# 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 models with Renode

**Oct. 3 - Webinar 6: Add and Debug and Pre-Existing model in PolarFire SoC**

**Nov. 7 - Webinar 7: How to write custom models – filters, offloading, acceleration etc**

**Dec. 5 - Webinar 8: Handling Binaries**

**Contd.**

# Second Thursdays

**Jan. 9 - Webinar 9: Run Linux on Renode (PolarFire SoC Model as a Quad-core SMP) – this is not a Linux / Buildroot tutorial**

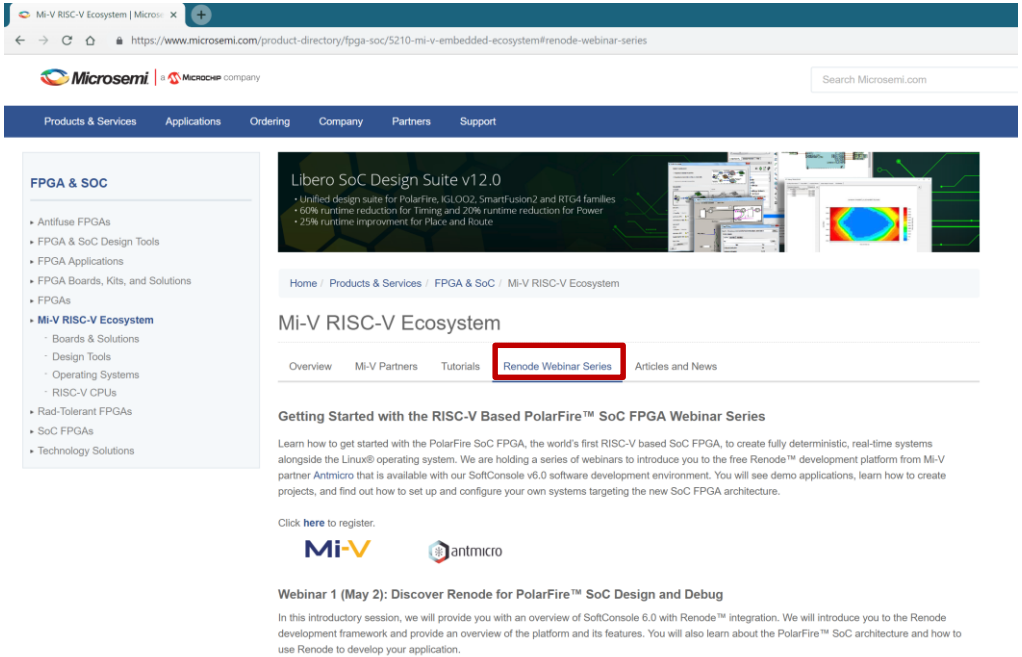
**Feb. 13 - Webinar 10: Build applications for Linux on PolarFire SoC**

**Mar. 12 - Webinar 11: Introduction to PolarFire SoC MSS Configuration and Software Flow**

**Apr. 9 - Webinar 12: Two baremetal Applications on PolarFire SoC**

**May 14 - Webinar 13: Linux + Real-Time (AMP Mode) on PolarFire SoC**

# Supporting Content




www.microsemi.com/Mi-V "Renode Webinar Series"

# Introduction

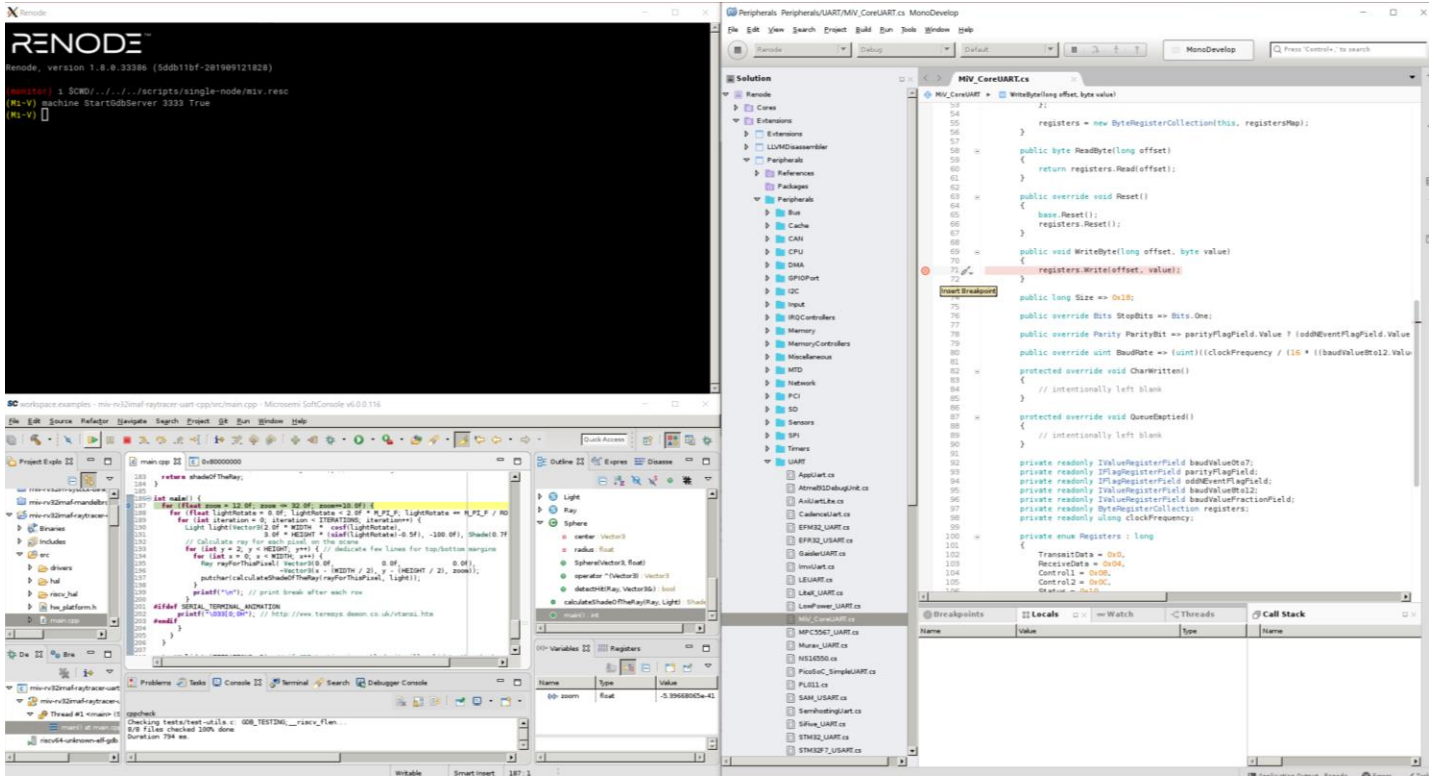
- Where are models in Renode
- Ways to add models
- How to add a Just In Time (JIT) compiled model
- Debugging a model using MonoDevelop
- Debugging a model using logs

# But first!

- **Have you watched our supporting video?**

- **A supporting video is available in the “Renode Webinar Series” page of the Mi-V ecosystem that outlines the set up steps required to use the tools shown in the webinar**

# But first!

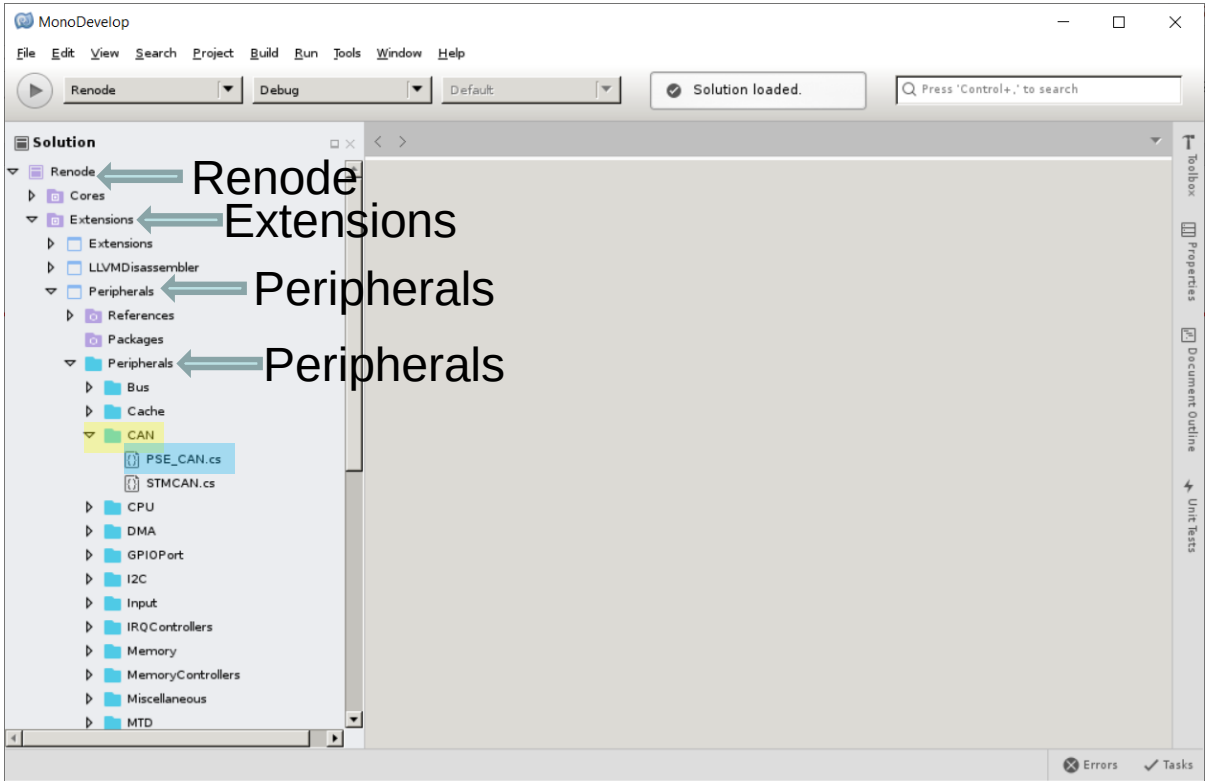

RENODE
Renode, version 1.8.0.33386 (5ddb11bf-201909121828)
(monitor) i $CWD/../../../scripts/single-node/miv.resc
(Mi-V) machine StartGdbServer 3333 True
(Mi-V)


# Where are models in Renode

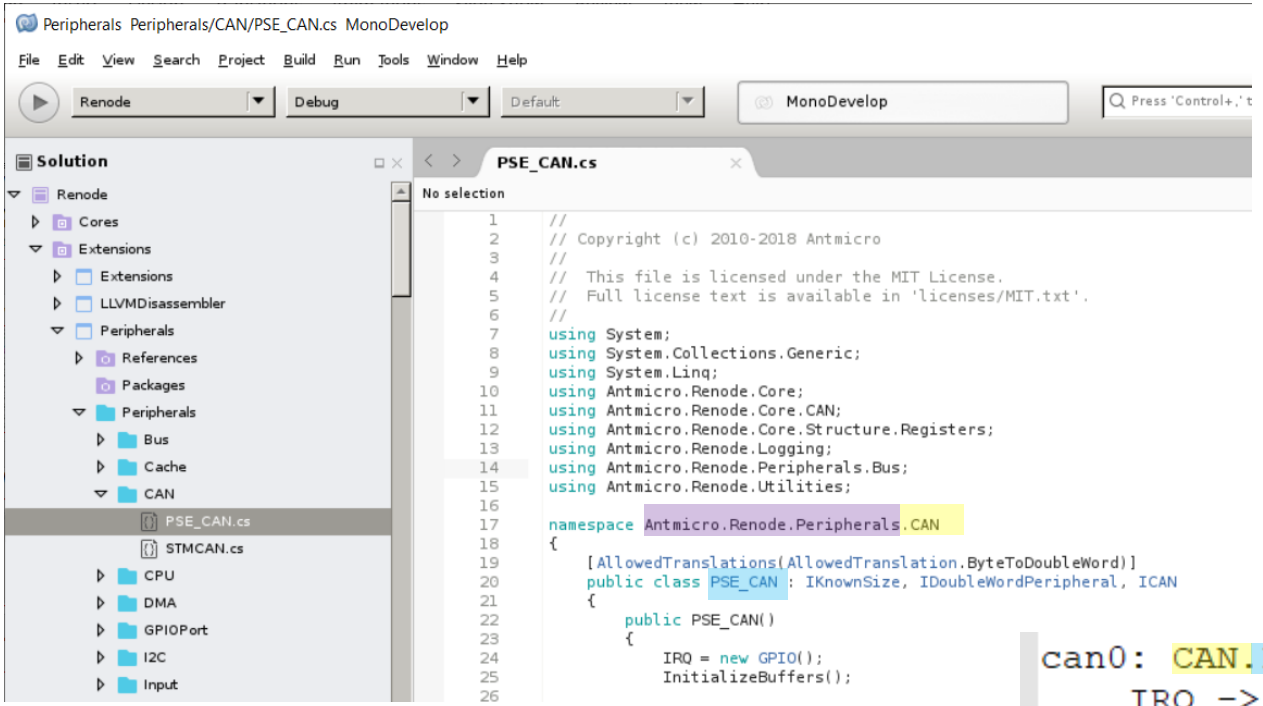
# Where are models in Renode

- Models can be added to Renode before the software is compiled
- They are then built with the software
- The software source code is available on GitHub
- https://github.com/renode

# Where are models in Renode

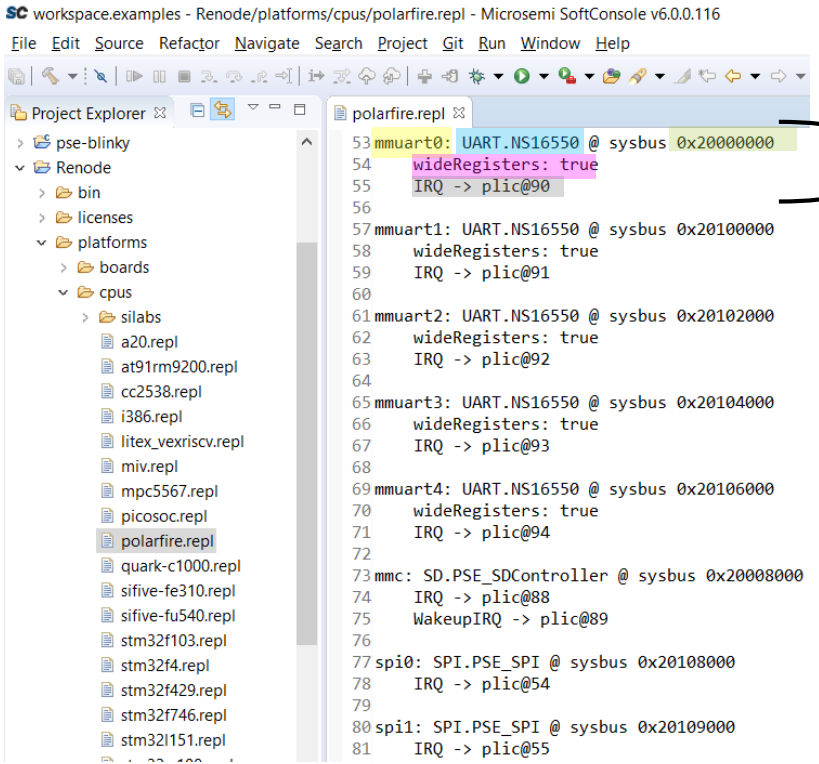


# Where are models in Renode




```
can0: CAN.PSE_CAN @ sysbus 0x2010C000
    IRQ -> plic@56
```

# Where are models in Renode




ADD UART to the system

**Connecting a peripheral:**

It depends on the model!

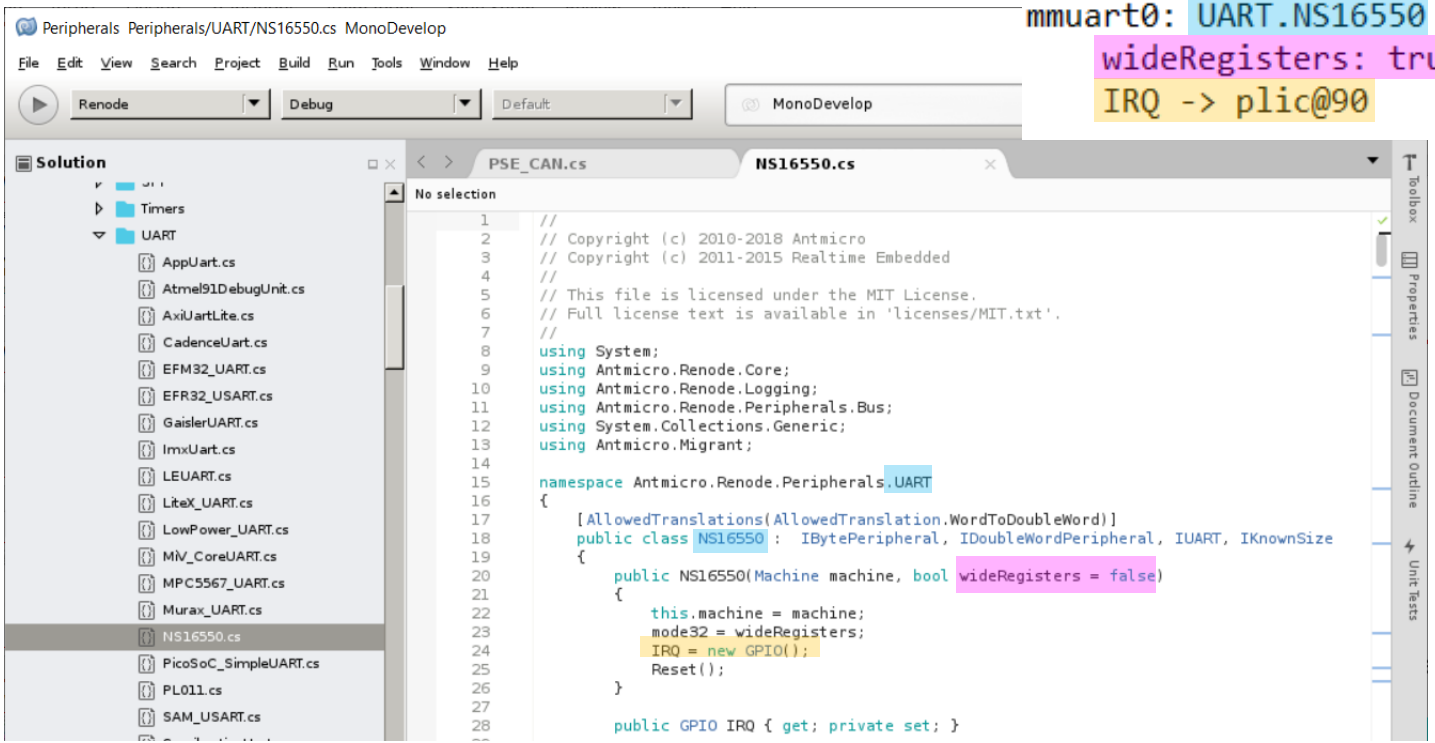
[name]: [namespace].[name] @ sysbus [address]
[parameters0]
[constructors0]
etc.

Parameters affect the peripherals operation
Some are optional, depending on default values

Constructors connect parts of the system together
E.g IRQ out to PLIC

# Where are models in Renode

```
mmuart0: UART.NS16550 @ sysbus 0x20000000
    wideRegisters: true
    IRQ -> plic@90
```



# Where are models in Renode

PSE_CAN.cs | NS16550.cs

selection

```
//
// Copyright (c) 2010-2018 Antmicro
// Copyright (c) 2011-2015 Realtime Embedded
//
// This file is licensed under the MIT License.
// Full license text is available in 'licenses/MIT.txt'.
//
using System;
using Antmicro.Renode.Core;
using Antmicro.Renode.Logging;
using Antmicro.Renode.Peripherals.Bus;
using System.Collections.Generic;
using Antmicro.Migrant;

namespace Antmicro.Renode.Peripherals.UART
{
    [AllowedTranslations(AllowedTranslation.WordToDoubleWord)]
    public class NS16550 :  IBytePeripheral, IDoubleWordPeripheral, IUART, IKnownSize
    {
        public NS16550(Machine machine, bool wideRegisters = false)
        {
            this.machine = machine;
            mode32 = wideRegisters;
            IRQ = new GPIO();
            Reset();
        }
```

Default value

Overwritten

Constructor

```
mmuart0: UART.NS16550 @ sysbus 0x20000000
    wideRegisters: true
    IRQ -> plic@90
```

```
wdog0: Timers.PSE_Watchdog @ sysbus 0x20001000
    frequency: 156250
    RefreshEnable -> plic@100 | e51@26
    Trigger -> plic@105 | e51@25
```

PSE_CAN.cs | NS16550.cs | PSE_Watchdog.cs

E_Watchdog ▸ DefineRegisters()

```
// This file is licensed under the MIT License.
// Full license text is available in 'licenses/MIT.txt'.
//
using System;
using Antmicro.Renode.Core;
using Antmicro.Renode.Peripherals.Bus;
using Antmicro.Renode.Core.Structure.Registers;
using Antmicro.Renode.Time;
using Antmicro.Renode.Logging;

namespace Antmicro.Renode.Peripherals.Timers
{
    public class PSE_Watchdog : BasicDoubleWordPeripheral, IKnownSize
    {
        public PSE_Watchdog(Machine machine, long frequency) : base(machine)
        {
            internalTimer = new LimitTimer(machine.ClockSource, frequency, this, String.Empty, T
            internalTimer.LimitReached += TimerLimitReached;

            RefreshEnable = new GPIO();
            Trigger = new GPIO();
        }

        public override void Reset()
        {
            base.Reset();
            Trigger.Unset();
            RefreshEnable.Unset();
            state = State.ForbiddenRegion;
            internalTimer.Reset();
        }

        public long Size => 0x1000;
```

No default value – must be provided

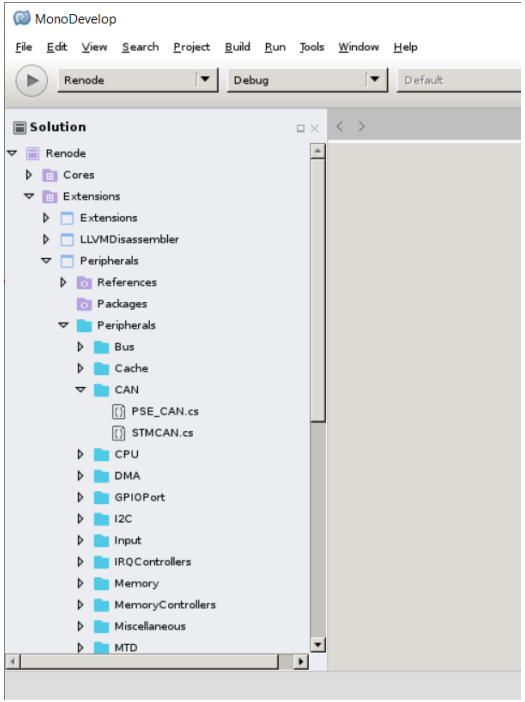

# Ways To Add models

# Ways To Add models

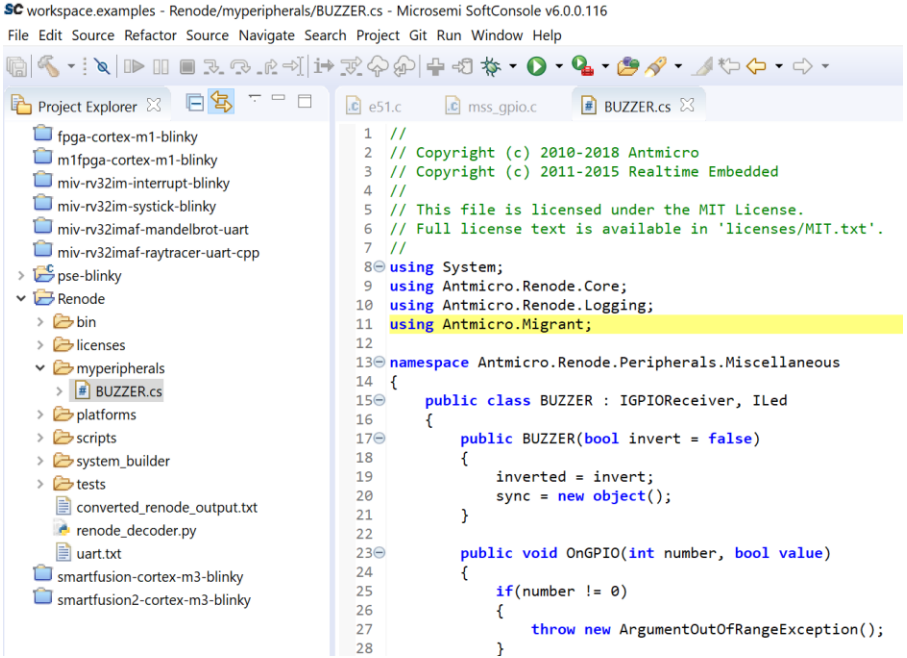
| Pre-compiled | JiT (Just in Time compiled) |
|---|---|
| Built in with Renode | Standalone file separate to Renode |
| Run faster | Can be modified without having to re-build |
| Can’t be edited without rebuilding Renode | Run slower than pre-compiled |
| Models included with Renode are precompiled | Develop models using JiT and then build them into Renode |
| Available on Windows and Linux | Only available on Linux |

# Ways To Add models

## Pre-compiled




## JiT




# How to add a Just In Time (JIT) compiled model

# How to add a Just In Time (JIT) compiled model

- **When to use @ in Renode:**
  - If using a relative or full path:
    - E.g “include @../../../scripts/single…….”
    - E.g “include @C:/Microsemi/SoftConsole_v6.0/….”
  - Not needed if using CWD:
    - E.g “include $CWD/../scripts/single….”

# How to add a Just In Time (JIT) compiled model

- **Include the C# file for the model**
  - include @[path_to_file]
- **Add the model to the system**
  - machine LoadPlatformDescriptionFromString “[sysbus_name]: [class].[name] @ sysbus [address]”

polarfire.repl

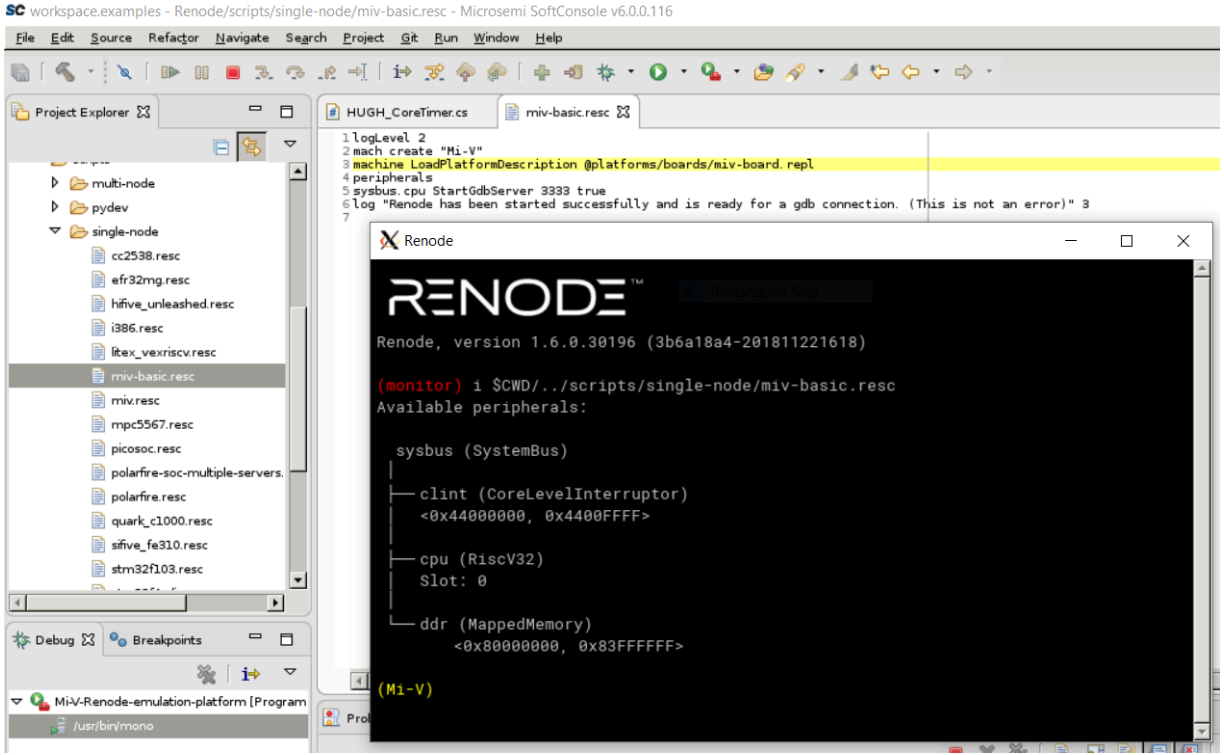
```
53 mmuart0: UART.NS16550 @ sysbus 0x20000000
54     wideRegisters: true
55     IRQ -> plic@90
56
```

[sysbus_name]: [namespace].[name] @ sysbus [address]

# How to add a Just In Time (JIT) compiled model

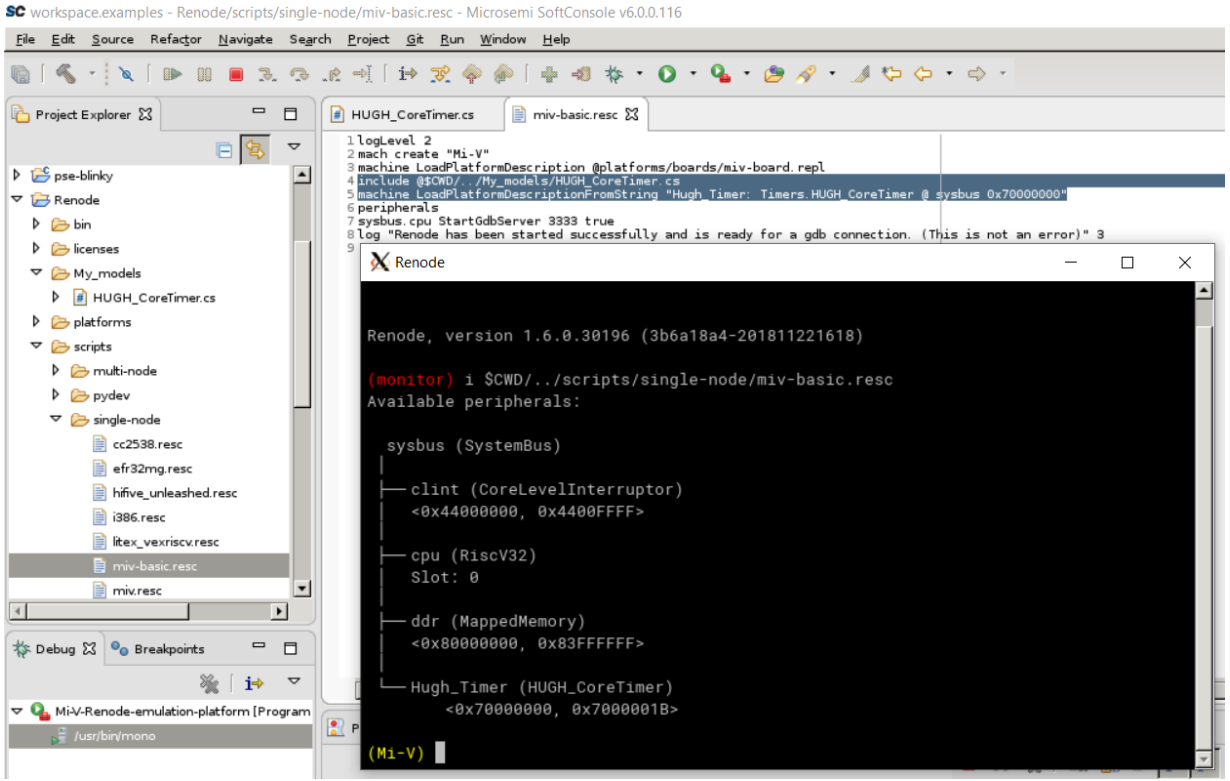
- **Commands used:**
  1. include $CWD/../My_models/HUGH_CoreTimer.cs
  2. machine LoadPlatformDescriptionFromString “Hugh_timer: Timers.Hugh_CoreTimer @sysbus 0x70000000”

# How to add a Just In Time (JIT) compiled model




**Alternatively add commands to the launch script**

# How to add a Just In Time (JIT) compiled model




**Alternatively add commands to the launch script**


# Debugging a model using MonoDevelop

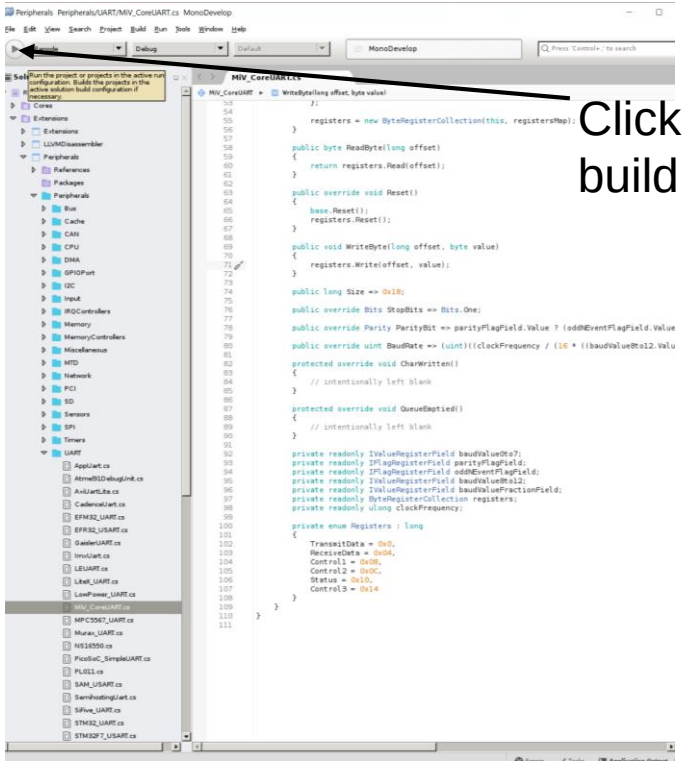
# Debugging a model using MonoDevelop

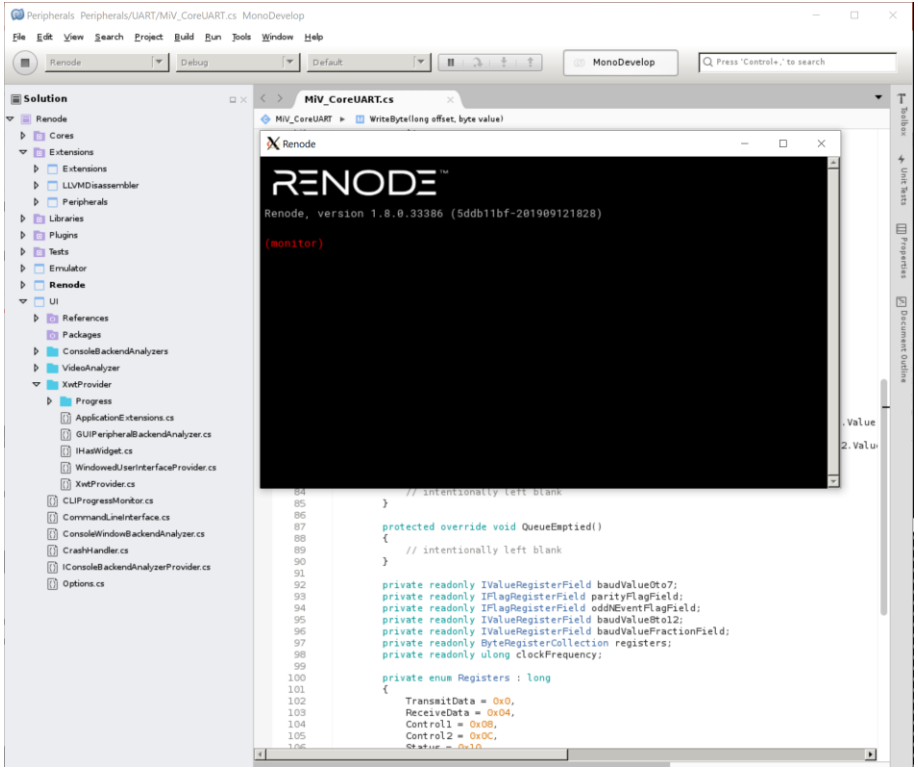
- **Two options for debugging a model:**
  1. Build Renode from source
     - Full visibility of  variables
     - Can suspend execution and debug
  2. Use logs
     - Can set log levels for peripherals and use the logs to debug

# Debugging a model using MonoDevelop

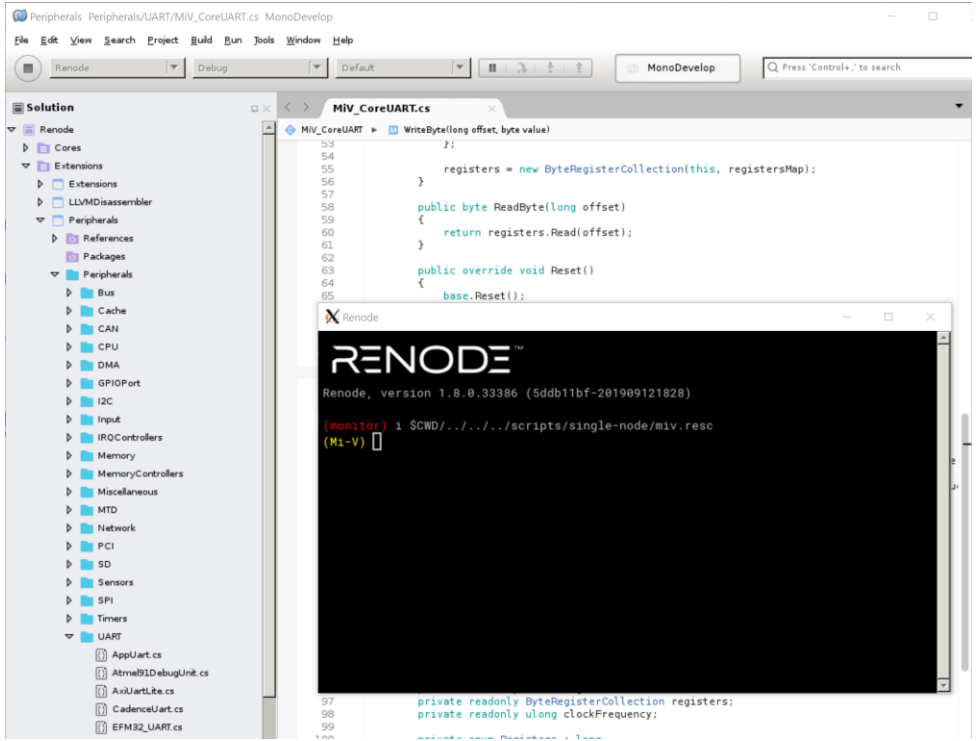
1. Follow the steps in the supporting video to build Renode from source (build v1.7.1 to match SC 6.0)
2. Launch Renode from your C# IDE (e.g monodevelop) and load the platform
3. Start the GDB server for debugging
4. Run the “attach to renode” debug session in SoftConsole
5. Set a breakpoint in the model being tested

# Debugging a model using MonoDevelop



# Debugging a model using MonoDevelop



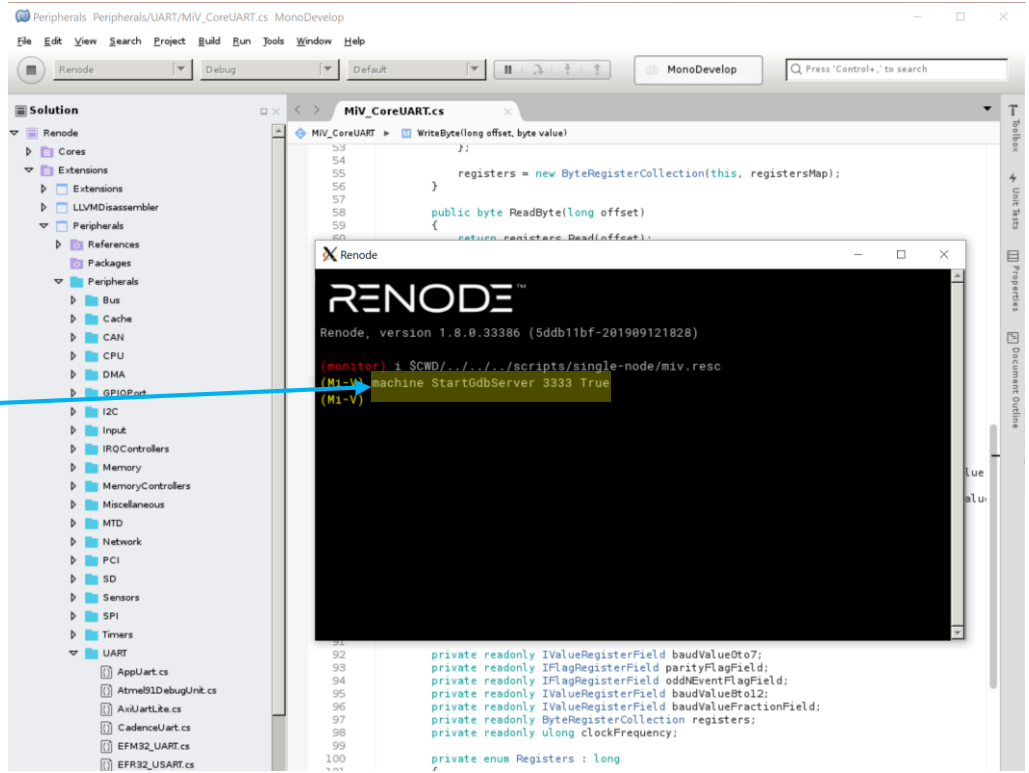

Paths to launch scripts:

**Mi-V system:**
$CWD/../../../scripts/single-node/miv.resc

**PolarFire system:**
$CWD/../../../scripts/single-node/polarfire.resc

# Debugging a model using MonoDevelop

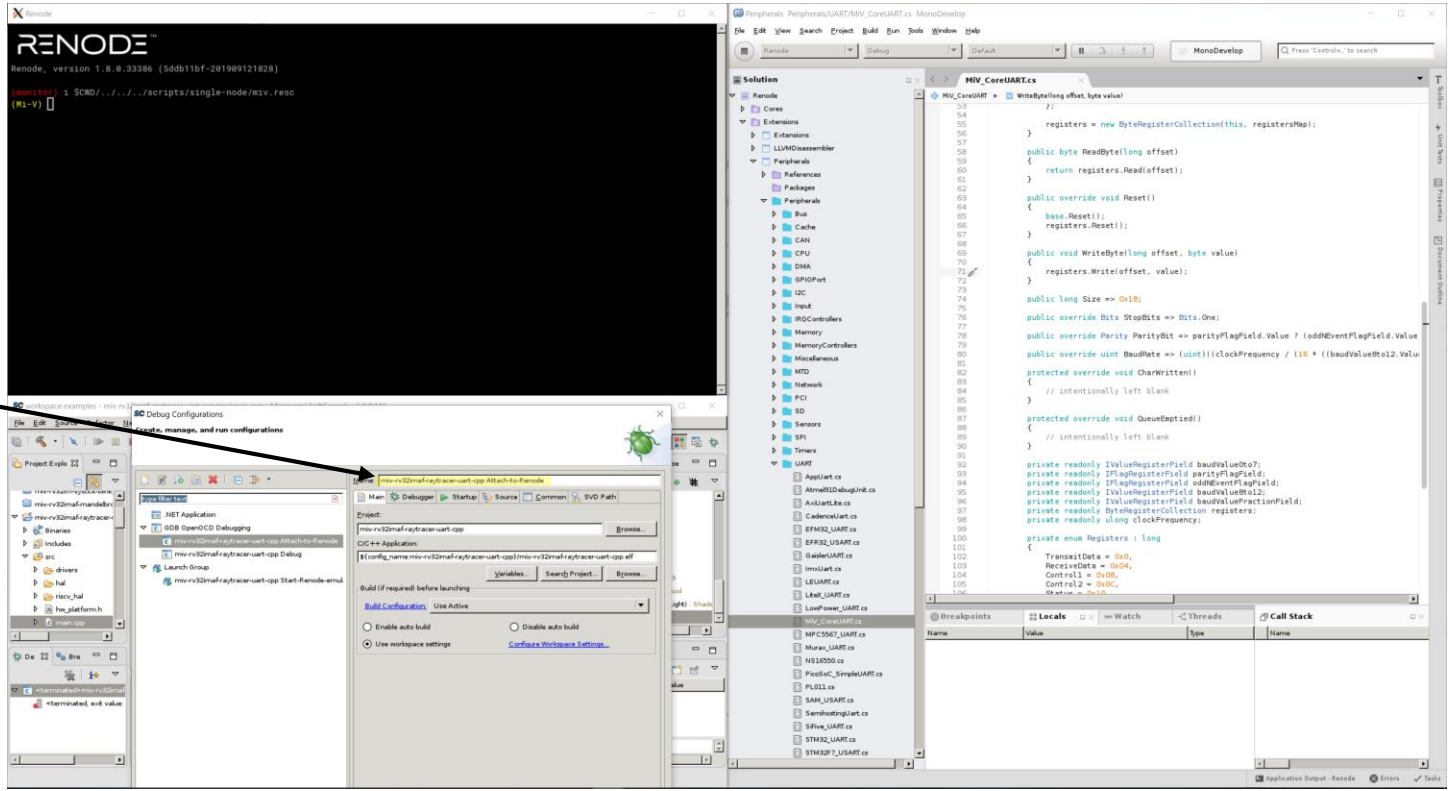
Start the GDB server for the machine



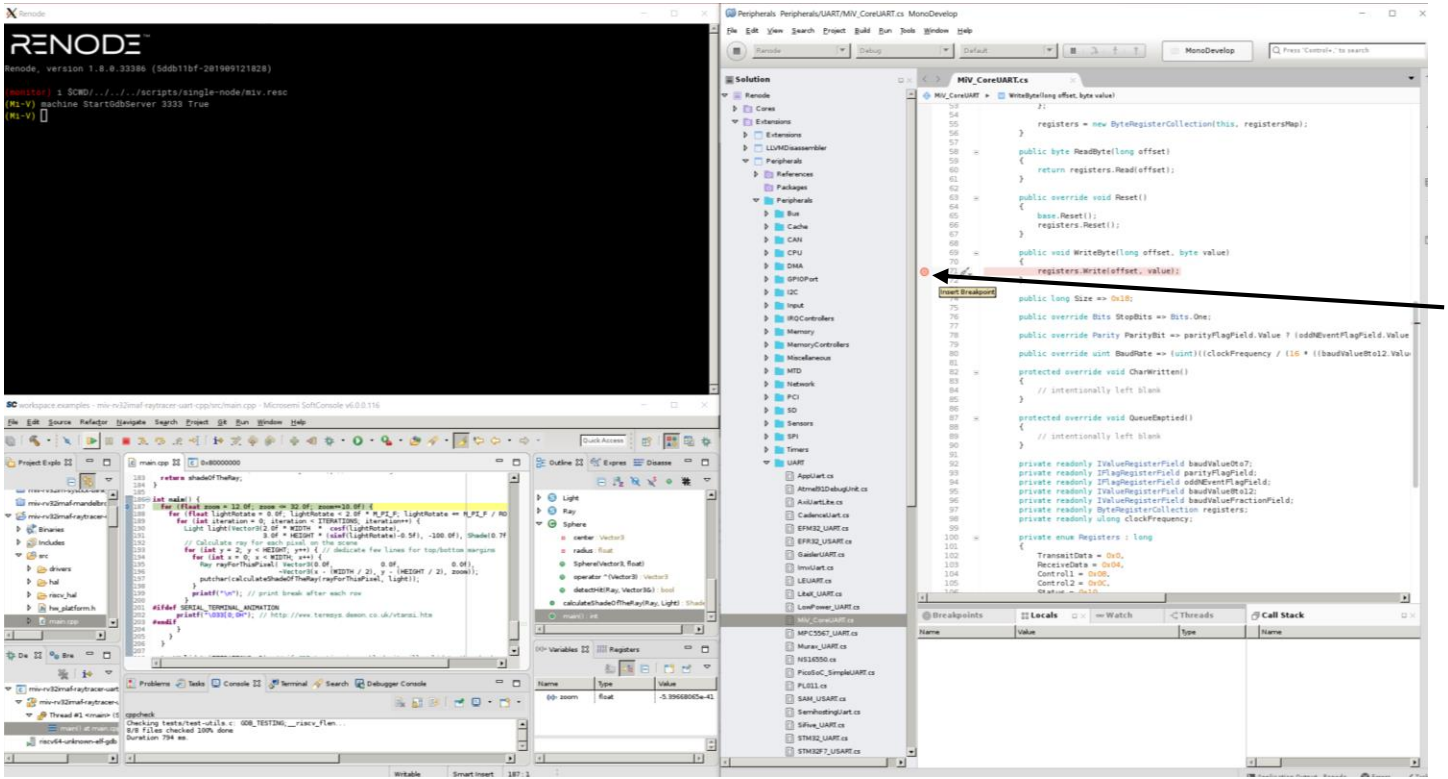
# Debugging a model using MonoDevelop

Use the “Attach-to-Renode” debug session to connect to Renode

Do not use the launch group – Renode is already running!

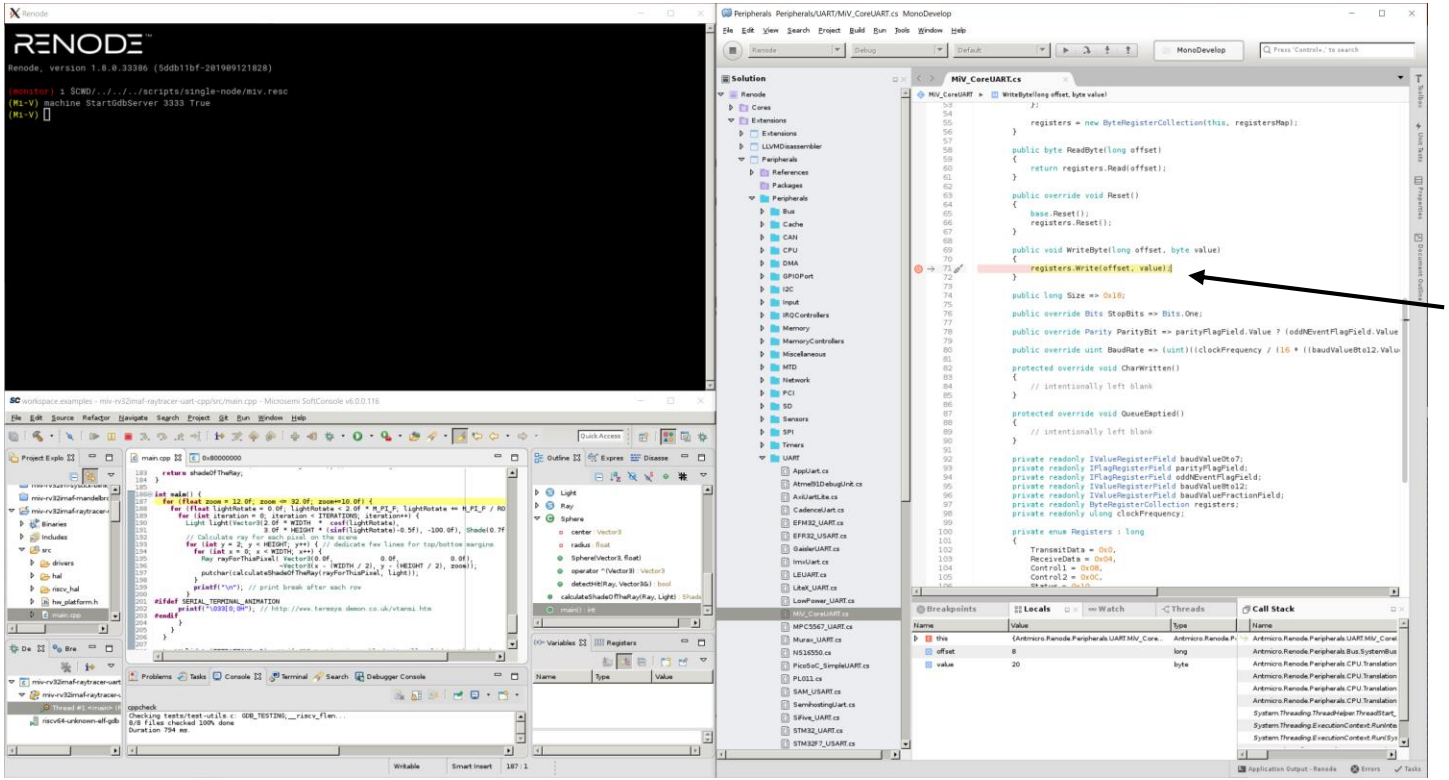


# Debugging a model using MonoDevelop



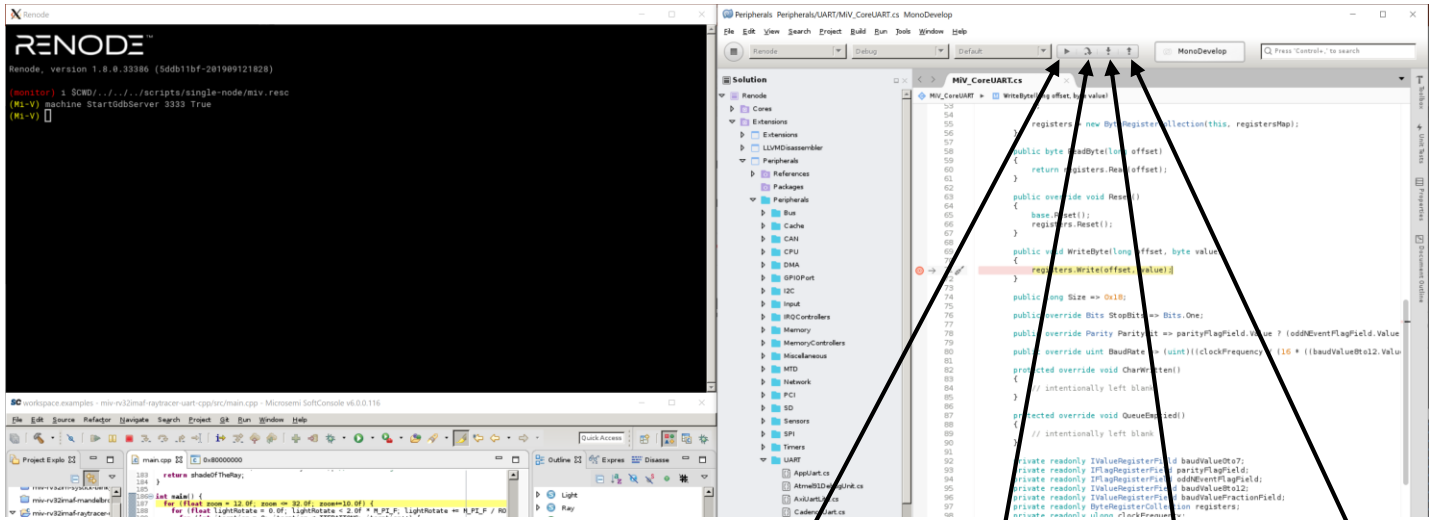

**Click on the left of a line to set a breakpoint**

# Debugging a model using MonoDevelop




Run software that uses the model so it reaches the breakpoint

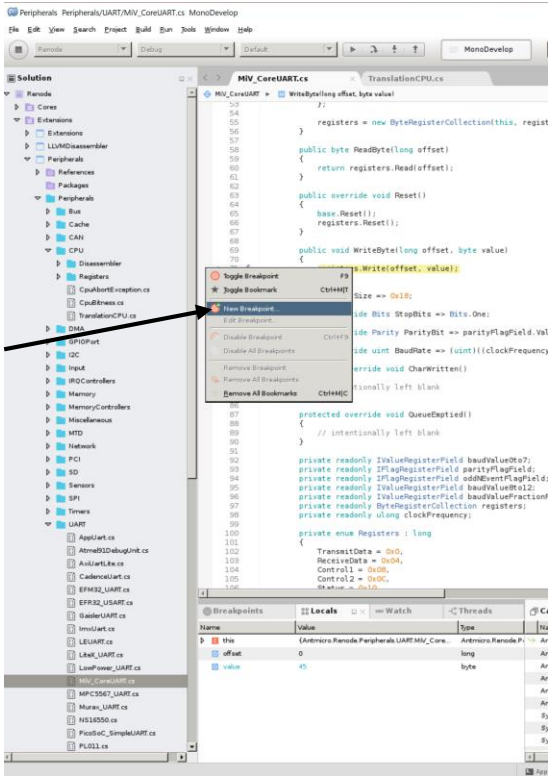
# Debugging a model using MonoDevelop

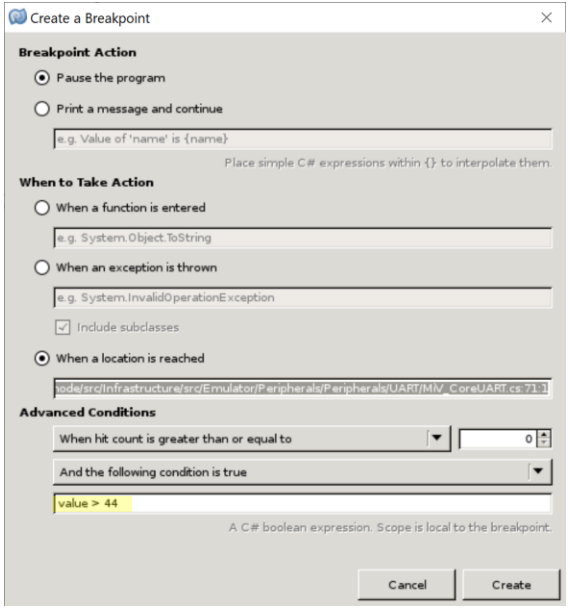


# Debugging a model using MonoDevelop

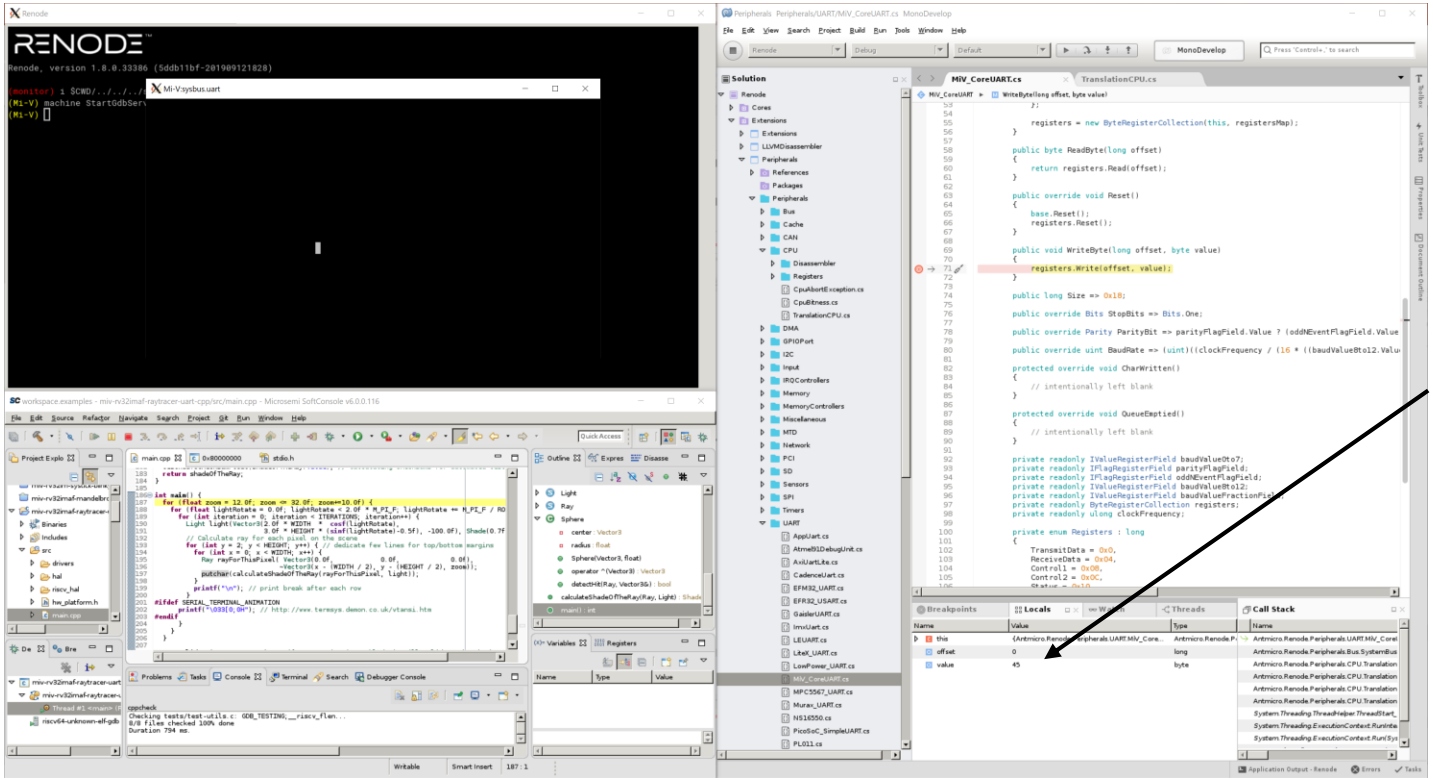
**Custom breakpoints can be created to only halt during certain conditions**

**Right click on the left and select “New Breakpoint…”**



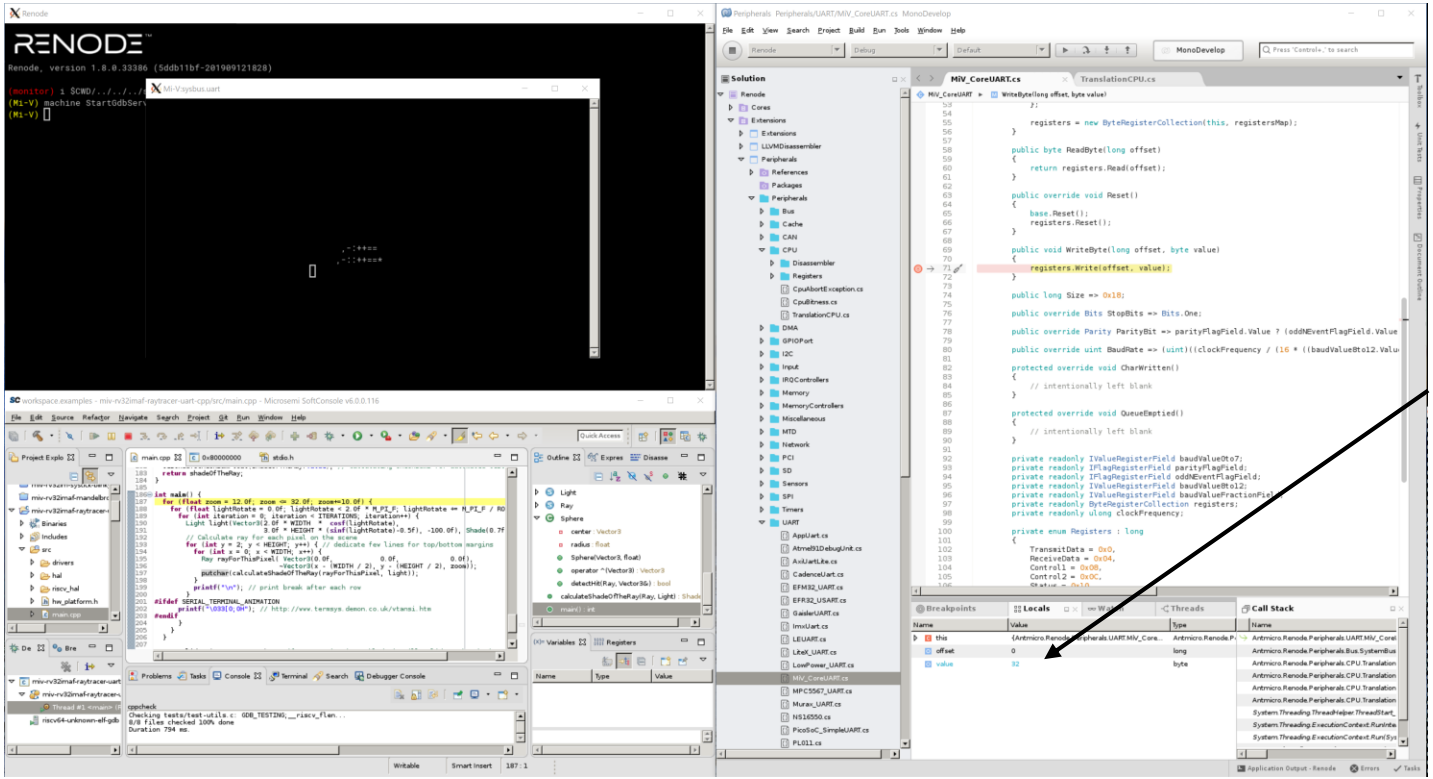



# Debugging a model using MonoDevelop



**Check variable values etc at the bottom**

# Debugging a model using MonoDevelop



Check variable values etc at the bottom


# Debugging a model using logs

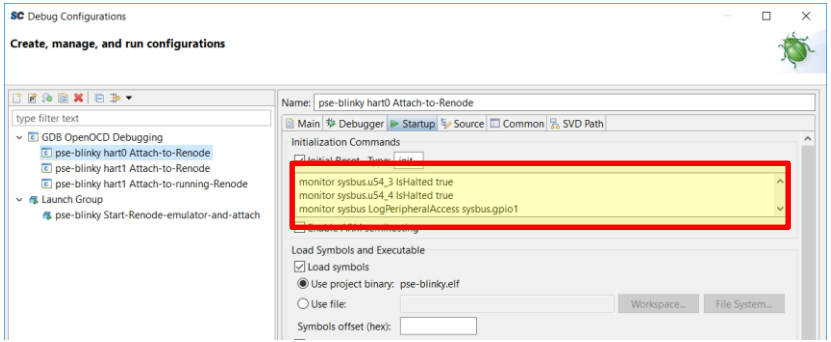
# Debugging a model using logs

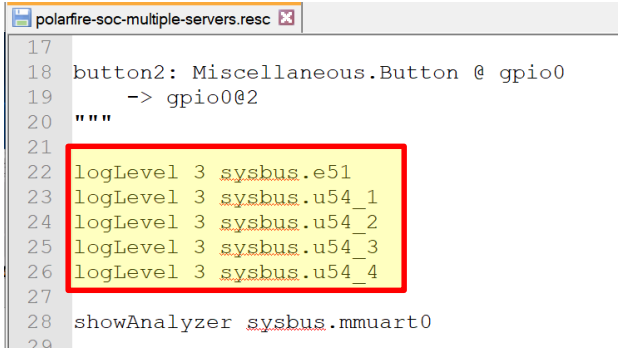
- **There are several logging levels available:**
  - -1 Noisy
  - 0 Debug
  - 1 Info
  - 2 Warning
  - 3 Error
- **Each can capture different levels of data**
- **Peripherals will produce a different level of log data depending on what has caused the log (e.g. uart print vs fatal error)**

# Debugging a model using logs

- **Logging (and other commands) can be set up automatically by passing the commands during start up or including them in the launch script**

- **E.G (pse-blinky debug configuration):**
  - monitor sysbus LogPeripheralAccess sysbus.gpio1
  - monitor logLevel -1 sysbus.gpio1
- **E.G (pse-blinky launch script):**
  - logLevel 3 sysbus.e51



# Debugging a model using logs

| Log Level | Log Peripheral Access |
| --- | --- |
| Produces different levels of content depending on log level selected | Creates a log entry every time the CPU tries to access a peripheral |
| Logging commands included in the peripheral model | Can be set individually per peripheral |
| Variable values can be included in the log | Log level must be set to 0 (debug) |
| | Peripheral must be on system bus |

# Debugging a model using logs

```
PolarFire-SoC-Renode-emulation-platform [Program] C:\Microsemi\SoftConsole_v6.0\renode\bin\Renode.exe
15:36:32.5299 [INFO] Including script: C:\Microsemi\SoftConsole_v6.0\renode\scripts\single-node\polarfire-soc-multiple-servers.resc
15:36:34.8709 [ERROR] Script: Renode has been started successfully and is ready for a gdb connection. (This is not an error)
15:36:36.3250 [ERROR] u54_1: CPU abort [PC=0x1000]: Trying to execute code outside RAM or ROM at 0x0000000000001000.
15:36:36.3250 [ERROR] u54_2: CPU abort [PC=0x1000]: Trying to execute code outside RAM or ROM at 0x0000000000001000.
```

Log level 3 (error)

```
PolarFire-SoC-Renode-emulation-platform [Program] C:\Microsemi\SoftConsole_v6.0\renode\bin\Renode.exe
15:42:20.5119 [INFO] Including script: C:\Microsemi\SoftConsole_v6.0\renode\scripts\single-node\polarfire-soc-multiple-servers.resc
15:42:22.9525 [ERROR] Script: Renode has been started successfully and is ready for a gdb connection. (This is not an error)
15:42:25.3827 [ERROR] u54_2: CPU abort [PC=0x1000]: Trying to execute code outside RAM or ROM at 0x0000000000001000.
15:42:25.3847 [ERROR] u54_1: CPU abort [PC=0x1000]: Trying to execute code outside RAM or ROM at 0x0000000000001000.
15:42:32.2031 [WARNING] gpio1: Unhandled write to offset 0x0. Unhandled bits: [2] when writing value 0x5. Tags: OutputBufferEnable (0x1).
15:42:32.2036 [WARNING] gpio1: Unhandled write to offset 0x4. Unhandled bits: [2] when writing value 0x5. Tags: OutputBufferEnable (0x1).
15:42:32.2036 [WARNING] gpio1: Unhandled write to offset 0x8. Unhandled bits: [2] when writing value 0x5. Tags: OutputBufferEnable (0x1).
```

Log level 2 (warning)

Log level 1 (info)

Log level 0 (Debug)

# Debugging a model using logs

PSE_Watchdog.cs

PSE_Watchdog ▸ DefineRegisters()

```
47                          this.Log(LogLevel.Debug, "Starting watchdog.");
48                          internalTimer.Enabled = true;
49                          SetState(State.ForbiddenRegion);
50                      }
51                      else if(state == State.RefreshRegion && value == WatchdogReset)
52                      {
53                          this.Log(LogLevel.Noisy, "Refreshing watchdog.");
54                          SetState(State.ForbiddenRegion);
55                      }
56                      else if(state == State.ForbiddenRegion && forbiddenRangeEnabled.Value)
57                      {
58                          this.Log(LogLevel.Warning, "Watchdog refreshed in forbidden region, triggering NMI.");
59                          SetState(State.AfterTrigger);
60                      }
61                  }, valueProviderCallback: _ => GetCurrentTimerValue(), name: "REFRESH")
62                  .WithWriteCallback((_, __) => locked.Value = true);
63              ;
```

E.G you could add the model as JIT and add more logging as needed to the file and re-launch Renode to add the increased logging

# Debugging a model using logs

```
PolarFire-SoC-Renode-emulation-platform [Program] C:\Microsemi\SoftConsole_v6.0\renode\bin\Renode.exe
15:42:20.5119 [INFO] Including script: C:\Microsemi\SoftConsole_v6.0\renode\scripts\single-node\polarfire-soc-multiple-servers.resc
15:42:22.9525 [ERROR] Script: Renode has been started successfully and is ready for a gdb connection. (This is not an error)
15:42:25.3827 [ERROR] u54_2: CPU abort [PC=0x1000]: Trying to execute code outside RAM or ROM at 0x0000000000001000.
15:42:25.3847 [ERROR] u54_1: CPU abort [PC=0x1000]: Trying to execute code outside RAM or ROM at 0x0000000000001000.
15:42:32.2031 [WARNING] gpio1: Unhandled write to offset 0x0. Unhandled bits: [2] when writing value 0x5. Tags: OutputBufferEnable (0x1).
15:42:32.2036 [WARNING] gpio1: Unhandled write to offset 0x4. Unhandled bits: [2] when writing value 0x5. Tags: OutputBufferEnable (0x1).
15:42:32.2036 [WARNING] gpio1: Unhandled write to offset 0x8. Unhandled bits: [2] when writing value 0x5. Tags: OutputBufferEnable (0x1).
```

Log level 0 (Debug) + LogPeripheralAccess

```
PolarFire-SoC-Renode-emulation-platform [Program] C:\Microsemi\SoftConsole_v6.0\renode\bin\Renode.exe
15:54:05.3206 [INFO] Including script: C:\Microsemi\SoftConsole_v6.0\renode\scripts\single-node\polarfire-soc-multiple-servers.resc
15:54:08.4482 [ERROR] Script: Renode has been started successfully and is ready for a gdb connection. (This is not an error)
15:54:10.9811 [ERROR] u54_1: CPU abort [PC=0x1000]: Trying to execute code outside RAM or ROM at 0x0000000000001000.
15:54:10.9811 [ERROR] u54_2: CPU abort [PC=0x1000]: Trying to execute code outside RAM or ROM at 0x0000000000001000.
15:54:55.7182 [DEBUG] gpio1: WriteUInt32 to 0x0 (unknown), value 0x5.
15:54:55.7336 [WARNING] gpio1: Unhandled write to offset 0x0. Unhandled bits: [2] when writing value 0x5. Tags: OutputBufferEnable (0x1).
15:54:55.7351 [DEBUG] gpio1: WriteUInt32 to 0x4 (unknown), value 0x5.
15:54:55.7356 [WARNING] gpio1: Unhandled write to offset 0x4. Unhandled bits: [2] when writing value 0x5. Tags: OutputBufferEnable (0x1).
15:54:55.7356 [DEBUG] gpio1: WriteUInt32 to 0x8 (unknown), value 0x5.
15:54:55.7356 [WARNING] gpio1: Unhandled write to offset 0x8. Unhandled bits: [2] when writing value 0x5. Tags: OutputBufferEnable (0x1).
15:54:55.7361 [DEBUG] gpio1: WriteUInt32 to 0x88 (OutputRegister), value 0x0.
15:54:55.7589 [DEBUG] gpio1: WriteUInt32 to 0xA4 (SetRegister), value 0x1.
15:54:55.7599 [DEBUG] gpio1: WriteUInt32 to 0xA4 (SetRegister), value 0x2.
15:54:55.7604 [DEBUG] gpio1: WriteUInt32 to 0xA4 (SetRegister), value 0x4.
15:54:55.7624 [DEBUG] gpio1: WriteUInt32 to 0xA0 (ClearRegister), value 0x1.
15:54:55.7624 [DEBUG] gpio1: WriteUInt32 to 0xA0 (ClearRegister), value 0x2.
15:54:55.7629 [DEBUG] gpio1: WriteUInt32 to 0xA0 (ClearRegister), value 0x4.
15:54:55.7649 [DEBUG] gpio1: WriteUInt32 to 0xA4 (SetRegister), value 0x1.
```

# Summary

- Where are models in Renode
- Ways to add models
- How to add a Just In Time (JIT) compiled model
- Debugging a model

# 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 models with Renode

Oct. 3 - Webinar 6: Add and Debug and Pre-Existing model in PolarFire SoC

**Nov. 7 - Webinar 7: How to write custom models – filters, offloading, acceleration etc**

**Dec. 5 - Webinar 8: Handling Binaries**

**Contd.**

# Second Thursdays

**Jan. 9 - Webinar 9: Run Linux on Renode (PolarFire SoC Model as a Quad-core SMP) – this is not a Linux / Buildroot tutorial**

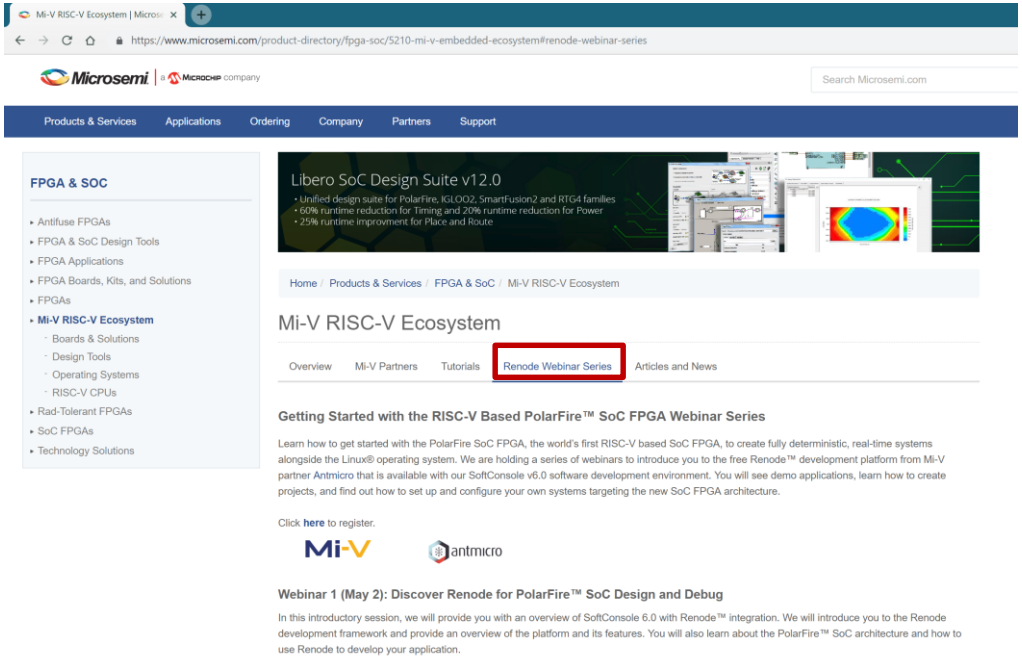
**Feb. 13 - Webinar 10: Build applications for Linux on PolarFire SoC**

**Mar. 12 - Webinar 11: Introduction to PolarFire SoC MSS Configuration and Software Flow**

**Apr. 9 - Webinar 12: Two baremetal Applications on PolarFire SoC**

**May 14 - Webinar 13: Linux + Real-Time (AMP Mode) on PolarFire SoC**

# Supporting Content




www.microsemi.com/Mi-V "Renode Webinar Series"


# Thank You