Safe FPGA Design Practices for Instrumentation and Control in Nuclear Plants

Miljko Bobrek[1], Richard T. Wood[1], Christina D. Ward[1], Stephen M. Killough[1], Don Bouldin[2], Michael E. Waterman[3]

1 Oak Ridge National Laboratory, Oak Ridge, TN
2 The University of Tennessee, ECE Department, Knoxville, TN
3 Nuclear Regulatory Commission, Washington, DC

Presentation Contents

- Problem Statement
- Introduction to FPGAs
- Safe design practices
- Validation and Verification
- Maintainability and Obsolescence
- Conclusions

Problem Statement

- Obsolete I&C systems in NP need to be replaced
- New plants are built that need modern I&C systems
- Modern, general purpose I&C systems are based on microprocessors, digital signal processors, microcontrollers, and FPGAs
- Currently used I&C systems in NP are based on analog electronics, low-level integration digital circuits, and microcontrollers many of which are obsolete.
- NRC needs guidance and acceptance criteria to help with licensing of FPGA-based safety-critical I&C systems in NP
- There is no ready-to-use regulatory document for FPGA-based safety systems

Introduction to FPGAs

What is an FPGA?

- Field Programmable Gate Array
- One of programmable logic devices such as GPP, mC, DSP, and ASIC.
- Contains unconnected basic logic elements such as AND gates, OR gates, and flip-flops.
- More complex FPGAs include adders, multipliers, memory blocks, and microprocessors.
- Interconnections are done by a designer using EDA tools.
- Some FPGAs can be reconfigured completely or partially during the development phase or during the exploitation phase
- FPGAs represent a higher level of integration of digital hardware, but they also involve software design.

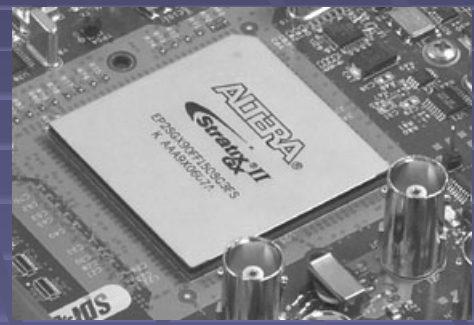
Introduction to FPGAs

FPGA History

- PROMs in 1950s
- PLAs in early 1970s
- PALs in late 1970s
- GALs in 1980s
- CPLDs in late 1970s
- FPGAs in late 1970s
- Platform FPGAs in late1990s

Introduction to FPGAs

X 10000 =



Source: Altera

Introduction to FPGAs

CPLD Structure

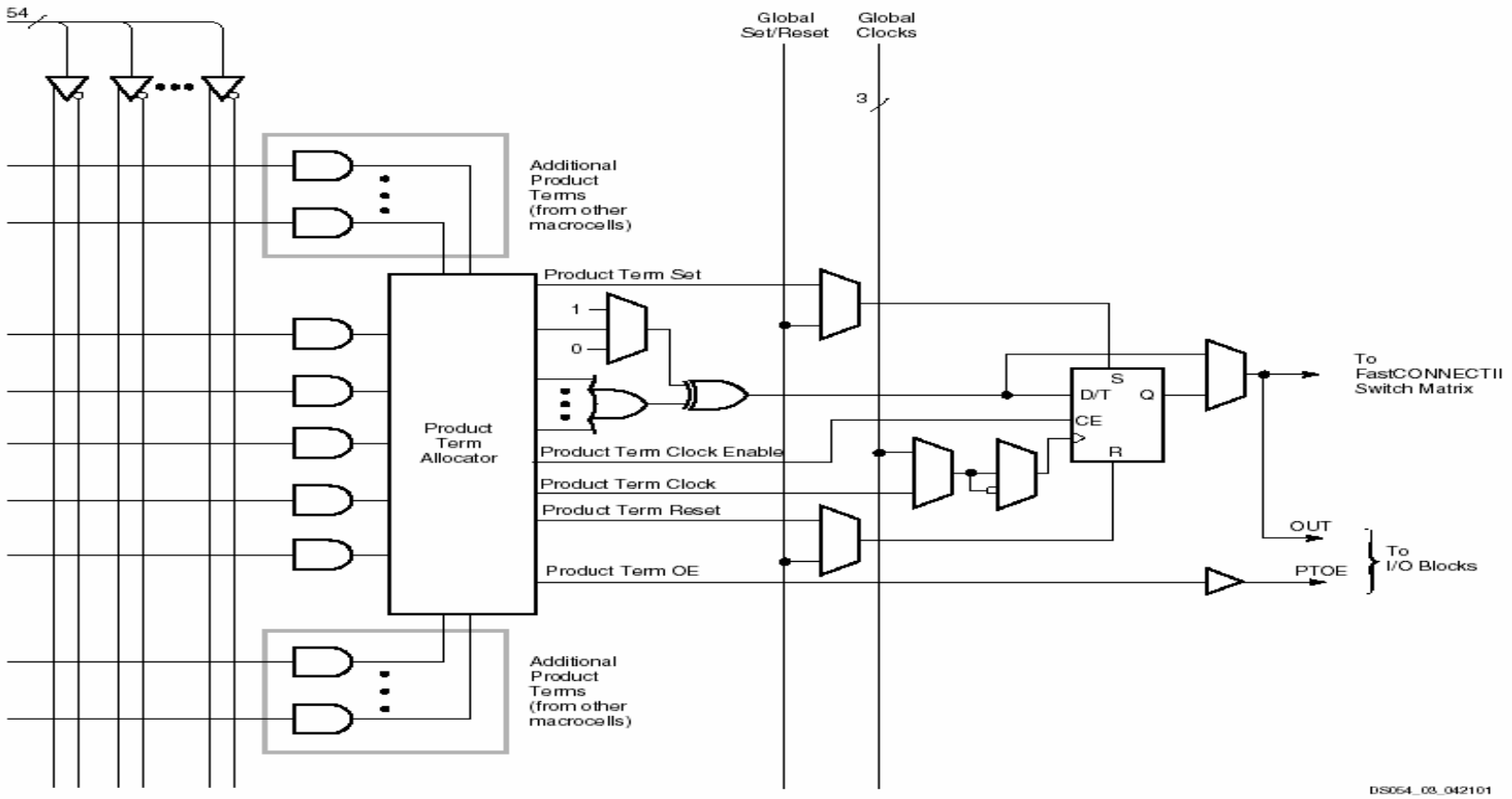
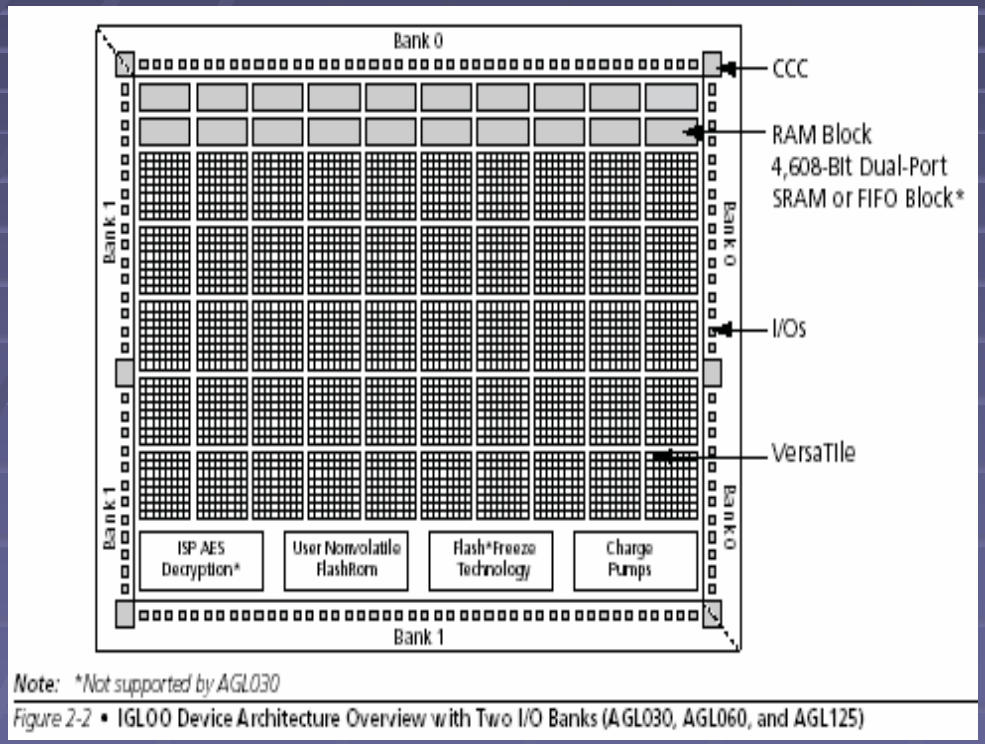


Figure 3: **XC9500XL Macrocell Within Function Block**

Source: Xilinx

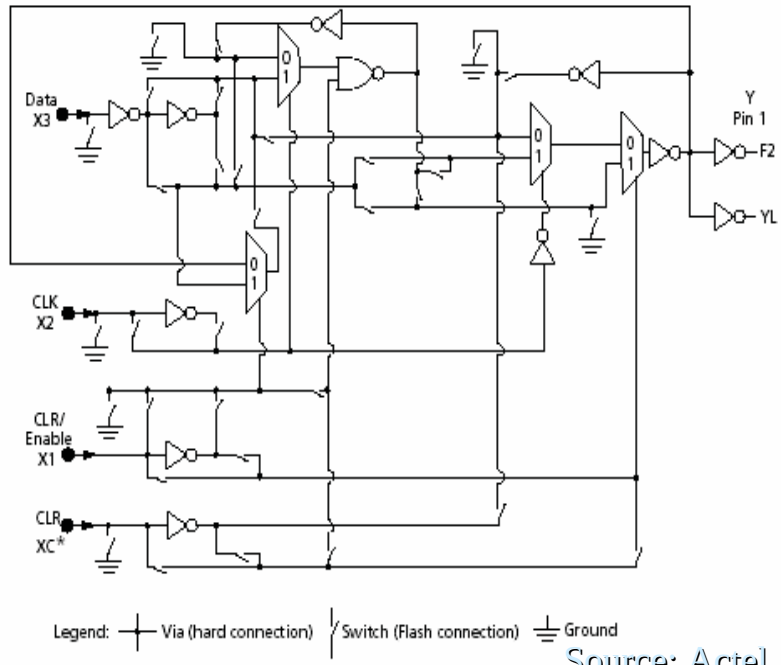
Introduction to FPGAs

FPGA Structure



Note: *Not supported by AGL030

Figure 2-2 • IGLOO Device Architecture Overview with Two I/O Banks (AGL030, AGL060, and AGL125)



Source: Actel

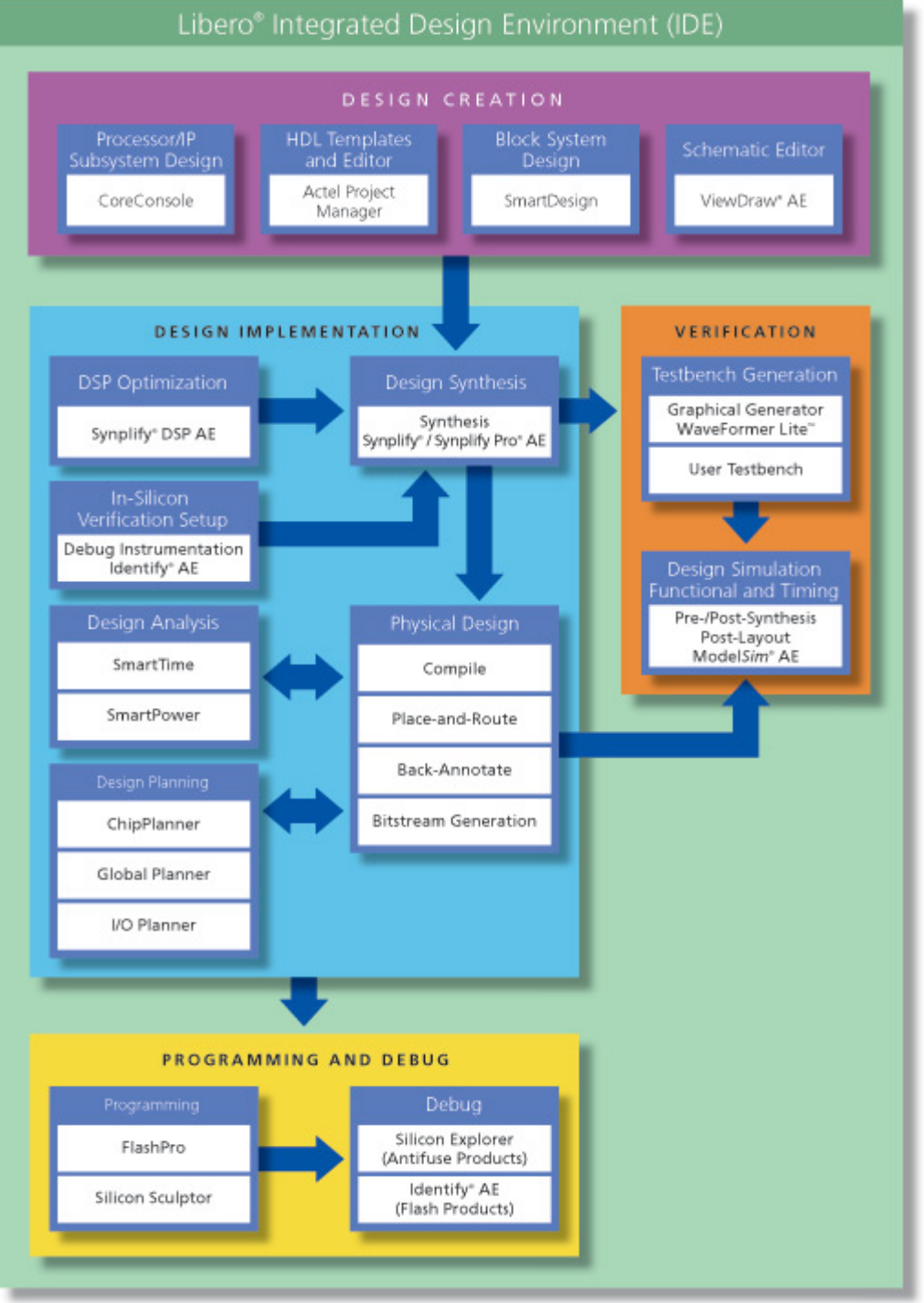
Note: *This input can only be connected to the global clock distribution network.

Figure 2-4 • IGLOO Core VersaTile

Introduction to FPGAs

FPGA Classifications

- By size
  - From several thousands to several millions of gates
- By packaging
  - Quad Flat Pack, Ball Grid Array
- By configuration
  - SRAM, EEPROM, FLASH, One_Time
- By application
  - General purpose, High speed, Large designs, Low power, Rad hard
- By manufacturer
  - Actel, Aeroflex, Altera, Atmel, Cypress, Lattice, Quicklogic, Xilinx



FPGA Design Methodology

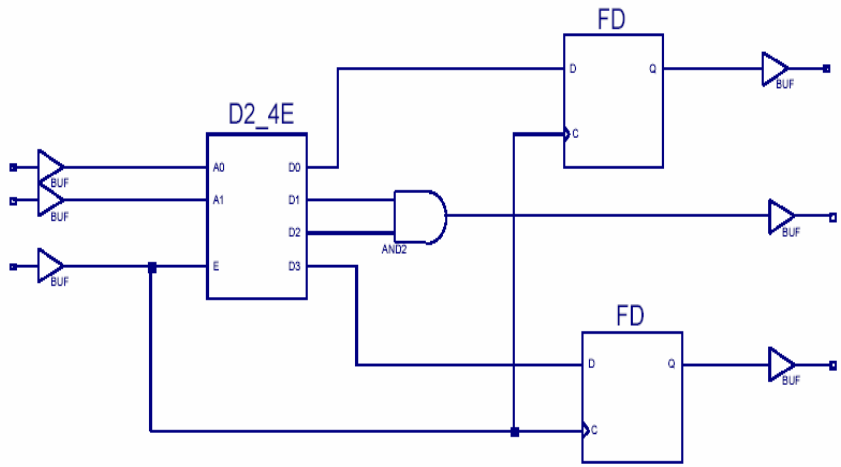
- Design entry
- Behavioral Simulation
- Synthesis
- Post Synthesis Simulation
- Place and Route
- Post P&R Simulation
- FPGA Configuration
- Hardware Verification

Source: Actel

Introduction to FPGAs

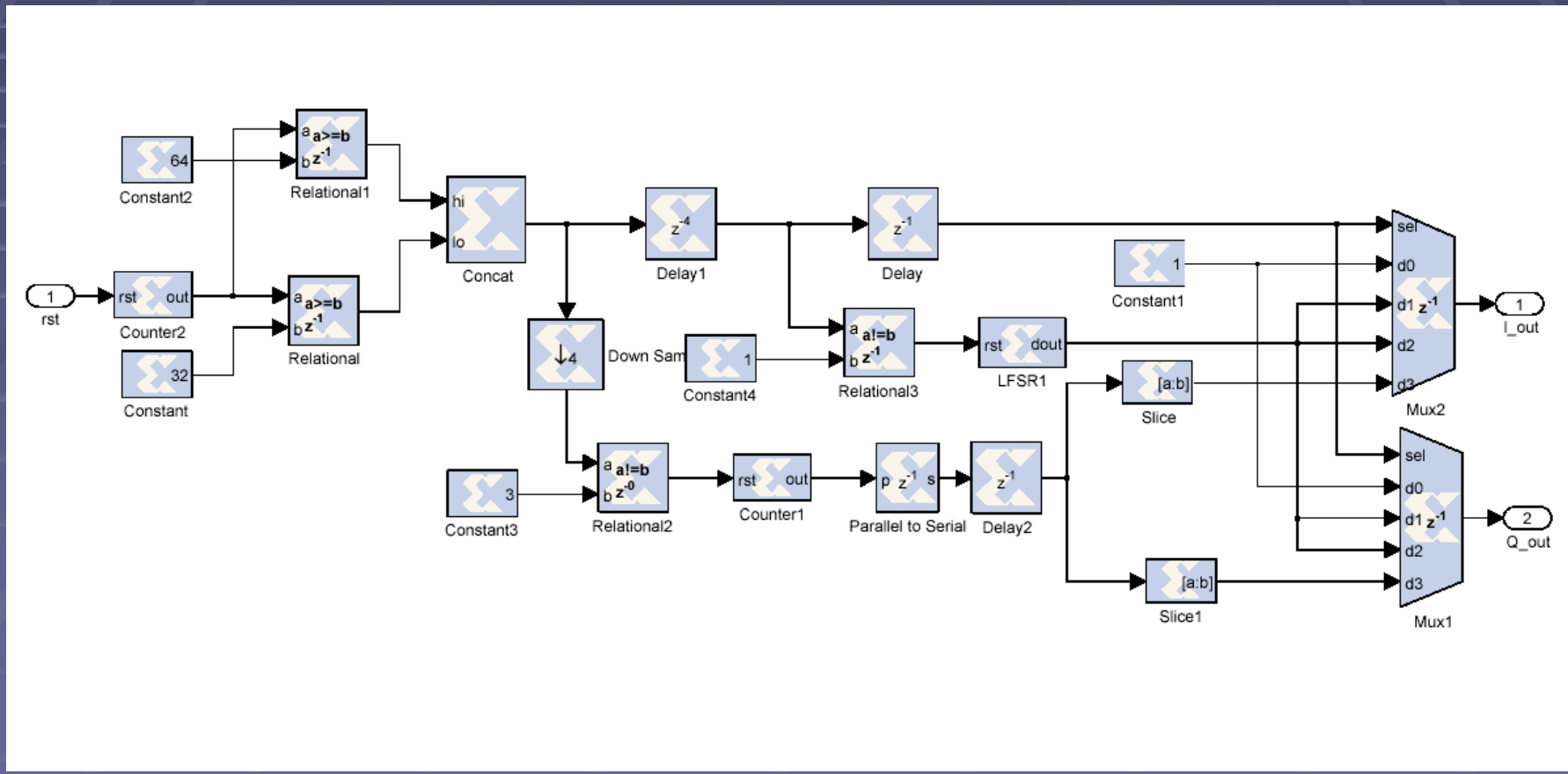
FPGA Design Entry

```
process(SCLK,RSTB,NCS)

begin
     if RSTB = '0' or NCS = '1' then
        BIT_CNT <= "0000";
    elsif SCLK'event and SCLK = '1' then
        if BIT_CNT <= "1001" then
           BIT_CNT <=  BIT_CNT + '1';
        else
           BIT_CNT <= BIT_CNT;
        end if;
   end if;
end process;
```



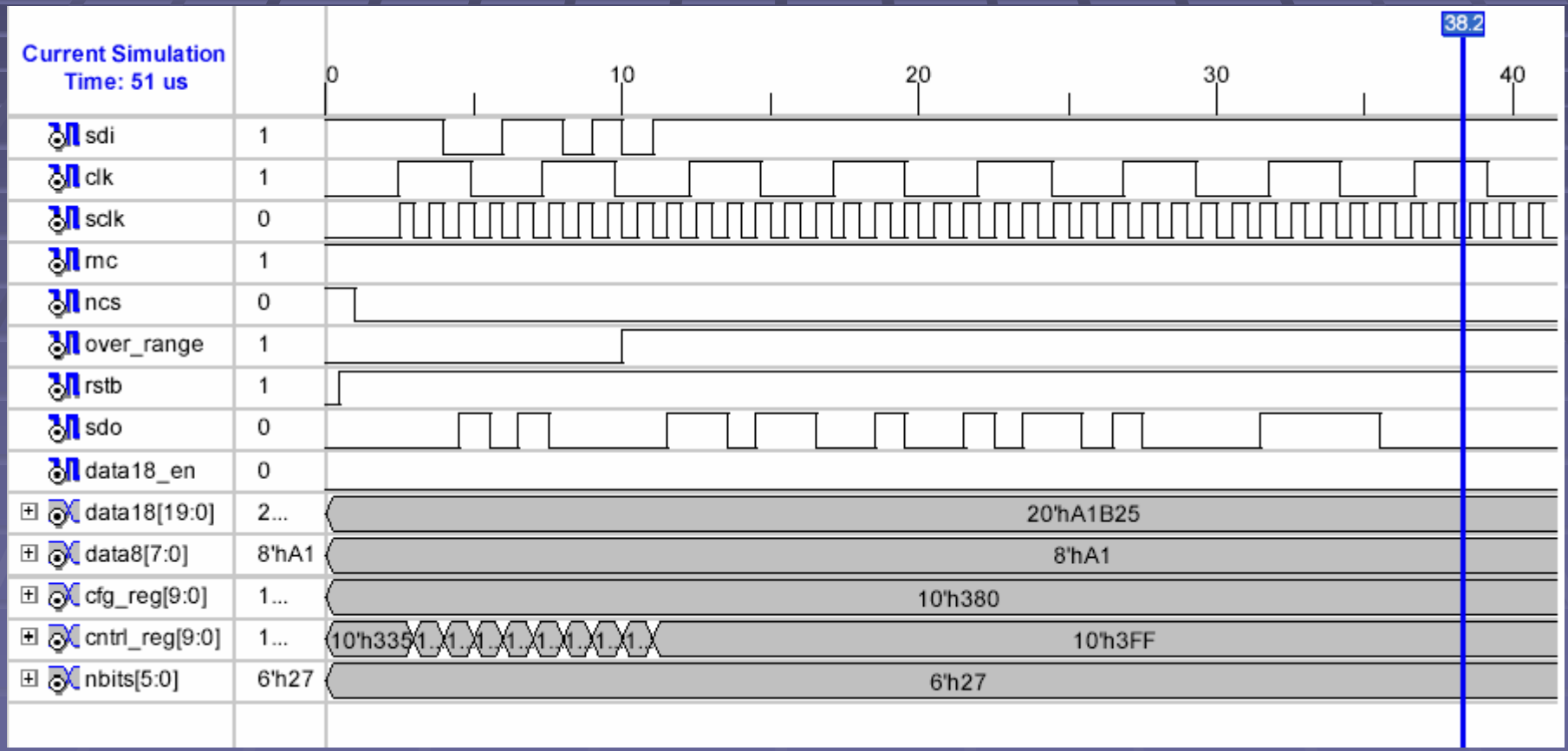
Introduction to FPGAs

FPGA Design Entry



Introduction to FPGAs

FPGA Design Simulation



Safe FPGA Design Practices

- System Level Design Practices.
- Board Level Design Practices.
- FPGA Level Design Practices.

System-Level Design Practices

- Design partitioning to FPGAs, DSPs, GPPs, memories, interfaces.
- Implementing multiple functions in a single FPGA.
- Mixing safety-critical and safety-noncritical designs into a single FPGA
- FPGA type and/or size selection.
- System partitioning with respect to Validation and Verification

Board-Level Design Practices

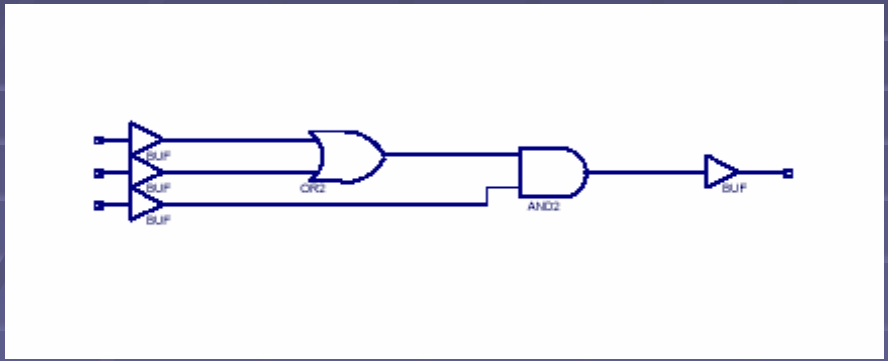
- Supply power fluctuations and ground bounce.
  - Proper power and ground plane design and PCB layer stacking
  - Selection and proper design of power regulators to satisfy maximum rise-time and power sequencing required for the FPGA power-up
  - Placement of power decoupling capacitors
  - Mitigation of Simultaneous Switching Outputs (SSO) effects.
  - Reducing the output rise time of the FPGA I/Os

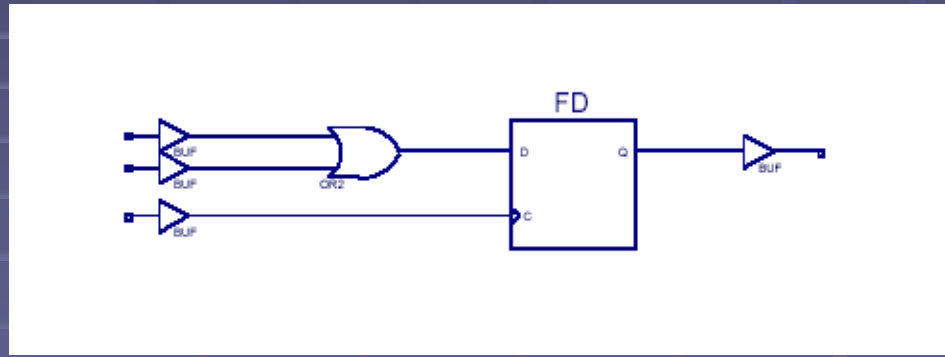
Board-Level Design Practices

- Separation of digital and analog parts of the design residing on the same board
- Proper termination and isolation of clock and other high-speed lines
- FPGA power dissipation and cooling
- Mitigation of whisker growth in lead-free solder

FPGA-Level Design Practices

Synchronous vs. asynchronous design





Asynchronous/Combinatorial

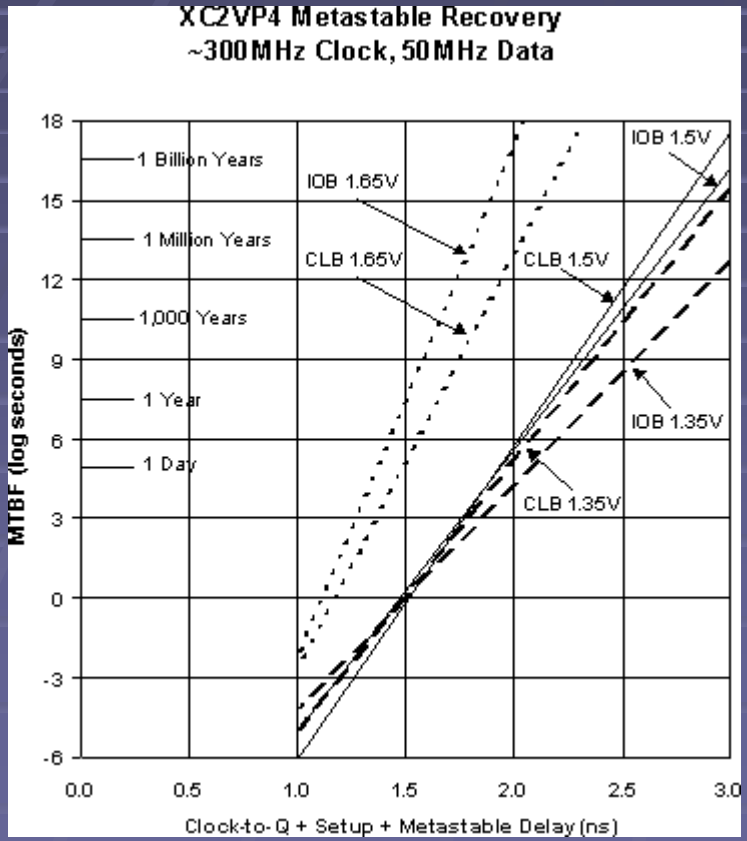
- Faster response, smaller design
- Prone to glitches, bus skews, timing issues

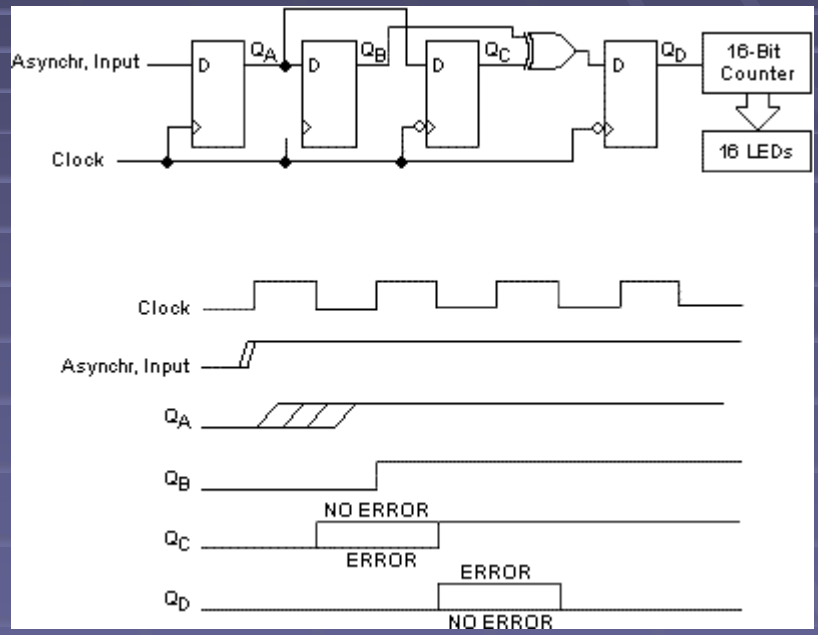
Synchronous/Sequential

- Slower but controlled response, larger design
- Eliminates glitches, bus skews, timing issues

FPGA-Level Design Practices

Metastability

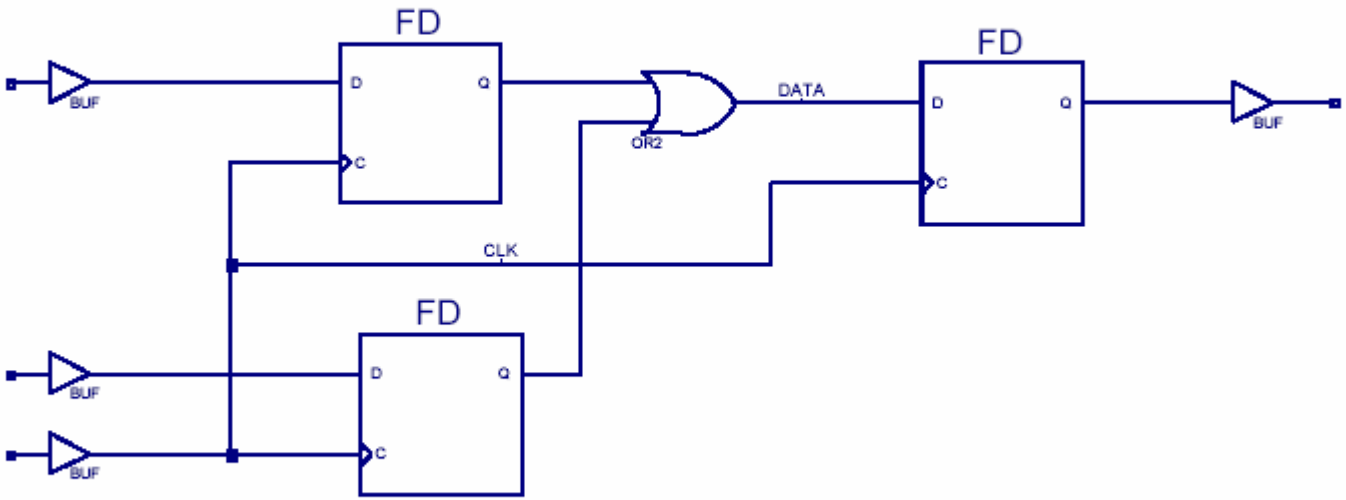




Source: Xilinx

FPGA-Level Design Practices

Global Routing Lines



FPGA-Level Design Practices

Global Routing Lines

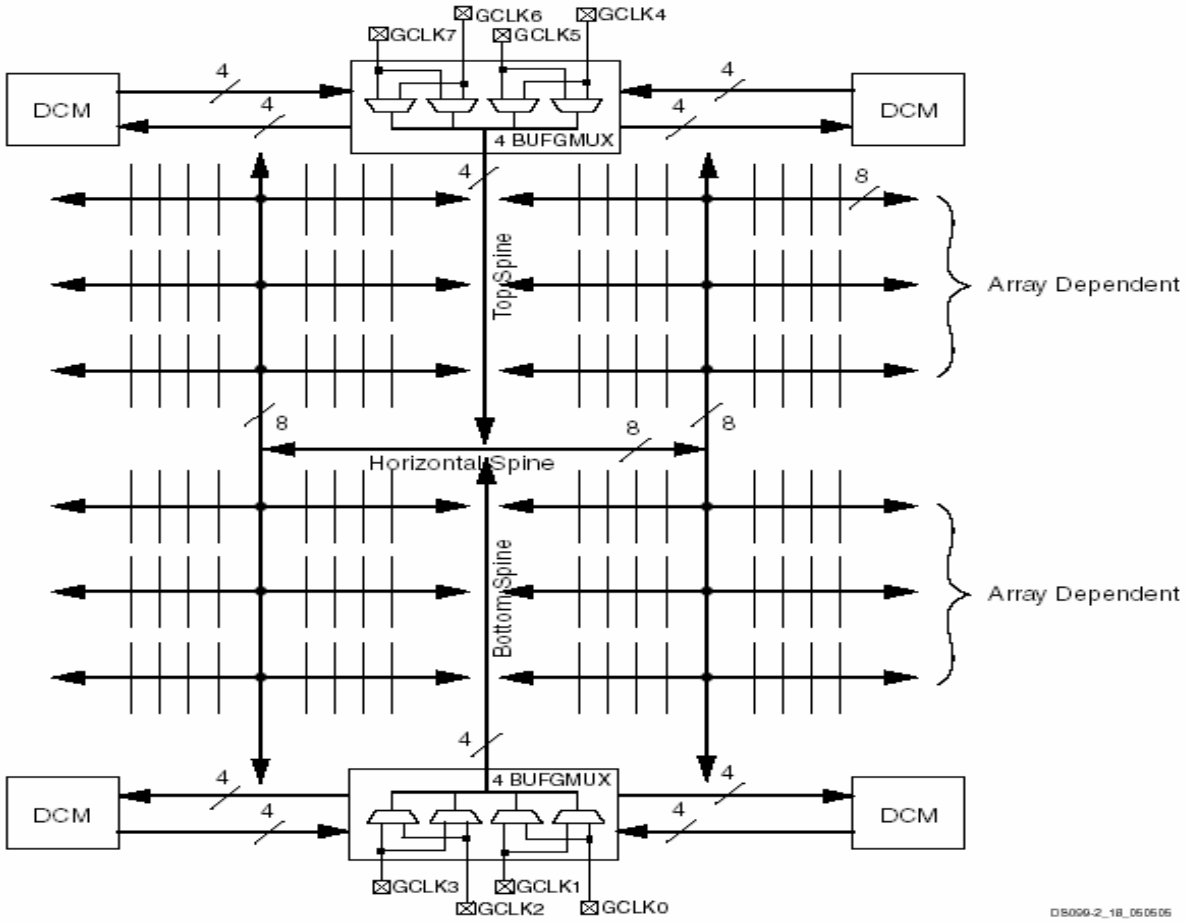
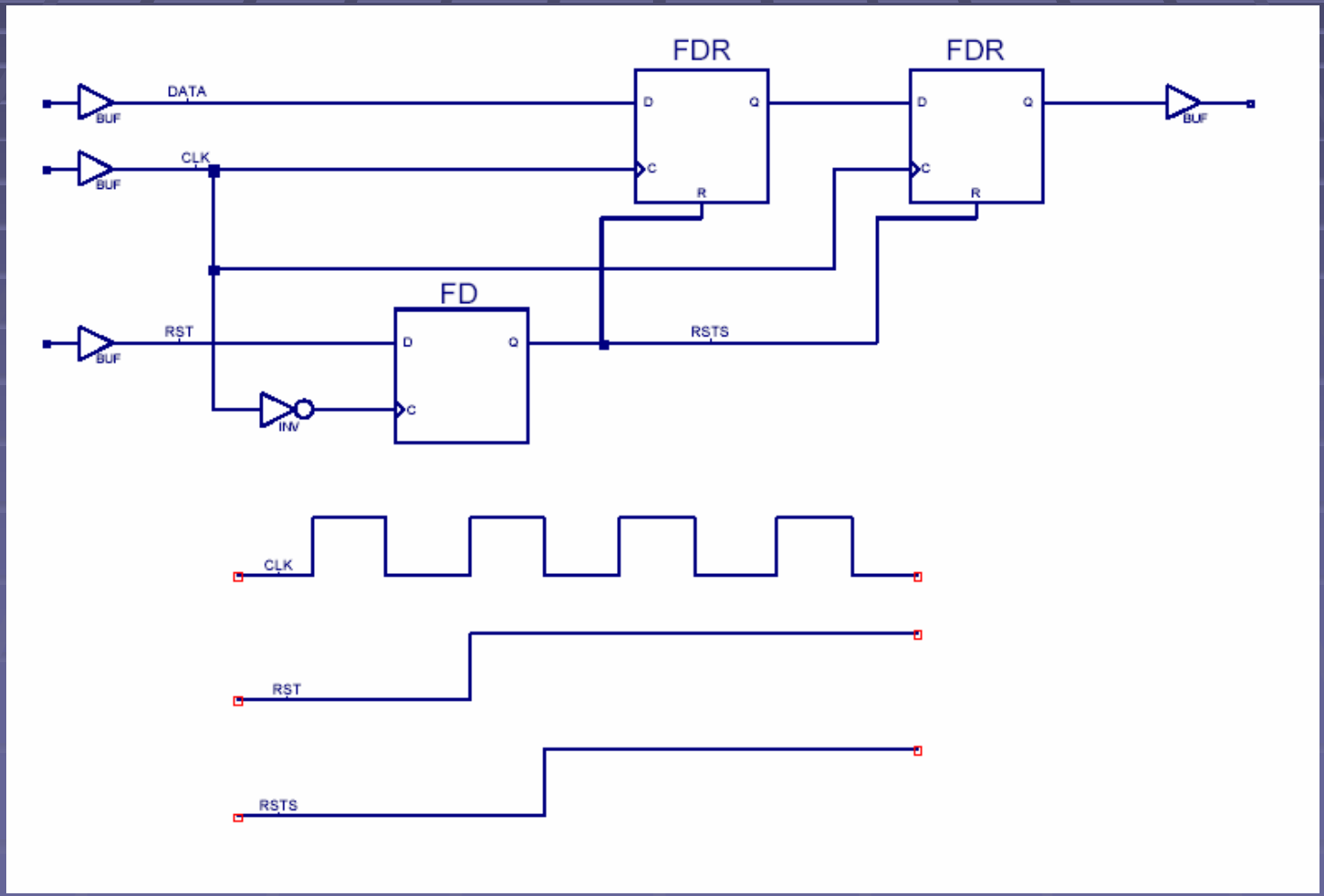


Figure 22: **Spartan-3 Clock Network (Top View)**

Source: Xilinx

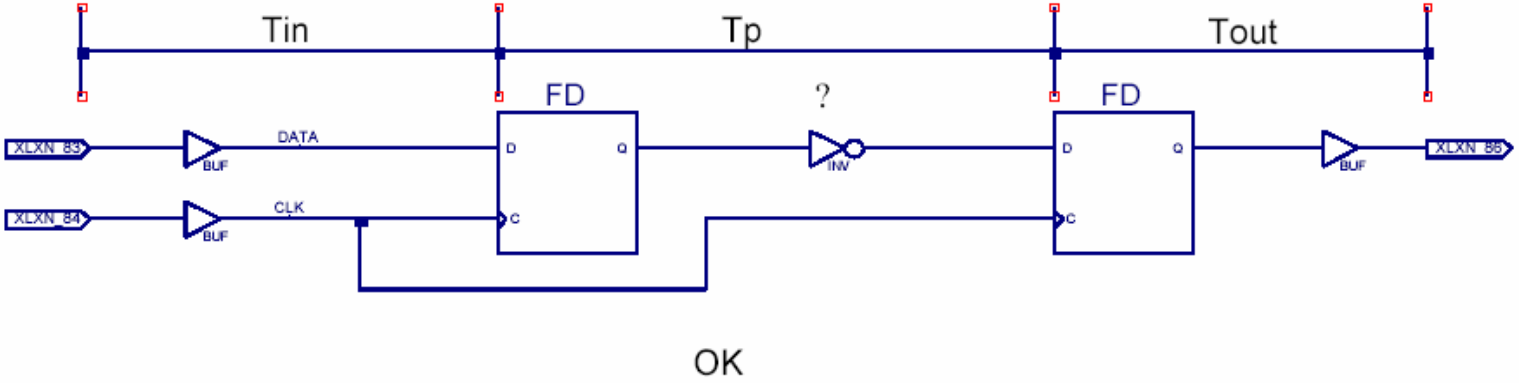
FPGA-Level Design Practices

FPGA Reset

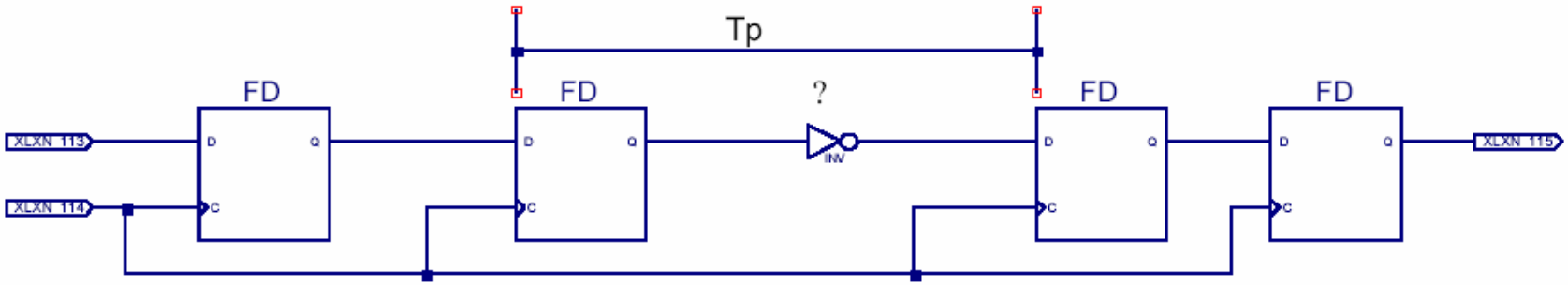


FPGA-Level Design Practices

Timing constraints



OK



BETTER

FPGA-Level Design Practices

HDL State Machine Design

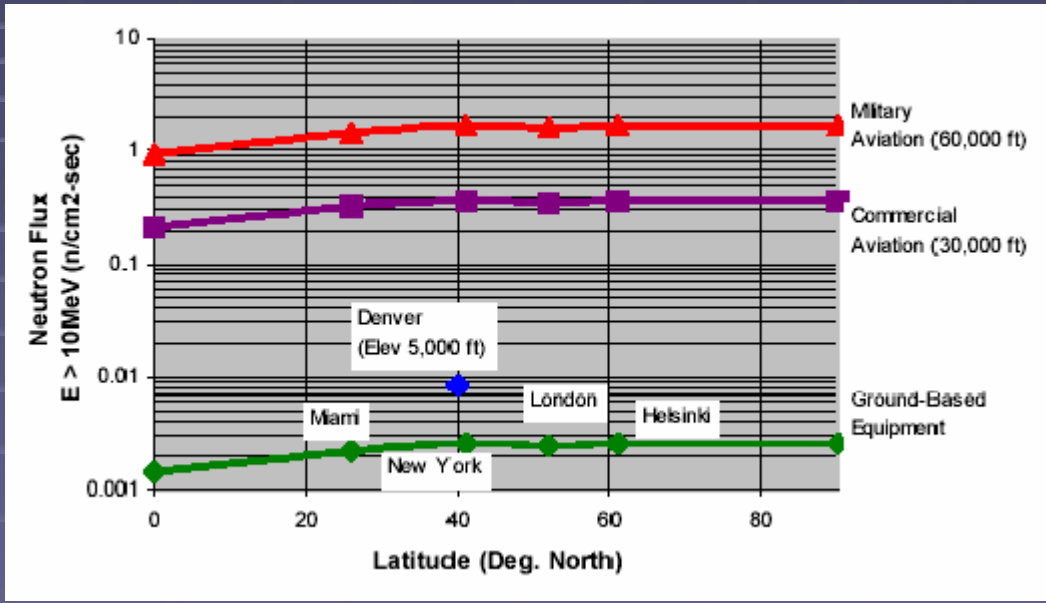
```
process(CLK,RST)

begin
     if RST = '0' or NCS = '1' then
       STATE <= "00";
       OUTPUT <= "00";
    elsif CLK'event and CLK = '1' then
       if STATE  =  "00" then
          if INPUT <= "00" then
            OUTPUT <= "00";
            STATE <= "00";
          else
            OUTPUT <= "01";
             STATE <= "01";
          end if;
       elsif STATE  =  "01" then
          if INPUT <= “10" then
            OUTPUT <= "01";
            STATE <= "01";
          else
             OUTPUT <= “10";
             STATE <= “10";
          end if;
```

```
       elsif STATE  =  “10" then
          if INPUT <= “11" then
            OUTPUT <= “10";
            STATE <= "00";
          else
            OUTPUT <= "00";
             STATE <= “10";
          end if;
=======================
       end if;
    end process;
```

```
       else
            OUTPUT <= “00";
            STATE <= "00";
```

FPGA-Level Design Practices

Single Event Upsets (SEU)



Neutron flux as a function of Altitude and Latitude

Process Technology	0 ft. Altitude
0.22µ	3.16E-05
0.18µ	5.02E-05
0.15µ	7.97E-05
0.13µ	1.26E-04

Number of SEU/day in a 1M gate SRAM FPGA

Source: Actel

FPGA-Level Design Practices

SEU Mitigation Techniques

- Using SEU immune or SEU tolerant FPGA devices
- Error Detection and Correction (EDAC)
- Module Redundancy
- Scheduled or Error-Initiated Reconfiguration
- Watchdog Timers

FPGA-Level Design Practices

SEU immune or tolerant FPGA devices

- One-time programmable FPGAs have the best immunity against SEU in their configuration memory
- Flash-based FPGAs offer some SEU immunity, but they can be reprogrammed
- SRAM-based FPGAs are most vulnerable to SEU, but they have largest gate counts
- All FPGAs are vulnerable to SEU in the user logic.
- Some FPGAs have triple redundant FFs built in

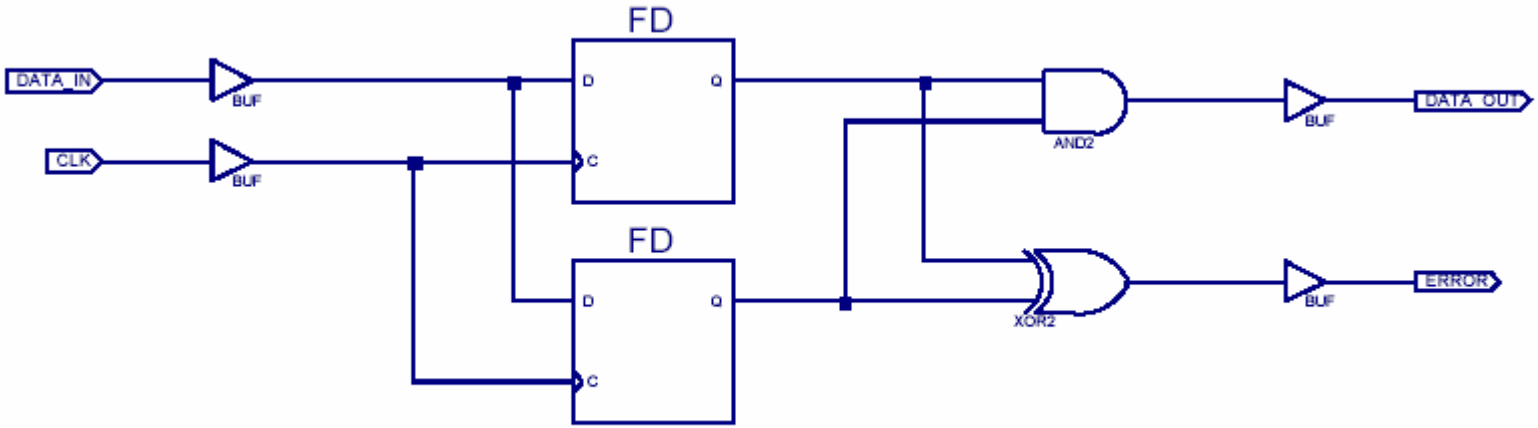
FPGA-Level Design Practices

Error Detection and Correction (EDAC)

- Parity bit
- Cyclic Redundancy Check (CRC)
- Hamming codes
- Reed-Solomon codes

FPGA-Level Design Practices

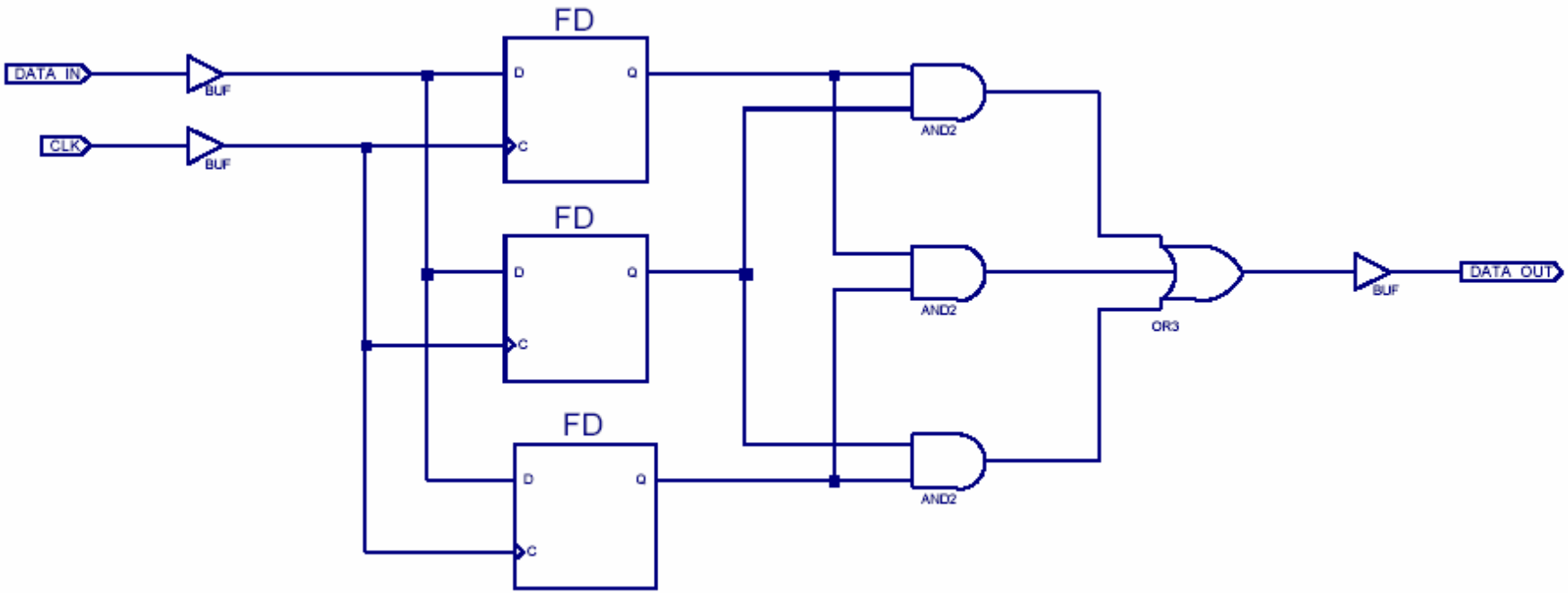
SEU Mitigation using Module Redundancy



Double Module Redundancy

FPGA-Level Design Practices

SEU Mitigation using Module Redundancy



Triple Module Redundancy

FPGA-Level Design Practices

Scheduled or Error-Initiated Reconfiguration

- Used in SRAM-based FPGAs
- FPGA gets preventively reconfigured, totally or partially when/if it can be safely halted for the duration of the reconfiguration process
- A logic inside an FPGA constantly runs SRAM configuration CRC, and initiates reconfiguration when detecting an error
- Other methods of error detection can be used to initiate reconfiguration

Validation and Verification of FPGA-based Systems

- Problems
  - Mainstream simulation and hardware verification used in non-safety design may not be appropriate for safety critical systems.
  - 100 % simulation coverage most often not practically possible
  - Formal verification methods and tools not mature enough

- Possible solutions
  - Simple designs allowing 100% coverage
  - Partitioning of large designs
  - Certification of EDA tools
  - Insertion-based verification
  - Hardware accelerated testing

Maintainability and Obsolescence of FPGA-based Systems

- Easier to maintain then software-based systems, but more involving than hardware-only systems
- Program retention from 10 to 20 years in SRAM- and FLASH-based FPGAs
- One-time FPGAs may experience increase in the resistance of their programmed interconnects over time
- Software tools are being upgraded on yearly basis, and they may not always be completely backward compatible.

Conclusions

- FPGA-based safety-critical systems require a specific set of design rules at the system level, at the board level, and the component level
- V&V of FPGA-based systems need to be adapted to include the safe FPGA design practices
- FPGAs need to be treated as hardware/software system throughout the design life cycle as well as during the licensing process
- Safe FPGA design practices need to be regulated so that I&C manufacturers, nuclear plants and regulatory authorities use the same reference