High Speed CMOS VLSI Design

Lecture 11: Clocking

1.0 Introduction

We have seen that generating and distributing clocks with little skew is essential to high speed circuit design. This lecture explores the issues involved and the sources of clock skew.

2.0 Clock Generation

Routing a single clock around a chip is a difficult problem. Routing multiple clocks with little skew between the clocks is even harder. Since most circuits two or more phases locally, the usual practice is to distribute a single global clock, then locally derive the multiple phases near where they are necessary. “Near” is usually defined by the maximum wire length a final stage clock buffer can drive before introducing unacceptable RC skews. This is on the order of 2 mm and gets smaller as clock frequencies increase and skews must decrease. Let us first look at the global clock generation, then explore local clock generation.

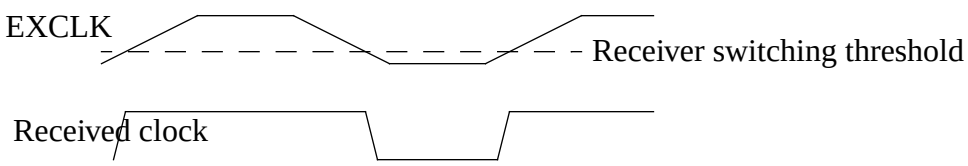
2.1 Global

At first glance, it seems a global clock could be produced by simply buffering the input from an external oscillator. There are two difficulties with this technique: phase alignment with the external clock, and duty cycle variation. These difficulties lead most designers to use a phase-locked loop (PLL) or delay-locked loop (DLL) in their global clock generator.

If a chip had no I/O, delays between the external clock and on-chip clock would not matter. However, if the chip communicates synchronously with other chips, the clocks of the various chips must be aligned. A common way to do this is to align all other chip clocks to the external reference with a PLL or DLL.

Even if alignment were not a problem, duty cycle uncertainty would be a challenge. Distributing a high-speed external clock over a PC board and through an IC package is challenging. Since the edge rates are not very sharp, it is difficult to identify exactly when a clock transitions. If the threshold detection is not at exactly 50%, there will be duty cycle variation in the detected signal, as shown in Figure 1.

FIGURE 1. Duty cycle variation in received clock caused by receiver threshold offset



To reduce this duty cycle uncertainty and more easily distribute fast clocks, the clock is often sent as a differential small-swing signal. A differential amplifier on the chip must restore the clock to a full-swing single signal. Offset voltages in the amplifier will still cause certain amounts of duty cycle uncertainty.

One way to eliminate duty cycle uncertainty is to provide an external double-frequency clock, then divide by two with a toggle flip-flop to produce a normal frequency clock with exactly 50% duty cycle. However, providing even a 1x clock at high frequency is hard, so providing a 2x clock has gone out of favor. Often a PLL is used to adjust the duty cycle. DEC does not use any loop on the 21164 Alpha, but instead uses a duty-cycle correction circuit which measures the duty cycle of the received clock and adjusts the generator to compensate, reducing duty cycle variation from +/- 5% to +/- 0.5%.

There are many ways to build loops to adjust the clocks. Usually an expert loop designer knows the secret incantations to make robust, reliable loop circuits. We’ll just explore the basics to understand some limitations of loops. All loops take a reference clock and use feedback control to create another clock with a particular phase and frequency relationship to the reference.

Since no loop is perfect, the output of the loop may have some phase error relative to the desired output. This error can change over time; the change is called jitter. Important specs on jitter are “cycle-to-cycle” and total. Cycle-to-cycle jitter is the maximum change in phase from one cycle to the next. It has many of the same characteristics as clock skew. Total jitter is the maximum variation of phase over many cycles. Two independent loops may vary in phase by total jitter.

Phase-locked loops use a voltage-controlled oscillator. The control voltage is adjusted until the PLL is oscillating with the same phase and frequency as the reference clock. The PLL can also do frequency multiplication, creating output frequencies of rational multiples of the reference. An error in the control voltage impacts the frequency of the output. Since phase is the integral of frequency, errors can accumulate over multiple cycles. This means that total jitter can be quite large.

Delay-locked loops use a voltage-controlled delay line. The reference clock is delayed to produce an output with the desired phase offset. For example, a phase offset of 360 degrees gives an output exactly aligned to the reference. Simple DLLs cannot do frequency multiplication. However, errors in the control voltage only impact the delay, not

frequency, so jitter does not accumulate over multiple cycles. DLLs therefore usually have better jitter performance.

Since noise on the control voltage causes jitter, it is important to remember that loops are really analog circuits and must be shielded from the noisy power supplies of digital chips.

2.2 Local

Once the global clock is generated and shipped to various units around the chip, local clock phases can also be generated. Local clock generators serve to buffer the huge clock load and to reshape the clocks. Examples of shaping include delaying edges, inverting the clock, and producing pulses or longer duty cycle waves.

Local clock generators require careful design to minimize the skew they introduce. Issues include input slope, delay matching, and layout matching.

The final driver should produce sharp edges so that variation of input trip-points on clock receivers doesn't show up as large clock skew. This usually means the final local buffer should be a fanout-of-3 inverter. The gate should be sized for equal rise and fall time to avoid duty cycle errors.

The other buffers in the local clock generator should be designed to match well across process variation so they don't introduce skew. For example, if one local clock generator used a 3-input NAND and another used a 3-input NOR, the local clocks could be skewed significantly in the FS and SF corners. Ideally, one could use only inverters in the buffer network, but most real chips need some amount of clock gating with a 2 or 3-input NAND gate. NOR gates are best avoided because they require huge PMOS transistors to drive large loads with equal rise and fall times.

Good layout of local clock buffers is necessary for low skews. Mask misalignment differs in the horizontal and vertical directions, so all transistors should be drawn in the same direction in clock buffers for best matching between buffers. Polysilicon is usually etched with plasma. The etch rate has some dependence on the total amount of material which must be removed. Therefore, when a large amount of poly is etched in a local region of the chip, the rate is slower and channel lengths of the remaining poly are longer than nominal. When a small amount is etched, the rate is faster and the channel lengths are shorter. To get closely matching transistors across the chip, the average polysilicon density in a region should be kept nearly constant. Sometimes, extra polysilicon wires are drawn around buffers to raise the density in sparse regions.

Delay chains in local clock buffers to create overlapping clocks will vary with process and do not track with clock frequency. Thus, a nominal 1/4 cycle delay at full frequency becomes only a small fraction of the cycle delay at low frequency. An alternative to such open-loop delay chains is to use a delay-locked loop in the local clock generator which can produce clock phases of arbitrary timing relationship independent of operating frequency and process parameters. Unfortunately, since it is infeasible to route many clocks all over the chip, a single global clock must be distributed to many local DLLs. Two clocks

in different local clock domains therefore see the worst case jitter between the independent DLLs. This jitter is usually far worse than the open-loop skew.

3.0 Clock Distribution

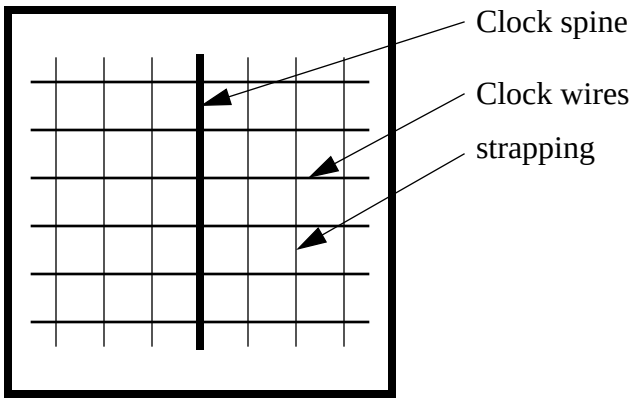
Distributing the clock is another challenge. Again, we can divide the problem into global distribution, from the phase locked loop to the local clock generators, and local distribution, from the local clock generators to clocked elements.

3.1 Global

When transistors were much slower than wires, clocks could be routed about the chip in an ad-hoc manner with little penalty. Now that wire delay is important, designers want to minimize the variation in wire RC delay between points on the chip. Popular global distribution networks include grids, H-trees, and hybrids.

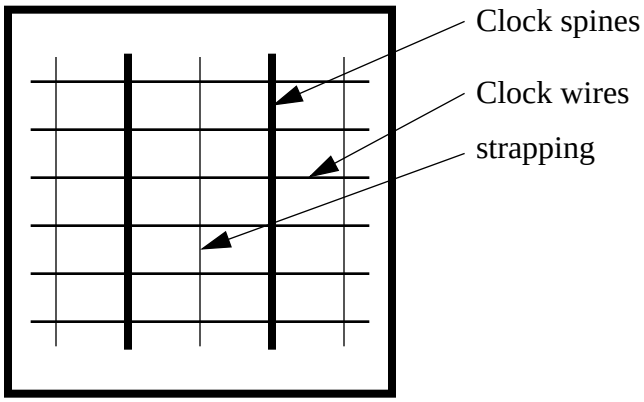
The Alpha 21064 demonstrated a clock grid, as conceptually shown in Figure 2. The final clock driver uses 35 cm of transistor width (!) to drive a 3.25 nF clock load. The clocks are driven horizontally from a clock spine at the center so there is very little skew near the spine and more skew near the edges. The chip is floorplanned so that most skew-sensitive blocks are as close to the spine as possible. The clock lines are strapped vertically. While this does not impact the RC delay from the spine horizontally to a load very much, it does guarantee the local skew between two points on adjacent horizontal lines stays small. The reported RC delay skews are under 300 ps.

FIGURE 2. Alpha 21064 Clock Grid



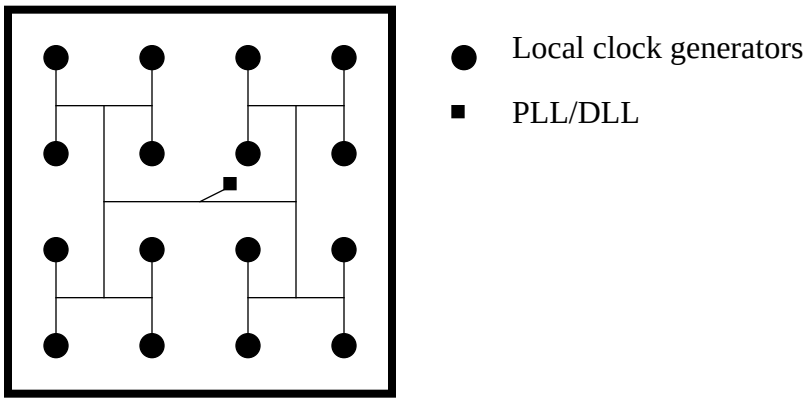
The Alpha 21164 [JSSC Nov. 96] finds that the skew from the spine to the edge of the chip is too large. Thus, it uses two clock spines, so that global clock lines never must travel more than 1/4 the length of the chip. The reported RC delay skews are 90 ps.

FIGURE 3. Alpha 21164 Clock Grid



Even with such a grid, a chip will see systematic skews caused by the RC delay driving a heavy load 1/4 of the chip away. An alternative is the H-tree network, shown in Figure 4. An H-tree is a recursive space-filling curve with beautiful mathematical properties that CAD researchers like. A key property is that the distance from the center of the tree to any end is equal. Therefore, the global clock can be distributed to local clock generators at each endpoint with nominally zero RC skew. Unfortunately, difference in loading presented by each generator leads to skew. This skew can be reduced by adding dummy capacitance to the lighter-loaded endpoints, at the expense of power consumption. H-tree layout is also a problem. If the tree were routed in a single layer, it would prevent any long signal routing in that layer. If the tree were routed in two layers, many vias are necessary to change layers each time the wire direction changes.

FIGURE 4. H-Tree Clock Distribution



The H-tree works reasonably well for course-grained distribution, but requires many branches for the final distribution. The Alpha 21264 [ISSCC97 slide supplement] uses an H-tree to distribute the clock to 16 regions around the chip, then uses a grid within the regions. They claim an RC skew of under 50 ps with this technique.

3.2 Local

Locally, clock distribution depends strongly on the actual logic which must be clocked. A good local distribution scheme will keep local wires short to minimize RC delay. In random logic, it may be best to layout all clocked elements in a small region to avoid a rats nest of clock wiring around the synthesized block. In datapaths, local clock buffers may sit on the edge of the datapath and drive their clock across the bits. If the RC skew across 64 bits of a datapath becomes unacceptably large, it can be cut in 4 by driving the datapath from either the middle or from both edges. Routing signals into the middle of the datapath is often inconvenient, so driving from the edges is usually more popular.

4.0 Skew Summary

Measuring total clock skew requires accounting for many components. Most companies presenting chips at conferences neglect to report some components in order to make their specs look better than they actually are.

Any system which uses a loop must deal with cycle to cycle jitter. This jitter can shorten the cycle time, so it has many of the same effects as clock skew. However, almost nobody reports jitter in their skew budget.

Clock skew results from variation in the global distribution, variation in the local buffer, variation in the local distribution, and variation in the switching point of the receiving gates. This variation results from both differences in RC delay and differences in gate delay. DEC makes pretty plots of the maximum difference in RC delay between points on a chip under otherwise nominal conditions and calls it their clock skew. This is very misleading.

The distribution skews are minimized by making the length and load on all global distribution lines as close to equal as possible and by immunizing the length (and hence RC delay) of local distribution lines. It is also important to limit the amount of buffering necessary, since buffers scattered around the chip see different voltage, temperature, and processing and hence have different delays. Finally, matching the local buffers in fanout and structure is vital.

Even doing everything right, reducing skew below 200 ps is very difficult. As processes scale, the raw gate delay improves, which benefits buffer delay variation. However, wires get longer and wire RC per length increases, making wire skew more severe. Also, process tolerances will probably get sloppier, making matching of devices worse across a die. To a certain extent, these problems can be countered by increasing the wiring and power resources dedicated to the clock network.

For current designs, skews are a small enough fraction of the cycle that they can be hidden with transparent latches or skew-tolerant domino. As cycle times shrink more rapidly than clock skew, it may become impossible to tolerate global skew. One solution is to eliminate the clocks, using asynchronous logic. We will look at asynchronous circuits in the future,

but such circuits have many severe problems of their own. A more immediate solution may be to operate local regions with better-controlled skew at high frequency, then communicate globally at lower frequency so that the skew is a smaller fraction of the cycle.