



Microsemi AC490 RTG4 FPGA: Building a Mi-V Processor Subsystem User Guide

[Home](#) » [Microsemi](#) » Microsemi AC490 RTG4 FPGA: Building a Mi-V Processor Subsystem User Guide 

Contents

- 1 [Microsemi AC490 RTG4 FPGA: Building a Mi-V Processor Subsystem](#)
- 2 [Building a Mi-V Processor Subsystem](#)
 - 2.1 [Design Description](#)
 - 2.2 [Software Implementation](#)
- 3 [Setting Up the Hardware](#)
- 4 [Running the Demo](#)
- 5 [Programming the Device Using Libero SoC](#)
- 6 [Running the TCL Script](#)
- 7 [Documents / Resources](#)
 - 7.1 [References](#)
- 8 [Related Posts](#)



Microsemi AC490 RTG4 FPGA: Building a Mi-V Processor Subsystem



Revision History

The revision history describes the changes that were implemented in the document. The changes are listed by revision, starting with the most current publication.

Revision 3.0

The following is a summary of the changes made in this revision.

- Updated the document for Libero SoC v2021.2.
- Updated Figure 1, page 3 through Figure 3, page 5.
- Replaced Figure 4, page 5, Figure 5, page 7, and Figure 18, page 17.
- Updated Table 2, page 6 and Table 3, page 7.
- Added Appendix 1: Programming the Device Using FlashPro Express, page 14.
- Added Appendix 3: Running the TCL Script, page 20.
- Removed the references to Libero version numbers.

Revision 2.0

The following is a summary of changes made in this revision.

- Added information about the COM port selection in Setting Up the Hardware, page 9.
- Updated how to select the appropriate COM port in Running the Demo, page 11.

Revision 1.0

The first publication of the document.

Building a Mi-V Processor Subsystem

Microchip offers the Mi-V processor IP, a 32-bit RISC-V processor and software toolchain to develop RISC-V processor based designs. RISC-V, a standard open Instruction Set Architecture (ISA) under the governance of the RISC-V Foundation, offers numerous benefits, which include enabling the open source community to test and improve cores at a faster pace than closed ISAs.

RTG4® FPGAs support Mi-V soft processor to run user applications. This application note describes how to build a Mi-V processor subsystem to execute a user application from the designated fabric RAMs or DDR memory.

Design Requirements

The following table lists the hardware and software requirements for running the demo.

Table 1 • Design Requirements

Requirement	Version
Hardware	
RTG4 Development Kit	Rev B
• RTG4 Development Board with RT4G150 - 1CG 1657PROTO FPGA	
• 12V, 5A AC power adapter	
• USB A to mini-B cable	
Host PC or Laptop	64-bit Windows 7 and 10

Software

- Libero® System-on-Chip (SoC)
- FlashPro Express
- SoftConsole

Note: Refer to the readme.txt file provided in the design files for the software versions used with this reference design.

Note: Libero SmartDesign and configuration screen shots shown in this guide are for illustration purpose only. Open the Libero design to see the latest updates.

Prerequisites

Before you start:

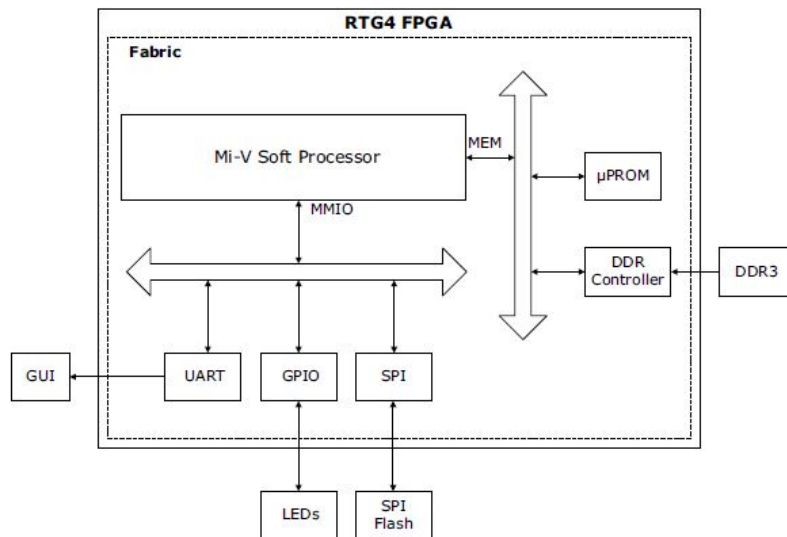
1. Download and install Libero SoC (as indicated in the website for this design) on the host PC from the following location: <https://www.microsemi.com/product-directory/design-resources/1750-libero-soc>
2. For demo design files download link: http://soc.microsemi.com/download/rsc/?f=rtg4_ac490_df

Design Description

The size of RTG4 μ PROM is 57 KB. User applications that do not exceed the μ PROM size can be stored in μ PROM and executed from internal Large SRAM memories (LSRAM). User applications that exceed the μ PROM size must be stored in an external non-volatile memory. In this case, a bootloader executing from μ PROM is required to initialize internal or external SRAM memories with the target application from the non-volatile memory. The reference design demonstrates the bootloader capability to copy the target application (of size 7 KB) from SPI flash to DDR memory, and execute from the DDR memory. The bootloader is executed from internal memories. The code section is located in μ PROM, and the data section is located in the internal Large SRAM (LSRAM).

Note: For more information about how to build the Mi-V bootloader Libero project and how to the build SoftConsole project, refer to TU0775: PolarFire FPGA: Building a Mi-V Processor Subsystem Tutorial. Figure 1 shows the top-level block diagram of the design.

Figure 1 • Top Level Block Diagram



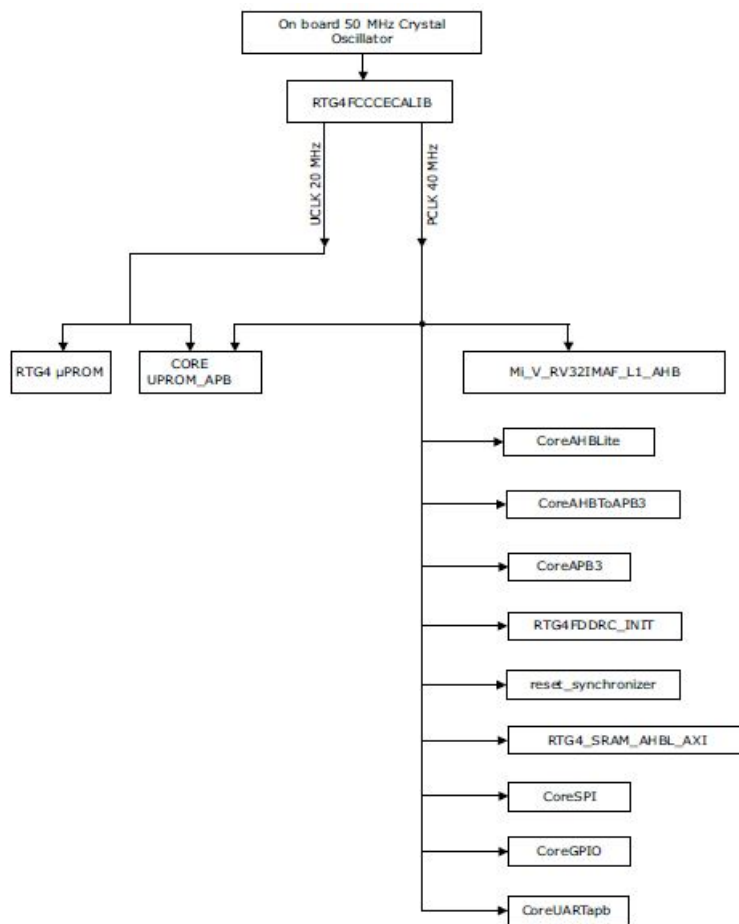
As shown in Figure 1, the following points describe the data flow of the design:

- The Mi-V processor executes the bootloader from the μ PROM and designated LSRAMs. The bootloader interfaces with the GUI through the CoreUartApb block and waits for the commands.
- When the SPI flash program command is received from the GUI, the bootloader programs the SPI flash with the target application received from the GUI.
- When the boot command is received from the GUI, the bootloader copies the application code from the SPI flash to DDR and then executes it from DDR.

Clocking Structure

There are two clock domains (40 MHz and 20 MHz) in the design. The on-board 50 MHz crystal oscillator is connected to the PF_CCC block which generates 40 MHz and 20 MHz clocks. The 40 MHz system clock drives the complete Mi-V processor subsystem except μ PROM. The 20 MHz clock drives the RTG4 μ PROM and the RTG4 μ PROM APB interface. RTG4 μ PROM supports a clock frequency of up to 30 MHz. DDR_FIC is configured for the AHB bus interface, which operates at 40 MHz. The DDR memory operates at 320 MHz. Figure 2 shows the clocking structure.

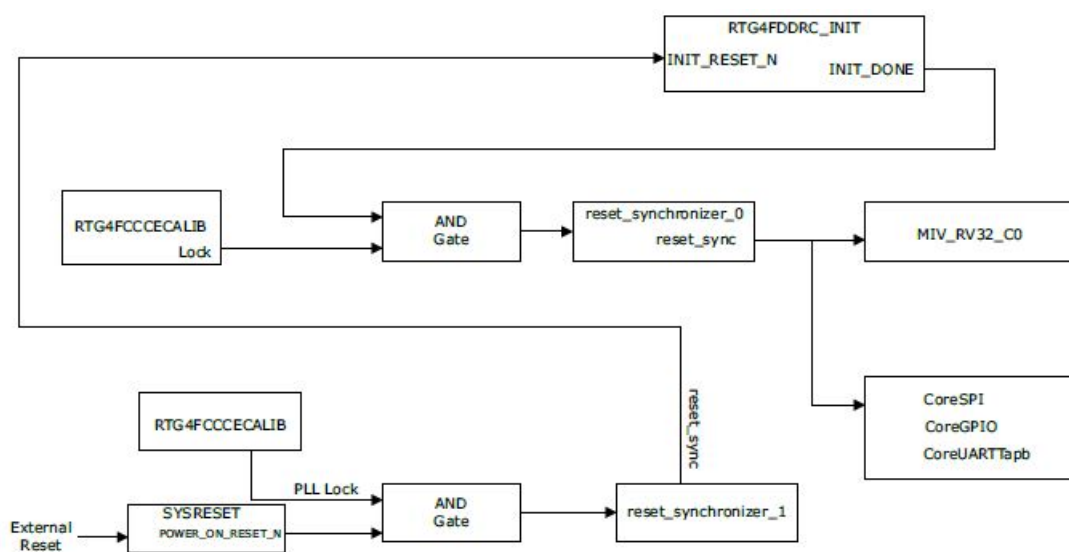
Figure 2 • Clocking Structure



Reset Structure

The POWER_ON_RESET_N and the LOCK signals are ANDed, and the output signal (INIT_RESET_N) is used to reset the RTG4FDDRC_INIT block. After releasing the FDDR reset, the FDDR controller gets initialized, and then the INIT_DONE signal is asserted. The INIT_DONE signal is used to reset the Mi-V processor, peripherals, and other blocks in the design.

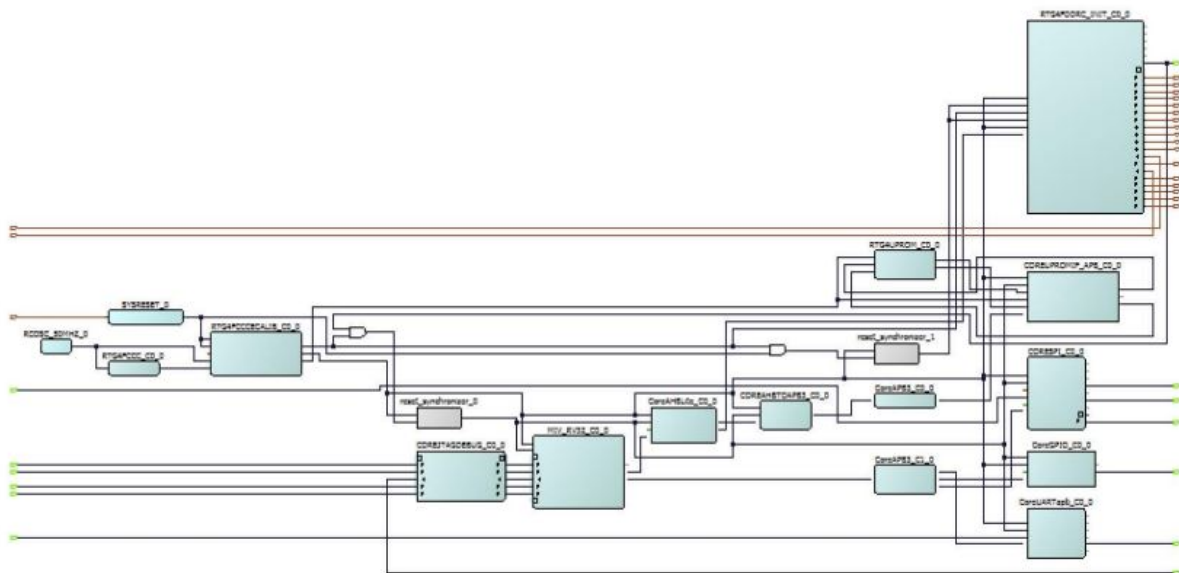
Figure 3 • Reset Structure



Hardware Implementation

Figure 4 shows the Libero design of the Mi-V reference design.

Figure 4 • SmartDesign Module



Note: Libero SmartDesign screenshot shown in this application note is for illustration purpose only. Open the Libero project to see the latest updates and IP versions.

IP Blocks

Figure 2 list the IP blocks used in the Mi-V processor subsystem reference design and their function.

Table 2 • IP Blocks1

IP Name	Function
MIV_RV32_C0	Mi-V soft processor.
CoreJTAGDEBUG	Facilitates the connection of Joint Test Action Group (JTAG) compatible soft core processors to the JTAG header for debugging. It provides fabric access to the JTAG interface using UJTAG macro.
CoreAHBLite	Multi-master AHB-Lite bus.
CoreAHBtoAPB3	Bridge between AHB master and APB slave.
CoreUARTapb, CoreSPI, and CoreGPIO	UART, SPI, and GPIO controllers with APB interface.
RTG4FCCCECALIB	Macro to access RTG4 CCC block. It is used to synthesize 55.5 MHz and 27.7 MHz clock frequencies from the CCC with an on-board 50 MHz reference clock.
RTG4UPROM	Used for storing the bootloader program.
RTG4UPROMIF_APB	CoreUPROMIF_APB is an APB wrapper core that provides read-only access to the μ PROM memory block within the RTG4 fabric via the APB interface. This facilitates easy access to the μ PROM for APB masters.
RTG4FDDRC_INIT	The Fabric External Memory DDR (FDDR) Configurator is used to configure the external DDR memory parameters.
CoreAPB3	CoreAPB3 is an advanced microcontroller bus architecture (AMBA) 3 advanced peripheral bus (APB) fabric for interconnecting between an APB master and up to 16 APB slaves.

All the IP user guides and handbooks are available from Libero SoC -> Catalog.

RTG4 μ PROM stores up to 10,400 36-bit words (374,400 bits of data). It supports only read operations during normal device operation after the device is programmed. The MIV_RV32_C0 processor core comprises an instruction fetch unit, an execution pipeline, and a data memory system. The MIV_RV32_C0 processor memory system includes instruction cache and data cache. The MIV_RV32_C0 core includes two external AHB interfaces- the AHB memory (MEM) bus master interface and the AHB Memory Mapped I/O (MMIO) bus master interface. The cache controller uses the AHB MEM interface to refill the instructions and the data caches. The AHB MMIO interface is used for an uncached access to I/O peripherals.

The memory maps of the AHB MMIO interface and the MEM interface are 0x60000000 to 0x6FFFFFFF and 0x80000000 to 0x8FFFFFFF, respectively. The processor's reset vector address is configurable. The MIV_RV32_C0's reset is an active-low signal, which must be de-asserted in sync with the system clock through a reset synchronizer.

The MIV_RV32_C0 processor accesses the application execution memory using the AHB MEM interface. The CoreAHBLite_C0_0 bus instance is configured to provide 16 slave slots, each of size 1 MB. The RTG μ PROM memory, and RTG4FDDRC blocks are connected to this bus. The μ PROM is used for storing the bootloader application.

The MIV_RV32_C0 processor directs the data transactions between addresses 0x60000000 and 0x6FFFFFFF to the MMIO interface. The MMIO interface is connected to the CoreAHBLite_C1_0 bus to communicate with peripherals connected to its slave slots. The CoreAHBLite_C1_0 bus instance is configured to provide 16 slave slots, each of size 256 MB. The UART, CoreSPI, and CoreGPIO peripherals are connected to the CoreAHBLite_C1_0 bus via the CoreAHBTOAPB3 bridge and the CoreAPB3 bus.

Memory Map

Table 3 lists the memory map of the memories and peripherals.

Table 3 • Memory Map

Peripherals	Start Address
TCM	0x70000000
μ PROM	0x80100000
DDR3	0x80200000
UART	0x60000000
GPIO	0x60001000
SPI	0x60002000

Software Implementation

The reference design files include the SoftConsole workspace that contains the following software projects:

- Bootloader
- Target Application

Bootloader

The bootloader application is programmed on the μ PROM during device programming. The bootloader implements the following functions:

- Programming the SPI Flash with the target application.
- Copying the target application from SPI Flash to DDR3 memory.
- Switching the program execution to the target application available in DDR3 memory.

The bootloader application must be executed from μ PROM with LSRAM as stack. Hence, the addresses of ROM and RAM in the linker script are set to the starting address of μ PROM and designated LSRAMs, respectively. The code section is executed from ROM and data section is executed from RAM as shown in Figure 5.

Figure 5 • Bootloader Linker Script

```

MEMORY
{
    envm (rx) : ORIGIN = 0x80100000, LENGTH = 32k
    ram (rwx) : ORIGIN = 0x70000000, LENGTH = 64k
}

RAM_START_ADDRESS = 0x70000000; /* Must be the same value MEMORY region
RAM_SIZE          = 64k;         /* Must be the same value MEMORY region
STACK_SIZE        = 4k;         /* needs to be calculated for your application
HEAP_SIZE         = 0k;         /* needs to be calculated for your application

```

The linker script (microsemi-riscv-ram_rom.ld) is available at the SoftConsole_Project\mivrv32im-bootloader folder of the design files.

Target Application

The target application blinks the onboard LEDs 1, 2, 3, and 4 and prints UART messages. The target application must be executed from DDR3 memory. Hence, the code and stack sections in the linker script are set to the starting address of DDR3 memory as shown in Figure 6.

Figure 6 • Target Application Linker Script

```

20 MEMORY
21 {
22     ram (rwx) : ORIGIN = 0x80200000, LENGTH = 64k
23 }
24
25 RAM_START_ADDRESS = 0x80200000; /* Must be the same value MEMORY region ram ORIGIN above. */
26 RAM_SIZE          = 64k;         /* Must be the same value MEMORY region ram LENGTH above. */
27 STACK_SIZE        = 4k;         /* needs to be calculated for your application */
28 HEAP_SIZE         = 0k;         /* needs to be calculated for your application */
--

```

The linker script (microsemi-riscv-ram.ld) is available at the SoftConsole_Project\miv-rv32imddr- application folder of the design files.

Setting Up the Hardware

The following steps describe how to set up the hardware:

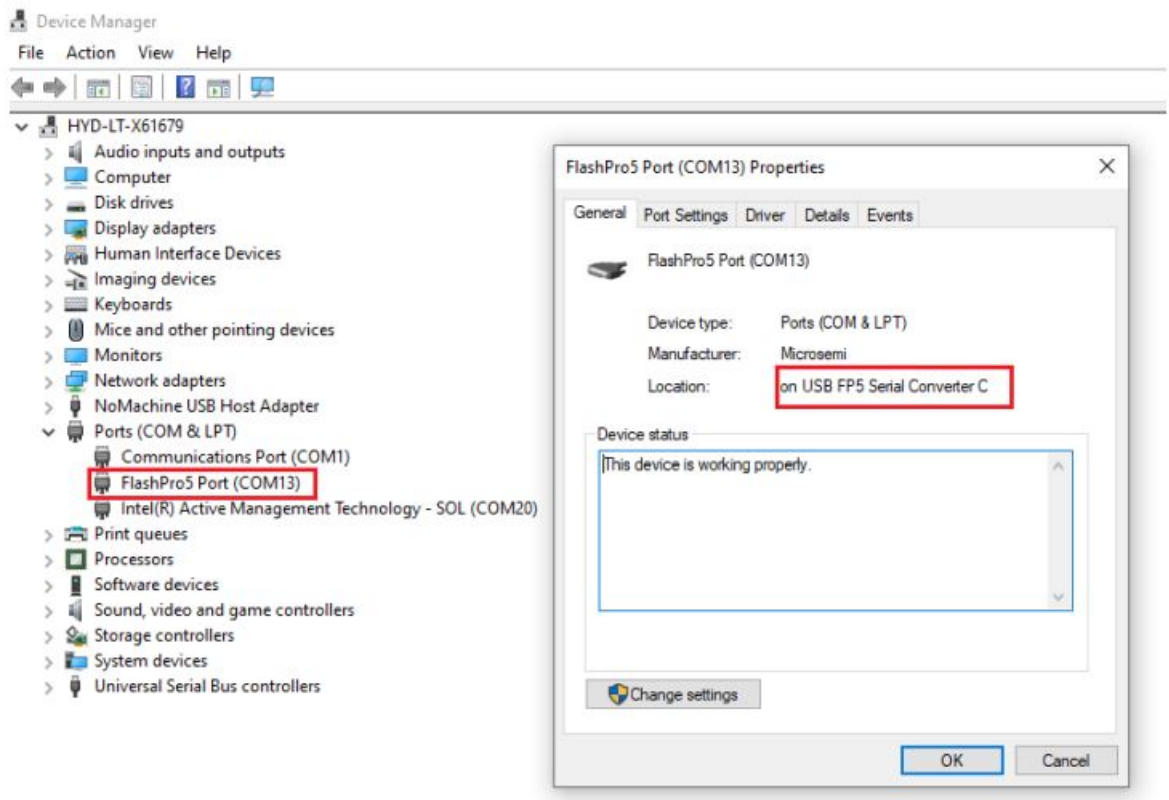
1. Ensure that the board is powered OFF using the SW6 switch.
2. Connect the jumpers on the RTG4 development kit, as shown in the following table:

Table 4 • Jumpers

Jumper	Pin From	Pin To	Comments
J11, J17, J19, J23, J26, J21, J32, and J27	1	2	Default
J16	2	3	Default
J33	1	2	Default
	3	4	

3. Connect the host PC to the J47 connector using the USB cable.
4. Ensure that the USB to UART bridge drivers are automatically detected. This can be verified in the device manager of the host PC.
5. As shown in Figure 7, the port properties of COM13 show that it is connected to USB Serial Converter C. Hence, COM13 is selected in this example. The COM port number is system specific.

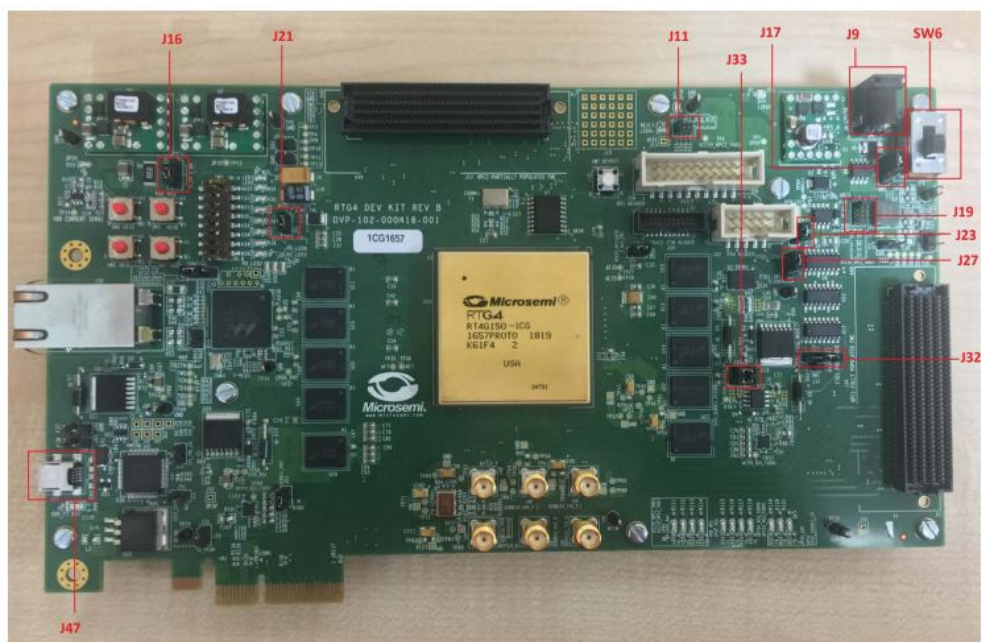
Figure 7 • Device Manager



Note: If the USB to UART bridge drivers are not installed, download and install the drivers from www.microsemi.com/documents/CDM_2.08.24_WHQL_Certified.zip.

6. Connect the power supply to J9 connector and switch ON the power supply switch, SW6.

Figure 8 • RTG4 Development Kit



Running the Demo

This chapter describes steps to program the RTG4 device with the reference design, programming the SPI Flash with the target application, and booting the target application from DDR memory using the Mi-V Bootloader GUI.

Running the demo involves the following steps:

1. Programming the RTG4 Device, page 11
2. Running the Mi-V Bootloader, page 11

Programming the RTG4 Device

The RTG4 device can be programmed either using FlashPro Express or Libero SOC.

- To program the RTG4 Development Kit with the job file provided as part of the design files using FlashPro Express software, refer to Appendix 1: Programming the Device Using FlashPro Express, page 14.
- To program the device using Libero SoC, refer to Appendix 2: Programming the Device Using Libero SoC, page 17.

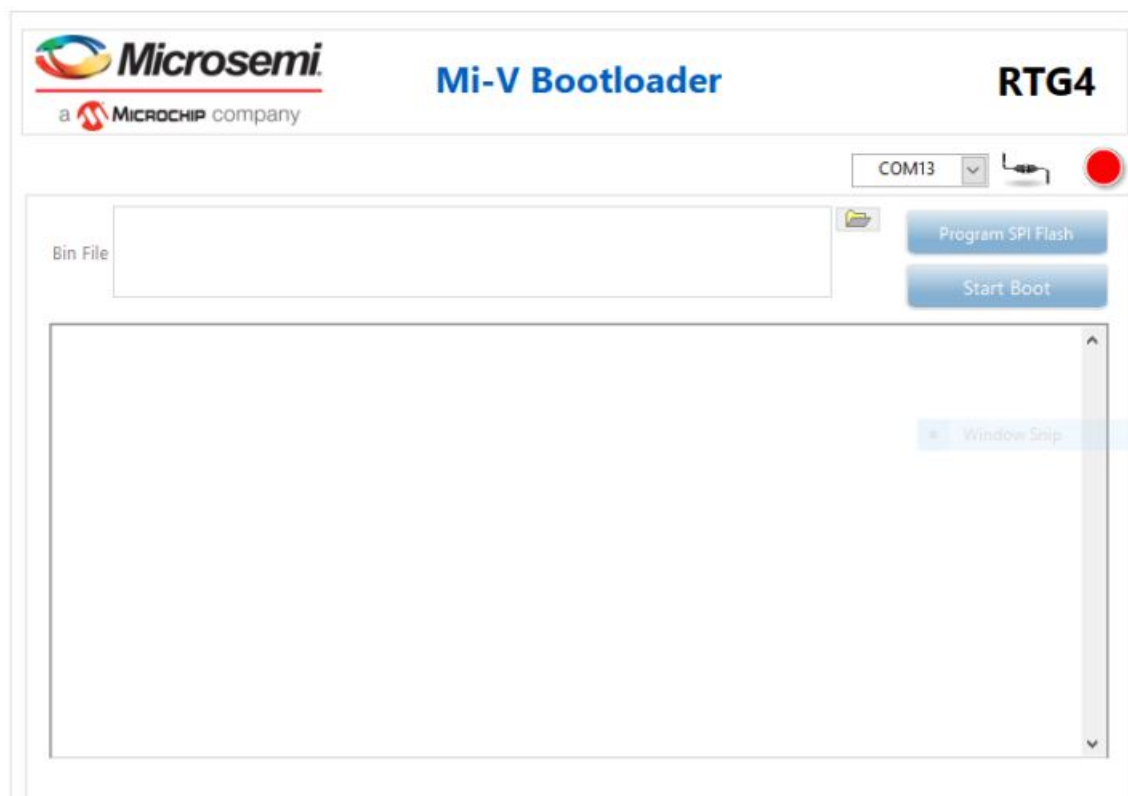
Running the Mi-V Bootloader

On successful completion of programming, follow these steps:

1. Run the setup.exe file available at the following design files location.
<\$Download_Directory>\rtg4_ac490_df\GUI_Installer\Mi-V Bootloader_Installer_V1.4
2. Follow the installation wizard to install the Bootloader GUI application.

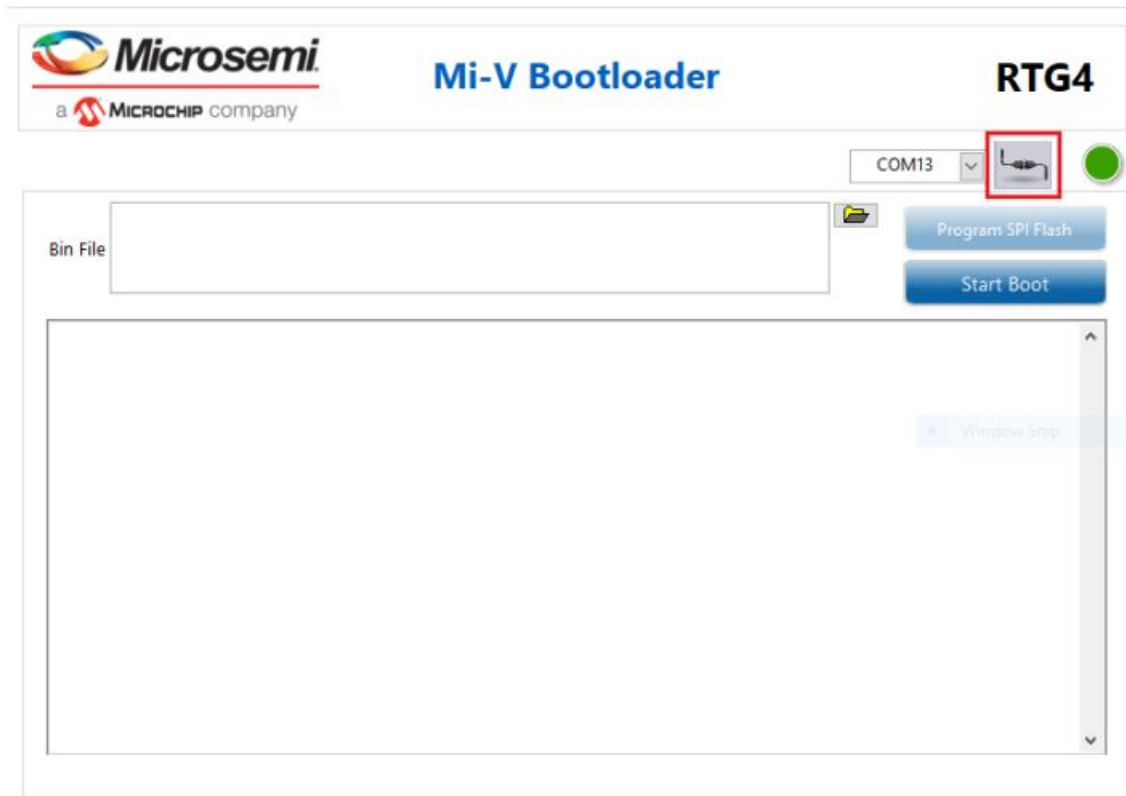
Figure 9 shows the RTG4 Mi-V Bootloader GUI.

Figure 9 • Mi-V Bootloader GUI



3. Select the COM port connected to USB Serial Converter C as shown in Figure 7.
4. Click the connect button. After successful connection the Red indicator turns Green as shown in Figure 10.

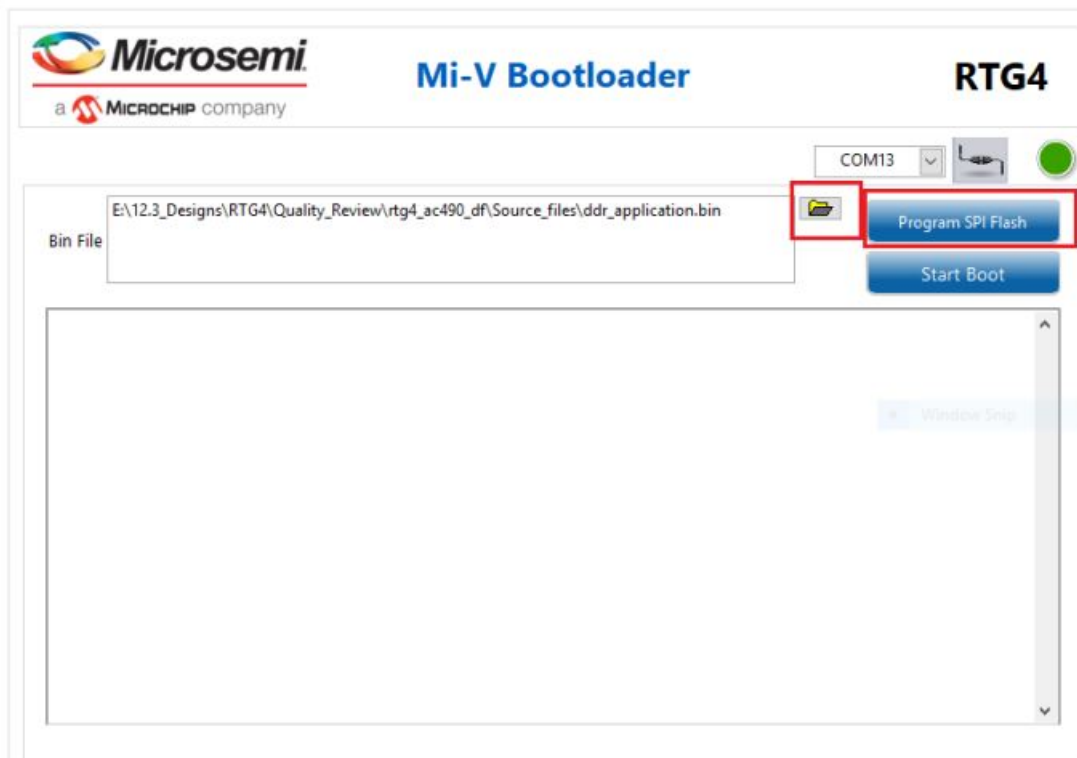
Figure 10 • Connect COM Port



5. Click the Import button and select the target application file (.bin). After importing, the path of the file is displayed on the GUI as shown in Figure 11.

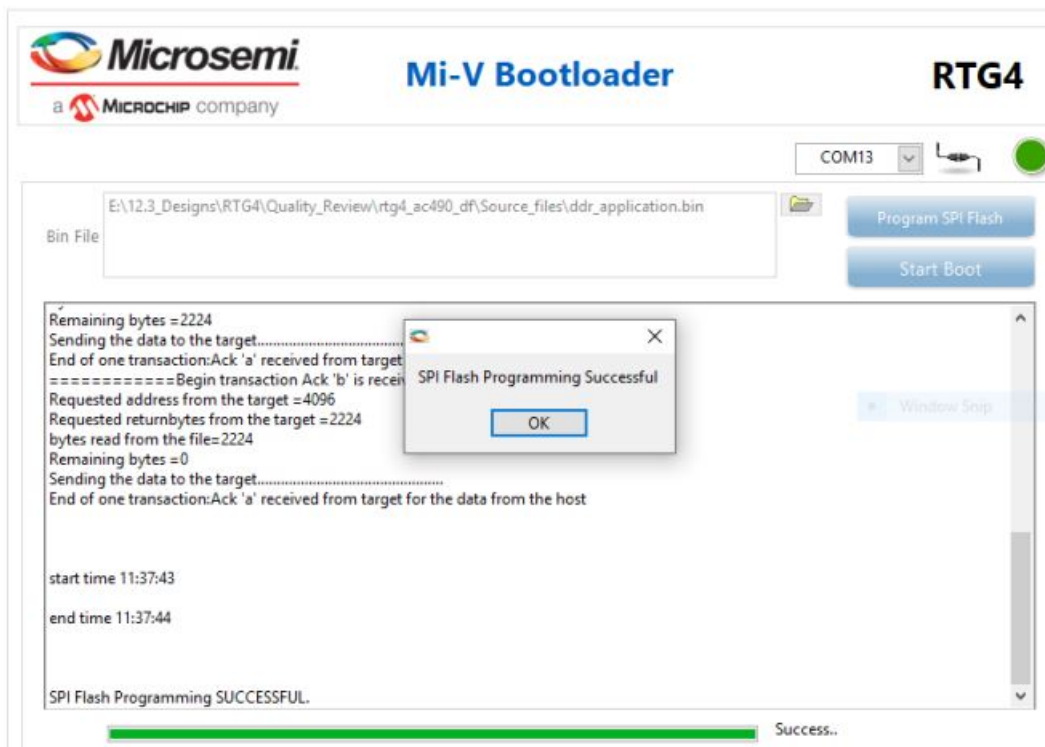
<\$Download_Directory>\rtg4_ac490_df\Source_files

Figure 11 • Import the Target Application File



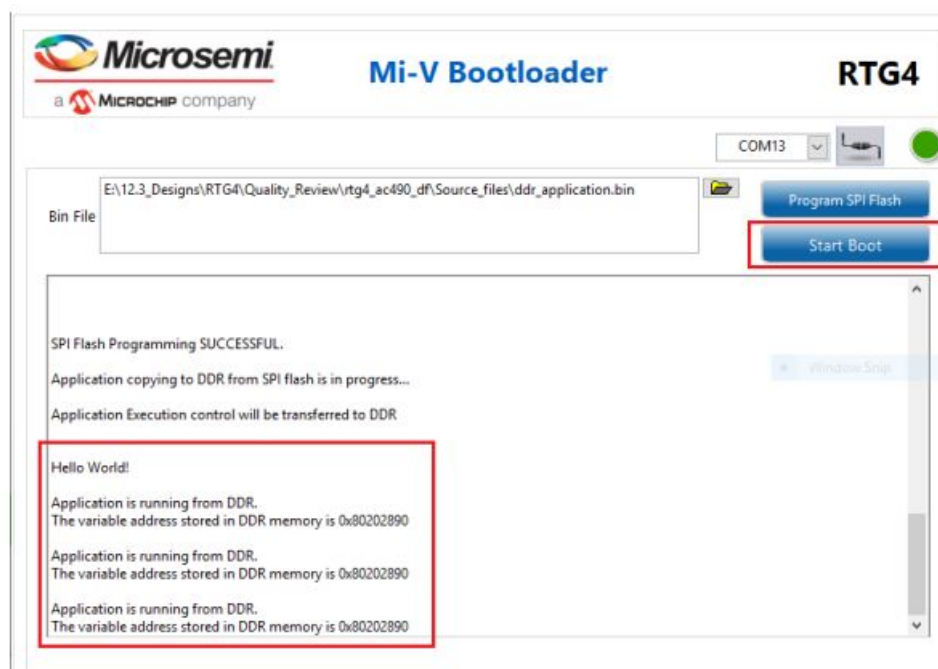
6. As shown in Figure 11, click Program SPI Flash option to program the target application on the SPI Flash. A pop-up is displayed after the SPI Flash is programmed as shown in Figure 12. Click OK.

Figure 12 • SPI Flash Programmed



7. Select the Start Boot option to copy the application from SPI Flash to DDR3 memory and start executing the application from DDR3 memory. After successful booting of the target application from DDR3 memory, the application prints UART messages and blinks on-board user LED1, 2, 3, and 4 as shown in Figure 13.

Figure 13 • Execute Application From DDR



8. The application is running from the DDR3 memory and this concludes the demo. Close the Mi-V Bootloader GUI.

Programming the Device Using FlashPro Express

This section describes how to program the RTG4 device with the programming job file using FlashPro Express.

To program the device, perform the following steps:

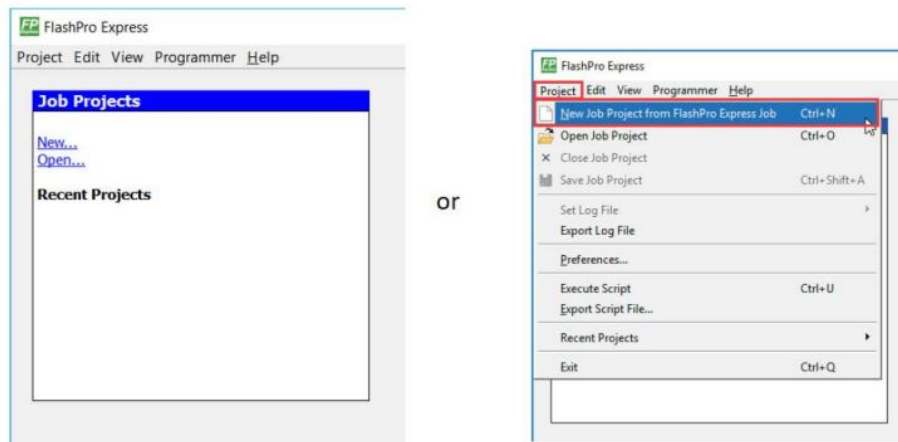
1. Ensure that the jumper settings on the board are the same as those listed in Table 3 of UG0617:

- Optionally, jumper J32 can be set to connect pins 2-3 when using an external FlashPro4, FlashPro5, or FlashPro6 programmer instead of the default jumper setting to use the embedded FlashPro5.

Note: The power supply switch, SW6 must be switched OFF while making the jumper connections.

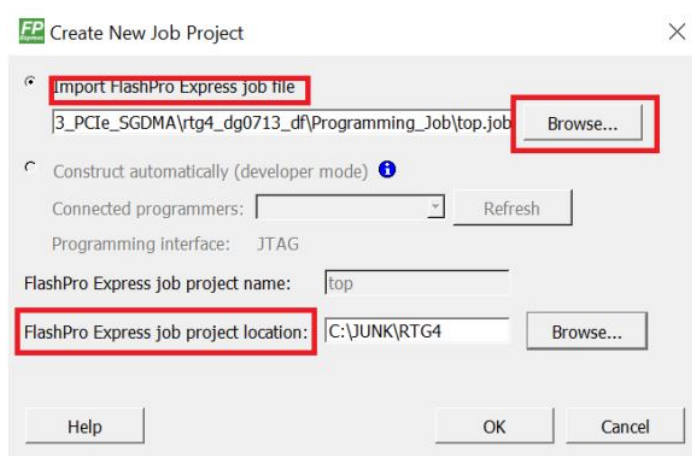
- Connect the power supply cable to the J9 connector on the board.
- Power ON the power supply switch SW6.
- If using the embedded FlashPro5, connect the USB cable to connector J47 and the host PC.
Alternatively, if using an external programmer, connect the ribbon cable to the JTAG header J22 and connect the programmer to the host PC.
- On the host PC, launch the FlashPro Express software.
- Click New or select New Job Project from FlashPro Express Job from Project menu to create a new job project, as shown in the following figure.

Figure 14 • FlashPro Express Job Project



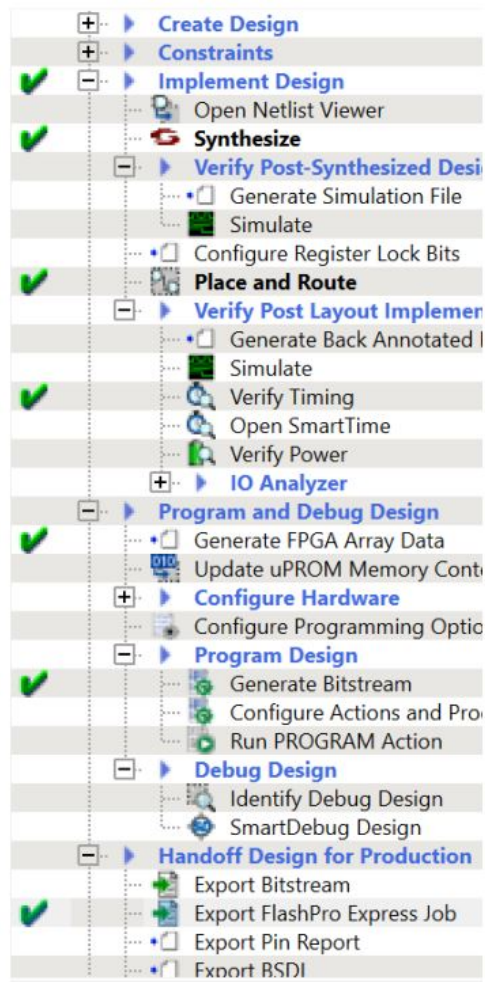
- Enter the following in the New Job Project from FlashPro Express Job dialog box:
 - Programming job file: Click Browse, and navigate to the location where the .job file is located and select the file. The default location is: <download_folder>\rtg4_ac490_df\Programming_Job
 - FlashPro Express job project location: Click Browse and navigate to the desired FlashPro Express project location.

Figure 15 • New Job Project from FlashPro Express Job



- Click OK. The required programming file is selected and ready to be programmed in the device.
- The FlashPro Express window appears as shown in the following figure. Confirm that a programmer number appears in the Programmer field. If it does not, confirm the board connections and click Refresh/Rescan Programmers.

Figure 16 • Programming the Device



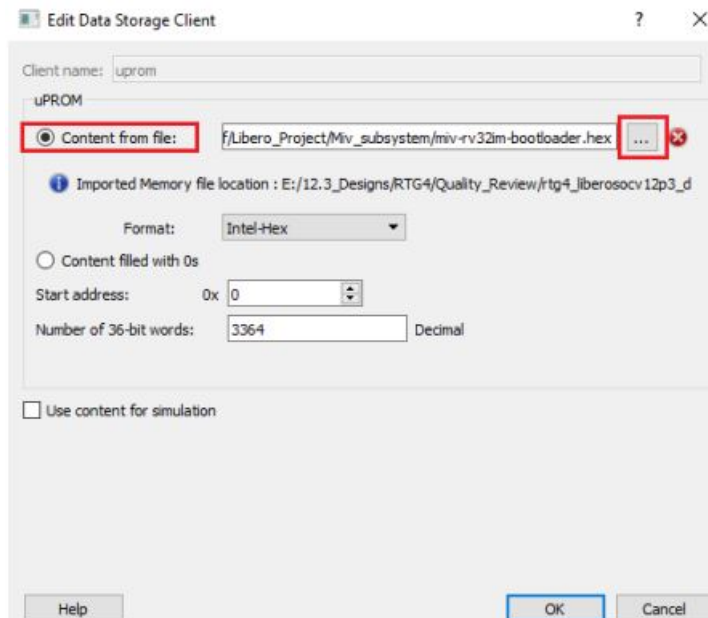
To program the RTG4 device, the Mi-V processor subsystem project must be opened in Libero SoC and the following steps must be re-run:

1. Update uPROM Memory Content: In this step, μ PROM is programmed with the bootloader application.
2. Bitstream Generation: In this step, the Job file is generated for the RTG4 device.
3. FPGA Programming: In this step, the RTG4 device is programmed using the Job file.

Follow these steps:

1. From Libero Design Flow, select Update uPROM Memory Content.
2. Create a client using the Add option.
3. Select the client and then choose the Edit option.
4. Select Content from file and then select the Browse option as shown in Figure 19.

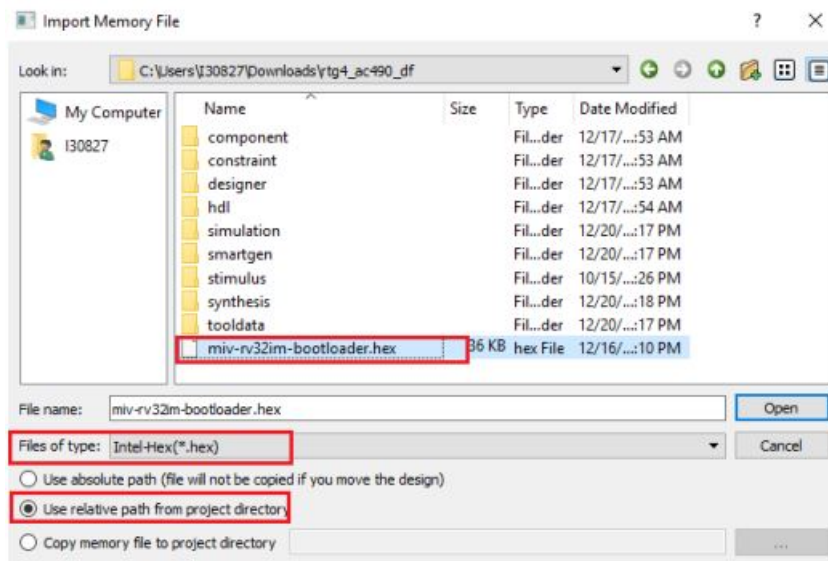
Figure 19 • Edit Data Storage Client



5. Navigate to the following design files location and select the miv-rv32im-bootloader.hex file as shown in Figure 20. <\$Download_Directory>\rtg4_ac490_df

- Set the File Type as Intel-Hex (*.hex).
- Select Use relative path from project directory.
- Click OK.

Figure 20 • Import Memory File

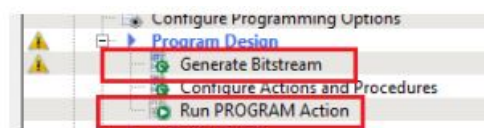


6. Click OK.

The μ PROM content is updated.

7. Double-click Generate Bitstream as shown in Figure 21.

Figure 21 • Generate Bitstream



8. Double-click Run PROGRAM Action to program the device as shown in Figure 21.

The RTG4 device is programmed. See Running the Demo, page 11.

Running the TCL Script

TCL scripts are provided in the design files folder under directory TCL_Scripts. If required, the design flow can be reproduced from Design Implementation till generation of job file.

To run the TCL, follow the steps below:

1. Launch the Libero software.
2. Select Project > Execute Script....
3. Click Browse and select script.tcl from the downloaded TCL_Scripts directory.
4. Click Run.

After successful execution of TCL script, Libero project is created within TCL_Scripts directory.
For more information about TCL scripts, refer to rtg4_ac490_df/TCL_Scripts/readme.txt.
Refer to Libero® SoC TCL Command Reference Guide for more details on TCL commands. Contact Technical Support for any queries encountered when running the TCL script.

Microsemi makes no warranty, representation, or guarantee regarding the information contained herein or the suitability of its products and services for any particular purpose, nor does Microsemi assume any liability whatsoever arising out of the application or use of any product or circuit. The products sold hereunder and any other products sold by Microsemi have been subject to limited testing and should not be used in conjunction with mission-critical equipment or applications. Any performance specifications are believed to be reliable but are not verified, and Buyer must conduct and complete all performance and other testing of the products, alone and together with, or installed in, any end-products. Buyer shall not rely on any data and performance specifications or parameters provided by Microsemi. It is the Buyer's responsibility to independently determine suitability of any products and to test and verify the same. The information provided by Microsemi hereunder is provided "as is, where is" and with all faults, and the entire risk associated with such information is entirely with the Buyer. Microsemi does not grant, explicitly or implicitly, to any party any patent rights, licenses, or any other IP rights, whether with regard to such information itself or anything described by such information. Information provided in this document is proprietary to Microsemi, and Microsemi reserves the right to make any changes to the information in this document or to any products and services at any time without notice.



About Microsemi

Microsemi, a wholly owned subsidiary of Microchip Technology Inc. (Nasdaq: MCHP), offers a comprehensive portfolio of semiconductor and system solutions for aerospace & defense, communications, data center and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; enterprise storage and communication solutions, security technologies and scalable anti-tamper products; Ethernet solutions; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Learn more at www.microsemi.com.








Microsemi Headquarters
One Enterprise, Aliso Viejo,
CA 92656 USA
Within the USA: +1 (800) 713-4113
Outside the USA: +1 (949) 380-6100
Sales: +1 (949) 380-6136
Fax: +1 (949) 215-4996
Email: sales.support@microsemi.com
www.microsemi.com

©2021 Microsemi, a wholly owned subsidiary of Microchip Technology Inc. All rights reserved. Microsemi and the Microsemi logo are registered trademarks of Microsemi Corporation. All other trademarks and service marks are the property of their respective owners

Documents / Resources

 	<p>Microsemi AC490 RTG4 FPGA: Building a Mi-V Processor Subsystem [pdf] User Guide AC490 RTG4 FPGA Building a Mi-V Processor Subsystem, AC490 RTG4, FPGA Building a Mi-V Processor Subsystem, Mi-V Processor Subsystem</p>
--	--

References

-  [FPGA Documentation | Microchip Technology](#)
-  [Microsemi | Semiconductor & System Solutions | Power Matters](#)
-  [microchip.com/en-us/application-notes/ac490](#)
-  [microsemi.com/document-portal/doc_download/1245481-libero-soc-v12-6-tcl-commands-reference-guide-for-smartfusion2-igloo2-and-rtg4](#)
-  [UG0362: Three-phase PWM v4.1 User Guide](#)
-  [microsemi.com/document-portal/doc_download/136945-tu0775-polarfire-fpga-building-a-mi-v-processor-subsystem-tutorial](#)
-  [Libero® SoC Design Suite Versions 2023.1 to 12.0 | Microchip Technology](#)