AMchip simulation and design verification

Francesco Crescioli

Simulation & DV goals

- Verify the global chip logic (the chip will match as we expect, with the latency we expect, in the configuration condition we decided, etc.)
- Verify the synthesis (the chip still behave as we expect?)
- Verify the post layout with backannotation (Encounter says that all the timing constraints are met, but were our timing constraints correctly describing the requirements of the chip?)

AMchip04: C++ AMchip model & VERILOG AM model

- Custom C++ framework for register-level modelling
  - It has been used also for AMchipFPGA and AMchip03 projects
- Developed independently from VHDL code
  - Crosscheck the global behavior of the chip
- Fast execution time

```
amchip04::amchip04(bool tv, int tmprev_en_Low)
  : logicbox(tv)
{
  setPinout(NCTRLIN+8*15+NOPCODE+1,NPAT_DATA*2,1+2);
  /* initialize TV package clock */
  if (usetv) Initialize(PERIOD, RESOLUTION);
  if (usetv) defClock(CLK, "CLK", ACTIVE_HIGH, HIGH);
  setPinName("clk", CLK_PIN);
  setClk(0, RISE); //, TRACING_CLK);
  if (usetv) Clock(CLK,DOWN,T0);
  if (usetv) Clock(CLK,UP,T1);
  // control input
  if (usetv) defSignal(SIGINIT_EV, "init_ev", ACTIVE_HIGH, LOW);
  setPinName("init_ev", INIT_EV_PIN);
  // input Bus0 - Bus7
  if (usetv) defBus(BUS0,        "BUS0",        15,      i2bs(0,15));
  setBusName("Bus0",15,BUS0_FIRST);
  if (usetv) defBus(BUS1,        "BUS1",        15,      i2bs(0,15));
  setBusName("Bus1",15,BUS1_FIRST);
  if (usetv) defBus(BUS2,        "BUS2",        15,      i2bs(0,15));
  setBusName("Bus2",15,BUS2_FIRST);
  if (usetv) defBus(BUS3,        "BUS3",        15,      i2bs(0,15));
  setBusName("Bus3",15,BUS3_FIRST);
  if (usetv) defBus(BUS4,        "BUS4",        15,      i2bs(0,15));
  setBusName("Bus4",15,BUS4_FIRST);
  if (usetv) defBus(BUS5,        "BUS5",        15,      i2bs(0,15));
  setBusName("Bus5",15,BUS5_FIRST);
  if (usetv) defBus(BUS6,        "BUS6",        15,      i2bs(0,15));
  setBusName("Bus6",15,BUS6_FIRST);
```

AMchip04: C++ test vectors

- C++ framework for test vector generation
  - Helper functions for JTAG: writeJTAG_reg(addr, len, data), ...
  - Helper functions for AMchip: writeBus(bus, data), send_init(), send_opcode(opcode), …
  - Easy to read tests for specific functionalities (JTAG, opcodes, …) and run conditions
  - Interface to AMchip C++ model
  - Produce a VERILOG testbench with cycle-accurate AMchip output prediction
- *Everything sync to one core clock!*

```
void test_mem3(amchip04 &chip)
{
  clearCheckTraceGroup(CLK_TRACES);
  clearCheckTraceGroup(TCK_TRACES);
  nCLKperTCK = 1;
  // reset it
  initamchip(&chip);
  setCheckTraceGroup(TCK_TRACES);
  jreset2runtest(&chip);
  // reset all patterns
  reset_patterns(chip, TOT_PATTERNS, 0);
  send_init(chip);
  send_init(chip);
  for (int k = 0; k < 2; ++k) {
  for (int z = 0; z < 64; ++z) {
  WriteScamReg(chip, CONFIG_REG, CONFIG_LENGTH, 0, 0x100, 0x00000009);
  WriteJDATA(chip, (0xAAAA>>k)&0x7fff, (0xAAAA>>k)&0x7fff,
(0xAAAA>>k)&0x7fff, (0xAAAA>>k)&0x7fff, (0xAAAA>>k)&0x7fff,
(0xAAAA>>k)&0x7fff, (0xAAAA>>k)&0x7fff, (0xAAAA>>k)&0x7fff);
  chip.cycle(TRACE_EXT);
  chip.cycle(TRACE_EXT);
  chip.cycle(TRACE_EXT);
  for(int i = z; i < 8192; i+=64) {
      WriteScamReg(chip, JPATT_ADDR, ADDR_SIZE, 0, 0x0, i);
      WriteScamReg(chip, WR_INCop, 0, 0, 0x0, 0x0);
      jSTAYruntest(&chip);
      jSTAYruntest(&chip);
      WriteScamReg(chip, BYPASS, 0, 0, 0x0, 0x0);
  }
```

AMchip04: VERILOG testbench

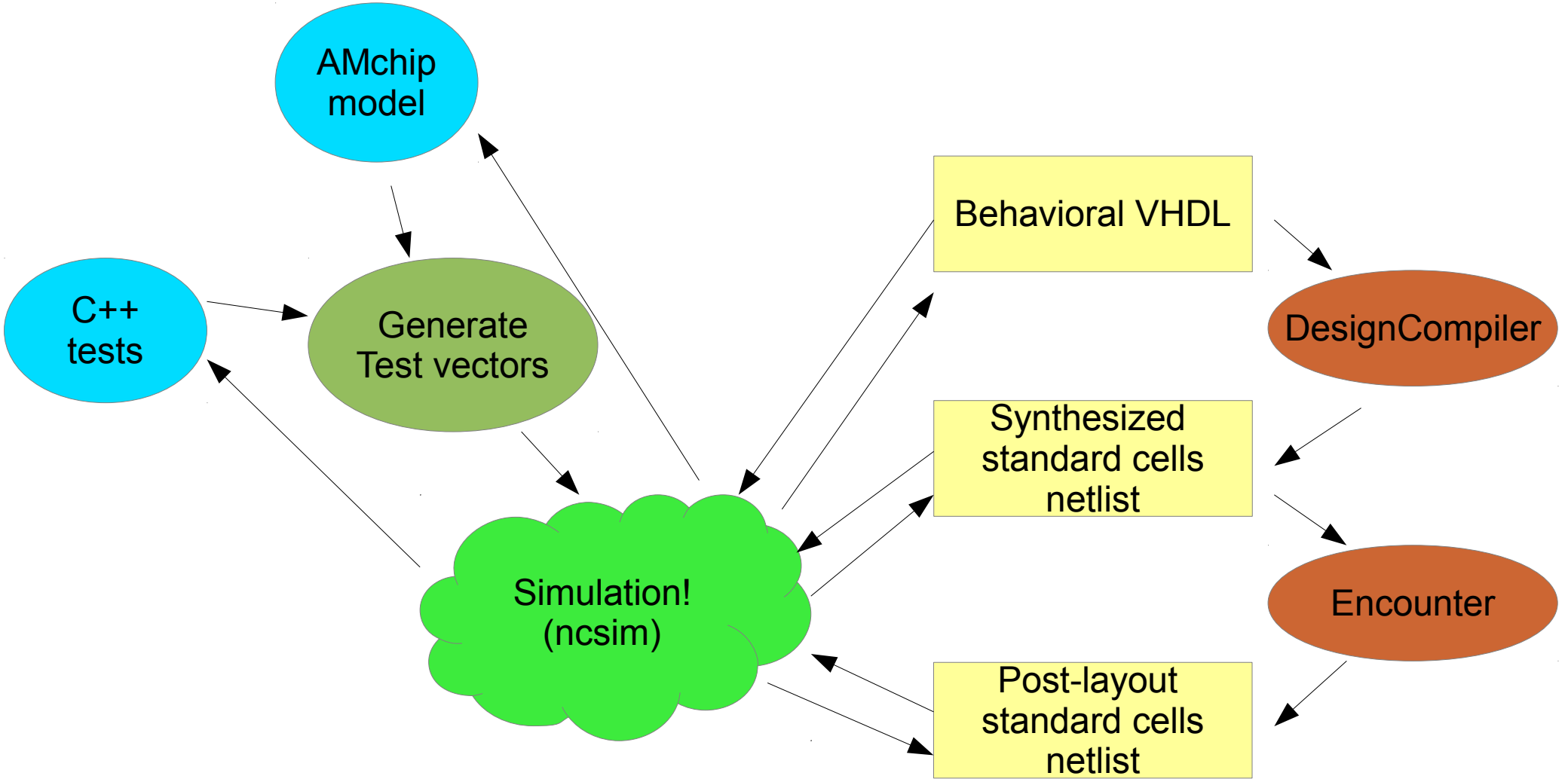
- VERILOG top tb produced by C++ program
- VERILOG behavioral full-custom AM model
- Text file with cycle-by-cycle input stimulus and output prediction
  - The stimulus-response data file can be easily converted in other formats. *It was used by the FPGA test card to test the real chip*.
- The VERILOG tb produces a report with test results summary and detailed errors (if any)
- Scripts to run multiple tests

```
initial
        begin
        if (!$value$plusargs("TESTDIR=%s", TX_TESTDIR))
          $finish;
        TX_FILE = $fopen({TX_TESTDIR,"/TV/results.txt"});
        TX_INFILE = {TX_TESTDIR,"/",TX_INAME,".",TX_ITYP};
        TX_CYCLE = 0;
        for (TX_I=0;TX_I<=TX_MEM_SIZE;TX_I=TX_I+1) TX_MEM[TX_I] = 4'hf;
        $fdisplay(TX_MESG,"Opening file %0s", TX_INFILE);
        $readmemh(TX_INFILE, TX_MEM);

        while (TX_CYCLE<=TX_MEM_SIZE)
                begin
                        fork
                        #7 CHECK_patt_data_b(TX_patt_data_b);
                        #7 CHECK_SAout_Low(TX_SAout_Low);
                        #7 CHECK_DAout_Low(TX_DAout_Low);
                        #9 CHECK_tdo(TX_tdo);
                        @(TX_CLOCK);
                        join
```

```
7FFF_7FFF_7FFF_7FFF_7FFF_7FFF_7FFF_7FFF_0000_0_0_0_1_1_0_0_0_1_1_0000__0__1__Z__0
0000_0000_0000_0000_0000_0000_0000_0000_0000_0_0_0_1_1_0_0_0_1_1_1E00__0__0__Z__0
0000_0000_0000_0000_0000_0000_0000_0000_0000_0_1_0_1_1_0_0_0_1_1_0000__0__1__Z__0
0000_0000_0000_0000_0000_0000_0000_0000_0000_0_0_0_1_1_0_0_0_1_1_0000__0__1__Z__0
0000_0000_0000_0000_0000_0000_0000_0000_0000_0_0_0_1_1_0_0_0_1_1_0000__1__1__Z__0
0000_0000_0000_0000_0000_0000_0000_0000_0000_0_0_0_1_1_0_0_0_1_1_0000__0__1__Z__0
0000_0000_0000_0000_0000_0000_0000_0000_0000_0_0_0_1_1_0_0_0_1_1_0000__0__1__Z__0
0000_0000_0000_0000_0000_0000_0000_0000_0000_0_0_0_1_1_0_0_0_1_1_1E00__0__0__Z__0
0000_0000_0000_0000_0000_0000_0000_0000_0000_0_0_0_1_1_0_0_0_1_1_3FC0__0__1__Z__0
```

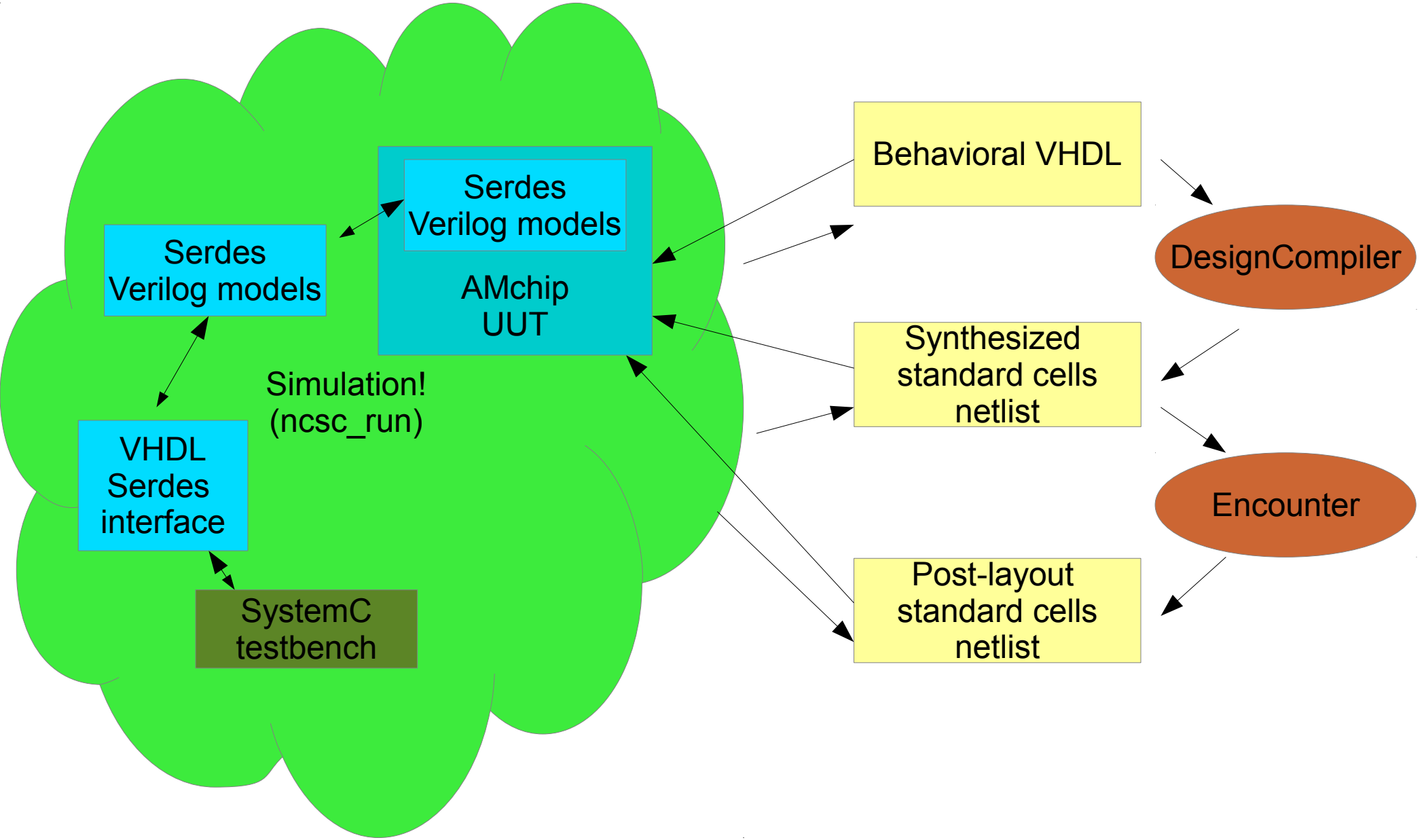
AMchip04: Verification flow

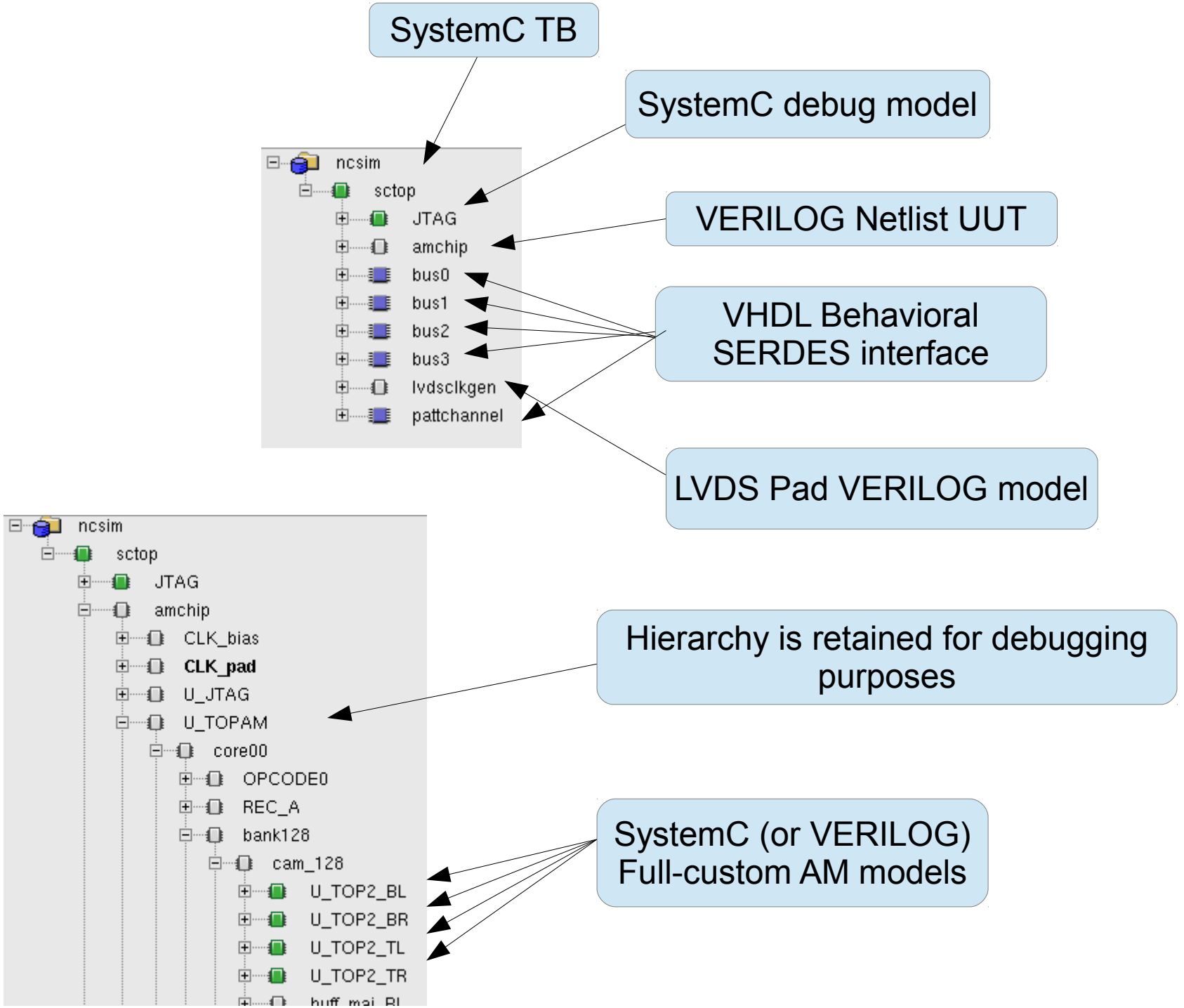


AMchip05_Mini@sic

- **NEW**: 2 Gbit serializers/deserializers for I/O
  - Many internal clocks (core clock, des clocks, ser clock)
  - *I/O pins not related to core clock*
  - How to model serdes with AMchip04 C++ framework?
- Solution:
  - port AMchip04 **C++** tests and model to **SystemC**
  - Run integrated SystemC + serdes model (VERILOG) + AMchip model (VERILOG) + other models (VHDL) simulation

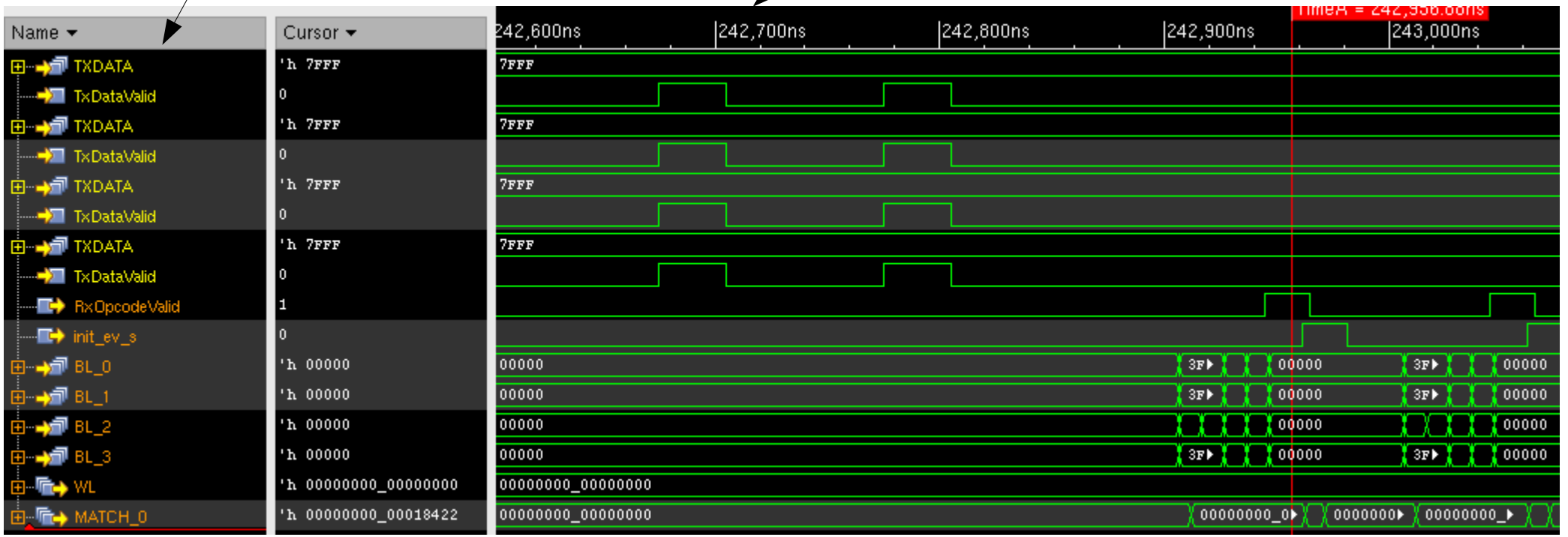
AMchip05_Mini@sic: Verification flow




SystemC TB
SystemC debug model
VERILOG Netlist UUT
VHDL Behavioral
SERDES interface
LVDS Pad VERILOG model
Hierarchy is retained for debugging
purposes
SystemC (or VERILOG)
Full-custom AM models
ncsim
sctop
JTAG
amchip
bus0
bus1
bus2
bus3
lvdsclkgen
pattchannel
ncsim
sctop
JTAG
amchip
CLK_bias
CLK_pad
U_JTAG
U_TOPAM
core00
OPCODE0
REC_A
bank128
cam_128
U_TOP2_BL
U_TOP2_BR
U_TOP2_TL
U_TOP2_TR

Signals from TB

Send some data



Signals inside AMChip

A match!

S&DV lessons learned and foreseen

- High level description (C++/SystemC) of the AMchip provides a very helpful feedback to frontend (VHDL) development
- Unified test vector generation system: test the global logic, synthesized logic, post-layout backannotated netlist and actual chip with the same stimuli!
- Cycle-by-cycle stimulus-response tests were very helpful and easy to handle
  - Not anymore! Serialized IOs run asynchronously, difficult to predict lock, spread spectrum modulation, etc.
  - Switch to a stream based logic: check the sequence, not the exact time of arrival