RISC-V 64 비트 FU540 셋업과 SDK 빌드 방법

최신 CPU 아키텍쳐 RISC-V 분석 및 실습

RISC-V 64 비트 FU540 셋업과 SDK 빌드 방법

커널연구회 (www.kernel.bz)

정재준 (rgbi3307@nate.com)

목차

Table of Contents

● RISC-V 64 비트 FU540 셋업과 SDK 빌드 방법 1

● 목차 2

● RISC-V 64 비트 FU540 부팅 3

FU540 보드 셋업하기 4

FU540 보드 실행하기 8

● FU540 소스 빌드 및 설치 8

다운로드 하기 9

빌드하기 10

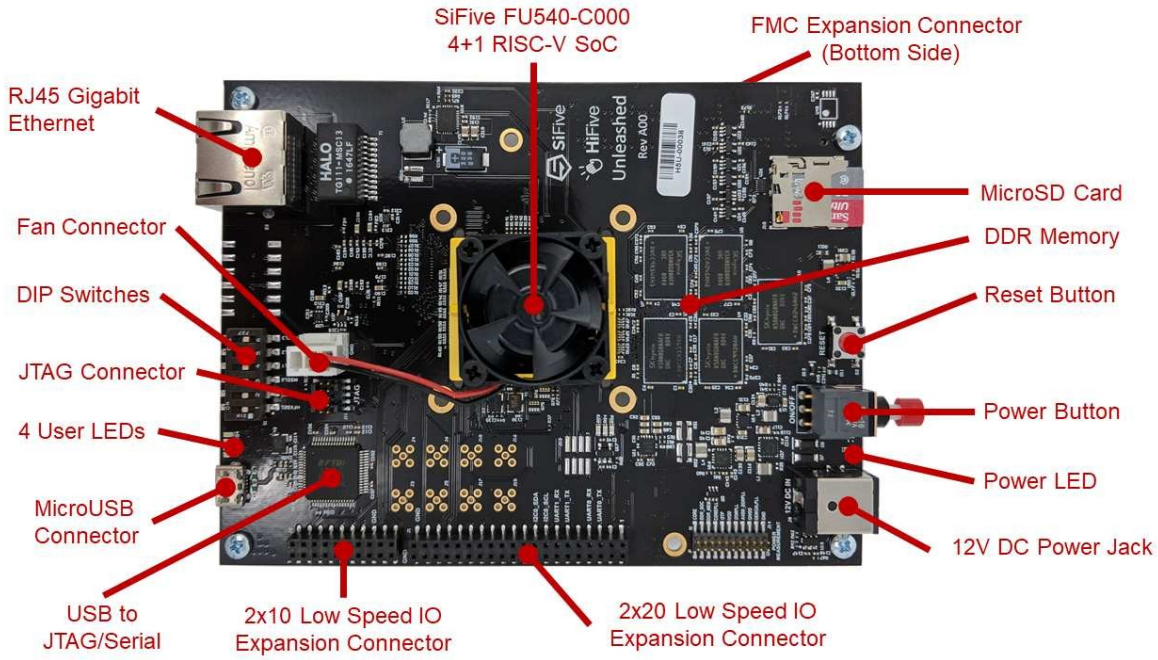
설치하기 12

부팅하기 17

RISC-V 64 비트 FU540 부팅

HiFive Unleashed 보드는 SiFive 의 64 비트 RISC-V 코어 4 개가 내장된 Freedom U540 이 탑재되어 있습니다. 이 보드는 리눅스 개발 플랫폼이며, 8GB DDR4, 32MB QuadSPI Flash, 기가비트 이더넷 포트, MicroSD 카드가 장착되어 있습니다. 또한 PCI Express(PCIe)을 지원하고 FMC 커넥터와 I/O 확장핀들을 통해서 여러가지 장치들을 연결할 수 있습니다. 이 보드의 외형은 다음과 같습니다.

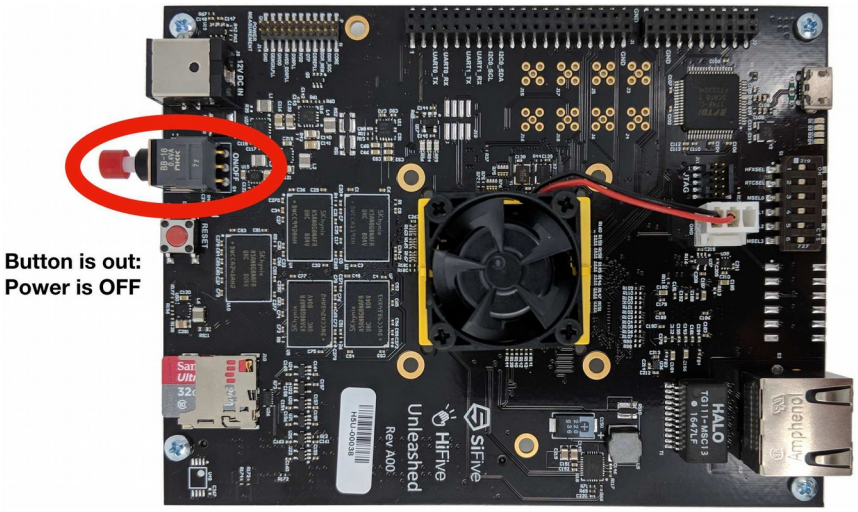
HiFive Unleashed(FU540) 보드 외형



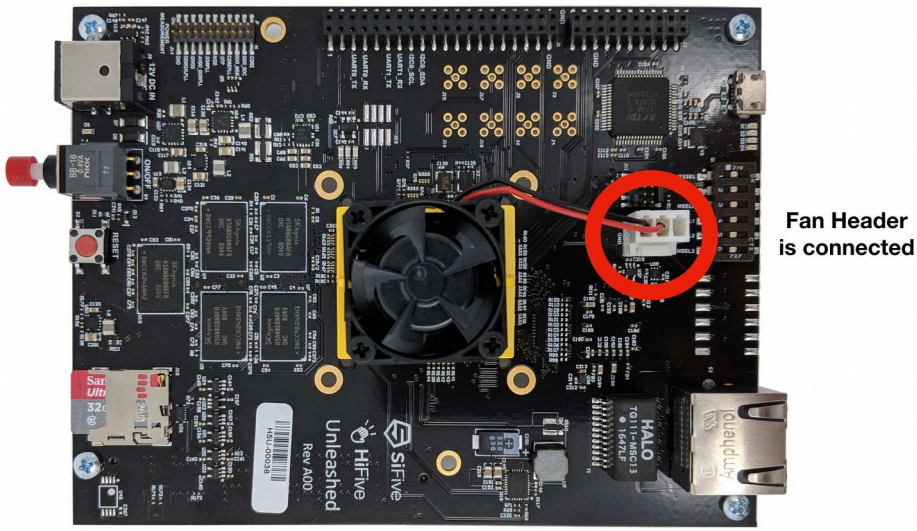
FU540 보드 셋업하기

다음과 같은 순서대로 보드 동작을 준비합니다.

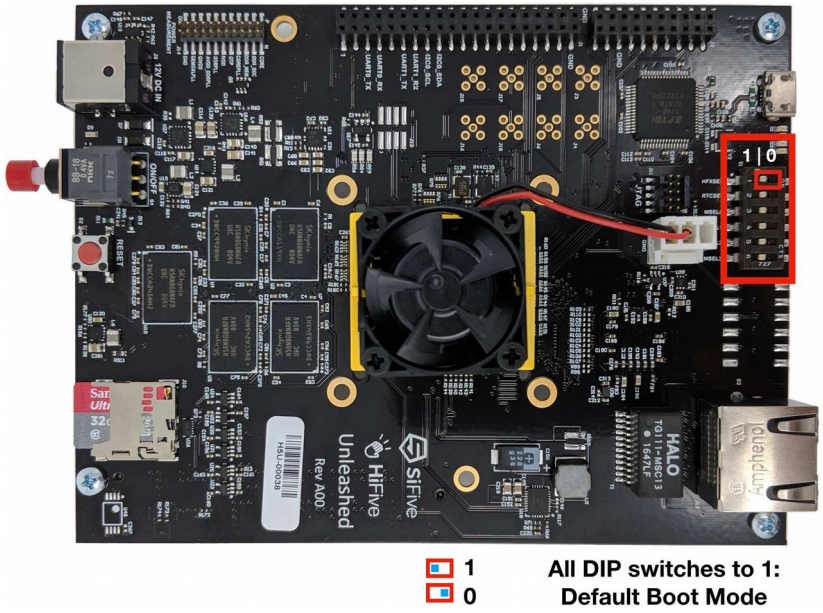
1. 먼저, 다음 그림과 같이 토글(push/pop) 전원 스위치를 pop 하여 Power Off 합니다.



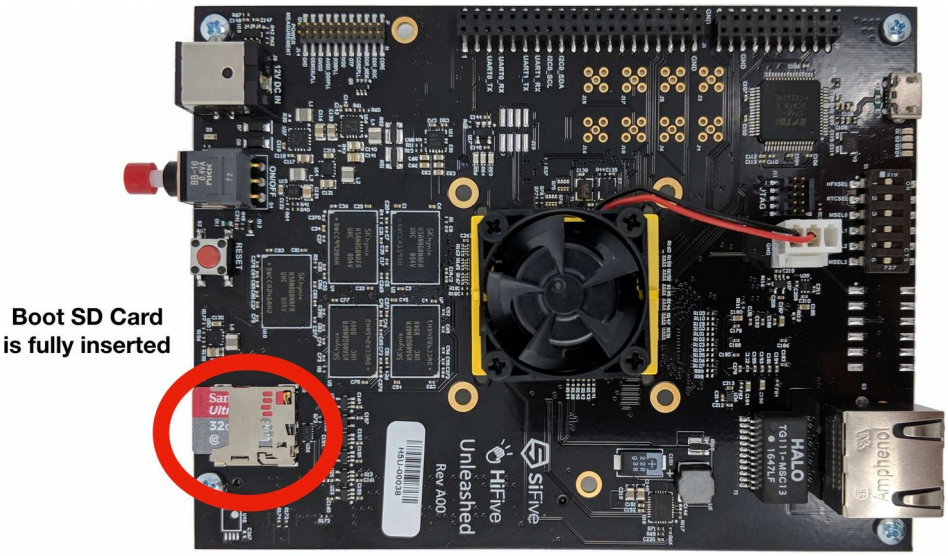
2. CPU 냉각팬이 헤더에 제대로 연결되어 있는지 확인합니다.



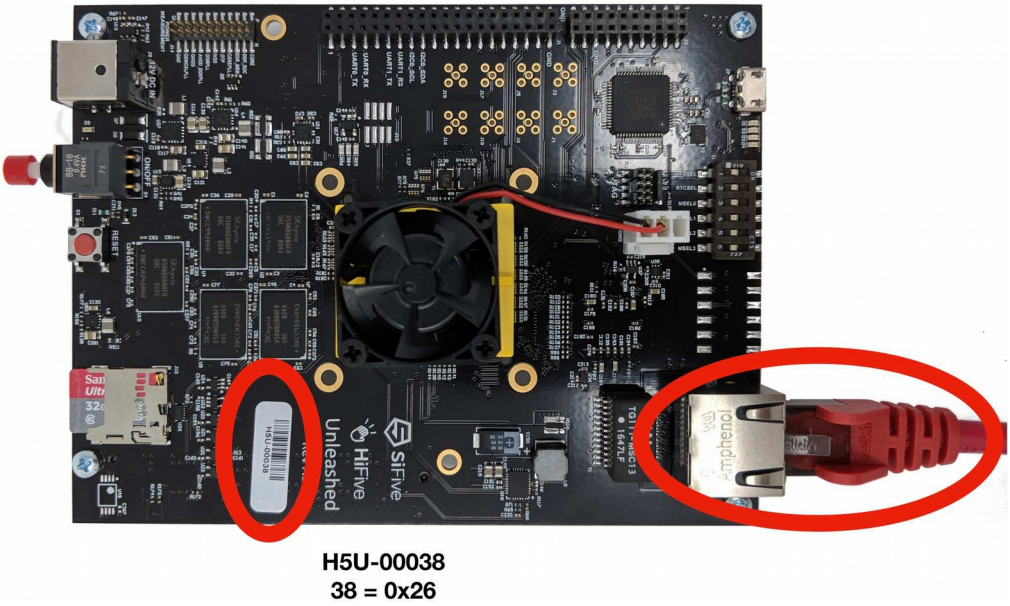
3. 딥스위치들을 모두 왼쪽으로 밀어서 모두 1 의 상태가 되도록 합니다. 이렇게 하면 MSEL 이 1111 로 되어서 Default Boot Mode 가 됩니다.



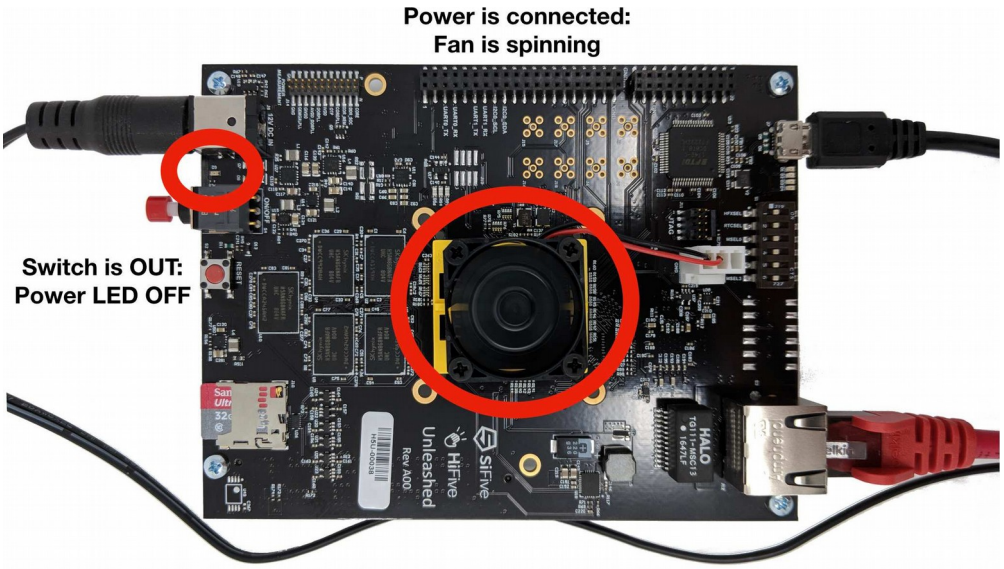
4. 리눅스가 탑재된 SD 카드를 삽입합니다. (SD 카드 설치 방법은 다음장 참조)



5. 다음과 같이 이더넷 케이블을 연결합니다. 보드가 부팅될때 DHCP 와 ssh 서버가 가동 됩니다.

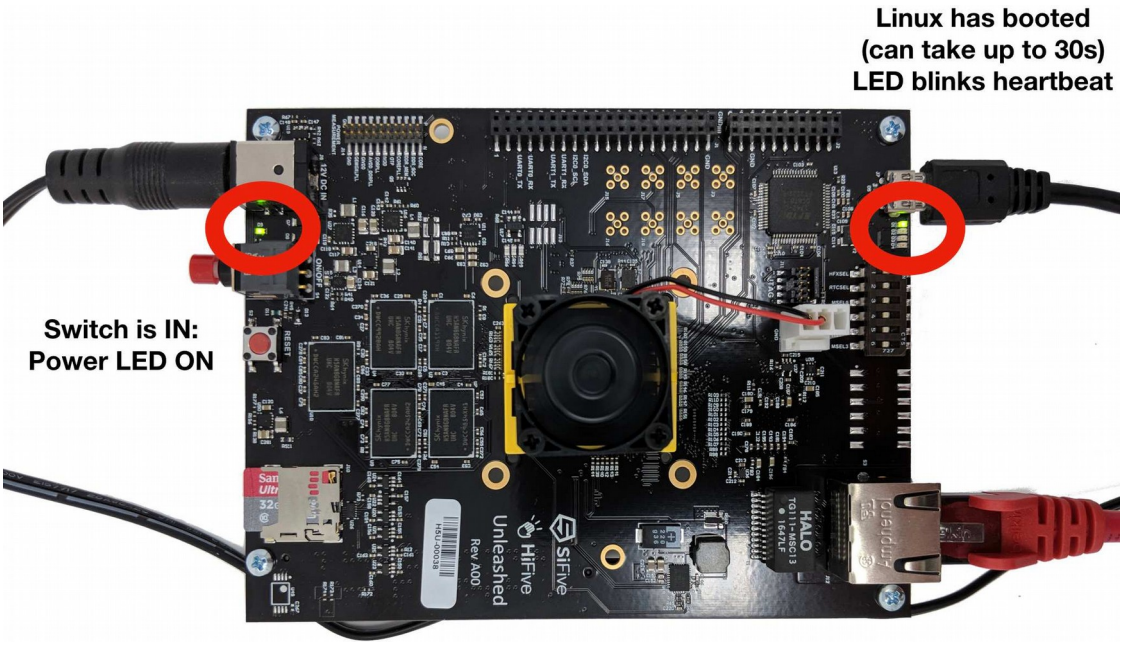


6. 12V 전원 케이블을 연결합니다. 12V 전원 케이블을 연결하면 토글 전원 스위치를 On 하지 않아도 CPU 냉각팬이 회전하기 시작합니다. 아직 보드는 부팅되지 않습니다. 마이크로 USB 케이블을 PC 에 연결합니다. 토글 전원 스위치를 Push 하여 On 하면 보드가 부팅됩니다.



마이크로 USB 케이블을 통하여 115200bps 의 시리얼 콘솔이 동작하고 부팅 메세지가 시리얼 콘솔에 출력됩니다. 아울러 openocd 와 함께 사용되는 JTAG 도 같이 실행할 수 있습니다.

전원이 인가된후 약 20 초가 지나면 리눅스 부팅이 끝나고 LED 가 깜박이기 시작합니다.



부팅이 마무리 되었습니다. 이제 다음장부터 리눅스 명령을 통하여 보드를 실행해 보겠습니다.

FU540 보드 실행하기

HiFive FU540 보드는 시리얼과 이더넷(ssh)을 통하여 여러가지 실행 작업을 할 수 있습니다.

시리얼은 ssh 에 비해서 속도가 느리므로 ssh 사용을 권장합니다.

전원이 인가되고 부팅될때 이더넷에 연결된 IP 주소를 DHCP 로 가지고 오고 ssh 서버가 시작됩니다.

MAC 주소는 70:b3:d5:92:fX:XX 이고 X:XX 는 보드의 시리얼 번호를 16 진수로 변환한 값입니다.

호스트명은 hifiveu 이고 r**oot 사용자 계정의 암호는 sifive** 입니다.

FU540 소스 빌드 및 설치

HiFive FU540 소스(부트로더, 커널)는 리눅스 우분투(Ubuntu 16.04) 환경에서 개발하고 테스트 되었습니다. 먼저 리눅스 우분투에 다음과 같은 패키지들을 설치 합니다.

패키지 설치

```
$ sudo apt-get install autoconf automake autotools-dev bc bison \
        build-essential curl flex gawk gdisk git gperf libgmp-dev \
        libmpc-dev libmpfr-dev libncurses-dev libssl-dev libtool \
        patchutils python screen texinfo unzip zlib1g-dev
```

다운로드 하기

위의 패키지 설치가 완료되면, 다음과 같이 freedom-u-sdk 을 다운로드 합니다.

freedom-u-sdk 다운로드

```
$ git clone https://github.com/sifive/freedom-u-sdk.git
$ cd freedom-u-sdk
$ git submodule update --init --recursive
```

이 작업은 네트워크 속도에 따라서 30 분에서 60 분정도 소요 됩니다. 위의 작업이 마무리 되면

다음과 같이 freedom-u-sdk 경로에 파일들이 다운로드 되어 저장되어 있습니다.

freedom-u-sdk 파일 리스트

```
jungjaejoon@HP7100U:~/Projects/risc-v/freedom-u-sdk$ ll
total 76
drwxr-xr-x 12 jungjaejoon jungjaejoon 4096  2월 25 17:43 ./
drwxrwxr-x  8 jungjaejoon jungjaejoon 4096  2월 25 17:43 ../
drwxr-xr-x  3 jungjaejoon jungjaejoon 4096  2월 25 17:43 bsp/
drwxr-xr-x 13 jungjaejoon jungjaejoon 4096  2월 25 17:59 buildroot/
drwxr-xr-x  2 jungjaejoon jungjaejoon 4096  2월 25 17:43 conf/
drwxr-xr-x  9 jungjaejoon jungjaejoon 4096  2월 25 18:40 .git/
-rw-r--r--  1 jungjaejoon jungjaejoon   31  2월 25 17:43 .gitignore
-rw-r--r--  1 jungjaejoon jungjaejoon  657  2월 25 17:43 .gitmodules
drwxr-xr-x 24 jungjaejoon jungjaejoon 4096  2월 25 18:40 linux/
-rw-r--r--  1 jungjaejoon jungjaejoon 8764  2월 25 17:43 Makefile
-rw-r--r--  1 jungjaejoon jungjaejoon 2169  2월 25 17:43 README.md
drwxr-xr-x  4 jungjaejoon jungjaejoon 4096  2월 25 18:00 riscv-fesvr/
drwxr-xr-x 12 jungjaejoon jungjaejoon 4096  2월 25 18:00 riscv-gnu-
toolchain/
drwxr-xr-x  9 jungjaejoon jungjaejoon 4096  2월 25 18:31 riscv-isa-sim/
drwxr-xr-x  9 jungjaejoon jungjaejoon 4096  2월 25 18:31 riscv-pk/
drwxr-xr-x 45 jungjaejoon jungjaejoon 4096  2월 25 18:31 riscv-qemu/
```

```
-rw-r--r--  1 jungjaejoon jungjaejoon  419  2월 25 17:43 .travis.yml
```

빌드하기

이제 위의 소스 파일들을 다음과 같은 명령으로 빌드하도록 하겠습니다.

freedom-u-sdk 빌드하기

```
$ unset RISCV
$ make
```

위의 빌드 작업은 컴퓨터 속도에 따라서 차이가 있지만 여러 시간이 걸립니다. 혹시 빌드 진행중에

다음과 같은 오류 메세지가 나타날 수 있습니다.

오류 메세지

```
autoreconf:
/home/jungjaejoon/Projects/risc-v/freedom-u-sdk/work/buildroot_initramfs/
host/usr/bin/automake failed with exit status: 255
package/pkg-generic.mk:185: recipe for target
'/home/jungjaejoon/Projects/risc-v/freedom-u-sdk/work/buildroot_initramfs/
build/libtirpc-1.0.1/.stamp_configured' failed
make[1]: ***
[/home/jungjaejoon/Projects/risc-v/freedom-u-sdk/work/buildroot_initramfs/
build/libtirpc-1.0.1/.stamp_configured] Error 1
make[1]: Leaving directory '/home/jungjaejoon/Projects/risc-v/freedom-u-
sdk/buildroot'
Makefile:90: recipe for target '/home/jungjaejoon/Projects/risc-v/freedom-
u-sdk/work/buildroot_initramfs/images/rootfs.tar' failed
make: ***
[/home/jungjaejoon/Projects/risc-v/freedom-u-sdk/work/buildroot_initramfs/
images/rootfs.tar] Error 2
```

위의 소스들은 우분투 16.04 버전에서 테스트 되었기 때문에 우분투 18.04 버전에서 빌드하게 되면 오류가 나타날 수 있습니다. 인터넷에서 검색해 보면 위와 같은 오류 내용이 보고 되고 있습니다. 이것을 필자는 다음과 같이 해결 했습니다.

우선 현재 작업경로인 freedom-u-sdk 경로 하위에 있는 buildroot 경로로 진입하여 아래와 같이 명령을 실행합니다.

추가 명령 실행

```
$ cd ./buildroot
$ git checkout sifive
$ make -j4

/* 환경설정 메뉴가 나타나면 저장하고 나와서 아래 명령 실행 */
$ cd ..
$ make
```

위와 같이 하면 이전에 빌드되던 내용이 계속 빌드되어 다음과 같이 완료됩니다.

빌드 완료 메세지

```
/* 이상 생략.. */

This image has been generated for an ISA of rv64imafdc and an ABI of lp64d
Find the image in work/bbl.bin, which should be written to a boot partition

To completely erase, reformat, and program a disk sdX, run:
  sudo make DISK=/dev/sdX format-boot-loader
  ... you will need gdisk and e2fsprogs installed
```

빌드된 파일들을 다음과 같이 work 경로에서 확인할 수 있습니다. bbl.bin 파일은 부트로더와 커널 이미지, 루트 파일시스템이 같이 합쳐진 파일입니다. buildroot_initramfs_sysroot 경로에는 리눅스 파일시스템 파일들이 있습니다. 이 파일들이 SD 카드에 설치 됩니다.

빌드된 파일 확인

```
jungjaejoon@HP7100U:~/Projects/risc-v/freedom-u-sdk$ ll work/
total 63540
drwxr-xr-x  7 jungjaejoon jungjaejoon     4096  2월 26 14:12 ./
drwxr-xr-x 17 jungjaejoon jungjaejoon     4096  2월 26 12:17 ../
-rwxr-xr-x  1 jungjaejoon jungjaejoon 16771572  2월 26 14:12 bbl.bin*
-rw-r--r--  1 jungjaejoon jungjaejoon 50314716  2월 26 14:12 bbl.hex
drwxr-xr-x  6 jungjaejoon jungjaejoon     4096  2월 26 13:49
buildroot_initramfs/
drwxr-xr-x 16 jungjaejoon jungjaejoon     4096  2월 26 14:02
buildroot_initramfs_sysroot/
drwxr-xr-x 22 jungjaejoon jungjaejoon     4096  2월 26 14:12 linux/
drwxr-xr-x  9 jungjaejoon jungjaejoon     4096  2월 26 12:49 riscv-gnu-
toolchain/
drwxr-xr-x  2 jungjaejoon jungjaejoon    12288  2월 26 14:12 riscv-pk/
```

설치하기

위에서 빌드한 이미지들을 마이크로 SD 카드에 설치하기 위해서, 마이크로 SD 카드를 우분투 PC USB 포트에 연결하면 다음과 같이 인식된 정보를 확인할 수 있습니다. 먼저, lsusb 명령을 통해서 인식된 장치 ID 와 제조사 정보를 확인합니다.

lsusb 명령으로 장치 ID 확인

```
root@HP7100U:/home/jungjaejoon# lsusb
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 005: ID 04f2:b5d5 Chicony Electronics Co., Ltd
Bus 001 Device 004: ID 8087:0aa7 Intel Corp.
```

```
Bus 001 Device 003: ID 0458:00dc KYE Systems Corp. (Mouse Systems)
Bus 001 Device 002: ID 0c45:7616 Microdia
Bus 001 Device 008: ID 048d:1336 Integrated Technology Express, Inc. SD/MMC Cardreader
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

위에서 "Integrated Technology Express, Inc. SD/MMC Cardreader"가 필자가 장착한 마이크로 SD 카드 리더기의 장치 인식 정보입니다.

그리고, 다음과 같이 mount 명령으로 마이크로 SD 카드에 접근할 수 있는 장치 노드명(/dev/sdb)을 확인합니다. 이 노드명은 리눅스 우분투가 설치된 PC 환경에 따라서 다르게 나타날 수 있습니다.

mount 명령으로 장치노드 확인

```
root@HP7100U:/home/jungjaejoon# mount

/dev/sdb1 on /media/jungjaejoon/CANAKIT type vfat
(rw,nosuid,nodev,relatime,uid=1000,gid=1000,fmask=0022,dmask=0022,codepage=
437,iocharset=iso8859-1,shortname=mixed,showexec,utf8,flush,errors=remount-
ro,uhelper=udisks2
```

이제 마이크로 SD 카드의 파티션 정보들을 설정하고 포맷하는 작업을 하겠습니다. (리눅스 root 계정에서 명령을 실행합니다.) 먼저 다음과 같이 gdisk 명령을 실행합니다.

gdisk 명령 실행

```
root@HP7100U:/home/jungjaejoon# gdisk /dev/sdb
GPT fdisk (gdisk) version 1.0.3

Partition table scan:
  MBR: MBR only
  BSD: not present
  APM: not present
  GPT: not present
```

```
************************************************************
Found invalid GPT and valid MBR; converting MBR to GPT format
in memory. THIS OPERATION IS POTENTIALLY DESTRUCTIVE! Exit by
typing 'q' if you don't want to convert your MBR partitions
to GPT format!
************************************************************


Command (? for help): p
Disk /dev/sdb: 15523840 sectors, 7.4 GiB
Model: Storage Device
Sector size (logical/physical): 512/512 bytes
Disk identifier (GUID): BE34B56A-2439-4BD5-9F1E-156AAC1C6B69
Partition table holds up to 128 entries
Main partition table begins at sector 2 and ends at sector 33
First usable sector is 34, last usable sector is 15523806
Partitions will be aligned on 2048-sector boundaries
Total free space is 189890 sectors (92.7 MiB)

Number  Start (sector)    End (sector)  Size       Code  Name
   1            8192        15342074   7.3 GiB     0700  Microsoft basic
data

Command (? for help):
```

위에서 Command 명령 프롬프트에 ? 를 입력하여 명령어 도움말을 확인합니다.

```
Command (? for help): ?
b     back up GPT data to a file
c     change a partition's name
d     delete a partition
i     show detailed information on a partition
l     list known partition types
n     add a new partition
o     create a new empty GUID partition table (GPT)
p     print the partition table
q     quit without saving changes
r     recovery and transformation options (experts only)
s     sort partitions
t     change a partition's type code
v     verify disk
w     write table to disk and exit
x     extra functionality (experts only)
```

```
?       print this menu
```

위에서 Command 명령 프롬프트에 d 을 입력하여 기존에 존재하고 있는 파티션을 삭제합니다.

그런다음, Command 명령 프롬프트에 p 를 입력하여 마이크로 SD 카드의 파티션 정보를 다시

확인합니다. 참고로 필자는 8GB 크기의 마이크로 SD 카드를 사용했습니다.

```
Command (? for help): p
Disk /dev/sdb: 15523840 sectors, 7.4 GiB
Model: Storage Device
Sector size (logical/physical): 512/512 bytes
Disk identifier (GUID): BE34B56A-2439-4BD5-9F1E-156AAC1C6B69
Partition table holds up to 128 entries
Main partition table begins at sector 2 and ends at sector 33
First usable sector is 34, last usable sector is 15523806
Partitions will be aligned on 2048-sector boundaries
Total free space is 15523773 sectors (7.4 GiB)

Number  Start (sector)    End (sector)  Size       Code  Name

Command (? for help):
```

이제, 다음과 같이 명령 프롬프트에 o 을 입력하여 모든 파티션들을 지우고 새롭고 MBR 파티션을

생성합니다.

```
Command (? for help): o
This option deletes all partitions and creates a new protective MBR.
Proceed? (Y/N): Y
```

그런다음 w 을 입력하여 작업한 정보를 저장한후 종료 합니다.

```
Command (? for help): w

Final checks complete. About to write GPT data. THIS WILL OVERWRITE
EXISTING
PARTITIONS!!

Do you want to proceed? (Y/N): Y
```

```
OK; writing new GUID partition table (GPT) to /dev/sdb.
The operation has completed successfully.
```

위와 같이 SD 카드의 파티션 설정 작업이 마무리되면, 다음과 같이 빌드 이미지들을 SD 카드에 쓰넣는 명령을 실행합니다.

SD 카드에 빌드 이미지 쓰넣기

```
$ sudo make DISK=/dev/sdb format-boot-loader
```

위의 명령을 실행하면, 다음과 같이 SD 카드의 첫번째 파티션에는 부트로더+커널+루트파일시스템이 저장되고 두번째 파티션에는 리눅스 파일 시스템이 마운트 됩니다.

```
sgdisk --clear
\
      --new=1:2048:67583  --change-name=1:bootloader --typecode=1:2E54B353-
1271-4842-806F-E436D6AF6985   \
      --new=2:264192:     --change-name=2:root       --typecode=2:0FC63DAF-
8483-4772-8E79-3D69D8477DE4 \
      /dev/sdb
Setting name!
partNum is 0
Setting name!
partNum is 1
The operation has completed successfully.
Error: Could not find bootloader partition for /dev/sdb
Makefile:229: recipe for target 'format-boot-loader' failed
make: *** [format-boot-loader] Error 1
```

만일, 위와 같이 “Error: Could not find bootloader partition for /dev/sdb” 오류 메세지가 나타나면 SD 카드의 파티션 정보와 Makefile 의 format-boot-loader 실행 스크립트가 올바르지 않은 경우 입니다. 이 문제에 대해서는 아래의 SiFive 포럼 웹사이트에 해결책을 설명하고 있습니다.

https://forums.sifive.com/t/issues-creating-a-bootable-sd/2087

SD 에 부트로더, 커널, 루트 파일 시스템 설치 작업이 마무리 되었습니다. 이제, SD 카드를 FU540 타겟 보드에 장착하고 부팅하도록 하겠습니다.

부팅하기

마이크로 SD 카드를 FU540 타겟보드에 장착하고 전원 스위치를 켜면 다음과 같은 부팅 메세지가 시리얼 콘솔에 출력되면서 부팅됩니다. 부팅 시간은 약 20~30 초 정도 소요 됩니다.

부팅 메세지

```
SiFive FSBL:       2018-03-20
HiFive-U serial #: 00000316
bbl loader

                SIFIVE, INC.

         55555555555555555555555
        5555                   5555
       5555                     5555
      5555                       5555
     5555       555555555555555555
    5555       5555555555555555555555
   5555                             5555
  5555                               5555
 5555                                 5555
555555555555555555555555555           55555
 55555           555555555           55555
   55555           55555           55555
     55555           5           55555
       55555                   55555
         55555               55555
           55555           55555
             55555       55555
```

```
                55555   55555
                  555555555
                    55555
                      5

           SiFive RISC-V Coreplex

/** 중간 생략 **/

Starting logging: OK
Starting mdev...
[    1.096439] mmc0: host does not support reading read-only switch,
assuming write-enable
[    1.103701] mmc0: new SDHC card on SPI
[    1.107780] mmcblk0: mmc0:0000 SL08G 7.40 GiB
[    1.138922]  mmcblk0: p1 p2
sort: /sys/devices/platform/Fixed: No such file or directory
modprobe: can't change directory to '/lib/modules': No such file or
directory
Initializing random number generator... done.
Starting network...
udhcpc (v1.24.2) started
Sending discover...
[    3.558223] macb 10090000.ethernet eth0: link up (100/Full)
Sending discover...
Sending discover...
No lease, failing
Starting dropbear sshd: OK

Welcome to Buildroot
buildroot login:
```

---

buildroot login: 프롬프트가 나타나면 사용자 아이디는 root 을 입력하고 암호는 sifive 을 입력하여 로그인 합니다.

리눅스 명령어를 사용하여 여러가지 작업 할 수 있는 환경이 되었습니다. 이제 RISC-V 64 비트 명령들을 리눅스에서 마음껏 테스트 해 보시기 바랍니다.

궁금한 점들은 커널연구회 정재준(rgbi3307@nate.com)에게 메일 주시기 바랍니다.

감사합니다.