

Design of Digital Circuits and Systems

Algorithms to Hardware I

Instructor: Justin Hsia

Teaching Assistants:

Colton Harris

Deepti Anoop

Gayathri Vadhyan

Jared Yoder

Lancelot Wathieu

Matthew Hung

Relevant Course Information

- ❖ Lab 3 reports due Friday (4/26)
- ❖ Lab 4 due next Friday (5/3)
- ❖ hw4 due on Monday (4/29)
- ❖ Anonymous mid-quarter survey on Canvas (due 4/29)
- ❖ Quiz 3 (ASM, ASMD) next Thursday (5/2)

Arithmetic Mean

- ❖ Design a sequential circuit that computes the mean M of k n -bit numbers stored in registers

\nearrow \nearrow width
 ■ e.g., accessing a RAM or register file with k addresses

- To save on hardware, you can only use one n -bit adder and have a single read port RAM

- ❖ Algorithm Pseudocode:

① $S = 0$
 for $i = 0$ to $k-1$
 $S = S + R[i]$
 end for
 $M = S/k$

② $S = 0$
 for $i = k-1$ to 0
 $S = S + R[i]$
 end for
 $M = S/k$

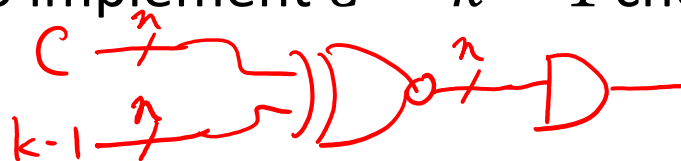
③ $M = 0$
 for $i = k-1$ to 0
 $M = M + R[i]/k$
 end for

\nearrow we'll look at #2, but these are all valid

Aside: Counter Variable

- ❖ Many sequential hardware algorithms utilize counters
- ❖ If both work, is there a preference?

- How to implement $C = k - 1$ check?

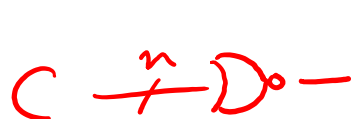


1 if inputs are the same

a	b	xnor
0	0	1
0	1	0
1	0	0
1	1	1

or $C - (k-1) = 0$ (subtractor + nor)

- How to implement $C = 0$ check?



a	b	nor
0	0	1
0	1	0
1	0	0
1	1	0

← only 1 if all inputs are 0

Arithmetic Mean Specification

❖ Datapath

- A k -address *register file* (only using `r_addr` and `r_data`)
- Reg file address stored in $\lceil \log_2(k) \rceil$ *down-counter* $A(i)$
- Sum stored in register S
- An n -bit *divider* circuit, as discussed last lecture

❖ Control

- Inputs *Start* and *Reset*, outputs *Ready* and *Done*
- Status signals: A_zero , $Div_done(?)$
- Control signals: $Load_regs$, Add , $Divide$, $Decr-A$

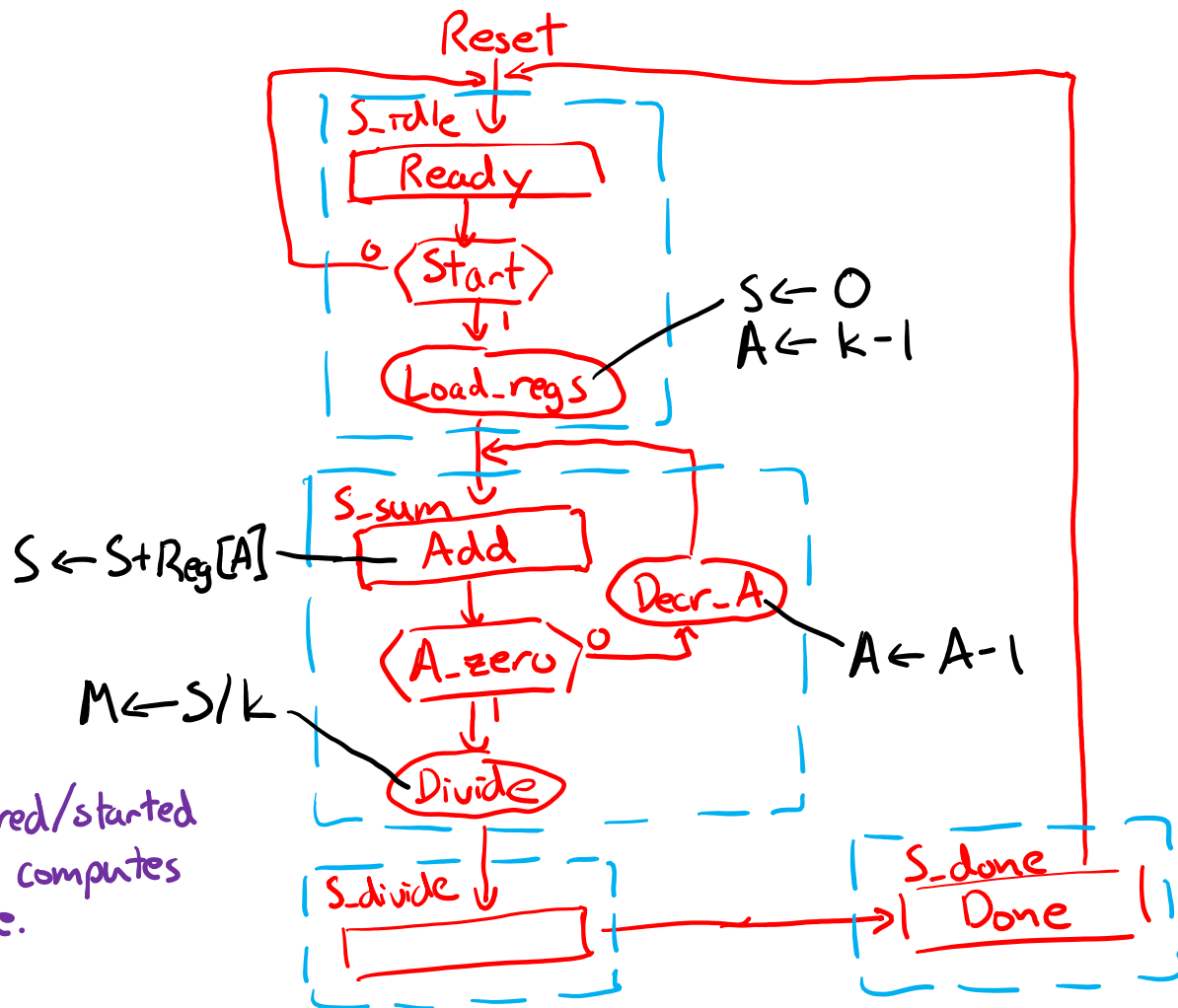
anything related
to control
flow

Arithmetic Mean (ASMD Chart, Initial)

- For now, ignore the details of the divider circuit

$S = 0$
 for $A = k-1$ to 0
 $S = S + \text{Reg}[A]$
 endfor
 $M = S/k$

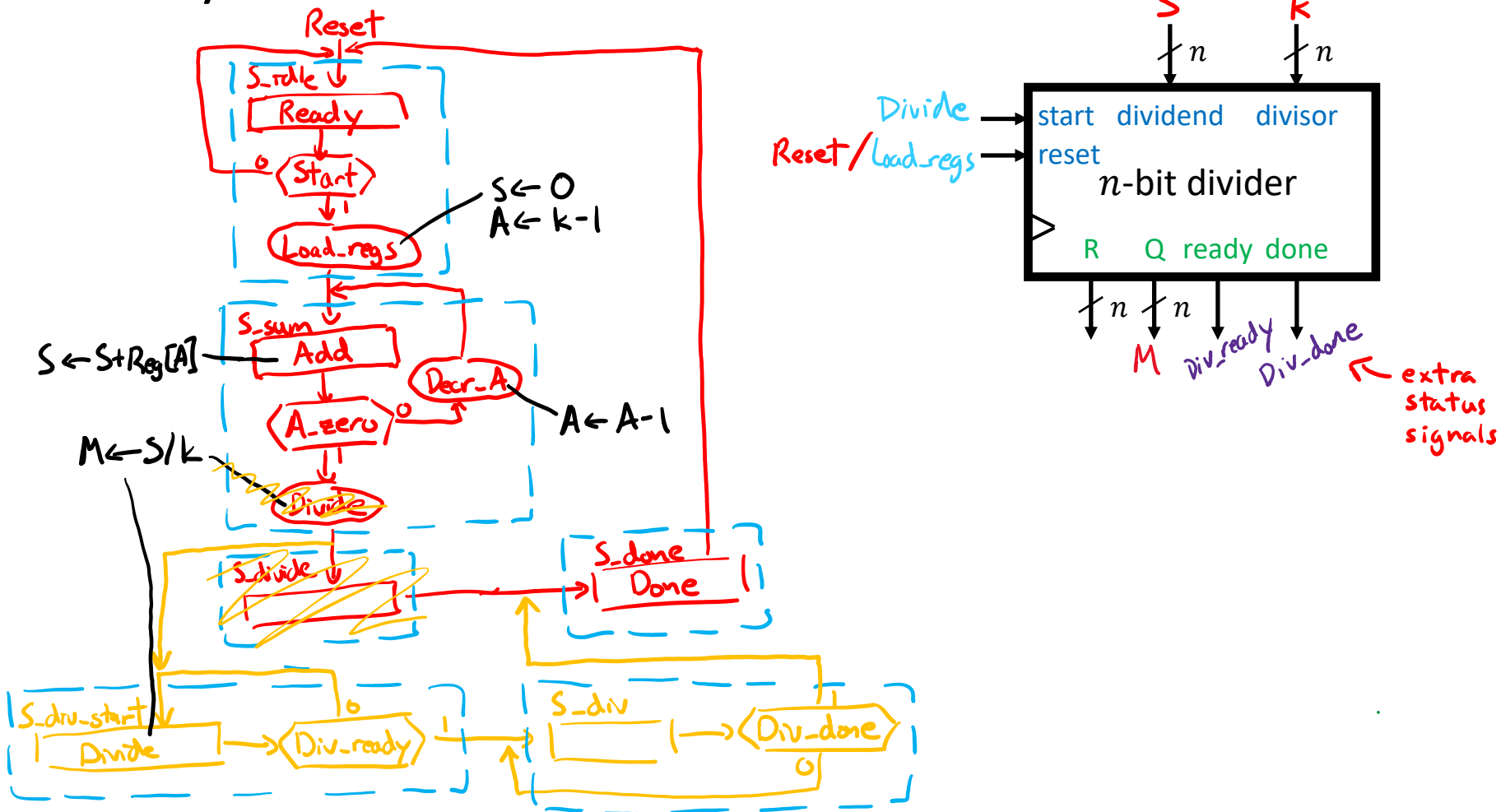
take RTL operations from pseudocode!



Note: the division is triggered/started exiting S_sum, but actually computes during the S_divide state.

Arithmetic Mean (ASMD Chart)

- Fix your ASMD chart based on the divider circuit:



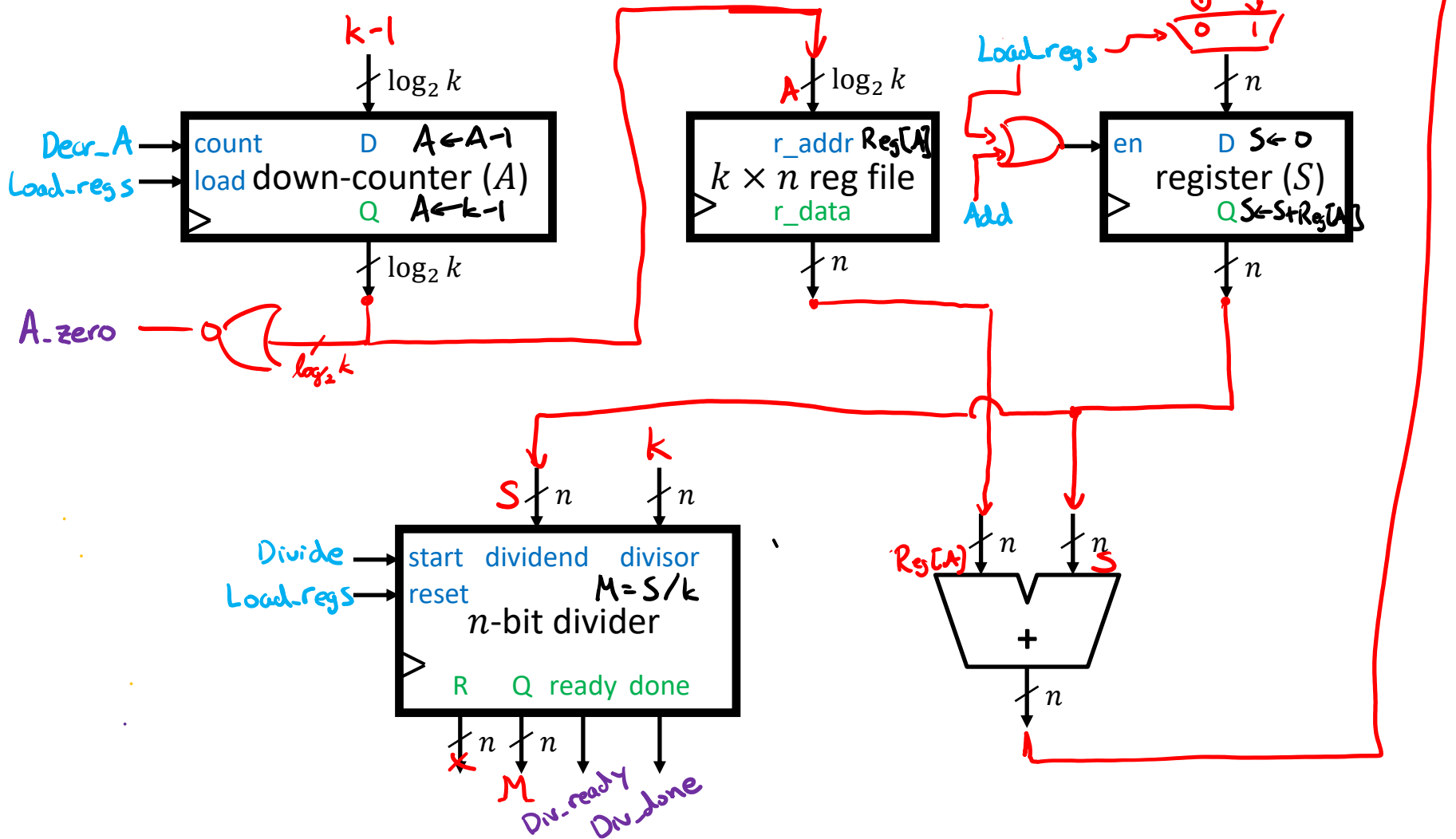
Arithmetic Mean Datapath

Control Signals:

- Load_regs
- Add
- Divide
- Decr-A

Status Signals:

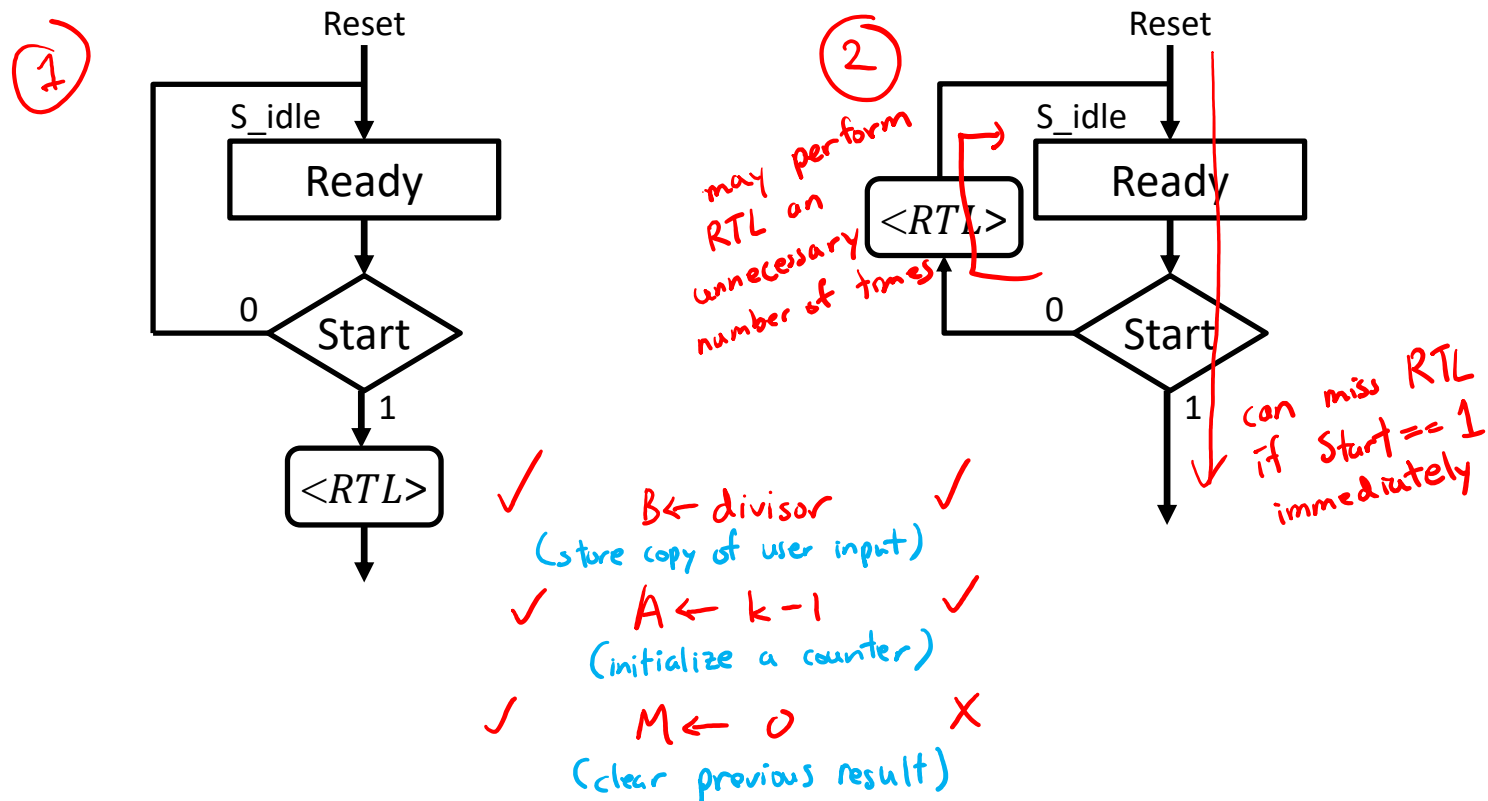
- A_zero
- Div_ready
- Div_done



Technology Break

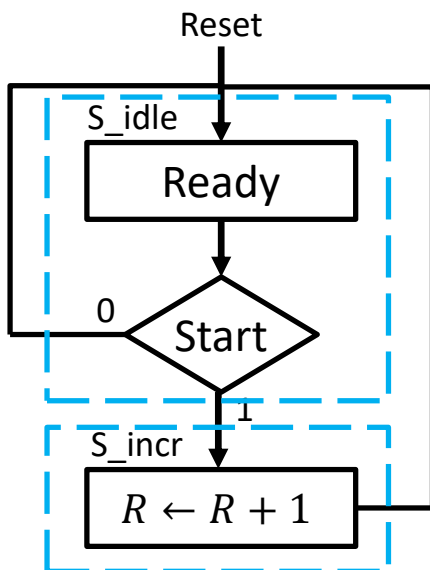
Aside: Load Loops

- ❖ For *some* initialization operations, you can get equivalent behavior from either the (1) outgoing edge or the (2) looping edge:

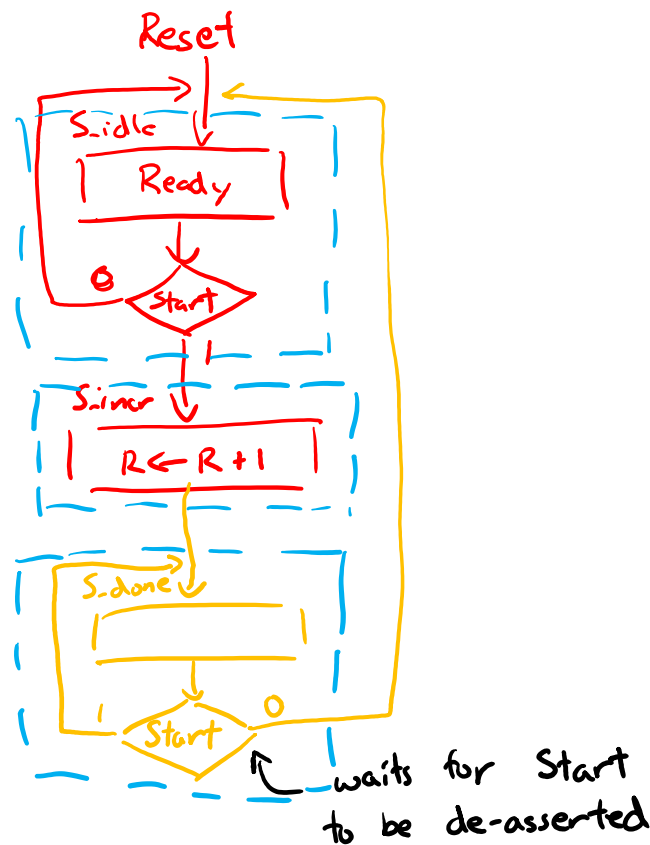


Aside: Start Loops

- ❖ What happens if we forget to de-assert *Start*?



- ❖ Fix:



Sorting Algorithm

- ❖ Design a circuit to sort k n -bit numbers stored in a set of registers in ascending order

Algorithm:

```

for i = 0 to k-2 do
  A = Reg[i]
  for j = i+1 to k-1 do
    B = Reg[j]
    if B < A then
      Reg[i] = B
      Reg[j] = A
      A = Reg[i]
    endif
  endfor
endfor
    
```

Example ($k = 4$): i to 2, j to 3

i	j	A	B	R[0]	R[1]	R[2]	R[3]
0	1	3	7 _x	3	7	1	0
0	2	3	1 _✓	3	7	1	0
0	3	1	0 _✓	1	7	3	0
1	2 _(i++)	7	3 _✓	0	7	3	1
1	3	3	1 _✓	0	3	7	1
2	3 _(i++)	7	3 _✓	0	1	7	3
				0	1	3	7

Sorting Algorithm Specification

❖ Datapath

- A k -address *register file* (assume only 1 port)
- Two $\lceil \log_2(k) \rceil$ *up-counters* i and j
- Two registers A and B
- An n -bit *comparator* circuit to check for $B < A$

❖ Control

- Inputs *Start* and *Reset*, outputs *Ready* and *Done*

- Status signals: i_done, j_done, B_lt_A

- Control signals: $Init_i, Init_j, Incr_i, Incr_j$

$Load_A, Load_B, Store_A, Store_B$
 $A = Reg[i], B = Reg[j], Reg[j] = A, Reg[i] = B$

Sorting Algorithm Specification

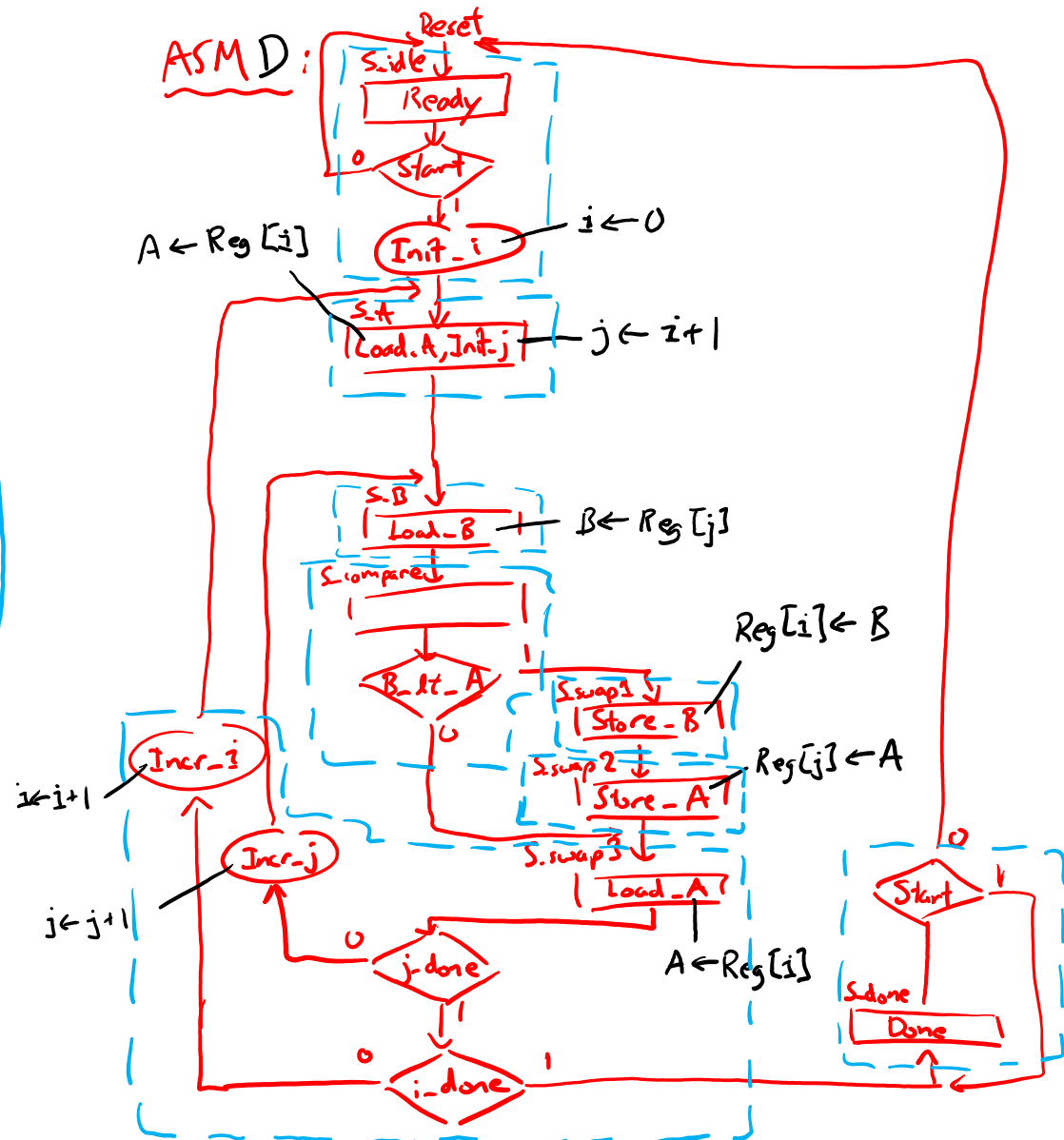
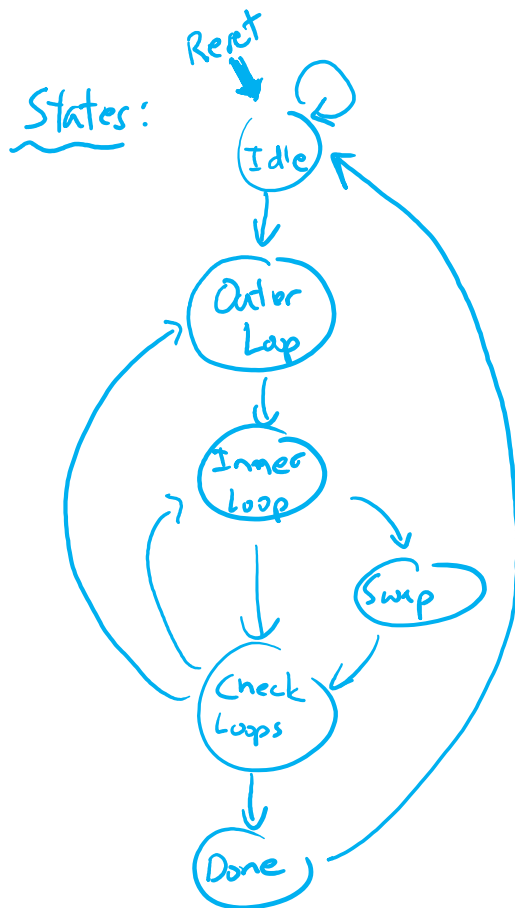
❖ Datapath

- A k -address *register file* (assume only 1 port)
- Two $\lceil \log_2(k) \rceil$ *up-counters* i and j
- Two registers A and B
- An n -bit *comparator* circuit to check for $B < A$

❖ Timing Notes:

- RTL operations in a state occur on the *next* clock trigger
- Can $i \leftarrow \underline{x}$ and $A \leftarrow \text{Reg}[i]$ be done simultaneously?
 \swarrow new
No, need a cycle for i to update to r-addr port of reg file
- Can $\text{Reg}[i] \leftarrow B$ and $\text{Reg}[j] \leftarrow A$ be done simultaneously?
 No, only 1 write port
- Swap operations *must* be done sequentially

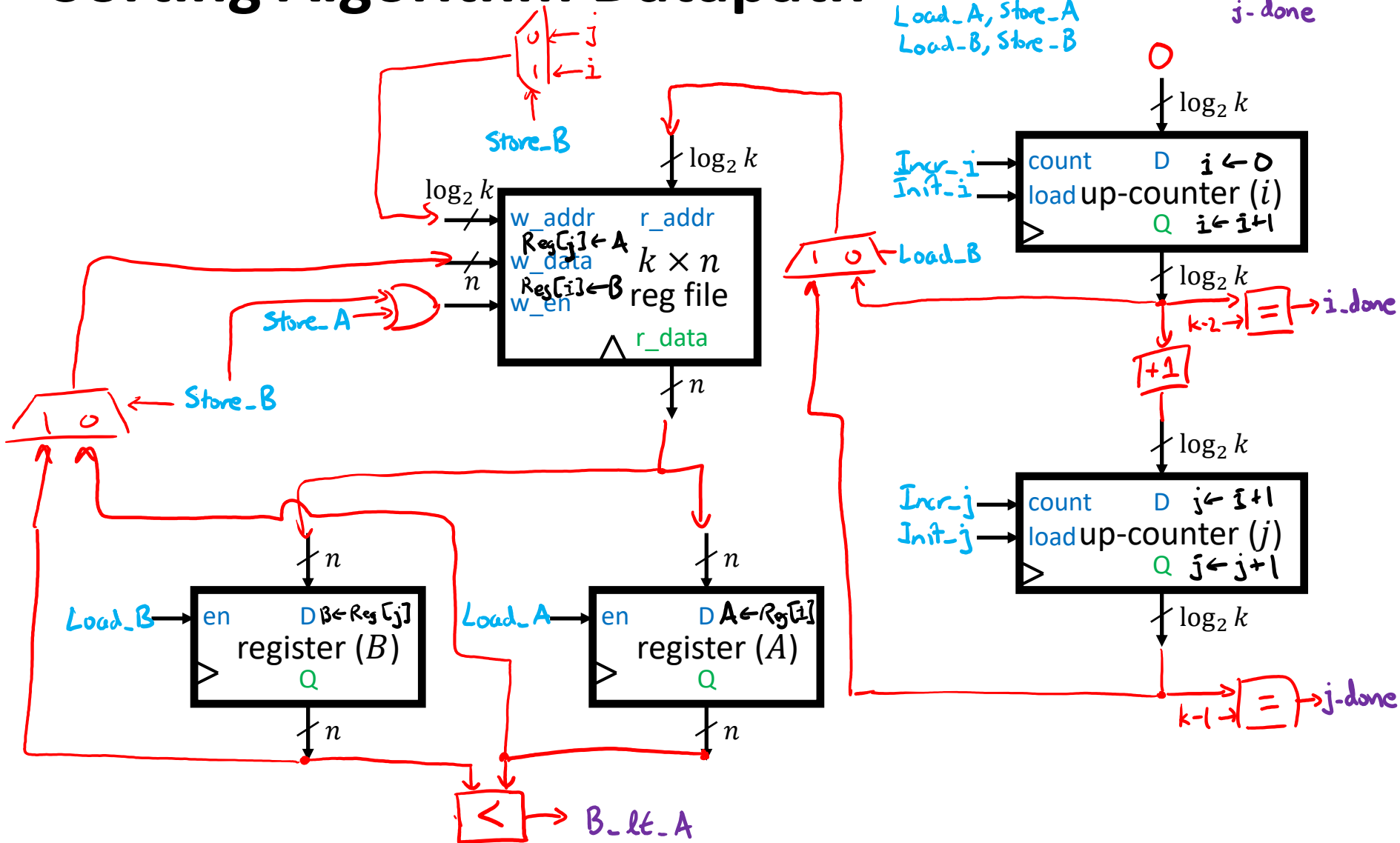
Sorting Algorithm (ASMD Chart)



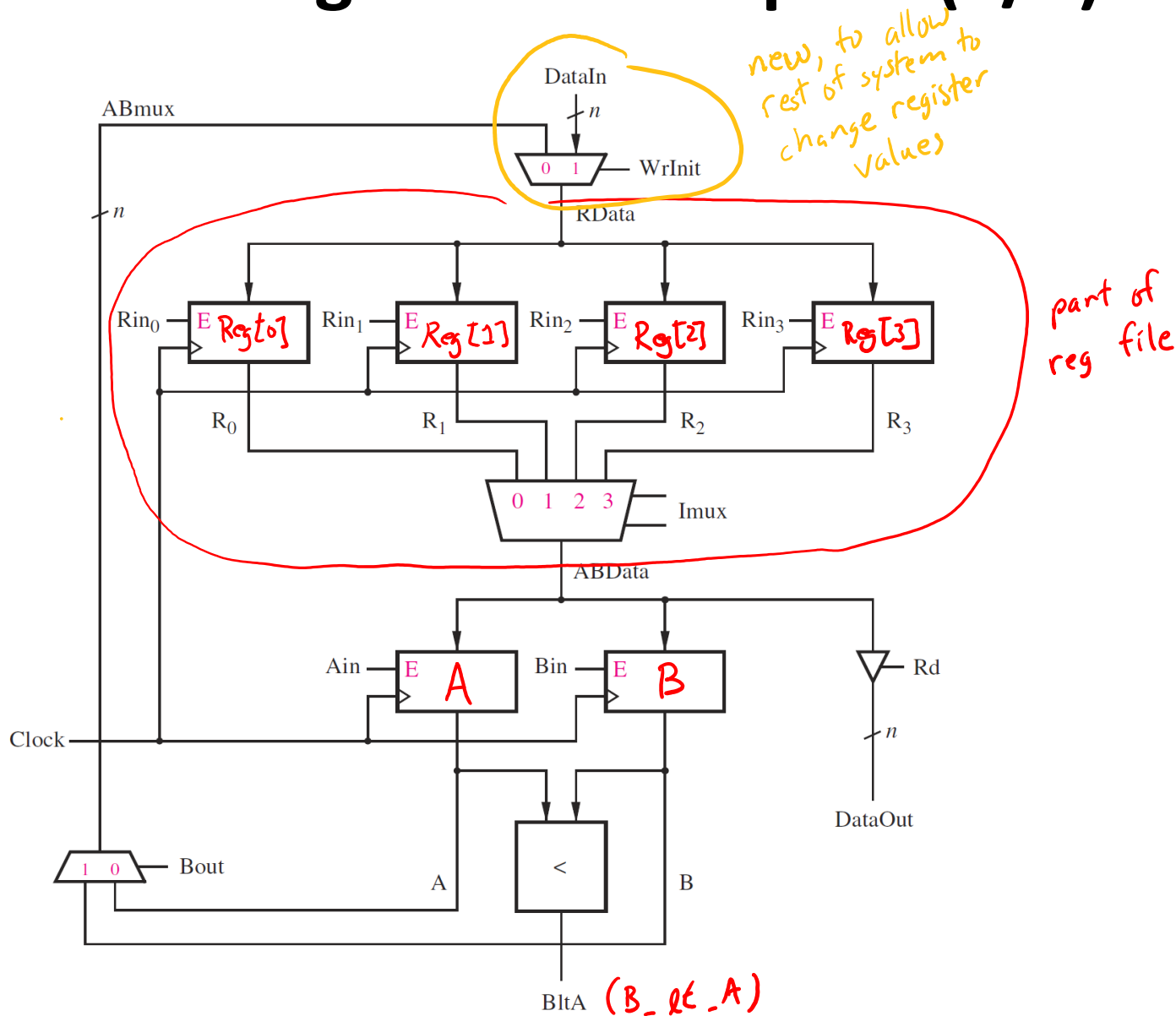
Sorting Algorithm Datapath

Control Signals:
 Init-*i*, Incr-*i*
 Init-*j*, Incr-*j*
 Load-*A*, Store-*A*
 Load-*B*, Store-*B*

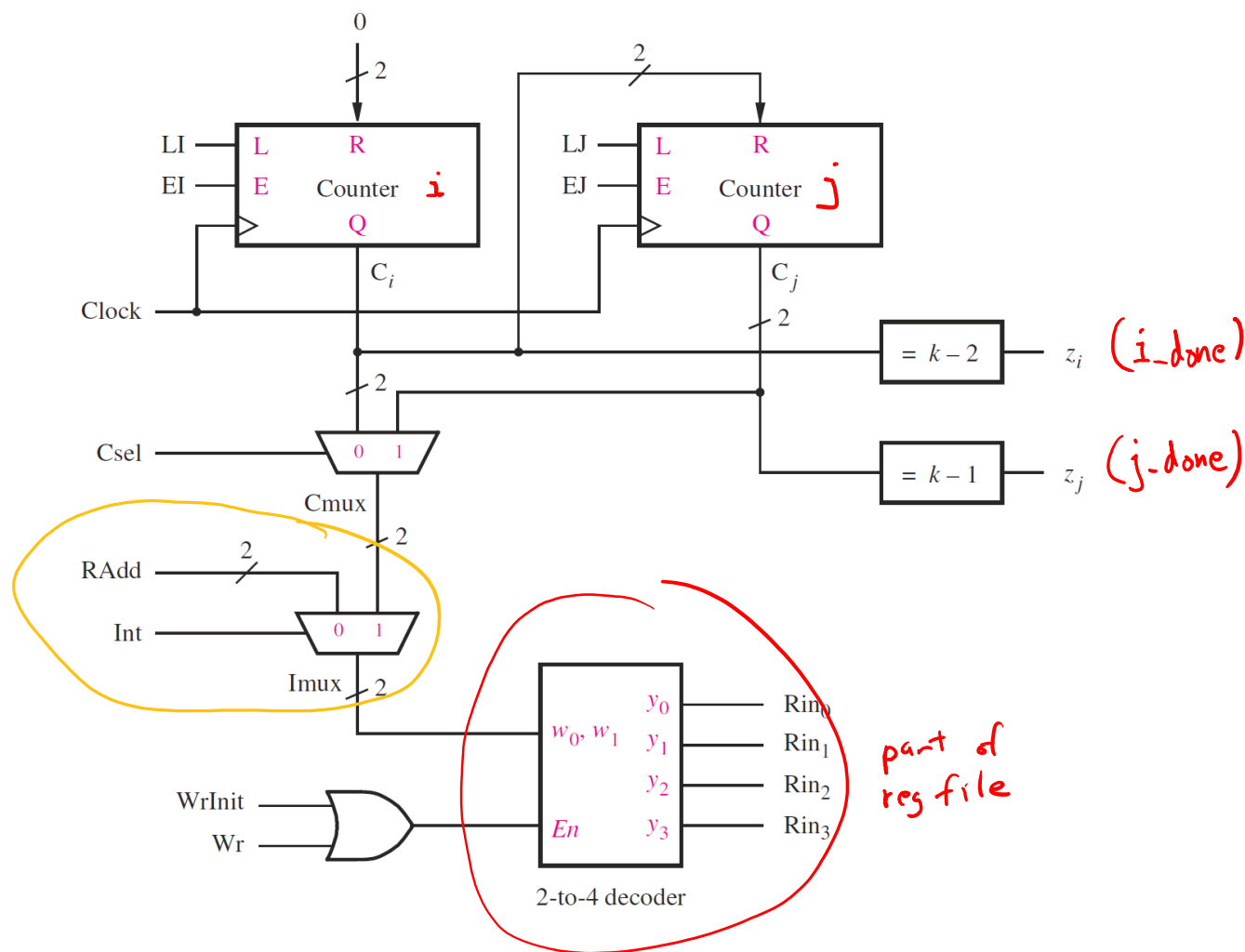
Status Signals:
 B-*lt*, A
i-done
j-done



Alternate Sort Algorithm Datapath (1/2)



Alternate Sort Algorithm Datapath (2/2)



new, to allow rest of system to change register values

part of reg file

Lab 4 Preview: Binary Search

- ❖ Design a circuit that searches a *sorted* array for a given value by checking the middle element of the remaining portion of the array we would expect to find the given number:

```
L = 0
R = n - 1
while L <= R do
    m = floor((L + R)/2)
    if A[m] < T then
        L = m + 1
    else if A[m] > T then
        R = m - 1
    else
        return m
    endif
endwhile
return unsuccessful
```