

## UNIT 10B

### Concurrency: Pipelining

15110 Principles of Computing, Carnegie  
Mellon University - CORTINA

1

## Pipelining

- Pipelining is similar to an assembly line.
  - Instead of completing one computation before starting another, each computation is split into simpler sub-steps, and computations are started as others are in progress.

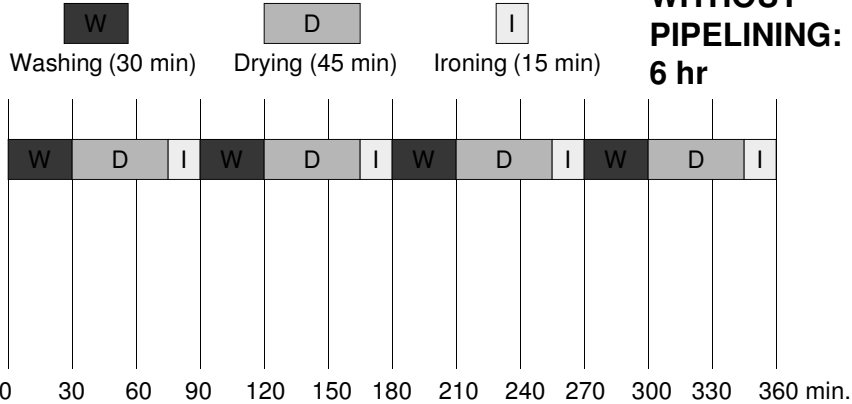


15110 Principles of Computing, Carnegie  
Mellon University - CORTINA

2

## Laundry Without Pipelining

Washing, Drying and Ironing four loads of laundry.

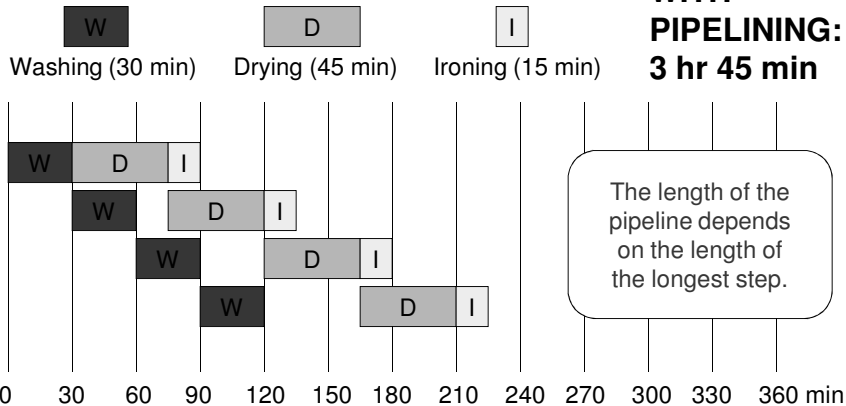


15110 Principles of Computing, Carnegie Mellon University - CORTINA

3

## Laundry With Pipelining

Washing, Drying and Ironing four loads of laundry.



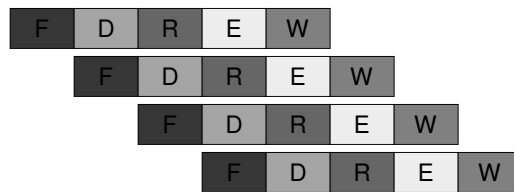
The length of the pipeline depends on the length of the longest step.

15110 Principles of Computing, Carnegie Mellon University - CORTINA

4

## Pipelining in Computing

- Fetch instruction from memory
- Decode the instruction
- Read data from registers
- Execute the instruction
- Write the result into a register



15110 Principles of Computing, Carnegie Mellon University - CORTINA

5

## Dealing with Dependencies

ADD R3, R2, R1 ← "Add the contents of R1 and R2 and store the results in R3."  
 ADD R5, R4, R3 ← This instruction depends on the result of the previous instruction. (This will hold up the pipeline.)  
 ADD R8, R7, R6  
 ADD R11, R10, R9

ADD R3, R2, R1  
 ADD R8, R7, R6  
 ADD R11, R10, R9  
 ADD R5, R4, R3

Reorder the instructions to minimize the delay on the pipeline due to the dependency, if possible.

15110 Principles of Computing, Carnegie Mellon University - CORTINA

6

## Dealing with Dependencies

A: ADD R3, R2, R1

SUB R6, R5, R4

BEQ R6, R3, A

MOV R2, R1

"Branch to label A if R3 = R6."

The BEQ instruction will stall in the pipeline since the final values of R3 and R6 are not known yet.

Possible solutions:

1. Assume the branch occurs. If we find later that R3 is not equal to R6, clear the pipeline and begin computing with the MOV instruction.
2. Start decoding the ADD and MOV instructions. When we know if R3 is equal to R6 or not, send the appropriate instructions into the pipeline for completion.

## Matrix Multiplication

	hw	paper	exam1	exam2	exam3	final		weight		average
student1	95	90	93	91	85	92			student1	
student2	73	80	75	63	79	75	hw	0.15	student2	
student3	85	73	80	85	88	91	paper	0.1	student3	
student4	50	65	50	60	56	47	exam1	0.15	student4	
student5	100	95	98	96	96	90	exam2	0.15	student5	
student6	75	75	75	75	75	75	exam3	0.15	student6	
student7	90	80	80	90	100	100	final	0.3	student7	
student8	88	80	80	70	60	55			student8	

# Matrix Multiplication

$$0 + 95 \cdot 0.15 + 90 \cdot 0.1 + 93 \cdot 0.15 + 91 \cdot 0.15 + 85 \cdot 0.15 + 92 \cdot 0.3 = 91.2$$

	hw	paper	exam1	exam2	exam3	final		weight		average
student1	95	90	93	91	85	92		0.15	student1	91.2
student2	73	80	75	63	79	75	hw	0.1	student2	
student3	85	73	80	85	88	91	paper	0.15	student3	
student4	50	65	50	60	56	47	exam1	0.15	student4	
student5	100	95	98	96	96	90	exam2	0.15	student5	
student6	75	75	75	75	75	75	exam3	0.15	student6	
student7	90	80	80	90	100	100	final	0.3	student7	
student8	88	80	80	70	60	55			student8	

# Matrix Multiplication

$$0 + 73 \cdot 0.15 + 80 \cdot 0.1 + 75 \cdot 0.15 + 63 \cdot 0.15 + 79 \cdot 0.15 + 75 \cdot 0.3 = 74.0$$

	hw	paper	exam1	exam2	exam3	final		weight		average
student1	95	90	93	91	85	92		0.15	student1	91.2
student2	73	80	75	63	79	75	hw	0.1	student2	74.0
student3	85	73	80	85	88	91	paper	0.15	student3	
student4	50	65	50	60	56	47	exam1	0.15	student4	
student5	100	95	98	96	96	90	exam2	0.15	student5	
student6	75	75	75	75	75	75	exam3	0.15	student6	
student7	90	80	80	90	100	100	final	0.3	student7	
student8	88	80	80	70	60	55			student8	

# Matrix Multiplication

$$0 + 85 \cdot 0.15 + 73 \cdot 0.1 + 80 \cdot 0.15 + 85 \cdot 0.15 + 88 \cdot 0.15 + 91 \cdot 0.3 = 85.3$$

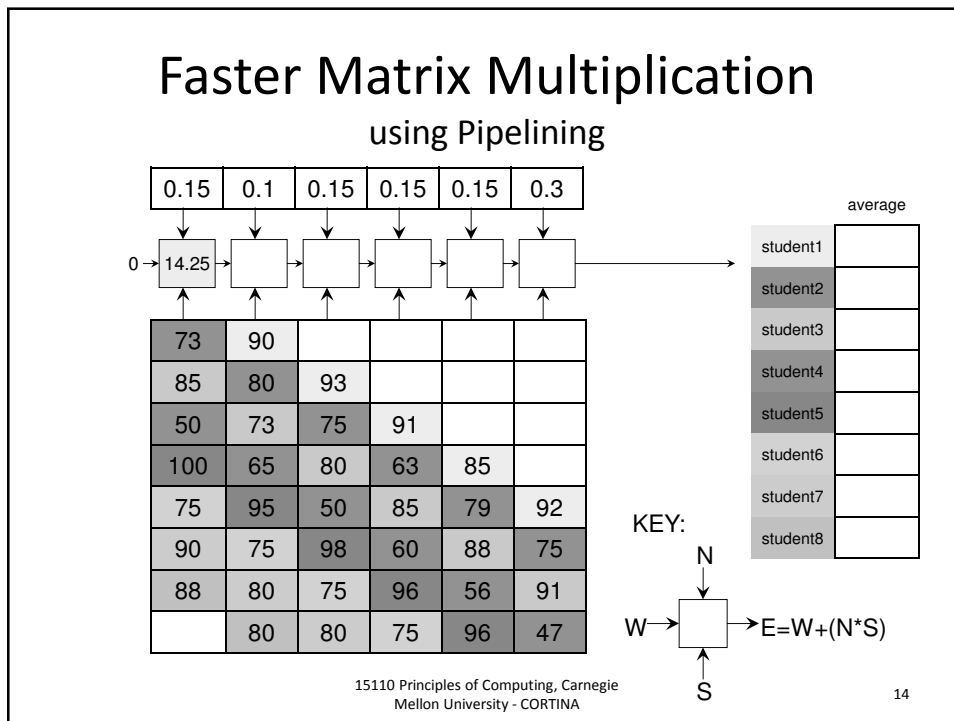
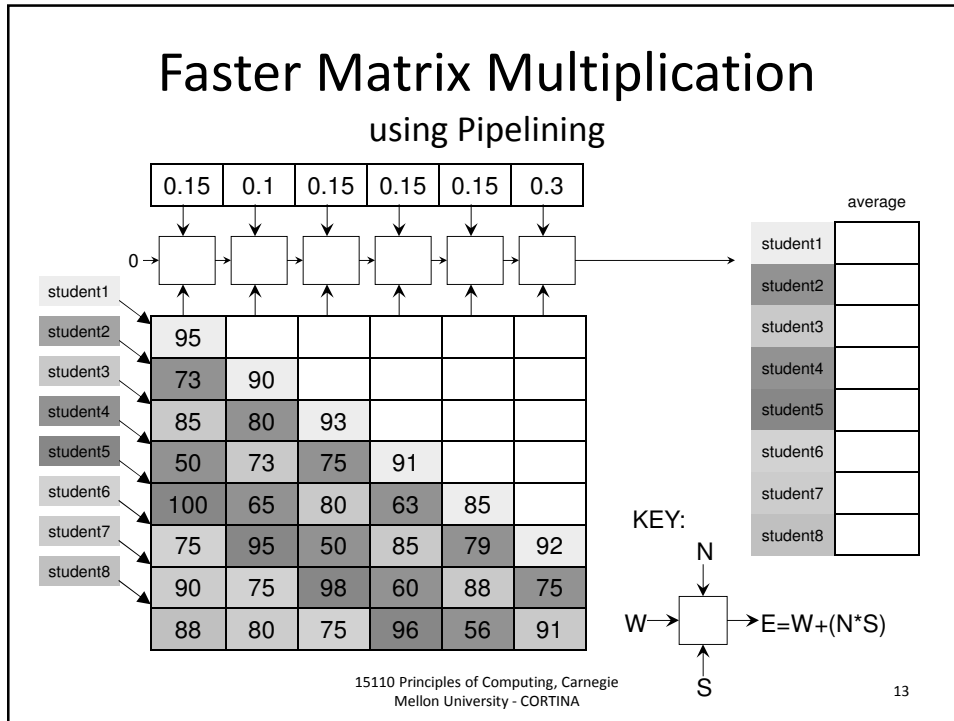
	hw	paper	exam1	exam2	exam3	final		weight		average
student1	95	90	93	91	85	92		0.15	student1	91.2
student2	73	80	75	63	79	75	hw	0.1	student2	74.0
student3	85	73	80	85	88	91	paper	0.15	student3	85.3
student4	50	65	50	60	56	47	exam1	0.15	student4	
student5	100	95	98	96	96	90	exam2	0.15	student5	
student6	75	75	75	75	75	75	exam3	0.15	student6	
student7	90	80	80	90	100	100	final	0.3	student7	
student8	88	80	80	70	60	55			student8	

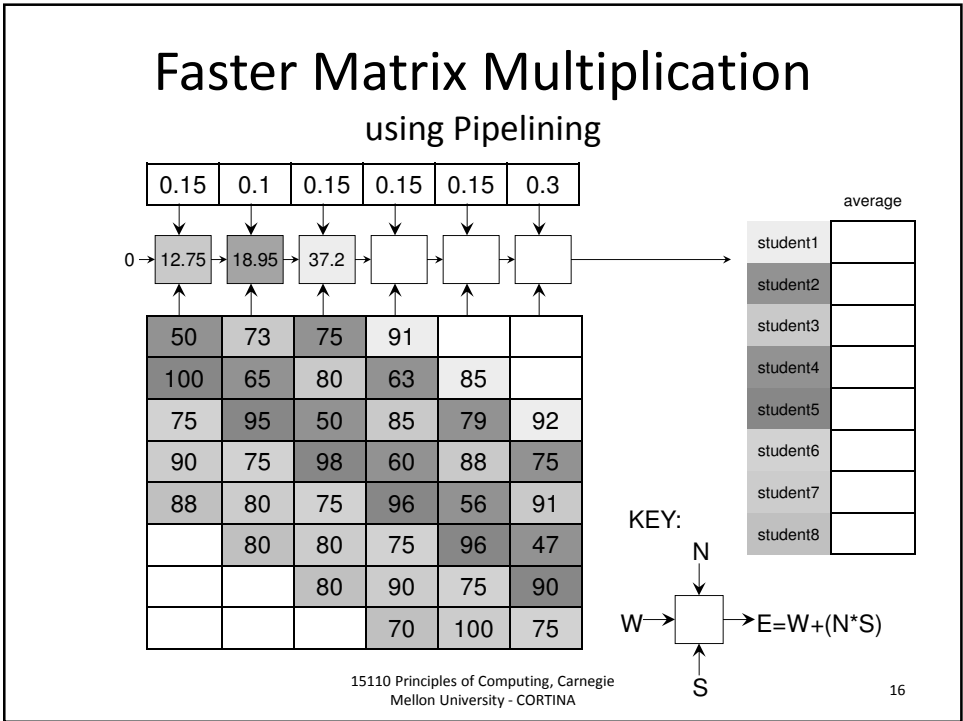
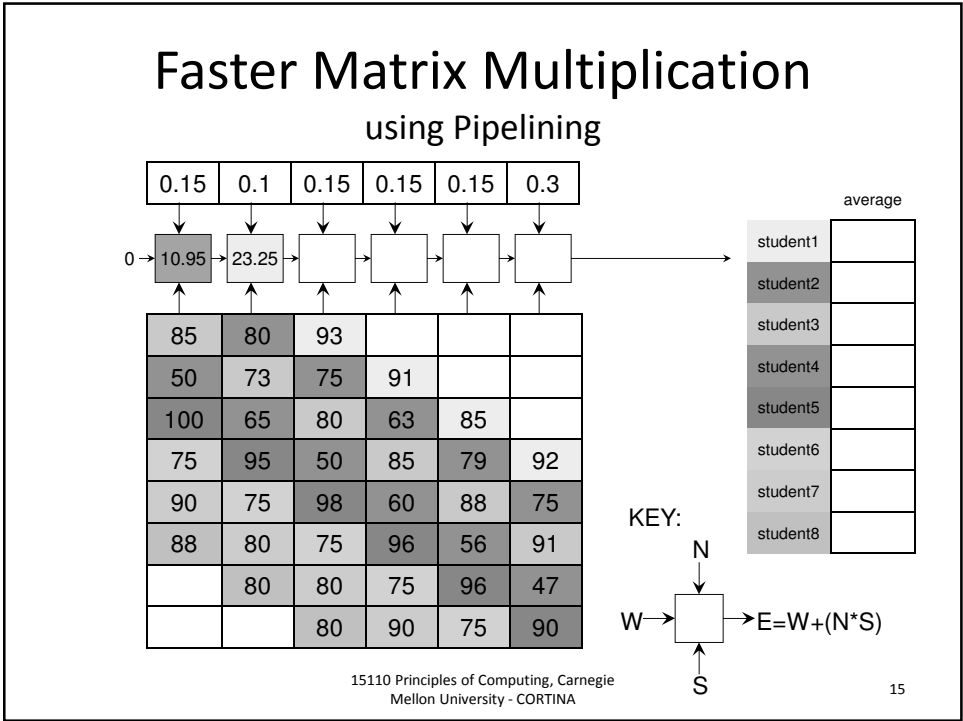
....and so on...

# Matrix Multiplication

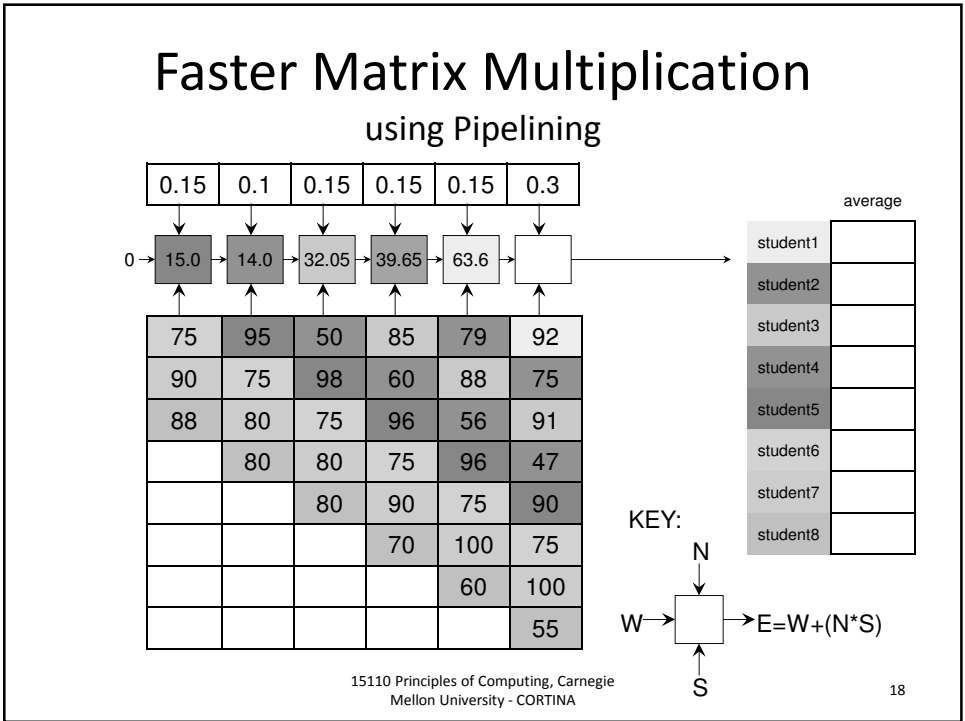
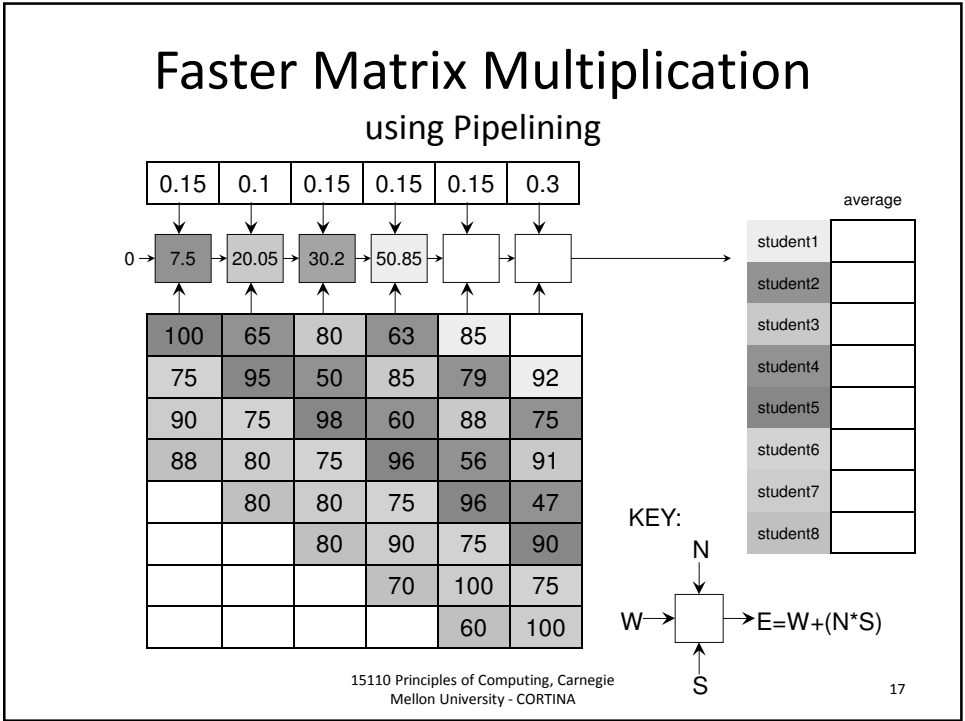
**If each multiply/add takes 1 time unit, this non-pipelined matrix multiplication takes 48 time units.**

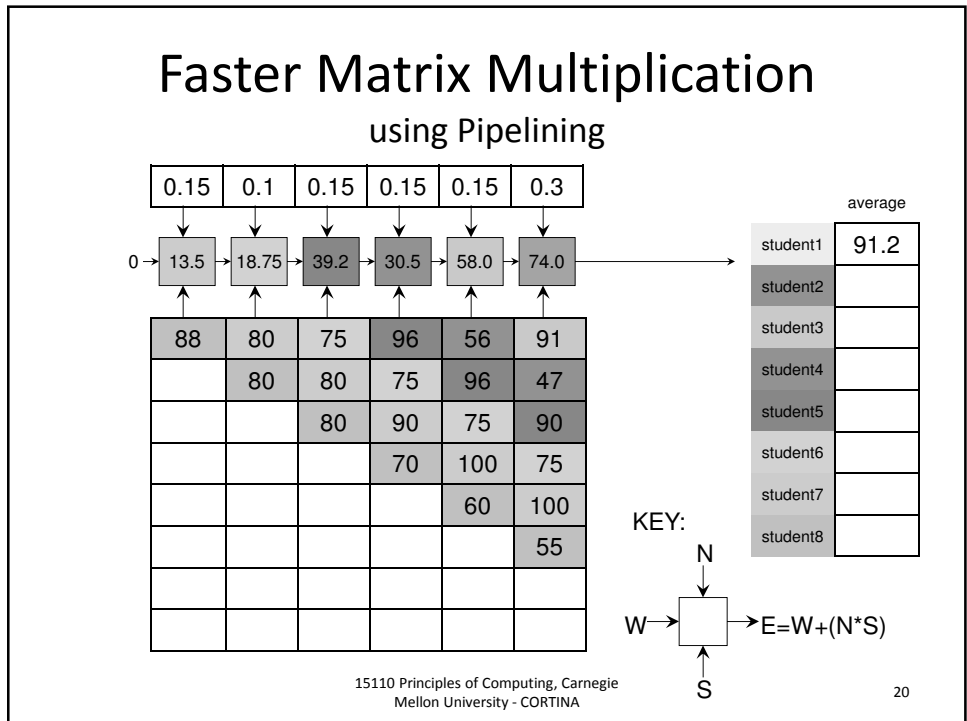
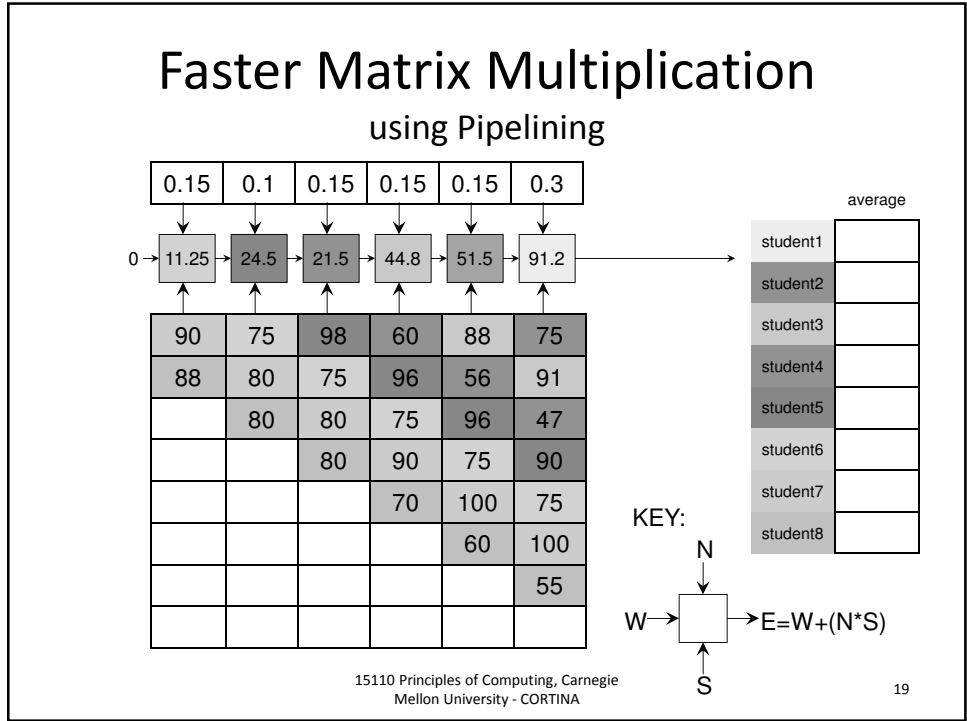
	hw	paper	exam1	exam2	exam3	final		weight		average
student1	95	90	93	91	85	92		0.15	student1	91.2
student2	73	80	75	63	79	75	hw	0.1	student2	74.0
student3	85	73	80	85	88	91	paper	0.15	student3	85.3
student4	50	65	50	60	56	47	exam1	0.15	student4	53.0
student5	100	95	98	96	96	90	exam2	0.15	student5	95.0
student6	75	75	75	75	75	75	exam3	0.15	student6	75.0
student7	90	80	80	90	100	100	final	0.3	student7	92.0
student8	88	80	80	70	60	55			student8	69.2

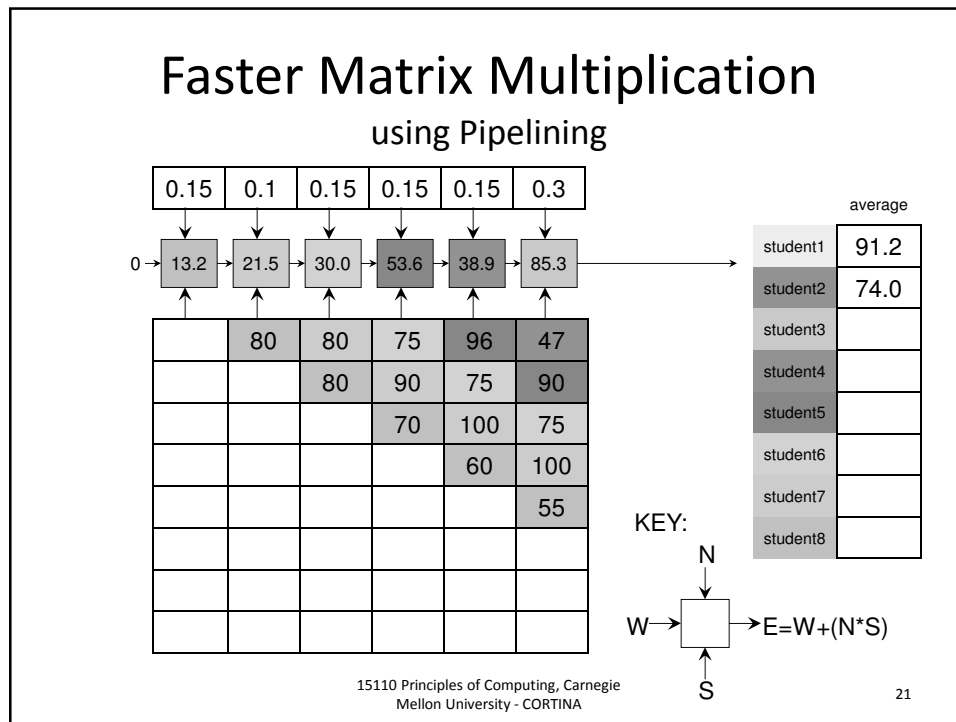




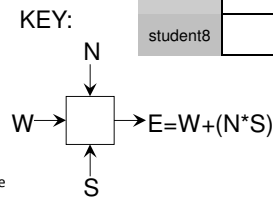




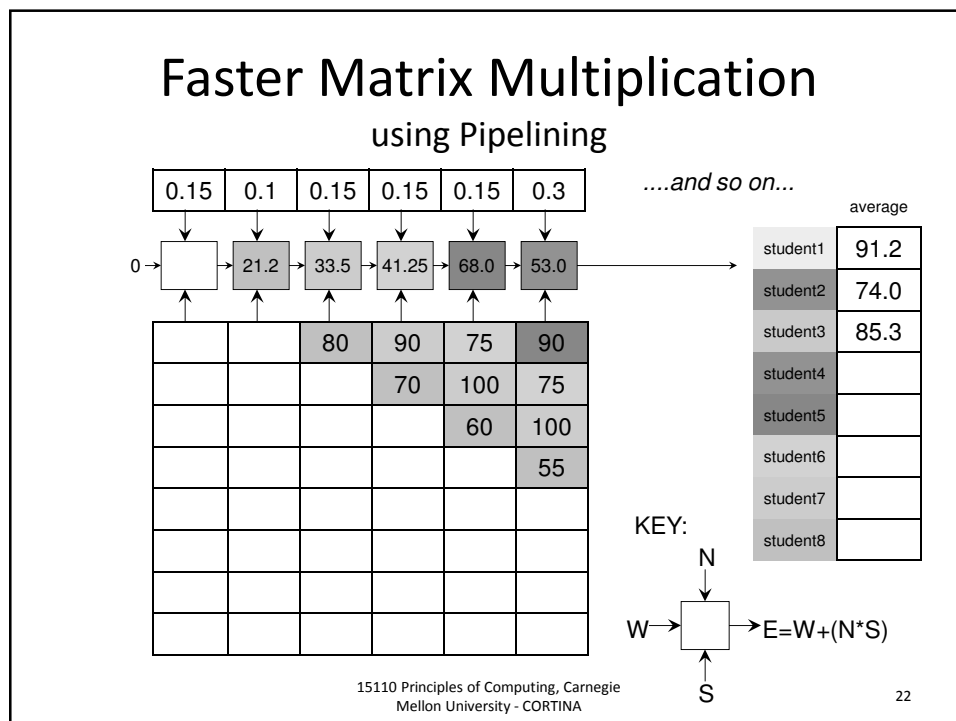




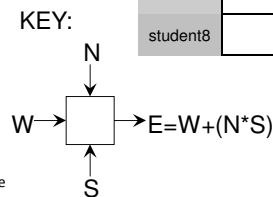
	average
student1	91.2
student2	74.0
student3	
student4	
student5	
student6	
student7	
student8	



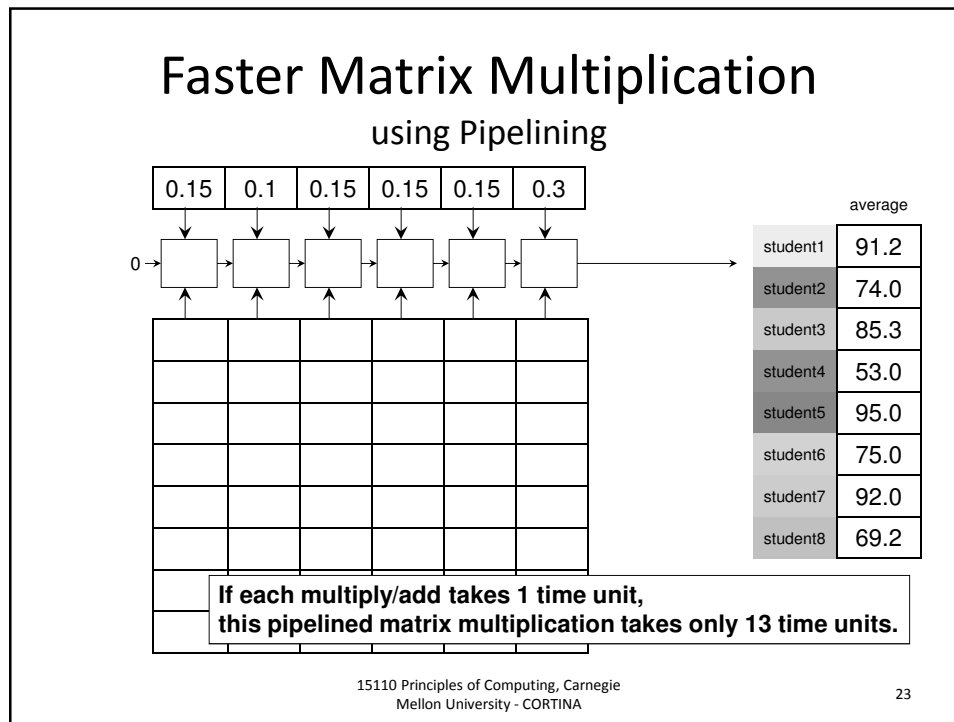
21



	average
student1	91.2
student2	74.0
student3	85.3
student4	
student5	
student6	
student7	
student8	



22



## Summary

- Pipelining allows us to perform operations concurrently to speed up our solutions.
- Pipelining requires additional support (hardware, facilities) in order to work successfully.