International Institute of Information Technology Bangalore

Carry Disregard Approximate Multipliers

April 16, 2024

1. Dilli Babu Porlapothula-MS2023005
2. Yogesh Goyal - IMT2021542
3. Akash Perla - IMT2021530
4. Vaibhav Tiwari - MT2023524
5. Pratikkumar AshokKumar Solanki - MT2023527

CONTENTS

Introduction

- Multiplication is one of the most common numerical operations and conventional exact multipliers have **significant critical path delays, area and power** consumption.

- Approximate computing to improve performance in computing systems.

- Leveraging human perceptual limitations to allow for approximations in error-tolerance applications like **Image Processing** and **Multimedia.**

- We propose **N-bit approximate** array multipliers based on **carry-disregarding**.

Background

- Multiplication consist of three main computation stages: **Partial Product (PP) generation, PP reduction and final addition.**
- We can employ approximation in any of these 3 stages.
- **Array multiplier** is similar to how we perform multiplication with pen and paper i.e. finding a partial product and adding them together.

		a_3	a_2	a_1	a_0		
x		b_3	b_2	b_1	b_0		
		p_{30}	p_{20}	p_{10}	p_{00}		
	p_{31}	p_{21}	p_{11}	p_{01}	x		
p_{32}	p_{22}	p_{12}	p_{02}	x	x		
p_{33}	p_{23}	p_{13}	p_{03}	x	x	x	
z_7	z_6	z_5	z_4	z_3	z_2	z_1	z_0

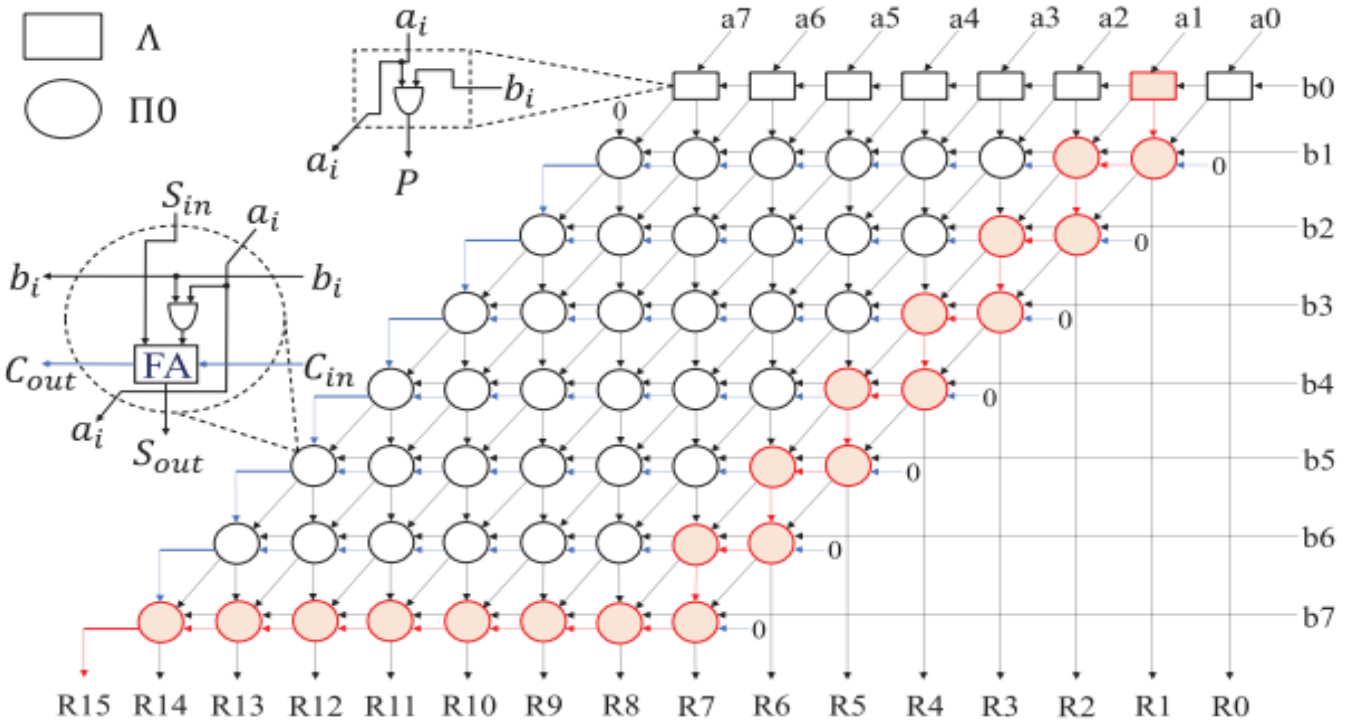


Fig. 1. Conventional exact 8-bit multiplier architecture and logic circuit of partial product unit (Π0).

Related works

- Tree-based structures like the **Wallace** and **Dadda** trees, which offer **lower delay** but consume **more power and area** compared to array-based designs and higher complexity .

- Additionally, **Operand truncation** (Truncating the input operands from N − 1 bits to h bits) reduces the size of adders and bit-width shifters. Hence, design metrics such as **area, delay, and power decrease**, while output **error increases.**

•The array-based multiplier's benefits: straightforward, uniform, modular architecture, and typically **less power consumption and area** .

•By integrating **carry disregarding in** array multipliers and refining the logical functions, we achieve significant improvements in critical metrics such as **delay, power consumption, and area utilization compared to previous methods.**

Proposed Work

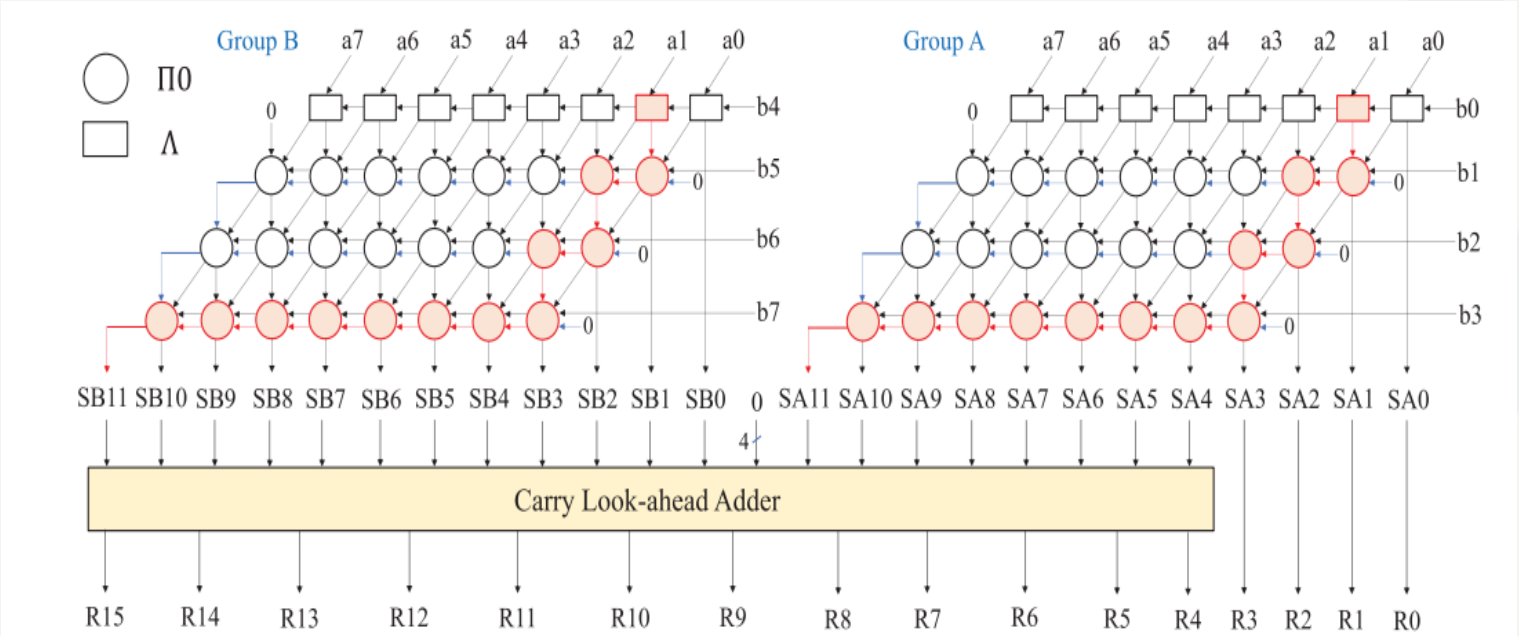
- By **distributive property** we can convert large scale multiplications into smaller one. we can convert exact 8 bit multiplication to two 8*4 multiplication.

$$A \times B = A(BH \times 2^4 + BL) = (A \times BH)2^4 + (A \times BL)$$

- A, B are 8 bit and BH and BL are MSB & LSB bits.

- The two 8*4 multipliers operate Parallely **reducing critical path delay.**

- But still the **dependency on carry** exists Between π0 blocks.



architecture of an exact 8-bit multiplier using two exact 8 × 4 multipliers.

International Institute of Information Technology Bangalore

Approximate Partial Product(PP) units

- **Carry Disregard PP Unit (π1):** 1 AND gate for 1bit mul & 1 XOR gate for determining the sum of Sin. No **Cin** and **Cout.**

- **Half-adder-Based PP Unit (π2):** 1AND gate, 1 HA for finding the sum of Sin with Λ output. It has **Cout** but disregards **Cin**.

- **Full-adder-Based PP Unit (π3):** 2 AND gates 1 FA for finding the sum of Sin with 2Λ output. It has **Cout** but disregards **Cin**. It is a combination of 2 **π0's** but reduces to only **1 Cout.**

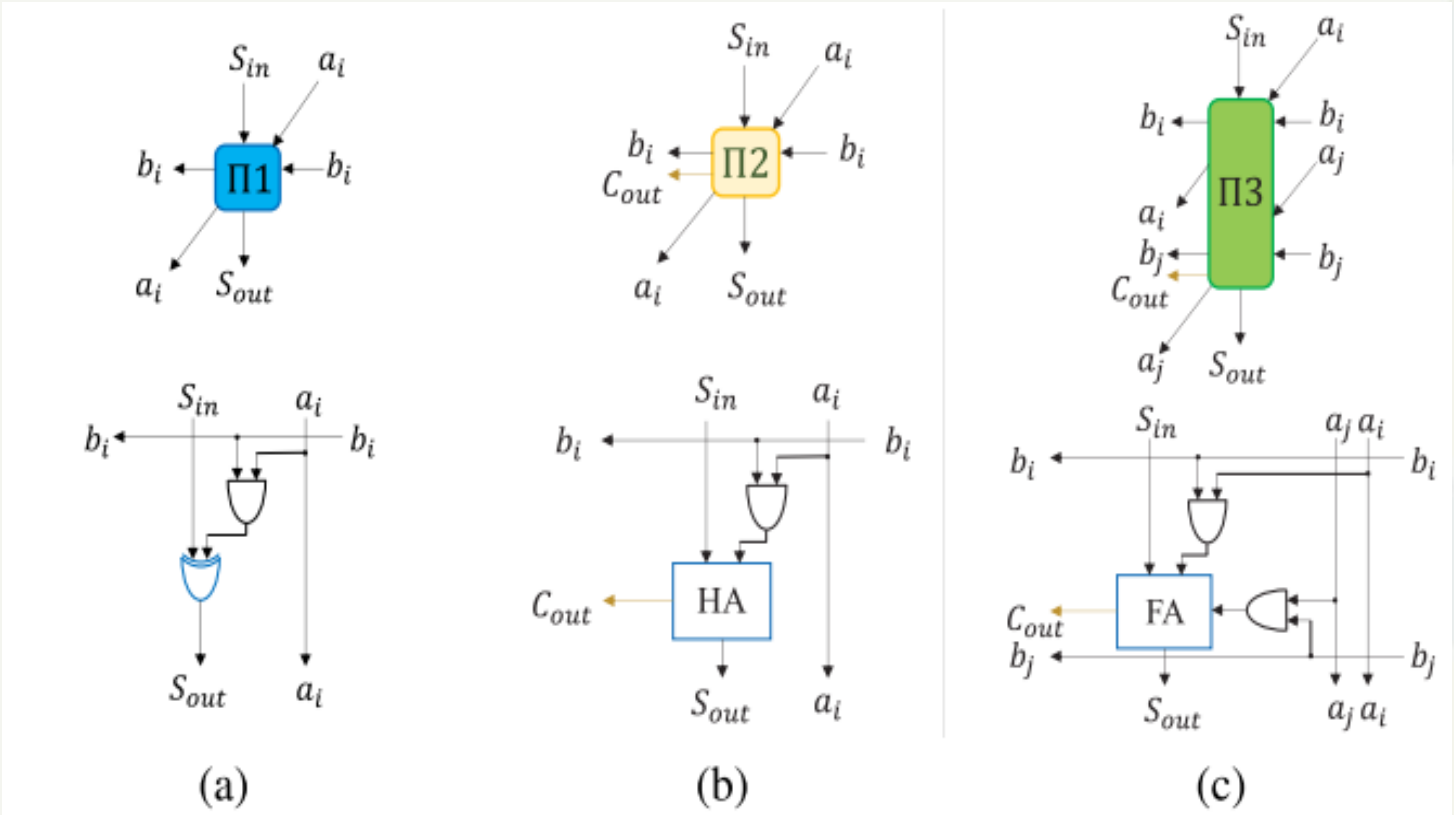


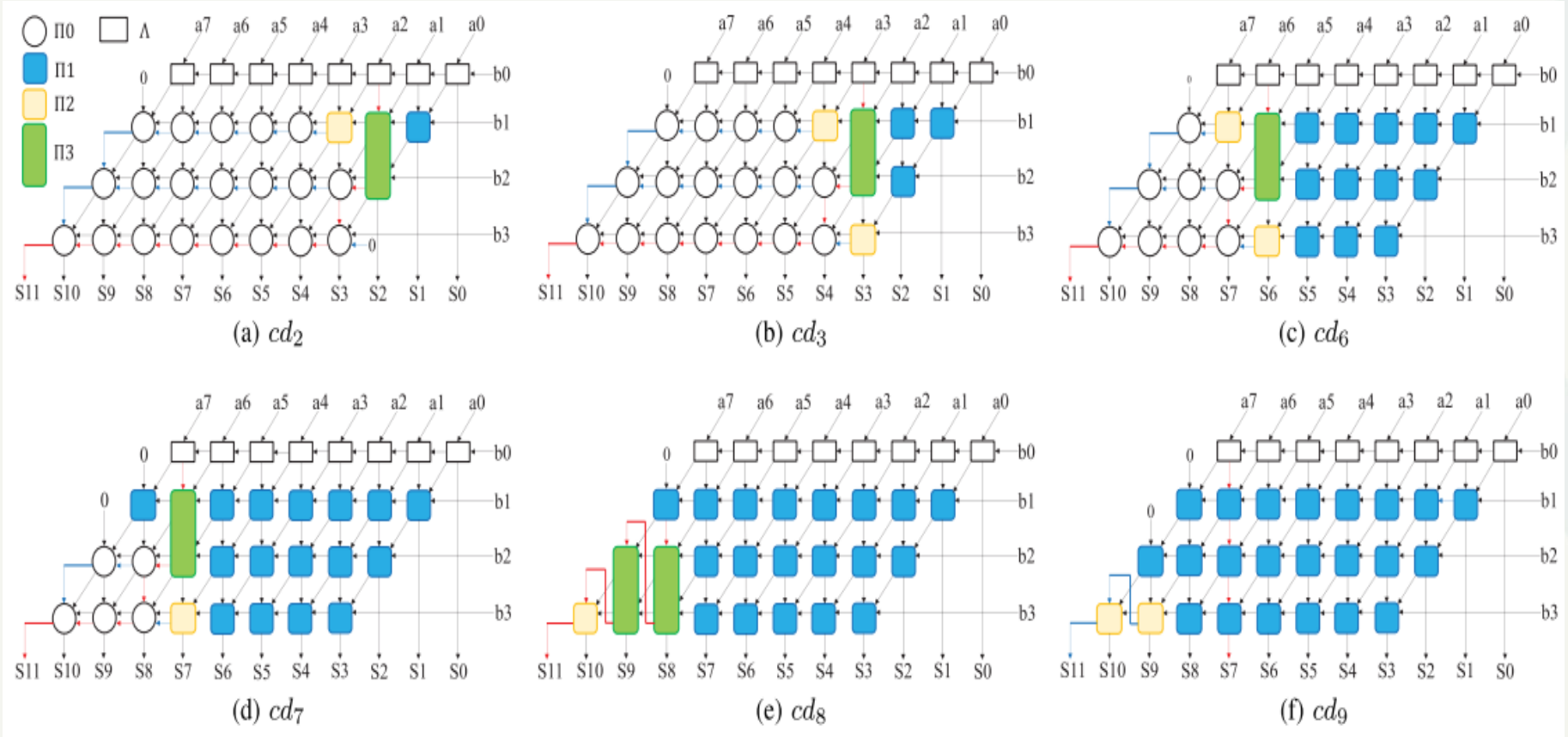
Fig. 3. Circuits and symbols of (a) Π1, (b) Π2, and (c) Π3.

International Institute of Information Technology Bangalore

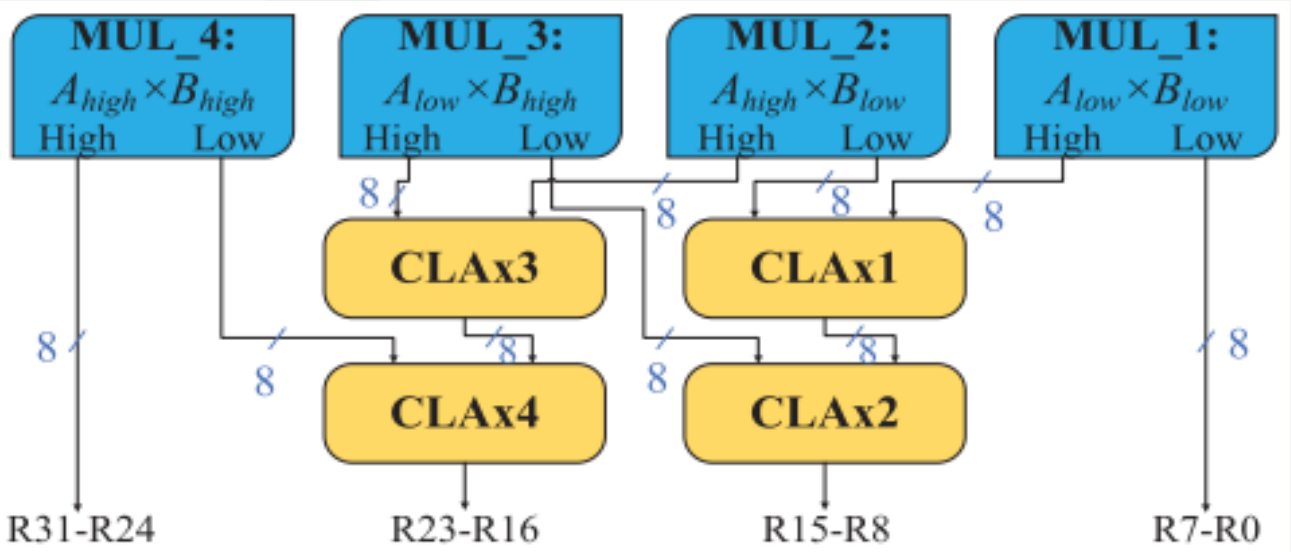
Proposed Approximate 8*4 Multipliers

•The **cd3** disregards all the carries up to Column 3, so all the elements up to this column are **π**1 type. In Column 4, there is an **π3** and an **π2**. Column 5 also contains an **π2**, so other elements of this multiplier are **π0s.**

•The other cd2,cd6,cd7 Cd8,cd9 follows similar Procedure.



Proposed Approximate 16 bit Multiplier



Proposed approximate 16-bit multiplier architecture.

PROPOSED APPROXIMATE CDM16S

Design	Mul_1: $A_{low}B_{low}$	Mul_2: $A_{high}B_{low}$	Mul_3: $A_{low}B_{high}$	Mul_4: $A_{high}B_{high}$
CDM16_0000	*Exact*	*Exact*	*Exact*	*Exact*
CDM16_3000	*CDM8_40*	*Exact*	*Exact*	*Exact*
CDM16_7000	*CDM8_84*	*Exact*	*Exact*	*Exact*
CDM16_8000	*CDM8_95*	*Exact*	*Exact*	*Exact*
CDM16_8330	*CDM8_95*	*CDM8_40*	*CDM8_40*	*Exact*
CDM16_b330	*CDM8_a8*	*CDM8_40*	*CDM8_40*	*Exact*
CDM16_f770	*CDM8_aa*	*CDM8_84*	*CDM8_84*	*Exact*
CDM16_f880	*CDM8_aa*	*CDM8_95*	*CDM8_95*	*Exact*
CDM16_f883	*CDM8_aa*	*CDM8_95*	*CDM8_95*	*CDM8_40*
CDM16_fbb3	*CDM8_aa*	*CDM8_a8*	*CDM8_a8*	*CDM8_40*
CDM16_fff7	*CDM8_aa*	*CDM8_aa*	*CDM8_aa*	*CDM8_84*
CDM16_fff8	*CDM8_aa*	*CDM8_aa*	*CDM8_aa*	*CDM8_95*
CDM16_fffb	*CDM8_aa*	*CDM8_aa*	*CDM8_aa*	*CDM8_a8*
CDM16_ffff	*CDM8_aa*	*CDM8_aa*	*CDM8_aa*	*CDM8_aa*

ACCURACY CRITERIA

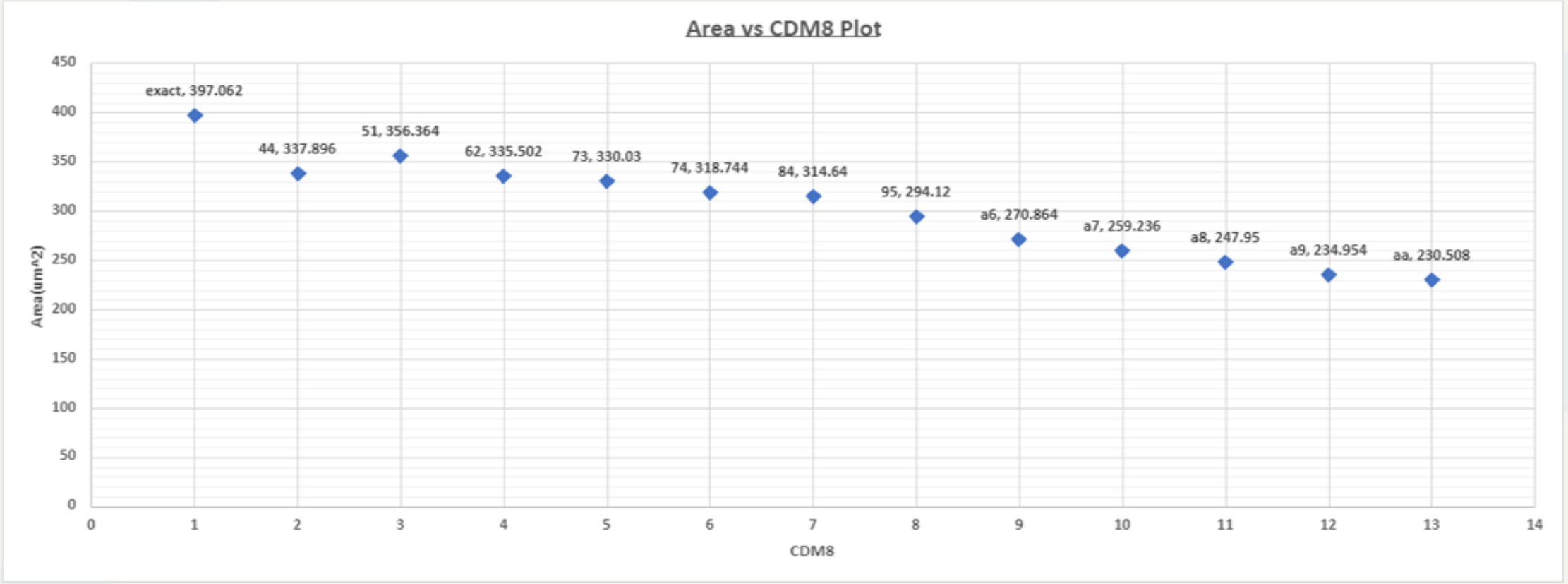
Accuracy Criteria	Equation	Description
Error (E_i)	$E_i = \Omega_i^P - \Omega_i^X$	Refers to the difference between the exact multiplier's output and the approximate multiplier's output [28].
Error Distance (ED_i)	$ED_i = \lvert E_i \rvert = \lvert \Omega_i^P - \Omega_i^X \rvert$	Represents the absolute value of E_i [19], [20], [28], [29].
Mean Error Distance (MED)	$\frac{1}{2^{2N}} \sum_{i=1}^{2^{2N}} ED_i$	Refers to the average of ED_i for all possible input combinations [19], [29].
Relative Error Distance (RED_i)	$ED_i/\Omega_i^P \quad \forall \Omega_i^P \neq 0$	Refers to the ratio of ED_i to the output of the corresponding exact multiplier that is not equal to zero [20], [28], [29].
Mean Relative Error Distance (MRED)	$\frac{1}{2^{2N}} \sum_{i=1}^{2^{2N}} RED_i$	Refers to the average of RED_i for all possible input combinations [19], [20], [28], [29].
Normalized Mean Error Distance (NMED)	$MED/(2^N - 1)^2$	Refers to the average of ED_i divided by the most considerable exact multiplier output, i.e., $(2^N - 1)^2$ [19], [20], [28], [29].
Probability of Correctness (PC)	$\#\Omega_C/2^{2N}$	Refers to the ratio of the number of correct outputs of the approximate multiplier to the total number of possible outputs. $\#\Omega_C$ is the number of correct outputs of the approximate multiplier [20], [28].
Number of Effective Bits (NoEB)	$2N - \log_2(1 + \sqrt{MSE})$	MSE is the mean squared error obtained as the average of ED_i^2 for all possible input combinations, i.e., $MSE = \frac{1}{2^{2N}} \sum_{i=1}^{2^{2N}} ED_i^2$ [20], [28].

RESULTS For 8 bit Multipliers

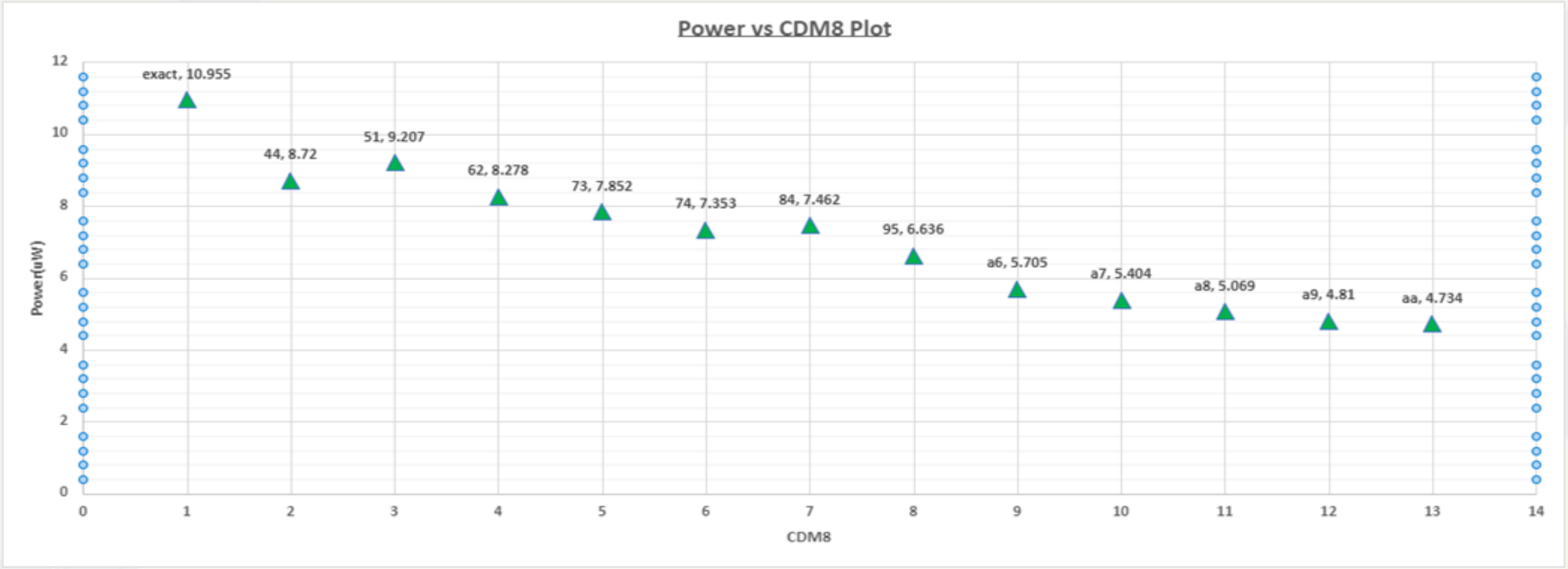
- We used **Verilog HDL** to describe the proposed approximate multipliers and **Genus tool** was used to **synthesize** the proposed designs with **45-nm technology** to get critical path delay, power consumption, and area.

	Hardware							Accuracy				
Proposed Method	Area(um2)	Power(uW)	Delay(nS)	PDP	PADP	PDEP	PADEP	MED	NMED	MRED	NoEB	PC(%)
Exact8	397.062	10.955	0.697	7.635635	3031.821	0	0	0	0	0	16	100
CDM8_44	337.896	8.72	0.565	4.9268	1664.746	0.065034	21.97465	97.75	0.0015	0.0132	8.4928	52.91
CDM8_51	356.364	9.207	0.552	5.082264	1811.136	**0.019821**	**7.06343**	14.25	0.00022	**0.0039**	11.2896	**59.57**
CDM8_62	335.502	8.278	0.5	4.139	1388.643	0.033526	11.24801	35.25	0.00054	0.0081	10.1011	51.1
CDM8_73	330.03	7.852	0.463	3.635476	1199.816	0.058895	19.43702	89.25	0.00137	0.0162	8.8668	42.52
CDM8_74	318.744	7.353	0.454	3.338262	1064.051	0.079784	25.43082	157.25	0.00242	0.0239	8.0797	37.09
CDM8_84	314.64	7.462	0.447	3.335514	1049.486	0.105736	33.26871	225.25	0.00346	0.0317	7.6338	33.58
CDM8_95	294.12	6.636	**0.404**	2.680944	788.5192	0.137801	40.52989	441.25	0.00679	0.0514	6.6605	29.05
CDM8_a6	270.864	5.705	0.475	2.709875	734.0076	0.205951	55.78458	777.25	0.01195	0.076	5.8071	26.4
CDM8_a7	259.236	5.404	0.475	2.5669	665.4329	0.276198	71.60058	1321.25	0.02032	0.1076	5.0014	24.64
CDM8_a8	247.95	5.069	0.47	2.38243	590.7235	0.365941	90.73513	2409.25	0.03705	0.1536	4.0895	23.06
CDM8_a9	234.954	4.81	0.466	2.24146	526.64	0.430584	101.1675	3689.25	0.05674	0.1921	3.3652	22.43
CDM8_aa	**230.508**	**4.734**	0.461	**2.182374**	**503.0547**	0.464627	107.1003	4713.25	0.07248	**0.2129**	2.8556	**22.26**

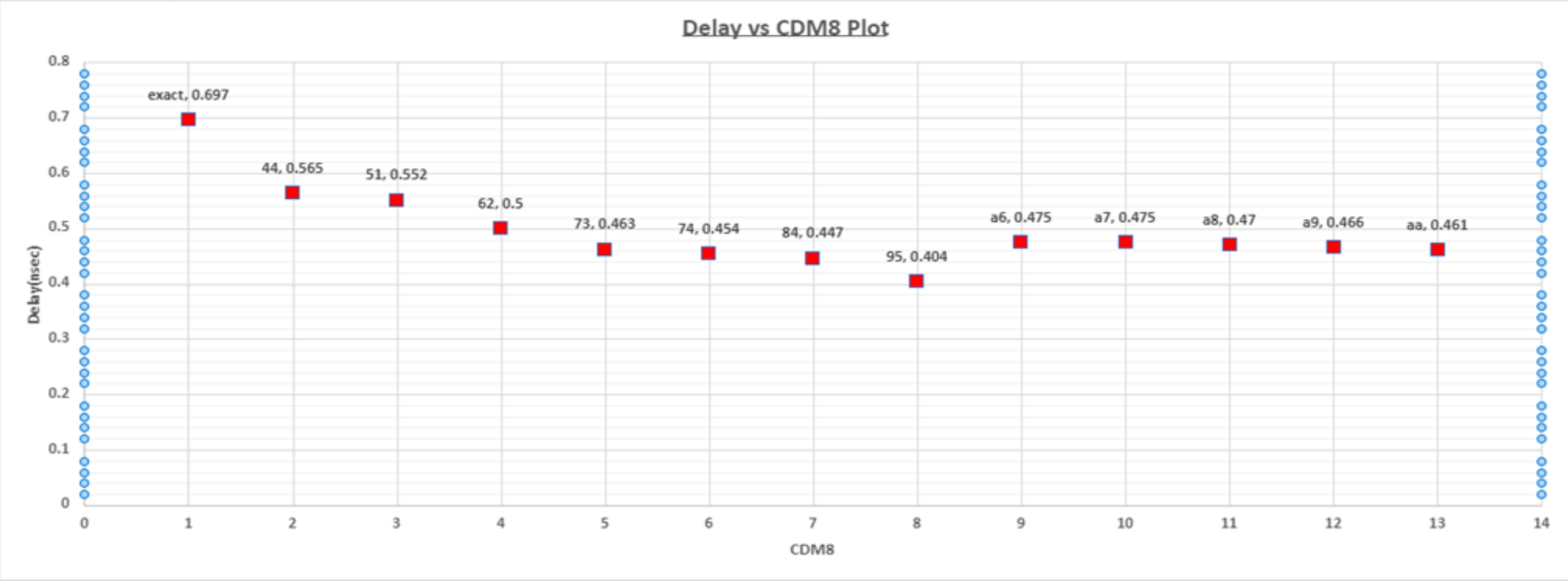
Area Graph


Area vs CDM8 Plot
Area (um^2)
CDM8
exact, 397.062
44, 337.896
51, 356.364
62, 335.502
73, 330.03
74, 318.744
84, 314.64
95, 294.12
a6, 270.864
a7, 259.236
a8, 247.95
a9, 234.954
aa, 230.508
450
400
350
300
250
200
150
100
50
0
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14

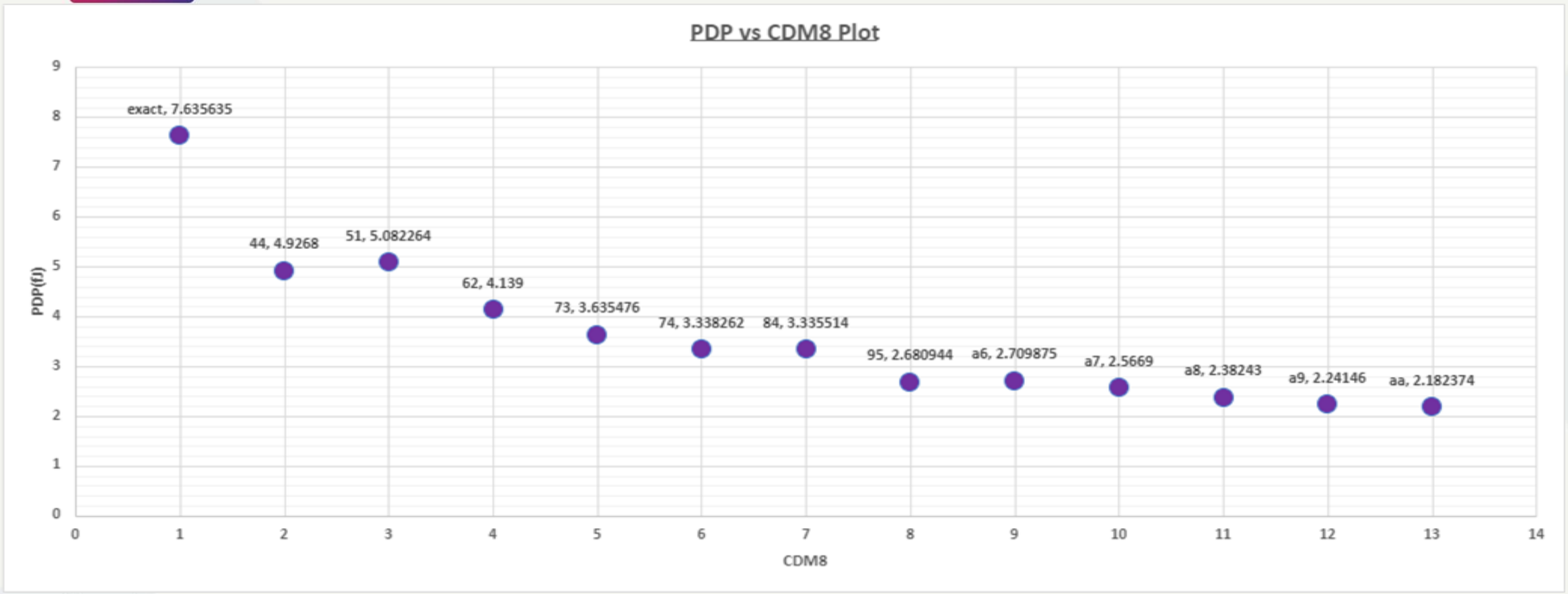
Power Graph


Power vs CDM8 Plot
Power(uW)
CDM8
exact, 10.955
44, 8.72
51, 9.207
62, 8.278
73, 7.852
74, 7.353
84, 7.462
95, 6.636
a6, 5.705
a7, 5.404
a8, 5.069
a9, 4.81
aa, 4.734
0
2
4
6
8
10
12
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14

Delay Graph


Delay vs CDM8 Plot
Delay(nsec)
CDM8
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
exact, 0.697
44, 0.565
51, 0.552
62, 0.5
73, 0.463
74, 0.454
84, 0.447
95, 0.404
a6, 0.475
a7, 0.475
a8, 0.47
a9, 0.466
aa, 0.461

PDP Graph


PDP vs CDM8 Plot
PDP(fJ)
CDM8
exact, 7.635635
44, 4.9268
51, 5.082264
62, 4.139
73, 3.635476
74, 3.338262
84, 3.335514
95, 2.680944
a6, 2.709875
a7, 2.5669
a8, 2.38243
a9, 2.24146
aa, 2.182374
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14

RESULTS For 16 bit Multipliers

	Hardware							Accuracy				
Proposed Method	Area(um2)	Power(uW)	Delay(nS)	PDP	PADP	PDEP	PADEP	MED	NMED	MRED	NoEB	PC(%)
Exact16	1709.658	92.43	1.61	148.8123	254418.1	0	0	0	0	0	32	100
CDM16_0000	1703.16	88.7	1.189	105.4643	179622.6	1.033339	1759.942	8221326	0.001914	**0.009798**	8.48	**19.6**
CDM16_3000	1716.156	82.58	1.215	100.3347	172190	0.983079	1687.118	8221327	0.001914	**0.009798**	8.48	13.85
CDM16_7000	1636.812	76.68	1.193	91.47924	149734.3	0.896405	1467.247	8221330	0.001914	0.009799	8.48	8.19
CDM16_8000	1620.738	75.62	1.192	90.13904	146091.8	0.883363	1431.699	8221324	0.001914	0.0098	8.48	7.54
CDM16_8330	1610.21	69.01	1.073	74.04773	119232.4	0.726334	1169.551	8221500	0.001914	0.009809	8.48	5.1
CDM16_b330	1602.27	61.22	0.958	58.64876	93971.15	0.575344	921.857	8221508	0.001914	0.00981	8.48	4.44
CDM16_f770	1430.244	50.72	0.958	48.58976	69495.21	0.484148	692.4503	8220003	0.001914	0.009964	8.48	1.98
CDM16_f880	1401.516	48.84	0.958	46.78872	65575.14	0.471256	660.4728	8220824	0.001914	0.010072	8.48	1.73
CDM16_f883	1415.538	45.71	0.716	32.72836	46328.24	0.359226	508.4987	8607336	0.002004	0.010976	8.45	1.55
CDM16_fbb3	1321.83	41.24	0.724	29.85776	39466.88	**0.34217**	**452.2905**	8613693	0.002006	0.01146	8.45	1.24
CDM16_fff7	1212.39	36.83	**0.721**	26.55443	32194.33	0.692327	839.3705	22980961	0.005351	0.026072	7.17	0.99
CDM16_fff8	1194.606	36.16	**0.721**	26.07136	31145	0.930148	1111.16	37133784	0.008646	0.035677	6.42	0.96
CDM16_fffb	1147.41	34.62	**0.721**	24.96102	28640.52	2.152214	2469.472	1.66E+08	0.038672	0.086223	4.05	0.9
CDM16_ffff	**1128.6**	**34.26**	**0.721**	**24.70146**	**27878.07**	2.858329	3225.911	3.17E+08	0.073815	0.115715	2.84	0.9

APPLICATION: Image Processing

1)We assess the use of **Image blurring** in Image Processing using **Gaussian Smoothing.**

2)The 3*3 kernel used for gaussian smoothing with $\sigma=1.5$ is $\begin{pmatrix} 97 & 121 & 97 \\ 121 & 151 & 121 \\ 97 & 121 & 97 \end{pmatrix}$

Performance Metrics:

1) **Peak Signal-to-Noise Ratio (PSNR):** PSNR measures the quality of an image by evaluating the mean squared error between the original and reconstructed images.

1) **Structural Similarity Index Measure (SSIM):** SSIM quantifies the similarity between two images based on luminance, contrast, and structure, reflecting human perception of image quality.

APPLICATION: Image Processing

	SSIM = 99.97%	SSIM = 99.62%	SSIM = 99.93%
Original image	PSNR= 63.53db	PSNR= 44.85db	PSNR= 59.1db
			
Exact8	**cdm8_40**	**cdm8_44**	**cdm8_51**

SSIM = 99.83%	SSIM = 99.77%	SSIM = 99.56%	SSIM = 99.55%
PSNR= 54.39db	PSNR= 48.6db	PSNR= 43.1db	PSNR= 41.5db
			
cdm8_62	**cdm8_73**	**cdm8_74**	**cdm8_84**

APPLICATION: Image Processing

SSIM = 98.66 %

PSNR= 36.04db

cdm8_95

SSIM = 95.58%

PSNR= 48.6db

cdm8_a6

SSIM = 88.03%

PSNR= 24.04db

cdm8_a7

SSIM = 74.56%

PSNR= 18.3db

cdm8_a8

SSIM = 68.93%

PSNR= 16.2db

cdm8_a9

SSIM = 68.93%

PSNR= 16.2db

cdm8_aa

APPLICATION: Image Processing

Design	SSIM (%)	PSNR (db)
cdm8_40	99.97	63.53
cdm8_44	99.63	44.86
cdm8_51	99.93	59.17
cdm8_62	99.84	54.39
cdm8_73	99.78	48.60
cdm8_74	99.57	43.14
cdm8_84	99.55	41.60
cdm8_95	98.66	36.05
cdm8_a6	95.59	30.13
cdm8_a7	88.04	24.04
cdm8_a8	74.56	18.37
cdm8_a9	68.93	16.24
cdm8_aa	68.93	16.24

Future Scope

- we plan to use **compressor** based multi-operand approximate adders in the process of PP reduction.

- For the final summation step in the process of multiplication, there are various types of adders, such as **CSA** and **Parallel Prefix Adder (PPA),** can be done for future work.

- Different applications have different error tolerances, so **dynamically adjusting the accuracy level** of the proposed multipliers in various applications is one of our future works

RESULTS SCREENSHOTS & Code

IMAGE PROCESSING VERILOG SIMULATION

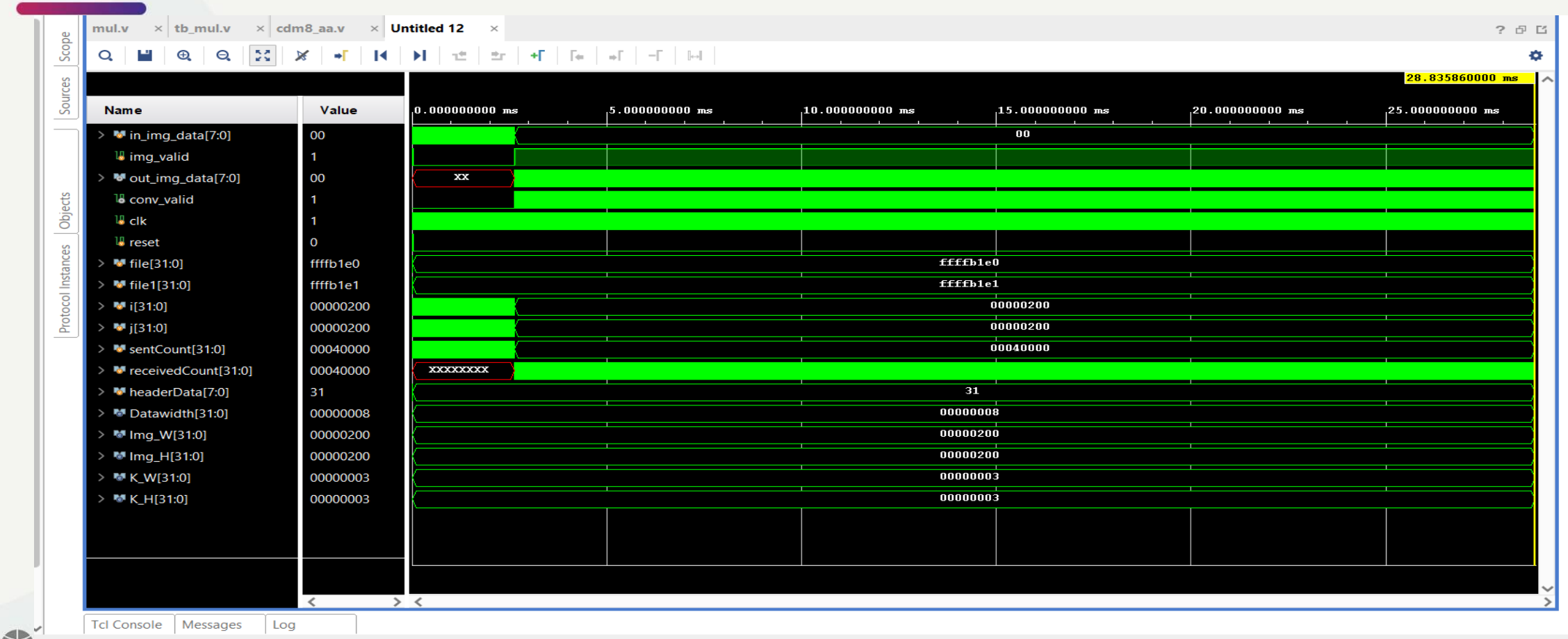


IMAGE PROCESSING – Python code for PSNR & SSIM

In [4]:

```
import cv2
from skimage.metrics import structural_similarity as ssim
from skimage.metrics import peak_signal_noise_ratio as psnr

# Load the original and blurred images
original_image = cv2.imread("original.bmp", cv2.IMREAD_GRAYSCALE)
blurred_image = cv2.imread("gs.bmp", cv2.IMREAD_GRAYSCALE)

# Calculate the Structural Similarity Index (SSI)
ssi_index, _ = ssim(original_image, blurred_image, full=True)

print("Structural Similarity Index (SSI):", ssi_index)

# Calculate the PSNR
psnr_value = psnr(original_image, blurred_image)

print("Peak Signal-to-Noise Ratio (PSNR):", psnr_value)
```

```
Structural Similarity Index (SSI): 0.8561510157043957
Peak Signal-to-Noise Ratio (PSNR): 24.863380714297026
```

GITHUB

https://github.com/Akash-Perla/Carry-Disregard-Approximate-Multipliers

REFERENCES

1. W. Liu, F. Lombardi, and M. Shulte, “A retrospective and prospective view of approximate computing point of view,” Proc. IEEE, vol. 108, no. 3, pp. 394–399, Mar. 2020.
2. P. Schober, S. N. Estiri, S. Aygun, N. TaheriNejad, and M. H. Najafi, “Sound source localization using stochastic computing,” in Proc. IEEE/ACM Int. Conf. Comput. Aided Design (ICCAD), Oct. 2022, pp. 1–9.
3. N. TaheriNejad and S. Shakibhamedan, “Energy-aware adaptive approximate computing for deep learning applications,” in Proc. IEEE Comput. Soc. Annu. Symp. VLSI (ISVLSI), Jul. 2022, p. 328.
4. P. Schober, S. N. Estiri, S. Aygun, A. H. Jalilvand, M. H. Najafi, and N. TaheriNejad, “Stochastic computing design and implementation of a sound source localization system,” IEEE J. Emerg. Sel. Topics Circuits Syst., vol. 13, no. 1, pp. 295–311, Mar. 2023.

International Institute of Information Technology Bangalore

26/C, Electronics City, Hosur Road,

Bengaluru – 560 100, Karnataka, India

www.iiitb.ac.in

https://www.facebook.com/IIITBofficial/

https://www.linkedin.com/school/iiit-bangalore/

https://www.instagram.com/iiitb_official/

https://twitter.com/IIITB_official

https://www.youtube.com/user/iiitbmedia