Engineering and Technology Journal
Journal homepage: https://etj.uotechnology.edu.iq

Hardware Resources Minimization for Implementing Bell Membership Function on FPGA

Najmah A. Habeeb[a]*, **Saad M. Abbas**[b]

[a] Electrical Engineering Dept., University of Technology-Iraq, Alsina'a street, 10066 Baghdad, Iraq.
[b]Medical Instrumentation Dept., Techniques Engineering, Al-Mustaqbal University, Babil, Iraq.
*Corresponding author Email: najmah.a.habeeb@uotechnology.edu.iq

HIGHLIGHTS

- The importance of optimizing Bell membership function designs for FPGA implementations was highligted.
- Seven approaches are proposed to reduce the hardware required for Bell function implementation on FPGA devices.
- Resource use, performance, and power are analyzed for Bell function designs on a Spartan-3A DSP FPGA kit.
- The polynomial curve fitting method is segmented to lower equation order and reduce hardware needed for Bell function.

ARTICLE INFO

Handling editor: Aws Al-Taie

Keywords:
ANFIS
Bell-shaped fuzzy set
Membership Function
Hardware resources minimization
FPGA

ABSTRACT

The complexity of hardware realization for the Bell function forces most researchers to replace it with trapezoidal, negatively affecting accuracy. The implementation complexity of the Bell function on FPGA comes from bundling the division operator by FPGA. This requires large hardware resources due to the limited number of logic elements in FPGA devices, such as Lock-Up Tables (LUTs), slices, and DSP48 units, specifically when utilizing the Spartan-3A DSP sc3sd3400a Xilinx FPGA kit. This work investigates hardware resource minimization of Bell Membership Functions (MFs) based on FPGA implementation. A comparative analysis of seven proposed designs with a predesigned system is studied, focusing on resource utilization and performance evaluation. Approach 1 uses the highest number of DSP48 blocks (10). Conversely, Approach 7 achieves the highest area minimization, using the least number of slices (79) and LUTs (144) with zero DSP48 blocks with a reduction rate equal to (97.7%) for slices and (96.7%) LUTs regarding previous work. The absence of multiplication blocks for implementing Approach 7 comes from modifying the precomputation method using half memory size. Performance evaluation indicates varying error rates of 1.06%, 1.14%, and 1.16% for Approaches 1, 2, and 3, respectively. Approaches 6 and 7 exhibit the lowest error rates (0.17%), suggesting superior accuracy, including the previous work. To sum up, the modification in Approach 7 is a very useful method that reduces the hardware platform area and enhances the performance. Thus, this approach is adopted as the best method for designing the Bell function by FPGA.

1. Introduction

In the contemporary landscape of intelligent systems, converging neural networks and fuzzy logic into an Adaptive Neuro-Fuzzy Inference System (ANFIS) has proven to be a groundbreaking approach for modeling and controlling complex, nonlinear systems. ANFIS benefits from the learning capabilities of neural networks and the interpretability of fuzzy logic, making it particularly suitable for applications requiring precision and adaptability [1]. The ANFIS model consists of five basic layers: Membership Layer, Fuzzification Layer, Normalization Layer, Defuzzification Layer, and Output Layer [2]. The Membership Layer contains many membership functions that return the membership degree of how a crisp value is mapped to an input space called the universe of discourse. The performance of ANFIS depends on the selection of the number and shape of membership functions, as these two factors influence the computational complexity and accuracy of the designed ANFIS-based model [3].

The most commonly shaped memberships are Triangular, Trapezoidal, Gaussian, and Bell [4]. Hence, Researchers have widely used these shapes of membership functions for the ANFIS-based model applications [5]. Using the Gaussian shape to design the ANFIS model that monitors the healthcare medicinal Internet of things. In Naderkhani et al., work [6], the ANFIS is employed to analyze and predict a nonparametric fuzzy regression function with non-fuzzy inputs and symmetric trapezoidal fuzzy outputs. The Triangular membership function is used in many ANFIS applications such as the FPGA-based ANFIS model to linearize the natural non-linear characteristics of NTC thermistors [7], the ANFIS model based on FPGA for the fitness

http://doi.org/10.30684/etj.2024.151873.1786
Received 15 June 2024; Received in revised form 20 September 2024; Accepted 30 September 2024; Available online 17 November 2024
2412-0758/University of Technology-Iraq, Baghdad, Iraq

function approximation method to reduce the execution latency of complex fitness functions [8], FPGA-based a neuro-fuzzy logic controller used to exhibit good control for tracking the angle of the limb's movement and provide an accurate amount of charge according to the desired reference angle [9,10], which implement the adaptive neuro-fuzzy model by FPGA to enhance PID performances within VCA for grid-connected wind system under nonlinear behaviors. Whereas the Bell-shape function employs different ANFIS applications such as an FPGA-based intelligent neuro-fuzzy sensor for driving style recognition [11], designing a classifier to detect the seizure signals from non-seizure signals automatically depending on the optimized adaptive neuro-fuzzy inference system based on FPGA [12], and using the FPGA-based ANFIS and Multi-Layer Perceptron (MLP) to build the ink drop spread instead of the active learning method which imposes high memory requirements and computational costs [13]. In Karataş et al., work [14], the triangular, trapezoidal, gaussian, and generalized Bell-shaped membership function units using FPGA for real-time Fuzzy Logic applications have been designed for further comparison. This research will be adopted for comparison purposes in this work because it builds the membership functions alone only.

Based on the previous literature survey, most of these works adopt the Triangular membership function because this function is easily constructed in hardware using the FPGA and does not need a large area from the hardware platform. The importance of the platform area is because each FPGA device contains a limited number of logic elements, such as Look-Up Tables (LUTs), slices, and DSP48 units, which makes it impossible to design any function that consumes a large number of these elements [15-17]. Additionally, due to the complexity of realizing the Bell function in hardware, most researchers have replaced it with Trapezoidal, as in [10]. Thus, the performance of the ANFIS will be inaccurate. The complexity of implementing the Bell function on FPGA is due to this function requiring bundling the division operator by FPGA, which is more difficult.

Based on all the mentioned drawbacks, optimization approaches are important to minimize the hardware resources required to implement the ANFIS membership functions on the FPGA platform. Therefore, this work proposes a different methodology to build the Bell membership function by the FPGA with high-performance and low device resources.

The rest of the paper is structured as follows: The subsequent section briefly discusses the Bell membership function. Section 3 explains the proposed approaches to designing and optimizing the Bell function based on FPGA, followed by experimental results and a discussion Section 4. Section 5 concludes this study.

2. Generalized Bell-Shaped Membership Function

Bell-shaped membership functions are also quite common when defining fuzzy numbers. A generic graph involving three parameters is shown in Figure 1 a, b , and c (a –responsible for its width, c –responsible for its center, and b –responsible for its slopes). This function is presented by the formula [18]:

$$f(x; a, b, c) = \frac{1}{1 + \left|\frac{x - c}{a}\right|^{2b}} \tag{1}$$

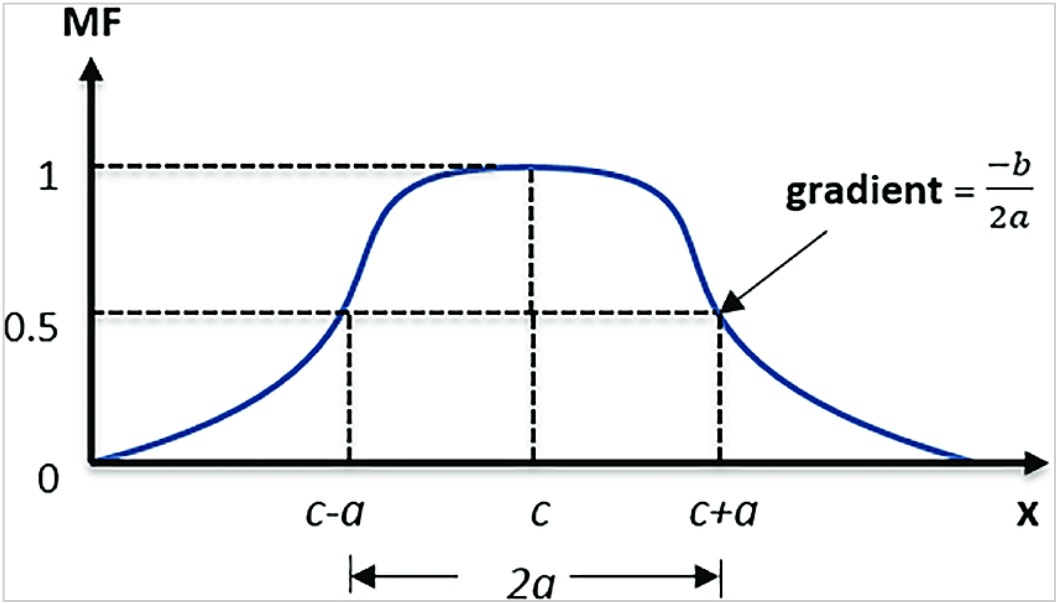

Figure 1: Bell-shaped fuzzy set [19]

3. Proposed design of generalized Bell membership function based on FPGA

Bell MF does not have a pre-designed block in the FPGA library, so utilizing some approaches to build this function is indispensable. These approaches are employed to optimize the implementation of the Bell MF in an FPGA device to minimize the number of slices and LUTs and reduce power dissipation with low latency. Hence, this section will present seven proposed Bell membership function designs with different optimization techniques and hardware implementation.

3.1 FPGA design of Bell MF based on hard FPGA blocks method (Approach 1)

The design of the Bell membership function in this method depends on the basic equation of Bell MF that is mentioned in Section 2. In this approach, hard FPGA blocks of the Xilinx System Generator are used to realize this equation in addition to the reciprocal function. According to Equation (1), there are three constants, which are ($a = 0.2, b = 2, and\ c = 0.5$,), and these values are chosen to correspond to the work desired.

Figure 2 shows the implementation of this function depending on Equation (1). Whereas the hardware realization of the term $\left(1 + \left|\frac{X-c}{a}\right|^{2b}\right)$ can be designed in many steps. Firstly, the input X will be subtracted from the $c = 0.5$ using the AddSub block to represent the term $(X - c)$ and then multiply the output with $\left(\frac{1}{a} = \frac{1}{0.2} = 5\right)$ using CMult1 block. Secondly, the Absolute block will take the absolute value to represent the term $\left(\left|\frac{X-c}{a}\right|\right)$. Thirdly, the power $(2b)$ is performed by the Mult and Mult1 blocks to define the terms $\left(\left|\frac{X-c}{a}\right|^{2}\right)$ and $\left(\left|\frac{X-c}{a}\right|^{2b}\right)$, respectively. Finally, the denominator will be represented by summing (1) with $\left(\left|\frac{X-c}{a}\right|^{2b}\right)$ using the AddSub1 block. Since the range of denominator value is relatively small (corresponding to the desired parameters) and can represented by a limited number of bits (10-bit), the reciprocal function based on the ROM approach is adopted in this design. As illustrated in Figure 2, the ROM block saves the precalculated magnitude of the reciprocal with 16-bit output resolution. The BitBasher1 block is used to slice the denominator bits to an appropriate bit number to be applied as an address to the memory.

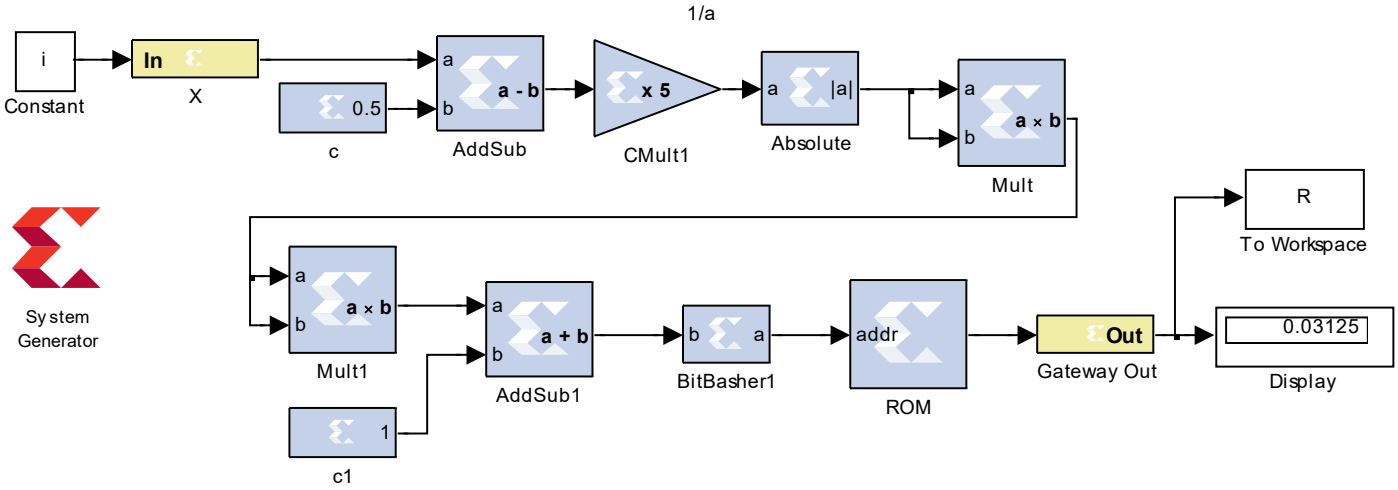

Figure 2: Bell MF based on hard blocks techniques

3.2 FPGA design of Bell MF based on polynomial curve fitting (Approach 2)

Polynomial curve fitting is one of the useful methods for converting any function to an equivalent designable equation. As shown in Figure 1, the Bell MF is summitry on the center, and the value of this function is unity in duration between (k) and $2(c - k)$, where k is the first point of x that produces the unity value for the Bell function at the left half of the function. The value of k can be determined according to function values and the slope. (b) in addition to the coordinate point when the function is approximately equal to zero (x_1, y_1) by $\left(k = \left(\frac{1-y_1}{b}\right) + x_1 = 0.45\right)$. Hence, this method is used to produce the equation of the Bell function for the duration between x_1 and k only. Therefore, the value of this function for all durations can be calculated according to the following Equation (2):

$$f(x;\sigma,c) = \begin{cases} f(x;\sigma,c) & if \quad x_1 \le x < k \\ 1 & if \quad k \le x < 2(c-k) \\ f(2c-x;\sigma,c) & if \quad x \ge 2(c-k) \end{cases} \tag{2}$$

The curve fitting equation is obtained using the MATLAB package and the curve fitting tool by taking different points as samples and orders to reach the typical representation for the desired duration of the bell membership function. Figure 3 shows the convergence of curve fitting approximation, denoted by the blue line, with the original MATLAB function of the bell, denoted by red circles. The Equation (3) can represent the desired duration of the bell.

$$f(x) = -170.9x^4 + 138.4x^3 - 27.36x^2 + 2.068x + 0.004226 \tag{3}$$

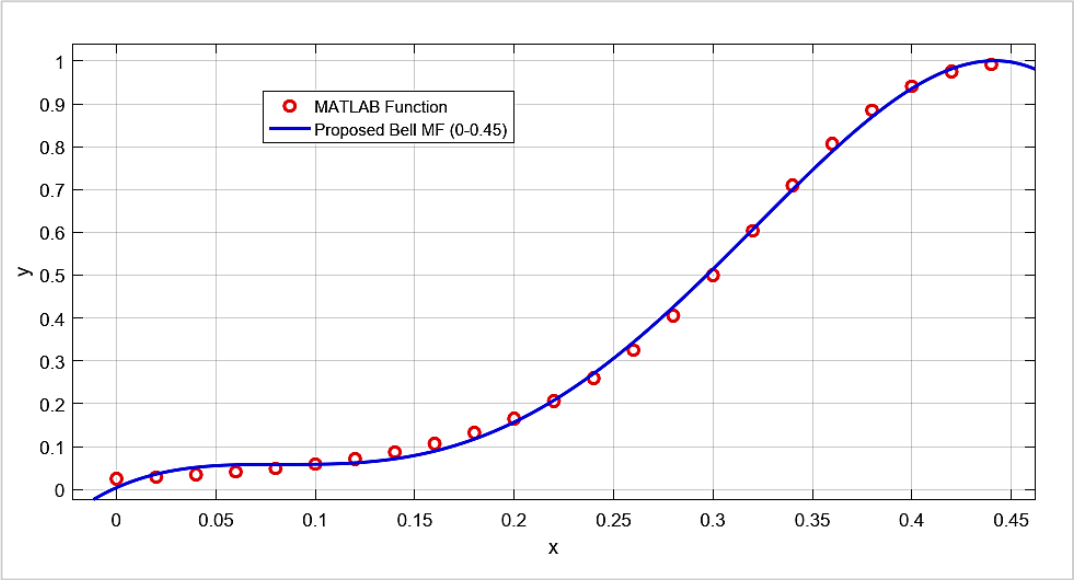

Figure 3: Polynomial Curve Fitting Plot for the desired duration of Bell MF

As shown in Figure 4, the multiplexer (Mux) detects the input in any two halves of the bell MF. The Relational block drives the multiplexer to select passing the input to be applied to the first half equation when the input (x) is less than c. Otherwise, the true signal will be produced from the Relational block to the multiplexer to choose the manipulation of the right half duration. This manipulation is required to prepare the input value to be able to apply in the polynomial equation by subtracting the x from double c which is represented by a shift left to perform $(2c - x)$.

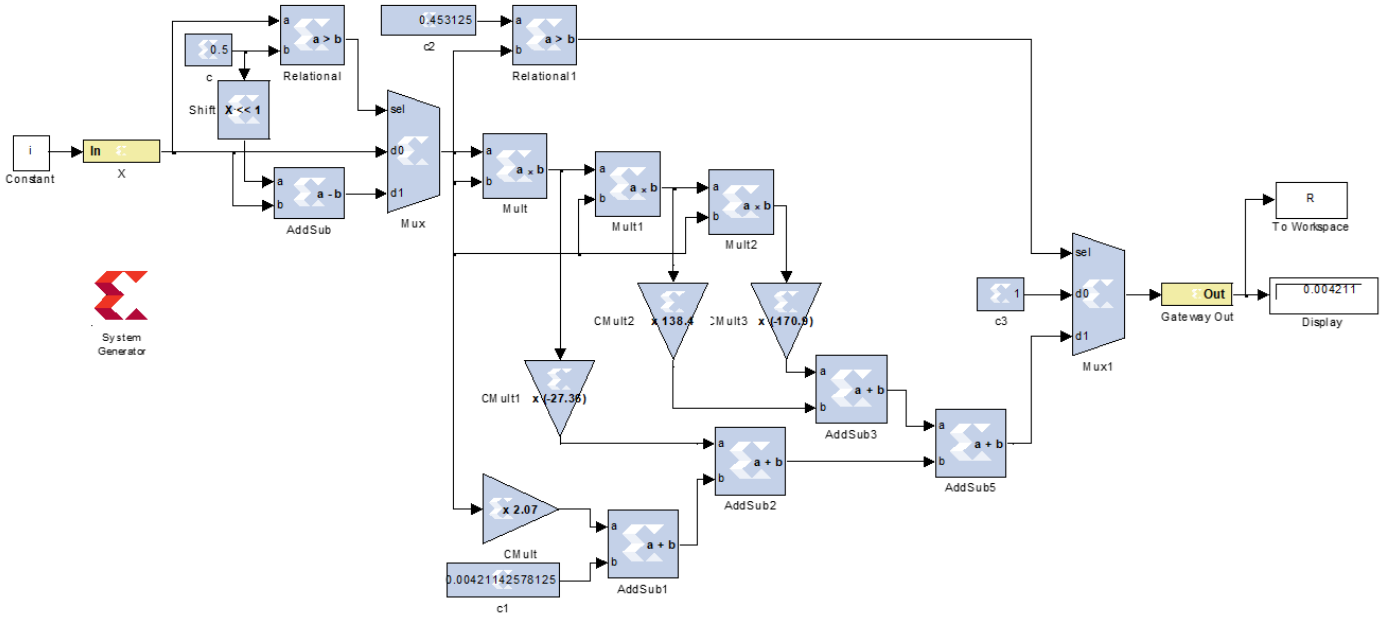

Figure 4: Bell MF based on Polynomial Curve Fitting Approach

After the input location e is achieved by Mux1 in the left or right half, Mux2 will be responsible for picking the unity part in the selected half. When the Relational2 block indicates the x is greater than k the Mux2 will select the unity value to represent the output of the Bell function. The first one is when x is equal to or smaller than (k). Else, the output of the polynomial equation will be chosen.

The polynomial equation consists of five terms of summation represented by (AddSub1 to AddSub4), as shown in Figure 4. In addition, four gain operations are defined by $CMult, CMult1, CMult2, and\ CMult3$ gain blocks to multiply the coefficient values $(2.07, -27.36, 138.4,$ and $-170.9)$ with ($x, x^2, x^3, and\ x^4$), respectively.

Furthermore, the Mult blocks are used to achieve the order value of x. Where Mult, Mult1, and Mult2 are used to produce $x^2, x^3, and\ x^4$, respectively. The final result of Equation (3) can be obtained at AddSub4, where the Display block produces the bell MF outcome.

3.3 FPGA design of gaussian MF based on shift-add approach (Approach 3)

In this approach, the same polynomial equation used in the previous design is adopted with a slight modification. This modification uses the shift-add operations instead of the gain operation to reduce the hardware resource consumption by the gain block (CMult). Figure 5 shows the suggested design based on the shift-add approach, represented by the Subsystem1 to Subsystem4 blocks.

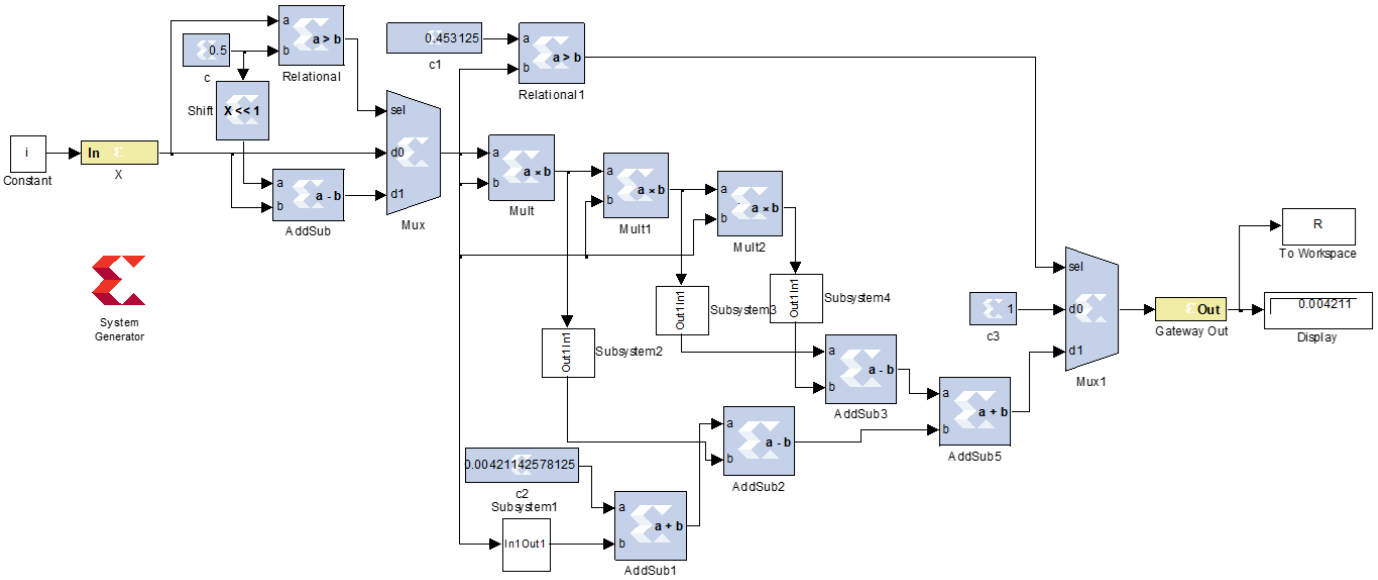

Figure 5: Bell MF based on Shift-Add Approach

Each block contains shift right and left blocks and AddSub components to represent the gain operation. The essential concept of this design is to convert the gain of the (x) by coefficients to a binary representation by shift-add operation. The number of shits blocks used to represent the gain operation is chosen carefully according to the value of the coefficient, where the number of right shift blocks depends on the integer magnitude of the coefficient, and the left shift blocks will be chosen corresponding to the coefficient fraction. For instance, Figure 6 illustrates the procedure of the Subsytem1 block for this technique that represents the $(2.068x)$ term. The one-time left shift operation represents the value of integer (2), whereas each right shift operation represents the active bit position after the binary point, which is in this case the right shift of $(00.0001000101)_2$ equal to $(0.068)_{10}$.

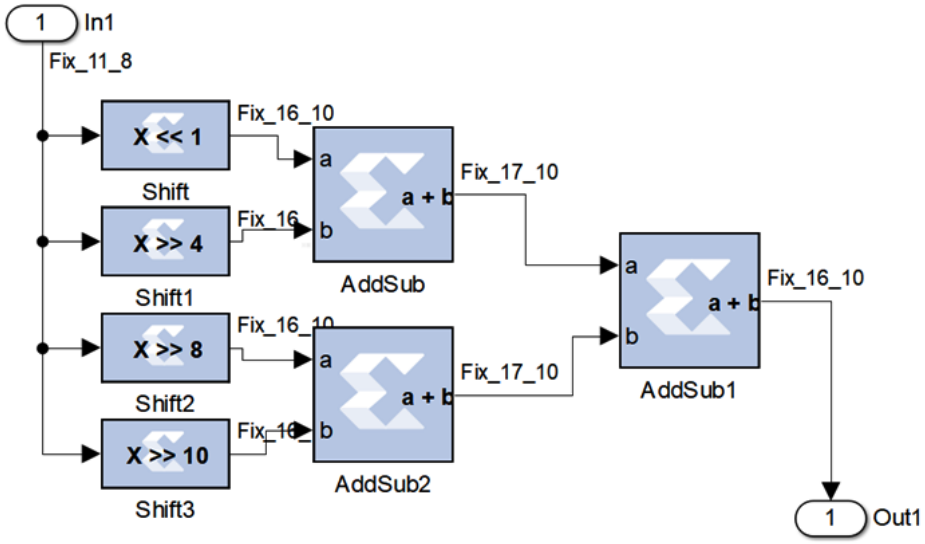

Figure 6: The Construction of Subsystem1 block that represents the Shift-Add operation

3.4 FPGA design of Bell MF based on decomposition of polynomial curve fitting technique (Approach 4)

The Bell function design in Approach 2 required a large area when implemented in the FPGA platform because it has three Mult blocks to construct the order of the polynomial equation. These blocks consume many DSP48 units. Hence, in this approach, the decomposition technique is used to reduce the order of the polynomial equation by dividing the desired duration of Bell MF

into two regions. The first region is (0) to (0.2), as shown in Figure 7. Red circles denote the curve fitting approximation, and the blue line denotes the original MATLAB function of the bell. The following equation can represent the first (region) part:

$$y_1 = 3.613x^2 - 0.0661x - 0.0288 \quad (4)$$

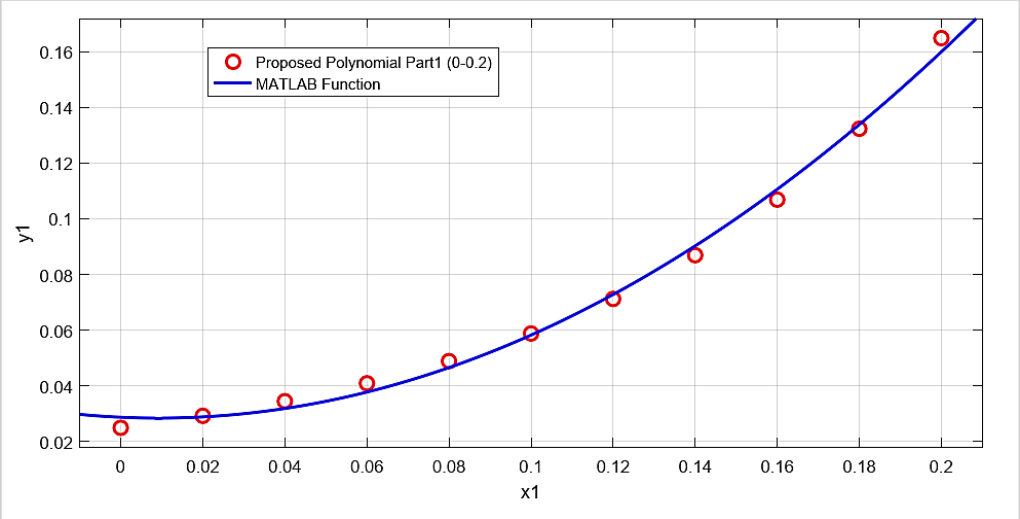

Figure 7: 1st Part of desired Bell MF duration

Finally, the second part of the Bell desired duration is from (0.2) to (0.45), as illustrated in Figure 8, and the second equation can be expressed as below:

$$y_2 = -113.4x^3 + 107x^2 - 28.61x + 2.5234 \quad (5)$$

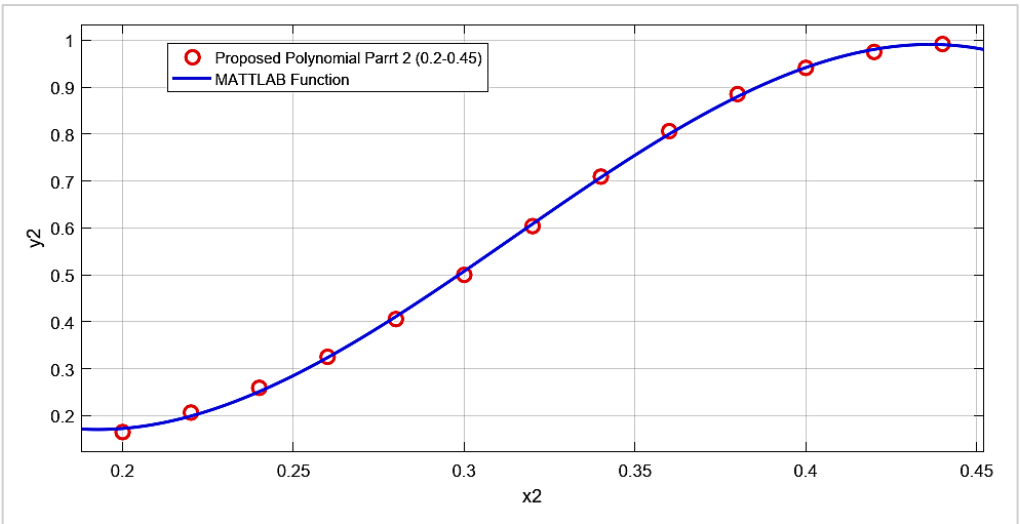

Figure 8: 2nd Part of desired Bell MF duration

Figure 9 shows the hardware implementation of the two equations for the proposed design. The essential idea behind dividing the desired duration of the Bell function into two parts is to reduce the number of multipliers from 3, as in the previous design, to only two.

The hardware implementation of the first part is represented by multiplying the input (x)(which is the output of the Mux block) by the coefficient (-0.0661) using the CMult3 and adding to the coefficient (0.0288) using the constant (c2) block. The output of the AddSub4 block is added with the term $(3.613x^2)$, which is represented by the CMult4 block. All these will construct Equation (4). Whereas the second part of this suggested design can be illustrated in Figure 9, the output of the Mult block is used to form x^2 which is multiplied by the coefficient (107) through the CMult block to perform the second term $(107x^2)$. The first term $(-113.4x^3)$ of the Equation (5) is achieved by the Mult1 and CMult1 blocks.

The selection of the two parts depends on the Relational1 block, which has two conditions. The first condition is choosing the first part when the input x is equal to or smaller than (0.2); therefore, the multiplexer (Mux1) will select the (d1) port. Whereas the second part can be selected when the input x is greater than (0.2), the multiplexer (Mux1) will choose the (d0).

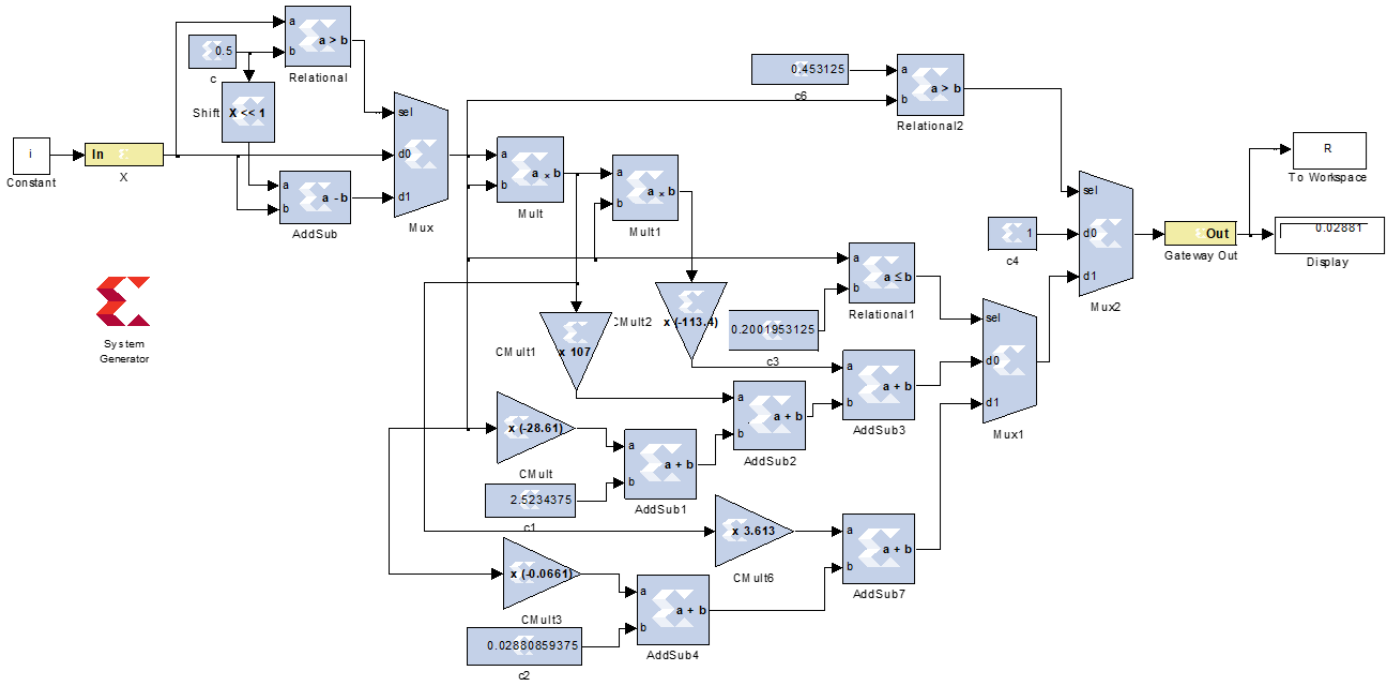

Figure 9: Bell MF based on decomposition technique

3.5 FPGA design of Bell MF based on decomposition and shift-add techniques (Approach 5)

Another modification can be used to modify the design of Bell MF by replacing each CMult block found in the previous approach with a shift-add block to represent the gain operation and reduce the number of slices and LUTs required to implement this function. Figure 10 shows the suggested design based on the shift add approach represented by Subsystem1 to Subsystem5 blocks. Each Subsystem block contains shift and AddSub block components to perform the gain operation with less hardware device area. This replacement is built in the same manner used in Approach 3.

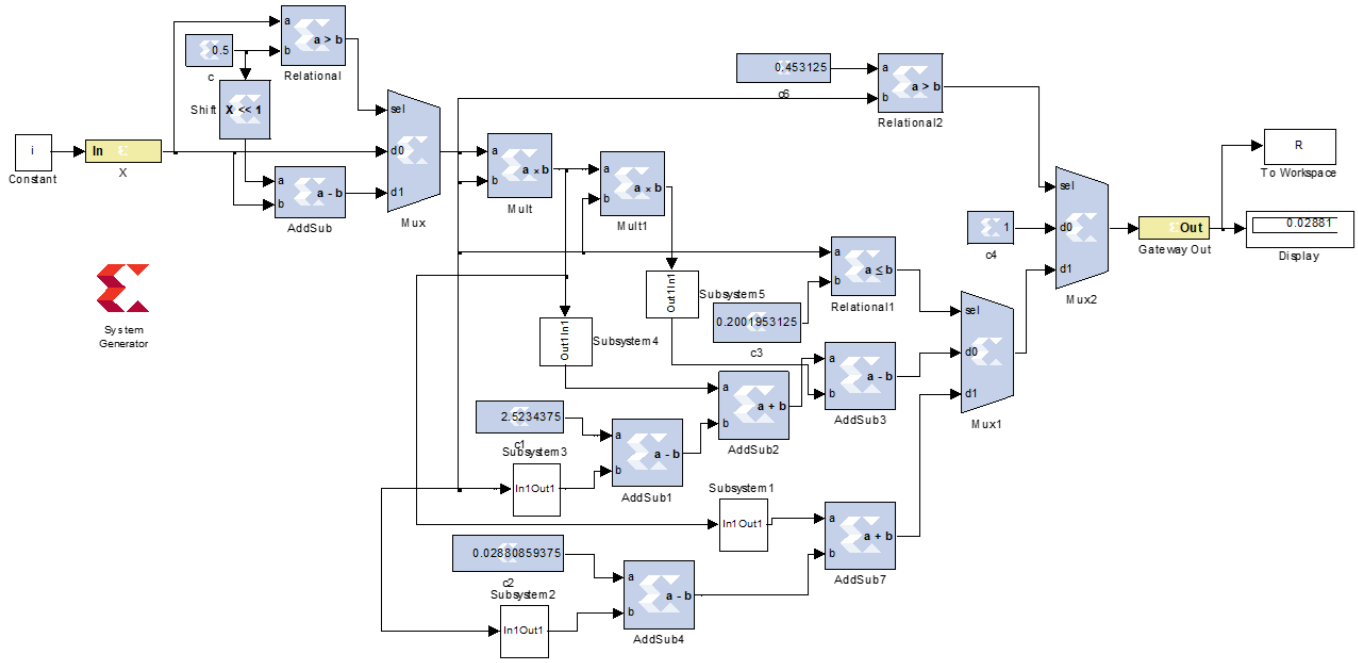

Figure 10: Bell MF based on decomposition and shift-add techniques

3.6 FPGA design of Bell MF based on precomputation technique (Approach 6)

The easy method to designing any function is pre-saving all expected output in ROM. Consequently, in this method, the input range of the Bell function is calculated and converted to binary representation. Therefore, evaluate the Bell function result for each input value and store them in ROM according to the input address. Corresponding to the input value range, the size of ROM, which is 256 locations, is chosen in this approach. Thus, the address will be 8 bits. The number of bits in each location of ROM is limited by the output resolution, where 16 bits of signed fixed point are adopted with 14 bits as a fraction. The BitBrasher block shown in Figure 11 is used to extract the least 8 significant bits from the input to be connected to the address of ROM.

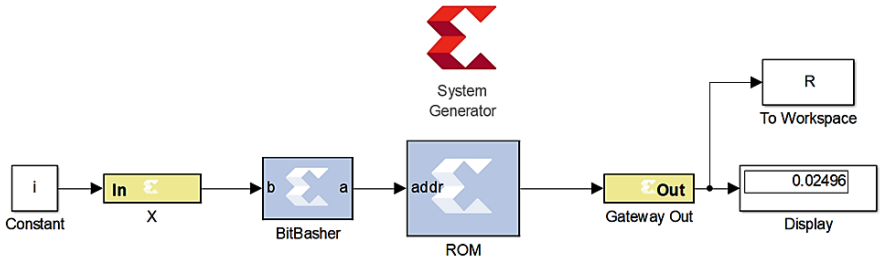

Figure 11: Bell MF based on precomputation technique

3.7 FPGA design of Bell MF based on modified precomputation technique (Approach 7)

Due to the Bell membership function being symmetrical across the center (c), the precomputation method can be reconstructed to minimize the device area needed to realize this approach in hardware. The reconstruction is accomplished by pre-saving the left half duration of the Bell MF in the ROM $(k\ to\ c)$, so the memory size will be reduced to half, which will be equal to 128 words. However, to obtain the value of this function for the other half, it should be adopted this formela $\big(f(x;\sigma,c) = f(2c - x;\sigma,c)\big)$. This equation is designed at the first stage of this approach, as demonstrated in Figure 12, where the Mux block is used to pass x if the input is less than or equal c. Otherwise, the value of $(2c - x)$ will appear at the output of the multiplexer. The decision of the multiplexer depends on the result of the Relational block that is used to compare the input with c. The Shift and AddSub blocks are employed to achieve the $(2c - x)$ term by doubling the value of c through the shift left operation, therefore, applying the subtraction operation. The output resolution of memory is 16-bit with a 14-bit fraction. This memory strobes the address from the BitBasher block used to pluck 7-bit from the multiplexer output.

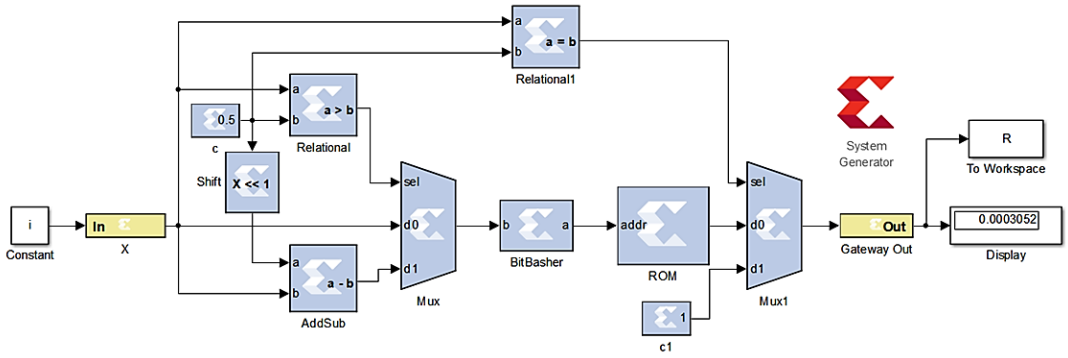

Figure 12: Bell MF based on modified precomputation technique

4. Experimental results and discussion of the proposed Bell membership functions

This section will discuss the results of the resource utilization, performance evaluation, and power consumption for the proposed designs of the Bell membership functions. All the suggested system designs are synthesized utilizing Spartan-3A DSP sc3sd3400a of the Xilinx FPGA kit.

4.1 Resource utilization of the proposed Bell MF designs

The hardware device utilization report analyzes the usage of resources in the proposed system architecture. It is included in Table 1 alongside the utilization report of [14]. As is clear from this table, the system suggested in [14], consumes a lot of hardware resources compared with the seven approaches proposed to realize the Bell MF. The large consumption of logic elements in [14], is due to using the basic representation of this function by writing the VHDL code, including a more complex design for the reciprocal function used to achieve the Bell equation. On the other hand, the proposed function based on approach

1 has the highest number of DSP48s equal to 10, because this method contains the highest number of multipliers that consume a lot of areas. After all, the Spartan-3A DSP platform has only (126) DSP48 units. In contrast, Approaches (2 and 3) and Approaches (4 and 5) have only (3 and 2), respectively, of the DSP48s, which can be considered a low consumption of multipliers compared to Approach 1.

Comparing the occupied slices and LUTs, the highest number of occupied slices and LUTs can be found in Approaches 1, 2, 3, 4, 5, and 6, which are as follows: (198, 129), (213, 391), (168, 279), (236, 394), (190, 311), and (101, 197), respectively. On the other hand, Approaches 6 and 7 have zero DSP48 blocks, which means there is no use for any multiplier in these two designs. However, Approach 7 has the lowest number (79) of occupied slices and (144) of LUTs with zero DSP48 block. Therefore, Approach 7 has the highest area minimization compared to the six proposed designs.

As a result, all suggested approaches have fewer hardware elements than those used in [14], to build the Bell function. Additionally, Approach 7 has the highest area minimization compared to the six proposed designs and [14], where this approach has 97.7% slice reduction and 96.7% LUTs reduction with respect to [14].

Table 1: Resource Utilization of the Proposed Bell MF Designs

Bell MF	No. of Slices	No. of LUTs	DSP48As
[14]	3544	4399	0
Approach 1	98	129	10
Approach 2	213	391	3
Approach 3	168	279	3
Approach 4	236	394	2
Approach 5	190	311	2
Approach 6	101	197	0
Approach 7	79	144	0

4.2 Performance evaluation and power consumption of the proposed Bell MF designs

In this section, two comparisons will be presented; the first is about the error rate, and the second is regarding power. To evaluate the performance error rate percentages for all functions and models that will be built in this work, the following equation will be adopted [20]:

$$Error\ Rate\ Percentage = \frac{|Estimated\ Value - True\ Value|}{True\ Value} * 100\% \quad (6)$$

where the estimated value represents the resulting value for the proposed system, while the true value represents the actual value of the Bell function in MATLAB.

Approaches 1, 2, and 3 have the highest error rate percentages (1.06%, 1.14%, and 1.16%), respectively, compared to the other four proposed designs, as shown in Table 2. In contrast, a slightly lower error percentage (0.49% and 0.61%) can be found in Approaches 4 and 5, respectively. The lowest error rate can be obtained by Approaches 6 and 7, with (0.17%) for each compared with the error rate of [14], which is 0.64%, and other approaches, as clear in the performance of the suggested approaches in Figure 13 (a-g). Besides, all the suggested approaches require one clock cycle to execute with a high speed equal to 791.766 MHz.

Table 2: Error Rate Percentage of the proposed Bell MF

Bell MF	Error Rate Percentage
[14]	0.64%
Approach 1	1.06%
Approach 2	1.14%
Approach 3	1.16%
Approach 4	0.49%
Approach 5	0.61%
Approach 6	0.17%
Approach 7	0.17%

The power consumption of the seven suggested designs for the Bell function is shown in Figure 14. It is obtained from the power consumption report summary produced by the Xilinx system generator after implementing the proposed design. The overall dissipating power can be defined based on the information provided in the figure. The proposed architecture based on techniques 2, 3, 4, and 5 has a power output of 183 mW, while Approaches 1 and 6 yield output power of 178 mW. The design employing method 7 can achieve a power consumption of 180 mW. Hence, the lowest power consumption is found in Approaches 1 and 6 because they depend on the basic FPGA blocks that consume a small amount of power. However, Approach 7 consumption of power is acceptable.

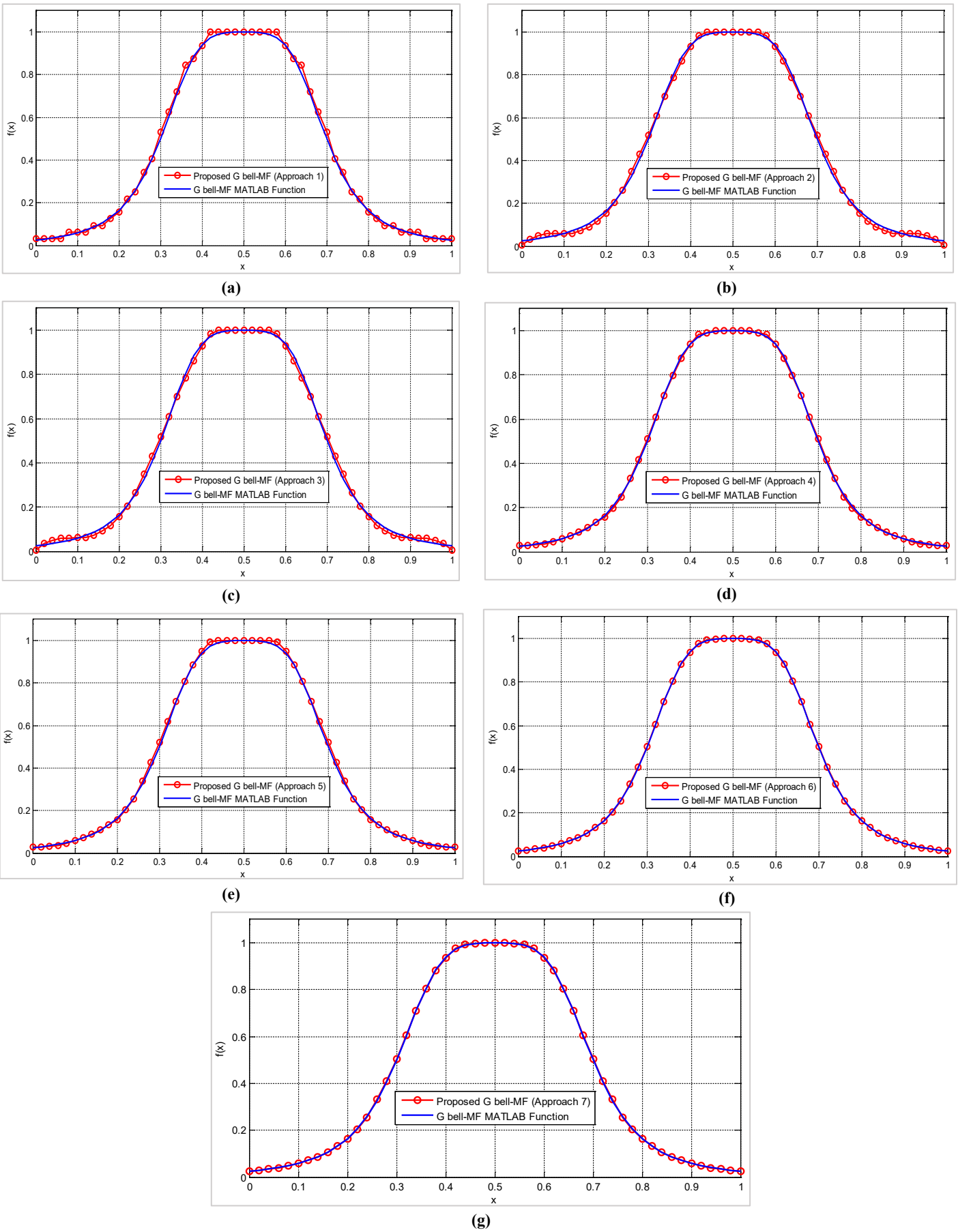

Figure 13: Performance of Bell membership functions based on: (a) hard FPGA block method, (b) polynomial curve fitting, (c) shift-add, (d) decomposition of polynomial curve fitting, (e) decomposition and shift-add, (f) precomputation, (g) modified precomputation

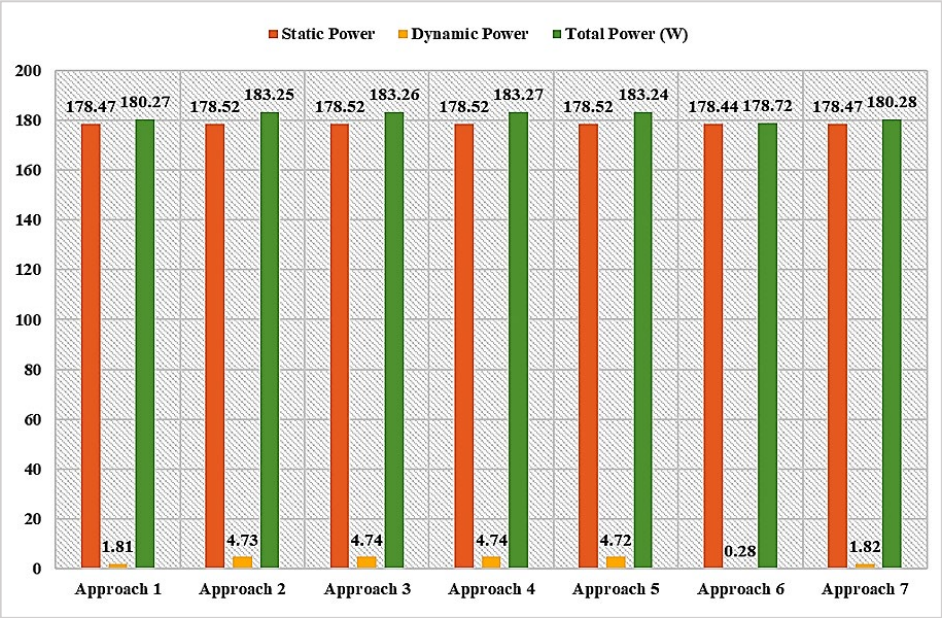

Figure 14: Power dissipation of the proposed Bell MF designs

5. Conclusion

This study highlights the importance of design selection and optimization of the Bell membership function for FPGA implementations. The proposed Bell membership functions were synthesized and implemented on the Spartan-3A DSP sc3sd3400a Xilinx FPGA kit. The proposed designs demonstrate significant improvements in resource utilization compared to the system described in previous works, which consumes a substantial amount of hardware resources. Additionally, the modified precomputation technique (Approach 7) emerged as the most resource-efficient design, utilizing the least number of slices (79) and LUTs (144) with zero DSP48 blocks. The absence of multiplication blocks for implementing this approach and using half the memory locations made it the best for designing Bell MF by FPGA. This indicates its superior area minimization capability. In contrast, the hard FPGA block method (Approach 1) showed the highest utilization of DSP48 blocks (10), which indicates a high multiplier consumption impacting area utilization adversely despite achieving certain performance benchmarks. Moreover, there is a clear trade-off between resource utilization, error rate, and power consumption. Designs like Approach 7 provide optimal resource usage and minimal error rates but may require further evaluation to ensure they meet all performance requirements. Higher resource-consuming designs, like Approach 1, may offer certain performance advantages but at the cost of increased hardware usage and potentially higher error rates. Designers must balance these factors based on the specific requirements of their application, prioritizing either minimal resource usage, low error rates, or acceptable power consumption.

Author contributions

Conceptualization, **N. Habeeb**, and **S. Abbas**; data curation, **N. Habeeb**, and **S. Abbas**; formal analysis, **N. Habeeb**; investigation, **N. Habeeb** and **S. Abbas**; methodology, **N. Habeeb**; project administration, **S. Abbas**, resources, **N. Habeeb**; software, **S. Abbas**; supervision, **S. Abbas**; validation, **S. Abbas**; visualization, **N. Habeeb**; writing—original draft preparation, **N. Habeeb**; writing—review and editing, **N. Habeeb** and **S. Abbas**. All authors have read and agreed to the published version of the manuscript.

Funding

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Data availability statement

The data that support the findings of this study are available on request from the corresponding author.

Conflicts of interest

The authors declare that there is no conflict of interest.

References

[1] A.N. Talpur, S. J. Abdulkadir, H. Alhussian, M. H. Hasan, N. Aziz, and A. Bamhdi, Deep Neuro-Fuzzy System application trends, challenges, and future perspectives: a systematic survey, Artif. Intell. Rev., 56 (2022) 865-913. https://doi.org/10.1007/S10462-022-10188-3

[2] A.M. I. S. Guerra, F. M. U. de Araújo, J. T. de Carvalho Neto, and R. G. Vieira, Survey on adaptative neural fuzzy inference system (ANFIS) architecture applied to photovoltaic systems, Energy Syst., 15 (2024) 505-541. https://doi.org/10.1007/S12667-022-00513-8/METRICS

[3] A.N. Talpur, M. N. M. Salleh, and K. Hussain, An investigation of membership functions on performance of ANFIS for solving classification problems, IOP Conf. Ser. Mater. Sci. Eng., 226 (2017) 012103. https://doi.org/10.1088/1757-899X/226/1/012103

[4] A.A. Y. Akal and A. E. A. El-Maaty, Fuzzy Assessment of Factors Influencing Quality Level of Highway Projects, Int. J. Manag. Fuzzy Syst., 2 (2016) 6-14. https://doi.org/10.11648/J.IJMFS.20160202.11

[5] A.A. K. Alazzawi and T. Ercan, Field-Programmable Gate Array Implementation of Adaptive Neuro-Fuzzy System Using Sensors Monitoring Health-Care Medicinal Internet of Things, J. Med. Imaging Heal. Informatics, 10 (2019) 169-177. https://doi.org/10.1166/JMIHI.2020.2693

[6] A.R. Naderkhani, M. H. Behzad, T. Razzaghnia, and R. Farnoosh, Fuzzy Regression Analysis Based on Fuzzy Neural Networks Using Trapezoidal Data, Int. J. Fuzzy Syst., 23 (2021) 1267-1280. https://doi.org/10.1007/S40815-020-01033-2/METRICS

[7] A.R. T. Jadhav, A Real Time Linearization of NTC Thermistor using Hybrid Neuro-Fuzzy Logic based on VLSI Technology, Int. J. Emerg. Trends Eng. Res., 8 (2020) 1570-1577. https://doi.org/10.30534/IJETER/2020/16852020

[8] A.M. Psarakis, A. Dounis, A. Almabrok, S. Stavrinidis, and G. Gkekas, An FPGA-Based Accelerated Optimization Algorithm for Real-Time Applications, J. Signal Process. Syst., 92 (2020)1155-1176. https://doi.org/10.1007/S11265-020-01522-5/METRICS

[9] A.E. Noorsal, S. Arof, S. Z. Yahaya, Z. Hussain, D. Kho, and Y. M. Ali, Design of an FPGA-Based Fuzzy Feedback Controller for Closed-Loop FES in Knee Joint Model, Micromachines, 12 (2021) 968. https://doi.org/10.3390/MI12080968

[10] A.M. Hermassi, S. Krim, Y. Kraiem, and M. A. Hajjaji, Adaptive neuro fuzzy technology to enhance PID performances within VCA for grid-connected wind system under nonlinear behaviors: FPGA hardware implementation, Comput. Electr. Eng., 117 (2042) 109264. https://doi.org/10.1016/J.COMPELECENG.2024.109264

[11] A.Ó. Mata-Carballeira, J. Gutiérrez-Zaballa, I. Del Campo, and V. Martínez, An FPGA-Based Neuro-Fuzzy Sensor for Personalized Driving Assistance, Sensors, 19 (2019) 4011. https://doi.org/10.3390/S19184011

[12] A.P. B. Indira and R. D. Krishna, Optimized adaptive neuro fuzzy inference system (OANFIS) based EEG signal analysis for seizure recognition on FPGA, Biomed. Signal Process. Control, 66 (2021) 102484. https://doi.org/10.1016/J.BSPC.2021.102484

[13] A.S. H. Klidbary, S. B. Shouraki, and B. Linares-Barranco, Digital hardware realization of a novel adaptive ink drop spread operator and its application in modeling and classification and on-chip training, Int. J. Mach. Learn. Cybern., 10 (2019) 2541-2561. https://doi.org/10.1007/S13042-018-0890-X/METRICS

[14] A.F. Karataş, İ. Koyuncu, M. Tuna, M. Alçın, Bulanık Mantık Üyelik Fonksiyonlarının FPGA Üzerinde Gerçeklenmesi, Bilgisayar Bilimleri ve Teknolojileri Dergisi, 1 (2020) 1-9. http://acikerisim.klu.edu.tr/xmlui/handle/20.500.11857/1336

[15] A.R. yousif, I. Hashim, and B. Abd, Implementation of Hyperbolic Sine and Cosine Functions Based on FPGA using different Approaches, Eng. Technol. J., 41 (2023) 1-16. https://doi.org/10.30684/ETJ.2023.139756.1440

[16] A.R. K. Yousif, I. A. Hashim, and B. H. Abd, Low Area FPGA Implementation of Hyperbolic Tangent Function, 6th Iraqi Int. Conf. Eng. Technol., its Appl. IICETA 2023, pp. 596-602, 2023. https://doi.org/10.1109/IICETA57613.2023.10351345

[17] A.H.H. Thannoon, I.A .Hashim, FPGA Implementation of Efficient Adaptive Filter Incorporating Systolic Architecture, Eng. Technol. J., 2 (2024) 261 -275. http://doi.org/10.30684/etj.2023.142877.1548

[18] A.S. Gupta, P. K. Biswas, B. Aljafari, S. B. Thanikanti, and S. K. Das, Modelling, simulation and performance comparison of different membership functions based fuzzy logic control for an active magnetic bearing system, J. Eng., 2023 (2023) e12229. https://doi.org/10.1049/tje2.12229

[19] A.M. Yilmaz, Evaluation of Total Antioxidant Capacity (TAS) by Using Fuzzy Logic, J. Adv. Math. Comput. Sci., 8 (2015) 433-446. https://doi.org/10.9734/BJMCS/2015/15919

[20] A.W. Guang, M. Baraldo, and M. Furlanut, Calculating percentage prediction error: a user's note, Pharmacol. Res., 32 (1995) 241-248. https://doi.org/10.1016/S1043-6618(05)80029-5