SoC Chapter 8: Fabrication

Modern SoC Design on Arm
End-of-Chapter Exercises

Q1 Chip Fabrication

List and describe the main layers of a modern silicon chip.

Answer:

First generation CMOS chips used very few layers. Wiring was formed with one layer of metal and one layer of polysilicon. Two layers of wiring is the minimum sensible, since otherwise nets cannot cross. Two types of diffusion were required for N and P channels, with transistors being formed wherever polysilicon crossed diffusion. It is worth being familiar with this classic 'Mead and Conway' VLSI stick diagrams from their 1980's book *Introduction to VLSI Systems*.

Today's chips have up to ten layers of metal, but the basic principles have hardly changed: all the active semiconductors (transistors) remain in a single shallow layer just under the top surface of the staring silicon wafer substrate. Although today, often copper is used instead of aluminium owing to its lower resistivity. Tungsten is also used. The lowest metal layers have a pitch similar to the node's geometry whereas the top layers are much lower resolution and are used mainly for power distribution. The intermediate layers are used for long distance inter- connection. See Figure 8.5 *Basic layers in planar CMOS VLSI*.

As of 2023, we can expect to see greater use of the third dimension. Although die stacking is common, within the die itself we can expect to see transistor stacking and power wiring moving to the other side of the wafer.

Q2 Place-and-Route

What problems can be found during net routing that would suggest a better placement is needed? How can these be anticipated during placement? Would a constructive placer take these considerations into account?

Answer:

Hotspots can occur with some placements that are eliminated by changing the placement. A 'hotspot' can be defined as any area that many nets need to pass through. Moving components further apart can always make more room for nets, but simply exchanging the sites for components of roughly the same size can be sufficient to eliminate hot spots. Additionally, with a given placement, some nets may be too long for timing closure or drive strength requirements. If fewer layer crossings are needed the area consumed by vias is reduced.

Mathematically speaking, if there are thousands of components, place-and-route selects a projection of a many thousand-dimensional structure (the netlist) into just over 2 dimensions. If the dimensionality mismatch were less, hotspots could be computed with precise equations. Instead, a numerous algorithm from graph theory must be heuristically combined to work

out which components have the greatest 'centrality'. Hotspots are anticipated and avoided by placing these components first.

A constructive placer uses heuristics to place each component in turn with little or no backtracking. Once all components are placed, a stochastic optimization process is typically deployed, that makes random swaps of component sites. So yes, both stages will reconsider wiring length after each operation. The nets are not routed at this point, so an as-the-crow-flies or Manhattan distance for each net might be used. Nets that feed multiple destinations have more than one potential structure and a heuristic metric that averages out the various possibilities may be used. In all algorithms, the placement will be strongly guided by net length and whether any maximum length limits are being approached. The density at a particular spot or routing channel will also be considered.

Physically aware synthesis tools may have generated or read-in floorplan guidance which the placer will take into account. Manual adjustments to the auto-placer output are also very common. See the answer about floorplans below.

Q3 FPGA vs ASIC

Why is an FPGA larger and slower than the equivalent ASIC?

Answer:

Software is slower than dedicated hardware owing to the 'Turing Tax': the overhead of making a general-purpose computer take on a specific role. The tax is manifested by instruction fetch and decode logic (the CPU control unit). Likewise, when a general-purpose FPGA chip takes on a specific application, overheads arise from its programmability.

In addition, FPGAs, like all gate arrays, are available in various fixed sizes and a larger chip than is strictly needed will be selected owing to 'round-up' effects. Moreover, a significant proportion of the silicon in an FPGA contains configuration logic (SRAM cells mostly) that are loaded at power-up time with the current design. This area is not needed in an ASIC. Also important is that the wiring on a FPGA consists of many pass-transistor-based multiplexor structures. The delay through a pass transistor is far less than a logic gate, but the channel resistance accumulates, adding to the effective net resistance. This increases the R/C net delay. And because the chip is larger than the equivalent ASIC, all nets are longer, probably by a factor of about 3.

The flip-flops and other components that form the active logic of the FPGA can themselves be of a similar performance as an ASIC. This holds since most FPGAs are made at nodes and fabs that are also used for ASICs. However, they may be more complicated than the ASIC equivalent, such as every flip-flop having a clock enable and reset and debug access etc.

Similar arguments apply to multipliers and large RAMs available in the programmable logic pool of most contemporary FPGAs.

Q4 FPGA Multiplication

How many FPGA DSP blocks are needed for a 32 × 32 multiplier? What is its latency? What difference

does it make if only 32 bits of the result are needed?

Answer:

The answer details clearly depend on the type of FPGA in use. Other variations arise from time/space folding where one DSP block is used for different parts of a long multiplication on successive clock cycles. [We do not consider more complex approaches, such as the Karatsuba algorithm, or make the trivial point that no DSP units are strictly needed, since multiplication can also be performed in everyday logic — bit serially indeed!]

In this answer we follow one popular FPGA family and assume the availability of a DSP that can natively perform a $24 \times x17$ multiplication, fully pipelined, with a latency of 3 and an output accumulator for partial products. [Other popular FPGA families do support 32×32 natively, so the answer would be unity for these.]

Since the native multiplier has no input equal to the input width, 4 multipliers will be needed in a long multiplication arrangement. It is probably easiest to make each of these 16×16 but an arrangement based on $(24 + 8) \times (15 + 17)$ could also be used. For unsigned operation, the bit extracts of the sub-words from the arguments are trivial wiring with no delay. Each of the four partial products can be performed in parallel on four DSP units, taking 3 clock cycles each, followed by a four-input adder that operates in one clock cycle. But for many applications, having a little more pipelining may be acceptable and is likely to work out cheaper in silicon area. Using one DSP unit over four successive clock cycles can exploit the integrated accumulator, summing one (shifted) partial product per cycle. The total latency would then be extended from 4 to 7 clock cycles. NB: the component would not be 'fully pipelined': a new argument cannot be accepted every clock cycle since a single accumulator is not sufficient to handle multiple running sums. [An intermediate design point would perform two products in parallel, requiring three-input addition.]

The question does not say which 32 bits of the 64-bit result are needed, but, following many ISAs, this would often be the low 32 bits. The multiplication of the high 16 bits of each operand by each other is then not needed. Also, signed multiplication is the same as unsigned for the low 32 bits of the answer.

On the other hand, if the integer arguments were thought of as the numerators of rational numbers with denominator 2^{31} (i.e., an approximation to the real numbers in the range $-1.0 < x < 1.0$), then their product never overflows and it would be the high 32 bits wanted as the result. This would need three partial products to be summed.

Q5 Production Test

Design a logic structure that will be very difficult to assess in a production test, but which does not include redundant logic. What is the problem? Could such a structure be needed in a real application?

Answer:

IC production test occurs twice in general: at wafer probe time and post-packaging. As explained in §8.8.2 *Test Program Generation*, a succession of test vectors is applied by a test ma-

chine in both cases. Since test machines are expensive, a short test program is always preferable. [Completed board and system testing is outside the scope of this chapter.]

There are many logic structures that are not amenable to vector tests, especially those containing a lot of RAM. In general, anything that has a lot of internal complexity and few external connections requires special attention to testing. To facilitate testing, logic that is redundant during normal use is often included. This includes built-in self-test logic (BIST), scan chains, JTAG ports, and other specific test modes that arrange the data paths into a form easier to test (e.g., the ROM and RAM in a micro-controller becomes directly addressable through the IO pads of the chip).

A secure processing unit, such as a smart card (chip-and-pin card), or electronic keys in general, will be hard to test with vectors. Many smart cards contain an integer version of the Java VM and a small operating system. A smart card necessarily has very few external connections. Such a device is typically constructed with tamper-proof techniques or design obfuscation to make reverse engineering hard. Making test pins available would instead facilitate reverse engineering! So, the surface available to connect test harnesses must be kept very low, which makes any such device intrinsically harder to test.

A chip or board with exposed JTAG or production test strappable links is liable to reverse engineering through its test modes. This has been demonstrated on some supposedly secure products. One countermeasure is a one-time-fuse: the device is fabricated in test mode, but once tested, the tester 'blows the fuse', stopping further operation in test mode. Many of the zapping techniques answerable for the question '*Yield Improvement Through Redundancy*' be- low are generally applicable, but for security against reverse engineering, those necessarily on the top-layer of the chip, such as ion-beam edits, are not ideal.

Another approach for security applications is backdoor keys. A 'super key', known only to the manufacturer, may be passed down the narrow interface to enable test modes. For instance, it may enable a machine code monitor CLI where memory and I/O locations can be examined. Using PKE techniques, the super key does not have to be physically stored in the device; only the matching public key needs be held.

Once the selection of active subsystems is complete, the redundant systems are no longer accessible and do not need further testing. Hence, we are back to the situation present in the first part of this answer.

Q6 Floorplanning (also Clock and Power Domains)

What principal data needs to be held in a floor plan? Can a good floor plan reduce the number of domain crossing and isolation components needed?

Answer:

A floor plan for a chip is set of tessellating rectangles that divides a (2D) squarish chip into oblongs. It is generally created manually. Most chips are represented by a single RTL top-level module that instantiates all of the subsystems. Each instance at that level has an instance name.

In basic terms, the floorplan will map each instance name to one of the oblongs.

The floor plan also typically contains the coordinates of the I/O bond pads, since these cannot be freely moved during placement without changing the device package. Also, I/O pads that are closely related, such as the two nets of a differential pair, or the set of nets making up an Ethernet PHY or DRAM channel, need to be closely placed for power supply and noise management reasons. This also facilitates PCB layout. For a small ASIC or a large FPGA, the floorplan is often read in by the auto-placement tool. For larger designs, the placer tool will be separately run for each oblong to lay out the components within it. The floorplan is the first step towards making sure that local interconnect within a subsystem is relatively short.

If chiplets are in use, there would be a floor plan for each chiplet. This needs to be additionally populated with the inter-chiplet bond pads.

Yes, the floor plan will be designed with close attention to the power and clock domain allocation. Power and clock distribution is easier if oblongs that require the same power and clock rails are adjacent. They form a common zone.

To power gate a subsystem (turn it on or off), the power distribution network needs to be partitioned accordingly. This is easiest if all of the parts of a subsystem are co-located in a floorplan oblong. The same goes for clock gating. Both power and clock nets need special attention in their design and layout. They can typically span the entire chip. Power nets must connect to on-chip decoupling capacitors and clock nets need to have low skew. Clocks may be driven by a nearby phase-locked loop to regenerate the reference clock.

The other inter-domain bridging components needed are clock gates, clock bridges (CBRIs) and isolation gates. Clock gates simply turn off a clock, driving the clock net to a fixed value without generating runt pulses. CBRIs avoid metastability and guarantee complete word/-packet integrity using padding cells between each datum. Power isolation cells avoid short-circuit currents arising when undefined input voltages arrive on the floating nets sourced by a turned-off region.

Q7 Slew Rate Limiting

Choose one of the reasons listed in the book for limiting the transition times in a design and expand upon the reasoning with examples, simulations or mathematical modelling. Why is the transition time especially important for clock signals?

Answer:

The slew rate of a digital logic transition can be defined in terms of voltage velocity, in V/s, (i.e. $\frac{\mathrm{d}v}{\mathrm{d}t}$). One can quote the average rate while a net is transitioning between V_{OH} and V_{OL}, in either direction. This is a useful metric, as is the maximum rate.

Some people might think a low slew-rate will intrinsically save energy, since a 'higher power' driver is required to change the potential of a capacitive net quickly. But changing it slowly requires the same amount of charge to drop the same amount of potential, so the energy used remains the same, even though the instantaneous power is less. Moreover, a receiving CMOS

circuit will waste more short-circuit energy when processing a low slew rate signal, as the period of time between nominal logic levels is increased. On the other hand, high slew-rate drivers tend to have higher static power dissipation, so there can be a saving in that respect.

The book mentions three further benefits of reducing slew rate: minimising on-chip power supply noise, minimising emitted RFI and minimising cross talk between one signal net and the one(s) next to it.

We now give some back-of-the-envelope style numerical estimates for each of these three effects. This simple analysis is within the scope of most EE degrees but will likely be unfamiliar to those who have specialised in digital logic and computer science.

Regarding supply noise: The current in a capacitor is given by $I = C\frac{\mathrm{d}v}{\mathrm{d}}$. By reducing the slew rate, the current is smaller (but lasts for a longer period). When charging the output capacitance, the voltage drop in a power supply network depends on its resistance and inductance. The ground bounces up in a similar way when the output is being discharged. The PDN resistive drop is proportional to the amount of current and the inductive drop is proportional to its rate of change.

Let us consider a 3V3 output pad on a chip that is fed by a power ring which is 1mm in each direction from the nearest on-chip decoupling capacitor. We assume an aluminium power conductor (typically on the top metal layer) that has square cross section of $25\,\mu m$ wide by $25\,\mu m$ high. The two sections of ring feeding it are identical and can be assumed to have little mutual inductance, so their effective inductance and resistance is divided by two. The ground supply is assumed to have the same impedances, but only one of the supplies is relevant, depending on whether the output is charging or discharging.

The resistivity of aluminium, ρ is 2.7×10^{-8}, so by the standard physics formulae, $R = \rho l / d^2$ we have 0.04Ω in each of the adjacent ring segments. These are effectively in parallel, giving 0.02Ω.

Resistive voltage droop is proportional to the instantaneous current. If the output pad feeds a load capacitance of 5 pF and supports the two slew rates shown in the table below, we compute the average current and the maximum resistive droop/bounce values as shown.

	Transition time (ns)			
0.2 to 3.1 V	5	ns	50	ns
Slew rate	580	V/us	58	V/us
Average charge/discharge current	3	mA	0.3	mA
Resistive droop	63	μV	6	μV
Inductive droop	2	mV	20	μV

Precise calculation of the inductance of the power ring segment is difficult without full modelling of the various return paths. But to an order magnitude, the partial inductance of a wire link of length L is $\mu_0 \cdot L$ where $\mu_0 = 4.\pi \times 10^{-7}$. Hence our links have a partial inductance of about $1\,nH$. It is a partial inductance since inductance can only properly be computed by also considering the return path in the circuit.

Voltages developed across an inductor depend directly on the rate of change of current ($V = L \cdot \frac{\mathrm{d}I}{\mathrm{d}}$). Figure 8.37 shows a typical current profile. By eye, its maximum rate of change can

be estimated as about one third of its average value during one tenth of its duration. Hence, $\frac{\mathrm{d}i}{\mathrm{d}}\max = \frac{I_{\max}/3}{\text{transition time}/10}$. These figures give the inductive droop results in the above table. Because both the average current is higher and the transition time is shorter, a quadratic effect from slew rate emerges.

Both forms of droop/bounce are less than a few millivolts, so are acceptable. The supply is well designed and has sufficient decoupling. We see the inductive drop is more significant than the resistive drop in this analysis. We should note that the resistive elements of the whole PDN sum whereas only the inductive drop from the nearest decoupling capacitor is significant. The spur from the ring to the pad would drop additional voltage, but this would not be considered power supply noise and again it is negligible.

If a large number of nets all switch at once, all fed from the same local supply, the accumulation of the inductive effects can become significant. Hence modelling with far more accuracy than sketched here is always warranted.

Regarding radiated RF interference: the FCC consumer specification is not to exceed 15 μV/m at a distance of 30 metres over all spectrum below 1 GHz. Converting this to a total power, by using $P = V^2/Z_0$ where $Z_0 = 377\Omega$ and multiplying by the surface area of a 30-metre sphere, we have $0.6\text{pW}/m^2 \times 11304\text{m}^2 = 10\text{nW}$. It is exceptionally easy to inadvertently radiate that level of RF energy with a short stub of wire. For instance, a sticking up conductor of 1 cm or equivalent unterminated pcb trace without a ground plane beneath acts as a 'monopole' antenna. It would radiate perfectly at $\frac{1}{4}$ wavelength frequency of $3 \cdot 10^8/(0.01 * 4) = 7.5\text{GHz}$. Indeed, a sine wave of 2.9 V peak-to-peak at one end would emit a power of $\left(\frac{2.9}{2\sqrt{2}}\right)^2/37.5 = 28\text{mW}$ of radio power, which is 64 dB above the 10 nW figure! (That was computed by converting to r.m.s. voltage and using $P = E^2/R$ where the resistance is the textbook figure for monopole antenna matching (half a dipole)).

7.5 GHz is well above SoC clock frequencies, but digital logic signals have harmonics owing to their square edges. A random NRZ bit sequence has a sync-shaped power spectrum, so rolls off reciprocally. Similarly, the monopole's radiation efficiency decreases when the wavelength is longer relative to its length, falling off quickly (40 dB/decade) when λ is less than $\frac{1}{1}^{th}$ of the monopole length. The $\frac{1}{1}^{th}$ frequency would be 1.875 GHz for a 1 cm stub, which is roughly the fifth harmonic of a 400 MHz NRZ signal. Hence the emissions would still be $64 - 20 \cdot \log(1/5) =$ 50 dB above the required limits.

Even following good layout approaches, where all stub or whip like structures are deliberately avoided (unless making an antenna of course!), using a lower slew rate greatly helps suppress emissions. Consider slew rates S at the two values mentioned above (normal 580 V/μs and limited 58 V/μs), by equating zero crossing gradients under the formula $S = 2\pi f A$ where A is the amplitude (half the peak-to-peak voltage) we obtain equivalent sine wave frequencies of 130 MHz and 13 MHz. The order-of-magnitude reduction in frequency content reduces the monopole emission efficiency a lot (maximum is 40 dB/decade × two decades = 80 dB), which is well worth having.

Regarding induced effects in neighbouring nets: Signal crosstalk arises when two nets run

together for a long distance in parallel. Because of the parasitic capacitance and mutual inductance between them, if one makes a transition the other experiences an induced noise voltage. Both effects are exacerbated if the transition has a high slew rate, since they are respectively proportional to the rate of change of voltage and current in the conductor.

In general, as well as slew-rate limiting of pads, slightly skewing the times at which output pads change by a small fraction of a clock cycle helps spread out the noise generation in the time domain, reducing its peak amplitude. It is the peak amplitude that can lead to logic margin violations (especially 'ground bounce').

For clock nets it is undesirable to reduce the edge speed since noise conversion to timing jitter is amplified with a slow edge. Typically there is one clock net qualifying many data and control nets, so the bulk of the above benefit(s) is/are achieved by applying slew rate limiting just to non-clock nets.

Q8 Conductor Delay Scaling

Why does the net delay become a larger proportion of the path delay as process geometries shrink?

Answer:

We have to consider net resistance and capacitance. Net inductance hardly matters since RC modelling of nets normally remains accurate enough. For FETs we mainly need to consider their drive strength and input capacitance. Simple formulae for each of these quantities should lead to an asymptotic analytical answer to the posed question.

A naive consideration pans out as follows: smaller geometries use smaller wires. If length and area are decreased in proportion, say by factors of α and α^2 respectively, resistance, given be $R = \rho l / A$, will go up by α. Smaller wires also have smaller capacitance. A plate capacitance model, $C = \epsilon_0 \epsilon_r A / d$, means a capacitor scales roughly linearly down as it is shrunk. This simple analysis implies the RC time constant is unchanged with a shrink, so Elmore delay along the net is unchanged.

If we further consider that vertical metal thickness is likely not shrunk as much as the planar features, do we start to see an effect? No. A reduced amount of scaling in the vertical dimension reduces the amount of reduction in both parasitic capacitive and resistance. But again, these only cancel out under simple modelling. Moreover, the total change of aspect ratio is only from about 1:1 to 2:1.

The current drive from a FET can be approximated with $\mu_n C_{ox} \frac{W}{L} [(V_{gs} - V_T) V_{ds} - V_{ds}^2/2]$. As a FET is scaled down, with constant length to width (aspect) ratio, W/L remains the same, so its transconductance is unchanged and it is able to deliver the same amount of current for a given gate voltage. Given that the driven capacitance is getting smaller, the gates themselves appear relatively faster. Hence more attention must be paid to the nets to speed them up to match. In addition, as input capacitance decreases, wiring capacitance becomes ever more dominant. This may be the main explanation, but other considerations now follow.

As nets have shrunk from the early days of VLSI, where their width was much greater than their

height, the dominant effect has moved away from plate capacitance the layers above and below. It moved, when height was equal to spacing, to being approximately parallel wire capacitors. Today, now height is greater than width, they resume plate capacitor behaviour, but with the predominant capacitance being to the next net on the same layer. For long busses, power or ground nets can be interleaved between signal nets to reduce crosstalk. These changes make a difference to cross talk modelling, but only make a factor of 2 or so difference in net capacitance models, so do not contribute to an asymptotic model.

The move to FINFETs gives a step change in transistor performance per unit area: the transistors are proportionally faster and the interconnect is accordingly more dominant. Again, this is a one-off change and does not contribute to an asymptotic effect, but it is perhaps a significant reason for why interconnect needs to be considered more seriously today.

A final consideration is that SoCs today contain more and more subsystems and die sizes have not shrunk as feature size has decreased. If anything, dice are getting bigger! So, although local interconnect may be part of the story, long distance interconnect is perhaps worthy of the most attention. This is especially so if round-trip time is important to system throughput, as discussed in Chapter 3.

Q9 Statistical Delay Modelling

Why would it be helpful to model the statistical variation of net delays instead of assuming all interconnect segments are at one BEOL corner?

Answer:

From 8.12.14: A front end-of-line (FEOL) variation is a change in active component parameters, such as the gain, V_T or on-resistance of a FET. A back end-of-line (BEOL) variation is a change in unwanted parasitic parameters, such as net resistance and stray capacitance.

FEOL variations generally vary from wafer to wafer or slowly across a wafer. This means that the variation in parameters is likely to be correlated. For instance, if every gate is a bit slower than nominal, the effects will accumulate systematically and applying a flat OCV factor is appropriate.

On the other hand, BEOL variations mainly effect nets and have much greater randomness. Some nets can have higher resistance or capacitance and others lower, leading to a mixture of faster and slower components along any one path. These variations do not accumulate systematically and, instead φ a n tend to cancel out. A random walk of n steps of size a has expected displacement $0.8 \cdot a \cdot (\sqrt{n})$, a lot less than $a \cdot n$.

A sales and marketing specification of the required percentage of dies that should meet a given target frequency dictates the maximum area allowable in the tail of the distribution of critical path(s). EDA tools can then aim for that metric.

Q10 Static Timing Analysis

-o- Create a list of the sources of timing uncertainty considered during STA. Are there any that were not

discussed?

Answer:

Table 8.8 '*Common sources of timing error*' lists four sources of timing uncertainty: clock jitter, hold constraint characterisation error, dynamic voltage noise and net variation.

There are quite a few other detailed sources of uncertainty, such as FET threshold or gain variations. These lead to both set-up and hold time characterisation variations, but only the latter was listed in the table. Why is this? It is emphasised elsewhere in these exercises that set-up time aspects are less critical than those for hold-times since they can stop a die working at any clock frequency. Also, most systematic process variations are accommodated in the multi-corner analysis.

The presented multi-corner analysis tracks systematic process variations well but did not list variation in variances. Stochastic variations in FET thresholds or gain will lead to timing variations.

-o- On-chip Parameter Variation: Give an example of OCV that is dependent on location and one that is dependent on time. Is there an example that depends on both location and time?

Answer:

Component-to-component parameter variation has random and systematic components. A location-based systematic change in a parameter (e.g., resistance of net or FET) from one corner of a chip to one of the others is called local OCV (on-chip variation). These changes arise from the manufacturing processes used. A processing step that spins the wafer will have a centre of rotation with zero angular velocity, whereas the edges of the wafer will be moving the fastest. The thickness of a liquid layer deposited while rotating will depend on the surface tension of the liquid and the centripetal force. The latter varies across each wafer and across larger dice, leading to local OCV. Equally, an electron or ion beams used in certain steps will have a greater angle of deflection at different points and ultra-violet or X-ray sources will also have less than ideal, uniform power density. Where the wafer is stepped for each chip, the variation is re- peated across each die in a similar way, but if the whole wafer is processed at once, the OCV will be more global. Similar effects arise between the first and last wafer of a batch, owing to equipment warming up and so on. Hence time of manufacture makes a difference.

The age of a chip, in terms of how long it's been powered up for, or the cumulative integral of the supply voltage or I/O pad current over time leads to time-dependent effects, especially electromigration.

OCV can be thought of as a parameterized generator of the first third of PVT (process, voltage and temperature) variation. PVT is often thought of stateless, in that the same die running at the same voltage and temperature will offer the same performance as the last time it was run at that voltage and temperature, but the age-based effects do change the chip.

The threshold voltage, transconductance and on resistance of FETs are very important parameters. All three suffer both spatial and temporal OCV. The spatial aspect arises from the man-

ufacturing unevenness mentioned above. The temporal effects arise from electromigration of the metal contacts, ageing of the Schottky metal to semiconductor contacts and charge accumulation in the substrate near the active FET channels.

-o- Negative Slack Amelioration 1: What kind of optimisations might be done to fix STA minimum timing violations with negative slack?

Answer:

The relevant illustration is the right-hand-side of Figure 8.59 '*Troublesome early and late path configurations for maximum (left) and minimum (right) timing*'. A hold time violation arises when the input to a flip-flop is not held long enough after the clock edge. The earliest time at which it might change is established with a minimum timing analysis.

Sometimes, it is only a few data inputs to a broadside register that are slightly suffering. One clear fix is then to artificially extend their net delay using circuitous routing. (Note, although the low output bits from various forms of adder often have lower combinational delay than the higher bits, owing to carry propagation, this observation is unhelpful when considering minimum timing: all bits may be equally likely to suffer from short paths and carries typically only affect the maximum timing!) If the problem arises from pass-transistor multiplexors, using an active buffer will fix the problem since it extends the path delay. An inverter is simpler than a buffer and may also solve the problem, but then the register output must be taken from the $\overline{Q}$ output, which is not normally a problem. Flip-flops with shorter hold time may be available for use, so switching to these is an option. Repipelining is another option, discussed below.

As emphasised in the book, devices that experience hold-time violations cannot work reliably at any clock frequency, so such devices cannot be speed binned. They can perhaps be binned for a narrower temperature range (e.g., commercial 0 to 70 degrees instead of industrial

-o- Negative Slack Amelioration 2: What kind of optimisation might be done to fix timing violations with negative slack in maximum timing analysis?

Answer:

The relevant illustration is now the left-hand-side of Figure 8.59. Negative slack under maximum timing analysis is the principal concern for achieving high clock frequency and is the 'classical' critical-path problem. The baseline approaches to negative slack are repipelining, reducing the clock frequency, routing the critical nets more directly, making the logic faster or skewing the clocks. Please see the points under *Exploiting Positive Slack Time* below.

-o- Exploiting Positive Slack Time: What kind of optimisations might be done to lower the power by reclaiming positive slack in a maximum timing analysis?

Answer:

Positive slack time is the period between when data will arrive at latest at the input of a flip-flop and the start of that flip-flop's set-up time. It is unnecessary.

As shown in Figure 4.43 '*Clock skewing*', positive slack can be reduced using clock skewing to help meet timing closure elsewhere, but this will not directly reduce power. Instead, the logic and nets that generate this signal can be changed in two principal ways to save power:

1. A slower or lower-power driver for the net will typically consume less static power. Such drivers can be made with smaller or differently doped transistors. Note the dynamic energy use will be unchanged if the net retains the same capacitance. Routing the net more directly (making it short) would reduce the dynamic energy, but would also increase the positive slack, which is not what the question is asking.
2. Alternatively, a state re-encoding can be used. At this stage of design development, a big change in state encoding or bus semantics is undesirable. But simple D-type migration (§4.4.2 '*Folding, Re-timing and Recoding*') can sometimes be locally applied, performing some of the work scheduled for the next clock cycle in the current clock cycle, thereby implementing a timing/layout-based repipeline operation.

-o- Hold Time Violations: Why can minimum timing violations not be fixed by decreasing the clock frequency?

Answer:

Reducing the clock frequency makes the next clock edge arrive later, effectively increasing the set-up time available or allowing an increase in the maximum timing limit. But minimum timing violations relate to a single clock edge and the spacing from the previous or the next (i.e. the clock period) does not matter. As stated just above, a hold time violation arises when the input to a flip-flop is not held long enough after the clock edge. Consider a shift-register arrangement: there is no logic between the Q output of one flop and the D input of the next. If the hold time specification is longer than the clock-to-Q time and there is negligible net delay, the shift register will not work at any frequency.

The above answers apply to single-cycle paths in a clock domain. Both set-up and hold times are affected by clock frequency variation with a multi-cycle path. See §4.9.6 '*Clock Skewing and Multi-cycle Paths*'.

-o- Why is it important that inputs to STA, like Liberty abstract timing models and SPEF netlists, conform to an IEEE standard?

Answer:

Standardization of timing details is important for contractual sign-off. The logic design and the cell library that it is implemented in are typically sourced from different companies. If there is a mismatch between interpretations of specifications, a chip may not meet timing when finally made. Using a well-established, industry proven methodology, as standardized by the IEEE or other august body, reduces the scope for ambiguity and makes it easier to establish fault in a dispute. See §8.12 '*Static Timing Analysis and Timing Sign-off*'.

Q11 Yield Improvement Through Redundancy

Describe how the die yield can be improved if a structure is replicated hundreds of times over a chip? Should the end user be involved in this process? Consult a recent DRAM chip data sheet and discuss the mechanisms likely to be used during a production test and at boot time.

Answer:

Production yield is the percentage of working die on a wafer or batch of wafers processed at once.

Devices can be provided with completely redundant additional subsystems exist to benefit yield. At production test time a strapping or zapping technique is used to swap out a failed subsystem for a working spare.

For instance, an FPGA with nominally 128 columns of programmable cells might be manufactured with an additional two, making 130 in total. If production test identifies one or two of the first 128 has a manufacturing fault, a renumbering system can be invoked that causes a faulty row to be skipped over as far as the user can see. Renumbering is needed in FPGAs, (unlike in, say, a RAM with additional columns) because there is a tolerable increase in inter-row wiring delay arising from skipping over a row, compared with routing signals to a spare column at the edge of the array and back again. That would be prohibitive both in terms of delay and additional wiring needs.

As mentioned in the book, there are various production strapping/zapping techniques available to deploy the redundant units. These include melting metal links with a high current, depositing or cutting metal links with ion beams and storing charge on floating gate transistors. The last is like a 1980's EPROMs, but fully covered in metal so that ultraviolet erase is not possible (other forms of radiation, such as electron beams, may still work though!)

Another example is an 8-core processor that is marketed as a 7-core processor since one core has failed wafer test. A zap disables all access to the failed core from the others and modifies the value returned in a read-only device identifier register so that the O/S can be aware and adapt.