Zero Forcing Detector for 2x2-MIMO System Hardware Implementation Based on Gram-Schmidt QR Decomposition.

Trung L. Nguyen*, Hoang H. Pham*, Long H. Vu*, and Vu-Duc Ngo*
* School of Electrical and Electronics Engineering, Hanoi University of Science and Technology, Vietnam
E-mail: trung.nl186076@sis.hust.edu.vn, hoang.ph182544@sis.hust.edu.vn, long.vh182926@sis.hust.edu.vn, duc.ngovu@hust.edu.vn

***Abstract*—In this paper, we apply pipeline method associated with Algorithmic State Machine with Datapath (ASMD) technique on implementing Zero Forcing (ZF) Detector base on Gram-Schmidt QR Decomposition (GS-QRD) on FPGA to improve the performance of system. Synthesis is performed by Vivado Xinlinx 2022 to validate the implementation, as well as to show the efficiency of the architecture.**

***Index Terms*—Zero Forcing (ZF), Gram-Schmidt QR Decomposition (GS-QRD), Multiple Input Multiple Output (MIMO).**

I. Introduction

A. Related works and motivations

Along with the impressive development of telecommunication, Multiple Input Multiple Output (MIMO) technique has been shown to be an efficient means of encountering fading phenomenon and/or increasing data rate in wireless channels [1]. Inheriting the idea of T.B Nguyen *et al.* from [1], or more particularly, Base on their idea of Group Detection for Massive MIMO systems, we try to implement a very fundamental and simple kinds of channel: a Zero-Forcing (ZF) Detector for a 2x2-MIMO system. While each H-component (H1, H2) is a separation of the channel matrix and has high length of each dimension, our channel matrix is just a 4x4-complex matrix, for simplicity. The efficiency of the method was approved in the paper, so that we have a very concrete foundation to deploy our implementation. We also consult the ZF-Detector block from the transmission model of a 2x2-MIMO channel with ZF detection from [2] as our referred model of architecture.

One of the biggest problem of implementing a ZF-Detector algorithm is finding the inverse of a matrix [2]. To resolve this stuff, people propose some effective solutions to avoid finding directly the inverse matrix such as QR Decomposition method. Accordingly, the computational complexity of the entire architecture is much depended on how we design the QR-Decomposition module. It leads to the consequence that we will evaluate the efficiency of our architecture by comparing our QR-Decomposition module's specifications to others, along with the entire ZF-Detector device.

Throughout the recent decades, many implementations for conventional MIMO detector and QR decomposition were deployed and achieves significant improvements. To come up with an idea of what we can do, we have studied a lot of them and built up our own proposition. In [3], P. Luethi *et al.* perform a VLSI Implementation of a High-Speed Iterative Sorted MMSE QR Decomposition. Accordingly, they lead out the throughput of their QR Decomposition that can be visually used to compare with our result. In [4], the authors introduce their design of Gram-Schmidt-based QR Decomposition for MIMO Detection. The architecture is implemented as a VLSI project and compared with previous works via their throughput performances and clock latency, also the used resources. Instead of deploy a design based on CMOS architecture as the two above propositions, the authors in [5] propose a new scalable QRD structure based on the master-slave-pair element that leads to the improvement of throughput/area efficient for MIMO Systems on FPGA. In [6], V.S Trinh *et al.* proposed the implementation Of low-complexity detector For high-rate spatial modulation, which results very high throughput, compared to the three mentions before. However, beside the very efficient algorithm, the implementation of hardware on FPGA seems not to be the most optimized one. All of them lead us to an idea of what we can do to re-design the hardware architecture, to optimize the resource and achieve higher throughput.

B. Contributions

In this paper, we use a new way to apply recent methods to design ZF-Detector's hardware architecture for 2x2-MIMO system to achieve higher throughput. We do not concentrate on redesign the algorithm but just simply refer to a very popular algorithm to resolve the inverse problem, which is Gram-Schmidt QR Decomposition (GS-QRD). Our new point is applying pipe-lining technique while breaking the architecture into small components, associating with ASMD implementing method. Consequently, we achieve significant results for both two types of data component's bit-width: 16-bit and 32-bit, which will be shown in the section RESULT.

The remainder of the paper is organized as follows:

- In Section II, our system model is demonstrated.
  - In part ***A. Related Theoretical Concepts***, we express an overview about Zero-Forcing concept, 2x2-MIMO System and QR Decomposition.
  - In part ***B. Data Format***, we explain the way we perform data throughout the architecture.

  - In part ***C. Algorithm***, we sketch out the ASMD and the TOP-Module of our architecture. There are three significant things to discuss: how we process the arithmetic square-rooting and dividing, then how we design the decomposition process.
  - In part ***D. Architecture***, we show out an illustration of our hardware implementation.
- In Section III, we figure out our synthesising results and how efficient the architecture is.
- Finally, we give a conclusion of what we have done and our achievement.

II. System model

A. Related Theoretical Concepts

1) Zero-Forcing Concept: Zero-forcing (ZF) technique is so named because it forces the residual inter-symbol interference (ISI) to zero [7]. Considering a typical channel model, where **x** is sent signal at the transmitter, y is received signal at the receiver, **H** is channel characteristics matrix, and n is Addictive White Gaussian Noise (AWGN), then we obtain the following equation:

$$\mathbf{y = Hx + n}. \tag{1}$$

One of the biggest problem of a transceiver system is how to reconstruct x after receiving y, and Zero-Forcing is one of the most popular technique to resolve this question, which is regarding to a conventional problem: Finding $\mathbf{x} \in R^N$ so that $(|\mathbf{y} - (\mathbf{Hx+n})|)^2$ reach its minimum value, where N is the number of dimensions of **x**. Apparently we can see the solution is $\mathbf{x}_{ZF} = \mathbf{H}^{-1}(\mathbf{y\text{-}n})$, where $\mathbf{H}^{-1}$ is the inverse matrix of **H**, if **H** is a complex matrix.

2) 2x2-MIMO System: A 2x2-MIMO System uses two antennas to establish up to two streams of data with the receiving device [8]. Therefore, we can consider **x**, **y** and **n** as 2x1 complex matrices, and **H** as a 2x2 complex matrix. To be more obvious, we can refer to Fig.1, which was mentioned in [2].

3) QR Decomposition: The QR decomposition (also called the QR factorization) of a matrix is a decomposition of the matrix into an orthogonal matrix and a triangular matrix [9]. A QR decomposition of a real square matrix **H** is a decomposition of **H** as **H = QR**, where **R** is an upper triangular matrix and **Q** is an orthogonal matrix.

QR Decomposition technique helps to avoid finding directly the inverse of channel matrix **H**. Particularly, if we have the channel equation as eq.1, then we can substitute **H=QR** to obtain: **y=QRx+n** *equivalent* $\mathbf{Q}^T\mathbf{y} = \mathbf{Rx} + \mathbf{Q}^T\mathbf{n}$ *equivalent* $\mathbf{Q}^T(\mathbf{y\text{-}n}) = \mathbf{Rx}$, where $\mathbf{Q}^T$ is the transpose of **Q**. While **R** is an upper triangular matrix and **Q** is an orthogonal matrix, we can easily find **x** with a low computational complexity.

One of the popular methods to find **Q** and **R** is Gram-Schmidt process, which will be discussed in detail in Part ***B. Algorithm***.

To be more obvious to follow, we denote that, the Gram-Schmidt process requires some complex computational arithmetic, such as square-rooting and division. We will discuss about them fist, before we turn to Gram-Schmidt algorithm.

B. Data Format

Each matrix component is a real number, use signed fixed point binary to represent each real number.

We develop our architecture for both two kinds of data format: one is 16-bit and another is 32-bit for data component's bit-width.

- For the 16-bit one, the sign-bit takes 1 bit - the most significant bit (MSB), the following 3 bits are for integer part, and the last 12 bits are for fractional part (or mantissa).
- For the format 32-bit, the number of bit for sign-bit, integer part and mantissa are 1, 7 and 24, respectively.

C. Algorithm

1) Square-root Arithmetic: Square-rooting is one of an essential arithmetic to acquire the Gram-Schmidt process. To perform square-root with fixed-point number as mentioned before, we refer a very effective algorithm from [10], as the following pseudo-code Algorithm1.

2) Non-restoring Division: Similar to the role of Square-root, Division is required to accomplish Gram-Schmidt process. For that arithmetic, we have just referred an algorithm from [11] as the following pseudo code Algorithm2. The algorithm we use here is a kind of very popular and effective one, named non-restoring division, which can be implemented for fixed-point number.

3) Gram-Schmidt QR Decomposition: Gram-Schmidt Process is a popular technique to resolve the QR decomposition problem. The algorithm of this method is refer from [12] and can be describe as Algorithm 3

4) Finding X: Denote the indexes of matrices as follow:

$$\mathbf{x} = \begin{bmatrix} x_1 & x_2 \\ x_3 & x_4 \\ x_5 & x_6 \\ x_7 & x_8 \end{bmatrix} \tag{2}$$

$$\mathbf{y} = \begin{bmatrix} y_1 & y_2 \\ y_3 & y_4 \\ y_5 & y_6 \\ y_7 & y_8 \end{bmatrix} \tag{3}$$

$$\mathbf{n} = \begin{bmatrix} n_1 & n_2 \\ n_3 & n_4 \\ n_5 & n_6 \\ n_7 & n_8 \end{bmatrix} \tag{4}$$

$$\mathbf{R} = \begin{bmatrix} R_1 & R_2 & R_3 & R_4 \\ R_5 & R_6 & R_7 & R_8 \\ R_9 & R_{10} & R_{11} & R_{12} \\ R_{13} & R_{14} & R_{15} & R_{16} \end{bmatrix} \tag{5}$$

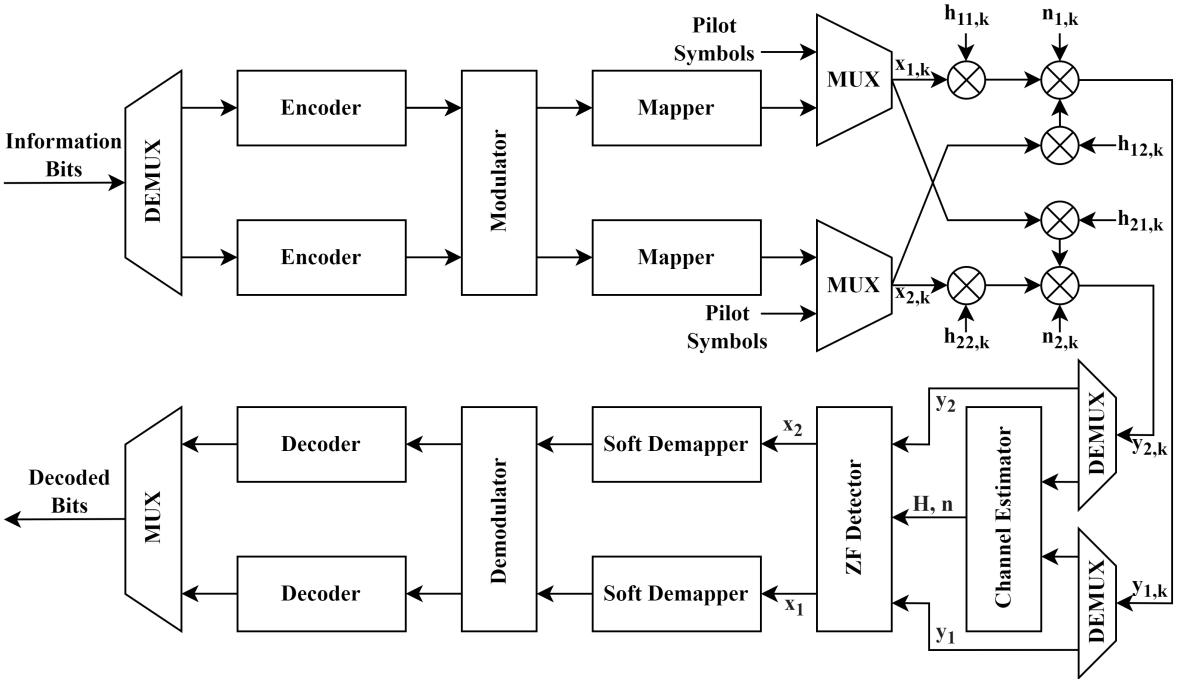

Fig. 1: Transmission model of a 2×2-MIMO with ZF detection

$$\mathbf{Q} = \begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{21} & q_{22} & q_{23} & q_{24} \\ q_{31} & q_{32} & q_{33} & q_{34} \\ q_{41} & q_{42} & q_{43} & q_{44} \end{bmatrix} \tag{6}$$

From the section above (II-A), we have $\mathbf{Q}^T(\mathbf{y} - \mathbf{n}) = \mathbf{Rx}$. Denoting the matrix component as above, we find matrix x by the process describe in Fig.4

D. Architecture

1) Pipeline Architecture: Our architecture use pipe-lining technique. Generally, each pipe-line block has the operation design as shown in Fig. 2.

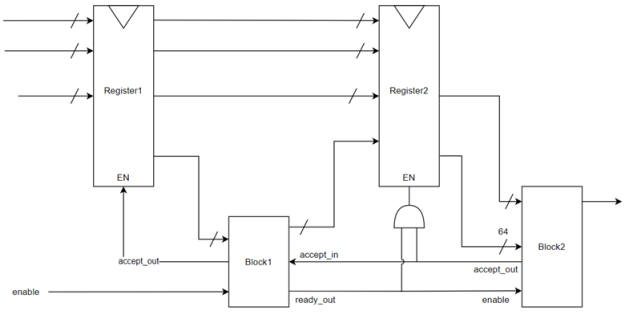

Fig. 2: Pipe-lining Block

When *Block1* is enabled, its *accept_out* allows *Register1* to work. After that, data from *Register1* is processed in *Block1*. When *Block1* finish calculating, its *ready_out* will enable *Block2*. At this moment, *Block2*'s *accept_out* and *Block1*'s *ready_out* allow to *Register2* to enable and load data of *Block1*'s output. At the same time, *Block2*'s *accept_out* makes *Block1* to be ready to enable more time.

The process is repeat again and again follows the finite state machine (FSM) in Fig 3

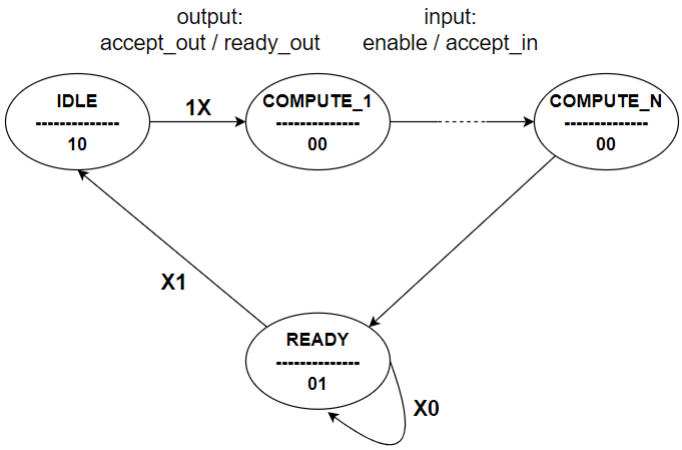

Fig. 3: Finite State Machine of The Design

2) Designing: As mentioned in the explanation of GS-QRD process, when calculating columns of matrix Q, we need the results from previous calculation. Without pipe-lining, we have to calculate those columns one-by-one, or in other words, we have to done calculating the previous ones to start the next one. It leads to a consequence that it will take so many clock circles to done calculating the entire process, especially while the input is provided continuously. Applying pipe-lining, we obtain so effective architecture. The architecture is shown in Fig. 4. As shown, each block takes inputs from previous block,

Algorithm 1 Square-root Algorithm

1: **Start**.
2: Prepare input data D (radicand), remainder R, square root Q (quotient), partial factor F, and bit-index i.
3: Initialize radicand with input data value, R=0, Q=0, F=0, i=n; n is MSB bit-index of D.
4: **If** radicand has odd digits **Then**
Expand radicand with a bit of "0" as MSB.
Go to step 5.
Else (even digits)
Go to step 5.
5: Divide the radicand into sub-groups which each sub-group consists of 2 digits starting from integer LSB.
6: Start the calculation from MSB sub-group to LSB sub-group. Treat current sub-group as current partial remainder.
$R_t = D_t[i : i-1]$; t is time index indicator.
7: Do a comparison whether current partial remainder is bigger or equal than current
If yes
Update Q; $Q_{t+1} = (Q_t << 1)|1$;
Update F; $F_{t+1} = ((F_t + F_t[0]) << 1)|1$;
Else
Update Q; $Q_{t+1} = (Q_t << 1)|0$;
Update F; $F_{t+1} = ((F_t + F_t[0]) << 1)|0$;
8: Do subtraction to partial remainder by the result value of factor multiplication; Then append the subtraction result with next subgroup data of D in the LSB position of partial remainder, in order to update R.
$R_{t+1} = ((R_t - (F_t * F_t[0])) << 2|D[i-2 : i-3]$;
9: **Update** the current indexes for next use;
t+1 is changed into t;
i-2 is changed into i;
i-3 is changed into i-1;
10: **If** the process is not over
Jump to step 7 and loop the process.
Else (process is over)
Latest Q value is final square root value.
Latest R value is final remainder value.
11: **End**.

Algorithm 2 Non-Restoring Division Algorithm

Input: Radicand X and word length n.
Output: Quotient Q and Remainder R.

Initialization $R_0 = X$.
for $i \leftarrow 1$ to n **do**
  if $2 * R_{i-1} \geq 0$ **then**
    $Q(i) = 1$
    $R_i = 2R_{i-1} - (2Q_i + 2^{-i})$
  else if $2 * R_{i-1} < 0$ **then**
    $Q(i) = -1$
    $R_i = 2R_{i-1} + (2Q_i - 2^{-i})$

Algorithm 3 Gram-Schmidt Process

Input: 4x4 real matrix $\boldsymbol{H}$.
Output: 4x4 orthogonal real matrix Q and 4x4 upper triangular real matrix R.
Denote:
$\mathbf{H} = [h_1\ h_2\ h_3\ h_4]$, *where* h_i is a column of $\mathbf{H}$, $i = 1, 2, 3, 4$.
$\mathbf{Q} = [q_1\ q_2\ q_3\ q_4]$, *where* q_i is a column of $\mathbf{Q}$, $i = 1, 2, 3, 4$.

1: Calculate the first column of $\mathbf{Q}$: $q_1 = \frac{h_1}{||h_1||}$
2: Calculate the second column from the first column of $\mathbf{Q}$: $q_{12} = -q_{21}, q_{22} = q_{11}, q_{32} = -q_{41}, q_{42} = q_{31}$, *where* the footnotes of q follow Equation (7);
3: Calculate the third column of $\mathbf{Q}$: $q_3 = \frac{h'_3}{||h'_3||}$, *where* $h'_3 = h_3 - < h_3, q_1 > q_1 - < h_3, q_2 > q_2$;
4: Calculate the last column from the third column of $\mathbf{Q}$: $q_{14} = -q_{23}, q_{24} = q_{13}, q_{34} = -q_{43}, q_{44} = q_{33}$, *where* the footnotes of q follow Equation (6);
5: Calculate matrix R: $\mathbf{R} = \mathbf{Q}^{-1}\mathbf{H} = \mathbf{Q}^T\mathbf{H}$

stores them and processes at suitable time, that helps to reduce the total number of wasting clocks.

III. Result

A. Verification

To verify the result of this architecture, we have implemented a software application to stimulate the detector by all the referred algorithms. This app is able to randomly generated a large number of test-cases, as the input matrix x, then calculate and show out corresponding outputs, which are verified by referred documentation' algorithm. Those obtain results are compared with the results from hardware simulation by Modelsim 2020.4, after taking those generated test-cases and producing corresponding outputs.

After performing several times, the accuracy of the 32-bit one always acquires 100% matching between software and hardware results, while the accuracy of 16-bit is around 90%.

B. Comparison

This architecture was synthesized on Zynq UltraScale+ ZCU106 Evaluation Platform by Xilinx Vivado 2021.2. The obtained results are shown in TABLE I with other cited works so that it is obvious to compare. Note that the target chip technologies are not the same regarding to different objects, also the difference of algorithms is apparent, so that the comparison is not completely fair. However, by the outstanding performance of detecting throughput, clearly we can not negate the dominance of our architecture.

Throughout the five mentioned documentations from TABLE I, the corresponding authors perform their propositions for MIMO system. All of them perform a VLSI implementation for QRD decomposition. The first two [3] and [4] deploy their design on CMOS technology, while the others use FPGA board. In [3], the authors describe the architecture and results of the first VLSI implementation of an iterative sorted QR decomposition pre-processor and achieve the performance as

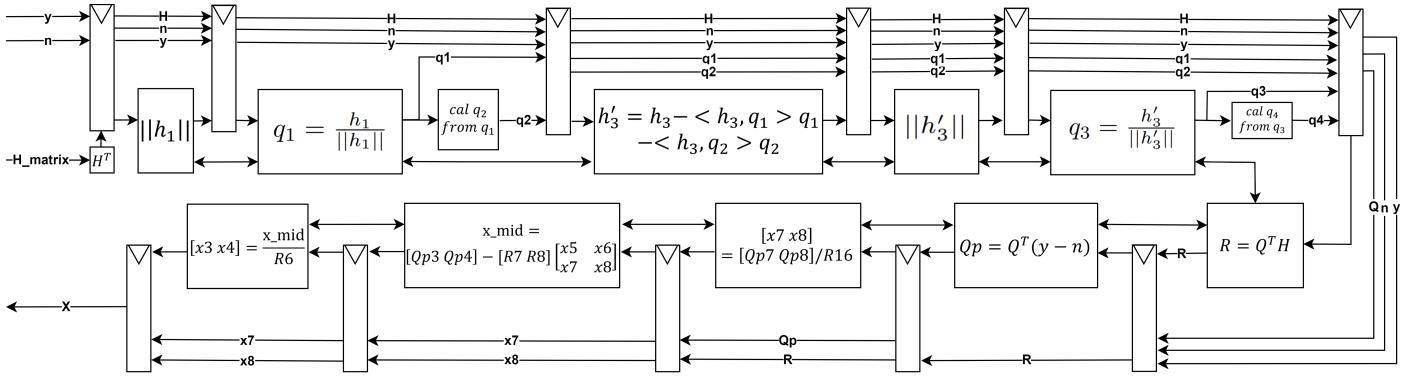

Fig. 4: Designed Architecture

TABLE I: SYNTHESIS RESULTS OF OUR ARCHITECTURE AND SOME RELATED WORKS

Section	[3]	[4]	[5]	[6]	This work			
					QR Decomposition		ZF Detector	
Algorithm	GR-SQRD	MGS-SQRD	GR	ISQRD Sorted MGS	GS-QRD			
Target Chip Technology	0.25µm 1P/5M CMOS	UMC 0.18 µm 1P/6M CMOS	Xilinx Virtex-6	Xilinx Virtex-5	Zynq UltraScale+ ZCU106 Evaluation Platform			
Element Bit-width	13-bit	20-bit	12-bit	12-bit	16-bit	32-bit	16-bit	32-bit
Maximum Frequency	166 MHz	162 MHz	128 MHz	365 MHz	467 MHz	293 MHz	467 MHz	293 MHz
Hardware Usage	54k gates	61.8k gates	1851 Slices	26128 FFs 17197 LUTs 260 DSPs	4,843 FFs 6684 LUTs 80 DSPs	11535 FFs 14092 LUTs 320 DSPs	6933 FFs 6564 LUTs 66 DSPs	15266 FFs 13640 LUTs 264 DSPs
Detecting Throughput	432.6 Mbps	499.2 Mbps	179.3 Mbps	2.9 Gbps	4.1 Gbps	2.6 Gbps	8.2 Gbps	5.3 Gbps
Latency	80 clocks	104 clocks	137 clocks	165 clocks	114 clocks	198 clocks	184 clocks	324 clocks

shown in the TABLE I. Meanwhile, In [4], Luethi *et al.* propose a VLSI architecture that integrated in 0.18 µm CMOS technology processes up to 1.56 million complex valued 4×4-dimensional matrices per second. The obtained result is high compared with the cited work and we can use it to compare with our architecture. In [5], the authors present a modified QRD algorithm for MIMO which is based on the basic master-slave-pair element and deeply pipelined CORDIC core. The achieved throughput and area efficiency is higher than other cited works, so that can be used to highlight our out-performed architecture. Finally, in [6], V.D.Ngo *et al.* introduced an architecture for HR-SM system, which has spectral efficiency of $(2(n_T - 1) + log_2 M) bits/s/Hz$. This architecture also used pipe-lining design method and acquired a high throughput. In our proposition, we design an architecture for a 2x2 MIMO system by pipe-lining method, by associated with ASMD, then the achieved results is outstanding compared with others.

IV. Conclusions

In this paper, we have applied a new way to implement ZF Detector for 2x2-MIMO within GS-QR Decomposition on FPGA. Accordingly, we achieve very promising performances as shown in TABLE I. The results have also been verified by software computation, as shown in section III-A, which is referred by the theoretical background from previous accepted works.

Throughout the above evaluation, it is conspicuous to see that the 16-bit architecture achieve much higher throughput, i.e. 8.246 Gbps at 467.07 MHz, but less accuracy compared with the 32-bit one. Apparently, the design for 32-bit architecture also gain an outstanding level of throughput, which is 5.268 Gbps, while it can run at the clock speed of 293.26 MHz. Not only that, the 32-bit one also reach a very high accuracy through a bunch of computational tests by C++ code (referring to part III-A). However, the 16-bit design is not a kind of breakdown. Due to the limitation of time and resources, we have just used the fixed-point structure for data representation. That leads to a fact that the maximum range of value that we can represent using that structure is quite bounded. Nevertheless, the 16-architecture lays the foundation for further development of using floating-point structure. Meanwhile, the

maximum range of value that a data component can represent will be significantly improved. Consequently, the accuracy problem of this kind of architecture will be resolved.

REFERENCES

[1] T.-B. Nguyen, T.-D. Nguyen, M.-T. Le, and V.-D. Ngo, "Efficiency zero-forcing detectors based on group detection for massive mimo systems," in *2017 International Conference on Advanced Technologies for Communications (ATC)*, 2017, pp. 48–53.

[2] I.-W. Lai, S. Godtmann, T.-D. Chiueh, G. Ascheid, and H. Meyr, "Asymptotic ber analysis for mimo-bicm with zero-forcing detectors assuming imperfect csi," in *2008 IEEE International Conference on Communications*, 2008, pp. 1238–1242.

[3] P. Luethi, A. Burg, S. Haene, D. Perels, N. Felber, and W. Fichtner, "Vlsi implementation of a high-speed iterative sorted mmse qr decomposition," in *2007 IEEE International Symposium on Circuits and Systems*, 2007, pp. 1421–1424.

[4] P. Luethi, C. Studer, S. Duetsch, E. Zgraggen, H. Kaeslin, N. Felber, and W. Fichtner, "Gram-schmidt-based qr decomposition for mimo detection: Vlsi implementation and comparison," in *APCCAS 2008 - 2008 IEEE Asia Pacific Conference on Circuits and Systems*, 2008, pp. 830–833.

[5] W. Zhao, J. Lin, and S.-C. Chan, "Throughput/area efficient fpga implementation of qr decomposition for mimo systems," in *2016 IEEE International Conference on Digital Signal Processing (DSP)*, 2016, pp. 522–526.

[6] V.-D. N. Van-Son Trinh, Thuong-Duc Duong, "High throughput fpga implementation of low-complexity detector for high-rate spatial modulation," 2020.

[7] M. Viswanathan, *Wireless Communication Systems in Matlab*, 2nd ed., 2020.

[8] 2x2 mimo. [Online]. Available: https://halberdbastion.com/resources/wireless/mimo/2x2-mimo

[9] (2022) Qr decomposition. [Online]. Available: https://en.wikipedia.org/wiki/QR_decomposition

[10] R. V. W. Putra, "A novel fixed-point square root algorithm and its digital hardware design," in *International Conference on ICT for Smart Society*, 2013, pp. 1–4.

[11] H. Thapliyal, T. S. S. Varun, E. Munoz-Coreas, K. A. Britt, and T. S. Humble, "Quantum circuit designs of integer division optimizing t-count and t-depth," in *2017 IEEE International Symposium on Nanoelectronic and Information Systems (iNIS)*, 2017, pp. 123–128.

[12] P. Desai, S. Aslan, and J. Saniie, "Fpga implementation of gram-schmidt qr decomposition using high level synthesis," in *2017 IEEE International Conference on Electro Information Technology (EIT)*, 2017, pp. 482–487.