Project 2 - FSM Synthesis

Synthesis of Project 1 FSM - credit card payment controller FSM

October 1, 2019

Arthur Hsueh
21582168
UBC - ELEC 402

Contents

List of Figures

1 FSM Adjustments

The FSM used for synthesis in Assignment 2 is the same one as Assignment 1, with a minor adjustment. In Assignment 1, the state parameters were 12 bits wide, which comprised of the state encoding (4 bits), a placeholder bit (1 bit), and some output bits (7 bits). To produce a usable FSM, I have removed the placeholder bit, so the states are now 11 bits wide. Other than that, the design of the FSM remains the same.

2 Mapped SystemVerilog

Below is the mapped SystemVerilog code generated by the Cadence from synthesis. For easier reading, the raw .sv file is also attached with this report

---

```
// Generated by Cadence Encounter(R) RTL Compiler RC14.13 - v14.10-s027_1

// Verification Directory fv/credit_card_payment_fsm

module credit_card_payment_fsm(clk, reset, process_init,
     visa_choice_in, mastercard_choice_in, amex_choice_in,
     pymt_amt_conf, pymt_amt_denied, pin_fail, pin_success,
     transaction_fail, transaction_success, light_bit, card_choice,
     pymt_amt_print, pin_process_init, pymt_process_init,
     process_abort);
  input clk, reset, process_init, visa_choice_in, mastercard_choice_in,
       amex_choice_in, pymt_amt_conf, pymt_amt_denied, pin_fail,
       pin_success, transaction_fail, transaction_success;
  output light_bit, pymt_amt_print, pin_process_init,
       pymt_process_init, process_abort;
  output [1:0] card_choice;
  wire clk, reset, process_init, visa_choice_in, mastercard_choice_in,
       amex_choice_in, pymt_amt_conf, pymt_amt_denied, pin_fail,
       pin_success, transaction_fail, transaction_success;
  wire light_bit, pymt_amt_print, pin_process_init, pymt_process_init,
       process_abort;
  wire [1:0] card_choice;
  wire [11:0] state;
  wire [1:0] fail_counter;
  wire n_0, n_1, n_2, n_3, n_4, n_5, n_6, n_7;
  wire n_8, n_9, n_10, n_11, n_12, n_13, n_14, n_15;
  wire n_16, n_17, n_18, n_19, n_20, n_21, n_22, n_23;
  wire n_24, n_25, n_26, n_27, n_28, n_29, n_30, n_31;
  wire n_32, n_33, n_34, n_35, n_36, n_37, n_38, n_39;
  wire n_40, n_41, n_42, n_43, n_44, n_45, n_46, n_47;
  wire n_48, n_49, n_50, n_51, n_52, n_53, n_54, n_55;
  wire n_56, n_57, n_58, n_59, n_60, n_61, n_62, n_63;
  wire n_64, n_65, n_66, n_67, n_68, n_69, n_70, n_71;
  wire n_72, n_73, n_74, n_75, n_76, n_77, n_78, n_79;
  wire n_80, n_81, n_82, n_83, n_84, n_85, n_86, n_87;
  wire n_88;
  DFFHQX1 \state_reg[6] (.CK (clk), .D (n_88), .Q (light_bit));
  DFFHQX1 \state_reg[8] (.CK (clk), .D (n_87), .Q (state[8]));
```

```
DFFHQX1 \state_reg[10] (.CK (clk), .D (n_86), .Q (state[10]));
NAND2X1 g2879(.A (n_71), .B (n_85), .Y (n_88));
NAND4XL g2880(.A (n_79), .B (n_47), .C (n_56), .D (n_81), .Y (n_87));
DFFHQX1 \state_reg[9] (.CK (clk), .D (n_83), .Q (state[9]));
OR2XL g2891(.A (n_84), .B (n_82), .Y (n_86));
DFFHQX1 \state_reg[0] (.CK (clk), .D (n_80), .Q (process_abort));
NOR4X1 g2882(.A (n_53), .B (n_48), .C (n_75), .D (n_84), .Y (n_85));
OR4X1 g2883(.A (n_37), .B (n_84), .C (n_63), .D (n_70), .Y (n_83));
OAI211X1 g2900(.A0 (n_2), .A1 (n_40), .B0 (n_77), .C0 (n_58), .Y
     (n_82));
NOR2X1 g2885(.A (n_78), .B (n_76), .Y (n_81));
DFFHQX1 \fail_counter_reg[0] (.CK (clk), .D (n_73), .Q
     (fail_counter[0]));
DFFHQX1 \state_reg[11] (.CK (clk), .D (n_72), .Q (state[11]));
OAI211X1 g2898(.A0 (n_0), .A1 (n_79), .B0 (n_67), .C0 (n_69), .Y
     (n_80));
DFFHQX1 \fail_counter_reg[1] (.CK (clk), .D (n_64), .Q
     (fail_counter[1]));
DFFHQX1 \state_reg[2] (.CK (clk), .D (n_66), .Q (pin_process_init));
OAI31X1 g2890(.A0 (pin_fail), .A1 (pin_success), .A2 (n_65), .B0
     (n_74), .Y (n_78));
INVX1 g2905(.A (n_76), .Y (n_77));
DFFHQX1 \state_reg[4] (.CK (clk), .D (n_57), .Q (card_choice[0]));
OAI31X1 g2892(.A0 (n_28), .A1 (n_62), .A2 (n_61), .B0 (n_74), .Y
     (n_75));
NOR2BX1 g2894(.AN (n_60), .B (reset), .Y (n_73));
NAND3BXL g2897(.AN (n_55), .B (n_71), .C (n_54), .Y (n_72));
DFFHQX1 \state_reg[5] (.CK (clk), .D (n_59), .Q (card_choice[1]));
OAI22X1 g2899(.A0 (n_49), .A1 (n_43), .B0 (pymt_amt_denied), .B1
     (n_68), .Y (n_70));
NAND2BX1 g2903(.AN (n_68), .B (pymt_amt_denied), .Y (n_69));
NAND2X1 g2906(.A (n_67), .B (n_68), .Y (n_76));
OAI21X1 g2910(.A0 (n_35), .A1 (n_65), .B0 (n_50), .Y (n_66));
NOR2X1 g2895(.A (reset), .B (n_52), .Y (n_64));
OAI32X1 g2907(.A0 (n_30), .A1 (n_62), .A2 (n_61), .B0
     (transaction_fail), .B1 (n_79), .Y (n_63));
OAI31X1 g2908(.A0 (n_51), .A1 (fail_counter[0]), .A2 (n_34), .B0
     (n_45), .Y (n_60));
INVX1 g2917(.A (n_58), .Y (n_59));
INVX1 g2919(.A (n_56), .Y (n_57));
NOR2X1 g2896(.A (n_55), .B (n_46), .Y (n_74));
OAI21X1 g2921(.A0 (pymt_amt_denied), .A1 (pymt_amt_conf), .B0 (n_53),
     .Y (n_54));
MXI2XL g2911(.A (n_33), .B (fail_counter[1]), .S0 (n_51), .Y (n_52));
NAND2X1 g2913(.A (pymt_amt_conf), .B (n_53), .Y (n_50));
NAND2BX1 g2914(.AN (pymt_amt_conf), .B (n_53), .Y (n_68));
NAND2X1 g2918(.A (n_49), .B (n_48), .Y (n_58));
OAI21X1 g2920(.A0 (visa_choice_in), .A1 (n_1), .B0 (n_48), .Y (n_56));
DFFHQX1 \state_reg[1] (.CK (clk), .D (n_39), .Q (pymt_process_init));
DFFHQX1 \state_reg[3] (.CK (clk), .D (n_42), .Q (pymt_amt_print));
NAND2X1 g2904(.A (n_41), .B (n_47), .Y (n_84));
NOR4X1 g2909(.A (reset), .B (n_32), .C (state[8]), .D (n_61), .Y
     (n_46));
NAND2X1 g2915(.A (n_51), .B (fail_counter[0]), .Y (n_45));
OAI32X1 g2922(.A0 (reset), .A1 (n_44), .A2 (n_17), .B0 (n_36), .B1
     (n_44), .Y (n_55));
INVX1 g2927(.A (n_48), .Y (n_43));
INVX1 g2933(.A (n_41), .Y (n_42));
OR2X1 g2924(.A (transaction_success), .B (n_40), .Y (n_79));
NOR2X1 g2925(.A (n_29), .B (n_65), .Y (n_39));
NAND2BX1 g2926(.AN (n_65), .B (n_6), .Y (n_67));
NOR3X1 g2928(.A (n_38), .B (state[8]), .C (n_61), .Y (n_48));
NOR3X1 g2929(.A (n_38), .B (n_25), .C (n_26), .Y (n_53));
NOR2X1 g2931(.A (n_36), .B (n_44), .Y (n_37));
OAI2BB1X1 g2934(.A0N (n_11), .A1N (n_22), .B0 (n_31), .Y (n_41));
NAND3BXL g2916(.AN (state[11]), .B (n_27), .C (pymt_amt_print), .Y
     (n_47));
AOI2BB1X1 g2930(.A0N (n_35), .A1N (n_34), .B0 (n_24), .Y (n_51));
NAND2BX1 g2932(.AN (n_71), .B (state[9]), .Y (n_40));
OR2X1 g2935(.A (n_71), .B (state[9]), .Y (n_65));
NOR2X1 g2936(.A (n_5), .B (n_34), .Y (n_33));
NAND3BXL g2938(.AN (pymt_amt_print), .B (n_21), .C
     (pymt_process_init), .Y (n_36));
OR4X1 g2939(.A (n_19), .B (process_abort), .C (light_bit), .D
     (state[11]), .Y (n_32));
NAND2X1 g2940(.A (n_31), .B (state[9]), .Y (n_38));
AOI2BB1X1 g2941(.A0N (n_29), .A1N (n_28), .B0 (n_31), .Y (n_30));
NOR3X1 g2937(.A (n_20), .B (pymt_process_init), .C (n_26), .Y (n_27));
OR3XL g2942(.A (n_28), .B (n_25), .C (n_44), .Y (n_71));
NOR4X1 g2943(.A (n_14), .B (n_26), .C (n_13), .D (n_23), .Y (n_24));
OR4X1 g2944(.A (n_10), .B (n_25), .C (n_23), .D (n_44), .Y (n_34));
NOR2X1 g2945(.A (n_28), .B (state[11]), .Y (n_31));
NAND3X1 g2948(.A (state[10]), .B (card_choice[1]), .C (n_16), .Y
     (n_22));
INVX1 g2946(.A (n_20), .Y (n_21));
NAND2BX1 g2950(.AN (n_23), .B (process_init), .Y (n_19));
NAND3X1 g2947(.A (n_18), .B (state[9]), .C (n_25), .Y (n_20));
NAND2X1 g2951(.A (n_18), .B (n_9), .Y (n_28));
NAND3BXL g2952(.AN (n_15), .B (pin_process_init), .C (n_12), .Y
     (n_17));
OAI32X1 g2959(.A0 (card_choice[0]), .A1 (state[9]), .A2 (state[8]),
     .B0 (n_8), .B1 (n_62), .Y (n_16));
NAND3BXL g2953(.AN (pin_success), .B (pin_fail), .C (n_3), .Y (n_35));
OR2XL g2955(.A (pin_process_init), .B (n_15), .Y (n_23));
OR2X1 g2956(.A (n_14), .B (n_61), .Y (n_44));
AOI31X1 g2957(.A0 (light_bit), .A1 (process_abort), .A2 (state[8]),
     .B0 (n_12), .Y (n_13));
NAND4XL g2954(.A (card_choice[0]), .B (state[9]), .C (state[8]), .D
     (n_7), .Y (n_11));
NOR3X1 g2958(.A (reset), .B (pin_process_init), .C (n_10), .Y (n_18));
NAND2X1 g2961(.A (n_4), .B (n_9), .Y (n_15));
NOR2X1 g2964(.A (state[8]), .B (n_10), .Y (n_12));
NAND3BXL g2965(.AN (card_choice[1]), .B (state[10]), .C (n_8), .Y
     (n_26));
NAND2X1 g2962(.A (n_8), .B (n_7), .Y (n_61));
NAND2BXL g2949(.AN (n_6), .B (pin_success), .Y (n_29));
XNOR2X1 g2966(.A (fail_counter[0]), .B (fail_counter[1]), .Y (n_5));
NAND2X1 g2972(.A (n_4), .B (state[8]), .Y (n_62));
AOI2BB1XL g2963(.A0N (amex_choice_in), .A1N (mastercard_choice_in),
     .B0 (visa_choice_in), .Y (n_49));
```

```
  INVX1 g2960(.A (n_6), .Y (n_3));
  NAND2BX1 g2968(.AN (process_abort), .B (light_bit), .Y (n_10));
  NOR2X1 g2970(.A (state[10]), .B (card_choice[1]), .Y (n_7));
  NOR2XL g2971(.A (transaction_success), .B (transaction_fail), .Y
       (n_2));
  NOR2BX1 g2967(.AN (amex_choice_in), .B (mastercard_choice_in), .Y
       (n_1));
  NOR2X1 g2969(.A (pymt_process_init), .B (pymt_amt_print), .Y (n_9));
  AND2X1 g2973(.A (fail_counter[1]), .B (fail_counter[0]), .Y (n_6));
  INVX1 g2976(.A (card_choice[0]), .Y (n_8));
  INVX1 g2977(.A (state[11]), .Y (n_14));
  INVX1 g2975(.A (state[8]), .Y (n_25));
  INVX1 g2978(.A (state[9]), .Y (n_4));
  INVXL g2974(.A (transaction_fail), .Y (n_0));
endmodule
```

3 Waveform Verification

The .sdf file generated by Cadence after synthesis is used to validate the synthesized design with the original 'higher' level design. To produce a comparable set of information, I used the same testbench as I did in Project 1, credit_card_payment_fsm_tb.sv to test with the new synthesized module.

The testbench file has been modified to use the synthesized module and the .sdf file is applied to the simulation. The resulting waveform is shown below.

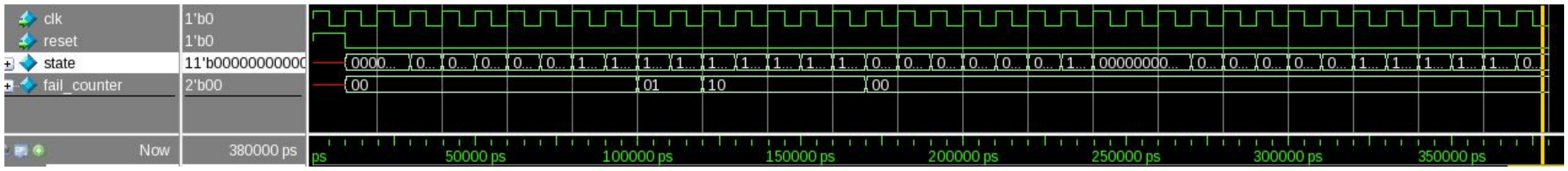

Figure 1: The simulated waveform tested with credit_card_payment_fsm_map.v

Although not 100% discernible because of the width of the state variable, the state is definitely switching. For easier viewing, the image is also attached to this report.

4 Cell Count Report

Below is the report generated by Cadence after synthesis, in the _area.rpt file. The produced file has also been attached with this report.

```
============================================================
  Generated by:         Encounter(R) RTL Compiler RC14.13 - v14.10-s027_1
  Generated on:         Sep 30 2019 06:00:27 pm
  Module:               credit_card_payment_fsm
  Technology library:   slow_vdd1v0 1.0
  Operating conditions: PVT_0P9V_125C (balanced_tree)
  Wireload mode:        enclosed
  Area mode:            timing library
============================================================

       Instance       Cells  Cell Area  Net Area  Total Area  Wireload
-------------------------------------------------------------------------------
credit_card_payment_fsm  110       208         0         208    <none> (D)

 (D) = wireload is default in technology library
```

The report shows that my FSM design uses 110 cells, which exceeds the requirement of 100 cells.