FPGA implementation of wavelet coherence for EEG and ERP signals

Yahya T. Qassim^{a,*}, Tim R.H. Cutmore^b, David D. Rowlands^a

^a School of Electronic Engineering, Griffith University, 170 Kessels Road, Nathan, Brisbane, QLD 4111, Australia

^b School of Applied Psychology, Griffith University, Mt Gravatt Campus, Kessels Road, Brisbane, QLD 4111, Australia

ARTICLE INFO

Article history:

Received 26 November 2015

Received in revised form 31 July 2016

Accepted 3 January 2017

Available online xxx

Keywords:

Event related potential (ERP)

Electroencephalogram (EEG)

Field programmable gate array (FPGA)

Image quality

Wavelet coherence (WC)

ABSTRACT

This paper presents the design and implementation of wavelet coherence (WC) processor on low cost field-programmable gate array (FPGA). This design is adapted to estimate the wavelet coherence between two EEG signals in a minimal delay in order to support real time applications. The produced CWT coefficients were saved in static RAM chips and prepared for the WC analysis starting with the smoothing operation as an essential computation for the WC algorithm. The WC algorithm was analyzed in the means of choosing the suitable word length for the stages of the design and to simplify the employed functions in the algorithm. Several controllers that handle signal transmission among the design components were designed using hardware description language (VHDL). By using 4 parallel-processing smoothing circuits, the design is capable to calculate the coherogram between two EEG signals (1024 point each) in a total time of 128.64 ms. Image quality methods were applied for coherogram comparison between hardware and software. Hardware results were compared against the rigorous software standard WC according to the following measures; normalized mean square error (NMSE), normalized average difference (NAD) and structural content (SC) are 0.0045, 0.0485 and 0.921 respectively.

© 2016 Published by Elsevier Ltd.

1. Introduction

Wavelet coherence (WC) has been widely used for the study of connectivity between pairs of brain channels that represent the correlation between two electroencephalogram (EEG) waveforms at localized time and frequency [1,2]. The EEGs are very weak electrical activities produced by the brain and can be measured from different sites on the scalp using metal electrodes. The maximum amplitude for the EEG signal can go up to 100 μ V and their frequency components are in the range 0.5–100 Hz [3]. However, the typical frequency range of an EEG epoch is 0.5–40 Hz [4]. Real time EEG applications are highly depend on the event related potential (ERP) components superimposed on the ongoing EEG and specifically, the P300 [5–6]. The P300 is a positive peak 300–900 ms over the EEG and can occur following a stimulus presented to the participant [7–9].

An EEG epoch of 1000 ms is sufficient to occupy the ERP P300 that plays a vital role in recent EEG applications. Examples of such applications are the brain computer interface (BCI) [10] and biofeedback studies [11]. These applications require real time algorithm processing of the data measured to provide the required information within a minimal delay [12].

WC has been used to reveal brain abnormalities and malfunctions such as the diagnosis of schizophrenia [13] or the corticomuscular functionality in detecting muscle fatigue [14]. In addition, WC may be used for biofeedback studies where typically existing biofeedback measures are normally use single electrode sites [15]. However, using two electrode sites allow a measure of the linkage (such as coher-

ence) which taps into internal communications or possibly information processing between cortical areas, reflecting more complex cognitive activity, especially attentional filtering between these sites. A single electrode site does not have access to this and it was desired to target this functionality for biofeedback change.

Previous works in the literature showed that the computation of WC depends on the off-line processing of EEG signals due to the intensive operations included in the algorithm [1,2,13]. However, real time processing of WC for EEG signals is still in demand [14]. The involvement of WC in real time EEG applications require high speed of computation due to the complexity of the mathematical functions implied in the WC algorithm required to construct the WC plane, the “coherogram” [16]. To achieve such computation speed, the Field Programmable Gate Array (FPGA) solution is utilized as proposed in this paper.

Previous work by the authors has outlined the validity of WC in the ERP gamma band for finding significant relations related to cognition [17]. The involvement of the continuous wavelet transform (CWT) in analysing the EEG using FPGA and the effect of the quantization error on the produced CWT scalogram in this design with design optimizations were previously investigated by the authors [18–20], in which the grey blocks in Fig. 1 refer. In [20] the Morlet CWT for EEG signals was implemented in the frequency domain using Spartan 3AN FPGA. The calculation of the wavelet coefficients at 37 scales was then optimized to 21 scales that are sufficient to cover the frequency range of the EEG (0.5–60 Hz). The produced CWT coefficients were stored in two SRAM chips on the FPGA platform. This paper proposes a VHDL and schematic based FPGA design implementation of the WC applied to EEG and ERP signals based on the CWT design in [19,20]. The focus is to speed up the processing of the WC algorithm thereby meeting real time require-

* Corresponding author.

Email address: yahyataherqassim@yahoo.com, y.qassim@griffith.edu.au (Y.T. Qassim)

Fig. 1. The CWT design and the WC design in the FPGA with the SRAM chips used for storing results. Gray cells refer to previously developed and used blocks [20].

ments. The design presented in this paper is able to perform the task of WC in few milliseconds which is sensible for real time EEG applications. There is a little in the literature on the reconfigurable computing of the CWT [21–23]. However, to the best of the authors' knowledge, no previous work is found in the literature targeting VLSI mapping of WC.

The WC plane requires two wavelet planes to be constructed in addition to other involved computations including complex multiplications, smoothing, division and the Cartesian to polar conversion [1,16]. To calculate the square wavelet coherence (WC) R_n^2 between two equal length time series x and y , the following formula can be applied [16]:

$$R_n^2(s) = \frac{|S(s^{-1}W_n^{xy}(s))|^2}{S(s^{-1}|W_n^x(s)|^2) \cdot (s^{-1}|W_n^y(s)|^2)} \quad (1)$$

where s is the wavelet scale, s^{-1} is for energy density, W_n^{xy} is the wavelet cross spectrum between x and y and W_n^x is the CWT for signal x . S is the smoothing operator in scale or time such that $S(W) = S_{scale}(W_n(s))$ or $S(W) = S_{time}(W_n(s))$.

According to the Schwarz inequality, the WC takes its values between 0 and 1. Without smoothing all the WC coefficients are trivially equal to 1. As a priority of operations, it can be noticed from (1), the numerator, that the smoothing operation S is applied before determining the absolute value and the square operation to the WCS whereas smoothing follows up the absolute value and the square operation to both auto-spectrums in the denominator. Only the magnitude WC is used in this article as it is the commonly used in analysis [2,13], and no citation to the wavelet phase coherence is included.

To apply (1) on a pair of EEG waveforms and to prepare for a reference WC diagram, two EEG signals were used from the frontal and central brain electrodes (Fz & Cz) as shown in Fig. 2. In this figure, the presented stimulus to the participant was at 200 ms and the P300 response can be seen at 600–800 ms. By applying (1) to compute the WC between the EEGs in Fig. 2, the resultant software-based (Matlab) WC is shown in Fig. 3a in graded colours ranging between 0 and

Fig. 2. Pair of simultaneous EEG epochs measured through the oddball test from the brain sites Fz-Cz and used to test the FPGA based WC design.

Fig. 3a. Software based wavelet coherence between the EEGs in Fig. 2. used for validation.

1; close to 1 means high correlation regions between the two EEGs whereas close to 0 refers to weak coherence between the two EEG epochs in the time-scale plane. Fig. 3a represents the reference WC diagram to the target figure generated by FPGA. This is because the FPGA based WC subject to the quantization error throughout the included mathematical functions in (1) that negatively affect the quality of the coherogram. This quality can be tested with image quality methods as objective indices to the hardware WC. Fig. 3b shows the resultant WC by taking the FPGA based two CWTs (point A in Fig. 1) and applies Eq. (1) to them using Matlab. The WC in Fig. 3b shows the degradation appears in the coherogram against the one in Fig. 3a as a result of using FPGA based CWTs. More degradation will be obviously added when (1) is completely performed in FPGA as this paper presents.

The structure of this paper is as follows: Section 2 introduces the EEG data acquired to test the design along with the analysis of WC algorithm. In addition, the smoothing function and the word length selection for the WC design are also included. In Section 3, the FPGA implementation with parallel data transfer and memory usage for the proposed architecture of WC is introduced. Section 4 presents the obtained results with comparison between the hardware and software WC planes using three different image quality methods, in addition to the analysis of performance and the synthesis report. Finally, Section 5 concludes this paper.

Fig. 3b. Wavelet coherence between the EEGs in Fig. 2 produced in Matlab by using their FPGA based CWT coefficients located at point A-Fig. 1.

2. Methods

2.1. EEG data acquisition

EEG waveforms were collected from participants according to the visual oddball test by presenting one of the letters *G*, *T*, *C* and *A* as frequent stimuli on 85% of trials and the letter *X* randomly appeared as the oddball on 15% of trials. The recorded EEG epoch was 1400 ms (only the first 1024 ms were used for analysis as continuing to our previous works [18–20]). The sampling frequency was 1000 Hz for the acquired EEG and a 32-electrode cap was used according to the 10/20 international system [24]. The EEG was collected from the brain channels frontal-Fz and central-Cz [17] and applied to test the FPGA based WC design. The Griffith University ethics number for human research was (PSY/92/09/HREC).

2.2. Details of wavelet coherence algorithm

To resolve the WC algorithm and make it applicable by hardware, a flow diagram is necessary to clearly refer to the mathematical functions included and to investigate for any possible simplification. The flow diagram in Fig. 4 shows the detailed operations in (1) with shaded blocks to discern some of these operations. It can be noticed from Fig. 4 that the coordinate rotation digital computer (CORDIC) technique [25] is required to compute the magnitude for the coefficients of each of CWT1, CWT2 and the WCS. However, the dashed blocks in Fig. 4 refer to operations that cancel each other as complementary functions (the square root and the square). This simplifies the existent complexity and eliminates the need for the CORDIC technique.

To calculate the WCS [16], two complex multipliers were involved. The real and imaginary components of the WCS were calculated in the fixed-point 2's complement number representation as well as the auto-spectrum for each of CWT1 and CWT2. Other operations that require investigation targeting suitable choices to be adapted in the hardware development are the smoothing operation and the selection of the appropriate word length.

Fig. 4. The flow diagram for the wavelet coherence algorithm.

2.3. Efficient smoothing

Owing to their existent high variability, the WCS and auto-spectrums require smoothing (averaging) to transfer their coefficients to more reliable form [26] and to give a valid measure for the WC. Either temporal smoothing or frequency smoothing is used to average the coefficients of the wavelet spectrums [26]. Frequency smoothing is the one applied in this work (explained later in this section) where better results were achieved by employing this method. An example on the impact of frequency smoothing can be seen in Fig. 5(a,b). Fig. 5(a) shows a wavelet auto-spectrum before smoothing and Fig. 5(b) presents the same spectrum after smoothing. It can be noticed that smoothed coefficients in Fig. 5(b) have less step-variation compared to the coefficients in Fig. 5(a). This is concluded from the increase in the number of graded colours that reflect these coefficients.

Frequency smoothing was adapted from [26] and modified to be suitable for hardware processing. Frequency smoothing is given by taking the average of coefficients along the scale axis. Suppose w is the coefficients of a wavelet spectrum matrix of time axis t and scale axis s and W is the smoothed coefficient, then frequency smoothing

Fig. 5. The effect of frequency smoothing on a wavelet auto-spectrum (a) before smoothing. (b) after smoothing.

can be applied at each time and scale as:

$$W_{t,s} = \frac{w(t,s)}{s} + \frac{w(t,s+1)}{s+1} + \frac{w(t,s+2)}{s+2} \quad (2)$$

As smoothing is included in Fig. 4, Eq. (2) was used to smooth the wavelet auto and cross-spectrums required to produce the WC in Fig. 3a and b as well. As it shows, (2) contains addition and division operations. Addition can be easily applied in hardware; however, the implementation of division is not simple. Division is considered an expensive operation in the terms of area and time especially when a remainder exists. Therefore, frequency smoothing was empirically modified to be suitable for FPGA. The modification was applied to the three division operations in the terms of (2), where the divisor numbers (denominators) were chosen as multiples of two. This can be achieved in hardware with the arithmetic shift right.

2.4. Word length

The selection of the appropriate word length for WC computation in FPGA was necessary to balance between sufficient accuracy and the occupied silicon area [27]. As the inputs to the WC algorithm are CWT1 and CWT2 produced by hardware and passed through several processes (Fig. 4), further quantization error was added when performing the WC algorithm that follow the CWT. A suitable word length for each stage was chosen in order to extract the WC with maximum accuracy.

The integer 2's complement number representation was utilized in the WC design. Fig. 1 illustrates the WC design developed in this paper which processes the wavelet coefficients of 2 EEG epochs stored in the SRAM chips as in [20] for 1 EEG, then it was extended to analyze 2 EEG epochs. As both of CWT1 and CWT2 coefficients were stored in the SRAM chips with 16 bit sample, doubling this word length can be applied to obtain high accuracy for the WC. A truncation was applied to the product of the wavelet auto-spectrums and in calculating the absolute value for the WCS to retain the 16 bit sample. The following word lengths were used throughout the stages of the design: 16 bit sample for each of the real and imaginary components of CWT1 and CWT2, 32 bit sample for the components of the WCS (doubled by multiplication), 32 bit sample for auto-spectrum1 and auto-spectrum2 (also doubled by multiplication). Therefore, 32 bits were used by the smoothing operation. The word length for the WCS-absolute value was extended to 64 bits (by square), then truncated to 32 bits and the same method was performed for the product

of auto-spectrum1 by auto-spectrum2. Lastly, 32 bits for the WC, although only the least 4 bits (out of 32 bits) considered the most important since the WC coefficients were represented by 16 colors. These word lengths for the sub-operations in the WC equation were applied to the two CWTs produced by FPGA (taken from point A in Fig. 1) and examined in Matlab. The result is the coherogram shown in Fig. 6. This figure shows a semi-hardware WC plane since again the CWTs were FPGA based and the WC was software based (Matlab- 32 bit fixed point). The purpose of Fig. 6 is to estimate the resultant WC by FPGA when using the above word lengths for WC computation. Fig. 6 shows similarity against the one in Fig. 3a-software based. The wider dynamic range used for the WC helped in reducing the probability of over/underflow occurrence and reduce the quantization error. The subjective view of WC for various examined word lengths and graded colors was sufficient to choose the one which based on 32 bit sample with 16 graded colors.

3. FPGA implementation

The available target technology for the WC design is the Altium NanoBoard provided with Spartan 3AN (XC3S1400AN) FPGA [28]. The applied architecture to design the WC is parallel in finding the required components and pipelined in total. For example, to calculate the real and imaginary components of the WCS, parallel circuits were used. However, fetching wavelet coefficients then starting computa-

Fig. 6. Wavelet coherence between the EEGs in Fig. 2 produced by using their FPGA based CWT coefficients located at point A-Fig. 1, using Matlab 32 bit point number representation with 16 graded colors.

tion while new coefficients are fetched at the same time is a pipelined process. To implement the WC on the FPGA, the flow diagram in Fig. 4 was followed starting from loading the real and imaginary components of CWT1 and CWT2. The WC design initiates fetching then processing these components from the SRAM chips ($2 \times$ IS61LV25616AL) to produce the coherogram. These chips represent the memory map for the FPGA based WC design. It was planned to use the locations of the SRAM chips to store the CWT coefficients as shown in Fig. 7 where the CWT coefficients at each scale and for each component require 1024 locations (400_{HEX}). For the WC, specified locations in the SRAM0 chip were allocated to store the lower 16 bits of the WC (see Table I). Only the least 4 bits are sent to the PC as they contain the useful data to represent the coherogram. The 16 higher bits of the WC were neglected.

It can be noticed from Table 1 that the allocated locations for the WC required only 19 K to represent the coherogram. This is because the wavelet coefficients at scale1 and scale21 were omitted to facilitate the smoothing operation by FPGA and the rest of coefficients were considered.

Fig. 7. The SRAM chips locations for CWT 1 & CWT2 coefficients.

Table 1
The occupied locations in the SRAM chips.

SRAM0 address (hexadecimal)	Data size in words	Data stored
0–53FF	21 K	CWT1-real component
5400–A7FF	21 K	CWT2-real component
A800–F3FF	19 K	WC-16 lower bits
SRAM1 address (hexadecimal)	Data size	Data stored
0–53FF	21 K	CWT1-imaginary component
5400–A7FF	21 K	CWT2-imaginary component

3.1. Design operation

VHDL code was developed for signal routing and operations required for calculating the WC in the FPGA. Initially, the coefficients of both CWT1 and CWT2 (each is $21 \text{ K} \times 16 \text{ bit}$) are loaded from the SRAM chips located on the Altium NanoBoard [29] and prepared for the WC computation (Fig. 1).

To smooth the values of the wavelet (auto and cross spectrum) at each time and scale, three terms were included for smoothing as shown in (2). However, as frequency smoothing was empirically modified; instead of using the divisors in (2), the divisors $d1$, $d2$ and $d3$ with five different patterns given in Table 2 were used for smoothing as formulated in (3):

$$W_{t,s} = \frac{w(t,s)}{d1} + \frac{w(t,s+1)}{d2} + \frac{w(t,s+2)}{d3} \quad (3)$$

In this context, the five proposed division sequences given in Table 2 ($d1$, $d2$ & $d3$) are sufficient for smooth testing and all are 2 and multiple of 2 numbers. These numbers are individually used to smooth the wavelet spectrums with Matlab. This is required to produce the WC based on the two CWTs determined in hardware (point A-Fig. 1) since all of the five sequences in Table 2 that represent the denominators in the terms of (3) are applicable by hardware. The best values for smoothing were found using three image quality methods. These methods were individually applied to test the five produced WC plots with the reference software one in Fig. 3a and to investigate which sequence in Table 2 gives the best smoothing. These methods are the normalized mean square error (NMSE), normalized average difference (NAD) and the structural content (SC) [30–32].

Table 3 shows the result of adapting each of the five divisor sequences in Table 2 on the produced hardware WC examined with the image quality methods. These methods give proper indication on the symmetry between software WC (Fig. 3a) against the semi-hardware WC. The approach of involving three methods overcomes the weakness in any single method alone. The optimum measures are NMSE = 0, NAD = 0 and SC = 1. It can be seen from Table 3 that the divisor sequence of $d1 = 2$, $d2 = 2$, $d3 = 2$ gives the best results since the two indices (NMSE & NAD) are close to zero and the SC is very similar for all methods. Therefore, the divisor sequence $d1 = 2$, $d2 =$

Table 2
The suggested divisor sequences for the smoothing operation in the WC algorithm.

Divisors	d1	d2	d3
Divisor sequence 1	2	2	2
Divisor sequence 2	2	4	4
Divisor sequence 3	2	8	8
Divisor sequence 4	2	4	8
Divisor sequence 5	2	2	4

Table 3
Divisor sequence results over full coherogram examined with three image quality methods against the software reference.

Divisor sequence d1, d2 & d3	NMSE	NAD	SC
2, 2, 2	0.0016	0.0218	0.9979
2, 4, 4	0.0019	0.0235	0.9981
2, 8, 8	0.0020	0.0240	0.9981
2, 4, 8	0.0020	0.0239	0.9976
2, 2, 4	0.0017	0.0225	0.9976
Software reference	0.0000	0.0000	1.0000

2, $d3 = 2$ was used in the smoothing operation to compute the WC in the FPGA.

Despite the high closeness between the results in Table 3 across the 5 sequences that goes up to 0.0004 for the NMSE, 0.0022 for the NAD and 0.0005 for the SC, the selection of the sequence which gives the closest results to the software reference is still significant. The reason belongs to the high number of smoothed points to configure one figure of WC, so improper single smoothing operation might lead to accumulated errors thereby loss in quality of the coherogram. As coming next, every single value of the WC requires 4 smoothing

operations to be produced. If we choose a divisor sequence of 2, 8, 8, the deviation from the software reference will be (4 times 0.0020 = 0.0080) for the NMSE whereas it is (0.0064) with the divisor sequence 2, 2, 2 on one WC value. This reflects an idea on the amount of loss in quality might occur to the final WC of size (19×1024) values.

The schematic diagram of the proposed smoothing circuit is shown in Fig. 8. An arithmetic shift right was performed to divide the circuit input by 2. Three values (32-bit each) were entered sequentially to the smoothing circuit in Fig. 8. Following the division process which was performed by right shift of one bit, these values were added together using a 32-bit adder and the final result is output-enabled when the enable pulse of the output register is given. The produced output represents the terms of (3) added together with the chosen dividers as $d1 = 2, d2 = 2, d3 = 2$. The circuit in Fig. 8 was repeated 4 times to smooth each of the 4 spectrums (2 auto-spectrums, the real and imaginary WCS) simultaneously.

To calculate one value for the WC at a time and scale, three values are required from each component of the four (CWT1 (*re* & *im*) and CWT2 (*re* & *im*)). Therefore, 12 values were fetched (6 from each SRAM) to produce one coherogram value. The coherogram was stored in the SRAM0 starting at location A800 for the 16 lower order bits of each WC coefficient.

Smoothing implementation in VHDL requires extensive access to the SRAM chips to fetch all the stored CWT coefficients. All the four components of both CWTs were available to calculate the WCS and the auto-spectrums. A shift of 1024 to each address occur every clock pulse going through 4×21 K addresses to smooth the wavelet auto and cross spectrum. By using 4 identical circuits, smoothing was performed 4 times simultaneously for each of the two components of the WCS and for the two auto-spectrums. Each smoothed coefficient at a time and scale requires three CWT coefficients to be produced as in (3), and this was applied for the four CWT components in Fig. 7. Therefore 12 values are needed to be fetched from the two SRAM chips to calculate one WC value. The address can be fed to both SRAMs at the same time and as a result, 6 clock cycles are required to fetch these 12 values. Fig. 9 shows how the switching through SRAM addresses is scheduled to achieve frequency smoothing. The same switching presented in Fig. 9 was followed in both SRAM0 and SRAM1 at the same time. It can be noticed that the first 6 fetched coefficients are at addresses 0000, 5400, 0400, 5800, 0800 and 5C00 which are entered into the WC processor. However, fetching CWT1-real from SRAM0 followed the address sequence 0000, 0400, 0800, 0001, 0401, 0801 etc. and fetching CWT2-real from SRAM0 fol-

Fig. 8. The proposed schematic for the 32-bit frequency smoothing circuit (one of four identical circuits). The output is controlled by the enable pulse.

S1		S1		S2		S2		S3		S3		S21		S21
0000	→	5400	→	0400	→	5800	→	0800	→	5C00	...	5000	→	A400
0001	→	5401	→	0401	→	5801	→	0801	→	5C01	...	5001	→	A401
0002	→	5402	→	0402	→	5802	→	0802	→	5C02	...	5002	→	A402
0003	→	5403	→	0403	→	5803	→	0803	→	5C03	...	5003	→	A403
.....	→		→	...	→		→		→	...	...		→	
03FF	→	57FF	→	07FF	→	5BFF	→	0BFF	→	5FFF	...	53FF	→	A7FF
CWT1		CWT2		CWT1		CWT2		CWT1		CWT2		CWT1		CWT2

Fig. 9. Switching between the locations of SRAM0 chip to access the required wavelet coefficients at a localized time & scale; light-grey cells stand for CWT1-real component and dark-grey cells represent CWT2-real components.

lowed the address sequence 5400, 5800, 5C00, 5401, 5801, 5C01 etc. The light-grey cells refer to the CWT1-real component and the dark-grey cells stand for the CWT2-real component. The interleaving in the fetching procedure was necessary to provide the CWT coefficients simultaneously to calculate the coefficients of WC.

As previously mentioned, only the least 4 bit of the WC were sent to the PC to display the WC in 16 different colours. Although the WC word length was initially represented by 32 bit, such wide number of bits was useful in providing wider dynamic range thereby reducing the probability of an over/underflow occurrence. When the number of colours increase in the WC plot to 32 or 64, some noisy features started to appear in the WC. The subjective inspection of WC figures at different number of colours was sufficient to choose the most suitable one [33]. The wide data path used to extract the WC cannot be reduced since the addition and multiplication performed to find the WCS and auto spectrums are VHDL operations on signed numbers where all the bit extension were involved in the operation including the sign bit.

The WC processor was designed in VHDL, imported to the Altium Designer schematic sheet and connected to other design components such as the SRAM chips, I/O units etc. The WC processor performs all the mentioned operations on CWT1 and CWT2. These include addressing the SRAM chips, calculating the cross-spectrum and auto-spectrums, applying the smoothing operation with the required switching between the addresses and also performs the division oper-

ation presented in Fig. 4 to produce the WC. The division operation was implemented as a concurrent statement in the VHDL code, where the WC output is updated with the respective values of WCS and auto-spectrums [33]. Fig. 10 shows a VHDL example code (only the architecture part) for the concurrent division operation where no clock signal is involved. For simplicity, instead of using inputs of 32 bit each, a code of 4 bit division is presented. Following calculating the WC values, these values are stored in SRAM0 as given in Table 1.

In Fig. 10, two 4-bit inputs represent the dividend (A) and the divisor (B), data_out is the result of division and DIVz is a flag equal logic 0 as long as the output is not zero and equal to logic 1 when the output is zero. Several internal signals are used to extend inputs and perform the division through steps of shift and compare. The sign of the result of division is produced in line 27 as an exclusive-or between the sign bits of both inputs where the code is allocated for dividing signed binary numbers. The result of division is updated whenever each or both of the inputs are changed. The WC resultant in dividing the WCS by the auto spectrum in an equivalent code, however, the extension was 32 bit for each.

4. Results

The FPGA-based WC can be seen in Fig. 11 where all the processing steps to produce this figure, were made by hardware starting

```

1
2  library IEEE;
3  use IEEE.std_logic_1164.all;
4  use IEEE.std_logic_unsigned.all;
5  entity fpdiv3 is
6      port (DIVz: out STD_LOGIC; A, B: in std_logic_vector (3 downto 0);
7            data_out: out std_logic_vector (3 downto 0));
8  end fpdiv3;
9
10 architecture fpdiv_arch3 of fpdiv3 is
11     signal REM0, REM1, REM2, REM3 : std_logic_vector (5 downto 0);
12     signal DIV0, DIV1, DIV2, DIV3 : std_logic_vector (5 downto 0);
13     signal Q_TEMP : std_logic_vector (3 downto 0);
14     signal Z0, Z1 : std_logic_vector (2 downto 0);
15     signal ZERO : std_logic;
16     begin
17         Z0 <= (others => '0');
18         Z1 <= (others => '0');
19         DIV0 <= Z0 & B(2 downto 0);
20         REM3 <= Z1 & A(2 downto 0);
21         DIV1 <= DIV0(4 downto 0) & '0';
22         DIV2 <= DIV1(4 downto 0) & '0';
23         DIV3 <= DIV2(4 downto 0) & '0';
24         Q_TEMP(0) <= '1' when (REM1 >= DIV0) else '0';
25         Q_TEMP(1) <= '1' when (REM2 >= DIV1) else '0';
26         Q_TEMP(2) <= '1' when (REM3 >= DIV2) else '0';
27         Q_TEMP(3) <= A(3) xor B(3);
28         REM2 <= REM3 - DIV2 when Q_TEMP(2) = '1' else REM3;
29         REM1 <= REM2 - DIV1 when Q_TEMP(1) = '1' else REM2;
30         REM0 <= REM1 - DIV0 when Q_TEMP(0) = '1' else REM1;
31         ZERO <= '1' when B(2 downto 0) = Z1 else '0';
32         DIVz <= '1' when ZERO = '1' else '0';
33         data_out <= (others => '0') when ZERO = '1' else Q_TEMP;
34     end fpdiv_arch3;
35

```

Fig. 10. An example of the concurrent division VHDL code. Inputs are 4 bit each.

Fig. 11. FPGA based wavelet coherence (the coherogram), point B-Fig. 1. Smoothed by using (3) with the 2, 2, 2 divisor sequence.

from the raw EEGs. The WC in this figure is represented with time axis of 1024 ms and a scale axis of 19 scales. The chosen scales correspond nearly to the typical EEG frequency range; scale 1 reflects the EEG frequency of 52.6 Hz and scale 19 corresponds to the frequency 2.3 Hz. The color bar of 16 graded colors was normalized to keep the WC in the range 0–1. The coherogram in Fig. 11 shows high similarity against the software one in Fig. 3a. The mentioned three image quality methods were used to objectively measure the quality between hardware coherogram and the software one (Fig. 11 versus Fig. 3a). For further verifications to the FPGA based WC, 10 randomly selected Fz-Cz EEG pairs were tested using the image quality methods. Table 4 shows the results of the validation where the NMSE and NAD are close to zero which indicates a high similarity. The SC is close to 1 which also indicates a high degree of similarity. The 3 measures for the 10 trials show that the hardware design gives little variation over trials. This can be concluded from the standard deviation (SD) which shows the spread of the individual measures to the mean whereas the mean gives the representative value across all trials. It can be seen that the mean is close to 0 for the NMSE and NAD hence it can be concluded that the hardware result is very close to the software one. This test verifies the operation of the WC design.

The required time to calculate the WC for 1024 ms and for 19 wavelet scales is measured as follows. To calculate, deliver and store in memory one coefficient of WC at a time and scale, 18 clock cycles were required: 7 clocks for addressing the SRAM chips (6 read (fetching CWT coefficients) & 1 write cycles (storing the WC coefficient)), 10 clocks for the pipelined functions within the WC processor and 1 clock cycle for sample separation. The FPGA NanoBoard is

Table 4

Three image quality methods to test 10 hardware wavelet coherence plots with their software counterparts (ideals for identical, NMSE = 0, NAD = 0, SC = 1).

Trial#	NMSE	NAD	SC
1	0.0045	0.0485	0.9210
2	0.0055	0.0560	0.9244
3	0.0061	0.0571	0.9228
4	0.0060	0.0586	0.9216
5	0.0058	0.0547	0.9308
6	0.0060	0.0570	0.9293
7	0.0051	0.0526	0.9234
8	0.0055	0.0555	0.9330
9	0.0050	0.0540	0.9207
10	0.0037	0.0488	0.9198
Mean	0.0053	0.0543	0.9247
SD	0.00076	0.0034	0.0047

provided with 200 MHz clock source, however, the peak operational frequency for the WC design was 40 MHz. Although the design critical path allows increasing the clock rate up to 57 MHz, the SRAM read/write speed and the serial port used to send results to the PC restrict the maximum clock rate. Therefore, the required total time to compute all the coefficients of the coherogram (Fig. 11), starting by fetching CWT1 and CWT2 from the SRAM chips for 19 K value with 18 clock/value at 40 MHz is:

$$T_{total} = \frac{1}{40 \text{ MHz}} \times 18 \frac{\text{clock}}{\text{coefficient}} \times 19 \text{ K (coefficient)}$$

$$= 8.7 \text{ ms}$$

The total time elapsed in calculating the dual CWT at 133 MHz was 1.14 ms (as it was 1 ms in [18] then by scale reduction and expanding the design to analyse 2 EEGs it was 1.14 ms). The exchange between the CWT design and the WC design can be applied using the multi booting technique to solve the problem of FPGA inadequate resources. This swap implementation requires 118.8 ms for reading the image file from the internal flash memory at 50 MHz. Therefore, the total time to estimate the WC from two-1024 ms EEG epochs was 8.7 ms (@40 MHz) + 1.14 ms (@133 MHz) + 118.8 ms (@50 MHz) = 128.64 ms. In our context, up to 1000 ms delay is acceptable for biofeedback, therefore, with the achieved run time, a WC based biofeedback is potential. The synthesis report for the WC design is shown in Table 5 and includes the occupied resources. A low usage of I/O pins of 17% was sufficient for the design. However, the FPGA multipliers, slices and LUTs have higher percentages of usage. The FPGA slices are the most used by this design as clear from the table.

A more powerful FPGA can be used in the future to perform both the CWT and WC implementations without the need for the multi-booting. In addition, with larger device, all data can be stored in the block RAM and no further need for the SRAM chips leading to achieve higher design performance. A future extension to this work is to adapt the developed WC architecture in a biofeedback study.

5. Conclusions

In this paper, WC processor between two EEG or ERP signals at high run speed was proposed based on reconfigurable computing (FPGA). The concentration of the paper was on the processing part of the WC algorithm in hardware and no existence to the analog interface as the EEG signals were previously stored in the FPGA block ram. By dismantling the WC algorithm into separate functions, the need for CORDIC technique was eliminated. A sufficient word length was used at each stage to obtain high accuracy in producing the coherogram. In addition, we presented an efficient frequency smoothing technique that was implemented on Spartan 3AN FPGA platform. Moreover, an efficient scheduling of memories to perform the smoothing operation which was managed in parallel by involving four identical circuits was outlined. The produced coherogram by FPGA was tested objectively by using three image quality methods that showed high closeness to the software based one. The achieved

Table 5

Synthesis report for the wavelet coherence FPGA design.

Device resources	Usage summary	% Usage
I/O pins	90	17%
Multipliers 18 × 18	20	62%
Slices	7585	67%
Total 4-input LUTs	14,032	62%

short computation time for the WC design indicated that the proposed implementation can be applied in a visual biofeedback.

Acknowledgment

This work is supported by the Ministry of Higher Education and Scientific Research of IRAQ.

References

- [1] J.-P. Lachaux, A. Lutz, D. Rudrauf, D. Cosmelli, M.L.V. Quyen, J. Martinerie, F. Varela, Estimating the time-course of coherence between single-trial brain signals: an introduction to wavelet coherence, *Neurophysiol. Clin.* 32 (2002) 157–174.
- [2] A. Klein, T. Sauer, A. Jedynak, W. Skrandies, Conventional and wavelet coherence applied to sensory-evoked electrical brain activity, *Biomed. Eng. IEEE Trans.* 53 (2006) 266–272.
- [3] M. Teplan, Fundamentals of EEG measurement, *Measure. Sci. Rev.* 2 (2002).
- [4] G. Benham, H. Rasey, J. Lubar, J. Frederick, A. Zoffuto, EEG power-spectral and coherence differences between attentional states during a complex auditory task, *J. Neurotherapy* 2 (1997) 1–9.
- [5] L.A. Farwell, E. Donchin, Talking off the top of your head: toward a mental prosthesis utilizing event-related brain potentials, *Electroen. Cephalogr. Clin. Neurophysiol.* 70 (6) (1988) 510–523.
- [6] J.L. Sirvent Blasco, E. Ianez, A. Ubeda, J.M. Azorin, Visual evoked potential-based brain-machine interface applications to assist disabled people, *Expert Syst. Appl.* 39 (9) (2012) 7908–7918.
- [7] E. Donchin, D.B. Smith, The contingent negative variation and the late positive wave of the average evoked potential, *Electroen. Clin. Neurophysiol.* 29 (1970) 201–203.
- [8] S. Sutton, M. Braren, J. Zubin, E.R. John, Evoked potential correlates of stimulus uncertainty, *Science* 150 (1965) 1187–1188.
- [9] A. Mouraux, G. Iannetti, Across-trial averaging of event-related EEG responses and beyond, *Magn. Resonance Imag.* 26 (2008) 1041–1054.
- [10] C.-C. Postelnicu, D. Talaba, P300-based brain-neuronal computer interaction for spelling applications, *Biomed. Eng. IEEE Trans.* 60 (2013) 534–543.
- [11] D. Varoqui, J. Froger, J.Y. Pélissier, B.G. Bardy, Effect of coordination biofeedback on (re) learning preferred postural patterns in post-stroke patients, *Motor Control* 15 (2011) 187–205.
- [12] J. Elshout, G.G. Molina, Review of Brain-Computer Interfaces based on the P300 evoked potential, *Lancet Neurol* 7 (2008) 1032–1043.
- [13] V. Sakkalis, T. Oikonomou, E. Pachou, I. Tollis, S. Micheloyannis, M. Zervakis, Time-significant wavelet coherence for the evaluation of schizophrenic brain activity using a graph theory approach, In: *Engineering in Medicine and Biology Society, 2006. EMBS'06. 28th Annual International Conference of the IEEE*, 2006, pp. 4265–4268.
- [14] Y. Qi, V. Siemionow, Y. Wanxiang, V. Sahgal, G.H. Yue, Single-trial EEG-EMG coherence analysis reveals muscle fatigue-related progressive alterations in corticomuscular coupling, *Neural Syst. Rehab. Eng. IEEE Trans.* 18 (2010) 97–106.
- [15] G. Pfurtscheller, C. Neuper, A. Guillot, C. Collet, EEG-based brain computer communication, *Neurophysiol. Found. Mental Motor Imagery* (2010) 203–212.
- [16] A. Grinsted, J.C. Moore, S. Jevrejeva, Application of the cross wavelet transform and wavelet coherence to geophysical time series, *Nonlinear Processes Geophys.* (2004) 561–566.
- [17] Y.T. Qassim, T.R. Cutmore, D.A. James, D.D. Rowlands, Wavelet coherence of EEG signals for a visual oddball task, *Comput. Biol. Med.* 43 (2013) 23–31.
- [18] Y.T. Qassim, T. Cutmore, D. Rowlands, Multiplier truncation in FPGA based CWT, In: *Communications and Information Technologies (ISCIT), 2012 International Symposium on*, 2012, pp. 947–951.
- [19] Y.T. Qassim, T. Cutmore, D. Rowlands, optimized FPGA based continuous wavelet transform, *Comput. Electr. Eng.* (2014). <http://dx.doi.org/10.1016/j.compeleceng.2014.11.012>.
- [20] Y.T. Qassim, T. Cutmore, D. James, D. Rowlands, FPGA implementation of Morlet continuous wavelet transform for EEG analysis, In: *Computer and Communication Engineering (ICCCCE), 2012 International Conference on*, 2012, pp. 59–64.
- [21] V. Bostanov, BCI competition 2003-data sets Ib and IIb: feature extraction from event-related brain potentials with the continuous wavelet transform and the t-value scalogram, *Biomed. Eng. IEEE Trans.* 51 (2004) 1057–1061.
- [22] S. Patil, E. Abel, Real time continuous wavelet transform implementation on a DSP processor, *J. Med. Eng. Technol.* 33 (2009) 223–231.
- [23] S. Kang, P. Xueezeng, P. Lingdi, A reconfigurable computing engine for wavelet transforms, In: *Parallel and Distributed Processing Symposium, 2007. IPDPS 2007. IEEE International*, 2007, pp. 1–5.
- [24] G.H. Klem, H. Lüders, H. Jasper, C. Elger, The ten-twenty electrode system of the international federation, in: G. Deuschal, A. Eisen (Eds.), *Recommendations for the Practice of Clinical Neurophysiology: Guidelines of the International Federation of Clinical Neurophysiology*, Elsevier, Amsterdam, 1999, pp. 3–6.
- [25] R. Andraka, A survey of CORDIC algorithms for FPGA based computers, In: *Proceedings of the 1998 ACM/SIGDA Sixth International Symposium on Field Programmable Gate Array*, 1998, pp. 191–200.
- [26] C. Torrence, G.P. Compo, A practical guide to wavelet analysis, *Bull. Am. Meteorol. Soc.* 79 (1998) 61–78.
- [27] J.E. Stine, O.M. Duverne, Variations on truncated multiplications, In: *Proceedings of the Euromicro Symposium on Digital System Design (DSD'03)*, 2003, pp. 112–119.
- [28] Xilinx, Inc, Spartan-3AN FPGA family data sheet, Accessed Feb. 6 http://www.xilinx.com/support/documentation/spartan-3an_data_sheets.htm, 2012.
- [29] Altium Limited, Nanoboard-3000, Accessed Feb. 6 <http://nb3000.altium.com/intro.html>, 2012.
- [30] A.M. Eskicioglu, P.S. Fisher, Image quality measures and their performance, *Commun. IEEE Trans.* 43 (1995) 2959–2965.
- [31] N. Bourbakis, Detecting similarities and differences in images using the PFF and LGG approaches, In: *Proceedings of the 14th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'02)*, 2002, pp. 355–362.
- [32] S. Poobal, G. Ravindran, The performance of fractal image compression on different imaging using objective quality measures, *Int. J. Eng. Sci.* 3 (2011) 525–530.
- [33] Y.T. Qassim, FPGA Design and Implementation of Wavelet Coherence for EEG Signals, PhD thesis, Accessed July 13, 2016.

Yahya T. Qassim received his BSc. Degree in electrical engineering in 1994 and the MSc. in computer engineering in 2004 both from University of Mosul in Iraq. He received his PhD degree from Griffith University, Australia in 2014. He is currently a casual teaching staff and researcher in the School of Electronic Engineering at Griffith University. His current research interests include biomedical signal processing, the design of high-speed digital circuit, reconfigurable computing, optimization in VHDL design and the FPGA implementation of DSP algorithms.

Tim R. H. Cutmore received his MSc in Computing and Information Science in 1989 and PhD in Psychology in 1987 both from Queens University in Canada. He presently is a Senior Lecturer in the School of Applied Psychology at Griffith University. Tim has published a number of studies on signal processing of EEG data. His current interest is in the application of pattern recognition algorithms for classifying event-related potentials (ERPs) in single trials, ERPs in relation to human cognition and signal processing methods for EEG and ERPs.

David D. Rowlands received his BSc. degree in Physics (1986) from Griffith University in Brisbane Australia and received his PhD

in Electronic Engineering (2001) from the same institution. David has researched in a number of different areas including semiconductor devices, computing hardware and signal analysis of biological signals. He is currently researching in signal processing of EEG and ECG, real time computing systems, Field Programmable Gate Array systems, and inertial sensors for human movement analysis.