EECS150 - Digital Design

Lecture 8 - CMOS Implementation Technologies

Feb 11, 2010

John Wawrzynek

Mux4 Testbench

```
module testmux4;
  reg [5:0] count = 6’b000000;
  reg a, b, c, d, expected;
  reg [1:0] S;
  wire f;
  mux4 myMux (.select(S), .in0(a), .in1(b), .in2(c), .in3(d), .out(f));
  initial
    begin
      repeat(64)
        begin
          {S, d, c, b, a} = count[5:0];
          case (S)
            2’b00:  expected = a;
            2’b01:  expected = b;
            2’b10:  expected = c;
            2’b11:  expected = d;
          endcase // case(S)
          #8 $strobe( "select=%b in0=%b in1=%b in2=%b in3=%b out=%b,
                       expected=%b time=%d", S, a, b, c, d, f, expected, $time);
          #2 count = count + 1’b1;
        end
      $stop;
    end
endmodule
```

Declaration and initialization all at once. Generally not available in synthesis.

DUT instantiation

Enumerate all possible input patterns.

Apply pattern to DUT

Behavioral model of mux4

$strobe displays data at a selected time. That time is just before simulation time is advanced (after all other events).

Delay to allow mux outputs to stabilize. Here we assume mux delay < 8ns.

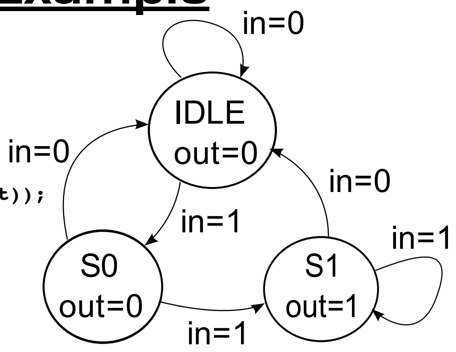
Wait a bit, then bump count.

Alternative to $strobe in this case,

#8 if (f != expected) $display(“Mismatch: ...);

FSM Testbench Example

Test all arcs.



```
module testFSM;
  reg in;
  wire out;
  reg clk=0, rst;
  reg expect;
  FSM1 myFSM (.out(out), .in(in), .clk(clk), .rst(rst));
  always #5 clk = ~clk;
  initial
    begin
      rst=1;
      #10 in=0; rst=0; expect=0;
      #10 in=1; rst=0; expect=0;
      #10 in=0; rst=0; expect=0;
      #10 in=1; rst=0; expect=0;
      #10 in=1; rst=0; expect=1;
      #10 in=1; rst=0; expect=1;
      #10 in=0; rst=0; expect=0;
      #10 $stop;
    end
  always
    begin
      #4 $strobe($time," in=%b, rst=%b, expect=%b out=%b", in, rst, expect, out);
      #6 ;
    end
endmodule
```

DUT instantiation

100MHz clk signal

start in IDLE

self-loop

transition to S0

transition to IDLE

transition to S0

transition to S1

self-loop

transition to IDLE

Note: Input changes are forced to occur on negative edge of clock.

Strobe output occurs 1ns before rising edge of clock.

FSM Verification/Debug is easier if you have access to state value also. Either 1) bring out to ports, or 2) use waveform viewer.

Final Words (for now) on Simulation

Testing is not always fun, but you should view it as part of the design process. Untested potentially buggy designs are a dime-a-dozen. Verified designs more rare and have real value.

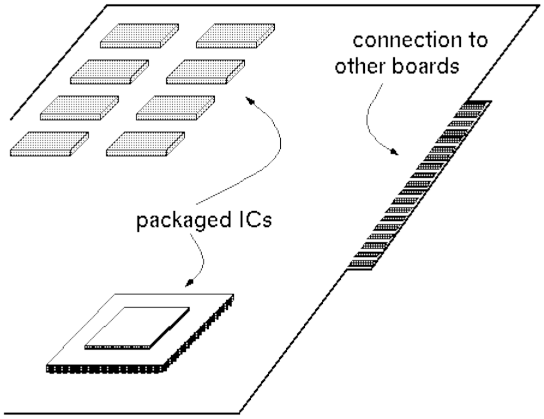
Devising a test strategy is an integral part of the the design process. It shows that you have your head around the design. It should not be an afterthought.

Overview of Physical Implementations

The stuff out of which we make systems.

- Integrated Circuits (ICs)
  - Combinational logic circuits, memory elements, analog interfaces.
- Printed Circuits (PC) boards
  - substrate for ICs and interconnection, distribution of CLK, Vdd, and GND signals, heat dissipation.
- Power Supplies
  - Converts line AC voltage to regulated DC low voltage levels.
- Chassis (rack, card case, ...)
  - holds boards, power supply, fans, provides physical interface to user or other systems.
- Connectors and Cables.

Printed Circuit Boards

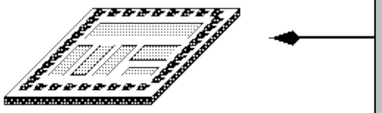


- fiberglass or ceramic
- 1-25 conductive layers
- 1-20in on a side
- IC packages are soldered down.

Multichip Modules (MCMs)

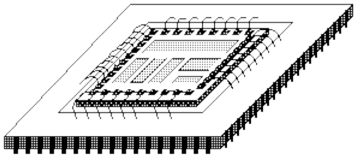
- Multiple chips directly connected to a substrate. (silicon, ceramic, plastic, fiberglass) without chip packages.

Integrated Circuits



- Primarily Crystalline Silicon
- 1mm - 25mm on a side
- 100 - 1000M transistors
- (25 - 250M "logic gates")
- 3 - 10 conductive layers
- 2010 - feature size ˜ 45nm = 0.045×10^{-6} m
- "CMOS" most common - complementary metal oxide semiconductor

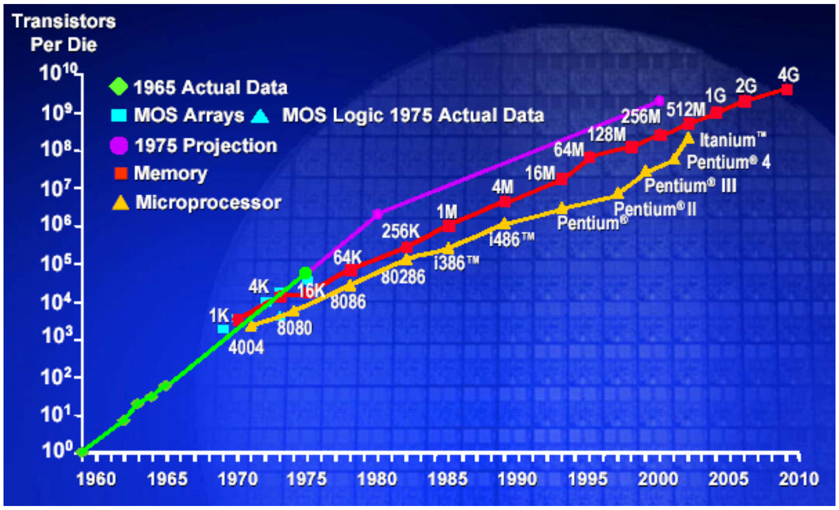
Chip in Package



- Package provides:
  - spreading of chip-level signal paths to board-level
  - heat dissipation.
- Ceramic or plastic with gold wires.

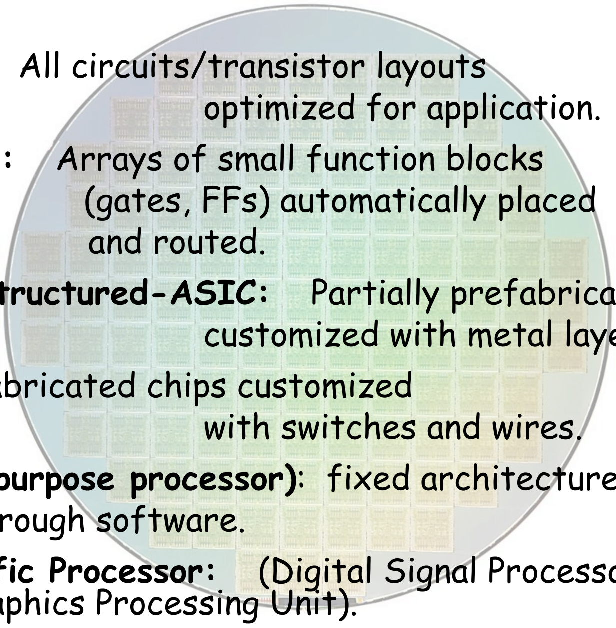
Integrated Circuits

- Moore's Law has fueled innovation for the last 3 decades.



- "Number of transistors on a die doubles every 18 months."
- What are the consequences of Moore's law?

Chip-level Function Implementation Alternatives



Full-custom: All circuits/transistor layouts optimized for application.

Standard-cell: Arrays of small function blocks (gates, FFs) automatically placed and routed.

} ASIC

Gate-array/structured-ASIC: Partially prefabricated wafers customized with metal layers.

FPGA: Prefabricated chips customized with switches and wires.

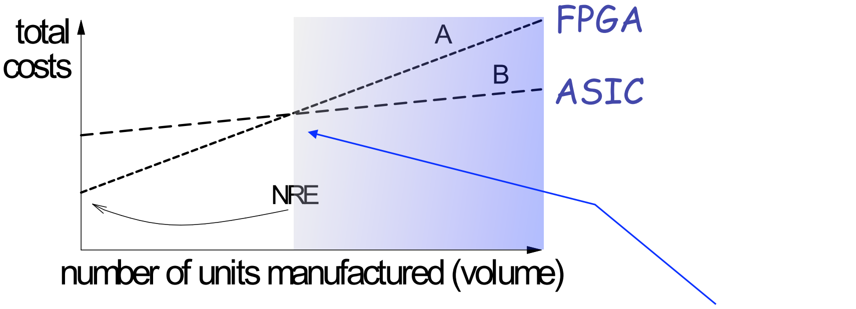
GPP (general purpose processor): fixed architecture customized through software.

Domain Specific Processor: (Digital Signal Processor, Network Processor, Graphics Processing Unit).

What are the important metrics of comparison?

Why FPGAs?

A tradeoff exists between NRE* cost and manufacturing costs:



The ASIC approach is only viable for products with very high volume (where NRE could be amortized), and which were not time to market (TTM) sensitive.

Cross-over point has moved to the right (favoring FPGA) implementation as ASIC NREs have increased.

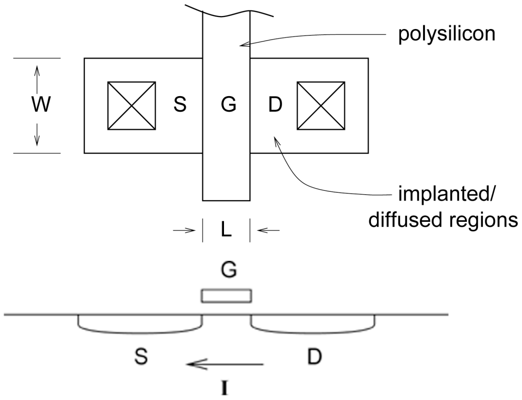
*Non-recurring Engineering Costs

CMOS Devices

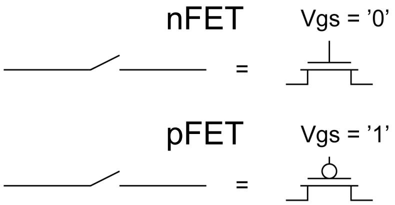
- MOSFET (Metal Oxide Semiconductor Field Effect Transistor).

Top View

Cross Section

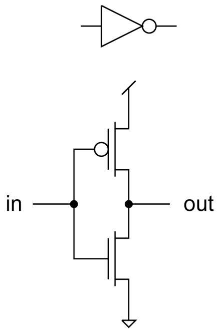


The gate acts like a capacitor. A high voltage on the gate attracts charge into the channel. If a voltage exists between the source and drain a current will flow. In its simplest approximation, the device acts like a switch.

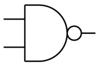


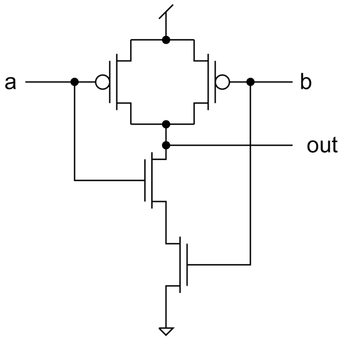
Transistor-level Logic Circuits

Inverter (NOT gate):



NAND gate:





Note:

- out = 0 iff a AND b =1 therefore out = (ab)'
- pFET network and nFET networks are duals of one another.

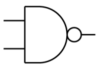
How about AND gate?

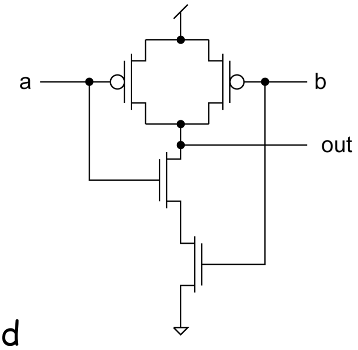
Transistor-level Logic Circuits

Simple rule for wiring up MOSFETs:

nFET is used only to pass logic zero.
pFET is used only to pass logic one.

For example, consider the NAND gate:

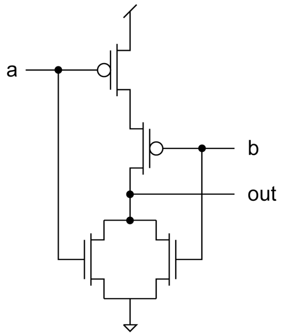


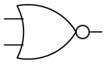


Note: This rule is sometimes violated by expert designers under special conditions.

Transistor-level Logic Circuits

NOR gate:





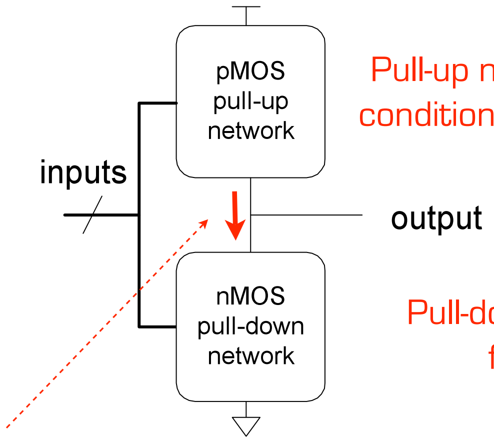
Note:

- out = 0 iff a OR b =1 therefore out = (a+b)’

- Again pFET network and nFET networks are duals of one another.

Other more complex functions are possible. Ex: out = (a+bc)’

CMOS Logic Gates in General



Pull-up network conducts under input conditions to generate a logic 1 output

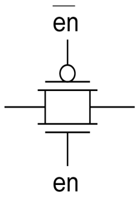
Pull-down network conducts for logic 0 output

Conductance must be mutually exclusive - else, short circuit!

Pull-up and pull-down networks are “topological duals”

Transmission Gate

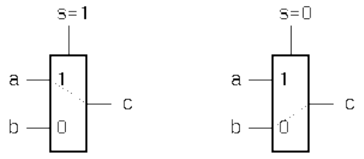
- Transmission gates are the way to build “switches” in CMOS.
- In general, both transistor types are needed:
  - nFET to pass zeros.
  - pFET to pass ones.
- The transmission gate is bi-directional (unlike logic gates).



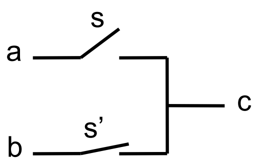
- Does not directly connect to Vdd and GND, but can be combined with logic gates or buffers to simplify many logic structures.

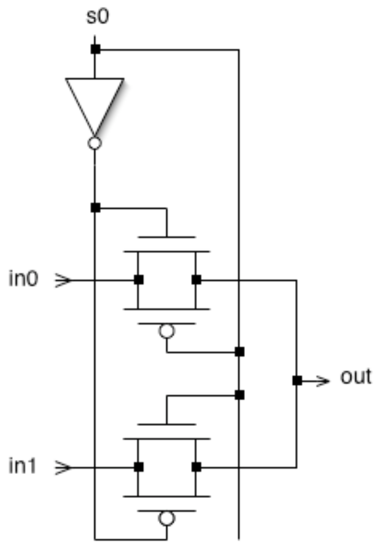
Transmission-gate Multiplexor

2-to-multiplexor:
C = sa + s'b



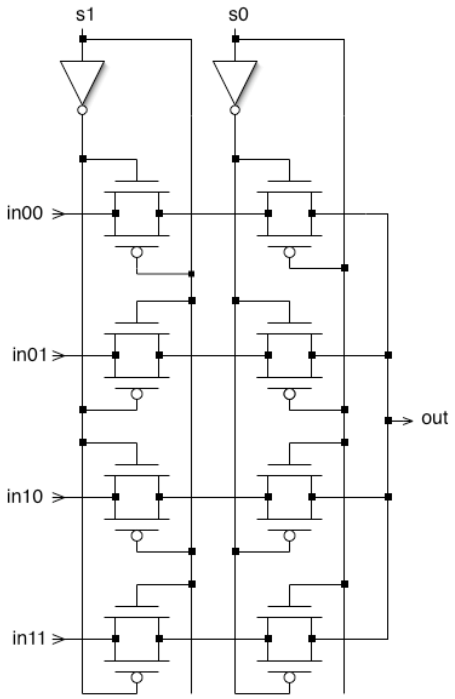
Switches simplify the implementation:





Compare the cost to logic gate implementation.

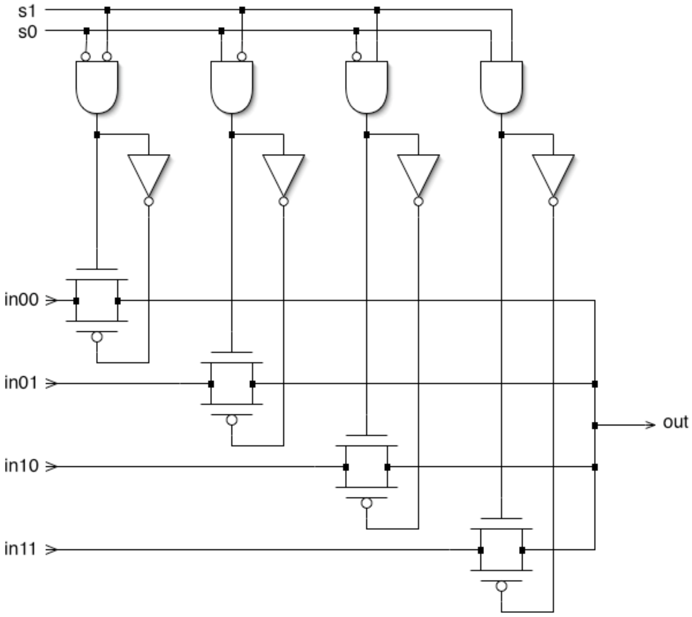
4-to-1 Transmission-gate Mux



- The series connection of pass-transistors in each branch effectively forms the AND of s1 and s0 (or their complement).
- Compare cost to logic gate implementation

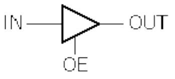
Alternative 4-to-1 Multiplexor

- This version has less delay from in to out.
- In both versions, care must be taken to avoid turning on multiple paths simultaneously (shorting together the inputs).



Tri-state Buffers

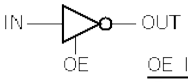
Tri-state Buffer:



OE	IN	OUT
0	0	Z
0	1	Z
1	0	0
1	1	1

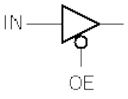
"high impedance" (output disconnected)

Variations:



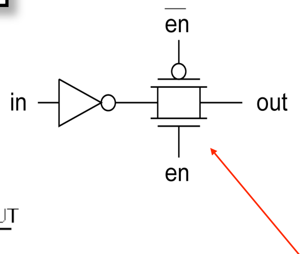
OE	IN	OUT
0	–	Z
1	0	1
1	1	0

Inverting buffer

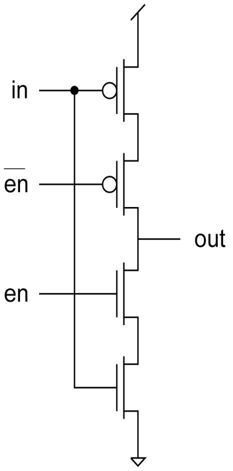


OE	IN	OUT
0	0	0
0	1	1
1	–	Z

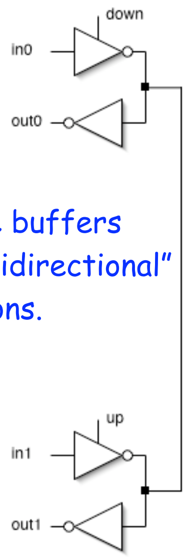
Inverted enable



transmission gate useful in implementation

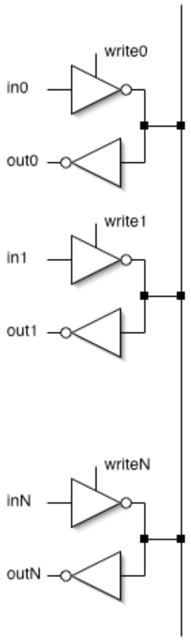


Tri-state Buffers



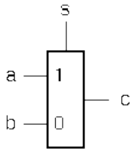
Tri-state buffers enable "bidirectional" connections.

Tri-state buffers are used when multiple circuits all connect to a common wire. Only one circuit at a time is allowed to drive the bus. All others "disconnect" their outputs, but can "listen".

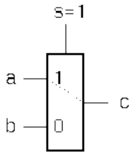


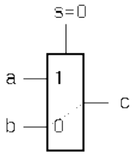
Tri-state Based Multiplexor

Multiplexor

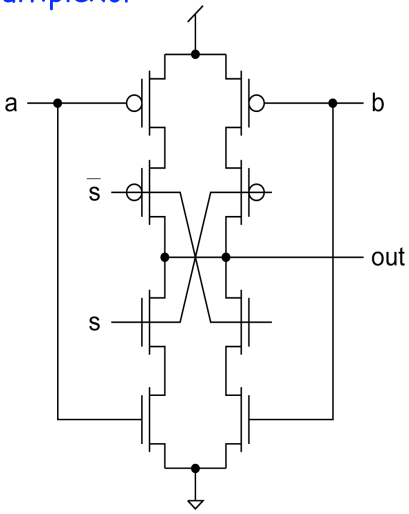


If s=1 then c=a else



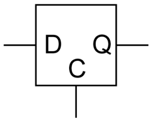


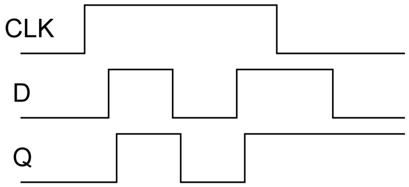
Transistor Circuit for inverting multiplexor:



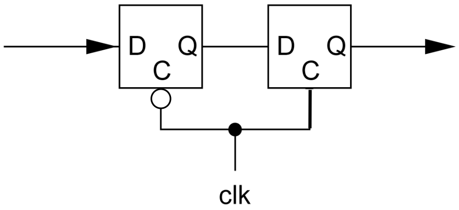
Latches and Flip-flops

Positive level-sensitive **latch**:

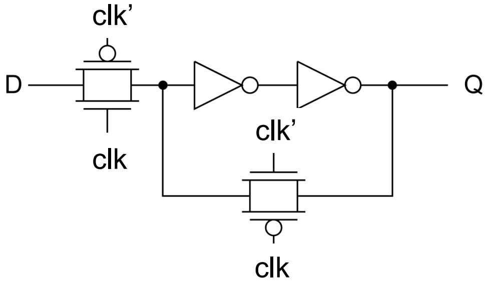




Positive Edge-triggered **flip-flop** built from two level-sensitive latches:



Latch Implementation:


clk'
D
clk
clk'
Q
clk