<UART>

**UART**(범용 비동기화 송수신기: Universal asynchronous receiver/transmitter)는 병렬 데이터의 형태를 직렬 방식으로 전환하여 데이터를 전송하는 컴퓨터 하드웨어의 일종입니다.

이때 U, Universal은 범용을 가리키는데 이는 자료 형태나 전송 속도를 직접 구성할 수 있고 실제 전기 신호 수준과 방식(이를테면 차분 신호)이 일반적으로 UART 바깥의 특정한 드라이버 회로를 통해 관리를 받습니다. 이름에서 알 수 있듯이 비동기 통신이므로 수신 쪽에서 동기신호를 찾아내어 데이터의 시작과 끝을 시간적으로 알아 처리할 수 있도록 약속되어 있습니다.

클럭 신호를 통해 그 비트의 논리 상태를 결정하여 데이터 통신을 합니다.직렬(Serial) 통신은 1개의 입출력 핀을 통해 8개 비트를 한 번에 전송하는 방법이며 이를 사용하기 위해서는 보내는 쪽(TX)과 받는 쪽(RX)에서 약속을 정해야 하는데, 이를 프로토콜(Protocol)이라고 합니다.

또한 0은 GND, 1은 VCC로 데이터를 전송하고, 받는 쪽에서는 GND와 VCC를 다시 0과 1의 이진 값으로 변환하여 사용합니다.

또한 보내는 쪽과 받는 쪽이 원할 하게 데이터를 교류하기 위해 데이터를 보내는 속도에 대한 약속, 프로토콜을 정해 통신합니다. 그리고 이와 관련되어 UART에서는 보내는 쪽(TX)와 받는 쪽(RX)에서 데이터를 보내는 속도를 보율(baud rate)이라고 하는데이때 TX와 RX가 동일한 속도로 데이터 통신한다 하더라도 정확하게 통신이 이루어지는 것은 아니며보내는 쪽(TX)이 항상 데이터를 보내는 것 또한 아닙니다. 필요한 경우에만 데이터를 보내고, 받는 쪽(RX)은 언제 보내는 쪽(TX)이 데이터를 보내는지, 그리고 어디서부터가 보내는 쪽(TX)에서 보낸 데이터의 시작인지를 '0'의 시작 비트(start bit)와 '1'의 정지 비트(stop bit)를 사용하여서 바이트 단위 통신(주로)으로 10비트 데이터를 전송합니다.

이때 UART 통신은 전이중 방식(full duplex)통신으로 송신과 수신을 동시에 진행할 수 있으며, 이를 위해서 2개의 범용 입출력 핀이 존재하는데 (시리얼포트 -> TX, RX) 그리고 이때 송신과 수신단의 신호레벨을 고려해야 하는데 신호 레벨은 UART의 신호 레벨(TTL)과 달라 별도의 변환 장치를 사용하여 레벨을 변환합니다.

****

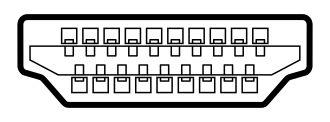
<HDMI>

HDMI(영어: High-Definition Multimedia Interface)는 비 압축 방식의 디지털 비디오/오디오 인터페이스 규격의 하나입니다. HDMI는 HDMI를 지원하는 셋탑박스 DVD 재생기 등의 멀티미디어 소스에서 AV기기, 모니터, 디지털 텔레비전 등의 장치들 사이의 인터페이스를 제공합니다.

HDMI는 컴퓨터와 디스플레이의 인터페이스 표준 규격인 DVI를 AV 전자제품용으로 변경한 것으로, 영상·음성을 압축하지 않고 플레이어에서 텔레비전 쪽으로 전송하기 때문에 디코더칩이나 소프트웨어가 별도로 필요 없다는 특징이 있습니다. 또한 그렇기 때문에 접속 기기끼리 호환성을 가집니다. 또, 영상·음성·제어 신호가 케이블 하나로 전송되기 때문에 번거로웠던 AV기기의 배선을 간단하게 할 수 있습니다. 또, 제어 신호도 보낼 수 있으므로 각 AV기기를 유기적으로 사용할 수 있습니다.

물리층은 변화 최소화 차분 신호 (TMDS), 신호의 암호화는 HDCP(High-bandwidth Digital Content Protection), 기기 간 인증은 EDID, 계 전체의 제어 계 접속은 CEC(Consumer Electronics Control)가 사용됩니다. 연결 커넥터에는 타입 A와 타입 B, 이렇게 2가지가 있으며, 표준은 타입 A입니다. 타입 B는 29 핀의 커넥터로 1080p(1920 x 1080)를 넘는 해상도를 지원하기 위해서 정의되었습니다. 소형 단자를 위한 미니 HDMI도 정의되어 있으며, 음성 신호에 대한 규격은 S/PDIF이다.

HDMI의 규격은 전기적 신호 규격, 핀 출력, 케이블과 커넥터 기구 규격에 대한 규정을 정의합니다.



(오른쪽이 1번 핀, 왼쪽이 19번 핀)

HDMI 규격서에는 3가지 타입의 커넥터를 정의한다. 각 타입은 서로 다른 시장에서 사용됩니다. 타입 A는 19핀이고, SDTV, EDTV, HDTV의 대역 폭을 지원합니다. 타입 B는 29핀이고, HDMI 1.0 버전에서 정의된 고해상도 규격입니다. 타입 C는 미니 HDMI라고 하며 소형 커넥터입니다. 타입 D는 마이크로 HDMI라고 하며 극소형 커넥터입니다. 타입 E는 자동차 연결 시스템용 입니다.

HDMI 케이블은 영상, 음성, CEC(Consumer Electronics Control) 신호(기계 제어 신호)를 실어 나릅니다. 앞서 말한 타입 A부터 E까지 모두 가능하며. HDMI 케이블의 규격은 데이터 처리량과 관계가 있으며, 모든 케이블들은 하위 호환된다. HDMI 케이블은 고급 사양이므로 일반적인 다른 케이블들 보다 고가였으나, 최근에는 동축 케이블, RCA 케이블 가격과 비슷한 수준으로 판매되고 있습니다.

<Wireless LAN>

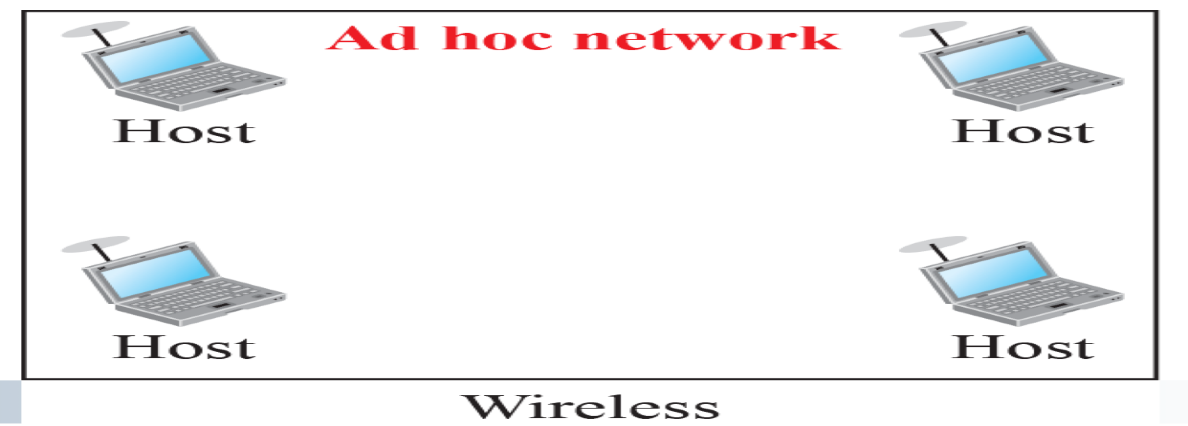
무선랜(wireless LAN)은 무선 신호 전달 방식(일반적으로 확산 대역 또는 직교 주파수 분할 다중화 방식)을 이용하여 두 대 이상의 장치를 연결하는 기술입니다. 이를 이용해 사용자는 근거리 지역에서 이동하면서도 지속적으로 네트워크에 접근할 수 있습니다.

오늘날 대부분의 무선랜 기술은 IEEE 802.11 표준에 기반하고 있으며, 와이파이(wi-fi)라는 이름으로 잘 알려져 있습니다. 무선랜은 한때 미국 국방부에 의해 LAWN (Local Area Wireless Network) 라고 불리기도 했습니다.

무선랜은 설치의 용이성으로 인해 가정 환경에서 매우 널리 쓰이게 되었습니다. 또한 커피숍과 같은 상업 시설들은 방문객들을 위해 무료로 사용이 가능한 무선랜 환경을 제공하기도 합니다. 주 이점으로는 가정이나 사무실에서 무선 네트워크 장비가 있는 곳이라면 무선 네트워크를 쉽게 사용할 수 있다는 편의성과 일반 노동 환경 밖에서도 인터넷에 접속할 수 있다는 휴대성, 장소를 옮겨 다니며 원하는 네트워크의 접속을 유지할 수 있다는 생산성, 무선 네트워크를 처음 설치만으로 하나 이상의 액세스 포인트를 지원할 수 있다는 배치면에서의 장점. 그 외에도 무선 네트워크는 기존의 장비를 사용하여 수많은 고객을 받아들일 수 있다는 확장성의 측면에서 다른 네트워크와의 통합의 용이가 입증되면서 무선랜은 인기를 끌게 되었으며 따라서 오늘날 소비자에게 파는 대부분의 컴퓨터들은 필수적인 모든 무선랜 기술이 미리 장착되어 출시되고 있습니다.

단점으로는 무선랜 성능이 좋지 않은 주변 컴퓨터가 무선 패킷을 가로챌 수 있을 뿐 아니라, 좋은 품질에 적은 돈을 소비하려는 사용자가 눈에 잘 띄는 곳에서 패킷을 가져갈 수 있다는 보안성과 범위를 넓히려면 리피터나 추가적인 액세스 포인트 구매가 필요하다는 점에서 지원범위의 한정, 다른 라디오 주파수 비슷하게 무선 네트워크 신호는 다양한 통신 간섭에 노출되어 있으며 따라서 신뢰성에 측면 또한 단점이 될 수 있다는 것입니다. 또한 대부분의 무선 네트워크는 일반적인 유선 네트워크에 비해 느리다는 속도의 단점이 있습니다.

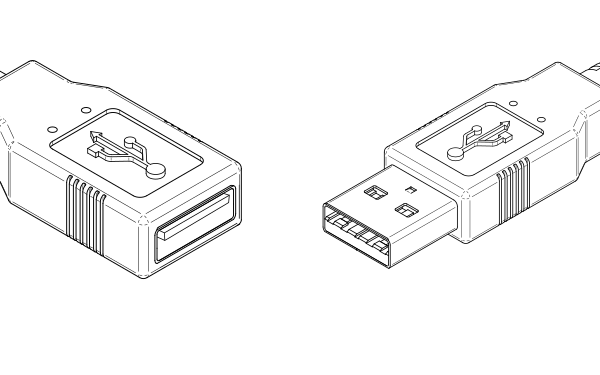
무선랜의 종류는 P2P ( P2P는 무선 장치가 직접 다른 컴퓨터와 통신할 수 있게 도와 준다.) ,브리지 (브리지는 보통 다른 종류의 네트워크를 연결하는 데 쓰일 수 있다. 브리지는 무선랜의 접속 포인트의 역할을 한다.)와 같은 것들이 있습니다.



<USB>

범용 직렬 버스(영어: Universal Serial Bus; USB, 범용 직렬 모선 (일반 직렬 모선))는 컴퓨터 본체와 다른 장치(주변 기기)를 연결하는 데 쓰이는 입출력 표준 프로토콜의 하나입니다. 직렬 포트, 병렬 포트 등 다양한 기존의 연결 방식을 대체하기 위하여 만들어졌습니다. 키보드, 마우스, 게임패드, 조이스틱, 스캐너, 디지털 카메라, 프린터, PDA, 저장장치와 같은 다양한 기기를 연결하는 데 사용되며, 이러한 기기 연결의 대부분은 표준 연결 방식을 이용하여 이루어집니다. USB는 PC를 위하여 개발되었지만, 지금은 PDA나 비디오 게임 콘솔 등에도 채택되어 사용되며, USB의 전원 공급 기능을 이용한 충전 용도로도 많이 사용됩니다. 경쟁 규격으로 언급되는 IEEE 1394가 주로 캠코더에 탑재되어 DV규격으로 주로 사용되는 반면에 USB는 다수의 저가격 기기에 채택되어 있습니다. USB의 구성으로는 USB의 가장 윗 부분에는 주 컨트롤러(host controller)가 있고, 이는 루트 허브를 통해 두 개의 USB 단자를 제공한다. 보통 이 단자에 주변 기기를 연결해 사용하며, 포트가 부족하면 허브를 연결하여 더 많은 포트를 마련할 수 있습니다. 하나의 주 컨트롤러에는 주변 기기를 나뭇가지 모양으로 127개까지 연결할 수 있다. USB 방식으로 연결된 주변 기기는 대부분 핫 스와핑을 지원하며, USB 방식으로 연결된 주변 기기에는 약간의 전력이 함께 공급됩니다. 따라서 보통은 외부 전원을 이용하지 않고도 주변 기기를 쉽게 사용할 수 있습니다. USB 2.0의 정격 전류용량은 5 V 500 mA이며, USB 3.2의 정격 전류용량은 5 V 900 mA이다. USB 표준의 이론상 최고 전송 속도는 다음과 같다. 직렬 통신보다 압도적인 빠른 전송의 비결은 하드웨어 기반의 압축 전송 기술입니다. USB 초기에는 Low speed로 연결되는 키보드, 마우스 같은 제품들이 있었지만 이제는 거의 쓰이지 않고 기존 인터페이스의 이용성 문제를 해결하고 USB에 장착된 모든 장치들의 소프트웨어 구성을 단순케 하며 외부 장치를 위한 데이터 속도를 증가시킬 수 있게 하여 speed면에서도 발전하였습니다. 그러나 USB의 속도에는 치명적인 제약이 있는데 주 컨트롤러(host controller)에 연결된 기기들 간에는 대역폭을 나누어 쓰게 되므로 장치가 늘어날수록 속도는 현저히 떨어집니다. USB 케이블은 USB 3.0 표준은 최대 케이블 길이는 규정하지 않으며 단지 모든 케이블이 전기 사양을 준수할 것을 요구한다: AWG(미국 전선 규격) 26와이어의 구리 케이블에서 실질적인 최대 길이는 3미터이다. 또한 USB에는 암놈과 숫놈이라고 결합부를 칭하는데 설명과 그림은 다음과 같습니다. 또한 USB 로고가 결합 시 같은 방향이 되도록 일정하게 설계 및 디자인 되어있습니다. 따라서 USB 암놈(female)의 USB 로고는 숫놈(male)의 USB 로고와 같은 위치에서 결합되도록 놓입니다.

(예시) USB의 암놈(왼쪽)과 숫놈(오른쪽)



<Bluetooth>

블루투스(영어: Bluetooth)는 1994년에 에릭슨이 최초로 개발한 디지털 통신 기기를 위한 개인 근거리 무선 통신 산업 표준입니다. ISM 대역에 포함되는 2.4~2.485GHz의 단파 UHF 전파를 이용하여 전자 장비 간의 짧은 거리의 데이터 통신 방식을 규정하는 블루투스는 개인용 컴퓨터에 이용되는 마우스, 키보드를 비롯해, 휴대전화 및 스마트폰, 태블릿, 스피커 등에서 문자 정보 및 음성 정보를 비교적 낮은 속도로 디지털 정보를 무선 통신을 통해 주고받는 용도로 사용되고 있습니다. 블루투스는 수 미터에서 수십 미터 정도의 거리를 둔 정보기기 사이에, 전파를 이용해서 간단한 정보를 교환하는데 사용됩니다. IEEE에서는 규격명 IEEE 802.15.1으로 등재되어 있으나, 현재 블루투스는 Bluetooth Special Interest Group (SIG)을 통해 관리되고 있습니다. 블루투스 SIG는 규격의 개발을 감시, 규격의 인증 프로그램의 관리 및 트레이드마크의 보호를 관장하고 있습니다. 또한 장비 제조사가 블루투스 장비로 인증을 받기 위해서는, SIG에서 제정한 표준 규격을 만족해야 합니다. 또한 블루투스의 사양과 특성을 보면 PANs의 산업 표준 중 하나인 블루투스는 ISM 대역인 2.45GHz를 사용합니다. 블루투스는 RS-232, 유선 USB를 대체하는 개념이며 이는 와이파이가 이더넷을 대체하는 개념과 유사하다. 암호화에는 SAFER을 사용한다. 장치끼리 믿음직한 연결을 성립하려면 키워드를 이용한 페어링이 이루어지는데, 이 과정이 없는 경우도 있습니다. 최근 블루투스 사양서 버전은 다음과 같은데 Bluetooth SIG는 블루투스 사양서 버전 4.0(Bluetooth Smart)이며 Bluetooth high speed는 Wi-Fi를 바탕으로, 클래식 블루투스는 기존의 레거시 블루투스 프로토콜을 바탕으로 합니다. 프로파일로는 블루투스는, 여러 종류의 장비로의 통신에 사용되는 규격인 이유로, 장비의 종류에 따라 규정되는 각각의 별도의 프로토콜이 존재합니다. 또한 이들 프로토콜의 사용법을 프로파일이라는 용어로 표준화하고 있습니다. 또한 통신하고자 하는 장비와 장비 간에, 동일한 프로파일을 가지고 있는 경우에만, 그 프로파일을 이용한 통신이 가능합니다.

대표적인 프로파일은 GAP(블루투스 장비의 접속/인증/암호화를 규정하는 프로토콜), SDAP (다른 블루투스 장비가 제공하는 기능(프로파일)을 참조하는 용도의 프로토콜), DUN (휴대전화 등을 통해 인터넷에 다이얼 업 접속을 할 때 사용되는 프로파일), FTP (컴퓨터 사이의 데이터 통신을 위한 프로파일),HID (컴퓨터 마우스 또는 키보드 등의 입력장비와 무선으로 연결하기 위한 프로파일),HSP (블루투스를 내장한 헤드셋과의 통신을 위한 프로파일. 모노 음성 수신 뿐만 아니라, 마이크로의 쌍방 통신도 규정.), HFP (차내 또는 헤드셋을 통한 핸즈프리 통화를 위한 프로파일. HSP 기능에 추가로 통신의 발신/착신 기능을 규정.), A2DP (음성을 리시버가 달린 헤드폰(또는 이어폰)으로 전송하는 프로파일. HSP/HFP가 모노 음성인 것에 반해, 스테레오 고음질 음성을 지원) , AVRCP (AV기기의 리모콘 기능을 구현하는 프로파일)과 같은 것들이 있으며 클래스 즉 블루투스는 통신에 이용되는 전파의 강도를 클래스라는 개념으로 규정하고 있습니다. 블루투스 지원 장비는 어떤 장비든 한 클래스로 분류됩니다. 쌍방이 같은 클래스일 필요는 없습니다. 또한 블루투스 클래스를 간단히 보면 Class 1은 출력 100mW, 도달거리 100m, Class 2는 2.5mW,10m이며 Class3 는 출력 2.5mW, 도달거리 10m, Class 4는 출력 0.5mW 로 볼 수 있습니다.

<GPIO>

다용도 입출력(general-purpose input/output, GPIO)은 입력이나 출력을 포함한 동작이 런타임 시에 사용자에 의해 제어될 수 있는 집적 회로나 전기 회로 기판의 디지털 신호 핀입니다.

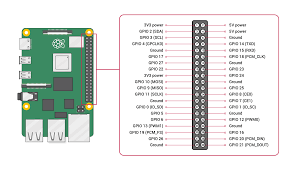
기존 컴퓨터(PC)에서는 그 내부 버스 구조와의 정형화된 인터페이스를 통해서만 가능하며 외부 주변장치와의 연결이 가능합니다. GPIO는 특정한 목적이 미리 정의되지 않으며 기본적으로는 사용되지 않습니다. 또한 계측기, 마이크로 컨트롤러, 임베디드 시스템 등에서는 GPIO를 통해 꽤 많은 범용 입출력 핀들을 사용하며, 외부 주변장치와 CPU간의 자료중개를 담당케 합니다.

또한, 칩 외부 핀에 바로 배선이 가능하고, 외부 장치를 구동할 정도의 전력도 제공하는 등 GPIO 포트를 통해, 전기적 신호의 출력 및 입력이 가능하도록 할 수 있으며 GPIO는 어셈블리 레벨의 회로망 설계자(집적 회로 GPIO는 회로 기판 설계자, 기판 레벨 GPIO는 시스템 통합자, S/I)에 의해 구현되어 있으며 사용 시에는 GPIO의 목적과 동작이 정의됩니다. 이때 설계자에 의해 결정된다는 것, control을 통해 I/O로 쓸 수 있는데 때에 따라 input, output으로 써야 하는데 이런 I/O는 Register를 통해서 programmable합니다.

이때 Pin을 통해 Mode를 정할 때는 GPIO로 Alternative Functionality로 쓰고, Alternative로 설정하면 GPIO로는 못쓰니까 off되고 off되면 하드웨어적으로 정해진 임무를 수행하게 됩니다. Pin의 상태를 활성화하면서는 Data Direction을 정할 수 있습니다. 이때는 지금부터 Pin이 사용 가능 하도록 Init하고, input, output이냐를 결정합니다.

GPIO는 Hardware면에서는 pin이 한 개 밖에 없는데, 이 pin을 Input과 Output 모두로 사용가능 하며 3 state buffer, Tristate Buffer로 구현이 됩니다. 3 state Buffer는 1, 0, High impedance (Hi-Z) 요런 3가지 종류의 상태를 가질 수 있는 Buffer를 말하며 Switch가 1일 때는 I값이 O에 그대로 전달되고 Swtich가 0일 때는 I값이 O에 전달되지 않고 회로가 열려 버려서 마치 회로가 Open 된 것처럼 됩니다. 즉 1, 0 (Output), High impedance (Input)이런 상태로 생각 할 수 있습니다.

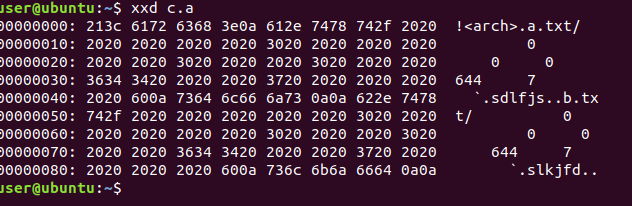
그리고, 제어 전용의 CPU등에서는 GPIO 핀 제어를 통해, GPIO 관련 제어용 수십~수백개의 레지스터를 이용하여 외부 제어가 가능한데 ARM CPU는 수십~수백개의 GPIO 포트 지원하며, BCM2835 CPU (라즈베리파이 RPi 1 장착)에는 41개 GPIO 제어 레지스터들이 있으며 54개 GPIO 포트 지원합니다.



[리눅스에서 binutils 실습]

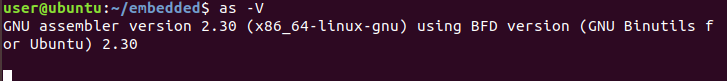
1. ar

 archive(static library)를 만들고 수정하고 푸는데 사용합니다. 아래 사진은 ar r c.a a.txt b.txt를 입력한 것으로 a.txt와 b,txt를 c.a 파일로 만드는데 사용하였습니다. 이후 ls를 통해 c.a파일이 생성된 것을 확인 할 수 있습니다. 그 아래 그림은 xxd c.a를 통해 그 내용에 대해서 확인 한 것입니다.



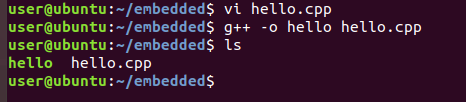
2. as

assembler (GNU Assembler) assembler의 version확인을 위한 옵션 -V를 같이 붙여 as -V를 작성한 결과입니다.



3. g++

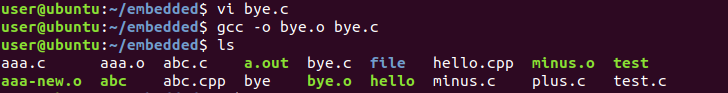
리눅스에서 c++ 프로젝트를 만들어 컴파일 하는 것입니다. vi hello.cpp파일로 cpp파일을 작성한 뒤 hello라는 이름의 object파일로 컴파일 하였습니다. 컴파일 된 파일을 실행한 결과 Hello world가 출력된 것을 통해 컴파일이 정상적으로 된 것을 확인하였습니다.





4. gcc

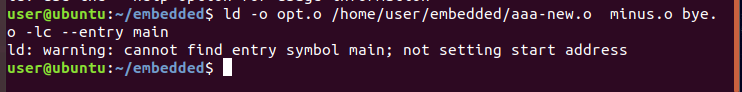
리눅스에서 c 프로젝트를 만들어 컴파일 하는 것입니다. Vi bye.c 파일로 c파일을 작성한 뒤 bye.o라는 이름의 object파일로 컴파일 하였습니다. ls를 통해 bye.o 가 정상적으로 컴파일 된 것을 확인하였습니다.



5. ld

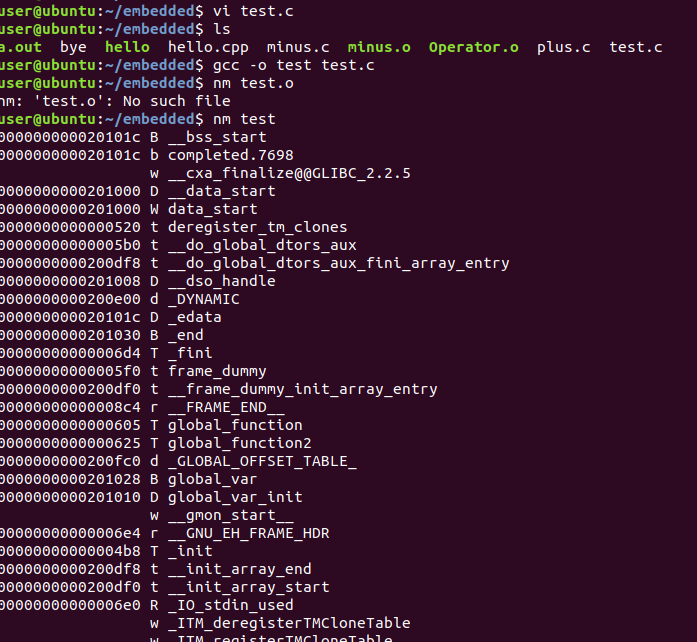
링킹을 위해 사용 합니다. (컴파일에서 생성된 objfile들을 하나로 묶는데 사용합니다.)

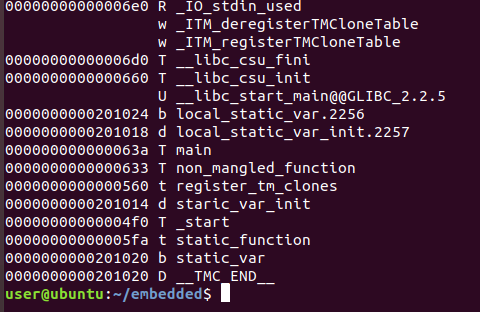
아래 그림은 ld -o opt.o 저장할 path(/home/user/embedded/aaa-new.o에 minus.o 와 bye.o라는 오브젝트 파일을 aaa-new.o라는 컴파일에서 생성한 objfile들을 하나로 묶었습니다.)



6. nm

object 파일의 심볼을 출력 하여 줍니다. 아래 그림은 test.c파일을 만든 뒤 test 이름으로 object파일로 작성 한 뒤, nm test를 통해 심볼을 출력한 결과 입니다.

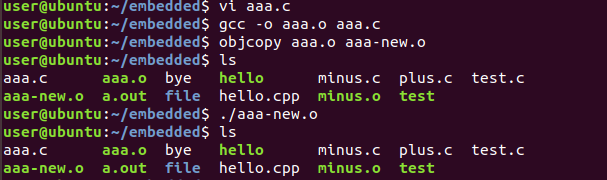




7. objcopy

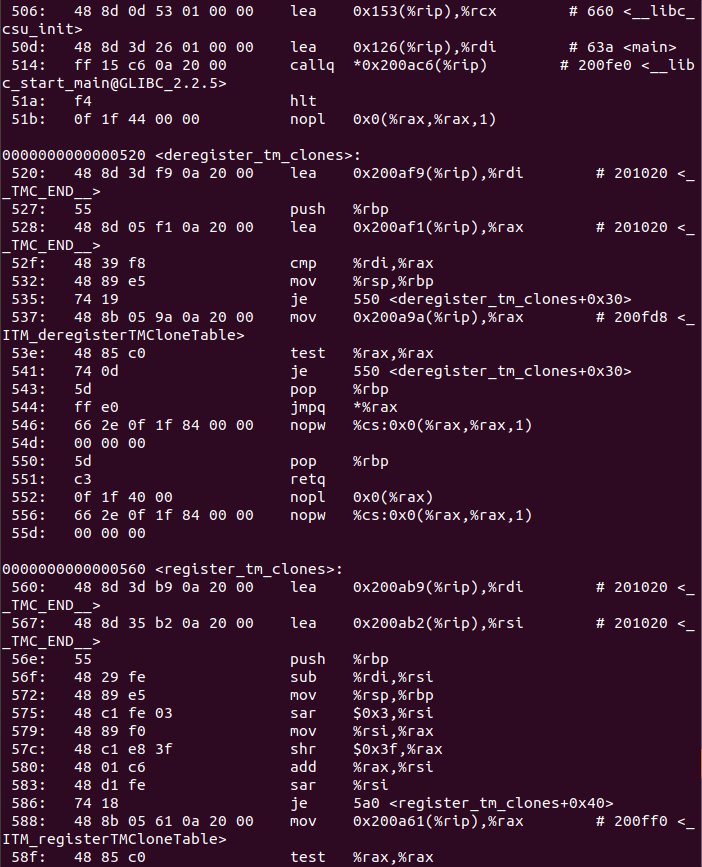
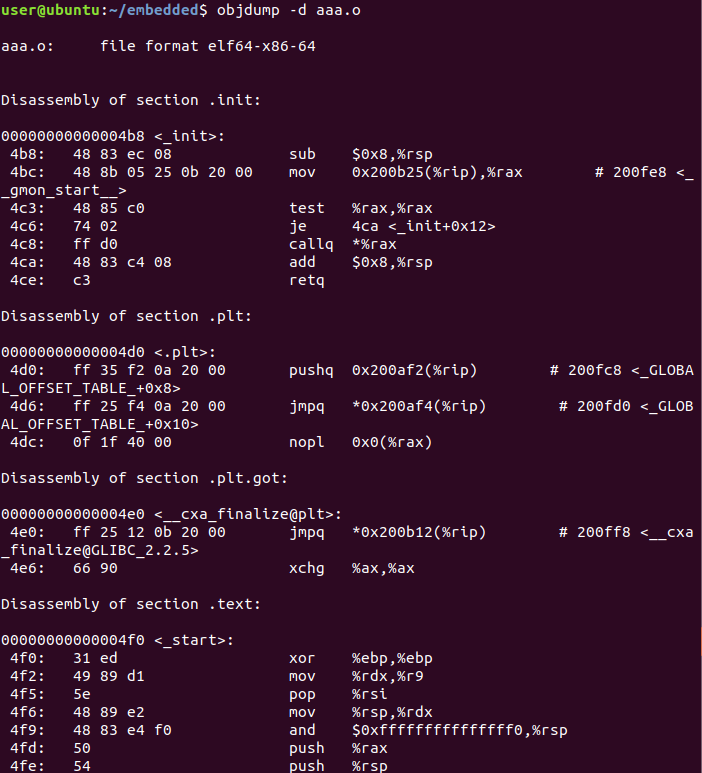
오브젝트 파일의 내용을 다른 파일로 복사하기 위해 사용합니다.

아래 그림은 aaa.c파일을 작성한 뒤 aaa.o파일로 object파일을 생성한 뒤 objcopy aaa.o aaa-new.o를 통해 aaa-new.o라는 새로운 오브젝트파일로 aaa.o를 copy하였습니다. Ls로 확인한 결과 aaa-new.o가 생성된 것을 확인 할 수 있습니다.



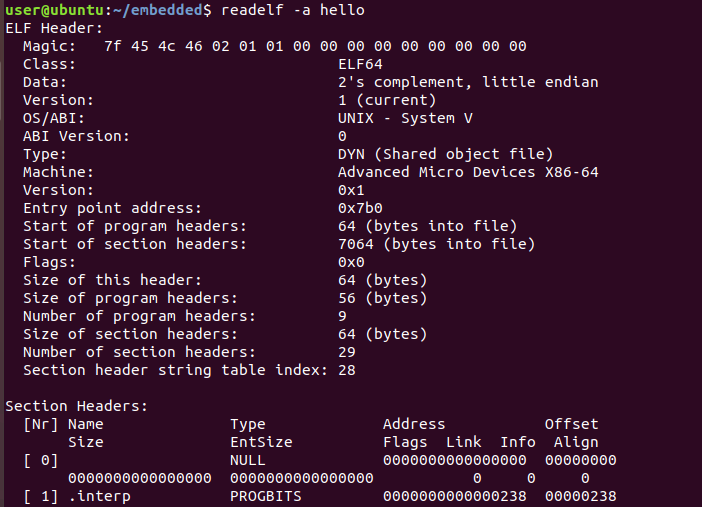
8. objdump

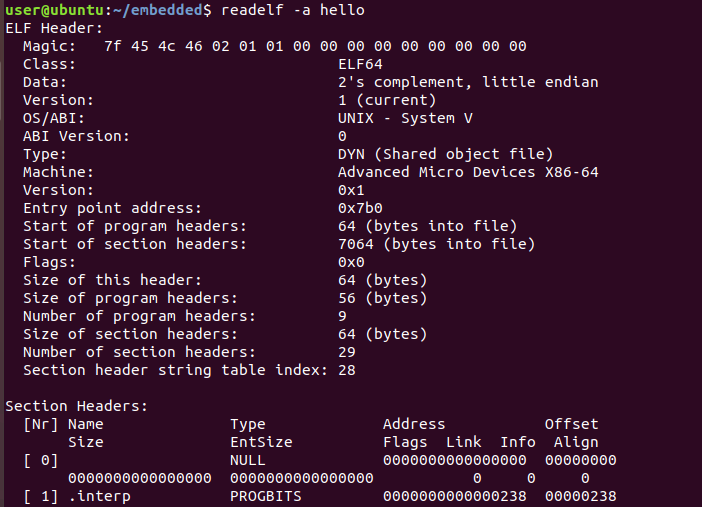
오브젝트 파일에 대한 정보를 출력합니다. dump 된내용은 standard out으로 나오니  redirection 하여 파일로 만들 수 있습니다. objdump -d aaa.o을 명령어 작성한 결과 이며 -d옵션을 통해 결과를 assembly code만 보여 주는 작업을 수행한 결과입니다.



9. readelf

ELF 파일에 대한 내용을 출력 합니다. -a옵션을 통해 전체 내용을 출력하였으며 그 아래 ELF Header:…와 같이 hello라는 오브젝트파일의 전체 ELF내용을 출력합니다.

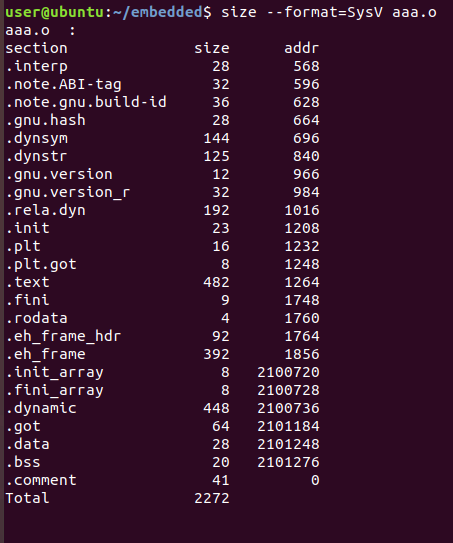




10. size

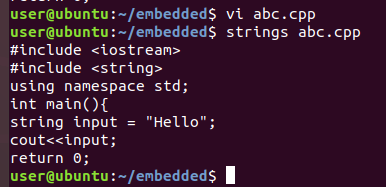
application의 각각의 영역(text, data, uninitialized sections)에 대한 크기와 전체 크기를 출력합니다.

size --format=SysV aaa.o 를 통해 aaa.o에 대한 크기와 전체크기를 출력합니다.



11. strings

표시할 수 있는 문자열을 출력합니다. abc.cpp에 있는 내용(문자열)이 출력된 것을 확인 할 수 있었습니다.



12. strip

strip aaa.o 즉, aaa.o 이름의 오브젝트 파일로 부터 심볼을 제거 합니다. Nm을 통해 obj파일의 심볼 내용을 확인 한 결과 no symbols, 즉 내용이 존재하지 않는 것을 확인 할 수 있었습니다.

