ECEN454 refsheet

https://github.com/joshua-wright

specifically for final

- exam 1 was on March 8 (2017.03.08)
- nothing specifically from previous exam

Propagation vs Contamination delay

- contamination delay (t_{cd}): time from input change to **any** output changing value
- kind of a best-case delay or smallest possible delay
- time counted when input crosses 50% of logical high voltage level
- propagation delay: time from when all inputs are stable to when all outputs are stable
- kind of like a worst-case delay

static vs dynamic power

- static power
  - consumed when circuit is not active
  - e.g. leakage current
  - static because no inputs are changing
- dynamic power
  - power consumed due to gates switching and stuff
  - e.g. charging/discharging parasitic caps, etc…

Latch vs Flip Flop

- Latch is level-sensitive, flip-flop is edge-sensitive
- register is flip-flop # Latch/flip flop designs
- pass transistor latch
  - pros: tiny, low clock load
  - con: voltage drop across is V_t, it's non-restoring
  - con: not really a latch because doesn't hold signal
- transmission gate
  - pro: no V_t drop
  - con: requires inverted clock
- inverting buffer
  - TODO
- tristate feedback
  - first complete latch here
  - risk of backdriving: downstream could change state if it is very strong
- buffered output
  - no backdrifing problem

- widely used
    - rather large and slow though, and large load on clock
- a flip-flop is just two latches back-to-back

Metastability

- stable works if it's not at a strong high or low but will settle somewhere
- metastable is where it might sit there, but if it must settle, you don't know which one it will settle to when perturbed

2017.03.22

Interconnect

- contacts
    - have resistance $2 - 20\Omega$
    - so use a bunch of them to reduce resistance between layers
- lumped RC model
    - when you lump together RC sections for speed
    - π-model (pi-model) is most commonly used
- distributed RC model: opposite of lumped model
- crosstalk
    - caused by parasitic cap between wires
    - one wire changing voltage can induce noise on adjacent wires and/or increase delay

B	ΔV	$C_{eff}(A)$	MCF
Constant	V_{DD}	$C_{gnd} + C_{adj}$	1
With A	0	C_{gnd}	0
Opposite A	$2V_{DD}$	$C_{gnd} + 2C_{adj}$	2

- **MCF: Miller Effect**: effect of crosstalk
    - quantifies crosstalk delay
    - use MCF factor to determine effective parasitic capacitance between two signal wires
- Elmore delay: split into segments and use pi-model (or some other model)
    - wire delay is quadratically proportional to length of wire
- repeaters:
    - worth it only if wire is sufficiently long
    - $t_{unbuf} = R(cx + C) + rx(cx/2 + C) =$
    - $t_{buf} = 2R(cx/2 + C) + rx(cx/4 + C) + t_b$
        - x: wire length
        - R: buffer output resistance
        - C: buffer input capacitance
        - r: wire unit length resistance
        - c: wire unit length capacitance
        - t_b: buffer delay time
    - time difference: $\Delta t = t_{buf} - t_{unbuf} = RC + t_b - rcx^2/4$

2017.03.27

Sequential Circuits

- sequential means that the circuit holds state: the output depends on both current and past input
- to model a state machine, you can unroll it to [register] → [combinational] → . . .
  - this way you can use regular timing stuff for it: arrival time, slack, etc. . .
- Mealy FSM: output of circuit is from combinational logic
- Moore FSM: output is from registers
- pipelined circuit: uses registers to hold state between clock cycles, because not all of the combinational logic can happen fast enough to work in a single clock cycle
  - pipelined circuits can use flip-flops or latches?
- the clock consumes 20-30% of the power on a chip
- the whole reason that registers are needed is because data (signals) moves through components at non-constant speed
- reset
  - can be sync or async
  - force low output when reset is high
- sequencing: it is (generally) equivalent to split the sequential logic in two, and then use two latches (one halfway through) instead of one large section of sequential logic with flip-flops at either end
  - this is called two phase clocking
  - you have to make sure the middle latch operates on clock-bar instead of clock

symbol	definition
t_{pd}	logic propagation delay
t_{cd}	logic contamination delay
t_{pcq}	Clk→Q propagation delay
t_{ccq}	Clk→Q contamination delay
t_{pdq}	D→Q propagation delay
t_{setup}	setup time of flip-flop/latch
t_{hold}	hold time of flip-flop/latch

- timing:
  - D→Q delay only makes sense for latches, since for flip-flops it is simply one clock cycle
- sequencing overhead and max delay:
  - T_c: cycle time
  - need for combinational logic to be fast enough
  - for single phase flip flop: $t_{pd} < T_c - (T_{setup} + T_{pcq})$
  - for two-phase latch: $t_{pd} = t_{pd1} + t_{pd2} < T_c - 2t_{pdq}$
  - max delay is dictated by cycle time and sequencing overhead; sequencing overhead does not effect minimum delay
- minimum delay:
  - for single phase flip flop: $t_{cd} > t_{hold} - t_{ccq}$
    - combinational stuff must be slow enough that it doesn't violate hold time of second flip flop
  - for two-phase latch: $t_{cd1}, t_{cd2} > t_{hold} - t_{ccq} - t_{non-overlap}$
    - hold time applies twice each cycle, once per each combinational circuit
    - $t_{non-overlap}$: phase between clock signals?
- time borrowing: for two-phase latch-based system, you can borrow time from one segment to the other while the latch is transparent
  - works (is possible) as long as the total circuit still completes within one clock cycle
- clock skew:
  - when uncertain: causes tighter required CL propagation delay and minimum CL contamination delay, and reduces opportunity for time borrowing.
  - when know constant skew: can help with same effect as time borrowing, except can also have effect for flip-flop based systems

- skew tolerance:
  - flip-flop circuits are not very skew tolerant because the setup and hold times define a specific window of time in which the data must arrive
  - latch circuits are more hold-time-tolerant because the data can arrive any time that the latch is transparent
- if setup times are violated, you can reduce clock speed to fix it
- if hold times are violated, it won't work at any clock speed

2017.03.29

Clocking

- H-Tree (clock distribution network)
  - laid out in such a way that the delay from the center to any leaf node is equal
- Grid clock distribution
  - grid on two or more levels
  - ensures low local skew, but there could still be larger chip-wide skew
- generally, the longer the delay between two point, the larger the worst-case clock skew
- Domino circuit: TODO
  - a kind of dynamic logic
  - when the clock pulse cascades through the circuit, activating the pMOS at each stage?

2017.04.03 (+more)

Memory

- random access or sequential access, or content addressable
- CAM: content-addressable memory
  - basically like an associative array, or database
  - designed to search all memory at once in a single operation (wikipedia)
  - used in caches
- SRAM:
  - 12T and 6T cell designs
  - 12T design: just a latch connected to a bitline
    - boring and impractical because it's too large
  - 6T design: cross-coupled inverters (3T each) +2T for bit and bit-bar lines
    - we focus on 6T design
  - read: pre-charge bit and bit-bar, raise wordline, read result
    - (need to pre-charge or else read is destructive?)
    - need sense amplifiers at the end of the bitlines because the signal will be weak, and the parasitic capacitance of the bitline depends on how many cells are 0 or 1
  - write: drive data onto bit and bit-bar and raise wordline to write
  - cross coupled inverters must be much larger than the bitline transistors so that it doesn't flip while reading
  - per each column, you need:
    - piece of decoder
    - bitline sense amplifier
    - column multiplexer
  - multi-port: means you can read from and write to the same cell in the same clock cycle and it works
    - implement by having multiple sequential read/write per cycle?
- decoders:
  - must be pitch-matched to SRAM cell width for layout efficiency/simplicity (requires very skinny gates)
  - large ($n > 4$) decoders are inefficient, thus it is often helpful to use predecoding: factor out common gates into a second stage of decoders.
    - smaller area as alternative, but same logical effort (since same number of gates traversed)

- twisted bitlines: bitlines are long and right next to each other, so there will be significant crosstalk and parasitic cap
  - solution: swap adjacent bit and bit-bar lines occasionally
  - reduces Miller factor (MCF)
- DRAM:
  - one transistor and one capacitor
    - transistor gates capacitor
  - no charge is 0, charge is 1
  - open transistor to read/write
  - read is destructive
  - must be refreshed periodically due to leakage
    - thus, it's always consuming power
- shift register:
  - can be implemented with cascade of flip-flops
    - can lead to hold time violations because no delay from clock to next data line? (TODO what? how?)
    - not very dense
  - store data in SRAM and store begin/end pointers
    - more dense than individual flip-flops
- Tapped Delay Line
  - shift register with programmable number of stages
  - implement using sequence of flip-flops, each with a bypass MUX from data in to out
- Queues: are a thing (TODO)
- ROM: Read Only Memory:
  - mask-programmed ROMs are one transistor per bit
- PROMs, EPROMS, etc
  - generally burn out specific fuses to program ROM
  - E for erasable

2017.04.10, 2017.04.12

Low Power Design

- not so important when running from a battery since batteries store energy, not power
- peak power
  - determines power/ground wiring and packaging limits
  - impacts signal/noise ratio
- reduce dynamic power: (AKA active power)
  - use smaller transistors
  - clock gating
  - reduce V_{DD}
  - reduce frequency
    - but be careful because that will make the computations take longer
- reduce static power
  - power gating
  - selectively use lower V_t devices
  - stacked devices
  - body bias
- clock gating: disable the clock signal to an expensive component
  - eliminates that component's dynamic power (but of course you can't use it)
  - doesn't change static power
- voltage islands: different V_t for different devices
  - allow low V_t devices to run slower because they don't need to be fast (maybe they're not in the critical path)
  - often different voltage levels are alternated row by row in standard-cell layouts
  - you must convert from one logic level to another, because high/low will be different

    - * sometimes it's possible to avoid conversion, e.g. by using a buffer that you would have anyway, with inverters built from half high-V_t transistors and half low-V_t transistors
  - there are all kinds of tricks for determining which components should be high/low voltage and where to put voltage islands and level converters
- DFVS: Dynamic Frequency and Voltage Scaling
  - requires:
    - programmable clock generator (PLL)
    - supply regulation loop to for setting minimum V_{DD}
    - software support for deciding best stable frequency and voltage for given load and performance goals
  - voltage regulation loop
    - voltage divider would be horribly inefficient
    - instead, rapidly charge/discharge capacitor
- stacked devices:
  - stacking devices reduces leakage
  - because leakage is exponentially proportional to V_{ds}
- body bias AKA dual V_t
  - changing V_t at runtime is called Adaptive Body-Biasing (ABB) or Dynamic Threshold Scaling (DTS)
    - requires triple well fabrication process, because what would have been lowest level substrate must itself sit inside a well that will be biased
  - design with low and high V_t
  - low V_t is faster than high V_t but consumes 10x more power
    - because reducing V_t increases sub-threshold leakage exponentially
    - but reducing V_t decreases gate delay (increasing performance)
  - to increase V_t, put a negative bias on V_{SB} of nMOS
    - use DC charge pump to get desired voltage
- power gating: use sleep transistors
  - basically cut off V_{DD} from power-hungry circuits when not in use
  - in sleep mode, the sleeping transistors are stacked with the transistors that are causing them to sleep, thus reducing leakage even more
- MTCMOS sleep transistors
  - combines power gating and dual V_t
  - TODO what is this?

2017.04.17

Packaging

- provides:
  - heat dispersion
  - mechanical stability
  - prevent thermal expansion
  - IO
- SMD: Surface Mount Device
- SMT: Surface Mount Technology
- package types
  - DIP: Dual In-line Package
  - QFP: Quad Flat Package
  - PLCC: Plastic leaded chip Carrier
  - BGA: Ball Grid Array
  - PGA: Pin Grid Array
  - MCM: Multi Chip Module
- DIP: Dual In-line Package
  - basic chip with two sides and pins pointing either down (through hole) or out to the sides (surface mount)
- QFP: Quad Flat Package
  - looks about the same as DIP with SMD pins but has pins pointing out of all 4 sides

- PLCC: Plastic leaded chip Carrier
  - like QFP but instead of pins has tiny pads (J-leads) on perimeter of chip.
  - can sit in a socket with spring connectors that contact those pads
- BGA: Ball Grid Array
  - can have lots of pins
  - can attach to substrate with flip chip or wires
  - underside has pads for where solder balls go to connect to board
- PGA: Pin Grid Array
  - like BGA but with pins
- MCM: Multi Chip Module
  - when you put multiple dies on a single chip
  - advantage over very large single die: you can then select just the working ones of each die component
  - disadvantage: means you'll probably want to test chips before you package them
- flip-chip
  - also called C4: Controlled Collapse Chip Connection
  - just flip the chip over, thus placing the connecting pads on the highest metal layer
  - affix chip to package using a BGA-style connection
  - advantages
    - higher pin density
    - can put pins anywhere without worrying about routing
- package parasitics
  - parasitics inductance isn't really a problem inside the chip, but is for connecting wires in packaging
- heat dissipation
  - formula: $\delta T = \theta_{ja} P$
    - δT: change in temperature on chip
    - θ_{ja}: thermal resistance of chip junction to ambient
    - P: power dissipation on chip
    - θ adds in series for chip→package and package→headsink
  - must cool chip to avoid things like thermal resistance and electro-migration
  - thermal resistance can cause the chip to fail, but that's only temporary
    - should work again after cooling
  - electro-migration: electrons moving where they shouldn't inside the chip
    - permanent build-up of problem
- power distribution
  - want to minimize power noise
  - voltage will drop due to
    - IR drop
    - $L * (di/dt)$ noise (from parasitics inductance)
  - decoupling capacitors
    - aka bypass caps
    - cap between V_{DD} and ground to smooth out voltage supply
    - use multiple caps in series to avoid problems with the resonant frequency of any single cap
- I/O
  - must protect against ESD
  - pad types:
    - V_{DD}/GND
    - output
    - input
    - bidirectional
    - analog
  - latch-up
    - when noise or density or something causes transistors to do weird things or something
    - TODO
  - output
    - need to drive relatively large off-chip loads, so usually require chain of successively larger buffers
    - protect against latch-up using guard ring: p+ inner ring and n+ outer ring (in n-well)
  - input

        - must do level conversion
        - must filter noise
            - use Schmitt trigger: basically doesn't cross threshold unless extra oomph provided (it's sticky)
    - bidirectional
        - combined input/output pad
        - chip has enable signal to decide direction
    - analog
        - must avoid distorting or delaying the signal at all (no buffering)
        - ESD protection circuit must not distort the signal
- ESD protection: TODO
    - ESD: Electro-Static Discharge
    - can damage circuits with too much current and voltage
    - limit current using resistor in series
    - limit voltage using two diodes:
        - one from ground to signal line (pointing toward signal line),
        - one from signal line to V_{DD} (pointing toward V_{DD})
- MOSIS IO pads
    - library of IO pad designs
    - provides IO protection and stuff

2017.04.26

Testing

- logic verification is very important
    - consumes 50% of total IC development time
- 3 types:
    - LV: Logic Verification
    - SD: Silicon Debug
    - MT: Manufacturing Test
- LV: Logic Verification
    - simulate chip and see if implementation is correct (does what expected)
    - when you can't test every possible combination (because too many bits and/or too much state), test edge cases
- SD: Silicon Debug
    - test fabricated chips
    - check for logic failures and electrical failures
    - logic: bad implementation
    - electrical: crosstalk, leakage, charge sharing, etc...
    - fixing bugs requires redesigning and re-fabricating the chip
- MT: Manufacturing Test
    - check that the chip was fabricated without error
        - e.g. no speck of dust came in and killed the chip
    - yield is always <100% because fabrication is never perfect
    - testing machines are super expensive so you gotta balance number of test cases (test coverage) vs cost
- Smoo Plot
    - used to diagnose chip failures
    - 2D plot of frequency vs V_{DD}
    - X or not depending if chip passed test at that frequency and supply voltage
    - can help to tell what is causing failures
- stuck-at fault
    - model a failure as stuck at 0 or 1
        - there's a whole bunch of ways that silicon could be manufactured incorrectly, but stuck-at fault models most of these approximately right
- observability: how easy it it is to observe a node at the output

    - to observe a node at the output, the output must be determined by that node
- controllability: how easy it is to force a node to a specific value
- combinational logic has good observability and controllability, finite state machines (FSMs) do not
- test pattern generation:
    - apply smallest possible sequence of test vectors necessary to show that each node is not stuck
    - circuits with better controllability and observability are more testable (can be fully tested with fewer test patterns)
- scan register, scan chain
    - add a mux to the input of all registers, linking them together in a big optional shift register
    - then you can shift pre-defined values into all registers for testing, and shift out the test results
    - makes the circuit more testable
- ATPG: Automatic Test Pattern Generation
    - given blocks of combinational logic, generates test patterns for use with scan chains
- built in self test
    - circuits that test themselves using built-in test pattern generators and verifiers
- PRSG: Pseudo-Random Sequence Generator
    - just a linear feedback shift register with input as XOR of state
    - used for random testing
- BILBO: Built-in Logic Block Observer
    - automatic tester
- boundary scan
    - basically a scan chain for all the pins (boundaries)
    - because some packaging types (e.g. BGA) are hard to make temporary sockets for