A Vector Systolic Accelerator for Multi-Precision Floating-Point High-Performance Computing

***Abstract*—There is an emerging need to design multi-precision floating-point (FP) accelerators for high-performance-computing (HPC) applications. However, the existing multi-precision design using high-precision-split method and low-precision-combination method suffers either low hardware utilization rate and long multiple clock-cycle processing period, respectively. In this paper, a new pipelined multi-precision FP processing element (PE) is developed with proposed redundancy-minimized bit-partitioning method. 3.8× throughput improving is achieved by the elaborate designed pipeline. Compared with the existing multi-precision FP-PE method, this work achieves 11.7%, 6.7%, 62.6% enhancement on energy-efficiency at FP16, FP32 and FP64 operations, respectively. Moreover, to further improve the system-level throughput and energy efficiency, a vector systolic accelerator is employed. Benefit from the pipelined vector FP-PE and vector systolic date reuse, the proposed accelerator exhibits the best energy-efficiency performance of 1193 GFLOPS/W at FP16, 298.3 GFLOPS/W at FP32 and 74.6 GFLOPS/W at FP64.**

***Index Terms*—multi-precision, vector, floating-point, Systolic.**

I. Introduction

Floating-point (FP) [1] based high-performance computing (HPC) has been widely utilized in scientific simulation, model training and big data analysis [2]–[4]. However, the high-precision FP calculation brings extremely high computing power and long latency time [5] [6]. Consequently, FP arithmetic of assorted precision is commonly employed in many intensive computing applications to improve performance, especially to accelerate scientific computations with multi-precision algorithms.

So far, great efforts have been spared on the design of multi-precision FP process element (PE) [7]–[10]. The main problem is that the multiplication part of those designs is too redundant, resulting in low resource utilization of the hardware. In particular, large-scale integration of these highly redundant designs will result in a significant waste of hardware resources. On the other hand, some works such as [9] achieved multi-precision by truncating the mantissa to lower bitwidth, which causes accuracy loss. [11], [12] proposed a bit-partitioning method of mantissa to minimize redundant bits and increase resource utilization. However, the total redundant bits per segment is still high due to fused multiplier.

In this work, we propose a vector systolic accelerator integrated with multi-precision FP vector processing element (PE) to overcome the above issues. Fig. 1 shows the overall architecture of the accelerator. The PE unit can support multi-precision FP calculations with minimized redundancy, and the systolic accelerator can support multi-precision FP matrix multiplication with improved energy-efficient performance compared with previous works.

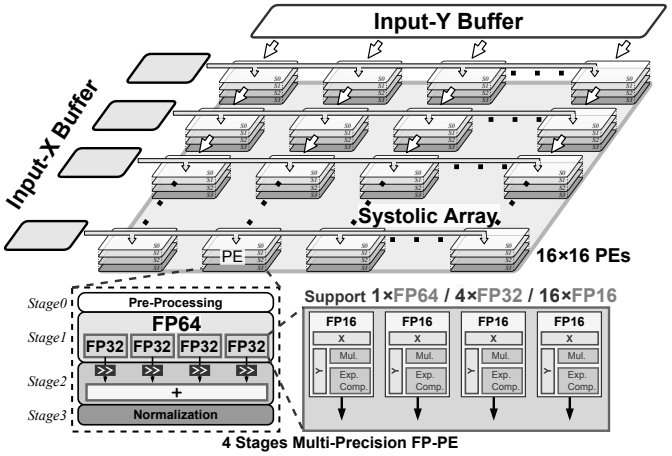

Fig. 1. Architecture of the vector systolic array, and the basic computation unit is pipelined multi-precision FP vector PE.

The remainder of this paper is organized as follows. Section II analyses proposed multi-precision minimized bit-partitioning method. Section III introduce the design of pipelined vector PE and vector systolic dataflow. Section VI and V provides the experiment results and conclusion, respectively.

II. Multi-Precision Minimized Bit-Partitioning

A. Multi-Precision Floating-Point Method

Based on the IEEE 745 standard [1], floating-point A and B can represented by Eq. 1 and 2. Then A times B can be expressed by Eq. 3. For the vector operation, the vector of $\boldsymbol{A}$ with length L dot product with the vector of $\boldsymbol{B}$ with length L can be expressed by Eq. 4, where the parameters used in Eq. 1 to 7 are summarized in Table I.

$$A = (-1)^{sa} \times M_a \times 2^{(E_a - bias)} \tag{1}$$

$$B = (-1)^{s_b} \times M_b \times 2^{(E_b - bias)} \tag{2}$$

$$A \times B = (-1)^{s_a + s_b}(M_a \times M_b)2^{(E_a + E_b - 2bias)} \tag{3}$$

Therefore, to implement multi-precision FP operation, the multi-precision operation of the mantissa is necessary to be designed elaborately.

$$\begin{aligned}\boldsymbol{A}\cdot\boldsymbol{B} &= \sum_{n=1}^{L}(-1)^{s_{an}+s_{bn}}(M_{bn}\times M_{an})2^{(E_{an}+E_{bn}-2bias)} \\ &= \sum_{l=0}^{\lceil M/\sigma\rceil-1}\sum_{k=0}^{\lceil M/\sigma\rceil-1}\sum_{n=1}^{L}\sum_{j=0}^{\sigma-1}\sum_{i=0}^{\sigma-1}(-1)^{s_n}(M_{an}[j+l\sigma] \\ &\quad \times M_{bn}[i+k\sigma])2^{i+j+(l+k)\sigma-2(M-1)+E_n}\end{aligned} \tag{4}$$

TABLE I
SUMMARY OF PARAMETERS IN EQUATION 1 TO 6

Parameter	Description
s_{an}, s_{bn}	The sign bit of n^{th} element of vector$\boldsymbol{A}$ and $\boldsymbol{B}$.
M_{an}, M_{bn}	The mantissa of n^{th} element of vector $\boldsymbol{A}$ and $\boldsymbol{B}$.
E_{an}, E_{bn}	The exponent of n^{th} element of $\boldsymbol{A}$ and $\boldsymbol{B}$.
$bias$	The $bias$ is 15, 127 and 1023 of FP16, FP32 and FP64.
M	M is 11, 24 and 53 of FP16, FP32 and FP64.
L	The vector length of the $\boldsymbol{A}$ and $\boldsymbol{B}$.
σ	The length of splitting mantissa per segment.

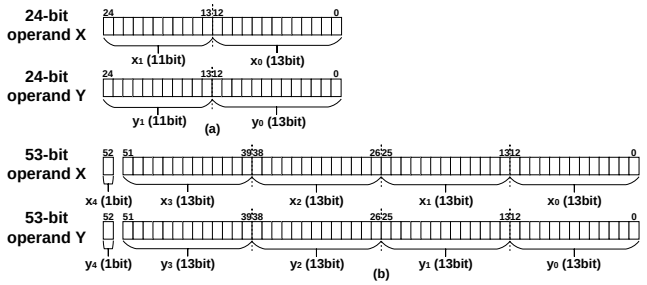

Fig. 2. Bit-partitioning of the operands. (a) 24-bit mantissa for FP32. (b) 53-bit mantissa for FP64.

B. Minimized Bit-Partitioning Analysis

Based on Eq. 4, Table II analysis the invalid operand in different value of σ. The total operand is the counts of all cycle in Eq. 4. The valid operand is the counts that FP operation actually needed. The invalid operand is the difference between the total operand and valid operand. Eq. 5 to 7 show the details.

$$Total\ Operand = \lceil M/\sigma \rceil^2 \times L \times \sigma^2 \tag{5}$$

$$Valid\ Operand = L \times M^2 \tag{6}$$

$$Invalid\ Operand = Total\ Operand - Valid\ Operand \tag{7}$$

In generally, the value of segments is depended on the ceil of M/σ as Table II shown. The selection of segments have to satisfy all three of these precision operations including FP16, FP32 and FP64. Therefore, the segments should be the maximum segments in all three precision. Sometimes, considering the maximum effective calculation, the segments selected should be the least common multiple (LCM) of the three precision segmentation numbers or as close to LCM as possible. In addition, to minimized the hardware cost, the cycle of n can be reused by changing the cycle of l and k in Eq. 4. Therefore, the total cycle of l, k and n ($\lceil M/\sigma \rceil^2 \times L$) should be less than segments. Under this principle, To find the minimized invalid operand, the analysis results are shown as the Table II.

For the value of σ is 13, the proposed work reduces the number of segments of FP64 to 4 because the value of M/σ is 4.07 which is closer to 4 instead of rounding up to 5. In this way, the proposed work is designed by the minimized bit-partitioning method with minimized total invalid operand 1168.

The specific bit-partitioning implementation scheme is shown in the Fig.2. The 24bit mantissa of FP32 is divided into two segments: 10bit and 13bit. The 53bit mantissa of FP64 is divided into two parts: 1 bit and 52bit, and the 52bit part is divided into four segments: 13, 13, 13, 13. The extra 1bit is processed separately.

TABLE II
ANALYSIS OF MINIMIZED BIT-PARTITIONING

Design		-	[11]	-	-	[13]	Proposed
Segments Length (σ)		11	12	13	14	15	13
Segments ($\lceil M/\sigma \rceil^2$)	FP16	1	1	1	1	1	1
	FP32	9	4	4	4	4	4
	FP64	25	25	25	16	16	16 ($\lfloor M/\sigma \rfloor^2$)
Segments Selected		27	25	25	16	16	16
Vector Length (L)	FP16	27	25	25	16	16	16
	FP32	3	6	6	4	4	4
	FP64	1	1	1	1	1	1
Total Operand		3267	3600	4225	3136	3600	2704
Invalid Operand	FP16	0	575	1200	1200	1664	768
	FP32	1539	144	769	832	1296	400
	FP64	458	791	1416	327	791	0
Total Invalid Operand		**1997**	**1510**	**3385**	**2359**	**3751**	**1168**

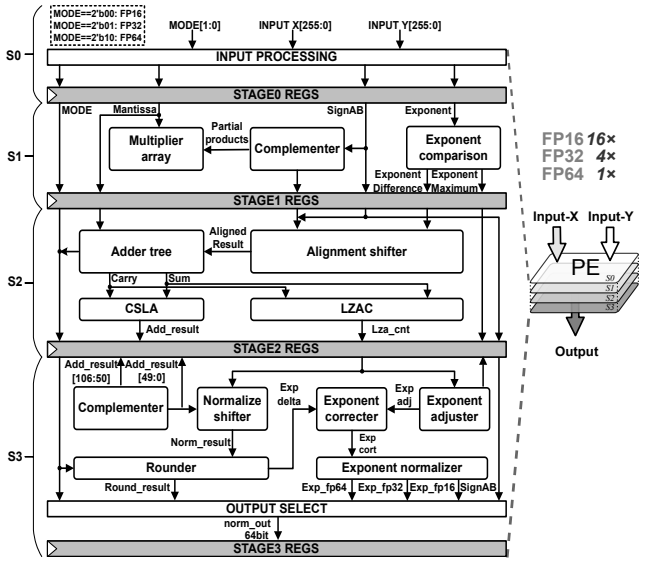

Fig. 3. Structure of vector FP-PE with 4 pipeline stages

III. PIPELINED PE AND VECTOR SYSTOLIC SYSTEM

A. Multi-Precision Vector FP PE with Pipeline

To improve the throughput of the proposed PE, 4 stages of pipeline is designed elaborately. The implementation of multi-precision vector FP-PE with pipeline is shown as Fig.3.

Stage0 is the input preprocessing stage which will split the 256-bit X and Y input into the corresponding sign bits, exponent bits and mantissa bits according to different mode.

After input prepossessing, exponent bits and mantissa bits will enter the exponential comparator and multiplier array respectively in **Stage1**. The exponential comparator compares sixteen exponents to get sixteen groups of exponent differences of 10-bit and one maximum exponent.

As the main part of **Stage1**, the multiplier array consists of 16 13-bit×13-bit multipliers. It implements vector multiplication to

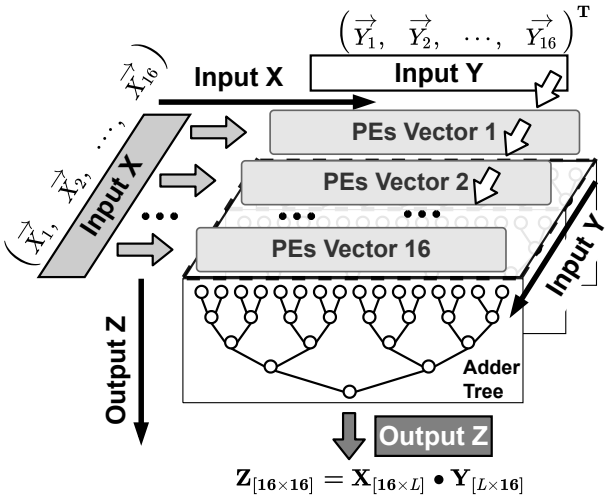

Fig. 4. Structure and dataflow of vector systolic array.

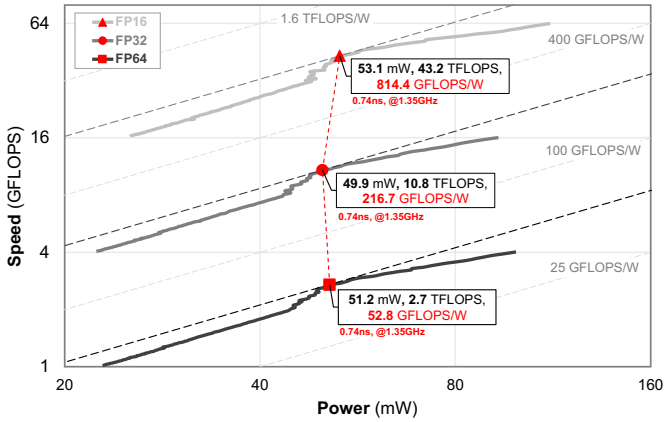

Fig. 5. Speed versus power at different timing-constraints.

generate 16 26-bit products which are the basis for 16×FP16, 4×FP32, 1×FP64 operation in one clock.

In **Stage2** as show in Fig.3, 16 products are sent to alignment shifter. The shift number for each products will depend on exponent differences and the sum of the segment number.After alignment, adder tree is used for vector accumulation.

Normalization is in **Stage3**, the LZAC [10], [14] is used. And the mantissa will be rounded according to Roundtoward-Positive. Meanwhile, the processing of abnormal values, 0 and invalid values are designed.

B. Vector Systolic Accelerator

Next, we will tend to the the integration of FP-PEs into systolic array for matrix multiplication. The vector systolic array structure and its dataflow are depicted in Fig. 4, in consideration of hardware cost and bandwidth limitations, the proposed array consists of 16 numbers of PEs vectors (PEVecs), each PEVec contains 16 multi-precision PEs as while as one 16-input FP adder tree. The introduce of systolic dataflow make it possible to perform 8192 FP16 operations, 2048 FP32 operations or 512 FP64 operations every single clock efficiently.

The input-**X** matrix consists of 16 row vectors $\vec{X_1} - \vec{X_{16}}$ with the shape of $[16 \times L]$, where length L of each vector is `256,64,16` with data format of `FP16,FP32,FP64`, respectively. The input-**Y** is similarly but transposed. The data flow can be described as follows. The input-**X** matrix data is **Stationary** that it will be first horizontally preloaded to the systolic array one-row-per-cycle and then remain for 16 clocks. The input-**Y** dataflow is **Systolic-Flow** and will pass through the systolic array in a row-by-row style. And the output-**Z** dataflow is **Pipelined** and will finally output from adder tree as the style of stage-by-stage. In each computation cycle, input **X** and **Y** are reused 16 times, which saves 16 times I/O bandwidth and the energy consumption with data access, at meanwhile, input-**X**'s stationary makes it possible to reduce the circuit toggle rate, which will further benefit energy performance.

IV. Experiment Results

A. Methodology

The performance evaluation is implemented on SMIC 28-nm process with 1.0 V and 25 °C. The synthesis, dynamic timing simulation and power performance simulation are implemented on Synopsys Design Compiler, VCS and PrimeTime PX.

In order to obtain the best timing-constraint for energy-efficiency of PE and use it in future higher-level evaluation, we set a timing-constraint range*[0.5:0.01:2.0]* in ns and get the Performance, Power and Area (PPA) of each timing-constraint.

Fig. 5 represents the speed and power with different timing-constraints, speed in GFLOPS and power in mW. The three lines correspond to the simulation results of FP16, FP32, and FP64, respectively. The coordinate is LOG4-LOG2 plot, and different dashed-lines are iso-energy-efficient lines corresponding different energy-efficient. The experimental result shows that our design obtains its best performance under the condition that timing-constraint of 0.74ns and frequency of 1351MHz. And the best energy efficiency is 814.4, 216.7, and 52.8 in GFLOPS/W with the three modes, respectively.

B. Performance Evaluation

Table III summarized the FP-PE performance in different works, including process, frequency, latency, area, voltage, power, throughput and energy efficiency.

As analyzed above, [7] splits high precision multiplier to support one FP64 operation and two FP32 operations, leading to long latency and large area overhead because of its large bitwidth multiplication. With the similar splitting method to support FP32 and FP16 operation, [16] also shows not that good performance. [9] which can perform FP32 operation and truncates the mantissa to lower bitwidth to support FP16 operation also have a poor latency and then lead to poor throughput and energy efficiency. These designs can all be summarized as splitting high-precision multipliers to implement low-precision floating-point operations, and they all face the problem of low hardware utilization rate.

[10] use 15-bit multipliers to optimized for FP128 operation. The implementation for FP16, FP32 and FP64 operations have large bit redundancy. So as shown in Table III, the area of

TABLE III
COMPARISON OF THE FP-PE PERFORMANCE WITH PREVIOUS WORKS.

Design		Node (nm)	Frequency (MHz)	Latency (ns)	Area (mm^2)	Voltage (V)	Power (mW)	Throughput (GFLOPS)			Energy Efficiency (GFLOPS/W)		
								FP16	FP32	FP64	FP16	FP32	FP64
ICECS	**2010 [7]**	130	291	10.29	0.287	1.2	35.2	-	0.58	0.29	-	33.13	16.52
ISSCC	**2012 [15]**	32	1450	2.07	0.045	1.05	60.0	2.90	1.45	-	96.67	52.00	-
ARAB	**2016 [9]**	90	357	8.4	0.082	1.0	250.5	1.43	1.43	-	11.41	11.41	-
arXiv	**2016 [16]**	28	1360	4.41	0.018	0.8	18.38	-	1.36	1.36	-	110.00	43.70
TCOM	**2019 [10]**	90	667	4.5	0.795	1.0	43.8	2.67	1.33	0.67	121.77	60.88	30.44
TVLSI	**2020 [17]**	22	923	3.25	0.049	0.8	57.41	3.69	1.85	0.92	497.67	199.70	74.83
DATE	**2021 [11]**	28	1351	2.96	0.0126	1.0	38.93	27.0	6.76	1.35	748.67	173.25	32.41
This Work	**PE (Without Pipeline)**	**28**	**337**	**2.96**	**0.0175**	**1.0**	**16.87**	**10.78**	**2.69**	**0.67**	**639.2**	**159.8**	**39.9**
	PE (With Pipeline)	**28**	**1351**	**2.96**	**0.0184**	**1.0**	**51.4**	**43.2**	**10.8**	**2.70**	**814.4**	**216.7**	**52.8**
	Systolic Accelerator	**28**	**1351**	**2.96**	**5.33**	**1.0**	**9275**	**11067**	**2767**	**691.7**	**1193**	**298.3**	**74.6**

this degisn is very large. In [17], best energy efficiency is achieved for FP64 operation with fully hardware utlization rate by contractive bitwidth processing ability. However, for FP16 operation, its energy efficiency is only half of that of our proposed design.

[11] has considered the redundancy bits achieved improved throughput and energy efficiency compared with previous work. But it still has not reached the minimum redundancy due to fused multiplier. Compared with [11], the proposed work based on the 13bit multiplier have 11.7%, 6.7%, 62.6% enhancement on energy efficiency at FP16, FP32 and FP64 operations, respectively. Compare with the design without pipeline, based on the elaborate designed pipeline, the proposed design achieve about 3.8× throughput improving. The largest throughput and energy efficiency can reach 51.2 GFLOPS and 814.4 GFLOPS/W.

Further improving throughput as while as energy efficiency, the vector systolic accelerator is designed for the matrix multiplication which achieve 11.06 TFLOPS at FP16, 2.76 TFLOPS at FP32 and 0.69 TFLOPS at FP64. The data reuse within systolic dataflow promises a lower circuit toggle rate and, therefore, a better PPA performance than a single FP-PE is achieved, which is 1193 GFLOPS/W at FP16, 298.3 GFLOPS/W at FP32 and 74.6 GFLOPS/W at FP64.

V. CONCLUSION

In this work, redundancy-minimized bit-partitioning method is proposed to develop a multi-precision vector FP PE with pipeline designed elaborately. To further improve throughput, the vector systolic accelerator is designed for the matrix multiplication. The developed FP vector PE supports 16× FP16, 4× FP32 and 1× FP64 MAC operation with almost 100% hardware utilization rate and using 256 PEs to design the vector systolic array. The energy-efficiency exhibits 11.7%, 6.7% and 62.6% improvements for FP16, FP32 and FP64 operations when compared with previous works. High energy-efficiency of 1193.2 GFLOPS/W at FP16, 298.31 GFLOPS/W at FP32, and 74.58 GFLOPS/W at FP64 have been achieved, which is benefit from the pipelined vector FP-PE and vector systolic date reuse.

REFERENCES

[1] "Ieee standard for floating-point arithmetic," *IEEE Std 754-2008*, pp. 1–70, 2008.

[2] D. Goldberg, "What every computer scientist should know about floating-point arithmetic," *ACM computing surveys (CSUR)*, vol. 23, no. 1, pp. 5–48, 1991.

[3] J. A. Kahle and *et al.*, "2.1 summit and sierra: designing ai/hpc supercomputers," in *2019 IEEE International Solid-State Circuits Conference-(ISSCC)*. IEEE, 2019, pp. 42–43.

[4] Y. Tanimura and *et al.*, "Building and evaluation of cloud storage and datasets services on ai and hpc converged infrastructure," in *2020 IEEE International Conference on Big Data (Big Data)*. IEEE, 2020, pp. 1992–2001.

[5] I. Sourdis and *et al.*, "Resilient cmps with mixed-grain reconfigurability," *changes*, vol. 9, p. 10, 2015.

[6] I. Sourdis, D. A. Khan, A. Malek, S. Tzilis, G. Smaragdos, and C. Strydis, "Resilient chip multiprocessors with mixed-grained reconfigurability," *IEEE Micro*, vol. 36, no. 1, pp. 35–45, Jan 2016.

[7] K. Manolopoulos and *et al.*, "An efficient dual-mode floating-point multiply-add fused unit," in *2010 17th IEEE International Conference on Electronics, Circuits and Systems*. IEEE, 2010, pp. 5–8.

[8] L. Huang and *et al.*, "A new architecture for multiple-precision floating-point multiply-add fused unit design," in *18th IEEE Symposium on Computer Arithmetic (ARITH'07)*. IEEE, 2007, pp. 69–76.

[9] S.-R. Kuang and *et al.*, "A multi-functional multi-precision 4d dot product unit with simd architecture," *Arabian Journal for Science and Engineering*, vol. 41, no. 8, pp. 3139–3151, 2016.

[10] H. Zhang and *et al.*, "Efficient multiple-precision floating-point fused multiply-add with mixed-precision support," *IEEE Transactions on Computers*, vol. 68, no. 7, pp. 1035–1048, 2019.

[11] W. Mao and *et al.*, "A reconfigurable multiple-precision floating-point dot product unit for high-performance computing," in *2021 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2021, pp. 1793–1798.

[12] W. Mao and *et al.*, "A configurable floating-point multiple-precision processing element for hpc and ai converged computing," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 2021.

[13] H. Zhang, D. Chen, and S. Ko, "Efficient multiple-precision floating-point fused multiply-add with mixed-precision support," *IEEE Transactions on Computers*, vol. 68, no. 7, pp. 1035–1048, July 2019.

[14] M. Schmookler and K. Nowka, "Leading zero anticipation and detection-a comparison of methods," in *Proceedings 15th IEEE Symposium on Computer Arithmetic. ARITH-15 2001*, 2001, pp. 7–12.

[15] H. Kaul and *et al.*, "A 1.45 ghz 52-to-162gflops/w variable-precision floating-point fused multiply-add unit with certainty tracking in 32nm cmos," in *2012 IEEE International Solid-State Circuits Conference*. IEEE, 2012, pp. 182–184.

[16] J. Pu and *et al.*, "Fpmax: A 106gflops/w at 217gflops/mm2 single-precision fpu, and a 43.7 gflops/w at 74.6 gflops/mm2 double-precision fpu, in 28nm utbb fdsoi," *arXiv preprint arXiv:1606.07852*, 2016.

[17] S. Mach and *et al.*, "Fpnew: An open-source multiformat floating-point unit architecture for energy-proportional transprecision computing," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 29, no. 4, pp. 774–787, 2020.