



# First Thursdays

---

May 2 - Webinar 1: Discover Renode for PolarFire® SoC Design and Debug

June 6 - Webinar 2: How to Get Started with Renode for PolarFire SoC

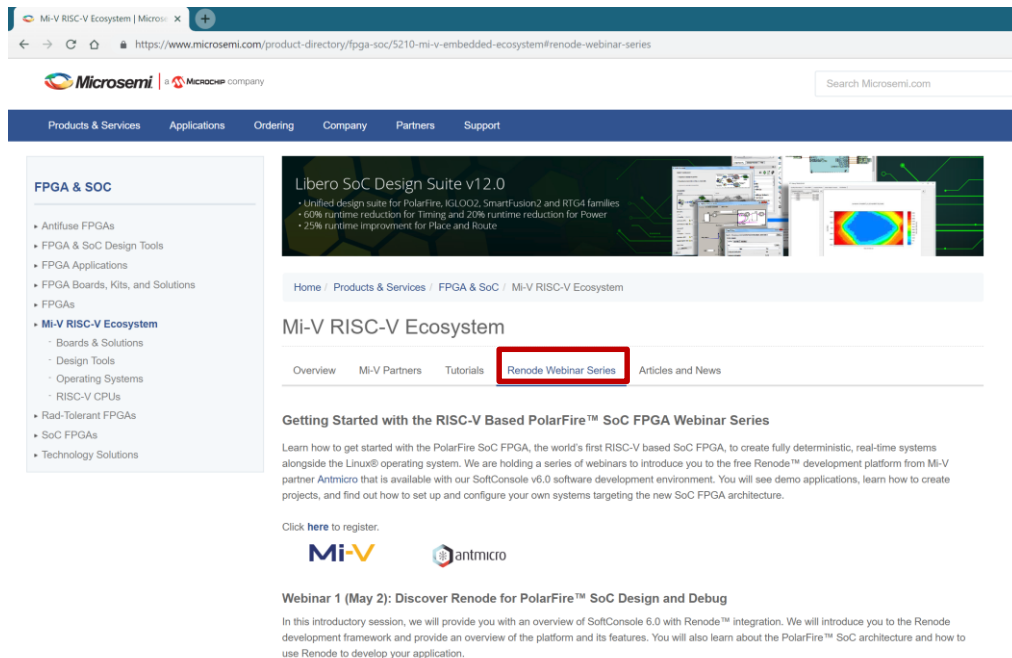
July 4 - Webinar 3: Learn to Debug a Bare-Metal PolarFire SoC Application with Renode

**Aug. 1 - Webinar 4: Tips and Tricks for Even Easier PolarFire SoC Debug with Renode**

**Sept. 5 - Webinar 5: Add and Debug PolarFire SoC Peripherals with Renode**

**Oct. 3 - Webinar 6: Add and debug a pre-existing peripheral in PolarFire SoC**

# Supporting Content



Mi-V RISC-V Ecosystem | Microsemi

Microsemi | a MICROCHIP company

Search Microsemi.com

Products & Services Applications Ordering Company Partners Support

**FPGA & SoC**

- Antifuse FPGAs
- FPGA & SoC Design Tools
- FPGA Applications
- FPGA Boards, Kits, and Solutions
- FPGAs
- **Mi-V RISC-V Ecosystem**
  - Boards & Solutions
  - Design Tools
  - Operating Systems
  - RISC-V CPUs
- Rad-Tolerant FPGAs
- SoC FPGAs
- Technology Solutions

**Libero SoC Design Suite v12.0**

- Unified design suite for PolarFire, IGLOO2, SmartFusion2 and RTG4 families
- 60% runtime reduction for Timing and 20% runtime reduction for Power
- 25% runtime improvement for Place and Route

Home / Products & Services / FPGA & SoC / Mi-V RISC-V Ecosystem


## Mi-V RISC-V Ecosystem

Overview Mi-V Partners Tutorials **Renode Webinar Series** Articles and News

### Getting Started with the RISC-V Based PolarFire™ SoC FPGA Webinar Series

Learn how to get started with the PolarFire SoC FPGA, the world's first RISC-V based SoC FPGA, to create fully deterministic, real-time systems alongside the Linux® operating system. We are holding a series of webinars to introduce you to the free Renode™ development platform from Mi-V partner Antmicro that is available with our SoftConsole v6.0 software development environment. You will see demo applications, learn how to create projects, and find out how to set up and configure your own systems targeting the new SoC FPGA architecture.

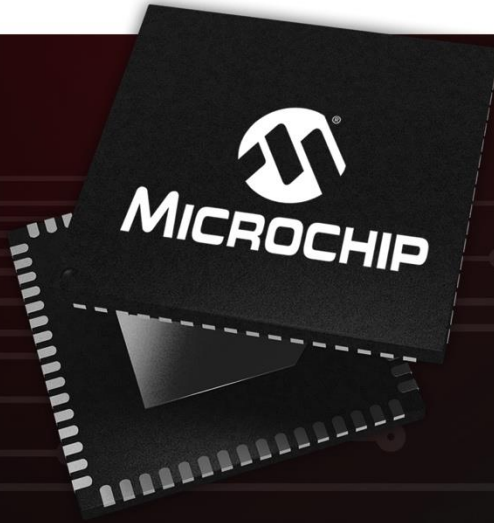
Click [here](#) to register.

**Mi-V** 

### Webinar 1 (May 2): Discover Renode for PolarFire™ SoC Design and Debug

In this introductory session, we will provide you with an overview of SoftConsole 6.0 with Renode™ integration. We will introduce you to the Renode development framework and provide an overview of the platform and its features. You will also learn about the PolarFire™ SoC architecture and how to use Renode to develop your application.

[www.microsemi.com/Mi-V](https://www.microsemi.com/Mi-V) “Renode Webinar Series”



A Leading Provider of Microcontroller, Security, Mixed-Signal, Analog & Flash-IP Solutions



**Getting Started with the RISC-V Based PolarFire® SoC FPGA Webinar Series**  
***Session 4: Tips and Tricks for Even Easier PolarFire SoC Debug with Renode***

*Hugh Breslin, Embedded Linux Engineer*

*Thursday Aug. 1, 2019*



# Tips and Tricks for Even Easier PolarFire® SoC Debug with Renode

---

- **Recap**
- **Using the Renode console**
- **Using Renode logs and their output**

# Recap

---

- **Webinar 1: Discover Renode for PolarFire® SoC Design and Debug**
  - Introduction to the Renode platform and PolarFire SoC
- **Webinar 2: How to Get Started with Renode for PolarFire SoC**
  - Installing SoftConsole with Renode on Windows® and Linux®
  - Demo of the Mi-V example projects
- **Webinar 3: Learn to Debug a Bare-Metal PolarFire SoC Application with Renode**
  - How to configure debug sessions for projects
  - How to run and debug the pse-blinky project

# Recap

---

- **Install SoftConsole v6.0 with Renode**
- **Able to launch the included sample projects and debug in Renode**
- **Set up debug configurations and configure launch groups for use with Renode**
- **Basic use of the Renode console**
- **Run PolarFire SoC applications on multiple harts**



# Learn to Debug a Bare-Metal PolarFire SoC Application with Renode

---

## Using The Renode Console

# Using the Renode Console

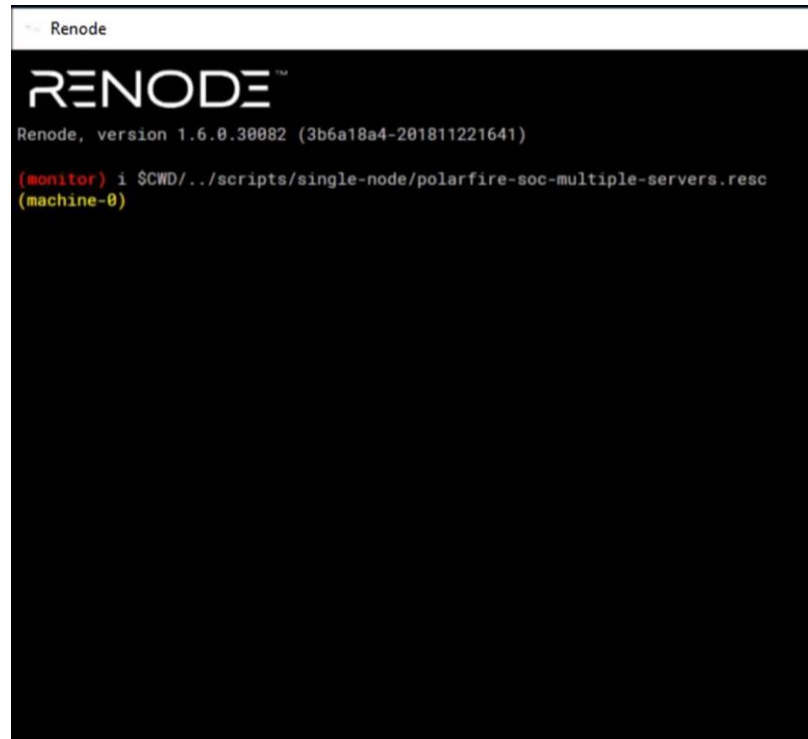
- Renode commands and peripheral names are case sensitive
- Incorrectly capitalizing a command or peripheral name will cause an error even if spelling is correct!

e.g.:

sysbus.e51 – ok!

sysbus.**E**51 – not ok

**S**ysbus.e51 – not ok



```
Renode
RENODE™
Renode, version 1.6.0.30082 (3b6a18a4-201811221641)
(monitor) i $CWD/../scripts/single-node/polarfire-soc-multiple-servers.resc
(machine-0)
```



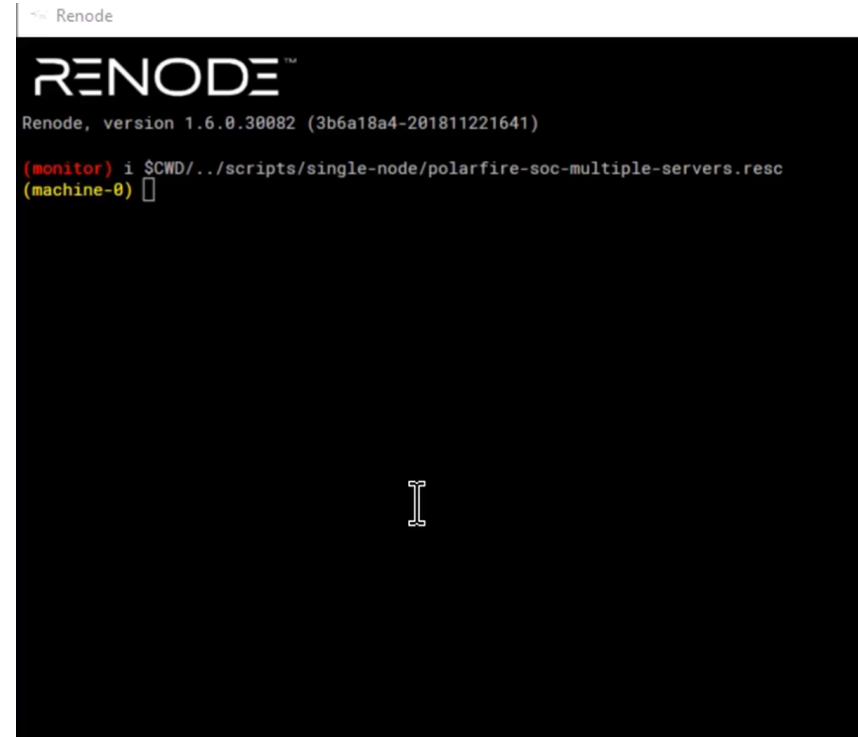
# Using the Renode Console

## Command: help

**Use / Example:** “help” “help [command]” “help logLevel”

**Output:** Prints available commands if no arguments and prints information about the command and usage examples if a command is given as an argument

**Notes:** Cannot be used for peripherals i.e. “help e51” will produce an error

A screenshot of the Renode console interface. The title bar at the top says "Renode". The main area has a black background with white text. It displays the "RENODE™" logo, followed by "Renode, version 1.6.0.30082 (3b6a18a4-201811221641)". Below this, there are two lines of text: "(monitor) i \$CWD/../../scripts/single-node/polarfire-soc-multiple-servers.resc" and "(machine-0) |". A white cursor is visible at the end of the second line.

```
Renode  
RENODE™  
Renode, version 1.6.0.30082 (3b6a18a4-201811221641)  
(monitor) i $CWD/../../scripts/single-node/polarfire-soc-multiple-servers.resc  
(machine-0) |
```

# Using the Renode Console

---

**Command:** peripherals

**Use/Example:** “peripherals”

**Output:** Prints peripherals connected to sysbus

**Notes:** Prints sysbus for the current machine



```
Renode
RENODE™
Renode, version 1.6.0.30082 (3b6a18a4-201811221641)
(monitor) i $CWD/../scripts/single-node/polarfire-soc-multiple-servers.resc
(machine-0) |
```

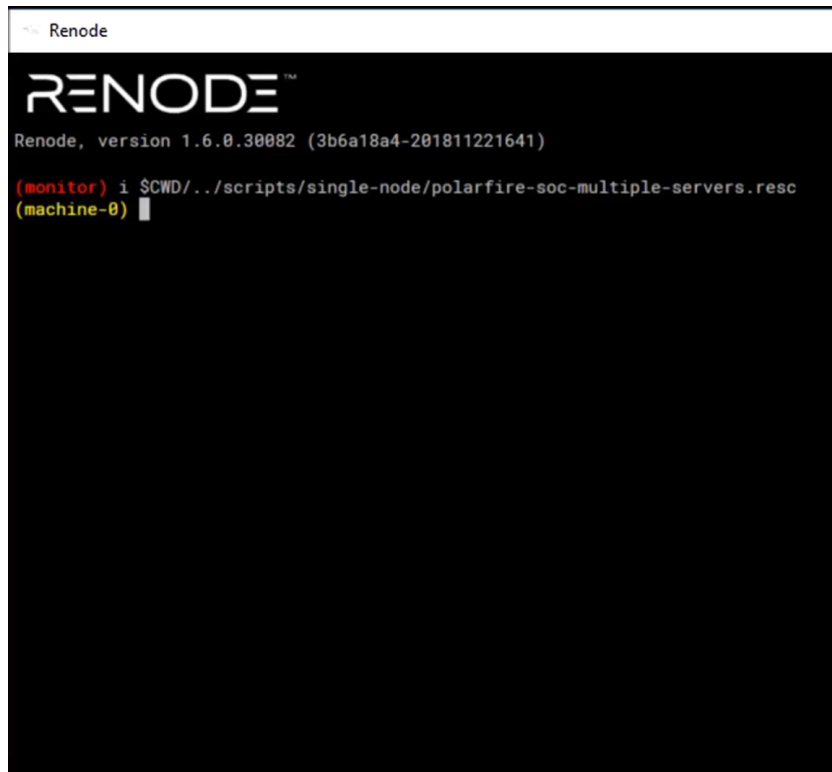
# Using the Renode Console

**Command:** using sysbus

**Use/Example:** “using sysbus”

**Output:** N/A

**Notes:** Tells Renode to assume you're using sysbus



```
Renode  
  
RENODE™  
Renode, version 1.6.0.30082 (3b6a18a4-201811221641)  
  
(monitor) i $CWD/./scripts/single-node/polarfire-soc-multiple-servers.resc  
(machine-0) |
```

# Using the Renode Console

**Command:** [peripheral name]

**Use/Example:** “e51”

**Output:** Prints methods and properties available to be used with the peripheral

**Notes:** Methods can be called to set or change functionality of a peripheral (e.g. reset) and properties are values describing the peripheral or its state (e.g. Architecture or IsHalted)



```
Renode  
  
RENODE™  
Renode, version 1.6.0.30082 (3b6a18a4-201811221641)  
(monitor) i $CWD/./scripts/single-node/polarfire-soc-multiple-servers.resc  
(machine-0) █
```

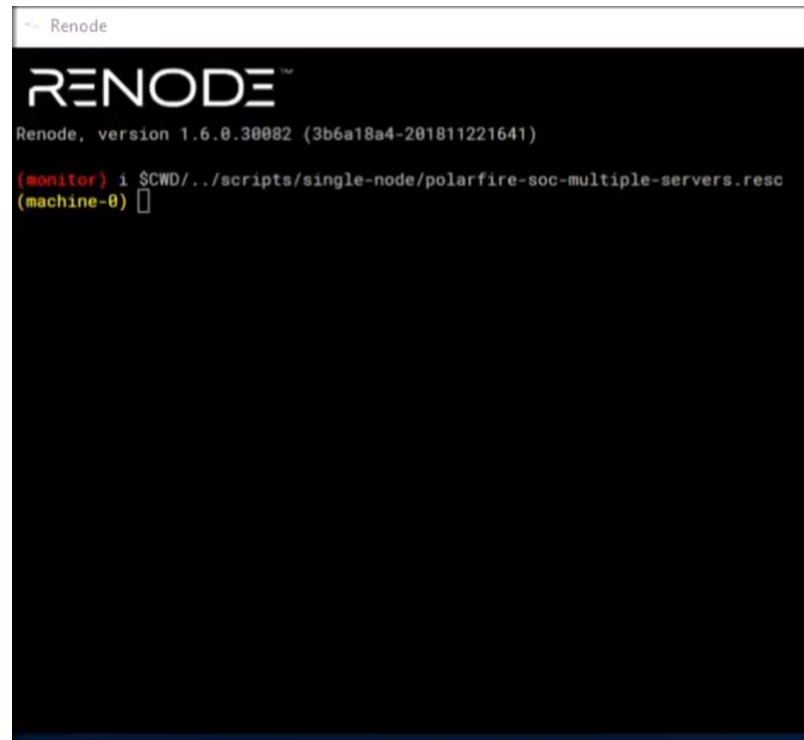
# Using the Renode Console

**Command:** [peripheral name] [method]  
[args]

**Use/Example:** “e51 AddHook 0x60000000 @main.py”  
“e51 GetRegistersValues”

**Output:** Depends on the return type of the method (if any)

**Notes:**



The screenshot shows a terminal window titled "Renode". Inside, the "RENODE" logo is displayed in large white letters. Below the logo, the text "Renode, version 1.6.0.30082 (3b6a18a4-201811221641)" is shown. The prompt "(monitor) " is followed by a command: "i \$CWD/../../scripts/single-node/polarfire-soc-multiple-servers.resc". Below this, the prompt "(machine-0) " is shown with a cursor.

# Using the Renode Console

Command: [peripheral name] [method] [args]

e.g.: - **String** GetCPUThreadName (**Machine** machine)

- **[return value]** {command} (**[arg type]** arg, **[arg type]** arg)
- **Void** AddHook (**UInt64** addr, **String** pythonScript)
- **String** DisassembleBlock (**UInt32** addr, **UInt32** flags = 0)
- **Boolean** HasGPIO()
- **Boolean** IsSetEvent(**Int32** number)

**Void:** No return value, **String:** Returns a string, **Boolean:** Returns true / false

Renode

**RENODE™**

Renode, version 1.6.0.30082 (3b6a18a4-201811221641)

```
(monitor) i $CWD/../scripts/single-node/polarfire-soc-multiple-servers.resc  
(machine-0) sysbus.e51
```

The following methods are available:

```
- Void AddHook (UInt64 addr, String pythonScript)  
- Void ClearHookAtBlockBegin ()  
- Void ClearPageAccessViaIo (UInt64 address)  
- Void ClearTranslationCache ()  
- Void DebugLog (String message)  
- String DisassembleBlock (UInt32 addr, UInt32 flags = 0)  
- Void Dispose ()  
- String GetCPUThreadName (Machine machine)  
- IEnumerable<Tuple<String, IGPIO>> GetGPIOs ()  
- Machine GetMachine ()  
- IEnumerable<CPURegister> GetRegisters ()  
- String[,] GetRegistersValues ()  
- RegisterValue GetRegisterUnsafe (Int32 register)  
- Boolean HasGPIO ()  
- Boolean IsInstructionSetEnabled (InstructionSet set)  
- Boolean IsSetEvent (Int32 number)  
- Void Log (LogLevel type, String message)  
- Void LogFunctionNames (Boolean value, String spaceSeparatedPrefixes = "")  
- Void LogUnhandledRead (Int64 offset)  
- Void LogUnhandledWrite (Int64 offset, Int64 value)  
- Void NoisyLog (String message)  
- Void OnGPIO (Int32 number, Boolean value)  
- Void Pause ()  
- Void RemoveAllHooks ()  
- Void RemoveHooksAt (UInt64 addr)  
- Void Reset ()  
- Void Resume ()  
- Void SetHookAtBlockBegin (String pythonScript)
```

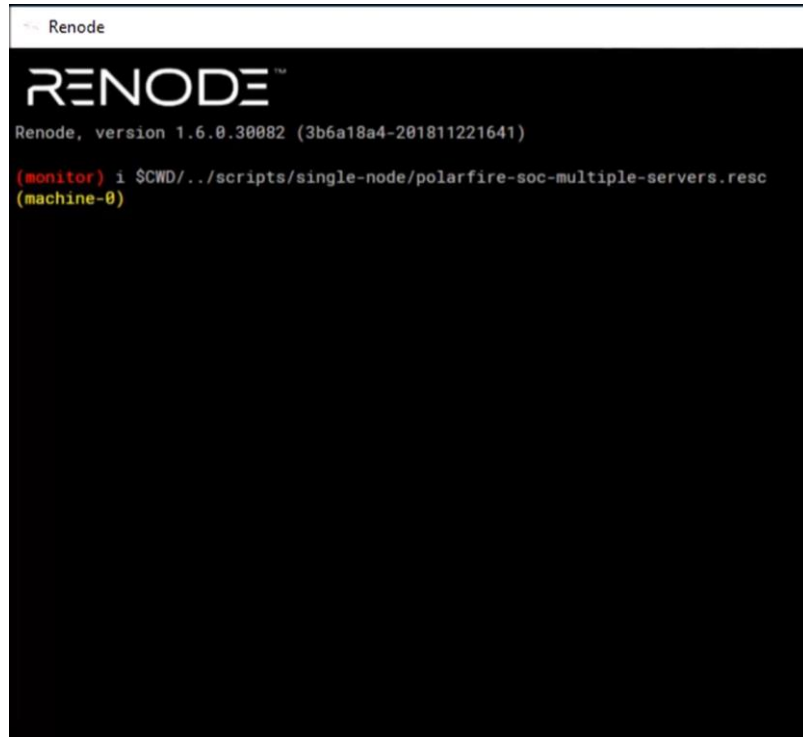
# Using the Renode Console

**Command:** [peripheral name] [property]  
[args]

**Use / Example:** “e51 Architecture”, “e51 IsHalted”,  
“e51 IsHalted True”

**Output:** None if setting, will print value if using get

**Notes:** Some properties are read-only and some can be set during execution, you can tell as read-only will only have “get” available and read and write properties will have “get and set” available

A screenshot of the Renode console interface. The window title is "Renode". The main area has a black background with white text. At the top, it says "RENODE™". Below that, it shows the version: "Renode, version 1.6.0.30082 (3b6a18a4-201811221641)". There are two lines of text in a monospaced font: "(monitor) i SCWD/..scripts/single-node/polarfire-soc-multiple-servers.resc" and "(machine-0)".

```
Renode
RENODE™
Renode, version 1.6.0.30082 (3b6a18a4-201811221641)
(monitor) i SCWD/..scripts/single-node/polarfire-soc-multiple-servers.resc
(machine-0)
```

# Using the Renode Console

## Command: [peripheral name] [property] [args]

- **String** Architecture

Available for **get**

- **String[]** AvailableDisassemblers

Available for **get**

- **Boolean** ChainingEnabled

Available for **get** and **set**

**get**: Returns the current value of the property

**set**: Sets the value of the property

“available for ‘get’”: read only

“available for ‘get’ and ‘set’”: read and write

The following properties are available:

- **String** Architecture  
available for **get**
- **String[]** AvailableDisassemblers  
available for **get**
- **SystemBus** Bus  
available for **get**
- **Boolean** ChainingEnabled  
available for **get** and **set**
- **Boolean** DisableInterruptsWhileStepping  
available for **get** and **set**
- **String** Disassembler  
available for **get** and **set**
- **Endianess** Endianness  
available for **get**
- **UInt64** ExecutedInstructions  
available for **get**
- **ExecutionMode** ExecutionMode  
available for **get** and **set**
- **String** GDBArchitecture  
available for **get**
- **UInt32** HartId  
available for **get** and **set**
- **UInt32** Id  
available for **get**



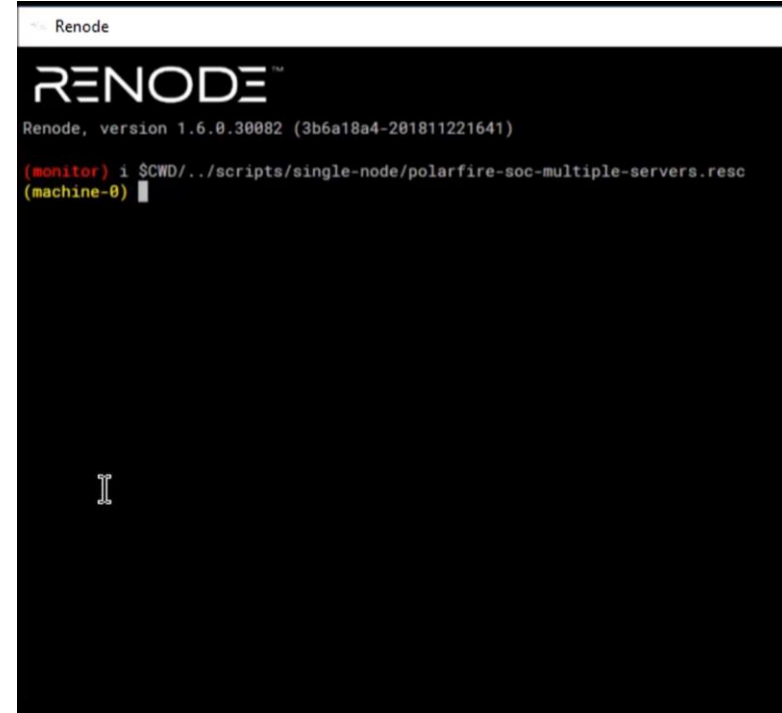
# Using the Renode Console

**Command:** [peripheral name].[sub-peripheral]

**Use/Example:** “gpio0.button0”

**Output:** Prints methods and properties available to be used with the peripheral

**Notes:** Methods can be called to set or change functionality of a peripheral (e.g reset) and properties are values describing the peripheral or its state (e.g Architecture or IsHalted)



The screenshot shows the Renode console window. At the top, it says "Renode". Below that is the "RENODE" logo. Then, it displays the version information: "Renode, version 1.6.0.30082 (3b6a18a4-201811221641)". The console is in a shell-like state with a prompt "(monitor) i \$CWD/./scripts/single-node/polarfire-soc-multiple-servers.resc". Below that, it shows "(machine-0)" with a cursor. A vertical cursor is visible in the lower part of the console area.

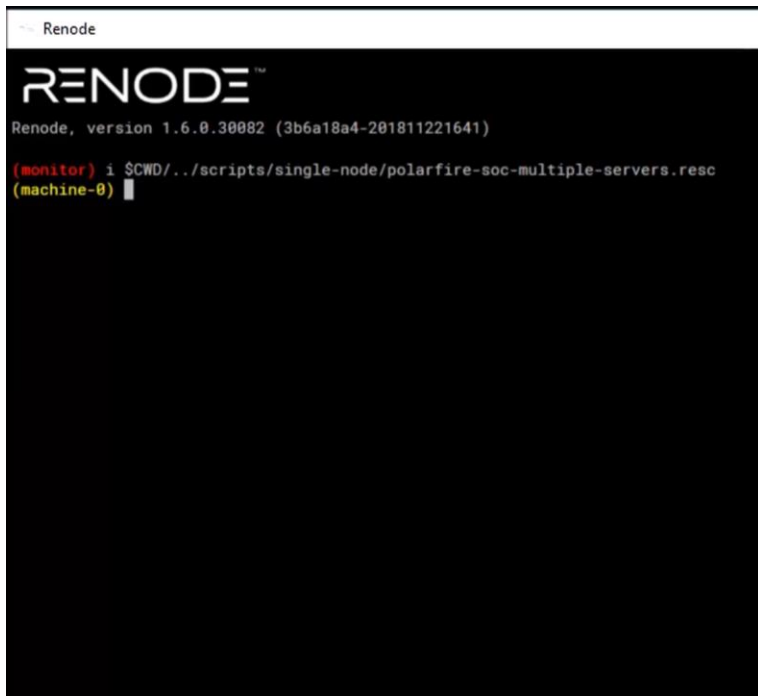
# Using the Renode Console

**Command:** [peripheral name].[sub-peripheral] [method]  
[args]

**Use / Example:** “gpio0.button0 GetGPIOs”,  
“gpio0.button0 PressAndRelease”

**Output:** None if setting, will print value if using get

**Notes:** Some properties are read-only and some can be set during execution, you can tell as read-only will only have “get” available and read and write properties will have “get and set” available



```
Renode
RENODE™
Renode, version 1.6.0.30082 (3b6a18a4-201811221641)
(monitor) i $CWD/../../scripts/single-node/polarfire-soc-multiple-servers.resc
(machine-0) █
```

# Using the Renode Console

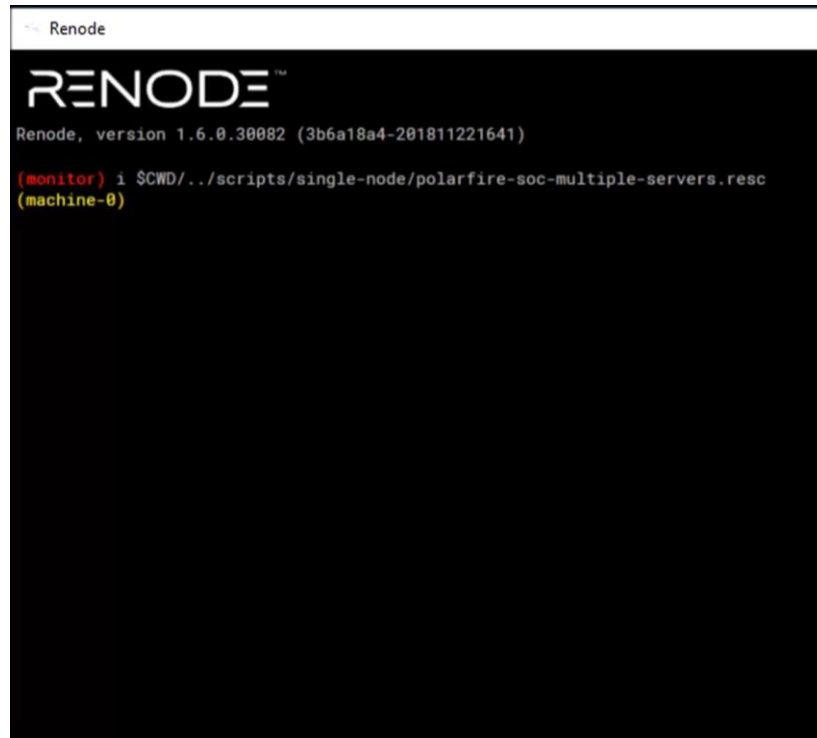
**Command:** `logLevel [level]`  
`[peripheral]`

**Use / Example:** “logLevel”, “logLevel -1”,  
“logLevel -1 sysbus.e51”

**Output:** No args: Peripherals currently being  
logged and their log levels

With args: No output

**Notes:**



The screenshot shows a terminal window titled "Renode". Inside, the "RENODE" logo is displayed in large white letters. Below the logo, the text "Renode, version 1.6.0.30082 (3b6a18a4-201811221641)" is shown. The prompt "(monitor) i" is followed by the command "\$CWD/../scripts/single-node/polarfire-soc-multiple-servers.resc". Below this, the prompt "(machine-0)" is visible, indicating the console is ready for interaction with the machine.

# **Learn to Debug a Bare-Metal PolarFire SoC Application with Renode**

---

## **Using Renode Logs and Their Output**

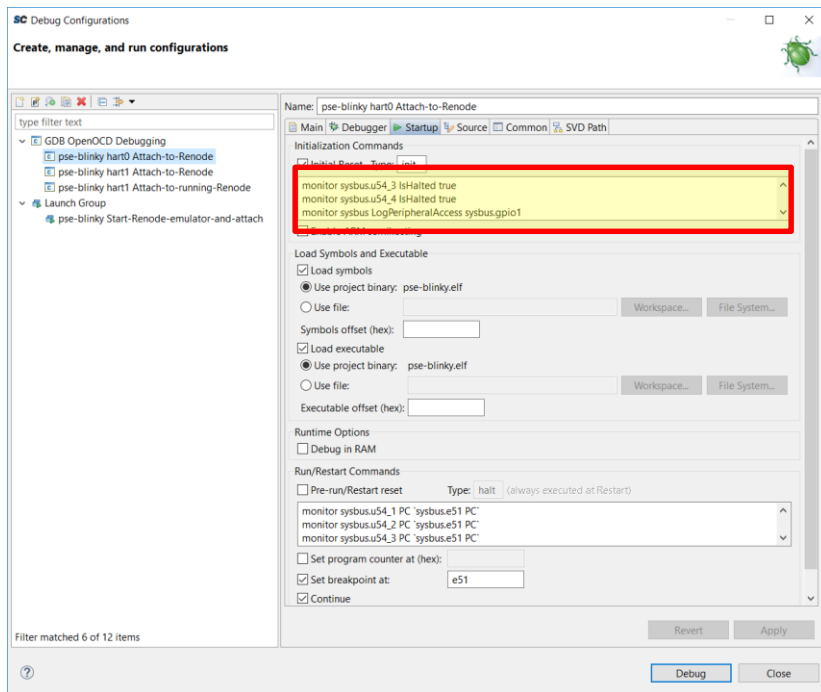
# Using Renode Logs and Their Output

---

- **There are several logging levels available:**
  - -1 Noisy
  - 0 Debug
  - 1 Info
  - 2 Warning
  - 3 Error
- **Each can capture different levels of data**
- **Peripherals will produce a different level of log data depending on what has caused the log (e.g. uart print vs fatal error)**

# Using Renode Logs and Their Output

- Logging (and other commands) can be set up automatically by passing the commands during start up
- E.G (pse-blinky debug configuration):
  - monitor sysbus.u54\_3 IsHalted true
  - monitor sysbus.u54\_4 IsHalted true
  - monitor sysbus LogPeripheralAccess sysbus.gpio1
  - monitor logLevel -1 sysbus.gpio1



# Using Renode Logs and Their Output

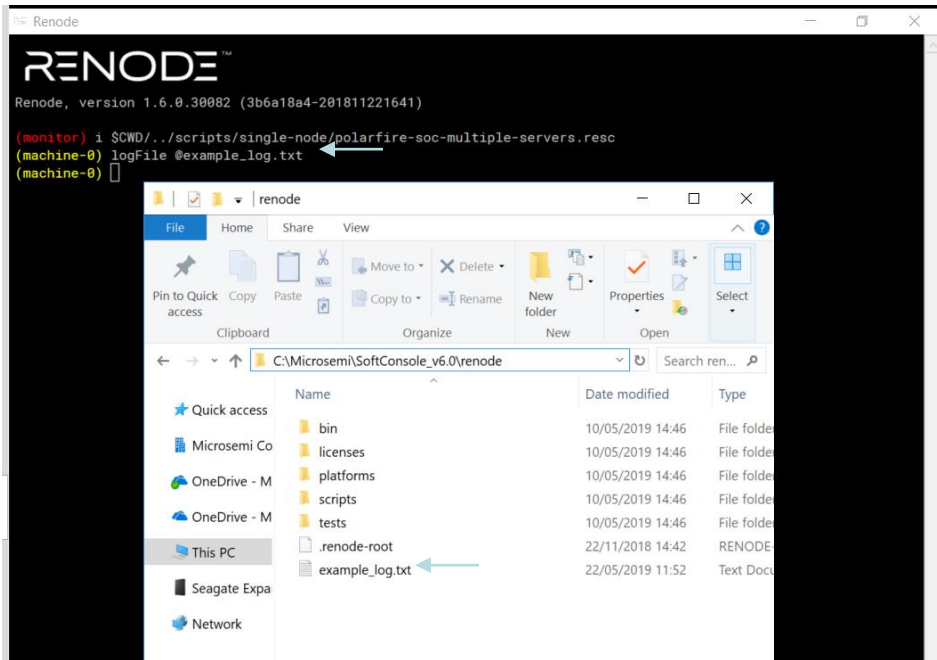
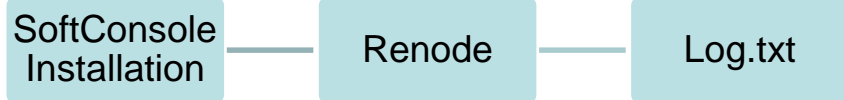
Logging can be set globally or set for individual peripherals

```
Renode
RENODE™
Renode, version 1.6.0.30082 (3b6a18a4-201811221641)
(monitor) i $CWD/../scripts/single-node/polarfire-soc-multiple-servers.resc
(machine-0) █
```

```
Renode
RENODE™
Renode, version 1.6.0.30082 (3b6a18a4-201811221641)
(monitor) i $CWD/../scripts/single-node/polarfire-soc-multiple-servers.resc
(machine-0)
```

# Using Renode Logs and Their Output

- Logs can be directed to a file
- The log file is created if it doesn't exist and existing logs are overwritten





# Using Renode Logs and Their Output

**Example:**

**logFile @uart.txt**

**logLevel 0 sysbus.mmuart0**

**Sysbus LogPeripheralAccess sysbus.mmuart0**

Microsemi > SoftConsole\_v6.0 > renode

Name	Date modified	Type	Size
bin	04/07/2019 08:21	File folder	
licenses	04/07/2019 08:21	File folder	
platforms	04/07/2019 08:21	File folder	
scripts	04/07/2019 08:21	File folder	
system_builder	08/07/2019 10:59	File folder	
tests	04/07/2019 08:21	File folder	
.renode-root	22/11/2018 14:42	RENODE-ROOT File	
uart.txt ←	11/07/2019 09:16	Text Document	

```
Renode
RENODE™
Renode, version 1.6.0.30082 (3b6a18a4-201811221641)

(monitor) i $CWD/../../scripts/single-node/polarfire-soc-multiple-servers.resc
(machine-0) █
```

# Using Renode Logs and Their Output

## logLevel 3 file



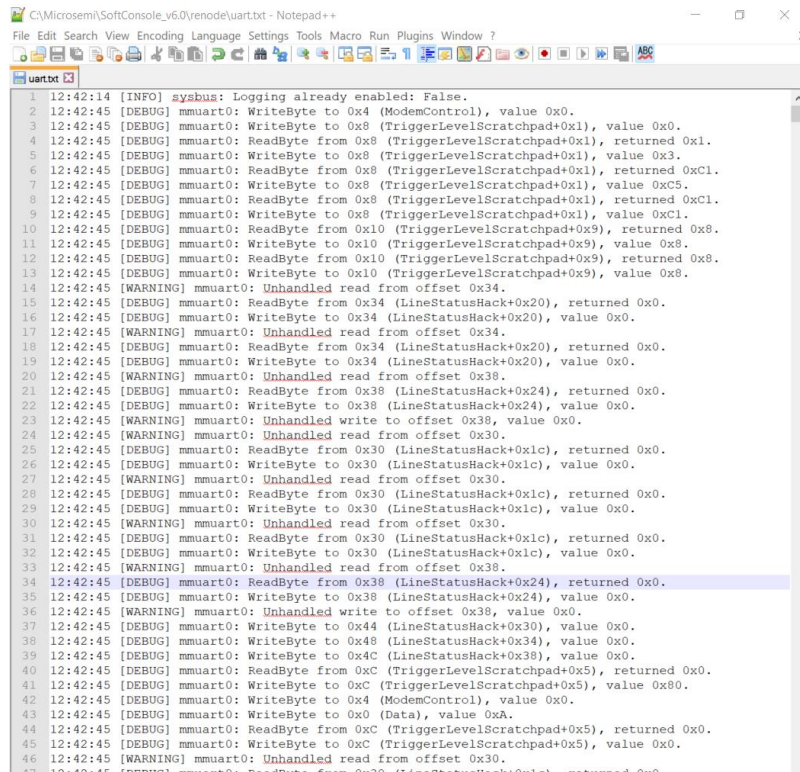
```
C:\Microsemi\SoftConsole_v6.0\renode\uart.txt - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window
uart.txt
1 12:33:35 [INFO] sysbus: Logging already enabled: False.
2 12:34:40 [DEBUG] e51: GDB packet received: Z0,800375c,4
3 12:34:40 [DEBUG] e51: Added hook @ 0x800375c
4 12:34:40 [DEBUG] e51: Sending response to GDB: OK
5 12:34:40 [DEBUG] e51: GDB packet received: Z0,8003b88,4
6 12:34:40 [DEBUG] e51: Added hook @ 0x8003b88
7 12:34:40 [DEBUG] e51: Sending response to GDB: OK
8 12:34:40 [DEBUG] e51: GDB packet received: Z0,8003e78,4
9 12:34:40 [DEBUG] e51: Added hook @ 0x8003e78
10 12:34:40 [DEBUG] e51: Sending response to GDB: OK
11 12:34:40 [DEBUG] e51: GDB packet received: c
12 12:34:40 [WARNING] sysbus: [CPU0: 0x8003CF0] WriteDouble
13 12:34:40 [WARNING] sysbus: [CPU0: 0x8003CFC] WriteDouble
14 12:34:40 [WARNING] sysbus: [CPU0: 0x8003D04] ReadDouble
15 12:34:40 [WARNING] sysbus: [CPU0: 0x8003D24] WriteDouble
16 12:34:40 [WARNING] sysbus: [CPU0: 0x8003D30] WriteDouble
17 12:34:40 [WARNING] sysbus: [CPU0: 0x8003D3C] WriteDouble
18 12:34:40 [WARNING] sysbus: [CPU0: 0x8003D44] WriteDouble
19 12:34:40 [WARNING] sysbus: [CPU0: 0x8003D4C] WriteDouble
20 12:34:40 [WARNING] sysbus: [CPU0: 0x8003D54] WriteDouble
21 12:34:40 [WARNING] sysbus: [CPU0: 0x8003D5C] WriteDouble
22 12:34:40 [WARNING] plic: Unhandled write to offset 0x2EC
23 12:34:40 [WARNING] plic: Unhandled write to offset 0x20C
24 12:34:40 [WARNING] plic: Unhandled write to offset 0x20C
25 12:34:40 [WARNING] sysbus: [CPU0: 0x8004EE4] ReadDouble
26 12:34:40 [WARNING] sysbus: [CPU0: 0x8004F00] WriteDouble
27 12:34:40 [WARNING] sysbus: [CPU0: 0x8004FA0] ReadDouble
28 12:34:40 [WARNING] sysbus: [CPU0: 0x8004FC0] WriteDouble
29 12:34:40 [WARNING] sysbus: [CPU0: 0x80051F8] ReadDouble
30 12:34:40 [WARNING] sysbus: [CPU0: 0x80051F8] ReadDouble
31 12:34:40 [WARNING] sysbus: [CPU0: 0x80051F8] ReadDouble
32 12:34:40 [WARNING] sysbus: [CPU0: 0x8004F24] ReadDouble
33 12:34:40 [WARNING] sysbus: [CPU0: 0x8004F40] WriteDouble
34 12:34:40 [WARNING] sysbus: [CPU0: 0x8004FE4] ReadDouble
35 12:34:40 [WARNING] sysbus: [CPU0: 0x8005004] WriteDouble
36 12:34:40 [DEBUG] gpiol: WriteUInt32 to 0x0 (unknown), va
37 12:34:40 [WARNING] gpiol: Unhandled write to offset 0x0.
38 12:34:40 [DEBUG] gpiol: WriteUInt32 to 0x4 (unknown), va
39 12:34:40 [WARNING] gpiol: Unhandled write to offset 0x4.
40 12:34:40 [DEBUG] gpiol: WriteUInt32 to 0x8 (unknown), va
41 12:34:40 [WARNING] gpiol: Unhandled write to offset 0x8.
42 12:34:40 [DEBUG] gpiol: WriteUInt32 to 0x88 (OutputRegis
43 12:34:40 [DEBUG] mmuart0: WriteByte to 0x4 (ModemControl
44 12:34:40 [DEBUG] mmuart0: WriteByte to 0x8 (TriggerLevel
45 12:34:40 [DEBUG] mmuart0: ReadByte from 0x8 (TriggerLeve
46 12:34:40 [DEBUG] mmuart0: WriteByte to 0x8 (TriggerLevel
47 12:34:40 [DEBUG] mmuart0: ReadByte from 0x8 (TriggerLevel
```



```
C:\Microsemi\SoftConsole_v6.0\renode\uart.txt - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window
uart.txt
1 12:42:14 [INFO] sysbus: Logging already enabled: False.
2 12:42:45 [DEBUG] mmuart0: WriteByte to 0x4 (ModemControl),
3 12:42:45 [DEBUG] mmuart0: WriteByte to 0x8 (TriggerLevelSc
4 12:42:45 [DEBUG] mmuart0: ReadByte from 0x8 (TriggerLevelSc
5 12:42:45 [DEBUG] mmuart0: WriteByte to 0x8 (TriggerLevelSc
6 12:42:45 [DEBUG] mmuart0: ReadByte from 0x8 (TriggerLevelSc
7 12:42:45 [DEBUG] mmuart0: WriteByte to 0x8 (TriggerLevelSc
8 12:42:45 [DEBUG] mmuart0: ReadByte from 0x8 (TriggerLevelSc
9 12:42:45 [DEBUG] mmuart0: WriteByte to 0x8 (TriggerLevelSc
10 12:42:45 [DEBUG] mmuart0: ReadByte from 0x10 (TriggerLevel
11 12:42:45 [DEBUG] mmuart0: WriteByte to 0x10 (TriggerLevel
12 12:42:45 [DEBUG] mmuart0: ReadByte from 0x10 (TriggerLevel
13 12:42:45 [DEBUG] mmuart0: WriteByte to 0x10 (TriggerLevel
14 12:42:45 [WARNING] mmuart0: Unhandled read from offset 0x:
15 12:42:45 [DEBUG] mmuart0: ReadByte from 0x34 (LineStatusH
16 12:42:45 [DEBUG] mmuart0: WriteByte to 0x34 (LineStatusH
17 12:42:45 [WARNING] mmuart0: Unhandled read from offset 0x:
18 12:42:45 [DEBUG] mmuart0: ReadByte from 0x34 (LineStatusH
19 12:42:45 [DEBUG] mmuart0: WriteByte to 0x34 (LineStatusH
20 12:42:45 [WARNING] mmuart0: Unhandled read from offset 0x:
21 12:42:45 [DEBUG] mmuart0: ReadByte from 0x38 (LineStatusH
22 12:42:45 [WARNING] mmuart0: WriteByte to 0x38 (LineStatusH
23 12:42:45 [WARNING] mmuart0: Unhandled write to offset 0x3:
24 12:42:45 [WARNING] mmuart0: Unhandled read from offset 0x:
25 12:42:45 [DEBUG] mmuart0: ReadByte from 0x30 (LineStatusH
26 12:42:45 [DEBUG] mmuart0: WriteByte to 0x30 (LineStatusH
27 12:42:45 [WARNING] mmuart0: Unhandled read from offset 0x:
28 12:42:45 [DEBUG] mmuart0: ReadByte from 0x30 (LineStatusH
29 12:42:45 [DEBUG] mmuart0: WriteByte to 0x30 (LineStatusH
30 12:42:45 [WARNING] mmuart0: Unhandled read from offset 0x:
31 12:42:45 [DEBUG] mmuart0: ReadByte from 0x30 (LineStatusH
32 12:42:45 [DEBUG] mmuart0: WriteByte to 0x30 (LineStatusH
33 12:42:45 [WARNING] mmuart0: Unhandled read from offset 0x:
34 12:42:45 [DEBUG] mmuart0: ReadByte from 0x38 (LineStatusH
35 12:42:45 [WARNING] mmuart0: WriteByte to 0x38 (LineStatusH
36 12:42:45 [WARNING] mmuart0: Unhandled write to offset 0x3:
37 12:42:45 [DEBUG] mmuart0: WriteByte to 0x44 (LineStatusH
38 12:42:45 [DEBUG] mmuart0: WriteByte to 0x48 (LineStatusH
39 12:42:45 [DEBUG] mmuart0: WriteByte to 0x4C (LineStatusH
40 12:42:45 [DEBUG] mmuart0: ReadByte from 0xC (TriggerLevelSc
41 12:42:45 [DEBUG] mmuart0: WriteByte to 0xC (TriggerLevelSc
42 12:42:45 [DEBUG] mmuart0: WriteByte to 0x4 (ModemControl),
43 12:42:45 [DEBUG] mmuart0: WriteByte to 0x0 (Data), value (
44 12:42:45 [DEBUG] mmuart0: ReadByte from 0xC (TriggerLevelSc
45 12:42:45 [DEBUG] mmuart0: WriteByte to 0xC (TriggerLevelSc
46 12:42:45 [WARNING] mmuart0: Unhandled read from offset 0x:
47 12:42:45 [WARNING] mmuart0: ReadByte from 0x34 (LineStatusH
```

# Using Renode Logs and Their Output

- Now there's a log producing a very large output file
- Only want to actually look at a small part of it / check if a value appears
- Use a script to parse it!



```
1 12:42:14 [INFO] sysbus: Logging already enabled: False.
2 12:42:45 [DEBUG] mmuart0: WriteByte to 0x4 (ModemControl), value 0x0.
3 12:42:45 [DEBUG] mmuart0: WriteByte to 0x8 (TriggerLevelScratchpad+0x1), value 0x0.
4 12:42:45 [DEBUG] mmuart0: ReadByte from 0x8 (TriggerLevelScratchpad+0x1), returned 0x1.
5 12:42:45 [DEBUG] mmuart0: WriteByte to 0x8 (TriggerLevelScratchpad+0x1), value 0x3.
6 12:42:45 [DEBUG] mmuart0: ReadByte from 0x8 (TriggerLevelScratchpad+0x1), returned 0xC1.
7 12:42:45 [DEBUG] mmuart0: WriteByte to 0x8 (TriggerLevelScratchpad+0x1), value 0xC5.
8 12:42:45 [DEBUG] mmuart0: ReadByte from 0x8 (TriggerLevelScratchpad+0x1), returned 0xC1.
9 12:42:45 [DEBUG] mmuart0: WriteByte to 0x8 (TriggerLevelScratchpad+0x1), value 0xC1.
10 12:42:45 [DEBUG] mmuart0: ReadByte from 0x10 (TriggerLevelScratchpad+0x9), returned 0x8.
11 12:42:45 [DEBUG] mmuart0: WriteByte to 0x10 (TriggerLevelScratchpad+0x9), value 0x8.
12 12:42:45 [DEBUG] mmuart0: ReadByte from 0x10 (TriggerLevelScratchpad+0x9), returned 0x8.
13 12:42:45 [DEBUG] mmuart0: WriteByte to 0x10 (TriggerLevelScratchpad+0x9), value 0x8.
14 12:42:45 [WARNING] mmuart0: Unhandled read from offset 0x34.
15 12:42:45 [DEBUG] mmuart0: ReadByte from 0x34 (LineStatusHack+0x20), returned 0x0.
16 12:42:45 [DEBUG] mmuart0: WriteByte to 0x34 (LineStatusHack+0x20), value 0x0.
17 12:42:45 [WARNING] mmuart0: Unhandled read from offset 0x34.
18 12:42:45 [DEBUG] mmuart0: ReadByte from 0x34 (LineStatusHack+0x20), returned 0x0.
19 12:42:45 [DEBUG] mmuart0: WriteByte to 0x34 (LineStatusHack+0x20), value 0x0.
20 12:42:45 [WARNING] mmuart0: Unhandled read from offset 0x38.
21 12:42:45 [DEBUG] mmuart0: ReadByte from 0x38 (LineStatusHack+0x24), returned 0x0.
22 12:42:45 [DEBUG] mmuart0: WriteByte to 0x38 (LineStatusHack+0x24), value 0x0.
23 12:42:45 [WARNING] mmuart0: Unhandled write to offset 0x38, value 0x0.
24 12:42:45 [WARNING] mmuart0: Unhandled read from offset 0x30.
25 12:42:45 [DEBUG] mmuart0: ReadByte from 0x30 (LineStatusHack+0x1c), returned 0x0.
26 12:42:45 [DEBUG] mmuart0: WriteByte to 0x30 (LineStatusHack+0x1c), value 0x0.
27 12:42:45 [WARNING] mmuart0: Unhandled read from offset 0x30.
28 12:42:45 [DEBUG] mmuart0: ReadByte from 0x30 (LineStatusHack+0x1c), returned 0x0.
29 12:42:45 [DEBUG] mmuart0: WriteByte to 0x30 (LineStatusHack+0x1c), value 0x0.
30 12:42:45 [WARNING] mmuart0: Unhandled read from offset 0x30.
31 12:42:45 [DEBUG] mmuart0: ReadByte from 0x30 (LineStatusHack+0x1c), returned 0x0.
32 12:42:45 [DEBUG] mmuart0: WriteByte to 0x30 (LineStatusHack+0x1c), value 0x0.
33 12:42:45 [WARNING] mmuart0: Unhandled read from offset 0x38.
34 12:42:45 [DEBUG] mmuart0: ReadByte from 0x38 (LineStatusHack+0x24), returned 0x0.
35 12:42:45 [DEBUG] mmuart0: WriteByte to 0x38 (LineStatusHack+0x24), value 0x0.
36 12:42:45 [WARNING] mmuart0: Unhandled write to offset 0x38, value 0x0.
37 12:42:45 [DEBUG] mmuart0: WriteByte to 0x44 (LineStatusHack+0x30), value 0x0.
38 12:42:45 [DEBUG] mmuart0: WriteByte to 0x48 (LineStatusHack+0x34), value 0x0.
39 12:42:45 [DEBUG] mmuart0: WriteByte to 0x4c (LineStatusHack+0x38), value 0x0.
40 12:42:45 [DEBUG] mmuart0: ReadByte from 0xc (TriggerLevelScratchpad+0x5), returned 0x0.
41 12:42:45 [DEBUG] mmuart0: WriteByte to 0xc (TriggerLevelScratchpad+0x5), value 0x80.
42 12:42:45 [DEBUG] mmuart0: WriteByte to 0x4 (ModemControl), value 0x0.
43 12:42:45 [DEBUG] mmuart0: WriteByte to 0x0 (Data), value 0xA.
44 12:42:45 [DEBUG] mmuart0: ReadByte from 0xc (TriggerLevelScratchpad+0x5), returned 0x0.
45 12:42:45 [DEBUG] mmuart0: WriteByte to 0xc (TriggerLevelScratchpad+0x5), value 0x0.
46 12:42:45 [WARNING] mmuart0: Unhandled read from offset 0x30.
```

# Using Renode Logs and Their Output

```
C:\Windows\System32\cmd.exe

C:\Microsemi\SoftConsole_v6.0\extras\workspace.examples\pse-blinky>py renode_decoder.py

Renode UART decoder complete
Stats:
Characters decoded: 14477
Lines decoded: 375
Non ASCII characters decoded: 25

C:\Microsemi\SoftConsole_v6.0\extras\workspace.examples\pse-blinky>
```



```
C:\Microsemi\SoftConsole_v6.0\renode\uart.txt - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?

uart.txt
1 12:42:14 [INFO] yabuzai: Logging already enabled: False.
2 12:42:45 [DEBUG] mmuart0: WriteByte to 0x4 (ModeControl), value 0x0.
3 12:42:45 [DEBUG] mmuart0: WriteByte to 0x8 (TriggerLevelScratchpad0x0), value 0x0.
4 12:42:45 [DEBUG] mmuart0: ReadByte from 0x8 (TriggerLevelScratchpad0x0), returned 0x1.
5 12:42:45 [DEBUG] mmuart0: WriteByte to 0x8 (TriggerLevelScratchpad0x0), value 0x3.
6 12:42:45 [DEBUG] mmuart0: ReadByte from 0x8 (TriggerLevelScratchpad0x0), returned 0xc1.
7 12:42:45 [DEBUG] mmuart0: WriteByte to 0x8 (TriggerLevelScratchpad0x0), value 0xc5.
8 12:42:45 [DEBUG] mmuart0: ReadByte from 0x8 (TriggerLevelScratchpad0x0), returned 0xc1.
9 12:42:45 [DEBUG] mmuart0: WriteByte to 0x8 (TriggerLevelScratchpad0x0), value 0xc1.
10 12:42:45 [DEBUG] mmuart0: ReadByte from 0x10 (TriggerLevelScratchpad0x8), returned 0x8.
11 12:42:45 [DEBUG] mmuart0: WriteByte to 0x10 (TriggerLevelScratchpad0x8), value 0x8.
12 12:42:45 [DEBUG] mmuart0: ReadByte from 0x10 (TriggerLevelScratchpad0x8), returned 0x8.
13 12:42:45 [DEBUG] mmuart0: WriteByte to 0x10 (TriggerLevelScratchpad0x8), value 0x8.
14 12:42:45 [WARNING] mmuart0: Unhandled read from offset 0x34.
15 12:42:45 [DEBUG] mmuart0: ReadByte from 0x34 (LineStatusAck0x20), returned 0x0.
16 12:42:45 [DEBUG] mmuart0: WriteByte to 0x34 (LineStatusAck0x20), value 0x0.
17 12:42:45 [WARNING] mmuart0: Unhandled read from offset 0x34.
18 12:42:45 [DEBUG] mmuart0: ReadByte from 0x34 (LineStatusAck0x20), returned 0x0.
19 12:42:45 [DEBUG] mmuart0: WriteByte to 0x34 (LineStatusAck0x20), value 0x0.
20 12:42:45 [WARNING] mmuart0: Unhandled read from offset 0x38.
21 12:42:45 [DEBUG] mmuart0: ReadByte from 0x38 (LineStatusAck0x24), returned 0x0.
22 12:42:45 [DEBUG] mmuart0: WriteByte to 0x38 (LineStatusAck0x24), value 0x0.
23 12:42:45 [WARNING] mmuart0: Unhandled write to offset 0x38, value 0x0.
24 12:42:45 [WARNING] mmuart0: Unhandled read from offset 0x30.
25 12:42:45 [DEBUG] mmuart0: ReadByte from 0x30 (LineStatusAck0x1c), returned 0x0.
26 12:42:45 [DEBUG] mmuart0: WriteByte to 0x30 (LineStatusAck0x1c), value 0x0.
27 12:42:45 [WARNING] mmuart0: Unhandled read from offset 0x30.
```

```
renode_decoder.py
1 import os
2 import sys
3
4 def renode_uart_decode():
5     if os.path.isfile(".././../renode/uart.txt"):
6         uart_file = open(".././../renode/uart.txt", "r")
7         output_file = open("../converted_renode_output.txt", "w")
8         buffer = ""
9         chars = 0
10        new_lines = 0
11        non_ascii_chars = 0
12
13        for line in uart_file:
14            chars = chars + 1
15            if "mmuart0: WriteByte" in line:
16                temp = line.strip()
17                temp = temp.split("value 0x")
18                temp = temp[1]
19                temp = str(temp[0])
20                hex = int(temp, 16)
21                if not hex < 11 and not hex > 127:
22                    buffer = buffer + str(bytearray.fromhex(temp).decode())
23            elif hex == 10:
24                output_file.write(buffer + "\n")
25                buffer = ""
26                new_lines = new_lines + 1
27            elif not hex == 13:
28                buffer = buffer
29                non_ascii_chars = non_ascii_chars + 1
30        output_file.close()
31
32        print("\nRenode UART decoder complete")
33        print("Stats:")
34        print("Characters decoded: " + str(chars))
35        print("Lines decoded: " + str(new_lines))
36        print("Non ASCII characters decoded: " + str(non_ascii_chars))
37    else:
38        print("File not found!")
39    return
40
41 renode_uart_decode()
```

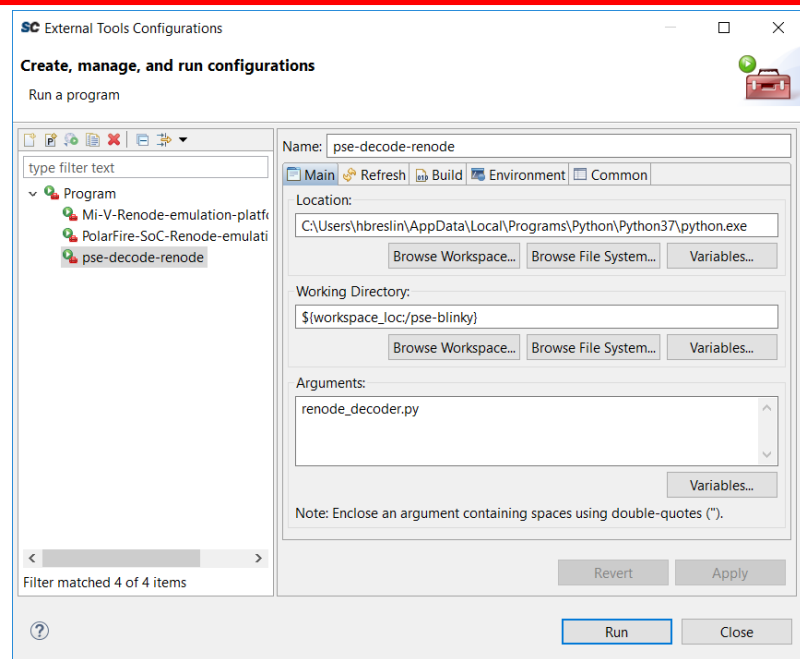


```
C:\Microsemi\SoftConsole_v6.0\extras\workspace.examples\pse-blinky\converted_renode_output.txt - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?

renode_decoder.py
1 Hello World from ebl (uart 0).
2 Starting u54_1 (uart 1) and u54_2 (uart 2).
3 Setting outputs 0, 1 and 2 to high
4 Setting outputs 0, 1 and 2 to low
5 Setting outputs 0, 1 and 2 to high
6 Setting outputs 0, 1 and 2 to low
7 Setting outputs 0, 1 and 2 to high
8 Setting outputs 0, 1 and 2 to low
9 Setting outputs 0, 1 and 2 to high
10 Setting outputs 0, 1 and 2 to low
11 Setting outputs 0, 1 and 2 to high
12 Setting outputs 0, 1 and 2 to low
13 Setting outputs 0, 1 and 2 to high
14 Setting outputs 0, 1 and 2 to low
15 Setting outputs 0, 1 and 2 to high
16 Setting outputs 0, 1 and 2 to low
17 Setting outputs 0, 1 and 2 to high
18 Setting outputs 0, 1 and 2 to low
19 Setting outputs 0, 1 and 2 to high
20 Setting outputs 0, 1 and 2 to low
21 Setting outputs 0, 1 and 2 to high
22 Setting outputs 0, 1 and 2 to low
23 Setting outputs 0, 1 and 2 to high
24 Setting outputs 0, 1 and 2 to low
25 Setting outputs 0, 1 and 2 to high
26 Setting outputs 0, 1 and 2 to low
27 Setting outputs 0, 1 and 2 to high
28 Setting outputs 0, 1 and 2 to low
```

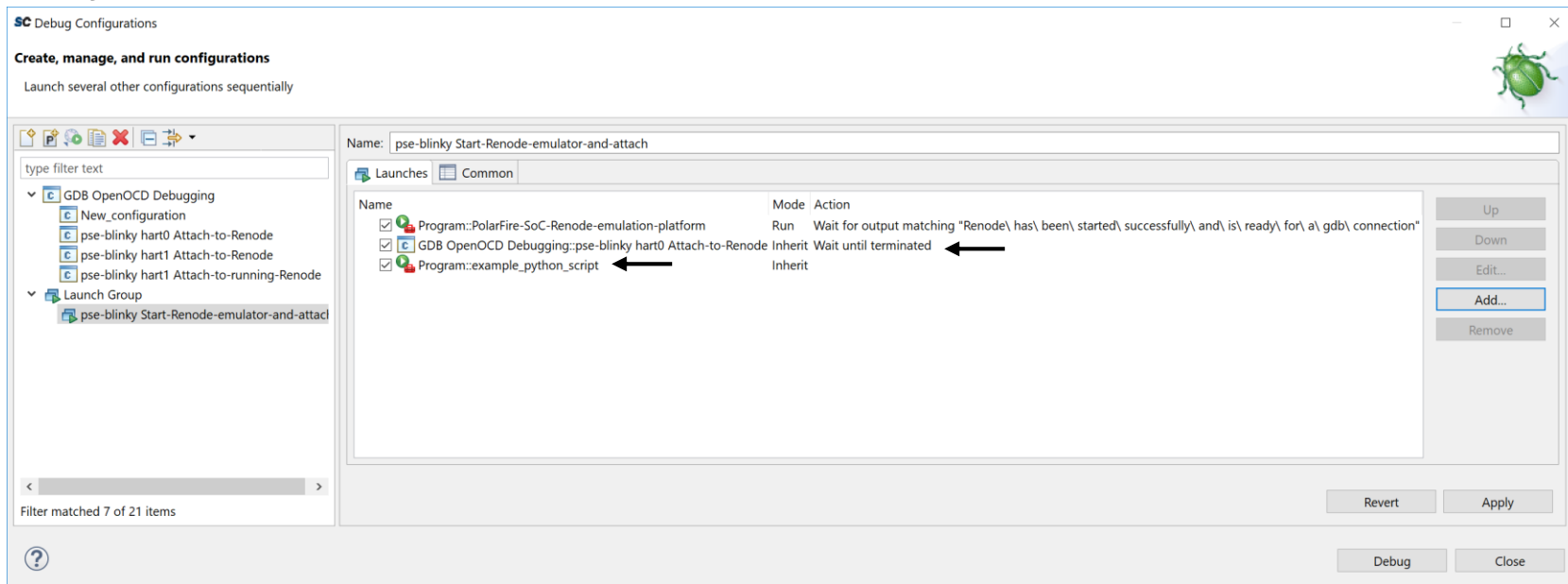
# Using Renode Logs and Their Output

- **Python scripts can be added as external tools**
  - Add the script to a launch group and run it after your debug session has been terminated
- **Add the Python script as a hook in Renode**



# Using Renode Logs and Their Output

- The python external tool can be called as part of a launch group
- They will then run from the Console in SoftConsole



# First Thursdays

---

May 2 - Webinar 1: Discover Renode for PolarFire® SoC Design and Debug

June 6 - Webinar 2: How to Get Started with Renode for PolarFire SoC

July 4 - Webinar 3: Learn to Debug a Bare-Metal PolarFire SoC Application with Renode

Aug. 1 - Webinar 4: Tips and Tricks for Even Easier PolarFire SoC Debug with Renode

**Sept. 5 - Webinar 5: Add and Debug PolarFire SoC Peripherals with Renode**

**Oct. 3 - Webinar 6: Add and debug a pre-existing peripheral in PolarFire SoC**

**We are planning our next series and would like your input on the content presented so far and what you would like to see covered**



**Thank You!**

**Questions?**

