DRAM and DDR Memory

Dynamic Random Access Memory Architecture

Your Name

February 2, 2026

Why DRAM Exists

Two Memory Types for Different Purposes:

- **SSDs:** Long-term storage
  - 3D arrays with trillions of cells
  - Terabytes capacity
  - Access time: $\sim$50 microseconds
- **DRAM:** Working memory
  - 2D arrays with billions of capacitor cells
  - Gigabytes capacity
  - Access time: $\sim$17 nanoseconds (**3000$\times$ faster**)

CPU processes only data stored in DRAM

How SSD and DRAM Work Together

Data Transfer Process:

- Data copied from SSD to DRAM during loading
- **Prefetching:** Moving data before it's needed
- System stores terabytes on SSD
- Frequently accessed data retrieved from DRAM in nanoseconds

Practical Application:

- Programs require loading into DRAM before execution
- Large applications need significant DRAM capacity
- Without DRAM: System performance 3,000$\times$ slower
- DRAM requires continuous power to maintain data

DRAM Module Overview (DIMM)

Physical Structure:

- DIMM = Dual Inline Memory Module
- Typical configuration: 8 DRAM chips per module
- Multiple slots on motherboard (usually 4)

System Integration:

- Connected to CPU via memory channels through motherboard
- Two independent memory channels
- Memory controller in CPU manages all communication
- CPU also manages separate connections to storage devices

Memory Channels in DDR5

Channel Structure:

- Each memory channel divided into Channel A and Channel B
- Independent operation, 32 bits transferred per channel

Signal Lines per Channel:

- 32 data wires for actual data transfer
- 21 address wires for memory location
- 7 control wires for commands

Parallel Processing:

- All 4 chips on channel receive same address/commands
- Data lines divided: each chip handles 8 bits
- Power managed by dedicated chips on module

Inside a DRAM Chip

Physical Components:

- Packaging contains interconnection matrix
- Ball grid array connects to die (main chip)

Die Organization (2 GB chip):

- 8 bank groups $\times$ 4 banks $=$ **32 banks total**
- Each bank: 65,536 rows $\times$ 8,192 columns
- Total: $\sim$17 billion memory cells
- Complex network of wires and supporting circuits

31-Bit Memory Addressing

Accessing 17 billion cells requires 31-bit address:

- **3 bits:** Bank group selection (8 groups)
- **2 bits:** Bank selection (4 per group)
- **16 bits:** Row selection (65,536 rows)
- **10 bits:** Column group (8,192 $\div$ 8)

Transmission Optimization:

- Address sent in two parts using 21 wires
- Part 1: Bank group + bank + row
- Part 2: Column address
- Each access reads/writes 8 bits simultaneously

Memory Cell Structure: 1T1C

Components (each stores 1 bit):

1. **Capacitor:**
   - Deep trench in silicon (few dozen nanometers)
   - Two conductive surfaces with dielectric insulator
   - 1V = binary 1, 0V = binary 0
2. **Access Transistor:**
   - Wordline activates transistor gate
   - Bitline connects to transistor channel
   - Controls capacitor access

How Memory Cells Store Data

Write Operation:

- Wordline ON $\rightarrow$ connects capacitor to bitline
- Charge flows to write 1 or discharge for 0

Read Operation:

- Wordline ON $\rightarrow$ measure capacitor charge
- Charge amount indicates stored value

Data Retention Challenge:

- Wordline OFF isolates capacitor
- Electron leakage through tiny transistor
- Requires periodic refresh

Memory Array Organization

Array Structure:

- **Wordlines:** Rows connecting to transistor gates
- **Bitlines:** Columns connecting to transistor channels
- Different vertical layers (no physical contact)

Operation Rules:

- Active wordline connects entire row to bitlines
- **Only ONE wordline active at a time**
- Multiple active wordlines cause data interference

Cell Access:

- Row decoder: 16 bits activate single wordline
- Column multiplexer: 10 bits select 8 bitlines

Reading from Memory Cells

Read Process:

1. CPU sends read command + 31-bit address
2. Select bank (5 bits)
3. Turn off all wordlines, precharge bitlines to 0.5V
4. Activate one wordline using 16-bit row address
5. Capacitors connect to bitlines:
    - Stored 1: bitline voltage increases
    - Stored 0: bitline voltage decreases
6. Sense amplifiers detect and amplify voltage change
7. Drive bitlines to full 1V or 0V
8. Row now "open" with all 8,192 bitlines at correct values
9. Column multiplexer selects 8 bitlines (10-bit address)
10. Read driver transmits 8 bits to CPU

Writing to Memory Cells

Write Process:

1. CPU sends write command + address + 8 data bits
2. Select bank (5 bits)
3. Isolate capacitors, precharge bitlines to 0.5V
4. Activate row (16-bit address)
5. Sense amplifiers open row
6. Column multiplexer connects 8 bitlines (10-bit address)
7. Write drivers override bitline voltages:
   - 1V for binary 1
   - 0V for binary 0
8. New voltages update capacitor charges
9. 8 bits successfully written

Note: All 4 chips operate concurrently with same address but different data

Refresh Operation

Why Refresh:

- Nanoscale transistors leak electrons over time
- Without refresh: data loss

Refresh Process:

1. Close all rows
2. Precharge bitlines to 0.5V
3. Open one row
4. Sense amplifiers restore full voltage (1V or 0V)
5. Repeat for each row

Timing:

- 50 nanoseconds per row
- ∼3 milliseconds for all 65,536 rows
- Occurs every 64 milliseconds per bank

Row Hits vs Row Misses

Row Hit (Page Hit):

- Address in already-open row
- Use only 10-bit column address
- Much faster (skip row opening)
- Can occur repeatedly

Row Miss:

- Address in different row
- Must close current row, open new row
- Significantly slower

Timing Parameters (clock cycles):

- Row hit to data delivery
- Row opening time
- Precharge time
- Row activation to precharge interval

Systems optimized to maximize row hits, avoid thrashing

Why 32 Banks?

Parallelism Benefits:

- Each bank operates independently
- Multiple rows open simultaneously across banks
- Increases row hit probability
- Reduces average access time

Bank Groups:

- 8 bank groups $\times$ 4 banks $=$ 32 total
- Refresh one bank per group while using others
- Minimizes refresh impact

Array Efficiency:

- Banks are tall and narrow
- Combined: 65,536 rows $\times$ 262,144 columns
- 31 divisions create flexibility for operations

Burst Buffer Optimization

Design:

- 128-bit temporary storage buffer
- 10-bit column address split: 6 bits (mux) + 4 bits (buffer)

Operation:

- 6 bits connect 128 cells to buffer
- 4 bits select 8 data locations from buffer
- Cycle through combinations: 16 sets $\times$ 8 bits
- **Burst length = 16**
- Total: 1,024 bits accessed quickly per chip

Benefits:

- Fast sequential data access
- Maintains random access granularity

Subarrays and Hierarchical Design

Challenge:

- Large 65,536 $\times$ 8,192 arrays
- Extremely long wordlines and bitlines

Solution:

- Subdivide into 1,024 $\times$ 1,024 subarrays
- Intermediate sense amplifiers per subarray
- Hierarchical row decoding

Benefits:

- Shorter bitlines $\rightarrow$ smaller capacitors
- Shorter wordlines $\rightarrow$ reduced capacitive load
- Faster transistor activation
- Improved performance

Differential Pair Architecture

Design:

- Two bitlines per sense amplifier
- Alternating rows connect to different bitlines
- Half bitlines active, half passive at any time

Cross-Coupled Inverter:

- Active bitline $= 1 \rightarrow$ passive $= 0$
- Active bitline $= 0 \rightarrow$ passive $= 1$
- Creates differential pair (opposite values)
- Passive bitline not connected to cells

Key Benefits:

1. Easy precharging: charge equalizes to 0.5V
2. Noise immunity from opposite charges
3. Reduced parasitic capacitance

Conclusion

Key Concepts Covered:

- DRAM role in memory hierarchy
- Internal structure: banks, arrays, cells
- Core operations: read, write, refresh
- Performance optimizations

DRAM Performance:

- 4.8 billion operations per second
- 17 nanosecond access time
- Continuous refresh (16 times/second per bank)

Significance:

- 3,000$\times$ faster than SSD access
- Critical for modern computing performance
- Enables efficient program execution

Thank You!