


 [sifive](#) / [freedom-u-sdk](#)

Freedom Unleashed Software Development Kit

★ 152 stars 🍴 101 forks

 Star Watch ▼<> **Code**

! Issues 42

🔗 Pull requests 6

▶ Actions

📁 Projects

📖 Wiki

🔗 2020.11 ▼

...

**davidlt** Update release notes ...

14 days ago 🕒 200

[View code](#)

README.md

SiFive Freedom Unleashed SDK

The new experimental Freedom Unleashed (FU) SDK is based on OpenEmbedded (OE). It's a minimal [meta-sifive](#) layer on top of [meta-riscv](#) to provide additional modifications and new disk image targets. Using OE you will be able to:

- build predefined disk images for QEMU and [SiFive HiFive Unleashed](#) development board (incl. [HiFive Unleashed Expansion Board](#) from Microsemi);
- build custom disk images with additional software packages from various third-party OE layers;
- quickly launch QEMU VM instance with your built disk image;
- build bootloader binaries (OpenSBI, U-Boot, U-Boot SPL);
- build Device Tree Binary (DTB);
- build Linux kernel images;
- easily modify disk partition layout.

[Berkeley Boot Loader \(BBL\)](#) is replaced by [OpenSBI](#) and uSD (microSD) disk images now also incl. FSBL (U-Boot SPL).

For more information on particular release see `ReleaseNotes` directory in [freedom-u-sdk](#) repository on GitHub.

The old SDK based on Buildroot is archived in [archive/buildroot](#) branch.

For advanced OE usage we advice to look into the following third-party manuals:

- [BitBake User Manual 3.1.2 by Yocto](#)
- [Yocto Project Reference Manual 3.1.2 by Yocto](#)
- [Yocto Project Complete Documentation \(MegaManual\) Set 3.1.2 by Yocto](#)
- [A practical guide to BitBake by Harald Achitz](#)

Quick Start

Install `repo` command from Google if not available on your host system. Please follow [the official instructions](#) by Google.

Then install a number of packages for BitBake (OE build tool) to work properly on your host system. BitBake itself depends on Python 3. Once you have Python 3 installed BitBake should be able to tell you most of the missing packages.

For Ubuntu 18.04 (or newer) install `python3-distutils` package.

Detailed instructions for various distributions can be found in "[Required Packages for the Build Host](#)" section in Yocto Project Reference Manual.

Creating Workspace

This needs to be done every time you want a clean setup based on the latest layers.

```
mkdir riscv-sifive && cd riscv-sifive
repo init -u git://github.com/sifive/meta-sifive -b 2020.11 -m tools/mar
repo sync
```

Creating a Working Branch

If you want to make modifications to existing layers then creating working branches in all repositories is advisable.

```
repo start work --all
```

Updating Existing Workspace

If you want to pull in the latest changes in all layers.

```
cd riscv-sifive
repo sync
repo rebase
```

Getting Build Tools (optional)

OpenEmbedded-Core requires GCC 6 or newer to be available on the host system. Your host system might have an older version of GCC if you use LTS (Long Term Support) Linux distribution (e.g. Ubuntu 16.04.6 has GCC 5.4.0). You could solve this issue by installing build tools. This requires less than 400MB of disk space. You can download pre-built one or build your own build tools.

Option 1: Installing OpenEmbedded-Core Build Tools (Pre-Built)

```
./openembedded-core/scripts/install-buildtools
```

The native SDK will be installed under `$BUILDDIR/./openembedded-core/buildtools` prefix.

Finally you should be able to use build tools:

```
./openembedded-core/buildtools/environment-setup-x86_64-pokysdk-linux
```

Option 2: Building Your Own Build Tools

Your host needs to have GCC 6 (or newer) or build tools installed from Option 1.

You can find pre-built tools from the same release source in GitHub release assets.

To build your own build tools execute the command below:

```
bitbake buildtools-extended-tarball
```

You can find the native SDK under `$BUILDDIR/tmp-glibc/deploy/sdk/` directory.

Now you can install build tools:

```
$BUILDDIR/tmp-glibc/deploy/sdk/x86_64-buildtools-extended-nativesdk-star
```

Finally you should be able to use your build tools:

```
. $BUILDDIR/../../openembedded-core/buildtools/environment-setup-x86_64-oes
```

Setting up Build Environment

This step has to be done after you modify your environment with toolchain you want to use otherwise wrong host tools might be available in the package build environment. For example, `gcc` from host system will be used for building `*-native` packages.

```
. ./meta-sifive/setup.sh
```

You can verify and fix your host tools by checking symlinks in `$BUILDDIR/tmp-glibc/hosttools` directory.

Configuring BitBake Parallel Number of Tasks/Jobs

There are 3 variables that control the number of parallel tasks/jobs BitBake will use: `BB_NUMBER_PARSE_THREADS`, `BB_NUMBER_THREADS` and `PARALLEL_MAKE`. The last two are the most important, and both are set to number of cores available on the system. You can set them in your `$BUILDDIR/conf/local.conf` or in your shell environment similar to how `MACHINE` is used (see next section). Example:

```
PARALLEL_MAKE="-j 4" BB_NUMBER_THREADS=4 MACHINE=freedom-u540 bitbake de
```

Leaving defaults could cause high load averages, high memory usage, high IO wait and could make your system unresponsive due to resources overuse. The defaults should be changed based on your system configuration.

Building Disk Images

There are two disk image targets added by meta-sifive layer:

- `demo-coreip-cli` - basic command line image (**recommended**);
- `demo-coreip-xfce4` - basic graphical disk image with [Xfce 4](#) desktop environment (requires HiFive Unleashed Expansion Board with supported GPUs, for example, **Radeon HD 6450** or Radeon HD 5450).

By default disk images do not include debug packages. If you want to produce disk images with debug packages append `-debug` (e.g. `demo-coreip-cli-debug`) to the disk image target.

There are two machine targets currently tested:

- `qemuriscv64` - RISC-V 64-bit (RV64GC) for QEMU virt machine;
- `freedom-u540` - SiFive HiFive Unleashed development board with or without HiFive Unleashed Expansion Board from Microsemi.

It's not possible to use disk images built for `freedom-u540` with QEMU 4.0 and instructions provided below.

Building disk images is CPU intensive, requires <10GB of sources downloaded over the Internet and <110GB of local storage.

Building disk image takes a single command which may take anything from 30 minutes to several hours depending on your hardware. Examples:

```
MACHINE=qemuriscv64 bitbake demo-coreip-cli
MACHINE=freedom-u540 bitbake demo-coreip-cli
MACHINE=freedom-u540 bitbake demo-coreip-xfce4
```

Running in QEMU

OE provides easy to use wrapper for QEMU:

```
MACHINE=qemuriscv64 runqemu nographic slirp
```

Running on Hardware

You will find all available build fragments (incl. disk images) in `$BUILDDIR/tmp-glibc/deploy/images/$MACHINE` where `MACHINE` is `freedom-u540` for this particular case.

Disk images files use `<image>-<machine>.<output_format>` format, for example,

`demo-coreip-cli-freedom-u540.wic.xz`. We are interested in `.wic.xz` disk images for writing to uSD card.

Be very careful while picking `/dev/sdX` device! Look at `dmesg`, `lsblk`, `blkid`, GNOME Disks, etc. before and after plugging in your uSD card to find a proper device. Double check it to avoid overwriting any of system disks/partitions!

Unmount any mounted partitions from uSD card before writing!

We advice to use 16GB or 32GB uSD cards. 8GB cards (shipped with HiFive Unleashed) can still be used with `demo-coreip-cli` CLI images.

Finally write uSD card:

```
xzcat demo-coreip-cli-freedom-u540.wic.xz | sudo dd of=/dev/sdX bs=512K
```

You will need to modify MSEL to allow using FSBL and OpenSBI + U-Boot bootloaders from uSD card instead of SPI-NOR Flash chip:

```

      USB      LED      Mode Select      Ethernet
+===|____|==*****==+-+--+--+--+--+=====|*****|===+
|                                     | | | | |X| | | | | |
|                                     | | | | | | |
|      HFXSEL->|X|X|X|X|X| |X|      |_____| |
|                                     +-+--+--+--+--+
|      RTCSEL-----/ 0 1 2 3 <--MSEL
|
|

```

You can login with `root` account. The password is `sifive`.

Connecting Using Serial Console

Connect your HiFive Unleashed to your PC using microUSB-USB cable to access serial console.

For macOS, run: `screen -L /dev/tty.usbserial-*01 115200`

For Linux, run: `screen -L /dev/serial/by-id/usb-FTDI_Dual_RS232-HS-if01-port0 115200`

The above commands might vary depending on your exact setup.

`-L` command will log all output to `screenlog.0` in your current working directory.

To quit screen, hit `ctrl - A` followed by `\` symbol. Finally agree to terminate all windows by typing `y`.

Connecting Using SSH

SSH daemon is started automatically.

The HiFive Unleashed behaves like any other network capable device (such as PC, laptop, and Single Board Computers like Raspberry Pi). Connect your HiFive Unleashed to your network (e.g. a router) and it will acquire IPv4 + DNS configuration using DHCP protocol. You can use your router management panel to get assigned IPv4 address or use the serial console to acquire it directly from the HiFive Unleashed (use `ip addr` command to print active network information). Finally you can SSH to the machine:

```
ssh -o PreferredAuthentications=password -o PubkeyAuthentication=no  
-o StrictHostKeyChecking=no -o "UserKnownHostsFile /dev/null"  
root@<IPv4>
```

Supported GPUs

CAICOS family of GPUs from AMD were supported from the beginning and thus are the most tested. In particular **Radeon HD 6450** is the most widely used and is highly recommended today. Other GPUs from the same family might also work. That could be: HD64xxM, HD7450, HD8450, R5 230, R5 235, R5 235X

This was extended to support newer AMD GPUs, but only POLARIS12 based RX 550 was tested so far.

Online Resizing of rootfs (Root File Partition)

It is highly advised to resize partitions offline (i.e. before booting the system). If you already booted the system and cannot do offline resizing then the following instructions should resize rootfs (root file partition) to full uSD capacity:

```
sgdisk -v /dev/mmcblk0  
sgdisk -e /dev/mmcblk0  
parted /dev/mmcblk0 resizepart 4 100%  
resize2fs /dev/mmcblk0p4  
sync
```

Contributions & Feedback

If you want to file issues, send patches and make feature/enhancement requests use [meta-sifive](#) or [freedom-u-sdk](#) repositories on GitHub.

You are also welcome to join [SiFive Forums](#) there we have [HiFive Unleashed](#) category for discussions.

Known Issues

1. Avoid overclocking SOC using CPUFreq if you are using HiFive Unleashed Expansion Board from Microsemi as this will hang the board. Hard reset will be required.
2. If Xfce4 desktop disk image is used with HiFive Unleashed Expansion Board and GPU then rebooting is required after the 1st boot.
3. OpenEmbedded Core (and thus meta-sifive) does not support eCryptFS or any other file system without long file names support. File systems must support filenames up to 200 characters in length.
4. BitBake requires UTF-8 based locale (e.g. `en_US.UTF-8`). You can choose any locale as long as it is UTF-8. This usually happens in containers (e.g. `ubuntu:18.04`). You can verify your locale by running `locale` command. On Ubuntu 18.04 you can change locale following these instructions:

```
apt update
apt install locales
locale-gen en_US.UTF-8
update-locale LC_ALL=en_US.UTF-8 LANG=en_US.UTF-8
export LANG="en_US.UTF-8"
locale
```

Releases 15



2020.11.00 You can change system default locale with `dpkg-reconfigure locales` command. Latest
14 days ago

[+ 14 releases](#)

Packages

No packages published

Contributors 13



[+ 2 contributors](#)