2022 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION EUROPE

MUNICH, GERMANY
DECEMBER 6 - 7, 2022

SAWD: Systemverilog Assertions Waveform-based Development tool

Ahmed Alsawi

QT Technologies Ireland

Qualcomm

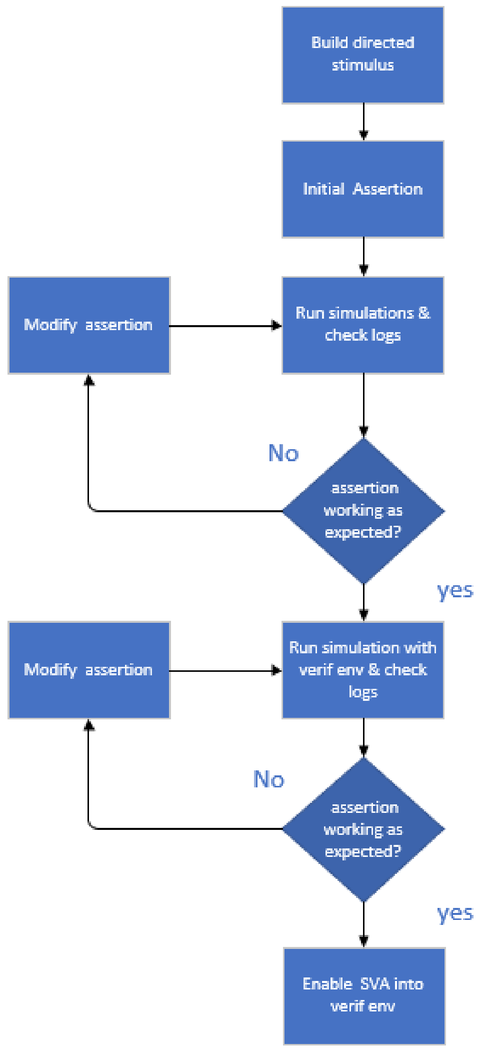
accellera SYSTEMS INITIATIVE

Agenda

- Motivation
- Waveform-based development methodology
- Implementation
  - SVA Frontend
  - Waveform Frontend
  - Evaluation engine
- Graphical user interface
- Example
- Conclusion

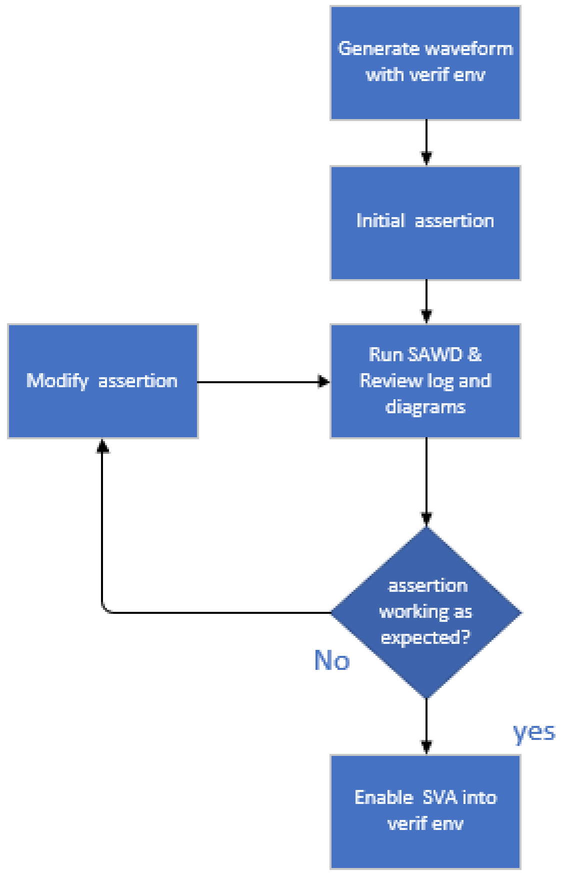
Motivation

- Systemverilog concurrent assertions provide a concise and complicated syntax to define temporal expressions
- The development process can take several iterations to modify, run, and analyze to verify the correctness of an assertion


Build directed stimulus
Initial Assertion
Modify assertion
Run simulations & check logs
No
assertion working as expected?
yes
Modify assertion
Run simulation with verif env & check logs
No
assertion working as expected?
yes
Enable SVA into verif env

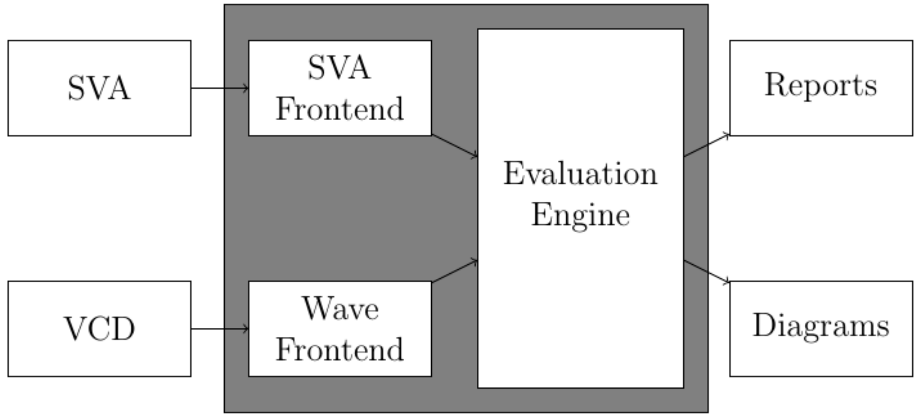
Waveform-based development methodology

- Evaluating SVA on simulator-agnostic waveform
- Generating failing/passing/vacuous reports
- Generating diagrams for evaluation attempts


Generate waveform with verif env
Initial assertion
Modify assertion
Run SAWD & Review log and diagrams
assertion working as expected?
No
yes
Enable SVA into verif env

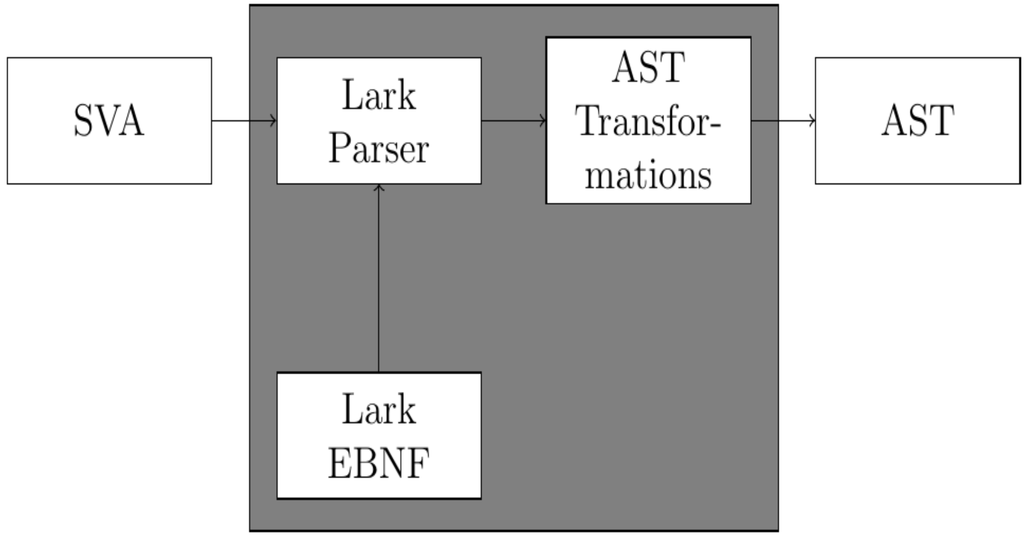
Implementation - Architecture

- SVA frontend parses SVA to generate Abstract Syntax Tree (AST)
- Wave Frontend parses VCD to generate Wave DB
- Evaluation engine uses AST and wave DB to generate SVA reports and diagrams.

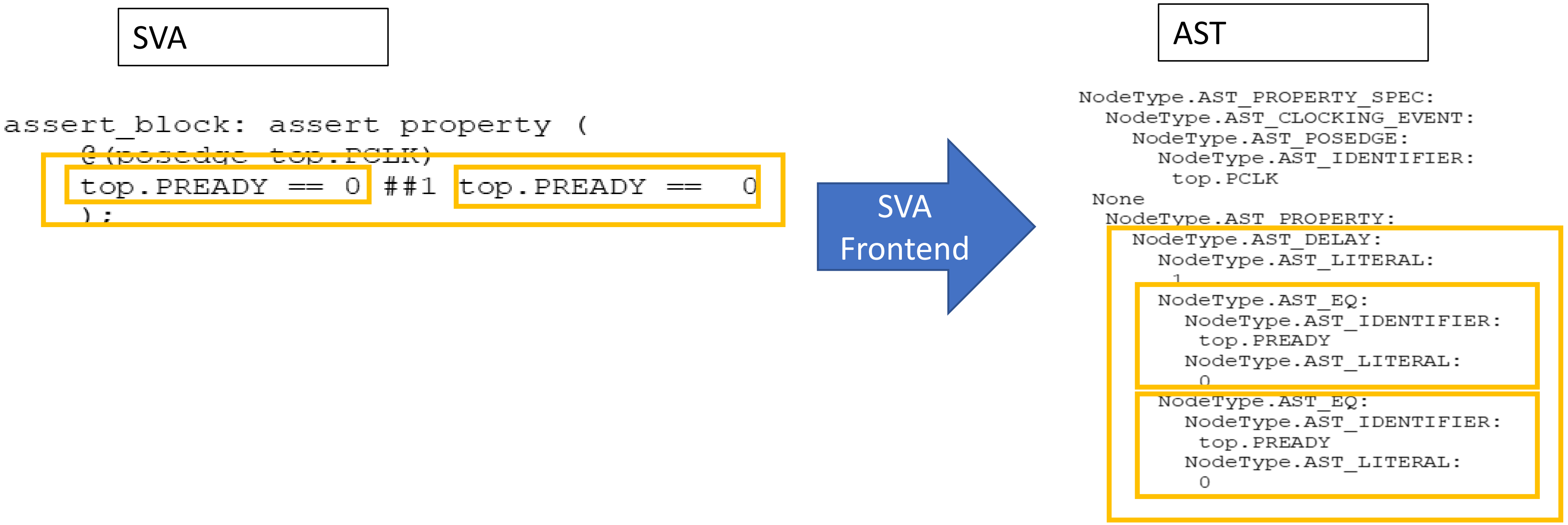


Implementation - SVA Frontend

- Lark parser provides lexical analysis and parser by reading Lark EBNF to generate a parse tree
- AST transformations are custom transformations implemented to transform the parse tree to AST



Implementation - AST Transformations

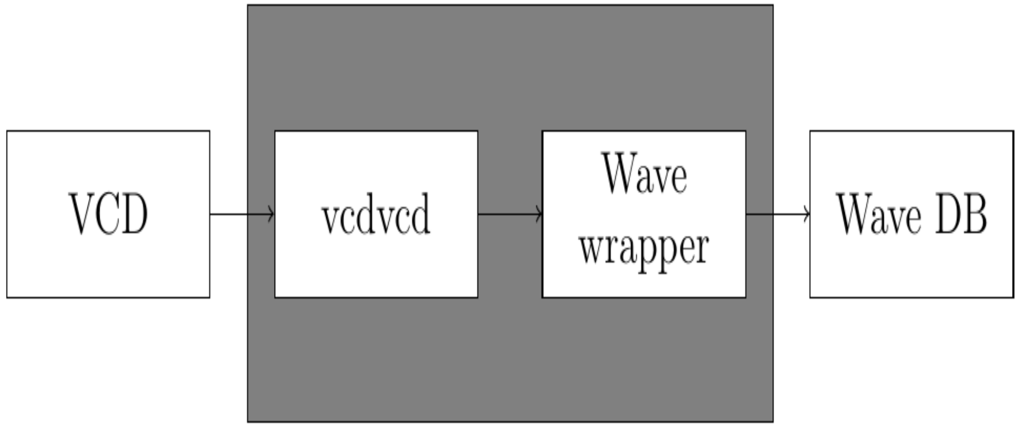


```
assert_block: assert property (
    @(posedge top.PCLK)
    top.PREADY == 0 ##1 top.PREADY ==  0
    );
```

```
NodeType.AST_PROPERTY_SPEC:
  NodeType.AST_CLOCKING_EVENT:
    NodeType.AST_POSEDGE:
      NodeType.AST_IDENTIFIER:
       top.PCLK
 None
  NodeType.AST_PROPERTY:
    NodeType.AST_DELAY:
      NodeType.AST_LITERAL:
       1
      NodeType.AST_EQ:
        NodeType.AST_IDENTIFIER:
         top.PREADY
        NodeType.AST_LITERAL:
         0
      NodeType.AST_EQ:
        NodeType.AST_IDENTIFIER:
         top.PREADY
        NodeType.AST_LITERAL:
         0
```

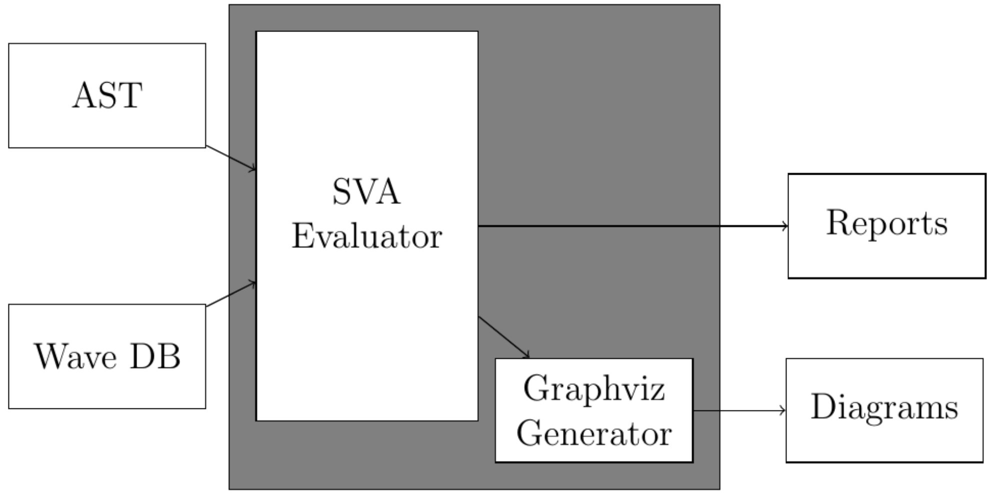
Implementation - Wave Frontend

- Wave frontend uses python package vcdvcd to parse vcd file
- Wave wrapper is an abstraction layer to provide wave DB to the evaluation engine

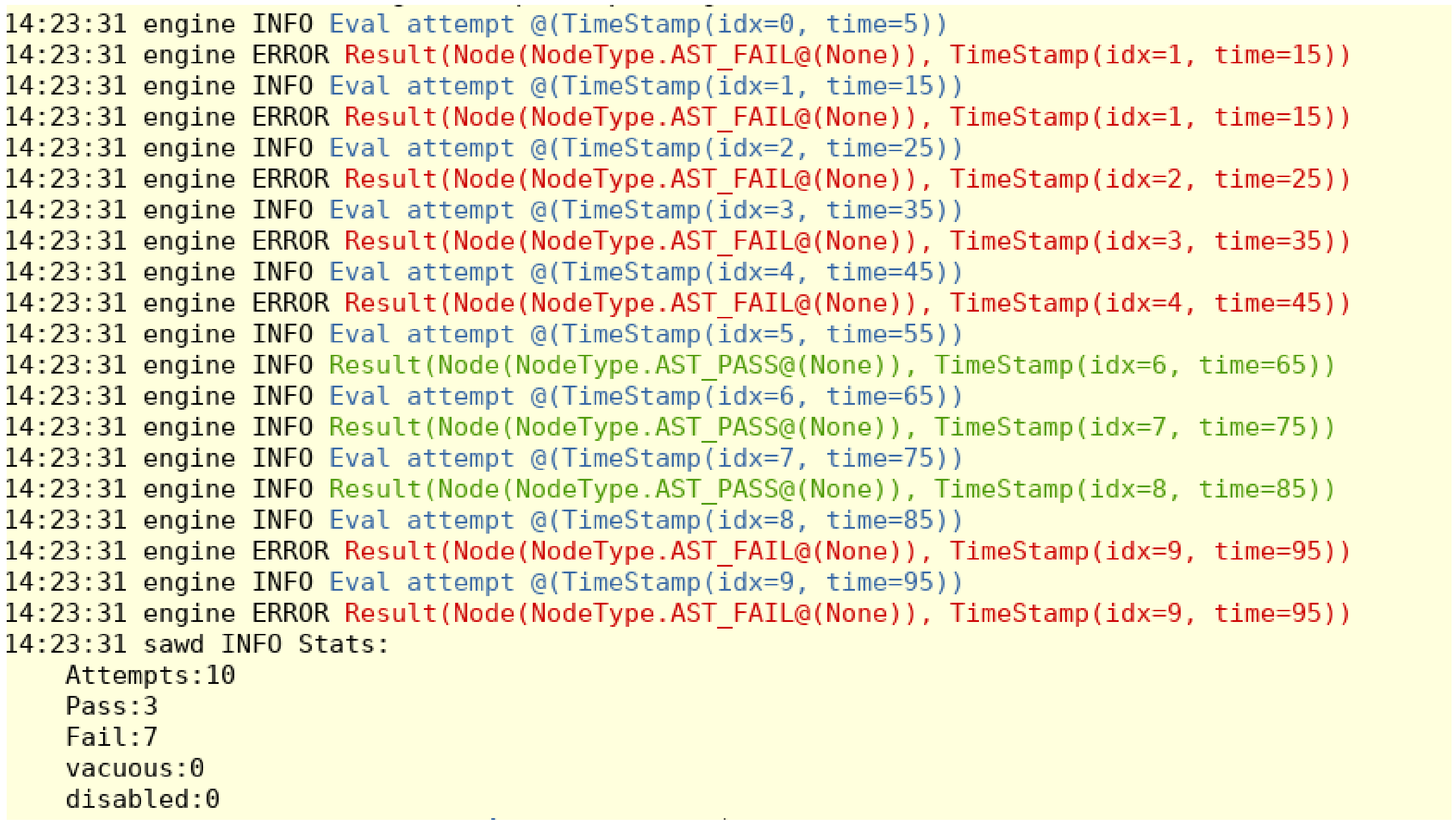


Implementation - Evaluation Engine

- The SVA evaluator processes AST and wave DB to generate reports for each evaluation attempt
- The Graphviz generator uses time-aware expression and Graphviz utility to generate attempts diagram



Implementation - Reports



```
14:23:31 engine INFO Eval attempt @(TimeStamp(idx=0, time=5))
14:23:31 engine ERROR Result(Node(NodeType.AST_FAIL@(None)), TimeStamp(idx=1, time=15))
14:23:31 engine INFO Eval attempt @(TimeStamp(idx=1, time=15))
14:23:31 engine ERROR Result(Node(NodeType.AST_FAIL@(None)), TimeStamp(idx=1, time=15))
14:23:31 engine INFO Eval attempt @(TimeStamp(idx=2, time=25))
14:23:31 engine ERROR Result(Node(NodeType.AST_FAIL@(None)), TimeStamp(idx=2, time=25))
14:23:31 engine INFO Eval attempt @(TimeStamp(idx=3, time=35))
14:23:31 engine ERROR Result(Node(NodeType.AST_FAIL@(None)), TimeStamp(idx=3, time=35))
14:23:31 engine INFO Eval attempt @(TimeStamp(idx=4, time=45))
14:23:31 engine ERROR Result(Node(NodeType.AST_FAIL@(None)), TimeStamp(idx=4, time=45))
14:23:31 engine INFO Eval attempt @(TimeStamp(idx=5, time=55))
14:23:31 engine INFO Result(Node(NodeType.AST_PASS@(None)), TimeStamp(idx=6, time=65))
14:23:31 engine INFO Eval attempt @(TimeStamp(idx=6, time=65))
14:23:31 engine INFO Result(Node(NodeType.AST_PASS@(None)), TimeStamp(idx=7, time=75))
14:23:31 engine INFO Eval attempt @(TimeStamp(idx=7, time=75))
14:23:31 engine INFO Result(Node(NodeType.AST_PASS@(None)), TimeStamp(idx=8, time=85))
14:23:31 engine INFO Eval attempt @(TimeStamp(idx=8, time=85))
14:23:31 engine ERROR Result(Node(NodeType.AST_FAIL@(None)), TimeStamp(idx=9, time=95))
14:23:31 engine INFO Eval attempt @(TimeStamp(idx=9, time=95))
14:23:31 engine ERROR Result(Node(NodeType.AST_FAIL@(None)), TimeStamp(idx=9, time=95))
14:23:31 sawd INFO Stats:
    Attempts:10
    Pass:3
    Fail:7
    vacuous:0
    disabled:0
```

Attempts report

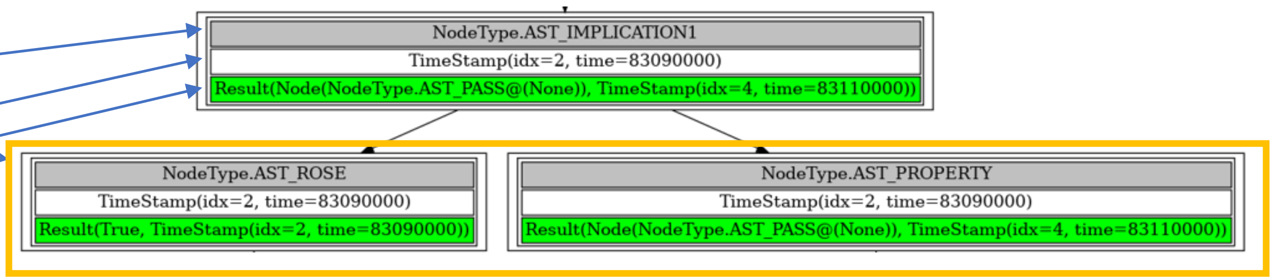
Stats report

Implementation - Time-Aware Expression Tree

- The time-aware expression tree is a data structure to keep track of start and end timestamps and expression results

```
class Node:
    """ """

    def __init__(self, type_,  *args, **kargs):
        self.type_      : NodeType  = type_
        self.children   :[Node]     = list(args)
        self.ts         : Result    = None
        self.evaluated : Result    = None
```



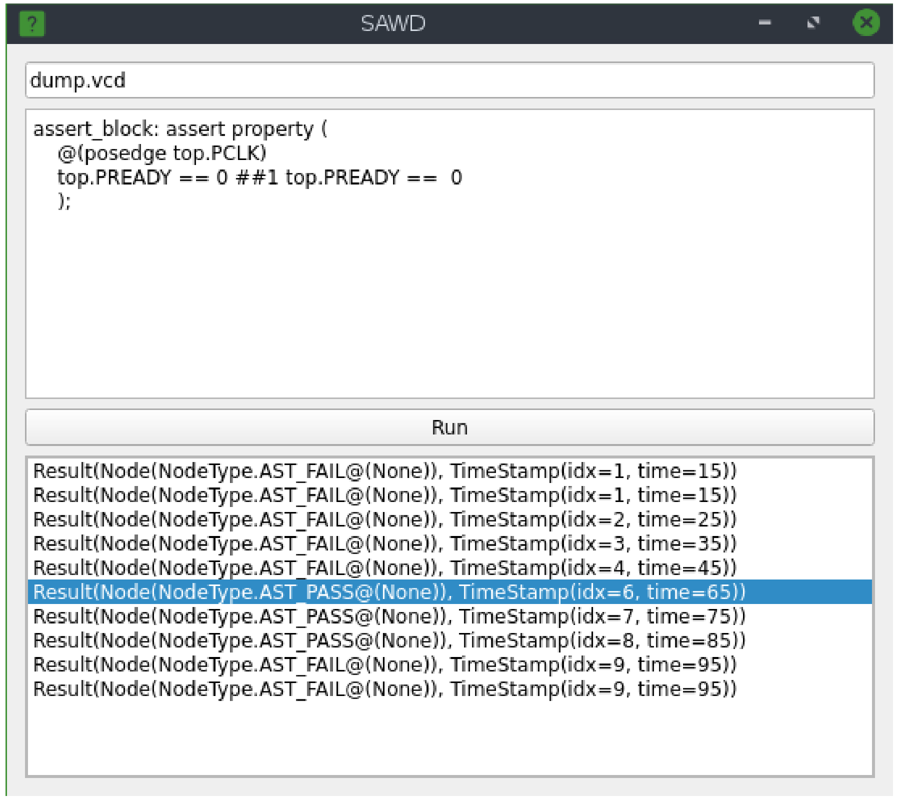
```
NodeType.AST_IMPLICATION1
TimeStamp(idx=2, time=83090000)
Result(Node(NodeType.AST_PASS@(None)), TimeStamp(idx=4, time=83110000))

NodeType.AST_ROSE
TimeStamp(idx=2, time=83090000)
Result(True, TimeStamp(idx=2, time=83090000))

NodeType.AST_PROPERTY
TimeStamp(idx=2, time=83090000)
Result(Node(NodeType.AST_PASS@(None)), TimeStamp(idx=4, time=83110000))
```

Graphical user interface

- SAWD Graphical User Interface uses PyQt5
  - Path to VCD file
  - SVA editor
  - Evaluation attempts result
- The evaluation attempts list is clickable to open evaluation attempt diagram in a separate window.



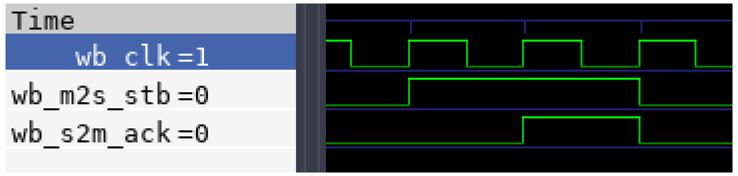
```
dump.vcd
```

```
assert_block: assert property (
   @(posedge top.PCLK)
   top.PREADY == 0 ##1 top.PREADY ==  0
   );
```

```
Result(Node(NodeType.AST_FAIL@(None)), TimeStamp(idx=1, time=15))
Result(Node(NodeType.AST_FAIL@(None)), TimeStamp(idx=1, time=15))
Result(Node(NodeType.AST_FAIL@(None)), TimeStamp(idx=2, time=25))
Result(Node(NodeType.AST_FAIL@(None)), TimeStamp(idx=3, time=35))
Result(Node(NodeType.AST_FAIL@(None)), TimeStamp(idx=4, time=45))
Result(Node(NodeType.AST_PASS@(None)), TimeStamp(idx=6, time=65))
Result(Node(NodeType.AST_PASS@(None)), TimeStamp(idx=7, time=75))
Result(Node(NodeType.AST_PASS@(None)), TimeStamp(idx=8, time=85))
Result(Node(NodeType.AST_FAIL@(None)), TimeStamp(idx=9, time=95))
Result(Node(NodeType.AST_FAIL@(None)), TimeStamp(idx=9, time=95))
```

Example – Initial SVA for wishbone stb/ack

VCD



SVA

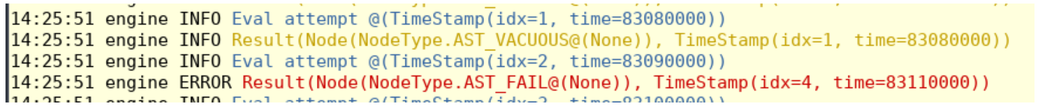
```
assert_block: assert property (
    @(posedge testbench.top.wb_clk)
    disable iff (testbench.top.wb_rst)
    $rose(testbench.top.wb_m2s_stb) |->

          ##1 $rose(testbench.top.wb_s2m_ack)

                 ##1 $fell(testbench.top.wb_s2m_ack) &&
    $rose(testbench.top.wb_m2s_stb)
    );
```

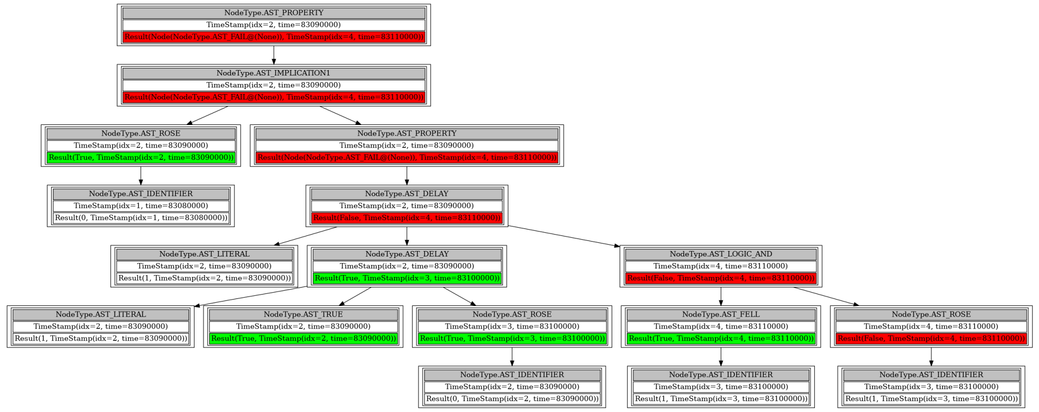
SAWD

Reports

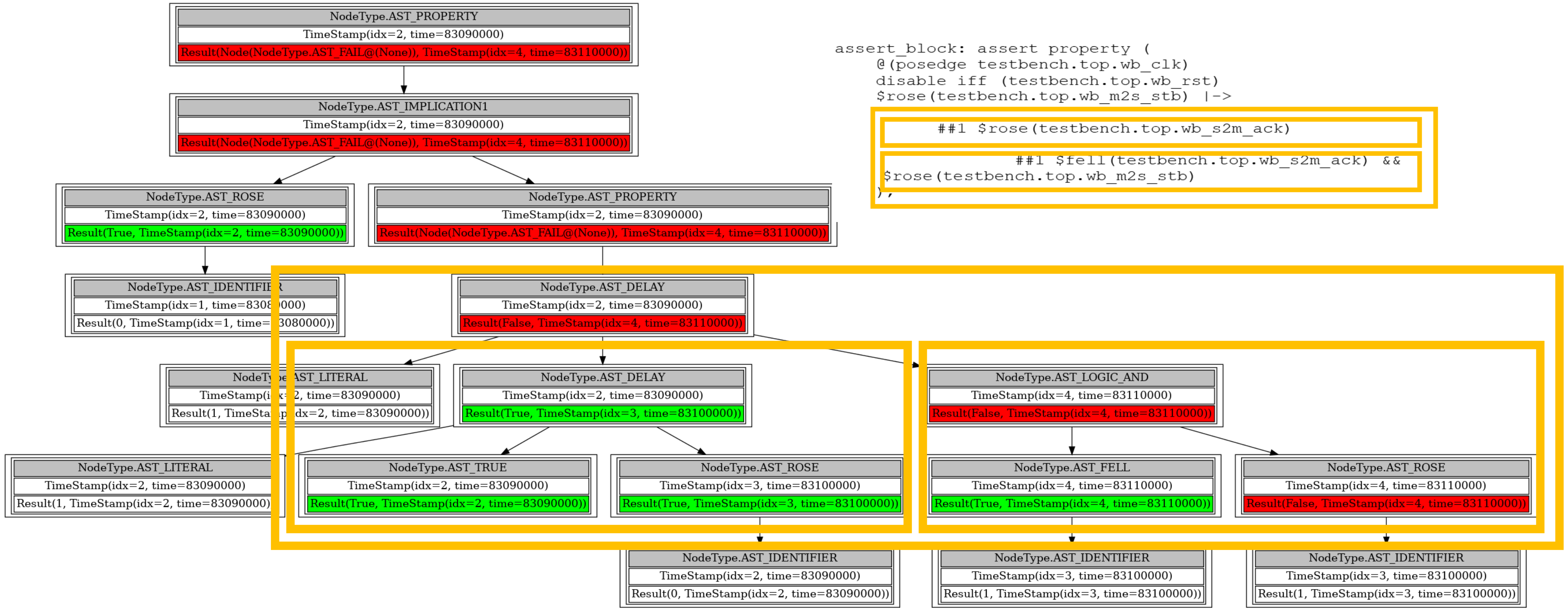


```
14:25:51 engine INFO Eval attempt @(TimeStamp(idx=1, time=83080000))
14:25:51 engine INFO Result(Node(NodeType.AST_VACUOUS@(None)), TimeStamp(idx=1, time=83080000))
14:25:51 engine INFO Eval attempt @(TimeStamp(idx=2, time=83090000))
14:25:51 engine ERROR Result(Node(NodeType.AST_FAIL@(None)), TimeStamp(idx=4, time=83110000))
```

Failing evaluation attempt


NodeType.AST_PROPERTY
TimeStamp(idx=2, time=83090000)
Result(Node(NodeType.AST_FAIL@(None)), TimeStamp(idx=4, time=83110000))
NodeType.AST_IMPLICATION1
TimeStamp(idx=2, time=83090000)
Result(Node(NodeType.AST_FAIL@(None)), TimeStamp(idx=4, time=83110000))
NodeType.AST_ROSE
TimeStamp(idx=2, time=83090000)
Result(True, TimeStamp(idx=2, time=83090000))
NodeType.AST_PROPERTY
TimeStamp(idx=2, time=83090000)
Result(Node(NodeType.AST_FAIL@(None)), TimeStamp(idx=4, time=83110000))
NodeType.AST_IDENTIFIER
TimeStamp(idx=1, time=83080000)
Result(0, TimeStamp(idx=1, time=83080000))
NodeType.AST_DELAY
TimeStamp(idx=2, time=83090000)
Result(False, TimeStamp(idx=4, time=83110000))
NodeType.AST_LITERAL
TimeStamp(idx=2, time=83090000)
Result(1, TimeStamp(idx=2, time=83090000))
NodeType.AST_DELAY
TimeStamp(idx=2, time=83090000)
Result(True, TimeStamp(idx=3, time=83100000))
NodeType.AST_LOGIC_AND
TimeStamp(idx=4, time=83110000)
Result(False, TimeStamp(idx=4, time=83110000))
NodeType.AST_LITERAL
TimeStamp(idx=2, time=83090000)
Result(1, TimeStamp(idx=2, time=83090000))
NodeType.AST_TRUE
TimeStamp(idx=2, time=83090000)
Result(True, TimeStamp(idx=2, time=83090000))
NodeType.AST_ROSE
TimeStamp(idx=3, time=83100000)
Result(True, TimeStamp(idx=3, time=83100000))
NodeType.AST_FELL
TimeStamp(idx=4, time=83110000)
Result(True, TimeStamp(idx=4, time=83110000))
NodeType.AST_ROSE
TimeStamp(idx=4, time=83110000)
Result(False, TimeStamp(idx=4, time=83110000))
NodeType.AST_IDENTIFIER
TimeStamp(idx=2, time=83090000)
Result(0, TimeStamp(idx=2, time=83090000))
NodeType.AST_IDENTIFIER
TimeStamp(idx=3, time=83100000)
Result(1, TimeStamp(idx=3, time=83100000))
NodeType.AST_IDENTIFIER
TimeStamp(idx=3, time=83100000)
Result(1, TimeStamp(idx=3, time=83100000))

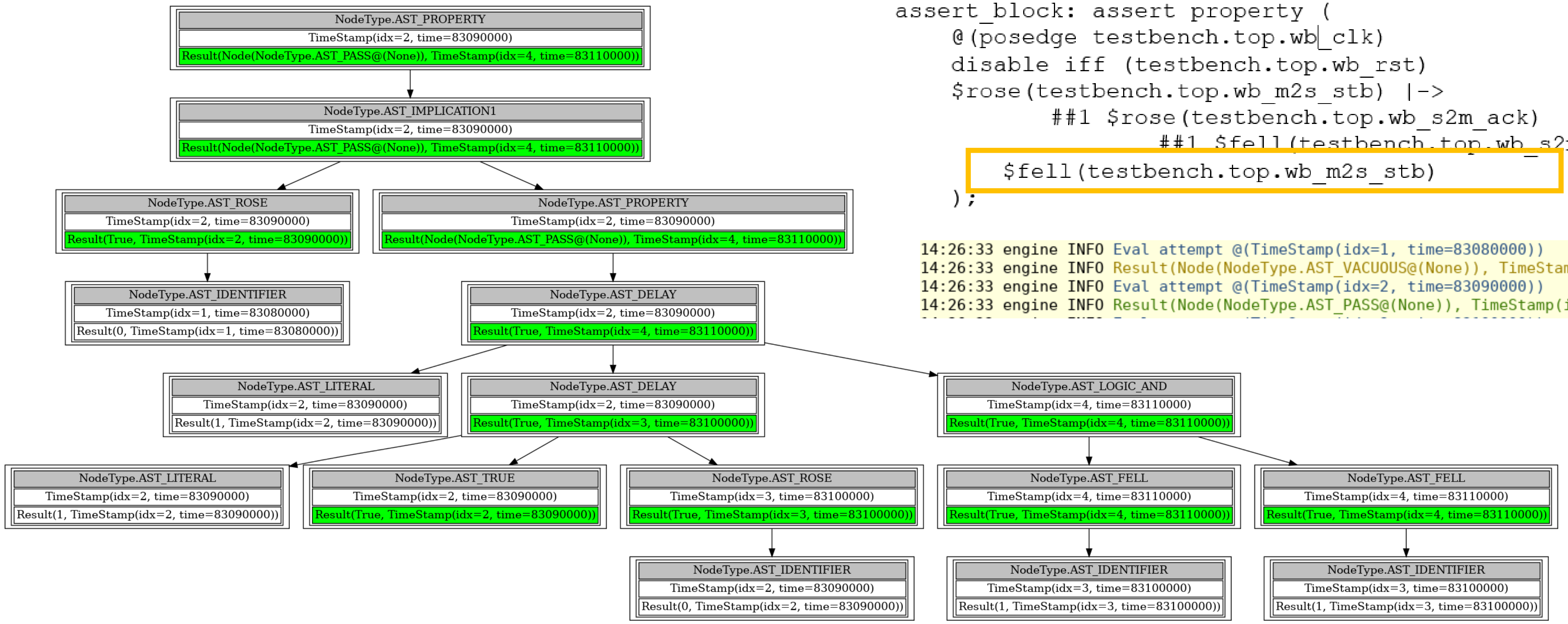
Example - Failing Attempt review



NodeType.AST_PROPERTY
TimeStamp(idx=2, time=83090000)
Result(Node(NodeType.AST_FAIL@(None)), TimeStamp(idx=4, time=83110000))
NodeType.AST_IMPLICATION1
TimeStamp(idx=2, time=83090000)
Result(Node(NodeType.AST_FAIL@(None)), TimeStamp(idx=4, time=83110000))
NodeType.AST_ROSE
TimeStamp(idx=2, time=83090000)
Result(True, TimeStamp(idx=2, time=83090000))
NodeType.AST_PROPERTY
TimeStamp(idx=2, time=83090000)
Result(Node(NodeType.AST_FAIL@(None)), TimeStamp(idx=4, time=83110000))
NodeType.AST_IDENTIFIER
TimeStamp(idx=1, time=83080000)
Result(0, TimeStamp(idx=1, time=83080000))
NodeType.AST_DELAY
TimeStamp(idx=2, time=83090000)
Result(False, TimeStamp(idx=4, time=83110000))
NodeType.AST_LITERAL
TimeStamp(idx=2, time=83090000)
Result(1, TimeStamp(idx=2, time=83090000))
NodeType.AST_DELAY
TimeStamp(idx=2, time=83090000)
Result(True, TimeStamp(idx=3, time=83100000))
NodeType.AST_LOGIC_AND
TimeStamp(idx=4, time=83110000)
Result(False, TimeStamp(idx=4, time=83110000))
NodeType.AST_LITERAL
TimeStamp(idx=2, time=83090000)
Result(1, TimeStamp(idx=2, time=83090000))
NodeType.AST_TRUE
TimeStamp(idx=2, time=83090000)
Result(True, TimeStamp(idx=2, time=83090000))
NodeType.AST_ROSE
TimeStamp(idx=3, time=83100000)
Result(True, TimeStamp(idx=3, time=83100000))
NodeType.AST_FELL
TimeStamp(idx=4, time=83110000)
Result(True, TimeStamp(idx=4, time=83110000))
NodeType.AST_ROSE
TimeStamp(idx=4, time=83110000)
Result(False, TimeStamp(idx=4, time=83110000))
NodeType.AST_IDENTIFIER
TimeStamp(idx=2, time=83090000)
Result(0, TimeStamp(idx=2, time=83090000))
NodeType.AST_IDENTIFIER
TimeStamp(idx=3, time=83100000)
Result(1, TimeStamp(idx=3, time=83100000))
NodeType.AST_IDENTIFIER
TimeStamp(idx=3, time=83100000)
Result(1, TimeStamp(idx=3, time=83100000))

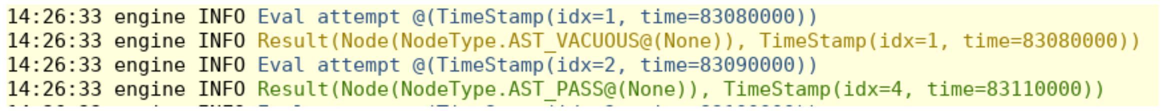
```
assert_block: assert property (
    @(posedge testbench.top.wb_clk)
    disable iff (testbench.top.wb_rst)
    $rose(testbench.top.wb_m2s_stb) |->
        ##1 $rose(testbench.top.wb_s2m_ack)
                ##1 $fell(testbench.top.wb_s2m_ack) &&
$rose(testbench.top.wb_m2s_stb)
);
```

Example - After changing $rose to $fell



NodeType.AST_PROPERTY
TimeStamp(idx=2, time=83090000)
Result(Node(NodeType.AST_PASS@(None)), TimeStamp(idx=4, time=83110000))
NodeType.AST_IMPLICATION1
TimeStamp(idx=2, time=83090000)
Result(Node(NodeType.AST_PASS@(None)), TimeStamp(idx=4, time=83110000))
NodeType.AST_ROSE
TimeStamp(idx=2, time=83090000)
Result(True, TimeStamp(idx=2, time=83090000))
NodeType.AST_PROPERTY
TimeStamp(idx=2, time=83090000)
Result(Node(NodeType.AST_PASS@(None)), TimeStamp(idx=4, time=83110000))
NodeType.AST_IDENTIFIER
TimeStamp(idx=1, time=83080000)
Result(0, TimeStamp(idx=1, time=83080000))
NodeType.AST_DELAY
TimeStamp(idx=2, time=83090000)
Result(True, TimeStamp(idx=4, time=83110000))
NodeType.AST_LITERAL
TimeStamp(idx=2, time=83090000)
Result(1, TimeStamp(idx=2, time=83090000))
NodeType.AST_DELAY
TimeStamp(idx=2, time=83090000)
Result(True, TimeStamp(idx=3, time=83100000))
NodeType.AST_LOGIC_AND
TimeStamp(idx=4, time=83110000)
Result(True, TimeStamp(idx=4, time=83110000))
NodeType.AST_LITERAL
TimeStamp(idx=2, time=83090000)
Result(1, TimeStamp(idx=2, time=83090000))
NodeType.AST_TRUE
TimeStamp(idx=2, time=83090000)
Result(True, TimeStamp(idx=2, time=83090000))
NodeType.AST_ROSE
TimeStamp(idx=3, time=83100000)
Result(True, TimeStamp(idx=3, time=83100000))
NodeType.AST_FELL
TimeStamp(idx=4, time=83110000)
Result(True, TimeStamp(idx=4, time=83110000))
NodeType.AST_FELL
TimeStamp(idx=4, time=83110000)
Result(True, TimeStamp(idx=4, time=83110000))
NodeType.AST_IDENTIFIER
TimeStamp(idx=2, time=83090000)
Result(0, TimeStamp(idx=2, time=83090000))
NodeType.AST_IDENTIFIER
TimeStamp(idx=3, time=83100000)
Result(1, TimeStamp(idx=3, time=83100000))
NodeType.AST_IDENTIFIER
TimeStamp(idx=3, time=83100000)
Result(1, TimeStamp(idx=3, time=83100000))

```
assert_block: assert property (
    @(posedge testbench.top.wb_clk)
    disable iff (testbench.top.wb_rst)
    $rose(testbench.top.wb_m2s_stb) |->
           ##1 $rose(testbench.top.wb_s2m_ack)
                   ##1 $fell(testbench.top.wb_s2m_ack) &&
       $fell(testbench.top.wb_m2s_stb)
    );
```



```
14:26:33 engine INFO Eval attempt @(TimeStamp(idx=1, time=83080000))
14:26:33 engine INFO Result(Node(NodeType.AST_VACUOUS@(None)), TimeStamp(idx=1, time=83080000))
14:26:33 engine INFO Eval attempt @(TimeStamp(idx=2, time=83090000))
14:26:33 engine INFO Result(Node(NodeType.AST_PASS@(None)), TimeStamp(idx=4, time=83110000))
```

Conclusion

- SAWD provides a tool to develop SVA by evaluating SVA on VCD directly without rerunning simulations
- The results show SVA evaluation reports and generated diagrams for passing/failing attempts
- Advantages
  - Simulator-agnostic and using only open-source packages
  - Faster SVA testing and shorter turn-around time
  - Help understand assertion evaluation attempts

Questions?

Contact Ahmed Alsawi at aalsawi@qti.qualcomm.com