Hardware co-simulation for a low complexity PAPR reduction scheme on an FPGA

Article *in* International Journal of Wireless and Mobile Computing · January 2017

DOI: 10.1504/IJWMC.2017.083053

Khalid Al-Hussaini
Thamar university
43 PUBLICATIONS **14** CITATIONS
SEE PROFILE

Borhanuddin Mohd Ali
Universiti Putra Malaysia
71 PUBLICATIONS **293** CITATIONS
SEE PROFILE

Pooria Varahram
Universiti Putra Malaysia
63 PUBLICATIONS **352** CITATIONS
SEE PROFILE

Ronan Farrell
National University of Ireland, Maynooth
180 PUBLICATIONS **559** CITATIONS
SEE PROFILE

Hardware co-simulation for a low complexity PAPR reduction scheme on an FPGA

Khalid Al-Hussaini* and Borhanuddin M. Ali

Department of Computer and Communications Systems Engineering,
Research Centre of Excellence for Wireless and Photonic Networks (WiPNET),
Universiti Putra Malaysia,
43400 UPM Serdang, Selangor, Malaysia
Email: khalid00alhussaini@gmail.com
Email: borhan@upm.edu.my
*Corresponding author

Pooria Varahram

Department of Electronic Engineering,
National University of Ireland,
Kildare, Ireland
Email: pooria.varahram@nuim.ie

Shaiful J. Hashim

Department of Computer and Communications Systems Engineering,
Research Centre of Excellence for Wireless and Photonic Networks (WiPNET),
Universiti Putra Malaysia,
43400 UPM Serdang, Selangor, Malaysia
Email: sjh@upm.edu.my

Ronan Farrell

Department of Electronic Engineering,
CTVR – The Telecommunication Research Centre,
Callan Institute,
National University of Ireland,
Kildare, Ireland
Email: rfarrell@eeng.nuim.ie

Abstract: This paper presents a novel low-complexity technique for reducing the Peak-to-Average Power Ratio (PAPR) in Orthogonal Frequency Division Multiplexing (OFDM) systems followed by an efficient hardware co-simulation implementation of this technique by using a Xilinx system generator on a Field Programmable Gate Array (FPGA). In this technique, each subblock is interleaved with the others, and a new optimisation scheme is introduced in which the number of iterations is equal only to the number of subblocks, which results in reduced processing time and less computation that, in turn, leads to reduced complexity. Furthermore, the proposed method focuses on simplifying the required hardware resources. Thus, it can be easily combined with other simplified techniques. The simulation results demonstrate that the new technique can effectively reduce the complexity up to 98.22% compared with the new existing Partial Transmit Sequence (PTS) techniques and yield a good Bit Error Rate (BER) performance. Through the comparison of performance between simulation and hardware, it is distinctly illustrated that the designed hardware block diagram is as workable as the simulation and the difference of the result is only 0.1 dB.

Keywords: OFDM; PAPR; PTS; hardware co-simulation; FPGA.

Reference to this paper should be made as follows: Al-Hussaini, K., Ali, B.M., Varahram, P., Hashim, S.J. and Farrell, R. (2017) 'Hardware co-simulation for a low complexity PAPR reduction scheme on an FPGA', *Int. J. Wireless and Mobile Computing*, Vol. 12, No. 1, pp.49–61.

Biographical notes: Khalid Al-Hussaini is currently a PhD candidate in Wireless Communication Engineering at the Department of Computer and Communications Systems Engineering, UPM, Malaysia. His research interests are MIMO systems, OFDM and multi-carrier systems, PAPR reduction in OFDM systems, and FPGA design.

Borhanuddin M. Ali is currently a Professor at the Department of Computer and Communications Systems Engineering, WiPNET Research Centre, UPM, Malaysia. He received his PhD from University of Wales, Cardiff, in 1985. His research interest spans the broad area of wireless communications and networks technology.

Pooria Varahram is currently a senior postdoctoral at the Department of Electronic Engineering, National University of Ireland, Kildare, Ireland. He received his PhD in Wireless Communication Engineering from the UPM in 2010. His research interests are PAPR reduction in OFDM systems and linearisation of power amplifiers.

Shaiful J. Hashim is currently an Associate Professor at the Department of Computer and Communications Systems Engineering, UPM, Malaysia. He received his PhD from Cardiff University, UK, in 2011 in the field of Electrical and Electronics Engineering. His research interests are energy-efficient wireless system and network security.

Ronan Farrell is currently a Professor at the Department of Electronic Engineering, CTVR Centre, Callan Institute, National University of Ireland, Kildare, Ireland. He received his PhD from University College Dublin in 1998. His research interests include wireless system design, electronics and radio systems.

1 Introduction

Modulation technique plays significant role as a component of communication systems. A novel technique known as Orthogonal Frequency Division Multiplexing (OFDM), which can be implemented in broadband wireless systems, has been designed and developed to fulfil the requirements of high data rate signals (Rahmatallah and Mohan, 2013). Generally, OFDM is a multicarrier transmission scheme, and, hence, it is a superior modulation technique that is appropriate for high speed data transmission owing to its robustness in multipath propagation environments, which is achieved by overcoming Inter-Symbol Interference (ISI) in fading channels, high data rates and bandwidth efficiency (Han and Lee, 2005; Jiang and Wu, 2008). In spite of OFDM features that make it an appropriate candidate for high speed data transmission compared to previous data transmission techniques, such as Code Division Multiple Access (CDMA), there are still drawbacks in OFDM systems. Despite the fact that the OFDM technique offers better features for transmitting signals compared to previous technologies, the Peak-to-Average Power Ratio (PAPR) is a critical challenge that needs to be reduced to ensure an efficient broadband communication system (Rahmatallah and Mohan, 2013; Jiang and Wu, 2008). The main criterion for an OFDM system is that it must be linear for a certain dynamic range; yet, the power amplifier usually presents obstacles for criterion, as power efficiency will decrease when PAPR increases (Jiang and Wu, 2008; Pandey and Tripathi, 2013). High PAPR leads to signal distortion whenever the signal is involved in the High Power Amplifier (HPA) non-linear region. The existence of signal distortion further leads to inter-modulation between subcarriers, as will out of band radiation due to saturation within the power amplifier; hence amplification will not only be inefficient, but it will also increase the transmitter cost and further lead to degradation of system performance (Al-Hussaini et al., 2016; Pandey and Tripathi, 2013; Lain et al., 2011). Implementation of the proposed PAPR reduction scheme into a Field Programmable Gate Array (FPGA) is vital to verify its effectiveness; thus, the specifications of each FPGA become crucial factors to take into consideration. In addition to that, there are several aspects that need to be taken into account during implementation on an FPGA board, such as the resolution of the respective FPGA, hardware resources and so on. Hardware resource usage should be as low as possible to ensure a low cost system. This paper will focus on the design of the proposed PAPR reduction technique based on the scrambling technique with low computational complexity and good PAPR performance.

2 Related work

The most popular techniques are Partial Transmit Sequence (PTS) and selected mapping (SLM), and both of these techniques reduce PAPR significantly without exhibiting output distortion (Rahmatallah and Mohan, 2013; Müller and Huber, 1997; Ibraheem et al., 2014). However, SLM has higher computational complexity than PTS, and this limits the implementation for large carriers. PTS is easier to implement, even though the computational complexity will increase exponentially with the increase of the number of subcarriers (Al-Hussaini et al., 2016; Ibraheem et al., 2014; Lain et al., 2011). This paper focuses on the simulation and implementation of the proposed technique approach to reduce PAPR and the computational complexity of the proposed technique will increase linearly with the increase of the number of subblock.

In Mukunthan and Dananjayan (2014), a study is carried out on a PTS scheme that adopts gray code, and the design is verified by a MATLAB simulation and implementation on an FPGA. The gray code PTS managed to lower the computational complexity and thus simplify the hardware implementation. This research is implemented on a Xilinx FPGA XC4VFX60, and resource consumption is much lower compared to conventional-PTS (C-PTS). The results show that the designed algorithm is workable.

In Liu et al. (2011), research is conducted on Enhanced PTS (EPTS), which deploys each new phase sequence to search for the most optimum phase sequence if the respective phase factor is still not the optimum. The design is then implemented on an FPGA model XC4VSX35-10FF668, and resource consumption does not go higher than 34 for each resource. The results of simulation and implementation illustrate that there is but a small difference due to some shortages of the FPGA.

In Varahram and Ali (2011), research is conducted on Dummy Sequence Insertion Enhanced Partial Transmit Sequence (DSI-EPTS), which is based on interleaved phase sequence and deploys each new phase sequence to search for the most optimum phase sequence if the respective phase factor is still not the optimum. The design is then implemented on an FPGA model XC4VSX35-10FF668, and resource consumption does not go higher than 27 for each resource.

3 System model

In OFDM systems, a data stream of rate *R* (in units of bps) is modulated to a Phase Shift Keying (PSK) or Quadrature Amplitude Modulation (QAM) scheme. A set of *N* mapped signals is converted into *N* parallel streams by using a serial-to-parallel converter. These sets are referred to as the OFDM symbols. Afterward, an inverse fast Fourier transform (IFFT) of length *N* is applied to produce orthogonal data subcarriers. Then, all of the orthogonal subcarriers are transmitted simultaneously over the symbol interval *T*. A complex baseband OFDM signal *x*(*t*) with *N* orthogonal subcarriers can be written as follows (Rahmatallah and Mohan, 2013; Han and Lee, 2005; Jiang and Wu, 2008; Müller and Huber, 1997):

$$x(t) = \frac{1}{\sqrt{N}} \sum_{k=1}^{N-1} X_k e^{j2\pi k \Delta f t} \tag{1}$$

where $\Delta f = \frac{1}{T}$ is the subcarrier spacing and X_k is the *k*th frequency domain signal in the OFDM scheme.

The PAPR is the ratio between the maximum instantaneous power and the average power of the OFDM signal (Jiang and Wu, 2008), that is:

$$PAPR(x(t)) = \frac{\max\limits_{0 \le t \le T} |x(t)|^2}{E\left[|x(t)|^2\right]} \tag{2}$$

where *E*[.] is the expected value operator. On the other hands, PAPR in terms of logarithmic can be defined as:

$$PAPR_{dB} = 10 \log PAPR(x(t)) \tag{3}$$

We can also describe the characteristics of the above power values in terms of their magnitudes by defining the crest factor (CF); $\text{CF} = \sqrt{PAPR}$. High peaks appear when *N* different mapped symbols phases in equation (1) are accumulated constructively (Han and Lee, 2005).

3.1 C-PTS-OFDM technique

At the transmitter in the C-PTS technique, the incoming serial random data vectors are mapped into QAM symbols and then converted from serial to parallel streams:

$$X = [X_0, X_1, \ldots, X_{N-1}]^T \tag{4}$$

Then, *X* is partitioned into *Q* disjoint subblocks, which are represented by the vectors X_q $(1 \le q < Q)$ of length *P*, where *N* = *QP* for certain integers *Q* and *P*. For $q = 1, \cdots, Q$, let the matrix *A* be the zero-padded of X_q, which can be written as follows:

$$A = \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1,Q} \\ A_{21} & A_{22} & \cdots & A_{2,Q} \\ \vdots & \vdots & \ddots & \vdots \\ A_{LN,1} & A_{LN,2} & \cdots & A_{LN,Q} \end{bmatrix} \tag{5}$$

where *L* is the oversampling factor. Then, let the matrix *F* be the zero-padded IFFT of *A*, which can be written as follows:

$$F = \begin{bmatrix} F_{11} & F_{12} & \cdots & F_{1,Q} \\ F_{21} & F_{22} & \cdots & F_{2,Q} \\ \vdots & \vdots & \ddots & \vdots \\ F_{LN,1} & F_{LN,2} & \cdots & F_{LN,Q} \end{bmatrix} \tag{6}$$

Then, the time domain sequences can be combined to minimise the PAPR, by applying the complex phase rotation factors $w = [w_1, w_2, \ldots, w_Q]^T$. The resulting time domain signal after combination can be written as follows:

$$x' = Fw \tag{7}$$

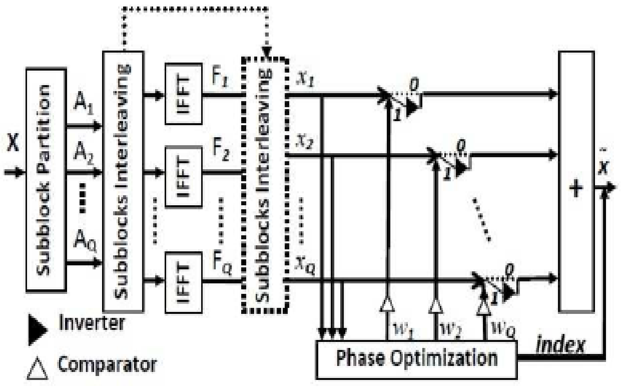
where $x' = [x'_1, x'_2, \ldots, x'_{LN}]$ is the block of optimised signal samples. Hence, the objective of the PTS technique is to design an optimal phase factor for the subblock set that minimises the PAPR. The objective of the optimisation problem is to identify optimum phases *w* that satisfy:

$$\{\hat{w}_1, \hat{w}_2, \ldots, \hat{w}_Q\} = \underbrace{\arg\min}_{\{w_1, w_2, \ldots, w_Q\}} \left(\underbrace{\max}_{1 \le k < LN} \left| \sum_{q=1}^{Q} w_q F_{k,q} \right| \right) \tag{8}$$

where $w_q \in \{\pm 1, \pm j\}$ and ($W = 4$). b_1 can be set equal to 1 without loss of performance. Therefore, in the PTS technique, it is necessary to test W^{Q-1} sets of distinct possible candidate vectors w to satisfy eqaution (8). Accordingly, the computational complexity of the PTS technique increases exponentially with Q.

In our proposed technique, there are only two phase weight factors, 0, 1 and the calculation and comparison of PAPRs is performed only among Q candidate phase sequences. The computational complexity of the proposed technique increases linearly with Q.

Figure 1 Block diagram of the proposed technique (SBI-PTS)



4 Analysis of the proposed technique

In this paper, a novel subblocks interleaving (SBI) scheme (Al-Hussaini et al., 2016) for PTS OFDM is implemented and analysed. The subblocks interleaver used in this technique is described below. As shown in Figure 1, the subblocks interleaver can be applied in the frequency domain (before IFFT) or in the time domain (after IFFT). In other words, the input of the subblocks interleaver can be the matrix A^T in equation (5) or the matrix F^T in equation (6).

4.1 SBI-PTS technique

The use of a subblocks interleaver offers a more constructive approach than that of the conventional interleaver technique. The main purpose of adopting the subblocks interleaver in SBI-PTS is to reduce PAPR by reducing the probability of two peaks being added together, which leads to a sudden shot up at the output envelope. In other words, through the subblocks interleave process, some high peaks might be cancelled out by a low peak and therefore reduce the probability of high PAPR being produced (Goldsmith, 2005; Sabbir and Makoto, 2013). We begin with a matrix of ($Q \times N$) symbols/samples and write them column-wise into an (N/B) × (QB) matrix. Then, we transpose the resulting matrix and read out the symbols/samples column-wise into an ($Q \times N$) matrix. Equations (9) and (10) present the matrices n for writing and $\pi(n)$ for reading. Consider, as an example, $N = 8$, $Q = 4$, $L = 1$ and $B = 4$. We write the matrix A^T or F^T as a (4 × 8) matrix as follows:

$$\begin{bmatrix} AF_{11} & AF_{12} & AF_{13} & \cdots & AF_{17} & AF_{18} \\ AF_{21} & AF_{22} & AF_{23} & \cdots & AF_{27} & AF_{28} \\ AF_{31} & AF_{32} & AF_{33} & \cdots & AF_{37} & AF_{38} \\ AF_{41} & AF_{42} & AF_{43} & \cdots & AF_{47} & AF_{48} \end{bmatrix}$$

Then, we write the above matrix column-wise into a (2 × 16) matrix as follows:

$$n = \begin{bmatrix} AF_{11} & AF_{15} \\ AF_{21} & AF_{25} \\ dr_{31} & AF_{35} \\ \vdots & \vdots \\ AF_{44} & AF_{47} \\ AF_{14} & AF_{18} \\ AF_{24} & AF_{28} \\ AF_{34} & AF_{38} \\ AF_{44} & AF_{48} \\ & \end{bmatrix} \tag{9}$$

We then transpose the resulting matrix and read out its contents column-wise into a (4 × 8) matrix as follows:

$$\pi(n) = \begin{bmatrix} AF_{11} & AF_{31} & AF_{12} & \ldots & AF_{14} & AF_{34} \\ AF_{15} & AF_{35} & AF_{16} & \ldots & AF_{18} & AF_{38} \\ AF_{21} & AF_{41} & AF_{22} & \ldots & AF_{24} & AF_{44} \\ AF_{25} & AF_{45} & AF_{26} & \ldots & AF_{28} & AF_{48} \end{bmatrix} \tag{10}$$

If the subblocks interleaver is applied in the frequency domain, the matrix $\pi(n)$ is the input of the IFFT blocks. If the subblocks interleaver is applied in the time domain, the matrix $\pi(n)$ is the input of the optimisation process block.

Now, a new phase optimisation scheme that permanently eliminates multiplicative operations is applied. Only two phase sequences, where the possible phases are {0, 1}, are required. First, all phase sequence possibilities are generated using an encoder of size $2^Q \times Q$. This encoder generates all possible phase sequences for a total of 2^Q phase sequences. For example, if $Q = 3$, the size of the encoder is 8 × 3, as shown in Table 1.

Then, the phase of each subblock is converted in accordance with the proposed weight of the phase rotation as follows:

$$\tilde{x} = \sum_{q=1}^{Q} (-1)^{w_q} x_q \tag{11}$$

where $w_q \in \{0,1\}$ and $\{x_q, q = 1,2,\ldots,Q\}$ are the input elements of the optimisation block. As shown in Figure 1, the comparator detects whether the phase factor is 0 or 1. If the weight of the phase factor is 0, the phase of the elements of the subblock does not change and they are passed directly to the summation unit; if the weight of the phase factor is 1, the phase is rotated by being passed through the inverter and is then passed to the summation unit. As the first step, the PAPR of the combined signal is calculated. We check the w_1 of the phase sequences of Table 1; if the PAPR at {000} is lower than the PAPR at {100}, then all phase-sequence possibilities with $w_1 = 1$ will be neglected (i.e. $\{w_1\ w_2\ w_3\} = \{100,101,110,111\}$). Hence, half of the phase sequences of Table 1 are eliminated. Afterward, we check the w_2 of the remaining phase sequences (i.e. $\{w_1\ w_2\ w_3\} = \{000,001,010,011\}$); if the PAPR at {000} is lower than the PAPR at {010}, then all phase sequences with $w_2 = 1$ will be neglected (i.e. $\{w_1\ w_2\ w_3\} = \{010,011\}$). Hence, half of the remaining sequences are eliminated. Finally, one of the final two sequences (i.e. $\{w_1\ w_2\ w_3\} = \{000,001\}$) will be the optimal sequence with a minimum PAPR.

Table 1 Candidate phase sequences using an 8 × 3 encoder

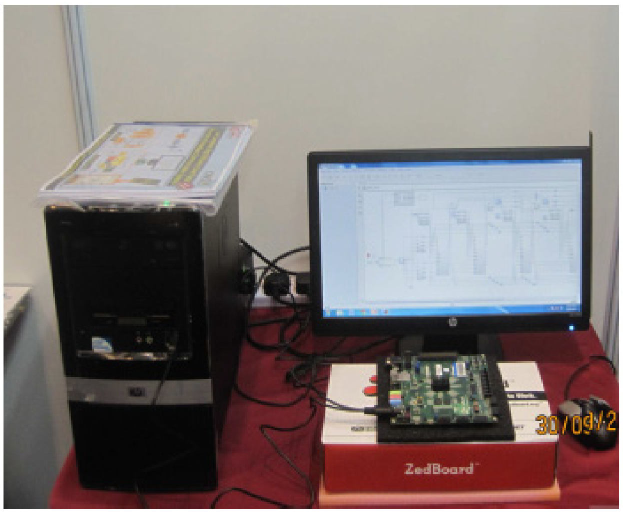
Index	$\{w_1\ w_2\ w_3\}$
1	{000}
2	{001}
3	{010}
4	{011}
5	{100}
6	{101}
7	{110}
8	{111}

5 Implementation of the proposed scheme

Computational complexity and side information that need to be transmitted are troublesome not only for software simulation, but also for hardware implementation on an FPGA. The phase sequence optimisation in Mukunthan and Dananjayan (2014), Liu et al. (2011) and Varahram and Ali (2011) is applied offline and PAPR calculation is only performed for the best sequence. In this paper, all implementation is performed online, including the phase sequence optimisation and PAPR calculations for all candidate phase sequence.

In this paper, a Xilinx System Generator combined with MATLAB Simulink provides an easier and efficient method of developing the FPGA system design and simulating it. Additionally, the hardware co-simulation feature of the software enables an easier method for testing and debugging the design effectively on the actual hardware as shown in Figure 2. Hardware implementation of this paper is carried out on a Xilinx Zedboard Zynq XC7Z020-CLG484-1, which works together with Vivado ISE design suite 14.4 software to determine the feasibility and flexibility of the designed scheme. Zedboard is a low-cost development environment specifically for the Xilinx Zynq^{TM} 7000 SoC, and the features of this board allow users to build special yet powerful designs. The features of this board include memory, processor, various output display and a connectivity port.

Figure 2 Overview of the HW co-simulation Xilinx



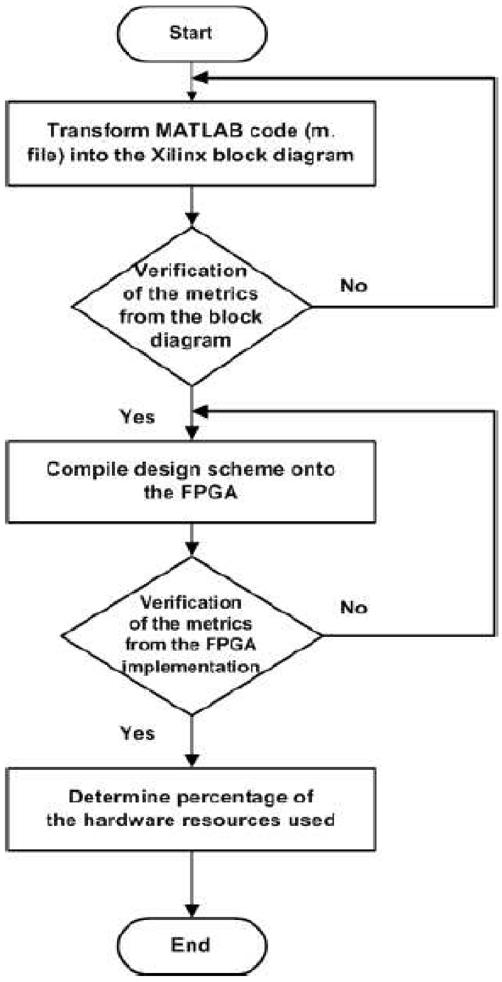
5.1 Hardware implementation flow

The steps taken to carry out the hardware implementation for the proposed technique (SBI-PTS) are briefly described in the flow chart in Figure 3. The first step is transforming the MATLAB code into a combination of a Simulink block diagram and Xilinx block diagram in the MATLAB environment with Xilinx SysGen. The simulation outcome of the block diagram is verified to ensure that we obtained the desired output before compiling the design onto FPGA. For the case in which the outcome is as desired, the design will be compiled onto FPGA, and another verification of metrics based on the FPGA outcome will be conducted. If both verifications are not satisfied, then we repeat the steps of transforming m.file through compiling the design onto FPGA again. The cycle will be repeated until the desired outcome is obtained from both the block diagram simulation and the FPGA implementation.

5.2 Hardware block diagram implementation

The implementation hardware block diagram for the proposed technique (SBI-PTS) is clearly shown that the design can be divided into six parts, including the partition block, IFFT blocks, subblock interleaver block, phase sequence generation block, optimisation block and PAPR calculation block. Each block is designed by utilising the Xilinx blockset, and each implements a different process; the design of each block will be further discussed later.

Figure 3 Hardware design flow


Start
Transform MATLAB code (m. file) into the Xilinx block diagram
Verification of the metrics from the block diagram
No
Yes
Compile design scheme onto the FPGA
Verification of the metrics from the FPGA implementation
No
Yes
Determine percentage of the hardware resources used
End

First, the input signal is imported by using the signal from the workspace block, and this block will import the signal in terms of the frame, which means that there are N symbols per frame. Since Xilinx SysGen handles a complex signal's real part and imaginary part separately, the signal is therefore broken into a real part and a imaginary part respectively by *complex to real-imag* block before the signal is fed in to the Xilinx SysGen environment. Next, both the real part and imaginary part of the signal will be fed into the *unbuffer* block, separately, to transform the frame-based signal into a sample based one. This is because Xilinx Sysgen does not support frame-based signal. The sample-based signal then goes through the six blocks accordingly, where each block is designed by adopting the Xilinx blockset.

To compile the design on FPGA, a *JTAG hardware co-simulation* block is generated as shown in Figure 4, and the amount of hardware resources used will be evaluated. Next, the FPGA is configured by using iMPACT software, followed by outputting the implementation outcome back to the MATLAB environment via a *signal to workspace* block. PAPR reduction performance from the hardware implementation can therefore be evaluated by plotting the Complementary Cumulative Distribution Function (CCDF) graph, as in MATLAB simulation.

5.3 *Partition block design*

The output signal of the *unbuffer* block will be fed into the partition block to perform the data partition. Again, the N = 265 symbol long data will be partitioned into Q = 4 subblocks where the actual data length will be N/4 symbols per block and rest of the empty space will be filled by zeros to maintain the length of N. Blocks used to design the partition includes the *counter*, *constant*, *relational*, *logical*, *convert* and *multiplier* blocks, and the designed partition block is illustrated in Figure 5. Input 1 and input 2 are shown in Figure 5, indicating the real part of the signal and the imaginary part of the signal, respectively, and there are eight outputs representing the real and imaginary signal of each subblock. Table 2 indicates the respective output signal for each subblock based on Figure 5. To partition data into the four subblocks with N/4 symbols each, four ranges of data are needed, where the first range of data will be input to the first subblock. The second range of data will be input to the second subblock, and so on. The range of data for each subblock is illustrated in Table 5. The *relational* block is used to design this data range, and data will be partitioned according to the count, which is performed by a *counter* block. For example, when the counter count is 83, the data should go to the second subblock. The input signal can therefore be partitioned by making use of the *mult* block to split the data according to the range created.

Figure 4 JTAG hardware co-simulation block

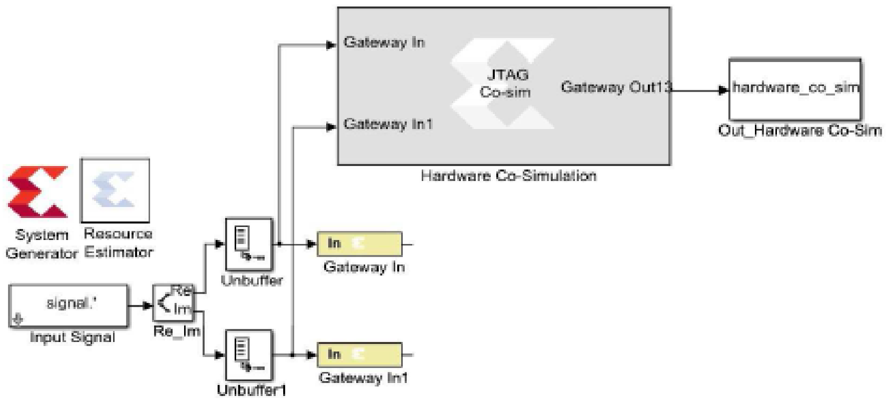


Table 2 The output port for each subblock

Subblock number (Q)	*Output port*	
	Real part	*Imaginary part*
1 (data range 1 to 64)	1	2
2 (data range 65 to 128)	3	4
3 (data range 129 to 192)	5	6
4 (data range 193 to 256)	7	8

5.4 *Inverse fast Fourier transform block design*

There is a total of four IFFT blocks being designed for hardware implementation, as shown in Figure 3. *Fast Fourier Transform 8.0* block with a type pipeline is used to perform the transformation of the signal. As shown in Figure 6, each *FFT* block has two input signals and two output signals, where the *data_tdata_xn_re* port and *data_tdata_xk_re* port indicate the input and output signal for the real part, while the *data_tdata_xn_im* port and *data_tdata_xk_im* port represent the input and output signal for the imaginary part.

To allow the *Fast Fourier Transform 8.0* block to act as the inverse fast Fourier transform (IFFT) block, the *config_tdata_fwd_inv* port is always set as low. Next, the *config_tvalid* port is always fixed to high to ensure configurations are valid at all times. The scaling factor that is input into the *config_tdata_scale_sch* port can be calculated in such a way that $N = 256 = 2^8$. The next step is to obtain a summation of eight from four stages of two bits. In this case, the four stages of two bits are designed in such a way that 01 10 10 11 is equal to 8. By converting this combination of binary numbers into decimal numbers, a scaling factor of 107 is obtained. The *data_tvalid* , *data_tlastanddata_tready* ports have to always be high throughout the IFFT transformation process. In addition to that, an appropriate amount of delay is needed for the IFFT transformation process to be carried out.

Figure 5 Block diagram of the partition block

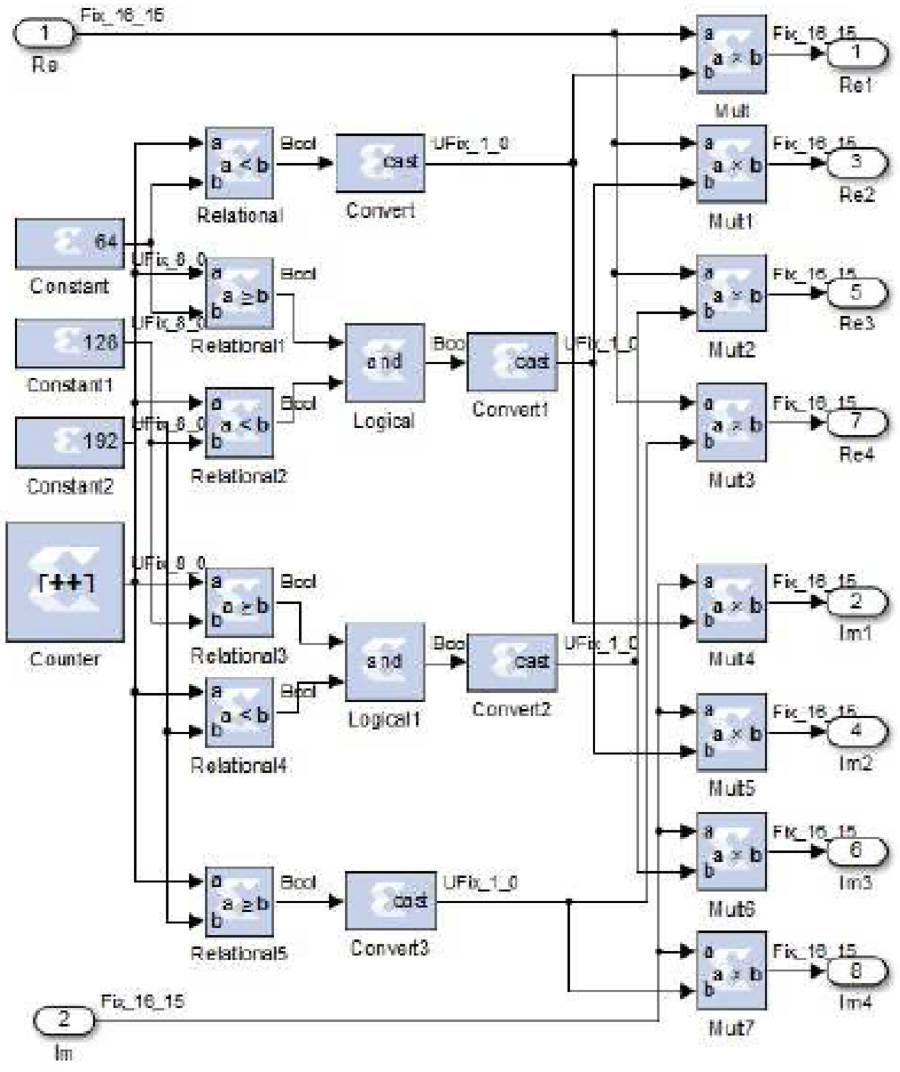
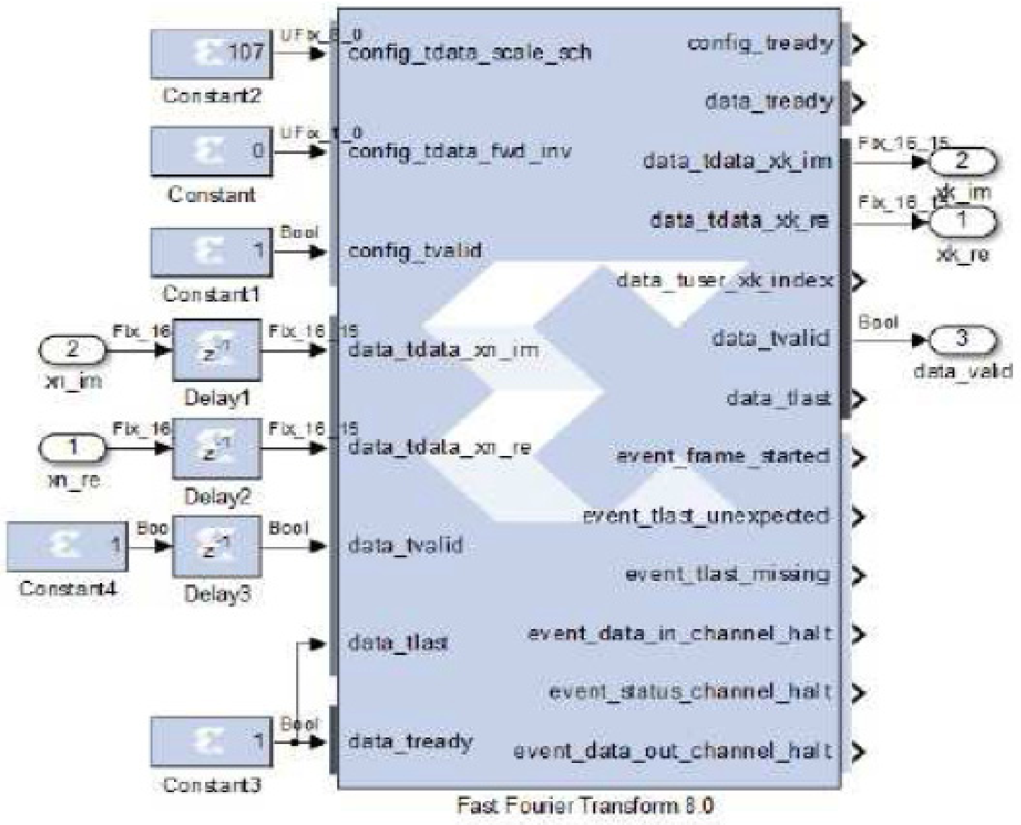


Figure 6 Block diagram of the inverse fast Fourier transform block



5.5 Subblock interleaver block design

Blocks used to design the subblock interleave process include *counter*, *slice*, *addsub*, *shift*, *shared memory*, *multiplexer* and *delay*. The main idea of the subblock interleave process is to rearrange the sequences of each data, as explained in Section 4, and two rounds of the interleave process are needed. Therefore, the first round of the interleave process is designed by adopting the *multiplexer*. Four ($Q = 4$) multiplexers are used, as shown in Figure 7. Basically, the idea of the first round interleave is that the first *multiplexer* will take the first eight output data, that is data 0 through data 7, from each subblock, while the second *multiplexer* will take the next eight output data, that is data 8 through data 15 from each subblock. The third *multiplexer* takes data 16 through data 23 from each subblock; the fourth *multiplexer* takes data 24 through data 31 from each subblock and the next 8 data again back to the first *multiplexer*. This is illustrated in Table 3 to provide a clearer picture of the first round of interleave.

A representing the first subblock, *B* representing the second subblock, *C* representing the third subblock and *D* representing the fourth subblock are illustrated in Table 3. The output data from each *multiplexer* is then rearranged according to the data sequence of simulation result. Combination of *counter*, *slice*, *addsub* and *shift* are employed to control the second time of rearrange process. The second round of rearranged data sequence will be written into *shared memory* block until the rearrange process completed. Once the second round of rearrange process completed, another *shared memory* block is utilised to read out the data.

Table 3 Example of data sequence after first interleave process

First	*Second*	*Third*	*Fourth*
A0–A7	A8–A15	A16–A23	A24–A31
B0–B7	B8–B15	B16–B23	B24–B31
C0–C7	C8–C15	C16–C23	C24–C31
D0–D7	D8–D15	D16–D23	D24–D31
A32–A39	A40–A47	A48–A55	A56–A63
B32–B39	B40–B47	B48–B55	B56–B63
⋮	⋮	⋮	⋮

5.6 Phase sequence generation block design

Phase sequence plays an important role in offering low PAPR. In the hardware implementation, the *multiplexer* is the main element used to design the phase sequence generation, as shown in Figure 8. The first input of each *multiplexer* is connected to the *constant* 1, while the second input of each *multiplexer* is connected to the *constant* –1. The output of each *multiplexer* depends on the input of the selector. For example, if the inputs of the selectors for all the multiplexers are 0000, respectively, then the output of each *multiplexer* will be 1111. Therefore, a phase sequence of 1111 will be generated by this block.

Figure 7 Block diagram of the subblock interleaver block

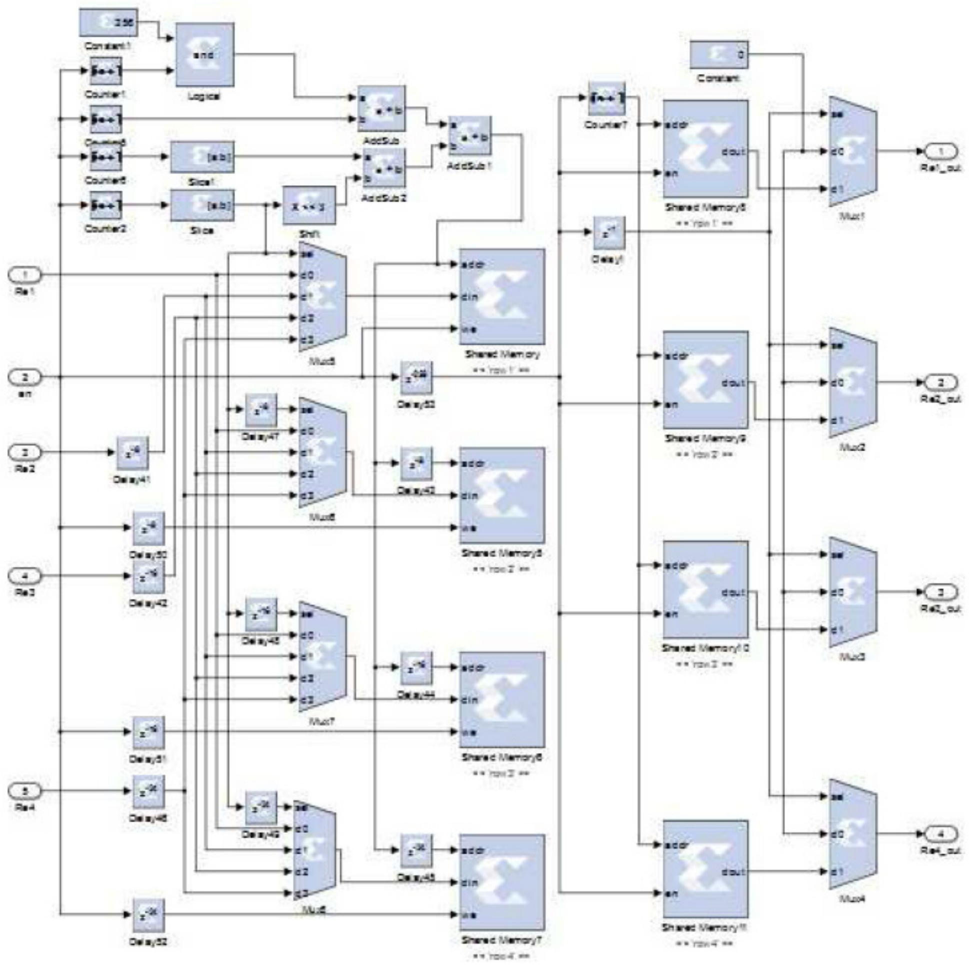
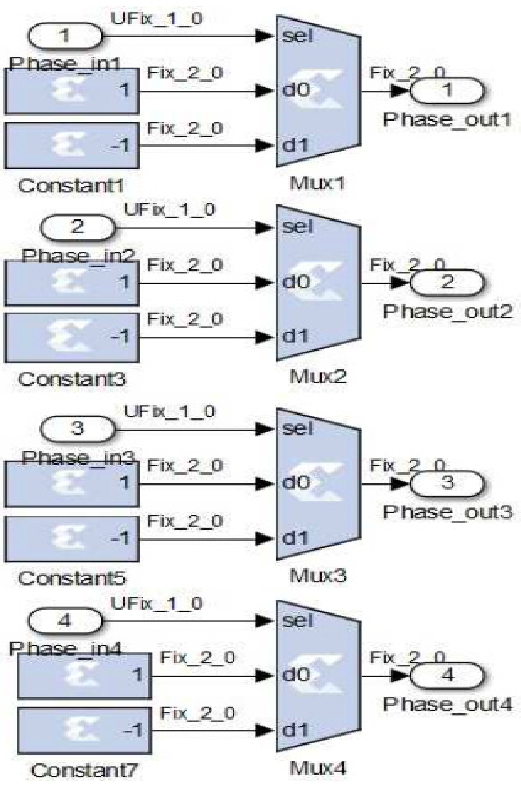


Figure 8 Block diagram of the phase sequence generation block



5.7 Optimisation block design

The optimisation block shown in Figure 9 mainly performs multiplication between the output signal from subblock interleaver block and the phase sequence generated. There is a total of eight *mult* blocks adopted to perform this multiplication, where four *mult* blocks represent the multiplication process of the real part of the signal with the phase sequence generated, while the other four *mult* blocks indicate the multiplication process of the imaginary part of the signal with the same generated phase sequence. There is no addition block that can perform the addition of four inputs at the same time, so two levels of additions are designed. In total, six *addsub* blocks are used to perform the addition process. The output signal from the optimisation block will then further proceed to the PAPR calculation block.

5.8 PAPR calculation block design

The function of the PAPR calculation block is to determine the lowest PAPR to ensure the performance of the OFDM system. As mentioned earlier, PAPR can be calculated by adopting equation (3), and there are two main parts to design, including determining the maximum signal power and the average signal power. Both the real part and the imaginary part of the input signal are first squared by adopting the *mult* block followed by summation via the *addsub* block to calculate the PAPR of the respective signal. To determine the average signal power, the *accumulator* block is used to accumulate all the data, and then divide them equally via the *shift* block. The data are shifted by 8, as shown in Figure 10, which means that it is divided by 2^8 or equal to $N = 256$. Next, the combination of the *multiplexer* block and the *relational* block is adopted to determine the maximum signal power. Four input multiplexers are used, and the selector is controlled by an input that acts as the *enable* port. For the case in which the *enable* port is equal to 0, the *multiplexer* will output 0, which is the value of input port 3 and input port 4. In contrast, for the case in which the *enable* port is equal to 1, the *multiplexer* will output the data from input port 0 or input port 1, and this is further based on whether the *relational* block output is high or low. For the case where the *relational* block output is high, the *multiplexer* will output the data of input port d0, while, in contrast, for the case in which the *relational* block output is low, the *multiplexer* will output the data of input port d1. This process will continue until the maximum signal power is determined. Next, the maximum signal power will be divided by the average signal power via the *divide* block, and the output of the division will further proceed and pass through the *natural logarithm* block. Owing to a limitation of the Xilinx blockset, the logarithm can be designed by using the *natural logarithm* block (ln) divided by the natural logarithm (ln(10)) as follows:

$$\log x = \frac{\ln x}{\ln 10} \tag{12}$$

Lastly, the PAPR in terms of the dB value can be obtained by multiplying the output value of the logarithm with the constant 10.

Figure 9 Block diagram of the optimisation block

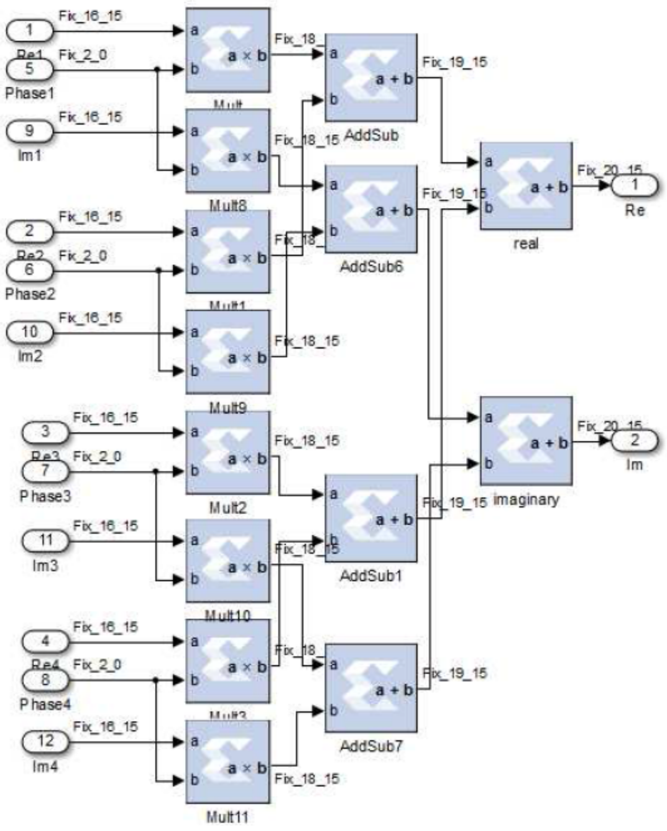
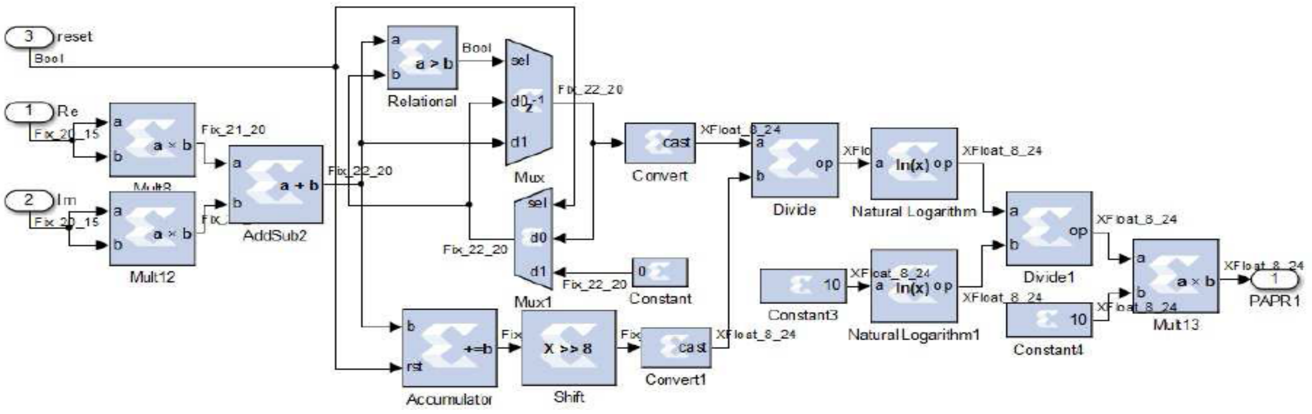


Figure 10 Block diagram of the PAPR calculation block



6 Experimental results and analysis

To measure the complexity reduction of the proposed technique against new existing PTS technique, the Computational Complexity Reduction Ratio (CCRR) of the proposed method over that of new existing PTS is applied, which is defined as follows:

$$CCRR = \left(1 - \frac{Complexity\ of\ SBI\_PTS\ tech.}{Complexity\ \ of\ \ New\_PTS\ tech.}\right) \times 100 \quad (13)$$

Based on equation (13), Table 4 provides a comparison of CCRRs of the parallel tabu search algorithm (Parallel TS-PTS) scheme (Taspinar et al., 2011), artificial bee colony algorithm (ABC-PTS) scheme (Wang et al., 2010), and successive local search using sequences (SLS) scheme (Cho et al., 2012) with the proposed technique (SBI-PTS) when $Q = 16, L = 4, W = 2, N = 256, T1 = 900, T2 = 900$, and $T3 = P_0 + (W-1)\sum_{q=1}^{Q-1} P_q = 138$. Given that the complexity caused by the number of complex multiplication and number of complex addition relies on the number of iterations, the number of iterations is considered in Table 1. This table shows that compared with the CCRR of the Parallel TS-PTS, the proposed technique (SBI-PTS) achieves a CCRR of 98.22%, whereas the CCRR of the proposed technique (SBI-PTS) is 98.22% compared with the ABC-PTS scheme and 88.40% compared with the SLS scheme. Clearly, the proposed technique (SBI-PTS) has the lowest computational complexity among all the compared low complexity PTS schemes.

Table 4 CCRRs of the proposed technique (SBI-PTS) compared to new existing PTS techniques for $W = 2$, $N = 256$

PTS schemes	*Iteration*	*No. of complex add.*	*Q =16*	*CCRR*
Parallel TS-PTS	T1	$N(Q-1)T1$	3,456,000	0%
SBI-PTS	Q	$N(Q-1)Q$	61,440	98.22%
ABC-PTS	T2	$N(Q-1)T2$	3,456,000	0%
SBI-PTS	Q	$N(Q-1)Q$	61,440	98.22%
SLS-PTS	T3	$N(Q-1)T3$	529,920	0%
SBI-PTS	Q	$N(Q-1)Q$	61,440	88.40%

Figure 11 CCDFs of the PAPRs of the SBI-PTS technique compared with the C-PTS and original OFDM for 16-QAM, $Q = 4$

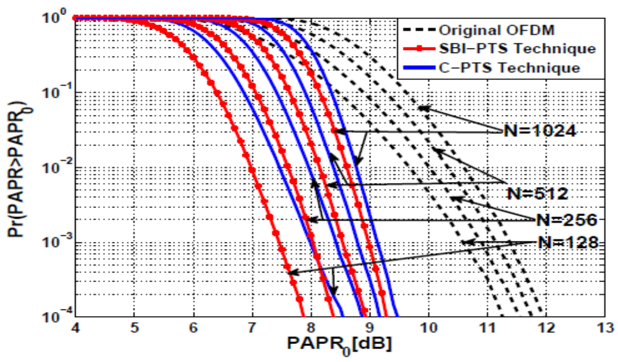


Figure 12 BER performance for the OFDM system and the SBI-PTS technique with 4-QAM, 16-QAM and $N = 256$

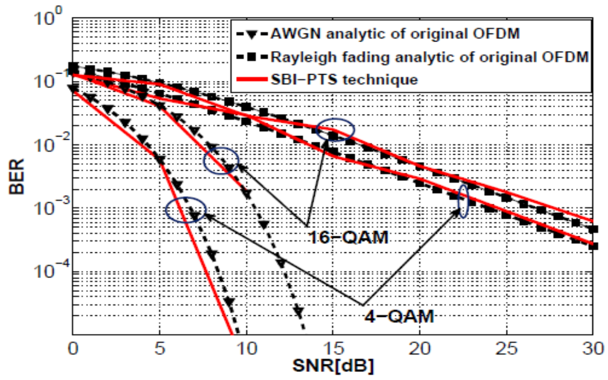
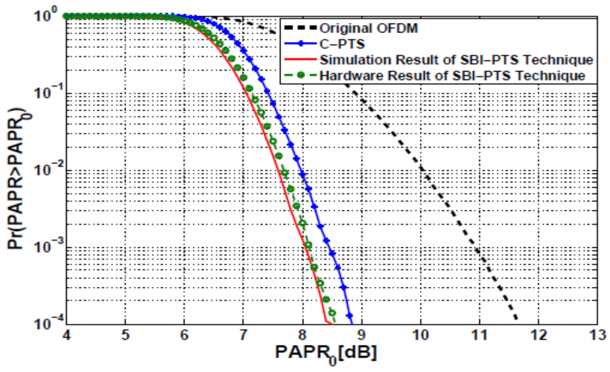


Figure 13 CCDF performance comparison between simulation and hardware for N =256 and Q =4



To evaluate the performance of the proposed technique and compare it with that of C-PTS and the original OFDM, simulations were performed using MATLAB. We employed 16-QAM modulation with various IFFT lengths of $N = \{128, 256, 512, 1024\}$, an oversampling factor of L = 4, and Q =4 subblocks. To obtain the CCDF, 10^5 random OFDM symbols were generated. The CCDFs of the proposed technique, C-PTS and the original OFDM for various numbers of subcarriers $N = \{128, 256, 512, 1024\}$ are presented in Figure 11. From this figure, it is evident that the proposed technique yields a 2 to 4 dB reduction in the PAPR with respect to the original OFDM transmission with only four iterations at a CCDF of 10^{-4}. The analytical BER expressions for M-ary QAM signalling in additive white Gaussian noise (AWGN) and multipath Rayleigh fading channel (Proakis and Masoud, 2008) are, respectively, given as:

$$P_e = \frac{2(M-1)}{M \log_2 M} Q\left(\sqrt{\frac{6E_b}{N_o} \cdot \frac{\log_2 M}{M^2-1}}\right) \tag{14}$$

$$P_e = \frac{M-1}{M \log_2 M}\left(1 - \sqrt{\frac{3\gamma log_2 M / (M^2-1)}{3\gamma log_2 M / (M^2-1)+1}}\right) \tag{15}$$

where γ and M denote E_b/N_o and the modulation order, respectively, while $Q(\ )$ is the standard Q-function defined as:

$$Q(\cdot) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{-t^2/2} dx \tag{16}$$

We employed 4-QAM and 16-QAM signalling with an IFFT length of N = 256 to evaluate the BER performance for AWGN and a multipath Rayleigh fading channel (with a maximum delay of 15 samples), and the results are presented in Figure 12. It is clear that the BER performance in the AWGN channel and the Rayleigh fading channel is consistent with the analytical results.

Hardware implementation was performed on a Xilinx Zedboard Zynq XC7Z020-1CLG484. Among five IFFT lengths, only one IFFT length is designed for hardware implementation. Hence, the IFFT length of N = 256 with the QPSK modulation scheme is adopted for implementation in the hardware environment. The amount of hardware resources used can be estimated easily via the JTAG hardware co-simulation block generation report, which will be generated at the same time the design is compiled onto the FPGA of the target board. The estimation of hardware resources used is one of the metrics of evaluation, and the hardware resources used to implement this project are listed in Table 5.

Table 5 Estimation of hardware resources used

7z020clg484-1	*Used*	*Available*	*Performance (%)*
Slice registers	25,621	106,400	24
Slice LUTs	43,408	53,200	81
Fully used LUT-FF pairs	21,187	44,119	48
Bonded IOBs	1	200	1
RAMB18E1/FIFO18E1s	12	280	4

Five main hardware resources to be evaluated are listed in Table 5, including a slice register, slice look up table, fully used lookup table–flip-flop pairs, bonded IO and RAM. The largest hardware resources demand was made on the slice LUT, as shown in Table 5, of which as much as 81% of the available amount was occupied, and most of that used was for logic. The second highest hardware resource utilisation belonged to the fully used LUT-FF pairs with 48% utilisation. From the available amount of slice registers, 24% were occupied, that is, out of 106,400 available slice registers, 25,621 were used. In further detail, out of the 25,621 used slice registers, 25,496 were as flip-flops, while another 125 were used for AND/OR logic. Lastly, the least used hardware resources included the bonded IO and RAM B18E1/FIFO18E1, of which only approximately 1% and 4% were used, respectively.

To ensure the feasibility of the hardware implementation design, the outcome of the hardware design was compared with the software simulation result through a CCDF graph. Symbols (10^4) were generated for both the software and hardware environment so that they could be compared fairly. The comparison result is presented in Figure 13. The performance of the hardware implementation is more or less the same as the performance of the software simulation, as shown in Figure 13. The PAPR reduction performed by both the software and hardware can be examined by reading the CCDF graph presented above, and the result of the software simulation is an approximately 3.3 dB reduction, while the result of the hardware implementation is an approximately 3.2 dB reduction. The difference of the results is only 0.1 dB. The minor difference of the results shows that the designed hardware block diagram that is compiled onto the FPGA of the Zedboard works as effectively as the software simulation in reducing the PAPR of the OFDM signal.

7 Conclusion

A novel PAPR technique and its FPGA implementation have been described. The technique is based on a subblocks interleaver and a new optimisation scheme in which only a two phase sequence and Q iterations are required. Interleaving is applied to reduce the computational complexity associated with weighting factors and also to reduce the PAPR and BER performance in the AWGN channel, and the Rayleigh fading channel is consistent with the analytical results. In this manner, a minimal PAPR can be obtained without the need for feedback, which conserves processing time and demands fewer computational resources, thus leading to lower complexity. Above all, this technique does not require side information and therefore offers increased transmission efficiency. Hence, compared with other PTS techniques, this technique is considered to be uniquely low in complexity and resource consumption while offering a superior performance. The FPGA implementation of this method is studied, and it has been shown that its PAPR performance is comparable with the simulation results.

Acknowledgement

This research has been funded by Universiti Putra Malaysia under the Prototype Development Research Grant Scheme (PRGS) of the Ministry of Higher Education, Malaysia (No. 5528700).

References

Al-Hussaini, K., Ali, B.M., Varahram, P., Hashim, S. and Farrell, R. (2016) 'A new subblocks interleaving PTS technique with minimum processing time for PAPR reduction in OFDM systems', *IET, The Journal of Engineering*, Vol. 1, No. 1, p.21, doi:10.1049/joe.2016.0074.

Cho, Y., No, J. and Shin, D. (2012) 'A new low-complexity PTS scheme based on successive local search using sequences', *IEEE Communications Letters*, Vol. 16, No. 9, pp.1470–1473.

Goldsmith, A. (2005) *Wireless Communications*, Chap. 12, Cambridge University Press, Cambridge.

Han, S.H. and Lee, J.H. (2005) 'An overview of peak-to-average power ratio reduction techniques for multicarrier transmission', *IEEE Wireless Communications*, Vol. 12, No. 2, pp.56–65.

Ibraheem, Z., Rahman, M.M., Yaakob, S.N., Razalli, M.S., Salman, F. and Ahmed, K.K. (2014) 'PTS method with combined partitioning schemes for improved PAPR reduction in OFDM system', *TELKOMNIKA Indonesian Journal of Electrical Engineering*, Vol. 12, No. 11, pp.7845–7853.

Jiang, T. and Wu, Y. (2008) 'An overview: peak-to-average power ratio reduction techniques for OFDM signals', *IEEE Transactions on Broadcasting*, Vol. 54, No. 2, pp.257–268.

Lain, J.K., Wu, S.Y. and Yang, P.H. (2011) 'PAPR reduction of OFDM signals using PTS a real-valued genetic approach', *EURASIP Journal on Wireless Communications and Networking*, Vol. 2011, No. 126, pp.1–8.

Liu, J., Zhang, W., Yuan, Z. and Ma, T. (2011) 'Low complexity PTS algorithm based on gray code and its FPGA implementation', *ICEMI 2011 10th International Conference*, 16–19 August, Chennai, India, pp.208–211.

Mukunthan, P. and Dananjayan, P. (2014) 'Modified PTS with interleaving for PAPR reduction of OFDM signal with QPSK subblock', *International Journal of Future Computer and Communication*, Vol. 3, pp.1–8.

Müller, S.H. and Huber, J.B. (1997) 'OFDM with reduced peak-to-average power ratio by optimum combination of partial transmit sequences', *Electronics Letters*, Vol. 33, No. 5, pp.368–369.

Pandey, P. and Tripathi, R. (2013) 'Computational complexity reduction of OFDM signals by PTS with alternate optimised grouping phase weighting method', *International Journal of Computer Applications*, Vol. 78, No. 1, pp.1–7.

Proakis, J.G. and Masoud, S. (2008) *Digital Communications*, McGraw-Hill, New York.

Rahmatallah, Y. and Mohan, S. (2013) 'Peak-to-average power ratio reduction in OFDM systems: a survey and taxonomy', *IEEE Communications Surveys and Tutorials*, Vol. 15, No. 4, pp.1567–1592.

Sabbir, A. and Makoto, K. (2013) 'Interleaving effects on BER fairness and PAPR in OFDMA system', *Telecommunications Systems*, Vol. 52, pp.183–193.

Taspinar, N., Kalinli, A. and Yildirim, M. (2011) 'Partial transmit sequences for PAPR reduction using parallel tabu search algorithm in OFDM systems', *IEEE Communications Letters*, Vol. 15, No. 9, pp.974–976.

Varahram, P. and Ali, B.M. (2011) 'FPGA implementation of novel peak-to-average power ratio reduction in orthogonal frequency division multiplexing systems', *2011 17th Asia-Pacific Conference on Communications (APCC)*, 2–5 October, Sabah, Malaysia, pp.264–268.

Wang, Y., Chen, W. and Tellambura, C. (2010) 'A PAPR reduction method based on artificial bee colony algorithm for OFDM signals', *IEEE Transactions on Wireless Communications*, Vol. 9, No. 10, pp.2994–2999.