CMR
INSTITUTE OF
TECHNOLOGY

USN □□□□□□□□□□

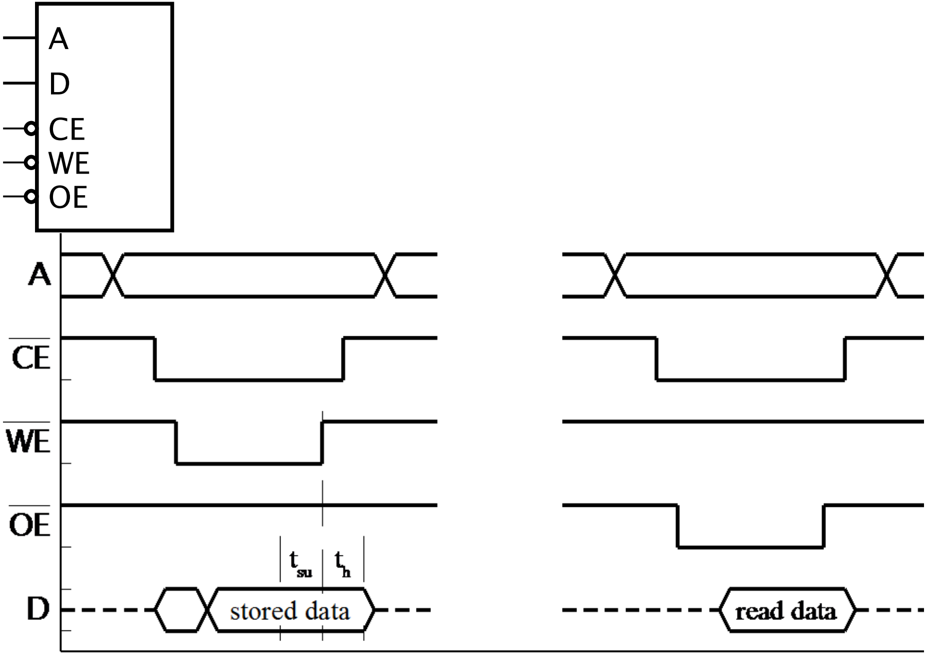
Internal Assesment Test – II/SCHEME AND SOLUTIONS

Sub:	DSDV							Code:	15EC663
Date:	/ 04 / 2019	Duration:	90 mins	Max Marks:	50	Sem:	VI	Branch:	Open Elective
	Answer all Questions								

		Marks	scheme	OBE CO	OBE RBT
1	**Explain synchronous and asynchronous Static RAMs.**	[10]		CO3	L2

SOLN:- Asynchronous SRAM

a. Data stored in 1-bit latch cells
b. Address decoded to enable a given cell
c. Usually use active-low control inputs
d. Not available as components in ASICs or FPGAs



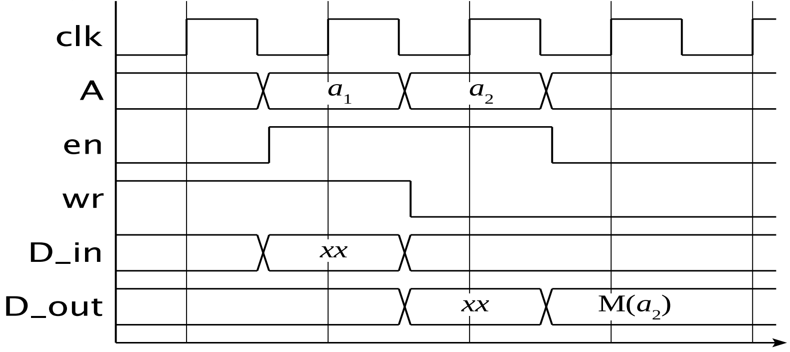
5M

Synchronous SRAM

a. Clocked storage registers for inputs
b. address, data and control inputs
c. stored on a clock edge
d. held for read/write cycle

Flow-through SSRAM

- no register on data output

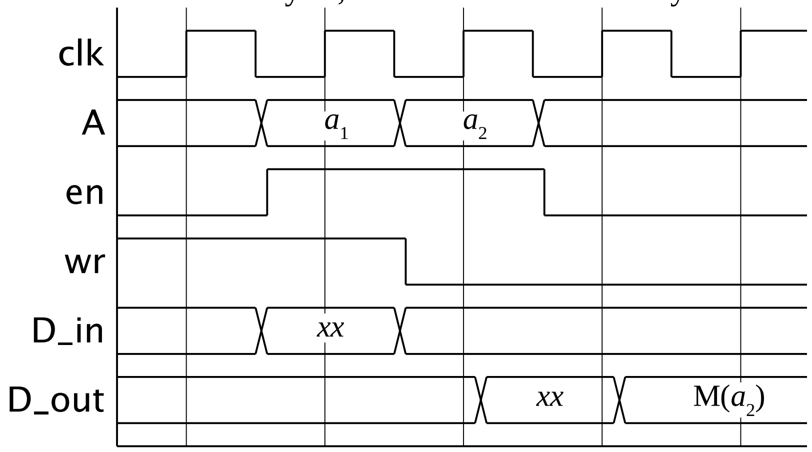


5M

Pipelined SSRAM

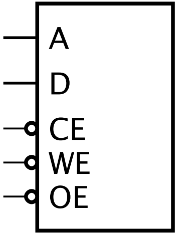
- Data output also has a register
    - More suitable for high-speed systems

- Access RAM in one cycle, use the data in the next cycle



3 Develop a verilog code for pipelined SSRAM.

Verilog code for pipelined SSRAM



```
reg        pipelined_en;
reg [15:0] pipelined_d_out;
...
Always @(posedge clk) begin
  if (pipelined_en) d_out <= pipelined_d_out;
  pipelined_en <= en;
  if (en)
    if (wr) begin
      data_RAM([a] <= d_in;  pipelined_d_out <= d_in;
    end
    else
      pipelined_d_out <= data_RAM[a];
end
```

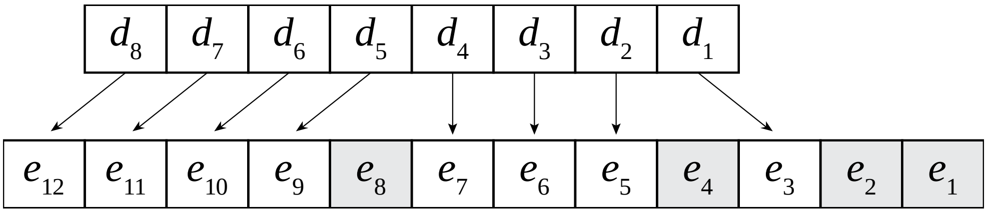
2 **What is Error correction? Obtain ECC for data byte: - 01100001. Find, if received ECC 000111000100 is having error, if so correct it.** [10] CO3 L3

Soln: Error-Correcting Codes (ECC)

- Allow identification of the flipped bit
- Hamming Codes
  - E.g., for single-bit-error correction of N-bit word, need $\log_2 N + 1$ extra bits
- Example: 8-bit word, $d_1 ... d_8$
  - 12-bit ECC code, $e_1 ... e_{12}$

3M

e_1, e_2, e_4, e_8 are check bits, the rest data



$$e_1 = e_3 \oplus e_5 \oplus e_7 \oplus e_9 \oplus e_{11}$$

$$e_2 = e_3 \oplus e_6 \oplus e_7 \oplus e_{10} \oplus e_{11}$$

$$e_4 = e_5 \oplus e_6 \oplus e_7 \oplus e_{12}$$

$$e_8 = e_9 \oplus e_{10} \oplus e_{11} \oplus e_{12}$$

ii) The check bits are

3M

- Every data bit covered by two or more check bits
- On write: Compute check bits and store with data
- On read: Recompute check bits and XOR with read check bits
  - result called the *syndrome*
- 0000 => no error
- If data bit flipped
  - covering bits of syndrome are 1
  - = binary code of flipped ECC bit
- If stored check bit flipped
  - that bit of syndrome is 1
- On error, unflip bit and rewrite memory location

$$e_1 = e_3 \oplus e_5 \oplus e_7 \oplus e_9 \oplus e_{11} = d_1 \oplus d_2 \oplus d_4 \oplus d_5 \oplus d_7 = 1 \oplus 0 \oplus 0 \oplus 0 \oplus 1 = 0$$

$$e_2 = e_3 \oplus e_6 \oplus e_7 \oplus e_{10} \oplus e_{11} = d_1 \oplus d_3 \oplus d_4 \oplus d_6 \oplus d_7 = 1 \oplus 0 \oplus 0 \oplus 1 \oplus 1 = 1$$

$$e_4 = e_5 \oplus e_6 \oplus e_7 \oplus e_{12} = d_2 \oplus d_3 \oplus d_4 \oplus d_8 = 0 \oplus 0 \oplus 0 \oplus 0 = 0$$

$$e_8 = e_9 \oplus e_{10} \oplus e_{11} \oplus e_{12} = d_5 \oplus d_6 \oplus d_7 \oplus d_8 = 0 \oplus 1 \oplus 1 \oplus 0 = 0$$

Thus the ECC word is 011000000110.

iii) Determine whether there is an error in the ECC word 000111000100, and if so, correct it.

4M

solution The check bits computed from the data bits of the ECC word are

$$e_1 = e_3 \oplus e_5 \oplus e_7 \oplus e_9 \oplus e_{11} = 1 \oplus 0 \oplus 1 \oplus 1 \oplus 0 = 1$$

$$e_2 = e_3 \oplus e_6 \oplus e_7 \oplus e_{10} \oplus e_{11} = 1 \oplus 0 \oplus 1 \oplus 0 \oplus 0 = 0$$

$$e_4 = e_5 \oplus e_6 \oplus e_7 \oplus e_{12} = 0 \oplus 0 \oplus 1 \oplus 0 = 1$$

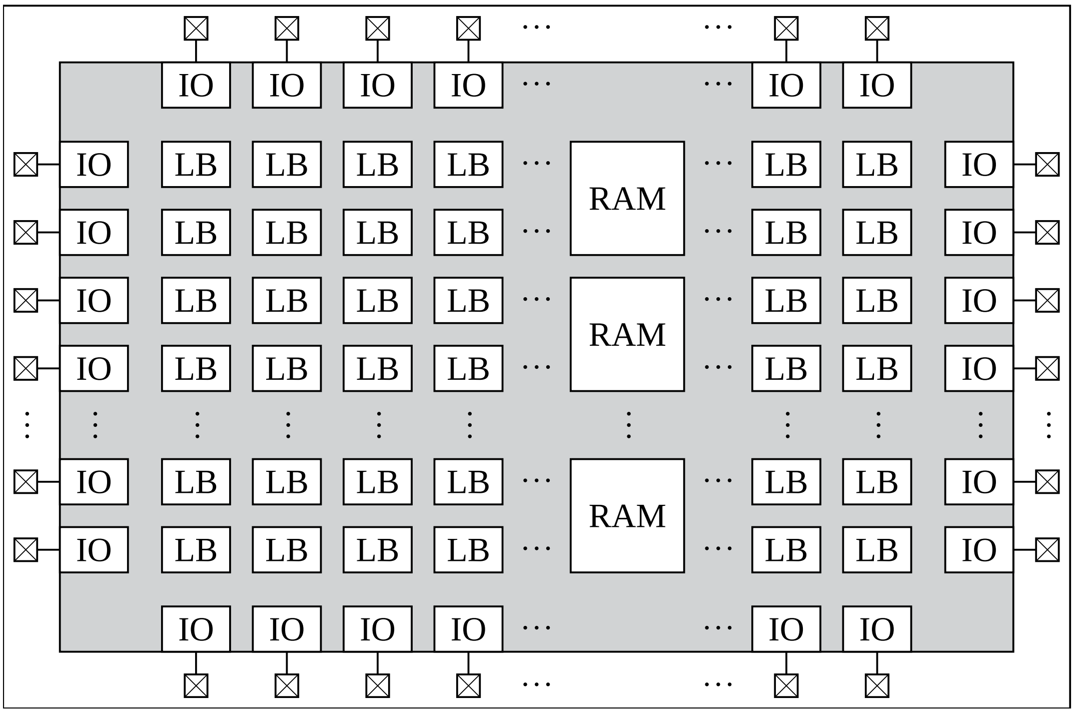
$$e_8 = e_9 \oplus e_{10} \oplus e_{11} \oplus e_{12} = 0 \oplus 0 \oplus 0 \oplus 1 = 1$$

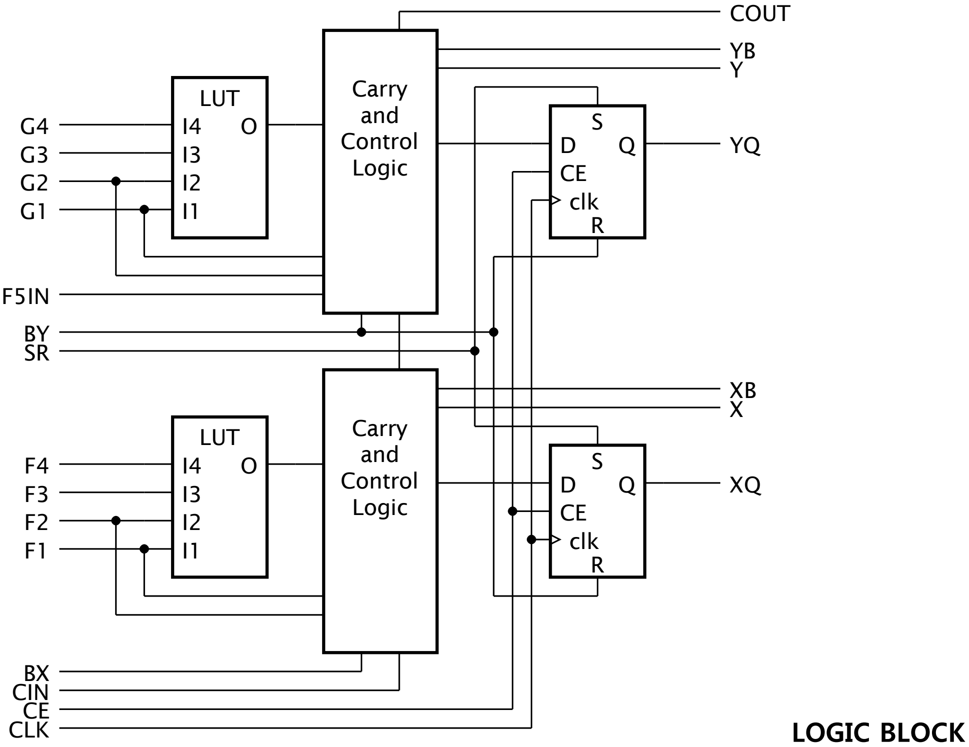
The syndrome is 1101 ⊕1000 _ 0101. Thus, there is an error in bit e5 of the read ECC. That bit should be flipped back from 0 to 1, giving the corrected ECC word 000111010100.

3 **Explain FPGA design with block diagram.** [10] CO4 L2

Soln:

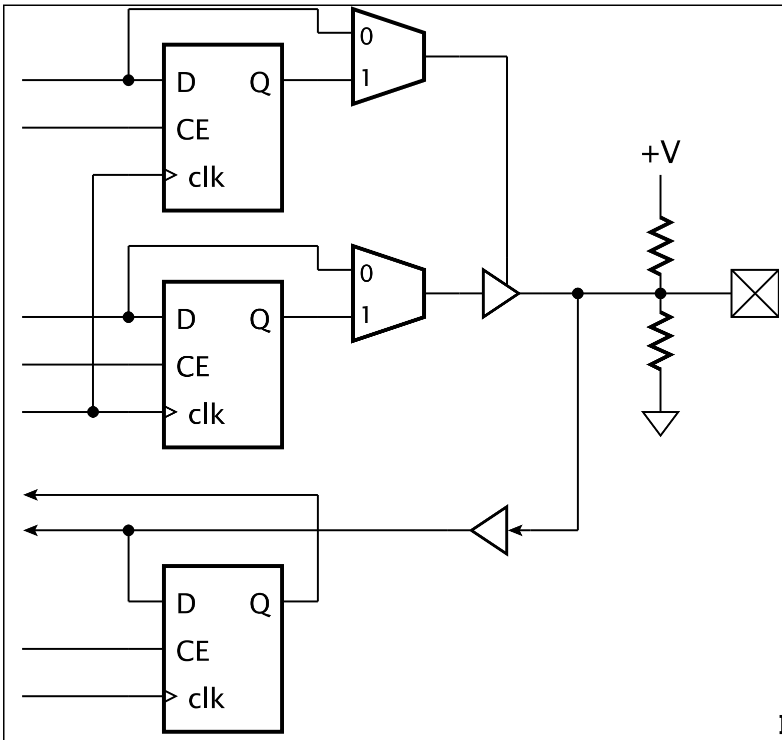
FPGA


IO
LB
RAM


COUT
YB
Y
LUT
I4
I3
I2
I1
O
G4
G3
G2
G1
Carry and Control Logic
S
D
Q
CE
clk
R
YQ
F5IN
BY
SR
XB
X
F4
F3
F2
F1
XQ
BX
CIN
CE
CLK
LOGIC BLOCK

4M

3M



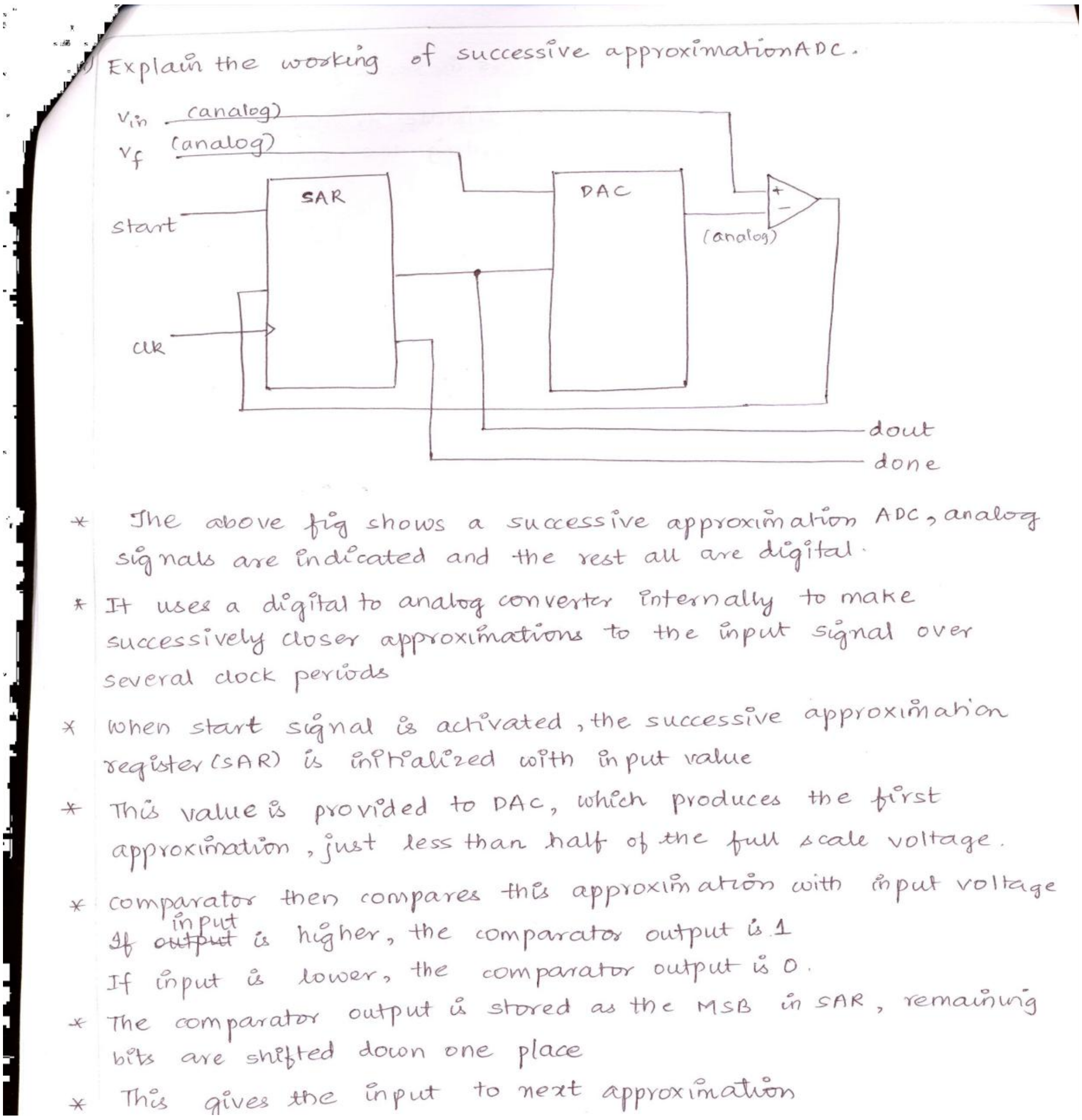
IO BLOCK

The basic elements within logic blocks are small 1-bit-wide asynchronous RAMs called lookup tables (LUTs). The LUT address inputs are connected to the inputs of the logic block. The content of an LUT determines the values of a Boolean function of the inputs. By programming the LUT content differently, we can implement any Boolean function of the inputs. The logic blocks also contain one or more flipflops and various multiplexers and other logic for selecting data sources and for connecting data to adjacent logic blocks. The logic block contains two such slices, together with a small amount of additional logic. Each slice consists of two 4-input LUTs, each of which can be programmed to implement any function of the four inputs. The carry and control logic consists of circuitry to combine the LUT outputs, an XOR gate and an AND gate for implementing adders and multipliers, as well as multiplexers that can be used to implement a fast carry chain.

The I/O block of an FPGA is typically organized as shown. The select inputs of the multiplexers are programmed to control whether the output is registered or combinational. The top flip-flop and multiplexer control the high-impedance state of the tristate driver that drives the pin as an output, and the middle flip-flop and multiplexer drive the output value. The output driver is programmable, allowing selection of logic levels (regular 5V TTL, low voltage TTL, or others) and control of the slew rate, that is, rate of voltage change at the output.

3M

4 **Explain the working of SAR Based ADC with example.** [10] CO4 L3



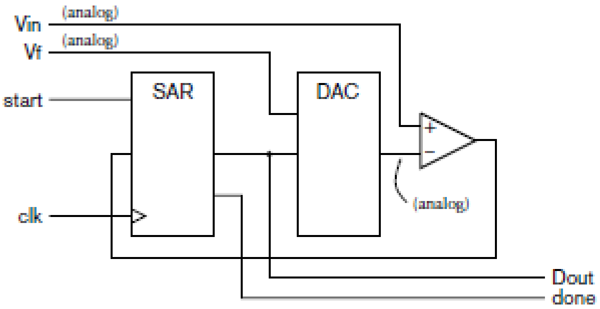
Explain the working of successive approximationADC.

Vin (analog)
Vf (analog)
SAR
DAC
start
(analog)
clk
dout
done

4M

* The above fig shows a successive approximation ADC, analog signals are indicated and the rest all are digital.
* It uses a digital to analog converter internally to make successively closer approximations to the input signal over several clock periods
* When start signal is activated, the successive approximation register (SAR) is initialized with input value
* This value is provided to DAC, which produces the first approximation, just less than half of the full scale voltage.
* comparator then compares this approximation with input voltage
If ~~output~~ input is higher, the comparator output is 1
If input is lower, the comparator output is 0.
* The comparator output is stored as the MSB in SAR, remaining bits are shifted down one place
* This gives the input to next approximation

6M

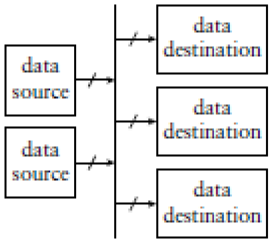


SAR based ADC

5 **Explain the parallel bus concept. Briefly discuss the solutions to overcome the disadvantages of parallel buses.** [10] CO4 L3

- Interconnect components in a system
  - Transfer bits of data in parallel
- Conceptual structure
  - All inputs and output connected

2M

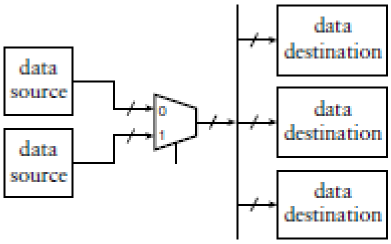


- Use multiplexer(s) to select among data sources
  - Can partition to aid placement on chip

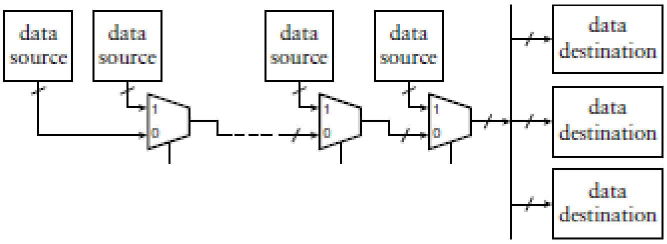
2M

Multiplexed buses

8.3 Parallel



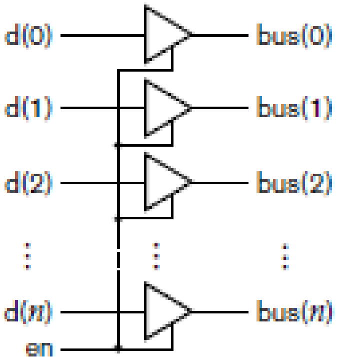
Distributed multiplexed bus



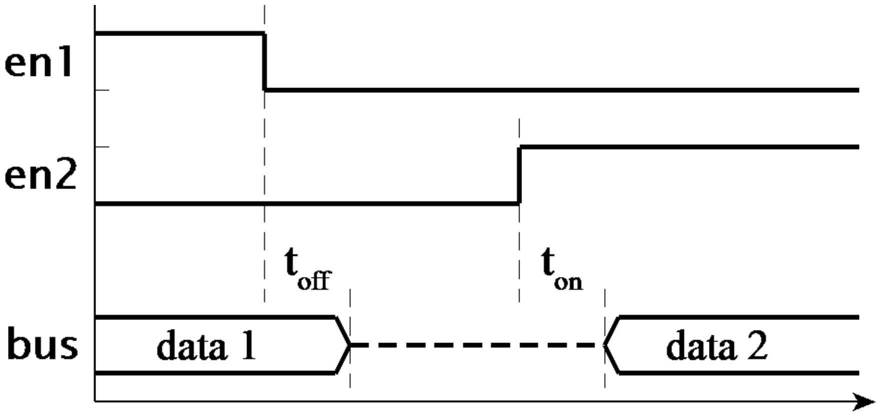
2M

- Use tristate drivers for data sources
  - Can “turn-off” (Hi-Z) when not supplying data
- Simplified bus wiring

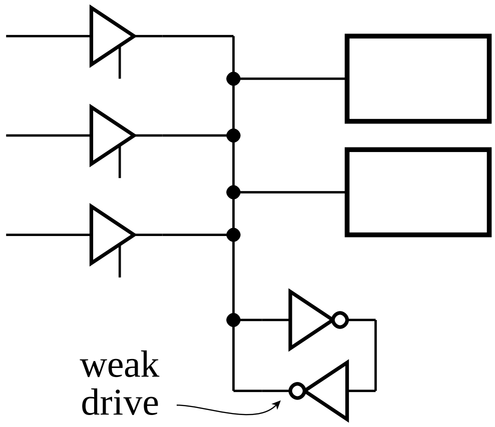
Tristate buses

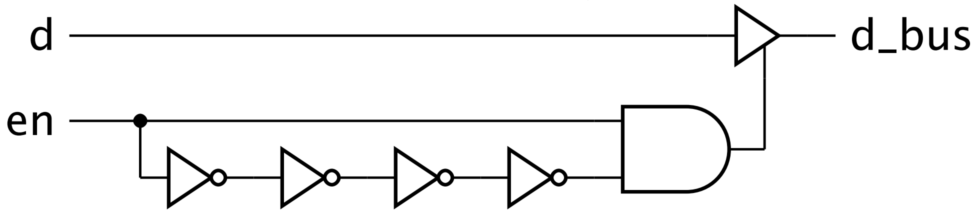


2M

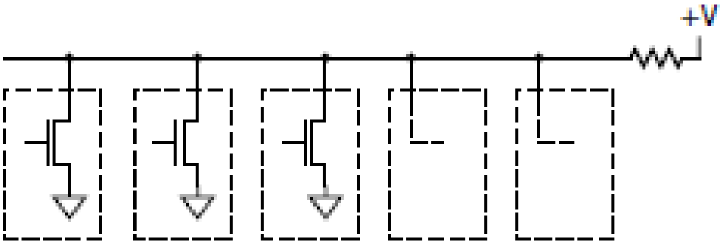


- Floating bus can cause spurious switching
  - Use pull-up resistors or weak keepers
- Need to avoid driver contention
  - Dead cycle between turn-off and turn-on
  - Or delayed enable





Open drain buses



2M

- Bus is 0 if any driver pulls it low
- If all drivers are off, bus is pulled high
  - Wired-AND
- Can also use open-collector drivers