EE201: Digital Circuits and Systems

Section 6 – Memory

Data memory types:

1. Random Access Memory which can be read & written
   - Static & Dynamic RAM

2. Read Only Memory which retains data
   - PROM, EPROM, EEPROM, Flash

Programmable Logic:

1. Programmable Arrays
   - PLDs, PALs, GALs
2. Complex Programmable Devices
   - CPLD, FPGA technology
6.1 Static RAM (SRAM)

- Static Random Access Memory
  - Static: Data value is retained as long as V_{DD} is present.
  - Random Access: Any location can read at a point in time. (Doesn’t need sequential addresses)

SRAM can be built using either:

- D-type latch
- 6-transistor CMOS RAM cell

6.1.1 D-type Latch

- Used for building CPU registers, etc
- Derived from inverted S-R flipflop

Inverted S-R flip-flop:

<table>
<thead>
<tr>
<th>/S</th>
<th>/R</th>
<th>Q'</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>X</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Q</td>
</tr>
</tbody>
</table>

D-type latch

<table>
<thead>
<tr>
<th>E</th>
<th>D</th>
<th>/S</th>
<th>/R</th>
<th>Q'</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>No Change</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>No Change</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

When the Enable line is zero (En=0)

- /S = /R = 1 and the inverting SR flipflop retains its previous value.

When the enable line is high (En=1)

- The value of data line D is latched into the flipflop.
Each BIT would need 16 transistors (NAND gate = 4 transistors)
- For large SRAM modules not very efficient.
  - 1-MB SRAM -> 8-Mb -> 128 Million transistors

6.1.2 Edge Triggered D-type Register
- For use with a combinational circuit it is more important to have devices respond to clock edges.
  - D-type Latch ‘works’ when $\text{En}=1$ or $\text{En}=0$
  - D-type Register ‘works’ when $\text{En}$ is rising or falling.

Edge triggered flip-flop for use in synchronous circuits.
- Uses 2 D-type transparent latches(Red Boxes) and 2 NOT gates

When the clock is low (Clk=0)
- The first D-type latch is ON,
- The value of D latched into first flipflop.

When clock goes high (Clk=1)
- The first D-type latch switches OFF and the second D-type latch is enabled.
- The output of latch 1 propagates through the second flipflop to the output.

Value of output is retained until next rising edge

Falling clock edges: Remove leftmost inverter from the circuit.
6.1.3 6-Transistor Cell (Cross Coupled Inverter)

- For larger SRAM modules the above circuit is not very efficient
  - Transistor count per bit is too high

TO READ:
- BIT lines are charged high
- Enable line WL is pulled high, switching access transistors M5 and M6 on
- If value stored in /Q is 0, value is accessed through access transistor M5 on /BL.
- If value stored in Q is 1, charged value of Bit line BL is pulled up to $V_{DD}$.
- Value is ‘sensed’ on BL and /BL.

TO WRITE:
- Apply value to be stored to Bit lines BL and /BL
- Enable line WL is triggered and input value is latched into storage cell
- BIT line drivers must be stronger than SRAM transistor cell to override previous values

While Enable line is held low, the inverters retain the previous value
Could use tri-state WE line on BIT to drive into specific state.
Transistor count per bit is only 6 + (line drivers & sense logic)
6.1.4 Addressed SRAM
- Can view RAM as N-bit by M-word black box:
  - $N$ input lines  \( D_{IN} \)
  - $N$ output lines  \( D_{OUT} \)
  - $A$ address lines \( 2^A = M \)
  - $WE$ write enable line  \( WE \)

6.1.4.1 Single SRAM Bit

When $A = 0$,
- Latch Enable is off.
  - Data cannot be written into the D-type latch
  - \( D_{OUT} = 0 \).

When $A = 1$
- Latch is Enabled
  - If $W = 1$ (Data-Write)
    - Data at \( D_{IN} \) can be written into the D-type latch
    - Output gate is enabled
  - If $W = 0$
    - New value on \( D_{IN} \) is not stored.
    - Output gate is enabled.

- Not very efficient since 1-bit address line can access 2 memory locations.
- This memory is 1-bit X 1-word RAM
  - Stores one 1-bit data value
### 6.1.4.1 1-bit X 2-word SRAM

<table>
<thead>
<tr>
<th>A</th>
<th>W</th>
<th>DI</th>
<th>FlipFlop Out</th>
<th>DO</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Q(t-1)</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Q(t-1)</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>Q(t-1)</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>Q(t-1)</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Q(t)</td>
<td>Q(t)</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>Q(t)</td>
<td>Q(t)</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

When address bit A1 = 0
- Cell1 is disabled and Cell0 is enabled
  - IF W = 1 : Value of D_IN is written to cell0
  - IF W = 0 : Data out is Cell0 OR 0

When address bit A1 = 1
- Cell0 is disabled and Cell1 is enabled
  - IF W = 1 : Value of D_IN is written to cell1
  - IF W = 0 : Data out is Cell1 OR 0

- Only 1 cell can be active at one time
- Output line is always driven by one cell
  - Important for shared bus
6.1.5 4-bit X 16-word SRAM

When CS = 1 AND A4 A3 A2 A1 = 0000
- Address decoder decodes A4-A1 to
  - 1000000000000000 (a0 = 1, a1-a15 = 0)
- Data at DI1 DI2 DI3 DI4 is written to address 0 when W = 1
- If W = 0, No new data is stored and address0 drives the output bus
- Contents of memory address 0 appear at output

Address decoder maps input address bits to row control signals
- Should only set one bit for every possible input
  - $2^A$ states where $A$ is the number of address lines

The CS (chip select) line allows the memory to be doubled with only one inverter [+ OR gates].
6.1.5 **Tri-State Outputs:**

- In previous examples, one location is enabled during each operation which can drive the output bus.

- If RAM is on shared bus, the RAM cannot be allowed to drive the bus at all times
  - Must have method of removing RAM from bus
  - Solution is to use Tri-State logic

![Diagram of Tri-State Outputs](image)

- Outputs from each cell are tri-state outputs.

- When not active the outputs are in high impedance.

- Can either use CS line to control when Hi-Z or another ‘global’ memory signal which controls the output OE

- Allows both other RAM cells and other devices to control data bus
6.2 Dynamic RAM (DRAM)

- SRAM requires a number of transistors per bit
  - Difficult to cost-effectively scale for larger memories

- DRAM utilises MOSFET capacitance to store data bit
  - Transistor per bit cost is approx 1

- SiO₂ insulates gate and substrate
  - Creating dielectric capacitor between gate and substrate

- Data bit is stored in this capacitance

- Each bit now only requires 1 MOSFET per bit.
  - However the charge stored in cell dissipates over time and must be recharged over time to avoid corruption

**DRAM Refresh**

- Must read data bit and write value back to cell.

- JEDEC standardises DRAM row refreshes at least every 64 ms.
  - All bits in row must be refreshed.

- Dedicated hardware control DRAM refresh
  - Refresh is transparent to user

- Above 64 Kbits, DRAM more economic than SRAM logic
  - Even with refresh.
6.2.1 DRAM Organisation

- DRAM is organised as “row by column” matrix.
  - Matrix stores $n$ 1-bit words
  - $N$ is determined by the number of address lines available

- Each matrix is parallelised to create word size memories
  - i.e.: 8 parallel 4Kx1-bit DRAM matrices creates an 4K * 8-bit RAM module
**Example**
An 8x8 array forms a 64 x 1 dynamic RAM

- The row and column select logic are comprised of address decoders.
  - 8-rows and 8-columns need 3-address bits each.

- Above block is 64x1-bit DRAM

- Diagram omits but matrix has 1 data I/O line.
  - Row and Column address control which bit is active
This block can be parallelised to create larger data word

Each bit of data word is read/wrote in parallel

**Example 2**

How to build 4k X 1 dynamic RAM?

Step 1: How to arrange row & columns
Step 2: 64 x 64 array provides 4096 (4K) bits
Step 4: How many address lines needed for 64-line decoder?

\[ 2^6 = 64 \]

Need 6 row lines and 6 column lines
Pin Requirements
- 12 Address Bits: 6-bit for row and 6-bits for column
- 3 Control Bits: WE, CS and OE
- 1 Data I/O bit bus
- \( V_{DD} \) and GND
Entire IC will require 18 pins

6.2.3 Multiplexed Address Lines
- To further reduce cost, DRAM uses multiplexed address
  - \( 4K \times 1 = 12 \) Address lines
  - \( 16M \times 1 = 24 \) Address lines

Since column address is independent of row address
- We can provide row address and then column address

- If 12-bit address bus is multiplexed we can address 16M DRAM
  with only 2 additional lines (RAS and CAS)
  - RAS : Row Address Strobe
  - CAS : Column Address Strobe

Multiplexed Address, Step by Step:
- Step 1: Latch out 12 row address bits
- Step 2: Strobe RAS line which causes DRAM to latch in row address
- Step 3: Wait until you are sure it has been registered and
latch out column address bits
- Step 4: Strobe CAS line which causes DRAM to latch in
  column address
- Step 5: Wait some period of time and read/write to data bus
6.2.4 DRAM Timing

- DRAM module is asynchronous
  - Timing depends on how long it takes to respond to each operation.

DRAM cannot be read as fast (or as easy) as SRAM
6.3 Read Only Memory (ROM)

- Disadvantage with RAM (static or dynamic) is that the contents of the memory are lost when power is removed.
  - Volatile memory: Content lost when $V_{DD}$ is removed (RAM)
  - Non-Volatile memory: Data is retained after $V_{DD}$ is gone.

ROM Types:
1) ROM : Most basic memory
2) PROM : Additional functionality on ROM
3) (UV) – EPROM : Can be reprogrammed
4) EEPROM : Can be reprogrammed (easier than EPROM)
5) Flash : Current technology (easier again)

6.3.1 ROM
- Most basic type of ROM is factory-programmed diode matrix
• The diode can be replaced by a multiple emitter transistor for each data word.
  o During manufacture, a diode is placed at those connections which are required by the customer.
  o Complete transistor array is programmed via fabrication mask
  o Those connections without diode cannot be changed later and vice-versa
    ▪ Memory is read-only

ROM is economic when several thousand devices are needed.

6.3.2 Programmable ROM (PROM)

• ROM device is factory-programmed device
  o Need to tell fab plant which connections to make/skip

• More useful solution would be to allow customer to program device in the field.
  o Place diode at every junction with nichrome fusible link in series.
  o Any link can be blown by selecting its address and applying a high voltage to its data output
  o Once fuse has been blown it cannot be repaired.
    ▪ Memory is read-only

Advantage over ROM
  • Device is ‘field-programmable’:
    o Customer can buy a blank PROM to programme
    o Manufacturer can made identical PROM for every customer, reducing cost

Disadvantages over ROM
  • Need 2 voltages:
    o Operating voltage
    o Programming voltage
6.3.3 Erasable PROM (EPROM)

- An EPROM can be programmed in a similar manner to a PROM
  - Each diode/fuse is replaced by 2-Gate MOS transistor
  - High programming voltage injects electrons into transistor
  - Process can be reversed by exposing device to UV light
    - UV lightwave reforms conductive channel

Advantage over ROM/PROM

- Device is field programmable plus the code can be re-written at a later date

Disadvantages over ROM/PROM

- Still need 2 voltages
- Quartz crystal is expensive.

6.3.4 Electrically Erasable PROM (EEPROM)

- Unlike EPROM, no UV light is required to erase memory.
  - Thin insulator layer allows voltage to erase

- Program/Erase voltage generated on-chip
  - Single $V_{DD}$ line required.

Advantage over ROM/PROM/EPROM

- Can be reprogrammed easier
- Only single power supply needed

Disadvantage over ROM/PROM/EPROM

- Thin gate insulator layer is damaged by erase/write operations
  - New EEPROM device has 1,000,000 cycles
6.3.5 Flash PROM (FLASH) [http://en.wikipedia.org/wiki/Flash_memory]

Flash memory stores information in an array of memory cells made from floating-gate transistors.

The floating gate may be conductive (typically polysilicon in most kinds of flash memory) or non-conductive (as in SONOS flash memory)

The ångström or angstrom (symbol Å) is equal to 0.1 nanometre or \(1 \times 10^{-10}\) metres.

**Programming Via Hot Electron Injection**

![Diagram of programming via hot electron injection](image-url)
Flash memory is an improvement on EEPROM technology and offers:

- Lower operational voltage and program voltage

**Flash Disadvantages**

- As gate insulator is thinned, the number of times it can be written is reduced. Flash might have 1,000,000 write cycles.
- Entire block (or page) must be erased at one time in flash, (byte can be erased in EEPROM)

**NAND Flash Accelerates Moore's Law**
All ROM memory is quite slow compared to DRAM and SRAM
- Typically try to use it to store code/data but execute code and manipulate data in RAM

6.3.5 ROM’s for Combinational Logic
- A suitably programmed ROM can generate any combinational logic function.
  - Number of inputs \( \leq \) Number of address lines
  - Number of outputs \( \leq \) Number of bits.

Example
- Converter from 3-bit binary to 7-segment display code. (0 = on, 1 = off)

<table>
<thead>
<tr>
<th>C</th>
<th>B</th>
<th>A</th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>f</th>
<th>g</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

An 8-word by 7-bit ROM is needed.
Characteristics of the various memory types

<table>
<thead>
<tr>
<th>Type</th>
<th>Volatile?</th>
<th>Writeable?</th>
<th>Erase Size</th>
<th>Max Erase Cycles</th>
<th>Cost (per Byte)</th>
<th>Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRAM</td>
<td>Yes</td>
<td>Yes</td>
<td>Byte</td>
<td>Unlimited</td>
<td>Expensive</td>
<td>Fast</td>
</tr>
<tr>
<td>DRAM</td>
<td>Yes</td>
<td>Yes</td>
<td>Byte</td>
<td>Unlimited</td>
<td>Moderate</td>
<td>Moderate</td>
</tr>
<tr>
<td>Masked ROM</td>
<td>No</td>
<td>No</td>
<td>n/a</td>
<td>n/a</td>
<td>Inexpensive</td>
<td>Fast</td>
</tr>
<tr>
<td>PROM</td>
<td>No</td>
<td>Once, with a device programmer</td>
<td>n/a</td>
<td>n/a</td>
<td>Moderate</td>
<td>Fast</td>
</tr>
<tr>
<td>EPROM</td>
<td>No</td>
<td>Yes, with a device programmer</td>
<td>Entire Chip</td>
<td>Limited (consult datasheet)</td>
<td>Moderate</td>
<td>Fast</td>
</tr>
<tr>
<td>EEPROM</td>
<td>No</td>
<td>Yes</td>
<td>Byte</td>
<td>Limited (consult datasheet)</td>
<td>Expensive</td>
<td>Fast to read, slow to erase/write</td>
</tr>
<tr>
<td>Flash</td>
<td>No</td>
<td>Yes</td>
<td>Sector</td>
<td>Limited (consult datasheet)</td>
<td>Moderate</td>
<td>Fast to read, slow to erase/write</td>
</tr>
<tr>
<td>NVRAM</td>
<td>No</td>
<td>Yes</td>
<td>Byte</td>
<td>Unlimited</td>
<td>Expensive (SRAM + battery)</td>
<td>Fast</td>
</tr>
</tbody>
</table>