Applying Transaction-level Debug and Analysis Techniques to DUT Simulated Activity Using Data-Mining Techniques

DVCon 2014

| Leo Chai, R&D, nVidia | Jun Zhao, R&D, Synopsys | Bindesh Patel, CAE, Synopsys

Content will be reformatted into a poster for printing
Abstraction of Simulation Activity

- Signal-level waveforms too fine-grained for most analysis
- **Transaction-level** provides the required clarity for bus-level activity for complex protocols
- Engineers today do the mapping manually
Right Tool for the Right Job

Transactions encapsulate and communicate chunks of data instead of discrete signals
Attributes can be
• Signals (data, addr, ..)
• Messages
• Strings
• Comments
• Counter
• Payload
• Anything

my_operation_var:

```
   do_add
op1=123
op2=45

   do_div
dividend=7
divisor=2

   do_div
dividend=43
divisor=23
```

```
30 55 85 90 115
```

Attributes

```
  Begin Time  End Time

  30  55     85  90
  op1=123
  op2=45

  85  90  115
  dividend=7
  divisor=2

  85  90  115
  dividend=43
  divisor=23
```
Representing Transactions in Debug Tool

- Streams
- Label {begin, end}
- Attributes
- Transactions shown as rectangular box enclosing attributes
- Overlapping transactions
Data Mining from Signal-Level Trace

Why SVA?
- Standard language
- Assertion languages have facilities to specify temporal sequence of events.
- SVA has local variables which can map to attributes for transactions
sequence core_memory_write;
logic [10:0] Addr;
logic [31:0] Data;

(1) ## 0
(EN == 1'b1 && WE == 1'b1,
  Addr = ADDR, Data = DI) ##1
(! (EN == 1'b1 && WE == 1'b1));
endsequence

sequence core_memory_read;
logic [10:0] Addr;
logic [31:0] Data;

(1) ## 0
(WE == 1'b0 && RST == 1'b0 &&
  RDInvalid == 1'b0, Addr = ADDR) ##1
(RDInvalid == 1'b0) ##1
(1, Data = DO);
endsequence

CORE_MEM_WRITE : assert
    property(@(posedge CLK)
        core_memory_write);

CORE_MEM_READ : assert
    property(@(posedge CLK)
        core_memory_read);
nVidia Flow and Code

- Code SVA to describe transaction information.
- Dump trace file which has protocol signal activity.
- Data-mine transactions.
- Load new generated trace file with transaction data.

```verilog
property APB_READ;
    logic [31:0] Addr;    // local variable to record attribute addr
    logic [31:0] Data;    // local variable to record attribute data
    logic [127:0] Client;   // local variable to record attribute client name
    @(posedge pclk) disable iff (disable_ntx_dump)
        (((psel && penable && !pwrite), Addr = paddr) |-> ##1
            (((psel && penable && !pwrite && pready)[->1]), Data = prdata, Client = "dtv");
endproperty

APB_READ_nTX : assert property(APB_READ);
```
In Real-Life

* Example from nVidia environment
Filtering works in a similar fashion

User-defined highlighting (regular expression based)
Protocols often have **complex relationships** between a hierarchy of transactions.
Future: A tool designed from the ground-up for Transaction-Debug

• Next-level Requirements are different than what a waveform can provide. User-driven application-level data mining is key
  – Sorting
  – Filtering
  – re-arranging

• Similar to commercial DBs and SQL

• Abstraction will become critical as the realities push signal-level analysis out of mainstream

• Research into intelligent recognition without any user input