# EE141 – Discussion #2

Brian Zimmer

9/12/2014 and 9/15/2014

## Coding Guidelines

The following are helpful guidelines that ensure your simulation results will match what they synthesized hardware will do:

1. When modeling sequential logic, use nonblocking assignments.
2. When modeling latches, use nonblocking assignments.
3. When modeling combinational logic with an always block, use blocking assignments.
4. When modeling both sequential and combinational logic within the same always block, use nonblocking assignments.
5. Do not mix blocking and nonblocking assignments in the same always block.
6. Do not make assignments to the same variable from more than one always block.
7. Use $strobe to display values that have been assigned using nonblocking assignments.
8. Do not make assignments using #0 delays.

#1 thing we will be checking in your Verilog submissions!

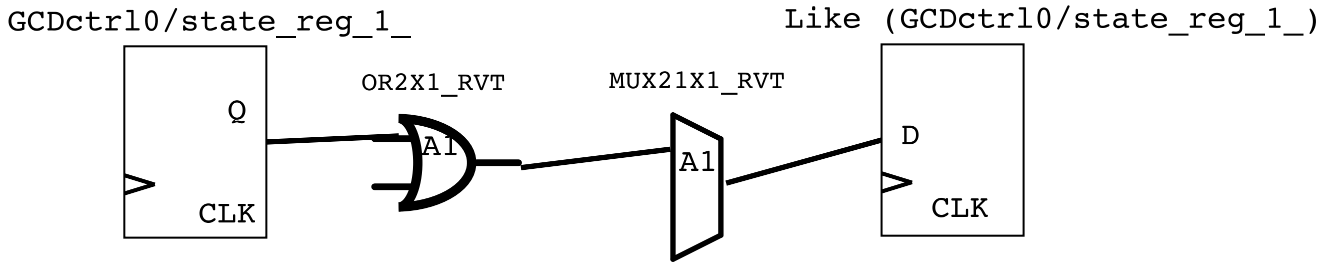
# Simulating Verilog…

```
# Simple simulation makefile
#
vcs_clock_period = 1.0

default : all

vsrcs = \
        decoder.v \
        decoder_tb.v \

VCS      = vcs -full64
VCS_OPTS = -notice -PP -line +lint=all,noVCDE +v2k -timescale=1ns/10ps -debug

vcs_sim = simv
$(vcs_sim) : Makefile $(vsrcs)
  $(VCS) $(VCS_OPTS) +incdir+./ -o $(vcs_sim) \
         +define+CLOCK_PERIOD=$(vcs_clock_period) \
         -sverilog $(vsrcs)

vpd = vcdplus.vpd
$(vpd): $(vcs_sim)
  ./simv +verbose=1
  date > timestamp

run: $(vpd)

.PHONY: run

all : $(vcs_sim)

#--------------------------------------------------------------------
# Clean up
#--------------------------------------------------------------------

junk += simv* csrc *.vpd *.key DVE* .vcs* timestamp

clean :
  rm -rf $(junk) *~ \#* *.log *.cmd *.daidir
```

**Make a directory,**
**Name this file ‘Makefile’,**
**Change vsrcs**
**Type ‘make run’**

## Timing

# Exercise: 28nm DFF

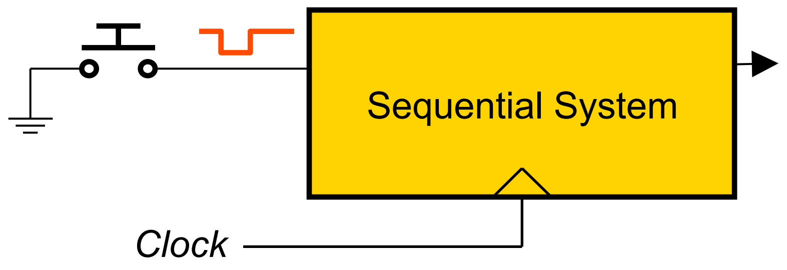


```
(CELL
  (CELLTYPE "DFFX1_RVT")
  (INSTANCE GCDctrl0/state_reg_1_)
  (DELAY
    (ABSOLUTE
    (IOPATH (posedge CLK) Q (0.094:0.095:0.095)
(0.101:0.102:0.102))
    (IOPATH (posedge CLK) QN (0.080:0.080:0.080)
(0.072:0.073:0.073))
    )
  )
  (TIMINGCHECK
    (WIDTH (posedge CLK) (0.043:0.043:0.043))
    (WIDTH (negedge CLK) (0.038:0.038:0.038))
    (SETUP (posedge D) (posedge CLK) (0.033:0.034:0.034))
    (SETUP (negedge D) (posedge CLK) (0.034:0.035:0.035))
    (HOLD (posedge D) (posedge CLK) (-0.017:-0.019:-0.019))
    (HOLD (negedge D) (posedge CLK) (-0.018:-0.019:-0.019))
  )
)


(INTERCONNECT GCDctrl0/state_reg_1_/QN GCDdpath0/U18/A1
(0.009:0.009:0.009))
(INTERCONNECT GCDctrl0/state_reg_1_/QN GCDdpath0/U18/A2
(0.018:0.018:0.018))
```

```
(CELL
  (CELLTYPE "OR2X1_RVT")
  (INSTANCE GCDdpath0/U18)
  (DELAY
    (ABSOLUTE
    (IOPATH A1 Y (0.037:0.038:0.038)
(0.038:0.039:0.039))
    (IOPATH A2 Y (0.034:0.034:0.034)
(0.032:0.033:0.033))
    )
  )
)

(CELL
  (CELLTYPE "MUX21X1_RVT")
  (INSTANCE GCDdpath0/U25)
  (DELAY
    (ABSOLUTE
    (IOPATH A1 Y (0.047:0.048:0.048)
(0.046:0.046:0.046))
    (IOPATH A2 Y (0.056:0.057:0.057)
(0.057:0.058:0.058))
    (IOPATH (posedge S0) Y (0.053:0.053:0.053)
(0.061:0.061:0.061))
    (IOPATH (negedge S0) Y (0.053:0.053:0.053)
(0.061:0.062:0.062))
    )
  )
)
```

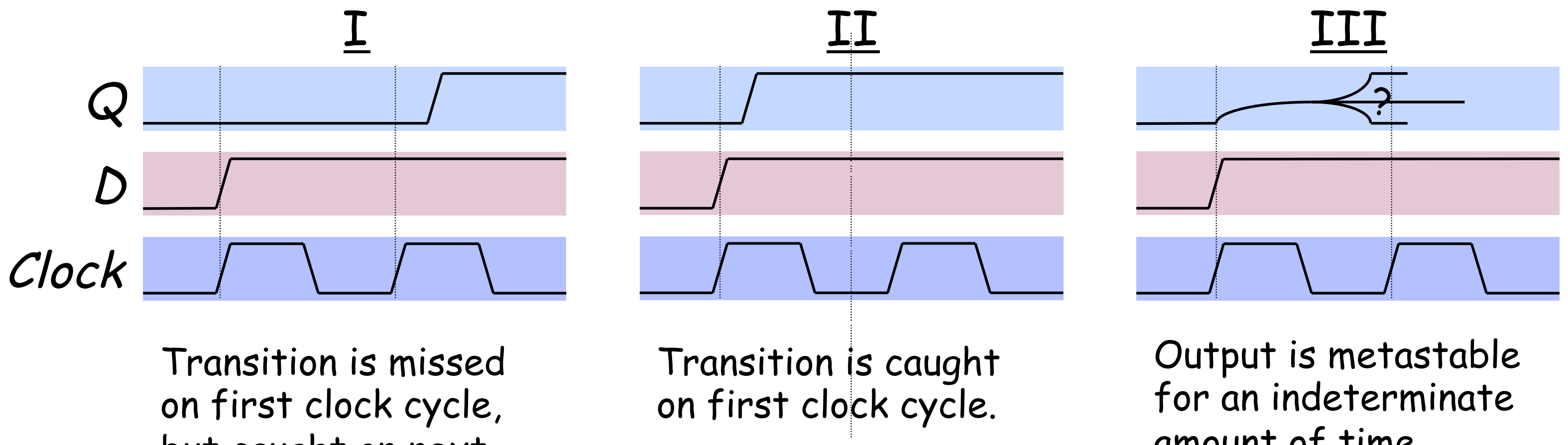
# Asynchronous Inputs in Sequential Systems

What about external signals?



*Can't guarantee setup and hold times will be met!*

When an asynchronous signal causes a setup/hold violation...

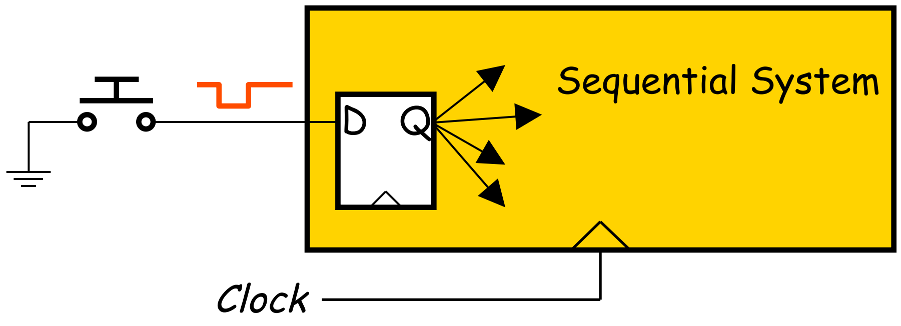


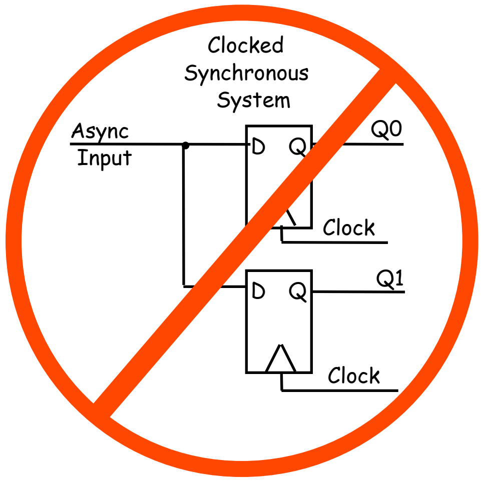
| I | II | III |
|---|---|---|
| Transition is missed on first clock cycle, but caught on next clock cycle. | Transition is caught on first clock cycle. | Output is metastable for an indeterminate amount of time. |

# Asynchronous Inputs in Sequential Systems

All of them can be, if more than one happens simultaneously within the same circuit.

*Guideline: ensure that external signals directly feed exactly one flip-flop*

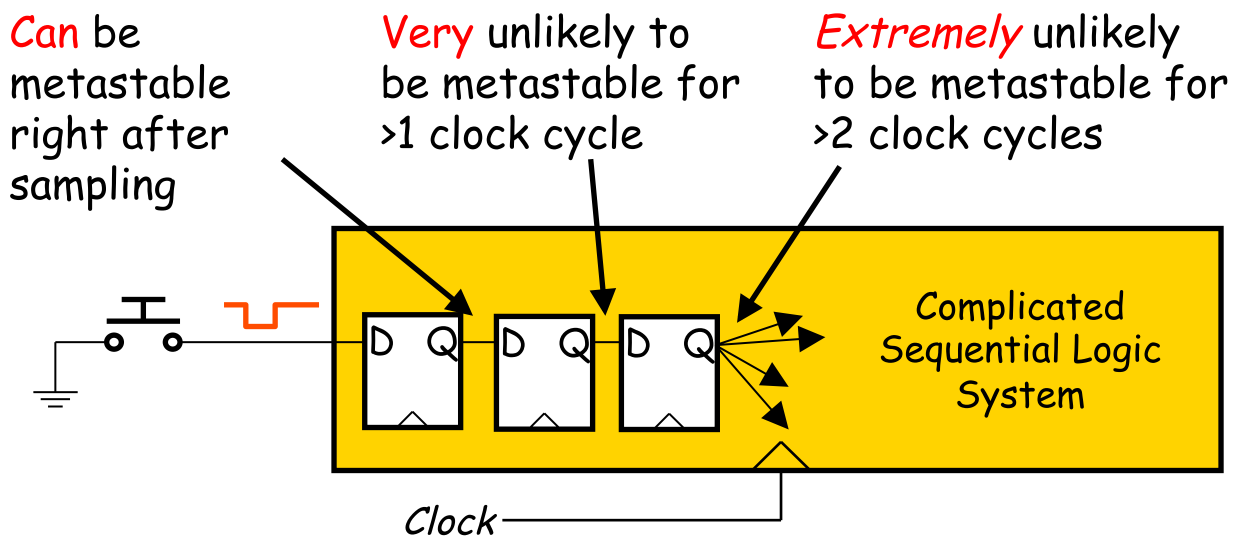




This prevents the possibility of I and II occurring in different places in the circuit, but what about metastability?

## Handling Metastability

- Preventing metastability turns out to be an impossible problem
- High gain of digital devices makes it likely that metastable conditions will resolve themselves quickly
- Solution to metastability: allow time for signals to stabilize

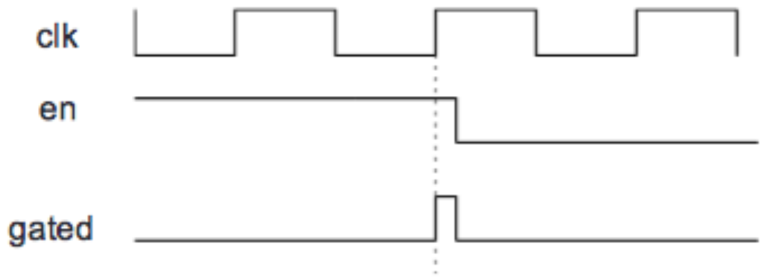

Can be metastable right after sampling
Very unlikely to be metastable for >1 clock cycle
Extremely unlikely to be metastable for >2 clock cycles
D Q
D Q
D Q
Complicated Sequential Logic System
Clock

*How many registers are necessary?*

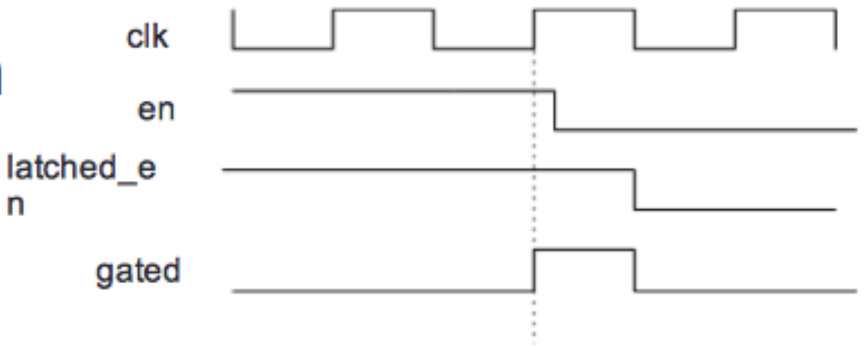
- ❑ Depends on many design parameters (clock speed, device speeds, …)
- ❑ In the labs, a pair of synchronization registers is sufficient

# Clock Gating Specifics

- Bad: Without latch
  - gated = clk & en
  - Is the pulse long enough?
  - en based on that cycle's extension bits

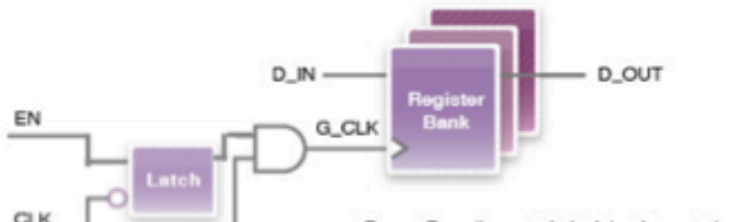


- Good: With active low latch
  - Latch will hold when clock is high, and pass when clock is low -> get full clock pulse



- In verilog: Use DC clock gating

```
always @(posedge clk)
    if(en)
        q <= d;
```



Power Compiler gated-clock implementation