EECS151/251A Discussion 13

Christopher Yarp

May. 3, 2019

Plan for Today

- Hamming Code Practice
- Faults
- Questions

Hamming Codes

- Using the 7 bit Hamming code in lecture (4 data, 3 parity), encode 0010
- Recall, in Hamming Codes, we start counting bit positions from 1, **not 0**
- Parity bits are placed at bit positions that are powers of 2 (1, 2, 4, 8, …)
- Bits in the group for parity bit 2^n contain all bit positions with a 1 in the nth most significant digit (counting from 1 as the LSb)
- Parity bits are set such that the group has even parity
  - If there is an even number of 1's in a group (not including the parity bit) the parity bit for that group is set to 0
  - If there is an odd number of 1's in a group (not including the parity bit) the parity bit for that group is set to 1
- The encoded word is: 0101010

Encoding Format:

1	**2**	**3**	**4**	**5**	**6**	**7**
001	010	011	100	101	110	111
P1	P2	D1	P3	D2	D3	D4

Parity Bit Groups:

	1	**2**	**3**	**4**	**5**	**6**	**7**
	001	010	011	100	101	110	111
	P1	P2	D1	P3	D2	D3	D4
P1	Y		Y		Y		Y
P2		Y	Y			Y	Y
P3				Y	Y	Y	Y

Parity Bit Groups:

1	**2**	**3**	**4**	**5**	**6**	**7**
001	010	011	100	101	110	111
0	1	0	1	0	1	0

Hamming Codes

- Decode 1010111 which was encoded with the 7 bit Hamming code in lecture (4 data, 3 parity).
- The parity is computed for each group
  - Group 1: C1 = P1+D1+D2+D4 = 1+1+1+1 = 0
    - No error detected in group 1
    - None of the bit's in positions ??1 have an error
    - P1, D1, D2, D4 are correct
  - Group 2: C2 = P2+D1+D3+D4 = 0+1+1+1 = 1
    - Error detected in group 1
    - One of the bits in positions ?1? has an error
    - P2, D1, D3, or D4 has an error
  - Groups 3: C3 = P3+D2+D3+D4 = 0+1+1+1 = 1
    - Error detected in group 3
    - ne of the bits in positions 1?? has an error
    - P3, D2, D3 or D4 has an error
- C3 C2 C1 provides the position of the error
  - The bit in position 110 has an error
  - The decoded value us 1101

	1 001	**2** 010	**3** 011	**4** 100	**5** 101	**6** 110	**7** 111
	P1	P2	D1	P3	D2	D3	D4
P1	Y		Y		Y		Y
P2		Y	Y			Y	Y
P3				Y	Y	Y	Y

Received:

1 001	**2** 010	**3** 011	**4** 100	**5** 101	**6** 110	**7** 111
1	0	1	0	1	1	1

Corrected:

1 001	**2** 010	**3** 011	**4** 100	**5** 101	**6** 110	**7** 111
1	0	1	0	1	0	1

Hamming Codes

- Implement a single error correcting Hamming code with 15 bits
- If p is the number of parity bits and d is the number of data bits, the following relationship must be held: $2^p \geq p + d + 1$ or $\mathrm{p} \geq \log_2(p + d + 1)$
- You were given that p+d = 15
- $\mathrm{p} \geq \log_2(16), \mathrm{p} \geq 4$
- Therefore, the 15 bit Hamming code contains 4 parity bits and 11 data bits
- The rate of this code is 11/15 = 0.733
- Compare this to the 7 bit code which had a rate of 4/7 = 0.571
- Parity bits go into positions that are powers of 2

	1 0001	**2** 0010	**3** 0011	**4** 0100	**5** 101	**6** 0110	**7** 0111	**8** 1000	**9** 1001	**10** 1010	**11** 1011	**12** 1100	**13** 1101	**14** 1110	**15** 1111
	P1	P2	D1	P3	D2	D3	D4	P4	D5	D6	D7	D8	D9	D10	D11
P1	Y		Y		Y		Y		Y		Y		Y		Y
P2		Y	Y			Y	Y			Y	Y			Y	Y
P3				Y	Y	Y	Y					Y	Y	Y	Y
P4								Y	Y	Y	Y	Y	Y	Y	Y

Faults

Ideally, every device would behave exactly as we had envisioned it. However, the real world has other plans!

Design Bugs (ex. logic bugs, timing closure bugs, power draw)

- Timing bugs can be identified through timing analysis.
  - Vivado and icc/Primetime do this for you and provide a timing report.
  - The design is modified and the timing analysis is re-run until no timing errors exist
- Power use can be estimated through power analysis
  - Basis analysis assumes some average transistor activity
  - More accurate analysis can be attained by capturing simulations of real workloads to derive realistic transistor activity factors
  - Vivado and icc/PrimetimePx can do both of these for you
- Logic bugs can be caught during the design phase through testing and formal methods
  - Verilog testbenches which can be run in vcs, ModelSim, and Vivado Simulator
- Checking that tools do not introduce bugs in your design
  - Formal verification of the equivalence of 2 designs and simulation of theoretically equivalent systems are some common strategies
  - Formality is a formal verification tool by Synopsys

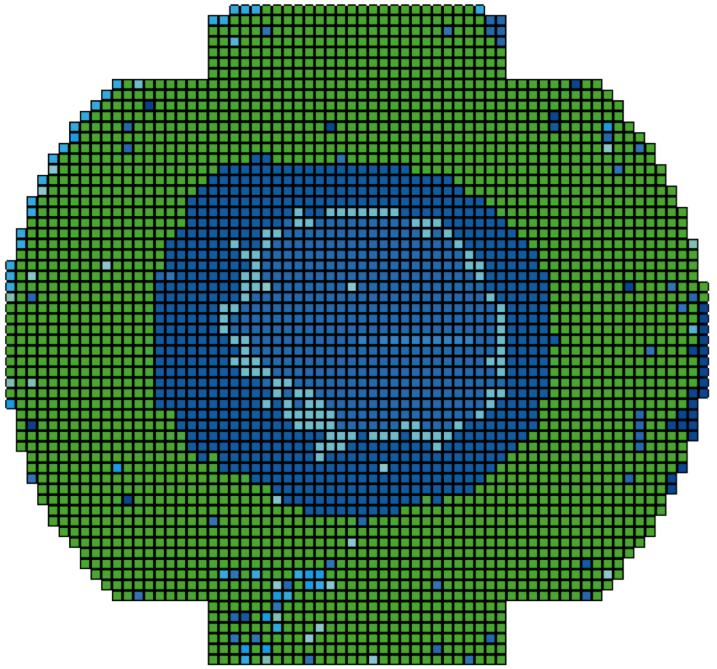
Manufacturing Defects

- A lot can go wrong when producing an IC
  - Wires can fuse
  - Impurities can contaminate the product
  - Process variations across runs, across shots of a mask, …
- Attempts are made to limit these effects at the fab (factory)
  - Limiting contamination by manufacturing in a Cleanroom with employees wearing special cleanroom attire
  - Source heavily controlled materials and perform extensive incoming quality control
  - Maintaining machines involved in production to high standards
  - Modeling process variations and their effects
    - Some models provided to IC designers

Photo Credit: Berkeley Marvell Nanofabrication Laboratory

Manufacturing Defects

- Fault and performance variations still occur despite our best efforts
- Mitigated by testing dies for functionality and performance
  - Sparing: use of extra on chip circuitry to account for faults
    - Spare rows/columns in DRAM
    - Disabling broken cores in multicore processors
  - Binning: sorting dies based on performance
    - High performance dies sold at a premium



A Bin Wafer Map:
Green = Bin 1 (Good Devices)

Image:
Y. A. Lin, "Parametric wafer map visualization," in *IEEE Computer Graphics and Applications*, vol. 19, no. 4, pp. 14-17, July-Aug. 1999. doi: 10.1109/38.773959

Runtime Faults

- Permanent Faults
  - Caused by some irreversible physical damage
    - Ex. Electromigration
- Transient: temporary faults caused by the environment (ex. high energy particles)
  - Tend to be randomly distributed across die
- Intermittent: temporary faults that tend to occur repeatedly in the same place
    - Possibly caused by a particular component operating sub-optimally in some situations
    - Ex. unexpected delay change due to temperature change causing a timing failure
    - May become permanent

Runtime Faults – Do we need to worry about them?

"The critical limiters to long-term reliability in a high performance microprocessor are time-dependent dielectric breakdown (TDDB) and electromigration (EM) [4]–[6]. The TDDB is strongly dependent on voltage and moderately dependent on temperature…. On the other hand, EM is enormously dependent on temperature and strongly dependent on voltage…. The failure rate can vary many orders of magnitude over the typical range of operating conditions found in microprocessors for desktop and laptop systems." [1]

[1] S. Sundaram *et al.*, "Bristol Ridge: A 28-nm x86 Performance-Enhanced Microprocessor Through System Power Management," in *IEEE Journal of Solid-State Circuits*, vol. 52, no. 1, pp. 89-97, Jan. 2017. doi: 10.1109/JSSC.2016.2623637

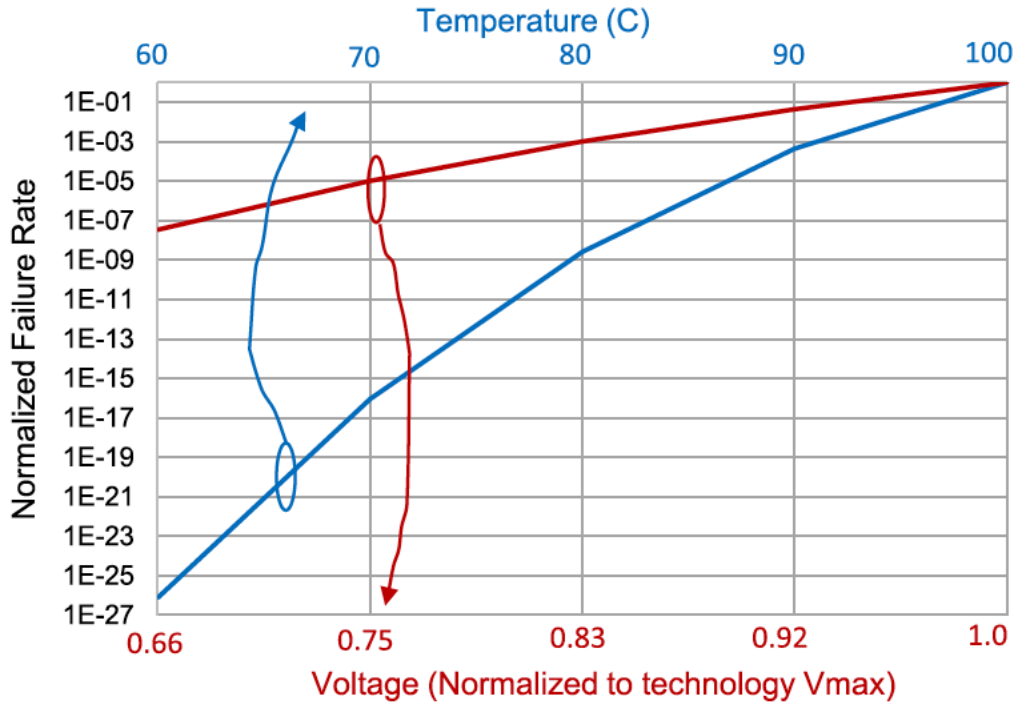
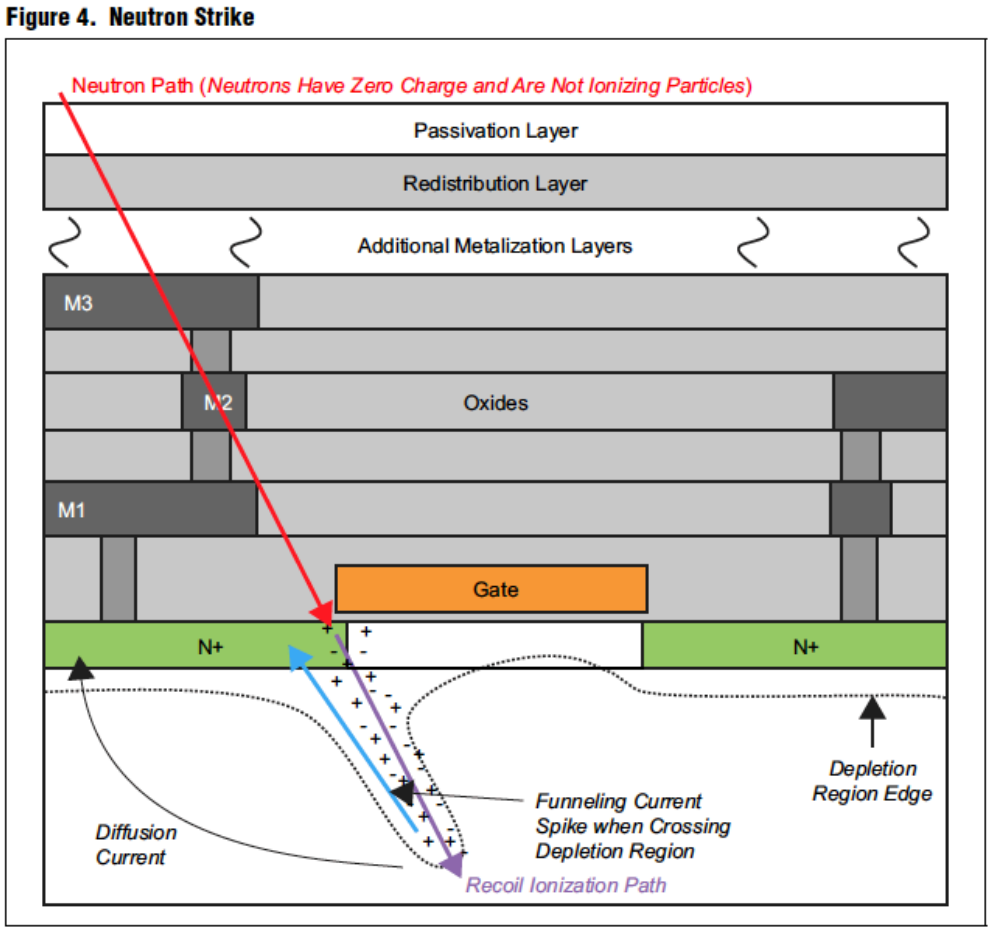


Fig. 2. EM failure rate versus temperature/voltage for copper interconnect (28 nm). [1]

Runtime Faults – Do we need to worry about them?

- Let's talk about transient effects.
- Many types of transient events are covered under the term Single-Event Effect (SEE).
  - One particular type is the Single Event Upset (SEU) where the state of a storage element is changed
- Caused by ionizing radiation [1]
  - Cosmic rays [2]
  - Alpha particles, particularly from packaging materials [2]
- See [3] for an interesting slide deck



Intel/Altera Rendering of a SEU [2]

[1] https://www.intel.com/content/www/us/en/programmable/products/fpga/features/stx-single-event-upset.html
[2] https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/wp/wp-01206-introduction-single-event-upsets.pdf
[3] https://indico.cern.ch/event/635099/contributions/2570672/attachments/1456364/2249943/Single_Event_Effecs_Radiation_Course_May_2017_SEE_CBP.pdf

Runtime Faults – Do we need to worry about them?

- Potential Negative Effects of SEU
  - Changing the bit configuration in an FPGA
    - Changes the programmed design
  - Changing a pointer address in a program
  - Changing a counter
  - … you get the idea …
- Mitigation Techniques:
  - FPGA vendors are aware of SEUs employ have several mitigation techniques in their products [2]
    - Include error correcting/checking and design methods that are SEU aware
  - ECC Memory is used in servers and workstations
  - Additional redundancy in safety critical and aerospace applications

- Why do our consumer computers not crash constantly due to SEUs?
  - They typically do not use ECC DRAM since it costs more
- How much data stored in memory is critical?
- There are typically a few important state elements and variables in a system.
- If a bit flip occurs in a …
  - Unused block of memory
  - An image
  - A text constant
  - …
- … it is unlikely that the SEU will crash the whole system.

[1] https://www.intel.com/content/www/us/en/programmable/products/fpga/features/stx-single-event-upset.html
[2] https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/wp/wp-01206-introduction-single-event-upsets.pdf
[3] https://indico.cern.ch/event/635099/contributions/2570672/attachments/1456364/2249943/Single_Event_Effecs_Radiation_Course_May_2017_SEE_CBP.pdf