

Automated and Reliable Low-Complexity SoC Design Methodology for EEG Artefacts Removal

Pranit N Jadhav

A Thesis Submitted to
Indian Institute of Technology Hyderabad
In Partial Fulfillment of the Requirements for
The Degree of Master of Technology



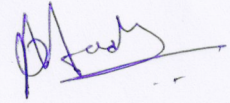
भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

Department of Electrical Engineering

June 2016

Declaration

I declare that this written submission represents my ideas in my own words, and where ideas or words of others have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be a cause for disciplinary action by the Institute and can also evoke penal action from the sources that have thus not been properly cited, or from whom proper permission has not been taken when needed.



(Signature)

Pranit N Jadhav

(EE13M1023)

Approval Sheet

This Thesis entitled Automated and Reliable Low-Complexity SoC Design Methodology for EEG Artefacts Removal by Pranit N Jadhav is approved for the degree of Master of Technology from IIT Hyderabad

Amitabhargava

(Dr. Amitabhargava) Adviser
Dept. Electrical Engineering

Ashudeb Dutta

(Dr. Ashudeb Dutta) Member
Dept. Electrical Engineering

Shishir Kumar

(Dr. Shishir Kumar) Member
Dept. Electrical Engineering

Sathya Prasi

(Dr. Sathya Prasi) External
Dept. Computer science &
Engineering.

Acknowledgements

Foremost, I would like to express my sincere gratitude to my advisor Dr. Amit Acharyya for the continuous support of my M.Tech study and research, for his patience, motivation, enthusiasm, and immense knowledge. His guidance helped me in all the time of research and my stay at Indian Institute of Technology Hyderabad. I could not have imagined having a better advisor and mentor for my M.Tech study.

Besides my advisor, I would like to thank other faculty members of Microelectronics and VLSI specialization: Dr. Asudeb Dutta, Dr. Shiv Govind Singh and Dr. Siva Rama Krishna Vanjari for their encouragement, insightful comments and help during course work and related projects.

Getting through my thesis required more than academic support, and I have many, many people to thank for listening to and, at times, having to tolerate me over the past three years. I cannot begin to express my gratitude and appreciation for their friendship. Deepali Nimbalkar, Aniket Ghole, Kunal Yadav, Pankaj Kumar Jha, Pravanjan Patra, Swati Bhardwaj, Sumit Naikwad, Radharamana Mohanty, Naresh Vemishetty, Rajkiran Choudhary, Parveen Nisha, Anuradha Balouria, Arvind Gautam, Madhuri Panwar, Lakhan and Shahnawaz Khan have been unwavering in their personal and professional support during the time I spent at the IIT Hyderabad. For many memorable evenings out and in, I must thank everyone above as well as my juniors: Prakash Lenka, Shashank Raghuraman and Harsha Prasad and all my labmates at Advanced Embedded System and IC Design Lab, IIT Hyderabad.

Last but not the least, I would like to thank my family: my parents Vasundhara and Namdeo Jadhav, for giving birth to me at the first place and supporting my decisions throughout life. With his own brand of humour, Sumit Jadhav, my elder brother, has been kind, supportive and never said no to me in any situation, since my birth. This thesis stands as a testament to your unconditional love and encouragement.

Dedication

To My Family, Relatives and Friends.....

Abstract

EEG is a non-invasive tool for neurodevelopmental disorder diagnosis (NDD) and treatment. However, EEG signal is mixed with other biological signals including Ocular and Muscular artefacts making it difficult to extract the diagnostic features. Therefore, the contaminated EEG channels are often discarded by the medical practitioners which may result in less accurate diagnosis. Independent Component Analysis (ICA) and wavelet-based algorithms require reference electrodes, which will create discomfort to the patient/children and cause hindrance to the diagnosis of the NDD and Brain Computer Interface (BCI). Therefore, it would be ideal if these artefacts can be removed real time and on hardware platform in an automated fashion and denoised EEG can be used for online diagnosis in a pervasive personalised healthcare environment without the need of any reference electrode. In this thesis we propose a reliable, robust and automated methodology to solve the aforementioned problem and its subsequent hardware implementation results are also presented. 100 EEG data from Physionet, Klinik für Epileptologie, Universität Bonn, Germany, Caltech EEG databases and 3 EEG data from 3 subjects from University of Southampton, UK have been studied and nine exhaustive case studies comprising of real and simulated data have been formulated and tested. The performance of the proposed methodology is measured in terms of correlation, regression and R-square statistics and the respective values lie above 80%, 79% and 65% with the gain in hardware complexity of 64.28% and hardware delay 53.58% compared to state-of-the-art approach. We believe the proposed methodology would be useful in next generation of pervasive healthcare for BCI and NDD diagnosis and treatment.

Contents

Declaration	ii
Approval Sheet	iii
Acknowledgements	iv
Abstract	vi
1 Introduction	1
1.1 Motivation	2
2 Literature Review	4
3 EEG Artefacts Detection and Removal	7
3.1 Denoising	7
3.2 Muscle Artefacts Detection and Removal	9
3.3 Blink Artefacts Detection and Removal	13
4 ARM Cortex M0+ Overview	17
4.1 ARM Cortex M0+	17
4.1.1 About the Processor	17
4.1.2 Features	17
4.1.3 Interfaces	18
4.1.4 Configurable Option	18
4.2 Functional Description	18
4.2.1 About the Functions	18
4.2.2 Interfaces	21
5 AMBA AHB-Lite Protocol Specification	23
5.1 Introduction	23
5.2 About the Protocol	23
5.3 Components	24
5.3.1 Master	25

5.3.2	Decoder	25
5.3.3	Multiplexor	25
5.3.4	Slave	25
5.4	Operation of AMBA AHB-Lite System	26
5.5	Signal Description	26
5.5.1	Global Signals	27
5.5.2	Master Signals	27
5.5.3	Slave Signals	27
5.5.4	Decoder Signals	27
5.5.5	Multiplexor Signals	27
5.6	Transfer	28
5.6.1	Basic Transfer	28
5.6.2	Transfer Types	29
5.6.3	Locked Transfer	29
5.6.4	Transfer Size	30
5.6.5	Burst Operation	30
5.7	Bus Interconnect	32
5.7.1	Address Decoding	32
5.7.2	Bus Interconnection	33
5.8	Slave Response Signaling	33
5.8.1	Slave transfer responses	34
6	Integration and Implementation	36
6.1	Configuration Options	36
6.2	Key Integration Task	36
6.3	Functional Integration Guidelines	39
6.3.1	Clocks	39
6.3.2	Reset	39
6.3.3	Interface	40
6.4	Key Implementation Points	41
6.5	SoC Development Results	41
7	Results & Discussion	44
8	Conclusion	52
	References	53

List of Figures

3.1	System Block Diagram	8
3.2	Input Noisy Signal	9
3.3	Output Denoised Signal	9
3.4	EEG Signal containing Muscle Artefacts	12
3.5	EEG Signal after removing Muscle Artefacts	12
3.6	EEG Signal containing Blink Artefacts	16
3.7	EEG Signal after removing Blink Artefacts	16
4.1	Functional Block Diagram	20
5.1	AHB-Lite Block Diagram	24
5.2	AHB Master Interface	25
5.3	AHB Slave Interface	26
5.4	Slave Select Signals	32
5.5	Multiplexor interconnection	33
6.1	Integration and Implementation Flow	38
6.2	Implementation Flow	39
6.3	Integration of AHB-Lite with ARM Cortex M0+	42
6.4	Integration of AHB-Lite interface with peripheral	43
7.1	Mixed artefact signal Analysis. The shaded area (light grey) shows the position where artefacts are manually added and removed. (A) Amplitude vs Time plot for 10 second of raw clean EEG signal. (B) Muscle and Blink artefacts are added in the signal (A) in a random fashion. (C) Signal obtained after artefacts removal when the optimized conditions are applied.	47
7.2	Real EEG Signal Analysis (21 Channel): Input to the System	48
7.3	Real EEG Signal Analysis (21 Channel): Output from MATLAB Simulation	48
7.4	Real EEG Signal Analysis (21 Channel): Output from the FPGA Implementation	48

7.5	Variation of Hardware Complexity in terms of Transistor Count with different word-length(n)	51
7.6	Variation of Hardware Delay with different word-length(n)	51

List of Tables

4.1	Processor Configurable Options	19
5.1	Global Signal	27
5.2	Master Signals	28
5.3	Slave Signals	29
5.4	Decoder Signals	29
5.5	Multiplexor Signals	30
5.6	Transfer Type Encoding	30
5.7	Transfer Size Encoding	31
5.8	Burst Signal Encoding	31
5.9	HRESP Signal	34
5.10	Transfer Response	34
6.1	Cortex M0+ Option Summary	37
6.2	Cortex M0+ Level Clocks	40
6.3	Cortex M0+ Level Reset	40
6.4	AHB-Lite Signals	40
7.1	Performance Metrics for different cases of artefact addition and real data simulation on hardware platform (FPGA)	45
7.2	Performance Comparison by varying ‘ x ’ in Muscle Artefact and Blink Artefact Cases of LM & GM for alternate and random addition of artefact	46
7.3	Comparison with different State-of-the-art methods	49
7.4	FPGA Resource Utilization	49

Chapter 1

Introduction

Neurodevelopmental disorders (NDD) including Attention Deficit Hyperactivity Disorder (ADHD), Schizophrenia, Down syndrome, Autism Spectrum Disorder (ASD), intellectual retardation, learning disablement are impairments in the evolution of the brain or the central nervous system which manifest early in development, often during infancy or before child enters into socio-academic education. While the symptoms and behaviour of NDD including language and speech learning, motor synchronization, behaviour, retention, imagination underdevelopment, communication [1] differ from individual to individual, some children with such disabilities in childhood develop permanent damages. For example, children with ASD show impairment in social interaction, deficit in communication and motor coordination, repetitive or stereotyped behaviour, lack of cognitive skills, language loss, and atypical visual perception [2, 3, 4, 5, 6]. To diagnose NDD for ASD, Autism Diagnostic Observation Schedule (ADOS) and Autism Diagnostic Interview-Revised (ADI-R) are used comprising of a series structured tasks and interviews respectively involving the interactions among patient, examiner and parents. The examiner identifies the patient's response to the tasks and suggests a proper treatment procedure [7]. However, such procedures involve a constant observation on the children, significant amount of parenting, treatment time and huge and long-term expenses. The recent Neuroimaging techniques that discovered an overgrowth of the cortical white matter and abnormal pattern in frontal and temporal lobe during prenatal and postnatal period of brain evolution generally require a sedation and radioactive dye [8]. Both of the above-mentioned procedure requires high quality medical facilities for intensive care in home environment. On the other hand EEG- electrical recording of the brain systematic activity along the scalp, measured by the voltage fluctuations resulting from the ionic current which flows within the neuron [8, 9, 10], is cost-effective and non-invasive tool for the exploration of different brain regions for cognitive and other event related activities of a subject [8, 10]. The diagnostic feature extracted from the EEG signals can reveal the

brain functionality in the area of particular task and can be used as a biomarker to classify between the NDD and healthy control. In fact, EEG has also been arguably the most widely used mechanism for acquiring brain computer interface (BCI) signals from the brain for the control of computers or other devices via the modulation of neuro-logical activity in the participant's brain without the need of any activation of the efferent nervous system [11]. However in daily life, EEG signal is mixed with other biological signals of non-interest [10, 12], including blink and muscle artefacts it is difficult to extract the diagnostic feature. Therefore, Medical Practitioners during offline visual observation, discard the EEG channels containing these artefacts [2] which may result in less accurate diagnosis [13]. Similarly, in case of online automated diagnosis using EEG in tele-health framework under internet of things, these artefacts may cause wrong diagnosis triggering false alarm and causing panic.

1.1 Motivation

Recently there is an attempt to propose an online and automated EEG artefacts removal scheme in [11, 14, 15] targeting BCI where, although the processing is done online, the acquisition of the data is still done off-line which deviates from the need of our envisaged goal of having a pervasive personalized healthcare monitoring system. Therefore, a robust methodology which would remove the effect of these artefacts as well as retrieve EEG amidst the presence of these artefacts would be extremely helpful for BCI and also for enhanced diagnosis of NDD in real-time online personalized home care environment. But since, the frequency spectrum of blink and muscular artefact overlap with the normal EEG signal, it poses a commendable challenge in achieving the target of retrieving artefact free EEG in real-time automated fashion on hardware platform. To tackle this challenge, researchers use Independent Component Analysis (ICA) Blind source Separation and wavelet based time frequency algorithm to remove the ocular artefacts [10, 16, 17, 18, 19, 20, 21, 22, 23] and muscular artefacts [13, 15, 20, 22, 24] from the EEG. Although these methods are noninvasive, they require external ocular electrodes near the eyes, which will cause discomfort thereby making it unsuitable for the personalized remote health care. Furthermore, such arrangements make the patients/children conscious about the presence of these extra electrodes, which may cause hindrance to the appropriate diagnosis of the NDD. In [12], the need of these ocular electrodes have been eradicated however, this recent method does the processing of the data acquired offline [12]. Therefore, it would be ideal if

- these artefacts can be removed real time in an automated fashion and denoised EEG can be obtained for online diagnosis in a pervasive personalised healthcare environment on low complexity hardware

platform without the need of any external ocular electrode;

- this can be achieved with comparable accuracy when compared with state of the art approaches
- this can be implemented in a low complexity fashion on a chip to ensure the battery backup, that drives electronics, sustains for longer time than the state of the art approaches.

Motivated by this, in this thesis we propose a reliable, robust and automated low complexity hardware design methodology to remove blink and muscular artefacts real time without the need of any extra electrode and subsequently its hardware results and performance comparison with the state-of-the-art approaches are also presented.

Chapter 2

Literature Review

V Krisnaveni et.al [16] proposed a method to automatically identify slow varying ocular artefact (OA) zones and applying wavelet based adaptive thresholding algorithm only to the identified OA zones, which avoids the removal of background EEG information. Adaptive thresholding applied only to the OA zone does not affect the low frequency components in the non-OA zones and also preserves the shape (waveform) of the EEG signal in non-artefact zones which is of very much importance in clinical diagnosis. But the methodology presented here was found to be computationally intensive and required EOG signal for reference.

Carrie A Joyce et.al [18] presents a method based on blind source separation (BSS) for automatic removal of electro-ocular artefacts from EEG data. BSS is a signal-processing methodology that includes independent component analysis (ICA). In contrast to previously explored ICA-based methods for artefact removal, this method is automated. Moreover, the BSS algorithm described herein can isolate correlated electro-ocular components with a high degree of accuracy. Although the focus is on eliminating ocular artefacts in EEG data, the approach can be extended to other sources of EEG contamination such as cardiac signals, environmental noise, and electrode drift, and adapted for use with magnetoencephalographic (MEG) data, a magnetic correlate of EEG. Use of BSS for removal of ocular artefacts is computationally intensive and requires EOG & EMG signals as reference electrode.

Kevin T Sweeney et. al [21] proposed a technique known as ensemble empirical mode decomposition with canonical correlation analysis (EEMD-CCA) which is capable of operating on single-channel measurements. The EEMD technique is first used to decompose the single-channel signal into a multidimensional signal. The CCA technique is then employed to isolate the artefact components from the underlying signal using second-order statistics resulting in clean (artefact-free) signal.

D.J. McFarland et.al. [25] developing an electroencephalographic (EEG)-based brain-computer interface (BCI) system that could provide an alternative communication channel for those who are totally paralyzed or have other severe motor impairments. This laboratory BCI system digitizes 64 EEG channels from the system user (i.e., the subject), performs real-time spatial filtering and spectral analyses, uses the results to control a video display, continually adapts its analysis algorithm so as to convert the user's EEG control as efficiently as possible into display control, provides performance data on-line to the system operator (i.e., the investigator), and stores all data for later off-line analyses.

Doha Safieddine et.al [26] compare the ability of two stochastic approaches of blind source separation, namely independent component analysis (ICA) and canonical correlation analysis (CCA), and of two deterministic approaches namely empirical mode decomposition (EMD) and wavelet transform (WT) to remove muscle artefacts from EEG signals. To quantitatively compare the performance of these four algorithms, epileptic spike-like EEG signals were simulated from two different source configurations and artificially contaminated with different levels of real EEG-recorded myogenic activity. The efficiency of CCA, ICA, EMD, and WT to correct the muscular artefact was evaluated both by calculating the normalized mean-squared error between denoised and original signals and by comparing the results of source localization obtained from artefact-free as well as noisy signals, before and after artefact correction. Results shows that, for less noisy data, and when spikes arose from a single cortical source, the myogenic artefact was best corrected with CCA and ICA. Otherwise when spikes originated from two distinct sources, either EMD or ICA offered the most reliable denoising result for highly noisy data, while WT offered the better denoising result for less noisy data. These results suggest that the performance of muscle artefact correction methods strongly depend on the level of data contamination, and of the source configuration underlying EEG signals.

Miguel A Sovierzoski et.al [27] analyse the electrical behaviour of eye blink events acquired by EEG electrodes, and also developed a neural network classifier to identify them. This eye blink event classifier will be used as a part of a hybrid classifier of epileptic form events.

Borna Nouredin et. al [28] proposed a methodology to measure how much artefact is removed is combined with a measure of how much an OA removal algorithm is likely to distort underlying EEG in a single metric. Though the method is online, it requires a reference electrode for EOG signal.

C Guerrero-Mosquera et. al [29] presented a method for eye movement artefacts removal based on independent component analysis (ICA) and recursive least squares (RLS) is presented. The proposed algorithm combines the effective ICA capacity of separating artefacts from brain waves, together with the online interference cancellation achieved by adaptive filtering. Eye blink, saccades, eyes opening and

closing produce changes of potentials at frontal areas. For this reason, the method uses as a reference the electrodes closest to the eyes, which register vertical and horizontal eye movements in the electroencephalogram (EEG) caused by these activities as an alternative of using extra dedicated electrooculogram (EOG) electrode. Both reference signals and EEG components are first projected into ICA domain and then the interference is estimated using the RLS algorithm.

Maria Ansatasiadou et. al [30] introduced an automatic method for detection and removal of muscle artefacts from scalp EEG recordings, based on canonical correlation analysis (CCA). A classifier is designed in order to automatically discriminate between contaminated and non-contaminated EEG epochs using features based on altered autocorrelation structure and spectral characteristics during periods when it is contaminated by muscle activity.

Manish Tibdewal et. al [31] introduce a combination of methods to detect the presence of eye blink artefact in the EEG signal. EEG data signals were fed to an Artificial Neural Network (ANN). The neural network is trained to identify whether the particular recording contains eye blink artefacts or not. Once confirmed that the EEG data signal contains artefacts, it is processed further by using wavelet transform to detect the zones for which the part of signal is contaminated. Further artefact removal algorithm is applied to detected portion in order to prepare clean EEG data signals. If the artefact removing methods are directly applied to raw EEG data without detecting the artefact zone, it may be removed some important cerebral activities.

Chapter 3

EEG Artefacts Detection and Removal

Figure 3.1 depicts the block diagram of the proposed methodology. A 21 and 64 channel EEG signal were mixed individually with muscle and ocular artefacts in random ratio, then passed through **FastICA** [32] block. Considering, \mathbf{A} = mixing matrix, \mathbf{S} = source matrix, \mathbf{X} = input mixed matrix, **FastICA** works on $\mathbf{X} = \mathbf{A} * \mathbf{S}$ and estimates $\mathbf{Y} = \mathbf{W} * \mathbf{X}$ where, $\mathbf{Y} \approx \mathbf{S}$, \mathbf{W} = Unmixing matrix and \mathbf{Y} = estimated independent component matrix. The proposed method works on the output of the FastICA where the fundamental building block is Discrete Wavelet Transform (DWT) which is computed by passing the EEG signal through series of high pass filter with impulse response ‘ \mathbf{h} ’ resulting detailed coefficient ‘ \mathbf{C}_d ’ (step.1) and low pass filter with impulse response ‘ \mathbf{g} ’ resulting in approximate coefficient ‘ \mathbf{C}_a ’ (step.1) then sampled down [33]. At every decomposition level, output filter has half the frequency band of the input, so the frequency resolution has doubled. The ‘**Haar**’ wavelet [26] is the simplest wavelet possible, since it implemented only using addition and subtractions and has been found computation efficient.

3.1 Denoising

The FastICA output presented in previous section and shown in Fig. 3.1, usually corrupted with noise frequency ranging from 50 to 60 Hz, are fed to the denoising block for removing the environmental and surrounding electrical noise. Denoising is applied as a pre-processing step in the analysis of data i.e. estimating the unknown signal which is inherit, from the noisy data sample. Wavelet based denoising removes the noise present in the signal without affecting its characteristics. Discrete Wavelet transform

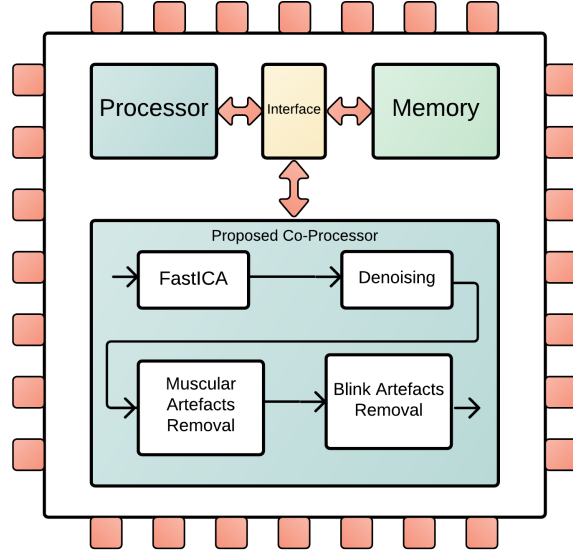


Figure 3.1: System Block Diagram

is applied to the signal, producing wavelet coefficients up to the level where noise is distinguished. Soft Thresholding method [34, 35] based on wavelets is applied to perform Denoising. The results of denoising block is shown in fig. 3.2 & 3.3 and corresponding pseudo code is presented in Pseudo code 1.

Pseudo Code 1: DWT Computation

Notations: F = sampling frequency, T = total time for which EEG is observed, C_a = DWT Approximate Coefficient, C_d = DWT Detailed Coefficient.

1. DWT Level 1

$$C_{aj}^1 = \frac{(f_i + f_{i+1})}{\sqrt{2}}, \quad C_{dj}^1 = \frac{(f_i - f_{i+1})}{\sqrt{2}}$$

j = number of coefficients in Level1; $1, 2, \dots, \frac{n}{2}$

2. DWT Level z

$$C_{ak}^z = \frac{(C_{aj}^{z-1} + C_{aj+1}^{z-1})}{\sqrt{2}}, \quad C_{dk}^z = \frac{(C_{aj}^{z-1} - C_{aj+1}^{z-1})}{\sqrt{2}}$$

k = number of coefficients in Level z ; $1, 2, \dots, \frac{n}{2^z}$

z = number of DWT Level Decomposition

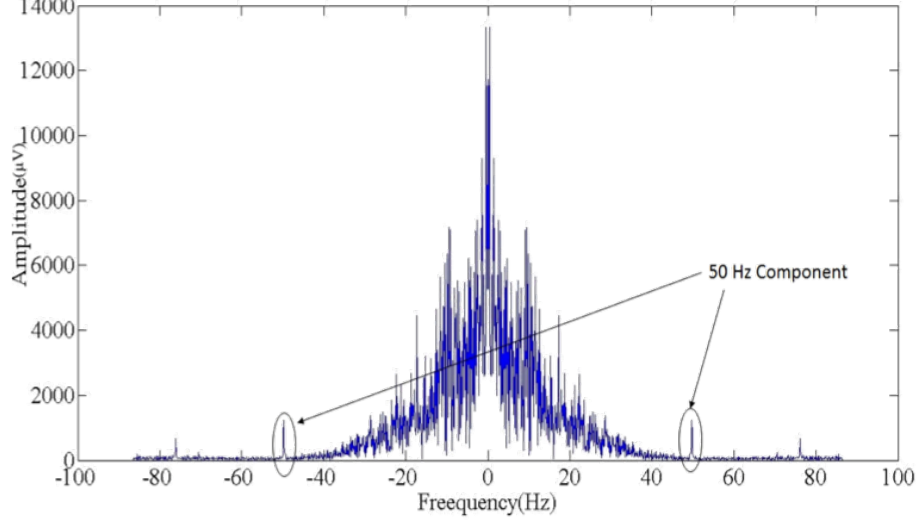


Figure 3.2: Input Noisy Signal

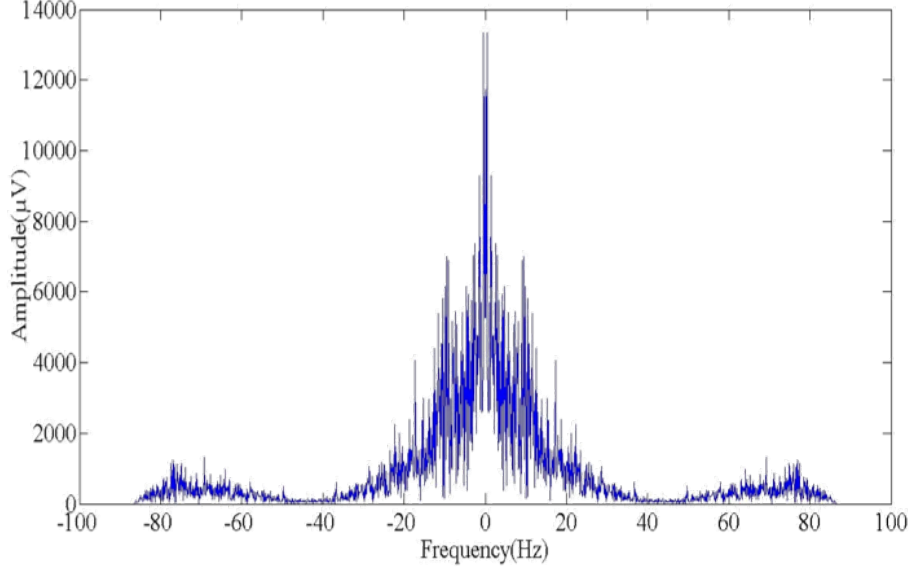


Figure 3.3: Output Denoised Signal

3.2 Muscle Artefacts Detection and Removal

Muscular or myogenic artefacts arise from the activity of different head muscle, neck movement, arm movement etc. which influence the EEG recordings [20, 35]. Myogenic artefacts lie in frequency range greater than **Beta band** (β) i.e. $16 - 31\text{Hz}$ [25] and have high power spectral density [26] than the normal EEG as shown in Fig. 2. First C_{dj}^1 and second C_{dk}^2 detailed coefficients are computed in step 1 of the pseudo code 2. Muscular artefacts overlap in C_{aj}^1 and C_{ak}^2 region, hence, both the coefficients are analysed here. After wavelet transform decomposition, the length of C_{aj}^1 is twice as that of C_{ak}^2 . Hence, alternate zero padding (step 2a of Pseudo code 2) is done making C_{aj}^1 and C_{ak}^2 equal in length. If sampling frequency is ‘F’ Hz and the signal is observed for ‘T’ sec then ‘FT’ number of samples are accumulated.

C_{aj}^1 and $C_{ak}^{(2*)}$ are divided into ' S_x ' equal frames resulting ' x ' number of samples per frame. Since ' \mathbf{FT} ' needs to be stored, higher ' \mathbf{FT} ' indicates more memory requirement. However, if ' \mathbf{FT} ' is high, the performance of the proposed methodology would be precise. Our method is targeted towards on-chip implementation for home user and hence the value of ' \mathbf{T} ' is kept as $10sec$. Also, varying the number of frames ' S_x ' in detailed coefficients would result in better performance for artefact removal and hence an optimum value of ' x '. To determine the maximum efficiency for artefact rejection in alternate and random order of muscle artefacts, samples per frame (x) was varied as 4, 10, 20, 33, 43, 66, 86, 107, 122 and 170, which equally divide ' \mathbf{FT} '. The corresponding Correlation, regression and R-square value has determined that the maximum similarity between clean input EEG and output EEG in the range of 66 to 107 for a particular ' \mathbf{FT} ' in Table 7.1. Wavelet power spectral density (WPS) can be computed as shown in step 2b of the pseudo code 2. Frame by frame comparison of $P_{(a,b)_{(a=1)}}$ & $P_{(a,b)_{(a=2)}}$ for $1 < b < S_x$ is done to find out the maximum as shown in step 3 of the pseudo code 2. Assume M_b denote the maximum after comparison and M be the mean. The mean of the calculated maxima M for each time frame is found using equation in step 4 of the pseudo code 2. The mean M is compared with $P_{(a,b)_{(a=1)}}$ & $P_{(a,b)_{(a=2)}}$ for $1 < b < S_x$. If the condition in step 4 is found to be true then all the samples in that particular frame is made zero. Similar procedure is applied for second detailed coefficient and corresponding samples of a particular frame is made zero. The signal reconstruction is carried out using the inverse wavelet transform [8] as depicted in equation of step 5 and reconstructed EEG f'_i is obtained. Figure 3.4 & 3.5 shows results of the methodology for Muscular artefact detection and removal. The corresponding code is given in Pseudo code 2.

Pseudo Code 2: Muscular Artefact Detection and Removal

Notations: F = sampling frequency, T = total time for which EEG is Observed, C_a = DWT Approximate Coefficient, C_d = DWT Detailed Coefficient, C_{dr+1}^{2*} = Detailed Coefficient obtained after alternate zero padding, a = level of DWT, r_x = counter, b = frame number $1, 2, \dots, S_x = \frac{n}{2x}$, S_x = total number of frames in detailed coefficient, x = number of samples/frames, r_{ca} & r_{cd} are reconstructed approximate coefficient, f'_i = Reconstructed EEG after Muscular artefact removal.

1. Wavelet Power Spectrum Calculation

(a) Alternate Zero Padding

$$C_{d_r}^{2*} = C_{a_k}^2 \text{ For } 1 < k < \frac{n}{4}$$

$$C_{d_{r+1}}^{2*} = 0 \text{ For } 1 < k < \frac{n}{2}$$

$$r = r + 2; \quad k = k + 1$$

(b) Wavelet Power Spectrum

$$P_{a,b} = \sum_{r_x}^{x-1} [C_{d_{r_x \times b}}^a]^2$$

2. Frame by Frame Comparison & Calculation of mean WPS

$$M_b = \max[(P_{a,b})_{a=1}, (P_{a,b})_{a=2}] \text{ For } 1 < b < S_x$$

3. Mean Calculation

$$\text{mean} = M = \frac{\sum_{b=1}^{S_x} M_b}{S_x}$$

4. Comparison of mean (M) with each level of WPS (Detect and Remove Muscle Artefacts)

$$P_{a,b} > M \text{ for } 1 < b < S_x \text{ \& } 1 < a < 2$$

Then,

$$C_{(d)r(x \times b)}^a = 0 \text{ for } 1 < r_x < x - 1$$

5. Reconstruction to get artefact free EEG

(a) Reconstruction Level 3

$$r_{C_{a_l}^3} = \frac{C_{a_m}^4 + C_{d_m}^4}{\sqrt{2}} \quad r_{C_{d_{l+1}}^3} = \frac{C_{a_m}^4 - C_{d_m}^4}{\sqrt{2}}$$

(b) Reconstruction Level 2

$$r_{C_{a_k}^2} = \frac{r_{C_{a_l}^3} + C_{d_l}^3}{\sqrt{2}} \quad r_{C_{d_{k+1}}^2} = \frac{r_{C_{a_l}^3} - C_{d_l}^3}{\sqrt{2}}$$

(c) Reconstruction Level 1

$$r_{C_{a_j}^1} = \frac{r_{C_{a_k}^2} + C_{d_k}^2}{\sqrt{2}} \quad r_{C_{d_{j+1}}^1} = \frac{r_{C_{a_k}^2} - C_{d_k}^2}{\sqrt{2}}$$

(d) Final Reconstruction to get clean EEG

$$f'_i = \frac{r_{C_{a_j}^1} + C_{d_j}^1}{\sqrt{2}} \quad f'_{i+1} = \frac{r_{C_{a_j}^1} - C_{d_j}^1}{\sqrt{2}}$$

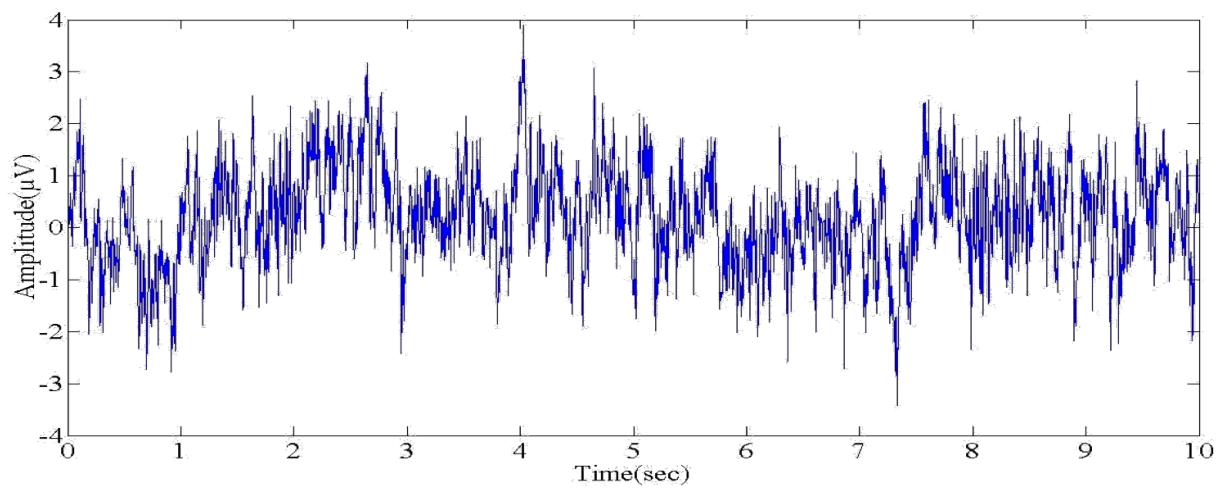


Figure 3.4: EEG Signal containing Muscle Artefacts

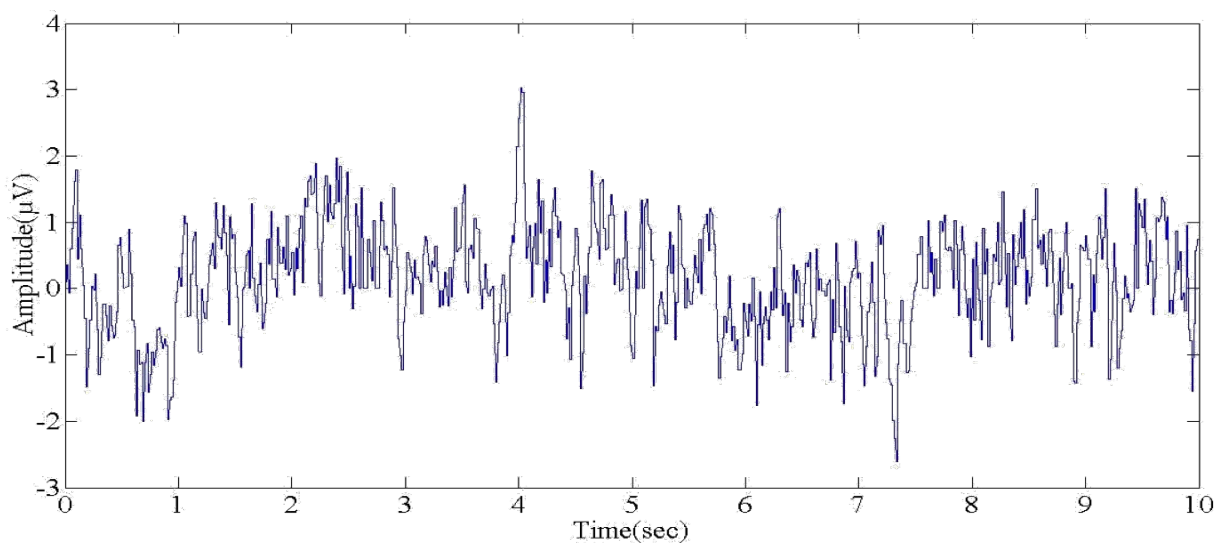


Figure 3.5: EEG Signal after removing Muscle Artefacts

3.3 Blink Artefacts Detection and Removal

An eye blink can last up to $400ms$ [18, 23] and lie in **Theta** (θ) i.e. $4 - 7Hz$ and **Mu** (μ) i.e. $8 - 12Hz$ frequency range of the EEG spectrum [35, 25]. These have a magnitude 10 times higher than the brain electrical signal [18]. It occurs as a large dip on the frontal channels **FP1-F3**, **FP2-F4**, **FP1-F7** and **FP2-F8**, [16, 35, 25, 27] (according to the International 10-20 System of Electrode Placement) because these channels are located nearest to eyes. The eyeball acts as a dipole, with cornea as positive pole with respect to the retina. When eye goes from open to close the electrode sense a downward reflection. Similarly, when eye goes from close to open an upward reflection occurs at the electrode. This results a high amplitude negative peak in EEG [15, 18, 26]. EEG signal decomposition is done till theta band of the EEG spectrum is reached to obtain C_{am}^4 . Further, the negative sample index of C_{am}^4 is taken and time domain mapping of all the negative peaks in C_{am}^4 is carried out as shown in step 1 of the pseudo code 3. In frequency domain, theta band is reached and corresponding time domain mapping is done to extract artefacts in that band only. For each of the selected negative sample after mapping in time domain, a window is created (step 2 of pseudo code 3) to effectively select the blink and the negative peak is obtained. This process is repeated for the entire range of the signal. The negative peaks obtained from steps 3 are stored and **Global Mean** (GM) is computed as indicated in step 4 of pseudo code 3. This GM is used as threshold to remove the eye blink artefact. Three cases are determined while removing the blink artefacts. In **case a**, samples after satisfying the condition are simply made zero as shown in step 5a. In **case b**, assigns value of GM to the artefact region (step 5b). In **case c**, **Local Mean** (LM) is calculated for each window in step 5c (ii). If the conditions are satisfied in step 5c (iii) corresponding values GM or LM of are assigned to the artefact region of the window. The highest value of correlation, regression and R-square among the three cases indicates the maximum performance and high efficiency for artefact removal as indicated in Table 7.1. Thus, the signal obtained is free from blink artefact and shown in Fig. 3.6 & 3.7. The corresponding code is given in Pseudo code 3.

Pseudo Code 3: Eye Blink Artefact Detection and Removal

Notations: m = number of coefficients in Level4 $1, 2, \dots, \frac{n}{16}$, neg_m = Store ‘ m ’ for negative C_{am}^4 , neg_m_low = Lower point of window, neg_m_high = Highest point of window, B = stores negative samples in f'_i for a particular window, d = total number of negative samples found in $f'_i(t)$, t = Time for which EEG signal is analysed. w = window number, d = number of negative samples in f'_i at

window, $B_{(w,d)} = 2$ dimensional vector to store negative samples.

1. Get negative values in DWT level 4

$$\text{if, } C_{am}^4 < 0$$

$$\text{then, } neg_C_{am}^4 = neg_m; neg_m = m$$

2. Time domain mapping of neg_m & create window around it

$$neg_m_low = neg_m - 0.2$$

$$neg_m_high = neg_m + 0.2$$

3. Find negative samples in the above created window in time domain

$$\text{If, } (t > neg_m_low \ \&\& \ t < neg_m_high)$$

$$\text{If, } (f'_i < 0),$$

$$\text{then, } B_d = f'_i(t) \quad \text{For, } neg_m_low < t < neg_m_high$$

4. Mean Calculation

- (a) Global Mean

$$GM = \frac{\sum_{c=1}^d B_c}{d}$$

5. Remove Artefacts

- (a) Make Sample Zero

$$\text{If, } f'_i < GM$$

$$\text{then, } f'_i = 0$$

- (b) Make sample value equal to GM

$$\text{If, } f'_i < GM$$

then, $f'_i = GM$

(c) Make sample equal to LM or GM depending upon lesser negativity

i. If, $(t > neg_m_low \ \&\& \ t < neg_m_high)$

If, $(f'_i < 0)$

then, $B_{(w,d)} = f'_i$

ii. Mean Calculation

$$LM_w = \frac{\sum_{c=1}^d B_{w,c}}{d}$$

iii. Remove Artefacts

If, $f'_i < GM$

If, $GM < LM_w$

then, $f'_i = GM$

Else if, $GM > LM_w$

then, $f'_i = LM_w$

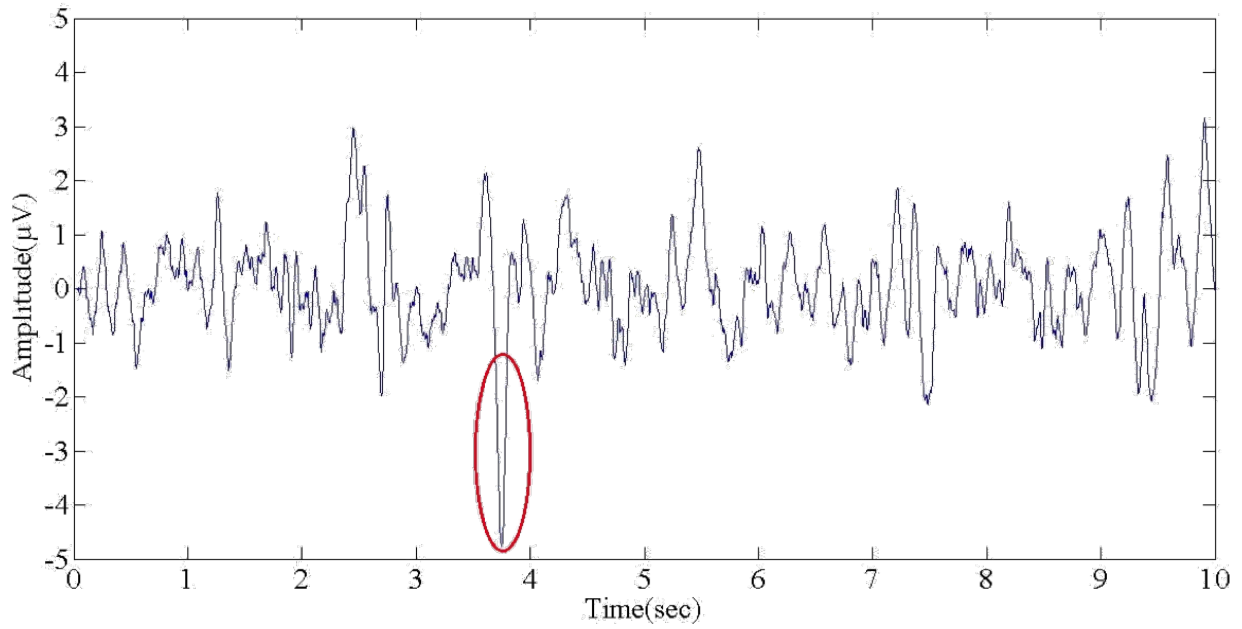


Figure 3.6: EEG Signal containing Blink Artefacts

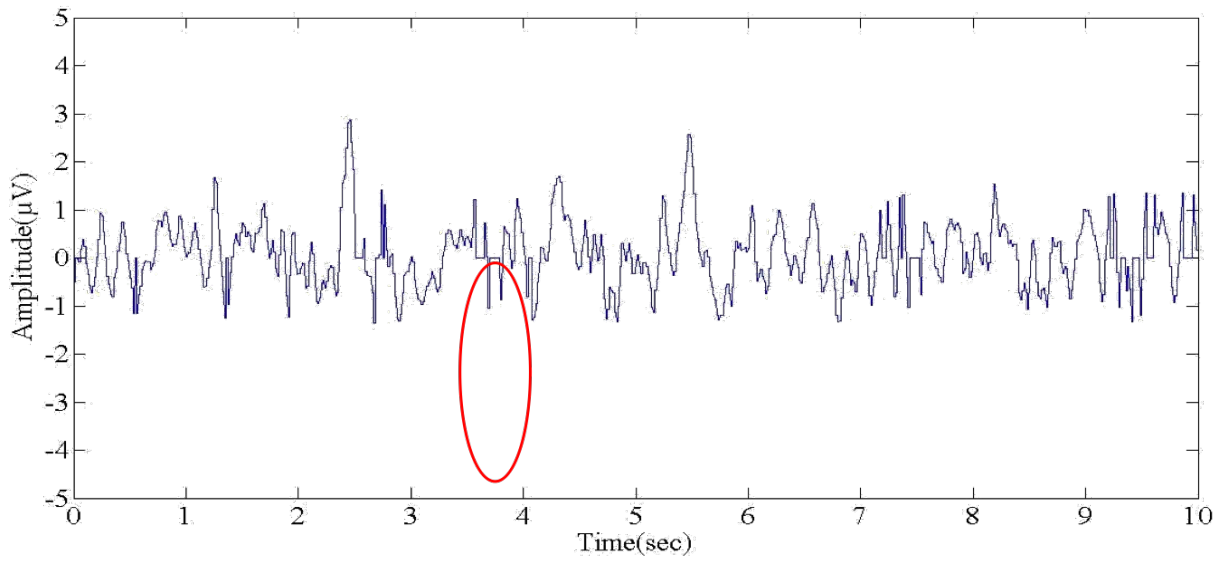


Figure 3.7: EEG Signal after removing Blink Artefacts

Chapter 4

ARM Cortex M0+ Overview

4.1 ARM Cortex M0+

4.1.1 About the Processor

The Cortex-M0+ processor is a very low gate count, highly energy efficient processor that is intended for microcontroller and deeply embedded applications that require an area optimized, low-power processor. The Cortex-M0+ processor supports State Retention and Power Gating (SRPG) with up to three power domains to enable very energy efficient silicon implementation and a trace interface

4.1.2 Features

The processor features and benefits are:

- Tight integration of system peripherals reduces area and development costs.
- Thumb instruction set combines high code density with 32-bit performance.
- Support for single-cycle I/O access.
- Power control optimization of system components.
- Integrated sleep modes for low power consumption.
- Fast code execution enables running the processor with a slower clock or increasing sleep mode time.
- Optimized code fetching for reduced flash and ROM power consumption.

- Hardware multiplier.
- Deterministic, high-performance interrupt handling for time-critical applications.
- Deterministic instruction cycle timing.
- Support for system level debug authentication.
- Serial Wire Debug reduces the number of pins required for debugging.
- Support for optional instruction trace.

4.1.3 Interfaces

The interfaces included in the processor for external access include:

1. External AHB-Lite interface
2. Debug Access Port (DAP)
3. Optional single-cycle I/O Port
4. Execution Trace Interface

4.1.4 Configurable Option

Table 4.1 shows the processor configuration options available at implementation time.

4.2 Functional Description

4.2.1 About the Functions

The Cortex-M0+ processor is a configurable, multistage, 32-bit RISC processor. It has an AMBA AHB-Lite interface and includes an NVIC component. It also has optional hardware debug, single-cycle I/O interfacing, and memory-protection functionality. The processor can execute Thumb code and is compatible with other Cortex-M profile processors.

Figure 4.1 shows the processor functional block diagram.

1. A low gate count processor that features:
 - The ARMv6-M Thumb instruction set.
 - Thumb-2 technology.

Table 4.1: Processor Configurable Options

Features	Configurable Option
Interrupts	External Interrupts 0 – 32
Data endianness	Little-endian or big-endian
SysTick timer	Present or absent
Number of watchpoint comparators	0, 1, 2
Number of breakpoint comparators	0, 1, 2, 3, 4
Halting debug support	Present or absent
Multiplier	Fast or small
Single-cycle I/O port	Present or absent
Wake-up interrupt controller	Supported or not supported
Vector Table Offset Register	Present or absent
Unprivileged/Privileged support	Present or absent
Memory Protection Unit	Not present or 8-region
Reset all registers	Present or absent
Instruction fetch width	16-bit only or mostly 32-bit

- Optionally, an ARMv6-M compliant 24-bit SysTick timer.
- A 32-bit hardware multiplier. This can be the standard single-cycle multiplier, or a 32-cycle multiplier that has a lower area and performance implementation.
- Support for either little-endian or byte invariant big-endian data accesses.
- The ability to have deterministic, fixed-latency, interrupt handling.
- Load/store multiple and multicycle multiply instructions that can be abandoned and restarted to facilitate rapid interrupt handling.
- Optionally, Unprivileged/Privileged support for improved system integrity.
- C Application Binary Interface compliant exception model.
- Low power sleep-mode entry using Wait For Interrupt (WFI), Wait For Event (WFE) instructions, or the return from interrupt sleep-on-exit feature.

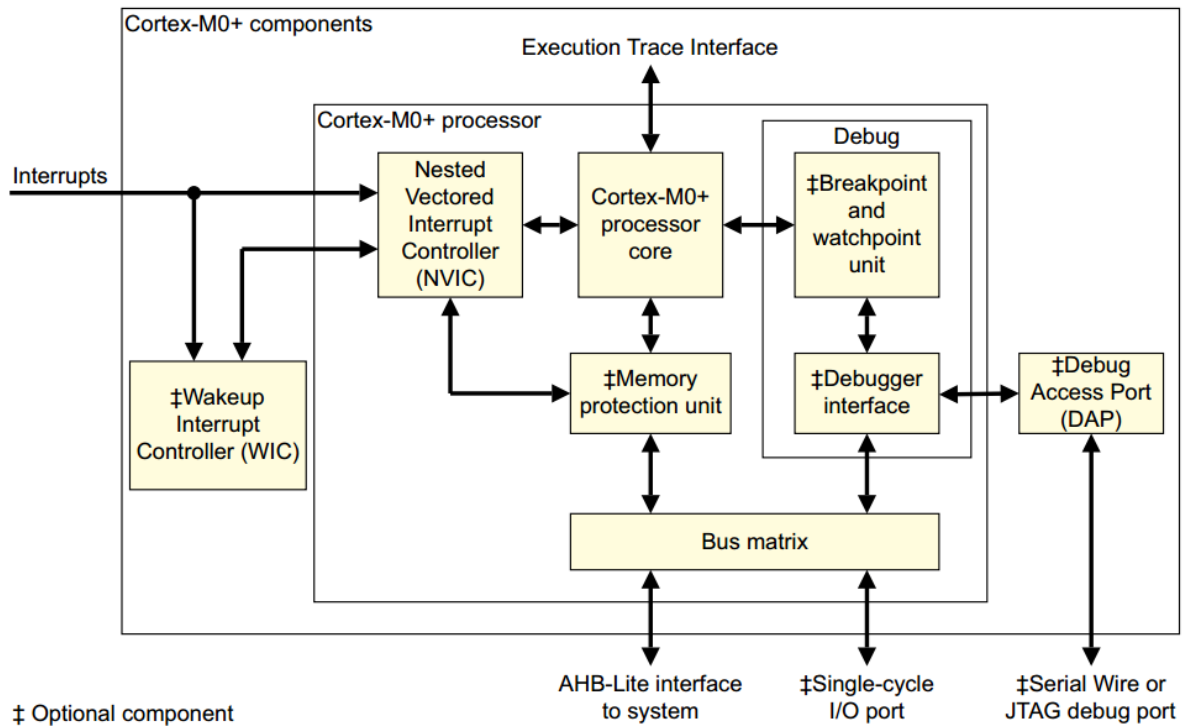


Figure 4.1: Functional Block Diagram

2. NVIC that features:

- Up to 32 external interrupt inputs, each with four levels of priority.
- Dedicated Non-Maskable Interrupt (NMI) input.
- Support for both level-sensitive and pulse-sensitive interrupt lines.
- Optional Wake-up Interrupt Controller (WIC), providing ultra-low power sleep mode support.
- Optional relocation of the vector table.

3. Optional debug support:

- Zero to four hardware breakpoints.
- Zero to two watchpoints.
- Program Counter Sampling Register (PCSR) for non-intrusive code profiling, if at least one hardware data watchpoint is implemented.
- Single step and vector catch capabilities.
- Support for unlimited software breakpoints using BKPT instruction.
- Non-intrusive access to core peripherals and zero-waitstate system slaves through a compact bus matrix. A debugger can access these devices, including memory, even when the processor

is running.

- Full access to core registers when the processor is halted.
- Optional, low gate-count CoreSight compliant debug access through a Debug Access Port (DAP) supporting either Serial Wire or JTAG debug connections.

4. Bus interfaces:

- Single 32-bit AMBA-3 AHB-Lite system interface that provides simple integration to all system peripherals and memory.
- Optional single 32-bit single-cycle I/O port.
- Optional single 32-bit slave port that supports the DAP.

5. Optional Memory Protection Unit (MPU):

- Eight user configurable memory regions.
- Eight sub-region disables per region.
- Execute never (XN) support.
- Default memory map support.

4.2.2 Interfaces

The interfaces included in the processor for external access include:

1. **External AHB-Lite interface:**

Transactions on the AHB-Lite interface are always marked as non-sequential. Processor accesses and debug accesses share the external interface to external AHB peripherals.

The processor accesses take priority over debug accesses

2. **Debug Access Port (DAP):**

The processor is implemented with either a low gate count Debug Access Port (DAP) or a full CoreSight DAP.

The low gate count Debug Access Port (DAP) provides a Serial Wire or JTAG debug-port, and connects to the processor slave port to provide full system-level debug access.

The full CoreSight DAP system enables the processor to provide full multiprocessor debug with simultaneous halt and release cross-triggering capabilities.

3. Optional single-cycle I/O Port:

The processor optionally implements a single-cycle I/O port that provides very high speed access to tightly-coupled peripherals, such as general-purpose-I/O (GPIO). The port is accessible both by loads and stores, from the processor and from the debugger.

A code cannot be executed from the I/O port.

4. Execution Trace Interface:

The processor optionally implements an interface for the Micro Trace Buffer execution trace component.

Chapter 5

AMBA AHB-Lite Protocol

Specification

5.1 Introduction

The ARM Advanced Microcontroller Bus Architecture (AMBA) is an open-standard, on-chip interconnect specification for the connection and management of functional blocks in system-on-a-chip (SoC) designs. It facilitates development of multi-processor designs with large numbers of controllers and peripherals. AHB is a bus protocol introduced in Advanced Microcontroller Bus Architecture version 2. In addition to previous release, it has the following features:

- large bus-widths (64/128 bit).

A simple transaction on the AHB consists of an address phase and a subsequent data phase (without wait states: only two bus-cycles). Access to the target device is controlled through a MUX (non-tristate), thereby admitting bus-access to one bus-master at a time. AHB-Lite is a subset of AHB formally defined in the AMBA 3 standard. This subset simplifies the design for a bus with a single master. This chapter provides an overview of the AHB-Lite protocol.

5.2 About the Protocol

AMBA AHB-Lite addresses the requirements of high-performance synthesizable designs. It is a bus interface that supports a single bus master and provides high-bandwidth operation. AHB-Lite implements the features required for high-performance, high clock frequency systems including:

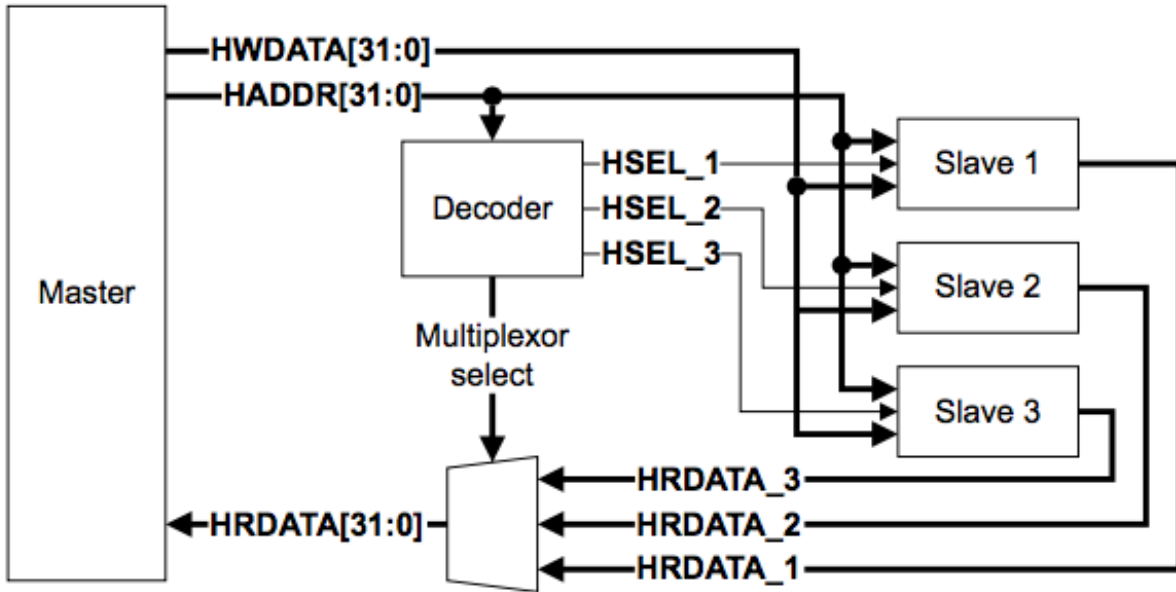


Figure 5.1: AHB-Lite Block Diagram

- burst transfers
- single-clock edge operation
- non-tristate implementation
- wide data bus configurations, 64, 128, 256, 512, and 1024 bits.

Figure 5.1 shows a single master AHB-Lite system design with one AHB-Lite master and three AHB-Lite slaves. The bus interconnect logic consists of one address decoder and a slave-to-master multiplexor. The decoder monitors the address from the master so that the appropriate slave is selected and the multiplexor routes the corresponding slave output data back to the master.

5.3 Components

The main component types of an AHB-Lite system are described below:

- Master
- Decoder
- Multiplexor
- Slave

5.3.1 Master

An AHB-Lite master provides address and control information to initiate read and write operations. Figure 5.2 shows an AHB-Lite master interface.

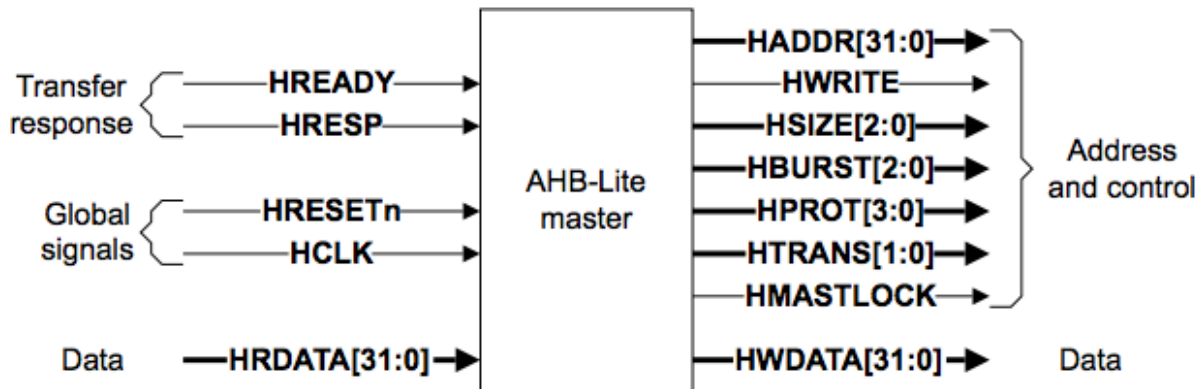


Figure 5.2: AHB Master Interface

5.3.2 Decoder

Decoder decodes the address of each transfer and provides a select signal for the slave that is involved in the transfer. It also provides a control signal to the multiplexor.

5.3.3 Multiplexor

A slave-to-master multiplexor is required to multiplex the read data bus and response signals from the slaves to the master. The decoder provides control for the multiplexor.

5.3.4 Slave

An AHB-Lite slave responds to transfers initiated by masters in the system. The slave uses the HSELx select signal from the decoder to control when it responds to a bus transfer. The slave signals back to the master. Figure 5.3 shows an AHB-Lite slave interface.

- the success
- failure
- or waiting of the data transfer.

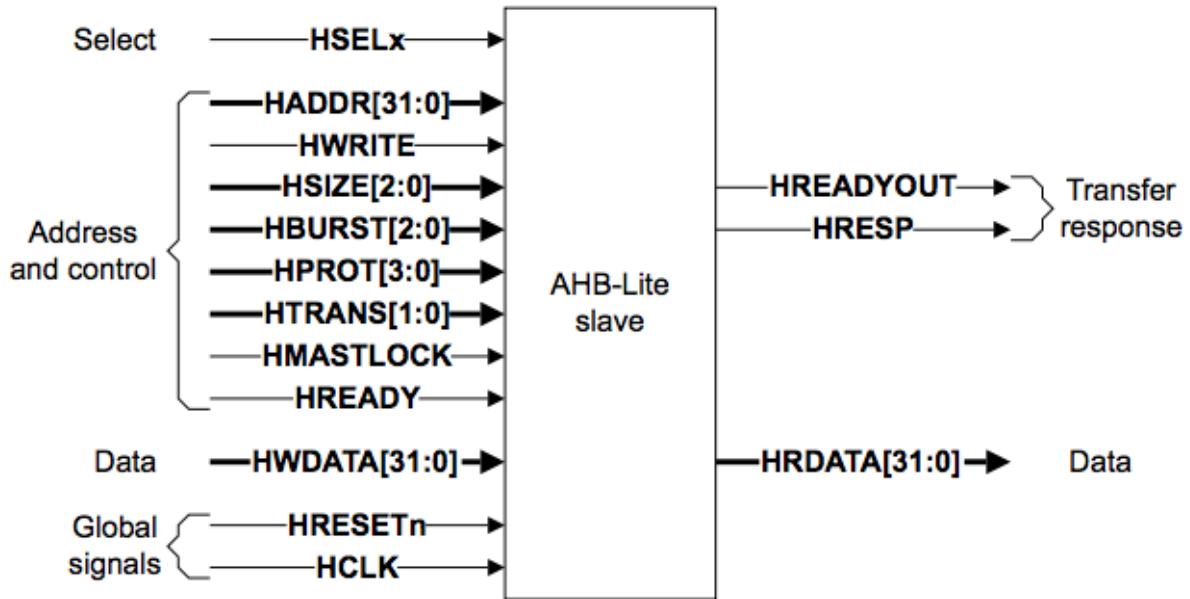


Figure 5.3: AHB Slave Interface

5.4 Operation of AMBA AHB-Lite System

The master starts a transfer by driving the address and control signals. These signals provide information about the address, direction, width of the transfer, and indicate if the transfer forms part of a burst. The write data bus moves data from the master to a slave, and the read data bus moves data from a slave to the master.

Every transfer consists of:

- **Address phase** one address and control cycle
- **Data phase** one or more cycles for the data.

A slave cannot request that the address phase is extended and therefore all slaves must be capable of sampling the address during this time. However, a slave can request that the master extends the data phase by using HREADY. This signal, when LOW, causes wait states to be inserted into the transfer and enables the slave to have extra time to provide or sample data.

The slave uses HRESP to indicate the success or failure of a transfer.

5.5 Signal Description

This section describes the protocol signals. It contains the following subsections:

- Global signals

- Master signals
- Slave signals
- Decoder signals
- Multiplexor signals

5.5.1 Global Signals

Table 5.1 lists the protocol of Global Signal.

Table 5.1: Global Signal

Name	Source	Description
HCLK	Clock Source	The bus clock times all bus transfers. All signal timings are related to the rising edge of HCLK .
HRESETn	Reset Controller	The bus reset signal is active LOW and resets the system and the bus.

5.5.2 Master Signals

Table 5.2 lists the protocol signals generated by a master.

5.5.3 Slave Signals

Table 5.3 lists the protocol signals generated by a slave.

5.5.4 Decoder Signals

Table 5.4 lists the protocol signals generated by the decoder.

5.5.5 Multiplexor Signals

Table 5.5 lists the protocol signals generated by the multiplexor.

Table 5.2: Master Signals

Name	Destination	Description
HADDR[31:0]	Slave and Decoder	The 32-bit system address bus.
HBURTS[2:0]	Slave	The burst type indicates if the transfer is a single transfer or forms part of a burst.
HMASTERLOCK	Slave	When HIGH, this signal indicates that the current transfer is part of a locked sequence.
HPROT[3:0]	Slave	The protection control signals provide additional information about a bus access and are primarily intended for use by any module that wants to implement some level of protection. The signals indicate if the transfer is an opcode fetch or data access, and if the transfer is a privileged mode access or user mode access.
HSIZE[2:0]	Slave	Indicates the size of the transfer.
HTRANS[1:0]	Slave	Indicates the transfer type of the current transfer.
HWDATA[31:0]	Slave	The write data bus transfers data from the master to the slaves during write operations. A minimum data bus width of 32 bits is recommended. However, this can be extended to enable higher bandwidth operation.
HWRITE	Slave	Indicates the transfer direction. When HIGH this signal indicates a write transfer and when LOW a read transfer.

5.6 Transfer

5.6.1 Basic Transfer

An AHB-Lite transfer consists of two phases:

Address Lasts for a single **HCLK** cycle unless its extended by the previous bus transfer.

Data That might require several **HCLK** cycles. Use the **HREADY** signal to control the number of clock cycles required to complete the transfer.

HWRITE controls the direction of data transfer to or from the master. Therefore, when:

- **HWRITE** is **HIGH**, it indicates a write transfer and the master broadcasts data on the write data bus, **HWDATA[31:0]**.
- **HWRITE** is **LOW**, a read transfer is performed and the slave must generate the data on the read data bus, **HRDATA[31:0]**.

Table 5.3: Slave Signals

Name	Destination	Description
HRDATA [31:0]	Multiplexor	During read operations, the read data bus transfers data from the selected slave to the multiplexor. The multiplexor then transfers the data to the master. A minimum data bus width of 32 bits is recommended. However, this can be extended to enable higher bandwidth operation.
HREADYOUT	Multiplexor	When HIGH , the HREADYOUT signal indicates that a transfer has finished on the bus. This signal can be driven LOW to extend a transfer.
HRESP	Multiplexor	When LOW , the HRESP signal indicates that the transfer status is OKAY . When HIGH , the HRESP signal indicates that the transfer status is ERROR .

Table 5.4: Decoder Signals

Name	Destination	Description
HSELx	Slave	Each AHB-Lite slave has its own slave select signal HSELx and this signal indicates that the current transfer is intended for the selected slave. When the slave is initially selected, it must also monitor the status of HREADY to ensure that the previous bus transfer has completed, before it responds to the current transfer.

5.6.2 Transfer Types

Transfers can be classified into one of four types, as controlled by **HTRANS**[1:0]. Table 5.6 lists these encoding scheme.

5.6.3 Locked Transfer

If the master requires locked accesses then it must also assert the **HMASTLOCK** signal. This signal indicates to any slave that the current transfer sequence is indivisible and must therefore be processed before any other transactions are processed.

Typically the locked transfer is used to maintain the integrity of a semaphore, by ensuring that the slave does not perform other operations between the read and write phases of a microprocessor **SWP** instruction.

Most slaves have no requirement to implement **HMASTLOCK** because they are only capable of performing transfers in the order they are received. Slaves that can be accessed by more than one master, for example, a *Multi-Port Memory Controller* (MPMC) must implement the **HMASTLOCK** signal.

Table 5.5: Multiplexor Signals

Name	Destination	Description
HRDATA[31:0]	Master	Read data bus, selected by the decoder.
HREADY	Master and Slave	When HIGH , the HREADY signal indicates to the master and all slaves, that the previous transfer is complete.
HRESP	Master	Transfer response, selected by the decoder.

Table 5.6: Transfer Type Encoding

HTRANS[1:0]	Type	Description
b'00	IDLE	A master uses an IDLE transfer when it does not want to perform a data transfer. Slaves must always provide a zero wait state OKAY response to IDLE transfers and the transfer must be ignored by the slave.
b'01	BUSY	The BUSY transfer type enables masters to insert idle cycles in the middle of a burst. This transfer type indicates that the master is continuing with a burst but the next transfer cannot take place immediately. When a master uses the BUSY transfer type the address and control signals must reflect the next transfer in the burst.
b'10	NONSEQ	Indicates a single transfer or the first transfer of a burst. The address and control signals are unrelated to the previous transfer. Single transfers on the bus are treated as bursts of length one and therefore the transfer type is NON-SEQUENTIAL.
b'11	SEQ	The remaining transfers in a burst are SEQUENTIAL and the address is related to the previous transfer. The control information is identical to the previous transfer.

5.6.4 Transfer Size

HSIZE[2:0] indicates the size of a data transfer. Table 5.7 lists the possible transfer sizes.

The transfer size set by **HSIZE** must be less than or equal to the width of the data bus. For example, with a 32 – bit data bus, **HSIZE** must only use the values b'000, b'001, or b'010. Use **HSIZE** in conjunction with **HBURST**, to determine the address boundary for wrapping bursts.

5.6.5 Burst Operation

Bursts of 4, 8, and 16-beats, undefined length bursts, and single transfers are defined in this protocol. It supports incrementing and wrapping bursts:

- Incrementing bursts access sequential locations and the address of each transfer in the burst is an increment of the previous address.

Table 5.7: Transfer Size Encoding

HSIZE[2:0]	Size(bits)	Description
b'000	8	Byte
b'001	16	Halfword
b'010	32	Word
b'011	64	DoubleWord
b'100	128	4 - Word Line
b'101	256	8 - Word Line
b'110	512	-
b'111	1024	-

- Wrapping bursts wrap when they cross an address boundary. The address boundary is calculated as the product of the number of beats in a burst and the size of the transfer. The number of beats are controlled by **HBURST** and the transfer size is controlled by **HSIZE**.

HBURST[2:0] controls the burst type. Table 5.8 lists the possible burst types.

Table 5.8: Burst Signal Encoding

HBURST[2:0]	Size(bits)	Description
b'000	SINGLE	Single Burst
b'001	INCR	Incremental Burts of undefined length
b'010	WRAP4	4 - Beat Wrapping Burst
b'011	INCR4	4 - Beat Incrementing Burst
b'100	WRAP8	8 - Beat Wrapping Burst
b'101	INCR8	8 - Beat Incrementing Burst
b'110	WRAP16	16 - Beat Wrapping Burst
b'111	INCR16	16 - Beat Incrementing Burst

5.7 Bus Interconnect

This chapter describes the additional interconnect logic required for AHB-Lite systems. It contains the following sections:

- Address Decoding
- Bus Interconnection

5.7.1 Address Decoding

A central address decoder provides a select signal, **HSEL_x**, for each slave on the bus. The select signal is a combinatorial decode of the high-order address signals.

A slave must only sample the **HSEL_x**, address, and control signals when **HREADY** is **HIGH**, indicating that the current transfer is completing. Under certain circumstances it is possible that **HSEL_x** is asserted when **HREADY** is **LOW**, but the selected slave has changed by the time the current transfer completes. The minimum address space that can be allocated to a single slave is 1KB. All masters are designed so that they do not perform incrementing transfers over a 1KB address boundary. This ensures that a burst never crosses an address decode boundary.

Figure 5.4 shows the **HSEL_x** slave select signals generated by the decoder.

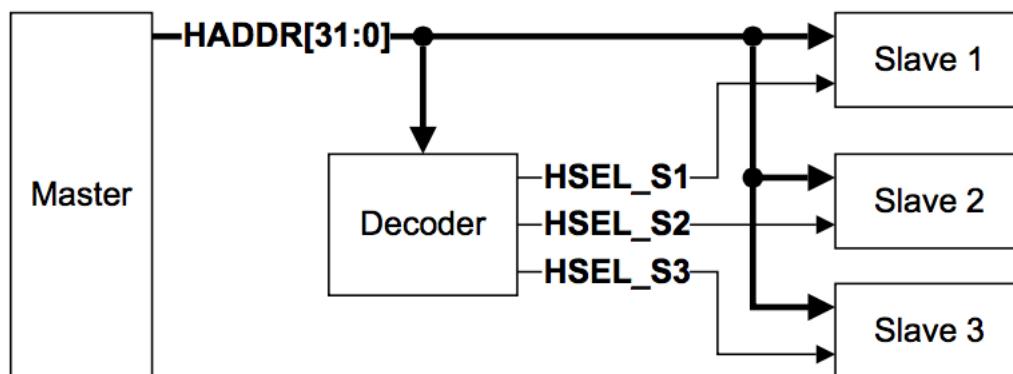


Figure 5.4: Slave Select Signals

5.7.1.1 Default slave

If a system design does not contain a completely filled memory map then you must implement an additional default slave to provide a response when any of the nonexistent address locations are accessed. If a **NONSEQUENTIAL** or **SEQUENTIAL** transfer is attempted to a nonexistent address location then

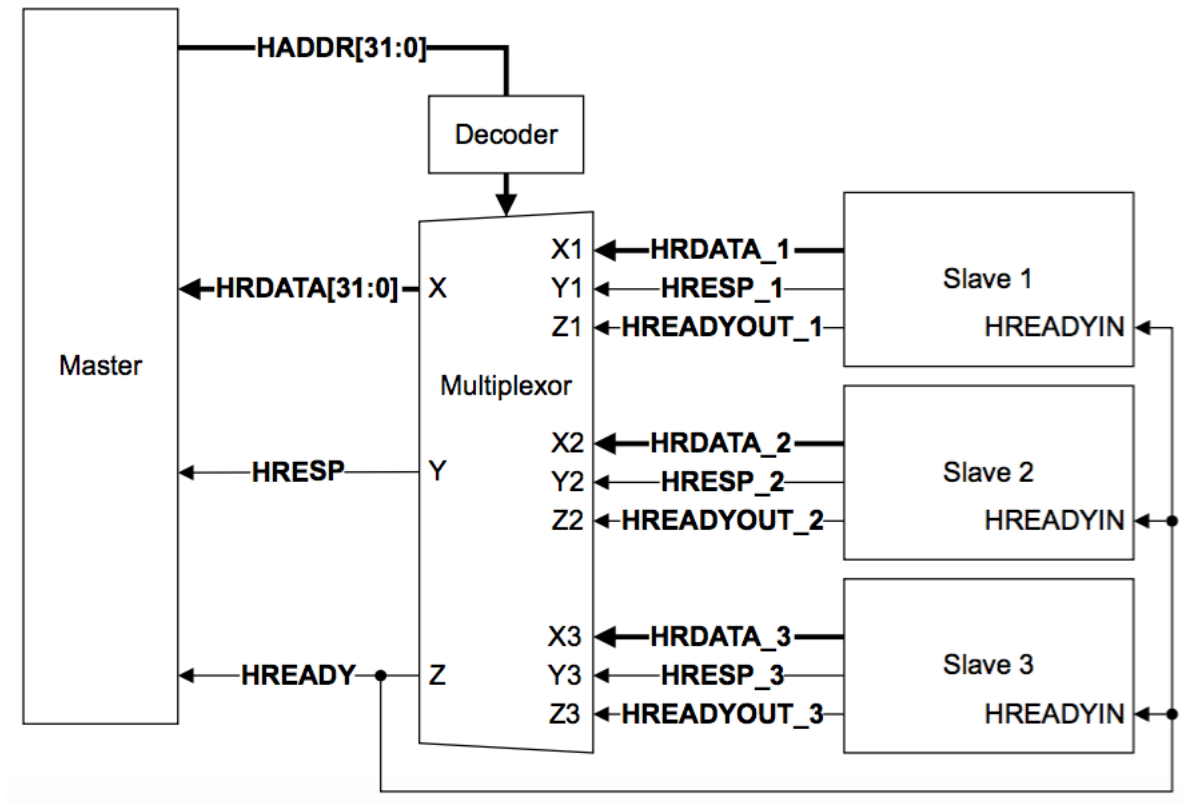


Figure 5.5: Multiplexor interconnection

the default slave provides an **ERROR** response. IDLE or BUSY transfers to nonexistent locations result in a zero wait state **OKAY** response.

5.7.2 Bus Interconnection

The AHB-Lite protocol is used with a central read data multiplexor interconnection scheme. The master drives out the address and control signals to all the slaves, with the decoder selecting the appropriate slave. Any response data from the selected slave, passes through the read data multiplexor to the master. Figure 5.5 shows the multiplexor interconnection structure required to implement an AHB-Lite design with three slaves.

5.8 Slave Response Signaling

This section describes the slave response signaling. It contains the following section:

- Slave transfer responses.

5.8.1 Slave transfer responses

After a master has started a transfer, the slave controls how the transfer progresses. A master cannot cancel a transfer after it has commenced.

A slave must provide a response that indicates the status of the transfer when it is accessed. The transfer status is provided by the **HRESP** signal. Table 5.9 lists the **HRESP** states.

Table 5.9: HRESP Signal

HRESP	Response	Description
0	OKAY	The transfer has either completed successfully or additional cycles are required for the slave to complete the request. The HREADY signal indicates whether the transfer is pending or complete.
1	ERROR	An error has occurred during the transfer. The error condition must be signaled to the master so that it is aware the transfer has been unsuccessful. A two-cycle response is required for an error condition with HREADY being asserted in the second cycle.

Table 5.9 shows that the complete transfer response is a combination of the **HRESP** and **HREADY** signals. Table 5.10 lists the complete transfer response based on the status of these two signals. This

Table 5.10: Transfer Response

HRESP	HREADY	
	0	1
0	Transfer pending	Successful transfer completed
1	ERROR response, first cycle	ERROR response, second cycle

means the slave can complete the transfer in the following three ways:

- immediately complete the transfer
- insert one or more wait states to enable time to complete the transfer
- signal an error to indicate that the transfer has failed.

These three slave transfer responses are described as:

- **Transfer done:** A successful completed transfer is signaled when **HREADY** is HIGH and **HRESP** is OKAY.
- **Transfer pending:** A typical slave uses **HREADY** to insert the appropriate number of wait states into the data phase of the transfer. The transfer then completes with **HREADY** HIGH

and an OKAY response to indicate the successful completion of the transfer.

When a slave inserts a number of wait states prior to completing the response, it must drive **HRESP** to OKAY.

- **ERROR response:** A slave uses the ERROR response to indicate some form of error condition with the associated transfer. Usually this denotes a protection error such as an attempt to write to a read-only memory location.

Although an OKAY response can be given in a single cycle, the ERROR response requires two cycles. To start the ERROR response, the slave drives **HRESP** HIGH to indicate ERROR while driving **HREADY** LOW to extend the transfer for one extra cycle. In the next cycle **HREADY** is driven HIGH to end the transfer and **HRESP** remains driven HIGH to indicate ERROR.

The two-cycle response is required because of the pipelined nature of the bus. By the time a slave starts to issue an ERROR response then the address for the following transfer has already been broadcast onto the bus. The two-cycle response provides sufficient time for the master to cancel this next access and drive **HTRANS[1:0]** to IDLE before the start of the next transfer.

Chapter 6

Integration and Implementation

This chapter gives an overview of the process of integrating and implementing the Cortex-M0+ processor with other peripheral.

Figure 6.1 shows the integration and implementation flow when you first integrate the Cortex-M0+ processor into your system and then implement your system.

Figure 6.2 shows the implementation flow.

6.1 Configuration Options

Table 6.1 shows the configuration options summary for Cortex M0+.

6.2 Key Integration Task

Following list the Cortex M0+ component level key integration task.

1. Connect the **SCLK**, **HCLK**, and **DCLK** clocks correctly.
2. Connect the **HRESETn** and **DBGRESETn** resets correctly.
3. Tie off or connect the following interface inputs appropriately:
 - External AHB-Lite interface.
 - AHB interface extensions,
 - I/O port.

Table 6.1: Cortex M0+ Option Summary

Parameter	Default Value	Supported Values	Description
ACG	1	0,1	Specifies if internal architectural clock gates are included to minimize dynamic power dissipation: 0 Exclude architectural clock gates. 1 Include architectural clock gates.
AHBSLV	1	0,1	Specifies the bus protocol implemented on the SLV port. This is a debug port. 0 The SLV port implements a Cortex-M0+ DAP specific protocol. 1 The SLV port implements a subset of AHB-Lite.
BE	0	0,1	Specifies the endianness for data transfers: 0 Little-endian. 1 Byte-invariant big-endian.
BKPT	4	0-4	Specifies the number of breakpoint unit comparators implemented.
DBG	1	0,1	Specifies whether or not the debug extensions are implemented: 0 Exclude debug functionality. 1 Include debug functional.
HWF	0	0,1	Half-word fetching only: 0 Fetch instructions using 32-bit AHB-Lite accesses whenever possible. 1 Fetch instructions using only 16-bit AHB-Lite accesses
IOP	0	0,1	I/O port: 0 Exclude I/O port functionality. 1 Include I/O port functionality.
IRQDIS	0	0,1	Disables support for individual interrupts. 32'h00000000 No IRQ disabled. 32'h0000FFFF IRQ[15:0] disabled.
MPU	0	0,8	Specifies the number of implemented Memory Protection Unit (MPU) regions: 0 Exclude MPU functionality. 8 Include MPU functionality(Eight MPU regions).
NUMIRQ	32	0-32	Specifies the highest interrupt number (NUMIRQ-1) of implemented user interrupts: 0 No functional IRQ lines 1 IRQ[0]. 2 IRQ[1:0] 32 IRQ[31:0].
RAR	0	0,1	Specifies whether all synchronous states or only architecturally required states are reset: 0 Only architecturally required state is reset. 1 All state is reset.
SMUL	0	0,1	Specifies the implemented multiplier: 0 Include the fast, single-cycle multiplier. 1 Include the small, 32-cycle multiplier.
SYST	1	0,1	Specifies whether or not the SysTick timer functionality is included: 0 Exclude the SysTick timer. 1 Include the SysTick timer.
USER	0	0,1	Unprivileged/Privileged support: 0 Exclude Unprivileged/Privileged support (that is, all accesses are Privileged). 1 Include Unprivileged/Privileged support.
VTOR	0	0,1	Vector Table Offset Register: 0 Exclude VTOR. 1 Include VTOR.
WIC	1	0,1	Specifies whether or not the WIC interface is implemented: 0 Exclude the WIC interface. 1 Include the WIC interface.
WICLINES	34	2-34	Specifies the lines supported by the WIC interface: 2 Only NMI and RXEV are supported. 3 NMI, RXEV, and IRQ[0] are supported. 4 NMI, RXEV, and IRQ[1:0] are supported 34 NMI, RXEV, and IRQ[31:0] are supported.
WPT	2	0-2	Specifies the number of watchpoint unit comparators implemented.

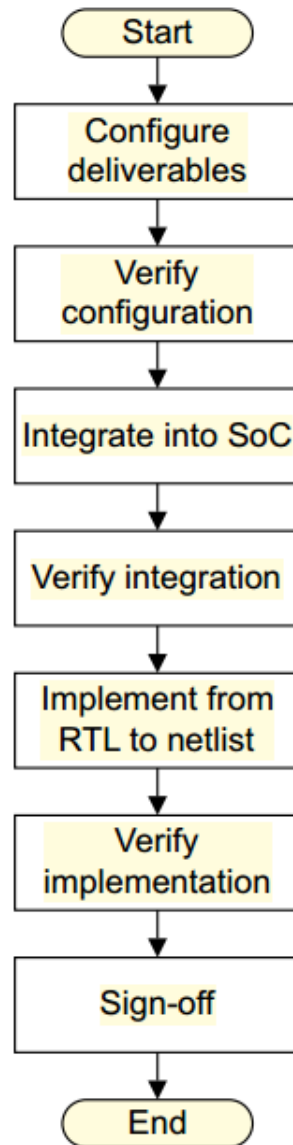


Figure 6.1: Integration and Implementation Flow

- Interrupt interface.
 - Debug slave interface.
 - Miscellaneous signals.
 - SysTick signals.
 - WIC interface.
4. Tie off the CoreSight ROM table base address.
 5. Verify your design.

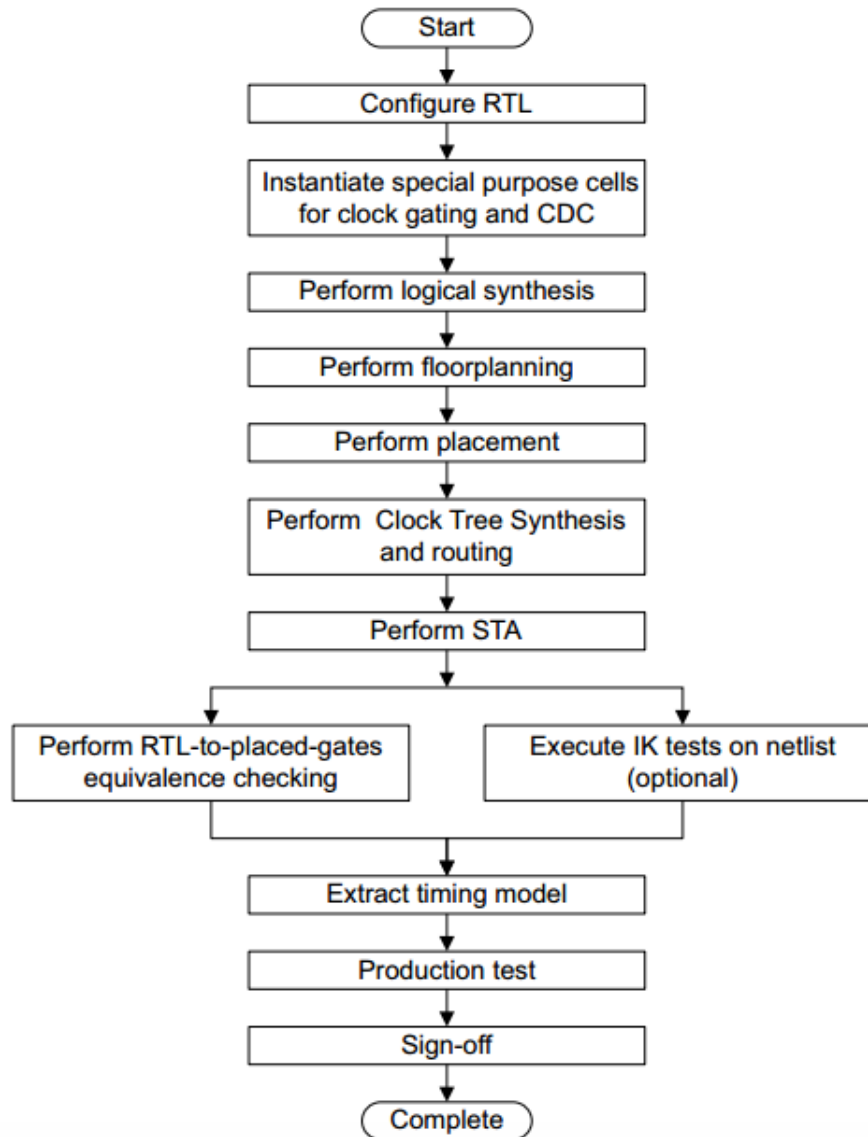


Figure 6.2: Implementation Flow

6.3 Functional Integration Guidelines

6.3.1 Clocks

Table 6.2 shows the clocks at the Cortex M0+ level of hierarchy.

6.3.2 Reset

Table 6.3 shows the resets at the Cortex M0+ level of hierarchy.

Table 6.2: Cortex M0+ Level Clocks

Name	Direction	Description	Connection Information
SCLK	Input	Free running clock that clocks a small amount of logic in the processor system domain.	SCLK must always be running unless the processor is inWIC-mode deep sleep and no debugger is connected.
HCLK	Input	Clock for the majority of the non-debug logic in the processor system domain.	HCLK must be derived directly from SCLK. Connect HCLK to the AHB layer that the processor is connected to.
DCLK	Input	Clock for the processor debug domain.	DCLK must be derived directly from SCLK. DCLK must always be driven while a debugger is connected. It can be gated when no debugger is connected.

Table 6.3: Cortex M0+ Level Reset

Name	Direction	Description	Connection Information
HRESETn	Input	Reset for the processor system domain and the AHB system	Deassert HRESETn synchronously to SCLK. Assert HRESETn on power-on. Assert HRESETn for at least two HCLK cycles.
DBGRESETn	Input	Reset for the processor debug domain	Deassert DBGRESETn synchronously to SCLK. Assert DBGRESETn on power-on. Assert DBGRESETn for at least two DCLK cycles. Tie DBGRESETn LOW when no debugger is connected.

6.3.3 Interface

This section describes the interface of Cortex M0+ processor to AHB-Lite Interface.

Understanding of AMBA AHB-Lite bus interface signals is must as described in chapter 5 AMBA AHB-Lite Protocol Specification.

Table 6.4 shows the AHB-Lite interface.

Table 6.4: AHB-Lite Signals

Name	Direction	Connection Information
HADDR[31:0]	Output	Connect to address decoders, arbiter, and slaves through the bus infrastructure.
HBURST[2:0]	Output	Connect to the AHB arbiter and slaves through the bus infrastructure.
HPROT[3:0]	Output	Connect to the slaves through the bus infrastructure.
HSIZE[2:0]	Output	Connect to the slaves through the bus infrastructure.
HTRANS[1:0]	Output	Connect to the AHB arbiter and slaves through the bus infrastructure.
HWRITE	Output	Connect to the slaves through the bus infrastructure.
HMASTLOCK	Output	
HWDATA[31:0]	Output	
HRDATA[31:0]	Input	
HREADY	Input	
HRESP	Input	

6.4 Key Implementation Points

This section contains a list of the main points to consider when you implement the Cortex-M0+ processor.

Following lists the key tasks for Implementation

1. Select top level of hierarchy to implement.
2. Configure the processor parameters.
3. Select appropriate library cells for clock gating and *Clock-Domain Crossing* (CDC) purposes.
4. If you require SRPG, ensure that the implementation level includes pins for power, retention and isolation control.
5. If you require SRPG, select appropriate UPF or CPF file and library cells for power gating.
6. Perform synthesis.
7. Determine optimum floorplan
8. Perform place and route
9. Perform LVS and DRC checks.
10. Perform timing verification.
11. Perform characterization.
12. Run DFT.
13. Perform formal verification using logical equivalence checking tools.
14. Optionally run the tests on the netlist with SDF annotation.
15. Perform sign-off in accordance with the agreed criteria and your sign-off obligations.
16. Sign-off your implementation.

6.5 SoC Development Results

Understanding of ARM Cortex M0+ and AMBA AHB-Lite protocol specification is important as described in Chapter 4 and Chapter 5 respectively. The processor must be configured according to the specification of the SoC. RTL code for all the peripherals which are to be integrated with the processor

must be functionally verified and free from violations. All the peripheral must be allocated with required memory and should response only to those addresses assigned to it.

A top wrapper must accept the transaction from the master when selected and the slave should response accordingly for read or write operation with the size of data coming in or going out. This top wrapper should generate **HRESP** and **HREADY** signals for the master to know about it's status and further process.

Figure 6.3 shows the integration of AMBA AHB-Lite system with ARM Cortex M0+. This is the first integration step towards development of SoC.

Figure 6.4 shows the integration of AHB-Lite with peripheral. This is the second step towards development of SoC.

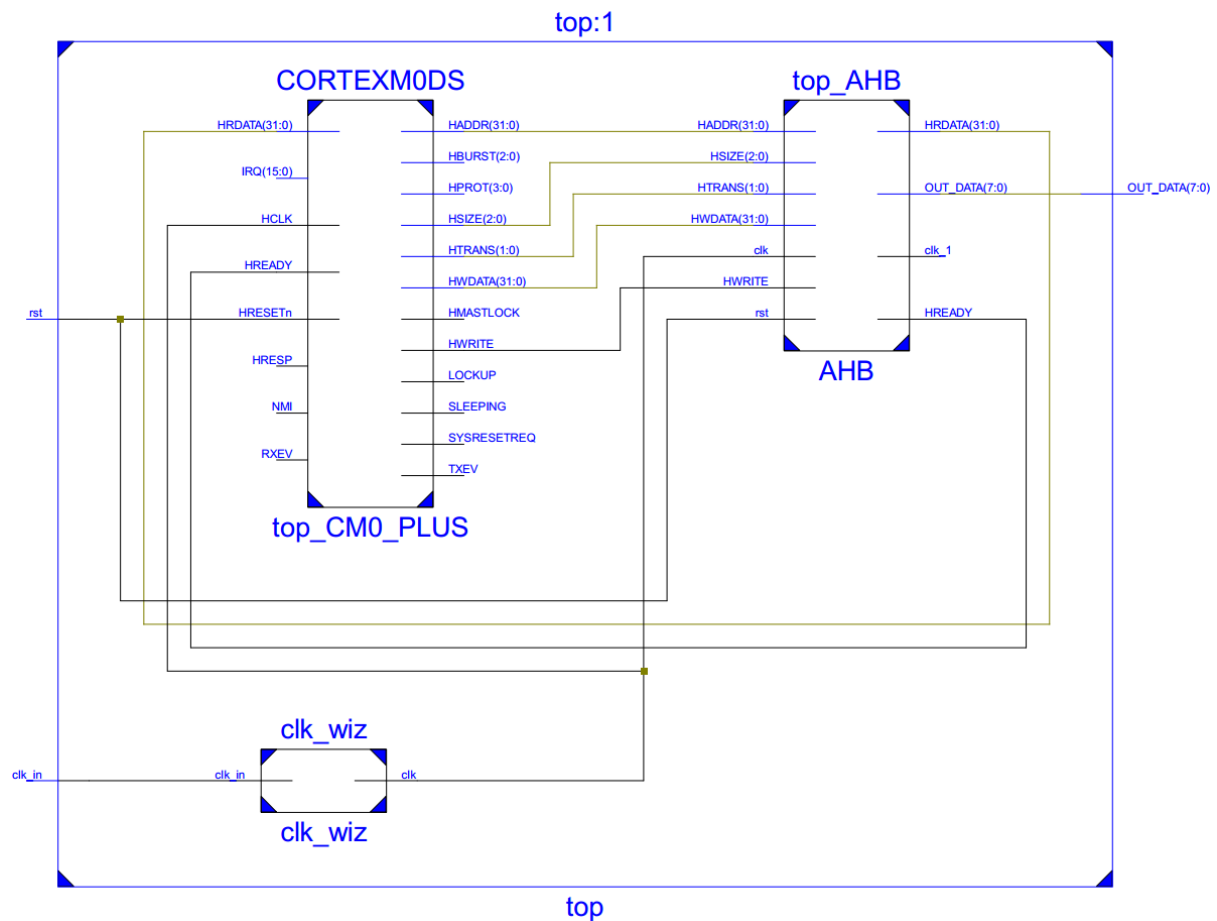


Figure 6.3: Integration of AHB-Lite with ARM Cortex M0+

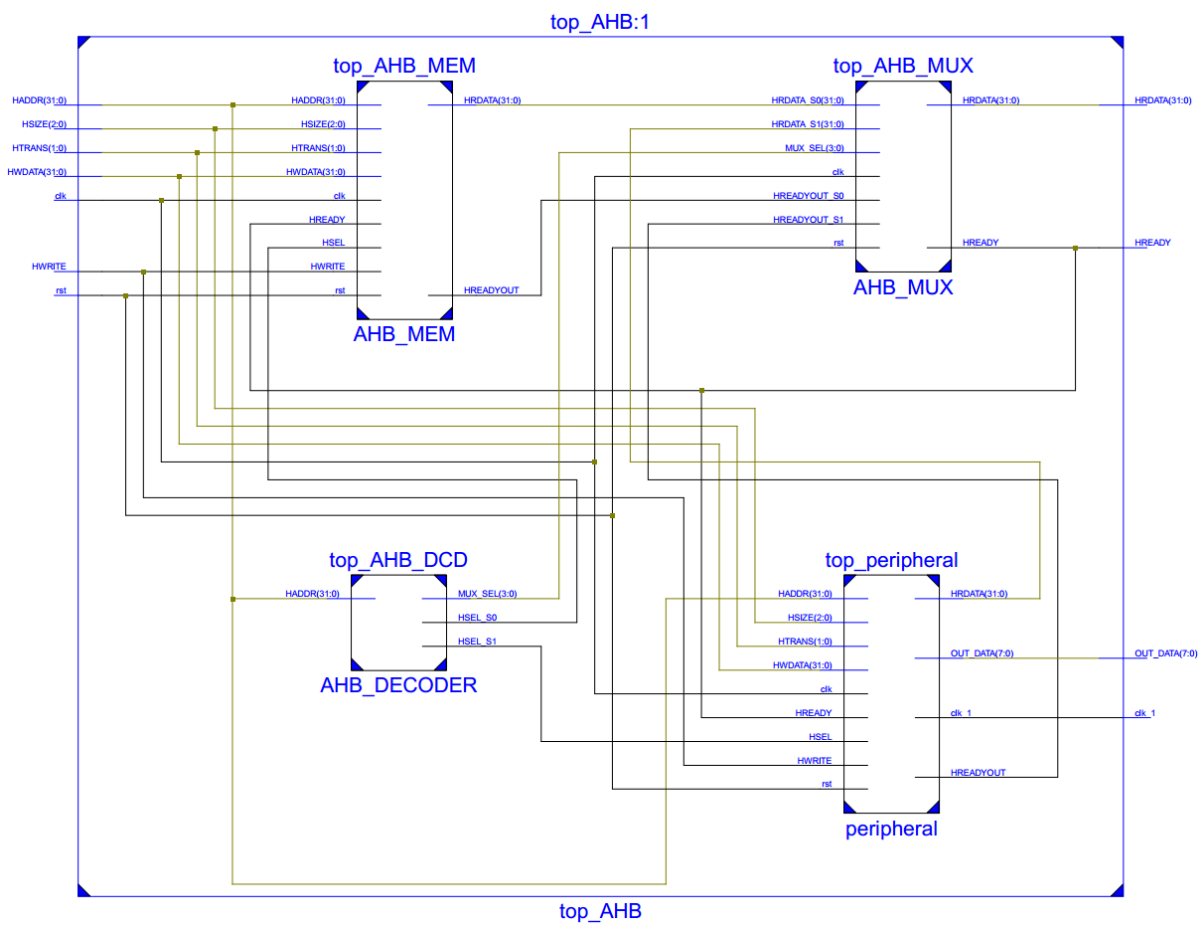


Figure 6.4: Integration of AHB-Lite interface with peripheral

Chapter 7

Results & Discussion

The EEG signals database obtained from Physionet [36], Klinik für Epileptologie Universität Bonn, Germany [37] consisting of five sets from healthy volunteer and epileptic patient with different activities, Caltech [38] and University of Southampton were recorded at sampling frequency of $256Hz$, $173.6Hz$, $160Hz$ and $500Hz$ respectively. A healthy volunteer (patient without any sufferings) EEG data of 10 sec was mixed with EOG and Muscular artefact manually in specific pattern mentioned in the cases below (Table 7.1). This mixed data was used in the initial analysis to prove the functionality and performance of the algorithm on hardware platform in the following manner depicted in Table 7.1. It is to be noted that each case has been validated against 100 EEG signals taken from the above mentioned databases. Case I (Table 7.1) consists of seven signals, which were left clean and hence a high value of Correlation Coefficient as expected, is experimentally determined along with Regression. Case II and Case III (Table 7.1), muscular artefacts are detected and removed. In case IV (Table 7.1), only blink artefacts were manually added and removed. Parameter ‘ \mathbf{x} ’ was varied to determine the optimum number of samples per frame for muscle artefact removal. From Table 7.2, the highest value of correlation, regression and R-square for Case II & III are observed at 86 samples per frame. Similarly from Table 7.2, for Case IV, if ‘only GM’ value is used as threshold, then high performance is expected. These optimised values are used in cases V, VI, VII, VIII & IX, where both the artefacts were added in combination of alternate and random manner as shown in Table 7.1 and corresponding correlation and regression values are observed. Result of the mixed artefact case (Case IX) is shown in Fig. 7.1. In Case X and XI (Table 7.1), the EEG signals of various subjects sampled at $256Hz$ (Physionet [36]), $160Hz$ (Caltech [38]) and $500Hz$ (University of Southampton) were observed for 10sec and optimized condition for muscle and blink artefact was used in the experiment. No external artefacts are added in case X and XI, since these

Table 7.1: Performance Metrics for different cases of artefact addition and real data simulation on hardware platform (**FPGA**)

Case	artefacts		Performance Metrics		
	Muscular artefacts	Blink artefacts	Average Correlation	Average Regression	Average R-Square
Case I	NO	NO	0.9416	0.8124	0.7941
Case II	Alternate	NO	0.7078	0.7300	0.6956
Case III	Random	NO	0.6275	0.7828	0.7043
Case IV	NO	Yes	0.8894	0.4667	0.6278
Case V	Random	Alternate	0.5649	0.6555	0.5545
Case VI	Alternate	Alternate (same frame)	0.9070	0.8736	0.8923
Case VII	Alternate	Alternate (Different frame)	0.8956	0.8243	0.8565
Case VIII	Alternate	Random	0.8725	0.8203	0.8427
Case IX	Random	Random	0.9357	0.8963	0.8897
Case X (Real Data)	Unknown Position	Unknown Position	0.8307	0.6680	0.7284
	Unknown Position	Unknown Position	0.7350	0.7791	0.7562
Case XI	Unknown Position	Unknown Position	0.8440	0.9326	0.5148

signals were already mixed with artefact to prove the effectiveness of the algorithm unlike case II to IX. The signal is detected with artefacts and removed as shown in Fig. 7.2, 7.3 & 7.4. Table 7.3 is the comparison of the proposed method with other recent methods.

The RRMSE, standard deviation, variance and mean error are higher than the other methods proposed. The other performance metrics like average coefficient correlation is higher than the other methods indicating an accurate and higher artefact removal procedure. The NMSE of the simulated and experimental EEG data was calculated for each of the noisy channel (having artefacts) and noise free channels (having no artefacts). The SNR of the proposed work was found to be $-18.54dB$ and hence in the simulated NMSE channel 15 and 18 were considered. A low value of NMSE indicates that the system is performing well in the noisy channel of the EEG dataset. The proposed methodology is capable of removing not only the low frequency blink artefact but also the high frequency muscle artefact in comparison with [11, 22]. It can be noticed from Table 7.3 that the performance of proposed methodology is 7% less in terms of correlation compared with [22]. However, such deviation is mainly attributed due to our proposed low-complex hardware design methodology compared to the software centric of approach of [22] where the hardware complexity and computational delay of the proposed methodology are 64.28% and 53.58% less compared to [22] making it favourable for the real time hardware design for NDD and

Table 7.2: Performance Comparison by varying ‘ \mathbf{x} ’ in Muscle Artefact and Blink Artefact Cases of **LM** & **GM** for alternate and random addition of artefact

Cases	Subcases	Average Correlation	Average Regression	Average R-Square
Case I	Blink alternate (only GM)	0.9796632	0.963957	0.8984671
	Blink alternate (LM & GM)	0.9753417	0.952481	0.8695557
	Blink Random (Only GM)	0.977621	0.948206	0.8910231
	Blink Random (LM & GM)	0.8661372	0.86631	0.7289345
Case II	Muscle alternate (4 samples/ frame)	0.4570492	0.752812	0.1924414
	Muscle alternate (10 samples/frame)	0.6455181	0.7207	0.3805284
	Muscle alternate (20 samples/frame)	0.7680168	0.758661	0.5112613
	Muscle alternate (33 samples/frame)	0.8089984	0.76075	0.5225469
	Muscle alternate (43 samples/frame)	0.8004564	0.73833	0.525052
	Muscle alternate (66 samples/frame)	0.8356125	0.764186	0.5249669
	Muscle alternate (86 samples/frame)	0.8719209	0.820555	0.6102836
	Muscle alternate (107 samples/frame)	0.8677898	0.795259	0.5460391
	Muscle alternate (122 samples/frame)	0.8341224	0.739985	0.4909559
	Muscle alternate (170 samples/frame)	0.8611689	0.774721	0.52037
Case III	Muscle random (4 samples/ frame)	0.5110784	0.815195	0.3140283
	Muscle random (10 samples/ frame)	0.726977	0.801797	0.4835464
	Muscle random (20 samples/frame)	0.8232031	0.809822	0.5858392
	Muscle random (33 samples/frame)	0.848105	0.81383	0.6078609
	Muscle random (43 samples/frame)	0.8669584	0.83035	0.6331701
	Muscle random (66 samples/frame)	0.8597717	0.7897	0.589286
	Muscle random (86 samples/frame)	0.8719209	0.820555	0.6102836
	Muscle random (107 samples/frame)	0.8563104	0.77265	0.5657873
	Muscle random (122 samples/frame)	0.8693699	0.791556	0.5873208
	Muscle random (170 samples/frame)	0.8651133	0.820209	0.6109856

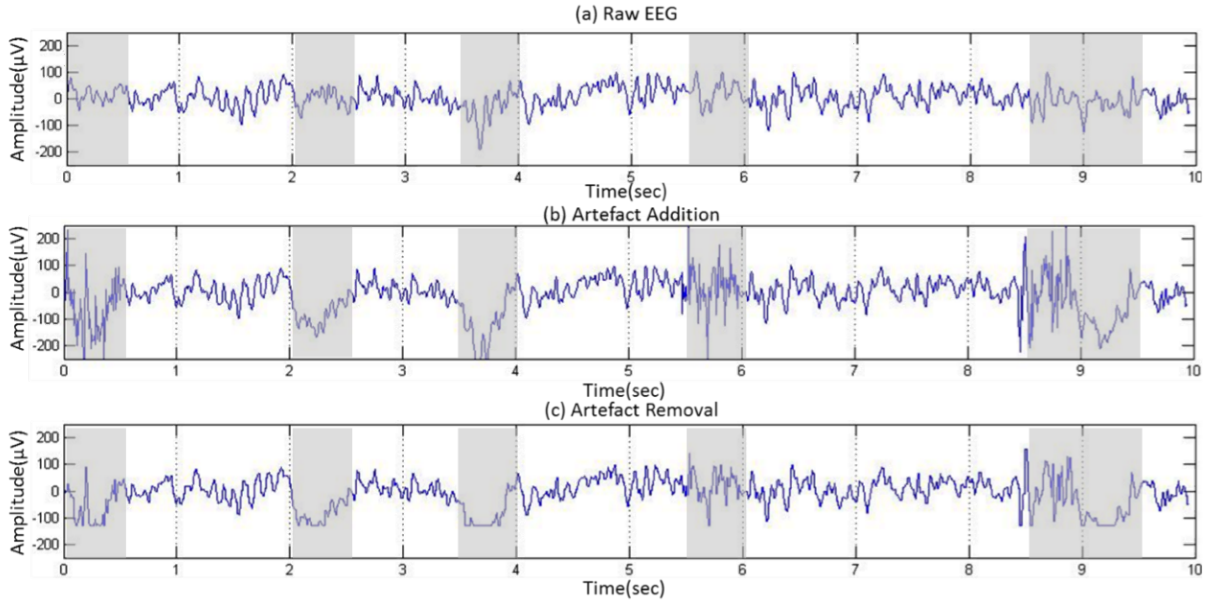


Figure 7.1: Mixed artefact signal Analysis. The shaded area (light grey) shows the position where artefacts are manually added and removed. **(A)** Amplitude vs Time plot for 10 second of raw clean EEG signal. **(B)** Muscle and Blink artefacts are added in the signal **(A)** in a random fashion. **(C)** Signal obtained after artefacts removal when the optimized conditions are applied.

BCI. The detailed complexity analysis of our proposed approach will be given later in this section. Fig 7.2, 7.3 & 7.4 shows 21 channel real EEG dataset from [27] and University of Southampton related to eye and head movement respectively. The waveform presented in Fig. 7.2, 7.3 & 7.4 shows the favourable comparison between the results of MATLAB simulation and FPGA prototyping. The percentage error on-hardware (FPGA) is found to be 9.5% for Fig. 7.2, 7.3 & 7.4. The hardware complexity of the proposed methodology is carried out in terms of the operations involved like adders, subtractors, multipliers, multiplexers and comparators. For calculating the hardware complexity in terms of the number of logic gates and transistor count we make the following assumptions from [42].

Assumption 1:- i) One n -bit adder and subtracter needs n full adders and full subtracters. ii) One n by n multiplier needs $n(n-2)$ Full Adder, n Half-adders and n^2 AND gates. iii) One n -bit comparator requires cascading of n number of 1-bit comparators which consists of 2 NOT gates, 2 AND gates and 1 NOR gate each. iv) An n -bit 2:1 MUX requires n number of 2:1 MUX consisting of 4 NAND gates each. v) One n -bit shifters requires n number of D flip-flops consisting of 4 NAND gates and 1 NOT gate each.

Assumption 2:- Transistor count for each of the blocks are as follows. i) 1-bit full adder has 24 transistors. ii) 1-bit half adder has 12. iii) 1-bit Full Subtracters has 28. iv) One 2-input AND gate has 6. v) One 2-input NAND gate has 4. vi) One NOT gate has 2. Fig 5(A) shows the bar graph

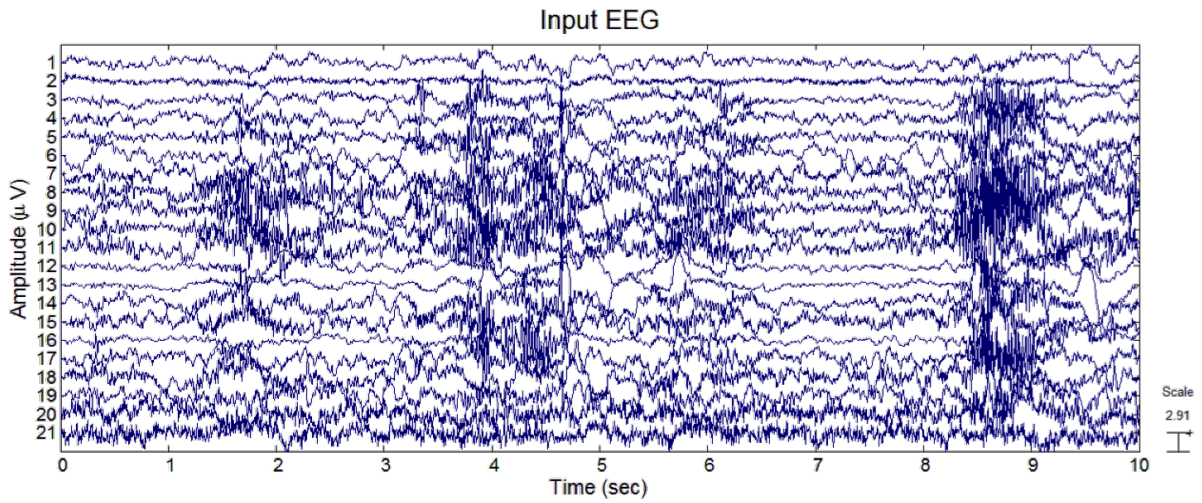


Figure 7.2: Real EEG Signal Analysis (21 Channel): Input to the System

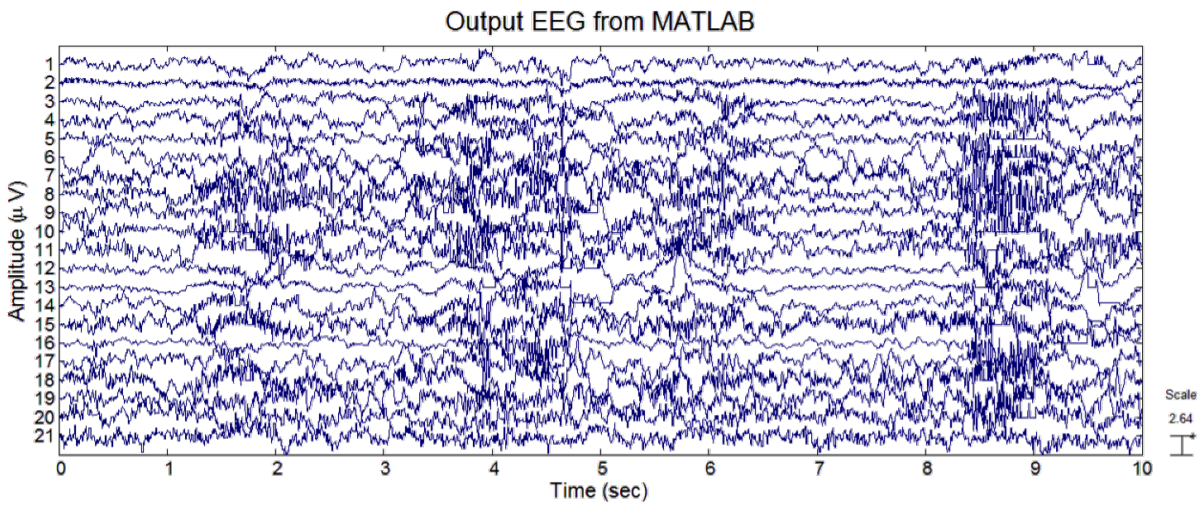


Figure 7.3: Real EEG Signal Analysis (21 Channel): Output from MATLAB Simulation

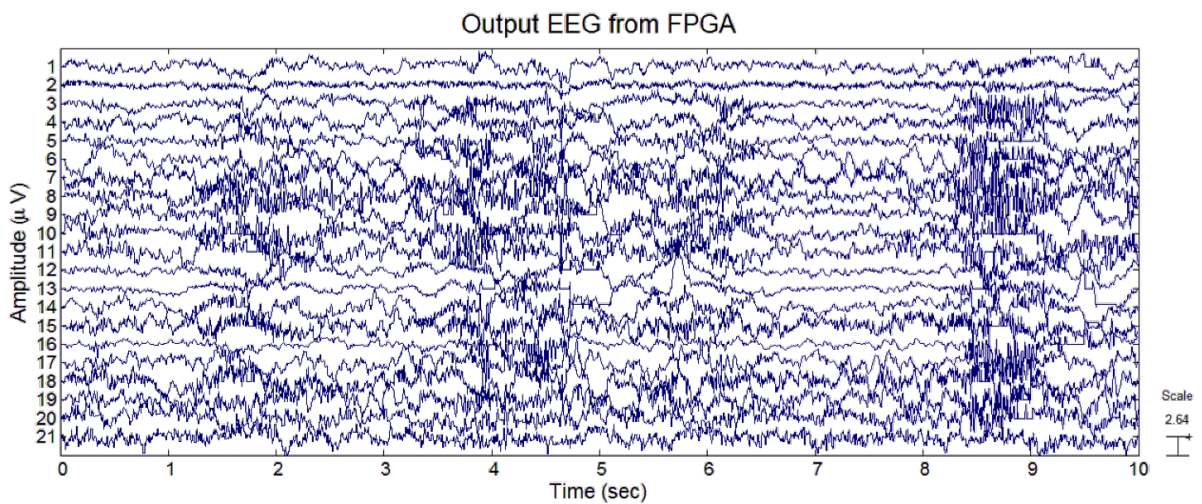


Figure 7.4: Real EEG Signal Analysis (21 Channel): Output from the FPGA Implementation

Table 7.3: Comparison with different State-of-the-art methods

Parameter	[Reference][Value]		This Work
RRMSE	[38] BSS_CCA=0.11; ICA(JADE)=0.25	[10] 0.31 to 0.42	0.4437
Normalised Correlation Coefficient	[39] ICA-RLS (Segment B) = 0.6264	[14] 0.796, [15] 0.76 [29] Zeroing ICA=0.5767, wICA=0.5817, [22] 0.863 to 0.956, [23] 0.776, [40] 0.77, [41] 0.755 to 0.833	0.8307
Standard Deviation	[40] 40.98	[11] 16.28 [23] 11	54.1524
Variance	[40] 1679.6	-	2932.5
Mean Absolute Error	[40] 28.07	-	41.3756
NMSE (simulation)	[31] <u>Noisy Channel</u> Ch-15(SNR=-15dB)=0.355 Ch-18(SNR)=-20dB=0.3140	-	<u>Noisy Channel</u> (SNR = -18.52dB)=0.3255
	<u>Noise Free Channel</u> Ch-15 = 0.9200 Ch-18 = 0.9755		<u>Noise Free Channel</u> 0.9623
NMSE (experimental)	[31] <u>Noisy Channel</u> Ch-15 = 0.1023 Ch-23 = 0.1604	-	<u>Noisy Channel</u> 0.2365
	<u>Noise Free Channel</u> Ch-15 = 0.8694 Ch-23 = 0.8280		<u>Noise Free Channel</u> 0.8759

Table 7.4: FPGA Resource Utilization

Logic Utilization	Used	Available	Utilization
Number of Slice Register	571	18224	3.13%
Number of Slice LUTs	933	9112	10.23%
Number of Fully used LUT-FF Pairs	433	489	88.54%
Number of Block RAMS/FIFO	8	32	25%
Number of BUFG/BUFGCTRLs	8	16	50%

for transistor count by varying the word-length (n), comparing the proposed methodology with other methods [11, 22] has indicated a lower transistor count thus implies low complex procedure. The system proposed here is designed for $n = 16$ and corresponding transistor count is 44,544. The hardware delay for arithmetic and logical block has been calculated taking into account the delay of the basic building block like NAND gate. Following assumptions are made as per [42].

Assumption 1:- i) No interconnect delay. Denoting the delay of two-input NAND gate to be Δ units, the delay of other blocks are calculated as ii) An n -bit two input full adder and full subtracter has $2n\Delta$ units delay. iii) $n - by - n$ multiplier has $8n\Delta$ units delay. iv) An $n - bit$ comparator has $9n\Delta$. v) An $n - bit$ 2 : 1 MUX has $5n\Delta$. vi) An $n - bit$ shifter has $4n\Delta$.

Figure 7.5 & 7.6 shows the hardware complexity and delay respectively of the proposed methodology with respect to the variation in the word-length (n) in comparison with other methods in [11, 22]. The graph indicates a lower complexity and delay for the computation of the proposed method by 64.28% and 53.58% respectively compared with [22] and 33% and 25.8% respectively compared with [11] for word length of 16 - bits. The proposed methodology has been designed and proved on Xilinx Spartan 6 FPGA board. The inputs to design under test (DUT) were given through a Block RAM created on FPGA and outputs were observed on monitor using ChipScopePro tool from Xilinx. Table 7.4 shows the estimated device utilisation summary of the hardware.

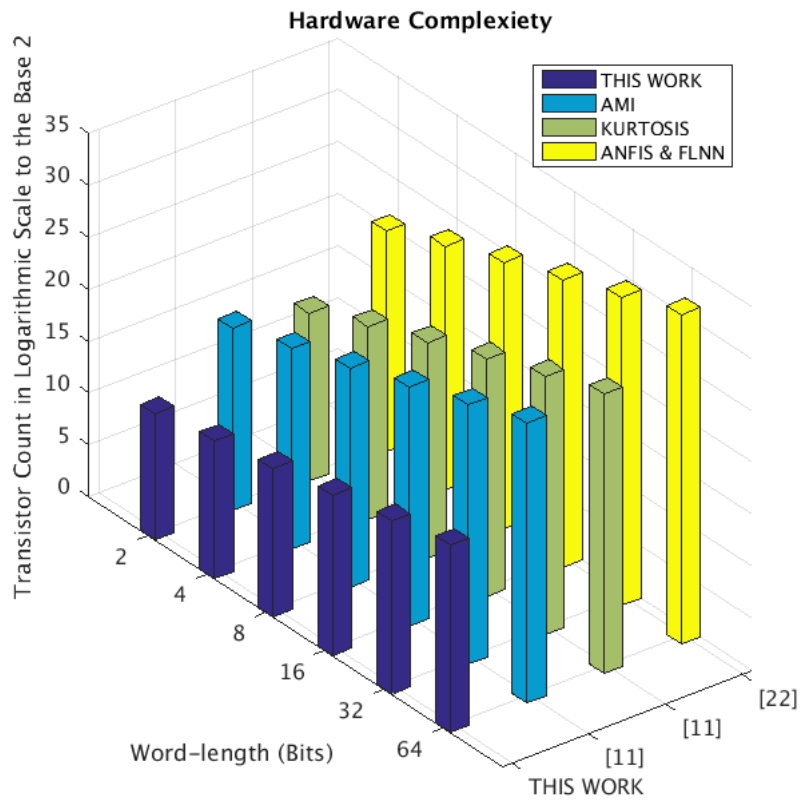


Figure 7.5: Variation of Hardware Complexity in terms of Transistor Count with different word-length(n)

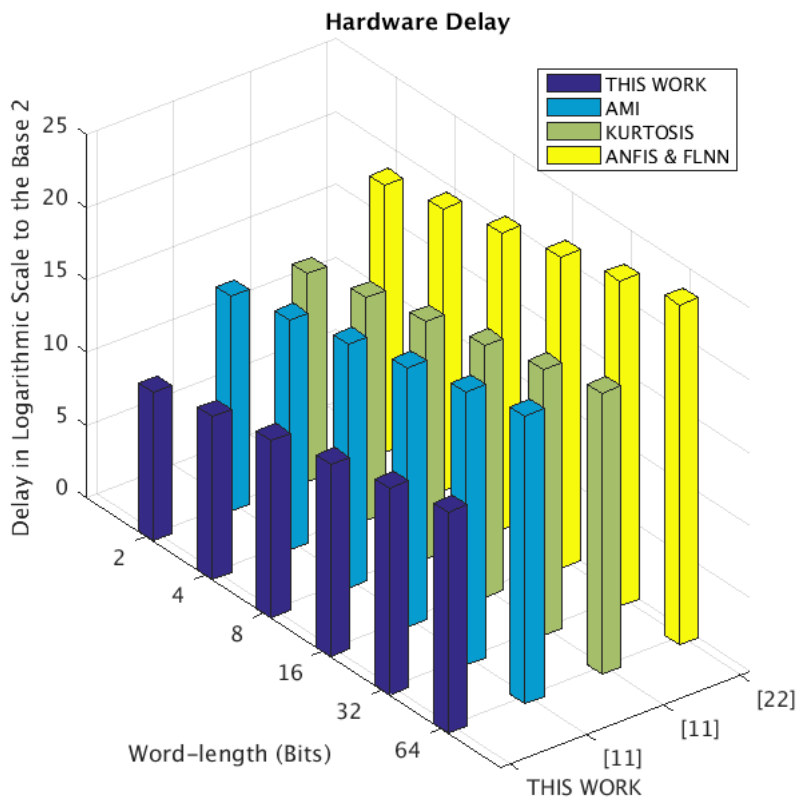


Figure 7.6: Variation of Hardware Delay with different word-length(n)

Chapter 8

Conclusion

The automated methodology proposed in this paper, unlike the state of the art methods, can remove blink and muscular artefacts without the need of any extra electrode. Its reliability and robustness is also established after exhaustive simulation study and analysis on both simulated and real data. The performance of the proposed methodology is measured in terms of correlation, regression and R-square statistics and it has been found that their average values lie above 80%, 75% and 76% respectively. Comparison of the simulation results and FPGA prototyping shows an error of about 9.5%. The total power consumption of the proposed methodology is about $76\mu\text{W}$. The satisfactory hardware results are also obtained when prototyped on FPGA platform, which shows the capability of the proposed methodology to be translated into a system on chip. We believe the proposed methodology would be useful in next generation pervasive healthcare for BCI and NDD diagnosis and treatment.

References

- [1] M. S. Gupta. Neurodevelopmental Disorders in Children Autism and ADHD. *Environmental Chemistry. com. April* 14.
- [2] E. Milne, A. Scope, O. Pascalis, D. Buckley, and S. Makeig. Independent component analysis reveals atypical electroencephalographic activity during visual perception in individuals with autism. *Biological psychiatry* 65, (2009) 22–30.
- [3] S. E. Levy, D. S. Mandell, and R. T. Schultz. Autism. *The Lancet* 374, (2009) 1627 – 1638.
- [4] M. Wadman. Autism’s fight for facts: A voice for science. *Nature* 28–31.
- [5] H. L. Needleman, A. Schell, D. Bellinger, A. Leviton, and E. N. Allred. The long-term effects of exposure to low doses of lead in childhood: an 11-year follow-up report. *New England journal of medicine* 322, (1990) 83–88.
- [6] D. L. Santesso, I. E. Drmic, M. K. Jetha, S. E. Bryson, J. O. Goldberg, G. B. Hall, K. J. Mathewson, S. J. Segalowitz, and L. A. Schmidt. An event-related source localization study of response monitoring and social impairments in autism spectrum disorder. *Psychophysiology* 48, (2011) 241–251.
- [7] T. Pistorius, C. Aldrich, L. Auret, and J. Pineda. Early Detection of risk of autism spectrum disorder based on recurrence quantification analysis of electroencephalographic signals. In Neural Engineering (NER), 2013 6th International IEEE/EMBS Conference on. IEEE, 2013 198–201.
- [8] S. Rondeau. Electroencephalogram use in Autistic Disorder Assessment. *Naturopathic Doctor News & Review* .
- [9] (2012). EEG Connectivity and Autism: Methodological and Clinical Features. *Psychophysiology* 49, (2012) S8.
- [10] R. Patel, S. Sengottuvel, M. Janawadkar, K. Gireesan, T. Radhakrishnan, and N. Mariyappa. Ocular artifact suppression from EEG using ensemble empirical mode decomposition with principal component analysis. *Computers & Electrical Engineering* .

- [11] I. Daly, R. Scherer, M. Billinger, and G. Müller-Putz. FORCE: Fully Online and automated artifact Removal for brain-Computer interfacing. *IEEE transactions on neural systems and rehabilitation engineering* 23, (2015) 725–736.
- [12] L. Frølich, T. S. Andersen, and M. Mørup. Classification of independent components of EEG into multiple artifact classes. *Psychophysiology* 52, (2015) 32–45.
- [13] T.-P. Jung, C. Humphries, T.-W. Lee, S. Makeig, M. J. McKeown, V. Iragui, and T. J. Sejnowski. Removing electroencephalographic artifacts: comparison between ICA and PCA. In *Neural Networks for Signal Processing VIII, 1998. Proceedings of the 1998 IEEE Signal Processing Society Workshop*. IEEE, 1998 63–72.
- [14] M. R. Mowla, S.-C. Ng, M. S. Zilany, and R. Paramesran. Artifacts-matched blind source separation and wavelet transform for multichannel EEG denoising. *Biomedical Signal Processing and Control* 22, (2015) 111–118.
- [15] W. Zhou and J. Gotman. Automatic removal of eye movement artifacts from the EEG using ICA and the dipole model. *Progress in Natural Science* 19, (2009) 1165–1170.
- [16] V. Krishnaveni, S. Jayaraman, S. Aravind, V. Hariharasudhan, and K. Ramadoss. Automatic identification and removal of ocular artifacts from EEG using wavelet transform. *Measurement science review* 6, (2006) 45–57.
- [17] M. Mennes, H. Wouters, B. Vanrumste, L. Lagae, and P. Stiers. Validation of ICA as a tool to remove eye movement artifacts from EEG/ERP. *Psychophysiology* 47, (2010) 1142–1150.
- [18] C. A. Joyce, I. F. Gorodnitsky, and M. Kutas. Automatic removal of eye movement and blink artifacts from EEG data using blind component separation. *Psychophysiology* 41, (2004) 313–325.
- [19] C. Burger and D. J. van den Heever. Removal of EOG artefacts by combining wavelet neural network and independent component analysis. *Biomedical Signal Processing and Control* 15, (2015) 67–79.
- [20] D. R. Achanccaray and M. A. Meggiolaro. Detection of artifacts from EEG data using wavelet transform, high-order statistics and neural networks. In *XVII Brazilian Conference on Automatica*. 2008 23.
- [21] K. T. Sweeney, S. F. McLoone, and T. E. Ward. The use of ensemble empirical mode decomposition with canonical correlation analysis as a novel artifact removal technique. *IEEE transactions on biomedical engineering* 60, (2013) 97–105.
- [22] J. Hu, C.-s. Wang, M. Wu, Y.-x. Du, Y. He, and J. She. Removal of EOG and EMG artifacts from EEG using combination of functional link neural network and adaptive neural fuzzy inference system. *Neurocomputing* 151, (2015) 278–287.

- [23] W.-D. Chang, H.-S. Cha, K. Kim, and C.-H. Im. Detection of eye blink artifacts from single prefrontal channel electroencephalogram. *Computer methods and programs in biomedicine* 124, (2016) 19–30.
- [24] B. Mijovic, M. De Vos, I. Gligorijevic, J. Taelman, and S. Van Huffel. Source separation from single-channel recordings by combining empirical-mode decomposition and independent component analysis. *IEEE transactions on biomedical engineering* 57, (2010) 2188–2196.
- [25] D. J. McFarland, A. T. Lefkowitz, and J. R. Wolpaw. Design and operation of an EEG-based brain-computer interface with digital signal processing technology. *Behavior Research Methods, Instruments, & Computers* 29, (1997) 337–345.
- [26] D. Safieddine, A. Kachenoura, L. Albera, G. Birot, A. Karfoul, A. Pasnicu, A. Biraben, F. Wendling, L. Senhadji, and I. Merlet. Removal of muscle artifact from EEG data: comparison between stochastic (ICA and CCA) and deterministic (EMD and wavelet-based) approaches. *EURASIP Journal on Advances in Signal Processing* 2012, (2012) 1–15.
- [27] M. A. Sovierzoski, F. I. Argoud, and F. M. de Azevedo. Identifying eye blinks in EEG signal analysis. In 2008 International Conference on Information Technology and Applications in Biomedicine. IEEE, 2008 406–409.
- [28] B. Nouredin, P. D. Lawrence, and G. E. Birch. Effects of task and EEG-based reference signal on performance of on-line ocular artifact removal from real EEG. In 2009 4th International IEEE/EMBS Conference on Neural Engineering. IEEE, 2009 614–617.
- [29] C. Guerrero-Mosquera and A. Navia-Vazquez. Automatic removal of ocular artefacts using adaptive filtering and independent component analysis for electroencephalogram data. *IET signal processing* 6, (2012) 99–106.
- [30] M. Anastasiadou, A. Hadjipapas, M. Christodoulakis, E. S. Papathanasiou, S. S. Papacostas, and G. D. Mitsis. Detection and removal of muscle artifacts from scalp EEG recordings in patients with epilepsy. In Bioinformatics and Bioengineering (BIBE), 2014 IEEE International Conference on. IEEE, 2014 291–296.
- [31] M. N. Tibdewal, R. Fate, M. Mahadevappa, and A. Ray. Detection and classification of Eye Blink Artifact in electroencephalogram through Discrete Wavelet Transform and Neural Network. In Pervasive Computing (ICPC), 2015 International Conference on. IEEE, 2015 1–6.
- [32] A. Acharyya, K. Maharatna, B. M. Al-Hashimi, and J. Reeve. Coordinate rotation based low complexity ND FastICA algorithm and architecture. *IEEE Transactions on Signal Processing* 59, (2011) 3997–4011.

- [33] J. S. Walker. A primer on wavelets and their scientific applications. CRC press, 2008.
- [34] J. Joy, S. Peter, and N. John. Denoising using soft thresholding. *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering* 2, (2013) 1027–1032.
- [35] P. Jadhav, D. Shanamugan, A. Chourasia, A. Ghole, A. Acharyya, and G. Naik. Automated detection and correction of eye blink and muscular artefacts in EEG signal for analysis of Autism Spectrum Disorder. In 2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE, 2014 1881–1884.
- [36] P. PhysioBank. PhysioNet: Components of a New Research Resource for Complex Physiologic Signals [Circulation Electronic Pages]. *Circulation* 101, (2000) e215–e220.
- [37] R. G. Andrzejak, K. Lehnertz, F. Mormann, C. Rieke, P. David, and C. E. Elger. Indications of nonlinear deterministic and finite-dimensional structures in time series of brain electrical activity: Dependence on recording region and brain state. *Physical Review E* 64, (2001) 061,907.
- [38] <http://www.vis.caltech.edu/~rodri/data.htm> .
- [39] W. De Clercq, A. Vergult, B. Vanrumste, W. Van Paesschen, and S. Van Huffel. Canonical correlation analysis applied to remove muscle artifacts from the electroencephalogram. *IEEE transactions on Biomedical Engineering* 53, (2006) 2583–2587.
- [40] R. Mahajan and B. I. Morshed. Unsupervised eye blink artifact denoising of EEG data with modified multiscale sample entropy, kurtosis, and Wavelet-ICA. *IEEE journal of biomedical and health informatics* 19, (2015) 158–165.
- [41] S. ORegan, S. Faul, and W. Marnane. Automatic detection of EEG artefacts arising from head movements using EEG and gyroscope signals. *Medical engineering & physics* 35, (2013) 867–874.
- [42] A. Acharyya, K. Maharatna, and B. M. Al-Hashimi. Algorithm and architecture for nD vector cross-product computation. *IEEE Transactions on Signal Processing* 59, (2011) 812–826.