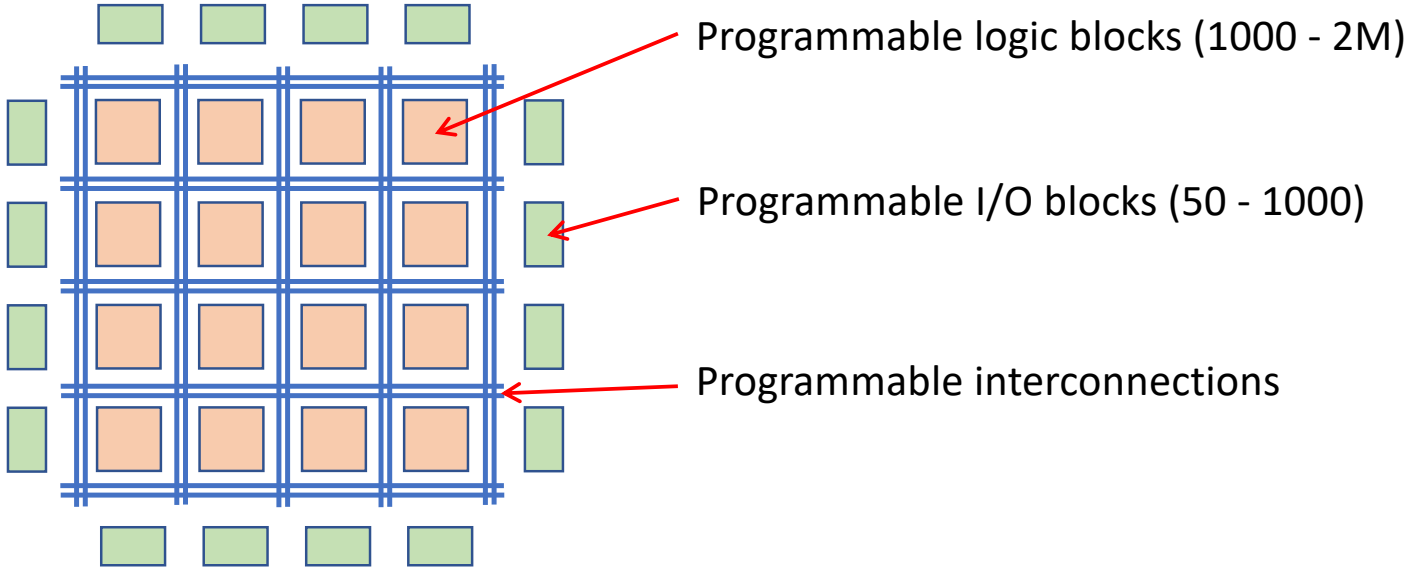
Getting Started with FPGAs

Jon Brumfitt
ESAC Electronics Club
19 September 2018

Overview

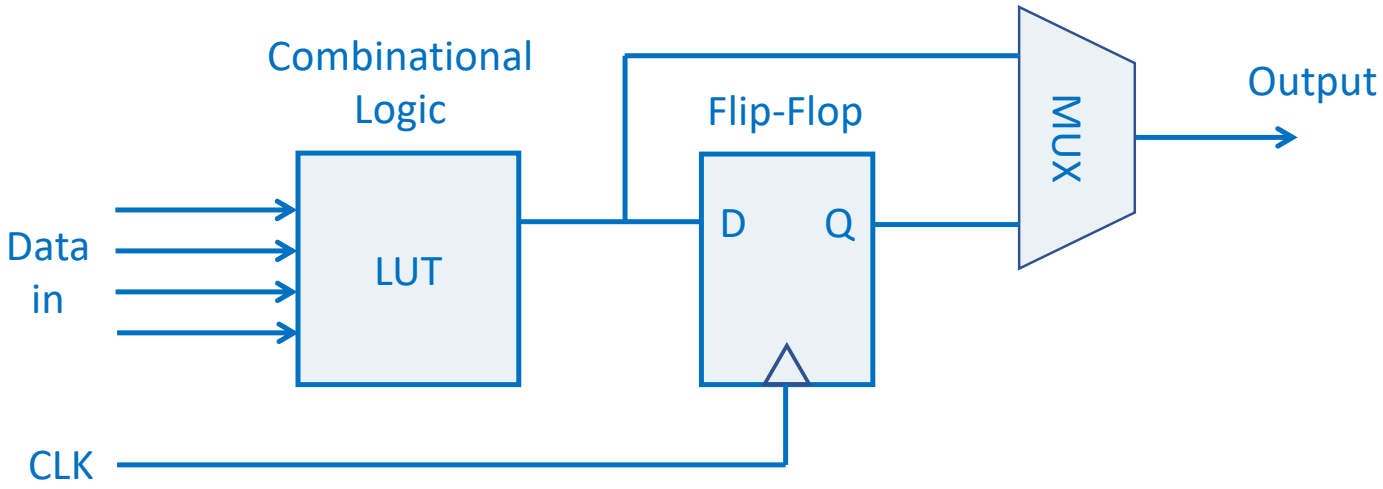
- Introduction to FPGAs
- Examples / demos
  - Simple counter
  - Stop watch
  - Frequency meter
- Hardware Description Languages
- Development boards
- FPGA chips & software tools
- How to get started

Basic FPGA Architecture



+ low-skew clock distribution network

Logic Block (simplified)



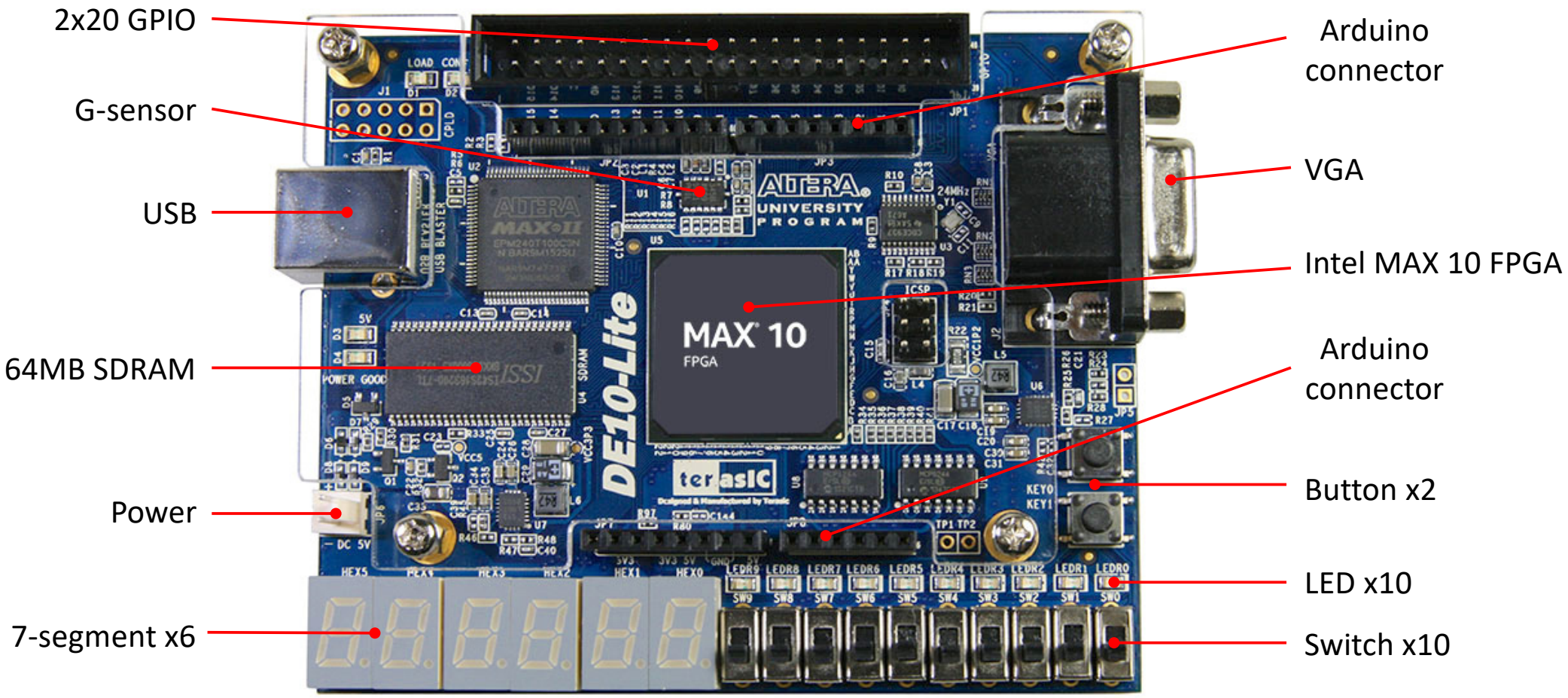
Specialized Blocks

- PLLs for clocks
- Block RAM
- Multipliers

} In almost all FPGAs

- Special DSP blocks
- High-speed transceivers
- PCIe interfaces
- Memory controllers
- ADCs
- Flash memory
- Hard CPU cores

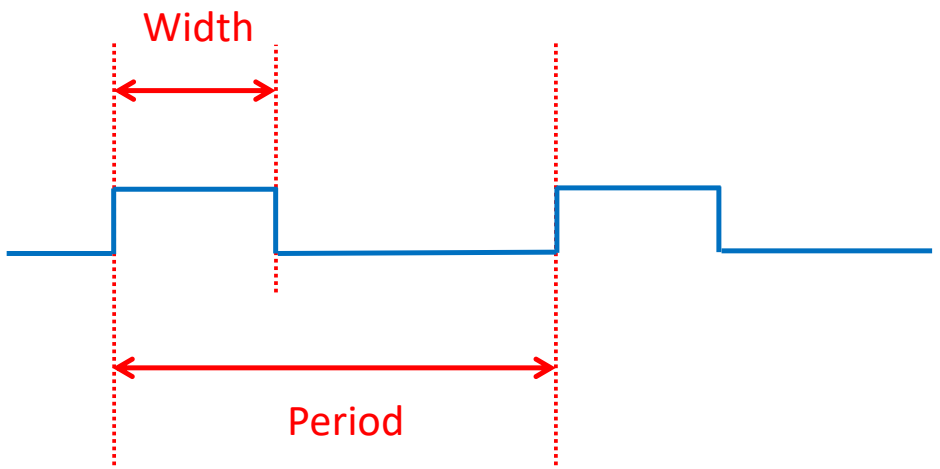
Terasic DE10-Lite



Example: Intel MAX10 (10M50DAF484C7G)

Logic blocks	50K
I/O pins	360
Package	484-pin BGA
Multipliers	144 18x18
Memory	1638 Kbit
Flash memory	5888 Kbit
ADCs	2
PLLs	4
Max clock	450 MHz

Demo 1 - Frequency Meter



Frequency = 1 / Period

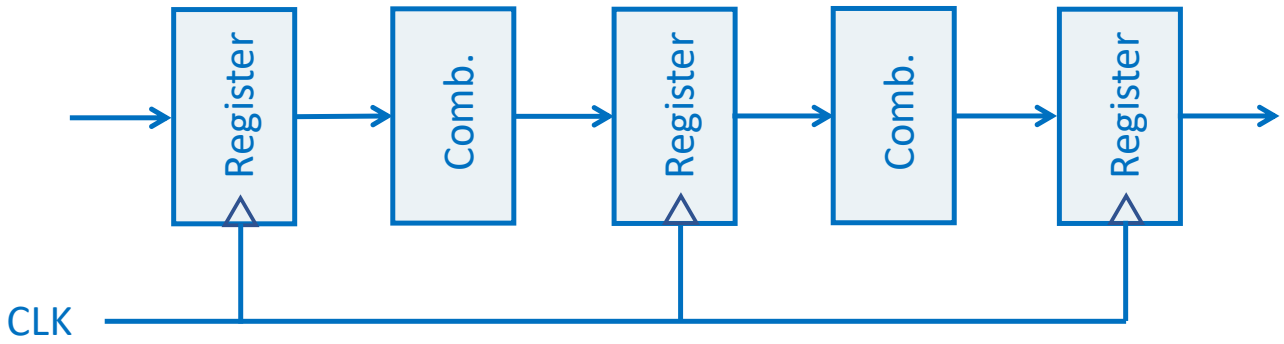
VHDL

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity counter is
   Port (clk   : in  std_logic;
         reset : in  std_logic;
         count : out std_logic_vector (29 downto 0));
end counter;

architecture behavioral of counter is
   signal cnt : std_logic_vector(29 downto 0) := (others => '0');
begin
   process(clk)
   begin
      if (rising_edge(clk)) then
         if (reset = '1') then
            cnt <= (others => '0');
         else
            cnt <= cnt+1;
         end if;
      end if;
   end process;

   count <= cnt;
end architecture;
```

SystemVerilog

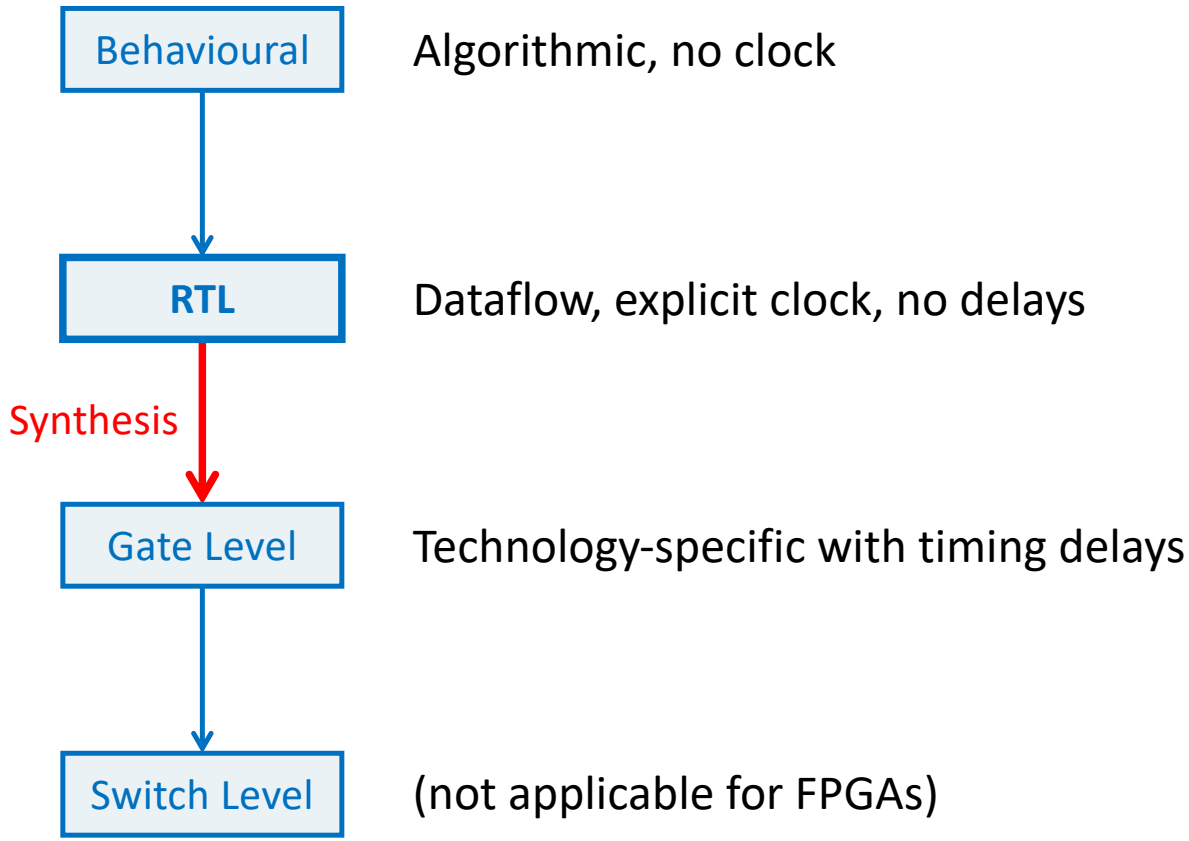
```
module counter (
    input  clk,
    input  reset,
    output logic [29:0] count);

    always_ff @(posedge clk) begin
       if(reset)
          count <= '0;
       else
          count <= count + 1'b1;
    end
endmodule
```

Register Transfer Level (RTL)



- Design abstraction of a synchronous system
- Dataflow between registers
- All registers (flip-flops) clocked simultaneously

Levels of Abstraction



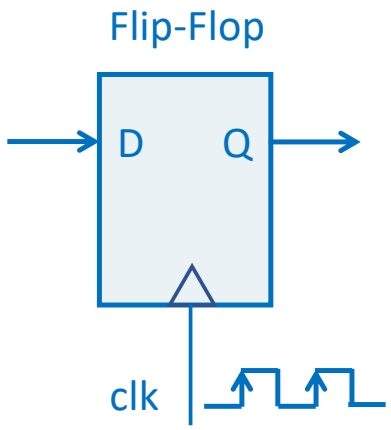
Algorithmic, no clock

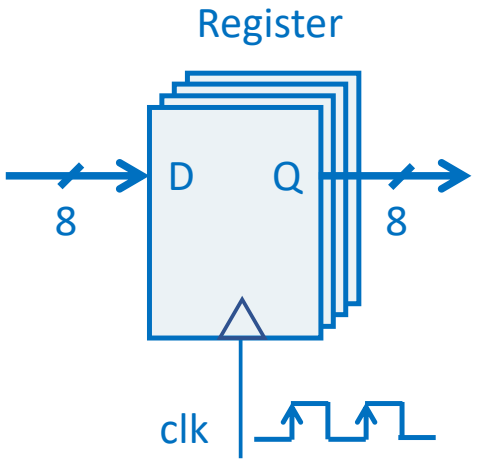
Dataflow, explicit clock, no delays

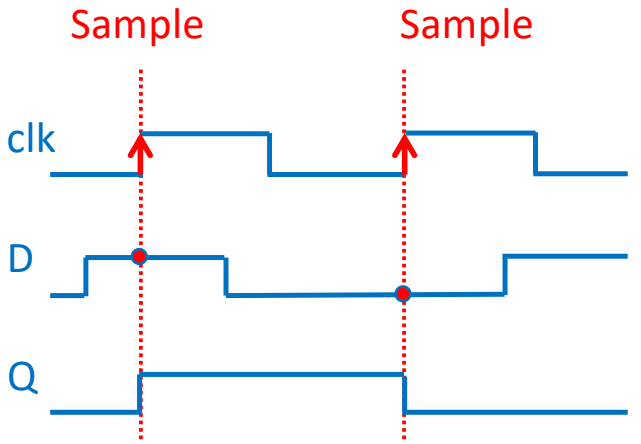
Technology-specific with timing delays

(not applicable for FPGAs)

D-Type Edge-Triggered Flip-Flops








```
logic D, Q;

always @(posedge clk)
    Q <= D;
```

```
logic [7:0] D, Q;

always @(posedge clk)
    Q <= D;
```

Demo 2 - Combinational Logic

Verilog:

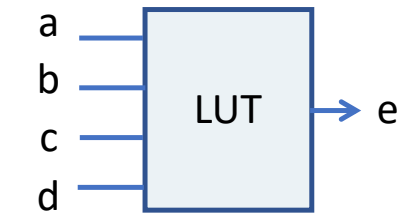
```
assign e = (a & b) | (c & d);
```

or:

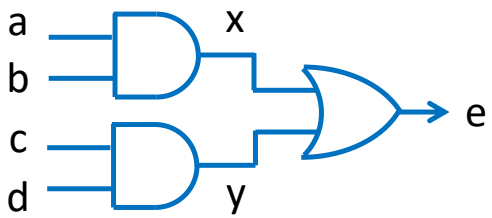
```
assign x = a & b;
assign y = c & d;
assign e = x | y;
```

a b c d	e
0 0 0 0	0
0 0 0 1	0
...	
1 1 1 0	1
1 1 1 1	1

Truth table



Look-up table



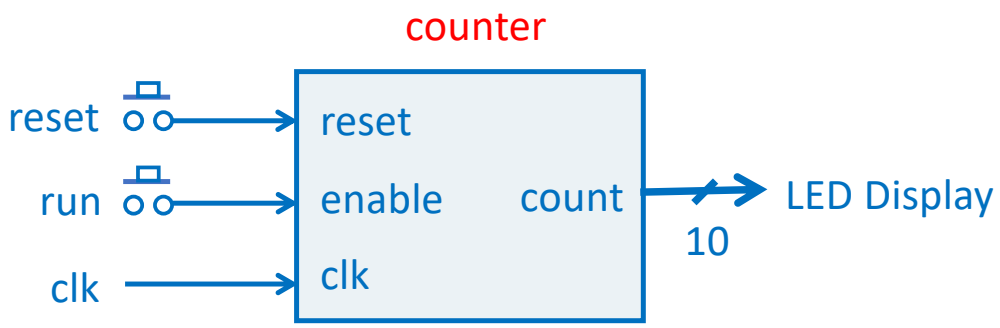
Gates

Quartus Settings File (QSF)

```
set_global_assignment -name FAMILY "MAX 10 FPGA"
set_global_assignment -name DEVICE 10M50DAF484C7G
set_global_assignment -name DEVICE_FILTER_SPEED_GRADE 7
...
set_location_assignment PIN_P11 -to MAX10_CLK1_50
set_location_assignment PIN_B8 -to KEY[0]
set_location_assignment PIN_A8 -to LEDR[0]
set_location_assignment PIN_C10 -to SW[0]
...
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to MAX10_CLK1_50
set_instance_assignment -name IO_STANDARD "3.3 V SCHMITT TRIGGER" -to KEY[0]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to LEDR[0]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to SW[0]
...
```

Note: Incorrect settings can damage FPGA!

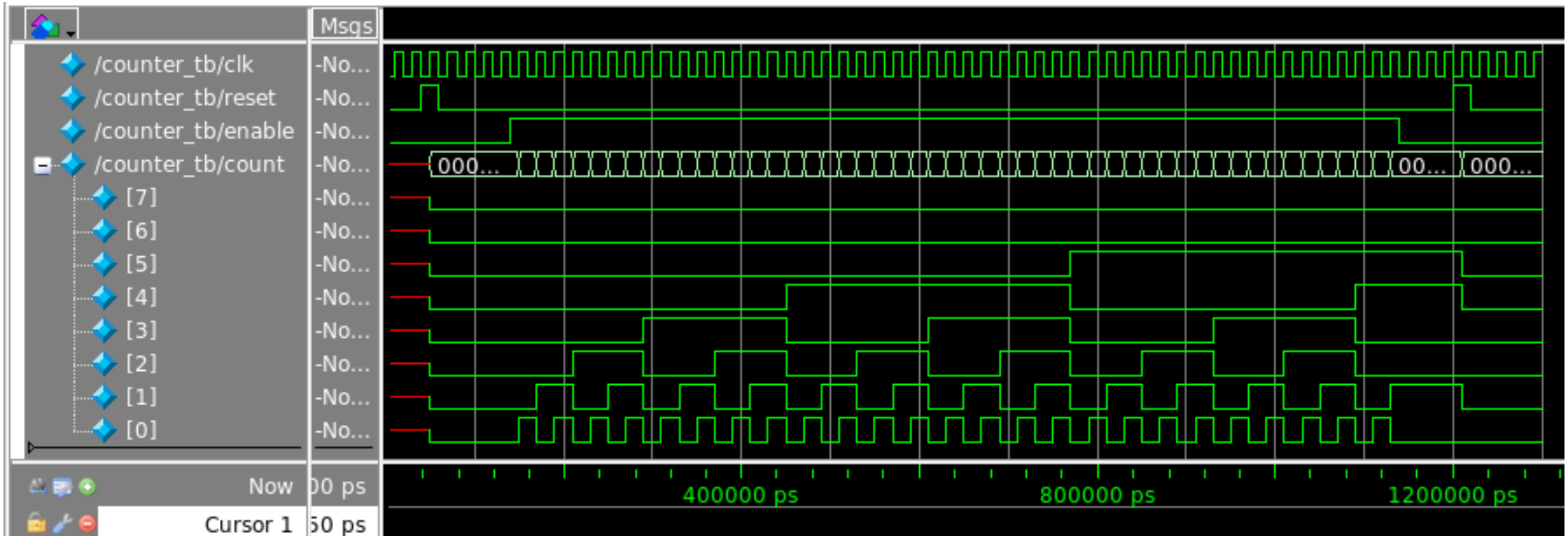
Demo 3 - Simple Counter




```
module counter #(parameter WIDTH=10) (
   input clk,
   input reset,
   input enable,
   output logic [WIDTH-1:0] count
);
   always_ff @(posedge clk) begin
      if(reset)
         count <= 0;
      else if(enable)
         count <= count + 1'b1;
    end
endmodule
```

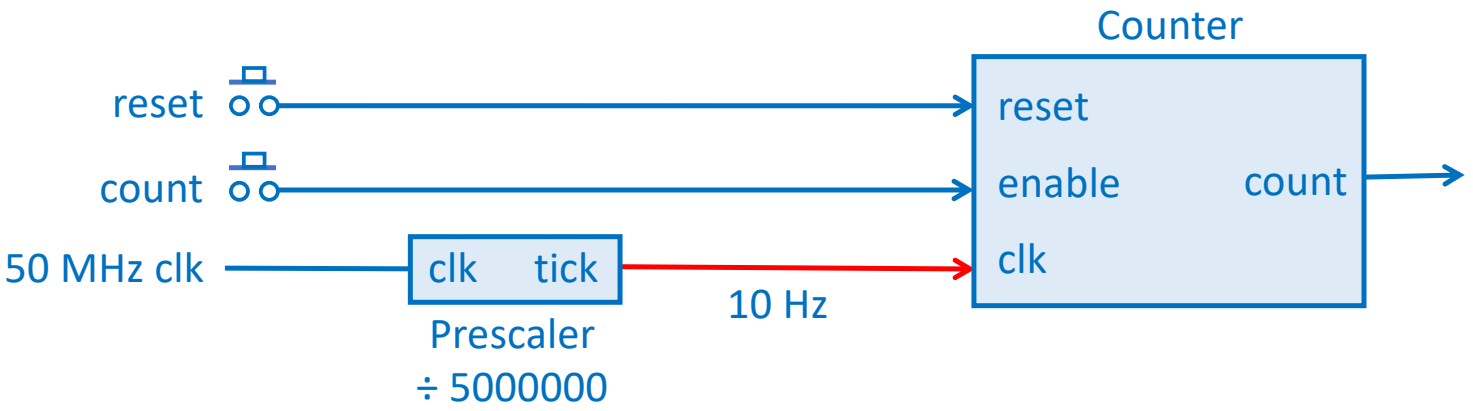
Note: DE10-Lite buttons are debounced

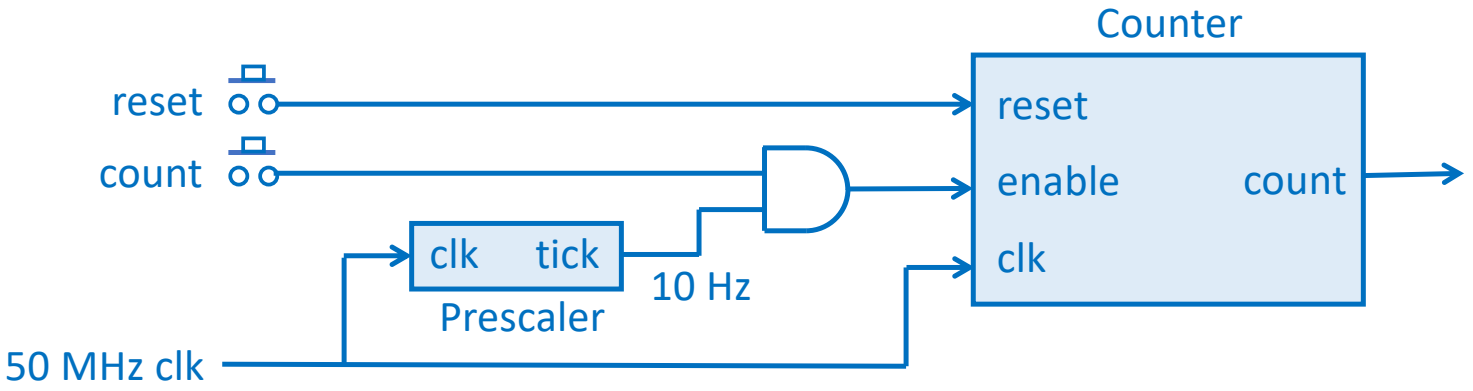
Simulation



- Graphical simulation for debugging
- Verification: assertions and test vectors

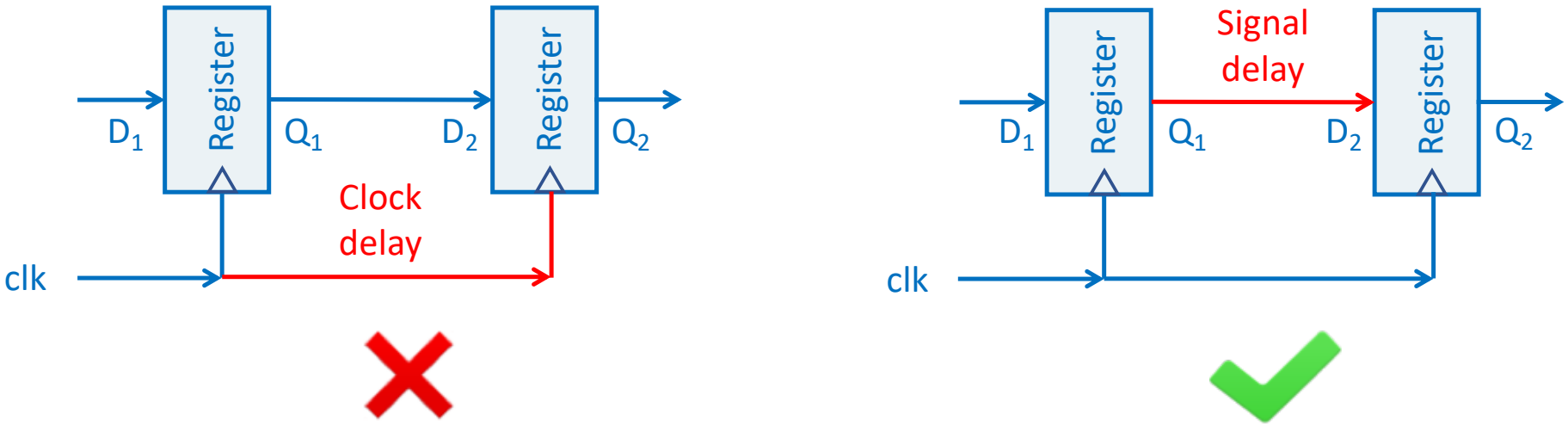
Prescaler





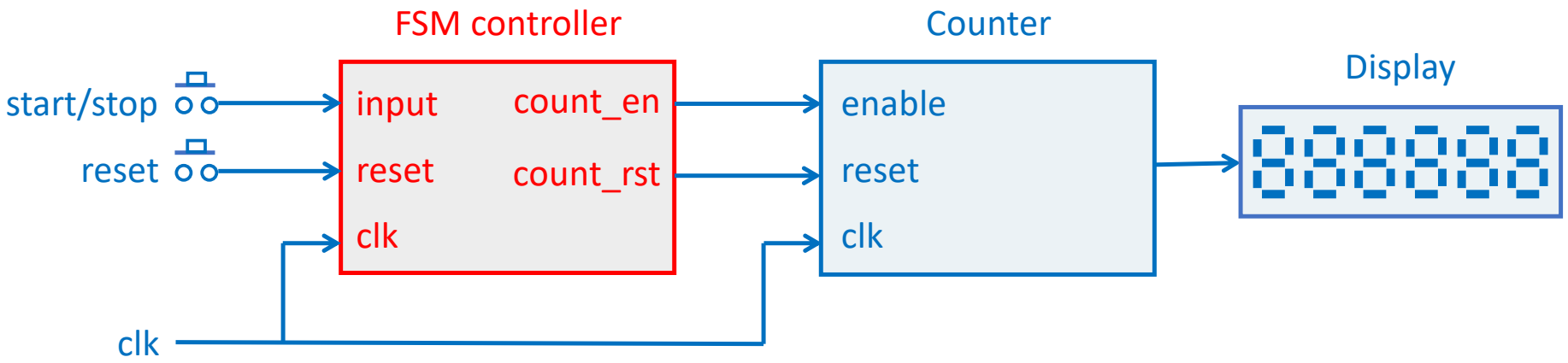
Note: Input synchronization ignored for now

Clock Skew



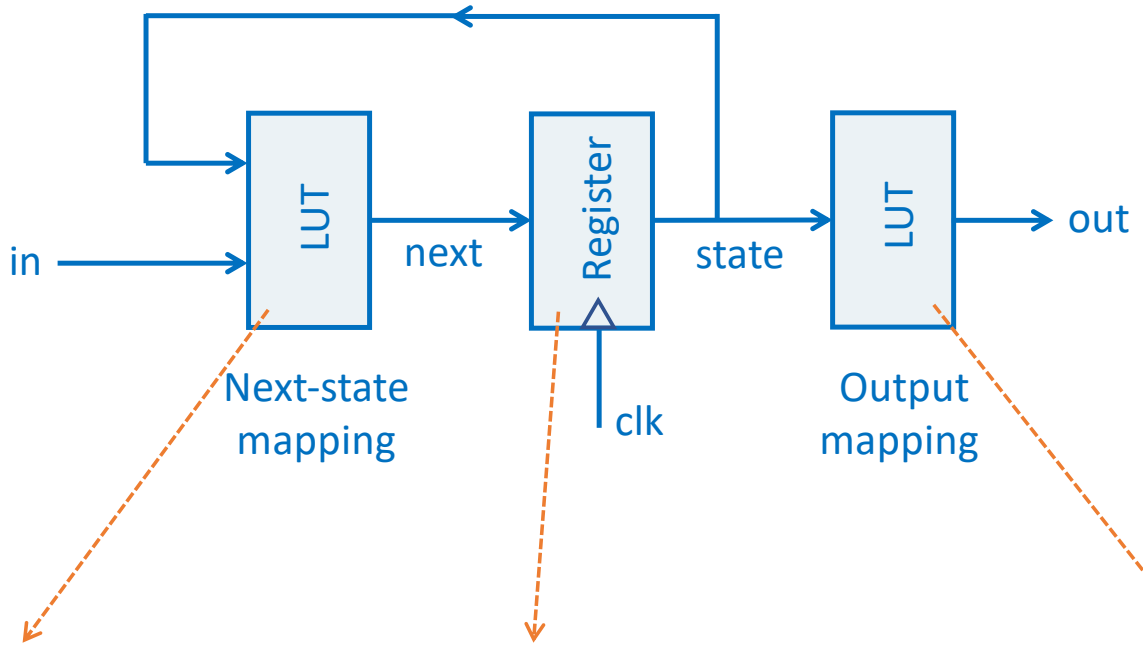
- All registers / D-types clocked by a single clock
- FPGA has a low-skew clock distribution network
- No clock gating

Demo 4: Stop-Watch



Note: Prescaler omitted for clarity

Finite State Machine (Moore)

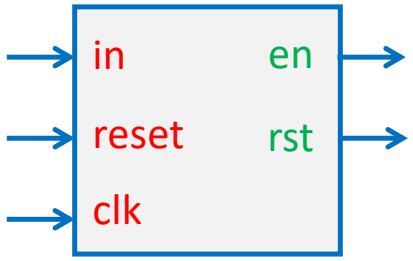


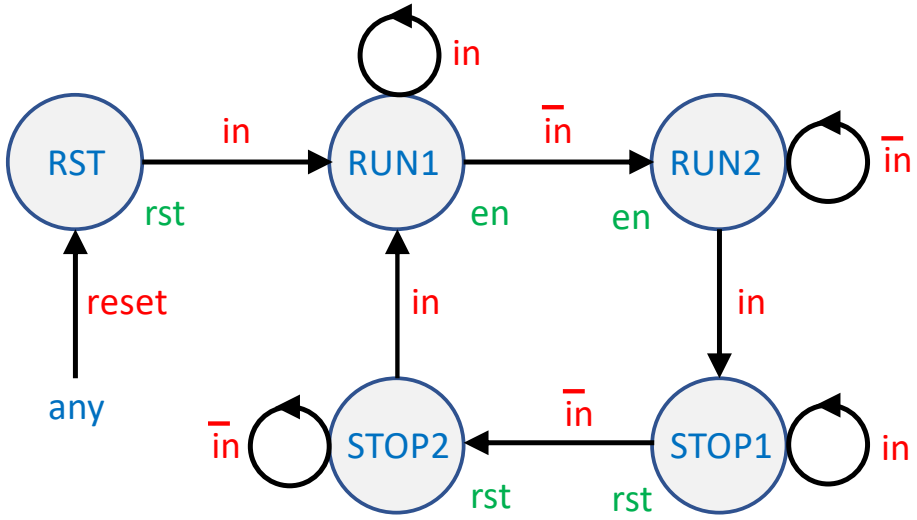

```
always_comb begin
   case(state)
      S0: next = in ? S1 : S0;
      S1: ...
      default: next = S0;
   endcase
end
```

```
always_ff @(posedge clk)
   if(reset)
      state <= S0;
   else
      state = next;
```

```
always_comb
   case(state)
      S0: out = 2'b00;
      S1: out = 2'b01;
      ...
      default: out = 2'b00;
   endcase
```

Stop Watch FSM

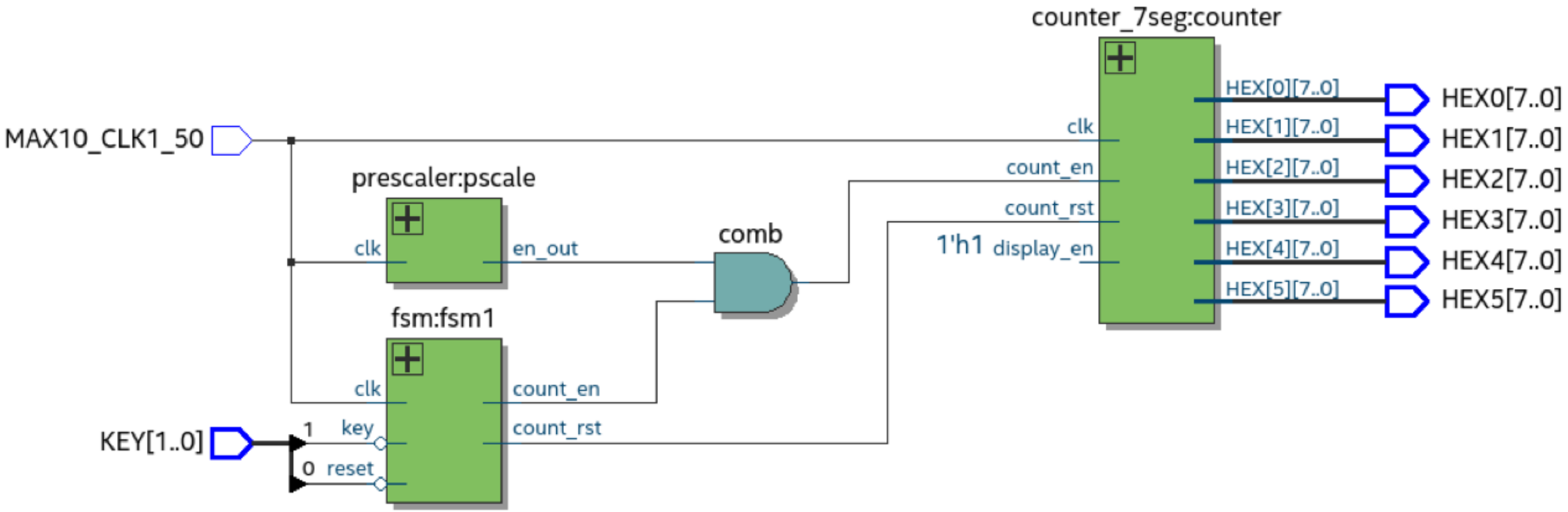





```
always_ff @(posedge clk) begin
   if(reset) state = RST;
   else case(state)
      RST:   state = in ? RUN1 : RST;
      RUN1:  state = in ? RUN1 : RUN2;
      RUN2:  state = in ? STOP1 : RUN2;
      STOP1: state = in ? STOP1 : STOP2;
      STOP2: state = in ? RUN1 : STOP2;
      default: state = RST;
   endcase
end
```

```
always_comb begin
   rst = 1'b0;
   en  = 1'b0;
   case(state)
      RST:          rst = 1'b1;
      RUN1,RUN2:    en  = 1'b1;
      STOP1,STOP2:  rst = 1'b1;
      default:      rst = 1'b1;
   endcase
end
```

Stop Watch


counter_7seg:counter
MAX10_CLK1_50
prescaler:pscale
clk
en_out
comb
fsm:fsm1
clk
count_en
KEY[1..0]
1
key
count_rst
0
reset
clk
count_en
count_rst
1'h1
display_en
HEX[0][7..0]
HEX[1][7..0]
HEX[2][7..0]
HEX[3][7..0]
HEX[4][7..0]
HEX[5][7..0]
HEX0[7..0]
HEX1[7..0]
HEX2[7..0]
HEX3[7..0]
HEX4[7..0]
HEX5[7..0]

FPGA Chips & Software Tools

Xilinx

- FPGAs: Spartan-7, Artix-7
- SOC: Zynq-7000
- Tools: Vivado (free WebPack version)

Intel (Altera)

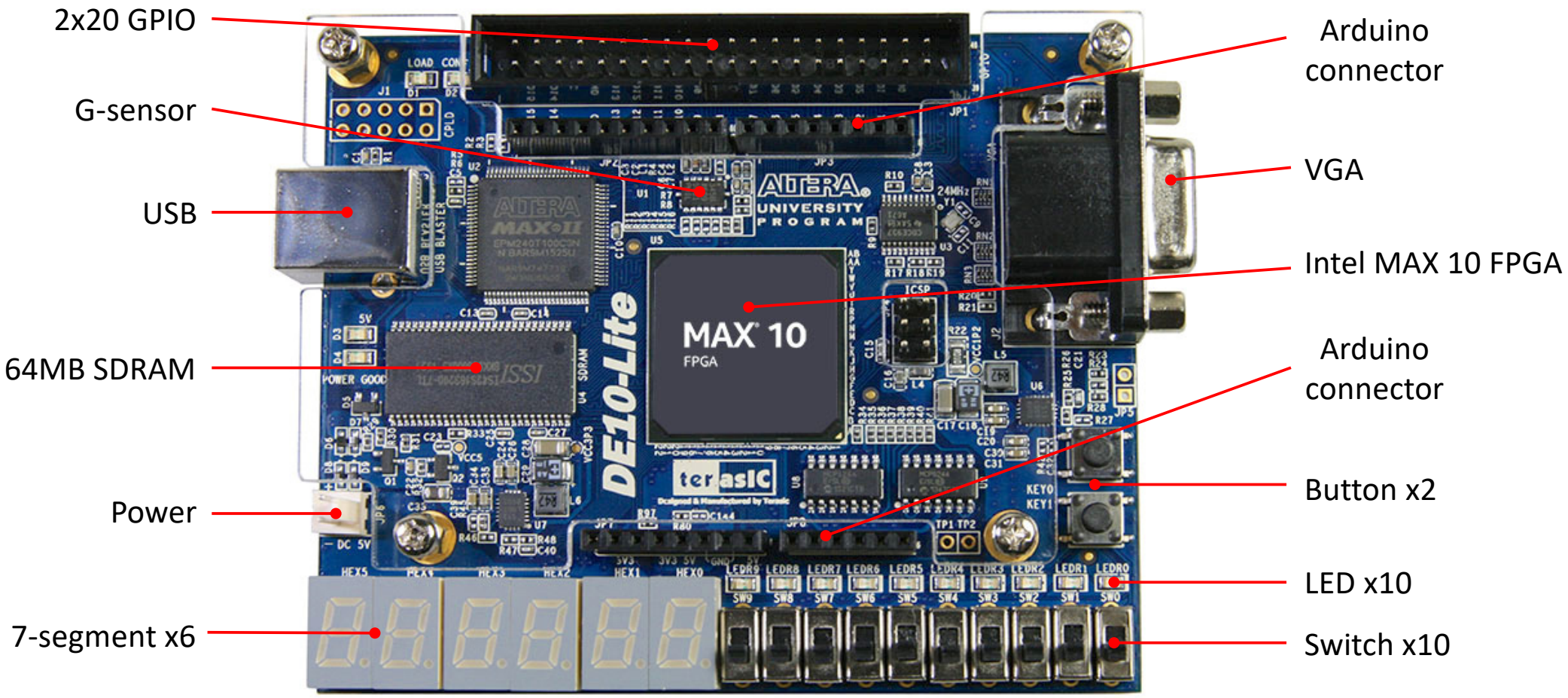
- FPGAs: MAX-10, Cyclone V, Cyclone-10
- SOC: Cyclone V SE
- Tools: Quartus (free ‘Lite’ version)

Lattice

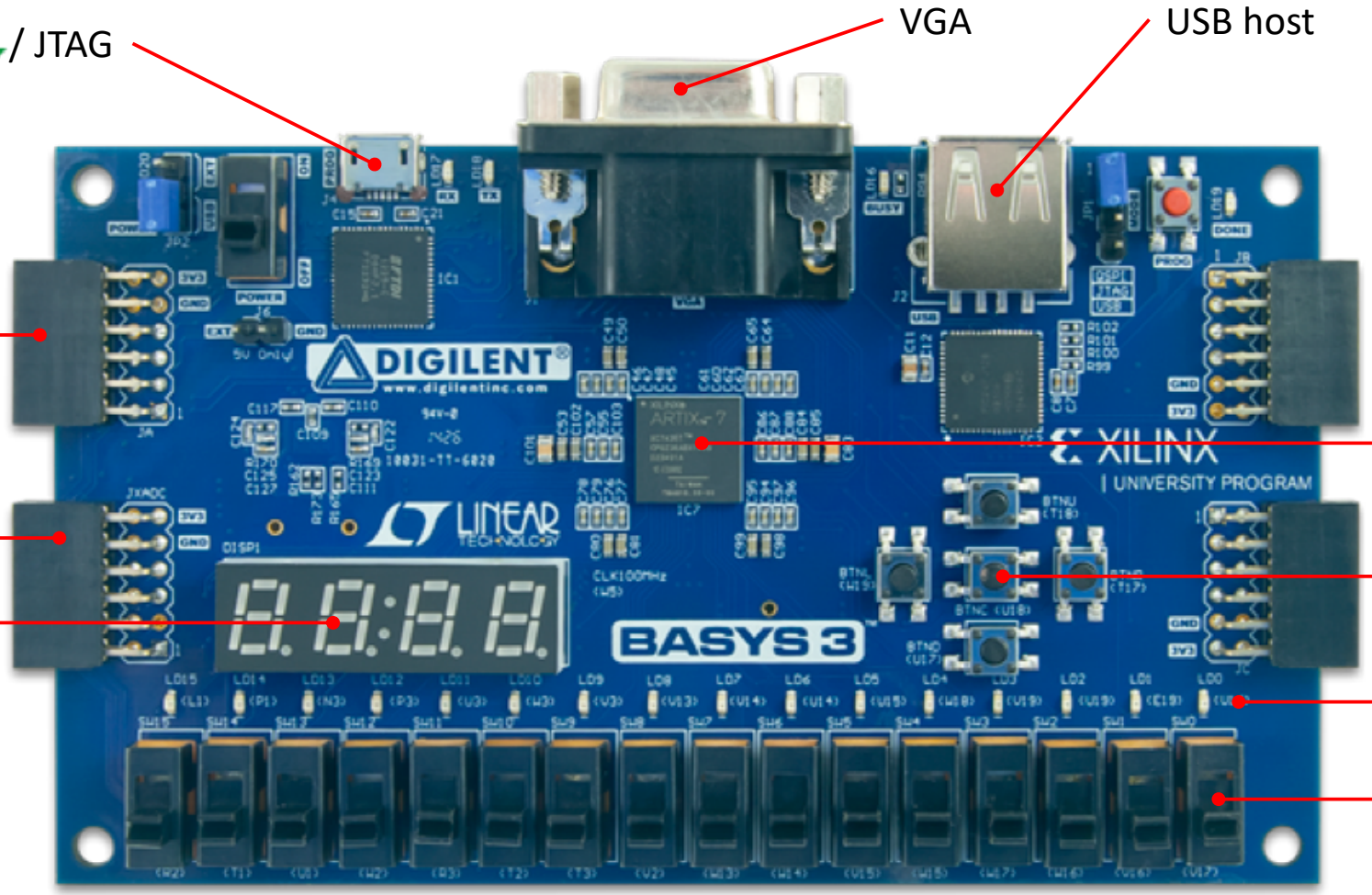
- FPGAs: ICE, ...
- Tools: ICEcube2, IceStorm (open source)

Microsemi (Actel)

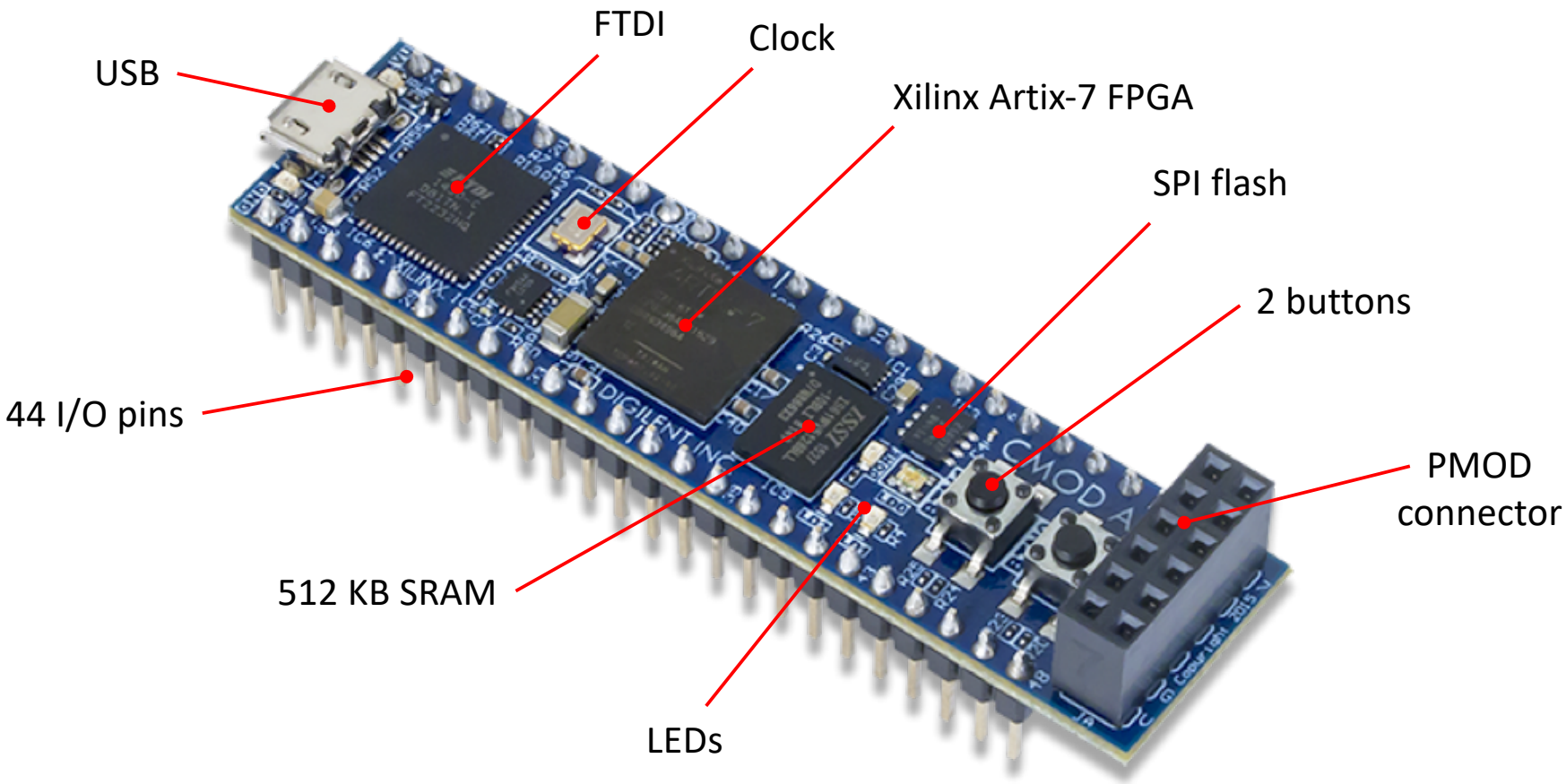
Terasic DE10-Lite



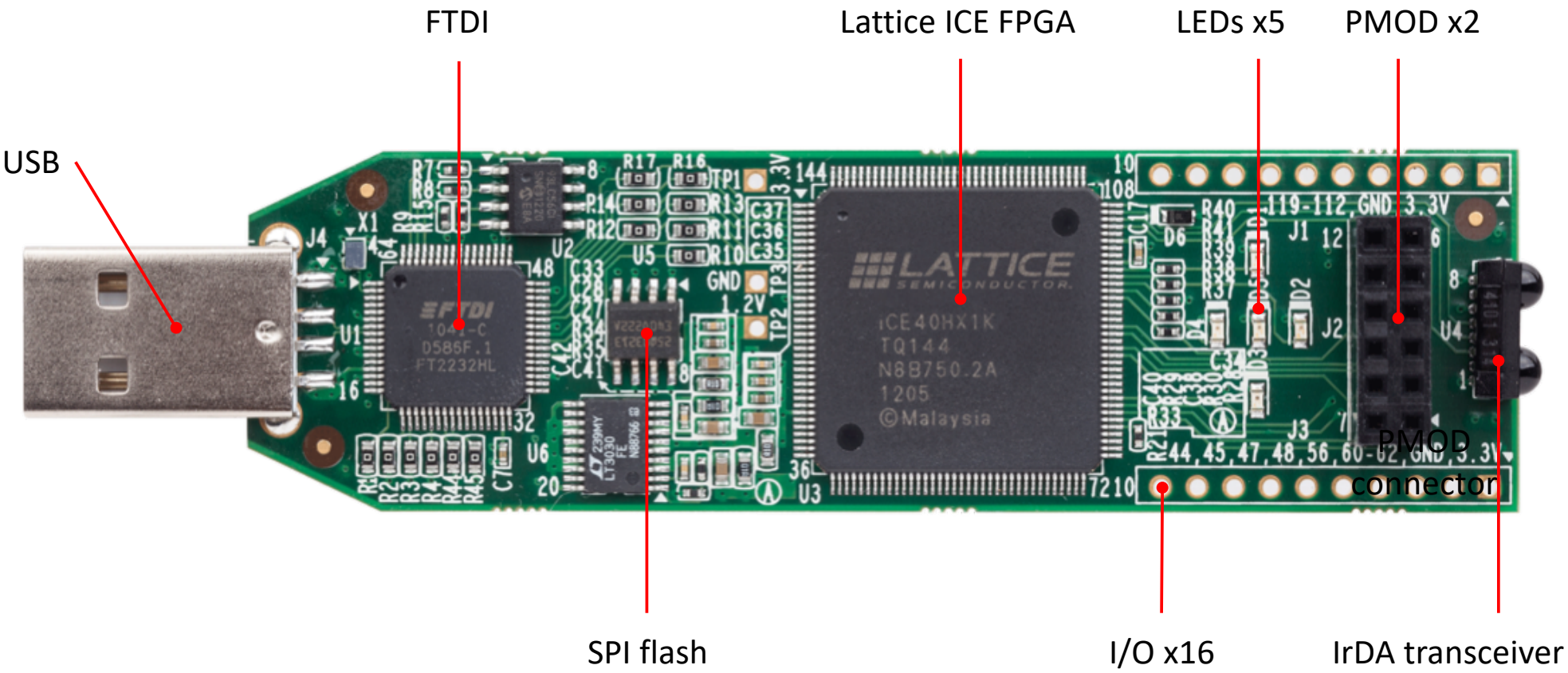
Digilent Basys 3



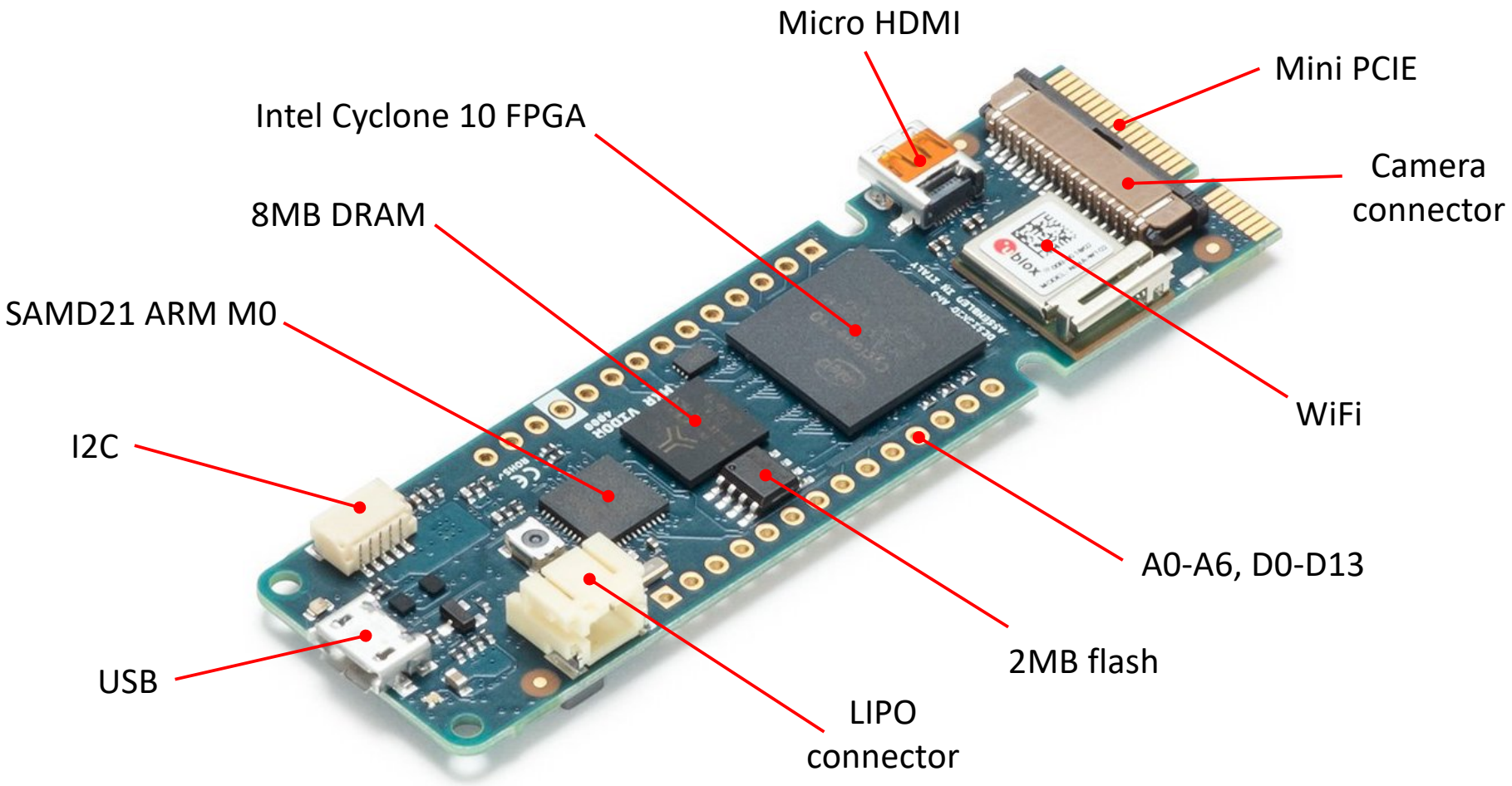
Digilent CMOD A7



Lattice IceStick



Arduino Vidor 4000



Hardware Description Languages

Mainstream

- VHDL
- Verilog (2005)
- SystemVerilog

Research

- CλaSH (Haskell)
- Chisel (Scala)
- Bluespec (Haskell)

Other

- SystemC (C++ library)
- MyHDL (Python)

Hardware Description Languages

VHDL

- Verbose syntax
- Strongly typed, user-defined types
- High-level constructs
- Deterministic simulation
- Packages for reusable components

Verilog (2005)

- Simple but more limited
- Suitable for ASICs as well as FPGAs

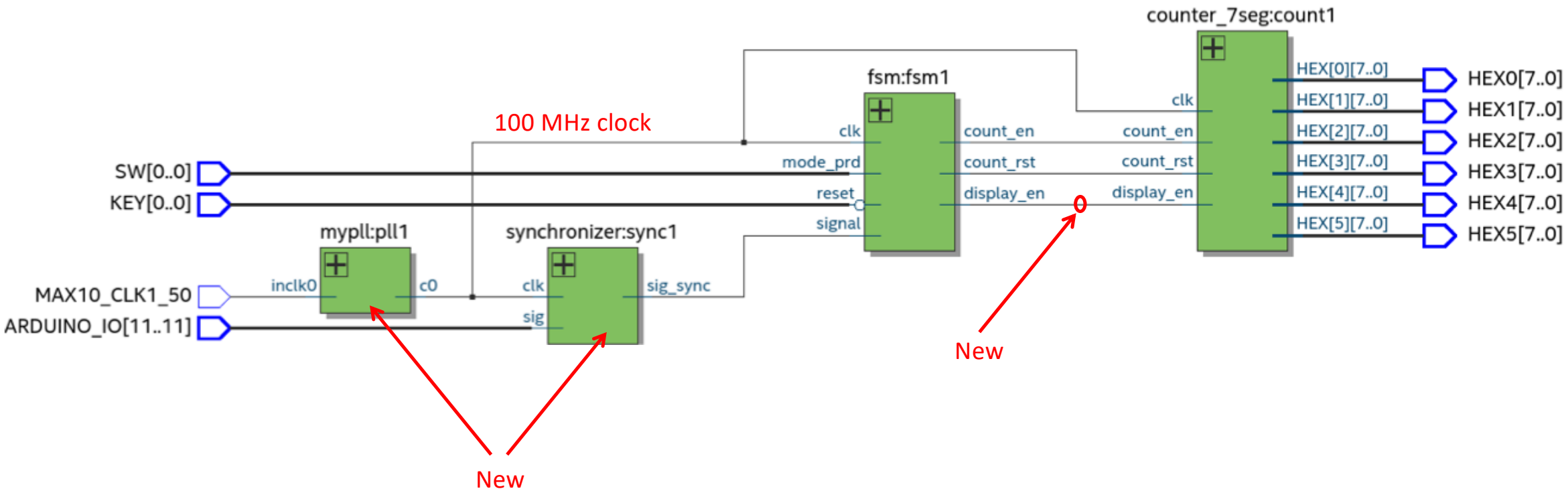
System Verilog

- Successor to Verilog
- Superset of Verilog with many new features
- A hardware design and verification language
- The Verilog & System Verilog standards have been merged

The Frequency Meter

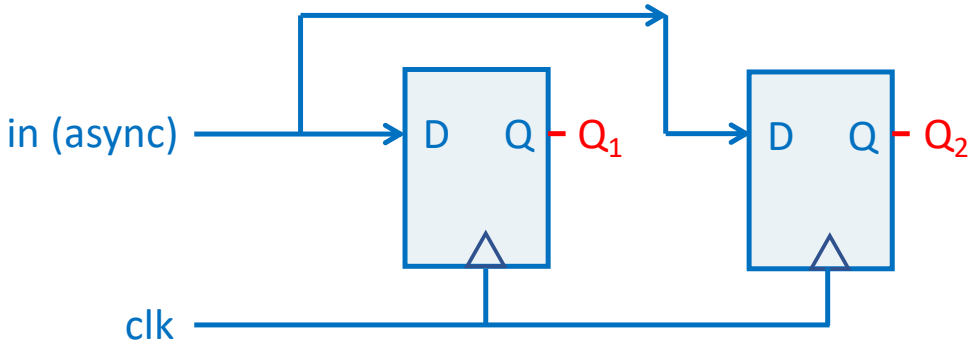
- We have the basic building blocks for the frequency meter
- Like the stop-watch but triggered from input signal instead of button
- Frequency, period and width measurement

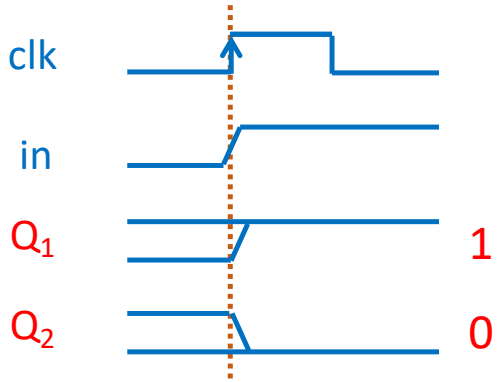
Frequency Meter (with period & width)

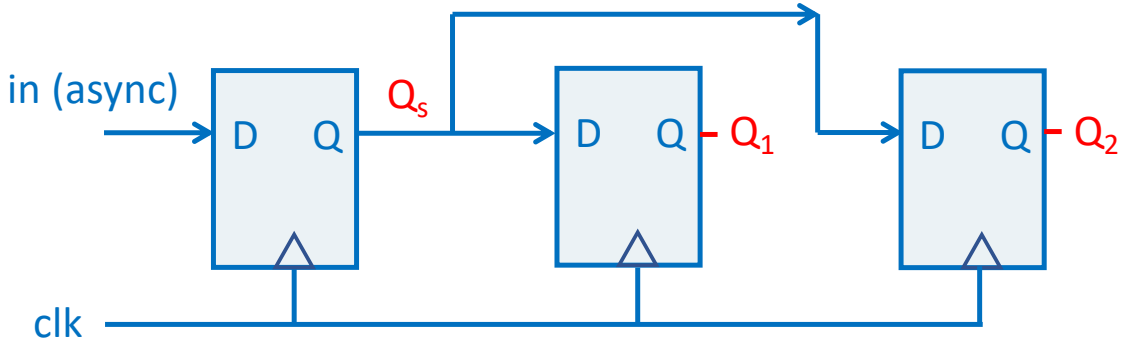


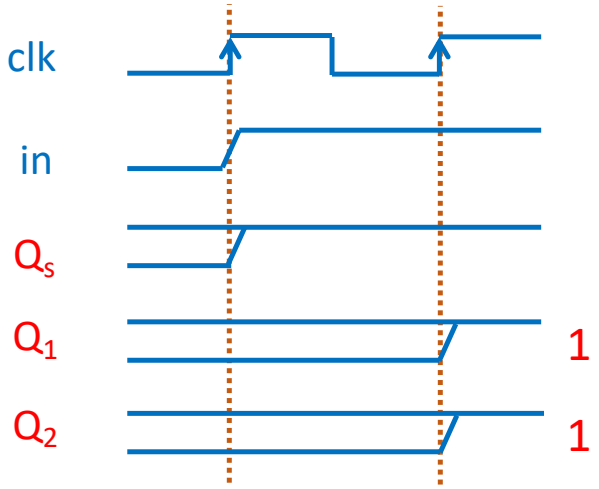
Note: Buttons and switches should ideally be synchronized too

Synchronising Inputs

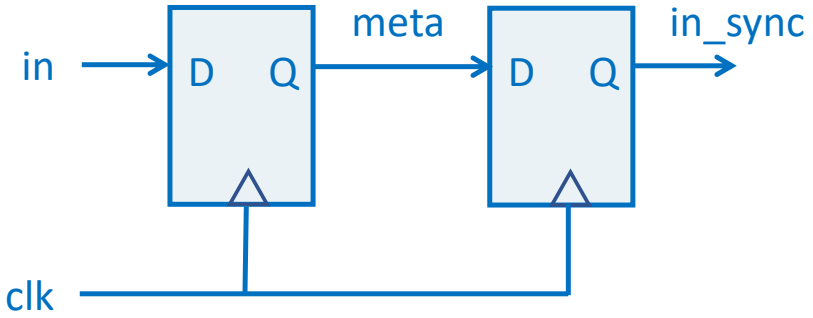


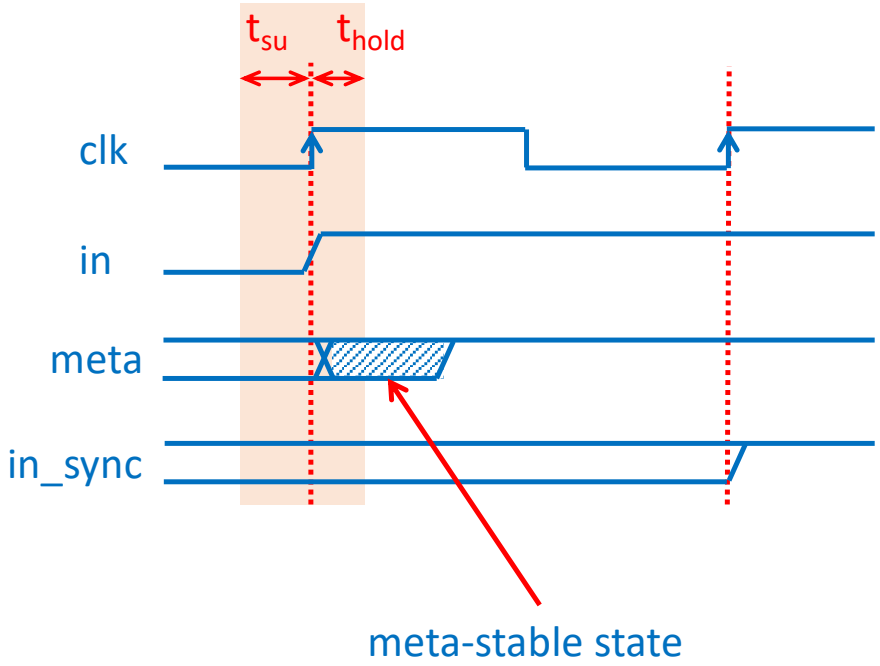






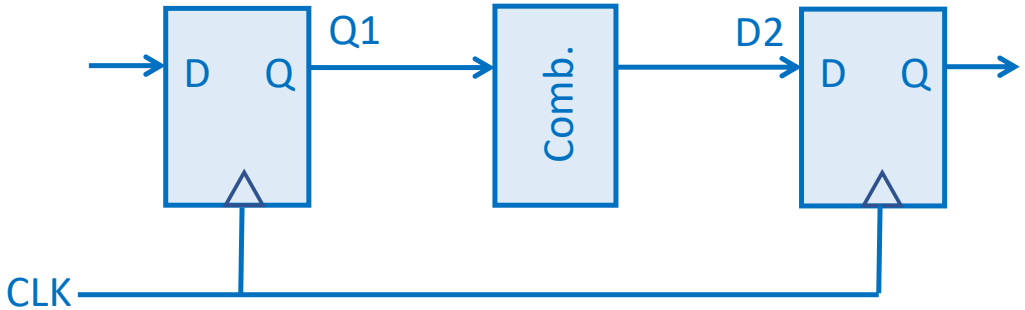
Metastability

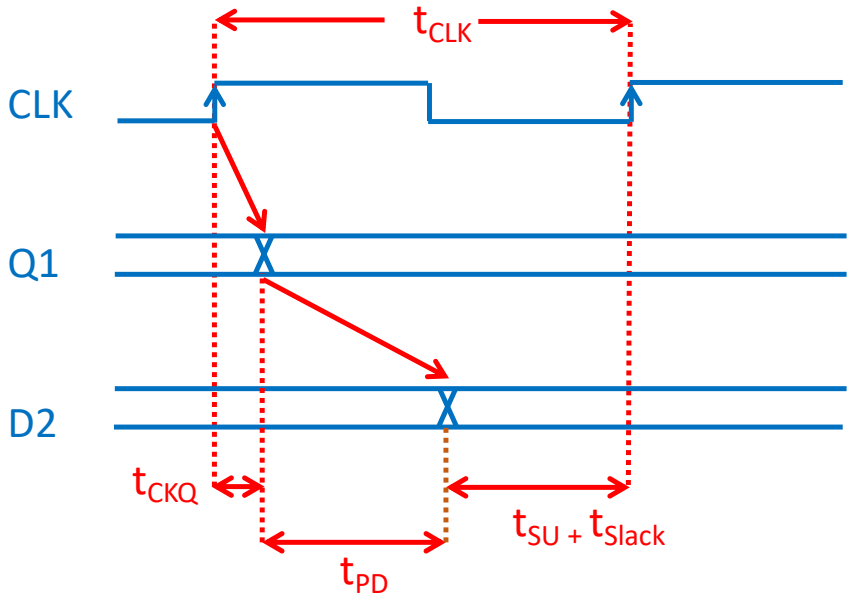





```
always_ff @(posedge clk) begin
  meta <= in;
  in_sync <= meta;
end
```

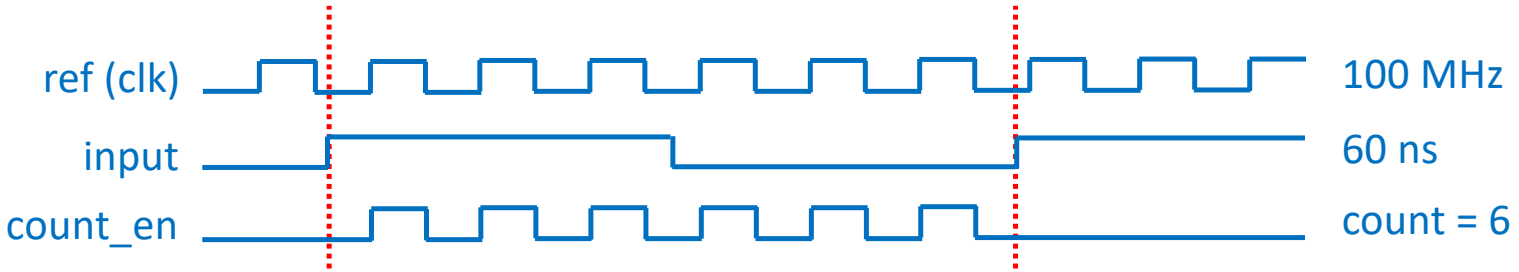
Timing Margin



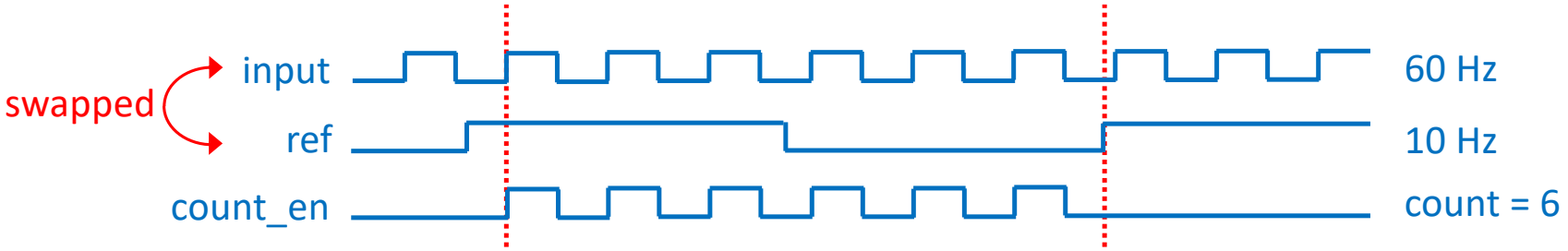


Frequency Measurement

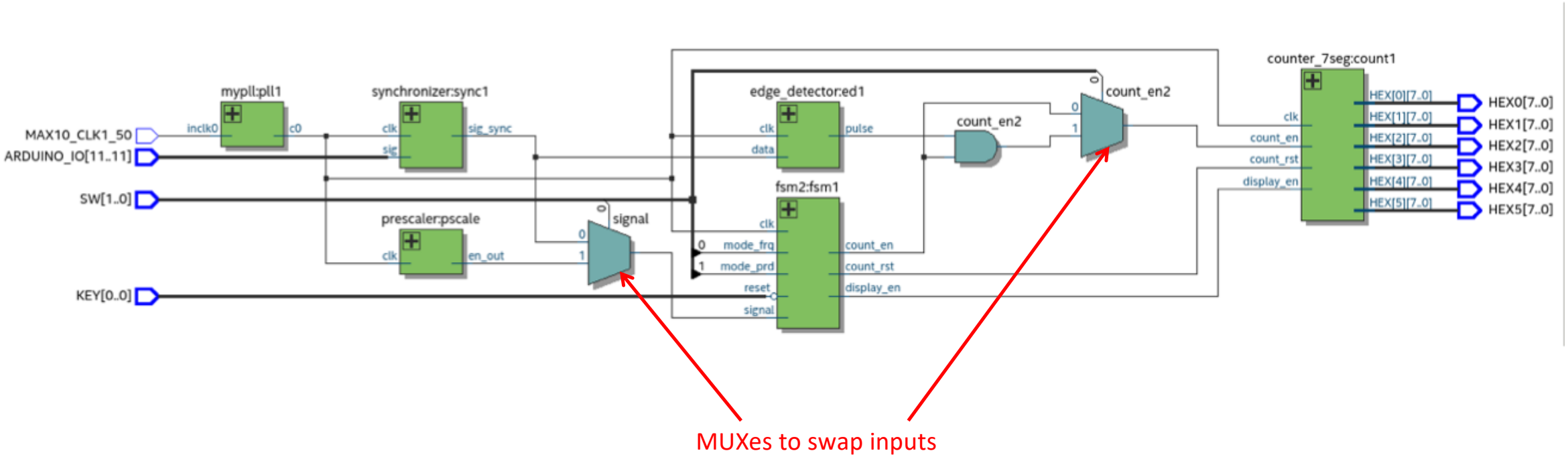
Period measurement



Frequency Measurement



Frequency Meter (with frequency, period & width)



Possible Enhancements

- Range switching
- Decimal point
- Delayed period/width update
- Equal precision method
- Soft CPU
- SCPI interface
- Faster clock: pipelined counter
- Single-shot mode
- Hardware: signal conditioning & TCXO

Conclusions - How to Get Started

- Get an evaluation board (IceStick is only €21, DE10_Lite €78)
- Choose an HDL: VHDL or (System)Verilog
- Start with a simple mini-project and gradually work up
    - Make your designs fully synchronous
    - Only use a single clock
- There are many good university lecture course notes on-line

Think hardware & digital design - You are not programming!

Further Information

FPGA Boards

- http://www.terasic.com.tw/en/
- https://store.digilentinc.com

Development Software

- https://fpgasoftware.intel.com/?edition=lite
- https://www.xilinx.com/support/download.html

Verilog and Digital Design

- http://www.ece.ubc.ca/~edc/7660.jan2017/lectures/
- http://www.ee.ic.ac.uk/pcheung/teaching/ee2_digital/

Other

- https://www.edaplayground.com
- https://opencores.org