netlist-paths: a tool for querying paths in a Verilog design

Jamie Hanlon

Graphcore

September 28, 2019

Background and motivation

- Synthesized logic looses much of its correspondence with the RTL due to flattening and optimisation.
- It is then difficult to correspond timing paths back to the source code.
- Being able to do this is important: to quickly identify where fixes are required and to avoid rebuilding a design with a speculative fix.

Example report (simplified):

Point	ref_name	Net
cts_inv_10478166254/CK->X	CKINV_CB	ctsbuf_net_839142901
cts_inv_10473166249/CK->X	CKINV	ctsbuf_net_837142899
cto_inv_169431/CK->X	CKINV_CB	cts482
cto_inv_169430/CK->X	CKINV	cts481
u_dbg_tdi_u_cbus_tdi_target_power_cg_19_0_0_latch/CK->Q	CKGTPLT_V7Y2	n47487
power_cg_269_latch/CK->Q	CKGTPLT_CBV7Y2	n47575
ctosc_drc_inst_206015/CK->X	CKINV	ctosc_drc_75
ctosc_drc_inst_206014/CK->X	CKINV	ctosc_drc_74
u_dbg_tdi_u_cbus_tdi_target_regs_reg/CK->Q	FSDPRBQ_V2FY2	u_dbg_tdi_exec_instr_24_
u8550/A1->X	NR2	n13464
SGI94_133056/A2->X	EN2_V2Y2	n13474
u5417/A2->X	AOI22	n4550
SGI156_133051/B->X	OAI21_V1Y2	n13490
u4065/A1->X	NR2	n4088
u41596/A1->X	ND2_MM	n24109
u4961/A1->X	NR2	n5629
u10133/A1->X	ND2_MM	n27510
u10134/A1->X	ND2_MM	n24210
RLB_135261/B->X	ND2B_V1	n41234
u40381/A1->X	ND2_CB	n24235
u45746/A1->X	AOI21_V1Y2	n14358
SGI38_135033/A2->X	NR3	n47198
u10157/A1->X	ND3	n14314
u10161/A2->X	NR2_CB	n14319
SGI188_133595/A1->X	ND2_MM	n44980
u41887/A->X	INV_S	n43774
u13107/A1->X	NR2	n43005
u53593/A2->X	ND2_CB	n25248
...		
SGI104_172925/B->X	NR2B_V1	n39867
u_exu_u_exin_x2_sel_data0e_valid_q_reg/D	FSDPRBQ_V2FY2	

The tool

- Uses modified Verilator to produce a flattened netlist of a Verilog design.
- Provides ways to query paths (similar to icc2):
  - Between two points (registers or ports)
  - Fanning in/out to/from a point
  - Through particular other points
  - Matches names with regexes
- Reports paths with names, types and locations at each step.
- Written in C++ with `boost::graph`.

Example: NVDLA

An open accelerator core for deep learning inference.[1]

Paths fan-in paths to a register:

```
$ netlist-paths netlist.graph --to u_calculator.calc_op1_fp_46_d1
...
Path 1
dla_core_clk                                                    NV_nvdla.v:86
nvdla_core_clk                                                  NV_NVDLA_partition_a.v:102
u_NV_NVDLA_cacc.nvdla_core_clk                                  NV_NVDLA_cacc.v:55
u_NV_NVDLA_cacc.u_slcg_op_2.nvdla_core_clk                      NV_NVDLA_CACC_slcg.v:21
u_NV_NVDLA_cacc.u_slcg_op_2.nvdla_core_clk_slcg_0.clk           NV_CLK_gate_power.v:10
u_NV_NVDLA_cacc.u_slcg_op_2.nvdla_core_clk_slcg_0.p_clkgate.CP CKLNQD12.v:17
u_NV_NVDLA_cacc.u_slcg_op_2.nvdla_core_clk_slcg_0.p_clkgate.Q  CKLNQD12.v:18
u_NV_NVDLA_cacc.u_slcg_op_2.nvdla_core_clk_slcg_0.clk_gated    NV_CLK_gate_power.v:11
u_NV_NVDLA_cacc.u_slcg_op_2.nvdla_core_gated_clk                NV_NVDLA_CACC_slcg.v:26
u_NV_NVDLA_cacc.nvdla_op_gated_clk_2                            NV_NVDLA_cacc.v:166
u_NV_NVDLA_cacc.u_calculator.nvdla_core_clk                     NV_NVDLA_CACC_calculator.v:82
u_NV_NVDLA_cacc.u_calculator.calc_op1_fp_46_d1                  NV_NVDLA_CACC_calculator.v:2032
...
```

[1] https://github.com/nvdla/hw

Example: NVDLA

Fan-out paths from a register:

```
$ netlist-paths netlist.graph --reportlogic --from u_accu_dbuf_4.r_nv_ram_rws_32x512.mbist_en_flop.Q
...
Path 225
Path 225
mbist_en_flop.Q                                          REG_SRC p_SDFCNQD1PO4.v:13
ASSIGNW                                                  LOGIC   nv_ram_rws_32x512_logic.v:154
mbist_en_r                                               WIRE    nv_ram_rws_32x512_logic.v:152
ASSIGNW                                                  LOGIC   nv_ram_rws_32x512_logic.v:193
muxed_Di_w0_S                                            WIRE    nv_ram_rws_32x512_logic.v:193
ALWAYS                                                   LOGIC   nv_ram_rws_32x512_logic.v:194
ASSIGN                                                   LOGIC   nv_ram_rws_32x512_logic.v:196
muxed_Di_w0                                              VAR     nv_ram_rws_32x512_logic.v:189
ASSIGNW                                                  LOGIC   nv_ram_rws_32x512_logic.v:1585
muxed_Data_A                                             WIRE    nv_ram_rws_32x512_logic.v:1582
ALWAYS                                                   LOGIC   nv_ram_rws_32x512_logic.v:1587
ASSIGN                                                   LOGIC   nv_ram_rws_32x512_logic.v:1589
muxed_Data_r0                                            VAR     nv_ram_rws_32x512_logic.v:1581
ASSIGNW                                                  LOGIC   nv_ram_rws_32x512_logic.v:1660
Data_reg_r0_D                                            WIRE    nv_ram_rws_32x512_logic.v:1660
ASSIGNW                                                  LOGIC   nv_ram_rws_32x512_logic.v:1667
ASSIGNW                                                  LOGIC   nv_ram_rws_32x512_logic.v:1667
testInst_Data_reg_r0_511_256.D                           PORT    ScanShareSel_JTAG_reg_ext_cg.v:31
ASSIGNW                                                  LOGIC   ScanShareSel_JTAG_reg_ext_cg.v:46
testInst_Data_reg_r0_511_256.next_Q                      WIRE    ScanShareSel_JTAG_reg_ext_cg.v:34
ASSIGNW                                                  LOGIC   ScanShareSel_JTAG_reg_ext_cg.v:57
ASSIGNW                                                  LOGIC   ScanShareSel_JTAG_reg_ext_cg.v:57
testInst_Data_reg_r0_511_256.Jreg_ff[195].SSS.nr.D       PORT    SDFQD1.v:17
ASSIGNW                                                  LOGIC   SDFQD1.v:22
testInst_Data_reg_r0_511_256.Jreg_ff[195].SSS.nr.sel     WIRE    SDFQD1.v:22
ALWAYS                                                   LOGIC   SDFQD1.v:23
testInst_Data_reg_r0_511_256.Jreg_ff[195].SSS.nr.Q       REG_DST SDFQD1.v:20
...
```

Improvements and plans

- More options to specify paths: eg internal/external, avoiding points.
- Improve dataflow analysis in Verilator to link variables rather than blocks.
- Avoid startup cost of repeatedly loading large netlist files.
- Use to assert properties of the netlist structure: eg no paths between sub hierarchies X and Y.
- Upstream the Verilator changes.

Links

- Github: `https://github.com/jameshanlon/netlist-paths`
- More details: `https://www.jameswhanlon.com/querying-logical-paths-in-a-verilog-design.html`
- Get in touch: `jamie.hanlon@graphcore.ai`