ERICSSON

TLV IN LUND

Introduction to Top Level Verification

PRESENTER

- Shkelqim Lahi
  - B.Sc.E. in Computer Engineering (Engineering College of Copenhagen 2004)
  - M. SC. EE. In “System on Chip” (LTH 2006)
  - Work experience
    - EMP 2006 – 2009
    - ST-Ericsson 2009 – 2013
    - Ericsson 2013 - ….
  - Verification Engineer

ERICSSON IN LUND

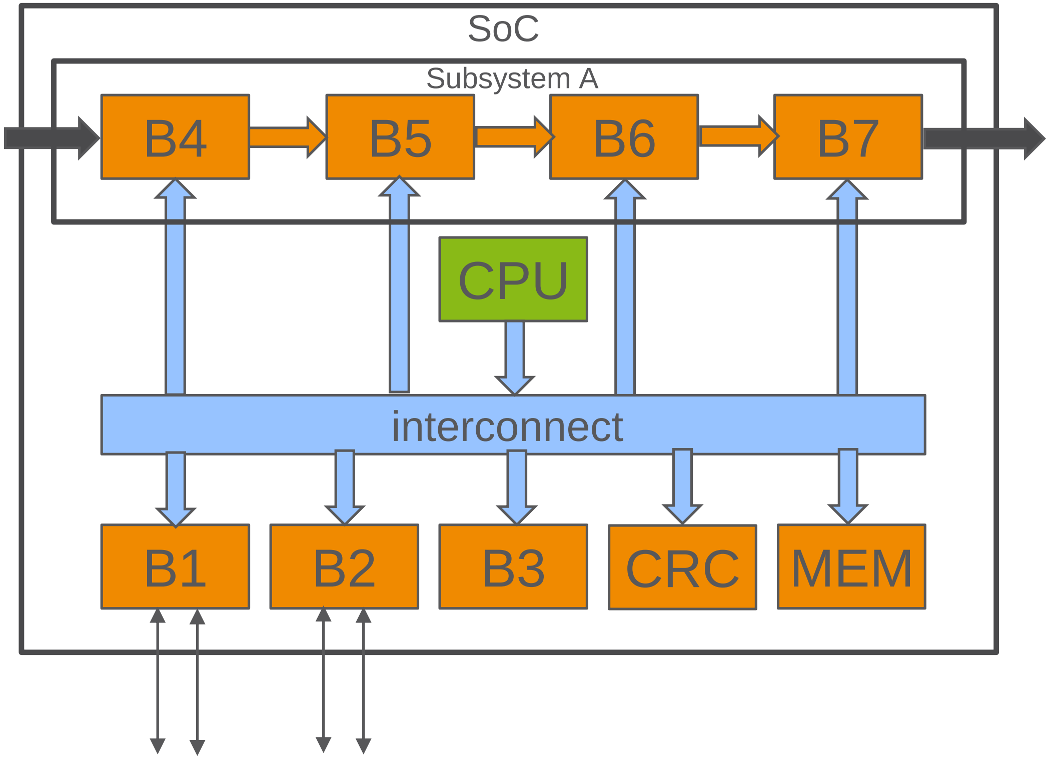
- Radio Product and variant Lund
  - Radio, analog and mix signals ASICs for 5G
  - FPGA Products
  - Design & Verification in all levels
- SOC Level I&V
  - Top Level Verification for Radio ASICs

AGENDA

Top Level Verification in focus

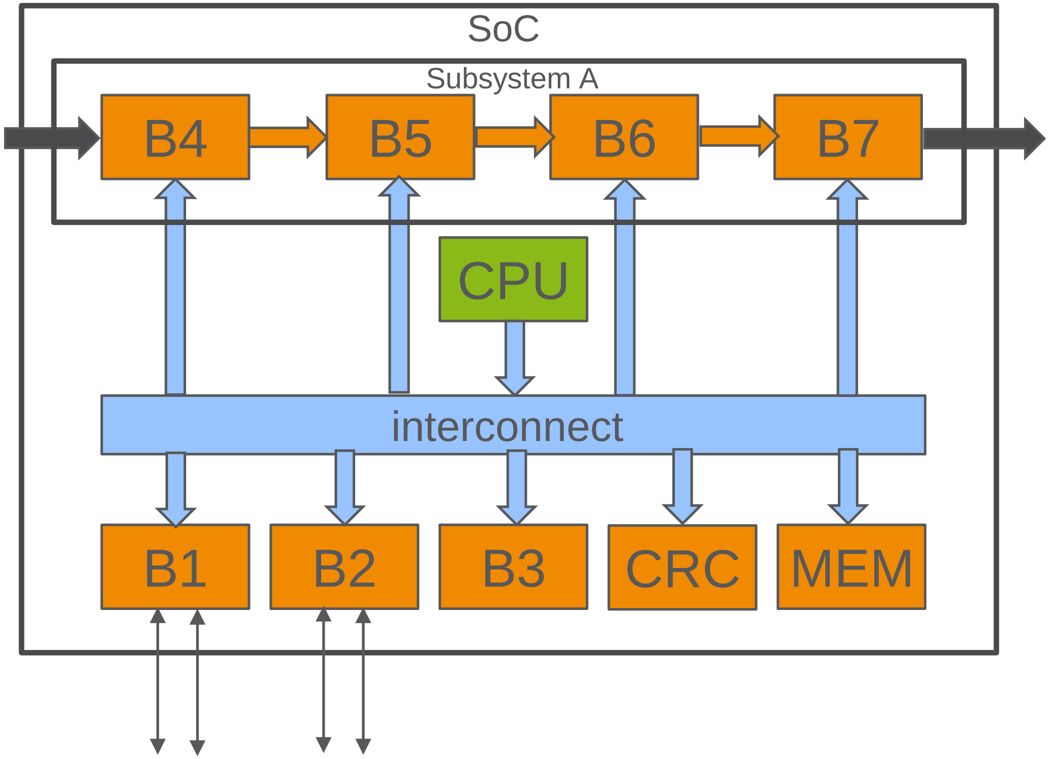
- **SoC Verification**
- What do we do?
- TLV SW environment
- Why Verification?
- TLV in other areas
- Q&A

SOC



VERIFICATION SCOPES

- Block Verification
  - Block functionality
  - Protocol verification
  - Full coverage e.g. code coverage
- Subsystem Verification
  - Integration of blocks
  - Functionality of combined blocks
- Top Level Verification
  - Integration test
  - Functional test of full system
  - Pad verification



TLV MISSION

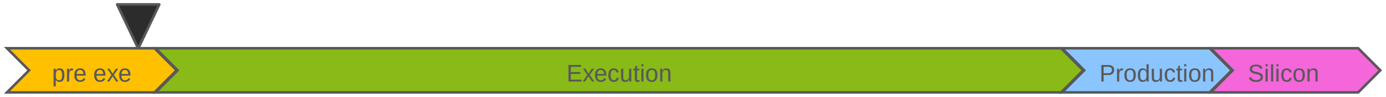
- Full responsability for the functionality of the ASIC/SoC
- Design implemented according to the Specification
- Involved in ASIC Bring up activities

AGENDA

- SoC Verification
- **What do we do?**
- TLV SW environment
- Why Verification?
- TLV in other areas
- Q&A

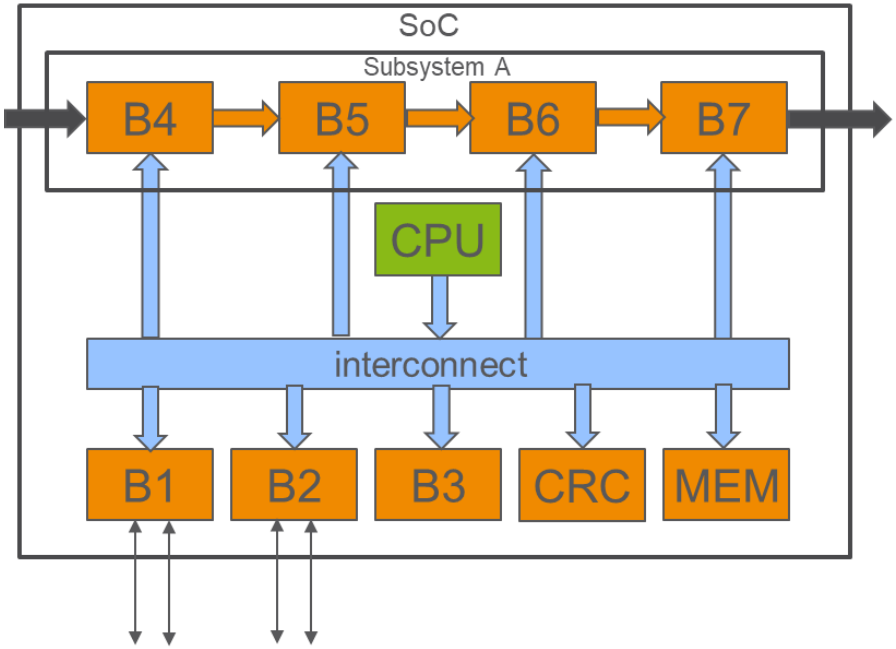
WHAT DO WE DO?

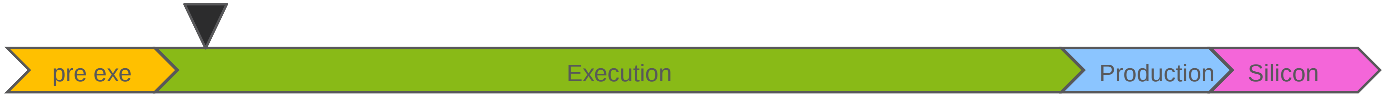
- Project start
  - Workshop and information meetings
  - Verification Planning
  - Prepare SW and TB environment for the Project
  - CPU architecture and compiler
    - ARM CA53, CA9, CA7, CA5
    - ARM CM4, CM7
    - ARM CR4



WHAT DO WE DO?

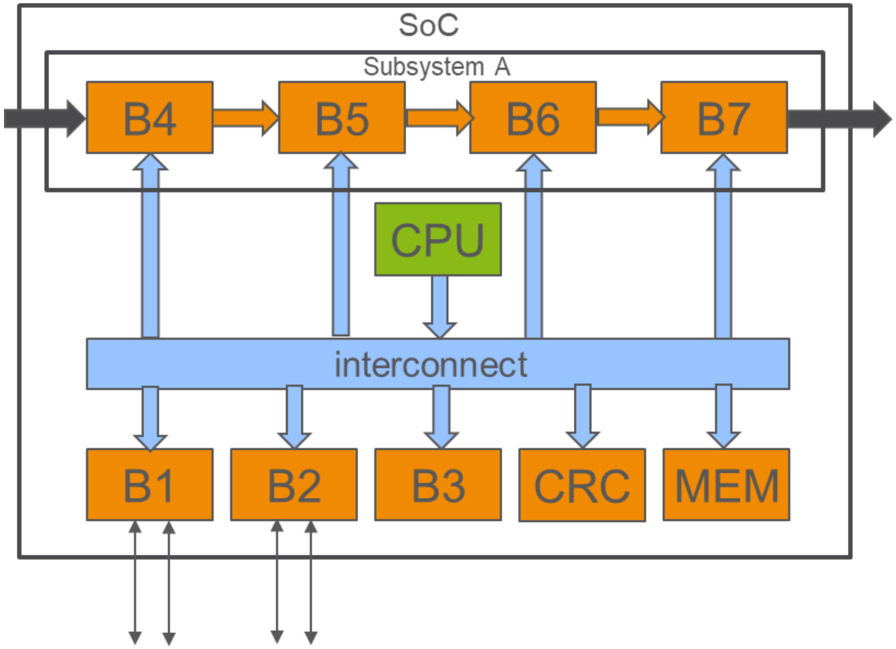
- Sanity test
  - Boot all CPUs and Cores
    - Clk & rst, Memories, Interconnect
  - Interconnect test
    - Access every block inte the system, one register on every block
  - Memory test
    - Access memories, few locations

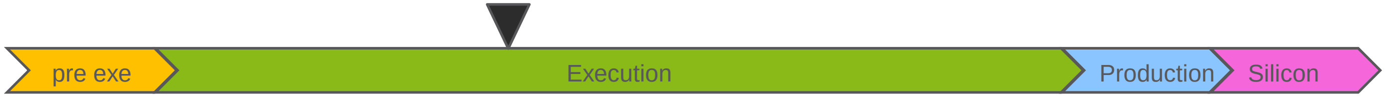




WHAT DO WE DO?

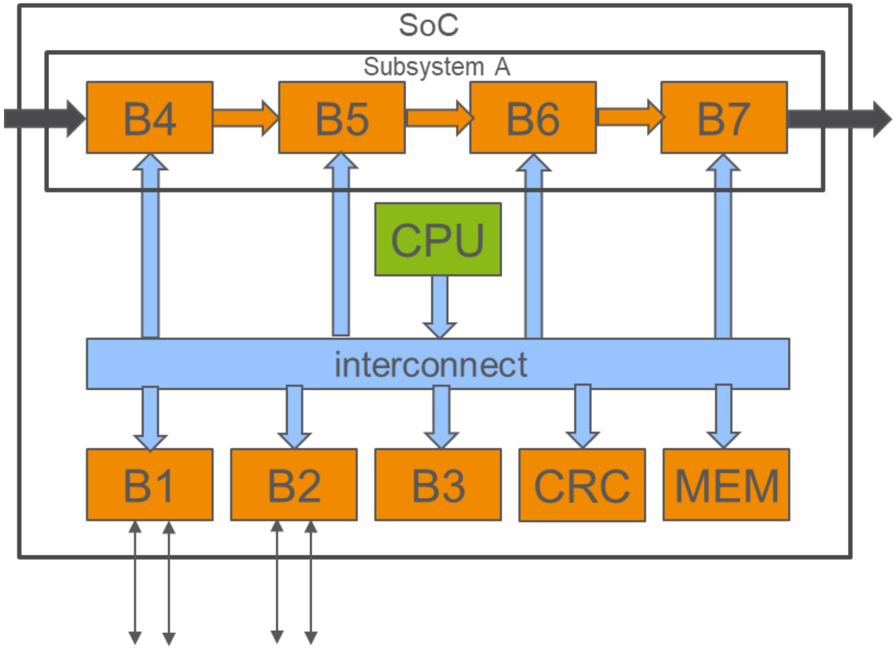
- Integration test
  - Integration test of Blocks with external interfaces (B1, B2)
    - E.g. UART, I2C, DDR
  - Integration test of internal blocks (B3)
    - E.g Timer, Interrupt controller
  - Integration test of Subsystems (SubSys A)
    - Verify all interfaces of the subsystem

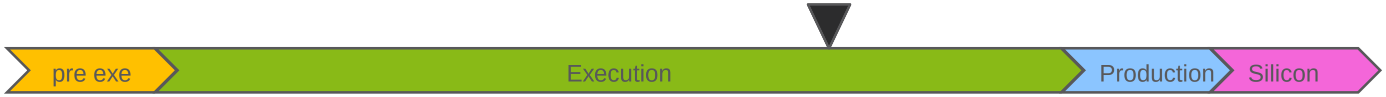




WHAT DO WE DO?

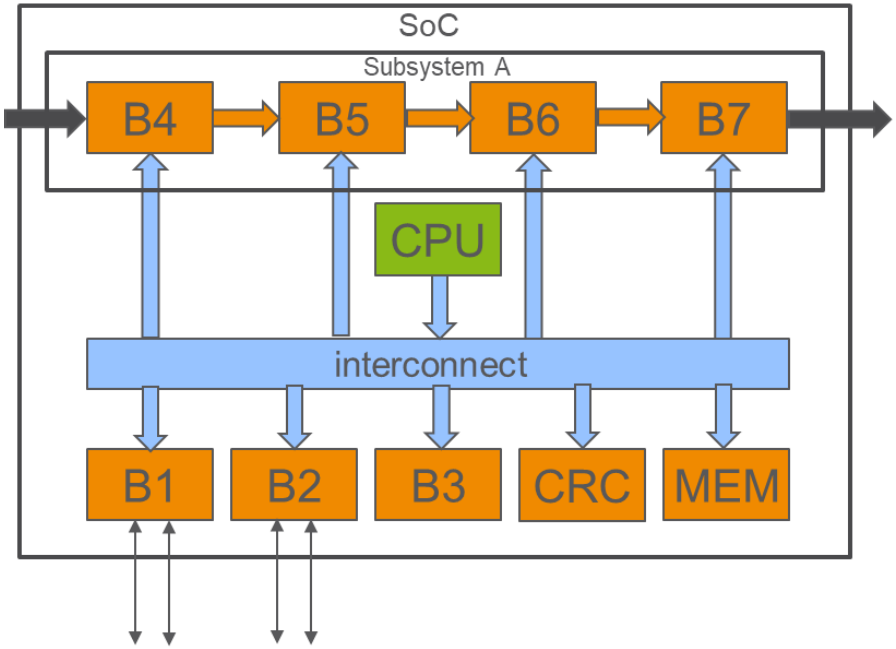
- Functional test
  - Functional test of complex blocks
    - Complement to Integration test
  - Functional use cases including several blocks
    - Different complexity levels
  - All supported boot modes
  - Cold and warm reset logic
  - Clock tree verification

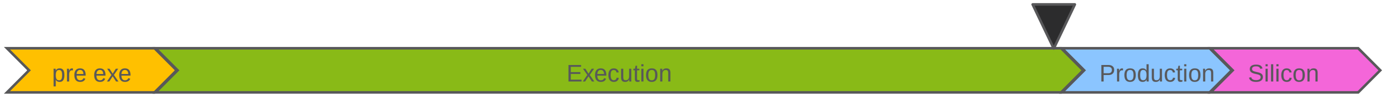




WHAT DO WE DO?

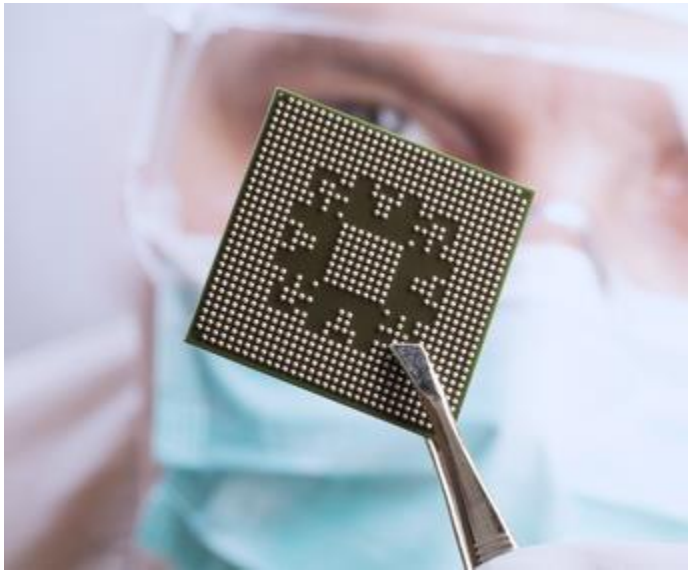
- Netlist Simulations
  - Run simple tests on the synthesized netlist
  - Without and with SDF (Timing information)
- Regression Run
  - Rerun all the tests in Regression mode
  - Quality stamp before ”tape out”

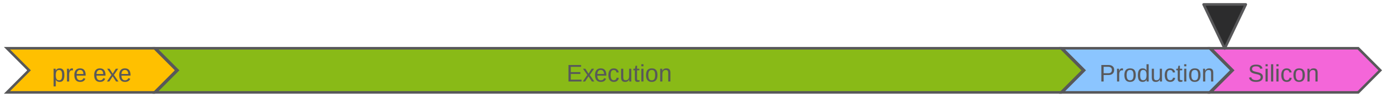




WHAT DO WE DO?

- ASIC Bring up
  - Run tests on the Silicon after production
  - Verify package
  - Sanity test of all external interfaces
- Support SW community in getting up to speed
  - Help debug HW related issues





pre exe

Execution

Production

Silicon

AGENDA

- › SoC Verification
- › What do we do?
- › **TLV SW environment**
- › Why Verification?
- › TLV in other areas
- › Q&A

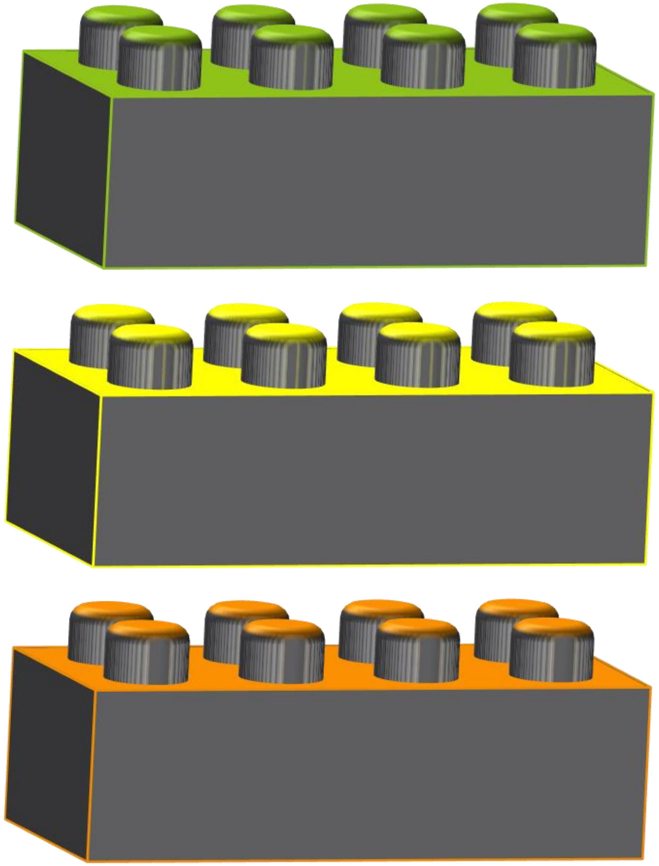
SW DRIVEN VERIFICATION

- The SW driven flow uses the existing cores in the DUT to verify the integration and/or functionality of the design
- Using SW the system is verified as it is intended to be used
- Since the test cases are implemented in SW they can easily be reused on different platforms
  - Virtual Platforms
  - Emulation
  - RTL Simulation
  - ASIC Bring-Up
- Directed tests are primarily used in the SW driven flow
  - Exhaustive testing is more suitable for constrained random flows
  - But there are also tools available on the market that can auto generate parts of the SW => enable exhaustive testing using SW

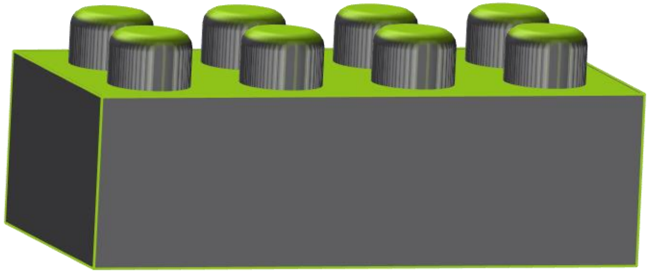
TEST ARCHITECTURE

Flexible **A**rchitecture **S**oftware **T**est bench

- TEST SW
  - Test SW including drivers and register mapping, in C.
  - ELF and HEX files are generated to be loaded into the memories.
- TC specific TB files
  - Additional DUT connections
  - Verification components
  - Assertions
  - Parameters and forces
- Common TB files
  - DUT
  - Standard TB components
  - Optional external memories

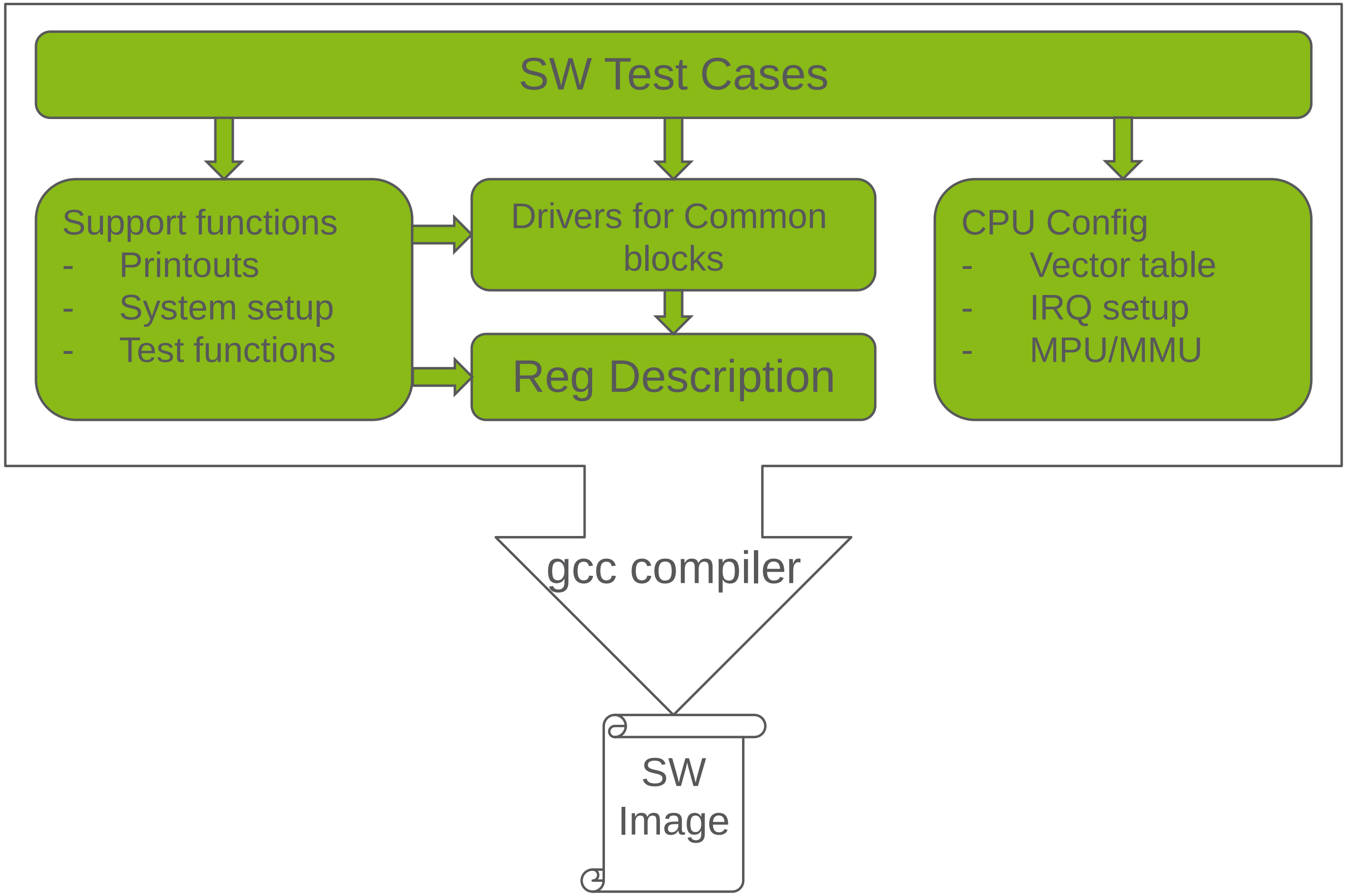


TEST SW



SW ARCHITECTURE

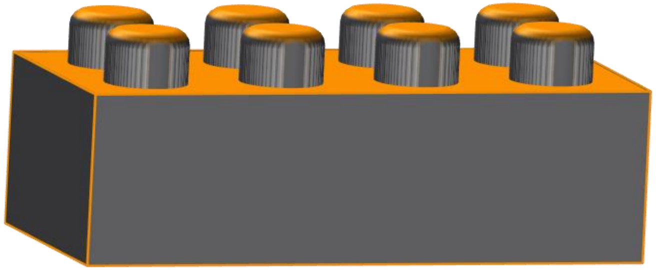
› C base SW



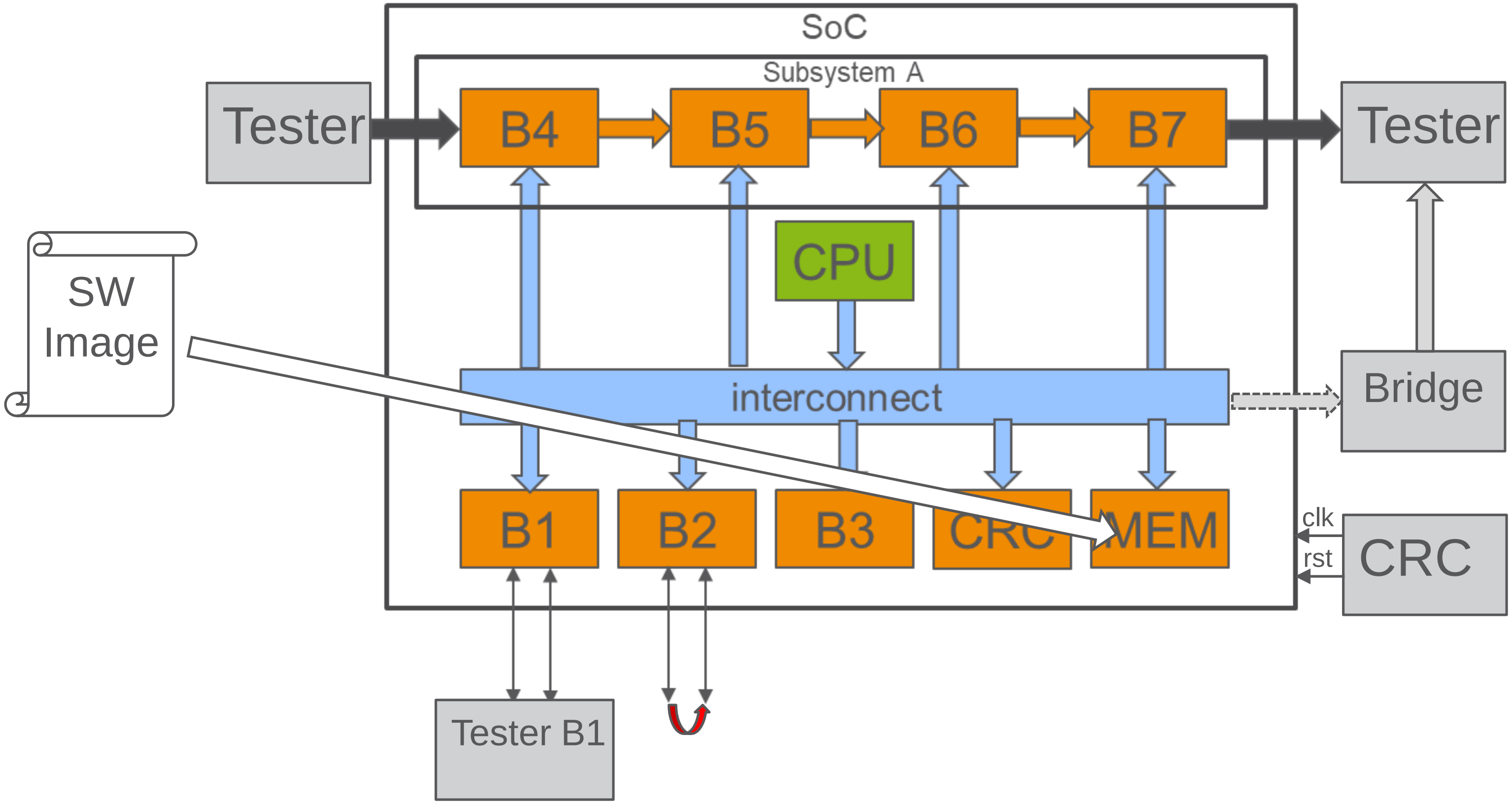
TEST CASES

- TC categories
  - Sanity – simple tests for sanity purpose (e.g. boot)
  - Register – complete register test of a block
  - Integration – Integration test of a block
  - Function – complex functional tests of the one or more blocks or subsystem
  - ABU – Sw test for ASIC bring up activities

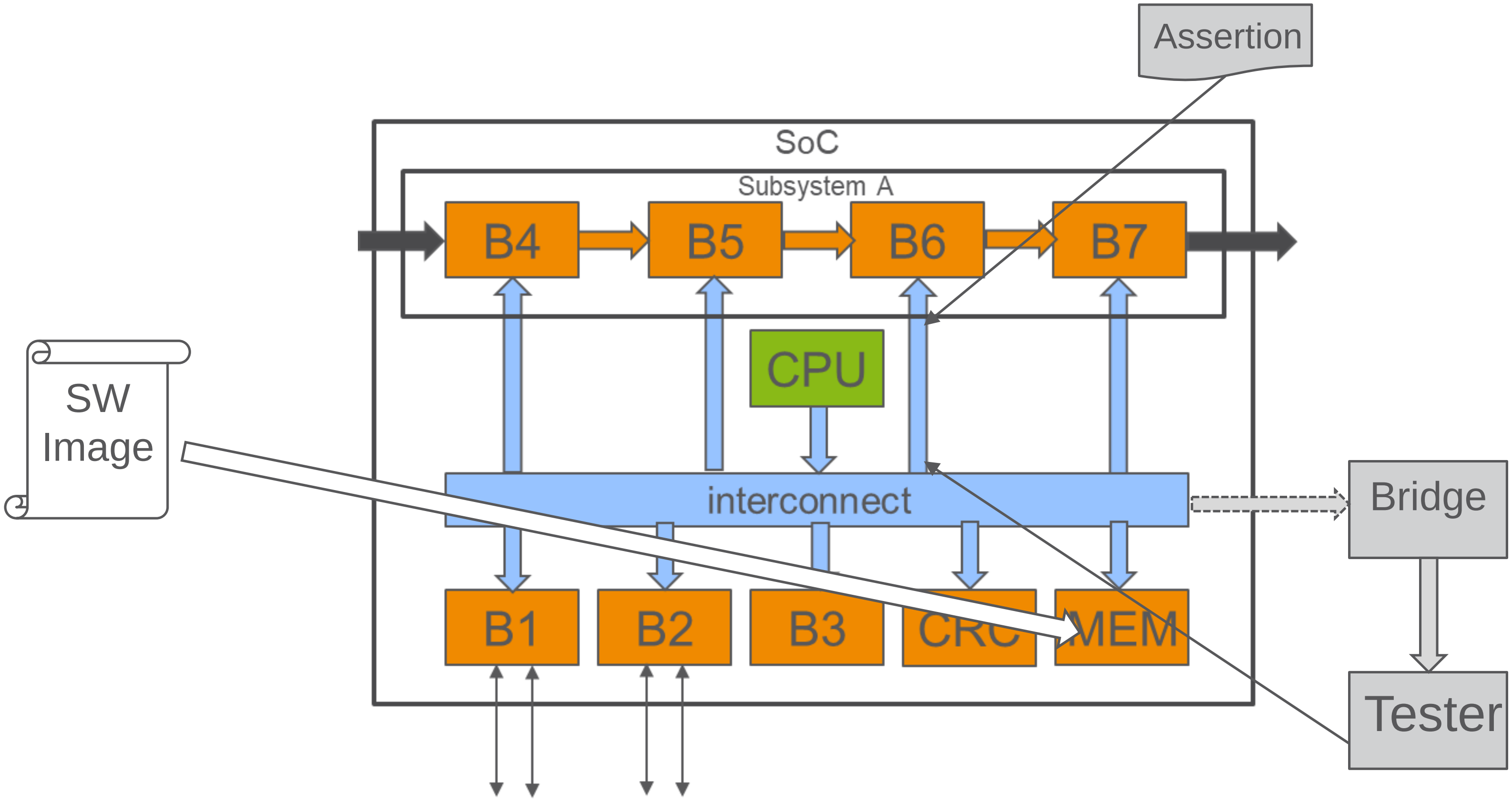
SW TESTBENCH



SW DRIVEN TB


SoC
Subsystem A
Tester
B4
B5
B6
B7
Tester
SW
Image
CPU
interconnect
Bridge
B1
B2
B3
CRC
MEM
clk
rst
CRC
Tester B1

INTEGRATION TEST WITH ASSERTION



DEBUG SW

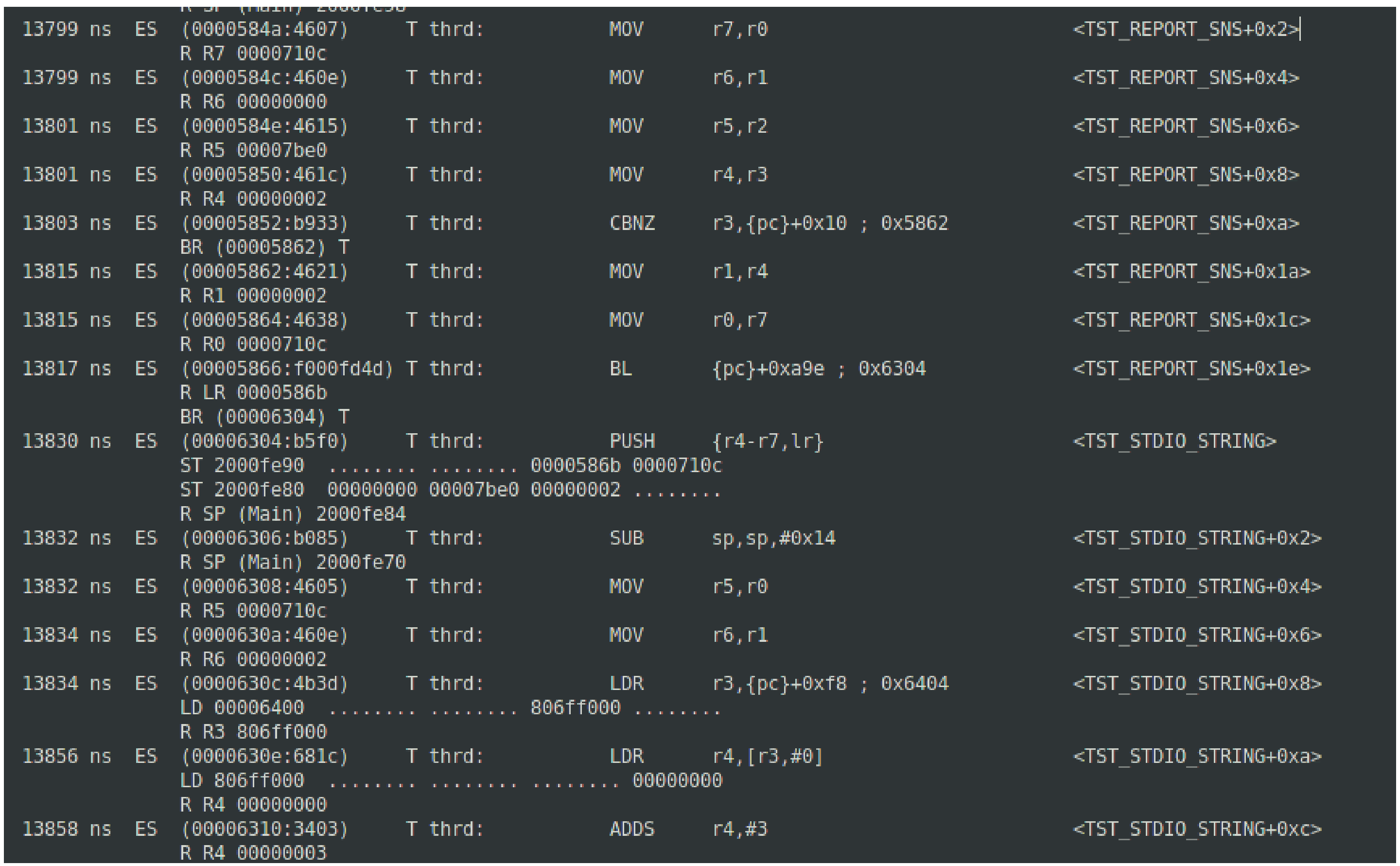
- How to debug the SW when running in simulation environment?
  - Printout!
    - Printf vs. Costumized functions

      ```
      printf(“read data %d from memory\n”, data);
      TST_REPORT_SNS(“read data “, data, “from memory\n”);
      ```

    - UART vs Costumized Tester
      - Uart is slow in simulation
      - Use costumized test component with dedicated registers for printouts
  - ARM Tarmac log
    - “Tarmac is a textual trace output. Fast Models supports the generation of traces that consistently track the execution and related activities in the model, particularly those that affect the state of the modeled IP.”

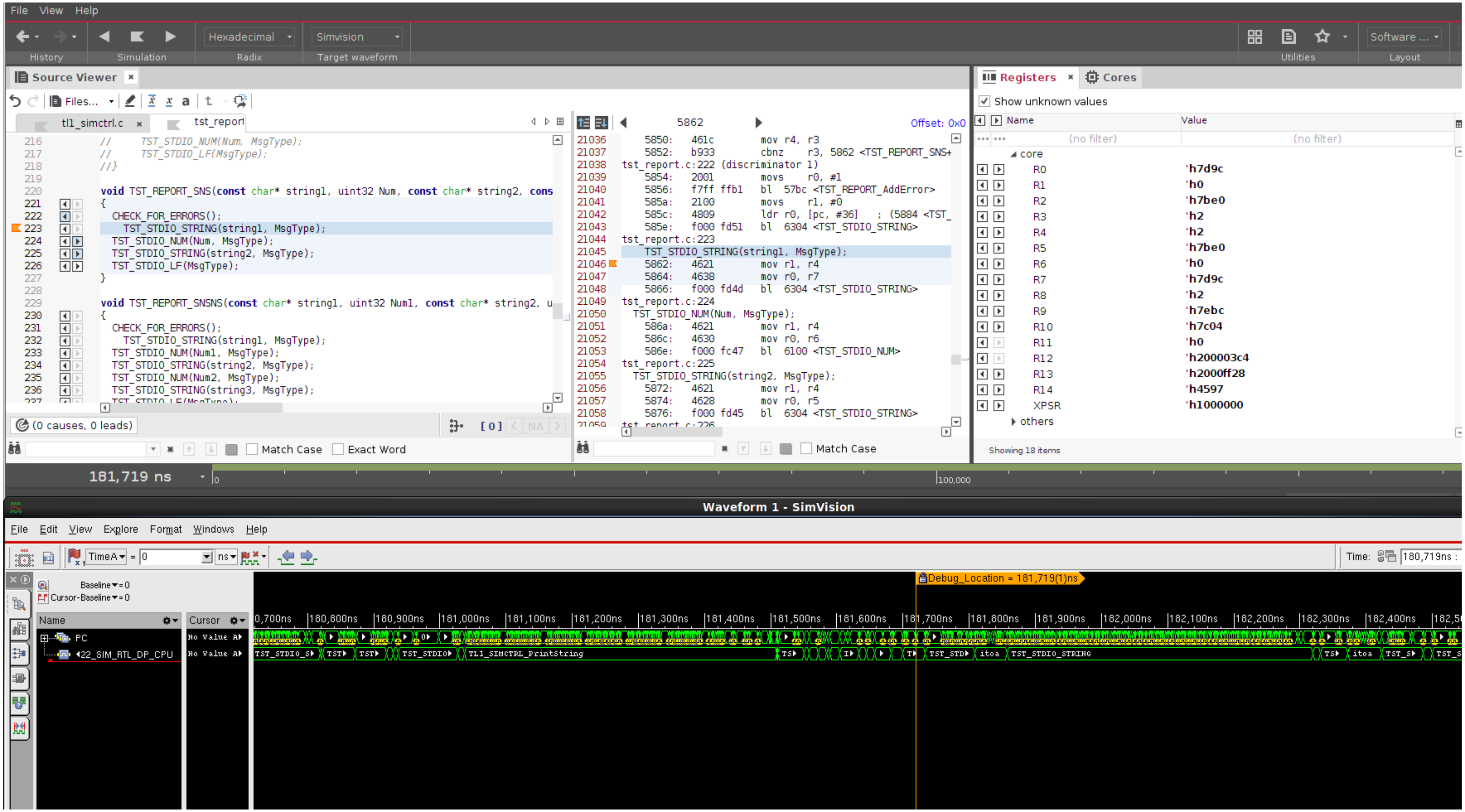
TARMAC LOG

› Tarmac log example



```
                   R SP (Main) 2000fe98
13799 ns  ES  (0000584a:4607)     T thrd:          MOV      r7,r0                          <TST_REPORT_SNS+0x2>
              R R7 0000710c
13799 ns  ES  (0000584c:460e)     T thrd:          MOV      r6,r1                          <TST_REPORT_SNS+0x4>
              R R6 00000000
13801 ns  ES  (0000584e:4615)     T thrd:          MOV      r5,r2                          <TST_REPORT_SNS+0x6>
              R R5 00007be0
13801 ns  ES  (00005850:461c)     T thrd:          MOV      r4,r3                          <TST_REPORT_SNS+0x8>
              R R4 00000002
13803 ns  ES  (00005852:b933)     T thrd:          CBNZ     r3,{pc}+0x10 ; 0x5862          <TST_REPORT_SNS+0xa>
              BR (00005862) T
13815 ns  ES  (00005862:4621)     T thrd:          MOV      r1,r4                          <TST_REPORT_SNS+0x1a>
              R R1 00000002
13815 ns  ES  (00005864:4638)     T thrd:          MOV      r0,r7                          <TST_REPORT_SNS+0x1c>
              R R0 0000710c
13817 ns  ES  (00005866:f000fd4d) T thrd:          BL       {pc}+0xa9e ; 0x6304            <TST_REPORT_SNS+0x1e>
              R LR 0000586b
              BR (00006304) T
13830 ns  ES  (00006304:b5f0)     T thrd:          PUSH     {r4-r7,lr}                     <TST_STDIO_STRING>
              ST 2000fe90  ........ ........ 0000586b 0000710c
              ST 2000fe80  00000000 00007be0 00000002 ........
              R SP (Main) 2000fe84
13832 ns  ES  (00006306:b085)     T thrd:          SUB      sp,sp,#0x14                    <TST_STDIO_STRING+0x2>
              R SP (Main) 2000fe70
13832 ns  ES  (00006308:4605)     T thrd:          MOV      r5,r0                          <TST_STDIO_STRING+0x4>
              R R5 0000710c
13834 ns  ES  (0000630a:460e)     T thrd:          MOV      r6,r1                          <TST_STDIO_STRING+0x6>
              R R6 00000002
13834 ns  ES  (0000630c:4b3d)     T thrd:          LDR      r3,{pc}+0xf8 ; 0x6404          <TST_STDIO_STRING+0x8>
              LD 00006400  ........ ........ 806ff000 ........
              R R3 806ff000
13856 ns  ES  (0000630e:681c)     T thrd:          LDR      r4,[r3,#0]                     <TST_STDIO_STRING+0xa>
              LD 806ff000  ........ ........ ........ 00000000
              R R4 00000000
13858 ns  ES  (00006310:3403)     T thrd:          ADDS     r4,#3                          <TST_STDIO_STRING+0xc>
              R R4 00000003
```

SW DEBUGGER



```
//     TST_STDIO_NUM(Num, MsgType);
//     TST_STDIO_LF(MsgType);
//}

void TST_REPORT_SNS(const char* string1, uint32 Num, const char* string2, cons
{
  CHECK_FOR_ERRORS();
    TST_STDIO_STRING(string1, MsgType);
  TST_STDIO_NUM(Num, MsgType);
  TST_STDIO_STRING(string2, MsgType);
  TST_STDIO_LF(MsgType);
}

void TST_REPORT_SNSNS(const char* string1, uint32 Num1, const char* string2, u
{
  CHECK_FOR_ERRORS();
    TST_STDIO_STRING(string1, MsgType);
  TST_STDIO_NUM(Num1, MsgType);
  TST_STDIO_STRING(string2, MsgType);
  TST_STDIO_NUM(Num2, MsgType);
  TST_STDIO_STRING(string3, MsgType);
```

```
    5850:   461c        mov r4, r3
    5852:   b933        cbnz    r3, 5862 <TST_REPORT_SNS+
tst_report.c:222 (discriminator 1)
    5854:   2001        movs    r0, #1
    5856:   f7ff ffb1   bl  57bc <TST_REPORT_AddError>
    585a:   2100        movs    r1, #0
    585c:   4809        ldr r0, [pc, #36]   ; (5884 <TST_
    585e:   f000 fd51   bl  6304 <TST_STDIO_STRING>
tst_report.c:223
    TST_STDIO_STRING(string1, MsgType);
    5862:   4621        mov r1, r4
    5864:   4638        mov r0, r7
    5866:   f000 fd4d   bl  6304 <TST_STDIO_STRING>
tst_report.c:224
  TST_STDIO_NUM(Num, MsgType);
    586a:   4621        mov r1, r4
    586c:   4630        mov r0, r6
    586e:   f000 fc47   bl  6100 <TST_STDIO_NUM>
tst_report.c:225
  TST_STDIO_STRING(string2, MsgType);
    5872:   4621        mov r1, r4
    5874:   4628        mov r0, r5
    5876:   f000 fd45   bl  6304 <TST_STDIO_STRING>
```

☑ Show unknown values

Name	Value
(no filter)	(no filter)
core	
R0	'h7d9c
R1	'h0
R2	'h7be0
R3	'h2
R4	'h2
R5	'h7be0
R6	'h0
R7	'h7d9c
R8	'h2
R9	'h7ebc
R10	'h7c04
R11	'h0
R12	'h200003c4
R13	'h2000ff28
R14	'h4597
XPSR	'h1000000
others	

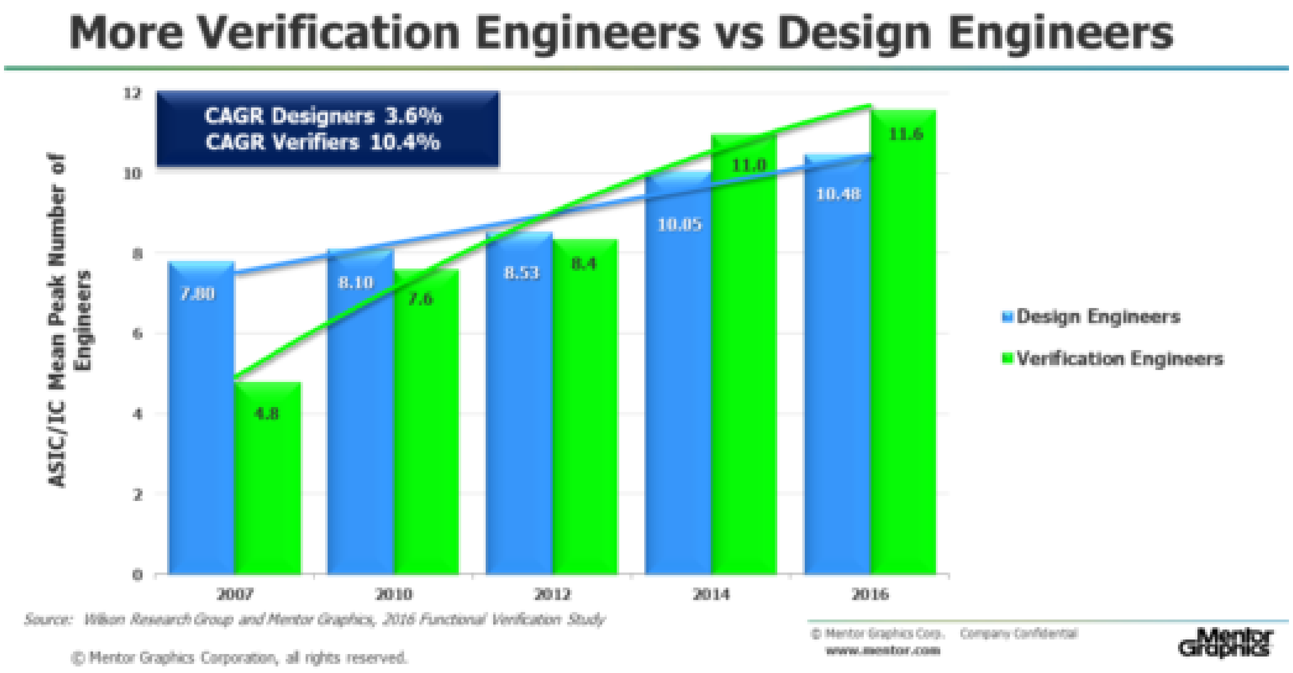
Showing 18 items

AGENDA

- SoC Verification
- What do we do?
- TLV SW environment
- **Why Verification?**
- TLV in other areas
- Q&A

INCREASING DEMAND

- The 2016 Wilson Research Group Functional Verification Study



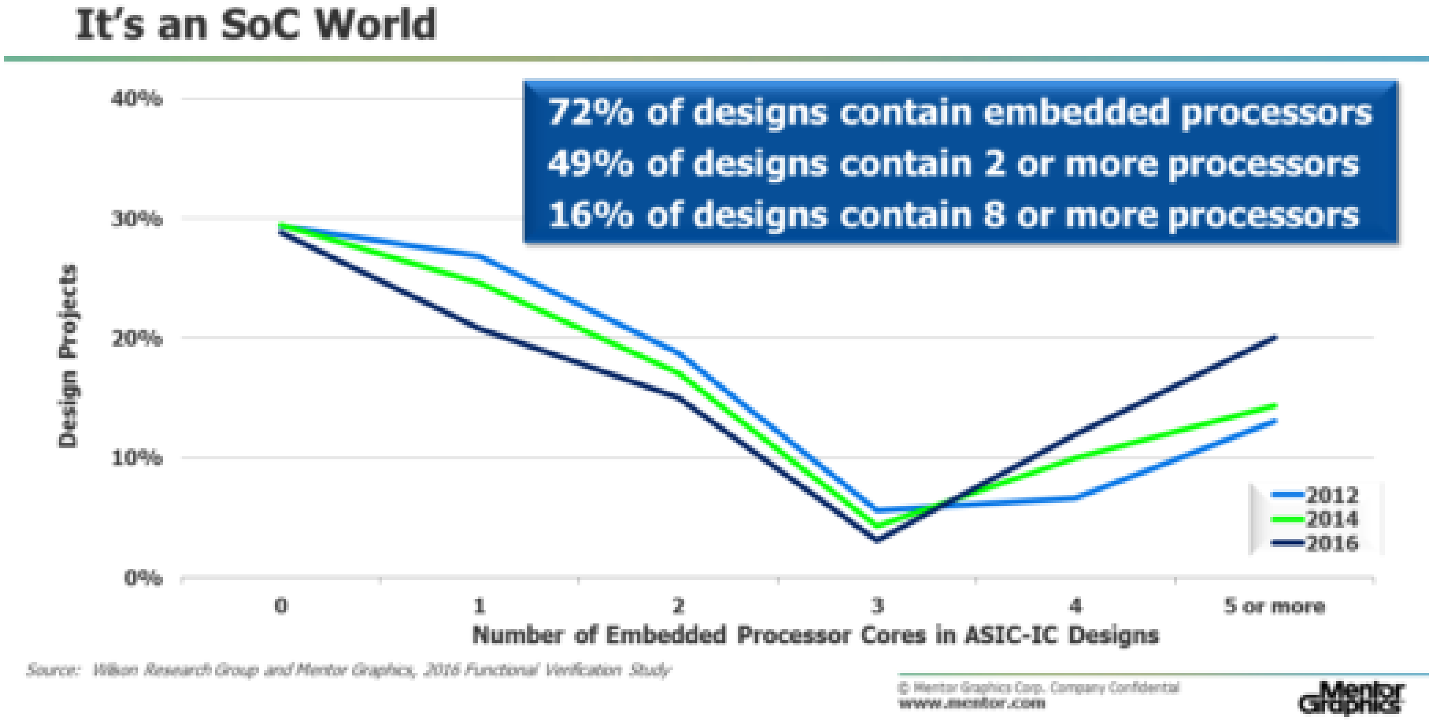
More Verification Engineers vs Design Engineers

Source: Wilson Research Group and Mentor Graphics, 2016 Functional Verification Study

© Mentor Graphics Corporation, all rights reserved.

Source: https://blogs.mentor.com/verificationhorizons/blog/2016/10/04/part-8-the-2016-wilson-research-group-functional-verification-study/

COMPLEX SOC



It's an SoC World

72% of designs contain embedded processors

49% of designs contain 2 or more processors

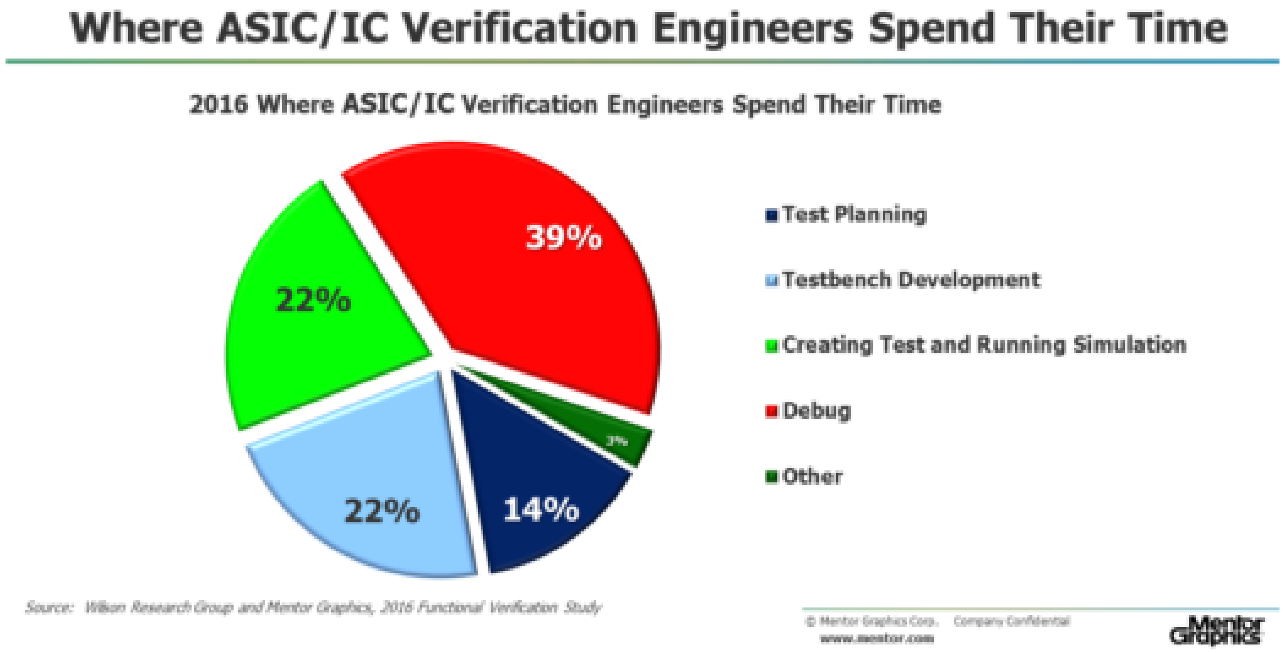
16% of designs contain 8 or more processors

Source: Wilson Research Group and Mentor Graphics, 2016 Functional Verification Study

© Mentor Graphics Corp. Company Confidential
www.mentor.com

Source: https://blogs.mentor.com/verificationhorizons/blog/2016/09/25/part-7-the-2016-wilson-research-group-functional-verification-stud/

TIMEPLAN



Where ASIC/IC Verification Engineers Spend Their Time

2016 Where ASIC/IC Verification Engineers Spend Their Time

Source: Wilson Research Group and Mentor Graphics, 2016 Functional Verification Study

© Mentor Graphics Corp. Company Confidential
www.mentor.com

Mentor Graphics

Source: https://blogs.mentor.com/verificationhorizons/blog/2016/10/04/part-8-the-2016-wilson-research-group-functional-verification-study/

WORKING IN TLV

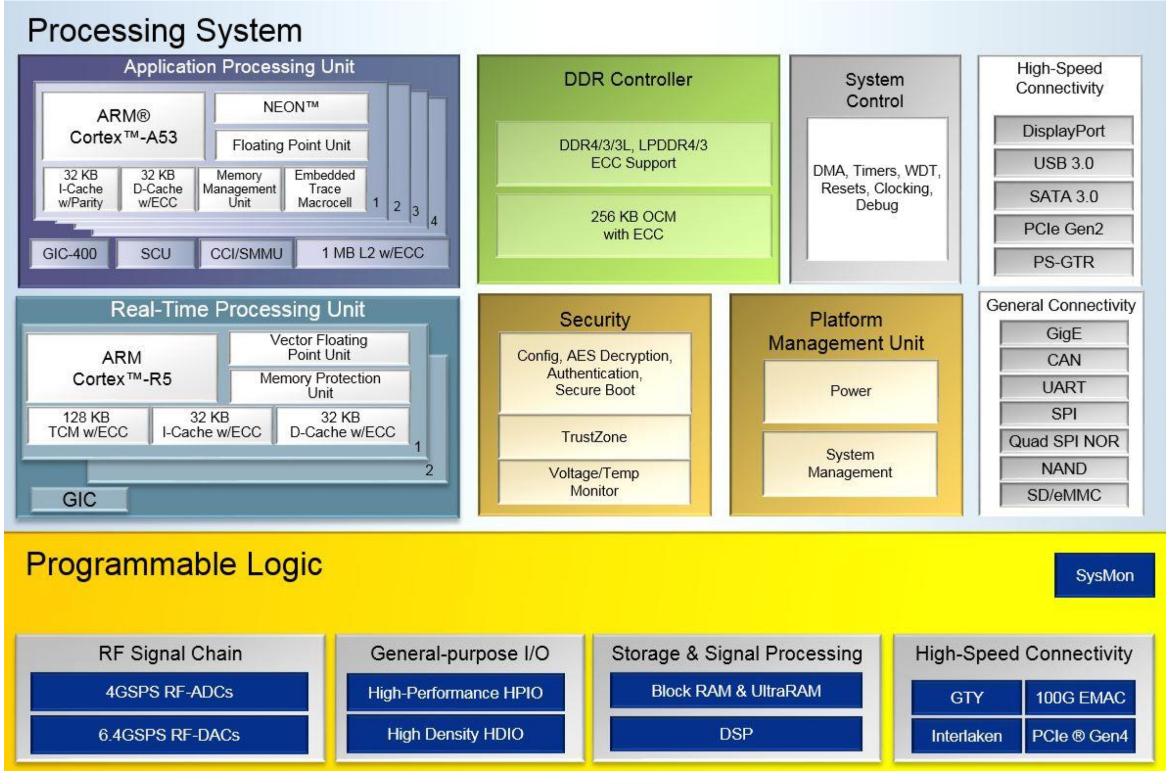
- Interesting
  - Different blocks and tools
- Challenging
  - Complex functionality
- Big responsibly
  - Responsible for the functionality of a ”very” expensive product
- Variation
  - Long project with various work packages
  - Work with both SW & HW

AGENDA

- SoC Verification
- What do we do?
- TLV SW environment
- Why Verification?
- **TLV in other areas**
- Q&A

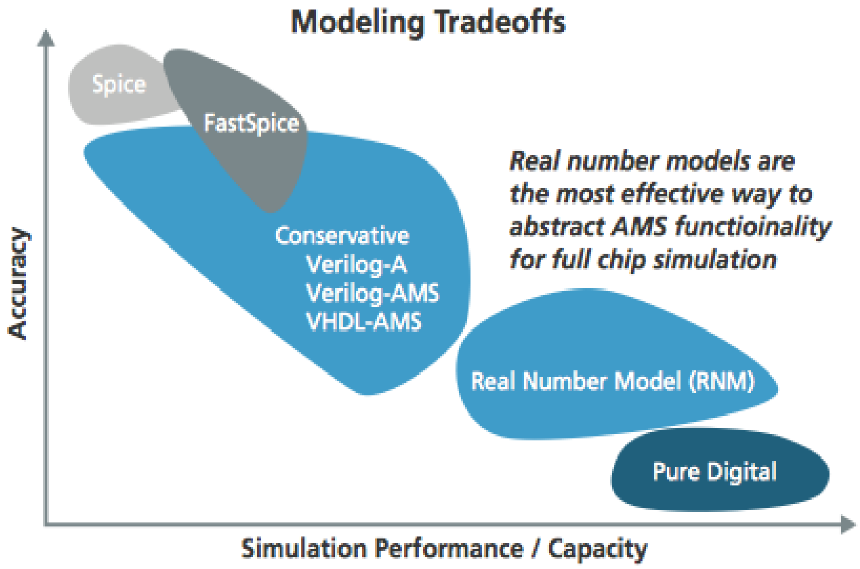
SW DRIVEN ON FPGA

- SoC on FPGAs
  - Zynq UltraScale from Xilinx
- Reuse between Simulation and Lab environment



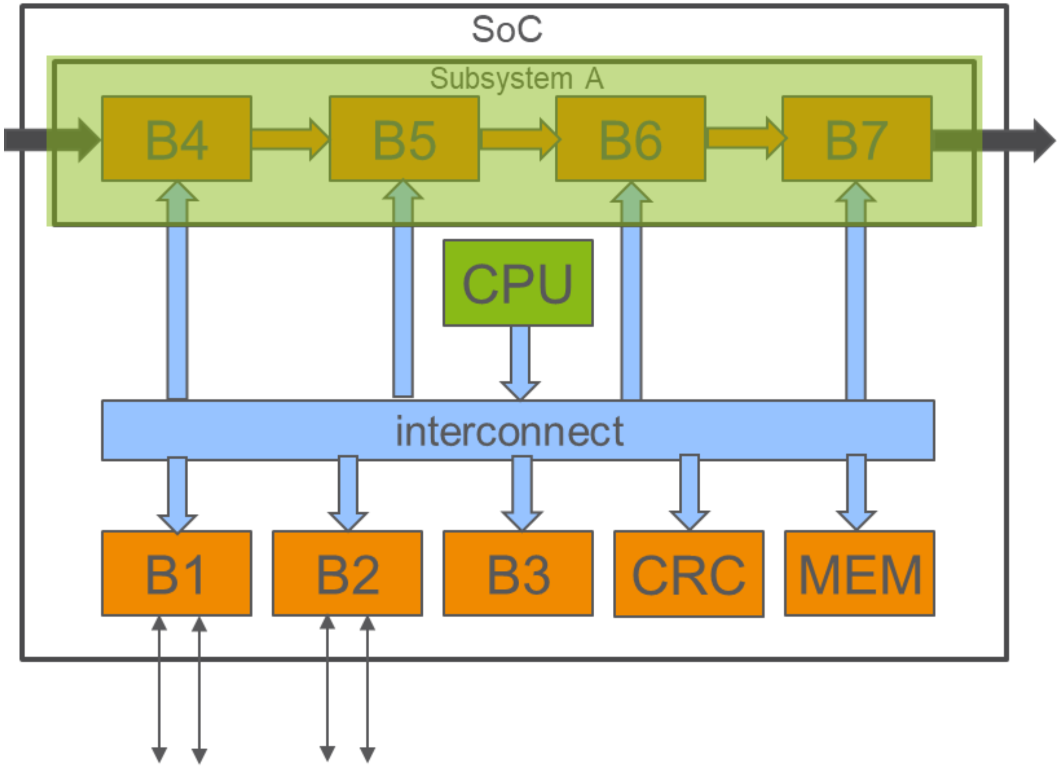
MIX SIGNAL ASIC

- Combined analog and digital design
- Perfect solution
  - SPICE + RTL simulations
  - Very slow or not possible due to complexity
- Use model for analog design
  - Real Number Modeling RNM
  - SystemVerilog SV-RNM
    - Supported by IEEE 1800-2012


Modeling Tradeoffs
Spice
FastSpice
Conservative
Verilog-A
Verilog-AMS
VHDL-AMS
Real number models are the most effective way to abstract AMS functioinality for full chip simulation
Real Number Model (RNM)
Pure Digital
Accuracy
Simulation Performance / Capacity

POWER AWARE

- Different power regions
- Isolation between the regions
- Simulate with UPF flow



AGENDA

- SoC Verification
- What do we do?
- TLV SW environment
- Why Verification?
- TLV in other areas
- **Q&A**

ERICSSON