

L10.1 Quiz 1 Wed. February 27, 2002

Friday, March 1, 2002
 1:00 – 2:00 PM, Room 50–340 – Walker
 1 handwritten ‘crib’ sheet allowed (both sides).

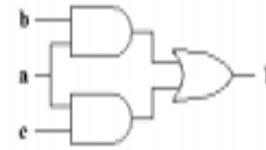
Venue
 Problem Sets 1–3, Lab 1, Lectures 1–7
 Understand VHDL entities, simple architectures.
 Understand concurrent statements:
 assignment, instantiation, when–else, with–select–when, and process.
 Understand sequential statements (only within a process):
 assignment, if–then–elsif–else, and case–when
 You are not responsible for generating (or checking) VHDL syntax.

General Topics
 Boolean Algebra and Elementary Logic Expressions
 Combinational Logic
 Latches and Edge Triggered Flip–flops
 PALs (PLDs)
 FSM
 Synchronization

L10.2 – L 2.10 Circling Groups Fri. February 8, 2002

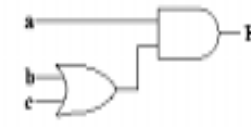


$F = a * b + a * c$



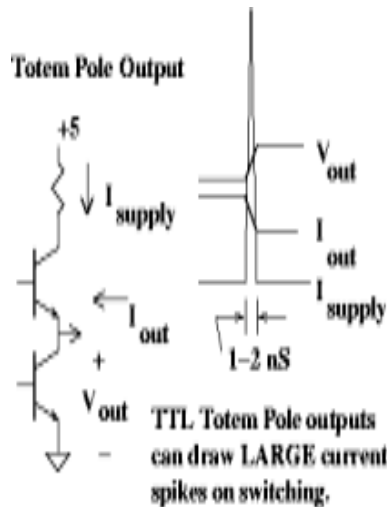
MSP: OR of ANDs
 Circle 1s

$F = a * (b + c)$



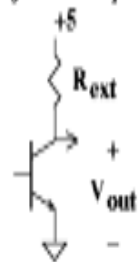
MPS: AND of ORs
 Circle 0s

L10.3 – L 2.19 Totem Pole Output Fri. February 8, 2002

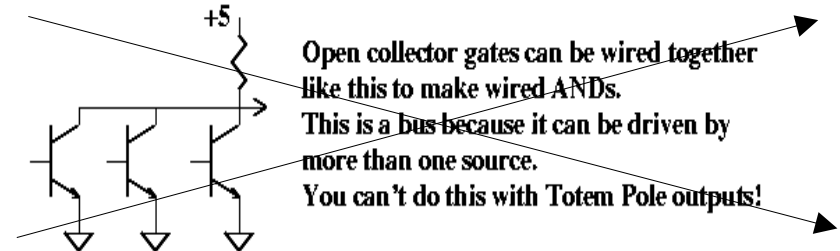


TTL Totem Pole outputs can draw LARGE current spikes on switching.

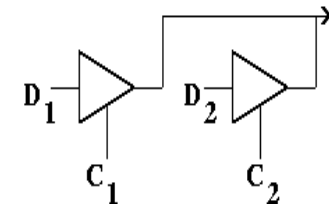
Some outputs are open collector: need a pull-up resistor. Speed is affected by R_{ext} and by external and junction capacitance.



L10.4 – L 2.20 Busses Fri. February 8, 2002



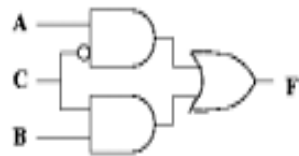
By controlling the gates on both transistors of a Totem Pole to be open, a high impedance is created (this is a tri–state). Control inputs C_1 and C_2 are output enables.



L10.5 – L 2.21 Hazards Fri. February 8, 2002

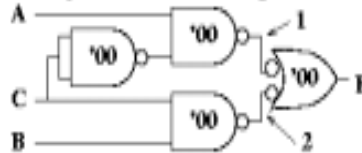
Static Hazards: Consider this function:

$$F = A * \bar{C} + B * C$$

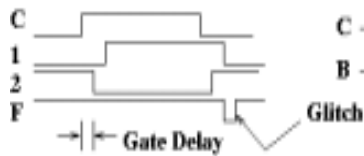


	AB			
C	00	01	11	10
0	0	0	1	1
1	0	1	1	0

Implemented with MSI gates:

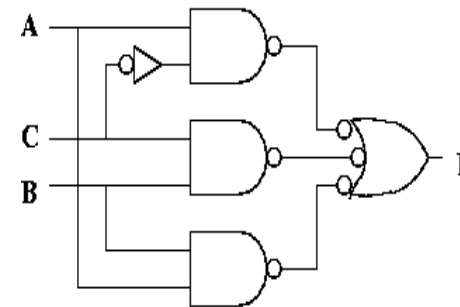


A = B = 1



L10.6 – L 2.22 Fixing Hazards Fri. February 8, 2002

The glitch is the result of timing differences in parallel data paths. It is associated with the function jumping between groupings or product terms on the K-map. To fix it, cover it up with another grouping or product term!

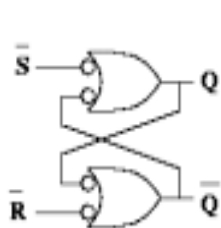


	AB			
C	00	01	11	10
0	0	0	1	1
1	0	1	1	0

$$F = A * \bar{C} + B * C + A * B$$

L10.7 – L4.2 Latch Wed. February 13, 2002

S-R Latch (74LS279)



\bar{S}	\bar{R}	Q	\bar{Q}
0	0	1	1
0	1	1	0
1	0	0	1
1	1	1	0
		0	1

} Both work!
 (Holds state)

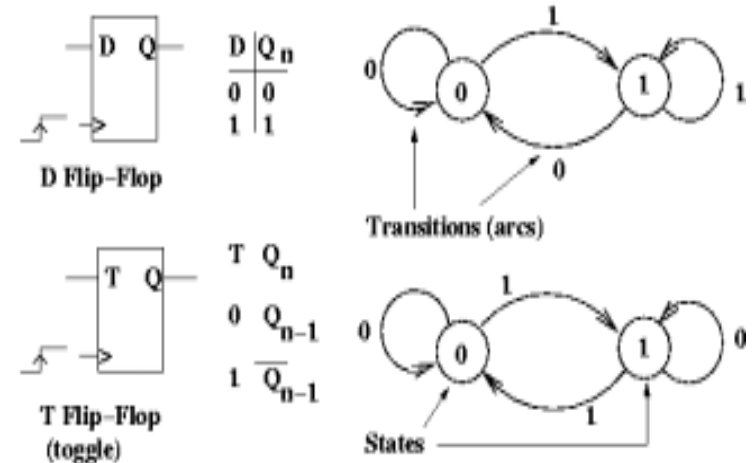
Question: What happens if \bar{S} and \bar{R} go 0 to 1 at the same time?

You can build a latch from NAND gates, but there is a packaged MSI version.

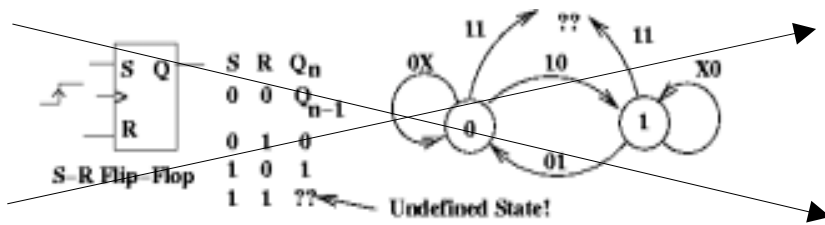
Question: What happens if you use NOR gates?

L10.8 – L4.7 D and T Flip-Flops Wed. February 13, 2002

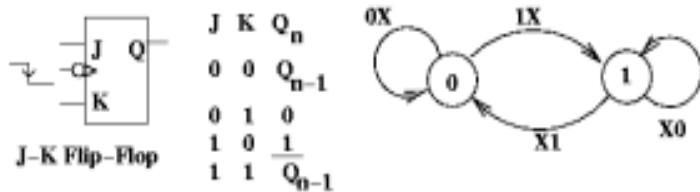
Flip-flops are simple finite state machines which have two-states.



L10.9 – L4.8 SR and JK Flip-Flops Wed. February 13, 2002



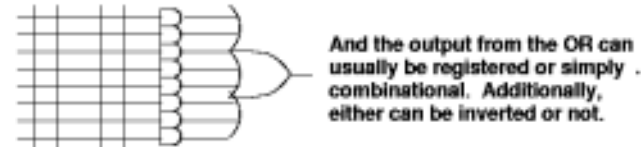
The SR flip-flop is an edge triggered version of the SR latch. It has an undefined state problem which is resolved by the JK flip-flop.



Note that this version of the JK flip-flop has a negative edge triggered clock as indicated by the bubble.

10.10 – L4.10 OR of ANDs Wed. February 13, 2002

The product terms produced by the ANDs are combined in large ORs.

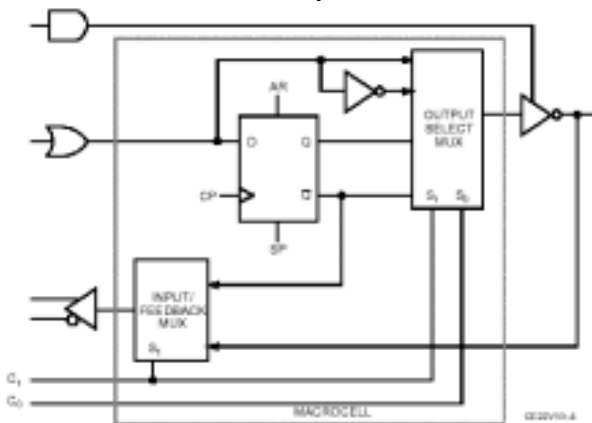


Pin	20V8(1)		20V8(2)		22V10	
	Use	Terms	Use	Terms	Use	Terms
23	I		I		IO	8
22	O	7	IO	8	IO	10
21	IO	7	IO	8	IO	12
20	IO	7	IO	8	IO	14
19	IO	7	IO	8	IO	16
18	IO	7	IO	8	IO	16
17	IO	7	IO	8	IO	14
16	IO	7	IO	8	IO	12
15	O	7	IO	8	IO	10
14	I		I		IO	8
13	I		IOE		I	

20V8 (1): All outputs are combinational (none registered).
 (2): Some outputs registered: 8 product terms for registered outputs only.

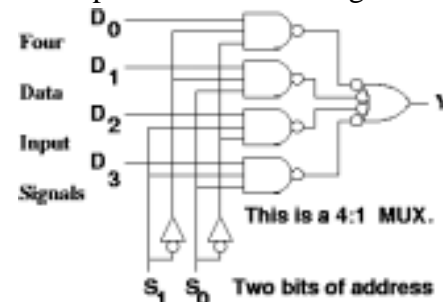
L10.11 – L4.14 22V10 Macrocell Wed. February 13, 2002

Flexible 'Macrocells'
 10 Macrocells, varying number of product terms
 The clock is still from pin 1.



L10.12 – L5.4 MUX (and its inverse) Fri. February 15, 2002

A multiplexer selects one signal according to an address.

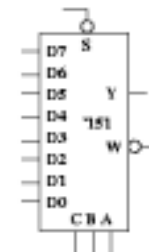


This is a 4:1 MUX.

Two bits of address



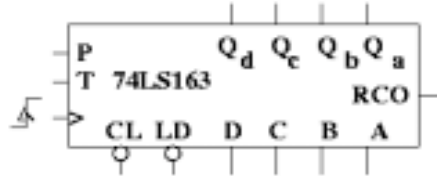
74LS151 is a 8:1 MUX.
 Strobe is active low.
 $Y = S \cdot D(CBA)$
 $W = \bar{Y}$



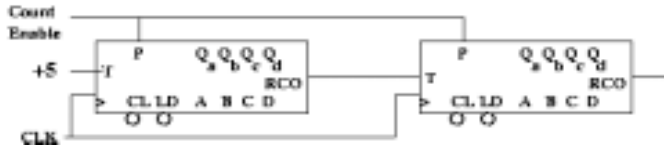
L10.13 – L5.10 Cascading Fri. February 15, 2002

The 163 will count ONLY if the P and T inputs are high.
 Note that the RCO is the AND of all four bits and T.

74LS163 has clear and load functions too.
 CL and LD are synchronous.
 if CL then $Q := 0$
 else if LD then $Q := I$
 else if $P * T = 1$ then $Q := Q + 1$
 else $Q := Q$

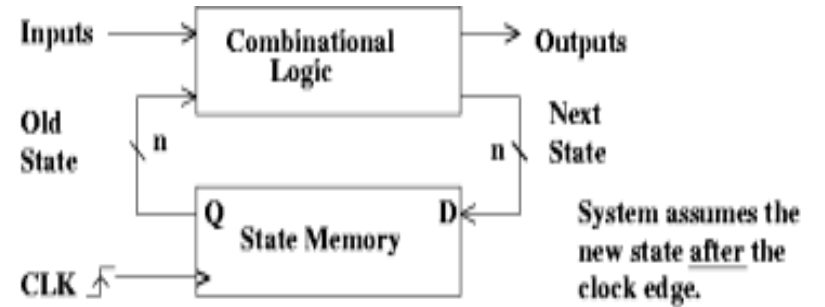


To cascade synchronous counters (to count more bits):
 P is "count Enable".
 RCO and T are daisy chained.



L10.14– L5.12 Finite State Machines Fri. February 15, 2002

Finite state machines are clocked sequential systems.

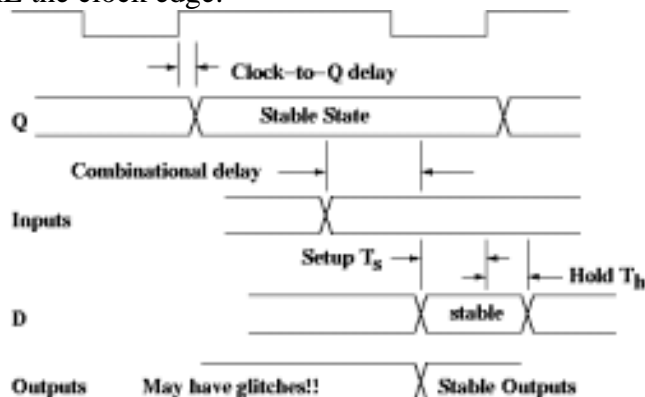


We have already seen simple FSMs in flip-flops and counters.
 But, you can do much more complex things with them.
 After a clock edge, the FSM assumes a state that depends on the state it WAS in and the inputs just before the clock edge.

L10.15 – L5.13 Timing of an FSM Fri. February 15, 2002

Clock Speed is limited by the flip-flop delay, combinational delay, and setup time.

The new state is determined by the inputs and the old state just BEFORE the clock edge.



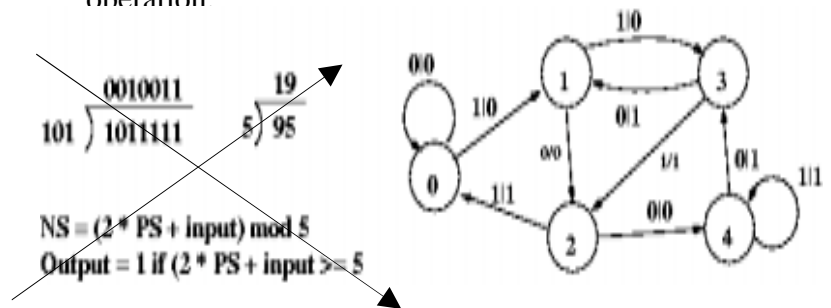
L10.16 – L5.19 State Diagram Meaning Fri. February 15, 2002

This FSM has a single input which represents a number.

The LSB comes first, another with each clock pulse.

The output, also bit serial, is similar.

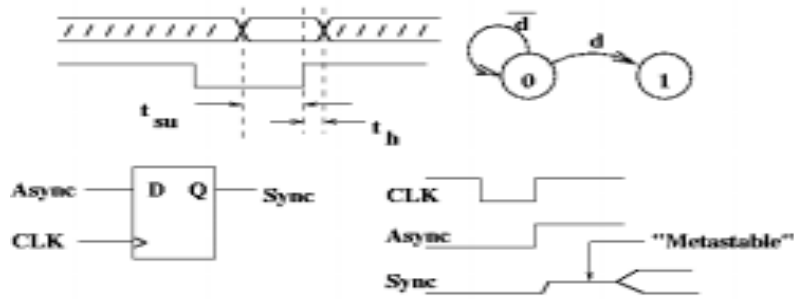
The state of the FSM is the remainder of the division operation



L10.17 – L7.13 Important Design Rule Wed. February 20, 2002

DESIGN RULE:

1. Synchronize ALL external signals.
2. Any asynchronous input must affect ONLY ONE flip-flop.



Any combinational logic with "Sync" as an input will be "glitchy" until after the metastable state has expired. In particular, do NOT use "Sync" as a CLK input.

L10.18 Avoiding Glitches – Register Wed. February 27, 2002

Creating glitch free outputs:

Register the output.

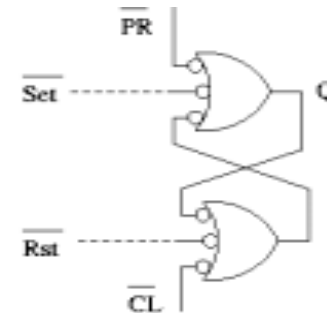
If a flip-flop does NOT change state upon occurrence of a clock pulse then it has no glitches, i.e.,

- 0 to 0 is guaranteed not to glitch to 1 in between and
- 1 to 1 is guaranteed not to glitch to 0 in between.

We assume, of course, that the setup and hold times are honored.

This is the output latch of an edge triggered flip-flop.

The set or reset pulses (negative true) are full pulses and only one occurs if the setup and hold times are adhered to.

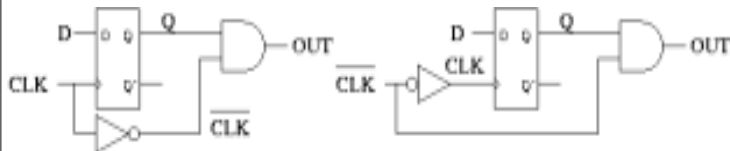


L10.19 Avoiding Glitches – Gating Wed. February 27, 2002

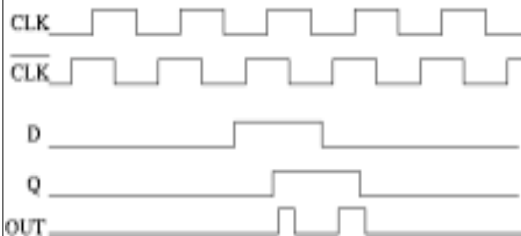
Gate the output with the clock (carefully).

This is another way of saying "Don't look at a combinational output until the output has settled down into its final state".

Make sure that the gated pulse does NOT have a glitch.



Suppose the inverter delay is large compared to the CLK to Q delay. Then there can be a glitch on the circuit to the left but not on the circuit to the right.



Remember –
 Do NOT use glitchy signals for CLK, PR, CLR, S, or R. Clock data into a register AFTER signals are stable.

L10.20 VHDL Identifiers Wed. February 27, 2002

Case Insensitive (but best not to rely on this)

First character must be a letter.

Letters, Digits, and Underscores (only)

Two underscores in succession are not allowed.

The last character cannot be an underscore.

Using reserved words is NOT allowed.

Reserved words are AQUA in later versions of emacs.

Using reserved words usually provokes an understandable error comment.

Legal Examples

CLK, Three_StateEnable, h23, Reg_12

Illegal Examples

_clk, 3_State_Enable, large#num, clk_, Three__State, register, begin

L10.21 VHDL Reserved Words Wed. February 27, 2002

Some are

abs access after
array disconnect file
guarded impure postponed
rem unaffected wait

In other words, there are too many to remember!

This is another good reason for "incremental" compilation.

Start with something that compiles and add code a block at a time!

To see what is a reserved word, notice the color in your editor.

L10.22 Don't Care, Ampersand Wed. February 27, 2002

Don't cares are represented by a – (hyphen).

'–' single quotes for a character

"——" double quotes for a string (vector)

(others => '–') for something independent of length

& (ampersand) to concatenate strings or signals

"01" & "111" is the same as "01111"

'0' & "1111" is the same as "01111"

Remember, VHDL is strongly typed.

a + b is valid only if a and b are of the same length.

The result is of the SAME length as a (or b).

If you want it to be one bit longer, then use

c <= ('0' & a) + ('0' & b)

-- Of course, c must be defined to be

-- one bit longer than a.

L10.23 Predefined Attributes Wed. February 27, 2002

s'event is read as "s tick event" where s is a signal name.

rising_edge(event) is the same as (s'event and event = '1')

as synthesis only worries about 1 and 0.

Transactions occur when a signal is evaluated, whether or not the signal value changes.

Time, if not specified, is in deltas.

s'event – true only if an event occurred in the
current delta cycle

s'active – true only if a transaction occurred on s

s'last_event – time elapsed since the last event on s

s'last_value – value of s before the previous event on s

s'last_active – time elapsed since the previous transaction on s

L10.24 Predefined Attributes, cont'd Wed. February 27, 2002

Signal attributes that create a signal

Mainly used for simulation

s'delayed[(time)] – creates a signal the same as s but delayed by the specified time

s'stable[(time)] – a Boolean signal that is true if s had no events for the specified time

s'quiet[(time)] – a Boolean signal that is true if s had no transactions for the specified time

s'transaction – a signal of type bit that changes for every transaction on s

L10.25 Array Attributes Wed. February 27, 2002

Signal s : std_logic_vector(7 downto 3)

s'left = 7 s'high = 7
s'right = 3 s'low = 3
s'length = 5

type rom is array (0 to 6, 3 down to 0) of std_logic;
signal r : rom;

r'left(1) = 0 r'high(1) = 6
r'left(2) = 3 r'high(2) = 3
r'right(1) = 6 r'low(1) = 0
r'right(2) = 0 r'low(2) = 0
r'length(1) = 7
r'length(2) = 4

L10.26 User Defined Attributed Wed. February 27, 2002

Type state_type is (idle, state1,state2);
attribute state_encoding of state_type: is sequential;
-- or one_hot, zero_hot, gray
attribute enum_encoding of state_type: is "11 01 00";
-- or whatever assignment you want to make
-- within an entity
attribute pin_numbers of counter:Entity is
"clk:13 reset:2 &
" count(3):3;
-- Note the space before count(3) above
-- within an entity
attribute pin_avoid of mydesign: entity is "21 24 26";
-- the following are not recommended
attribute lab_force of mysig: signal is a1;
attribute node_num of buried: signal is 202;
attribute low_power of mydesign: entity is "b g e";
attribute slew_rate of count(3): signal is slow; -- or fast

L10.27 These are Sometimes Useful Wed. February 27, 2002

Sum splitting occurs because of too many product terms.

Balanced (default) has better timing but uses more macrocells.
Cascaded uses fewer macrocells and is slower.

attribute sum_split of mysig: signal is cascaded;

The synthesis_off attribute is used to make the signal a factoring point.

Making a signal a factoring point can result in a reduction of product terms for a subsequent signal.

Registered equations are natural factoring points so only use synthesis_off on combinational signals.

attribute synthesis_off of sel: signal is true;