Digital Design and Implementation of DCSK Modem

CND 111 Final Project.
Under Supervision of Prof. Khaled Mohamed.

Omar T. Amer[1], **Yahia A. Hatem**[2],
Mostafa M. Darwish[3] **Mennatullah Mohamed**[4]

[1]V23009946
[2]V23009902
[3]V23009976
[4]V23010369

Center of Nanoelectronics & Devices
American University in Cairo
Cairo, Egypt
12/2/2023

Abstract

TODO: Write Abstract.

Preface

Symbols

Symbol	Meaning
T_s	Symbol duration.
e_k	Transmitter output.
β	Number of chaos chips in a DCSK frame, 2β =Spreading factor.
k	Chip index.
x	Chaos chip.

Abbreviations

Abbreviation	Meaning
AWGN	Additive White Gaussian Noise
TX	Transmitter.
RX	Receiver.
MODEM	Modulator/Demodulator.
MAC	Multiply and Accumulate.
PISO	Parallel-In Serial-Out Shift Register.
SIPO	Serial-In Parallel-Out Shift Register.

Text Styles

HDL Module

Technical Specs

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter discusses the following:

- Introduction to Differential Chaos Shift-Keying.
- Logistic Map Overview.
- Proposed Architecture.

1.1 Introduction to DCSK

Differential chaos shift-keying (**DCSK**) is a modulation scheme based on using *chaos* to modulate a signal instead of a sinusoidal carrier. The reference signal (chaos) is sent over the channel, then after half a T_s the reference signal is modulated with the infromation signal and sent to the RX side. The composition of a DCSK frame can be seen in the Equation. 1.1.

$$e_k = \begin{cases} x_k & \text{for } 1 < k < \beta \\ s_i x_{k-\beta} & \text{for } \beta < k \leq 2\beta \end{cases} \tag{1.1}$$

Where β is an integer and k is the chip index.

e_k is sent over an AWGN channel, the signal at the RX side is given by:

$$r_{sig} = e_k + n_k \tag{1.2}$$

The received signal is correlated with a delayed version of itself (delayed by half a symbol's duartion) and then the sign of the result is the demodulated infromation bit.

1.2 An Overview of the Logistic Map

Many methods exists to generate a chaotic signal. The Logistic Map is the most simple to understand and is the least complex to implement. The Logistic Map is given by the following equation:

$$n_{i+1} = r \times n_i(1 - n_i) \tag{1.3}$$

Most values of r beyond ≈ 3.56995 exhibit chaotic behavior. We can see in Figure **??** The effect of different values of r on n_{i+1}.

insert diagram here

Chapter 2

Architecture

This chapter discusses the architecture of following modules:

- Transmitter.
  - Chaos Generator.
  - Modulator.
  - Transmitter Finite State Machine.
- Receiver.

In this chapter, we discuss the architecture of the modem, as well as the design choices made. Figure. 2 shows our proposed architecture to implement the modem. Figure. 2 shows the demodulator.

2.1 Transmitter

The modulator is split into three parts:

1. Chaos generator.
2. Modulator.
3. Serializer.

2.1.1 Chaos Generator [RE-WRITE]

The chaos is generated with a Logistic Map with $r = 4$. The value 4 was chosen to simplify the multiplication by r into a shift left by 2. The logistic map operates on Q2.6 numbers. From equation 1.3 we can see that if n_i is 8-bits in length, then n_{i+1} is 16-bits. This means that each cycle, the Logistic Map The multiplication is performed with the radix-4 Booth multiplication algorithm. As it can be seen from the block digaram is shown in Fig.**??**, all partial products are calculated in parallel meaning that each clock cycle, the module is capable of generating 16 bits of chaos.

However, calculating all four partial products in a single sycle each cycle consumes a lot of power. This can happen when the spreading factor is equal to 16. Also, 16-bits of chaos is not suffecient for fast data rates. To solve this, we need to calculate a large number of chaos bits in a short amount of time. The Chaos Expander takes 16 w y7awel l 256. kamel b2a

2.1.2 Modulator

The purpose of the Modulator is to modulate the chaos bits with the information bits. The message is first loaded into the Message PISO and is then serially output. The output bit is xor-ed with a delayed version of the chaos bit from Chaos Generator (delayed by β).

Table 2.1: Modulator Pin-Out

Name	Width	Direction	Description
`i_clk`	1	Input	Positive edge clock.
`i_arst_n`	1	Input	Active-low asynchronous reset.
`i_msg_bit`	1	Input	The input message bit.
`i_chaos_bit`	1	Input	The bit of chaos to be modulated.
`i_frame_half`	1	Input	Asserted if (and only if) the index of the chip being sent is less than β.
`i_sf`	2	Input	The encoded spreading factor.
`o_modulated_bit`	1	Output	The serial output of the Modulator.

Figure 2.1: Transmitter Block Diagram

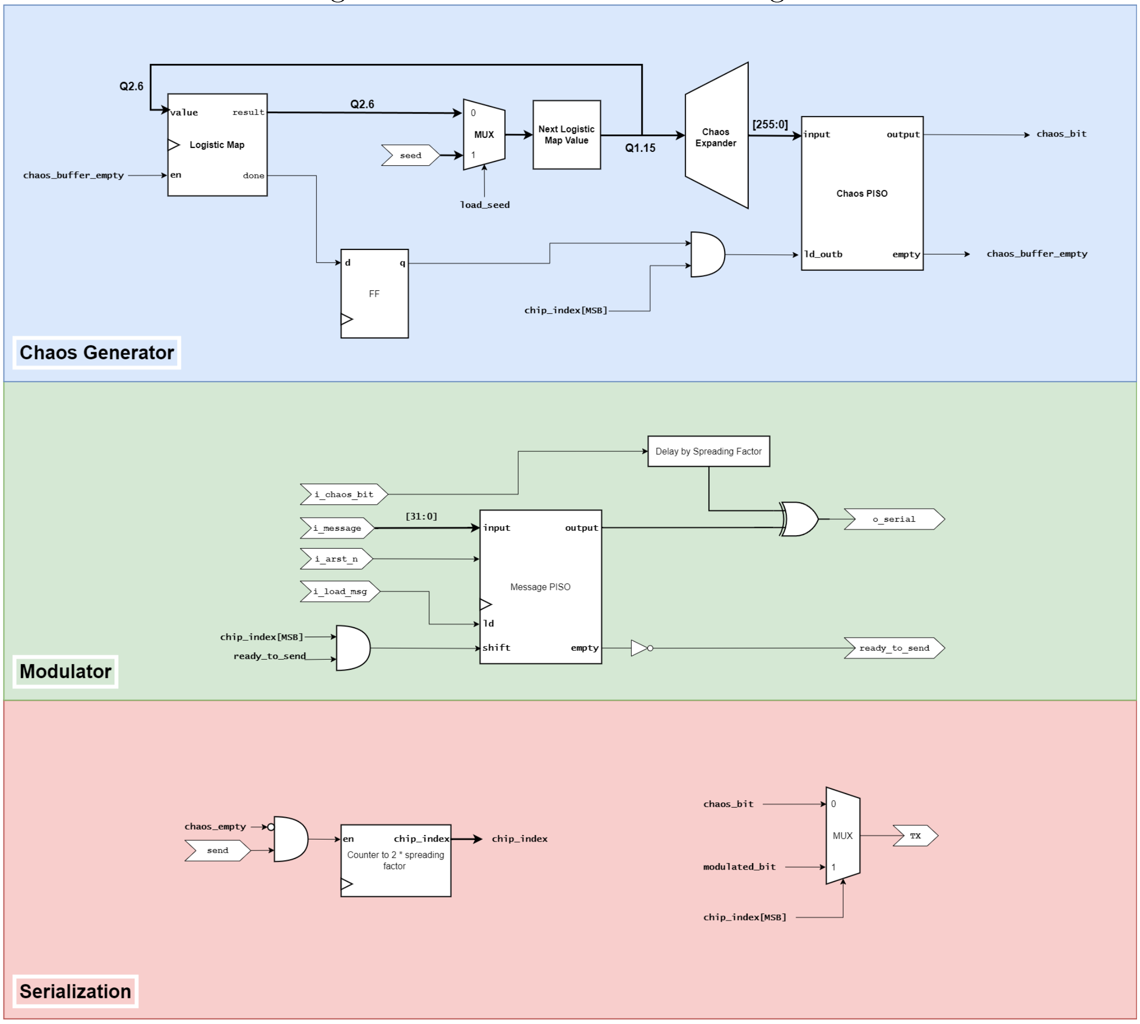
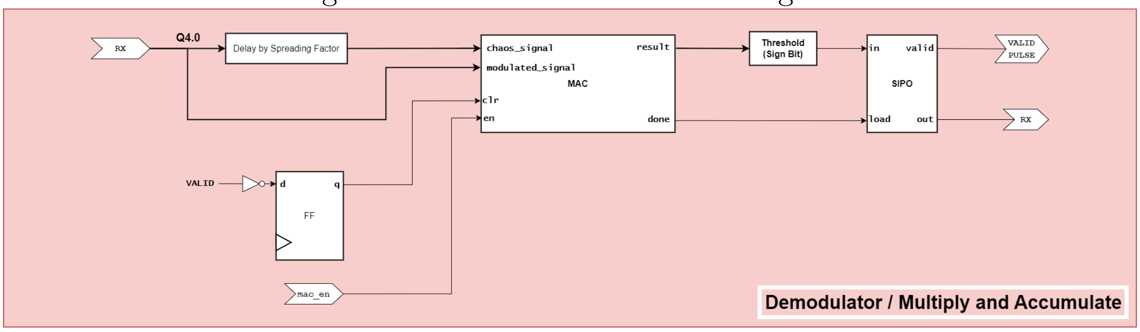


Figure 2.2: Demodulator Block Diagram



2.1.3 Serializer

This module is the implementation of Equation 1.1. Based on the index of the current chip being sent, It choses to send the chaos bit, or the modulated bit.

Table 2.2: Serializer Pin-Out

Name	Width	Direction	Description
`i_clk`	1	Input	Positive edge clock.
`i_arst_n`	1	Input	Active-low asynchronous reset.
`i_chaos_bit`	1	Input	Chaos bit
`i_modulated_bit`	1	Input	Modulated bit
`i_send`	1	Input	A pulse that signals the Chip Counter to start counting.
`o_tx`	1	Output	Serially output bit.

2.2 Receiver

At the heart of the receiver lies the MAC block, which by design, is a correlator. We don't care about the whole result of the accumulation, but only its sign. The sign bit is the demodulated information bit. A SIPO buffer is used to store and output all 32 serially-demodulated bits. w nktb h2a 7war enena bnst3ml 4 bits 34an 7war el nosie wel kalam el raye2 d2 (ya rb yb2a s7 bs)

Table 2.3: Demodulator Pin-Out

Name	Width	Direction	Description
`i_clk`	1	Input	Positive edge clock.
`i_arst_n`	1	Input	Active-low asynchronous reset.
`i_rx`	Q4.0	Input	Recevied signal.
`i_recv`	1	Input	Start Receiving.
`o_valid`	1	Output	Asserted for one cycle after all message bits have been demodulated.
`o_msg`	32	Output	The demodulated message.