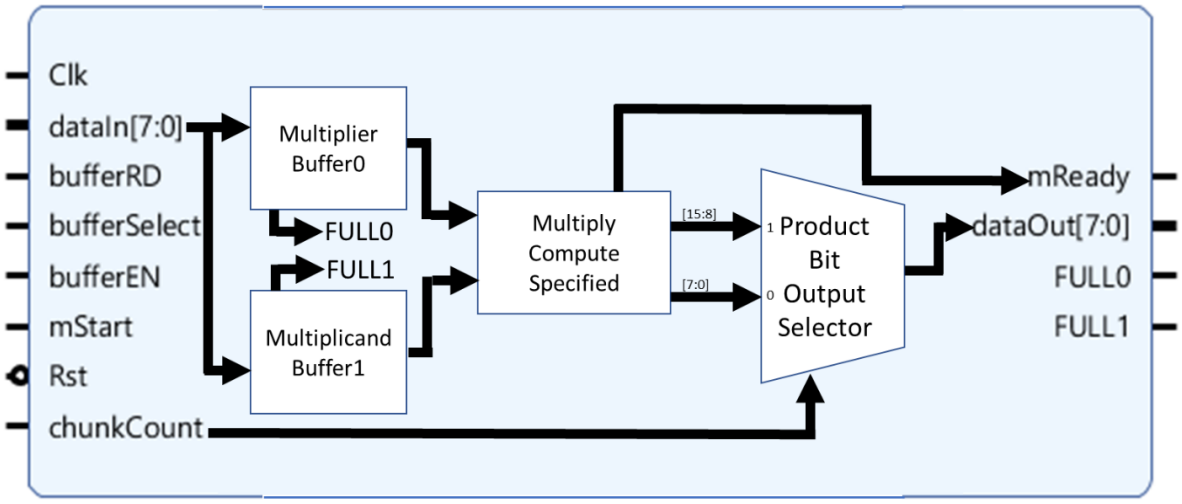
Multiplier Benchmark

Overview

Internal Block Diagram:



- The only portion of the multiplier that changes between data type is the “Multiply Compute” block.
- The block can be adjusted to n-bit data input. This benchmark will demonstrate n-bit inputs of n = 8-16-32.
- For charts shown below, data will be represented in hexadecimal, using corresponding notation.
- When using lower n-bit representations, for fixed and floating point specifically, there will be error associated with the output values due to not being able to accurately represent the number with the given bit count. This error is calculated and shown below the output data.
- To ensure all signals are zeroed and set properly, there will be a 2.5ns reset delay at the beginning. The first positive edge after this delay is when cycles to complete will begin to be counted, up until the last positive edge when all needed values are extracted.
- Input vectors used will be the same bit patterns, but due to notation structure difference, they will represent different numbers.
- Error is not a focus of the simulation but is monitored. Calculated with the following formula.
    - $Percent\ Error = \left|\frac{Calculated - Expected}{Expected}\right| * 100\%$

Integer Multiplier

Data Structure:

Simple base conversion.

Example Data:

$22 = 16 + 4 + 2 = 2^4 + 2^2 + 2^1 = 10110$

Simulation Data:

N-Bit Count	*Multiplier Hex Representation*	*Multiplier Decimal Notation*	*Multiplicand Hex Representation*	*Multiplicand Decimal Representation*
8	0xfa	250	0x25	37
16	0xfafa	64250	0x25ff	9727
32	0xfafafafa	4210752250	0xa925ff	11085311

N-Bit Count	*Calculated Output Hex Representation*	*Calculated Output Decimal Representation*	*Cycles to Complete*	*Expected Output Decimal Representation*	*Error (%)*
8	0x2422	9250	5	9250	0.00
16	0x25402106	624959750	5	624959750	0.00
32	0x00a5d4eff5502106	4.66774e16	5	4.66774e16	0.00

Floating Point Multiplier

Data Structure:

N-Bit Count	*Sign Bits*	*Bias Bits (Bias Value)*	*Mantissa Bits*
8	1	4 (-7)	3
16	1	5 (-15)	10
32	1	8 (-127)	23

Example Data:

N-Bit Count	*Decimal Value*	*Bit Representation*			*Expanded Form*
		Sign	*Bias*	*Mantissa*	
8-bit	+48.0	0	1100	100	$+2^{12-7} * (1 + \frac{1}{2})$
16-bit	-48.0	1	10100	10000000000	$-2^{20-15} * (1 + \frac{1}{2})$
32-bit	+48.0	0	10000100	10000000000000000000000	$+2^{132-127} * (1 + \frac{1}{2})$

Simulation Data:

N-Bit Count	*Multiplier Hex Representation*	*Multiplier Decimal Notation*	*Multiplicand Hex Representation*	*Multiplicand Decimal Representation*
8	0xfa	-320	0x25	+0.203125
16	0xfafa	-5.715e+4	0x25ff	2.342e-2
32	0xfafafafa	-6.51582312038e+35	0x00a925ff	+1.55338292809e-38

N-Bit Count	*Calculated Output Hex Representation*	*Calculated Output Decimal Representation*	*Cycles to Complete*	*Expected Output Decimal Representation*	*Error (%)*
8	0xe8	-64	4	-65	1.54
16	0xe66f	-1647	4	-1339	18.70
32	0xbc4ba9d4	-1.2430627e-2	4	-0.01012156839	22.81

Fixed Point Multiplier

Data Structure:

N-Bit Count	*Integer Bits*	*Fractional Bits*
8	4	4
16	8	8
32	16	16

Example Data:

N-Bit Count	*Decimal Value*	*Bit Representation*		*Expanded Form*
		Integer	*Fractional*	
8-bit	12.875	1100	1110	$2^3 + 2^2 + 2^{-1} + 2^{-2} + 2^{-3}$
16-bit	100.00390625	1100100	00000001	$2^6 + 2^5 + 2^2 + 2^{-8}$
32-bit	4100.250244	1000000000100	0010000000001	$2^{12} + 2^2 + {}^{-2} + 2^{-12}$

Simulation Data:

(Overflow occurred with the previous bit values, so they were substituted.)

N-Bit Count	*Multiplier Hex Representation*	*Multiplier Decimal Notation*	*Multiplicand Hex Representation*	*Multiplicand Decimal Representation*
8	0x29	2.5625	0x44	4.25
16	0x2929	41.16015625	0x051f	5.12109375
32	0x027d1100	637.06640625	0x003d1100	61.06640625

N-Bit Count	*Calculated Output Hex Representation*	*Calculated Output Decimal Representation*	*Cycles to Complete*	*Expected Output Decimal Representation*	*Error (%)*
8	0xae	10.875	4	10.890625	0.14
16	0xd2c8	210.78125	4	210.7850189	0.00
32	0x97c9	38903.78515625	4	68903.35597	0.00