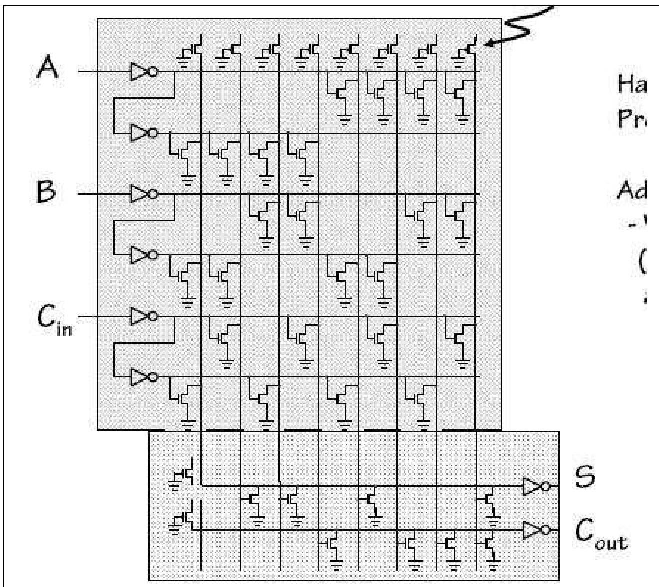
Daniel Leeds, 6.004 R10, March 15, 2006; Quiz #2 Review

Fine print:
Quiz is closed-book, no calculators; covers up to L09 (Pipelining)/R10 (this recitation)

Practice, practice, practice:
Follow "Previous terms" link from http://6004.csail.mit.edu, pick a semester (the more recent, the better), click on the "Announcements" page for the semester, and find the PDF for Quiz 2 solutions. (For semesters from a few years ago, some (or all) of the material may be in Quiz 1.) Don't read the answers until you first figure them out for yourself!

Another perspective on the material – Margaret Chong's Handbook:
Follow "Handouts" link from http://6004.csail.mit.edu, click on handbook link near the bottom of the page.

Good topics to know:
Synthesis of Combinational Logic
ROMs

	$S = \bar{A}\bar{B}C + \bar{A}B\bar{C} + A\bar{B}\bar{C} + ABC$ Each vertical ("word") line represents an **and** term. The word line takes the and of all input lines $(A, B, C, \bar{A}, \bar{B}, \bar{C})$ connected to it via NFET. For example, right-most line is ABC Horizontal ("bit") line represents the **or** of all vertical word lines connected to it. (For example, the right-most word line is connected, via NFET, to the bit line for S, so S includes a term for ABC.)

Karnaugh Maps

"A truth table arranged so that terms which differ by exactly one variable are adjacent to one another"

Strategy (circling the 1s):

Circle largest N-dimensional rectangles (with sides of 2^N)

Keep circling largest remaining rectangles (even if they overlap) until no 1s are left

Choose fewest rectangles to cover all the ones -> Minimal SOP form

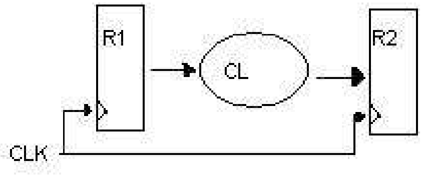
Include terms for all prime implicants to construct a lenient circuit.

Sequential Logic

Our registers latch in new value at rising clock edge

$t_{SETUP,R1}$ and $t_{HOLD,R1}$ must be met by R1's input to guarantee valid output at R1

$t_{CD,R1}$ and $t_{PD,R1}$ relate to R1 output after clock edge



$t_{HOLD,R2} < t_{CD,R1} + t_{CD,CL}$

$t_{CLK} > t_{SETUP,R2} + t_{PD,R1} + t_{PD,CL}$ ↔ $t_{SETUP,R2} < t_{CLK} – (t_{PD,R1} + t_{PD,R2})$

FSMs

A Finite State Machine consists of k states. Each state has a set of transition rules for each input, and may have one or many outputs.

Hardware Implementation

ROM-register loop

2^n states, m inputs, p outputs -> $p*2^{(n+m)}$ bits in ROM (i.e., output entries in FSM truth table)

Synchronization/Metastability

Time-based problem – inputs occur too close together in time (asynchronous inputs)

Metastable state resolves in **unbounded** time

"Solution:" Cascading flip-flops increases the probability that the metastable state will be resolved to valid logic value before our logic circuitry sees it.

Alternative "solution:" realize you are supposed to be using combinational logic. Combinational logic has valid output in bounded time.

Pipelining

Typically, better performance means:

decreasing latency and/or

increasing throughput

Pipelines:

can increase throughput (not always)

may increase latency (not always)

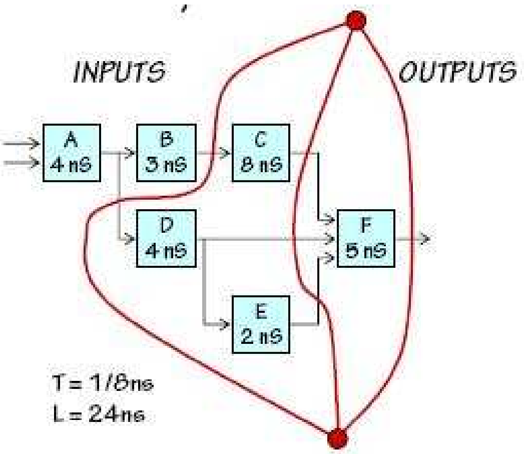
Un-pipelined circuit: Latency = 1 / Throughput

In pipelined circuit: Latency = (# pipeline stages) * (t_{PD} at longest pipeline stage)

Methodology:
1 Draw a line that crosses every output in the circuit and mark the endpoints as terminal points.

2 Continue to draw new “separating lines” between the terminal points across various circuit connections; add a pipeline register at every point where a separating line crosses a connection in the circuit.

Can place multiple registers between two circuit components, but each register must be part of a separate pipeline stage.



Circuit Interleaving: Interleaving n copies of a device increases throughput n-fold.

Pipelining starting with ibm 7094 circa 1969