

# StarFive StarStudio User Guide

Version: 0.3 Date: 2022/10/20 Doc ID: Dubhe-UGEN-005

## **Legal Statements**

Important legal notice before reading our documentation.

### **PROPRIETARY NOTICE**

Copyright © Shanghai StarFive Technology Co., Ltd., 2018-2022. All rights reserved.

Information in this document is provided "as is," with all faults. Contents may be periodically updated or revised due to the product development. Shanghai StarFive Technology Co., Ltd. (hereinafter "StarFive") reserves the right to make changes without further notice to any products herein.

StarFive 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.

StarFive 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.

All material appearing in this document is protected by copyright and is the property of StarFive. You may use this document or any part of the document for internal or educational purposes only, provided you do not modify, edit or take out of context the information in this document in any manner. Information contained in this document may be used, at your sole risk, for any purposes. StarFive authorizes you to copy this document, provided that you retain all copyright and other proprietary notices contained in the original materials on any copies of the materials and that you comply strictly with these terms. This copyright permission does not constitute an endorsement of the products or services.

### **Contact Us**

Address: Room 502, Building 2, No. 61 Shengxia Rd., China (Shanghai) Pilot Free Trade Zone, Shanghai, 201203, China

Website: http://www.starfivetech.com

Email: sales@starfivetech.com(sales) , support@starfivetech.com(support)

## Contents

List of Tables	4
List of Figures	5
Legal Statements	ii
Preface	. vi
1. Using StarFive StarStudio with Linux SDK	7
1.1. Prerequisites	7
1.2. Start Up StarFive StarStudio	8
1.3. Importing eSDK	8
1.4. Modifying Packages	9
1.5. Importing Packages to Project Explorer	10
1.6. Building Packages and QSPI Image	11
1.7. Running QEMU	12
1.8. Flashing Binary into FPGA	13
1.9. Debugging OpenSBI or Linux Kernel	14
1.9.1. Installing Minicom on Linux to View FPGA Output	15
2. Using StarFive StarStudio with Bare Metal SDK	18
2.1. Prerequisites	18
2.2. Start Up StarFive StarStudio	18
2.3. Importing StarFive Bare Metal SDK	18
2.4. Building and Cleaning Program	19
2.5. Adding New Compile Program	20
2.6. Debugging with Single Core or Multi Cores	22
3. Debugging on StarFive StarStudio	24
3.1. Debug Perspective	24
3.2. Memory View	24
3.3. Registers View	25
3.4. Setting Breakpoint	25
3.5. Disassembly View	26

## **List of Tables**

Table 0-1 Revision History......vi



## List of Figures

Figure 1-1 Example Command and Output	7
Figure 1-2 Example Command and Output	8
Figure 1-3 StarFive Linux SDK	9
Figure 1-4 Terminal Interface	10
Figure 1-5 Import Wizard	
Figure 1-6 Choose Flat Mode	
Figure 1-7 Imported Package	11
Figure 1-8 Example Output	12
Figure 1-9 Build QSPI Image	
Figure 1-10 Example Output	
Figure 1-11 Popup Window	
Figure 1-12 Example Output	
Figure 1-13 Auto-populated Configuration	
Figure 1-14 Example Output	15
Figure 1-15 Example Output	
Figure 1-16 Example Output	
Figure 1-17 Example Interface	16
Figure 1-18 Example Baud Rate	17
Figure 1-19 Example Output	17
Figure 2-1 StarFive Baremetal SDK View	
Figure 2-2 Build Program Output	19
Figure 2-3 Example Build Outputs	20
Figure 2-4 Clean Program Output	20
Figure 2-5 Example Pop Up	
Figure 2-6 Example Output	21
Figure 2-8 Example Interface	22
Figure 2-9 Example Output	23
Figure 3-1 Example Debug Perspective	
Figure 3-2 Memory View	25
Figure 3-3 Registers View	25
Figure 3-4 Breakpoints View	25
Figure 3-5 Toggle Breakpoint	26
Figure 3-6 Debugger Console	
Figure 3-7 Disassembly View	27

## Preface

About this guide and technical support information.

### About this document

This document mainly provides the users with the necessary information to use StarFive StarStudio.

StarFive StarStudio provides users an interface to interact with StarFive Bare Metal SDK and StarFive Linux SDK to perform various actions.

This document is intended to help users:

- Use StarFive StarStudio with Linux SDK as described in Using StarFive StarStudio with Linux SDK (on page 7).
- Use StarFive StarStudio with Bare Metal SDK as described in <u>Using StarFive StarStudio with Bare Metal SDK (on page</u> <u>18)</u>.

### **Revision History**

### Table 0-1 Revision History

Version	Released	Change Description
0.1	2022/06/16	Preliminary version.
0.2	2022/09/07	Updated the product name.
0.3	2022/10/20	Editorial updates.

### Notes and notices

The following notes and notices might appear in this guide:

Tip:

Suggests how to apply the information in a topic or step.

Note:

Explains a special case or expands on an important point.

Important:

Points out critical information concerning a topic or step.

Indicates that an action or step can cause loss of data, security problems, or performance issues.

### Warning:

Indicates that an action or step can result in physical harm or cause damage to hardware.

## 1. Using StarFive StarStudio with Linux SDK

This chapter provides steps to use StarFive StarStudio with Dubhe Linux SDK.

It contains the following procedures:

- 1. Complete the Prerequisites (on page 7)
- 2. Start Up StarFive StarStudio (on page 8)
- 3. Importing eSDK (on page 8)
- 4. Modifying Packages (on page 9)
- 5. Importing Packages to Project Explorer (on page 10)
- 6. Building Packages and QSPI Image (on page 11)
- 7. Running QEMU (on page 12)
- 8. Flashing Binary into FPGA (on page 13)
- 9. Debugging OpenSBI or Linux Kernel (on page 14)

### **1.1. Prerequisites**

Make sure you extract and install both StarFive StarStudio and StarFive Linux eSDK on your host machine before starting up the StarFive StarStudio:

1. Copy the StarStudio-beta-202206.tar.gz file into your directory and untar it:

```
$ tar -zxvf StarStudio-beta-202206.tar.gz
```

### Result

The extracted folder is ready to use.

2. Install the Linux eSDK by running the following two scripts:

° \$ ./x86\_64-buildtools-nativesdk-standalone-nodistro.0.sh

Figure 1-1 Example Command and Output

```
thioh@thioh-VirtualBox: ~/build4/build/tmp-glibc/deploy/sdk
 F1
                                                      0
/xthloh@thloh=virtualBox:~/build4/build/tmp-glibc/deploy/sdk$ ./x86_64-buildtoo
ls-nativesdk-standalone-nodistro.0.sh
Build tools installer version nodistro.0
   Enter target directory for SDK (default: /usr/local/oecore-x86_64):
You are about to install the SDK to "/usr/local/oecore-x86_64". Proceed [Y/n]? y
Setting it up...done
SDK has been successfully set up and is ready to be used.
Each time you wish to use the SDK in a new shell session, you need to source the
environment setup script e.g.
$ . /usr/local/oecore-x86_64/environment-setup-x86_64-oesdk-linux
thloh@thloh-VirtualBox:~/build4/build/tmp-glibc/deploy/sdk$
```

° \$ ./oecore-x86\_64-riscv64-toolchain-ext-nodistro.0.sh

#### Figure 1-2 Example Command and Output

• •		
<pre>mdcfpga1@mdcfpga1-VirtualBox:~/jy_workspace\$ ./oecore-x86_64-riscv64-toolchain-ext-nodistro.0.sh StarFive Linux SDK Extensible SDK installer version nodistro.0</pre>		
Enter target directory for SDK (default: ~/nodistro sdk): ~/iv workspace/esdk openocd		
You are about to install the SDK to "/home/mdcfpga1/jv workspace/esdk openoed". Proceed [Y/n]? Y		
Extracting SDK		
Setting it up		
Extracting buildtools		
Preparing build system		
Loading cache: 100%   ETA::		
	I DTA	0.00.40

### 1.2. Start Up StarFive StarStudio

To start up the StarFive StarStudio, perform the following steps:

- 1. Go to the StarFive StarStudio folder and open the executable to open StarFive StarStudio
- 2. Select Workspace for StarFive StarStudio IDE and click Launch.
- 3. Close the Welcome tab.

### 1.3. Importing eSDK

Perform the following steps to import eSDK:

- 1. Click Import Projects under Project Explorer on the left side of your StarFive StarStudio workbench.
- 2. Select StarFive Dubhe Linux SDK under the StarFive folder from the Import Wizard, and click Next.
- 3. Import StarFive Linux SDK project as prompted:
  - Project Name: The project name. For example, linux\_sdk.
  - Existing eSDK Location: The directory of the installed eSDK that you want to import. For example, /home/ jytan/linux\_sdk.
  - Olimex OpenOCD Config File Location: The openOCD Config file location. The default file is <eSDK>/tmp/ deploy/images/starfive-dubhe/olimex-openocd\_s5.cfg.
- 4. Click Finish and the directory of the eSDK will be imported.

#### Note:

The indexing for the build folder tmp/ is disabled as the directory is too large.

**Result:** 

You will also be greeted with the StarFive Linux SDK View:

### Figure 1-3 StarFive Linux SDK

StarFive Linux SDK ×		
eSDK Project:		~
eSDK environment script:	environment-setup-riscv64-oe-linux Default	]
Devtool Modify to Generate So	ource Codes: Modify P	ackage
Import package source codes:	└── Load Packages Import	.]
Build imported package:	Load Imported Packages Build Packages	ckage
Build QSPI Image:	Build QS	PI Image
Run RISC-V 64-bit QEMU:	Run QEM	1U
Flash QSPI Image into FPGA:	Settings	
	1 C M M	

For instructions to modify the package, see the Modifying Packages (on page 9) section.

### 1.4. Modifying Packages

Perform the following steps to modify the packages:

1. On the StarFive Linux SDK tab, configure the following:

eSDK Project: The eSDK project that is imported in <u>Importing eSDK (on page 8)</u>. For example, choose linux\_sdk.

• **Devtool Modify to Generate Source Codes**: The package that you want to do development on. For example, **opensbi**.

Note:

If you want to use command lines to modify packages, refer to the *Developing Applications or Modifying Packages* chapter in the *Dubhe Linux SDK User Guide*.

2. Click Modify Package.

**Result:** 

### | 1 - Using StarFive StarStudio with Linux SDK

The package source code is pulled into the workspace directory, and a terminal in StarFive StarStudio will be running the command:

### **Figure 1-4 Terminal Interface**

🗖 StarFive Linux SDK 😂			- 8	🕒 Console 🖉 Terminal 🛿 🔍 💭 🖓 👘 🖓 👘
eSDK Project:	linux_sdk		~	SDK environment now set up; additionally you may now run devtool to perform development
eSDK environment script:	environment-setup-risc	v64-oe-linux	Default	ks. Run devtoolhelp for further details. MOTE: Starting bitbake server Loading cache: 100% [###################################
Devtool Modify to Generate So	ource Codes:			Loaded 3865 entries from dependency cache. NOTE: Resolving any missing task queue dependencies
opensbi			Modify Package	Initialising tasks: 100%  ###################################
Import package source codes:	·	✓ Load Packages	Import	NOTE: Executing Tasks NOTE: Tasks Summary: Attempted 83 tasks of which 80 didn't need to be rerun and all succ ed.
Build imported package:		Load Imported Packages	Build Package	INFO: Source tree extracted to /home/jytan/linux_sdk/workspace/sources/opensbi INFO: Using source tree as build directory since recipe inherits autotools-brokensep
Build QSPI Image:			Build QSPI Image	LINFU: Recipe opensol now set up to build from /nome/jytan/linux_sdk/workspace/sources/ope bi
Run RISC-V 64-bit QEMU:			Run QEMU	
Flash QSPI Image into FPGA:			Settings	

## **1.5. Importing Packages to Project Explorer**

After the modification process is done, you can now import the package into StarFive StarStudio's Project Explorer. To import the opensbi package, perform the following steps:

- 1. Select the eSDK project under Import package source codes, and click Load Packages button on the StarFive Linux SDK tab to load the packages available.
- 2. Press the Import... button. Result:

you will be directed to a simple import wizard. The data will be auto-populated:

#### Figure 1-5 Import Wizard

Import a StarFive Linux SDK Package

### Import Existing Code

Create a new Makefile project from existing code in that same directory

Project Name  pensbi Existing Code Location /home/jytan/linux_sdk/workspace/sources/openst	Dİ	Browse
?	Cancel	Finish

3. Press **Finish** to complete the import package process.

4. (Optional) If the Project Explorer is still the same after you import the packages, you will need to click the settings button on **Project Explorer**, and go to Projects **Presentation** to choose it in a **Flat** mode:

Figure 1-0 Choose Flat	widde					
ြဲ Project Explorer 🛿	🗄 Outline			🗖 StarFive Linux SDK 🛛		
	E	\$ 7	000	eSDK Project:	linux	çsdk
🕶 😅 linux_sdk				Projects Presentation	•	<u>F</u> lat
🕨 🛞 Build Targets				Top Level Elements	•	• <u>H</u> ierarchica
🕨 🗁 buildtools				Select Working Set		· · · · · · · · · · · · · · · · · · ·
🕨 🗁 cache				Deselect Working Set		Codes:
🕨 🗁 conf				Edit Active Working Set		
downloads				🤪 1 Window Working Set		
Iayers				Package Presentation		
Sstate-cache			5	Show 'Deferenced Libraries	'Node	SDI
Sysroots			<u> </u>		Node	shi
WORKSpace			-	Filters and Customization		
Imp interview cooker						
	udenion.iog	IV.				
	d system log			Run RISC-V 64-bit QEMU:		
The share of the second	a_system.tog					
site-config-risc	v64-oe-linux			Flash QSPUmage into FPG	uA:	
version-riscv64	-oe-linux					
					•	

### Figure 1-6 Choose Flat Mode

### Result:

After choosing the settings, you will now notice that the package imported is in Project Explorer now:

### Figure 1-7 Imported Package



## 1.6. Building Packages and QSPI Image

After modifying your package's source code, you can now build the package.

1. Click **Build Package** to load the packages available and to build the package. **Result:** 

### Figure 1-8 Example Output

🗖 StarFive Linux SDK ⊠			-	• • •	onsole 🍠 Terminal 🛙	= x   fa 🔒 🖬   fa 🧬 😑 🗖
eSDK Project:	linux_sdk			SDK	Closed>Devtool build o environment now set	pensbi 🛿 up; additionally you may now run devtool to perform development tasks.
eSDK environment script:	environment-setup-	riscv64-oe-linux	Def	fault NOTE	: Starting bitbake s Reconnecting to bi	urther detalls. server Ltbake server
Devtool Modify to Generate Se	ource Codes:			NOTE NOTE NOTE	: Retrying server co : Reconnecting to bi : Previous bitbake i	nnection (#1) itbake server instance shutting down?, waiting to retry
opensbi			Mod	dify Pa( NOTE Load	: Retrying server co ing cache: 100%  ###	onnection (#2) #################################
Import package source codes:	opensbi	Load Package	es Imp	Dort Pars	ed 3864 entries from ing recipes: 100%  # ing of 2491 .bb file	n dependency cache. ####################################
Build imported package:	opensbi	Load Importe	ed Packages Bui	ld Pack Load	rs. ing cache: 100%  ###	######################################
Build QSPI Image:			Bui	ld QSP Pars Pars	ed 3864 entries from ing recipes: 100%  ∔ ing of 2491 .bb file	n dependency cache. ####################################
Run RISC-V 64-bit QEMU:			Run	QEML erro	rs. : Resolving any miss	sing task queue dependencies
Flash QSPI Image into FPGA:			Set	tings Chec Ssta	ialising tasks: 100% king sstate mirror o te summary: Wanted 1	ה  ###################################
				NOTE NOTE es/l	: Executing Tasks : linux-starfive-dev inux-starfive-dev	<pre>/: compiling from external source tree /home/jytan/linux_sdk/workspace/sourc</pre>
				NOTE	: opensbi: compiling : Tasks Summary: Att	g from external source tree /home/jytan/linux sdk/workspace/sources/opensbi cempted 3101 tasks of which 3045 didn't need to be rerun and all succeeded.
				NOTE	: do_populate_sysi : do_package_qa: 0	Jummary. oot: 0.0% sstate reuse(0 setscene, 6 scratch) 0.0% sstate reuse(0 setscene, 2 scratch)
				NOTE	: do_package: 0.0 : do_packagedata:	<pre>s sstate reuse(0 setscene, 3 scratch) 0.0% sstate reuse(0 setscene, 3 scratch)</pre>
				NOTE	: do_package_write : do_populate_lic:	e_ipk: 0.0% sstate reuse(0 setscene, 2 scratch) : 0.0% sstate reuse(0 setscene, 1 scratch)

2. (Optional) Build a new updated QSPI Image by clicking the Build QSPI Image. button.

### Figure 1-9 Build QSPI Image

📃 Console 🖉 Terminal 🛿	
📮 <closed> Build QSPI Imag</closed>	e 🛛
SDK environment now set	up; additionally you may now run devtool to perform development tasks.
Run devtoolhelp for f	urther details.
NOTE: Starting bitbake s	erver
NOTE: Reconnecting to bi	tbake server
NOTE: Retrying server co	nnection (#1)
NOTE: Reconnecting to bi	tbake server
NOTE: Previous bilbake in	nstance shutting downr, walting to retry
Loading cache: 100%  ###	######################################
Loaded 3864 entries from	dependency cache.
Parsing recipes: 100%  #	######################################
Parsing of 2491 .bb file	s complete (2489 cached, 2 parsed). 3865 targets, 172 skipped, 0 masked, 0
errors.	
WARNING: Skipping recipe	linux-starfive-dev as it doesn't produce a package with the same name
INFO: Building image qsp.	i image with the following additional packages: opensbi
Loading cache: 100% ####	######################################
Date ind recipes: 100% 1#	
Parsing of 2491 hh file	s complete (2488 cached 3 parsed) 3865 targets 172 skipped 0 masked 0
errors.	s comptete (2400 cachea, 5 parsea). 5005 cargets, 172 skippea, 6 maskea, 6
NOTE: Resolving any miss	ing task gueue dependencies
Initialising tasks: 100%	.  ####################################
Checking sstate mirror o	bject availability: 100%  ###################################
Sstate summary: Wanted 2	76 Local 0 Network 0 Missed 276 Current 1097 (0% match, 79% complete)
Removing 2 stale sstate	objects for arch starfive_dubhe: 100%  ###################################
NOTE: Executing Tasks	ampted 2022 tasks of which 2010 didn't need to be remun and all succeeded
NOTE: HASKS SUMMIATY: ALL	empled 5955 lasks of which 5916 didn't need to be ferun and all succeeded.
NOTE: do package ga: 0	0% sstate reuse(0 setscene, 2 scratch)
NOTE: do package: 0.0%	sstate reuse(0 setscene. 1 scratch)
NOTE: do package write	ipk: 0.0% sstate reuse(0 setscene, 2 scratch)
INFO: Successfully built	qspi-image. You can find output files in /home/jytan/linux_sdk/tmp/deploy/
images/starfive-dubhe	

## 1.7. Running QEMU

To run a QEMU image, press the **QEMU** button.

The following is the example output:

### Figure 1-10 Example Output

📮 Console 🖉 Terminal 🗙	📃 🙌 🗓 🔒 🔒 🗎 🖿 🗖
🖳 Run QEMU 🗡	
<pre>runqemu - INFO - Using p runqemu - INFO - If this e runqemu skip tap0. runqemu - INFO - Network runqemu - INFO - Network runqemu - INFO - Running native/1.0-r1/recipe-sys ci,netdev=net0,mac=52:54 cript=no -object rng-rar =rng0 -initrd /home/jyta initramfs-starfive-dubke -m 2048 -serial mon:stdi p/deploy/images/starfive-du m0 rw debugshell mem=20</pre>	preconfigured tap device tap0 s is not intended, touch /tmp/qemu-tap-locks/tap0.skip to mak a configuration: ip=192.168.7.2::192.168.7.1:255.255.255.0 g /home/jytan/esdk_openocd/tmp/work/x86_64-linux/qemu-helper- proot-native/usr/bin/qemu-system-riscv64 -device virtio-net-p e:00:12:34:02 -netdev tap,id=net0,ifname=tap0,script=no,downs dom,filename=/dev/urandom,id=rng0 -device virtio-rng-pci,rng an/esdk_openocd/tmp/deploy/images/starfive-dubhe/dubhe-image- e.cpio.gz -smp 8 -machine virt -cpu rv64,x-h=true,x-b=true to -serial null -nographic -bios /home/jytan/esdk_openocd/tmp/d e-dubhe/fw_payload.bin -kernel /home/jytan/esdk_openocd/tmp/d bhe/Image-initramfs-starfive-dubhe.bin -append 'root=/dev/ra
OpenSBI v1.0	
Platform Name Platform Features Platform HART Count Platform IPI Device Platform Timer Device Platform Console Device Platform HSM Device Platform Reboot Device Platform Shutdown Device Firmware Base Firmware Size Runtime SBI Version Domain0 Name	<pre>: riscv-virtio,gemu : medeleg : 8 : aclint-mswi : aclint*mtimer @ 10000000Hz : uart8250 : : sifive_test : sifive_test : 0x80000000 : 312 KB : 0.3 : root</pre>

## **1.8. Flashing Binary into FPGA**

To flash the generated FPGA Binary Image (QSPI-Image.bin) into the FPGA, perform the following step:

Press **Settings...** button on the Flash QSPI Image into the FPGA row: **Result:** 

An FPGA Window will pop out:

### Figure 1-11 Popup Window

### FPGA Binary QSPI-Image Flash Assistant

 Choose the intended QSPI-Image.bin to flash into the FPGA. It will be flashed starting from the 0x20000000 address.

QSPI Image Directory:	1	Browse
Starting Flash Address:	0x2000000	
Offset Address:	0x0	
An openood config file will be gen	Cancel	OK and to wait for a
while for the image to be flashed,	depending on the size of the binary file.	ed to wait for a
Console Prerminal		n 🕫 – 🗉
<pre><closed> FPGA Flashing SDK environment now set asks. Run devtoolhelp for Open On-Chip Debugger 0 Licensed under GNU GPL For bug reports, read http://openocd. configure work area configure work area com configure flash bank</closed></pre>	Image: Signature         up; additionally you may now run devtool to perform         further details.         .11.0+dev-02358-ga037b20f2-dirty (2022-05-18-07:11)         v2         org/doc/doxygen/bugs.html         plete         mplete	n development t
Info : ftdi: if you exp do_sample_edge falling" Info : clock speed 2000 Info : JTAG tap: riscv. 0000, ver: 0x0) Info : [riscv.cpu] data Info : Disabling abstra Info : Examined RISC-V Info : hart 0: XLEN=64 [riscv.cpu] Target succ Info : starting gdb ser Info : Listening on por Info : JTAG tap: riscv. 0000, ver: 0x0)	erience problems at higher adapter clocks, try the o 0 KHz cpu tap/device found: 0x00000cfd (mfg: 0x67e ( <unkno count=4 progbufsize=16 ct command reads from CSRs. core; found 2 harts , misa=0x80000000001411af essfully examined. ver for riscv.cpu on 3333 t 3333 for gdb connections cpu tap/device found: 0x00000cfd (mfg: 0x67e (<unkno< td=""><td>:ommand "ftdi t own&gt;), part: 0x own&gt;), part: 0x</td></unkno<></unkno 	:ommand "ftdi t own>), part: 0x own>), part: 0x
Info : Found flash devi I am flashing Info : Disabling abstra Done flashing!!! shutdown command invoke	ce 'mac 66ulg45g' (ID 0x003b25c2) ct command writes to CSRs. d	

## 1.9. Debugging OpenSBI or Linux Kernel

### Prerequisite:

The StarFive Linux eSDK project, openSBI or Linux Kernel (linux-starfive-dev), is imported into StarFive StarStudio, and two configurations are auto-populated: **openSBI** and **Linux Kernel**:

To debug OpennSBI or Linux Kernel, perform the following steps:

- 1. Navigate to Run > Debug Configurations... and double click GDB OpenOCD Debugging. There will be two default configs available:
  - Figure 1-13 Auto-populated Configuration
  - GDB OpenOCD Debugging
    - C StarFive Default Linux Kernel Config
    - StarFive Default OpenSBI Config
- 2. Choose the configuration (**openSBI** in this example) and press **Debug** to debug with the config. **Result:**

You can now debug OpenSBI! For a detailed introduction to debugging on StarFive StarStudio, see <u>Debugging on</u> StarFive StarStudio (on page 24).

### Figure 1-14 Example Output

File Edit Navigate	Search Project Run Window Help		
🖬 🕶 📾  📾 🗄	🛢 i 🐂 i 🗴   🕨 🗉 🛎 🖄 🎗 🦻 🖄   🕪 🗟 🗷	८८ । क <b>र 0 र % र</b> । छ ⋪ र । छ र छ र छ र अध ⇔ र ० र ा	Q   🖻 🖥
🏇 Debug 🚡 Project E	xplorer 🖼 Disassembly 😫 👘 🗇	C Debugger Console 13	<b>□</b> • □
	Enter location here 👻 🚯 🖄 😫 📑 😁 🕴	OpenOCD Spike OpenSBI [GDB OpenOCD Debugging] /home/jytan/nodistro_sdk/tmp/sys;oots/x80_64/usr/bin/risc464-oe-linux/risc44-oe-linux/risc464-oe-linux/risc46-oe-linux/risc46-oe-linux/risc46-oe-linux/risc46-oe-linux/risc44-oe-linux/risc46-oe-linux/risc464-oe-linux/risc44-oe-linux/risc464-oe-linux/risc44-oe-linux/risc44-oe-linux/risc44-oe-linux/risc44-oe-linux/risc44-oe-linux/risc44-oe-linux/risc44-oe-linux/risc44-oe-linux/risc44-oe-linux/risc44-oe-linux/risc44-oe-linux/risc44-oe-linux/risc44-oe-linux/risc44-oe-linux	(
00000000000000000000000000000000000000	<pre>jal rs.0200000302 cfw_boot_hart&gt;     add u8, u8, u2, rsc     add u8, u8, u2, rsc     add u8, u8, rsc     add u8, u8, rsc     add u8, u8, rsc     u8, u8, u2, u2, u2, u2, u3, u2, u2, u3, u3, u2, u3, u3, u2, u3, u3, u3, u3, u3, u3, u3, u3, u3, u3</pre>	For help. type "help". Type "approx word to served to compade related to 'Nord". Marning: the current Impage does not matter this frame. Function main from of defauld. No source file named main. Stop Stop Stop Stop Stop Stap	
63	amoadd.w a6, a7, (a6)	🖸 Console 🗿 fw_base.5 🕴 🕮 Registers 🛪 Progress 🚹 Problems 💽 Executables 🕕 Memory	
0000000000031 64 0000000000000000000000000000000000	ambodo.v ap.dv/to) ambodo.v ap.dv/to) more af.ecoModele cuil_relecate_copy_dene Nutre inc. load_start Nutre inc. ap.def ind to inc. ap.def i	<pre>5 5 For DRU BR 80, 50, 51 52, 52 5 11 37, 1 try lottery 5 2 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3</pre>	
			6

You will need to connect to the FPGA via minicom to view the FPGA's output. For instructions for setting up minicom, see Installing Minicom on Linux to View FPGA Output (on page 15).

### 1.9.1. Installing Minicom on Linux to View FPGA Output

To connect to the FPGA via minicom to view the FPGA's output, perform the following steps:

1. To install minicom on Linux, run this command:

\$ sudo apt-get install minicom

You can check which serial port the FTDI USB UART (not the Olimex JTAG) is connected to using the command \$ 1s
 -lr /dev/serial. Ensure that the FTDI USB UART has the correct serial number for Dubhe (in case you are connecting to more than 1 FTDI UARTs).

| 1 - Using StarFive StarStudio with Linux SDK

Figure 1-15 Example Output	
ndcfpga1@ndcfpga1-VirtualBox:-/D /dev/serial: total 0 drwxr-xr-x 2 root root 80 Mei 2 drwxr-xr-x 2 root root 80 Mei 2	esktop\$ ls -lR /dev/serial 0 13:27 by-id 0 13:27 by-path
<sup>y</sup> /dev/serial/by-id: total 0 lrwxrwxrwx 1 root root 13 Mei 2 lrwxrwxrwx 1 root root 13 Mei 2	0 13:27 usb-15ba_Olimex_OpenOCD_JTAG_ARM-USB-TINY-H_OLDBAA13-if01-port0 ->//ttyUSB0 0 13:27 usb-FTDI_FT232R_USB_UART_A20021XZ-if00-port0 ->//ttyUSB1

3. Execute the following command to set up minicom:

\$ sudo minicom -s

#### **Result:**

You will be brought to an interface shown below:

#### Figure 1-16 Example Output



- put
- 4. Go to **Serial port setup** and press A. Change the serial device to the port that the FTDI is connected to (for example: / dev/ttyUSB1). Press Enter after typing the serial port name, and type F to set the **Hardware Flow Control** to **No**. Ensure that the baud rate in **E** is **115200**.

Figure	e 1-17 Example Interface	
F	-	jytan@jytan: ~/Desktop 🛛 🗐 🗆 🛛 😣
	<ul> <li>A - Serial Device</li> <li>B - Lockfile Location</li> <li>C - Callin Program</li> <li>D - Callout Program</li> <li>E - Bps/Par/Bits</li> <li>F - Hardware Flow Control</li> <li>G - Software Flow Control</li> <li>Change which setting?</li> </ul>	: /dev/ttyUSB1 : /var/lock : : 115200 8N1 : No : No
	Screen and keyboa   Save setup as dfl   Save setup as   Exit   Exit from Minicom	rd

5. Go to Save setup as dfl and press Enter to save the configuration. Then, go down to Exit from Minicom to exit.

#### -[configuration]--[configuration]. Filenames and paths Filenames and paths File transfer protocols File transfer protocols Serial port setup Serial port setup Modem and dialing Modem and dialing Screen and keyboard Screen and keyboard Save setup as dfl Save setup as dfl Save setup as.. Save setup as.. Exit Exit Exit from Minicom Exit from Minicon

6. After successfully saving the configuration, you can now use this command in the terminal to get access to the FTDI USB port:



After successfully saving the configuration, you can now use this command in the terminal to get access to the FTDI USB port:

\$ sudo minicom

Figure 1-18 Example Baud Rate

## 2. Using StarFive StarStudio with Bare Metal SDK

This chapter provides steps to use StarFive StarStudio with Dubhe Bare Metal SDK.

It contains the following procedures:

- 1. Complete Prerequisites (on page 18)
- 2. Start Up StarFive StarStudio (on page 8)
- 3. Importing StarFive Bare Metal SDK (on page 18)
- 4. Building and Cleaning Program (on page 19)
- 5. Adding New Compile Program (on page 20)
- 6. Debugging with Single Core or Multi Cores (on page 22)

### 2.1. Prerequisites

Make sure you perform the following steps before starting up the StarFive StarStudio:

- 1. Extracted StarFive StarStudio on your host machine.
- 2. Copy the StarStudio-beta-202206.tar.gz file into your directory and untar it:

\$ tar -zxvf StarStudio-beta-202206.tar.gz

#### Result

The extracted folder is ready to use.

- 3. Extracted and installed StarFive Bare Metal SDK on your host machine
- 4. Have a pre-built StarFive bare metal toolchain on your host machine.



To obtain Bare Metal SDK and toolchain files, please refer to the *Dubhe Bare Metal SDK User Guide* for the relevant links and steps to install.

## 2.2. Start Up StarFive StarStudio

To start up the StarFive StarStudio, perform the following steps:

- 1. Go to the StarFive StarStudio folder and open the executable to open StarFive StarStudio.
- 2. Select Workspace for StarFive StarStudio IDE and click Launch.
- 3. Close the **Welcome** tab.

## 2.3. Importing StarFive Bare Metal SDK

- 1. Click Import Projects in Project Explorer on the left side of your StarFive StarStudio workbench:
- 2. Select StarFive Dubhe Baremetal SDK under the StarFive folder from the Import Wizard, and click Next:
- 3. Choose your existing Bare Metal SDK:
- 4. Import the Bare Metal SDK project as prompted:
  - **Project Name**: The project name. For example, bare-metal-sdk2.
  - Existing SDK Location: The directory of the existing bare metal that you want to import. For example, /home/ jytan/Desktop/baremetal/SDK/20220513/bare-metal-sdk.

 Click Finish and the files and directory of the SDK will be imported. Result:

You will be greeted with the StarFive Bare Metal SDK view:

□ StarFive Baremetal SDK ⊠	
Project Name	$\sim$
Toolchain Path:	
	Browse
PROGRAM	└ └oad Programs from Project
TARGET	~
CONFIGURATION	~
BUILD PROGRAM CLEAN PROGRAM	ADD NEW PROGRAM
Debug ELF with FPGA	
Debug Config:	UEBUG

### Figure 2-1 StarFive Baremetal SDK View



You can also re-open the Baremetal view by going to Window > Show View > StarFive Baremetal SDK:

6. To start compiling programs for Dubhe, you will need to fill in the **Project Name** and the**Toolchain Path** (Fill in with the pre-built bare metal toolchain) on the StarFive Bare Metal SDK view.

## 2.4. Building and Cleaning Program

This section provides steps to build and clean the program:

- 1. Choose the **PROGRAM**, and click **Load Programs from Project** so that it will load the programs located in the directory <br/>
  <br/>
  sparemetal\_SDK>/software.
- 2. Choose the **Target** and **Configuration** for compiling, and click **BUILD PROGRAM** to build the program with the selected toolchain.

Result:

After the program is built, you will see an output in the Console View:

### Figure 2-2 Build Program Output



3. Locate the build outputs in the directory <baremetal\_SDK>/software/<program>/<config> after the build is finished:

Figure 2-3 Example Build	Outputs		
눱 Project Explorer 🛛	🕮 📴 Outline	🖻 🕏 🍸 🖇	
🕶 🕞 > bare-metal-sd	<b>lk</b> [bare-metal-sdk ma	aster]	
🕨 📴 > psb			
🕨 📴 freedom-me	tal		
🕨 📴 FreeRTOS-me	etal		
🕨 📴 > scripts			
🔻 📴 > software			
🔻 📴 > atomics			
🔻 🔄 > releas	se		0
🗟 aton	nics.elf	1	
📑 aton	nics.hex		
📑 aton	nics.lst		
📑 aton	nics.map		X
4. (Optional) You can also click (	CLEAN PROGRAM to remove	the program build directory fro	om the Baremetal SDK folder.
Result:			
Figure 2-4 Clean Program	Output		
📃 Console 🛙			

```
CDT Build Console [bare-metal-sdk]

10:03:13 **** Clean-only build of configuration Default for project bare-metal-sdk ****

make PROGRAM=atomics TARGET=dubhe-rtl CONFIGURATION=release clean

make -C /home/jytan/Desktop/baremetal/SDK/20220513/bare-metal-sdk/software/atomics PORT_DIR= clean

make[]: Entering directory '/home/jytan/Desktop/baremetal/SDK/20220513/bare-metal-sdk/software/atomics'

rm -f atomics atomics.hex

make[1]: Leaving directory '/home/jytan/Desktop/baremetal/SDK/20220513/bare-metal-sdk/software/atomics'

rm -rf /home/jytan/Desktop/baremetal/SDK/20220513/bare-metal-sdk/software/atomics'

rm -rf /home/jytan/Desktop/baremetal/SDK/20220513/bare-metal-sdk/software/atomics'

rm -rf scripts/elf2hex/build scripts/elf2hex/install

rm -rf /home/jytan/Desktop/baremetal/SDK/20220513/bare-metal-sdk/bsp/dubhe-rtl/install

rm -rf /home/jytan/Desktop/baremetal/SDK/20220513/bare-metal-sdk/bsp/dubhe-rtl/install

rm -rf /home/jytan/Desktop/baremetal/SDK/20220513/bare-metal-sdk/bsp/dubhe-rtl/install

rm -rf /home/jytan/Desktop/baremetal/SDK/20220513/bare-metal-sdk/bsp/dubhe-rtl/build/release

10:03:13 Build Finished. 0 errors, 0 warnings. (took 301ms)
```

## 2.5. Adding New Compile Program

You can also add a new program via StarFive StarStudio plugin.

1. Choose the project and click Add New Program in the Baremetal View and a pop up will appear:

	Figure 2-5 Example Pop Up	)		
				8
	Add New package	to StarFive Baremetal SDK		
	<ol> <li>Type the intended directory. A sample</li> </ol>	package name to be added into the Baremetal SDK Makefile and C code template will be created as		
	Project selected:	bare-metal-sdk		
	New Program Name:			
2.	② Enter the project name, for ex <b>Result:</b>	Cancel ample, testing666 in New Program Name, and press OK.	ĸ	
	A new folder will be created in	n the Bare Metal SDK, with sample templates of a Makefile and a template C f	ile:	
	Figure 2-6 Example Output			
	🔻 🔄 > testing666			
	testing666.c			
	🚴 Makefile			
	Makefile 💽 testing666	.c 🖾		
		rFive, Inc */		
	<pre>#include <stdio.h></stdio.h></pre>			
	⊖static void welcome(v	oid)		
	<pre>{     printf("\r\n");     printf("*******     printf("********     printf("********     printf("*********     printf("**** Descr     printf("**** The t     printf("*********     printf("*********** </pre>	**************************************		
	}	-		
	⊖int main() {			
	<pre>welcome(); printf("[INFO]: t return 0; }</pre>	esting666 PASSED!\r\n");		

### | 2 - Using StarFive StarStudio with Bare Metal SDK

3. You can edit the C file and Makefile to any code for development, and the program can also be built via the StarFive Baremetal SDK View:

### Figure 2-8 Example Interface

•	•								
🚡 Makefile 🛛 🖬 te	esting666.c	🗖 StarFive Baremetal SDK 🛿							- 8
Project Name	bare-metal-so	lk							~
Toolchain Path:									
/home/jytan/Desk	top/baremeta	l/baremetal-0430							Browse
PROGRAM	testing666							~	Load Programs from Project
TARGET	dubhe-rtl								~
CONFIGURATION	release								~
BUILD PROGRAM	CLEAN PROC	GRAM							ADD NEW PROGRAM
Debug ELF with FPG	A								
Debug Config:									DEBUG
							1		
💷 Console 🛿						¥ 🥹 🕯	🎭 🖬 🖬 🏝	B.	
CDT Build Console [ba End of search lis	are-metal-sdk]								
"/home/jytan/Des	ktop/baremeta	al/baremetal-0430/bin/riscv	64-unknown-elf-ld" -	m elf64lriscvg	-sections -Map	testing666.ma	ap /tmp/testing	666-e	?7cae7.0start-group -lc
<pre>make[1]: Leaving o mv /home/jytan/De mv /home/jytan/De touch -c /home/jytan/De</pre>	sktop/baremet sktop/baremet sktop/baremet	iome/jytan/Desktop/Daremeta tal/SDK/20220513/bare-metal tal/SDK/20220513/bare-metal paremetal/SDK/20220513/bare	-sdk/software/testin -sdk/software/testin -sdk/software/testin	metal-sok/softwar g666/testing666 / g666/testing666.m testing666/releas	e/testingboo' nome/jytan/Deskt ap /home/jytan/D e/testing666 elf	op/baremetal/ esktop/bareme	/SDK/20220513/b etal/SDK/202205	are-m 13/ba	netal-sdk/software/testing6 are-metal-sdk/software/test
/home/jytan/Deskto /home/jytan/Deskto text data	op/baremetal/ op/baremetal/ bss de	/baremetal-0430/bin/riscv64 /baremetal-0430/bin/riscv64 ec hex filename	-unknown-elf-objdump -unknown-elf-size /h	sourceall-h ome/jytan/Desktop	eadersdemangl /baremetal/SDK/2	eline-numb 0220513/bare-	berswide /ho -metal-sdk/soft	me/jy ware/	/tan/Desktop/baremetal/SDK/ /testing666/release/testing
24598 9216 scripts/elf2hex/i	20560 5437 nstall/bin/ri	74 d466 /home/jytan/Desk iscv64-unknown-elf-elf2hex	top/baremetal/SDK/20 output /home/jytan	220513/bare-metal /Desktop/baremeta	-sdk/software/te L/SDK/20220513/b	sting666/rele are-metal-sd	ease/testing666 k/software/test	.elf ing66	66/release/testing666.hex -
10:09:48 Build Fi	nished. 0 err	rors, 0 warnings. (took 653	ms )					ß	

## 2.6. Debugging with Single Core or Multi Cores

### Prerequisite:

Use the debug function, make sure the followings are completed:

- The bare metal SDK is imported as described in Importing StarFive Bare Metal SDK (on page 18).
- The minicom is installed as described in Installing Minicom on Linux to View FPGA Output (on page 15).

This section provides steps to debug with single core or multi cores.

- 1. Navigate to the **Debug Config** under **Debug ELF with FPGA** on the **StarFive Baremetal SDK** view, and select **Single Core** or **Dual Core**.
- 2. Launch minicom to show output from FPGA, click Debug and openOCD will launch to connect with the FPGA.

#### Note:

You will need to open a Debugger Console (Window > Show View > Debugger Console) to send commands on GDB for debugging.

```
3. Click DEBUG.
```

Result:

### Figure 2-9 Example Output

s 🗠 Terminat 🕶	Jun 14 16:33 ●	a 🕶 🖷
		mdcfpga1@mdcfpga1-VirtualBox: ~/Desktop 🔍 🗏 💷 🕻
	eclipse-workspace - Eclipse IDE	*****
File Edit Navigate Search Project Run Window Help		**************************************
1 1 × 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	IP 등 2	Ctint vector interrupt Demo
🏶 Debug 🗙 🏠 Project Explorer 🛛 📄 🐘 🖇 📟 🗖	Debugger Console ×      StarFive Baremetal SDK     Smetal_driver_sifive_test0_exit(	*** The NEW_PACKAGE demo code ***
C StarFive Dubhe Single Core [GDB OpenOCD Debugging]     SNEW PACK ACE alf		***************************************
"     "     "     Thread #1 (Suspended : Signal : SIGINT:Interrupt)	StarFive Dubne Single Core [GDB OpenOCD Debugging] /home/mdcfpga1/jy_workspace/bin/ri GNU gdb (Starfive GCC-Baremetal-2022-04-30) 10.1	[INFO]: NEW_PACKAGE PASSED!
<pre>metal_driver_sifive_test0_exit() at 0x80005c0e</pre>	copyright (C) 2020 Free Software Foundation, Inc. License GPLv3+: GNU GPL version 3 or later <htp: gnu.org="" gpl.html="" licenses=""></htp:>	
energiand and a composition of the second	This is free software: you are free to change and redistribute it. There is NO WARRANTY, to the extent permitted by law.	
ji riscv64-unknown-elf-gdb	Type "snow copying" and "snow warranty" for details. This GOB was configured as "host=x86 64-build pc-linux-gnutarget=riscv64- Type "charge configured is" configured confi	
Disassembly ×	For bug reporting instructions, please see:	
Enter location here V 🕲 🛱 😼 🗂 🗂 🕯	Find the GDB manual and other documentation resources online at:	
0000000080005bf9: 0x95a02133 0000000080005bfd: andi a4.a47	For help, type "help".	
000000000005bff: li a1,-31 0000000000005c01: li s10,21	Type "apropos word" to search for commands related to "word".	
0000000080005c03: srai a5,a5,0x23 0000000808005c05: c.nop 13	c Continuing.	
0000000080005c07: 0x9300 000000080005c09: begz a1,0x80005c31 <_metal_drive	Program received signal SIGINT, Interrupt.	
0000000080005c0b: 0x8888 0000000080005c0d: addi gp,gp,16	0x0000000080005c0e inmetal_driver_sifive_test0_exit ()	
0000000080005cU1: 0X9/00 000000080005cU1: c.nop 13	Country Wardshire & Reality of Street Market	
00000000000005c13: 043300 00000000000005c15: srai al,al,0x1 00000000000005c17: ld s0 0(5)	Console all Registers × The Problems Steelables Memory	
0000000000005C17: tu 50,0(d) 00000000080005C19: fsd fs1,40(a0) 0000000080005C1b: addi s0 s0 128	✓      ✓      ✓      ✓      ✓      ✓      ✓	
0000000080005c1d: srli a2,a2,0x11 0000000080005c1d: srli a2,a2,0x11	mizero 0	
000000000005c21: C.nop 9 000000000000005c23: addi s0.sp.416	1117a 0x80005f56 11115p 0x800094b0	CTRL-A Z for help   115200 8N1   NOR   Minicon 2.7.1   VI102   Offline   tyUSB
0000000080005c25: sra1 a0,a0,0x1 0000000080005c27: 0xaa8082f7	0x80008c68	
0000000080005c2b: addi a5,a5,-31 0000000080005c2d: c.slli zero,0x9	iiiito 8	

## 3. Debugging on StarFive StarStudio

When debugging on StarFive StarStudio IDE, a lot of useful tools or views can assist you to debug more efficiently and effectively. This chapter will introduce the Debug perspective and various views that are useful when debugging.

- Debug Perspective (on page 24)
- Memory View (on page 24)
- Registers View (on page 25)
- <u>Setting Breakpoint (on page 25)</u>
- Disassembly View (on page 26)

## 3.1. Debug Perspective

During debugging, you can open the **Debug Perspective** on StarFive StarStudio to have a more productive view as shown in the following figure.

### Figure 3-1 Example Debug Perspective



To open the Debug Perspective:

- 1. Go to Window > Perspective > Open Perspective > Other... > Debug
- 2. (Optional) If you still need some useful Views such as Disassembly View, you can go to Window > Show View to open the view that is needed.

### 3.2. Memory View

The **Memory** view in the **Debug** perspective enables users to monitor and modify the process memory. The process memory is presented as memory monitors, and users can specify the base address or location of each memory monitor.

You can add or remove memory monitors in the **Memory** view, and each memory monitor can display the memory content in different data formats such as hexadecimal or ascii.

### Figure 3-2 Memory View

📮 Console 🚟 Registers 🖹 Problems 🕥 Executable	s 🚺 Memory	×					1019 1010	Ľ 🛃	⇒ti	<b>U</b> 😵	80 -	8	- 8
Monitors 🕂 🐈 💥	0x8000564a	: 0x8000564	1A <hex> <math>\times</math></hex>	🖶 New Rer	derings								
	Address	0 - 3	4 - 7	8 - B	C - F								
0x8000564a	80005640	233CD4FC	EFF01FB2	AA841735	01001305								
	80005650	65A4EFF0	8FB8A270	02742685	E2646561								
	80005660	82805971	22F006F4	001826EC	0CE410E8								
	80005670	14EC18F0	1CF42338	0403233C	1403F327								
	80005680	0034BC6B	13078400	233CE4FC	898B8144								
	80005690	99E7A270	02742685	E2646561	8280AA84								
	800056A0	17350100	1305059F	EFF06FB0	833684FD								
	800056B0	26868145	0145EFF0	FFAAAA84	17350100								
	000055550	10054500	FFFACEDI	10700074	200552004								

## 3.3. Registers View

StarFive StarStudio comes with a **Registers** view, that enables users to view the info of general-purpose registers for the target while debugging. Register values that have been changed would also be highlighted in the **Registers** view when the program stops. Users can also modify or assign the memory values of the target via **Registers** view.

Figure 3-3 Registe	ers View		
Console 🚟 Registers ×	🕈 Problems 🕡 Executables 🏮 Memory		
Name	Value	Description	
		General Purpose and FRU Regi	ster Group
1819 zero	0		
1000 ra	0x8000090e <sbi_init+642></sbi_init+642>		
<sup>1010</sup> sp	0x80040F30		
<sup>1819</sup> gp	0x0		
1889 <b>tp</b>	0x80041000		
1919 <b>to</b>	0		
1110 <b>t1</b>	4672		
3.4. Setting	Breakpoint		
Users can set break	point during debugging.		
Figure 3-4 Breakp	points View		
We variables			
🗹 🞜 [function: sbi_ir	hit] [type: Hardware]		
ਵ ਫ਼ੀ [function: sbi_p	rintf] [type: Hardware]		
No details to display for	the current selection.		

Users can view, delete, or deactivate breakpoints and modify breakpoint properties with the **Breakpoints** view in StarFive StarStudio. For users to set a breakpoint, users can either:

### | 3 - Debugging on StarFive StarStudio

• Right click in the left margin in the editor View and select **Toggle Breakpoint**. Double clicking the left margin of the line works too:

### Figure 3-5 Toggle Breakpoint

	Sta	rFive Linux SDK	Bebugger (	Console	🧟 sbi_init.c 🛛	
<b>4</b>	47 48 49 50	sbi_printf("\ <u>OPENSB</u> #endif	nOpenSBI v <sup>s</sup> BI_VERSION_M	%d.%d\n", <u>0</u> MINOR);	PENSBI_VERSI	DN_MAJOR,
	51 52 53	<pre>#ifdef OPENSBI_BU     sbi_printf("B #endif</pre>	JILD_TIME_S Build time:	TAMP %s\n", OPE	NSBI_BUILD_T	<pre>IME_STAMP);</pre>
G	EA				)	
	T	oggle Brea <u>k</u> point	S	shift+Ctrl+B		
	<u>A</u>	dd Breakpoint			OPENSBI_BUIL	D_COMPILER_VERSION);

• Alternatively, users can also set breakpoints during debugging. Via the debugger console, the user can set either a hardware breakpoint (**hbreak**) or a software breakpoint (**break**).

### Figure 3-6 Debugger Console



## 3.5. Disassembly View

The **Disassembly** view allows users to view the disassembled code during the stepping of code while debugging. The current executed assembly code will be indicated by an arrow marker and highlighted in the view.



igure 3-7 Disasse	mbly Vi	ew		
Disassembly ×			•	
	E	nter location here 👻 🔱 🖄 😫 📑 🖻	8	
00000000800006a4:	mv	s1,a0		
00000000800006a6:	csrr	s3,mhartid		
00000000800006aa:	li	a5,127		
00000000800006ae:	sext.w	s2,s3		
00000000800006b2:	ld	a0,48(a0)		
00000000800006b4:	bltu	a5,s2,0x800006da <sbi init+78=""></sbi>		
00000000800006b8:	begz	a0,0x800006da <sbi init+78=""></sbi>		
00000000800006ba:	lw	s4,80(a0)		
00000000800006be:	mv	a1,s2		
00000000800006c0:	jal	ra,0x80001362 <sbi hart="" index="" platform=""></sbi>		
00000000800006c4:	sext.w	a0,a0		
00000000800006c6;	baeu	a0.s4.0x800006da <sbi init+78=""></sbi>		
00000000800006ca:	ld	a5,32(s1)		
00000000800006cc:	li	a4.1		
00000000800006ce:	bea	a5.a4.0x80000866 <sbi init+474=""></sbi>		
00000000800006d2:	li	a4.3		
00000000800006d4:	bea	a5.a4.0x80000872 <sbi init+486=""></sbi>		
00000000000000000000000000000000000000	begz	a5.0x800006de <sbi init+82=""></sbi>		
00000000800006da:	ial	ra.0x800097ba <sbi hang="" hart=""></sbi>		
00000000800006de:	li	a0.85		
000000000800006e2:	ial	ra.0x80003c56 <misa extension="" imp=""></misa>		
00000000800006e6:	bnez	a0.0x80000872 <sbi init+486=""></sbi>		
00000000800006ea:	csrr	s6.mie		
00000000800006ee:	csrsi	mie.8		
00000000800006f2:	auipc	a0.0x18		
00000000800006f6	addi	a0.a01754 # 0x80018018 <coldboot lock=""></coldboot>		
00000000800006fa:	ial	ra.0x800049ae <spin lock=""></spin>		
00000000000000000000000000000000000000	srai	a5.s2.0x6		
000000000000000000000000000000000000000	slli	a5, a5, 0x3		
0000000000000702	auinc	s4.0x18		
0000000080000708	addi	\$4,\$4,-1460 # 0x80018150 <coldboot td="" wait<=""><td>hmasl</td><td></td></coldboot>	hmasl	
000000008000070c:	add	\$4, \$4, a5		
000000008000070e	ld	a5.0(s4)		
0000000080000712	li	\$5.1		