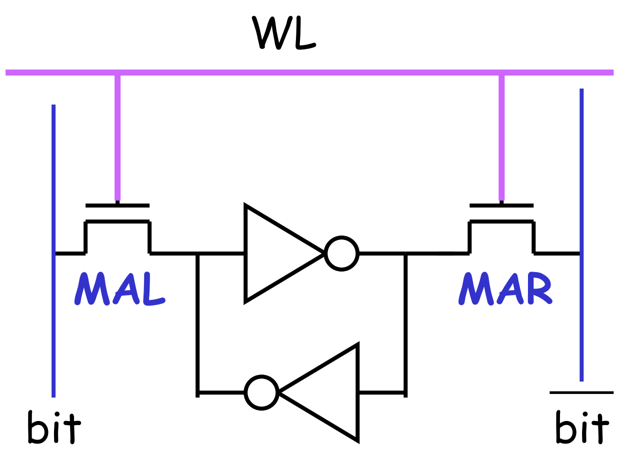
Memory Basics

- RAM: Random Access Memory
  - historically defined as memory array with individual bit access
  - refers to memory with both Read and Write capabilities
- ROM: Read Only Memory
  - no capabilities for “online” memory Write operations
  - Write typically requires high voltages or erasing by UV light
- Volatility of Memory
  - volatile memory loses data over time or when power is removed
    - RAM is volatile
  - non-volatile memory stores date even when power is removed
    - ROM is non-volatile
- Static vs. Dynamic Memory
  - Static: holds data as long as power is applied (SRAM)
  - Dynamic: must be refreshed periodically (DRAM)

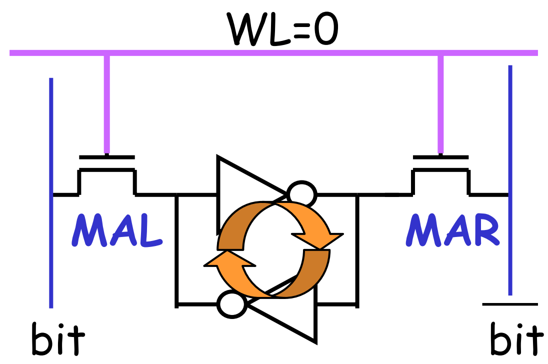
SRAM Basics

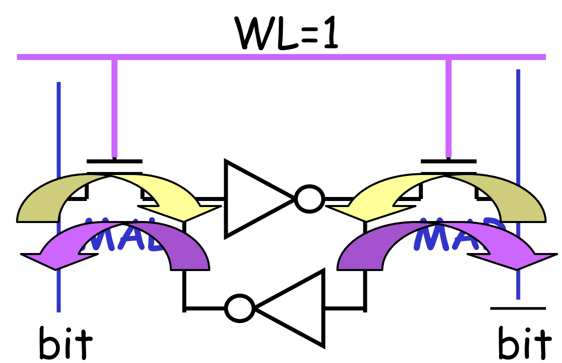
- SRAM = Static Random Access Memory
  - Static: holds data as long as power is applied
  - Volatile: can not hold data if power is removed
- 3 Operation States
  - hold
  - write
  - read
- Basic 6T (6 transistor) SRAM Cell
  - bistable (cross-coupled) INVs for storage
  - access transistors MAL & MAR
    - access to stored data for read and write
  - word line, WL, controls access
    - WL = 0, hold operation
    - WL = 1, read or write operation



SRAM Operations

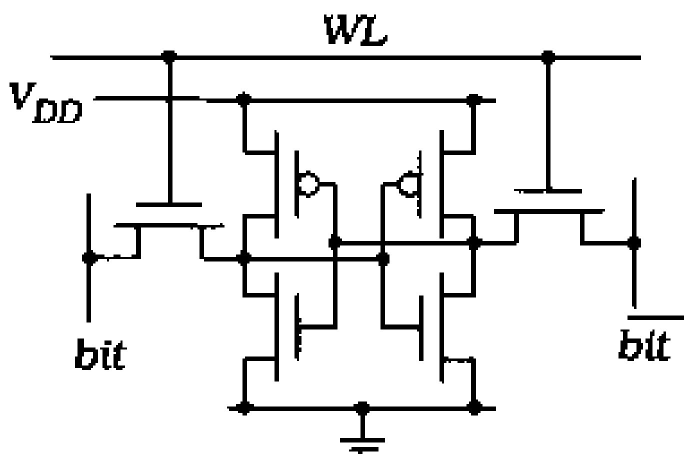
- Hold
  - word line = 0, access transistors are OFF
  - data held in latch
- Write
  - word line = 1, access tx are ON
  - new data (voltage) applied to bit and bit_bar
  - data in latch overwritten with new value
- Read
  - word line = 1, access tx are ON
  - bit and bit_bar read by a sense amplifier
- Sense Amplifier
  - basically a simple differential amplifier
  - comparing the difference between bit and bit_bar
    - if bit > bit_bar, output is 1
    - if bit < bit_bar, output is 0
    - allows output to be set quickly without fully charging/discharging bit line



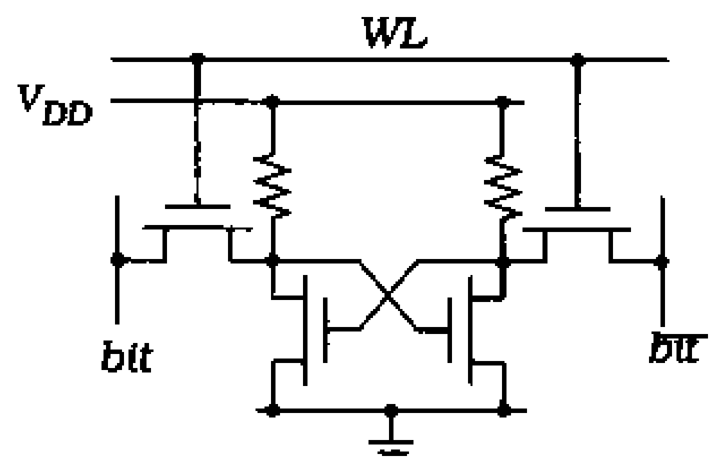


SRAM Bit Cell Circuit

- Two SRAM cells dominate CMOS industry
  - 6T Cell
    - all CMOS transistors
    - better noise immunity
  - 4T Cell
    - replaces pMOS with high resistance (~1GΩ) resistors
    - slightly smaller than 6T cell
    - requires an extra high-resistance process layer



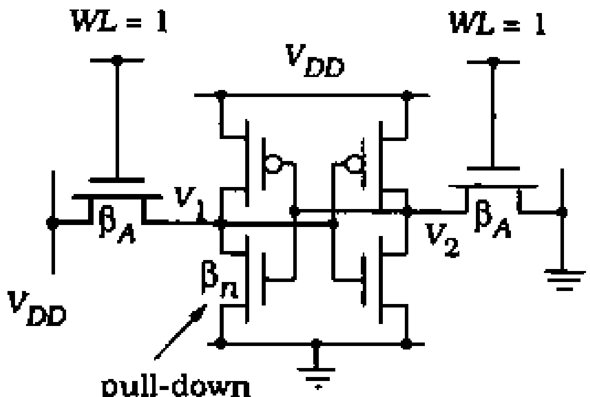
(a) 6T cell



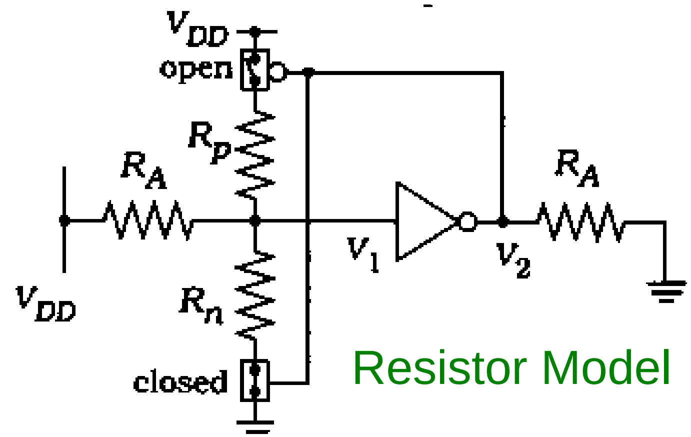
(b) 4T cell with poly resistors

6T Cell Design

- Critical Design Challenge
  - inverter sizing
    - to ensure good hold **and** easy/fast overwrite
  - use minimum sized transistors to save area
    - unless more robust design required
- Write Operation
  - both bit and bit_bar applied
    - inputs to inverters both change
    - unlike DFF where one INV overrides the other
  - critical size ratio, β_A/β_n
    - see resistor model
      - want R_n & R_p larger than R_A
        - » so voltage will drop across R_n, R_p
    - typical value, $\beta_A/\beta_n=2$
      - so $R_n = 2\ R_A$
      - set by ratio $(W/L)_A$ to $(W/L)_n$



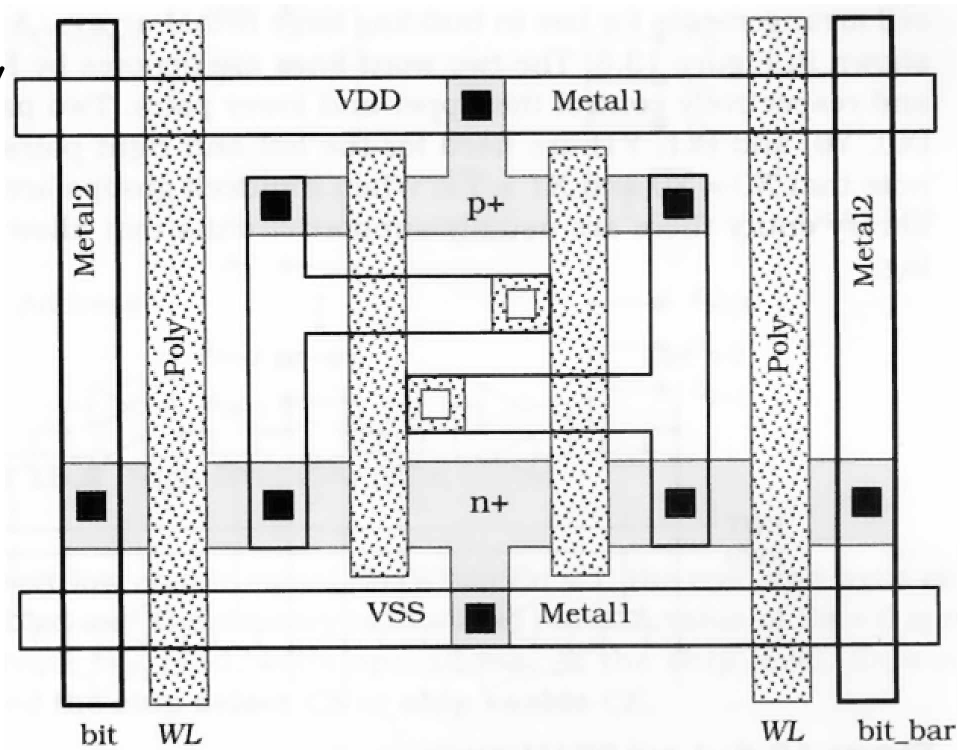
Write 1 Operation



Resistor Model

SRAM Cell Layout

- Design Challenge
  - minimum cell size (for high density SRAM array)
  - with good access to word and bit lines
- Example Layout
  - note WL routed in poly
    - will create a large RC delay for large SRAM array

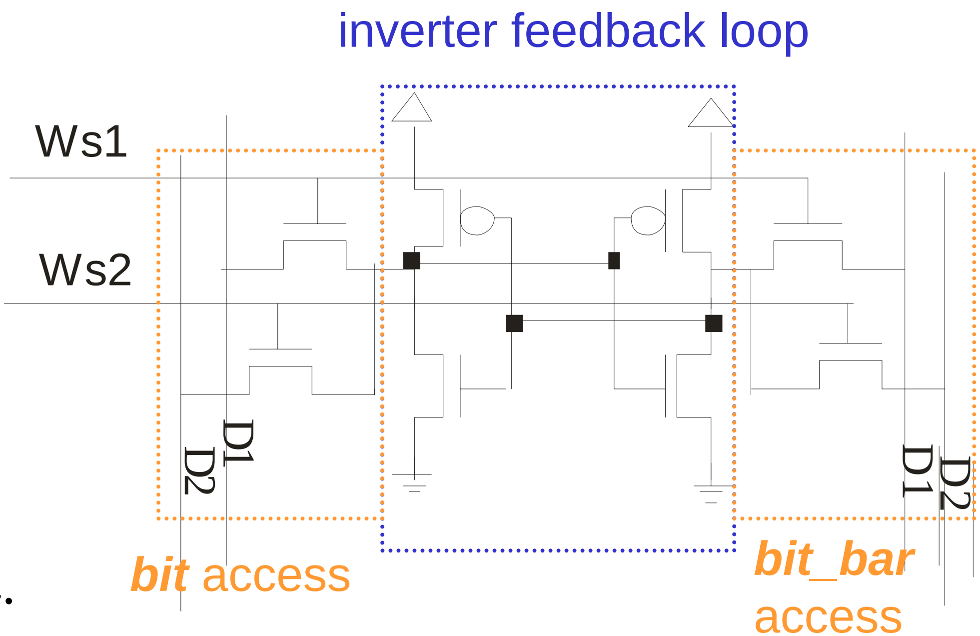


Multi-Port SRAM

- Allows multiple access to the same SRAM cell simultaneously.
  - Provide high data bandwidth.

- Applications
  - Register file
  - Cache
  - Network switch
  - ASIC etc.

- A multi-port SRAM cell schematic. Each port has
  - two access transistors
  - two bit line
  - one word selection line.
  - one address decoder



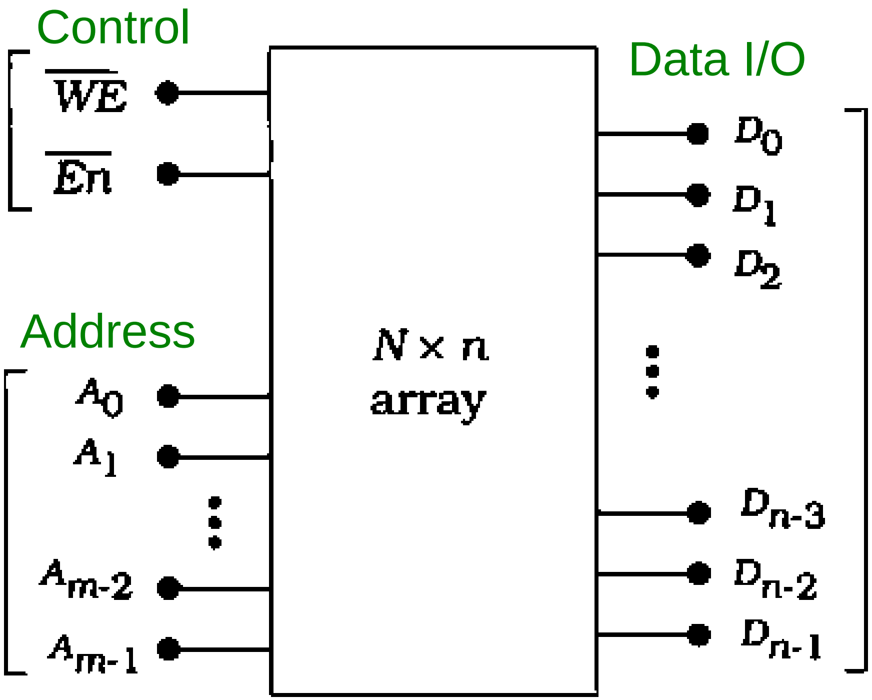
Multi-Port SRAM (cont.)

- Challenges in multi-ports SRAM.
  - layout size increases quadratically with # of ports
    - more word selection lines
    - more bitline lines
  - → lower speed and higher power consumption

- Multi-port SRAM options for ECE410 Design Project
  - Two ports
    - 1 port read and write
    - 1 port read only
  - Three ports
    - 2 ports for read and 1 port for write

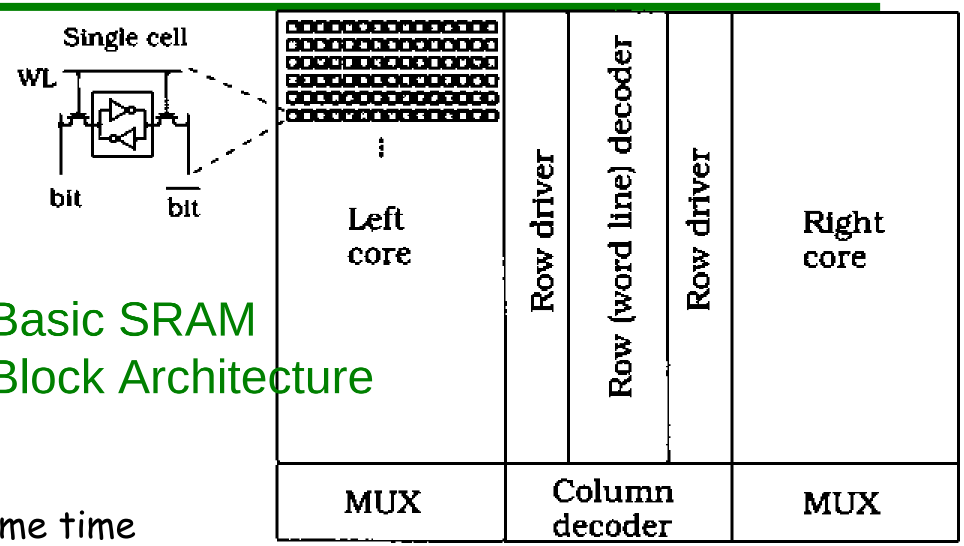
SRAM Arrays

- N x n array of 1-bit cells
  - n = byte width; 8, 16, 32, etc.
  - N = number of bytes
  - m = number of address bits
    - max N = 2^m
- Array I/O
  - data, in and out
    - Dn-1 - D0
  - address
    - Am-1 - A0
  - control
    - varies with design
    - WE = write enable (assert low)
      - WE=1=read, WE=0=write
    - En = block enable (assert low)
      - used as chip enable (CE) for an SRAM chip

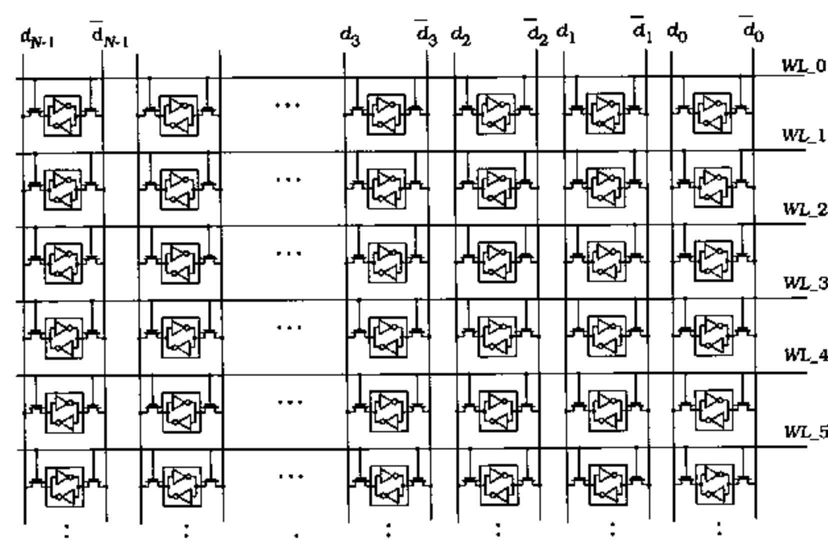


SRAM Block Architecture

- Example: 2-Core design
  - core width = k•n
    - n = SRAM word size; 8, 16, etc.
    - k = multiplier factor, 2,3,4,etc.
  - shared word-line circuits
    - horizontal word lines
    - WL set by **row decoder**
      - placed in center of 2 cores
      - WL in both cores selected at same time
- Addressing Operation
  - address word determines which row is active (which WL =1) via **row decoder**
  - row decoder outputs feed **row drivers**
    - buffers to drive large WL capacitance
- Physical Design
  - layout scheme matches regular patterning shown in schematic
    - horizontal and vertical routing



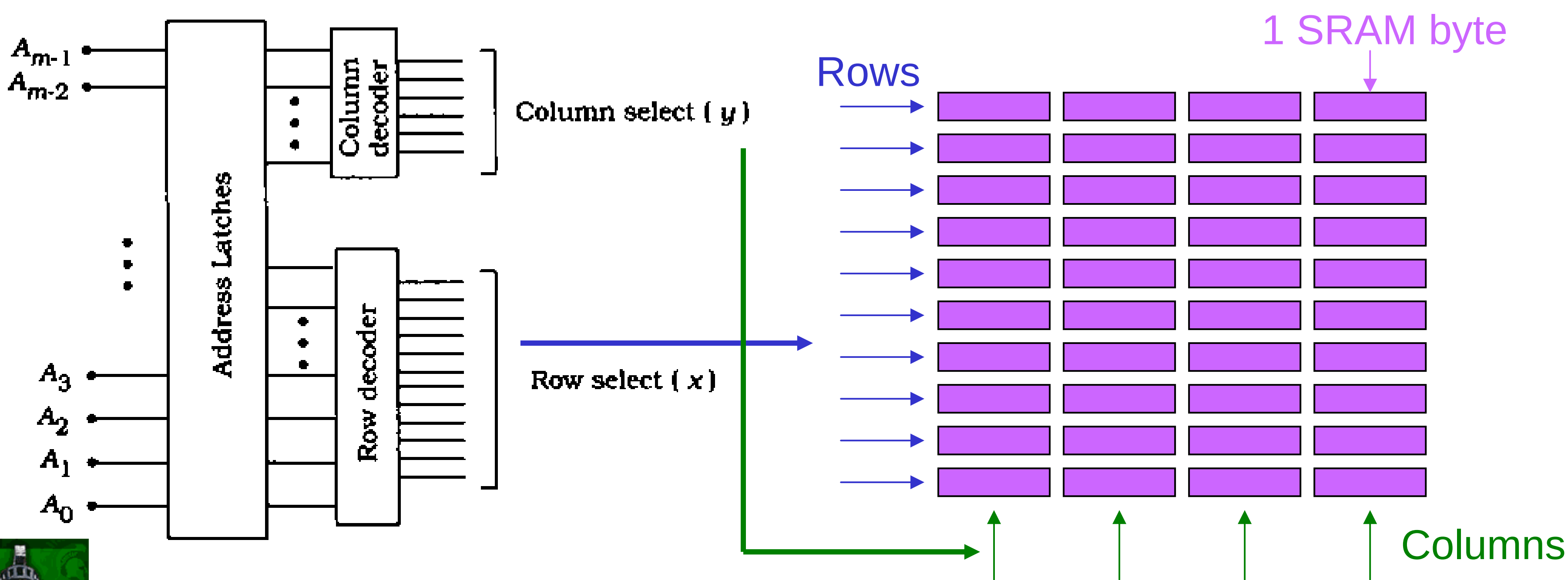
Basic SRAM Block Architecture



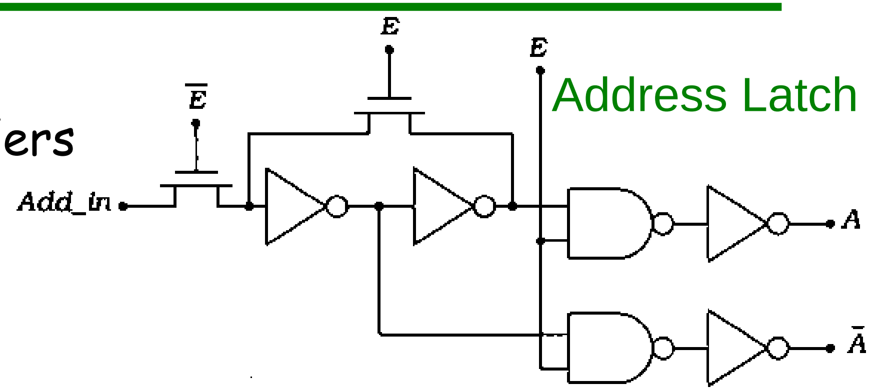
Expanded Core View

SRAM Array Addressing

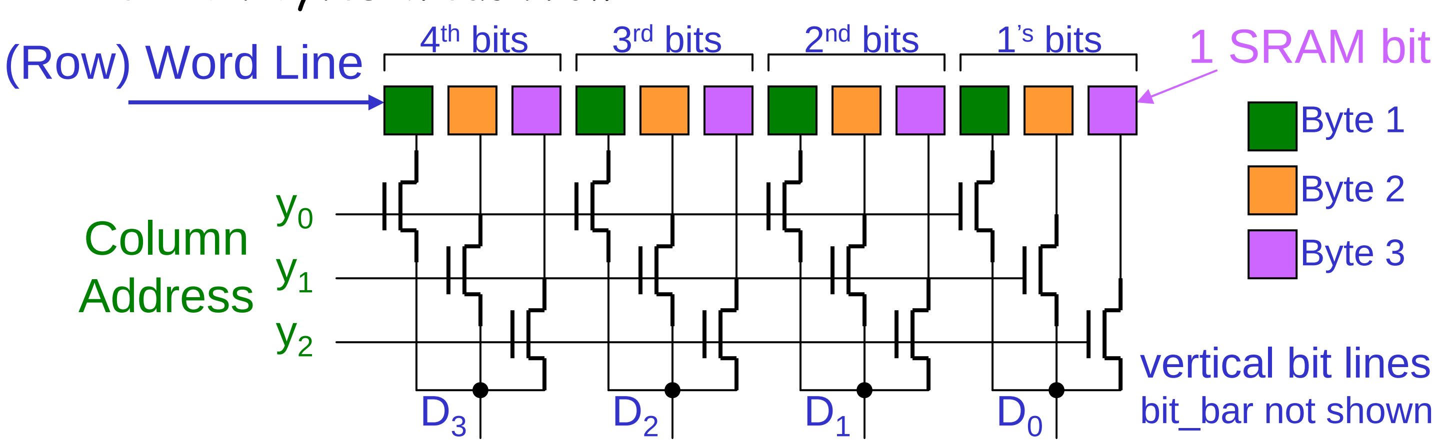
- Standard SRAM Addressing Scheme
  - consider a ***N*** x ***n*** SRAM array
    - N = number of bytes, e.g., 512, 2k
    - n = byte size, e.g., 8 or 16
  - ***m*** address bits are divided into ***x*** row bits and ***y*** column bits (x+y=m)
    - address bits are encoded so that $2^m = N$
    - array organized with both both vertical and horizontal stacks of bytes



SRAM Array Addressing

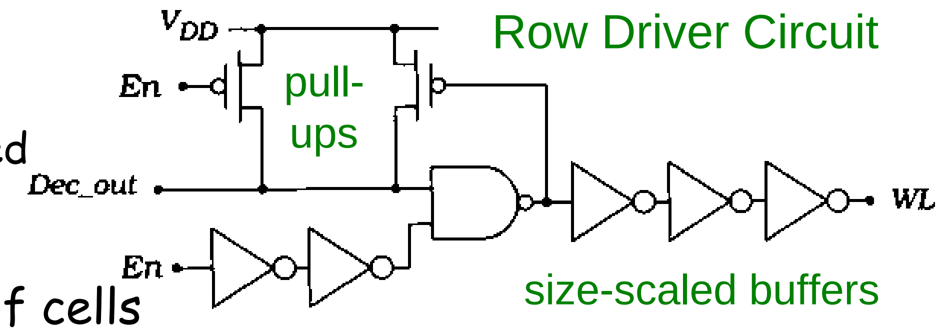


- Address Latch
  - D-latch with enable and output buffers
  - outputs both A and A_bar
- Address Bits
  - Row address bits = Word Lines, WL
  - Column address bits select a subset of bits activated by WL
- Column Organization
  - typically, organized physically by bits, not by bytes
  - Example, SRAM with 4-bit bytes in 3 columns (y=3)
    - 3 4-bit bytes in each row

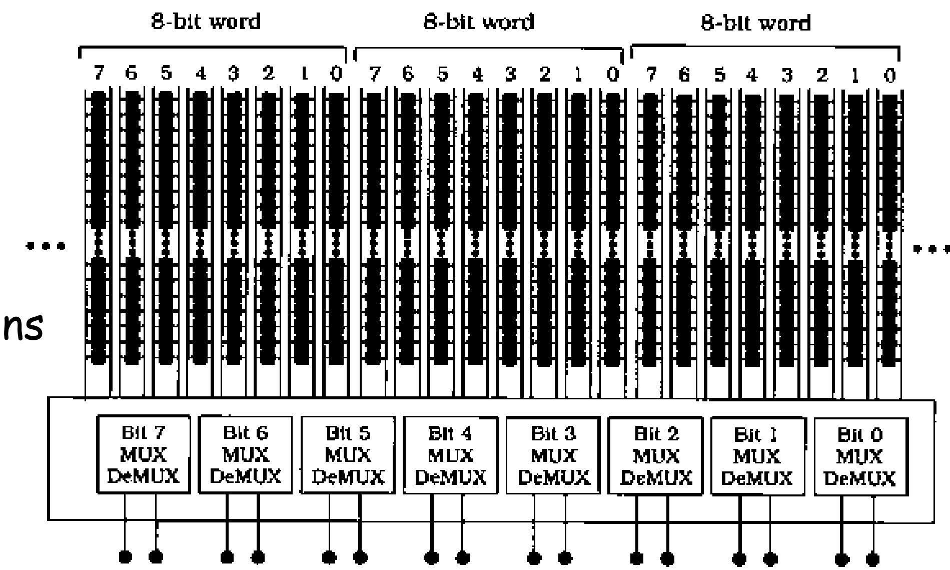


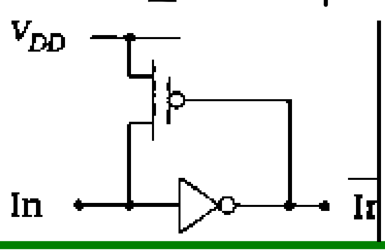
SRAM Array Column Circuits

- SRAM Row Driver
  - decoder output, Dec_out
  - enable, En, after address bits decoded



- Row Decoder/Driver activate a <u>row</u> of cells
  - each 2-core row contains 2k bytes (2k•n bits)
- Column Multiplexers
  - address signals select one of the k bytes as final output not used in row decoder
  - figure shows example for k=3
    - for an 8-bit RAM (word size)
  - MUX used for Read operations
  - De MUX used for Write op.s
- Column Drivers
  - bit/bit_bar output for Write operations

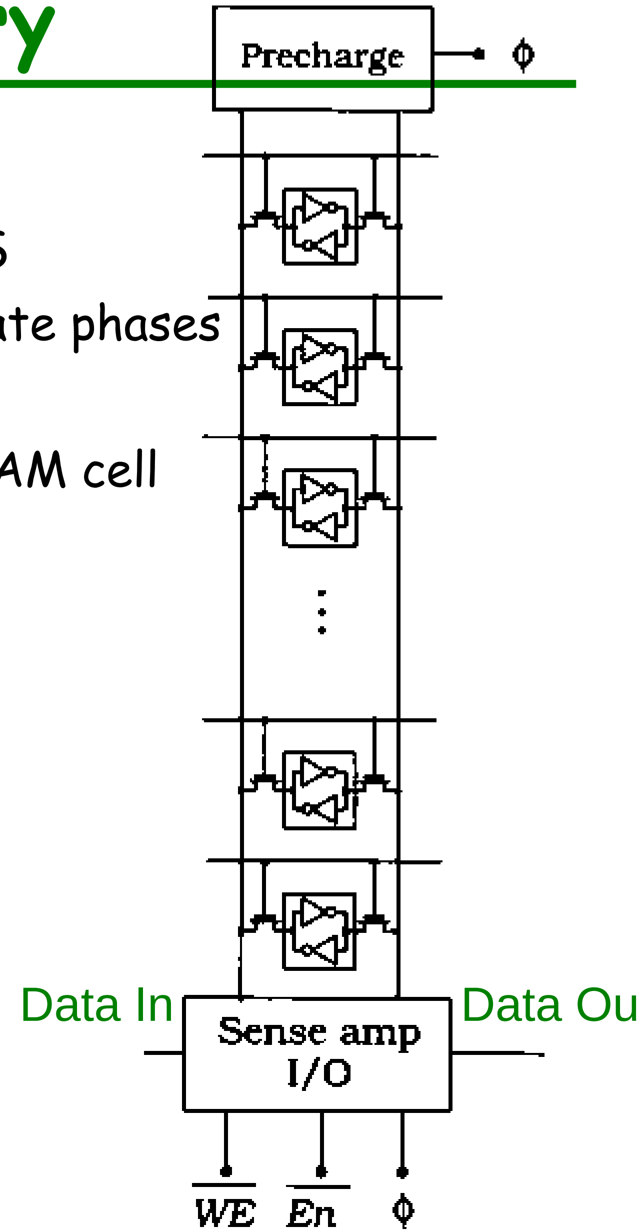




Column Driver Circuit

Column Circuitry

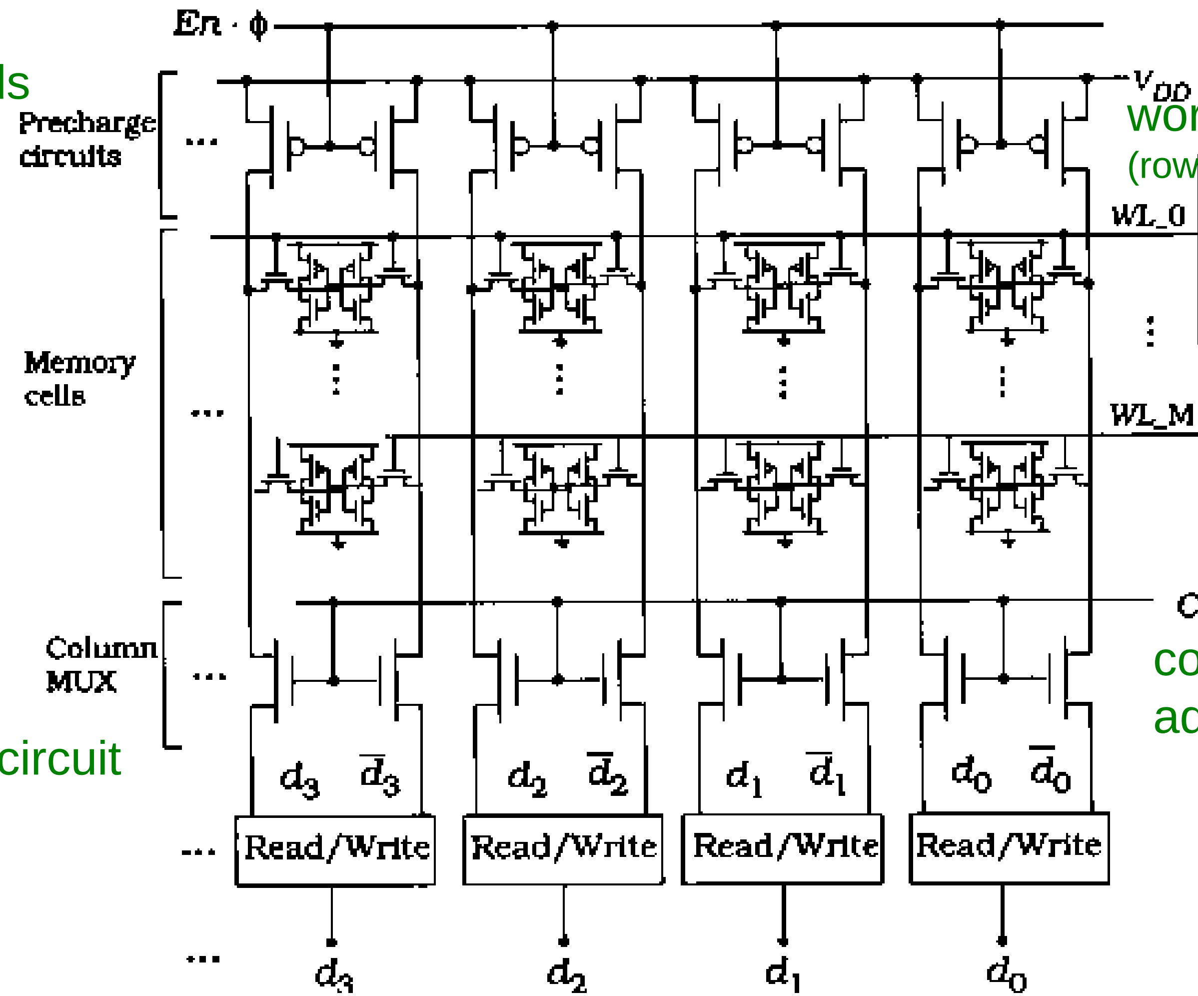
- Precharge Concept
  - common to use dynamic circuits in SRAMS
    - dynamic circuits have precharge and evaluate phases
  - precharge high capacitance on bit lines
    - avoids heavy capacitive loading on each SRAM cell
- Precharge Phase
  - all bit lines pulled to VDD
  - all bit_bar to ground
- Evaluate Phase
  - bits activated by WL connect to bit lines
    - if data = 1, keep precharged value
    - if data = 0, discharge bit line



Bit line (column) Circuitry

- expanded (transistor-level) view of SRAM column

pMOS precharge loads
- charge when $\phi = 0$



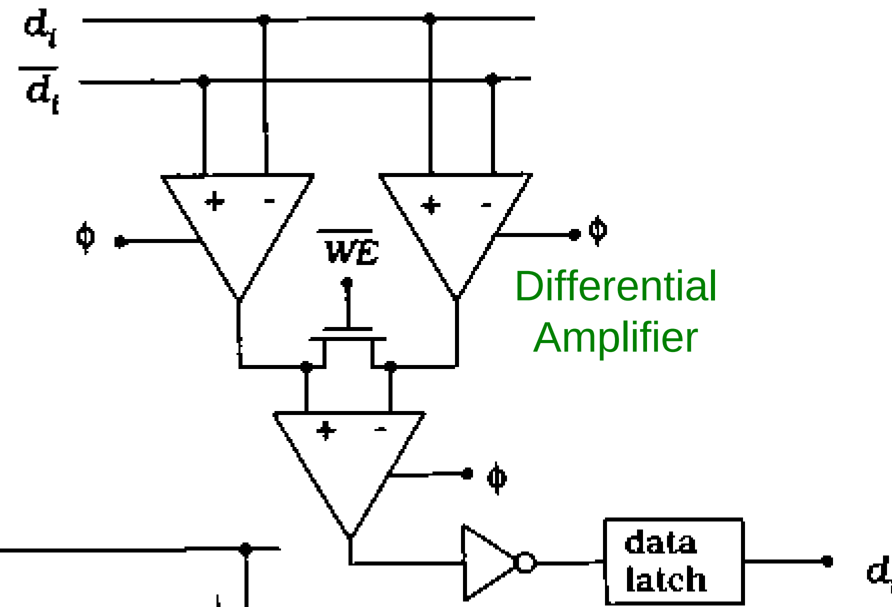
word lines
(row address)

nMOS switches select which column/bit is passed to Read/Write circuit

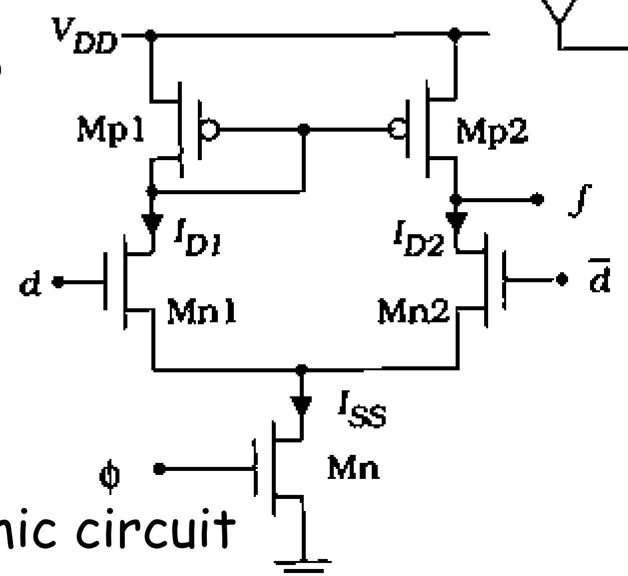
column address

Sense Amplifiers

- Read sensing scheme
  - look at differential signal
    - bit and bit_bar
  - can get output before bit lines fully charge/discharge by amplifying differential signals

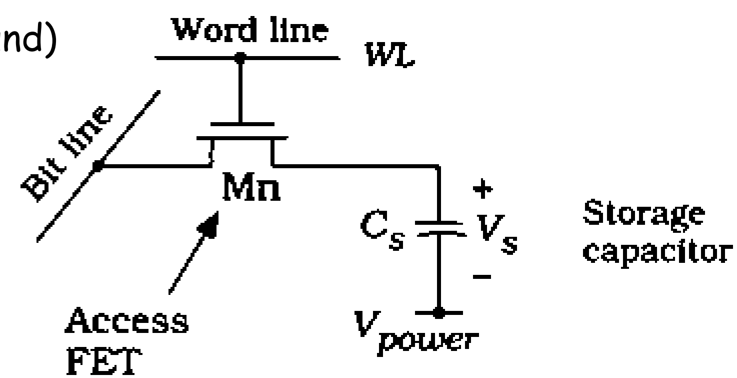


- Differential Amplifier
  - simple analog circuit
  - output high
    - if bit > bit_bar
  - output low
    - if bit_bar > bit
  - can implement as dynamic circuit



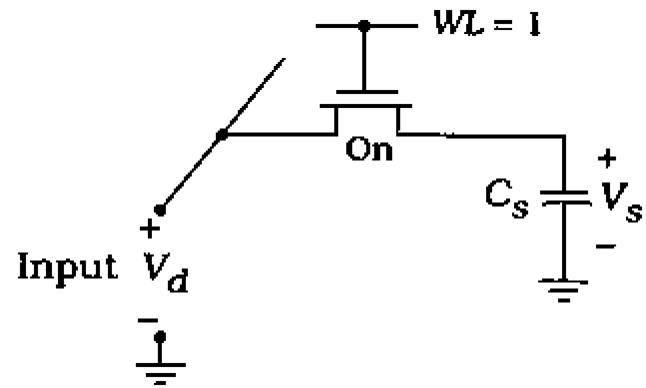
DRAM Basics

- DRAM = Dynamic Random Access Memory
  - Dynamic: must be refreshed periodically
  - Volatile: loses data when power is removed
- Comparison to SRAM
  - DRAM is smaller & less expensive per bit
  - SRAM is faster
  - DRAM requires more peripheral circuitry
- 1T DRAM Cell
  - single access nFET
  - storage capacitor (referenced to VDD or Ground)
  - control input: word line, WL
  - data I/O: bit line

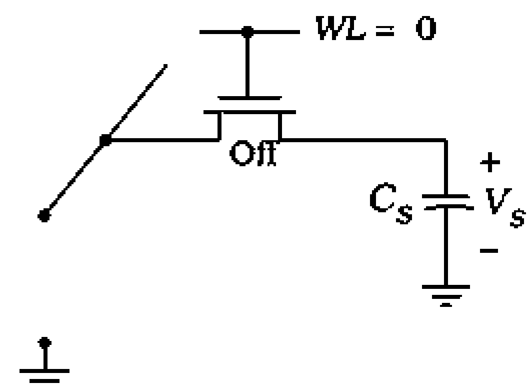


DRAM Operation

- RAM data is held on the storage capacitor
  - temporary –due to leakage currents which drain charge
- Charge Storage
  - if Cs is charged to Vs
  - Qs = Cs Vs
    - if Vs = 0, then Qs = 0: LOGIC 0
    - if Vs = large, then Qs > 0: LOGIC 1
- Write Operation
  - turn on access transistor: WL = VDD
  - apply voltage, Vd (high or low), to bit line
  - Cs is charged (or discharged)
  - if Vd = 0
    - Vs = 0, Qs = 0, store logic 0
  - if Vd = VDD
    - Vs = VDD-Vtn, Qs = Cs(VDD=Vtn), logic 1
- Hold Operation
  - turn off access transistor: WL = 0
    - charge held on Cs



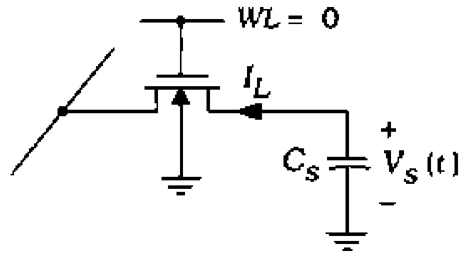
(a) Write operation

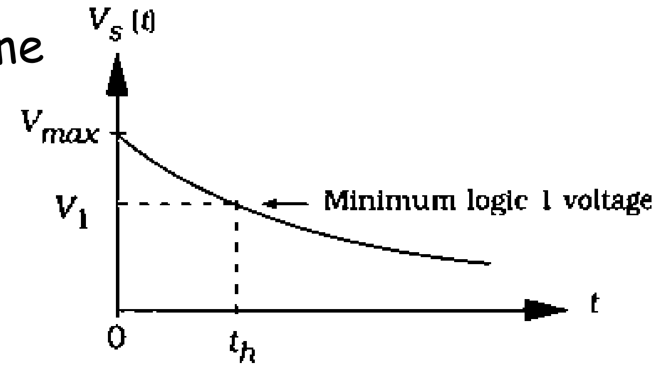


(b) Hold

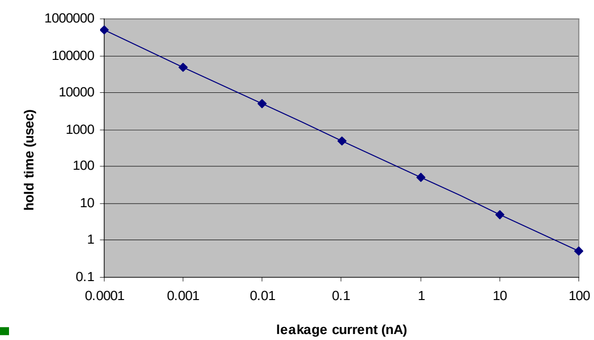
Hold Time

- During Hold, leakage currents will slowly discharge Cs
  - due to leakage in the access transistor when it is OFF
  - $I_L = -\delta Qs/\delta t = -Cs\ \delta Vs/\delta t$
    - if I_L is known, can determine discharge time





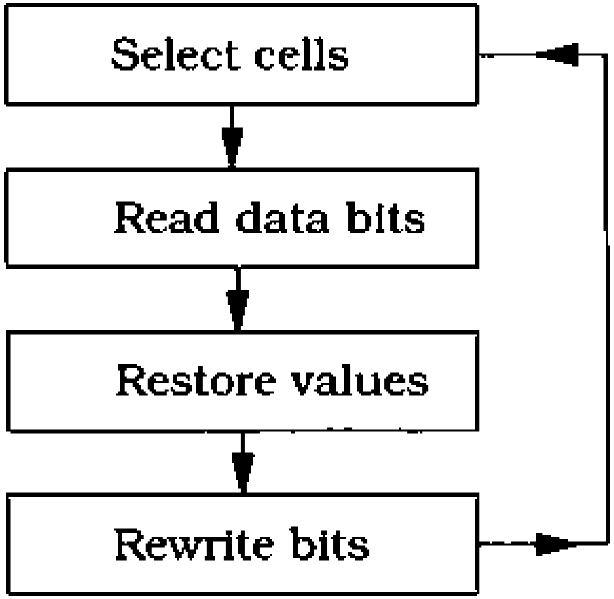
- Hold Time, t_h
  - max time voltage on Cs is high enough to be a logic 1
    - = time to discharge from Vmax to V1 (in figure above)
  - $t_h = (Cs/I_L)(\Delta Vs)$, if we estimate I_L as a constant
    - desire large hold time
    - t_h increases with larger Cs and lower I_L
    - typical value, $t_h = 50\mu sec$
      - with I_L = 1nA, Cs=50fF, and ΔVs=1V



error in textbook, says 0.5µsec near Eqn. 13.

Refresh Rate

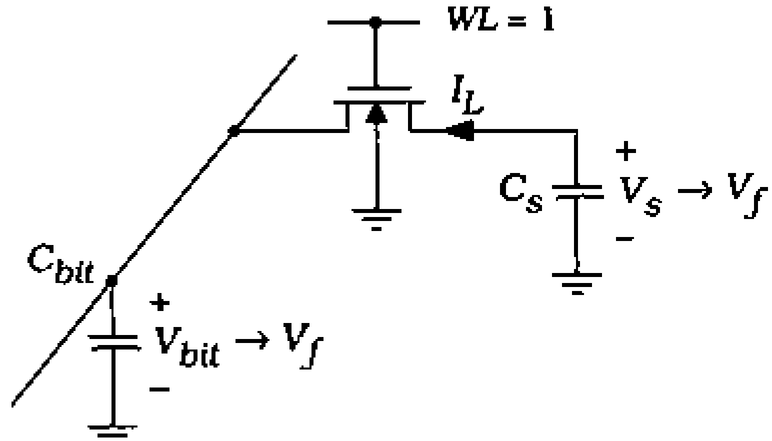
- DRAM is "Dynamic", data is stored for only short time
- Refresh Operation
  - to hold data as long as power is applied, data must be refreshed
  - periodically read every cell
    - amplify cell data
    - rewrite data to cell
- Refresh Rate, $f_{refresh}$
  - frequency at which cells must be refreshed to maintain data
  - $f_{refresh} = 1 / 2t_h$
  - must include refresh circuitry in a DRAM circuit



Refresh operation

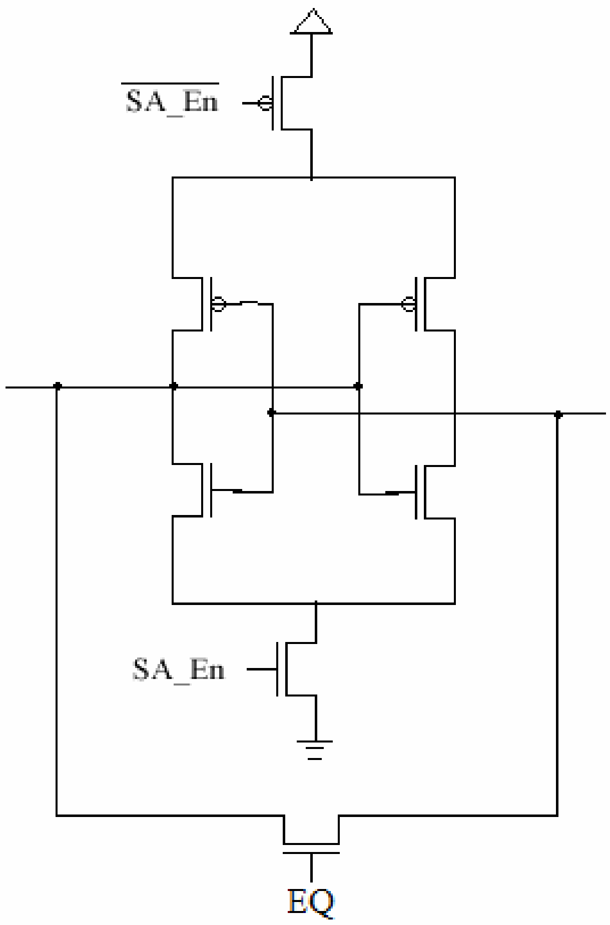
DRAM Read Operation

- Read Operation
  - turn on access transistor
  - charge on Cs is redistributed on the bit line capacitance, Cbit
  - this will change the bit line voltage, Vbit
  - which is amplified to read a 1 or 0
- Charge Redistribution
  - initial charge on Cs: Qs = Cs Vs
  - redistributed on Cbit until
    - Vbit = Vs = Vf (final voltage)
  - Qs = Cs Vf + Cbit Vf
  - Cs Vs = Vf (Cs + Cbit)
    - due to charge conservation
  - Vf = Cs Vs / (Cs + Cbit), which is always less than Vs
    - Vf typically very small and requires a good sense amplifier



DRAM Read Operation (cont.)

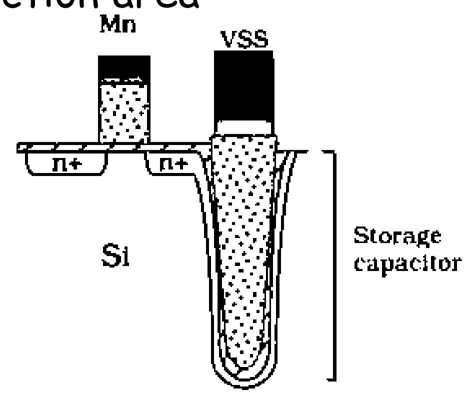
- DRAM Read Operation is Destructive
  - charge redistribution destroys the stored information
  - read operation must contain a simultaneous rewrite
- Sense Amplifier
  - SA_En is the enable for the sense amplifier
  - when EQ is high both sides of the sense amp are shorted together. The circuit then holds at it's midpoint voltage creating a precharge.
  - the input and output of the sense amp share the same node which allows for a simultaneous rewrite



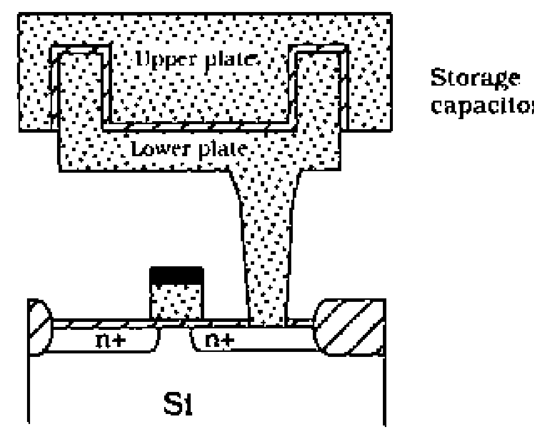
http://jas.eng.buffalo.edu/education/system/senseamp/

DRAM Physical Design

- Physical design (layout) is CRITICAL in DRAM
  - high density is required for commercial success
  - current technology provides > 1Gb on a DRAM chip
- Must minimize area of the 1T DRAM cell
  - typically only 30% of the chip is needed for peripherals (refresh, etc.)
- For DRAM in CMOS, must minimize area of storage capacitor
  - but, large capacitor (> 40fF) is good to increase hold time, t_h
- Storage Capacitor Examples
  - trench capacitor
    - junction cap. with large junction area
    - using etched pit
  - stacked capacitor
    - cap. on top of access tx
    - using poly plate capacitor

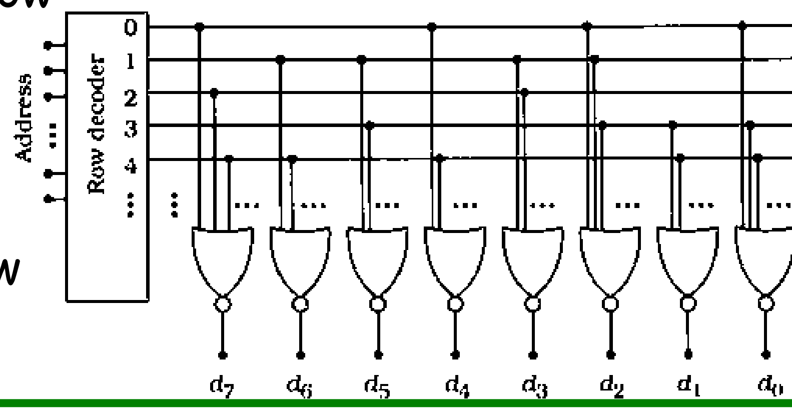


A DRAM cell using a trench capacitor



ROM Basics

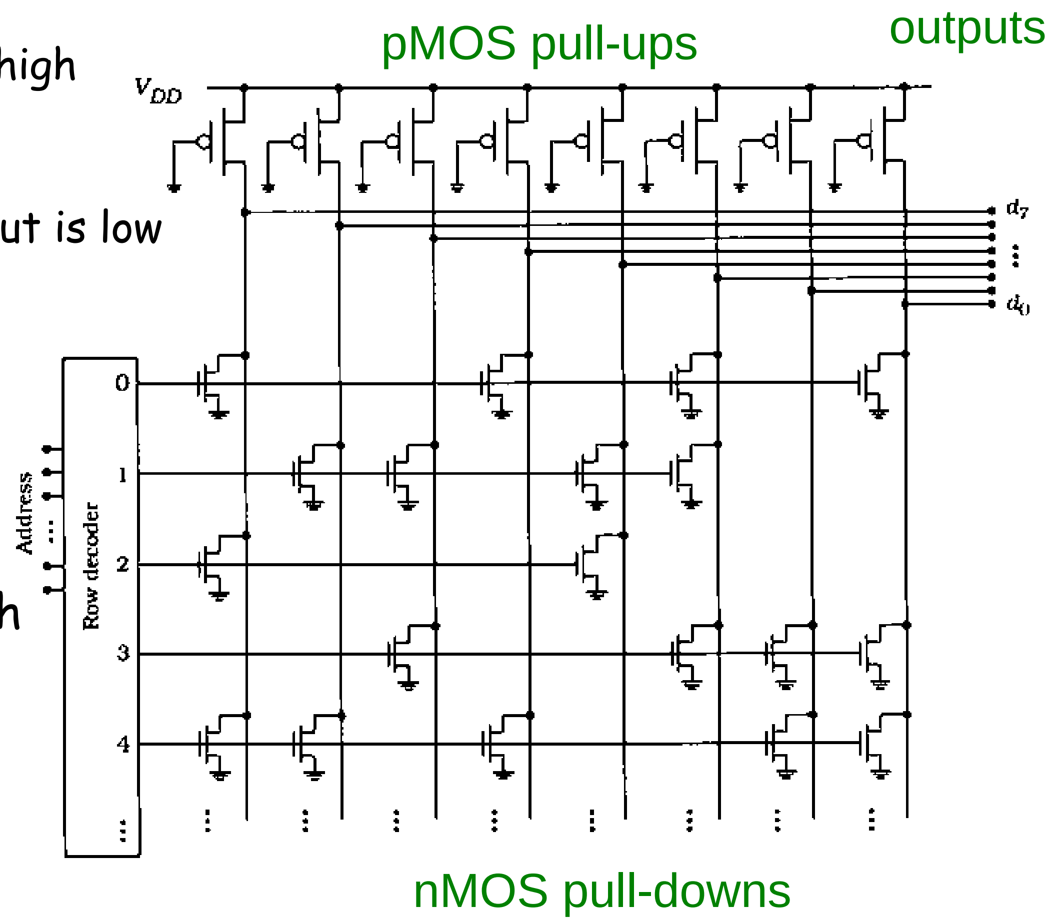
- ROM: Read Only Memory
  - no capabilities for “online” memory Write operations
  - data programmed
    - during fabrication: ROM
    - with high voltages: PROM
    - by control logic: PLA
  - Non-volatile: data stored even when power is removed
- NOR-based ROM
  - Example: 8b words stored by NOR-based ROM
  - address selects an active ‘row’
  - each output bit connected to the active row will be high
  - otherwise, output will be low



Row	Data
0	01101010
1	10010011
2	01110111
3	11011000
4	00101100

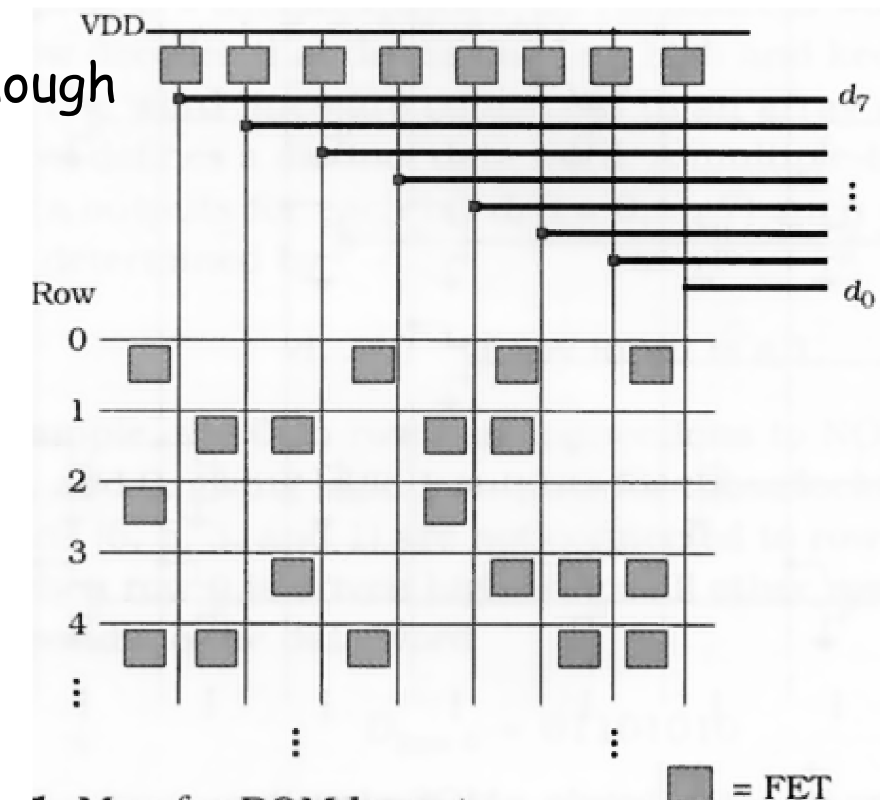
Pseudo-nMOS ROM

- Pseudo-nMOS
  - always ON active pMOS load
    - pulls output high if nMOS is off
  - controlled nMOS switch
    - pulls output low if input is high
    - competes with pMOS
      - must be sized properly
    - consumes power when output is low
- ROM Structure
  - address is decoded to choose and active 'row'
  - each row line turns on nMOS where output is zero
  - otherwise, output stays high
- Set ROM Data
  - by selectively connecting nMOS to the output lines



ROM Arrays

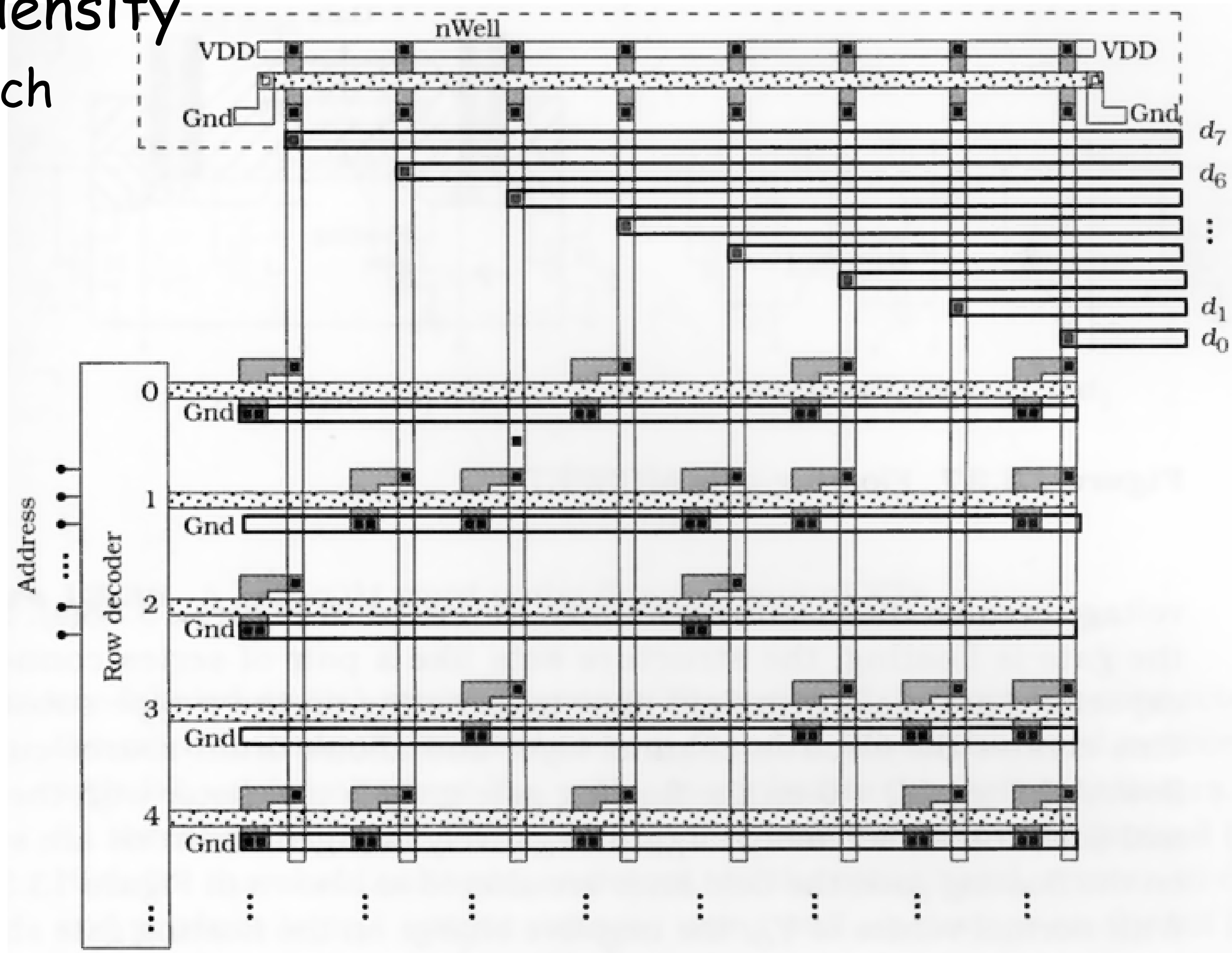
- Pseudo nMOS Arrays
  - most common style for large ROMS
- Design Concerns
  - nMOS must "overdrive" pMOS
  - need $\beta_n > \beta_p$ so that V_{OL} is low enough
    - must set Wn > Wp
    - but, this also increases row line capacitance
  - requires careful analog design
- Programming Methods
  - mask programmable
    - create nMOS at all points
    - define data with poly contacts
  - layout programmable
    - only place nMOS where needed
    - shown in figure →



5 Map for ROM layout

ROM Array Layout

- very "regular" layout
- high packing density
  - one tx for each data point

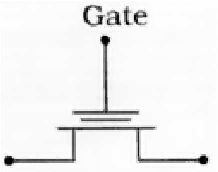


Programmable ROM

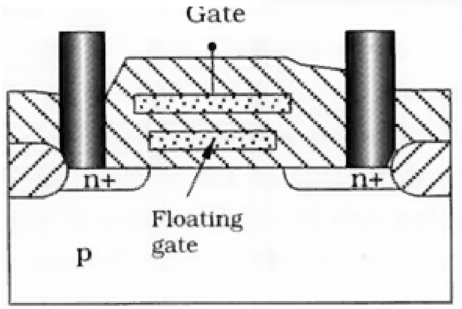
- PROM
  - programmable by user
    - using special program tools/modes
  - read only memory
    - during normal use
  - non-volatile
- Read Operation
  - like any ROM: address bits select output bit combinations
- Write Operation
  - typically requires high voltage (~15V) control inputs to set data
- Erase Operation
  - to change data
  - EPROM: erasable PROM: uses UV light to reset all bits
  - EEPROM: electrically-erasable PROM, erase with control voltage

PROM Storage Cells

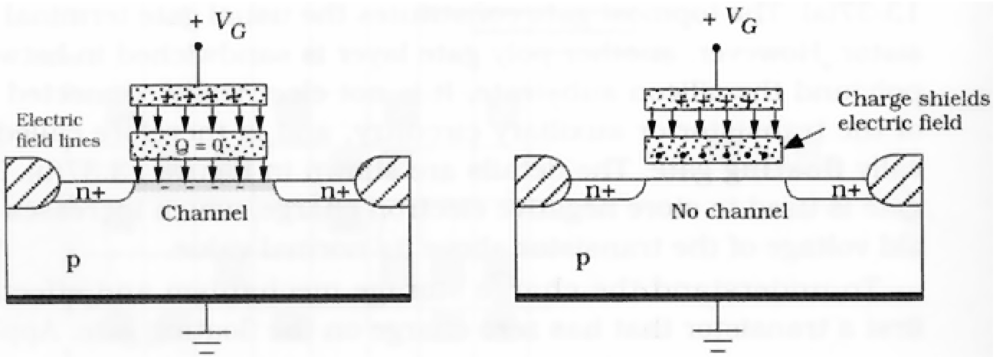
- Physical Structure
  - pair of stacked poly gates
    - top gate acts as normal access/control gate
    - bottom gate is 'floating', changes threshold voltage
- Cell Operation
  - no charge on floating gate
    - transistor has normal Vtn
  - negative charge on floating gate
    - opposes action of applied gate voltage
    - keeps transistor turned off
      - unless a high Vtn_H is applied; Vtn_H > VDD so will not turn on with normal voltages



(a) Symbol



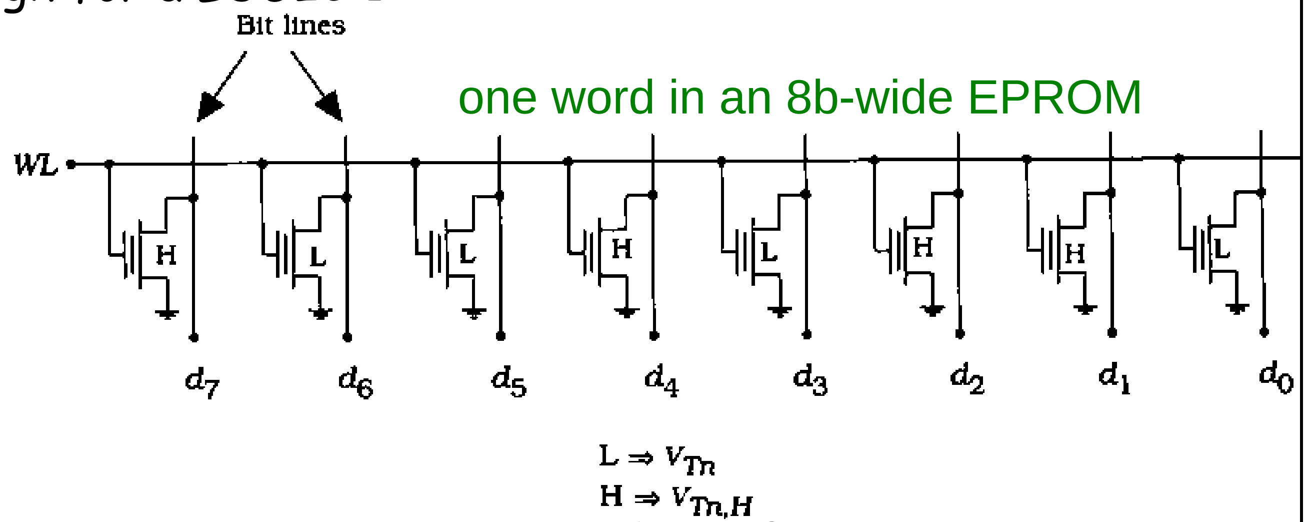
(b) Structure



(a) Normal V_{Tn} state (b) Increased $V_{Tn,H}$ state

EPROM Arrays

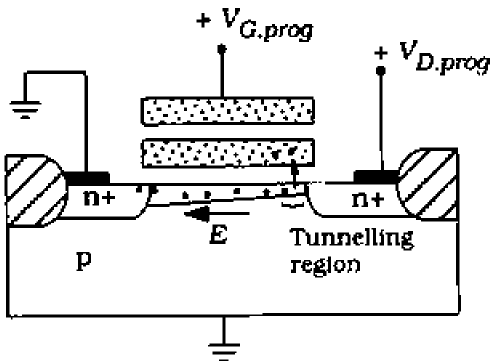
- Structure is similar to a RAM Array
- WL selects which word of data will connect to output
- When WL is high
  - each tx in the selected data byte will set the output bit line
    - if floating gate has no charge, bit line will pull down for a LOGIC 0
    - if floating gate is charged, tx will not turn on and bit line will remain high for a LOGIC 1



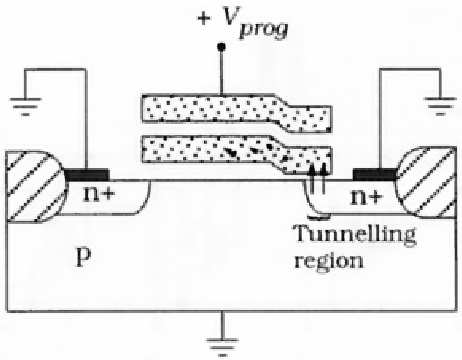
- Column circuitry can be used to form arrays, as in RAM

Programming & Erasing E²PROMs

- Programming techniques
  - hot electron method
    - charge (electrons) transferred to the floating gate by quantum mechanical tunneling of hot electrons (high energy electrons)
    - accomplished by applying a high voltage (~12-30V) to the drain node
    - charge can remain on floating gate for 10-20 years!



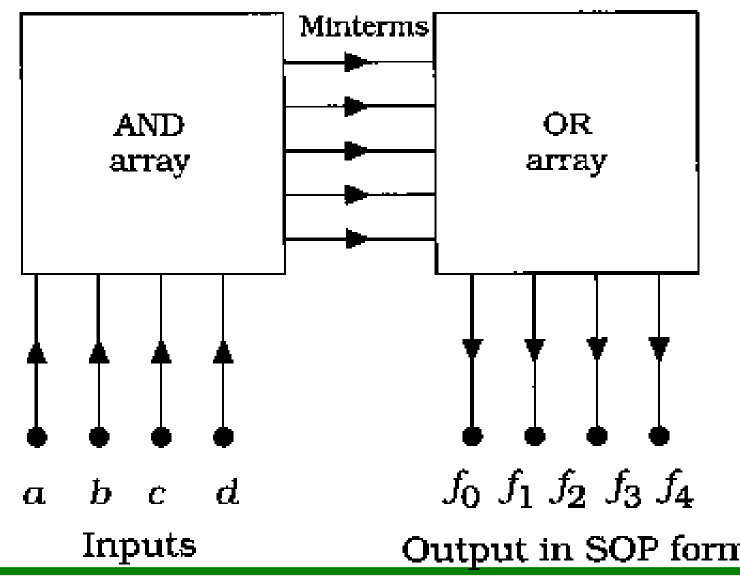
  - Fowler-Nordheim emission
    - uses modified gate geometry to allow quantum mechanical tunneling from the drain into the floating gate



- Erasure Techniques
  - bit erasure: by reversing programming voltages
  - flash EPROMs: erase large block simultaneously

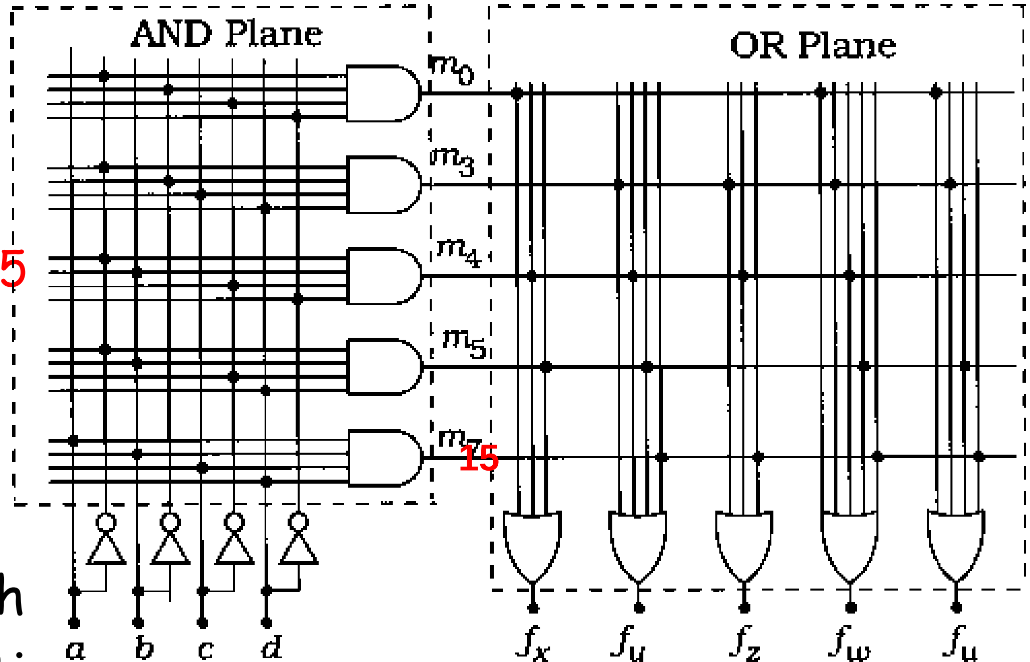
Programmable Logic Arrays

- Programmable Logic Array: PLA
  - circuit which can be programmed to provide various logic functions
- Example: Sum-of-Products PLA
  - with four inputs (a, b, c, d), the possible SOP outputs are
    - $f = \Sigma\, m_i(a,b,c,d)$ [OR minterms]
    - where $m_i(a,b,c,d)$ are the minterms [AND inputs]
  - has an AND-OR structure which can be reproduced in circuits



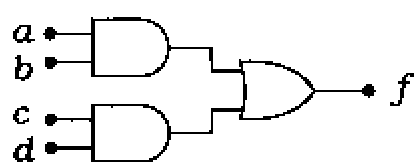
AND-OR PLA Implementation

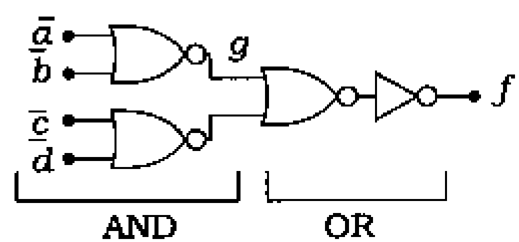
- Logic Array Diagram
  - example for
    - fx = m0 + m4 + m5
    - fy = m3 + m4 + m5 + m15
    - etc.



error in text

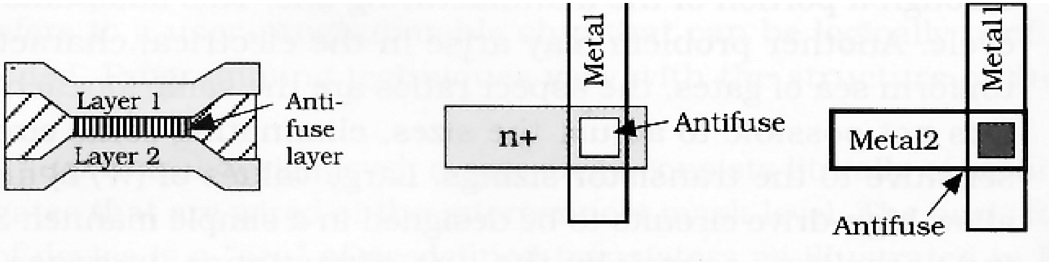
- Programming PLA
  - transistor switch at each optional connection location
  - turn tx on to make connection
- VLSI Implementation
  - replace AND-OR with NOR gates





Gate Arrays

- Gate array chip contains
  - a huge array of logic gates
  - programmable connections
    - allows gates to be combined to make larger functions (e.g., DFF)
- Field Programmable Gate Array (FGPA)
  - connections can be programmed easily to redefine function
  - can have more than 100,000 logic gates on an FPGA
    - capable of emulating complex functions, like a 32-bit microprocessor
  - program techniques: the antifuse concept
    - physical design: built-in fuses where connections might be wanted
    - high current short-circuits the fuse to create low resistance path



(a) Structure (b) Metal-active (c) Metal-metal