EXPERIMENTAL ANALYSIS OF MATRIX MULTIPLICATION FUNCTIONAL UNITS

Brian Hickmann, Dennis Bradford

Motivation

- AI is driving development of several new matrix-multiplication accelerators
- However, IEEE 754 standard gives significant implementation-specific flexibility in its definition of the dot product operation
  - Summation order
  - Internal format width
  - Rounding points
  - Exception reporting
- Accelerator microarchitecture details are typically not well documented
- This work details a series of experiments that can be used to better understand the design of these accelerators
  - Exploit above flexibility to gain insight into design
- Applied this method to Tensor Core within NVIDIA V100 GPUs

Methodology

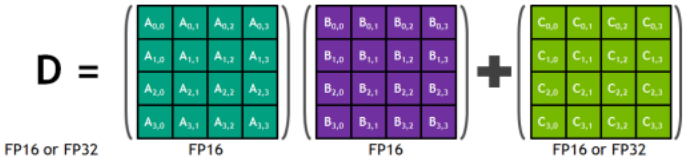
- Wanted to investigate several properties of the design:
  - NaN/Exception Behavior?
  - Rounding modes / locations?
  - How is the accumulator integrated?
  - Internal precision width?
  - Order of operations?
  - Interconnection of design units?
- First explored available documentation to understand:
  - What is the SW interface?
  - What is the smallest design unit?
- Next we designed several rounds of experiments to try to answer each question
  - Test vectors always permuted values across all inputs to understand any ordering dependencies

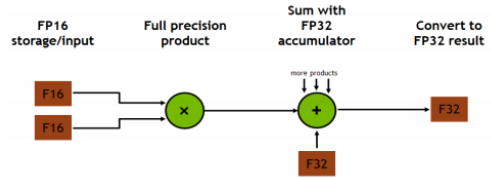
Test Vector Examples

Question	Test
Order of operations?	$2^{30} + -2^{30} + 2^{-14}$ or Big + -Big + Small Depending on operation order, expect 0.0 or 2^{-14} as result
Internal precision?	$2^{30} + 2^{N}$ where N in {30,-28} Expect 2^{N} to disappear at edge of datapath width
Rounding points and modes?	Selected products to create various “L”, “R”, and “S” bits with both positive and negative results.
Accumulator ordering and rounding?	Repeated above testing, introducing C accumulator value to understand order of summation and rounding.

Volta Tensor Cores

- Each Tensor core performs matrix multiply-accumulate or dot-product operation
  - Input data size is FP16, accumulator is FP16 or FP32
- Exposed through CUDA “wmma” instruction
  - 16x16, 4x32, and 32x4 matrices supported
  - Wrote test software using 16x16 matrix size
- Initial testing done on smallest 4-input dot product element:
  - $D_0 = a_3*b_3 + a_2*b_2 + a_1*b_1 + a_0*b_0 + C_0$





*Images from Volta Whitepaper

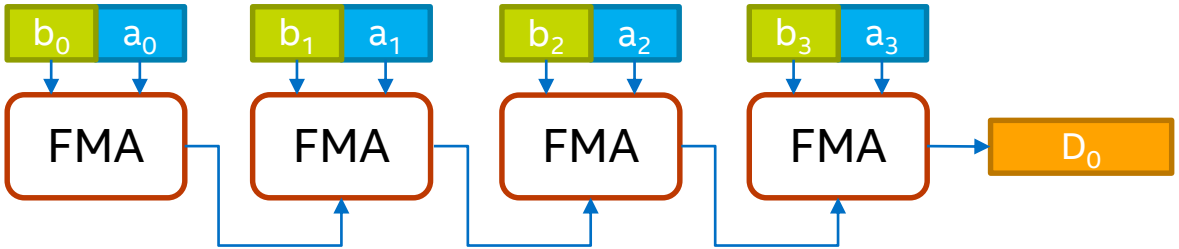
Test Vector Examples

Question	Test
Order of operations?	$2^{30} + -2^{30} + 2^{-14}$ or Big + -Big + Small Depending on operation order, expect 0.0 or 2^{-14} as result
Internal precision?	$2^{30} + -2^{30} + 2^{N}$ where N in {30,-28} Expect 2^{N} to disappear at edge of datapath width
Rounding points and modes?	Selected products to create various “L”, “R”, and “S” bits with both positive and negative results.
Accumulator ordering and handling?	Repeated above testing, introducing C accumulator value to understand order of summation and rounding.

Possible Micro-Architectures – Chain of FMAs

Test 1: $2^{30} + -2^{30} + 2^{-14} + 0 = 2^{-14}$

Test 2: $2^{-14} + 0 + 2^{30} + -2^{30} = 0$



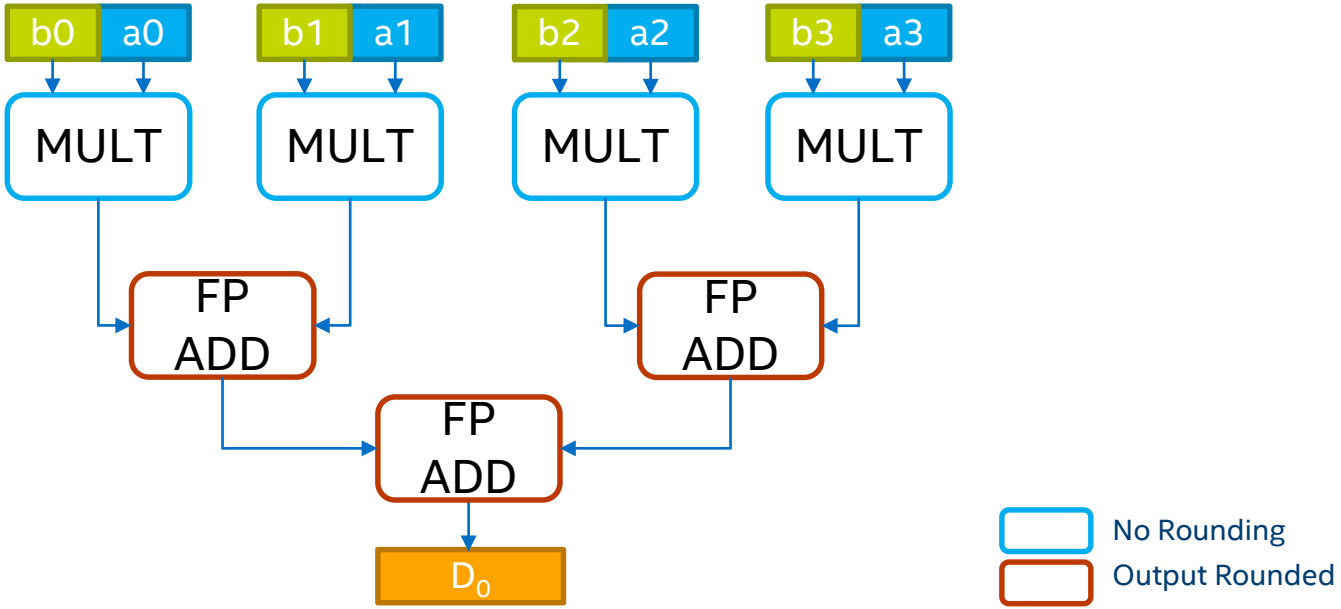
No Rounding

Output Rounded

Possible Micro-Architectures – Tree of FP Adds

Test 1: $2^{30} + -2^{30} + 2^{-14} + 0 = 2^{-14}$

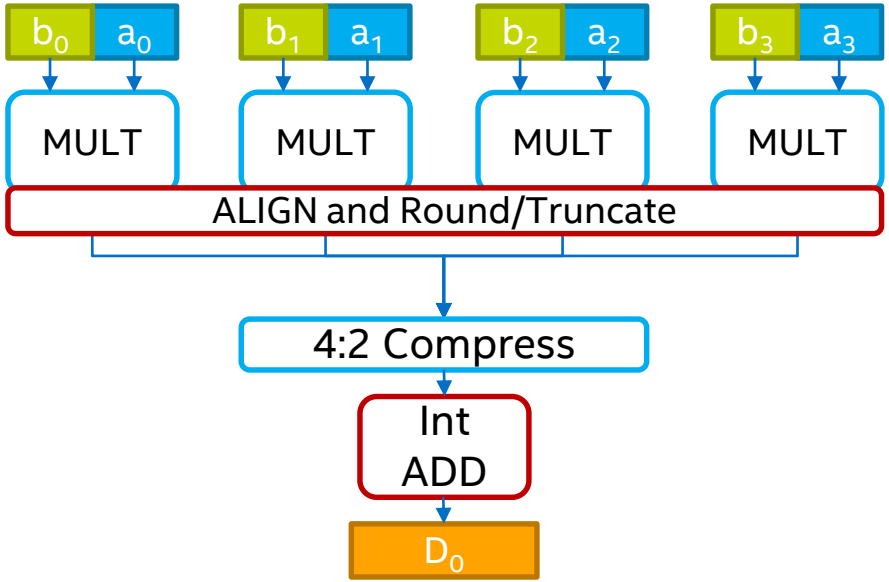
Test 2: $2^{-14} + 2^{30} + 0 + -2^{30} = 0$



Possible Micro-Architectures – Tree of INT Adds

Test 1: $2^{30} + -2^{30} + 2^{-14} + 0 = 0$

Test 2: $2^{-14} + 2^{30} + 0 + -2^{30} = 0$



No Rounding

Output Rounded

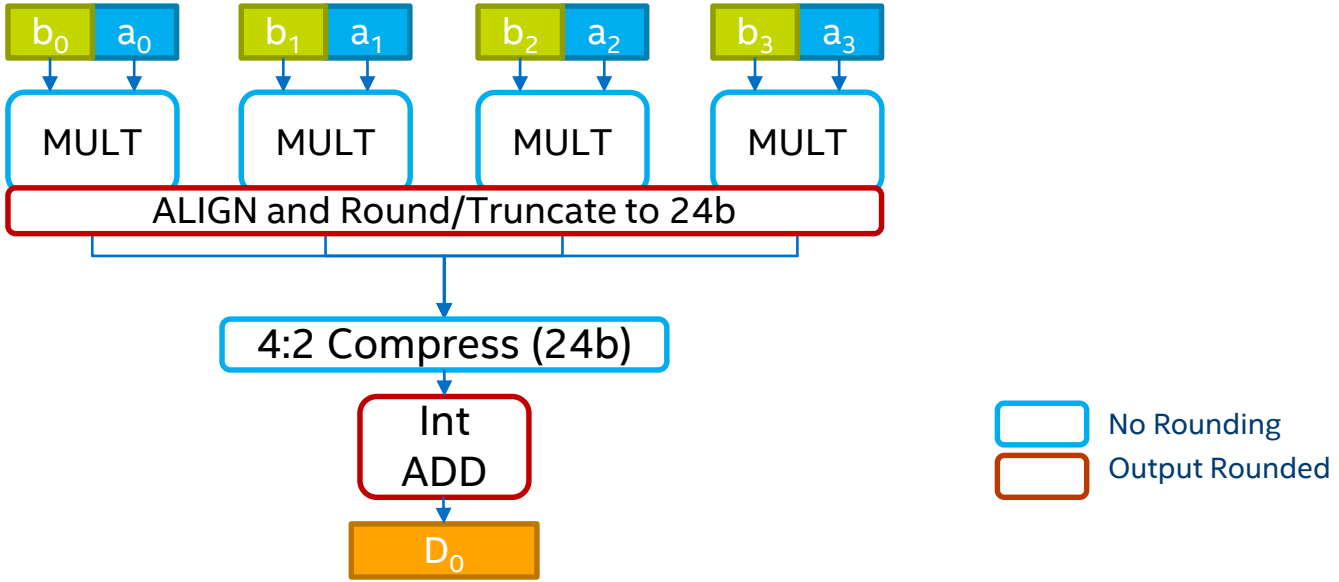
Test Vector Examples

Question	Test
Order of operations?	$2^{30} + -2^{30} + 2^{-14}$ or Big + -Big + Small Depending on operation order, expect 0.0 or 2^{-14} as result
Internal precision?	$2^{30} + -2^{30} + 2^{N}$ where N in {30,-28} Expect 2^{N} to disappear at edge of datapath width
Rounding points and modes?	Selected products to create various “L”, “R”, and “S” bits with both positive and negative results.
Accumulator ordering and handling?	Repeated above testing, introducing C accumulator with critical values to understand order of summation and rounding.

Internal Datapath Width Experimental Results

Test (N=7): $2^{30} + -2^{30} + 2^{7} + 0 = 2^{7}$

Test (N=6): $2^{30} + -2^{30} + 2^{6} + 0 = 0$

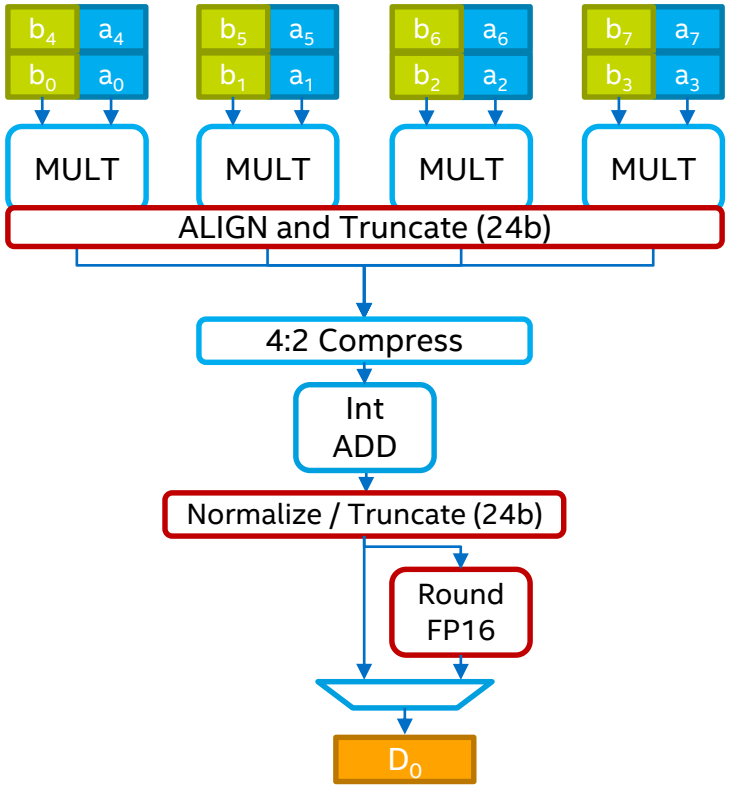


Test Vector Examples

Question	Test
Order of operations?	$2^{30} + -2^{30} + 2^{-14}$ or Big + -Big + Small Depending on operation order, expect 0.0 or 2^{-14} as result
Internal precision?	$2^{30} + -2^{30} + 2^{N}$ where N in {30,-28} Expect 2^{N} to disappear at edge of datapath width
Rounding points and modes?	Selected products to create various “L”, “R”, and “S” bits with both positive and negative results.
Accumulator ordering and handling?	Repeated above testing, introducing C accumulator value to understand order of summation and rounding.

Best Estimate of Tensor Core Microarchitecture

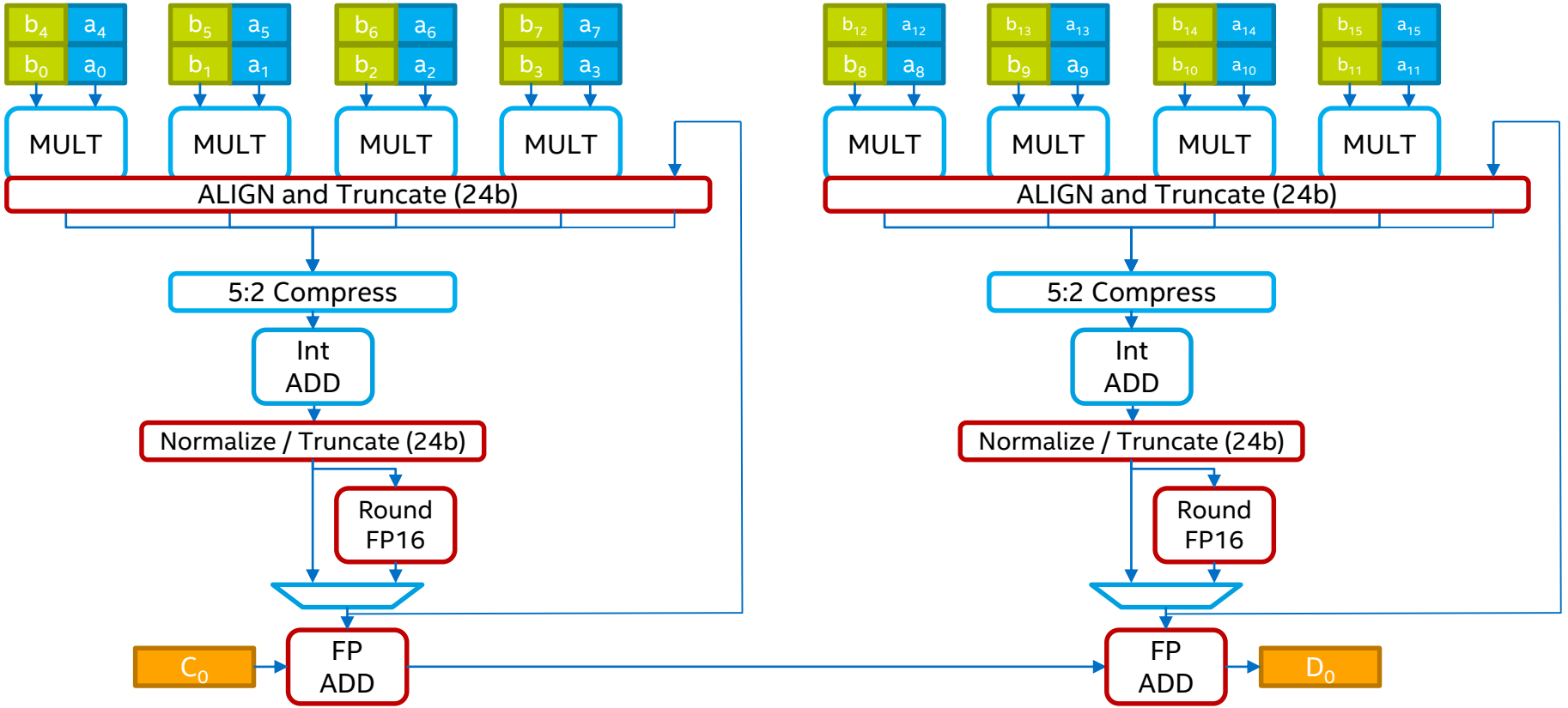
- FP32 Round Test: +/- $(1.0 + 2^{-23} + 2^{-24})$
  - No round up, so result truncated
- Integer overflow test: +/- $(1.0 + 1.0 + 2^{-23} + 2^{-24})$
  - Result is normalized and truncated to 24b
  - 2^{nd} rounding point for FP32
- FP16 Round Test: +/- $(1.0 + 2^{-10} + 2^{-11})$
  - Indicates FP16 results use RNE
- FP16 Datapath width: +/- $(1.0 + 2^{-10} + 2^{N})$
  - 2^{N} in {-12, -30}, sticky bit for rounding
  - Sticky bit truncated off at 24b.



Test Vector Examples

Question	Test
Order of operations?	$2^{30} + -2^{30} + 2^{-14}$ or Big + -Big + Small Depending on operation order, expect 0.0 or 2^{-14} as result
Internal precision?	$2^{30} + -2^{30} + 2^{N}$ where N in {30,-28} Expect 2^{N} to disappear at edge of datapath width
Rounding points and modes?	Selected products to create various “L”, “R”, and “S” bits with both positive and negative results.
Accumulator ordering and handling?	Repeated above testing, introducing C accumulator value to understand order of summation and rounding. Also expanded testing to full 16x16 matrix

Best Estimate of Tensor Core Microarchitecture



Conclusions / Future Work

- Described a testing methodology that uses software visible inputs/outputs to explore a matrix multiplication unit microarchitecture
  - Iterative testing exploits rounding modes and order of operations to gain insight into design
- Applied the methodology to Tensor Core units in NVIDIA V100 GPU
  - By analyzing many rounds of testing, we were able to synthesize a detailed estimate of the design microarchitecture.
- In future work we would like to this same methods to other designs, such as Google's TPU


intel®
experience
what's inside™

Results – Internal Architecture

- FP16 results are rounding using Round to Nearest Even
  - FP16 subnormals correctly handled
- FP32 results are rounded using truncation (Round to Zero)
  - FP32 subnormals NOT correctly handled, flushed to zero
- Internal Architecture is NOT chain of FMAs or tree of FP adders
  - FP32 Test vector (products): 2^{30} + -2^{30} + 2^{-14} or Big + -Big + small
  - Expect result of 0.0 or 2^{-14} depending on order of summation and rounding.
  - Tensor Core results were always 0.0, which implies no internal rounding.
- Internal datapath width is truncated to 24 bits, even if integer overflow
  - By varying the exponent difference between largest and smallest product, we found that all bits after the 24^{th} bit were truncated (not rounded) away.

Results – Top-level Architecture

- Testing for interconnection between dot-product units
  - Expanded testing to all 16 elements of A=[a_0..a_{15}] and B=[b_0..b_{15}] inputs
  - Call each dot-product result T_0, T_1, T_2, and T_3 (T_0 = [a_0, a_1, a_2, a_3] * [b_0, b_1, b_2, b_3])
- FP16: Tensor results always rounded using RNE
- FP32: Tensor results rounded with RNE or with truncation
  - Division found when inputs permuted between groups (T_0, T_1) and (T_2, T_3)
  - Implies that intermediate summation results are added with products directly
- C Matrix always added to result using RNE
- Summation order is: **$(C_0 + (T_0 + T_1)) + (T_2 + T_3)$**