ISSN: 0975-833X

Available online at http://www.journalcra.com

International Journal of Current Research
Vol. 17, Issue, 06, pp.33207-33213, June, 2025
DOI: https://doi.org/10.24941/ijcr.49095.06.2025

INTERNATIONAL JOURNAL OF CURRENT RESEARCH

RESEARCH ARTICLE

SOFTWARE-DEFINED RADIO-BASED IMPLEMENTATION OF 1024-QAM TRANSCEIVER ONTO AN FPGA USING REED-SOLOMON CODING-DECODING AND MATLAB-SIMULINK MODEL AND USING MATCHED FILTER DETECTION

***Rehan Muzammil**

Department of Electronics Engineering, Aligarh Muslim University, Aligarh, India

ARTICLE INFO

Article History:
Received 09th March, 2025
Received in revised form
21st April, 2025
Accepted 19th May, 2025
Published online 24th June, 2025

Keywords:

QAM, Reed-Solomon-Coding-Decoding, FPGA, SIMULINK, Software Defined Radio, Matched Filter.

ABSTRACT

Higher-order quadrature amplitude modulation (QAM) schemes such as1024-QAM are expected to increase data rates within a limited bandwidth in 5G mobile systems. Higher-order QAM also improves bandwidth efficiency. In this paper, a complete 1024-QAM transceiver is designed and implemented using model-based development in Matlab/Simulink using Matched Filter Demodulation/Detection. This paper also uses forward error correction (FEC) as (15,8) Reed-Solomon Coding-Decoding. The complete transceiver is implemented onto a Kintex-7 FPGA. The reconfigurability and speed of the FPGAs is the driving force for this approach. Model-based design development saves much time as far as the design is concerned. The results are in the form of a comparison between input and output bit streams.

Citation: Rehan Muzammil. 2025. *"*Software-defined Radio-based Implementation of 1024-qam Transceiver onto an FPGA using Reed-Solomon Coding-Decoding and Matlab-Simulink model and using Matched filter Detection*". International Journal of Current Research, 17, (06), 33207-33213.*

INTRODUCTION

One of the objectives of 5G systems is to increase data rates and improve user throughput. A peak data rate of more than 20 Gbps is expected. The wide bandwidth, Carrier aggregation, and multiple-input multiple-output (MIMO) spatial multiplexing can be implemented in 5G mobile systems as they are implemented in 4G mobile systems. A higher-order modulation scheme can increase the data rate within a given bandwidth. As defined by the 3GPP Release 8, the 4G systems apply modulation schemes like QPSK, 16-QAM, and 64-QAM for symbol modulation in the OFDM multiplexing. 3GPP Release 12 introduced the implementation of 256-QAM, and more recently, 3GPP Release 15 introduced 1024-QAM support. However, as these modulation schemes are susceptible to nonlinear distortion, noise, interference, and multipath fading, the applications for the 256-QAM and 1024-QAM schemes are currently limited (1). Based on Long Term Evolution (LTE) and LTE-Advanced systems, 5G mobile systems are being developed. The main requirements for 5G mobile systems are an increased data rate, an increase in system capacity, a lower latency, and higher connectivity density, like in the Internet of Things (IoT) (3). 5G mobile systems aim to increase system capacity and data rates.The system capacity and data rates can be grown in 5G mobile systems using spatial multiplexing by using the MIMO antenna. The data rates can be increased in such systems using higher-order modulation schemes within the given bandwidth. However, with limited applications and conditions in implementing the 1024-QAM mobile procedures described in 3GPP Release 15 (4). Another application, the cable television system, was built initially for terrestrial analog broadcasting services where the transmission line was based on coaxial-type cable. 64-QAM and 256-QAM modulation formats are presently viable for digital cable television, which can transmit at a data rate of 30 to 40 Mbps over a transmission bandwidth of 6 MHz. Broadcasting systems are upgraded and digitized in the 21st century. The Hybrid Optical Fiber and Coaxial (HFC) is called the physical layer of the cable television system. A 1024-QAM modulation scheme achieves higher spectrum efficiency, which carries 10 bits per symbol (2). The Open Cable standard adopted the 64-QAM and 256-QAM modulation schemes for the CATV broadcasting services and data transmission. Many cable operators have upgraded responding to increased consumer networks several years ago and increased the frequency range for the downstream transmission of up to 1 GHz. As a result, most cable operators currently offer high-quality video-type services, offering an enhanced analog TV package and about a hundred digital TV channels. As the analog channel undergoes digitization, new channels are introduced, and cable operators offer realistic broadcasting and bandwidth-demanding services such as UHDTV. A significant capacity problem in the coaxial parts of the hybrid fiber coax network has been observed by the introduction of 4K-UHD (3840x2160), 8K-UHD(7680x4320),and band-intensive realistic broadcasting TV services (5). For this reason, 1024-QAM modulation schemes for UHDTV have been proposed in previous research where the two points have not been addressed are the burst noise handling issue and the practical implementation issue (5). For the burst noise, Reed-Solomon coding-decoding is an ideal solution. For practical implementation issues, FPGA is a perfect solution. As the number of devices connected to the internet is increasing and the amount of data handled is becoming huge, a high throughput rate is demanded for wireless local area networks (WLAN). MIMO and 1024-QAM systems are the candidates for higher throughput rates in the transceivers with higher-density modulation schemes. RX EVM has a more significant impact on transmission performance than the TX EVM in delivering the 1024-QAM OFDM signal and using MIMO transceivers (6, 9). A 3rd-generation (3G) enhanced and a 4th-generation (4G) technology has

been diffused worldwide for Long Term Evolution (LTE). Furthermore, well-conducted studies are done on LTE-Advanced and beyond. Among these efforts, it is crucial to improve average user throughput and cell edge user throughput performance and further increase system capacity. The amount of data traffic on mobile communication systems has increased rapidly, and the data rates requested by end-users have also increased. It ischallenging to produce a precise transmitter/modulator, and receiver/demodulator error vector magnitude (EVM) must be evaluated when 256-QAM and 1024-QAM are implemented. However, the Bit Error Rate (BER) is not clarified by link-level computer simulations ofthe EVM effect on transmission performance (7). Very little work has been published regarding the transmitters for the 1024-QAM single carrier signals. Most higher-order constellation works use Doherty amplifier structures and out-phasing, which also require feedback linearization using digital pre-distortion (DPD). Switched-capacitor techniques using CMOS are also investigated for 256-QAM, and feedback linearization is needed (8).
This paper gives a practical implementation of a 1024-QAM transceiver with FEC in the form of (15,8) Reed-Solomon coding-decoding. This coding-decoding can detect up to 3 symbols or 12 bits of burst error and correct them. This paper is organized as follows: The II section gives the system architecture. Section III provides an overview of the 1024-QAM multiplexing and Modulation. Section IV illustrates 1024-QAM De-Multiplexing, Demodulation, and Detection. Section V describes the (15, 8) Reed-Solomon coding. Section VI describes the (15,8) Reed-Solomon decoding. Section VII gives an overview of the hardware implementation and Real-time results. Section VIII concludes.

System Architecture: The 1024-QAM transceiver is divided into two sections, which are the transmitter and the receiver section. These sections are connected in the form of an FPGA-in-the-loop configuration. This system only describes the baseband section of the transceiver. This is well justified as almost all the computations are performed in the baseband section of the Software Defined Radio. The radio's IF and RF sections can be easily connected to the transceiver baseband section and are not considered in this paper. The transceiver is run onto the FPGA as Software-Hardware co-simulation using a System Generator where some of the transceiver runs on the host CPU and the rest runs on the FPGA. The top-level system architecture is given in Figure 1. The 1024-QAM transceiver is implemented into the subsystem, shown in Figure 2.

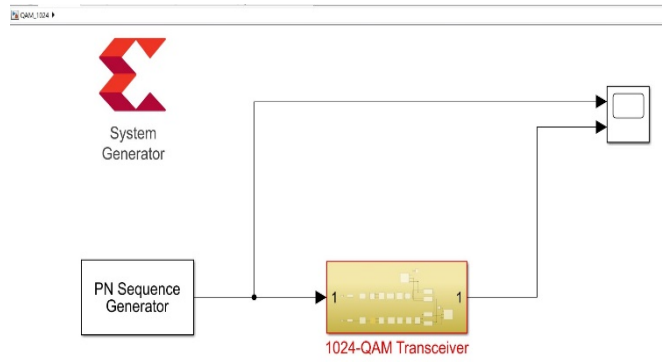

Fig. 1. QAM-1024 Top Level Model

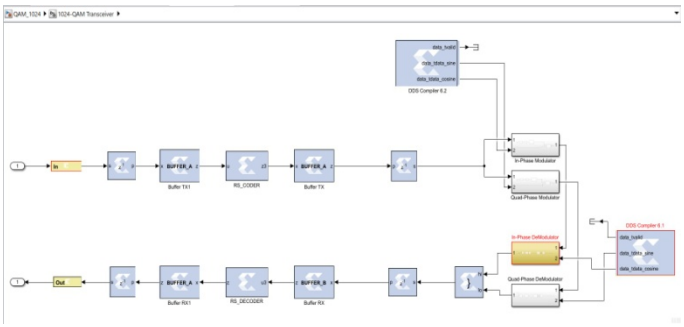

Fig. 2. The 1024-QAM Transceiver Model.

Figure 1 illustrates that the PN Sequence Generator generates the random bit stream. This random bit stream is injected into the 1024-QAM Model as the transmitted bit stream. At the entry into the 1024-QAM Model, the serial bit stream undergoes the serial-to-parallel conversion with an output symbol stream 32-bit wide (8 symbols of 4-bit each). This serves as the message to the (15,8) Reed-Solomon coder. The coder adds additional symbols and makes it 15 symbols widemakes it an output of 60 bits. These 60 bits are then passed through a parallel-to-serial converter to give a result of 10-bit streams. These 10 bits are the actual bits of the 1024-QAM Modulation. This 10-bit stream is truncated into two 5-bit streams using slice blocks. This is depicted in Figure 3.Upto the In-phase and Quad-phase Modulator constitute the transmitter. Everything after that comprises the receiver. Figure4 illustrates the In-phase and Quad-phase Demodulator-Detector. Figure 5 shows the algorithm for Matched Filter Detection of the incoming 1024-QAM Modulated waveform. After the Demodulator is the Detector section. After this, there is a serial-to-parallel bit stream with the production of 60-bit streams. This 60-bit stream is injected into the Reed-Solomon Decoder block. The decoder performs the decoding, corrects up to three symbols of error (12 bits maximum), removes the parity bits, and gives an output of 8 symbols (32 bits wide). These 32 bits are the message bits that are then input to the parallel-to-serial converter, showing a result of single-bit streams. This bit stream constitutes the output of the 1024-QAM Model subsystem, which is given to the Vector Scope for viewing the output bit stream. This output bit stream is compared with the input bit stream to discover possible errors. Since no noise is added, these bit streams should match completely. The RS-Coder-Decoder is realized onto a VHDL file and imported into the Simulink Model using the Black-Box functionality of the Matlab-Simulink. This functionality converts the VHDL file into the Matlab Function into the Simulink Model for smooth functioning. Before this is done, the VHDL file should be Synthesized and Implemented using Xilinx Vivado Tool.

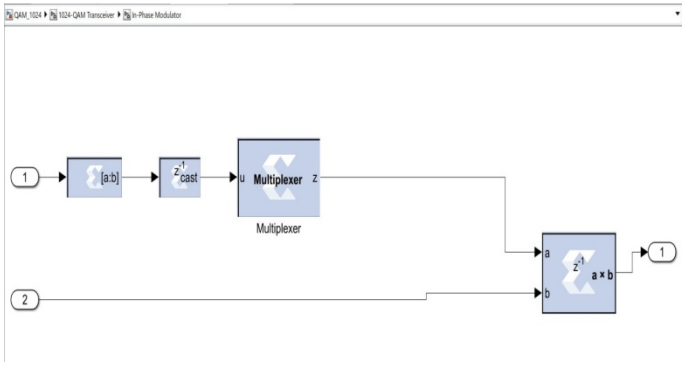

Fig. 3. The In-phase and Quadrature phase 1024-QAM Multiplexer / Modulator

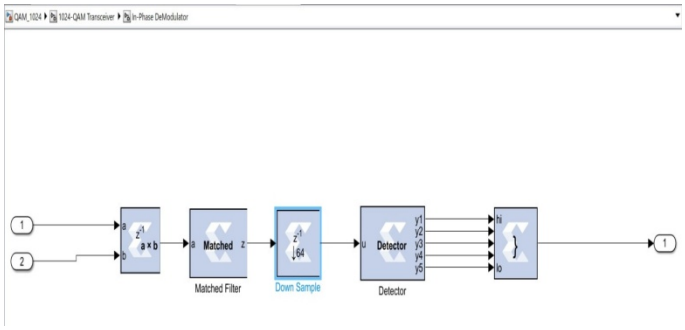

Fig. 4. The In-phase and Quadrature phase Demodulator / Detector

1024 QAM Multiplexer / Modulator: There are two inputs to this subsystem, as shown in Figure 3. One is the Cosine / Sine wave, and the other is the input bit stream of 5-bit each. The is passed through the Multiplexer, which gives an output depicted in Table 1.

```
function [z] = Matched(a)
%MATCHED Summary of this function goes here
%   Detailed explanation goes here

proto = {xlSigned, 70, 50};
proto1 = {xlUnsigned, 6, 0};
persistent sum, sum = xl_state(0.0, proto);
persistent count, count = xl_state(0,proto1);
persistent b, b = xl_state(0.0,proto);
persistent c, c = xl_state(1.8,proto);
persistent d, d = xl_state(0.0323, proto);

b = xfix(proto, a);
sum = xfix(proto, (sum + b));
c = xfix(proto,d*sum);
z = c;
disp(z);
count = count + 1;
if(count == 64)
    count = 0;
    sum = xfix(proto, 0.0);
end

end
```

Fig. 5. The Matched Filter Algorithm for the Demodulator

Table 1. In-phase/ Quad-phase 1024-QAM Multiplexing

Input bits (Gray Coded)	Output Amplitude
01000	-155
01001	-145
01011	-135
01010	-125
01110	-115
01111	-105
01101	-95
01100	-85
00100	-75
00101	-65
00111	-55
00110	-45
00010	-35
00011	-25
00001	-15
00000	-5
10000	+5
10001	+15
10011	+25
10010	+35
10110	+45
10111	+55
10101	+65
10100	+75
11100	+85
11101	+95
11111	+105
11110	+115
11010	+125
11011	+135
11001	+145
11000	+155

The Gray coded bits ensure that the BER is not that severe. After the Multiplexer, a multiplier multiplies the amplitude generated in the Multiplexer, as shown in Table 1. The Cosine / Sine wave generated is with unit amplitude. This is called the Modulation, and the result is that the phase and amplitude of the Cosine / Sine wave are modified according to Table 1. The output is the Modulated Cosine / Sine wave carrier. This is then directly fed to the receiver, where it first encounters the Demodulator and, subsequently, the Detector, which is described in the next section.

Demodulation and Detection: The input signal to the receiver is first demodulated, which is performed by multiplying the input waveform by the carrier generated in the receiver. This carrier here may not be phase-locked but of the same amplitude as the one developed in the transmitter. This is then passed through the digital Matched Filter, where the filter is sampled at the interval of 64 samples (one cycle of the sine/cosine waves consists of 64 samples). This sampled component has all the information about the symbol transmitted. After the sample is acquired, it is passed through the Detector, where the input amplitude is converted back to the bits. The in-phase and quad-phase Demodulator gives out five bits each to be concatenated to make it a 10-bit wide symbol. The Detector is depicted in Table 2. The Matched Filter algorithm is depicted in Figure 5.

Table 2. The Detector (DeMultiplexer) – In-Phase & Quad-Phase

Input Amplitude to Detector (A)	Output bits Generated
A <= -150	01000
-150< A <= -140	01001
-140< A <= -130	01011
-130< A <= -120	01010
-120< A <= -110	01110
-110< A <= -100	01111
-100< A <= -90	01101
-90< A <= -80	01100
-80< A <= -70	00100
-70< A <= -60	00101
-60< A <= -50	00111
-50< A <= -40	00110
-40< A <= -30	00010
-30< A <= -20	00011
-20< A <= -10	00001
-10< A <= 0	00000
0 < A <= +10	10000
+10< A <= +20	10001
+20< A <= +30	10011
+30< A <= +40	10010
+40< A <= +50	10110
+50 < A <= +60	10111
+60< A <= +70	10101
+70< A <= +80	10100
+80< A <= +90	11100
+90< A <= +100	11101
+100< A <= +110	11111
+110< A <= +120	11110
+120< A <= +130	11010
+130< A <= +140	11011
+140< A <= +150	11001
+150< A	11000

(15, 8) Reed-Solomon Coding

Reed-Solomon codes are non-binary cyclic codes with symbols of m-bit sequences, where m is a positive integer with a value greater than 2 (10). R-S (n,k) codes on m-bit symbols exist for all n and k, for which,

$$0 < k < n < 2^m + 2 \quad (1)$$

Where k is the number of data symbols encoded, and n is the number of code symbols in the encoded block (10). For the most conventional R-S (n, k) code (10),

$$(n, k) = (2^m - 1, 2^m - 1 - 2t) \quad (2)$$

Where t is the symbol-error correcting capability of the code, and n-k = 2t is the number of parity symbols.

In this work, n = 15, k = 8, and t = 3. Hence, we can say that this R-S code can correct three symbols of errors. Each symbol is 4bits in length. If the data is in chunks of 32bits, we can say that each byte or 8-bit fed to the R-S coder istwo symbols wide. The algorithm for generating the additional seven symbols of parity is described in the following paragraphs. To understand the encoding and decoding process of the R-S codes, it is necessary to venture into the area of

finite fields known as Galois Fields (GF). For any prime number, p, a limited field denoted GF (p) contains p elements. Here, p = 2m, represented by a power of α. An infinite set of components, F, is formed by starting with details {0, 1, α} and generating the additional elements by progressively multiplying the last entry by α, which yields the following:

$$F = \{0,1, \alpha, \alpha^2, \ldots \alpha^j \ldots .\} \quad (3)$$

Here, a finite set of elements of GF (2^m) is formed from F. In this application, m=4; hence, there should be 16 elements in F. A class of polynomials called primitive polynomials is interesting because such functions define the finite fields GF (2^m) needed to determine R-S codes. Here, in this case, the bits per symbol are four; hence the primitive polynomial is:

$$1 + x + x^4 = 0 \quad (4)$$

Putting x = α and rearranging, we get,

$$-1 - \alpha = \alpha^4 \quad (5)$$

Since in binary filed -1 = +1 we can write,

$$1 + \alpha = \alpha^4 \quad (6)$$

From Equation (6), we can derive all the elements of GF (2^m), which are:

$$\{0, \alpha^0, \alpha^1, \alpha^2, \alpha^3, \alpha^4, \alpha^5, \alpha^6, \alpha^7, \alpha^8, \alpha^9, \alpha^{10}, \alpha^{11}, \alpha^{12}, \alpha^{13}, \alpha^{14}\}$$

The symbols are depicted in Table 3.

The multiplication table is given in Tables 4 and 5 for m=4 and primitive polynomial $1+x+x^4$ (all values are powers of α)

The addition table is shown in Tables 6 and 7:

Table 3. Symbols Table

Field Elements	Basic elements			
Powers of α	X^3	X^2	X^1	X^0
0	0	0	0	0
α^0	0	0	0	1
α^1	0	0	1	0
α^2	0	1	0	0
α^3	1	0	0	0
α^4	0	0	1	1
α^5	0	1	1	0
α^6	1	1	0	0
α^7	1	0	1	1
α^8	0	1	0	1
α^9	1	0	1	0
α^{10}	0	1	1	1
α^{11}	1	1	1	0
α^{12}	1	1	1	1
α^{13}	1	1	0	1
α^{14}	1	0	0	1
α^{15}	0	0	0	1

From α^{15}onward, the symbols repeat.

Table 4. Multiplication Table

	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6
1	1	2	3	4	5	6	7
2	2	3	4	5	6	7	8
3	3	4	5	6	7	8	9
4	4	5	6	7	8	9	10
5	5	6	7	8	9	10	11
6	6	7	8	9	10	11	12
7	7	8	9	10	11	12	13
8	8	9	10	11	12	13	14
9	9	10	11	12	13	14	0
10	10	11	12	13	14	0	1
11	11	12	13	14	0	1	2
12	12	13	14	0	1	2	3
13	13	14	0	1	2	3	4
14	14	0	1	2	3	4	5

Table 5. Multiplication Table

	7	8	9	10	11	12	13	14
0	7	8	9	10	11	12	13	14
1	8	9	10	11	12	13	14	0
2	9	10	11	12	13	14	0	1
3	10	11	12	13	14	0	1	2
4	11	12	13	14	0	1	2	3
5	12	13	14	0	1	2	3	4
6	13	14	0	1	2	3	4	5
7	14	0	1	2	3	4	5	6
8	0	1	2	3	4	5	6	7
9	1	2	3	4	5	6	7	8
10	2	3	4	5	6	7	8	9
11	3	4	5	6	7	8	9	10
12	4	5	6	7	8	9	10	11
13	5	6	7	8	9	10	11	12
14	6	7	8	9	10	11	12	13

Table 6. Addition Table

	0	1	2	3	4	5	6
0	zero	4	8	14	1	10	13
1	4	Zero	5	9	0	2	11
2	8	5	Zero	6	10	1	3
3	14	9	6	Zero	7	11	2
4	1	0	10	7	Zero	8	12
5	10	2	1	11	8	Zero	9
6	13	11	3	2	12	9	Zero
7	9	14	12	4	3	13	10
8	2	10	0	13	5	4	14
9	7	3	11	1	14	6	5
10	5	8	4	12	2	0	7
11	12	6	9	5	13	3	1
12	11	13	7	10	6	14	4
13	6	12	14	8	11	7	0
14	3	7	13	0	9	12	8

Table 7. Addition Table

	7	8	9	10	11	12	13	14
0	9	2	7	5	12	11	6	3
1	14	10	3	8	6	13	12	7
2	12	0	11	4	9	7	14	13
3	4	13	1	12	5	10	8	0
4	3	5	14	2	13	6	11	9
5	13	4	6	0	3	14	7	12
6	10	14	5	7	1	4	0	8
7	Zero	11	0	6	8	2	5	1
8	11	Zero	12	1	7	9	3	6
9	0	12	Zero	13	2	8	10	4
10	6	1	13	Zero	14	3	9	11
11	8	7	2	14	Zero	0	4	10
12	2	9	8	3	0	Zero	1	5
13	5	3	10	9	4	1	Zero	2
14	1	6	4	11	10	5	2	Zero

It may be emphasized here that all the values in the above tables depict powers of α.

The next step is to find the generator polynomial, which is given as:

$$g(X) = g_0 + g_1 X + g_2 X^2 + g_3 X^3 + \cdots + g_{2t-1} X^{2t-1} + X^{2t} \quad (7)$$

The degree of the generator polynomial equals the number of parity symbols. We describe the generator polynomial in terms of its 2t = n-k = 7 roots as follows:

$$g(X) = (X - \alpha)(X - \alpha^2) \ldots . (X - \alpha^7) \quad (8)$$

Equation (8) can be simplified using Table 3 and modulo two additions. All the negative signs are converted into positive signs. We get,

$$g(X) = X^7 + \alpha^6 X^6 + \alpha^{13} X^5 + \alpha^{12} X^4 + \alpha^1 X^3 + \alpha^{10} X^2 + \alpha^{11} X + \alpha^{13} \quad (9)$$

Next, we find out the parity polynomial, which is given as,

$$p(X) = X^{n-k} m(X) modulog(X) \quad (10)$$

Where m(X) is the message polynomial and (n-k) = 7. Hence, the codeword becomes:

$$U(X) = p(X) + X^{n-k} m(X) \quad (11)$$

Reed-Solomon Decoding: Error is introduced into the codeword after the codeword is transmitted over a frequency-selective fading channel, e(X). Hence, the received code polynomial is,

$$r(X) = U(X) + e(X) \quad (12)$$

To remove the errors, a maximum of 3 symbols or 12 bits in 15 symbols or 60 bits of the codeword syndrome is calculated, which is given as,

$$S_i = r(X)|_{X-\alpha^i} = r(\alpha^i) \quad (13)$$

Where i = 1, …, n-k. Hence, seven syndrome values are calculated using Equation (13) and Table 3. If all the syndrome values are zero, it signifies that there is no error. If some of the syndrome values are non-zero, there is an error.

The next step is to find the error location. The error locator polynomial σ(X) can be described as follows:

$$\sigma(X) = (1 + \beta_1 X)(1 + \beta_2 X) \dots (1 + \beta_v X) \quad (14)$$

$$\sigma(X) = 1 + \sigma_1 X + \sigma_2 X^2 + \cdots + \sigma_v X^v \quad (15)$$

We form the matrix from the syndromes as,

$$\begin{bmatrix} S_1 & S_2 & S_3 \\ S_2 & S_3 & S_4 \\ S_3 & S_4 & S_5 \end{bmatrix} \begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \sigma_3 \end{bmatrix} = \begin{bmatrix} S_4 \\ S_5 \\ S_6 \end{bmatrix} \quad (16)$$

Next, we find the values of σ_1, σ_2, σ_3 by finding the inverse of the matrix on the left-hand side. If the determinant of the matrix on the left-hand side is zero, there may be fewer symbol errors than 3. Then, we form a matrix for two symbol errors,

$$\begin{bmatrix} S_1 & S_2 \\ S_2 & S_3 \end{bmatrix} \begin{bmatrix} \sigma_1 \\ \sigma_2 \end{bmatrix} = \begin{bmatrix} S_3 \\ S_4 \end{bmatrix} \quad (17)$$

Next, we find the inverse of the matrix on the left-hand side. If still, the determinant is zero, which signifies one symbol error. Then, we are left with S_1. If S_1 is zero, it suggests more than three symbol errors. It is emphasized here that the determinant is checked with the matrices in Equations16 and 17 in that order only. Hence, if there are more than three symbol errors, they cannot be corrected as there is no way of finding and fixing more than three symbol errors in R-S (15, 8) codes. Once we find out the value of σ_1, σ_2, σ_3 from Equation (16), we can write,

$$\sigma(X) = 1 + \sigma_1 X + \sigma_2 X^2 + \sigma_3 X^3 \quad (18)$$

From Equation (18), we can find out the error locations as,

$$\sigma(\alpha^i) = 1 + \sigma^1(\alpha^i)_1 + \sigma^2(\alpha^i)_2 + \sigma^3(\alpha^i)_3 \quad (19)$$

Where i = 0, 1, 2, ……, 14.

From the values obtained in Equation (19), we can say there is an error wherever the value of $\sigma(\alpha^i)$ is zero. Now, at this stage, we have found out the error locations. The next step is to find out the error values, which are calculated as,

$$S_1 = r(\alpha) = e_1\beta_1 + e_2\beta_2 + e_3\beta_3 \quad (20)$$

$$S_2 = r(\alpha^2) = e_1\beta_1^2 + e_2\beta_2^2 + e_3\beta_3^2 \quad (21)$$

$$S_3 = r(\alpha^3) = e_1\beta_1^3 + e_2\beta_2^3 + e_3\beta_3^3 \quad (22)$$

It can be written in matrix form as,

$$\begin{bmatrix} \beta_1 & \beta_2 & \beta_3 \\ \beta_1^2 & \beta_2^2 & \beta_3^2 \\ \beta_1^3 & \beta_2^3 & \beta_3^3 \end{bmatrix} \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} = \begin{bmatrix} S_1 \\ S_2 \\ S_3 \end{bmatrix} \quad (23)$$

To find the value of e_1, e_2, and e_3, we first see the inverse of the matrix on the left-hand side. Once we get these values, we go in for the symbol error correction. It may be emphasized here that R-S codes do whole symbol correction whether there is a single-bit or multiple-bit error in the symbol. The values of these error codes are modulo two,added with the symbols at the corresponding locations, which remove the errors.

$$\hat{e}(X) = e_1 X^{j_1} + e_2 X^{j_2} + e_3 X^{j_3} \quad (24)$$

Where j_1, j_2, and j_3 are the error locations.

$$\hat{U}(X) = r(X) + \hat{e}(X) \quad (25)$$

The whole process is performed for 2-symbol or 1-symbol errors similarly. The difference is that instead of 3 symbol errors, the location of 2 or 1 symbol error is obtained.

Hardware Implementation and Real-Time Results: The hardware implementation of the 1024-QAM Transceiver is performed onto the Kintex-7 FPGA in the Hardware-Software Co-Simulation using System Generator. The System Generator parameters are illustrated in Figures 6 and 7. Figure 7 shows that the FPGA clock period is one nanosecond, and the Simulink System Period is 1 / fs_fpga where fs_fpga = 1MHz. The DDS compiler settings are illustrated in Figure 8. This Figure shows that the waveform generated is Cosine and Sine Simultaneous. The System Clock = 1 MHz, which is reciprocal of the Simulink System Period. The Real-Time results for the 1024-QAM transceiver are illustrated in Figures 9 and 10. The Figure shows that the top bit stream is the transmitted bitstream, and the bottom bitstream is the received bitstream. Here, it is observed that the transmitted and the received bitstream match except for a minor latency, which is obvious. The Hardware-Software Co-simulation Model of the Transceiver is depicted in Figure 11.

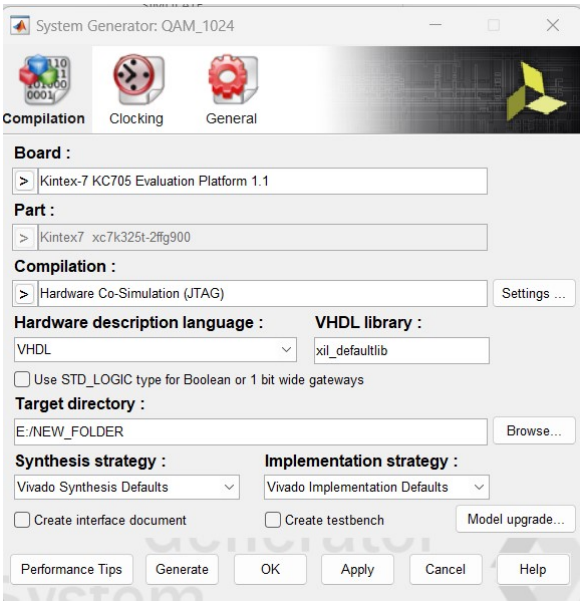

Fig. 6. The System Generator Compilation

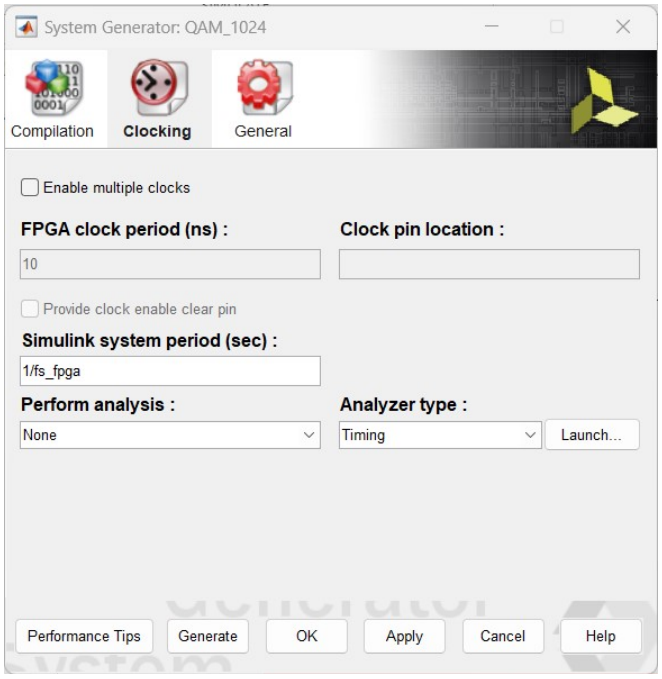

Fig. 7. The System Generator Clocking

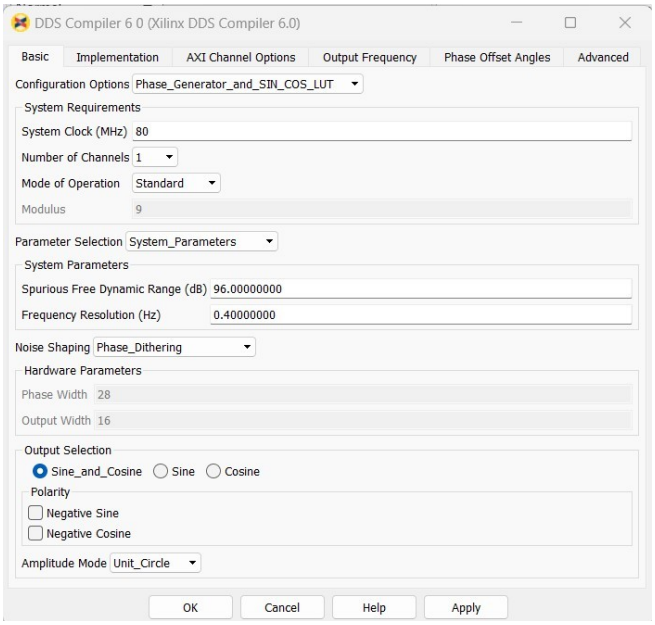

Fig. 8. DDS Compiler Basic settings

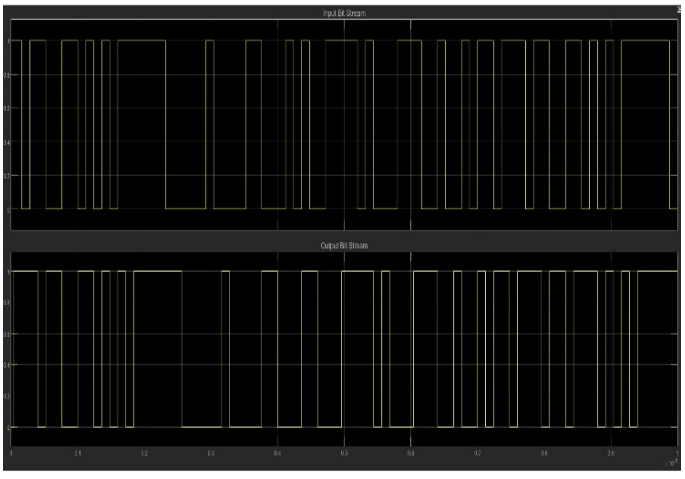

Fig. 9. Real Time Results – View I

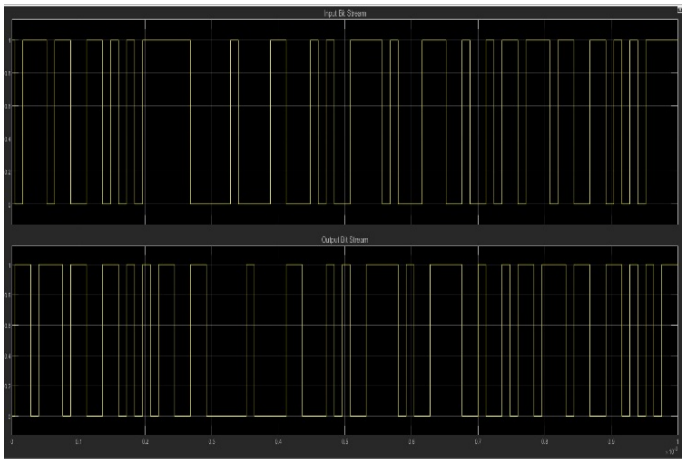

Fig. 10. Real-Time Results – View II

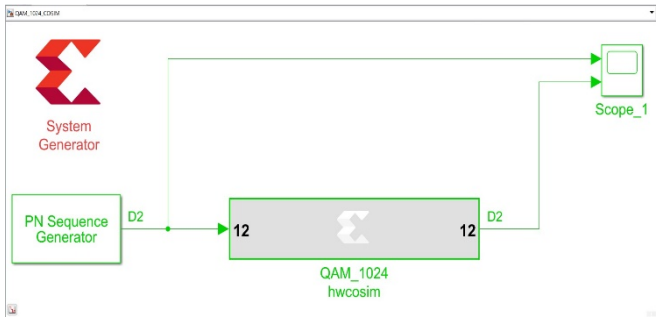

Fig. 11. Hardware-Software Cosimulation Model

CONCLUSION

The 1024-QAM transceiver is designed and implemented using the MATLAB / SIMULINK Model. The Forward Error correction used here is (15,8) Reed-Solomon Coding-Decoding, which works perfectly well. The Demodulator used in the receiver portion is the Matched Filter Demodulation and the Detection. The Modulator and Demodulator are working fine. The real-time results show that the transmitted bitstream and the received bitstream perfectly match each other except for a minor latency.

REFERENCES

1. Iwamoto, M. S. Matsuoka, H. Iwasaki and H. Otsuka, "Transmission performance of OFDM with 1024-QAM in the presence of EVM degradation," 2014 IEEE Asia Pacific Conference on Wireless and Mobile, Bali, Indonesia, 2014, pp. 12-16, doi: 10.1109/APWiMob.2014.6920262.
2. Kim, S. -H., J. Choi, J. Kim, M. S. Hossen and K. -D. Kim, "Design and implementation of the 1024-QAM RF transmission system for UHD cable TV broadcasting," 16th International Conference on Advanced Communication Technology, Pyeongchang, Korea (South), 2014, pp. 1317-1324, doi: 10.1109/ICACT.2014.6779173.
3. Lu E. *et al*., "10.4 A 4×4 Dual-Band Dual-Concurrent WiFi 802.11ax Transceiver with Integrated LNA, PA and T/R Switch Achieving +20dBm 1024-QAM MCS11 Pout and −43dB EVM Floor in 55nm CMOS," 2020 IEEE International Solid- State Circuits Conference - (ISSCC), San Francisco, CA, USA, 2020, pp. 178-180, doi: 10.1109/ISSCC19947.2020.9063127.
4. McCune, E. "Precision and Efficient 1024-QAM Transmitter Based on the Switch-mode Mixer Modulator," 2019 IEEE Radio and Wireless Symposium (RWS),

Orlando, FL, USA, 2019, pp. 1-4, doi: 10.1109/RWS.2019.8714465.

5. Nakamura, N., T. Kurakake, K. Ovamada and S. Itoh, "Performance evaluation of 1024 QAM signal transmission in cable television distribution systems by software demodulation method," The 8th International Conference on Communication Systems, 2002. ICCS 2002., Singapore, 2002, pp. 534-538 vol.1, doi: 10.1109/ICCS.2002.1182532.R. Nicole, "Title of paper with only first word capitalized," J. Name Stand. Abbrev., in press.
6. Omura, Y. F. Kemmochi, K. Fujisawa and H. Otsuka, "Implementation of MCS Incorporating 1024-QAM and Beam-Based Transmission in 3D-BF," 2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring), Antwerp, Belgium, 2020, pp. 1-4, doi: 10.1109/VTC2020-Spring4859 0.2020.9128907.
7. Ota, T. M. Nakamura and H. Otsuka, "Performance evaluation of OFDM-based 256- and 1024-QAM in multipath fading propagation conditions," 2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN), Milan, Italy, 2017, pp. 554-556, doi: 10.1109/ICUFN.2017.7993849.M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.
8. S. Kawai *et al*., "A 1024-QAM capable WLAN receiver with −56.3 dB image rejection ratio using self-calibration technique," 2017 IEEE International Symposium on Circuits and Systems (ISCAS), Baltimore, MD, USA, 2017, pp. 1-4, doi: 10.1109/ISCAS.2017.8050600.
9. Senda K. and H. Otsuka, "Transmission Performance of Superposed Modulation Using QPSK and 1024-QAM in Downlink NOMA," 2019 IEEE VTS Asia Pacific Wireless Communications Symposium (APWCS), Singapore, 2019, pp. 1-5, doi: 10.1109/VTS-APWCS.2019.8851620.I. S. Jacobs and C. P. Bean, "Fine particles, thin films and exchange anisotropy," in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.
10. Sklar, Bernard, “Reed-Solomon Codes”, hscc.cs.nthu.tw.

****