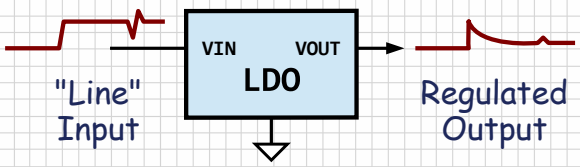
A UVM SystemVerilog Testbench for Directed & Random Testing of an AMS LDO Voltage Regulator

Charles Dančak (charles@betasoft.org)
Betasoft Consulting, Santa Clara, California

2024 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES
SAN JOSE, CA, USA
MARCH 4-7, 2024

Verification Goals

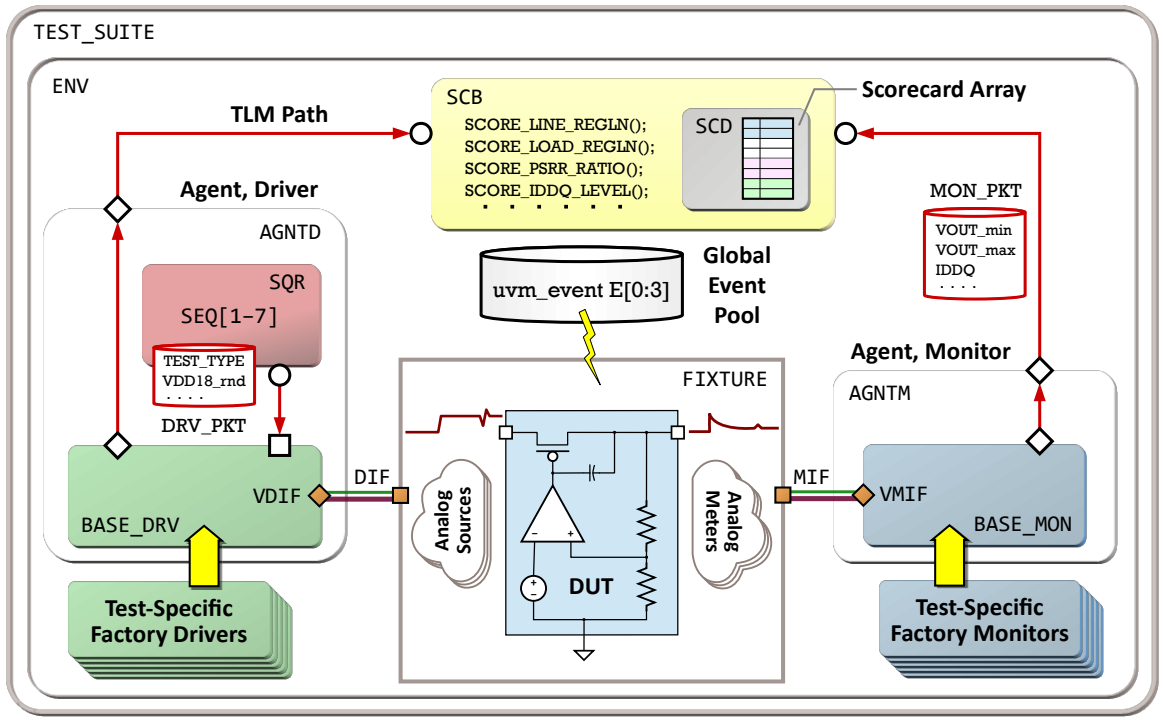
- A UVM testbench for an AMS DUT.
- Run 8 benchtop-style directed tests to verify that voltage is regulated.
- Randomize a sequence of such tests to reach unanticipated corner cases.



Coding Tactics

- Factory drivers/monitors to isolate UVM classes from testing details.
- UVM events for testbench timing.
- Use `randsequence` to generate test sequence, with no random instability.

UVM Testbench for Low-Dropout Voltage Regulator




```
APPLY_LOAD_REGLN_tf():
  «Apply new stimulus.»
//Allow DUT to settle:
  #tSETTLE;
//Trigger pool event:
  E[3].trigger();
```

Global Event `E[3]`

```
CHECK_LOAD_REGLN_tf():
//Await pool event:
  E[3].wait_trigger();
//Sample measured VOUT:
  VOUT = VMIF.A.VOUT;
```

FIXTURE

```
HOPPER: H
//Set up event pool.
```

VDIF

VMIF

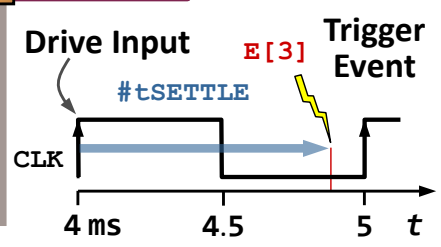
Interface Bus (Hierarchical)

Drive Test Type → `LOAD_REGLN`

```
always @(
  «Detect LOAD_REGLN type.»
) begin:LOAD_BLK
  //Wait for E[3]:
    H.E[3].wait_trigger();
    «Activate measurement.»
  end:  LOAD_BLK
```

Event Array `E[0:3]` for Timing

Fixture with Hopper



Voltage Regulator Test Types

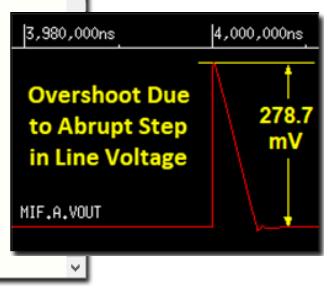
Code	Test Type	Analog Quantity Measured
4'h0	IDLE_SUITE	Default enumerated test type.
4'h1	LINE_REGLN	$\Delta VOUT$, as $VDD18$ varies from min to max over input range.
4'h2	LOAD_REGLN	$\Delta VOUT$, as $ILOAD$ varies from max to min over load range.
4'h3	PSRR_RATIO	Rejection of 10-kHz input hum.
4'h4	IDDQ_LEVEL	Quiescent $IDDQ$ under low load.
4'h5	LINE_TRANS	$VOUT$ overshoot, undershoot.
4'h6	LOAD_TRANS	$VOUT$ undershoot, overshoot.
4'h7	TRIM_LEVEL	Trimmed $VOUT$ levels [n = 1–16].
4'h8	CTRL_MODES	$VOUT$ for ENA, RET, BYP modes.
4'h9	RAND_TRANS	Run LINE_TRANS, LOAD_TRANS tests in a random sequence.

```
function new(...);  //RAND_TRANS constructor.
  .   .   .   .   .
//Generate random ORDER of test types:
  for (int I = 1; I <= TRIALS; I++)
  begin:RS_LOOP
    randsequence(TRANSIENT)
      TRANSIENT:  RS_LINE_TRANS := 7
                | RS_LOAD_TRANS := 3;
      RS_LINE_TRANS: {ORDER[I] = LINE_TRANS;};
      RS_LOAD_TRANS: {ORDER[I] = LOAD_TRANS;};
    endsequence
  end:  RS_LOOP
endfunction: new
```

Generate Test Sequence with Random Stability

```
~/LDO/150
524 UVM_INFO UVM/CDNS-1.2/sv/src/base/uvm_objection.svh
525 @ 21.950 ms: reporter [TEST_DONE] proceed to 'extract'
526
527   <Scorecard (SCD) for RAND_TRANS Test Suite >
528   ------------------------------------------------
529     1:  LINE_TRANS:  15.67839 mV        PASS
530     2:  LINE_TRANS: 278.73296 mV        FAIL
531     3:  LOAD_TRANS:  25.41294 mV        PASS
532     4:  LOAD_TRANS:  57.61085 mV        PASS
533     5:  LINE_TRANS:  79.06746 mV        PASS
534     6:  LOAD_TRANS:  46.00126 mV        PASS
535     7:  LINE_TRANS: 164.67493 mV        FAIL
536     8:  LINE_TRANS: 335.82871 mV        FAIL
537     9:  LOAD_TRANS:  19.11920 mV        PASS
538    10:  LINE_TRANS: 105.49742 mV        FAIL
539
540         RAND_TRANS Test Failures:        4
541         RAND_TRANS Test TRIALS:         10
542
"RAND_TRANS.log" 581L, 17608B written
```

Simulated `LINE_TRANS` Overshoot



ACKNOWLEDGMENTS

Thanks to Jaeha Kim of Scientific Analog for valuable feedback, and for providing the EDA tools. And to Rafael Betancourt for the LDO design. The author is also a SystemVerilog instructor with UC San Diego, Dept. of Extended Studies, La Jolla CA.