High Speed CMOS VLSI Design

Lecture 7: Dynamic Circuits

1.0 Introduction to Dynamic Gates

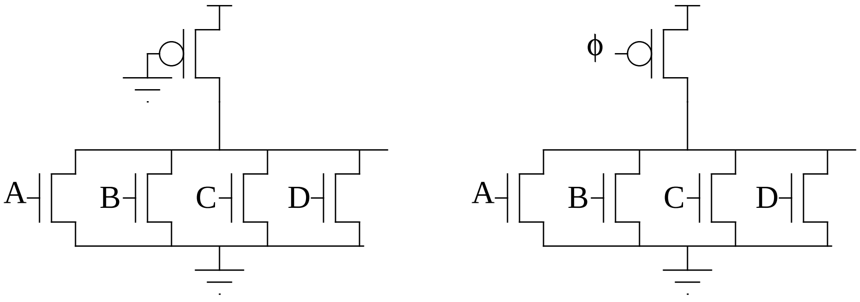
Static logic is great for its robustness. However, it is too slow to meet requirements for high speed processor critical paths. Thus, designers are increasingly turning to dynamic logic in hopes of achieving a 1.5 - 2x speedup over static gates.

The main drawback of static gates is the fact that inputs must drive both NMOS and PMOS transistors. Only one of the two transistors is ever on, meaning that the input capacitance of the other transistor loads the critical path without increasing the current drive of the gate. Moreover, the PMOS transistors must be large and thus add much capacitance. If we could construct gates which only have NMOS transistors in the critical path, circuits would run much faster.

Pseudo-NMOS logic achieves this goal by replacing the PMOS stack with a single grounded PMOS transistor serving as a resistive pullup. Thus, the NMOS pulldowns can be very fast. Unfortunately, the PMOS transistor fights against the NMOS during a falling transition, slowing the fall time. Also, it must be weaker than the NMOS, so the rise time is not very good. Finally, when the output is low, there is a path from VDD to GND wasting power.

An alternative scheme is to connect the PMOS transistor to a clock, instead of ground. This topology is called a dynamic gate. Figure 1 shows the similarity between dynamic logic and pseudo-NMOS gates.

FIGURE 1. Pseudo-NMOS and Dynamic gates



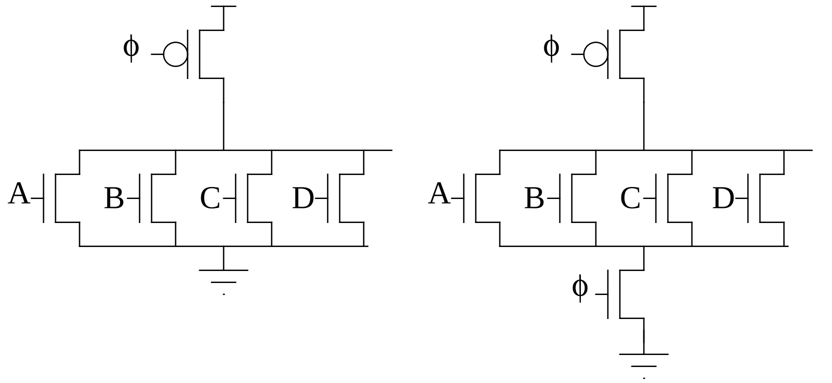
Pseudo-NMOS NOR4

Dynamic NOR4

Dynamic gates operate in two phases: precharge and evaluation. During the precharge phase, the clock is low, turning on the PMOS device and pulling the output high. During evaluation, the clock is high, turning off the PMOS device. The output may “evaluate” low through the NMOS transistor stack.

The **PRECHARGE** rule of dynamic gates states that *there should be no active path to ground during precharge*, so that the PMOS transistor can fully precharge the output high without contention with the NMOS pulldowns. Sometimes this can be achieved by guaranteeing that some inputs are low. For example, with a NOR4 gate, all four inputs must be low. For a NAND4 gate, only one of the series pulldown transistors must be low. It is not always possible to guarantee this condition, so often an extra clocked evaluation transistor is placed at the bottom of the pulldown stack, as shown in Figure 2.

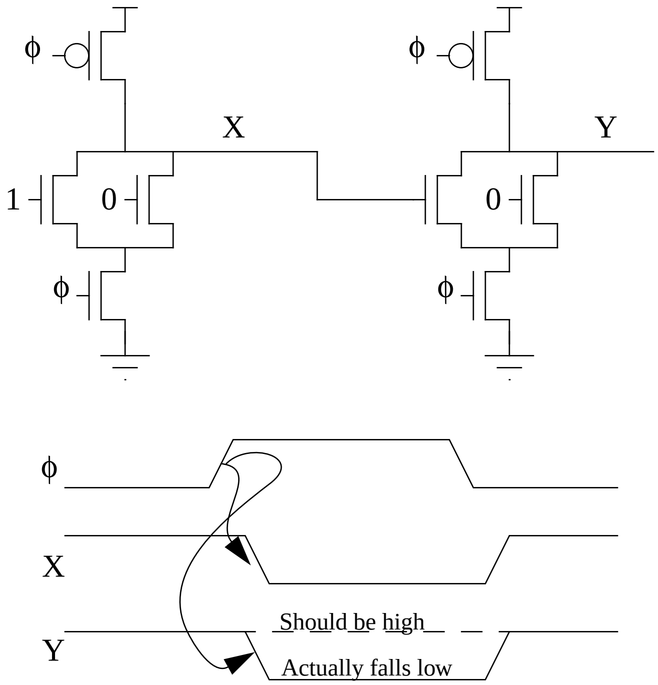
FIGURE 2. Dynamic gates with and without clocked evaluation transistors



No Clocked Evaluation Transistor   Clocked Evaluation Transistor

Another limitation of dynamic gates arises when one dynamic gate directly drives the next, as shown in Figure 3. When ϕ is low, both gates precharge high. When ϕ rises, both gates begin evaluating. Since the first gate has a high input, its output X falls low. The second NOR gate should therefore produce a high output. Unfortunately, the second NOR gate initially received the high value at X and evaluates low. By the time X falls, Y may have become corrupted. Since the precharge transistors are off, Y has no way of recovering to the correct high value during the evaluation phase. The circuit produces an incorrect result.

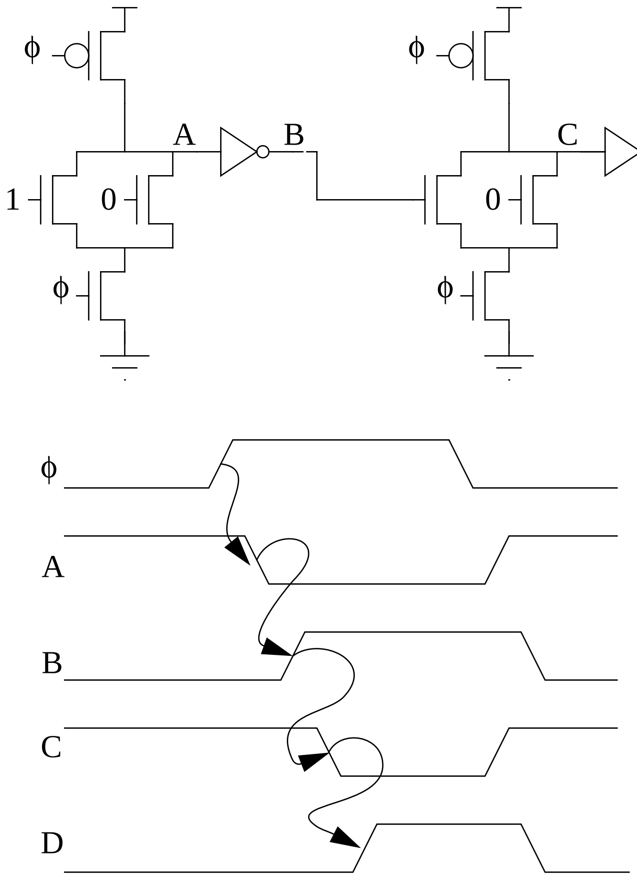
FIGURE 3. Incorrect operation of cascaded dynamic gates



The problem arose because the second dynamic gate had an input switch from high to low while the gate was evaluating. The output had pulled low while the input was high, then cannot recover to a correct high value. To avoid this problem, dynamic gates must obey the **MONOTONICITY** rule: *all inputs to dynamic gates should make only low to high transitions while the gates are evaluating*. Inputs must be “monotonically rising,” meaning they can stay low, stay high, or may rise, but may not fall.

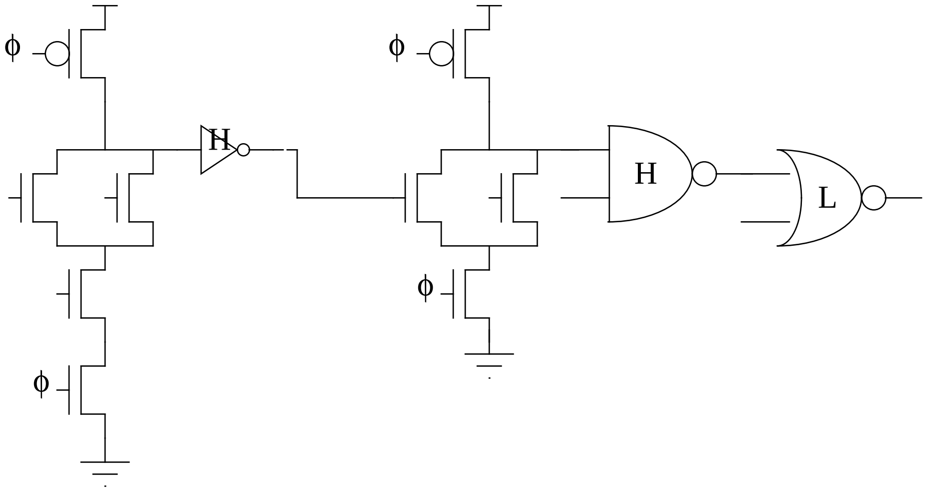
An easy way to achieve this condition is to insert an inverting static gate between each dynamic gate, as shown in Figure 4. Now the gate operates correctly. The dynamic/static gate pair is called a domino gate. All domino gates in a cycle are precharged simultaneously, like dominos being set up, then one may trigger the next which triggers the next, like a chain of dominos falling.

FIGURE 4. Correct operation with domino gates



In general, any inverting static gate, not just an inverter, may be used between dynamic gates. Domino gates precharge simultaneously, but evaluate sequentially. Therefore, the evaluation transition is generally more critical than the precharge transition. For best speed, the static gate should be skewed high to favor its critical rising output. If the static gate then drives another static gate instead of a dynamic gate, the next static gate can be skewed low, as shown in Figure 5. However, if too many static gates are included in series, precharge may become critical.

FIGURE 5. Dynamic and skewed static gates



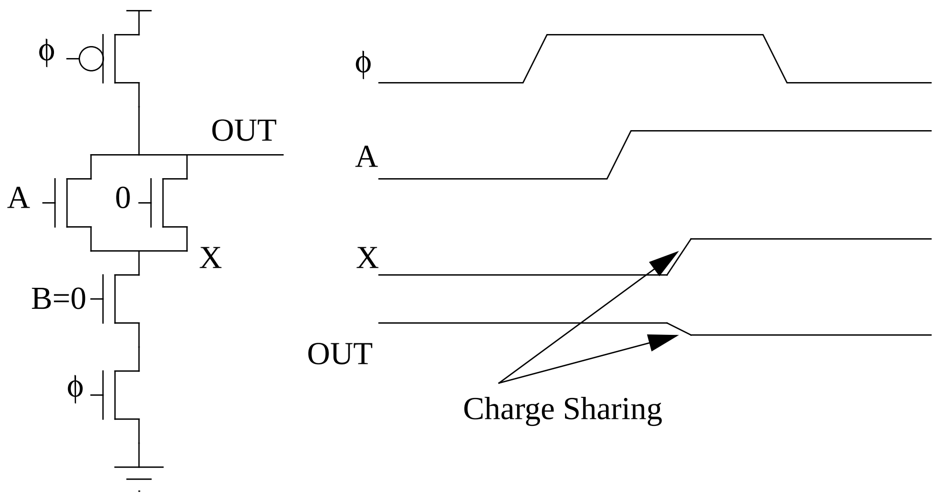
2.0 Domino Robustness Rules

Domino gates which obey the precharge and monotonicity rules will be logically correct. However, domino circuits are much more sensitive to noise than static circuits and therefore additional electrical rules are requires to ensure correct operation. If static gates receive too much noise, they can glitch, but eventually settle to the correct output. If dynamic gates receive too much noise, they can incorrectly evaluate and never recover.

2.1 Charge Sharing

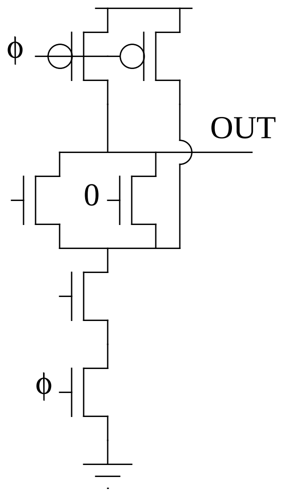
Charge sharing is one important dynamic gate failure mode. When a dynamic gate drives a small load, the internal diffusion capacitances may become comparable to the load capacitance. If the diffusion capacitances are low when evaluation begins, they may share charge with the load capacitance, causing the output voltage to droop from the capacitive voltage divider. For example, consider the dynamic gate in Figure 6. Suppose node X is initially low, perhaps from operation in the previous cycle. Suppose after evaluation begins, input A rises, but input B stays low. The capacitance on node X will share charge with the output capacitance, resulting in a dip on the output voltage. If the ratio of charge is too large, the output will fall by more than the noise margin of the next gate and produce an incorrect result.

FIGURE 6. Charge sharing



There are several ways to keep charge sharing to acceptable levels. One is to precharge internal diffusion nodes so that they will not disturb the output. This can be done with small "secondary" precharge transistors connected to the internal nodes, as shown in Figure 7. Each precharge device adds extra diffusion capacitance which slightly slows the gate, so it is best to only precharge the smallest number of internal nodes necessary to satisfy noise margins. A rule of thumb is that precharging every other node is sufficient for most gates.

FIGURE 7. Secondary precharge transistor



Another way to limit charge sharing noise is to make the load capacitance large, i.e. to have the dynamic gate drive a large fanout. Some dynamic libraries set minimum fanouts for the dynamic gates to limit charge sharing.

Of course, only certain dynamic gates are prone to charge sharing. A dynamic NOR gate can never experience charge sharing because it has no internal nodes. Conversely, a complex AND-OR-invert structure is most susceptible because it has large amounts of internal diffusion.

2.2 Coupling

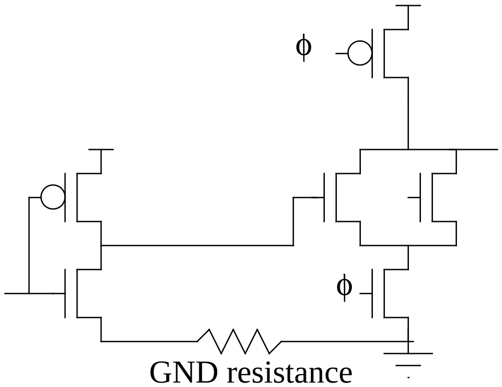
A second common problem with dynamic circuits is coupling noise. The output node of a dynamic gate is sensitive to coupling because there is no active transistor holding the output high to fight against downward coupling from adjacent switching lines. Moreover, since the receiver is a high skew gate, the noise margin is lower. The input of a dynamic gate is also sensitive because although there is an active gate driving the line, the noise margin is only V_t and a small amount of coupling is sufficient to turn on the dynamic gate and cause it to incorrectly evaluate low.

Every dynamic input and output therefore has a budget for coupling noise. To keep this coupling low, the designer can use a variety of tactics. If it is possible, a simple solution is to make adjacent lines only switch while the dynamic gate is in precharge and hence insensitive to noise. Keeping the wires short keeps the ratio of coupling capacitance to load capacitance small and reduces noise problems. Increasing the spacing between wires is also helpful.

2.3 Power Supply Noise

Another source of noise in dynamic gates comes from power supply variation across the die. Suppose a static inverter drives a dynamic input and that the ground potential of the inverter is higher than that at the dynamic gate due to IR drops across the ground network. If the ground difference exceeds Vt, the dynamic gate will turn on and incorrectly evaluate.

FIGURE 8. Power supply noise between static and dynamic gates

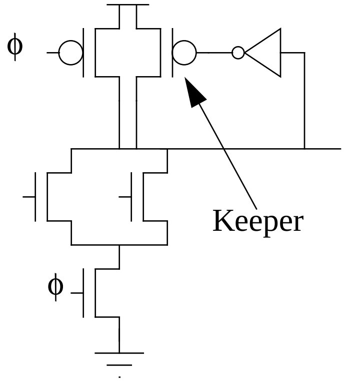


2.4 Subthreshold Leakage

When a dynamic gate is in evaluation with the inputs low and the output high, the output remains high due to charge stored on the load capacitance. All transistors leak slightly due to subthreshold conduction; such leakage may gradually drain charge off the load capacitance and slowly pull the output low. While this is not a problem at normal operating frequencies because evaluation will end long before much leakage occurs, it does present a problem for chips with clocks that may stop. Such cases arise if the clocks are stopped for scan, when a portion of the chip like the floating point unit is shut down to save power, or when the entire chip is placed in a sleep mode. It is worst for gates with wide NOR structures with many parallel leakage paths. It is also a growing problem because threshold voltages are scaling.

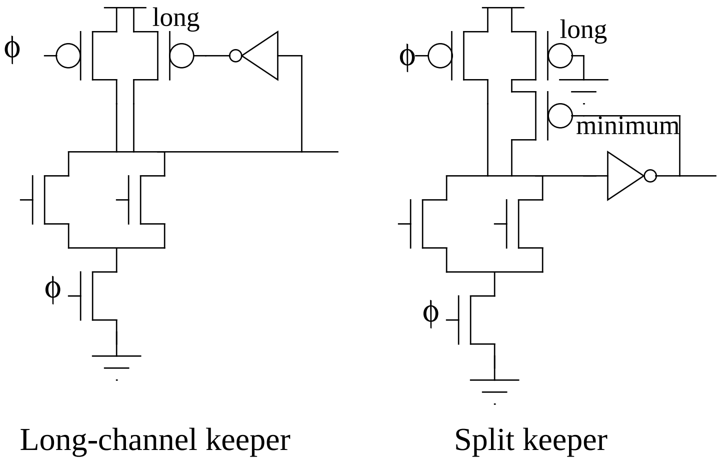
To counter leakage on nodes that may float high for extended periods, a dynamic gate may use a weak PMOS transistor called a keeper, as shown in Figure 9. The keeper should be sized as small as possible because when the dynamic output begins switching low, the keeper fights the NMOS evaluation transistors. The keeper should be at least a factor of 4-10 weaker than the pulldown transistors; such keepers typically increase the delay of the gate by 4-6%. After the dynamic output falls far enough, the inverter rises and turns off the keeper to avoid unnecessary power dissipation. If the static gate following a dynamic gate is an inverter, it may be used to control the keeper; otherwise an additional small inverter is necessary.

FIGURE 9. Keeper to counter subthreshold leakage



This keeper arrangement works well for dynamic gates with medium or large transistors. For dynamic gates with small transistors, even a minimum width PMOS transistor is stronger than desired for the keeper. Thus, the channel length of the PMOS transistor may have to be increased. However, if the keeper is driven by the same inverter which is driving the next stage, the loading of the long-channel keeper slows the inverter. To prevent this the keeper can be broken into two transistors, one being a long channel device that is always on and the other being a minimum size device controlled by the inverter. These variations are shown in Figure 10.

FIGURE 10. Keepers for small dynamic gates



Long-channel keeper

Split keeper

Some designers use keepers to reduce coupling noise or charge sharing. Unfortunately, small keepers are too weak to help very much in these cases. Such noise occurs rapidly, causing sudden dips in the output voltage. The keeper will slowly pull the output back to the correct high value, but by the time a weak keeper can help very much, the next stage has probably been tripped and will send a glitch through subsequent stages of logic. Therefore, do not expect much noise margin benefit from weak keepers.

In very noisy environments, it may be worthwhile to increase the size of the keeper substantially. This offers the possibility of trading off lower speed for better noise margins. For example, if a dynamic gate suffers noise on its input, a moderate sized keeper is sufficient to hold the output high when the NMOS devices have barely turned on due to noise.

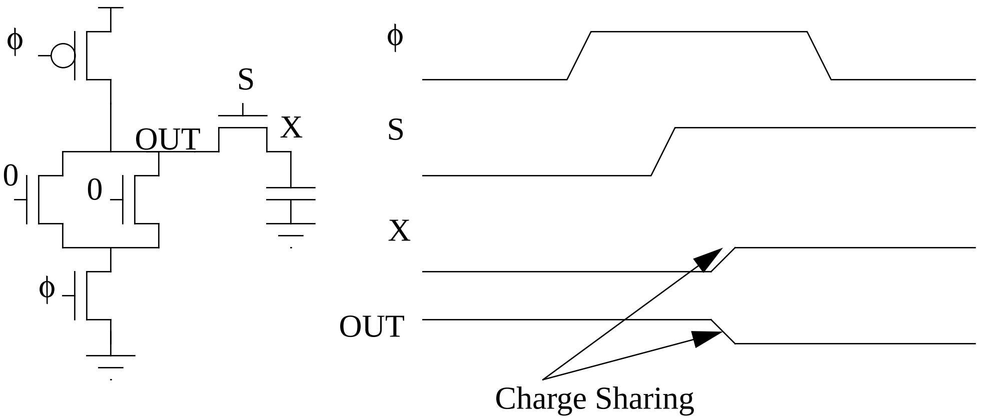
DEC’s design methodology for the Alpha 21164 sets a minimum operating frequency of 1/10 the nominal frequency. At this minimum frequency, leakage is only an issue on wide NOR gates; therefore, keepers are only used on wide NORs.

2.5 Pass Transistor / Dynamic Gate interactions

Combining dynamic gates with pass transistors leads to a number of electrical problems. Backdriving and charge sharing are the major concerns.

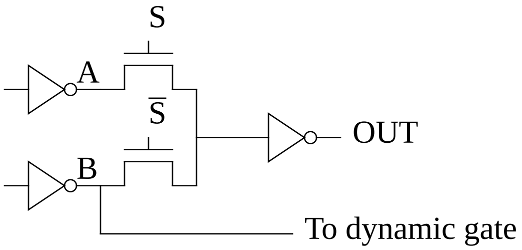
Charge sharing can occur if a dynamic gate drives the diffusion input of a pass transistor, as shown in Figure 11. Suppose the pass transistor is off during precharge and that the far side of the pass transistor is at zero volts. If the pass transistor then turns on during evaluation, the charge on the dynamic output will share with the charge on the far side of the pass transistor, corrupting the result.

FIGURE 11. Charge sharing with pass transistors



As discussed in the last lecture, back-driving of pass transistors can occur when two select signals are both momentarily high. If S and S are both high, the inverters driving A and B will fight. If A is high and B is low, the fighting will cause A to dip and B to rise slightly. This noise on B can falsely cause a dynamic gate also driven by B to evaluate.

FIGURE 12. Back driving pass transistors



The moral of these problems is that neither the inputs nor outputs of dynamic gates should ever be connected to nodes which also connect to the diffusion terminal of a pass transistor.

2.6 Charge Injection into Substrate

Certain circuits such as I/O drivers powering highly inductive loads are prone to ringing which may send the output below GND. In such a case, the drain-substrate junction becomes forward biased and injects charge into the substrate. This charge may be collected on nearby dynamic nodes, corrupting the value. To prevent this, circuits prone to charge injection should be placed far away from dynamic logic. They should also be surrounded by guard rings. Charge injection is only a concern with special-purpose circuits and therefore is not part of the noise budget of most dynamic gates.

2.7 Summary of Domino Electrical Rules

In summary, domino gates must be checked for the following electrical rules:

- Neither inputs nor outputs of dynamic gates should connect to diffusion terminals of pass transistors.
- Nodes which may float indefinitely must be held by keepers.
- Noise margins on dynamic inputs and outputs must be checked.

The noise margin rules can be broken down into dynamic inputs and outputs. Dynamic inputs have a noise margin of only Vt. Major noise sources are power supply variation and coupling. Dynamic outputs have a noise margin set by the high-skewed gate following the dynamic gates. Major noise sources are charge sharing, coupling, and power supply variation.

Noise margins on outputs can be increased by reducing the skew of the subsequent gate. Noise margins on inputs can be increased with a moderate strength keeper.

A detailed noise budget prevents the need for overdesign. For example, power supply variation is large across a die, but small within a local area. Charge sharing is large for some gates with tall stacks and few secondary precharge devices, but zero for wide NOR gates. If worst-case power supply and charge sharing noise were assumed for all gates, coupling noise requirements would be very stringent and would lead to heavy shielding or wide spacing of lines. If actual noise is computed given the gate topology and power supply routing, coupling noise limits may be relaxed.

3.0 Monotonicity Issues

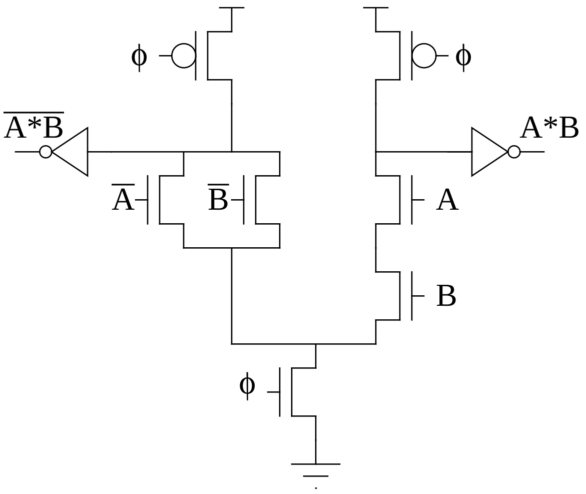
Standard domino gates can only implement non-inverting functions because every gate contains two inversions. Unfortunately, most interesting circuits require inversion somewhere. In particular, non-monotonic functions such as XOR cannot be implemented with a domino gate because the output might rise or fall in response to one input rising depending on the state of the other input.

3.1 Dual-rail Domino

A popular way to solve this problem is to use both true and complementary inputs and outputs, labeled _h and _l, respectively. The _h signal is asserted to mean a signal is 1. The _l signal is asserted to indicate a signal is 0. Neither _h nor _l are asserted when the gate is precharged and has not yet evaluated. If both _h and _l are asserted, an error has occurred. Such circuits are called Differential Cascode Voltage Switch (DCVS) or dual-rail domino.

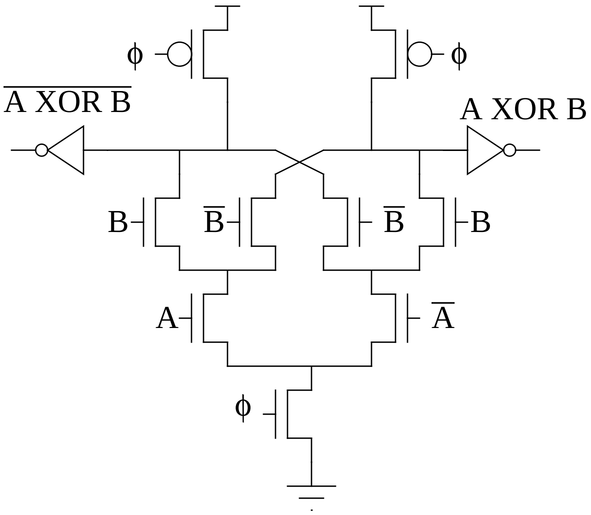
For example, a dual-rail AND/NAND gate is shown in Figure 13. It accepts inputs a_h, a_l, b_h, and b_l and produces out_h and out_l:

FIGURE 13. Dual-rail AND/NAND gate



The complementary output is built using DeMorgan’s law, so series transistors in one stack become parallel transistors in the other stack and vice versa. Notice how the clocked pulldown device can be shared between the two stacks. In more complex gates, additional transistors can be shared. For example, a dual-rail XOR/XNOR gate is shown in Figure 14. If sharing was not done, nine transistors would be required in the pulldown network. Since A and $\overline{A}$ are shared between the true and complementary paths, only seven transistors are necessary.

FIGURE 14. Dual-rail XOR/XNOR gate

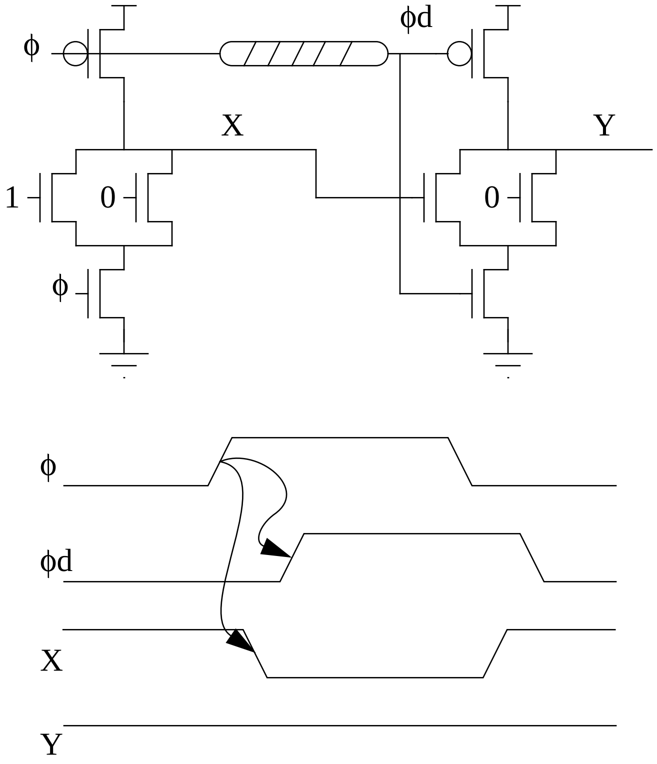


With dual-rail logic, all logic functions can be implemented. The drawback of any flavor of dual-rail circuit, dynamic or static, is the requirement of twice as many wires and almost twice as many transistors. As more transistors become available, transistor count becomes less of an issue, but wiring density is always a problem. Moreover, since dual-rail circuits use both complementary pulldown networks, wide gates cannot be efficiently implemented. In particular, wide NORs can be built well with conventional single-rail domino, but require tall pulldown stacks to produce complementary dual-rail outputs.

3.2 Self-timed Domino

An alternative to dual-rail domino is self-timed logic. Remember that the monotonicity rule only requires monotonic inputs while a dynamic gate is in evaluation. If we delay the evaluation clock to a dynamic gate until the inputs have settled, we can allow non-monotonic inputs. For example, consider a modification of the problem circuit from Figure 3, now shown in Figure 15:

FIGURE 15. Incorrect operation of cascaded dynamic gates

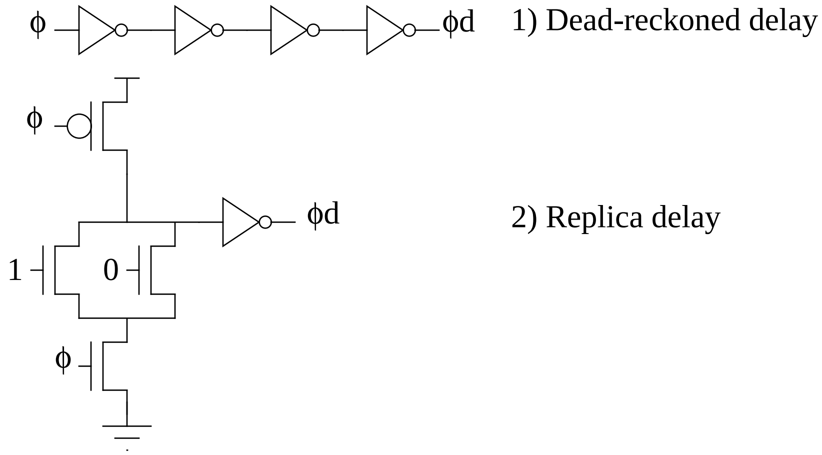


In this circuit, a magic delay element produces a self-timed clock φd which does not rise until node X has had time to evaluate. Thus, the second gate obeys the monotonicity rule and node Y correctly remains high.

The challenge of self-timed circuits is to produce clocks which arrive at exactly the right time. If φd arrived too early, Y would incorrectly pull low. If φd arrived too late, the circuit will run more slowly. Since it is impossible to exactly match two delays, the designer must always make φd nominally arrive slightly late so that if the magic delay turns out to be shorter than expected, the circuit will still work. Unfortunately, if φd then arrives later than expected, the circuit will be slowed further. The difference between the delay to X and the worst case delay to φd is called the self-timing margin.

Self-timing margin depends on how well magic delays can be made to track with the actual gate delay. Two magic delay generators are shown in Figure 16. The first one uses inverters, which may change speed much differently than dynamic NOR gates over process and environmental corners. Self-timed margins of 30-50% are usually necessary; insufficient margin results in chip failure at any clock speed. The second delay generator uses a replica of the circuit it is attempting to match. This generator will track much better and can use a reduced margin, such as 10-20% plus the delay of the inverter. Replica delays should always use the worst-case input patterns (in this case, only one of the NOR inputs being high), and should match the layout and loading of the circuit they must track.

FIGURE 16. Delay generator circuits



Self-timed circuits are especially popular in dynamic PLAs. A NOR-NOR PLA consists of an AND plane and an OR plane constructed from wide NOR gates. The OR plane must not begin evaluating until the AND plane has finished. This can be done by adding an extra row to the AND plane with worst-case loading; it’s completion can kick off the self-timed start of the OR plane.

Self-timed circuits are also used in SRAM sense amplifiers and in certain other special-purpose non-monotonic dynamic logic. They are best to use when the advantages of wide NOR structures and compact gates outweigh the cost of the self-timing margin. Wide-scale self-timing has not yet been proved on large chips and is somewhat frightening because insufficient self-timing margin leads to irrecoverable chip failure.

3.3 Application: Integer Execution Unit

Lets consider how monotonicity issues impact a real circuit, such as an integer execution unit. Consider a six-way superscalar processor with four integer pipes. Each integer pipe contains a 64-bit adder which must run as fast as possible.

Fast adders are usually constructed with logarithmic carry select structures which are monotonic until the last gate, which is a multiplexor requiring true and complementary select lines. When designers used fully static circuits, the adders have no monotonicity issues. When designers first adopted dynamic circuits, they often built single-rail dynamic logic to implement all but the final stage, then used a static inverter and static multiplexor to select the output. Unfortunately, the transition back from static to dynamic logic, needed to use the result in the next cycle of computation, adds a budget for clock skew to the critical path, as we will learn in a future lecture. Therefore, there is strong incentive to build the adder entirely with dynamic logic.

The fastest reported adders, such as HP's Ling adder, are now built with fully dual-rail domino. The adders take true and complementary inputs and produce true and complementary outputs. Such adders have latencies of around 7 FO4 inverter delays.

Unfortunately for our super-scalar processor, each integer pipe requires two source busses and a result bus, for a total of 12 busses in the entire integer execution unit. If each bus is constructed with dual-rail logic, 24 metal lines are required in each bitslice. To reduce coupling, these lines may be run with greater than minimum spacing or even with shielding. Wire count on such datapaths is becoming a serious issue.

An alternative is to build single-rail logic with a self-timed clock on the final multiplexor which enables it when the non-monotonic inputs have settled. If the single-rail logic is just one half of the dual-rail design, the delay will increase significantly because the single-rail and dual-rail logic delays are almost the same but the single-rail circuits add a hefty self-timing margin. Instead, the single-rail logic may be redesigned to employ several stages of wide self-timed dynamic NOR functions which implement the carry chain taking advantage of the speed of dynamic NORs.