (SEND=0 1 S “Door Open”) OK<LF>
R<AT+NSEVT=0100 0 100 2 1 1 (DO=100 1 0) OK<LF>
R<AT+NSEVT=0100 0 100 3 3 1 (SEND=0 1 S “Door Close”) OK<LF>
R<AT+NSEVT=0100 0 100 4 3 1 (DO=100 1 1) OK<LF>

[Set Example]
T>AT+NSEVT=100 4 3 1 (DO=100 2 0)<LF>
R<AT+NSEVT=0100 0 100 4 3 1 (DO=100 2 0)OK<LF>
```

10.50 AT+DSEVT

```
Deep sleep wake up event, trigger IO 에 사용자 명령을 지정
쓰기/전달/읽기/문의(Set/Send/Get/Query)지원

[Set/Send format]
AT+DSEVT=<DADDR> <SLOT> <IO> <ACTIVATE> <(CMD)><CR/LF/CRLF>
        SLOT            1~4, Slot index, AT command 를 저장할 수 있는 공간
        IO              1/3/11, 1: GPIO1, 3: GPIO3, 11: Button1
        ACTIVATE        0/1, Slot index 에 저장된 AT command 의 실행 여부
                        0: Disable, 1: ACTIVATE
        CMD             Deep sleep 에서 깨어난 후 수행할 AT command

[Set/Send Response]
AT+DSEVT=<DADDR> <RSSI> <BATT> <SLOT> <IO> <ACTIVATE> <(CMD)>OK<LF>

[Get/Query format]
AT+DSEVT?[<DADDR> <SLOT>]<CR/LF/CRLF>
        SLOT            1~4, Slot index

[Get/Query Response]
GET/QUERY 명령에서 SLOT Option 을 생략하면 Slot index 를 1 ~ 4 까지 모두 출력
AT+DSEVT=<OADDR> <RSSI> <BATT> <SLOT> <IO> <ACTIVATE> <(CMD)> OK<LF>
```

```
[Get Example]
T>AT+DSEVT?100 1<LF>
R<AT+DSEVT=0100 0 100 1 1 1 (SEND=0 1 S “Door Open”) OK<LF>

T>AT+DSEVT?100<LF>
R<AT+DSEVT=0100 0 100 1 1 1 (SEND=0 1 S “Door Open”) OK<LF>
R<AT+DSEVT=0100 0 100 2 1 1 (DO=100 1 0) OK<LF>
R<AT+DSEVT=0100 0 100 3 3 1 (SEND=0 1 S “Door Close”) OK<LF>
R<AT+DSEVT=0100 0 100 4 3 1 (DO=100 1 1) OK<LF>

[Set Example]
T>AT+DSEVT=100 4 3 1 (DO=100 2 0)<LF>
R<AT+DSEVT=0100 0 100 4 3 1 (DO=100 2 0)OK<LF>
```

10.51 AT+DSWUP

```
Configure IO to wake from deep sleep
Deep sleep 을 깨울 수 있는 IO(GPIO1, GPIO3) 지정, Button1 은 항상 깨울 수 있음
Positive edge 로 Wake up 기능을 설정하면 Pad 는 자동으로 Pull-down 으로 설정된다.
Negative edge 로 Wake up 기능을 설정하면 Pad 는 자동으로 Pull-up 으로 설정된다.
쓰기/전달/읽기/문의(Set/Send/Get/Query)지원
```

```
[Set/Send format]
AT+DSWUP=<DADDR> <IO1_CONF> <IO3_CONF> <CR/LF/CRLF>
        XXX_CONF        Set each IO to wake up from sleep or not
                        0: Disable wake up
                        1: Enable wake-up on positive edge, and set IO Pad to pull-down
                        2: Enable wake-up on negative edge, and set IO Pad to pull-up
[Set/Send Response]
AT+DSWUP=<DADDR> <RSSI> <BATT> <IO1_CONF> <IO3_CONF>OK<LF>

[Get/Query format]
AT+DSWUP?[<DADDR>]<CR/LF/CRLF>

[Get/Query Response]
AT+DSWUP=<OADDR> <RSSI> <BATT> <IO1_CONF> <IO3_CONF> OK<LF>

[Get Example]
T>AT+DSWUP?100<LF>
R<AT+DSWUP=0100 0 100 1 1 OK<LF>

[Set Example]
T>AT+DSWUP=100 1 0<LF>                  #Only GPIO1 can wake up from deep sleep mode
R<AT+DSWUP=0100 0 100 1 0 OK<LF>
```

10.52 AT+SLEEP

Sleep mode 실행, AP 는 Sleep mode 로 진입하지 않는다
실행/전달(Execute/Send) 지원

[Execute/Send format]
AT+SLEEP=<DADDR> <PERIOD> <MODE> <AGAIN> <CR/LF/CRLF>

PERIOD	0 ~ 2,073,600 [sec], Time out of Sleep timer, 최대 24 일 Set 0 to disable the sleep timer
MODE	Sleep mode 0/3/4 0: Disable 3: Normal sleep 4: Deep sleep
AGAIN	깨어난 후 다시 잠들기(Wake up and go back to sleep again) 0: Disable 1: Enable

[Execute Response]
없음. Sleep 명령이 실행되면, 즉시 UART, I2C, RF Block 이 중지됨

[Send Response]
AT+SLEEP=<OADDR> <RSSI> <BATT> OK<LF>

DADDR	목적지 SADDR
RSSI	수신신호세기[dBm]
BATT	배터리용량[%]

[Send Example]
AP(0000)에서 Node(0100)에게 명령전달: Normal sleep 에 들어가고, 1 시간 마다 깨어난 뒤 다시 잠들도록 설정
T>AT+SLEEP=0100 3600 3 1<LF>

10.53 AT+PER

PER(Packet error rate) test 실행
실행/전달(Execute/Send) 지원

[Execute/Send format]
AT+PER=<DADDR> <TADDR> <NUM> <LENGTH> <INTERVAL> <CR/LF/CRLF>

DADDR	Test 를 수행할 device 의 SADDR, 이 Device 에서 Test packet 을 송신한다.
TADDR	PER 을 측정할 상대 device 의 SADDR, 이 Device 로 Test packet 을 전송
NUM	1~65535, Number of test trials, Test packet 을 전송할 횟수
LENGTH	0~64, Data length of test packet
INTERVAL	1~1000ms, Test packet 을 송신할 때 시간 간격

[Execute Response]
시험 중에는 진행 상황을 다음 메시지로 알림
AT+PER.TX=<TADDR> <RSSI> <BATT> <COUNT> <TXCNT> <ACKCNT> <RMRSSI> OK/FAIL(ACK/TX)<LF>

TADDR	Test packet 을 보낼 디바이스 주소

PER 시험의 상대 디바이스(PER test packet 을 수신한 device)는 수신한 Test packet 을 다음과 같이 알림
AT+PER.RX=<OADDR> <RSSI> <BATT> <INDEX> <RXCNT> <TNUM> OK<LF>

```
        OADDR           Test packet 을 송신하는 디바이스 주소

시험이 종료되면 PER 결과를 다음 메시지로 알림
AT+IND=<OADDR> <RSSI> <BATT> <EVENT:"PER"> <TADDR> <NUM> <TX> <RX> <AVRSSI> <AVRMRSSI> OK<LF>

[PER Example]
AP(0)에서 node(100)으로 데이터 길이 32Byte 인 PER test packet 을 10ms 간격으로 20 회 전송
T>AT+PER=0 100 20 32 10 <LF>
R<AT+PER.TX=0100 -95 100 1 1 1 -82 OK<LF>
R<AT+PER.TX=0100   0 100 2 2 1  NA FAIL(ACK)<LF>        #ACK 수신 못함
R<AT+PER.TX=0100 -96 100 3 3 2 -81 OK<LF>
R<AT+PER.TX=0100   0 100 4 3 2  NA FAIL(TX)<LF>         #TX fail, RF busy
R<AT+PER.TX=0100 -90 100 5 4 3 -82 OK<LF>
…
R<AT+PER.TX=100 -94 100 20 18 16 -82 OK<LF>             #20 회 중 TX 성공 18 회, ACK 수신 16 회
R<AT+IND=0000 0 100 PER 0100 20 18 16 -94 -82 OK<LF>    #결과: 수신평균 -94dBm, 상대방 수신평균-82dBm

상대 디바이스(0100)에서 출력되는 PER test packet 수신 알림 메시지
R<AT+PER.RX=0000 -82 100  1 1 20 OK<LF>
R<AT+PER.RX=0000 -81 100  3 2 20 OK<LF>
R<AT+PER.RX=0000 -82 100  5 3 20 OK<LF>
…
R<AT+PER.RX=0000 -82 100 20 16 20 OK<LF>                #총 20 개 중 16 개 수신
```

10.54 AT+PER.TX

```
PER(Packet error rate) test 를 수행할 때 Test packet 을 전송한 결과를 알림
Test Packet 송신을 완료하고 ACK 를 수신하거나, Time out 이 될 경우 알림
무선 채널이 혼잡하여 송신하지 못할 경우(CSMA/CA 실패)에도 알림
알림(Indication) 메시지
```

```
[Indication format]
AT+PER.TX=<TADDR> <RSSI> <BATT> <COUNT> <TXCNT> <ACKCNT> <RMRSSI> OK/FAIL(ACK/TX)<LF>
        TADDR           PER 을 측정할 상대 device 의 SADDR
        RSSI            Test packet 에 대해 수신한 ACK 의 RSSI
        COUNT           1~65535, Count of test trials, Test packet 을 전송한 Counter
        TXCNT           Test packet 송신 성공 count
        ACKCNT          송신한 Test packet 에 대해 수신한 ACK count
        RMRSSI          상대방이 수신한 Test packet 의 RSSI

[Example]
AP(0)에서 node(100)으로 데이터 길이 32Byte 인 PER test packet 을 10ms 간격으로 20 회 전송
T>AT+PER=0 100 20 32 10 <LF>
R<AT+PER.TX=0100 -95 100 1 1 1 -82 OK<LF>               #1 번째 송신 성공, 100 이 수신한 RSSI 는 -82dBm
R<AT+PER.TX=0100   0 100 2 2 1  NA FAIL(ACK)<LF>        #ACK 수신 못함, 송신은 성공,
R<AT+PER.TX=0100 -96 100 3 3 2 -81 OK<LF>               #3 번째 송신 성공, 100 이 수신한 RSSI 는 -82dBm
R<AT+PER.TX=0100   0 100 4 3 2  NA FAIL(TX)<LF>         #TX fail, RF busy
R<AT+PER.TX=0100 -90 100 5 4 3 -82 OK<LF>
```

```
…
R<AT+PER.TX=100 -94 100 20 18 16 -82 OK<LF>              #20 회 중 TX 성공 18 회, ACK 수신 16 회
R<AT+IND=0000 0 100 PER 0100 20 18 16 -94 -82 OK<LF>    #결과: 수신평균 -94dBm, 상대방 수신평균-82dBm
```

10.55 AT+PER.RX

PER(Packet error rate) test 를 수행할 때 Test packet 을 수신한 device 에서 수신 결과를 알림
Test Packet 을 수신했을 때 알림
새로 Test 를 수행하기 전에 수신 Counter 를 초기화하기 위해 반드시 AT+PERSTOP 명령을 수행해야 함
알림(Indication) 메시지

[Indication format]
AT+PER.RX=<OADDR> <RSSI> <BATT> <COUNT> <RXCNT> <TNUM> OK<LF>

OADDR	PER Test packet 을 송신한 device 의 SADDR
RSSI	수신한 Test packet 의 RSSI
COUNT	1~65535, Count of test trials, Test packet 을 전송한 Counter
RXCNT	Test packet 수신 count
TNUM	PER Test 의 총 전송 횟수

[Example]
PER test packet 수신 알림 메시지, test packet 송신한 device 는 AP(0000)

```
R<AT+PER.RX=0000 -82 100  1 1 20 OK<LF>          #20 개 중 1 번째 송신한 packet 수신 성공, 1 개 수신
R<AT+PER.RX=0000 -81 100  3 2 20 OK<LF>          #20 개 중 3 번째 송신한 packet 수신 성공, 2 개 수신
R<AT+PER.RX=0000 -82 100  5 3 20 OK<LF>          #20 개 중 5 번째 송신한 packet 수신 성공, 3 개 수신
…
R<AT+PER.RX=0000 -82 100 20 16 20 OK<LF>         #20 개 중 20 번째 송신한 packet 수신 성공, 16 개 수신
```

10.56 AT+PERSTOP

PER(Packet error rate) test 중지
PER Test packet 을 송신 수행 중이면, 이를 중지하고, 수신 중이면 수신 Counter 를 초기화 한다.
새로 PER 시험을 진행하기 전에 Test packet 을 수신하는 device 는 반드시 이를 수행해야 한다.
실행/전달(Execute/Send) 지원

[Execute/Send format]
AT+PERSTOP=<DADDR> <CR/LF/CRLF>

DADDR	Test 를 중지할 device 의 SADDR

[Execute/Send Response]
AT+PERSTOP=<OADDR> <RSSI> <BATT> OK<LF>

[Example]

```
T>AT+PERSTOP=0 <LF>
R>AT+PERSTOP=0 0 100 OK<LF>
```

PER Test 상대방 device 도 중지(Counter 초기화)

```
T>AT+PERSTOP=100 <LF>
R>AT+PERSTOP=0100 0 100 OK<LF>
```

10.57 AT+SCAN

RF channel 에 대해 수신단의 RSSI 를 측정
AP 의 RF CH 을 선정하기 전에 혼잡하지 않은 주파수를 찾기 위해 사용한다.
실행/전달(Execute/Send)지원

[Execute/Send format]
AT+SCAN=<DADDR> <CH> <SCAN_T><CR/LF/CRLF>

DADDR	측정을 수행할 Device 주소	
CH	0~21,	측정할 RF Channel number
	0	전체 CH 에 대해 측정, 측정이 모두 끝난 후에 취합 결과를 IND 로 응답
	1~21	지정된 CH 만 측정
SCAN_T	1~60[sec], Scan time, 측정 시간	

[Execute/Send Response]
각 RF CH 마다 결과를 응답한다.
AT+SCAN=<OADDR> <RSSI> <BATT> <CH> <MIN_RSSI> <MAX_RSSI> <AVG_RSSI> OK<LF>

DADDR	Scan 을 수행한 Device
RSSI	수신신호세기[dBm]
BATT	배터리용량[%]
CH	Scan 한 RF channel number
MIN_RSSI	Minimum RSSI
MAX_RSSI	Maximum RSSI
AVG_RSSI	Average RSSI

전체 CH 을 Scan 한 경우, 다음 IND 메시지로 Maximum RSSI 가 가장 낮은 CH 을 Report 한다.
AT+IND=<OADDR> <RSSI> <BATT> <EVENT:"SCAN"> <CH> <MAX_RSSI> <OK/FAIL><LF>

[Example]

```
T>AT+SCAN=0 0 5<LF>                              #모든 CH 을 CH 당 5 초씩 Scan
R<AT+SCAN=0000 0 100 1 -109 -100 -108 OK<LF>     #CH1:  Min(-109dBm), Max(-100dBm), Avg(-108dBm)
R<AT+SCAN=0000 0 100 2 -110 -105 -105 OK<LF>     #CH2:  Min(-110dBm), Max(-105dBm), Avg(-105dBm)
R<AT+SCAN=0000 0 100 3  -97  -95  -99 OK<LF>     #CH3:  Min( -97dBm), Max( -95dBm), Avg( -99dBm)
…
R<AT+SCAN=0000 0 100 21 -107 -102 -105 OK<LF>    #CH21: Min(-107dBm), Max(-102dBm), Avg(-105dBm)
R<AT+IND=0000 0 100 SCAN 2 -105 OK<LF>           #전체 채널 중 CH2 을 추천, MAX RSSI 가 가장 낮음
```

10.58 AT+TIMEOUT

버튼으로 Join 이나 초기화를 할 때, 남은 사용자 입력 시간을 알림
알림(Indication) 메시지

[Indication format]
AT+TIMEOUT=<OADDR> <RSSI> <BATT> <TIMEOUT> OK<LF>

OADDR	Timer 작동 중인 Device
TIMEOUT	[sec], 사용자 입력을 할 수 있는 남은 시간

```
[Example]
버튼으로 초기화를 할 때, 초기화 모드 진입 후에 IO/Event 초기화를 위해 버튼 2 를 누르고 있으면
R<AT+TIMEOUT=0000 0 100 5 OK<LF>                #Timer 시작, 5 초 남음
R<AT+TIMEOUT=0000 0 100 4 OK<LF>                #Timer 4 초 남음
R<AT+TIMEOUT=0000 0 100 3 OK<LF>                #Timer 3 초 남음
R<AT+TIMEOUT=0000 0 100 2 OK<LF>                #Timer 2 초 남음
R<AT+TIMEOUT=0000 0 100 1 OK<LF>                #Timer 1 초 남음
R<AT+TIMEOUT=0000 0 100 0 OK<LF>                #Timer 만료, 버튼을 떼도 됨
R<AT+IND=0000 0 100 INIT 1 OK<LF>               #초기화 완료(IO/Event 초기화)
```

10.59 AT+REPORT

```
IND 메시지를 자동으로 AP 로 전송하는 기능 설정
쓰기/전달/읽기/문의(Set/Send/Get/Query)지원

[Set/Send format]
AT+REPORT=<DADDR> <EN><CR/LF/CRLF>
        EN              0/1,    IND 메시지 전송 기능
                        0       IND 메시지 전송 기능 중지
                        1       IND 메시지 전송 기능 활성화

[Set/Send Response]
AT+REPORT=<DADDR> <RSSI> <BATT> <EN> OK<LF>
        DADDR           목적지 SADDR
        RSSI            수신신호세기[dBm]
        BATT            배터리용량[%]
        EN              IND 메시지 전송 모드

[Get/Query format]
AT+REPORT?[DADDR]<CR/LF/CRLF>

[Get/Query Response]
AT+REPORT=<OADDR> <RSSI> <BATT> <EN> OK<LF>
        OADDR           응답을 보낸 디바이스 주소
        RSSI            수신신호세기[dBm]
        BATT            배터리용량[%]
        EN              IND 메시지 전송 모드

[Get Example]
T>AT+REPORT?0100<LF>
R<AT+REPORT=0100 0 100 0 OK<LF>         # IND 자동 전송 기능 중지

[Set Example]
T>AT+REPORT=0100 1<LF>
R<AT+REPORT=0100 0 100 1 OK<CR>         # IND 자동 전송 기능 활성화
```

10.60 AT+IND

Event 발생을 알림
알림(Indication) 지원

```
[Indication format]
AT+IND=<OADDR> <RSSI> <BATT> <EVENT> [<PARAMS> … ] <OK/FAIL><LF>
        OADDR   IND 를 생성한 디바이스 주소
        EVENT   IND 를 발생시킨 Event, 이에 따라 각기 다른 파라메터가 정의 되어 있다.
                DIEVT   DI event 발생 알림      PARAMS: <IO> <DIN>
                RESET   Reset 알림              PARAMS: <REASON>
                I2C     I2C Read 완료 알림      PARAMS: <DATA> … <DATA>
                ADC     ADC Read 완료 알림      PARAMS: <ADC>
                JOIN    Join complete 알림      PARAMS: 없음
                EXIT    network exit 알림       PARAMS: 없음
                INIT    버튼으로 Init 완료 알림 PARAMS: <LEVEL>
                SCAN    RF CH scan 결과 알림    PARAMS: <CH> <MAX_RSSI>
                PER     PER 시험 종료 알림      PARAMS: <ADDR> <NUM> <TX> <RX> <AVRSSI> <AVRMRSSI>

[Indication Example]
R<AT+IND=0100 -85 100 DIEVT 3 1 OK<LF>  #DIEVT 알림, GPIO3 의 DIN 값이 1 로 변함
R<AT+IND=0100 -85 100 RESET 2 OK<LF>            #RESET 알림, Reset pin 에 의한 Reset
R<AT+IND=0100 -85 100 I2C 30 45 OK<LF>  #I2C Read 알림, I2C 읽은 값 30, 45
R<AT+IND=0100 -85 100 ADC 2569 OK<LF>   #ADC 알림, ADC 값 2569mV
R<AT+IND=0300 -85 100 JOIN OK<LF>               #JOIN 완료 알림, 신규 Node 0300 네트워크 가입
R<AT+IND=0300 -85 100 EXIT OK<LF>               #EXIT 알림, Node 0300 네트워크 나감
R<AT+IND=0100 -85 100 INIT 1 OK<LF>             #INIT 알림, IO/Event 설정 초기화
R<AT+IND=0000 -85 100 SCAN 3 -105 OK<LF>        #SCAN 알림, MAX RSSI 가 -105dBm 인 CH 3 을 추천
R<AT+IND=0000 0 100 PER 0100 20 18 16 -94 -82 OK<LF>    #PER 시험 결과, Node 100 과 PER 시험
```