FPGA Design Techniques

© 2002 Xilinx, Inc. All Rights Reserved

Objectives

After completing this module, you will be able to:

- Increase design performance by duplicating flip-flops
- Increase design performance by adding pipeline stages
- Increase board performance by using I/O flip-flops
- Build reliable synchronization circuits

© 2002 Xilinx, Inc. All Rights Reserved

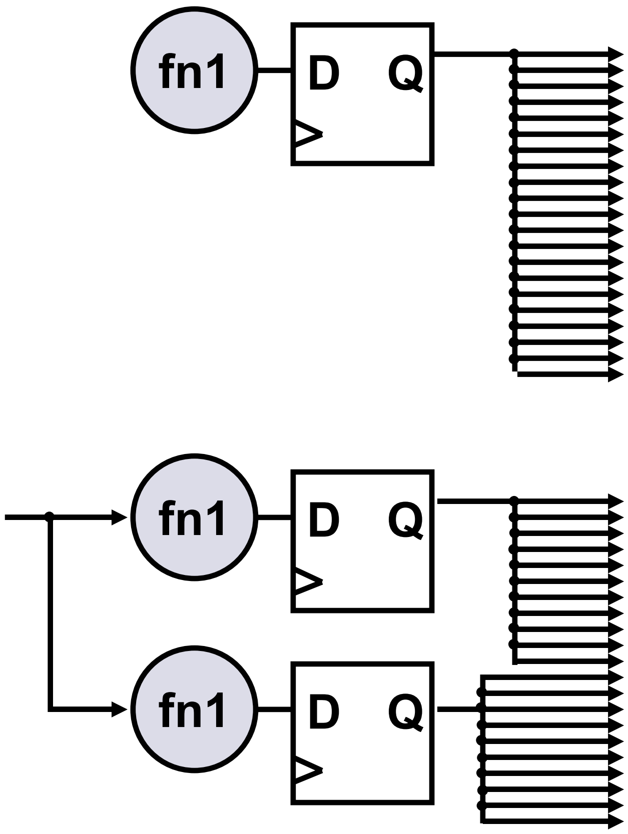
Outline

- **Duplicating Flip-flops**
- Pipelining
- I/O Flip-flops
- Synchronization Circuits
- Summary

© 2002 Xilinx, Inc. All Rights Reserved

Duplicating Flip-Flops

- **High-fanout nets can be slow and hard to route**
- **Duplicating flip-flops can fix both problems**
  - Reduced fanout shortens net delays
  - Each flip-flop can fanout to a different physical region of the chip to help with routing
- **Design tradeoffs**
  - Gain routability and performance
  - Design area increases



© 2002 Xilinx, Inc. All Rights Reserved

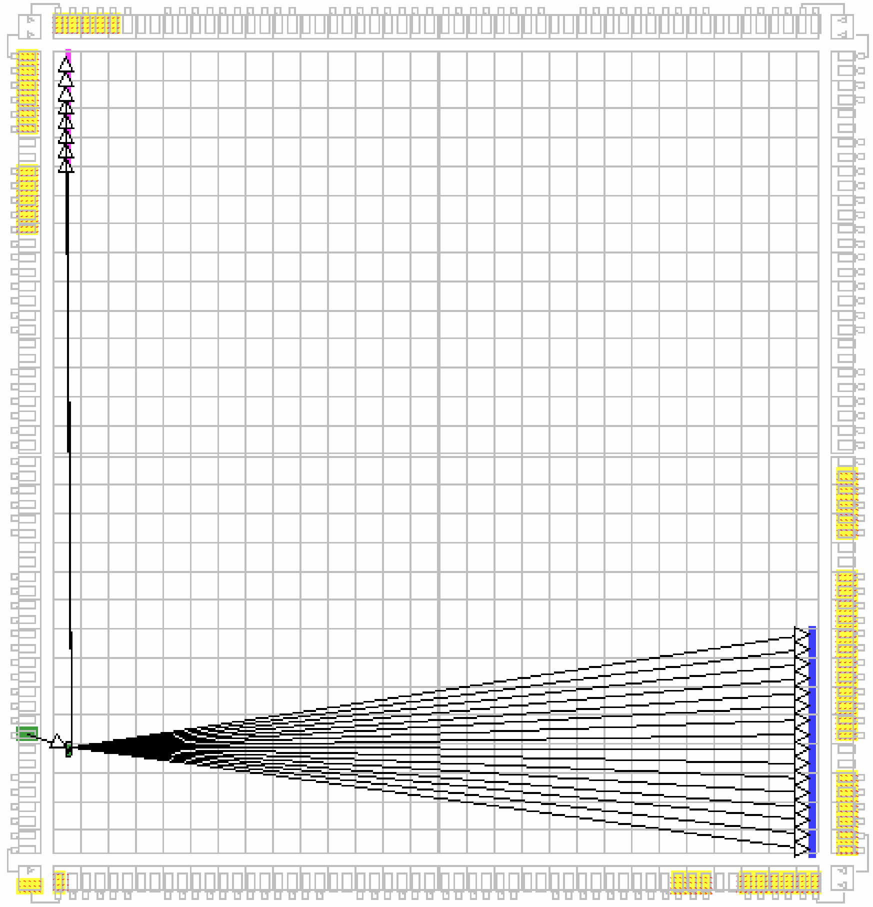
Tips on Duplicating Flip-Flops

- **Name duplicated flip-flops _a, _b: *NOT* _1, _2**
  - Numbered flip-flops are mapped by default into the same slice
  - You generally want duplicated flip-flops to be separated
    - Especially if the loads are spread across the chip
- **Explicitly create duplicate flip-flops in your HDL code**
  - Most synthesis tools have automatic fanout-control features
    - However, they do not always pick the best division of loads
    - Also, duplicated flip-flops will be named _1, _2
  - Many synthesis tools will optimize-out duplicated flip-flops
    - Tell your synthesis tool to keep redundant logic
- **Do not duplicate flip-flops that are sourced by asynchronous signals**
  - Synchronize the signal first
  - Feed synchronized signal to multiple flip-flops

© 2002 Xilinx, Inc. All Rights Reserved

Duplicating Flip-Flops Example

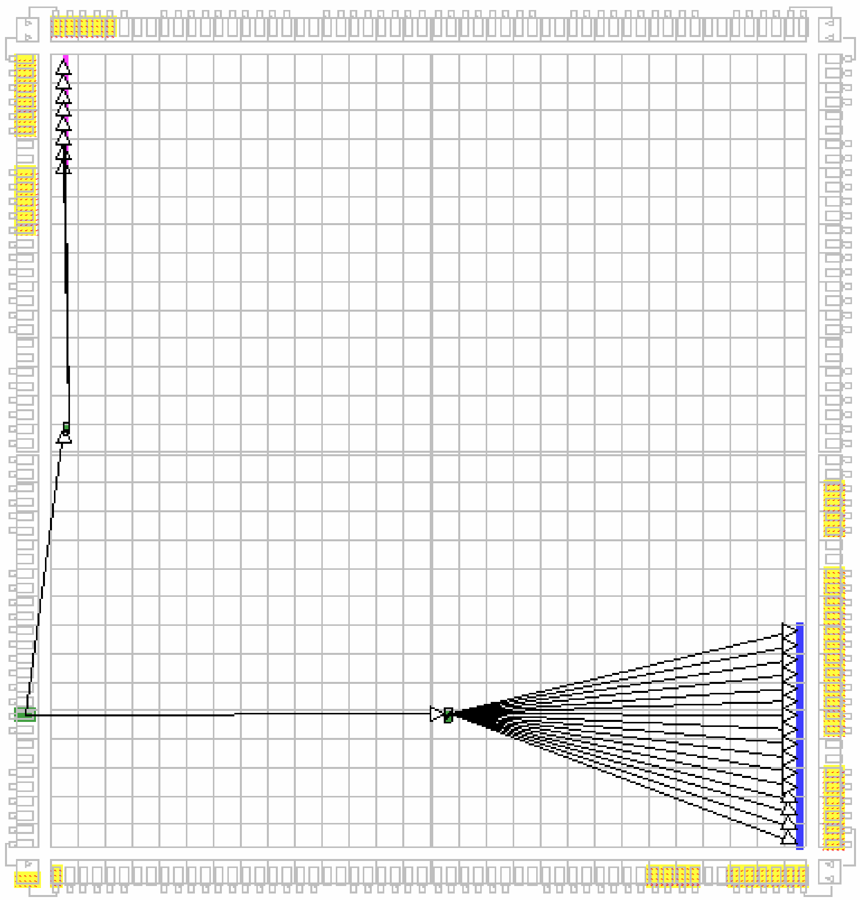
- Source flip-flop drives two register banks that are constrained to different regions of the chip
- Source flip-flop and pad are not constrained
- PERIOD = 5 ns timing constraint
- Implemented with default options
- Longest path = 6.806 ns
  - Fails to meet timing constraint



© 2002 Xilinx, Inc. All Rights Reserved

Duplicating Flip-Flops Example

- **Source flip-flop has been duplicated**
- **Each flip-flop drives a region of the chip**
  - Each flip-flop can be placed closer to the register that it is driving
  - Shorter routing delays
- **Longest path = 4.666 ns**
  - Meets timing constraint



© 2002 Xilinx, Inc. All Rights Reserved

Outline

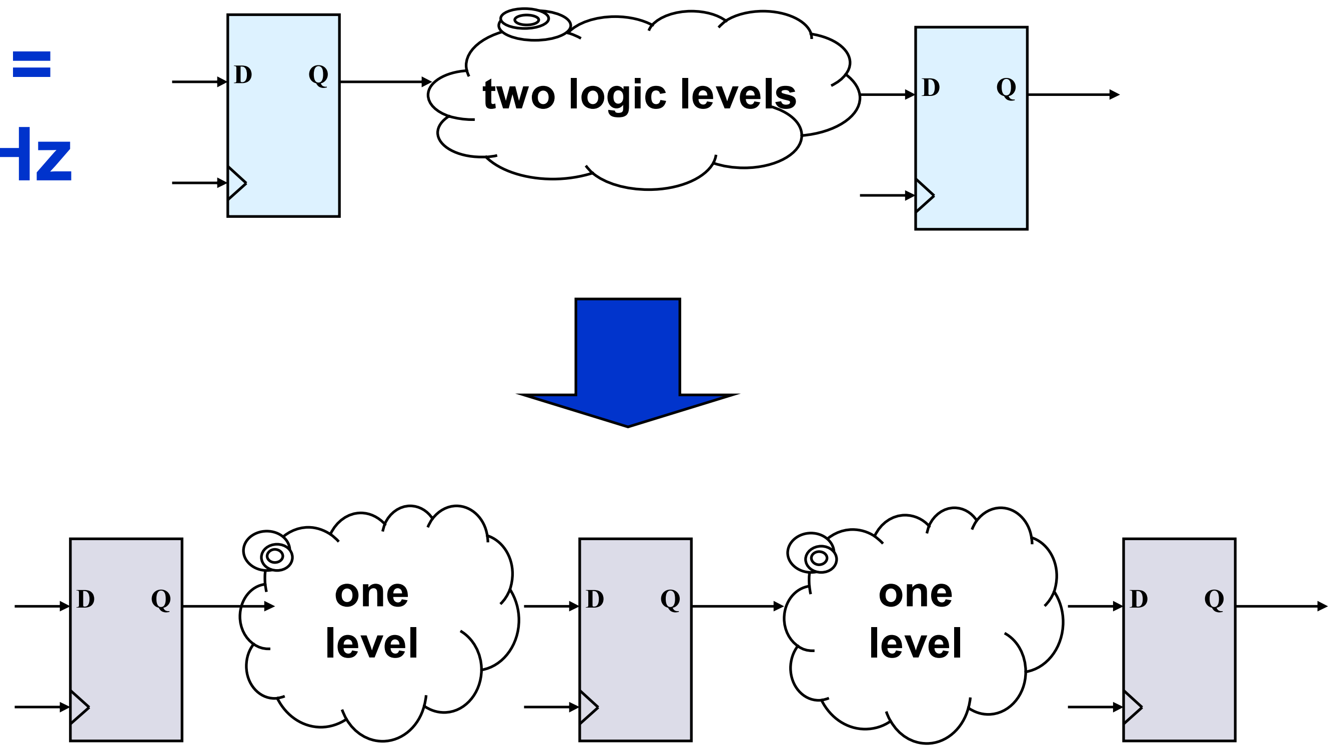
- Duplicating Flip-Flops
- **Pipelining**
- I/O Flip-flops
- Synchronization Circuits
- Summary

© 2002 Xilinx, Inc. All Rights Reserved

Pipelining Concept

f_{MAX} = *n* MHz

$f_{MAX} \approx$ 2*n* MHz



© 2002 Xilinx, Inc. All Rights Reserved

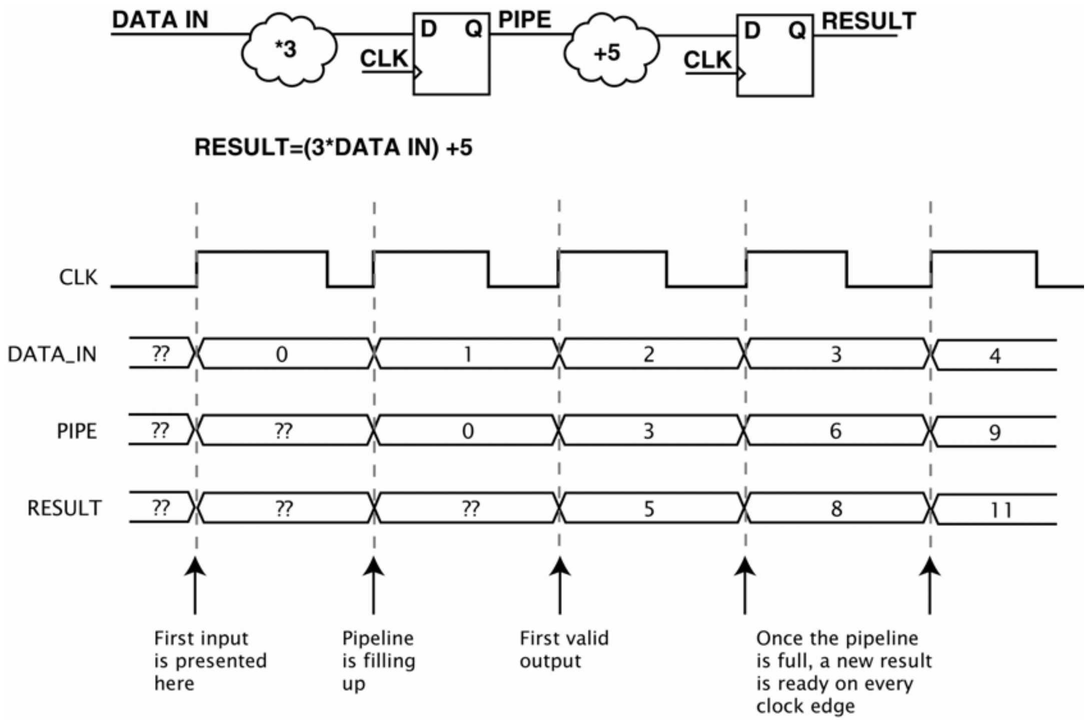
Pipelining Considerations

- **Are there enough flip-flops available?**
  - Refer to the Map Report
  - You generally will not run out of FFS
- **Are there multiple logic levels between flip-flops?**
  - If there is only one logic level between flip-flops, pipelining will not improve performance
    - Exception: Long carry-logic chains can benefit from pipelining
  - Refer to the Post-Map Static Timing Report or Post-Place & Route Static Timing Report
- **Can the system tolerate *latency*?**

© 2002 Xilinx, Inc. All Rights Reserved

Latency in Pipelines

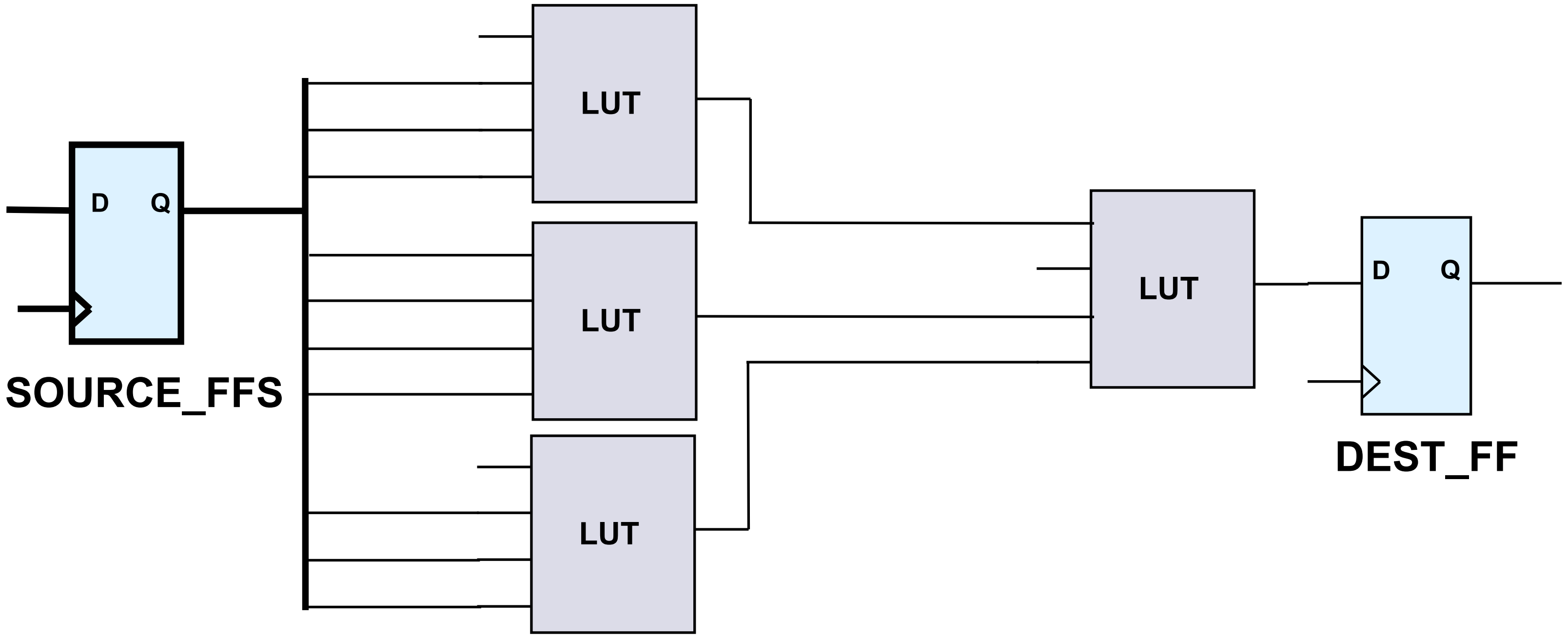
- **Each pipeline stage adds one clock cycle of delay before the first output will be available**
  - Also called “filling the pipeline”
- **After the pipeline is filled, a new output is available every clock cycle**



© 2002 Xilinx, Inc. All Rights Reserved

Pipelining Example

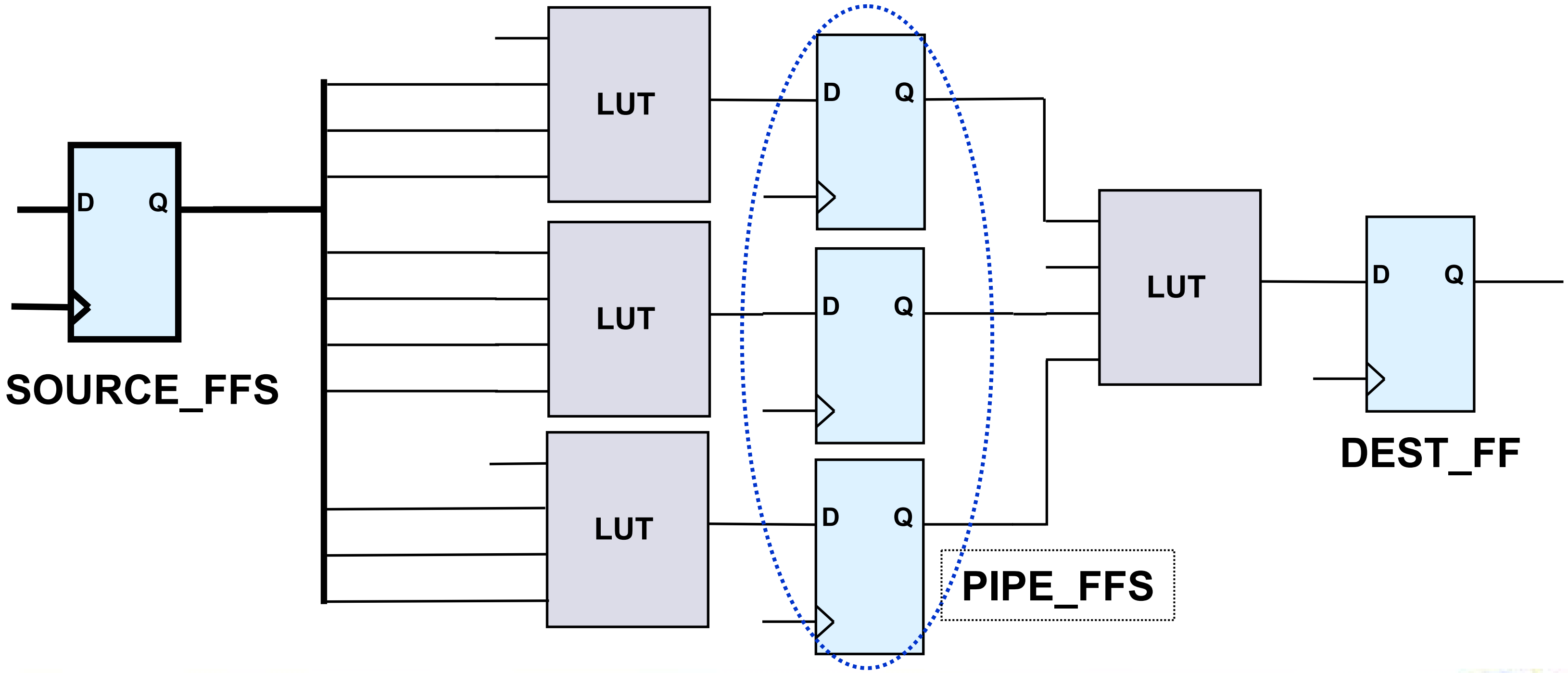
- **Original circuit**
  - Two logic levels between SOURCE_FFS and DEST_FF
  - f_{MAX} = ~207 MHz



© 2002 Xilinx, Inc. All Rights Reserved

Pipelining Example

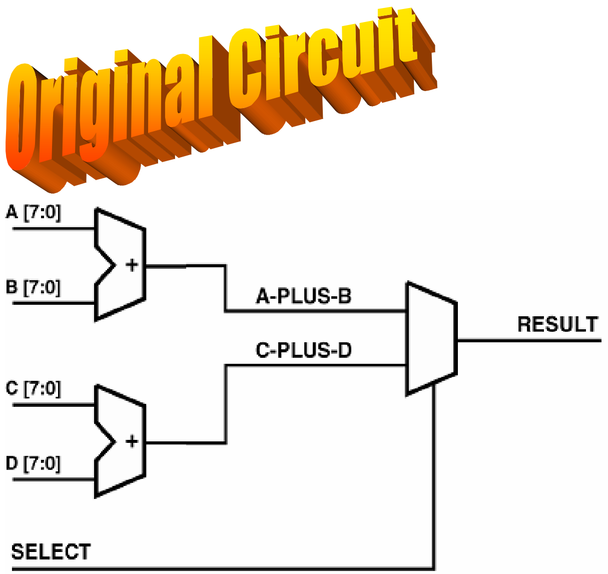
- **Pipelined circuit**
  - One logic level between each set of flip-flops
  - f_{MAX} = ~347 MHz

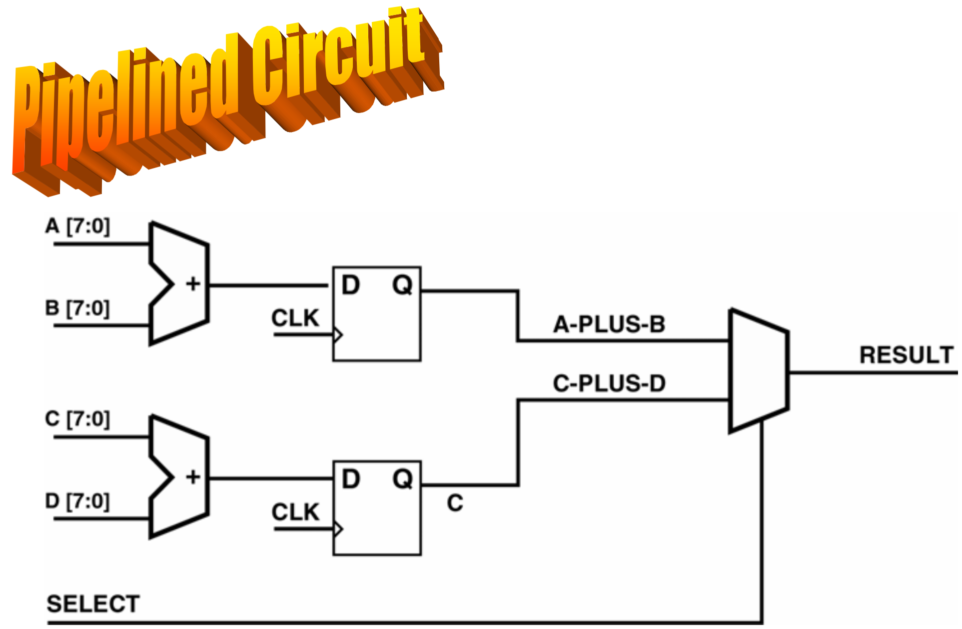


© 2002 Xilinx, Inc. All Rights Reserved

Pipelining Questions

- Given the original circuit, what is wrong with the pipelined circuit?
- How can the problem be corrected?

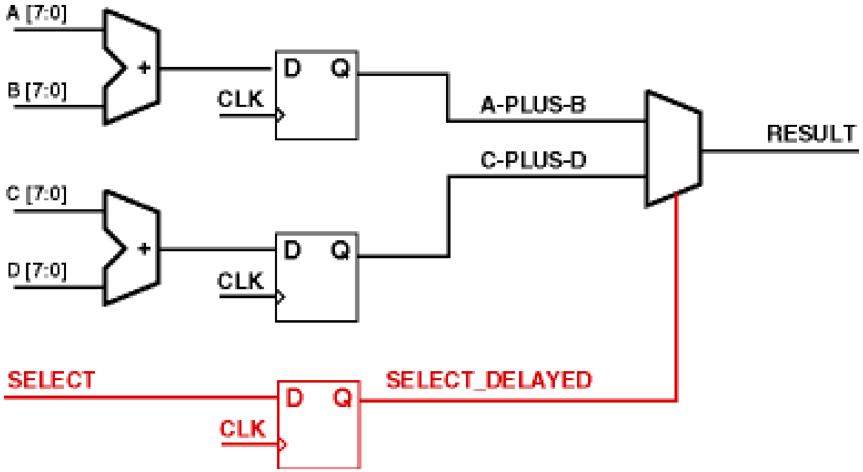




© 2002 Xilinx, Inc. All Rights Reserved

Pipelining Answers

- **What is wrong with the pipelined circuit?**
  - Latency mismatch
  - Older data is mixed with newer data
  - Circuit output is incorrect
- **How can the problem be corrected?**
  - Add a flip-flop on *SELECT*
  - All data inputs now experience the same amount of latency



© 2002 Xilinx, Inc. All Rights Reserved

Outline

- Duplicating Flip-Flops
- Pipelining
- **I/O Flip-flops**
- Synchronization Circuits
- Summary

© 2002 Xilinx, Inc. All Rights Reserved

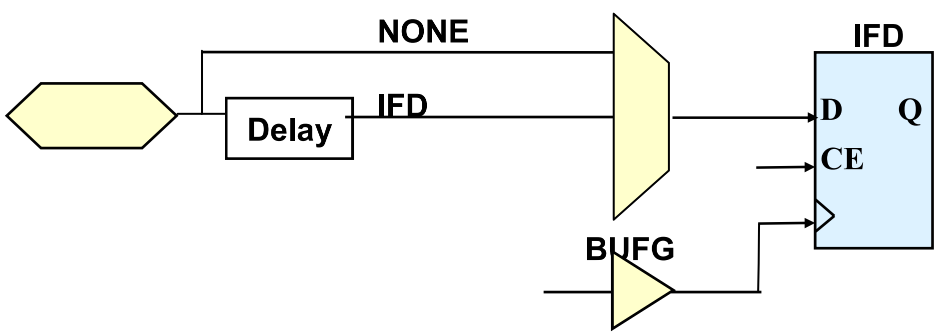
I/O Flip-Flop Overview

- **Each Virtex™-II I/O Block contains six flip-flops**
  - IFD on the input, OFD on the output, ENBFF on the 3-state enable
  - Single or Double Data Rate support
- **Benefits of using I/O flip-flops**
  - Guaranteed setup, hold, and clock-to-out times when the clock signal comes from a BUFG
  - Programmable delay on input flip-flop
  - Programmable slew rate on output flip-flop

© 2002 Xilinx, Inc. All Rights Reserved

Programmable Input Delay

- **Delay is added to the D input of the IFD or ILD to guarantee 0 ns hold time**
  - Trades an increase in setup time for a decrease in hold time
- **Delay is controlled by the IOBDELAY attribute**
  - NONE is the default if CLK is driven by a DCM
  - IFD is the default if CLK is not driven by a DCM



© 2002 Xilinx, Inc. All Rights Reserved

Programmable Input Delay Example

- **XC2V250-5 (v1.90 Virtex™-II speed files)**
  - CLK is a DCM output (default delay is NONE)
    - IFD t_{SU} = 4.0 ns t_H = 0.0 ns
    - NONE t_{SU} = 1.3 ns t_H = 0.0 ns
  - CLK is not a DCM output (default delay is IFD)
    - IFD t_{SU} = 1.2 ns t_H = 0.0 ns
    - NONE t_{SU} = -1.5 ns t_H = 1.7 ns

© 2002 Xilinx, Inc. All Rights Reserved

Programmable Output Slew Rate

- **Only the LVTTL and LVCMOS I/O standards support programmable slew rate**
- **Slew rate can be SLOW or FAST**
  - By default, the slew rate is SLOW
- **Slew rate is controlled by attributes**
  - Specified in the Constraints Editor or the UCF file

© 2002 Xilinx, Inc. All Rights Reserved

Accessing I/O Flip-Flops

- **During synthesis**
  - Timing-driven synthesis can force flip-flops into IOBs
  - Some tools support attributes or synthesis directives to mark flip-flops for placement in an IOB
- **Xilinx Constraint Editor**
  - Select the Misc tab
  - Specify individual registers that should be placed in the IOB
    - Need to know the instance name of each register
- **During the Map phase of implementation**
  - Map Properties, Pack I/O Registers/Latches into IOBs is on by default
  - Timing-driven packing will also move registers into IOBs for critical paths

© 2002 Xilinx, Inc. All Rights Reserved

Outline

- Duplicating Flip-Flops
- Pipelining
- I/O Flip-flops
- **Synchronization Circuits**
- Summary

© 2002 Xilinx, Inc. All Rights Reserved

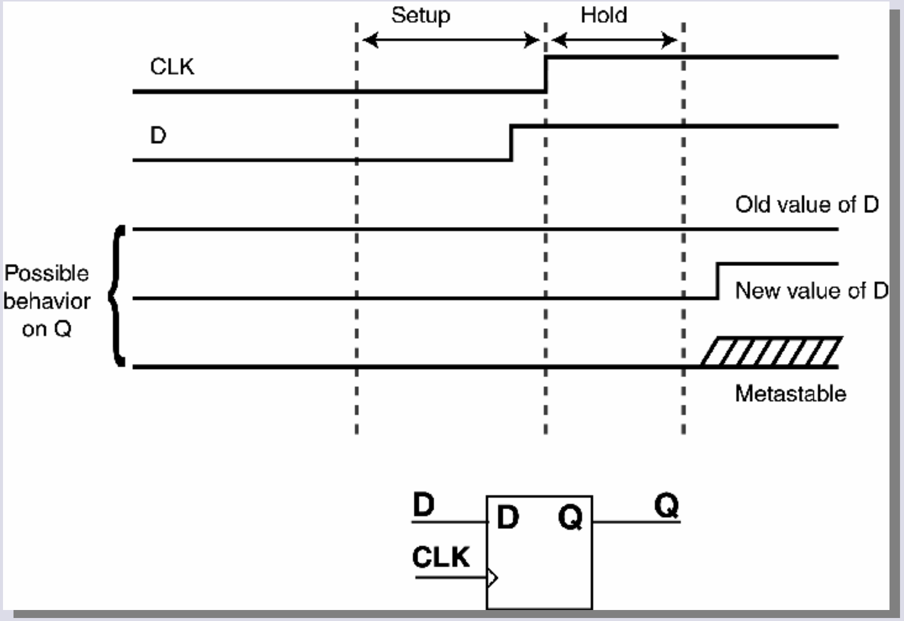
Synchronization Circuits

- **What is a synchronization circuit?**
  - Captures an asynchronous input signal and outputs it on a clock edge
- **Why do I need synchronization circuits?**
  - Prevent setup and hold time violations
  - End result is a more reliable design
- **When do I need synchronization circuits?**
  - Signals cross between *unrelated* clock domains
    - Between related clock domains, a path-specific timing constraint is sufficient
  - Chip inputs that are asynchronous

© 2002 Xilinx, Inc. All Rights Reserved

Setup and Hold Time Violations

- **Violations occur when the flip-flop input changes too close to a clock edge**
- **Three possible results:**
  - Flip-flop clocks in old data value
  - Flip-flop clocks in new data value
  - Flip-flop output becomes *metastable*



© 2002 Xilinx, Inc. All Rights Reserved

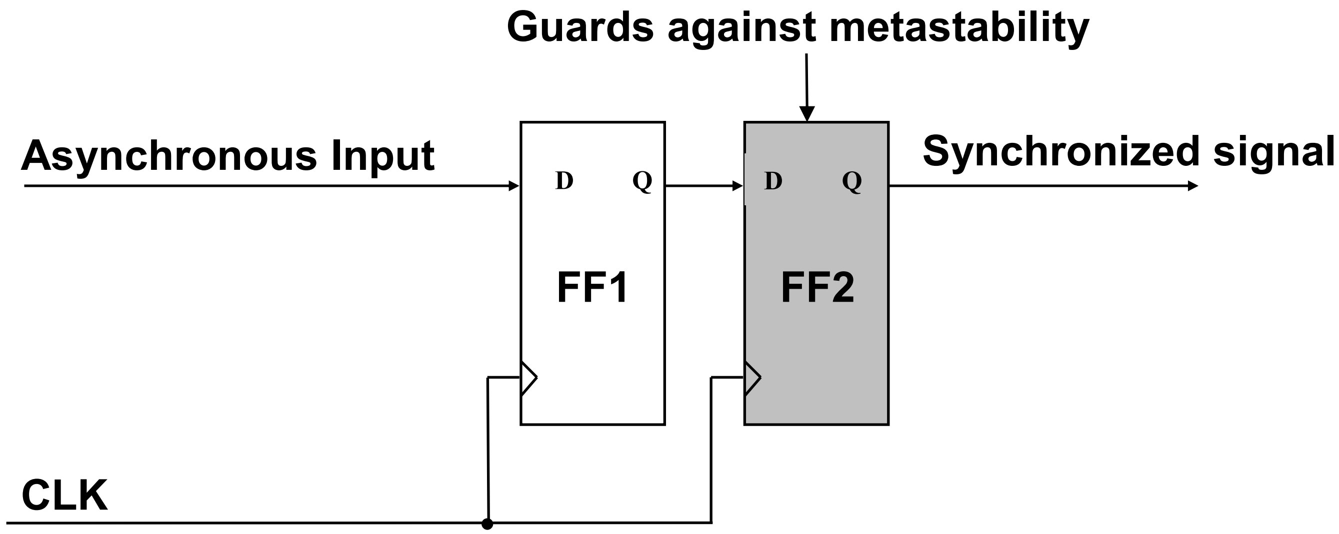
Metastability

- **Flip-flop output enters a transitory state**
  - Neither a valid 0 nor a valid 1
    - May be interpreted as 0 by some loads and as 1 by others
  - Remains in this state for an unpredictable length of time before settling to a valid 0 or 1
- **Due to its statistical nature, the occurrence of metastable events can only be reduced, not eliminated**
- **Mean Time Between Failures (MTBF) is exponentially related to the length of time the flip-flop is given to recover**
  - A few extra ns of recovery time can dramatically reduce the chances of a metastable event
- **The circuits shown in this section allow a full clock cycle for metastable recovery**

© 2002 Xilinx, Inc. All Rights Reserved

Synchronization Circuit #1

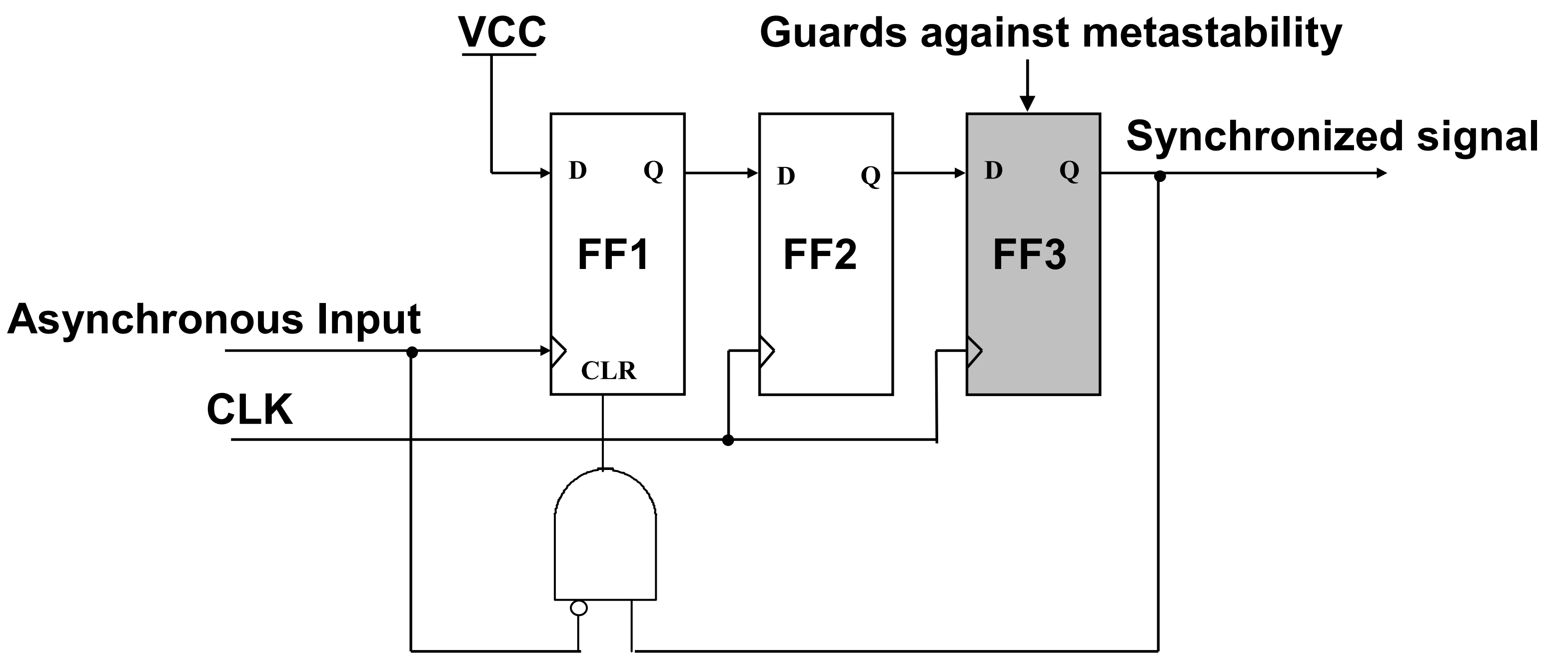
- Use when input pulses will always be at least one clock period wide
- The "extra" flip-flops guard against metastability



© 2002 Xilinx, Inc. All Rights Reserved

Synchronization Circuit #2

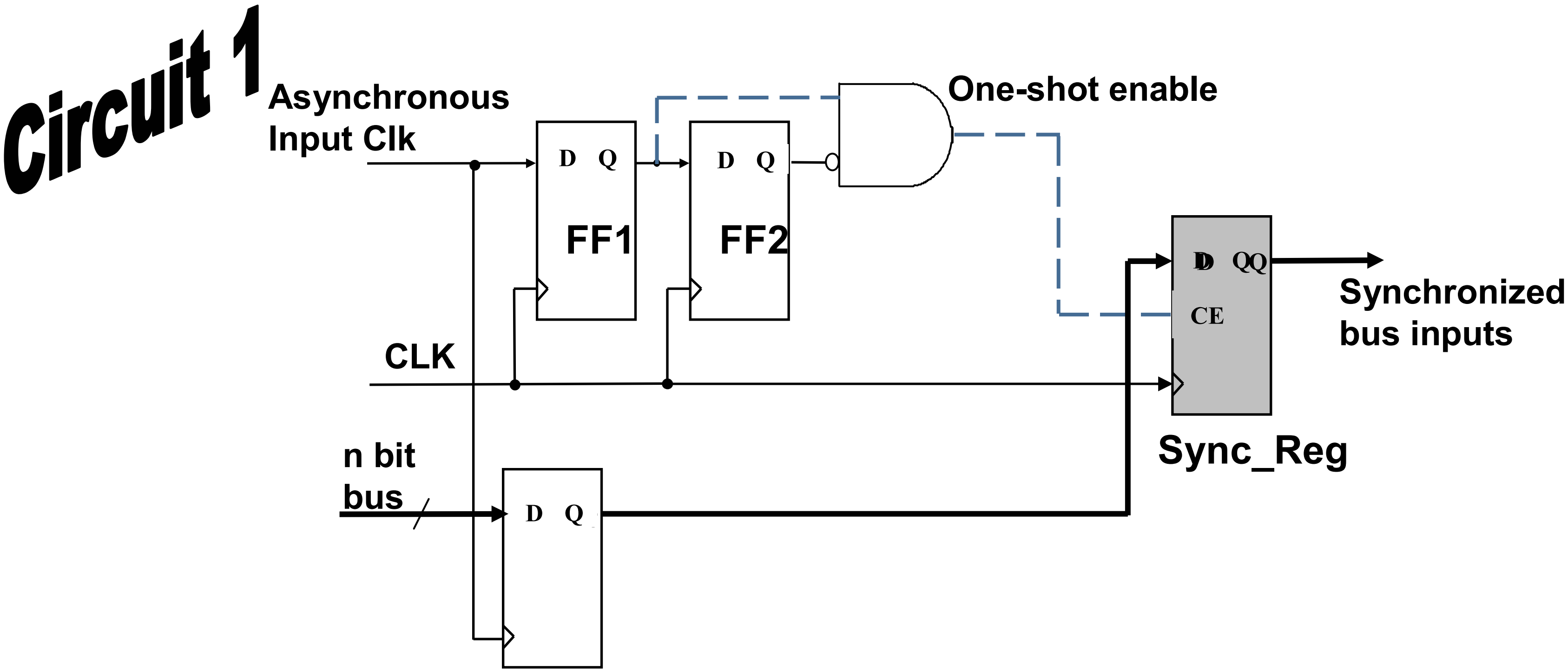
- **Use when input pulses may be less than one clock period wide**
  - FF1 captures short pulses



© 2002 Xilinx, Inc. All Rights Reserved

Capturing a Bus

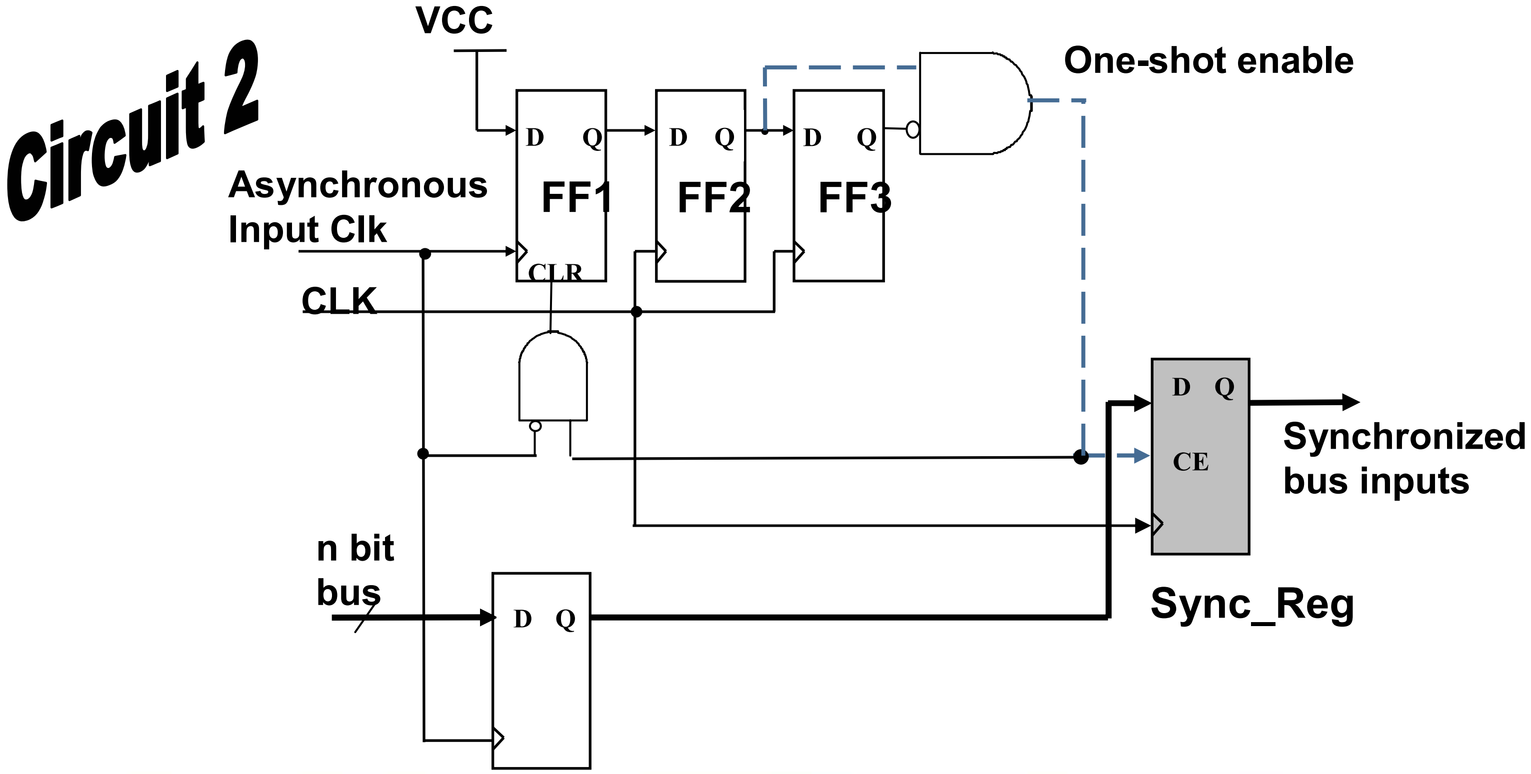
- **Leading edge detector**
  - Input pulses must be at least one CLK period wide



© 2002 Xilinx, Inc. All Rights Reserved

Capturing a Bus

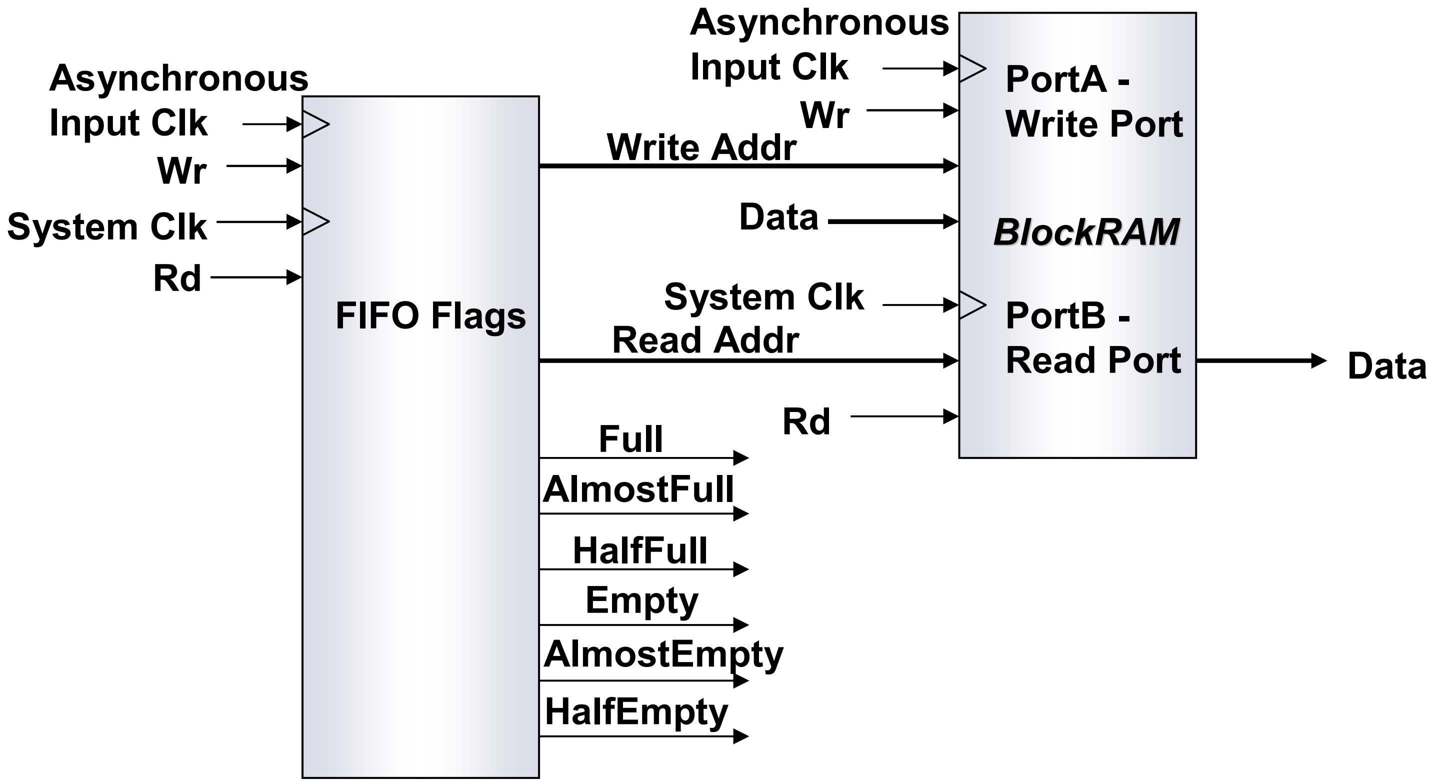
- **Leading edge detector**
  - Input pulses may be less than one CLK period wide



© 2002 Xilinx, Inc. All Rights Reserved

Synchronization Circuit #3

- Use a FIFO to cross domains



© 2002 Xilinx, Inc. All Rights Reserved

Outline

- Duplicating Flip-Flops
- Pipelining
- I/O Flip-flops
- Synchronization Circuits
- **Summary**

© 2002 Xilinx, Inc. All Rights Reserved

Review Questions

- High fanout is one reason to duplicate a flip-flop. What is another reason?
- Give an example of a time when you do not need to resynchronize a signal that crosses between clock domains
- What is the purpose of the “extra” flip-flop in the synchronization circuits shown in this module?

© 2002 Xilinx, Inc. All Rights Reserved

Answers

- **High fanout is one reason to duplicate a flip-flop. What is another reason?**
  - Loads are divided between multiple locations on the chip
- **Give an example of a time when you do not need to resynchronize a signal that crosses between clock domains**
  - Well-defined phase relationship between the clocks
  - Example: Clocks are the same frequency, 180º out of phase
  - Use a path-specific constraint to ensure that data paths will meet timing
- **What is the purpose of the “extra” flip-flop in the synchronization circuits shown in this module?**
  - To allow the first flip-flop time to recover from metastability

© 2002 Xilinx, Inc. All Rights Reserved

Summary

- **You can increase circuit performance by:**
  - Duplicating flip-flops
  - Adding pipeline stages
  - Using I/O flip-flops
- **Some tradeoffs**
  - Duplicating flip-flops increases circuit area
  - Pipelining introduces latency
  - I/O flip-flop programmable delay trades off setup time and hold time
- **Synchronization circuits increase reliability**

© 2002 Xilinx, Inc. All Rights Reserved

Where Can I Learn More?

- **Data Book: http://support.xilinx.com → Documentation → Databook**
  - Switching Characteristics
  - Detailed Functional Description > Input/Output Blocks (IOBs)
- **Application Notes: http://support.xilinx.com → Documentation → App Notes**
  - XAPP094: Metastability Recovery
  - XAPP225: Data to Clock Phase Alignment

© 2002 Xilinx, Inc. All Rights Reserved