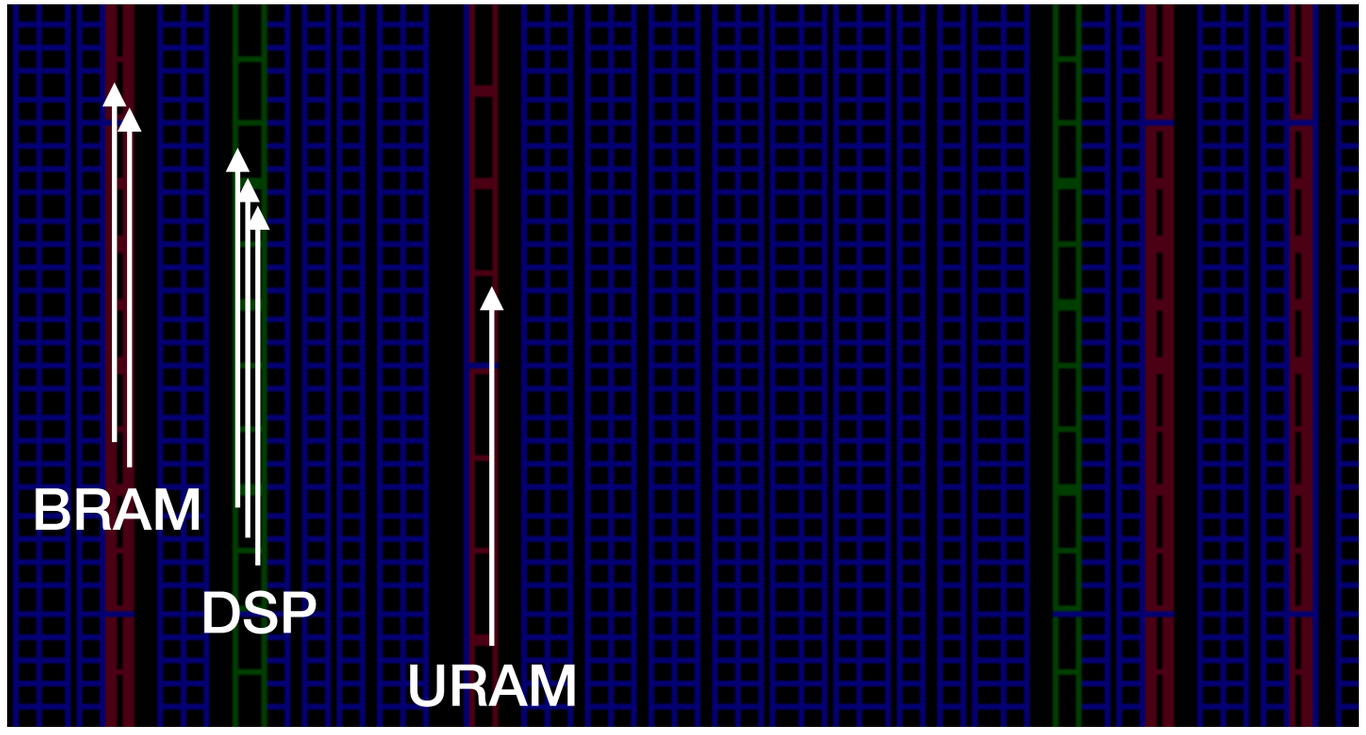


Scaling the Cascades

Interconnect-aware FPGA implementation of Machine Learning problems

Anand Samajdar, Tushar Garg, Tushar Krishna, Nachiket Kapre
nachiket@uwaterloo.ca

UNIVERSITY OF WATERLOO
Georgia Tech

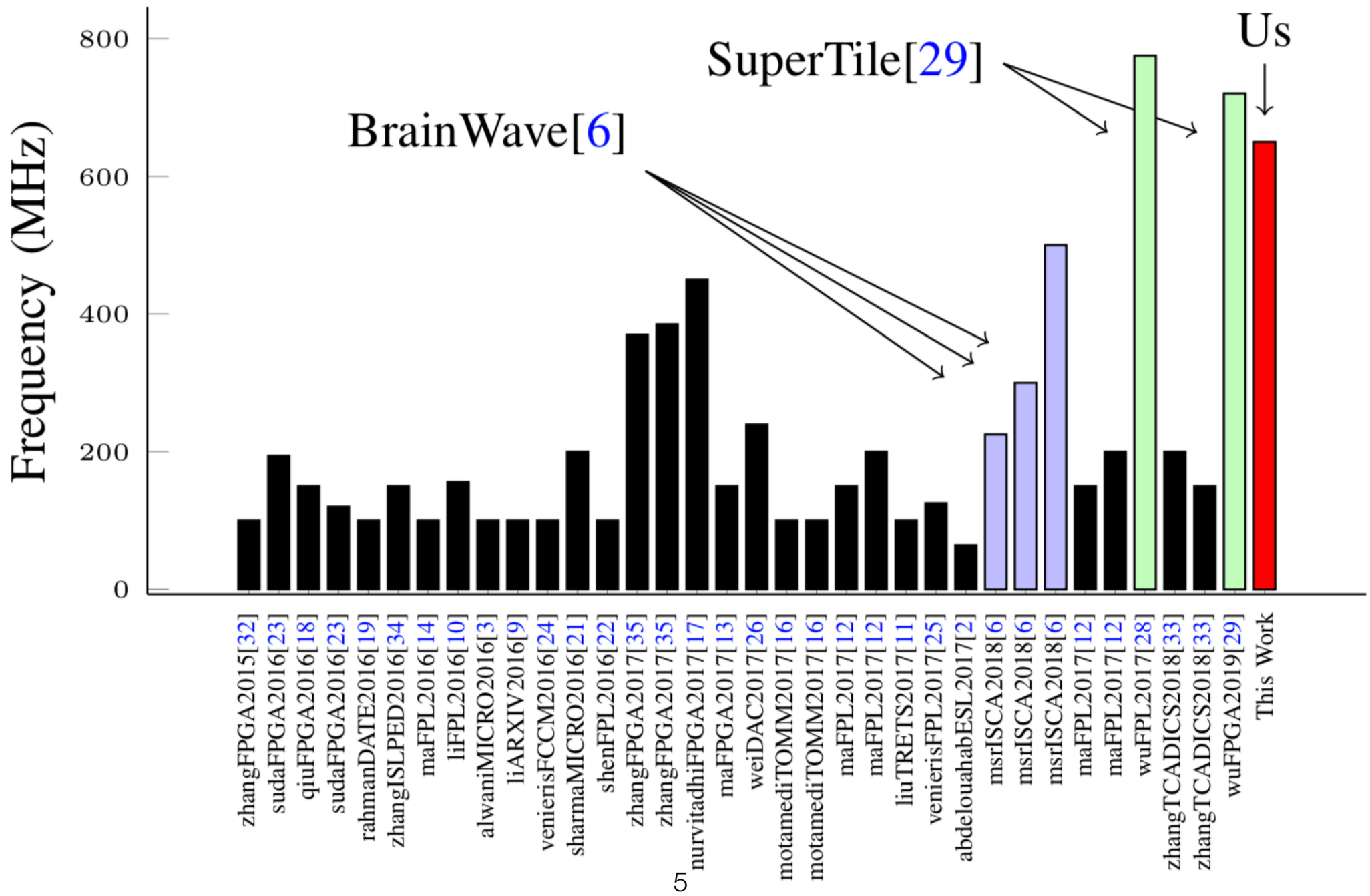

Vancouver
Surrey
Abbotsford
BRITISH COLUMBIA
Fraser
CANADIAN CASCADES
Similkameen
Mount Baker
(10,781')
NORTH CASCADES
Skagit
Mount Vernon
Lake Chelan
Everett
Seattle
Wenatchee
Tacoma
Olympia
Mount Rainier
(14,410')
WASHINGTON
Cowlitz
Yakima
Yakima
Mount St. Helens
(8,365')
Longview
Mount Adams
(12,281')
Columbia River Gorge
Columbia
Portland
Gresham
The Dalles
Mount Hood
(11,249')
Salem
Willamette
Deschutes
Pacific Ocean
Corvallis
Madras
Three Sisters
(10,363')
Eugene
Bend
Wickiup Reservoir
Umpqua
OREGON
WESTERN CASCADES
Crater Lake
Rogue
Mount McLoughlin
(9,495')
Upper Klamath Lake
Klamath Falls
Goose Lake
Klamath
Mount Shasta
(14,179')
Pit
CALIFORNIA
Lassen Peak
(10,462')
About 280 miles (451 km) across


Vancouver
Surrey
Abbotsford
BRITISH COLUMBIA
Fraser
Similkameen
Mount Baker
Skagit
Mount Vernon
Lake Chelan
Everett
Seattle
Wenatchee
Tacoma
Olympia
Mount Rainier
(14,410')
WASHINGTON
Cowlitz
Yakima
Yakima
Mount St. Helens
(8,365')
Longview
Mount Adams
(12,281')
Columbia River Gorge
Columbia
Portland
Gresham
The Dalles
Mount Hood
(11,249')
Salem
Willamette
Deschutes
Madras
Pacific Ocean
Corvallis
Three Sisters
(10,363')
Eugene
Bend
Wickiup Reservoir
Umpqua
OREGON
Crater Lake
Rogue
WESTERN CASCADES
Mount McLoughlin
(9,495')
Upper Klamath Lake
Klamath Falls
Goose Lake
Klamath
Mount Shasta
Pit
CALIFORNIA
Lassen Peak
(10,462')
About 280 miles (451 km) across

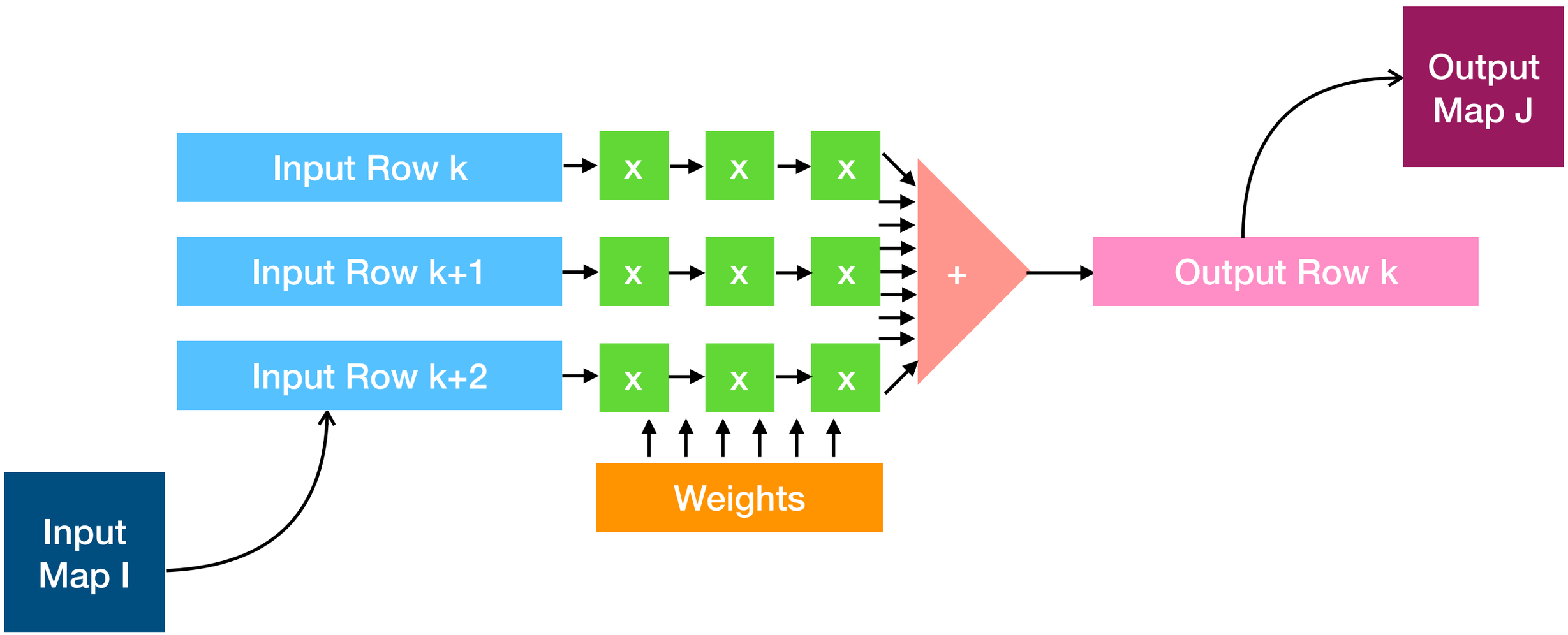
Claim

- Hard FPGA interconnect (cascades) efficiently supports **nearest neighbour communication** + **reuse** in ML workloads
- Three kinds of UltraScale+ cascades [DSP, BRAM, URAM]
  - Combination of (1) pixel, (2) row, (3) map reuse
- Deliverables:
  - 650 MHz full-chip operation
  - 7x **better** latency, 30% **lower** throughput than the formidable Xilinx SuperTile design for GoogLeNet v1

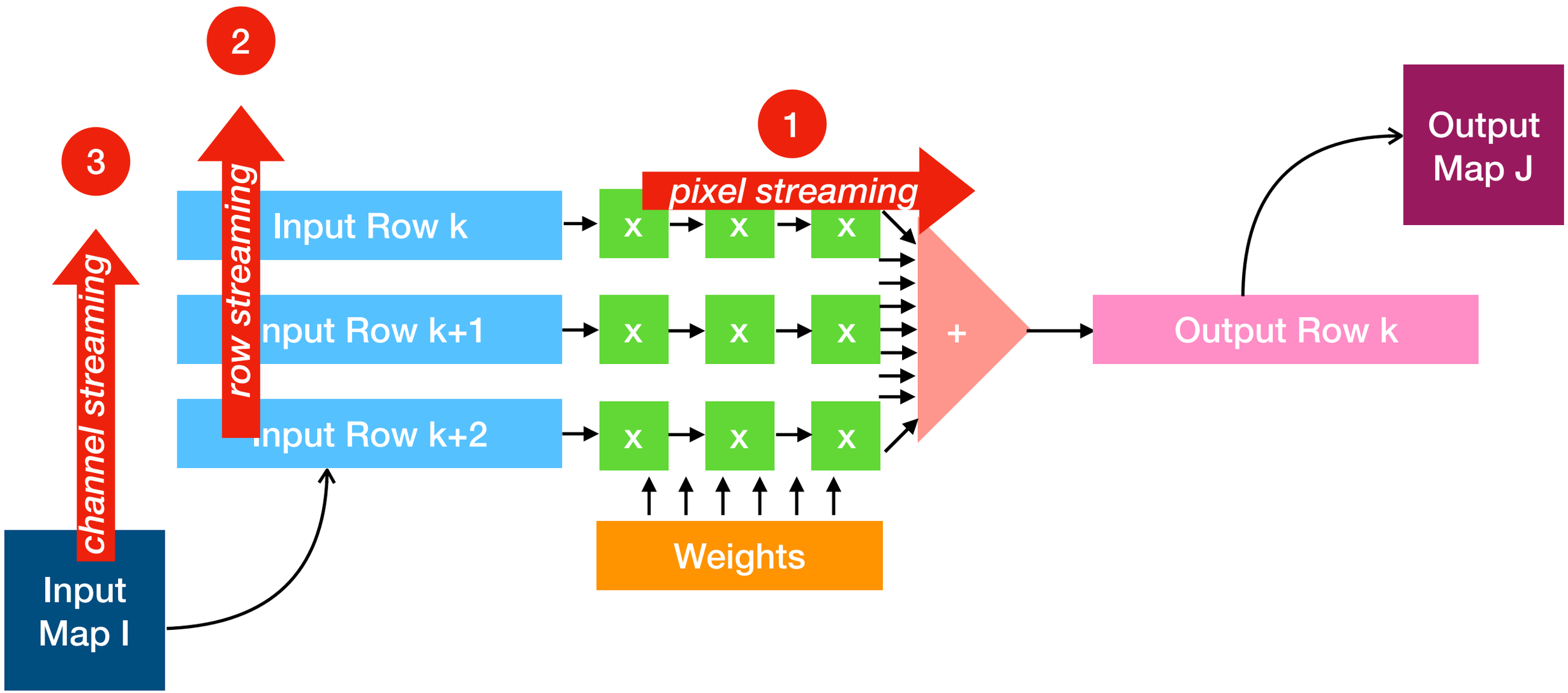
Landscape of FPGA+ML accelerators



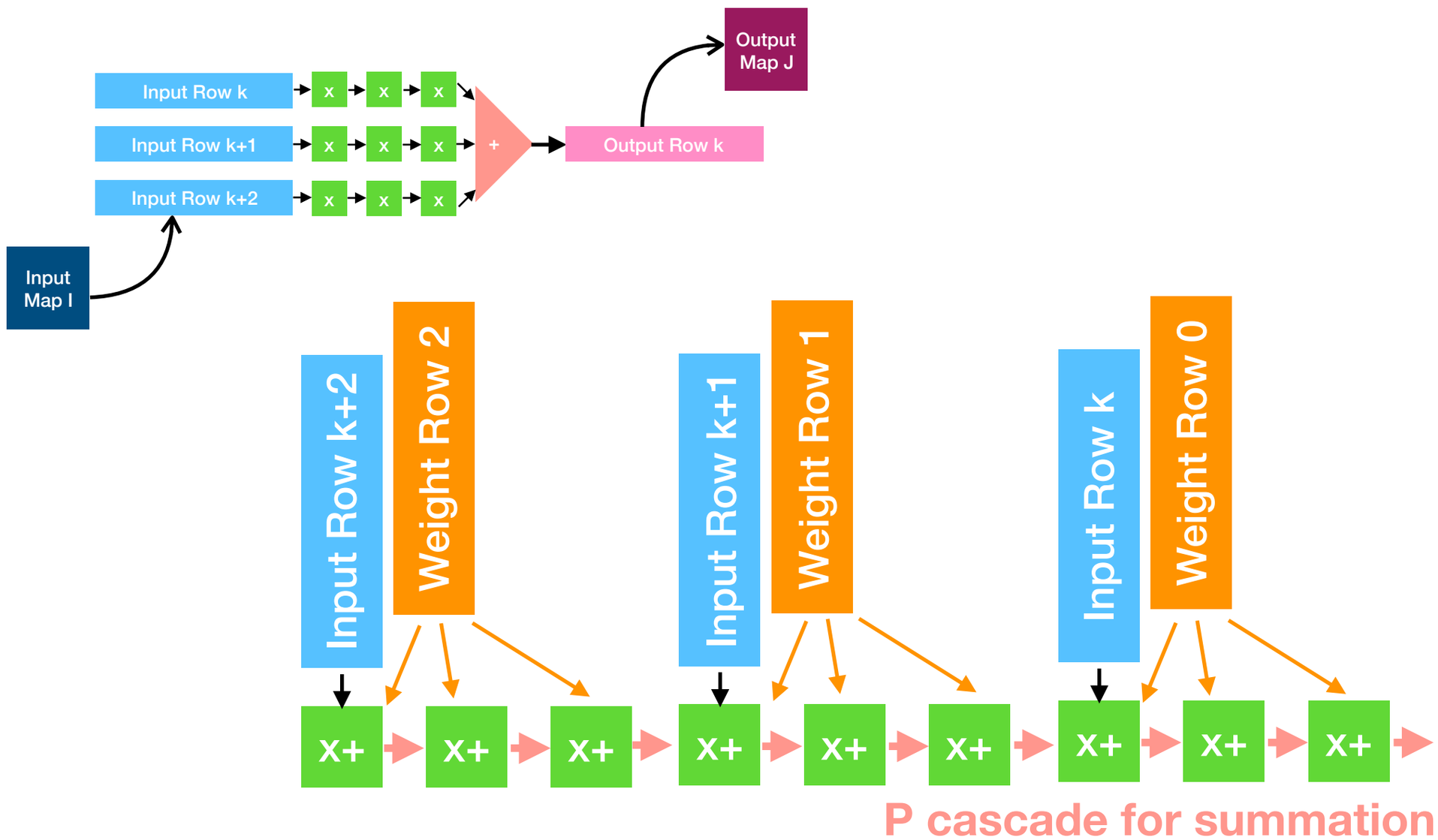
Communication Requirements of 3x3 Convolution



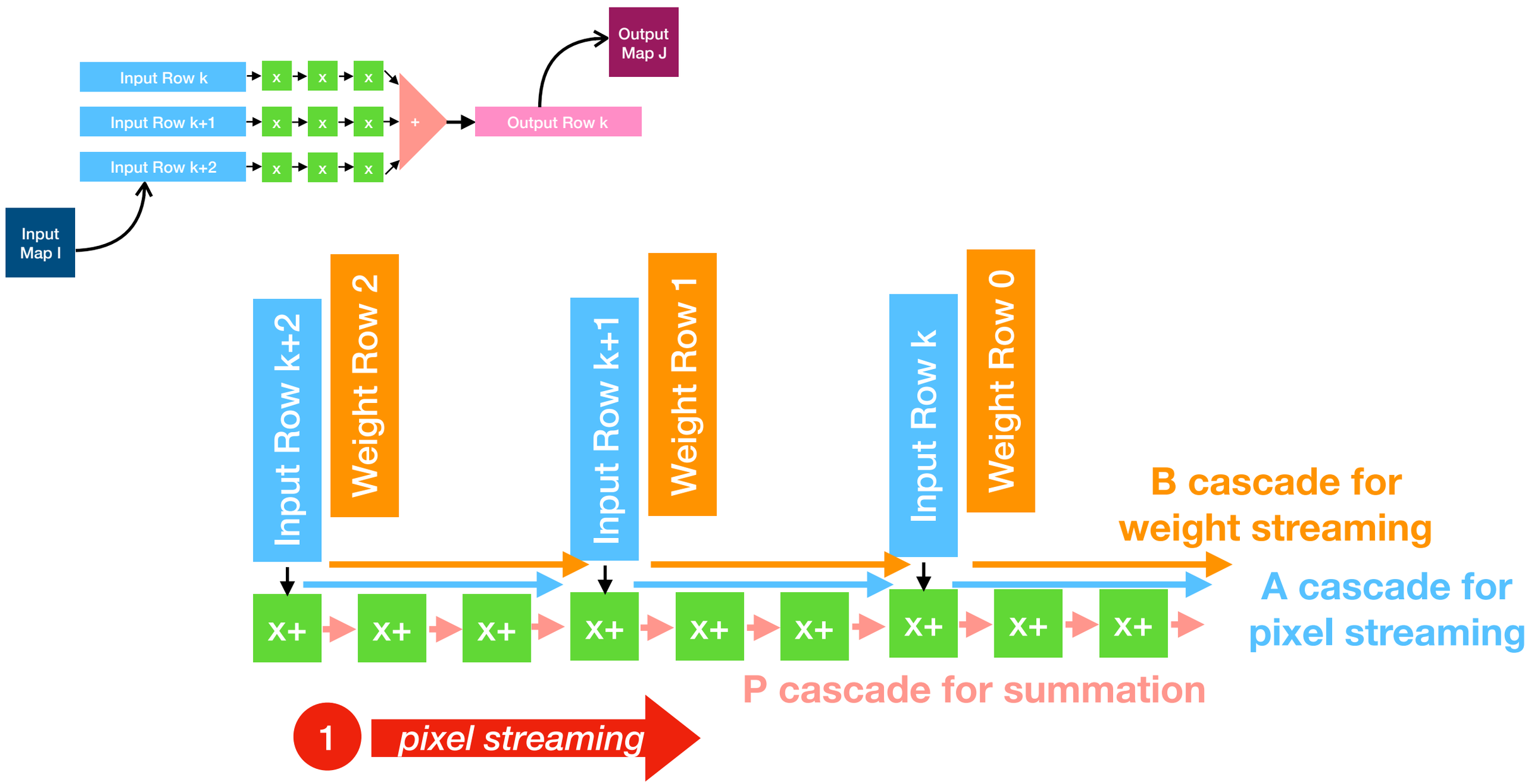
Communication Requirements of 3x3 Convolution



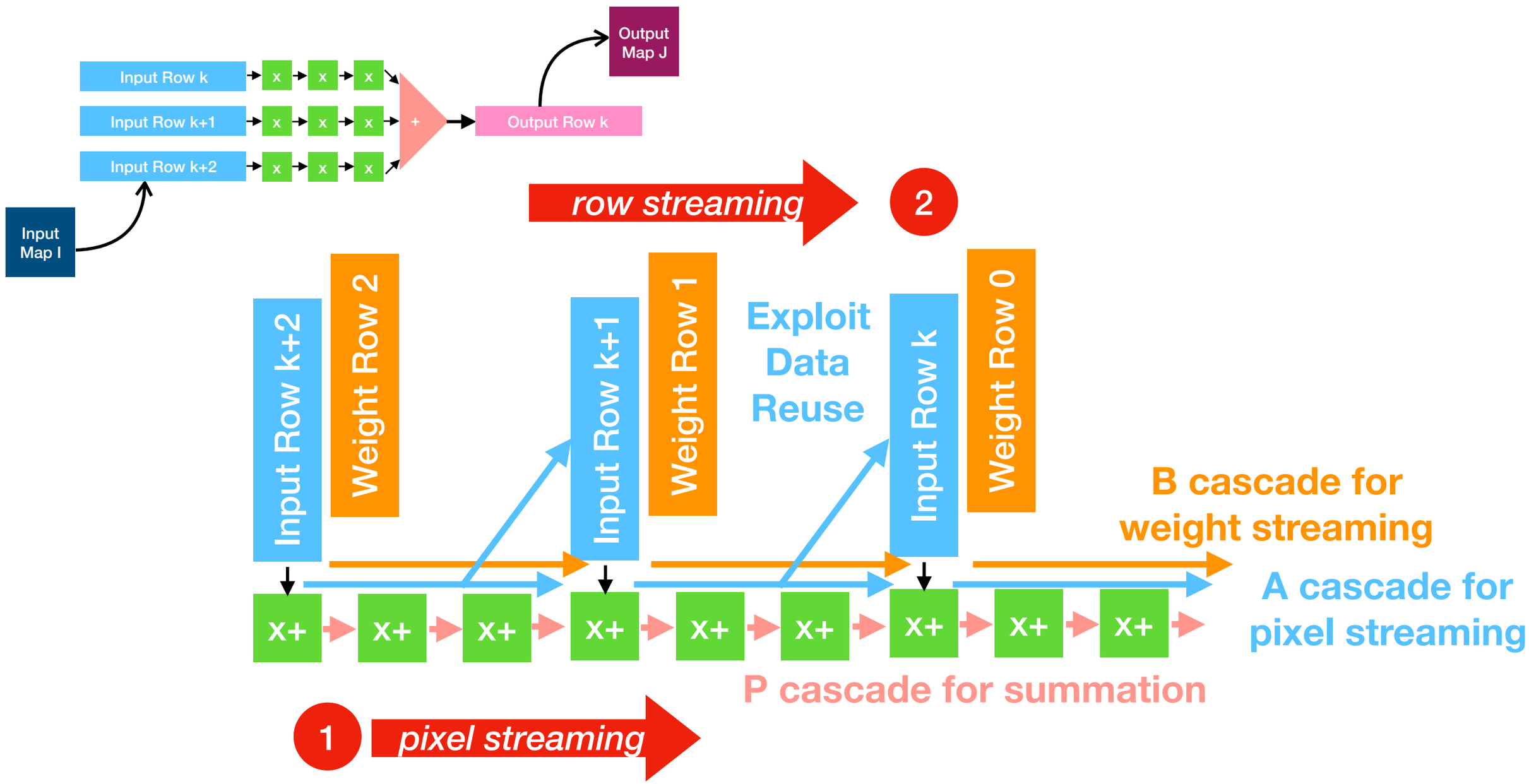
Reuse Patterns



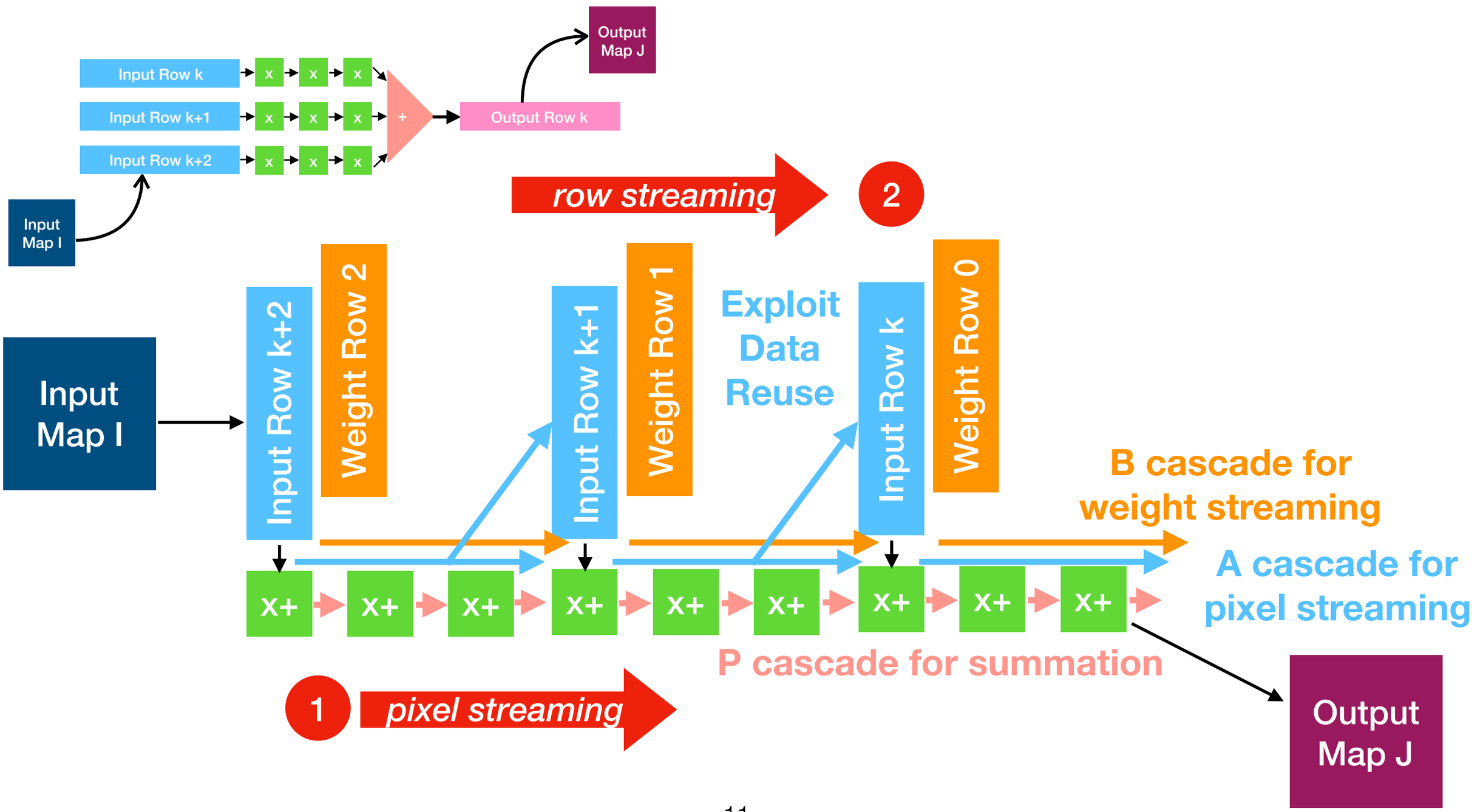
Reuse Patterns



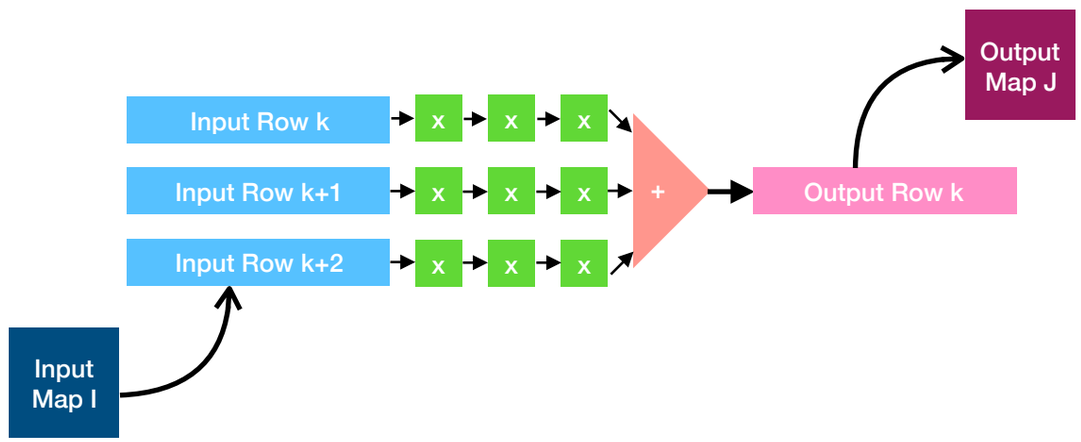
Reuse Patterns

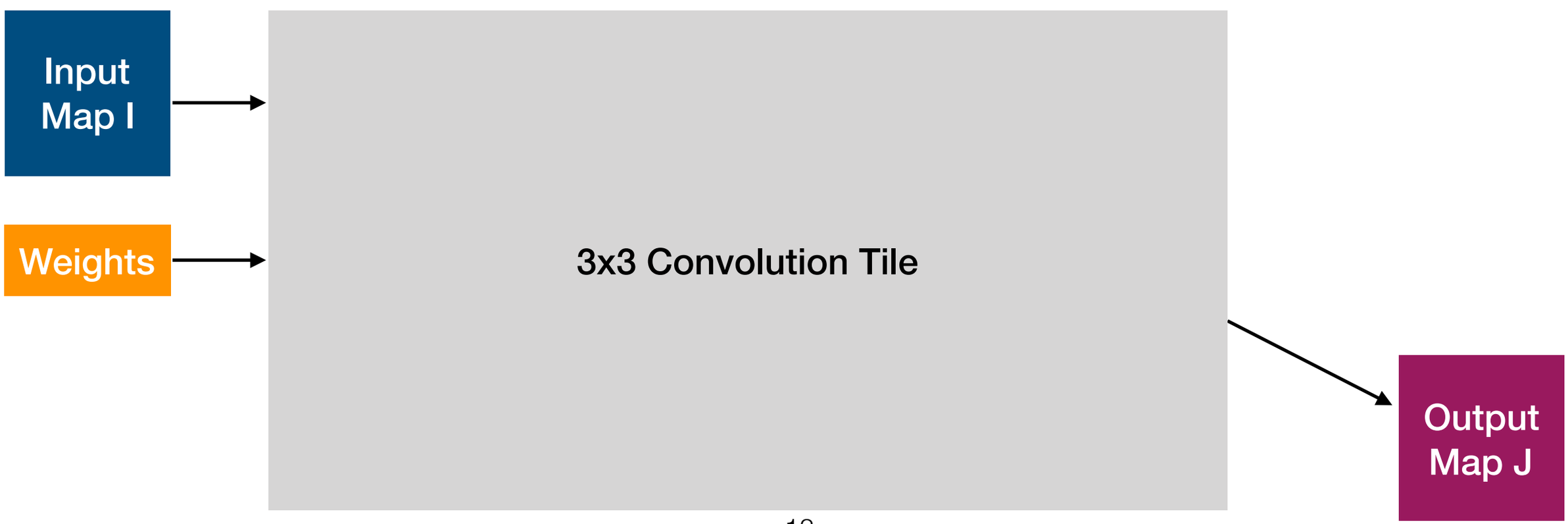


Reuse Patterns

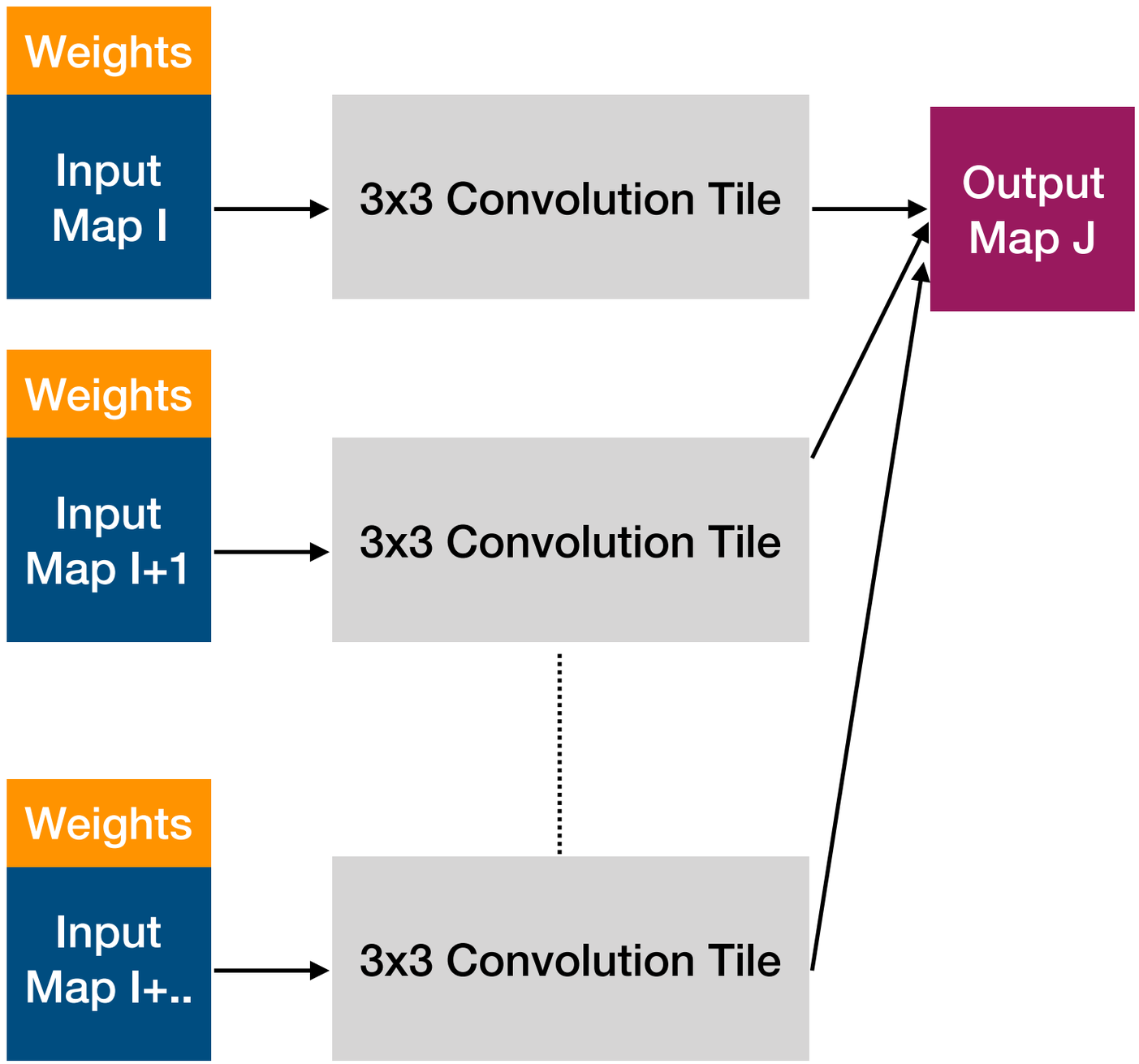


Reuse Patterns

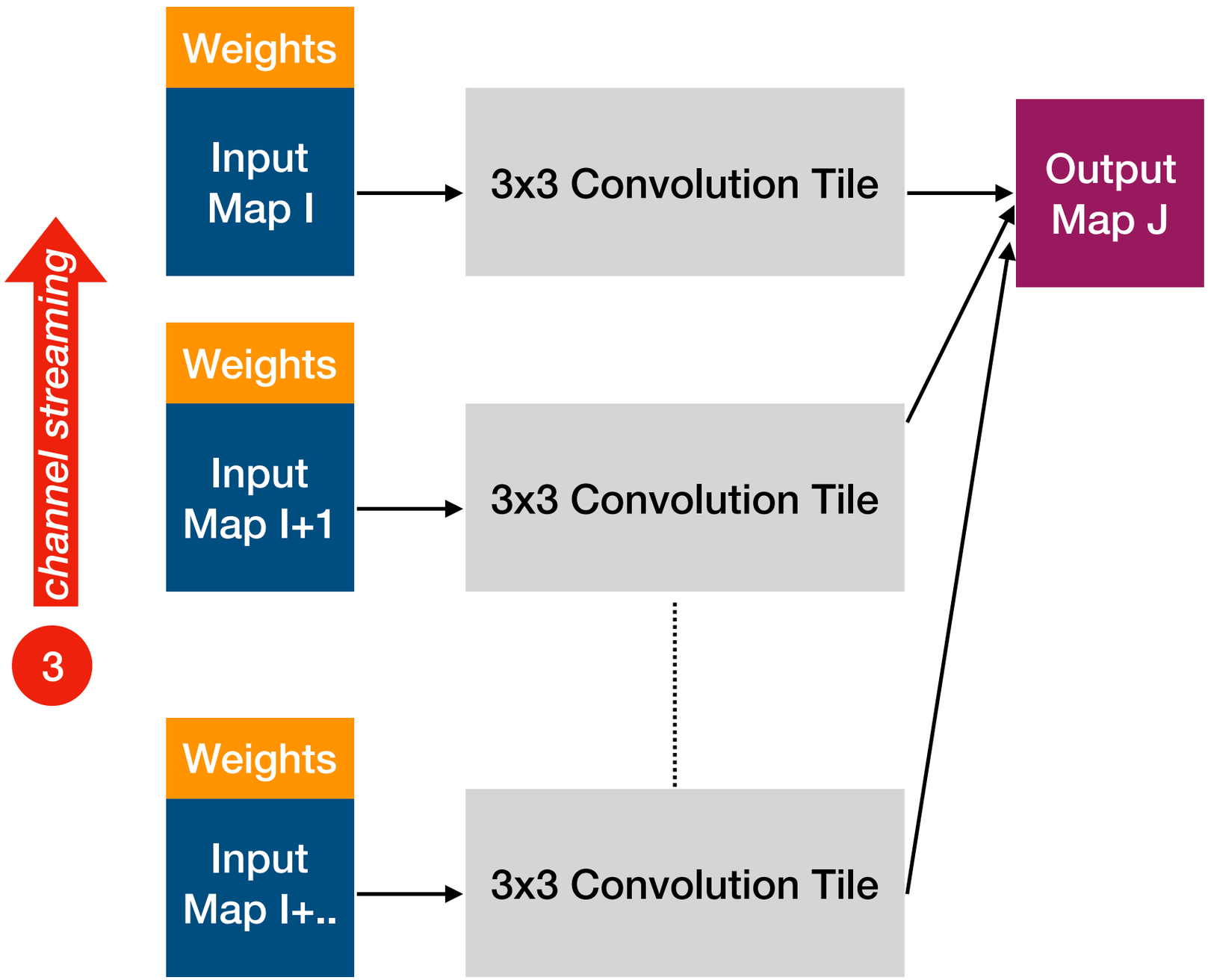




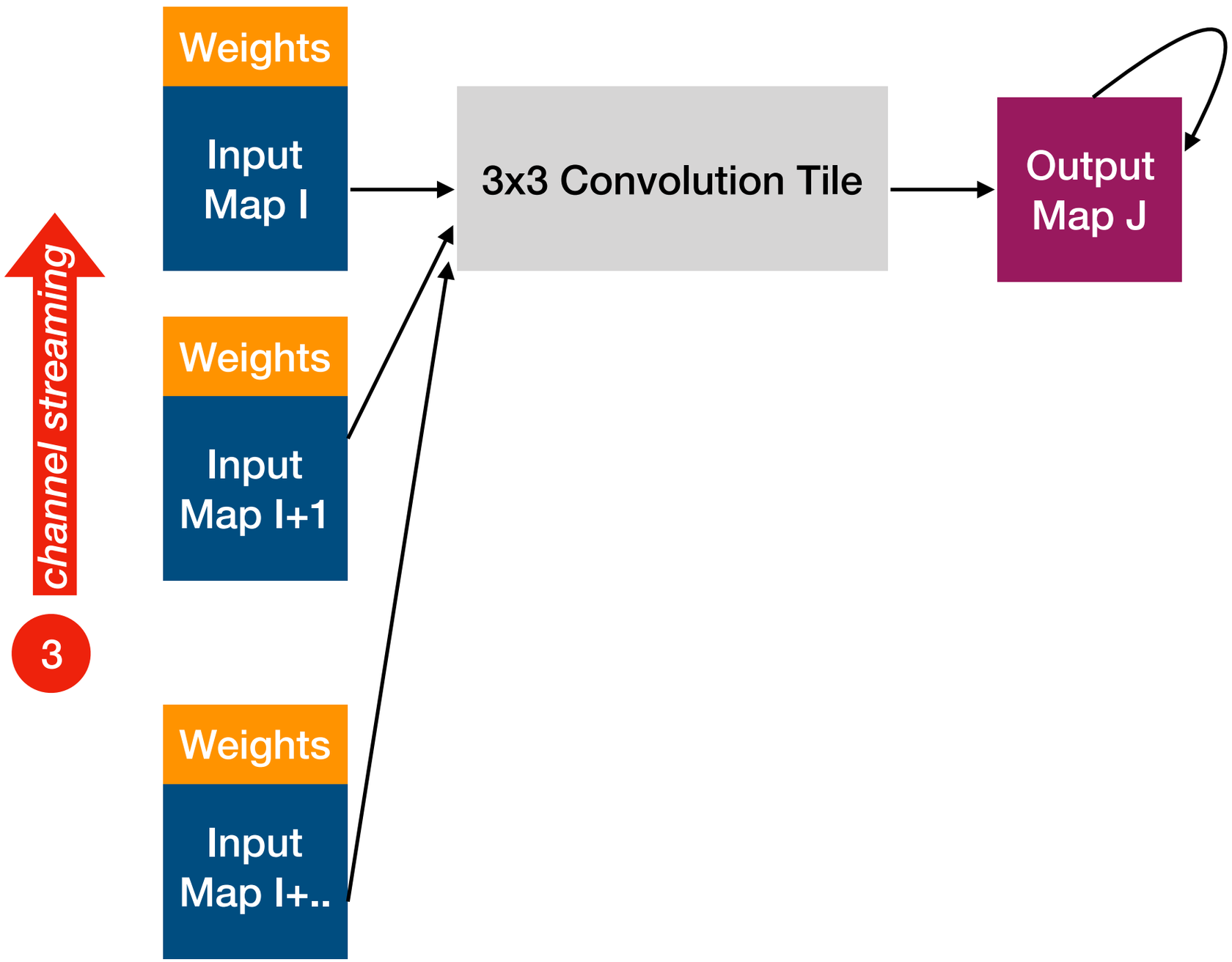
Reuse Patterns



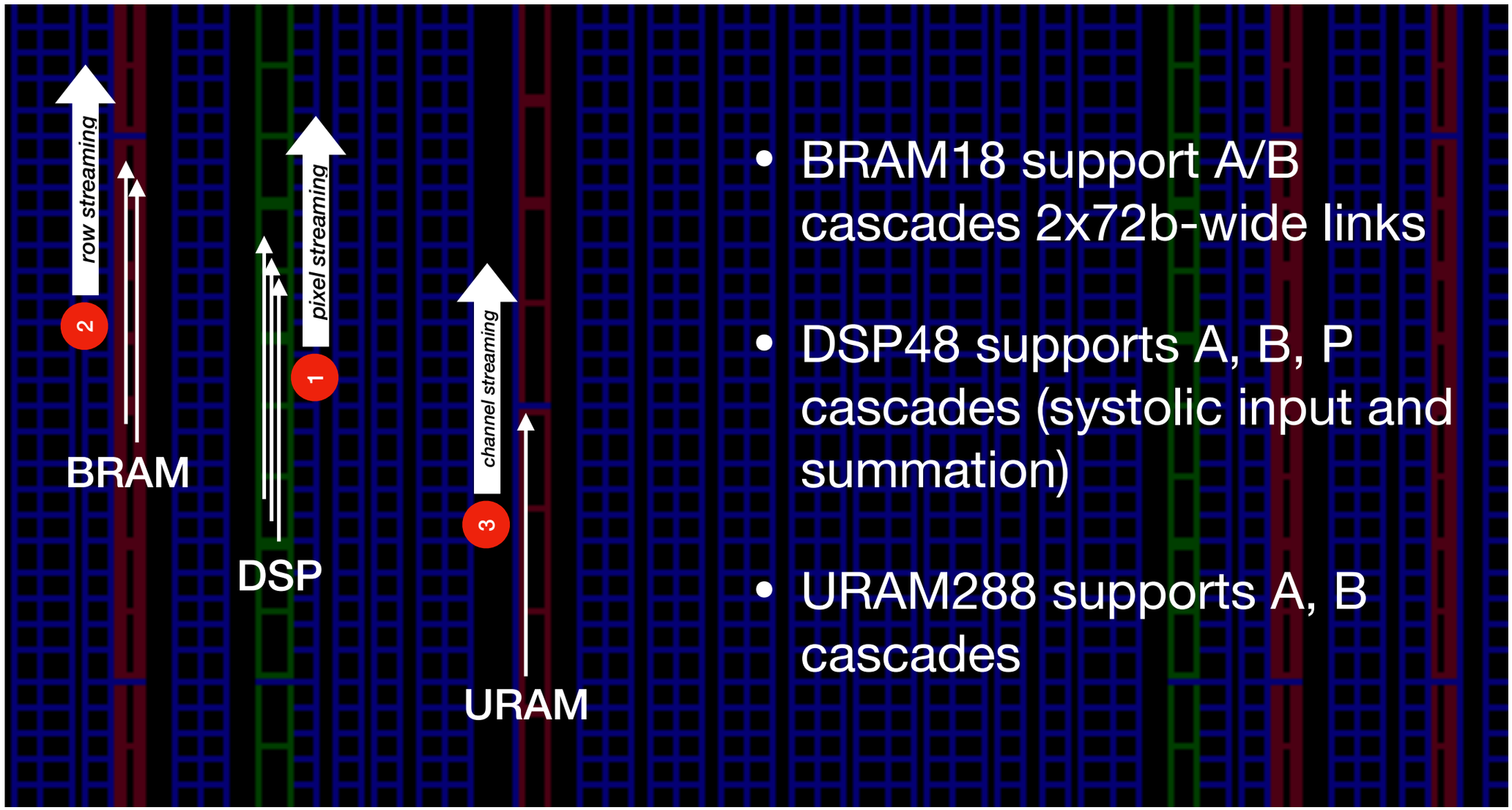
Reuse Patterns



Reuse Patterns



Xilinx UltraScale+ FPGA Cascades

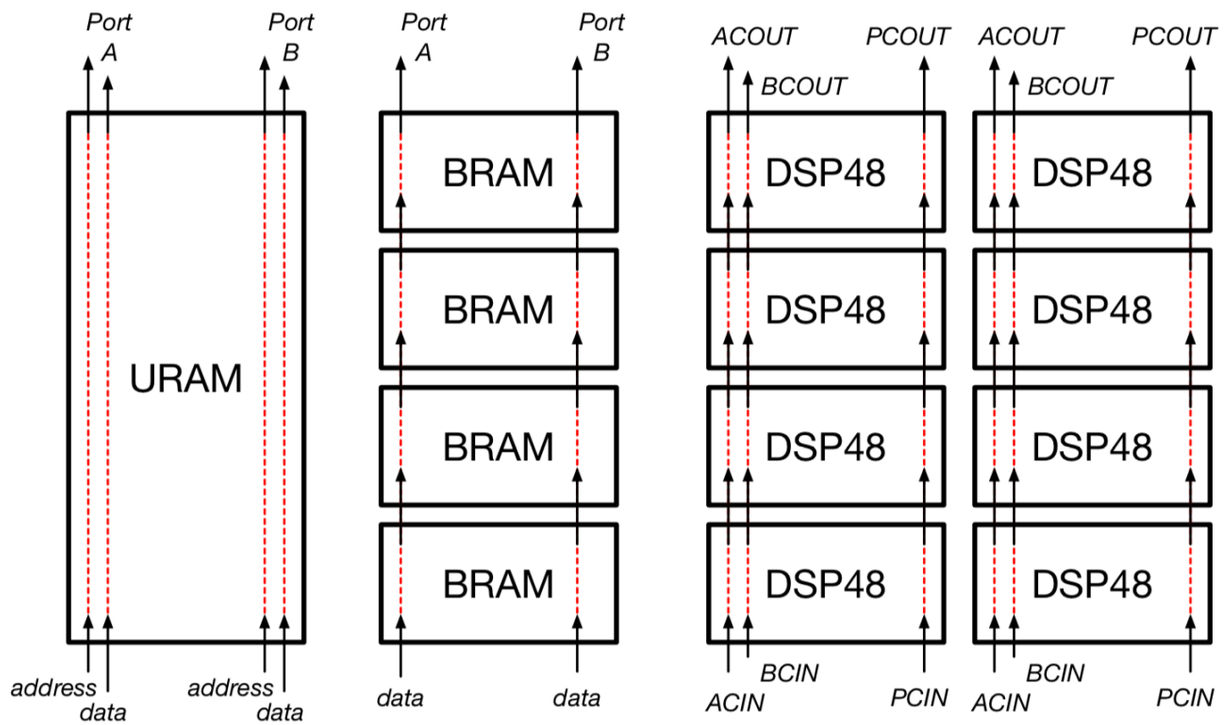


- BRAM18 support A/B cascades 2x72b-wide links
- DSP48 supports A, B, P cascades (systolic input and summation)
- URAM288 supports A, B cascades

Outline

- Understanding Cascades
- Assembling the FPGA accelerator + FPGA Layout
- MLPerf Evaluation
- Conclusions + Discussion

Promise of Cascades

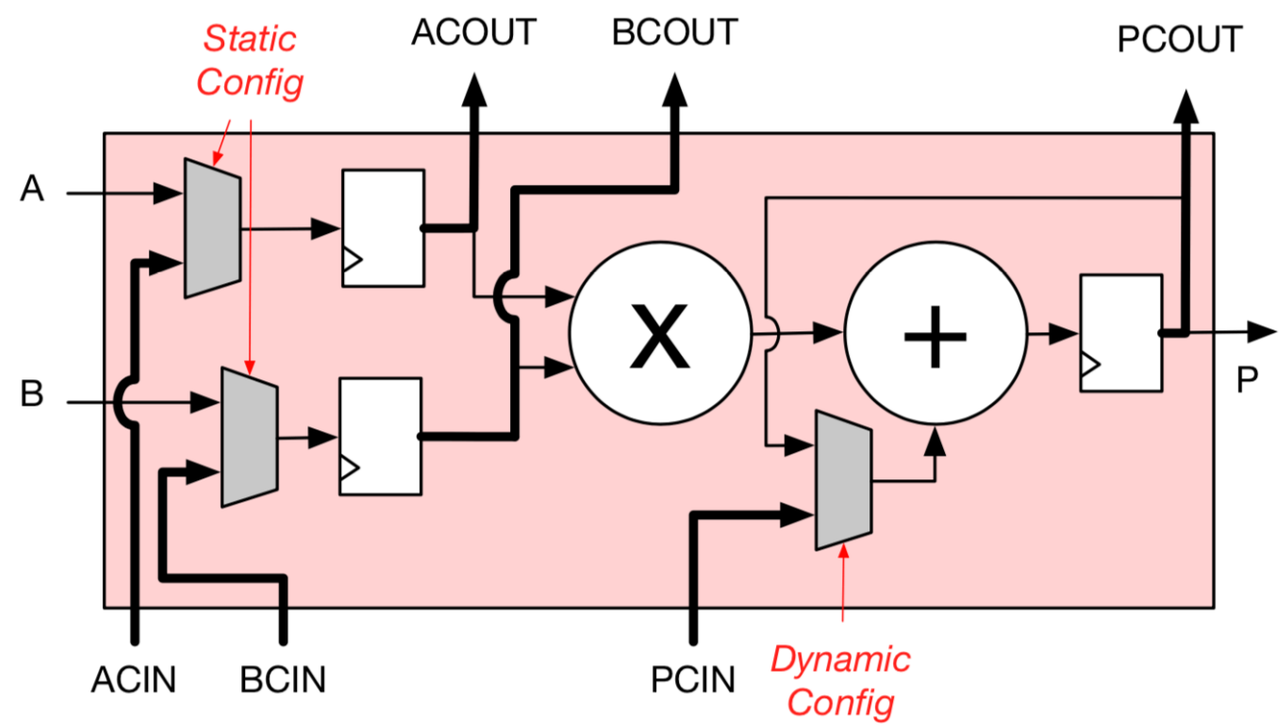


- Absorb data movement onto dedicated interconnect vs. General-purpose wiring

- Higher clock frequency operation, layout-friendly architecture

Our approach

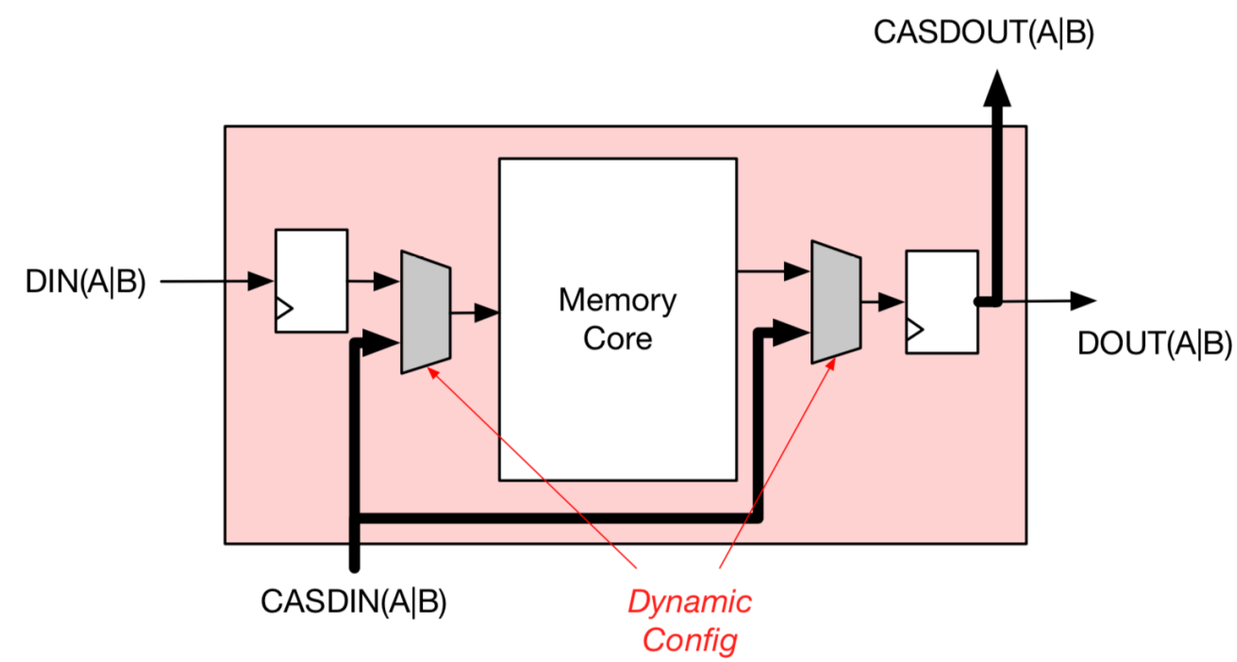
- Exploit cascades aggressively!
- **DSP48**
  - For 3x3 convolution, length-9 cascades
  - P cascade for summation (like INT8 paper)
  - A cascade for systolic retiming (like DSP48E2 user guide)
  - B cascades for weights (our contribution)



(a) DSP48 cascade (891 MHz)

Our approach

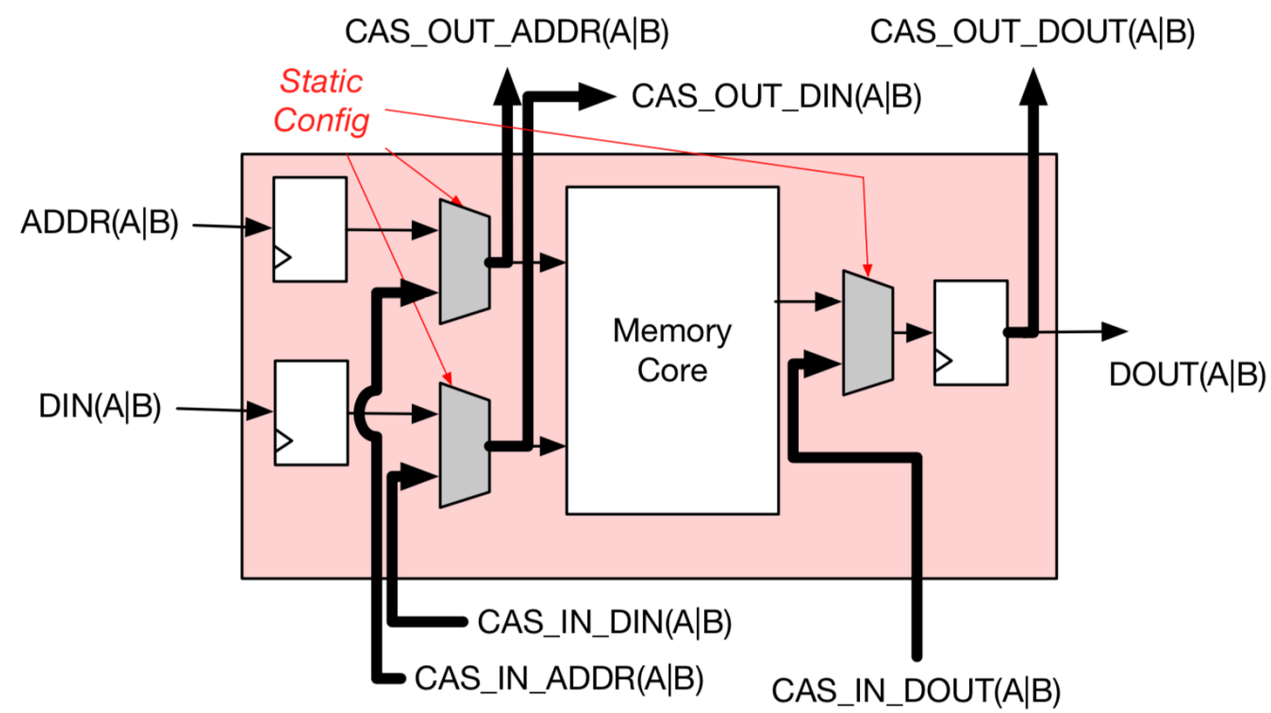
- Exploit cascades aggressively!
- **RAMB18E2** (our contribution)
  - For 3x3 convolution, only need 3 BRAM-long chains
  - A/B cascade for shift operation
  - Swap between A and B to keep one read port available.



(b) RAMB18 cascade (825 MHz))

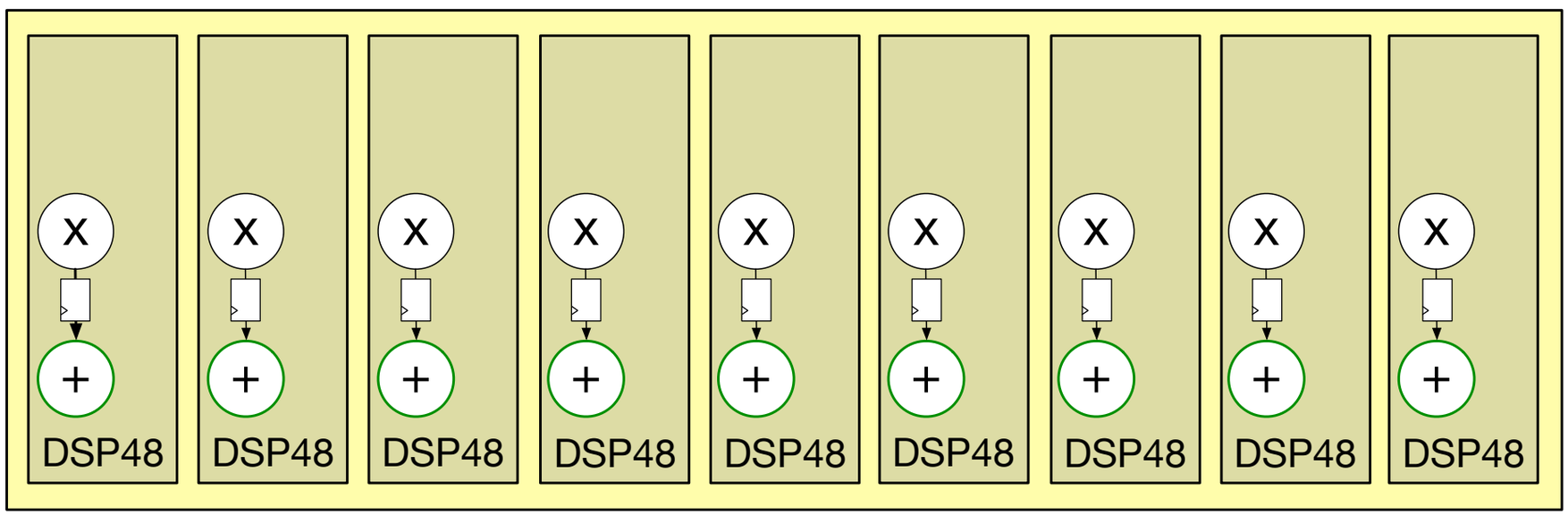
Our approach

- Exploit cascades aggressively!
- **URAM288** (our contribution)
  - Alternating A/B cascades of length-2
  - Both data + addresses cascades
    - Shift operation tricky!
  - Due to 72b width, and resource ratios, **idle cycles** available for realizing shifts

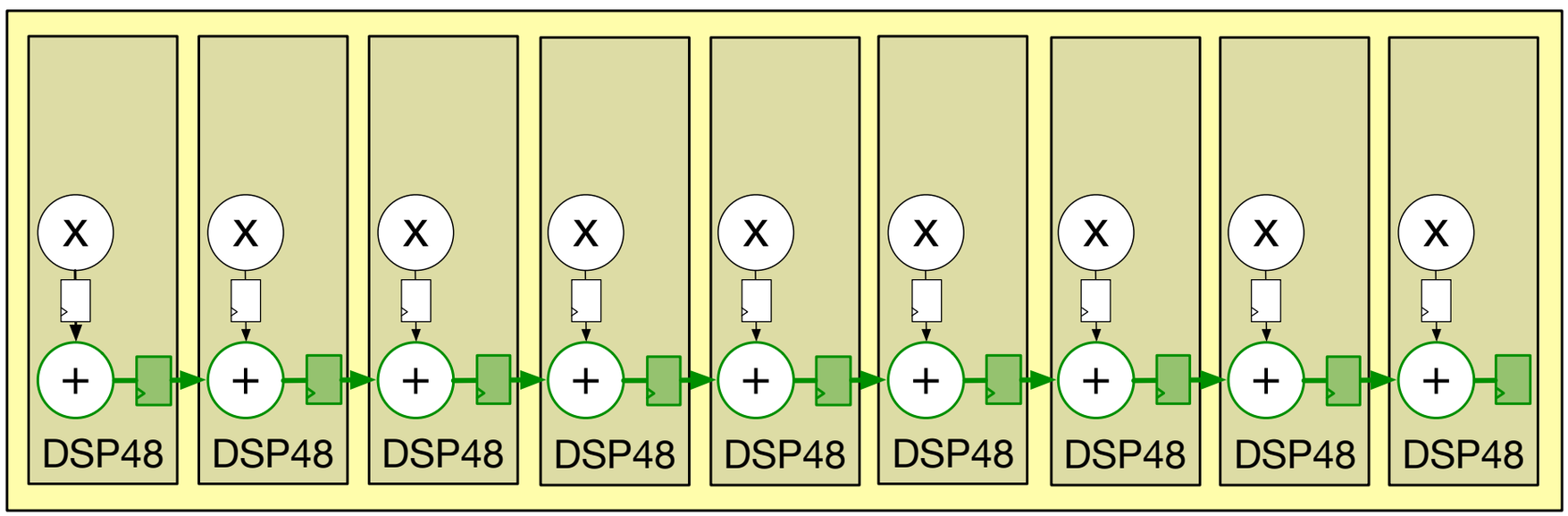


(c) URAM288 cascade (650 MHz)

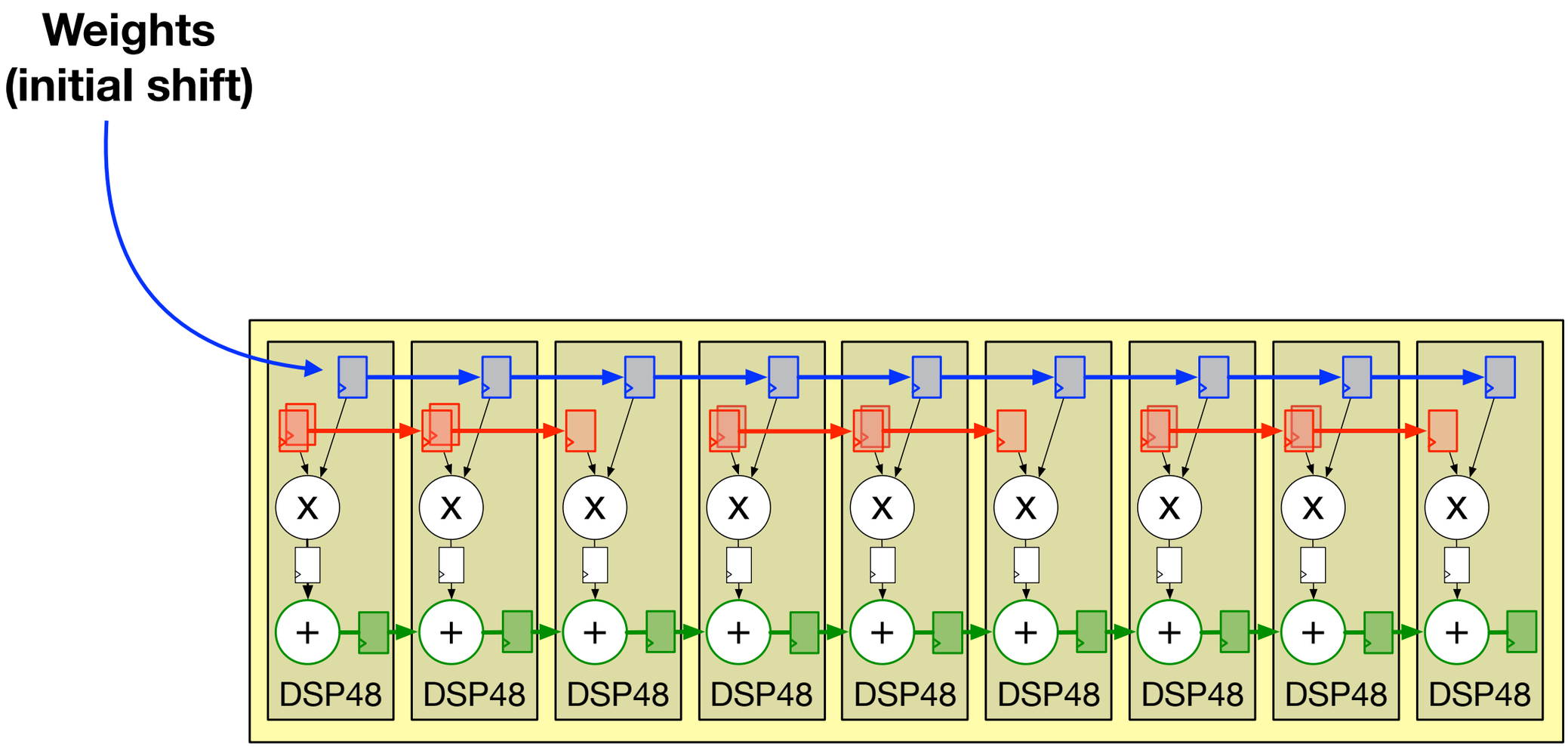
Putting it together



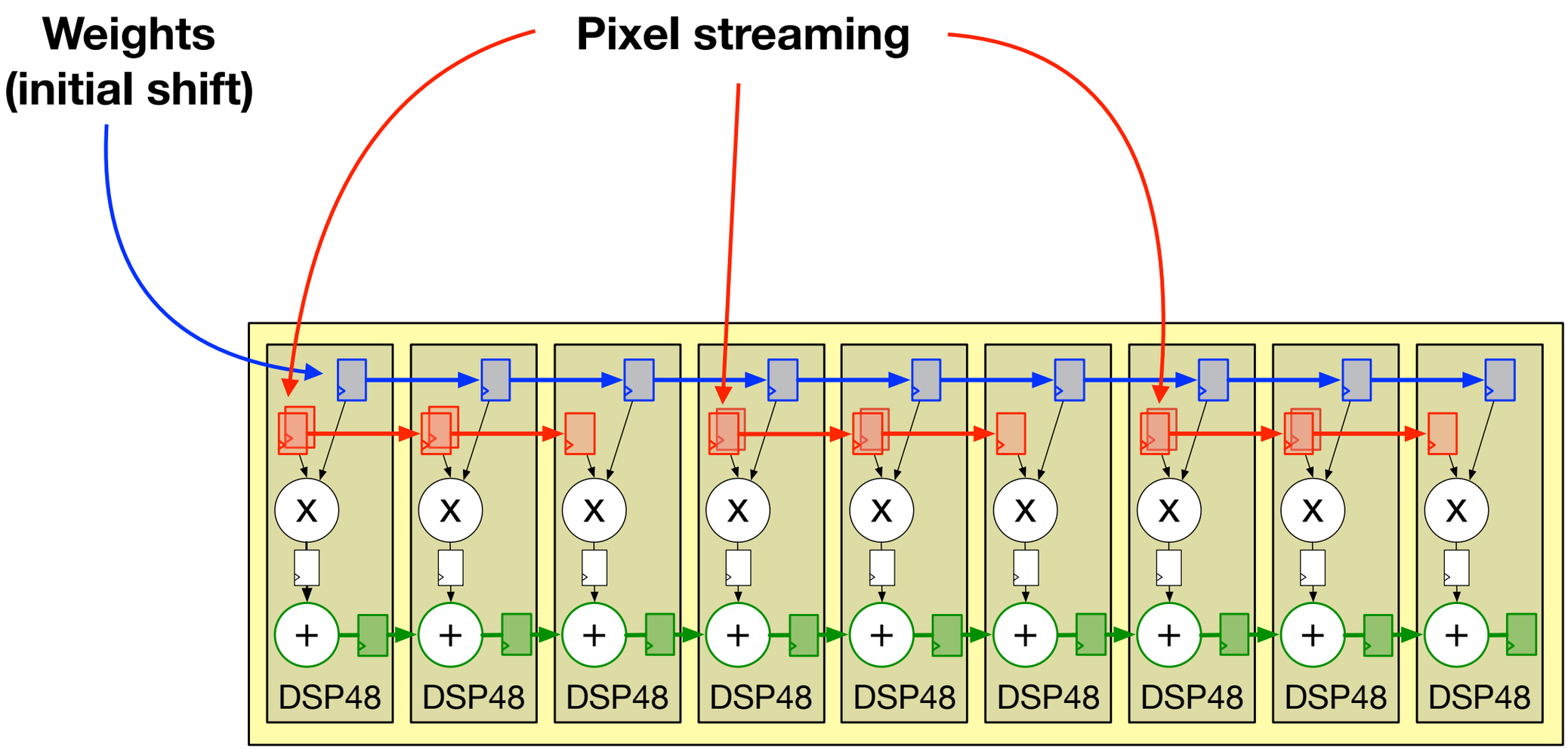
Putting it together



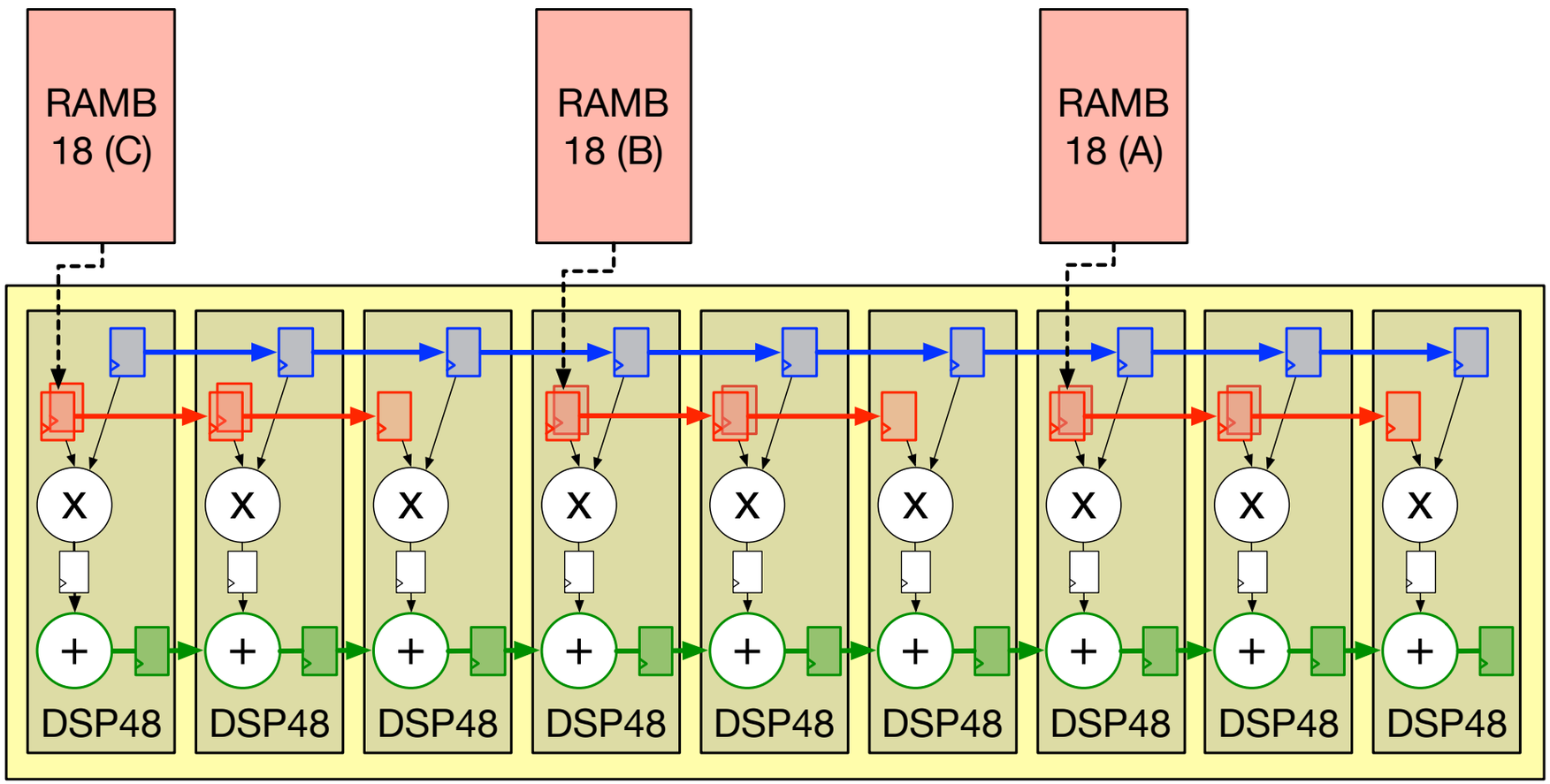
Putting it together



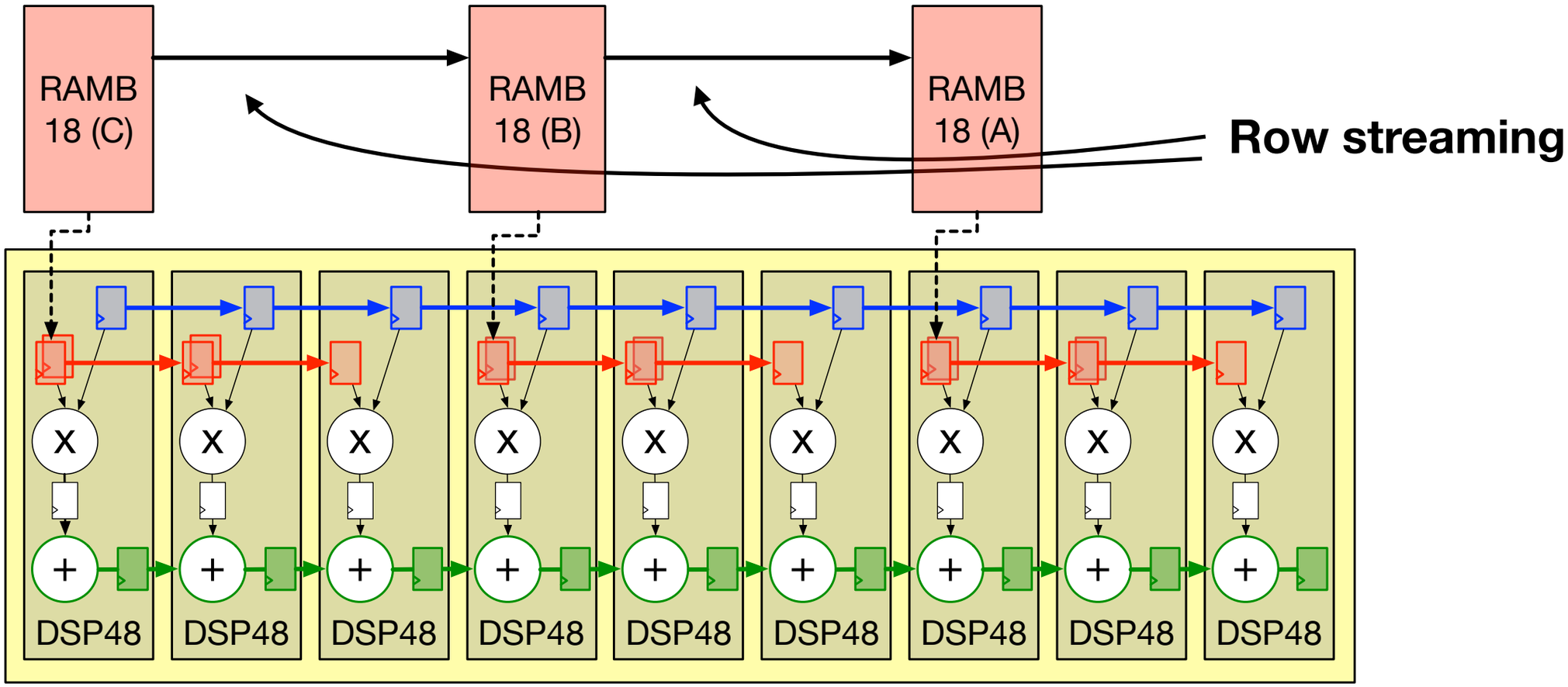
Putting it together



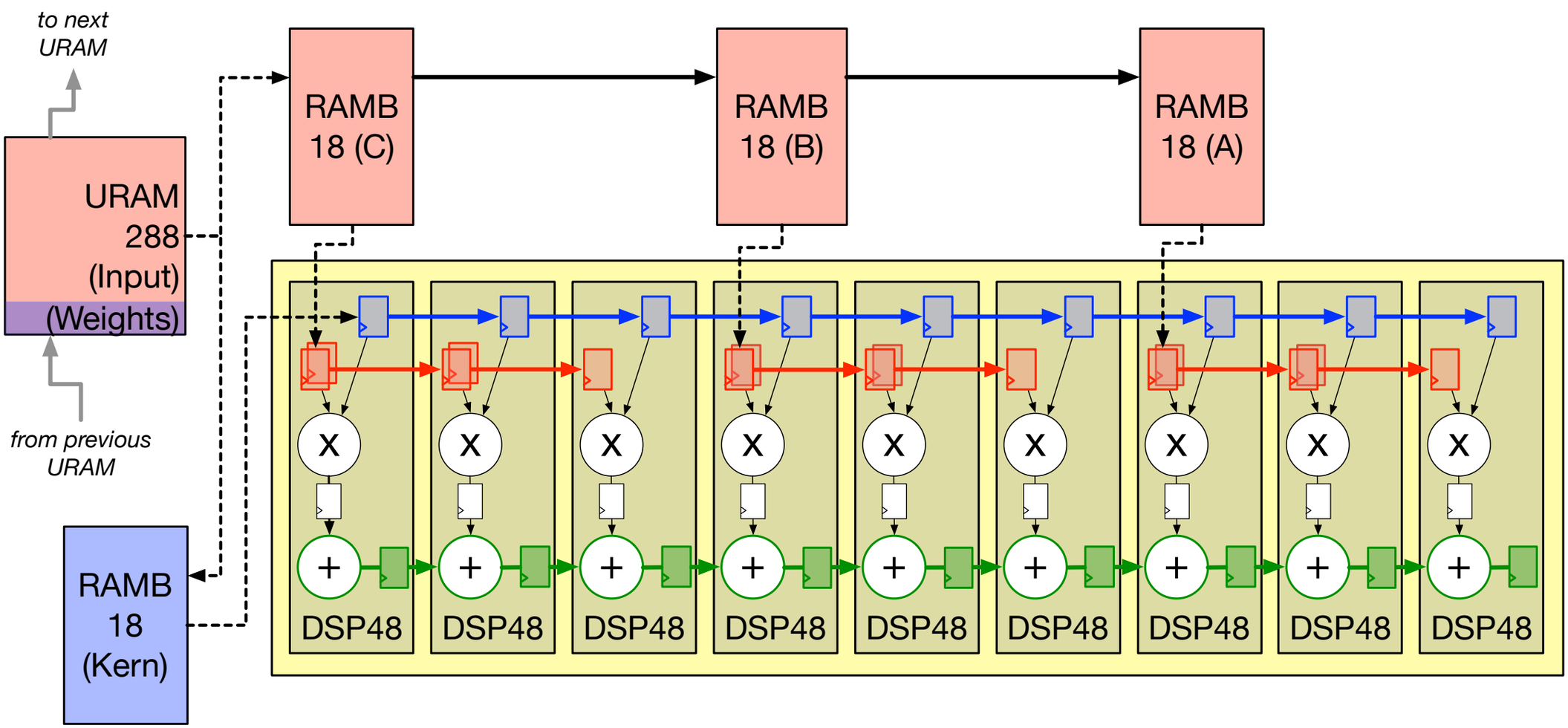
Putting it together



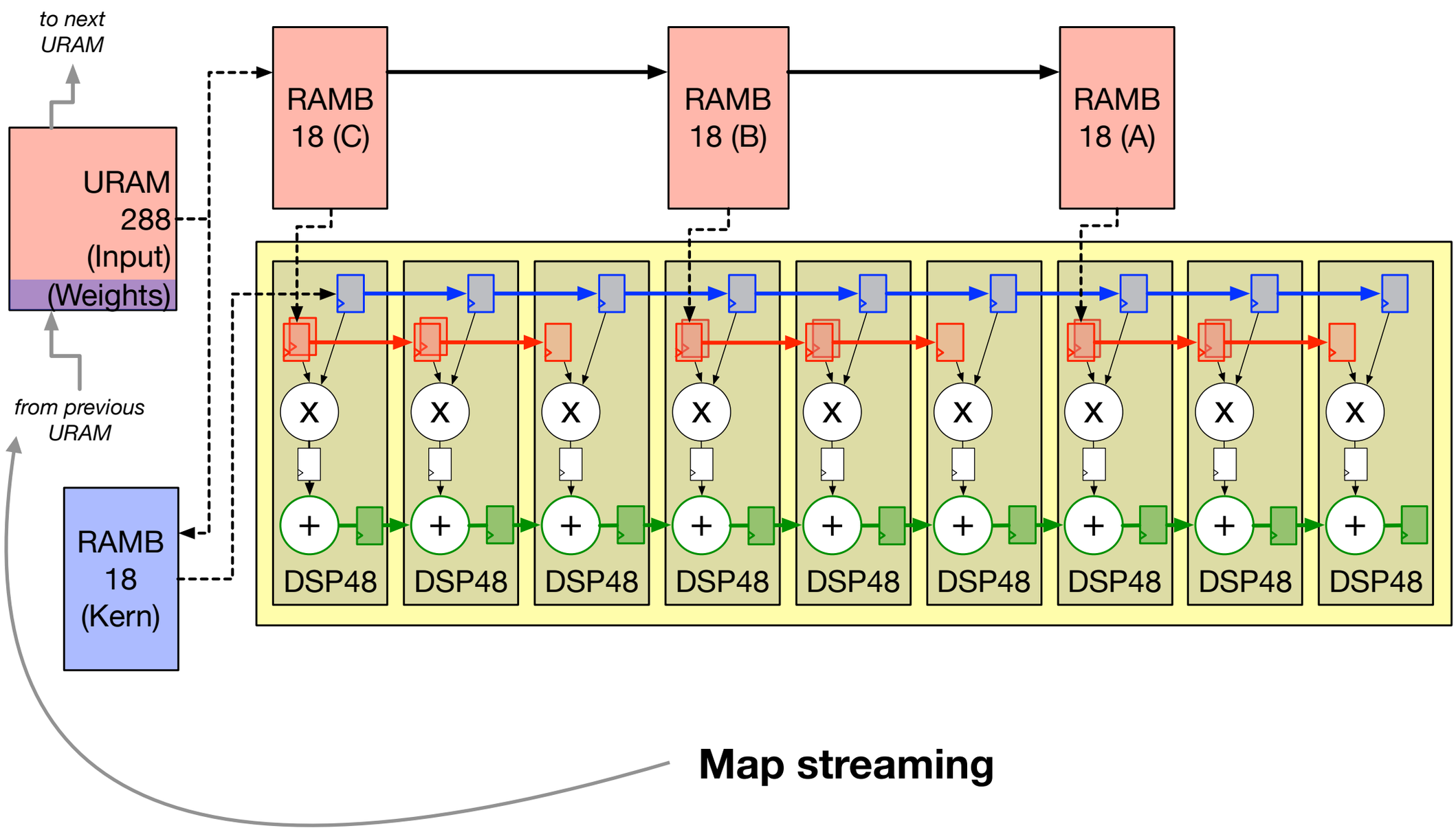
Putting it together



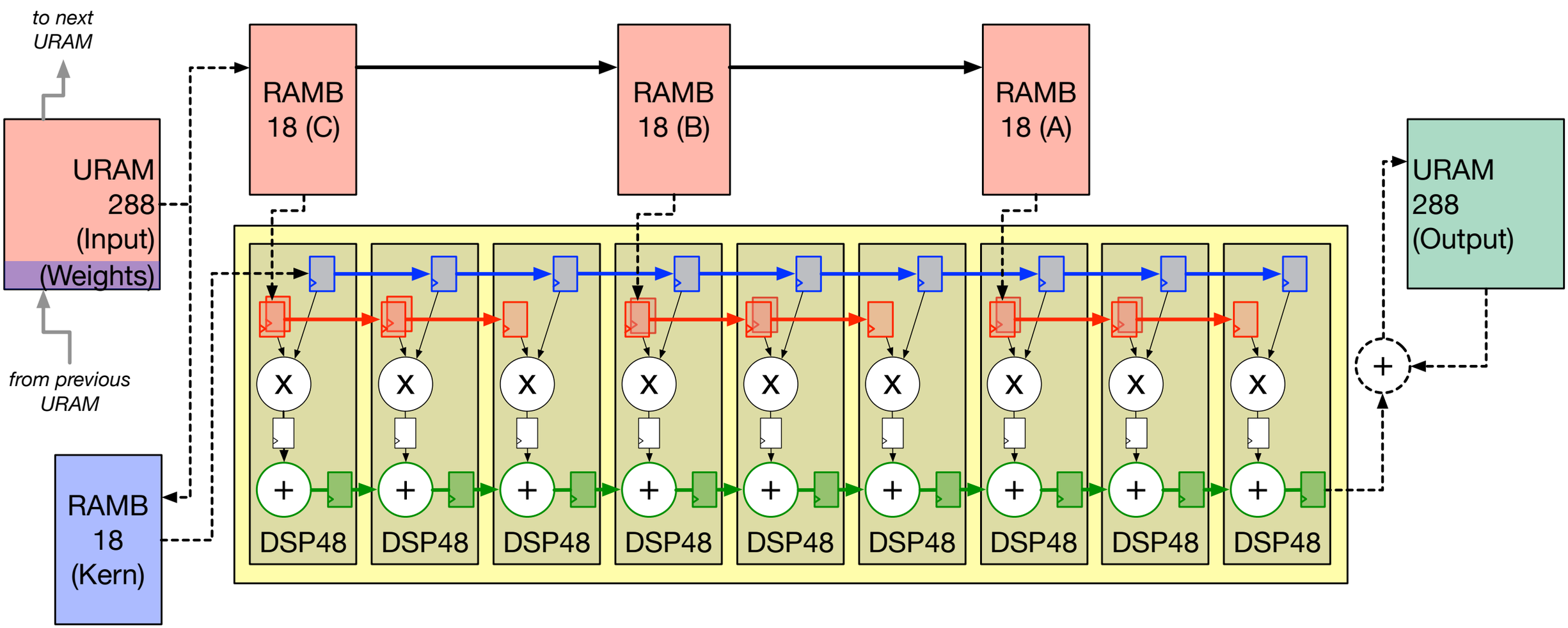
Putting it together



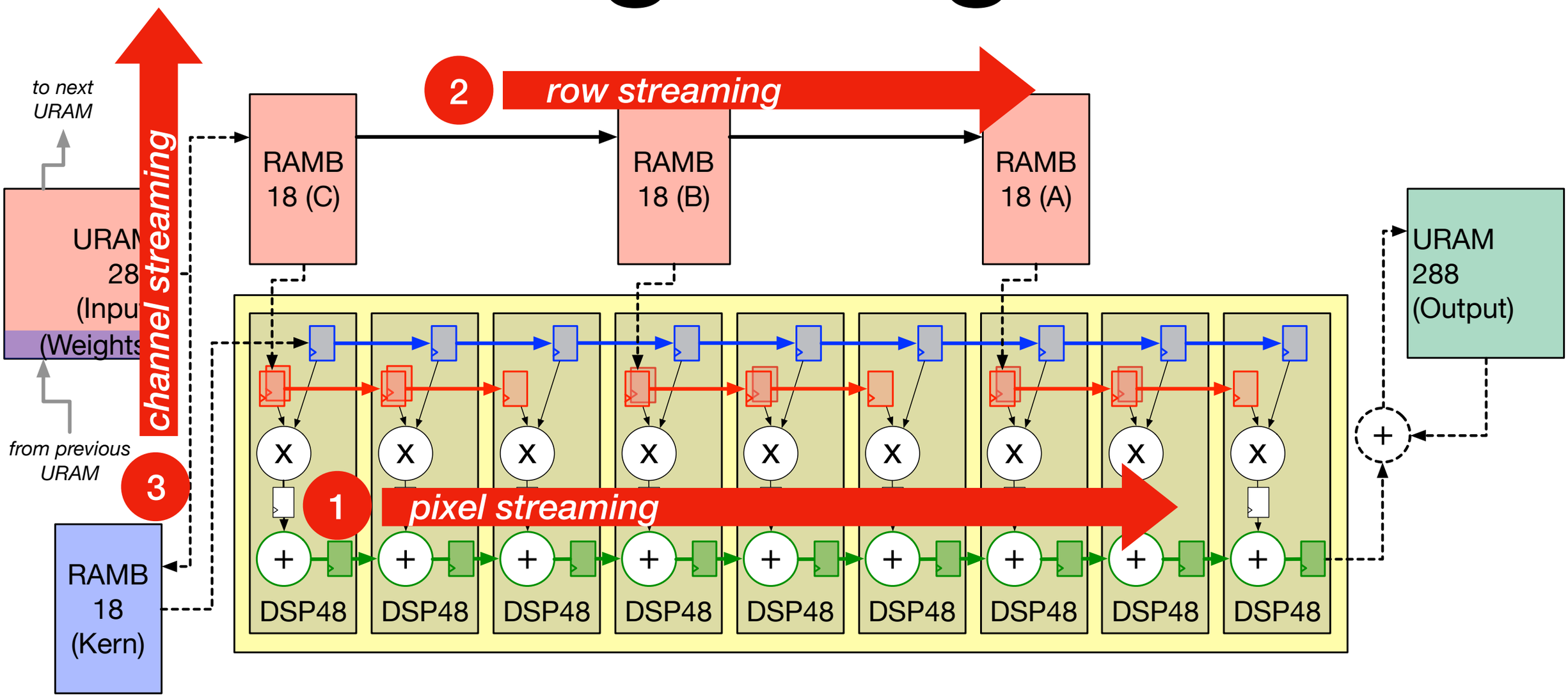
Putting it together



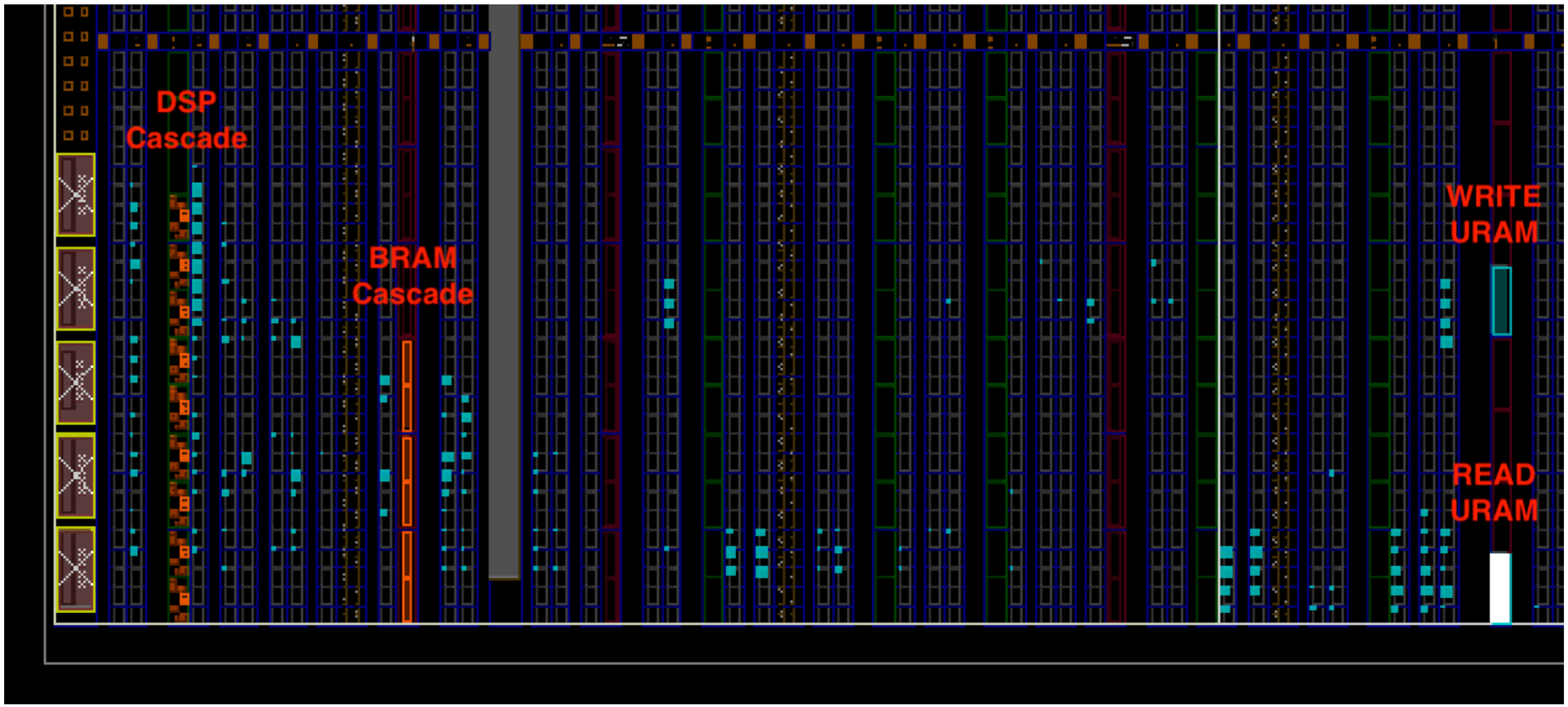
Putting it together



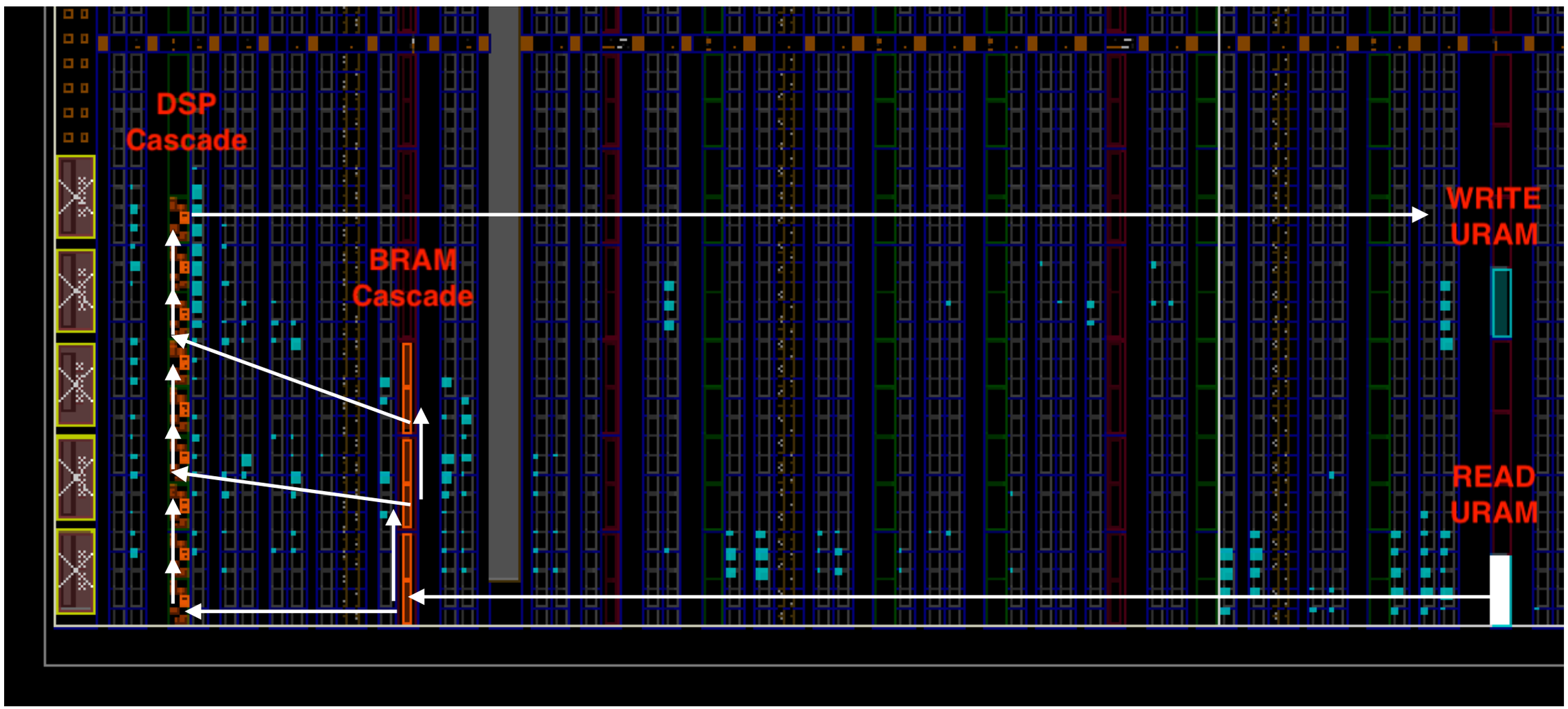
Putting it together



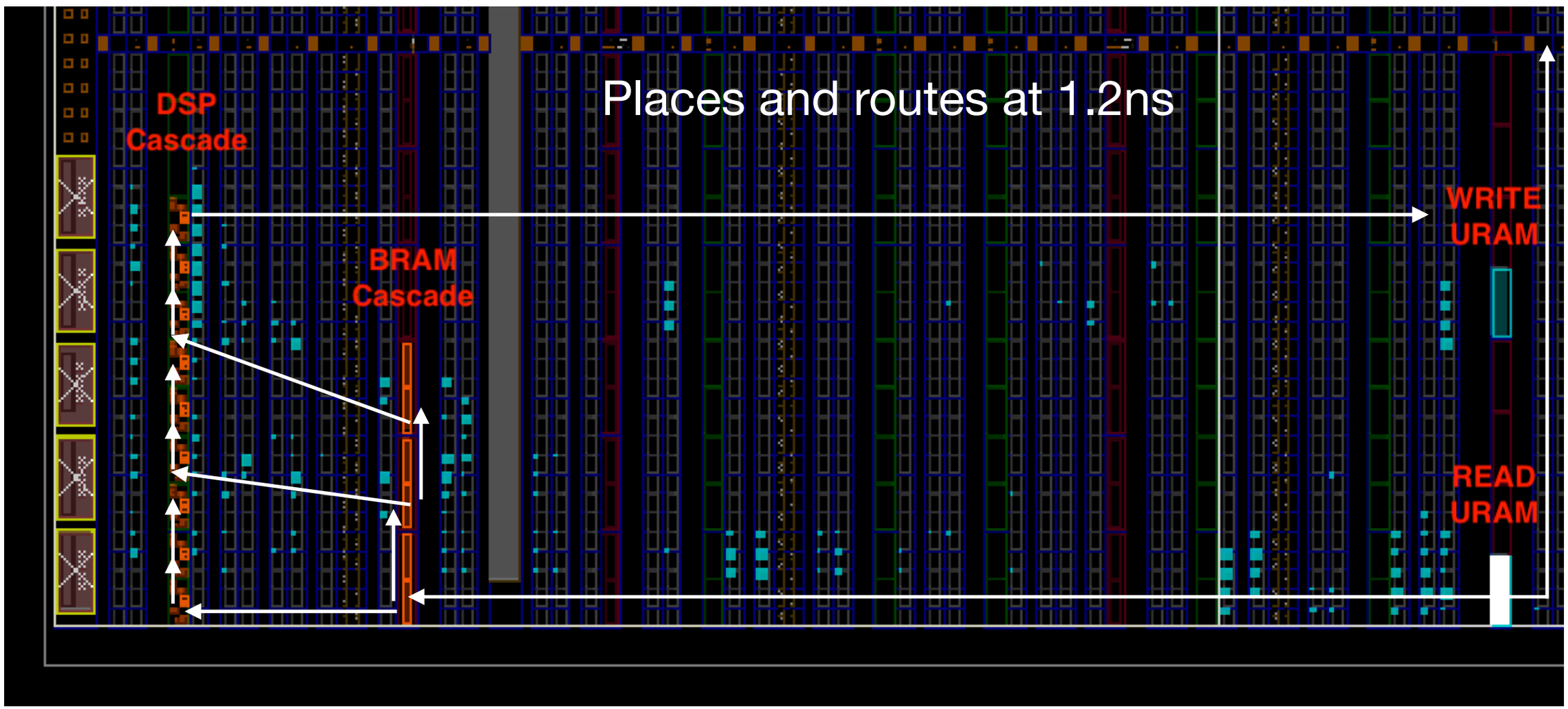
A 3x3 tile layout



A 3x3 tile layout

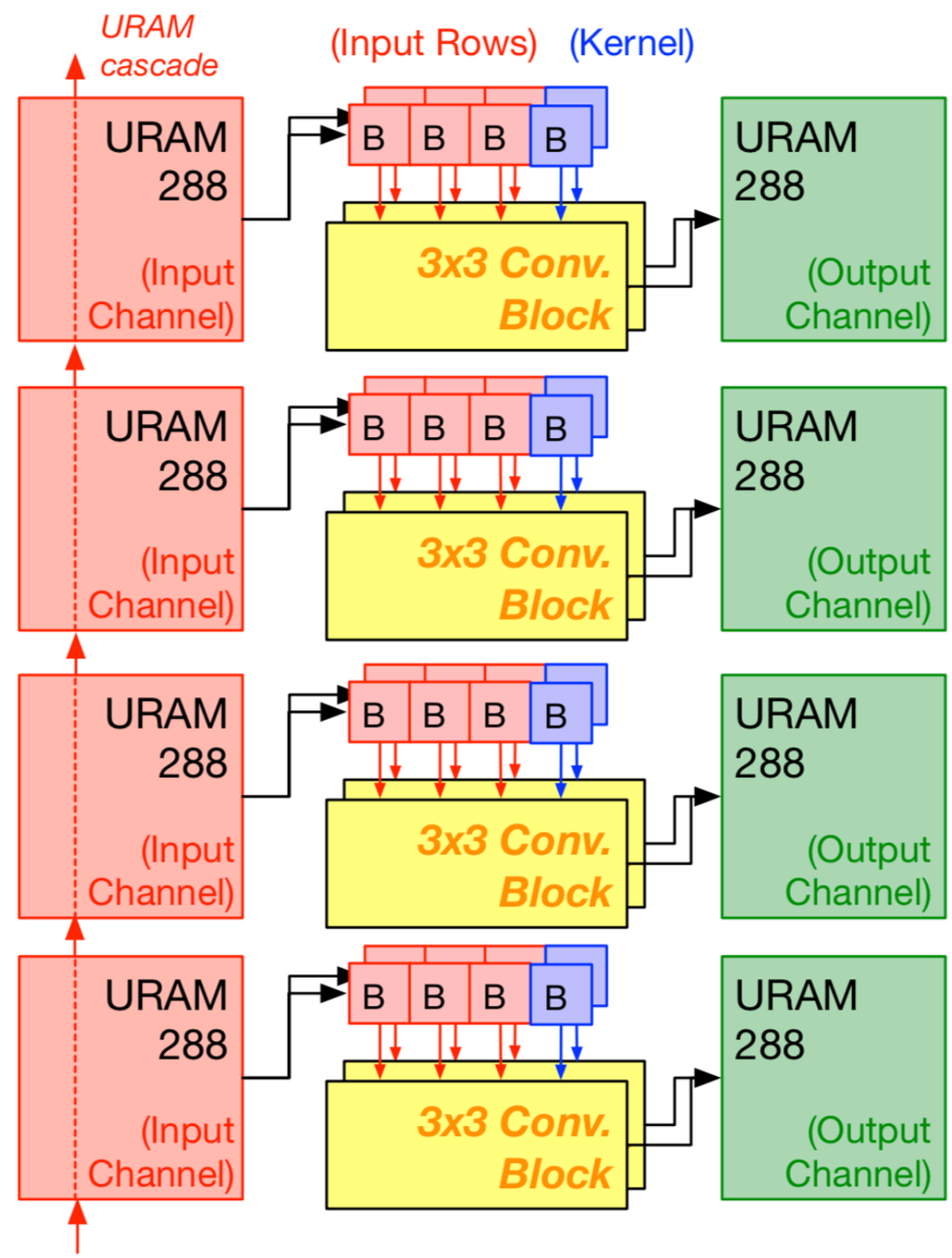


A 3x3 tile layout



Tiling the design

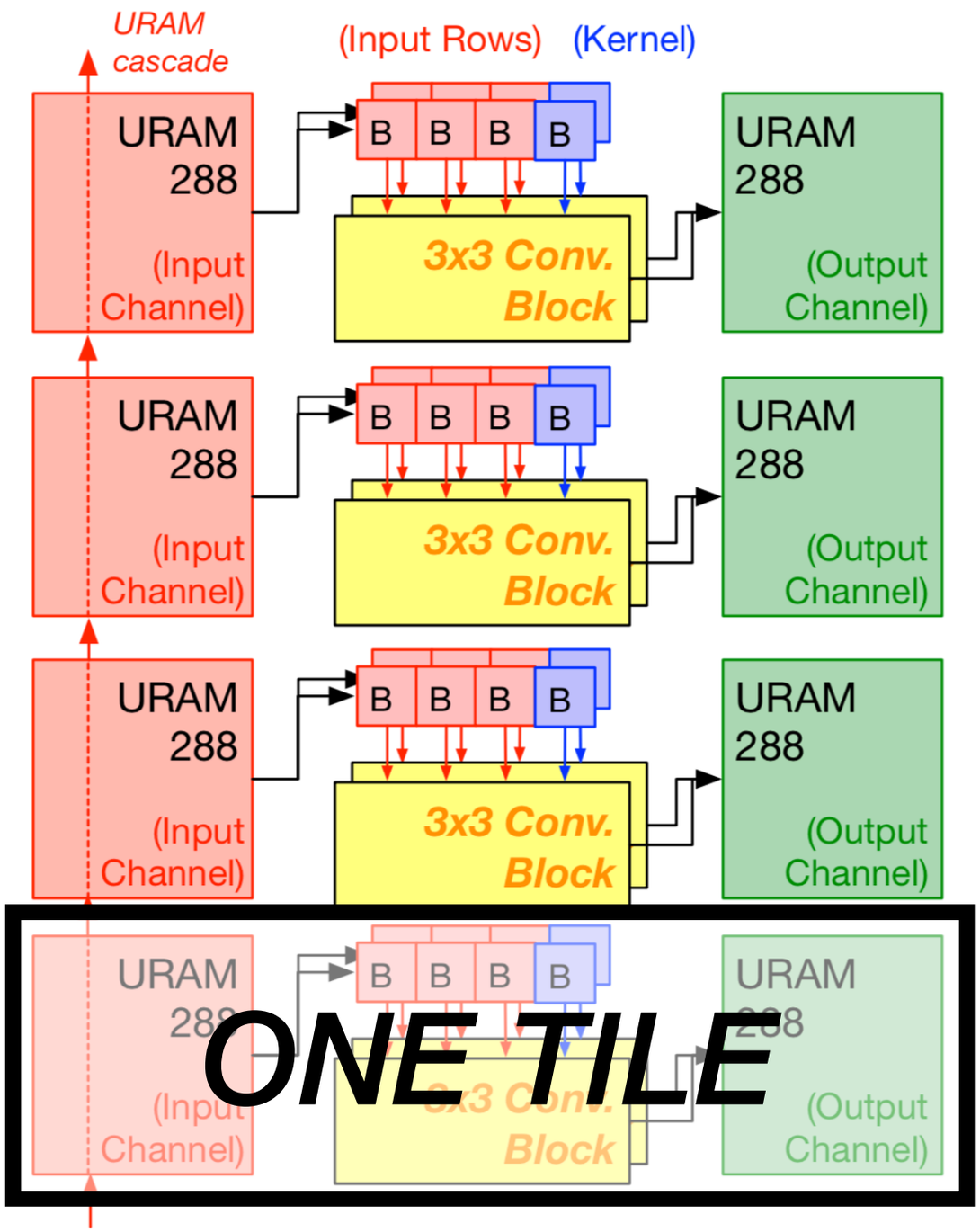
- VU37P device has specific resource mix
  - *For each URAM, you get 4.2 BRAMs and 9.4 DSP48s*
- Repeating pattern must conform to this ratio
  - One tile: 2 URAMs, 8 BRAMs, 18 DSPs
- Physical layout XDC constraints must account for irregular column arrangement of hard resources



(a) 3×3 Convolution Tiles

Tiling the design

- VU37P device has specific resource mix
  - *For each URAM, you get 4.2 BRAMs and 9.4 DSP48s*
- Repeating pattern must conform to this ratio
  - One tile: 2 URAMs, 8 BRAMs, 18 DSPs
- Physical layout XDC constraints must account for irregular column arrangement of hard resources

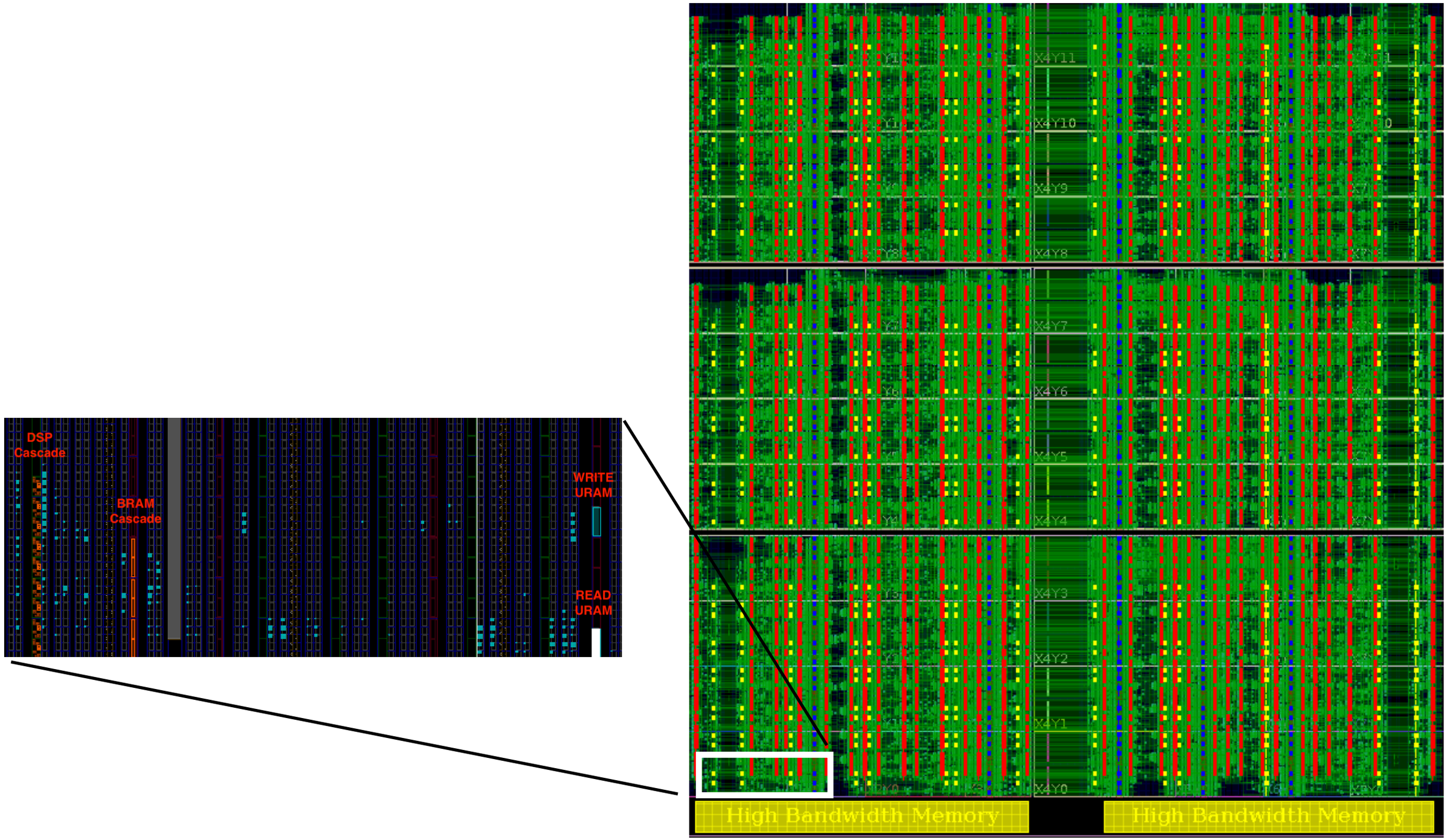


(a) 3×3 Convolution Tiles

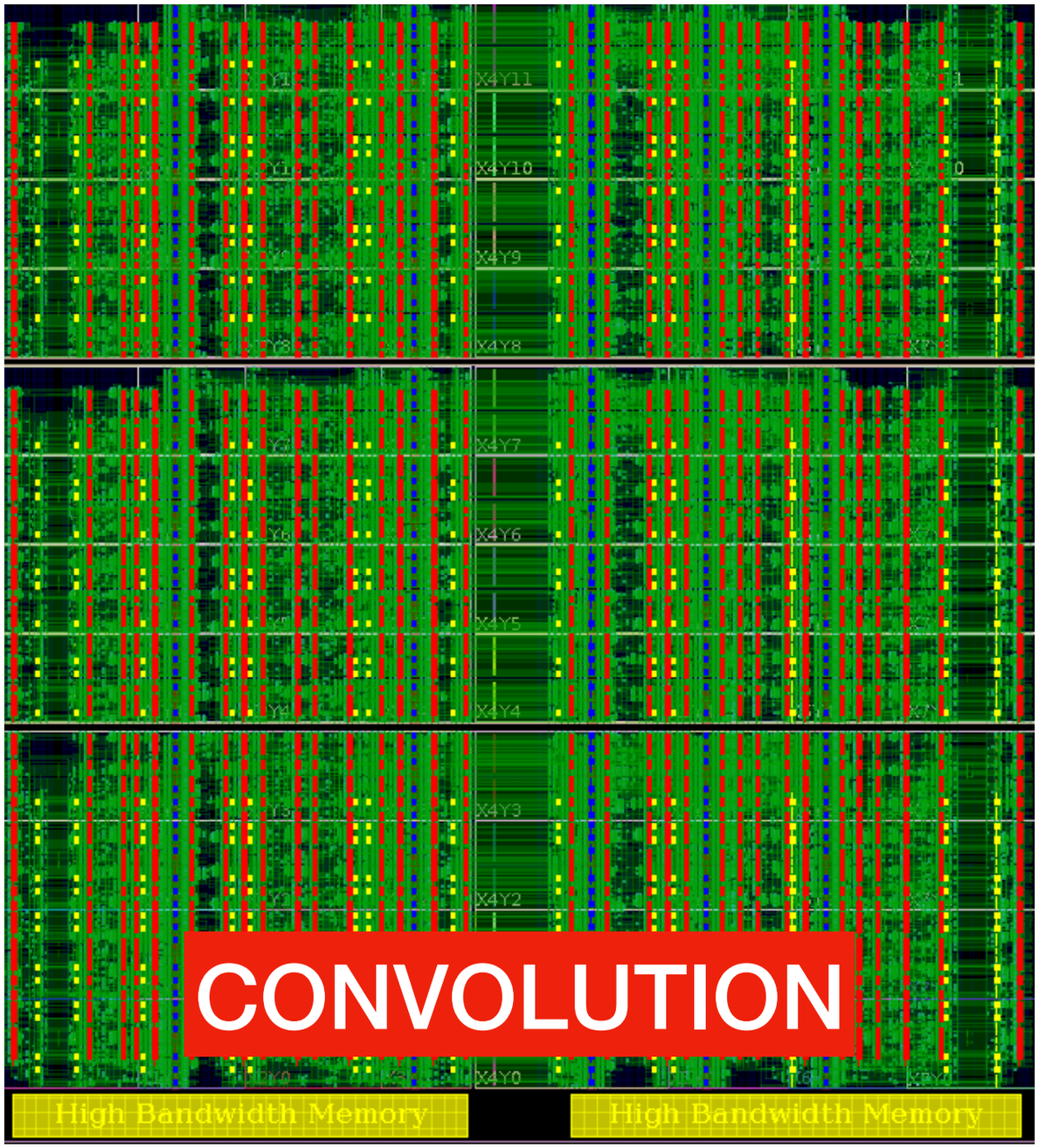
Matrix-Matrix Multiplication

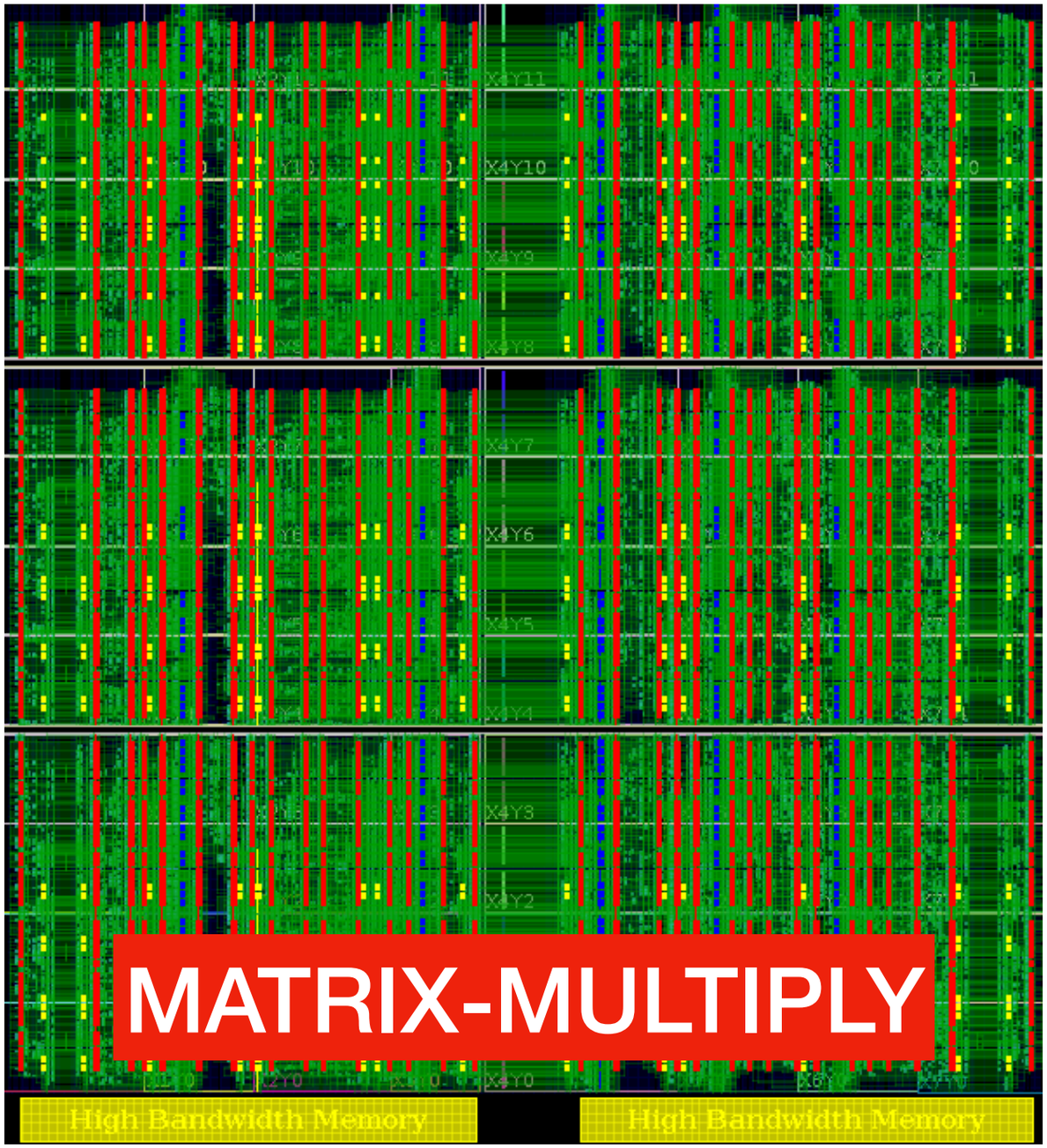
- Limited Reuse opportunities
- Split large matrix across URAMs
  - Each URAM stores a set of complete rows —> allows result vector to be independently processed.
- Partial vector results then circulated across the chip in a ring-like fashion —> using BRAM cascades
- URAM cascades only used for loading matrix at start

VU37P Layout

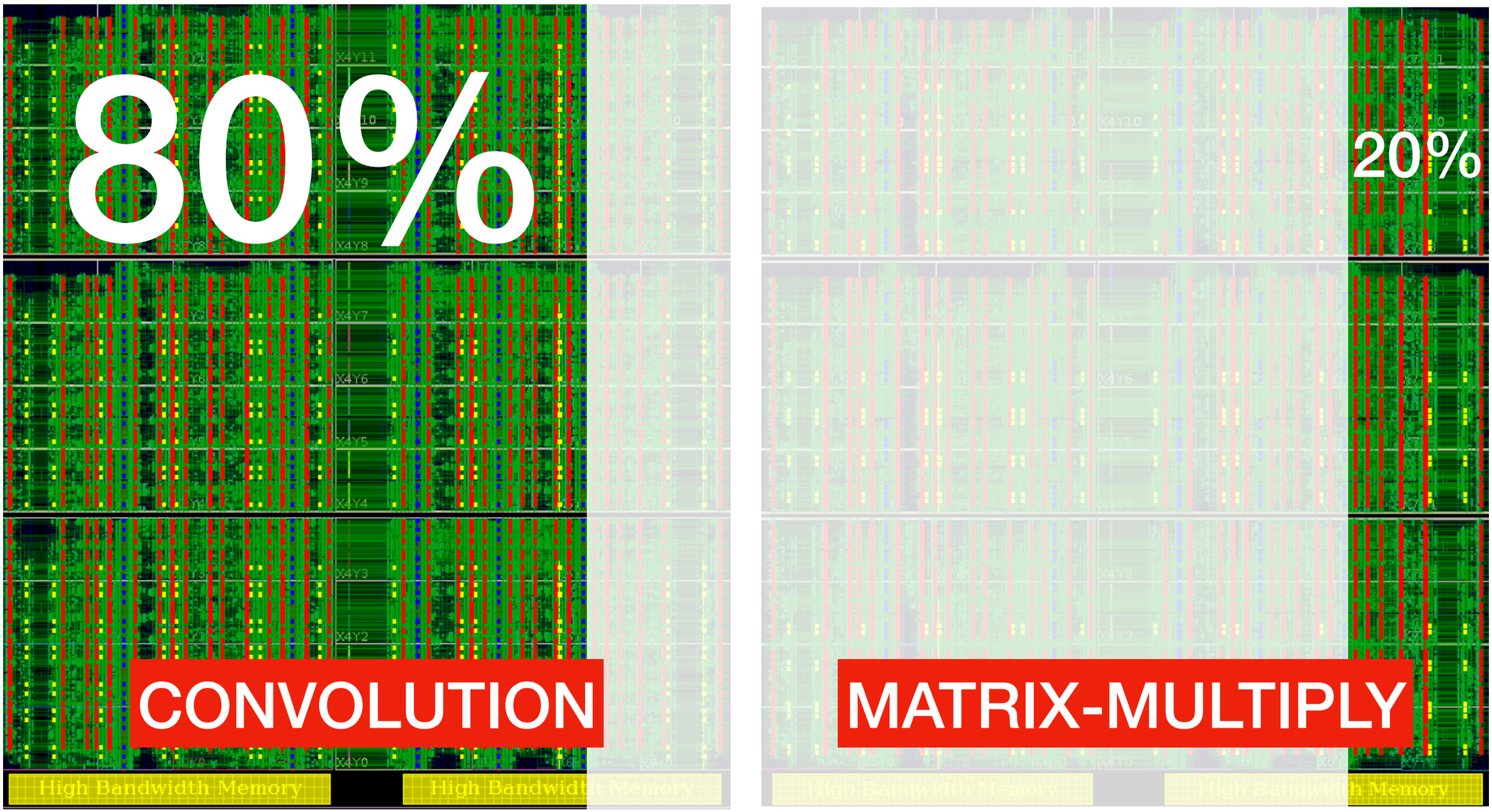


VU37P FPGA Mapping

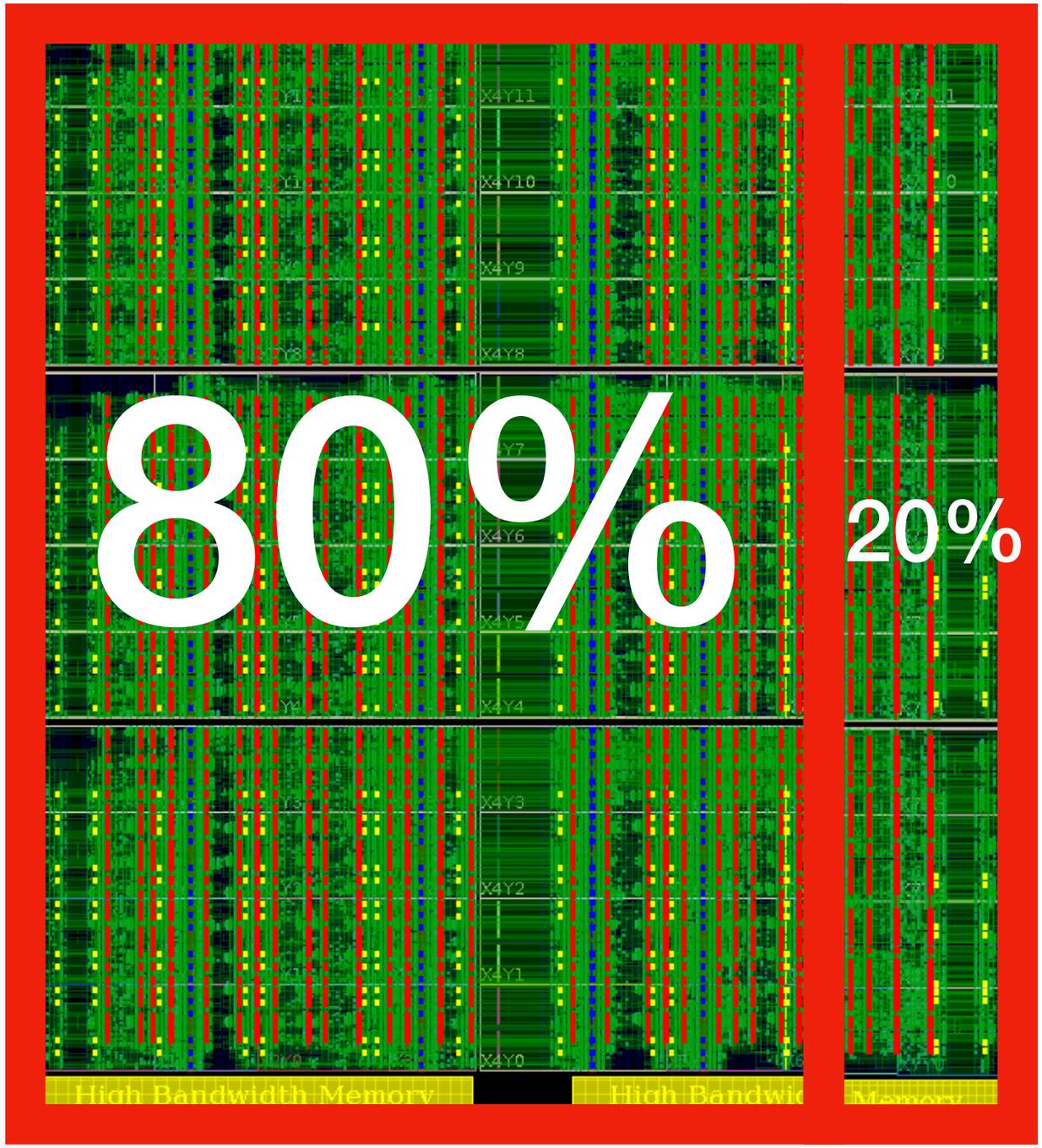




VU37P FPGA Mapping



VU37P FPGA Mapping

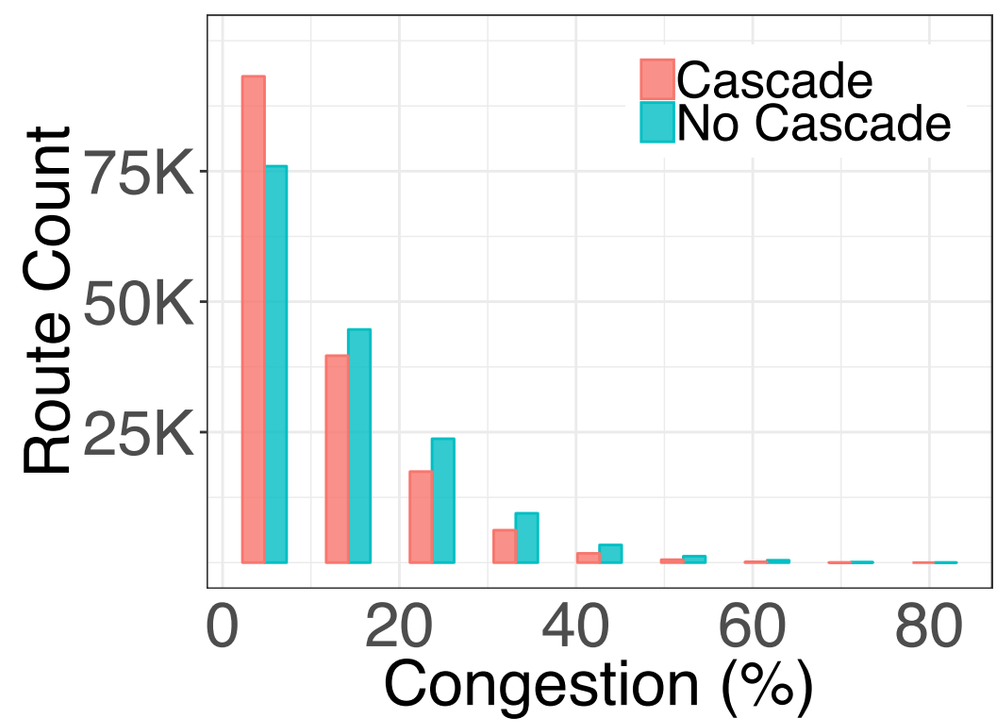


Effect of using cascades

Design	Size	LUTs			FFs			Clk (ns)			Net Util. (%)		
		Fabric	Cascade	%	Fabric	Cascade	%	Fabric	Cascade	%	Fabric	Cascade	%
Convolution	Block	325	327	0%	1.3K	1K	30%	0.9	0.9	0%	0.01	0.01	0%
	Tile (2 blocks)	424	435	-2%	1.9K	1.5K	26%	0.9	1	-10%	0.02	0.02	0%
	Full-Chip	20.9K	21.1K	-1%	95.3K	72.1K	32%	1.4	1.4	0%	12.8	9.6	33%
Matrix-Vector Multiplication	Block	98	98	0%	775	688	12%	1	0.9	10%	0.01	0.01	0%
	Tile (4 blocks)	375	374	0%	2.3K	1.9K	21%	1.1	0.9	22%	0.04	0.05	-8%
	Full-Chip	90.2K	90.2K	0%	56.8K	46.6K	21%	1.5	1.3	15%	9.3	8	16%

TABLE II: Resource and Frequency Trends for Convolution and Matrix Vector Multiplication blocks, tiles and full-chip layouts.

- Registers in hard interconnect save us fabric registers for other pipelining needs
- Clock period marginally better
- Obvious reduction in interconnect utilization

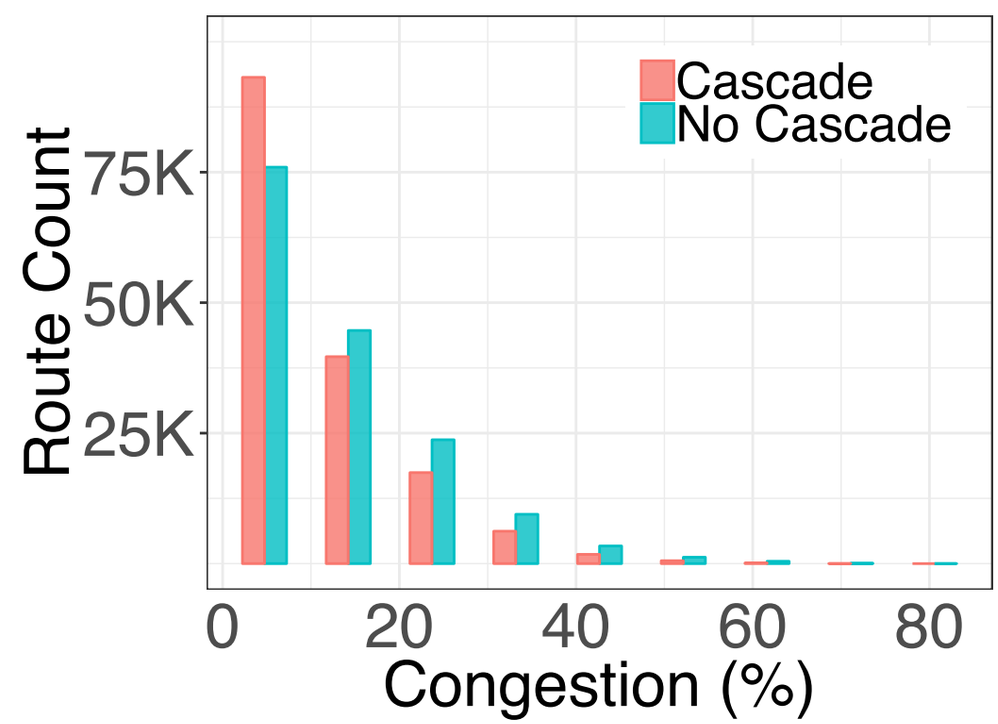


Effect of using cascades

Design	Size	LUTs			FFs			Clk (ns)			Net Util. (%)		
		Fabric	Cascade	%	Fabric	Cascade	%	Fabric	Cascade	%	Fabric	Cascade	%
Convolution	Block	325	327	0%	1.3K	1K	30%	0.9	0.9	0%	0.01	0.01	0%
	Tile (2 blocks)	424	435	-2%	1.9K	1.5K	26%	0.9	1	-10%	0.02	0.02	0%
	Full-Chip	20.9K	21.1K	-1%	95.3K →	72.1K	32%	1.4	1.4	0%	12.8 →	9.6	33%
Matrix-Vector Multiplication	Block	98	98	0%	775	688	12%	1	0.9	10%	0.01	0.01	0%
	Tile (4 blocks)	375	374	0%	2.3K	1.9K	21%	1.1	0.9	22%	0.04	0.05	-8%
	Full-Chip	90.2K	90.2K	0%	56.8K →	46.6K	21%	1.5 →	1.3	15%	9.3 →	8	16%

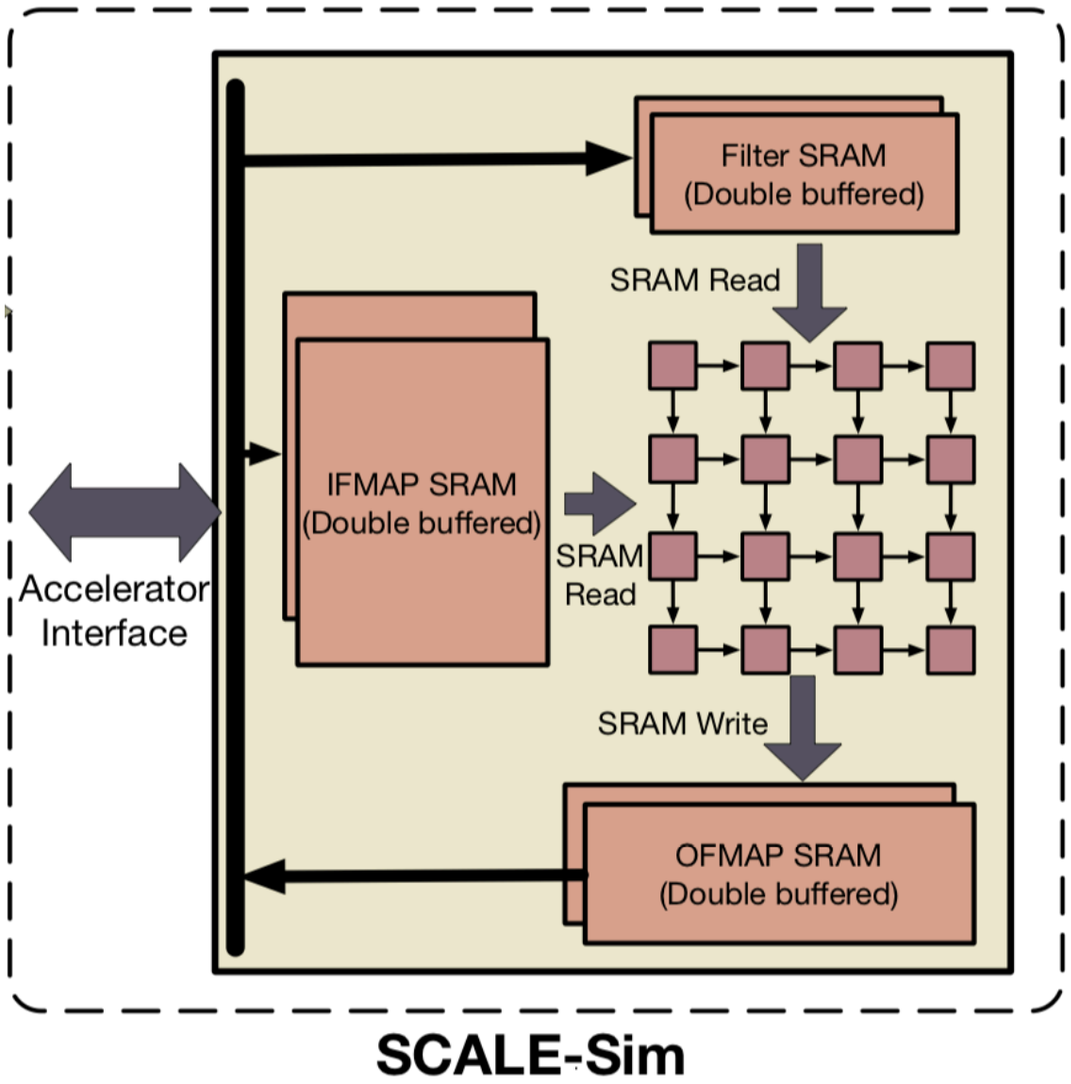
TABLE II: Resource and Frequency Trends for Convolution and Matrix Vector Multiplication blocks, tiles and full-chip layouts.

- Registers in hard interconnect save us fabric registers for other pipelining needs
- Clock period marginally better
- Obvious reduction in interconnect utilization



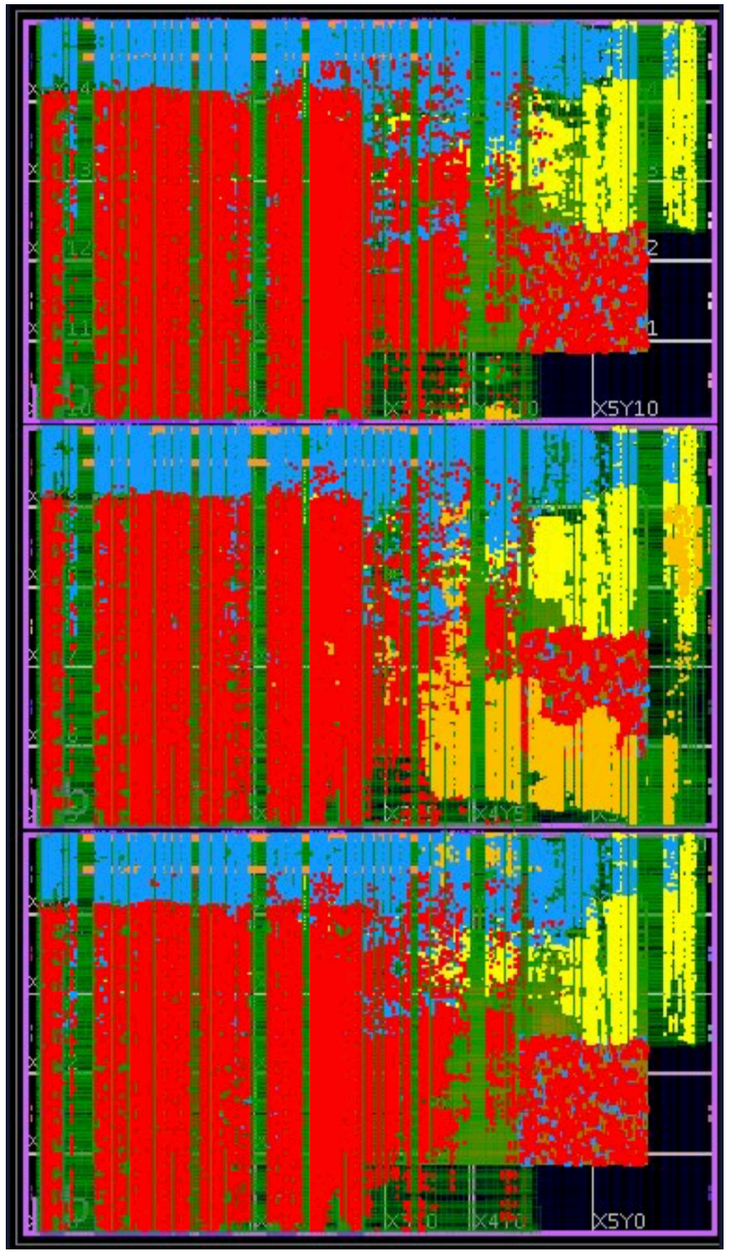
Evaluation Methodology

- We use the SCALE-Sim cycle-accurate simulator
  - https://github.com/ARM-software/SCALE-Sim
  - Map URAMs -> IFMAP/ OFMAP SRAMs
  - BRAMs and DSP cascades => systolic array links
- VU37P can fit systolic array of size **960x9** (conv), **480x9** (mm)



Xilinx SuperTile

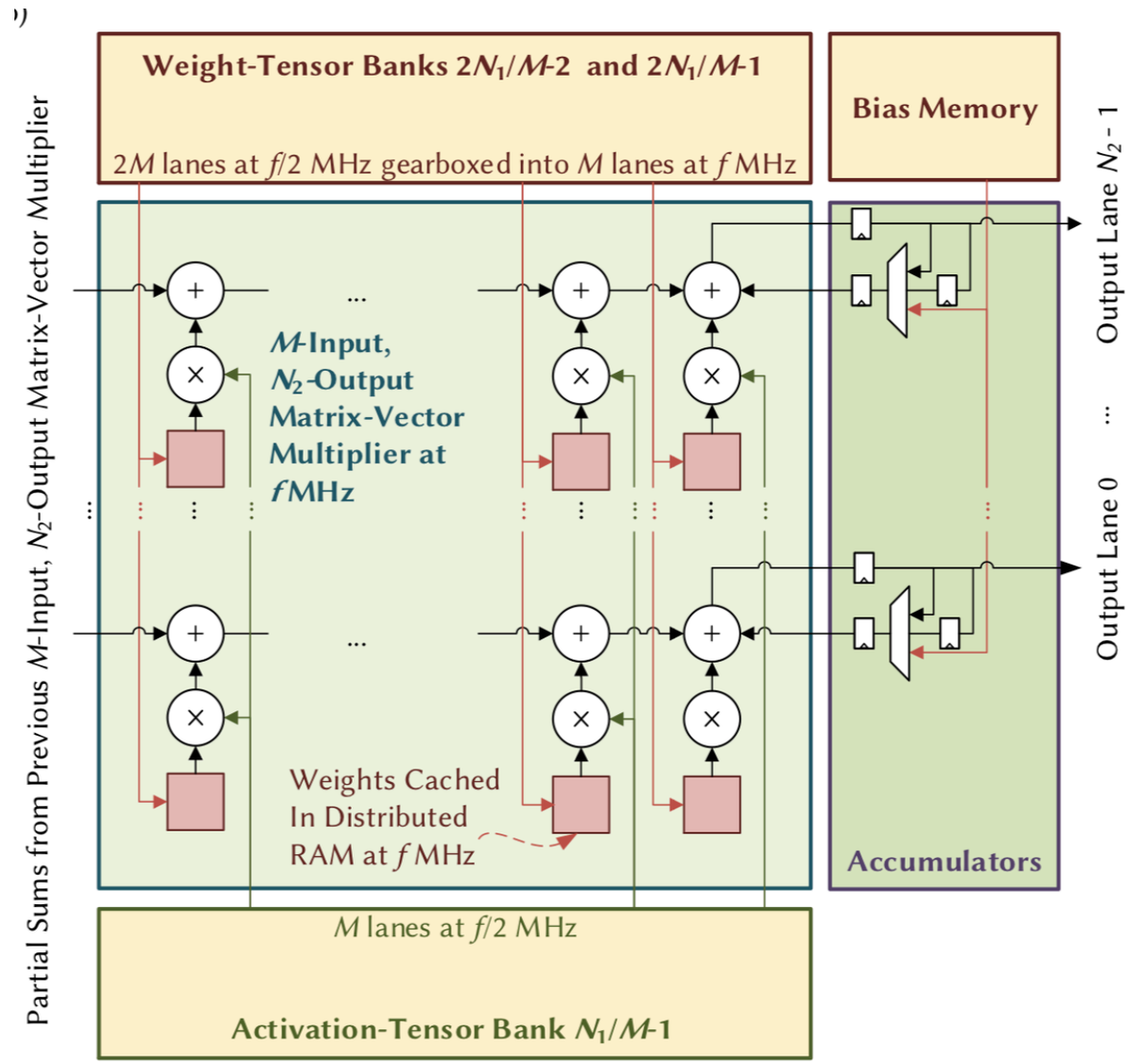
- GoogLeNet v1 mapped to VU9P FPGA (Amazon F1)
- 3046 images/s + 3.3ms latency
- Scorching **720 MHz** operation!
- Mind-numbing **88%** overall efficiency



http://isfpga.org/slides/Compute-Efficient_Neural-Network_Acceleration.pdf

Why is SuperTile so good?

- **Base Design**
  - High-frequency layout using DSP cascades,
  - Systolic data movement in fabric
- Throughput boost
  - Decompose the systolic array into sub-arrays
  - Perform pipelining across CNN layers
  - Sacrifice some latency to significantly boost throughput!



https://dl.acm.org/citation.cfm?id=3293925

MLPerf Benchmarks

- *Caveat*: Result not verified by MLPerf. MLPerf name and logo are trademarks. See www.mlperf.org for more information.

- Different ML workloads

- Various domains

- Different compute complexity

Topology	Operation Count			Storage (bytes)	
(MLPerf)	All	Conv	MM	$\sum$Wts.	Activ.
AlphaGoZero	352M	352M	353K	1.5M	92K
DeepSpeech2	1.7G	1.7G	74K	355K	6.5M
FasterRCNN	3.5G	1.6G	1.8G	13M	802K
NCF	11M	0	11M	11M	138K
Resnet50	3.4G	1.6G	1.8G	25M	802K
Sentimental	210M	0	210M	172K	30.7M
GoogLeNet	1.3G	1.3G	46M	6.8M	200K

MLPerf Benchmarks

- *Caveat*: Result not verified by MLPerf. MLPerf name and logo are trademarks. See www.mlperf.org for more information.

- Different ML workloads

- Various domains

- Different compute complexity

TABLE I: MLPerf and GoogLeNet benchmark characteristics.

Topology	**Operation Count**			**Storage** (bytes)	
(MLPerf)	All	Conv	MM	$\sum$Wts.	Activ.
AlphaGoZero	352M	352M	353K	1.5M	92K
DeepSpeech2	1.7G	1.7G	74K	355K	6.5M
FasterRCNN	3.5G	1.6G	1.8G	13M	802K
NCF	11M	0	11M	11M	138K
Resnet50	3.4G	1.6G	1.8G	25M	802K
Sentimental	210M	0	210M	172K	30.7M
Transformer	113M	35M	78M	77M	4096
GoogLeNet	1.3G	1.3G	46M	6.8M	200K

MLPerf Benchmarks

- *Caveat*: Result not verified by MLPerf. MLPerf name and logo are trademarks. See www.mlperf.org for more information.

- Different ML workloads

- Various domains

- Different compute complexity

35 MB URAM Capacity!

TABLE I: MLPerf and Goog[illegible] b[illegible]hmark characteristics.

Topology	Operation Count			Storage (bytes)	
(MLPerf)	All	Conv	MM	$\sum$Wts.	Activ.
AlphaGoZero	352M	352M	353K	1.5M	92K
DeepSpeech2	1.7G	1.7G	74K	355K	6.5M
FasterRCNN	3.5G	1.6G	1.8G	13M	802K
NCF	11M	0	11M	11M	138K
Resnet50	3.4G	1.6G	1.8G	25M	802K
Sentimental	210M	0	210M	172K	30.7M
Transformer	113M	35M	78M	77M	4096
GoogLeNet	1.3G	1.3G	46M	6.8M	200K

Performance Results

Topology (MLPerf)	**Ratio** (Conv:MM)	**Cycles**	**Time** (ms)	**Tput.** (inference/s)
AlphaGoZero	90:10	60K	0.09	10K
DeepSpeech2	60:40	1.2M	1.89	528
FasterRCNN	30:70	903K	1.38	719
NCF	0:100	2.4K	0.003	260K
Resnet50	30:70	848K	1.3	766
Sentimental	100:0	24K	0.037	27K
GoogLeNet (Us)	70:30	261K	0.40	2.4K
GoogLeNet (SuperTile [30])	-	-	3.3	3K

Performance Results

Topology (MLPerf)	**Ratio** (Conv:MM)	**Cycles**	**Time** (ms)	**Tput.** (inference/s)
AlphaGoZero	90:10	60K	0.09	10K
DeepSpeech2	60:40	1.2M	1.89	528
FasterRCNN	30:70	903K	1.38	719
NCF	0:100	2.4K	0.003	260K
Resnet50	30:70	848K	1.3	766
Sentimental	100:0	24K	0.037	27K
GoogLeNet (Us)	70:30	261K	0.40	2.4K
GoogLeNet (SuperTile [30])	-	-	3.3	3K

Performance Results

Topology (MLPerf)	**Ratio** (Conv:MM)	**Cycles**	**Time** (ms)	**Tput.** (inference/s)
AlphaGoZero	90:10	60K	0.09	10K
DeepSpeech2	60:40	1.2M	1.89	528
FasterRCNN	30:70		1.38	719
NCF	0:100		0.003	260K
Resnet50	30:70		1.3	766
Sentimental	100:0		0.037	27K
GoogLeNet (Us)	70:30	261K	0.40	2.4K
GoogLeNet (SuperTile [30])	-	-	3.3	3K

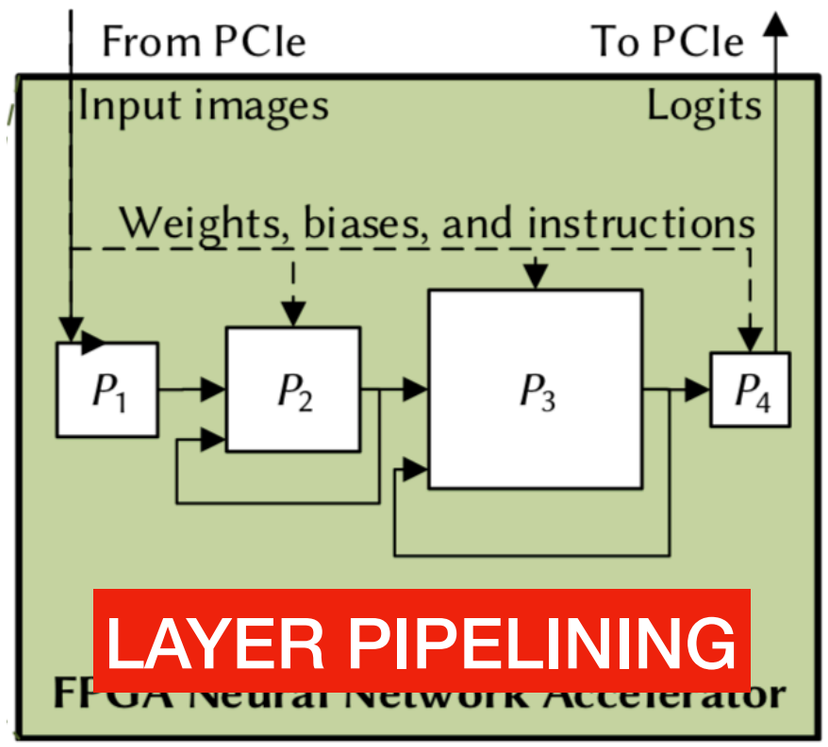
7x Lower Latency

Performance Results

Topology (MLPerf)	**Ratio** (Conv:MM)	**Cycles**	**Time** (ms)	**Tput.** (inference/s)
AlphaGoZero	90:10	60K	0.09	10K
DeepSpeech2	60:40	1.2M	1.89	528
FasterRCNN	30:70	903K	1.38	71
NCF	0:100	2.4K	0.003	26
Resnet50	30:70	848K	1.3	76
Sentimental	100:0	24K	0.037	27K
GoogLeNet (Us)	70:30	261K	0.40	2.4K
GoogLeNet (SuperTile [30])	-	-	3.3	3K

30% Lower Throughput

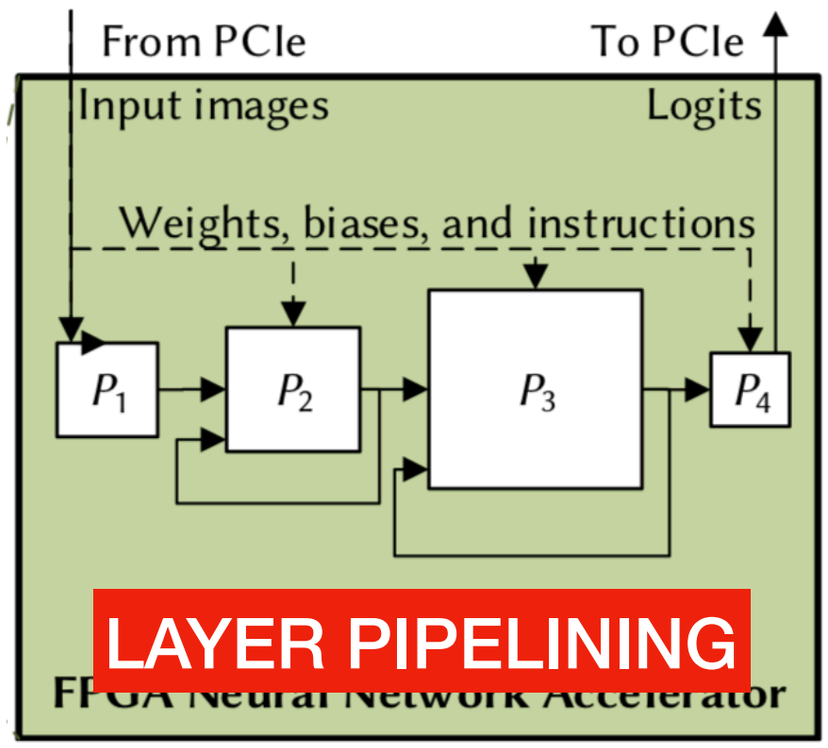
Performance Results

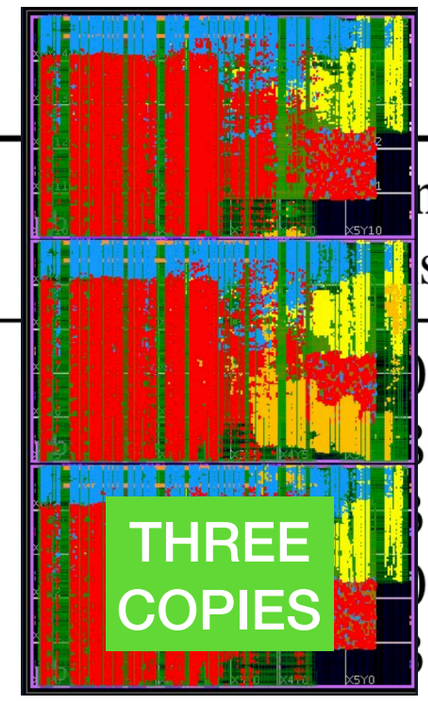


io nv:MM)	Cycles	Time (ms)	Tput. (inference/s)
0	60K	0.09	10K
0	1.2M	1.89	528
0	903K	1.38	71
0	2.4K	0.003	26
0	848K	1.3	76
Sentimental 100:0	24K	0.037	27K
GoogLeNet (Us) 70:30	261K	0.40	2.4K
GoogLeNet (SuperTile [30]) -	-	3.3	3K

30% Lower Throughput

Performance Results





io nv:MM)	me s)	Tput. (inference/s)
0	09	10K
0	9	528
0	8	71
0	003	26
0	3	76
Sentimental 100:0	24K 0.037	27K
GoogLeNet (Us) 70:30	261K 0.40	2.4K
GoogLeNet (SuperTile [30]) -	- 3.3	3K

30% Lower Throughput

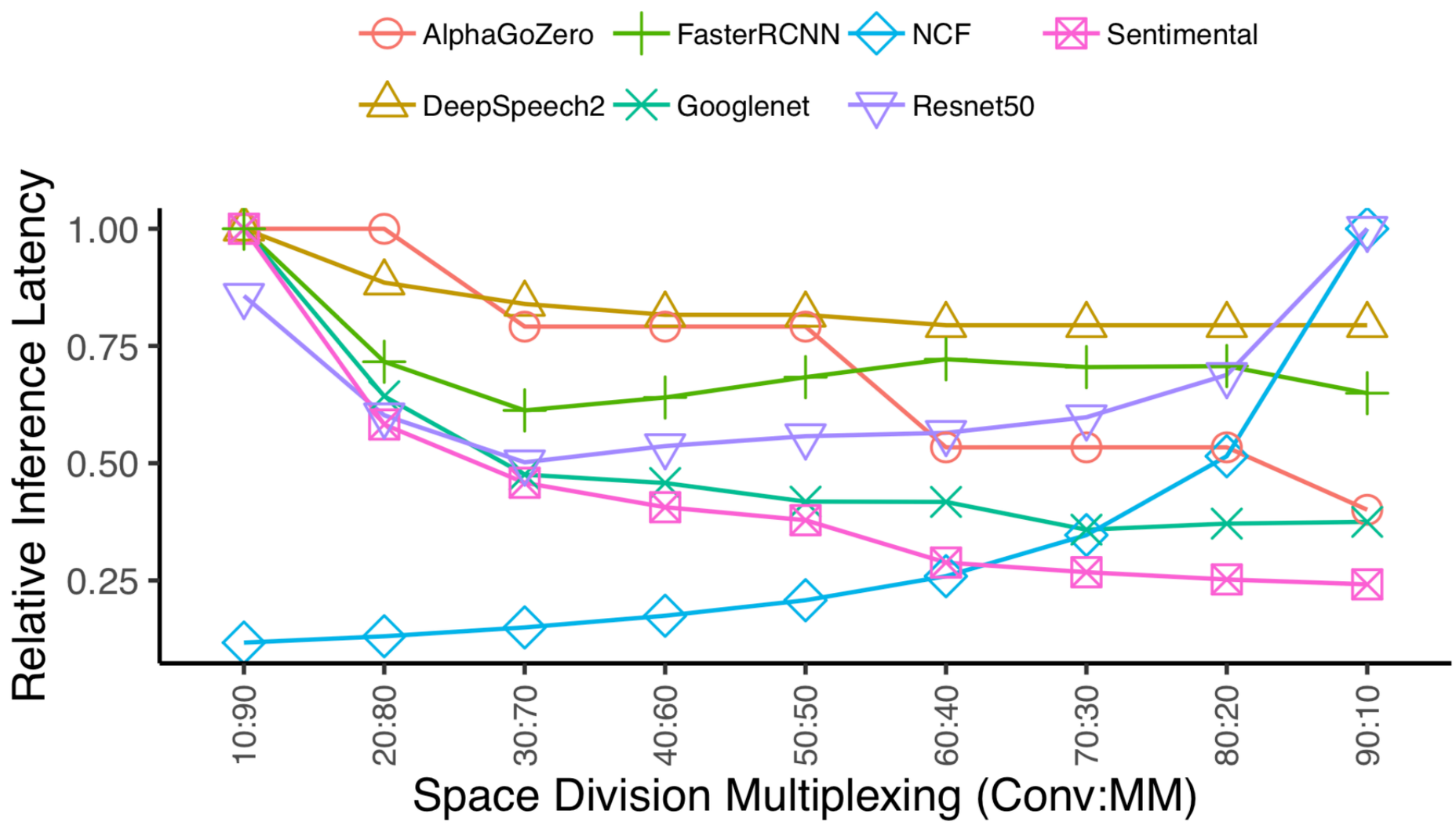
Performance Results

Topology (MLPerf)	**Ratio** (Conv:MM)	**Cycles**	**Time** (ms)	**Tput.** (inference/s)
AlphaGoZero	90:10	60K	0.09	10K
DeepSpeech2	60:40	1.2M	1.89	528
FasterRCNN	30:70	903K	1.38	719
NCF	0:100	2.4K	0.003	260K
Resnet50	30:70	848K	1.3	766
Sentimental	100:0	24K	0.037	27K
GoogLeNet (Us)	70:30	261K	0.40	2.4K
GoogLeNet (SuperTile [30])	-	-	3.3	3K

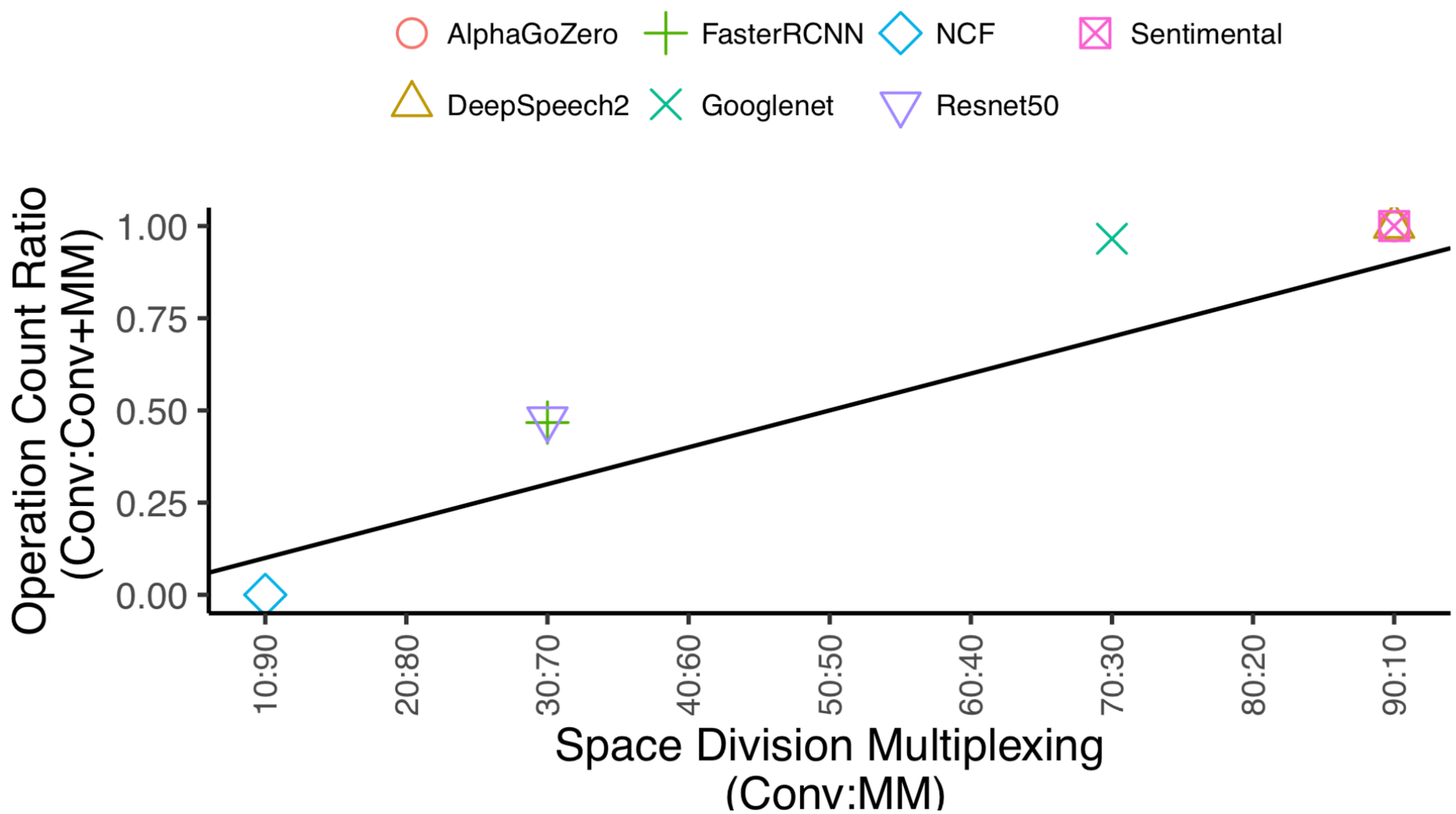
Performance Results

Topology (MLPerf)	**Ratio** (Conv:MM)	**Cycles**	**Time** (ms)	**Tput.** (inference/s)
AlphaGoZero	90:10	60K	0.09	10K
DeepSpeech2	60:40	1.2M	1.89	528
FasterRCNN	30:70	903K	1.38	719
NCF	0:100	2.4K	0.003	260K
Resnet50	30:70	848K	1.3	766
Sentimental	100:0	24K	0.037	27K
GoogLeNet (Us)	70:30	261K	0.40	2.4K
GoogLeNet (SuperTile [30])	-	-	3.3	3K

Optimizing the mapping



Compute Characteristics



Conclusions

- 650+ MHz operation for FPGA ML accelerator tailored for Xilinx UltraScale+
- 7x better latency, 30% poorer throughput vs. Xilinx SuperTile
- Hard interconnect cascades save us 30% registers + 12% on clock period vs. Fabric interconnect

Discussion

- **URAM Bandwidth balance** — Matrix-Multiplication performance suffers due to missing bandwidth from URAM —> *Give us 144b ports vs 72b ports!*

- **Dynamic Control** — Can build unified Conv + MM blocks if data flow in cascades even more programmble —> *Give us more control!*

- **Abandon Versal** — Versal architecture is not an FPGA. Improve DSPs, BRAMs, URAMs + hard interconnect instead—> *Stay true to your roots, Xilinx!*

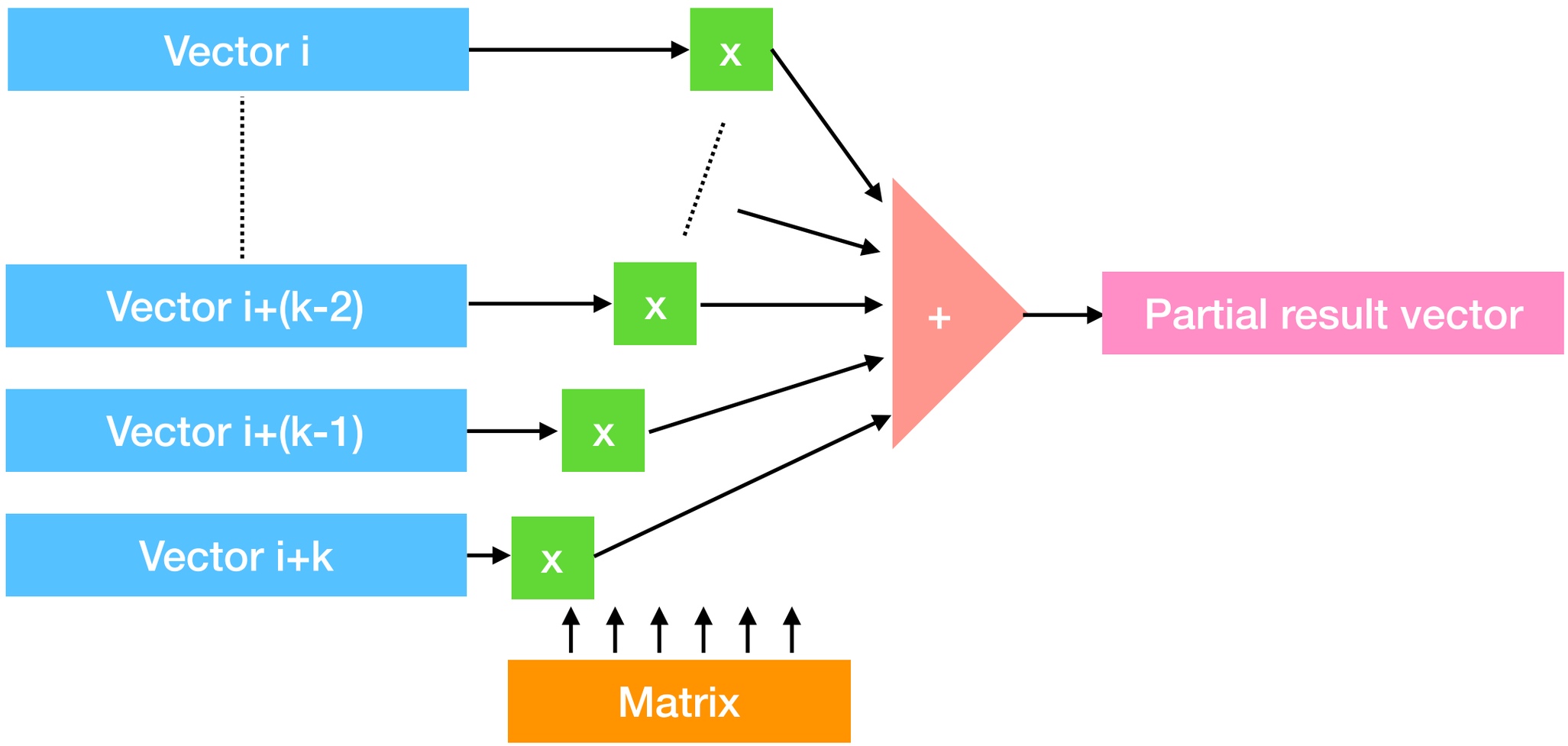
Discussion

- **URAM Bandwidth balance** — Matrix-Multiplication performance suffers due to missing bandwidth from URAM —> *Give us 144b ports vs 72b ports!*

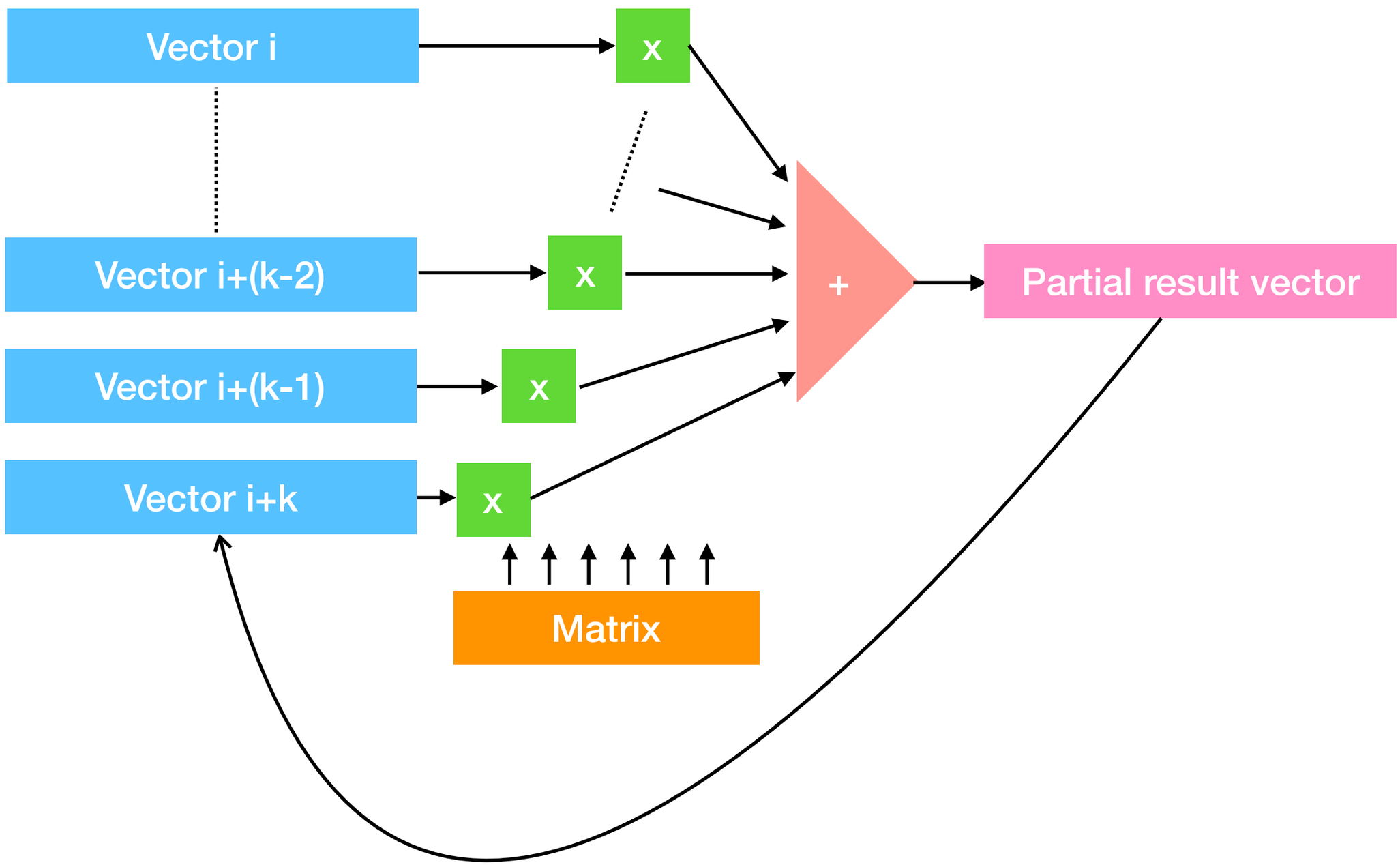
- **Dynamic Control** — Can build unified Conv + MM blocks if data flow in cascades even more programmble —> *Give us more control!*

- **Abandon Versal** — Versal architecture is not an FPGA. Improve DSPs, BRAMs, URAMs + hard interconnect instead—> *Stay true to your roots, Xilinx!*

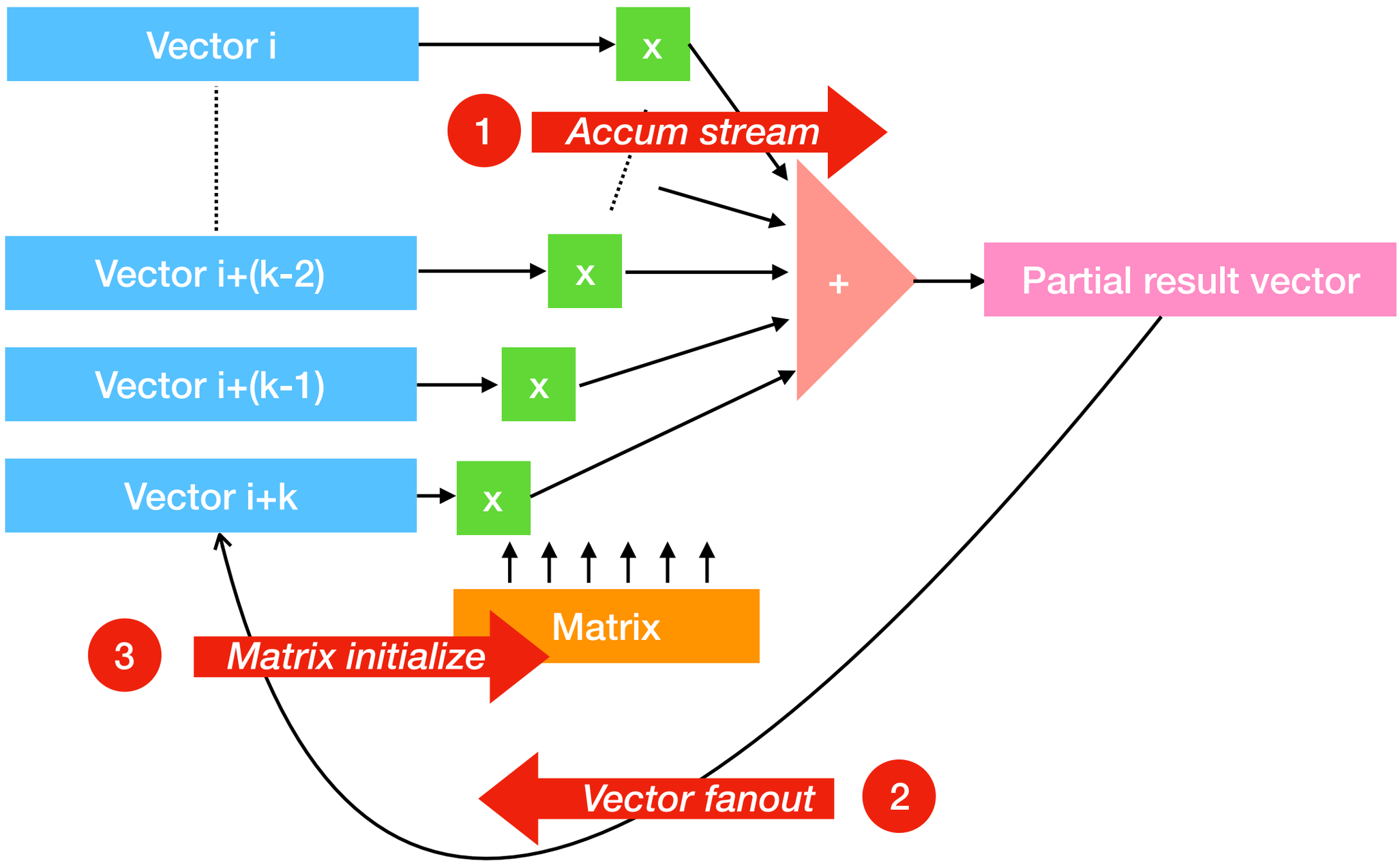
Communication Requirements of Matrix Multiplication



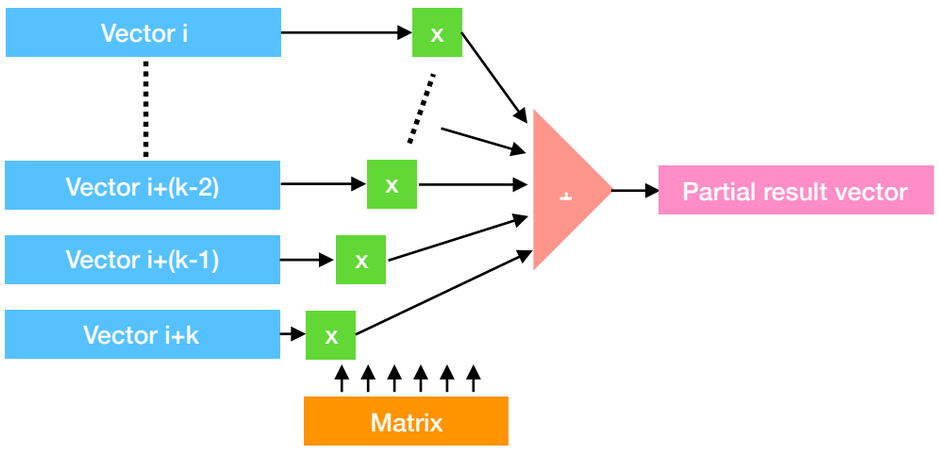
Communication Requirements of Matrix Multiplication

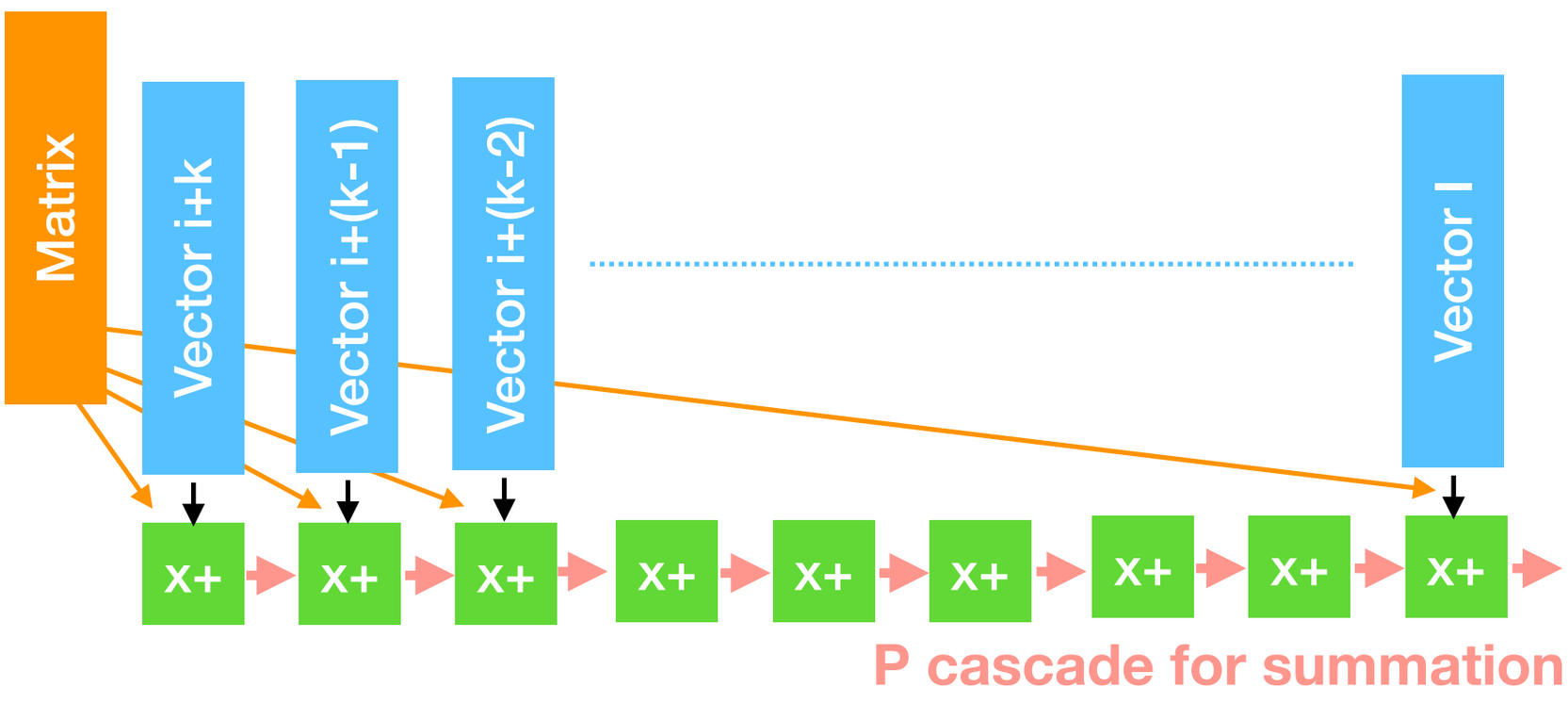


Communication Requirements of Matrix Multiplication

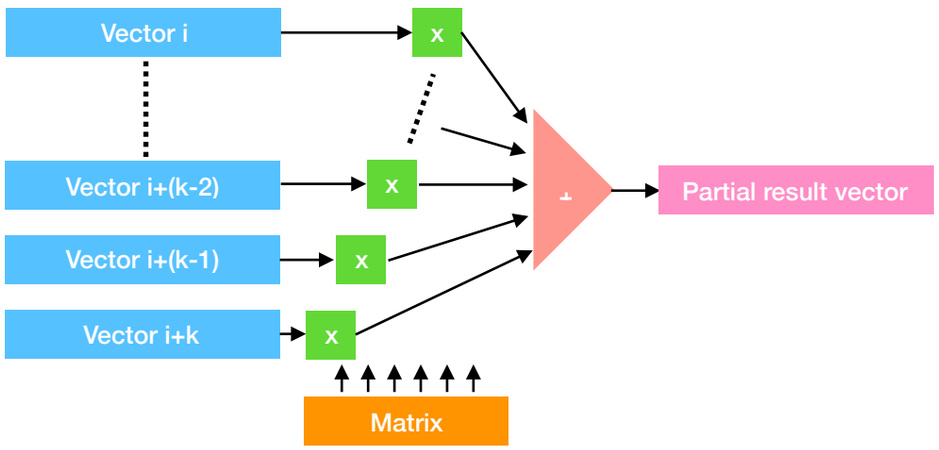


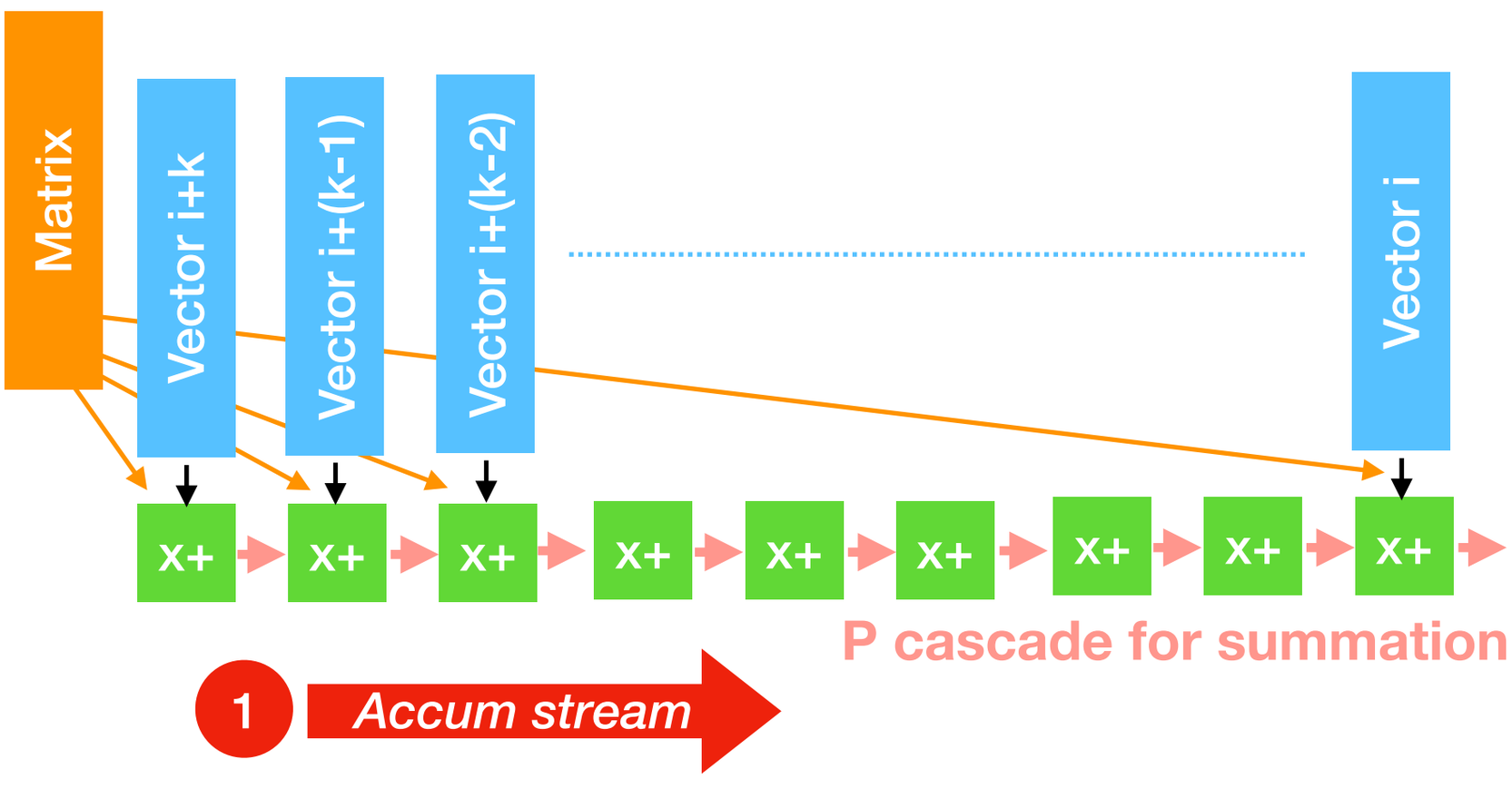
Communication Pattern



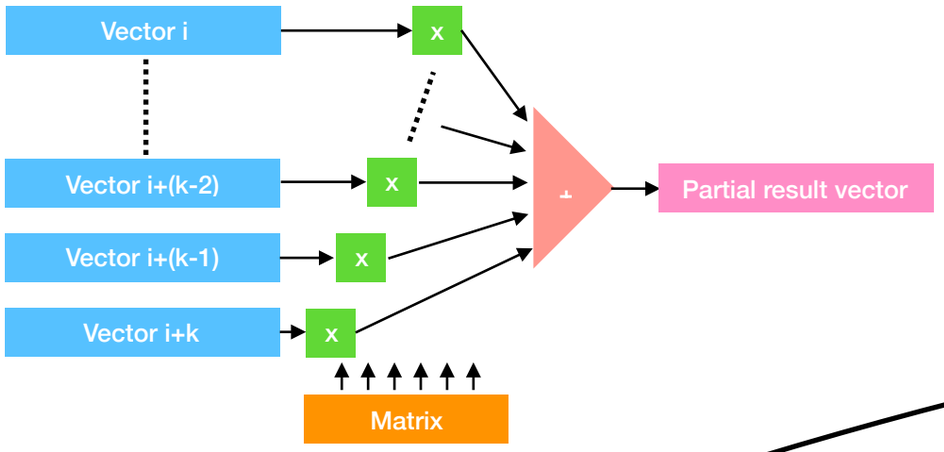


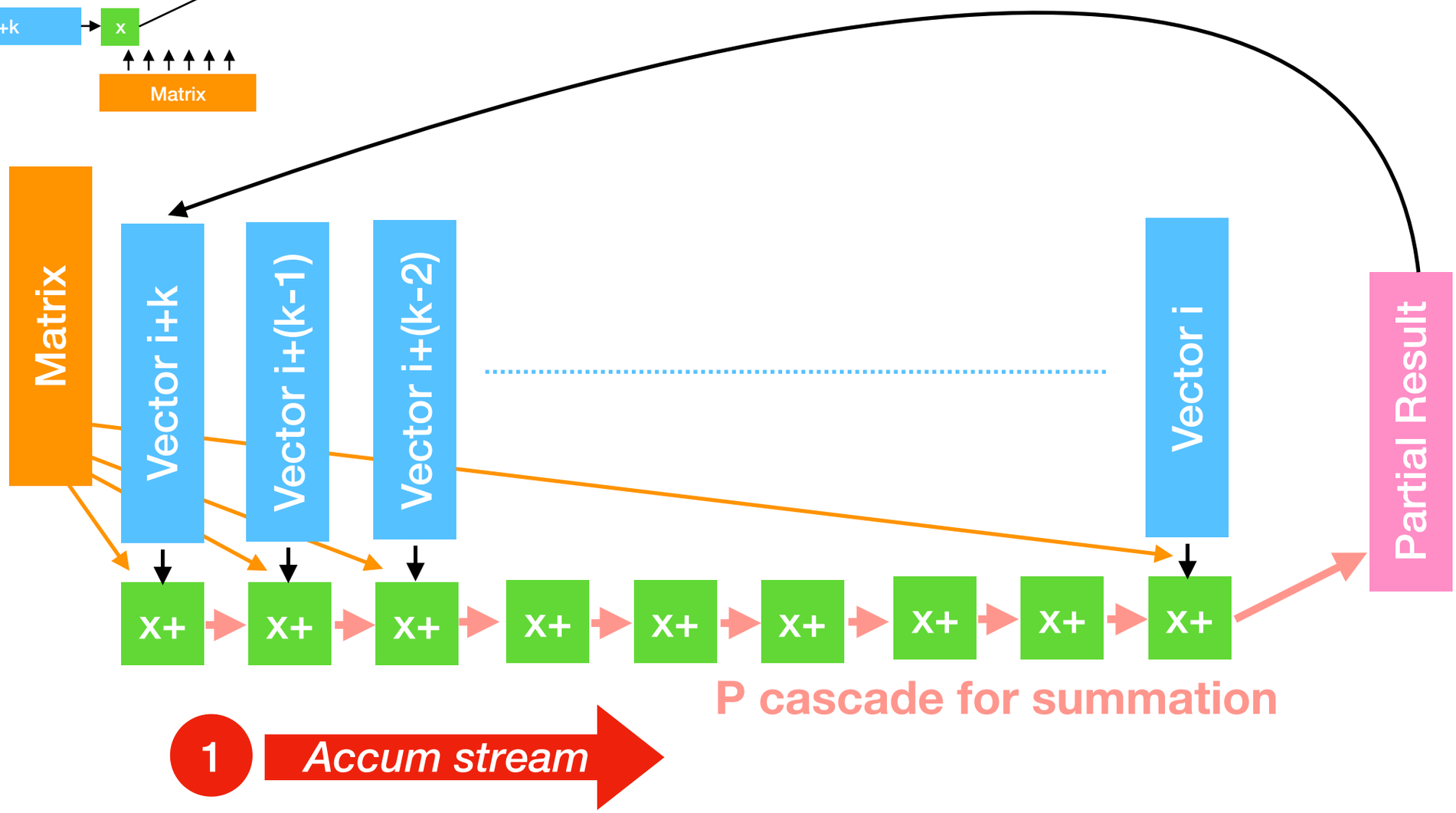
Communication Pattern



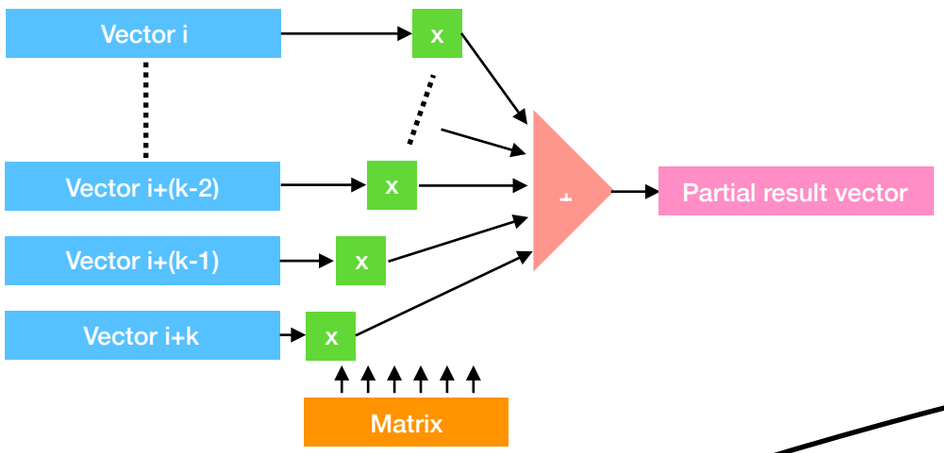


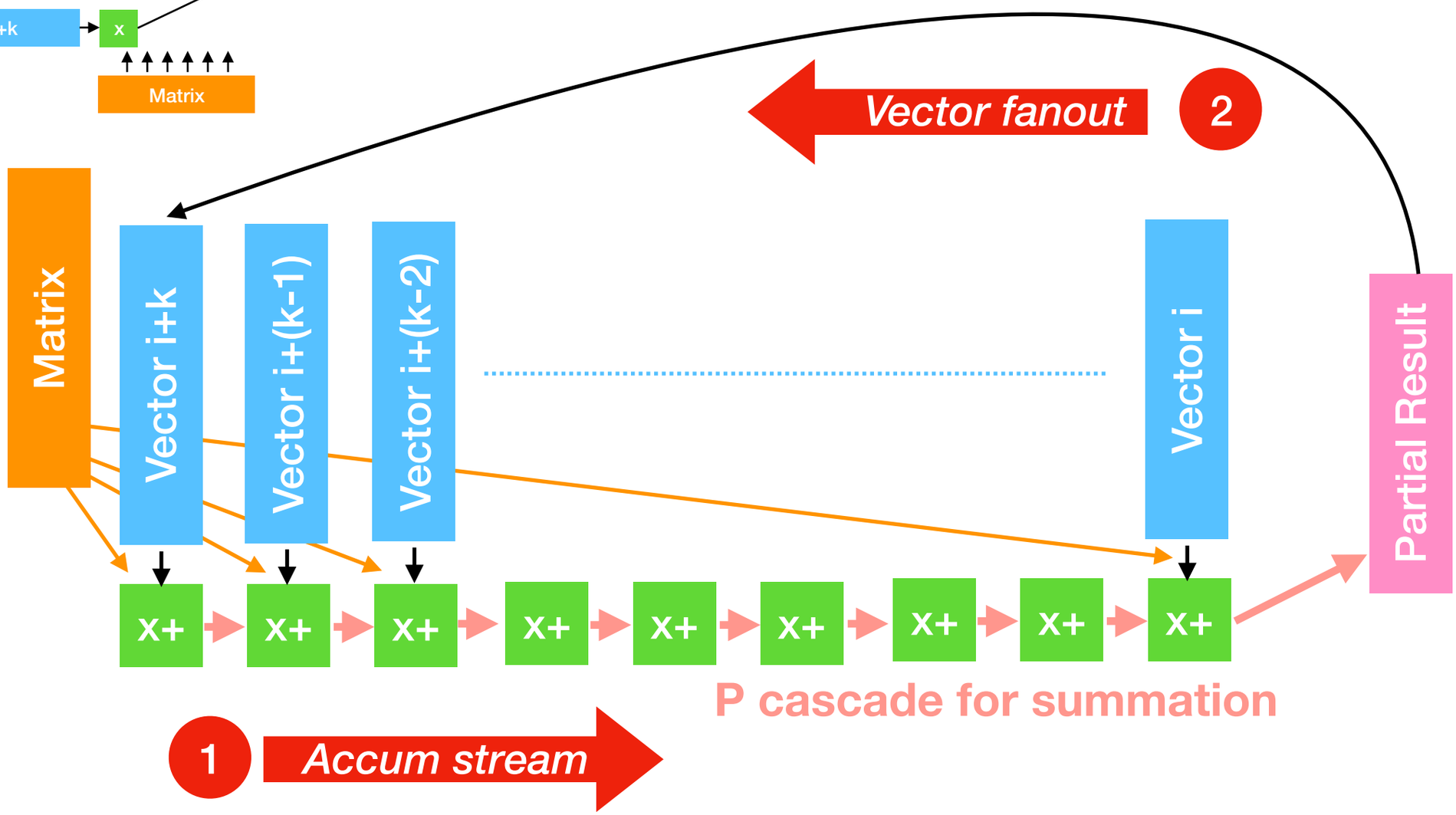
Communication Pattern



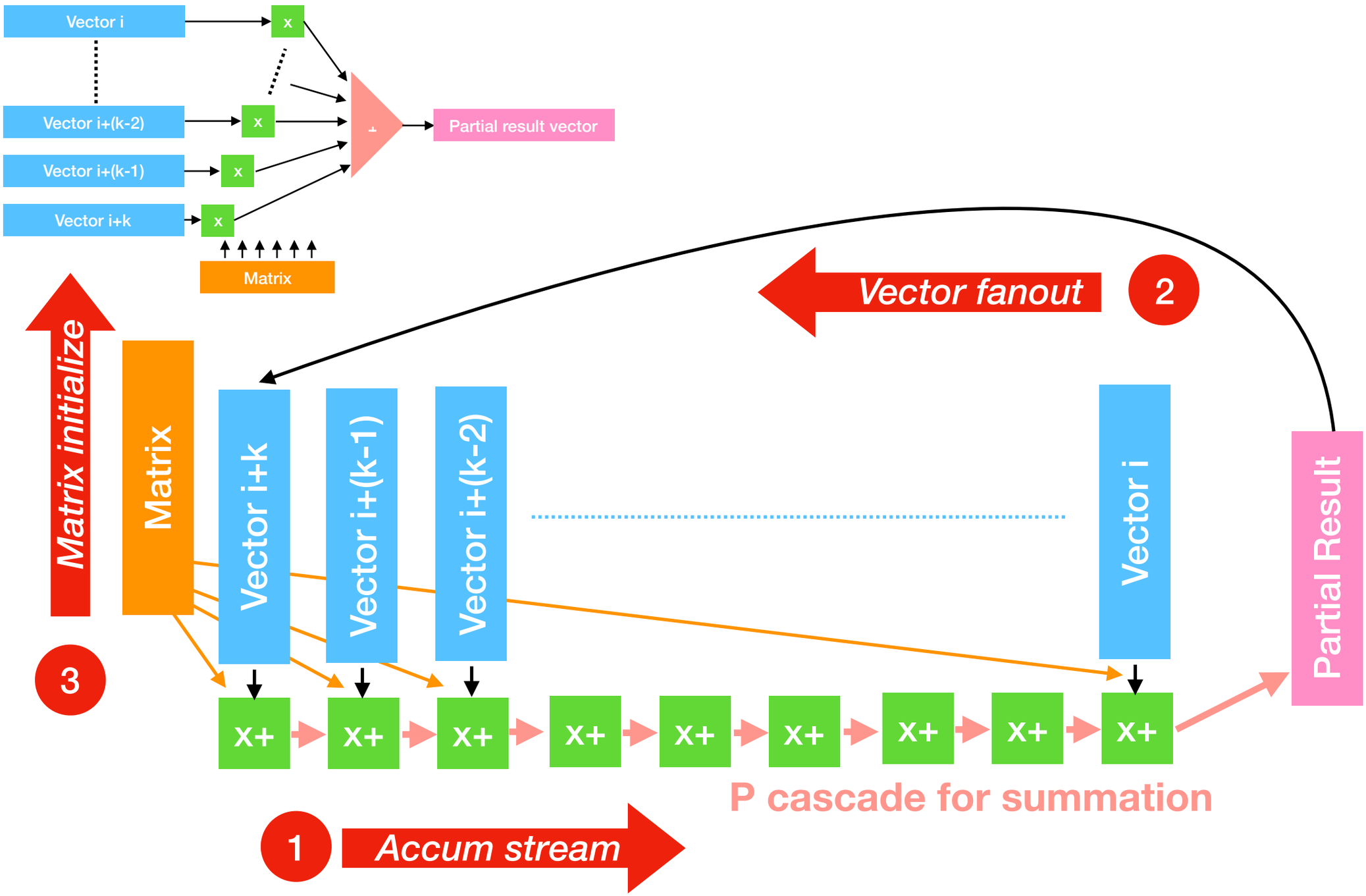


Communication Pattern

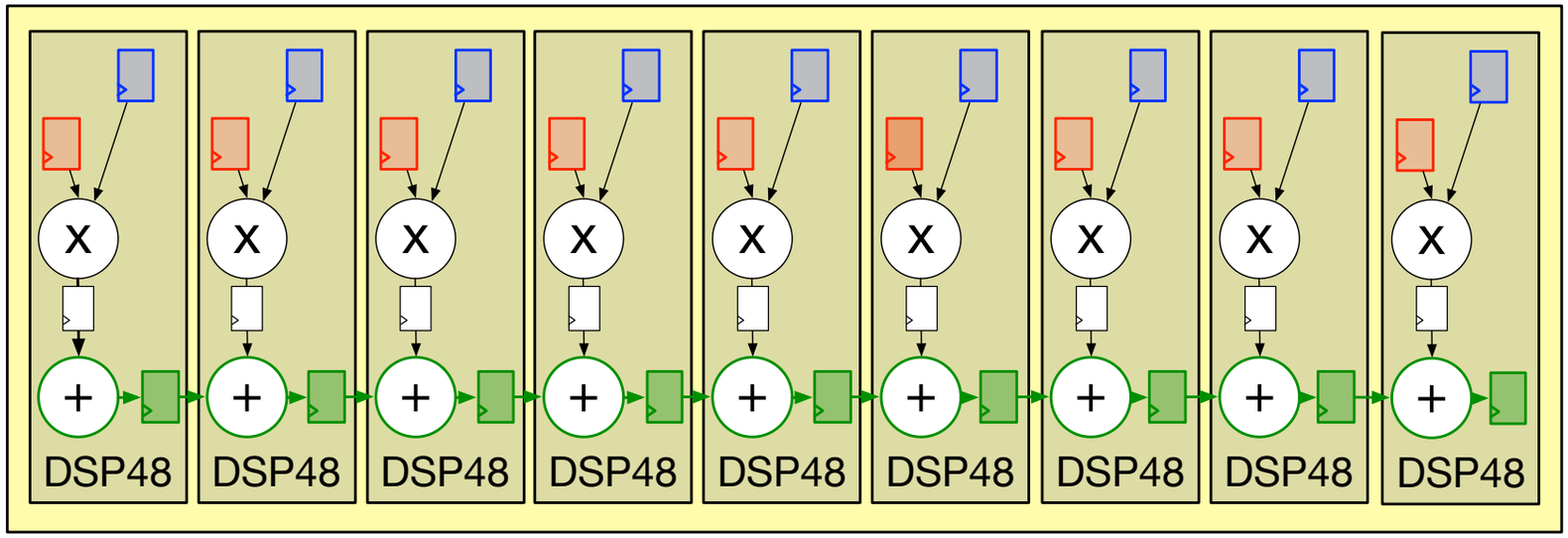




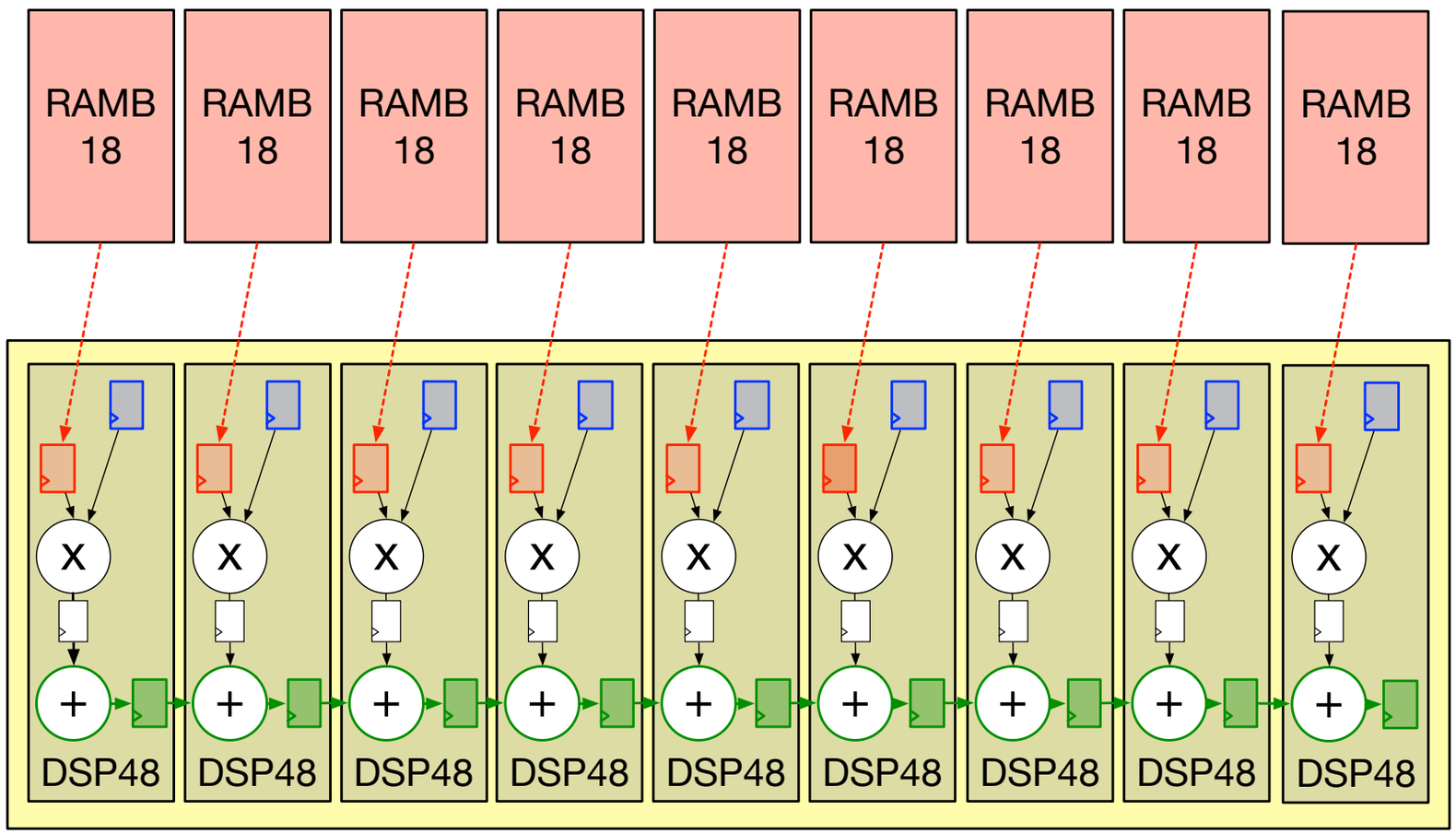
Communication Pattern



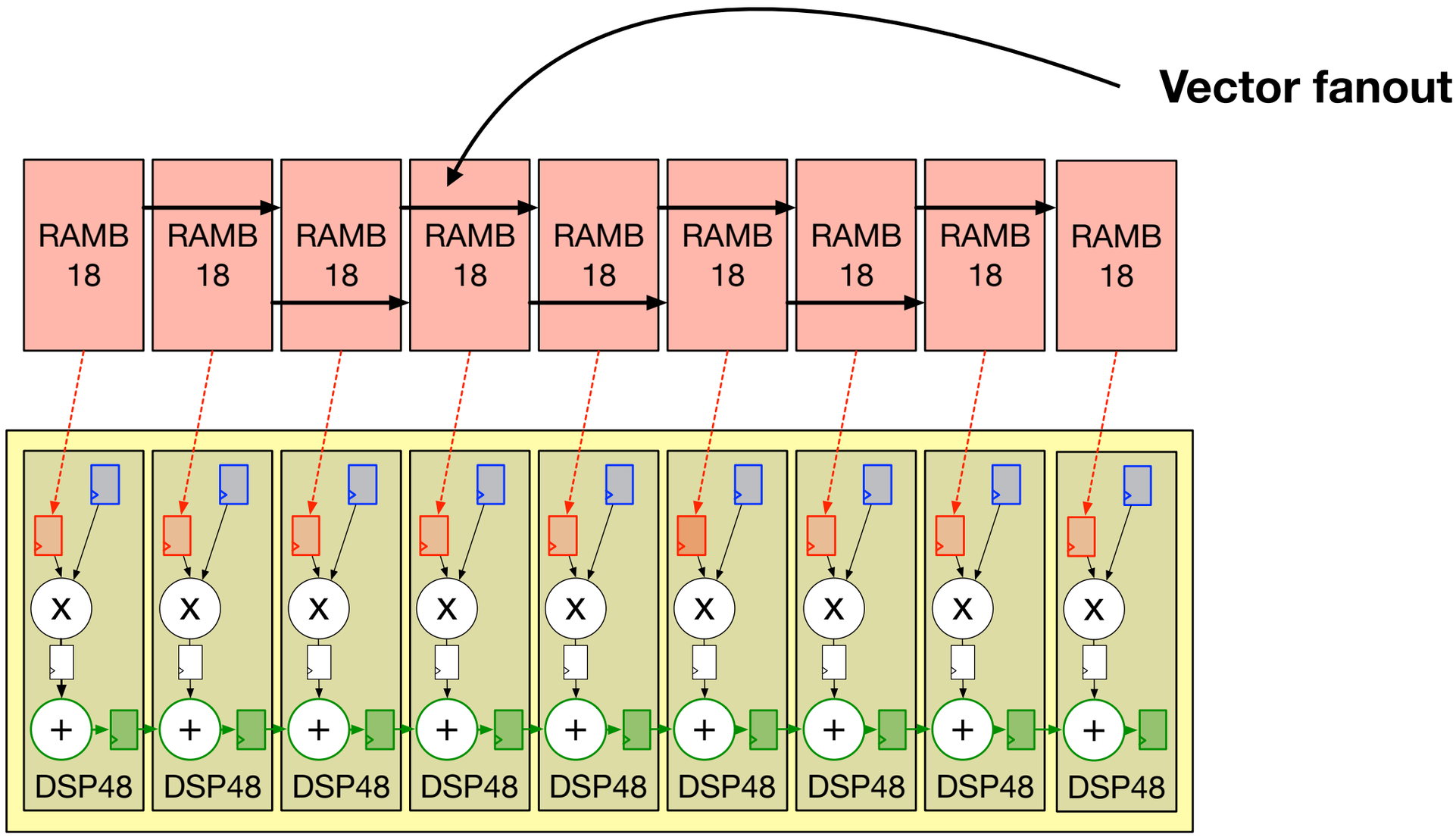
Matrix-Matrix Multiplication



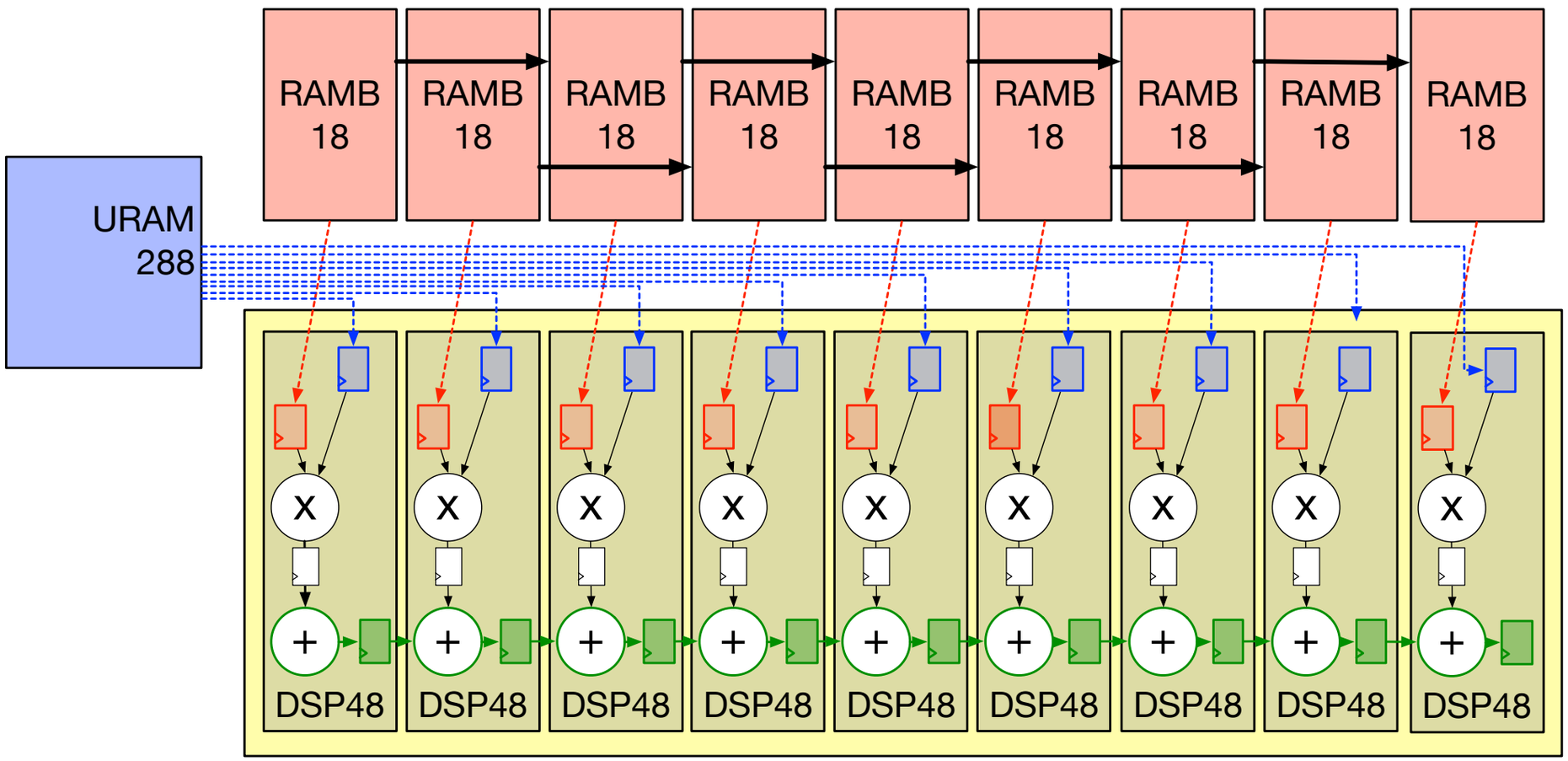
Matrix-Matrix Multiplication



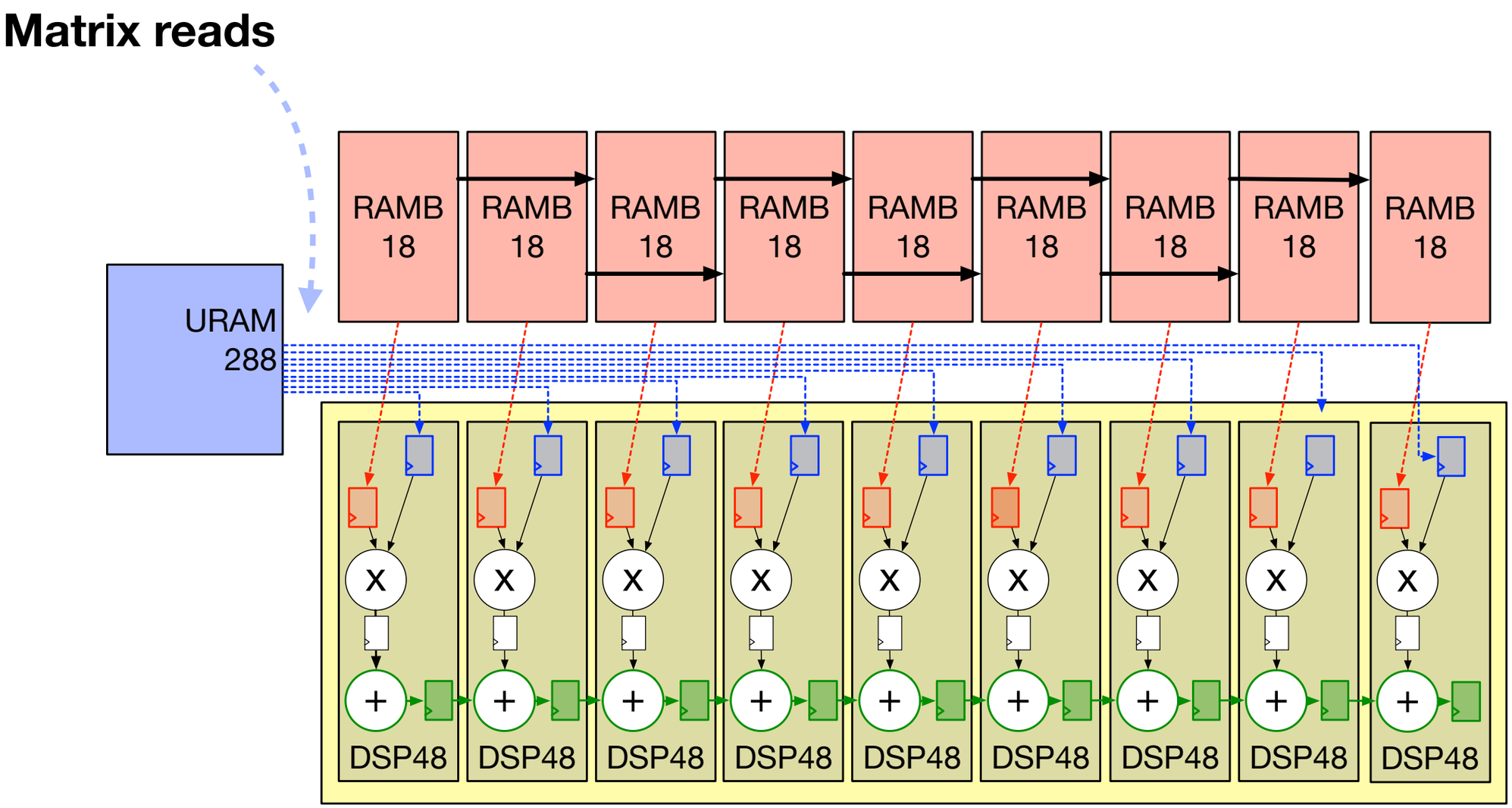
Matrix-Matrix Multiplication



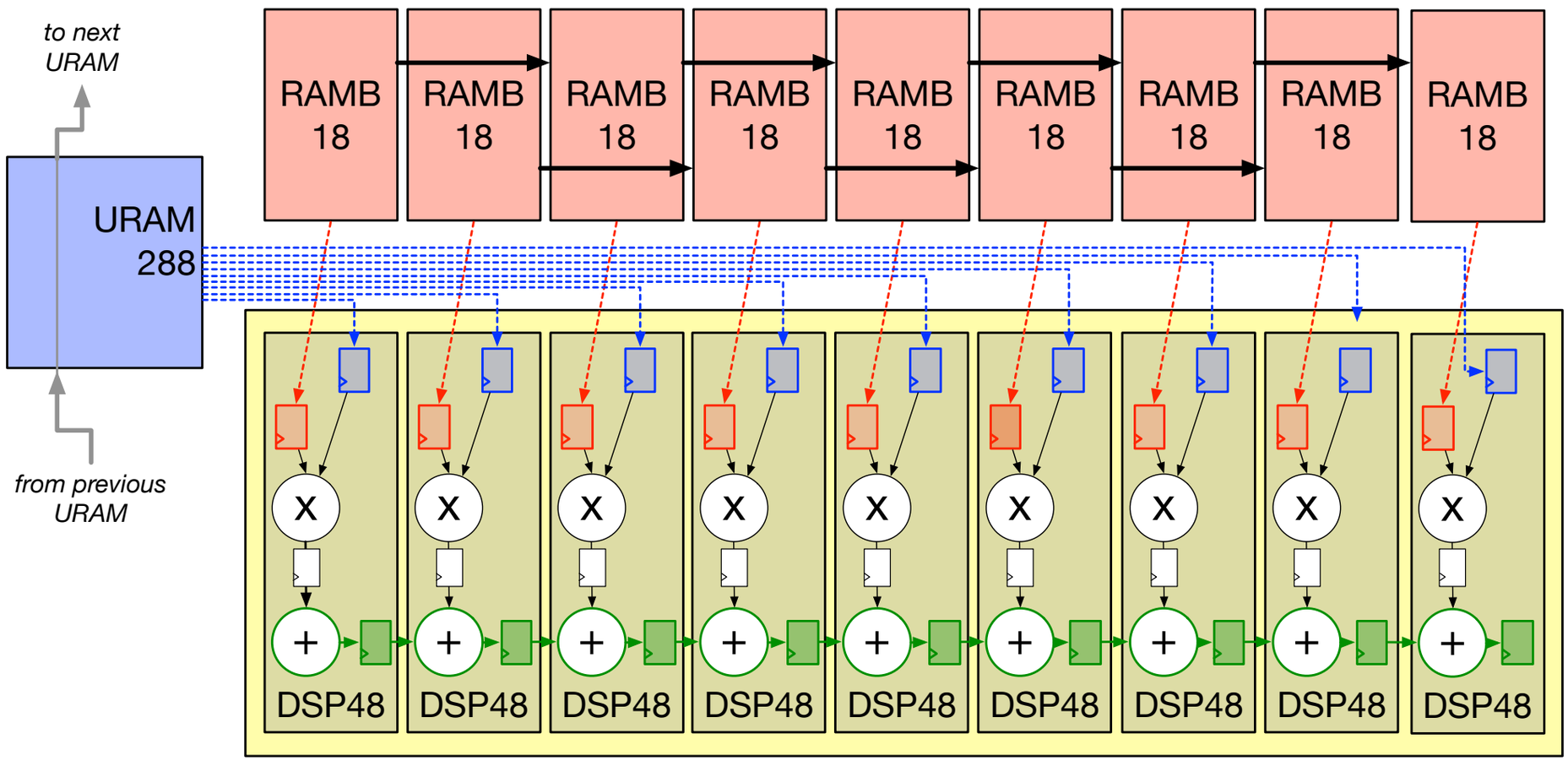
Matrix-Matrix Multiplication



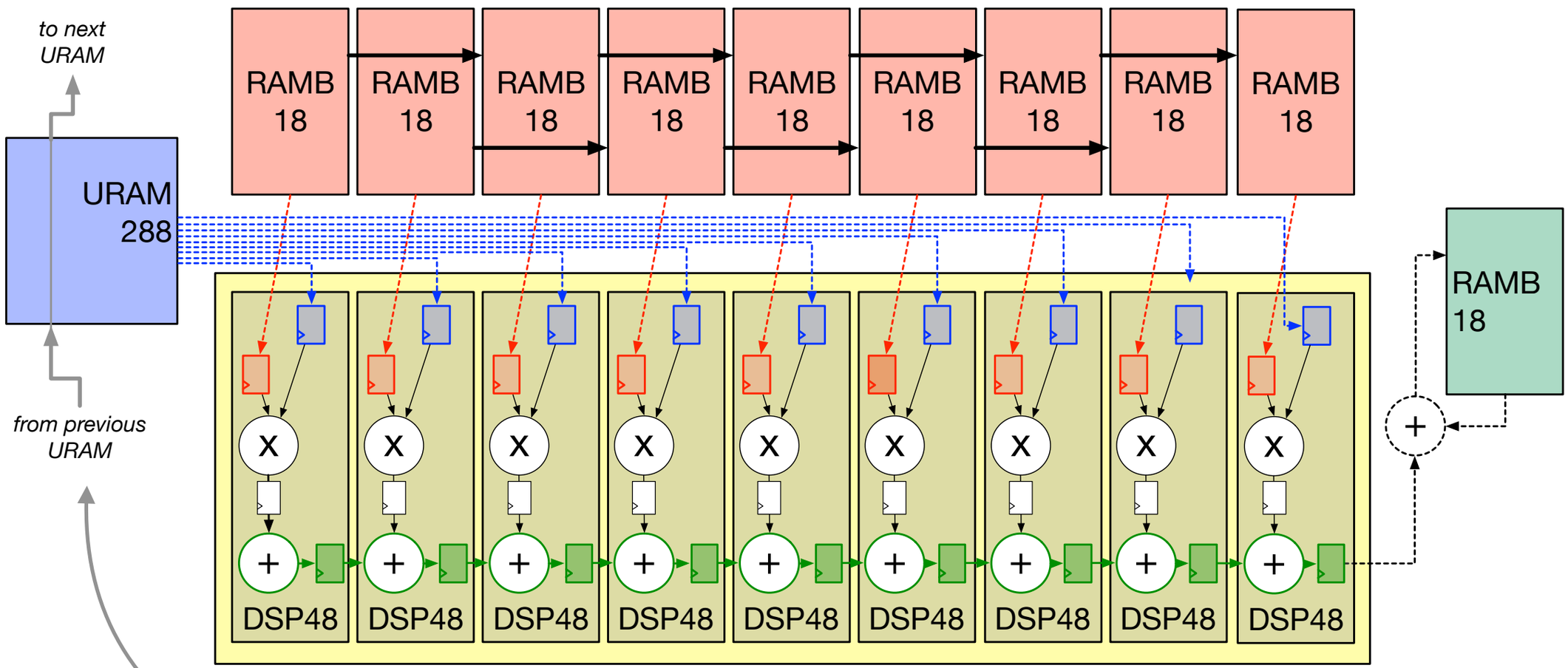
Matrix-Matrix Multiplication



Matrix-Matrix Multiplication



Matrix-Matrix Multiplication



Initial loading

MLPerf Benchmarks

- *Caveat*: **Result not verified by MLPerf.** MLPerf name and logo are trademarks. See www.mlperf.org for more information.
- Different ML workloads
- Various domains
- Different compute complexity
- SoftMax not implemented in hardware

TABLE I: MLPerf and GoogLeNet benchmark characteristics.

Topology	**Operation Count**			**Storage** (bytes)	
(MLPerf)	All	Conv	MM	$\sum$Wts.	Activ.
AlphaGoZero	352M	352M	353K	1.5M	92K
DeepSpeech2	1.7G	1.7G	74K	355K	6.5M
FasterRCNN	3.5G	1.6G	1.8G	13M	802K
NCF	11M	0	11M	11M	138K
Resnet50	3.4G	1.6G	1.8G	25M	802K
Sentimental	210M	0	210M	172K	30.7M
~~Transformer~~	~~113M~~	~~35M~~	~~78M~~	~~77M~~	~~4096~~
GoogLeNet	1.3G	1.3G	46M	6.8M	200K

Why is SuperTile so good?

- Base Design
  - High-frequency layout using DSP cascades,
  - LUT RAMs for weights,
  - Systolic data movement in fabric
- **Throughput boost**
  - Decompose the systolic array into sub-arrays
  - Perform pipelining across CNN layers
  - Sacrifice some latency to significantly boost throughput!

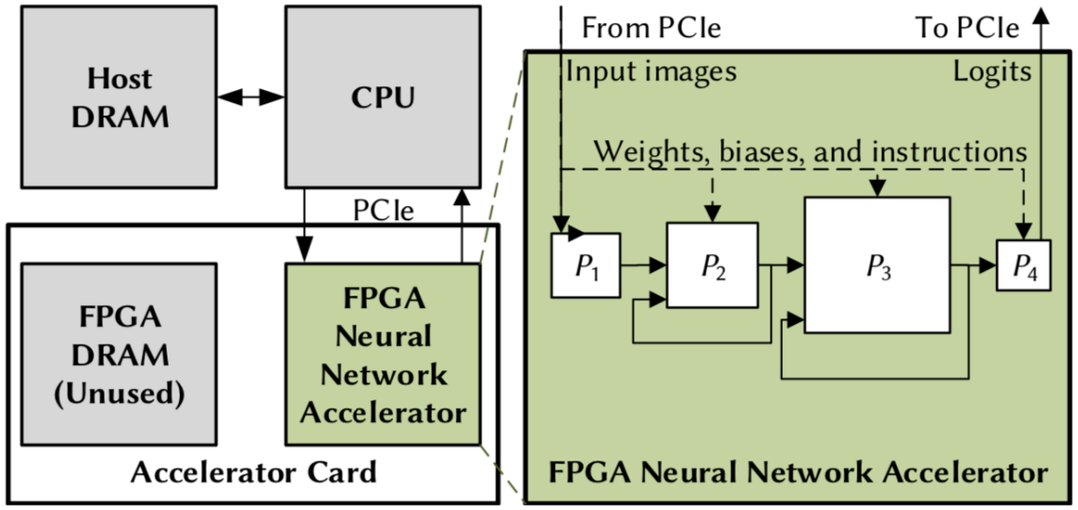


Figure 1: Neural-network accelerator. © Ephrem Wu.

Xilinx INT8 optimization

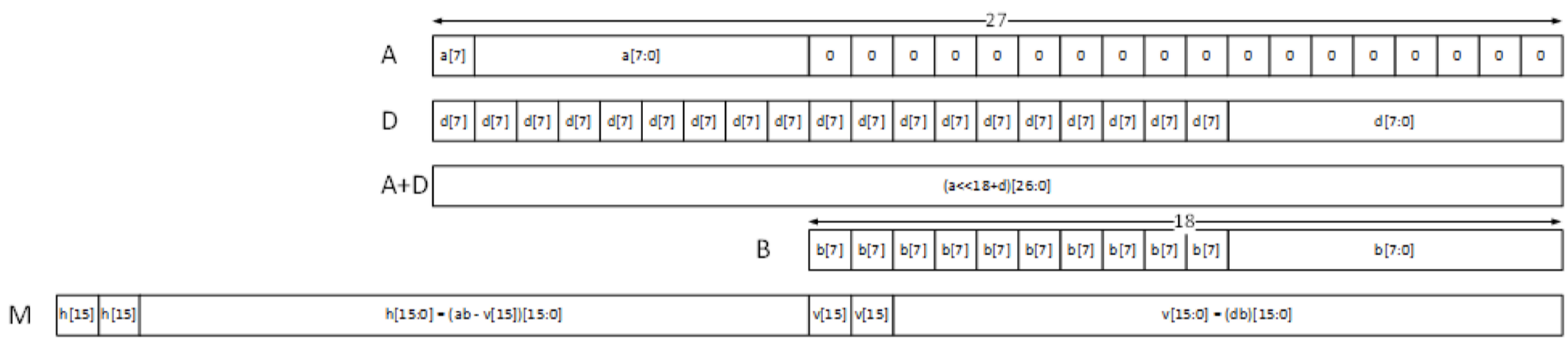


Figure 7: **Two Products in a Packed Word**

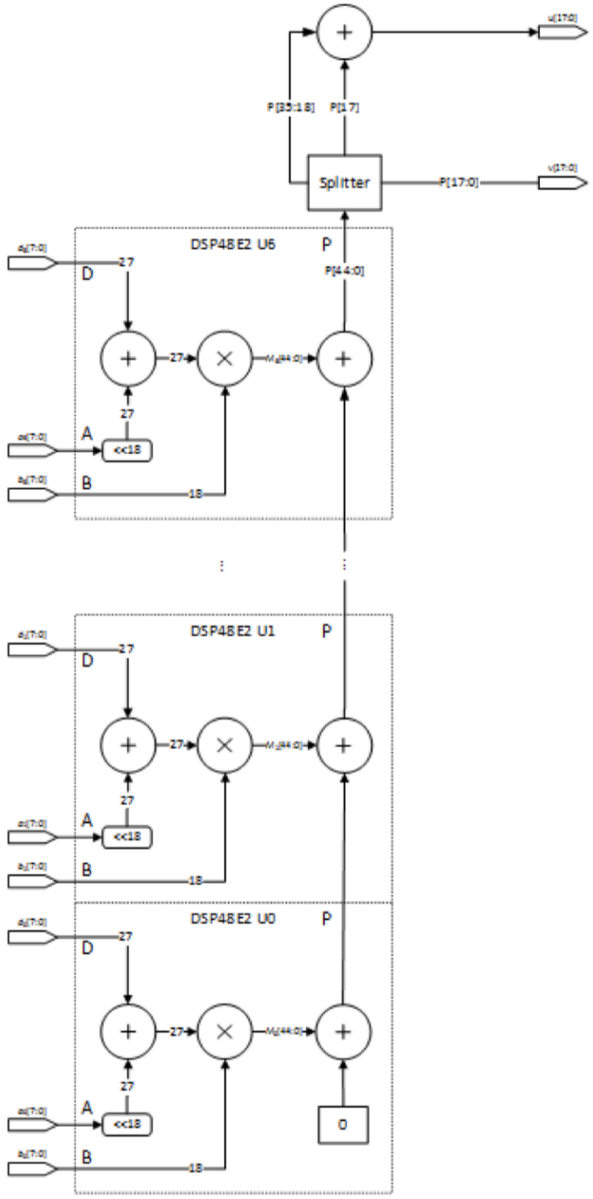
- Soft-fracture a DSP48 27x18 multiplier to compute two 8x8 multiplications with a common operand
- A*B and D*B computed in together when A, B, D are 8 inputs

https://www.xilinx.com/support/documentation/white_papers/wp486-deep-learning-int8.pdf

https://www.xilinx.com/support/documentation/white_papers/wp487-int8-acceleration.pdf

Xilinx INT8 optimization

- Create DSP cascades of length 7 to accumulate multiple product terms
- Enough precision headroom to avoid overflow for this length of the chain
- Some fabric operation needed for final accumulations, or if 3x3 convolution support required



https://www.xilinx.com/support/documentation/white_papers/wp486-deep-learning-int8.pdf

https://www.xilinx.com/support/documentation/white_papers/wp487-int8-acceleration.pdf

Xilinx DSP48 Systolic Mode

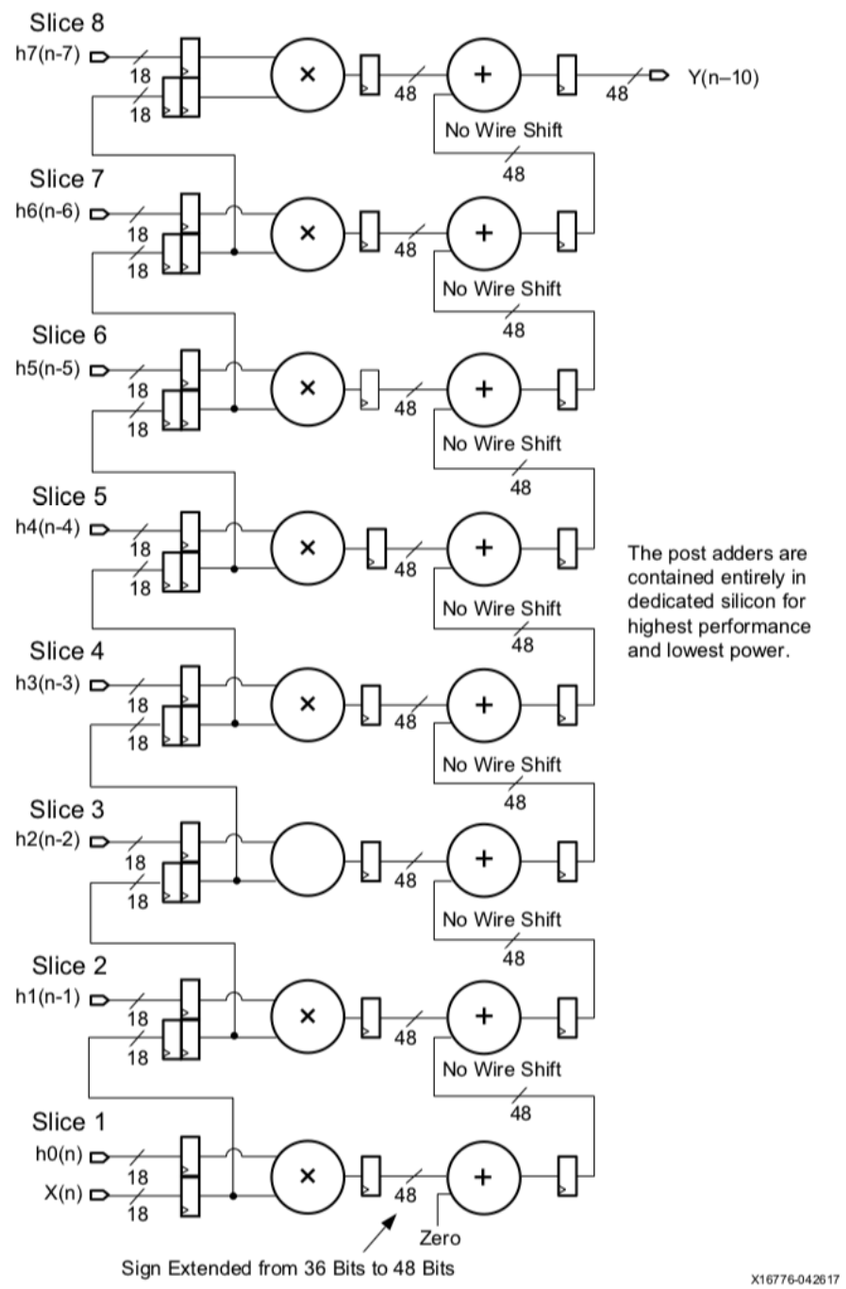


Figure 4-5: **Adder Cascade**

https://www.xilinx.com/support/documentation/user_guides/ug479_7Series_DSP48E1.pdf