Jurnal Teknologi

Full paper

ASIC Design of Natural Frequency of ECG Signal for Atrial Fibrillation Detection Module using High-Level Synthesis Approach

Nurul Ashikin Abdul-Kadir, Norlaili Mat Safri*, Mohd Afzan Othman

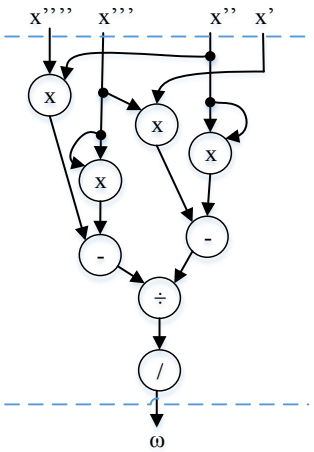
Faculty of Electrical Engineering, Universiti Teknologi Malaysia, 81310 UTM Johor Bahru, Johor, Malaysia

Corresponding author: norlaili@utm.my

Article history

Received : 25 March 2015
Received in revised form :
11 April 2015
Accepted : 13 April 2015

Graphical abstract



Abstract

The growth of interest in the development of reduced-scale electrocardiogram (ECG) system based on field-programmable gated-array (FPGA) design platform is increasing. This study provides initial result of mapping digital signal processing to hardware design for specific purpose. In this paper, a part of digital signal processing for atrial fibrillation classification was implemented to register-transfer level (RTL) design. The specific part was feature extraction of ECG signal. The algorithm of ECG signal feature extraction was natural frequency from second-order system for detecting atrial fibrillation. By applying high-level synthesis method, three designs were implemented for natural frequency behavior. The designs were two Single-Cycle (Design 1 and Design 2) and Multi-Cycle fully-constraint (Design 3), of which logic utilization consist of 2530, 36 and 1, respectively. Performance evaluation among all designs were compared.

Keywords: Atrial fibrillation; electrocardiogram; high-level synthesis; logic utilization; natural frequency; second-order system

Abstrak

Pertumbuhan kepentingan dalam pembangunan sistem ECG berskala kecil berdasarkan platform reka bentuk FPGA telah muncul. Kajian ini menyediakan keputusan awal memeta pemprosesan isyarat digital kepada reka bentuk perkakasan yang tertentu. Dalam kajian ini, sebahagian daripada pemprosesan isyarat digital untuk pengelasan AF dilaksanakan kepada reka bentuk RTL. Bahagian khusus tersebut ialah penyarian sifat isyarat ECG. Algoritma di penyarian sifat ialah frekuensi asli isyarat ECG dari sistem tertib kedua untuk mengesan AF. Dengan mengaplikasikan kaedah *high-level synthesis*, tiga reka bentuk diaplikasikan untuk sifat frekuensi asli. Reka bentuk tersebut adalah dua *Single-Cycle* (*Design 1* dan *Design 2*) dan *Multi-Cycle (Design 3)*, yang mana *logic utilization* merangkumi 2530, 36 dan 1, masing-masing. Persembahan antara seni bina yang direka bentuk dibandingkan.

Kata kunci: Fibrilasi atrium; elektrokardiogram; sintesis aras-tinggi; frekuensi semula jadi; sistem tertib kedua

© 2015 Penerbit UTM Press. All rights reserved.

1.0 INTRODUCTION

Emergence of healthcare services in acute care can provide patients with more potential benefits such as in intensive care and medical imaging services [1–3]. Electrocardiograph (ECG) monitoring device plays vital role to measure and diagnose human heart conditions or any irregularities occurred. Furthermore, the growth of very large scale integrated circuits (VLSI) in medical diagnostic tools for digital signal processing applications can provide low cost ECG with real-time processing, suitable for monitoring and alert system. Such inventions can benefit patients, i.e. patients can do their regular habit as their mobility increased, reduce stress of being connected to medical system through wireless data transmission, give immediate analysis, lower price, and ease of monitoring. Specifically, digital signal processing applications in electrocardiogram (ECG) system with a particular emphasis on the impact of embedded system / system-on-chip (SoC) have emerged. Digital bio-signal processing system of ECG for detection, classification [4–8] and data compression [9, 10] had been developed using FPGA platform.

Heart abnormality such as atrial fibrillation (AF) is a problem of the heart's electrical system. AF can disturb the normal rhythm between the atrias and ventricles. Heart beat can increase to 600 beats per minute with ventricular rates above 100 beats per minute. Atrias can no longer pump blood to ventricles and cause heart attack, high blood pressure, coronary heart disease, or heart valve disease, or sometimes can cause no symptoms. When atrial muscle contraction fibrillate irregularly, or AF, ischemic stroke could happen [11] Women at higher risk of AF which need more genomic, clinical and biomarkers

profile [12] for the society not just focused on certain population [13, 14].

A previous study has shown the viability of second order system to classify ECG signal between normal sinus rhythm of healthy human and atrial fibrillation patient [15, 16]. Therefore this study illustrates the high-level design method to develop atrial fibrillation (AF) digital signal processing application-specific module in hardware design. The hardware designed in this study embraced specific purpose by relating natural frequency, ω, algorithm of ECG signal from second order system into hardware language for FPGA prototyping. Steps taken starting from the algorithmic level to construct the register-transfer level (RTL) model for the proposed design were discussed. High-level synthesis is the mapping of a behavioral description of a digital system into RTL design, or transforming the algorithm into an RTL design [17].

2.0 METHODOLOGY

Several techniques must be highlighted to map the algorithmic description in the form of data flow graph (DFG) for RTL design which is suitable for FPGA prototyping of AF detection system. Three phases were included, parsing, transformation and RTL synthesis, to implement the hardware design [18–20]. Parsing phase was where less resources are needed by factoring the algorithm or removing redundant operations in the algorithm. Transformation phase was where DFG corresponding to the algorithm are allocated and scheduled to minimize resources and whole computation time. Finally the RTL synthesis phase was where RTLs were coded and designed for the specific algorithm. This study used algorithm of natural frequency of ECG signal which was derived from second-order system. The selected algorithm was chosen based on previous study to characterize and classify atrial fibrillation and normal sinus rhythm [15, 16].

2.1 Phasing Phase

Derivation of algorithm in Equation (1) were discussed in previous study [15,21] where ω is natural frequency of ECG signal; x, x', x'', x''' and x'''' are differential of x over time, t, of first to forth derivation respectively.

$$\omega^2 = \frac{x''.x''''-(x''')^2}{x'.x'''-(x'')^2} \quad (1)$$

A value of ω can be obtained by having all the parameters (x', x'', x''' and x'''') in time-domain, which consider the value of x for five times at a time. Additionally, Equation (2) shows the algorithm of x'''' which needs five value of x before it can be calculated (i.e. x_0, x_{-1}, x_{-2}, x_{-3}, x_{-4}, which shows x value at time 0, 1, 2, 3, 4, respectively, where '-' indicates time delay).

$$x_n'''' = \frac{x_{n-4}-4x_{n-3}+6x_{n-2}-4x_{n-1}+x_n}{t^4} \quad (2)$$

As an early step, Equation (1) is expressed with variables as Equation (3), where x', x'', x''' and x'''' were defined by m, n, p, and q, respectively. The first to fourth derivation of x in (1) are expressed by Equation (4) to (7), respectively. Parameters a, b, c, d, and e represent x value in time delay of x_n, x_{n-1}, x_{n-2}, x_{n-3}, x_{n-4}, respectively [21]. This to ensure the mapping of algorithm to hardware design is correct and later to be used with continuous signal, or apply pipelining technique in hardware design. Referring to Equation (2), the most suitable value of t is 4 [15].

$$w = \frac{n.q-p.p}{m.p-n.n} \quad (3)$$

$$m = \frac{b-a}{t} \quad (4)$$

$$n = \frac{c-2b+a}{t*t} \quad (5)$$

$$p = \frac{d-3c+3b-a}{t*t*t} \quad (6)$$

$$q = \frac{e-4d+6c-4b+a}{t*t*t*t} \quad (7)$$

2.2 Transformation Phase

Algorithms as in Equation (3) to (7) were transformed into dataflow graph (DFG). This study comprised of three architectures in transformation phase, i.e. Design 1, Design 2, and Design 3. Design 1 is single-cycle design as Equation (3) to (7). Design 2 is the conversion of Equation (3) to (7) by replacing t with 4. Design 3 is multi-cycle fully-constraint design, using only 1 multiplier/divider and 1 add/subtract in a cycle.

Total of seven resources were needed to map the algorithm of Equation (3) in single-cycle design while two resources (1 multiplier/divider and 1 adder/sub) were allowed in multi-cycle fully constrained design. Additional resources were needed to construct the RTL design for natural frequency module. These are shown by Equation (4) to (7) for parameters needed. Therefore, the designs were instantiated into 5 submodules correspond to algorithms of Equation (3) to (7). All three designs included of 5-submodule, namely *m*-, *n*-, *p*-, *q*-, and *w*-modules. The top-level module was named as *NaturalFrequencyModule*.

Figure 1 shows the DFG for (a) single-cycle design and (b) multi-cycle design with optimal resource utilization of *w*-module. Figure 1(c) shows the top-level module for the *NaturalFrequencyModule*. There were five sub-modules which correspond to each algorithm of Equation (3) to (7). Modules *m*-, *n*-, *p*-, and *q*- are concurrent to each other and sequential with *w*-modules.

2.3 RTL Synthesis Phase

Each algorithm was included into top module as shown in Figure 1(c). Three design types were synthesized using SystemVerilog HDL. The platform was Quartus II v13.1. Example of RTL code is shown in Table 1.

Table 1 RTL code for *q*-module (S = State)

Design 1	Design 2	Design 3
S1 : q ← (6*c – 4*b + a + (e – 4*d)) / (t*t*t*t) ;	S1 : q ← (6*c – 4*b + a + (e – 4*d)) / 256 ;	S1 : R1 ← R1 – 4*R2 ; S2 : R1 ← 6*R3 + R1 ; S3 : R1 ← R1 – 4*R3 ; S4 : R1 ← (R1 + R5) / 256 ;

Table 1 shows the RTL code for *q*-module of three designs under study. The *q*-modules for all design were chosen as it employs maximum number of inputs (*a*, *b*, *c*, *d* and *e*) before the output (of the *q*-module, that is *q*) can be generated. *q*-module involved four states in Design 3. The *q*-module needed more inputs than other submodules (*m*-, *n*-, *p*-, *q*-modules) to provide the correct output of the module. The *w*-module consists of five

states in Design 3, which was previously shown in Figure 2(b). Since *m*-, *n*-, *p*-, and *q*-modules are concurrently arranged in the top-level design, therefore the required states for those modules were synchronized. Thus the outputs of *m*-, *n*-, *p*-, and *q*-modules will be the synchronized inputs to *w*-module.

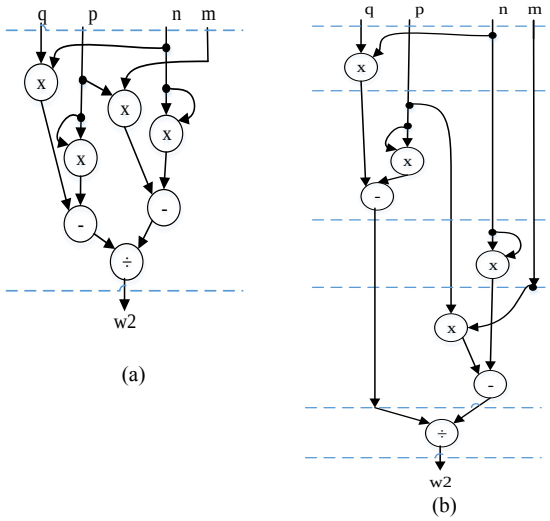


Figure 1 (a) Design 1 : single-cycle design, (b) Design 2 : multi-cycle design, (c) functional block diagram of top-level module, namely *NaturalFrequencyModule*

3.0 RESULTS AND DISCUSSION

The algorithms of Equation (3) to (7) were defined as sub-module; *w*-module, *m*-module, *n*-module, *p*-module and *q*-module, as in Figure 1(c) respectively. Each sub-module was synthesized separately in QuartusII. From Figure 1(c), the outputs of *m*-, *n*-, *p*-, *q*-modules were connected to be the inputs for *w*-module, while the inputs for the top-level module, i.e. *NaturalFrequencyModule* were signals *a*, *b*, *c*, *d* and *e*, and the output is *w*.

Initially, the inputs were stored in registers during first state. Mapping of algorithm to hardware design was realized by designing one state of single-cycle design. The difference between single-cycle designs (Design 1 and Design 2) was the used of fixed value of *t* (Design 2), instead of using multiplier(s) to find the multiplication of *t* (Design 2). Referring to algorithms of Equation (4) to (7) of which *t*s were replaced by integer 4, 16, 64, and 256, respectively, in Design 2. Design 1 used Equation (4) to (7) directly as shown in the algorithms. Theoretically, *t* is time, where in our previous study had shown the optimum value of *t* was 4 seconds for characterizing AF from ECG signal [15,16]. There was only one state designed for both types of single-cycle design (Design 1 and 2). Three, four and five states were involved in rescheduling *p*-, *q*- and *w*-modules while one and two states involved in *m*- and *n*-submodules, respectively, for Design 3. Since the design of *m*-, *n*-, *p*-, *q*-modules were concurrent to each other, the output of those modules were synchronized. The number of states of each module are summarize in Table 2.

Table 2 The number of states of each module

	Module	Design 1	Design 2	Design 3
Total States	*m*	Each module has a state	Each module has a state	1
	n			2
	p			3
	q			4
	w			5

3.1 RTL Synthesis

After synthesizing the RTL code of each module in QuartusII, the output waveforms were observed and monitored. Every design input and output was monitored at timing waveform, Figure 3. Meanwhile, Table 3 shows the truth table for top-level design, i.e. *NaturalFrequencyModule*. All calculations based on algorithms of Equation (3) to (7), with inputs labeled as *a*, *b*, *c*, *d* and *e* while the outputs are *w*, *m*, *n*, *p*, *q*, respectively. The port-connection between submodules and top-level module are shown previously in Figure 1(c), and synthesized into RTL code as shown in Figure 2.

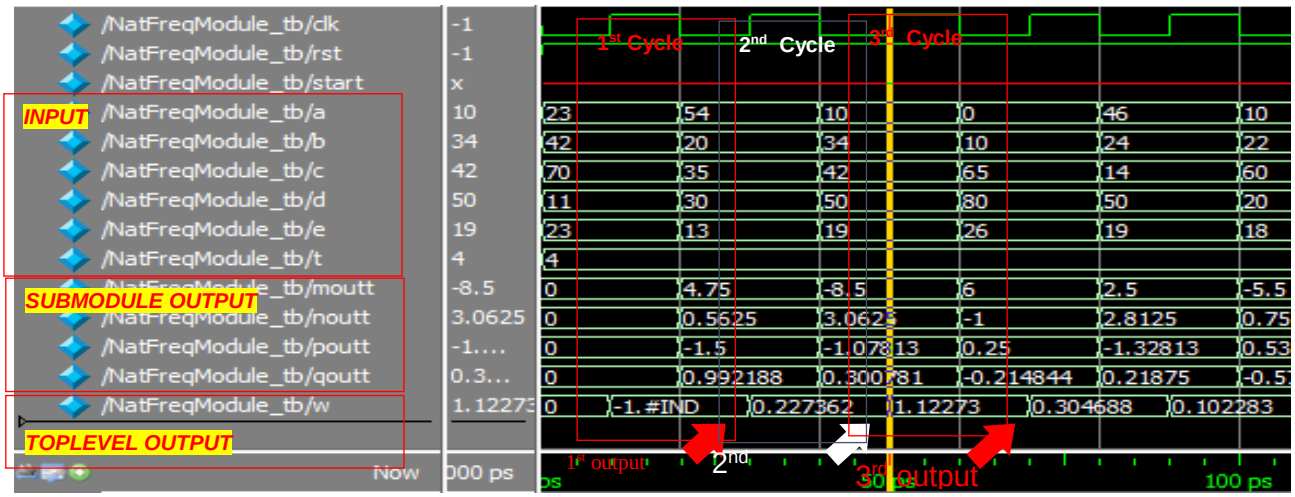
Table 3 The truth table for *NaturalFrequencyModule*

Input Seq-uence	Input					Submodule Output					Top level Output
	a	***b***	***c***	***d***	***e***	***m***	***n***	***p***	***q***	***w***	***w***
1	23	42	70	11	23	4.75	0.56	-1.50	0.99	0.22	0.22
2	54	20	35	30	13	-8.50	3.06	-1.07	0.30	1.12	1.12
3	10	34	42	50	19	6.00	-1.00	0.25	-0.21	0.30	0.30

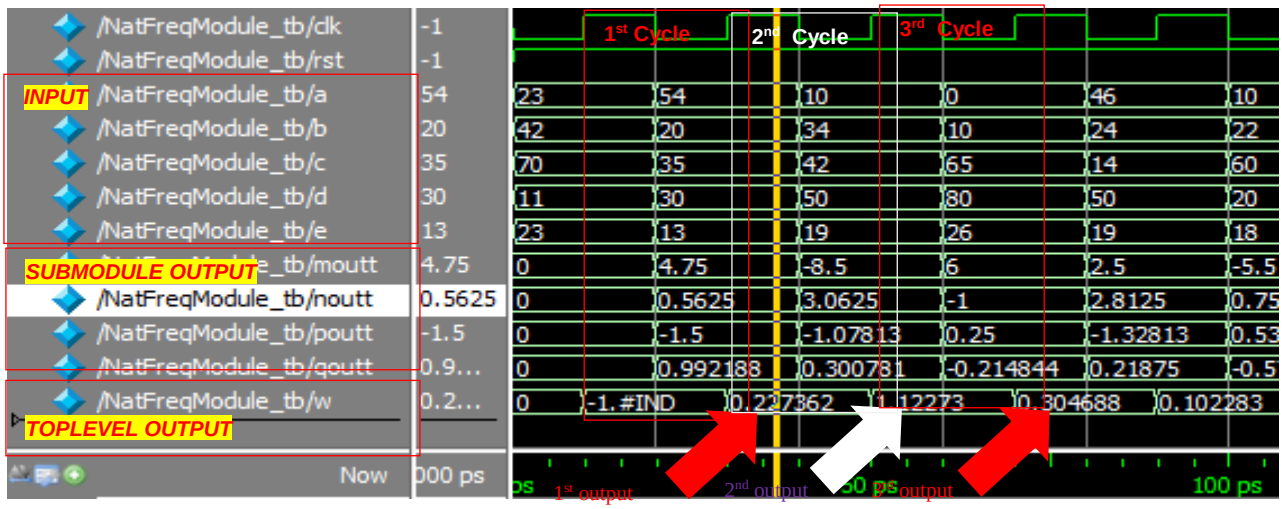
```
module NatFreqModule(
input rst, clk, start,
input var real a, b, c, d, e,
output real w, moutt, noutt, poutt, qoutt, woutt, //aoutt, boutt
output logic done
);

real m, n, p, q, w2, mout, nout, pout, qout ;

wmodule u5 (.m(m), .n(n), .p(p), .q(q), .w2(w),
          .clk(clk), .rst(rst) , .start(start), .done(done) ); //
qmodule u4 (.a(a), .b(b), .c(c), .d(d), .e(e),
          .q(q), .clk(clk), .rst(rst), .start(start) );
pmodule u3 (.a(a), .b(b), .c(c), .d(d), .p(p),
          .clk(clk), .rst(rst) , .start(start) );
nmodule u2 (.a(a), .b(b), .c(c), .n(n), .clk(clk),
          .rst(rst), .start(start) );
mmodule u1 (.a(a), .b(b), .m(m), .clk(clk),
          .rst(rst), .start(start)); // , .done(done));

//to observe m n p q value at every clock cycle...
always_ff@(negedge clk)
begin
    qoutt = q;
    poutt = p;
    noutt = n;
    moutt = m;
end
```

Figure 2 RTL code for top-level design of *NaturalFrequencyModule*

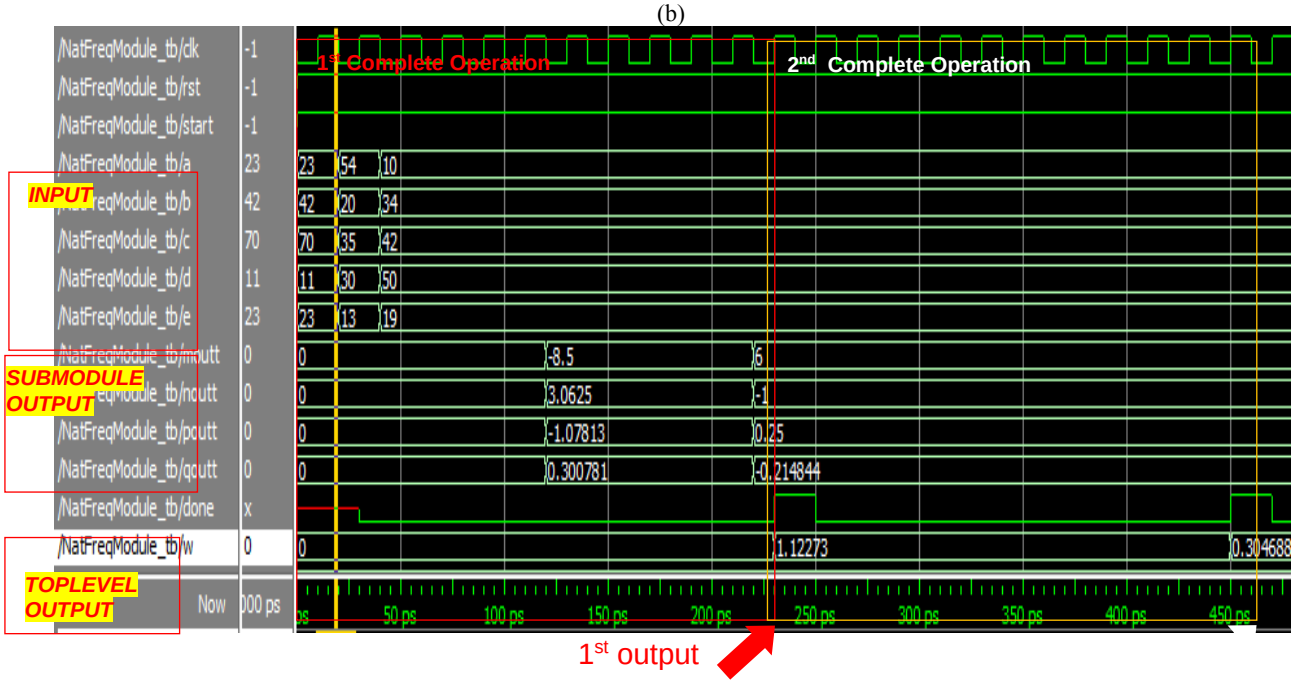
From Figure 1(a) and Figure 1(b), the input and output are the same for every clock cycle in both Design 1 and 2, respectively. This happens as the only value that change in both designs is *t*(s), although both designs are single-cycle design. When given the first group of inputs ($a = 23$, $b = 42$, $c = 70$, $d = 11$, and $e = 23$) given at time 0, the output of the top-level design can be observed after 1 clock-cycle for Design 1 and 2.



(a)



(b)



(c)

Figure 3 Waveform of top-level module, *NaturalFrequencyModule* of (a) Design 1, (b) Design 2, and (c) Design 3

The output, *w*, of *NaturalFrequencyModule* is 0.227362 after first clock cycle, and the output for each submodule was labeled as *moutt*, *noutt*, *pout* and *qoutt* for *m*-, *n*-, *p*-, *q*-modules in Figure 3. The values obtained from timing waveform of Figure 3(a) and Figure 3(b) for *m*-, *n*-, *p*-, *q*-modules are 4.75, 0.5625, -1.5, and 0.992188,
respectively.

Submodules *m*, *n*, *p*, and *q* design used positive-edge clock-trigger while w-submodule used negative-edge clock-trigger. The *w*-module was set to have different clock cycles to show that differently triggered clock cycle can delay the output by half of a clock cycle. In consideration of Design 1 and Design 2 comprising single-cycle design, all given inputs need 1 cycle to produce the output. Therefore, the second batch of inputs produce output after second clock cycle, which was 1.12273 as shown in Figure 3(a) and Figure 3(b). According to truth table in Table 3, the output of top-level module for second inputs ($a = 54$, $b = 20$, $c = 35$, $d = 30$, and $e = 13$) is 1.12. Next, when given third inputs to the *NaturalFrequencyModule* ($a = 10$, $b = 34$, $c = 42$, $d = 50$, and $e = 19$), the output obtained was 0.304688. All values for submodules and top-module output obtained from Figure 3 are same as truth table of Table 3. Therefore, both Design 1 and Design 2 were verified as both truth table and output waveform produced the same result.

Inputs for algorithm of Equation (3) were listed as *m*, *n*, *p*, *q* and the output was *w*, of which in Figure 3(c), were labeled as *moutt*, *noutt*, *poutt*, *qoutt*, and *woutt*, respectively. In Design 3, five clock-cycle needed to complete the concurrent modules, which are *m*-, *n*-, *p*-, and *q*-module. These modules, *m*-, *n*-, *p*-, *q*-module, must generate output and pass it to *w*-module. Then five more clock cycles were needed to complete the process for algorithm of Equation (3) (refer to Table 2). Since the inputs to the top-level module was given every 20 *p*s, and control input (labeled as *done* signal) is low or '0', w-module will be invoked. Refer to Figure 3(c), the first inputs ($a = 23$, $b = 42$, $c = 70$, $d = 11$, and $e = 23$) are neglected, while the second inputs ($a = 54$, $b = 20$, $c = 35$, $d = 30$, and $e = 13$) were fed into registers and produced submodule outputs, *moutt* = *-8.5*, *noutt* = *3.0625*, *pout* = *-1.07813*, *qoutt* = *0.300781*. The top-level (*NaturalFrequencyModule*) output is 1.12273 after 5 clock-cycle of positive-edge triggered output of *m*-, *n*-, *p*-, *q*-module. At this time, signal done was high, or '1'. Furthermore, when given the third inputs ($a = 10$, $b = 34$, $c = 42$, $d = 50$, and $e = 19$) to the *NaturalFrequencyModule*, the submodules output obtained as such, *moutt* = *6*, *noutt* = *-1*, *pout* = *0.25*, *qoutt* = *-0.214844*. The top-level module produced output of *w* = *0.304688* when *done* signal is high or '1'.

It is shown from truth table of Table 3, that the waveform output of Figure 3 are the same if every number is rounded to 2 decimal points. The outputs were valid and verified. Therefore, RTL code for Design 1, Design 2, and Design 3 were verified.

Referring to Figure 3, the performance of each design are summarized in Table 4. During the design, *QuartusII* default clock speed was used, thus, producing the same f_{max} at 50 GHz. Algorithm of natural frequency, Equation (3), for Design 1, Design 2, and Design 3 needed 20 ps, 20 ps, and 220 ps, respectively, for a complete operation. According to the DFG, Design 2 had less of 5 mul/div than Design 1 but both designs consist of 13 add/sub, while Design 3 only needs 1 mul/div and 1 add/sub. The result from *QuartusII* compilation shows total logic utilization for Design 1, Design 2, and Design 3 are 2530, 36, and 1, respectively. Total DSP blocks are 12 blocks for Design 1, while both Design 2 and Design 3 have no DSP blocks. Therefore, Design 3 needed the least resources among the proposed designs even though execution time taken for a complete operation was longest. More study had to be taken for Design 3 to optimize the performance because current study only emphasised the mapping of algorithm into RTL design using high-level synthesis approach, in order to find solution of multi-cycle inputs complexity and resource scheduling [22].

4.0 CONCLUSION AND FUTURE WORKS

It has been shown that algorithm of natural frequency, ω of ECG signal obtained from second-order system can be successfully implemented in hardware design based on high-level (HL) synthesis approach. Several architectures, i.e. Single-Cycle and Multi-Cycle fully-constraint were compared and it has been found that even though Design 3 comprised the least utilization of logic but needed longest time for one complete execution. Design 2 ranked second in logic utilization and provided less time than Design 3 and same execution time as Design 1. Therefore, Design 2 is chosen as the better design among the proposed designs during this study.

Optimization of resource utilization, such as usage of shift registers to do multiplication and division are intended to be further explored. Furthermore, the inputs variable in this study will be prolonged as single input which change over time, to realize the real-time ECG signal processing for detecting AF in human. Furthermore, multi-cycle operations using pipelined functions can enhance the performance of system design based on high-level synthesis approach.

Table 4 The performance analysis using Cyclone V 5CEFA731I7

Criteria	Design 1 : Single-Cycle	Design 2 : Single-Cycle (t = 4)	Design 3 : Multi-Cycle fully constraint
Cycle Count	1	1	11
Execution Time (for 1 complete operation)	1 cycle or 20 ps	1 cycle or 20 ps	11 cycles or 220 ps
Clock Speed (f_{max}, MHz)	1 / 20 p = 50 GHz	1 / 20 ps = 50 GHz	1 / (220 ps / 11cycles) = 50 GHz
Resource Utilization (according to DFG)	13 add/sub and 21 mul/div	13 add/sub and 16 mul/div	1 add/sub and 1 mul/div
Logic Utilization (QuartusII Summary)	2530	36	1
DSP Blocks	12	0	0

Acknowledgement

The authors would like to express their appreciation to Universiti Teknologi Malaysia and MOHE for supporting and funding this study under Zamalah UTM Scholarship for N. A. Abdul-Kadir and ERGS Grant No. R.J130000.7823.4L062. The authors would like to express gratitude to Nor Amin Rohimi (CITD) for his guidance in HDL and QuartusII. The authors are also indebted to Prof. Mohamed Khalil Hani and participants of MEL1183/01-20142015.

References

[1] M. R. Chassrin and R. W. Galvin. 1998. The Urgent Need to Improve Health Care Quality Institute of Medicine National Roundtable on Health Care Quality. *The Journal of the American Medical Association*. 280(11): 1000–1005.

[2] J. M. Grimshaw and A. Hutchinson. 1995. Clinical Practice Guidelines—Do They Enhance Value for Money in Health Care. *Oxford Journal - British Medical Bulletin*. 51(4): 927–940.

[3] D. W. Shapiro, R. Lasker, A. Bindman, P. R. Lee. 1993. Containing Costs While Improving Quality of Care: The Role of Profiling and Practice Guidelines. *Annual Reviews Public Health*. 14: 219–241.

[4] S. R. Chowdhury. 2012. Field Programmable Gate Array Based Fuzzy Neural Signal Processing System for Differential Diagnosis of QRS Complex Tachycardia and Tachyarrhythmia in Noisy ECG Signals. *Journal of Medical Systems*. 36(2): 765–75.

[5] Y. Sun and A. C. Cheng. 2012. Machine Learning on-a-Chip: a High-Performance Low-Power Reusable Neuron Architecture for Artificial Neural Networks in ECG Classifications. *Computers in Biology and Medicine*. 42(7): 751–7.

[6] C-H. Lin and G. W. Lin. 2012. FPGA Implementation of Fractal Patterns Classifier for Multiple Cardiac Arrhythmias Detection. *Journal of Biomedical Science and Engineering*. 05(03): 120–132.

[7] H. K. Chatterjee, R. Gupta and M. Mitra. 2012. Real time P and T Wave Detection from ECG using FPGA. *Procedia Technology*. 4: 840–844.

[8] M. Cvikl and A. Zemva. 2010. FPGA-oriented HW/SW Implementation of ECG Beat Detection and Classification Algorithm. *Digital Signal Processing*. 20(1): 238–248.

[9] W-C. Fang, H-C. Huang and S-Y. Tseng. 2013. Design of Heart Rate Variability Processor For Portable 3-Lead ECG Monitoring System-On-Chip. *Expert Systems with Applications*. 40(5): 1491–1504.

[10] D. M. Ballesteros, D. Marcela and M. Andrés. 2012. FPGA Compression Of ECG Signals by using Modified Convolution Scheme of the Discrete Wavelet Transform. *Ingeniare Revista chilena de ingenieria*. 20(1): 8–16.

[11] P. A. Wolf, R. B. D'Agostino, A. J. Belanger, W. B. Kannel. 1991. Probability of Stroke: A Risk Profile from the Framingham Study. *Stroke*. 22(3): 312–318.

[12] B. M. Everett, N. R. Cook, D. Conen, D. I. Chasman, P. M. Ridker, C. M. Albert, 2013. Novel Genetic Markers Improve Measures of Atrial Fibrillation Risk Prediction. *European Heart Journal*. 34(29): 2243–2251.

[13] S. A. Lubitz and D. Husser. 2013. Genomic Risk Scores in Atrial Fibrillation: Predicting the Unpredictable? *European Heart Journal*. 34(29): 2227–2229.

[14] S. A., Lubitz, C. Moser, L. Sullivan, M. Rienstra, J. D. Fontes, M. L. Villalon, M. Pai, D. D. McManus, R. B. Schnabel, J. W. Magnani, X. Yin, D., Levy, M. J. Pencina, M. G. Larson, P. T. Ellinor and E. J. Benjamin. 2013. Atrial Fibrillation Patterns and Risks of Subsequent Stroke, Heart Failure, or Death in the Community. *Journal of the American Heart Association*. 2:e000126.

[15] N. A. Abdul-Kadir, N. M. Safri and M. A. Othman. 2014. Classification of Paroxysmal Atrial Fibrillation using Second Order System. *Jurnal Teknologi (Science & Engineering)*. 67(3): 57–64.

[16] N. A. Abdul-Kadir, N. Mat Safri and M. A. Othman. 2014. Effect of ECG Episodes on Parameters Extraction for Paroxysmal Atrial Fibrillation Classification. IEEE Conference on Biomedical Engineering and Sciences. 874–877.

[17] D. D. Gajski and L. Ramachandran. 1994. Introduction to High-Level Synthesis. *IEEE Des. Test Comput*. 11(4): 44–54.

[18] M. K. Hani. 2014. *Advanced RTL Design for FPGA with SystemVerilog*. 2nd ed. Desktop Publisher.

[19] D. Corvino, I. Epicoco, F. Ferrandi, F. Fummi, D. Sciuto, P. Milano and P. Leonardo. 1998. Automatic VHDL Restructuring for RTL Synthesis Optimization and Testability Improvement. Proceedings International Conference on Computer Design : VLSI in Computers and Processors. 436–441.

[20] C. W., Rose and M. Albarran. 1975. Modeling and Design Description of Hierarchical Hardware/Software Systems. Proceedings of the 12th Design Automation Conference. Massachusetts, USA. 421–430.

[21] N. A. Abdul-Kadir , N. Mat Safri and M. A. Othman. 2015. DSP ASIC Module Design for Natural Frequency of ECG Signal. Submitted to IEEE International Circuits and Systems Symposium.

[22] C-T. Hwang, J-H. Lee and Y-C. Hsu. 1991. A Formal Approach to the Scheduling Problem in High Level Synthesis. *IEEE Tranaction on. Computational Design of Integrated Circuits System.* 10(4): 464–475.