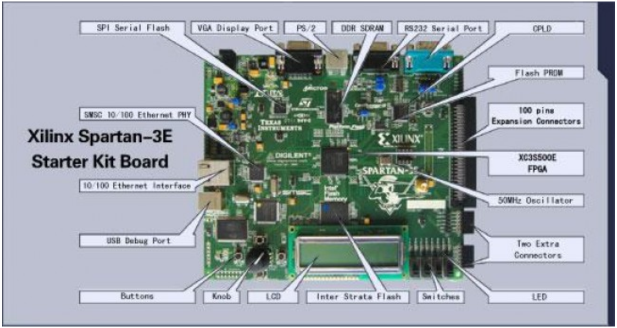
CS122A: Intermediate Embedded and Real Time Operating Systems

Jeffrey McDaniel

University of California, Riverside

FPGA - Introduction



What is an FPGA?

- Field Programmable Gate Array
- Re-programmable hardware used generally for prototyping or hardware accelerators

Programming an FPGA

- Programmed using VHDL

Programming an FPGA

- Programmed using VHDL
    - VHSIC HDL

Programming an FPGA

- Programmed using VHDL
    - VHSIC HDL
    - Very High Speed Integrated Circuit Hardware Design Language

Programming an FPGA

- Programmed using VHDL
  - VHSIC HDL
  - Very High Speed Integrated Circuit Hardware Design Language
- Difficult to program

Programming an FPGA

- Programmed using VHDL
    - VHSIC HDL
    - Very High Speed Integrated Circuit Hardware Design Language
- Difficult to program
- Designing a circuit (parallel) not a sequential program

Programming an FPGA

- Programmed using VHDL
  - VHSIC HDL
  - Very High Speed Integrated Circuit Hardware Design Language
- Difficult to program
- Designing a circuit (parallel) not a sequential program
- However, straightforward method of converting SM to VHDL, similar to SM to C

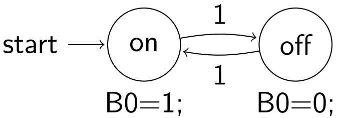
VHDL - Basics

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY MainSystem IS
  PORT(
    A_i: IN std_logic_vector(7 DOWNTO 0);
    B_o: OUT std_logic_vector(7 DOWNTO 0);
    clk_i: IN std_logic;
    rst_i: IN std_logic
  );
END MainSystem;
```

- A new system is declared as an *entity*
- The inputs and output signals are defined
- The example has 8 inputs (A_i) and 8 outputs (B_o)

SynchSM to VHDL Example

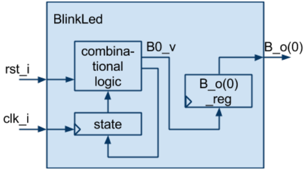


```
ARCHITECTURE blink OF MainSystem IS
BEGIN

BlinkLed: PROCESS(clk_i)
  TYPE states IS (LedOff,LedOn);
  VARIABLE state: states := LedOff;
  VARIABLE B0_v: std_logic := '0';
BEGIN
  IF (clk_i='1' AND clk_i'EVENT) THEN
    IF (rst_i='1') THEN
      state := LedOff;
      B0_v := '0';
    ELSE -- SM case stmts go here
      CASE state IS -- Transitions
        WHEN LedOff =>
          state := LedOn;
        WHEN LedOn =>
          state := LedOff;
        WHEN OTHERS =>
          state := LedOff;
      END CASE;
      CASE state IS -- Actions
                WHEN LedOff =>
          B0_v := '0';
        WHEN LedOn =>
          B0_v := '1';
      END CASE;
    END IF;
    B_o(0) <= B0_v; -- update port
  END IF;
END PROCESS;

END blink;
```

SynchSM to VHDL Example

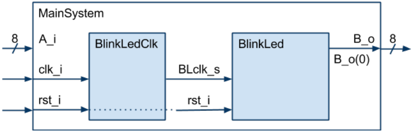


```
ARCHITECTURE blink OF MainSystem IS
BEGIN

BlinkLed: PROCESS(clk_i)
  TYPE states IS (LedOff,LedOn);
  VARIABLE state: states := LedOff;
  VARIABLE B0_v: std_logic := '0';
BEGIN
  IF (clk_i='1' AND clk_i'EVENT) THEN
    IF (rst_i='1') THEN
      state := LedOff;
      B0_v := '0';
    ELSE -- SM case stmts go here
      CASE state IS -- Transitions
        WHEN LedOff =>
          state := LedOn;
        WHEN LedOn =>
          state := LedOff;
        WHEN OTHERS =>
          state := LedOff;
      END CASE;
      CASE state IS -- Actions
              WHEN LedOff =>
          B0_v := '0';
        WHEN LedOn =>
          B0_v := '1';
      END CASE;
    END IF;
    B_o(0) <= B0_v; -- update port
  END IF;
END PROCESS;

END blink;
```

Tick Rate

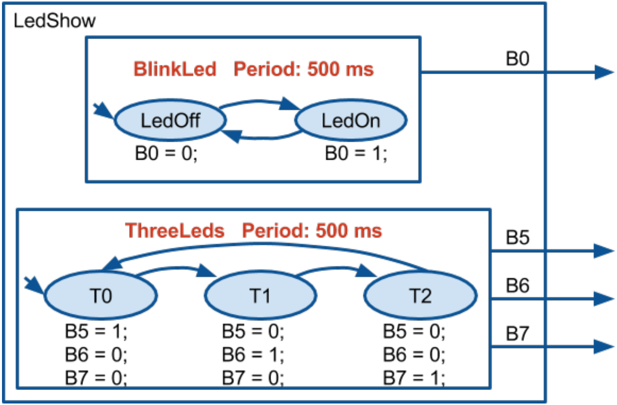


```
ARCHITECTURE blink OF MainSystem IS
  SIGNAL BLclk_s: std_logic;
BEGIN

BlinkLed: PROCESS(BLclk_s)
  ...
  IF (BLclk_s='1' AND BLclk_s'EVENT) THEN
  ...
END PROCESS;

BlinkLedClk: PROCESS(clk_i) -- Suppose 1 MHz
  VARIABLE cnt: INTEGER := 0;
BEGIN
  -- 1 MHz means 1 microsecond (us) per tick
  -- 500 ms * 1000us/ms = 500,000us
  IF (clk_i='1' AND clk_i'EVENT) THEN
    IF (rst_i='1') THEN
      cnt := 0;
    ELSE
      cnt := cnt + 1;
      IF (cnt = 500000) THEN
        BLclk_s <= '1';
        cnt := 0;
      ELSE
        BLclk_s <= '0';
      END IF;
    END IF;
  END IF;
END PROCESS;
END blink;
```

Multiple Synch SM's



Multiple Synch SM’s

```
ARCHITECTURE blink OF MainSystem IS
BEGIN

BlinkLed: PROCESS(clk_i)
  TYPE states IS (LedOff,LedOn);
  VARIABLE state: states := LedOff;
  VARIABLE B0_v: std_logic := ’0’;
BEGIN
  IF (clk_i=’1’ AND clk_i’EVENT) THEN
    IF (rst_i=’1’) THEN
      state := LedOff;
      B0_v := ’0’;
    ELSE -- SM case stmts go here
      CASE state IS -- Transitions
        WHEN LedOff =>
          state := LedOn;
        WHEN LedOn =>
          state := LedOff;
        WHEN OTHERS =>
          state := LedOff;
      END CASE;
      CASE state IS -- Actions
              WHEN LedOff =>
          B0_v := ’0’;
        WHEN LedOn =>
          B0_v := ’1’;
      END CASE;
    END IF;
    B_o(0) <= B0_v; -- update port
  END IF;
END PROCESS;

END blink;
```

Multiple Synch SM's

```
ARCHITECTURE blink OF MainSystem IS
  SIGNAL BLclk_s: std_logic;
BEGIN

BlinkLed: PROCESS(BLclk_s)
  ...
  IF (BLclk_s='1' AND BLclk_s'EVENT) THEN
  ...
END PROCESS;

BlinkLedClk: PROCESS(clk_i) -- Suppose 1 MHz
  VARIABLE cnt: INTEGER := 0;
BEGIN
  -- 1 MHz means 1 microsecond (us) per tick
  -- 500 ms * 1000us/ms = 500,000us
  IF (clk_i='1' AND clk_i'EVENT) THEN
    IF (rst_i='1') THEN
      cnt := 0;
    ELSE
      cnt := cnt + 1;
      IF (cnt = 500000) THEN
        BLclk_s <= '1';
        cnt := 0;
      ELSE
        BLclk_s <= '0';
      END IF;
    END IF;
  END IF;
END PROCESS;
END blink;
```

Multiple Synch SM's

```
ARCHITECTURE ThreeLeds OF MainSystem IS
  SIGNAL TLclk_s: std_logic;
BEGIN

ThreeLeds: PROCESS(TL2clk_s)
  TYPE states IS (T0, T1, T2);
  VARIABLE state : states := T0;

  VARIABLE B_v : std_logic_vector(2 DOWNTO 0);

BEGIN
  IF (TLclk_s='1' AND TLclk_s'EVENT) THEN
    IF (rst_i='1') THEN
      state := T0;
    ELSE
      CASE state IS -- Transitions
        WHEN T0 =>
          state := T1;
        WHEN T1 =>
          state := T2;
        WHEN T2 =>
          state := T0;
        WHEN OTHERS =>
          state := T0;
      END CASE;
      CASE state IS -- Transitions
        WHEN T0 =>
          B_v <= "001";
        WHEN T1 =>
          B_v <= "010";
        WHEN T2 =>
          B_v <= "100";
      END CASE;
    END IF;
    B_o(7 DOWNTO 5) <= B_v;
  END IF;
END PROCESS;

END ThreeLeds;
```

Multiple Synch SM's

```
ARCHITECTURE lightShow OF MainSystem IS
  SIGNAL BLclk_s,TLclk_s: std_logic;
BEGIN

-----

BlinkLed: PROCESS(BLclk_s)
  ...
END PROCESS;

BlinkLedClk: PROCESS(clk_i) -- gen 300 ms BLclk_s
  ...
END PROCESS;

----

ThreeLeds: PROCESS(TLclk_s)
  ...
END PROCESS;

ThreeLedsClk: PROCESS(clk_i) -- gen 200 ms TLclk_s
  ...
END PROCESS;

END lightShow;
```