CS 194-6
Digital Systems Project Laboratory

Lecture 7: DRAM

2008-10-27

John Lazzaro
(www.cs.berkeley.edu/~lazzaro)

TA: Greg Gibeling

www-inst.eecs.berkeley.edu/~cs194-6/
Today: What’s inside a DRAM chip?

- Silicon technology review
- DRAM core cells
- Memory arrays
- Error correcting codes (ECC DRAM) (if time permits)
Moore’s Law for CPUs and DRAMs

Transistors Per Die

10^10
10^9
10^8
10^7
10^6
10^5
10^4
10^3
10^2
10^1
10^0


1K 4K 16K 64K 256K 1M 4M 16M 64M 128M 256M 512M 1G 2G

1965 Data (Moore)
Memory
Microprocessor

Main driver: device scaling ...

Secondary driver: Wafer size

Wafer size conversions offset trend of increasing wafer processing cost

Source: Intel

Thus, cost per transistor plummets ...

Capacitance
Recall: Building a capacitor

- **Top Plate**: Conducts electricity well. (metal, doped polysilicon)
- **Bottom Plate**: Conducts electricity well. (metal, doped polysilicon)
- **Dielectric**: An insulator. Does not conduct electricity at all. (air, glass (silicon dioxide))
Recall: Capacitors in action

Because the dielectric is an insulator, and does not conduct.

After circuit “settles” ...

\[ Q = C \times V = C \times 1.5 \text{ Volts (D cell)} \]

- **Q**: Charge stored on capacitor
- **C**: The capacitance of the device: function of device shape and type of dielectric.

After battery is removed:

- Still, \( Q = C \times 1.5 \text{ Volts} \)
- Capacitor “remembers” charge
Capacitors and current ...

\[ Q = C \ V \]

Differentiate with respect to time ... if \( C \neq C(t) \) ...

\[ \frac{dQ}{dt} = C \frac{dV}{dt} \]

\( I \) is defined as \( \frac{dQ}{dt} \) ...

\[ I = C \frac{dV}{dt} \]

Observation: If a voltage change \( dV \) occurs in zero time \( (dt = 0) \), the current \( I \) is infinite (impossible).

The voltage across a capacitor cannot change instantaneously. And by \( Q = C \ V \), the charge stored on a capacitor cannot change instantaneously.
Storing computational state as charge

State is coded as the amount of energy stored by a device.

State is read by sensing the amount of energy

Problems: noise changes $Q$ (up or down), parasitics leak or source $Q$. Fortunately, $Q$ cannot change instantaneously, but that only gets us in the ballpark.
How do we fight noise and win?

Store more energy than we expect from the noise.

\[ Q = CV. \] To store more charge, use a bigger V or make a bigger C.

Cost: Power, chip size.

Represent state as charge in ways that are robust to noise.

Example: 1 bit per capacitor. Write 1.5 volts on C. To read C, measure V. V > 0.75 volts is a “1”. V < 0.75 volts is a “0”.

Cost: Could have stored many bits on that capacitor.

Correct small state errors that are introduced by noise.  

Ex: read C every 1 ms. Is \( V > 0.75 \) volts? Write back 1.5V (yes) or 0V (no).

Cost: Complexity.
MOS Transistors

Two diodes and a capacitor in an interesting arrangement. So, we begin with a diode review ...
Diodes in action ...

Resistor

Light emitting diode (LED)

Light on?

Yes!

Light on?

No!
Diode is off
\[ I = -I_0 \]

Diode is on
\[ I = I_0 \exp(V/V_0) \]

\[ I = I_0 \left[ \exp(V/V_0) - 1 \right] \]

Io range: 1fA to 1nA
Vo range: 25mV to 60mV
Making a diode on a silicon wafer

Periodic Table of the Elements

* Lanthanide Series
+ Actinide Series
A pure ("intrinsic") silicon crystal ...

Conducts electricity better than an insulator, worse than a conductor.

Why? Most electrons (dots) are in a full "valence" band. Moving in the band is difficult. Especially near 0 degrees K.

Lots of room, but few electrons.

Forbidden "band gap"

Many electrons, but packed too tight to move.
Intrinsic silicon crystal as T rises ...

Some valence band electrons diffuse into the conduction band.

These electrons leave behind “holes” in the valence band, allowing remaining electrons to move easier.

We think of “holes” as positive carriers ...

More electrons, better conduction
We “engineer” crystal with impurities ...

Periodic Table of the Elements

<table>
<thead>
<tr>
<th></th>
<th>I</th>
<th>II</th>
<th>III</th>
<th>IV</th>
<th>V</th>
<th>VI</th>
<th>VII</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>H</td>
<td>He</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Li</td>
<td>Na</td>
<td>Be</td>
<td>Mg</td>
<td>Al</td>
<td>Si</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>B</td>
<td>Ca</td>
<td>Sc</td>
<td>Ti</td>
<td>Cr</td>
<td>Mn</td>
<td>Fe</td>
</tr>
<tr>
<td>4</td>
<td>K</td>
<td>Sr</td>
<td>Y</td>
<td>Zr</td>
<td>Nb</td>
<td>Mo</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Rb</td>
<td>Cs</td>
<td>Ba</td>
<td>Th</td>
<td>Ac</td>
<td>Hf</td>
<td>Ta</td>
</tr>
<tr>
<td>6</td>
<td>Fr</td>
<td>Ra</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* Lanthanide Series

<table>
<thead>
<tr>
<th></th>
<th>Ce</th>
<th>Pr</th>
<th>Nd</th>
<th>Pm</th>
<th>Sm</th>
<th>Eu</th>
<th>Gd</th>
<th>Tb</th>
<th>Dy</th>
<th>Ho</th>
</tr>
</thead>
<tbody>
<tr>
<td>58</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

+ Actinide Series

<table>
<thead>
<tr>
<th></th>
<th>Th</th>
<th>Pa</th>
<th>U</th>
<th>Np</th>
<th>Pu</th>
<th>Am</th>
<th>Cm</th>
<th>Bk</th>
<th>Cf</th>
<th>Es</th>
</tr>
</thead>
<tbody>
<tr>
<td>90</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

...
N-type silicon: add donor atoms

Use diffusion or ion implantation to replace some of the Si atoms with As.

Arsenics has an extra electron that is "donates" to the conduction band.

\( n^+ \): heavy doping. \( n^- \): light doping.

Electrons from donor atoms. Improves conductivity.

No change in the number of holes.
**P-type silicon: add acceptor atoms**

Use diffusion or ion implantation to replace some of the Si atoms with Boron.

Boron has one fewer electron than Si. It can accept valence band electrons, creating holes.

\[ p^+ : \text{heavy doping.} \quad p^- : \text{light doping.} \]

No change in conduction band electron count.

Conduction band

Accept energy

Valence band

Number of holes increased, conductivity improves.
How to make a silicon diode ...

Cathode: -

Anode: +

At $V = 0$, "hill" too high for electrons to diffuse up.

For holes, going "downhill" is hard.

$V$ controls hill.
Diodes: Current vs Voltage

\[ I = I_o \left[ \exp\left(\frac{V}{V_o}\right) - 1 \right] \]

- Diode is off: \[ I = -I_o \]
- Diode is on: \[ I = I_o \exp\left(\frac{V}{V_o}\right) \]

- Io range: 1fA to 1nA
- Vo range: 25mV to 60 mV
Note: IC Diodes are biased “off”!

V1, V2 > 0V. Diodes “off”, only current is Io “leakage”.

\[ I = I_0 \left[ \exp\left(\frac{V}{V_0}\right) - 1 \right] \]

Anodes of all diodes on wafer connected to ground.
MOS Transistors

Two diodes and a capacitor in an interesting arrangement ...
What we want: the perfect switch.

Switch is off. V1 is not connected to V2.

Switch is on. V1 is connected to V2.

We want to turn a p-type region into an n-type region under voltage control.

We need electrons to fill valence holes and add conduction band electrons.
An n-channel MOS transistor (nFET)

Vd = 1V  
\( \downarrow I \approx nA \)

Vg = 0V  

Vs = 0V  

Polysilicon gate, dielectric, and substrate form a capacitor.

nFET is off (I is “leakage”)

Vd = 1V  
\( \downarrow I \approx \mu A \)

Vg = 1V  

Vs = 0V  

Vg = 1V, small region near the surface turns from p-type to n-type.

nFET is on.
Drawing an nFET

“Mask” drawings sent to the fabrication facility to make the chips.
Mask set for an n-Fet (circa 1986)

Mask set for an n-Fet (circa 1986)

\[ \begin{align*}
V_d &= 1V \\
V_g &= 0V \\
V_s &= 0V
\end{align*} \]

- \#1: n+ diffusion
- \#2: poly (gate)
- \#3: diff contact
- \#4: metal

Top-down view:

Layers to do:
- p-Fet not shown.
- Modern processes have 6 to 10 metal layers (or more)
  (in 1986: 2).
"Design rules" for masks, 1986 ...

Poly overhang. So that if masks are misaligned, we still get "---" in channel.

Minimum gate length. So that the source and drain depletion regions do not meet!

Metal rules: Contact separation from channel, one fixed contact size, overlap rules with metal, etc ...

#1: n+ diffusion
#2: poly (gate)
#3: diff contact
#4: metal

Poly overhang. So that the source and drain depletion regions do not meet!

Minimum gate length.

Metal rules:
- Contact separation from channel, one fixed contact size, overlap rules with metal, etc ...

#1: n+ diffusion
#2: poly (gate)
#3: diff contact
#4: metal
Fabrication
Mask set for an n-Fet ...

Top-down view:

- $V_d = 1V$
- $V_g = 1V$
- $V_s = 0V$

$I = \mu A$

Masks:
- #1: n+ diffusion
- #2: poly (gate)
- #3: diff contact
- #4: metal

How does a fab use a mask set to make an IC?
Start with an un-doped wafer ...

UV hardens exposed resist. A wafer wash leaves only hard resist.

Steps

#1: dope wafer p-
#2: grow gate oxide
#3: grow undoped polysilicon
#4: spin on photoresist
#5: place positive poly mask and expose with UV.
Wet etch to remove unmasked ...

HF acid etches through poly and oxide, but not hardened resist.

After etch and resist removal
Use diffusion mask to implant n-type accelerated donor atoms

Notice how donor atoms are blocked by gate and do not enter channel.

Thus, the channel is “self-aligned”, precise mask alignment is not needed!
Metallization completes device

Grow a thick oxide on top of the wafer.

Mask and etch to make contact holes.

Put a layer of metal on chip. Be sure to fill in the holes!
Final product ...

Top-down view:

"The planar process"

Jean Hoerni,
Fairchild Semiconductor
1958
p-channel Transistors
p-Fet: Change polarity of everything

$V_{\text{well}} = V_s = 1V \quad V_g = 0V \quad V_d = 0V$

New “n-well” mask

“Mobility” of holes is slower than electrons.
p-Fets drive less current than n-Fets, all else being equal.
Device Equations
Recall: Our old “switch” model ...

We begin by modeling transistors that are “off”

A “on” p-FET fills up the capacitor with charge.

A “on” n-FET empties the bucket.
Recall: Why diode current is $I = \exp(V)$ ...

Wafer cross-section

Wafer doped p-type

deployment region

cathode: -

anode: +

p-region

depletion region

At $V = 0$, "hill" too high for electrons to diffuse up.

n+ region

For holes, going "downhill" is hard.

no carriers

$V$ controls hill.
A simple model for “off” transistor ...

\[ V_d = 1V \]
\[ V_g = 0.2V \]
\[ V_s = V_{\text{sub}} = 0V \]

\[ I_{\text{ds}} = I_0 \left[ \exp \left( \frac{\kappa V_g - V_s}{V_o} \right) \right] \left[ 1 - \exp \left( -\frac{V_{ds}}{V_o} \right) \right] \]

\( \kappa = 0.7 \)

\( I_0 \sim 100fA, \ V_o = kT/q = 25mV, \kappa = 0.7 \)

Current flows when electrons diffuse to the “gate wall” top

# electrons that reach top goes up as wall comes down, implies

\[ I_{\text{ds}} \sim \exp(V_g) \]
Sad fact of life: Leakage current

Even when a logic gate isn’t switching, it burns power.

\[ 0V = V_{IN} \]

\[ I_{Gate} \]

\[ I_{Sub} \]

\[ V_{OUT} \]

\[ C_L \]

\[ I_{Sub}: \text{Even when this nFet is off, it passes an Ioff leakage current.} \]

We can engineer any Ioff we like, but a lower Ioff also results in a lower Ion, and thus the lower the clock speed.
Data Plot: Transistor Off Current @ 25nm

\[ I_{ds} = I_0 \left[ \exp\left(\frac{\kappa V_g - V_s}{V_0}\right) \right] \left[ 1 - \exp\left(-\frac{V_{ds}}{V_0}\right) \right] \]

- \( I_{ds} \) = 1.2 mA = \( I_{on} \)
- \( 0.25 = V_t \)
- \( 0.7 = V_{dd} \)
- \( I_{off} \approx 10 \text{nA} \)
Simple model for “on” transistor ...

\[ V_d = 2V \quad V_g = 1V \quad V_s = V_{\text{sub}} = 0V \]

\[ I_d = \mu A \]

\[ Q = CV \quad f(\text{length}, \text{velocity}) \]

\[ I_{ds} = \left( \frac{\mu \epsilon W}{LD} \right) [V_g - V_{\text{th}}] [V_d] \]

If \( V_d > V_g - V_{\text{th}} \), channel physics change:

\[ I_{ds} = \left( \frac{\mu \epsilon W}{2LD} \right) [V_g - V_{\text{th}}]^2 \]

\[ W = \text{transistor width}, \quad L = \text{length}, \quad D = \text{capacitor plate distance} \]

\[ \mu \text{ is velocity, } \epsilon \text{ is dielectric constant} \]
Data Plot: Transistor On Current @ 25nm

\[ I_{ds} = \left[ \frac{(\mu \epsilon W)}{2LD} \right] [V_{gs} - V_{th}]^{1.x} \]

25nm is a small device! The simple model no longer captures how "on" device works.

\[ 0.25 = V_t \]

\[ 0.7 = V_{dd} \]

\[ I_{off} = 0 \text{ ???} \]

\[ 1.2 \text{ mA} = I_{on} \]
Dynamic Memory (DRAM)
Recall: Capacitors in action

Because the dielectric is an insulator, and does not conduct.

After circuit “settles” ...

Q = C V = C * 1.5 Volts (D cell)

Q: Charge stored on capacitor

C: The capacitance of the device: function of device shape and type of dielectric.

After battery is removed:

Still, Q = C * 1.5 Volts

Capacitor “remembers” charge
DRAM cell: 1 transistor, 1 capacitor

“Bit Line”  “Word Line”  Vdd

Capacitor

Word Line and Vdd run on “z-axis”

Word Line and Vdd run on “z-axis”

Why Vcap values start out at ground.

Diode leakage current.

Vdd

Vcap

Diode leakage current.
A 4 x 4 DRAM array (16 bits) ....
Invented after SRAM, by Robert Dennard

United States Patent Office

3,387,286
Patented June 4, 1968

1

FIELD-EFFECT TRANSISTOR MEMORY
Robert H. Dennard, Croton-on-Hudson, N.Y., assignor to
International Business Machines Corporation, Armonk,
N.Y., a corporation of New York
Filed July 14, 1967, Ser. No. 653,415
21 Claims. (Cl. 340—173)

2

tinent in disclosing various concepts and structures which
have been developed in the application of field-effect tran-
sistors to different types of memory applications, the pri-
mary thrust up to this time in conventional read-write
random access memories has been to connect a plurality
of field-effect transistors in each cell in a latch config-
uration. Memories of this type require a large number of
active devices in each cell and therefore each cell re-

UC Regents Fall 2008 © UCB
DRAM Circuit Challenge #1: Writing

Vdd - Vth. Bad, we store less charge. Why do we not get Vdd?

\[ I_{ds} = \frac{(\mu \varepsilon W)}{(2LD)} [V_{gs} - V_{th}]^2 \]
but “turns off” when \( V_{gs} \leq V_{th} \)

\[ V_{gs} = V_{dd} - V_c. \] When \( V_{dd} - V_c = V_{th} \), charging effectively stops!
DRAM Challenge #2: Destructive Reads

Bit Line (initialized to a low voltage)

Word Line

Vgs

0 -> Vdd Vc -> 0

+++++++ (stored charge from cell)

Raising the word line removes the charge from every cell it connects too!

Must write back after each read.
DRAM Circuit Challenge #3a: Sensing

Assume $C_{cell} = 1 \text{ fF}$

Word line may have 2000 nFet drains, assume word line $C$ of 100 fF, or $100\times C_{cell}$.

$C_{cell}$ holds $Q = C_{cell} \times (V_{dd} - V_{th})$

When we dump this charge onto the word line, what voltage do we see?

$$dV = \frac{C_{cell} \times (V_{dd} - V_{th})}{100 \times C_{cell}}$$

$$dV = \frac{(V_{dd} - V_{th})}{100} = \text{tens of millivolts!}$$

In practice, scale array to get a 60mV signal.
How do we reliably sense a 60mV signal?

Compare the word line against the voltage on a “dummy” word line.

“Dummy” word line. Cells hold no charge.

Word line to sense

“sense amp”

Dummy word line
DRAM Challenge #4: Leakage ...

Parasitic currents leak away charge.

Solution: “Refresh”, by reading cells at regular intervals (tens of milliseconds)
Cell capacitor holds 25,000 electrons (or less). Cosmic rays that constantly bombard us can release the charge!

Solution: Store extra bits to detect and correct random bit flips (ECC).
DRAM Challenge 6: Yield

If one bit is bad, do we throw chip away?

Solution: add extra word lines (i.e. 80 when you only need 64). During testing, find the bad word lines, and use high current to burn away “fuses” put on chip to remove them.

Extra word lines. Used for “sparing”.

[...]
Energy Storage and Process Scaling

Recall: Process Scaling ("Moore's Law")

Due to reducing $V$ and $C$ (length and width of Cs decrease, but plate distance gets smaller).

Recent slope more shallow because $V$ is being scaled less aggressively.

DRAM Challenge 7: Scaling

Each generation of IC technology, we shrink width and length of cell.

As $C_{cell}$ and drain capacitances scale together, number of bits per word line stays constant.

$$dV \approx 60 \text{ mV} = \frac{[C_{cell} \cdot (V_{dd} - V_{th})]}{[100 \cdot C_{cell}]}$$

Problem 1: Number of arrays per chip grows!
Problem 2: $V_{dd}$ may need to scale down too!

Solution: Constant Innovation of Cell Capacitors!
Poly-diffusion Ccell is ancient history

Word Line and Vdd run on “z-axis”
Modern cells: “trench” capacitors

Figure 4

SEM photomicrograph of 0.25-μm trench DRAM cell suitable for scaling to 0.15 μm and below. Reprinted with permission from [17]; © 1995 IEEE.
Modern cells: “stacked” capacitors
Memory Arrays

SYNCHRONOUS DRAM

MT48LC32M4A2 – 8 Meg x 4 x 4 banks
MT48LC16M8A2 – 4 Meg x 8 x 4 banks
MT48LC8M16A2 – 2 Meg x 16 x 4 banks

For the latest data sheet, please refer to the Micron Web site: www.micron.com/dramds
People buy DRAM for the bits. “Edge” circuits are overhead.

So, we amortize the edge circuits over big arrays.
A “bank” of 32 Mb (128Mb chip -> 4 banks)

- 4096 rows
- 2048 columns
- 33,554,432 usable bits (tester found good bits in bigger array)

12-bit row address input

4096 decoder

2048 bits delivered by sense amps

Select requested bits, send off the chip
Recall DRAM Challenge #3b: Sensing

How do we reliably sense a 60mV signal?

Compare the word line against the voltage on a "dummy" word line.

[...]

"Dummy" word line. Cells hold no charge.

Word line to sense

"sense amp"

Dummy word line
Corresponds to row read into sense amps

Slow! This 7.5 ns period DRAM (133 MHz) can do row reads at only 75 ns (13 MHz).

Plus, need to add selection time.

DRAM has high latency to first bit out. A fact of life.

12-bit row address input

1 of 4096 decoder

4096 rows

2048 columns

33,554,432 usable bits
(tester found good bits in bigger array)

2048 bits delivered by sense amps

Select requested bits, send off the chip

Cal
An ill-timed refresh may add to latency

Parasitic currents leak away charge.

Solution: “Refresh” by reading cells at regular intervals (tens of milliseconds)
Latency is not the same as bandwidth!

Thus, push to faster DRAM interfaces

What if we want all of the 2048 bits?
In row access time (75 ns) we can do
10 transfers at 133 MHz.
8-bit chip bus -> 10 x 8 = 80 bits << 2048
Now the row access time looks fast!

12-bit row address input

1 of 4096 decoder

4096 rows
33,554,432 usable bits
(tester found good bits in bigger array)

2048 columns

2048 bits delivered by sense amps

Select requested bits, send off the chip
Sadly, it’s rarely this good ...

What if we want all of the 2048 bits?
The “we” for a CPU would be the program running on the CPU.
It’s more likely that ... 20% of the memory accesses need a new row access ... not good.

- 4096 rows
- 33,554,432 usable bits (tester found good bits in bigger array)
- 2048 columns
- 2048 bits delivered by sense amps
- Select requested bits, send off the chip
**Columns:** Design the right interface for CPUs to request the subset of a column of data it wishes:

- 2048 bits delivered by sense amps
- Select requested bits, send off the chip

**Interleaving:** Design the right interface to the 4 memory banks on the chip, so several row requests run in parallel.

- Bank 1
- Bank 2
- Bank 3
- Bank 4
Off-chip interface for the Micron part ...

A clocked bus protocol (133 MHz)

Note! This example is best-case! To access a new row, a slow ACTIVE command must run before the READ.

DRAM is controlled via commands (READ, WRITE, REFRESH, ...)

Synchronous data output.

From Micron 128 Mb SDRAM data sheet (on "resources" web page)
Opening a row before reading ...

CAS Latency

44 ns

6 ns

70 ns between row opens
Interleave: Access all 4 banks in parallel

NOTE: Each READ command may be to any bank. DQM is LOW.

Figure 8
Random READ Accesses
ECC and DRAM
ECC: Memory in an imperfect world

ECC == Error Correcting Codes

Detecting and correcting RAM bit errors
DRAM Challenge: Cosmic Rays ...

Cell capacitor holds 25,000 electrons (or less). Cosmic rays that constantly bombard us can release the charge!

Cosmic ray hit.
Practical effect of a cosmic ray ...

```
ADDIU R1, R0, 7
SW R1, 100(R0)
```

Address 100: 0b00...0111

Cosmic ray hit.

```
LW R1, 100(R0)
```

Address 100: 0b00...0011

After LW, R1 holds 3 but it should hold 7.
Bit flips on memory holding instructions are bad too!
To “detect” errors -- add ‘P’, a parity bit

Extra “parity” bit for every word. Not seen by software. Hardware computes it on every write, so that the number of 1’s in every 33 bit word is even (even parity).

Address 100: 0b00...0111 1

Does this work if two bits flip? If three?

Cosmic ray hit.

Address 100: 0b00...0011 1

On a read, count the number of 1s. If odd, a bit flipped.

So, halt the program and reboot? Application may know if this bit matters, but there’s no API to ask it ...
Error Correction: Hamming Codes ...


Famous quote:

“Computers are not for numbers.
Computers are for understanding.”
Trick: Compute parity of subsets of bits

Consider 4 bit words. Add 3 parity bits.

$D_3D_2D_1D_0$  $P_2P_1P_0$

$0110$  $???$

Each parity bit computed on a subset of bits

$P_2 = D_3 \text{xor} D_2 \text{xor} D_1 = 0 \text{xor} 1 \text{xor} 1 = 0$

$P_1 = D_3 \text{xor} D_2 \text{xor} D_0 = 0 \text{xor} 1 \text{xor} 0 = 1$

$P_0 = D_3 \text{xor} D_1 \text{xor} D_0 = 0 \text{xor} 1 \text{xor} 0 = 1$

Use this word bit arrangement

$D_3D_2D_1P_2D_0P_1P_0$

$0110011$

"Just believe" for now, we will justify later ...
Case #1: No cosmic ray hits

We write: \( D_3D_2D_1P_2D_0P_1P_0 \)

Later, we read: \( D_3D_2D_1P_2D_0P_1P_0 \)

No errors ... but how do we know that?

On readout we compute:

\[
\begin{align*}
P_2 \text{ xor } D_3 \text{ xor } D_2 \text{ xor } D_1 &= 0 \text{ xor } 0 \text{ xor } 1 \text{ xor } 1 = 0 = C_2 \\
\text{If } C_2C_1C_0 &= 0 \\
\text{no errors} \\

P_1 \text{ xor } D_3 \text{ xor } D_2 \text{ xor } D_0 &= 1 \text{ xor } 0 \text{ xor } 1 \text{ xor } 0 = 0 = C_1 \\

P_0 \text{ xor } D_3 \text{ xor } D_1 \text{ xor } D_0 &= 1 \text{ xor } 0 \text{ xor } 1 \text{ xor } 0 = 0 = C_0 \\

\text{These equations come from how we computed } P_2P_1P_0
\end{align*}
\]

\[
\begin{align*}
P_2 &= D_3 \text{ xor } D_2 \text{ xor } D_1 = 0 \text{ xor } 1 \text{ xor } 1 = 0 \\
P_1 &= D_3 \text{ xor } D_2 \text{ xor } D_0 = 0 \text{ xor } 1 \text{ xor } 0 = 1 \\
P_0 &= D_3 \text{ xor } D_1 \text{ xor } D_0 = 0 \text{ xor } 1 \text{ xor } 0 = 1
\end{align*}
\]
Case #2: A cosmic ray hits ...

We write: $D_3D_2D_1P_2D_0P_1P_0$

Later, we read: $D_3D_2D_1P_2D_0P_1P_0$

Cosmic ray hit D1. But how do we know that?

On readout we compute:

$$P_2 \text{ xor } D_3 \text{ xor } D_2 \text{ xor } D_1 = 0 \text{ xor } 0 \text{ xor } 1 \text{ xor } 0 = 1 = C_2$$

$$C_2C_1C_0 = b101 = 5$$

$$P_1 \text{ xor } D_3 \text{ xor } D_2 \text{ xor } D_0 = 1 \text{ xor } 0 \text{ xor } 1 \text{ xor } 0 = 0 = C_1$$

What does “5” mean?

$$P_0 \text{ xor } D_3 \text{ xor } D_2 \text{ xor } D_1 = 1 \text{ xor } 0 \text{ xor } 0 \text{ xor } 0 = 1 = C_0$$

Note: we number the least significant bit with 1, not 0! 0 is reserved for “no errors”.

The position of the flipped bit! To repair, just flip it back ...

7 6 5 4 3 2 1

$D_3D_2D_1P_2D_0P_1P_0$

0 1 0 0 0 1 1
Why did we choose “3” parity bits?

Consider 4 bit words. Add 3 parity bits.

\[ D_3D_2D_1D_0 \quad \text{P}_2\text{P}_1\text{P}_0 \quad \text{A C}_i \text{ in } C_2C_1C_0 \text{ exists for each P}_i. \]

Observation: The \( C_2C_1C_0 \) bits need to encode the “no error” condition, plus a number for each bit (both data and parity bits)

For “p” parity bits and “d” data bits:

\[ d + p + 1 \leq 2^p \]
Why did we arrange bits as we did?

Consider 4 bit words. Add 3 parity bits.

Let's denote:
- \( D_3D_2D_1D_0 \) for the 4 data bits,
- \( P_0P_1P_2 \) for the 3 parity bits.

How do we re-arrange bits?

Start by numbering, 1 to 7. With this order, an odd parity means an error in 1, 3, 5, or 7. So, \( P_0 \) is the right parity bit to use:

- An odd parity means a mistake must be in 2, 3, 6, or 7 -- the four numbers possible if \( C_1 = 1 \).

A \( C_i \) in \( C_2C_1C_0 \) exists for each \( P_i \).

Etc ... each bit narrows down the suspect bits, until it is certain.
Why did we arrange bits as we did?

Consider 4 bit words. Add 3 parity bits.

$$D_3D_2D_1D_0$$

$$P_2P_1P_0$$

$$P_2 = D_3 \text{xor} D_2 \text{xor} D_1$$

$$P_1 = D_3 \text{xor} D_2 \text{xor} D_0$$

$$P_0 = D_3 \text{xor} D_1 \text{xor} D_0$$

7 bits can code 128 numbers, but only 16 of these numbers are legal.

It takes 3 bit flips to move from one legal number to another (for all 16 numbers)

If only one bit flips, we can always figure out the “closest” legal number, and correct.
What if 2 cosmic rays hit?

We write: \[D_3D_2D_1P_2D_0P_1P_0\]

Later, we read: \[D_3D_2D_1P_2D_0P_1P_0\]

On readout we compute:

\[P_2 \text{ xor } D_3 \text{ xor } D_2 \text{ xor } D_1 = 0 \text{ xor } 0 \text{ xor } 1 \text{ xor } 0 = 1 = C_2\]
\[P_1 \text{ xor } D_3 \text{ xor } D_2 \text{ xor } D_0 = 0 \text{ xor } 0 \text{ xor } 1 \text{ xor } 0 = 1 = C_1\]
\[P_0 \text{ xor } D_3 \text{ xor } D_2 \text{ xor } D_0 = 1 \text{ xor } 0 \text{ xor } 0 \text{ xor } 0 = 1 = C_0\]

What does “7” mean?

Note: it does do 2-bit “detect” (since C3 C2 C1 does not code 0), but it does not let us know that we can’t correct ...

“Correcting” this bit makes things worse! Thus, this code corrects “single” bits only.

Cosmic ray hit D1 and P1.

\[C_2C_1C_0 = b111 = 7\]
Next Monday:

<table>
<thead>
<tr>
<th>M</th>
<th>Student Presentations:</th>
</tr>
</thead>
<tbody>
<tr>
<td>11/3</td>
<td>Implementation Design Review</td>
</tr>
</tbody>
</table>

This Friday:
RAMP Gold Meeting, BWRC

Final Presentation
Fri, Dec 5