



# **SiFive HiFive Unleashed Getting Started Guide**

## **v1p3**

© SiFive, Inc.

# SiFive HiFive Unleashed Getting Started Guide

## Proprietary Notice

Copyright © 2021, SiFive Inc. All rights reserved.

HiFive Unleashed Getting Started Guide by SiFive, Inc. is licensed under Attribution-NonCommercial-NoDerivatives 4.0 International. To view a copy of this license, visit <http://creativecommons.org/licenses/by-nc-nd/4.0>

Information in this document is provided “as is,” with all faults.

SiFive expressly disclaims all warranties, representations, and conditions of any kind, whether express or implied, including, but not limited to, the implied warranties or conditions of merchantability, fitness for a particular purpose and non-infringement.

SiFive does not assume any liability rising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation indirect, incidental, special, exemplary, or consequential damages.

SiFive reserves the right to make changes without further notice to any products herein.

## Release Information

Version	Date	Changes
v1p3	March 3, 2021	<ul style="list-style-type: none"><li>• Added Creative Commons license</li></ul>
v1p2	February 18, 2020	<ul style="list-style-type: none"><li>• Updated software development flow</li></ul>
v1p1	April 06, 2018	<ul style="list-style-type: none"><li>• Updated introduction</li><li>• Added component diagrams</li><li>• Added links to schematics and design files</li><li>• Added software development flow</li></ul>
v1p0	March 09, 2018	<ul style="list-style-type: none"><li>• Initial Release</li></ul>

# Contents

<b>1</b>	<b>Introduction .....</b>	<b>3</b>
1.1	HiFive Unleashed Components .....	3
1.2	HiFive Unleashed Schematics.....	3
<b>2</b>	<b>Required Hardware .....</b>	<b>4</b>
<b>3</b>	<b>Optional Hardware .....</b>	<b>5</b>
<b>4</b>	<b>Board Setup .....</b>	<b>6</b>
<b>5</b>	<b>Boot and Run .....</b>	<b>11</b>
5.1	Connecting with ssh .....	11
5.2	Connecting with USB console .....	11
5.3	Knobs on the HiFive Unleashed .....	12
5.4	HiFive Unleashed Boot.....	12
<b>6</b>	<b>Firmware Update .....</b>	<b>14</b>
<b>7</b>	<b>Software Development Flow .....</b>	<b>15</b>
7.1	Supported Platforms .....	15
7.2	Software Development with the Freedom Unleashed SDK .....	15
7.2.1	Install Prerequisite Packages .....	15
7.2.2	Clone the freedom-u-sdk Repository .....	16
7.2.3	Build the System .....	16
7.2.4	Copy the System to an SD Card .....	16
<b>8</b>	<b>Connector Pinout.....</b>	<b>19</b>
8.1	FMC Connector .....	19
8.2	Low Speed I/O Expansion Connectors.....	24
8.3	MicroUSB Connector.....	25

---

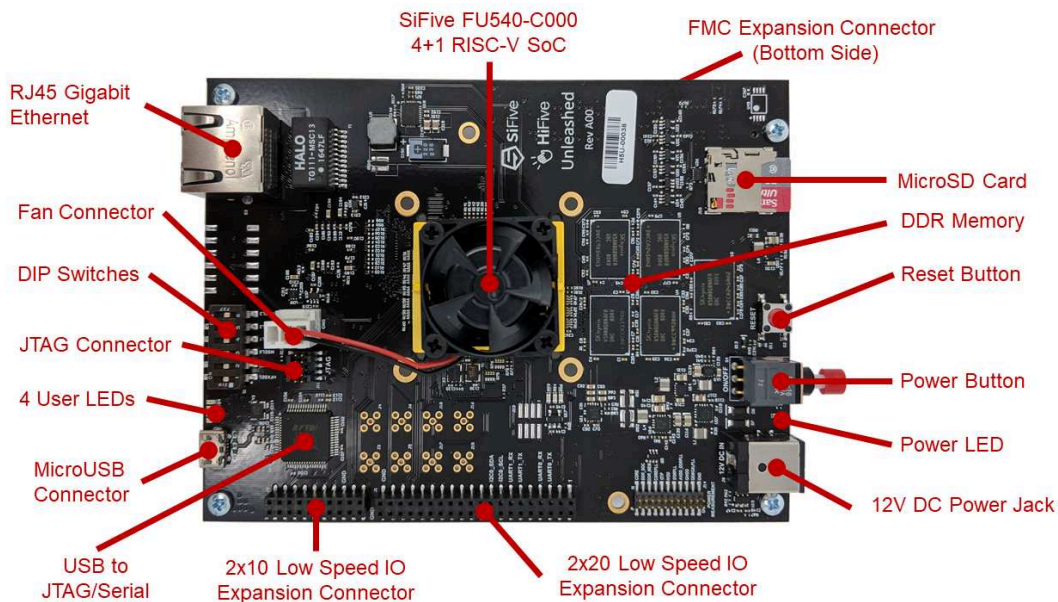
8.4	MicroSD Card Connector .....	25
8.5	JTAG Connector .....	25

# 1

## Introduction

HiFive Unleashed is a Linux development platform for SiFive's Freedom U540 SoC, the world's first 4+1 64-bit multi-core Linux-capable RISC-V SoC. The HiFive Unleashed has 8GB DDR4, 32MB QuadSPI Flash, a Gigabit Ethernet port, and a MicroSD card slot for more external storage. Additionally, the HiFive Unleashed supports adding new features, such as PCI Express (PCIe), through the FMC connector and low-speed I/O expansion connectors.

### 1.1 HiFive Unleashed Components



### 1.2 HiFive Unleashed Schematics

Schematics and design files for HiFive Unleashed are available at:  
<https://www.sifive.com/products/hifive-unleashed>

## 2

# Required Hardware

Using the HiFive Unleashed Development Kit requires the following hardware:

### **HiFive Unleashed Development Kit**

SiFive's HiFive Unleashed development kit is based around SiFive's FU540-C000 SoC. It can be purchased from Crowd Supply:

<http://www.crowdsupply.com/sifive/hifive-unleashed>

### **USB A to Micro-B Cable**

Any standard USB Type A Male to Micro-B Male cable can be used to interface with the HiFive Unleashed.

<http://store.digilentinc.com/usb-a-to-micro-b-cable>

### **12V DC Power Wall Adapter**

The HiFive Unleashed Development Kit requires an external 12V, 2A power supply. When not powering directly from the FMC expansion module, a 12V DC Wall Adapter with a 5.5 outer diameter, 2.5 mm center-positive barrel plug, must be plugged into DC power jack on the HiFive Unleashed. If powering directly from the FMC expansion module, then unplug the 12V DC power wall adapter from the development kit. Power should not be supplied from 2 different external sources.

# 3

## Optional Hardware

### **Ethernet Cable**

HiFive Unleashed Development Kit has an RJ45 Ethernet port that enables the FU540-C000 to connect to an 10/100/1000 Base-T network.

### **FMC Expansion Carrier Card**

Additional functionality can be added to the HiFive Unleashed Development Kit through the FMC connector. The FMC expansion carrier card typically has an FPGA to enable third-party developers to add new logic to the development kit.

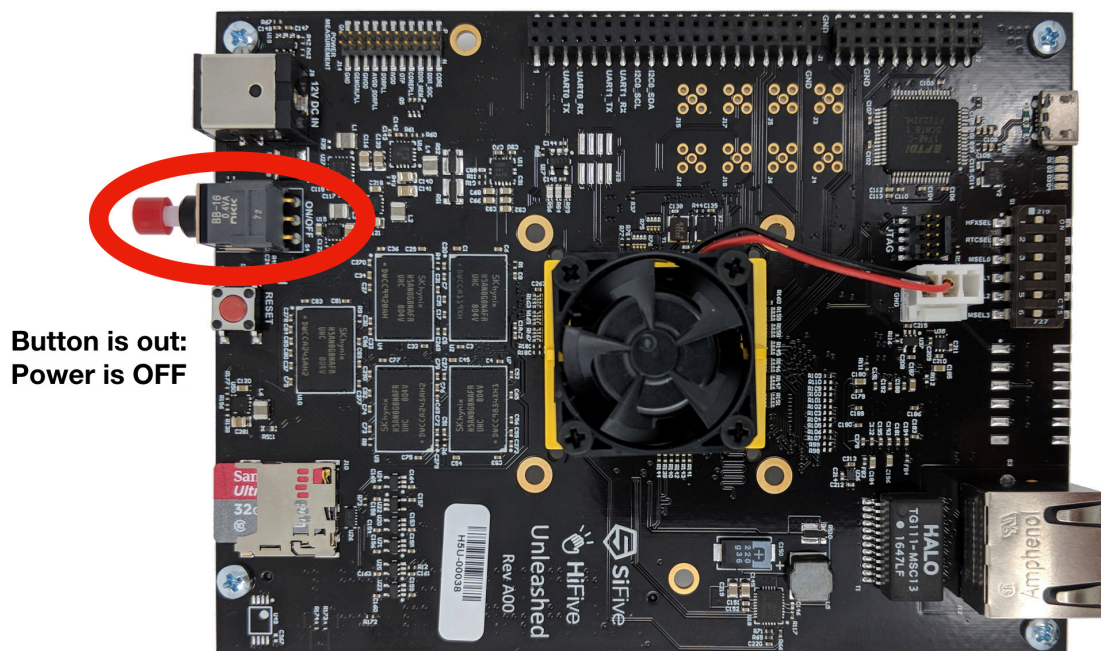
The HiFive Unleashed Development Kit can be directly powered from the FMC expansion carrier card. When powering from the FMC expansion carrier card, do not plug in the 12V DC adapter into the HiFive Unleashed.

## 4

# Board Setup

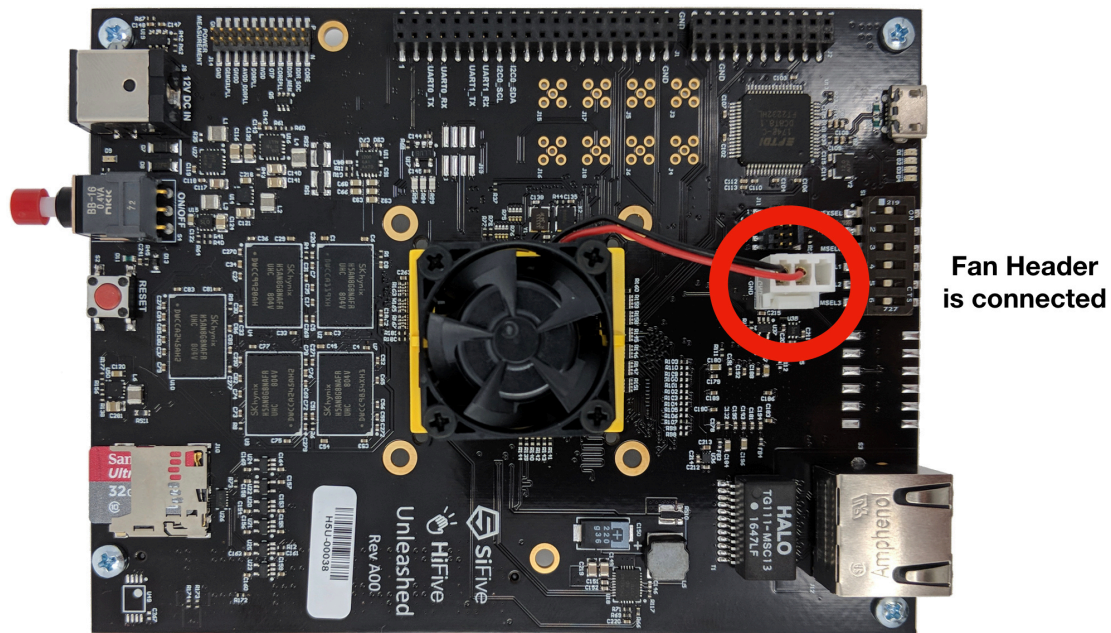
To prepare the board for use, follow these steps:

- Unplug the HiFive Unleashed and switch the power off (red button sticking out). The board is still powered even when the switch is off, so handle with care whenever power is applied.

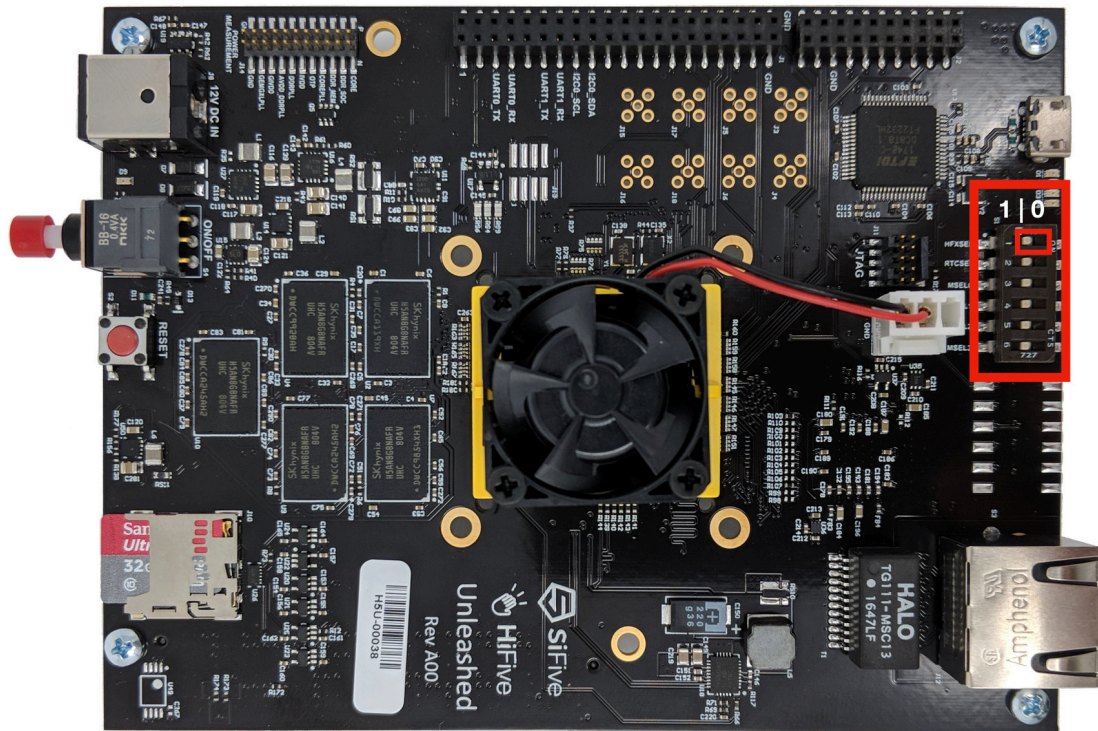




- Ensure the fan is plugged in.



- Set all pins in the DIP-switch block to the LEFT. The ON position=0; therefore, this sets MSEL to mode 1111. See the boot modes table in next section for more information on MSEL.

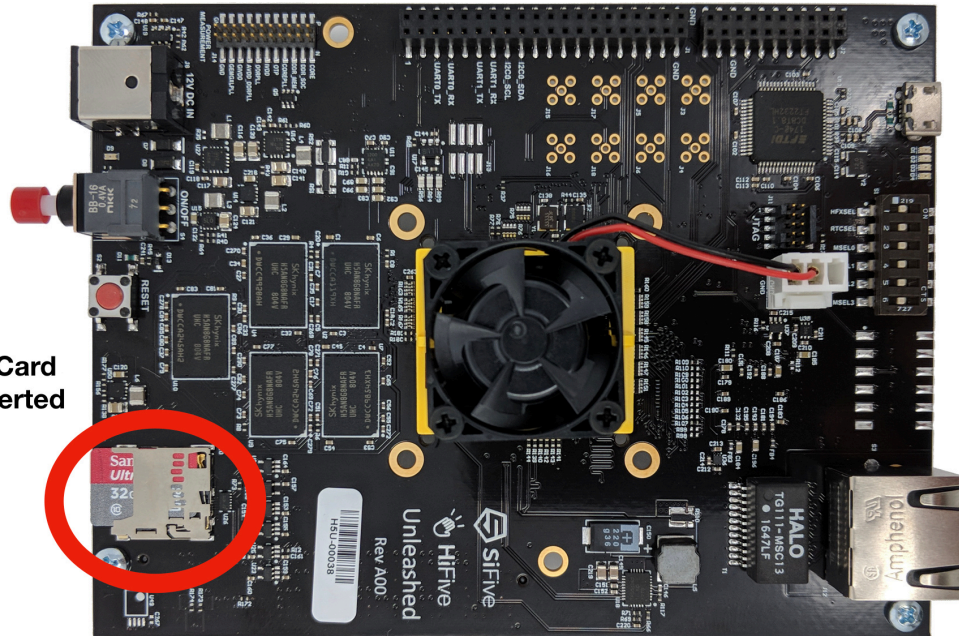


All DIP switches to 1:  
Default Boot Mode

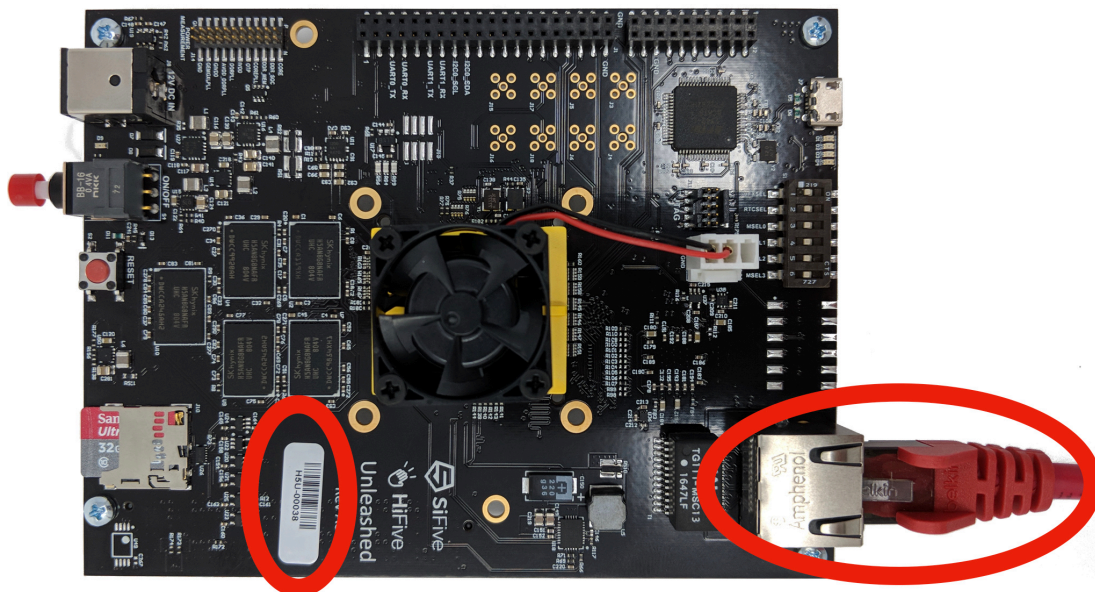


- Insert an SD-card programmed with bbl+linux. The github repository [sifive/freedom-u-sdk](https://github.com/sifive/freedom-u-sdk) produces an image suitable for writing to a partition with GUID type 2E54B353-1271-4842-806F-E436D6AF6985.

Boot SD Card  
is fully inserted

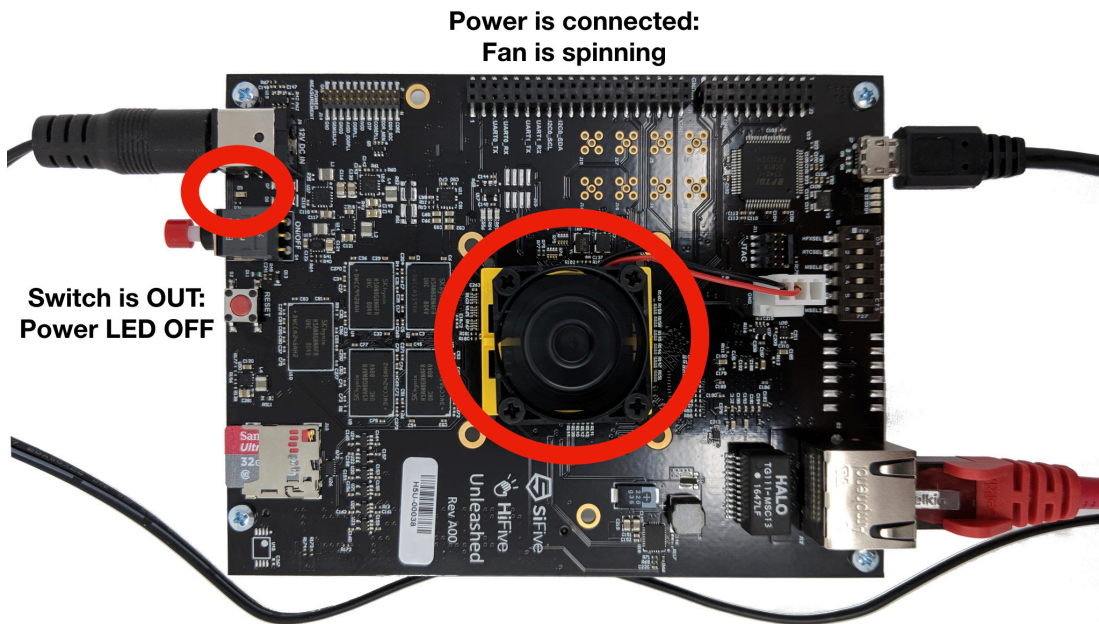


- If available, connect the board to a network switch. The board will run DHCP on boot and start an ssh server. The MAC address is 70:b3:d5:92:fx:xx, where x:xx is replaced by the board number converted to hexadecimal. For example, if the board is H5U-00063, then the last digits of the MAC address are 0:3f.

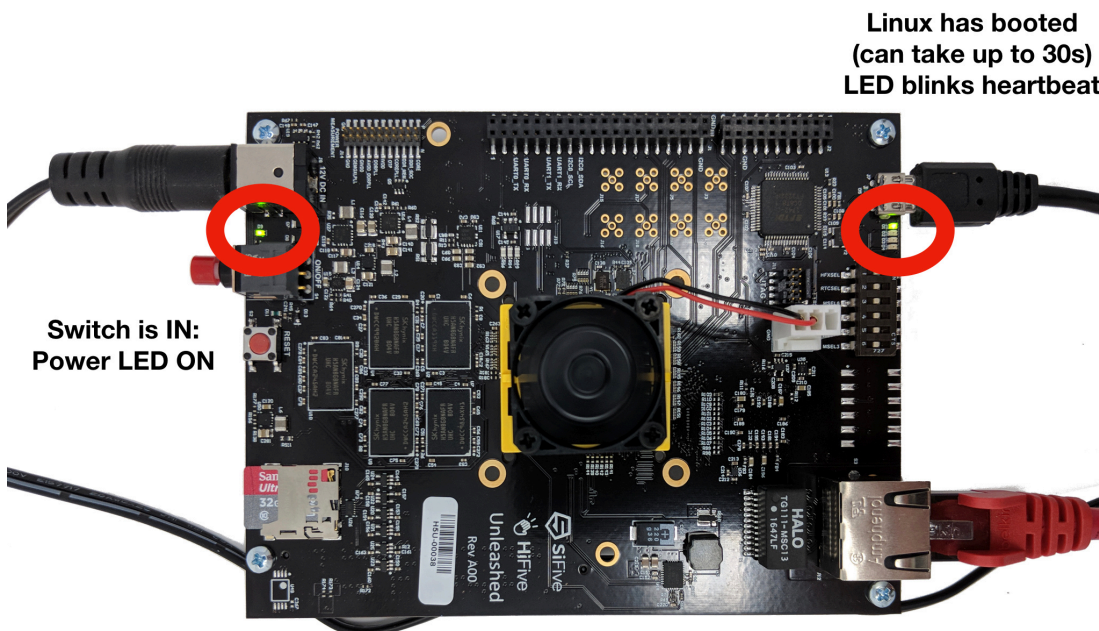


H5U-00038  
38 = 0x26

- If available, connect the board via USB to a developer machine. The USB connector has two serial interfaces. The first contains the linux console running at 115200Hz. The second provides JTAG suitable for use with openocd.
- Plug in the HiFive Unleashed; the fan should begin to spin.



- Turn on the HiFive Unleashed; after 30s a LED should begin to regularly blink a heartbeat.



# 5

## Boot and Run

Follow the steps from the previous section to power on the board. Once the heartbeat LED is pulsing regularly, you can connect to the board.

There are two ways to connect to the HiFive Unleashed: ssh and serial. The serial interface is slower and cannot receive files, so ssh is recommended.

### 5.1 Connecting with ssh

On power-on, the default freedom-u-sdk image obtains an IP address from DHCP and starts an ssh server. The MAC address of the board is 70:b3:d5:92:fx:xx, where x:xx is replaced by the board number converted to hexadecimal. Ask your network administrator which IP address your board has obtained. Add the following to your `.ssh/config`, filling in the IP address:

```
Host hifiveu
    HostName          xxx.yyy.zzz.aaa
    User              root
    StrictHostKeyChecking no
    UserKnownHostsFile /dev/null
```

To connect, run `ssh hifiveu`. The password is `sifive`.

### 5.2 Connecting with USB console

With a USB cable connected to the HiFive Unleashed, you can access the console. The console both shows the linux boot process and can be used to log in to the device.

From Mac OS, run: `sudo screen /dev/tty.usbserial-*01 115200`

From Linux, run: `sudo screen /dev/serial/by-path/*-port0 115200`

Hit enter a few times to see the login prompt. Username `root`, password `sifive`.

To quit screen, hit `Control-a, \, y`. If you do not quit properly, you can end up with multiple screens running at the same time. This manifests as lost characters both sent and received.

## 5.3 Knobs on the HiFive Unleashed

SD cards formatted by the `freedom-u-sdk` contains a second partition which can be used for persistent storage. You can access this storage by executing `mount /dev/mmcblk0p2 /mnt`. However, the linux `mmc_spi` driver is extremely slow, so this should not be used for large files.

Report core frequency: `echo $(cat /sys/devices/platform/soc/10000000.prci/rate)`

## 5.4 HiFive Unleashed Boot

The Freedom U500 supports several boot methods. The general flow follows multiple stages:

1. ROM0 (0x1004): Decode MSEL and jump to the address found in `0x1100 + 8*MSEL`. For `MSEL > 4`, the process proceeds to the next step.
2. ROM1 (0x1\_0000): Decode MSEL to determine which media contains the First Stage Boot Loader (FSBL). The media is expected to be formatted with a GPT partition table (even on SPI flash). The entire contents of the first partition with GUID type `5B193300-FC78-40CD-8002-E86C45580B47` are downloaded into the L2 sideband memory and execution transfers there. ROM1 contains a hard-coded DTB block passed in `a1` to the FSBL, which ignores it.
3. FSBL (0x800\_0000): Initialize core PPLs, DDR, and Ethernet PHY. Decode MSEL to determine which media contains the Berkeley Boot Loader (BBL). The media is expected to contain a GPT-formatted partition table. The contents of the first partition with GUID type `2E54B353-1271-4842-806F-E436D6AF6985` are downloaded into DDR memory and execution transfers there. The FSBL contains an embedded DTB describing both the Freedom U500 SoC and HiFive Unleashed board. The DTB is updated with the board MAC address and DDR capacity and passed to BBL in `a1`.
4. BBL (0x8000\_0000): Initialize the RISC-V supervisor binary interface (SBI). Jump to the embedded linux kernel payload, passing a redacted DTB in `a1`.
5. Linux (0xffff\_ffe0\_0000\_0000): Initializes all devices, starts DHCP + ssh.

Both ROM1 and the FSBL find the next boot loader stage based on the MSEL setting. All possible values are enumerated below. The three SPI interfaces on the Freedom U500 SoC can be used to download media either from SPI flash (using x4 data pins or x1) or an SD card, using the SPI protocol. ROM1 downloads the FSBL at 10MHz, while the FSBL uses 50MHz for SPI and 20MHz for SD.

**Table 1:** MSEL Boot Mode Straps

MSEL	FSBL	BBL	Purpose
0000	-	-	loops forever waiting for debugger
0001	-	-	jump directly to 0x2000_0000 (memory-mapped SPI0)
0010	-	-	jump directly to 0x3000_0000 (memory-mapped SPI1)
0011	-	-	jump directly to 0x4000_0000 (uncached ChipLink)
0100	-	-	jump directly to 0x6000_0000 (cached ChipLink)
0101	SPI0 x1	SPI0 x1	-
0110	SPI0 x4	SPI0 x4	Rescue image from flash (preprogrammed)
0111	SPI1 x4	SPI1 x4	-
1000	SPI1 SD	SPI1 SD	-
1001	SPI2 x1	SPI2 x1	-
1010	SPI0 x4	SPI1 SD	-
1011	SPI2 SD	SPI2 SD	Rescue image from SD card
1100	SPI1 x1	SPI2 SD	-
1101	SPI1 x4	SPI2 SD	-
1110	SPI0 x1	SPI2 SD	-
1111	SPI0 x4	SPI2 SD	Default boot mode



## 6

# Firmware Update

As new features are added, the HiFive Unleashed may receive periodic updates. These updates can include changes to the First Stage Boot Loader (FSBL), the Device Tree Blob (DTB), the recovery Linux kernel, or the default user kernel. To keep things simple, all of these are updated in lock-step. The newest firmware can always be found on the SiFive website.

Firmware updates for the HiFive Unleashed consist of two files:

1. `hifive-unleashed-a00-A.B-YYYY-MM-DD.gpt`, an image which should be written to the flash chip. This contains the FSBL, DTB, and recovery kernel.
2. `hifive-unleashed-a00-A.B-YYYY-MM-DD.bin`, an image which may optionally be written to the SD card. This contains the default user kernel.

To upgrade, follow these steps:

1. Switch the MSEL DIP switches to 0110
2. Power-on the board
3. Copy the two files to the board with `scp`
4. Execute: `flashcp -v hifive-unleashed-a00-A.B-YYYY-MM-DD.gpt /dev/mtd0` to upgrade the contents of the flash.
5. Optional: `cat hifive-unleashed-a00-A.B-YYYY-MM-DD.bin > /dev/mmcblk0p1; sync` to upgrade the contents of the SD card. This may be necessary if the DTB is changed in a manner incompatible with older kernels. However, this step will also destroy any customizations the user has made to the kernel.
6. Restore the MSEL DIP switches to 1111



# 7

## Software Development Flow

This document, as well as the binaries shipped with the HiFive Unleashed (and available on the SiFive website), concern an older, Buildroot-based version of freedom-u-sdk, which lives on the v1\_0 branch. This Buildroot version will be deprecated in the future.

A new, OpenEmbedded-based version of freedom-u-sdk is available at:

<http://github.com/sifive/freedom-u-sdk> (default branch)

To try the OpenEmbedded version, visit the freedom-u-sdk GitHub repo for instructions on building a disk image, installing it on a SD card, and booting it on your HiFive Unleashed. Note that some details in Chapters 4, 5, 6, and 7 of this document are specific to the Buildroot version.

### 7.1 Supported Platforms

This document assumes you are running on a modern Linux system. It has been tested on "bare metal" Linux systems and Linux systems running as a virtual machine. The process documented here was tested using Ubuntu 16.04.4, and this operating system is the only version which is guaranteed to run with the Buildroot-based version.

### 7.2 Software Development with the Freedom Unleashed SDK

#### 7.2.1 Install Prerequisite Packages

Before starting, use the apt-get command to install prerequisite packages:

```
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
```

### 7.2.2 Clone the freedom-u-sdk Repository

After installing the prerequisite packages, clone and initialize the `freedom-u-sdk` with these commands:

```
git clone https://github.com/sifive/freedom-u-sdk.git
cd freedom-u-sdk
git checkout v1_0
git submodule update --init --recursive
```

Note: this command can take up to thirty minutes or one hour to complete.

### 7.2.3 Build the System

Once the repository has been cloned and its sub-modules initialized, you are ready to build the system. Simply issue a `make` command from the `freedom-u-sdk` directory:

```
unset RISC_V
make
```

Note: this command can take several hours on a fast machine.

### 7.2.4 Copy the System to an SD Card

Assuming all prerequisite packages were installed and the `freedom-u-sdk` repository's sub-modules were successfully initialized, the `make` command from the previous step *should* complete with a message describing how to copy the boot image to an SD card.

Before executing this command, you will need to know the device name of your SD card. You will also need to delete Master Boot Record (MBR) partitions from the SD card and create a GUID Partition Table with no partitions defined.

Under current revisions of Ubuntu Linux, it is not uncommon for SD cards to be auto-mounted upon insertion if they contain a valid partition table and file system. You may use the `mount` command to determine if this is the case. They should be unmounted before proceeding.

For example purposes, we will assume the device name of your SD card is `/dev/exb`. (This is almost certainly *not* the name of an actual device; you will need to substitute the name of your actual SD card device for `/dev/exb` in the commands which follow.)

If your SD card was auto-mounted when you inserted it, you can likely find the device name by running the `mount` command. You may also find the name of the device by running the `dmesg` command and looking for log entries with the word "mounted".

If this is the first time you have installed a `freedom-u-sdk` image onto an SD card, it likely still has a legacy partition table in the Master Boot Record. Before proceeding, you must delete par-

titions in the legacy partition table and create a new GUID Partition Table (GPT). You only need to do this once, so if you have done this before, you should skip this step.

The transcript below shows the user using the `gdisk` program to delete partitions defined in the MBR partition table and creating a GPT (GUID Partition Table). Bolded text is user input. Before proceeding to the next step, ensure that you have followed a similar procedure. The next step will fail if an existing MBR partition table remains on the SD card.

```
$ sudo gdisk /dev/exb
GPT fdisk (gdisk) version 1.0.1

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!

Warning! Secondary partition table overlaps the last partition by
33 blocks!
You will need to delete this partition or resize it in another utility.

Command (? for help): p
Disk /dev/exb: 61440 sectors, 30.0 MiB
Logical sector size: 512 bytes
Disk identifier (GUID): E2725DE1-5C7D-47A4-AADB-1C87907A1E50
Partition table holds up to 128 entries
First usable sector is 34, last usable sector is 61406
Partitions will be aligned on 2048-sector boundaries
Total free space is 2014 sectors (1007.0 KiB)

Number  Start (sector)    End (sector)  Size      Code  Name
   1            2048             10239     4.0 MiB    8300   Linux filesystem
   2           10240             61439    25.0 MiB    8300   Linux filesystem

Command (? for help): d
Partition number (1-2): 2

Command (? for help): d
Using 1

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

Command (? for help): p
Disk /dev/exb: 61440 sectors, 30.0 MiB
Logical sector size: 512 bytes
Disk identifier (GUID): E1CE61B3-E5AD-422D-8974-A612BBEC9DE1
Partition table holds up to 128 entries
First usable sector is 34, last usable sector is 61406
Partitions will be aligned on 2048-sector boundaries
```

---

```
Total free space is 61373 sectors (30.0 MiB)

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

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/exb.
Warning: The kernel is still using the old partition table.
The new table will be used at the next reboot or after you
run partprobe(8) or kpartx(8)
The operation has completed successfully.
$
```

Once the MBR partition table and its partitions have been deleted, you can continue with the process of writing the system to the SD card:

```
sudo make DISK=/dev/exb format-boot-loader
```

This command is known to fail with slower USB SD card devices. It *should* succeed if repeated.

The SD card should now contain a bootable linux image. Refer to previous chapters for instructions in how to boot your HiFive Unleashed board.

If you encounter difficulties while following this procedure, please visit the SiFive Developer Forums ( <https://forums.sifive.com/> ). Other users may have encountered similar problems and posted usable solutions.

# 8

## Connector Pinout

### 8.1 FMC Connector

HiFive Unleashed has an VITA 57 FPGA Mezzanine Card (FMC) HPC connector (Samtec ASP-134602-01) to connect to an FPGA on an FMC carrier card (mating connector: Samtec ASP-134486-01). The communication interface between the HiFive Unleashed and the FPGA is ChipLink. The HiFive Unleashed can be powered directly from the FMC carrier card through the FMC connector.

**Table 2:** FMC Connector Pinout

Pin Name	Voltage Rail	Direction	FMC Pin Number
CHILINK_RX_CLK	1.8V	INPUT	G2
CHILINK_RX_RST	1.8V	INPUT	G37
CHILINK_RX_SEND	1.8V	INPUT	G36
CHILINK_RX_DAT0	1.8V	INPUT	D20
CHILINK_RX_DAT1	1.8V	INPUT	D21
CHILINK_RX_DAT2	1.8V	INPUT	C22
CHILINK_RX_DAT3	1.8V	INPUT	C23
CHILINK_RX_DAT4	1.8V	INPUT	H22
CHILINK_RX_DAT5	1.8V	INPUT	H23
CHILINK_RX_DAT6	1.8V	INPUT	G21
CHILINK_RX_DAT7	1.8V	INPUT	G22
CHILINK_RX_DAT8	1.8V	INPUT	H25

**Table 2:** FMC Connector Pinout

Pin Name	Voltage Rail	Direction	FMC Pin Number
CHIPLINK_RX_DAT9	1.8V	INPUT	H26
CHIPLINK_RX_DAT10	1.8V	INPUT	G24
CHIPLINK_RX_DAT11	1.8V	INPUT	G25
CHIPLINK_RX_DAT12	1.8V	INPUT	D23
CHIPLINK_RX_DAT13	1.8V	INPUT	D24
CHIPLINK_RX_DAT14	1.8V	INPUT	H28
CHIPLINK_RX_DAT15	1.8V	INPUT	H29
CHIPLINK_RX_DAT16	1.8V	INPUT	G27
CHIPLINK_RX_DAT17	1.8V	INPUT	G28
CHIPLINK_RX_DAT18	1.8V	INPUT	D26
CHIPLINK_RX_DAT19	1.8V	INPUT	D27
CHIPLINK_RX_DAT20	1.8V	INPUT	C26
CHIPLINK_RX_DAT21	1.8V	INPUT	C27
CHIPLINK_RX_DAT22	1.8V	INPUT	H31
CHIPLINK_RX_DAT23	1.8V	INPUT	H32
CHIPLINK_RX_DAT24	1.8V	INPUT	G30
CHIPLINK_RX_DAT25	1.8V	INPUT	G31
CHIPLINK_RX_DAT26	1.8V	INPUT	H34
CHIPLINK_RX_DAT27	1.8V	INPUT	H35
CHIPLINK_RX_DAT28	1.8V	INPUT	G33
CHIPLINK_RX_DAT29	1.8V	INPUT	G34
CHIPLINK_RX_DAT30	1.8V	INPUT	H37
CHIPLINK_RX_DAT31	1.8V	INPUT	H38
CHIPLINK_TX_CLK	1.8V	OUTPUT	H4
CHIPLINK_TX_RST	1.8V	OUTPUT	G19
CHIPLINK_TX_SEND	1.8V	OUTPUT	G18

**Table 2:** FMC Connector Pinout

Pin Name	Voltage Rail	Direction	FMC Pin Number
CHIPLINK_TX_DAT0	1.8V	OUTPUT	G6
CHIPLINK_TX_DAT1	1.8V	OUTPUT	G7
CHIPLINK_TX_DAT2	1.8V	OUTPUT	D8
CHIPLINK_TX_DAT3	1.8V	OUTPUT	D9
CHIPLINK_TX_DAT4	1.8V	OUTPUT	H7
CHIPLINK_TX_DAT5	1.8V	OUTPUT	H8
CHIPLINK_TX_DAT6	1.8V	OUTPUT	G9
CHIPLINK_TX_DAT7	1.8V	OUTPUT	G10
CHIPLINK_TX_DAT8	1.8V	OUTPUT	H10
CHIPLINK_TX_DAT9	1.8V	OUTPUT	H11
CHIPLINK_TX_DAT10	1.8V	OUTPUT	D11
CHIPLINK_TX_DAT11	1.8V	OUTPUT	D12
CHIPLINK_TX_DAT12	1.8V	OUTPUT	C10
CHIPLINK_TX_DAT13	1.8V	OUTPUT	C11
CHIPLINK_TX_DAT14	1.8V	OUTPUT	H13
CHIPLINK_TX_DAT15	1.8V	OUTPUT	H14
CHIPLINK_TX_DAT16	1.8V	OUTPUT	G12
CHIPLINK_TX_DAT17	1.8V	OUTPUT	G13
CHIPLINK_TX_DAT18	1.8V	OUTPUT	D14
CHIPLINK_TX_DAT19	1.8V	OUTPUT	D15
CHIPLINK_TX_DAT20	1.8V	OUTPUT	C14
CHIPLINK_TX_DAT21	1.8V	OUTPUT	C15
CHIPLINK_TX_DAT22	1.8V	OUTPUT	H16
CHIPLINK_TX_DAT23	1.8V	OUTPUT	H17
CHIPLINK_TX_DAT24	1.8V	OUTPUT	G15
CHIPLINK_TX_DAT25	1.8V	OUTPUT	G16

**Table 2:** FMC Connector Pinout

Pin Name	Voltage Rail	Direction	FMC Pin Number
CHIPLINK_TX_DAT26	1.8V	OUTPUT	D17
CHIPLINK_TX_DAT27	1.8V	OUTPUT	D18
CHIPLINK_TX_DAT28	1.8V	OUTPUT	C18
CHIPLINK_TX_DAT29	1.8V	OUTPUT	C19
CHIPLINK_TX_DAT30	1.8V	OUTPUT	H19
CHIPLINK_TX_DAT31	1.8V	OUTPUT	H20
RESET_N	1.8V	Open Drain	H5
VDD_DCIN	12V	Voltage Input	C35, C37
VDD_1V8	1.8V	Voltage Output	H1
PG_M2C	Open Drain	Open Drain	F1
PRSNT_M2C_L	GND	Open Drain	H2
GND	GND	GND	A1, A4, A5, A8, A9, A12, A13, A16, A17, A20, A21, A24, A25, A28, A29, A32, A33, A36, A37, A40, B2, B3, B6, B7, B10, B11, B14, B15, B18, B19, B22, B23, B26, B27, B30, B31, B34, B35, B38, B39, C1, C4, C5, C8, C9, C12, C13, C16, C17, C20, C21, C24, C25, C28, C29, C32, C33, C36, C38, C40, D2, D3, D6, D7, D10, D13, D16, D19, D22, D25, D28, D37, D39, E1, E4, E5, E8, E11, E14, E17, E20, E23, E26, E29, E32, E35, E38, E40, F2, F3, F6, F9, F12, F15, F18, F21, F24, F27, F30, F33, F36, F39, G1, G4, G5, G8, G11, G14, G17, G20, G23, G26, G29, G32, G35, G38, G40, H3, H6, H9, H12, H15, H18, H21, H24, H27, H30, H33, H36, H39, J1, J4, J5, J8, J11, J14, J17, J20, J23, J26, J29, J32, J35, J38, J40, K2, K3, K6, K9, K12,



**Table 2:** FMC Connector Pinout

Pin Name	Voltage Rail	Direction	FMC Pin Number
			K15, K18, K21, K24, K27, K30, K33, K36, K39

### Pin and Signal Description

CHIPLINK\_RX\_CLK: ChipLink Receive Clock from the FMC carrier card to HiFive Unleashed.

CHIPLINK\_RX\_RST: ChipLink Receive Reset from the FMC carrier card to HiFive Unleashed.

CHIPLINK\_RX\_SEND: ChipLink Receive Send strobe from the FMC carrier card to HiFive Unleashed.

CHIPLINK\_RX\_DATA[31:0]: ChipLink Receive Data bus from the FMC carrier card to HiFive Unleashed.

CHIPLINK\_TX\_CLK: ChipLink Transmit Clock from HiFive Unleashed to the FMC carrier card.

CHIPLINK\_TX\_RST: ChipLink Transmit Reset from HiFive Unleashed to the FMC carrier card.

CHIPLINK\_TX\_SEND: ChipLink Transmit Send strobe from HiFive Unleashed to the FMC carrier card.

CHIPLINK\_TX\_DATA[31:0]: ChipLink Transmit Data bus from HiFive Unleashed to the FMC carrier card.

RESET\_N: Bidirectional, Open Drain active-low system-level reset. Either the HiFive Unleashed or the FMC carrier card can initiate a system level reset by pulling this pin low. The pull up resistor is on the HiFive Unleashed.

VDD\_DCIN: 12V voltage rail from the FMC carrier card to HiFive Unleashed.

VDD\_1V8: 1.8V I/O voltage from HiFive Unleashed to the FMC carrier card.

PG\_M2C: Open-drain, active high power good signal to indicate that HiFiveUnleashed has powered up correctly. The pull up resistor is on the FMC carrier card.

PRSNT\_M2C\_L: Active-low signal to indicate to the FMC carrier card that HiFive Unleashed is present. The pull up resistor is on the FMC carrier card.

All other pins are No Connect on the FMC connector.

## 8.2 Low Speed I/O Expansion Connectors

There are two low speed I/O expansion connectors, a 2mm pitch 2x20 and a 2mm pitch 2x10 female headers.

The 2x20 connector follows the 96Boards EE specification expansion connector pinout. All I/O voltage are at 1.8V.

**Table 3:** 2x20 Low Speed I/O Expansion Connector Pinout

Net Name	Pin Name	#	#	Pin Name	Net Name
GND	GND	1	2	GND	GND
No Connect	UART0_CTS	3	4	PWR_BTN_N	PWR_BTN_N
UART0_TX	UART0_TXD	5	6	RST_BTN_N	RESET_N
UART0_RX	UART0_RXD	7	8	SPI0_SCLK	QSPI1_SCK
No Connect	UART0_RTS	9	10	SPI0_DIN	QSPI1_DQ1
UART1_TX	UART1_TXD	11	12	SPI0_CS	QSPI1_CS0
UART1_RX	UART1_RXD	13	14	SPI0_DOUT	QSPI1_DQ0
I2C0_SCL	I2C0_SCL	15	16	PCM_FS	No Connect
I2C0_SDA	I2C0_SDA	17	18	PCM_CLK	No Connect
No Connect	I2C1_SCL	19	20	PCM_DO	No Connect
No Connect	I2C1_SDA	21	22	PCM_DI	No Connect
GPIO0	GPIO-A	23	24	GPIO-B	GPIO1
GPIO2	GPIO-C	25	26	GPIO-D	GPIO3
GPIO4	GPIO-E	27	28	GPIO-F	GPIO5
GPIO6	GPIO-G	29	30	GPIO-H	GPIO7
GPIO8	GPIO-I	31	32	GPIO-J	GPIO9
GPIO15	GPIO-K	33	34	GPIO-L	No Connect
VDD_1V8	+1V8	35	36	SYS_DCIN	VDD_DCIN
VDD_5V0	+5V	37	38	SYS_DCIN	VDD_DCIN
GND	GND	39	40	GND	GND

The 2x10 connector has additional low speed I/Os. All I/O voltage are at 1.8V.

**Table 4:** 2x10 Low Speed I/O Expansion Connector Pinout

Net Name	Direction	#	#	Direction	Net Name
PWM0_0	OUTPUT	1	2	OUTPUT	PWM0_2
PWM0_1	OUTPUT	3	4	OUTPUT	PWM0_3
GEMGXL_RST (Reset to Ethernet PHY)	OUTPUT	5	6	BIDIRECTIONAL	QSPI1_DQ2
FASTLINK_FAIL (Output from Ethernet PHY)	OUTPUT	7	8	BIDIRECTIONAL	QSPI1_DQ3
EN_VDD_SD (Enable Power to SD Card)	OUTPUT	9	10	OUTPUT	QSPI1_CS1
SD_CD (SD Card Detect)	OUTPUT	11	12	OUTPUT	QSPI1_CS2
(NC)	NC	13	14	OUTPUT	QSPI1_CS3
PWM1_0	OUTPUT	15	16	GND	GND
PWM1_1	OUTPUT	17	18	OUTPUT	PWM1_2
GND	GND	19	20	OUTPUT	PWM1_3

## 8.3 MicroUSB Connector

The MicroUSB connector provides a connection to UART0 and JTAG from a Host PC. If needed, drivers for the FTDI FT2232H can be found at:

<http://www.ftdichip.com/FTDrivers.htm>

## 8.4 MicroSD Card Connector

The MicroSD card connector enables adding more storage with an external MicroSD card. The connector is accessible from the Freedom U540 through SPI0.

## 8.5 JTAG Connector

The JTAG connector is a 1.27 mm pitch 2x5 male header (SAMTEC FTSH-105 or equivalent) that is compliant to the RISC-V External Debug Support V0.13 specification.

More information about the RISC-V External Debug can be found at:

<https://www.sifive.com/documentation/risc-v/risc-v-external-debug-support>

**Table 5:** JTAG Connector Pinout

Pin Name	#	#	Pin Name
TCK	1	2	GND
TDO	3	4	VCC
TMS	5	6	SRSTn
(NC)	7	8	(TRSTn/NC)
TDI	9	10	GND