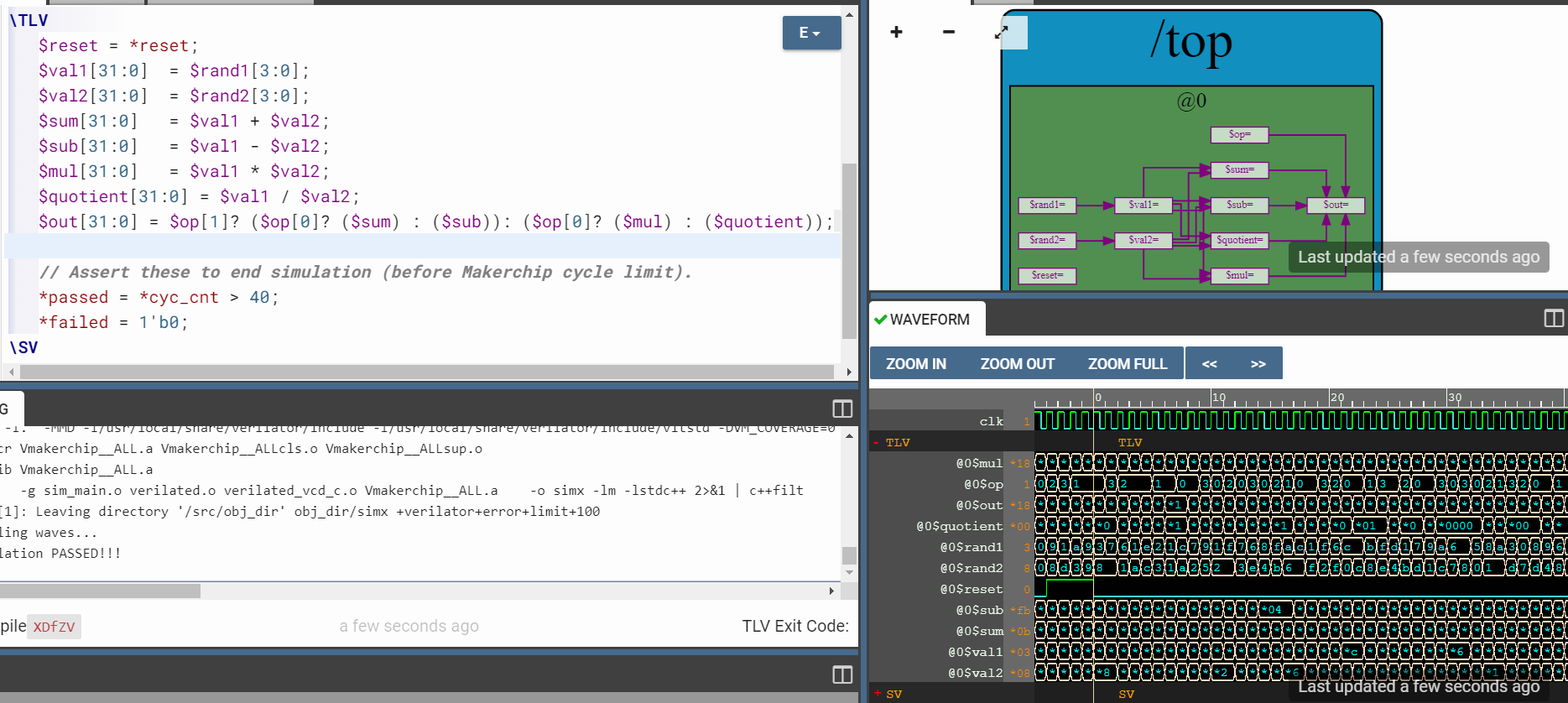
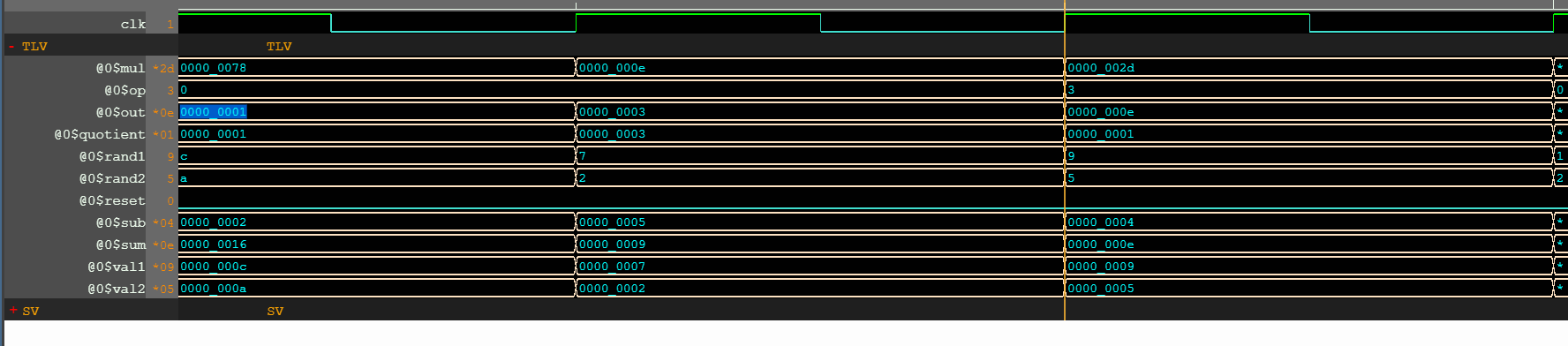
**Day3 TLVERILOG ASSIGNMENTS:**

**Combinational calculator using TLVERILOG:**





Code:

\m4\_TLV\_version 1d: tl-x.org

\SV

// =========================================

// Welcome! Try the tutorials via the menu.

// =========================================

// Default Makerchip TL-Verilog Code Template

// Macro providing required top-level module definition, random

// stimulus support, and Verilator config.

m4\_makerchip\_module // (Expanded in Nav-TLV pane.)

\TLV

$reset = \*reset;

$val1[31:0] = $rand1[3:0];

$val2[31:0] = $rand2[3:0];

$sum[31:0] = $val1 + $val2;

$sub[31:0] = $val1 - $val2;

$mul[31:0] = $val1 \* $val2;

$quotient[31:0] = $val1 / $val2;

$out[31:0] = $op[1]? ($op[0]? ($sum) : ($sub)): ($op[0]? ($mul) : ($quotient));

// Assert these to end simulation (before Makerchip cycle limit).

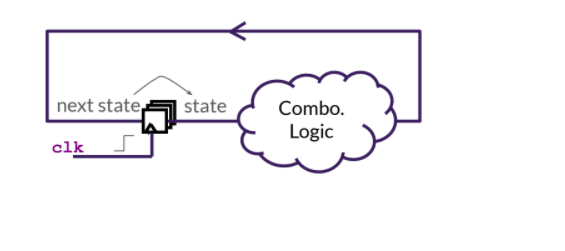
\*passed = \*cyc\_cnt > 40;

\*failed = 1'b0;

\SV

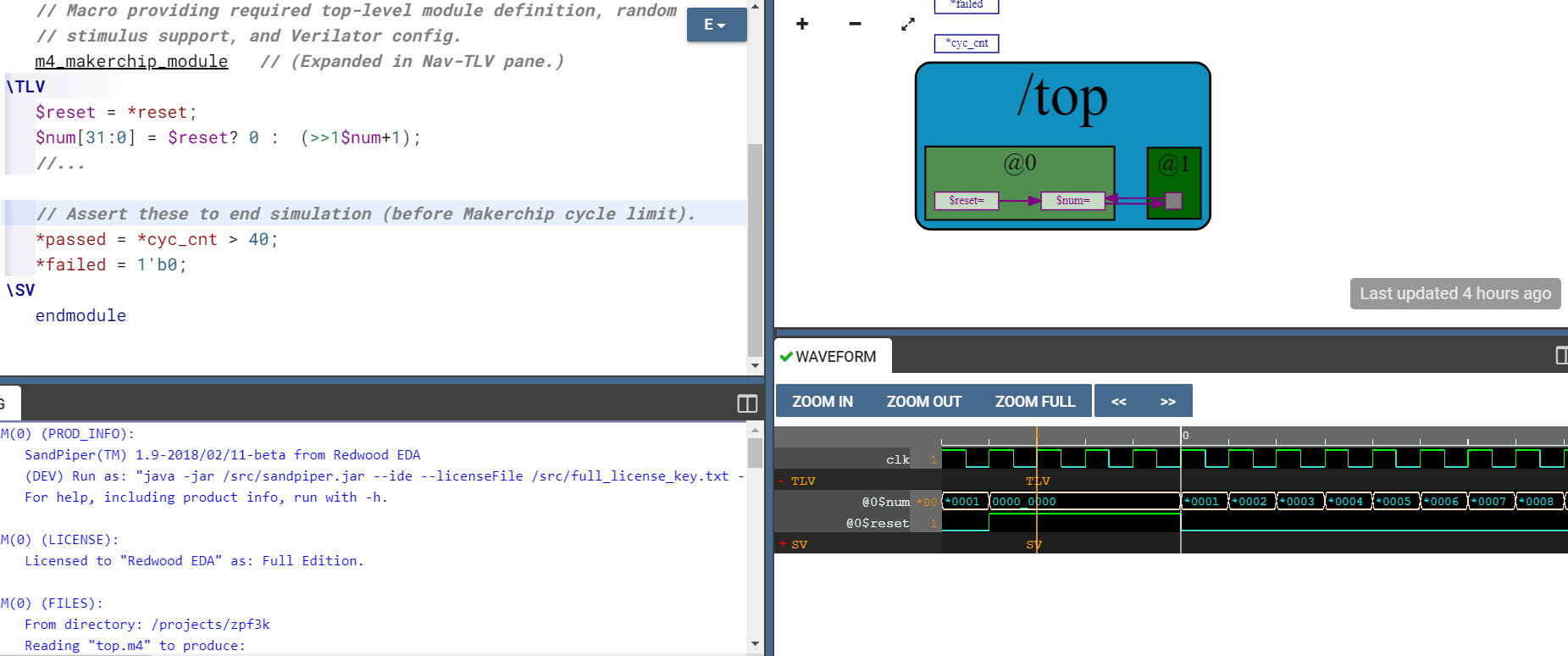
endmodule

**SEQUENTIAL CIRCUITS: [implementation is something like below]**



Next stage of any flop is obtained by applying combinational logic on previous stage.

**Counter**: combo logic for counter contains one adder and a [2:1] mux, selection line of mux is reset



>>1$num in the code means the datacapturedafter one clockadvance

Code:

\m4\_TLV\_version 1d: tl-x.org

\SV

// Macro providing required top-level module definition, random

// stimulus support, and Verilator config.

m4\_makerchip\_module // (Expanded in Nav-TLV pane.)

\TLV

$reset = \*reset;

$num[31:0] = $reset? 0 : (>>1$num+1);

//...

// Assert these to end simulation (before Makerchip cycle limit).

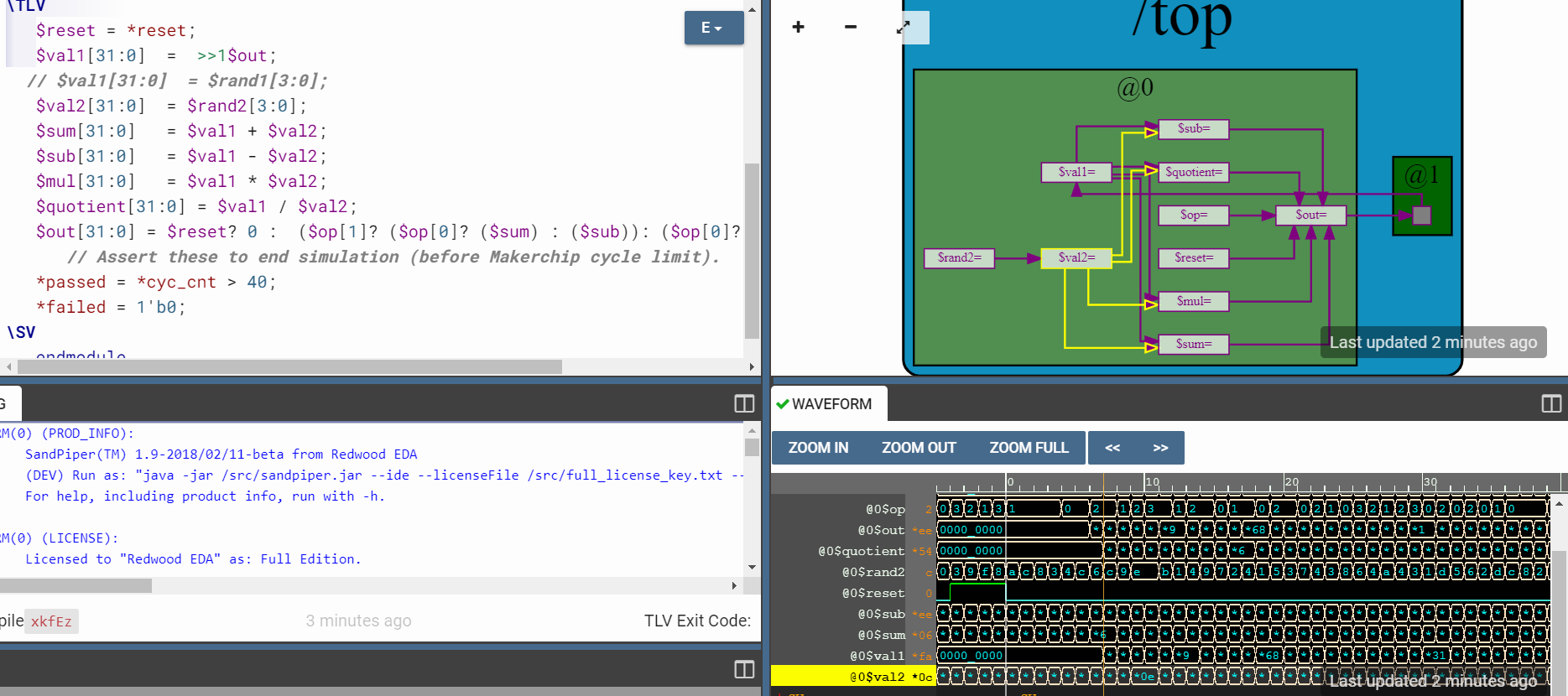
\*passed = \*cyc\_cnt > 40;

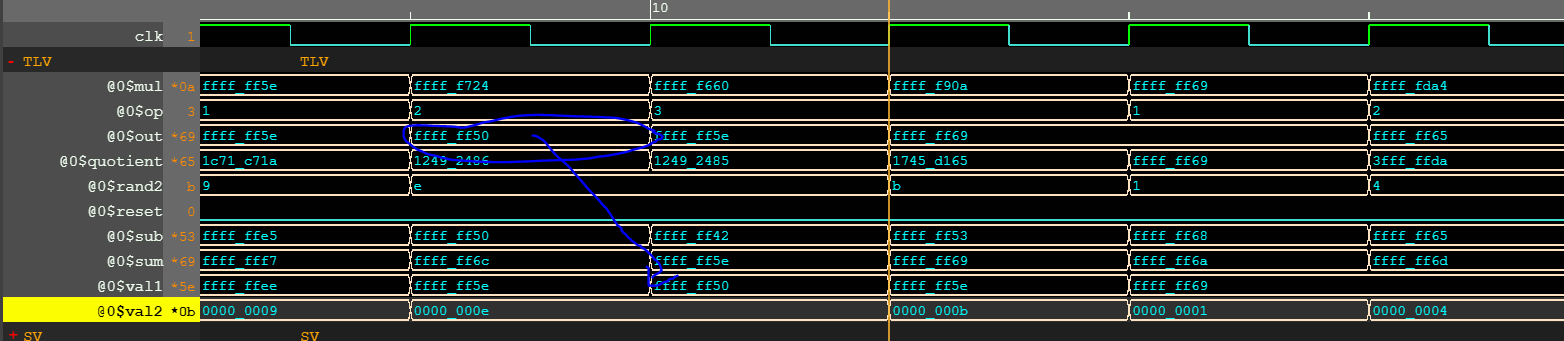
\*failed = 1'b0;

\SV

endmodule

**Sequential calculator:**





The data gets copied from out to val1 in next clock cycle.

Code:

\m4\_TLV\_version 1d: tl-x.org

\SV

// =========================================

// Welcome! Try the tutorials via the menu.

// =========================================

// Default Makerchip TL-Verilog Code Template

// Macro providing required top-level module definition, random

// stimulus support, and Verilator config.

m4\_makerchip\_module // (Expanded in Nav-TLV pane.)

\TLV

$reset = \*reset;

$val1[31:0] = >>1$out;

// $val1[31:0] = $rand1[3:0];

$val2[31:0] = $rand2[3:0];

$sum[31:0] = $val1 + $val2;

$sub[31:0] = $val1 - $val2;

$mul[31:0] = $val1 \* $val2;

$quotient[31:0] = $val1 / $val2;

$out[31:0] = $reset? 0 : ($op[1]? ($op[0]? ($sum) : ($sub)): ($op[0]? ($mul) : ($quotient)));

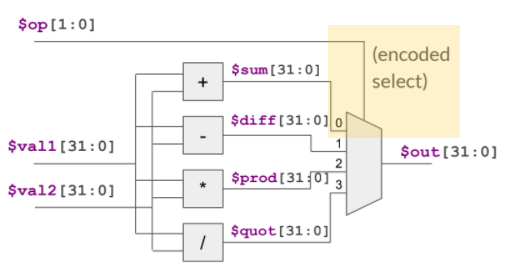
// Assert these to end simulation (before Makerchip cycle limit).

\*passed = \*cyc\_cnt > 40;

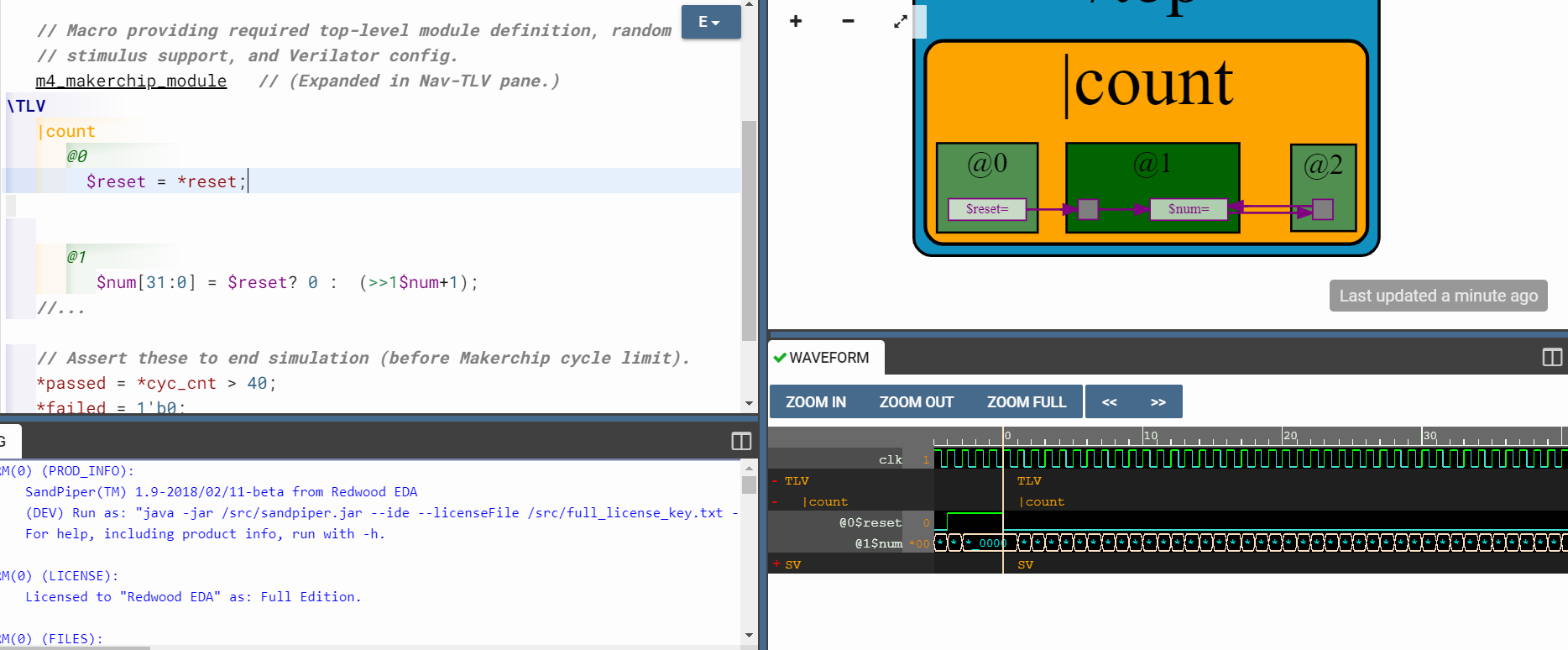
\*failed = 1'b0;

\SV

endmodule

combo logic: 

**Counterwithpipeline:**



Code:

m4\_makerchip\_module // (Expanded in Nav-TLV pane.)

\TLV

|count //pipe

@0 //stage

$reset = \*reset;

@1

$num[31:0] = $reset? 0 : (>>1$num+1);

//...

// Assert these to end simulation (before Makerchip cycle limit).

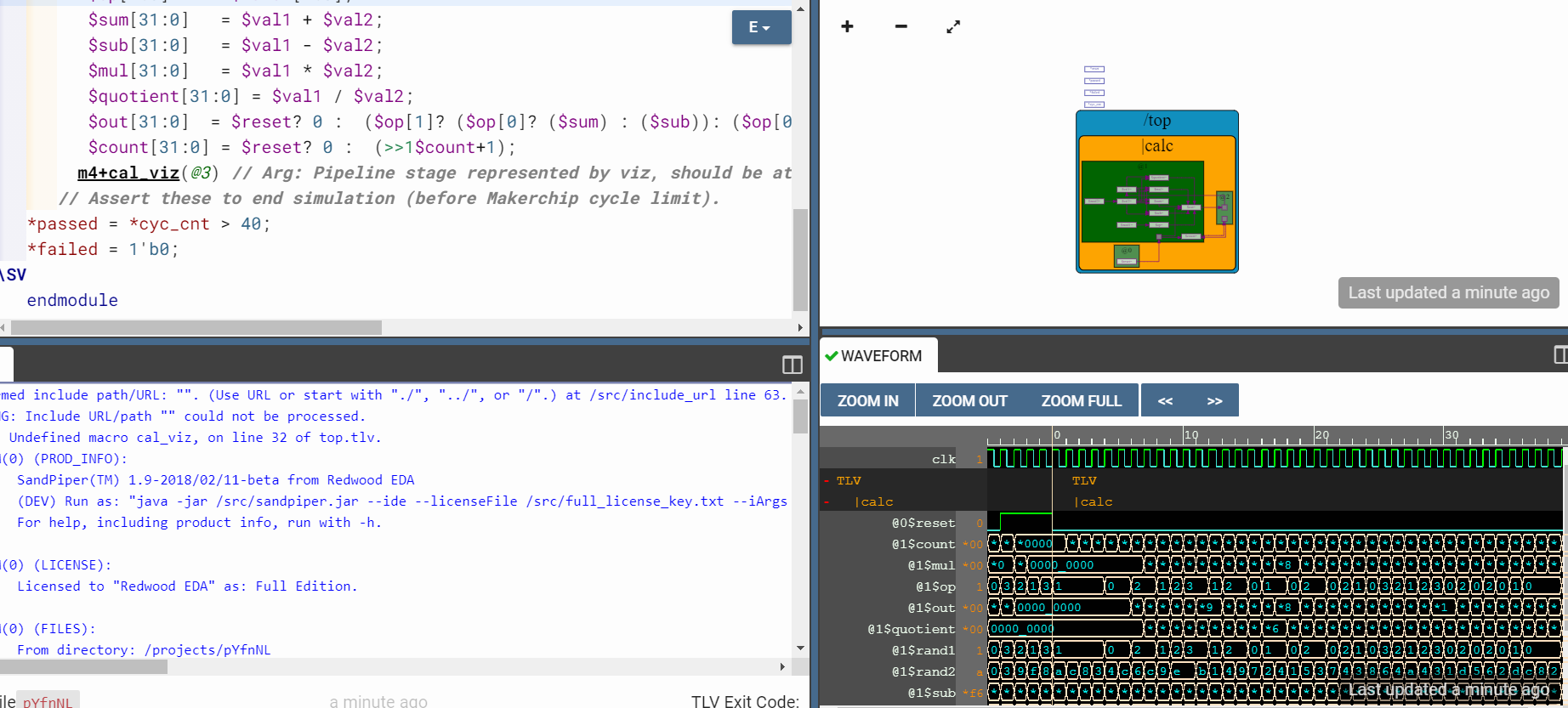
\*passed = \*cyc\_cnt > 40;

\*failed = 1'b0;

\SV

endmodule

**Counter+calculatorwithsinglecyclepipeline:**



\m4\_TLV\_version 1d: tl-x.org

\SV

// =========================================

// Welcome! Try the tutorials via the menu.

// =========================================

// Default Makerchip TL-Verilog Code Template

m4\_include\_lib //(['https://raw.githubusercontent.com/stevehoover/RISC-V\_MYTH\_Workshop/bd1f186fde018ff9e3fd80597b7397a1c862cf15/tlv\_lib/calculator\_shell\_lib.tlv'])

\SV

// Macro providing required top-level module definition, random

// stimulus support, and Verilator config.

m4\_makerchip\_module // (Expanded in Nav-TLV pane.)

//m4+cal\_viz(@3)

\TLV

|calc

@0

$reset = \*reset;

@1

$val1[31:0] = >>1$out;

// $val1[31:0] = $rand1[3:0];

$val2[31:0] = $rand2[3:0];

$op[1:0] = $rand1[1:0];

$sum[31:0] = $val1 + $val2;

$sub[31:0] = $val1 - $val2;

$mul[31:0] = $val1 \* $val2;

$quotient[31:0] = $val1 / $val2;

$out[31:0] = $reset? 0 : ($op[1]? ($op[0]? ($sum) : ($sub)): ($op[0]? ($mul) : ($quotient)));

$count[31:0] = $reset? 0 : (>>1$count+1);

m4+cal\_viz(@3) // Arg: Pipeline stage represented by viz, should be atleast equal to last stage of CALCULATOR logic.

// Assert these to end simulation (before Makerchip cycle limit).

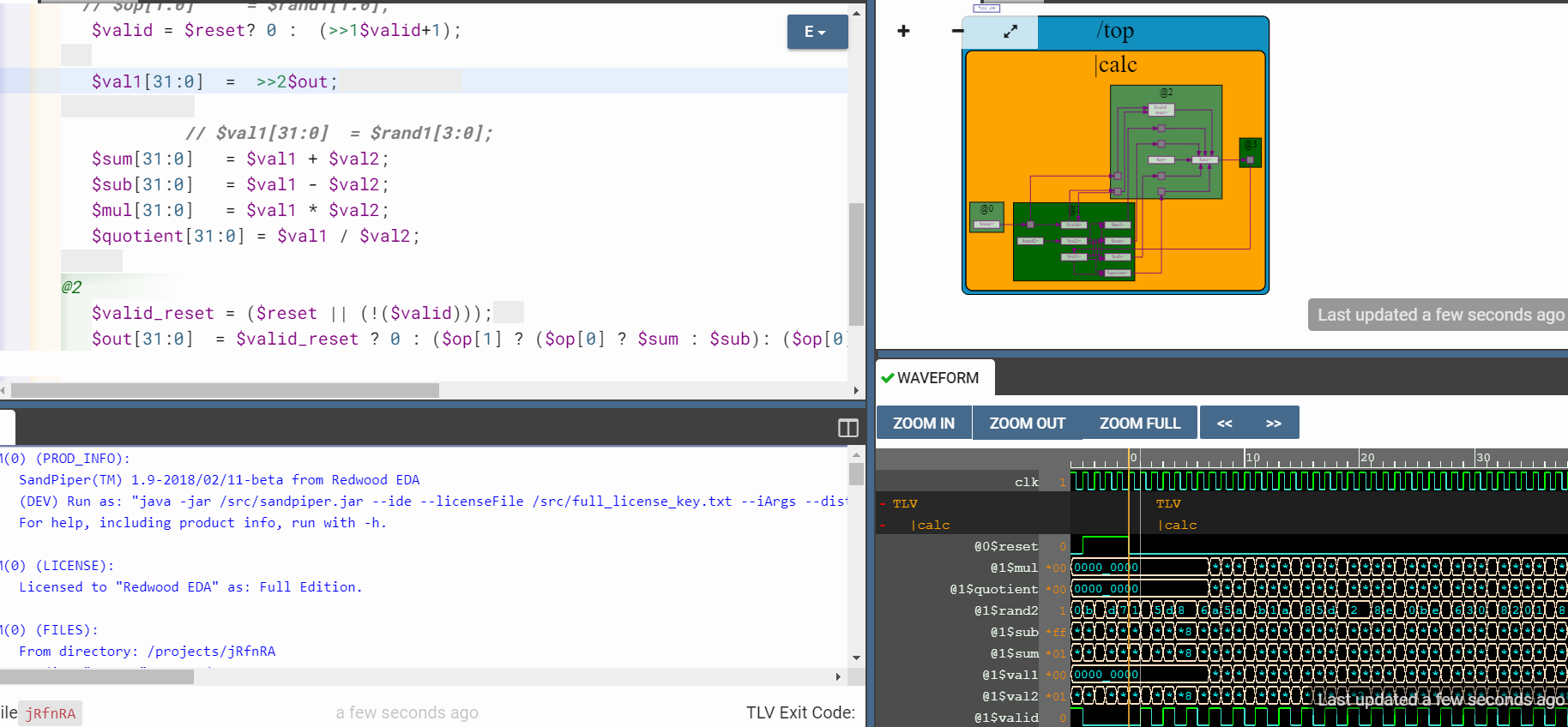
\*passed = \*cyc\_cnt > 40;

\*failed = 1'b0;

\SV

endmodule

**2:calculator with double cycle pipeline**



Code:

\m4\_TLV\_version 1d: tl-x.org

\SV

// =========================================

// Welcome! Try the tutorials via the menu.

// =========================================

// Default Makerchip TL-Verilog Code Template

m4\_include\_lib(['https://raw.githubusercontent.com/stevehoover/RISC-V\_MYTH\_Workshop/bd1f186fde018ff9e3fd80597b7397a1c862cf15/tlv\_lib/calculator\_shell\_lib.tlv'])

\SV

// Macro providing required top-level module definition, random

// stimulus support, and Verilator config.

m4\_makerchip\_module // (Expanded in Nav-TLV pane.)

//m4+cal\_viz(@3)

\TLV

|calc

@0

$reset = \*reset;

@1

$val2[31:0] = $rand2[3:0];

// $op[1:0] = $rand1[1:0];

$valid = $reset? 0 : (>>1$valid+1);

$val1[31:0] = >>2$out;

// $val1[31:0] = $rand1[3:0];

$sum[31:0] = $val1 + $val2;

$sub[31:0] = $val1 - $val2;

$mul[31:0] = $val1 \* $val2;

$quotient[31:0] = $val1 / $val2;

@2

$valid\_reset = ($reset || (!($valid)));

$out[31:0] = $valid\_reset ? 0 : ($op[1] ? ($op[0] ? $sum : $sub): ($op[0] ? $mul : $quotient ));

// m4+cal\_viz(@2) // Arg: Pipeline stage represented by viz, should be atleast equal to last stage of CALCULATOR logic.

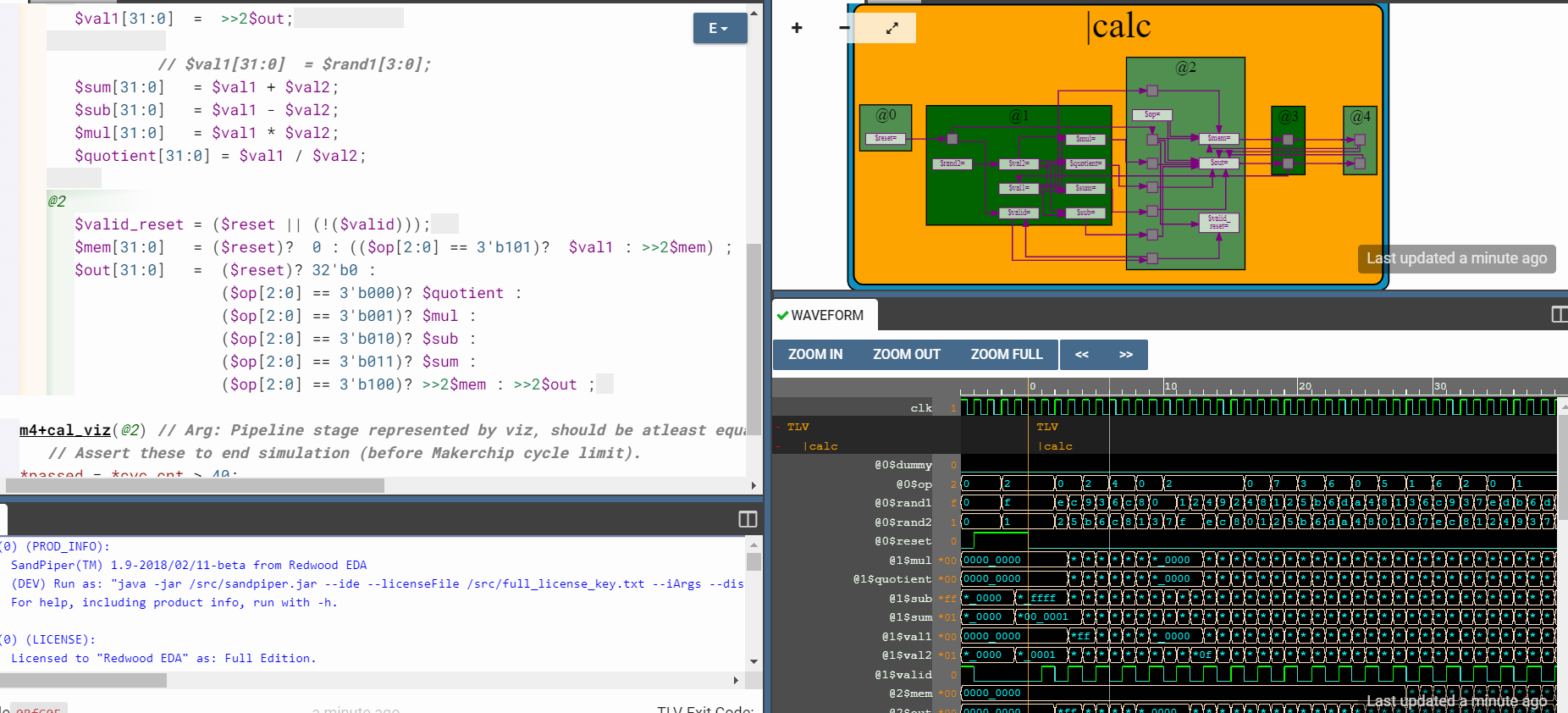
// Assert these to end simulation (before Makerchip cycle limit).

\*passed = \*cyc\_cnt > 40;

\*failed = 1'b0;

\SV

endmodule

**Calculator with single value memory:**

**Code**:

\m4\_TLV\_version 1d: tl-x.org

\SV

// =========================================

// Welcome! Try the tutorials via the menu.

// =========================================

// Default Makerchip TL-Verilog Code Template

m4\_include\_lib(['https://raw.githubusercontent.com/stevehoover/RISC-V\_MYTH\_Workshop/bd1f186fde018ff9e3fd80597b7397a1c862cf15/tlv\_lib/calculator\_shell\_lib.tlv'])

\SV

// Macro providing required top-level module definition, random

// stimulus support, and Verilator config.

m4\_makerchip\_module // (Expanded in Nav-TLV pane.)

//m4+cal\_viz(@3)

\TLV

|calc

@0

$reset = \*reset;

@1

$val2[31:0] = $rand2[3:0];

// $op[1:0] = $rand1[1:0];

$valid = ($reset)? 0 : (>>1$valid+1);

$val1[31:0] = >>2$out;

// $val1[31:0] = $rand1[3:0];

$sum[31:0] = $val1 + $val2;

$sub[31:0] = $val1 - $val2;

$mul[31:0] = $val1 \* $val2;

$quotient[31:0] = $val1 / $val2;

@2

$valid\_reset = ($reset || (!($valid)));

$mem[31:0] = ($reset)? 0 : (($op[2:0] == 3'b101)? $val1 : >>2$mem) ;

$out[31:0] = ($reset)? 32'b0 :

($op[2:0] == 3'b000)? $quotient :

($op[2:0] == 3'b001)? $mul :

($op[2:0] == 3'b010)? $sub :

($op[2:0] == 3'b011)? $sum :

($op[2:0] == 3'b100)? >>2$mem : >>2$out ;

m4+cal\_viz(@2) // Arg: Pipeline stage represented by viz, should be atleast equal to last stage of CALCULATOR logic.

// Assert these to end simulation (before Makerchip cycle limit).

\*passed = \*cyc\_cnt > 40;

\*failed = 1'b0;

\SV

endmodule