

UART 해킹 기초

mongii@grayhash

UART란?

- Universal asynchronous receiver/transmitter
 - 범용 비동기 송/수신기
- 직렬(Serial) 통신 프로토콜
 - 데이터 송신/수신 시 각각 하나의 LINE만 이용
- 하드웨어 통신 규약의 한 종류
- “프로토콜이 매우 간단함”

UART의 장점들

- 프로토콜이 단순하다.
- 관련 프로그램 구하기가 쉽다.
 - Putty, Xshell, 하이퍼 터미널, ...
- 관련 장비를 구하기가 쉽다.
 - USB-to-Serial, USB-to-UART

해커가 UART를 통해 얻을 수 있는 것들

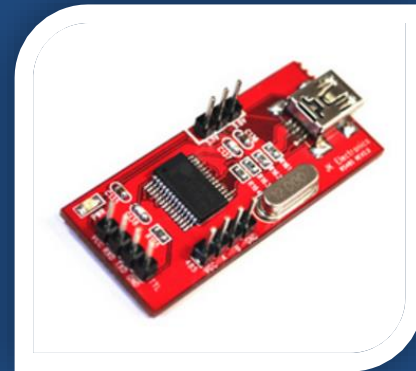
- 커널, OS 메시지
 - 취약점 공략에 필요한 각종 정보 획득
- 디버그 메시지
 - Ex> `printf("initializing network adaptor ok\n");`
- 오류 메시지
 - Ex> Segmentation fault, command not found

해커가 UART를 통해 얻을 수 있는 것들

- Hidden or Setting Menu
- 부트로더(Bootloader)
 - 펌웨어 획득
 - 새로운 펌웨어 Writing
- 커맨드 셸(Command Shell)
 - 펌웨어, 바이너리 획득
 - 동적 분석 가능

UART 해킹을 위한 필요 장비

- USB to Serial
 - Rabbit
 - <http://bit.ly/29wTgof>
- 점퍼 케이블
 - <http://bit.ly/29ExctC>



UART 해킹을 위한 필요 장비

- 멀티테스터
 - DM-300A
 - <http://bit.ly/29vyfxZ>



- Logic Analyzer
 - <https://www.saleae.com/>
 - <http://bit.ly/29ywZZw>



UART Pin의 구성

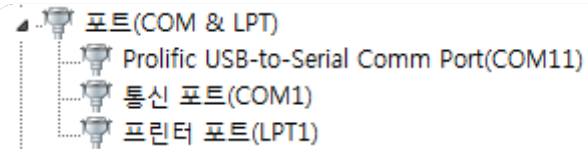
- 총 4개의 핀 사용
 - TX : 데이터 송신 핀
 - RX : 데이터 수신 핀
 - GND : 그라운드
 - VCC : 전압



- TX와 RX는 항상 자신의 입장에서 봐야 한다.
 - PC의 TX : PC에서 데이터 송신
 - 공유기의 TX : 공유기에서 데이터 송신

UART 연결 절차 요약

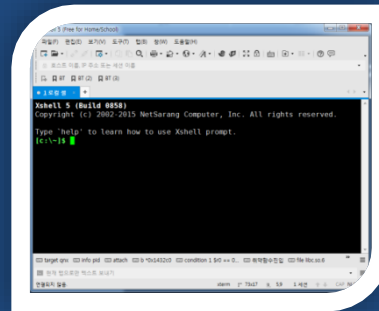
- 관련 USB 드라이버 설치
 - CP2102, PL2303, FTDI 등



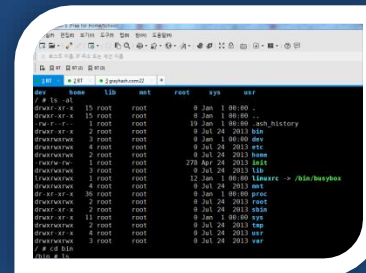
- 점퍼 케이블 연결



- 터미널 소프트웨어 설치
 - Putty
 - Xshell
 - screen

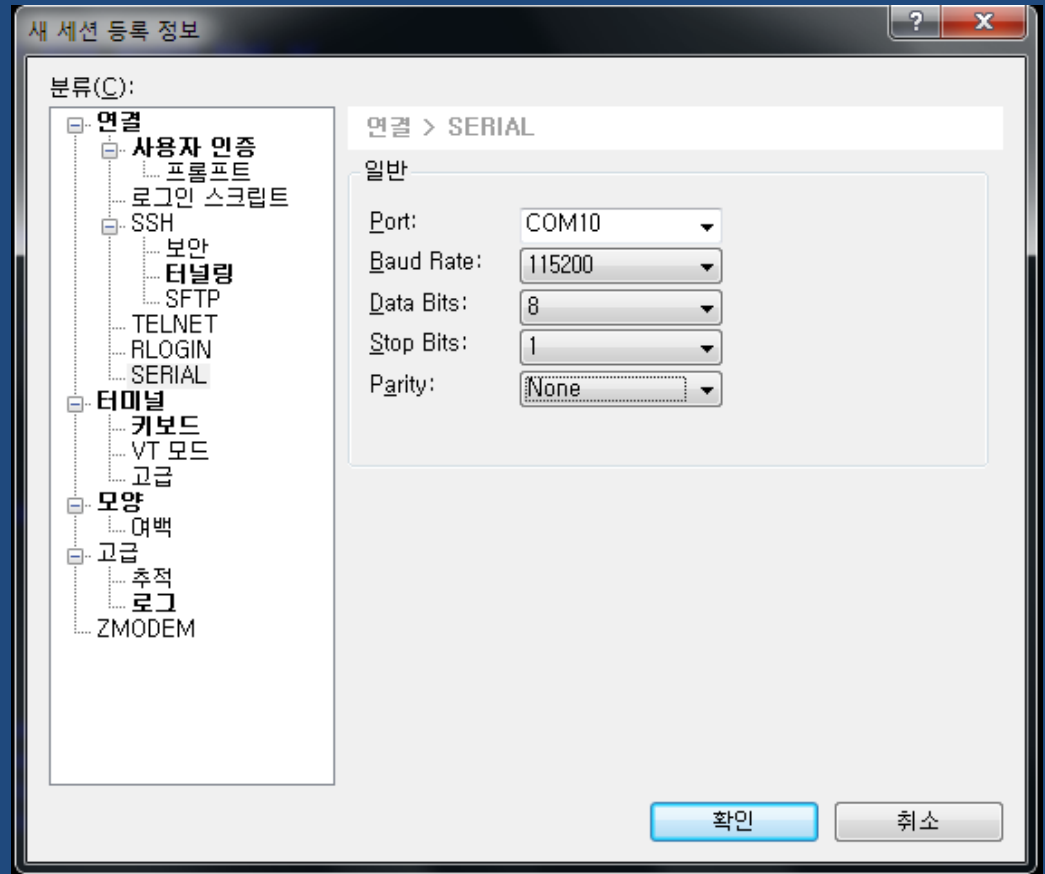


- 연결 정보 설정 및 연결 수행



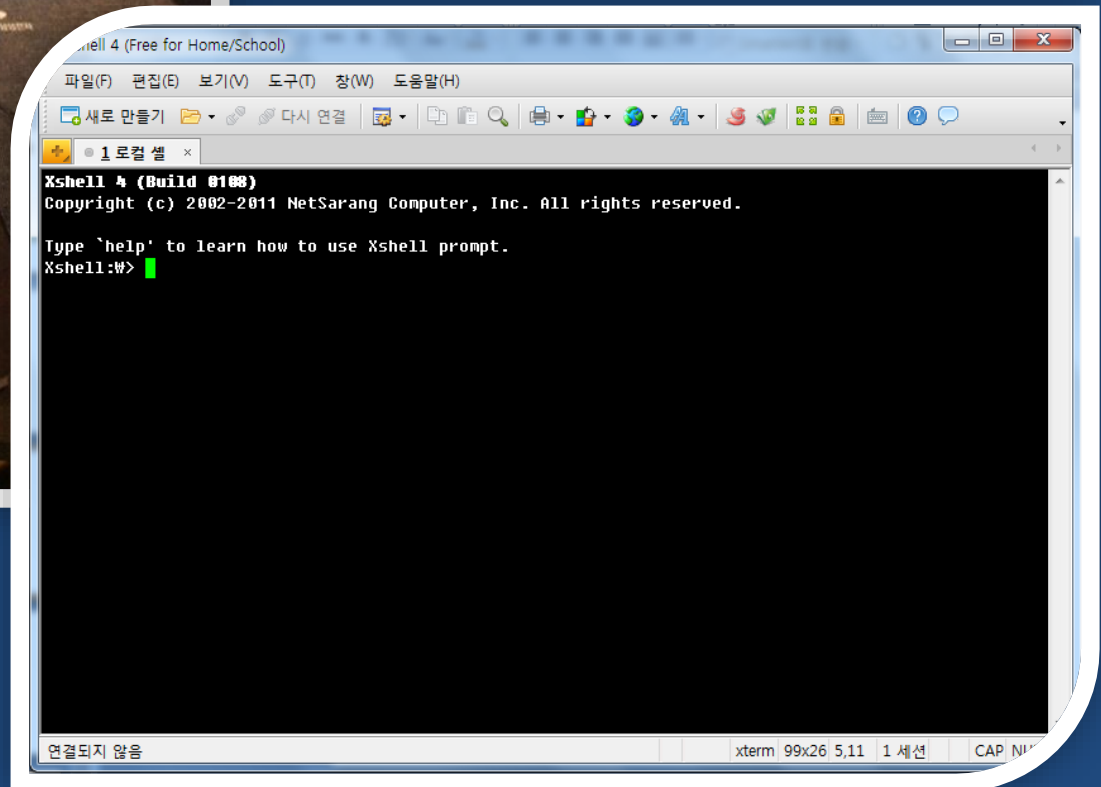
접속 설정

- COM 포트 번호
- Baudrate
- Data Bits
- Stop Bits
- Parity



UART 연결

- USB-UART 연결 및 터미널 프로그램 실행



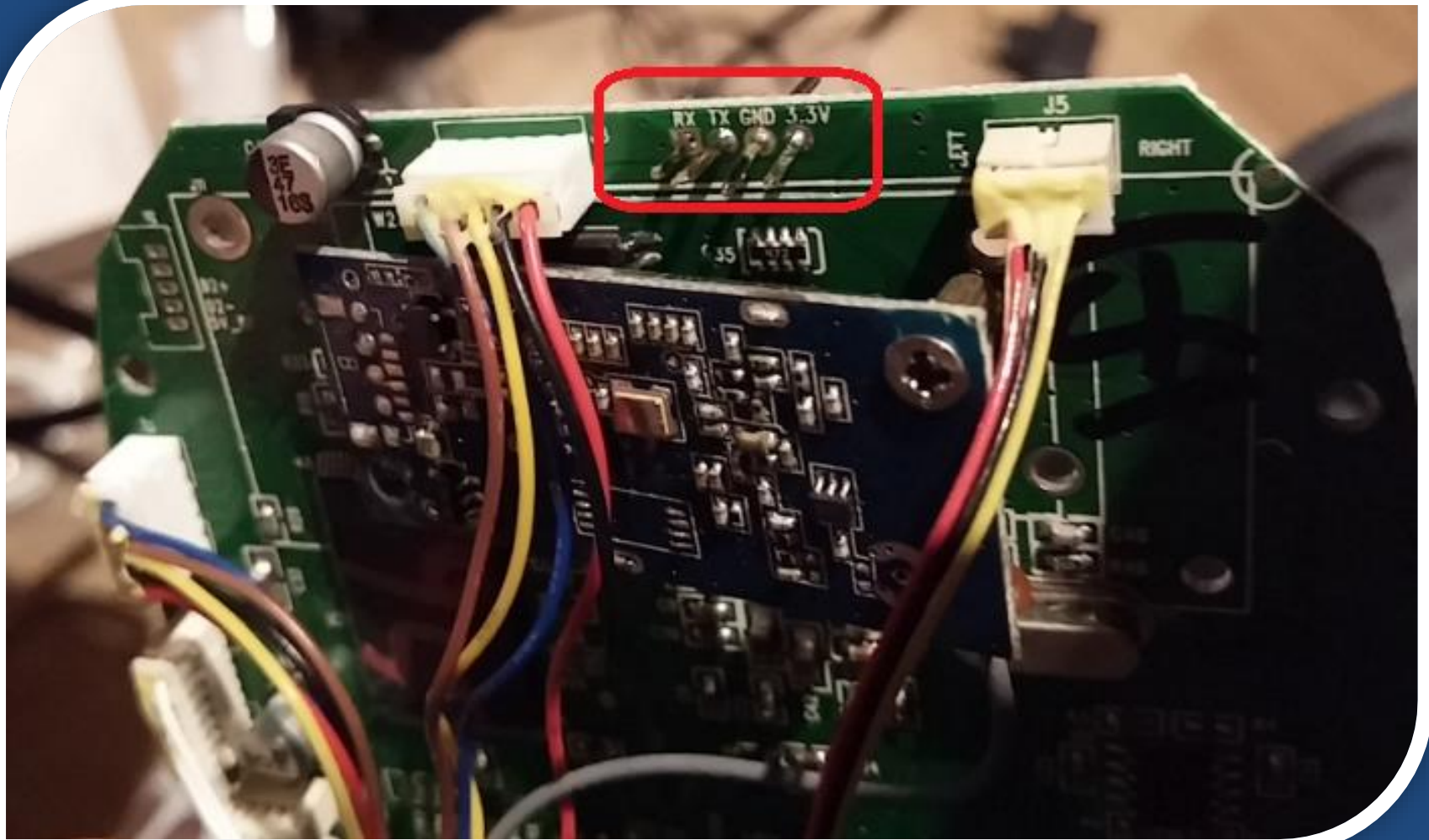
실습 - IPTIME UART 연결

1. IPTIME에서 UART 핀을 찾아보세요.
2. UART 연결을 시도해 보세요.

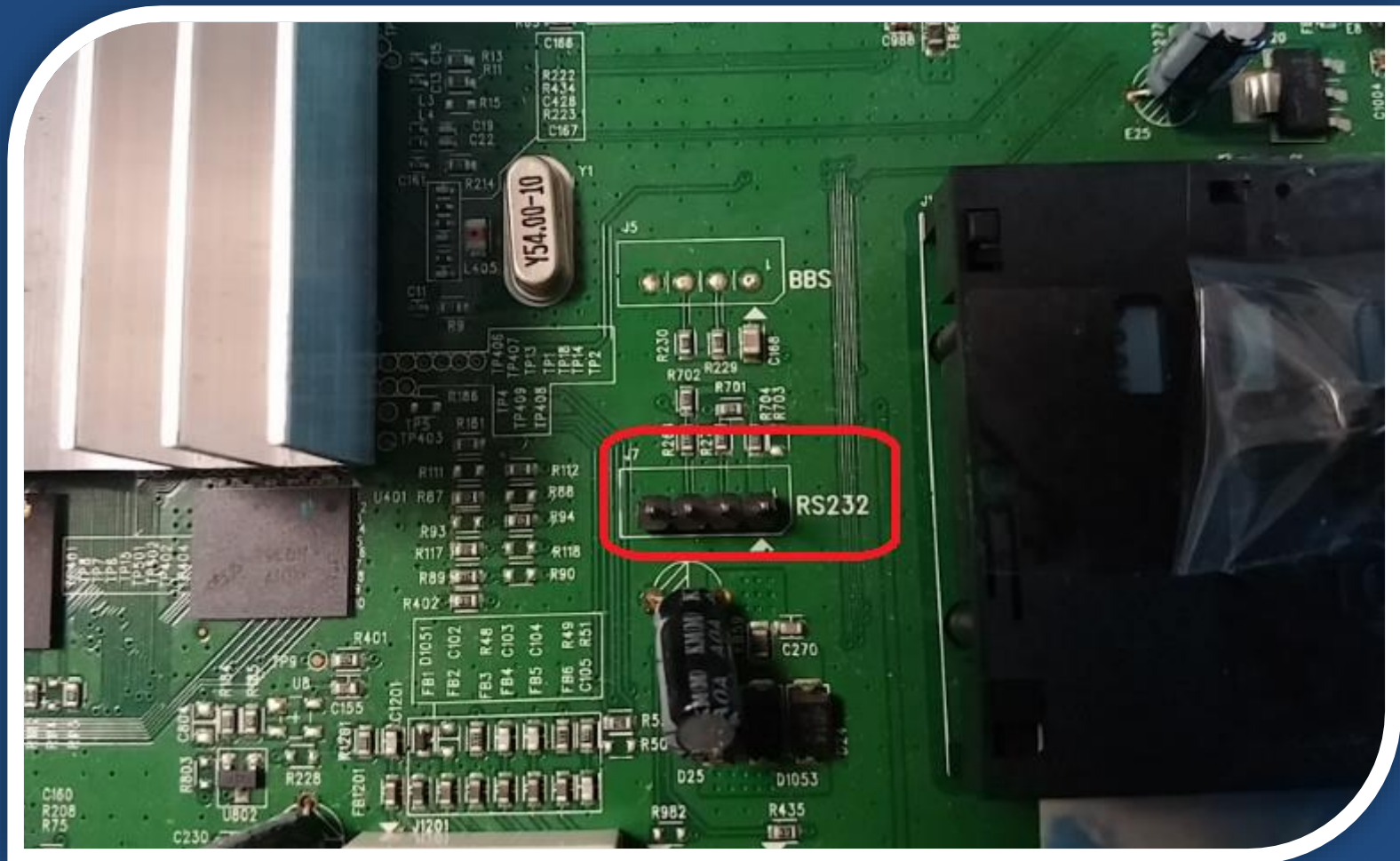
주의 : UART 장비의 VCC 핀은 사용하지 않습니다.

UART 핀 찾기

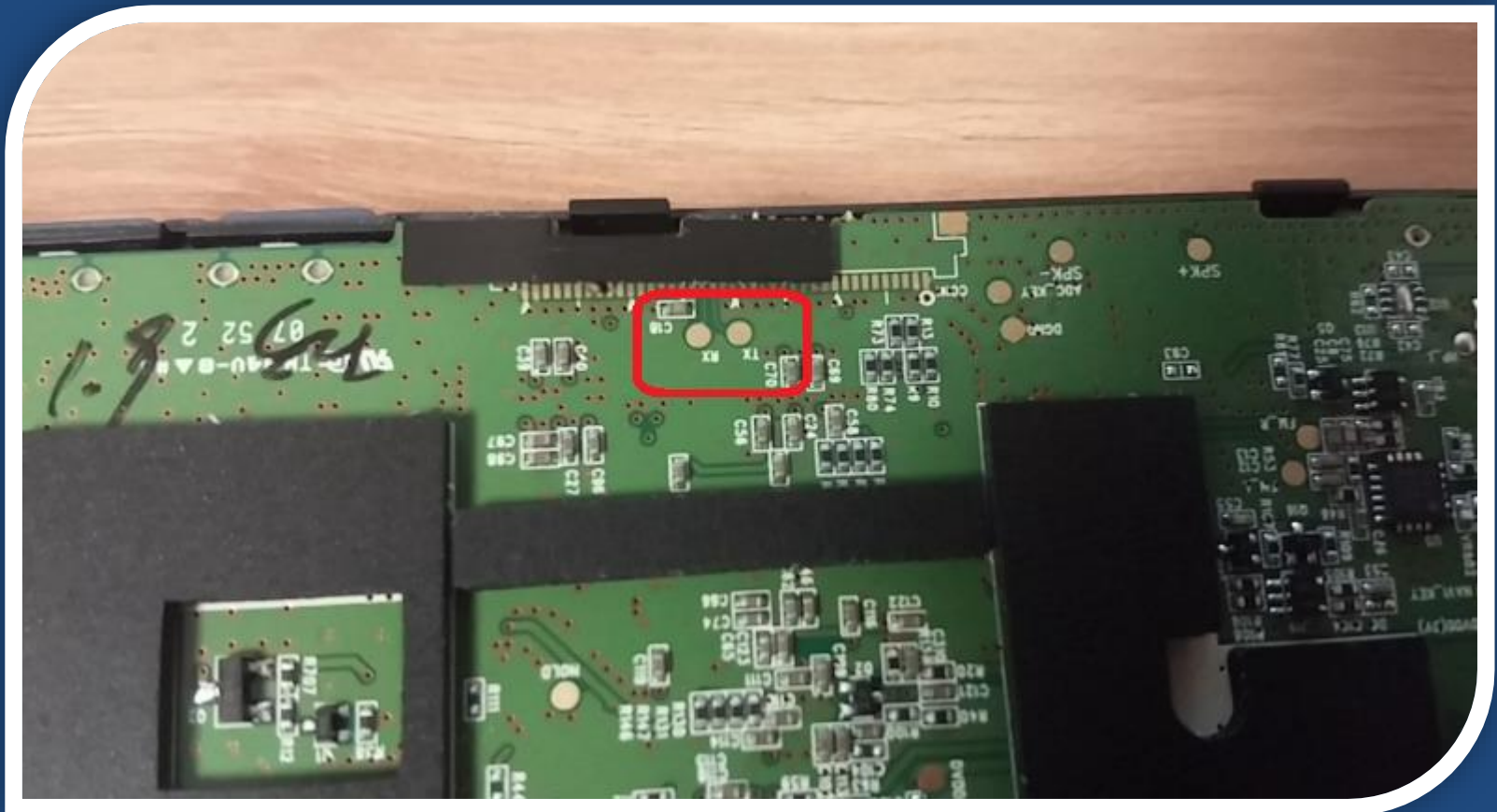
1. PCB의 식자로 확인



1. PCB의 식자로 확인



1. PCB의 식자로 확인

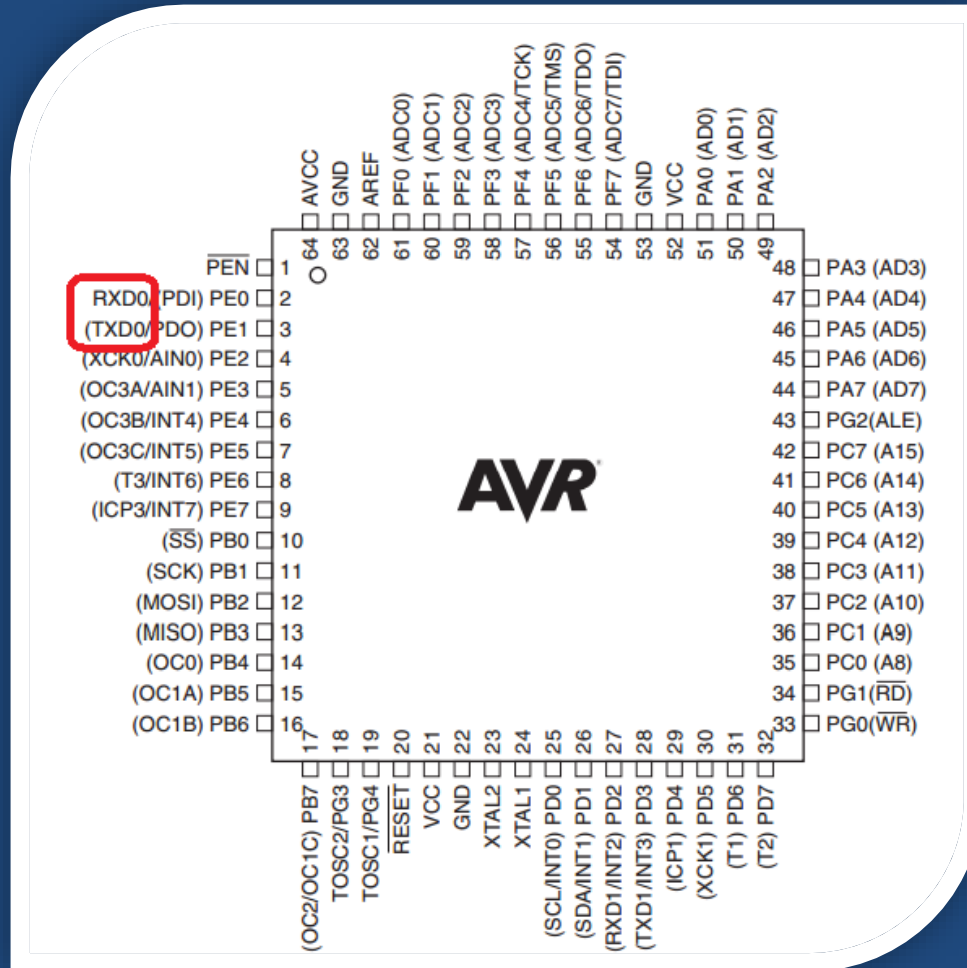


2. 4핀 배열로 확인

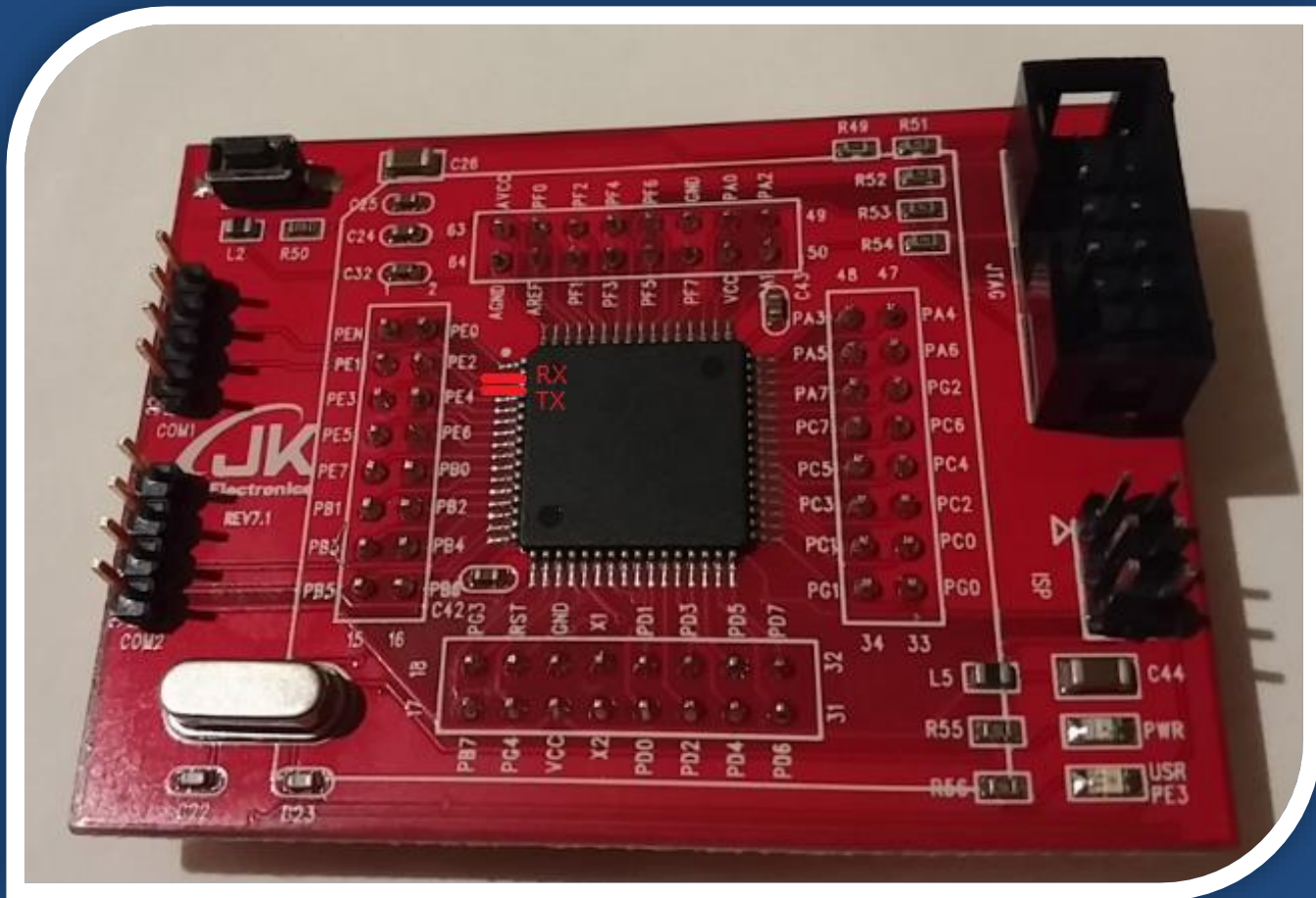


* 핀이 아닌 PAD로 존재할 수도 있다.

3. Datasheet를 보고 핀 따라가기

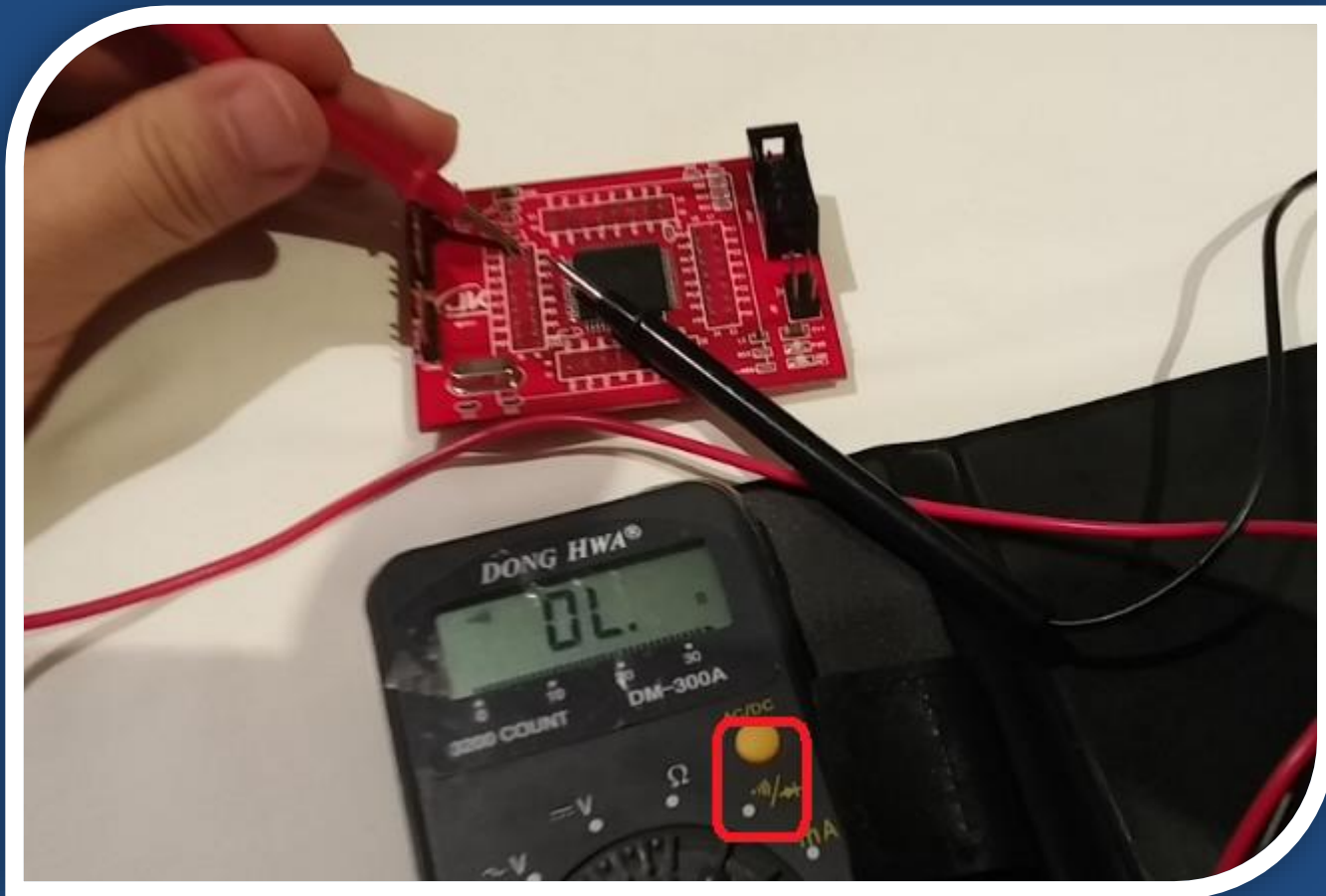


3. Datasheet를 보고 핀 따라가기



3. Datasheet를 보고 핀 따라가기

- 멀티테스터 활용



RX/TX/VCC/GND
핀 찾기

1. 멀티테스터로 찾기

- 5v(혹은 3.3v)가 잡힐 때의 (-) 리드선 = GND
- (+) 리드선 = VCC or TX
- 나머지 하나 : RX

2. LED를 이용한 방법

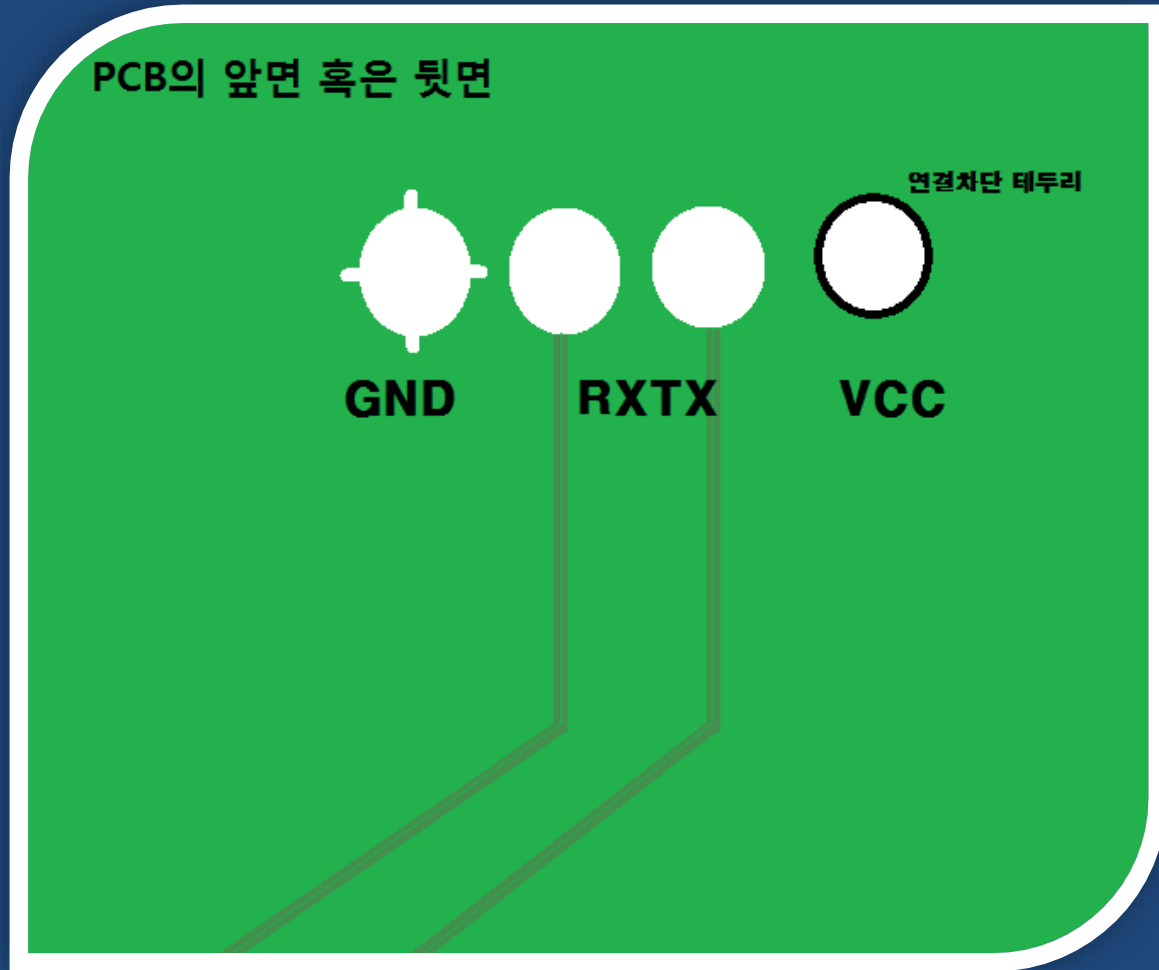
- LED에 불이 들어올 때 LED의 (-)극 : GND
- 지속적으로 불이 들어오는 핀 : VCC
- 전원 ON 시 깜빡이는 핀 : TX
- 나머지 하나 : RX



3. PCB의 특성을 이용한 방법

- PCB의 배경(초록 바탕) 부분의 특성
 - 일반적으로 윗면 혹은 아랫면은 모두 GND(-)
 - 사용빈도가 높은 - 를 용이하게 제공하기 위함
 - 회로의 노이즈를 줄이기 위함
- 즉, PCB 윗면 혹은 아랫면으로 연결된 핀은 GND
- RX와 TX는 라인으로 MCU에 연결되어 있음
- VCC는 PCB의 다른 층으로 연결이 되어 있음

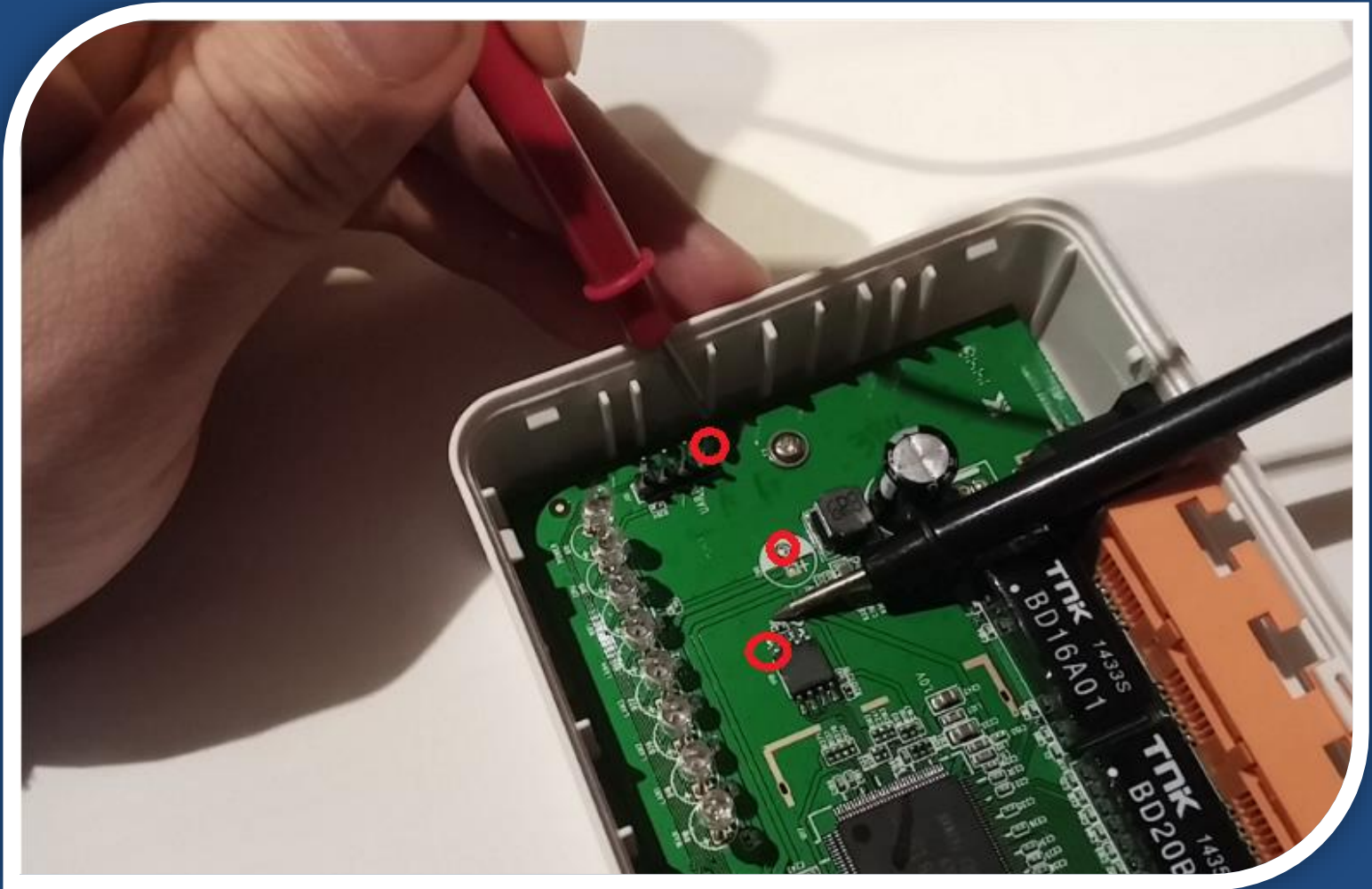
3. PCB의 특성을 이용한 방법



4. 멀티테스터 통전 테스트로 찾기

- 멀티테스터를 통전 테스트 모드로 설정
- PCB 내의 확실한 (+) 혹은 (-)에 연결
 - 소리가 나는지 확인
- Datasheet가 있는 IC의 (+) 혹은 (-)에 연결
 - 소리가 나는지 확인

4. 멀티테스터 통전 테스트로 찾기



Baudrate 찾기

Baudrate(보레이트)

- Clock을 사용하지 않기 때문에 HIGH/LOW를 구분할 수 있는 기준 필요
- Baudrate = 1초에 몇 개의 HIGH/LOW 신호를 보낼 것이냐를 정의
- 높을 수록 데이터 전송 속도가 빨라짐

Baudrate 찾기

- 자주 사용되는 값 Brute Force
 - 115200 (가장 많음)
 - 57600
 - 38400
 - 19200
 - 9600
- 신호분석기를 이용한 방법 (차후 진행)

깨진 글자가 나오는 경우

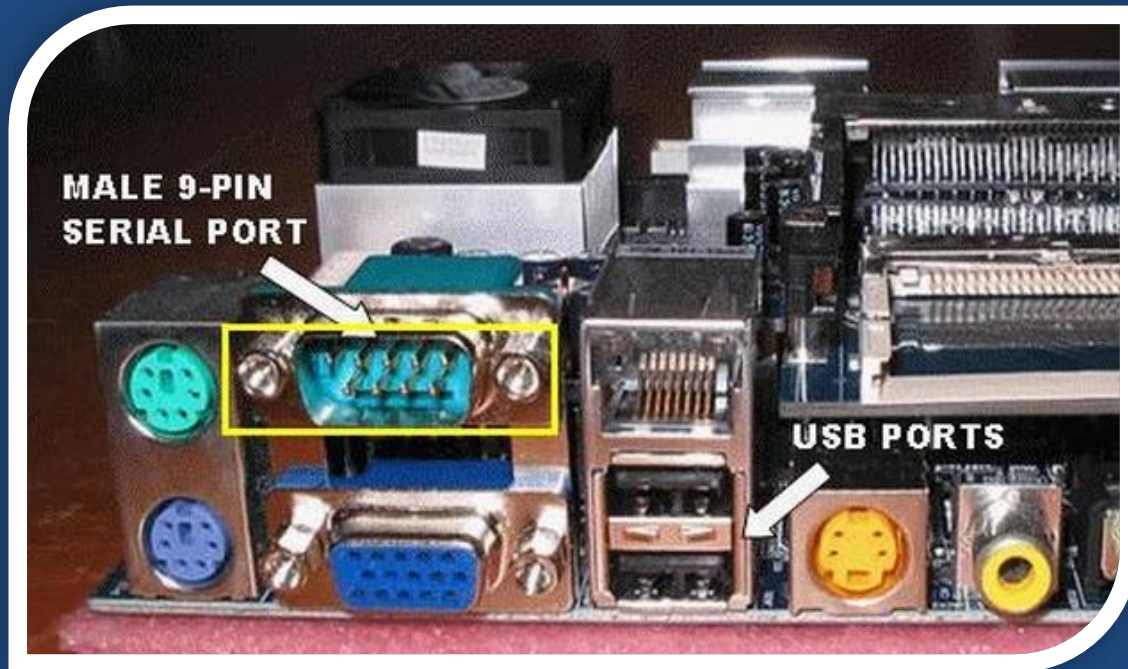
- Baudrate가 안 맞을 경우
 - 올바른 값으로 맞춰준다
- GND가 안 맞을 경우
 - GND 핀을 제대로 연결해 준다

VCC를 연결하는 이유

- 어댑터 없이 간편하게 전원 공급
- 방향 : UART 장치 → 대상 기기
- 전원이 들어와있는 상태라면 VCC는 불필요
- 즉, VCC는 그냥 서비스!

과거의 UART

- 오래된 desktop PC에서나 볼 수 있는..



대표적인 UART 장비들

- 시리얼 모뎀

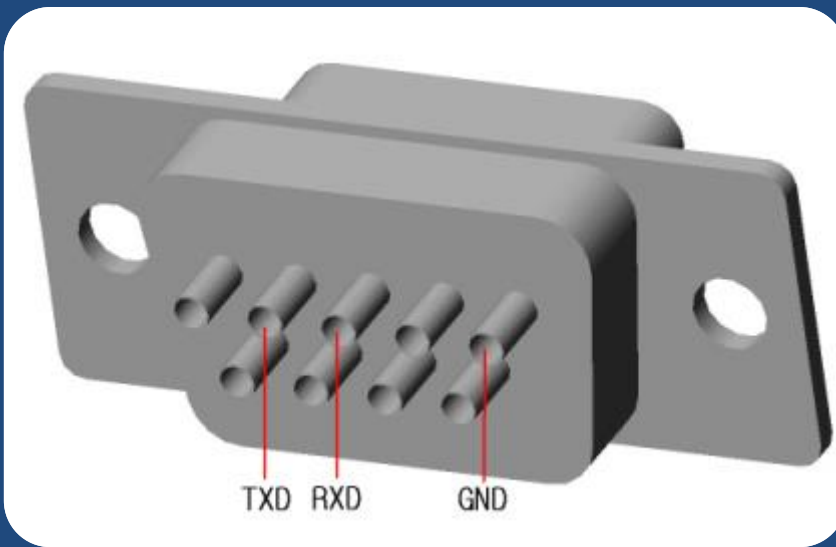


- 시리얼 마우스



UART<->PC 연결을 위한 장비

- 시리얼 포트, 케이블
 - OS 상에 COM(n)으로 잡힘
 - 하지만 요즘 PC, 노트북엔 시리얼 포트가 없음



12000원

UART<->USB 변환칩

- 주요 제품
 - FT232 (FTDI)
 - PL2303
 - CP2102

FTDI FT232RL
FT232시리즈의 SOPT타입의 버전



상품코드 5737
판매가격 5,300원 (부가세 미포함)
제조사 FTDI
적립금 44원
평균준비기간 1~2일
브랜드 FTDI[브랜드를바라가기]

수량

바로구매  장바구니  관심상품

[🔍 큰이미지 보기](#)

- USB to Serial 장비 안에 존재
- 각각에 맞는 드라이버 사용 필요

전기 신호 분석하기

(UART 프로토콜 분석)

Logic Analyzer 소개

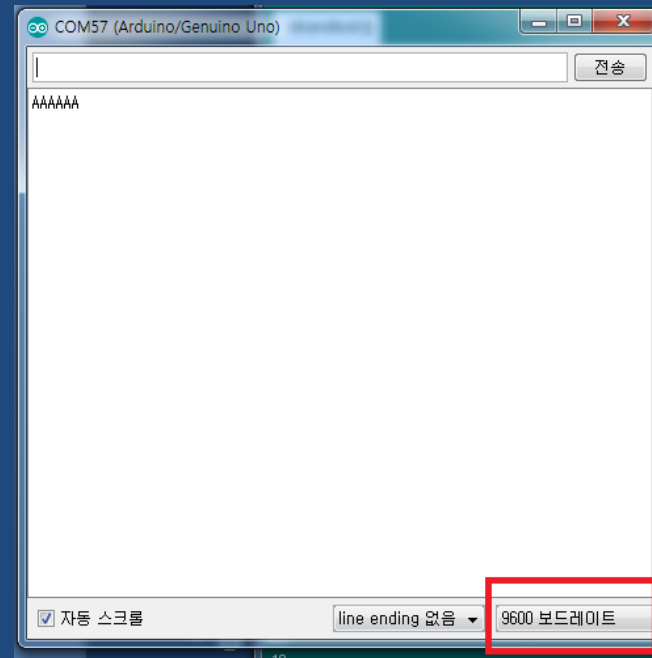
- 전기 신호를 캡처하여 0과 1로 보여주는 장비



UART 신호 분석 실습

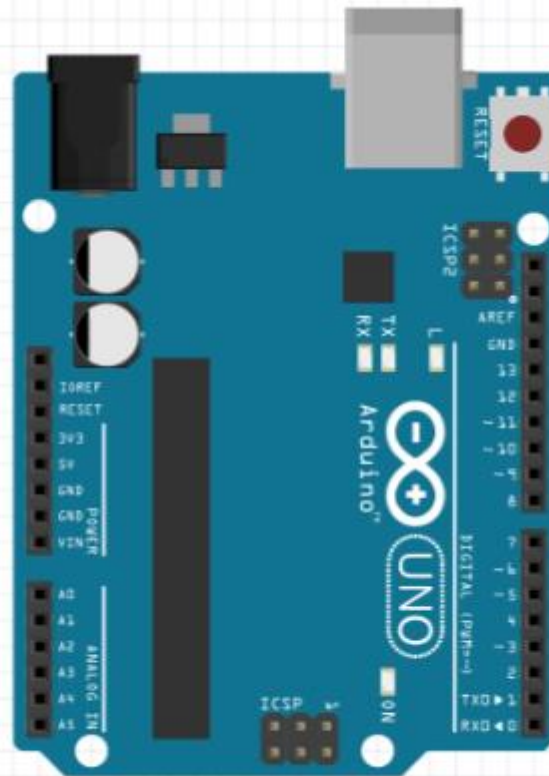
아두이노 UART 프로그래밍

```
void setup() {  
  Serial.begin(9600);  
}  
  
void loop() {  
  Serial.print("A");  
  delay(1000);  
}
```



UART의 TX 핀 스니핑

MCU : Atmega 328
Input voltage : 7V-12V
Operating voltage : 5V
CPU Speed : 16MHZ
Analog In/Out : 6/0
Digital IO/PWM : 14/6
EEPROM : 1KB
SRAM : 2KB
Flash : 32KB
UART : 1
USB : Regular



ARDUINO PIN	MICROCONTROLLER PIN
0	PD0(RXD)
1	PD1(TXD)
2	PD2(INT0)
3	PD3(INT1)
4	PD4
5	PD5
6	PD6
7	PD7
8	PB0
9	PB1
10	PB2(SS ⁺)
11	PB3(MOSI)
12	PB4(MISO)
13	PB5(SCK)
A0	PC0
A1	PC1
A2	PC2
A3	PC3
A4	PC4(SDA)
A5	PC5(SCL)

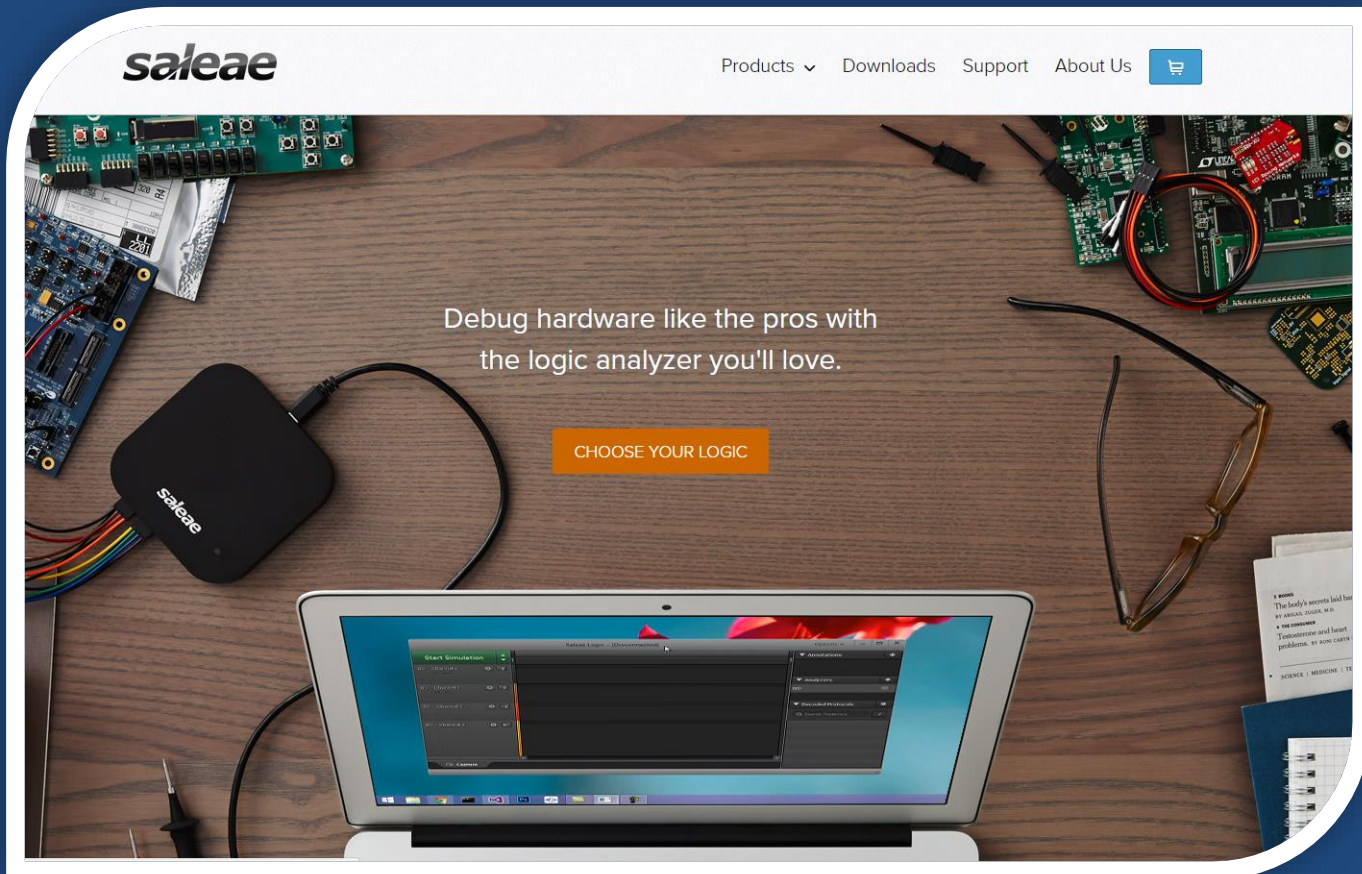
Logic Analyzer 연결

- GND = GND
- CH0 = TX



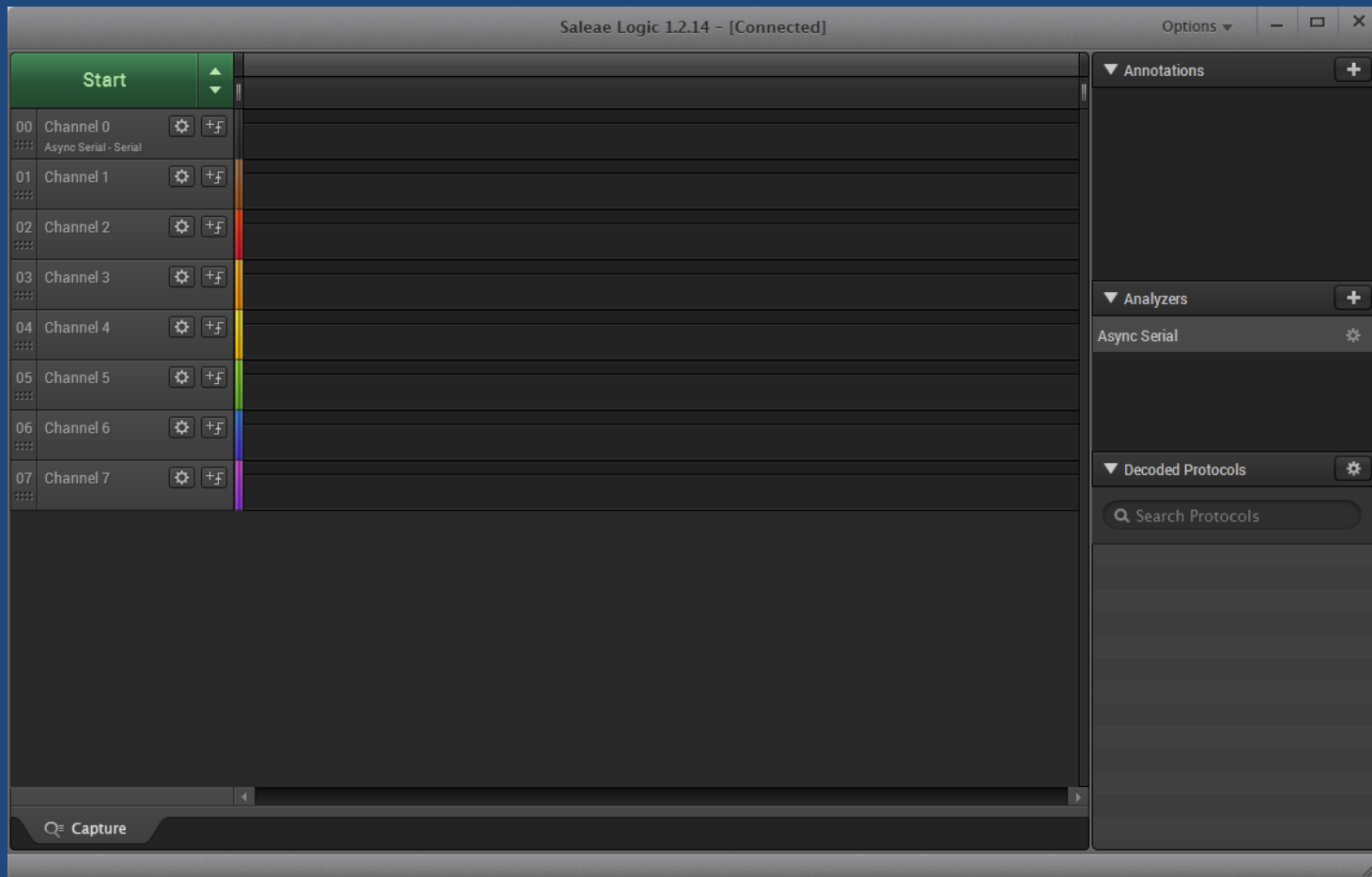
Saleae Logic Analyzer 설치

- <https://www.saleae.com/>



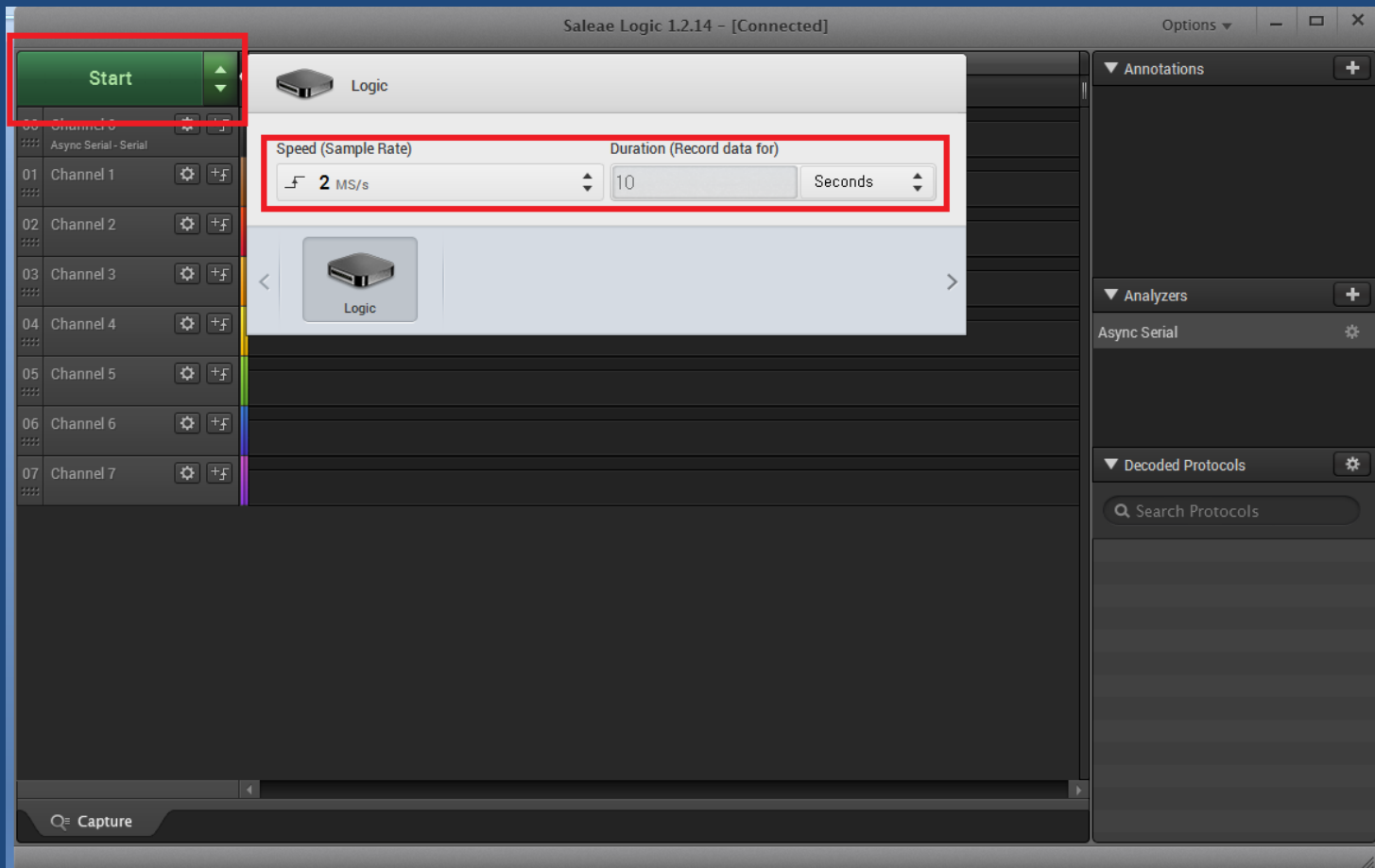
Logic Analyzer 실행

- “Connected” 확인

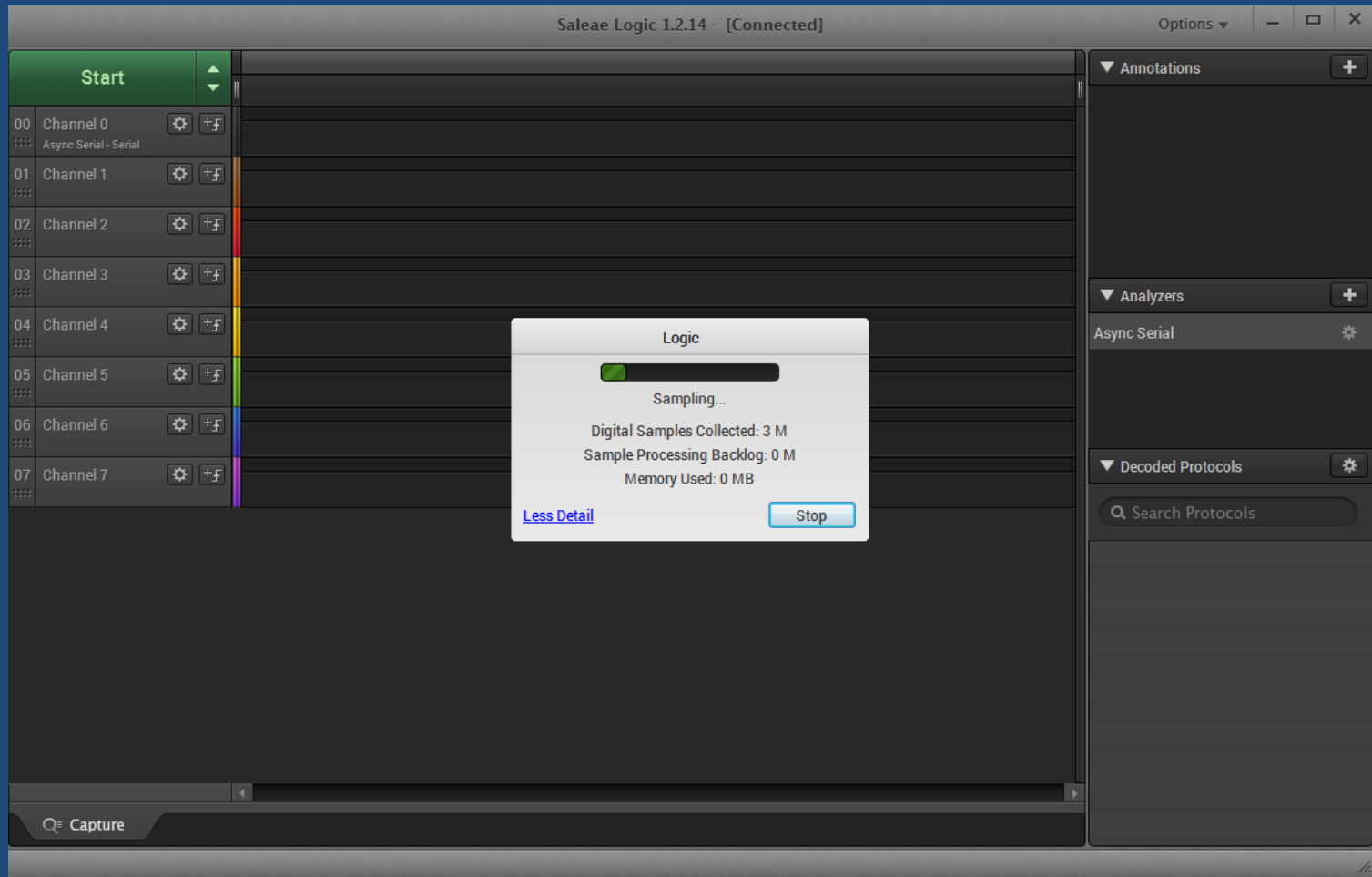


신호 분석 시작

- Speed, duration 설정 후 Start 버튼 클릭

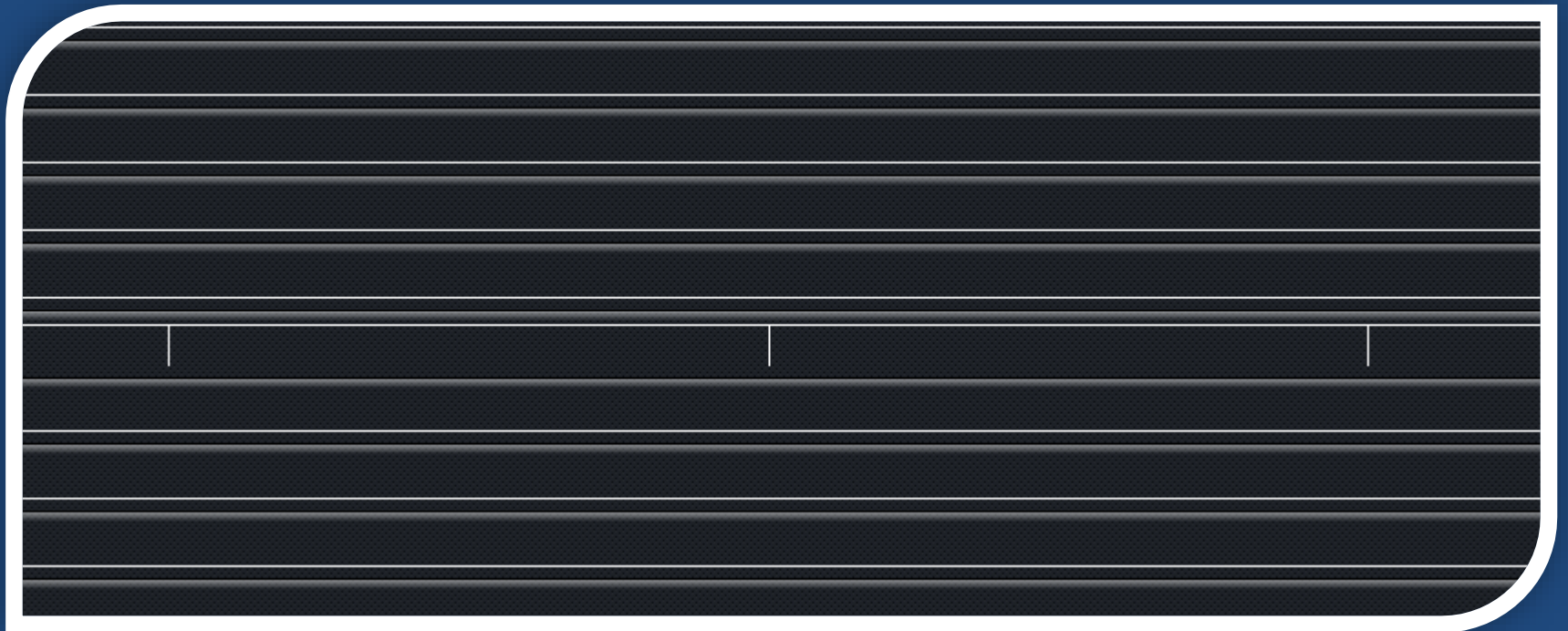


신호 분석 시작



UART의 TX 핀 스니핑 결과

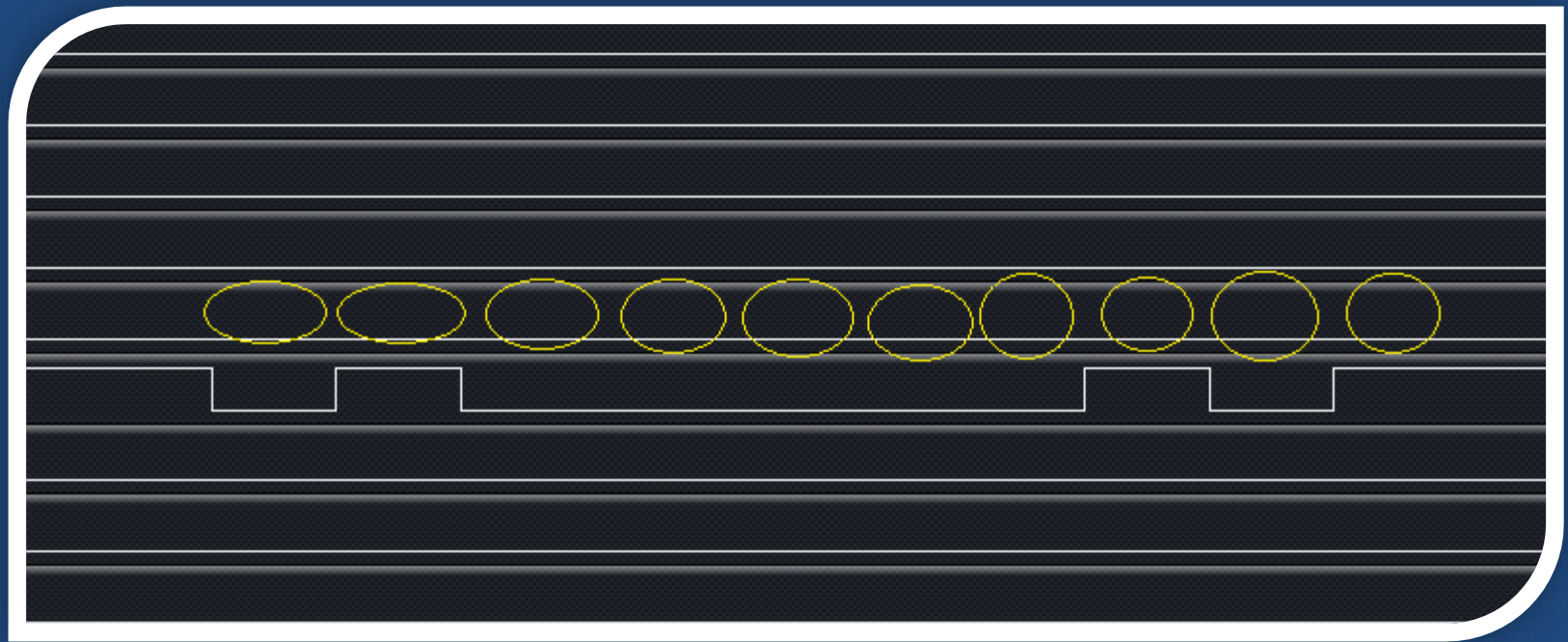
- 1초 간격의 신호 반응을 볼 수 있음
- 보이지 않으면 마우스휠(혹은 키보드 +/-)로 조정



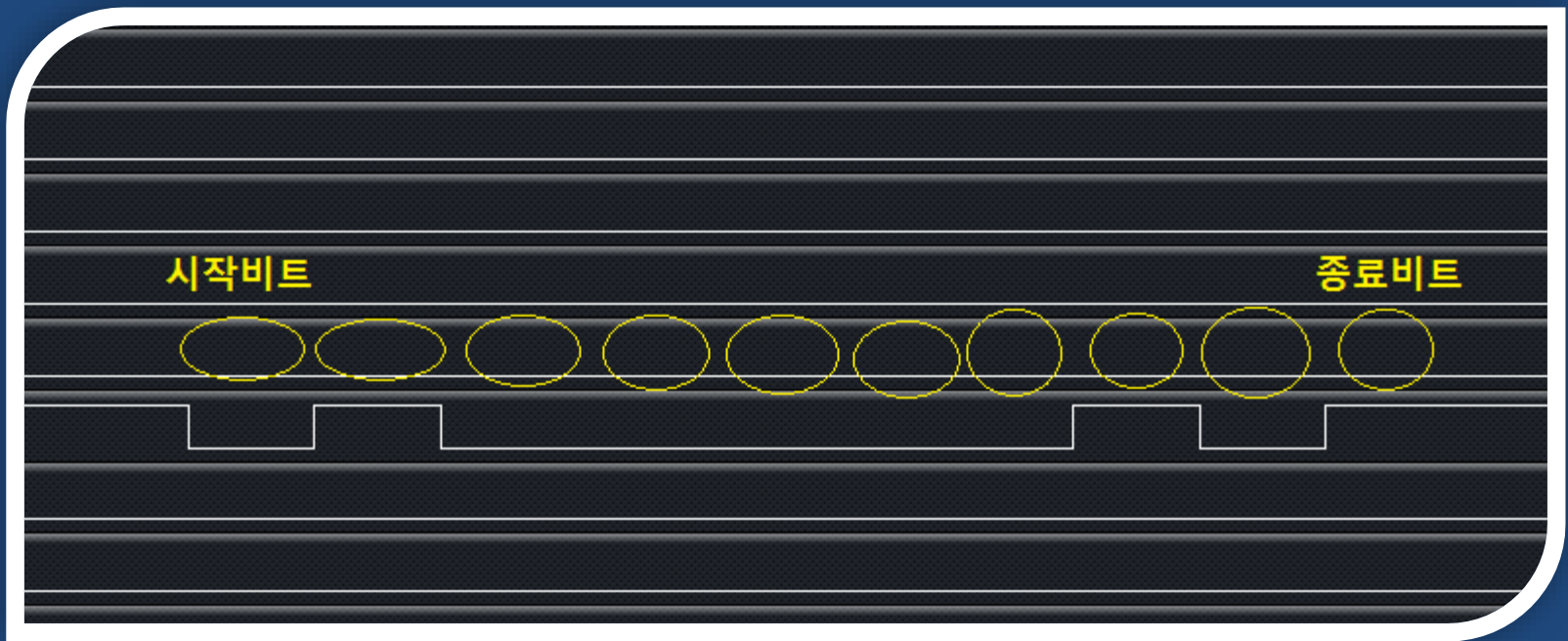
마우스휠을 이용하여 확대



신호를 비트 단위로 구분

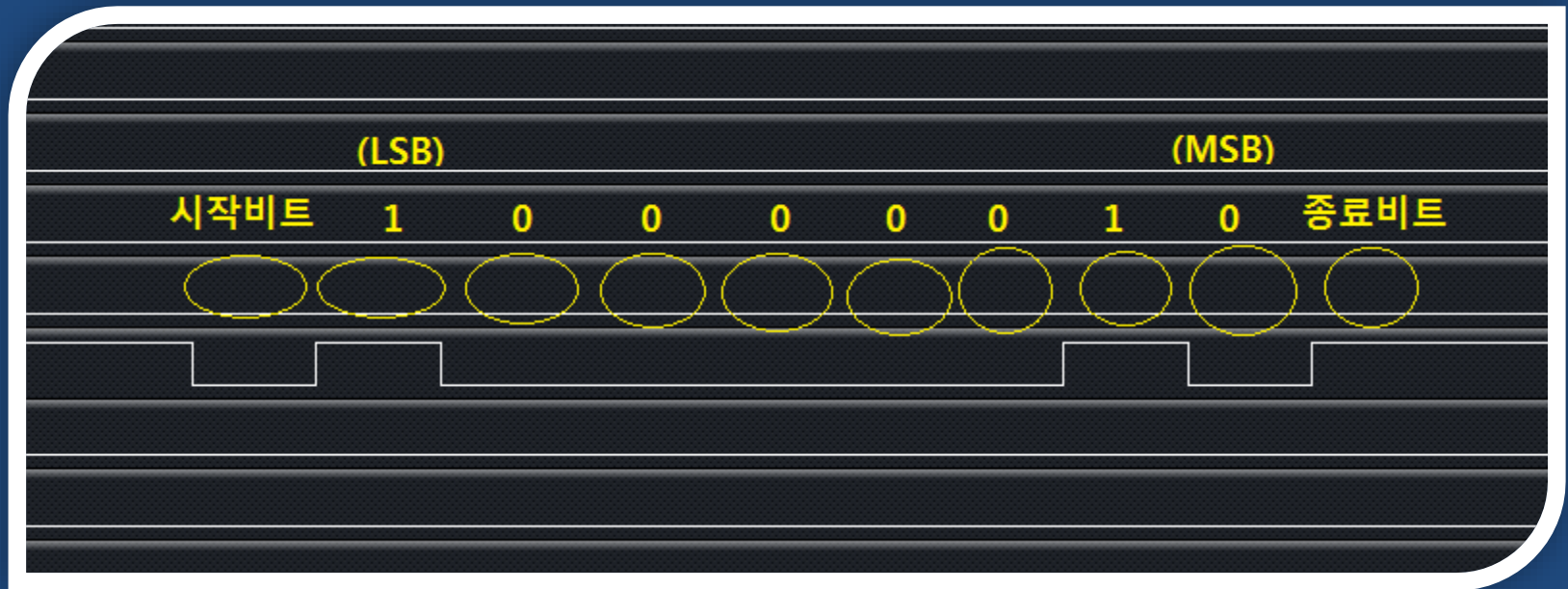


시작 비트와 종료 비트

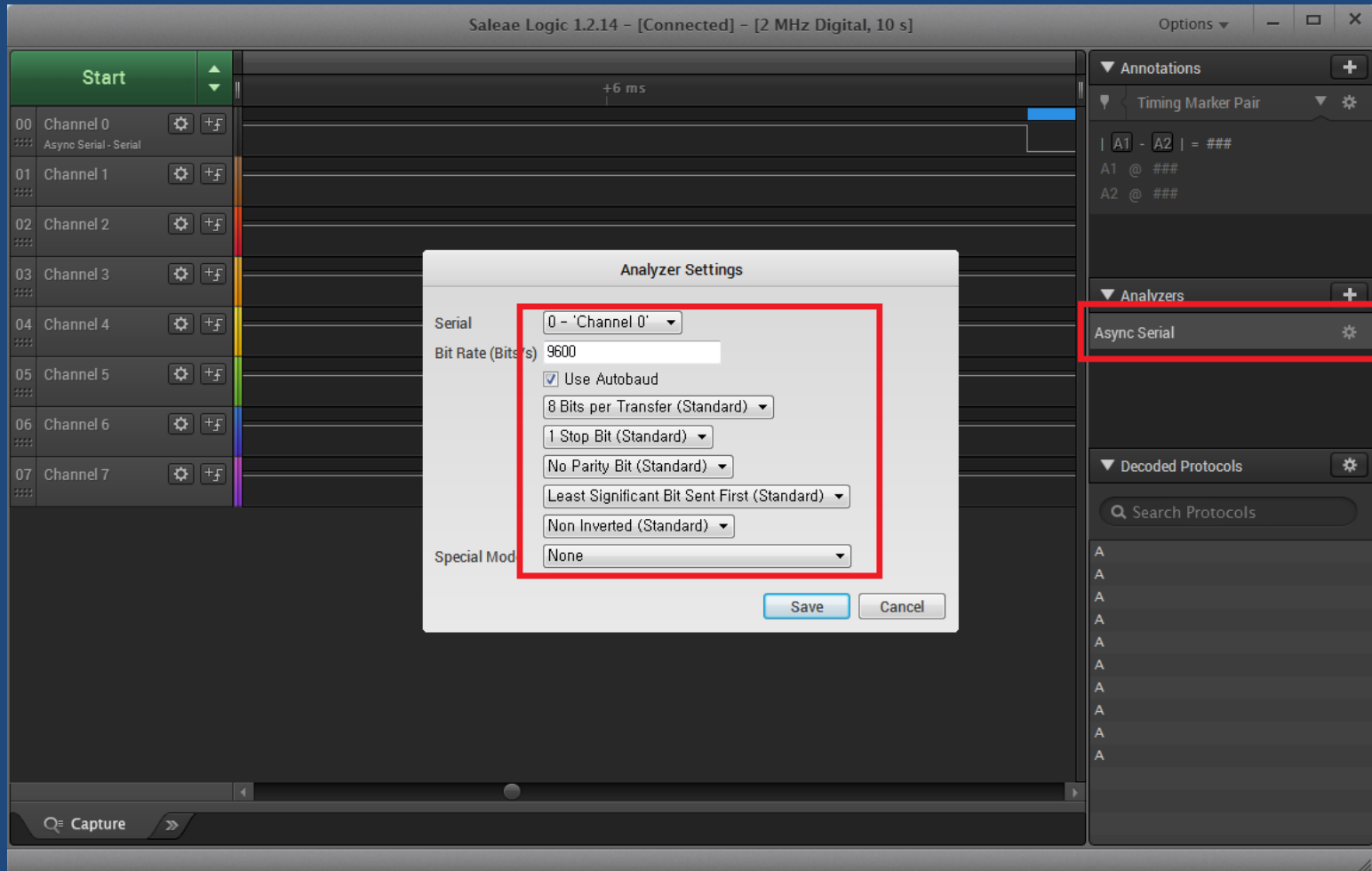


데이터 비트 분석

- $0b01000001 = 0x41 = \text{'A'}$

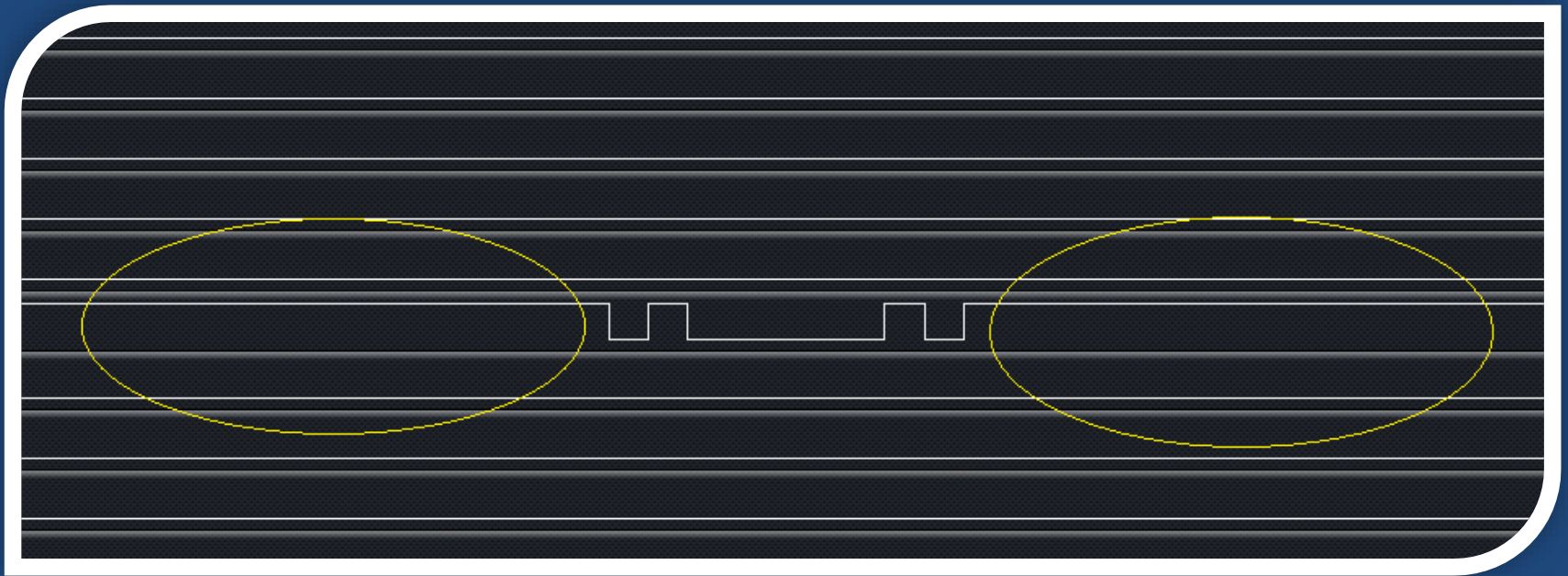


설정 변경이 필요한 경우



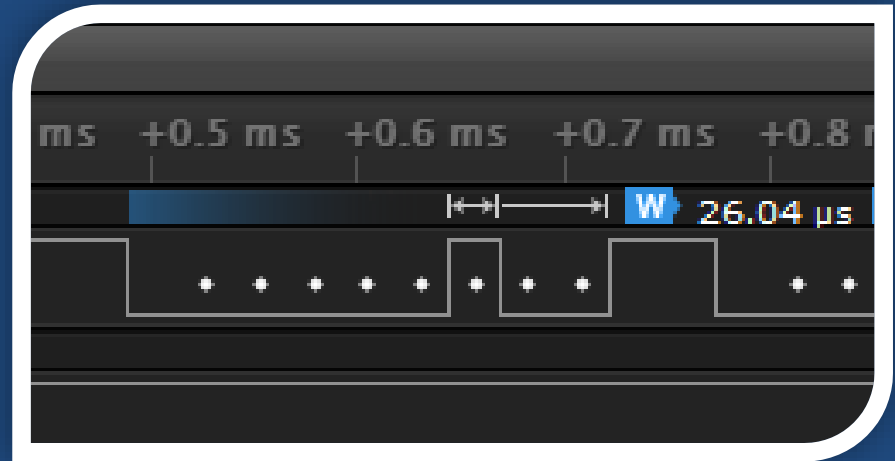
TX 핀의 기본 상태는 HIGH다.

- TX 핀과 GND에 LED를 연결했을 때 빛이 나는 이유



신호 분석을 통해 baudrate 알아내기

- 9600 : 100 μ s
- 14400 : 69 μ s
- 38400 : 26 μ s
- 57600 : 17 μ s
- 115200 : 8.6 μ s



스마트폰 UART 해킹 실습

스마트폰 JIG

- JIG란?

- 다른 무언가를 연결하기 위한 틀

지그는 영어의 "jig" 입니다. 때로는 "치구"라고 불려지기도 합니다. 아마도 영문의 Jig를 일본식 발음으로 만들어져 전해진것이 아닐까 합니다. 한국어로 하자면 "틀"이라고 할수 있겠지요.

반복 되는 기계 혹은 목공 공작을 할때, Jig를 사용하여 반복 된 작업을 하게 됩니다.

가령 예를 들자면, 같은 규격의 여러 금속체에 같은 좌표/장소에 드릴 프레스로 구멍을 뚫어야 할 경우, 일정 금속체에 맞는 Jig를 만들어 비이스에 걸어 놓고, 금속체를 Jig에 고정하여 금속을 가공하게 되면, 최소한의 오차로 반복적인 가공을 할수 있겠죠? 이럴때 사용하는것이 Jig 입니다.



오른쪽의 사진은 비행기 프로펠러 제작시, 장착 마운트의 구멍을 일정하게 뚫기 위해, Jig (여러개의 구멍이 있는 알루미늄 금속체)를 이용하여 반복 작업을 하는것 이지요. 기계 공작에서도 똑 같은 방법이 적용 됩니다.

도움 되셨길 바랍니다.

스마트폰 JIG

- 제한된 usb 핀들로 여러가지 기능을 구현
 - 충전
 - Usb OTG(On-the-go)
 - Audio Dock
 - Download mode
 - 그리고 UART
- USB 핀의 저항값을 체크하여 구분

JIG의 활용성

- 루팅 카운트 초기화
 - 루팅 카운트가 0이 아닐 경우 A/S에 지장
- UART 모드 부팅
- 강제 다운로드 모드 부팅
- Audio Dock 모드 Enable

갤럭시S의 JIG 기능

- 마이크로 USB 포트의 저항값을 변경하여 숨겨진 기능 사용 가능
- <http://forum.xda-developers.com/showthread.php?t=820275>

```
Code:
RID_USB_OTG_MODE,          /* 0 0 0 0 0   GND          USB OTG Mode          */
RID_AUD_SEND_END_BTN,     /* 0 0 0 0 1   2K           Audio Send_End Button*/
RID_AUD_REMOTE_S1_BTN,    /* 0 0 0 1 0   2.604K      Audio Remote S1 Button */
RID_AUD_REMOTE_S2_BTN,    /* 0 0 0 1 1   3.208K      Audio Remote S2 Button
RID_AUD_REMOTE_S3_BTN,    /* 0 0 1 0 0   4.014K      Audio Remote S3 Button */
RID_AUD_REMOTE_S4_BTN,    /* 0 0 1 0 1   4.82K       Audio Remote S4 Button */
RID_AUD_REMOTE_S5_BTN,    /* 0 0 1 1 0   6.03K       Audio Remote S5 Button */
RID_AUD_REMOTE_S6_BTN,    /* 0 0 1 1 1   8.03K       Audio Remote S6 Button */
RID_AUD_REMOTE_S7_BTN,    /* 0 1 0 0 0   10.03K      Audio Remote S7 Button */
RID_AUD_REMOTE_S8_BTN,    /* 0 1 0 0 1   12.03K      Audio Remote S8 Button */
RID_AUD_REMOTE_S9_BTN,    /* 0 1 0 1 0   14.46K      Audio Remote S9 Button */
RID_AUD_REMOTE_S10_BTN,   /* 0 1 0 1 1   17.26K      Audio Remote S10 Button */
RID_AUD_REMOTE_S11_BTN,   /* 0 1 1 0 0   20.5K       Audio Remote S11 Button */
RID_AUD_REMOTE_S12_BTN,   /* 0 1 1 0 1   24.07K      Audio Remote S12 Button */
RID_RESERVED_1,          /* 0 1 1 1 0   28.7K       Reserved Accessory #1 */
RID_RESERVED_2,          /* 0 1 1 1 1   34K         Reserved Accessory #2 */
RID_RESERVED_3,          /* 1 0 0 0 0   40.2K       Reserved Accessory #3 */
RID_RESERVED_4,          /* 1 0 0 0 1   49.9K       Reserved Accessory #4 */
RID_RESERVED_5,          /* 1 0 0 1 0   64.9K       Reserved Accessory #5 */
RID_AUD_DEV_TY_2,        /* 1 0 0 1 1   80.07K      Audio Device Type 2 */
RID_PHONE_PWD_DEV,       /* 1 0 1 0 0   102K        Phone Powered Device */
RID_TTY_CONVERTER,       /* 1 0 1 0 1   121K        TTY Converter */
RID_UART_CABLE,          /* 1 0 1 1 0   150K        UART Cable */
RID_CEA936A_TY_1,        /* 1 0 1 1 1   200K        CEA936A Type-1 Charger(1) */
RID_FM_BOOT_OFF_USB,     /* 1 1 0 0 0   255K        Factory Mode Boot OFF-USB */
RID_FM_BOOT_ON_USB,      /* 1 1 0 0 1   301K        Factory Mode Boot ON-USB */
RID_AUD_VDO_CABLE,       /* 1 1 0 1 0   365K        Audio/Video Cable */
RID_CEA936A_TY_2,        /* 1 1 0 1 1   442K        CEA936A Type-2 Charger(1) */
RID_FM_BOOT_OFF_UART,    /* 1 1 1 0 0   523K        Factory Mode Boot OFF-UART */
RID_FM_BOOT_ON_UART,     /* 1 1 1 0 1   619K        Factory Mode Boot ON-UART */
RID_AUD_DEV_TY_1_REMOTE, /* 1 1 1 1 0   1000.07K    Audio Device Type 1 with F
RID_AUD_DEV_TY_1_SEND = RID_AUD_DEV_TY_1_REMOTE , /* 1 1 1 1 0   1002K
RID_USB_MODE              /* 1 1 1 1 1   00000000   USB Mode_Dedicated Charger or App
```

FSA9480 USB switch chip

FAIRCHILD
SEMICONDUCTOR®



FSA9480 — USB 2.0 Accessory Switch

FSA9480 — USB 2.0 Accessory Switch

Features

- Automatically Detects USB Accessories:
 - USB OTG Mode
 - CEA-936-A Car Kit and Chargers
 - Headsets
 - Video Cable
 - Factory Mode Cables
 - **UART**
 - TTY Converter
 - USB Data Cable
 - Chargers
- Auto-configures Connections with Independent Override Capability
- Integrated Audio Amplifier Generates Required Bias for CEA-936-A Car Kit Audio
- Automatic Low-Power Mode When No Accessory is Attached
- Integrated Over-Voltage and Over Current Protection FET on V_{BUS} for Fault Isolation
- Negative-Swing-Capable Audio Channel

Description

The FSA9480 is a USB port accessory detector and switch. The FSA9480 is fully controlled using I²C™ and enables USB data, stereo and mono audio, video, microphone, and UART data to use a common connector port. It is designed for compatibility with CEA-936-A car kit adapters, USB 2.0 signaling, and USB OTG (on-the-go). The architecture is designed to allow audio signals to swing below ground so a common USB and headphone jack can be used for personal media players and portable peripheral devices.

The FSA9480 meets USB specification Rev. 2.0, micro-USB specification, and CEA-936-A.

FSA9480

Block Descriptions

This block is used for factory-mode test and debug as described in Table 1.

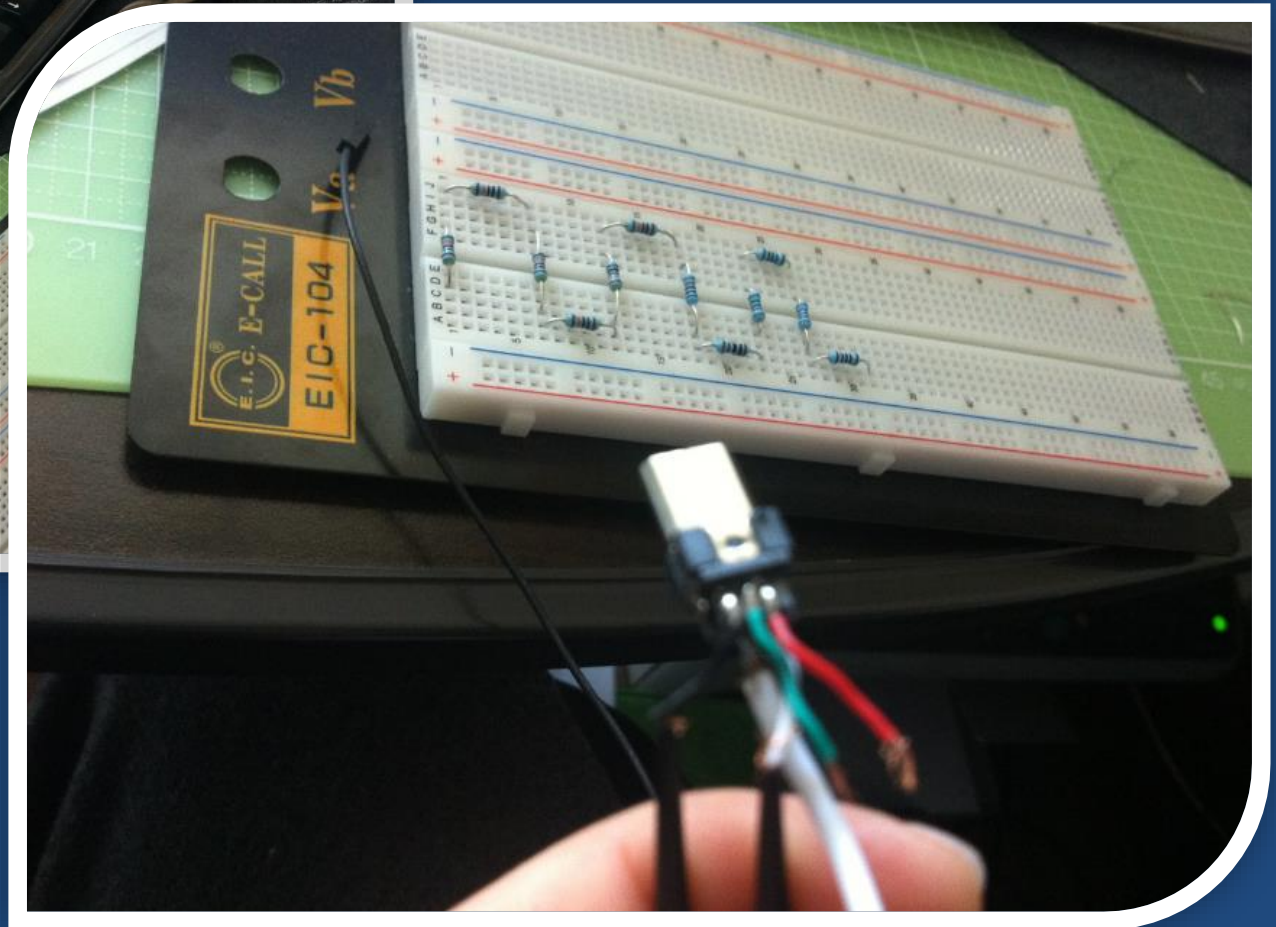
Table 1. Boot Table

Factory Mode	ID Resistor	JIG	BOOT	Switch Connections
UART, Boot Off	523K Ω	LOW	LOW	Table 3 - Accessory Auto Configuration Table
USB, Boot On	301K Ω	LOW	HIGH	Table 3 - Accessory Auto Configuration Table
USB, Boot Off	255K Ω	LOW	LOW	Table 3 - Accessory Auto Configuration Table
UART, Boot On	619K Ω	LOW	HIGH	Table 3 - Accessory Auto Configuration Table
No Factory Mode	N/A	Hi-Z	LOW	See Notes 1 and 2

Notes:

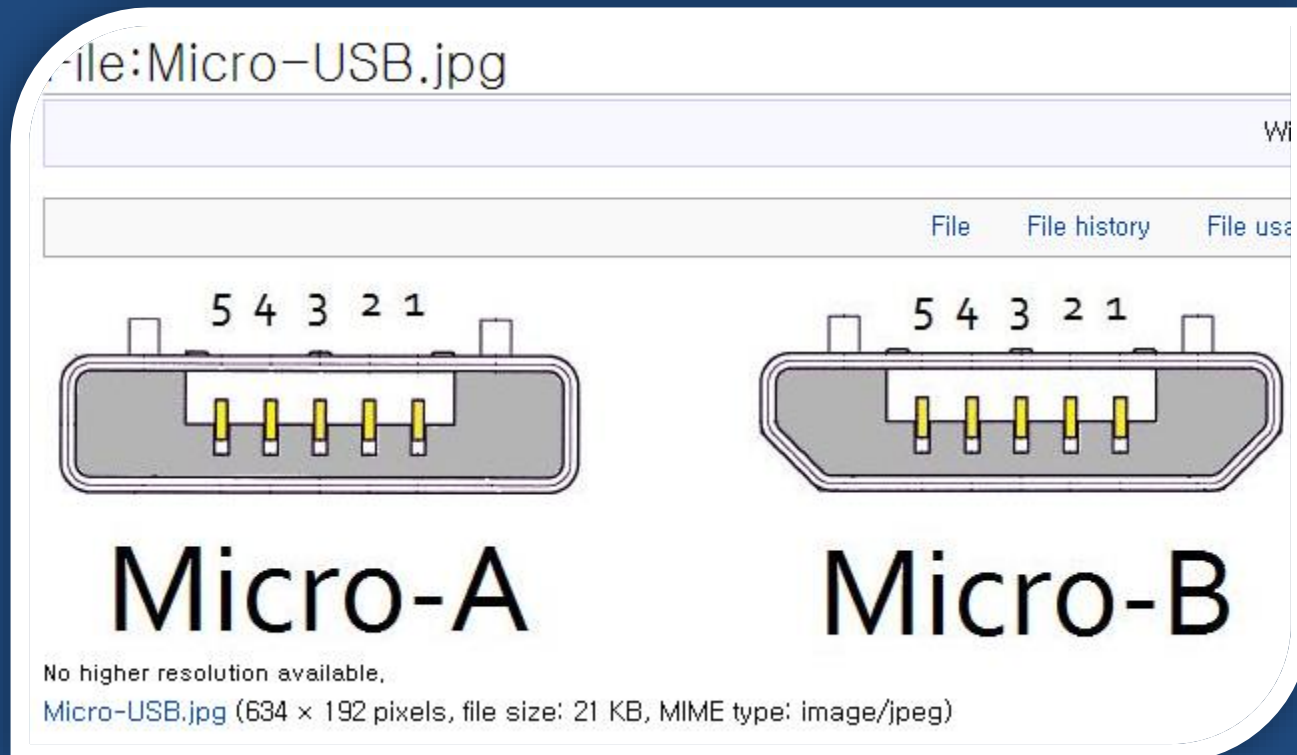
1. JIG pin description: Low signals the phone to power up. When disabled this open drain floats and the pin is Hi-Z
2. BOOT pin description: This boot pin level directs the baseband processor which mode to boot in when the JIG box is attached, the JIG pin has been driven LOW, and the phone is powered up.

JIG Cable 만들기

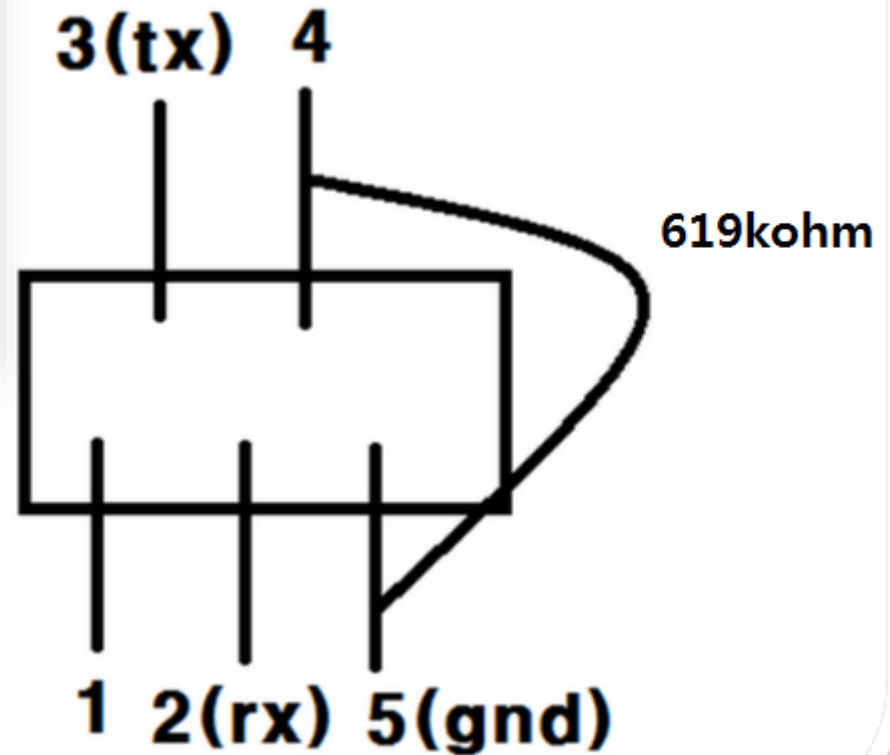
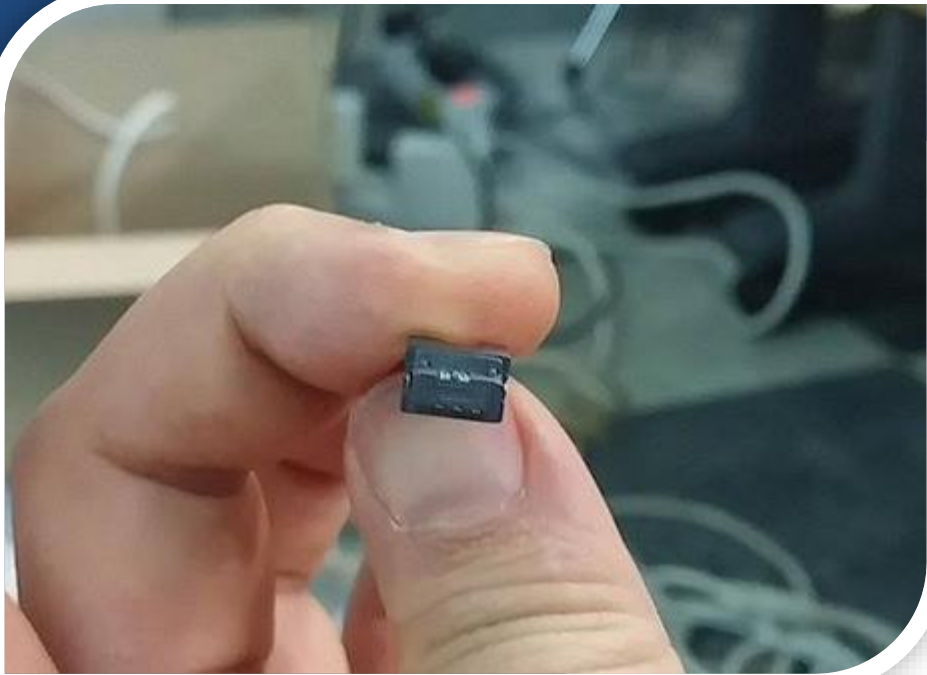


Micro USB 핀 연결 방법

- <http://commons.wikimedia.org/wiki/File:Micro-USB.jpg>



Micro USB 핀 연결 방법



Micro USB 커넥터

- <http://devicemart.co.kr/goods/view.php?seq=29454>

MUYM-051-1BSX0



상품코드 **29454**
판매가격 **530원** (부가세 미포함)
제조사 OEM
적립금 0원
평균준비기간 3~4일
브랜드 OEM[브랜드를바라가기]








수량

[큰이미지 보기](#)

[바로구매](#) [장바구니](#) [관심상품](#)

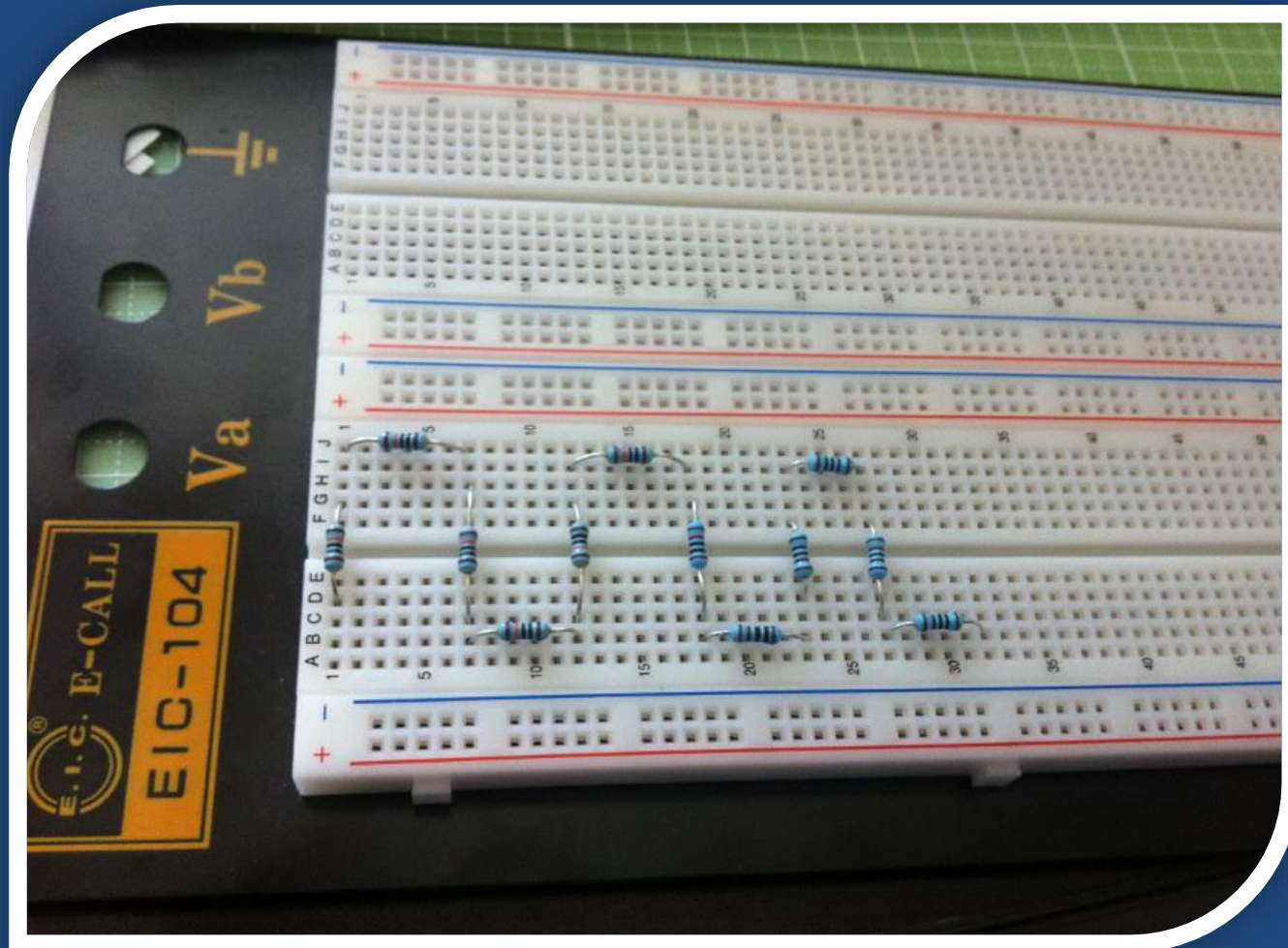
크기별 저항

- <http://devicemart.co.kr/goods/view.php?seq=1963>

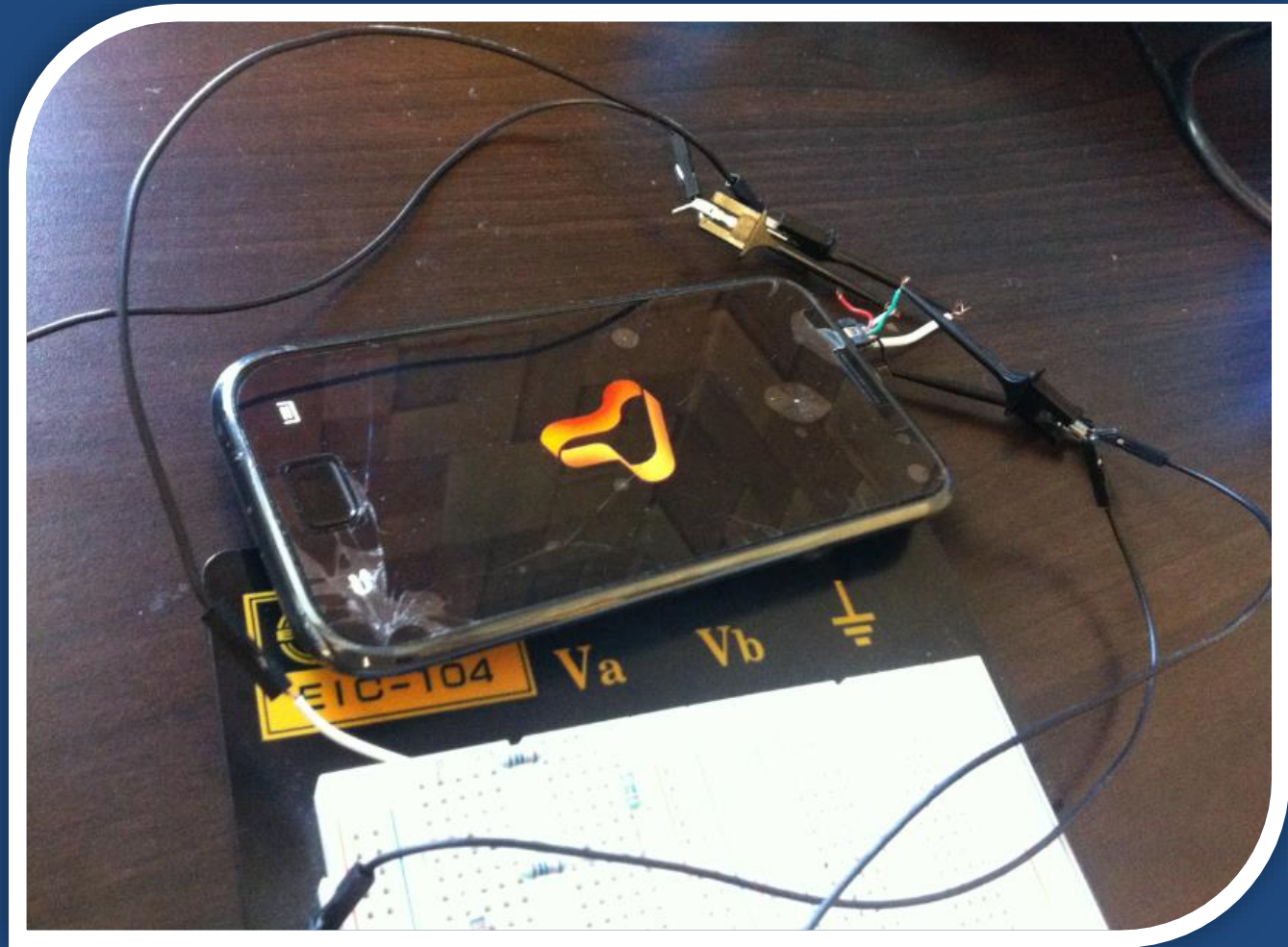
선택	삭제	상품명	가격 (부가세 포함)	수량	합계	배송비	삭제
<input checked="" type="checkbox"/>		1/4W 1% Axial Resistor 105F (1M Ω) 옵션 : 10개	220원 적 0	<input type="text" value="1"/>	220원	기본배송	삭제
<input checked="" type="checkbox"/>		1/4W 1% Axial Resistor 104F (100K Ω) 옵션 : 10개	220원 적 0	<input type="text" value="1"/>	220원	기본배송	삭제
<input checked="" type="checkbox"/>		1/4W 1% Axial Resistor 103F (10K Ω) 옵션 : 10개	220원 적 0	<input type="text" value="1"/>	220원	기본배송	삭제
<input checked="" type="checkbox"/>		1/4W 1% Axial Resistor 102F (1K Ω) 옵션 : 10개	220원 적 0	<input type="text" value="1"/>	220원	기본배송	삭제
<input checked="" type="checkbox"/>		1/4W 1% Axial Resistor 101F (100 Ω) 옵션 : 10개	220원 적 0	<input type="text" value="1"/>	220원	기본배송	삭제
<input checked="" type="checkbox"/>		1/4W 1% Axial Resistor 100F (10 Ω) 옵션 : 10개	220원 적 0	<input type="text" value="1"/>	220원	기본배송	삭제
<input checked="" type="checkbox"/>		1/4W 1% Axial Resistor 1R0F (1 Ω) 옵션 : 10개	220원 적 0	<input type="text" value="1"/>	220원	기본배송	삭제

총구매금액 : 4,040원 (상품가격 1,400원 + 부가세 : 140 + 배송비 2,500원)
* 배송비는 할인가에 따라 조금씩 달라질수도 있습니다

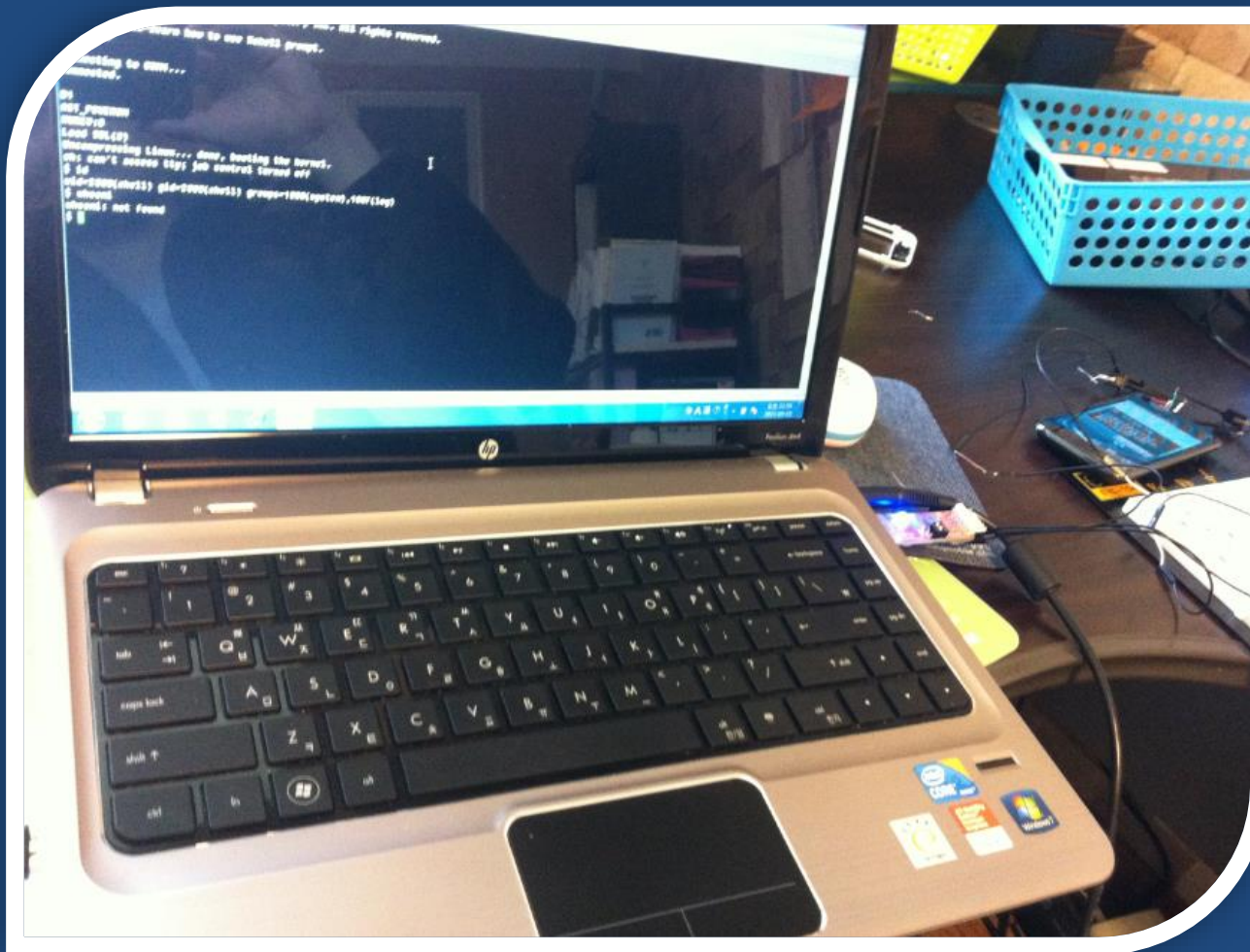
619Kohm 만들기



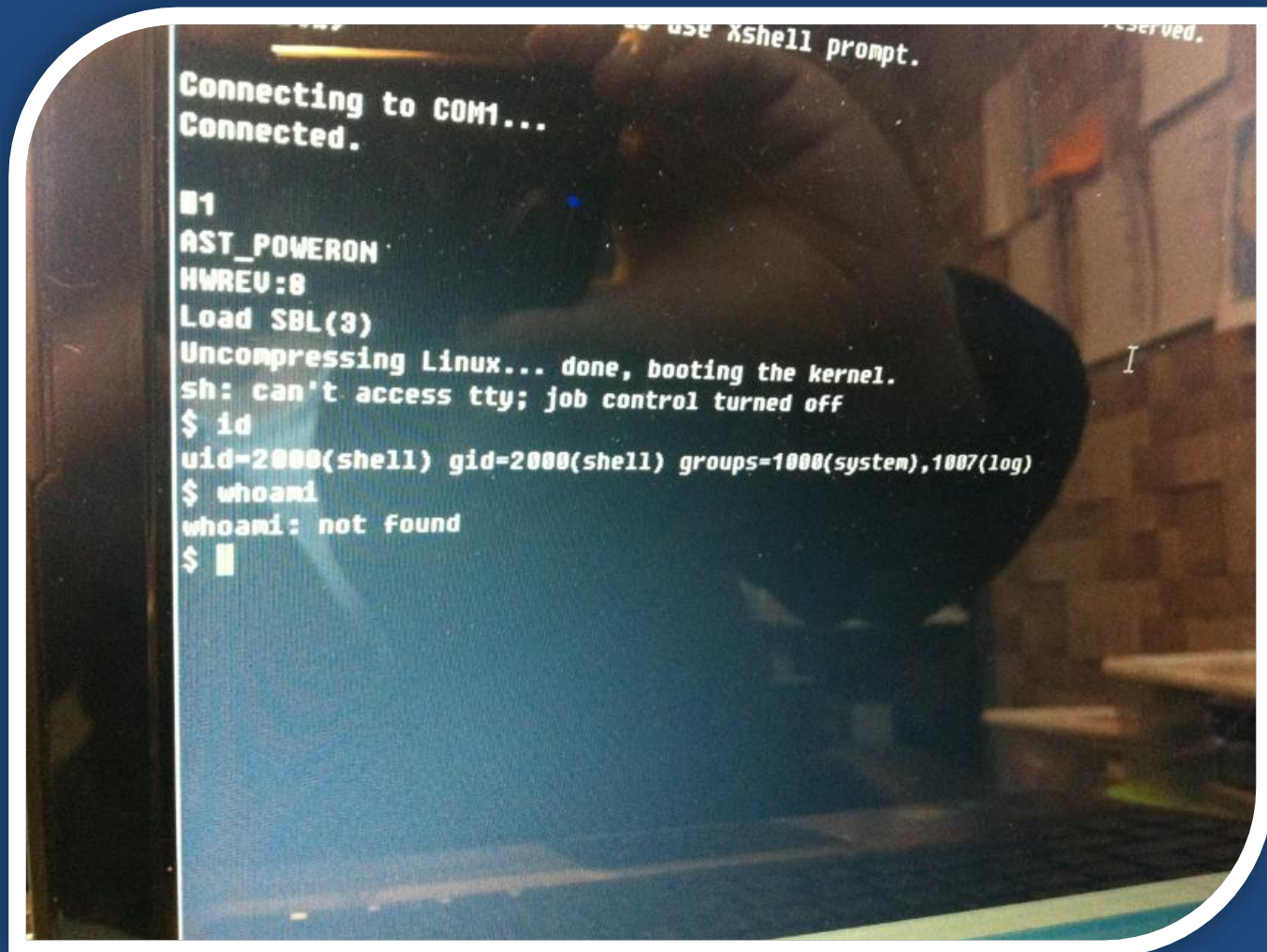
JIG 회로 연결



USB-UART 연결



Shell 확인

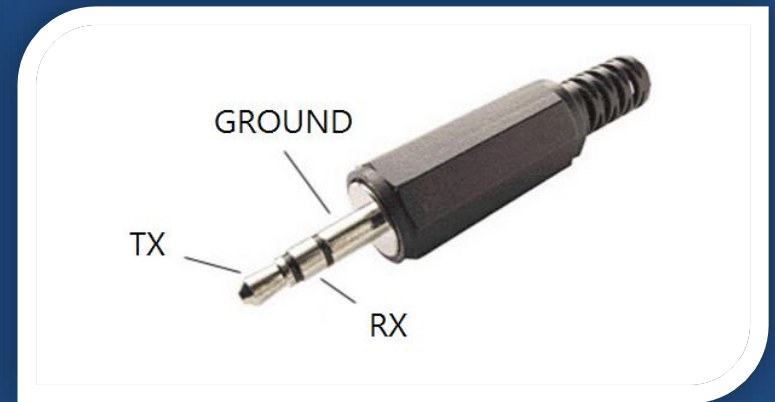


CPU에 바로 UART 연결하기

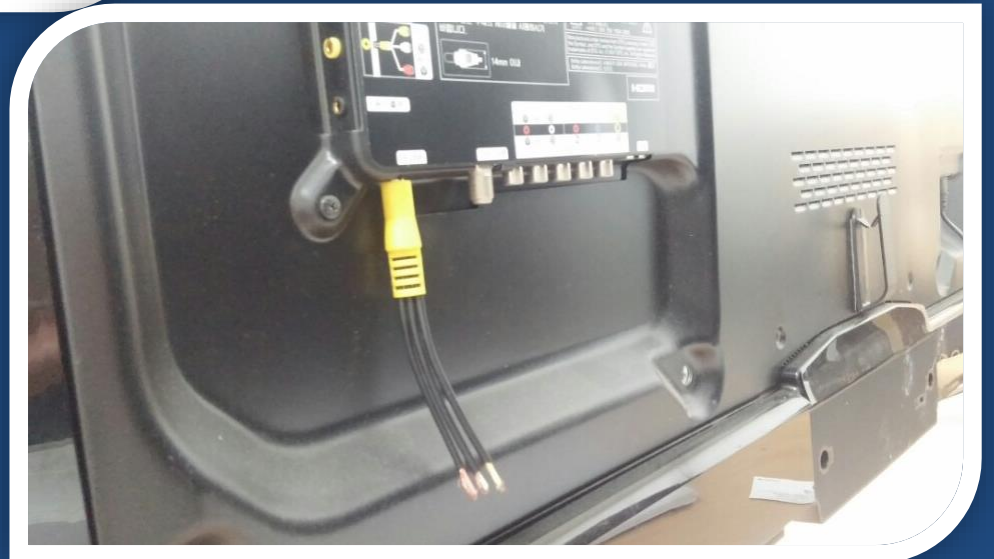
UART on Audio jack

UART on Audio jack

- UART 포트가 Audio Jack 형태인 경우가 있음

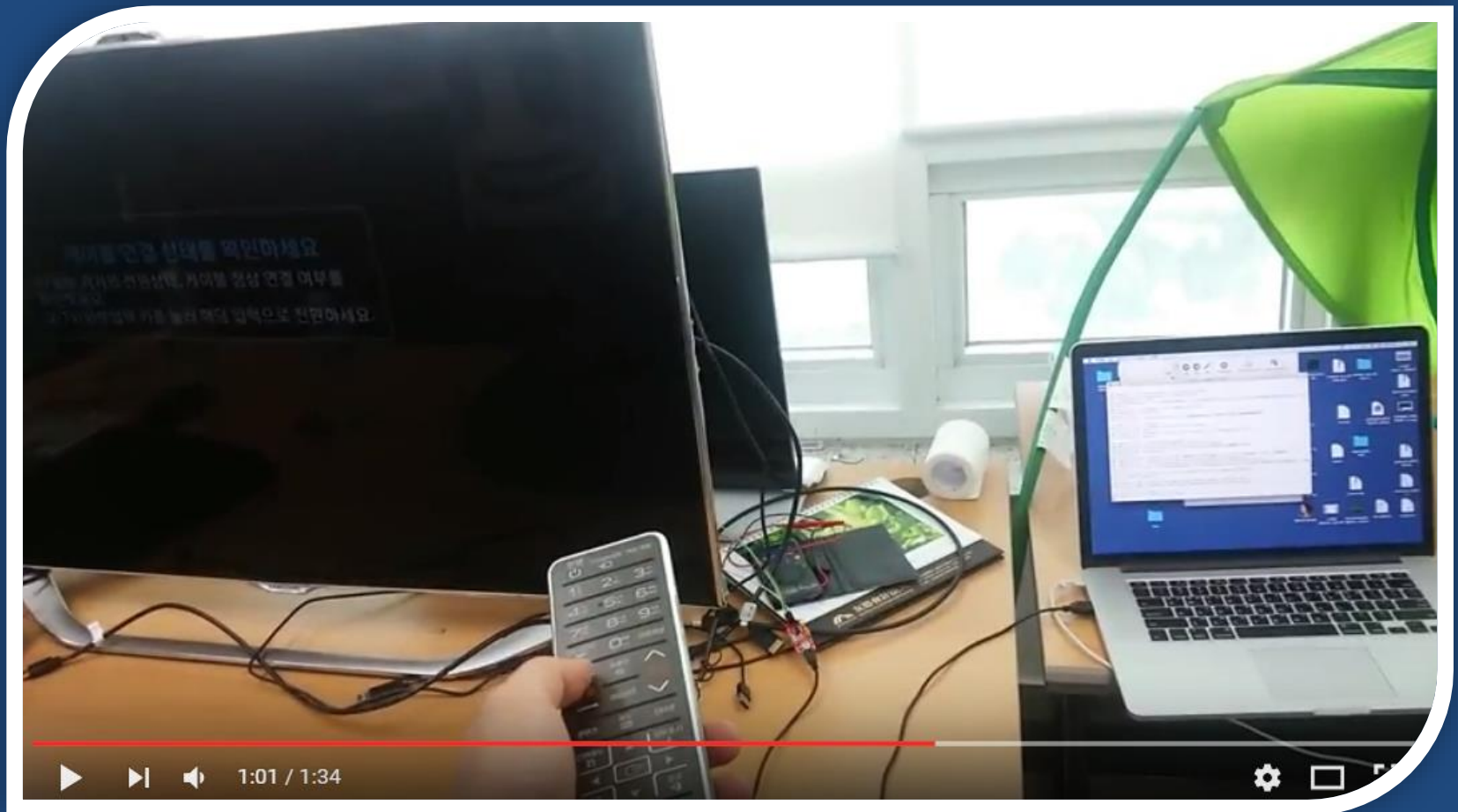


Audio Jack UART 예제

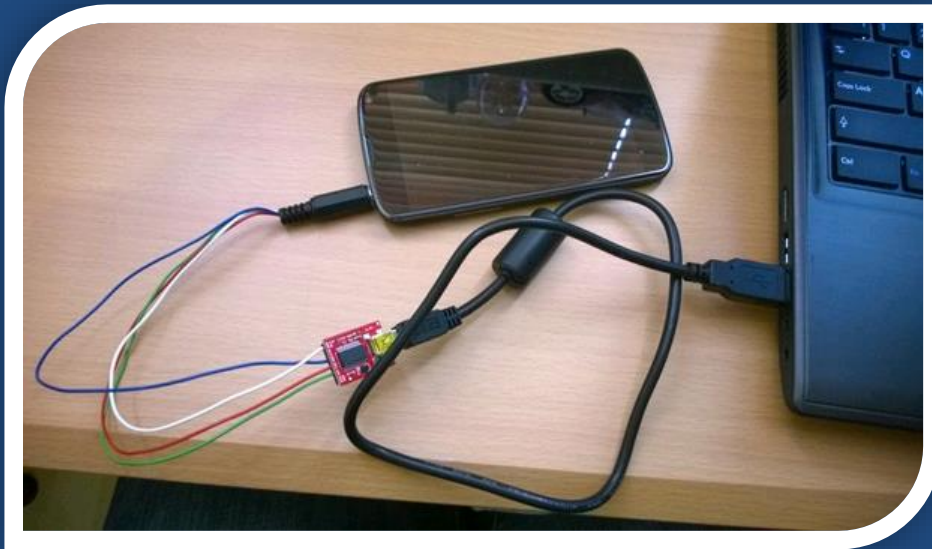


동영상

- <https://www.youtube.com/watch?v=NHawn8XvVQE>



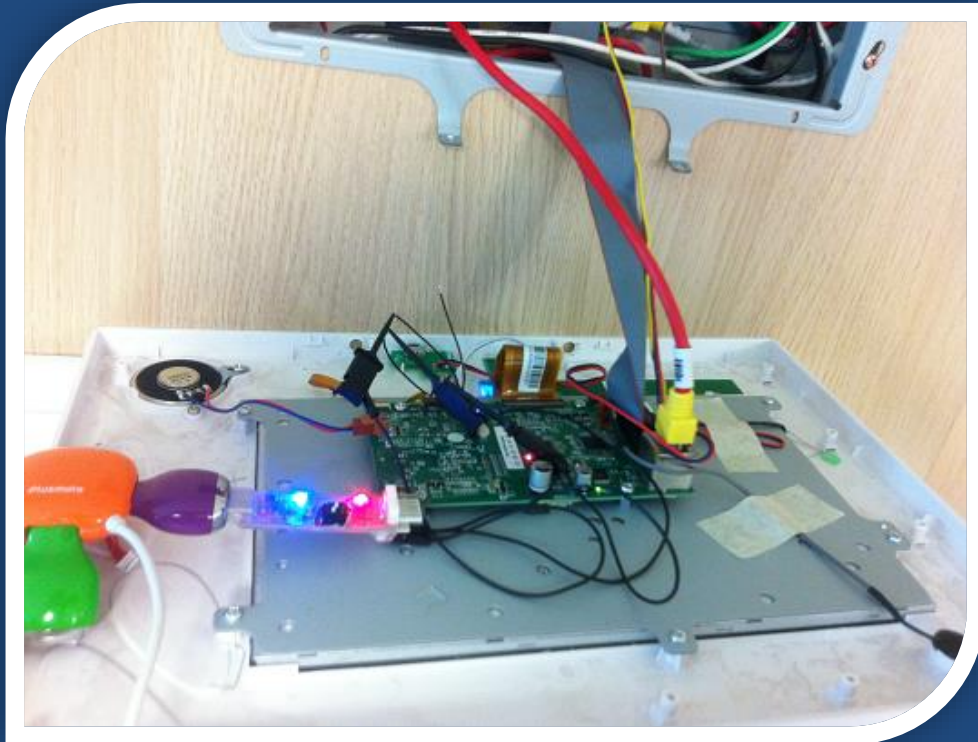
Nexus UART on Audio jack



```
welcome to mako bootloader
[90] cable type from shared memory: 8
[130] reboot_mode restart reason = reboot
[320] kernel @ 80208000 (5677280 bytes)
[330] ramdisk @ 81800000 (357803 bytes)
[330] get_display_kcal = 0, 0, 0, x
[330] Booting Linux
[340] Power on reason 65281
[340] Power on reason 65281
[340] booting linux @ 0x80208000, ramdisk @ 0x81800000 (357803)
[350] cmdline: console=ttyHSL0,115200,n8 androidboot.hardware=mako lpj=67677 uart_console=enable lcd_maker_id=primary
lge.hreset=off lge.reset=mode_reset gpt=enable lge.kcal=0|0|0|x lge.rev=rev_11 mdm_force_dump_enabled androidboot.emmc=true
androidboot.serialn[ 0.000000] Booting Linux on physical CPU 0
[ 0.000000] Initializing cgroup subsys cpu
[ 0.000000] Linux version 3.4.0-perf-g7ce11cd (android-build@vpbs1.mtv.corp.google.com) (gcc version 4.6.x-google 20120106 (prerelease)
(GCC) ) #1 SMP PREEMPT Tue Jan 29 11:41:33 PST 2013
[ 0.000000] CPU: ARMv7 Processor [510f06f2] revision 2 (ARMv7), cr=10c5387d
[ 0.000000] CPU: PIPT / VIPT nonaliasing data cache, PIPT instruction cache
[ 0.000000] Machine: QCT APQ8064 MAKO
[ 0.000000] Truncating memory at 0xc0000000 to fit in 32-bit physical address space
[ 0.000000] memory pool 3 (start fe9ff000 size 1600000) initialized
[ 0.000000] Initialized persistent memory from 88700000-887fffff
[ 0.000000] Memory policy: ECC disabled, Data cache writealloc
[ 0.000000] socinfo_init: v6, id=109, ver=1.1, raw_id=1817, raw_ver=1817, hw_plat=8, hw_plat_ver=65536
[ 0.000000] accessory_chip=0 hw_plat_subtype=1
```

월패드 UART 해킹 영상

<https://www.youtube.com/watch?v=usyakFpspKs>



UART 해킹 방어책

- Disable UART hardware port when product release
- Disable UART codes in the software
- Authenticate UART communication

결론

- UART는 하드웨어 해킹의 시작!
- 기기로부터 많은 것들을 얻어낼 수 있다!
- 운이 좋으면 Shell도 획득할 수 있다!

감사합니다.