SiFive HiFive1 Bluetooth Module Owner’s Manual

SiFive HiFive1 Bluetooth Module Owner’s Manual

Contents

1 Proprietary Notice
1.1 Release Information
2 HiFive1 Diagram
3 Required Hardware
4 Optional Hardware
5 Board Setup
6 Boot and Run
7 Software Development Flow
8 HiFive1 Additional Notes
9 HiFive 1 Pinout
10 For More Information
11 Documents / Resources
12 Related Posts

Proprietary Notice

Release Information

Version	Date	Changes
1.0.5	March 25, 2021	Added Creative Commons license
1.0.4	June 21, 2017	Added info on Freedom Studio and recompiled binaries
1.0.3	April 11, 2017	Corrected OTP Contents, some helpful hints for using screen
1.0.2	Jan 3, 2017	Added RGB LED Pin out and corrections to udev rules
1.0.1	Dec 21, 2016	Corrections to Software Development Flow
1.0.0	Dec 20, 2016	First Release

HiFive1 Diagram

HiFive1 Components

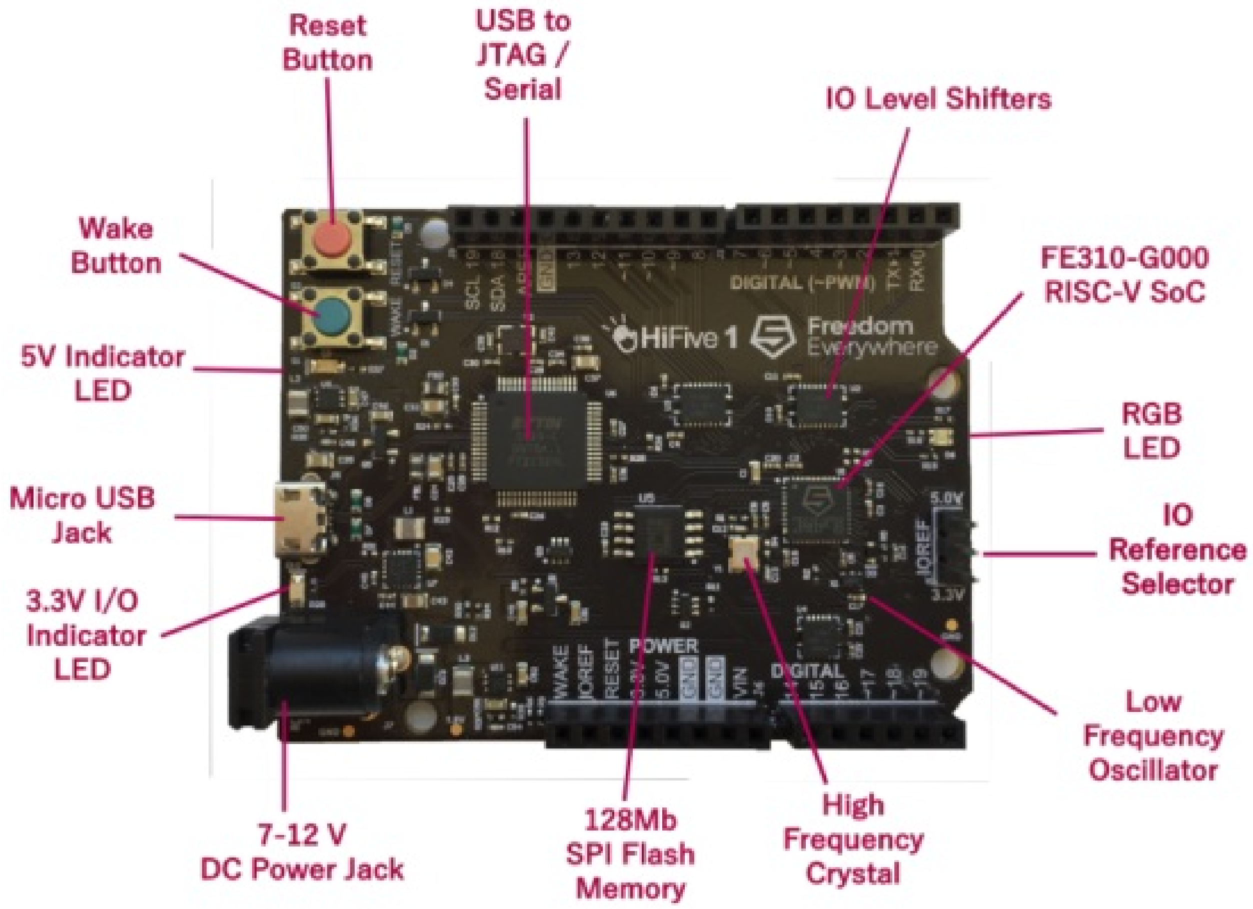


Figure 1.1: HiFive1 Board

Figure 1.1 shows the HiFive1 with the parts which are described in this document

HiFive1 Schematics

Schematics and design files for the HiFive1 are available at http:/dev.sifive.com/hifive1

Required Hardware

Using the HiFive1 Dev Kit requires the following hardware:

HiFive1 Dev Kit
SiFive's HiFive1 development board is based around the FE310-G000 chip, an SoC based around a RISC-V RV32IMAC core. It can be purchased from Crowd Supply:
http://www.crowdsupply.com/sifive/hifive1

USB A to Micro-B Cable

Any standard USB Type A Male to Micro-B Male cable can be used to interface with the HiFive1.
http://store.digilentinc.com/usb-a-to-micro-b-cable/

Optional Hardware

External Power Supply
While the USB A to Micro-B cable is necessary for programming and communicating with the HiFive1, the board can run off of an external 7-12V DC supply as well, or a USB power supply or battery pack.

IOREF Jumper
Your HiFive1 comes with a single jumper for selecting the IO Reference voltage. If you need to replace it, these can be purchased from DigiKey:**http://www.digikey.com/product-detail/en/sullins-connector-solutions/SPC02SYAN/ S9001-ND/76375**

Compatible Shields
Shields are devices which are designed to fit on the I/O headers on devices which match the Arduino form factor, to provide additional functionality. The shield manufacturer usually provides libraries designed to work with the Arduino IDE.

We have tested the following shields with the HiFive1 and ported their libraries

- dafruit's Resistive Touchscreen & LED Display:
  https://www.adafruit.com/products/1651
  Library: http://github.com/sifive/Adafruit_ILI9341
- Adafruit's BLE SPI Friend:
  https://www.adafruit.com/products/2633
  Library: http://github.com/sifive/Adafruit_BluefruitLE_nRF51

Generally, shields which communicate with SPI, UART, and digital I/Os should be easy to use with the HiFive1, but their supporting library may need minor tweaks to recognize the HiFive1.

Shields with Analog input requirements will need an adapter as the FE310-G000 does not include analog components.

Board Setup

Installing the IOREF Jumper

Your HiFive1 will come with a jumper installed on J1, to select the IO Reference Level. Depending on the shields you want to drive, select 3.3V or 5.V using the jumper. You must install the jumper on one side or the other in order to drive or read any signals on the IO headers.

Connecting the USB Interface

Connect the USB Type A to Micro-B cable between the USB port of the HiFive1 and the host machine. This provides UART console access to the HiFive1 as well as a 5V power source for the board. This is also how software will be programmed and debugged on the HiFive1.

Connect the other end of the cable to your host machine. You will see the green power indication LEDs D10 and D9 light up. This means that both the main 5V supply is on, as well as the 3.3V "mostly off" supply.

Boot and Run

The HiFive1 comes programmed with a simple boot loader and a demo software program which prints to the UART and cycles through the RGB LED in a rainbow pattern. You can respond on the UART to indicate that the LEDs are changing and get a "PASS" message.

This program will be overwritten in the SPI Flash when you program new software into the board with the SDK, but the boot loader code will not be modified.

Using a terminal emulator such as GNU screen on Linux, open a console connection from the host computer to the HiFive1.

Set the following parameters:

Speed	115200
Parity	None
Data bits	8
Stop bits	1
Hardware Flow	None

For example, on Linux using GNU Screen:

```
sudo screen /dev/ttyUSB1 115200
```

You can use Ctrl-a k to “kill” (exit) the running screen session.

Depending on your setup, you may need additional drivers or permissions to communicate over the USB port.

If you are running on Ubuntu-style Linux, the below is an example of steps you may need to follow to access your dev kit without sudo permissions:

1. With your board’s debug interface connected, make sure your device shows up with the lsusb command: > lsusb…
   Bus XXX Device XXX: ID 0403:6010 Future Technology Devices
   International, Ltd FT2232C Dual USB-UART/FIFO IC
2. Set the udev rules to allow the device to be accessed by the plugdev group:
   > sudo vi /etc/udev/rules.d/99-openocd.rules Add the following lines and save the file (if they are not already there): # These are for the HiFive1 Board SUBSYSTEM==”usb”, ATTR{idVendor}==”0403″, ATTR{idProduct}==”6010″, MODE=”664″, GROUP=”plugdev” SUBSYSTEM==”tty”, ATTRS{idVendor}==”0403″, ATTRS{idProduct}==”6010″, MODE=”664″, GROUP=”plugdev” # These are for the Olimex Debugger for use with E310 Arty Dev Kit SUBSYSTEM==”usb”, ATTR{idVendor}==”15ba”, ATTR{idProduct}==”002a”, MODE=”664″, GROUP=”plugdev” SUBSYSTEM==”tty”, ATTRS{idVendor}==”15ba”, ATTRS{idProduct}==”002a”, MODE=”664″, GROUP=”plugdev”
3. See if your board shows up as a serial device belonging to the plugdev group:
   > ls /dev/ttyUSB*
   /dev/ttyUSB0 /dev/ttyUSB1
   (If you have other serial devices or multiple boards attached, you may have more devices listed). For serial

communication with the UART, you will always want to select the higher number of the pair, in this example /dev/ttyUSB1.

> ls -l /dev/ttyUSB1

crw-rw-r– 1 root plugdev 188, 1 Nov 28 12:53 /dev/ttyUSB1

4. Add yourself to the plugdev group. You can use the whoami command to determine your user name.

   > whoami your user name > sudo usermod -a -G plugdev your user name

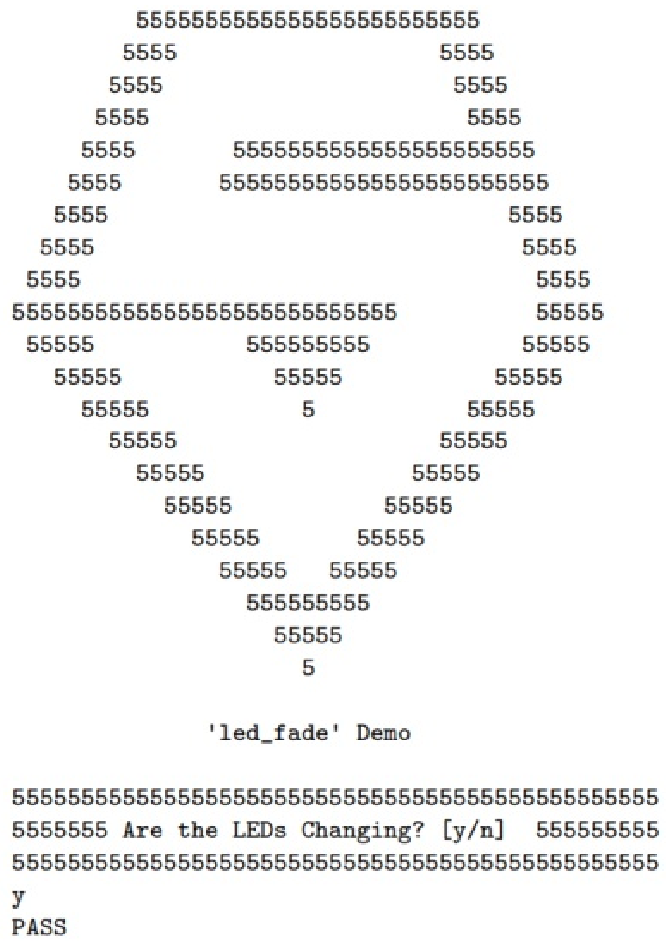
5. Log out and log back in, then check that you're now a member of the plugdev group:

   > groups

   … plugdev …

Now you should be able to access the serial (UART) and debug interface without sudo permissions.

Terminal Log
If you have your serial setup correctly, this is what you will see on your terminal (you may need to hit the 'Reset' button to restart the program):



```
         5555555555555555555555555
        5555                   5555
       5555                     5555
      5555                       5555
     5555       5555555555555555555555
    5555       555555555555555555555555
   5555                             5555
  5555                               5555
 5555                                 5555
5555555555555555555555555555          55555
 55555           555555555           55555
   55555           55555           55555
     55555           5           55555
       55555                   55555
         55555               55555
           55555           55555
             55555       55555
               55555   55555
                 555555555
                   55555
                     5

              'led_fade' Demo

55555555555555555555555555555555555555555555555
5555555 Are the LEDs Changing? [y/n]  555555555
55555555555555555555555555555555555555555555555
y
PASS
```

Software Development Flow

The HiFive1's boot code contains a jump to the external SPI Flash on the board, at address 0x20400000. You can change the program which the dev kit runs by using the debug/programming interface to flash a new compiled program into the SPI Flash. SiFive supports several methods of obtaining the software development tool chain. Freedom Studio (6.2) is an Eclipse-Based IDE which bundles everything you need into one download. You can also compile the source yourself and run command line tools with the Freedom E SDK (6.3). Finally, there is a board package for use with the Arduino IDE (6.4). These different development versions will all install the same set of tools, but the versions, install paths and associated software libraries and examples are different for each.

Supported Platforms
Freedom Studio is supported on Linux, macOS, and Windows.
Building the tools manually and the Arduino IDE are currently supported only on Linux and macOS.

Software Development Using Freedom Studio IDE
SiFive recommends software development for the HiFive1 with the Eclipse-based Freedom Studio IDE. Freedom Studio is supported for Windows, macOS, and Linux. When using this method, the recompiled tools and drivers are automatically installed, you do not need to download or install it separately to get tools and example code.

You can obtain Freedom Studio from the SiFive website: https://www.sifive.com/product/tools/

More information on how to use it can be found in the Freedom Studio Manual:
https://www.sifive.com/documentation/tools/freedom-studio-manual

Software Development Using Command Line Tools
Obtaining the Freedom E SDK Toolchain
The Freedom E Software Development Kit provides everything required to compile, customize, and debug C, C++ and/or RISC-V assembly programs: GCC 7.1.0 cross-compilation tool chain, RISC-V enabled GDB and OpenOCD, etc. The SDK also includes example code projects. The tool chain can either be compiled from source in the SDK, or downloaded from SiFive's website as recompiled binary packages.

Obtaining the Freedom E SDK Recompiled Binaries Recompiled binaries can be obtained from SiFive's website: https://www.sifive.com/product/tools/

To get example projects without downloading all of the tool chain sources, you can do a shallow clone of the Freedom E SDK:

```
git clone https://github.com/sifive/freedom-e-sdk.git
```

Compiling the Freedom E SDK Tools from Source
This step is not necessary if you have downloaded the recompiled binaries as described in the previous section.

To clone the Freedom E SDK git repository:

```
git clone –recursive https://github.com/sifive/freedom-e-sdk.git
```

Install all the necessary packages described in the repository's README.md file.

To build the software toolchain:

```
cd freedom-e-sdk
make tools BOARD=freedom-e300-hifive1
```

To keep your software tool chain up to date with the upstream repository:

```
cd freedom-e-sdk
git pull origin master
git submodule update –init –recursive
make tools BOARD=freedom-e300-hifive1
```

Compiling Software Programs
To build a C program that will be loaded by the debugger/programmer into the SPI Flash, use the Freedom E SDK

to compile. Examples are provided in the software/ directory. To build the program:

```
cd freedom-e-sdk
make software PROGRAM=demo gpio BOARD=freedom-e300-hifive1
```

To compile the Dhrystone benchmark instead:

```
cd freedom-e-sdk
make software PROGRAM=dhrystone BOARD=freedom-e300-hifive1
```

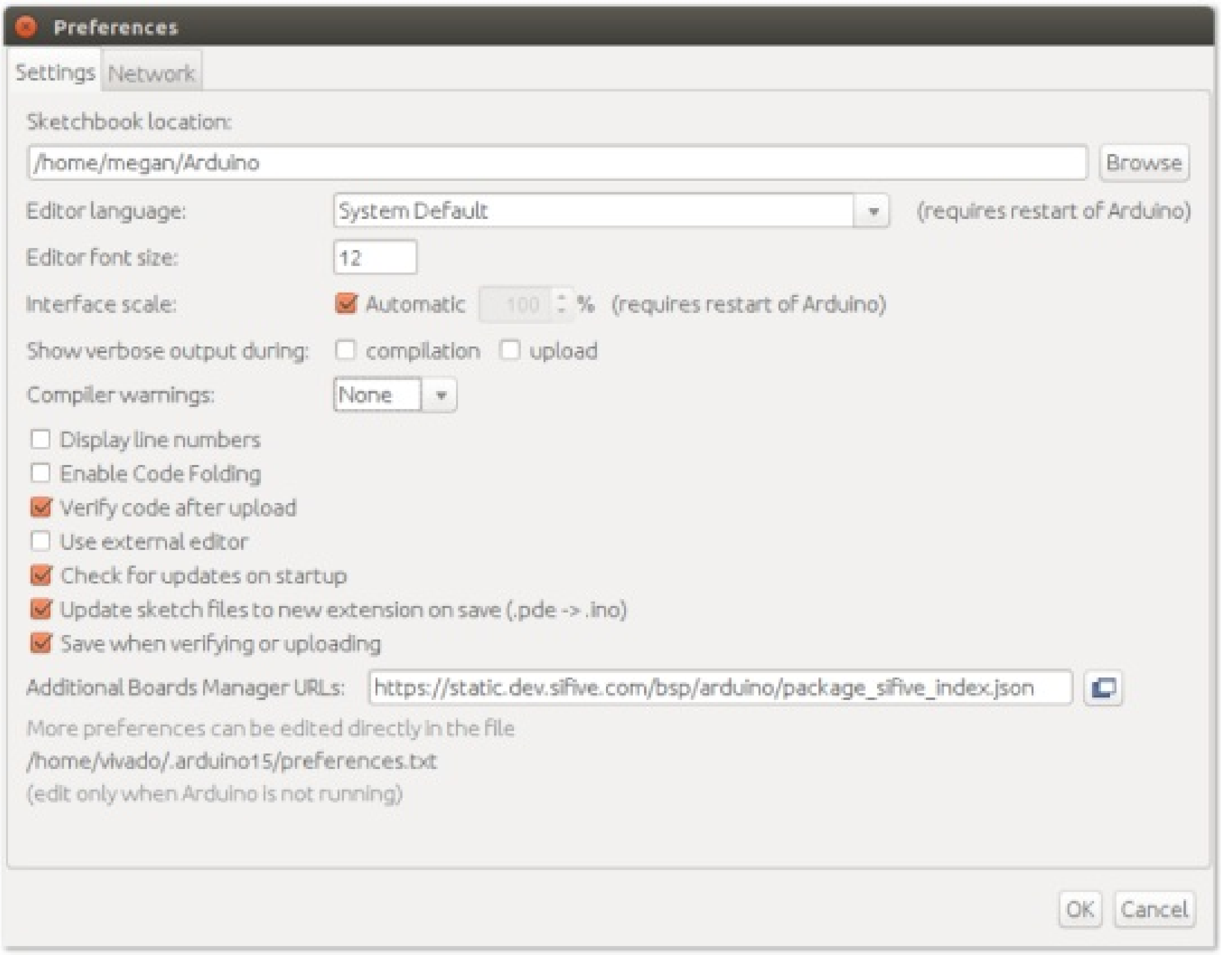
Uploading Software Programs
To upload the program to the SPI flash, connect the board's debug interface as described in Then execute:
cd freedom-e-sdk make upload PROGRAM= BOARD=freedom-e300-hifive1

Debugging Running Programs
To debug your program with GDB, connect your board and launch OpenOCD in one terminal window:
cd freedom-e-sdk make run openocd BOARD=freedom-e300-hifive1
In a second terminal window, launch the debugger, which will automatically connect to the running OpenOCD target:
cd freedom-e-sdk make run gdb PROGRAM= BOARD=freedom-e300-hifive1
This will automatically launch OpenOCD and GDB, connect to the board, and halt the currently running program. You can step through the running program with step, or load the new program using load. The usual suite of GDB commands are available to set breakpoints, examine and modify memory, continue execution, etc.

Software Development Using the Arduino IDE
SiFive also supports software development for the HiFive1 with the Arduino IDE. When using this method, the Freedom E SDK is automatically installed, so you do not need to install it separately. Follow these steps:

Installing the HiFive1 Board Package

1. Download and install the Arduino IDE, following the instructions at
   https://www.arduino.cc/en/Guide/HomePage
2. Launch the Arduino IDE
3. Navigate to File → Preferences and add the SiFive additional Board Manager URL as shown in **Figure 6.1:**
   http://static.dev.sifive.com/bsp/arduino/package_sifive_index.json
4. Add the SiFive development kit boards using the Board Manager:
   Tools → Board → Board Manager.
   Search for "SiFive" and click Install to download and install the package. Restart your Arduino IDE, then find the HiFive1 under Tools → Board, as shown in **Figure 6.2**.



Sketchbook location: /home/megan/Arduino

Editor language: System Default (requires restart of Arduino)

Editor font size: 12

Interface scale: ☑ Automatic 100 % (requires restart of Arduino)

Show verbose output during: ☐ compilation ☐ upload

Compiler warnings: None

☐ Display line numbers
☐ Enable Code Folding
☑ Verify code after upload
☐ Use external editor
☑ Check for updates on startup
☑ Update sketch files to new extension on save (.pde -> .ino)
☑ Save when verifying or uploading

Additional Boards Manager URLs: https://static.dev.sifive.com/bsp/arduino/package_sifive_index.json

More preferences can be edited directly in the file
/home/vivado/.arduino15/preferences.txt
(edit only when Arduino is not running)

Figure 6.1: Adding the SiFive Dev Kits Board URL to your Arduino IDE

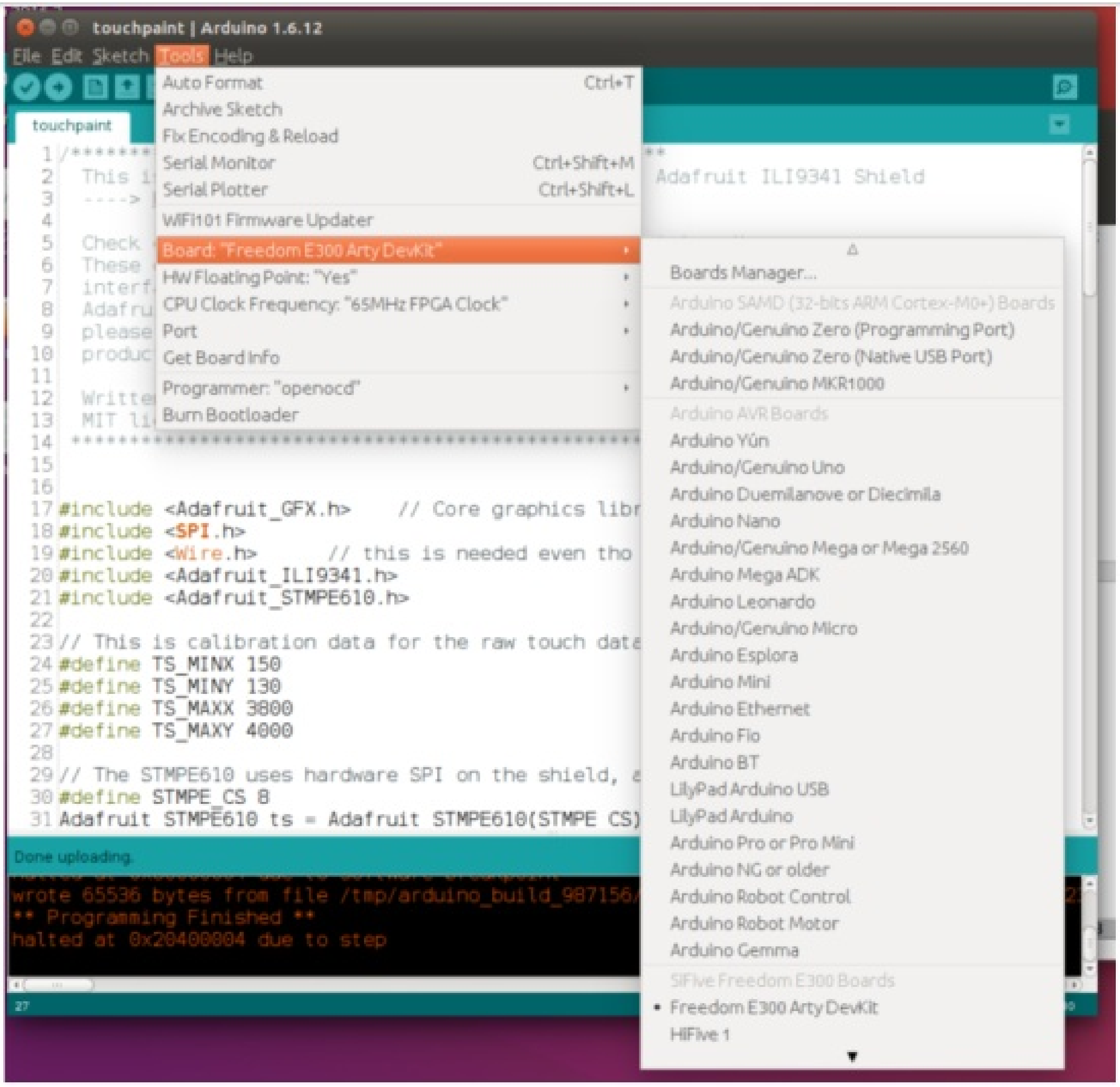
5. Select SiFive OpenOCD as the Programmer from the Tools menu.
6. To compile and upload a simple example program, select

   File → Examples → Basics → Blink

   Click the “upload” button in the Arduino IDE, your program will compile and upload to your Dev Kit, and the green LED will blink.

Open Source Board Support Package Code

The code installed with the Board package is open-source, and available to view or download at:
http://github.com/sifive/cinco



Board: "Freedom E300 Arty DevKit"
HW Floating Point: "Yes"
CPU Clock Frequency: "65MHz FPGA Clock"
Programmer: "openocd"

```
/*******                                              **
 This i                                                Adafruit ILI9341 Shield
 ---->

 Check
 These
 interf
 Adafru
 please
 produc

 Writte
 MIT li
 **************************************************


#include <Adafruit_GFX.h>    // Core graphics libr
#include <SPI.h>
#include <Wire.h>      // this is needed even tho
#include <Adafruit_ILI9341.h>
#include <Adafruit_STMPE610.h>

// This is calibration data for the raw touch data
#define TS_MINX 150
#define TS_MINY 130
#define TS_MAXX 3800
#define TS_MAXY 4000

// The STMPE610 uses hardware SPI on the shield, a
#define STMPE_CS 8
Adafruit_STMPE610 ts = Adafruit_STMPE610(STMPE_CS)
```

```
wrote 65536 bytes from file /tmp/arduino_build_987156/
** Programming Finished **
halted at 0x20400004 due to step
```

Figure 6.2: Selecting HiFive1 in the Arduino ID

HiFive1 Additional Notes

This chapter gives some additional information about how your FE310-G000 chip is configured on the HiFive1 board.

To understand the FE310-G000 internals, please refer to
https://dev.sifive.com/documentation/freedom-e310-g000-manual/

Freedom E310 Pinout

The Freedom E310 design on the HiFive1 has 32 SW-accessible GPIO registers. Not all of these are available on the 48-pin package and are therefore not connected to the GPIO Headers on the board. Table 7.1 gives the mapping between the software GPIO offset and labeled pin on the board.
Figure ?? gives a graphical depition of the HiFive1's pin functions.

OTP Contents

The FE310-G000 on the HiFive1 board contains a one-time programmable memory (OTP). This OTP is shipped with code that enables your HiFive1 to boot out of reset (vs. loading a program with the programmer and running it). The OTP also contains trim values for the high-frequency oscillator on your FE310-G000 and a unique identifier.

The OTP contents are critical for the proper functioning of your FE310-G000 and HiFive1 board. You are

strongly discouraged from modifying the OTP contents. OTP contents can NOT be restored if modified.

HiFive1 Boot Sequence

The HiFive1 Board is shipped with a modifiable boot loader at the begnning of SPI Flash (0x20000000). At the end of this program's execution the core jumps to the main user portion of code at 0x20400000. This program is designed to allow quick boot, but also a "safe" reboot option if a "bad" program is flashed into the HiFive1's SPI Flash. A "bad" program is one which makes it impossible for the programmer to communicate with the HiFive1. For example, a program which disables FE310's active clock, or which puts the FE310 to sleep with no way of waking it up.

Table 7.1: HiFive1 GPIO Offset to Board Pin Number

HiFive1 Pin Number	GPIO Offset	IOF0	IOF1	LED
0	16	UART0:RX UART0:TX	PWM1 1 PWM1 0 PWM1 2 PWM1 3	GREEN BLUE RED
1	17			
2	18			
3	19			
4	20			
5	21			
6	22			
7	23			
8	0	SPI1:SS0 PI1:SD0/MOSI SPI1:SD1/MISO SPI1:SCK	PWM0 0 PWM0 1 PWM0 2 PWM0 3	–
9	1			
10	2			
11	3			

12	4			
13	5			
14		*Not Connected* SPI1:SS2 SPI1:SS3	PWM2 0 PWM2 1 PWM2 2 PWM2 3	–
15	9			
16	10			
17	11			
18	12			
19	13			

Bad programs can always be restarted using the RESET button, and using the “safe” boot loader can be halted before they perform any unsafe behavior.

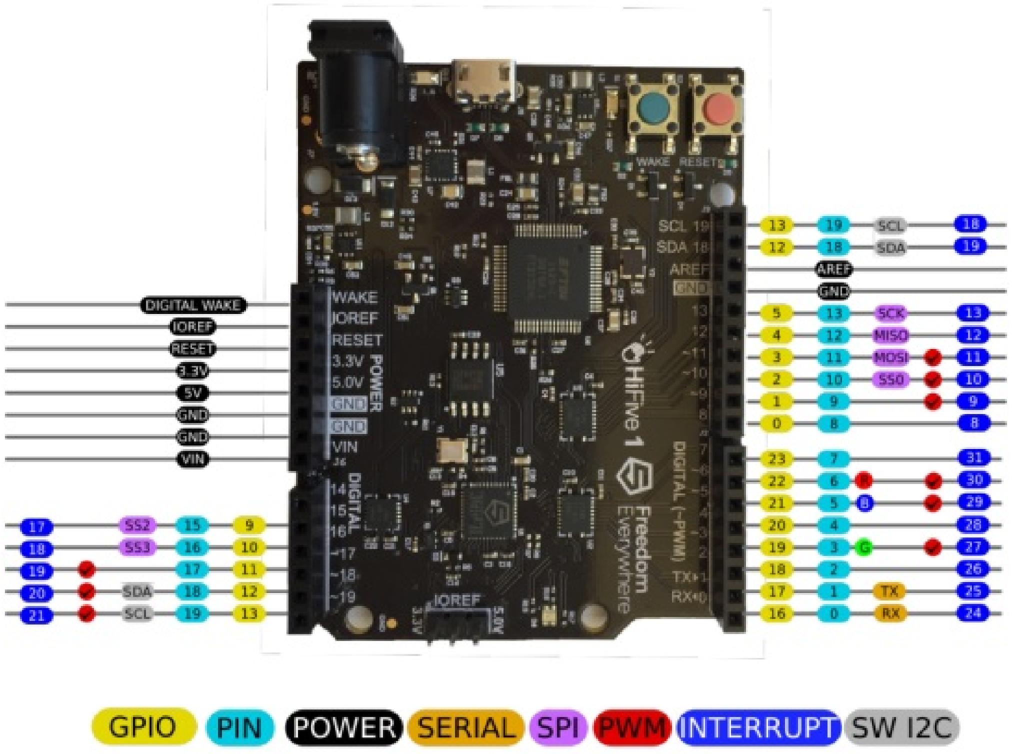
To activate “normal” boot mode, press the RESET button on the HiFive1. After approximately 1s, the green LED will flash for 1/2 second, then the user program will execute.

To activate “safe” boot mode, press the RESET button. When the green LED flashes, immediately press the RESET button again. After 1 second, the red LED will blink. The user program will not execute, and the programmer can connect. To exit “safe” boot mode, press the RESET button a final time.

Table 7.2: HiFive1 Default OTP Contents

OTP Offset	Value	Field Description
0x0000	7f50106f (j 0x1FF4)	Reset Vector from 0x1000. Executes a relative jump to the last fenc e block in the OTP.
0x0004	00000000	Included Config String from 0x1004. This is left unburden as the Co nfig String specification is not yet finalized.
0x1FE4	*Varies*	Board Identifier
0x1FE8	00000000	LFROSC Trim Setting, left unburned on HiFive1
0x1FEC	*Varies*	HFROSC Trim Setting for 72MHz
0x1FF0	0x00000001	Indicates the other fields were successfully programmed
0x1FF4	0x0000000F (fence)	This functions as a NOP which allows later modification of the pro-gram the OTP executes. This word could be modified to a short rel-ative jump elsewhere in the OTP. The target instruction block shoul d also start with a fence instruction to allow later modifications to be chained in this way.
0x1FF8	200002b7 (lui t0, 0x200 00)	Load t0 with 0x20000000
0x1FFC	00028067 (jr t0)	Jump to start of SPI Flash

HiFive 1 Pinout



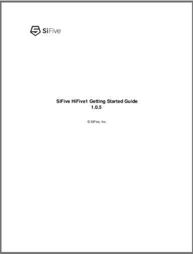
For More Information

Additional information, the latest version of this guide, and supporting files can be found at http://dev.sifive.com/hifive1.

Questions are best answered on the SiFive Forums at http://forums.sifive.com.

More information about RISC-V in general is available at http://riscv.org.

Documents / Resources

 SiFive SiFive HiFive1 Getting Started Guide 1.0.5 © SiFive, Inc.	SiFive HiFive1 Bluetooth Module [pdf] Owner's Manual HiFive1 Bluetooth Module, HiFive1, Bluetooth Module