## University of California at Berkeley College of Engineering Department of Electrical Engineering and Computer Sciences

EECS150 Spring 2000 J. Wawrzynek E. Caspi

Homework #5 – Solution

First, some notes about transistors.

In our simplified transistor model for digital applications, a transistor acts as a switch controlled by the gate voltage. When on, a transistor provides a resistive path between source and drain. When off, the source to drain path has high impedance ("high Z") and is effectively an open circuit. N-type transistors (nfets) can reliably transmit a 0 signal, whereas P-type transistors (pfets) can reliably transmit a 1 signal. To transmit either logic level, we need an nfet-pfet pair in parallel, commonly called a *transmission gate*. Because MOSFETs are naturally inverting, CMOS design lends itself to making *inverting logic*, where basic gates and structures compute an inverted output (*e.g.* inverter, NAND, NOR).



Static CMOS design requires that we use an nfet network connected to GND to generate the 0-valued output of a function and a complementary pfet network connected to  $V_{DD}$  to generate the 1-valued output. Exactly one of the two networks is on at a time. The CMOS inverter is the simplest static CMOS gate: 1 nfet, 1 pfet. A NAND structure requires a series nfet network and a parallel pfet network. A NOR structure requires a parallel nfet network and a series pfet network. CMOS structures are simple to design, robust to noise, and do not consume power except when switching. One problem with these structures is that if the input floats to a value between 0 and 1 (e.g.  $V_{DD}$ /2), then the nfet and pfet networks may turn on simultaneously, causing excessive current draw from  $V_{DD}$  to GND and possibly damaging the circuit.

Design using transmission gates is a bit different. Because transmission gates pass 0 and 1 equally well, their source/drain need not be connected directly to  $V_{DD}$  or GND. Instead, they are typically used to selectively pass an input signal (akin to classical switch-based logic). Transmission gates are often used in multiplexor-like structures where multiple transmission gates share a common output but only one is allowed to write at a time.

 The (a) 3-input NAND gate and (b) 2-input OR gate are static CMOS structures. For (b), we use a NOR followed by an inverter. The (c) multiplexor uses a pair of transmission gates with common output.



(a) 3-input NAND

(b) 2-input OR

(c) 2-to-1 multiplexor

2. This circuit implements a 2-input NOR.



3. The speed of transition of an RC circuit depends on its *RC time constant*:  $\mathbf{t} = RC$ . Specifically,  $\mathbf{t}$  is the time required for the output to swing through  $(1-1/e) \approx 62\%$  of its transition. We do not actually need to calculate anything here. The important thing to realize is that larger *R* and *C* make for a slower transition. By comparing transition times we can determine relative *R* values for two incarnations of the given circuit with identical *C*. Similarly, we can determine relative *C* values for two incarnations of the given circuit with identical *R*. Comparing the bottom timing graphs, both of which use C1, we can conclude that  $\underline{R1} < \underline{R2}$  (the diagram with R1 switches faster). Comparing the right timing graphs, both of which use R1, we can conclude that  $\underline{C1} < \underline{C2}$  (the diagram with C1 switches faster).



4. We assume that the latch is *transparent* (passes input to output) when the clock is high and *holding* (maintains output) when the clock is low. In the graph below, the gray regions indicate low clock, when the latch is holding, and Q does not change. Outside those regions, the latch is transparent, and Q follows D with a small delay.



5. The *critical path* of a circuit is the path from input to output with greatest delay. In the given circuit, there are several paths passing through 3 NAND gates. The critical delay, however, is actually 4 NAND gates due to the *fanout* at y (see diagram below). In CMOS circuits, the delay of a gate really boils down to how fast the gate can switch its output level. Switching requires charging (or discharging) the load capacitance, which arises primarily from the wires and other gates attached to the output. Outputting to *n* gates increases the capacitance, and hence the switching

delay, by a factor of n. In the given circuit, the NAND gate with inputs c, d fans out to 2 other gates, incurring double delay. Hence the critical path delay is 4 NAND gates, not 3. The critical path is shown below. It is equivalent from c to x as from d to x.



6. The block-level diagram of a 2-LUT is shown below. Recall that a 2-LUT is a programmable circuit that can implement any function of 2 inputs by storing the function's values in a lookup table. We need 4 flip-flops to store the 2<sup>2</sup>=4 values of the function, plus a 4-to-1 multiplexor to choose which value to output. Each flip-flop can be built from a pair of level-sensitive latches with opposite clocks. This is called a *master- slave* flip-flop. The 4-to-1 multiplexor can be decomposed into a tree of 2-to-1 multiplexors, where each 2-to-1 is built from a pair of tri-state buffers. The block and transistor diagrams for all these basic components are shown below. The complete circuit follows.



2-LUT block diagram



Positive edge triggered flip-flop



4-to-1 multiplexor decomposition into 2-to-1 muxes



2-to-1 mux using tri-states



Tri-state buffer



Level-sensitive D Latch (transparent on clk high, holding on clk low)

