2023

**Extended, exactly solvable chaotic oscillator**

Micah Tseng

Follow this and additional works at: https://louis.uah.edu/uah-theses

**Recommended Citation**

This Thesis is brought to you for free and open access by the UAH Electronic Theses and Dissertations at LOUIS. It has been accepted for inclusion in Theses by an authorized administrator of LOUIS.
EXTENDED, EXACTLY SOLVABLE
CHAOTIC OSCILLATOR

Micah Tseng

A THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Engineering
in
Electrical and Computer Engineering
to
The Graduate School
of
The University of Alabama in Huntsville

December 2023

Approved by:
Dr. Aubrey Beal, Research Advisor/Committee Chair
Dr. Laurie Joiner, Committee Member
Dr. Ned Corron, Committee Member
Dr. Aleksandar Milenkovic, Department Chair
Dr. Shankar Mahalingam, College Dean
Dr. Jon Hakkila, Graduate Dean
Abstract

EXTENDED, EXACTLY SOLVABLE CHAOTIC OSCILLATOR

Micah Tseng

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Engineering

Electrical and Computer Engineering
The University of Alabama in Huntsville
December 2023

Although the study of chaotic systems is theoretically mature for abounding examples, few are readily applicable to engineering problems. One notable barrier is the lack of analytic solution to guide or validate an engineered intention. In this thesis, a small set of chaotic systems known for their analytic solutions is expanded. Specifically, a known solvable second order solvable chaotic oscillator with a simple matched filter is extended such that the data rate is decoupled from the natural oscillation frequency of the oscillator. This development allows for high frequency applications without the need for high frequency switching. An extended, exact analytic basis function solution and return map are presented. The oscillator is validated via electronic hardware at audio frequencies where these experimental results closely match theoretical expectations.
Acknowledgements

I am deeply indebted to many people for not only this work, but for their encouragement to explore things I find exciting. I am privileged to work and play with some of the best people I’ve ever met. I would like to thank my committee for their tireless teaching, discussions, and help. They all have my utmost respect not only for their skill, but for the kindness, dignity and integrity that pervades their lives.

I would like to thank my advisor, Dr. Aubrey Beal, for his enduring patience, guidance and friendship. Without his presence, I never would have started a graduate education, and I certainly never would have finished.

There are also countless people who have been load barring walls in my life, people who have been with me through it all. I would not be where I am today, nor would this work exist without them. In particular, I would like to thank Nathan and Joel who saw me and believed in me a over a decade ago when I deeply needed it.

Finally, Mattan, Larry and Sarah: you all know your places in my life.

Thank you.
Table of Contents

Abstract ................................................................. ii

Acknowledgements ..................................................... iv

Table of Contents ...................................................... viii

List of Figures ............................................................ ix

Chapter 1. Introduction .................................................. 1

Chapter 2. Background ................................................... 4

2.1 Chaos ................................................................. 4

2.1.1 Maps ............................................................. 5

2.1.2 Orbits ............................................................ 6

2.1.3 Lyapunov Exponent ............................................. 7

2.1.4 Boundedness .................................................... 8

2.1.5 Definition of Chaos ............................................. 8

2.2 Symbolic Dynamics .................................................. 9

2.2.1 Full Shift ....................................................... 10

2.2.2 Subshift ........................................................ 11
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.2.3</td>
<td>Shift Map and Shift Space</td>
<td>13</td>
</tr>
<tr>
<td>2.2.4</td>
<td>Partitions</td>
<td>14</td>
</tr>
<tr>
<td>2.3</td>
<td>Entropy</td>
<td>16</td>
</tr>
<tr>
<td>2.3.1</td>
<td>Metric Entropy of Stochastic Systems</td>
<td>17</td>
</tr>
<tr>
<td>2.3.2</td>
<td>Metric Entropy of Deterministic Systems</td>
<td>19</td>
</tr>
<tr>
<td>2.3.3</td>
<td>Topological Entropy</td>
<td>22</td>
</tr>
<tr>
<td>2.4</td>
<td>Solvable Chaos</td>
<td>25</td>
</tr>
<tr>
<td>2.4.1</td>
<td>Basis Function Solution</td>
<td>25</td>
</tr>
<tr>
<td>2.4.2</td>
<td>Return Map</td>
<td>29</td>
</tr>
<tr>
<td>2.4.3</td>
<td>Half Period Basis Function</td>
<td>30</td>
</tr>
<tr>
<td>2.5</td>
<td>Extended Oscillator</td>
<td>32</td>
</tr>
</tbody>
</table>

**Chapter 3. Extended Second Order System**  

<table>
<thead>
<tr>
<th>Table</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Introduction</td>
<td>34</td>
</tr>
<tr>
<td>3.2</td>
<td>Oscillator</td>
<td>34</td>
</tr>
<tr>
<td>3.3</td>
<td>Return Map</td>
<td>39</td>
</tr>
<tr>
<td>3.4</td>
<td>Hardware Implementation</td>
<td>42</td>
</tr>
<tr>
<td>3.4.1</td>
<td>Analog</td>
<td>43</td>
</tr>
<tr>
<td>3.4.2</td>
<td>Digital</td>
<td>50</td>
</tr>
<tr>
<td>3.4.3</td>
<td>Implementation Summary</td>
<td>52</td>
</tr>
</tbody>
</table>
Appendix F. Reverse Impedance of Floating GIC
### List of Figures

2.1 Time series of tent map given in Eq. (2.1) for $B = 2$ (left), current vs. future iterate for $B = 2$ (center) and current vs. future iterate for $B = 1.5$ (right). ................................................. 6

2.2 Example transition diagrams for a symbolic dynamical system with two symbols, $a$ and $b$. .......................................................... 11

2.3 Bernoulli shift map for $B = 2$ on the left and $B = (1 + \sqrt{5})/2 = \varphi$ on the right. ................................................................. 21

2.4 Time series (left) and basis pulse (right) for the original full period basis function at $\beta = \ln(2)$. ................................. 26

2.5 Return map, Eq. (2.28), of original full period oscillator. ......... 30

2.6 Time series (left) and basis pulse (right) for the half period basis function at $\beta = \ln(4)$. ................................................ 32

3.1 Time series of $u(t)$ and $s(t)$ for $k = 1, 2, 3, 4$ started with the same arbitrary initial condition. ........................................... 38

3.2 Basis functions, $P_k(t)$, for $k = 1, 2, 3, 4$. ............................ 39

3.3 Return map $u_n$ for odd $k$ on the left and even $k$ on the right when $\beta = \ln(4)$. ................................................................. 41

3.4 Oscillator Schematic. ....................................................... 43

3.5 Negative Resistor. ......................................................... 47

3.6 Generic General Impedance Converter (GIC). ....................... 48

3.7 Negative RLC filter with GIC. ........................................... 49

3.8 One Shot pulse generator with $M$ NOT gates to create a delay line where $M$ is an even integer. ........................................ 51

3.9 Debounce filter timing diagram. ........................................ 52

3.10 Detailed analog schematic of extended second order solvable chaotic oscillator. .................................................. 53
3.11 Time series of theoretical analytic solution ($u_{theo}$) and measured hardware oscillator ($u_{meas}$) for various values of $k$.  
3.12 Maps from hardware for various values of $k$.  

A.1 The linear superposition of two half period basis functions to give one full period basis function.  

D.1 Example topological (left) and Markov (right) partition regions ($I_n$ for $n = 1, 2, 3, 4$) over arbitrary map, $u_n$.  

E.1 Detailed debounce filter timing diagram.
Chapter 1. Introduction

In the large field of dynamics, chaos captures curiosity by formally encapsulating an oxymoron: bounded, exponential growth. If a chaotic system is instantiated twice with two different, but infinitesimally close initial conditions, the two instances of the system will exponentially separate from each other in a finite period of time yet remain bounded. It is asymptotically unstable, growing exponentially, but doesn’t ”explode” to infinity. The seemingly impossible contradiction of bounded divergence creates a second oxymoron: deterministic randomness. A chaotic system behaves randomly though it is entirely deterministic. There are no random variables, no coin flips, and no hidden physical realizations, yet these systems have positive entropy; they have information content like random signals. These two paradoxes of chaos have challenged us for over a hundred years: from Henri Poincaré in the 1890’s studying the $n$-body problem to Edward Lorenz examining meteorology in the 1960’s (1). Chaos arises naturally in our physical world, yet can also be used in engineering and in some fields chaos has an integral place in the theoretical framework of the field.

One difficulty to applying chaos to engineering problems is that chaos is usually nonlinear (2) and lacks an analytic solution. While it is possible to use a non-solvable chaotic system for engineering (see Scott Hayes’ work(3; 4; 5)), solv-
able chaos allows for engineering without approximation. In 2010, Corron et al. engineered an exactly solvable second order chaotic oscillator (6). The remarkable thing about the oscillator is that not only is it analytically solvable, but the solution is a convolution sum of a symbol sequence and a simple basis function. With a solution of this form, the oscillator has a fixed matched filter to detect the symbol sequence at the optimal signal to noise ratio (SNR) in the presence of additive white gaussian noise (AWGN). The matched filter is an infinite impulse response filter formed by an RLC circuit and an integrate-and-dump. A few years later in 2016, Corron et al. presented a first order oscillator that also has a stable infinite impulse response matched filter. The fact that chaos arose naturally in the optimal waveform for stable infinite impulse response filters prompted the conjecture by Corron and Blakely that the optimal communications waveform for any stable infinite impulse response filter is chaotic (7). In 2020, the conjecture was proven for a large class of filters. The matched waveform for stable infinite impulse response filters with an all pole transfer function and a finite number of poles (8) is chaotic. Filters included in this class are the ubiquitous Butterworth, Chebyshev (type I), and Bessel filters, giving chaos a place at near the heart of communications. The relation between chaos and optimal communications filters provides impetus for examining more exactly solvable chaotic oscillators.

In the following thesis, the second order solvable chaotic oscillator originally presented in 2010 by Corron et al. is extended by separating the data rate from the oscillation frequency. By decoupling the data rate from the natural oscillation frequency, the oscillator resembles a communications waveform with a
carrier frequency greater than the data rate; only here, the data is determined by the dynamics of the oscillator and not an external message. The extended oscillator admits a new analytic basis function and closed form solution. Since the oscillator admits a closed form expression for the basis function, it also has a fixed matched filter. Thus, the extended oscillator increases the small set of known chaotic systems with closed form analytic solutions.

This thesis is organized as follows: Chapter 2 provides an introduction to chaos and some of the tools used in the field of nonlinear dynamics and chaos with an emphasis on solvable chaos. Chapter 3 develops the extended oscillator mathematically and in hardware. First the analytic basis function solution is developed and then a return map for the system is derived, providing insight into the properties of the system. Then a discrete hardware implementation at audio frequencies is presented using an analog negative RLC filter and a digital FPGA. From the hardware results, a time series plot and return map for multiple sets of natural oscillation frequencies and data rates is presented. Finally, Chapter 4 concludes this work and points to avenues of future research.
Chapter 2. Background

2.1 Chaos

In this section, the basic mathematical tools to formally understand chaos will be introduced. Chaos captures two oxymorons: bounded exponential divergence and deterministic randomness. Implicit to both of these is the larger goal of dynamics: to understand the evolution of a function (or map) in time. For any dynamical map, an orbit defines the evolution of a point in time. As these points evolve in time, they may converge to a single value or they may start oscillating between a small set of points or they may show more interesting behavior. If two orbits of a map are started from two infinitesimally close initial conditions and they exponentially separate within a finite period of time, never converging while remaining bounded, they are called chaotic orbits as defined by Alligood, Sawyer and Yorke (ASY). The key is the exponential separation which is defined by the Lyapunov exponent. Next, the large world of symbolic dynamics opens the door to qualitative analysis. Rather than examining orbits defined in the real numbers, a conjugate symbolic dynamical system can be constructed through partitions to encapsulate the dynamics of the system through a discrete set of symbols. These symbols herald to the work of Claude Shannon (9) and indeed provide one link to entropy. Shannon introduced entropy as a measure of information, a mea-
sure of randomness for stochastic random variables. The Russian mathematician, Kolmogorov, and his student, Sinai extended Shannon’s entropy to deterministic systems which is now referred to as Kolmogorov-Sinai (KS) entropy. KS entropy provides the basis of the aforementioned deterministic randomness paradox of chaos: chaos has a positive KS entropy. Chaos, a deterministic system, has information. The KS entropy is related to the Lyapunov exponent through the Pesin entropy identity providing a link between the sensitive dependence upon initial conditions of chaos and its information content.

2.1.1 Maps

In nonlinear dynamics, maps are used to describe a system and how it evolves through time (continuous or discrete) from one point to the next. In the larger mathematical world, maps are a more general structure than functions. The relate one space to another by any relation or morphism. In this thesis, the maps of primary concern are one dimensional discrete maps, mapping an interval of real numbers onto itself; i.e., a map \( T : I \to I \) for some interval \( I = [a, b] \in \mathbb{R} \). For example, the one dimensional discrete tent map is defined by the recurrence relation

\[
T : u_{n+1} = \begin{cases} 
Bu_n, & u_n < 0.5 \\
B(1 - u_n), & u_n \geq 0.5 
\end{cases}
\]  

(2.1)

for \( 0 < B < 2 \) and \( 0 \leq u_0 \leq 1 \). Given some initial condition, \( u_0 \), the recurrence relation in Eq. (2.1) maps the interval \([0, 1]\) to itself: \( T : [0, 1] \to [0, 1] \). For \( B = 2 \), the map is surjective (onto). The evolution of the map through time is
dependent not only upon time, but also upon the initial condition, similar to how the solution to a differential equation is dependent upon an initial condition. The time series of the tent map given in Eq. (2.1) for $B = 2$ is plotted on the left in Figure 2.1. However to better illustrate the structure of discrete one dimensional maps, the future iterate is commonly plotted against the previous iterate as shown in the center and right plots of Figure 2.1 for $B = 2$ and $B = 1.5$ respectively. By plotting the future iterate against the current iterate, the evolution from one point to the next may be easily seen independent of time.

![Figure 2.1: Time series of tent map given in Eq. (2.1) for $B = 2$ (left), current vs. future iterate for $B = 2$ (center) and current vs. future iterate for $B = 1.5$ (right).](image)

### 2.1.2 Orbits

An orbit is the set of all points a map visits given some initial condition. Mathematically, given a map $f$ and initial condition, $x_0$, an orbit is the set of points $\{x_0, f(x_0), f(f(x_0)), \ldots\}$; i.e., a orbit is the entire trajectory of a map started at some initial condition. There are a few different types of orbits that receive special distinction: fixed points and periodic orbits. A fixed
point is an orbit that is constant; for a given initial condition $x$, the orbit is defined by $x = f(x)$. For example, the tent map given in Eq. (2.1) has fixed points at $u_n = 0$ and $u_n = B/(B + 1)$. A periodic orbit with period $n$ is an orbit that for $n = 2, 3, 4, \ldots$, $x_n = x$; after $n$ evolutions the sequence repeats. For example, the tent map in Eq. (2.1) has a second order periodic orbit of $\{B/(1 + B^2), B^2/(1 + B^2), B/(1 + B^2), \ldots\}$ and a third order periodic orbit of $\{B/(1 + B^3), B^2/(1 + B^3), B^3/(1 + B^3), B/(1 + B^3), \ldots\}$. Chaotic orbits are not periodic (nor asymptotically periodic), but non-periodicity is not the only requirement for chaos; chaotic orbits must also show a sensitive dependence upon initial conditions as defined by the Lyapunov exponent.

### 2.1.3 Lyapunov Exponent

The Lyapunov exponent is the exponential rate at which orbits with infinitesimally close initial conditions separate from each other. The Lyapunov exponent of the orbit with initial position $x_0$ on map $T : \mathbb{R} \to \mathbb{R}$ is defined as

\[
\Lambda_{x_0} = \lim_{n \to \infty} \frac{1}{n} \left( \ln |T'(x_0)| + \ln |T'(x_1)| + \ldots \ln |T'(x_n)| \right). \tag{2.2}
\]

For example, consider the tent map defined in Eq. (2.1) for $B = 2$. The Lyapunov exponent may be calculated as

\[
\Lambda_{u_n} = \lim_{n \to \infty} \frac{1}{n} \sum_{i=1}^{n} \ln |u'_i| = \ln(2). \tag{2.3}
\]
for all orbits except those that contain the point $1/2$ since the derivative at $1/2$
does not exist. One important thing to note is that within the definition of the
Lyapunov exponent is an implicit dependence upon scale and time; it averages
the growth of the map as time evolves. For discrete maps, Lyapunov exponent is
given per iteration; this becomes important when considering systems in hardware
where time is measured in seconds. With exponential divergence defined by the
Lyapunov exponent, now the boundedness of chaos may be examined.

2.1.4 Boundedness

One of the key aspects of chaos is that though chaotic orbits exponentially
separate from each other, they are still bounded. Mathematically, a real, bi-
infinite sequence, $\left( s_n \right)_{n \in \mathbb{Z}}$, is bounded above by $M \in \mathbb{R}$ if $s_n \leq M \ \forall \ n \in \mathbb{Z}$. Similarly, a real, bi-infinite sequence, $\left( s_n \right)_{n \in \mathbb{Z}}$, is bounded below by $m \in \mathbb{R}$ if $s_n \geq m \ \forall \ n \in \mathbb{Z}$. For example, the tent map given in Eq. (2.1) is bounded
on interval $[0, 1]$ for $0 < B \leq 2$.

2.1.5 Definition of Chaos

There are two common definitions of chaos: the Alligood, Sauer and York
(ASY) (10) and Devaney (12) definition. They are equivalent to each other (13),
but differ in one distinct point: the ASY definition defines chaos for a single
orbit, whereas the Devaney definition defines chaos for an entire system (or set of
orbits). For the purposes required here, the simpler ASY definition is sufficient.
The (ASY) definition for a chaotic orbit is as follows (10; 13). Let $T$ be a map of the real line, $T : \mathbb{R} \rightarrow \mathbb{R}$. Let a bounded orbit from initial position $x_0$ be defined as $\text{orbit}(x_0) = \{x_0, T(x_0), T(T(x_0)), \ldots\}$. The bounded orbit is chaotic if,

- $\text{orbit}(x_0)$ is not asymptotically periodic ($\text{orbit}(x_0)$ does not become periodic after some finite period of time)

- the Lyapunov exponent of $T$ operating from initial position $x_0$ is greater than zero

In other words, the orbit must not converge to a periodic orbit, row diverge exponentially, and still be bounded. The tent map in Eq. (2.1) for initial conditions of irrational numbers fulfill the conditions of ASY chaos. Now with chaos formally defined, a brief departure from chaos will be made to introduce symbolic dynamics.

### 2.2 Symbolic Dynamics

Symbolic dynamics is a powerful tool providing a glue between continuous and discrete time dynamical systems and information theory, but beyond use as an intermediary tool, symbolic dynamics is a rich mathematical field in and of itself (14; 15). Continuous and discrete time dynamics studies how maps (usually of real numbers) evolve through time. Symbolic dynamics studies how symbols in an alphabet evolve through time. Rather than study orbits of real numbers, symbolic dynamics studies sequences of symbols. As a full mathematical field, symbolic dynamics has its own terminology, definitions, axioms and operators,
but only the most brief introduction to symbolic dynamics will be given here; for a quick introduction of symbolic dynamics see (14) and for a complete treatment see (15).

2.2.1 Full Shift

The core of symbolic dynamics is the full shift; it defines every possible sequence of symbols (letters) in an alphabet. Let \( \mathcal{A} \) denote a symbol set or alphabet which will be assumed to be finite here. A full shift, \( \mathcal{A}^\mathbb{Z} \), over alphabet \( \mathcal{A} \) (or the full \( \mathcal{A} \)-shift) is defined as

\[
\mathcal{A}^\mathbb{Z} = \{ x = (x_i)_{i \in \mathbb{Z}} : x_i \in \mathcal{A} \ \forall \ i \in \mathbb{Z} \},
\]

(2.4)

where \((x_i)_{i \in \mathbb{Z}}\) denotes a bi-infinite sequence. A full shift over the alphabet \( \mathcal{A} \) is the collection of all bi-infinite sequences of symbols of \( \mathcal{A} \). For example, if \( \mathcal{A} = \{a, b\} \), then a portion of one sequence in the full shift may be

\[...ababbaabbabbaa...
\]

(2.5)

It is important to note that symbols have no value other than labeling a letter in an alphabet; symbols could be letters (\( \{a, b, c,\ldots\} \)) or numbers (\( \{-1, 1\} \)) or words in the English language (\( \{\text{raccoon, dog, cat}\} \)). Sets of symbols in a sequence of symbols are grouped together into blocks/words similar to how letters in the English alphabet are grouped together to form words. For example, \( aaaaa \) or \( abbb \) are two different four symbol blocks of the alphabet \( \mathcal{A} \). To help understand how
the system evolves, it is often useful to visualize the possible transitions between symbols (or blocks) in a transition diagram as shown in Figure 2.2 for the alphabet $\mathcal{A} = \{a, b\}$.

![Transition Diagrams](image)

**Figure 2.2:** Example transition diagrams for a symbolic dynamical system with two symbols, $a$ and $b$.

In the symbolic system represented by the left transition diagram in Figure 2.2, any transition is possible from any symbol to any symbol; this transition diagram represents a full shift. In the symbolic system represented by the right transition diagram in Figure 2.2, any transition is possible except from the symbol $a$ to itself. That means the block $aa$ will never occur in the sequence of symbols outputted by the system; since all sequences with the block $aa$ cannot occur and a full shift is the set of all possible sequences using a given alphabet, this system is no longer a full shift and instead is called a subshift.

### 2.2.2 Subshift

A subshift, $X^\mathcal{F}$, is a subset of a full shift $\mathcal{A}^\mathbb{Z}$ composed of all sequences except those sequences containing a block in the forbidden set, $\mathcal{F} \in \mathcal{A}^\mathbb{Z}$. There is no restriction on $\mathcal{F}$; it may be finite or infinite or may contain only the empty set, $\emptyset$. In the case $\mathcal{F} = \emptyset$, the subshift equals the full shift: $X^\emptyset = \mathcal{A}^\mathbb{Z}$. If $\mathcal{F}$ is finite, the subshift is called a subshift of finite type (SFT). For example, if $\mathcal{F} = \{aa\}$ for
\( \mathcal{A} = \{a, b\} \) no sequence containing the word \( aa \) is allowed as shown in the right transition diagram in Figure 2.2. A selection of a valid sequence in \( X^F \) may be

\[
...babbabbb...
\]  

(2.6)

The forbidden words are commonly called a grammar restriction upon the full shift \( \mathcal{A}^\mathbb{Z} \). Another way to visualize transitions of a symbolic dynamical system is to use a transition matrix. The transition matrix, \( \mathbf{A} \), is an \( n \times n \) matrix with entry \( a_{ij} \) at the \( i^{th} \) row and \( j^{th} \) column. If entry \( a_{ij} \) is a one, it denotes a possible transition from the \( i^{th} \) symbol to the \( j^{th} \) symbol. If entry \( a_{ij} \) is a zero, it denotes no transition is possible from the \( i^{th} \) symbol to the \( j^{th} \) symbol. Some literature refers to a transition matrix as a topological Markov chain. For example, the subshift \( X^F \) of \( \mathcal{A} = \{a, b\} \) with \( F = \{aa\} \) has a transition matrix,

\[
\mathbf{A} = \begin{bmatrix}
0 & 1 \\
1 & 1
\end{bmatrix}.
\]  

(2.7)

The first row denotes that when the current symbol is \( a \), the next symbol must be \( b \). The bottom row denotes that when the current symbol is \( b \), the next symbol could either be \( a \) or another \( b \). On the surface the transition matrix looks quite similar to a stochastic matrix, but there is one distinct difference: a stochastic matrix says what transitions are probable, a transition matrix says if a transition is possible. Transition matrices have no dependence upon probability.
Up until now, the focus has been on the symbols in symbolic dynamics, but there is also an operator.

### 2.2.3 Shift Map and Shift Space

The main operator in symbolic dynamical systems is the shift map. The shift map, $\sigma$, maps $\mathcal{A}^\mathbb{Z}$ onto itself by mapping the point $x_i$ to the next (left) iterate $x_{i+1}$. That is, $\sigma : \mathcal{A}^\mathbb{Z} \to \mathcal{A}^\mathbb{Z}$ and $\sigma(x_i) = x_{i+1}$. Combining the shift map operator with a subshift gives a shift space. A shift space is the pair $(X, \sigma_X)$, where $X$ is a subshift and $\sigma_X$ is the shift operator restricted only to the subset $X$.\textsuperscript{1} The shift operator is so remarkably simple, it can be confusing; it literally shifts every letter in the sequence one position to the left. For example, consider the selection of a sequence in the full shift,

\[ \ldots ababba \bar{a}bbab \ldots \quad (2.8) \]

where the bar over $a$ denotes the current symbol in time. Applying the shift operator, $\sigma$, gives the sequence,

\[ \ldots ababba \bar{a}bbab \ldots \quad (2.9) \]

Every symbol has been shifted to the left. The simplicity of the shift operator makes it appealing in comparison to the complexity of some continuous or discrete

\textsuperscript{1}Here the language can get rather confusing. In some literature a subshift is called a shift space or vice versa. The issues can be further complicated when introducing the discrete dynamical system also called a shift map (like a Bernoulli shift map) that has a conjugate symbolic dynamical system with a shift map, $\sigma$. Unfortunately only context can provide distinction.
dynamical systems. Is there a way to utilize the simplicity of the shift operator when studying continuous and discrete dynamical systems? The answer, is yes. Continuous or discrete dynamical systems can be mathematically conjugate to symbolic dynamical systems. Thus, rather than examining orbits of real numbers with perhaps intricate nonlinearities, sequences of a finite set of symbols and a simple shift operator can be studied without loss of generality. The entities that relates symbolic dynamical systems and continuous or discrete dynamical systems are partitions and conjugacy.

2.2.4 Partitions

Partitions are the glue that connect continuous or discrete dynamical systems to symbolic dynamical systems. Consider the tent map given in Eq. (2.1) for $B = 2$ and shown in the center of Figure 2.1. Let $\mathcal{P}$ be a partition at $u_n = 1/2$ such that two partition regions $I_0$ and $I_1$ are defined as $I_0 = [0, 1/2)$ and $I_1 = [1/2, 1]$. The union of $I_0$ and $I_1$ covers the bounds of the tent map so every point of the tent map falls in an partition region: $I_0 \cup I_1 = [1,0]$. When the tent map, $T$, at time $n$, enters the region $I_0$ let the symbolic dynamical sequence, $(x_n)$, take the symbol 0 and when the tent map, $T$, at time $n$, enters the region $I_1$, let the symbolic dynamical sequence, $(x_n)$, take the symbol 1: i.e., if $u_n \in I_0 \rightarrow x_n = 0$ and if $u_n \in I_1 \rightarrow x_n = 1$. For example, given the arbitrary initial condition $u_0 = \sqrt{2} - 1$, the first four iterates of the tent map correlate to the following
iterates of a symbolic sequence, \((x_n)\),

\[
\begin{align*}
u_0 &\approx 0.4142 & T \to u_1 &\approx 0.8283 & T \to u_2 &\approx 0.3434 & T \to u_3 &\approx 0.6867 & T \to \ldots \\
\downarrow \mathcal{P} & & \downarrow \mathcal{P} & & \downarrow \mathcal{P} & & \downarrow \mathcal{P} & & \downarrow \mathcal{P} \\
x_0 = 0 & \to x_1 = 1 & \to x_2 = 0 & \to x_3 = 1 & \to \ldots
\end{align*}
\]

As \(T\) iterates from \(u_n\) to \(u_{n+1}\), the shift operator, \(\sigma\), shifts the symbols mapped by the partition to the left in time. The partition at \(1/2\) of the tent map is called a generating partition.\(^2\) Every orbit in a map partitioned with a generating partition has a unique symbol sequence in the conjugate symbolic dynamical system. Since each orbit has a unique conjugate symbol sequence all of the dynamics of the map are captured in the symbol sequence and the symbol sequences may be studied instead of the orbits without loss of generality. For a simple example, consider again the tent map for \(B = 2\) started with the initial condition \(u_0 = \sqrt{2} - 1\) with the conjugate symbol sequence, \((x_n)\) created by the generating partition at \(u_n = 1/2\). Let \(v_n\) be a second orbit of the tent map for \(B = 2\) with initial condition \(v_0 = \sqrt{2} - 1 - 0.01\) and has the conjugate symbol sequence, \((y_n)\) created by the generating partition at \(v_n = 1/2\). Let \(w_n\) be a third orbit of the tent map for \(B = 2\) with initial condition \(w_0 = \sqrt{2} - 1 \times 1E - 16\) and has the conjugate symbol sequence, \((z_n)\) created by the generating partition at \(w_n = 1/2\). The tent map

\(^2\)There are a few different types of partitions with different properties. Every topological partition admits symbolic dynamics, but only generating partitions admit a unique symbol sequence for each orbit(13). Please see Appendix D for details.
has a Lyapunov exponent of ln(2) so close orbits will separate from each other at a rate of ln(2); the closer the initial conditions are to each other, the longer it will take for them to separate. The same behavior can be seen in the conjugate symbolic dynamical systems. The initial conditions \( u_0 \) and \( v_0 \) differ by 0.01. The first four shifts of their conjugate symbol sequences, \( (x_n) \) and \( (y_n) \) are the same, but the fifth shift differs: \( x_4 = 1 \) and \( y_4 = 0 \). Compare the orbits \( u_n \) and \( w_n \) with the initial conditions \( u_0 \) and \( w_0 \) which differ by \( 1 \times 10^{-16} \). The first fifty-two shifts of the symbol sequences, \( (x_n) \) and \( (w_n) \) are the same and the fifty-third shift differs: \( x_{52} = 0 \) and \( w_{52} = 1 \). In other words, the smaller the difference in the initial conditions to the discrete dynamical system, the tent map, the more shift operations are required by the symbolic dynamical system to see the difference. This example illustrates one way the dynamics of a discrete or continuous dynamical system may be studied by examining the conjugate symbolic dynamical system. Further, mathematical conjugacy guarantees certain entities are invariant between both systems; one important invariant is topological entropy\(^{14} \).

Since topological entropy is invariant, the topological entropy of the tent map and its conjugate symbolic dynamical system are the same. Now with the tool of symbolic dynamics, the second oxymoron of chaos can be discussed: deterministic randomness.

### 2.3 Entropy

Entropy is the core measure in the field of information theory. First introduced to the world in 1948 by Claude E. Shannon, entropy is the measure of
information in a given event or system (9). There are many different types of
entropy that are largely application specific, but they all share the same principles developed by Shannon. Shannon quantified the uncertainty of the events as
information: the more uncertain an outcome to an event, the more information
that outcome gives about the event. The measure of the amount of information
in an event is entropy, \( h \), and the units of entropy are \textit{bits or nats} (9).\footnote{Some literature prefers to use \textit{nats} (such as (16)) while others prefer \textit{bits} (for example (9)). The two units of entropy, \textit{bits} and \textit{nats} are related by \( x_{\text{bits}} = x_{\text{nats}} / \ln(2) \) and \( x_{\text{nats}} = x_{\text{bits}} / \log(e) \).} Information theory is a massive field and only the barest introduction to entropy will be provided here to aid in understanding chaos. Three forms of entropy will be discussed: first, metric entropy for stochastic random variables given by Shannon will be presented as an introduction to entropy. Then Kolmogorov-Sinai (KS) entropy for deterministic systems will be introduced. Finally, a completely different type of entropy, topological entropy will be introduced.

\subsection*{2.3.1 Metric Entropy of Stochastic Systems}

Entropy is the measure of information from a given event. If the event is
highly predictable (low uncertainty) then little information is gained from it. For
element, consider the sequence of independent outcomes from an arbitrary event:

\begin{equation}
\text{abababababababababababababab...} \tag{2.10}
\end{equation}

In the above sequence, the letter \( a \) is always followed by the letter \( b \) which is
followed by the letter \( a \); there is a large amount of redundancy, low uncertainty
and thus low entropy. Alternatively, consider a different sequence of independent outcomes from an arbitrary event:

\[
abbbbaabaabbaaabbaaababababb\ldots 
\]  

(2.11)

There is no apparent pattern to the sequence and there is no way to predict the next symbol; there is little redundancy, high uncertainty, and thus high entropy. This example illustrates the inverse relationship between redundancy and information. More information means less redundancy. More redundancy means less information. For random variables, this idea can be formulated easily as given by Shannon (9). For a discrete random variable, \( X \), with probability mass function \( p_X(x) \), the Shannon entropy \( h(X) \) of \( X \) is given by

\[
h(X) = - \sum_x p_X(x) \log_2(p_X(x)).
\]  

(2.12)

To better understand entropy, consider two extreme cases. Consider the random variable \( X \) with two outcomes: \( a \) and \( b \). If the probability for \( a \) is 1 and the probability for \( b \) is 0, then the entropy of \( X \) is zero because every draw of \( X \) is the same and there is no uncertainty. Alternatively, consider when the probability for \( a \) is 0.5 and the probability for \( b \) is 0.5; the entropy of \( X \) is 1 bit or approximately 0.693 nats, the maximum entropy possible for a two symbol system. Since the probability of \( a \) and \( b \) are equal, there is no redundancy in the system and each draw on \( X \) has maximum uncertainty. Since entropy depends upon uncertainty, it
is easy to see how random, stochastic, systems are entropic, but it is also possible for deterministic systems to have entropy.

### 2.3.2 Metric Entropy of Deterministic Systems

Some events, such as coin flips or horse races, have so many variables and dimensions there is a large amount of uncertainty to their outcome and thus they have positive entropy. Such events are usually quantified through probability theory using random variables and Shannon’s entropy can be used to measure the information in their outcomes. Though it may be intuitive that stochastic systems have positive entropy, deterministic systems without any random variable may also have positive entropy. Even though a deterministic system is governed by a known rule, it may have positive entropy because there is uncertainty in the observer’s knowledge of the initial condition to the system (17). To illustrate this idea, consider the Bernoulli shift map shown in Figure 2.3 for $B = 2$ on the left and $B = \varphi$ (the Golden Ratio) on the right and given by

$$u_{n+1} = \text{mod}_1(Bu_n),$$

(2.13)

where $0 < B \leq 2$ and mod$_1$ is the modulus base 1 function. For $B = 2$, Consider the generating partition at $1/2$ and let the symbol for the left partition region be $L$ and the symbol for the right partition region be $R$. Let the initial condition $u_0$ be an irrational number on $[0, 1]$, which is a realistic assumption due to the density of irrational numbers. An infinite number of digits are required to exactly express irrational numbers; it would take an infinite amount of memory to exactly
store an irrational number. However, the tent map can describe (in the absence of noise) an irrational number perfectly. Since $u_0$ is contained in the interval $[0, 1]$, it must map to either the code $L$ or $R$; to continue the example, suppose $u_0 \rightarrow L$. This means $u_0$ must be some irrational number on the interval $[0, 0.5]$. Then consider the next iterate $u_1$; the shift map takes the interval $u_0$ is on, $[0, 0.5]$ and stretches it to fill the interval $[0, 1]$. Thus if $u_1$ less than $1/2$ that means $u_0$ must have been on the interval $[0, 0.25]$ and if $u_1$ is greater than $1/2$, $u_0$ must have been on the interval $[0.25, 0.5]$. Again, to continue the example, suppose $u_1 \in [0.5, 1]$ which means $u_0 \in [0.25, 0.5]$. The next iterate, $u_2$, stretches the interval of $u_1$, $[0.5, 1]$, to the interval $[1, 2]$ and then the mod function folds $[1, 2]$ back to $[0, 1]$; that means, the interval of $u_0$, $[0.25, 0.5]$ has been stretched and folded twice now. If $u_2 \in [0, 0.5]$, that implies $u_1 \in [0.5, 0.75]$ which implies $u_0 \in [0.25, 0.375]$ or if $u_2 \in [0.5, 1]$, that implies $u_1 \in [0.75, 1]$ which implies $u_0 \in [0.375, 0.5]$. Expressed concisely, the stretching and folding up to $u_3$ is given by

\[
\begin{align*}
  u_0 & \rightarrow L \rightarrow u_0 \in [0, 0.5] \\
  u_1 & \rightarrow R \rightarrow u_0 \in [0.25, 0.5] \\
  u_2 & \rightarrow R \rightarrow u_0 \in [0.375, 0.5] \\
  u_3 & \rightarrow L \rightarrow u_0 \in [0.375, 0.4375] \\
  \ldots
\end{align*}
\]

Notice how with each iterate, the interval from which the initial condition, $u_0$ may have come from is cut in half; i.e., one bit of information is revealed about the
Bernoulli shift map for $B = 2$ on the left and $B = (1 + \sqrt{5})/2 = \varphi$ on the right.

initial condition. With each iteration, the shift map acts like a magnifying glass, stretching the interval from which the initial conditions originates to a scale the observer can distinguish one bit of information. It is this stretching and folding mechanism that gives chaos entropy.

Formally, the metric entropy of a deterministic system is called Kolmogorov-Sinai (KS) entropy, $h_{KS}$, and was created by Kolmogorov and Sinai in 1959 (18; 19; 20). There are various methods for calculating $h_{KS}$ depending on the application, but perhaps the most beautiful illustration of the relation between chaos and entropy is through Pesin’s equality (21; 13). For diffeomorphism\(^4\), $T$, with an absolutely continuous invariant Lebesgue measure, and Lyapunov exponents $\Lambda_i$, the KS entropy in nats is defined as

$$h_{KS}(T) = \sum_{i: \Lambda_i > 0} \Lambda_i.$$  (2.14)

\(^4\)A diffeomorphism is a differentiable map with a differentiable inverse that maps two smooth manifolds one-to-one and onto.
A slightly looser relation is provided by Ruelle (22; 16). Given a differentiable map, $T$, with an ergodic invariant measure with compact support on a finite-dimension manifold\textsuperscript{5} the KS entropy in \textit{nats} is bounded by

$$h_{KS}(T) \leq \sum_{i; \Lambda_i > 0} \Lambda_i,$$  \hspace{1cm} (2.15)

where $\Lambda_i$ are the Lyapunov exponents.

Pesin and Ruelle’s relations cannot be applied to our simple example shift map or tent map due to the discontinuities for both Pesin and Ruelle’s relations and the lack of bijection for Pesin’s equality\textsuperscript{6}, but the intuition they give is valuable. Like in the example of the shift map given above, the Lyapunov exponent defines the stretching over the interval. The larger the Lyapunov exponent, the faster (over a given time interval) the map stretches and the more information is revealed. With this intuition in mind, another important type of entropy can be discussed.

### 2.3.3 Topological Entropy

Both Shannon entropy and KS entropy are different forms of metric entropy; they require a measure and the probability of events. Topological entropy is not a metric entropy. Introduced by Adler, Konheim and McAndrew in 1965,\textsuperscript{7}

\textsuperscript{5}The general idea in English is that the map has to be smooth enough to differentiate, the ruler to measure nearness doesn’t change as the map evolves, the map is bounded and the map has a finite number of dimensions.

\textsuperscript{6}It is possible to calculate the metric entropy of the shift map and tent map by constructing a PDF using the Frobenius-Perron operator and integrating over the density, however, that is beyond the scope here(13).
Topological entropy tells of what “can” happen where metric entropy tells of what will “probably” happen (23). Topological entropy does not depend on probabilities and provides an upper bound on metric entropy. One of the most useful properties of topological entropy is that it is an invariant of conjugacy (14); i.e., the topological entropy of a continuous or discrete dynamical system is the same as its conjugate symbolic dynamical system. There are a few different ways of calculating topological entropy, but a method developed by Robinson (24) provides a beautiful link to symbolic dynamics. Consider a symbolic dynamical system with subshift $X$ and transition matrix $A$. The topological entropy, $h_T(X)$ is the logarithm of the spectral radius of the transition matrix $A$. That is,

$$h_T(X) = \log_2(\rho(A)),$$

(2.16)

where $\rho(A)$ is the spectral radius of $A$.$^7$ For example, the shift map defined in Eq. (2.13) has a generating partition (which is also Markov; see Appendix D for definition) at $1/2$. Since each partition fully maps to the whole interval, the transition matrix for the conjugate symbolic dynamical system is given as

$$A = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}.$$  

(2.17)

---

$^7$The spectral radius, $\rho$, of a matrix, $A$, with $N \in \mathbb{N}_1$ eigenvalues, $\lambda_n$, is defined as, $\rho(A) = \max\{|\lambda_n| : 0 \leq n < N\}$.
The spectral radius of $A$ is $\rho(A) = 2$, so the topological entropy of the symbolic dynamical system is $h_T = \log_2(2) = 1$ bit and using the the fact that topological entropy is invariant under conjugacy, the topological entropy of the Bernoulli shift map for $B = 2$ is, $h_T = 1$ bit. That means the shift map is capable of producing one bit of information; this example has no grammar restrictions (any word is allowed) and the conjugate dynamical system is a full shift. If the growth rate, $B$, is reduced to $B = \varphi$ where $\varphi$ is the Golden Ratio, and a partition that is generating is placed at $(1 - \sqrt{5})/2$ with the left region coded as $L$ and the right region coded as $R$, the transition matrix for the conjugate symbolic dynamical system is given as

$$A = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}.$$

(2.18)

The subshift of finite type conjugate to the Bernoulli shift map with $B = \varphi$ may be written as, $X^F$ for $F = \{RR\}$. By reducing the growth rate (the stretching rate) of the shift map, a word in the conjugate symbolic dynamical system ($RR$) that is possible in a full shift has been made impossible. The loss of the word can be seen in the topological entropy: the spectral radius of $A$ is $\rho(A) \approx 1.618$ and the topological entropy is $h_T \approx 0.694$ bits. Reducing the growth rate $B$ and loosing the word $RR$ has reduced the topological entropy in the system by $\approx 0.306$ bits.
The past section has provided an extremely brief introduction to entropy with emphasis placed on the entropy of chaotic systems. Chaos, though deterministic, has positive entropy as shown by both KS entropy and topological entropy. The information comes from the resolution of uncertainty to the observer of the initial condition to the system. For more information about the entropic properties of chaos see (13; 15). Now with all the basics to nonlinear dynamics and chaos covered, solvable chaos can be examined.

2.4 Solvable Chaos

Chaos has generally been thought to be only nonlinear and without closed form solution(1). However, in *A Matched Filter for Chaos*(6), Corron *et al.* show a chaotic oscillator that admits an exact analytic solution. In this section the mathematics of the solvable chaotic oscillator are explored beginning with the basis function solution.

2.4.1 Basis Function Solution

The analytic solution of the second order solvable chaotic oscillator can be written as a discrete linear convolution between a basis function and a symbol sequence (6). Consider the hybrid differential equation with continuous state $u(t)$ and discrete state $s(t)$,

\[ \ddot{u} - 2\beta \dot{u} + (\omega^2 + \beta^2)(u(t) - s(t)) = 0. \]  

(2.19)
Figure 2.4: Time series (left) and basis pulse (right) for the original full period basis function at $\beta = \ln(2)$.

where $\omega = 2\pi k$, $k = 1$ and $\beta$ is fixed such that $0 < \beta \leq \ln(2)$. The transitions of $s(t)$ are defined by the guard condition,

$$\dot{u}(t) = 0 \rightarrow s(t) = \text{sgn}(u(t)), \quad (2.20)$$

where

$$\text{sgn}(x) = \begin{cases} 
-1, & x < 0 \\
+1, & x \geq 0
\end{cases} \quad (2.21)$$

The guard condition is triggered when the derivative, $\dot{u}(t)$, equals zero and when the guard condition is met, the forcing function, $s(t)$, is set to the sign of $u(t)$. The time series is plotted in Figure 2.4. The guard condition given in Eq. (2.20) is met at every half time period ($t = 0, 1/2, 1, 3/2...$). However, by induction the forcing function $s(t)$ can only change every whole time period ($t = 0, 1, 2,...$)(6). Thus, the forcing function, $s(t)$, is a binary symmetric rectangular
pulse series as shown in Figure 2.4 and may be written as

$$s(t) = \sum_{n=-\infty}^{\infty} s_n p_1(t - n),$$

(2.22)

where the rectangular pulse $p_\tau(t)$ is defined as

$$p_\tau(t) = \begin{cases} 
1, & 0 < t < \tau \\
0, & \text{else}
\end{cases}$$

(2.23)

for $\tau = 1$. The discrete state $s_n \in \{-1, 1\}$ for $n \leq t < n + 1$ is defined as

$$s_n = s(t)|_{t=n} = \text{sgn}(u(t))|_{t=n}$$

(2.24)

for $n \in \mathbb{Z}$. Similar to a communications system, the discrete state $s_n$ is the $n^{th}$ symbol with possible values $\{-1, +1\}$. As will be discussed in the next section, the discrete state $s_n$ maps directly to the symbols in a conjugate symbolic dynamical system. Assume the solution $u(t)$ to the differential equation (2.19) has the form

$$u(t) = \sum_{n=-\infty}^{\infty} s_n P_{\text{orig}}(t - n),$$

(2.25)

where $P_{\text{orig}}(t)$ is a continuous basis function. Solving the differential equation of the form Eq. (2.25) admits a static basis function plotted in Figure 2.4 and given
by,

\[ P_{\text{orig}}(t) = \begin{cases} 
(1 - e^{-\beta})e^{\beta t}[\cos(\omega t) - \frac{\beta}{\omega}\sin(\omega t)], & t \leq 0 \\
1 - e^{\beta(t-1)}[\cos(\omega t) - \frac{\beta}{\omega}\sin(\omega t)], & 0 < t \leq 1 \\
0, & t > 1 
\end{cases} \quad (2.26) \]

Thus, the hybrid differential equation admits a solution that is a discrete linear convolution of a basis function with a symbol sequence. The important thing about a solution of this form is that the associated waveform, \( u(t) \), has a fixed matched filter. A matched filter provides the optimal SNR given AWGN for detecting the symbols of a given system\((4)\). For a given waveform, the impulse response of the matched filter for the optimal peak pulse SNR in AWGN at sample time \( T \) is the time-reversed and amplitude scaled basis function of the waveform sampled shifted by time \( T \). The impulse response of the matched filter, \( h(t) \), for the solvable chaotic oscillator is given by \((6)\)

\[ h(t) = \alpha P_{\text{orig}}(T - t), \quad (2.27) \]

where \( \alpha \) is an amplitude scaling constant and \( T \) is the sample time. The impulse response (barring the necessary shift of one time unit to maintain causality) is a stable infinite impulse response filter. This gives credence to the conjecture by Corron and Blakely that the optimal communications waveform for any stable infinite impulse response filter is chaotic \((7)\).
2.4.2 Return Map

The symbols of the oscillator are determined by the initial conditions. This can be shown clearly by constructing a return map from sampling the system at every full period: \( u_n = u(n) \) for \( n \in \mathbb{Z} \). In *A Matched Filter for Chaos* (6), a recursive derivation is presented.\(^8\) The state \( u_n \in \mathbb{R} \) is iterated via the recurrence relation,

\[
    u_{n+1} = e^\beta u_n - (e^\beta - 1)s_n, 
\]

and the discrete state \( s_n \in \{\pm 1\} \) is by

\[
    s_{n+1} = \text{sgn}(u_{n+1}),
\]

where \( \text{sgn}(x) \) is given by Eq. (2.21). The return map given in Eq. (2.28) forms a shift map plotted in Figure 2.5. The shift map has a generating partition at \( u_n = 0 \) so the discrete values \( \{ -1, +1 \} \) may be coded as symbols in a conjugate symbolic dynamical system; i.e., every trajectory of the continuous time solution \( u(t) \) given an initial condition \( u(0) = u_0 \) maps uniquely to a sequence of symbols given by \( s_n \). In other words, given an initial condition, \( u_0 \), the set of \( s_n \in \{-1, 1\} \) is a set of discrete states that map one-to-one and onto a set of symbols in a conjugate symbolic dynamical system. The symbols in the conjugate symbolic dynamical system may be anything: \( \{A, B\} \), \( \{\text{bob},\text{joe}\} \), or they could be the convenient labels \( \{-1, +1\} \) that map directly to the domain of the discrete state \( s_n \). To aid

\(^8\)There are two methods for deriving the the return map: a recursive derivation as detailed in Appendix B or it can be derived from the basis function as shown in the Chapter 3.
readability, the mathematically rigorous distinction between the discrete state, $s_n$, and the conjugate symbolic dynamical system will be relaxed and $s_n$ will often be referred to as a “symbol” though mathematically it is actually a discrete state that maps to a symbol in a conjugate symbolic dynamical system. For $\beta = \ln(2)$, the conjugate symbolic dynamical system forms a full shift with topological entropy $h_T = 1\text{bit}$. That means, the return map and the continuous dynamical system also have a topological entropy of $1\text{bit}$, given a one period time scale.

![Figure 2.5](image)

**Figure 2.5:** Return map, Eq. (2.28), of original full period oscillator.

### 2.4.3 Half Period Basis Function

A few years after *A Matched Filter for Chaos* was published and in the context of folded band chaos, Corron *et al.* presented a half period basis function that is a more general basis function for the second order system given in Eqs. (2.19, 2.20) (25). Consider the differential equation given in Eq. (2.19) with the same guard condition as given in Eq. (2.20) and constant $\omega = 2\pi k$ for $k = 1$. However, now extend the range of $\beta$ to $0 < \beta \leq \ln(4)$. As before $\dot{u}(t) = 0$ when $t = 0, 1/2, 1, 3/2..., $ but now by extending $\beta$, the forcing function $s(t)$ now may
change every half time period (6; 25). The forcing function is now given by

\[ s(t) = \sum_{n=-\infty}^{\infty} s_n p_{1/2}(t - \frac{n}{2}), \quad (2.30) \]

where the symbols \( s_n \in \{ \pm 1 \} \) are defined as \( s_n = s(n/2) \) for \( n \in \mathbb{Z} \). Assume the solution \( u(t) \) to the differential equation (2.19) has the form

\[ u(t) = \sum_{n=-\infty}^{\infty} s_n P(t - \frac{n}{2}), \quad (2.31) \]

where \( P(t) \) is assumed to be continuous. Solving the differential equation for the basis function gives (25),

\[ P(t) = \begin{cases} 
(1 + e^{-\beta/2})e^{\beta t}[\cos(\omega t) - \frac{\beta}{\omega} \sin(\omega t)], & t \leq 0 \\
1 + e^{\beta(t-1/2)}[\cos(\omega t) - \frac{\beta}{\omega} \sin(\omega t)], & 0 < t \leq \frac{1}{2} \\
0, & t > \frac{1}{2} 
\end{cases} \quad (2.32) \]

The half period basis function doubles the symbol rate of the system. The half period basis function is more general than the full period because the original basis function in Eq. (2.26) can be composed of two half period basis pulses. For a detailed comparison of the two basis functions and the full vs. half period timing, please see Appendix A. No return map for the half period timing has been published as of this writing so one will be derived in Chapter 3. To avoid confusion the half period timing will be used by default for the rest of this thesis.
2.5 Extended Oscillator

In the second order solvable chaotic oscillator, the symbol rate is tied directly to the oscillation frequency through the guard condition; as the oscillation frequency is increased, the discrete symbol switching frequency must also increase. For high frequency applications, the switching rate may become a limiting factor. Previous research in high frequency solvable chaos has focused on the analog natural oscillation frequency with techniques such as using reverse-time chaos (26) or nested negative impedance converters (27) in hardware. Neither technique addresses discrete switching limitations.

In this thesis, the exactly solvable second order chaotic oscillator will be extended to decouple the symbol rate from the natural oscillation frequency. By decoupling the symbol rate from the natural oscillation frequency, a hardware-independent system is developed that can be applied to any application with specific limitations in discrete switching and analog oscillation frequency. In the second order solvable chaotic oscillator, the natural oscillation frequency $\omega$ was
set to $\omega = 2\pi k$ for $k = 1$. Here, $k$ will be considered for all natural numbers, $k = 1, 2, 3...$, while maintaining a constant symbol pulse width. Thus, the symbol rate will be maintained while the oscillation frequency can change by integer multiples. The relation between $k$ and the natural oscillation frequency has been observed previously in reverse time chaos (2), but has not been reported in forward time chaos. The next chapter will develop the extended oscillator.
Chapter 3. Extended Second Order System

3.1 Introduction

In the following Chapter, the extended second order oscillator for $k \in \mathbb{N}_1$ is developed in both mathematics and hardware. First, an analytic solution in the form of a convolution between a symbol sequence and basis function is derived. Then, a return map derived from the basis function solution. To maintain readability, the details of the derivations may be found in Appendices B and C. Next, the oscillator is implemented in hardware using a mixture of analog and digital components and hardware results are presented.

3.2 Oscillator

The goal of the extended oscillator is to separate the oscillation frequency from the data rate. In the original second order oscillator, the oscillation frequency of the sinusoids, $\omega$, was defined as $\omega = 2\pi k$ for $k = 1$. The data rate in the original oscillator is set by the guard condition; a new symbol is produced every time the derivative equals zero. To separate the oscillation frequency from the data rate, consider holding the symbol rate constant and setting the oscillation frequency by $\omega = 2\pi k$ for $k \in \mathbb{N}_1$. Consider the hybrid oscillator equation with continuous
state \( u(t) \in \mathbb{R} \) and a discrete state \( s(t) \in \{\pm 1\} \),

\[
\ddot{u} - 2\beta \dot{u} + (\omega^2 + \beta^2)(u - s) = 0 \\
\dot{u} = 0 \quad \rightarrow m = m + 1 \\
m = k \quad \rightarrow s = \text{sgn}(u) \& m = 0,
\]

where fixed constants \( \omega = 2\pi k, k \in \mathbb{N}_1, 0 < \beta \leq \ln(4) \), \( m \) is an integer counter and \( \text{sgn}(x) \) is defined by Eq. (2.21). There are two main parts to the oscillator: an unstable differential equation and a guard condition that forms a symmetric forcing function. The key to the oscillator is that as the oscillation frequency changes with \( k \), the guard condition also changes to maintain a constant symbol time period no matter the value of \( k \). The constant, \( \omega \), sets the natural oscillation frequency of the oscillator; as \( k \) is increased the oscillation frequency also increases. The guard condition occurs when \( m = k \). Every time the derivative crosses zero (\( \dot{u} = 0 \)), the counter \( m \) is increased until the guard condition triggers at \( m = k \), upon which the sign of \( u(t) \) is sampled for the forcing function and the counter is reset. For a detailed derivation, please see Appendix C as here only the main points are outlined.

First, observe two fixed points for \( u = 1 \) with \( s = 1 \) and \( u = -1 \) and \( s = -1 \). The fixed point does not depend on the guard condition or \( k \). Later the fixed points can be observed in the hardware results, but for now they will be ignored.
Since the second order oscillator in Eq. (3.1) shows a guard condition trigger at a constant 1/2 time interval, a discrete state \( u_n \) may be set such that \( u_n = u(n/2) \) for \( n \in \mathbb{Z} \) and the symbols at time \( t = n/2 \) may be defined as \( s_n = \text{sgn}(u_n) \). With the symbols defined at every 1/2 time interval, the forcing function may be written as

\[
s(t) = \sum_{n=-\infty}^{\infty} s_n p_{\frac{1}{2}}\left(t - \frac{n}{2}\right),
\]

(3.2)

where \( n \) is incremented with each 1/2 time interval, \( s_n \in \{\pm 1\} \), for \( n/2 \leq t < (n + 1)/2 \) and the pulse \( p_{\frac{1}{2}}(t) \) is defined by Eq. (2.23) for \( \tau = 1/2 \). Assume the solution \( u(t) \) to the differential equation (2.19) has the form

\[
u(t) = \sum_{n=-\infty}^{\infty} s_n P\left(t - \frac{n}{2}\right),
\]

(3.3)

where \( P(t) \) is a continuous basis function. Assume the solution is continuous and the initial condition is applied at a transition point (when the guard condition is met): \( u(0) = u_0 \) and \( \dot{u}(0) = 0 \). Since the forcing function is discontinuous with three regions, basis function must be piecewise continuous with three regions defined as,

\[
P(t) = \begin{cases} 
Q_1(t), & t \leq 0 \\
Q_2(t), & 0 < t \leq 1/2 \\
0, & t > 1/2 
\end{cases}
\]

(3.4)
where the third region \( t > 1/2 \) is forced to zero for existence. As shown in Appendix A, the half period basis function given in Eq. (2.31) for \( k = 1 \) is valid for all odd \( k \) since the constant \( \omega \) is only expended when matching boundary conditions between the three regions. In other words, for all odd \( k \) the basis function is given by

\[
P_{\text{odd } k}(t) = \begin{cases} 
(1 + e^{-\beta/2})e^{\beta t}[\cos(\omega t) - \frac{\beta}{\omega} \sin(\omega t)], & t \leq 0 \\
1 + e^{\beta(t-1/2)}[\cos(\omega t) - \frac{\beta}{\omega} \sin(\omega t)], & 0 < t \leq \frac{1}{2} \\
0, & t > \frac{1}{2} \end{cases} 
\]  

(3.5)

The basis function for all even \( k \), \( P_{\text{even } k}(t) \), may be found by solving Eq. (3.1) as

\[
P_{\text{even } k}(t) = \begin{cases} 
(1 - e^{-\beta/2})e^{\beta t}[\cos(\omega t) - \frac{\beta}{\omega} \sin(\omega t)], & t \leq 0 \\
1 - e^{\beta(t-1/2)}[\cos(\omega t) - \frac{\beta}{\omega} \sin(\omega t)], & 0 < t \leq \frac{1}{2} \\
0, & t > \frac{1}{2} \end{cases} 
\]  

(3.6)

The solutions of \( P(t) \) for even and odd values of \( k \) can be unified to give

\[
P_k(t) = \begin{cases} 
(1 - (-1)^k e^{-\beta/2})e^{\beta t}[\cos(\omega t) - \frac{\beta}{\omega} \sin(\omega t)], & t \leq 0 \\
1 - (-1)^k e^{\beta(t-1/2)}[\cos(\omega t) - \frac{\beta}{\omega} \sin(\omega t)], & 0 < t \leq \frac{1}{2} \\
0, & t > \frac{1}{2} \end{cases} 
\]  

(3.7)

The basis function \( P_k(t) \) has a constant pulse period no matter the value of \( k \) keeping a constant symbol size.
Since the oscillator has a fixed basis function, it also has a fixed matched filter. The matched filter may be constructed in a similar manner as the matched filter for the original full period oscillator. For a given \( k \), the impulse response, \( h_k(t) \) of the matched filter for the oscillator in Eq. (3.1) is given by

\[
h_k(t) = \alpha P_k(T-t),
\]

where \( \alpha \) is an amplitude scaling constant and \( T \) is the sample time. Similar to the matched filter for the original oscillator, the matched filter for the extended oscillator is a stable infinite impulse response filter.

**Figure 3.1:** Time series of \( u(t) \) and \( s(t) \) for \( k = 1, 2, 3, 4 \) started with the same arbitrary initial condition.
3.3 Return Map

A return map may be derived from the basis function solution. Since the symbols show regular timing on a $1/2$ period, the return map $u_n = u(n/2)$ for $n \in \mathbb{Z}$ will be considered. The key to the derivation is that the basis pulse shows the state, $u(t)$, is only dependent upon the current state and all future states since $P(t) = 0$ for $t > 1/2$.\(^1\) The state $u_n$ contains no information about previous states, but does contain information about the future states. This fact simplifies the derivation of the return map greatly. Solving the solution Eq. (3.3)

---

\(^1\)Mathematically, since $u$ is dependent only upon the current state and future state it is called a semi-flow(13).
for $t = (n + 1)/2$ gives

$$u_{n+1} = u\left(\frac{n + 1}{2}\right) = \sum_{m=-\infty}^{\infty} s_m P\left(\frac{n + 1 - m}{2}\right).$$ \hspace{1cm} (3.9)

Using the fact that the current state only depends on the current and future states, the bounds of the summation may be simplified to

$$u_{n+1} = \sum_{m=n+1}^{\infty} s_m P\left(\frac{n - m + 1}{2}\right).$$ \hspace{1cm} (3.10)

Solving the basis function for $t = (l + 1)/2$ for $l = 0, -1, -2...$ returns

$$P\left(\frac{l+1}{2}\right) = (1 - (-1)^k e^{-\beta/2}) e^{\beta(l+1)/2} (-1)^k (l+1) = (-1)^k e^{\beta/2} P\left(\frac{l}{2}\right).$$ \hspace{1cm} (3.11)

Substituting back into Eq. (3.10) gives

$$u_{n+1} = (-1)^k e^{\beta/2} \sum_{m=n+1}^{\infty} s_m P\left(\frac{n - m}{2}\right).$$ \hspace{1cm} (3.12)

Notice the right side of Eq. (3.12) is similar to the summation for $u_n$; it is only missing a scaled $s_n$ symbol which may be added and subtracted,

$$u_{n+1} = (-1)^k e^{\beta/2} \sum_{m=n}^{\infty} s_m P\left(\frac{n - m}{2}\right) - (-1)^k e^{\beta/2} s_n P(0).$$ \hspace{1cm} (3.13)

The second term on the right side of Eq. (3.13) is $u_n$. Substituting for $P(0)$ and simplifying Eq. (3.13) reveals the return map for the extended oscillator is given
by

\[ u_{n+1} = (-1)^k e^{\beta/2} u_n + s_n (1 - (-1)^k e^{\beta/2}). \]  

(3.14)

The return map given in Eq. (3.14) is a negative shift map for odd \( k \) and a positive shift map for even \( k \). The only way \( k \) effects the symbol sequence (and therefore the information content) is through being even or odd; i.e., for a given initial condition, the symbol sequences for all odd \( k \) are the same and for a given initial condition the symbol sequences for all even \( k \) are the same. This can be clearly seen in the time series shown in Figure 3.1. For \( k = 1, 2, 3, 4 \) the oscillator was started at an the same arbitrary initial condition. As a result, the forcing function \( s(t) \) has two wave forms dependent upon whether \( k \) is even or odd.

![Figure 3.3: Return map \( u_n \) for odd \( k \) on the left and even \( k \) on the right when \( \beta = \ln(4) \)].

The map is bounded by the interval

\[ \left[ -\left( \frac{e^{\beta/2} - (-1)^k}{e^{\beta/2} - 1} \right), \left( \frac{e^{\beta/2} - (-1)^k}{e^{\beta/2} - 1} \right) \right]. \]  

(3.15)
Notice for even $k$, the map is bounded by the fixed interval $[-1, 1]$ and for odd $k$ the interval ranges from $[-\infty, +\infty]$ for $\beta \to 0$ to $[-3, 3]$ for $\beta = \ln(4)$.

From the return map, the Lyapunov exponent can be found as $\Lambda = \ln(e^{\beta/2}) = \beta/2$; for the maximum growth rate, $\beta = \ln(4)$, the Lyapunov exponent is $\Lambda = \ln(2)$. While the original full period system also had a Lyapunov exponent of $\Lambda = \ln(2)$, it is important to note the time scale difference. The extended Lyapunov exponent is $\Lambda = \ln(2)$ per half time period whereas the original Lyapunov exponent is $\Lambda = \ln(2)$ for a full time period. Similarly, the topological entropy for the maximum growth rate, $\beta = \ln(4)$, is 1 bit per half time period whereas in the original system the topological entropy for the maximum growth rate is 1 bit per full time period. Thus, the half period system has twice the symbol rate as the original full period system. For a full comparison, please see Appendix A.

So far, all the mathematics of the extended exactly solvable chaotic oscillator for $k \in \mathbb{N}_1$ have been developed. The oscillator admits a closed form basis function solution and a return map. Now, implementing the oscillator in hardware can be examined.

### 3.4 Hardware Implementation

The following section examines an audio frequency, discrete hardware implementation of the extended chaotic oscillator. As a hybrid differential equation of both a continuous ($u$) and discrete ($s$) state, the hardware implementation of the oscillator is a hybrid of both analog and digital components as shown in Figure 3.4. The continuous state $u$ is implemented using an analog -RLC filter. The
digital forcing function and guard condition are implemented using an FPGA. Then, the two domains were married together using comparators (analog to digital) and buffers (digital to analog). In the following sections, the analog and digital portions will be considered separately.

3.4.1 Analog

The analog portion of the oscillator implements the continuous state \( u(t) \) with a -RLC filter is shown in Figure 3.4. The next subsection will consider how to implement \( k \) in the filter. Then the practical implementations of the negative resistor and inductor will be examined before moving onto the digital portions of the oscillator.

Figure 3.4: Oscillator Schematic.
3.4.1.1 Natural Oscillation Frequency vs. Time Period Per Symbol

As shown in Section 3.2, changing $k$ changes the frequency of the sinusoids, but to change the natural oscillation frequency of a -RLC filter would require the inductor and capacitor to change. For practical purposes, this is undesirable. An alternate implementation method arises from the guard condition in Eq. (3.1). Rather than change the natural oscillation frequency, keep a constant natural oscillation frequency and change the time period of the symbol. That is, keep the inductor and capacitor the same while still adjusting the maximum value of the counter. The trick is that since the relative time scale is changing, the growth rate $\beta$ must also change with the time scale. To see how this works, consider solving the unstable -RLC filter in Figure 3.4 to get

$$\ddot{u} - \frac{1}{RC}\dot{u} + \frac{1}{LC}(u(t) - s(t)) = 0.$$  \hspace{1cm} (3.16)

By matching the coefficients of Eq. (3.16) to the coefficients of the ideal ODE given in Eq. (3.1), the constants $\beta$ and $\omega$ can be related to the circuit components by

$$\beta_s = \frac{1}{2RC}$$  \hspace{1cm} (3.17)

and

$$\omega_s = \sqrt{\frac{1}{LC} - \beta_s^2}.$$  \hspace{1cm} (3.18)

One subtle point is in the units of $\beta$ and $\omega$. Up till now, all the math has been computed with an implied unit for $\beta$ of $1/symbol$ and $\omega$ of $radians/symbol$. Eqs.
(3.17, 3.18) have units of 1/s and radians/s for $\beta_s$ and $\omega_s$ respectively. The mathematical constants $\beta$ and $\omega$ can be related to their physical realizations by

$$\beta_s = \beta f$$

and

$$\omega_s = \omega f,$$

where $f$ is the oscillation frequency of the oscillator in symbol/s. The frequency $f$ can be found by utilizing $\omega_s = 2\pi k f$ and solving Eq. (3.18) to find

$$f = \sqrt{\frac{1}{LC(4\pi^2 k^2 + \beta^2)}}.$$  

(3.21)

Since $0 < \beta^2 \leq ln(4)^2 < 4\pi^2 k^2 \\forall k$, $f$ can be approximated by

$$f \approx \frac{1}{2\pi k \sqrt{LC}}.$$  

(3.22)

It is rather inconvenient to measure frequency in symbol/s, so $f$ can be converted to $f_s$ with units of Hz by

$$f_s = kf,$$

(3.23)

and now $f_s$ and $\omega_s$ are related by $\omega_s = \omega f = 2\pi f_s$. With all the units straightened out, the goal of changing the symbol period rather than the natural oscillation frequency with $k$ can be addressed. To set $\omega_s$, let $L$ and $C$ be fixed constants.
such that
\[ \omega_s = \omega_f \approx \frac{2\pi k}{2\pi k \sqrt{LC}} = \frac{1}{\sqrt{LC}}. \]  
(3.24)

Thus, the natural oscillation frequency is fixed, but to change the symbol period, the growth rate per second, \( \beta_s \), must be adjusted to account for \( k \). Assuming the growth rate per symbol, \( \beta \), and the oscillations per second, \( f_s \), are fixed, the relation between the growth rate per second and \( k \) can be found as
\[ \beta = \frac{\beta_s}{f} = \frac{\beta_s k}{f_s}. \]  
(3.25)

From Eq. (3.25), to keep a constant \( \beta \), the growth rate per second, \( \beta_s \), must be inversely proportional to \( k \). As \( k \) increases and the length of the symbol in seconds increases, the growth rate per second must decrease, otherwise the system will be unstable. The growth rate per second, \( \beta_s \) can be easily adjusted in hardware as shown by substituting for \( \beta_s \) in Eq. (3.25) to give
\[ \beta = \frac{\beta_s}{f} = \frac{k}{f_s RC}. \]  
(3.26)

To keep a constant \( \beta \), set \( R \propto k \). Thus, by fixing \( L \) and \( C \) to set \( \omega_s \), \( R \) can be used to maintain a constant growth rate \( \beta \) as \( k \) is adjusted by counting the zero crossings of the derivative digitally. Now, the circuitry of the -RLC filter may be examined more carefully beginning with the negative resistor.
3.4.1.2 Negative Resistor

Since the second order oscillator is unstable, energy needs to be injected into the system which can be done readily through a negative resistor. A normal positive resistor, consumes current as a voltage is applied across it. A negative resistor pushes current back (generates current) as a voltage is applied across it, adding energy into the system. The schematic for a negative resistor is given in Figure 3.5. For $R_1 = R_2$ the impedance looking into the input is $-R$.

![Negative Resistor Schematic](image)

**Figure 3.5:** Negative Resistor.

3.4.1.3 General Impedance Converter

Using a real inductor in the oscillator has two disadvantages: first, it can be hard to buy or build inductors with specific values and second, large inductors (such as those needed for audio frequencies) can have a large parasitic series resistances. Both issues can be solved using a General Impedance Converter (GIC). GIC’s can be easily built for large, low-frequency, inductances.
Consider the GIC shown in Figure 3.6. From the virtual grounds, the voltages at each port are equivalent: \( V_1 = V_2 \). Solving the circuits for the currents gives the relation

\[
I_1 = \frac{Z_2Z_4}{Z_1Z_3} I_2 = f(s) I_2. \tag{3.27}
\]

Traditionally, the GIC is used in a grounded configuration where one of the ports is connected to an impedance \( Z \) which is terminated to ground (28). For example, if port 2 of the GIC is connected to the grounded impedance \( Z \), the impedance looking into port 1 is \( f(s)Z \). However, for this oscillator consider using the GIC as shown in Figure 3.7. The function \( f(s) \) can be found as

\[
f(s) = \frac{R_2R_4}{(1/C)s}R_3 = Ks. \tag{3.28}
\]

Solving the circuit in Figure 3.7 for \( I_2 \) gives

\[
I_2 = \frac{1}{f(s)} I_1 = \frac{S(s) - u}{R_0f(s)} = \frac{S(s) - u}{R_0Ks}. \tag{3.29}
\]
The voltage at node $u$ can be found as

$$u = \frac{-R/Cs}{-R + 1/Cs} I_2 = \frac{-RI_2}{1 - RCs}. \quad (3.30)$$

Let $R_0K = L$ and solve for $u$ using $I_2$ to give

$$s^2u - \frac{1}{RC}s + \frac{1}{LC}(u - S(s)) = 0, \quad (3.31)$$

which after transforming to the time domain is the same as the differential equation in Eq. (3.16). This shows that for $L = KR_0$, the GIC is a valid replacement for the inductor.\(^2\) Notice that for ideal op-amps, the GIC has no series resistance.

With all the analog components of the -RLC filter, now the digital parts can be considered.

\(^2\)From the current relation given in Eq. (3.27), it is evident that the floating GIC is not symmetric and the impedance looking into $R_0$ from the forcing function is not simple. See Appendix F for the impedance viewed by the forcing function.
3.4.2 Digital

The digital portions of the oscillator were implemented using an FPGA. As shown in Figure 3.4, the analog signals $u$ and $du$ are compared against ground to create digital signals $\overline{u}$ and $\overline{du}$. The digital signals are then fed into the FPGA which is composed of a few parts: a one shot module to convert the positive and negative edges triggers of the digital $\overline{du}$ signal to pulses, a special zero-delay debounce filter to prevent multiple triggers, a counter with a parameter input of $k$, and finally a single D flip flop to sample and hold the sign of $\overline{u}$ at the $k^{th}$ trigger. The output of the D flip flop is the digital signal $\overline{s}$ which is then buffered to $\pm 1$ volt and fed back into the analog filter completing the oscillator feedback loop.

3.4.2.1 One Shot

The one shot, shown in Figure 3.8, converts positive and negative edges to small pulses. The one shot works by XOR’ing the input against a delayed version of itself. In the small time period where the undelayed input to the XOR differs from the delayed input, the output of the XOR is high and at all other times the output of the XOR is low. The delay line is comprised of a large, even number of inverters and the cumulative propagation delay creates the delay. Inverters are used instead of buffers to minimize pulse length distortion (29). Since the implementation optimizer will remove unnecessary gates, the optimizer is disabled for the delay line. The Verilog code for the delay line (and Xilinx
optimizer disable code) is given in Appendix E.2 and the full one shot Verilog code is given in Appendix E.3.

![M NOT gates](image)

**Figure 3.8:** One Shot pulse generator with $M$ NOT gates to create a delay line where $M$ is an even integer.

### 3.4.2.2 Debounce Filter

In the oscillator equation given in Eq. (3.1), the guard condition assumes the sign of $u$ is sampled on the instant the guard condition is fulfilled. Of course, that is impossible because all electronics have some inherent delay, but an effort must be made to reduce the delay between the guard condition trigger and the sample time as much as possible. The output of the $du$ comparator is naturally noisy which causes the one shot module to output multiple pulses at each zero crossing. Most debouncing schemes use averaging to remove false triggers, but averaging naturally introduces delay which can cause incorrect sampling of the sign of $u$.

To solve this, the assumption is made that the first pulse from the output of the one-shot, in a certain period of time, is correct and all others in that time period can be ignored. This is a reasonable assumption since the signal is a sinusoid which crosses zero at a set period. That is, the approximately fixed timing of the derivative zero-crossings can be used utilized in a filter design. The
Figure 3.9: Debounce filter timing diagram.

timing diagram for the filter is shown in Figure 3.9. The filter operates by first waiting for a pulse. In the waiting phase, the output of the filter is wired to the input of the filter so that the first pulse can pass un-delayed. Upon the negative edge of the first pulse, the block register is set to high and a timer is started. When the block register is high, the output of the filter is set to zero. When the timer finishes, the block register is set to low and the filter is reset to await another pulse. For a detailed operational description and the Verilog code, please see Appendix E.1. With the digital components addressed, now everything can be put together.

3.4.3 Implementation Summary

A detailed schematic of the analog circuitry is given in Figure 3.10. The TL082 op-amp was selected to be used for all op-amps in the circuit and the LM339 comparator was used for all comparators. The rails for the op-amps and comparators were set to \( \pm 15V \). No scaling of the math equations was required since for \( \beta \) close to \( \ln(4) \), the map is bounded well below the rail voltages of \( \pm 15V \) (see Eq. (3.15)). The Xilinx Artix-7 FPGA on a Digilent Arty-A7 development board was selected (30). To aid in protecting the FPGA, SN74LVC8T
Figure 3.10: Detailed analog schematic of extended second order solvable chaotic oscillator.
digital buffers were added in between the comparators and the FPGA. Using the component values as shown in Figure 3.10, the GIC has an inductance of 26mH looking in from node $u$. The oscillator has a theoretical oscillation frequency of $\omega_s \approx 28,000 \text{ radians/s}$ or $f_s \approx 4.5\text{kHz}$. The delay through the FPGA from a guard condition trigger to change in the forcing function was measured to be approximately 1ns. Since the oscillator is operating at a period of $1/f_s \approx 222\mu\text{s}$, the delay through the FPGA is negligible.

### 3.5 Hardware Results

The hardware oscillator was tested for multiple values of $k$ (adjusting $\beta$ with the negative resistor to maintain approximately the same growth, $\beta$). Figure 3.11 shows a snippet of the time series for $k = 1, 2, 3, 4$. Since the hardware changed the time period per symbol with $k$ rather than the natural oscillation frequency, to illustrate the changing oscillation frequency per symbol period the length of time series displayed has been normalized to $k$. That is, for $k = 1$, 3ms is shown and for $k = 2$, 6ms is shown, etc. After normalization the symbol pulse size is approximately the same for all values of $k$.

The oscillator had a measured oscillation frequency of approximately 4.2kHz, differing from the expected frequency of 4.5kHz by 300Hz. The difference can be accounted for by capacitive parasitics and component tolerances. Using an LCR meter to measure the capacitance between the node $u$ and ground reveals a capacitance of 55nF compared to the theoretical capacitance of 47nF. Using the
measured capacitance of 55nF in Eq. (3.24) gives a theoretical oscillation frequency of 4.2kHz, matching the measured oscillation frequency.

The maps for the four different values of $k$ are shown in Figure 3.12. For the odd values of $k$, the maps are negative shift maps and for the even values of $k$ the maps are positive shift maps as described by Eq. (3.14). The fixed points of (3.1) can be seen in the maps at $(\pm 1, \pm 1)$ from the discontinuous breaks for the odd valued $k$’s and by the slight curving in the maps for the even values of $k$. Around the fixed point, the oscillations are vanishingly small so the hardware struggles to amplify the small changes; hence the discontinuities in the maps for odd $k$ and the curving in the maps for even $k$. The effect of the fixed points can be seen in the time series as small perturbations in the period of the symbol when the state $u_n \approx \pm 1$. The closer the state $u_n$ is to $\pm 1$ the greater the distortion.

**Figure 3.11:** Time series of theoretical analytic solution ($u_{theo}$) and measured hardware oscillator ($u_{meas}$) for various values of $k$. 

55
Figure 3.12: Maps from hardware for various values of $k$. 
Chapter 4. Conclusion

In this thesis, the small set of chaotic systems with analytic solutions was enlarged by extending a known, second order solvable chaotic oscillator. In the original second order solvable chaotic oscillator, the natural oscillation frequency was directly tied to the symbol switching rate. In this work, the oscillator was extended to allow the natural oscillation frequency, $\omega$, to take any integer multiple of $2\pi$ (i.e., $\omega = 2\pi k$ for $k \in \mathbb{N}_1$), relative to the switching time period. A new analytic solution and basis function based on a half period symbol time were developed for the extended oscillator in addition to a new return map. The symbols generated by the oscillator were shown to only depend upon whether $k$ was even or odd and thus changing the natural oscillation frequency of the oscillator does not effect the information coding.

Then, the oscillator was implemented in hardware using a discrete analog negative RLC filter and a digital FPGA. For convenience, rather than changing the absolute natural oscillation frequency to change with $k$, the time period per symbol was changed digitally on the FPGA. Thus, the natural oscillation frequency relative to the symbol period could be easily adjusted without needing to change the inductor and capacitor. Multiple time series were taken from the hardware for various values of $k$ and both time series and return maps of the data.
were shown. From the maps, the hardware results closely match the theoretical return maps though the hardware naturally struggles around the fixed points.

Future work will focus on the properties of the extended oscillator such as orthogonality. It can be shown that the derivative of the basis function given in Eq. (3.7), $P'_k(t)$, is orthogonal for different values of $k$ over the interval $[0, 1/2]$ with respect to the weight function $e^{-2\beta t}$; this gives justification to explore the existence of larger regions and different weight functions of orthogonality. Orthogonality also opens interesting avenues of new waveforms such as the superposition of multiple oscillators with different $k$ values. Further, orthogonality of the derivative gives inklings of possible relations to larger generalities such as eigenfunctions that may be explored.
References


Appendix A. Full vs. Half Period Solutions

In the original A Matched Filter for Chaos(6) paper β was limited to $0 < \beta \leq \ln(2)$ which leads to a solution based on a full period symbol width. Later, a half period basis function was published that is a solution to the same oscillator and extends β to $0 < \beta \leq \ln(4)$. Both the full and half period solutions are valid and the full period solution is actually a linear superposition of the half period solution. The solutions can be compared in two ways: from the basis functions or from the shift maps. Both are presented here.

A.1 Basis Function Comparison

First note that $P_{\text{orig}}(t)$ is twice as long as $P(t)$. As shown in the original solution (6), by limiting $\beta \leq \ln(2)$, even though the guard condition is met at every half time period, the forcing function $s_n = \text{sgn}(u_n)$ only changes every other time the guard condition is met; i.e., at every full time period. The symbol rate of the original system is half that of the extended system. The original basis function can be derived from the summation of the extended basis function, $P(t)$, and $P(t)$ shifted by $1/2$ as plotted in Figure A.1 and given by

$$P_{\text{orig}}(t) = P(t) + P\left(t - \frac{1}{2}\right). \quad (A.1)$$
That is, the full period solution is the linear superposition with unit weights of two half period basis functions. The relation given in Eq. (A.1) suggests some grammar restriction in the relation between the original and extended solutions and indeed, a grammar restriction may be seen in the shift maps.

**Figure A.1:** The linear superposition of two half period basis functions to give one full period basis function.

### A.2 Shift Map Comparison

To compare the original and extended shift maps, one must realize they are based on two different time scales: the original shift map comes from sampling at every 1 time period and the extended solution comes from sampling at every 1/2 time period. Both shift maps in their respective time scale, are two symbol systems \((s_n \in \{+1, -1\})\) and have a maximum entropy of one bit. However, to adequately compare them, the same time scale must be maintained. For the following analysis, the time scale of 1 time period is arbitrarily elected and the extended shift map is re-coded over the new time scale.
Let $v_m$ be a two dimension vector of symbols $s_n \in \{-1, +1\}$ such that $v_m \in \{[s_n, s_{n+1}]\}$ and $n = 2m$ for $m \in \mathbb{Z}$. Let the symbols $s_n$ be defined by the half period shift map given in Eq. (3.14) for $k = 1$. The shift map $v_m$ is a new coding of the new extended shift map over 1 time period. Notice that $v_m$ is a four symbol system and has a maximum entropy of 2 bits in one time period. In the same time period as the original shift map $s_n^{\text{orig}}$, the symbol rate has doubled.

To see the relation between the original shift map and the extended shift map, consider the Subshift of Finite Type (SFT) (15), $v_m^F$ with the forbidden symbols $F = \{[-1, 1], [1, -1]\}$. In other words, apply a grammar restriction upon $v_m$ such that it only uses the symbols $\{[-1, -1], [1, 1]\}$ or $s_n = s_{n+1}$. With this grammar restriction, the recursive equation for $u_n$ may be simplified by substituting $u_{n+1}$ into $u_{n+2}$.

\begin{align}
  u_{n+2} &= -\left[-u_ne^{\beta/2} + (e^{\beta/2} + 1)s_n\right]e^{\beta/2} + (e^{\beta/2} + 1)s_{n+1} \\
  u_{n+2} &= u_ne^\beta - (e^\beta - 1)s_n. \quad (A.2)
\end{align}

It is evident that $u_n^{\text{orig}} = u_n$ for $n = 2m$ and restricting the grammar such that $s_n = s_{n+1} = s_n^{\text{orig}}$. Thus, the original shift map can be consider a subshift of finite type of the extended solution reducing the symbol rate and maximum entropy by half.

Whether viewed from the basis functions or from the shift maps by restricting the $\beta \leq \ln(2)$, the original solution reduces the symbol rate to half of maxi-
mum rate via restricting symbols to only appear in pairs. There are four possible
two bit words available to the extended system: \{[−1, −1], [−1, 1] [1, −1] [1, 1]\},
but the original solution can only produce two of those words: \{[−1, −1], [1, 1]\}.
Thus it may be argued since the original solution is an SFT of the extended
solution, the extended solution is more general.
Appendix B. Detailed Recursive Solution

In the following appendix, a recursive solution for the extended half period higher \( k \) oscillator is presented similar in style to the recursive solution in A Matched Filter for Chaos (6) for the full period \( k = 1 \) oscillator. The recursive solution provides an alternate method to deriving the return map for the oscillator. To simplify the derivation, first consider \( k = 1 \) for the hybrid differential equation of a continuous state \( u(t) \in \mathbb{R} \) and a discrete state \( s(t) \in \{\pm 1\} \) given in Eq. (3.1) and reprinted here for convenience:

\[
\ddot{u} - 2\beta \dot{u} + (\omega^2 + \beta^2)(u - s) = 0, \quad (B.1)
\]

where \( \omega \) and \( \beta \) are fixed parameters. Let \( \omega = 2k\pi, \ 0 < \beta \leq \ln(4) \) and for now only consider \( k = 1 \). Since \( k \) is limited to one, the guard guard condition can be simplified to

\[
\dot{u}(t) = 0 \rightarrow s(t) = \text{sgn}(u(t)). \quad (B.2)
\]

To analyze the evolution of Eq. (B.1) through time, let the initial conditions be, \( u(0) = u_0, \ \dot{u}(0) = 0, \) and \( s_0 = \text{sgn}(u_0) \). Examination of Eq. (3.1) reveals a trivial fixed point of \( |u_0| = 1 \) and \( s(t) = s_0 \) which will be ignored for now, but does show itself in the hardware data in Section 3.5. For the given initial
conditions, \( u(t) \) may be solved from \( t = 0 \) until the first guard condition is met to give

\[
u(t) = s_0 + (u_0 - s_0)e^{\beta t} \left( \cos(\omega t) - \frac{\beta}{\omega} \sin(\omega t) \right) . \tag{B.3}
\]

To find when the next guard condition is met, the derivative of Eq. (B.3) may be taken and set to equal to zero

\[
\dot{u}(t) = -\frac{\omega^2 + \beta^2}{\omega} (u_0 - s_0)e^{\beta t} \sin(\omega t) = 0. \tag{B.4}
\]

Solving Eq. B.4 for \( t \) reveals the first the guard condition is met when \( t = 1/2 \).

To find the forcing function, Eq. (B.3) is solved for \( t = 1/2 \),

\[
u(1/2) = -u_0 + s_0(1 + e^{\beta/2}). \tag{B.5}
\]

In the original full period recursive solution(6), the growth rate, \( \beta \), was limited to \( 0 < \beta \leq \ln(2) \) which meant the forcing function could not change over a 1/2 time period: \( \text{sgn}(u(1/2)) = \text{sgn}(u(0)) = s_0 \). However, by extending \( \beta \) to \( 0 < \beta \leq \ln(4) \), the sign of \( u(1/2) \) may have changed so \( \text{sgn}(u(1/2)) \) may not be equal to \( \text{sgn}(u(0)) \). Thus the discrete state \( s(t) \) must be updated. To simplify updating the discrete state, set the continuous state \( u_1 \) such that

\[
u_1 = u(1/2) \tag{B.6}
\]
and then the discrete state, $s_1$, can be set to

$$s_1 = s(1/2) = \text{sgn}(u_1). \quad (B.7)$$

Since this state change occurred when $\dot{u}(t) = 0$, the time-invariant property of the solution Eq. (B.3) can be used to shift Eq. (B.3) by a time interval of $\Delta t = 1/2$ and update the initial conditions $s_0$ to $s_1$ and $u_0$ to $u_1$. This results in the solution

$$u(t) = s_1 + (u_1 - s_1)e^{\beta(t-\frac{1}{2})}$$

$$\times \left( \cos(\omega(t - \frac{1}{2})) - \frac{\beta}{\omega} \sin(\omega(t - \frac{1}{2})) \right). \quad (B.8)$$

Eq. (B.8) is valid from $t = 1/2$ to the next guard condition. The next guard condition can be found by the same process of taking the derivative of Eq. (B.8) and solving $t$ when $\dot{u}(t) = 0$ to give

$$\dot{u}(t) = -\frac{\omega^2 + \beta^2}{\omega} (u_1 - s_1)e^{\beta(t-1/2)} \sin(\omega t - \pi) = 0. \quad (B.9)$$

This reveals the next guard condition occurs when $t = 1$. To find the forcing function, Eq. (B.8) can be solved at $t = 1$ to give a new states $u_2$ and $s_2$:

$$u_2 = u(1) = -u_1 + s_1(1 + e^{\beta/2}) \quad \text{(B.10)}$$

$$s_2 = \text{sgn}(u_2). \quad \text{(B.11)}$$
With the new states $s_2$ and $u_2$, the solution in Eq. (B.8) can be shifted to be valid over the interval $[1, 3/2]$. The pattern of shifting and iterating can continue indefinitely on every 1/2 time period since the guard condition is met at $t = n/2$ for $n = 0, 1, 2,...$. For a set of initial conditions $(u_0, s_0)$, the solution $u(t)$ may be written recursively as

$$u(t) = s_n + (u_n - s_n)e^{\beta(t - \frac{n}{2})} \times \left( \cos(\omega(t - \frac{n}{2})) - \frac{\beta}{\omega} \sin(\omega(t - \frac{n}{2})) \right), \quad (B.12)$$

which is valid for $n/2 \leq t < (n + 1)/2$ and with the discrete iterates for $n$ defined by

$$u_{n+1} = -u_ne^{\beta/2} + s_n(1 + e^{\beta/2}) \quad (B.13)$$

$$s_{n+1} = \text{sgn}(u_{n+1}). \quad (B.14)$$

Consider when $k > 1$ and the full guard condition given in Eq. (3.1). For all $k$, the symbol width is a 1/2 time period, but the natural oscillation frequency changes. At no point in the derivation of Eq. (B.12) was $\omega$ expanded and simplified for $k = 1$ so Eq. (B.12) is valid for $k \in \mathbb{N}_1$. The return map given in Eq. (B.13) was simplified by expanding $\omega$ in terms of $k$. To find the return map for all $k$, consider solving Eq. (B.12) for $t = (n + 1)/2$ to find $u((n + 1)/2) = u_{n+1}$,

$$u_{n+1} = s_n + (u_n - s_n)e^{\beta/2} \left( \cos\left(\frac{\omega}{2}\right) - \frac{\beta}{\omega} \sin\left(\frac{\omega}{2}\right) \right). \quad (B.15)$$
For $\omega = 2\pi k$ and $k = 1, 2, 3, \ldots$, the sinusoids can be simplified by $\sin\left(\frac{\omega}{2}\right) = 0 \forall k$ and $\cos\left(\frac{\omega}{2}\right) = (-1)^k$ to give the return map,

$$u_{n+1} = (-1)^k u_n e^{\beta/2} + s_n (1 - (-1)^k e^{\beta/2}). \quad (B.16)$$

The return map in Eq. (B.16) is identical to the return map given in Eq. (3.14) derived from the basis function providing a second derivation method for the return map.
Appendix C. Detailed Basis Function Solution

Herein the derivation of the basis function for the extended solvable chaotic oscillator is presented in detail. Consider the oscillator equation given in Eq. (3.1). Since the guard condition is met at regular timings of $1/2$ period, the forcing function can be defined as a series of symmetric rectangular pulses defined by Eq. (3.2) and reproduced here for convenience:

$$s(t) = \sum_{n=-\infty}^{\infty} s_n p_{1/2} \left( t - \frac{n}{2} \right).$$

The pulse, $p_{1/2}(t)$, is defined by Eq. (2.23) for $\tau = 1/2$ and $n$ is incremented with each $1/2$ time interval. The symbols $s_n \in \{\pm 1\}$ are defined by $s_n = s(n)$ for $n/2 \leq t < (n + 1)/2$. The symbols may be found from an initial condition $u_0$ by iterating the return map given in Eq. (3.14). Now assume the solution $u(t)$ to the differential equation (3.1) has the form given in Eq. (2.31):

$$u(t) = \sum_{n=-\infty}^{\infty} s_n P \left( t - \frac{n}{2} \right),$$

where $P(t)$ is a continuous basis function. Assume the solution is continuous and the initial condition is applied at a transition point (when the guard condition is met): $u(0) = u_0$ and $\dot{u}(0) = 0$. Substituting the proposed solution (C.2) into the
differential equation given in Eq. (3.1) gives
\[ \sum_{n=-\infty}^{\infty} \ddot{P} \left( t - \frac{n}{2} \right) - 2\beta \dot{P} \left( t - \frac{n}{2} \right) + \left( \omega^2 + \beta^2 \right) \left[ P \left( t - \frac{n}{2} \right) - p_{\frac{1}{2}} \left( t - \frac{n}{2} \right) \right] = 0. \]

Since this is a time invariant system and every new transition \( s_n \) occurs only at the guard condition when the \( k^{th} \) \( \dot{u}(t) = 0 \), only one item in the summation needs to be considered. That is,
\[ \ddot{P}(t) - 2\beta \dot{P}(t) + \left( \omega^2 + \beta^2 \right)(P(t) - p_{\frac{1}{2}}(t)) = 0. \]  
(C.3)

Since the pulse \( p_{\frac{1}{2}}(t) \) is piecewise discontinuous, all three regions must be solved for separately. Let the three solution regions be defined as
\[ P(t) = \begin{cases} 
Q_1(t), & t \leq 0 \\
Q_2(t), & 0 < t \leq 1/2 \\
0, & t > 1/2 
\end{cases}. \]  
(C.4)

To ensure \( P(t) \) can exist in real life (i.e., it is bounded), the last region \( t > 1/2 \) is forced to zero. Consider the first region, \( t \leq 0 \). For \( t \leq 0 \), \( p_{\frac{1}{2}}(t) = 0 \), the differential equation for the first region simplifies to a homogeneous equation:
\[ \ddot{Q}_1(t) - 2\beta \dot{Q}_1(t) + \left( \omega^2 + \beta^2 \right)Q_1(t) = 0, \]  
(C.5)
which admits the general solution,

\[ Q_1(t) = \alpha_1 e^{(\beta + j\omega)t} + \alpha_2 e^{(\beta - j\omega)t}. \]  

(C.6)

Since the initial condition at some time \( t_0 \) is applied at a guard condition \( \dot{u}(t_0) = 0 \) and \( u(t_0) = c_1 \) where \( c_1 \) is some constant, the solution may be simplified to

\[ Q_1(t) = c_1 e^{\beta t} [\cos(\omega t) - \frac{\beta}{\omega} \sin(\omega t)]. \]  

(C.7)

Now consider the second region, \( Q_2 \). For \( 0 < t \leq 1/2 \), the pulse \( p_2(t) = 1 \) so Eq. (C.3) simplifies to

\[ \ddot{Q}_2(t) - 2\beta \dot{Q}_2(t) + (\omega^2 + \beta^2)(Q_2(t) - 1) = 0, \]  

(C.8)

which after applying the initial conditions \( \dot{u}(t_0 + 1/2) = 0 \) and \( u(t_0) = c_2 \), admits the solution

\[ Q_2(t) = 1 + c_2 e^{\beta t} [\cos(\omega t) - \frac{\beta}{\omega} \sin(\omega t)], \]  

(C.9)

where \( c_2 \) is some constant. Since \( P(t) \) must be continuous, the constants \( c_1 \) and \( c_2 \) must be used to match the solutions at the region boundaries of \( t = 0 \) and \( t = 1/2 \). For simplification, \( \omega \) is expanded in terms of \( k \) and since the sinusoidal functions are only dependent upon even/odd \( k \), the basis function may be written
as

\[ P_k(t) = \begin{cases} 
(1 - (-1)^k e^{-\beta/2})e^{\beta t}[\cos(\omega t) - \frac{\beta}{\omega} \sin(\omega t)], & t \leq 0 \\
1 - (-1)^k e^{\beta(t-1/2)}[\cos(\omega t) - \frac{\beta}{\omega} \sin(\omega t)], & 0 < t \leq \frac{1}{2} \\
0, & t > \frac{1}{2}
\end{cases} \]  \hspace{1cm} (C.10)

This completes the derivation of the half period basis function for \( k \in \mathbb{N}_1 \).
Appendix D. Partitions

Partitions are the glue that connects continuous dynamical systems with symbolic dynamical systems. They facilitate many different useful calculations and theorems to provide insight into how a dynamical system functions. However, there are many important details to the different partition types and how to properly apply them. Three partition types will be discussed here: topological, Markov and generating partitions. The discussion will favor the English language over detailed topological mathematics. For more mathematical definitions, please see (13). Further the discussion will be limited to one dimension though all partition types may be extended to higher dimensions.

D.1 Topological Partition

A topological partition connects a continuous dynamical system to a symbolic dynamical system. Consider a one-dimensional system in $\mathbb{R}^1$ with a map $T$ that operates on the interval $I = [c, d]$ such that $T : I \to I$. A topological partition, $P$, divides the interval $I$ into $p$ intervals, $I_0, I_1, ... I_{p-1}$, such that $I_i = [c_i, c_{i+1}]$ for $c = c_0 < c_1 < ... c_p = d$. The union of the intervals must cover $I$. That is, $I_0 \cup I_1 \cup ... \cup I_{p-1} = I$ (13). The entire map must fall into one of the partition intervals; every point in the trajectory of the map must land in a
partition interval so that no points lie outside of $I$. An example of a topological partition for an arbitrary map is shown on the left in Figure D.1: notice that all points of the map are contained within a partition interval.

The topological partition connects the dynamical system to a symbolic dynamical system through coding. For example, consider the arbitrary map given on the left in Figure D.1. Let each partition be coded by a symbol such that when the map enters a partition, the symbolic dynamical system, $s$, produces a symbol such that $T \in I_1 \rightarrow s = A$, $T \in I_2 \rightarrow s = B$, $T \in I_3 \rightarrow s = C$ and $T \in I_4 \rightarrow s = D$. Thus, the trajectory of the map may be represented by the sequence of symbols, but in general the for a topological partition, the orbits of the map do not uniquely map to a symbol sequence. Different partitions will produce different symbol sequences for the same map trajectory and some partitions have beneficial properties. One property of great importance is memory which is quantified with Markov partitions.

**D.2 Markov Partition**

A Markov partition of a map $T$, is a topological partition where there exists a homeomorphism\(^1\), $\tau_i$, from each partition region $I_i$ to a union of partition regions. That is, each region maps continuously, onto, one-to-one and is invertable onto a union of other regions (13). For example, the map on the left in Figure D.1 is not a Markov. There are many points where the left partition in Figure D.1 violates the Markov requirements, but one example is that the partition interval

\(^1\)A homeomorphism is a continuous one-to-one function with a continuous inverse between two metric spaces(31).
$I_{n+1}$ only partially maps to itself, $I_2$, and thus is not bijective. On the other hand, the map on the right in Figure D.1 is a Markov partition of the same map.

The reason these partitions are called Markov is that similarly to probability theory, these partitions contain finite memory. In probability theory, the next probabilities of the outcomes of the next evolution Markov Random Process is given by the condition probabilities on the current state; the entire history of the process is contained in the current state. Notice in a Markov partition, the interval the map enters in the next evolution depends only on the current state.

**D.3 Generating Partition**

A generating partition maps every trajectory in the dynamical system to a unique symbol sequence in the conjugate symbolic dynamical system. That is, given a symbol sequence and a generating partition, one then knows the trajectory
of the map. Note, if a partition is Markov then it is also generating, but if
the partition is generating, it is not necessarily Markov. For example, since the
partition in Figure D.1 is Markov, then it is also generating (13).
Appendix E. Verilog

E.1 Debounce Filter

The debounce filter used in the oscillator cannot have any significant delay (such as would be present in a traditional averaging filter) because any delay can cause incorrect sampling. So to remove spurious pulses from the output of the one shot, the reasonable assumption is made that the first pulse is the desired pulse and all others within a certain period of time can be safely filtered. The verilog code for the filter is given in Listing 1 and Figure E.1 is a detailed timing diagram. On a high design level, the Xilinx Artix-7 FPGA (like most FPGA’s) only has single edge triggered flip flops which means it can only trigger on either the positive or negative edge of a given signal. To get double edge behavior, two registers that control each other must be used. In the verilog code, the block and clear block registers control each other to achieve double edge behavior. To understand the behavior better, consider the standard operation of the filter.

The core of the filter is the output set to the input and’ed with the inverse of the block register. If the block register is high, the output is set to low. At the start of a filter cycle, the filter waits for a pulse from the one shot: both the block and clear block control registers are low so the input can flow freely to the output. Upon the negative edge of the first pulse at the input, the block register is set to
high and the output is forced low no matter what the input is. Setting the block register high starts a counter that uses an internal 10MHz clock on the FPGA. When the counter finishes, the clear_block register is set to high. The positive edge of the clear_block register resets the block register opening the output to the input again. On the next counter clock cycle, the block register then resets the timer and the clear_block register finishing the filter cycle.

**Figure E.1:** Detailed debounce filter timing diagram.
module periodic_blocking_filter(
  pbf_in,
  pbf_out,
  clk
);

// IO
input wire pbf_in;
input wire clk;
output wire pbf_out;

// Internal Wiring
reg block = 0;
reg clear_block = 0;
parameter max = 4000; // delay = 10*max [ns]
integer wait_timer = 0;

// Assign the blocking...
assign pbf_out = pbf_in & ~block;

// Engage and disengage blocking
always @(negedge filter_input or posedge clear_block)
begin
  if (clear_block)
    block <= 0;
  else
    block <=1;
end

// Timer
always @(posedge clk)
begin
  if (~block)
    begin
      wait_timer <= 0;
      clear_block <= 0;
    end
  else if (wait_timer < max)
    wait_timer <= wait_timer + 1;
  else if (wait_timer >= max)
    clear_block <= 1;
end
endmodule

Listing E.1: Periodic Blocking Filter Verilog
E.2 Delay Line

The verilog for a delay line is given in Listing 2. The key to the delay line is the Xilinx “don’t touch” synthesis directive which prevents the optimizer from removing the thousands of inverters needed to create the delay. The optimizer is disabled per wire so line 14 in Listing 2 disables the optimizer for all the wires in between the inverters.

```verilog
module delay_line(
    s_in,
    s_out);

// IO
input wire s_in;
output wire s_out;

// Generation and Parameters
parameter n = 9001;
genvar i;

// Internal wiring; Xilinx Don’t Touch
(* dont_touch = "yes" *) wire [n-1:0] delay;

// Assign the inputs into and out of the generator
assign delay[0] = s_in;
assign s_out = delay[n-1];

// generate the delay
generate
  for (i = 0; i < n-1; i = i + 1)
    begin: generate_delay
      assign delay[i+1] = ~delay[i];
    end
endgenerate

endmodule
```

Listing E.2: Delay Line Verilog
E.3 One Shot

The one shot verilog module implements the schematic given in Section 3.4.2. The one shot module instantiates a delay line, then XOR’s the input with a delayed copy of the input.

```verilog
module one_shot(
    os_in, os_out,
);
  // IO
  input wire os_in;
  output wire os_out;
  // Internal
  wire xor_out;

  // Delay the input
  delay_line dl(
    .s_in(os_in),
    .s_out(dl_out)
  );

  // XOR the input with the output to create a pulse.
  assign xor_out = os_in ^ dl_out;

  assign os_out = xor_out;
endmodule
```

Listing E.3: One Shot Verilog
Appendix F. Reverse Impedance of Floating GIC

The floating GIC is a non-symmetric devise so the impedance looking into the GIC from the forcing function in Figure 3.7 is non-trivial. Consider the reciprocal GIC current relation given by

\[ I_1 = f(s)I_2 = KsI_2. \] (F.1)

From Figure 3.7, the impedance looking into \( R_0 \) from the forcing function may be found by solving for a test voltage \( V_s \) at the node of \( S(s) \) to get

\[ V_s = I_1R_0 + u = \frac{-RI_2}{1 - RCs}. \] (F.2)

Then the impedance looking into the GIC from \( R_0 \) is

\[ Z_{in s} = \frac{V_s}{I_1} = R_0 + \frac{-R}{(Ks)(1 - RCs)} = R_0 - \frac{R}{Ks} - \frac{R^2C}{K(1 - RCs)}. \] (F.3)

Thus the impedance, \( Z_{in s} \), looking into the GIC from the forcing function is not an inductor since the Laplace variable, \( s \), is only in the denominator.