Design and Implementation of a Multiplier free FPGA based OFDM Transmitter

Bashar Adel Esttaifan
(M. Sc., Assist. Lec.)
Electronics and Communications Eng. Dept.
Collage of Engineering - University of Baghdad
bashar_stephan@yahoo.com

Oday AbdulLateef AbdulRidha (Ph. D., Lec.)
Electronics and Communications Eng. Dept.
Collage of Engineering - University of Baghdad
oday_ridha@yahoo.com

Waleed Ameen Mahmoud
(Ph. D., Prof.)
Electrical Eng. Dept.
Collage of Engineering - University of Baghdad
profwaleed54@yahoo.com

ABSTRACT

Orthogonal Frequency Division Multiplexing (OFDM) is an efficient multi-carrier technique. The core operation in the OFDM systems is the FFT/IFFT unit that requires a large amount of hardware resources and processing delay. The developments in implementation techniques likes Field Programmable Gate Array (FPGA) technologies have made OFDM a feasible option. The goal of this paper is to design and implement an OFDM transmitter based on Altera FPGA using Quartus software. The proposed transmitter is carried out to simplify the Fourier transform calculation by using decoder instead of multipliers. After programming ALTERA DE2 FPGA kit with implemented project, several practical tests have been done starting from monitoring all the results of the implemented blocks (VHDL code) and compare them with corresponding results from simulation system implemented in matlab 2010a. The results of these practical tests show that the suggested approach gives a significant improvement in reducing complexity and processing delays (45 nsec) in comparison with the conventional implementations of OFDM transmitter.

KEYWORDS: OFDM transmitter implementation; IFFT implementation; IDFT implementation, FPGA.

تصميم و بناء مرسلة بدون (Multiplier) تعتمد نظام مزج تقسيمات التردد المتعامدة باستعمال تقنية مصفوفة البوابات المنطقية القابلة للبرمجة

م.م. بشار عادل أسطيفان
قسم الهندسة الالكترونية و الاتصالات
كلية الهندسة – جامعة بغداد
بغداد – العراق

م.د. عدي عبد اللطيف عبد الرضا
قسم الهندسة الالكترونية و الاتصالات
كلية الهندسة – جامعة بغداد
بغداد – العراق

أ.د. وليد امين محمود الجوهر
قسم الهندسة الكهربائية
كلية الهندسة – جامعة بغداد
بغداد – العراق

الخلاصة

مزج تقسيمات التردد المتعامدة هو تقنية فعالة من التقنيات المتعددة النواقل. العملية الاساسية في نظام مزج تقسيمات التردد المتعامدة هي عملية تحويل فورير (و عكسه) والتي تتطلب وقت معالجة كبير و كثير من الدوائر الالكترونية. التطور في تقنيات البناء كتقنية مصفوفة البوابات المنطقية القابلة للبرمجة و التي جعلت نظام مزج تقسيمات التردد المتعامدة خيارا متاحا و مفضلا. الهدف المنشود في هذا البحث هو تصميم و بناء مرسلة تعتمد تقنية مزج تقسيمات التردد المتعامدة اعتمادا على مصفوفة البوابات المنطقية القابلة للبرمجة باستعمال برنامج Quartus. المرسل المفترح يُبَسط حسابات تحويل فورير باستعمال (Decoder) واحد بدل من جميع الـ (Multipliers). بعد برمجة جهاز (ALTERA DE2 FPGA) بالمشروع الذي تم بناءه، اجرينا عدة اختبارات عملية، ابتداءا من مراقبة النتائج من جميع الدوائر التي تم بنائها (بواسطة لغة VHDL) و مقارنتها مع ما يناظرها من نتائج النظام المحاكي المبني بواسطة برنامج (matlab 2010a). أظهرت النتائج العملية ان الطريقة المقترحة اعطت تحسنا كبيرا في تقليل تعقيد النظام و زمن المعالجة مقارنة (45 نانو ثانية) بالانظمة التقليدية والخاصة ببناء نظم مزج تقسيمات التردد المتعامدة.

كلمات رئيسية: بناء مازج تقسيمات التردد المتعامدة، بناء تحويل فورير السريع، بناء تحويل فورير العادي، مصفوفة البوابات المنطقية القابلة للبرمجة.

I. INTRODUCTION

The core block of OFDM transmitter is the transform block (IDFT or IFFT) because it takes the most hardware resources that used to implement OFDM system. Furthermore this block required relatively large processing time. To minimize processing time as well as complexity of the OFDM transmitter the transform calculation must be developed.

A. DFT and FFT

A complete direct calculation of a N-point DFT requires $(N–1)^2$ complex multiplications and N×(N–1) complex additions. It can be seen that the computational complexity is in the order of N^2. For N >8, direct calculation of the IDFT is too computational intensive and not practical for implementation in hardware as listed in **Table 1**. So the idea of FFT (i.e. IFFT) is brought forward. The total number of complex multiplications is reduced to ($N/2 \log_2 N$) in FFT and the total number of complex addition is reduced to ($N \log_2 N$), but for a large N i.e. (N>32), FFT also be not efficient [Ludman, 1987].

B. DFT and IDFT in Matrix Form [Ludman, 1987]

It is instructive to view the DFT and IDFT as linear transformations on sequences $\{x(n)\}$ and $\{X(k)\}$, respectively. Let us define an N-point vector $\mathbf{x}_N$ of the signal sequence $x(n)$, $n = 0, 1, \ldots, N-1$, and an N-point vector $\mathbf{X}_N$ of frequency samples, and an $N \times N$ matrix $\mathbf{W}_N$ as

$$\mathbf{x}_N = \begin{bmatrix} x(0) \\ x(1) \\ \vdots \\ x(N-1) \end{bmatrix}, \quad \mathbf{X}_N = \begin{bmatrix} X(0) \\ X(1) \\ \vdots \\ X(N-1) \end{bmatrix}$$

$$\mathbf{W}_N = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & W_N & W_N^2 & \cdots & W_N^{N-1} \\ 1 & W_N^2 & W_N^4 & \cdots & W_N^{2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & W_N^{N-1} & W_N^{2(N-1)} & \cdots & W_N^{(N-1)(N-1)} \end{bmatrix} \quad \textbf{eq. (1)}$$

$\mathbf{W}_N$ can be simplified by writing the powers of all elements in modulo-N notation, instead of the previous form, i.e.

$$\mathbf{W}_N = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & W_N & W_N^2 & \cdots & W_N^{N-1} \\ 1 & W_N^2 & W_N^4 & \cdots & W_N^{N-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & W_N^{N-1} & W_N^{N-2} & \cdots & W_N^1 \end{bmatrix} \quad \textbf{eq. (2)}$$

With these definitions, the N-point DFT may be expressed in matrix form as:

$$\mathbf{X}_N = \mathbf{W}_N \mathbf{x}_N, \quad \textbf{eq. (3)}$$

where $\mathbf{W}_N$ is the matrix of the linear transformation. We observe that $\mathbf{W}_N$ is a symmetric matrix. If it is assumed that the inverse of $\mathbf{W}_N$ exists, then eq. (3) can be inverted by pre-multiplying both sides by $\mathbf{W}_N^{-1}$. Thus we obtain:

$$\mathbf{x}_N = \mathbf{W}_N^{-1} \mathbf{X}_N, \quad \textbf{eq. (4)}$$

But this is just an expression for the IDFT. In fact, the IDFT can be expressed in matrix form as:

$$\mathbf{x}_N = \frac{1}{N} \mathbf{W}_N^* \mathbf{X}_N, \quad \textbf{eq. (5)}$$

where $\mathbf{W}_N^*$ denotes the complex conjugate of the matrix $\mathbf{W}_N$. Comparison of eq. (4) with eq. (5) leads to conclude that:

$$\mathbf{W}_N^{-1} = \frac{1}{N} \mathbf{W}_N^* \quad \textbf{eq. (6)}$$

which, in turn, implies that:

$$\mathbf{W}_N \mathbf{W}_N^* = N \mathbf{I}_N \quad \textbf{eq. (7)}$$

where $\mathbf{I}_N$ is an $N \times N$ identity matrix. Therefore, the matrix $\mathbf{W}_N$ in the transformation is an **orthogonal (unitary) matrix**. Furthermore, its inverse exists and is given as $\mathbf{W}_N^*/N$.

II. PROPOSED METHOD TO IMPLEMENT IDFT FOR OFDM TRANSMITTER [Esttaifan, 2011]

For the OFDM transmitter shown in **Fig 1**, the input sequence (data) must be either ‘0’ or ‘1’; the mapper converts the input data to complex symbols according to modulation type. The inputs to the IDFT block are depending on mapper type (modulation type i.e. BPSK, QPSK or QAM). So the circuit that uses to calculate the IDFT can be

simplified according to the limited possible values for mapper output as seen in **Table 2**.

The proposed method can be briefed by using *decoder* instead of *all multipliers to choose between cases according to input data*. The twiddle factors ($\mathbf{W}_N$ = **Re** +i **Im**) must be saved in memory to use them in calculations of IDFT in proposed OFDM transmitter, two ROMs must be used to save twiddle factors ($\mathbf{W}_N$) (one for real **(Re)** and the other for imaginary parts **(Im)**), but in the proposed methods, an extra memory cells must be used to save the scaled twiddle factors ($X\mathbf{W}_N$). The scaled twiddle factors ($X\mathbf{W}_N$) can be explained by the following example.

Let us take an example, where we want to find the result of:

$$W_{16}^{m} * (-3+3i)$$

Where: $(W_{16}^{m} = a + i\,b)$ and
((-3+3i): a point in 16-QAM modulator)

Then the answer must be:

$$(a+ib)*(-3+3i)=([-3*a]-[3*b])+i([3*a]+[-3*b]) \quad \textbf{eq. (8)}$$

According to eq. (8), 4 multipliers and 2 adders must be used to find the result of multiplying two complex numbers given, but if the result of (3a) and (3b) were saved in memory {(3a) and (3b) are represents the **scaled twiddle factors matrix** ($X\mathbf{W}_N$)}, so, only calling them from memory and adding them to find the answer and the complex multiplication be just two adder.

For the above example *(16-QAM modulator)* it is important to save (**a**) and (**b**) with the presence of (**3a**) and (**3b**) matrices in memory too ((**a**) and (**b**) are represents the **conventional twiddle factors matrix** ($\mathbf{W}_N$)), and using decoder for mapping (or selecting) between cases to enable adders to complete calculations.

Fig. 2 represents the proposed OFDM transmitter (including: S/P, mapper, IDFT, cyclic prefix adder, P/S).

- S/P represented by D-FFs; the decoder is represents the mapper, it will control the adder circuit which acts as IDFT block.
- Cyclic prefix adder and P/S are built in control unit block which is also responsible of clock generation, managing the start of transmission, giving the control signals (read, write and addresses) to all memories and synchronizing between all blocks.
- Each adder block is responsible of finding only one IDFT element at output, this element is the resultant of adding or subtracting N various twiddle factors according to IDFT matrix. It is more suitable to rename adder block to *Cumulative Adder Block* (CAB). Each CAB has two adders work concurrently, one to calculate real part and the other for imaginary.
- Every Cumulative Adder Block has two registers to save the cumulative results, one register to save real part and the other to save imaginary part. The length of each register is equal to the precision of twiddle factors (p). Initially, the contents of the two registers will be zeros, then CAB call these contents and add (or subtract) them with the new data and the results will be written on the registers.
- The processing algorithm in this proposed method has two fields, first, pipeline processing which is done by each cumulative adder block internally to find individual (each one) IDFT output, this processing algorithm will done N times; secondly, concurrent processing, it is done by the N cumulative adders blocks that works in parallel.
- D-FFs and Decoder will work serially, also Control unit and memory works serially but they are concurrently processing with respect to D-FF and Decoder together so the processing time of the two groups must be calculated and the larger one must be taken in account only, and must be added to the delay of one cumulative adder block (all adders are parallel to each other) to obtain the total processing time for OFDM transmitter.

III. COMPLEXITY AND TIME CALCULATIONS FOR PROPOSED OFDM TRANSMITTER [Esttaifan, 2011]

The internal construction of D-FF, decoder, Memory, CAB and control unit (**Fig. 2**) will used to calculate processing time.

TABLE 3 will summarize the number of blocks and processing time for a specific number of IDFT The total processing time can be written:

$$P_T=\{\text{Larger of } [(P_{D\text{-}FFs}+P_{decoder}), (P_{Memory}+P_{control})]\}+P_{CAB} \quad \text{eq. (9)}$$

Where:

P_T = total processing time in G.
$P_{D\text{-}FFs}$= processing time due to D-FFs = delay due to one D-FF× number of D-FFs = $2G*([\log_2(M)]-1)$.

$P_{decoder}$= processing time due to decoder = 2G.
P_{Memory}= processing time due to memory = 3G.
$P_{control}$= processing time due to control unit = 3G.
P_{CAB}= processing time due to Cumulative Adder Block = 3G.

The complexity of DAC and ADC is reduced because no multiplier used, when multiplying data (16 bit length) with twiddle factor (16 bit length) the result must be 30 bit length so it was not efficient to use converters with 30 bit complexity.

IV. IMPLEMENTED OFDM TRANSMITTER [Esttaifan, 2011]

To be familiar with this proposed method an example must be used to prove its efficiency against other methods that used to implement OFDM transmitter, it assumed QPSK (M=4) modulator as mapper, IFFT length (N) will be 16 and cyclic prefix added was 4, so in **Fig. 3**, according **Table 3**:

One D-FF may be used to store data, decoder (2:4) to select one of the four cases according to the input came from D-FF output (Q) and the other input before D-FF (Q+1) so the Decoder must enabled once every two clock.

If the input x(n) = [1 1 1 1 1 1 0 0 1 0 0 1 1 1 0 1 1 0 1 0 0 0 0 1 0 1 1 0 0 0 0 1].
After first clock to D-FF, in1 and in2 (two inputs of Decoder in **Fig. 3**) will be 1 and 1 respectively so that Op1, Op2 and Op3 must be inactive i.e. '0' and Op4 will be active '1' which is represents (-1+0i) in QPSK constellation (from Decoder truth point (N) in addition of number of point in constellation (mapper type) (M), gate propagation delay (G) and the precision of twiddle factor (P). table: **Table 4**). The Cumulative adder blocks are modified to represent the four cases of mapper:

- Case 1: In1= '0', In2 = '0', which mean +1+0i in QPSK constellation
  $(+1+0i) * W_{16}^{m} = (+1+0i) * (a + bi) = \underline{a + bi}$
- Case 2: In1= '1', In2 = '0', which mean 0+i in QPSK constellation
  $(0+i) * W_{16}^{m} = (0+i) * (a + bi) = \underline{-b + ai}$
- Case 3: In1= '0', In2 = '1', which mean 0-i in QPSK constellation
  $(0-i) * W_{16}^{m} = (0-i) * (a + bi) = \underline{b - ai}$
- Case 4: In1= '1', In2 = '1', which mean -1+0i in QPSK constellation
  $(-1+0i) * W_{16}^{m} = (-1+0i) *(a + bi) = \underline{-a - bi}$

So that only the conventional twiddle factors matrices (**a** and **b**) are needed in calculations (in QPSK), the memories (ROM) attached to first cumulative adder block must contain the first row of IDFT matrix, at the same manner the memories attached to second cumulative adder block must contain the second row of IDFT matrix and so on.
For the given x(n) in this example, **Table 5** shows the contents of two registers that attached to first CAB to save the cumulative results, the contents of registers must be updated after each input (2 bit for QPSK), where a_1 to a_{16} are the real parts of the twiddle factors for the first row in IDFT matrix and b_1 to b_{16} are the imaginary parts of the twiddle factors for the first row in IDFT matrix.

V. COMPARISON OF THEORETICAL AND PRACTICAL RESULTS OF IMPLEMENTED OFDM TRANSMITTER [Esttaifan, 2011]

A comparison must be done between theoretical results obtained in Matlab and practical results calculated by implemented system using FPGA.
It assumed the previous example, QPSK modulator as mapper (M=4) and IFFT length (N) will be 16 and the input data:
x(n) = [1 1 1 1 1 1 0 0 1 0 0 1 1 1 0 1 1 0 1 0 0 0 0 1 0 1 1 0 0 0 0 1]
Fig. 4 shows the schematic diagram of the implemented OFDM transmitter generated by Quartus 10.0. We use hierarchal methodology to build this system.

The difference (error) between matlab results and the proposed system did not exceed (0.001) at worst cases which is clearly shown in Table 6.

A monitoring program was implemented inside FPGA as well as OFDM transmitter to show IDFT results for any input sequence.
This program used Red LEDs numbered from **LEDR0** to **LEDR14** to view any element; to show an element is just to select its sequence by using switches numbered from **SW0** to **SW4 (from 0000 to 1111)**. By using the monitoring program, the result x(n) can be shown in **Fig. 5**.

VI. COMPARISON BETWEEN THE PROPOSED METHOD AND OTHERS

We implemented OFDM transmitter with the following parameters to compare the results with other official and standard cores, N=256, with BPSK mapper (M = 2), according to equation (9) and the obtained practical results the total propagation delay is:

$$P_T = \{\text{Larger of } [(0G+3G), (3G+3G)]\}+3G = 9G$$

If the gate propagation delay (G) is typically assumed to be 5 nsec then

$$P_T = 9 * 5 \text{ nsec} = 45 \text{ nsec}$$

This 45 nsec will be the processing time(P_T)for any number of IDFT length (N) after receiving the final bit of data for BPSK mapper (M=2) because P_T did not depend on N but it related to ***mapper type (M)*** *as shown in TABLE 3 and eq(9).*

A comparison between multi - IDFT (or IFFT) cores with respect to complexity was carried out in **Table 7** from practical implementations.

VII. CONCLUSIONS

1. The number of complex multipliers in the proposed method that used in transmitter was ZERO so that the processing time significantly decreased.
2. The number of complex adders in transmitter was reduced because of the pipelining in calculations.
3. The complexity (number of bits) in ADC and DAC had reduced.
4. The length of IDFT (in OFDM system) available to be implemented in any FPGA was increased because there is no limit due to number of multiplier.

VIII. REFERENCES

[1]. Altera Datasheet, "FFT Mega-Core function user guide", Version 10, TABLE 1-4, TABLE 1-10, Altera Inc., Dec. 2010.

[2]. Bashar Adel Esttaifan, "Design and Implementation of OFDM Modem based on FPGA", M.Sc. Thesis, University of Baghdad- Iraq, 2011.

[3]. Jesús García1, Juan A. Michell, Gustavo Ruiz, Angel M. Burón "FPGA realization of a Split Radix FFT processor", Dept. de Electrónicay Computadores, Facultad de Ciencias, Univ. de Cantabria, Avda. Los Castros s/n, 39005 Santander, spain. Proc. of SPIE Vol. 6590 65900P-1, 2007.

[4]. L.C. Ludman, "Fundamental of digital signal processing", 2nd edition, John Wiley & Sons, 1987.

[5]. Xilinx Datasheet, "Xilinx Logic Core Fast Fourier Transform", Version 4.1, Xilinx Inc., Feb. 2007.

List of Symbol

Symbol	Notation
G	= Gate propagation delay
M	= Number of symbols in the QAM
N	= Number of modulated symbols; length of FFT sequence; number of serial data elements; dimension of a square packet
p	= Data precision
P_{CAB}	= Processing time due to Cumulative Adder Block
$P_{control}$	= Processing time due to control unit
P_{decode}	= Processing time due to decoder
$P_{D\text{-}FFs}$	= Processing time due to D-FFs
P_{Memory}	= Processing time due to memory
P_T	= Total processing time
W_N	= Twiddle factor
X(K)	= N-point set of equally-separated frequency samples (FFT output)
x(n)	= N-point time sequence $x(n)$ (FFT input)

List of Abbreviations

ADC	**A**nalog to **D**igital **C**onverter
CAB	**C**umulative **A**dder **B**lock
DAC	**D**igital to **A**nalog **C**onverter
FFT	**F**ast **F**ourier **T**ransform
FPGA	**F**ield **P**rogrammable **G**ate **A**rray
IDFT	**I**nverse **D**iscrete **F**ourier **T**ransform
IFFT	**I**nverse **F**ast **F**ourier **T**ransform
LE	**L**ogic **E**lement
LUT	**L**ook **U**p **T**able
OFDM	**O**rthogonal **F**requency **D**ivision
PSK	**P**hase **S**hift **K**eying
QAM	**Q**uadrature **A**mplitude **M**odulator
RAM	**R**andom **A**ccess **M**emory
ROM	**R**ead **O**nly **M**emory
VHDL	**V**HSIC **H**ardware **D**escription **L**anguage

Matrix Notations

$\mathbf{W}_N$	= FFT matrix for a vector of length N, based on the twiddle factor
$\mathbf{x}_N$	= Input vector of FFT in matrix form, of length N
$\mathbf{X}_N$	= Output vector of FFT in matrix form, of length N

TABLE 1. Transform Complexity.

N	Decimation-in-frequency FFT		Direct Method	
	Complex additions	Complex multiplier	Complex additions	Complex multiplier
	$N \log_2 N$	$N/2 \log_2 N$	$N(N-1)$	$(N-1)^2$
4	8	4	12	9
8	24	12	56	49
16	64	32	240	225
32	160	80	992	961
64	384	192	4032	3969
128	896	448	16256	16129
1024	10240	5120	1047552	1046529

TABLE 2. Possible values (constellation) for some modulation types.

odulation type	Number of points	Values
BPSK	2	+1, -1
QPSK	4	+1+0i, +1-0i, 0+i, 0-i
16 QAM	16	+1+i, +1-i, -1+i, -1-i, +3+i, +3-i, -3+i, -3-i, +1+3i, +1-3i, -1+3i, -1-3i, +3+3i, +3-3i, -3+3i, -3-3i.

TABLE 3. System Complexity and Processing time calculations.

Block name	Number of block needed for N IDFT point	Delay for one block	Processing time for N point IDFT	category of working
D-FF	$[\log_2(M)]-1$	$2G^1$	$2G *([\log_2(M)]-1)$	All D-FFs works serially
Decoder	1decoder with($\log_2(M)$: M) type	$2G^1$	2G * 1	----------
Cumulative Adder block	N	$3G^1$	3G * 1	All CABs works concurrently
Logic register (memory)	$(0.5\sqrt{M}*N^2*P)+(2NP)$ bit	$3G^1$	3G * 1	All memories accesses concurrently
Control unit	1	$3G^1$	3G * 1	----------
[1]Note: G is the propagation delay due to one gate				

TABLE 4. Decoder equivalent to QPSK mapper.

Mapper output equivalence	Inputs		Outputs			
	In 2	In 1	Op 1	Op 2	Op 3	Op 4
+1+0i	0	0	1	0	0	0
0+i	0	1	0	1	0	0
0-i	1	0	0	0	1	0
-1+0i	1	1	0	0	0	1

TABLE 5. Contents of the two register attached to first CAB for input x(n).

Seq	input	Corresponding constellation	Multiplication result	Contents of Result1_real register	Contents of Result1_imag register
0	no input	----------	0+0j	0	0
1	11	-1+0i	$-a_1 - b_1i$	$-a_1$	$-b_1$
2	11	-1+0i	$-a_2 - b_2i$	$-a_1-a_2$	$-b_1-b_2$
3	11	-1+0i	$-a_3 - b_3i$	$-a_1-a_2-a_3$	$-b_1-b_2-b_3$
4	00	+1+0i	$a_4 + b_4i$	$-a_1-a_2-a_3+a_4$	$-b_1-b_2-b_3+b_4$
5	10	0+i	$-b_5 + a_5i$	$-a_1-a_2-a_3+a_4-b_5$	$-b_1-b_2-b_3+b_4+a_5$
6	01	0-i	$b_6 - a_6i$	$-a_1-a_2-a_3+a_4-b_5+b_6$	$-b_1-b_2-b_3+b_4+a_5-a_6$
7	11	-1+0i	$-a_7 - b_7i$	$-a_1-a_2-a_3+a_4-b_5+b_6-a_7$	$-b_1-b_2-b_3+b_4+a_5\ a_6-b_7$
8	01	0-i	$b_8 - a_8i$	$-a_1-a_2-a_3+a_4-b_5+b_6-a_7+b_8$	$-b_1-b_2-b_3+b_4+a_5-a_6-b_7-a_8$
9	10	0+i	$-b_9 + a_9i$	$-a_1-a_2-a_3+a_4-b_5+b_6-a_7+b_8-b_9$	$-b_1-b_2-b_3+b_4+a_5-a_6-b_7-a_8+a_9$
10	10	0+i	$-b_{10} + a_{10}i$	$-a_1-a_2-a_3+a_4-b_5+b_6-a_7+b_8-b_9-b_{10}$	$-b_1-b_2-b_3+b_4+a_5-a_6-b_7-a_8+a_9+a_{10}$
11	00	+1+0i	$a_{11} + b_{11}i$	$-a_1-a_2-a_3+a_4-b_5+b_6-a_7+b_8-b_9-b_{10}+a_{11}$	$-b_1-b_2-b_3+b_4+a_5-a_6-b_7-a_8+a_9+a_{10}+b_{11}$
12	01	0-i	$b_{12} - a_{12}i$	$-a_1-a_2-a_3+a_4-b_5+b_6-a_7+b_8-b_9-b_{10}+a_{11}+b_{12}$	$-b_1-b_2-b_3+b_4+a_5-a_6-b_7-a_8+a_9+a_{10}+b_{11}-a_{12}$
13	01	0-i	$b_{13} - a_{13}i$	$-a_1-a_2-a_3+a_4-b_5+b_6-a_7+b_8-b_9-b_{10}+a_{11}+b_{12}+b_{13}$	$-b_1-b_2-b_3+b_4+a_5-a_6-b_7-a_8+a_9+a_{10}+b_{11}-a_{12}-a_{13}$
14	10	0+i	$-b_{14} + a_{14}i$	$-a_1-a_2-a_3+a_4-b_5+b_6-a_7+b_8-b_9-b_{10}+a_{11}+b_{12}+b_{13}-b_{14}$	$-b_1-b_2-b_3+b_4+a_5-a_6-b_7-a_8+a_9+a_{10}+b_{11}-a_{12}-a_{13}+a_{14}$
15	00	+1+0i	$a_{15} + b_{15}i$	$-a_1-a_2-a_3+a_4-b_5+b_6-a_7+b_8-b_9-b_{10}+a_{11}+b_{12}+b_{13}-b_{14}+a_{15}$	$-b_1-b_2-b_3+b_4+a_5-a_6-b_7-a_8+a_9+a_{10}+b_{11}-a_{12}-a_{13}+a_{14}+b_{15}$
16	01	0-i	$b_{16} - a_{16}i$	$-a_1-a_2-a_3+a_4-b_5+b_6-a_7+b_8-b_9-b_{10}+a_{11}+b_{12}+b_{13}-b_{14}+a_{15}+b_{16}$	$-b_1-b_2-b_3+b_4+a_5-a_6-b_7-a_8+a_9+a_{10}+b_{11}-a_{12}-a_{13}+a_{14}+b_{15}-a_{16}$

TABLE 6 Results comparison for transmitter.

seq.	Matlab results	Implemented transmitter results		
		Real part	Imaginary part	Equivalent
1	-2.0000	111100000110000	000000000000000	-2.000+0.000i
2	3.2515 - 2.9554i	000110010110101	111010001110101	3.253-2.995i
3	1.0000 + 0.4142i	000001111101000	000000110011110	1.000+0.414i
4	1.1077 - 6.4861i	000010001010011	110011010101001	1.107-6.487i
5	-4.0000 + 4.0000i	111000001100000	000111110100000	-4.000+4.000i
6	1.8225 + 1.7208i	000011100011111	000011010111001	0.911+1.721i
7	2.4142 + 1.8284i	000100101101110	000011100100100	2.414+1.828i
8	0.1270 + 0.8032i	000000001111111	000001100100011	0.127+0.803i
9	4.0000 - 2.0000i	000111110100000	111100000110000	4.000-2.000i
10	-2.4231 - 1.8730i	111011010000111	111100010101111	-2.425-1.873i
11	1.0000 - 2.4142i	000001111101000	111011010010010	1.000-2.414i
12	3.7208 - 2.3423i	000111010001001	111011011011011	3.721-2.341i
13	-2.0000 + 2.0000i	111100000110000	000011111010000	-2.000+2.000i
14	-6.6509 - 0.8923i	110011000000101	111110010000011	-6.651-0.893i
15	-0.4142 - 3.8284i	111111001100010	111000100001100	-0.414-3.828i
16	-0.9554 - 3.9747i	111110001000101	111000001111001	-0.955-3.975i

TABLE 7. Comparisons between four cores of Inverse transform for streaming input.

Processor	Proposed IDFT core [Esttaifan, 2011]	Altera IFFT stratix core [Altera, 2010]	Altera IFFT cyclone III core [Altera, 2010]	Xilinx IFFT core [Xilinx, 2007]	IFFT core [García1, 2007]
Number of points	256	256	256	256	256
Data width (bit)	16	16	16	16	16
Combinational ALUTs	1680	2094	3437	2027	6702
Logic register	16521	3715	3906	-------	6498
18×18 multipliers	0	12	12	30	48
Memory	0	20 (9K)	20 (9K)	3(36K)	5 (4K)
Execution time P_T (µs)	0.045	0.58	1.11	0.59	0.123
Processing frequency (MHz)	1000	442	231	432	350

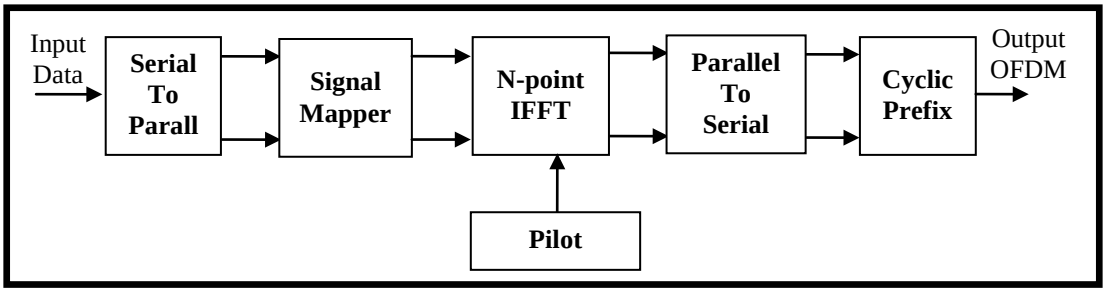


Figure 1. OFDM transmitter.

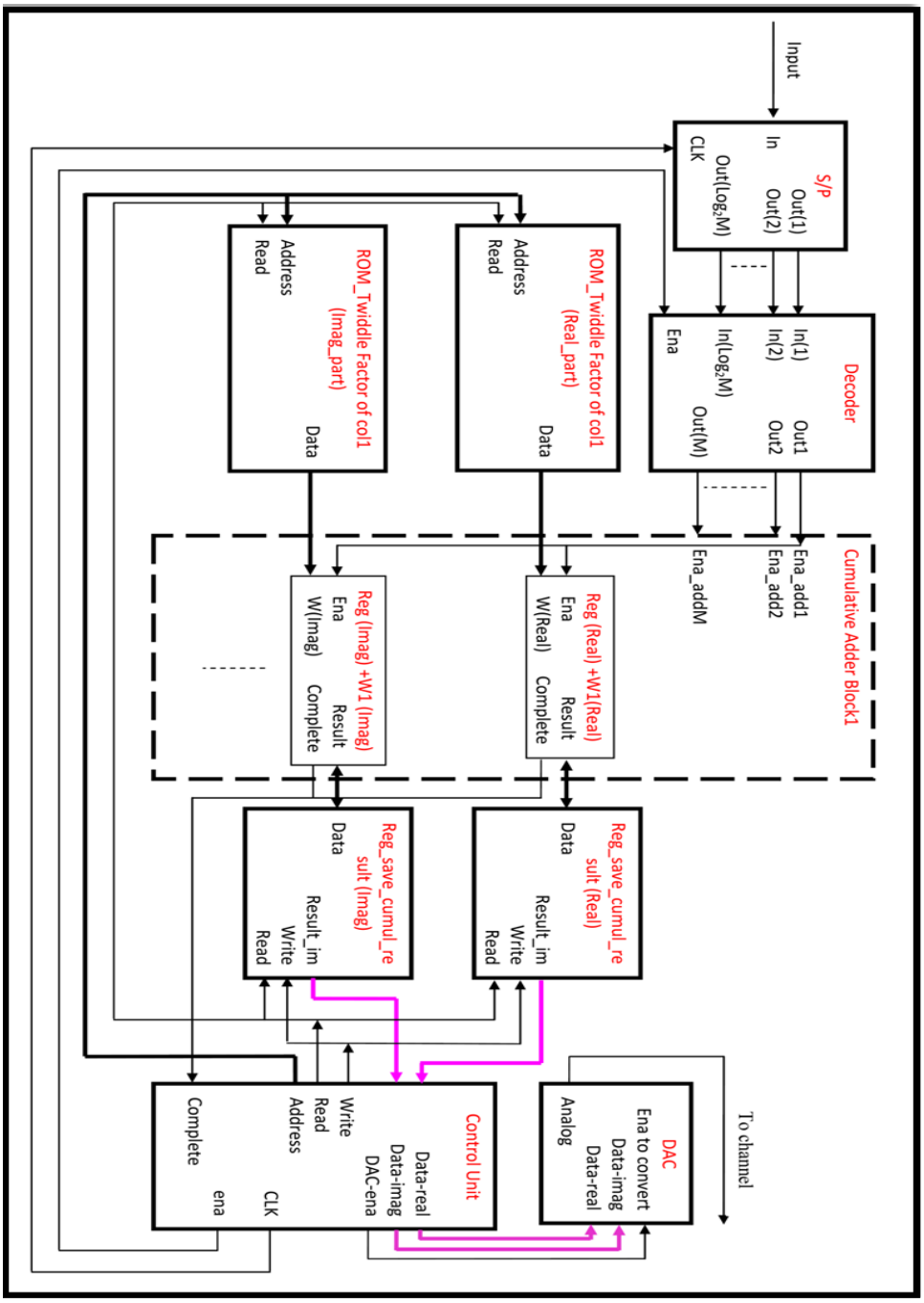


Figure 2. Block Diagram for the proposed OFDM transmitter.

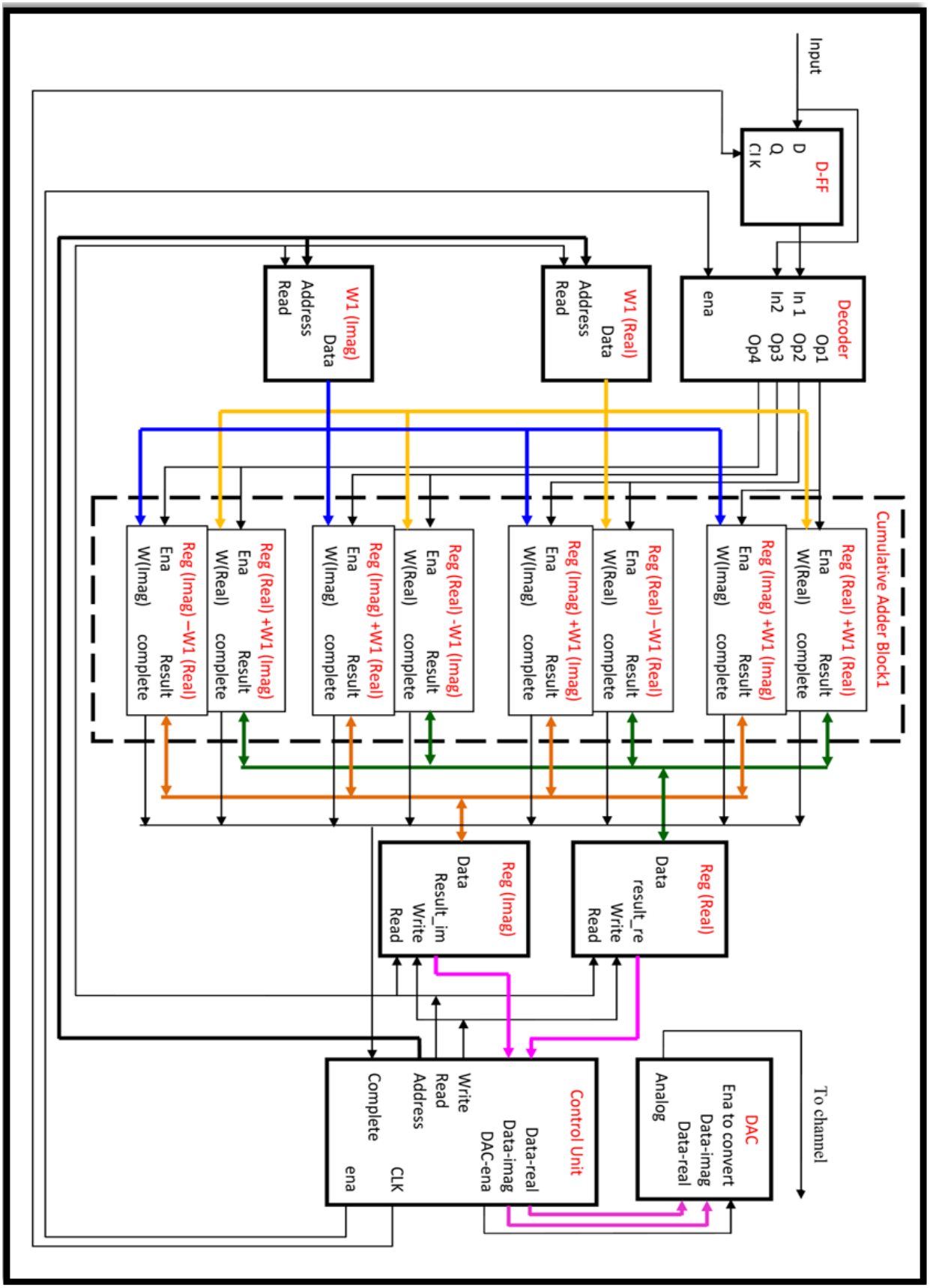


Figure 3. Block Diagram for the implemented OFDM transmitter.

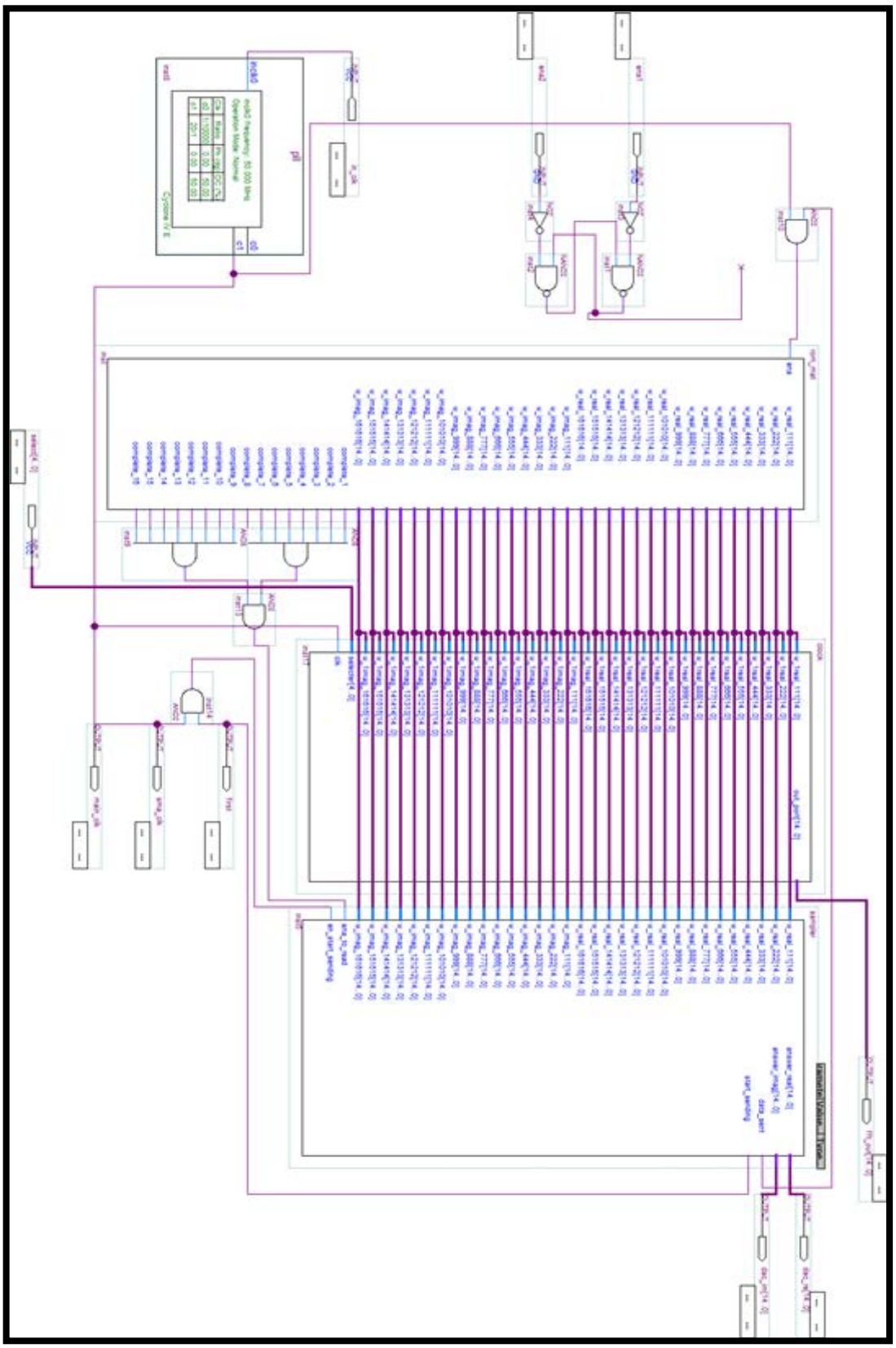
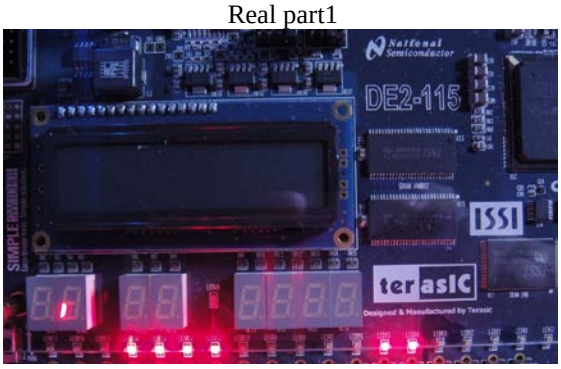
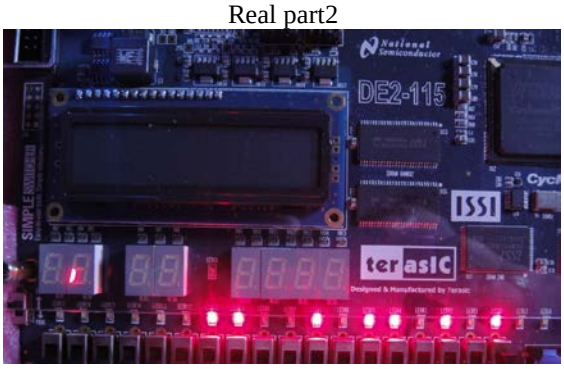


Figure 4. schematic diagram of the implemented OFDM transmitter generated by Quartus 10.0

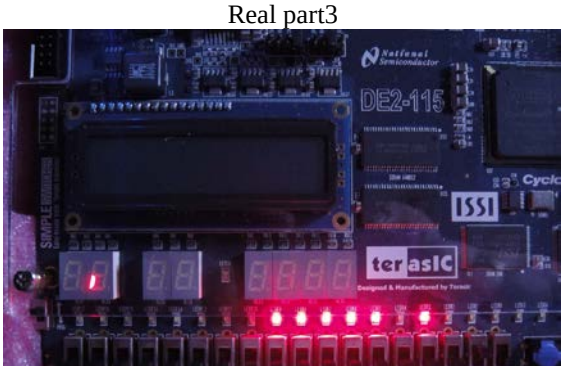
Real part1



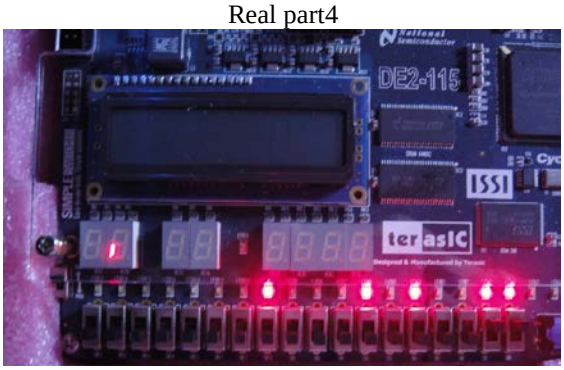
Real part2



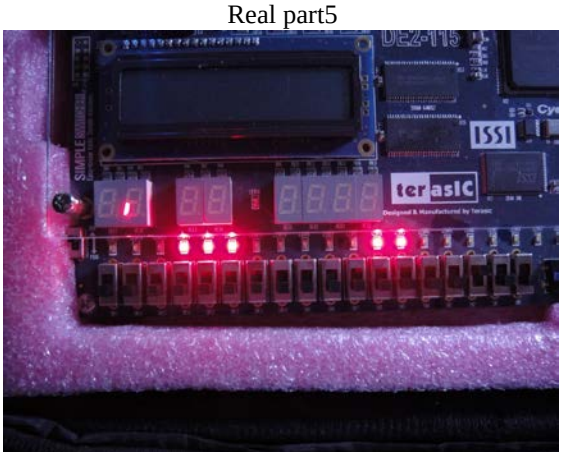
Real part3



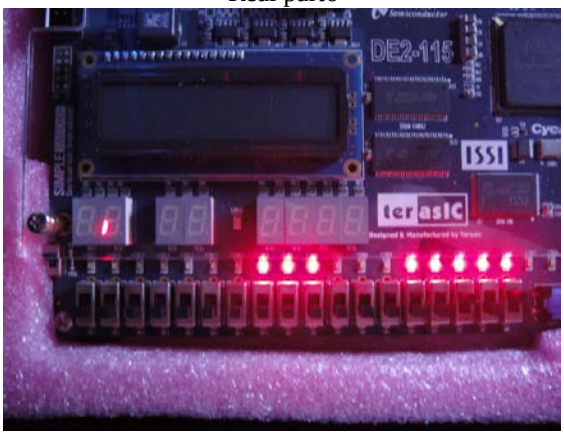
Real part4



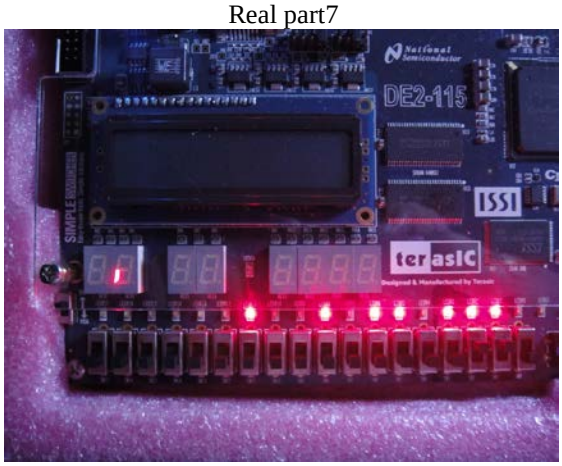
Real part5



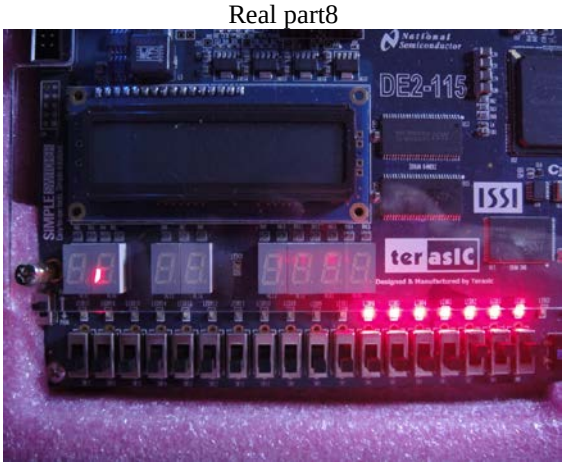
Real part6



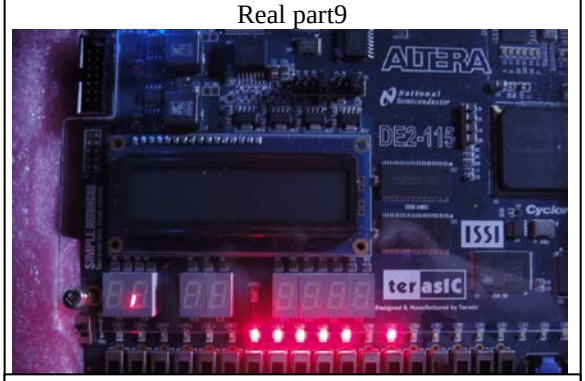
Real part7



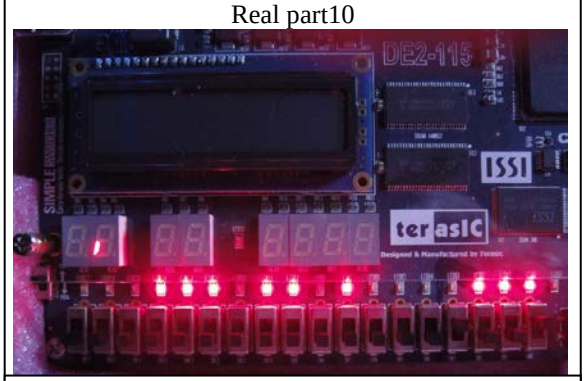
Real part8



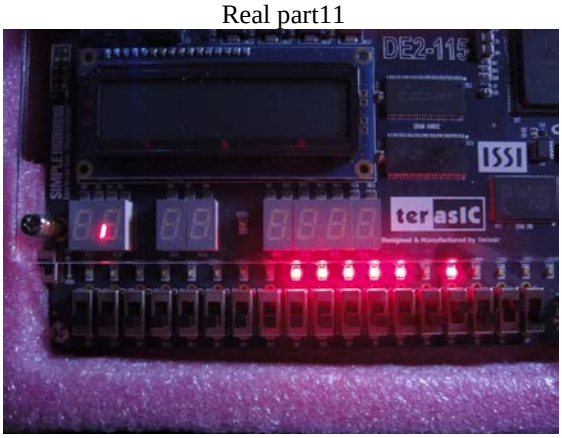
Real part9



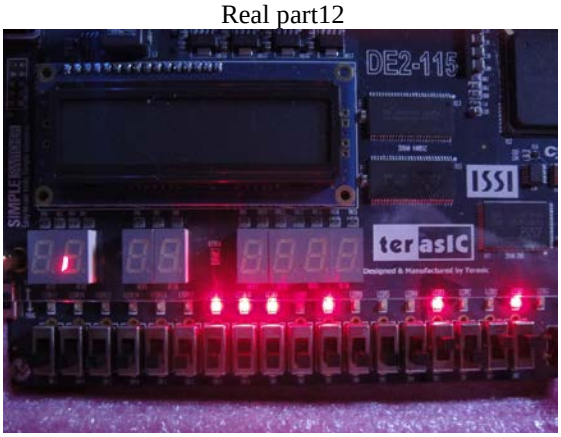
Real part10



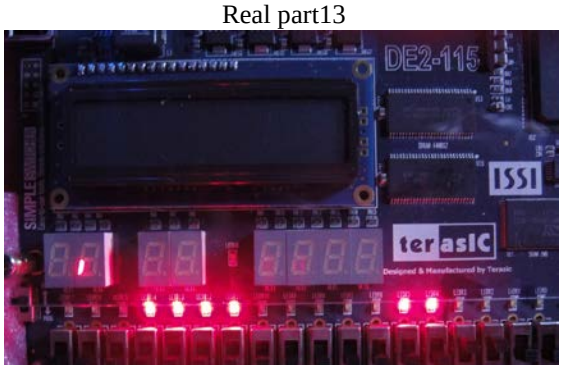
Real part11



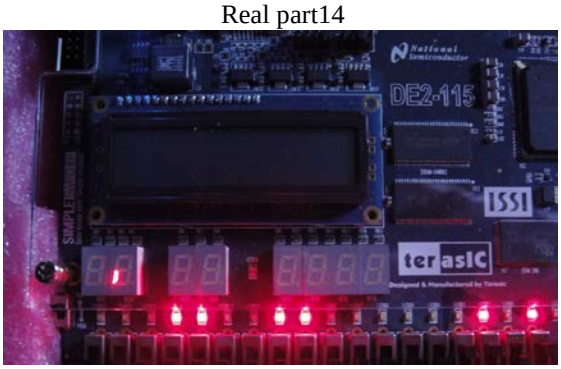
Real part12



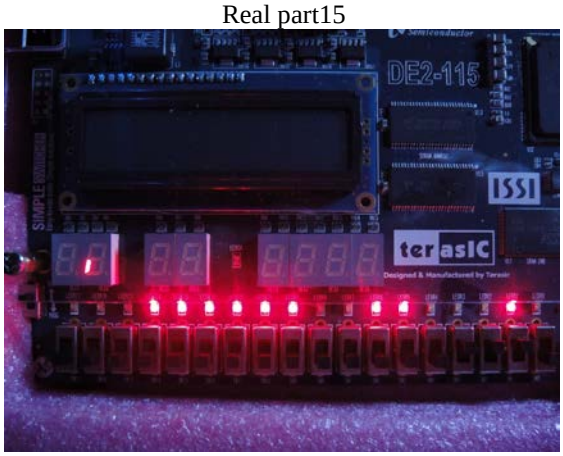
Real part13



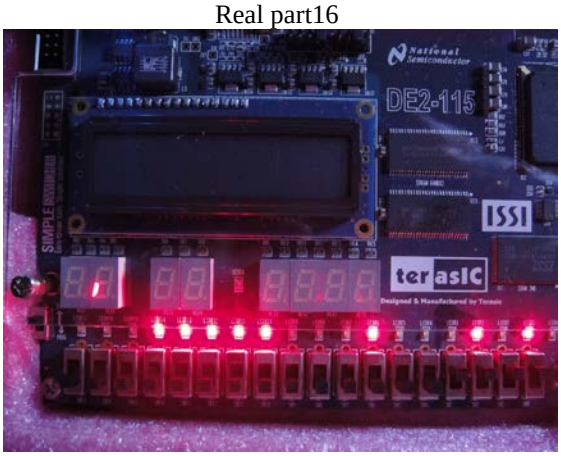
Real part14



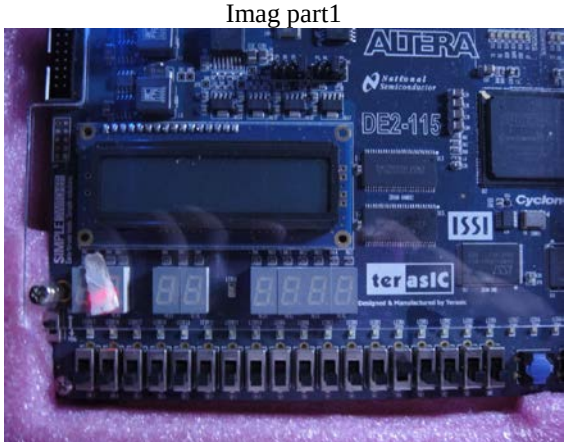
Real part15



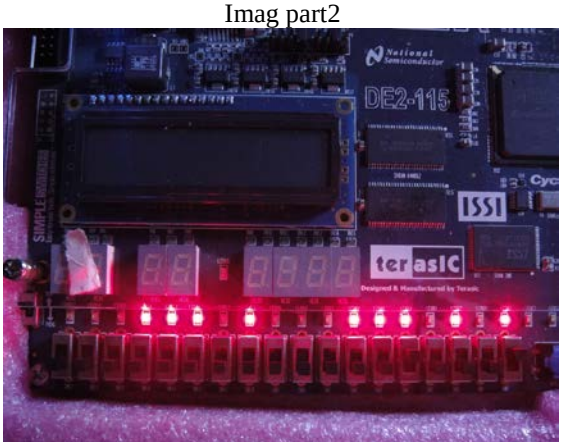
Real part16



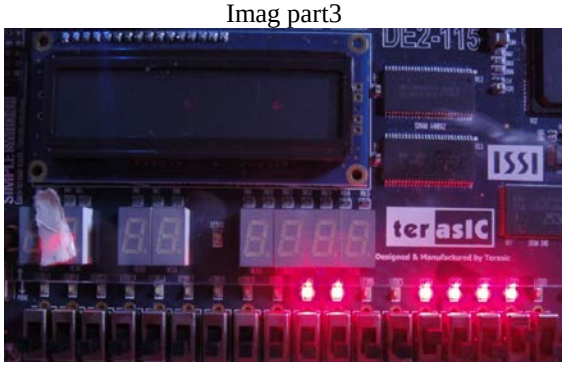
Imag part1



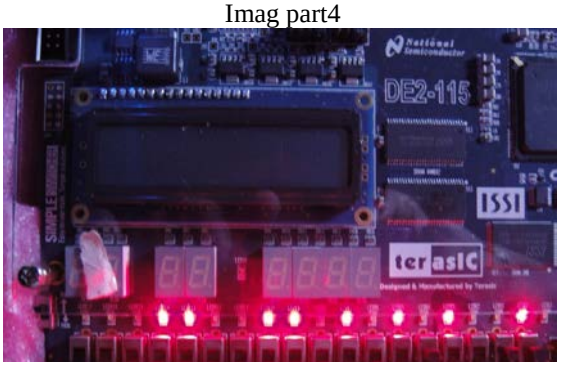
Imag part2



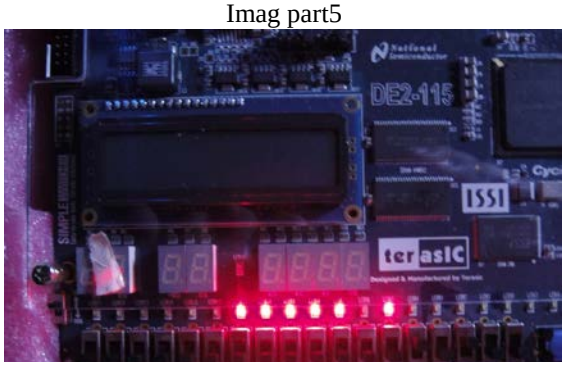
Imag part3



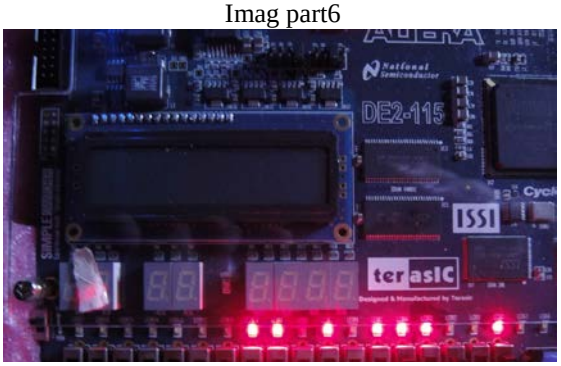
Imag part4



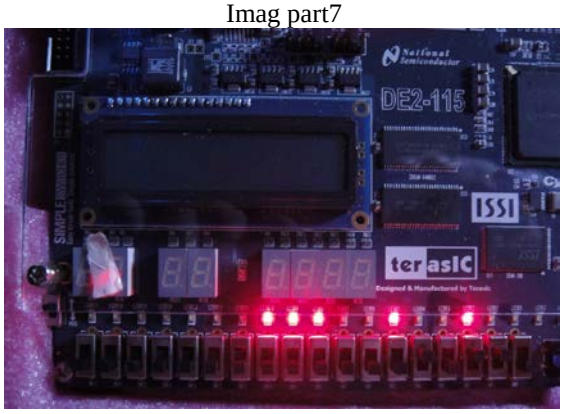
Imag part5



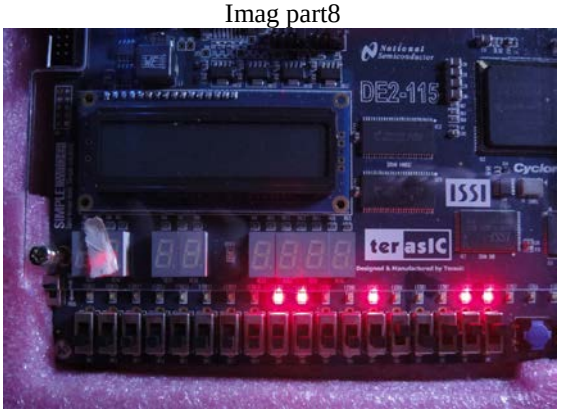
Imag part6



Imag part7



Imag part8



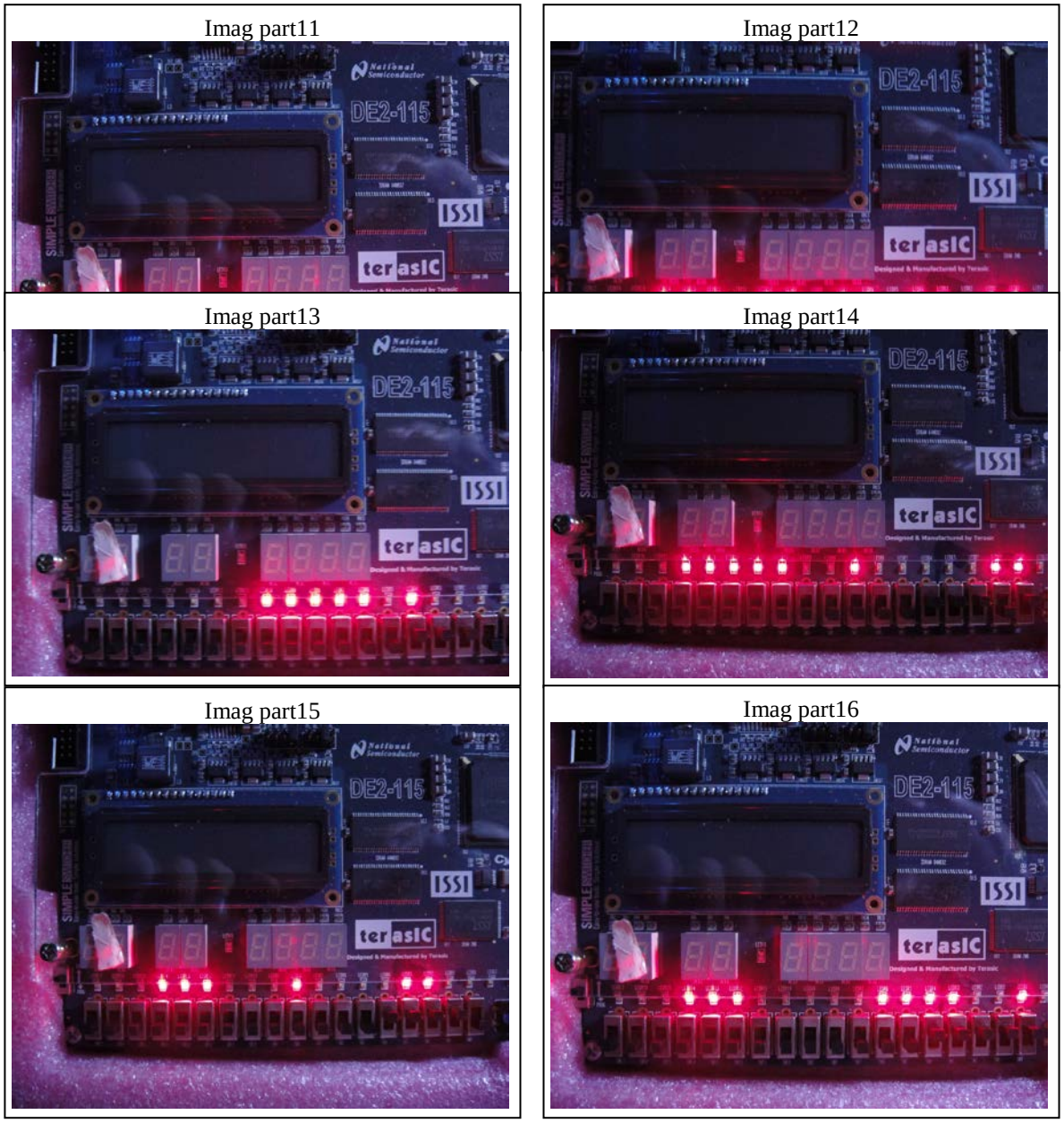


Figure 5. IDFT results at transmitter