Verdi – Embedded Software Debug

Daniel Grabowski - NVIDIA
Alex Wakefield - Synopsys

September 11, 2014
SNUG Boston

Agenda

Overview

NVIDIA Evaluation

Verdi HW/SW Recorders

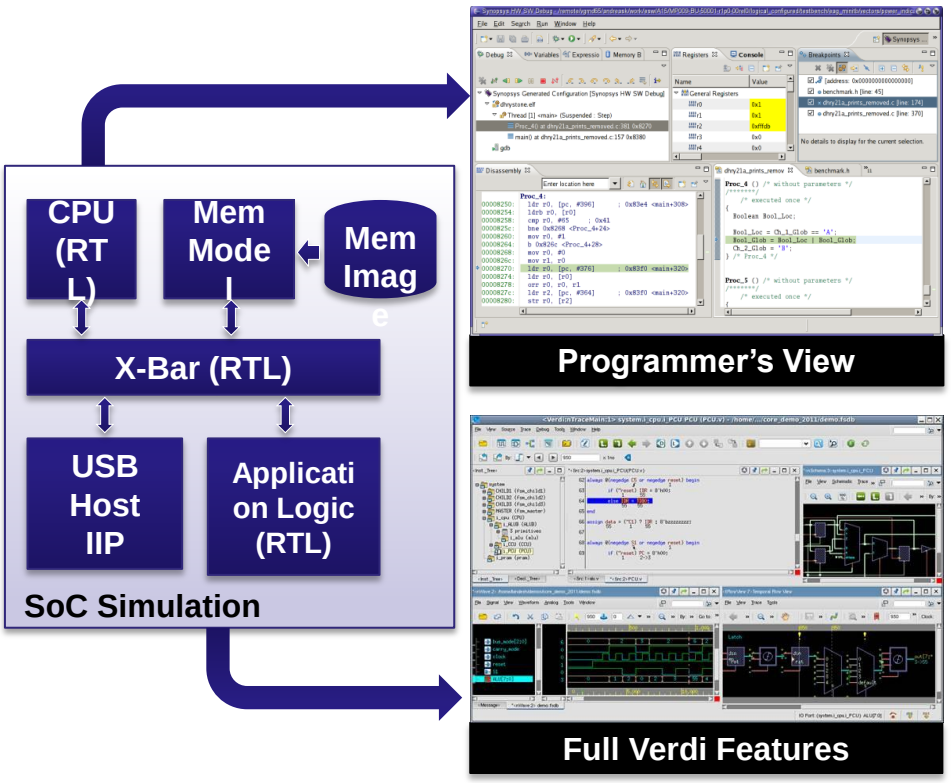
Software Debug

Conclusion

Overview

Overview

- Enables co-debug between RTL and SW
- HW/SW debug synchronized in time
- View C/Assembly source, C variables, stack, memory
- Debug multiple cores simultaneously
- Supports all ARM cores
- Easy to support custom cores



NVIDIA Evaluation

NVIDIA Evaluation

Overview

- Tegra SoC with 4 ARM cores
  - Using supplied recorders
- C/C++ based testing API
- Integrated HW/SW debug into existing flow

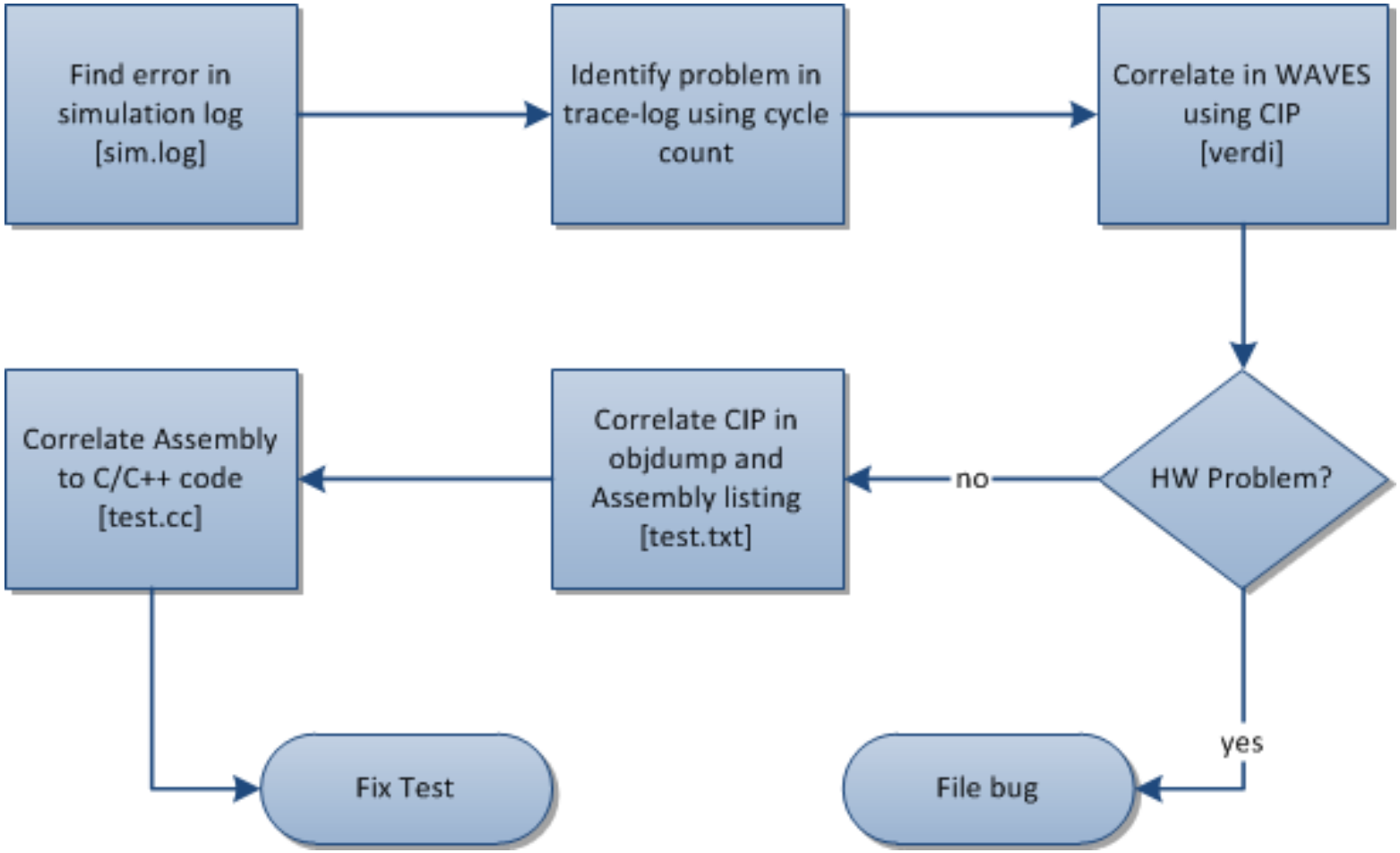
NVIDIA Evaluation

Workflow Comparison

- Existing Workflow
  - Required several files/tools to be opened simultaneously
    - Assembly/C/C++ source
    - Assembly listing of C/C++ source
    - ELF objdump
    - Simulation log
    - Trace-log file
    - Waveform viewer
  - Manual correlation through files using instruction address
  - Manual correlation between assembly instructions and C-code

NVIDIA Evaluation

Workflow Comparison



NVIDIA Evaluation

Workflow Comparison

- Verdi HW/SW Debug Workflow
  - Only requires simulation log and Verdi
    - Clever use of transaction debug could move log into Verdi
  - Identification of problem is accelerated with interactive stepping, variable watch points, and memory visibility
  - Source code, assembly, waves are synchronized within debugger
  - C/C++ correlated to Assembly using GDB

NVIDIA Evaluation

Integration

- Requires module in root “verdiHwswDebugTop”
  - Module name can be overridden in invocation dialogue
  - Contains recorders for ARM CPUs
    - Name of corresponding trace-log
    - Clock name required when trace-log uses clock ticks
    - One per CPU core
- Dump verdiHwswDebugTop into FSDB
  - Requires “+parameter”
  - Requires “+all”
  - Dump into separate FSDB
    - Allows limited scope for main waveform dump
    - Can quickly load smaller FSDB for HW/SW cosim only

NVIDIA Evaluation

Integration

- C/C++ compiled for debug
  - Add –g option to debug
    - Adds required debug symbols to ELF
  - Add –O0 to include most detailed info

NVIDIA Evaluation

Integration – verdiHwswDebugTop.sv

- Example template included with Verdi installation

```
`timescale 1ps/1ps
`include "verdiRecorder.svp"

verdiRecorder #(.cpuId(0), .tarmacFileName(`HWSW_COSIM_FILENAME_0))
                  cpu0(.cpuClock(`HWSW_COSIM_CLK_0));
   …
verdiRecorder #(.cpuId(3), .tarmacFileName(`HWSW_COSIM_FILENAME_3))
                  cpu3(.cpuClock(`HWSW_COSIM_CLK_3));

initial begin
    string filename = "hwsw_debug";
    $fsdbDumpfile(filename);
    $fsdbDumpvars(0, "verdiHwswDebugTop", "+all", "+parameter",
                     {"+fsdbfile+",filename});
end

// Common initialization -
//Leave this at close to the end of the module.

`VERDI_HWSW_INIT_TOP

endmodule: verdiHwswDebugTop
```

NVIDIA Evaluation

Simulation

- Requires running with trace-log and waves
- Requires additional plusargs
  - +verdi_hwsw_exe=<relative path to ELF binary>

```
Ex: +verdi_hwsw_exe=ordering.8-A.32.axf
```

Verdi HW/SW Recorders

Verdi HW/SW Recorders

- Verdi HW/SW Debug requires information in FSDB
  - Time in nsec or clock cycle counts
  - Program counter values
  - Register changes (writes to the register file)
  - CPU -> L1 memory read/write address, data, size
- ARM IP
  - Text-based log with required information
- Custom Cores
  - Create custom recorder
  - Record information directly from RTL or parse a custom log file

Verdi HW/SW Recorders

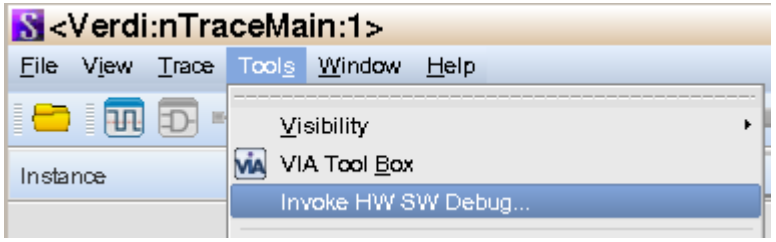
- One recorder per physical CPU core
- Clock pin
  - Required if instruction trace is recorded using cycle-count or clock-count
  - Tied to ‘h0 if trace is recorded using time stamps

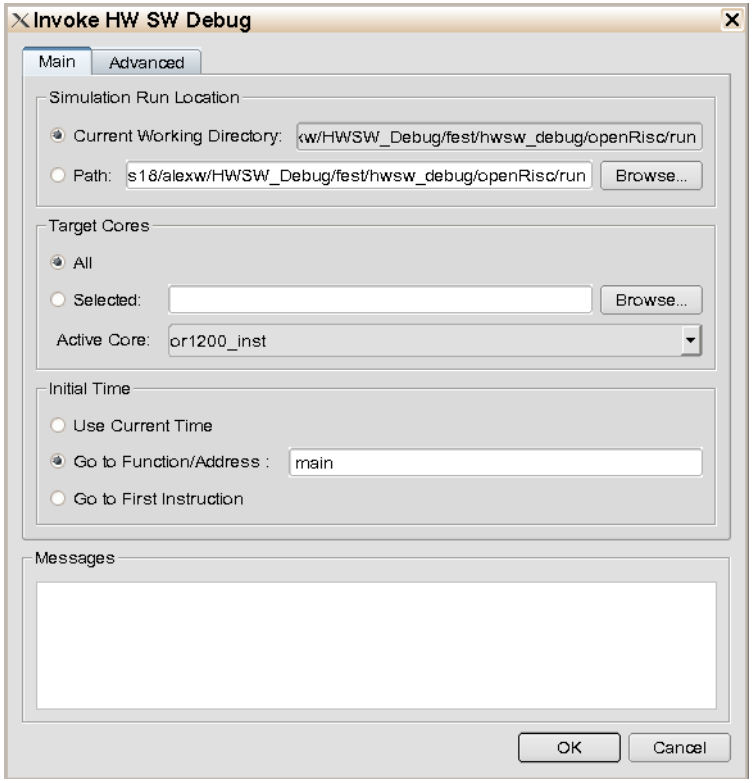
Software Debug

Software Debug

Verdi HW/SW Invocation

- Invoke Verdi with HW/SW FSDB
- Tools->Invoke HW SW Debug…
- Adjust preferences in HW/SW Debug dialogue

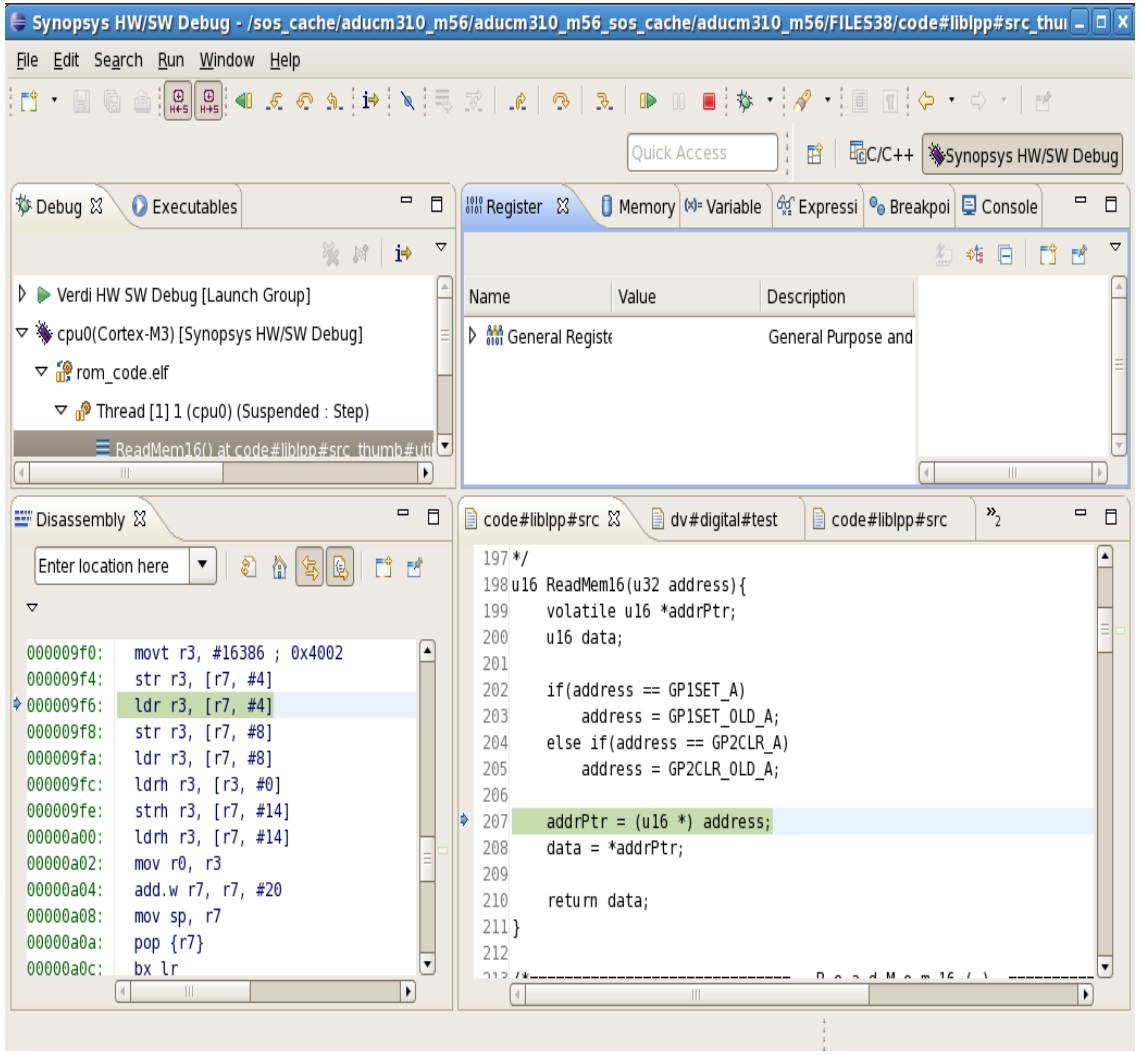




Software Debug

Debug Features

- Time Synchronization
  - Verdi and Eclipse synchronized in time
- Eclipse IDE
  - C/C++ source code view
  - Disassembly view
  - Stack view
  - Register view
  - Variables
  - Memory
  - Expressions
  - Breakpoints



Conclusion

Conclusion

- Easy to integrate
  - ARM recorders provided
  - Eclipse provided
  - Minimal additional RTL
- Familiar UI
  - Verdi
  - Eclipse
- Streamlines workflow
  - From multiple files and context switches to single application


Thank You

Backup

NVIDIA Evaluation

Integration – verdiHwswDebugTop.sv

- Set timescale
  - Issues may occur if timescale is not sufficient to capture clock

```
`timescale 1ps/1ps
```

- Include recorder

```
`include "verdiRecorder.svp"
```

- Module must be "verdiHwswDebugTop"

```
module verdiHwswDebugTop;
```

NVIDIA Evaluation

Integation – verdiHwswDebugTop.sv

- Instance Recorders

```
verdiRecorder #(.cpuId(0), .tarmacFileName(`HWSW_COSIM_FILENAME_0))
              cpu0(.cpuClock(`HWSW_COSIM_CLK_0));
   …
verdiRecorder #(.cpuId(3), .tarmacFileName(`HWSW_COSIM_FILENAME_3))
              cpu3(.cpuClock(`HWSW_COSIM_CLK_3));
```

- Setup FSDB dump

```
 initial begin
       string filename = "hwsw_debug";
        $fsdbDumpfile(filename);
       $fsdbDumpvars(0, "verdiHwswDebugTop", "+all", "+parameter",
{"+fsdbfile+",filename});
    end
```

NVIDIA Evaluation

Integration – verdiHwswDebugTop.sv

- Common initialization

```
// Common initialization -
//Leave this at close to the end of the module.
    `VERDI_HWSW_INIT_TOP

endmodule: verdiHwswDebugTop
```