6.978 LECTURE #10. OCT. 17, 1978

- TODAY: THE SORTER (CONT.). YIELD STATISTICS.

- SEMINARS: AM PLANNING TO HAVE SEVERAL CURR. ACT. RES. VISIT. THESE ARE OF GEN. INTEREST TO CLASS. WHERE POSS. WILL SCHED DURING CLASS HOURS. (OPEN)
- FIRST SEMINAR: DOUGLAS FAIRBAIRN, ON ICARUS & INT. LAY. SYS. HOPEFULLY TUES OCT 31. IF SO MIDTERM WILL BE MOVED TO THURS. NOV 2. (WILL KNOW NEXT TIME)
- MIDTERM: WORTH 1/3 OF FINAL. ON BASICS. THE HW ASSIGNED THIS TIME IS LAST BEFORE MIDTERM.

HW #4: SOME REALLY ELEGANT SOLNS. THOSE WHO GOT 10 or 10+ ARE WELL PREP. TO BEGIN PROJECTS.

MULTIBIT COMP 1 BIT VERT SLICE IN λ^2

~4600	Dean Brock	5280	Guy Steele
~4600	Andy Boughton	5300	Alan Snyder
5160	Randy Bryant	5328	Siu Ho Lam
5202	David Ottlen		

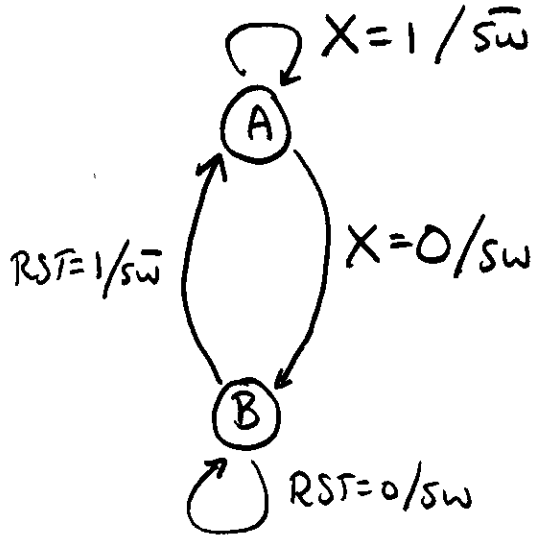
LAB: Glen Mirocker (alum) will open today ~3:30 to begin testability
Bill Henke: present more on CIF subset will use.

HW#5: The sorter is a good example of a major project ... The ideas we're covering will help you anticipate the work involved, the pitfalls you may encounter.

I tried to be clever on HW#5! Leaving some pitfalls for you to discover. Unfortunately I made a blunder which probably caused you unnecessary confusion: SLIDE -- ERROR

Continue with SORTER today. If you've made an effort to understand its function, you'll find the material today easy to follow and perhaps rather interesting!

- HAVE THE SORTER FIGURE (IN HW #5) ON SLIDE OR (ON BLACKBOARD.)
- USE WHITEBOARD(?) FOR SOME CIRCUIT FIGURES AND FOR CLARIFYING RECIRC VS LOADING.
- Review Sorter Fcn: LOAD, SORT.
- In HW #5, wanted to show use of very simple 2 state FSM ... showing that other than PLA might be best way to implement in simply cases. Unfortunately there was a superficial error in the transitions listed.
- Lets correct that error and look at a simple circuit implementing that FSM.

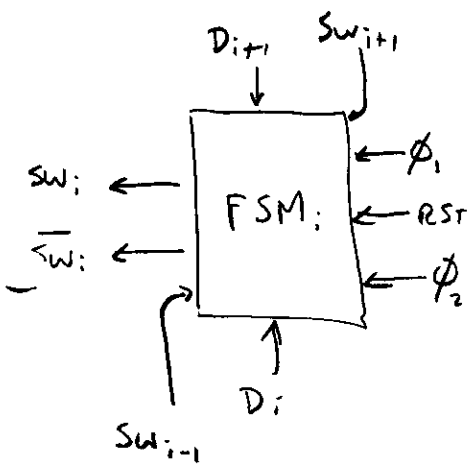


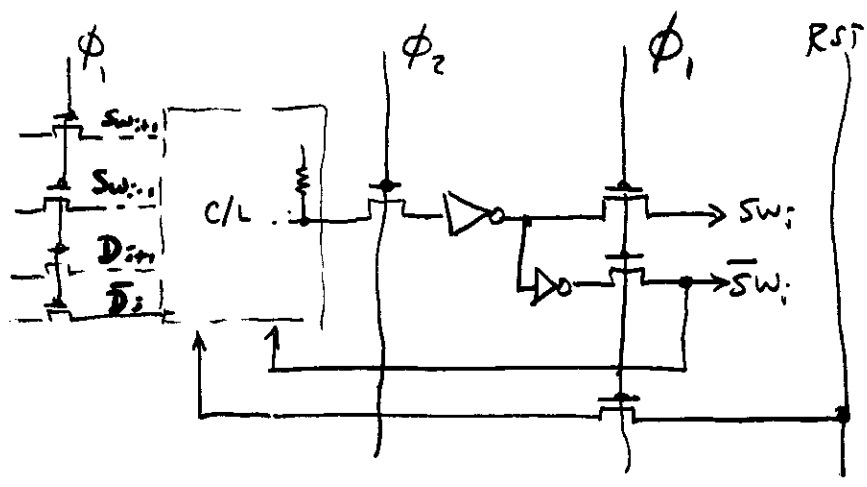
$$X = SW_{i+1} + SW_{i-1} + D_{i+1} + \overline{D_i} + RST$$

STATE A: NOT SWAPPING

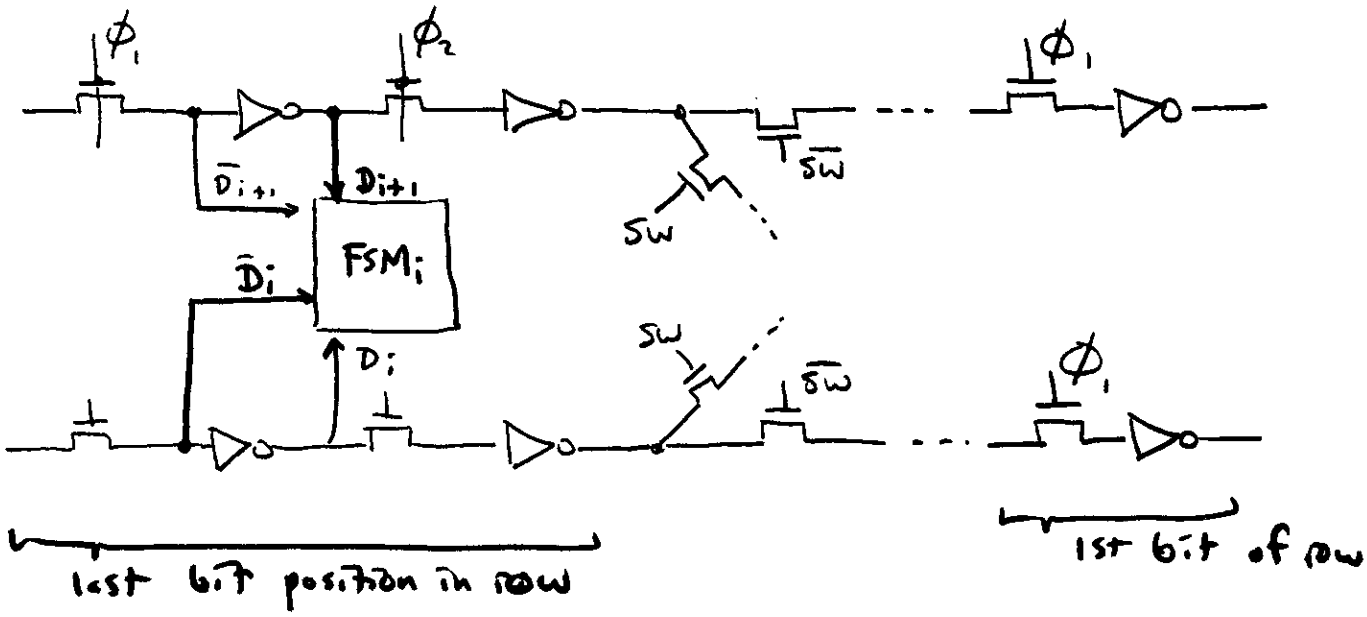
STATE B: SWAPPING

- FORM FOR INSERTION INTO THE SORTER:





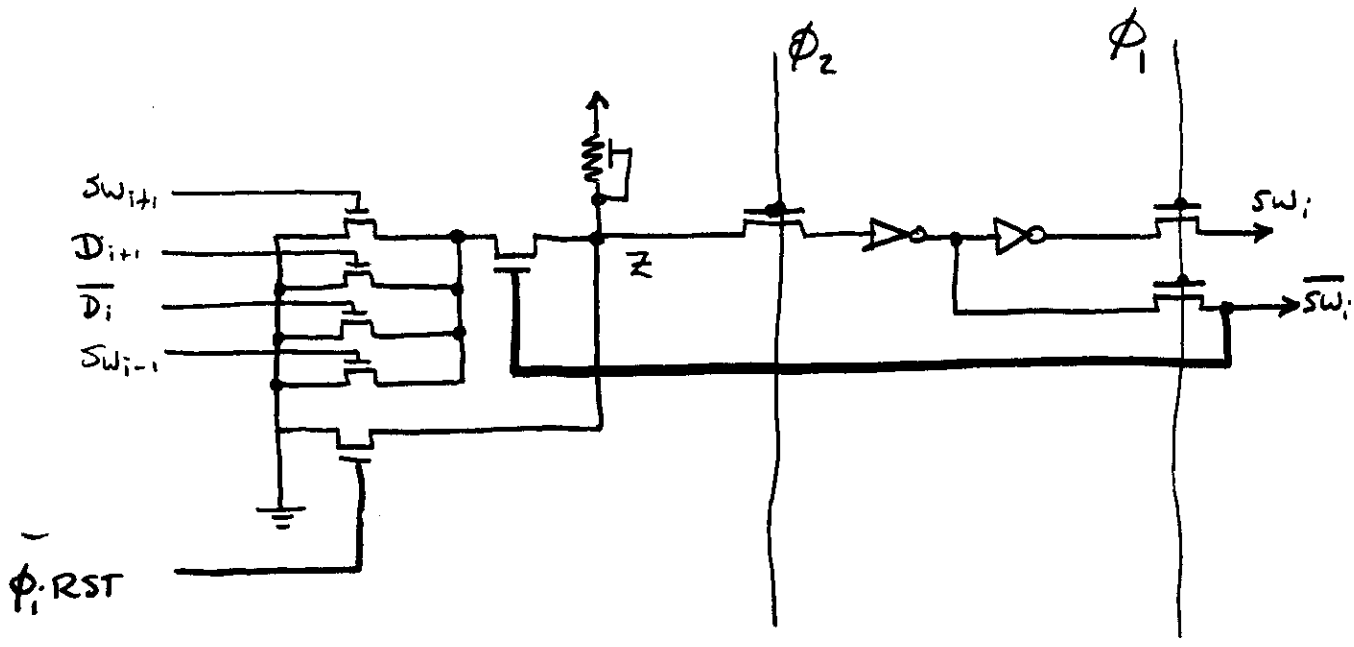
- ALL INPUTS SW_{i+1}, SW_{i-1}, D_{i+1}, D_i ARE TO BE TAKEN FROM POINTS "SET UP DURING ϕ_1": FOR EXAMPLE



- NOTE ALSO THAT THE OUTPUTS $\overline{SW_i}$, SW_i ARE SET UP INTO THE SWITCHES DURING THE FOLLOWING ϕ_1. ALSO, THEY INPUT ADJACENT FSM's DURING " " ϕ_1.

NOTE THAT A ROW SHOULD CONTAIN WORD$_i$ + 1 BITS

- HERE IS A SIMPLE CIRCUIT IMPLEMENTING FSM$_i$:



- EXPLANATION: > Reset PULLS DOWN Z, SETS OUTPUT TO $\overline{SW} = 1$.
  > THIS ENABLES NAND PART OF GATE.
  > THEN, IF RST OFF, AND ALL OTHER INPUTS = 0, Z IS PULLED UP AND SWAPPING IS INITIATED.
  > ONCE SW = 1, GATE IS DISABLED FROM PULLING DOWN AGAIN.

- NOTE: USE OF $\phi_2 - \phi_1$ CLOCK SCHEME MEANS WE DO HAVE TO WORRY ABOUT RELATIVE DELAYS CAUSING HAZARDS / RACES.

OK, THAT WAS THE HW PROBLEM: BUT WHAT WAS THE REAL PROBLEM? SHOULD WE JUMP IN AND START STICK DIAGRAMMING, & THEN BEGIN LAYOUT?

AH! NO! Because there is a fatal error in the algorithm!

We need a third state: (A) Not swapping (yet)
(B) Swapping
(C) Dont SWAP

Otherwise, we could have two rows in proper order, surrounded by other rows in proper order, that improperly begin swapping on first encountering a $\binom{0}{1}$ even though they previously encountered a $\binom{1}{0}$.

- So, we must add a state to "Remember Not To Swap" if we encounter $\overline{D}_{i+1} = 0$, $D_i = 0$.

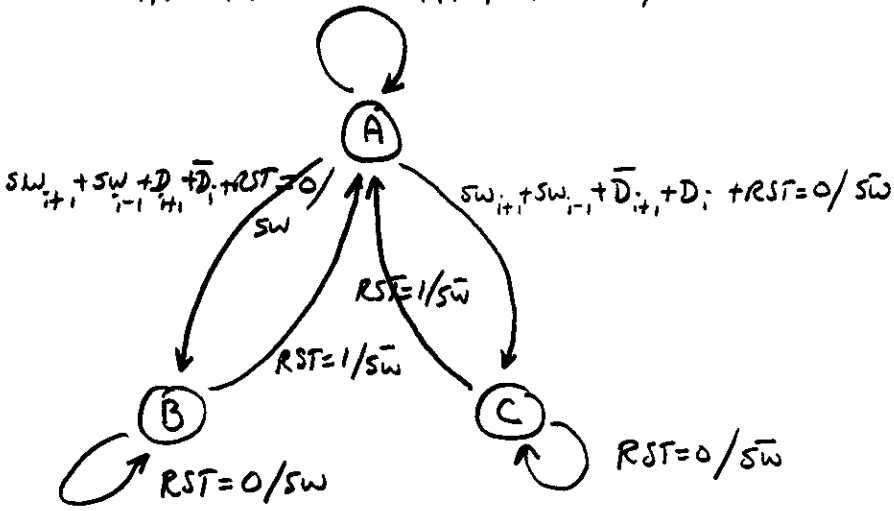
- A STATE DIAGRAM FOR THE CORRECT THREE STATE FSM:

and an extension of the previous circuit diagram to implement the three state FSM: is given in the next HW assignment.

$SW_{i+1} + SW_{i-1} + D_{i+1}D_i + \bar{D}_{i+1}\bar{D}_i + RST = 1/S\bar{W}$

STATE DIAGRAM:

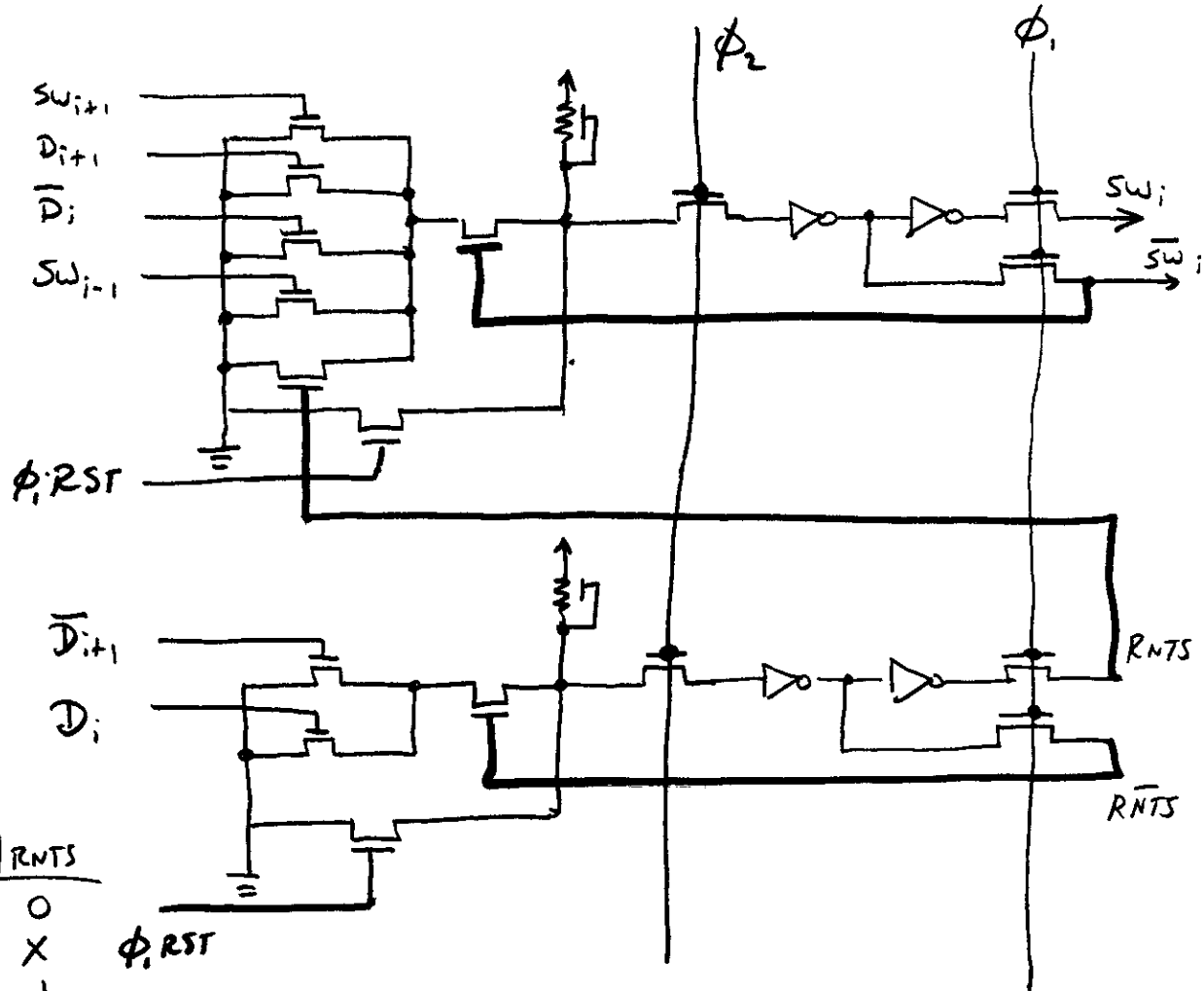
A: NOT YET SWAPPING/RESET
B: SWAPPING
C: REMEMBER NOT TO SWAP



i.e. If $SW_{i+1} + SW_{i-1} + RST = 1$, STAY IN (A)

If $SW_{i+1} + SW_{i-1} + RST = 0$, THEN IF $D_{i+1} = D_i$ STAY IN (A)
IF $D_{i+1} > D_i$ GO TO (B)
IF $D_{i+1} < D_i$ GO TO (C)

ONE POSSIBLE CIRCUIT IMPLEMENTING FSM:

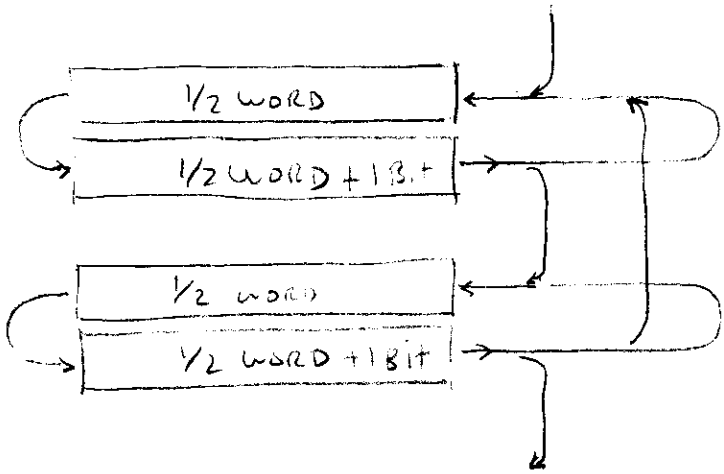


ST	SW	RNTS
A	0	0
B	1	X
C	0	1

- FOR HW #6, I'd LIKE TO FIND A GOOD, EASY TO LAYOUT CIRCUIT. NOTE THE PROBLEM OF THE PITCH OF ONE FSM VS THE SHIFT REGISTER.

  ALSO, DON'T WANT TO HAVE LONG LINES FOR FEEDBACK. MIGHT HAVE TO DRIVE THEM: SOLUTION:

  WRAP-AROUND EACH SR, BRINGING THE TWO ENDS TOGETHER. OF COURSE THIS STARTS TO KLUGE-UP THE SWAPPING OF FSM, BECAUSE NOW THE LOAD CIRCUITRY IS THERE ALSO:



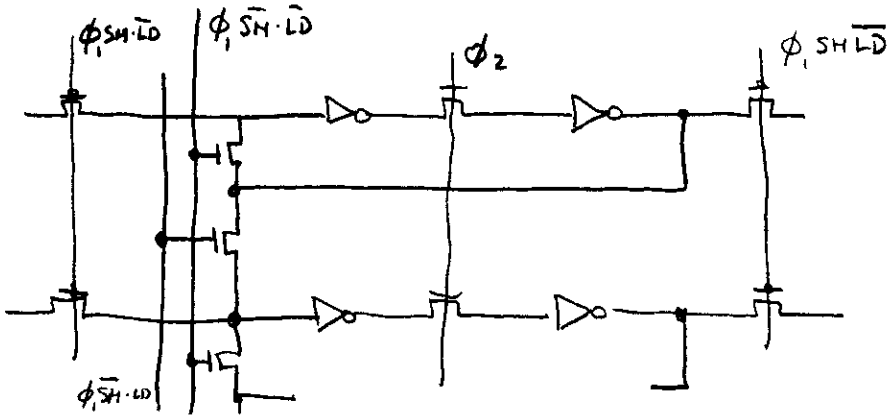
[CONSTRAINT: FSM; SHOULD LIE ON SR ROW PAIR PITCH.]

[MAYBE USE 8:1 MIN PULLDOWN SR HOWEVER]

- TO EXPLORE TENTATIVE STICK DIAGRAMS: STICK DIAGRAM ON A GRID ITS A GOOD WAY TO GET A FEELING FOR DENSITY. (LET RED OR GREEN RUN RIGHT BY BLUE, CTR.).

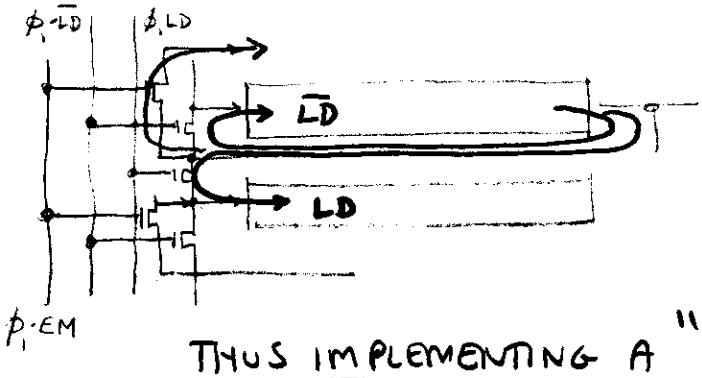
- FOR OPEN ENDED PROBLEM: TRY TO FIND A SIMPLER FSM; CIRCUIT, OR ONE THAT MIGHT LAYOUT MORE EASILY.

- OTHER EXTENSIONS: EASY TO MAKE //LOAD/EMPTY. PUT FOLLOWING INTO THE SR AT EVERY BIT:



[THIS MAY SPEED UP LOAD/EMPTY, BUT REALLY KLUDGES UP THE SR ARRAY]

- NOTE ALSO THAT IF WE ADD ANOTHER PATH AT THE "LEFT" (CONCEPTUALLY) AS FOLLOWS:



THUS IMPLEMENTING A "SORTING STACK"

- IN THIS CASE WE NEED TO HAVE A ZERO FILL-IN AT THE BOTTOM. NOTE THAT WE CAN BE SORTING AS WE FILL. THUS THE DATA CAN BE WITHDRAWN AS SOON AS THE STACK IS FULL!

- THUS THE STACK & TOP/BOTTOM SORTER BOTH HAVE A REP TIME OF 2N WORD SHIFTS. BUT THE STACK HAS A DELAY OF ONLY 2N WHILE THE TOP-BOTTOM HAS A DELAY OF 3N.

> ANOTHER SYSTEM LAYOUT PROBLEM WITH THE SORTER:

WE'D LIKE TO MAKE REALLY BIG SORTERS.

BUT THE THING IS TALL & SKINNY. HOW DO WE SNAKE IT AROUND ON THE CHIP? HOW DO WE ROUTE POWER AND GROUND? ETC. What about MULTIPLE CHIP SORTERS?

> WE'LL COME BACK TO THE SORTER IN AWHILE!