Hybrid Dot-Product Design for FP-Enabled FPGAs

Bogdan Pasca

Intel

ARITH 2019, June 10-12, 2019

Hybrid Dot-Product Design for FP-Enabled FPGAs

Bogdan Pasca

Intel

ARITH 2019, June 10-12, 2019

Context

- FPGAs intersting **neural network training** accelerators
- training: mostly **dot-products** in forward+backward propagation
- current industry-standard: bfloat16 multiplications, SP reduction
- bfloat16 vs SP: 2X bandwidth

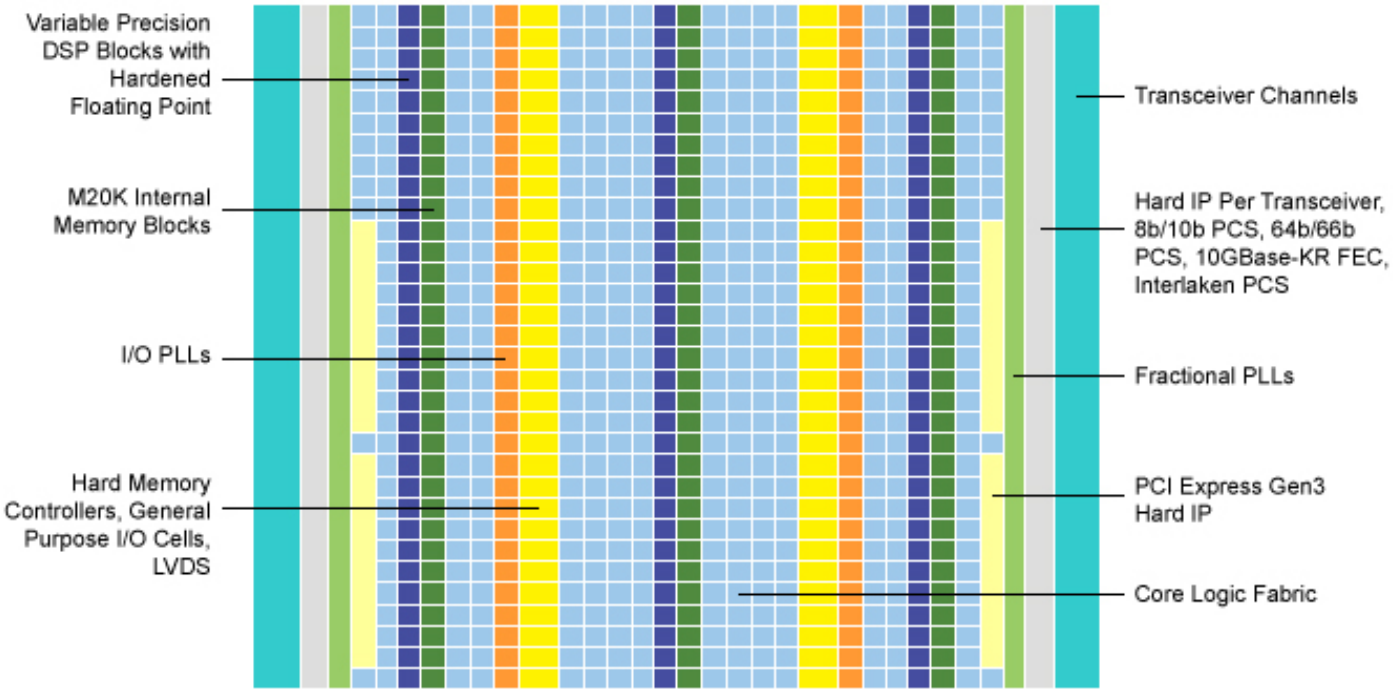
Context

- FPGAs intersting **neural network training** accelerators
- training: mostly **dot-products** in forward+backward propagation
- current industry-standard: bfloat16 multiplications, SP reduction
- bfloat16 vs SP: 2X bandwidth

Goal $\rightarrow$ Find a dot-product implementation that:

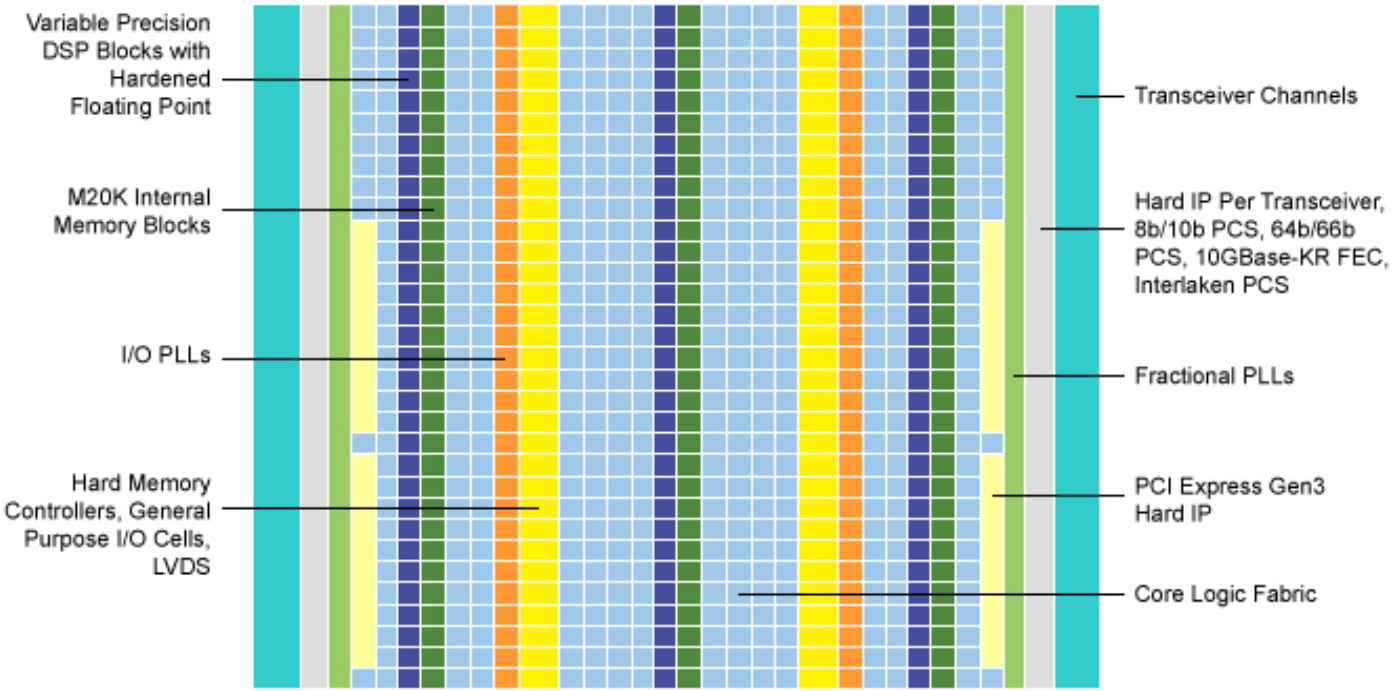
- maintains an accuracy comparable to bfloat16+SP
- maximizes the dot-product density for a given FPGA

Density



- FPGA devices have a various mix of resources
- increasing compute density $\rightarrow$ make efficient use of existing mix

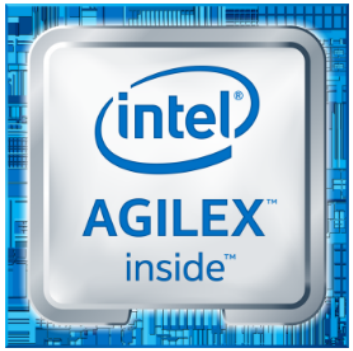
Density



- FPGA devices have a various mix of resources
- increasing compute density $\rightarrow$ make efficient use of existing mix

Focus on "Core Logic Fabric" and VP DSP Blocks

Density - some current devices



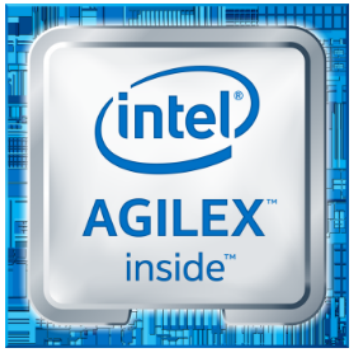
Intel® Agilex™ FPGAs

Intel® Stratix® Series

Intel® Arria® Series

Intel® Cyclone® Series

Density - some current devices



Intel® Agilex™ FPGAs

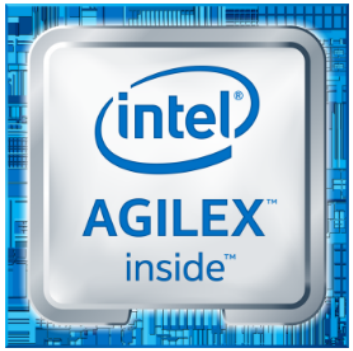
Intel® Stratix® Series

Intel® Arria® Series

Intel® Cyclone® Series

PRODUCT LINE		GX 160 SX 160	GX 220 SX 220	GX 270 SX 270	GX 320 SX 320	GX 480 SX 480	GX SX
Resources	LEs (K)	160	220	270	320	480	5
	System logic elements (K)	210	288	354	419	629	7
	Adaptive logic modules (ALMs)	61,510	83,730	101,620	118,730	181,790	217
	Registers	246,040	334,920	406,480	474,920	727,160	868
	M20K memory blocks	440	588	750	891	1,438	1,
	M20K memory (Mb)	9	11	15	17	28	
	MLAB memory (Mb)	1.0	1.8	2.4	2.8	4.3	
	Hardened single-precision floating-point multipliers/adders	156/156	191/191	830/830	985/985	1,368/1,368	1,523
	18 x 19 multipliers	312	382	1,660	1,970	2,736	3,
	Peak fixed-point performance (GMACS)[1]	343	420	1,826	2,167	3,010	3,
	Peak floating-point performance (GFLOPS)	140	172	747	887	1,231	1,

Density - some current devices



Intel® Agilex™ FPGAs

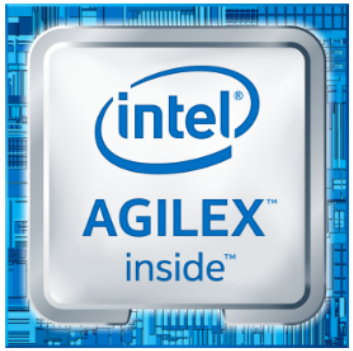
Intel® Stratix® Series

Intel® Arria® Series

Intel® Cyclone® Series

PRODUCT LINE		GX 160 SX 160	GX 220 SX 220	GX 270 SX 270	GX 320 SX 320	GX 480 SX 480	GX SX
Resources	LEs (K)	160	220	270	320	480	5
	System logic elements (K)	210	288	354	419	629	7
	Adaptive logic modules (ALMs)	61,510	83,730	101,620	118,730	181,790	217
	Registers	246,040	334,920	406,480	474,920	727,160	868
	M20K memory blocks	440	588	750	891	1,438	1,
	M20K memory (Mb)	9	11	15	17	28	
	MLAB memory (Mb)	1.0	1.8	2.4	2.8	4.3	
	Hardened single-precision floating-point multipliers/adders	156/156	191/191	830/830	985/985	1,368/1,368	1,523
	18 x 19 multipliers	312	382	1,660	1,970	2,736	3,
	Peak fixed-point performance (GMACS)[1]	343	420	1,826	2,167	3,010	3,
	Peak floating-point performance (GFLOPS)	140	172	747	887	1,231	1,

Density - some current devices



Intel® Agilex™ FPGAs

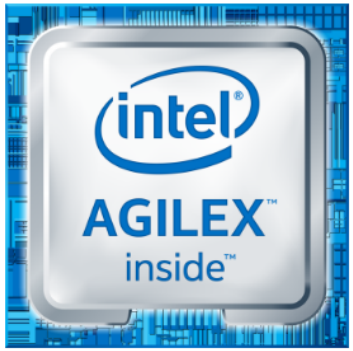
Intel® Stratix® Series

Intel® Arria® Series

Intel® Cyclone® Series

PRODUCT LINE		GX 160 SX 160	GX 220 SX 220	GX 270 SX 270	GX 320 SX 320	GX 480 SX 480	GX SX
Resources	LEs (K)	160	220	270	320	480	5
	System logic elements (K)	210	288	354	419	629	7
	Adaptive logic modules (ALMs)	61,510	83,730	101,620	118,730	181,790	217
	Registers	246,040	334,920	406,480	474,920	727,160	868
	M20K memory blocks	440	588	750	891	1,438	1,
	M20K memory (Mb)	9	11	15	17	28	
	MLAB memory (Mb)	1.0	1.8	2.4	2.8	4.3	
	Hardened single-precision floating-point multipliers/adders	156/156	191/191	830/830	985/985	1,368/1,368	1,523
	18 x 19 multipliers	312	382	1,660	1,970	2,736	3,
	Peak fixed-point performance (GMACS)[1] Peak floating-point performance (GFLOPS)	394	459	122	120	132	1

Density - some current devices



Intel® Agilex™ FPGAs

Intel® Stratix® Series

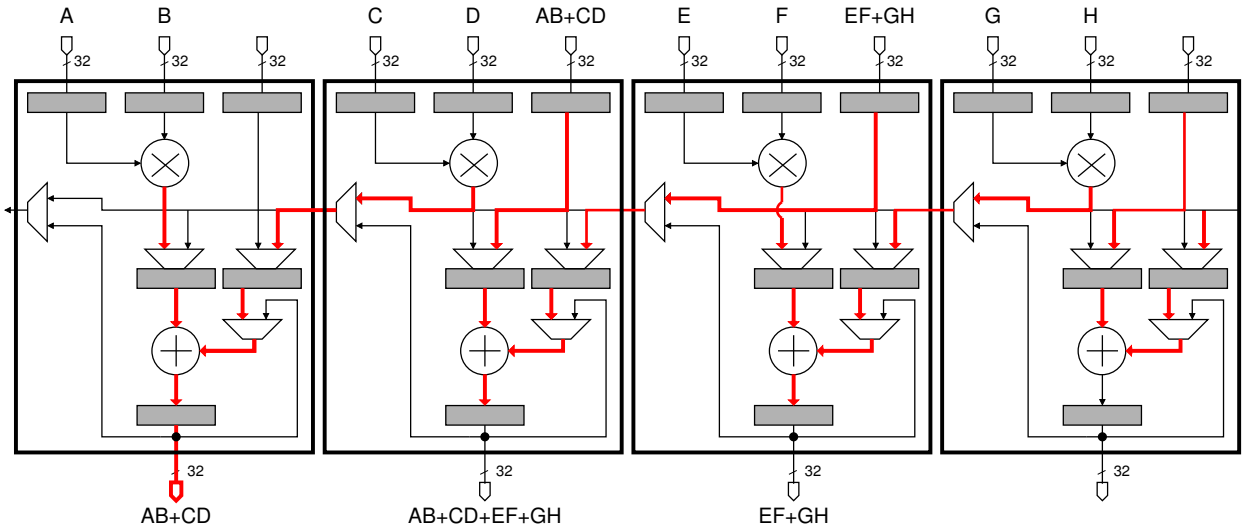
Intel® Arria® Series

Intel® Cyclone® Series

Hardened single-precision floating-point multiplers/adders	156/156
18 x 19 multipliers	312

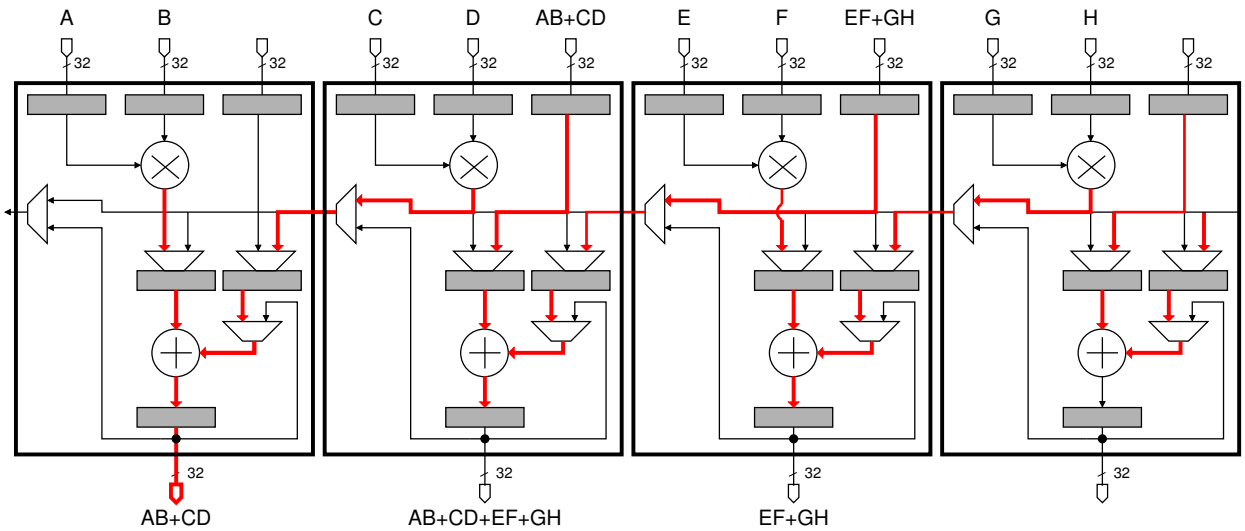
Background

- Intel FPGAs: DSP blocks implement SP mult-add
- N-element SP dot-product = N DSPs



Background

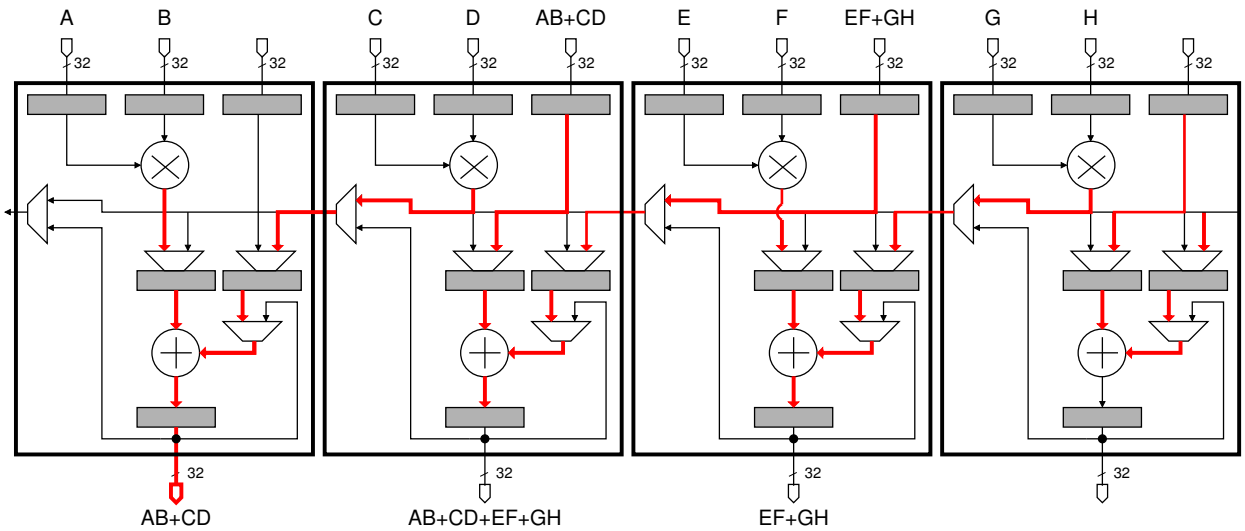
- Intel FPGAs: DSP blocks implement SP mult-add
- N-element SP dot-product = N DSPs



- soft-logic-only solution
  - bfloat16 multiplier $\rightarrow$ 2/DSP
  - SP FP adder 1 $\rightarrow$ 1/DSP
  - N-element dot product: $C_{DSP} = N/2 + N - 1$

Background

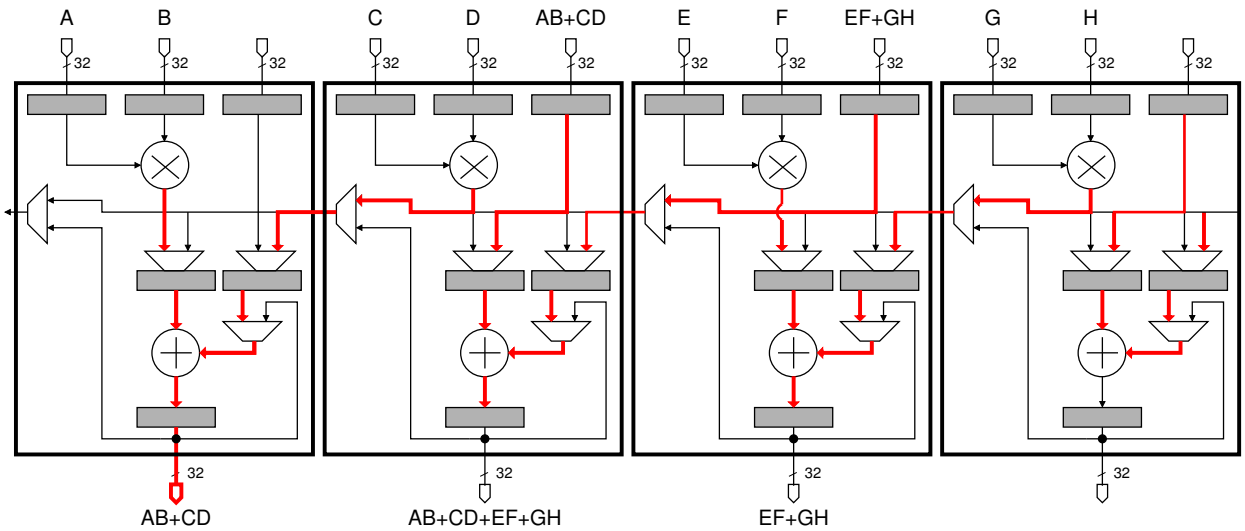
- Intel FPGAs: DSP blocks implement SP mult-add
- N-element SP dot-product = N DSPs



- soft-logic-only solution
  - bfloat16 multiplier $\rightarrow$ 2/DSP
  - SP FP adder 1 $\rightarrow$ 1/DSP
  - N-element dot product: $C_{DSP} = N/2 + N - 1$
  - adjust ratio: migrate SP FP adders to logic (300-400 ALMs/add)

Background

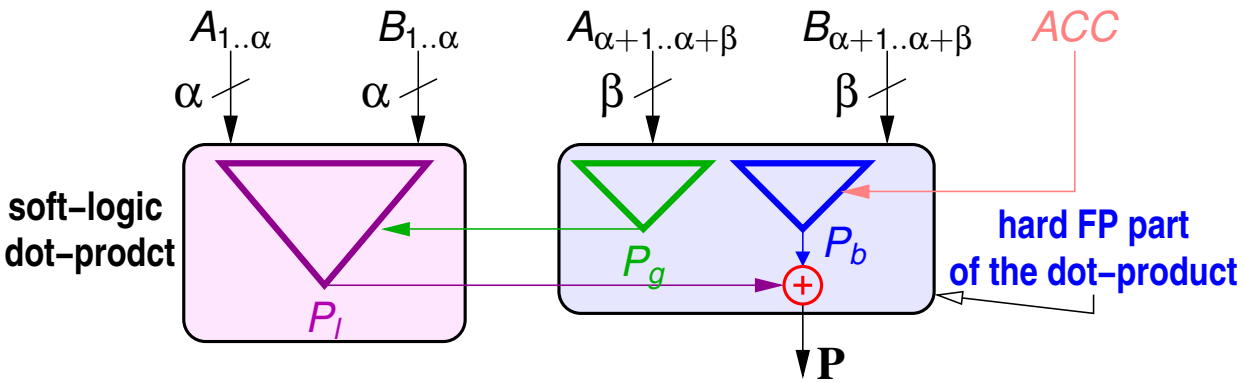
- Intel FPGAs: DSP blocks implement SP mult-add
- N-element SP dot-product = N DSPs



- soft-logic-only solution
  - bfloat16 multiplier $\rightarrow$ 2/DSP
  - SP FP adder 1 $\rightarrow$ 1/DSP
  - N-element dot product: $C_{DSP} = N/2 + N - 1$
  - adjust ratio: migrate SP FP adders to logic (300-400 ALMs/add)
  - solution is too large

How do we solve this?

Our implementation

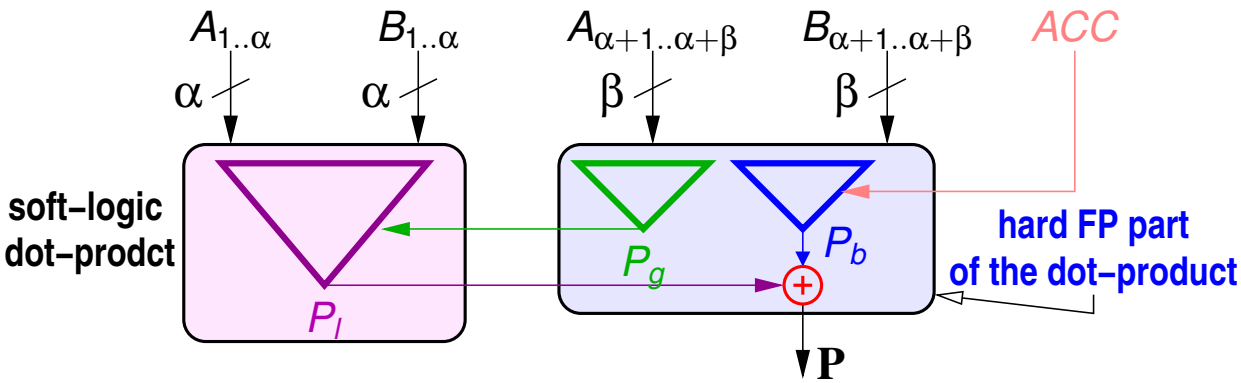


$$N = \alpha + \beta$$

$$C_{ALM} = f(\alpha, w)$$

$$C_{DSP} = \alpha/4 + \beta$$

Our implementation



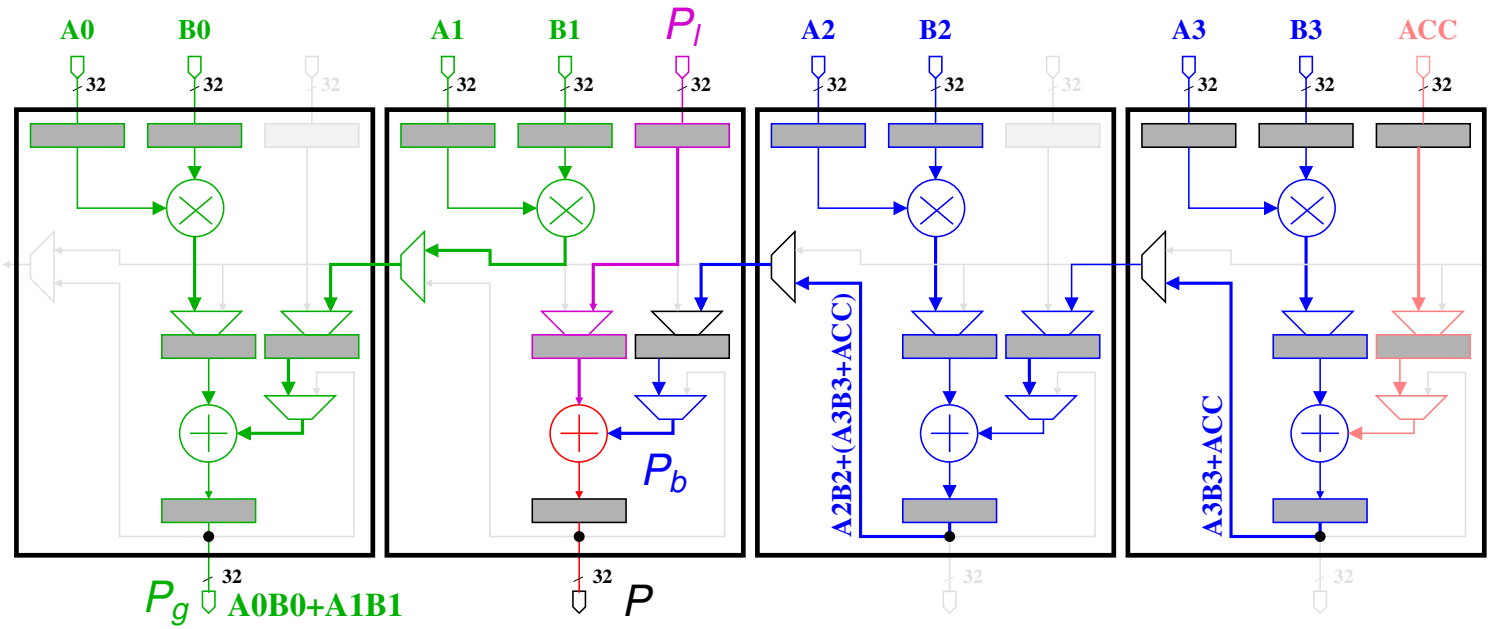
$$N = \alpha + \beta$$

$$C_{ALM} = f(\alpha, w)$$

$$C_{DSP} = \alpha/4 + \beta$$

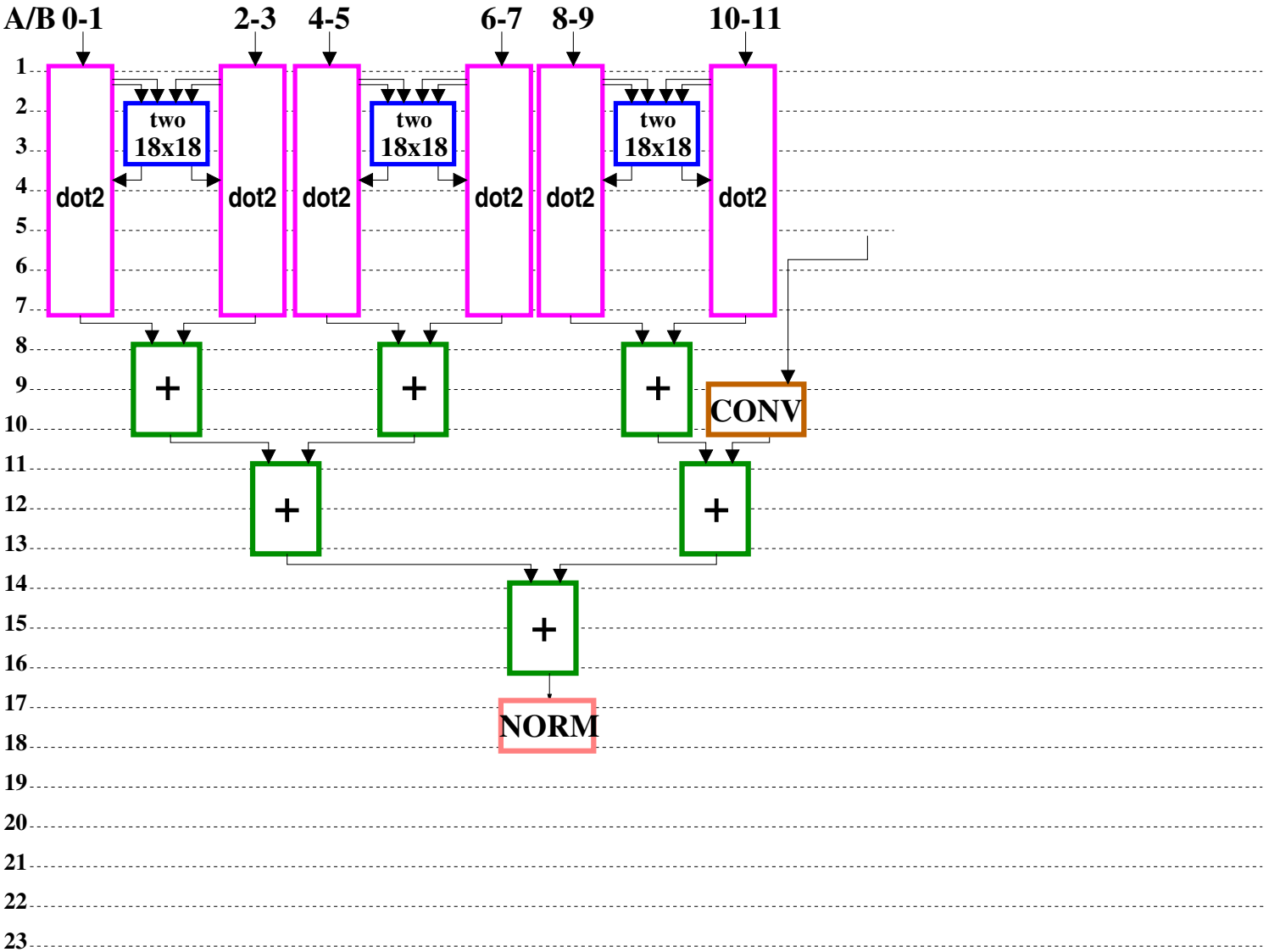
Objective: $C_{ALM}/C_{DSP} \approx$ device ALM/DSP ratio

Hard FP part

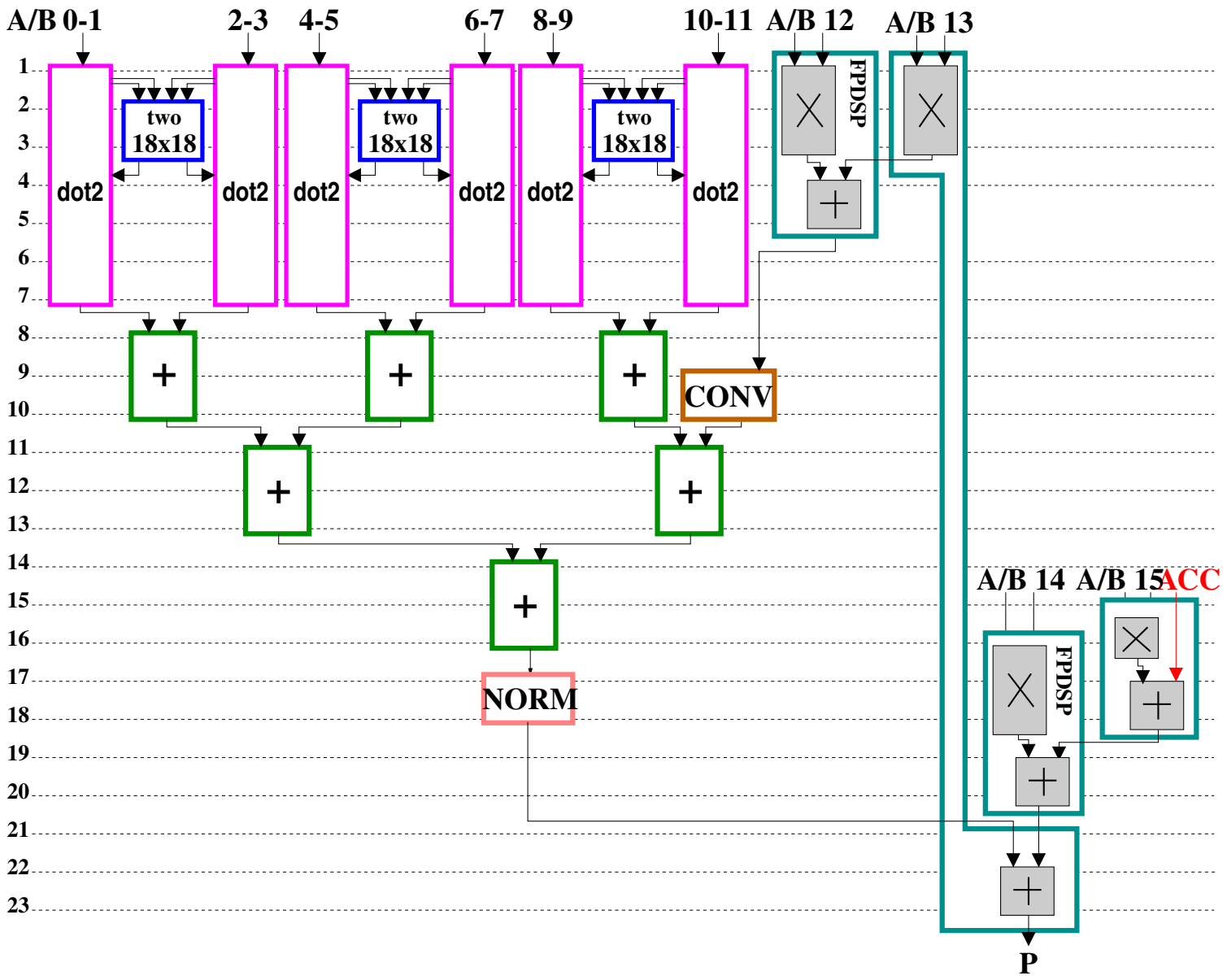


- SP accumulation integrated
- P_g will merge with the logic-based dot product
- P_l recirculated, added with P_b using spare adder

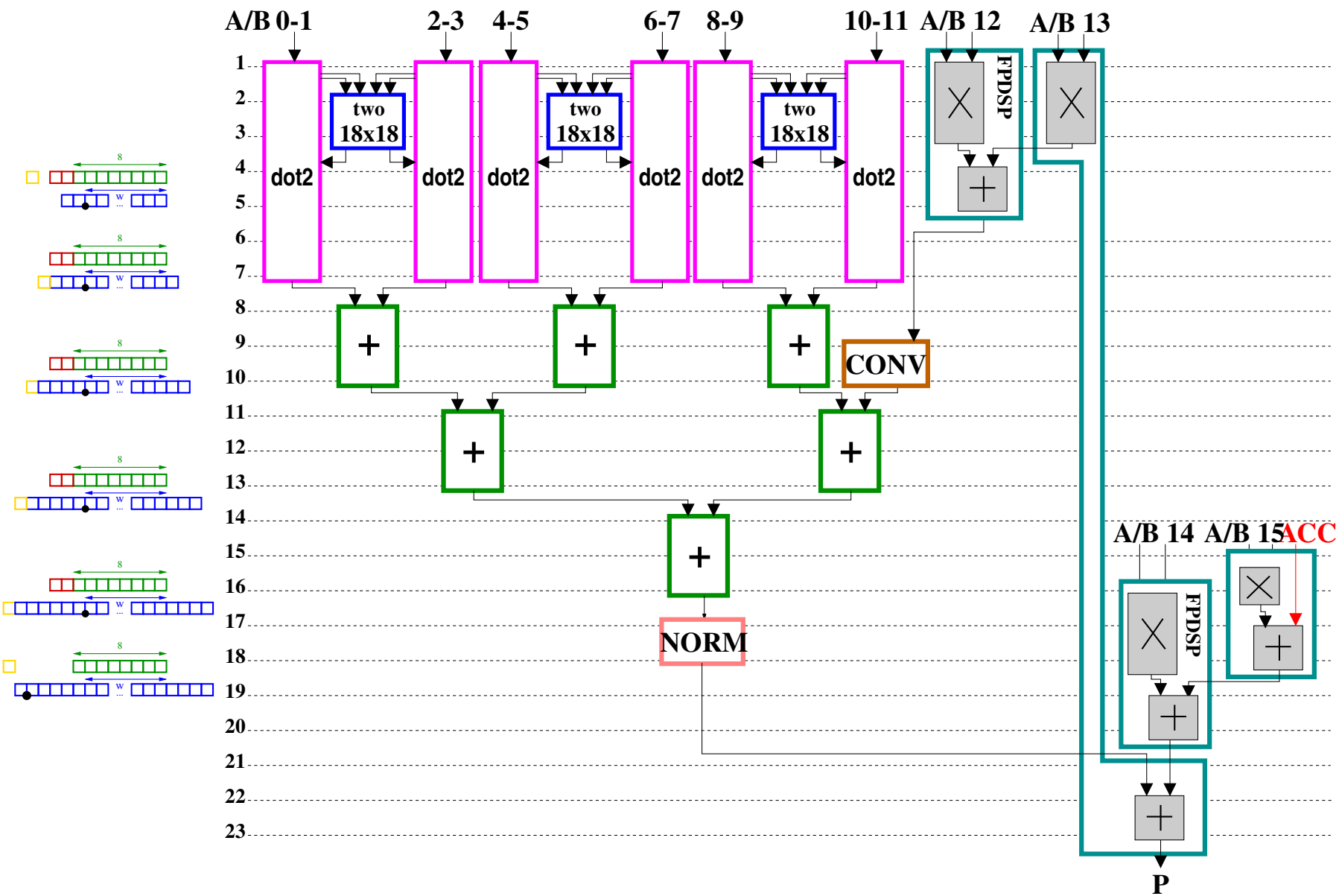
Soft FP part


A/B 0-1
2-3
4-5
6-7
8-9
10-11
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
dot2
two 18x18
+
CONV
NORM

Soft FP part


A/B 0-1
2-3
4-5
6-7
8-9
10-11
A/B 12
A/B 13
A/B 14
A/B 15
ACC
two 18x18
two 18x18
two 18x18
dot2
dot2
dot2
dot2
dot2
dot2
FPDSP
FPDSP
CONV
NORM
P
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23

Soft FP part


A/B 0-1
2-3
4-5
6-7
8-9
10-11
A/B 12
A/B 13
A/B 14
A/B 15
ACC
dot2
two 18x18
FPDSP
CONV
NORM
P
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23

Soft FP part - fused

Multipliers

- $1 DSP = 2 \times 18x18 = 4 \times 8x8$ mantissa multipliers (+ALMs)
- skip multiplier normalization
- $RN \rightarrow RZ_w$
- extend exponent to avoid overflow/underlow

Soft FP part - fused

Multipliers

- $1 DSP = 2 \times 18x18 = 4 \times 8x8$ mantissa multipliers (+ALMs)
- skip multiplier normalization
- $RN \rightarrow RZ_w$
- extend exponent to avoid overflow/underlow

Adders

- (except first layer inputs) operate on 2's complement mantissas
- mantissa grows by 1 (+1 optional) bit(s) every stage
- mantissa format changes from (SM, 1, wF)$\rightarrow$ (2C, 1+1+L, w+L)
- after final adder, normalization converts to SP

Soft FP part - fused

Multipliers

- $1DSP = 2 \times 18x18 = 4 \times 8x8$ mantissa multipliers (+ALMs)
- skip multiplier normalization
- $RN \rightarrow RZ_w$
- extend exponent to avoid overflow/underlow

Adders

- (except first layer inputs) operate on 2's complement mantissas
- mantissa grows by 1 (+1 optional) bit(s) every stage
- mantissa format changes from (SM, 1, wF)$\rightarrow$ (2C, 1+1+L, w+L)
- after final adder, normalization converts to SP
- intermediary normalization may be introduced for large α

Accuracy (Average)

- w - knob to control the accuracy
- e_c - exponents centered, e_s - the exponent span
- $e_c = 0, e_s = 10$ - inputs generated in $(2^{-10} \cdot 2, 2^{10} \cdot 2)$

Table: Average relative error comparison between the proposed hybrid dot-product and a typical AI bfloat16+SP implementation for $n = 16$, $\alpha = 12$, $\beta = 4$, $\beta_g = 2$, $\beta_b = 2$

Config	Param	Proposed	AI
$e_c = 0$, $e_s = 5$	$w = 7$	1.287601e-02	4.570449e-03
	$w = 8$	6.172194e-03	
	$w = 9$	2.935275e-03	
$e_c = 0$, $e_s = 10$	$w = 7$	7.934867e-03	3.402314e-03
	$w = 8$	4.120781e-03	
	$w = 9$	1.864206e-03	
$e_c = 0$, $e_s = 20$	$w = 7$	6.672454e-03	2.996574e-03
	$w = 8$	3.161355e-03	
	$w = 9$	1.588372e-03	

Density

$$r_{dot} = C_{DSP} / ALMs$$

Config	Param	ALMs	DSPs	r_{dot}
$n = 16$	$w = 7$	1030	7	147
$\alpha = 12, \beta = 4$	$w = 8$	1075		153
$\beta_g = 2, \beta_b = 2$	$w = 9$	1141		163
$n = 16$	$w = 7$	863	8.5	102
$\alpha = 10, \beta = 6$	$w = 8$	894		106
$\beta_g = 4, \beta_b = 2$	$w = 9$	948		112

Density

$$r_{dot} = C_{DSP} / ALMs$$

Config	Param	ALMs	DSPs	r_{dot}
$n = 16$	$w = 7$	1030	7	147
$\alpha = 12, \beta = 4$	$w = 8$	1075		153
$\beta_g = 2, \beta_b = 2$	$w = 9$	1141		163
$n = 16$	$w = 7$	863	8.5	102
$\alpha = 10, \beta = 6$	$w = 8$	894		106
$\beta_g = 4, \beta_b = 2$	$w = 9$	948		112

PRODUCT LINE		GX 160 SX 160	GX 220 SX 220	GX 270 SX 270	GX 320 SX 320	GX 480 SX 480	GX SX
Resources	LEs (K)	160	220	270	320	480	5
	System logic elements (K)	210	288	354	419	629	7
	Adaptive logic modules (ALMs)	61,510	83,730	101,620	118,730	181,790	217
	Registers	246,040	334,920	406,480	474,920	727,160	868
	M20K memory blocks	440	588	750	891	1,438	1,
	M20K memory (Mb)	9	11	15	17	28	
	MLAB memory (Mb)	1.0	1.8	2.4	2.8	4.3	5
	Hardened single-precision floating-point multipliers/adders	156/156	191/191	830/830	985/985	1,368/1,368	1,523
	18 x 19 multipliers	312	382	1,660	1,970	2,736	3,
	Peak fixed-point performance (GMACS)[1] Peak floating-point performance (GFLOPS)	394	459	122	120	132	1

Open questions

- reduction tree topology?

Open questions

- reduction tree topology?
- accuracy?

Open questions

- reduction tree topology?
- accuracy?
- resource ratio - account for plumbing?

Open questions

- reduction tree topology?
- accuracy?
- resource ratio - account for plumbing?
- integration/syncronization?

Open questions

- reduction tree topology?
- accuracy?
- resource ratio - account for plumbing?
- integration/syncronization?
- design/portability?

Open questions

- reduction tree topology?
- accuracy?
- resource ratio - account for plumbing?
- integration/syncronization?
- design/portability?
- routability?