Sharif University of Technology
Department of Computer Engineering

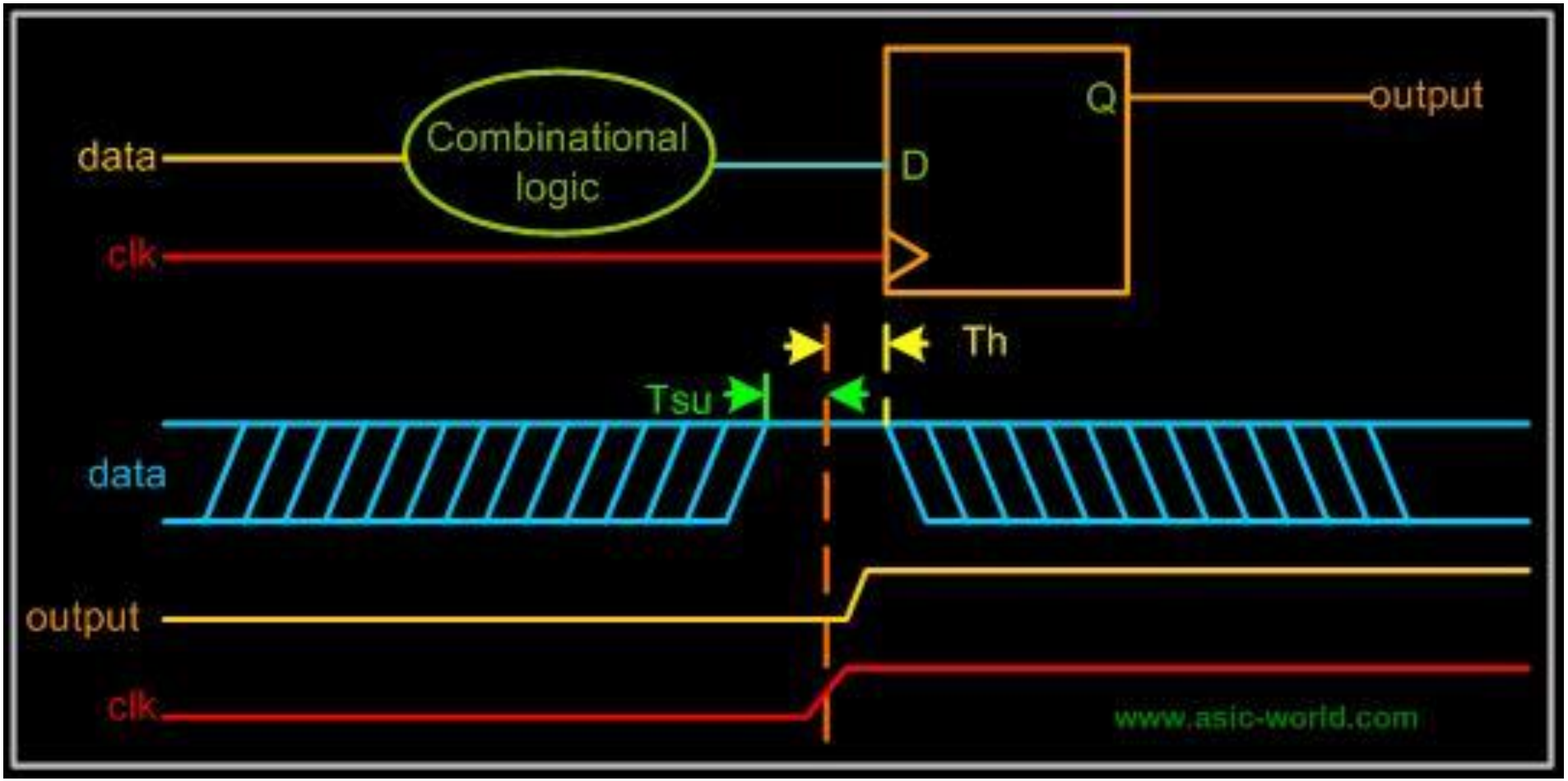
Digital System Design

Metastability &
Clock Domain Crossing (CDC)

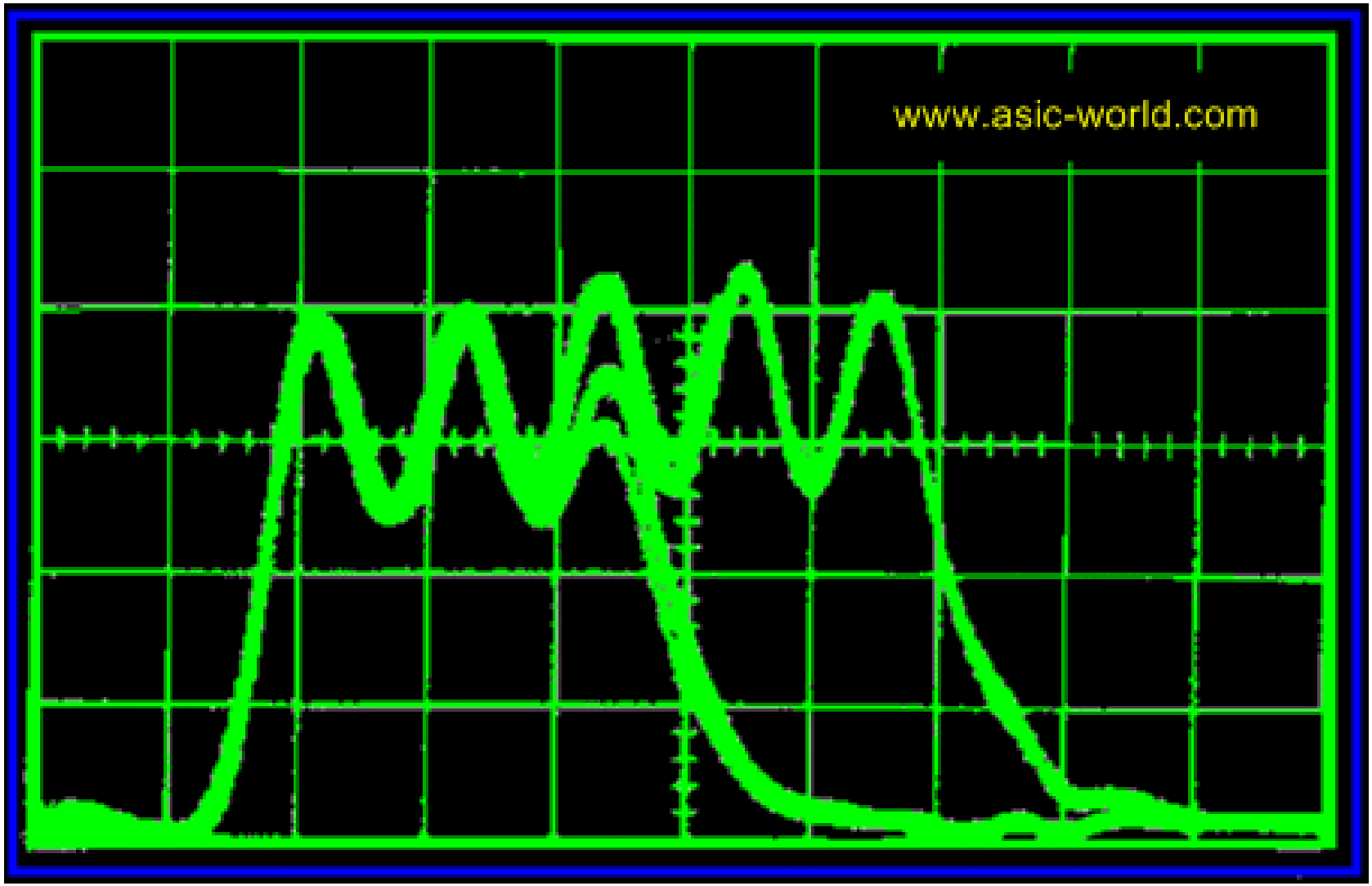
Siavash Bayat-Sarmadi

Introduction

- Flip-flop setup & hold times



Metastability



Metastable State

- A state for a flip-flop with unpredictable output
  - Quasi stable state
  - Output oscillating between 0 and 1
  - Taking longer time to settle down than usual
  - As an outcome of timing violations
    - Setup or hold issues
      - E.g., d/reset and clock changing at the same time

What are the causes? (1)

- Asynchronous input signal that can change at any time
  - Not in designer's control
- Too much clock skew
  - Rise and fall time more than the tolerable values

What are the causes? (2)

- Data input changing in the critical window
  - Critical window = setup + hold window
  - As an outcome of long combinational delay
- When interfacing two clock domains
  - Different frequencies
  - Same frequency, different phases
  - Discussed later in this lecture

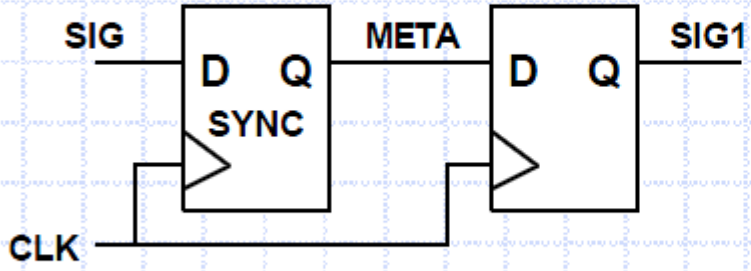
How to tolerate? (1)

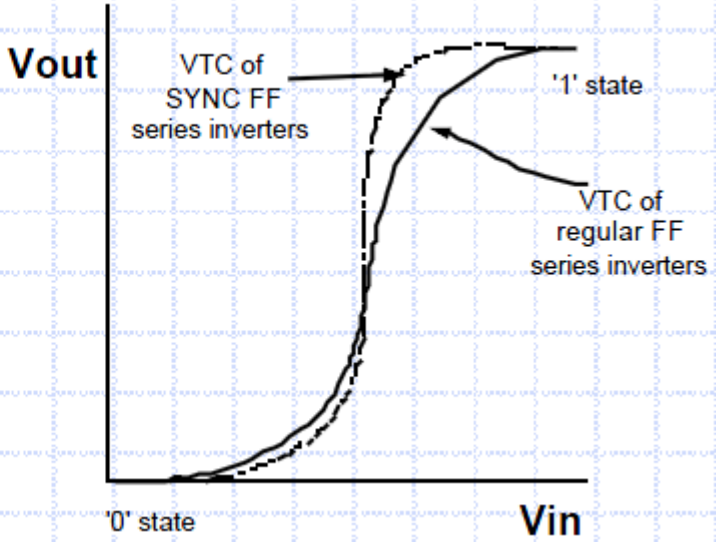
- Long enough clock period
  - For the quasi state to settle down
  - And for the follow-up path to the next flip-flop
  - Performance degradation, not always acceptable
- Metastable-hardened flip-flops
  - Reduced setup and hold times as much as possible

How to tolerate? (2)

SYNC Flip Flop

- SYNC Flip Flops are available in some ASIC libraries
  - Better MTBF characteristics due to high gain in the feedback path
  - Very large (5x regular FF) and very high power

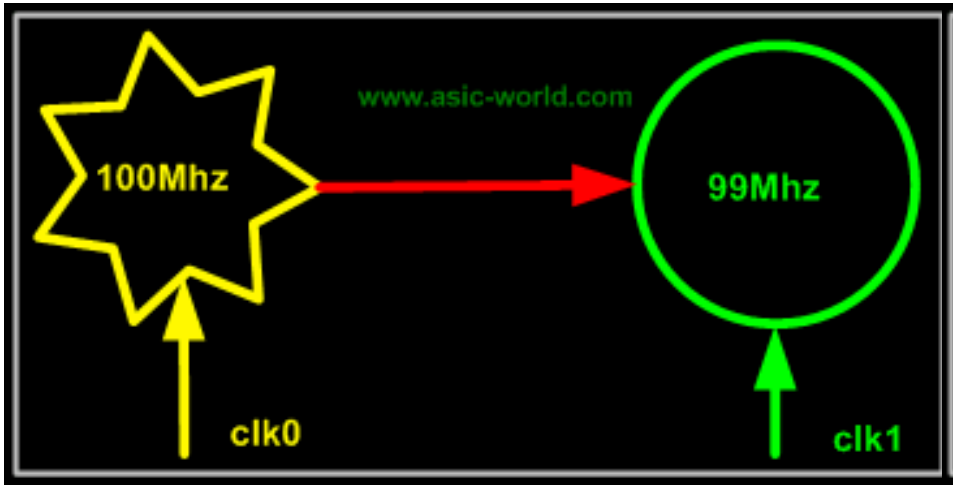


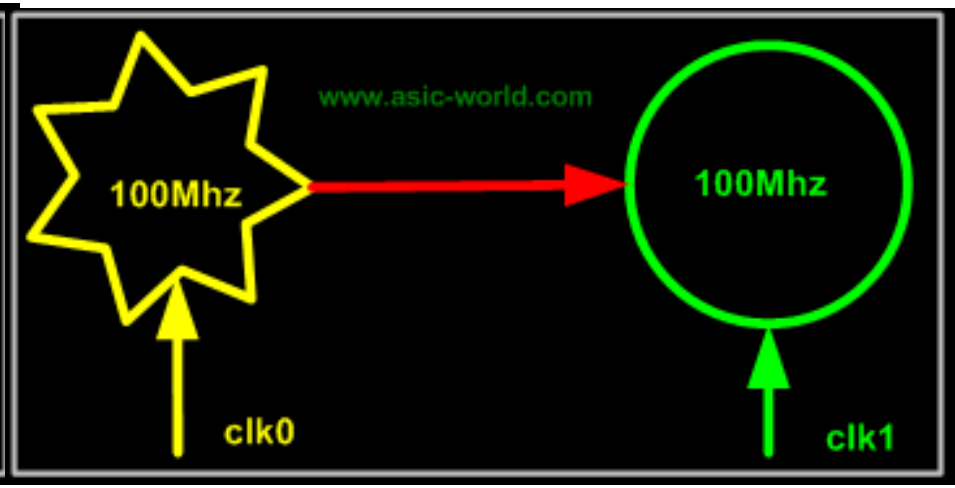


Interfacing Two Clock Domains

Introduction

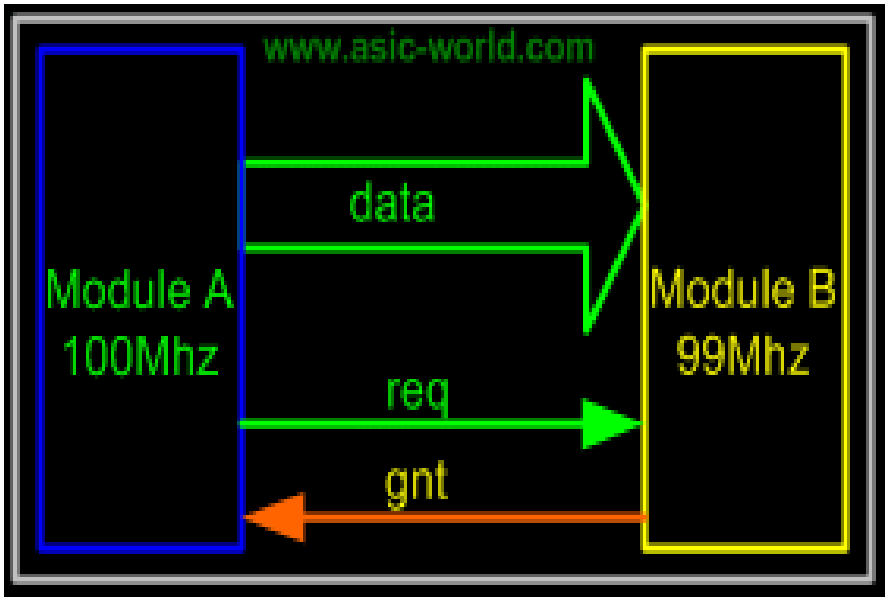
- Any two systems considered asynchronous to each other when working at
  - Different frequencies
  - Same frequency, but different phases





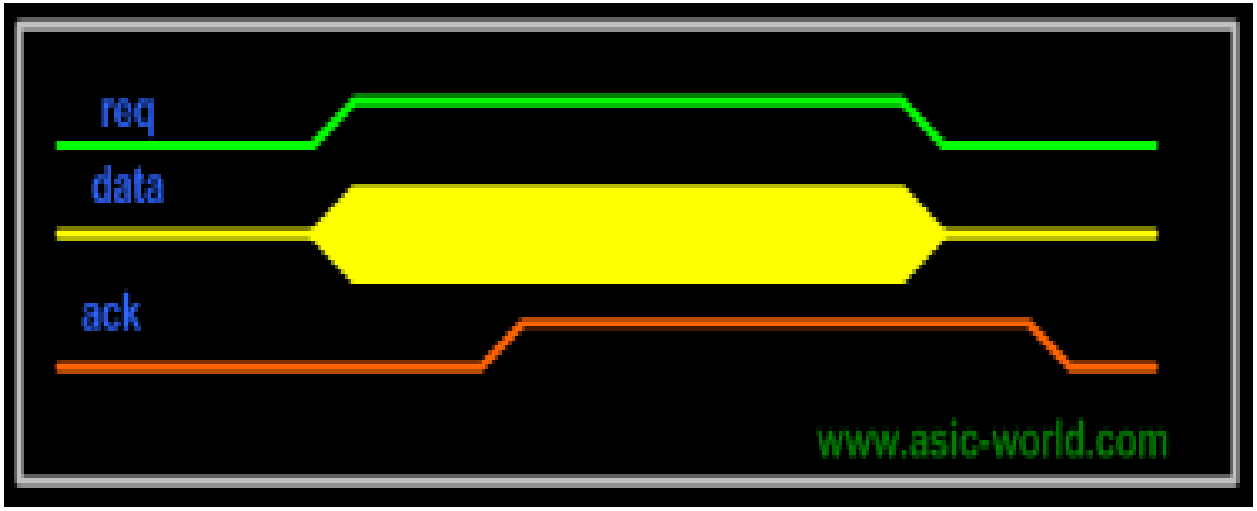
How to do data transfer?

- Handshake signaling
  - System A sends data to system B based on
    - the handshake signals
      - req
      - Ack
- Asynchronous FIFO
  - Two Clocks
  - Two data interfaces



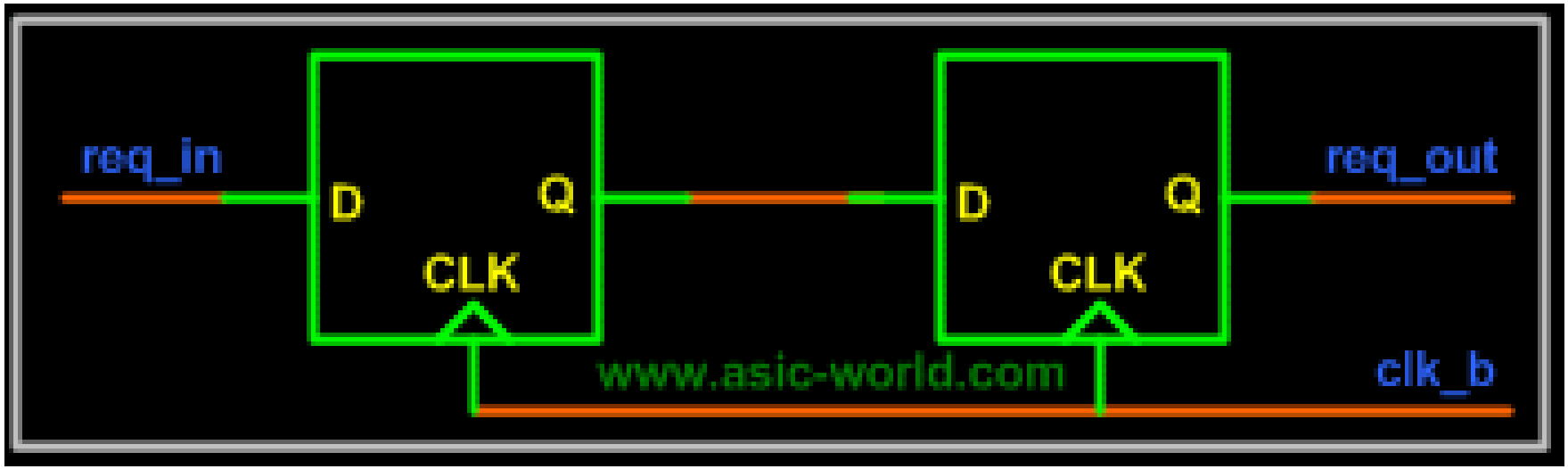
Handshake Signaling (1)

- Transmitter asserts req
  - asking the receiver to accept the data on the data bus
- Receiver asserts ack
  - asserting that it has accepted the data



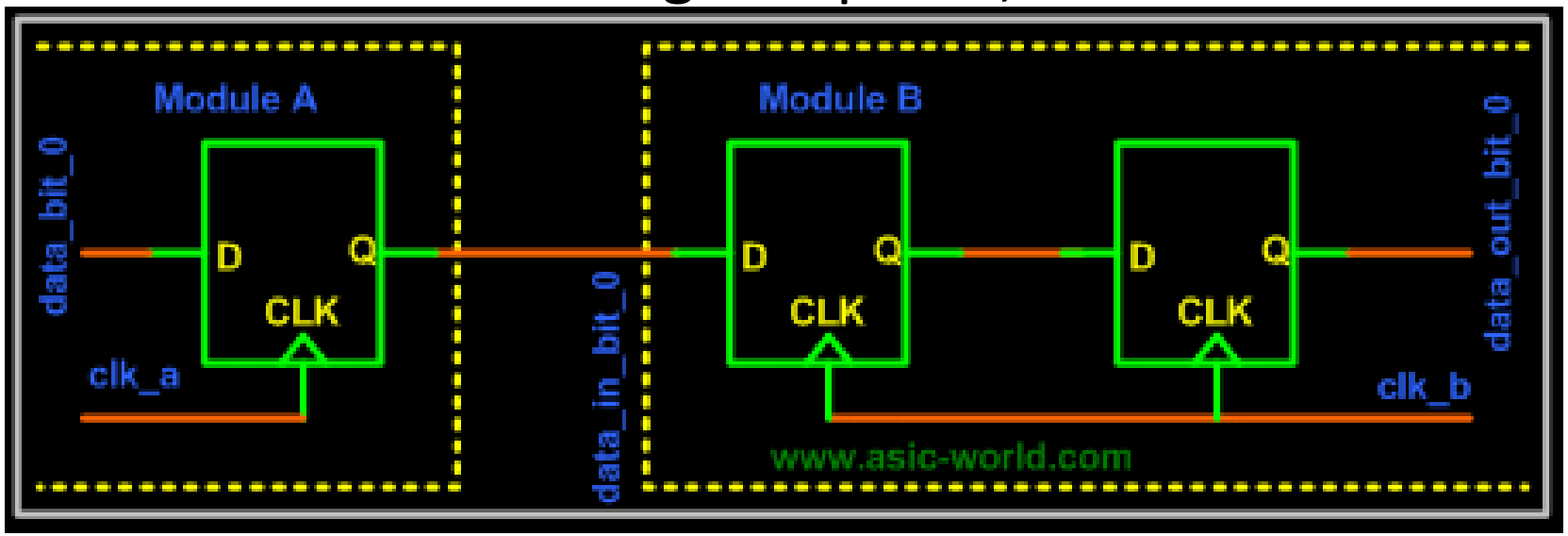
Handshake Signaling (2)

- Asynchronous input to the receiver
  - Setup/hold time violations possible
    - Double or triple stage synchronizers
      - Req
        - necessary



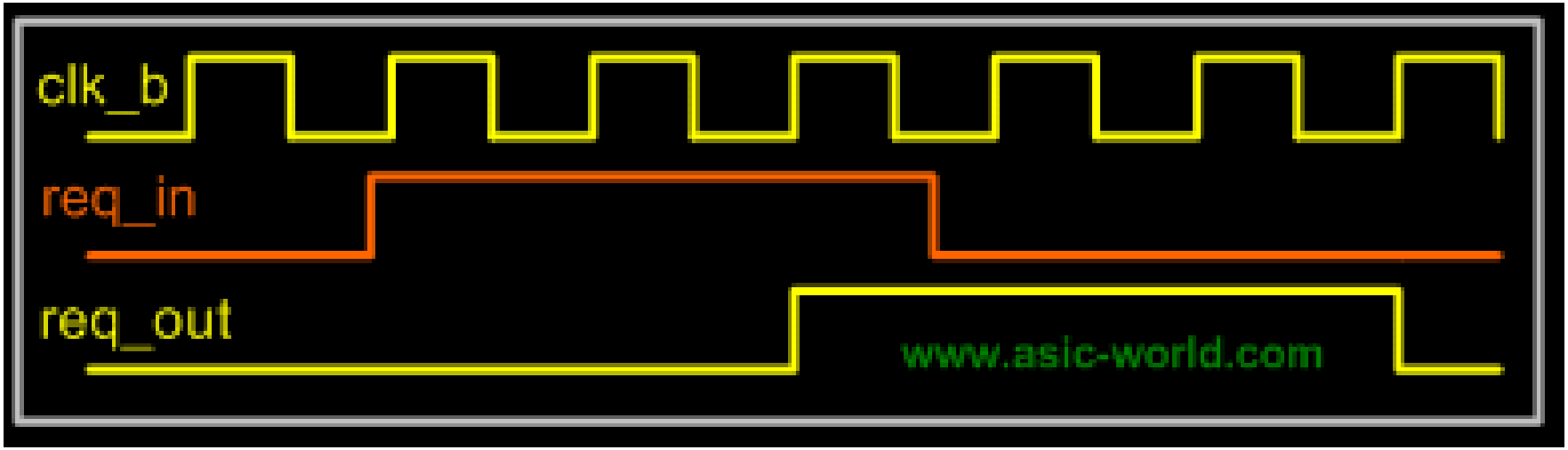
Handshake Signaling (3)

- Asynchronous input to the receiver
  - Setup/hold time violations possible
    - Double or triple stage synchronizers
      - Data
        - Lots of logic required, usu. not used



Handshake Signaling (4)

- A lot of clock cycles wasted handshaking
  - Use Asynchronous FIFO



  - High frequency input

Asynchronous FIFO (1)

- Two interfaces
  - One for writing the data into the FIFO
  - One for reading the data out
- Two clocks
  - One for writing
  - One for reading

Asynchronous FIFO (2)

- FIFO full
  - Used by system A
  - Driven by write clock (system A clock)
- FIFO empty
  - Used by system B
  - Driven by read clock (system B clock)

Asynchronous FIFO (3)

- Compared with handshaking
  - Higher performance
  - More resources