



*Mind Music*

**A Report on the Mind-Modulated Music Interface  
(MMMI)**

**Ben Swift**

**A thesis submitted for the degree of  
Bachelor of Science with Honours in Computer Science of  
The Australian National University**



---

# Declaration

---

This thesis is an account of research undertaken between February 2007 and October 2007 at The Department of Computer Science, Faculty of Engineering and Information Technology, The Australian National University, Canberra, Australia.

Except where acknowledged in the customary manner, the material presented in this thesis is my own. To the best of my knowledge, it is original research. It has not been previously submitted in whole or in part for credit for any course at any university.

---

Ben Swift  
November 2007



---

# Thanks

---

I would like to thank my family, for all their love and support. This would not have been possible without them.

I would also like to thank my supervisor, Henry, for his encouragement and advice over the past 18 months. He has provided the carrot and the stick in just the right measure, and his enthusiasm for the brain and how it works got me interested in this subject in the first place.

A warm thank you as well to the rest of the Mind Attention Interface Group. To James, for his flashes of inspiration and for keeping me on my toes. To Roy, for all his hard work building the bits of the system that I take for granted. To Torb, for his friendship and also for lifts to and from uni when I didn't want to ride in the rain. To Jocelyn for her helpful comments on this thesis. Thanks also to Jayeyong, Hugh, and all the other members of the Department who have contributed ideas and expertise.

A big thank you to the Department of Computer Science, and the Apple University Consortium, for their generous financial support for this Honours year of study.

Finally, to all my friends. To those who helped me out when I had assignments due, to those who made comments on my thesis, to those who offered sage advice in times of trouble, to those who helped me relax with a cold beer, thank you. Your prayers and support have been greatly appreciated.



---

# Abstract

---

The Mind-Modulated Music Interface (MMMI) provides a new way to create music. The system is designed to harness the musical processing power of the brain, allowing musicians to control the output of the system just by paying attention to it. The Phase Synchrony Engine, a component of the MMMI, is used to detect musical processing in the brain via EEG in real-time. This information is then used to modulate the output of a Music Generation Engine in real-time. A feedback loop is set up in which the brain interfaces with the music in a more direct way than has previously been achievable. In benchmarks, the MMMI has been shown to produce accurate results, and the MMMI is poised to investigate this new frontier in music creation.



---

# List of Abbreviations

---

**MMMI** Mind-Modulated Music Interface

**MAI** Mind-Attention Interface

**BCI** Brain-Computer Interface

**PSE** Phase Synchrony Engine

**MGE** Music Generation Engine

**BCMI** Brain-Computer Music Interface

**SSVEP** Steady-State Visual Evoked Potentials

**EEG** Electroencephalogram

**fMRI** Functional Magnetic Resonance Imaging

**MIDI** Musical Instrument Digital Interface

**ASC** Advanced Studies Course

**ANU** Australian National University

**MASE** Mind-Attention Spectral Engine

**DFT** Discrete Fourier Transform

**TCP** Transmission Control Protocol

**UDP** User Datagram Protocol

**QE** *Quantitative evaluation of linear and nonlinear methods characterizing interdependencies between brain signals*

**PRL** Physical Review Letters QE phase synchrony detection method



---

# Contents

---

<b>Declaration</b>	<b>iii</b>
<b>1 Introduction</b>	<b>3</b>
<b>2 A Next Generation Music Interface</b>	<b>5</b>
2.1 Music: A Historical Interface . . . . .	5
2.2 The Brain: Releasing the Music Within . . . . .	6
2.3 The Mind Attention Interface (MAI) . . . . .	8
2.4 The Rest of this Thesis . . . . .	8
<b>3 Literature Review</b>	<b>9</b>
3.1 Brain-Computer Interfaces . . . . .	9
3.2 Measuring Neural Activity . . . . .	9
3.3 Analysing Neural Activity . . . . .	10
3.4 Natural vs Artificial BCI Design . . . . .	12
3.5 Musical BCI Development . . . . .	13
3.6 Measures of Musical Processing in the Brain . . . . .	13
3.7 Functional Connectivity in the Brain . . . . .	15
<b>4 MMMI Overview</b>	<b>19</b>
4.1 Functional Connectivity in the MMMI . . . . .	19
4.2 MMMI Architecture . . . . .	19
4.3 The Mind Attention Interface . . . . .	20
4.3.1 Design Philosophy . . . . .	20
4.3.2 Previous Work by the Author . . . . .	21
4.3.3 The Wedge . . . . .	23
4.4 MMMI System Architecture . . . . .	24
4.4.1 MAI Server . . . . .	24
4.4.2 EEG Neural Signal Measurement . . . . .	26
4.4.3 Phase Synchrony Engine . . . . .	26
4.4.4 Connection Kit . . . . .	27
4.4.5 Music Generation Engine . . . . .	27
<b>5 Real-Time System Design</b>	<b>29</b>
5.1 Real-Time Feedback in the MMMI . . . . .	29
5.2 Simulink and the Phase Synchrony Engine . . . . .	30
5.3 Real-Time Simulink . . . . .	31
5.4 xPC Target . . . . .	31

---

<b>6</b>	<b>Phase Synchrony Engine Design</b>	<b>33</b>
6.1	Signal Input and Initial Processing . . . . .	33
6.2	Phase Estimation . . . . .	36
6.3	Characterising Phase Synchrony . . . . .	40
6.4	Algorithmic Complexity . . . . .	42
<b>7</b>	<b>Phase Synchrony Engine Testing</b>	<b>45</b>
7.1	Testing with Artificial Data . . . . .	45
7.2	Phase Coupled Systems . . . . .	45
7.3	Quantitative Evaluation of the Phase Synchrony Engine . . . . .	48
7.4	Further Testing . . . . .	49
<b>8</b>	<b>Music Generation</b>	<b>53</b>
8.1	Music Generation Environment . . . . .	53
8.2	Musical Aesthetics . . . . .	54
8.2.1	Rhythmic and Dynamic Structure . . . . .	55
8.2.2	Harmonic Structure . . . . .	57
<b>9</b>	<b>Status and Future Work</b>	<b>59</b>
9.1	MMMI Status . . . . .	59
9.2	Future Work . . . . .	59
9.2.1	User Testing . . . . .	59
9.2.2	Migration to Open-Source Software . . . . .	60
9.2.3	Further PSE and MGE Development . . . . .	60
9.2.4	The MMMI as a Creative Tool . . . . .	60
<b>10</b>	<b>Conclusion</b>	<b>61</b>
<b>A</b>	<b>Phase Synchrony Engine Block Diagram</b>	<b>63</b>
<b>B</b>	<b>Previous MAI Projects by the Author</b>	<b>75</b>
B.1	The MASE . . . . .	75
B.2	MASE-MAI Integration . . . . .	76
<b>C</b>	<b>The Mind Attention Interface Project</b>	<b>79</b>
C.1	MAI System Architecture . . . . .	79
C.2	Other MAI Projects . . . . .	81
C.2.1	Measuring Immersion in a Virtual Environment . . . . .	81
C.2.2	Immersive Computer Games in a Virtual Environment . . . . .	81
	<b>Bibliography</b>	<b>83</b>

---

# List of Figures

---

2.1	The stages of musical expression . . . . .	7
2.2	The MMMI music creation approach . . . . .	7
3.1	General BCI structure . . . . .	10
3.2	The P300 Speller . . . . .	11
3.3	Inter-region neuronal connections . . . . .	16
4.1	MMMI structure . . . . .	20
4.2	Comparison between the MAI and the MMMI . . . . .	22
4.3	The Wedge . . . . .	23
4.4	The MMMI Wedge configuration . . . . .	24
4.5	Data transmission in the MMMI . . . . .	25
5.1	The Simulink PSE polling the input port buffer in real-time . . . . .	32
6.1	Information flow through the PSE . . . . .	33
6.2	Gamma-band filter response . . . . .	34
6.3	The Biosemi EEG cap . . . . .	34
6.4	Each signal windows is processed individually by the PSE . . . . .	35
6.5	Phase of $e^{\frac{j\pi n}{3}}$ (in degrees) for one period, $n = 0, \dots, 5$ . . . . .	37
6.6	What is the phase of this signal? . . . . .	37
6.7	The DFT overlap between $Z[k]$ and $Z^*[((-k))_N]$ . . . . .	39
6.8	The components of the PSE algorithm . . . . .	42
7.1	Test data system structure . . . . .	45
7.2	The effect of the coupling parameter $c$ on the phase terms for $c = 0.1$ (weak synchrony) and $c = 0.9$ (strong synchrony) . . . . .	47
7.3	Artificial test data results . . . . .	48
7.4	Testing the validity of the generated analytic signal . . . . .	50
7.5	Log magnitude FFT for original and analytic signals . . . . .	51
8.1	Information flow through the MGE . . . . .	53
8.2	Aesthetic influences in the MGE . . . . .	55
8.3	An active hit may spawn a new hit . . . . .	56
8.4	The groove creation process . . . . .	56
8.5	The tonnetz . . . . .	57
C.1	MAI architecture . . . . .	79



# Introduction

---

Listening to music is largely a *passive* experience. When playing a CD, the CD player is unaware if anyone is listening or not, and the music playback is the same in either case. When a musician plays an instrument, however, it is an *active* experience: the musician is in direct control of their musical output. In these cases, there is a clear separation between the passive and active musical experience.

The Mind-Modulated Music Interface (MMMI) blurs this distinction, allowing a musician to interact with the musical output of a system directly with their mind. This approach combines the surprise and anticipation of the passive experience with the control and interactivity of the active musical experience in a way which has not been previously explored. The goal of the MMMI is to allow people to experience music in an entirely new way by allowing the mind to be an active collaborator in the musical experience.

This report describes the design, construction and testing of the MMMI, and suggests possible directions for the future of the project.



# A Next Generation Music Interface

---

## 2.1 Music: A Historical Interface

All human cultural groups exhibit their own form of musical expression [1], and the creation and appreciation of music is a skill intimately linked to the human experience. Throughout the ages, the musical traditions of cultural groups have been shaped by the particular musical instruments at their disposal. As musical instrument technology has changed, new avenues of musical expression have emerged and have sometimes become synonymous with certain cultural periods in history; from the earliest tribal drumming in Africa to the harpsichord of the Baroque period to the Moog synthesizer of the 1970s. A musical instrument is an application of technology to the design of an interface for producing music. As interfaces, musical instruments employ many different human sensorimotor modalities, from blowing into and manipulating the valves on a trumpet, to fingering the keys on a piano, drawing a bow across the strings of a violin, and so on. These interfaces have been designed to allow musicians to express their musical ideas. Just as a steering wheel is an interface designed to give us the ability to drive, a musical instrument is an interface which allows us to venture into the world of musical expression.

Musical creativity must, inevitably, be filtered through the technical constraints imposed by a given interface. It is impossible, regardless of the talent of the musician involved, to coax a major triad (three notes sounding simultaneously) out of a trumpet, or a staccato passage out of a ringing bell. These interface constraints are both a blessing and a curse; while they limit a musician's musical output to a subspace of all musical expression, it is often in grappling with the limitations of a given instrument that a musician may produce his or her finest work. The musical output of a musician is a product of both their own internal musical inspiration, and the characteristics of their musical interface.

## 2.2 The Brain: Releasing the Music Within

One famous example of musical inspiration is the genesis of the Beatles song ‘Yesterday’. In *Paul McCartney: Many Years From Now* [2], McCartney recalls

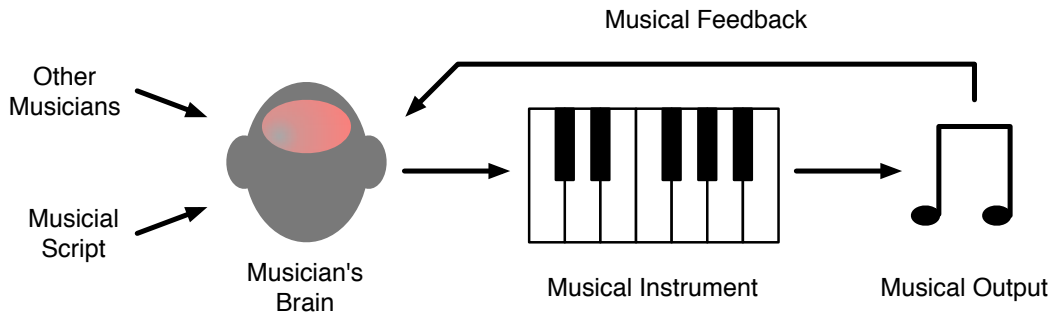
I woke up with a lovely tune in my head. I thought, “That’s great, I wonder what that is?” There was an upright piano next to me, to the right of the bed by the window. I got out of bed, sat at the piano, found G, found F sharp minor 7th – and that leads you through then to B to E minor, and finally back to E. It all leads forward logically. I liked the melody a lot, but because I’d dreamed it, I couldn’t believe I’d written it. I thought, “No, I’ve never written anything like this before.” But I had the tune, which was the most magic thing!

This is one example of the enigmatic nature of “musical inspiration” - a complex subject which is beyond the scope of this thesis (see [3] for an interesting discussion). Ultimately, though, the brain is the source of all creativity, and all art and science throughout human history is a testament to the creative output of the human brain.

In recent decades, concerted attempts have been made to explain the inner workings of the mammalian brain itself. Medical and technological advances have enabled the study of the human brain in great detail, and many insights have been gained from studying the brains of other animals in laboratory environments. While the philosophical question of whether the brain can ever truly understand itself remains unanswered, great strides have been made in understanding both low-level concepts, such as inter-neural signalling and cortical topology (texts such as [4] provide a useful reference), as well as higher-level concepts such as memory [5] and intelligence [6].

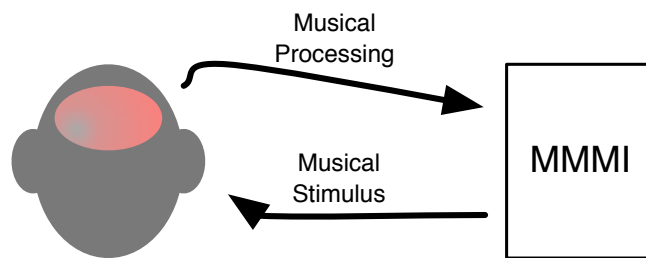
The 21st Century promises to be a period of great strides in our understanding of the brain. As we develop a better understanding of the brain, we will grow closer to demystifying the elusive concept of musical, and in fact all artistic, creativity. This will suggest exciting new directions in the design of music creation interfaces as we seek to unleash the full creative power of the brain. The purpose of this thesis is to present a novel music creation system, the Mind-Modulated Music Interface - MMMI (“triple-M I”). As developed through the course of this Honours year of study, the MMMI is not yet ready to revolutionise the way we create music overnight, but rather it is a prototype which suggests a way that the art of making music can benefit from advances in technology and in our recent understanding of how the brain works.

To elaborate on the discussion of the previous section, the instruments with which musicians conventionally make music create a necessary level of abstraction between the musical ideas in the brain of the musician and their musical output as perceived by a listener (see Fig. 2.1). With training, this gap can be bridged to some extent: a novice musician may express frustration at the ease with which they can hear a melody in their head compared with the difficulty they experience in coaxing the same melody out of their instrument. Conversely, an experienced musician may not be conscious of this process at all, effortlessly forming beautiful melodies while all of the translation happens subconsciously.



**Figure 2.1:** The stages of musical expression

The MMMI takes a different approach to music creation, which is shown schematically in Fig. 2.2. The idea is to directly measure the musical creativity in a musician's brain, using data obtained from an electroencephalogram (EEG), and then to hand this information off to an 'expert system' for musical realisation. Ultimately, a musician should be able to drive the MMMI by thinking musical thoughts without the need to worry about their realisation in a practical way on a musical instrument. Of course, as mentioned above, the processes involved with musical creativity in the brain are not well understood, and 'expert systems' for automatic music creation are similarly immature, so the approach taken in the MMMI is to build a generative music engine whose output is *modulated* by measures of musical processing in the brain which have been derived in real time from EEG data. By modulating the musical experience in this way, the musician should, in principle, still get a sense that the musical output of the system is *responsive* to their musical sensibilities.



**Figure 2.2:** The MMMI music creation approach

The MMMI is an ambitious project, and because it incorporates cutting-edge ideas from neuroscience, the particular musical processing measures used in the project are not perfect. As such, the particular implementation of the MMMI described in this thesis is a preliminary one, and the software implementation of the MMMI has, therefore, been designed in a modular way, so that new discoveries in neuroscience may be incorporated into the system as they occur. The MMMI pilot study described has the twin merits of being a prototype of a genuinely new way of controlling musical expression and performance as well as providing the infrastructure which can be extended to build a full-blown musical instrument over time.

## 2.3 The Mind Attention Interface (MAI)

The MMMI project is part of a larger umbrella project known as the Mind Attention Interface (MAI) which aspires to build immersive Brain-Computer Interfaces (BCIs) for virtual environments. Existing at the intersection of BCIs and Virtual Reality (VR) technology, the goal of the MAI is to use direct measurements of physiological activity to modulate and enhance the degree of “immersion” experienced by a participant. The MAI has a much broader scope than the MMMI, in that it involves both visual and aural stimuli and feedback. It is also driven by other physiological measurements, particularly eye gaze and head position as well as EEG. The relationship between the MMMI and the MAI is discussed in Chp. 4.

The main MAI concept was originally developed by Dr. Henry Gardner and PhD student James Sheridan from the ANU Department of Computer Science and all of the MAI component projects share a common distributed client-server software architecture which has been primarily developed by Zhen Yang for his MPhil research [7].

## 2.4 The Rest of this Thesis

The rest of this thesis describes the work which the author has done in the course of this Honours year. This work includes the invention of the MMMI concept, as well as the decisions involved with its design and construction. It describes the genesis of the MMMI idea as a new application of recent developments in neuroscience and Brain-Computer Interface design. It then describes the implementation of the system as an exercise in software design, and also the process of testing the system.

In particular, this thesis describes the two key components of the MMMI, the **Phase Synchrony Engine** (Chapters 6 and 7) and the **Music Generation Engine** (Chp. 8). These components represent the majority of the research content of the thesis. The thesis also proposes a test methodology for examining the success of the MMMI as an environment for musical expression.

---

# Literature Review

---

## 3.1 Brain-Computer Interfaces

A Brain-Computer Interface (BCI) is a system in which computer control information is extracted from direct measurements of neural activity in a user's brain. Development on such systems began in the 1970s, with the term 'Brain-Computer Interface' first being used by Jacques Vidal [8]. Since then, the advent of low-cost, digital computers and signal measurement technology have led to an increased interest in BCI design and research in the scientific community. An interesting review of BCI history can be found in [9]. In designing a BCI, there are two main questions to answer:

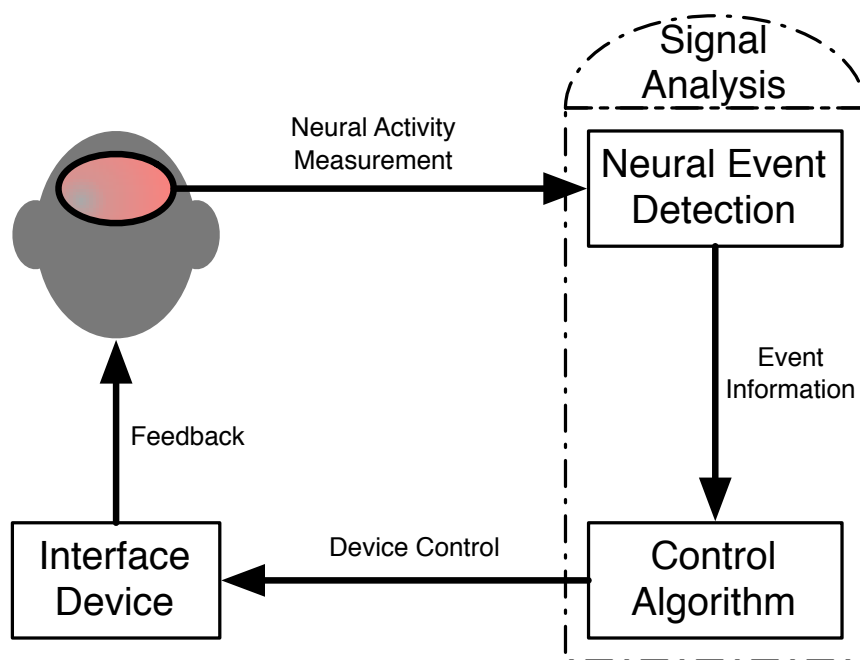
- **Measurement** - What is the best way to measure the neural activity in the brain?
- **Analysis** - How best to extract the desired control information from the neural data stream?

The general structure of a BCI is shown in Fig. 3.1. Within this framework, there are many different design options. The design choices made are largely dependent on the goals of the particular BCI.

## 3.2 Measuring Neural Activity

Measurement of neural signals can be done either invasively or non-invasively. Invasive measurement techniques usually involve implanting electrodes into the brain to directly measure the electrical activity in the cortex [10]. This provides a cleaner signal than non-invasive techniques, but is a more drastic procedure involving major surgery. Consequently, this method is more often used in animals [11] than with people, but it has been particularly used in patients with impaired brain function [12]. Non-invasive techniques sacrifice signal quality for ease of use and the ability to connect a user up without the need for risky and irreversible surgery and are, of course, much more suitable for use in the MMMI.

The two main signal-acquisition technologies in use in modern non-invasive BCIs are functional Magnetic Resonance Imaging (fMRI) and the electroencephalogram (EEG).



**Figure 3.1:** General BCI structure

fMRI works by measuring the level of blood oxygenation in the brain, and the measurement apparatus is a large machine which the participant must sit inside. Just like any other organ, the brain requires oxygen to function, and this can be exploited to measure the approximate amount of neural activity in any brain region [13]. fMRI provides excellent spatial resolution, but poor temporal resolution. In comparison, EEG provides much finer temporal resolution than fMRI, at the expense of spatial resolution. An EEG measures neural activity by placing a network of electrodes in direct contact with the scalp. These electrodes measure the net electric field generated by the synchronous firing of groups of similarly oriented neurons [14]; the electric field propagates through the cerebro-spinal fluid and skull and is detected by the scalp electrodes.

Because of their complementary strengths and weaknesses, fMRI and EEG are sometimes used simultaneously [15]. Although there are also other non-invasive ways to measure the activity in the brain, these two technologies are the basis of most current BCI research. EEG is the method used in the Mind Attention Interface.

### 3.3 Analysing Neural Activity

The goal of a BCI is, ultimately, to allow a user to control a system directly with their brain. To do this, the signal analysis component of a BCI is usually a two-stage task: specific ‘neural events’ must firstly be detected in the neural signal, and these events are then mapped to control actions in the interface (see Fig. 3.1).

There are many well-known neural events which have been used for BCIs. Some of the most common are:

- A **P300 response** is associated with the recognition of anticipated stimuli. P300 based BCIs often ask participants to monitor a stream of varying stimuli (either auditory or visual) whilst trying to recognise a particular target stimulus from that stream. When the user detects the target stimulus, a spike in neural activity is evident approximately 300ms after the event detection [16]. The magnitude of the spike is inversely proportional to the frequency of the target stimulus. This type of experimental setup is known as the *oddball* paradigm [17], as the target stimulus is often an ‘odd-one-out’ from a group of presented stimuli.
- **Imagined Movement** is the action of having a participant imagine moving a limb, without actually doing so. This ‘imagined movement’ can be detected by an decrease in mu-rhythm (8 - 12Hz) power in the opposite side sensorimotor cortex [18]. Actual movement of the limb is not required. Given that left and right limb imagined movements are easily distinguishable, this technique is often used for left-right selection in BCI menus.
- **Steady-State Visual Evoked Potentials (SSVEPs)** are oscillations which are established in the brain when a participant fixates on a flickering visual stimulus. The SSVEP oscillates at the same frequency as the flickering frequency of the visual stimulus. By designing a visual interface with targets which flicker at different frequencies, the SSVEP can be used to determine which target the participant is looking at [19].
- Other, alternative, neural events have also been used in BCI design [20].

All these techniques rely on the known response and activation profile of the brain to different sensory inputs or processes. In most cases, these neural events are relatively easy to detect, and robust classification algorithms exist [19], [21], [22].

A	G	M	S	Y	*	
B	H	N	T	Z	*	
C	I	O	U	*	TALK	
D	J	P	V	FLN	SPAC	
E	K	Q	W	*	BKSP	
F	L	R	X	SPL	QUIT	

Highlighted Column

**Figure 3.2:** The P300 Speller

An example of how conventional neural events can be used to control an interface is the *P300 speller*, first proposed in 1988 [23]. The goal of this interface is to spell words one letter at a time. The letters of the alphabet, as well as some basic editing

operations, are presented in a  $6 \times 6$  grid as shown in Fig. 3.2. The participant is required to focus on the character they wish to select in the grid. Each row and column of the grid is highlighted, one at a time. The participant is looking out for their desired character to be highlighted, when this occurs it triggers a P300 response in the participant's brain. The desired character is highlighted twice: once in a row and once in a column. By selecting the row and column with the greatest measured P300 response, the desired character can be inferred. In the original study of this method [23], a communication rate of 12 bits/min was achieved (the neural event used in this interface is the P300 response, the control model is that of the P300 speller). In subsequent work, the exact same P300 speller paradigm was able to achieve a communication rate of 84.7 bits/min by improving the neural signal-detection algorithm [22].

The analysis of neural events in BCI design is informed by advances in neuroscience and our understanding of how the brain works. The design of new BCIs also provides motivation and a practical testbed for new theories of brain function. BCI research is therefore a dynamic discipline, whose evolution is driven by both technology and fundamental science.

### 3.4 Natural vs Artificial BCI Design

Several of the established BCI paradigms based on the detection of the neural events mentioned in the previous section have been shown to be quite effective. As previously mentioned, the P300 speller [22] gives a communication rate of 84.7 bits/min with 95% accuracy. However, the neural events which form the basis of this human-computer communication often have very little to do with the desired outcome or goal of the interface itself. While this may provide an effective interface methodology, there is little correlation between the desired outcome of the interface (composing a word, typing a letter etc.) and the neural events required to achieve the goal (imagined limb movement). For this reason, we shall call these conventional interfaces “*artificial*” BCIs.

At the other end of the spectrum are BCI designs where there is a direct and obvious relationship between the neural events to be detected and the purpose of the interface. These “*natural*” interfaces should afford the construction of systems which augment, rather than derail, task-related neural activity and should leave the user with more cognitive resources to complete a task than they would have access to unaided. This natural interface design necessitates the detection and quantification of higher-level neural processes such as cognition, emotion and creativity as well as other task-specific resources. Less is known, however, about these correlates of higher level neural activity and the detection of their neural events and activation patterns is a more difficult task than for artificial data, and the control rate of such systems is not comparable to the simpler approaches used in artificial BCI design [24].

In reality, these categories represent not so much discrete classes as a continuum, and any BCI design will fall somewhere between the two. However, most current BCIs are located towards the ‘artificial’ end of the BCI continuum. This is primarily due to the lack of understanding of the neural events required to construct a useful ‘natural’

---

BCI.

### 3.5 Musical BCI Development

Musical BCI technology, while still in its infancy, has already been demonstrated as a viable approach to music creation [25]. Using established BCI control techniques, a control signal which may be used to select a character from a character set (as in the P300 speller) to enable writing can be mapped to a musical synthesiser. With appropriate musical expertise in creating these mappings, highly musical output can be created even with the low bit-rates associated with current BCI designs.

Pioneering work in this type of musical BCI has been carried out by the Brain Computer Music Interface (BCMI) group at Plymouth University in the UK. In the BCMI system [26], a participant's neural signal is processed using several different techniques (such as imagined movement, alpha-band power and eye blink detection), effectively giving multiple simultaneous control streams. These control streams are then combined to drive a musical synthesis engine. The 'biomusician' can, with training, produce the neural patterns required to consciously produce a desired control signal. Other biological data, collected by heart rate and muscle contraction sensors, is also used to drive the music creation processes. Some of the control signals measured in the Plymouth BCMI are eye blinks (as detected from the EEG signal) and imagined hand movement: the musical output is turned on and off by the biomusician's eye blinks and it is subject to filtering and panning based on imagined movement of the left and right hands. With training, the biomusician can control the panning of the music consciously by using the appropriate lateral trigger. This approach proved successful enough to enable a concert to be given demonstrating the use of the BCMI in action in 2005 [27].

The BCMI falls towards the artificial end of the BCMI spectrum. While a skilled biomusician may have the holistic musical output of the system in mind, low level control is achieved by distinctly non-musical thought patterns such as imagined movement. This is not necessarily a bad thing, it merely represents a conscious decision by the designers. The neural events utilised in the system are (relatively) well understood, and there exist robust detection algorithms, so that the BCMI is able to achieve a control accuracy greater than 95%. The BCMI does not, however, utilise the natural musical processes in the brain and this is a primary point of difference between it and the MMMI.

### 3.6 Measures of Musical Processing in the Brain

What, then, are *natural musical processes* in the brain? Is it possible to quantify the musical processing or attention of a given individual? It is first helpful to examine what is meant by the phrase *musical processing*.

In the MMMI, the idea of musical processing is defined to be the neural patterns and cognitive processes associated with creating and analysing music. Every human being is capable of listening to music. Indeed, most people find it a pleasant experience,

although the degree of pleasure is dependent on the type of music and the musical tastes of the listener. Creating music, on the other hand, is considered a special skill, and is generally only practised by those who have been trained to do so (although this training may take many forms). Those trained in music are also often highly skilled in musical analysis, recognising trends and patterns in music (such as harmonic structures) to which the casual listener is oblivious. How is this musical skill manifested in the brain? What activation patterns in the brain of a musician capable of which allow them to create music?

To answer this question, several studies have investigated the differences between the brains of musicians and non-musicians. Professional keyboard players (more than one hour practice per day) have been shown to have greater brain volume in several brain regions when compared to amateur and non-musicians ([28]; similar results have been reported in [29], [30] and others). These results show quantifiable changes in the brain as a result of musical training, but these changes are static measures of long term musical development rather than the dynamic measures of instantaneous musical processes. If the MMMI is to be used to create music in real time, a more dynamic measure of musical processing is required.

A more promising result for the MMMI is a recent study by Joydeep Bhattacharya and Helmuth Petsche which monitored musicians and non-musicians via EEG while listening to music [31]. In particular, this study monitored *phase synchrony* in the participant's neural signals as they listened attentively to various types of music. Phase synchrony is an indicator of *functional connectivity*, which represents the task-specific neural connections between different brain regions. The musicians showed a significant increase in synchronised activity in the gamma frequency band (30-50Hz) over the non-musicians when listening attentively to the music. Gamma-band activity is known to be involved in the perception of auditory stimulus [32]. There was also a control condition where the participants listened to plain text being read, and no difference in phase synchronisation was detected between the groups in this case. The musicians in this study all had at least 5 years of formal music study.

In the Bhattacharya & Petsche study, the musicians were not creating the music themselves, they were merely listening. However, the authors suggested that the result may be due to the increased musical ability of the musicians: a musician's experience with creating and working with music allows them to engage with music on a deeper level. They may be simultaneously analysing the melody, rhythm, chord progression, dynamics and timbre of the music. All of these separate *perceptual streams* are then integrated in the brain to form a complete picture of the musical stimulus. It is this *perceptual integration* which causes the increase in functional connectivity. Given that the non-musicians are not attending to the musical stimulus on as many levels (due to their relative lack of experience in musical analysis), the amount of functional connectivity in the brain is also smaller.

Other studies have shown *similarities* between musician and non-musician listeners in EEG during musical perception. Changes in spectral power, particularly in the alpha band between 7Hz and 12Hz, has been exhibited during attentive listening to music [33]. This effect has also been noticed when music is used to aid performance

---

in various tasks involving logic and spatial reasoning [34]. There are certainly many aspects of the brain's response to music which are universal. However, as the study by Bhattacharya & Petsche shows, there are musical processes in the brain of a musician which are not present in those without musical ability. These differences provide clues as to the mechanism of musical creativity in the brain.

Bhattacharya and Petsche's study is not the final work on musical processing in the brain. It merely suggests one neural correlate of musical expertise in musical processing. The dynamic nature of this measure is its main advantage in a real-time environment such as the MMMI. While functional connectivity may not entirely measure the elusive 'musical creativity' of the brain, it is a promising indicator of the processes that are specific to a musical brain.

### 3.7 Functional Connectivity in the Brain

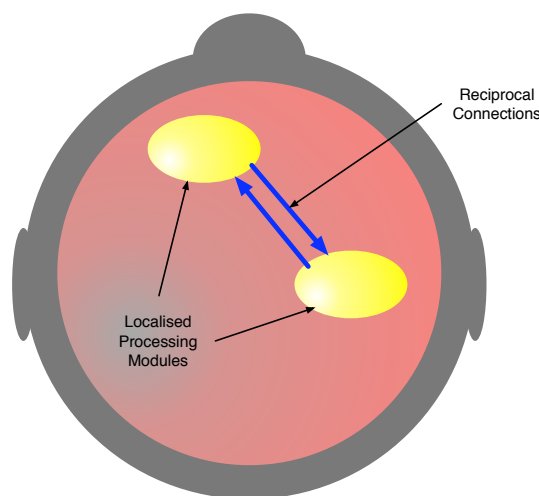
The cortex is a thin sheet of neural tissue on the outside of the human brain which comprises about 60% of the brain by volume, and plays a central role in information processing [35]. The way the cortex processes information is highly complex, using both *modular* and *distributed* processing techniques [36]. The modular nature of this processing is manifested in the specialisation of different cortical regions. Vision and sound, for instance, are processed in distinct areas, and these areas have a well-defined spatial profile. The specialisation of the different areas of the brain has been historically studied by observing patients with localised brain damage but these days it is usually associated with fMRI studies.

The cortex can be seen to be made up of distinct processing modules, each differing in specialisation and location [37]. From higher-level modules, processing can be further decomposed into specialised submodules, each carrying out a particular component of the overall task. These modules can be arranged hierarchically, with each level of the hierarchy representing a different level of specialisation, and these different modules give the brain an enormous amount of processing power: they give us the ability to process, in parallel, five sensory input streams, to produce complex, co-ordinated motor-control information, and enable all the different tasks and emotions which constitute our thoughts and behaviour.

Although the brain can be viewed as a collection of specialised modules, this is not the way people perceive the world around them. The information from all the different modules is combined to form a holistic model of the world. For example, when looking at a moving car an observer is not aware, separately, of the noise the car makes, the sight of the car approaching, and the feel of the wind as it rushes past. The observer perceives the car as a single object, and the different sensory streams conspire to build a unified idea of the car in the mind of the observer. The question of exactly how these disparate information streams (from the specialised modules) are integrated into a cognitive whole is known as the *integration problem*. Recent results have suggested that the solution to the integration problem may lie in the temporal profile of neural activity in the brain [38].

While each region of the brain has its own processing speciality, these regions are

all highly connected by millions of ‘inter-region’ neuronal links (see Fig. 3.3). When two different modules are communicating, their activity is synchronised due to the active connections between them [39]. Synchronous activity between two different modules is an indicator of communication and integration between the two modules. Depending on the type of information being processed, different modules will be required to communicate in this way. This type of connectivity is known as *functional connectivity*. Functional connectivity is task-specific; it changes depending on the type of information the brain is processing. The *functional connections* between brain modules typically last between 100 and 300ms [40].



**Figure 3.3:** Inter-region neuronal connections

Typically, functional connectivity in the brain is measured by determining the amount of *phase synchrony* present in the neural signal, often in a specific frequency band. Mathematical techniques have been developed for extracting phase synchrony information from the neural signal, and a quantitative analysis of several of these techniques can be found in [41], which also suggests a benchmarking methodology for determining the accuracy of a given phase synchrony detection method. The use of this validation procedure to test the MMMI will be discussed in Chp. 7.

The study by Bhattacharya and Petsche [31] found higher levels of phase synchrony in the gamma frequency band (30-50Hz) in musicians compared with non-musicians, which suggests more extensive functional connections in musicians while attending to music. In the light of functional connectivity as a characteristic of cognitive integration, this result makes some intuitive sense: the detailed musical analysis streams in a musician’s brain would need to be integrated to give a complete picture of the musical stimulus. This integration is characterised by multiple functional connections between regions of the cortex, which would result in an increase in phase synchronised activity. Conversely, the non-musicians’s musical analysis was not as detailed, and therefore, there was less information to integrate, resulting in lower amounts of functional connectivity.

Although detailed musical processing, it seems, bears shows some of the characteris-

---

tics of the distributed neural processing common to more general integration problems, it is important to note that the Bhattacharya & Petsche result has been specifically demonstrated for enhanced musical processing by *musicians*, it is thus appropriate for use in the MMMI.



---

# MMMI Overview

---

## 4.1 Functional Connectivity in the MMMI

The result by Bhattacharya and Petsche discussed in Chp. 3 is an exciting one for the idea of building a Mind-Modulated Musical Interface. Here, it seems, is a viable time-varying measure of musical processing in the brain. The MMMI challenge, then, is to harness this information in a usable music creation interface. The functional connectivity measure is not particularly ‘information rich’. Therefore, it is not feasible to turn it into music directly, as the resulting music would be very simplistic. An alternate approach is to use the functional connectivity measure of musical processing to modulate the musical output of an distinct musical source. This modulation would allow a user to have meaningful musical control over the musical output of the system with even a small amount of neural musical processing data.

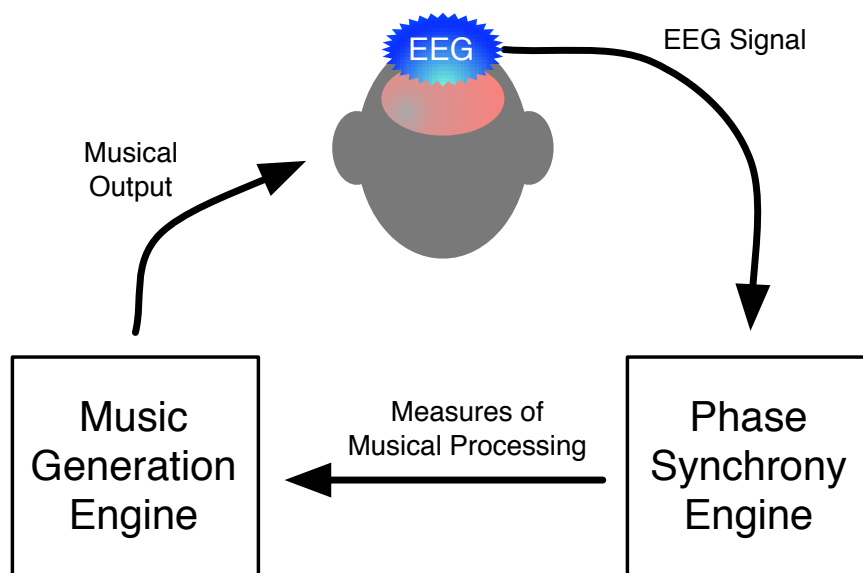
An appropriate music-generation engine for the MMMI must be responsive to the musical input extracted from the neural signal. An intuitive way to do this is to generate music which varies along some characteristic dimension, such as pitch, volume, tempo, or note density, and to directly map this characteristic dimension to measurements of musical processing in the participant’s brain. For example, an increase in the level of phase synchrony could cause the generated music to increase in tempo, and vice versa. While such a simple example may seem unsatisfactory from a musical standpoint, it provides a starting point for the music creation methodology to be employed in the MMMI and also suggests possible human factors tests which could follow its development: whether the musician *feels* that their musical thought processes affect the characteristics of the generated music.

A key idea in the MMMI is the idea of feedback: the musician listens to the output of the system, and, in processing this output, their musical skills are engaged in their brain. This musical neural activity is then measured in real-time and fed back into the music generation engine, which adjusts its output accordingly. The brain and the music generation engine form a closed-loop system, and their interaction over time represents a new outlet for musical processing and creativity (see Fig. 4.1)

## 4.2 MMMI Architecture

As a research problem, the MMMI consists of two separate tasks:

1. Measuring the amount of musical processing in the brain
2. Generating a musical stimulus



**Figure 4.1:** MMMI structure

These tasks are reflected in the basic structure of the MMMI. The system is composed of two basic components, the Phase Synchrony Engine (PSE) and the Music Generation Engine (MGE), as shown in Fig. 4.1. The PSE is responsible for measuring the amount of musical processing occurring in the musician’s brain. This information is then passed to the MGE, which generates musical output to be played back to the musician. The interaction between these two components governs the behaviour of the system. This real-time interaction of the musician’s brain and the generated musical stimulus defines a new method of musical expression, which is the stated aim of the MMMI.

While these two components constitute the research element of the MMMI, there are other tasks essential to the operation of the MMMI. The participant’s neural activity is measured via EEG, and the musical stimulus is output via a multichannel speaker array. The Mind Attention Interface (MAI) system is an existing distributed system architecture for connecting the components of the MMMI together for real-time BCI operation. We shall now consider the MAI system and its use in the MMMI.

## 4.3 The Mind Attention Interface

### 4.3.1 Design Philosophy

The concept of “attention” is a difficult one to define, but a definition given by one of the fathers of modern psychology, William James, in 1890 is [42]

---

“... the taking possession by the mind, in clear and vivid form, of one out of what seem several simultaneously possible objects or trains of thought... It implies withdrawal from some things in order to deal effectively with others, and is a condition which has a real opposite in the confused, dazed, scatterbrained state...”

Attention plays a crucial role in human information processing. In attending to a particular stimulus, the brain’s resources are devoted to that processing, which allows us to filter out things of interest from the chaos of everyday life. The idea of the Mind Attention Interface (MAI) is to harness a participant’s attention in determining their use of the interface. The MAI system measures both neural activity (via EEG) and eye-gaze information (via two face-monitoring cameras) to determine the state of the participant’s attention. The MAI is therefore not strictly a BCI, because non-neural measurements are also a part of the interface. The eye gaze data provides useful information about the visual attention of the participant [43] by monitoring the rate of eye movements, including involuntary saccades, and by calculating the objects which are the focus of the visual field. This idea of augmenting neural activity measurements with other physiological data is an advantage in the difficult task of measuring attention.

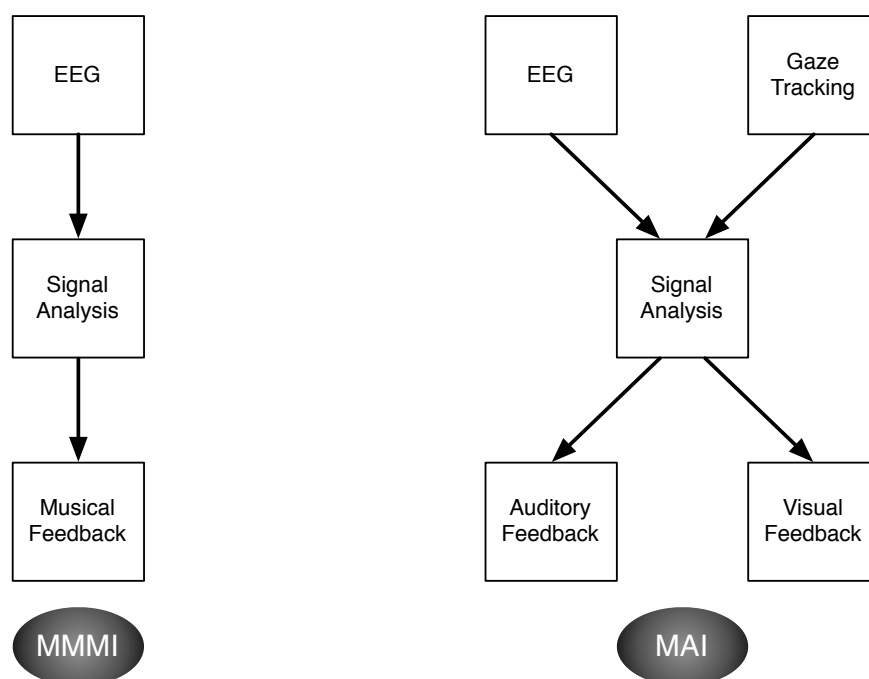
The MAI is currently under active development as the PhD project of James Sheridan, under the supervision of Henry Gardner. The task of extracting information about attention from the EEG and eye gaze measurements is a difficult one, but the last decade in particular has seen the development of attention measurement techniques (see, for example, [44], [45], [46]), and some of these techniques are currently being developed for use in the MAI.

The MMMI, while a distinct project to the MAI, has some similarities in its approach. Both interfaces are designed to augment the natural processes in the brain rather than derail them. In some ways, the MMMI can be considered an application of the MAI ‘attention-modulated interface’ idea to the specific task of *music creation*. Because of these similarities, a great deal of the infrastructure required to implement both interfaces is common to both projects. A comparison of the interfaces can be seen in Fig. 4.2. To see how this has been achieved, it is necessary to describe the MAI system in greater detail. Additional information about the wider MAI project is contained in Appendix C.

### 4.3.2 Previous Work by the Author

The author’s collaboration with the MAI team began in July 2006 and continued through the summer of 2006/2007. Prior to commencing this Honours year of study, the author had already completed two 6-unit ‘Advanced Study Courses’ (ASCs) in this field. It is, therefore, necessary to describe this work as a precursor to the MMMI.

The first ASC was completed in November 2006. At that time, the MAI EEG signals were not being analysed in any meaningful way. The goal of the ASC was to build a module for *real-time* spectral analysis of the EEG signal and to incorporate it into the MAI software system. This module, the Mind Attention Spectral Engine



**Figure 4.2:** Comparison between the MAI and the MMMI

(MASE), employed a window-based, Fourier-transform approach to calculate the power spectrum of an EEG signal, and it was verified to give correct output when verified against artificial test data. The MASE is described in more detail in Appendix B.

The second ASC was completed in December-February of 2006/2007. With the MASE completed and verified as a standalone unit, the aim of this ASC was to achieve real-time communication between the MASE and the rest of the MAI. This involved developing an interface between the MASE (written in Simulink, part of the MATLAB package [47]) and the MAI server [7] (written in C++). This proved to be quite difficult, as Simulink is designed primarily for offline simulation rather than real-time operation (the reasons for this, as well as a rationale for the choice of the Simulink environment, are discussed in Chp. 5). At the end of the summer project an interface had been constructed and demonstrated, but several of the synchronisation issues encountered during the project were not completely resolved. At this stage, the MASE was connected and ready for use with the rest of the MAI, but the MAI interface was incomplete.

The MMMI itself has been developed within the timeframe of the 2007 Honours course: the MASE is *not* a part of the MMMI, as more advanced signal analysis techniques have been developed for use in the MMMI. The main benefit of the MASE development in the previous ASC projects was the familiarisation gained with the general architecture of the MAI. The lessons learnt in the development of the MASE have been very useful in guiding the development of the MMMI this year.

### 4.3.3 The Wedge

The feedback interface used in the MAI system is known as “The Wedge”. The Wedge is an immersive audiovisual environment designed by Henry Gardner and Rod Boswell [48]. Two large, back-projected screens meet at a right angle, with the participant located in the angle formed by the screens. The Wedge is capable of 3D imaging using LCD shutter glasses. The Wedge audio system consists of eight speakers arranged as the vertices of a cuboid. This *sound cuboid* arrangement allows precise sound localisation and panning effects. A diagrammatic representation of the wedge setup is given in Fig. 4.3.

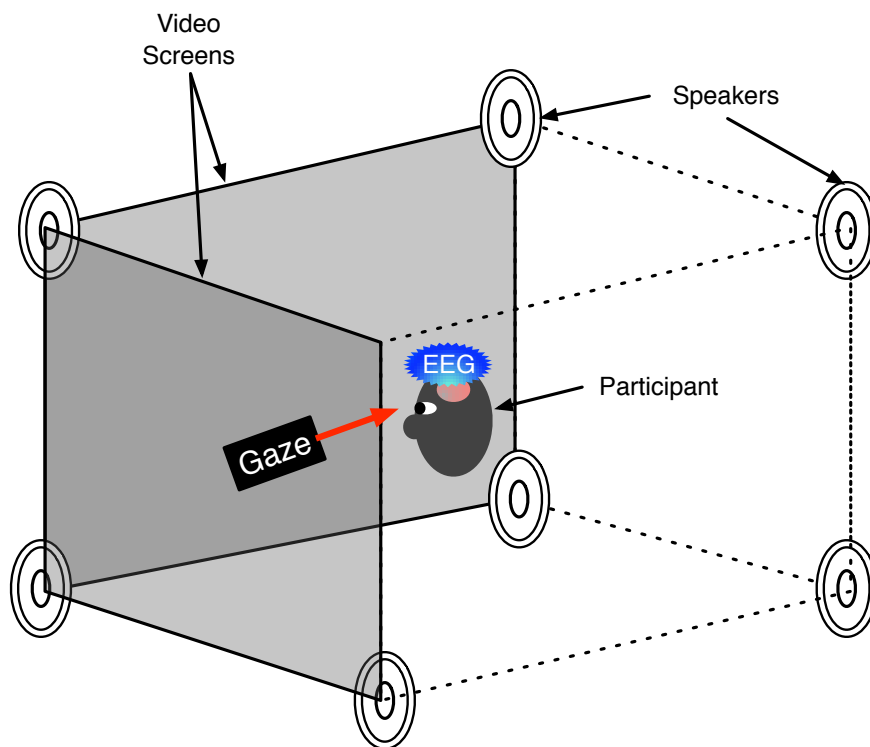
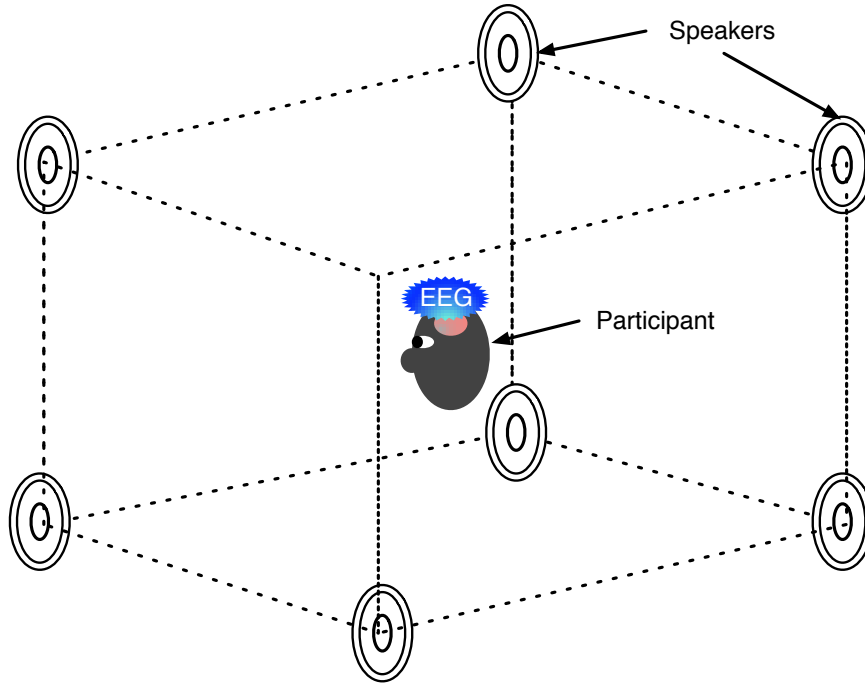


Figure 4.3: The Wedge

The physical arrangement of the MAI is a pair of moveable displays which can be arranged in a ‘V’ shape and inserted into the space of the wedge. When the participant sits at the MAI screens, their EEG and eye gaze can be monitored. Using this data, the participant’s attention is then measured as a response to the visual and audio stimulus presented on the screens and through the speakers. The interface is designed to allow the user to interact in a natural way with whatever stimulus commands their attention. The goal is not so much to determine what the the user must pay attention to, but to present them with several options and measure their natural attentional response. The MAI approach falls into the natural BCI category (see Sec. 3.4). It is designed to *observe* the user’s attention without dictating the shape it must take.

The MMMI, unlike the MAI, does not involve any visual stimulus or gaze tracking.

The MMMI therefore uses a subset of the Wedge’s functionality. In MMMI operation, the participant sits and listens to the musical output through the 8 Wedge *sound cuboid* speakers. The participant’s EEG is then used to measure their musical attention. The devices used in the MMMI feedback interface are shown in Fig. 4.4.



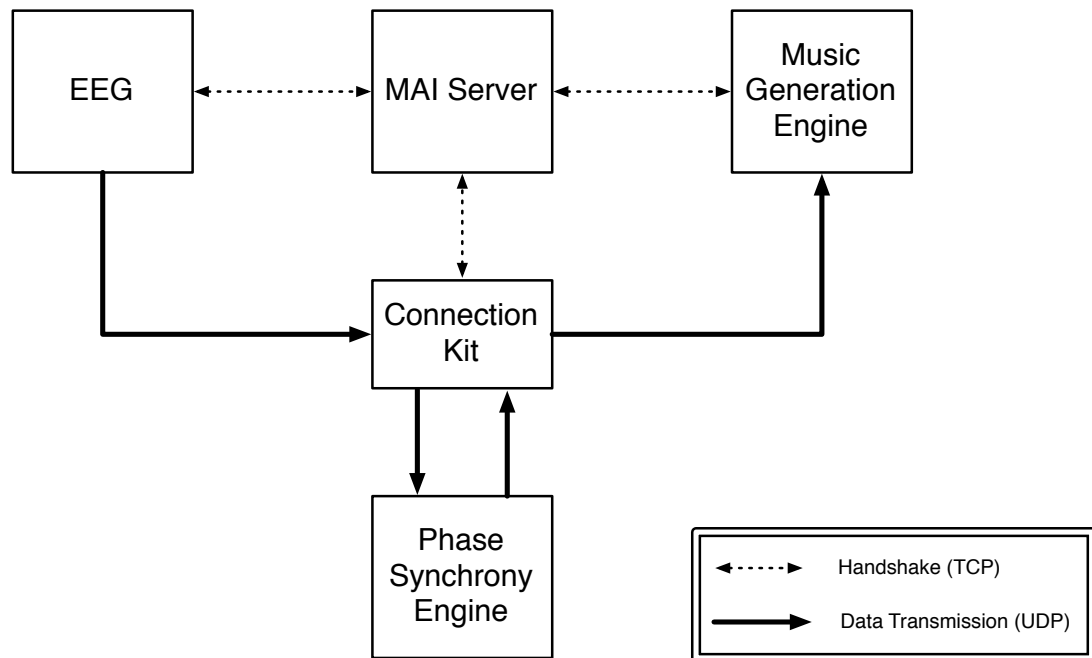
**Figure 4.4:** The MMMI Wedge configuration

## 4.4 MMMI System Architecture

The MMMI system shares a network communication architecture with the Mind Attention Interface. The architecture of the MMMI, shown in Fig. 4.5, is based around the *MAI Server*. The MAI server protocol allows the primary components of the MMMI, the Phase Synchrony Engine and the Music Generation Engine to communicate in real-time. The MAI Server protocol has been developed by Zhen Yang, a member of the MAI research team, in his Masters project [7].

### 4.4.1 MAI Server

The MAI Server protocol is an extension of an existing protocol called the Neuroserver protocol [49], which is designed specifically for EEG-based BCIs. The Neuroserver protocol is designed to allow real-time processing of the EEG signal by compatible processing modules. The functionality of these modules is up to the BCI designer, as long as the interface conventions of the protocol are maintained. However, the Neuroserver protocol was designed for EEG input only. Other physiological measurement inputs such as eye gaze (which is used in the MAI) are not supported.



**Figure 4.5:** Data transmission in the MMMI

There are two types of connections specified by the MAI protocol: handshaking and data transmission. A handshaking connection is established whenever a device connects to the MAI Server. The handshaking procedure involves the exchange of header information about that particular device, such as the name and type of the device, whether it intends to transmit or receive data, and the nature of the data. This information is sent as a string over a TCP connection. TCP is a connection oriented protocol, and also includes inbuilt error checking and acknowledgement [50]. While this adds overhead to the transmission process, the header data is only exchanged at the beginning of each session, and therefore performance is a secondary consideration to reliability. The handshaking procedure allows the MAI Server to maintain a list of all the connected devices at any point in time. This list can then be requested by new devices, in order to determine if the desired data stream is available.

The other type of connection supported by the MAI Server is a data transmission connection. A device can connect to the server, determine what other devices are connected, and then request any available data stream. This data transfer does not actually occur over a connection in the strict sense. The data transmission is performed with UDP multicasting. UDP, unlike TCP, is a connectionless protocol, so packets are merely sent, and the sender does not receive any acknowledgement that the packet has been received safely [50]. This lack of acknowledgement makes UDP much more efficient than TCP, although this efficiency comes at the expense of reliability. Furthermore, the data is sent directly to the interested devices rather than going through the MAI server first, removing a possible bottleneck in the system. After handshaking, a device informs the MAI server of its desire to receive data from any other connected device, and any

subsequent UDP packets are forwarded directly to the application. The interaction of the various system components and the MAI Server is shown in Fig. 4.5.

Each UDP data packet contains a counter representing the ‘packet index’ of that particular data stream. This information allows some low-level error checking at the receiving end, to mitigate the lack of checking in the UDP protocol. This error checking cannot be as comprehensive as the error checking in TCP (no checksums are involved) so the data integrity is not guaranteed. It is useful, however, in eliminating jitter and ensuring the received packets are processed in the correct order. Apart from this packet index, the format of the data is largely unspecified - any string can be sent as a payload in the UDP packet. This allows for flexibility in the type of data which can be sent. The disadvantage of this approach is that the format of a given data type (such as an EEG signal, or musical processing index) must be determined, *a priori*, to allow the data to be successfully unpacked at the receiving end. This is perfectly acceptable in the MMMI, as the structure of the system remains static during operation.

#### 4.4.2 EEG Neural Signal Measurement

The electroencephalogram (EEG) is used in the MMMI to measure the neural activity of the participant. The EEG measures the electric potential due to neural activity using electrodes distributed across the scalp. Recent technological advances have seen the introduction of the digital EEG, which can record this potential at discrete time intervals, and this digital output is ideally suited to further digital signal processing. Given the use of EEG for signal acquisition in the Bhattacharya & Petsche study, a similar approach is a natural choice for the MMMI.

The MMMI EEG apparatus is a Biosemi<sup>TM</sup> digital EEG, which records at 16 scalp locations (16 channels) at a sampling rate of 2048 Hz with 24-bit resolution [51]. The Biosemi uses active electrode technology, which is more resistant to noise than older passive electrode designs. The Biosemi EEG interfaces with the rest of the system using a custom MAI driver developed by Zhen Yang.

#### 4.4.3 Phase Synchrony Engine

The aim of the Phase Synchrony Engine (PSE) is to provide a time-varying indication  $\rho$  of the musician’s neural state, which is necessary for the construction of a responsive interface. The study by Bhattacharya & Petsche [31] provides the model for the type of musical processing to be detected by the PSE. In that study, the degree of phase synchrony between EEG electrode pairs was used to measure the functional connectivity in the brain. However, the neural signal analysis was performed offline, so this technique is unacceptable for use in the MMMI. The goal of the PSE is therefore to adapt the phase synchrony measurement techniques used in the Bhattacharya & Petsche for real-time operation. The PSE is written in the Simulink & xPC Target modelling language (see Chp. 5). The algorithm used in the PSE is described in Chp. 6, and the PSE operation is compared with existing benchmarks in Chp. 7.

---

#### 4.4.4 Connection Kit

The Simulink & xPC Target implementation of the PSE, as described in Sec. 5, is primarily designed for numerical processing. It also only has limited string processing functionality, and is not able to communicate using the MAI server protocol. To deal with this problem, an intermediate C++ application was developed by Zhen Yang called the Phase Synchrony Engine Connection Kit. The connection kit sits between the MAI server and the PSE (see Fig. 4.5), and interfaces with the MAI server using the MAI protocol. Any data to be sent to the PSE is then sent first to the connection kit using the string based MAI protocol. This data is then parsed into a numerical format (32 bit integers) and forwarded on to the xPC Target machine running the PSE via UDP. After processing, the PSE sends the processed results (that is, the measured phase synchrony  $\rho$ ) back to the connection kit, where it is reformatted to the MAI protocol and sent out to the rest of the MMMI. The connection kit is transparent to the operation of the MMMI; the system operates as if the connection kit were doing the PSE processing. While this is a slightly cumbersome workaround, the decision to implement the PSE in the Simulink language has many benefits which outweigh the disadvantages. Given the crucial role of the PSE in the system, the connection kit was a necessary concession in the construction of the MMMI.

#### 4.4.5 Music Generation Engine

The Music Generation Engine is responsible for the musical output which is played back to the participant. This is a crucial component of the system; the measurements of functional connectivity in the brain must be used in a meaningful way if the goal of an interactive musical experience is to be achieved. The output of the PSE is a scalar parameter  $\rho$  representing the measured musical processing in the musician's brain, at an output sample rate of 4Hz (see Sec. 6.4). This information is used to modulate the musical output of the Music Generation Engine (MGE). The MMMI is designed to allow the participant to concentrate on thinking *musical thoughts*, while the MGE responds to these musical thought processes in a natural way. The music creation process is transparent, in the sense that the musical thoughts of the participant *directly* affect the musical output of the system. No other musical control interface (such as an instrument or control panel) is required. (see Chp. 8)



---

# Real-Time System Design

---

## 5.1 Real-Time Feedback in the MMMI

Music-creation interfaces can be designed for real-time or for offline use. A guitar, for instance, is a real time interface. The musical processes associated with playing the guitar occur as it is being played; musical thoughts and ideas can be expressed instantaneously. There is also direct feedback: the result of playing a string is audible in the (very small) time it takes for the sound to reach the ear. Many software sequencers, on the other hand, are designed for offline music creation. The musical stimulus is mapped out ahead of time by the musician, and played back upon request. The musical creativity takes place ahead of the performance, during playback the computer is simply following instructions.

This issue is further complicated when written music is played by a musician on an instrument. The composer of the music certainly exercised the musical processing power of his brain in writing the music, and this was all done prior to the music being played. However, the performer of the music is also utilising the musical power of her own brain in playing the music, by playing the instrument, and in adding feeling and expression to the performance. In improvisation, these two steps are combined; the music is conceived and played almost simultaneously by the musician.

It is important to remember what type of musical processing we are detecting in the Phase Synchrony Engine (PSE). According to [31], the increased phase synchrony was detected in musicians while *attentively listening* to the musical stimulus. What we require, therefore, is to have the MMMI participants listen attentively to the musical output of the MGE. Whether this information is used instantaneously or offline depends on the objective of the interface.

As an example of an offline application, a musician's attentive musical processes could be logged during a MMMI operation, and afterwards compared with the musical stimulus. Such an experiment may provide interesting insights into the way that certain characteristics of the music affect the musician's attention. Indeed, the relationship between the variation in the music, and the ebb and flow of the listener's musical attention, is not well understood. This question has implications for composers, who are seeking to write music which connects with the listener on some level.

However, the stated focus of the MMMI is on the real-time interaction between the

musical stimulus and the musical attention of the musician. We therefore require output of the PSE to affect the MGE output straight away. This decision has implications for the design of the MMMI. In particular, all the components only have a finite amount of time to do their ‘work’, otherwise some components of the system will begin to lag behind others, and this de-synchronisation has unacceptable consequences for the MMMI as a real-time interface.

The primary job of the MAI server is to maintain synchronous communication amongst the separate processing modules of the MMMI. The operation of each system component requires a non-zero amount of time, and this introduces latency into the system. This latency may be caused by the processing time required in calculations, the time taken for packets to traverse the network, or necessary delays introduced by the processing algorithms themselves. In designing a real-time system, these latencies must be anticipated, and accommodations must be made to ensure the system remains synchronised and on time.

*Hard real-time* describes the requirement that certain processes occur at a specified absolute time. In the MMMI, however, this is not strictly necessary. What *is* necessary in the MMMI is synchronisation. If, for example, the PSE becomes desynchronised to the EEG input stream, then the phase synchronisation in the EEG data stream may become impossible to detect. The various components must be aware of their operation *relative* to the other components in the system, so that synchronous communication may be maintained.

## 5.2 Simulink and the Phase Synchrony Engine

The PSE is written in the Simulink modelling language, which is part of the MATLAB package by the Mathworks [47]. As mentioned in Sec. 4.4.3, Simulink is a graphical, block-based environment for matrix manipulation and simulation of dynamic systems. Low level component ‘blocks’ can be connected together to emulate the structure of complex systems. Such a collection of blocks is known as a *model*, and this is the standard filetype for Simulink. Given appropriate initial conditions, these models can then be run, and the results of the simulation can be analysed. Simulink allows users to test and analyse complex systems without having to physically build them. It is widely used in the fields of digital signal processing and control engineering [52]. The Simulink environment is also used in the Plymouth musical BCI project, the BCMI [25].

One advantage of Simulink is the wide variety of add-on ‘toolboxes’ available for the software. These toolboxes add a variety of domain specific blocks to those in the standard library. The *Signal Processing Blockset* provides signal processing functionality, such as the FFT and digital filtering, to the Simulink package. This blockset is utilised heavily in the PSE.

The obvious disadvantage with using Simulink in the PSE is the cost: Simulink is a commercial product, and, particularly once the necessary add-ons are purchased, the licence is expensive. For this reason, implementing the PSE algorithms in a low level language such as C++ (the same as the MAI server) was considered. However, expressive power of Simulink in DSP applications provides benefit’s which justify the

---

expense. The choice of Simulink greatly reduced the development time of the PSE, allowing it to reach maturity in the short period of the Honours year.

### 5.3 Real-Time Simulink

The Simulink environment was initially developed as an environment for offline simulation. While the various toolboxes and add-ons extend its functionality to other problem domains, a *simulation oriented* design approach is still evident in some aspects of Simulink operation. In particular, the Simulink model in use has no concept of absolute time; it simply executes as fast as the hardware will allow. In a simulation environment, this is ideal; simulating the operation of a complex system for 24 hours of simulation time, for example, may only take five minutes of real time. This is one of the benefits of running such simulations, as the long term behaviour of a system can be determined far sooner than is possible by actually building the system and monitoring its behaviour over time.

This becomes a problem, however, when Simulink is dependent on data from an external source, such as is the case in the MMMI. If the hardware running the Simulink model is too powerful, the Simulink model will ‘run ahead’ of the data input, resulting in a buffer underrun. Similarly, if the hardware is not powerful enough, then the Simulink model will not be able to keep up with the data inflow, and a buffer overflow will result. While, in theory, if the hardware is perfectly matched to the task at hand, the devices may stay synchronised, it is very difficult to achieve this in practice and it is an unacceptable solution to the MMMI problem.

As mentioned in Sec. 4.3.2, an attempt was made at achieving synchronous MMMI operation with Simulink, with no success. Simulink’s ignorance of absolute time proved too large a hurdle to synchronous operation. As the PSE is an integral component of the MMMI, a solution to the synchronisation problem was found in a Simulink add-on called xPC Target [53].

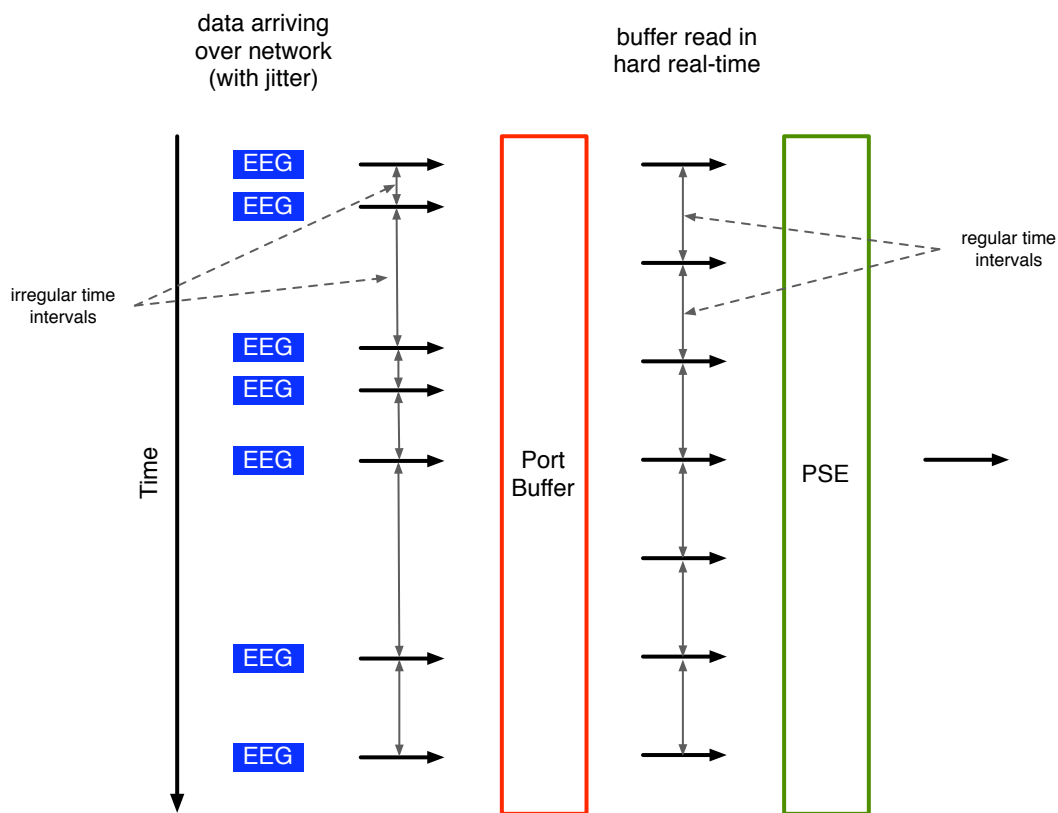
### 5.4 xPC Target

Simulink is by default an interpreted, rather than a compiled, language. Once the model is constructed, the simulation can be performed by clicking a button in the IDE. However, for increased performance, it is also possible to generate C code from a given model. This generated C code can also be generated for a specific hardware platform, including many embedded devices. The C code generation procedure for the PSE is performed with a Simulink add-on called the ‘xPC Target’ toolkit.

The xPC Target toolkit allows the creation of a boot disk containing a special, lightweight operating system developed by the Mathworks specifically designed for running Simulink models. Upon booting any x86 computer with this boot disk, the computer then becomes a ‘target’ box for the Simulink process running on a ‘host’ computer. Optimised C code for a given model is then built by the host computer and downloaded to the target computer via a network connection. The purpose of

this special xPC Target OS is to allow the execution of the Simulink model in *hard real-time*.

Using xPC target, then, the PSE would execute its processing at the known sample rate of the input. This prevents the PSE ‘running ahead of itself’ and causing the buffer underrun problems experienced in early tests. The PSE input uses a polling procedure to receive EEG datagrams from the Connection Kit (see Sec. 4.4.4). Since the input rate of the incoming data is known (the 2048Hz EEG signal), the PSE is set to perform its processing at the appropriate rate (see Fig. 5.1). Testing has shown no significant packet loss between the PSE and the rest of the M MMI when using this procedure.



**Figure 5.1:** The Simulink PSE polling the input port buffer in real-time

The completed PSE is a Simulink model consisting of approximately 200 blocks, which is reproduced in Appendix A. This is then compiled to C code and executed on a target PC running the xPC Target OS during M MMI operation. The host PC is an Intel Core 2 Duo 2.13GHz with 2GB RAM. The target computer, which actually runs the PSE, is an Intel Pentium 4 2.4GHz with 512MB RAM. This hardware setup has proved sufficiently powerful to run the PSE in real-time.

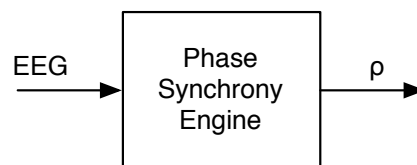
---

# Phase Synchrony Engine Design

---

## 6.1 Signal Input and Initial Processing

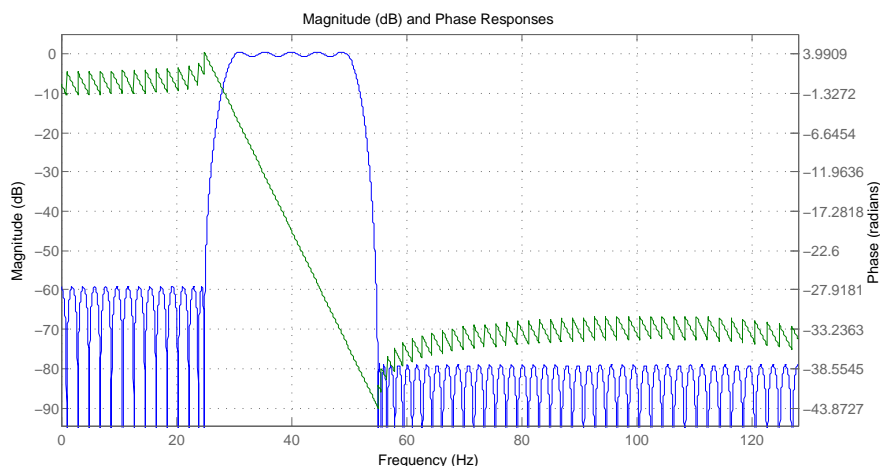
The Phase Synchrony Engine takes, as input, 16 channels of EEG activity at a sampling frequency of 2048Hz. By comparing the degree of phase synchrony between all the EEG channels, the PSE obtains a global (that is, over the whole mesh of electrodes) measure of phase synchrony in the neural signal, denoted by  $\rho$  (see figure 6.1). This measure is normalised to be between 0 and 1, with a value of 0 corresponding to no phase synchrony in the neural signal and a value of 1 representing totally phase coupled EEG channels.



**Figure 6.1:** Information flow through the PSE

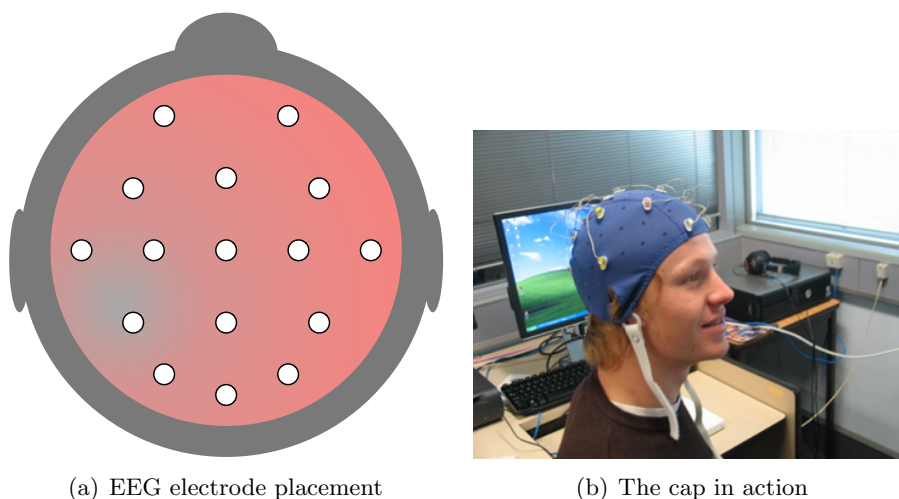
As a first processing step, the PSE uses a 30Hz - 50Hz bandpass filter to isolate the gamma band activity in the EEG signal, which is the frequency band of interest in measuring the musical processing in the brain. The filter used is a FIR Equiripple bandpass filter ( $\omega_{c1} = 30Hz, \omega_{c2} = 50Hz$ ) of order 130. The magnitude and phase response of the filter are shown in Fig. 6.2. This filter was chosen because of its linear phase response, and therefore the group delay is constant. The constant group delay minimises the phase distortion caused by the filtering process. The filter merely induces a delay on a narrowband signal. The important relative phase relationships in the signal are preserved.

The raw EEG signal is downsampled after filtering, and the bandpass filter also acts as an anti-aliasing filter. The sampling theorem then dictates that the largest resolvable frequency in a sampled signal is equal to half the sampling frequency. To represent the signal components up to 50Hz, the sampling rate for the neural signal must be at least  $2 \times 50 = 100Hz$ . The EEG input sample rate of 2048Hz far exceeds the requirements of the system, and this unnecessarily high sample rate increases the cost of the processing in the PSE. The EEG input signal is therefore downsampled to



**Figure 6.2:** Gamma-band filter response

128Hz for processing in the PSE. The bandpass filter has significant attenuation at 64Hz, ensuring minimal aliasing distortion caused by the downsampling of the signal.



**Figure 6.3:** The Biosemi EEG cap

Each of the 16 EEG channels represents the signal from a given electrode in the EEG cap. The location of the EEG electrodes is shown in Fig. 6.3(a). In the PSE, we wish to detect the degree of phase synchrony between the various EEG channels. The higher the degree of phase synchrony between the channels, the greater the functional connectivity in the brain.

The PSE takes a window-based approach to measuring the amount of synchrony in the EEG signal. Each EEG channel is buffered into sequential length  $N$  signal windows as shown in Fig. 6.4. The phase synchrony between all the EEG channels is measured for *that* particular signal window. The output rate of the PSE is then governed by the window length

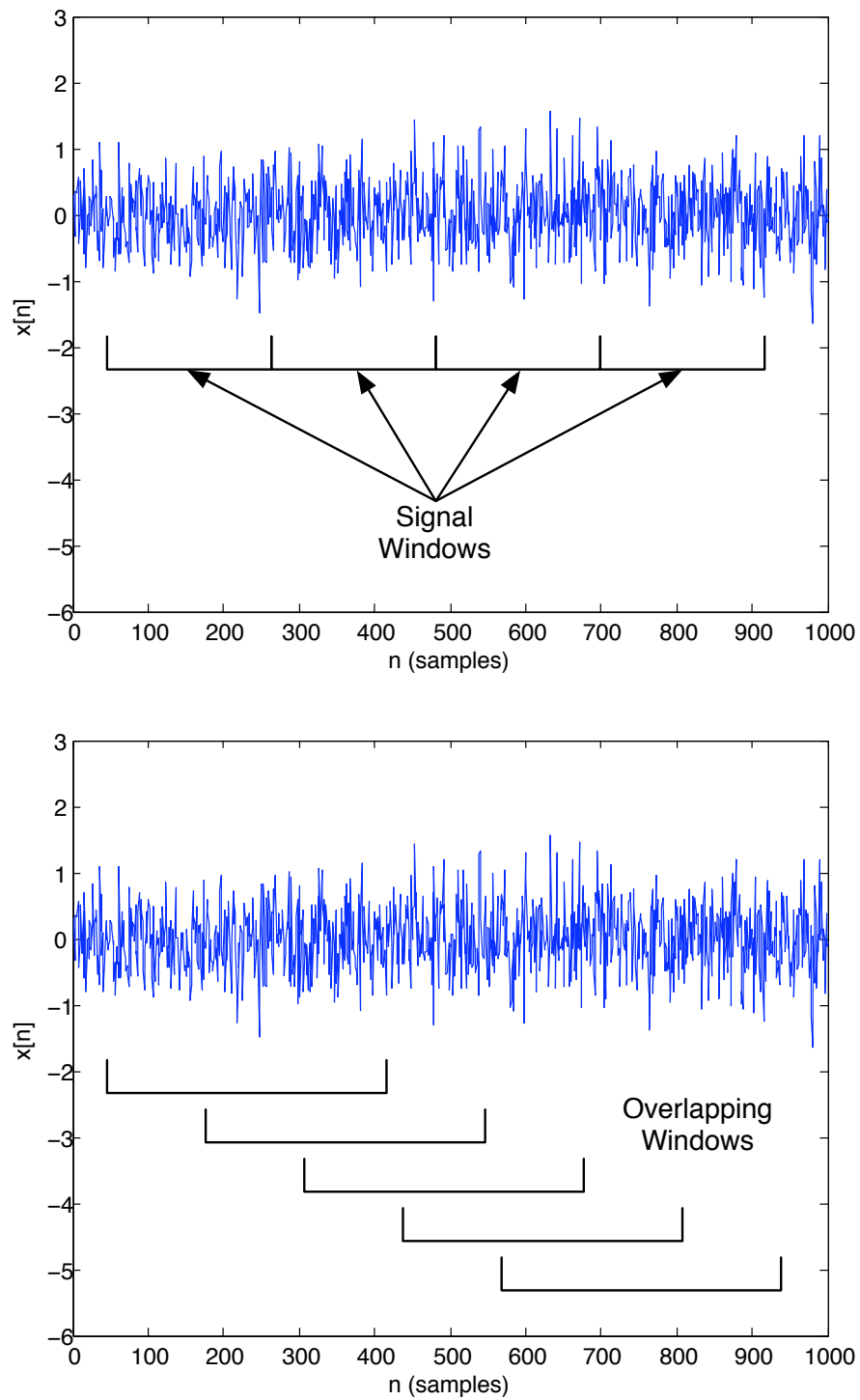


Figure 6.4: Each signal windows is processed individually by the PSE

$$\text{PSE output rate} = \frac{\text{EEG sample rate}}{\text{window length}}$$

These signal windows can also be ‘overlapped’, to increase the output rate of the PSE. The window overlap parameter (the number of consecutive windows a given sample appears in) can be tuned to allow real-time operation on a given hardware platform. The PSE uses length  $N = 128$  (corresponding to a window length of one second) signal windows with a  $4\times$  overlap, a configuration which has allowed real-time operation on the MMMI hardware.

There are two stages to the task of determining the phase relationship between two length  $N$  signal segments

1. The PSE must determine the phase evolution of the signal segments
2. The PSE must compare the phase of the two segments to determine the synchronisation degree

The next two sections discuss each of these stages in detail.

## 6.2 Phase Estimation

Consider the discrete time length  $N$  signal segment

$$x_{exp}[n] = e^{j\omega n} \quad \text{for } N = 0 \dots N - 1$$

$x_{exp}[n]$  is a complex exponential with frequency  $\omega$ . The value of  $x_{exp}[n]$  over one period with  $\omega = \pi/3$  is shown in Fig. 6.5. The phase of a complex sinusoid is well defined, and is given by

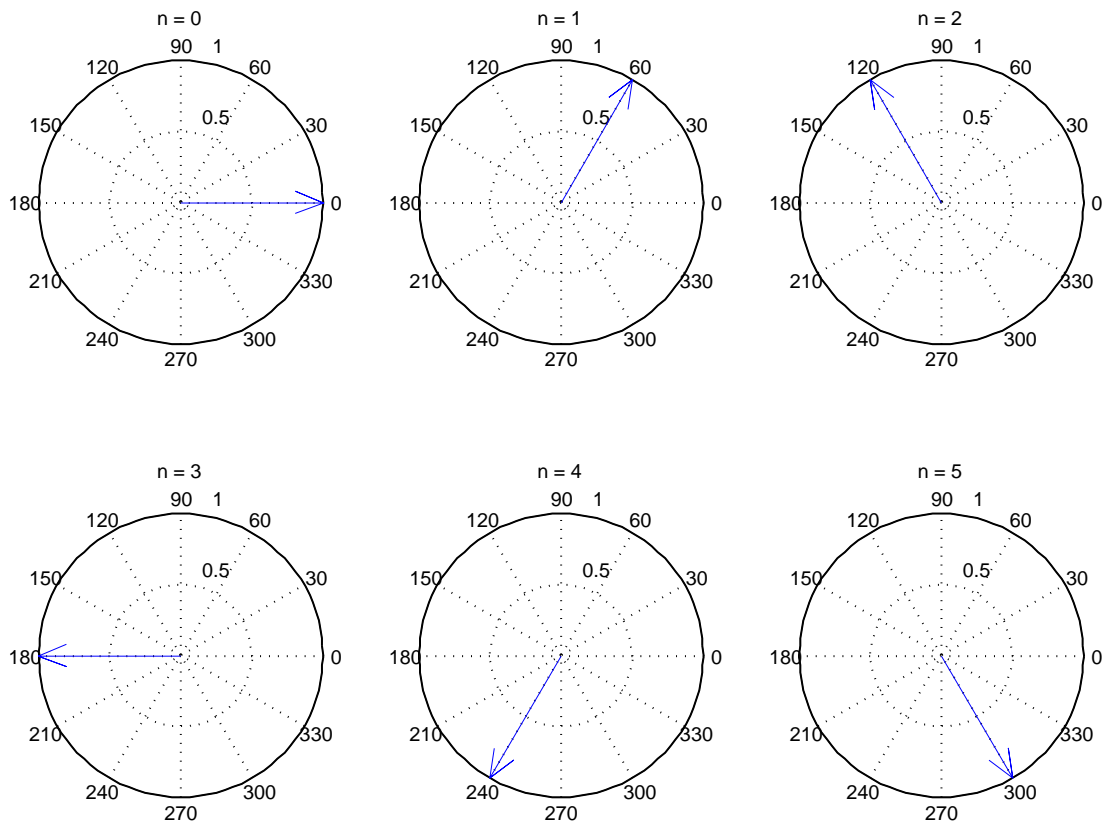
$$\begin{aligned} \phi_{exp}[n] &= \arg(x_{exp}[n]) \\ &= \arg(e^{j\omega n}) \\ &= \omega n \pmod{2\pi} \quad (0 \leq \phi[n] \leq 2\pi) \end{aligned} \quad (6.1)$$

Determining the phase of a *general* signal segment  $x[n]$ , however, is not so straightforward. For a non-periodic signal segment, such as the EEG segment in Fig. 6.6, it is not immediately clear what is meant by the phase of the signal. How, then do we determine the phase relationships between EEG signal segments in the PSE?

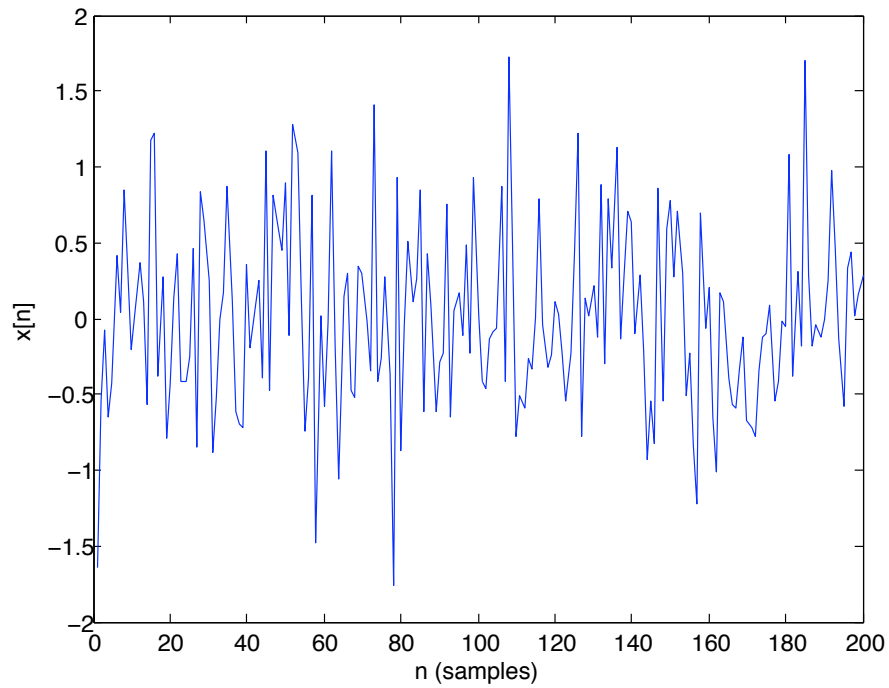
Any length  $N$  signal segment  $x[n]$  can be represented as a sum of complex exponential components using the Discrete Fourier Transform (DFT) synthesis equation [54]

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j(2\pi/N)kn} \quad (6.2)$$

Each of these complex exponential components has a well defined phase given by 6.1. However, if  $x[n]$  is real, as is the case with the EEG signal segments, then the DFT is



**Figure 6.5:** Phase of  $e^{j\frac{\pi n}{3}}$  (in degrees) for one period,  $n = 0, \dots, 5$



**Figure 6.6:** What is the phase of this signal?

conjugate symmetric, that is

$$\begin{aligned} X[k] &= X^*[((-k))_N] \\ &= X^*[N - k] \quad \text{for } 1 < k < N \end{aligned} \quad (6.3)$$

$$(6.4)$$

where  $((-k))_N$  denotes  $k$  modulo  $N$ . For a real signal, then, we can write the DFT equation 6.2 as

$$\begin{aligned} x[n] &= \frac{1}{N} \sum_{k=0}^{N/2-1} X[k]e^{j(2\pi/N)kn} + X[N - k]e^{j(2\pi/N)(N-k)n} \\ &= \frac{1}{N} \sum_{k=0}^{N/2-1} X[k]e^{j(2\pi/N)kn} + X^*[k]e^{-j(2\pi/N)kn} \end{aligned} \quad (6.5)$$

Each complex exponential term in the first sum in equation 6.5 is a complex conjugate of a term in the second sum. When these complex exponential terms are added together, the phase cancellation results in a purely real signal  $x[n]$ . The phase information associated with each of the complex exponential components is then lost, the phase of  $x[n]$  (being real) is either 0 (if  $x[n] > 0$ ),  $-\pi$  (if  $x[n] < 0$ ) or undefined (if  $x[n] = 0$ ).

One approach to this ‘phase masking’ problem involves the construction of the so-called *analytic signal*. For a given real signal  $x[n]$  the associated analytic signal  $z[n]$  is defined such that

$$\text{Re}\{z[n]\} = x[n], \quad Z[k] = 0 \quad \text{for } N/2 < k < N \quad (6.6)$$

In continuous-time, the analytic signal  $z(t)$  is so named because it is an *analytic* function of the variable  $t$ . Analytic functions have other nice properties such as continuous differentiability, and both real and complex analytic signals are well understood (see, for example, [55]). In discrete-time, however, many of these properties are no longer well defined, however the name analytic signal is still used to express the connection to the continuous-time case.

Since  $Z[k]$  is zero for  $N/2 < k < N$ ,  $Z[k]$  is only conjugate symmetric in the trivial case  $Z[k] = 0 \quad \forall k$ . The analytic signal  $z[n]$  is therefore, in general, complex. We can then write

$$z[n] = z_r[n] + jz_i[n] \quad (6.7)$$

where  $z_r[n]$  and  $z_i[n]$  are real and represent the real and imaginary parts, respectively, of the analytic signal. Now, by the linearity of the DFT,

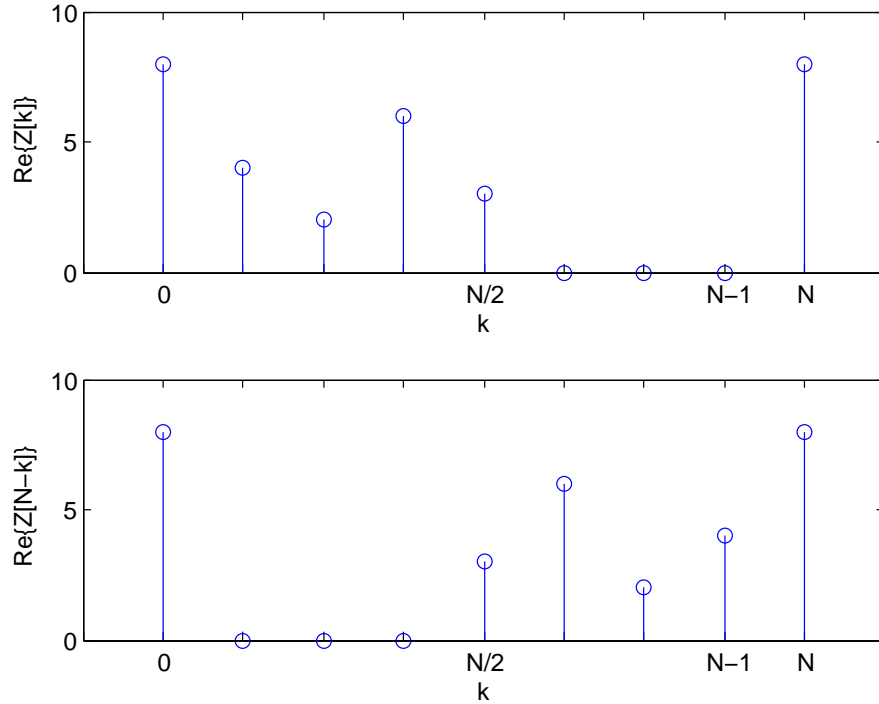
$$\begin{aligned} Z[k] &= Z_r[k] + jZ_i[k] \\ &= X[k] + jZ_i[k] \end{aligned} \quad (6.8)$$

where in 6.8 we have used the fact that  $Z_r[k]$  is the DFT of the real part of  $z[n]$ , which is equal to  $x[n]$  by construction. Furthermore, since  $z_r[n]$  is real, by the symmetry

property (6.3)

$$Z_r[k] = X[k] = \frac{1}{2} [Z[k] + Z^*[N - k]] \quad (6.9)$$

Since  $Z[k] = 0$  for  $N/2 < k < N$ ,  $Z[k]$  and  $Z^*[N - k]$  only overlap at  $k = 0$  and



**Figure 6.7:** The DFT overlap between  $Z[k]$  and  $Z^*[((-k))_N]$

$k = N/2$ . This is shown graphically in Fig. 6.7, although for clarity only the real part of  $Z[k]$  is shown. Furthermore,  $Z[k] = Z^*[N - k]$  for  $k = 0$  and  $k = N/2$ .  $Z[k]$  can therefore be completely recovered from 6.9

$$X[k] = \frac{1}{2} [Z[k] + Z[0]\delta[n] + Z[N/2]\delta[n - N/2]] \quad (6.10)$$

Because  $x[n]$  is known, we can evaluate  $X[k]$  using the DFT analysis equation

$$X[k] = \sum_{n=0}^{N-1} x[n]e^{-j(2\pi/N)kn} \quad (6.11)$$

Rearranging 6.10, we can express  $Z[k]$  in terms of the known quantity  $X[k]$

$$Z[k] = \begin{cases} X[k], & \text{for } k = 0, N/2 \\ 2X[k], & \text{for } 0 < k < N/2 \\ 0, & \text{for } N/2 < k < N \end{cases} \quad (6.12)$$

From  $Z[k]$ , we can construct the analytic signal  $z[n]$  using the DFT synthesis equa-

tion 6.2

$$z[n] = \frac{1}{N} \sum_{k=0}^{N-1} Z[k] e^{j(2\pi/N)kn}$$

By imposing the constraints 6.6 on the analytic signal, the relationship between  $Z[k]$  and  $X[k]$  is explicitly determined. This type of constraint-based relationship is called a *Hilbert Transform* relationship, and these relationships occur often in signal processing [56]. Computing the analytic signal  $z[n]$  in the PSE is therefore a three step process:

1. Calculate  $X[k]$  from  $x[n]$  using the DFT analysis equation 6.11
2. Construct  $Z[k]$  from  $X[k]$  using the relationship given in 6.12
3. Produce  $z[n]$  from  $Z[k]$  using the DFT synthesis equation 6.2

Each of these DFT evaluations can be implemented efficiently in Simulink using the Fast Fourier Transform (FFT). The PSE Simulink model is shown in Appendix A. The phase of the analytic signal is easily determined,

$$\phi[n] = \tan^{-1} \left( \frac{\text{Im}\{z[n]\}}{\text{Re}\{z[n]\}} \right) \quad (6.13)$$

Because the input signal  $x[n]$  is not necessarily periodic, the phase  $\phi[n]$  is meaningful only in the statistical sense. However, the phase information can be used effectively to quantify the degree of synchrony between between two signal segments, as we shall see in the next section.

### 6.3 Characterising Phase Synchrony

Once the phase evolution of a signal segment  $\phi[n]$  is determined, we wish to characterise the phase relationship between two signal segments  $x_a[n]$  and  $x_b[n]$  (with  $0 \leq n \leq N - 1$ ).

Firstly, we use the procedure described in the previous section to obtain the phase information  $\phi_a[n]$  and  $\phi_b[n]$ . To do this, we first compute the phase difference between the two signals over the whole signal window.

$$\phi_{a,b} = \text{ARG}(\phi_a[n] - \phi_b[n]) \quad (6.14)$$

where  $\text{ARG}(\theta)$  represents the principal value of the argument ( $-\pi < \text{ARG}(\theta) \leq \pi$ ). Two signals are phase-locked if they satisfy the condition

$$\phi_{a,b}[n] = \text{constant} \quad \forall n \quad (6.15)$$

Two signals  $x_a[n]$  and  $x_b[n]$  satisfying the condition 6.15 are said to be *synchronised* in phase. It is important to note that this condition places no constraints on the signal

amplitudes  $|x_a[n]|$  and  $|x_b[n]|$ . For noisy signals, this condition is weakened [57] to

$$c_1 < |\phi_{a,b}[n]| = c_2 \forall n \quad (c_1, c_2 \text{ constant}) \quad (6.16)$$

In the noisy case, the phase difference is no longer necessarily constant, but must fall in some range given by  $c_1$  and  $c_2$ .

In general, the two signals  $x_a[n]$  and  $x_b[n]$  may be perfectly phase coupled, completely phase independent, or they may lie somewhere in between (containing both phase-locked and independent components). The job of the PSE is to quantify the degree of phase synchrony between the signals. To do this, we examine the distribution of the absolute value of the phase difference  $\phi_{a,b}[n]$  over the whole signal window, specifically the set

$$\Phi_{a,b} = \{|\phi_{a,b}[n]| : 0 \leq n \leq N - 1\} \quad (6.17)$$

If the phases of the two signals are correlated, then the distribution of phase difference values in the set  $\Phi_{a,b}$  will show a strong peak around the absolute value of the phase lag between the two signals. If the phases of the two signals are uncorrelated, their difference will be distributed uniformly.

One measure of the ‘spread’ of  $\Phi_{a,b}$  is given by the Shannon information entropy [58]. To calculate the entropy, we must first estimate the underlying distribution of  $\Phi_{a,b}$ . The simplest way to do this is to divide the range of  $\Phi_{a,b}$  (that is, 0 to  $\pi$ ) into  $P$  bins, and the probability  $p_i$  associated with each bin is given by the proportion of  $\Phi_{a,b}$  values falling in the  $i$ th bin

$$p_i = \frac{|\{\theta \in \Phi_{a,b} : \frac{(i-1)\pi}{P} \leq \theta < \frac{i\pi}{P}\}|}{N} \quad (6.18)$$

In the PSE,  $P = 2 * \log N$  bins are used, which is the same as in Bhattacharya & Petsche [31]. The entropy of the distribution is then given by

$$H(\Phi_{a,b}) = - \sum_{i=1}^P p_i \ln(p_i) \quad (6.19)$$

The entropy value is normalised to a value between 0 (representing no phase synchrony) and 1 (representing perfect phase synchronisation) to give the synchronisation index  $\rho_{a,b}$

$$\rho_{a,b} = \frac{E_{max} - H(\Phi_{a,b})}{E_{max}} \quad (6.20)$$

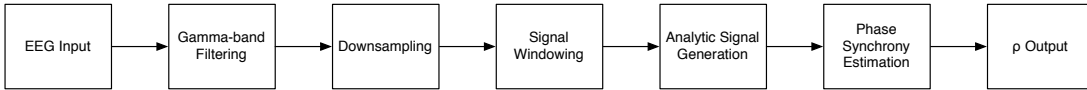
where  $E_{max} = \ln(P)$  is the maximum entropy of the distribution.

In the PSE, we wish to calculate an *overall* measure of phase synchrony in the brain. The procedure described up to this point is only suitable for determining the phase synchrony index  $\rho_{a,b}$  between two EEG signal epochs  $x_a[n]$  and  $x_b[n]$ . The overall synchronisation index  $\rho$  is given by the average pairwise synchronisation index over all

possible EEG channel pairs, that is

$$\rho = \frac{\sum_{a=1}^{16} \sum_{b=a+1}^{16} \rho_{a,b}}{120} \quad (6.21)$$

The quantity  $\rho$  is then an overall index of the degree of phase synchronisation in the EEG signal, which was the desired output of the PSE. The complete process is shown in Fig. 6.8.



**Figure 6.8:** The components of the PSE algorithm

## 6.4 Algorithmic Complexity

As mentioned in Sec. 5.1, it is necessary for the PSE to calculate this neural synchrony index  $\rho$  in real time. As the length  $N$  signal segments are processed by the PSE algorithm, this processing must be completed before the next signal segment arrives for processing. If the PSE processing is not completed in the available time step the PSE will ‘fall behind’ and will not be able to catch up, resulting in a buffer overflow.

The PSE operates on one signal window at a time, and each window contains 16 length  $N$  signal segments, one for each EEG channel. For a given window, the most computationally expensive parts of the PSE algorithm are the FFT (equation 6.2) and the pairwise entropy calculation (equation 6.21). The time complexity of the FFT algorithm is [59]

$$O(N \log N) \quad (6.22)$$

The number of different pairs of EEG channels can be seen as the handshake problem in disguise. There are the same number of distinct EEG channel pairs as there are possible unique handshakes in a group of people. For  $M$  EEG channels, the number of pairs of channel windows which must be compared to estimate the synchrony index is therefore [60]

$$\frac{M(M-1)}{2}, \text{ which is } O(M^2) \quad (6.23)$$

Comparing the complexity expressions 6.22 and 6.23, it is clear that the cost of the FFT will ultimately dominate if  $M < \log N$ , while the synchrony estimation will dominate if  $M > \log N$ . However, given the practical nature of the PSE system, we are interested not only in the asymptotic behaviour of the algorithm, but also in the absolute time complexity for given values of  $N$  and  $M$ . The exact multiplicative constants associated with (6.22) and (6.23) depend on the implementation of the algorithms on the PSE hardware, and as such are difficult to compute. In the PSE, the number of EEG channels  $M$  is fixed at 16, and a one second signal window dictates a value of  $N = 128$  due to the downsampled input signal at 128Hz.

---

The other factor to consider in the performance of the PSE is the amount of signal window overlap. The PSE calculates one  $\rho$  value per window. By increasing the amount of overlap, the output frequency of the PSE is increased, at the expense of an increase in the number of numerical operations per second. This window overlap parameter can be used to tune the PSE for real-time performance. In the PSE, with  $M$  and  $N$  fixed as described above, a  $4\times$  window overlap has provided real-time performance in the MMMI. The output sample rate of the PSE is the product of the window length and the window overlap factor. The output sample rate is therefore  $1 \times 4 = 4\text{Hz}$ .



---

# Phase Synchrony Engine Testing

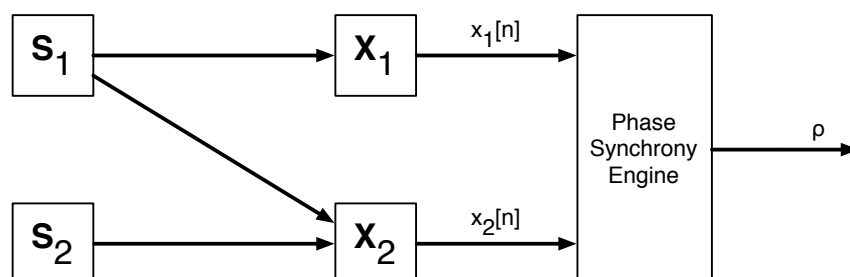
---

## 7.1 Testing with Artificial Data

Using the procedure described in Chp. 6, the PSE provides a measure  $\rho$  of the degree of synchrony in the input EEG signal. Verifying the accuracy of the PSE is difficult, because it is not known apriori what the value of  $\rho$  should be. To test the PSE, it is necessary to construct an artificial signal with phase synchrony characteristics which *are* known, and can be compared to the output of the PSE.

This validation approach is adopted in a 2006 paper published in Physical Review Letters, entitled *Quantitative evaluation of linear and nonlinear methods characterizing interdependencies between brain signals* [41]. In this paper (which shall hereafter be referred to as the *QE* paper), several approaches to the phase synchrony detection problem in the literature were quantitatively compared using artificial test data. Furthermore, one of the approaches tested uses a similar technique to the PSE to detect neural synchrony (we shall refer to this technique as PRL for Physical Review Letters). The PRL performance on the *QE* artificial test data is given in the paper, and these results provide meaningful benchmark data for the PSE. While the testing in *QE* was performed offline, a comparison with the PSE is an excellent test of the validity of the real-time approach adopted in the PSE.

## 7.2 Phase Coupled Systems



**Figure 7.1:** Test data system structure

The test data used in *QE* and which we shall use to test the PSE is based on a

simple, two-source system (see Fig. 7.1). Two sources,  $S_1$  and  $S_2$  generate data with independent phase profiles. These sources are detected by two sensors,  $X_1$  and  $X_2$ , producing signals  $x_1[n]$  and  $x_2[n]$ . The output of source  $S_1$  is present in both  $x_1[n]$  and  $x_2[n]$ , while the output of source  $S_2$  is present in  $x_2[n]$  only. The influences shown in Fig. 7.1 do not necessarily indicate simple linear mixing of the two sources at  $X_2$ , merely that the signal  $x_2[n]$  is influenced by both  $S_1$  and  $S_2$ . The influence of each component is governed by the coupling parameter  $c$ .

There are many ways to model this influence mathematically. In one model used in *QE*, the parameter  $c$  determines the degree of phase synchronisation between two generated narrowband signals  $x_1[n]$  and  $x_2[n]$ . Four lowpass filtered (cutoff frequency 0.1Hz) white noises  $F_1$ ,  $F_2$ ,  $F_3$  and  $F_4$  are used to generate the signals

$$x_1[n] = A_1 \cos(2\pi f_0 n + \phi_1) \quad (7.1)$$

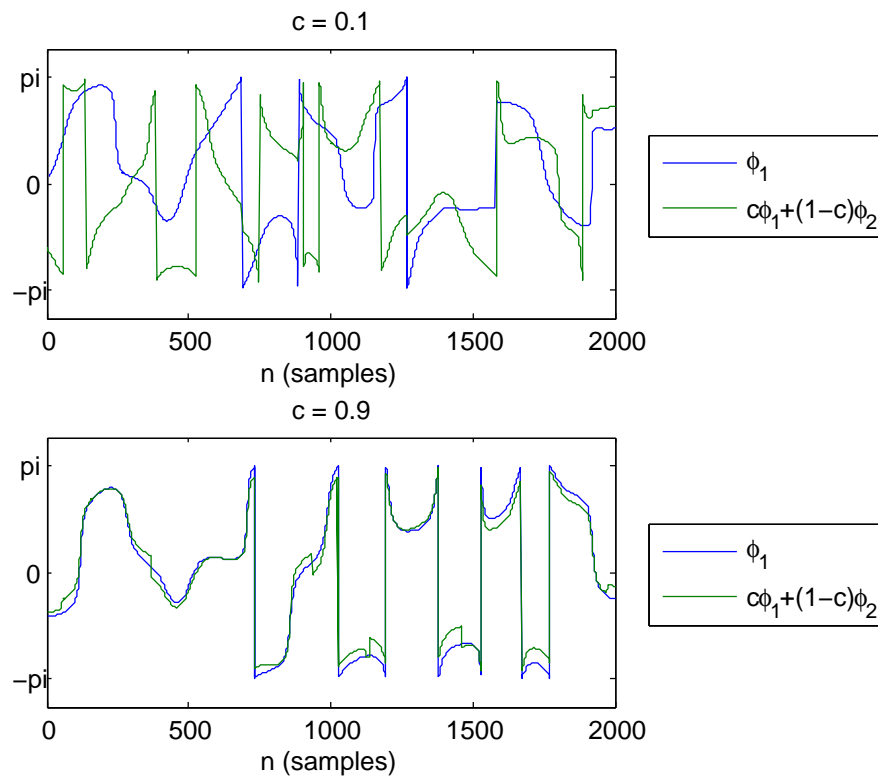
$$x_2[n] = A_2 \cos(2\pi f_0 n + c_1 \phi_1 + (1 - c_1) \phi_2) \quad (7.2)$$

where

$$\begin{aligned} A_1 &= \sqrt{F_1^2 + F_2^2} \\ A_2 &= \sqrt{F_3^2 + F_4^2} \\ \phi_1 &= \tan^{-1} \left( \frac{F_1}{F_2} \right) \\ \phi_2 &= \tan^{-1} \left( \frac{F_3}{F_4} \right) \end{aligned}$$

The generated narrowband signals have a fundamental frequency centered at  $f_0$ . In testing the PSE, a value of  $f_0 = 40\text{Hz}$  was used so that the generated signal would lie in the 30Hz - 50Hz gamma frequency band (the frequency band to which the PSE is sensitive). The phase relationship between the generated signals is dependent on the parameter  $c$  in equation 7.2. When  $c = 0$ , the phase of the two signals  $x_1[n]$  and  $x_2[n]$  is completely determined by the *independent* quantities  $\phi_1$  and  $\phi_2$  respectively. When  $c = 1$ , the phase of *both* signals is determined by  $\phi_1$ , and the phase is the same for both signals. When  $0 < c < 1$ , the phase of  $x_2[n]$  is only partially determined by the phase of  $x_1[n]$ . By varying the parameter  $c$ , we can vary the degree of phase synchrony between the two signals as shown in Fig. 7.2. Notice that the quantities  $A_1$  and  $A_2$ , which govern the amplitudes of  $x_1[n]$  and  $x_2[n]$ , are independent. This is important for measuring functional connectivity in the brain: we wish to detect a phase relationship between two signals even if their amplitude is uncorrelated [61].

It is important to note that this test model is not physiologically accurate. The system described in Fig. 7.1 and in equations (7.1) and (7.2) is not designed to accurately model the neural activity associated with functional connectivity in the brain. It is designed, rather, to allow the generation of signals with a *known* phase relationship, and this is extremely useful for testing phase synchrony detection methods such as the one used in the PSE. The lack of a neurological basis for the model does not diminish



**Figure 7.2:** The effect of the coupling parameter  $c$  on the phase terms for  $c = 0.1$  (weak synchrony) and  $c = 0.9$  (strong synchrony)

it's usefulness as a test mechanism [62].

### 7.3 Quantitative Evaluation of the Phase Synchrony Engine

The test procedure used in *QE* involves generating paired signals  $x_1[n]$  and  $x_2[n]$  with equations (7.1) and (7.2). The sampling frequency of the generated signals is 2048Hz, the same as the EEG input sampling frequency (see Sec. 6.1). The generated signals are each ten minutes (600 seconds) in length. Multiple pairs of test signals are generated with coupling degree  $c$  ranging from 0 to 1 in increments of 0.1. Each pair (for a fixed  $c$ ) of signals is processed by the PSE in it's entirety. For a 600 second long input signal, the PSE will process  $600 \times 4 = 2400$   $\rho$  values (one second windows with a  $4 \times$  overlap). The mean and variance of these output  $\rho$  values are then are then calculated.

As mentioned in Sec. 7.1, a 'phase difference entropy' based phase synchrony detection technique (PRL) similar to the one used in the PSE was also tested in the QE study, and the results are provided in the paper. The performance of the PRL systems is used as a benchmark for the PSE in determine if the PSE generates correct output.

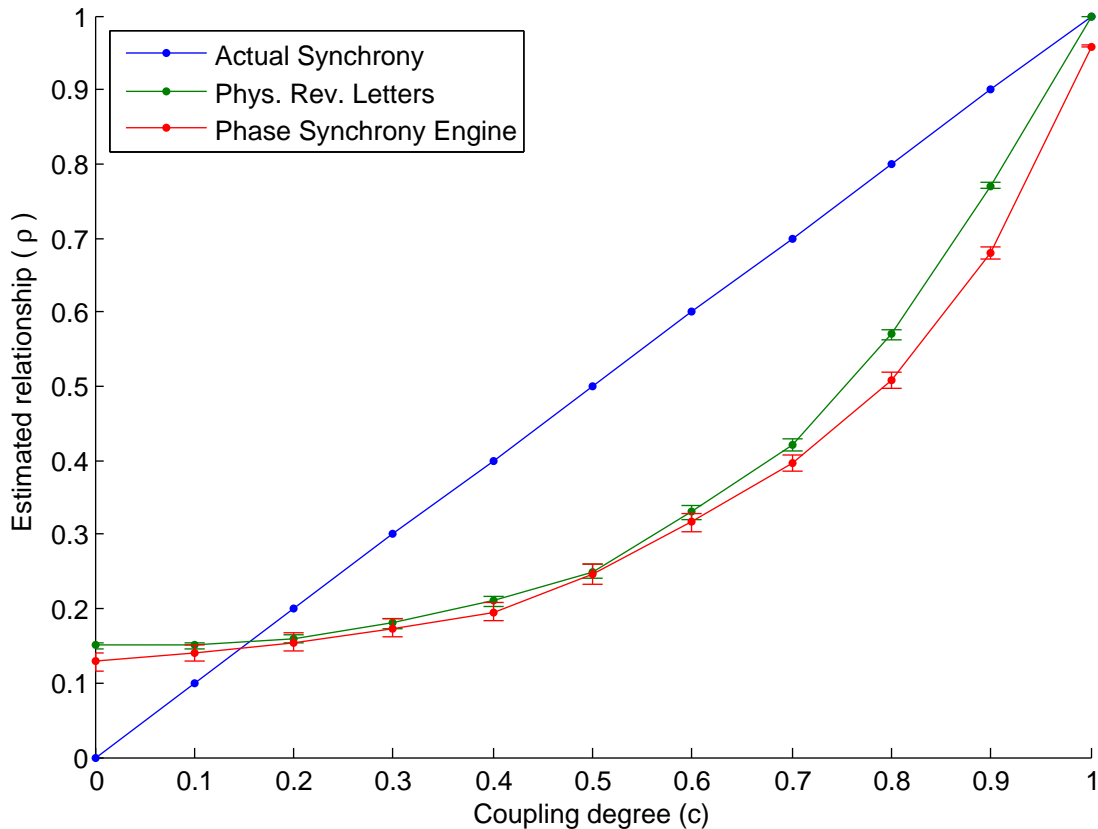


Figure 7.3: Artificial test data results

The performance of the ideal phase synchrony detection system ( $\rho = c$ ) is shown

as a blue line. The performance of the PSE is plotted in red, while the performance of the PRL method used in *QE* is plotted in green. The standard error over the ten minute test signal is also shown.

Clearly, both systems fall short of the ideal performance. For independent signals ( $c = 0$ ), both the PSE and PRL systems overestimate the degree of synchrony between the test signals ( $\rho > c$ ). As  $c$  increases beyond 0.2, both systems begin to underestimate the degree of synchrony between the signals ( $\rho < c$ ). The response of both the PSE and PRL is monotonic, so an increase in phase coupling  $c$  cannot lead to a decrease in detected synchrony  $\rho$  or vice versa. This is important because it shows that as  $c$  varies, both the PSE and PRL will accurately detect the *direction* of the change, even if the amount of change may have some error.

For  $0.1 \leq c \leq 0.6$ , the response of both the PSE and PRL is equal to within the standard error in the calculation. As  $c$  increases past 0.6, the PSE becomes less sensitive than the PRL. The probable reason for this is due to the phase distortion introduced in the gamma-band filtering stage of the PSE (see Sec. 6.1). The group delay of the filter introduces a phase-lag in the phase information extracted by the PSE. As  $c$  increases, the phase difference changes more quickly, and a greater change in phase can occur in the fixed time lag introduced by the filter. The PRL system does not suffer from this problem, because the system is interested in *any* phase-synchronous activity rather than in a specific frequency band, so the bandpass filtering step is not required.

The results in Fig. 7.3 are pleasing for the PSE, they are acceptably close to the benchmark PRL results from the *QE* paper. For use in a real-time music generation interface such as the MMMI it is the changes in phase synchrony over time which we are interested in, and the PSE is well behaved with regard to changes in the phase synchrony parameter  $c$ .

## 7.4 Further Testing

The artificial test data given by equations (7.1) and (7.2) can also be used to test individual components of the PSE algorithm. To test the analytic signal generation procedure, we can examine if the constraints (6.6) are satisfied. By (6.6), the input signal  $x[n]$  and the real part of the generated analytic signal  $z[n]$  should be equal. In Fig. 7.4 their absolute difference (plotted in the bottom axes) is shown for a 500 sample period, and is less than  $3 \times 10^{-16}$  for the entire period. If this difference is considered the *processing noise* introduced by the PSE, then the signal-to-noise ratio is approximately  $10^{15}$ , an excellent result.

The other half of the constraint (6.6) is the requirement that  $Z[k] = 0$  for  $N/2 < k < N$ . The magnitude FFT of the original signal ( $X[k]$ ) and the analytic signal ( $Z[k]$ ) is shown in Fig. 7.5. Again, the negative frequency components ( $-\pi < \omega < 0$ ) of  $Z[k]$  are clearly removed, while the positive frequency components ( $0 < \omega < \pi$ ) are unaffected. The analytic signal conditions (6.6) appear to be fulfilled. These tests further support the validity of the PSE algorithm.

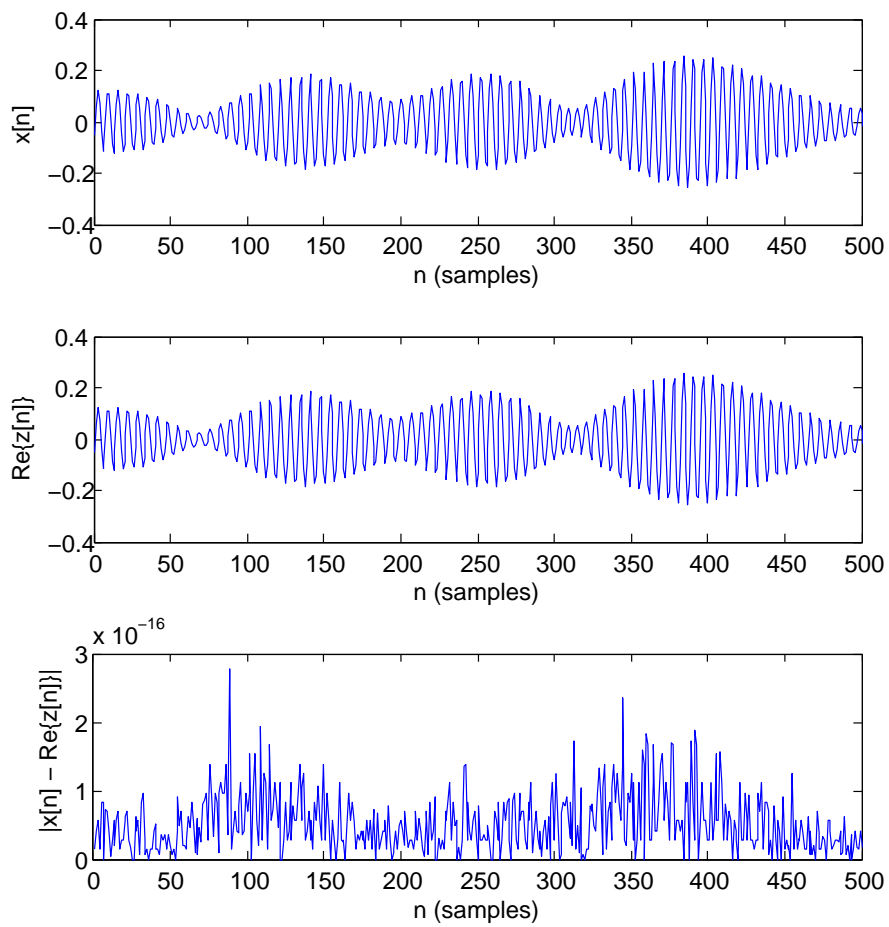


Figure 7.4: Testing the validity of the generated analytic signal

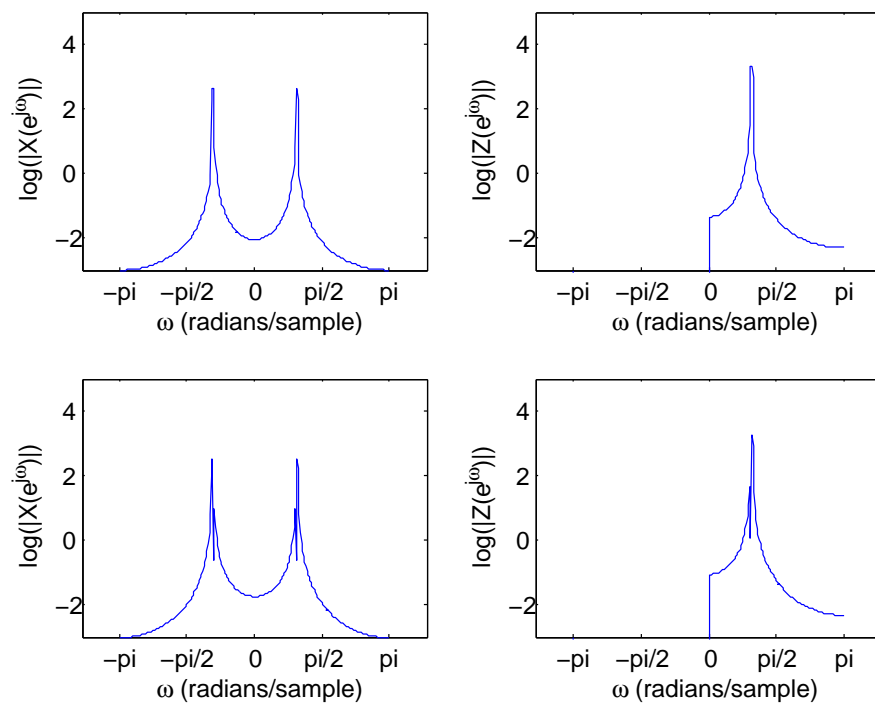


Figure 7.5: Log magnitude FFT for original and analytic signals



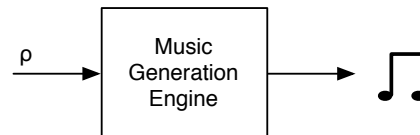
---

# Music Generation

---

## 8.1 Music Generation Environment

Generating music based on the output of the PSE is a very open-ended task, and a great deal of musical domain knowledge can be used. The generated music must be musically interesting to engage the participant’s attention. The MGE output is therefore not deterministic, and stochastic processes are utilised extensively to ensure the stimulus is constantly changing, even if  $\rho$  remains constant. The MGE is, in essence, a set of rules which govern this stochastic music generation (Fig. 8.1). Designing these rules is an artistic endeavour, because the rules ultimately dictate the artistic merit of the generated output.



**Figure 8.1:** Information flow through the MGE

One of the key goals of the MMMI is for the musician to *feel* that their musical attention affects the musical stimulus in some meaningful way. The musical output of the MGE must therefore vary along some characteristic dimension. This dimension can then be parameterised by the index of neural musical processing,  $\rho$ . In the MMMI, the MGE output varies in *note density*, that is, the number of musical events (notes) per unit of time. Note density has been shown to influence the amount of *tension* perceived in music [63]. Note density is a relatively simple concept to encode in a stochastic music generation engine, the likelihood of a note being played can be simply made proportional to  $\rho$ . For a fixed value of  $\rho$  the actual note density of the output will vary with time, but the *expected value* of the note density remains constant. Exactly how the parameter  $\rho$  influences the note density of the music is described in Sec. 8.2

The MGE has been implemented in the *impromptu* environment developed by Andrew Sorensen [64]. Impromptu is an interactive programming environment for real-time music creation based on the Scheme programming language, a Lisp dialect. Musical structure is defined in accordance with the MIDI protocol [65], where individual notes are well defined events, with associated pitch, duration and loudness

values. *impromptu* also features accurate temporal scheduling, allowing note events to be scheduled for playback at the appropriate time.

As a functional language, Scheme is naturally suited to recursion. *impromptu* allows *temporal recursion*, in which the callback time of a recursive function can be specified. A function can be defined to, say, schedule one bar of music for playback. After the notes have been scheduled, the function can then *schedule itself* to be called again after a delay of one bar. The function, upon callback, will then schedule the notes of the bar again before rescheduling itself for callback again. This technique makes *impromptu* ideal for generating repetitive music, such as dance music. Of course, the bar of music to be played can also be changed before callback, allowing *impromptu* to produce dynamic, time-varying music as well.

A MIDI stream from *impromptu* contains musical control information only, it is not an audio signal. MIDI control information must be processed by a synthesiser, which turns the MIDI information into sound. The synthesisers can be either hardware or software based, and in the MGE the synthesiser used is the Logic Studio package by Apple® [66]. The synthesiser output is played back to the MMMI participant in the sound cuboid (see Fig. 4.3).

## 8.2 Musical Aesthetics

The MGE output consists of four different instrumental *agents*: a Drum Agent, Bass Agent, Keyboard Agent and Lead Agent. Each of these musical *agents* is played back using a synthesiser sound designed to emulate the corresponding real instrument (drums, bass guitar, keyboard or lead guitar). The MGE generates music on a bar-by-bar basis, scheduling one bar of music at a time. Each instrument has their bar generated in advance, and *impromptu* then schedules and plays back the bar through the Logic Synthesiser. The bars are designed to be somewhat persistent; each instrument does not completely change every bar. In fact, the likelihood of any instrument changing their bar to be played is proportional to  $\rho$ . Low values of  $\rho$  decrease the likelihood of a given instrument changing the bar of music to be played. This results in music which is repetitive, and evolves slowly over time. High values of  $\rho$ , on the other hand, cause the musical output of each instrument to change more frequently. The musical output of a high  $\rho$  system is therefore much less stable, and the musical output is much more dynamic. This also provides a clear difference in the generated music along the  $\rho$  continuum.

There are three different aesthetic aspects to the musical output of the MGE.

- The **Rhythmic** structure of the musical output
- The **Harmonic** structure of the musical output
- The **Dynamic** structure of the musical output

The aesthetic aspects of the generated music are governed by the behaviour of the different musical agents. The inter-agent interaction has been designed to resemble

the dynamics of a conventional four-piece band. The music generation algorithm used in the MGE allows the agents to influence each other in the three structural aspects described above (this influence is shown in Fig. 8.2). This co-operation allows the agents to produce coherent musical stimulus.

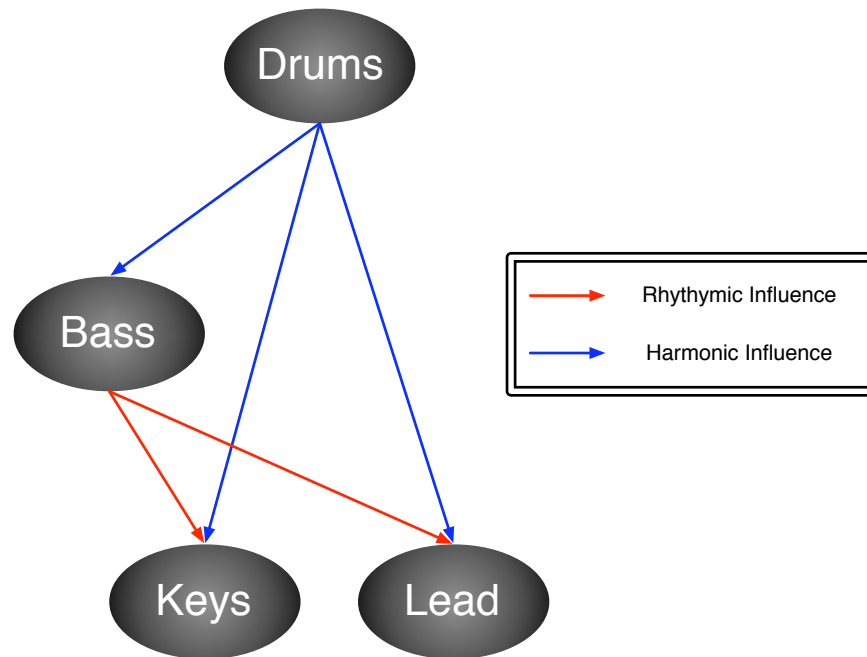
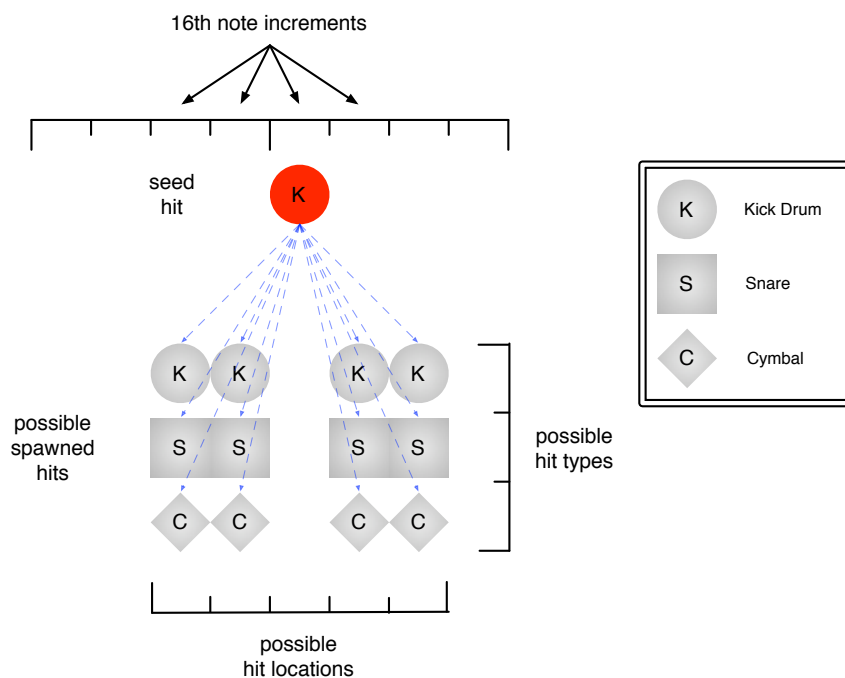


Figure 8.2: Aesthetic influences in the MGE

### 8.2.1 Rhythmic and Dynamic Structure

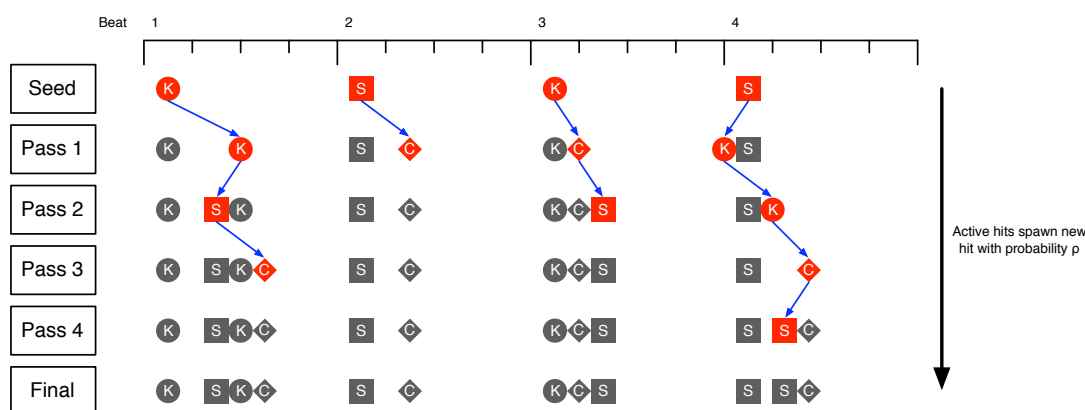
The musical output of the PSE is generated one musical bar at a time. The rhythmic structure of the music concerns the temporal relationships between the notes of the bar. As in a real band, the Drum Agent governs the rhythmic structure of the bar for all the MGE agents.

Each bar of drums generated by the Drum Agent is a sequence of drum hits, known as a *groove*. The hits can be either a kick drum hit, a snare hit or a cymbal hit. The hits can occur in 16th note steps, that is, 16 hits to a bar. In generating a new groove, the Drum Agent starts with a set of hits which constitute a basic groove. This initial groove is then iteratively built upon to generate more complex grooves. In this iterative process, the Drum agent makes successive passes over the groove. Each time, existing hits (seeds) in the bar spawn new hits with probability  $\rho$ . The spawned hits will occur close to the seed hit, and is necessarily of the same hit type (the possible options are shown in Fig. 8.3). Spawned hits can then spawn further new hits in successive iterations. Any hit which spawns a new hit is called an *active* hit. The hit remains active until it fails to spawn a new hit for the first time, it then becomes *inactive*. The hit spawning process continues until there are no remaining active hits. The whole process is shown in Fig. 8.4. Dynamically, the loudness of a spawned hit is also less



**Figure 8.3:** An active hit may spawn a new hit

than its seed hit. The initial groove hits will therefore be the loudest, which provides the musical output with a musical ‘pulse’.



**Figure 8.4:** The groove creation process

The number of hits in the final groove (which governs the note density of the musical output) is proportional to  $\rho$ . If  $\rho$  is close to 1, the likelihood of new hits being spawned is high, and the groove will therefore be more complex. The groove generated by the Drum Agent also provides the rhythmic structure for the other musical agents, so the note density of the entire ensemble is governed by  $\rho$ .

### 8.2.2 Harmonic Structure

The harmonic structure of the bar is determined by the groove generated by the Drum Agent. The Bass, Keyboard and Lead Agents, however, are tuned instruments. The timing of the notes to be played is dictated by the groove, but the pitch of the notes is unspecified. If the tuned instruments do not play in tune, the musical output of the system will be dischordant and jarring. It is necessary to define a harmonic structure to shape the output of the tuned instruments. The structure used in the MGE is called the *tonnetz* (a subset of the tonnetz is shown in Fig. 8.5).

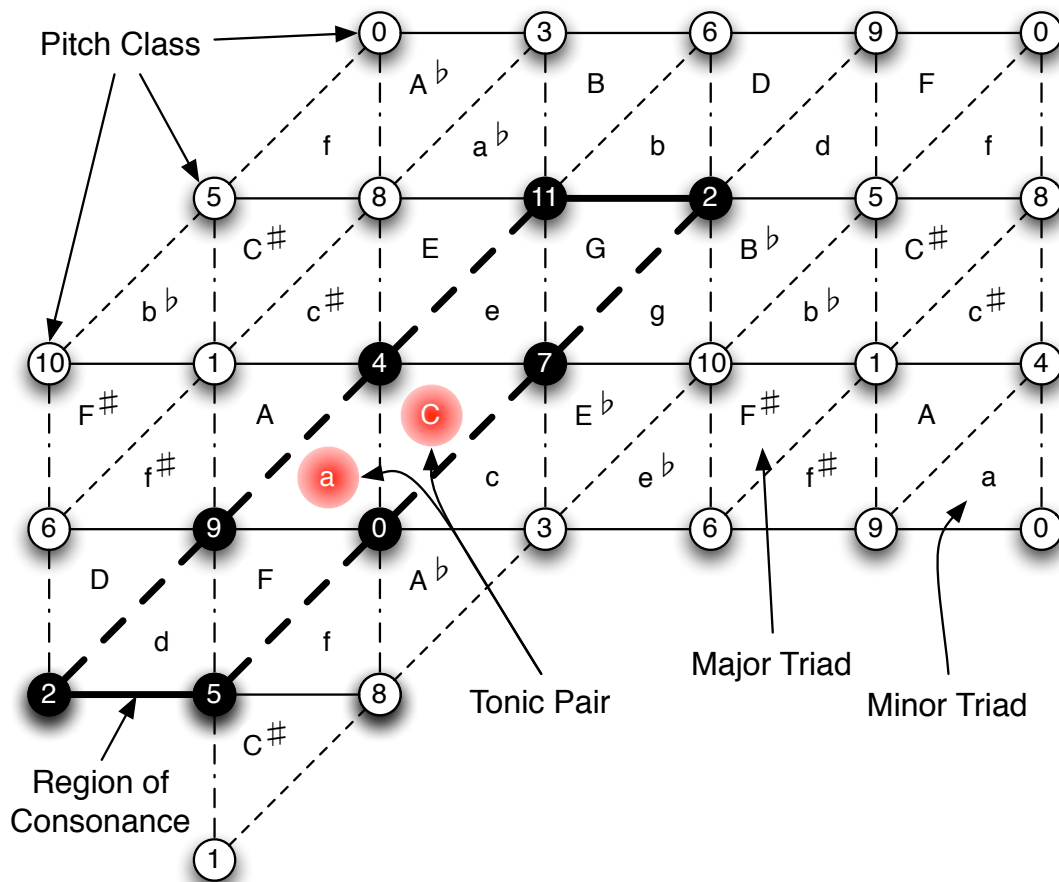


Figure 8.5: The tonnetz

Western musical theory defines a set of musical relationships between notes which are accepted as ‘musically pleasing’ [67]. This theory forms the basis for the harmonic relationships present in the output of the MGE. The tonnetz is a way of representing the 12 tones of the western musical scale first proposed by Euler in 1739 [68]. The notes of the scale are arranged in a lattice, while musical chords are represented as the vertices of triangles in the tonnetz. In the tonnetz, these acceptable relationships are represented by subsets of the tonnetz structure. These regions form a basis for the acceptable musical notes played by the musical agents in the MGE.

The core harmonic concept in the MGE is the *tonic* chord pair. For a given tonic pair (shown red in Fig. 8.5), the acceptable notes to play lie in a well defined ‘region of consonance’. For a given tonic, any notes from the associated region of consonance will be acceptable [69].

In the MGE, the Bass Agent determines the tonic chord for a given bar. When a new bar is generated, the Bass Agent changes the tonic with probability  $\rho$ . If the tonic is changed, the new tonic is determined by a random walk in the tonnetz. When the tonic is determined, so also is the region of consonance for the Keyboard and Lead Agents. This effectively defines a set of acceptable notes from which the notes to be played in the current bar can be selected.

For a given bar, the three tuned instruments play a note corresponding to each hit in the groove produced by the Drum Agent. The timing and loudness of each note is inherited from the hit in the drum groove. The pitch is selected with a random walk through the acceptable pitches for the given tonic. The notes selected are guaranteed to be pleasant-sounding by the structure of the tonnetz. The note density of the tuned instruments is the same as the drum groove, which is in turn dependent on  $\rho$ .

The note density of the output is the most noticeable difference in the musical output for different  $\rho$  values. When  $\rho$  is close to 1 (which indicates a large amount of musical processing), the music is frantic and rhythmically complex. When  $\rho$  is close to zero (indicating a low amount of musical processing), the music is subdued and sparse. The musician can, therefore, control the complexity of the music merely by attentively listening to it. The musical *responsiveness* of the system is one of the major contributions of the MMMI to the field of music creation.

---

# Status and Future Work

---

## 9.1 MMMI Status

The MMMI is ready to be used to create music. All of the components have been shown to work properly, and real-time operation has been achieved. In informal testing, the musical output of the MMMI has proved to be dynamic and engaging. The MMMI is ready to allow musicians to interact with their music in a way which has previously been impossible.

The MMMI is also in the process of being presented to the wider scientific community. A paper based on the MMMI, *Mind-Modulated Music in the Mind Attention Interface* [70], has been accepted for presentation at the 2007 Australasian Conference on Computer-Human Interaction. The theme for the conference is *Entertaining User Interfaces*. The author is attending the conference in Adelaide in November to present his work. The MMMI also features in a paper submitted for the 2008 IEEE Virtual Reality Conference in Nevada, USA [71].

## 9.2 Future Work

### 9.2.1 User Testing

With the construction of the MMMI complete, the next stage is to test the experience of musicians using the interface. Unfortunately, a planned user study of the interface has not been performed due to time constraints. We wish to determine whether the participant *feels* as though their musical attention is affection the musical output of the system.

The next step in the development in the MMMI is to perform a user test. In the planned user test, subjects will use the MMMI for two ten minute sessions. In one of the sessions, the musical output of the system will be responsive to the musical processes in the participant's brain, as described in this thesis. In the other session, the musical output of the system will vary independently of the musical processes in the brain. The participant will simply be asked to determine which session was which.

The proposed test is deliberately simple. In a single session test, where the participant was simply asked whether they felt that the music was responsive or not, the participant may have been compelled to say that the musical output was responsive

even if it was not. Because musical perception is subjective by nature, the risk of such *false positives* would be great. In the proposed test, the participant knows that *one* of the sessions only was the responsive session, but not which one. The test is therefore less susceptible to false positives, and gives a clearer indication of the effectiveness of the MMMI. The planned test is due to be carried out in the new year.

### 9.2.2 Migration to Open-Source Software

As discussed in Chp. 5, the PSE is built using the proprietary Simulink package from the Mathworks. Because of this, some licensing issues have hindered the development of the MMMI, particularly towards the end of the Honours project. Planned user tests and further benchmarking have needed to be postponed due to the unavailability of the Simulink licence.

For this reason, it is desirable to rebuild the PSE using only open-source software tools. This would eliminate the licensing problems associated with Simulink development. As a further benefit, the PSE code would be able to be released to the wider research community, allowing neural synchrony processing to be easily implemented in other BCI systems.

### 9.2.3 Further PSE and MGE Development

Because the field of Brain-Computer Interfaces is so dynamic, the MMMI is also a work in progress. In the future, it would be advantageous to incorporate other signal processing techniques in parallel to the PSE. This could provide a richer and more complete picture of the elusive musical processes in the brain.

The MGE is a crucial part of the MMMI, but it is far from the only conceivable musical mapping for the parameter  $\rho$ . It is possible, due to the modular design of the MMMI, to insert a completely different algorithmic composition unit into the MMMI to be driven by the measures of functional connectivity from the PSE. This would allow musicians to ‘bring their own’ music generation algorithms, but interface with them directly via the MMMI. This is an interesting option to be explored in the future.

### 9.2.4 The MMMI as a Creative Tool

Finally, the author hopes to be able to use the MMMI as a musical tool for composition and performance. As an alternative outlet for musical creativity, the MMMI has the potential to provide a revolutionary new way to express the musical processing power of the brain. The prospect of interacting with music in new and interesting ways has driven the development of the MMMI thus far and will continue to influence it’s development in the future.

---

# Conclusion

---

The Mind-Modulated Music Interface shows promise. Phase synchrony in the neural signal has been shown to be an indicator of musical processing in the brain, and the tests described in this thesis show the Phase Synchrony Engine to be a reliable tool for detecting phase synchrony. The use of the MMMI as a compositional tool can now begin in earnest.

The persistence of musical expression throughout history suggests that the art of creating music will always be practised, and the question of how music will be created and experienced in the future is a fascinating one. In the MMMI, we have suggested one possible direction for the future of music creation. By directly including the mind of the musician in the music creation experience, the musical processing of the musician is harnessed in a way which is not possible in conventional musical instruments. Ultimately it is hoped that the MMMI can contribute not only to the way music is experienced, but also to our knowledge of the way music is processed in the brain.



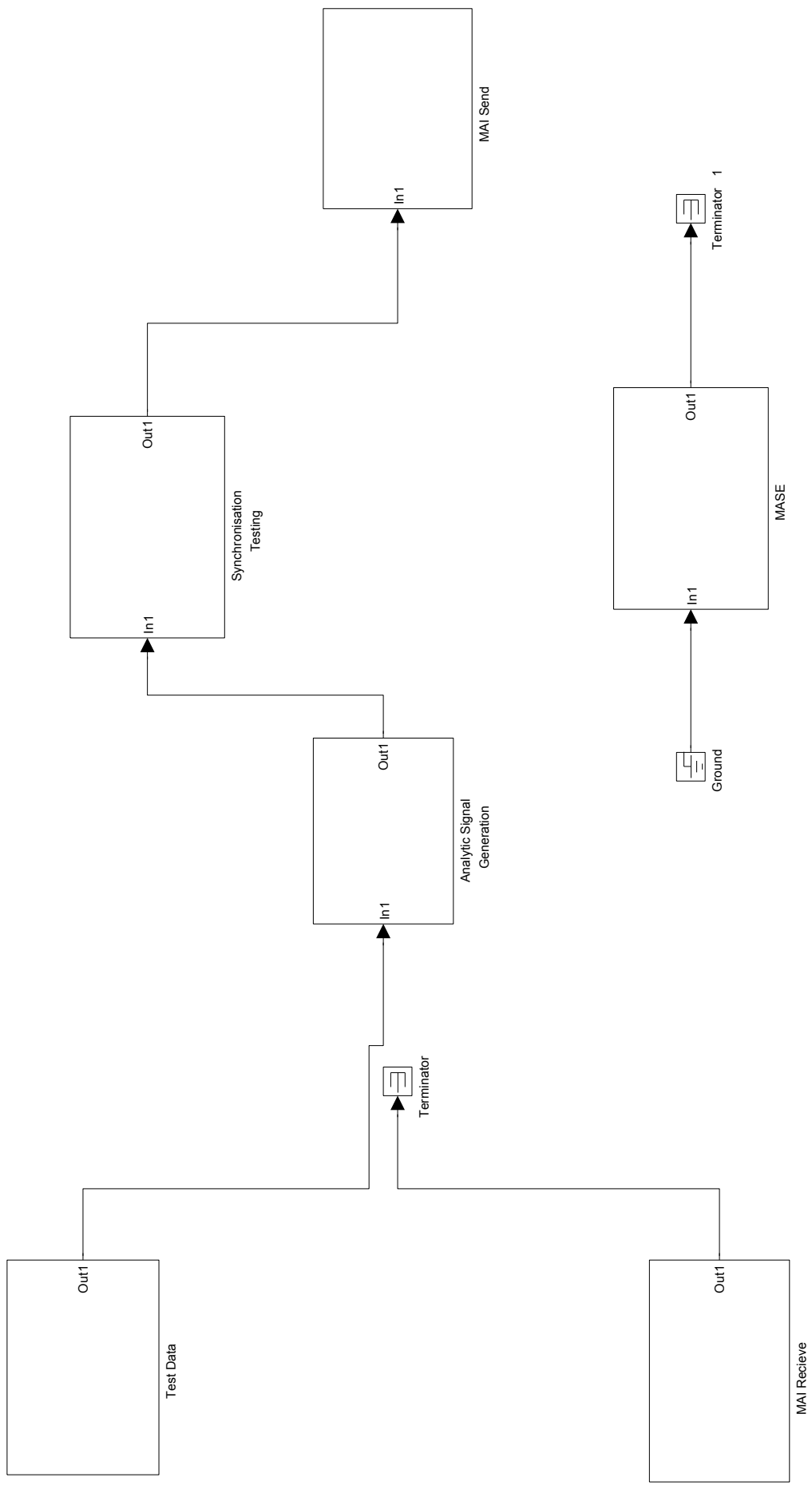
---

# Phase Synchrony Engine Block Diagram

---

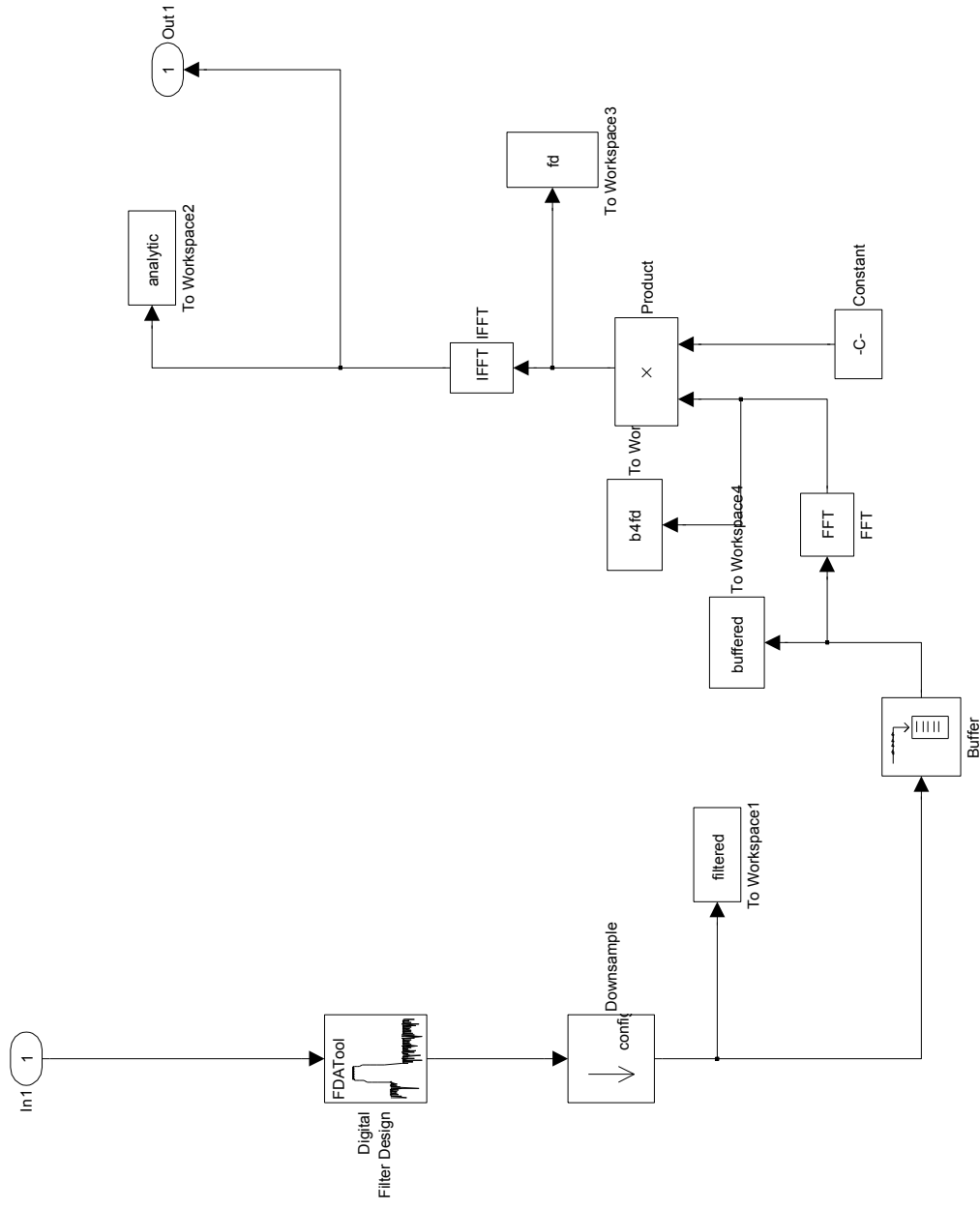
The following pages contain a printout of the complete PSE as a Simulink model. The model has several layers of abstraction, and each subsystem is shown on a separate page.

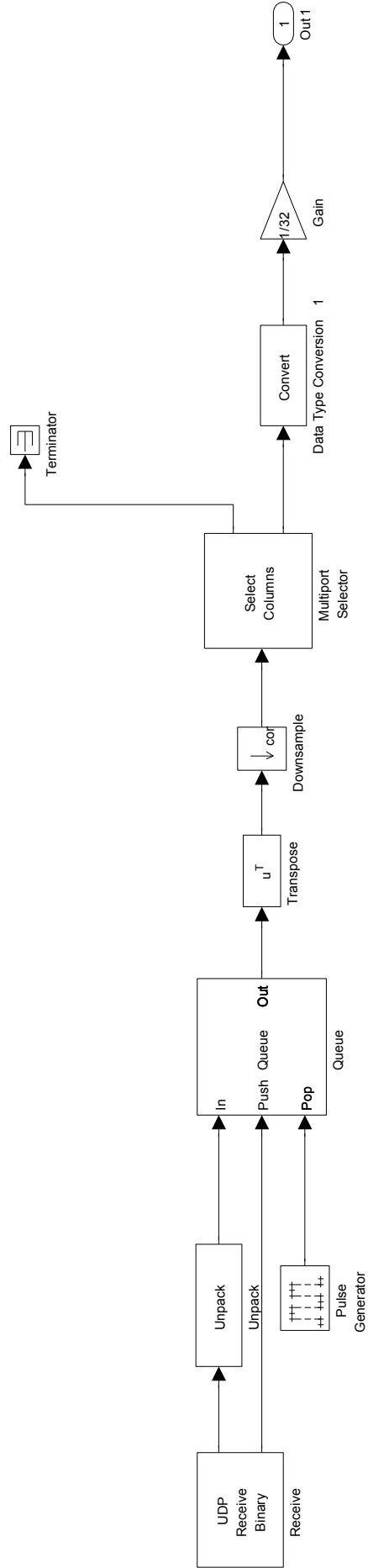
PSE



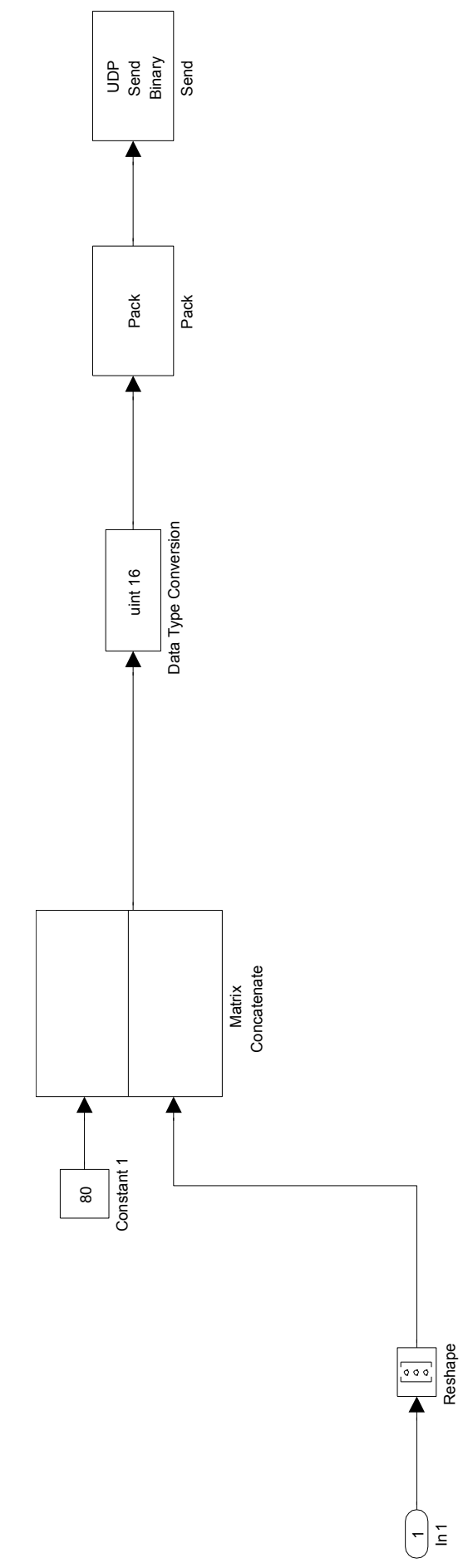
C:\Program Files\MATLAB\R2007 a\work\pcaise\IPSE.mdl

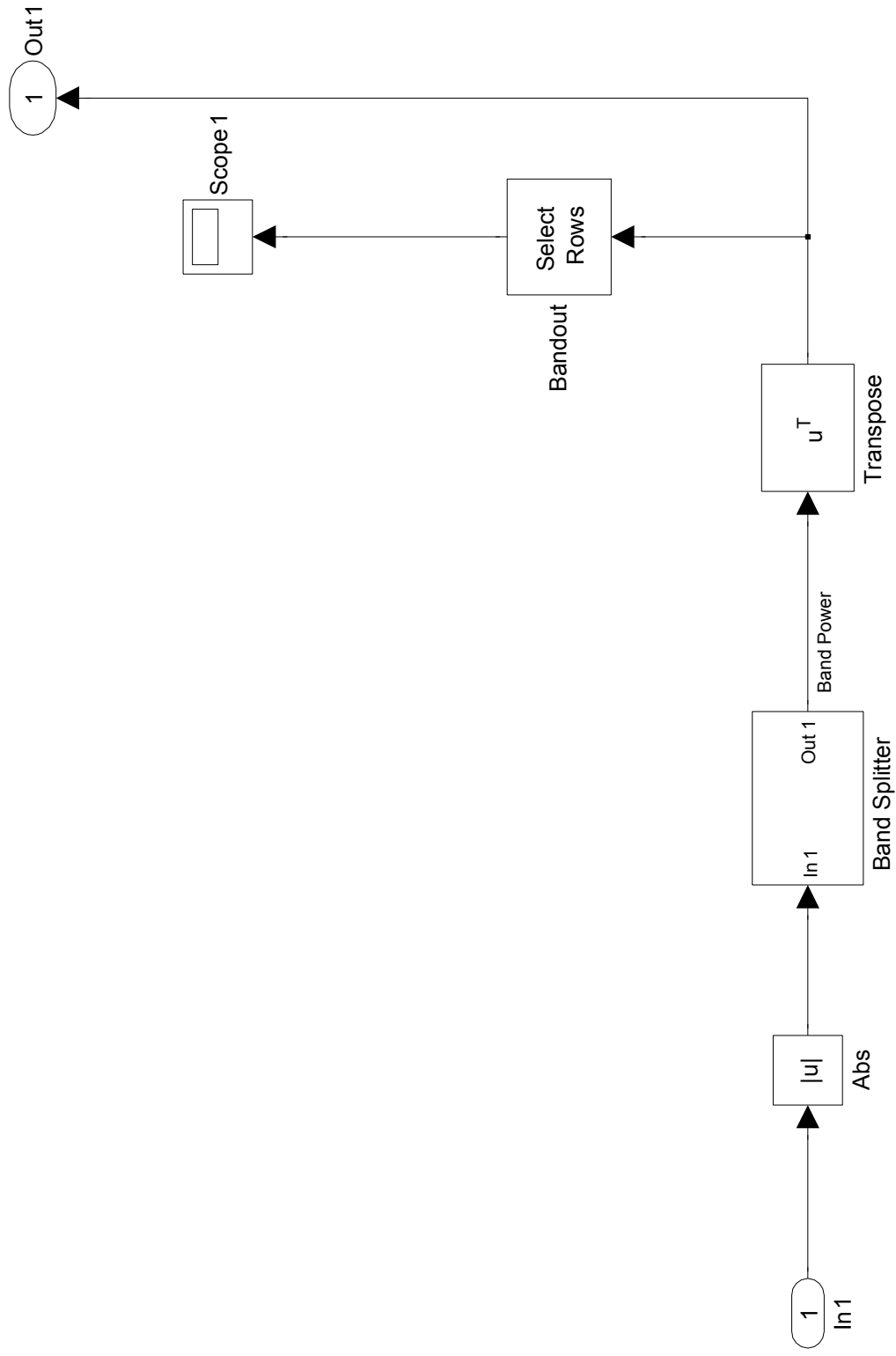
# PSE/Analytic Signal Generation



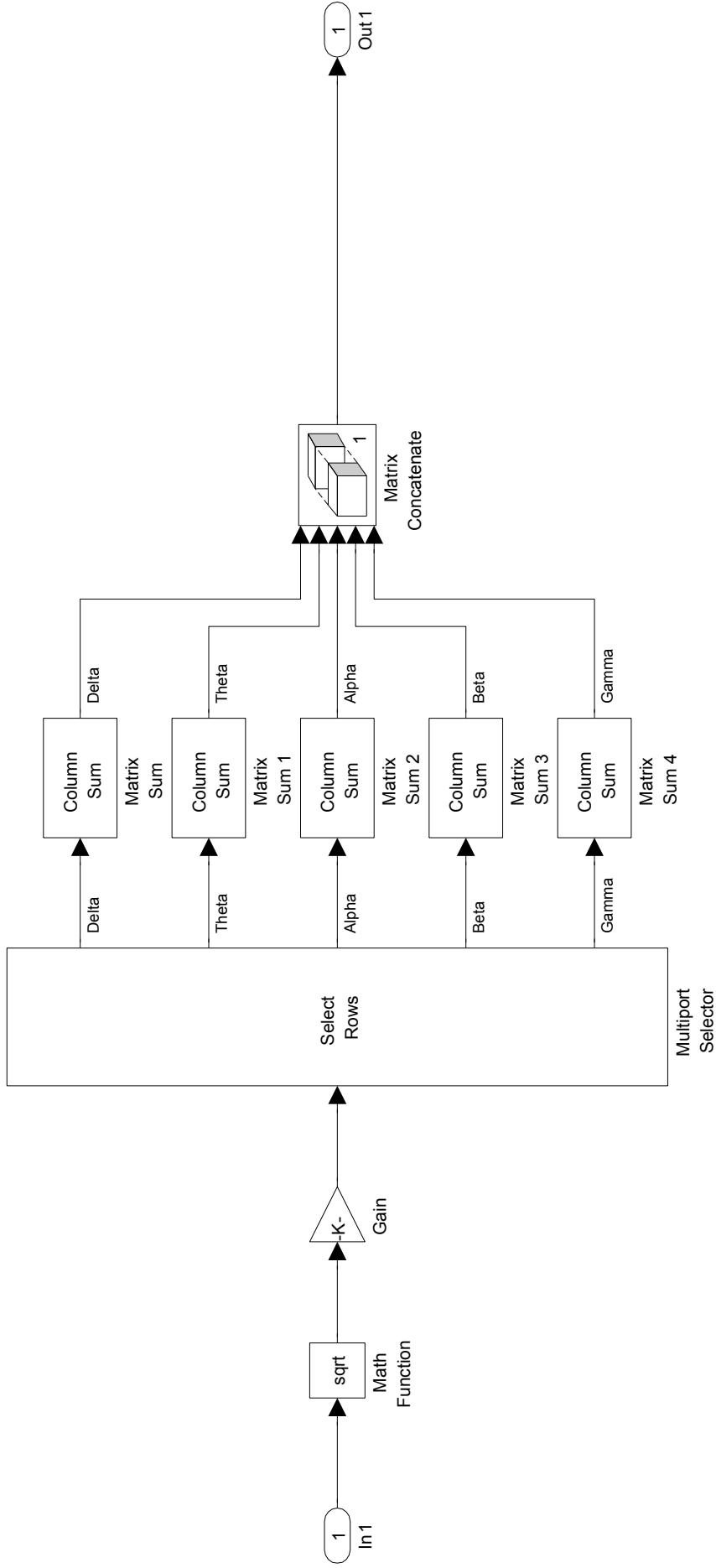


PSE/MAI Send

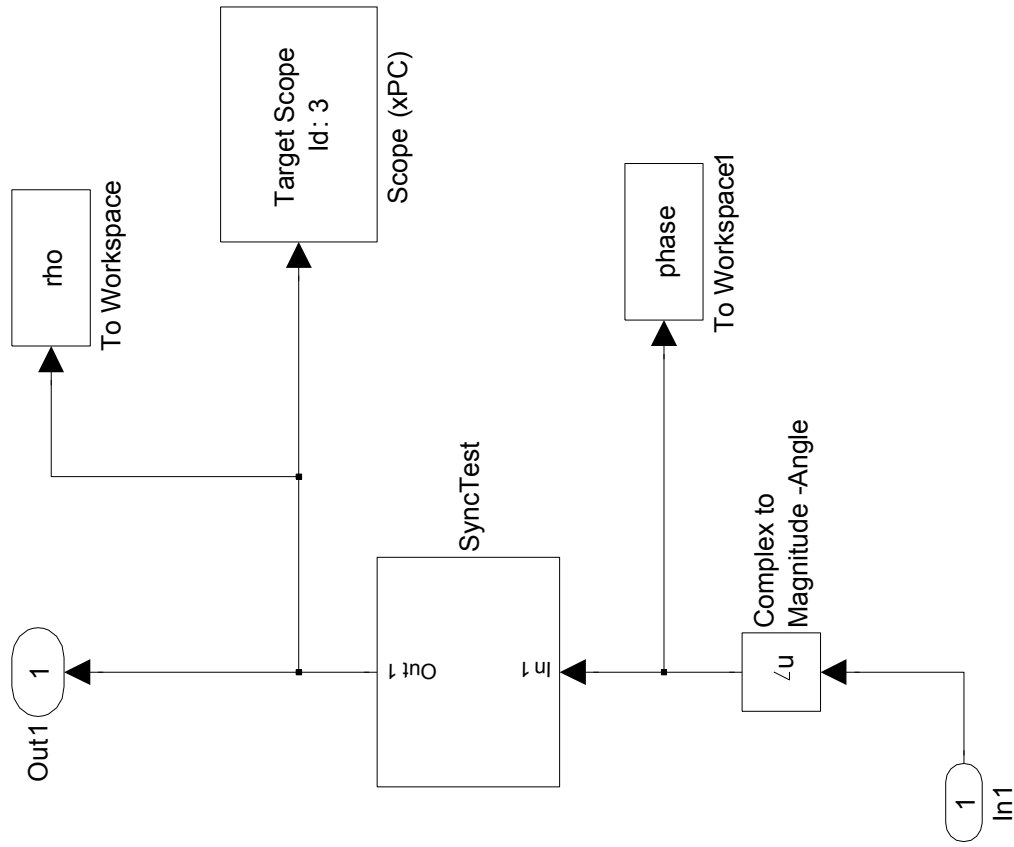




PSE /MASE /Band Splitter

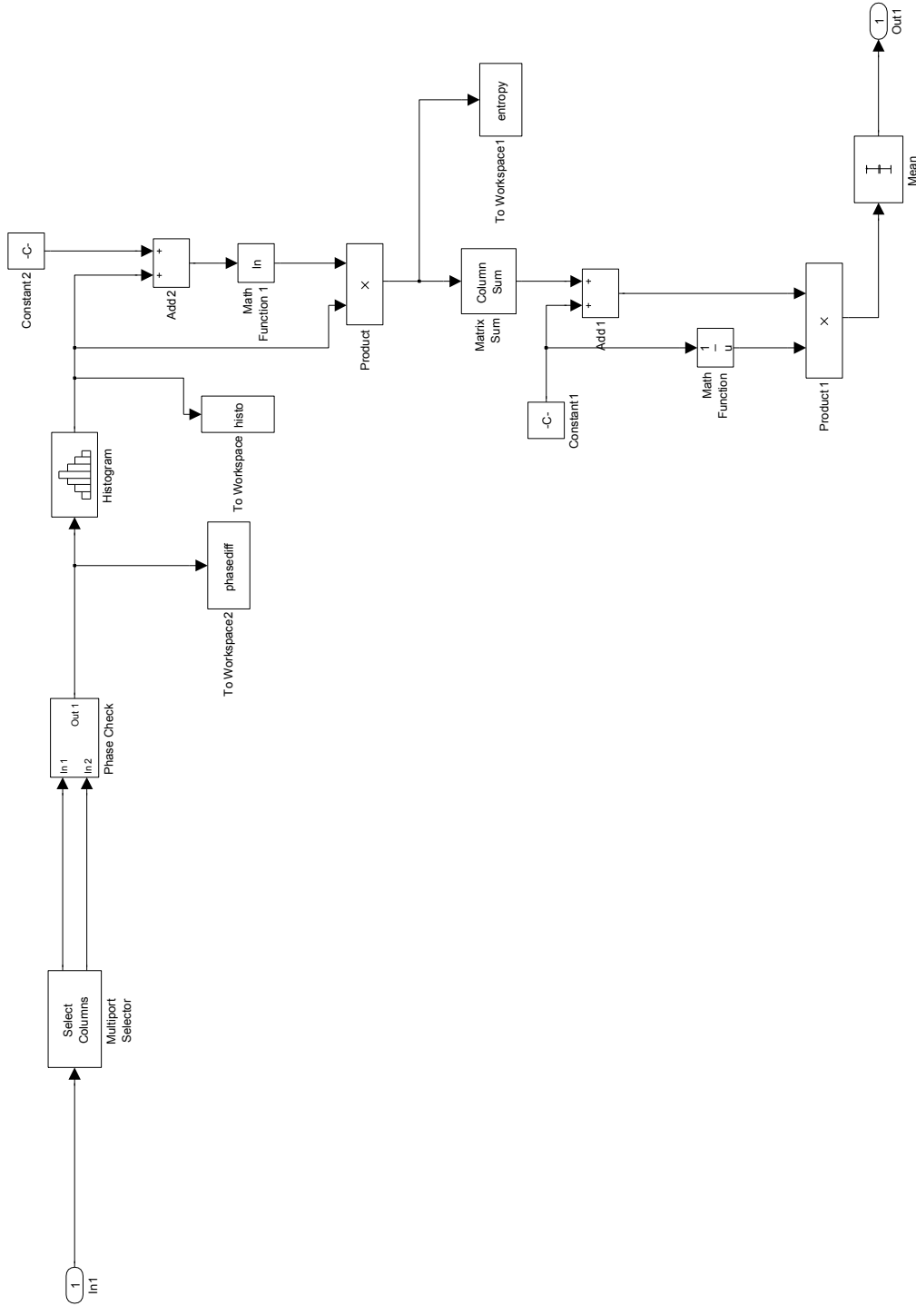


PSE/Synchronisation Testing



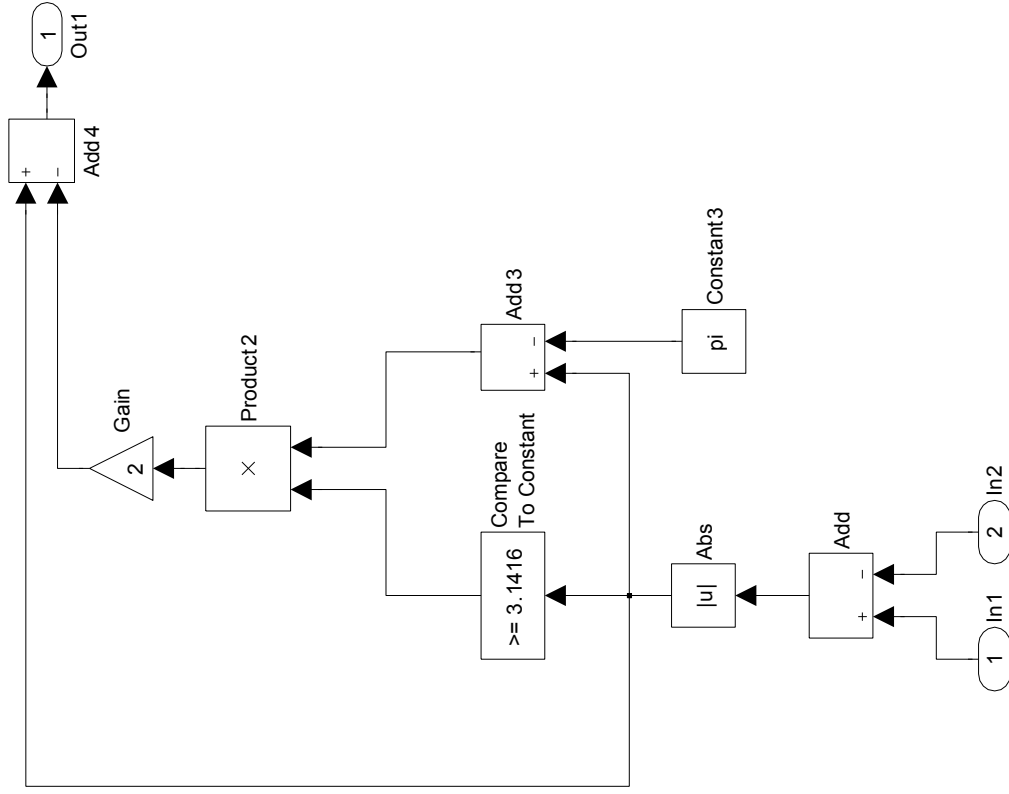
C:\Program Files \MATLAB \R2007 a\work\pcmaise \PSE .mdl

# PSE/Synchronisation Testing/SyncTest

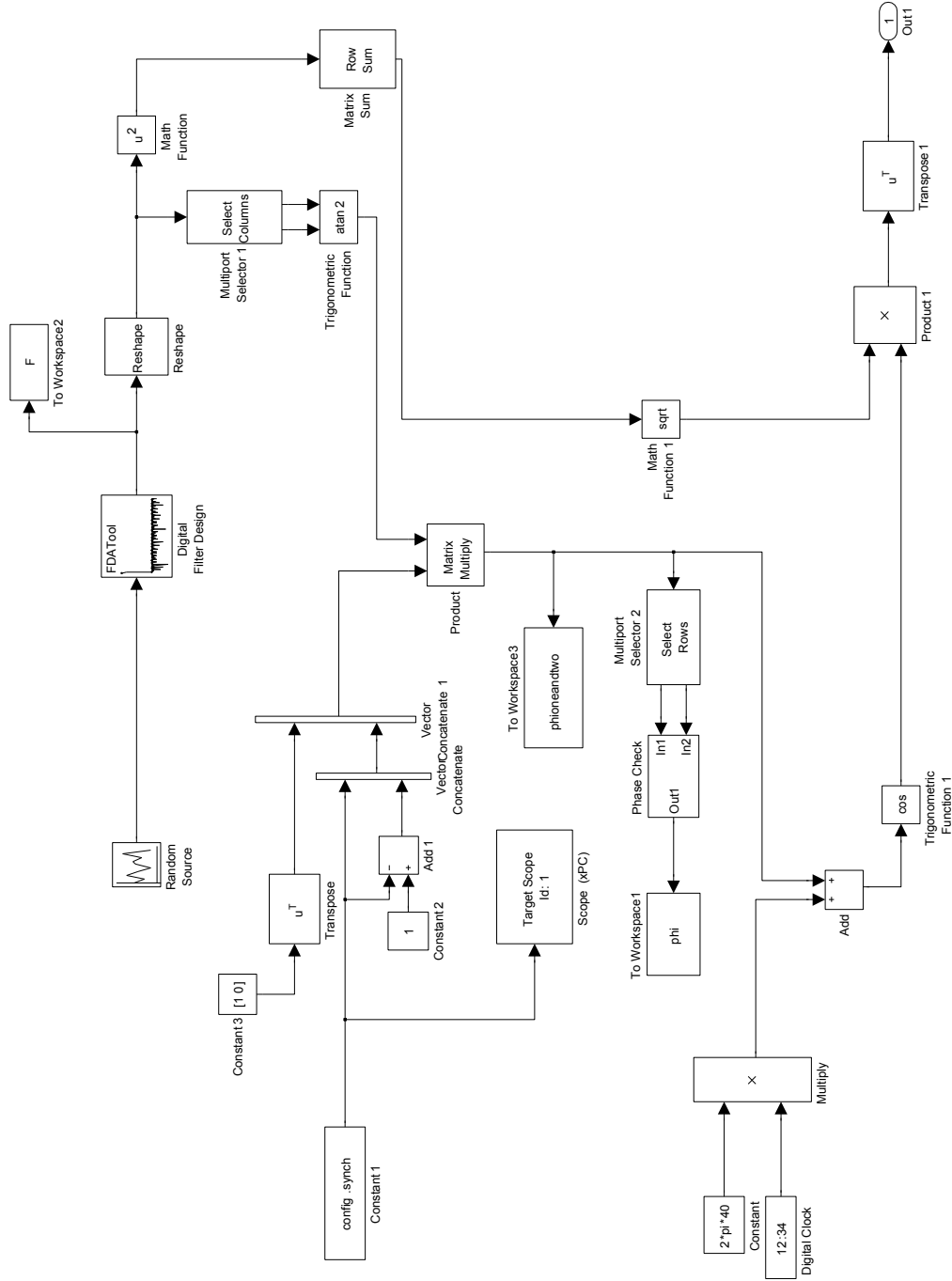


C:\Program Files \MATLAB \R2007 a\work\pcaise \PSE .mdl

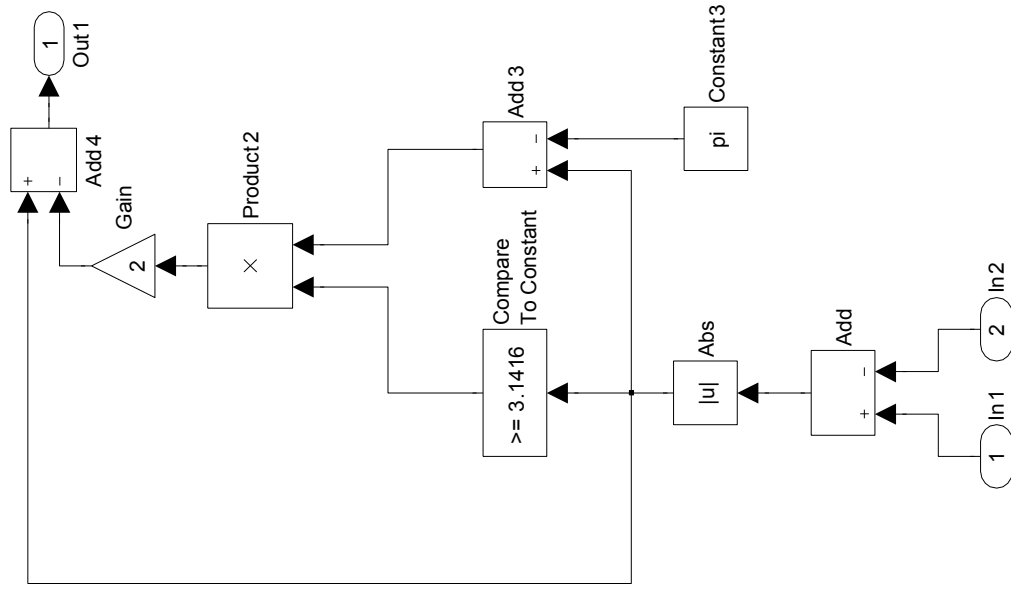
PSE/Synchronisation Testing/SyncTest/Phase Check



PSE/Test Data



C:\Program Files \MATLAB \R2007 a\work\pcaise \PSE .mdl



---

# Previous MAI Projects by the Author

---

Prior to this honours year of study, the author completed two smaller development projects for the Mind-Attention Interface. These projects were each of one semester's duration, and assessed as a 6 unit Advanced Studies course (ASC) as part of the Bachelor of Philosophy degree. While the work done in these projects is not used in the MMMI, it is important to describe clearly the work which was completed prior to the author's Honours year to avoid confusion.

## B.1 The MASE

A great deal of information can be extracted from the temporal profile of neural activity in a person's brain [72]. In particular, researchers have divided the neural signal frequency spectrum up to 50Hz into five frequency bands:

$\delta$  - band (0.5 - 2 Hz)

$\theta$  - band (3 - 7 Hz)

$\alpha$  - band (8 - 12 Hz)

$\beta$  - band (13 - 29 Hz)

$\gamma$  - band (30 - 100 Hz)

The amount of power in each of these frequency bands is known to correlate with various types of mental processing. In fact, the presence of a dominant alpha-wave in the resting brain was first discovered in 1924 by Hans Berger [73]. These frequency bands are relatively well understood, and have been the subject of a large amount of study. It is therefore desirable to measure the spectral power of a given EEG in each of these frequency bands as an initial function of the signal processing layer of the MMMI. The goal of the project was to build a signal processing module, the Mind Attention Spectral Engine (MASE), to perform this task in real-time.

The author designed and built the MASE using the MATLAB & Simulink package by the Mathworks [47]. Simulink is a graphical, block based environment with domain-specific tools available for building signal processing and control systems. The graphical nature of Simulink means it is intuitive to learn, and it contains many high-level signal processing routines built-in.

The spectral decomposition performed in the MASE was done using short-time Fourier transforms (STFTs). The STFT procedure used in the MASE involves taking short (1 second) sections of the signal, called epochs, and decomposing each epoch individually using a FFT. The energy in each frequency band is the sum of the appropriate terms in the magnitude FFT.

Using this procedure, the power in any given frequency band is determined for the duration of the signal epoch. By repeatedly evaluating this procedure for sequential signal epochs, a time-varying measure of the spectral power of the EEG signal can be obtained.

The MASE was designed and constructed in Simulink by the author in the Semester 2 2006 ASC. As a final part of this project, the MASE was also tested using artificial data with known spectral content. The MASE was shown to give accurate results and also to be relatively impervious to noise. At the completion of this ASC, the MASE was a standalone entity in Simulink, communication with the rest of the MAI had not yet been achieved.

The MASE is not used in the MAMI. However, the Simulink model is still maintained, and it's functionality is still available to other MAI projects. Given the importance of spectral analysis in many BCI design paradigms, the MASE remains a useful part of the MAI signal processing layer.

## **B.2 MASE-MAI Integration**

The second ASC undertaken by the author took place as part of a Summer Scholarship offered by the DCS in the summer of 2006/2007. The aim of this ASC was to continue the development of the MASE, and in particular to integrate the MASE into the MAI system. This task primarily involved the construction of a interface allowing Simulink (which the MASE is written in) to communicate with the rest of the MAI system using the MAI communication protocol.

Data transmission in the MAI uses the UDP protocol, so a suite of custom Simulink blocks for sending and receiving UDP messages was used to connect the MASE to the rest of the system. These blocks were developed by Giampiero Campa, and are freely available from the MATLAB Central File Exchange [74]. The blocks await MAI data packets on the designated UDP port, and pass this information through to the MASE for processing.

By the end of the ASC, communication between the MASE and the rest of the MAI system had been demonstrated. However, testing of the system revealed significant packet loss in this communication, and the MASE output was therefore not an accurate measure of the spectral content of the EEG signal. Further details of the problems encountered in this project are given in Chp. 5. Although many solutions to this

problem were attempted, the MASE was not able to be successfully integrated in the course of the ASC.

The problem of reliable, real-time communication between the signal processing layer tools (implemented in Simulink) and the rest of the MAI remained outstanding at the end of the author's ASC projects. A *successful* connection between Simulink and the MAI was not achieved until the 2007 Honours course.



---

# The Mind Attention Interface Project

---

There are many aspects of the MAI project which are not directly relevant to the MAMI. This section provides background information for the interested reader.

## C.1 MAI System Architecture

A great deal of the design work for the MAI software system has been done by Zhen Yang, another member of the MAI research team, for his Master's Thesis *Infrastructure Development for a Mind Attention Interface*. The MAI software system is composed of four layers (Fig. C.1).

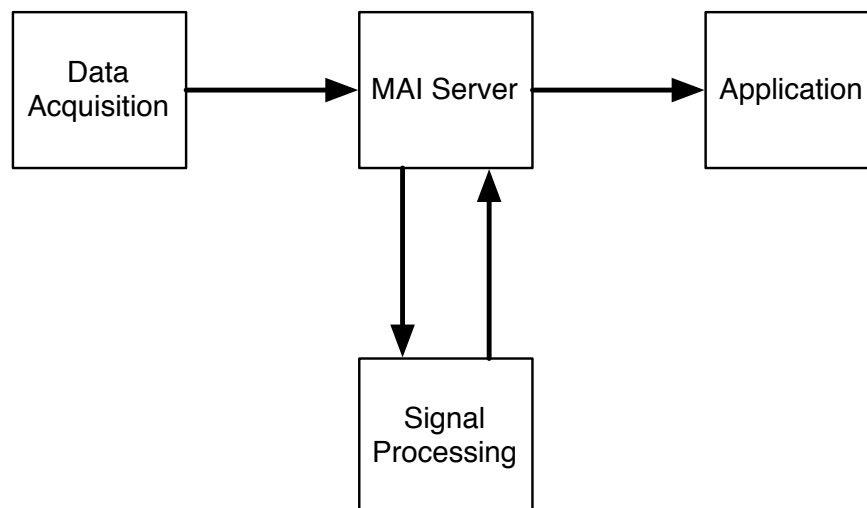


Figure C.1: MAI architecture

- The **MAI Server** is the heart of the MAI system. The MAI server uses a custom communication protocol based on an open source protocol called Neuroserver [49]. The Neuroserver protocol is part of the OpenEEG project, an attempt to build

an EEG system from scratch using low-cost electronic hardware. The MAI server co-ordinates data flow between the different components of the system by keeping a list of all currently connected components. Any component wishing to receive data from another component can select from this list which device it wishes to listen to. This ‘handshaking’ is performed over a TCP/IP connection for reliability. The data is then transferred directly between the respective components. The data transfer uses a connectionless multicast UDP datagram stream due to the lower network overhead of UDP and the large bandwidth required. The job of the MAI server is to manage these connections so that the different components of the MAI system, distributed over a network, can operate cohesively in real-time.

- The **Data Acquisition Layer** contains the devices for monitoring the physiological signals of the MAI participant. The current hardware configuration of the MAI contains a Biosemi *ActiveTwo* [51] digital EEG (for neural signal capture) and Seeing Machines *Facelab* [75] system (for gaze tracking). The data acquisition layer includes custom drivers written for each of these devices to make their measurement data available to the rest of the system. These drivers allow the hardware communicate with the MAI server and the other system components using the MAI server protocol. The system is designed to be extensible in the future to include other devices such as an electrocardiogram (for heart monitoring) and measurements of galvanic skin response (for measuring arousal).
- The **Signal Processing Layer** is responsible for high-level analysis and feature extraction of the raw data from the data acquisition layer. This transformed data is then available to other clients for visualisation and sonification. The signal processing layer can be viewed as an ‘intermediate’ layer between the DAQ and the application layer. Currently, all the signal processing algorithms in the MAI are written in MATLAB & Simulink [47]. This environment specialises in the efficient implementation of signal processing and other numerically intensive algorithms coupled with rapid development and prototyping. This signal processing layer has been primarily the responsibility of the present author throughout the development of the MAI and it’s chronology is described in Sec. B.
- The **Application Layer** is, potentially, the most interesting in the MAI. The applications in this layer govern the operation of the MAI as an interface. They must determine how the measurements of the users attention are utilised, and how the visual and auditory feedback is to be presented to the user in the wedge. These applications can use either the raw data from the data acquisition layer, or higher level features from the signal processing layer, or some combination of both. An application can be implemented using any programming language as long as it adheres to the MAI server protocol. One convenient software option is a package called BrainBay [76], a rapid-development environment which contains many standard tools for BCI development. BrainBay is also a part of the OpenEEG project, and as such is fully compatible with the (Neuroserver-based) MAI

---

communication protocol. BrainBay is currently being used for MAI application development by James Sheridan. Currently, there are not many completed modules in the application layer, although the MMMI can be considered a special case of the MAI.

The layered design of the MAI allows for a great deal of design flexibility. New devices, signal processing algorithms and applications can easily connect to the MAI server and are ready to use in the system. The MAI architecture can therefore be used to support other interface projects such as the MMMI. Utilising the data acquisition layer EEG functionality already in place, development of the MMMI has been concentrated on the key parts of the project, namely measuring the musical processing in the brain (its signal processing layer) and generating musical output (its application layer). As the portfolio of MAI projects grows and diversifies, the number of tools at the disposal of an MAI designer will also increase. The MAI therefore has the potential to become a feature rich development suite for the next generation of *natural* BCIs.

## C.2 Other MAI Projects

Several projects which use the MAI are already in development. Aside from the MMMI, PhD student Jaeyong Kim and final-year Software Engineering student Torben Schou are currently developing projects using the MAI infrastructure.

### C.2.1 Measuring Immersion in a Virtual Environment

Jaeyong Kim is currently investigating the amount of *presence* experienced by participants in virtual environments. Presence is defined as the psychological sense of *being there* in an immersive environment [77]. Presence is typically measured subjectively, such as having the participant move a slider while using the interface [78] or fill out a questionnaire afterwards [79]. Recent studies have also suggested objective measures of presence calculated from direct physiological measurements such as EEG [80].

The goal of Jaeyong's project is to directly compare subjective and objective measurements of presence in a virtual environment. Using the Wedge to display the immersive environments, participants are asked to record their sense of presence in real-time with a slider, while simultaneously having measures of their presence directly monitored via EEG. This study shall have important consequences for the comparison of other presence studies in the literature. Jaeyong's project uses many of the modules from the MAI, particularly for EEG acquisition and signal processing. It also involves the creation of new data acquisition layer objects, in particular the slider for recording user-reported presence. The MAI server allows these components to be added with relative ease.

### C.2.2 Immersive Computer Games in a Virtual Environment

Torben Schou has developed an immersive virtual world in the wedge. In particular, Torben has incorporated the Nintendo Wiimote [81], a motion-sensitive video game

controller and pointing device made by Nintendo, into the wedge environment. The Wiimote has been used to control a participant's movement around a virtual map of the ANU. In the past, interactive virtual worlds have been created in wedge-like setups. Torben's aim is to design a control interface for such a virtual world which maximises the sense of involvement and immersion experienced by the user. In the tasks of looking and moving around in the virtual world, the Wiimote allows for a more natural mapping than a conventional mouse and keyboard setup. This minimises the mental overhead on the user, and increases the subjective feeling of immersion. Initial user tests have yielded promising results regarding the intuitive nature of this control interface.

Torben's project does not use any direct physiological measurements in its control interface. This sets it apart somewhat from the other projects in the MAI portfolio. It is, however, concerned with the creation of a more natural interface for its chosen task, and in doing so freeing the user to concentrate on exploring a virtual world rather than struggling with basic control issues. This work may provide significant results for further virtual reality interface design in other MAI projects, and the Wiimote is also an interesting possibility as another interface option to the MAI.

---

# Bibliography

---

- [1] N.L. Wallin, B. Merker, and S. Brown. *The Origins of Music*. Mit Pr, 2000.
- [2] B. Miles. *Paul McCartney: Many Years from Now*. Owl Books, 1998.
- [3] I. Deliège and G.A. Wiggins. *Musical creativity: multidisciplinary research in theory and practice*. Psychology Press, 2006.
- [4] E.R. Kandel, J.H. Schwartz, and T.M. Jessell. *Principles of Neural Science*. Appleton & Lange, 2000.
- [5] E.R. Kandel. In search of memory: the emergency of a new science of mind. *J Clin Invest*, 116(5):1131, 2006.
- [6] J. Hawkins and S. Blakeslee. *On Intelligence*. Owl Books, 2005.
- [7] Yang Zhen. Infrastructure Development for a Mind Attention Interface. Master's thesis, Australian National University, September 2007.
- [8] J.J. Vidal. Toward Direct Brain-Computer Communication. *Annual Review of Biophysics and Bioengineering*, 2(1):157–180, 1973.
- [9] J.R. Wolpaw, N. Birbaumer, D.J. McFarland, G. Pfurtscheller, and T.M. Vaughan. Brain-computer interfaces for communication and control. *Clinical Neurophysiology*, 113(6):767–791, 2002.
- [10] T.N. Lal, T. Hinterberger, G. Widman, M. Schroder, NJ Hill, W. Rosenstiel, C.E. Elger, B. Scholkopf, and N. Birbaumer. Methods towards invasive human brain computer interfaces. *Advances in Neural Information Processing Systems*, 17:737–744, 2005.
- [11] S. Lin et al. Self-Organization of Firing Activities in Monkey's Motor Cortex: Trajectory Computation from Spike Signals. *Neural Computation*, 9(3):607–621, 1997.
- [12] L.R. Hochberg, M.D. Serruya, G.M. Friehs, J.A. Mukand, M. Saleh, A.H. Caplan, A. Branner, D. Chen, R.D. Penn, and J.P. Donoghue. Neuronal ensemble control of prosthetic devices by a human with tetraplegia. *Nature*, 442:164–171, 2006.
- [13] N.K. Logothetis, J. Pauls, M. Augath, T. Trinath, and A. Oeltermann. Neurophysiological investigation of the basis of the fMRI signal. *Nature*, 412:150–157, 2001.

- 
- [14] I.J. Rampil et al. A primer for EEG signal processing in anesthesia. *Anesthesiology*, 89(4):980–1002, 1998.
- [15] R.I. Goldman, J.M. Stern, J. Engel, and M.S. Cohen. Acquiring simultaneous EEG and functional MRI. *Clinical Neurophysiology*, 111(11):1974–1980, 2000.
- [16] M. Rappaport. P300 response under active and passive attentional states and uni-and bimodality stimulus presentation conditions. *Journal of Neuropsychiatry and Clinical Neurosciences*, 2(4):399–407, 1990.
- [17] SJ Segalowitz and KL Barnes. The reliability of ERP components in the auditory oddball paradigm. *Psychophysiology*, 30(5):451–9, 1993.
- [18] W.D. Penny, S.J. Roberts, and M.J. Stokes. Imagined hand movements identified from the EEG mu-rhythm. *Journal of Neuroscience Methods*, 1998.
- [19] X. Gao, D. Xu, M. Cheng, and S. Gao. A BCI-based environmental controller for the motion-disabled. *Neural Systems and Rehabilitation Engineering, IEEE Transactions on [see also IEEE Trans. on Rehabilitation Engineering]*, 11(2):137–140, 2003.
- [20] JR Wolpaw, DJ McFarland, TM Vaughan, and G. Schalk. The Wadsworth Center brain-computer interface (BCI) research and development program. *Neural Systems and Rehabilitation Engineering, IEEE Transactions on [see also IEEE Trans. on Rehabilitation Engineering]*, 11(2):1–4, 2003.
- [21] G. Pfurtscheller, C. Neuper, C. Guger, W. Harkam, H. Ramoser, A. Schlogl, B. Obermaier, and M. Pregenzer. Current trends in Graz brain-computer interface (BCI) research. *Rehabilitation Engineering, IEEE Transactions on [see also IEEE Trans. on Neural Systems and Rehabilitation]*, 8(2):216–219, 2000.
- [22] M. Kaper, P. Meinicke, U. Grossekhoefer, T. Lingner, and H. Ritter. BCI competition 2003-data set IIB: support vector machines for the P300 speller paradigm. *Biomedical Engineering, IEEE Transactions on*, 51(6):1073–1076, 2004.
- [23] LA Farwell and E. Donchin. Talking off the top of your head: toward a mental prosthesis utilizing event-related brain potentials. *Electroencephalogr Clin Neurophysiol*, 70(6):510–23, 1988.
- [24] C.S. Green and D. Bavelier. Action video game modifies visual selective attention. *Nature*, 423(6939):534–537, 2003.
- [25] E. Miranda and A. Brouse. Toward direct brain-computer musical interfaces. *Proceedings of the 2005 conference on New interfaces for musical expression*, pages 216–219, 2005.
- [26] B. Arslan, A. Brouse, C. Simon, R. Lehembre, J. Castet, J.J. Filatriau, and Q. Noirhomme. A real time music synthesis environment driven with biological signals. *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, 2, 2006.

- 
- [27] B. Arslan, A. Brouse, J. Castet, J.J. Filatriau, R. Lehembre, Q. Noirhomme, and C. Simon. Biologically-driven Musical Instrument. *Proceedings of eINTERFACE05 Summer workshop on multimodal interfaces*.
- [28] C. Gaser and G. Schlaug. Brain Structures Differ between Musicians and Non-Musicians. *Journal of Neuroscience*, 23(27):9240–9245, 2003.
- [29] TF Munte, E. Altenmuller, and L. Jancke. The musician’s brain as a model of neuroplasticity. *Nat Rev Neurosci*, 3(6):473–8, 2002.
- [30] G. Schlaug, LHL Lee, V. Thangaraj, RR Edelman, and S. Warach. Macrostructural adaptation of the cerebellum in musicians. *Soc. Neurosci*, 24(842.7), 1998.
- [31] J. Bhattacharya and H. Petsche. Phase synchrony analysis of EEG during music perception reveals changes in functional connectivity due to musical expertise. *Signal Processing*, 85(11):2161–2177, 2005.
- [32] M. Schulte, A. Knief, A. Seither-Preisler, and C. Pantev. Gestalt recognition in a virtual melody experiment. *Biomag2000, Proceedings of the 12th International Conference on Biomagnetism, Helsinki University of Technology, Espoo, Finland*, pages 107–110, 2001.
- [33] C.M. Krause, B. Porn, A.H. Lang, and M. Laine. Relative alpha desynchronization and synchronization during perception of music. *Scandinavian Journal of Psychology*, 40(3):209–215, 1999.
- [34] N. Jausovec, K. Jausovec, and I. Gerlic. The influence of Mozart’s music on brain activity in the process of learning. *Clin Neurophysiol*, 2006.
- [35] V.B. Mountcastle. *Perceptual Neuroscience: The Cerebral Cortex*. Harvard University Press, 1998.
- [36] N. Ramnani, T.E.J. Behrens, W. Penny, and P.M. Matthews. New approaches for exploring anatomical and functional connectivity in the human brain. *Biological Psychiatry*, 56(9):613–619, 2004.
- [37] D.T. Yew and D.T.W. Yew. *Basic Neuroanatomy*. World Scientific, 1996.
- [38] F. Varela, J.P. Lachaux, E. Rodriguez, and J. Martinerie. The brainweb: phase synchronization and large-scale integration. *Nature Reviews Neuroscience*, 2(4):229–239, 2001.
- [39] M.M. Mesulam. Large-scale neurocognitive networks and distributed processing for attention, memory, and language. *Annals of Neurology*, 28:597–613, 1990.
- [40] J.P. Lachaux, E. Rodriguez, J. Martinerie, C. Adam, D. Hasboun, and F.J. Varela. A quantitative study of gamma-band activity in human intracranial recordings triggered by visual stimuli. *European Journal of Neuroscience*, 12(7):2608–2622, 2000.

- [41] K. Ansari-Asl, L. Senhadji, J.J. Bellanger, and F. Wendling. Quantitative evaluation of linear and nonlinear methods characterizing interdependencies between brain signals. *Physical Review E*, 74(3):31916, 2006.
- [42] W. James. *The Principles of Psychology*. H. Holt, 1890.
- [43] J.K. Hietanen. Does your gaze direction and head orientation shift my visual attention. *Neuroreport*, 10(16):3443–3447, 1999.
- [44] D. Chen and R. Vertegaal. Using mental load for managing interruptions in physiologically attentive user interfaces. *Conference on Human Factors in Computing Systems*, pages 1513–1516, 2004.
- [45] A. Kapoor, RW Picard, and Y. Ivanov. Probabilistic combination of multiple modalities to detect interest. *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, 3, 2004.
- [46] A.M. Schuller and B. Rossion. Spatial attention triggered by eye gaze enhances and speeds up visual processing in upper and lower visual fields beyond early striate visual processing. *Clinical Neurophysiology*, 116(11):2565–2576, 2005.
- [47] Matlab Simulink. [www.mathworks.com](http://www.mathworks.com).
- [48] H. Gardner and R. Boswell. The wedge virtual reality theatre. *Proceedings of the Apple University Consortium Conference, Townsville, Qld, Australia, Sept*, pages 23–26, 2001.
- [49] Rudi Cilibrasi. OpenEEG. <http://openeeg.sourceforge.net/doc/sw/NeuroServer/>.
- [50] W.R. Stevens, B. Fenner, and A.M. Rudoff. *Unix Network Programming*. 2004.
- [51] Biosemi ActiveTwo. [www.biosemi.com](http://www.biosemi.com).
- [52] J. Hwang, B. Milne, N. Shirazi, and J. Stroomer. System Level Tools for DSP in FPGAs. *FPL 2001*, pages 534–543.
- [53] Simulink xPC Target Toolkit. [www.mathworks.com/products/xpctarget/](http://www.mathworks.com/products/xpctarget/).
- [54] A.V. Oppenheim, R.W. Schaffer, and J.R. Buck. *Discrete-time signal processing. Prentice-Hall Signal Processing Series*, page 870, 1999.
- [55] R.C. Gunning and H. Rossi. *Analytic functions of several complex variables*. Prentice-Hall Englewood Cliffs, NJ, 1965.
- [56] S.L. Hahn. *Hilbert transforms in signal processing*. Artech House Boston, 1996.
- [57] M.G. Rosenblum, A.S. Pikovsky, and J. Kurths. Phase Synchronization of Chaotic Oscillators. *Physical Review Letters*, 76(11):1804–1807, 1996.
- [58] R.M. Gray. *Entropy and information theory*. Springer-Verlag New York, Inc. New York, NY, USA, 1990.

- 
- [59] T.H. Corman, C.E. Leiserson, R.L. Rivest, et al. Introduction to Algorithms. *MIT Press, Cambridge, MA*, 2001.
- [60] S.G. Krantz. *Techniques of Problem Solving*. American Mathematical Society, 1997.
- [61] L. Song, E. Gysels, and E. Gordon. Phase synchrony rate for the recognition of motor imagery in BCI. *Advances in Neural Info. Proc. Sys*, 18, 2006.
- [62] K. Ansari-Asl, J.J. Bellanger, F. Bartolomei, F. Wendling, and L. Senhadji. Time-frequency characterization of interdependencies in nonstationary signals: application to epileptic EEG. *Biomedical Engineering, IEEE Transactions on*, 52(7):1218–1226, 2005.
- [63] C.L. Krumhansl. A perceptual analysis of Mozarts Piano Sonata K. 282: Segmentation, tension, and musical ideas. *Music Perception*, 13(3):401–432, 1996.
- [64] A. Sorensen. Impromptu: An interactive programming environment for composition and performance. *Proceedings of the Australasian Computer Music Conference 2005*, pages 149–153.
- [65] C. Roads. The Computer Music Tutorial. 1996.
- [66] Logic Pro. [www.apple.com/logicpro/](http://www.apple.com/logicpro/).
- [67] L.B. Meyer. *Style and Music: Theory, History, and Ideology*. 1996.
- [68] D. Tymoczko. *The Geometry of Musical Chords*, 2006.
- [69] CL Krumhansl and RN Shepard. Quantification of the hierarchy of tonal functions within a diatonic context. *J Exp Psychol Hum Percept Perform*, 5(4):579–94, 1979.
- [70] B Swift, J Sheridan, Zhen Yang, and Henry J. Gardner. Mind-Modulated Music in the Mind Attention Interface. November 2007. Australasian Conference on Computer-Human Interaction.
- [71] J Sheridan, Zhen Yang, B Swift, Henry J. Gardner, A Riddell, and A James. Construction of a Mind Attention Interface. 2008. submitted to IEEE Virtual Reality Conference.
- [72] E. Basar, C. Basar-Eroglu, S. Karakas, and M. Schurmann. Are cognitive processes manifested in event-related gamma, alpha, theta and delta oscillations in the EEG. *Neuroscience Letters*, 259:165–168, 1999.
- [73] H. Berger. Über das Elektrenkephalogramm des Menschen. *European Archives of Psychiatry and Clinical Neuroscience*, 87(1):527–570, 1929.
- [74] Giampiero Campa. UDP/IP Blockset. [www.mathworks.com/matlabcentral/fileexchange/loadFile](http://www.mathworks.com/matlabcentral/fileexchange/loadFile) 2005.

- [75] faceLAB 4. [www.seeingmachines.com](http://www.seeingmachines.com).
- [76] BrainBay. [www.shifz.org/brainbay/index.htm](http://www.shifz.org/brainbay/index.htm).
- [77] M. Slater, M. Usoh, and A. Steed. Depth of presence in virtual environments. *Presence: Teleoperators and Virtual Environments*, 3(2):130–144, 1994.
- [78] J. Freeman, SE Avons, D.E. Pearson, and W.A. IJsselsteijn. Effects of Sensory Information and Prior Experience on Direct Subjective Ratings of Presence. *Presence: Teleoperators and Virtual Environments*, 8(1):1–13, 1999.
- [79] M.J. Schuemie, P. van der Straaten, M. Krijn, and C.A.P.G. van der Mast. Research on Presence in Virtual Reality: A Survey. *CyberPsychology & Behavior*, 4(2):183–201, 2001.
- [80] A. Schlögl, M. Slater, and G. Pfurtscheller. Presence Research and EEG. *Proceedings of the 5th International Workshop on Presence*, 2002.
- [81] Nintendo Wii. [wii.nintendo.com](http://wii.nintendo.com).