

시작에 앞서서...

Smart HID 제품군 이란 ?

스마트기기 (삼성, 애플사의 스마트폰 및 스마트패드)와의 블루투스 통신이 가능하도록
HID Profile 을 내장시킨 펌테크 Bluetooth Embedded Module 제품군을 의미합니다.

FB155BC_SMD(HID) 소개

FB155BC_SMD(HID)는 **HID Profile**을 지원하는 **Bluetooth Embedded Module** 제품으로 현재 여러분이
주로 사용중인 유선방식의 USB 또는 PS2 방식의 Keyboard, Mouse 와 블루투스 통신을 통해 각종
스마트기기 (Samsung, Apple 사 등의 스마트폰, 스마트패드 등) 의 입력장치로 사용이 가능하도록
만들어 드립니다.

HID Profile 이란 ? → Human Interface Device 의 약자로 PC 등과의 사용자 인터페이스에 관련된 장치
즉 Keyboard, Mouse 등의 지원에 관련된 블루투스 프로파일을 의미합니다.

Smart HID 제품군 소개

Bluetooth Embedded Module Type



FB155BC_SMD(HID)
Class2 Type



FB155BC(HID)
Class2 Type



FB755AC (HID)
Class1 Type



FB155BS (HID)
Class2 Type



FB755AS (HID)
Class1 Type

Bluetooth Module Type



MD-4DR (HID)
Class2 Type



F1E22 (HID)
Class2 Type



M1-4DR (HID)
Class1 Type



HBG1X3N (HID)
Class1 Type

FB155BC_SMD(HID)

Application Guide

1. FB155BC_SMD(HID) 제품개요

HID Profile 지원



2.4 Ghz
블루투스

적용가능한 제품

- 삼성 Galaxy S
- 삼성 Gallsy Tab
- 애플 Iphone
- 애플 Ipad

• LG, 팬택등 HID 프로파일을
지원하는 모든 스마트폰

FB155BC_SMD(HID)는 HID Profile 을 지원하는 블루투스 임베디드 모듈 제품으로 여러분이 현재 사용중인 유선 방식의 USB 또는 PS/2 방식의 Keyboard, Mouse 와 PS/2 방식을 사용하는 다양한 제품군 (바코드 스캐너, 조이스틱, Etc) 을 블루투스 통신을 통해 각종 스마트기기 (스마트폰,패드 등)의 입력장치로 사용이 가능하도록 만들어 드립니다.

HID Profile 이란 ? → Human Interface Device 의 약자로 PC 등과의 사용자 인터페이스에 관련된 Keyboard, Mouse 등의 지원에 관련된 블루투스 프로파일을 의미합니다.

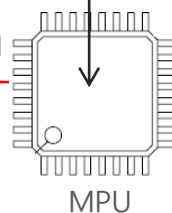
HID Profile 지원



FB155BC_SMD(HID)

PS/2 Protocol -> HID Protocol
변환 소프트웨어 실행파일 내장

HID Protocol



PS2 Protocol



Keyboard (USB or PS/2)

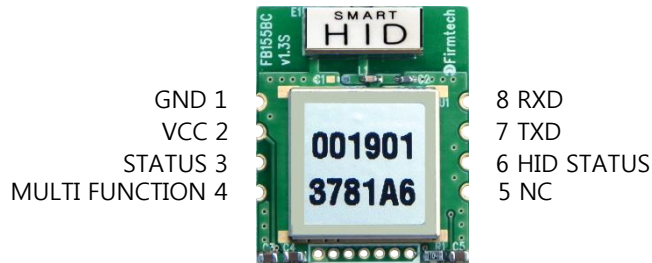


Mouse (USB or PS/2)



PS/2 지원제품
(바코드리더기 등)

2. FB155BC_SMD(HID) 구성핀 설명



Dimension : 18(Width) x 21.8(Length) x 3.4(Height) mm

- FB155BC_SMD(HID)는 HID Keyboard/Mouse Report를 시리얼(UART)로 입력 받아 무선으로 송신하는 Bluetooth 장치입니다.
- FB155BC_SMD(HID)는 반드시 Smart_HID Host가 있어야 무선 데이터 송신이 가능합니다.
- FB155BC_SMD(HID)는 반드시 HID Keyboard/Mouse Report 포맷을 입력 받아야 무선 데이터 송신이 가능합니다.

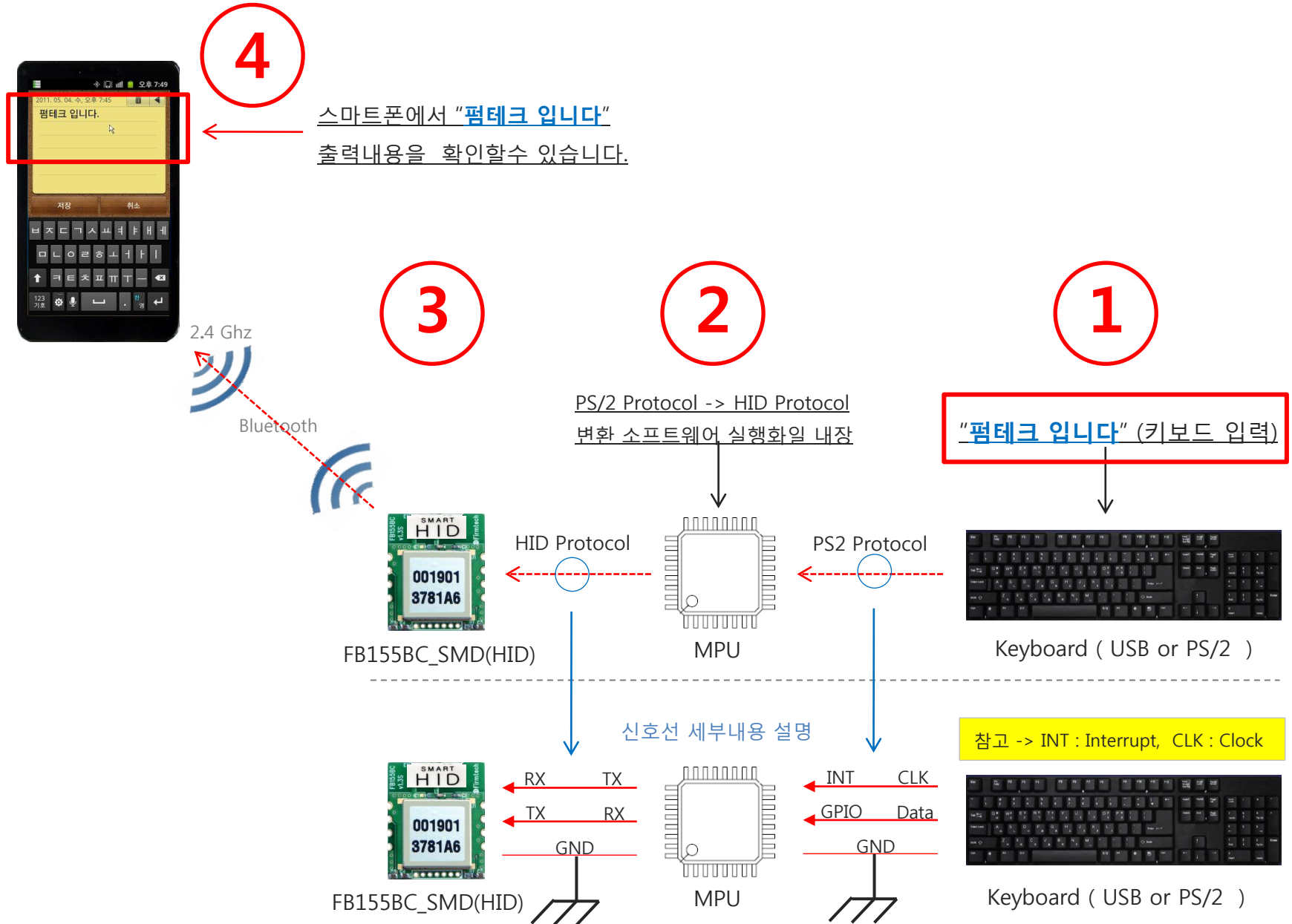
NO	Name	Description	Direction
1	GND	Ground	-
2	VCC	3.3V DC	I
3	STATUS	STATUS LED	O
4	MULTI FUNCTION	Multi Function Control	I
5	NC	NC	-
6	HID STATUS	HID Status	O
7	TXD	Transfer Data	O
8	RXD	Received Data	I

FB155BC_SMD(HID)는 FB155BC, FB155BC_SMD와 호환이 되지 않습니다.

FB155BC_SMD(HID) : HID Device Profile 운영

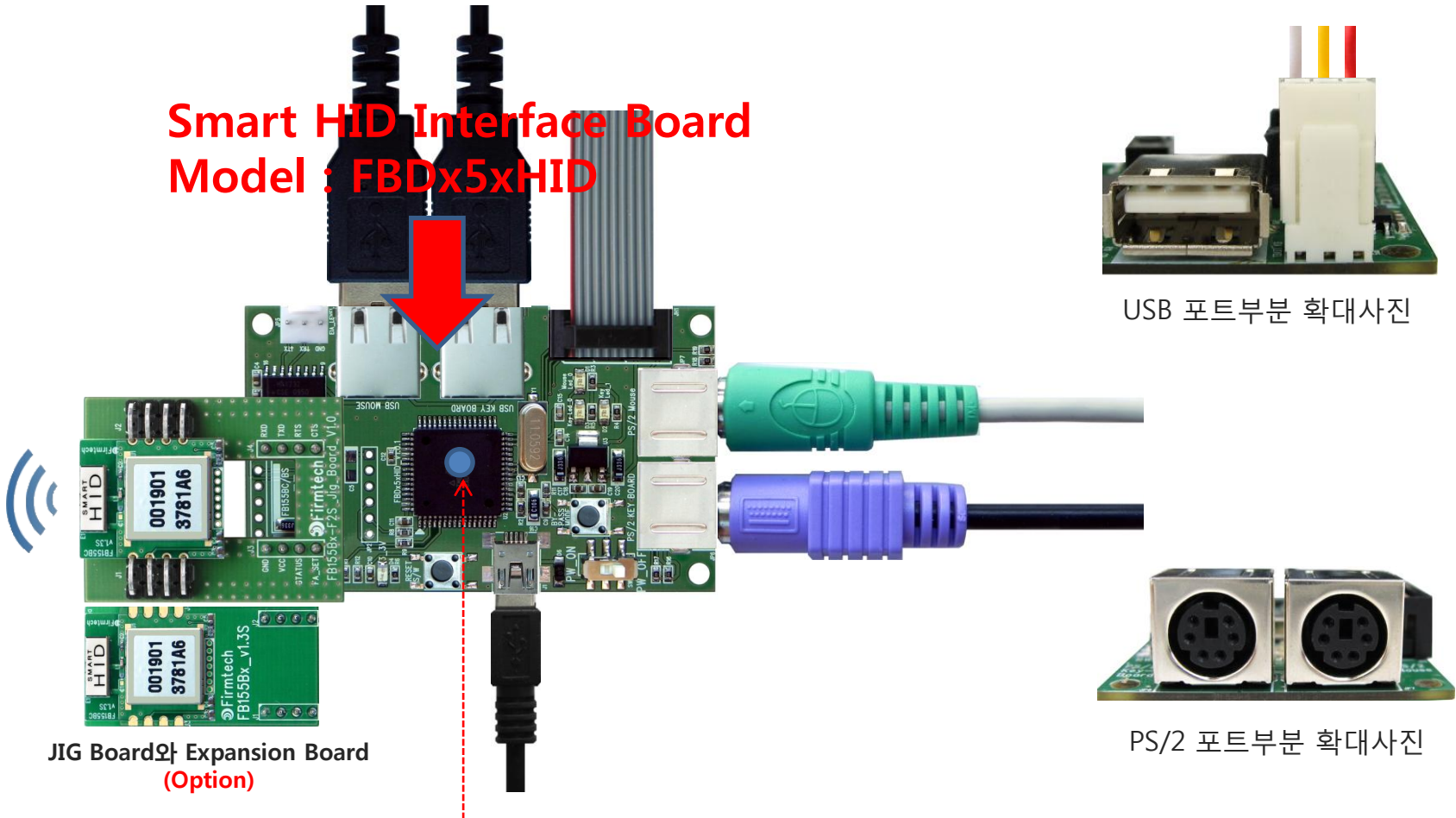
FB155BC, FB155BC_SMD : Serial Port Profile 운영

3. FB155BC_SMD(HID) 사용시 Keyboard 입력내용이 스마트폰으로 전달되는 과정



4. FB155BC_SMD(HID) 전용 Test Board 소개

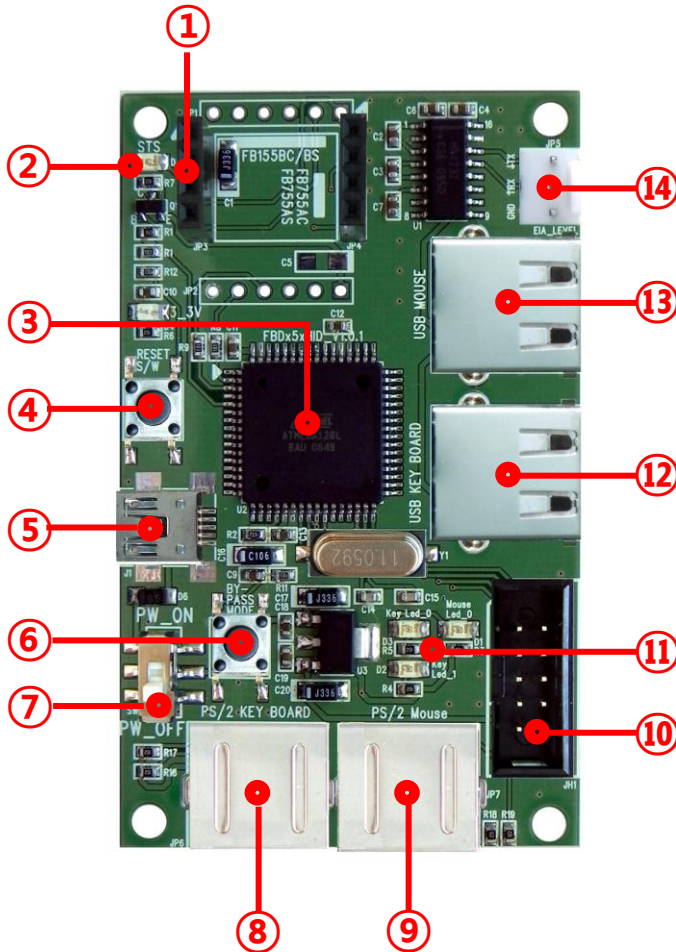
Smart HID Interface Board 는 FB155BC_SMD(HID) 전용 Test Board 로서 아래의 그림에서 보시는 것과 같이 USB 및 PS/2 방식의 Keyboard , Mouse 인터페이스를 제공합니다. PS/2 포트에는 Keyboard, Mouse 외에도 **PS/2 통신방식을 사용하는** 바코드리더기, 조이스틱 , 기타 장치 사용이 가능합니다.



Atmega128 플래시 메모리 영역에 "PS/2 Protocol -> HID Protocol 변환" 및 "Serial (RS232) Protocol -> HID Protocol 변환"에 관련된 기능이 구현된 Binary File 이 download 되어 사용자에게 공급 됩니다.

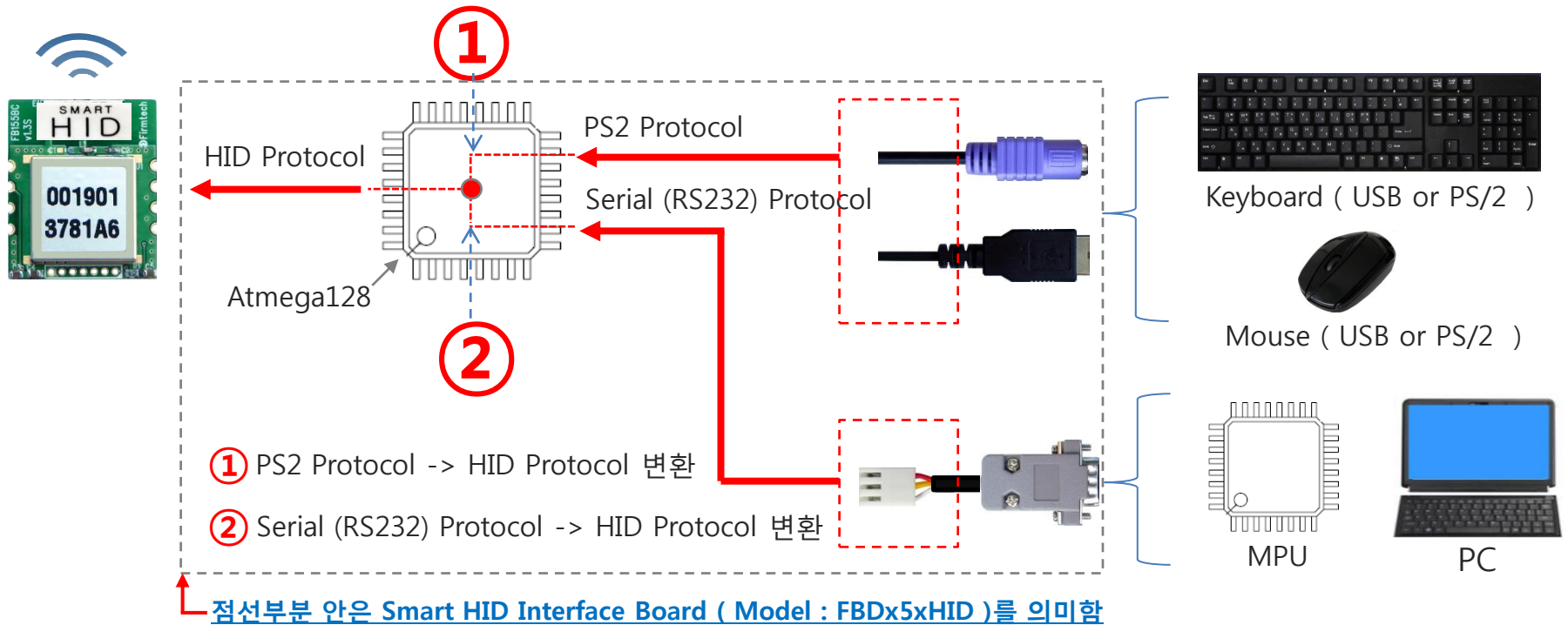
소스화일(C언어)은 "Serial (RS232) Protocol -> HID Protocol 변환" 소스가 사용자에게 공급 됩니다.

5. Smart HID Interface Board (Model : FBDx5xHID) 구성 소개



NO	Description
1	FB155BC(HID) Connector
2	BT Status LED
3	ATmega128
4	Reset Switch
5	USB Power Connector
6	Bypass Switch
7	Power ON/OFF Switch
8	PS2 Keyboard Connector
9	PS2 Mouse Connector
10	AVR ISP Loader Connector
11	Keyboard, Mouse Status LED
12	USB Keyboard Connector
13	USB Mouse Connector
14	RS-232 Connector

6. Smart HID Interface Board 구매시 제공하는 Firmware 종류 설명



Smart HID Interface Board (Model : FBDx5xHID) 의 구성품인 Atmega128 (MPU) 의 Flash memory 에는 그림상에 표시된 ① 기능이 구현된 실행 Binary File 이 download 되어 사용자에게 공급이 됩니다.

① 는 PS/2 Protocol 을 HID Protocol 로 변환시키는 펌웨어 를 의미합니다.

② 는 Serial (RS232) Protocol 을 HID Protocol 로 변환시키는 펌웨어 를 의미합니다.

Smart HID Interface Board (Model : FBDx5xHID) 구매자 분께 제공되는 펌웨어는 다음과 같습니다.

1. ① 기능이 구현된 실행 Binary File 제공
2. ② 에 관련된 프로그램 소스(C언어) 제공

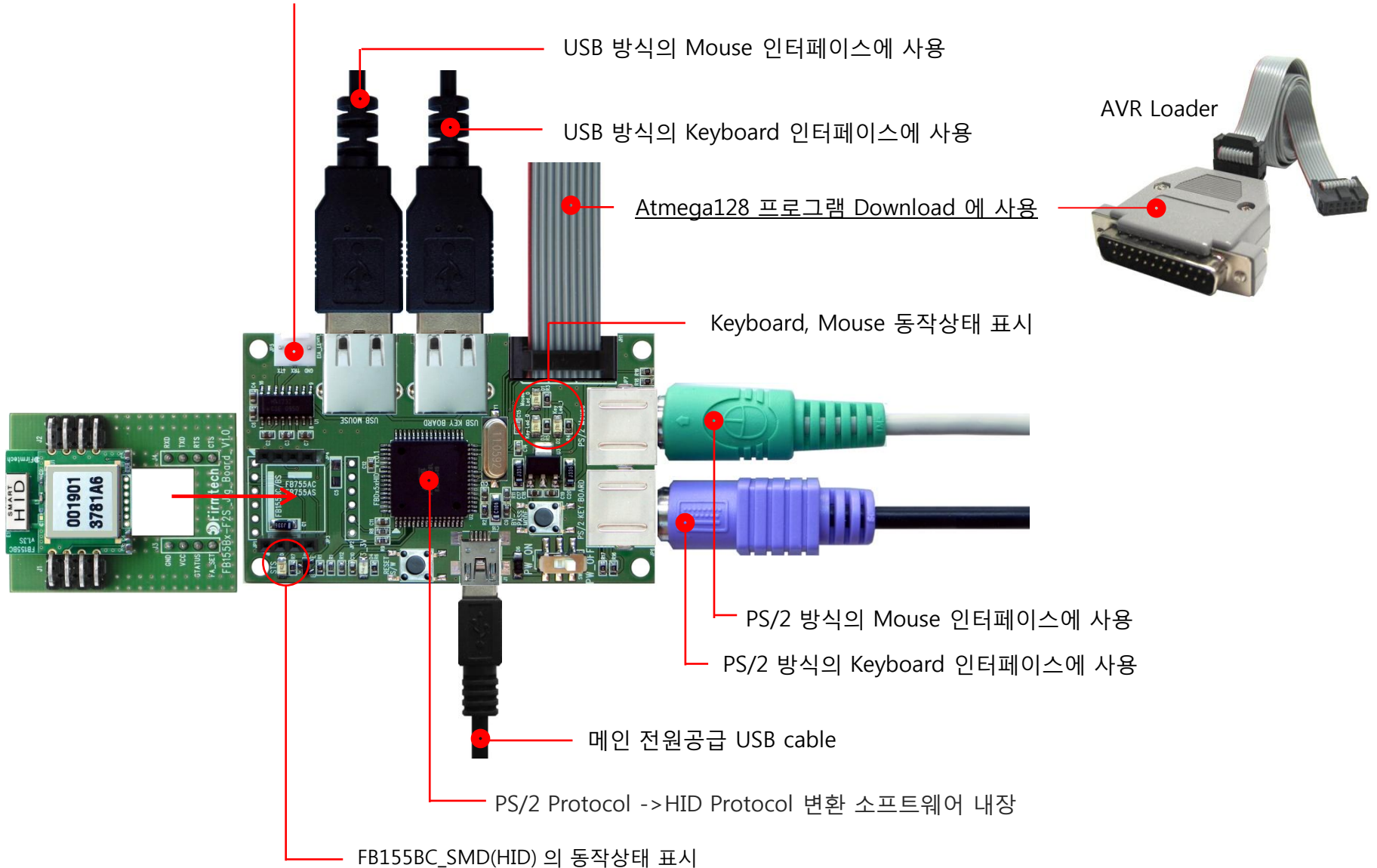
단 ① 에 관련된 프로그램 소스(C언어) 는 제공되지 않습니다.

① 의 프로그램 소스가 필요하신 분은 별도로 당사의 영업부에 문의를 해주시기 바랍니다.

7. Smart HID Interface Board 의 주요역할 (1)

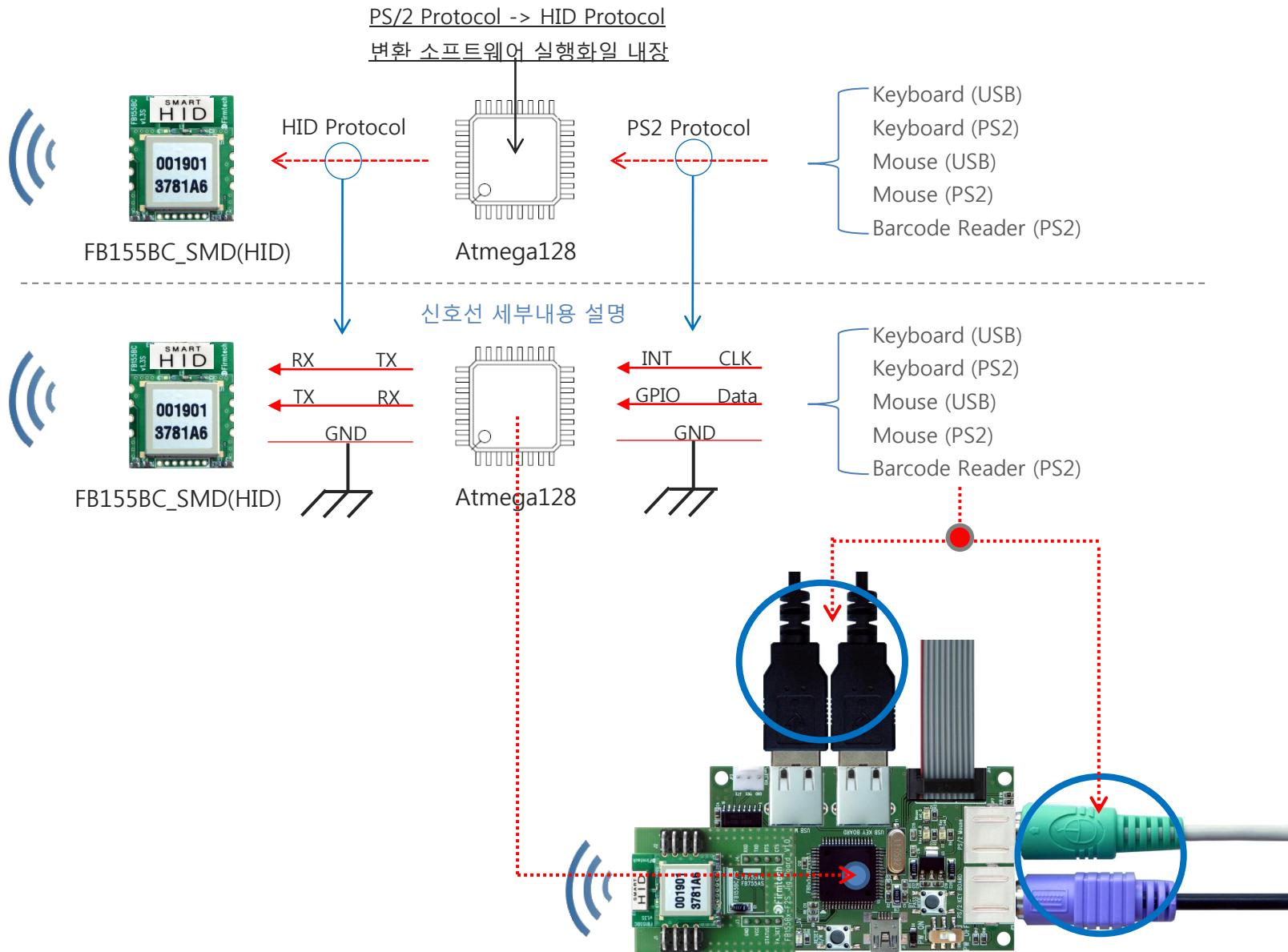
→ FB155BC SMD(HID) 기본성능 테스트, 동작상태 모니터링, 펌웨어 업데이트기능 지원

Atmega128 과 Serial (TX,RX,GND) 로 연결되어 타 디바이스와의 Serial (RS232)통신에 사용, Atmega128 내부상태 모니터링에 사용

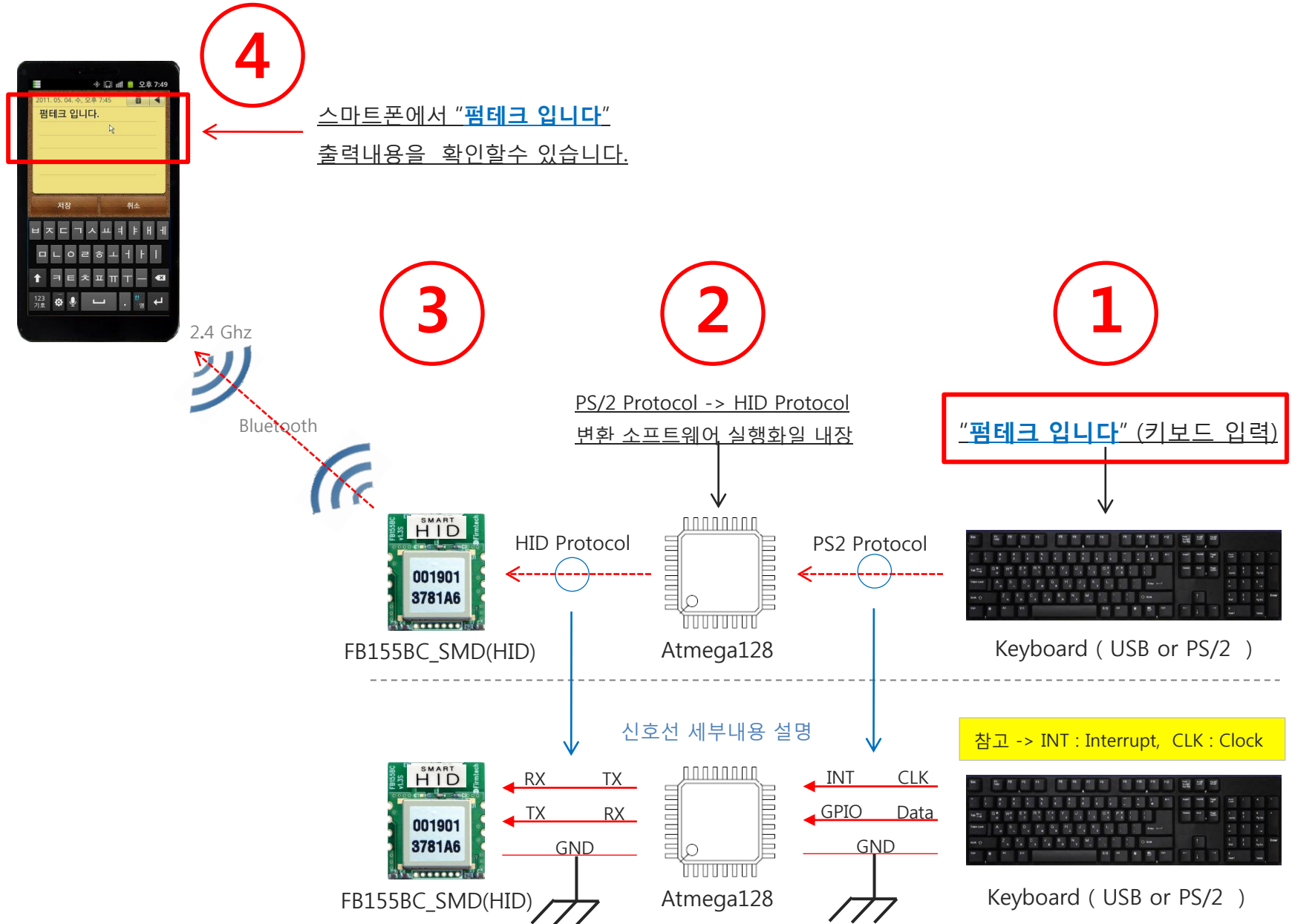


8. Smart HID Interface Board 의 주요역할 (2)

→ PS/2 Protocol -> HID Protocol 변환기능 지원



참고 : PS/2 Protocol -> HID Protocol 변환기능 활용예

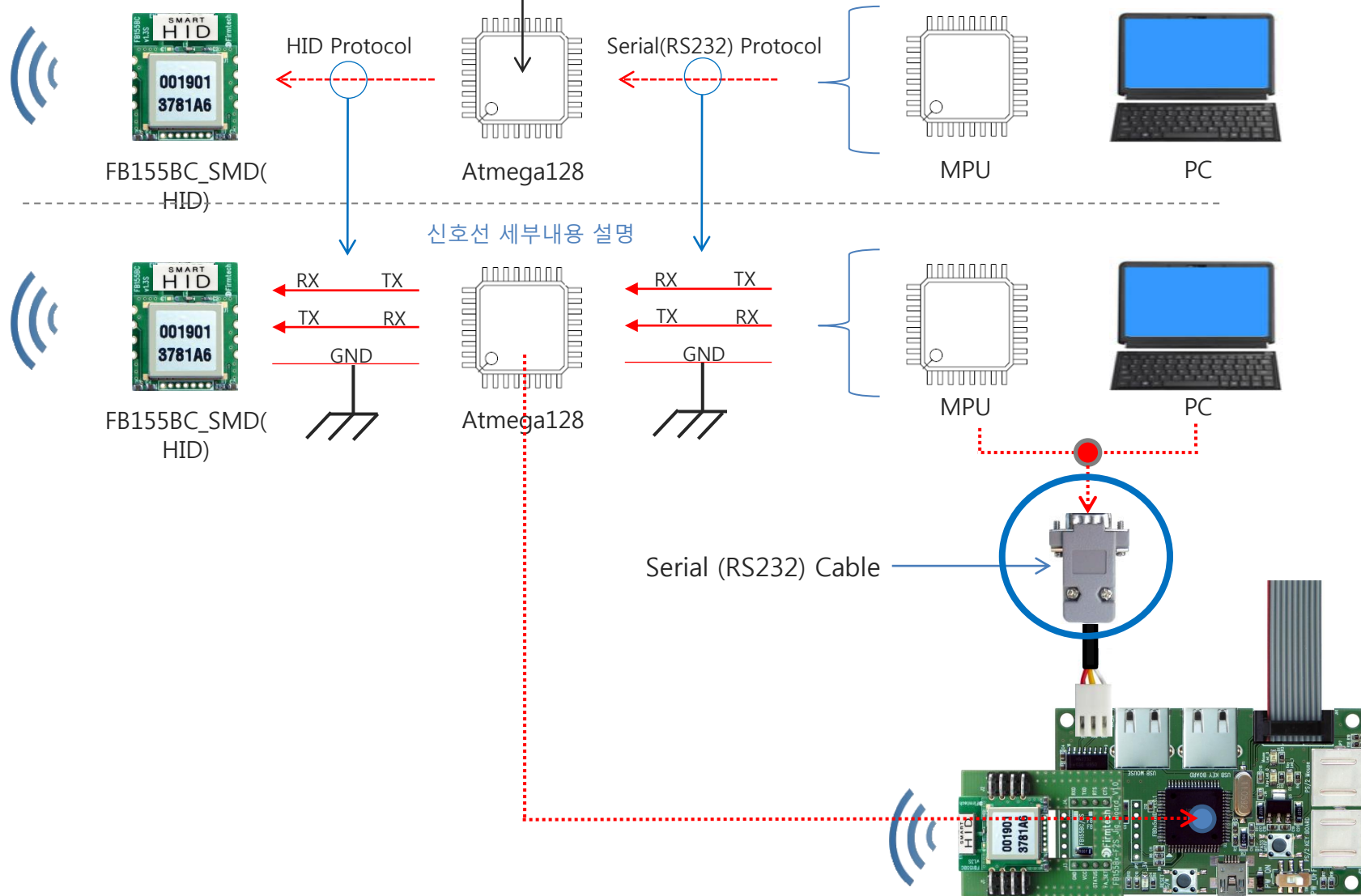


9. Smart HID Interface Board 의 주요역할 (3)

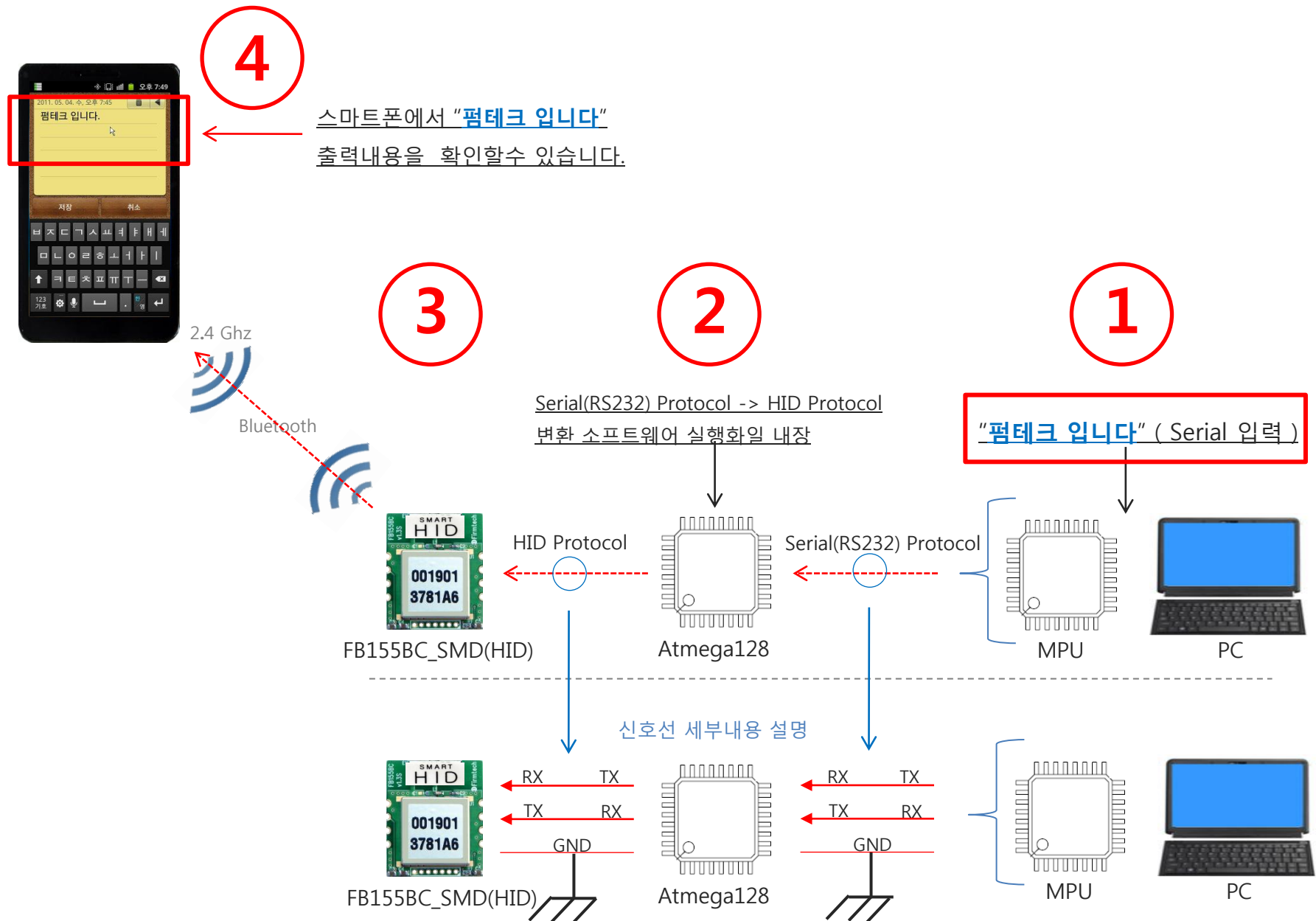
→ Serial (RS232) Protocol -> HID Protocol 변환기능 지원

Serial(RS232) Protocol -> HID Protocol

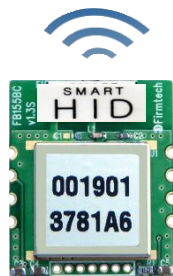
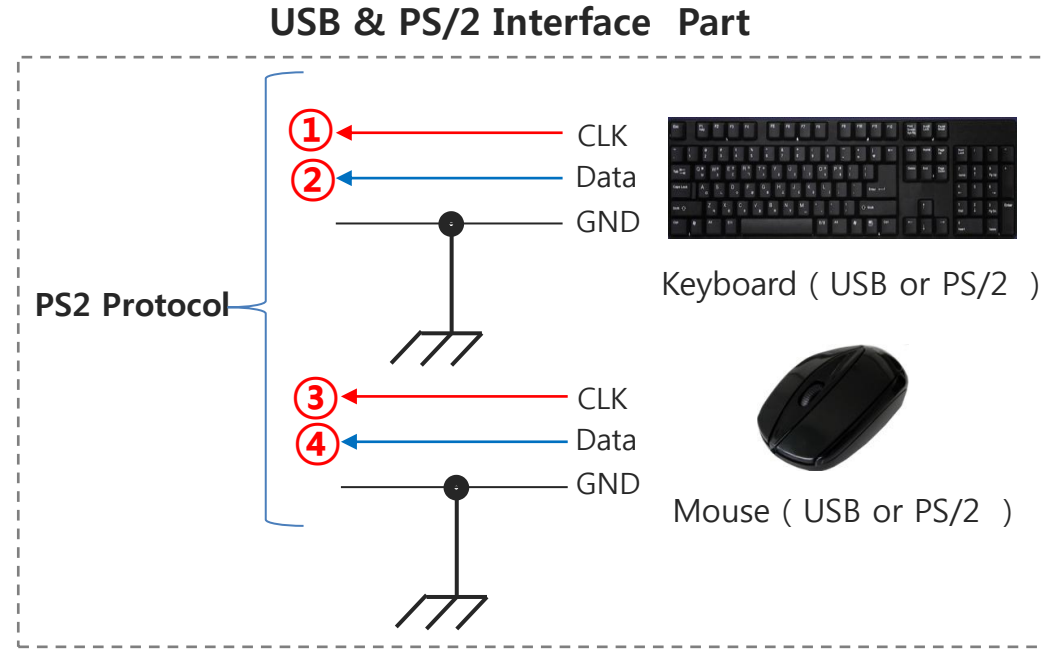
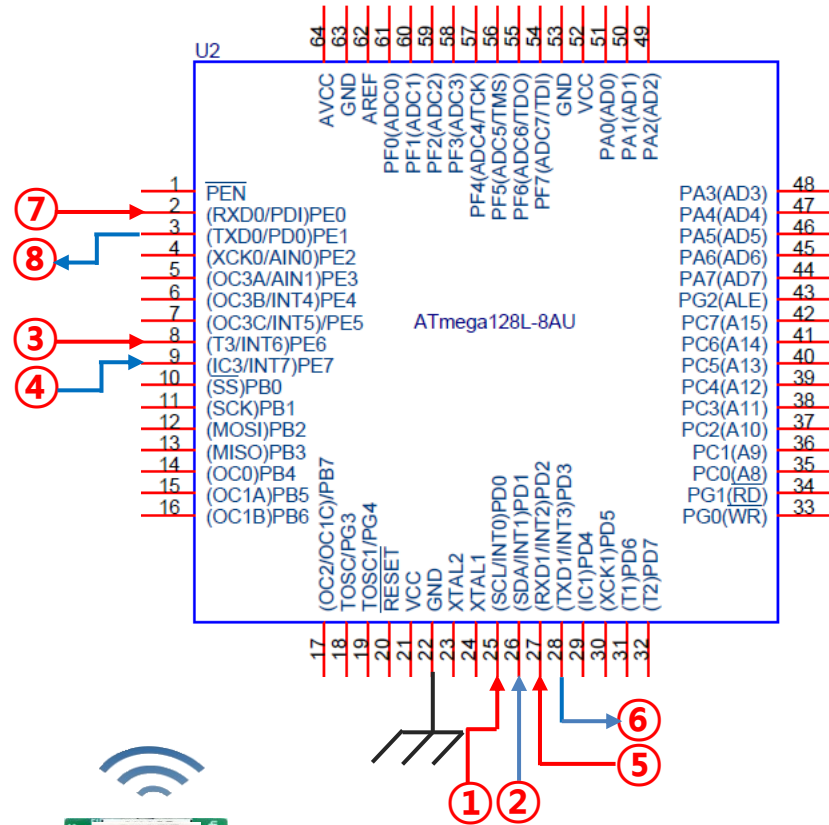
변환 소프트웨어 실행파일 내장



참고 : Serial (RS232) Protocol -> HID Protocol 변환기능 활용예



10. Smart HID Interface Board (Model : FBDx5xHID) 주요부분 연결도

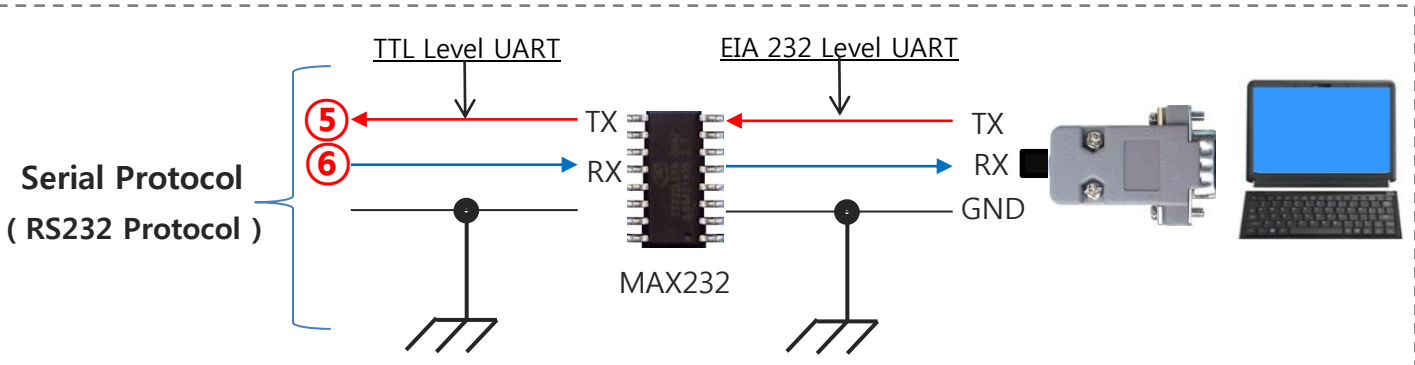


FB155BC_SMD(HID)







HID Protocol

Serial Protocol (RS232 Protocol)

Serial (RS232) Interface Part



11. Smart HID Interface Board 구성품 소개

모델명	사진	설명
Smart_HID Interface Board (Model : FBDx5xHID)		Keyboard/Mouse 데이터를 HID Report 로 변환
AVR Loader		AVR 프로그램 다운로더
USB cable		USB 전원 공급 케이블 (5 Pin)
CD		Source, Datasheet, Manual, Utility CD
RS-232 Cable		RS-232 시리얼 연결 케이블
JIG Board (Option)		FB155BC_SMD(HID) 모듈을 결착할 수 있는 JIG Board
Expansion Board (Option)		FB155BC_SMD(HID) 모듈을 붙일 수 있는 Expansion Board

참고사항 1. HID_Host 와 HID_Device 의 정의

- HID_Host는 데이터 처리 장치입니다.

- HID_Host는 PC와 스마트기기 (Samsung, Apple 사 등의 스마트폰, 스마트패드 등) 이 될 수 있습니다.
- HID_Host는 무선으로 수신받은 USB 또는 PS/2 방식의 Keyboard/Mouse 데이터를 처리합니다.
- HID_Host가 되기 위해서는 HID Profile중 **HID_Host** 가 운영되는 Bluetooth 장치가 있어야 합니다.

- HID_Device는 데이터 입력 장치입니다.

- HID_Device는 무선 키보드/무선 마우스, 또는 무선 스캐너 등이 될 수 있습니다.
- HID_Device는 USB 또는 PS/2 방식의 Keyboard, Mouse 데이터를 무선 Bluetooth 방식으로 송신합니다.
- HID_Device가 되기 위해서는 HID Profile중 **HID_Device**가 운영되는 Bluetooth 장치가 있어야 합니다.

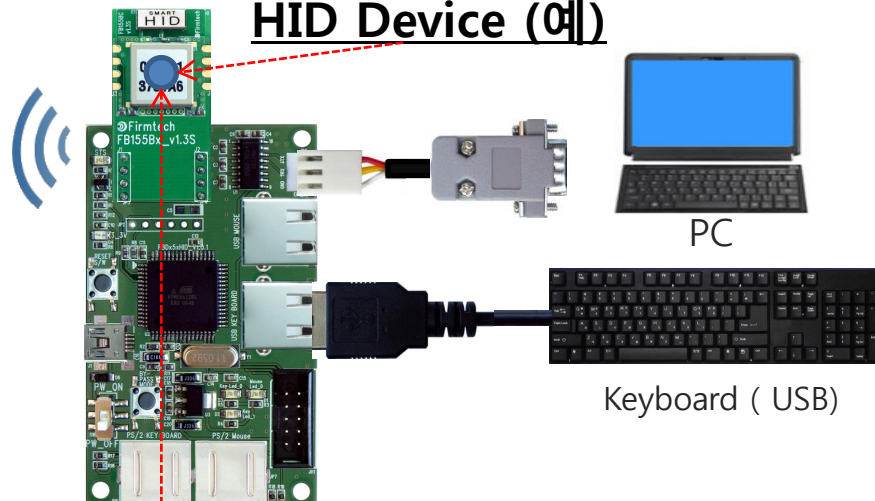
HID Host (예)



HID Profile 지원
스마트기기

2.4 Ghz
블루투스

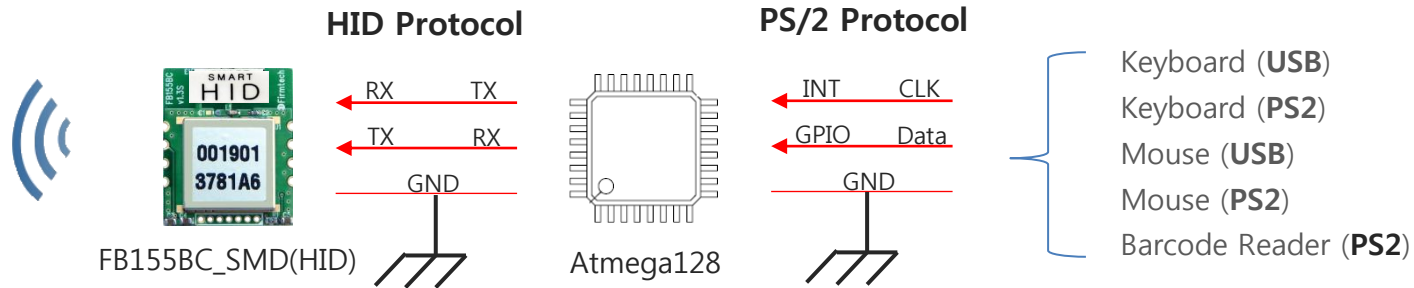
HID Device (예)



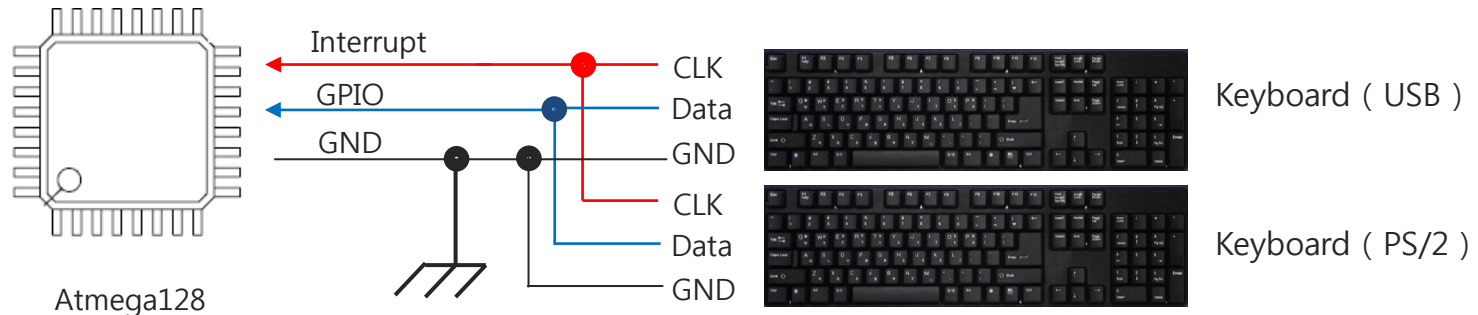
HID Profile 지원
FB155BC_SMD(HID)

참고사항 2. 제품사용시 꼭 확인하실 사항

1. USB 방식의 Keyboard/Mouse는 USB Protocol만 지원되는 제품과 USB Protocol과 PS/2 Protocol을 동시에 지원하는 제품으로 구분됩니다. **Smart HID Interface board(Model : FBDx5xHID)를 사용하기 위해서는 USB 방식의 Keyboard/Mouse중에서 반드시 PS/2 Protocol을 지원하는 제품을 사용해야 합니다.**



2. Smart_HID Interface Board 는 아래의 그림에서와 같이 USB 및 PS/2 방식의 Keyboard , Mouse 는 신호선을 공유합니다.



따라서

USB Keyboard 를 연결한 경우 PS/2 Keyboard 연결은 금지됩니다. (동시사용 불가하며 둘중의 하나를 선택하여 사용하세요)

USB Mouse 를 연결한 경우 PS/2 Mouse 연결은 금지됩니다. (동시사용 불가하며 둘중의 하나를 선택하여 사용하세요)

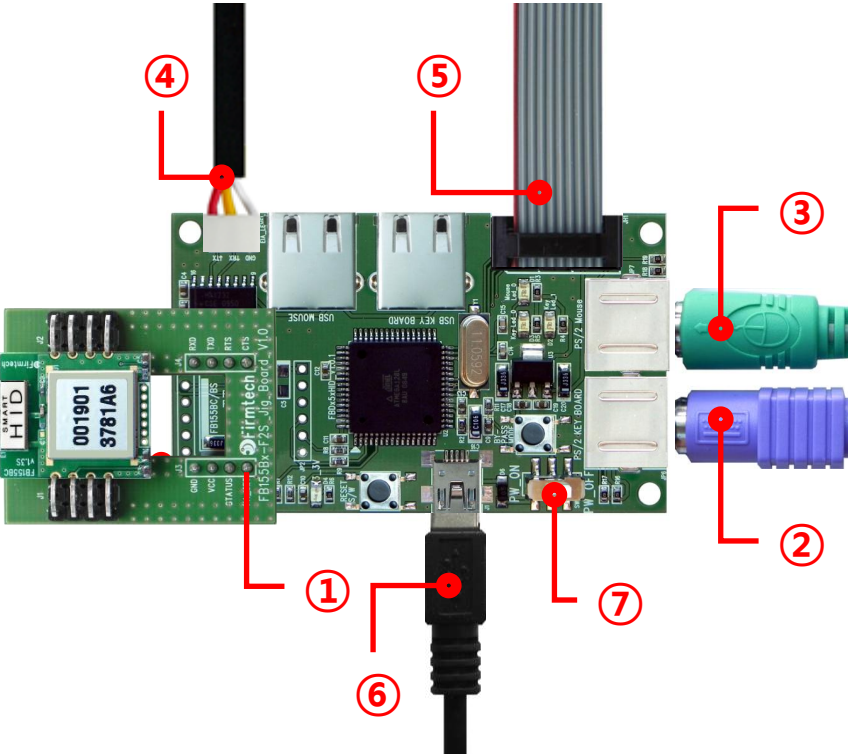
PS2_to_HID

HEX File Download

12. PS2_to_HID 프로그램 다운로드

(1) FB155BC_SMD(HID) 장착 및 Cable 연결

- Smart_HID Interface Board에서 운영되는 프로그램을 수정한 경우, 사용자는 ATmega128에 새로운 프로그램을 다운로드 해야 합니다.



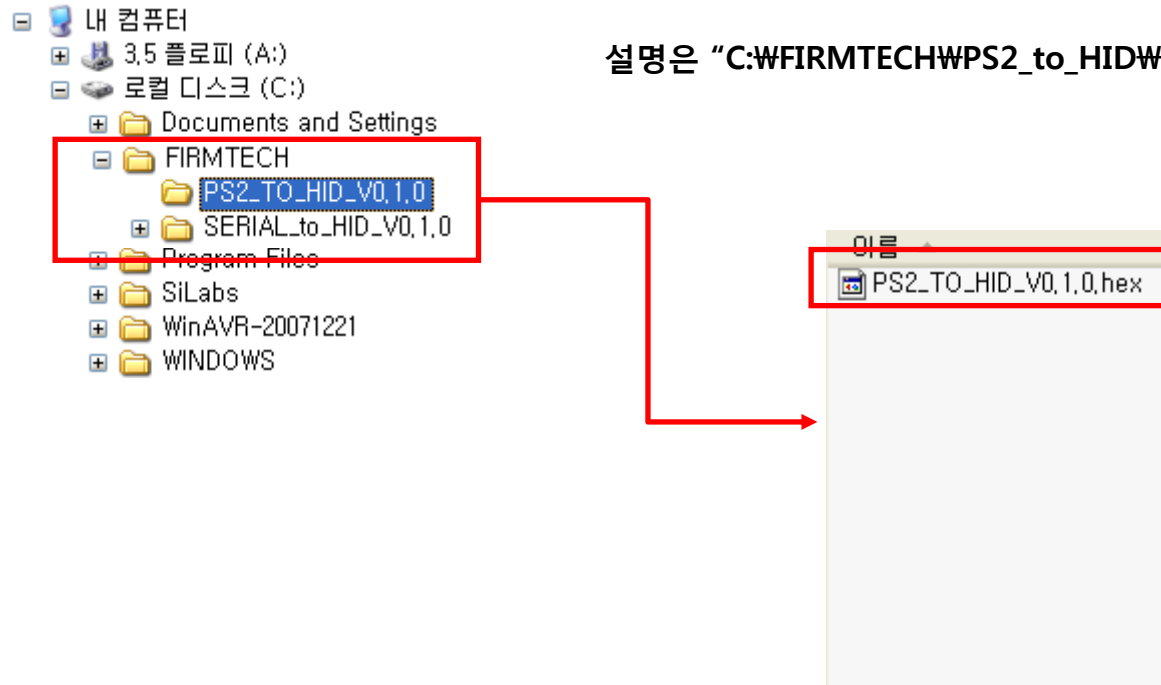
NO	연결 및 체크사항
1	Smart_HID Interface Board에 FB155BC_SMD(HID) 장착
2	Smart_HID Interface Board에 키보드(또는 PS2-Scanner) 장착
3	Smart_HID Interface Board에 마우스 장착
4	Smart_HID Interface Board와 RS-232 Cable을 연결하여 PC와 연결
5	Smart_HID Interface Board와 AVR Loader(다운로드 케이블)연결하여 PC와 연결
6	Smart_HID Interface Board와 USB Power Cable을 연결하여 PC와 연결
7	Smart_HID Interface Board 전원 ON

12. PS2_to_HID 프로그램 다운로드

(2) PS2_to_HID.hex 파일 위치

CD로 제공되는 PS2_to_HID 실행파일은 사용자가 알맞은 폴더를 만들어 복사 합니다.

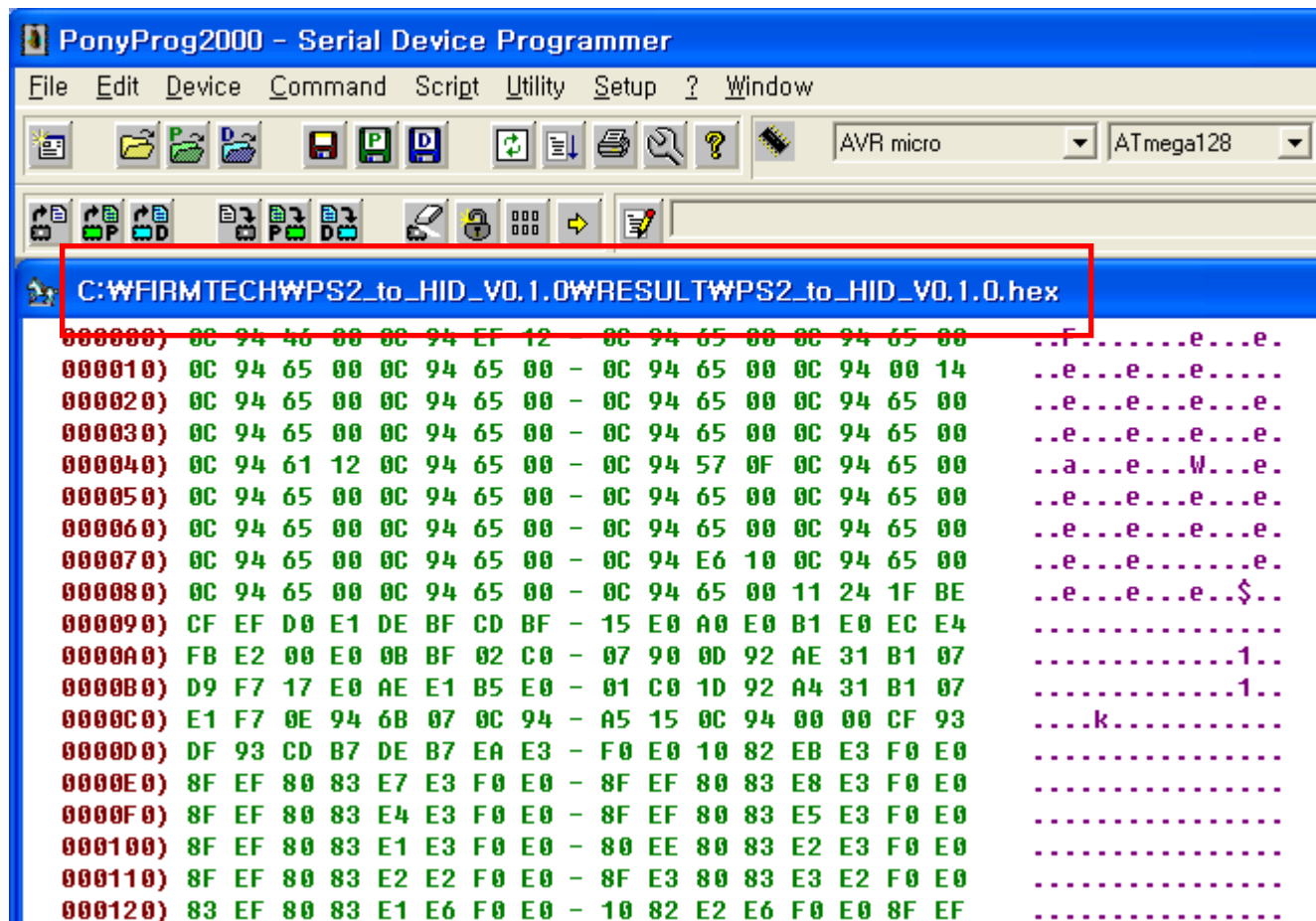
설명은 "C:\WFIRMTECH\PS2_to_HID"폴더로 설명합니다.



12. PS2_to_HID 프로그램 다운로드

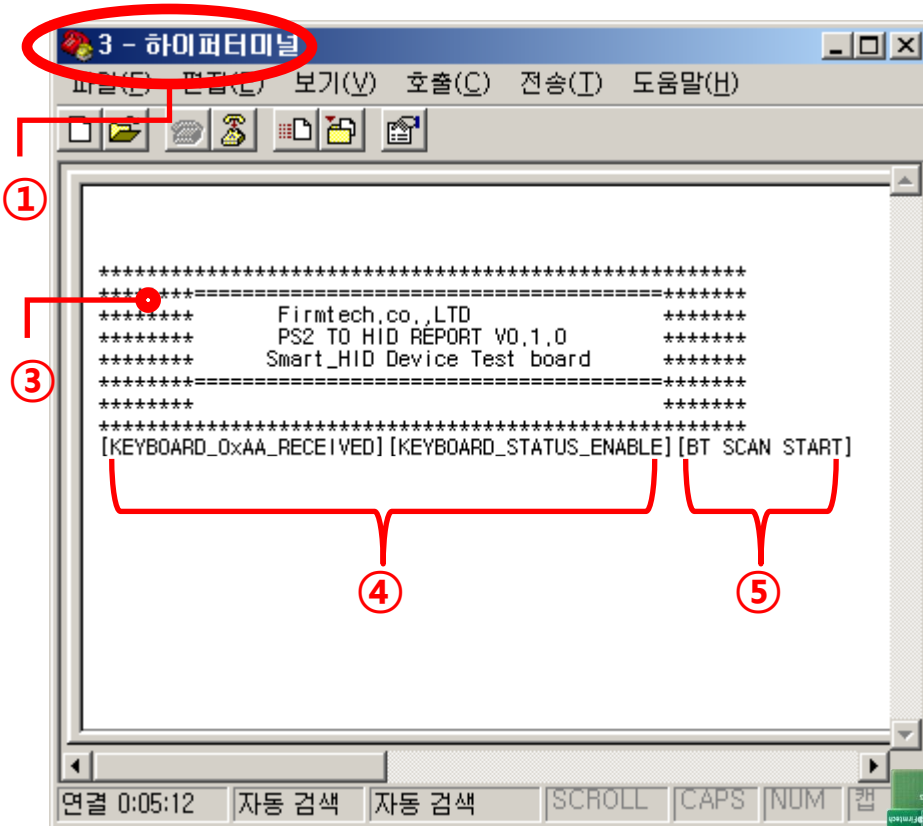
(3) PonyProg 프로그램을 사용하여 PS2_to_HID.hex 프로그램 다운로드

Tool에 대한 설명은 펌테크 홈페이지에서
"FB800ED_Tools Guide.PDF"파일을 참고 하십시오.

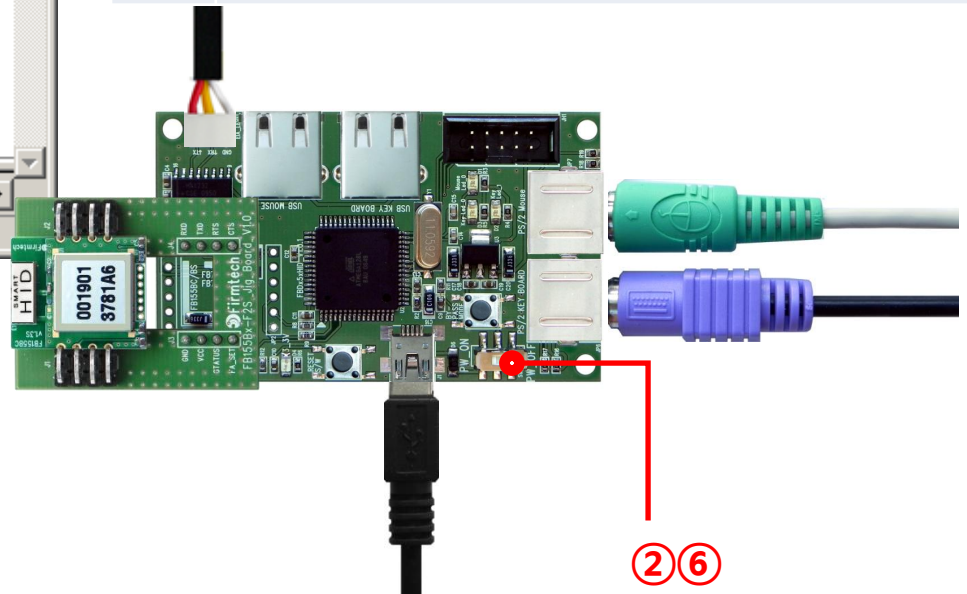


12. PS2_to_HID 프로그램 다운로드

(4) PS2_to_HID 프로그램 동작 확인



NO	연결 및 체크사항
1	프로그램 다운로드 완료된 Smart_HID Interface Board와 연결된 PC의 하이퍼 터미널 Open <ul style="list-style-type: none">- PC의 포트는 3번을 사용하는 것으로 가정- 통신 속도는 115200 선택- 흐름제어 없음
2	Smart_HID Interface Board 전원 ON
3	하이퍼 터미널에 운영 프로그램 정보 출력 <ul style="list-style-type: none">- Smart_HID Interface 프로그램 운영 확인 가능
4	하이퍼 터미널에 Keyboard 상태 출력
5	하이퍼 터미널에 BT Module Scan 메시지 출력
6	Smart_HID Interface Board 전원 OFF



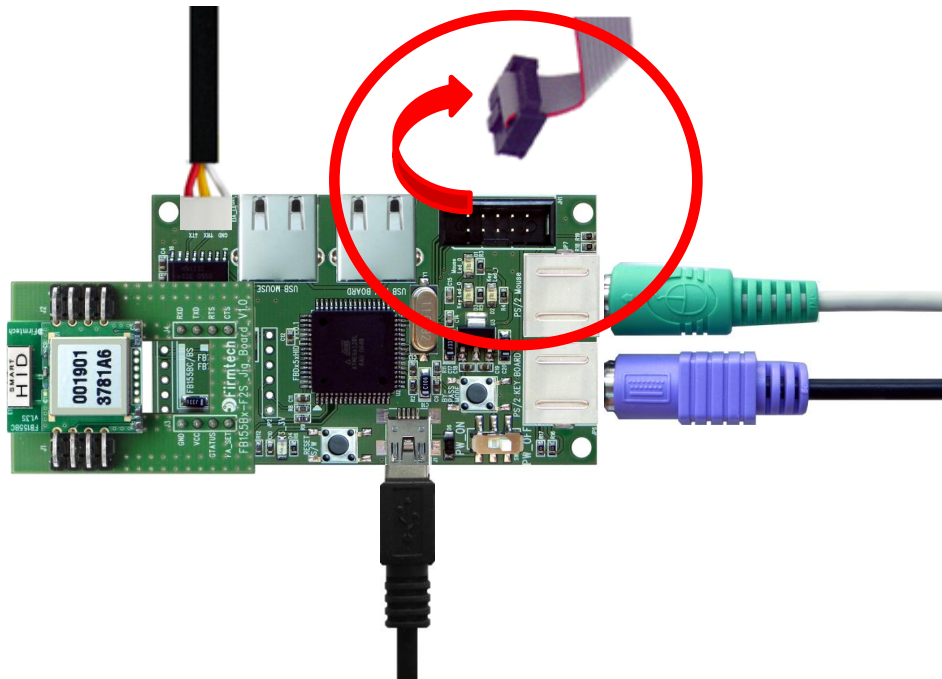
■ 체크 사항

Smart_HID Interface Board의 전원을 ON 했을 때, FB155BC_SMD(HID)로부터 “BTWIN HID Slave mode start” 메시지를 수신하지 못하는 경우, AVR Loader가 PC와 연결되어 있는지 체크합니다.

(하이퍼 터미널에 [ERROR, STEP:4]라고 출력됨)

AVR Loader가 Smart_HID Interface Board와 PC에 연결되어 있는 상태에서, Ponyprog 프로그램이 PC에서 실행되어 있지 않으면, Smart_HID Device Module에서 출력되는 시리얼 데이터가 Smart_HID Interface Board에 정상적으로 수신되지 않습니다.

그러므로, 프로그램 다운로드가 완료되었다면 AVR Loader를 Smart_HID Interface Board와 연결하지 않습니다.



PS2_to_HID를 이용한

FB155BC_SMD(HID)

&

Smart_HID Host

(PHONE: Galaxy-S)

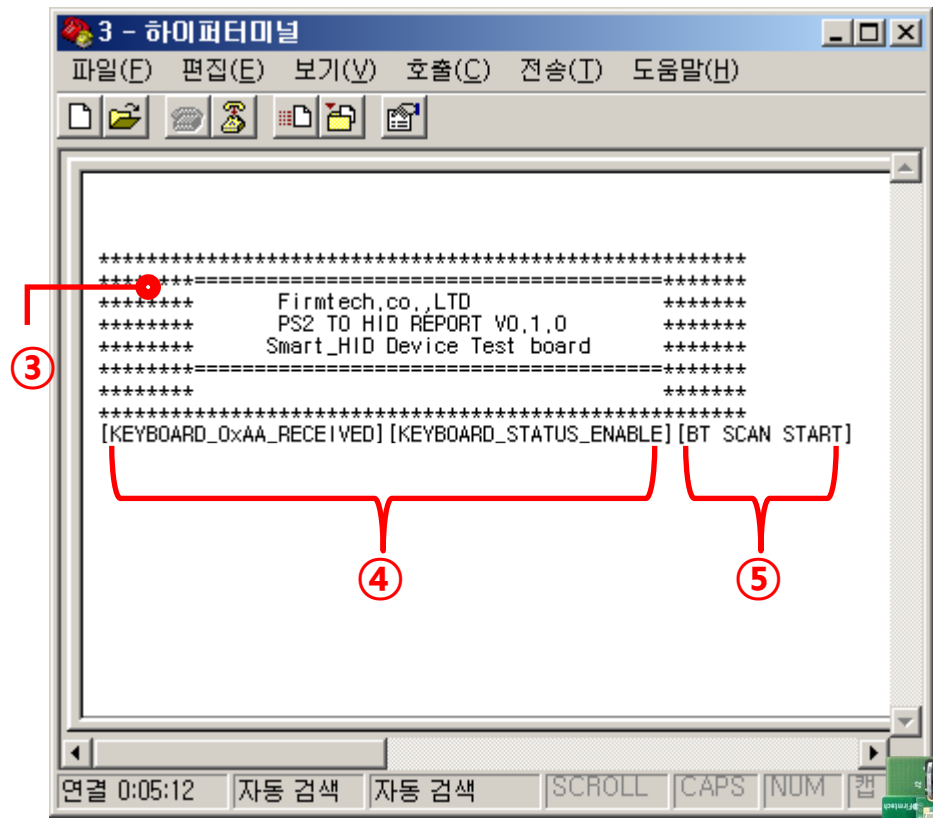
연결 진행

13. Smart_HID를 사용하기 위한 기본 사항

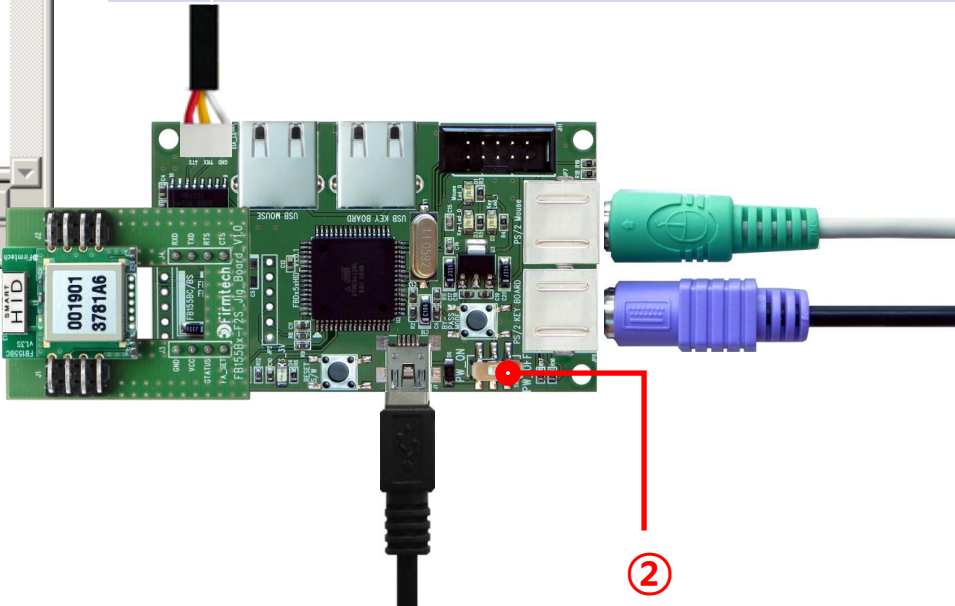
- Smart_HID Host는 **PHONE(Galaxy-S)**을 사용합니다.
- FB155BC_SMD(HID)는 HID Device Profile이 운영되고 있어야 합니다.
- Smart_HID Interface Board는 **PS2_to_HID** 프로그램이 운영되고 있어야 합니다.
- Keyboard(또는 PS2-Scanner)와 Mouse가 있어야 합니다.

NO	Description
1	FB155BC_SMD(HID)를 Smart_HID Interface Board에 장착
2	Smart_HID Interface Board에 Keyboard와 Mouse 장착
3	Smart_HID Interface Board 전원 ON => FB155BC_SMD(HID) Scan 동작 확인
4	Galaxy-S 메인메뉴=>환경설정=>무선 및 네트워크=>블루투스 설정=>검색=>연결
5	Galaxy-S의 메모 실행
6	Smart_HID Interface Board와 연결된 Keyboard & Mouse 동작
7	Galaxy-S의 메모에서 문자 입력 & Mouse 동작 확인

14. Smart_HID Interface Board 전원 ON



NO	연결 및 체크사항
1	FB155BC_SMD(HID)는 Galaxy-S가 동작되기 전에 SCAN 작업을 진행해야 합니다. 그러므로 Smart_HID Interface Board의 전원 ON을 미리 진행해야 합니다.
2	FB155BC_SMD(HID)가 장착된 Smart_HID Interface Board 전원 ON
3	하이퍼 터미널에 운영 프로그램 정보 출력 - Smart_HID Interface 프로그램 운영 확인 가능
4	하이퍼 터미널에 Keyboard 상태 출력
5	하이퍼 터미널에 BT Module Scan 메시지 출력 (Smart_HID Interface Board의 Status LED는 깜빡거립니다.)



15. Galaxy-S “메인메뉴” 선택



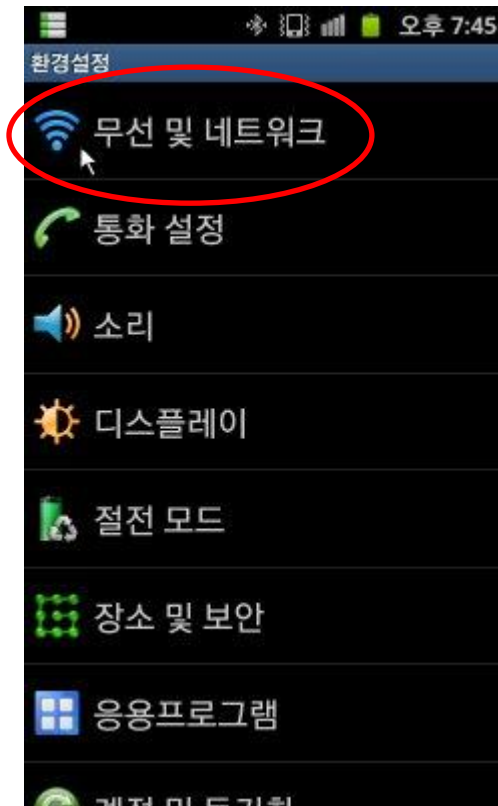
Galaxy-S 화면상에서 “메인메뉴” 아이콘을 클릭합니다.

16. Galaxy-S “환경설정” 선택



메인메뉴 중 “환경설정” 아이콘을 클릭합니다.

17. Galaxy-S “무선 및 네트워크” 선택



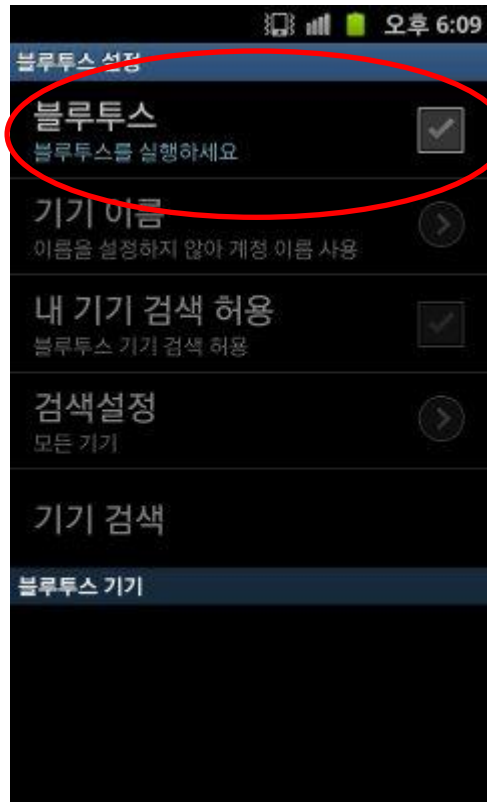
환경설정 메뉴 중 “무선 및 네트워크” 아이콘을 클릭합니다.

18. Galaxy-S “블루투스 설정” 선택



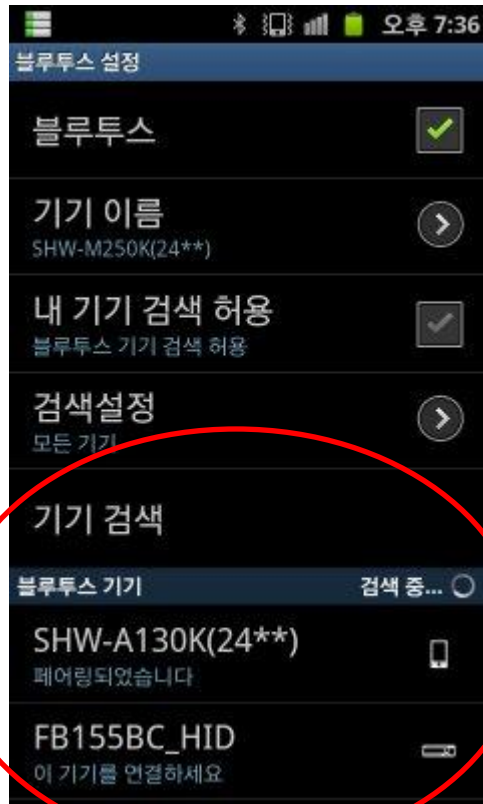
무선 및 네트워크 메뉴 중 “블루투스 설정” 아이콘을 클릭합니다.

19. Galaxy-S Bluetooth “ON”



Bluetooth 기능을 “ON” 시킵니다.

20. Galaxy-S Bluetooth 검색 진행



Bluetooth 기능이 ON되면 자동으로 주변의 Bluetooth 장치를 검색합니다.

"기기 검색"을 클릭하면 수동으로 주변의 Bluetooth 장치를 검색합니다.

검색된 Bluetooth 장치 리스트 중,
"FB155BC_HID"가 있는지 확인합니다.

"FB155BC_HID"가 없으면, 다시 검색을 진행하거나 "FB155BC_HID"가 Scan 작업을 정확하게 수행 중인지 확인합니다.

21. FB155BC_HID를 선택하여 연결 진행



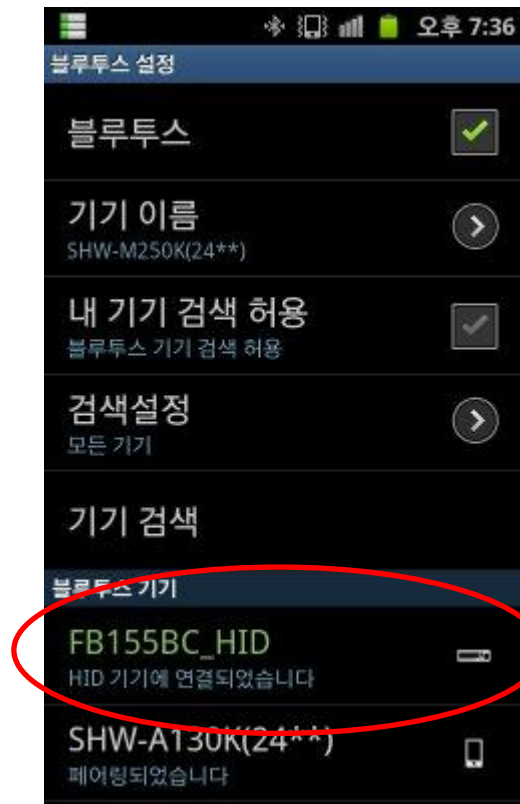
"FB155BC_HID"를 클릭하여 연결을 진행합니다.

22. 연결 요청

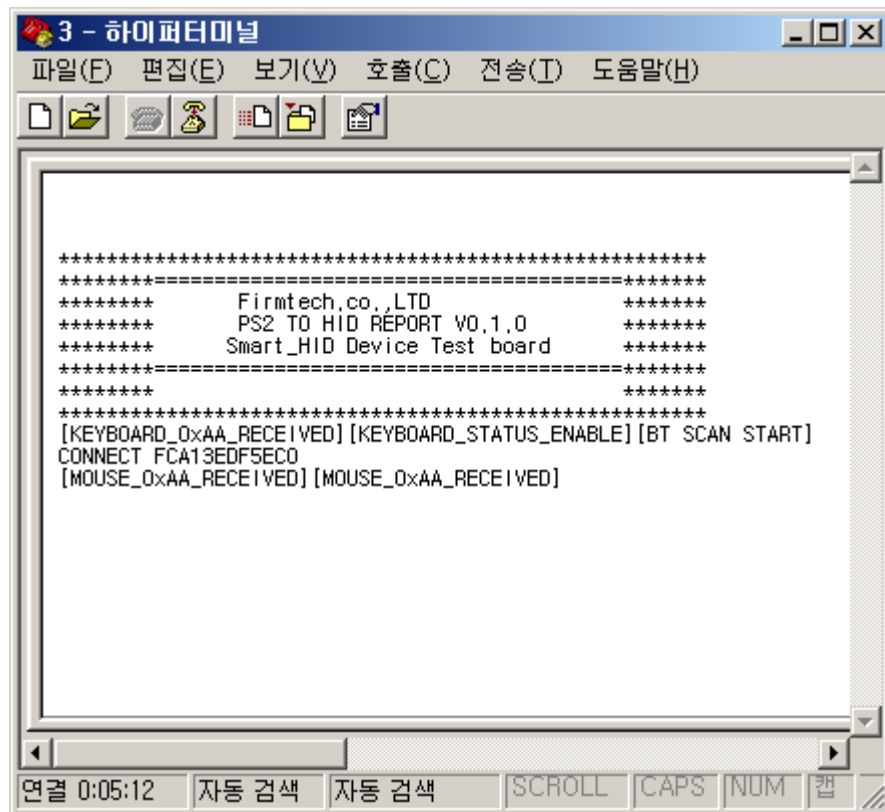


블루투스 연결 요청 화면이 나오는 경우
"연결"을 클릭하여 연결을 진행합니다.

23. FB155BC_HID와 연결 완료



24. Smart_HID Interface Board 연결 후 상태



“CONNECT xxxxxxxxxxxx”라는 메시지는 Bluetooth 장치가 연결된 것을 나타냅니다.

Bluetooth 장치가 연결되면 Mouse 초기설정을 진행합니다.

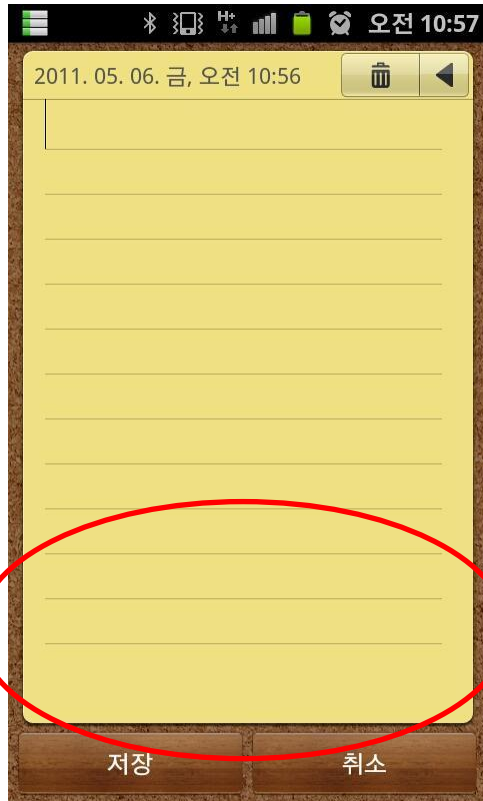
첫번째 출력되는 [MOUSE_0xAA_RECEIVED] 메시지는 Mouse에 최초 전원 인가시 발생하는 데이터에 의해 출력됩니다.

두번째 출력되는 [MOUSE_0xAA_RECEIVED] 메시지는 Mouse의 초기설정 진행 시 발생하는 데이터에 의해 출력됩니다.

25. Galaxy-S의 메모 실행



26. Galaxy-S의 자판 활성화



메모를 실행시키고, 화면을 터치해서 자판이 활성화 된 상태를 만들어야 합니다.

Galaxy-S의 자판이 활성화 되어 있지 않으면 한/영의 입력이 정상적으로 진행되지 않는 경우가 있습니다.

숫자만 입력이 되는 경우, 반드시 자판을 1회 이상 활성화 시켜야 합니다.

27. 무선 키보드 데이터 출력된 화면



28. 한영 전환



Galaxy-S는 기본적으로 한글을 출력합니다.

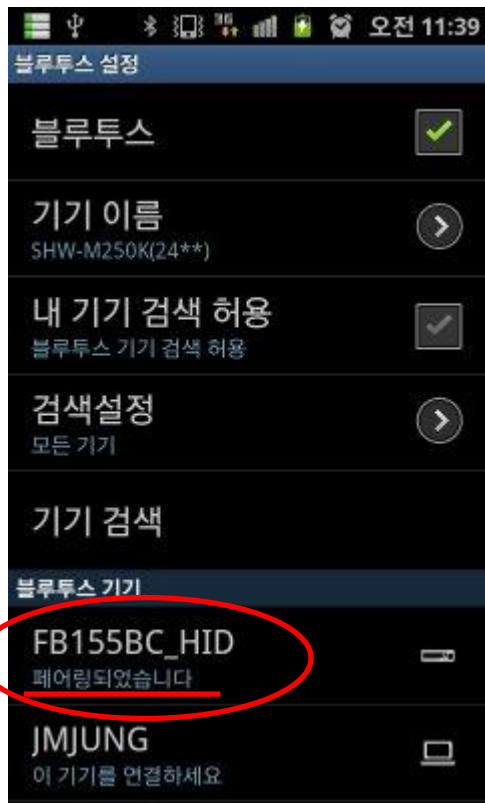
한글에서 영문 변환을 위해서는 무선 키보드 상에서 "한/영"키를 누르면 한영 변환이 이루어 집니다.

영문에서 한글 변환도 위와 같은 방법으로 진행합니다.

Galaxy-S는 블루투스 연결과 동시에 무선 키보드와 무선 마우스가 모두 동작 가능합니다.

29. 재 연결을 위한 Bluetooth 기기 상태 체크

1. Galaxy-S의 경우, Bluetooth 장치의 연결 종료를 Galaxy-S에서 진행하지 않고 상대방 장치에서 진행하는 경우(전원 OFF등), 상대방 장치의 상태가 "페어링되었습니다"로 나타납니다.
2. 상대방 장치의 상태가 "페어링되었습니다"로 되어 있는 경우 재 연결이 진행되지 않습니다. 이 경우, 리스트에서 해당 장치를 길게 클릭하여 "연결 취소"를 진행해야 재 연결이 가능합니다.
3. Galaxy-S의 Bluetooth Stack의 특성상, "페어링되었습니다"로 남아있는 장치와는 재 연결이 불가능 합니다.



PS2_to_HID를 이용한

FB155BC_SMD(HID)

&

Smart_HID Host

(iPad)

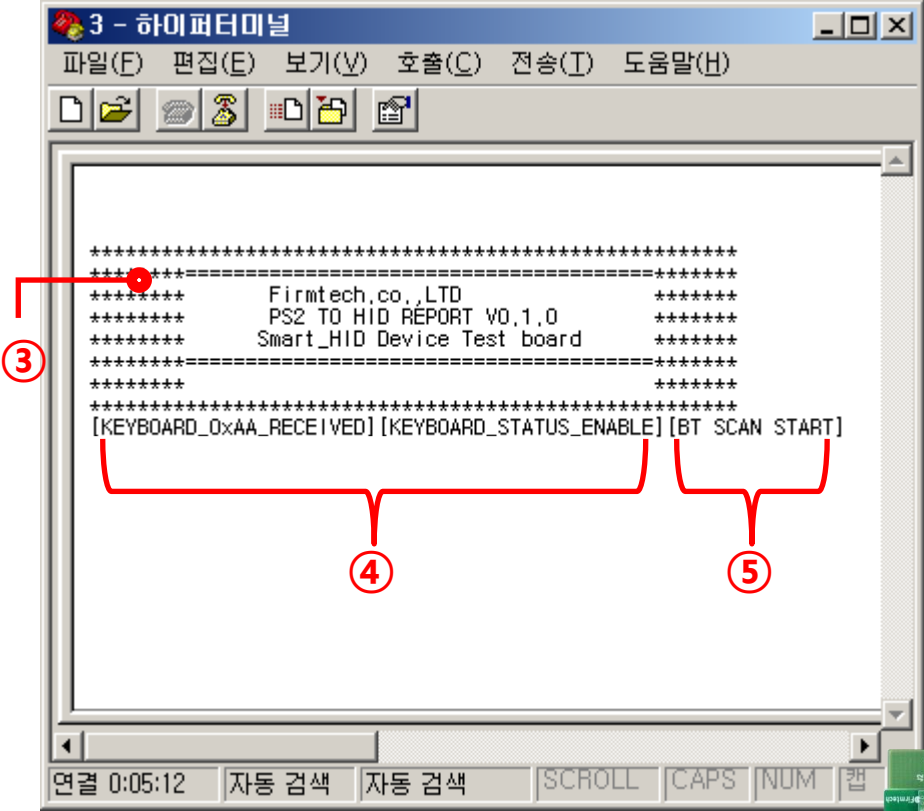
연결 진행

30. Smart_HID를 사용하기 위한 기본 사항

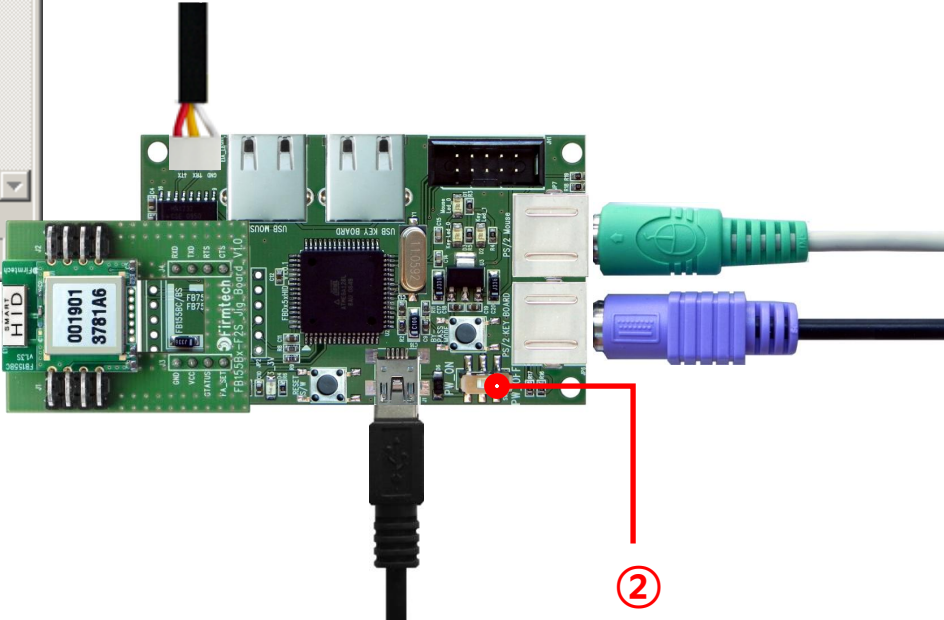
- Smart_HID Host는 **iPad**를 사용합니다.
- FB155BC_SMD(HID)는 HID Device Profile이 운영되고 있어야 합니다.
- Smart_HID Interface Board는 **PS2_to_HID** 프로그램이 운영되고 있어야 합니다.
- Keyboard(또는 PS2-Scanner)가 있어야 합니다.
(iPad에서는 무선 Mouse를 동작시킬 수 없습니다.)

NO	Description
1	FB155BC_SMD(HID)를 Smart_HID Interface Board에 장착
2	Smart_HID Interface Board에 Keyboard와 Mouse 장착 (iPad에서 무선 Mouse는 동작되지 않으나 Mouse 장착을 기본으로 설명합니다.)
3	Smart_HID Interface Board 전원 ON => FB155BC_SMD(HID) Scan 동작 확인
4	iPad 설정=>일반=>Bluetooth=>검색=>연결
5	iPad의 메모 실행
6	Smart_HID Interface Board와 연결된 Keyboard 동작
7	iPad의 메모에서 문자 입력 확인

31. Smart_HID Interface Board 전원 ON



NO	연결 및 체크사항
1	FB155BC_SMD(HID)는 iPad가 동작되기 전에 SCAN작업을 진행해야 합니다. 그러므로 Smart_HID Interface Board의 전원 ON을 미리 진행해야 합니다.
2	FB155BC_SMD(HID)가 장착된 Smart_HID Interface Board 전원 ON
3	하이퍼 터미널에 운영 프로그램 정보 출력 - Smart_HID Interface 프로그램 운영 확인 가능
4	하이퍼 터미널에 Keyboard 상태 출력
5	하이퍼 터미널에 BT Module Scan 메시지 출력 (Smart_HID Interface Board의 Status LED는 깜빡거립니다.)



32. iPad “설정” 선택



iPad 화면상에서 “**설정**” 아이콘을 클릭합니다.

33. iPad “일반” 선택



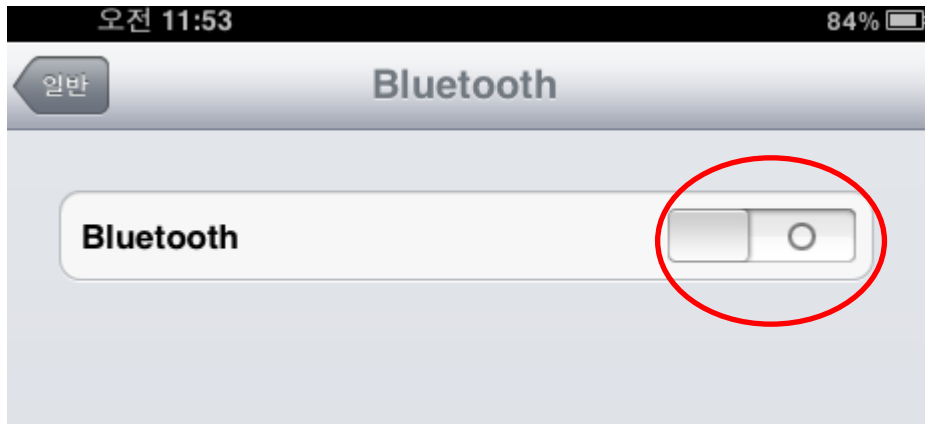
설정 메뉴 중 “**일반**” 아이콘을 클릭합니다.

34. iPad “Bluetooth” 선택



일반 메뉴 중 “**Bluetooth**” 아이콘을 클릭합니다.

35. iPad Bluetooth "ON"



Bluetooth 기능을 "ON" 시킵니다.

36. iPad Bluetooth 검색 진행



Bluetooth 기능이 ON되면 자동으로 주변의 Bluetooth 장치를 검색합니다.

검색된 Bluetooth 장치 리스트 중, "FB155BC_HID"가 있는지 확인합니다.

"FB155BC_HID"가 없으면, 다시 검색을 진행하거나 "FB155BC_HID"가 Scan 작업을 정확하게 수행 중인지 확인합니다.

37. FB155BC_HID를 선택하여 연결 진행

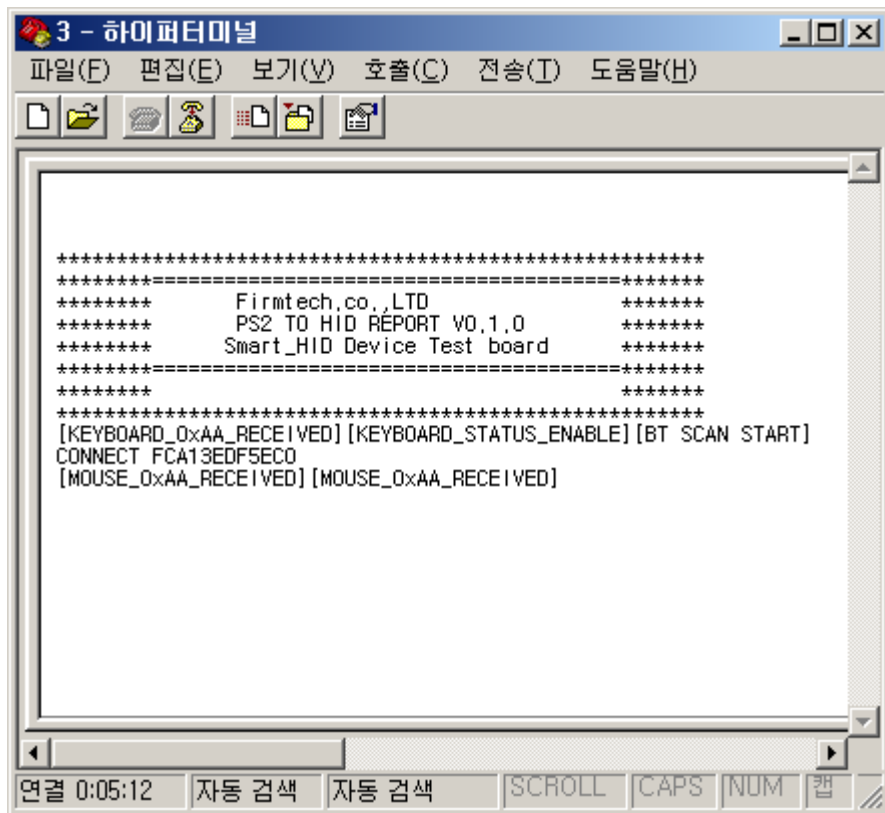


"FB155BC_HID"를 클릭하여 연결을 진행합니다.

38. FB155BC_HID와 연결 완료



39. Smart_HID Interface Board 연결 후 상태



“CONNECT xxxxxxxxxxxx”라는 메시지는 Bluetooth 장치가 연결된 것을 나타냅니다.

Bluetooth 장치가 연결되면 Mouse 초기설정을 진행합니다.

첫번째 출력되는 [MOUSE_0xAA_RECEIVED] 메시지는 Mouse에 최초 전원 인가시 발생하는 데이터에 의해 출력됩니다.

두번째 출력되는 [MOUSE_0xAA_RECEIVED] 메시지는 Mouse의 초기설정 진행 시 발생하는 데이터에 의해 출력됩니다.

40. iPad의 메모 실행



41. iPad의 메모 커서 활성화



메모를 실행시키고, 화면을 터치해서 커서가 활성화 된 상태를 만들어야 합니다.

iPad의 커서가 활성화 되어 있지 않으면 키보드의 입력이 준비되지 않은 상태입니다. 즉, 커서가 활성화 되지 않으면 키보드 입력이 진행되지 않습니다.

42. 무선 키보드 데이터 출력된 화면



43. 한영 전환



iPad는 기본적으로 영문을 출력합니다.

영문에서 한글 변환을 위해서는 무선 키보드 상에서 "Win"키와 "Space"키를 동시에 누르면 "**한영 변환 메뉴**"가 출력됩니다.("Win"키는 누른 상태 유지) 이 상태에서 다시 한번 "Space"키를 누르면 한글로 변환이 됩니다. (파란색 메뉴가 현재 선택된 언어 입니다.)

한글에서 영문 변환도 위와 같은 방법으로 진행합니다.

iPad(iPhone포함)는 무선 Mouse를 지원하지 않습니다. FB155BC_SMD(HID)를 이용하여 무선 Mouse 데이터를 송신해도 iPad에서는 아무런 동작을 하지 않습니다.

PS2_to_HID를 이용한

FB155BC_SMD(HID)

&

Smart_HID Host

(Windows XP Bluetooth Stack)

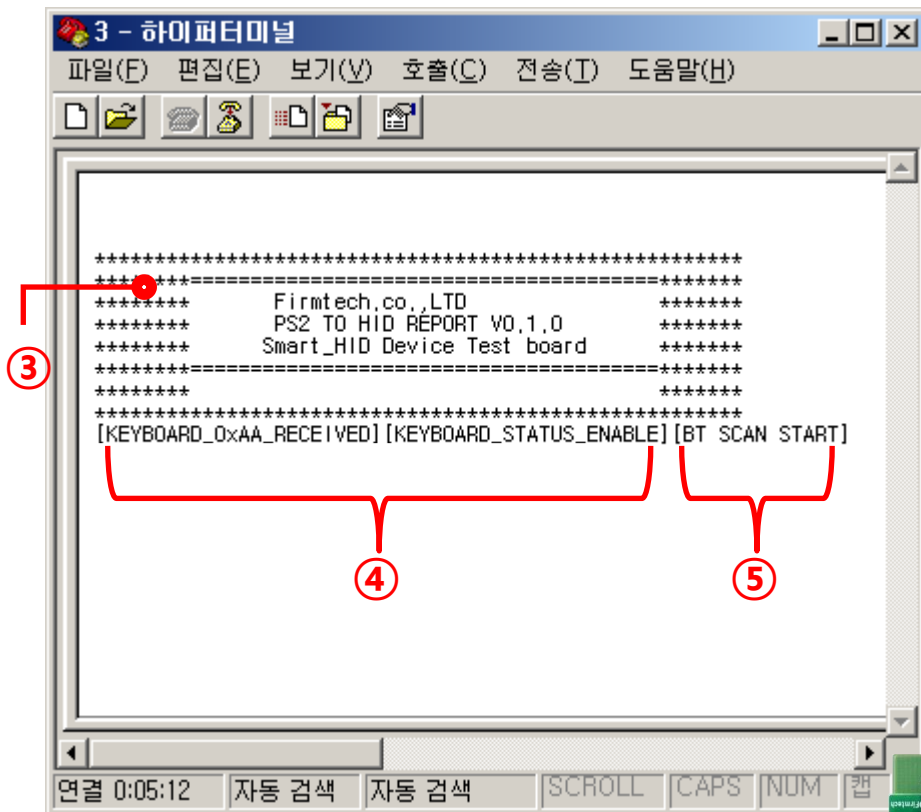
연결 진행

44. Smart_HID를 사용하기 위한 기본 사항

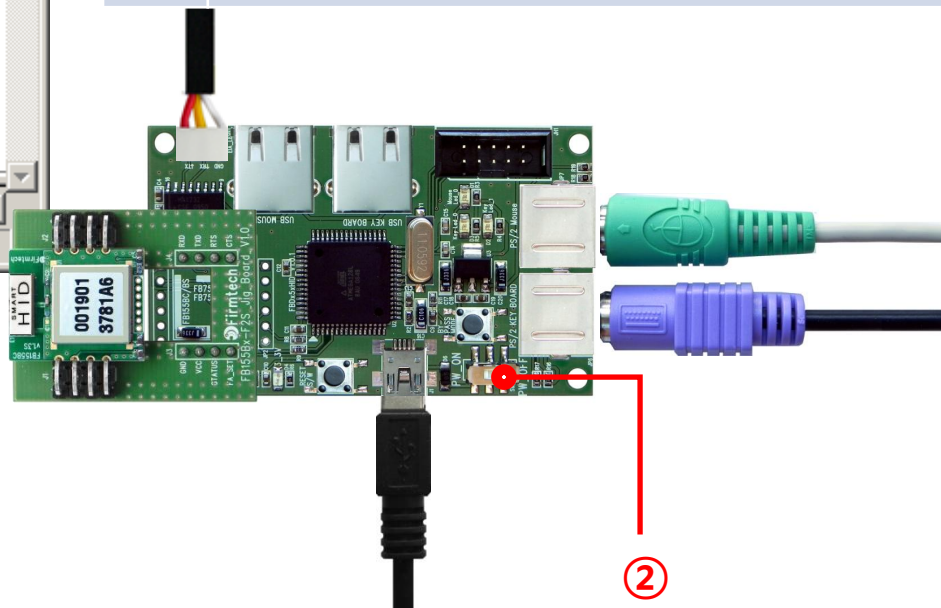
- Smart_HID Host는 **Windows XP에서 제공하는 Bluetooth Stack(동글)**을 사용합니다.
- FB155BC_SMD(HID)는 HID Device Profile이 운영되고 있어야 합니다.
- Smart_HID Interface Board는 **PS2_to_HID** 프로그램이 운영되고 있어야 합니다.
- Keyboard(또는 PS2-Scanner)와 Mouse가 있어야 합니다.

NO	Description
1	FB155BC_SMD(HID)는 Smart_HID Interface Board에 장착
2	Smart_HID Interface Board에 Keyboard와 Mouse 장착
3	Smart_HID Interface Board 전원 ON => FB155BC_SMD(HID) Scan 동작 확인
4	Smart_HID Host를 PC에 연결(동글) => FB155BC_SMD(HID) 검색
5	FB155BC_SMD(HID)와 연결 진행
6	PC의 HID Driver가 동작된것 확인
7	PC의 입력 프로그램(메모장 또는 Word 등) 실행
8	Smart_HID Interface Board와 연결된 Keyboard & Mouse 동작
9	PC의 입력 프로그램에서 문자 입력 & Mouse 동작 확인

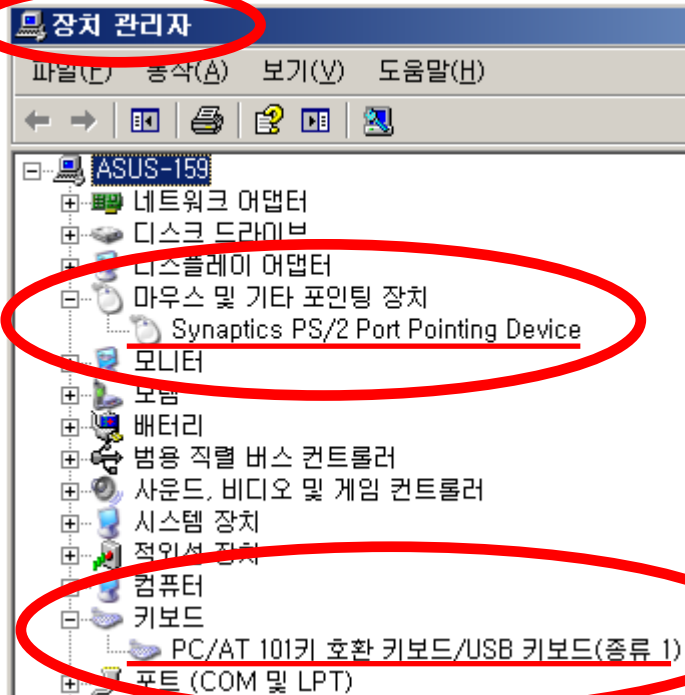
45. Smart_HID Interface Board 전원 ON



NO	연결 및 체크사항
1	FB155BC SMD(HID)는 Smart_HID Host가 동작되기 전에 SCAN작업을 진행해야 합니다. 그러므로 Smart_HID Interface Board의 전원 ON을 미리 진행해야 합니다.
2	FB155BC_SMD(HID)가 장착된 Smart_HID Interface Board 전원 ON
3	하이퍼 터미널에 운영 프로그램 정보 출력 - Smart_HID Interface 프로그램 운영 확인 가능
4	하이퍼 터미널에 Keyboard 상태 출력
5	하이퍼 터미널에 BT Module Scan 메시지 출력 (Smart_HID Interface Board의 Status LED는 깜빡거립니다.)



46. 키보드/마우스 장치관리자 확인



Windows XP Bluetooth Stack을 실행하기 전에 "장치관리자"에서 마우스와 키보드의 Driver를 확인 합니다.

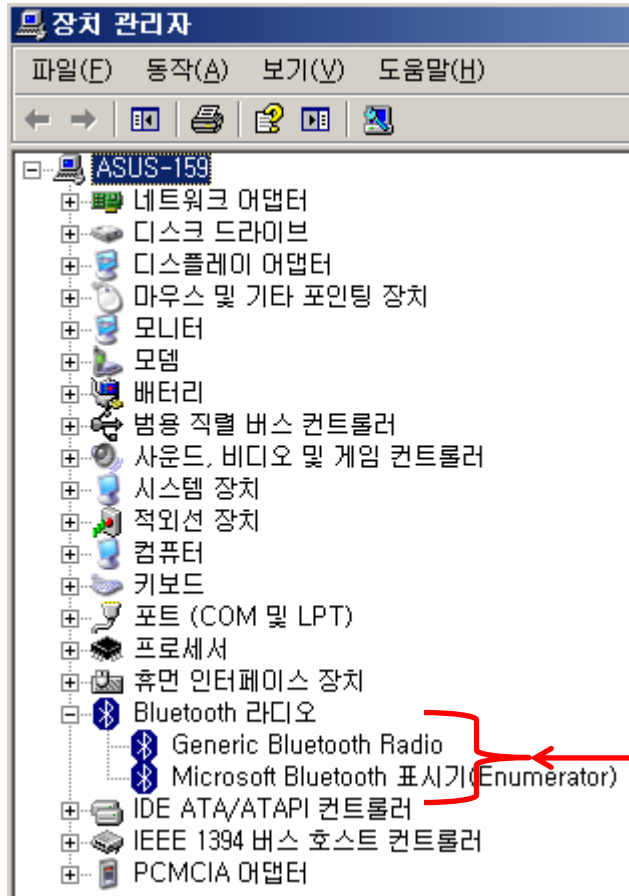
무선 키보드와 무선 마우스의 사용이 **불가능한 상태** 입니다.

47. Smart_HID Host와 PC 연결



① Smart_HID Host(동글)를 PC와 연결 합니다.

② PC의 오른쪽 하단에 Bluetooth 마크가 활성화 됩니다.



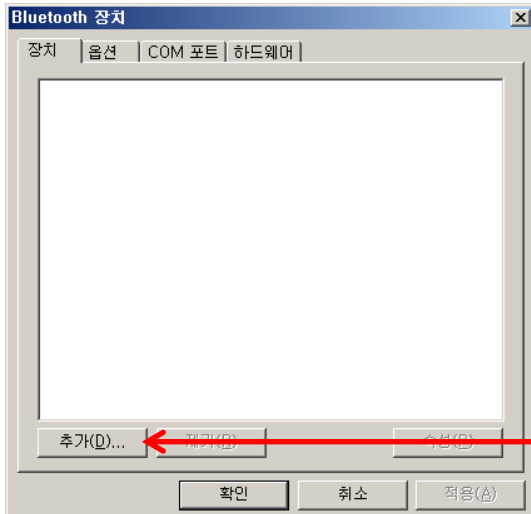
③ Windows XP Bluetooth Stack의 사용이 가능한 상태입니다. 장치관리자에서 Bluetooth 장치의 사용이 가능한지 한번 더 확인 합니다.

Smart_HID Host는 Windows XP에서 제공하는 Bluetooth Stack을 사용합니다. 기타 다른 상용 Bluetooth Stack을 사용하는 경우, 사용자가 적절히 연결을 진행해야 합니다. (기타 Bluetooth Stack의 특성상(버전 등) 연결이 진행되지 않는 Bluetooth Stack이 있을 수 있습니다.)

48. Windows XP Bluetooth Stack - Step 1

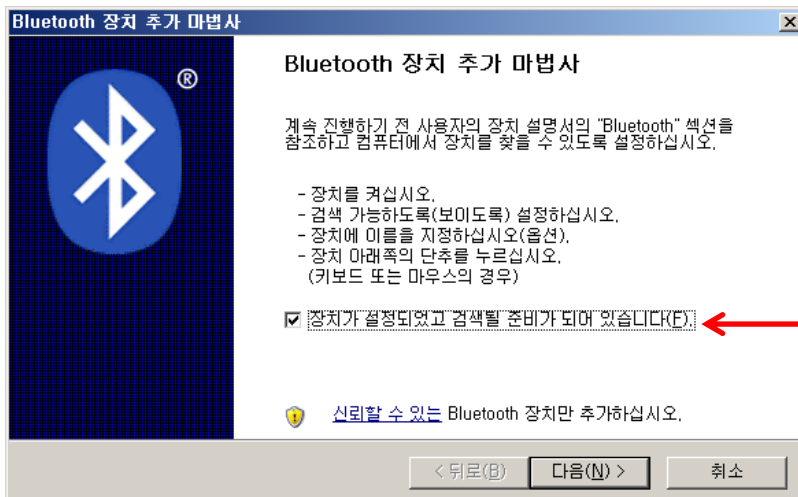


① Bluetooth 마크를 더블클릭 합니다.

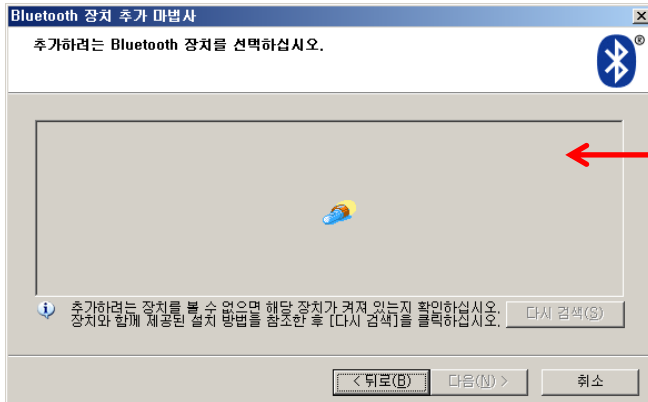


② "Bluetooth 장치"창이 나오면 "추가"버튼을 클릭합니다.

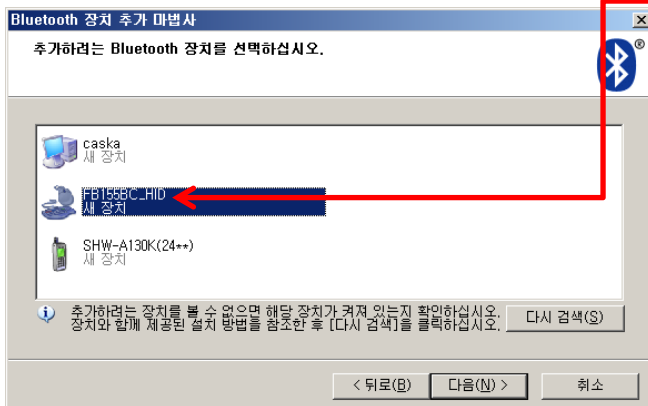
③ "Bluetooth 장치 추가 마법사"창이 나오면 "장치가 설정되었고 검색될 준비가 되어 있습니다."에 체크하고 "다음"버튼을 클릭합니다.



49. Windows XP Bluetooth Stack - Step 2

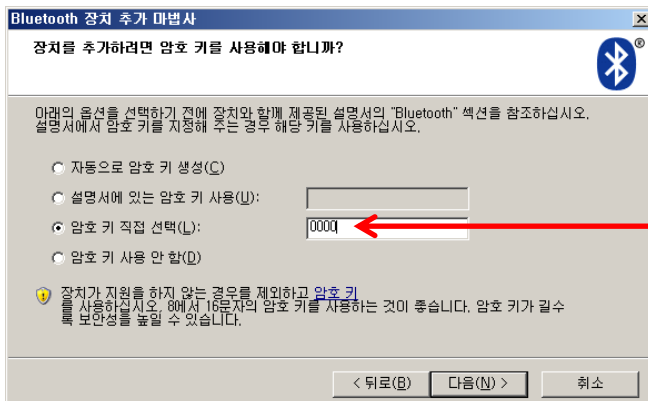


① 검색이 완료될 때까지 기다립니다.



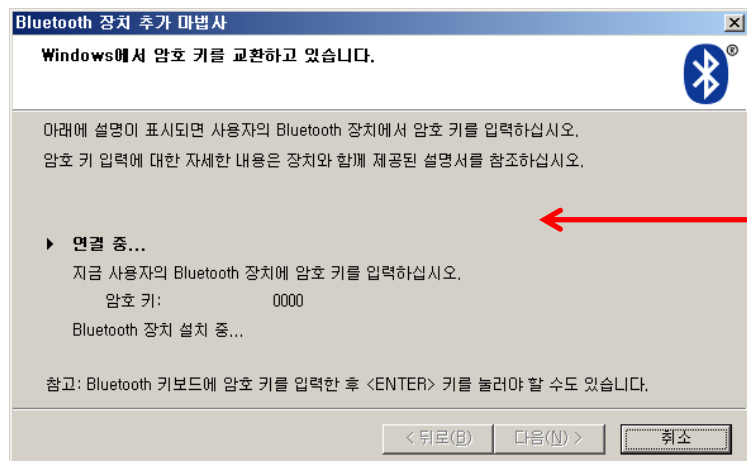
② 검색이 완료되면, “FB155BC_HID”라고 되어 있는 키보드/마우스 모양의 장치를 선택하고 “다음”버튼을 클릭합니다.

검색된 리스트에 “FB155BC_HID”가 없으면, 다시 검색을 진행하거나 “FB155BC_HID”가 Scan 작업을 정확하게 수행 중인지 확인합니다.

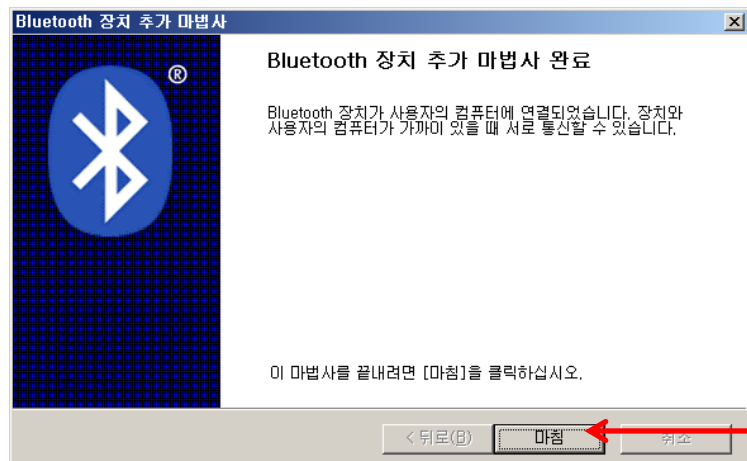


③ “암호 키 직접 선택”을 선택하고 “0000”을 입력합니다.
“다음”버튼을 클릭합니다.

50. Windows XP Bluetooth Stack - Step 3

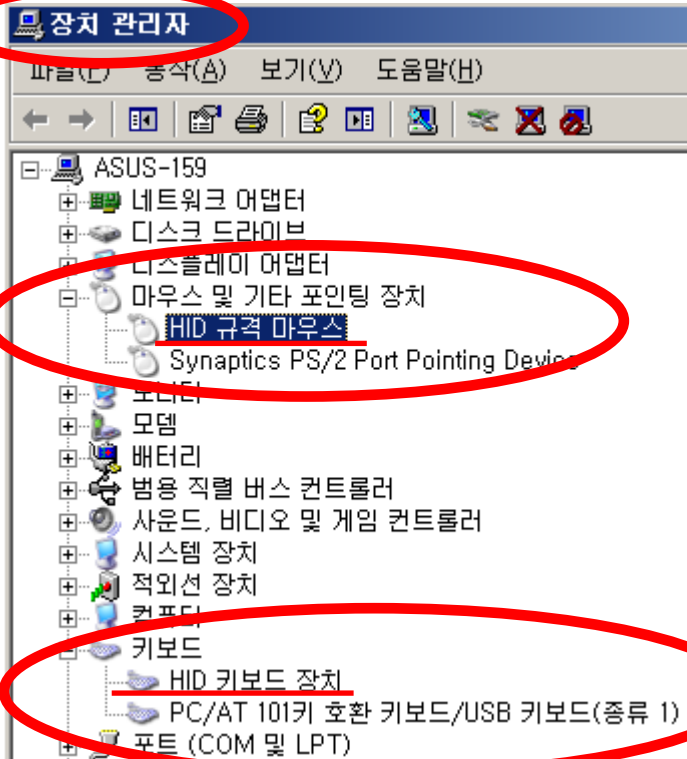


① 장치 설정이 완료될 때까지 기다립니다.



② "Bluetooth 장치 추가 마법사"가 완료되면 "마침"버튼을 클릭합니다.

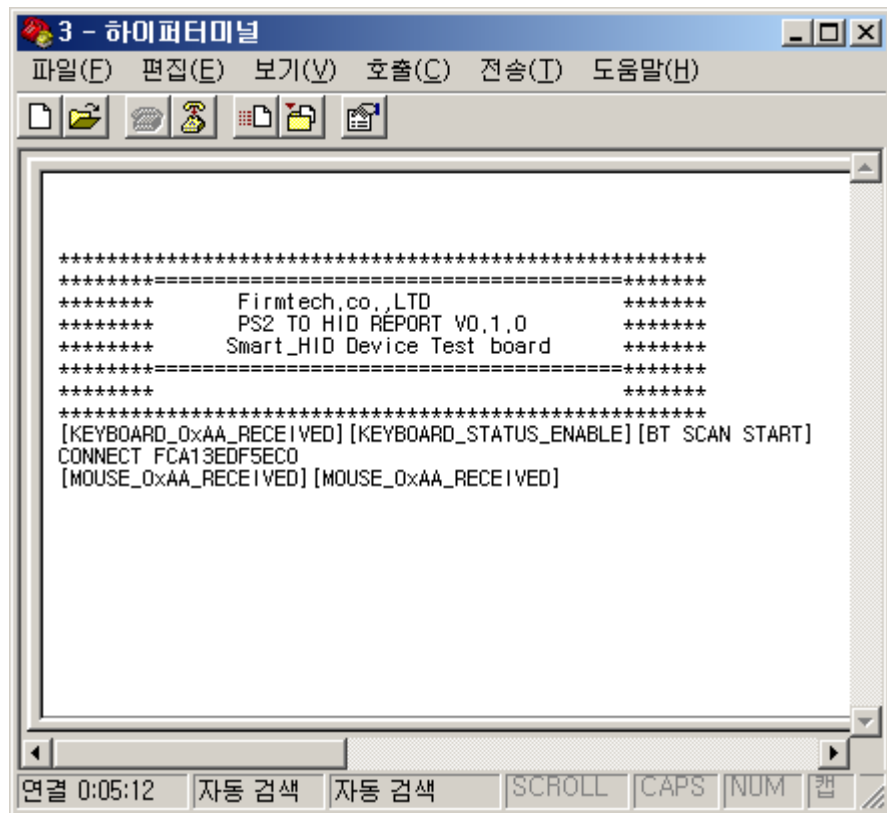
51. Bluetooth HID Driver 확인



“장치관리자”에서 마우스와 키보드의 Driver를 확인 합니다.

새로이 “HID 규격 마우스”와 “HID 키보드 장치”가 생성되어 있다면, 무선 키보드와 무선 마우스의 사용이 **가능한 상태** 입니다.

52. Smart_HID Interface Board 연결 후 상태



“CONNECT xxxxxxxxxxxx”라는 메시지는 Bluetooth 장치가 연결된 것을 나타냅니다.

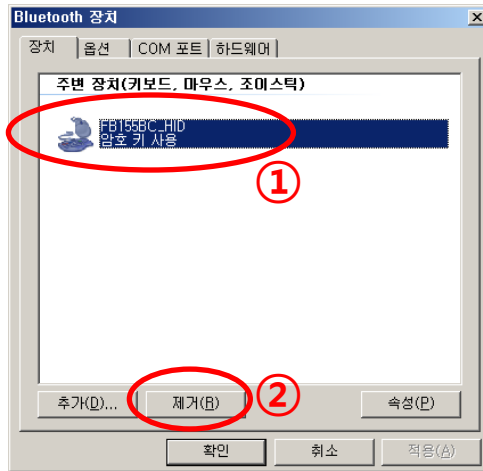
Bluetooth 장치가 연결되면 Mouse 초기설정을 진행합니다.

첫번째 출력되는 [MOUSE_0xAA_RECEIVED]메시지는 Mouse에 최초 전원 인가시 발생하는 데이터에 의해 출력됩니다.

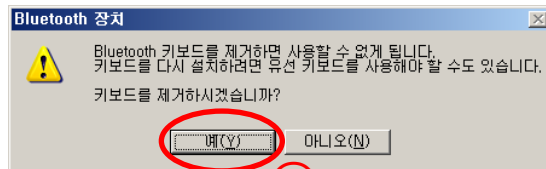
두번째 출력되는 [MOUSE_0xAA_RECEIVED]메시지는 Mouse의 초기설정 진행 시 발생하는 데이터에 의해 출력됩니다.

PC의 입력 프로그램(한글/워드 등)에서 문자 입력 & Mouse 동작 확인

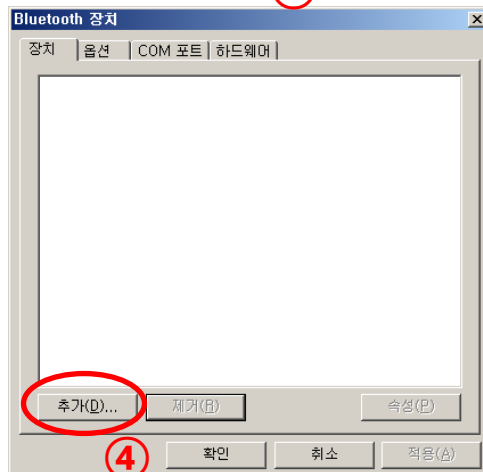
53. 재 연결을 위한 드라이버 제거



1. Bluetooth 장치의 연결을 종료하고 다시 연결하고자 하는 경우, 반드시 기존에 연결했던 Bluetooth 장치를 **"제거"**하고 **다시 "추가"**를 진행해야 합니다.



2. Windows XP에서 제공하는 Bluetooth Stack의 특성 상, 기존에 추가 했던 장치를 연결할 수 있는 방법이 없습니다. 반드시 **"제거"**를 진행하고 다시 연결을 진행해야 합니다.



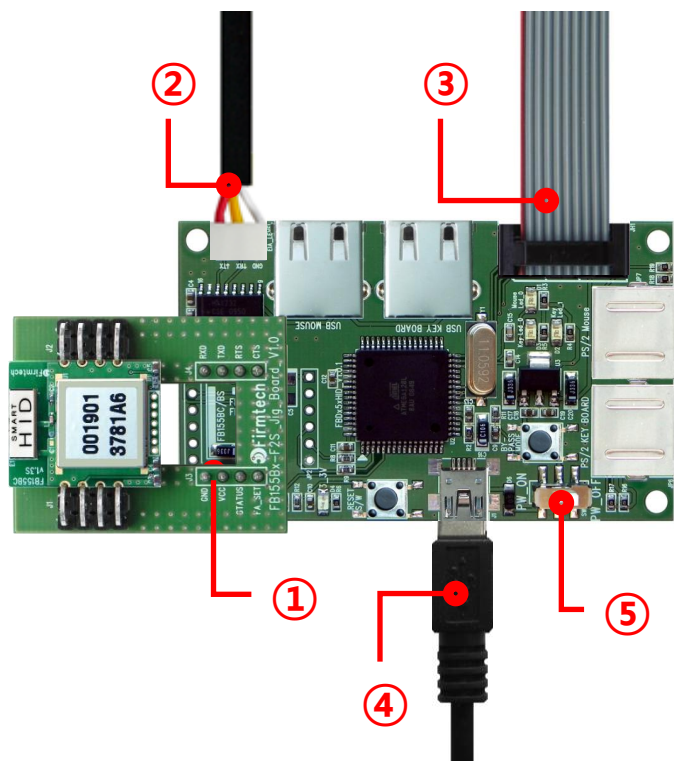
SERIAL_to_HID

HEX File Download

54. SERIAL_to_HID 프로그램 다운로드

(1) FB155BC_SMD(HID) 장착 및 Cable 연결

- Smart_HID Interface Board에서 운영되는 프로그램을 수정한 경우, 사용자는 ATmega128에 새로운 프로그램을 다운로드 해야 합니다.



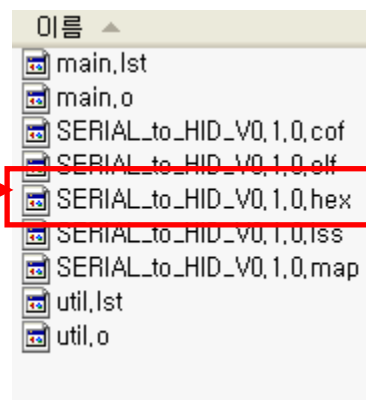
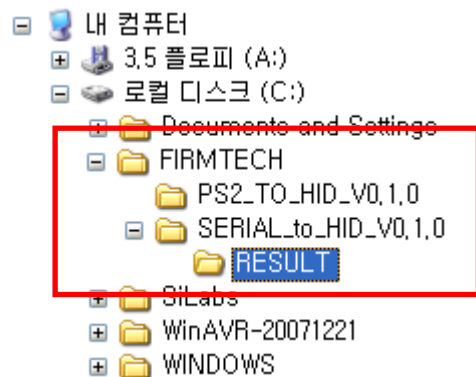
NO	연결 및 체크사항
1	Smart_HID Interface Board에 FB155BC_SMD(HID) 장착
2	Smart_HID Interface Board와 RS-232 Cable을 연결하여 PC와 연결
3	Smart_HID Interface Board와 AVR Loader(다운로드 케이블)연결하여 PC와 연결
4	Smart_HID Interface Board와 USB Power Cable을 연결하여 PC와 연결
5	Smart_HID Interface Board 전원 ON

54. SERIAL_to_HID 프로그램 다운로드

(2) SERIAL_to_HID.hex 파일 위치

CD로 제공되는 SERIAL_to_HID 프로그램은 사용자가 알맞은 폴더를 만들어 복사 합니다.

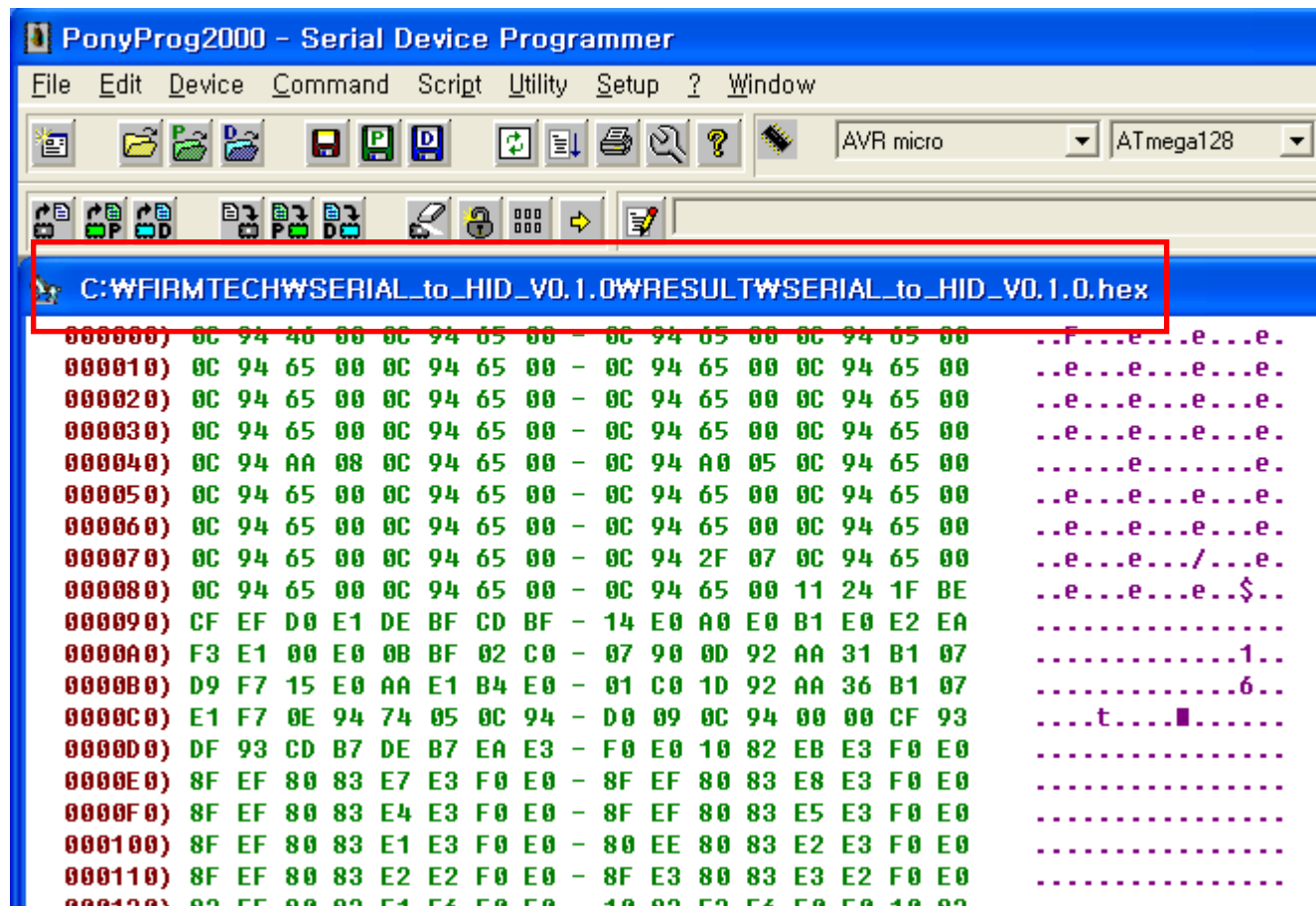
설명은 "C:\WFIRMTECH\SERIAL_to_HID\W"폴더로 설명합니다.



54. SERIAL_to_HID 프로그램 다운로드

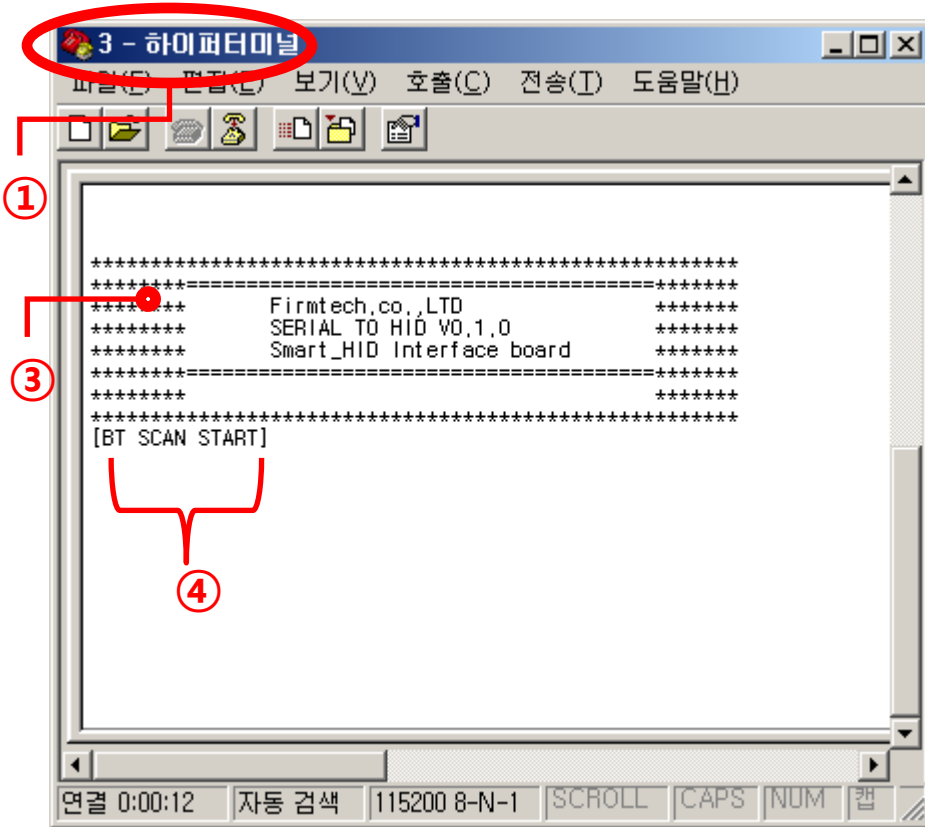
(3) PonyProg 프로그램을 사용하여 SERIAL_to_HID.hex 프로그램 다운로드

Tool에 대한 설명은 펌테크 홈페이지에서
"FB800ED_Tools Guide.PDF"파일을 참고 하십시오.

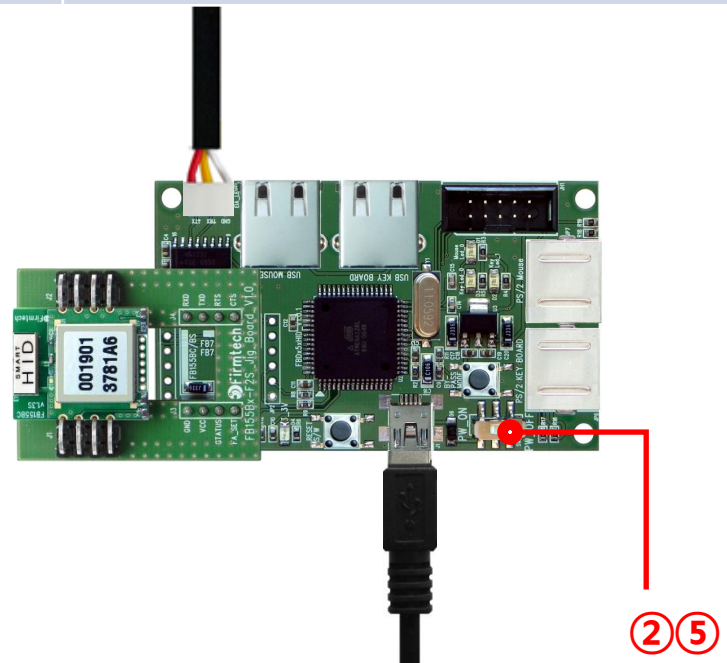


54. SERIAL_to_HID 프로그램 다운로드

(4) SERIAL_to_HID 프로그램 동작 확인



NO	연결 및 체크사항
1	프로그램 다운로드 완료된 Smart_HID Interface Board와 연결된 PC의 하이퍼 터미널 Open - PC의 포트는 3번을 사용하는 것으로 가정 - 통신 속도는 115200 선택 - 흐름제어 없음
2	Smart_HID Interface Board 전원 ON
3	하이퍼 터미널에 운영 프로그램 정보 출력 - Smart_HID Interface 프로그램 운영 확인 가능
4	하이퍼 터미널에 BT Module Scan 메시지 출력
5	Smart_HID Interface Board 전원 OFF



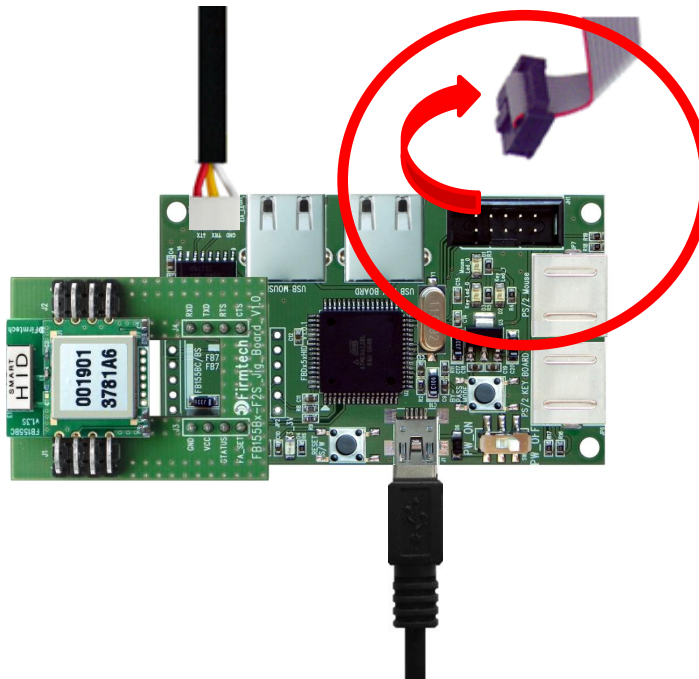
☞ 체크 사항

Smart_HID Interface Board의 전원을 ON 했을 때, FB155BC_SMD(HID)로부터 “BTWIN HID Slave mode start” 메시지를 수신하지 못하는 경우, AVR Loader가 PC와 연결되어 있는지 체크합니다.

(하이퍼 터미널에 [ERROR, STEP:2]라고 출력됨)

AVR Loader가 Smart_HID Interface Board와 PC에 연결되어 있는 상태에서, Ponyprog 프로그램이 PC에서 실행되어 있지 않으면, Smart_HID Device Module에서 출력되는 시리얼 데이터가 Smart_HID Interface Board에 정상적으로 수신되지 않습니다.

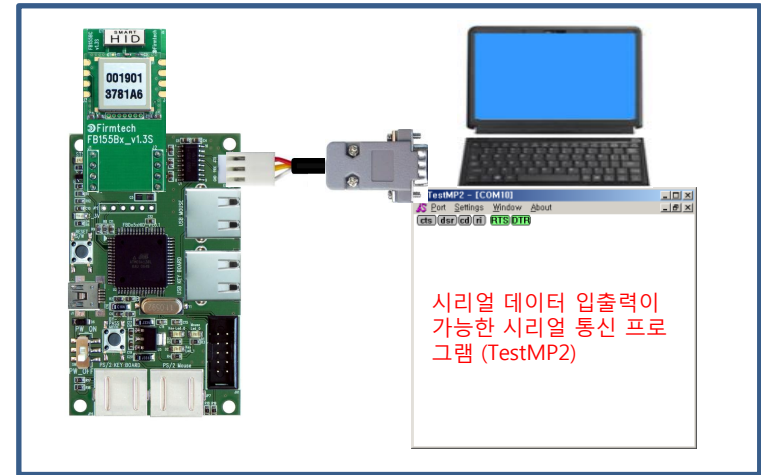
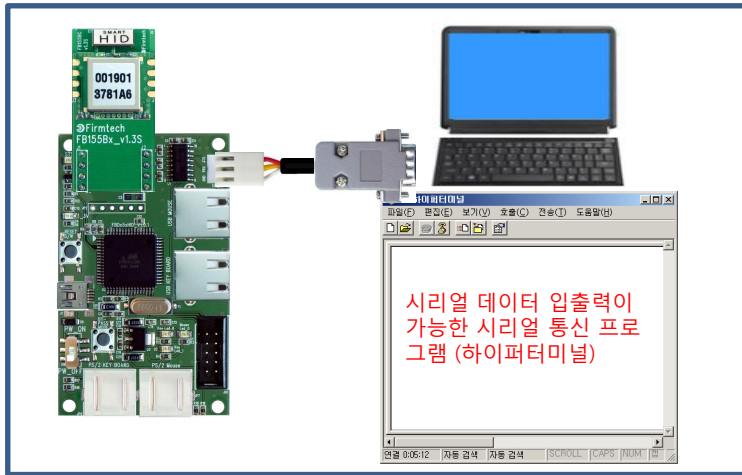
그러므로, 프로그램 다운로드가 완료되었다면 AVR Loader를 Smart_HID Interface Board와 연결하지 않습니다.



SERIAL_to_HID

시리얼 데이터 입력의 이해

55. 시리얼 데이터 입력의 이해



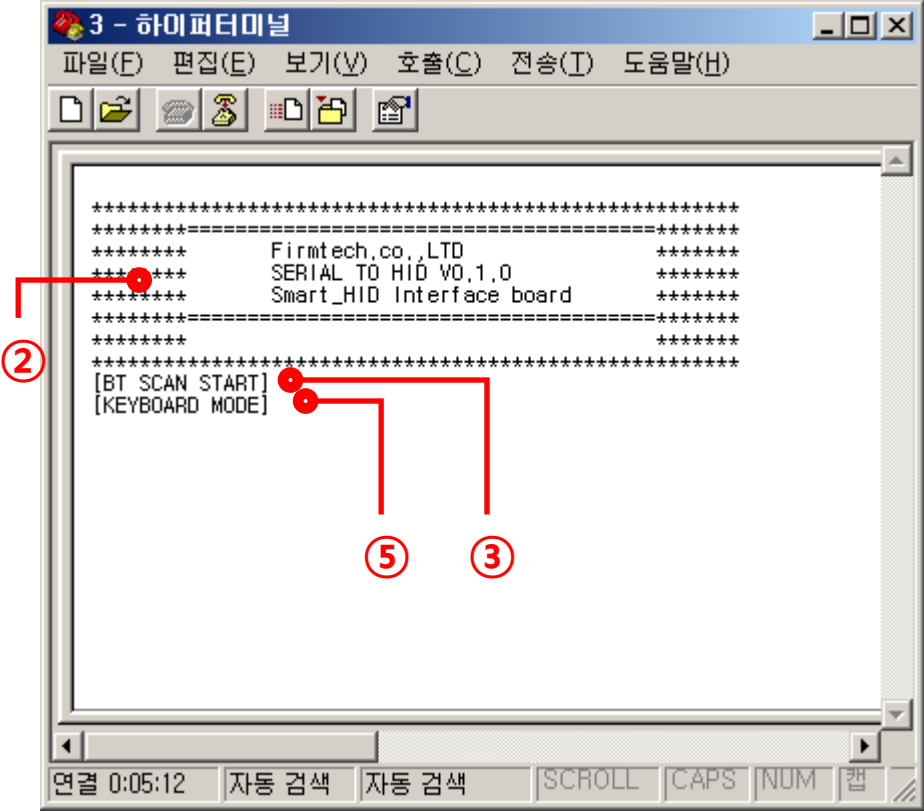
1. 하이퍼터미널에 데이터를 입력하면, 하이퍼터미널은 키보드로부터 입력된 데이터를 Smart_HID Interface보드로 송신합니다.
2. PC 키보드에서 입력 가능한 데이터는 ASCII Code Table의 0x00에서 0x7F 까지의 값입니다. 그러므로 ASCII Code Table에 정의 되어 있지 않은 데이터는 입력되지 않습니다.
3. 하이퍼터미널이라는 시리얼 통신 프로그램은 키보드로부터 입력 받은 데이터를 그대로 송신하거나, 특수한 형태로 변경하여 송신을 진행합니다.
4. PC 키보드의 데이터 입력 형태를 한글로 변경하여 하이퍼터미널에 데이터를 입력하면, 하이퍼터미널은 입력된 키보드의 데이터를 0x7F 이상의 2바이트 값으로 재 해석하여 송신하게 됩니다. (Smart_HID Interface에 데이터를 입력하는 경우, **키보드의 데이터 입력 형태는 반드시 영문으로만 입력해야 합니다.**)
5. ASCII Code Table에 정의되어 있지 않은 데이터(방향키, Caps Lock, F1 등등)는 Smart_HID Interface Board에 입력 되지 않습니다. 그러나 하이퍼터미널의 특성상 입력되는 몇 가지의 키가 있으나, 정상적인 ASCII Code 값이 아닌 하이퍼터미널만의 특수한 조합으로 이루어 집니다. (위 방향 화살표=> 0x1B, 0x5B, 0x41, F1 => 0x1B, 0x4F, 0x50)
6. 하이퍼터미널에서 특수한 조합으로 이루어지는 값은 다른 시리얼통신프로그램에서는 입력이 이루어 지지 않습니다. (펄테크 홈페이지에서 제공되는 TestMP2에서 확인 가능)

56. 하이퍼터미널에 입력된 시리얼 데이터를 확인하기 위한 기본 사항

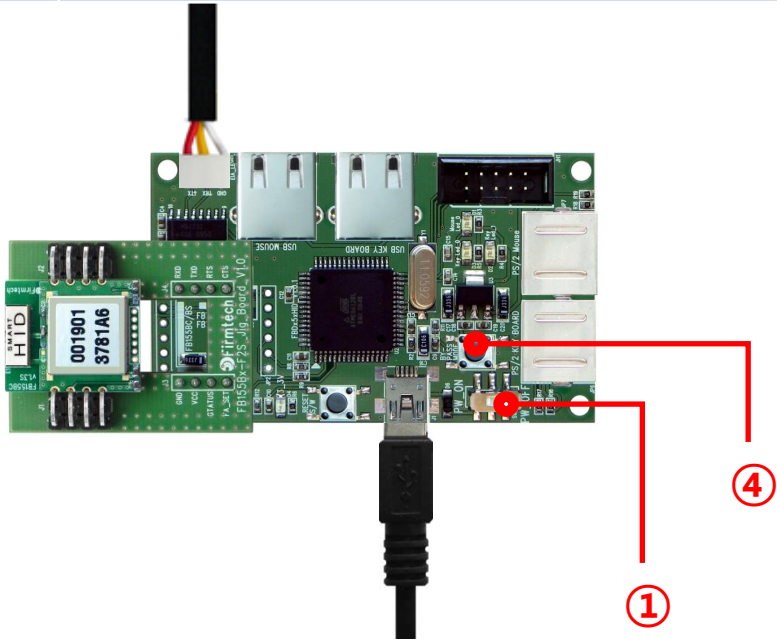
- FB155BC_SMD(HID)는 HID Device Profile이 운영되고 있어야 합니다.
- Smart_HID Interface Board는 **SERIAL_to_HID** 프로그램이 운영되고 있어야 합니다.
- 하이퍼터미널과 같은 시리얼 입력 프로그램이 있어야 합니다.
- 하이퍼터미널을 사용하여 Serial Data를 입력합니다.

NO	Description
1	FB155BC_SMD(HID)를 Smart_HID Interface Board에 장착
2	Smart_HID Interface Board 전원 ON => FB155BC_SMD(HID) Scan 동작 확인
3	Smart_HID Interface Board의 Bypass Switch 누름 => Keyboard Mode 확인
4	하이퍼터미널을 사용하여 Serial Data 입력

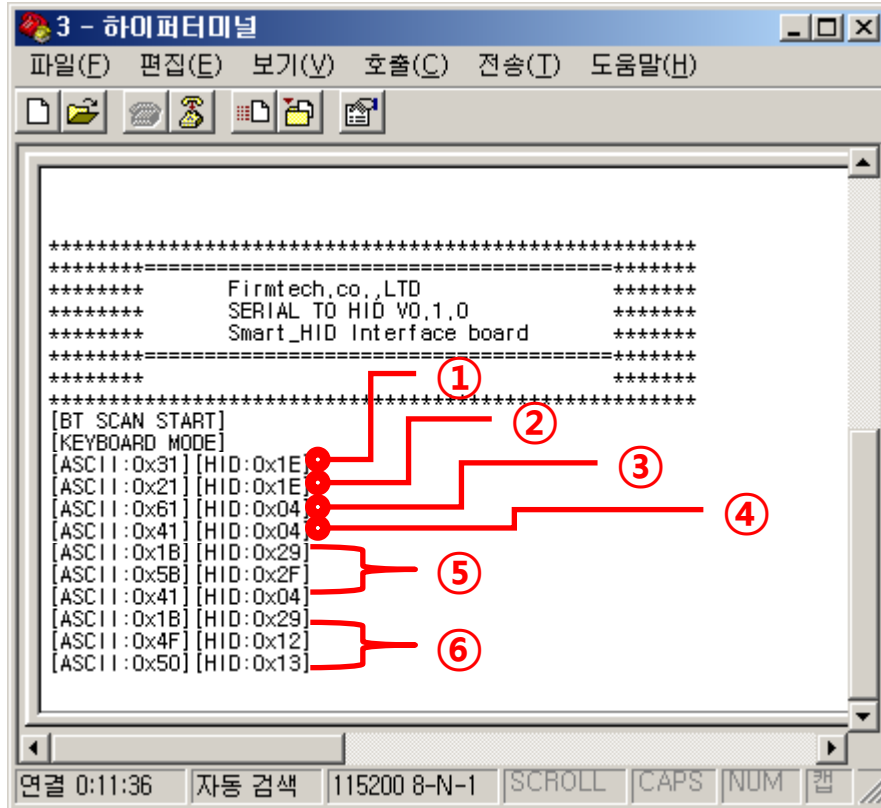
57. Keyboard Mode 진입하기



NO	연결 및 체크사항
1	FB155BC_SMD(HID)가 장착된 Smart_HID Interface Board 전원 ON
2	하이퍼 터미널에 운영 프로그램 정보 출력 - Smart_HID Interface 프로그램 운영 확인 가능
3	하이퍼 터미널에 BT Module Scan 메시지 출력 (Smart_HID Interface Board의 Status LED는 깜빡거립니다.)
4	Bypass Switch를 누릅니다.
5	FB155BC_SMD(HID)의 Scan 동작이 멈추고, Keyboard Mode가 동작됩니다.



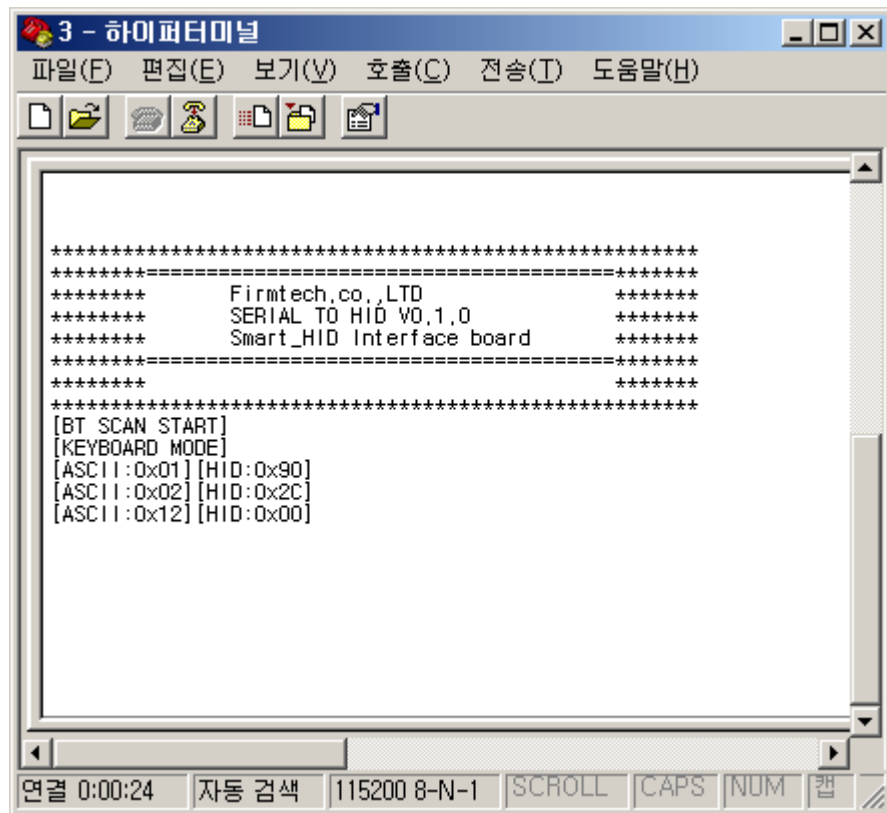
58. 하이퍼터미널을 이용한 Serial Data 입력



NO	연결 및 체크사항
1	하이퍼터미널에 키보드를 이용하여 1을 입력 합니다. => 1의 Hex값인 0x31과 HID Report값인 0x1E 출력
2	하이퍼터미널에 키보드를 이용하여 !를 입력 합니다. => !의 Hex값인 0x21과 HID Report값인 0x1E 출력
3	하이퍼터미널에 키보드를 이용하여 a를 입력 합니다. => a의 Hex값인 0x61과 HID Report값인 0x04 출력
4	하이퍼터미널에 키보드를 이용하여 A를 입력 합니다. => A의 Hex값인 0x41과 HID Report값인 0x04 출력
5	하이퍼터미널에 키보드를 이용하여 위 방향 화살표를 입력 합니다. => 하이퍼터미널에서 임의적으로 조합된 Hex값인 0x1B, 0x5B, 0x41 출력 (HID Report 값은 의미 없음)
6	하이퍼터미널에 키보드를 이용하여 F1키를 입력 합니다. => 하이퍼터미널에서 임의적으로 조합된 Hex값인 0x1B, 0x4F, 0x50 출력 (HID Report 값은 의미 없음)

1. 하이퍼터미널은 키보드로부터 입력된 데이터를 Smart_HID Interface보드로 송신합니다.
2. Smart_HID Interface보드는 입력된 시리얼 데이터(ASCII)를 리턴하고, 그에 해당하는 HID Report 값을 출력합니다.
3. 하이퍼터미널이라는 시리얼 통신 프로그램은 키보드로부터 입력 받은 데이터를 그대로 송신하거나, 특수한 형태로 변경하여 송신을 진행합니다.
4. 하이퍼터미널의 입력 방식은 영문으로만 입력해야 합니다.

59. 한영 전환을 위한 Serial Data



HID_Host의 한영 전환을 위해서는 특정한 HID_Report값을 송신해야 합니다.

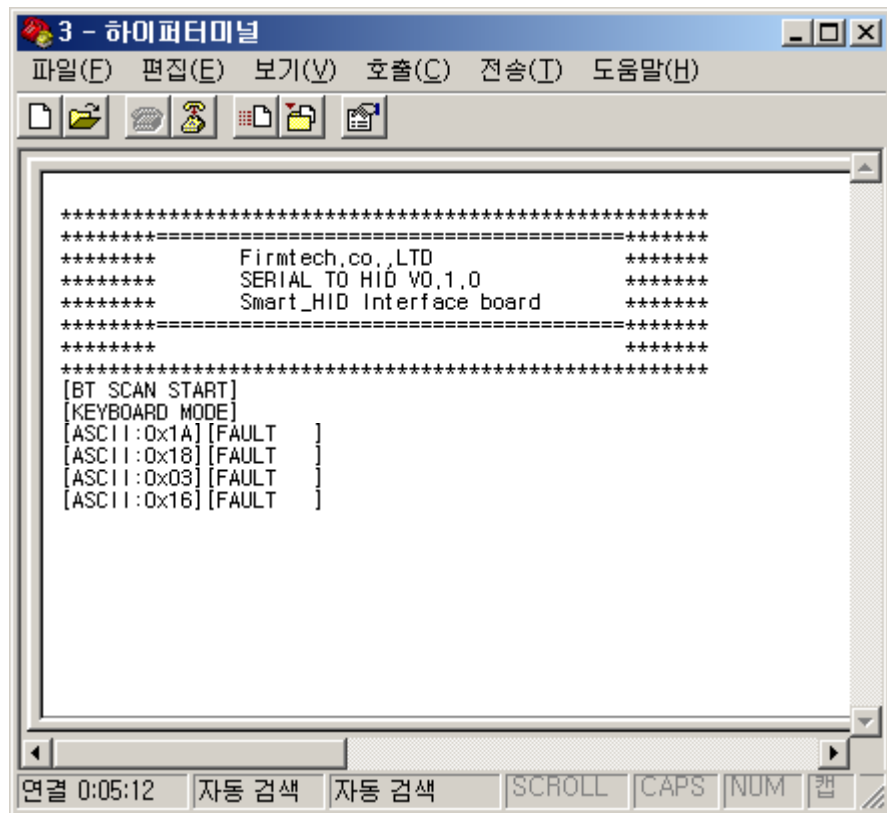
Galaxy-S또는 PC로 설정된 HID_Host에는 HID_Report 0x90을 송신해야 한영 전환이 이루어 집니다.

iPhone 또는 iPad로 설정된 HID_Host에는 HID_Report 0x2C를 송신해야 한영 전환이 이루어 집니다.

Note Book으로 설정된 HID_Host에는 HID_Report Alt를 송신해야 한영 전환이 이루어 지는 종류가 있습니다.

1. 하이퍼터미널을 사용하여 HID_Report 0x90을 만들기 위해서는 "Ctrl+a"를 입력합니다. "Ctrl+a"를 입력하면 Hex값 0x01이 입력되고, 0x01에 의해 HID Report 0x90이 출력됩니다.
2. 하이퍼터미널을 사용하여 HID_Report 0x2C를 만들기 위해서는 "Ctrl+b"를 입력합니다. "Ctrl+b"를 입력하면 Hex값 0x02가 입력되고, 0x02에 의해 HID_Report 0x2C가 출력됩니다.
3. 하이퍼터미널을 사용하여 HID_Report Alt를 만들기 위해서는 "Ctrl+r"을 입력합니다. "Ctrl+r"을 입력하면 Hex값 0x12가 입력되고, 0x12에 의해 Alt가 생성됩니다. (출력되는 0x00은 의미가 없습니다.)

60. ASCII Code Table 이외의 데이터 생성



```
*****
*****Firmtech,co.,LTD*****
*****SERIAL TO HID V0.1.0*****
*****Smart_HID Interface board*****
*****
*****
[BT SCAN START]
[KEYBOARD MODE]
[ASCII:0x1A] [FAULT]
[ASCII:0x18] [FAULT]
[ASCII:0x03] [FAULT]
[ASCII:0x16] [FAULT]
```

Serial Data를 HID_Report로 변경하는 경우, ASCII Code Table에 구성된 코드값 이외의 데이터는 HID_Report로 변경하기가 어렵습니다.

그러나, 0x00 ~ 0x1F 사이의 값을 이용하여 필요한 HID_Report로 변경하는 것은 가능합니다.

한영 전환을 위한 HID_Report를 "0x01(Ctrl+a)"와 "0x02(Ctrl+b)"로 구성한 것과 같은 방식으로, 특정한 HID_Report의 구성이 가능합니다.

제공되는 샘플 소스에는 몇 개의 키를 제외하고는 0x00 ~ 0x1F 사이의 값이 HID_Report와 맵핑이 되어 있지 않습니다.

하이퍼터미널에 "0x1A(Ctrl+z)", "0x18(Ctrl+x)", "0x03(Ctrl+c)", "0x16(Ctrl+v)"를 입력하면 "[FAULT]"라는 메시지를 출력하는데, 이것은 샘플소스에서 HID_Report와 맵핑이 되지 않은 Hex값이 입력된 것을 나타냅니다.

SERIAL_to_HID를 이용한

FB155BC_SMD(HID)

&

Smart_HID Host

(PHONE: Galaxy-S)

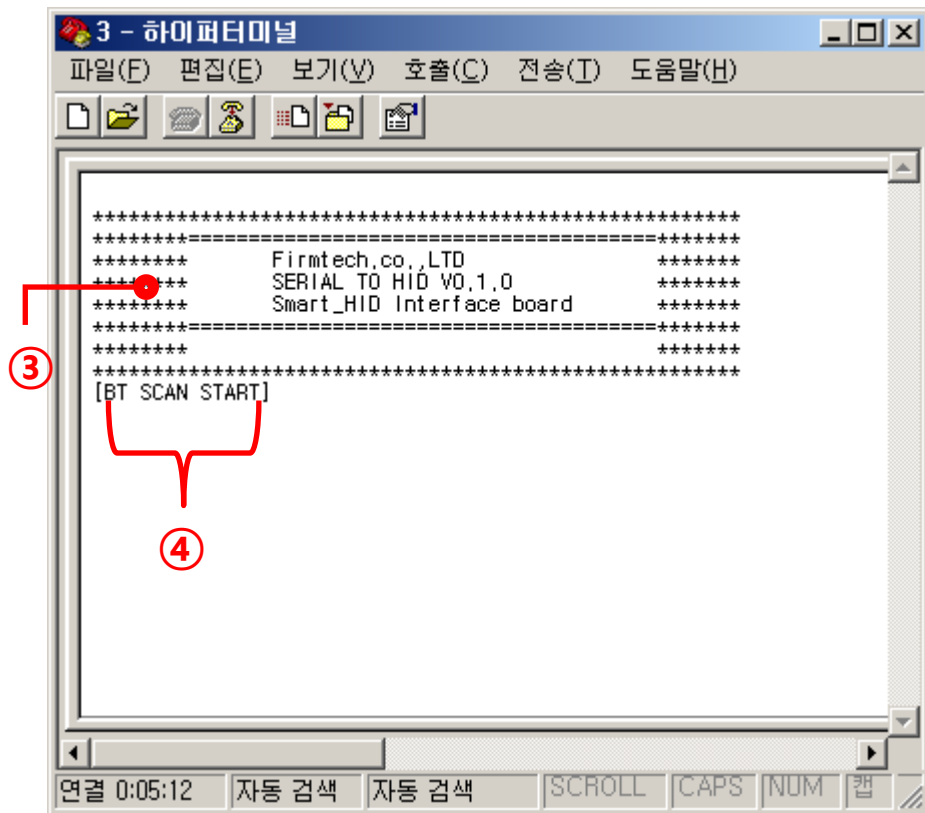
연결 진행

61. Smart_HID를 사용하기 위한 기본 사항

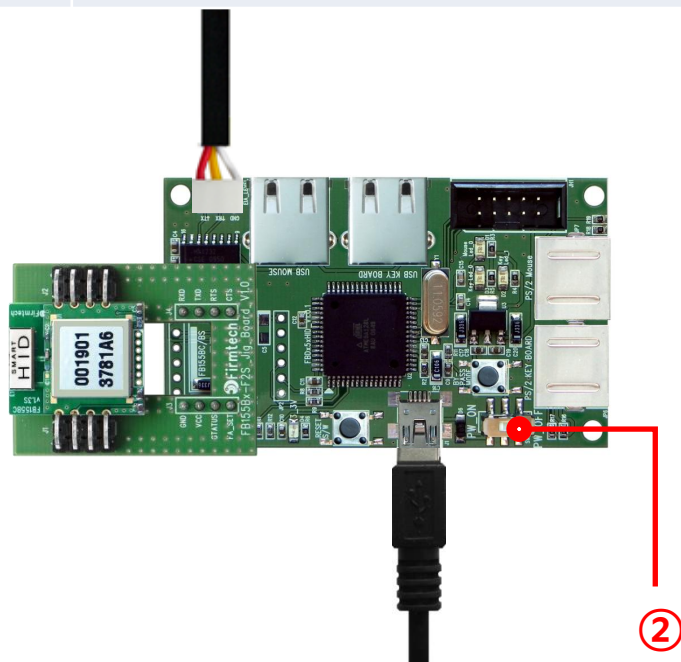
- Smart_HID Host는 **PHONE(Galaxy-S)**을 사용합니다.
- FB155BC_SMD(HID)는 HID Device Profile이 운영되고 있어야 합니다.
- Smart_HID Interface Board는 **SERIAL_to_HID** 프로그램이 운영되고 있어야 합니다.
- 하이퍼터미널을 사용하여 Serial Data를 입력합니다.

NO	Description
1	FB155BC_SMD(HID)를 Smart_HID Interface Board에 장착
2	Smart_HID Interface Board 전원 ON => FB155BC_SMD(HID) Scan 동작 확인
3	Galaxy-S 메인메뉴=>환경설정=>무선 및 네트워크=>블루투스 설정=>검색=>연결
4	Galaxy-S의 메모 실행
5	하이퍼터미널을 사용하여 Serial Data 입력
6	Galaxy-S의 메모에서 문자 입력 확인

62. Smart_HID Interface Board 전원 ON



NO	연결 및 체크사항
1	FB155BC_SMD(HID)는 Galaxy-S가 동작되기 전에 SCAN 작업을 진행해야 합니다. 그러므로 Smart_HID Interface Board의 전원 ON을 미리 진행해야 합니다.
2	FB155BC_SMD(HID)가 장착된 Smart_HID Interface Board 전원 ON
3	하이퍼 터미널에 운영 프로그램 정보 출력 - Smart_HID Interface 프로그램 운영 확인 가능
4	하이퍼 터미널에 BT Module Scan 메시지 출력 (Smart_HID Interface Board의 Status LED는 깜빡거립니다.)



63. Galaxy-S “메인메뉴” 선택



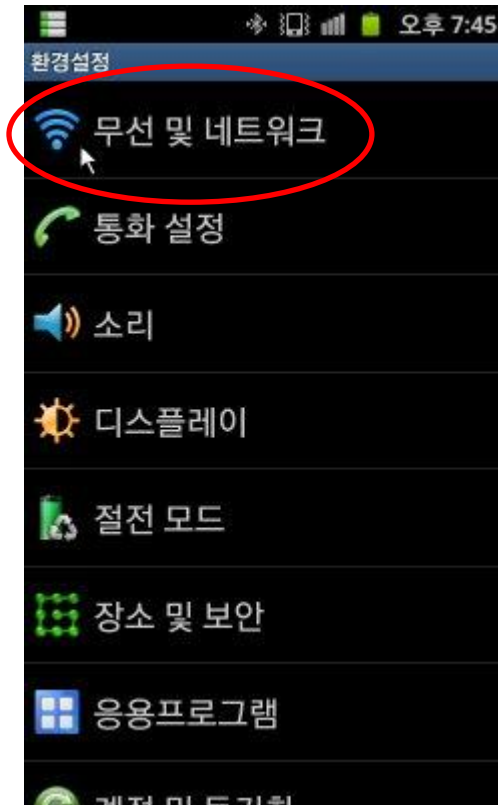
Galaxy-S 화면상에서 “메인메뉴” 아이콘을 클릭합니다.

64. Galaxy-S “환경설정” 선택



메인메뉴 중 “환경설정” 아이콘을 클릭합니다.

65. Galaxy-S “무선 및 네트워크” 선택



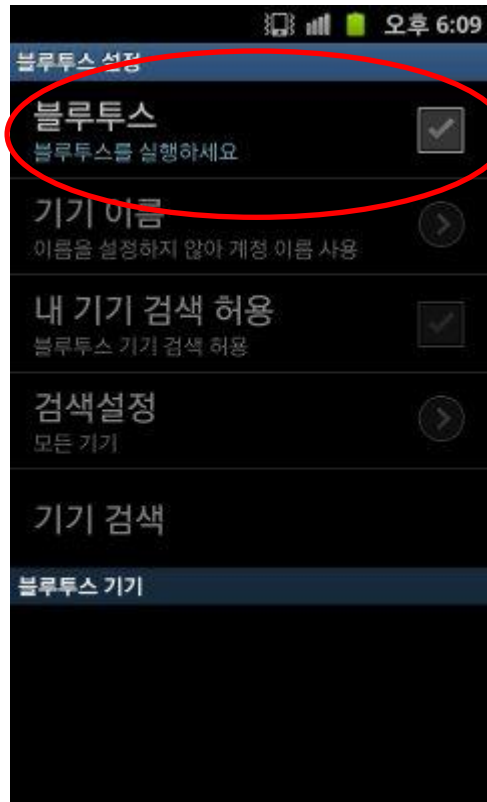
환경설정 메뉴 중 “무선 및 네트워크” 아이콘을 클릭합니다.

66. Galaxy-S “블루투스 설정” 선택



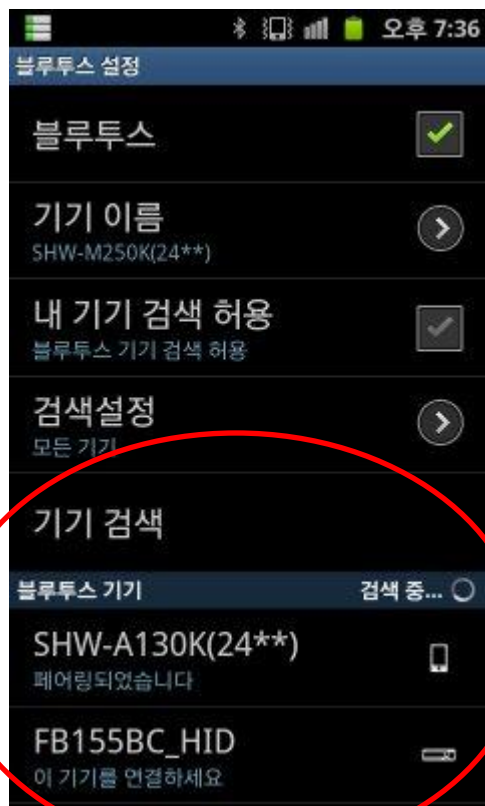
무선 및 네트워크 메뉴 중 “블루투스 설정” 아이콘을 클릭합니다.

67. Galaxy-S Bluetooth “ON”



Bluetooth 기능을 “ON” 시킵니다.

68. Galaxy-S Bluetooth 검색 진행



Bluetooth 기능이 ON되면 자동으로 주변의 Bluetooth 장치를 검색합니다.

"기기 검색"을 클릭하면 수동으로 주변의 Bluetooth 장치를 검색합니다.

검색된 Bluetooth 장치 리스트 중,
"FB155BC_HID"가 있는지 확인합니다.

"FB155BC_HID"가 없으면, 다시 검색을 진행하거나 "FB155BC_HID"가 Scan 작업을 정확하게 수행 중인지 확인합니다.

69. FB155BC_HID를 선택하여 연결 진행



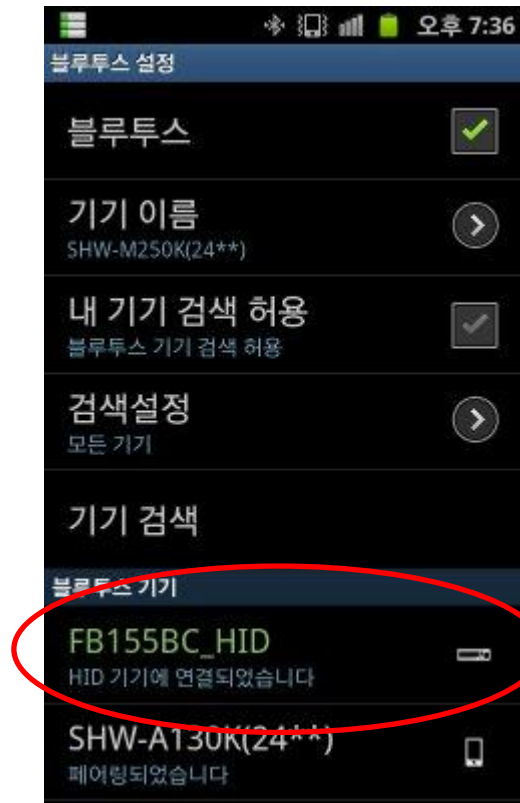
"FB155BC_HID"를 클릭하여 연결을 진행합니다.

70. 연결 요청

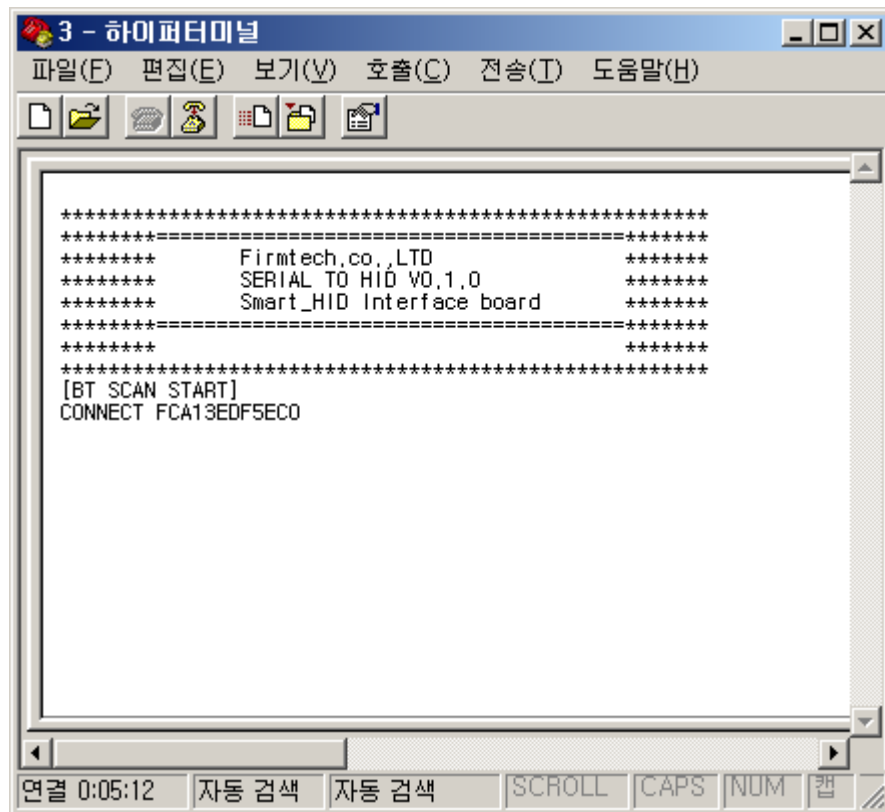


블루투스 연결 요청 화면이 나오는 경우
"연결"을 클릭하여 연결을 진행합니다.

71. FB155BC_HID와 연결 완료



72. Smart_HID Interface Board 연결 후 상태

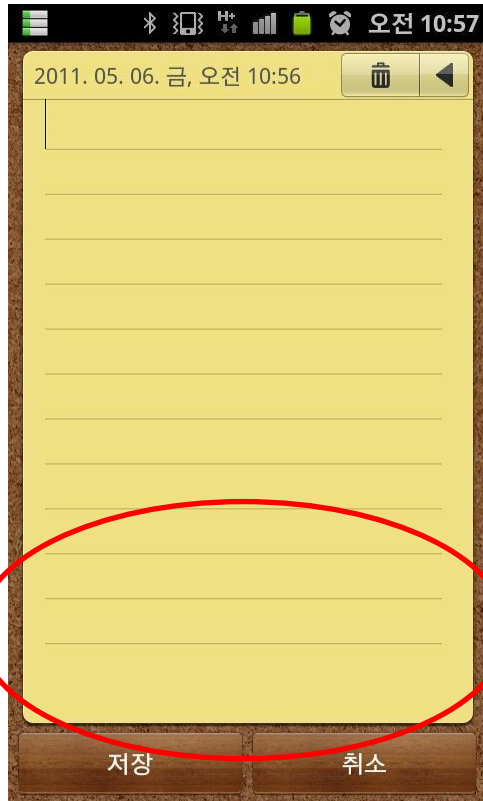


"CONNECT xxxxxxxxxxxx"라는 메시지는 Bluetooth 장치가 연결된 것을 나타냅니다.

73. Galaxy-S의 메모 실행



74. Galaxy-S의 자판 활성화

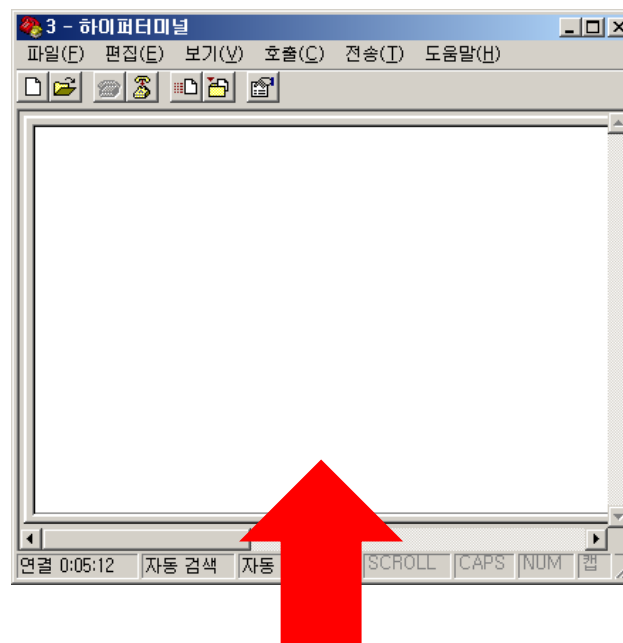


메모를 실행시키고, 화면을 터치해서 자판이 활성화 된 상태를 만들어야 합니다.

Galaxy-S의 자판이 활성화 되어 있지 않으면 한/영의 입력이 정상적으로 진행되지 않는 경우가 있습니다.

숫자만 입력이 되는 경우, 반드시 자판을 1회 이상 활성화 시켜야 합니다.

75. 무선 시리얼 데이터 출력된 화면

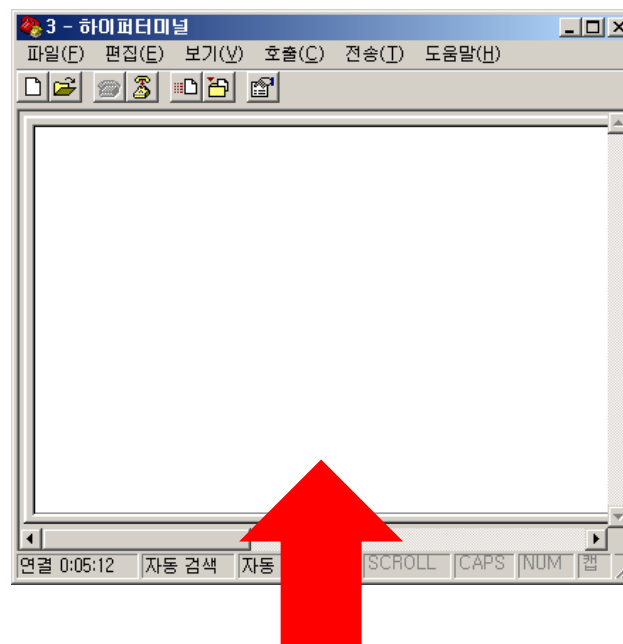


PC와 연결된 Keyboard를 사용하여 하이퍼터미널에 시리얼 데이터를 입력합니다.

"펌테크 입니다."를 입력하면 Galaxy-s에 문자가 출력됩니다.

하이퍼터미널에 데이터를 입력할 때, PC의 입력방식은 반드시 영문으로 되어 있어야 합니다.

76. 한영 전환



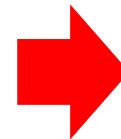
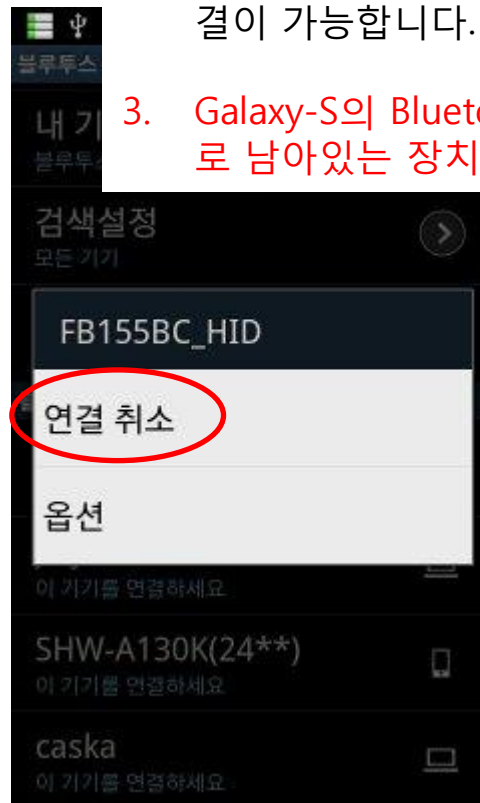
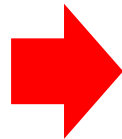
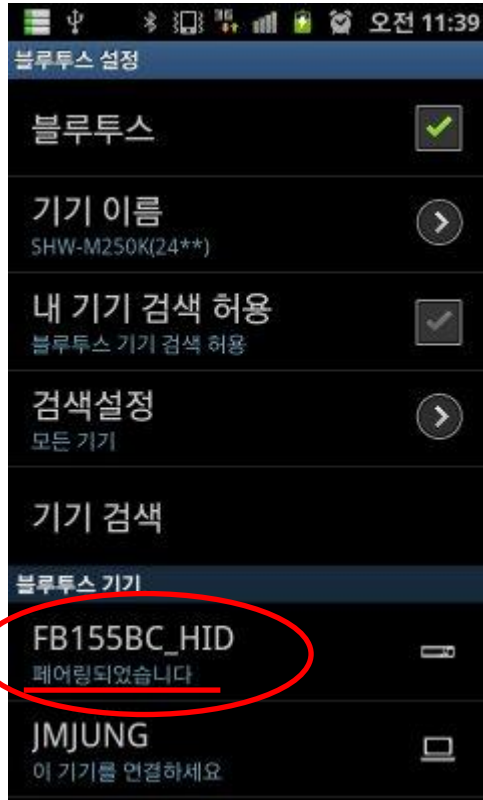
한/영 변환을 위해서는, PC와 연결된 Keyboard를 사용하여 하이퍼터미널에 "0x01(Ctrl+a)"를 입력합니다.

"Firmtech"를 입력하면 Galaxy-s에 문자가 출력됩니다.

하이퍼터미널에 데이터를 입력할 때, PC의 입력방식은 반드시 영문으로 되어 있어야 합니다.

77. 재 연결을 위한 Bluetooth 기기 상태 체크

1. Galaxy-S의 경우, Bluetooth 장치의 연결 종료를 Galaxy-S에서 진행하지 않고 상대방 장치에서 진행하는 경우(전원 OFF등), 상대방 장치의 상태가 "페어링되었습니다"로 나타납니다.
2. 상대방 장치의 상태가 "페어링되었습니다"로 되어 있는 경우 재 연결이 진행되지 않습니다. 이 경우, 리스트에서 해당 장치를 길게 클릭하여 "연결 취소"를 진행해야 재 연결이 가능합니다.
3. Galaxy-S의 Bluetooth Stack의 특성상, "페어링되었습니다"로 남아있는 장치와는 재 연결이 불가능 합니다.



SERIAL_to_HID를 이용한

FB155BC_SMD(HID)

&

Smart_HID Host

(iPad)

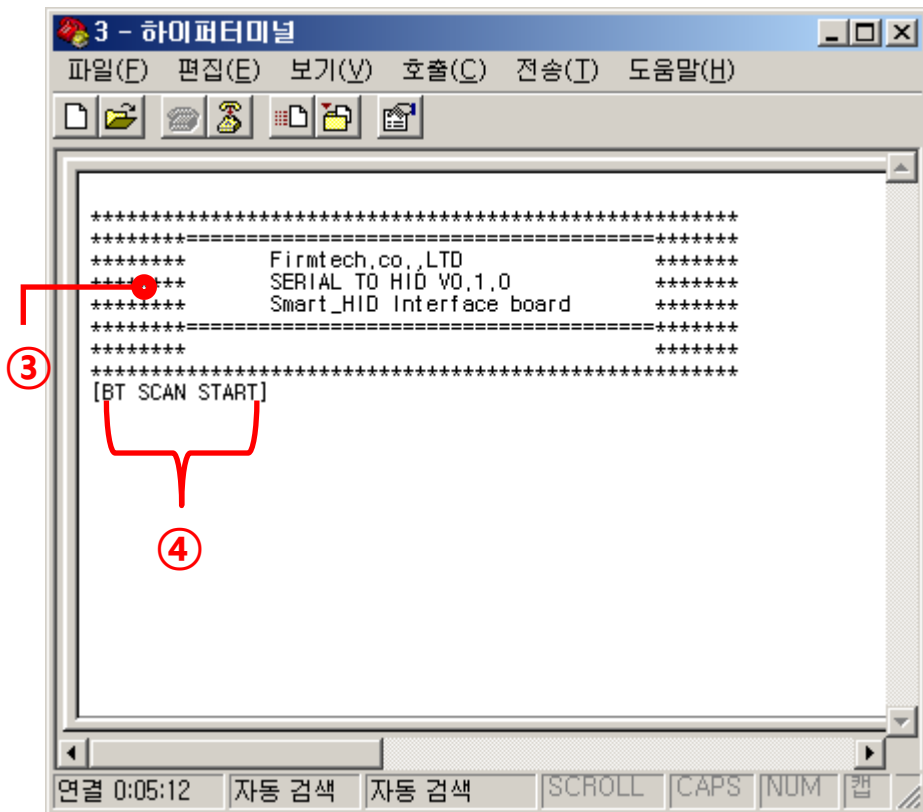
연결 진행

78. Smart_HID를 사용하기 위한 기본 사항

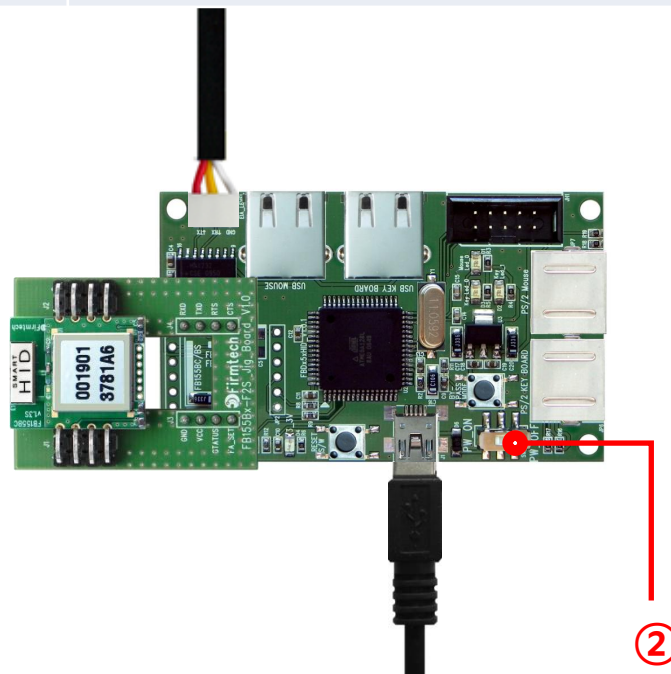
- Smart_HID Host는 **iPad**를 사용합니다.
- FB155BC_SMD(HID)는 HID Device Profile이 운영되고 있어야 합니다.
- Smart_HID Interface Board는 **SERIAL_to_HID** 프로그램이 운영되고 있어야 합니다.
- 하이퍼터미널을 사용하여 Serial Data를 입력합니다.

NO	Description
1	FB155BC_SMD(HID)를 Smart_HID Interface Board에 장착
2	Smart_HID Interface Board 전원 ON => FB155BC_SMD(HID) Scan 동작 확인
3	iPad 설정=>일반=>Bluetooth=>검색=>연결
4	iPad의 메모 실행
5	하이퍼터미널을 사용하여 Serial Data 입력
6	iPad의 메모에서 문자 입력 확인

79. Smart_HID Interface Board 전원 ON



NO	연결 및 체크사항
1	FB155BC_SMD(HID)는 iPad가 동작되기 전에 SCAN작업을 진행해야 합니다. 그러므로 Smart_HID Interface Board의 전원 ON을 미리 진행해야 합니다.
2	FB155BC_SMD(HID)가 장착된 Smart_HID Interface Board 전원 ON
3	하이퍼 터미널에 운영 프로그램 정보 출력 - Smart_HID Interface 프로그램 운영 확인 가능
4	하이퍼 터미널에 BT Module Scan 메시지 출력 (Smart_HID Interface Board의 Status LED는 깜빡거립니다.)



80. iPad “설정” 선택



iPad 화면상에서 “**설정**” 아이콘을 클릭합니다.

81. iPad “일반” 선택



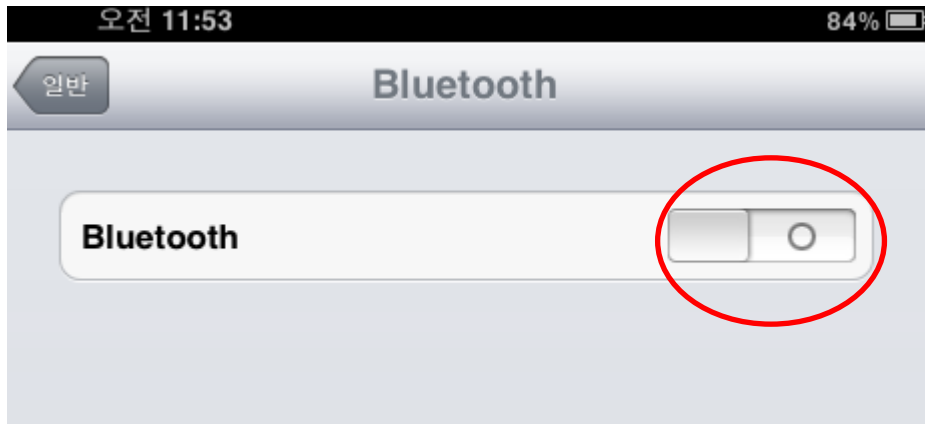
설정 메뉴 중 “**일반**” 아이콘을 클릭합니다.

82. iPad “Bluetooth” 선택



일반 메뉴 중 “**Bluetooth**” 아이콘을 클릭합니다.

83. iPad Bluetooth "ON"



Bluetooth 기능을 "ON" 시킵니다.

84. iPad Bluetooth 검색 진행



Bluetooth 기능이 ON되면 자동으로 주변의 Bluetooth 장치를 검색합니다.

검색된 Bluetooth 장치 리스트 중, "FB155BC_HID"가 있는지 확인합니다.

"FB155BC_HID"가 없으면, 다시 검색을 진행하거나 "FB155BC_HID"가 Scan 작업을 정확하게 수행 중인지 확인합니다.

85. FB155BC_HID를 선택하여 연결 진행

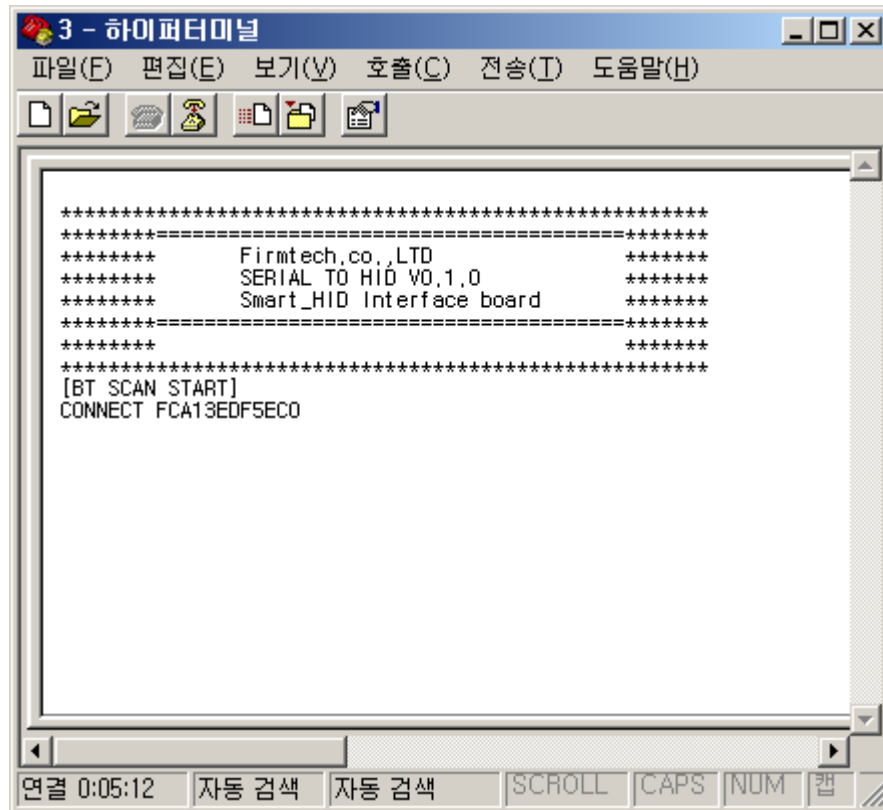


"FB155BC_HID"를 클릭하여 연결을 진행합니다.

86. FB155BC_HID와 연결 완료



87. Smart_HID Interface Board 연결 후 상태



"CONNECT xxxxxxxxxxxx"라는 메시지는 Bluetooth 장치가 연결된 것을 나타냅니다.

88. iPad의 메모 실행



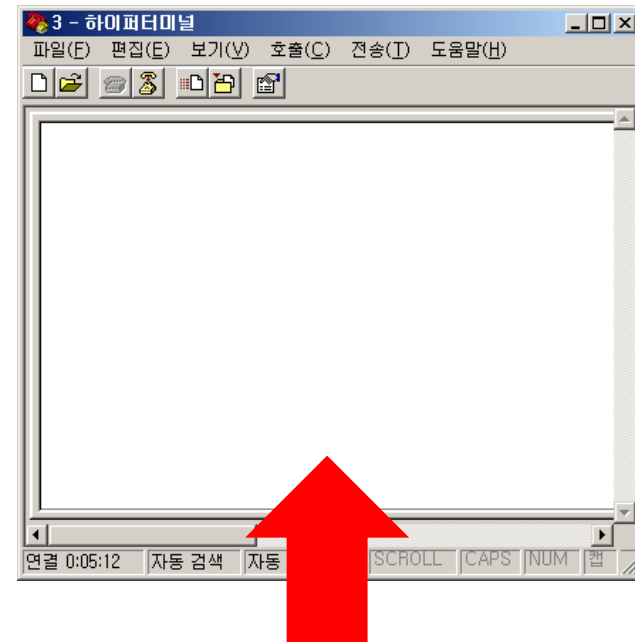
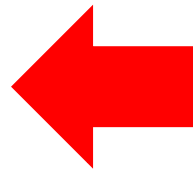
89. iPad의 메모 커서 활성화



메모를 실행시키고, 화면을 터치해서 커서가 활성화 된 상태를 만들어야 합니다.

iPad의 커서가 활성화 되어 있지 않으면 키보드의 입력이 준비되지 않은 상태입니다. 즉, 커서가 활성화 되지 않으면 키보드 입력이 진행되지 않습니다.

90. 무선 시리얼 데이터 출력된 화면



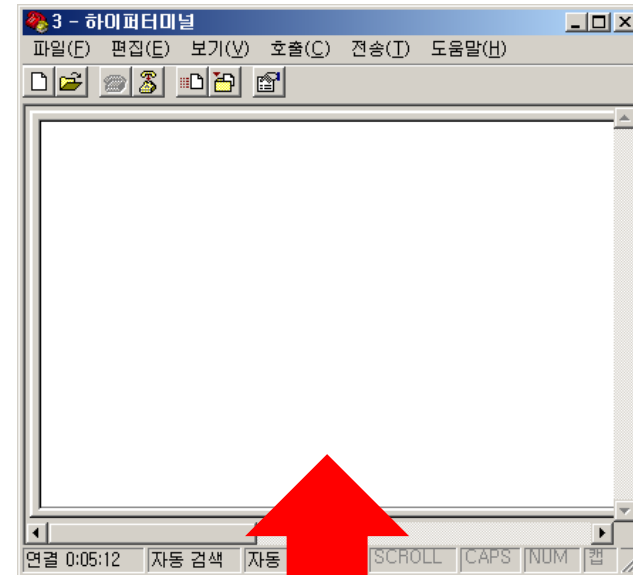
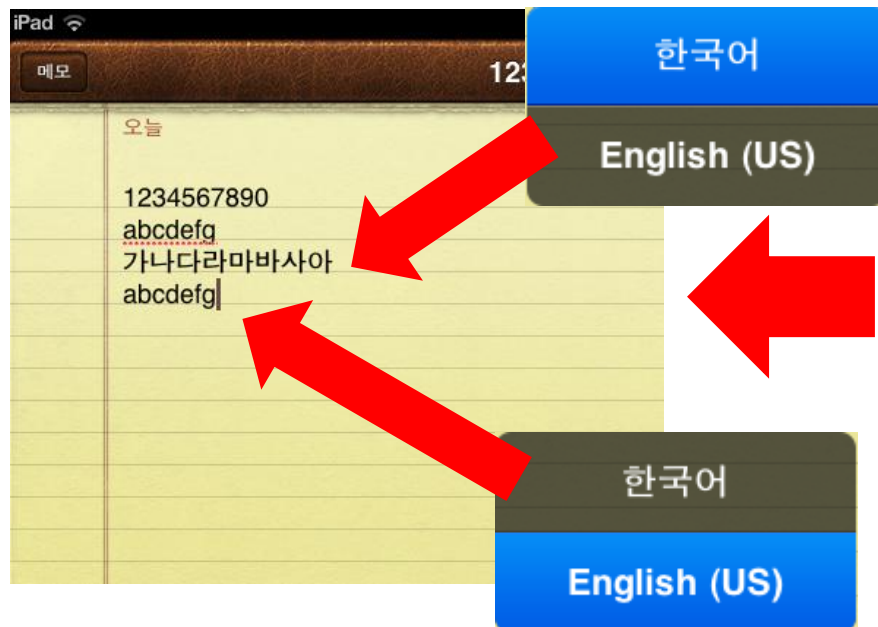
PC와 연결된 Keyboard를 사용하여 하이퍼터미널에 시리얼 데이터를 입력합니다.

"1234567890"을 입력하면 iPad에 문자가 출력됩니다.

"abcdefg"를 입력하면 iPad에 문자가 출력됩니다.

하이퍼터미널에 데이터를 입력할 때, PC의 입력방식은 반드시 영문으로 되어 있어야 합니다.

91. 한영 전환



한/영 변환을 위해서는, PC와 연결된 Keyboard를 사용하여 하이퍼터미널에 "0x02(Ctrl+b)"를 입력합니다. (1회 입력 시 변환 메뉴 출력, 2회 입력 시 변환됨)

"가나다라마바사아"를 입력하면 Galaxy-s에 문자가 출력됩니다.

하이퍼터미널에 데이터를 입력할 때, PC의 입력방식은 반드시 영문으로 되어 있어야 합니다.

SERIAL_to_HID

프로그램 소스 분석

Main Process : main.c

Utility Process : util.c

92. SERIAL_to_HID 프로그램 Main Process

(1) Main Process : `INIT_PORT()`

```
void INIT_PORT(void)
{
    DDRA = 0x00;
    PORTA = 0xff;
    //
    DDRB = 0xff;
    PORTB = 0xff;
    //
    DDRC = 0xff;
    PORTC = 0xff;
    //
    DDRD = 0xe0;
    PORTD = 0xff;
    //
    DDRE = 0x3f;
    PORTE = 0xf3;
    //
    DDRF = 0x00;
    PORTF = 0x00;
    //
    DDRG = 0xff;
    PORTG = 0xff;
} ? end INIT_PORT ?
```

< ATmega128 A Port >

1. A Port 1은 Smart_HID Interface 보드의 Tack Switch 연결
2. A Port 입력(0) 설정, 초기값 High(1) 설정

< ATmega128 D Port >

1. D Port 0/1은 Smart_HID Interface 보드의 PS2 Connector 1 연결
2. D Port 5/6/7은 Smart_HID Interface 보드의 LED 1/2/3 연결
3. D Port 입(0)/출(1)력 설정, 초기값 High(1) 설정

< ATmega128 E Port >

1. E Port 6/7은 Smart_HID Interface 보드의 PS2 Connector 2 연결
2. E Port 3은 BT Module의 전원 ON/OFF 제어용으로 사용
3. E Port 입(0)/출(1)력 설정, 초기값 Low(0)/High(1) 설정

(2) Main Process : `INIT_FLAG()`

```
void INIT_FLAG(void)
{
    user_step = USER_STEP_1;
    old_user_step = USER_STEP_0;
    //
    tact_switch_value = 0;
    //
    cr0_check_flag = 0;
    cr1_check_flag = 0;
    lf_check_flag = 0;
}
```

< Main Process Flag 초기화 >

1. 사용하는 전역 변수 초기화 진행

92. SERIAL_to_HID 프로그램 Main Process

(3) Main Process : **DISPLAY_ERROR()**

```
void DISPLAY_ERROR(void)
{
    DISPSTR_UART1((unsigned char *)"[ERROR, STEP:");
    PUTCHAR_UART1(HEX2CHAR(old_user_step));
    DISPSTR_UART1((unsigned char *)"]\r\n");
    //
    user_step = USER_STEP_30;
}
```

< Main Process ERROR 출력 함수 >

1. Main Process 진행 중, ERROR 발생 시 현재의 스텝과 ERROR를 알리는 함수

(4) Main Process : **READ_PORT(unsigned char port_number)**

```
unsigned char READ_PORT(unsigned char port_number)
{
    unsigned char return_avr_port_value;
    //
    //
    switch(port_number)
    {
        case READ_TACT_SWITCH:
            return_avr_port_value = PINA & 0x03;
            break;
        default:
            break;
    }
    return return_avr_port_value;
}
```

< 포트 읽기 함수 >

1. 읽고자 하는 포트 넘버(port_number)를 이용하여 ATmega128에 연결되어 있는 포트의 상태 값을 읽는 함수
2. Port_number에 따라 각각의 포트 상태 값을 읽어 반환한다.
3. A Port에 연결된 Tact Switch 상태 읽기 가능

92. SERIAL_to_HID 프로그램 Main Process

(5) Main Process : **WRITE_PORT(unsigned char port_number)**

```
void WRITE_PORT(unsigned char port_number)
{
    switch(port_number)
    {
        case BT_MODULE_OFF:
            cbi(PORTE,PE3);
            break;
        case BT_MODULE_ON:
            sbi(PORTE,PE3);
            break;
        default:
            break;
    }
}
```

< 포트 쓰기 함수 >

1. 변경하고자 하는 포트 넘버(port_number)를 이용하여 ATmega128에 연결되어 있는 포트의 상태 값을 출력하는 함수
2. Port_number에 따라 각각의 포트 상태 값을 출력한다.
3. cbi()함수 : 해당 포트의 비트 값을 Low로 출력
4. sbi()함수 : 해당 포트의 비트 값을 High로 출력
5. E Port 3에 연결되어 있는 BT Module 전원 제어 용 TR 제어 가능

(6) Main Process : **CHECK_NEXT_STEP()**

```
void CHECK_USER_NEXT_STEP(void)
{
    old_user_step = user_step;
    user_step = USER_STEP_0;
    //
    //타임아웃 시간으로 약 3초를 설정한다.
    SET_WAIT_TIME(999); //333*3 = 3s
}
```

< 스텝 저장 함수 >

1. 현재 진행 중인 스텝을 저장하고, 특정한 스텝 (USER_STEP_0)을 운영하기 위한 스텝 저장 함수

92. SERIAL_to_HID 프로그램 Main Process

(7) Main Process : RETURN_OLD_USER_STEP()

```
void RETURN_OLD_USER_STEP(void)
{
    user_step = old_user_step;
    user_step++;
}
```

< 스텝 복귀 함수 >

1. 저장했던 스텝을 복귀 시키고, 복귀된 현재 스텝을 증가시켜 다음 스텝을 진행하기 위한 스텝 복귀 함수

(8) Main Process : BYPASS_MODE()

```
void BYPASS_MODE(void)
{
    if(CHECK_RX_BUF_UART0())
    {
        PUTCHAR_UART1(GETCHAR_UART0());
    }
    //
    if(CHECK_RX_BUF_UART1())
    {
        PUTCHAR_UART0(GETCHAR_UART1());
    }
}
```

< 통과 함수 >

1. ATMeag128의 UART 0으로 입력된 데이터를 ATMeag128의 UART 1로 출력하는 함수
2. ATMeag128의 UART 1로 입력된 데이터를 ATMeag128의 UART0으로 출력하는 함수
3. BT Module과 사용자(PC)가 직접 통신하는 경우 사용

92. SERIAL_to_HID 프로그램 Main Process

(9) Main Process : **DISPLAY_INFORMATION()**

```
void DISPLAY_INFORMATION(void)
{
    DISPSTR_UART1((unsigned char *)"\\r\\n\\r\\n\\r\\n*****\\r\\n");
    DISPSTR_UART1((unsigned char *)"*****=====\\r\\n");
    DISPSTR_UART1((unsigned char *)"*****      Firmtech.co.,LTD      *****\\r\\n");
    //
    DISPSTR_UART1((unsigned char *)"*****      ");
    DISPSTR_UART1((unsigned char *)"SERIAL TO HID V0.1.0 ");
    DISPSTR_UART1((unsigned char *)"*****\\r\\n");
    //
    DISPSTR_UART1((unsigned char *)"*****      Smart_HID Interface board      *****\\r\\n");
    DISPSTR_UART1((unsigned char *)"*****=====\\r\\n");
    DISPSTR_UART1((unsigned char *)"*****\\r\\n");
    DISPSTR_UART1((unsigned char *)"*****\\r\\n");
}
```

< 프로그램 정보 출력 함수 >

1. 운영중인 프로그램 정보를 시리얼로 출력하는 함수

92. SERIAL_to_HID 프로그램 Main Process

(10) Main Process : CHECK_CR_LF_PROCESS()

```
void CHECK_CR_LF_PROCESS(void)
{
    unsigned int loop_count;
    unsigned char check_point;
    //
    //
    memset(uart0_parsing_buf,0x00,60);
    uart0_length = 0;
    //
    if((cr0_check_flag > 1)&&(lf_check_flag > 1))
    {
        check_point = 0;
        uart0_length = CHECK_RX_BUF_UART0();
        if(uart0_length>0)
        {
            //UART0에 입력된 데이터를 uart0_parsing_buf에 수집한다.
            for(loop_count = 0;loop_count < uart0_length;loop_count++)
            {
                uart0_parsing_buf[loop_count] = GETCHAR_UART0();
                if((uart0_parsing_buf[loop_count] == 0x0a)&&(uart0_parsing_buf[loop_count-1] == 0x0d))
                {
                    check_point++;
                    if(check_point > 1)
                    {
                        cr0_check_flag = cr0_check_flag - 2;
                        lf_check_flag = lf_check_flag - 2;
                        //
                        RETURN_OLD_USER_STEP();
                        break;
                    }
                }
            }
        }
    }
    // ? end if (cr0_check_flag>1)&&(... ?
} ? end CHECK_CR_LF_PROCESS ?
```

< CR-LF 체크 함수 >

1. Main Process의 각 스텝은 BT Module에서 출력되는 메시지를 기준으로 운영됨.
2. 각 스텝마다 BT Module에서 출력되는 메시지를 검사하기 위해 본 함수를 콜 한다.
3. BT Module에서 출력된 데이터에 CR(Carriage Return)과 LF(Line Feed)가 있는지 체크하는 함수
4. BT Module에서 출력되는 메시지는 CR과 LF에 의해 구분됨(예 - CRLF메시지CRLF)
5. BT Module에서 출력된 메시지 중 CR과 LF가 각각 1개 이상인 경우, 하나 이상의 메시지가 출력된 것으로 인지하고 BT Module에서 출력된 메시지를 uart0_parsing_buf에 저장한다.
6. 다음 체크를 위해 CR과 LF 체크 플래그 -2 감소
7. 본 함수를 콜 하기 이전의 스텝으로 복귀하기 위해 복귀 함수 호출

92. SERIAL_to_HID 프로그램 Main Process

(11) Main Process : **PARSING_RECEIVED_MESSAGE()**

```
void PARSING_RECEIVED_MESSAGE(void)
{
    if(((uart0_length > 23)&&(! memcmp("\r\nCONNECT",&uart0_parsing_buf[uart0_length-24],10))))
    {
        user_step++;
    }
}
```

< 수신 메시지 비교 함수 >

1. Bluetooth Module로부터 수신 받은 메시지가 "CONNECT"인지 비교한다.
2. Bluetooth Module이 연결되면 "CONNECT 00189A012345"라는 메시지를 출력한다.
3. "00189A012345"는 상대방 Bluetooth 장치의 어드레스를 나타낸다. Bluetooth 어드레스는 상대방 장치에 따라 다르게 출력된다.
4. 상대방 장치와 연결되면, user_step을 증가시켜 다음 스텝을 진행 가능하게 만든다.

92. SERIAL_to_HID 프로그램 Main Process

(12) Main Process : PRINT_HEXBYTE()

```
void PRINT_HEXBYTE(unsigned char i)
{
    unsigned char h, l;
    //
    //
    h = i & 0xF0; // High nibble
    h = h >> 4;
    h = h + '0';
    if (h > '9')
        h = h + 7;
    l = (i & 0x0F) + '0'; // Low nibble
    if (l > '9')
        l = l + 7;
    //
    PUTCHAR_UART1(h);
    PUTCHAR_UART1(l);
}
```

< HEX를 Char로 변경하는 함수 >

1. 1바이트의 HEX값을 2바이트의 Char로 변경하는 함수
2. 하이퍼터미널과 같은 시리얼 통신 프로그램에서 HEX값의 데이터가 표시되기 위해서는 문자(Char) 형태로 변경되어야 함
3. HEX값 0x01은 Char 타입의 0x30('0')과 0x31('1')로 변경됨
4. HEX값 0x4F는 Char 타입의 0x34('4')와 0x46('F')로 변경됨

92. SERIAL_to_HID 프로그램 Main Process

(13) Main Process : **SEND_HID_REPORT()** – 1th

```
void SEND_HID_REPORT(unsigned char key_value, unsigned char modifier_value)
{
    // for Chage Korean/ English of iPad
    if(modifier_value == 0x08)
    {
        PUTCHAR_UART0(0x0c);
        PUTCHAR_UART0(0x00);
        PUTCHAR_UART0(0xa1);
        PUTCHAR_UART0(0x01);
        //
        PUTCHAR_UART0(modifier_value);
        //
        PUTCHAR_UART0(0x00);
        //
        PUTCHAR_UART0(0x00); // key value
        //
        PUTCHAR_UART0(0x00);
        PUTCHAR_UART0(0x00);
        PUTCHAR_UART0(0x00);
        PUTCHAR_UART0(0x00);
        PUTCHAR_UART0(0x00);
        //
        WAIT_1MS(10);
    } // end if modifier_value==0x08 ?
}
```

< HID_Report 송신 함수 >

1. HID_Report값으로 변경된 시리얼 데이터를 HID_Report Format으로 Bluetooth Module에 송신하는 함수
2. HID_Report값은 대표적으로 modifier_value와 Key_value로 구분됨
3. iPad의 한/영 전환은 modifier_value를 0x08로 설정한 HID_Report Format을 먼저 송신해야 함

92. SERIAL_to_HID 프로그램 Main Process

(14) Main Process : **SEND_HID_REPORT()** – 2th

```
    PUTCHAR_UART0(0x0c);
    PUTCHAR_UART0(0x00);
    PUTCHAR_UART0(0xa1);
    PUTCHAR_UART0(0x01);
    //
    PUTCHAR_UART0(modifier_value);
    //
    PUTCHAR_UART0(0x00);
    //
    PUTCHAR_UART0(key_value);
    //
    PUTCHAR_UART0(0x00);
    PUTCHAR_UART0(0x00);
    PUTCHAR_UART0(0x00);
    PUTCHAR_UART0(0x00);
    PUTCHAR_UART0(0x00);
    //
    // Alt key do not transmit cancellation
    if(modifier_value != 0x40)
    {
        // key value cancellation
        WAIT_1MS(10);
        //
        PUTCHAR_UART0(0x0c);
        PUTCHAR_UART0(0x00);
        PUTCHAR_UART0(0xa1);
        PUTCHAR_UART0(0x01);
        //
        PUTCHAR_UART0(modifier_value);
        //
        PUTCHAR_UART0(0x00);
        //
        PUTCHAR_UART0(0x00); // key value
        //
        PUTCHAR_UART0(0x00);
        PUTCHAR_UART0(0x00);
        PUTCHAR_UART0(0x00);
        PUTCHAR_UART0(0x00);
        PUTCHAR_UART0(0x00);
    } // ? end if modifier_value != 0x40 ?
} // ? end SEND_HID_REPORT ?
```

< HID_Report 송신 함수 >

1. HID_Report값으로 변경된 시리얼 데이터를 HID_Report Format으로 Bluetooth Module에 송신하는 함수
2. HID_Report값은 대표적으로 modifier_value와 Key_value로 구분됨
3. 기본적으로, HID_Report Format을 송신하면, Key값의 해제 코드를 송신해야 함
4. 그러나, Alt키(0x40)를 이용한 한/영 전환(Note Book)은 Key값의 해제 코드를 송신하지 않음

92. SERIAL_to_HID 프로그램 Main Process

(15) Main Process : **USERS_OPERATION()** – 1th

```
void USERS_OPERATION(void)
{
    unsigned int loop_count;
    unsigned char check_point;
    unsigned char serial_data;
    unsigned char modifier;
    //
    //
    switch(user_step)
    {
        // BT Module에서 출력되는 메시지 수집 함수 진행.
        case USER_STEP_0:
            CHECK_CR_LF_PROCESS();
            //
            // 약 3초동안 시리얼 데이터를 받지 못하면 타임 아웃으로 판단한다.
            if(GET_WAIT_TIME()==0)
            {
                DISPLAY_ERROR();
            }
            break;
        case USER_STEP_1:
            // 타임아웃 시간으로 약 3초를 설정한다.
            SET_WAIT_TIME(999); // 333*3 = 3s
            //
            WRITE_PORT(BT_MODULE_ON);
            user_step++;
            break;
```

< 사용자 지정 동작 함수 >

1. **User_step_0**은 Bluetooth Module에서 출력되는 메시지를 수집한다.
2. **User_step_1**은 타임아웃 시간을 3초로 설정하고 다음 스텝으로 진행한다.

92. SERIAL_to_HID 프로그램 Main Process

(16) Main Process : **USERS_OPERATION()** – 2th

< 사용자 지정 동작 함수 >

1. User_step_3은 Bluetooth Module로부터 “BTWIN HID Slave mode start”메시지를 수신했는지 체크한다.
2. User_step_4는 Bluetooth Module로부터 “OK”메시지를 수신했는지 체크한다.
3. User_step_5는 Bluetooth Module에게 “at+btscan”명령어를 전달한다.
4. User_step_6은 Bluetooth Module로부터 “OK”메시지를 수신했는지 체크한다.
5. Bluetooth Module에 “at+btscan”명령 전달 후 “OK”메시지를 수신하면, 다른 Bluetooth 장치에서 검색이 가능한 상태가 되는 것이다.

```
case USER_STEP_2:
    CHECK_USER_NEXT_STEP();
    break;
case USER_STEP_3:
    if((uart0_length > 29)&&(! memcmp("\r\nBTWIN HID Slave mode start\r\n",&uart0_parsing_buf[uart0_length-30],30)))
        CHECK_USER_NEXT_STEP();
    else
        DISPLAY_ERROR();
    break;
case USER_STEP_4:
    if(((uart0_length > 5)&&(! memcmp("\r\nOK\r\n",&uart0_parsing_buf[uart0_length-6],6))))
        user_step++;
    else
        DISPLAY_ERROR();
    break;
case USER_STEP_5:
    DISPSTR_UART0((unsigned char *)"at+btscan\r");
    CHECK_USER_NEXT_STEP();
    break;
case USER_STEP_6:
    if(((uart0_length > 5)&&(! memcmp("\r\nOK\r\n",&uart0_parsing_buf[uart0_length-6],6))))
    {
        DISPSTR_UART1((unsigned char *)"[BT SCAN START]");
        //
        user_step++;
    }
    else
        DISPLAY_ERROR();
    break;
```


92. SERIAL_to_HID 프로그램 Main Process

(17) Main Process : **USERS_OPERATION()** – 3th

```
case USER_STEP_7:
    if((cr0_check_flag > 1)&&(lf_check_flag > 1))
    {
        memset(uart0_parsing_buf,0x00,60);
        uart0_length = 0;
        //
        check_point = 0;
        uart0_length = CHECK_RX_BUF_UART0();
        if(uart0_length>0)
        {
            //UART0에 입력된 데이터를 uart0_parsing_buf에 수집한다.
            for(loop_count = 0;loop_count < uart0_length;loop_count++)
            {
                uart0_parsing_buf[loop_count] = GETCHAR_UART0();
                PUTCHAR_UART1(uart0_parsing_buf[loop_count]);
                if((uart0_parsing_buf[loop_count] == 0x0a)&&(uart0_parsing_buf[loop_count-1] == 0x0d))
                    check_point++;
                if(check_point > 1)
                {
                    cr0_check_flag = cr0_check_flag - 2;
                    lf_check_flag = lf_check_flag - 2;
                    //
                    PARSING_RECEIVED_MESSAGE();
                    break;
                }
            }
        }
    }
    } ? end if (cr0_check_flag>1)&&(... ?
    //
    tact_switch_value = READ_PORT(READ_TACT_SWITCH);
    //
    //if SW1 Press, Keyboard Mode
    if(tact_switch_value == 0x01)
    {
        DISPSTR_UART1((unsigned char *)"\\r\\n[KEYBOARD MODE]\\r\\n");
        WRITE_PORT(BT_MODULE_OFF);
        user_step = USER_STEP_9;
    }
    //
    if(CHECK_RX_BUF_UART1())
    {
        PUTCHAR_UART0(GETCHAR_UART1());
    }
    break;
```

< 사용자 지정 동작 함수 >

1. User_step_7은 Bluetooth Module로부터 수신된 메시지가 있는지 체크한다.
2. SCAN 동작 이후, 다른 Bluetooth 장치와 연결이 되면 "CONNECT ..."라는 메시지를 출력한다.
3. PARSING_RECEIVED_MESSAGE()함수에서 연결 시 출력되는 메시지를 체크한다.
4. Bypass Switch의 상태를 체크하여, Bluetooth 장치가 다른 장치와 연결 전에 Bypass Switch가 눌리면 Bluetooth 장치의 전원을 OFF 하고 User_step_9로 변경한다.
5. User_step_9는 입력된 시리얼 데이터를 HID_Report값으로 변경하여 시리얼로 출력하는 기능을 수행한다.
6. 사용자(PC)가 입력한 시리얼 데이터를 Bluetooth Module에 송신한다. 이 부분은 Bluetooth Module에 Pin Code를 입력해야 하는 경우 사용한다.

92. SERIAL_to_HID 프로그램 Main Process

(18) Main Process : USERS_OPERATION() – 4th

```
case USER_STEP_8:
{
    serial_data = GETCHAR_UART1();
    for(loop_count = 0; serial_hid_table[loop_count][0] != serial_data && serial_hid_table[loop_count][0]; loop_count++);
    if((serial_hid_table[loop_count][0] == serial_data)&&(serial_data != 0))
    {
        //Shift가 눌린 상태의 key value
        if(((serial_data > 0x20) && (serial_data < 0x27))
        || ((serial_data > 0x27) && (serial_data < 0x2C))
        || (serial_data == 0x3A)
        || (serial_data == 0x3C)
        || ((serial_data > 0x3D) && (serial_data < 0x5B))
        || ((serial_data > 0x5D) && (serial_data < 0x60))
        || ((serial_data > 0x7A) && (serial_data < 0x7F)))
        {
            modifier = 0x02; //Left Shift ON
        }
        //Ctrl+r가 눌린 상태의 key value: Alt Key 대체. Note Book의 한/영 전환에 사용.
        else if(serial_data == 0x12)
        {
            modifier = 0x40; //Right Alt ON
        }
        //Ctrl+b가 눌린 상태의 key value: iPad의 한/영 전환에 사용.
        else if(serial_data == 0x02)
        {
            modifier = 0x08; //Left GUI ON
        }
        else
            modifier = 0x00;
        //
        //Ctrl+a는 Galaxy or PC(or Note Book)의 한/영 전환에 사용.
        //
        SEND_HID_REPORT(serial_hid_table[loop_count][1], modifier);
    } ? end if (serial_hid_table[loo... ?
    else
    {
        DISPSTR_UART1((unsigned char *)"[NOT MATCHED]\r\n");
    }
} ? end if CHECK_RX_BUF_UART1() ?
break;
```

< 사용자 지정 동작 함수 >

1. User_step_8은 시리얼로 입력된 데이터를 HID_Report 값으로 변경하여 Bluetooth Module에 전달한다.
2. 시리얼로 입력된 데이터가 serial_hid_table 있는지 비교하고, 같은 값이 있으면 시리얼 데이터를 HID_Report 값으로 변환한다.
3. Shift키가 눌린 상태의 시리얼 데이터가 입력된 경우, HID_Report Modifier 값을 변경한다.
4. Ctrl키가 눌린 상태의 시리얼 데이터가 입력된 경우, HID_Report Modifier 값을 변경한다.
5. SEND_HID_REPORT() 함수를 이용하여 HID_Report로 변경된 값을 Bluetooth Module에 입력한다.

92. SERIAL_to_HID 프로그램 Main Process

(19) Main Process : USERS_OPERATION() – 5th

```
case USER_STEP_9:
    if(CHECK_RX_BUF_UART1())
    {
        serial_data = GETCHAR_UART1();
        DISPSTR_UART1((unsigned char *)" [ASCII:0x");
        PRINT_HEXBYTE(serial_data);
        PUTCHAR_UART1(' ');
        //
        for(loop_count = 0; serial_hid_table[loop_count][0] != serial_data && serial_hid_table[loop_count][0]; loop_count++);
        if((serial_hid_table[loop_count][0] == serial_data)&&(serial_data != 0))
        {
            DISPSTR_UART1((unsigned char *)" [HID:0x");
            PRINT_HEXBYTE(serial_hid_table[loop_count][1]);
            PUTCHAR_UART1(' ');
            DISPSTR_UART1((unsigned char *)"\r\n");
        }
        else
        {
            DISPSTR_UART1((unsigned char *)" [FAULT ]\r\n");
        }
    }
    break;
default:
    BYPASS_MODE();
    //
    if(timer0_counter < 100)
    {
        cbi(PORTD,PD5);
        cbi(PORTD,PD6);
        cbi(PORTD,PD7);
    }
    else
    {
        sbi(PORTD,PD5);
        sbi(PORTD,PD6);
        sbi(PORTD,PD7);
    }
    break;
} ? end switch user_step ?
} ? end USERS_OPERATION ?
```

< 사용자 지정 동작 함수 >

1. User_step_9는 시리얼로 입력된 데이터를 ASCII Code 값과 HID_Report 값으로 변경하여 시리얼로 출력한다.
2. Bypass Mode는 Bluetooth Module에서 출력되는 메시지를 사용자(PC)에게 전달하고, 사용자(PC)가 입력하는 데이터를 Bluetooth Module에 전달하는 기능을 수행한다.

92. SERIAL_to_HID 프로그램 Main Process

(20) Main Process : `main()`

```
int main(void)
{
    INIT_PORT();
    INIT_UART1(BAUD_115200);
    INIT_UART0(BAUD_115200);
    INIT_TIMER0();
    sei();
    //
    //BT Module 전원 OFF
    WRITE_PORT(BT_MODULE_OFF);
    //
    //프로그램 내에서 사용하는 Flag 초기화.
    INIT_FLAG();
    //
    //운영되는 프로그램 정보 출력.
    DISPLAY_INFORMATION();
    //
    tact_switch_value = READ_PORT(READ_TACT_SWITCH);
    //
    //if SW1 Press, Bypass Mode
    if(tact_switch_value == 0x01)
    {
        DISPSTR_UART1((unsigned char *)"[BYPASS MODE]\r\n");
        //
        //BT Module 전원 ON, BT Module Start
        WRITE_PORT(BT_MODULE_ON);
        //
        user_step = USER_STEP_30;
    }
    //
    while(1)
    {
        USERS_OPERATION();
    }
}
} ? end main ?
```

< 메인 함수 >

1. ATmega128 포트 초기화
2. ATmega128 UART0/1 초기화
3. ATmega128 Timer 0 초기화
4. 인터럽트 Enable 설정
5. BT Module OFF 설정
6. Main Process에서 사용하는 플래그 초기화 함수 콜
7. 운영 프로그램 정보 출력
8. Tact Switch 상태 체크: Tact Switch(Bypass Switch)를 누른 상태에서 동작을 시키면, Bypass Mode로 동작된다.
9. 사용자 지정 동작 함수: USER_OPERATION()를 콜 한다.

93. SERIAL_to_HID 프로그램 Utility Process

(1) Utility Process : UART 0 Function – 1th

```
SIGNAL(SIG_UART0_RECV)
{
    rx0_buf[p_rx0_wr++] = UDR0;
    if(rx0_buf[p_rx0_wr-1] == CR_VALUE)cr0_check_flag++;
    if(cr0_check_flag > 0)
    {
        if(rx0_buf[p_rx0_wr-1] == LF_VALUE)lf_check_flag++;
    }
    if (p_rx0_wr > UART0_BUF_SIZE-1)p_rx0_wr = 0;
}
//
void INIT_UART0(unsigned char baudrate)
{
    unsigned int i;

    UBRROH = 0;
    if(baudrate == BAUD_9600){UBRR0L = 71;}/* 9600 BAUD at 11.0592MHz*/
    else if(baudrate == BAUD_19200){UBRR0L = 35;}/* 19200 BAUD at 11.0592MHz*/
    else if(baudrate == BAUD_38400){UBRR0L = 17;}/* 38400 BAUD at 11.0592MHz*/
    else if(baudrate == BAUD_57600){UBRR0L = 11;}/* 57600 BAUD at 11.0592MHz*/
    else if(baudrate == BAUD_115200){UBRR0L = 5;}/* 115200 BAUD at 11.0592MHz*/
    else {UBRR0L = 71;}/* default 9600 BAUD at 11.0592MHz*/
    UCSR0B = (1<<RXCIE)|(1<<RXEN)|(1<<TXEN);
    p_rx0_wr = 0;
    p_rx0_rd = 0;
    for (i=0; i<UART0_BUF_SIZE; i++) rx0_buf[i] = 0;
}
//
unsigned char PUTCHAR_UART0 (unsigned char c)
{
    while (!(UCSR0A & 0x20)) ;
    UDR0 = c;
    return 0;
}
```

< ATmega128 UART0 interrupt vector >

1. UART0으로 데이터가 입력된 경우 수행되는 부분
2. UART0으로 입력된 데이터는 UDR0 에 저장되어 있음
3. UDR0 에 저장되어 있는 1바이트의 데이터를 rx0_buf [] 버퍼에 저장
4. 수신 데이터가 CR(0x0d)인 경우 플래그 체크
5. 수신 데이터가 LF(0x0a)인 경우 플래그 체크

< ATmega128 UART0초기화 >

1. UBR0L 설정에 따른 시리얼 데이터 통신 속도 설정
2. 수신 완료 인터럽트 허용 (RXCIE)
3. UART0 수신 부 동작 허용 (RXEN)
4. UART0 송신 부 동작 허용 (TXEN)
5. 수신 데이터 저장 버퍼 및 수신 데이터 포인터 초기값 설정

< ATmega128 UART0 출력 함수 >

1. UART0으로 한 바이트의 시리얼 데이터를 출력할 때 사용하는 함수

93. SERIAL_to_HID 프로그램 Utility Process

(2) Utility Process : UART 0 Function – 2th

```
void DISPSTR_UART0 (unsigned char *s)
{
    while (*s != '\0')
        PUTCHAR_UART0(*s++);
}
//
unsigned int CHECK_RX_BUF_UART0(void)
{
    unsigned int len;

    if (p_rx0_wr == p_rx0_rd)
    {
        len = 0;
    }
    else
    {
        if (p_rx0_wr > p_rx0_rd) len = p_rx0_wr - p_rx0_rd;
        else len = UART0_BUF_SIZE + p_rx0_wr - p_rx0_rd;
    }
    return len;
}
//
unsigned char GETCHAR_UART0 (void)
{
    unsigned char ch;

    ch = rx0_buf[p_rx0_rd];
    p_rx0_rd++;
    if (p_rx0_rd > UART0_BUF_SIZE-1) p_rx0_rd = 0;
    return ch;
}
//
void DISPLAY_CR_LF_UART0()
{
    PUTCHAR_UART0(0x0a);
    PUTCHAR_UART0(0x0d);
}
```

< DISPSTR_UART0() >

1. 문자열로 된 시리얼 데이터를 UART0 포트로 출력

< CHECK_RX_BUF_UART0() >

1. UART0로 입력된 데이터는 rx0_buf[] 버퍼에 저장
2. CHECK_RX_BUF_UART0() 함수는 rx0_buf[] 버퍼에 저장되어 있는 데이터의 개수를 체크하는 함수

< GETCHAR_UART0() >

1. rx0_buf[]에 저장되어 있는 데이터를 1바이트씩 꺼내오는 함수

< DISPLAY_CR_LF_UART0() >

1. UART0으로 CR(0Xd)와 LF(0x0a)를 출력하는 함수

93. SERIAL_to_HID 프로그램 Utility Process

(3) Utility Process : UART 1 Function – 1th

```
SIGNAL(SIG_UART1_RECV)
{
    rx1_buf[p_rx1_wr++] = UDR1;
    if(rx1_buf[p_rx1_wr-1] == CR_VALUE) cr1_check_flag++;
    if (p_rx1_wr > UART1_BUF_SIZE-1) p_rx1_wr = 0;
}
//
void INIT_UART1(unsigned char baudrate)
{
    unsigned int i;

    UBRR1H = 0;
    if(baudrate == BAUD_9600){UBRR1L = 71;} /* 9600 BAUD at 11.0592MHz*/
    else if(baudrate == BAUD_19200){UBRR1L = 35;} /* 19200 BAUD at 11.0592MHz*/
    else if(baudrate == BAUD_38400){UBRR1L = 17;} /* 38400 BAUD at 11.0592MHz*/
    else if(baudrate == BAUD_57600){UBRR1L = 11;} /* 57600 BAUD at 11.0592MHz*/
    else if(baudrate == BAUD_115200){UBRR1L = 5;} /* 115200 BAUD at 11.0592MHz*/
    else {UBRR1L = 71;} /* default 9600 BAUD at 11.0592MHz*/
    UCSR1B = (1<<RXCIE)|(1<<RXEN)|(1<<TXEN);
    p_rx1_wr = 0;
    p_rx1_rd = 0;
    for (i=0; i<UART1_BUF_SIZE; i++) rx1_buf[i] = 0;
}
//
unsigned char PUTCHAR_UART1 (unsigned char c)
{
    while (!(UCSR1A & 0x20)) ;
    UDR1 = c;
    return 0;
}
```

< ATmega128 UART1 interrupt vector >

1. UART1로 데이터가 입력된 경우 수행되는 부분
2. UART1로 입력된 데이터는 UDR1 에 저장되어 있음
3. UDR1 에 저장되어 있는 1바이트의 데이터를 rx1_buf [] 버퍼에 저장
4. 수신 데이터가 CR(0x0d)인 경우 플래그 체크

< ATmega128 UART1초기화 >

1. UBRR1L 설정에 따른 시리얼 데이터 통신 속도 설정
2. 수신 완료 인터럽트 허용 (RXCIE)
3. UART1 수신 부 동작 허용 (RXEN)
4. UART1 송신 부 동작 허용 (TXEN)
5. 수신 데이터 저장 버퍼 및 수신 데이터 포인터 초기값 설정

< ATmega128 UART1 출력 함수 >

1. UART1로 한 바이트의 시리얼 데이터를 출력할 때 사용하는 함수

93. SERIAL_to_HID 프로그램 Utility Process

(4) Utility Process : UART 1 Function – 2th

```
void DISPSTR_UART1 (unsigned char *s)
{
    while (*s != "\0")
        PUTCHAR_UART1(*s++);
}
//
unsigned int CHECK_RX_BUF_UART1(void)
{
    unsigned int len;

    if (p_rx1_wr == p_rx1_rd)
    {
        len = 0;
    }
    else
    {
        if (p_rx1_wr > p_rx1_rd) len = p_rx1_wr - p_rx1_rd;
        else len = UART1_BUF_SIZE + p_rx1_wr - p_rx1_rd;
    }
    return len;
}
//
unsigned char GETCHAR_UART1 (void)
{
    unsigned char ch;

    ch = rx1_buf[p_rx1_rd];
    p_rx1_rd++;
    if (p_rx1_rd > UART1_BUF_SIZE-1) p_rx1_rd = 0;
    return ch;
}
//
void DISPLAY_CR_LF_UART1()
{
    PUTCHAR_UART1(0x0a);
    PUTCHAR_UART1(0x0d);
}
```

< DISPSTR_UART1() >

1. 문자열로 된 시리얼 데이터를 UART1 포트에 출력

< CHECK_RX_BUF_UART1() >

1. UART1로 입력된 데이터는 rx1_buf[] 버퍼에 저장
2. CHECK_RX_BUF_UART1() 함수는 rx1_buf[] 버퍼에 저장되어 있는 데이터의 개수를 체크하는 함수

< GETCHAR_UART1() >

1. rx1_buf[]에 저장되어 있는 데이터를 1바이트씩 꺼내는 함수

< DISPLAY_CR_LF_UART1() >

1. UART1으로 CR(0x0d)와 LF(0x0a)를 출력하는 함수

93. SERIAL_to_HID 프로그램 Utility Process

(5) Utility Process : Timer 0 Function

```
SIGNAL(SIG_OVERFLOW0)
{
    timer0_counter++;
    if (timer0_counter > 200 )
    {
        timer0_counter = 0;
    }
    //
    if(check_wait_time)check_wait_time--;
}
//
void INIT_TIMER0(void)
{
    TIMSK |= 1 << TOIE0;
    TCNT0 = 0;
    TCCR0 = 5;
    timer0_counter = 0;
    //
    check_wait_time = 0;
}
//
void SET_WAIT_TIME(unsigned int time)
{
    cli();
    check_wait_time = time;
    sei();
}
//
unsigned int GET_WAIT_TIME()
{
    unsigned int t;
    //
    cli();
    t = check_wait_time;
    sei();
    //
    return t;
}
```

< ATmega128 Timer0 interrupt vector >

1. Timer0으로 설정된 시간이 되면 수행되는 부분
2. 설정된 시간마다 timer0_counter를 1씩 증가
3. timer0_counter가 200보다 커지면 timer0_counter를 0으로 설정

< ATmega128 Timer0 초기화 >

1. Timer0의 인터럽트 허용 설정 (TIMSK |=1<<TOIE0)
2. Timer0의 카운터는 0부터 시작 (TCNT0 = 0)
3. Timer0의 분주 비 128 설정 (TCCR0 = 5 = 0x05 = 0000 0101)
4. Application에서 사용하기 위한 timer0_counter = 0 설정

< 사용자 지정 대기 함수 >

1. Timer0를 시용하여 일정 시간을 체크하는 함수
2. 일정 시간 설정을 위해 SET_WAIT_TIME()함수를 사용한다.
3. 설정된 일정 시간의 상태를 알기 위해 GET_WAIT_TIME()함수를 사용한다.

93. SERIAL_to_HID 프로그램 Utility Process

(6) Utility Process : Other Function

```
unsigned char HEX2CHAR(unsigned char c)
{
    if ((c == 0 || c > 0) && (c < 10))
        return '0' + c;
    if (c >= 10 && c <= 15)
        return 'A' + c - 10;
    //
    return c;
}
```

< HEX2CHAR() >

1. 1 바이트의 HEX 값을 1 바이트의 Char로 변경
2. 하이퍼 터미널과 같은 시리얼 통신 프로그램에서 정상적으로 데이터가 표시 되기 위해서는 HEX 값을 Char 형태로 변경해야 함
3. HEX값 0x1은 Char 타입의 1(HEX값으로 0x31)로 변경됨

```
void WAIT_1MS(unsigned int cnt)
{
    unsigned int i;
    for (i = 0; i < cnt; i++) WAIT_1US(1000);
}
//
void WAIT_1US(unsigned int cnt)
{
    unsigned int i;
    for (i = 0; i < cnt; i++) ;
}
```

< WAIT_1MS() >

1. 약 1ms동안 대기하는 함수

< WAIT_1US() >

1. 약 1us동안 대기하는 함수

94. SERIAL_to_HID 프로그램 컴파일 : WINAVR 사용

Tool에 대한 설명은 펌테크 홈페이지에서
"FB800ED_Tools Guide.PDF"파일을 참고 하십시오.

```
C:\WINDOWS\system32\cmd.exe

C:\WFIRTECH\SERIAL_to_HID_U0.1.0>make
----- begin -----
avr-gcc --version
avr-gcc (GCC) 4.2.2 (WinAVR 20071221)
Copyright (C) 2007 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

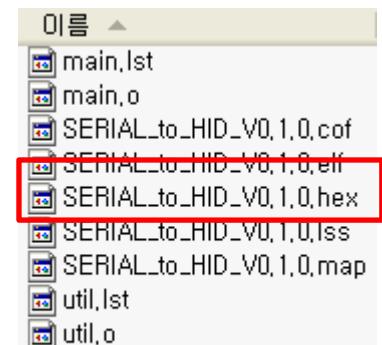
#cc-avr-objcopy -j .eeprom --set-section-flags=.eeprom="alloc,load" --change-sect
ion-lma .eeprom=0 -O ihex SERIAL_to_HID_U0.1.0.elf SERIAL_to_HID_U0.1.0.eep
avr-objcopy --debugging --change-section-address .data=0x8000000 --change-section
-address .bss=0x8000000 --change-section-address .noinit=0x8000000 --change-sectio
n-address .eeprom=0x8100000 -O coff-avr SERIAL_to_HID_U0.1.0.elf SERIAL_to_HID_U0
.1.0.cof
Warning: file C:/WINDOWS/TEMP/ccNQGKrd.s not found in symbol table, ignoring
Warning: ignoring function __vectors() outside any compilation unit
Warning: ignoring function __bad_interrupt() outside any compilation unit
avr-objcopy: --change-section-vma .eeprom+0xff7f0000 never used
avr-objcopy: --change-section-lma .eeprom+0xff7f0000 never used
avr-objcopy: --change-section-vma .noinit+0xff800000 never used
avr-objcopy: --change-section-lma .noinit+0xff800000 never used
cp SERIAL_to_HID_U0.1.0.cof RESULT/SERIAL_to_HID_U0.1.0.cof
Size after:
   text    data    bss    dec    hex filename
     0    5820     0    5820    16bc SERIAL_to_HID_U0.1.0.hex
----- end -----

C:\WFIRTECH\SERIAL_to_HID_U0.1.0>
```

< 컴파일 방법 >

- Window Command 창을 이용 하여 컴파일에 사용할 파일이 있는 위치로 이동
- make clean입력 후 Enter Key 입력하여 기존의 파일 삭제
- make 입력 후 Enter Key 입력
- ERROR 없이 진행된 경우, HEX 파일과 기타 파일 생성

< make 실행 이후 RESULT 폴더 >



Memo