



基于Efinix FPGA的RISC-V 处理器开发入门指南

Bruce Chen 陈弘
+8613880416630
brucece@efinixinc.com

Richard Zhu 朱仁昌
+8613735539530
richardz@efinixinc.com

需要准备的工具

● 硬件开发环境

- Efinity开发软件
- Python 3.7 For Windows
- ModelSim仿真软件
- EFX RISC-V1.0硬件工程实例
- C232HM-DDHSL-D电缆

● 软件开发环境

- 集成开发环境GNU MCU Eclipse IDE
- 编译工具GNU GCC
- GNU嵌入工具链
- 调试工具OpenOCD
- 软件工程实例

● 在线调试工具

- RS232-TTL串口电缆
- Trion T20 BGA256开发板
- C232HM-DDHSL-D电缆



RISC-V软件开发环境配置操作指南

开发工具套件

- 集成开发环境GNU MCU Eclipse IDE
 - <https://github.com/gnu-mcu-eclipse/org.eclipse.epp.packages/release>
- 编译工具GNU GCC
 - <https://github.com/gnu-mcu-eclipse/windows-build-tools/releases>
- GNU 嵌入工具链
 - <https://www.sifive.com/boards>
- 调试工具OpenOCD
 - Efinix公司提供
- 工程实例
 - Efinix公司提供

工程实例

RISCV_10 > software > standalone



common



dhrystone



gpioDemo



gpioInterruptDemo



include



timerDemo



uartDemo

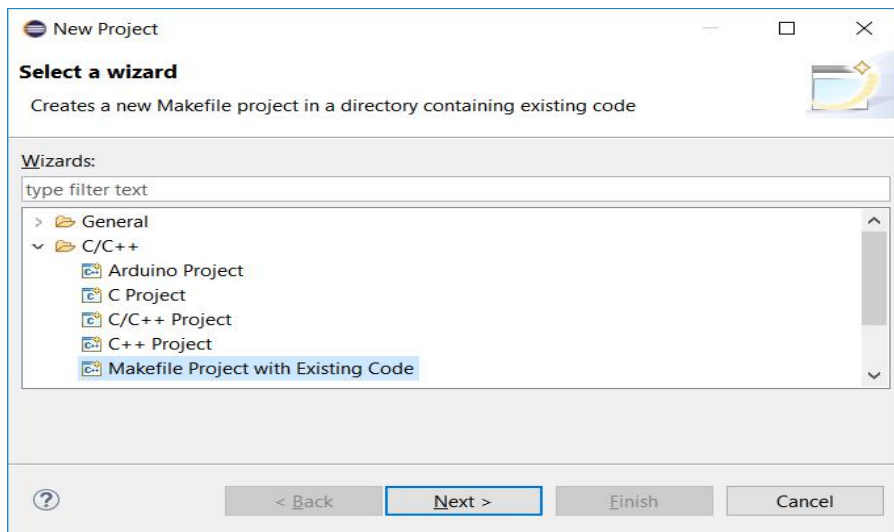
- Common
 - 链接和mk文件
- Dhrystone
- gpioDemo
- gpioInterruptDemo
- include
 - Library available
- timerDemo
- uartDemo

Eclipse的开发环境配置

- 新建项目
- 编译环境配置
- 调试环境配置

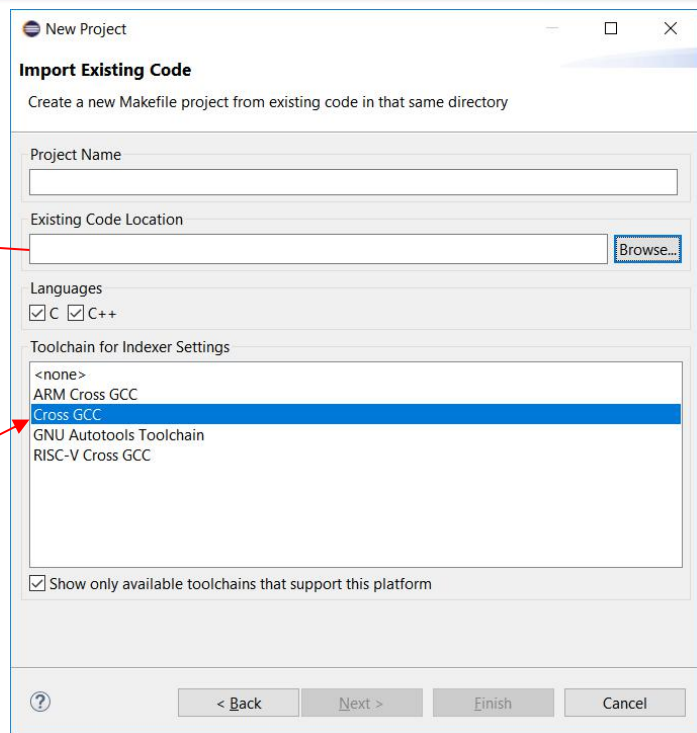
新建工程-选择工程类型

- 打开Eclipse IDE开发环境
 - 选择New -> Project
 - Makefile Project with Existing Code
- 点击Next



新建工程-选择源程序目录和编译工具

● 指定源代码目录 (本例为 gpioDemo)



● 指定编译工具

- Cross GGC

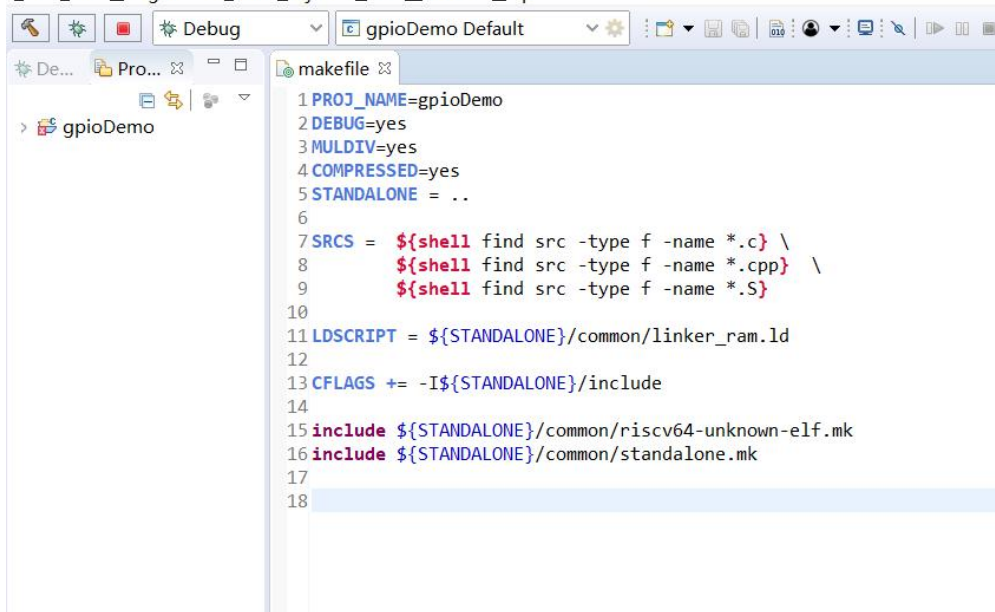
新建工程-编辑和检查makefile文件

● 双击makefile

○ 对比下图检查路径和内容

eclipse-workspace - gpioDemo/makefile - Eclipse IDE

File Edit Navigate Search Project Run Window Help



```
1 PROJ_NAME=gpioDemo
2 DEBUG=yes
3 MULDIV=yes
4 COMPRESSED=yes
5 STANDALONE = ..
6
7 SRCS = ${shell find src -type f -name *.c} \
8       ${shell find src -type f -name *.cpp} \
9       ${shell find src -type f -name *.S}
10
11 LDSCRIPT = ${STANDALONE}/common/linker_ram.ld
12
13 CFLAGS += -I${STANDALONE}/include
14
15 include ${STANDALONE}/common/riscv64-unknown-elf.mk
16 include ${STANDALONE}/common/standalone.mk
17
18
```

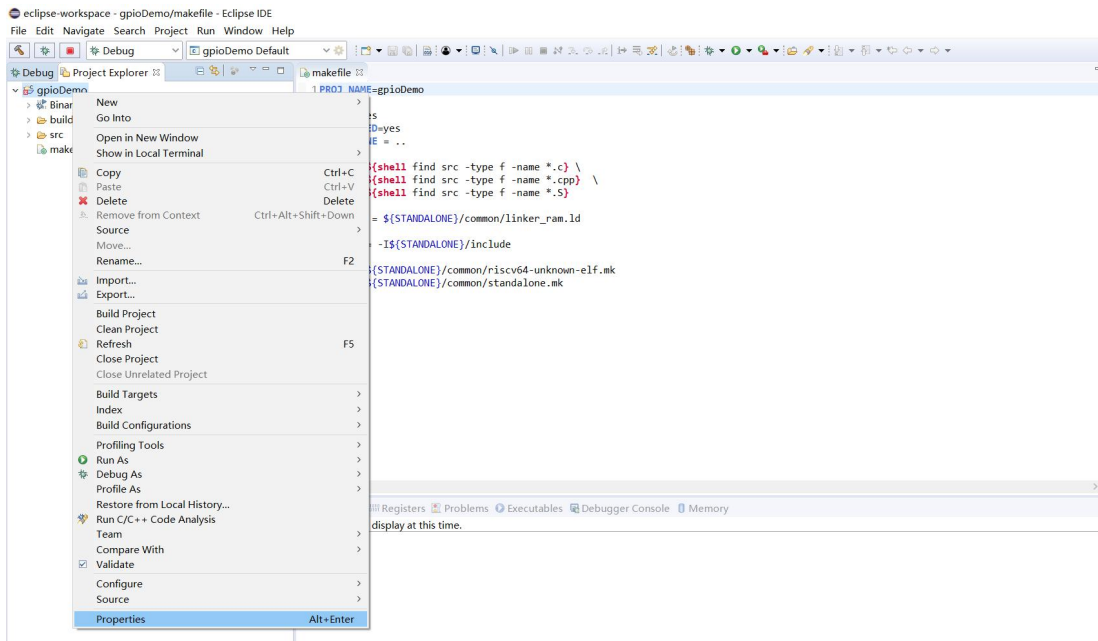
Eclipse的开发环境配置

- 新建项目
- 编译环境配置
- 调试环境配置

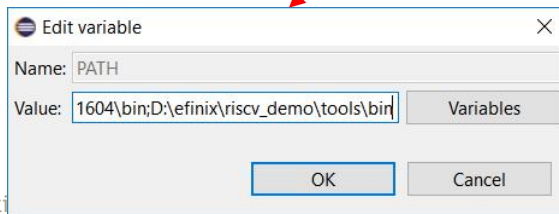
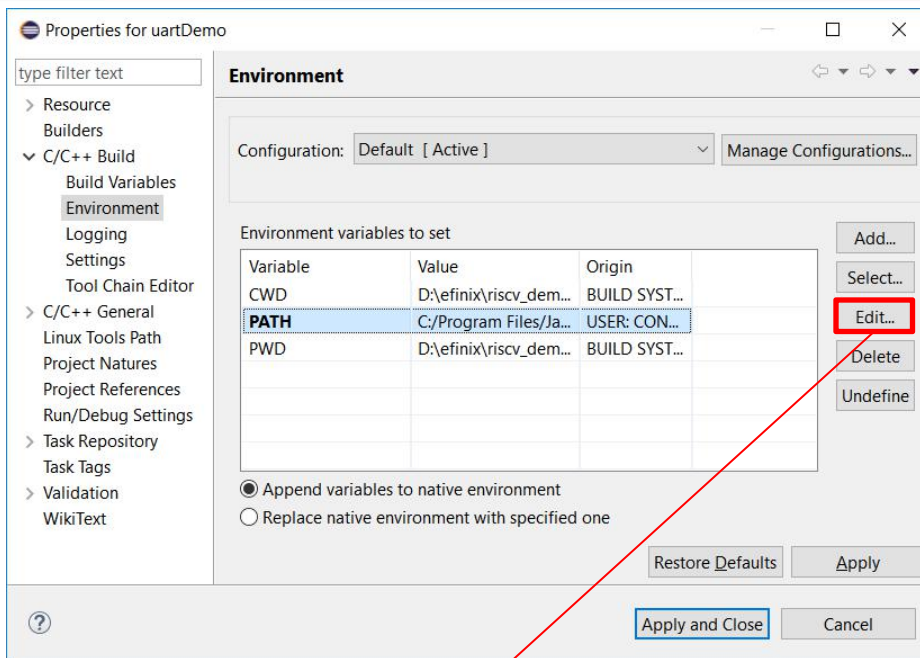
配置编译器路径(1)

选中工程 (本例为 gpioDemo)

- Right click
- Properties









配置编译器路径(2)









配置编译器路径(3)

添加GNU MCU Build Tool工具包路径

| DATA (D:) > efinix > riscv_demo > GNU MCU Eclipse > Build Tools > 2.11-20180428-1604 > bin | | | |
|---|----------------------|-------------|--------|
| Name | Date modified | Type | Size |
|  busybox.exe | 4/29/2018 12:08 A... | Application | 560 KB |
|  echo.exe | 4/29/2018 12:08 A... | Application | 560 KB |
|  make.exe | 4/29/2018 12:08 A... | Application | 234 KB |
|  mkdir.exe | 4/29/2018 12:08 A... | Application | 560 KB |
|  rm.exe | 4/29/2018 12:08 A... | Application | 560 KB |
|  sh.exe | 4/29/2018 12:08 A... | Application | 560 KB |

添加GNU GCC工具包路径

| DATA (D:) > efinix > riscv_demo > tools > bin | | | |
|---|-------------------|-----------------------|----------|
| Name | Date modified | Type | Size |
|  libexpat-1.dll | 3/6/2019 10:07 PM | Application extens... | 607 KB |
|  riscv64-unknown-elf-addr2line.exe | 3/6/2019 10:10 PM | Application | 1,037 KB |
|  riscv64-unknown-elf-ar.exe | 3/6/2019 10:10 PM | Application | 1,068 KB |
|  riscv64-unknown-elf-as.exe | 3/6/2019 10:10 PM | Application | 1,399 KB |
|  riscv64-unknown-elf-c++exe | 3/6/2019 11:38 PM | Application | 1,672 KB |
|  riscv64-unknown-elf-c++filt.exe | 3/6/2019 10:10 PM | Application | 1,032 KB |

备注：将包含上面两个工具包的文件夹路径添加到上一页PPT所示的Path环境变量里

Eclipse的开发环境配置

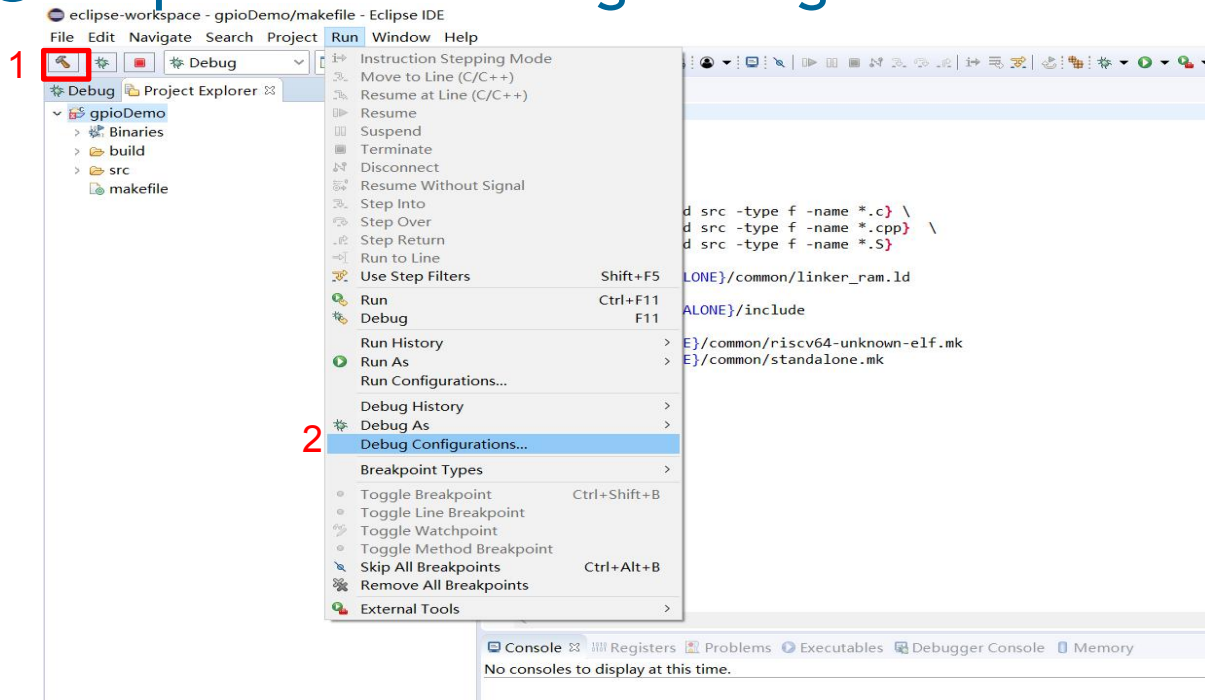
- 新建项目
- 编译环境配置
- 调试环境配置

调试环境配置(1)

● Step1 Build

● Step2 Run

Debug Configuration



调试环境配置(2)

● 添加调试工具和配置文件相关路径

Debug Configurations

Create, manage, and run configurations

Name: uartDemo Default

Main | Debugger | Startup | Source | Common | SVD Path

OpenOCD Setup

☒ Start OpenOCD locally

Executable path: D:\efinix\ riscv_demo\openocd_VexRiscv\bin\win64\openocd.exe

Actual executable: D:\efinix\ riscv_demo\openocd_VexRiscv\bin\win64\openocd.exe

(to change it use the [global](#) or [workspace](#) preferences pages or the [project](#) properties page)

GDB port: 3333

Telnet port: 4444

Tcl port: 6666

Config options: -f "D:\efinix\ riscv_demo\openocd_riscv-spinal\tcl\interface\ftdi\c232hm.cfg" -c 'set MURAX_CPU0_YAML D:\efinix\ riscv_demo\EFX_Riscv-master_20190514\cpu0.yaml' -f "D:\efinix\ riscv_demo\openocd_riscv-spinal\tcl\target\murax.cfg"

☒ Allocate console for OpenOCD ☐ Allocate console for the telnet connection

GDB Client Setup

☒ Start GDB session

Executable name: D:\efinix\ riscv_demo\tools\bin\riscv64-unknown-elf-gdb.exe

Actual executable: D:\efinix\ riscv_demo\tools\bin\riscv64-unknown-elf-gdb.exe

Other options:

Commands: set remotetimeout 60
set arch riscvrv32
set mem inaccessible-by-default off

Remote Target

Host name or IP address: localhost

Port number: 3333

☐ Force thread list update on suspend

Filter matched 15 of 19 items

Debug Configurations

1

2

3

Restore defaults

Revert Apply

Debug Close

调试环境配置(3)

Step1 OpenOCD 路径

Executable path: D:\efinix\riscv_demo\openocd_VexRiscv\bin\win64\openocd.exe





DATA (D:) > efinix > riscv_demo > openocd_VexRiscv > bin > win64

| Name | Date modified | Type | Size |
|---|--------------------|-------------|----------|
|  openocd.exe | 5/15/2019 12:00 PM | Application | 6,140 KB |

Step2 c232hm.cfg路径

Config options: -f "D:\efinix\riscv_demo\openocd_riscv-riscv_spinal\tcl\interface\ftdi\c232hm.cfg" -c 'set MURAX_CPU0_YAML D:\efinix\riscv_demo\EFX_Riscv-master_20190514\cpu0.yaml' -f "D:\efinix\riscv_demo\openocd_riscv-riscv_spinal\tcl\target\murax.cfg"

DATA (D:) > efinix > riscv_demo > openocd_riscv-riscv_spinal > tcl > interface >

| Name | Date modified | Type | Size |
|--|-------------------|---------------|------|
|  ft232r.cfg | 4/3/2019 12:06 AM | Text Document | 1 KB |
|  imx-native.cfg | 4/3/2019 12:06 AM | Text Document | 1 KB |
|  jlink.cfg | 4/3/2019 12:06 AM | Text Document | 1 KB |
|  jtag_tcp.cfg | 4/3/2019 12:06 AM | Text Document | 1 KB |

调试环境配置(4)

● Step3 cpu0.yaml 和 murax.cfg路径

Config options: -f "D:\efinix\riscv_demo\openocd_riscv-riscv_spinal\tcl\interface\ftdi\c232hm.cfg" -c 'set MURAX_CPU0_YAML D:\efinix\riscv_demo\EFX_Riscv-master_20190514\cpu0.yaml' -f "D:\efinix\riscv_demo\openocd_riscv-riscv_spinal\tcl\target\murax.cfg"

DATA (D:) > efinix > riscv_demo > EFX_Riscv-master_20190514 >

| Name | Date modified | Type | Size |
|-------------|-------------------|---------------|------|
| ext | 5/14/2019 6:26 PM | File folder | |
| hardware | 5/14/2019 6:26 PM | File folder | |
| project | 5/14/2019 6:26 PM | File folder | |
| software | 5/14/2019 6:26 PM | File folder | |
| test | 5/14/2019 6:26 PM | File folder | |
| .gitignore | 5/10/2019 7:26 PM | Text Document | 1 KB |
| .gitmodules | 5/10/2019 7:26 PM | Text Document | 1 KB |
| build.sbt | 5/10/2019 7:26 PM | SBT File | 1 KB |
| cpu0.yaml | 5/10/2019 7:26 PM | YAML File | 1 KB |

DATA (D:) > efinix > riscv_demo > openocd_riscv-riscv_spinal > tcl > target






| Name | Date modified | Type | Size |
|---------------|-------------------|---------------|------|
| murax.cfg | 4/3/2019 12:06 AM | Text Document | 1 KB |
| murax_xip.cfg | 4/3/2019 12:06 AM | Text Document | 1 KB |
| nds32v2.cfg | 4/3/2019 12:06 AM | Text Document | 1 KB |
| nds32v3.cfg | 4/3/2019 12:06 AM | Text Document | 1 KB |

调试环境配置(5)

● Step4 GDB 调试工具路径

Executable name: D:\efinix\riscv_demo\tools\bin\riscv64-unknown-elf-gdb.exe

· DATA (D:) > efinix > riscv_demo > tools > bin

| Name | Date modified | Type | Size |
|--|-------------------|-------------|----------|
|  riscv64-unknown-elf-gcc-ranlib.exe | 3/6/2019 11:38 PM | Application | 290 KB |
|  riscv64-unknown-elf-gcov.exe | 3/6/2019 11:38 PM | Application | 2,747 KB |
|  riscv64-unknown-elf-gcov-dump.exe | 3/6/2019 11:38 PM | Application | 1,000 KB |
|  riscv64-unknown-elf-gcov-tool.exe | 3/6/2019 11:38 PM | Application | 1,201 KB |
|  riscv64-unknown-elf-gdb.exe | 3/6/2019 11:57 PM | Application | 9,492 KB |

● 在Debug 配置菜单下执行Debug

Debug

Close

● 退出Debug配置点击继续运行

eclipse-workspace - uartDemo/src/main.c - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help





RISC-V硬件开发、仿真和固化操作指南

硬件开发环境

● 硬件开发所需相关软件和准备

- Efinity开发软件
- Python 3.7 For Windows
- ModelSim仿真软件
- 硬件工程实例
- C232HM-DDHSL-D电缆
- 软件工程编译生成的bin文件

硬件工程实例文件清单和说明



在Modelsim里仿真RISC-V

- Test bench代码：TestRiscVAXI.v
- RiscV处理器代码：EfxRiscvSoc.v
- RiscV顶层代码：EFX_Riscv.v
- 初始化头文件：Init_File.v

注意：打开Init_File.v，确保`define Simulation没有被注释掉，确保SoftBinFile正确路径

//需要固化程序，将“Simulation”注释掉

→ `define Simulation

//路径必须使用“/”

//SoftBinFile需要完整的路径名，可以采用绝对地址也可以采用相对地址

→ `define SoftBinFile "D:/Efinix/RISC_V/software/standalone/gpioDemo/build/gpioDemo.bin"
`define SoftHexFile0 "gpioDemo0.Hex"
`define SoftHexFile1 "gpioDemo1.Hex"
`define SoftHexFile2 "gpioDemo2.Hex"
`define SoftHexFile3 "gpioDemo3.Hex"

Modelsim仿真结果

对应于这一段C代码生成的BIN文件运行在RISC-V里，用Modelsim功能仿真产生的波形里可以看到GPIO计数值累加和UART串口的信号翻转。

```
void main() {  
    init();  
    uint32_t counter = 0;  
    uint32_t count_val=10;  
    uart_writeStr(UART_A, "EFX RiscV GPIO & UART Demo!\n\r");  
    while(1){  
        if(counter++ == count_val){  
            GPIO_A->OUTPUT = GPIO_A->OUTPUT + 1;  
            counter = 0;  
        }  
        if (uart_readOccupancy(UART_A)){  
            uart_read(UART_A);  
            uart_writeStr(UART_A, "Input a count value:");  
            count_val = uart_readuint(UART_A);  
            counter = 0;  
        }  
    }  
}
```



RISC-V ROM初始化文件生成

- ℓ 在固化RISC-V之前需要将软件生成的.BIN文件转化成FPGA编译需要的ROM初始化.HEX文件
- ℓ 由于在RISC-V的FPGA硬件设计里ROM被例化成了四块独立的Block RAM所以还需要对生成的初始化文件进行分割
- ℓ 文件类型转换和分割都由SoftwareBinToHex.py完成

//需要固化程序, 将"Simulation"注释掉

```
`define Simulation
```

//路径必须使用"/"

//SoftBinFile需要完整的路径名, 可以采用绝对地址也可以采用相对地址

软件生成的BIN文件的路径和文件名

Hex存储文件的路径和文件名

```
`define SoftBinFile      "D:/Efinix/RISC_V/software/standalone/gpioDemo/build/gpioDemo.bin"  
`define SoftHexFile0     "gpioDemo0.Hex"  
`define SoftHexFile1     "gpioDemo1.Hex"  
`define SoftHexFile2     "gpioDemo2.Hex"  
`define SoftHexFile3     "gpioDemo3.Hex"
```

如果这里没有设置或注释掉了, SoftwareBinToHex.py会使用代码中缺省路径或文件

Verilog的Include文件名

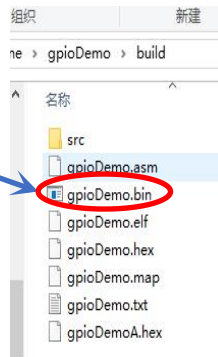
软件生成的BIN文件的路径和文件名

Hex存储文件的路径和文件名

```
InitFileName  = "Init_File.v"  
SrcFileName   = "gpioDemo.bin"  
TagFileName0  = "gpioDemo0.Hex"  
TagFileName1  = "gpioDemo1.Hex"  
TagFileName2  = "gpioDemo2.Hex"  
TagFileName3  = "gpioDemo3.Hex"
```

软件生成的BIN文件的路径

.settings
build
src
.cproject
.project
makefile



RISC-V ROM初始化文件生成步骤

- 打开硬件工程根目录下的Init_File.v，保证SoftBinFile路径指向软件工程生成的.BIN文件并将`define Simulation注释掉
- 双击运行RISC-V硬件工程目录下的SoftwareBinToHex.py
- 完成.bin文件到.hex文件的转换和分割

RISC-V ROM初始化文件生成结果

```
D:\Efinix\Design\RISC_V\EFX_RiscV_Axi\EFX_RiscV_Axi_V10>SoftwareBin2Hex.py
```

```
2019-06-11 15:47:45
```

```
Source Bin File : D:/Efinix/RISC_V/software/standalone/gpioDemo/build/gpioDemo.bin
```

```
Target Hex File : gpioDemo0.Hex
```

```
Target Hex File : gpioDemo1.Hex
```

```
Target Hex File : gpioDemo2.Hex
```

```
Target Hex File : gpioDemo3.Hex
```

```
Conversion success!!!
```

用Efinity编译RISC-V处理器工程

- 进入Efinity，打开EFX_RiscV_Axi.xml工程
- 选择生成SPI Passive比特文件
 - 选择PS模式可实现FPGA PS加载和CPU JTAG调试接口复用
 - 也可选择生成SPI Active比特文件通过T20开发板板载USB电缆以任意模式加载FPGA
- 运行SoftwareBinToHex.py，并编译整个工程生成FPGA比特流文件

注意：

- 1、运行SoftwareBinToHex.py之前，保证Init_File.v和SoftwareBinToHex.py在工程目录下；
- 2、打开Init_File.v，注释掉`define Simulation（不然编译会出错），检查并确保SoftBinFile正确路径

```
1 //需要固化程序，将“Simulation”注释掉
2
3
4 //`define Simulation
5
6 //路径必须使用“/”
7 //文件定义需要完整的路径名，可以采用绝对地址也可以采用相对地址
8
9 `define SoftBinFile "D:/RISCV/RiscV_v1.0/software/standalone/gpioDemo/build/gpioDemo.bin"
10 `define SoftHexFile0 "gpioDemo0.Hex"
11 `define SoftHexFile1 "gpioDemo1.Hex"
12 `define SoftHexFile2 "gpioDemo2.Hex"
13 `define SoftHexFile3 "gpioDemo3.Hex"
```

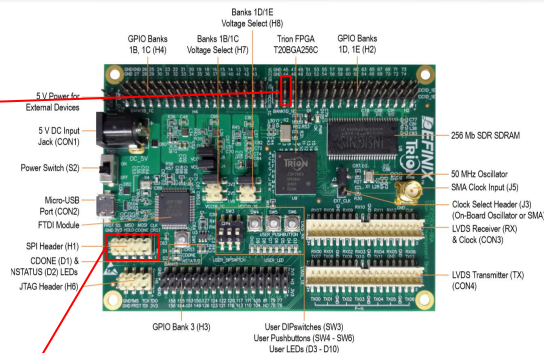
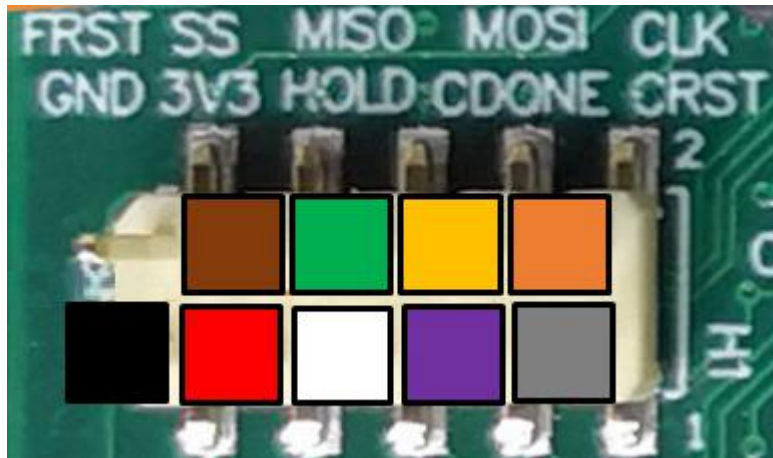
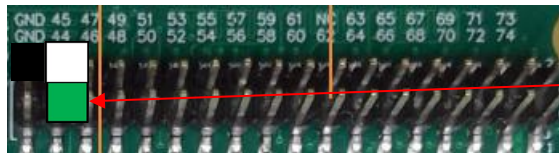
编译结果

资源使用情况

性能

| Core Resource | |
|----------------|--------------|
| Inputs | 14 / 470 |
| Outputs | 18 / 595 |
| Clocks | 2 / 16 |
| Logic Elements | 3968 / 19728 |
| Memory Blocks | 70 / 204 |
| Multipliers | 4 / 36 |
| Timing | |
| Least Slack | -0.656 ns |
| io_axiClk | 63.193 MHz |
| io_jtag_tck | 106.475 MHz |

C232HM/RS232-TTL电缆和T20F256开发板连接示意图



● 在线调试工具

- RS232-TTL串口电缆
- Trion T20 BGA256开发板
- C232HM-DDHSL-D电缆

FPGA PS加载和CPU JTAG调试接口复用说明

由FTDI C232HM电缆JTAG模式和SPI模式下管脚定义可见，只要在设计中将CPU的JTAG相关管脚锁定到对应的FPGA PS模式下载管脚上，即可实现CPU的JTAG调试接口与FPGA PS加载的接口复用。

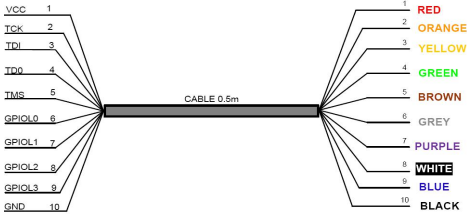
| | | FTDI | | Flash | | FPGA | | RISC-V |
|----|--------|----------|---------|-----------|-----------|--------------|----------------|-------------|
| NO | Color | JTAG Pin | SPI Pin | SPI Flash | Conf. Pin | T20 GPIO | T20 Demo Board | Pin name |
| 1 | RED | VCC | VCC | VCC | VCC | VCC | VCC | |
| 2 | ORANGE | TCK | SK | SK | CCK | GPIO_01_CCK | CLK | io_jtag_tck |
| 3 | YELLOW | TDI | DO | SDI | CDI0 | GPIO_08_CDI0 | MOSI | io_jtag_tdi |
| 4 | GREEN | TDO | DI | SDO | CDI1 | GPIO_09_CDI1 | MISO | io_jtag_tdo |
| 5 | BROWN | TMS | CS | CS_B | SS_N | GPIO_00_SS | SS | io_jtag_tms |
| 6 | GREY | GPIO0 | GPIO0 | | Creset | | CRST | |
| 7 | PURPLE | GPIO1 | GPIO1 | | ConfDone | | CDONE | |
| 8 | WHITE | GPIO2 | GPIO2 | | | | | |
| 9 | BLUE | GPIO3 | GPIO3 | | | | | |
| 10 | BLACK | GND | GND | GND | GND | GND | GND | GND |

| Colour | Pin Number | Name | Type | Description |
|--------|------------|------|--------|--------------------|
| Orange | 2 | SK | Output | Serial Clock |
| Yellow | 3 | DO | Output | Serial data output |
| Green | 4 | DI | Input | Serial Data Input |
| Brown | 5 | CS | Output | Serial Chip Select |

Table 3.3 MPSSE Option SPI - Signal Descriptions

| Colour | Pin Number | Name | Type | Description |
|--------|------------|------|--------|----------------------|
| Orange | 2 | TCK | Output | Test Interface Clock |
| Yellow | 3 | TDI | Input | Test Data Input |
| Green | 4 | TDO | Output | Test Data Output |
| Brown | 5 | TMS | Output | Test Mode Select |

Table 3.2 MPSSE Option JTAG - Signal Descriptions



注意：CPU的JTAG调试口的四个管脚是不同于FPGA下载专用JTAG口的，可以锁定在FPGA的任意可用IO上。FPGA的PS模式加载管脚在加载完成以后可以作为IO使用。这里将CPU的JTAG管脚锁定到FPGA的SPI管脚上，实现同一根电缆PS加载FPGA完成后到CPU JTAG调试无缝转换。

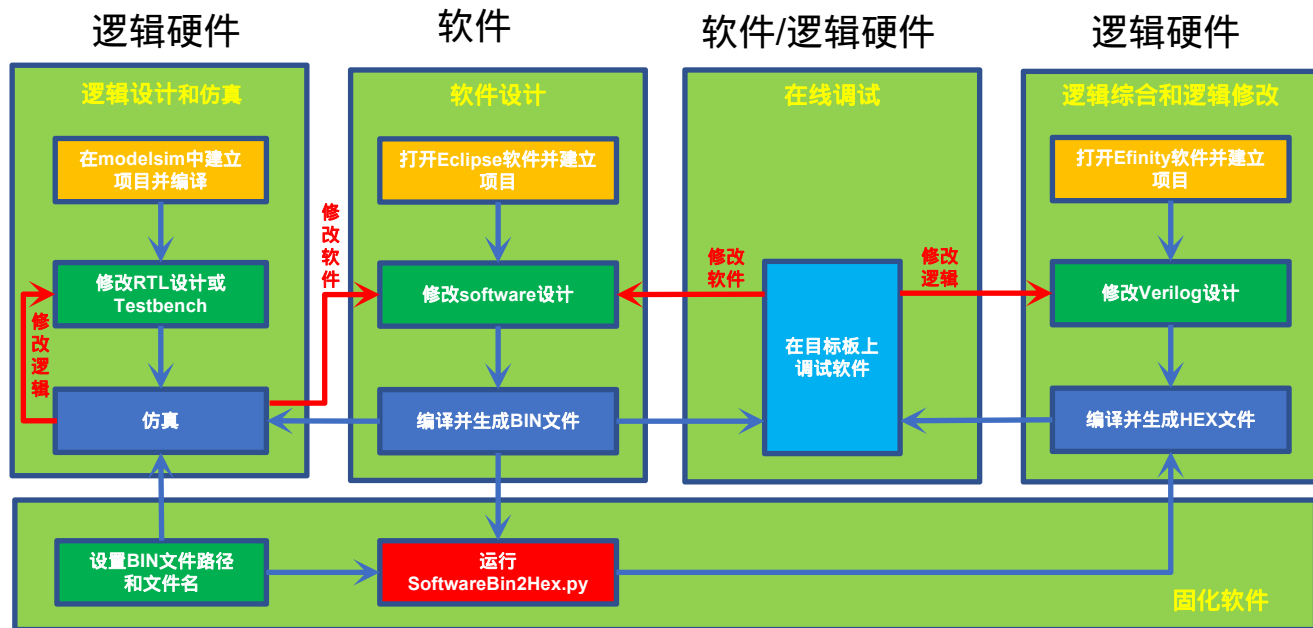
PS模式下载FPGA配置文件

- 按照下表对应关系连接C232HM-DDHSL-D电缆和T20 开发板SPI插座
- 打开Programmer选择PS模式下载
- 下载完成以后可以观察到USER_LED D3-D10按C程序的计数闪烁
- 下载成功后，如果看不到闪烁，原因是C程序计数间隔初始值太小，可通过调整软件工程里main.c里counter_val的初始值重新编译软件工程生成新的bin文件，重复此前的步骤来观察到直观的计数闪烁，也可以通过连接串口直接修改计数值来观察LED计数闪烁的变化

| FTDI | | | | Flash | FPGA | | | RISC-V |
|------|--------|----------|---------|-----------|-----------|--------------|----------------|-------------|
| NO | Color | JTAG Pin | SPI Pin | SPI Flash | Conf. Pin | T20 GPIO | T20 Demo Board | Pin name |
| 1 | RED | VCC | VCC | VCC | VCC | VCC | VCC | VCC |
| 2 | ORANGE | TCK | SK | SCK | CCK | GPIO_01_CCK | CLK | io_jtag_tck |
| 3 | YELLOW | TDI | DO | SDI | CDI0 | GPIO_08_CDI0 | MOSI | io_jtag_tdi |
| 4 | GREEN | TDO | DI | SDO | CDI1 | GPIO_09_CDI1 | MISO | io_jtag_tdo |
| 5 | BROWN | TMS | CS | CS_B | SS_N | GPIO_00_SS | SS | io_jtag_tms |
| 6 | GREY | GPIO0 | GPIO0 | | Creset | | CRST | |
| 7 | PURPLE | GPIO1 | GPIO1 | | ConfDone | | CDONE | |
| 8 | WHITE | GPIO2 | GPIO2 | HOLD | | | HOLD | |
| 9 | BLUE | GPIO3 | GPIO3 | | | | | |
| 10 | BLACK | GND | GND | GND | GND | GND | GND | GND |

```
void main() {
    init();
    uint32_t counter = 0;
    uint32_t count_val=10;
    uart_writeStr(UART_A, "EFX Riscv GPIO & UART Demo!\n\r");
    while(1){
        if(counter++ == count_val){
            GPIO_A->OUTPUT = GPIO_A->OUTPUT + 1;
            counter = 0;
        }
        if (uart_readOccupancy(UART_A)){
            uart_read(UART_A);
            uart_writeStr(UART_A, "Input a count value:");
            count_val = uart_readuint(UART_A);
            counter = 0;
        }
    }
}
```


RISC-V软件开发交互流程



技术支持联系方式



<https://www.efinixinc.com/support/login.php>

Bruce Chen 陈弘
+8613880416630
brucec@efinixinc.com

Richard Zhu 朱仁昌
+8613735539530
richardz@efinixinc.com

谢谢！