Hussein Shakor Mogheer [1,*]

Khamees Khalaf Hasan [2]

[1] Department of Communications
Engineering
College of Engineering
University of Diyala
Diyala, Iraq

[2] Department of Electrical Engineering
College of Engineering
Tikrit University
Tikrit, Iraq

Implementation of Clock Gating for Power Optimizing in Synchronous Design

ABSTRACT

Huffman coding is very important technique in information theory. Compression technique is the technology for reducing the amount of data used to denote any content without decreasing the quality. Furthermore, Clock gating is an effective method for decreasing power consumption in a sequence design. It saves more power by dividing the main clock and distributing the clock to the logic blocks only when there is a need for those blocks to be activated. This paper aim to design Huffman coding and decoding process with proposing a novel method of clock gating to achieve low power consumption. Huffman design is executed by expending ASIC design procedures. With the purpose of executing the encoder and decoder structures, 130 nm typical cell technology libraries are utilized for ASIC implementation. The simulations are completed by utilizing Modelsim tool. The design of coding and decoding process has been made using Verilog HDL language. Moreover, it carried out using Quartus II 14.1 Web Edition (64-Bit).

تطبيق تقنية النبضة لتحسين الطاقة في التصميم المتزامن

الخلاصة

تشفير هوفمان هي تقنية مهمة جدا في نظرية المعلومات. تقنية الضغط هي التكنولوجيا لتقليل حجم البيانات المستخدمة لتمثيل أي محتوى بدون التاثير على الكفاءة. علاوة على ذلك, تقنية النابضة هي فعالة للحد من استهلاك الطاقة في تصميم الدوائر التتابعية. حيث انها توفر المزيد من الطاقة عن طريق تقسيم النبضة الرئيسية وتوزيع النبضة إلى كتل المنطق فقط عندما يكون هناك حاجة لتلك الكتل لتفعيلها. يهدف هذا العمل لتصميم هوفمان لضغط وعملية فك الضغط مع اقتراح طريقة جديدة بتقنية النابضة لتحقيق انخفاض استهلاك الطاقة. تصميم هوفمان نفذ باستخدام إجراءات تطبيق محددة الدوائر المتكاملة. لغرض تنفيذ تركيب الضغط وفك الضغط, المكتبة 130 نانومتر استخدمت لتنفيذ تطبيق محددة الدوائر المتكاملة. تم إنشاء بنية التشفير وفك التشفير باستخدام لغة الفريلوك HDL لتغطية جميع وظائف. بالإضافة إلى ذلك, تم تنفيذ المحاكاة باستخدام ModelSim-ألتيرا (Quartus II 14.1) طبعة الإنترنت (64 بت). في النهاية, هذه التقنية تؤدي الى تقليل في استهلاك القدرة الديناميكية الى53 %. لذلك التصميم المقترح فعال للاستخدام في النظم لتقليل استخدام الطاقة في الاجهزة الحديثة.

1. INTRODUCTION

Text data compression is a demonstration process of the data. The original text is encoded with small quantities of bits. Data compression goal is to lessen the redundancy of the content and accumulate or transmit a particular in an effective technique [1]. The advantage of this technique to epitomize Huffman decoder is that the original data is less memory size required and renovation promptly [2]. Based on computation algorithms, compression processes are divided into two categories such as loss/lossless [3]. In general, medical images are compressed in a lossless way

in order to reservation details and to avoid incorrect analysis. Huffman coding target is reducing data size as it is a type of the Variable Length Codes (VLC). The supports of this technique are actual use of channel bandwidth and data packing size [2]. This source coding procedure declines bits quantity in the information once it equaled to the ASCII demonstration of the sequence [4]. Text coding is an application of information confining that encrypts the initial paragraph using several parts [5]. Purpose of information compression is to decrease repetition of the document in order to hold or send a special through a beneficial method [6]. Furthermore, the process

* Corresponding author: E-mail : en.hussein.eee13@gmail.com

requirement of low amount of power and great performance in electronic devices is pointed to the study of low. Fig. 1 presents design input-output ends, data length for each block of both the decoder and encoder and then adding a novel signal to show the encoder output data.
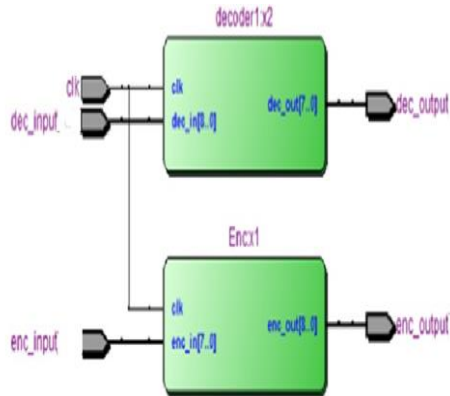

Fig. 1. Top-level Huffman design.

2. SYSTEM ARCHITECTURE

The suggested technique is included two sections: compression section (encoder) and decompression section (decoder) as shown in Fig. 1. First module is assumed to get 8-bit data through frequencies starts from (20, 40, 60, 80, 100, 120,140,160 and 180) MHz. Then it clocks and encodes it into 9 bits code. Essentially, second module utilized to decompress input data in form of 9 bits output text data. Conversely, decoder part utilized to decode information quantity of 8 bits output data (the equal arrangement of the data input). Hardware Description Language (HDLs) and their simulators allow designers to partition their system into components that can work correctly and communicated with one another [7]. Huffman design consists of two main modules which are encoder and decoder in addition to the code of top level Huffman. That implies that full designs involve three codes, two for modules and one for top-level design. Furthermore, each code design should have a corresponding code for test bench. All code designs and test bench are written in Verilog HDL language.

2.1. Implementation of Encoder

In this work, the encoder is implemented using Huffman tree. The last one is implemented in Verilog platform using the binary tree. This Huffman tree is stored in the LUT to give the corresponding encoded output corresponding character. The encoder retrieves the code for each symbol from a map and shifts it out one bit at the time. The decoder is obtained from the tree by adding acts from the leaves back to the top of the tree. If a state is not a leaf of the tree and its encoding is n, then the encodings of its two children are 2n+1 and 2n+2 respectively. Character input which is given to the encoder acts as input to the LUT which gives corresponding encoded word on the data bus which is given to a shift register so as to serially shift the data out. As it is a variable length coding, in order to determine the end of the code word for each character while shifting out, one more bit is added to the end of the code word in the LUT is made as 1. The code word is

logically shifted out till it contains only 1 at its LSB. Then, next character is loaded from the comparator. Apart from this, the encoder should generate an enable signal to the decoder so that the decoder knows when the valid data is presented to it.

2.2. Implementation of Decoder

Huffman decoding method is somewhat more complex. Both encoding and decoding should be done with respect to the same tree. Therefore, same data which is stored in the encoder LUT is stored in the decoder LUT in a different way. In Fig. 2, the block diagram of Huffman decoder which clearly explains the operation is shown. The block diagram is for a decoder in which the coded value is first stored in the buffer and then shifted using a LIFO. The shifted value is then stored in the temporary register which is then compared with respective codes stored in the LUT. Then, the character is finally decoded. In this method, inside the decoder block first a buffer is first presented inside the decoder block in order to store the output of the encoder part. A Last Input First Output (LIFO) which will shift the coded values stored inside the buffer is presented next to it. This shifted code is then stored inside a temporary register of bit size. Both the coded value and predetermined Huffman tree which are stored inside the LUT are compared to obtain a decoded output with respect to the corresponding coded state. Fig. 3 shows the RTL viewer of top-level Huffman design without PMC.
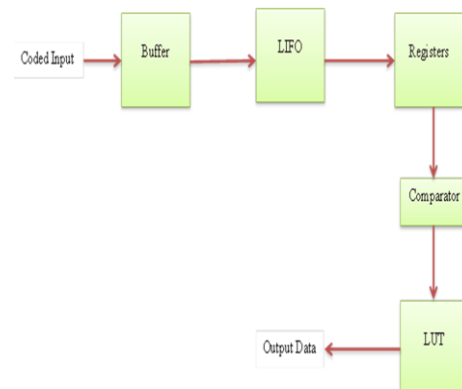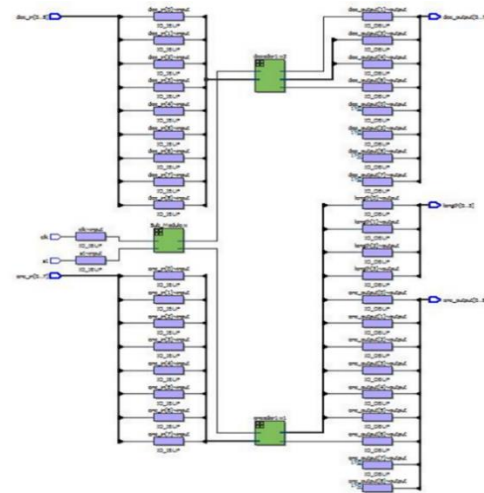

Fig. 2. Block diagram of decoder Huffman.

Fig. 3. Technology map viewer of Huffman with clock gating.

2.3. Constructing the Tree

In this method, a Huffman design is implemented using binary tree to get the smallest size of data compression, which is built upon using the frequencies corresponding to the characters. Figure 4, shows a binary tree in which the branch values are arranged based on its construction using the characters and respective frequencies. The main idea of compression is to assigns smaller codes to the character that happen more frequently and longer codes to this character happen less frequently. Both encoding and decoding should be done on the same tree. Therefore, the data stored in the encoder block is stored in the encoder LUT.
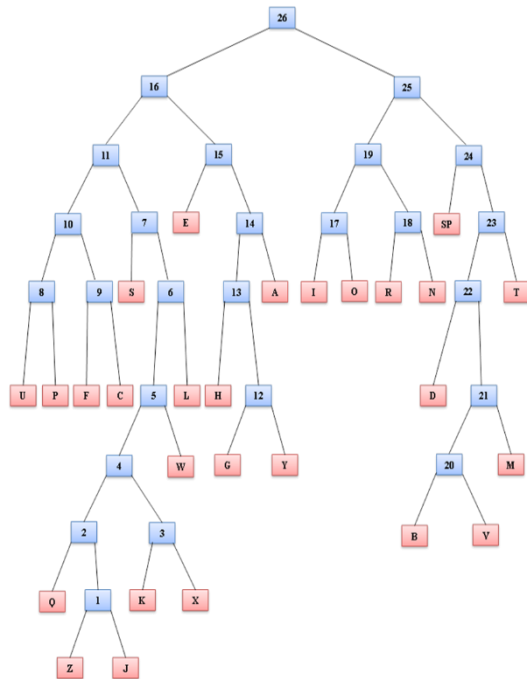

Fig. 4. Huffman binary tree.

3. HDL SIMULATION

Quartus II I4.1 software has a simulator which can be used to simulate the behaviour and performance of Huffman design for implementation in Altera's programm-able logic. A simulator used to test designed Huffman and observe the outputs produced in response. Additionally, to being able to see the validated values on the input and output pins of the design, it is also possible to probe the internal nodes of the system. The simulator makes use of the waveform editor which makes representation of signals easy. A testable hardware of Huffman coding in HDL for the manipulation and valuation of Huffman design can be completed at the concurrently. Hence, it is not only the design main and its test bench verification that can be sophisticated with a regular algorithm, while entirely the abilities of software (like structures, usage of functions and complicated data) are made obtainable. Programming Language Interface (PLI) gives the required paths to the internal information construction of the organized form. Therefore, test processes can be implemented in such a mixed situation also simply without combining with original design. In Huffman design, language constructs in agreement to Verilog semantics and syntax realize the

internal part of a module. These constructs are achieved to facilitate the depiction of hardware components in view of Huffman design processes such as simulation, synthesis, and specialization of test benches to identify test data and observer track responses. In this order, it is considered such as the test bench of the design. Fig. 5 presents the process of validation model that include the design using a HDL test bench. Verilog forms (be seen by dotted point) of its design being tested is capable in view of the classification of Verilog hardware design. On the other hand, language concepts utilized in a testbench are used to provide suitable input information or apply data kept in a text file of the module being verified to analyze or show its outputs.
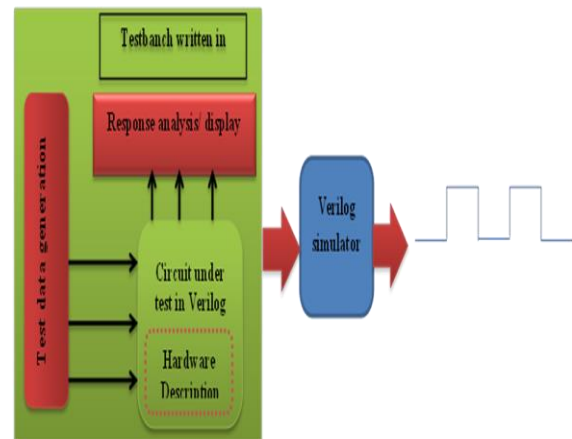

Fig. 5. Process of simulations in Verilog.

4. SYNOPSYS POWER COMPILER RESULTS

The key to appropriate power analysis tools is the automatic reducing power method. This way it benefits for designers to match power statements without degrading outcomes or time of Huffman. Synopsys power compiler is a tool used to automatically downsizing power dissipation at the Gate Level and Register Transfer Level of design. At the system elaboration mode of RTL, the power compiler performs automatic clock gating to decrease the power dissipation. After uploading a full Huffman design in Synopsys tool, with specific system restrictions, the power compiler implements improvements for the area, timing, and power with each other [7]. Fig. 6, shows input requirements for Synopsys tool to produce the netlist.
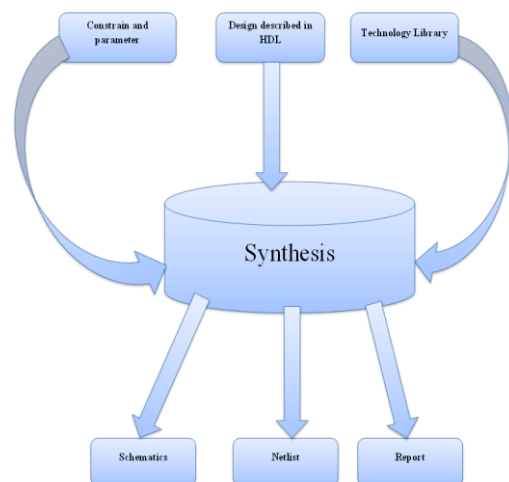

Fig. 6. Inputs and outputs of synthesis process.

Simulation mechanism is used as Synopsys instrument is the Synopsys Verilog compiler (SVC). It is essentially designed to authenticate and debug plans. Debugging the module is realized with trails principal, which are:

1- Compiling the source code.
2- Implementing the verification.
3- Viewing and debugging the waveform results.

4.1. Power Definition

Clearly, design entire power dispersion is usually consisting of two measures: the static and dynamic power [9]. These are the foundation of consumed power in CMOS circuit's elements and static and dynamic depletion [10] as shown below:

$$P = P\text{static} + P\text{dynamic} \qquad (1)$$

4.1.1. Dynamic Power

Two main sections are involved of dynamic power dissipation: one is the short circuit power as a result of the nonzero increase and decrease time of input waveforms and in addition, is the switching power as a result of charging and discharging of load capacitance. The switching power of a particular gate can be specified by [10]:

$$PD = \alpha\, CL\, VDD2f \qquad (2)$$

where
α is the converting activity, f is the process frequency, CL is the load capacitance, VDD is the source voltage. The short circuit power of an unloaded inverter can be roughly known by:

$$PSC = \beta\,(VDD - Vth\,)\,\frac{3\,\tau}{12T} \qquad (3)$$

where
β is the transistor constant, τ is the increase/decrease time, T (1/f) is the delay.

4.1.2. Leakage Power

While a transistor is not switching, the Static (leakage) power is generated. It comes from sub threshold, gate and junction leakage currents [9]. Leakage power is correspondent to dynamic power in state of nanometer approaches with less threshold voltages and thin gate oxides. In certain cases, it can even control the generally depletion power [11]. The analysis of the design to both static and dynamic powers can be completed by means of Synopsys scheme compiler logic synthesizer. The design has been mapped with 130 nm technology and ASIC strategy structure, which is official for Huffman design, as a results it harvests a low power consumption and comparatively upper performance [12].

5. SIMULATION RESULTS

Lettering three testbanch codes for Huffman design (encoder, decoder, top_level) are resulted to execute the recommended technique of Huffman encoder and decoder utilizing Verilog HDL and simulate them with ModelSim-Altera 10.3c (Quartus II 14.1) Starter Edition after. Fig. 7,

outlines the waveform of Huffman encoder utilizing only one signal for input data and two signals for output (length & encoder output), where the output length is demonstrated by the additional length signal to the output of the compression plan.

In Fig. 8, the utilitarian simulations for that Huffman decoder would implement utilizing Modelsim outfits. In the event of text file both encoder furthermore decoder is attainable. The decoder is utilized may be used on decipher the Huffman encoded information. The input and the output ports should be well-defined before composing any program in Verilog [13].

The waveform of Huffman methodology where the input furthermore output signals for encoder what's more decoder need aid advertised throughout the simulation top-quality Huffman is shown Fig. 9.
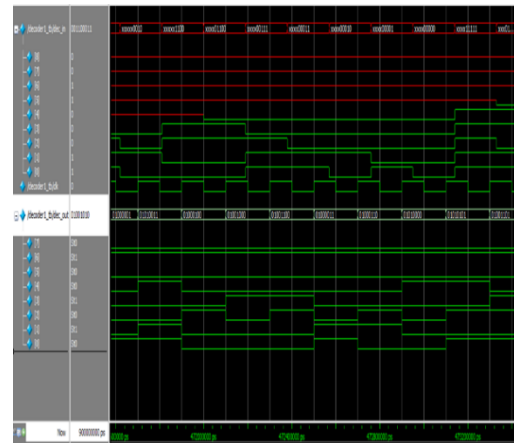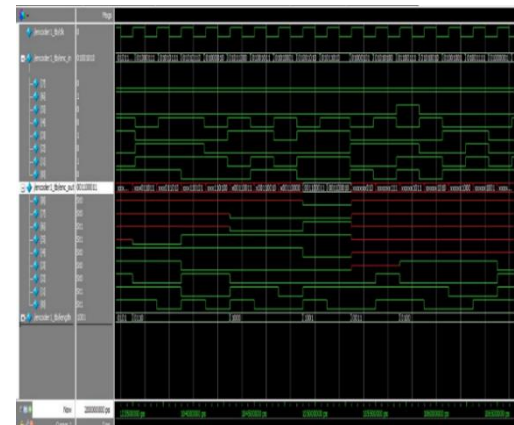

Fig. 7. Encoder simulation.

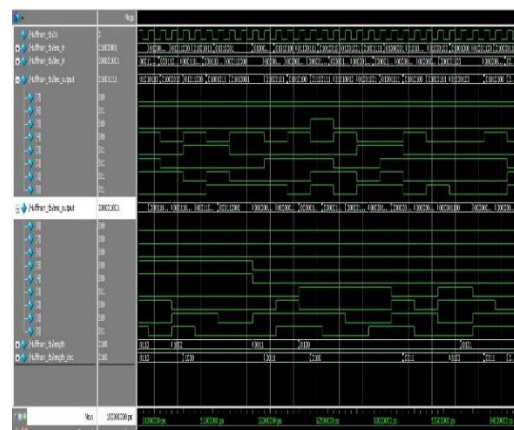**Fig. 8.** Decoder Simulation.

Fig. 9. Huffman design simulation.

The design limitation has scales of frequencies which utilized within the proposed plan. Each frequency read for dynamic power in two categories (switching & internal) and leakage power as shown in Table 2. In each process of analysis, the frequency the value of slack time and critical point for the design should be considered. Table 1 represents power analyses for Huffman design in original state without using clock gating performance.

Table 1
Power report for traditional Huffman design.

Frequency (MH)	Time (NS)	Internal power (mw)	Switching power (mw)	Dynamic power (mw)	Leakage power (mw)	Total power (mw)
20	50	0.0146	0.0012	0.0159	0.0002	0.0161
40	25	0.0293	0.0025	0.0318	0.0002	0.0320
60	16.6666	0.0437	0.0036	0.0473	0.0002	0.0476
80	12.5	0.0583	0.0049	0.0632	0.0002	0.0635
100	10	0.0731	0.0062	0.0793	0.0002	0.0796
120	8.3333	0.0877	0.0074	0.0952	0.0002	0.0954
140	7.1428	01028	0.0088	0.1115	0.0002	0.1118
160	6.25	0.1174	0.0100	0.1273	0.0002	0.1276
180	5.5555	0.1314	0.0110	0.1423	0.0002	0.1426

6. AND BASED CLOCK GATING DESIGN

In sequential circuit, two input AND gates are inserted in the logic for clock gating [14]. Clock gating for power reduction is considered. This predominant technique is used for power saving. In this step of the design, the procedure is to connect two AND gates connected in such a way that a clock signal that is able to switch on one module and switch another one off is created. AND enable signal with clock signal is also created for the first AND gate. However, an inverter is put in the second AND gate to organize the output signal between encoder and decoder as shown in Fig. 11. Huffman project deliberated in this paper involves of two efficient blocks modules: encoder as well as decoder. Every module is synchronous which proceeds the inputs are evaluated on the existence of the clock. The AND gating with clock consequence technique can be applied to the Huffman design, where only one functional division (encoder or decoder) activates while the additional does not. Therefore, the enable input of the AND based will give the clock pulse either the encoder or decoder part [15]. RTL viewer of AND based as planned in Verilog is shown in Fig. 10. The RTL view of the schematic Huffman design is shown in Fig. 1. The clock is connected to either the encoder or the decoder according to the enable input. The synthesis produces the device progression for the netlist in addition to the verification. The netlist detail displays that the AND based clock gating used less power consumption, but it generates higher delay for the same frequency used. Firstly, a clock signal that has ability of achieving the function of clock gating is created. When applying the clock signal to the top-level module, the target module is switched on and the second module is switched off. Through this, power consumption can be significantly reduced as shown in Fig. 12. Each frequency read for dynamic power in two categories (switching & internal) and leakage power with clock gating as shown in Table 2. Gain ratio for power reduction was calculated according the equation below [16]:

$$\text{Percentage Power Reduction} = \frac{\text{total power without clock gating} - \text{total power with clock gating}}{\text{total power without clock gating}} \times 100 \qquad (4)$$

Table 2
Power report for clock gating Huffman design.

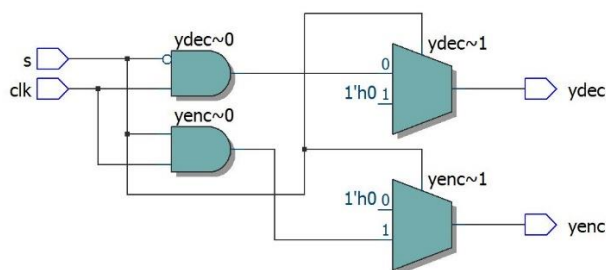| Frequency (MH)	Time (NS)	Internal power (mw)	Switching power (mw)	Dynamic power (mw)	Leakage power (mw)	Total power (mw)	Percentage (%)
20	50	0.0070	0.0004	0.0074	0.00009	0.0075	53.4
40	25	0.0141	0.0008	0.0149	0.00009	0.0150	53.1
60	16.6666	0.0211	0.0012	0.0223	0.00009	0.0224	53
80	12.5	0.0282	0.0016	0.0298	0.00009	0.0299	53
100	10	0.0353	0.0020	0.0373	0.00009	0.0374	53
120	8.3333	0.0423	0.0024	0.0447	0.00009	0.0448	53
140	7.1428	0.0495	0.0028	0.0524	0.00009	0.0525	53
160	6.25	0.0564	0.0032	0.0597	0.00009	0.0598	53.1
180	5.5555	0.0635	0.0036	0.0671	0.00009	0.0672	53

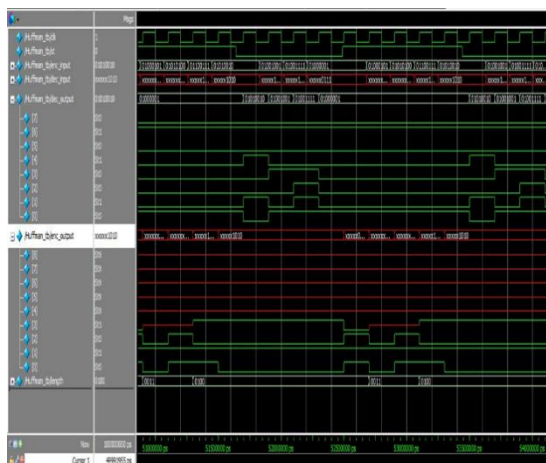**Fig. 10.** RTL viewer of clock gating technique.

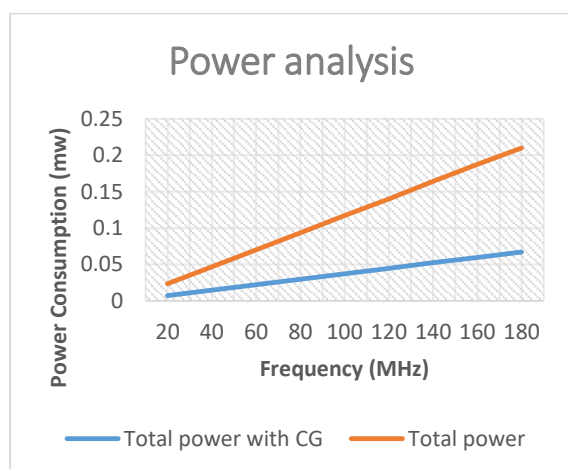**Fig. 11.** Waveform simulation of Huffman with clock gating.

Fig. 12. power consumption analysis.

According to Table 2, the output of the design reduced power consumption up to half or more for different scale of frequency used for each modules of Huffman design. Table 2 shows the power consumption percentage of Huffman output signals. The maximum power consumption percentage is achieved in Huffman design with clock gating at frequency 20MHz. It reduces the power consumption up to 53.4% of the total power consumption using the same frequency scale applied in Huffman without clock gating technique, gate-level optimization operates at the netlist generated by logic gate synthesis into generating a technology precise netlist. It involves mapping, delay optimization and design rule fixing. Furthermore, design compiler attempts to fix violations without affecting area results and timing, but if required, it does violate.

7. CONCLUSION

Huffman plan is realized on ASIC stages with the binary tree. Table 1 presents Huffman power study results in traditional state. It gives all the power types are consumed for various scales of frequencies. Synopsys power compiler is used for logic synthesizer. The design has been planned with library 130 nm technology for analysis power constrains. In this paper, clock gating method with AND based clock gating is stated to Huffman design. Table 2 shows Huffman power analysis results by using clock gating. The power study is approved in place of a range of clock frequencies with all the forms of executed and analyzed. According to the obtained results, the power reduces up to 53% for each scale of frequency used. There is evident reduction in total power consumption by means of AND based clock gating usage.

REFERENCES

[1] Almelkar M, Gandhe S. Implementation of lossless image compression using FPGA. *International Journal of Emerging Technology and Advanced Engineering* 2014; **4**: 2250-2459.

[2] Suvvari V, Murthy MB. VLSI Implementation of Huffman decoder using binary tree algorithm. *International Journal of Electronics and Communication Engineering &Technology* 2013; **4** (6): 85-92.

[3] Marimuthu M, Muthaiah R, Swaminathan P. An overview of image compression techniques. *Research Journal of Applied Sciences, Engineering and Technology* 2012; **4** (24): 5381-5386.

[4] Anjana PM, Ajeesh AV. FPGA based iterative JSC decoding of Huffman encoded data for a communication system. *International Journal of Engi-neering Research and Technology* 2012; **3** (2): 1811-1814

[5] Hameed M, Khmag A, Zaman F, Ramli AR. A new lossless method of Huffman coding for text data compression and decompression process with FPGA implementation. *Journal of Engineering and Applied Sciences* 2016; **100** (3): 402-407.

[6] Almelkar M, Gandhe S. Implementation of lossless image compression using FPGA. *International Journal of Emerging Technology and Advanced Engine-ering* 2014; **4**: 2250-2459.

[7] Brown SD. Fundamentals of digital logic with Verilog design: Tata McGraw-Hill Education; 2007.

[8] Jayasekar A. Low power digital design using asynchronous logic. Ph.D. Thesis. San Jose State University; Washington, United States: 2011.

[9] Aanandam SK. Deterministic clock gating for low power VLSI design. Ph.D. Thesis. National Institute of Technology; Rourkela: 2007.

[10] Nejat M, Abdevand MM, Farahani AM. A novel circuit topology for clock-gating-cell suitable for sub/near-threshold designs. *Computer Architecture*

and Digital Systems (CADS), 2013 17th CSI International Symposium on: IEEE; 2013. pp. 45-49.

[11] Kang L. Design and implementation of a decompression engine for a Huffman-based compressed data cache. MSc. Thesis. Chalmers University of Technology; Göteborg, Sweden: 2014.

[12] Soundarya G, Bhavani S. Comparison of hybrid codes for MRI brain image compression. *Research Journal Applied Science Engineering & Technology* 2012; **4** (24): 5367-5371.

[13] Maadi M. An 8b/10b encoding serializer/ deserializer (serdes) circuit for high speed communication applications using a dc balanced. Partitioned-Block, 8b/10b Transmission Code: 2015.

[14] Kathuria J, Ayoubkhan M, Noor A. A review of clock gating techniques. *MIT International Journal of Electronics and Communication Engineering* 2011; **1** (2): 106-114.

[15] Mitra A. Design and implementation of low power 16 bit ALU with clock gating. *International Journal of Advanced Research in Computer Engineering & Technology* 2013; **2** (6): 2139-2142.

[16] Kulkarni R, Kulkarni S. Energy efficient implementation of 16-Bit ALU using block enabled clock gating technique. *India Conference (INDICON), 2014 Annual IEEE*: IEEE; 2014. pp. 1-6.