Part IA Digital Electronics Notes

1 K-Map

- Plot all minterms in a grid, indexed using grey code
- Group terms:
  - Having a size equal to a power of 2
  - With groups as large as possible
  - Groups can wrap around corners and edges
- To find the product of sums form, group the zero terms and apply DeMorgan's law
- Plot don't cares as an X and these can be in or outside of a group
- **Cover** - A term covers a minterm when that minterm is part of that term
- **Prime Implicant** - A term that can not be further combined
- **Essential Prime Implicant** - A prime implicant that covers a minterm that no other prime implicant covers
- **Covering Set** - A minimum set of prime implicants which includes all essential terms plus any other prime implicants required to cover all minterms

2 Q-M Method

- List all minterms and don't cares as binary numbers, group based on number of ones
- Compare elements in adjacent groups, if they differ by just one then put into second column, with differing bit blanked out
- Repeat, labelling and unpaired elements with an asterisk
- Make a prime implication chart
  - Label columns with minterms (exclude don't cares)
  - Label rows with those elements with asterisks, and the numbers they could represent
  - Put an X in the locations when there is a cross over, eg. 4 on row and column
- Look for column with only one X in, cross out any minterms in the same row as this lone X
- Find as few primes as possible to cover the remaining crosses

3 Combinational Logic

- Combinational logic circuits have no memory
- An n-level combinational logic circuit has two gates between input and output

3.1 Boolean Algebra

- Commutation
  - $a + b = b + a$
  - $a.b = b.a$
- Association
  - $(a + b) + c = a + (b + c)$
  - $(a.b).c = a.(b.c)$
- Distribution
  - $a.(b + c + ..) = a.b + a.c + ...$
  - $a + (b.c....) = (a + b).(a + c)....$
- Absorption
  - $a + a.c = a$
  - $a.(a + c) = a$
- DeMorgran's Law
  - $\overline{a + b + c + ...} = \overline{a}.\overline{b}.\overline{c}....$
  - $\overline{a.b.c....} = \overline{a} + \overline{b} + \overline{c} + ...$

3.2 Normal Forms

- Disjunctive Normal Form
  - A boolean function expressed as ORing of its minterms
  - Said to be in sum of products form
- Conjunctive Normal Form
  - Maxterms are expressions that are only false for one row of a truth table
  - The conjunctive normal form is the ANDing of an expressions maxterms
  - Said to be in product of sums form

3.3 Hazards

- It takes time for a gate's output to change in response to a change of inputs, this is gate propogation delay
- **Static Hazard** - The output undergoes a momentary transition when one input changes, when it is supposed to remain unchanged
  - Solved by adding another term in the K-map, covering overlapping the essential terms
  - Do this on complimented k-map for a 0 hazard
- **Dynamic Hazard** - The output changes more than once when it is supposed to change just one

4 Adders

4.1 Half Adder

- Adds together two, single bit binary numbers a and b, has the following equations

$$sum = \overline{a}.b + a.\overline{b}$$

$$c_{out} = a.b$$

4.2 Full Adder

- Adds together two single bit binary numbers with a carry bit, has the following equations

$$sum = \overline{c}_{in}.(\overline{a}.b + a.\overline{b} + c_{in}.\overline{(\overline{a}.b + a.\overline{b})}$$

$$c_{out} = b.a + c_{in}.(b + a)$$

4.3 Ripple Carry Adder

- To add two n bit numbers chain together n full adders
- If you compliment one of the numbers (a) and set the first carry in bit to 1 then you have implemented a subtractor, with the a subtracted

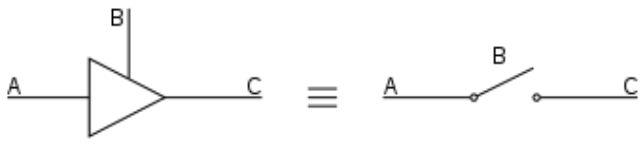
4.4 Fast Carry Generation

- **Fast Carry** - Unlike carry adder, doesn't require previous adders to have been completed
- Different Carries
  - Carry Kill - The carry out is aways zero, when $k_i = \overline{a}_i.\overline{b}_i$
  - Carry Propagate - The carry is the same as the carry in, when $p_i = a_i \oplus b_i$
  - Carry Generate - The carry our is generated independently of carry in, when $g_i = a_i.b_i$
- The $i + 1^{\text{th}}$ carry is given by $c_{i+1} = g_i + c_i.p_i$

5 Implementing Combinational Logic in Other Ways

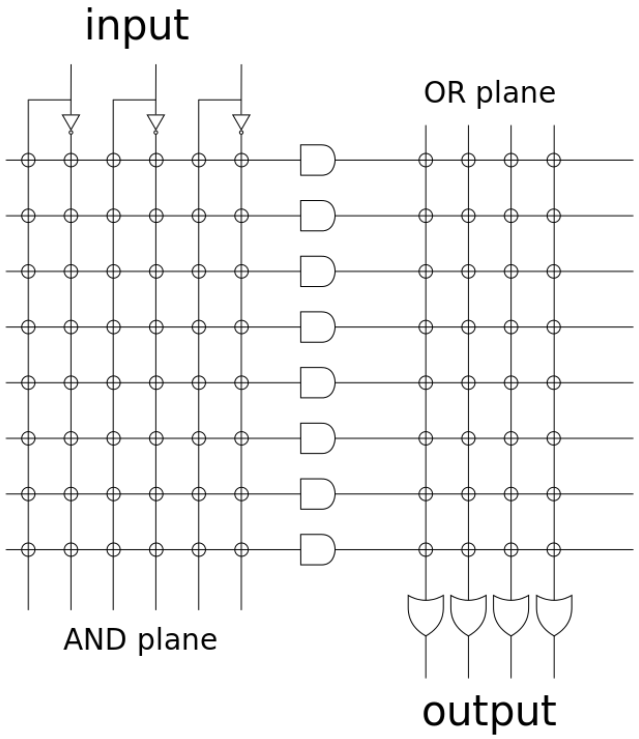
5.1 ROM

- Read only memory
- Acts like a look up table
- Can be efficient if lots of minterms must be generate, otherwise will have few non-zero entries
- CPU uses address bus to specify location data being read/written to
- Data bus used to convey data too and from location
- A tristate buffer is used to control what can output to a bus



- The following control signals are used
  - OE - Output enable, enable output buffers
  - WE - Write enable, determines whether data is written or read (not for ROM)
  - CS - Chip select, determines if the chip is activated

5.2 PAL



- Programmed by selectively removing connections in the AND and OR planes
- These are controlled by fuses or memory bits

6 Components

- **Multiplexor** - Chooses one of many inputs
- **Demultiplexos** - A single input is directed to one output
- **Decoder** - Enables a specific output

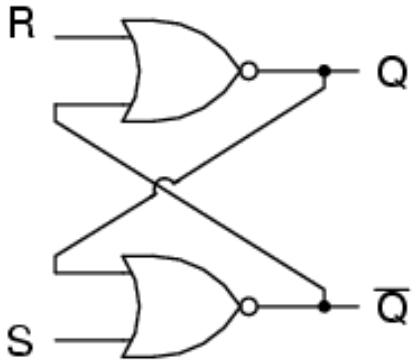
7 Sequential Logic

7.1 Excitation and Characteristic Tables

- **Characteristic Table** - Gives the next state of the output given the current inputs and state
- **Excitation Table** - Gives the required inputs to achieve a particular next state given a current state

7.2 Latches

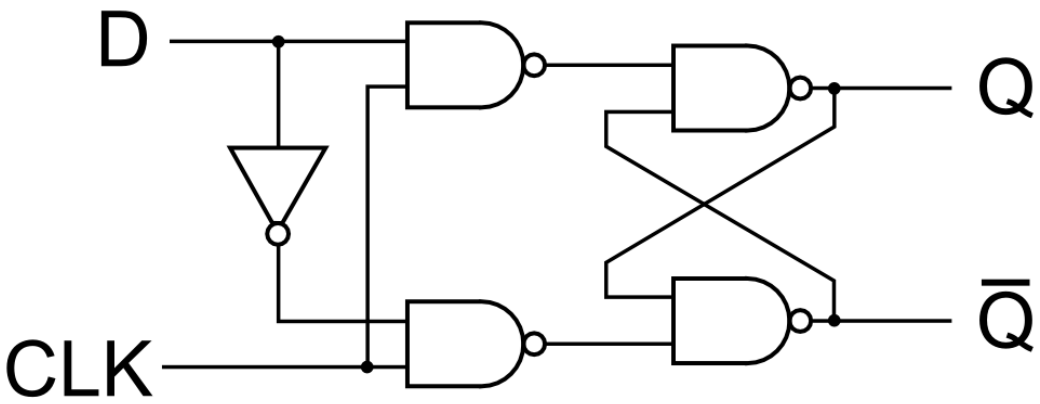
- **RS Latch**



S	R	Q	$\overline{Q}$
0	0	latch	latch
0	1	0	1
1	0	1	0
1	1	0	0

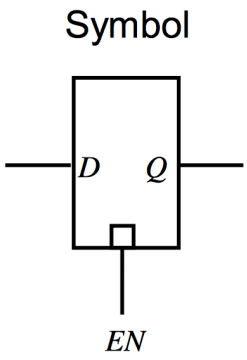
- Synchonisation:
  - Asynchronous - Output changes as a direct result of input changing
  - Synchronous - Output only changes as a result of a global enabling signal or clock

- **Transparent D Latch**

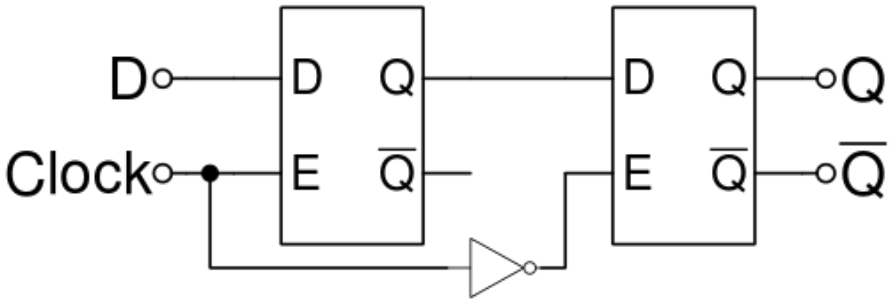


D	EN	Q'	$\overline{Q}'$	comment
X	0	Q	$\overline{Q}$	RS hold
0	1	0	1	RS reset
1	1	1	0	RS set

  - This level triggered
  - Is represented using the following symbol



- **Master-Slave D Flip-Flop**
  - Edge triggered - Q changes with the rising edge of CLK



- **J-K Flip-Flop**
  - Extension of a D flip flop
  - Has the following state transition table

J	K	Q'	$\bar{Q}$'	Comment
0	0	Q	$\bar{Q}$	Hold
0	1	0	1	Reset
1	0	1	0	Set
1	1	$\bar{Q}$	Q	Toggle

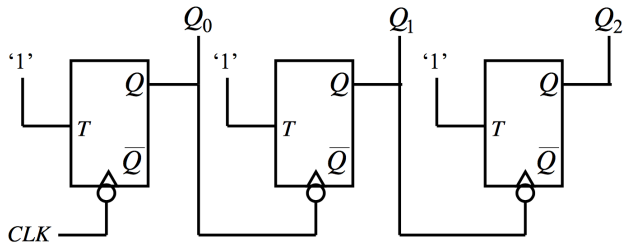
- **T Flip-Flop**
  - Extension of a J-K flip flop
  - Equivalent to connecting J and K together
  - Has the following state transition table

T	Q'	$\bar{Q}$'	Comment
0	Q	$\bar{Q}$	Hold
1	$\bar{Q}$	Q	Toggle

7.3 Sequential Circuits

- **Timing**
  - Setup Time - Minimum time that the data must be stable at the input before the clock edge
  - Hold Time - Minimum time that the data must remain stable on the flip-flop input after the click edge

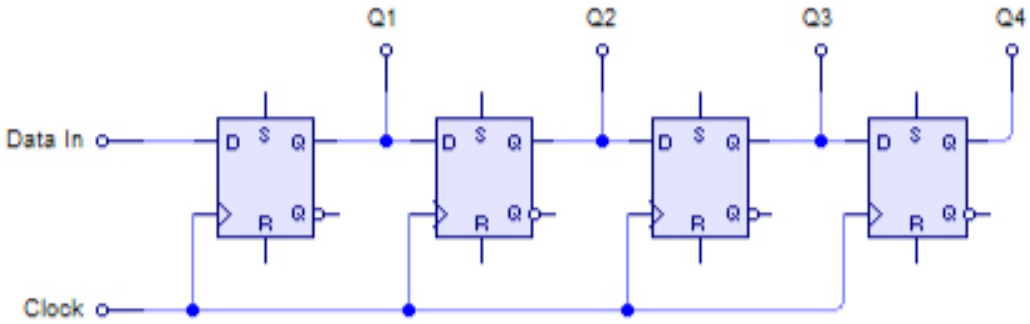
- **Ripple Counter**



  - Not synchronous
  - Outputs don't change at the same time
  - Propagation delay builds up

- **Synchronous Counter** - Use Global clock signal, use the following steps
  - Write down state transition table
  - Determine FF excitation table
  - Determine the combinations logic necessary to generate the required FF excitation from the current states

- **Shift Register** - Converts serial data into parallel data

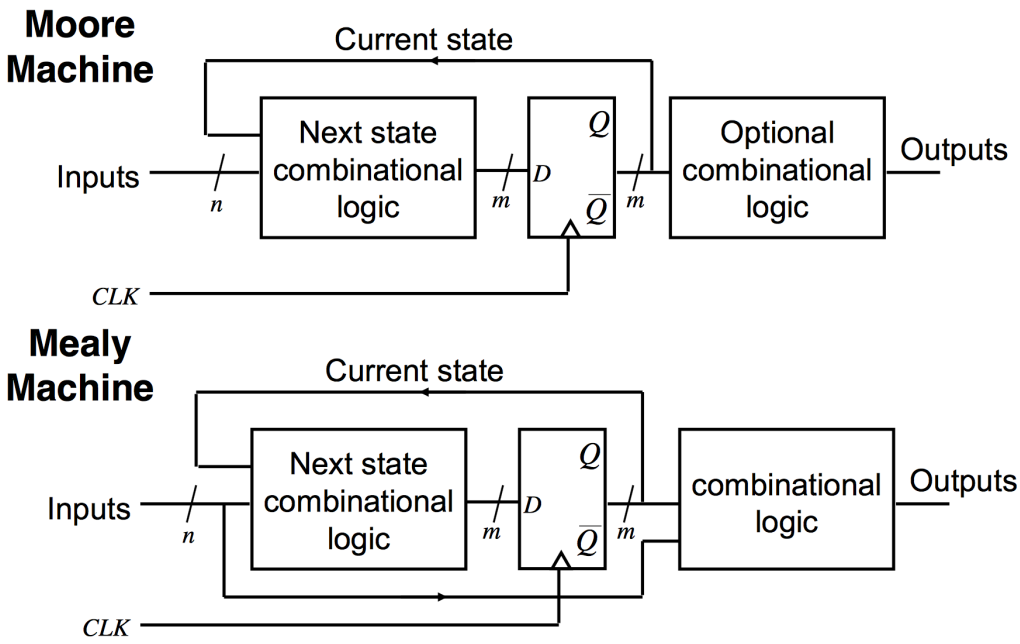


7.4 Finite State Machine

7.4.1 Types of Machine

- **Finite State Machines** - Deterministic machine that produces outputs which depends on its internal state and external inputs
- **States** - The set of internal memorised values, shown as circles on the state diagram
- **Inputs** - External stimuli, labelled as arcs on the state diagram
- **Outputs** - Results from the FSM
- **Moore Machines**
  - Output comes directly from clocked flip flops
  - Define logic based on state
- **Mealy Machine**

  - Outputs depend upon timing of inputs
  - Each transition has Input/Output
- Difference between them



  - Mealy must have output logic that uses both the input and state

7.4.2 Implementation

- When designing the machine, ensure that every start up state will eventually reach a valid state
  - If not then use asynchronous clear and preset FF inputs to set to a known state at power up
  - If it does reach a valid start for all start states it is said to self-start
- Wiring complexity can be as bigger issue as gate complexity
- **Storing State**
  - **Sequential State Assignment**
    - Just count up
  - **Sliding State Assignment**

c	b	a	c'	b'	a'
0	0	0	0	0	1
0	0	1	0	1	1
0	1	1	1	1	0
1	1	0	1	0	0
1	0	0	0	0	0

  - **Shift Register Assignment**
    - Output from final shift register is connected to the input of the first
    - Data continues to cycle through the register

e	d	c	b	a	e'	d'	c'	b'	a'
0	0	0	1	1	0	0	1	1	0
0	0	1	1	0	0	1	1	0	0
0	1	1	0	0	1	1	0	0	0
1	1	0	0	0	1	0	0	0	1
1	0	0	0	1	0	0	0	1	1

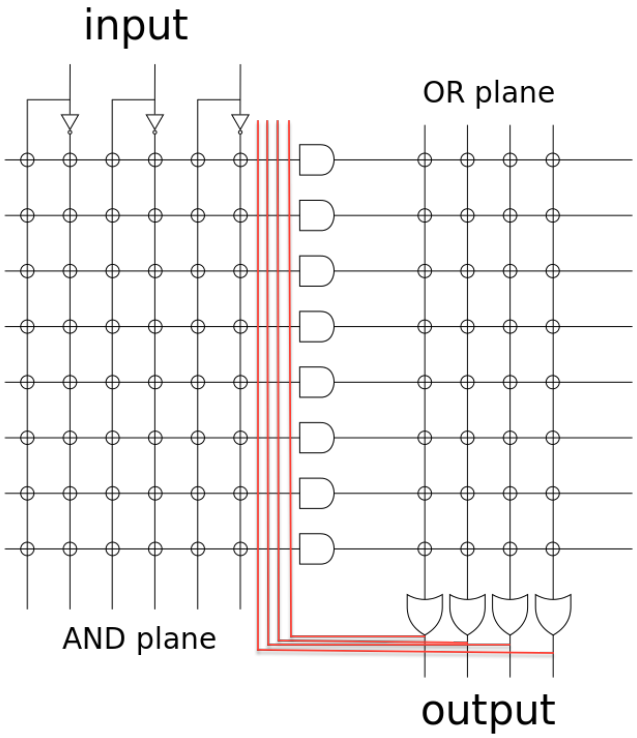
  - **One Hot State Encoding**
    - Shift register assignment buy with only one flip-flop holds a 1
    - Only one flip-flop has a 1
    - One flip-flop per state
    - Results in simple fast state machines
    - Outputs are generated by ORing together appropriate flip-flop outputs

- State minimisation - Using row matching
  - Draw out the state transition table, eg.

Current State	Next State X=0	Next State X=1	Output X=0	Output X=1
A	B	C	0	0
⋮	⋮	⋮	⋮	⋮

  - Combine any states with the same next states and outputs
  - Repeat until no further combinations can be done

- Can use GAL (generic array logic) devices to implement FSMs
  - Similar to PALs but outputs from OR plane go back into AND plane



- Can use FPGAs (field programmable logic array)
  - A sea of wires controlled by look up tables
  - Look up tables downloaded from memory

8 Electricity

8.1 Electric Laws

- Current is the rate of charge

$$Q = It$$

- Electromotive force is the energy per unit charge provided by a battery

$$V = \frac{E}{Q}$$

- Power is the rate of energy dissipation

$$P = \frac{E}{t} = \frac{QV}{t} = IV$$

- Capacitors store charge
  - Charging

$$I = I_0 e^{\frac{-t}{CR}}$$

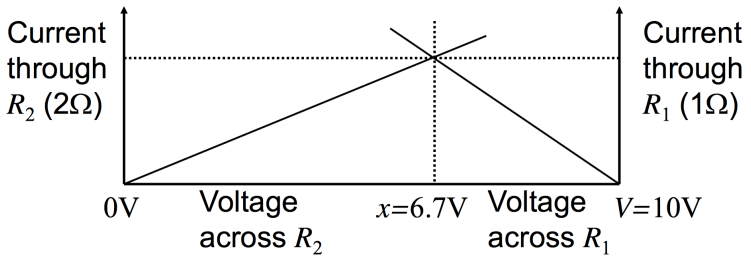
$$V_{out} = V_{cell}\left(1 - e^{\frac{-t}{CR}}\right)$$

  - Discharging

$$V_{out} = V_{cell} e^{\frac{-t}{CR}}$$

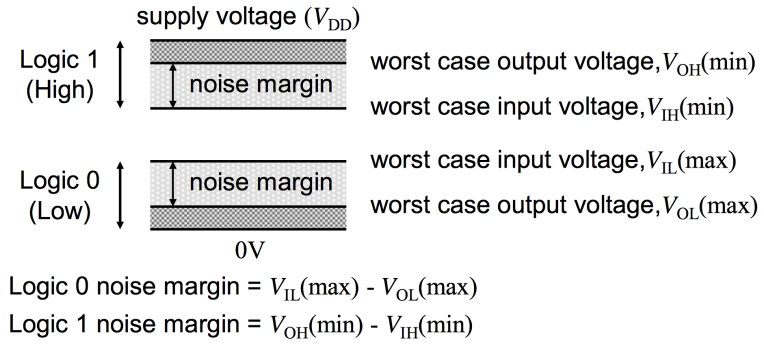
8.2 Circuit Theory

- Kirchhoff's Current Law - The sum of currents entering a junction is zero
- Kirchhoff's Voltage Law - In any closed loop of an electric circuit the sum of all the voltages in that loop is zero
- Graphical Approach
  - Plot the I-V characteristics of the components, with one starting at the left hand side of the axis, the other on the right
  - Where the cross gives the current and voltages through them, eg.



8.3 Noise Margin

- The tolerence for noise is quantified in terms of the noise margin



Logic 0 noise margin = $V_{IL}(max) - V_{OL}(max)$

Logic 1 noise margin = $V_{OH}(min) - V_{IH}(min)$

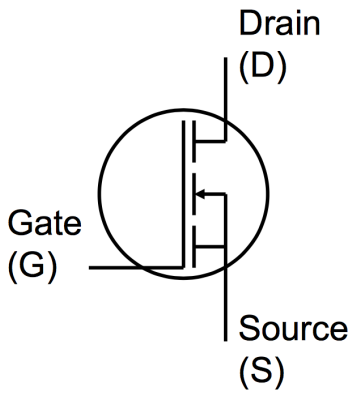
8.4 Semiconductors

- A material with a low electron density
- Silicon is a poor conductor - a semiconductor
  - As temperature rises ($> 0°K$) electrons break bonds and can move freely
- n-type Silicon
  - The silicon is doped with Arsenic, giving it an extra electron
  - This electron will break free and move
  - The free electron will conduct electricity
  - Arsenic is a donor
- p-type Silicon
  - The silicon is doped with boron
  - This leaves a hole, an absent valence electron
  - This hole can move throughout the lattice
  - The hole has a positive charge and therefore causes a flow of current
  - Boron is an acceptor
- p-n Junction
  - n-type and p-type semiconductors are combined
  - The electrons and holes diffuse across junction because of the large concentration gradient
  - Electrons diffusing out of n-type leaves positively charged donor atoms
  - Holes diffusing out of p-type leaves negatively charged acceptor atoms
  - This creates a depleted region, with no holes or electrons, this is the space charge region
  - Depleted region causes an electric field that opposes diffusion
  - Equilibrium reached where no more charges move across junction
  - The field's pd is known as the **contact potential**
- Biased p-n Junction
  - The current throw a p-n junction depends on its direction

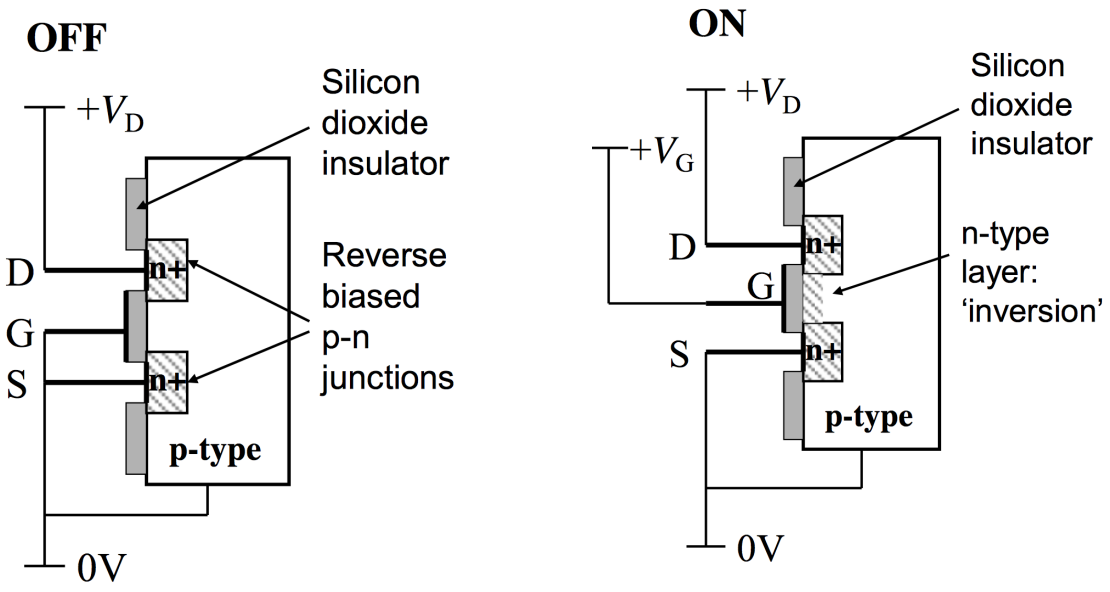
- **Reverse Bias** - n-type positive
  - n-type positive, electrons removed, holes removed from p-type
  - Space charge region and its electric field increased
  - **Causes a small current**
- **Foward Bias** - n-type negative
  - n-type negative, electrons pushed towards junction, holes also pushed towards junction in p-type
  - Space charge region and its electric fields reduced
  - Diffusion current increases
  - **Causes a large current**
- This is a diode

9 Transistors

9.1 MOSFET

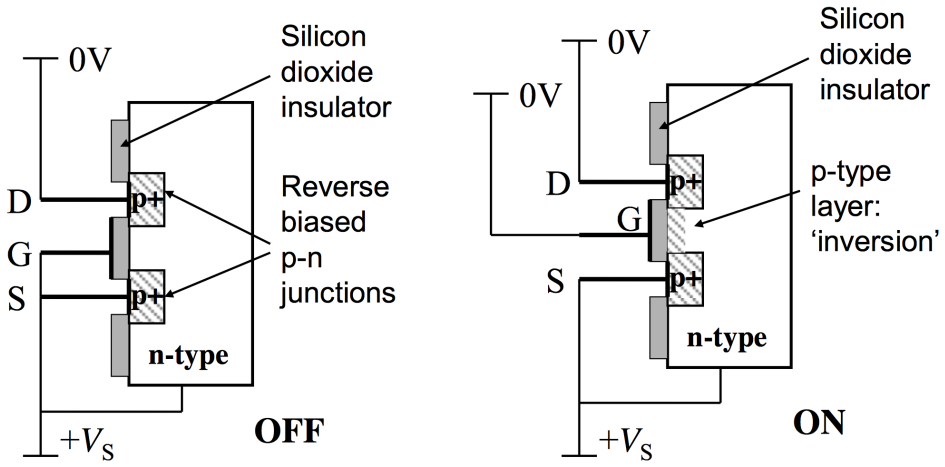


- The current flow from D to S is controlled by the voltage applied between G and S (V_{GS})
- When $V_{GS} = 0V$ the transistor is off, $I_{DS} = 0A$
- **n-channel MOSFET**



- When the gate voltage V_g is positive electrons are attracted to G
- This creates an n-type channel and a path from S to D
- Only occurs when V_G is larger than the threshold voltage, typically 0.3 to 0.7V

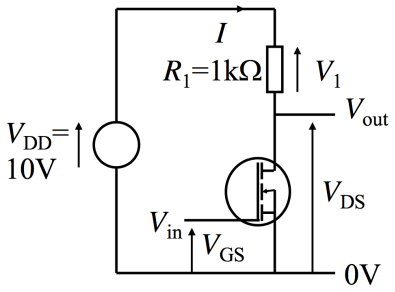
- **p-channel MOSFET**



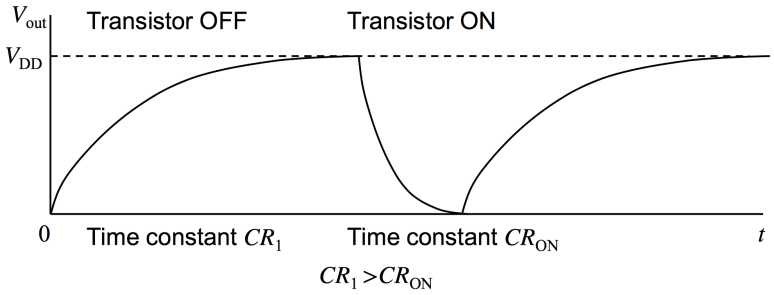
- This is the opposite to an n-channel

9.1.1 n-MOS Transistors

- MOSFETs can be used to create an n-MOS inverter



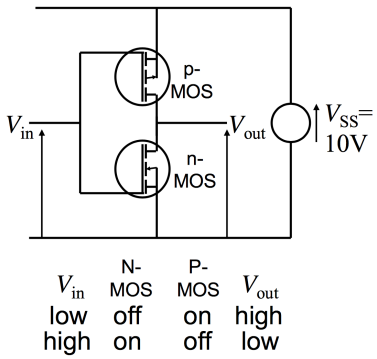
  - This does not have the ideal characteristic the we would want
- n-MOS can be used to make other logic functions (eg. NAND or NOR)
  - It has fundamental problems
    - Speed of operation
    - Power consumption
  - Main speed limitation is due to stray capacitance, giving the following graph



  - When the transistor is on current follows throw R_1, dissipating power

9.1.2 CMOS Logic

- Uses n-channel and p-channel MOS transistors
- CMOS Inverter



  - Has much better characteristics than n-MOS inverter
  - Only dissipates energy when switching (when both transistors are on)
- Can be used to build logic gates
- Best to stick to one logic family (eg. CMOS, NMOS, ECL, TTL)