COL215 Digital Logic and System Design

Adders and Multipliers in FPGA

29 September 2017

FPGA Manufacturers

ALTERA®

First FPGA device in 1984
EPROM based

XILINX

Largest market share

First FPGA device in 1985
SRAM based

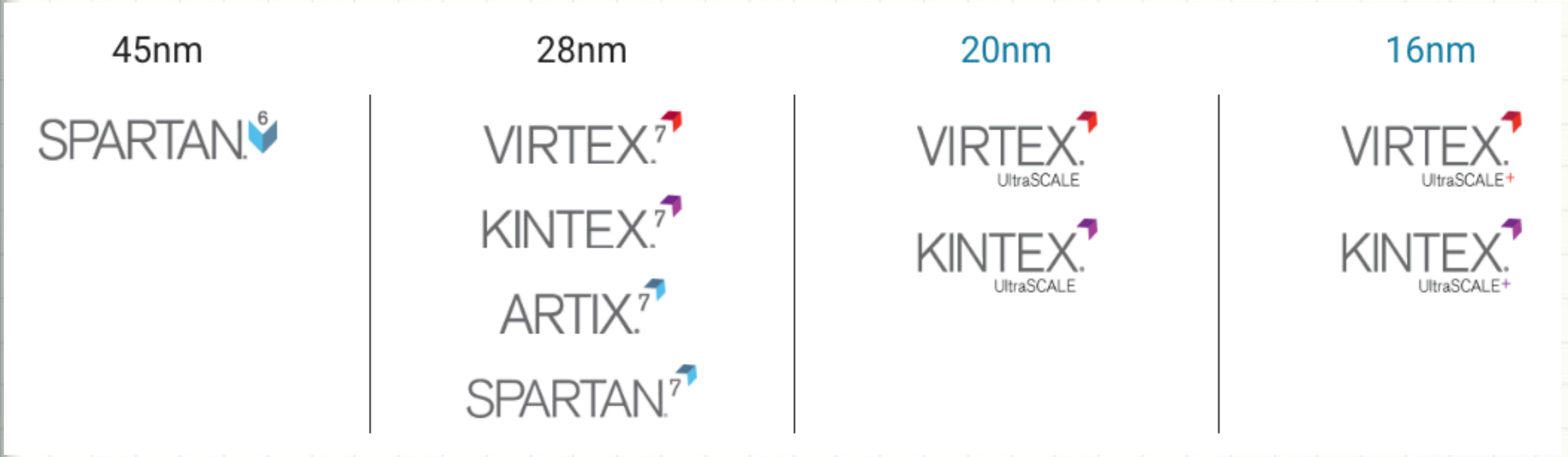
Actel → Microsemi®

Anti fuse based

LATTICE SEMICONDUCTOR

QuickLogic

Xilinx FPGA Portfolio

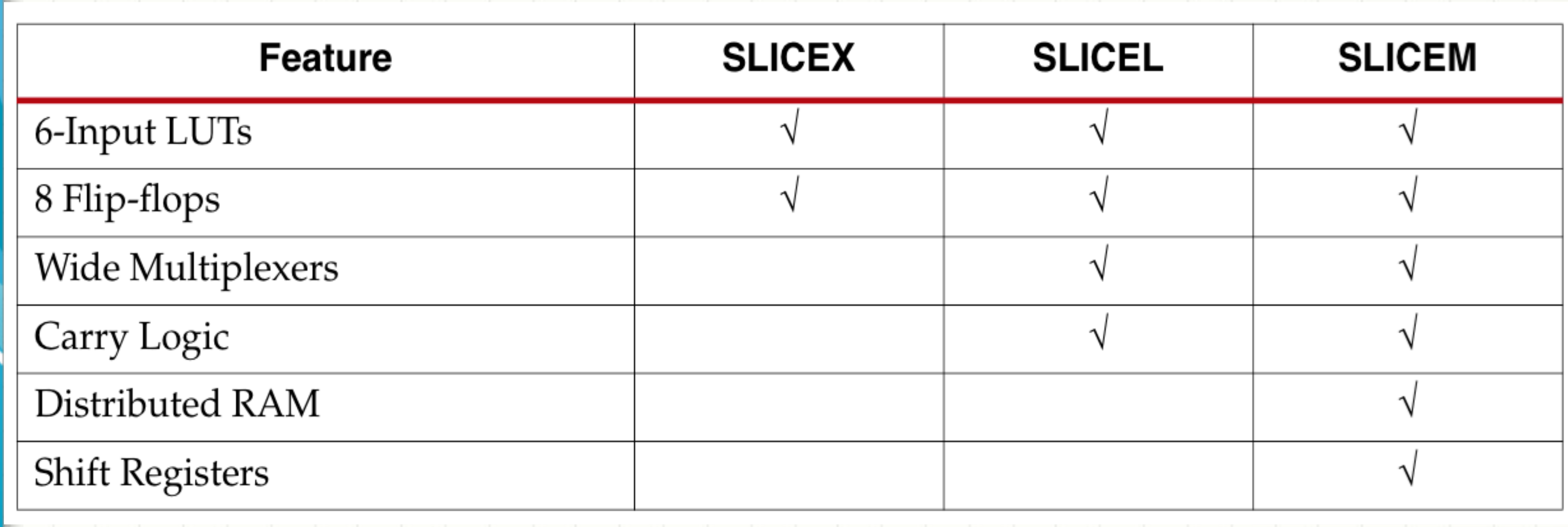


XC2064 :	64 CLBs,	56 IOBs,	18 MHz
XC7A35T-1CPG236C :	5200 slices,	236 pins,	464 MHz
	90 DSPs,	1800 KB RAM	
XC7VX1140T :	178,000 slices,	1100 pins,	540-740 MHz
	3,360 DSPs,	67,680 Kb RAM	

Xilinx Spartan 6 FPGA

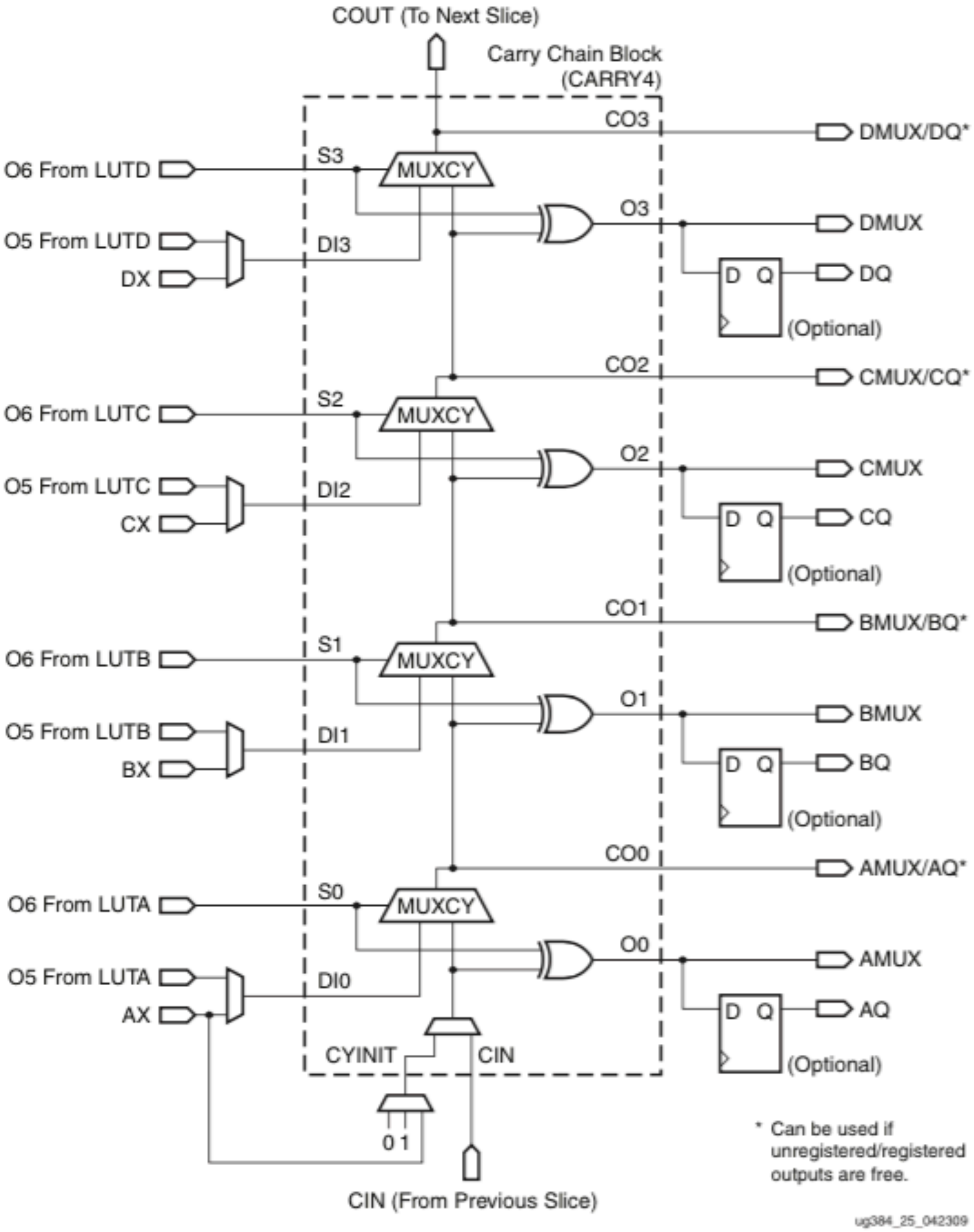
- Configurable Logic Blocks (CLBs)
- DSP48A1 slices
- Block RAMs
- DRAM controllers
- Clock Management Tiles (CMTs)
- Input/Output interfaces

Types of slices

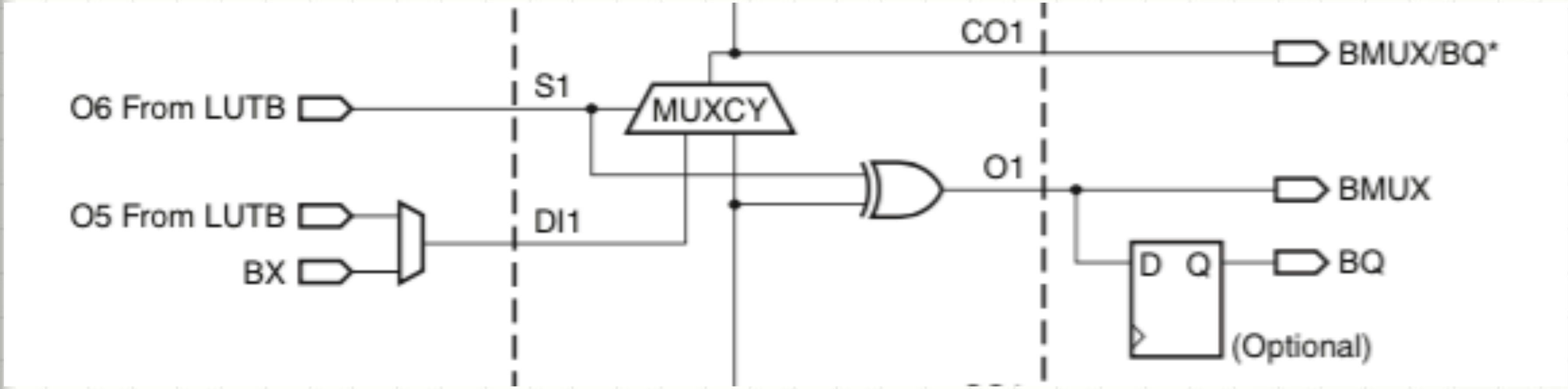


Feature	SLICEX	SLICEL	SLICEM
6-Input LUTs	√	√	√
8 Flip-flops	√	√	√
Wide Multiplexers		√	√
Carry Logic		√	√
Distributed RAM			√
Shift Registers			√

Fast Carry Chain



Fast Carry Chain



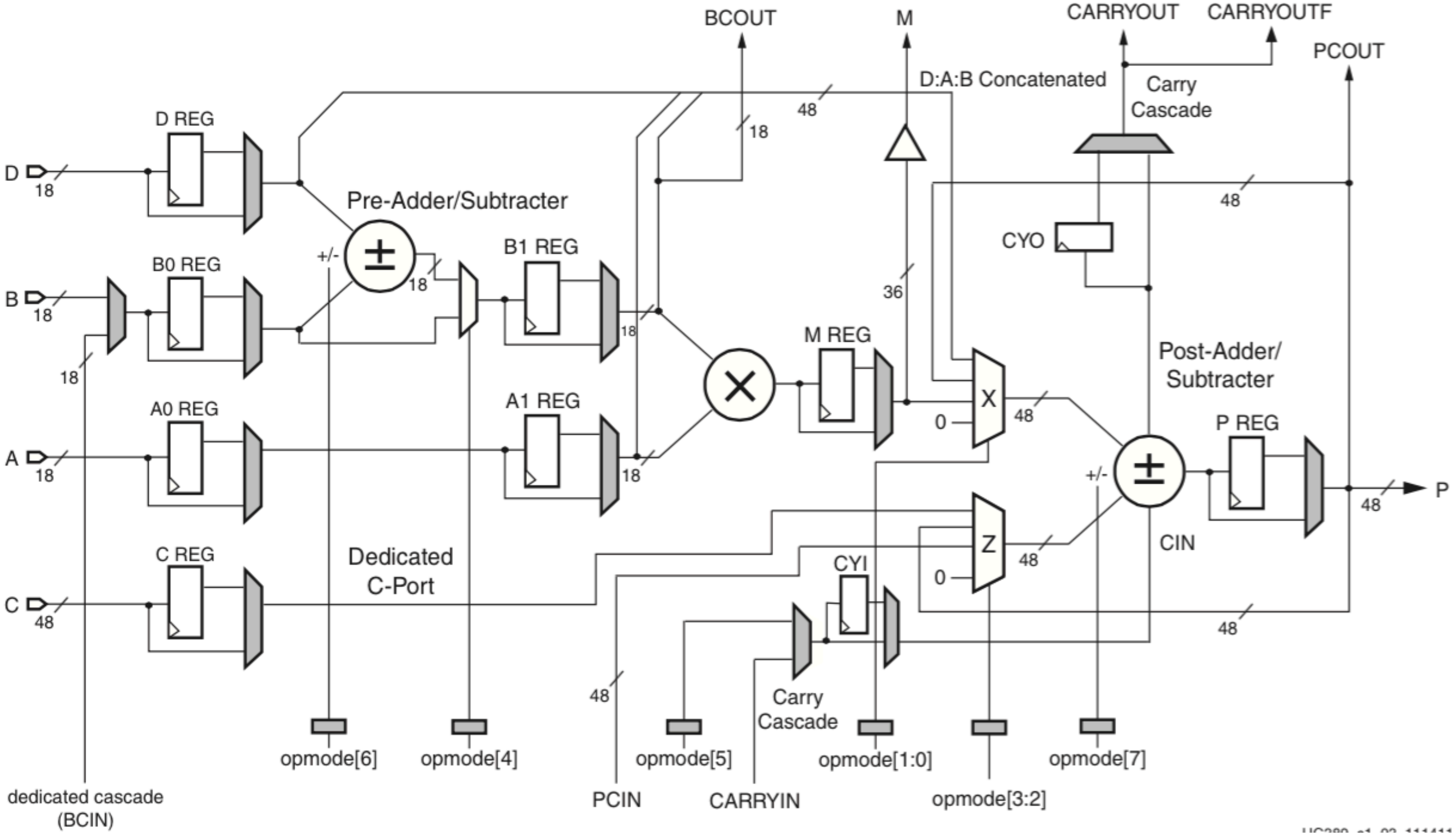
Xilinx Spartan 6 FPGA

- Configurable Logic Blocks (CLBs)
- DSP48A1 slices
- Block RAMs
- DRAM controllers
- Clock Management Tiles (CMTs)
- Input/Output interfaces

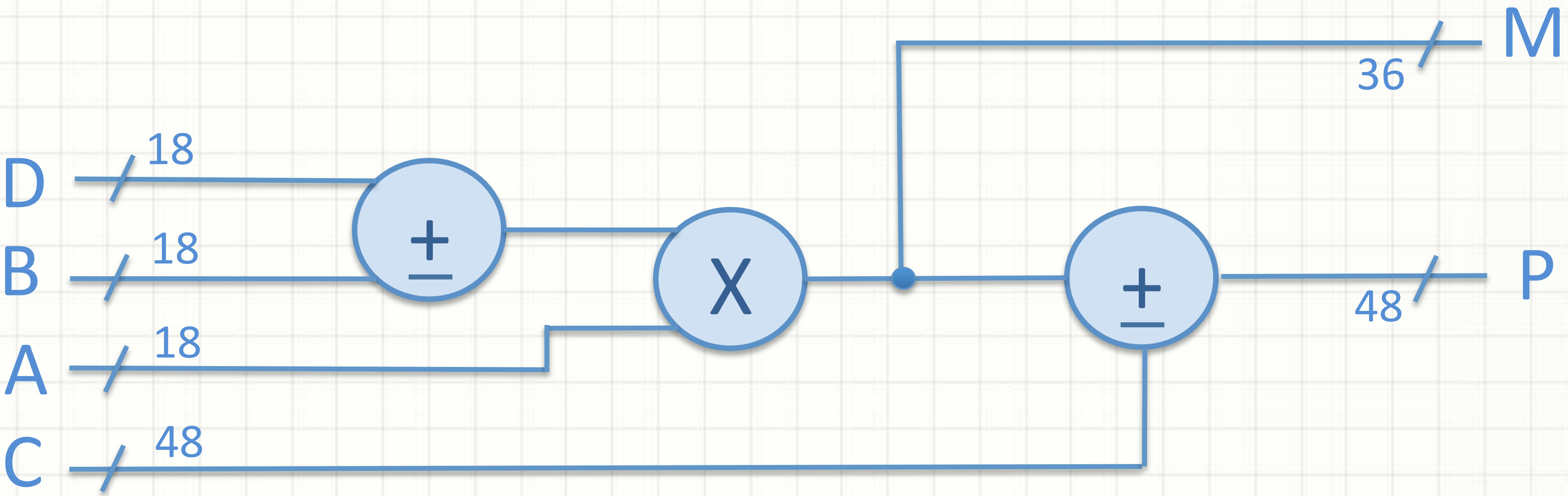
DSP48A1 Slice

- 48 bit add / subtract
- 18 x 18 multiplication with 36 bit product
- Designed for multiply-accumulate operations
- Pipelined operation possible
- Multiple modules can be cascaded for larger data sizes

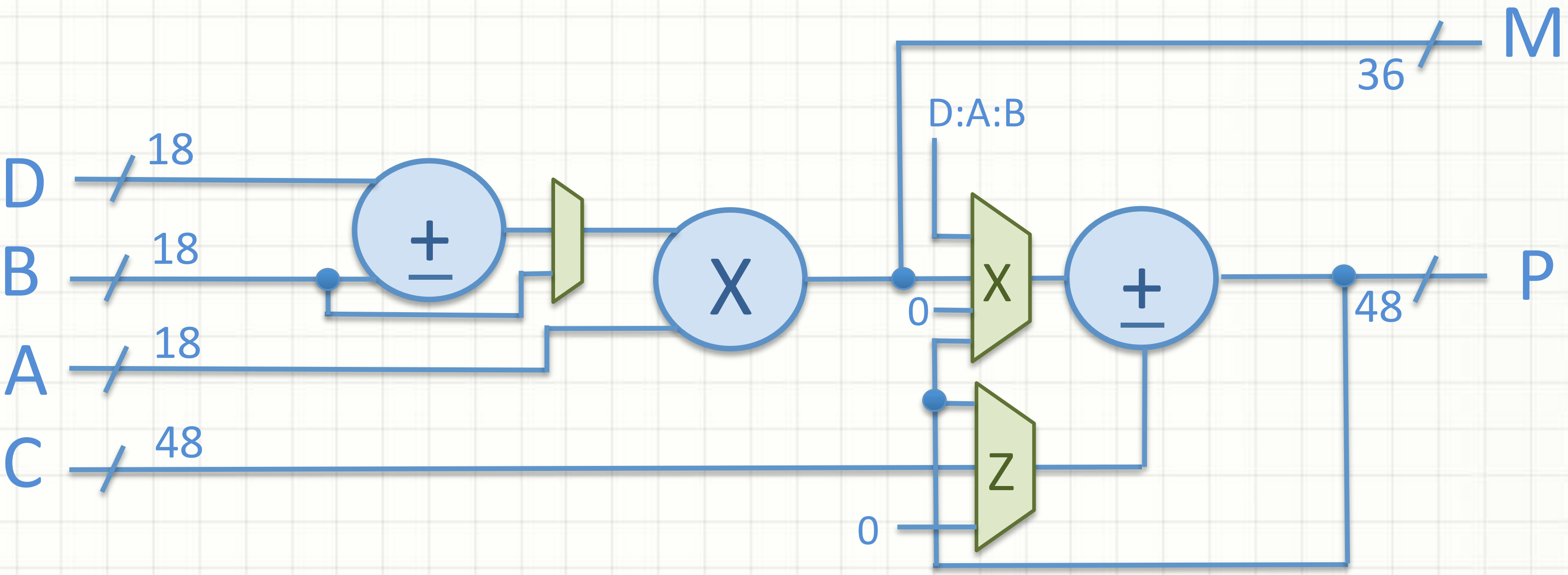
Block diagram



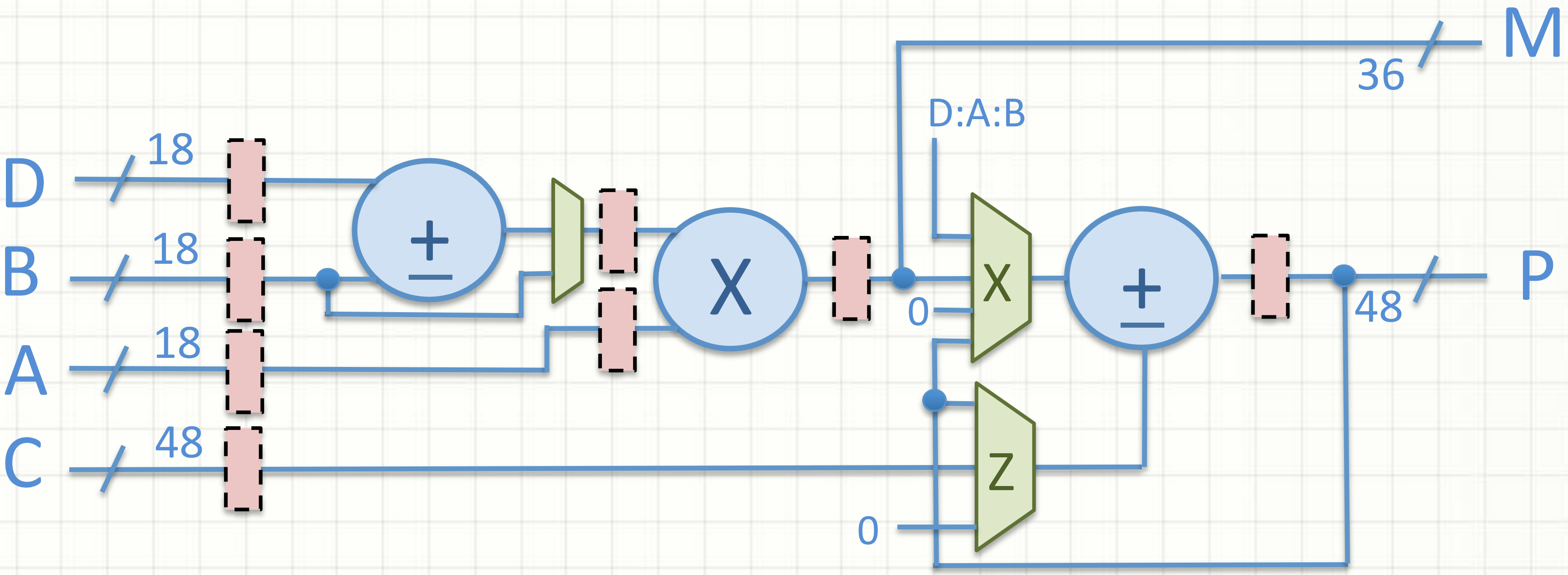
Simplified block diagram



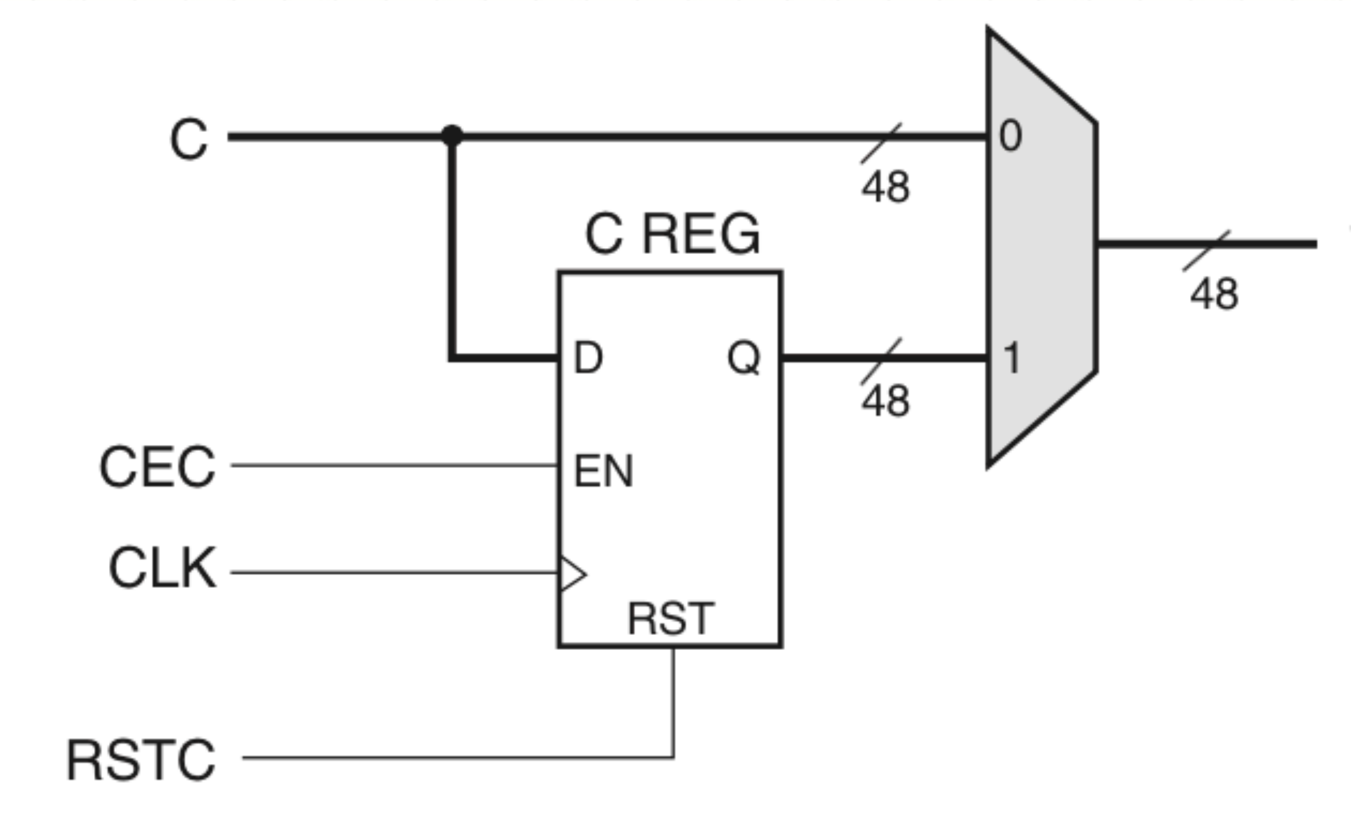
Accumulation function



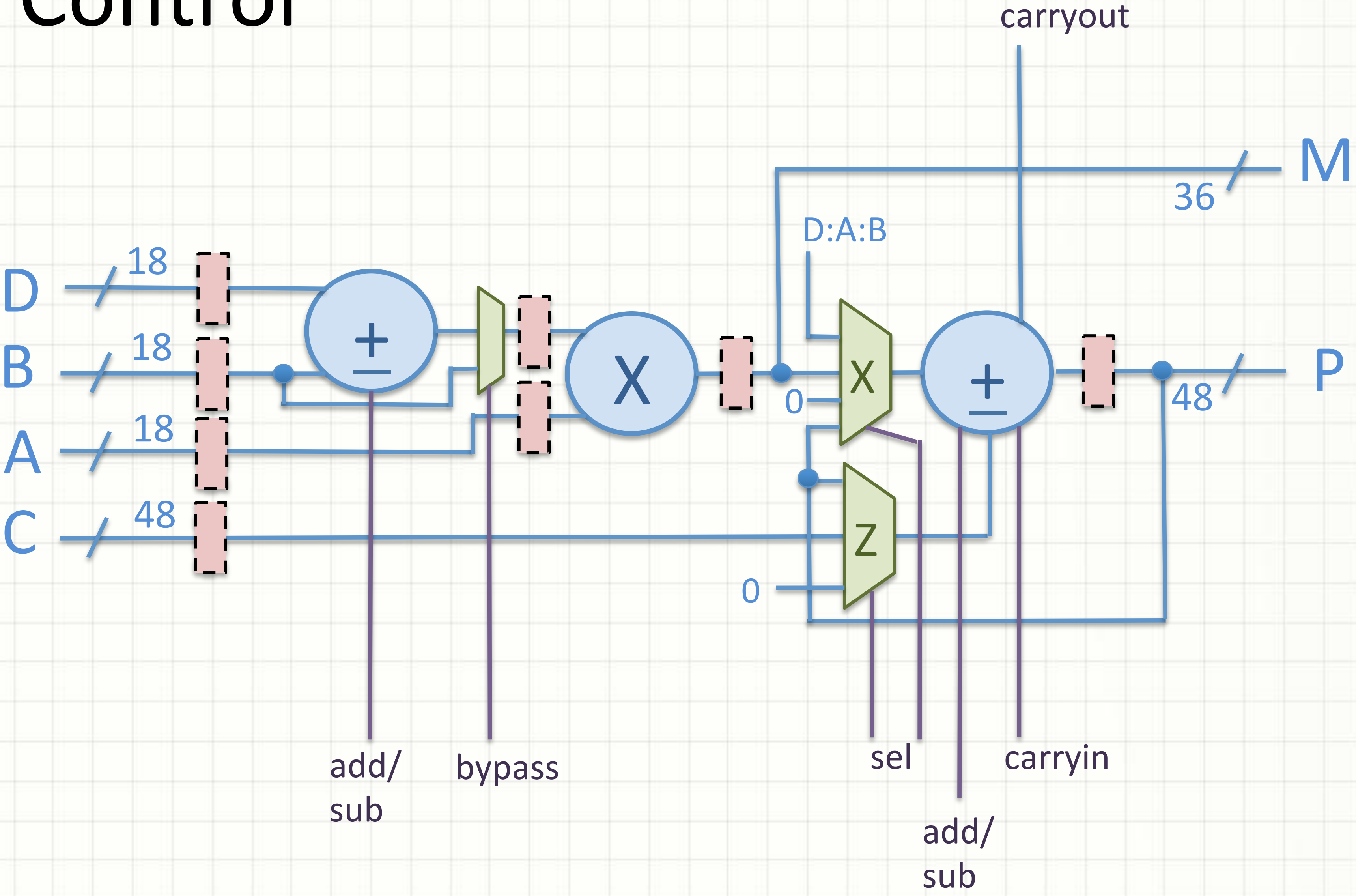
Pipelining registers



Register configuration



Control

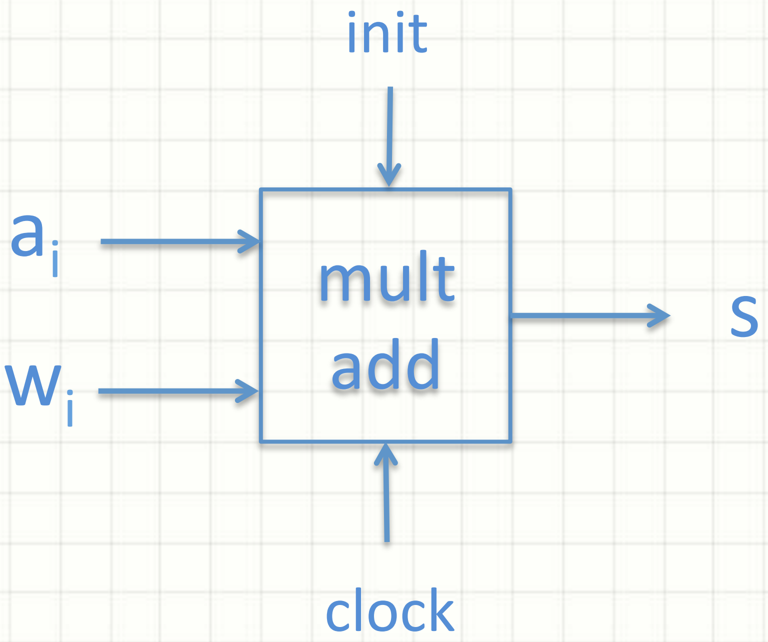

carryout
M
36
D:A:B
D
18
B
18
A
18
C
48
±
X
0
X
±
48
P
Z
0
add/
sub
bypass
sel
carryin
add/
sub

Computing weighted sum

- $s = w_0 \times a_0 + w_1 \times a_1 \; . \; . \; . \; w_{n-1} \times a_{n-1}$
  - Weights : w_i
  - Data values : a_i
- Common operation in
  - digital signal processing
  - digital image processing

    (filtering, convolution, correlation, spectrum computation, . . .)

Block diagram



init

a_i

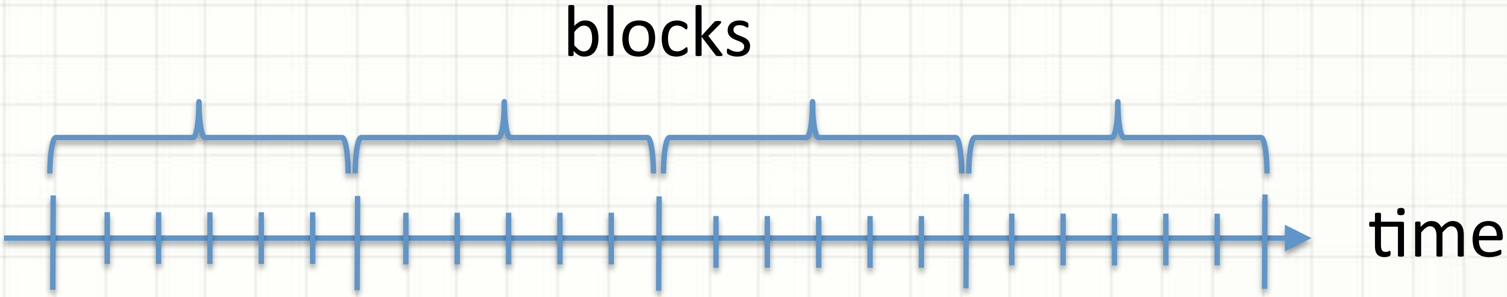
w_i

mult add

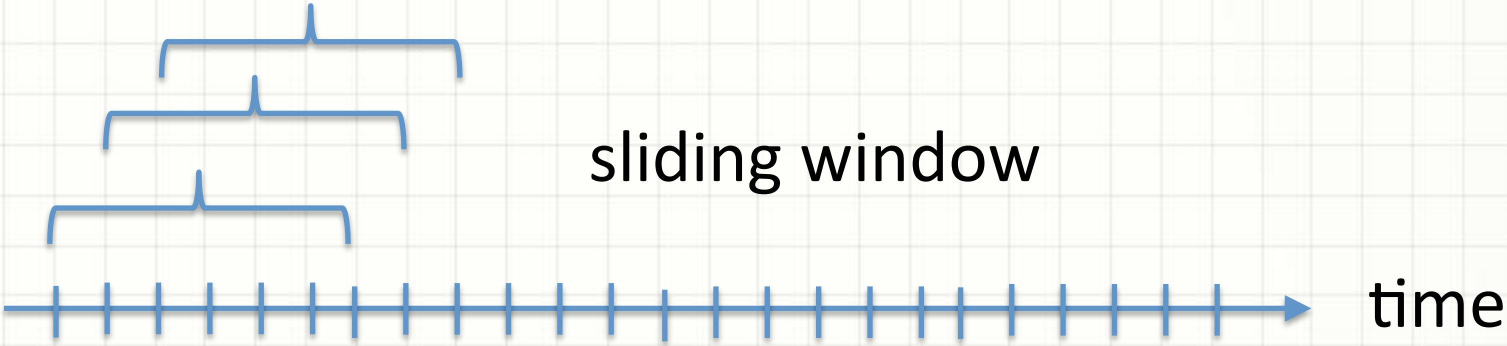
s

clock

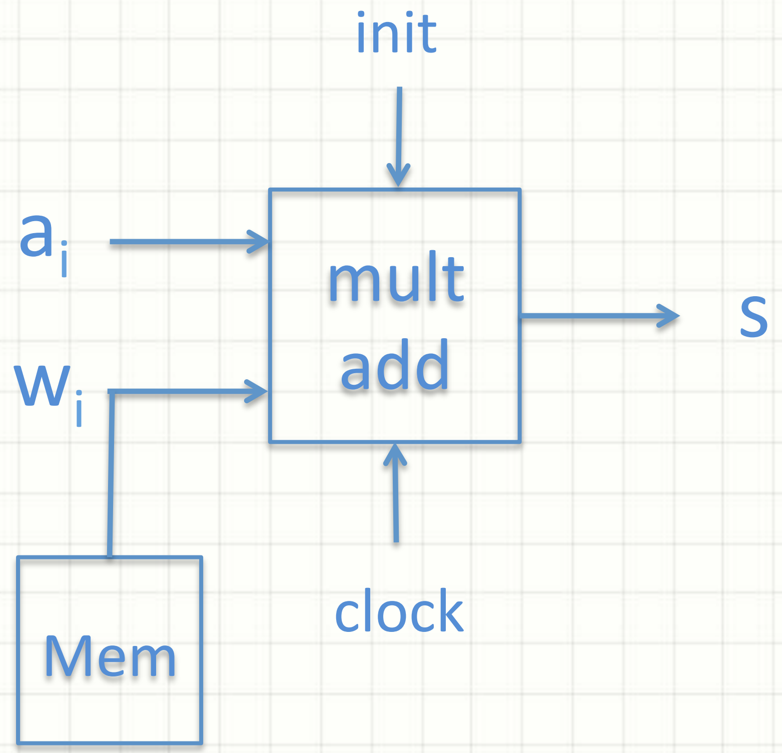
Block by Block Computation



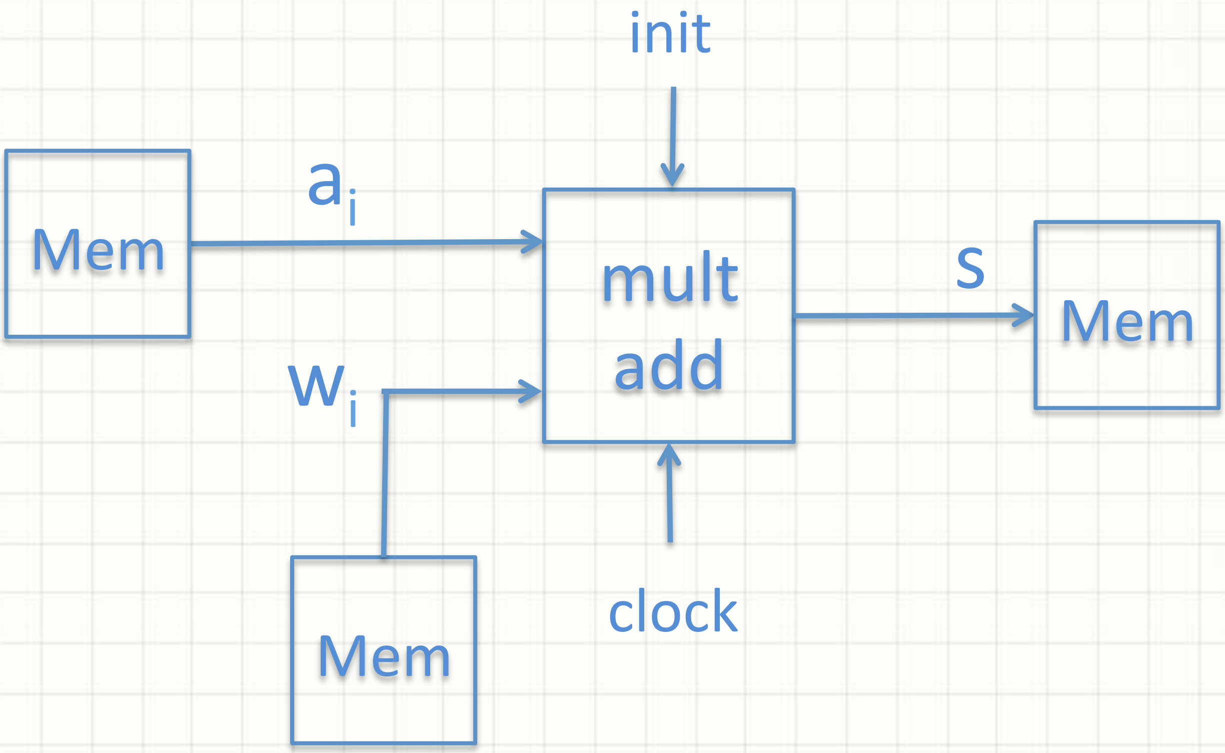
Computation with sliding window



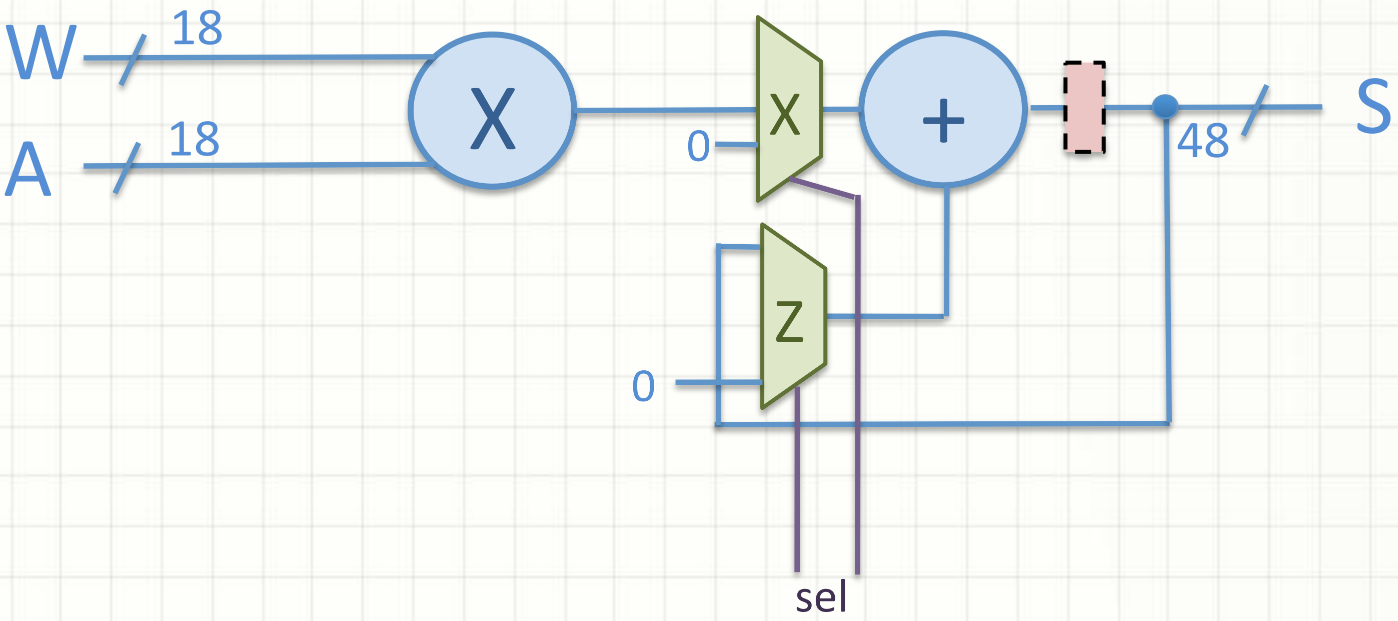
Weights in memory, Data streamed in, Sum streamed out



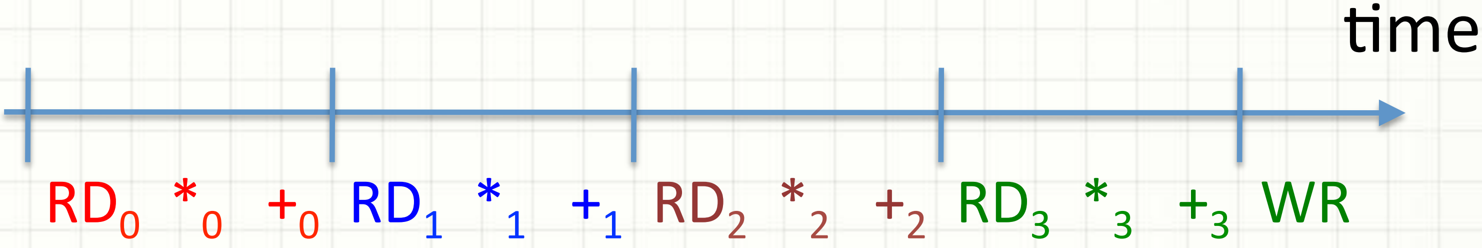
Weights, Data and Sum, all in memory



Weighted sum implementation



Timings

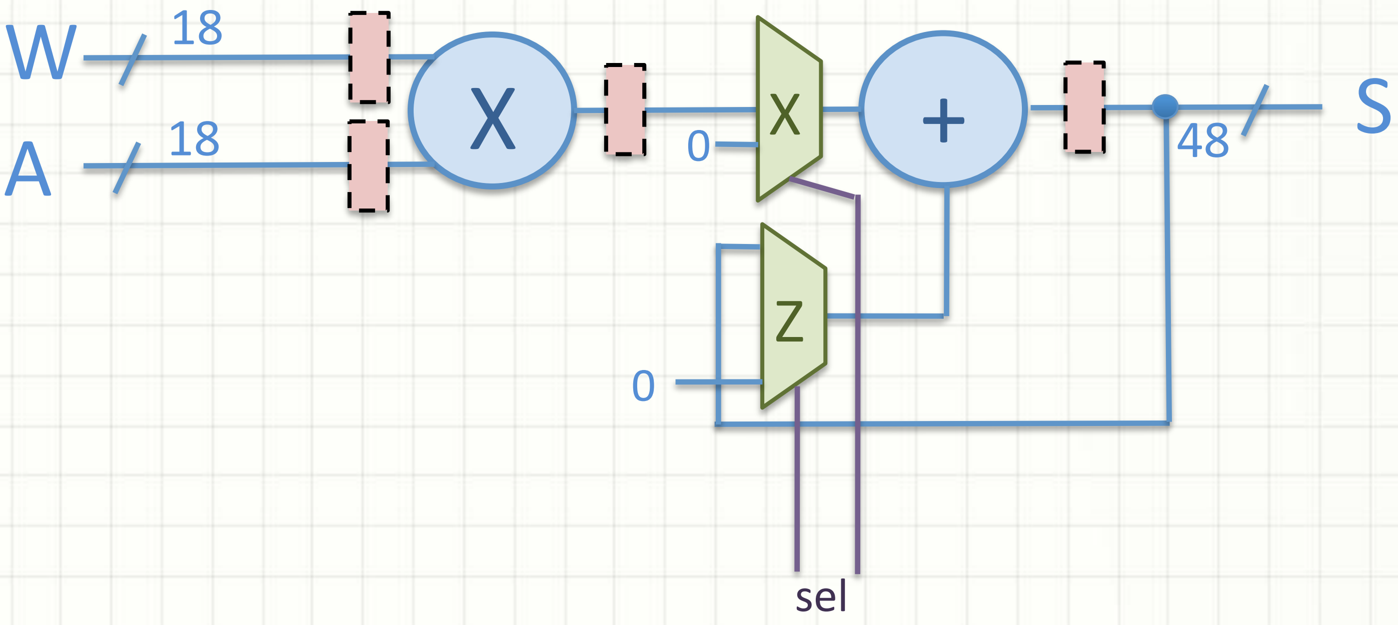


time

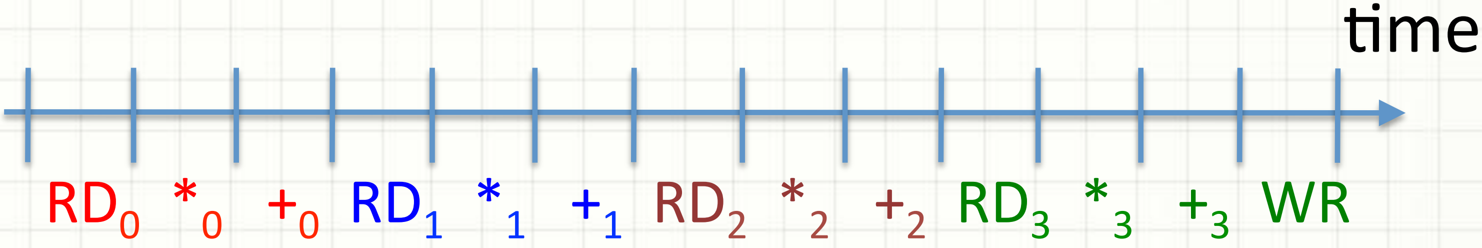
RD_0 $*_0$ $+_0$ RD_1 $*_1$ $+_1$ RD_2 $*_2$ $+_2$ RD_3 $*_3$ $+_3$ WR

Weighted sum implementation

Introduce registers



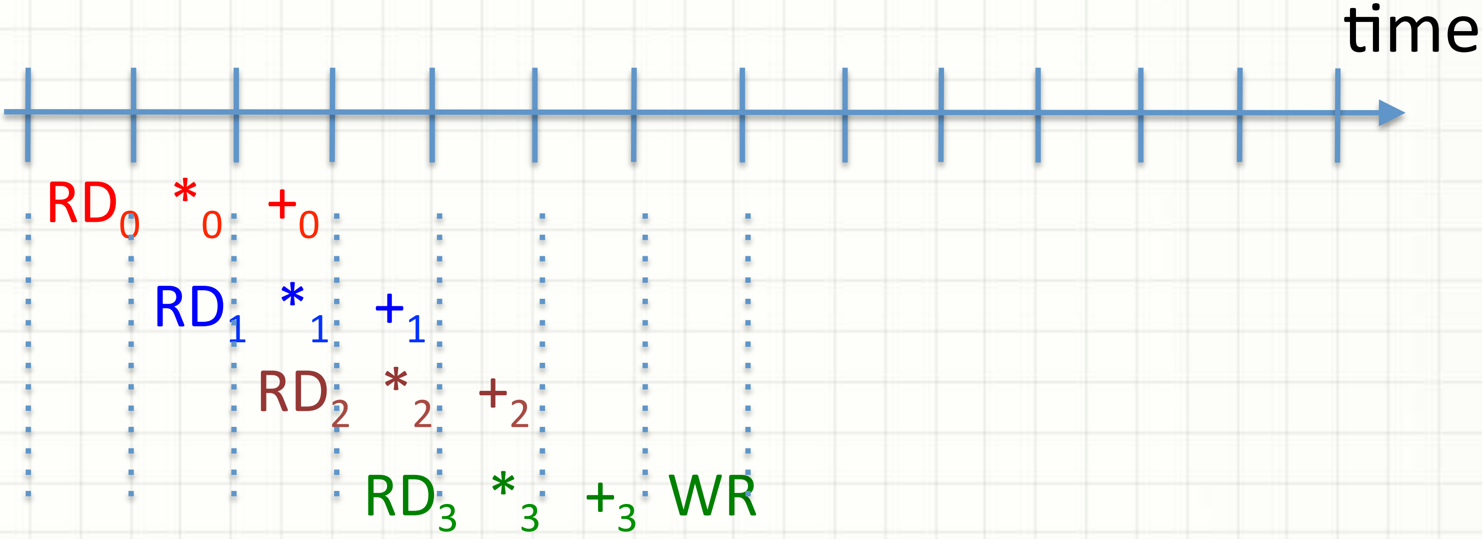
Timings with registers



time

RD_0 $*_0$ $+_0$ RD_1 $*_1$ $+_1$ RD_2 $*_2$ $+_2$ RD_3 $*_3$ $+_3$ WR

Timings with pipelining



time

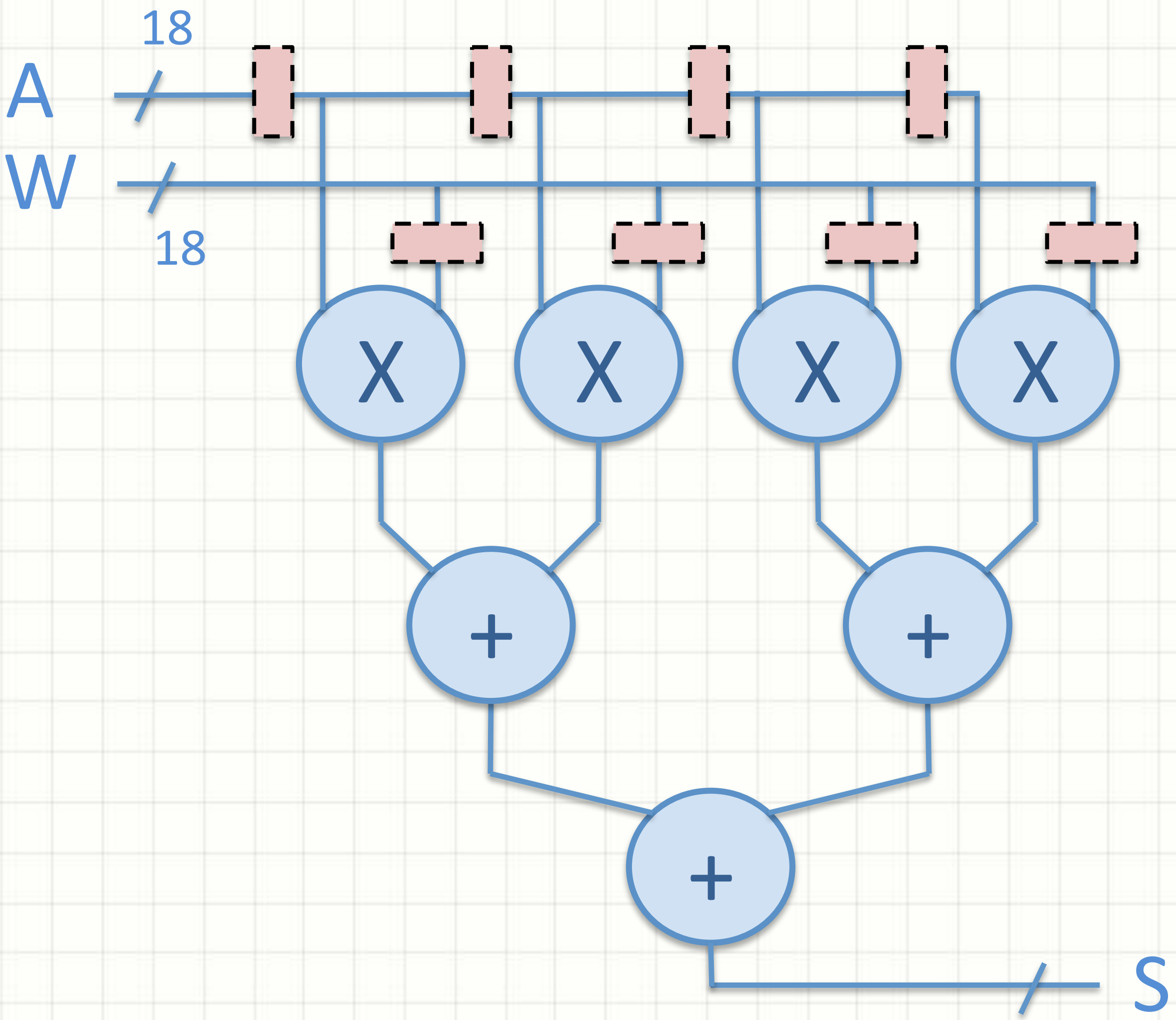
RD_0 $*_0$ $+_0$

RD_1 $*_1$ $+_1$

RD_2 $*_2$ $+_2$

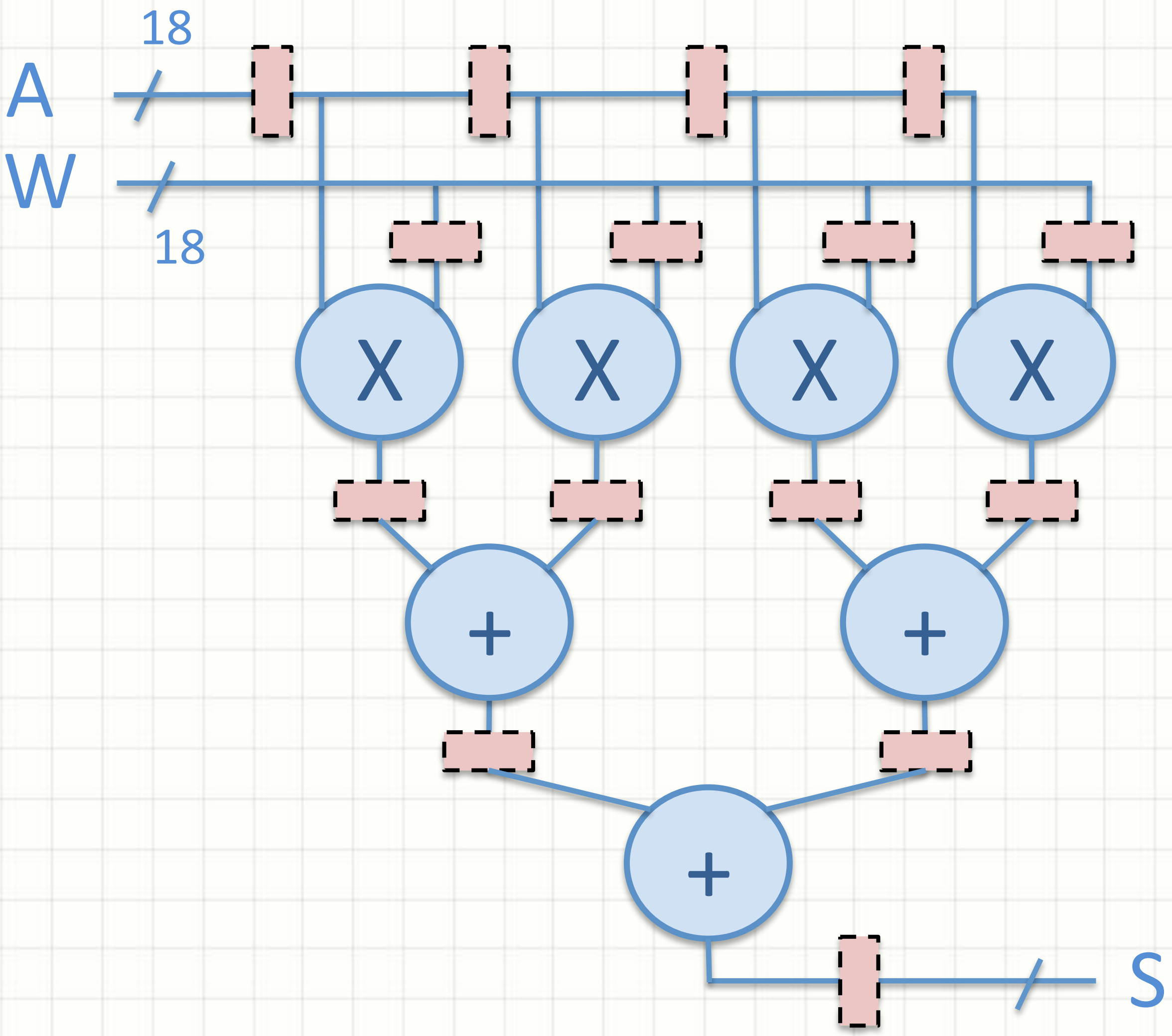
RD_3 $*_3$ $+_3$ WR

Weighted sum, sliding window



non-pipelined

Weighted sum, sliding window



pipelined

THANKS