



**DESIGN AND IMPLEMENTATION OF A HIGH DATA RATE QPSK DEMODULATOR
FOR NANOSATELLITES**

by

Joel S. BIYOGHE

Thesis submitted in fulfilment of the requirements for the degree

Master of Engineering: Electrical Engineering

in the Faculty of Engineering

at the Cape Peninsula University of Technology

Supervisor: Prof. Robert R. Van ZYL

Bellville Campus

December 2017

CPUT Copyright Information

The thesis may not be published either in part (in scholarly, scientific or technical journals), or as a whole (as a monograph), unless permission has been obtained from the university.

DECLARATION

I, Joel S. BIYOGHE, declare that the contents of this thesis represent my own unaided work, and that the thesis has not previously been submitted for academic examination towards any qualification. Furthermore, it represents my own opinions and not necessarily those of the Cape Peninsula University of Technology.

Signed

Date

ABSTRACT

This dissertation presents the development of a quadrature phase shift keying (QPSK) demodulator for nanosatellites that complies with both the limited resources associated with nanosatellites as well as the flexibility and configurability required for a software defined radio (SDR) platform. This research project is a component of a bigger project, which is to develop a high-speed receiver for nanosatellites, and aims to provide a practical solution to the need for communication technologies that support emerging nanosatellite applications, such as Earth observation and communications.

The development of the QPSK demodulator follows an all-digital implementation approach. The main reason for selecting this approach is to have a system that is flexible and reconfigurable to comply with the SDR requirements. Another reason for selecting this approach is to comply with the low noise system, low power consumption as well as the small size and weight requirements associated with nanosatellites. The QPSK demodulator is implemented on an *IGLOO2* Field Programmable Gate Array (FPGA), due to its robustness to radiation and high-speed capability.

Initially, the techniques used to design each subsystem of the QPSK demodulator are selected. Then, algorithms to digitally implement the designed subsystems are produced. Thereafter, the code for the digital QPSK demodulator is written and verified in Matlab first. The simulation of the Matlab-based QPSK demodulator performs satisfactorily. Subsequently, the code to implement the QPSK demodulator on an FPGA (*IGLOO2*) has been written in Libero, using VHSIC Hardware Description Language (VHDL). The resulting FPGA-based QPSK demodulator has been emulated in Libero (an integration and development environment (IDE) for Microsemi FPGAs) using a *test-bench* as well as other *analysis tools*. The test-bench results are visualized using Modelsim.

The results show that the demodulator can support data rates up to 13.25 Mbps if 16 samples-per-symbols are used, and up to 26.5 Mbps if 8 samples-per-symbols are used. It also has a very good bit-error-rate performance, which is simulated to be within a factor of 5 of the theoretical limit of QPSK modulation. Finally, the demodulator consumes less than 15 mW at the maximum operating speed. and has been coded to mitigate the effects of space radiation and noise contribution by the demodulator itself.

ACKNOWLEDGEMENTS

I would like to thank God, the beginning of all things.

I would like to thank all those who have contributed in any manner possible to me completing this work.

I would like to thank Prof. Robert van Zyl for his continuous support and motivation at various levels necessary into the completion of this work.

I would like to thank Mr Samuel Booyesen (Sampie) for his continued assistance in this project, and for his technical support. Thank you for your time, dedication and patience in assisting me.

I would like to thank Mr Charl Jooste. Your assistance, brother, at the inception of this project did not pass unnoticed.

I would like to thank Dr. Ifriky Tadadjeu Sokeng for his advice, motivation and assistance when necessary.

To the F'SATI Community, thank you for your assistance when and where needed.

A big thank you to my brothers and sisters who have never given up on me during the difficult times I had gone through towards completing this work: Jackson, Leon, Odette, Akim, Massima, Yannick, Gael, Craig, Lilie, and others.

To my families, thank you!

I would like to thank God, the end of all things.

DEDICATION

To all those who believed in me!

TABLE OF CONTENT

DECLARATION	II
ABSTRACT	III
ACKNOWLEDGEMENTS	IV
DEDICATION.....	V
TABLE OF CONTENT	VI
LIST OF FIGURES	XIV
LIST OF TABLES	XVIII
ABREVIATIONS	XIX
Chapter 1 : Introduction	1
1.1. Introduction.....	1
1.2. Background	2
1.2.1. Classification of satellites.....	2
1.2.2. Overview of CubeSats	2
1.2.2.1. Form factor	2
1.2.2.2. Evolution of CubeSat missions	3
1.2.3. Overview of the F'SATI/CPUT nanosatellite program	4
1.3. Problem statement and proposed solution.....	5
1.3.1. Research problem	5
1.3.2. Proposed solution to the problem	6
1.4. Research objectives	6
1.5. Performance specification.....	7
1.6. Research questions	7
1.7. Research delineation	7
1.8. Significance of the research.....	8
1.9. Research methodology	8
1.10. Structure of the thesis	9
Chapter 2 : Overview of digital modulation schemes with focus on the qpsk technique.....	10
2.1. Introduction.....	10

2.2.	Classification of digital modulation schemes	11
2.3.	Performance parameters of digital modulation schemes.....	12
2.3.1.	Spectral efficiency	12
2.3.1.1.	Data and symbol rate	13
2.3.1.2.	Theoretical signal bandwidth	13
2.3.1.3.	Pulse shaping filtering	14
2.3.1.4.	Practical signal bandwidth	15
2.3.1.5.	Bandwidth efficiency.....	16
2.3.2.	RF power efficiency (or BER performance).....	18
2.3.2.1.	Signal power and noise power.....	18
2.3.2.2.	Signal-to-noise ratio.....	18
2.3.2.3.	Symbol period	19
2.3.2.4.	Symbol energy (E_s) and bit energy (E_b):	19
2.3.2.5.	Noise power spectral density (N_0).....	19
2.3.2.6.	E_s/N_0 and E_b/N_0 ratios.....	19
2.3.2.7.	Symbol error rate (SER).....	20
2.3.2.8.	Bit error rate (BER).....	21
2.3.2.9.	Summary.....	21
2.3.3.	System complexity.....	22
2.3.4.	Communication medium	22
2.3.5.	Summary	22
2.4.	Comparison of digital modulations schemes.....	22
2.4.1.	Application of interest	23
2.4.2.	General remarks on non-coherent detection and amplitude shift keying modulation schemes	23
2.4.3.	Comparison of coherent PSKs and FSKs	24
2.4.3.1.	Spectral efficiency	24

2.4.3.2. Bit-error-rate	25
2.4.4. Choice of the QPSK modulation scheme	25
2.5. QPSK scheme	27
2.5.1. QPSK modulation	27
2.5.1.1. Process	27
2.5.1.2. Evaluation parameters.....	29
2.5.2. QPSK demodulation	29
2.5.2.1. Process	29
2.5.2.2. Evaluation parameters.....	33
2.6. Chapter summary	33
Chapter 3 QPSK Modulation.....	34
3.1. Introduction.....	34
3.2. Transmitted QPSK signal model	34
3.2.1. Unipolar to bipolar non-return-to-zero process.....	34
3.2.2. Demultiplexing process with Gray-code mapping	35
3.2.3. Pulse-shaping filtering	36
3.2.3.1. Importance of pulse shaping filtering	36
3.2.3.2. Selection of the root-raised-cosine filter.....	37
3.2.3.3. Design of the root-raised cosine filter.....	38
3.2.3.4. Resulting baseband signals.....	41
3.2.4. Passband carrier signals	41
3.2.4.1. Defining the modulation factor	41
3.2.4.2. Generation of the passband carrier signals	42
3.2.5. Transmitted QPSK signal.....	43
3.3. The received QPSK signal.....	44
3.4. Chapter Summary.....	46
Chapter 4 : QPSK Demodulator Design	48

4.1.	Introduction.....	48
4.2.	Implementation approach	49
4.2.1.	Selection parameters.....	49
4.2.1.1.	Size, weight and power	49
4.2.1.2.	System quality (noise figure)	50
4.2.1.3.	System operational speed	50
4.2.1.4.	Flexibility and re-configurability.....	50
4.2.2.	Generic comparison of the three implementation approaches	50
4.3.	Design theory	53
4.3.1.	Analog-to-digital converter	53
4.3.1.1.	Sampling frequency.....	53
4.3.1.2.	Sensitivity	54
4.3.1.3.	Signal-to-noise ratio.....	54
4.3.1.4.	Range of input signal amplitudes.....	54
4.3.2.	Automatic Gain Control.....	55
4.3.3.	Carrier recovery system.....	56
4.3.3.1.	Purpose.....	56
4.3.3.2.	Design techniques	57
1)	Closed loop versus open loop techniques	57
2)	m^{th} power law closed-loop technique.....	59
3)	Phase error computation closed-loop technique.....	60
4)	Selection of the single loop phase error computation carrier recovery technique	63
4.3.3.3.	Design process of a PLL-based digital carrier recovery system.....	65
1)	Design parameters.....	66
2)	Define the desired system characteristics	66
3)	Express the frequency response of the system	67
4)	Correlator block design	68

5) PED design and definition of its transfer function $P(z)$	68
6) Loop filter design and definition of its transfer function $F(z)$	71
7) NCO design and definition of its transfer function $N(z)$	72
8) Deriving the system transfer function	73
9) Determine the control loop coefficients k_p , k_o , k_1 and k_2	74
4.3.3.4. Measured parameters from the CRS	76
4.3.4. Correlator block	76
4.3.4.1. Design techniques	77
Selection of matched-filter technique.....	78
4.3.4.2. Design procedure	78
1) List of the design parameters	79
2) Design of the mixer	79
3) Design of the matched-filter	79
4.3.4.3. Resulting correlator block	80
4.3.4.4. Measured parameters of the correlator.....	80
4.3.5. Timing recovery system.....	81
4.3.5.1. Role of the TRS.....	81
4.3.5.2. Design techniques	81
1) Standard TRS configuration	81
2) Selection of the asynchronous passband sampling configuration.....	83
4.3.5.3. Design process of the TRS.....	84
1) Design parameters.....	84
2) System characteristics and frequency response	85
3) TED design and definition of its transfer function $T(z)$	85
4) Loop filter design and definition of its transfer function $F(z)$	90
5) NCO design and definition of its transfer function $N(z)$	90
6) Interpolator design	90

7)	Resulting timing recovery system and its transfer function	90
8)	Determine the coefficients k_p , k_o , k_1 and k_2	91
4.3.5.4.	Measured parameters from the TRS	92
4.3.6.	Detection and decision block	92
4.3.7.	Global performance evaluation	93
4.4.	Technical design.....	94
4.4.1.	Global design specifications and assumptions for the QPSK demodulator	95
4.4.2.	ADC selection.....	95
4.4.2.1.	Defining the desired number of samples-per-cycle	95
4.4.2.2.	Defining the desired sampling frequency	96
4.4.2.3.	Input signal amplitudes consideration	96
4.4.2.4.	Selected ADC.....	96
4.4.3.	Automatic gain control	97
4.4.4.	Correlator block	97
4.4.5.	Carrier recovery system.....	98
4.4.6.	Timing recovery system.....	99
4.4.7.	Detection and decision block	101
4.5.	Algorithms for digital implementation of subsystems.....	101
4.5.1.	Automatic gain control	101
4.5.2.	Correlator blocks	102
4.5.3.	Carrier recovery system.....	103
4.5.4.	Timing recovery system.....	106
4.5.5.	Detection and decision block	106
4.6.	Chapter summary	107
Chapter 5 :	MATLAB Implementation of the QPSK demodulator DESIGN	108
5.1.	Introduction.....	108
5.2.	Sampled QPSK signal	108

5.3.	Correlator block simulation	109
5.4.	Carrier recover system simulation.....	110
5.5.	Timing recovery system simulation	112
5.6.	Detection and decision block technical design and simulation	113
5.7.	Evaluation of the simulated QPSK demodulator performance.....	114
5.7.1.	Quality of phase constellation for various <i>SNR</i> of input signal	114
5.7.2.	Rotation effect of the carrier recovery process on the phase constellation.....	115
5.7.3.	Grouping effect of the timing acquisition process on the phase constellation	115
5.7.4.	BER versus <i>SNR</i> performance	116
5.8.	Chapter Summary.....	116
Chapter 6 : FPGA implementation of the QPSK demodulator		117
6.1.	Introduction.....	117
6.2.	Hardware selection	117
6.2.1.	Comparitive study of programmable devices	117
6.2.2.	Selection of FPGA device	118
6.3.	Development steps and evaluation tools of FPGA-based systems	119
6.3.1.	Integration and development environment	119
6.3.2.	Parameters analysis process.....	120
6.3.3.	Functional testing tools	121
6.4.	Generation of FPGA code for the demodulator	121
6.4.1.	Good VHDL coding practices.....	121
6.4.2.	Code development	122
6.5.	Emulation results of the developed FPGA-based QPSK demodulator	124
6.5.1.	Parameters analysis	124
6.5.2.	Functional test	124
6.6.	Interpretation and evaluation of results	129
6.6.1.	Resource utilisation	129

6.6.2. Operational results.....	130
6.7. Chapter summary	131
Chapter 7 : Conclusion and Recommendations	133
7.1. Introduction.....	133
7.2. Meeting the research objectives	133
7.2.1. Performance of the developed QPSK demodulator system	134
7.2.1.1. Maximum supported data rate	134
7.2.1.2. Operating frequency spectrum band.....	134
7.2.1.3. BER performance of the QPSK demodulator.....	134
7.2.1.4. Adherence to PC104 standard.....	134
7.2.1.5. Power consumption	135
7.2.1.6. Flexibility and reconfigurability	135
7.2.1.7. Mitigation of the effects of space radiation at design and development level ...	135
7.2.2. Understanding the performance parameters of digital modulation schemes	135
7.2.3. Investigation of the most adequate implementation approach.....	136
7.2.4. Understanding the various techniques used for the design of subsystems	136
7.2.5. Contribution and novelty of the research.....	136
7.3. Recommendations.....	137
REFERENCES	137
APPENDICES.....	147
Appendix-A: The Matlab Code of the QPSK demodulator	147
Appendix-B: The VHDL Code of the QPSK demodulator.....	152
Appendix-C: The System's Resource Analysis Reports	156
Appendix-D: The Test-Bench for emulation of the FPGA-based QPSK demodulator	158

LIST OF FIGURES

Figure 1.1: An illustration of the up- and downlink between a satellite and a ground station (adapted from Ippolito, 2008:9-27; Maral & Bousquet, 2009:163)	1
Figure 1.2: Illustration of a communication link employing digital modulation	1
Figure 1.3: Drawings of a (a) 1U and (b) 3U CubeSat (from Lan <i>et al.</i> , 2007:4; Munaka, 2009:21)	3
Figure 1.4: All CubeSats launched from 2000 to 2017 (adapted from Stewartwout, 2017).....	4
Figure 1.5: (a) Photo of ZACUBE-1 before launch and (b) 3D rendering of ZACUBE-2	5
Figure 2.1: Digital communication system (adapted from Sklar, 2008:55; Booyesen, 2009:3).....	10
Figure 2.2: A basic illustration of the ASK, FSK and PSK modulation (From Ippolito, 2008:356)	11
Figure 2.3: Classification of digital modulation techniques (from Sreekanth, 2003:42)	12
Figure 2.4: Time and frequency domain representation of a square pulse (from Proakis, 1995:207; Lathi & Ding, 2010:78)	13
Figure 2.5: Time and frequency responses of a raised cosine filter for three values of the roll-off factor α (from Gentile, 2007:3;4)	15
Figure 2.6: Power spectral density of MSK, BPSK, and QPSK (From Ke-Lin & Swamy, 2010:214; Torlak, 2013:3).....	24
Figure 2.7: (a) Probability of symbol error versus E_b/N_0 for coherently detected MPSK signals; (b) Probability of bit error versus E_b/N_0 for coherently detected MFSK signals (From Proakis, 1995:263; 272).....	25
Figure 2.8: Block diagram of a digital pass band communication system employing the QPSK modulation scheme (adapted from Haykin, 2000:359).....	27
Figure 2.9: An illustration of the QPSK modulation process (adapted from Haykin, 2000:359)..	27
Figure 2.10: Constellation diagram of a QPSK signal (adapted from Zafar & Farooq, 2008:6; Hasan, <i>et al.</i> , 2010:23)	29
Figure 2.11: Illustration of the QPSK demodulation process (adapted from Popescu <i>et al.</i> , 2011:366; Haykin, 2014:361).....	30
Figure 3.1: (a) Digital communication system based on the QPSK modulation scheme (adapted from Sklar, 2008:167); (b) Standard block diagram of the QPSK modulator	34
Figure 3.2: Illustration of the unipolar-to-bipolar bit stream conversion.....	35
Figure 3.3: A portion of the generated i - and q - channels bit streams	36

Figure 3.4: Time and frequency domain representation of a square pulse (from Proakis, 1995:207; Lathi & Ding, 2010:78)	36
Figure 3.5: Illustration of matched filtering using a rectangular pulse (adapted from Ashok, 2007:116; Lathi & Ding, 2010:365)	37
Figure 3.6: Illustration of a bit stream from a pulse shaping RCF (from Glover & Grant, 1998:364)	38
Figure 3.7: The designed root-raised-cosine filter (RRCF)	40
Figure 3.8: Baseband signals corresponding to the i - and q - bits streams in Figure 3.3	41
Figure 3.9: Time domain (a) and frequency domain (b) representation of the generated passband carrier signals	42
Figure 3.10: The transmitted QPSK signal	43
Figure 3.11: The time and frequency representation of the received QPSK signal	46
Figure 4.1: (a) Digital communication system based on the QPSK modulation scheme (adapted from Sklar, 2008:167); (b) Standard block diagram of the QPSK demodulator (adapted Glover & Grant, 1998:398)	49
Figure 4.2: Block diagram of the AGC	55
Figure 4.3: Ideal transfer function of the AGC (from Rosu, 2017:2)	56
Figure 4.4: Illustration of the feed-forward carrier recovery process (adapted from Chen, 2004:41-42)	57
Figure 4.5: Basic illustration of a phase-locked loop, (a) continuous time domain (adapted from Nicoloso, 1997:14) and (b) discrete time domain (adapted from Booysen, 2009:23)	58
Figure 4.6: Illustration of the m^{th} power loop carrier recovery technique	59
Figure 4.7: Carrier phase synchronisation configurations for MPSKs and MQAMs.	60
Figure 4.8: The arc-tan PEDs (DD and DA)	61
Figure 4.9: (a) Simplified data-aided PED, (b) digital Costas loop PED	62
Figure 4.10: Discrete time PLL-based carrier recovery system for QPSK, using a <i>single-loop phase error computation technique</i> (adapted from Peebles, 1987:261; Dick <i>et al.</i> , 2004:59)	65
Figure 4.11: Linearised phase model of the PLL-based CRS under design	65
Figure 4.12: Digital Costas' loop phase error detection configuration	69
Figure 4.13: PED S-curve	70
Figure 4.14: Graph of PED gain k_p versus phase error φ	71
Figure 4.15: Linearised phase model of the PED block	71
Figure 4.16: Illustration of the first order loop filter	72

Figure 4.17: A basic block diagram of a numerically controlled oscillator (NCO)	73
Figure 4.18: Schematic of the digital carrier recovery system under design	74
Figure 4.19: A linearised phase model of a second-order discrete domain PLL (adapted from <i>Booyesen, 2009:23</i>)	74
Figure 4.20: Integrator correlators (adapted from Peebles, 1987:256; Lathi & Ding, 2010:545)	77
Figure 4.21: Matched-filter correlators	78
Figure 4.22: Schematic of the matched-filter correlators under design	80
Figure 4.23: Standard TRS configurations (adapted from Booyesen, 2009:6)	82
Figure 4.24: Preferred TRS implementation configuration	83
Figure 4.25: Illustration of the early-late timing error detection algorithm	86
Figure 4.26: Block diagram of the early-late timing error detector	87
Figure 4.27: Illustration of the zero-crossing timing error detection algorithm	88
Figure 4.28: Block diagram of the zero-crossing timing error detector	88
Figure 4.29: Illustration of the TED S-curve for three different received baseband signal amplitudes	89
Figure 4.30: Linearised phase model of the timing error detector	89
Figure 4.31: Technical block diagram of the digital timing recovery system under design	91
Figure 4.32: A linearised phase model of a second-order discrete time-domain PLL	91
Figure 4.33: Illustration of the detection and decision block	92
Figure 4.34: Detailed block diagram of the digital QPSK demodulator under design	94
Figure 4.35: Impulse response of the designed matched-filter	98
Figure 4.36: <i>S-curve</i> of the designed PED	99
Figure 4.37: The standard (left) and the NCO (right) LUT coefficients	99
Figure 4.38: <i>S-curve</i> of the designed early-late TED	100
Figure 4.39: Algorithm for the digital implementation of the AGC process	102
Figure 4.40: Illustration of the convolution process: (a) pipeline implementation (adapted from <i>Ahamed & Lawrence, 1997:256</i>) and (b) multi-cycle implementation (adapted from <i>Mitra, 2006:433</i>)	103
Figure 4.41: Flowchart of the local carriers generation process	105
Figure 4.42: Flow chart of (a) the interpolation control and (b) the interpolation process	106
Figure 5.1: Sampled QPSK input signal	109
Figure 5.2: The reproduced baseband bit streams	110
Figure 5.3: Step phase response of the carrier recovery system	111
Figure 5.4: Local carrier signals generated by the NCO	111

Figure 5.5: Step phase response of the developed timing recovery system	112
Figure 5.6: TRS output clock signal at different instants of the recovery process	112
Figure 5.7: Reproduced versus transmitted bit streams on both the i - and q -channel	113
Figure 5.8: Phase constellation of demodulated received signal, for different input SNRs	114
Figure 5.9: Illustration of constellation rotation as the phase-locking process progresses	115
Figure 5.10: Illustration of the constellation stabilisation during the timing acquisition process	115
Figure 5.11: Simulated and theoretical BER versus SNR plot for the QPSK demodulator	116
Figure 6.1: Design flow in <i>Libero</i> for the development of FPGA-based systems	120
Figure 6.2: Block diagram of the FPGA firmware of the QPSK demodulator	123
Figure 6.3: Test-bench input QPSK signal	125
Figure 6.4: Baseband signals reproduced by the correlator blocks	126
Figure 6.5: Simulated carrier recovery phase-step response	126
Figure 6.6: Generated local carrier signals from the carrier recovery system	126
Figure 6.7: Simulated timing recovery phase-step response	127
Figure 6.8: Clock signal generated by the timing recovery system for bit detection by the DDB	127
Figure 6.9: Reproduced bit stream i -(top) and q -(bottom) channels	127
Figure 6.10: Phase constellation diagrams with (a) alternating and (b) random bit streams with an input SNR of 12 dB	128
Figure 6.11: Simulated BER versus SNR plot	129

LIST OF TABLES

Table 1.1: Classification of satellites according to their mass (adapted from Stark <i>et al.</i> , 2003:582; Royi, 2011:2)	2
Table 1.2: CubeSat dimensions, mass and estimated available power (from Calcutt & Tetley, 1994:185; Carnahan, 2013:3; Singarayar <i>et al.</i> , 2013:7).....	3
Table 2.1: Bandwidth efficiency of various coherent and non-coherent MFSK modulation schemes.....	17
Table 2.2: Bandwidth efficiency of coherent MPSK modulation schemes.....	17
Table 2.3: Digital modulation schemes and their applications	26
Table 4.1: Advantages and disadvantages of the three implementation approaches	51
Table 4.2: Summary of the algorithms to digitally implement each block of the CRS.....	103
Table 6.1: Comparison of programmable hardware	117
Table 6.2: Resource usage analysis of the QPSK demodulator implementation on the FPGA	124

ABREVIATIONS

ADC	Analog-to-Digital Converter
AGC	Automatic Gain Control
ASK	Amplitude Shift Keying
AWGN	Additive White Gaussian Noise
BER	Bit-Error-Rate
BPSK	Binary Phase Shift Keying
CRS	Carrier Recovery System
dB	Decibel
DDB	Detection and Decision Block
DDS	Direct Discrete Synthesiser
DSB	Double-Side Bandwidth
E_b/N_o	Energy per bit/Noise spectral density
EL	Early-Late
F'SATI	French South African Institute of Technology
FE	Front-End
FIR	Finite Impulse Response
FPGA	Field Programmable Gate Arrays
FSK	Frequency Shift Keying
GMSK	Gaussian Minimum Shift Keying
GS	Ground Station
HF	High Frequencies
HPD	Hardware Programmable Device
Hz	Hertz
IC	Interpolation Controller
IF	Intermediate Frequency
kbps	Kilobits Per Second
LEO	Low Earth Orbit
LF	Loop Filter
LUT	Look-Up-Table
MASK	M-ary Amplitude Shift Keying
Mbps	Mega-Bits Per Second
MF	Matched Filter

MFSK	M-ary Frequency Shift Keying
MPSK	M-ary Phase Shift Keying
MQAM	M-ary-Quadrature Amplitude Modulations
MSK	Minimum Shift Keying
MspS	Mega-Symbols Per Second
MSps	Mega-Samples Per Second
NCO	Numerically Controlled Oscillator
NRZ	Non-Return-to-Zero
P2S	Parallel-to-Serial
PED	Phase Error Detector
PLL	Phase-Locked Loop
PSF	Pulse Shaping Filter
PSK	Phase Shift Keying
QAM	Quadrature Amplitude Modulation
QPSK	Quadrature Phase Shift Keying
RCF	Raised Cosine Filter
RF	Radio Frequency
RRCF	Route Raised Cosine Filter
Rx	Receiver
S2P	Serial-to-Parallel
SDR	Software Defined Radio
SER	Symbol-Error-Rate
SF	Scale-down Factor
SNR	Signal-to-Noise Ratio in dB
SSB	Single-Side Bandwidth
TED	Timing Error Detector
TRS	Timing Recovery System
Tx	Transmitter
UHF	Ultra High Frequencies
VCO	Voltage Controlled Oscillator
VHDL	VHSIC Hardware Description Language
VHF	Very High Frequencies
ZC	Zero-Crossing

CHAPTER 1: INTRODUCTION

1.1. Introduction

The research project reported in this thesis is the *design and implementation of a high data rate QPSK demodulator for nanosatellites*. This project forms part of a broader project, which is the development of a high-speed transceiver for nanosatellites (CubeSats), capable of supporting fast down- and uplink communication, while complying with the limitation of resources associated with nanosatellites. Figure 1.1 below illustrates the communication up- and downlinks between the satellite and the ground station (Maral & Bousquet, 1998:3).

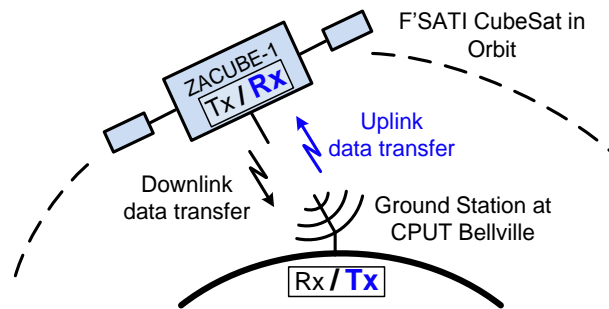


Figure 1.1: An illustration of the up- and downlink between a satellite and a ground station (adapted from Ippolito, 2008:9-27; Maral & Bousquet, 2009:163)

The focus of this research will be on the uplink and, more specifically, on the digital modulator and demodulator highlighted in Figure 1.2 below, which gives a more technical representation of an Earth-to-space communication link.

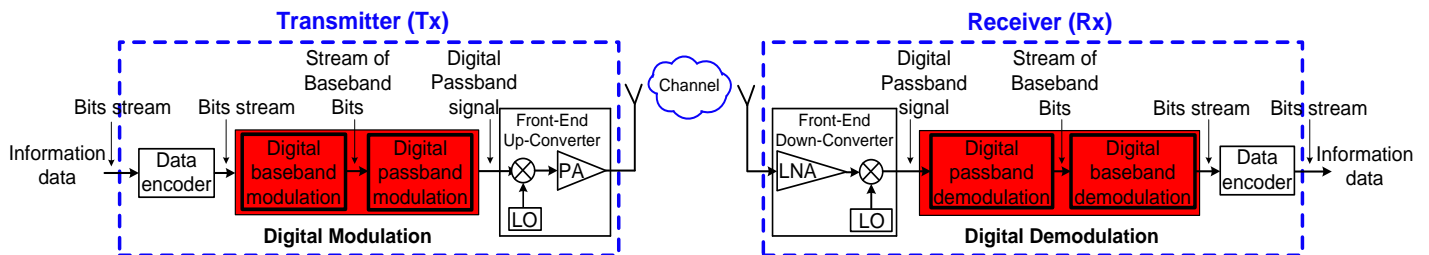


Figure 1.2: Illustration of a communication link employing digital modulation

This chapter gives the background to the research and states the research problem. Thereafter, it presents the research objectives, specifications, questions, delineation and significance.

1.2. Background

1.2.1. Classification of satellites

The journey of the satellite industry started in 1957 when the Soviets put the first man-made satellite into space. The satellite, named Spuknik-1, weighed 84 kg and its applications were limited to measuring the density and temperature of the upper atmosphere (Pratt *et al.*, 2003:3). Since then, the rapid growth of the satellite industry has led to critical technology advancement in the space sector. Satellites have since been used for various applications, including telecommunication, remote sensing, in-situ monitoring and science experiments (Maini & Agrawal, 2007:4; Mitra, 2008:7). For a long time, the high cost of satellite development prevented many entities from getting involved in the space industry, due to constrained budgets. In order to palliate the problem, extremely small satellites with limited capabilities started to be developed at significantly lower cost; they are classified as nanosatellites. Table 1.1 below lists existing categories of satellites.

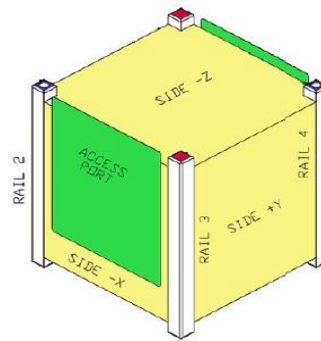
**Table 1.1: Classification of satellites according to their mass
(adapted from Stark *et al.*, 2003:582; Royi, 2011:2)**

Satellite categories	Mass (kg)	Estimated orbit life span (years)	Estimated Cost (£Millions)
Large	> 1000	>10	>100
Small	500 – 1000	8 – 10	25-100
Mini	100 – 500	5 – 8	7-25
Micro	10 – 100	3 – 5	1-7
Nano	1 – 10	1 – 3	0.1-1
Pico	< 1	1 – 2	<0.1

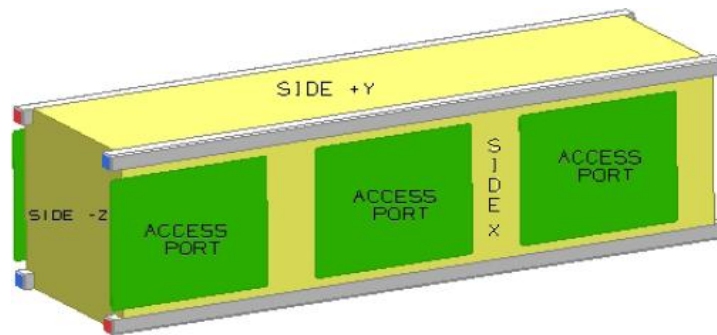
1.2.2. Overview of CubeSats

1.2.2.1. Form factor

A CubeSat is a standard type of nanosatellite, which consists of one or more cubic units. A cubic unit, also known as a 1U, has a base of 10 cm x 10 cm and a height of 10 cm (Singarayar *et al.*, 2013:7). Many CubeSat form factors, such as two units (2U), three units (3U), six (6U)...etc., can be generated by combining cubic units (McNutt *et al.*, 2009:2; 4). The CubeSat standard was developed in 1999 by Prof. Jordi Puig-Suari from the California Polytechnic State University (Cal Poly) and Prof. Bob Twiggs from Stanford University, which means that CubeSats have only been around for just over two decades (Lenz *et al.*, 2002; Manukata, 2009:5). Figure 1.3 gives 3-dimensional illustrations of a (a) 1U and (b) 3U CubeSat..



(a)



(b)

Figure 1.3: Drawings of a (a) 1U and (b) 3U CubeSat (from Lan *et al.*, 2007:4; Munaka, 2009:21)

Each unit has a maximum mass of 1.33 kg (Carnahan, 2013:3). Due to very small *aperture area*, solar panels only capture a limited amount of solar energy, which limits the available power to a few watts (Calcutt & Tetley, 1994:185). Table 1.2 summarises the characteristics of three basic units of CubeSats.

Table 1.2: CubeSat dimensions, mass and estimated available power (from Calcutt & Tetley, 1994:185; Carnahan, 2013:3; Singarayar *et al.*, 2013:7)

CubeSat form factor	Base [cm]	Standard height [cm]	Max height [cm]	Standard weight [kg]	Maximum weight [kg]	Estimated Power available (per orbit cycle) [W]
One unit (1U)	10x10	10	11.3	1	1.33	3
Two unit (2U)	10x10	20	22.7	2	2.70	5
Three unit (3U)	10x10	30	34.0	3	3.90	8

1.2.2.2. Evolution of CubeSat missions

The main aim of CubeSats was originally to enable academic institutions to get participate in space programmes with relatively low budgets (Selva & Krejci, 2011:52). The idea was quickly

adopted by many universities across the globe, and also gained interest from bigger satellite developers. By 2009, more than 100 groups around the world were involved in CubeSat development programmes, including the French-South African Institute of Technology (F'SATI) at the Cape Peninsula University of Technology (CPUT) (Manukata, 2009:5). To date (2017), more than 700 CubeSats have been developed and launched. Figure 1.4 displays the number of CubeSats launched per year since 2000 up to 2017.

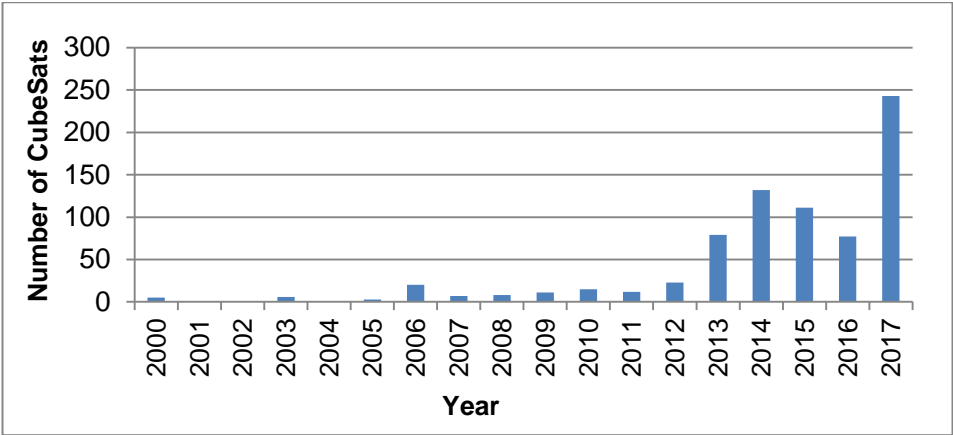


Figure 1.4: All CubeSats launched from 2000 to 2017 (adapted from Stewartwout, 2017)

Initially, CubeSats were developed for basic applications, such as *low quality Earth imaging, carrying small science experiments, testing basic communication capabilities, or testing the space reliability of newly designed components*. These applications could be achieved with the basic technology available at the time.

However, CubeSat developers now want to provide with their nanosatellites, applications such as *high-quality Earth observation (Selva & Krejci, 2011:52), in-situ monitoring (Singarayar et al., 2013), instant point-to-point communication, reprogrammable computing and space weather monitoring*. These applications drive the requirement for new and advanced nanosatellite technologies, such as in communications, computing, control, power and imaging, *and would not be supported by the low-speed communication techniques originally used (Hart, 2000:2)*.

1.2.3. Overview of the F'SATI/CPUT nanosatellite program

The French-South African Institute of Technology (F'SATI) at the Cape Peninsula University of Technology (CPUT) introduced a satellite systems engineering programme in 2008, which involves a CubeSat development programme. The global objective of the programme is to develop human capacity, to grow and sustain satellite and space engineering in South Africa

first, and in Africa at large (van Zyl, 2011). The programme offers a platform to graduate students to enhance their knowledge of space science, and to work on space technology projects. The programme started with the development of a 1U CubeSat. The design and development of the various subsystems of these CubeSats constitute many postgraduate research projects at CPUT.

On 21 November 2013, F'SATI/CPUT launched their first CubeSat, a 1U CubeSat named ZACUBE-1 and shown in Figure 1.5. Currently, F'SATI is developing a more advanced 3U CubeSat named ZACUBE-2. Figure 1.5 below shows the two satellites.

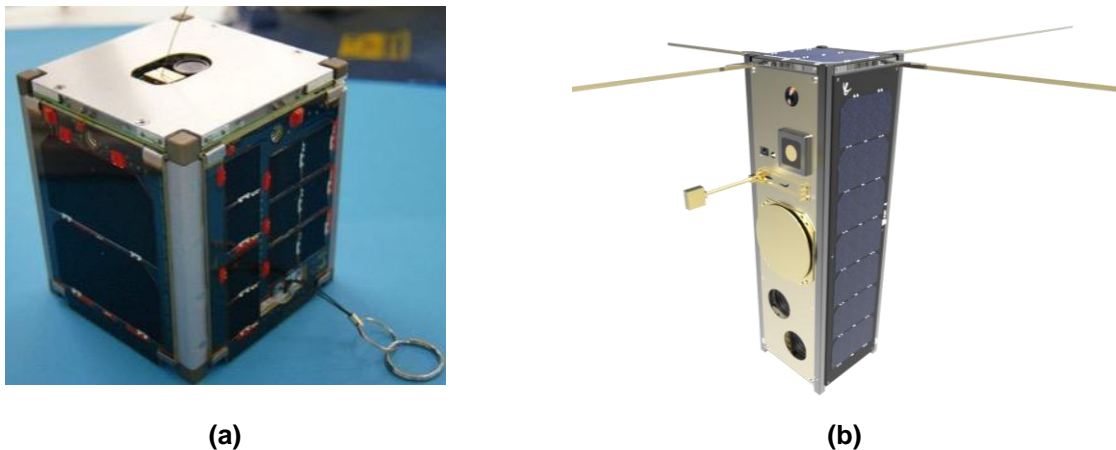


Figure 1.5: (a) Photo of ZACUBE-1 before launch and (b) 3D rendering of ZACUBE-2

1.3. Problem statement and proposed solution

1.3.1. Research problem

As mentioned earlier, CubeSats had initially been used for basic applications that generally did not require high-speed communications, for both up- and downlinks. Consequently, low data rate communication modulation techniques, such as AFSK, GMSK and BPSK, and supporting data rates lower than 0.5 Mbps, were suitable.

Therefore, there is a pressing need to investigate and implement communication techniques that enable higher communication speeds, while complying with the limited resources available onboard CubeSats.

1.3.2. Proposed solution to the problem

As an attempt to meeting the existing requirement for high-speed communication links, F'SATI has initiated research on this matter. The quadrature phase shift keying (QPSK) modulation scheme was identified as one of the most suitable techniques for the desired high-speed communication links for nanosatellites. The support of this choice is presented in Chapter 2.

Consequently, F'SATI has recently developed a high data rate transmitter for nanosatellite downlinks, using the *QPSK modulation scheme*. This S-band transmitter can achieve data rates of up to 10 Mbps (Jooste *et al.*, 2014:2). It is being flown on various missions from different groups across the world, such as UKube-1, the United Kingdom's first CubeSat currently in orbit.

This radio clearly supports the current requirements for high-speed downlinks. However, since high-speed uplink communication is also required, F'SATI is currently aiming to develop a high data rate receiver for nanosatellites uplinks, based on the QPSK modulation scheme to complement the existing S-band transmitter. *The high data rate QPSK demodulator developed through this project and presented in this thesis is a major component of the high data rate receiver.*

1.4. Research objectives

The objectives of this research project are to:

- design and implement a digital demodulator based on the QPSK modulation scheme, which can support high data rate communications as specified in Section 1.5;
- ensure that the developed QPSK demodulator complies with the limited resources associated with nanosatellites;
- understand the technical difference between digital modulation schemes and be able to support the selection of one for a given application;
- investigate which implementation approach (an all-analog, all-digital or hybrid of both) would be the most adequate to implement the desired QPSK demodulator;
- identify, for each subsystem of the QPSK demodulator, existing techniques used to implement the subsystem, and then select the one that would be most suitable to accomplish the expected task while complying with the constraints set;
- investigate possible means of mitigating or minimising the space radiation effects on the resulted system at design and development level; and to

- attempt to be creative through the design and development of the various subsystems in order to be able to either come up with a novel implementation technique of a subsystem or produce a novel overall system output.

1.5. Performance specification

The performance specification of the QPSK demodulator is summarised as follows:

- support data rates up to 10 Mbps;
- operate within available amateur VHF-, UHF-, L-, S-, C-, or X-bands;
- comply with the *PC104* standard; have DC power consumption less than 100 mW;
- be flexible and re-configurable for the purpose of software-defined-radio (SDR);
- present some level of robustness against space radiation; and the
- implementation must be of good quality in the sense that the simulated BER should be within a factor of 10 of the theoretical limit ($BER_{\text{simulated}} \leq 10 \times BER_{\text{theoretical}}$).

1.6. Research questions

The following questions will be guidelines for this research:

1. What are the constraints involved in the design of a communication system for nanosatellites?
2. What are the criteria involved in comparing various digital modulation techniques for nanosatellite communication links?
3. What is the relation between the spectrum-band used and the maximum achievable data rate for a QPSK signal?
4. What are the performance parameters of a digital demodulator?
5. What are the criteria for comparing implementation methods of a specific modulation scheme to be used for nanosatellite communication?
6. How can space radiation effects on an electronic system be mitigated at system design and development levels?

1.7. Research delineation

The following delimitations have been set for this project:

- only one approach (component-based, software-based or hybrid) will be implemented for the practical product; the comparison will be done in terms of a literature study;

- given that the all-digital implementation has been performed, and due to difficulties encountered in producing suitable hardware for this application, the QPSK demodulator has been validated through emulation in an integration and development environment (IDE) of FPGAs (in this case, Libero for the case of Microsemi FPGAs), which according to experts, is accepted as a reliable validation tool (Blanchard, 2017; Booysen, 2017);
- the research here will be limited to the reproduction of the transmitted data-stream, using the incoming IF pass-band signals, and no down-conversion or data decoding, which include error correction, will be implemented.

1.8. Significance of the research

From the human capacity development perspective, the significance of this research project, in line with the vision of F'SATI and CPUT, is to have introduced and developed one more African student for the space industry in general and nanosatellites in particular; these are high technology industries. From the technical perspective, the significance of this research project is to make available to the community of nanosatellites developers, a receiver that supports high data rate communications up to 10 Mbps. This is an important contribution to the nanosatellite industry for a few reasons. The first reason is that CubeSat developers, such as Planet-Labs and Spire-Inc, who currently have advanced communication technologies, do not make their products available to the community (Planet Labs, 2017; Spire, 2017). The second reason is that of all the nanosatellites communication products available to date, no CubeSat receiver can support uplink communications beyond 5 Mbps (Clyde-Space, 2017; F'SATI, 2017; Gomspace, 2017; ISIS, 2017; Syrlinks, 2017; Tethers Unlimited, 2017). Note that, making such a high speed receiver available to the nanosatellite community, will allow nanosatellite developers to establish both high-speed uplink and high-speed inter-satellites communications, with their nanosatellites. This will subsequently, make it possible to achieve with nanosatellites, desired applications, such as *in-situ monitoring*, *instant point-to-point communication*, *space weather monitoring* and *high-quality Earth observation*; which require instant transfer of large amount of data.

1.9. Research methodology

The research methodology and plan is summarised in chronological order as follows:

- 1) The performance of various modulation schemes is investigated and the choice of the QPSK scheme motivated; QPSK modulation and demodulation processes are then investigated in further detail.

- 2) Assess and compare the merits of *all-analog*, *all-digital* and *hybrid* implementations of QPSK demodulators.
- 3) Design various subsystems of the QPSK demodulator based on both the literature review and the technical specifications set, and generate algorithms for the all-digital implementation of each subsystem.
- 4) Do a primary implementation of the designed digital subsystems in Matlab and simulate the digital QPSK demodulator.
- 5) Write VHDL code to implement the designed digital subsystems on an FPGA and test the FPGA-based QPSK demodulator.

1.10. Structure of the thesis

Chapter 1 gives an introduction and background to the research done. It presents the research problem, objectives, significance as well as the research questions.

Chapter 2 compares various digital modulation schemes and support the choice of the QPSK technique for this application.

Chapter 3 presents the implementation of the QPSK modulation, which includes the generation of the QPSK signal that will be used to test the digital QPSK demodulator under design.

Chapter 4 discusses the design of the QPSK demodulator. It compares the merits of *all-analog*, *all-digital* and *hybrid* implementations of QPSK demodulators. It describes the design techniques and design process of each subsystem of the QPSK demodulator. It presents the technical design of each subsystem and gives the algorithms developed to digitally implement these designed subsystems.

Chapter 5 presents the Matlab implementation of the designed QPSK demodulator as well as all the simulation results obtained.

Chapter 6 discusses the implementation on FPGA of the designed digital QPSK demodulator. It presents the selection of the FPGA chip, the VHDL coding, the emulation of the resulting FPGA-based QPSK demodulator as well as the analysis of the obtained results.

Chapter 7 concludes this thesis and presents possible recommendations. It establishes a link between the project objectives set and the results achieved.

CHAPTER 2: OVERVIEW OF DIGITAL MODULATION SCHEMES WITH FOCUS ON THE QPSK TECHNIQUE

2.1. Introduction

Digital passband communication includes digital modulation at the transmitter and digital demodulation at the receiver. Modulation encompasses baseband modulation followed by passband modulation; demodulation comprises passband demodulation followed by baseband demodulation. A schematic of a typical digital communications system is shown in Figure 2.1.

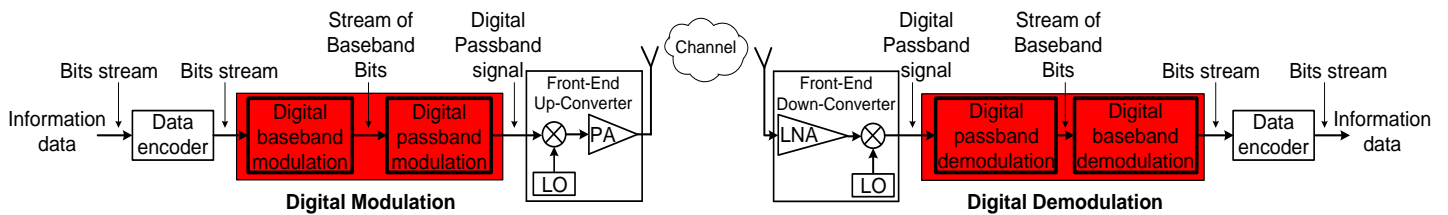


Figure 2.1: Digital communication system (adapted from Sklar, 2008:55; Booyesen, 2009:3)

An electric waveform is a sinusoidal signal generally expressed in terms of its amplitude (A), frequency (f) and phase (θ) as (Kapadia, 2012:30):

$$g(t) = A\cos(2\pi ft + \theta) \quad (2.1)$$

Digital passband modulation involves shifting (varying) the amplitude, frequency or phase of an electric wave, using a stream of digital baseband waveform (Sklar, 2008: 169; 174). The three basic modulation families include amplitude shift keying (ASK), frequency shift keying (FSK) and phase shift keying (PSK) (Memon *et al.*, nd:2). There also exist hybrids of these, such as quadrature-amplitude modulation (QAM) (Zafar & Farooq, 2008:65).

Digital modulation provides improved security and flexibility (Mishra *et al.*, 2010:1; Thede, 1996:199); however, it is generally associated with two main issues, namely, *channel bandwidth limitation* and *available power limitation* (Pasupathy, 1979:14; Vertat & Mraz, 2013:389). In some applications, the bandwidth limitation is the biggest constraint, and in others, the power limitation is the biggest constraint. In order to attempt to resolve this problem at design level, it is necessary to choose the digital modulation technique that is the most suitable for the given application.

In this chapter, the classification of digital modulation schemes is discussed, followed by the performance parameters of these modulation schemes. Thereafter, the modulation schemes are compared and the selection of the QPSK for this application motivated. Finally, the QPSK modulation scheme (modulation and demodulation processes) is presented.

2.2. Classification of digital modulation schemes

The ASK, FSK and PSK schemes mentioned above are different families of digital modulation, and are illustrated in Figure 2.2 for a random bit stream.

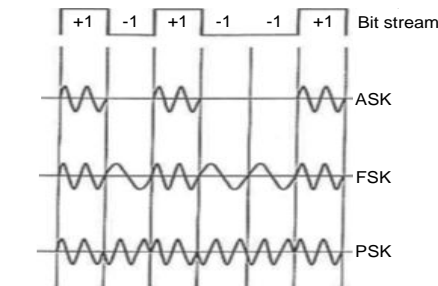


Figure 2.2: A basic illustration of the ASK, FSK and PSK modulation (From Ippolito, 2008:356)

Digital modulation schemes can be classified in two categories, namely *coherent detection* and *non-coherent detection* schemes. In coherent detection techniques, the local carrier(s) at the receiver requires the same phase and frequency as the received carrier in order to demodulate the passband signal (Mitra, 2008:81). In non-coherent detection techniques, the receiver does not need to have knowledge of the incoming carrier parameters. Instead, it uses a set of defined local carriers. Non-coherent detection techniques, therefore, do not require carrier recovery systems (CRS), resulting in less complex implementations (Pursley, 2005:418). On the other hand, coherent detection techniques present improved power efficiencies compared to incoherent detection techniques, but require a carrier recovery system, and are therefore, more complex to implement (Sklar, 2008:171).

Modulation techniques with two symbols ($M = 2$) are *binary* (B) and they include the BPSK, BASK and BFSK modulation techniques. Alternatively, when the number of symbols exceeds two ($M > 2$), the result is a series of M-ary modulations. For PSK, there are coherent MPSKs, which include QPSK, 8PSK, 16PS, and so on, as well as non-coherent MPSKs, known as M-ary differential PSKs (MDPSK) (Tao, 2013:46). Similarly, for FSK, there are coherent and non-coherent MFSKs. The minimum shift keying (MSK) and quadrature amplitude modulation (QAM)

are two examples of hybrid modulation schemes, and are all coherent detection techniques (Hranac, 2001:7-9). Figure 2.3 summarises the families of the digital modulation schemes.

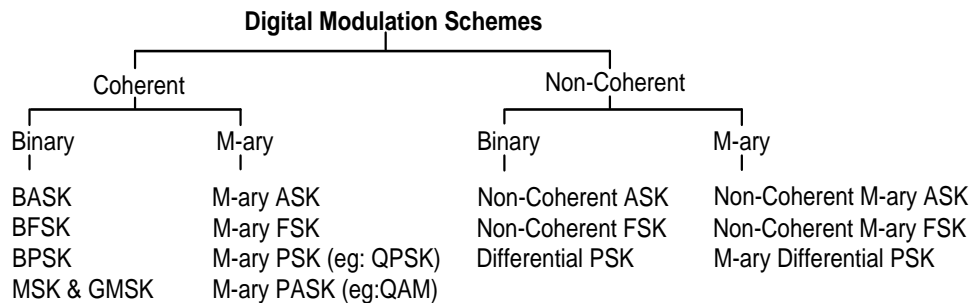


Figure 2.3: Classification of digital modulation techniques (from Sreekanth, 2003:42)

The selection of the digital modulation scheme for a given application is a very important step in the communication system design, as it helps resolve at design level the challenges of available power and bandwidth limitations encountered in the digital communication environment.

2.3. Performance parameters of digital modulation schemes

The parameters generally used to compare various modulation schemes and the selection of a particular scheme, include (Pasupathy, 1979:14; Webber & Dahnoun, 1996:10; Vertat & Mraz, 2013:389):

- spectral efficiency;
- power efficiency;
- complexity of implementation; and
- type of communication channel.

Each of these parameters are discussed in further detail in the following sections.

2.3.1. Spectral efficiency

The spectral efficiency is used to evaluate how efficiently a modulation scheme can use the available bandwidth to allow successful transmission of signals through the communication system at the desired data rate (Pasupathy, 1979:14). Essentially, it tells us how much bandwidth a given modulation scheme will use to transmit a signal at a certain data rate (Mulally & Lefevre, nd:1). In order to determine the spectral efficiency of a modulation scheme, the following parameters must be understood:

- data and symbol rate;

- theoretical signal bandwidth;
- pulse shaping; and
- practical signal bandwidth.

2.3.1.1. Data and symbol rate

The *data rate* or *bit rate* (R_b), is the rate at which bits are being transmitted through the communication system; it is expressed in bits-per-second (bps) . Consequently, the time used to transmit “one bit” is called *bit period* (T_b) or *bit length*.

A *symbol* in digital communication is the signaling element that is transmitted through the communication medium (Fitton, 2012:4). A symbol is a combination of a number of bits (n). The number of symbols for a modulation scheme is given by $M = 2^n$; that is, $n = \log_2(M)$. MPSK, MFSK, and MASK...etc. denote the PSKs, FSKs and ASKs modulation techniques where the number of possible symbols is $M = 2^n$.

The *symbol rate* describes the rate at which signaling elements are transmitted through the communication link (Forouzan, 2007:142; 143). It is often denoted (R_s), and is related to the desired data rate (R_b) by the number of bits in a symbol (n) as:

$$R_s = \frac{R_d}{\log_2(M)} \quad [\text{symbols-per-second (sps)}] \quad (2.2)$$

2.3.1.2. Theoretical signal bandwidth

The spectral usage of a signal can be described as the frequency band occupied by the *modulated signal* across a defined carrier (Forouzan, 2007:69). It is determined from the symbol rate (R_s), which is the inverse of the *symbol period* (T_s). The frequency spectrum of a rectangular pulse is presented in Figure 2.4.

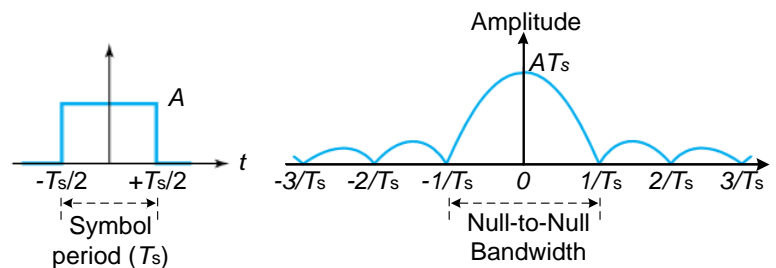


Figure 2.4: Time and frequency domain representation of a square pulse (from Proakis, 1995:207; Lathi & Ding, 2010:78)

The bulk of the signal power is confined within the main lobe; consequently, the *null-bandwidth* and the *null-to-null-bandwidth* are the theoretical bandwidths for single-side-band (SSB) and double-side-band (DSB) signals, respectively (Torlak, 2013:12):

$$\begin{aligned} B_{\text{theoretical}} &= R_s \text{ [Hz]} \quad (\text{for SSB signals}) \\ B_{\text{theoretical}} &= 2R_s \text{ [Hz]} \quad (\text{for DSB signals}) \end{aligned} \tag{2.3}$$

Two major problems often make the usage of square pulses difficult:

- the spectrum of the ideal square pulse extends to infinity with side lobes; and
- in some applications, the null-to-null-bandwidth is greater than the available bandwidth.

To resolve these problems and ensure that the signal's bandwidth is reduced to an allowable range, filtering of the square pulse is often performed and is referred to as *pulse shaping*. A *raised-cosine-filter* is the most commonly used for this purpose (Micheli, nd:28).

2.3.1.3. Pulse shaping filtering

Raised-cosine-filtering (RCF) is a *pulse shaping* technique used to reduce the bandwidth occupied by square pulses (Fitton, 2012:8). Here, bits are manipulated by a non-rectangular pulse shape defined by the impulse response $h_{\text{RC}}(t)$ of the RCF (Gentile, 2002:52).

The RCF uses Nyquist's principle, which states that when a signal is transmitted with a symbol rate R_s , it must have a minimum SSB bandwidth $B_0 = 0.5R_s$ (Hz) in order to be detected with minimum error at the receiver. Therefore, using a roll-off factor (α), the RCF defines the SSB bandwidth of the transmitted signal (B_T) between a minimum (B_0) and a maximum (R_s) (Wong & Lok, 2014:4.6). The roll-off factor (α) can take any value from 0 to 1, and sets the link between the Nyquist's bandwidth (B_0) and the desired SSB transmission bandwidth (B_T) as

$$\alpha = \frac{B_T - B_0}{B_0} = \frac{2B_T - R_s}{R_s} \tag{2.4}$$

The frequency response $H_{\text{RC}}(f)$ and the impulse response $h_{\text{RC}}(t)$ of the RCF can be expressed in terms of α and B_0 as (Sklar, 2008:139-140; Haykin, 2014:454):

$$H_{RC}(f) = \begin{cases} \frac{1}{2B_0}, & 0 < |f| < f_1 \quad \text{and } f_1 = (1-\alpha)B_0 \\ \frac{1}{4B_0} \left\{ 1 + \cos \left[\frac{\pi}{2B_0\alpha} (|f| - f_1) \right] \right\}, & f_1 < |f| < 2B_0 - f_1 \quad \text{and } f_1 = (1-\alpha)B_0 \\ 0, & |f| > 2B_0 - f_1 \quad \text{and } f_1 = (1-\alpha)B_0 \end{cases} \quad (2.5)$$

$$h_{RC}(t) = \text{sinc}(2B_0t) \frac{\cos(2\pi\alpha B_0t)}{1 - 16\alpha^2 B_0^2 t^2} \quad (2.6)$$

Both the frequency and impulse responses of the RCF are illustrated in Figure 2.5 for various values of α .

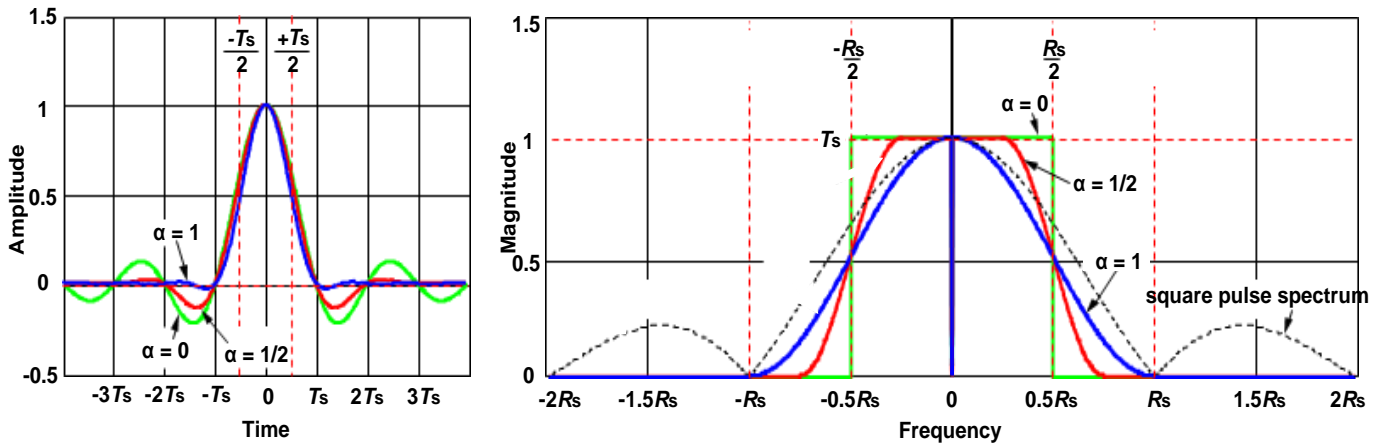


Figure 2.5: Time and frequency responses of a raised cosine filter for three values of the roll-off factor α (from Gentile, 2007:3;4)

2.3.1.4. Practical signal bandwidth

Pulse-shaping allows for a more practical expression of the transmitted signal bandwidth; in *baseband transmission*, the bandwidth (B_{T-BB}) of the transmitted pulses can be expressed in terms of the roll-off factor (α) and the desired data rate ($R_b = R_s$) as (Simon, 2001:223):

$$B_{T-BB} = 0.5(1+\alpha)R_s \quad [\text{Hz}] \quad (2.7)$$

In *pass-band transmission*, the expression of the bandwidth of the transmitted signal will depend on the type of modulation used. For modulation schemes that produce a single-side-band modulated signal, the signal bandwidth (B_{T-SSB}) can be expressed in terms of the symbol rate (R_s), the data rate (R_d) and the number of symbols (M) as:

$$B_{T-SSB} = \frac{1}{2}(1+\alpha)R_s = \frac{1}{2}(1+\alpha)\frac{R_d}{\log_2(M)} \quad [\text{Hz}] \quad (2.8)$$

Also, for modulation schemes such as PSK and ASK, where the modulated signal is a double-side-band (DSB) signal, the signal bandwidth (B_{T-DSB}) can be expressed as (Tao, 2013:72):

$$B_{T-DSB} = (1+\alpha)R_s = (1+\alpha)\frac{R_d}{\log_2(M)} \quad [\text{Hz}] \quad (2.9)$$

In *band-limited applications* where high data rates are required, α is generally chosen between 0.3 and 0.5 in order to reduce signal bandwidth usage to significantly less than R_s for SSB signals and $2R_s$ for DSB signals.

FSK modulation schemes present a different expression for the bandwidth (B_{T-FSK}) of the modulated signal. The minimum acceptable bandwidth can be expressed as a function of the symbol rate (R_s), the desired data rate (R_d) and the number of symbols of the modulation scheme (M) as (Sackey, 2006:38-43):

$$B_{T-FSK}(\text{coherent}) = \frac{1}{2}M \times R_s = R_d \frac{M}{2\log_2(M)} \quad [\text{Hz}] \quad (\text{for Coherent FSK})$$

$$B_{T-FSK}(\text{noncoherent}) = M \times R_s = R_d \frac{M}{\log_2(M)} \quad [\text{Hz}] \quad (\text{for non-Coherent FSK}) \quad (2.10)$$

2.3.1.5. Bandwidth efficiency

Using all the parameters presented above, the bandwidth efficiency can be described as the ratio of the data rate (R_d) of transmission to the bandwidth of the transmitted signal (B_T), expressed in bits/s/Hz (Pasupathy, 1979:14):

$$BW_{\text{eff}} = \frac{R_d}{B_T} \quad [\text{bits/s/Hz}] \quad (2.11)$$

Bandwidth efficiency informs the designer how much bandwidth is used to transmit a signal at a certain data rate, using a given modulation scheme (Edward & David: 1994:223).

Applying equations (2.8), (2.9), (2.10) and (2.11), the bandwidth efficiency for modulation schemes that produce SSB and DSB signals is given in terms of the roll-off factor (α), the number of symbols (M) and the desired data rate (R_d) as:

$$BW_{\text{eff}}(\text{SSB}) = \frac{2\log_2(M)}{1+\alpha} \quad [\text{bits/s/Hz}] \quad (2.12)$$

$$BW_{\text{eff}}(\text{DSB}) = \frac{\log_2(M)}{1+\alpha} \quad [\text{bits/s/Hz}].$$

The bandwidth efficiency will be a maximum with $\alpha = 0$. However, $\alpha = 0$ is not often used because it leads to an infinite impulse response of the RCF, which cannot be implemented practically.

Finally, for FSK modulation, the bandwidth efficiency can be expressed in terms of the number of symbols (M) from equations (2.10) and (2.11) as:

$$\begin{aligned} BW_{\text{eff}}(\text{FSK - coherent}) &= \frac{2\log_2(M)}{M} \quad [\text{bits/s/Hz}] \\ BW_{\text{eff}}(\text{FSK - noncoherent}) &= \frac{\log_2(M)}{M} \quad [\text{bits/s/Hz}] \end{aligned} \quad (2.13)$$

These expressions represent the *maximum possible bandwidth efficiency* that can be achieved with an FSK modulation scheme with M possible symbols.

Table 2.1 and Table 2.2 summarise the maximum achievable bandwidth efficiency of various FSK and PSK modulation schemes for a desired data rate R_d .

Table 2.1: Bandwidth efficiency of various coherent and non-coherent MFSK modulation schemes

Example of modulation scheme	Coherent FSK				Non-coherent FSK		
	MSK	BFSK	8FSK	16FSK	BFSK	4FSK	16FSK
Number of symbols (M)	4	2	8	16	2	8	16
Modulation index ($\log_2(M)$)	2	1	3	4	1	3	4
Minimum signal bandwidth (BW_{FSK}), i.e. $\alpha=0$.	$1/2 R_d$	R_d	$4/3 R_d$	$2 R_d$	$2 R_d$	$8/3 R_d$	$4 R_d$
Maximum bandwidth efficiency (BW_{eff})	2	1	3/4	1/2	1/2	3/8	1/4

Table 2.2: Bandwidth efficiency of coherent MPSK modulation schemes

Example of modulation scheme	BPSK	QPSK	8PSK	16PSK	64PSK
Number of symbols (M)	2	4	8	16	64
Modulation index ($\log_2(M)$)	1	2	3	4	6
Minimum signal bandwidth (BW_{DSB}), i.e. $\alpha=0$.	R_d	$1/2 R_d$	$1/3 R_d$	$1/4 R_d$	$1/6 R_d$
Maximum bandwidth efficiency ($BW_{\text{eff}}(\text{DSB})$)	1	2	3	4	6

2.3.2. RF power efficiency (or BER performance)

RF power efficiency is the ability of a modulation technique to preserve fidelity (good BER) of the communication link even at low RF power; hence, a low SNR (Torlak, 2013:3). RF power efficiency of a modulation scheme is observed through the probability of bit error (P_b) or bit-error-rate (BER) of that modulation technique, for a given value of E_b/N_o , and hence, a given SNR (Mulally & Lefevre, nd:3). This parameter is, therefore, used to compare modulation schemes in terms of error performance.

In order to determine the probability of bit error (P_b) of a modulation scheme, the following parameters must be understood:

- signal power (S) the noise power (N);
- signal-to-noise ratio (SNR);
- symbol energy (E_s), bit energy (E_b) and noise power spectral density (N_o);
- E_s/N_o and E_b/N_o ratios; and
- probability of symbol error (P_s) and probability of bit error (P_b).

2.3.2.1. Signal power and noise power

The *signal power* (S) here refers to the average power of the modulated signal that is transmitted through the communication medium (Sklar, 2008:224).

The *noise power* (N) here refers to the average power of the impairment (noise) through the communication path (Tipper, 2015:2). It includes the noise from both the transmission medium and the components of the receiver up to the detection point (Forouzan, 2007:84; Ippolito, 2008:61).

2.3.2.2. Signal-to-noise ratio

As the name indicates, the *signal-to-noise ratio* (SNR) is a parameter that describes the power level of the transmitted signal (S) relative to the noise power (N) in the communication medium (Hranac & Currivan, 2006:1). It is expressed as:

$$SNR = 10\log(S/N) \text{ [dB]} \quad (2.14)$$

It describes how clear the signal level is relative to the noise level. Its digital equivalence is the E_b/N_o (energy per bit to noise) ratio.

2.3.2.3. Symbol period

The symbol period (T_s) is the time taken to transmit a complete symbol (or signaling element) through the communication medium; it is the inverse of symbol rate (R_s). For a modulation scheme with M symbols, and transmission data rate (R_d); T_s is:

$$T_s = \frac{1}{R_s} = \frac{\log_2(M)}{R_d} \quad [\text{second}] \quad (2.15)$$

2.3.2.4. Symbol energy (E_s) and bit energy (E_b):

The symbol energy is the energy of the transmitted signal element, and is the product of the signal average power (S) and the symbol period (T_s).

The bit energy is the energy contained in the bits used to modulate the carrier signal. It is evaluated by dividing the symbol energy by the number of bits per a symbol ($\log_2(M)$):

$$E_s = S \times T_s \quad [\text{Joule}] \quad (2.16)$$

$$E_b = \frac{E_s}{\log_2(M)} = \frac{S}{R_d} \quad [\text{Joule}] \quad (2.17)$$

2.3.2.5. Noise power spectral density (N_o)

The *noise power spectral density* (N_o) can be described as the distribution of the average noise power (N) over the transmission bandwidth (B_T) (Sklar, 2008:117) as:

$$N_o = N / B_T \quad [\text{Watt-per-Hertz}] \quad (2.18)$$

2.3.2.6. E_s/N_o and E_b/N_o ratios

The E_s/N_o ratio is the ratio of the symbol energy to the noise power spectral density and is used to estimate the probability of symbol error (P_s) for a given modulation scheme (Tao, 2013:48).

Using equation (2.16) and (2.18) above, the E_s/N_o can be expressed in terms of the S/N ratio as:

$$E_s / N_o = \frac{E_s}{\log_2(M)} = \frac{S \cdot T_s}{N/B_T} = (S/N) \cdot (B_T T_s) \quad (2.19)$$

The product ($B_T T_s$) is known as the “*time-bandwidth-product*”.

The E_b/N_0 ratio is a figure of merit of digital modulation, which helps to determine the bit-error-rate of a modulation scheme (Selva & Krejci, 2011:54). It is obtained from the S/N ratio, and is often called signal-to-noise per bit (Tipper, 2015:14). For a given value of SNR , the E_b/N_0 will be different from one modulation scheme to another, and therefore constitutes a good platform to compare the power efficiency of various modulation schemes. The relationship between the E_b/N_0 ratio and the S/N ratio can be derived in terms the signal bandwidth (B_T), symbol period (T_s) and the number of symbols of the modulation scheme (M) as (Forouzan, 2007:226):

$$E_b / N_0 = \frac{E_s / N_0}{\log_2(M)} = (S / N) \frac{B_T T_s}{\log_2(M)} \quad (2.20)$$

Using the expressions of the signal bandwidth (B_T) presented in equations (2.8), (2.9) and (2.10), for SSB, DSB and FSKs signals respectively, and knowing the modulation scheme used, the E_b/N_0 ratio can be expressed in terms of the S/N ratio and the pulse shaping factor (α). This equation can be rewritten in decibel as:

$$E_b / N_0 = \text{SNR} + 10 \log \left(\frac{B_T}{R_b} \right) \quad [\text{dB}] \quad (2.21)$$

Note that, generally, B_T (Hz) $<$ R_s (Hz), which means that E_b/N_0 (dB) $<$ SNR (dB).

2.3.2.7. Symbol error rate (SER)

The probability of symbol error (P_s) is a performance parameter of a digital communication system, which describes the probability of making a symbol error during signal detection at the receiver. It is a function of the E_s/N_0 , which in turn depends on the defined S/N ratio and the modulation scheme used.

For M -ary phase shift keying (MPSK), the probability of symbol error (P_s) can be approximated using the E_s/N_0 ratio as (Sklar, 2008:232; Glover & Grant, 1998:389):

$$P_s (M) \approx 2 \times Q \left\{ \sqrt{\frac{2E_s}{N_0}} \sin \left(\frac{\pi}{M} \right) \right\} \quad (2.22)$$

For the M -ary frequency shift keying (MFSK) modulations, which use coherent detection, the probability of symbol error (P_s) can be approximated using the E_s/N_0 ratio as (Sklar, 2008:232):

$$P_s(M) \approx \frac{1}{M} \exp\left(-\frac{E_s}{N_o}\right) \times \sum_{k=2}^M (-1)^k \times \left(\frac{M!}{k!(M-k)!}\right) \exp\left(\frac{E_s}{kN_o}\right) \quad (2.23)$$

The main reason why the probability of error is often calculated is to identify which modulation scheme yields the minimum bit errors for a given S/N ratio. Therefore, the symbol-error-rate (SER) does not really reflect the rate at which bit errors occur, since one bit error in the symbol causes a symbol error even if all other bits are correct. A more appropriate parameter called *probability of bit errors* (P_b), or *bit-error-rate* (BER), is often used.

2.3.2.8. Bit error rate (BER)

The *bit-error-rate* (BER) or *probability of bit error* (P_b) describes the probability of receiving an incorrect bit at the receiver (Mulally & Lefevre, nd:3). The BER depends on the E_b/N_o ratio, the type of bit-to-symbol assignment (or bit-mapping) used for modulation, as well as the type of modulation (amplitude, frequency or phase) used (Glover & Grant, 1998:389). For $M > 2$, the BER can be estimated from SER and the link between the two depends on the type of modulation and the type of bit-to-symbol-assignment used during the modulation. The *Gray code mapping* is the most used of all bit-mapping as it minimises the chances of having many bit errors from one symbol error (Zafar & Farooq, 2008:64). When using the Gray code bit-mapping, the relation between the P_b and P_s can be approximated for MPSK and coherent MFSK modulations as (Glover & Grant, 1998:390; Pursley, 2005:379):

$$P_b \approx \frac{P_s}{\log_2(M)}, \quad (\text{for PSKs}) \quad (2.24)$$

$$P_b \approx \frac{P_s M}{2(M-1)}, \quad (\text{for FSKs}) \quad (2.25)$$

2.3.2.9. Summary

In summary, power efficiency is a very important parameter as it allows the designer to know which modulation schemes will be more suitable in power-limited applications. The most power efficient of two modulation schemes is the one that provides a lower BER from a given value of S/N or E_b/N_o .

2.3.3. System complexity

The complexity here refers to the level of difficulty and expertise involved in the implementation of a given modulation scheme. It also includes practical parameters, such as the *size* of the resulting system and the *power consumption*. Some modulation schemes may exhibit good spectral and power efficiencies, but being too complex to implement may convince the designer to consider an alternative technique that is less complex to implement (Haykin, 2000:419).

2.3.4. Communication medium

The communication medium, or channel, plays an important role in selecting a modulation scheme. The different types of communication mediums that exist in free-space communication include:

- multi-path fading channel; and
- non-fading channel

Different channels affect the transmitted signal differently; some making the reconstruction of the signal more complex than others. As a result, certain modulation schemes are not suitable to use in certain channels because the channel will affect the transmitted modulated signal in such a way that could make its demodulation unpractical. For example, coherent modulations are generally avoided in multi-path fading channels (Sklar, 2008:541). Therefore, it is necessary to consider the type of communication channel when selecting the digital modulation scheme for an application in order to ensure that the selected modulation scheme will be able to preserve the fidelity of the communication system (Hranac & Currivan, 2006:1).

2.3.5. Summary

The spectral efficiency and BER performance are the main parameters used to compare digital modulation schemes for a specific application. The selection of a suitable modulation scheme for a given application remains an important trade-off process between spectral efficiency, power efficiency, and complexity.

2.4. Comparison of digital modulations schemes

The selection of a modulation scheme depends on how well it meets the requirements of the application of interest in terms of the performance parameters discussed above.

2.4.1. Application of interest

The communication system required in this work is for nanosatellite space-to-earth communication links (both uplinks and downlinks) in the amateur frequency bands. The major constraints that drive the implementation of the desired communication systems can be describe as follows:

1. Nanosatellites in general and CubeSats in particular have extremely limited resources, such as available size, weight and power (*SWaP*). The limitation of size also necessitates the use of electrically small antennas with relatively low gain. Consequently, the signal-to-noise ratio of the signal at the receiver will be limited. This means that in order to achieve a good BER through the link, a modulation scheme that exhibits a relatively good BER performance even at modest *SNR* levels should be used.
2. The amateur frequency bands to be used are limited in terms of bandwidth; whereas relatively high data rates are aimed to be achieved through this communication system. Therefore, there is a need to use modulation schemes that exhibit good bandwidth efficiency in order to achieve the desired high data rate.
3. Limited level of complexity is required, which generally reduces the DC power consumption compared to more complex systems architectures.
4. The communication medium is noisy but not fading, and therefore, both coherent and non-coherent modulation could be utilised for the link.

The selection of the most suitable modulation technique for this application will now be covered, considering the constraints of the application as defined above.

2.4.2. General remarks on non-coherent detection and amplitude shift keying modulation schemes

Non-coherent detection techniques do not require timing knowledge of the incoming carrier and result in poorer power efficiency than coherent detection techniques. They consequently exhibit an increased bit-error-rate (Sklar, 2008:171). They are of great interest in applications where the channel distorts the signal severely, such as multi-path fading channels. Most satellites communicate with the ground station via a direct link through free space (Ippolito, 2008:204). Although the space environment is quite noisy and attenuates the signal considerably, it is not seen as a fading channel when used for direct link. For this reason, coherent detection modulation is preferred (Webber & Dahnoun, 1996:12; Fitton, 2012:22).

In the ASK, the information is carried in the amplitude of the carrier signal. This leads to a signal that has a non-constant envelope and that is very susceptible to noise (Deng *et al.*, 2006: 2577; Fitton, 2012:5). Because the space-to-earth medium is quite noisy, ASK modulation techniques are not suitable for wireless and satellite applications (Mishra *et al.*, 2010:553; Webber & Dahnoun, 1996:11). For this reason, the choice narrows to coherent FSK and PSK modulations, and possibly a form of CPM called MSK modulation.

2.4.3. Comparison of coherent PSKs and FSKs

2.4.3.1. Spectral efficiency

Table 2.2 revealed that higher order PSK modulation schemes present better spectral efficiencies; that is, QPSK has a higher spectral efficiency than BPSK, but lower than the 8PSK and 16PSK. MPSK and MQAM are *spectrally efficient* modulation schemes (Vertat & Mraz, 2013:389). Table 2.1 showed that coherent MFSKs present poor spectral efficiencies, and that the efficiency deteriorates with higher values of M. This means that binary BFSK has a higher bandwidth efficiency than 4FSK and 8FSK. The two tables show that coherent BFSK has the best spectral efficiency of all coherent MFSK, but its spectral efficiency is similar to that of the BPSK, and hence, smaller than that of the QPSK and higher order MPSK schemes. However, there is a special type of coherent FSK, called the minimum shift keying (MSK), which presents the best spectral efficiency of all FSKs and has characteristics similar to QPSK. Figure 2.6 shows the power spectral density of the BPSK, MSK, and QPSK schemes.

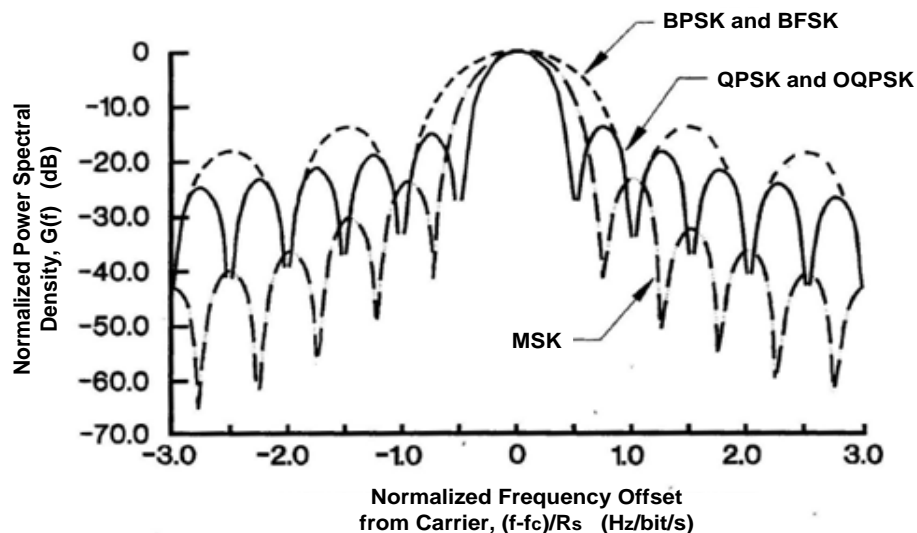
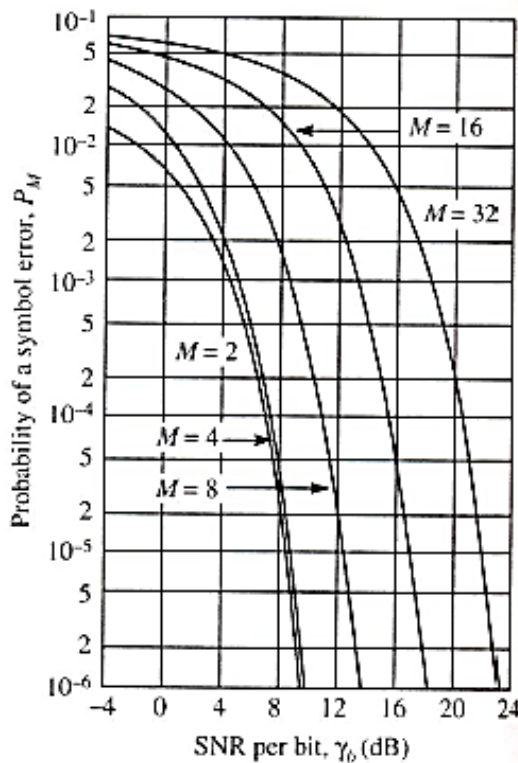


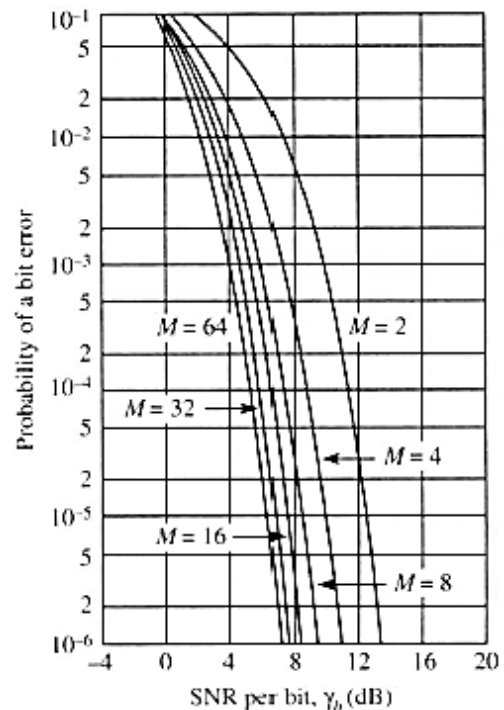
Figure 2.6: Power spectral density of MSK, BPSK, and QPSK (From Ke-Lin & Swamy, 2010:214; Torlak, 2013:3)

2.4.3.2. Bit-error-rate

Figure 2.7 (a) shows that for MPSK modulation schemes, the higher the order (M), the worse the BER performance becomes. This means that BPSK and QPSK present a better BER performance than 8PSK, 16PSK, and 16QAM ...etc. On the other hand, Figure 2.7 (b) shows that for coherent MFSK, the higher the order (M), the better the BER performance (Vertat & Mraz, 2013:389). This means that coherent BFSK and the MSK yield a worse BER performance than 8FSK, and 16FSK ...etc. From the graphs in Figure 2.7 it can be concluded that higher order MFSK give better BER performances than all MPSK schemes. However, lower order MPSK (BPSK and QPSK) present a better BER performance than lower order FSKs, such as BFSK and 4FSK. One special type of FSK, called minimum shift keying (MSK), presents characteristics similar to the QPSK in terms of both spectral efficiency and probability of bit error.



(a) M-ary PSK



(b) Orthogonal MFSK

Figure 2.7: (a) Probability of symbol error versus E_b/N_0 for coherently detected MPSK signals; (b) Probability of bit error versus E_b/N_0 for coherently detected MFSK signals (From Proakis, 1995:263; 272).

2.4.4. Choice of the QPSK modulation scheme

From the study in the previous sections, the following conclusions can be made:

1. High order MPSKs have very high spectral efficiencies and can, therefore, achieve high data rates in limited bandwidth applications. However, their poor BER performances and high DC-power consumption (Haykin, 2000:419) make them unsuitable for power-constrained applications, such as is the case for nanosatellites.
2. High order coherent MFSK have very good BER performances. However, their extremely poor spectral efficiencies make them unsuitable for such applications where high data rates are required with limited frequency spectrum available.
3. Both low order coherent MFSKs (for example, audio frequency shift keying (AFSK), BFSK, and MSK) and low order coherent MPSKs (BPSK and QPSK/OQPSK) present approximately the same BER performance, which in fact is acceptable for this application. However, of all these modulations schemes, QPSK not only presents the best spectral efficiency, but also the one that supports relatively high data rates within a limited frequency band (Memon *et al.*, nd: 2), as required in this application.

The QPSK modulation technique clearly stands out as the most suitable modulation scheme for this application. Further support for this observation is given in Table 2.3, listing various digital modulation schemes and applications in which they are most often used.

Table 2.3: Digital modulation schemes and their applications

Modulation Format	Applications
MSK, GMSK	GSM, CDPD
BPSK	Deep space telemetry, cable modems
QPSK, $\pi/4$ DPSK	Satellite, CDMA, NADC, TETRA, PHS, PDC, LMDS, DVB-S, TETS, Cable modems
OQPSK	CDMA, Satellite
FSK, GFSK	DECT, gaging, RAM mobile data, AMPS, CTs, ERMES, Land Mobil
8 and 16 VSB	North American digital TV (ATV), Broadcast, Cable
8PSK	Satellite, Aircraft telemetry pilots for monitoring broadcast video systems
16QAM	Microwave digital radio, modems, DVB-C, DVB-T
32QAM	Terrestrial microwave, DVB-T

From Table 2.3 it is evident that for satellite communications, QPSK (or 4PSK) and 8PSK are generally preferred. However, for nanosatellite communications where power limitation is a huge concern, QPSK becomes a first choice as it is more power efficient. For these reasons, the QPSK scheme was selected.

2.5. QPSK scheme

An expanded version of the systems block diagram in Figure 2.1 is presented in Figure 2.8 where the QPSK modulation and demodulation processes are outlined with basic blocks. Note that this section is not discussing the QPSK modulator and demodulator design theory, but is however, giving an overview of the QPSK modulation and demodulation processes.

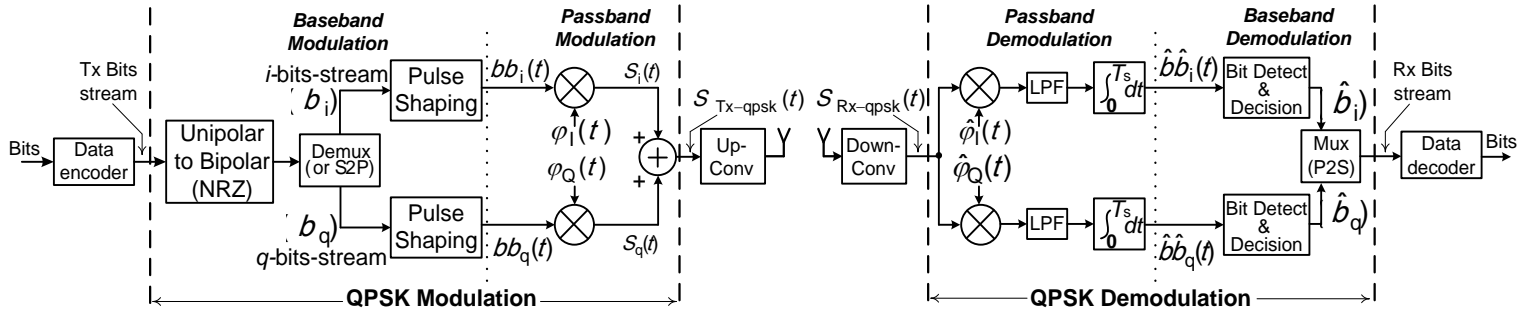


Figure 2.8: Block diagram of a digital pass band communication system employing the QPSK modulation scheme (adapted from Haykin, 2000:359).

2.5.1. QPSK modulation

2.5.1.1. Process

QPSK modulation is an advancement of the basic BPSK modulation scheme and consists of performing BPSK on two carrier signals that are 90° out of phase, using two incoming binary sequences (Hasan, *et al.*, 2010:23). The two BPSK signals obtained are then combined to form the QPSK signal as illustrated in Figure 2.8. Figure 2.9 gives a more detailed description of the modulation process with signaling information.

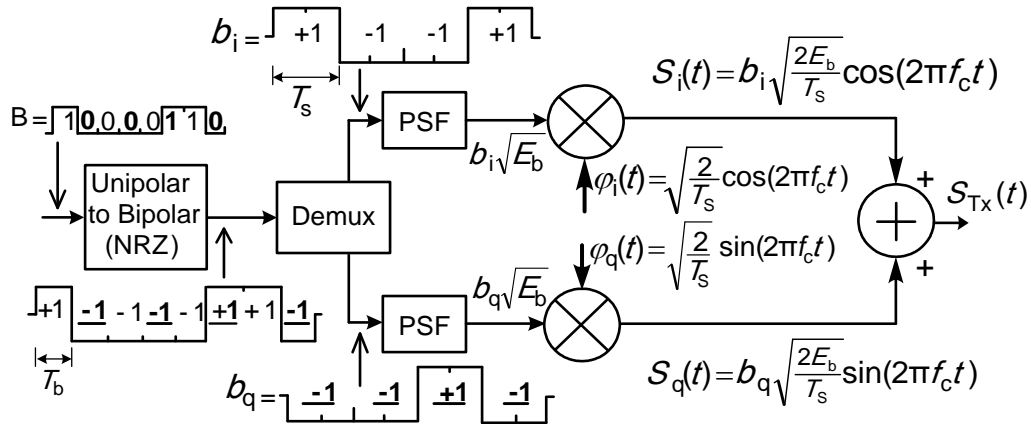


Figure 2.9: An illustration of the QPSK modulation process (adapted from Haykin, 2000:359)

With reference to above diagram, an original unipolar bit stream “B” of 1s and 0s is first transformed into a bipolar bit stream of 1s and –1s by the “*unipolar to bipolar converter block*”. The bipolar bit stream has a bit period of $T_b = 1/R_d$, where R_d is the desired bit rate. This bipolar bit stream is then separated into two bit sequences using a “*demultiplexer block*”. One sequence contains all the odd bits of the input bits stream, and is called the “*i-channel bit-stream*”, denoted “ b_i ”. The other sequence contains all the even bits of the input bits stream, and is called the “*q-channel bit-stream*”, noted “ b_q ”. The bit separation is done using bit mapping techniques, which amongst many, include “Gray-coding” and its derivatives (Zafar & Farooq, 2008:64). The period of the bit streams b_i and b_q is $T_s = 2T_b$, where T_s is the “symbol period” and is the inverse of the symbol rate ($R_s = R_d/2$). Each bit (b_i and b_q) is, thereafter, passed through a pulse shaping filter (PSF) to produce baseband signals ($b_i\sqrt{E_b}$) and ($b_q\sqrt{E_b}$), where E_b is the bit energy. The pulse-shaping process of the incoming bits is often done using either a rectangular pulse, a raised-cosine-filter or a root-raised-cosine-filter’s impulse response (Gentile, 2007:1;3). The baseband signal in the *i*-channel is used to modulate the in-phase carrier $\varphi_i(t)$, resulting in a BPSK modulated symbol $S_i(t)$, as shown in Figure 2.9 above. The baseband signal in the *q*-channel is also used to modulate the quadrature carrier $\varphi_q(t)$, resulting in another BPSK modulated symbol $S_q(t)$. The two obtained BPSK symbols are then summed over every symbol period $T_s = 2T_b$ to give one symbol of the QPSK signal ($S_{TX}(t)$), which is transmitted. The same process repeats continuously over every symbol period “ T_s ”, until all the bits of b_i and b_q are transmitted.

Each BPSK signal has two possible symbols: $S_{i1}(t)$ and $S_{i2}(t)$ for the *i*-channel BPSK; and $S_{q1}(t)$ and $S_{q2}(t)$ for the *q*-channel BPSK, as follows (Hasan, *et al.*, 2010:23; Song & Yao, nd:600):

$$\begin{aligned} S_{i1}(t) &= +\sqrt{\frac{2E_b}{T_s}} \cos(2\pi f_c t) \text{ and } S_{i2}(t) = -\sqrt{\frac{2E_b}{T_s}} \cos(2\pi f_c t) \\ S_{q1}(t) &= +\sqrt{\frac{2E_b}{T_s}} \sin(2\pi f_c t) \text{ and } S_{q2}(t) = -\sqrt{\frac{2E_b}{T_s}} \sin(2\pi f_c t) \end{aligned} \quad (2.26)$$

The QPSK signal has four possible symbols. A general expression of the QPSK symbol can be written as (Popescu *et al.*, 2011:365; Memon *et al.*, nd:2; Haykin, 2014:358):

$$S_k(t) = +\sqrt{\frac{2E_s}{T_s}} \cos\left[2\pi f_c t + (2k-1)\frac{\pi}{4}\right]; \quad 0 \leq t \leq T_s \text{ and } k = 1,2,3,4 \quad (2.27)$$

Written with all the parameters, the general expression of the QPSK signal can now be described as (Popescu *et al.*, 2011:366; Memon *et al.*, nd:3):

$$S_{Tx}(t) = b_i \sqrt{\frac{2E_b}{T_s}} \cos(2\pi f_c t) + b_q \sqrt{\frac{2E_b}{T_s}} \sin(2\pi f_c t); \quad b_i = \pm 1 \text{ and } b_q = \pm 1 \quad (2.28)$$

Every symbol of the QPSK signal is called a “dibit”, as it contains one bit from both the i - and q -components (Popescu *et al.*, 2011:366). Using Gray-code mapping, the four phase values of a symbol are 45° for (1,1), 135° for (1,0), -135° for (0,0) and -45° for (0,1) (Memon *et al.*, nd: 2). The space diagram in Figure 2.10 shows a plot of each phase and the corresponding dibit.

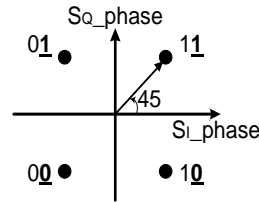


Figure 2.10: Constellation diagram of a QPSK signal (adapted from Zafar & Farooq, 2008:6; Hasan, *et al.*, 2010:23)

The period T_s of the transmitted QPSK symbol is twice the period of the original bits (i.e. $T_s = 2T_b$). Consequently, the symbol rate is half the original bit rate (i.e. $R_s = 0.5R_b$). In BPSK modulation, $T_s = T_b$ and $R_s = R_b$. For this reason, the spectral efficiency of QPSK modulation is twice that of BPSK modulation (Deng *et al.*, 2006:2578; Simon, 2001:4).

2.5.1.2. Evaluation parameters

Performance parameters measured at the output of the modulator generally include:

- symbol rate (R_s);
- bandwidth of the pass-band signal (B_{T-DSB}); and
- average power of the generated symbol (P_s) and, hence, the symbol energy (E_s).

2.5.2. QPSK demodulation

2.5.2.1. Process

Since the QPSK signal is a combination of two BPSK signals that are orthogonal to each other, each BPSK signal (the i -channel and q -channel BPSK signals) can be demodulated independently at the receiver by using the appropriate carrier for each signal. Figure 2.11 presents a basic block diagram of the QPSK demodulation process, in which the two BPSK signals are being demodulated independently on orthogonal carriers (cosine and sine).

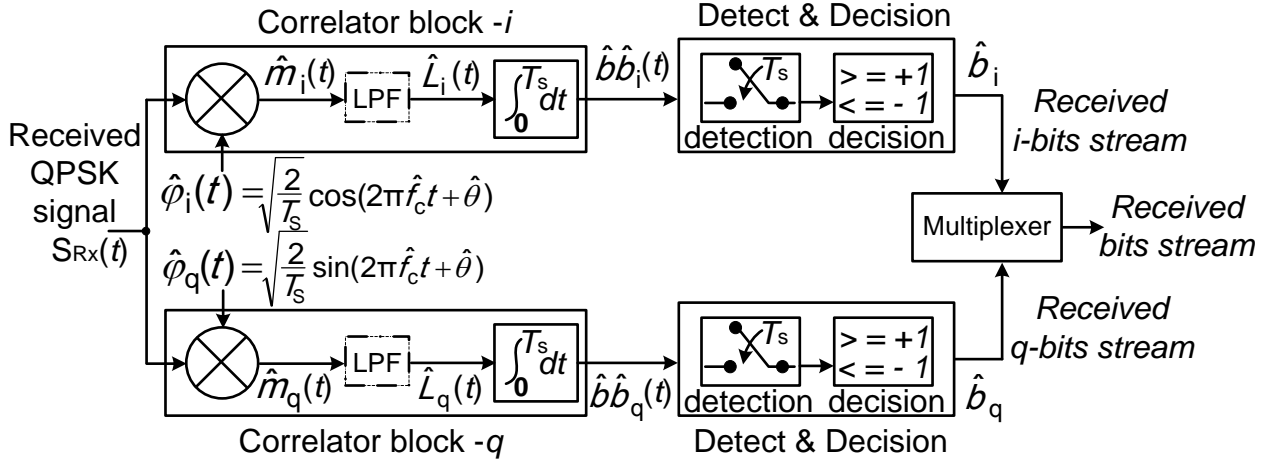


Figure 2.11: Illustration of the QPSK demodulation process (adapted from Popescu *et al.*, 2011:366; Haykin, 2014:361)

The QPSK signal arrives at the receiver with a certain phase (θ) due to travelling distance (Tao, 2013:13). For a QPSK system operating in an additive white Gaussian noise (AWGN) channel, the received noisy QPSK signal $S_{Rx}(t)$ can then be expressed as:

$$S_{Rx}(t) = b_i \sqrt{\frac{2E_b}{T_s}} \cos(2\pi f_c t + \theta) + b_q \sqrt{\frac{2E_b}{T_s}} \sin(2\pi f_c t + \theta) + n(t) \quad (2.29)$$

with $n(t)$ the white Gaussian noise with zero mean and power spectral density $N/2$ (Haykin, 2014:362). With reference to the block diagram in Figure 2.11, two orthogonal passband carriers, $\hat{\phi}_i(t) = \sqrt{\frac{2}{T_s}} \cos(2\pi \hat{f}_c t + \hat{\theta})$ and $\hat{\phi}_q(t) = \sqrt{\frac{2}{T_s}} \sin(2\pi \hat{f}_c t + \hat{\theta})$ are used to demodulate the

received QPSK symbol, where \hat{f}_c and $\hat{\theta}$ are, respectively, their frequency and phase. These two signals are called local passband carrier signals. According to the coherent detection principle, these local passband carrier signals should be identical to the passband carrier signals of the received QPSK symbol ($\phi_i(t)$ and $\phi_q(t)$); that is, \hat{f}_c should equal to f_c , and $\hat{\theta}$ should be equal to θ (Goldfarb, 2008:7; Liu *et al.*, 2008:3).

The output signals of the mixers $\hat{m}_i(t)$ and $\hat{m}_q(t)$ are expressed as ($\omega_c = 2\pi f_c$, $\hat{\omega}_c = 2\pi \hat{f}_c$):

$$\hat{m}_i(t) = b_i \sqrt{E_b} \frac{2}{T_s} \cos(\omega_c t + \theta) \cos(\hat{\omega}_c t + \hat{\theta}) + b_q \sqrt{E_b} \frac{2}{T_s} \sin(\omega_c t + \theta) \cos(\hat{\omega}_c t + \hat{\theta}) + n(t) \quad (2.30)$$

$$\begin{aligned}\hat{m}_i(t) = & b_i \sqrt{E_b} \frac{1}{T_s} \times \left\{ +\cos[(\hat{\omega}_c - \omega_c)t + (\hat{\theta} - \theta)] + \cos[(\hat{\omega}_c + \omega_c)t + (\hat{\theta} + \theta)] \right\} + \\ & b_q \sqrt{E_b} \frac{1}{T_s} \times \left\{ -\sin[(\hat{\omega}_c - \omega_c)t + (\hat{\theta} - \theta)] + \sin[(\hat{\omega}_c + \omega_c)t + (\hat{\theta} + \theta)] \right\} + n(t)\end{aligned}\quad (2.31)$$

$$\hat{m}_q(t) = b_i \sqrt{E_b} \frac{2}{T_s} \cos(\omega_c t + \theta) \sin(\hat{\omega}_c t + \hat{\theta}) + b_q \sqrt{E_b} \frac{2}{T_s} \sin(\omega_c t + \theta) \sin(\hat{\omega}_c t + \hat{\theta}) + n(t) \quad (2.32)$$

$$\begin{aligned}\hat{m}_q(t) = & b_i \sqrt{E_b} \frac{1}{T_s} \times \left\{ -\sin[(\omega_c - \hat{\omega}_c)t + (\theta - \hat{\theta})] + \sin[(\omega_c + \hat{\omega}_c)t + (\theta + \hat{\theta})] \right\} + \\ & b_q \sqrt{E_b} \frac{1}{T_s} \times \left\{ +\cos[(\omega_c - \hat{\omega}_c)t + (\theta - \hat{\theta})] - \cos[(\omega_c + \hat{\omega}_c)t + (\theta + \hat{\theta})] \right\} + n(t)\end{aligned}\quad (2.33)$$

When the conditions for coherent detection are satisfied, that is, $\hat{f}_c \rightarrow f_c$ and $\hat{\theta} \rightarrow \theta$, the expressions of the mixers' output signals can be simplified as:

$$\begin{aligned}\hat{m}_i(t) \approx & b_i \sqrt{E_b} \frac{1}{T_s} \times \left\{ +\cos[0+0] + \cos[2\omega_c t + 2\theta] \right\} + \\ & b_q \sqrt{E_b} \frac{1}{T_s} \times \left\{ -\sin[0+0] + \sin[2\omega_c t + 2\theta] \right\} + n(t)\end{aligned}\quad (2.34)$$

$$\begin{aligned}\hat{m}_q(t) \approx & b_i \sqrt{E_b} \frac{1}{T_s} \times \left\{ -\sin[0+0] + \sin[2\omega_c t + 2\theta] \right\} + \\ & b_q \sqrt{E_b} \frac{1}{T_s} \times \left\{ +\cos[0+0] - \cos[2\omega_c t + 2\theta] \right\} + n(t)\end{aligned}\quad (2.35)$$

By using a low pass filter with a cut-off frequency equal to the expected carrier frequency ($f_{cu} = f_c$), higher frequency components are removed. The baseband signals $\hat{L}_i(t)$ and $\hat{L}_q(t)$ out of the LPF are:

$$\hat{L}_i(t) \approx b_i \sqrt{E_b} \frac{1}{T_s} \times \cos(0) - b_q \sqrt{E_b} \frac{1}{T_s} \sin(0) + n(t) \approx b_i \sqrt{E_b} \frac{1}{T_s} + n(t) \quad (2.36)$$

$$\hat{L}_q(t) \approx -b_i \sqrt{E_b} \frac{1}{T_s} \times \sin(0) + b_q \sqrt{E_b} \frac{1}{T_s} \cos(0) + n(t) \approx b_q \sqrt{E_b} \frac{1}{T_s} + n(t) \quad (2.37)$$

These signals are then integrated over the period of the transmitted symbol (T_s) to produce the final baseband signals $\hat{b}\hat{b}_i(t)$ and $\hat{b}\hat{b}_q(t)$:

$$\hat{b}\hat{b}_i(t) = \int_0^{T_s} \hat{L}_i(t) dt = \sqrt{E_b} \cdot b_i + n(t) \quad (2.38)$$

$$\hat{b}\hat{b}_q(t) = \int_0^{T_s} \hat{L}_q(t) dt = \sqrt{E_b} \cdot b_q + n(t) \quad (2.39)$$

The above baseband signals will peak at an arbitrary time (τ) within the symbol length. Therefore, when captured at their peaks, and assuming the E_b/N_0 is not too poor (>3 dB), the data captured by the detection block will have the same signs as the transmitted bits (b_i and b_q). Subsequently, using a hard limiter to make a bit decision, the reproduced bits (\hat{b}_i and \hat{b}_q) will be the same as the transmitted bits:

$$\hat{b}_i = b_i \quad (2.40)$$

$$\hat{b}_q = b_q \quad (2.41)$$

Note that in the case of QPSK demodulation, two important conditions were set in order to reproduce the received bits with minimum probability of decision error:

- the condition for coherent detection must be satisfied; and
- the baseband symbols must be captured at their peak time.

Since the demodulator does not have knowledge of both the incoming phase and the possible frequency shift of the incoming QPSK carrier signals, a carrier recovery system (CRS) is generally employed in the demodulator to acquire the phase and frequency of the incoming passband carriers and adjust the phase and frequency of the local passband carriers accordingly. In a similar manner, concerning the peak time in the received symbol, ideally, when the demodulator is synchronised with the arrival of the QPSK symbols, the peak of the baseband symbol logically occurs at the symbol period T_s . This then repeats after every symbol period. However, practically, the demodulator is not always synchronised with the arrival of the signal and consequently does not have knowledge of the arrival of the symbols. Therefore, a timing recovery system (TRS) is generally employed by the demodulator in order to acquire the peak time of the reproduced baseband symbols and capture the symbols accordingly. These two subsystems of the QPSK demodulator are discussed in more detail in Chapter 4.

2.5.2.2. Evaluation parameters

Performance parameters measured at the output of the demodulator generally include:

- carrier recovery performance: acquisition time and loop bandwidth, both being evaluated from the phase response of the carrier recovery system;
- timing recovery performance: acquisition time and loop bandwidth, both being evaluated from the phase response of the timing recovery system;
- eye diagram;
- phase constellation diagram of the received symbols;
- bit stream reproduced by the detection and decision system;
- the BER through the demodulator for various values of SNR;
- achievable communication bit rate (R_d); and
- DC power consumption of the complete system for various data rates.

2.6. Chapter summary

This chapter briefly discussed digital modulation schemes. It classified the families of digital modulation, including frequency shift keying (FSK), amplitude shift keying (ASK), phase-shift-keying (PSK), and hybrids of these, for both coherent and non-coherent detection formats. The chapter also described the system performance parameters of digital modulation schemes, which include spectral efficiency, power efficiency, and complexity of the implementation. These parameters were then used to compare digital modulation schemes. It has been shown that higher order FSK modulation schemes, generally, have good power efficiency but poor spectral efficiency. Similarly, higher order PSK schemes, generally, have good spectral efficiency, but poor power efficiency. Lower order PSKs and FSKs (generally binary) exhibit similar performances; that is, good power efficiency, but poor spectral efficiency. However, quadrature phase shift keying (QPSK) modulation presents the best trade-off between power and spectral efficiency. This makes it the preferred choice for applications, such as nanosatellite missions where there is a limitation on both available power and spectrum. On this basis and on the basis of lesser complexity compared to higher order PSKs, the QPSK scheme is selected for this application. An overview of QPSK modulation and demodulation was given. The demodulator will be discussed in much more detail in Chapter 4. QPSK modulation is described in further detail in the next chapter.

CHAPTER 3

QPSK MODULATION

3.1. Introduction

Although this work focuses on the design and implementation of a QPSK demodulator, it is informative to give an overview of the design and implementation of a QPSK modulator to provide the necessary context to the subsequent work. This chapter briefly describes the generation of the QPSK signal model that will be used to evaluate the QPSK demodulator under design. Figure 3.1 recalls the QPSK modulation process.

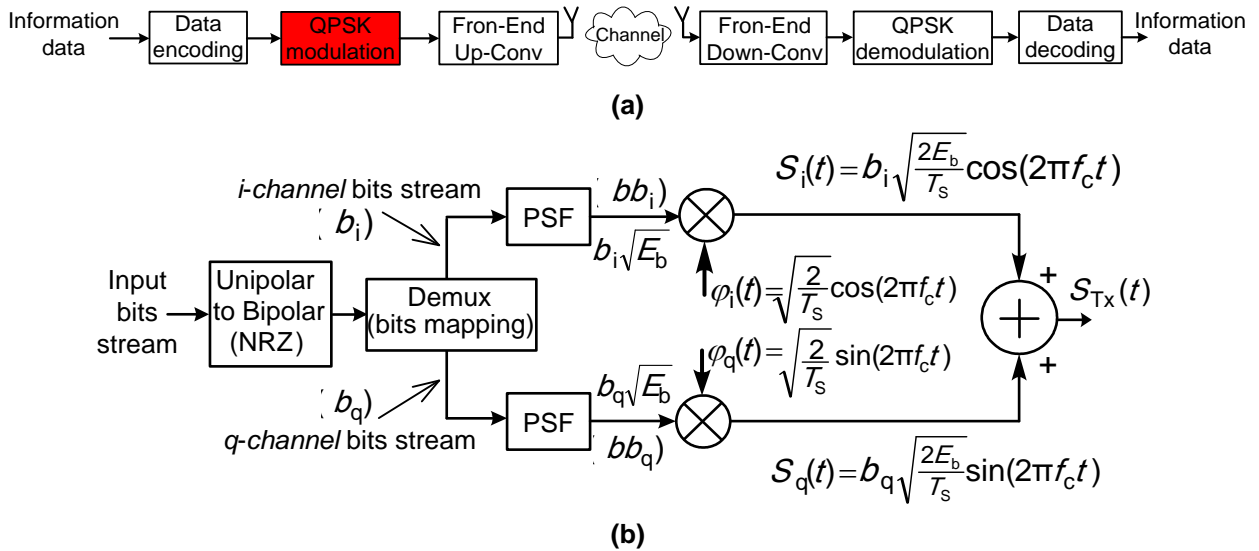


Figure 3.1: (a) Digital communication system based on the QPSK modulation scheme (adapted from Sklar, 2008:167); (b) Standard block diagram of the QPSK modulator

3.2. Transmitted QPSK signal model

The following discussion is done with reference to Figure 3.1.

3.2.1. Unipolar to bipolar non-return-to-zero process

The modulator receives the bit stream that has to be transmitted. Such a bit stream is often a combination of the information bit stream and additional encoding bits. With reference to the modulator, this bit stream is called the “*input bit stream*” to the modulator. Since this bit stream is in most cases a unipolar bit stream (consists of 1s and 0s), the first step in the QPSK modulator is often to make it a bipolar bit stream (consists of +1s and –1s). This is achieved by using a

presented earlier in Figure 2.10. The bit separation process has been implemented in Matlab following the Gray-code mapping, and the simulation result is displayed in Figure 3.3.

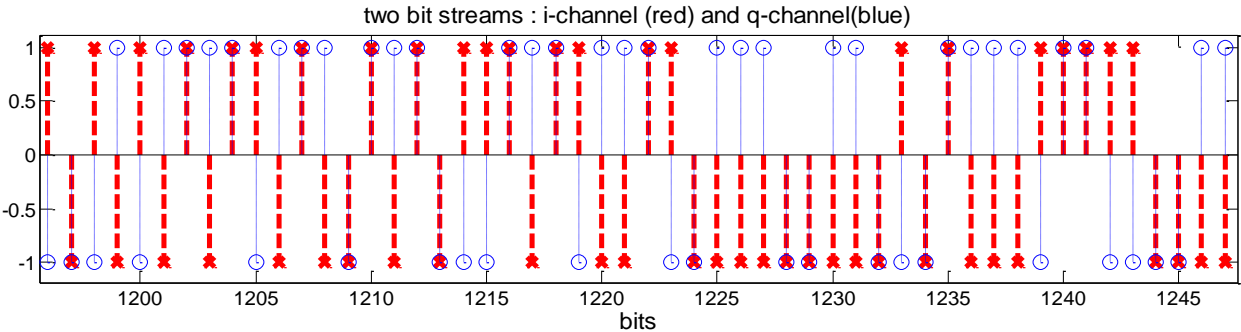


Figure 3.3: A portion of the generated *i*- and *q*- channels bit streams

3.2.3. Pulse-shaping filtering

3.2.3.1. Importance of pulse shaping filtering

Rectangular or square pulses that represent bits in a baseband signal pose challenges both in time and frequency domains.

In the frequency domain, the spectrum of an ideal square pulse extends to infinity with side lobes; in some applications, its null-to-null-bandwidth is greater than the available channel bandwidth (Lathi & Ding, 2010:78; Proakis, 1995:207). The spectrum of the rectangular pulse is illustrated in Figure 3.4.

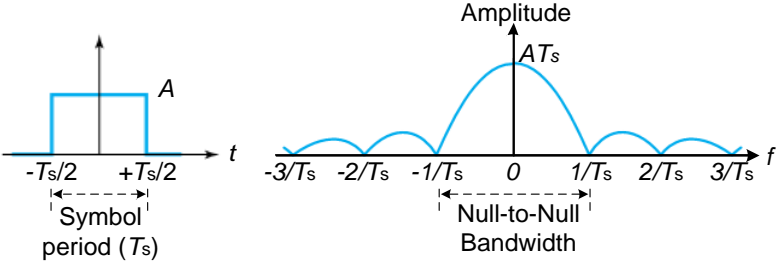


Figure 3.4: Time and frequency domain representation of a square pulse (from Proakis, 1995:207; Lathi & Ding, 2010:78)

In order to describe the time domain challenge, it is necessary to state that in most cases, for coherent detection modulation schemes, the pulse-shaping filter used for modulation is also used as the matched-filter for demodulation (Litwin, 2001:36; Ke-Lin & Swamy, 2010:225). Consequently, the output signal of the matched-filter is the convolution of two identical pulses. Therefore, for square pulses, the matched filter’s output signal is a triangular pulse. The problem

with the resulting time domain triangular pulse is that practically, it generally produces inter-symbol-interference (ISI) (Jalali, 2012:29) as illustrated in Figure 3.5.

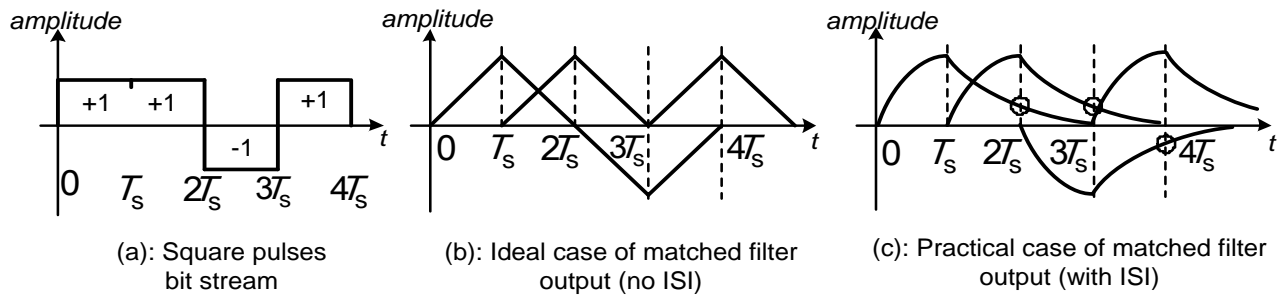


Figure 3.5: Illustration of matched filtering using a rectangular pulse (adapted from Ashok, 2007:116; Lathi & Ding, 2010:365)

Due to these outlined challenges, rectangular pulses are not often used, and pulse-shaping filters (PSF) are usually deployed. Clearly, the usefulness of the PSF is to ensure that the spectrum of the resulting baseband signal is finite and its null-to-null bandwidth is as small as possible; as well as to minimise or eliminate the ISI. Currently, some commonly used PSFs include the raised-cosine and the root-raised-cosine filters (RCF and RRCF).

3.2.3.2. Selection of the root-raised-cosine filter

The *root-raised-cosine filter* is a commonly used pulse-shaping filtering technique for PSK modulation schemes. As the name indicates, the RRCF is a filter which frequency response is literally the square root of the frequency response of the raised cosine filter: $H_{RRCF}(f) = \sqrt{H_{RCF}(f)}$ (Gentile, 2007:3-5). Note that the raised-cosine filter was discussed in Section 2.3.1.3. A few reasons for using the RRCF are that in the frequency domain, similar to the raised-cosine filter, the RRCF makes the spectrum of the resulting baseband signal finite and also minimises the null-to-null bandwidth of the signal compared to the square signal (Fitton, 2012:8). In time domain, when a root-raised-cosine pulse is transmitted, and the matched-filter's impulse response is logically a root-raised-cosine pulse, the matched-filter's output pulse that is the convolution of two RRC pulses, is a raised cosine pulse (Wong & Lok, 2014:4.8). The raised cosine pulse has an important feature in the time domain; it minimises the possibility of inter-symbol interference (ISI), as it is a maximum at $0 T_s$ and is zero at consecutive incidences of $k T_s$ (Stevens, 1995:42). T_s is the period of the transmitted symbol. Figure 3.6 illustrates a concatenation of bit pulses using a raised-cosine pulse. Therefore, by using the RRC pulse-shaping filter, the demodulated baseband pulses will be raised-cosine pulses, which have

minimum ISI. Again, it solves the time domain problem of square pulses. Subsequent to the above analysis, the root-raised-cosine pulse-shaping filter has been selected for this application.

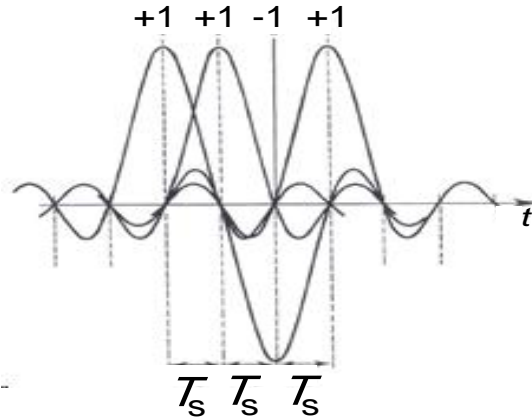


Figure 3.6: Illustration of a bit stream from a pulse shaping RCF (from Glover & Grant, 1998:364)

3.2.3.3. Design of the root-raised cosine filter

The frequency and impulse responses of the raised-cosine filter were discussed and illustrated in Section 2.3.1.3. The frequency response of the root-raised cosine filter ($H_{\text{RRCF}}(f)$) and its impulse response $h_{\text{RRCF}}(t)$ can be expressed as (Chen, 2004:25-29; Gentile, 2007:3-5; Haykin, 2014:459):

$$H_{\text{RRCF}}(f) = \begin{cases} \frac{1}{\sqrt{2B_0}}, & 0 < |f| < f_1 \quad \text{and } f_1 = (1-\alpha)B_0 \\ \frac{1}{\sqrt{2B_0}} \cos\left\{\frac{\pi}{4B_0\alpha} [f - B_0(1-\alpha)]\right\}, & f_1 < |f| < 2B_0 - f_1 \quad \text{and } f_1 = (1-\alpha)B_0 \\ 0, & |f| > 2B_0 - f_1 \quad \text{and } f_1 = (1-\alpha)B_0 \end{cases} \quad (3.1)$$

$$h_{\text{RRCF}}(t) = \frac{\sqrt{2B_0}}{1-(8\alpha B_0 t)^2} \left\{ \frac{\sin[2\pi B_0(1-\alpha)t]}{2\pi B_0 t} + \frac{4\alpha}{\pi} \cos[2\pi B_0(1+\alpha)t] \right\} \quad (3.2)$$

where $B_0 = 0.5R_s$ is the Nyquist bandwidth, and α is the roll-off factor, which is used to adjust the null-to-null bandwidth of the resulting pulse. The relationship between α and the occupied bandwidth (B_T) of the resulting pulse was given in equations (2.7) through (2.10). Pulse-shaping filters are mostly implemented using the FIR approach in order to have a finite length and be

made causal much more easily (Saramaki, 1993:155; Bozic, 1979:40). The discrete time domain expression of the above root-raised-cosine filter is (Haykin, 2014:456-462):

$$h_{\text{RRCF}}(n) = \frac{\sqrt{R_s}}{1 - (4\alpha \times n/L_{\text{symbol}})^2} \left\{ \frac{\sin[\pi(1-\alpha) \times n/L_{\text{symbol}}]}{\pi \times n/L_{\text{symbol}}} + \frac{4\alpha}{\pi} \cos\left[\frac{\pi(1+\alpha) \times n}{L_{\text{symbol}}}\right] \right\} \quad (3.3)$$

where $-L_{\text{RRCF}}/2 \leq n \leq +L_{\text{RRCF}}/2$, and $L_{\text{RRCF}} = k \times L_{\text{symbol}}$ is the desired length of the filter, L_{symbol} is the desired symbol length and k is an arbitrary integer number generally greater than 1. To complete the design of the digital RRCF described by the impulse response in equation (3.3), one needs to define α and L_{RRCF} .

In terms of defining the length of the pulse-shaping filter (L_{RRCF}) for this application, the symbol length needed for the generation of the pulse shaping filter's coefficients has been chosen arbitrarily as 16 points and the constant k as 2. Subsequently, the length of the filter is twice the symbol length; $L_{\text{RRCF}} = 32$ points.

In terms of defining the value of α , it is recalled that α falls between 0 and 1. Using a too small value (i.e. $\alpha \approx 0$) will produce a pulse with a very narrow bandwidth, but at the expense of having the pulse's impulse response with high side lobes. This may cause inter-symbol-interference (ISI) during demodulation of the baseband signal. High ISI leads to poor bit-error-rate performance. Likewise, using a too high value (i.e. $\alpha \approx 1$) will lead to a pulse's impulse response with almost no side lobes; which means no ISI (improved bit-error-rates), but at the expense of having a pulse with a wide bandwidth (Gentile, 2007:4). In addition to these guidelines, the work by Chattopadhyay & Sanyal (2009:108-111) shows that for the QPSK scheme, when using a raised-cosine or root-raised-cosine filter, the range of α values between 0.22 and 0.37 provide the most reasonable overall system' performance in terms of phase error, bandwidth efficiency and bit-error-rate. That work further shows that $\alpha \approx 0.35$ gives the best possible performance trade-off between the generated signal phase error, occupied bandwidth efficiency and bit-error-rate (Chattopadhyay & Sanyal, 2009:110;111). On the basis of these reasons, the value of α for this work has been chosen as 0.36.

In terms of defining the symbol rate (R_s), this work aims to achieve data rates up to 10 Mbps. Therefore, the data rate of $R_d = 10$ Mbps will be used as the reference point to carry the design of the desired QPSK demodulator. Subsequently, the defined symbol rate throughout this design will be $R_s = 5$ Msps. Note, however, that this is simply to carry out the design, but the developed system will obviously be flexible to accommodate any data rate up to its achievable maximum.

The occupied bandwidth of the baseband signal is $B_T = (1 + \alpha) \cdot R_s = (1 + 0.36) \cdot 5 = 6.8$ MHz. The simulated impulse and frequency responses of the designed root-raised-cosine pulse-shaping filter are presented in Figure 3.7. The designed RRCF has 32 taps, but for a better illustration of the filter's shape, it is presented here with 96 taps.

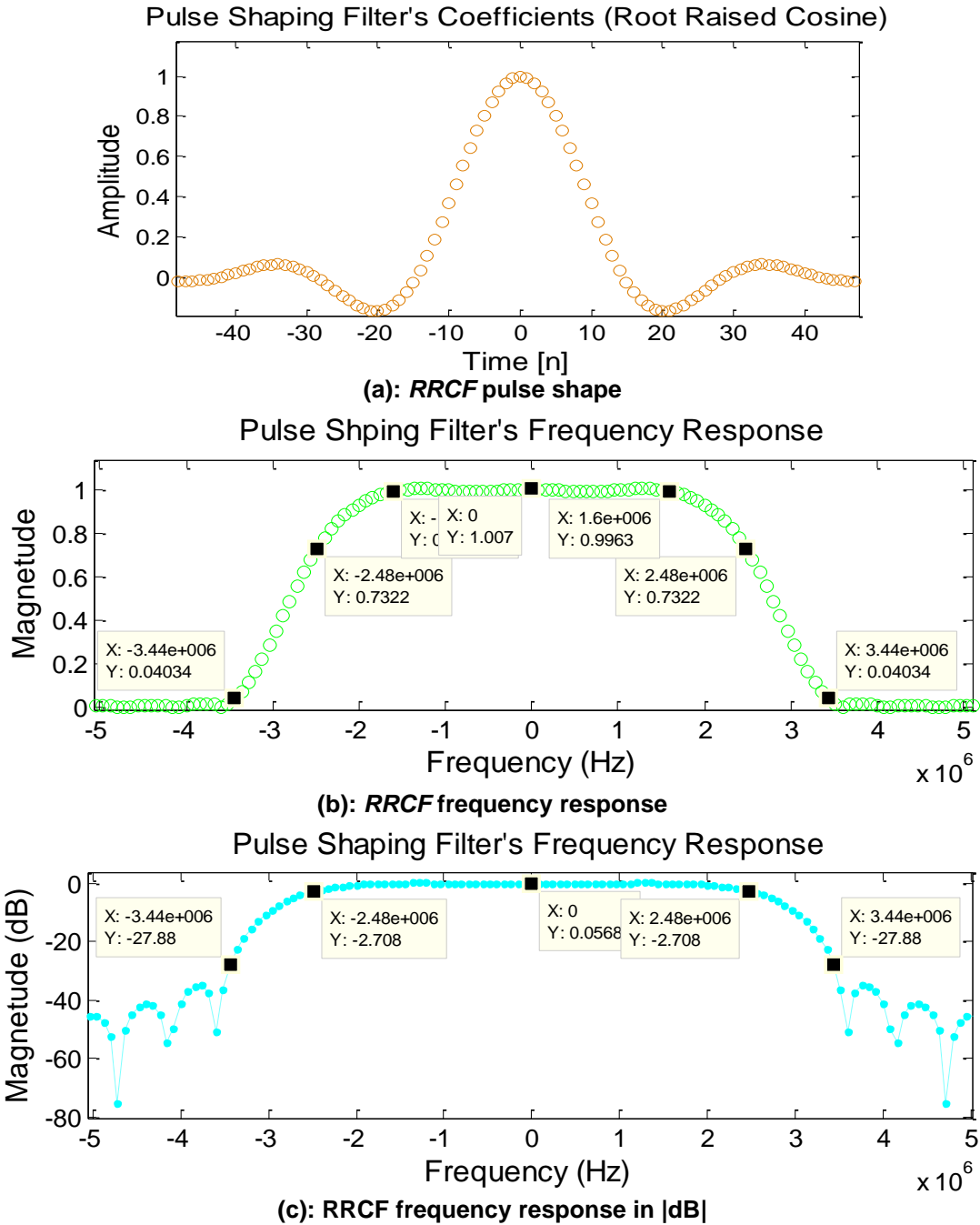


Figure 3.7: The designed root-raised-cosine filter (RRCF)

The measured null-to-null bandwidth of the filter is $3.44 - (-3.44) = 6.88$ MHz, which is very close to the theoretical null-to-null bandwidth expected (6.8 MHz). This proves that the filter is properly designed.

3.2.3.4. Resulting baseband signals

The RRC pulse-shaping filter is used to generate baseband pulse streams from the channel bit streams presented earlier in Figure 3.3. Figure 3.8 illustrates the baseband bit stream corresponding to the section of the original bit stream presented above.

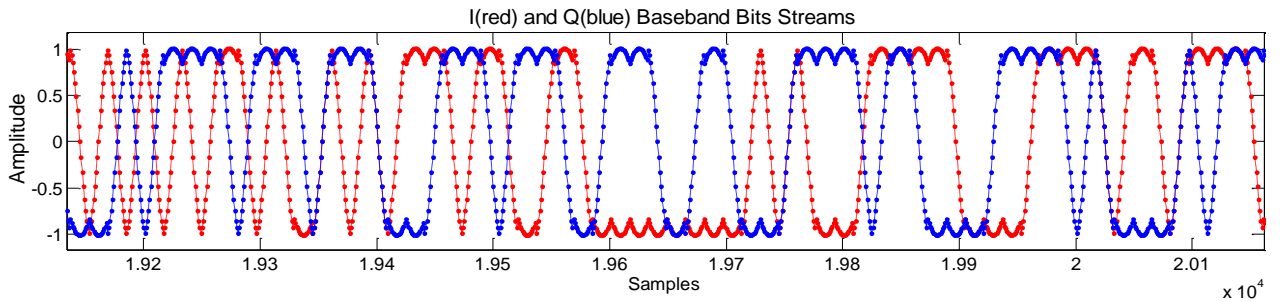


Figure 3.8: Baseband signals corresponding to the i - and q - bits streams in Figure 3.3

3.2.4. Passband carrier signals

3.2.4.1. Defining the modulation factor

Since the passband carrier signals are modulated by the stream of baseband pulses on each channel (i - and q -), it is often an important system design consideration to define how many cycles of the passband carrier should correspond to one baseband pulse. This parameter is referred to as the modulation factor (M_f) and is a positive integer, and is related to the frequency of the passband carrier through:

$$f_c = M_f \cdot R_s. \quad (3.4)$$

It is important to consider this parameters because in some cases, such as when the demodulator is implemented all-digitally, it influences the maximum data rate that can be achieved by the system. In fact, when the demodulator is implemented fully digitally, the ADC samples the passband signal and not the baseband ones. Subsequently, with N_{cycle} being the number of sample per carrier (or sampling rate), the ADC sampling frequency (f_{sa}) can be expressed as:

$$f_{\text{sa}} = N_{\text{cycle}} \cdot f_c = N_{\text{cycle}} \cdot M_f \cdot R_s \quad (3.5)$$

$$R_s = f_{sa} / (N_{cycle} \cdot M_f) \quad (3.6)$$

The equation (3.6) shows that if the ADC has a maximum sampling frequency capability (f_{sa-max}), N_{cycle} and M_f need to be their lowest possible values to achieve the maximum possible symbol rate (R_{s-max}). Therefore, the modulation factor has been set to its smallest possible value of $M_f=2$.

Note that in the case of the all-digital implementation of the demodulator under development, it will be crucial to keep the product $M_f \cdot N_{cycle}$ constant at all time, no matter the data rates at which one wants to transmit. The reason for this is that the digital matched-filter is designed to match the incoming baseband pulses, and is therefore designed to have a fixed discrete time pulse width, which is defined by $N_{symbol} = M_f \cdot N_{cycle} = constant$. Failing to uphold this constraint will produce different discrete time domain widths of the received baseband pulses, which will not match the matched-filter, and consequently will produce unexpected baseband pulses at the output of the matched filter.

3.2.4.2. Generation of the passband carrier signals

The orthogonal passband carriers signals generally used in the QPSK modulation scheme are the cosine and sine. The passband carrier frequency is $f_c = 2 \times 5 = 10$ MHz, from the discussion in the previous Section. The two passband carrier signals generated (cosine and sine) are shown in Figure 3.9. Note that the signals have normalised amplitudes.

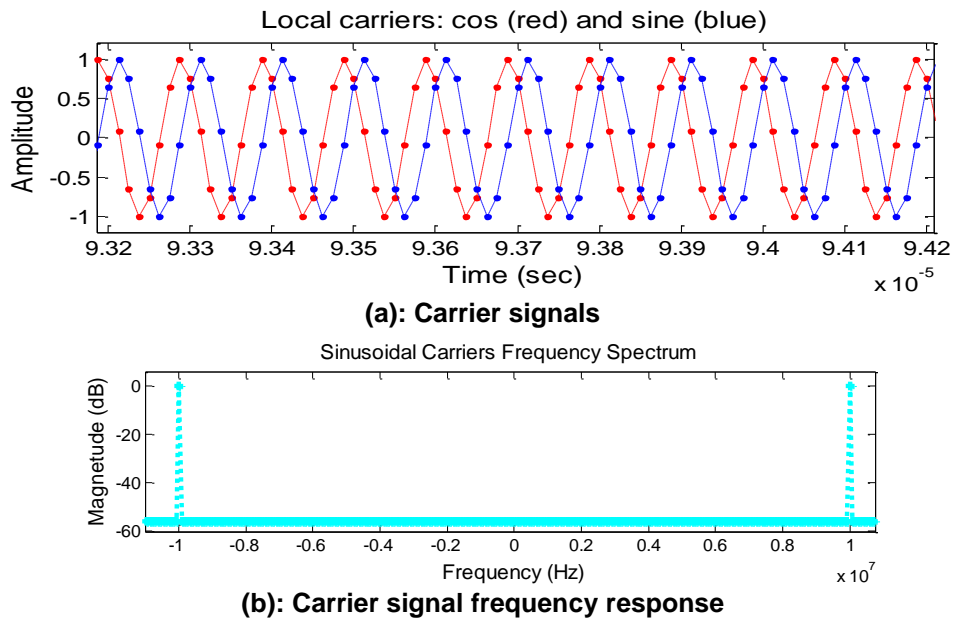
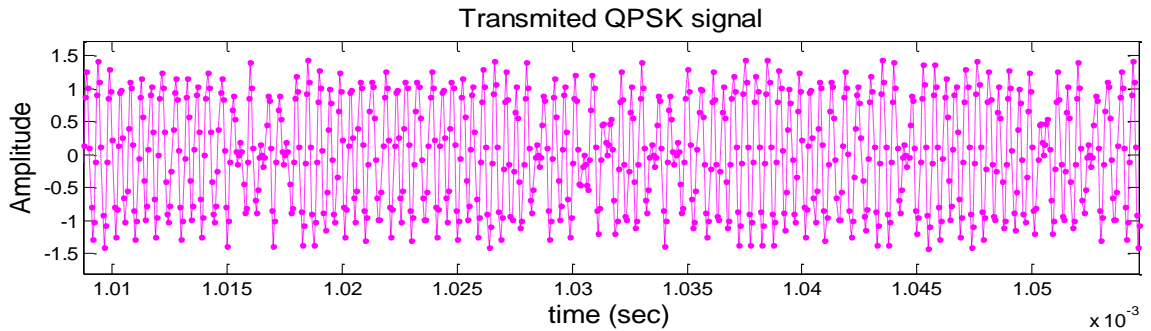


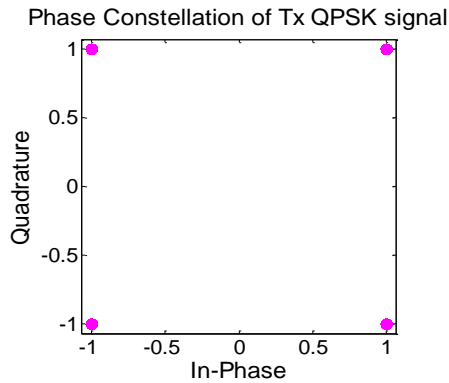
Figure 3.9: Time domain (a) and frequency domain (b) representation of the generated passband carrier signals

3.2.5. Transmitted QPSK signal

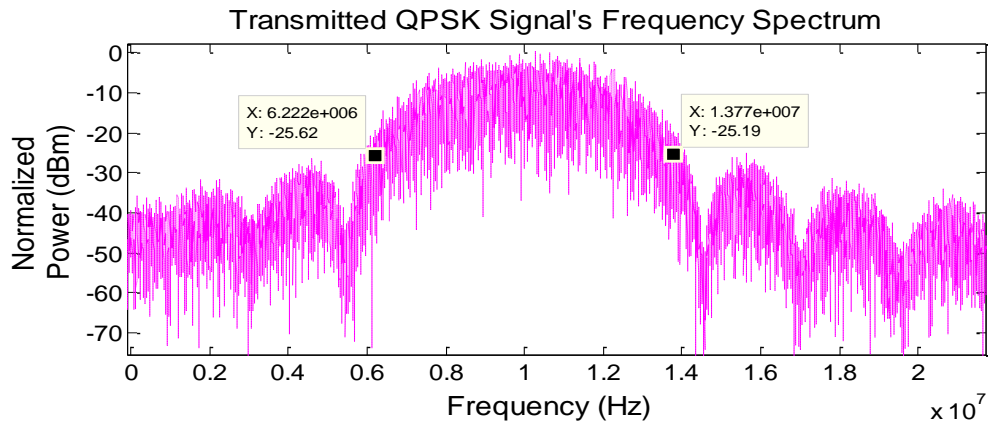
The baseband pulse streams bb_i and bb_q are used to modulate the cosine and sine carriers, respectively. This results in two BPSK-modulated signals $S_i(t)$ and $S_q(t)$. These two BPSK signals are summed over every symbol period $T_s = 1/R_s$ to form the QPSK signal that is transmitted at the desired rate R_s . Figure 3.10 displays the generated QPSK signal with normalised amplitude.



(a): Time domain representation of the transmitted QPSK signal



(b): Phase domain representation of the transmitted QPSK signal



(c): Normalised frequency response of the transmitted QPSK signal

Figure 3.10: The transmitted QPSK signal

Note that the signal is still noise-free and has normalised amplitude. The amplitude of the modulated signal will be defined as required at the input port of the demodulator.

3.3. The received QPSK signal

The received signal at the input port of the demodulator consists of the received carrier signal and additive white Gaussian noise (AWGN) picked up through the communication channel and the receiver front end block. For the design of the demodulator, it is often important to have an estimation of the received signal characteristics, such as the carrier frequency, range of amplitudes and the range of signal-to-noise ratio, in order to select an adequate ADC, as well as design the AGC accordingly. This is because these subsystems are generally developed to operate optimally over a specific range of input signal amplitudes, known as their dynamic range. However, if the estimation of the received signal amplitudes becomes a complex task, an off-the-shelf ADC can be used and the AGC designed with a reasonable dynamic range. Subsequently, the user will have to ensure that the input signal amplitudes fall within the dynamic range of the ADC and AGC, by using additional gain blocks before the demodulator.

For this application, in terms of frequency, the received carrier frequency is generally equal to that of the transmitted carrier frequency, except for possible minor frequency drifts during communication, due to the doppler effect. Such frequency drifts are compensated by the carrier recovery system, in the case of coherent detection modulation, such as the QPSK.

In terms of amplitude, the power of the received signal (P_{total}) is a combination of the received carrier signal power (P_s) and the AWGN power (P_n), such that:

$$P_s = P_n \text{ (dBm)} + SNR \quad \text{[dBm]} \quad (3.7)$$

$$P_{\text{total}} = 20 \log_{10} \left(1 + 10^{\left(\frac{SNR}{20} \right)} \right) + P_n \text{ (dBm)} \quad \text{[dBm]} \quad (3.8)$$

where, SNR is the voltage signal-to-noise ratio. The estimation of P_{total} can be done either by estimating P_s and P_n directly, or by estimating P_n and SNR . With reference to Figure 3.1 (a), the power P_s of the carrier signal received at the input port of the demodulator, through free-space, can be expressed as (Faruque, 2015:21):

$$P_s = P_{s\text{-Tx}} \cdot G_{\text{ant-Tx}} \cdot \left(\frac{\lambda^2}{4\pi d} \right) \cdot G_{\text{ant-Rx}} \cdot G_{\text{fe-Rx}} \quad (3.9)$$

where, P_{s-Tx} , G_{ant-Tx} , λ , d , G_{ant-Rx} , and G_{fe-Rx} are respectively the transmitted passband carrier signal power, gain of the transmitter antenna, signal wave length, the line of sight communication distance, gain of the receiver antenna, and the gain of the receiver front-end block. The estimation of the received carrier signal power requires knowledge of these dynamic parameters, and is, therefore, a relatively complex task. This option is discarded for this work. Considering the other option left, the expected SNR generally varies from a defined minimum up to a designated maximum value, where 0 dB is logically the worst case scenario of SNR. Again, in the case of free-space communication, the channel is modeled as an AWGN channel, with an estimated noise power spectral density $N_0 = -174$ dBm/Hz (Thomas, 2016). Consequently, for a passband signal with an occupied bandwidth B_T (Hz), the noise power (P_{n-fe}) at the input port of the receiver's front-end block (see Figure 3.1 (a)) is estimated as:

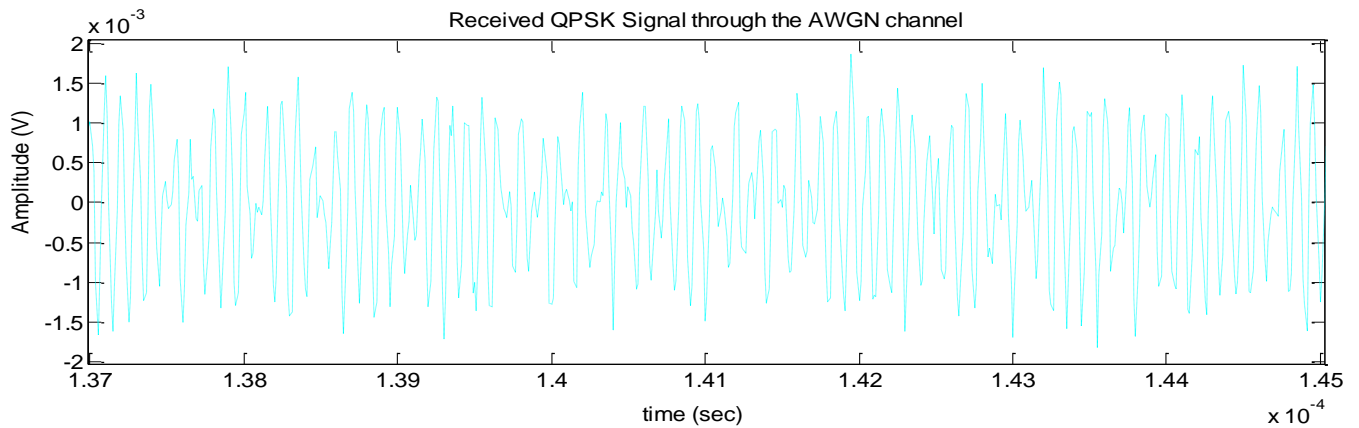
$$P_{n-fe} = -174 + 10\log(B_T[\text{Hz}]) \quad [\text{dBm}] \quad (3.10)$$

Subsequently, the noise power P_n at the input port of the demodulator can be estimated as:

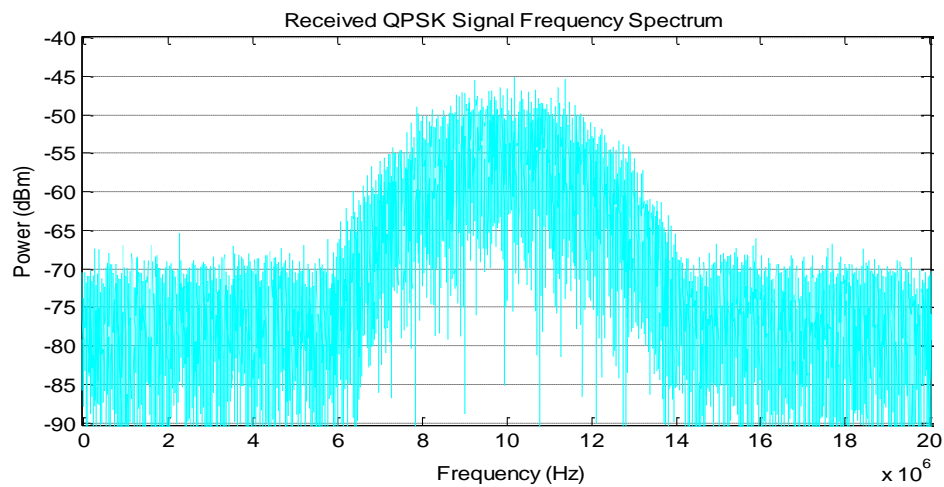
$$P_n = P_{n-fe} + G_{fe-Rx} + F_{fe-Rx} \quad [\text{dBm}] \quad (3.11)$$

where G_{fe-Rx} is the power gain of the front-end block and F_{fe-Rx} is its noise figure. Similar to the previous case, the estimation of the AWGN power at the input port of the demodulator requires knowledge of undefined parameters, such as the occupied passband signal bandwidth, receiver front-end power gain and noise figure; making, this option also relatively complex.

Therefore, to avoid the two complex estimation options above, an alternative option used for the sake of illustrating the received QPSK signal at the input port of the demodulator, has been to arbitrary set the AWGN power to $P_n = -60$ dBm. With this value of P_n , the value of P_{total} has been estimated using equation (3.8) for various values of SNR from 0 dB up to any designated maximum. However, due to space constraint in this document, only the case of 12 dB SNR is illustrated throughout this document. Subsequently, for 12 dB SNR, the received carrier signal power is $P_s = -48$ dBm and the total received signal power (including noise) is $P_{total} = -45$ dBm, which corresponds to a total voltage amplitude of $V_{peak-total} = 1.7674$ mV in a 50Ω . The time and frequency domain representations of the received QPSK signals are given in Figure 3.11.



(a): Time domain representation of the received QPSK signal



(b): Frequency spectrum of the received QPSK signal

Figure 3.11: The time and frequency representation of the received QPSK signal

The spectrum presented in Figure 3.11 (b) resembles to the one presented in Figure 3.10 (c), despite the fact that the side lobes (which can still be identified) are fading away due to the noise floor being too high. *With this generated model, the technical design of various subsystems facilitated, as well as the simulation and testing of the resulting digital products.*

3.4. Chapter Summary

This chapter presented the generation of the expected QPSK signal at the input port of the demodulator. The system design specifications have been defined, as well as all assumptions necessary to produce an accurate model of the expected signal. A QPSK signal was generated in the modulation process at the transmitter with a power normalised to 0 dBm. In order to produce accurate estimations of the expected signal power at input port of the demodulator, the noise power was first arbitrarily set to -60 dBm. The signal-to-noise ratio was assumed to range

from a minimum of 0 dB to a designated maximum. The noise power and the signal-to-noise range were then used to determine the received signal level at the input port of the demodulator for various SNR. Finally, the model of the expected QPSK signal illustrated in this document was generated with a signal-to-noise ratio of 12 dB, which results in a total signal voltage amplitude of 1.7674 mV and a signal power of about -45 dBm. This model will be used to facilitate the technical design of the demodulator subsystems as well as the necessary simulations and testing. All the power calculations had been performed assuming a 50Ω system.

CHAPTER 4: QPSK DEMODULATOR DESIGN

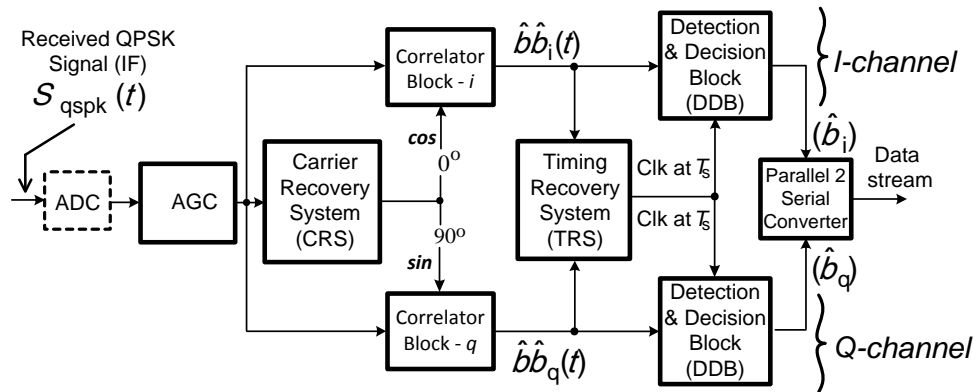
4.1. Introduction

This chapter focuses on the design of a QPSK demodulator that consists of the basic blocks as shown in Figure 2.8 and Figure 4.1. It starts with the necessary design theory, followed by the selection of the implemented technique for each subsystem. It then presents the technical design of each subsystem with reference to the selected technique and required design specifications. Since the resulting QPSK demodulator is expected to be implemented digitally, the design stage will be completed with the generation of algorithms to digitally implement each subsystem.

In addition to the *correlator block* (CB) and the *detection and decision block* (DDB) that were already highlighted, the demodulator also comprises a *carrier recovery system* (CRS) that is used to extract the desired phase and frequency from the incoming QPSK signal and adjust the local carriers accordingly to achieve “coherent demodulation”. Also, in order for the decision block to make bit decisions with minimum error, the detection block must capture the baseband bits at their peaks. However, since it has no knowledge of the peak position of the incoming baseband signal, a *timing recovery system* (TRS) is then used to extract the peak timing from the baseband signal and to transfer the information to the detection block. Optimum performance of the CRS and TRS requires the amplitude of the incoming signal to be relatively constant, when these systems employ phase-locked loop. Carrier and timing recovery systems employ phase-locked loops (PLLs) (Tatu *et al.*, 2002:2437; Booyesen, 2009:66). Therefore, in applications such as nanosatellite-to-ground communication, where the signal level varies during the satellite overpass, *automatic gain control* (AGC) is implemented at the input of the demodulator. Finally, if the design involves some software implementation, an *analog-to-digital converter* (ADC) is required. Figure 4.1 illustrates a standard block diagram of the QPSK demodulator.



(a)



(b)

Figure 4.1: (a) Digital communication system based on the QPSK modulation scheme (adapted from Sklar, 2008:167); (b) Standard block diagram of the QPSK demodulator (adapted Glover & Grant, 1998:398)

4.2. Implementation approach

The first step in the design of an electronic system is often to define whether the system would be implemented *fully analog*, *fully digital* or a *hybrid* thereof. The selection of the type of implementation is generally based on a set of parameters, which include both the application of the desired system as well as the system's design specifications.

4.2.1. Selection parameters

The parameters used to select the type of implementation of a system are briefly presented below in the context of the desired system (QPSK modulator and demodulator) and its application for nanosatellites (Webber & Dahnoun, 1996:10).

4.2.1.1. Size, weight and power

Nanosatellites have very small size and weight, and consequently limited available power for their operation. Therefore, the desired system should have a small size and weight, and very low power consumption (Larson & Wertz, 1999:307).

4.2.1.2. System quality (noise figure)

The space environment is very noisy and nanosatellites have very limited available power. These two factors lead to limited signal-to-noise ratios throughout the communication systems, and subsequently to poor bit-error-rate performance. Therefore, communications system implementations with low noise figures are critical.

4.2.1.3. System operational speed

The desired communication link is required to achieve data rates as high as 10 Mbps and more, which requires the system to have a high operational speed. The operational speed of electronic systems depends on the performance of the critical components, such as ADCs and DACs.

4.2.1.4. Flexibility and re-configurability

The desired QPSK system is part of a bigger project, which is a software defined radio (SDR). Such systems generally require their subsystems and components to be highly flexible and re-configurable for adaptive implementations.

4.2.2. Generic comparison of the three implementation approaches

Table 4.1 summarises the comparison in terms of advantages and disadvantages of the three implementation approaches.

Table 4.1: Advantages and disadvantages of the three implementation approaches

	All-analog	All-digital	Hybrid
Advantages	<ul style="list-style-type: none"> • <i>Extremely high speed:</i> The analog implementation can achieve literally any desired speed as there is no limitation due to ADC (Cagri & Schumacher, 2011:1). 	<ul style="list-style-type: none"> • <i>Relatively low space and power usage:</i> The entire system is realised in one physical programmable component. • <i>High flexibility and re-configurability:</i> System functionality is implemented in software and can therefore be modified readily (Webber & Dahnoun, 1996:10). • <i>Good system quality:</i> The system is implemented in software, which makes it almost noise-free; with noise improvement options (Mishra <i>et al.</i>, 2010:551; Sarkar, 2012:2; Torlak, 2013:3). • <i>Limited complexity:</i> There is a limited complexity of interfacing components. 	<ul style="list-style-type: none"> • <i>Very high speed:</i> In the hybrid implementation, processes of the demodulation chain that demand very fast operations would be implemented in analog; processes that do not demand very fast operation would be implemented digitally. Consequently, the hybrid implementation can also achieve very high communication speeds.

Disadvantages	<ul style="list-style-type: none"> • <i>Relatively high space and power usage:</i> Many active components are typically used. • <i>Low flexibility:</i> This implementation is not readily reconfigurable. • <i>Relatively high noise generation:</i> Many active components coupled together will lead to elevated noise generation. • <i>Quite complex to implement:</i> Many different active components to group and ensure they work well together. 	<ul style="list-style-type: none"> • <i>Limited speed:</i> The power consumption of the ADC is related to its sampling rate. The limited power available on CubeSats, therefore, limits the implemented ADC sampling rate and, hence, the overall operational speed of the device (Webber & Dahnoun, 1996:10). 	<ul style="list-style-type: none"> • <i>Medium space and power usage:</i> Active analog components used will occupy some space and use more power. • <i>Limited flexibility:</i> Subsystems that are implemented in software can be reconfigured. • <i>High self-noise generation:</i> Many active components coupled together, and transfer of data between analog and digital domains. • <i>Complex to implement:</i> It is generally a complex task to interface analog and digital components (Mahlab, 2015:4).
Possible solutions	<ul style="list-style-type: none"> • Use components with very low noise figures. 	<ul style="list-style-type: none"> • Use high speed ADCs and programmable devices. 	<ul style="list-style-type: none"> • Minimise transfer of data between analog and digital domains, and use components with low noise figures.

Considering the advantages and disadvantages of each implementation presented in this table, it seems logical that the all-digital implementation will be the most adequate choice for the desired system and application. *Therefore, the all-digital implementation is preferred in this work.* However, note that this is an overall view, and due to the operational dynamics of the subsystems of the desired demodulator, the decision of the implementation approach for each subsystem will be further discussed and supported in the theoretical design stage below.

4.3. Design theory

The theory underlying each subsystem is discussed in this section. For each subsystem, the role is briefly stated, followed by a description of relevant design techniques and the selection of the most suitable technique for this application. Thereafter, based on the selected technique, the operating principle, design assumptions and design procedure of the subsystems are presented. In closure, the performance parameters of the overall demodulator are highlighted.

The following discussions of each subsystem is done with reference to Figure 4.1.

4.3.1. Analog-to-digital converter

The analog-to-digital converter (ADC) is placed at the analog input port of any system that is implemented digitally. Figure 4.1 shows the ADC at the input port of the QPSK demodulator, as the demodulator is aimed to be implemented fully digitally. The selection of the ADC is an important stage in the design. The ADC parameters discussed below are considered when choosing an ADC for a given application.

4.3.1.1. Sampling frequency

In the case where the analog signal is a passband signal, the desired ADC sampling frequency (f_{sa}) is defined by the carrier frequency (f_c), the signal bandwidth (B_{signal}) and the desired samples-per-cycle (N_{cycle}). In this process, the Nyquist criterion must be met; that is, $f_{sa} \geq 2 f_{\text{max}}$, where $f_{\text{max}} = f_c + (B_{\text{signal}}/2)$ (Thede, 1996:202). Also, in the case where the analog signal is a baseband signal, the desired ADC sampling frequency is defined by the baseband bit rate, also known as the symbol rate (R_s) and the desired number of samples-per-symbol (N_{symbol}). Again, the Nyquist criterion must be met; that is, $f_{sa} \geq 2 f_{\text{max}}$, where $f_{\text{max}} = R_s$ (Hz).

Note that in the case where the demodulator under design is implemented digitally, both the defined modulation factor (M_f) at the modulation and the defined number of samples-per-cycle (N_{cycle}) at the receiver must be kept constant no matter the data rate at which one wants to transmit. The reason for this is that the digital matched filter is designed to match the incoming baseband pulses, and is therefore, designed to have a fixed discrete time pulse width, which is defined by $N_{\text{symbol}} = M_f \cdot N_{\text{cycle}} = \text{constant}$.

4.3.1.2. Sensitivity

The sensitivity of the ADC (S_{adc}) is the smallest variation of the input analog signal level that can be detected by the ADC (Mitra, 2006:212). It is expressed in terms of the defined ADC dynamic range (D_{adc}) and the ADC bit resolution (R_{adc}) as:

$$S_{\text{adc}} = \frac{D_{\text{adc}}}{2^{R_{\text{adc}}}} \quad [\text{V/bit}] \quad (4.1)$$

4.3.1.3. Signal-to-noise ratio

Another important parameter of the ADC is its signal-to-noise ratio (SNR), which is a function of the quantisation noise (Q_n) of the ADC. This can be expressed in terms of the ADC sensitivity as (Widrow *et al.*, 1996:357):

$$Q_{n\text{-power}} = \frac{S_{\text{adc}}^2}{12} \quad [\text{W}] \quad (4.2)$$

In a 1 Ω impedance system, the quantisation noise level can be derived as:

$$Q_{n\text{-rms}} = \frac{S_{\text{adc}}}{\sqrt{12}} \quad [\text{V}] \quad (4.3)$$

The maximum SNR of an ADC can then be expressed as (Chan & Li, 2007:11):

$$SNR_{\text{adc-max}} = \frac{V_{p\text{-adc-max-rms}}}{Q_{n\text{-rms}}} \quad (4.4)$$

In decibel (dB), the above expression results in (Man, 2012:3):

$$SNR_{\text{adc-max}} = 1.7 + 6.02 \times R_{\text{adc}} \quad [\text{dB}] \quad (4.5)$$

where R_{adc} is the ADC bits resolution. For 12 bits ADC ($R_{\text{adc}} = 12$), the maximum SNR is 74 dB.

4.3.1.4. Range of input signal amplitudes

Theoretically, the range of input signal amplitudes that the ADC will detect without distortion is between the ADC quantisation noise amplitude ($Q_{n\text{-peak}}$) and the ADC reference voltage ($V_{p\text{-adc-ref}}$). However, practically, it is advisable to set the minimum acceptable input signal amplitude to $20 \times Q_{n\text{-peak}}$, for a proper digital representation of the received signal with minimum

amplitude (Booyesen, 2017). Therefore, when selecting an ADC, one should ensure that the range of expected signal amplitudes fall within the ADC input amplitude range. Note that, if the ADC cannot accommodate the expected amplitude range of the incoming signal, one can add gain blocks before the ADC, or alternatively, an automatic gain controller (AGC) can be used before the ADC in order to provide a relatively constant signal amplitude to the ADC.

4.3.2. Automatic Gain Control

Carrier and timing recovery systems employ phase-locked loops (PLLs). It is, therefore, necessary to keep the power level of the input signal to the demodulator constant within about 1 dB, because the loop bandwidth of such PLL-based subsystems is influenced by the amplitude level of the input signal to the demodulator (Tatu *et al.*, 2002:2437; Booyesen, 2009:66). In the case of mobile communications, such as nanosatellite-to-ground-station links, the received signal levels vary considerably during the satellite’s overpass. The role of the automatic gain control (AGC) is, therefore, to ensure that an almost constant signal level is fed into the demodulator even if the received signal level varies (Brillant, 2008:517).

The AGC can be implemented in analog, or digitally. The advantage of the analog implementation is that an almost constant signal is fed to the ADC, which results in the ADC not requiring a wide digital range. However, its main disadvantage is the complexity of the implementation. On the other hand, the disadvantage of the digital implementation is that the ADC is placed before the AGC; and consequently, a signal with a range of amplitudes is fed into the ADC, forcing the ADC to have a wider digital range. But its main advantage is the reduced complexity of the implementation, compared to the analog implementation. For this work, a digital implementation has been selected. Figure 4.2 shows a block diagram of the AGC to be implemented.

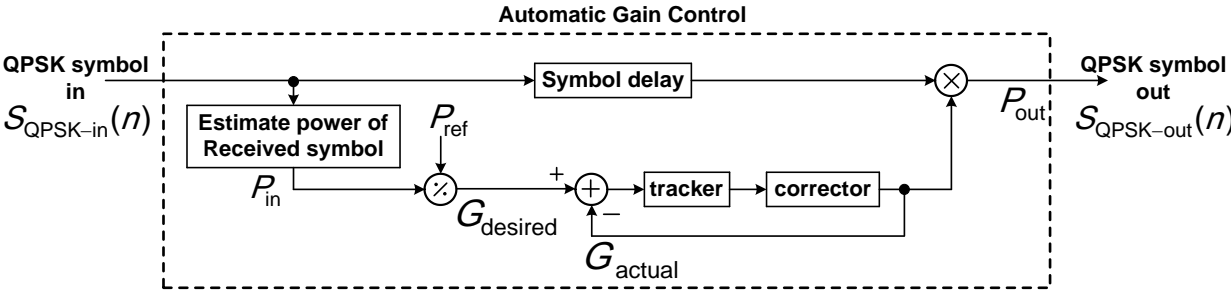


Figure 4.2: Block diagram of the AGC

The concept of the AGC relies on estimating the power of the incoming QPSK symbol using the sampled data. The estimated power (P_{in}) is compared to the reference power (P_{ref}) to define the desired gain of the AGC ($G_{desired}$):

$$G_{desired} = \frac{P_{ref}}{P_{in}} \quad (4.6)$$

A control loop is then used to lock the actual gain of the AGC (G_{actual}) to the desired gain. Once the loop system has locked, the new actual AGC gain (which is now equal to the desired gain) amplifies the input signal to produce an output signal with the desired power ($P_{out} = P_{ref}$). Note that the reference power (P_{ref}) here is the constant power desired at the output port of the AGC. An AGC is generally designed to operate optimally over a designated range of input signal amplitudes (from V_1 to V_2 in Figure 4.3), over which, it delivers the desired constant output signal amplitude (Rosu, 2017:2). If the input signal amplitude is not within the range, the AGC becomes unoperational. Figure 4.3 illustrates the ideal transfer function of the AGC.

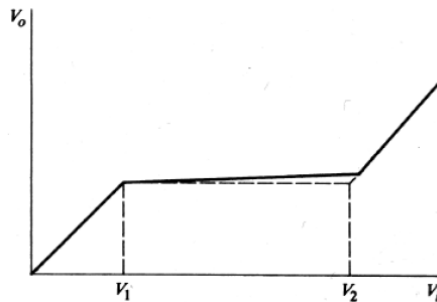


Figure 4.3: Ideal transfer function of the AGC (from Rosu, 2017:2)

From the above discussion, it is necessary to have an estimation of the maximum and minimum expected signal amplitudes, in order to define the constant output amplitude of the AGC, and subsequently design the AGC. It is advisable to use the maximum expected input signal amplitude as the constant output amplitude of the AGC, so that the minimum expected input signal amplitude corresponds to the maximum gain of the AGC.

4.3.3. Carrier recovery system

4.3.3.1. Purpose

The main purpose of the carrier recovery system (CRS) in a receiver is to ensure that the *phase* and *frequency* of the local carriers are locked to the phase and frequency of the received signal (Haykin, 2000:345). It is used to *extract* the phase of the received signal and *adjust* the phase of the local carrier accordingly (Berner, 2005:13). The phase and frequency synchronisation

process consists of an *estimation process* where a phase error is evaluated and a *correction process* where the phase of the local carrier is adjusted according to the estimated error signal (Chen, 2004:34).

4.3.3.2. Design techniques

1) Closed loop versus open loop techniques

Carrier recovery systems can be realised with either a *closed-loop (or feedback) approach*, or an *open-loop (or feed-forward) approach*. The classification is done according to how the phase or frequency offsets are extracted from the received signal (Brillant, 2008:709).

The *open-loop approach* includes techniques that use a *feed-forward* principle to achieve frequency and phase synchronisation. They are basically performed in three blocks, including frequency estimation, phase estimation and phase tracking (Chen, 2004:41). Figure 4.4 gives an illustration of the feed-forward recovery concept. The frequency offset estimator uses the input baseband symbols and some algorithms to estimate the frequency offset of the received symbol. The frequency is subsequently corrected using a look-up-table (LUT) and a phase rotator. Then, the frequency-locked symbol is used by the phase offset estimator to estimate the phase offset of the received baseband symbol. Subsequently, the phase of the baseband symbol is corrected using a phase rotator and a LUT. The final stage is the carrier tracking block where both frequency and phase offsets are tracked.

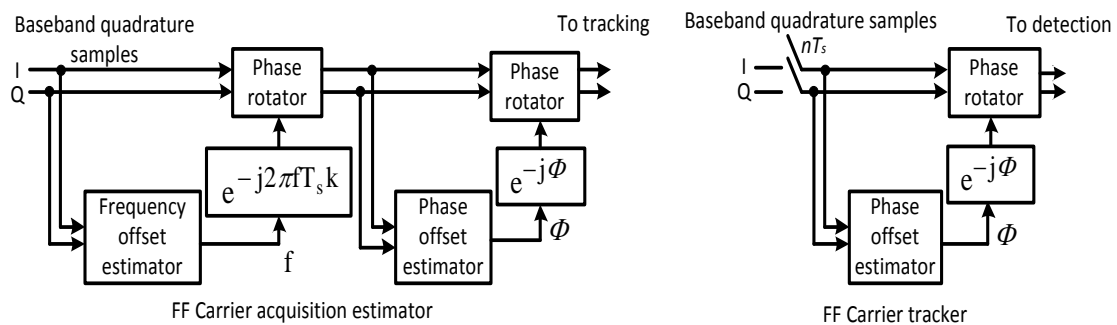


Figure 4.4: Illustration of the feed-forward carrier recovery process (adapted from Chen, 2004:41-42)

The performance of these techniques are characterised by (Chen, 2004:38; Brilliant, 2008:709-710):

- short acquisition time and large frequency offset estimation range;

- small number of samples required for accurate acquisition of unknown parameters;
- adequate for applications where spontaneous packet (bursts) transmission is done, for example in time division multiple access (TDMA) communication systems; and
- adequate for applications where the signal suffers severe deterioration in the channel.

However, some limitations of these techniques include:

- the requirement for carrier tracking for long bursts, and
- inefficient operations in continuous transmission applications.

Feed-forward approaches are not commonly used. They are mainly used in applications, such as *burst mode communication*, where fast acquisition is required; or multipath fading applications (Brillant, 2008:709; Yoshida, 1995:771). Some algorithms used for *phase estimation* in feed-forward implementations include the *maximum likelihood* (ML) estimator and variations thereof (Nicoloso, 1997:44-64; Chen, 2004:45-47).

The *closed-loop approach* includes recovery techniques that use the principle of a phase-locked loop (PLL) to acquire the phase error between the incoming carrier and the local carrier and then use the error signal to correct/adjust the phase and frequency of the local carrier (Nicoloso, 1997:35). Figure 4.5 gives an illustration of a linear phase model of a PLL in continuous and discrete time domains.

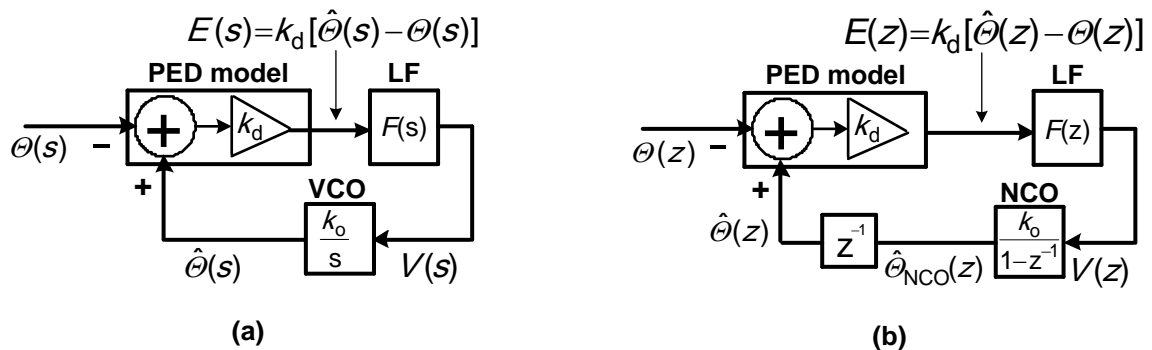


Figure 4.5: Basic illustration of a phase-locked loop, (a) continuous time domain (adapted from Nicoloso, 1997:14) and (b) discrete time domain (adapted from Booyesen, 2009:23)

Synchronisation techniques employing a closed-loop approach generally present the following advantages (Chen, 2004:38):

- no need for carrier tracking, and the recovery is continuously done in the feedback; and
- suitable for continuous or relatively long duration transmissions.

However, their main limitations include (Yoshida, 1995:771; Glover & Grant, 1998:384):

- phase ambiguity during detection, leading to possible bit inversion;
- unsuitable for applications where there is a fast variation in frequency or phase; and
- generally have a long acquisition time.

Considering the advantages listed above and based on the preference (comfort) of the designer of this system, only closed-loop carrier recovery techniques will be considered. Two such techniques are described here.

2) m^{th} power law closed-loop technique

One carrier synchronisation technique often used for MPSKs, and more specifically BPSK and QPSK, is the m^{th} power loop, where m is 2, 4 ... 16, etc. (Jain *et al.*, 2003:56). Figure 4.6 shows a block diagram of this technique.

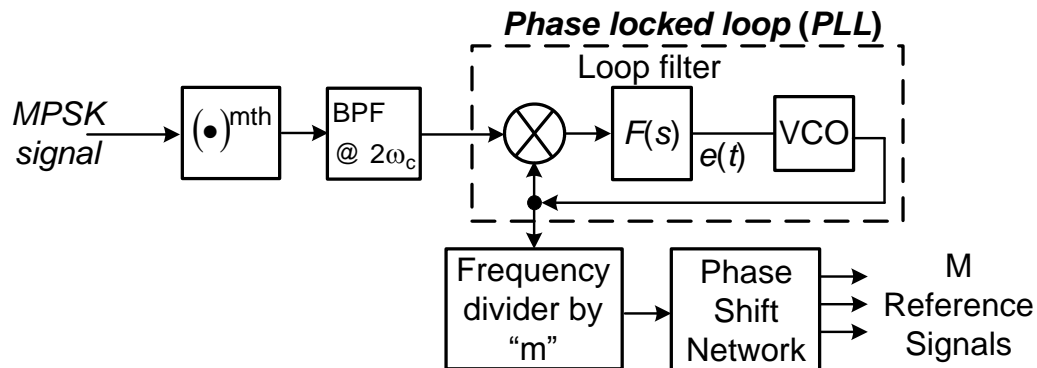


Figure 4.6: Illustration of the m^{th} power loop carrier recovery technique
(adapted from Pursley, 2005:305; Chitode, 2009:2.47)

The technique consists of raising the incoming carrier to the m^{th} -power through the m^{th} -power law block. The result then passes through a bandpass filter (BPF), which is tuned to the expected carrier frequency. The m^{th} power of the input signal multiplies the input carrier frequency by m . The phase-locked loop tracks the resulting carrier frequency, $m \times f_c$. The output frequency from the VCO, which is $m \times f_c$, is then divided by " m " to yield to the desired carrier frequency " f_c " (Sreekanth, 2003:184; Pursley, 2005:306). The phase shift network is then used to generate the different carriers that go to the correlator blocks used.

This technique is more suitable for *burst mode, fast communication* applications (Steber, 2001:5). Also, it is mostly used when the recovery system is not implemented digitally.

3) Phase error computation closed-loop technique

Closed-loop carrier recovery techniques for MPSKs that are based on the computation of the phase error between the incoming and the local carriers can be implemented using two configurations. These configurations depend on whether the phase adjustment is done on the passband signal (i.e. before the matched filter) or on the baseband signal (i.e. after the matched filter). The two configurations are presented in Figure 4.7.

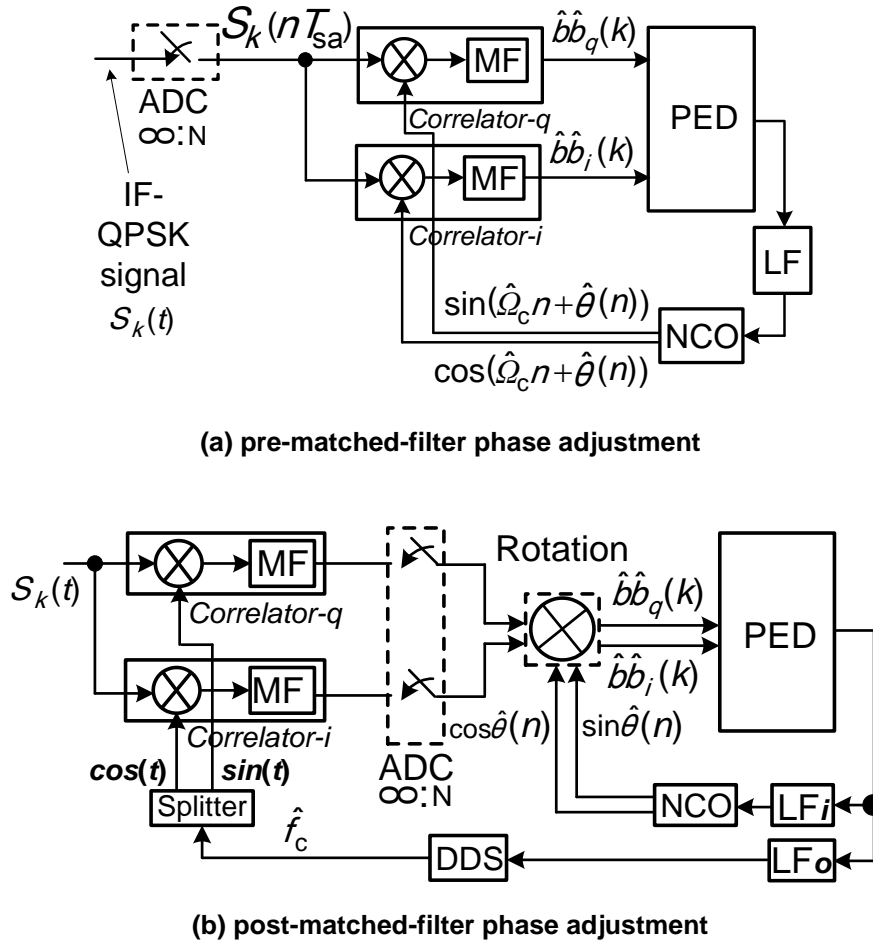


Figure 4.7: Carrier phase synchronisation configurations for MPSKs and MQAMs.

The operating principle of the system is the same for both configurations and can be described as presented below.

The *phase error detection* (PED) technique operates under the principle that when the k^{th} symbol is transmitted with bits b_i and b_q , the transmitted constellation point “ $s_k(b_i, b_q)$ ” has a phase of:

$$\theta_{Tx}(k) = \tan^{-1}(b_q / b_i) \quad (4.7)$$

The baseband bits reproduced at the demodulator ($\hat{b}\hat{b}_q$ and $\hat{b}\hat{b}_i$) produce a constellation point $\hat{s}_k(\hat{b}\hat{b}_i, \hat{b}\hat{b}_q)$ with a phase:

$$\theta_{Rx}(k) = \tan^{-1}\{\hat{b}\hat{b}_q(k) / \hat{b}\hat{b}_i(k)\} \quad (4.8)$$

Therefore, the phase error of the received k^{th} symbol is computed as:

$$e(k) = \theta_{Rx}(k) - \theta_{Tx}(k) = \tan^{-1}\{\hat{b}\hat{b}_q(k) / \hat{b}\hat{b}_i(k)\} - \tan^{-1}\{b_q(k) / b_i(k)\} \quad (4.9)$$

If the transmitted data (b_i and b_q) are not known, the bit decision on the reproduced baseband (i.e. $\text{sgn}(\hat{b}\hat{b}_i(k))$ and $\text{sgn}(\hat{b}\hat{b}_q(k))$) can then be used. This yields a phase error signal $e(k)$:

$$e(k) = \theta_{Rx}(k) - \theta_{Tx}(k) = \tan^{-1}\{\hat{b}\hat{b}_q(k) / \hat{b}\hat{b}_i(k)\} - \tan^{-1}\{\text{sgn}(\hat{b}\hat{b}_q(k)) / \text{sgn}(\hat{b}\hat{b}_i(k))\} \quad (4.10)$$

The phase error detection (PED) blocks implementing equations (4.9) and (4.10) are shown in Figure 4.8. This PED technique is known as the “arc-tan PED”; where (a) is *data-aided* (DA) and (b) is *decision-directed* (DD). The data-aided concept means that the actual data that was transmitted, is known by the receiver; the decision-directed concept means that the algorithm uses the bit decisions made, based on the received baseband symbols. The main drawback of the arc-tan PED (both DA and DD) is the difficulty to implement the *arc-tan function*.

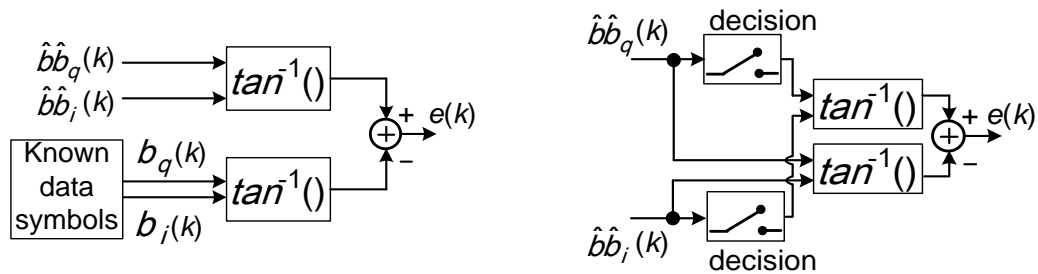


Figure 4.8: The arc-tan PEDs (DD and DA)

The phase error detector based on equation (4.9) and (4.10) will require four-quadrant arc-tan operations, which can be quite complex to compute. Therefore, a much simpler way of realising the phase error detector would be by taking the sine of the error ($e(k)$), to yield:

$$\sin(\theta_{Rx}(k) - \theta_{Tx}(k)) = \sin(\theta_{Rx}(k)) \cdot \cos(\theta_{Tx}(k)) - \cos(\theta_{Rx}(k)) \cdot \sin(\theta_{Tx}(k)) \quad (4.11)$$

$$\sin(\theta_{Rx}(k) - \theta_{Tx}(k)) = \frac{\hat{b}\hat{b}_q(k) \cdot b_i(k) - \hat{b}\hat{b}_i(k) \cdot b_q(k)}{\sqrt{(\hat{b}\hat{b}_i(k))^2 + (\hat{b}\hat{b}_q(k))^2} \sqrt{(b_i(k))^2 + (b_q(k))^2}} \quad (4.12)$$

Since the denominator of equation (4.12) is always positive, it can be absorbed in the gain (k_p) of the phase error detector; and the error signal ($e(k)$) can then be estimated using only its numerator as:

$$e(k) = \hat{b}\hat{b}_q(k) \cdot b_i(k) - \hat{b}\hat{b}_i(k) \cdot b_q(k) \quad (4.13)$$

Again, if the transmitted data (b_i and b_q) are not known, the decision of the received bits can be used, and the error signal becomes:

$$e(k) = \hat{b}\hat{b}_q(k) \cdot \text{sgn}(\hat{b}\hat{b}_i(k)) - \hat{b}\hat{b}_i(k) \cdot \text{sgn}(\hat{b}\hat{b}_q(k)) \quad (4.14)$$

The PED blocks implementing equation (4.13) and (4.14) are shown in Figure 4.9.

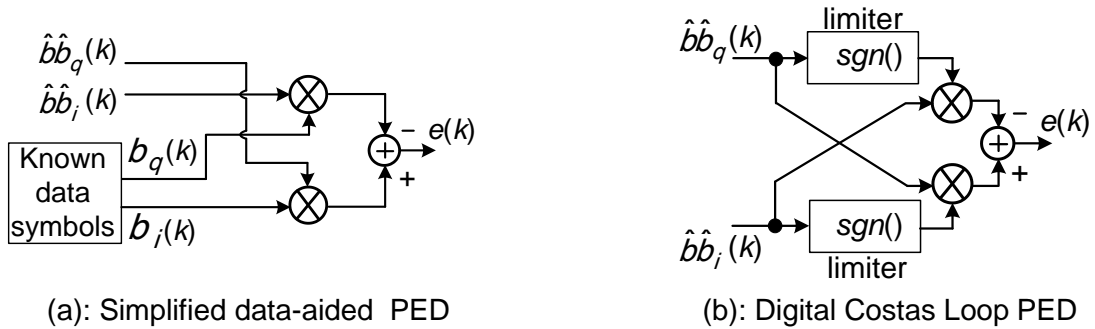


Figure 4.9: (a) Simplified data-aided PED, (b) digital Costas loop PED.

The *loop filter* following the PED block (refer to Figure 4.6) tracks the initial phase error between the incoming and the local signal. It is expressed by its transfer function $F(z)$ and will be discussed later.

A *controlled oscillator* is then used to adjust the new phase to be used by the local carriers and to generate the new local carriers after every symbol received. If configuration (b) in Figure 4.7 is used, a direct discrete synthesiser (DDS) is used as the oscillator; if configuration (a) is used, the numerically controlled oscillator (NCO) is used. The NCO is expressed by its transfer function $N(z)$ and local carrier generator block noted (LUT); these will be discussed later.

The *selection of the configuration* to use amongst the two presented in Figure 4.7 is a very important step of the system's design as it influences the performances of the demodulator, such as the achievable communication speed, the recovery system's loop bandwidth and the system complexity.

4) Selection of the single loop phase error computation carrier recovery technique

The choice of one carrier recovery technique over another is mainly driven by the application of interest and the comfort of the designer. For this application, closed-loop techniques are preferred to open-loop ones as already discussed earlier. Amongst the existing closed-loop carrier recovery techniques that can be used for MPSK demodulation, the preferred technique is the *phase error computation technique*. The main reason for this choice is that this technique is the most commonly used method for carrier synchronisation of MPSKs. It is also simpler to implement than other techniques; more especially when the *Costa's loop* PED (Figure 4.9(b)) is used. Also, it is suitable for *non-fading-channels*, and *continuous communication* applications (Ke-Lin & Swamy, 2010:227-228). For a CRS that is based on the phase error computation technique, the selection of the PED algorithm amongst the four presented (two in Figure 4.8 and two in Figure 4.9) has to be discussed and will be covered later during the PED design process.

The decision of which configuration to implement between the two presented in Figure 4.7 is generally based on which configuration will be able to achieve the desired communication speed, for a given maximum sampling frequency available, and how complex the resulting implementation is.

The *pre-matched-filter phase adjustment (or single-loop) configuration* is more suitable for either the all-analog or the all-digital implementation of the system, in which a *second-order* loop will suffice to acquire and synchronise both the input phase and frequency offsets. In the case of the all-digital implementation, both the matched-filters and the local carriers (cosine and sine) have to be generated digitally. An accurate model of the matched-filter is needed in order to reproduce the received baseband symbols accurately, and accurate models of the cosine and sine carriers are needed for a more accurate phase error detection process. In order to generate an accurate model of the matched-filter digitally, the number of samples-per-symbol (N_{symbol}) should be reasonably high. According to Booyen (2017), at least 8 samples-per-symbol will be needed to accurately model a matched filter digitally; that is, $N_{\text{symbol}} \geq 8$. In such a case, the ADC sampling frequency will be $f_{\text{sa}} = N_{\text{symbol}} \times R_s$. Likewise, in order to generate an accurate model of a cosine and sine cycle, the number of samples-per-cycle (N_{cycle}) should also be reasonably

high. Again, according to Booyesen (2017), at least 8 samples-per-cycle will be needed to produce a quite accurate model of a sine and cosine cycles; that is, $N_{\text{cycle}} \geq 8$. In such a case, the ADC sampling frequency will be $f_{\text{sa}} = N_{\text{cycle}} \times M_f \times R_s$, where M_f is the modulation factor, also known as the number of carrier cycles per transmitted symbol. The link between N_{symbol} and N_{cycle} is then $N_{\text{symbol}} = N_{\text{cycle}} \times M_f$. Therefore, during the all-digital implementation, the values of N_{symbol} and N_{cycle} to use at demodulation, as well as the value of M_f to use at modulation, should take into consideration these abovementioned limits of N_{symbol} and N_{cycle} , as well as the f_{sa} capability of the ADC and the desired maximum R_s .

The *post-matched-filter phase adjustment* (or *double loops*) *configuration* is more suitable for the hybrid implementation of the system in which the outer loop is often used for frequency synchronisation and the inner loop for phase locking. This configuration can achieve more accurate carrier recovery systems by using two loop filters with different loop bandwidths. Also, in this configuration, the local carriers and the matched-filters are implemented in analog, and the ADC sampling is done on the reproduced baseband symbols. Therefore, sampling rates as low as two samples-per-symbol (i.e. $N_{\text{symbol}} = 2$) can suffice for both the phase error detection (PED) of the carrier recovery system and the timing recovery system (TRS). Since the ADC sampling frequency will be $f_{\text{sa}} = N_{\text{symbol}} \times R_s$, for a given maximum sampling frequency, the maximum achievable communication symbol rate R_s will depend on the choice of N_{symbol} .

For this application, *the single-loop configuration is preferred to the double-loop configuration* on the basis that it is

- more suitable for the all-digital implementation of the system;
- capable of achieving the desired data rates up to 10 Mbps, even when using an ADC with a maximal sampling capability as low as 100 MSps, which would be easier to get off-the-shelf; and
- simpler to implement.

The block diagram of the *digital carrier recovery system* to be designed and implemented is presented in Figure 4.10. Figure 4.11 gives the linearized phase model of this PLL-based CRS with a closed-loop transfer function $G(z)$.

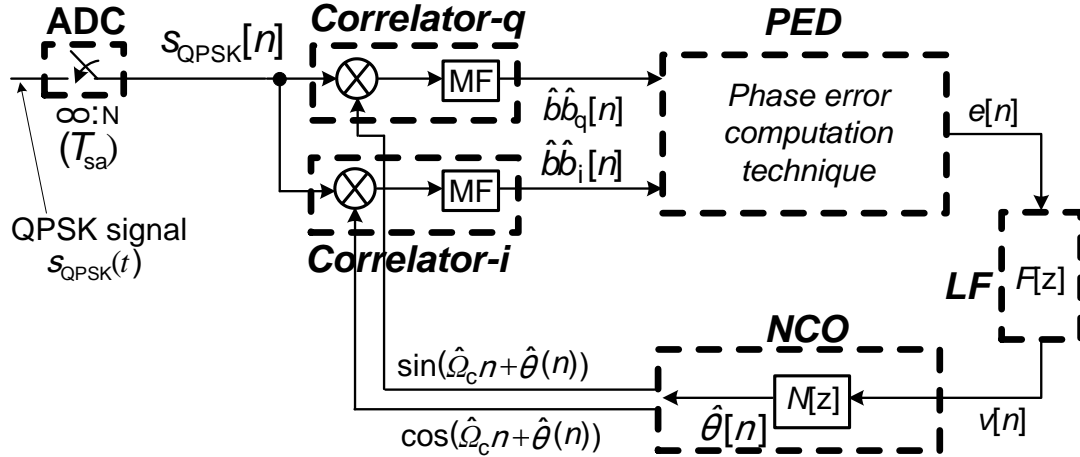


Figure 4.10: Discrete time PLL-based carrier recovery system for QPSK, using a *single-loop phase error computation technique* (adapted from Peebles, 1987:261; Dick et al., 2004:59)

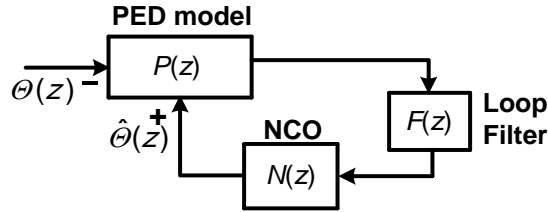


Figure 4.11: Linearised phase model of the PLL-based CRS under design

4.3.3.3. Design process of a PLL-based digital carrier recovery system

This section discusses the design process of the CRS presented in Figure 4.10, which employs the PLL principle outlined in Figure 4.11. It describes the traditional design process of a phase-locked loop and emphasises the design of each subsystem. The design stages are as follows:

- 1) identify the design parameters of the system, from both the control system theory and the digital signal processing perspectives;
- 2) define the desired PLL system characteristics; that is, the desired *order*, *loop-bandwidth* (B_L), *acquisition time* (T_{acq}) and *percentage overshoot* (PO);
- 3) express the frequency response $H(s)$ of the system based on the order defined in step 2:
 - a) express the frequency response $H(s)$ of the system in terms of the damping ratio (ζ) and natural resonant frequency (ω_n);
 - b) determine the damping ratio (ζ) and the natural resonant frequency (ω_n), in terms of the loop bandwidth (B_L), acquisition time (T_{acq}), or percentage overshoot (PO);
- 4) design the correlator blocks;
- 5) design the PED and define the transfer function $P(z)$;

- 6) design the loop filter and define the transfer function $F(z)$;
- 7) design the NCO and define the transfer function $N(z)$;
- 8) derive the system transfer function $G(z)$, with the block coefficients k_p , k_o , k_1 , and k_2 ; and
- 9) determine the control loop coefficients k_p , k_o , k_1 , and k_2, \dots, k_n , in terms of (ζ) and (ω_n) .

1) Design parameters

For the traditional phase-locked loop design process, the following parameters must be defined:

- desired system order;
- step response type and percentage overshoot (PO),
- loop bandwidth (B_L); and
- acquisition time (T_{acq}).

Likewise, for the digital implementation process, the following parameters must be defined:

- Symbol rate expected and Modulation factor: R_s and M_f
- Carrier frequency expected: $f_c = R_s \times M_f$
- Samples-per-cycle desired: N_{cycle}
- Samples-per-symbol: $N_{symbol} = M_f \times N_{cycle}$
- Sampling frequency desired: $f_{sa} = N_{cycle} \times f_c = N_{symbol} \times R_s$
- Sensitivity of the ADC: $S_{adc} = D_{adc} / (2^{R_{adc}})$

2) Define the desired system characteristics

The *order* of the system is generally the first design parameter that is defined. It describes the number of variables that can be tracked by the system. A first-order system would generally track one variable and a second-order would track two variables (Keane & Hurst, nd:1). The designed carrier recovery systems must track the phase offset and the steady-state frequency offset; therefore, a second-order PLL is preferred (Nicoloso, 1997:15).

The *percentage overshoot (PO)* is usually the next parameter to be defined for closed-loop control systems. A second order system could be under-damped (high PO), over-damped (reduced PO), or critically damped (no PO). In this application, the resulting type of response of the system was not used as the starting point of the design, but rather the desired system loop bandwidth. However, the under-damped system response is preferred for this application on

basis that a relatively short acquisition time is needed, which generally results in high percentage overshoot.

The *loop-bandwidth* and/or the *acquisition time* are the next parameters to be defined. The *loop-bandwidth* determines the noise performance of the system and the *acquisition time* determines its locking performance. If the bandwidth is wide, the acquisition time will be short; but that will lead to elevated noise and jitter in the system, which will negatively affect its performance. On the other hand, a narrow bandwidth will require a long acquisition time, but the system will be less noisy. It is good practice to keep the loop-bandwidth narrow enough to minimise noise reception, but wide enough to accommodate possible frequency drifts throughout the communication channel. Therefore, when the expected carrier frequency (f_c) is in order of MHz and beyond, it is advisable to choose a loop bandwidth $B_L \approx 1\%R_s$ (Thomas, 2016). When the control loop is implemented digitally, the specification of the loop-bandwidth takes into consideration the number of samples-per-symbol (N_{symbol}) generated by the NCO of the loop. With “ γ ” being a percentage of R_s (e.g. 1% or 2%), it can be stated that:

$$B_L = \gamma \times R_s = \left(\frac{\gamma}{N_{\text{symbol}}} \right) \times f_{\text{sa}} \quad \Rightarrow \quad B_L \times T_{\text{sa}} = \left(\frac{\gamma}{N_{\text{symbol}}} \right) \quad (4.15)$$

The relationship between the loop-bandwidth (B_L) and acquisition time (T_{acq}) is not linear, and can therefore only be simulated (Nicoloso, 1997:16; 28). However, for second-order systems, that relation can be approximated as (Booyesen, 2009:89):

$$B_{L\text{-phase}} \approx \frac{1.3}{T_{\text{acq}}} \quad (\text{for phase lock}) \quad (4.16)$$

$$B_{L\text{-frequency}} \approx \sqrt[3]{\frac{4(\Delta f)^2}{T_{\text{acq}}}} \quad (\text{for frequency lock}) \quad (4.17)$$

3) Express the frequency response of the system

The frequency response $H(s)$ of a second-order PLL can be modeled in terms of the damping ratio (ζ) of the system and the natural resonant frequency (ω_n) as (Michael, 2002:14):

$$H(s) = \frac{2\zeta\omega_n s + \omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \quad (4.18)$$

The discrete time-domain frequency response $H(z)$ can be derived from $H(s)$ using the “*Tustin’s transformation*” $\left(s = \frac{2}{T_{sa}} \times \frac{1-z^{-1}}{1+z^{-1}} \right)$, with T_{sa} the sampling period in seconds:

$$H(z) = \frac{z^{-2} \cdot \frac{2\omega_n^2}{\frac{4}{T_{sa}^2} + \frac{4\zeta\omega_n}{T_{sa}} + \omega_n^2} + z^{-1} \cdot \frac{\omega_n^2 - \frac{4\zeta\omega_n}{T_{sa}}}{\frac{4}{T_{sa}^2} + \frac{4\zeta\omega_n}{T_{sa}} + \omega_n^2} + \frac{\omega_n^2 + \frac{4\zeta\omega_n}{T_{sa}}}{\frac{4}{T_{sa}^2} + \frac{4\zeta\omega_n}{T_{sa}} + \omega_n^2}}{z^{-2} \cdot \frac{\frac{4}{T_{sa}^2} - \frac{4\zeta\omega_n}{T_{sa}} + \omega_n^2}{\frac{4}{T_{sa}^2} + \frac{4\zeta\omega_n}{T_{sa}} + \omega_n^2} + z^{-1} \cdot \frac{2\omega_n^2 - \frac{8}{T_{sa}^2}}{\frac{4}{T_{sa}^2} + \frac{4\zeta\omega_n}{T_{sa}} + \omega_n^2} + 1} \quad (4.19)$$

For a second-order closed-loop system, the damping ratio (ζ) and the natural frequency (ω_n) can be expressed in terms of the desired loop-bandwidth (B_L) as (Michael, 2002:14; 15; 21):

$$B_L \approx 0.5 \omega_n (\zeta + 1/(4 \zeta)) \quad (4.20)$$

Also, for a second-order system, the *acquisition time* (T_{acq}) tends to be optimal when $\zeta = 1/\sqrt{2}$ and the poles are near 0 (i.e. $\omega_n \ll 2\pi/T_{acq}$). It can then be expressed as (Nicoloso, 1997:22):

$$T_{acq} \approx 4/\zeta \omega_n \quad (4.21)$$

Lastly, for a second-order system the percentage overshoot can be expressed in terms of ζ as:

$$PO \approx 100e^{-\pi\zeta/\sqrt{1-\zeta^2}} \quad (4.22)$$

4) Correlator block design

In this configuration of the carrier recovery system, the correlator blocks form part of the carrier recovery loop. The design of the correlator is covered in Section 4.3.4.

5) PED design and definition of its transfer function $P(z)$

The phase error detection methods used in this type of carrier recovery technique have been highlighted earlier in Figure 4.8 and Figure 4.9. These methods include the arc-tan data-aided, arc-tan non-data-aided and simplified data-aided methods, as well as the simplified non-data-aided method, also commonly known as the digital Costas PED technique. The choice of PED technique is generally driven by the application and the comfort of the designer, since they

perform similarly. In this application, the Costas loop technique has been selected on the basis that it is the most used PED method for QPSK demodulation. It is designed to remove the 90° phase shift due to bit-transition, and only output a phase error signal relative to the input phase offset (Booysen, 2009:56). Figure 4.12 recalls the digital Costa's loop PED technique.

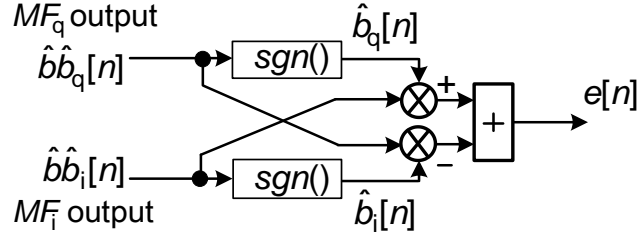


Figure 4.12: Digital Costas' loop phase error detection configuration

Substituting this Costas loop configuration into the PED block in Figure 4.10, the phase error computation process can be reviewed as described below (Tytgat *et al.*, nd:1; 4).

Neglecting the noise through the communication system, the received QPSK signal is expressed as:

$$S_{R_x}(n) = b_i \sqrt{E_b} \sqrt{\frac{2}{T_s}} \cos(2\pi f_c n T_s + n\theta) + b_q \sqrt{E_b} \sqrt{\frac{2}{T_s}} \sin(2\pi f_c n T_s + n\theta) \quad (4.23)$$

Assuming that the local carriers have the same frequency (f_c) as the incoming carrier and an initial phase $\hat{\theta}$, the local carriers signals are $\cos(2\pi f_c n T_s + n\hat{\theta})$ and $\sin(2\pi f_c n T_s + n\hat{\theta})$.

In the correlator blocks, the mixing of $S_{R_x}(n)$ with the local carrier signals, followed by the matched-filtering, will generate the baseband signals at the output of the matched-filters as:

$$\hat{b}_i[n] = b_i \cos(\hat{\theta} - \theta) - b_q \sin(\hat{\theta} - \theta) \quad (4.24)$$

$$\hat{b}_q[n] = b_i \sin(\hat{\theta} - \theta) + b_q \cos(\hat{\theta} - \theta) \quad (4.25)$$

In the *Costa's* PED, if the phase error ($\varphi = \hat{\theta} - \theta$) is so that $-45^\circ < \varphi < +45^\circ$, the outputs of the limiters will be:

$$\hat{b}_i[n] = b_i \text{ and } \hat{b}_q[n] = b_q \quad (4.26)$$

The error signal $e[n]$ can then be expressed as:

$$\begin{aligned}
e[n] &= \hat{b}_i[n] \cdot \hat{b}_q[n] - \hat{b}_q[n] \cdot \hat{b}_i[n] \\
&= b_i^2 \sin(\hat{\theta} - \theta) + b_q^2 \sin(\hat{\theta} - \theta) + b_i \cdot b_q \cos(\hat{\theta} - \theta) - b_q \cdot b_i \cos(\hat{\theta} - \theta) \\
e[n] &= 2 \sin(\hat{\theta} - \theta)
\end{aligned} \tag{4.27}$$

which can then be approximated as:

$$e[n] \propto \hat{\theta} - \theta; \quad \text{for } |\hat{\theta} - \theta| < \frac{\pi}{4} \tag{4.28}$$

It clearly results that, since the sine function is monotonic in the interval -45° to $+45^\circ$, the error signal obtained is proportional to the phase error ($\varphi = \hat{\theta} - \theta$) (Dick *et al.*, 2004:59). In addition, due to the PED configuration that compensates for the 90° phase ambiguity of the QPSK signal, the error signal obtained in the interval -45° to $+45^\circ$ will repeat itself at intervals of 90° . Consequently, the phase errors $\varphi = 0^\circ, 90^\circ, 180^\circ$ and 270° (or -90°) will produce the same error signal $e[n] = 0$. These four phase errors are called the lock-stable points. The graph of the error signal versus phase error, known as the “S-curve”, is given in Figure 4.13 for QPSK.

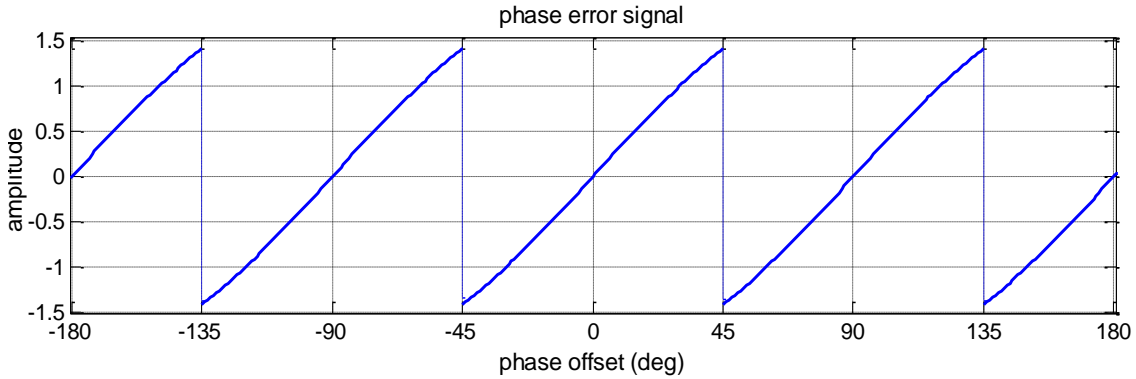


Figure 4.13: PED S-curve

From the S-curve, a linear phase model of the error signal $e[n]$ in terms of the phase error ($\varphi = \hat{\theta} - \theta$) can be derived as:

$$e[n] = k_p \cdot \varphi \tag{4.29}$$

where k_p is the gain of the PED and can be determined as follows:

$$k_p = \frac{d[e(n)]}{d\varphi} \approx \frac{d[2\sin(\varphi)]}{d\varphi} \approx 2\cos(\varphi) \tag{4.30}$$

The graph of the PED gain k_p versus phase error φ is shown in Figure 4.14.

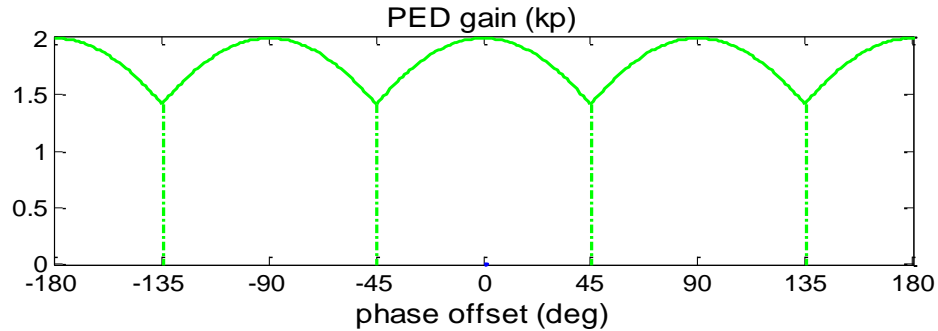


Figure 4.14: Graph of PED gain k_p versus phase error ϕ

The gain k_p oscillates between 1.414 and 2, and peaks when the phase error is a multiple of 90° . It is important to highlight that this gain is a function of the power of the input signal (Chitode, 2009:2.48). The linear phase model of the PED block shown in Figure 4.15 was drawn based on equation (4.29), and its transfer function is $P(z) = k_p$.

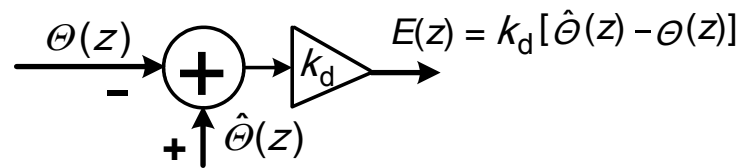


Figure 4.15: Linearised phase model of the PED block

6) Loop filter design and definition of its transfer function $F(z)$

With reference to Figure 4.10, the loop filter uses the error signal $e[n]$ to track the phase error ϕ and produces an output $v[n]$. It is at the heart of the PLL design. The order of the loop filter (or the controller) depends on the desired order of the closed-loop system under development. If the desired order is “p”, the loop filter order is “p-1”. The order of the closed-loop system is defined by the number of perfect integrators or perfect accumulators in the loop; a perfect integrator is set to have only one pole placed at $s = 0$. Consequently, the voltage-controlled oscillator (VCO) is defined as perfect integrators as it only has one pole at $s = 0$ or DC (Nicoloso, 1997:14).

In this application, the carrier recovery system needs to track both the “phase-offset” and the “frequency offset with a finite-steady-state-error”. Therefore, a second order PLL will suffice (Nicoloso, 1997:15). Consequently, a first-order loop filter will be used. The transfer function of a first-order loop filter $F_1(s)$ is given in equation (4.28) and its implementation block diagram illustrated in Figure 4.16.

$$F_1(s) = k_1 + \frac{k_2}{s} \quad (4.31)$$

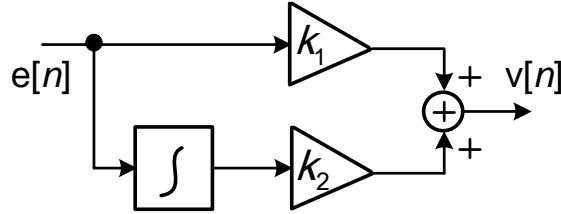


Figure 4.16: Illustration of the first order loop filter

The discrete time domain transfer function $F_1(z)$ can be obtained from the s-domain transfer functions above using the *Tustin's method* as (VadHAVKER, 2007:4; 6; 8):

$$F_1(z) = k_1 + k_2 \frac{z}{z-1} \quad (4.32)$$

The output signal $v[n]$ will be expressed in terms of the error signal $e[n]$ as:

$$v[n] = k_1 e[n] + k_2 \sum_{m=0}^n e[m] \quad (4.33)$$

Coefficients k_1 and k_2 are the design parameters of the loop filter and will be determined later.

7) NCO design and definition of its transfer function $N(z)$

The VCO is a perfect integrator, with transfer function $V(s) = k_o/s$. Consequently, using the *forward difference method* (VadHAVKER, 2007:4; 6; 8), the numerically controlled oscillator (NCO) in the discrete time-domain is a perfect accumulator with a transfer function:

$$N(z) = k_o \left(\frac{z}{z-1} \right) \times z^{-1} \quad (4.34)$$

where k_o is the gain of the NCO. It, therefore, estimates the new phase $\hat{\theta}_{\text{NCO}}[n]$ required for generation of the new local carrier signals as:

$$\hat{\theta}_{\text{NCO}}[n] = k_o \sum_{p=0}^n v[p] \quad (4.35)$$

This means that the NCO fundamentally consists of *an accumulator with a gain block*, followed by *a unit delay block* that waits for a symbol duration before it transfers out the new phase $\hat{\theta}[n]$. In addition to the fundamental blocks, the NCO is completed by a *carrier signal (cosine and sine)*

generator block when used in a recovery loop where local carrier signals must be generated. This block uses the new NCO phase $\hat{\theta}[n]$ obtained after every symbol in order to generate cycles of local carrier signals that will be used to process the next symbol. It uses a *lookup-table* (LUT) and some algorithms to adjust the phase offset (*PO*) and the frequency offset (*FO*). A model of the NCO for the carrier recovery system is given in Figure 4.17.

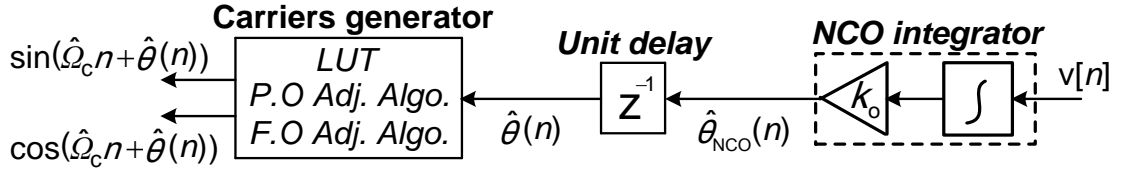


Figure 4.17: A basic block diagram of a numerically controlled oscillator (NCO)

The LUT used in the local carrier generator block contains a standard cycle of the cosine function, which is generated as follows:

$$h_{\text{LUT}}(n) = A_{\text{LUT}} \cdot \cos(2\pi n/360), 0 \leq n \leq 359 \quad (4.36)$$

where n is the number of points (coefficients) of the cycle, and A_{LUT} is the desired amplitude of the local carrier signals. Since it is better to work with integer coefficients, rather than floating point ones, the integer coefficients are obtained by using the ADC sensitivity S_{adc} as:

$$h_{\text{LUT-integers}} = \text{Round}(h_{\text{LUT}}/S_{\text{adc}}) \quad (4.37)$$

The amplitude (A_{LUT}) can be given any value between S_{adc} and $V_{\text{ref-adc-peak}}$, but it is better to keep it as low as possible, while still ensuring that the cosine cycle's shape is conserved with the resulting digital coefficients. The reason for keeping the amplitude low is to have a LUT with small integer coefficients, so to avoid large integer numbers resulting from the mixer block.

8) Deriving the system transfer function

Figure 4.18 provides a more detailed schematic of the carrier recovery system under design.

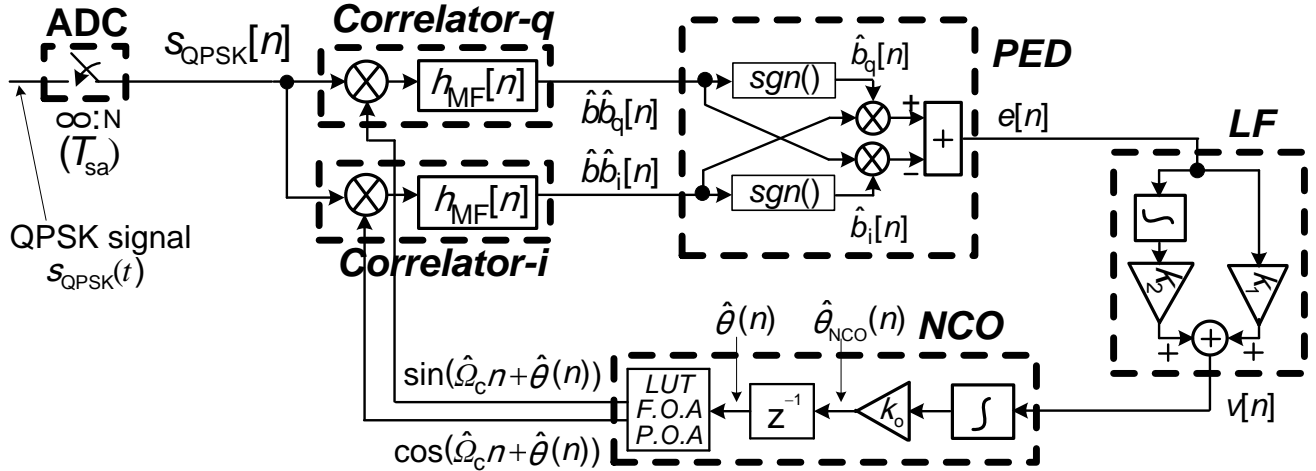


Figure 4.18: Schematic of the digital carrier recovery system under design

Using the linearised phase model of the PED, loop filter and NCO blocks described earlier, the linearised phase model of the carrier recovery system is derived as shown in Figure 4.19.

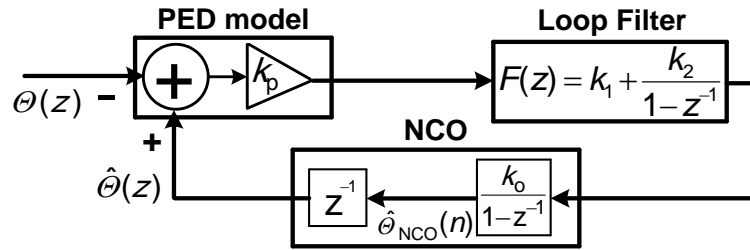


Figure 4.19: A linearised phase model of a second-order discrete domain PLL (adapted from Booyesen, 2009:23)

The transfer function $G(z)$ of the second-order discrete time domain PLL in Figure 4.18 can be derived in terms of the coefficients k_p , k_0 , k_1 and k_2 as:

$$G(z) = \frac{\hat{\theta}[z]}{\theta[z]} = \frac{-k_p k_0 k_1 z^{-2} + [k_p k_0 (k_1 + k_2)] z^{-1}}{(1 - k_p k_0 k_1) z^{-2} + [k_p k_0 (k_1 + k_2) - 2] z^{-1} + 1} \quad (4.38)$$

9) Determine the control loop coefficients k_p , k_0 , k_1 and k_2

The frequency response $H(z)$ and the transfer function $G(z)$ should be equal; therefore, by equating like coefficients from the denominator of $H(z)$ and $G(z)$, the constants k_p , k_0 , k_1 and k_2 can be expressed in terms of ζ and ω_n as:

$$k_d k_o k_1 = \frac{8\zeta \omega_n T_{sa}}{(\omega_n T_{sa})^2 + 4\zeta \omega_n T_{sa} + 4} \quad (4.39)$$

$$k_d k_o k_2 = \frac{4(\omega_n T_{sa})^2}{(\omega_n T_{sa})^2 + 4\zeta \omega_n T_{sa} + 4} \quad (4.40)$$

$$k_1 = \frac{2\zeta}{\omega_n T_{sa}} k_2 \quad (4.41)$$

These expressions can be rewritten in terms of the loop bandwidth (B_L) and sampling period (T_{sa}) by using equations (4.15) and (4.20) as:

$$k_d k_o k_1 = \frac{8\zeta \left(\frac{2}{\zeta + 1/4\zeta} \right) \frac{\gamma}{N_{symbol}}}{\left(\frac{2}{\zeta + 1/4\zeta} \right)^2 \left(\frac{\gamma}{N_{symbol}} \right)^2 + 4\zeta \left(\frac{2}{\zeta + 1/4\zeta} \right) \frac{\gamma}{N_{symbol}} + 4} \quad (4.42)$$

$$k_d k_o k_2 = \frac{4 \left(\frac{2}{\zeta + 1/4\zeta} \right)^2 \left(\frac{\gamma}{N_{symbol}} \right)^2}{\left(\frac{2}{\zeta + 1/4\zeta} \right)^2 \left(\frac{\gamma}{N_{symbol}} \right)^2 + 4\zeta \left(\frac{2}{\zeta + 1/4\zeta} \right) \frac{\gamma}{N_{symbol}} + 4} \quad (4.43)$$

$$\frac{k_1}{k_2} = \frac{4\zeta}{\left(\frac{2}{\zeta + 1/4\zeta} \right) \frac{\gamma}{N_{symbol}}} \quad (4.44)$$

The bulk of the PLL system design consists of determining the coefficients k_p , k_o , k_1 and k_2 . This requires the designer to first define the sampling period (T_{sa}), the loop bandwidth (B_L), the acquisition time (T_{acc}) and the percentage overshoot (PO). Thereafter, the damping ratio (ζ) and the natural resonant frequency (ω_n) are defined from these parameters. Then, having defined ζ , ω_n and T_{sa} , the first coefficient to be determined is generally k_p . The gain k_p of the PED is usually evaluated through simulation of the developed PED. As stated earlier, it theoretically ranges between 1.54 and 2, but practically depends on the power of the input signal (Chitode, 2009:2.48). This is the reason why it is highly recommended to keep the power of the input

signal to the demodulator as constant as possible, to within $\pm 1\text{dB}$, in order to ensure that this gain remains relatively fixed (Tatu *et al.*, 2002:2437; Booyesen, 2009:66). The gain of the PED defines the coefficients of the loop filter, which in turn controls the loop bandwidth. Finally, having defined k_p , ζ , ω_n and T_{sa} , the determination of k_0 , k_1 and k_2 can be done through an iterative process by computing the above equations. If the gain k_0 of the NCO is known, the gains k_1 and k_2 of the loop filter can be evaluated using equations (4.42) and (4.43), respectively. Alternatively, if the gain k_0 of the NCO is not defined, one can set a value for k_1 and then calculate k_2 using equation (4.44), and thereafter, evaluate k_0 using equation (4.42) or (4.43).

4.3.3.4. Measured parameters from the CRS

The performance of a loop system, such as for PLL-based carrier recovery, is evaluated through its *phase response*. The phase response is measured at the output of the NCO integrator and is used to evaluate the following performance parameters:

- system lock accuracy: how accurately does the recovery system lock to the input phase offset;
- acquisition time: the time taken by the system to lock to the input phase offset (i.e. to get a phase difference less than 5°);
- loop-bandwidth: estimated from the acquisition time T_{acq} using equation (4.16) for second-order PLL systems; and
- percentage overshoot: how much phase overshoot is present.

4.3.4. Correlator block

Figure 3.1 shows a correlator receiver with two channels (*i*- and *q*-). This is because for the QPSK signal, the number of possible symbols is $M = 4$ and the number of channels is then $N = \log_2(M) = 2$ (Edward & David: 1994:233; Geza *et al.*, 2002:713).

In the demodulation process of a digitally modulated signal, correlator blocks are used to perform the *passband-to-baseband* demodulation of the incoming symbol; that is, to regenerate the baseband bits embedded in the received passband symbol. The resulting baseband data $\hat{b}\hat{b}_n(t)$ is then used for bit decision. Note that for coherent detection, the local carrier signals $\hat{\phi}_i(t)$ and $\hat{\phi}_q(t)$ used at the receiver should be the same as the carrier signals

$\varphi_1(t) = \cos(\omega_c t)$ and $\varphi_q(t) = \sin(\omega_c t)$ used for modulation. Also note that for arbitrary MPSK modulation, $N = \log_2(M)$ carriers will be used (Pursley, 2005:294).

4.3.4.1. Design techniques

Historically, the correlator blocks for baseband regeneration were being implemented using *integrators*. This was mainly due to the system being implemented in analog. However, with the advancement of technology, a way of implementing the correlation of a baseband signal digitally is now achieved by using a *matched-filter*.

The *integrator correlator* consists of a mixer, a low-pass filter and an integrator. For QPSK signals, two correlator blocks are used for demodulation, as shown in Figure 4.20.

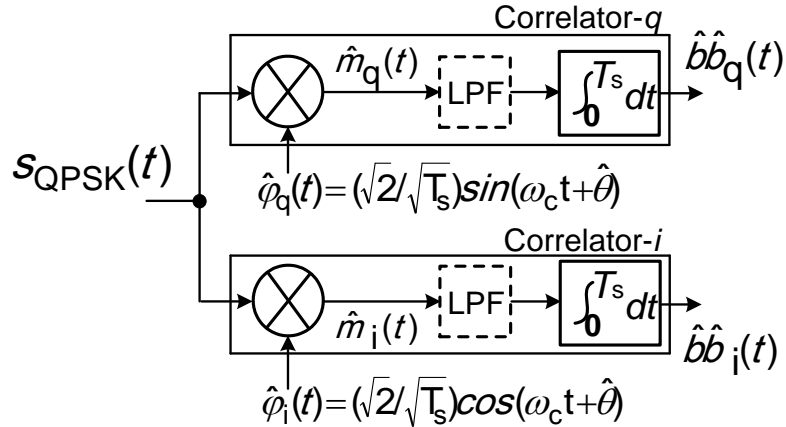


Figure 4.20: Integrator correlators (adapted from Peebles, 1987:256; Lathi & Ding, 2010:545)

The operation of the integrator correlator block was described in Section 2.5.2.1. There is no specific disadvantage to implementing this integrator technique, other than that it is mostly used for analog implementations of the correlator block. This is not an option in digital cases.

The *matched-filter correlator* is being preferred to the integrator one. More particularly, in the digital implementation of the correlator block, the matched-filter is used instead of the integrator. This technique consists of using the same filter used for pulse shaping in the transmitter; hence, the name “matched-filter”. When the shape of the incoming baseband bit matches the shape of the matched-filter’s impulse response, the output signal of the matched-filter is equivalent to the integrator’s output signal (Peebles, 1987:345; Haykin, 2014:342). Practically, the matched-filter is generally designed to also act like a low-pass filter. Consequently, the matched-filter correlator

block consists only of a mixer and a matched-filter, as illustrated in Figure 4.21 for the case of QPSK demodulation.

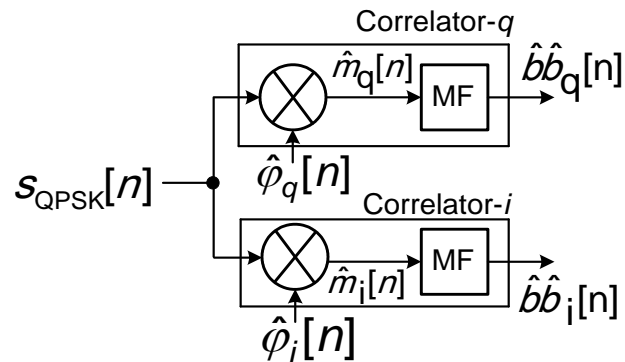


Figure 4.21: Matched-filter correlators

The operation of the matched-filter correlator block is exactly the same as the one described earlier in section 2.5.2.1 for the integrator correlator block.

Selection of matched-filter technique

One important advantage of the matched-filter technique is that beyond it being used to implement correlators digitally, it is also now being used in analog applications. Another advantage is that the coefficients of a digital matched-filter are variable and can therefore, easily be adjusted to improve the performance of the correlator block, without having to change physical components (Liu *et al.*, 2008:4). One additional advantage of the matched-filter correlator over the integrator correlator is that with the matched-filter, when using the root-raised-cosine impulse response, the output pulse of the matched-filter is a root-raised-cosine, and has minimum probability of inter-symbol-interference. Such an option is not possible with the integrator correlator. Based on these reasons, the matched-filter correlator has been selected for this work.

4.3.4.2. Design procedure

The procedure to design correlator blocks can be outlined as follows:

1. specify the design parameters, both for the system's technical design and for the digital implementation when necessary;
2. design the mixer; and
3. design the matched-filter.

1) List of the design parameters

The following parameters should be specified for the design of matched-filter correlator block:

- transmitter pulse-shaping filter and roll-off: RRCF and α
- symbol rate expected and modulation factor: R_s and M_f
- samples-per-cycle and samples-per-symbol desired: N_{cycle} and N_{symbol}
- sampling frequency: $f_{\text{sa}} = N_{\text{symbol}} \times R_s = N_{\text{cycle}} \times f_c$
- sensitivity of the ADC: $S_{\text{adc}} = D_{\text{adc}} / (2^{R_{\text{adc}}})$

2) Design of the mixer

A mixer in communication systems multiplies an input signal with a given frequency (f_{in}) with a local signal with a certain frequency (f_{lo}). Mixing two signals of frequency f_1 and f_2 results in an infinite number of frequency components, called *intermodulation products*, due to the non-linearity of the mixer (Pozar, 2005:503). These frequency components are of the form:

$$m f_1 + n f_2; \text{ where } m, n = 0, \pm 1; \pm 2; \pm 3 \dots \quad (4.45)$$

Using an appropriate filter will then allow the retrieval of only the signal with the desired frequency. In the digital implementation, a mixer is a sample-per-sample multiplication of two input signals. It is implemented using the multiplication operator and does not, therefore, need to be designed.

3) Design of the matched-filter

The matched-filter is used to maximise the E_b/N_0 ratio of the reproduced baseband signal so as to minimise the probability of error during bit decision (Edward & David: 1994:224; Glover & Grant, 1998:280). As stated earlier, the impulse response of the matched-filter is identical to the *pulse-shaping filter* used at the modulation. The discrete time domain expression of the root-raised-cosine pulse-shaping filter was given earlier in equation (3.3). The discrete time-domain expression of the matched-filter (MF) that is identical to the RRCF used at modulation is:

$$h_{\text{MF}}(n) = A_{\text{MF}} \times h_{\text{RRCF}}(n) / A_{\text{RRCF}} \quad (4.46)$$

where A_{RRCF} is the original amplitude of the RRCF, A_{MF} is the desired amplitude of the MF and the coefficients of the impulse response of the root-raised-cosine pulse-shaping filter. Again, the

amplitude A_{MF} of the MF can be given any value between S_{adc} and $V_{ref-adc-peak}$. However, it is better to keep it as low as possible, whilst still ensuring that the resulting digital coefficients conserve the filter's shape. The reason for keeping the MF coefficients as small as possible is to avoid operations of large integers resulting from the matched-filtering process. The digital coefficients of the matched-filter ($h_{MF-integers}$) are:

$$h_{MF-integers} = \text{Round}(h_{MF}/S_{adc}) \quad (4.47)$$

The impulse response $h_{MF-integers}[n]$ and frequency response $H_{MF-integers}[z]$ of the designed RRC matched-filter will be presented once all the parameters listed above have been defined.

4.3.4.3. Resulting correlator block

The correlators include the mixer and the matched-filter with an impulse response $h_{MF}[n]$, which act both as a low pass filter and an equaliser block. Figure 4.22 gives the schematic of the correlator, which is to be used in this work.

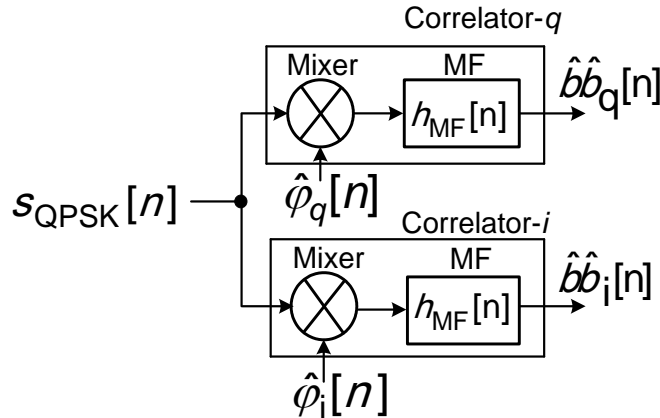


Figure 4.22: Schematic of the matched-filter correlators under design

4.3.4.4. Measured parameters of the correlator

The output signal of the correlators blocks is the reproduced baseband bits. An accumulation of these baseband bits produces the “*eye diagram*”. The quality of the baseband signal is generally observed through the opening of the eye diagram. The diagram reveals the following information:

- the level of phase and frequency match between the local carrier and the incoming carrier; and
- the level of the *SNR* of the incoming passband signal at the input of the demodulator.

If the eye is narrow, it means that the amplitudes of baseband bits are small. This can be due to high phase-mismatch between the two carriers; or in case the phase is matched, this will be due to low SNR of the incoming passband QPSK signal. The consequence in this case will be the high probability of errors during the bit decision process. Alternatively, if the eye is wide, it means that the amplitudes of baseband bits are high, resulting in a good E_b/N_o . This improves the phase and frequency match between the two carriers, and most importantly, the SNR of the input signal. The result is that there is less ambiguity during bit decision, and therefore, an improved BER performance.

4.3.5. Timing recovery system

4.3.5.1. Role of the TRS

In order to ensure the reproduction of the transmitted bit stream at the receiver with minimum error, the conditions for coherent detection techniques are:

- a) phase and frequency match between local and incoming carriers; and
- b) sampling the output of the matched-filter at its peak points.

Because the receiver has no knowledge of the incoming signal's characteristics, the first condition is generally satisfied by using a carrier recovery system, which ensures that the phase and frequency of the local carrier are locked to the phase and frequency of the received signal. Since the receiver does not have knowledge of the instant at which the peak occurs in the baseband signal, the second condition is satisfied by using a symbol timing recovery system. This system block extracts the peak timing from the baseband signal, and instructs the detection and decision block (DDB) to sample the baseband signal for bit decision (Proakis, 1995:362; Bae *et al.*, 1997:1366).

4.3.5.2. Design techniques

1) Standard TRS configuration

The TRS implementation is based on a PLL and consists of a timing error detector (TED) to generate the timing error signal, a loop-filter to track the timing error and the NCO/VCO to readjust the search timing (Torres, 2006:1). An important stage in developing a TRS is to define how it will be linked to the ADC. The standard ADC configurations can be synchronous or asynchronous to the TRS output timing. Furthermore, the ADC sampling can be done on the passband signal (*pre-matched-filter*) or on the baseband signal (*post-matched-filter*). These two

variations lead to four standard timing recovery system configurations, which are presented in Figure 4.23.

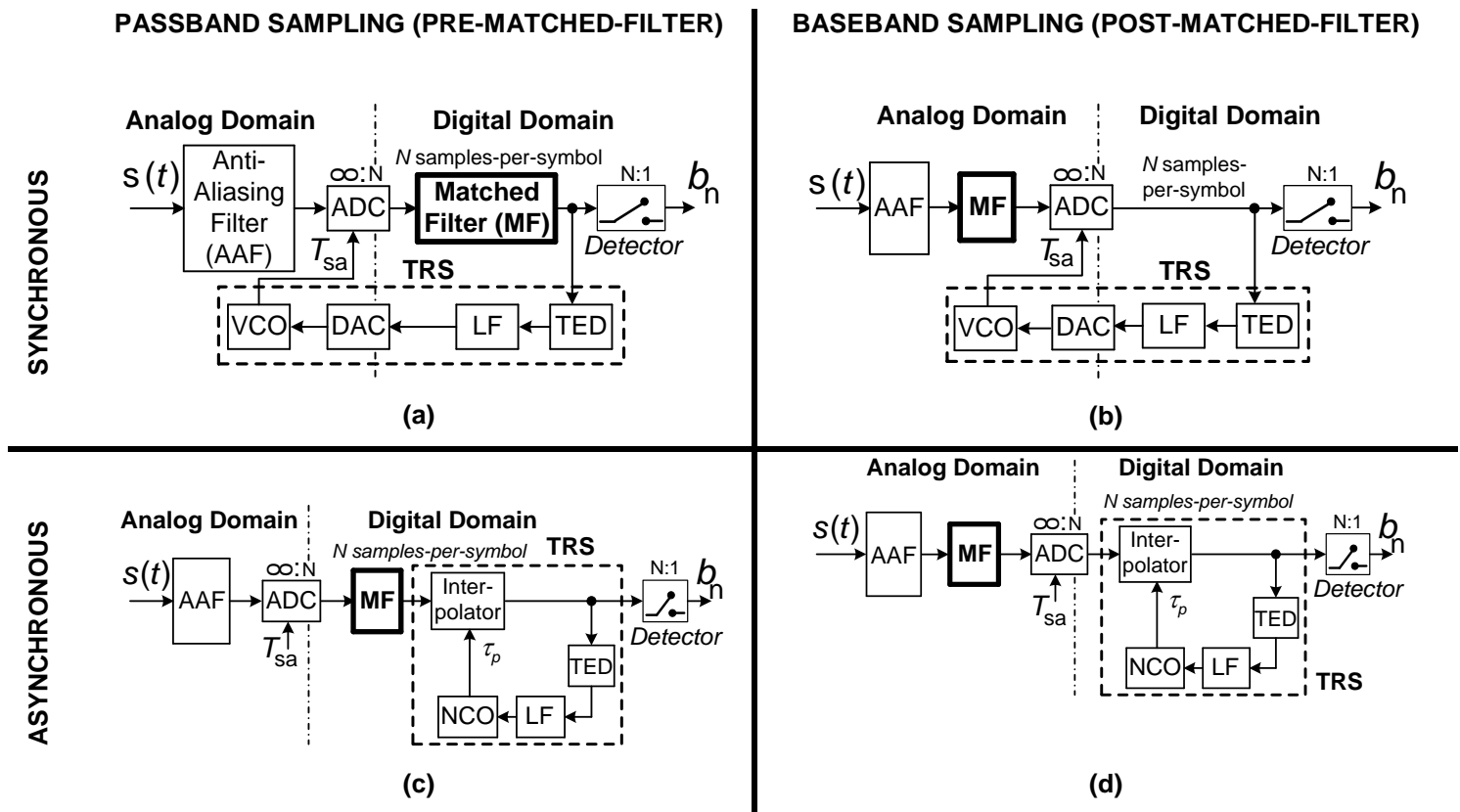


Figure 4.23: Standard TRS configurations (adapted from Booyesen, 2009:6)

When the baseband signal is sampled ((b) and (d)), the matched-filter is analog; the resulting system is a hybrid of analog and digital components. In this case, since the baseband signal at the output of the matched-filter is analog, two samples-per-received-symbol can suffice in order to perform the timing recovery. Consequently, with a fixed sampling frequency (f_{sa}), the achievable symbol rate (R_s) will be $R_s = f_{sa}/2$. However, when the passband signal is sampled ((a) and (c)), the matched filter is implemented digitally. In this case, a reasonably high number of samples-per-symbol is needed in order to implement a digital matched-filter adequately (Booyesen, 2009:7). At least 8 samples per symbol would be needed in order to adequately model the matched-filter (Booyesen, 2017). This will result in a symbol rate $R_s = f_{sa}/8$.

In the synchronous configuration, the clock of the ADC is directly controlled by the output timing of the TRS. Since the TRS is implemented digitally, this configuration will result in a hybrid implementation of the demodulator. In the asynchronous configuration, the clock of the ADC is fixed, and the output timing of the TRS from the NCO is not controlling the ADC, but rather an

interpolator that is used for peak-search adjustment from one symbol to another. This configuration is, therefore, suitable for the all-digital implementation.

2) Selection of the asynchronous passband sampling configuration

Considering the above discussion, the *asynchronous passband sampling configuration* (Figure 4.23 (c)) has been implemented in this work. First, the asynchronous configurations ((c) and (d)) are preferred to the synchronous configurations ((a) and (b)) on the basis that they are simpler to implement and can even be done all-digitially, whereas the synchronous configurations are more complex to implement due to the control of the analog ADC timing by digital components of the TRS. This eventually leads to multiple interconnections between analog and digital components. Second, the passband sampling ((a) and (c)) is preferred to the baseband sampling ((b) and (d)) on the basis that it can also achieve the desired high data rates of the baseband sampling configuration, while guaranteeing a less complex implementation (an analog matched-filter can be tricky to work with). Therefore, the rejection of configurations (a), (b) and (d), leaves (c) as the preferred configuration for this application. Also note that this configuration (c) favors the all-digital implementation; just as the configuration selected for the carrier recovery system.

The selected TRS configuration for this application is shown in Figure 4.23.

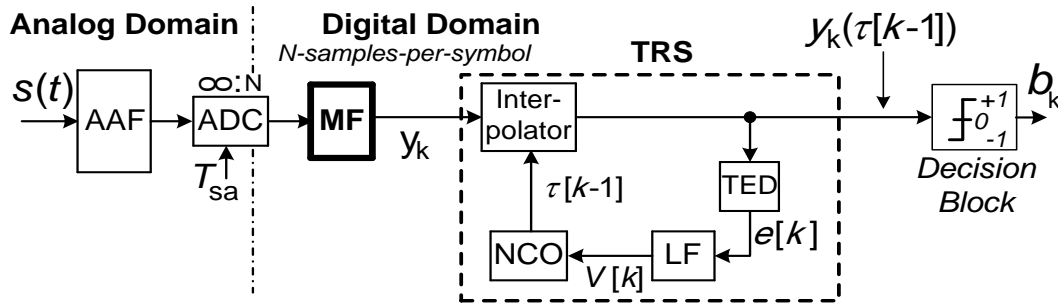


Figure 4.24: Preferred TRS implementation configuration

The operation of this timing recovery system can briefly be described as follows. The system operates on the principle of a PLL. The objective is to find the peak position (or timing) of the received baseband symbol. Initially, in order to process the first arriving symbol ($k=1$), the NCO will assume an initial arbitrary peak position or timing " $\tau(k-1) = 0$ ". Based on that timing, the interpolator will determine the amplitude values at specific points of the baseband symbol as needed by the TED. Using the baseband data obtained from the interpolator, the TED will estimate the timing error between the actual peak position and the assumed peak position, and then generate the error signal $e[k]$. The loop filter will then track that initial timing error and

output a signal $v[k]$. The NCO will re-adjust its previous timing according to the loop filter's output signal, and its new timing $\tau(k)$ will be used to process the next symbol ($k+1$). This process repeats after every symbol has been processed, until the loop system accurately locks to the actual peak timing of the received baseband symbols. The design of the subsystems (interpolator, TED, loop filter and NCO) is discussed below.

4.3.5.3. Design process of the TRS

The traditional design process of a PLL was already presented in 4.3.3.3. In this section, the same process will be followed, and possible additions specific to the timing recovery system will be presented. The design stages are, therefore, as follows:

- 1) identify the design parameters of the system, both from the control system theory side and the digital signal processing side;
- 2) define the system's natural characteristics:
 - a) define the desired PLL characteristics; that is, the desired *order*, *loop-bandwidth* (B_L), *acquisition time* (T_{acq}) and *percentage overshoot* (PO);
 - b) based on the desired system order, express the frequency response $H(s)$ of the system in terms of the damping ratio (ζ) and natural resonant frequency (ω_n);
 - c) determine the damping ratio (ζ) and the natural resonant frequency (ω_n) in terms of the loop bandwidth (B_L), acquisition time (T_{acq}), or percentage overshoot (PO);
- 3) design the TED and define the transfer function $T(z)$;
- 4) design the loop filter and define the transfer function $F(z)$;
- 5) design the NCO and define the transfer function $N(z)$;
- 6) derive the system transfer function $G(z)$, with the block coefficients k_p, k_o, k_1, k_2 ; and
- 7) determine the control loop coefficients $k_p, k_o, k_1, k_2 \dots k_n$, in terms of ζ and ω_n .

1) Design parameters

As stated earlier, the parameters for the traditional PLL design process are the system's order, percentage overshoot (PO), loop bandwidth (B_L) and acquisition time (T_{acq}).

The parameters required for the digital implementation of the timing recovery system are:

- Pulse-shaping filter used in the matched filter: RRCF and α
- Matched-filter output maximum level: $MF_{out-max-lim}$
- Symbol period: $T_s = 1/ R_s$

- Number of samples-per-symbol: N_{symbol}
- Sampling frequency: $f_{\text{sa}} = N_{\text{symbol}} \times R_s$
- Initial peak search timing: τ_i
- Sampling distance between search points: ΔT_s

2) System characteristics and frequency response

As discussed previously, a *second-order* PLL is preferred for the optimal tracking of the initial timing error. Since the symbol rate of the expected baseband signal is greater than 1 Msps, therefore, the loop bandwidth can be chosen between 1% and 2% of the symbol rate (Thomas, 2016). This means that $B_L = \mathcal{X} \times R_s$, where “ \mathcal{X} ” is between 0.01 and 0.02. Because it is a second order system, the link between the loop bandwidth and the damping ratio remains the same. The frequency response of the second order PLL is, therefore:

$$H_{\text{TRS}}(s) = \frac{2\zeta\omega_n s + \omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \quad (4.48)$$

The relationships between the damping ratio (ζ), the natural resonant frequency (ω_n), the loop bandwidth (B_L), the acquisition time (T_{acq}), and the percentage overshoot (PO), for a second order closed-loop control system are given by equations (4.20), (4.21) and (4.22).

3) TED design and definition of its transfer function $T(z)$

There are various ways of implementing a timing error detector. The existing techniques are (Keane & Hurst, nd:1; Le-Ngoc & Feher, 1980:1994; Litwin, 2001:44):

- maximum likelihood (ML);
- minimum mean square error (MMSE);
- zero forcing (ZF);
- early-late;
- Gardner (or zero-crossing); and
- Mueller and Muller.

Some of these techniques will be able to function in extremely low *SNR* applications and in applications with zero-excess bandwidth, but will have decision errors; whereas others require a good *SNR* and adequate excess bandwidth, but will have almost no decision error (Wang,

2002:21-22). Even though some sources in the literature claim that certain TED techniques are more suitable for low data rate communications, whereas others are more suitable for high data rate communications, it remains fair to say that the choice of a TED is mostly based on the *designer's comfort*, since these algorithms present similar features and performances.

In the case of coherent modulation schemes, such as MPSKs and MQAMs, the *early-late* and *zero-crossing (Gardner)* techniques are the most often-used TEDs (Stevens, 1995:28; Litwin, 2001:44; Spalvieri, 2013:7). On this basis, only these two will be discussed in this section.

In the early-late method, the algorithm searches for the peak position (or peak timing) within the received baseband symbol. Figure 4.25 illustrates the principle of the *early-late algorithm*.

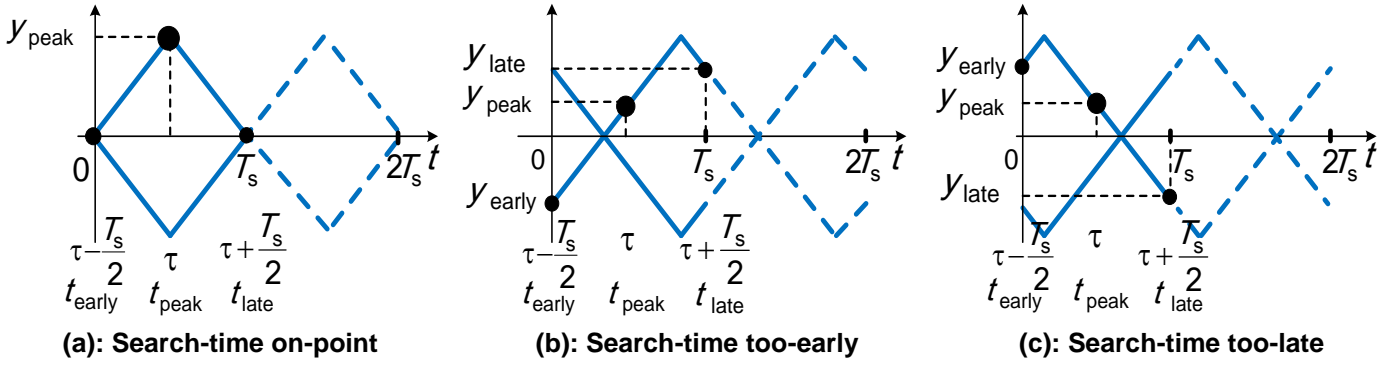


Figure 4.25: Illustration of the early-late timing error detection algorithm

Knowing that the symbol period (or symbol length) is T_s , the peak-search starts at any arbitrary timing “ τ ” within the symbol. In order to know if the peak in the k^{th} received symbol occurs at the timing “ τ ”, a sample of the baseband symbol is taken at $t_{\text{peak}} = \tau$, together with two side samples that are equidistant from τ at an interval ΔT_s . The sample taken at $t_{\text{early}} = \tau - \Delta T_s$ is called the “early sample” and its amplitude is noted $y_k(t_{\text{early}})$, or simply y_{early} ; where “ y_k ” represents the k^{th} baseband symbol received. The sample taken at $t_{\text{late}} = \tau + \Delta T_s$ is called the “late sample”, and its amplitude is noted $y_k(t_{\text{late}})$, or simply y_{late} . The sample taken at $t_{\text{peak}} = \tau$ is called the “peak search sample” and its amplitude is noted $y_k(t_{\text{peak}})$, or simply y_{peak} . Using the three samples, the search consists of comparing the early sample and the late sample with respect to the peak search sample. The comparison is mathematically implemented by computing an error signal $e[k]$:

$$e_{\text{el}}[k] = \text{sgn}(y_{\text{peak}}) \cdot \{y_{\text{late}} - y_{\text{early}}\} \quad (4.49)$$

If the peak is occurring at the estimated time $t_{\text{peak}} = \tau$ (see Figure 4.25a), the amplitudes of the two side samples will be closed to 0. As a result, the error-signal will be 0 or close to 0. Else, if

the peak search time $t_{\text{peak}} = \tau$ is too early compared to the actual pick timing (Figure 4.25b), the amplitudes of the side samples will be distinct and opposite, and the resulting error signal will be positive. The amplitude of the error signal will be proportional to the timing error between the current search timing and actual peak timing. Alternatively, if the peak search timing $t_{\text{peak}} = \tau$ is too late compared to the actual peak timing (Figure 4.25c), the amplitudes of the side samples will be distinct and opposite, and the resulting error signal will be negative. Again, the amplitude of the error signal will be proportional to the timing error between the current search timing (τ) and actual peak timing on the incoming symbol ($\hat{\tau}$).

In order to simplify the implementation of the early-late algorithm, the equidistant time ΔT_s is generally taken at $1/2T_s$ as shown on Figure 4.25. This makes it easier to understand the link between the error signal and the input timing error, and allows for the use of two samples from the current symbol and the third one from the previous symbol. As a result, instead of using three samples-per-symbol, two samples will be used per symbol, leading to an improvement in the achievable symbol rates (Makolomakwe, 2013:62). Note that, in the case of the all-digital implementation where the sampling is done in the passband, the discrete baseband symbol is represented by a high number of samples. Therefore, the symbol length will be defined by the number of samples-per-symbol (N_{symbol}), and the timing will be expressed in terms of the sample number within the symbol length. Figure 4.26 presents a block diagram of the *early-late* TED.

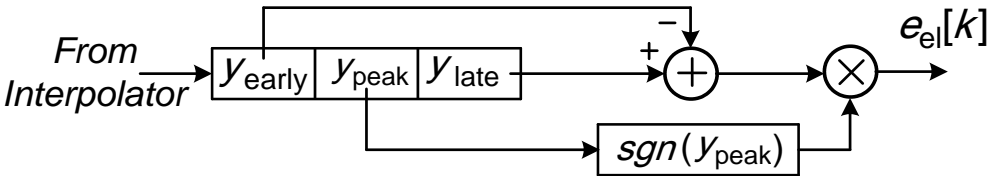


Figure 4.26: Block diagram of the early-late timing error detector

The *zero-crossing algorithm* operates in a similar manner, with the difference being that it is searching for the zero-crossing instead of the peak. Figure 4.27 illustrates the algorithm.

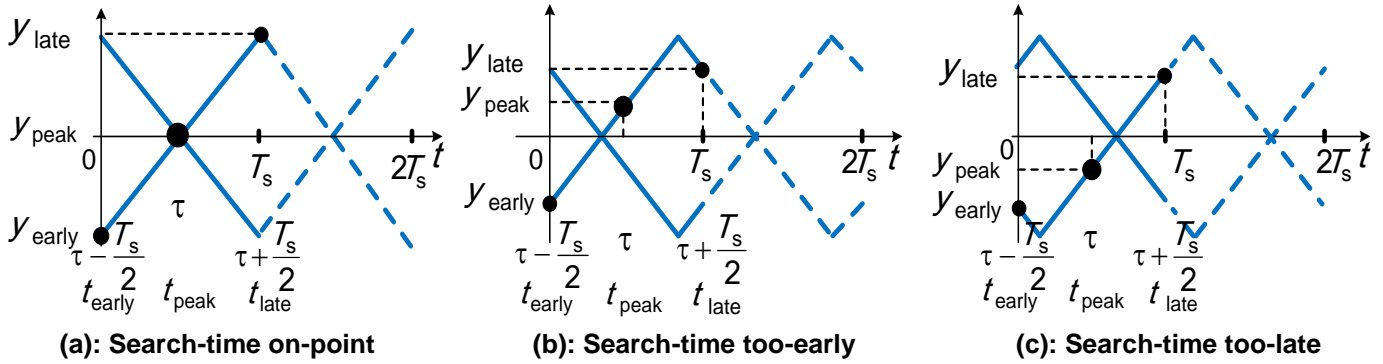


Figure 4.27: Illustration of the zero-crossing timing error detection algorithm

In order to know if a given search timing (τ) is at the zero-crossing, two side samples are taken equidistant to τ , and most preferably at an interval $\Delta T_s = 1/2 T_s$. The search process is then computed in the form of an error signal $e_{zc}[k]$:

$$e_{zc}[k] = y_{peak} \cdot \{sng(y_{late}) - sng(y_{early})\} \quad (4.50)$$

The amplitude of error signal is proportional to the timing error between the search point (τ) and the actual zero-crossing point on the symbol ($\hat{\tau}$). Figure 4.28 presents a block diagram of the zero-crossing TED.

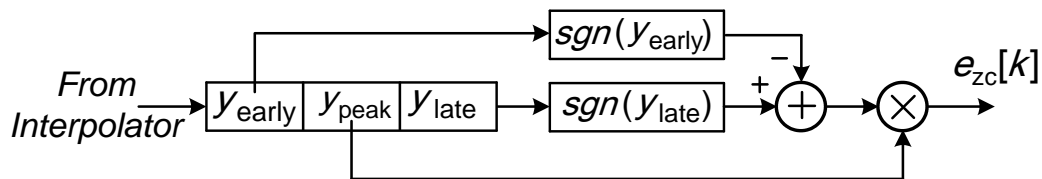


Figure 4.28: Block diagram of the zero-crossing timing error detector

The selection of the early-late algorithm for this application is simply based on the designer's comfort since it has a similar performance to the zero-crossing (Gardner). That is, they can both be used for high data rates links by using two samples-per-symbol, and they have the same accuracy and acquisition time (Booyesen 2009:36; Makolomakwe, 2013:62). The graph of the error signal's amplitude versus the input timing error is known as the "S-curve of the TED". The S-curve of the early-late TED is plotted in Figure 4.29 for three different amplitudes of the received baseband signal.

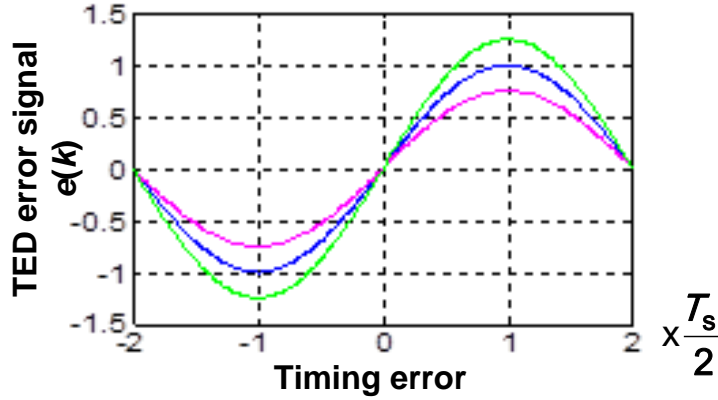


Figure 4.29: Illustration of the TED S-curve for three different received baseband signal amplitudes.

Two important remarks can be made from the S-curve. First, for small timing errors (i.e. less than 75% of $T_s/2$), the error signal is indeed approximately linearly proportional to the input timing error. The linear relationship between the input timing offset and the error signal is described in equation (4.51). Figure 4.30 illustrates the linear phase model of the TED.

$$e_{el} [k] = k_p [\tau(k) - \hat{\tau}(k)] \quad (4.51)$$

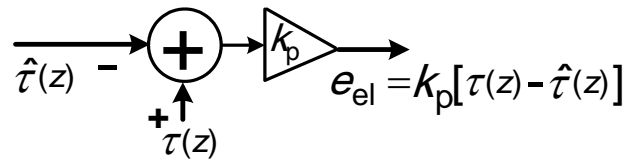


Figure 4.30: Linearised phase model of the timing error detector

In equation (4.51), k_p is the gain of the TED, and “ $\tau(k) - \hat{\tau}(k)$ ” is the input timing error. The slope of the curve, which is also the TED gain, is different for different amplitudes of the received baseband signal. Since the gain of the TED will define the coefficients of the loop filter, which in turn controls the loop bandwidth of the PLL, it is required to keep the amplitude level of the input signal to the demodulator constant. This ensures that this gain remains approximately fixed (Tatu *et al.*, 2002:2437; Booyesen, 2009:66). Note that the TED gain should be determined with alternating bits. Since the maximum possible timing error is half the symbol rate ($T_s/2$) and will lead to an error signal equal to the peak-to-peak amplitude of the baseband signal, the TED gain (k_p) can be estimated as (Booyesen, 2009:19):

$$k_p \approx \hat{b} \hat{b}_{\text{peak-to-peak}} / (T_s/2) \quad [\text{per-unit time-offset}] \quad (4.52)$$

$$k_p \approx 2 \hat{b} \hat{b}_{\text{peak-to-peak}} / N_{\text{symbol}} \quad [\text{per sample offset}] \quad (4.53)$$

The error signal for any timing error Δt between the actual peak timing and search peak timing is

$$e_{ei}(k) = k_p \Delta t = 2 \times \hat{b} \hat{b}_{\text{peak-to-peak}} \times \Delta t / (T_s).$$

4) Loop filter design and definition of its transfer function $F(z)$

The timing recovery system under design is a second-order PLL. Therefore, a first-order loop filter will be used. The first-order loop filter is discussed earlier and its discrete time-domain transfer function is given as $F(z) = k_1 + k_2 \times z / (z-1)$.

5) NCO design and definition of its transfer function $N(z)$

The NCO was also presented earlier and its fundamental blocks are described by $N(z) = k_o \times z / (z-1)$. However, when used in the timing recovery system, the NCO is complemented by an interpolation control block (IC), which determines the timings (T_{early} , T_{peak} and T_{late}) at which the interpolation should be done, based on the new NCO search timing $\tau(k)$.

6) Interpolator design

The interpolator estimates amplitude values Y_{early} , Y_{peak} and Y_{late} of the baseband signal from the matched-filter, at the defined points T_{early} , T_{peak} and T_{late} , respectively. There are many ways of doing the interpolation, including the *linear*, *neighbor* and *polynomial* methods (Yang *et al.*, nd:1-4). In certain cases where the number of samples-per-symbol is significantly high (≥ 8), the timing T_{early} , T_{peak} and T_{late} will be defined in terms of the available samples (eg. sample 1, 4 and 8). Therefore, the interpolation will just be to use the amplitude values of these samples, as there will be enough samples to accurately model the baseband symbol.

7) Resulting timing recovery system and its transfer function

Figure 4.31 presents the technical block diagram of the carrier recovery system under design.

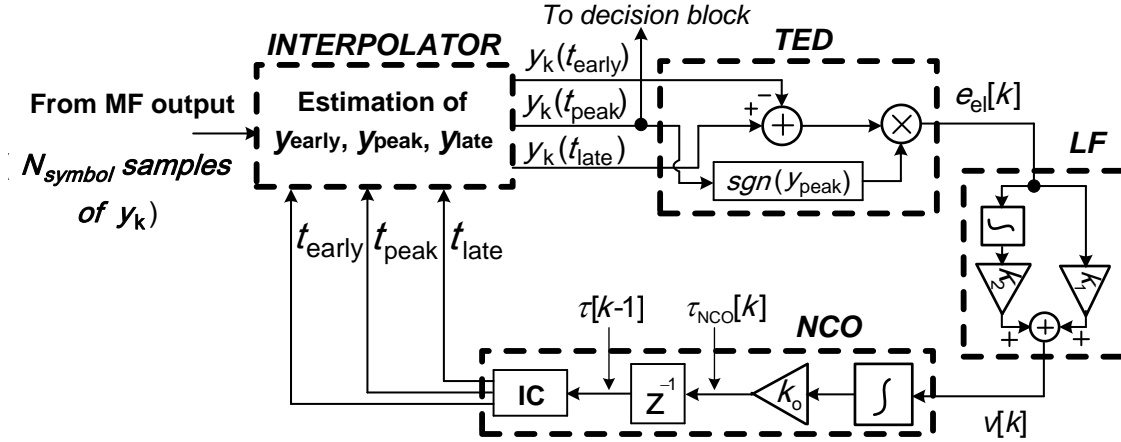


Figure 4.31: Technical block diagram of the digital timing recovery system under design

Again, using the linearised phase model of the TED, loop filter and NCO described earlier, the linearised phase model of the timing recovery system is derived as shown in Figure 4.32.

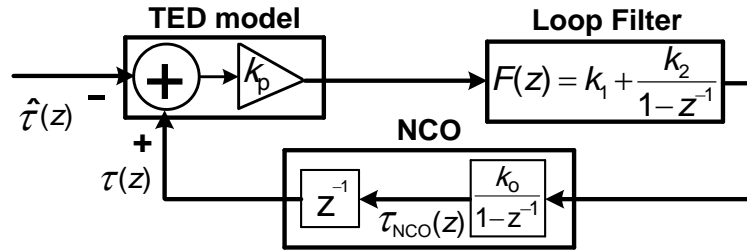


Figure 4.32: A linearised phase model of a second-order discrete time-domain PLL

The transfer function $G_{\text{TRS}}(z)$ of the discrete domain PLL presented in Figure 4.32 can be derived in terms of the coefficients k_p , k_0 , k_1 and k_2 as:

$$G_{\text{TRS}}(z) = \frac{\tau(z)}{\hat{\tau}(z)} = \frac{-k_p k_0 k_1 z^{-2} + [k_p k_0 (k_1 + k_2)] z^{-1}}{(1 - k_p k_0 k_1) z^{-2} + [k_p k_0 (k_1 + k_2) - 2] z^{-1} + 1} \quad (4.54)$$

8) Determine the coefficients k_p , k_0 , k_1 and k_2

By equating like coefficients in the denominator of $H_{\text{TRS}}(z)$ and $G_{\text{TRS}}(z)$, the expressions of the constants k_p , k_0 , k_1 and k_2 in terms of ζ and ω_n are derived as described in equations (4.39) to (4.44). After defining the sampling period (T_{sa}), the loop bandwidth (B_L), the acquisition time (T_{acq}) and/or the percentage overshoot (PO), the damping ratio (ζ) and the natural resonant frequency (ω_n) are evaluated. Thereafter, the gain k_p of the TED is determined. This gain is evaluated practically through simulation of the developed TED. Note that the TED gain should

be tested using alternating bits. Finally, the determination of k_0 , k_1 and k_2 is done through an iterative process by using equations (4.39) through (4.44).

4.3.5.4. Measured parameters from the TRS

The phase response (or peak timing response) of the timing recovery system is measured at the output of its NCO. It reveals how well the TRS has acquired (locked to) the appropriate peak timing of the received baseband data. Other parameters, such as the system *accuracy*, *acquisition time* and the *loop bandwidth*, can then be evaluated from the system's phase response. The output signal to the TRS is the symbol's sampling clock, which is used by the detection block to capture the received baseband symbol at their peaks.

4.3.6. Detection and decision block

The detection block captures the baseband symbol from the correlator block at its peak to minimise bit decision errors; hence, the timing recovery system is used to extract the peak timing from the baseband symbol. The detection and decision can be described with a switch with a decision-threshold as shown in Figure 4.33.

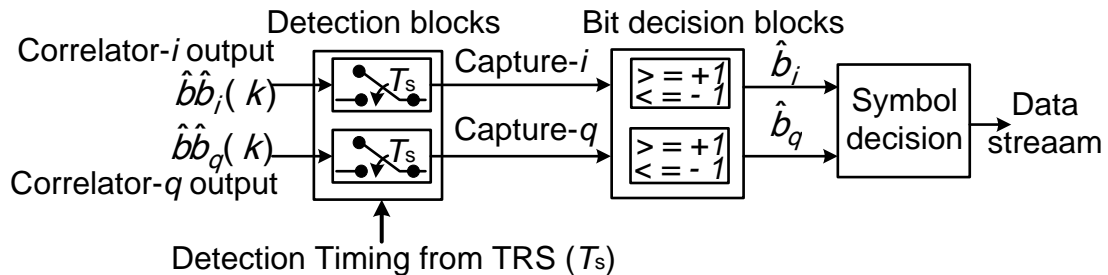


Figure 4.33: Illustration of the detection and decision block

In most instances, the detection block is incorporated in the timing recovery process when an interpolator is used to estimate the amplitude of the baseband symbols at specific points.

The role of the decision block is to make a reliable decision of the received bit, based on the sampled value of the baseband symbol (Lau *et al.*, 2001:1). Essentially, the sampled baseband amplitude is compared to a threshold, and a decision of whether the received bit is +1 or -1 is made accordingly. Generally, the threshold is set to 0; however, it is better to set its absolute value above 0, to avoid making bit decisions even when no signal is being transmitted.

The output signals from the decision and detection block are the reproduced i and q bit streams and the streams of sampled baseband values. There is no performance parameter specific to the detection and decision block. However, since this is the last block of the demodulation system, its outputs are generally used to evaluate the overall performance of the demodulation process discussed in section 4.3.7.

4.3.7. Global performance evaluation

The global parameters often used to evaluate the performance of the demodulator include:

- quality of the system, often referred to as bit-error-rate (BER) performance;
- achievable communication speed, often referred to as data rate (R_d) performance; and
- SWaP performance, also known as size, weight and power evaluation.

The quality of the developed demodulator system is commonly evaluated through measurement of the BER for a given input SNR; measurement of the BER for various input SNRs allows for generating BER versus SNR curves. However, the calculation of the BER in reality often involves numerous additional bit processing steps, such as error correction and decoding. Therefore, the most basic way of estimating the quality of the demodulator system is through the *phase constellation diagram* of the received symbols. If the constellation is too scattered, one of the subsystems is not working properly; the CRS may not be locking properly to the incoming carrier, the correlator blocks may not be reproducing the baseband bits adequately, or the TRS may not extract the correct bit position accurately. If the points in the constellation are well-grouped around the expected symbol points, then all the subsystems operate properly.

The speed performance is evaluated by determining how fast the system can process the incoming signal. This performance depends mostly on the sampling capability of the ADC, and how fast the programmable devices can execute the processes being implemented.

In terms of the size, weight and power performance, the power performance generally matters the most for digital implementations, as the space and weight of the resulting implementation are limited to the programmable hardware. The power performance is evaluated through the power consumption of the ADC and programmable devices used during the execution of their tasks.

4.4. Technical design

Using the resulting digital block diagram of each subsystem from the discussion above, the technical block diagram of the digital QPSK demodulator under design is given in Figure 4.34

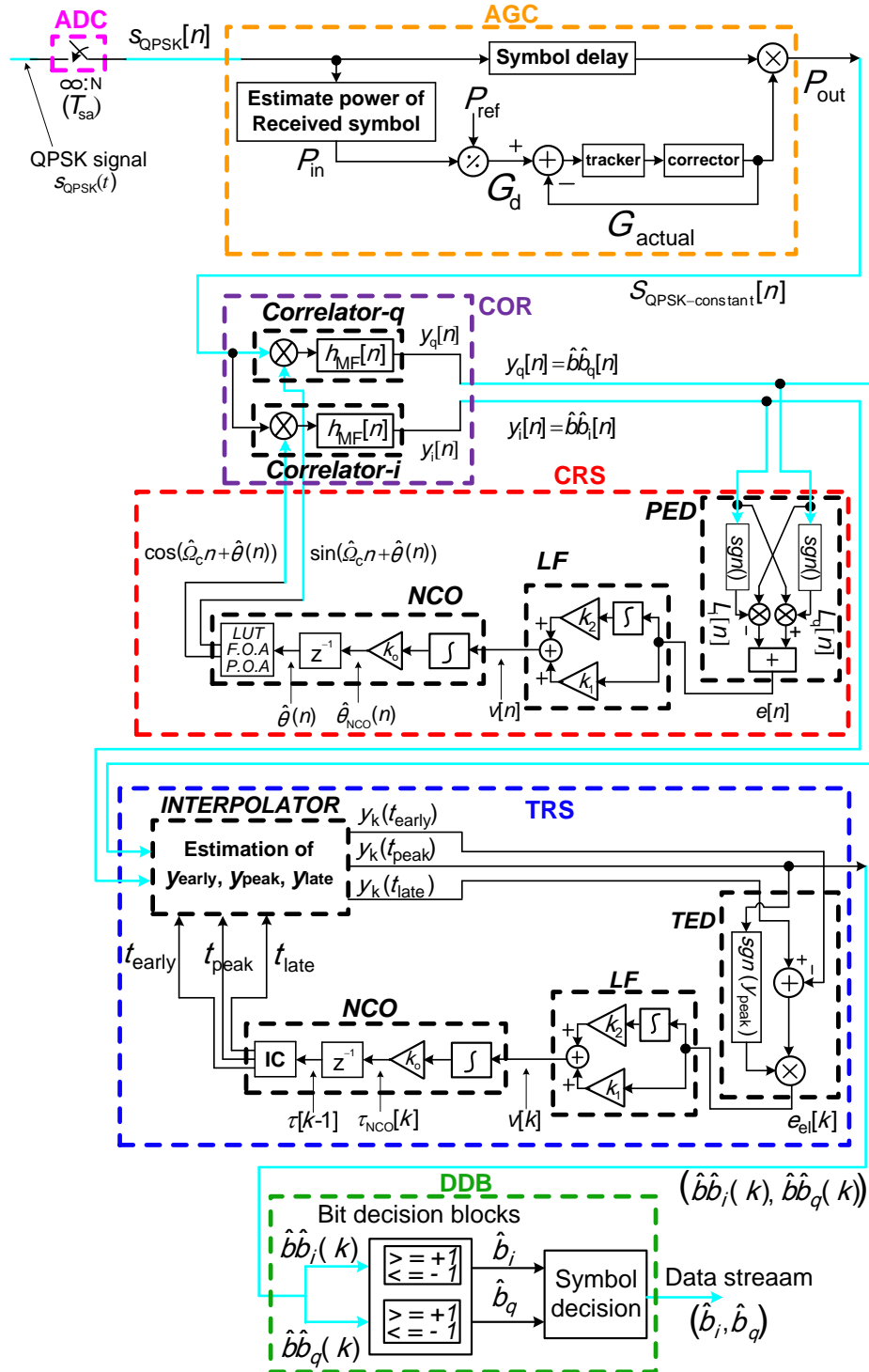


Figure 4.34: Detailed block diagram of the digital QPSK demodulator under design

This section presents the calculations needed to complete the design of each subsystem of the QPSK demodulator as outlined in the previous sections. Note that the discussion will follow the top-to-bottom order of subsystems presented in Figure 4.34.

4.4.1. Global design specifications and assumptions for the QPSK demodulator

The input signal to the QPSK demodulator is expected to have the following parameters:

- Pulse-shaping filter used: root-raised-cosine, with $\alpha = 0.36$;
- Expected data rate: Up to 10 Mbps;
- Symbol rate test: $R_s = 5$ Msps (for data rate of 10 Mbps);
- Modulation factor: $M_f = 2$ cycles-per-symbol;
- Carrier frequency expected: Up to 2×5 MHz = 10 MHz (for data rate of 10 Mbps).

4.4.2. ADC selection

The selection of an appropriate ADC is the first stage of the technical design process, since the desired system is to be implemented fully digitally.

4.4.2.1. Defining the desired number of samples-per-cycle

The relationship between the necessary sampling frequency (f_{sa}), the desired number of samples-per-cycle (N_{cycle}), and the expected symbol rate (R_s) has been given earlier as: $f_{sa} = f_c \cdot N_{cycle} = M_f \cdot N_{cycle} \cdot R_s$. It has been stated earlier that given a maximum sampling frequency (f_{sa-max}), in order to maximise the achievable symbol rates, it is best to keep M_f and N_{cycle} at their respective possible minima. On this basis, M_f was selected as 2 (see section 3.2.4.1). Based on the discussion in Section 4.3.3.2 (4), the acceptable minimum for N_{cycle} in the case of an all-digital implementation is 8 samples-per-cycle, for an accurate modeling of a cosine and sine cycles. Therefore, for this implementation, N_{cycle} is chosen as 8. The resulting number of samples-per-symbol is $N_{symbol} = M_f \cdot N_{cycle} = 2 \times 8 = 16$ samples-per-symbol. This is also the discrete time width of the expected pulse (or symbol).

Note that since the matched-filter is designed to match the incoming baseband pulse, its width should be equal to the width of the incoming signal. Therefore, since the demodulator system is implemented digitally in this application, the matched-filter will be designed to match an incoming pulse with a discrete time width of $N_{symbol} = 16$ samples-per-symbol. Therefore, since N_{cycle} is

fixed to 8 in the demodulator, $M_f=2$ must be used at modulation, no matter the data rate at which one is aiming to transmit.

4.4.2.2. Defining the desired sampling frequency

The expected carrier frequency is 10 MHz for a symbol rate of 5 Msps (for 10 Mbps). With the number of samples-per-symbol defined as $N_{\text{symbol}} = 16$, the sampling frequency should be $f_{\text{sa}} = N_{\text{symbol}} \times R_s = 16 \times 5 = 80$ MSps. In order to achieve higher data rates than 10 Mbps, the selected ADC should be able to sample at frequencies higher than 80 MSps.

4.4.2.3. Input signal amplitudes consideration

It was said earlier in Section 4.3.1.4 that input signal amplitudes should be fall within the ADC dynamic range, which practically is between $20 \times Q_{n\text{-peak}}$ and $V_{p\text{-adc-ref}}$. There are three ways to select an ADC considering the input signal amplitudes, which include to:

1. estimate the minimum and maximum expected signal amplitudes and SNRs, then select an ADC which accommodates these amplitudes and SNRs;
2. use a very high resolution ADC (eg: 16 bit ADC), which provides a very large dynamic range, capable of accommodating large signal variations;
3. choose an arbitrary ADC that is accessible off-the-shelf, with an average resolution (eg: 12 bit ADC), and ensure that the received signal amplitudes fall within the voltage dynamic range of selected ADC.

For this application, since the estimation of received signal amplitude is complex and very high resolution ADCs are both costly and not available off-the-shelf, the third option listed above has been preferred. Therefore, off-the-shelf ADCs with an average resolution should be looked at.

4.4.2.4. Selected ADC

Based on the above discussion, the ADC12DJ3200 from Texas Instruments (Texas Instruments Incorporated, 2017) has been identified and can be used for this application. It can sample at rates up to 3200 MSps with a low power consumption of 200 mW at 100 MSps. It has a bit resolution of $R_{\text{adc}} = 12$, with a possible reference setup of $D_{\text{adc}} = V_{\text{ref-adc-peak-peak}} = 1$ Vpp, which gives a quantisation noise $Q_{n\text{-rms}} = 0.07$ mV and a maximum signal-to-noise ratio $SNR_{\text{adc-max}} = 74$ dB. From the discussion in Section 4.3.1.4, the acceptable range of input signal amplitudes of the ADC is between $20 \times Q_{n\text{-peak}} = 20 \times (0.07 \times 1.42) = 2$ mV and $V_{p\text{-adc-ref}} = 500$ mV.

4.4.3. Automatic gain control

With reference to the QPSK demodulator configuration shown in Figure 4.34, the AGC delivers a passband QPSK signal with a constant amplitude to the rest of the demodulator, irrespective of the amplitude of the QPSK signal received at the input port of the demodulator. Since the estimation of the input signal amplitudes has not been done due to its complexity, the AGC has been designed with an arbitrary dynamic range as discussed in first paragraph of Section 3.3. Since the ADC dictates the acceptable range of input signal amplitudes to the demodulator, the AGC has, therefore, been designed using the selected ADC input voltage range, which is between 2mV and 500 mV. This corresponds to a power level of 39.2 nW (−44 dBm) to 2.5 mW (4 dBm) respectively in a 50 Ω system. By setting the constant output signal amplitude to be equal to the maximum acceptable input amplitude, that is, $V_{\text{ref}} = 500$ mV (or $P_{\text{ref}} = 2.5$ mW in 50 Ω), the minimum and maximum AGC power gains are $G_{\text{min}} = 10\log(P_{\text{ref}}/P_{\text{in-max}}) = 0$ dB and $G_{\text{max}} = 10\log(P_{\text{ref}}/P_{\text{in-min}}) = 48$ dB respectively.

Note that in this application, the expected signal power does not have a constant level throughout the overpass. Subsequently, a standard second-order closed loop PLL configuration will not be able to track and adjust the power variation adequately (Nicoloso, 1997:15). Therefore, in order to adjust the power accordingly for every received symbol, an instantaneous gain estimation process described in equation (4.6) will be used for every received symbol. Since the process is implemented digitally, the received symbol's amplitude will be used to estimate the power of the received symbol. Subsequently, having an estimate of the received symbol's power, the required AGC gain is calculated accordingly as prescribed above, and the corresponding output symbol is generated at the desired power level.

4.4.4. Correlator block

The mixer of the correlator block is implemented digitally using a multiplication operator. The discrete-time impulse response of the matched-filter was presented in equations (4.46) and (3.3). The expected symbol rate is $R_s = 0.5R_d = 5$ Msps for a 10 Mbps data rate. In the ADC selection above, the number of samples-per-symbol is set to be $N_{\text{symbol}} = 16$, which leads to a sampling frequency $f_{\text{sa}} = N_{\text{symbol}} \times R_s = 16 R_s$. The sensitivity for a 12-bit ADC is $S_{\text{adc}} = 0.244$ mV/bit. The roll-off factor of the RRCF is set at $\alpha = 0.36$. The length of the filter is chosen to be $L_{\text{RRCF}} = N_{\text{symbol}} = 16$ taps, and its amplitude is chosen to be $A_{\text{MF}} = 30S_{\text{adc}} = 7.32$ mV. Substituting all these values into equation (4.46) yields:

$$h_{MF}(n) = \frac{0.0061}{1 - (0.18 \times n)^2} \left\{ \frac{8}{n\pi} \sin[0.08 \times \pi \times n] + \frac{1.44}{\pi} \cos[0.08 \times \pi \times n] \right\}; -8 \leq n \leq 7 \quad (4.55)$$

The obtained floating-point coefficients are converted into integers using equation (4.47) and the resulting coefficients are shown in Figure 4.35. *Note that the graph is plotted with a matched-filter length of 96 points, simply to provide a clearer illustration of the impulse response.*

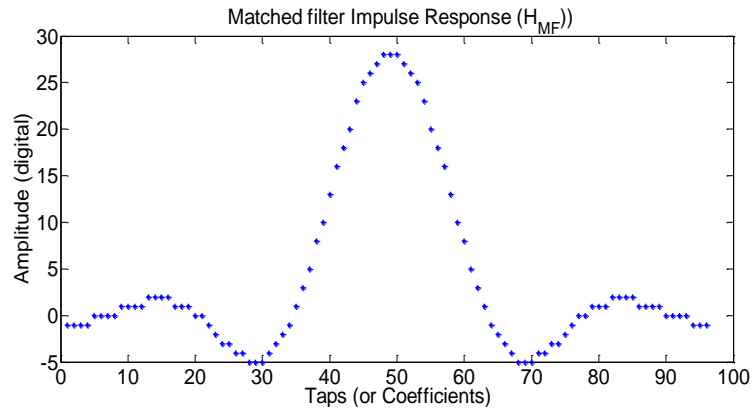


Figure 4.35: Impulse response of the designed matched-filter

4.4.5. Carrier recovery system

For the traditional PLL design process, the closed-loop system response type has been chosen to be a second-order in order to track both the phase offset and the steady-state frequency offset. The design of a second-order PLL was described in Section 4.3.3.3 and summarised in equations (4.39) to (4.44), where $\zeta = 1/\sqrt{2}$ for an optimal acquisition time.

The determination of the PED gain (k_p) is a major step towards completing the design of the carrier recovery system.. This requires the implementation of the correlator blocks, as well as the PED algorithm that was previously selected. The implementation of these blocks is digital and as discussed in Sections 4.5.2 and 4.5.3. The PED *S-curve* obtained is shown in Figure 4.36. The error is evaluated for the main phase cycle (-45° to $+45^\circ$) only, since it repeats with a 90° period. The data points in the graph are 5° apart. The PED gain has been estimated from this graph to be $k_p \approx 2$. Note that, should the PED gain not be within the boundaries specified in Figure 4.14, the baseband amplitudes can be scaled up or down until the desired gain is within the range. In this case, the baseband amplitudes have been scaled down to a fixed digital value of 40, in order to obtain the results in Figure 4.35.

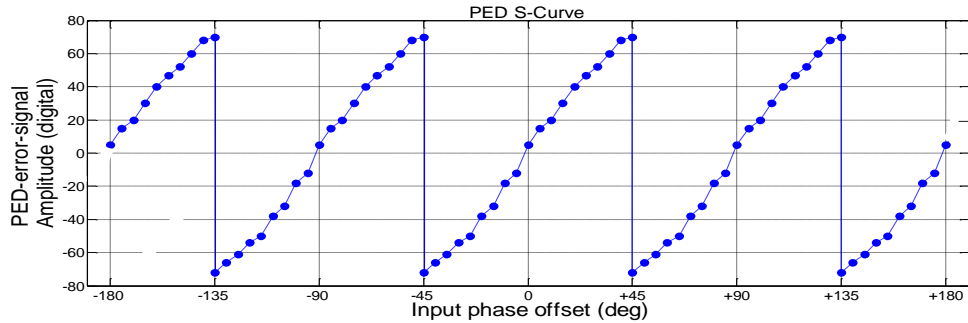


Figure 4.36: S-curve of the designed PED

Since the NCO gain k_0 is not known initially, the coefficient k_1 of the loop-filter is set to $k_1 = 1$; using equation (4.44), the coefficient k_2 is evaluated as $k_2 \approx 1/1024$. In order to make the output of the loop-filter integers, without changing the gain of the system, the coefficients are multiplied by 1024 and a scale-down block with a factor of $1/1024$ is inserted at the output of the loop-filter.

Subsequently, in equation (4.42), the gain of the NCO is evaluated as $k_0 = 1/512$. The expression of the standard cosine cycle for the look-up table is presented in equation (4.36). In order to convert the floating point coefficients into integer coefficients, the amplitude of the look-up-table is set to $A_{LUT} = 60S_{adc} = 14.64$ mV, based on the recommendation discussed in the NCO design section. Floating point coefficients are converted to integer coefficients using equation (4.37). Figure 4.37 shows a cycle of the standard cosine function and the generated LUT cycle with integer coefficients for use in the NCO.

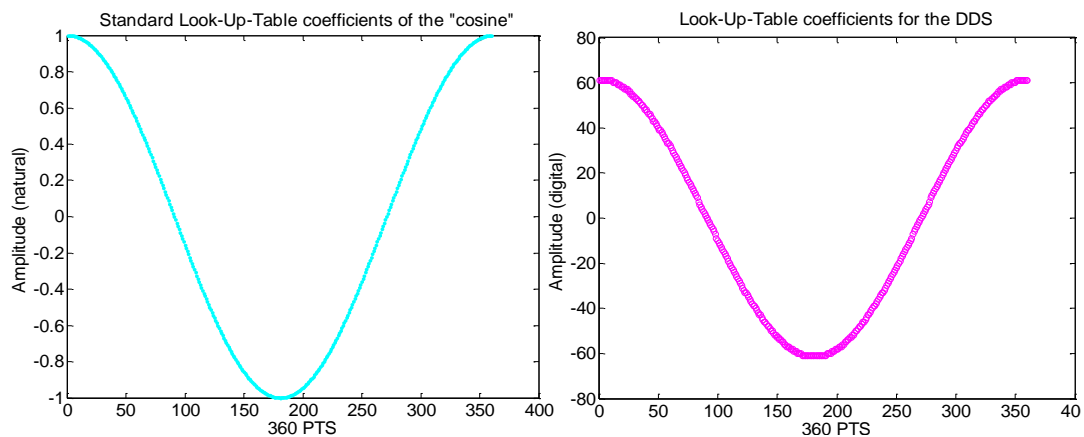


Figure 4.37: The standard (left) and the NCO (right) LUT coefficients

4.4.6. Timing recovery system

Following the traditional PLL design process, the closed-loop timing recovery system response type has been chosen to be of a second-order, and the system loop bandwidth

$B_L = \mathcal{X} R_s = 1\% R_s$. The acquisition time linked to the loop bandwidth is $T_{acq} = 1.3/B_L$ for the second-order system. Likewise, for the digital implementation process of the selected TRS topology, the following important design parameters have been defined:

- Digital amplitude of the baseband signal: $MF_{out-max-lim} = 40$
- Initial search position: $\tau_i = 0$;
- Sampling distance between the 3 search points: $\Delta T_s = \frac{1}{2} T_s = 8$ sample spacing

Other design parameters, such as the RRCF roll-off factor α , the symbol duration $T_s = 1/R_s$, the number of samples-per-symbol N_{symbol} and the sampling frequency $f_{sa} = N_{symbol} \times R_s$ have already been defined in both the design specifications and ADC selection sections.

The design of a second-order PLL-based timing recovery system was described in Section 4.3.3.3 and summarised in equations (4.39) to (4.44). Completing the TRS design consists of determining the gain k_p of the TED, the coefficients k_1 and k_2 of the loop filter and the gain k_0 of the NCO, as discussed earlier.

Using the design parameters specified above, Matlab code has been developed to execute the algorithms of the interpolation control, the interpolation and the early-late TED summarised in Section 4.5.4. The error signal is measured for different input timing offsets, and the “S-curve” of the TED is generated as shown in Figure 4.38. From this plot, the gain k_p of the TED can be evaluated as $k_p \approx 10$, which is equal to the theoretical one defined by $k_p = MF_{pp} / (N_{symbol} / 2) = (40+40)/(16/2) = 10$. Note that this result is based on the fact that the baseband signal amplitude is fixed to a digital value of 40.

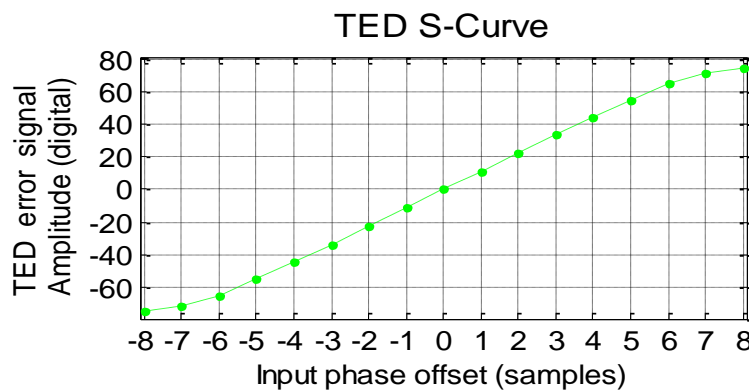


Figure 4.38: S-curve of the designed early-late TED

Since the NCO gain k_0 is initially not known, the coefficient k_1 of the loop-filter is set to $k_1 = 1$, and the coefficient k_2 is derived as $k_2 \approx 1/1024$ using equation (4.44). With these floating point

numbers, the loop filter's output signal will also be floating point numbers. The loop filter will therefore not be integratable, since most programmable hardware does not operate with floating points, but only with integers. Therefore, in order to ensure that the output of the loop-filter is integer numbers, and without changing the gain of the system, the coefficients k_1 and k_2 are multiplied by 1024; a scale-down block with a factor of 1/1024 is then placed at the output of the loop filter.

Substituting k_1 in equation (4.42), the gain of the NCO is evaluated as $k_0 \approx 1/1024$. The output of the NCO is the supposed peak timing (τ) of the processed symbol, in the form of a sample number between 1 (the first sample in the baseband symbol) and N_{symbol} (the last sample).

4.4.7. Detection and decision block

There is no specific technical design aspect of the detection and decision block. The detection process has already been implemented in the TRS through the interpolation function. For the decision block, the decision threshold has been set to 5, based on the observation of the worst-case scenario of the baseband signal amplitude.

4.5. Algorithms for digital implementation of subsystems

This section presents the algorithms developed to implement the processes of each subsystem of the QPSK demodulator discussed above. The discussion here follows the same order.

4.5.1. Automatic gain control

The flowchart in Figure 4.39 gives the algorithm that describes the digital implementation of the AGC process.

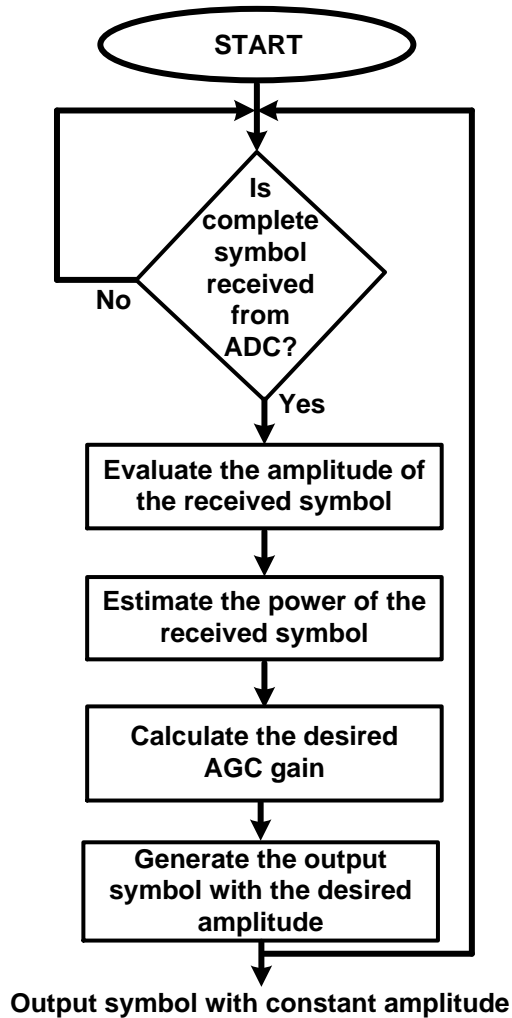


Figure 4.39: Algorithm for the digital implementation of the AGC process

4.5.2. Correlator blocks

The matched-filter correlator block designed in Sections 4.3.4.2 and 4.4.4 consists simply of a mixer and a matched-filter. In the digital implementation, a mixer is implemented using a multiplication operator with a gain of 1. The matched-filter, however, is a linear time-invariant (LTI) system like many other FIR filters. The main advantage of realising the matched-filter digitally is the ease of control of the coefficients for improved phase and frequency responses (Thede, 1996:199; Katiar & Anju, 2012:1). The matched-filter, being an LTI system, can therefore be modeled as a convolution of the input signal $x[n]$ with the filter's impulse response $h_{MF}[n]$, which mathematically can be expressed as (Ahamed & Lawrence, 1997:254; Kapadia, 2012:7):

$$y_{MF}(n) = x_{in}(n) \otimes h_{MF}(n) = \sum_{k=1}^N x_{in}[n - k + 1] \times h_{MF}[k] \quad (4.56)$$

where N is the length of the matched-filter. The convolution process is generally implemented in software with either the *pipeline approach*, or *multiple cycle approach* (Schlichter, 1999:5). Figure 4.40 illustrates the two implementations approaches.

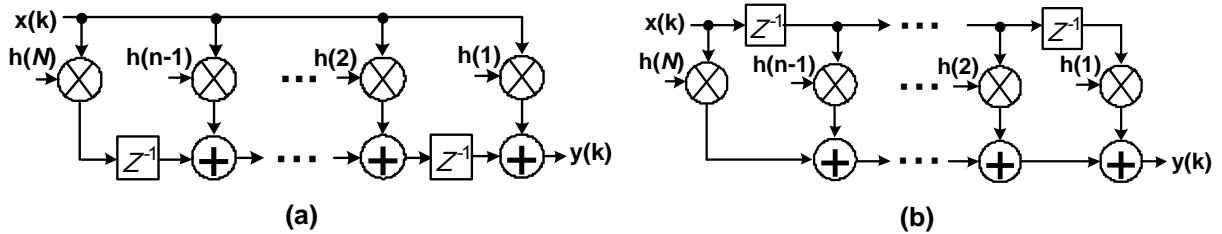


Figure 4.40: Illustration of the convolution process: (a) pipeline implementation (adapted from Ahamed & Lawrence, 1997:256) and (b) multi-cycle implementation (adapted from Mitra, 2006:433)

The pipeline algorithm has the advantage of being time-efficient, but requires a significant amount of memory; the multi-cycle algorithm has the advantage of being memory-efficient, but requires a considerable computation time (Blanchard, 2017). The pipeline technique also reduces system noise because operations are done progressively and not in bulk, as in the case of the multi-cycle technique. Finally, due to its progressive operation process, the pipeline programming technique would result in less internal heating of the system, compared to the multi-cycle technique. This makes the system less vulnerable to impairment, such as from radiation (Dusseau, 2017).

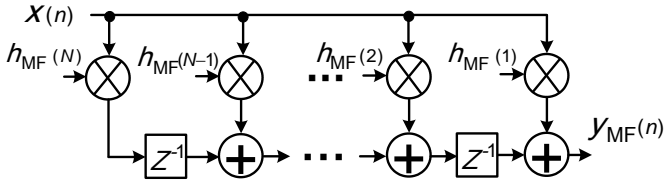
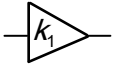
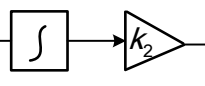
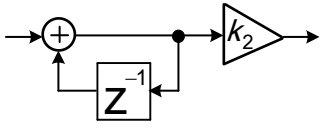
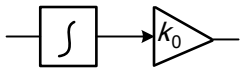
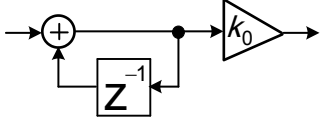
Therefore, for this application which requires high-speed, low noise and robustness against radiation, the pipeline algorithm is used to implement the filtering process.

4.5.3. Carrier recovery system

Table 4.2 summarises the algorithms used for the digital implementation of blocks of the CRS.

Table 4.2: Summary of the algorithms to digitally implement each block of the CRS

	Block	Diagram Schematic	Algorithm
Correlator	mixer	\otimes	<i>multiplication operator</i> “x”
	matched-filter	$h_{MF}[n]$	pipeline configuration of a convolution process

			
PED	signum	$\text{sgn}()$	<i>signum-func</i>
	multipliers	\otimes	<i>multiplication operator</i>
	summing point	\oplus	<i>addition operator</i>
Loop Filter	proportional		<i>multiplication by a constant</i> k_1
	integrator		
	summing point	\oplus	<i>addition operator</i>
NCO	integrator		
	symbol unit delay	Z^{-1} symbol	<i>wait-function</i> "wait for 1 symbol duration"
	local carrier generator	$h_{\text{LUT}}[n]$ P.O.A F.O.A	the <i>local carrier generation algorithm</i> is presented in Figure 4.41

Note that an integrator is naturally implemented using the pipeline technique as shown by the algorithm's implementation flow in Table 4.2. The merit of the pipeline algorithm over the multi-cycle one was discussed in Section 4.5.2. The local carrier generator implement three internal processes, namely the local carrier phase adjustment (POA), frequency offset compensation (FOA) and the generation of the cosine and sine cycles. Since the LUT has 360 points, and in order to generate the desired cycle of cosine and sine carriers, N_{cycle} samples are taken from the 360 points of the LUT. These N_{cycle} points are spaced at intervals calculated as:

$$LUT_{\text{spacing}} = 360/N_{\text{cycle}} \quad (4.57)$$

Since the desired number of samples-per-cycle is $N_{\text{cycle}} = 8$, the spacing will be $LUT_{\text{spacing}} = 45$ points. The *phase adjustment* is such that the new NCO phase ($\hat{\theta}[n]$) represents the address (between 1 and 360) of the first sample, which is to be taken from the LUT. The other samples are then taken consecutively at the calculated spacing interval to form complete *cosine* and *sine* cycles with N_{cycle} points. The *frequency offset compensation* also takes place during the carrier cycles' generation. This rather complex process is implemented by estimating the phase variation beyond the expected lock time, which is due to the steady-state frequency offset. Figure 4.41 shows the local carrier generation process.

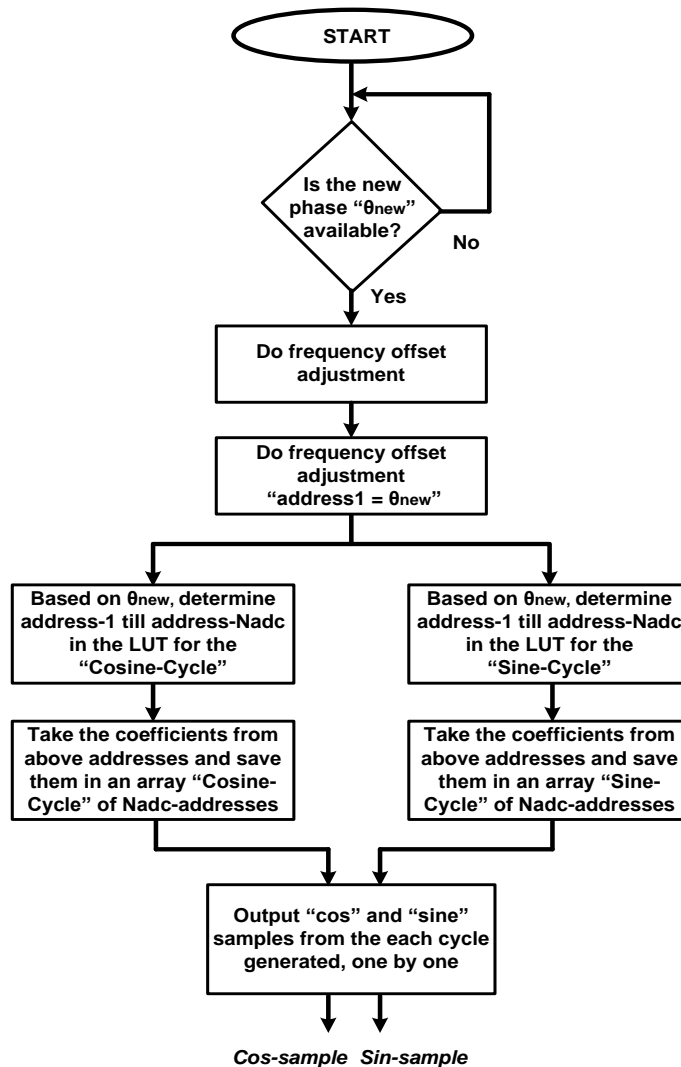


Figure 4.41: Flowchart of the local carriers generation process

4.5.4. Timing recovery system

The timing recovery system designed consists of an interpolator block, a timing error detection block, a loop filter and the NCO subsystem. From Figure 4.31 and Figure 4.18, it appears that the subsystems for the CRS and TRS are using the same components, except the interpolator and the interpolator control. One can, therefore, refer to Table 4.2 for the algorithms developed to digitally implement components of the TED, loop-filter and the NCO.

The interpolation control (IC) and the interpolation processes are executed as described by the flowcharts in Figure 4.42.

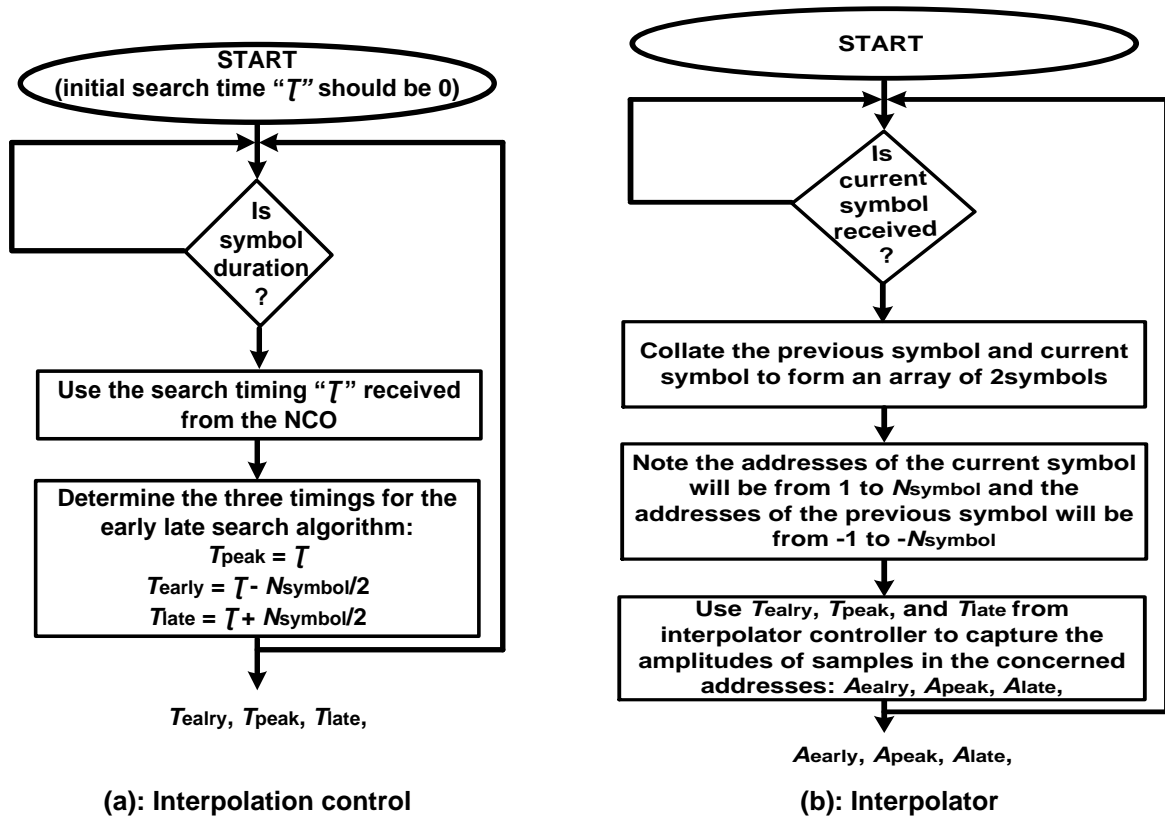


Figure 4.42: Flow chart of (a) the interpolation control and (b) the interpolation process

4.5.5. Detection and decision block

There is no specific algorithm needed to implement the DDB, because the capture of the reproduced baseband signals is achieved by the interpolator. The bit decision is implemented by using a comparator operator that is available in most software development libraries.

4.6. Chapter summary

This chapter covered the design theory of each subsystem of the QPSK demodulator, which resulted in the selection of specific techniques to implement each subsystem of the demodulator.

A single-loop PLL-based configuration was selected for the CRS. A second-order CRS was preferred in order to track both the phase and frequency offsets. A digital Costas loop technique was selected for the PED of CRS, followed by a first-order loop filter and a numerically controlled oscillator, which generates the desired local carrier signals. A matched-filter correlator was selected for this implementation, which includes a root-raised-cosine matched-filter.

A second-order single-loop PLL configuration was also selected to implement the TRS. The asynchronous-passband sampling topology was selected for the TRS. The early-late technique was selected to implement the timing error detector block of the TRS.

The chapter also presented the calculations and assumptions needed to complete the technical design of each subsystem, based on the techniques selected and the specifications set.

Finally, the algorithms necessary to digitally implement each subsystem have been generated in this chapter. Most of the algorithms are based on the pipeline implementation in order to minimise the processing time as well as to limit vulnerability to radiation. The algorithms and technical designs are used to implement the digital QPSK demodulator on a programmable platform. This is described in the following chapters.

CHAPTER 5: MATLAB IMPLEMENTATION OF THE QPSK DEMODULATOR DESIGN

5.1. Introduction

This chapter presents the implementation of the designed digital QPSK demodulator in Matlab to validate the design, before its implementation on a programmable hardware platform. Note should be taken that the Matlab implementation only evaluates the system performance in terms of the operations and the resulting outputs. It does not evaluate its *physical* performance, such as minimum execution time, maximum power usage and resource usage (memory, gates, multipliers, etc.); these can only be evaluated on a hardware development platform as presented in Chapter 6.

5.2. Sampled QPSK signal

A model of the QPSK signal received at the input port of the QPSK demodulator was generated and presented in Section 3.3. The model has been generated with an arbitrary assumption that the additive white Gaussian noise power at the input port of the demodulator is -60 dBm. Subsequently, based on this assumption, the QPSK signal can be generated for any value of the signal-to-noise ratio, from 0 dB up to the maximal *SNR* value acceptable by the selected ADC, and for different bit rates. However, due to space constraints in this document, only the simulation performed with 12 dB SNR and 10 Mbps of bit rate, are presented. For a *SNR* of 12 dB and with an assumed AWGN power level of -60 dBm at the input port of the demodulator, the total power of the noisy QPSK signal received at the input of the demodulator is about -45 dBm, which corresponds to a total amplitude of 1.7674 mV in a 50Ω system, as illustrated in Figure 3.11. Similarly, for 10 Mbps, the double-side-band bandwidth of the transmitted QPSK signal is about 6.8 MHz, when the root-raised-cosine pulse shaping filter is used, with a roll-factor $\alpha = 0.36$, as illustrated in Figure 3.10 (c). The choice of the pulse-shaping filter and the value of α have already been motivated in Section 3.2.3.3.

Using the 12-bit ADC selected earlier in section 4.4.2.4, with a dynamic range of 1 Vpp, the ADC sensitivity is $S_{\text{adc}} = 0.244$ mV/bit. Subsequently, the digital amplitude of the above discussed QPSK signal is $1.7674/0.244 = 8$ as shown in Figure 5.1. Also, an important design assumption for the all-digital implementation made in section 4.4.2.1 was that the number of samples-per-cycle of the received QPSK carrier signals should be $N_{\text{cycle}} = 8$, and the number of carrier-cycles-

per-transmitted-symbol should be $M_{\text{factor}} = 2$, at the modulation. Subsequently, the number of samples-per-symbol is $N_{\text{symbol}} = M_{\text{factor}} \cdot N_{\text{cycle}} = 16$. Therefore, the required sampling frequency for the above discussed QPSK signal is $f_{\text{sa}} = N_{\text{symbol}} \cdot R_s = 16 \cdot (10/2) = 80 \text{ MHz}$ (for 10 Mbps). The sampled QPSK signal is presented in Figure 5.1.

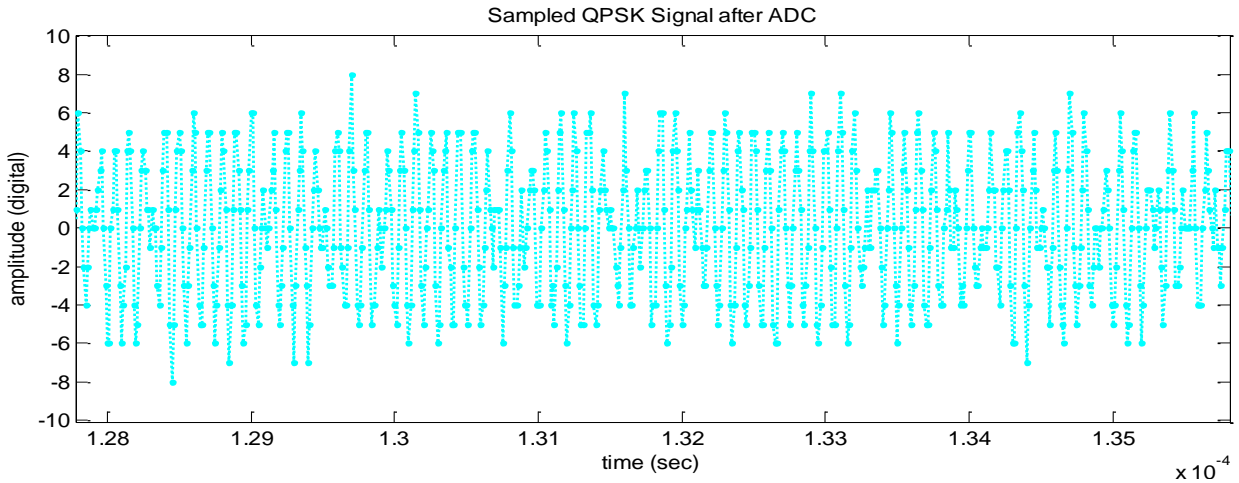
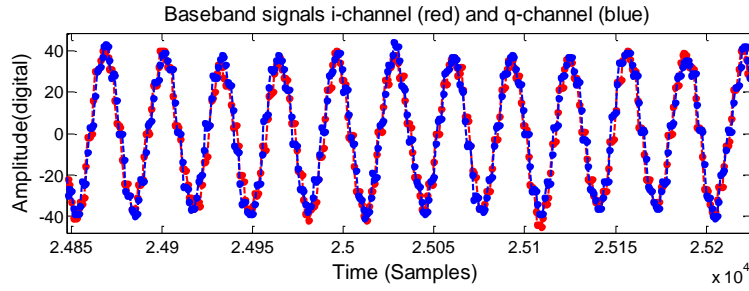


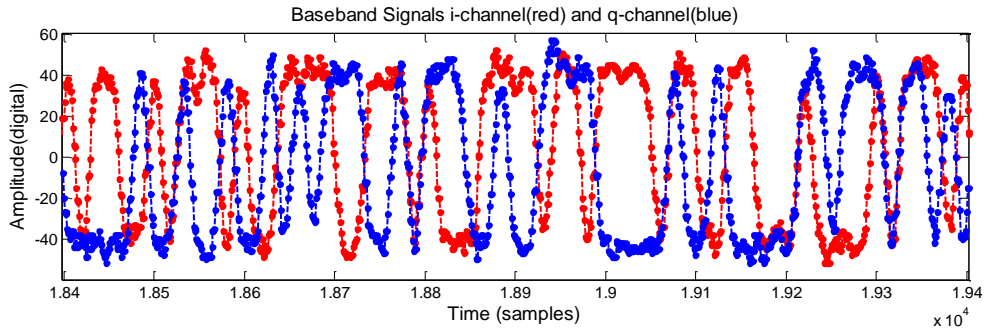
Figure 5.1: Sampled QPSK input signal

5.3. Correlator block simulation

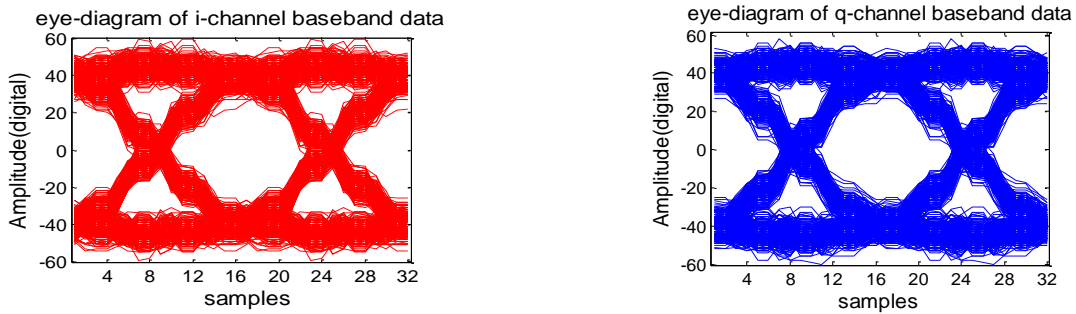
Using the coefficients of the designed matched-filter (see Figure 4.35) and the pipeline filtering algorithm (see Figure 4.40), the mixing and matched-filtering processes are implemented in Matlab, as given in Appendix A. The digital correlator blocks are simulated using the sampled QPSK signal and a set of carrier signals, initially generated by the local-carrier generator block, with assumed phase and frequency lock. The initial simulations yield output signals from the mixer and matched-filter blocks with huge amplitudes. The operation of high amplitude signal would be more time consuming for the programmable device; therefore, the signals were attenuated to lower values by factors of 2^2 for the mixers and 2^7 for the matched-filter. The simulation results obtained with the transmitted data, both alternating and random bit streams, are presented in Figure 5.2. The eye-diagrams of the resulting baseband symbols have been generated for each channel (see Figure 5.2 (c)), in order to evaluate the quality of the correlator blocks.



(a): The regenerated baseband signals with alternating bit stream



(b): The regenerated baseband signals with random bit stream



(c): The eye-diagrams of the *i*- and *q*- baseband bit streams

Figure 5.2: The reproduced baseband bit streams

The simulation results, confirm that the mixing and matched-filtering processes executed in the correlator block are operating properly. The baseband signals have the proper shapes for both the alternating and random bit streams. The eye-diagrams are also well-formed. These results have been obtained under the assumption that the local carriers are locked to the incoming carrier. However, in real life, a carrier recovery system needs to ensure that such synchronisation is achieved.

5.4. Carrier recover system simulation

The second-order PLL-based carrier recovery system has been presented in Section 4.4.5, with the PED gain k_p , loop filter's coefficients k_1 and k_2 , and the NCO gain k_0 determined as 2, 1, 2^{10} and $1/2^9$, respectively. The algorithms to digitally implement the various blocks of the CRS have

been presented in Table 4.2. The Matlab implementation of the algorithms are presented in Appendix A. The simulated phase step response of the closed-loop CRS to an input phase error of 45° is presented in Figure 5.3 for both alternating and random bit streams. Figure 5.4 shows the local carrier signals (cosine and sine) produced by the NCO during the carrier acquisition process.

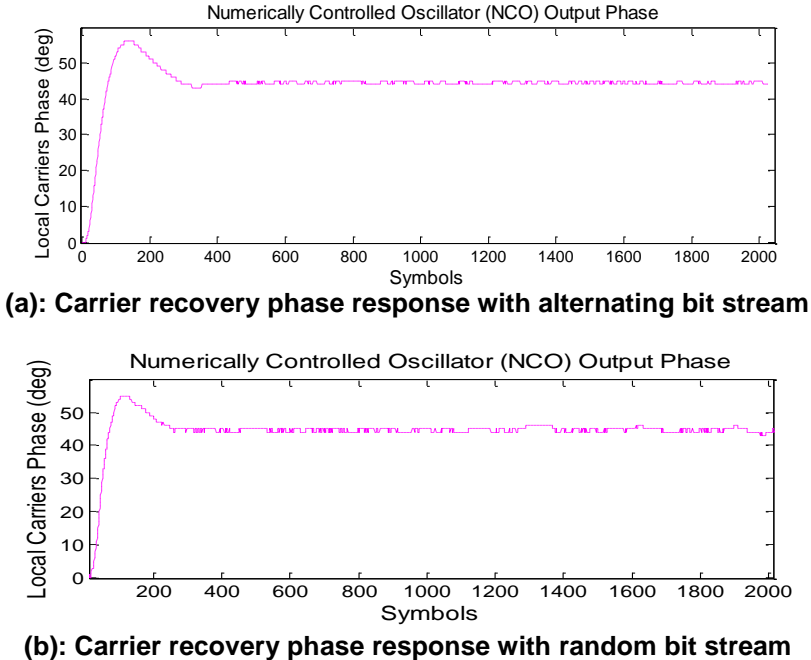


Figure 5.3: Step phase response of the carrier recovery system

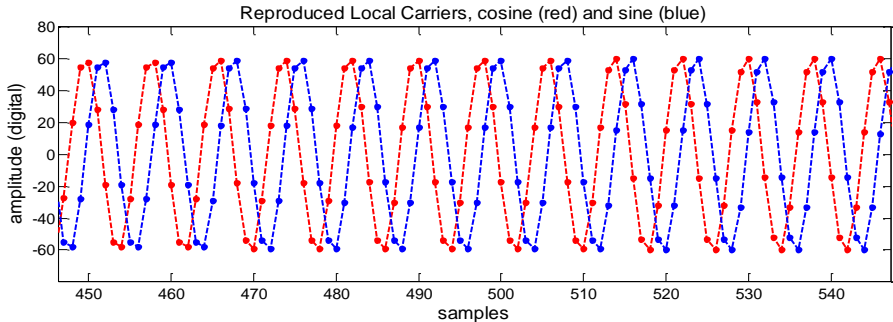


Figure 5.4: Local carrier signals generated by the NCO

Figure 5.3 shows the phase step response of the carrier recovery system as observed from the output phase of the NCO. The graphs show that the system locks accurately to the input phase offset within only $\pm 1^\circ$. The graphs also show that it takes about 200 symbols to acquire the phase offset, which means $T_{acq} = 200T_s$; consequently, the loop bandwidth of this second-order system would be about $0.65\%R_s$, which is close to the desired $1\%R_s$. Finally, the graphs show that the maximum overshoot for a 45° phase-offset is about 55° , which means a percentage

overshoot of 22%, which is deemed acceptable. Figure 5.4 shows a portion of the generated local carrier signals (cosine and sine) during the phase recovery process. One can observe how the phase is gradually changing symbol after symbol, until the phase lock is achieved.

Having verified the simulated carrier phase and frequency locking processes, which guarantee that baseband symbols will be produced by the correlator blocks accurately, a timing recovery system is required. The TRS ensures that the peak timing in every baseband bit is properly known for the detection of the baseband at its peaks, and hence, to minimise bit decision errors.

5.5. Timing recovery system simulation

The second-order PLL-based TRS has been presented in Section 4.4.6, with the PED gain k_p , loop filter’s coefficients k_1 and k_2 , and the NCO gain k_o determined as 10, 1, 2^{10} and $1/2^{10}$, respectively. The algorithms to digitally implement the various blocks of the TRS have been presented in Table 4.2 and Section 4.5.4. The Matlab implementation of the algorithms are presented in Appendix A. Again, the simulations have been executed using the sampled QPSK signal. The phase step response of the closed-loop TRS to an input time offset of $T_s/2 = N_{\text{symbol}}/2 = 8$ samples is presented in Figure 5.5 for both alternating and random bit streams. Figure 5.6 shows the clock signal generated by the TRS for bit detection.

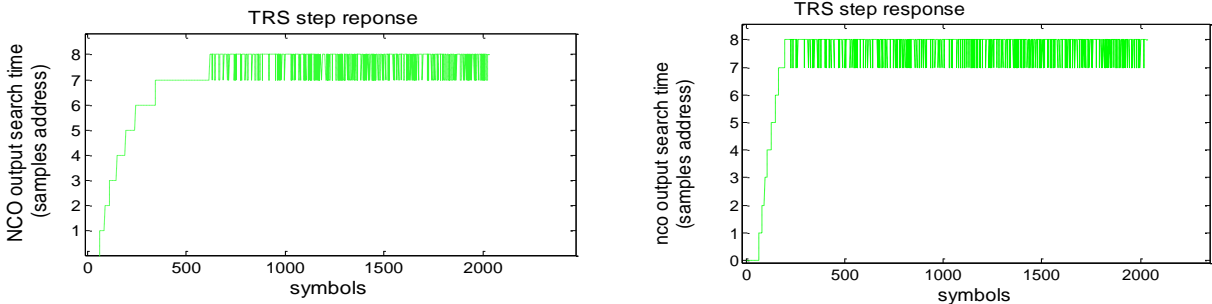


Figure 5.5: Step phase response of the developed timing recovery system

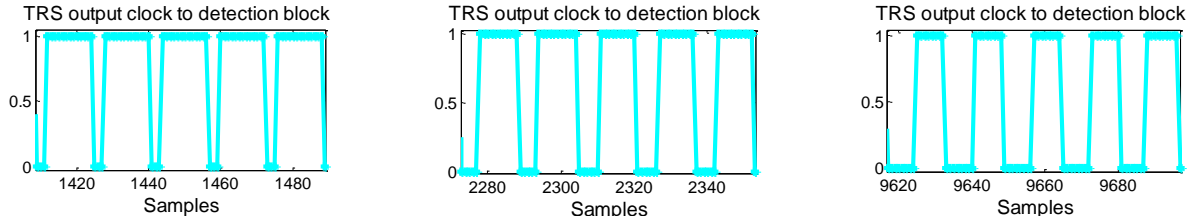


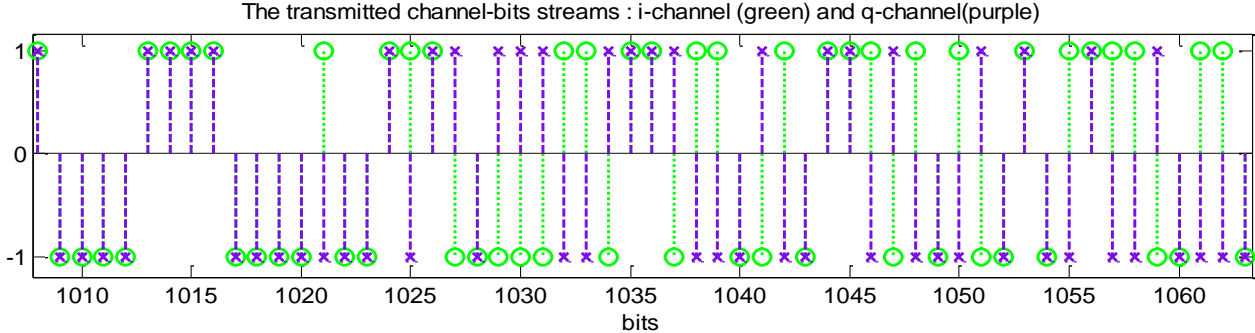
Figure 5.6: TRS output clock signal at different instants of the recovery process

The graphs show that the system locks accurately to the input timing offset to within only 1 sample accuracy. The graph in Figure 5.5 (a) shows that for random data, it takes about 250

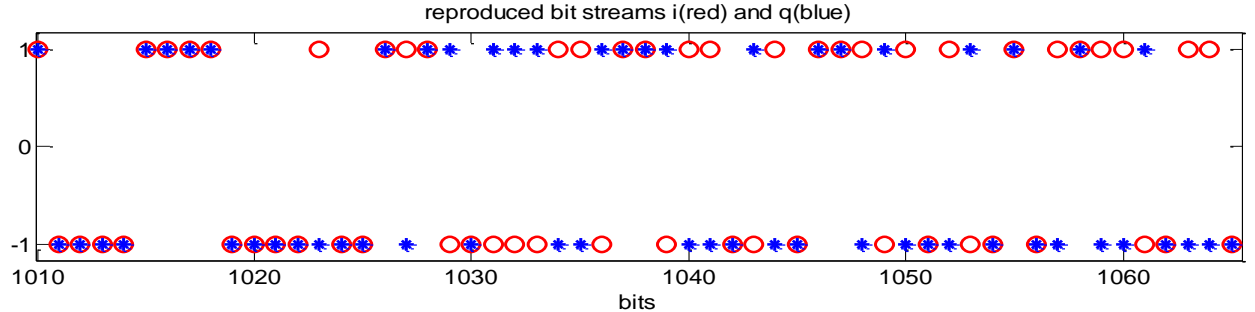
symbols to acquire the phase-offset, which means $T_{acq} = 250T_s$; consequently, the loop bandwidth of this second-order system would be about $0.55\%R_s$. The graph in Figure 5.5 (b) shows that for alternating data, it takes about 175 symbols to acquire the phase offset, which means $T_{acq} = 175T_s$; consequently, the loop bandwidth of this second-order system would be about $0.75\%R_s$. For both cases, this is close to the desired $1\%R_s$. Figure 5.6 shows the generated clock at different instants of the timing acquisition process. It is evident that the clock's rising edge time gradually adjusts until the desired clock signal is reached.

5.6. Detection and decision block technical design and simulation

The detection process has been performed in the interpolation of the timing recovery system. A portion of the code executing the decision process is presented Appendix A. Figure 5.7 (a) presents a portion of the transmitted *i*- and *q*- channel bits streams, and Figure 5.7 (b) presents the corresponding portion of the reproduced bit streams for both the *i*- and *q*-channel, as obtained from the simulation of the subsystem.



(a): a portion of the of the transmitted *i*- and *q*- bit streams



(b): The corresponding portion of the reproduced *i*- and *q*- bit streams

Figure 5.7:Reproduced versus transmitted bit streams on both the *i*- and *q*-channel

5.7. Evaluation of the simulated QPSK demodulator performance

As stated earlier, the Matlab simulation does not evaluate the *physical* performance of the demodulator, but validates the implementation of the algorithms as well as the theoretical performance of the system.

5.7.1. Quality of phase constellation for various SNR of input signal

The phase constellations generated from the simulations, for different input signal-to-noise ratios, are presented in Figure 5.8. As an initial observation, the graphs show well-structured constellations, which is evidence of a well-performing QPSK demodulator. Another observation is that the higher the SNR of the incoming QPSK signal, the more concentrated the constellation points. This is a logical observation, because the quality of the reproduced baseband signals is also a function of the SNR of the incoming PQKS signal.

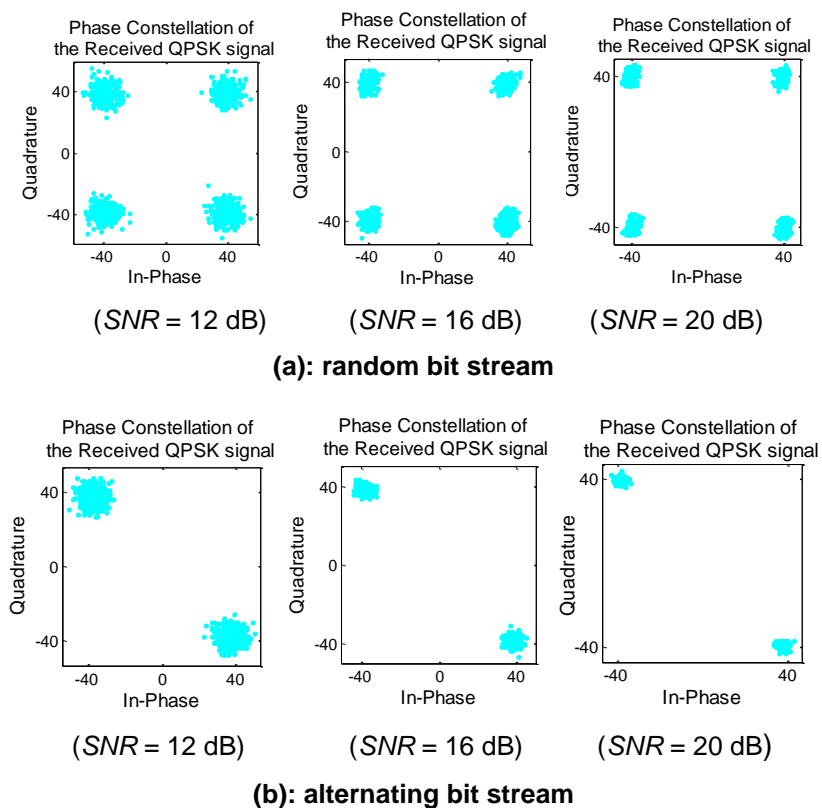


Figure 5.8: Phase constellation of demodulated received signal, for different input SNRs

5.7.2. Rotation effect of the carrier recovery process on the phase constellation

Figure 5.9 shows how the constellation of the received demodulated QPSK signal behaves during the carrier acquisition process. It can be seen that the constellation rotates as the phase locking process progresses until the symbols reach the expected position. This is because the amplitude of the baseband symbols on both the i - and q -channel vary as a function of the input phase-error.

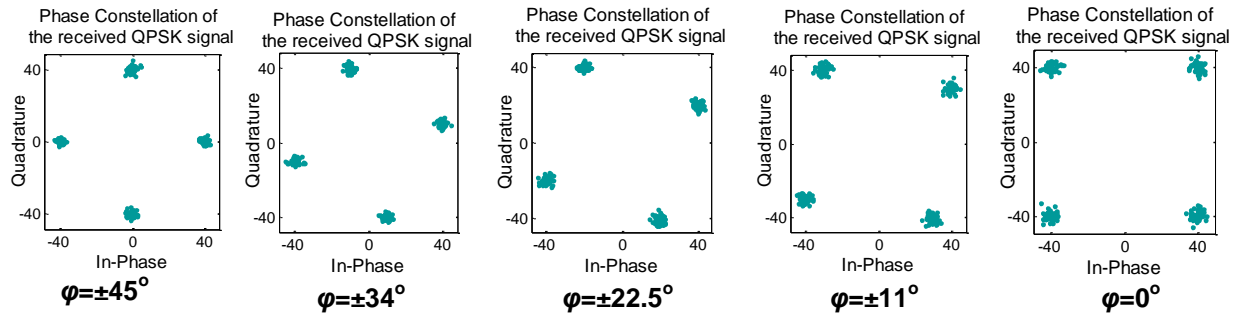


Figure 5.9: Illustration of constellation rotation as the phase-locking process progresses

5.7.3. Grouping effect of the timing acquisition process on the phase constellation

Figure 5.10 shows how the constellation of the received demodulated QPSK signal behaves as the timing acquisition process progresses. It can be seen that the phase constellation regroups during the timing acquisition process. Initially, the constellation points are spread over the quadrant, and gradually moving into the four constellation points as the timing acquisition process unfolds.

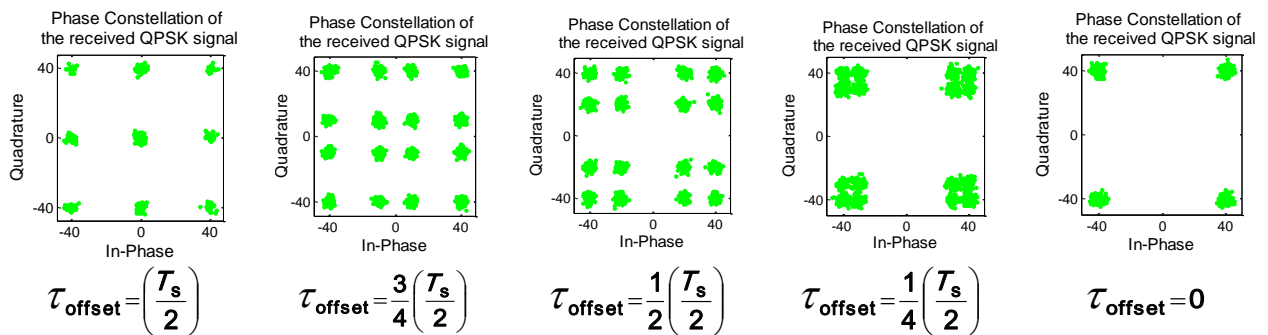


Figure 5.10: Illustration of the constellation stabilisation during the timing acquisition process

5.7.4. BER versus SNR performance

The simulated bit-error-rate as a function of the input signal-to-noise ratio is shown in Figure 5.11, together with the theoretical curve.

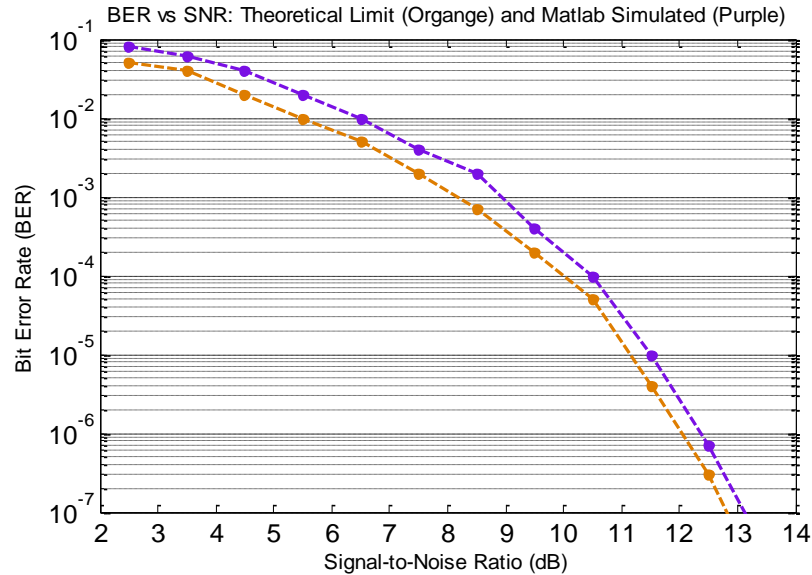


Figure 5.11: Simulated and theoretical BER versus SNR plot for the QPSK demodulator

It is evident from Figure 5.11 that the simulated results are in close agreement with the theoretical curve; in overall to within a factor of 2.

5.8. Chapter Summary

This chapter has discussed the Matlab implementation of the algorithms of the designed QPSK demodulator. The Matlab implementation is an effective validation tool of the algorithms and selection of the various parameters before implementing the system on a hardware platform. Clearly, this implementation does not evaluate the system's physical performance, such as power usage, maximum timing and resource usage. The various subsystem have been simulated using the sampled QPSK signal previously defined. The simulation results show that the carrier recovery system locks the local carriers properly to the incoming carriers, with a phase acquisition time of about $200 T_s$ and a loop bandwidth of about $(1.3/200) \times R_s$. Furthermore, the timing recovery system acquires the symbol peak time accurately, with an acquisition time of about $400 T_s$, and a loop bandwidth of about $(1.3/400) \times R_s$, for random data transmission. The performance of the developed system as described by the BER vs SNR plot has been found to be in close agreement with the expected theoretical performance, and validates the algorithms of the overall demodulator.

CHAPTER 6: FPGA IMPLEMENTATION OF THE QPSK DEMODULATOR

6.1. Introduction

The preceding chapter presented the simulation results of the Matlab implementation of the QPSK demodulator. In this chapter, the practical implementation of the QPSK demodulator is presented. First, the selection of a suitable programmable hardware platform for the application at hand is described. This is followed by an overview of the development of the system as well as the analysis and testing tools utilised, based on the selected hardware. The implementation of the QPSK demodulator in VHDL (very high-speed integrated circuit “VHSIC” hardware description language) is presented, followed by an evaluation of the system performance.

6.2. Hardware selection

6.2.1. Comparative study of programmable devices

Many types of programmable hardware devices are commercially available, including microcontroller (MCU), general purpose processor (GPP), graphic processing unit (GPU), digital signal processor (DSP) and field-programmable gate array (FPGA) devices. These devices are designed with different features. For example, some will have an improved DSP capability or flexibility, whereas others will be more power-efficient or more compact. Table 6.1, adapted from (Steven, 2011:8), summarises an overview of the various programmable hardware devices and their respective features.

Table 6.1: Comparison of programmable hardware

Feature	GPP	DSP	GPU	FPGA	Microcontroller
DSP operations & speed	Moderate	high	high	high	low
General operations	Good	Poor	Poor	Poor	Good
Flexibility	High	Low	Moderate	High	Poor
Available resources	Moderate	Small	Large	Large	Moderate
Power efficiency	Moderate	Good	Poor	Moderate	Good
Common brands	Intel, AMD	Texas Inst.	nVidia, AMD	Xilinx, Actel	PIC, Arduino
Programming language	C, Java	C, Assembly	CUDA, C	Verilog, VHDL	C, Assembly

The demodulator under design will form part of a software defined radio project for nanosatellites that is currently being undertaken at the Cape Peninsula University of Technology. This application requires the development platform to be flexible, power efficient, and to have DSP operations capability. The FPGA has high flexibility, large available resources, fast DSP capability and moderate power efficiency. It, therefore, stands out as a suitable hardware

programmable development platform for this application. Moreover, an FPGA has the capability to execute digital signal processing in parallel; decreasing the processing time considerably.

6.2.2. Selection of FPGA device

The selection of the FPGA depends strongly on the primary constraints of the application, such as processing speed, power consumption or radiation hardness. In CubeSat applications, the power consumption and the robustness against space radiation are two major constraints. However, due to the high-speed communication requirements of the demodulator, the processing speed is also a major constraint in the case of the all-digital implementation.

There are four well-known manufacturers of FPGAs; Xilinx, Altera, Lattice Semiconductor and Microsemi (also known as Actel). Although FPGAs from the different manufacturers now tend to present similar characteristics, the design technology employed by each manufacturer remains quite different from another. Due to this difference in the design and manufacturing technology used by each manufacturer, FPGAs from different manufacturers tend to have areas where they outshine others.

In terms of robustness against radiation, Xilinx, for example, uses a *FLASH*-based design, whereas Altera and Microsemi employ SRAM-based and anti-fused-based designs, respectively (Yaseen, 2016). In *FLASH*- and SRAM-based FPGAs, the memory of the architecture can be erased, making them vulnerable to radiation that can cause single-event-upsets. On the other hand, the anti-fused-based FPGAs cannot have their memory erased easily, improving their robustness in high radiation environments (Cunha, 2011:938). Therefore, if the application involves high radiation environments such as space, anti-fused-based FPGAs from, for example, Microsemi and Altera should be considered.

In terms of speed performance and power consumption, Xilinx FPGAs present better speed performance, whereas the Microsemi and the Lattice products have lower power consumption (Johnson, 2011).

Considering the above arguments, Microsemi FPGAs seem to be more suitable for the current application, since they exhibit both low power consumption, robustness against radiation, and can achieve relatively high processing speeds. Consequently, they have been considered for the implementation of the demodulator.

A few FPGAs manufactured by Microsemi include the *Polar Fire*, *ProASIC*, *Fusion Mixed Signal*, *IGLOO* and the *IGLOO2* (Microsemi Corporation, 2017). Of these, the *IGLOO2* has the lowest power consumption (Johnson, 2011), Moreover, it has already been used in the development environment at CPU, with a complete development kit available. Also, the *IGLOO2* has enough capacity to support the complete firmware of the QPSK modulator and demodulator. Finally, in terms of operating speed, as discussed in Section 6.2.1, FPGA in general can support relatively high processing speeds; the *IGLOO2*, in particular, can operate with main clock speeds up to 142 MHz (Microsemi Corporation, 2016:129). Such a speed capability is sufficient to achieve the data rates aimed for in this application. Therefore, the *IGLOO2* FPGA has been selected for the implementation of the digital QPSK demodulator.

6.3. Development steps and evaluation tools of FPGA-based systems

6.3.1. Integration and development environment

The development of a digital system is done in the *integration and development environment* (IDE), which is made available by the manufacturer of the programmable device to be used. For systems implemented on FPGAs, the development process often includes the following stages:

- creation of the design (generation of the architecture and syntax check);
- setting of the constraints;
- implementation of the design (synthesis, compiling, placing and routing); and
- loading of the system on the chip.

The IDE for Microsemi FPGAs is called “Liberio”. Figure 6.1 shows the design flow of FPGA-based systems in *Liberio*.

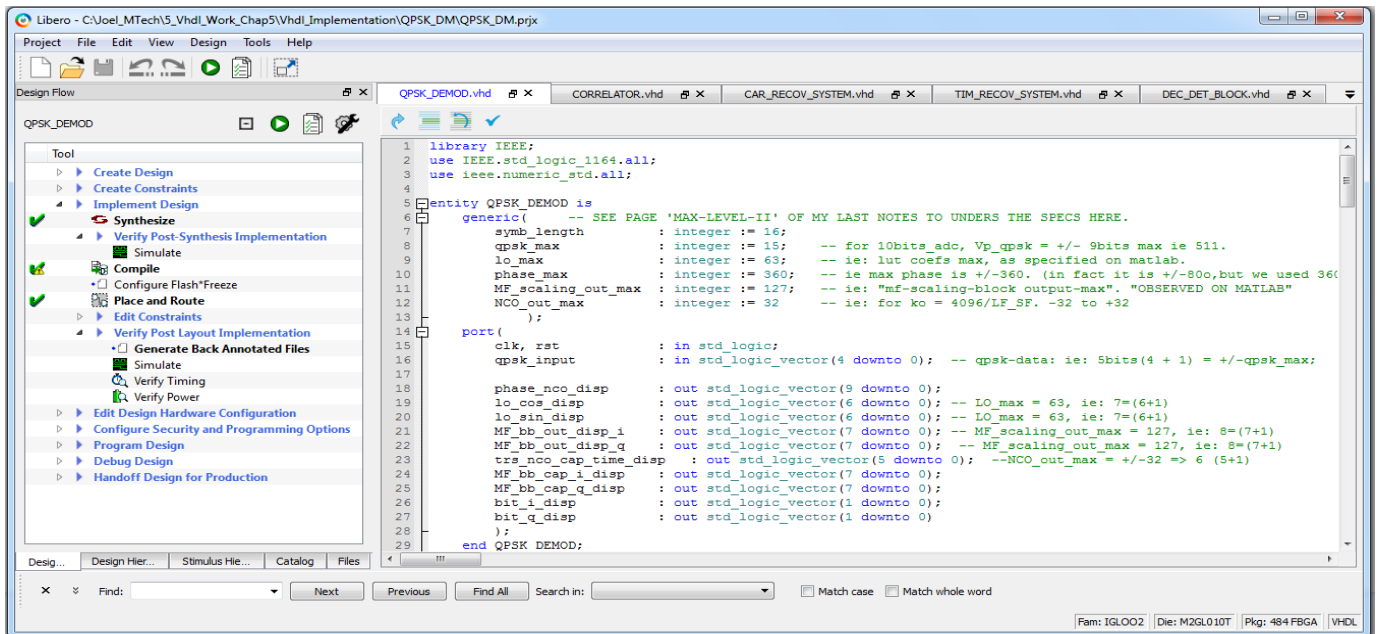


Figure 6.1: A view of Libero Graphical User Interface (GUI)

There are two main types of evaluations that are performed on the developed FPGA code namely parameters analysis and functional testing.

6.3.2. Parameters analysis process

In order to analyse the performance parameters of the generated code, such as the power usage, execution time, memory and other resource usage, the development environment uses sophisticated analysis processes. The first analysis comprises syntax checking, which verifies that the code implementation follows the VHDL syntax rules and good practices. The next analysis relates to the compilation of the code. This stage verifies that all the software processes and components developed in the system will execute properly on the hardware platform. The resulting report provides information regarding the system parameters, such as the memory usage, the number of gates and multipliers. Verification of the compiled code is followed by the placing and routing of the system. The tool analyse both the system's static and dynamic time and power behaviour. A report is generated that informs the developer if the time and/or power constraints will be met, based on the placing and routing performed. The report outcome also includes information such as the static power consumption of the system and the maximum clock frequency.

6.3.3. Functional testing tools

Regarding the functional test of the developed code, a tool called “*test-bench*” is used by the development platform. *Test-bench* is a platform created in the integration and development environment (IDE) in order to run functional tests of the developed architecture as if it was running on a chip, and observe the relevant signals on a virtual scope. It describes the behavior of all control input signals, such as the *reset*, *clock* and *enable*, and specifies certain operating parameters, such as the *global clock frequency* and *subsequent clock frequencies* (if any). It also defines how input data are acquired and transferred within the system for processing. The code/firmware can be tested by the *test-bench* at all the stages of development; before compilation (pre-synthesis implementation), after compilation (post-synthesis implementation) and after placement and routing (post-layout implementation).

It is important to note that the test-bench tests performed at the post-layout implementation stage are usually very close to the actual results obtained on the chip. Therefore, these tests are often accepted to validate a developed system (Blanchard, 2017; Booyesen, 2017).

The monitoring of the test-bench simulations is done using *ModelSim*. ModelSim is a verification and simulation tool for a code that is based on hardware descriptive languages (HDL) such as the VHDL, Verilog and system Verilog (Mentor Graphics Inc., 2012:9). It is a product of *Graphics Corporation*, and measures and displays all necessary input and output signals specified in the test-bench.

6.4. Generation of FPGA code for the demodulator

6.4.1. Good VHDL coding practices

VHDL is one of the programming languages used to programme FPGAs. It allows for tasks to be executed in parallel (concurrently) or in series (sequentially). The capability to execute tasks concurrently makes this language a very important tool in high-speed applications, such as the one at hand.

The art of developing algorithms in VHDL involves a careful interplay between the execution time and the memory required for the implementation of a given task. Some codes can be written using the *pipeline implementation* approach and others can be written using the *multi-cycle implementation* approach. The choice of coding approach depends on the requirements of the application with regards the execution time and memory.

Time and memory usage can also be minimised through the proper selection of the data types used. For example, large integers and floating point numbers are processed with enormous difficulty by hardware, and require large amounts of memory and longer execution times. Therefore, ensuring that the numbers used are relatively small will lessen the time and memory burden on the platform.

6.4.2. Code development

As stated earlier, the digital QPSK demodulator has been developed in *libero*, which is the IDE for microsemi *IGLOO* FPGAs. It should be noted that the VHDL code that has been generated, is totally generic; all the functions implemented, have been obtained from the standard VHDL libraries, or generated from scratch. The generated code can, therefore, be executed on any family of FPGA without restriction to a specific one.

Using the VHDL syntax, the code for each subsystem has been developed, based on both the technical design and the algorithms covered in Chapter 4. The code of the *top level* (the complete QPSK demodulator system) is presented in Appendix B. Due to space constraint in this document, the codes of the bottom levels (all subsystems and their sub-blocks) can not be presented; however, all these codes are available in the CD made for this project. Figure 6.2. shows the block diagram of the FPGA implementation of the QPSK demodulator.

6.5. Emulation results of the developed FPGA-based QPSK demodulator

6.5.1. Parameters analysis

The implementation of the QPSK demodulator on the FPGA has been evaluated and analysed. The summary report of the logic, memory, timing and power analyses are presented in Appendix C. Table 6.2 lists the results for certain pertinent parameters.

Table 6.2: Resource usage analysis of the QPSK demodulator implementation on the FPGA

	Item	COR	CRS	TRS	DDB	demodulator	Available
Resource Usage (Memory Analysis)	4LUT	1329	3838	616	291	6074	12084
	DFF	1346	1404	519	394	3663	12084
	User I/O	24	31	53	15	46	231
	Single-ended I/O	24	31	53	15	46	231
	MACC	14	5	0	2	21	22
	Chip Global	2	2	2	2	2	8
Timing Analysis	Min clock period	6.819 ns	8.829ns	8.438 ns	6.614ns	9.052 ns	-
	Max clock frequency	146 MHz	113 MHz	118 MHz	153MHz	106 MHz	-
							-
Power Usage	Static power	2.1mW	5mW	4mW	0.6mW	12 mW	-
	Dynamic power	0 mW	0 mW	0 mW	0 mW	0 mW	-

6.5.2. Functional test

The *test-bench* generated to perform the functional testing of the QPSK demodulator is presented in Appendix D and is based on the following emulation considerations:

- the input signal is the QPSK signal as reported in Chapter 3, Section 3.3. As stated earlier in the mentioned section, the QPSK signal is generated for different SNRs, with an AWGN power of -60 dBm, but only the emulation performed with a SNR of 12 dB is

presented. The total power of the received QPSK signal at the input port of the demodulator is about -45 dBm, which corresponds to a total amplitude of 1.7674 mV in a 50Ω system.

- the test data and symbol rates of transmission are set at $R_d = 10$ Mbps and $R_s = 5$ Msps, respectively;
- the carrier frequency is, therefore, set at $f_c = 2R_s = 10$ MHz;
- the ADC sampling frequency is assumed to be $f_{sa} = 8f_c = 16R_s = 80$ MHz, with a 12-bit resolution and a dynamic range of 1 Vpp;
- the global/system clock frequency is set at $f_{clk} = f_{sa} = 80$ MHz;
- samples of the QPSK signal have been read from a data file and transferred into the system under test by the test-bench, and at the sampling rate defined above;
- the system then processes the input samples sequentially, over every cycle of the system's clock (f_{clk}) as defined above; and
- the test has been performed after the place and route stage (timing and power analyses), in order to have results that are as close as possible to the on-chip performance.

Note that the tests were run with different values of the SNR of the input signal, one value at a time. However, due to space constraint in this document, only the results obtained with 12 dB SNR are presented here (Figure 6.3 through Figure 6.9), because it is the worst case scenario of SNR necessary to achieve a BER of 10^{-6} for the QPSK scheme.

The results displayed in Figure 6.3 through Figure 6.9 have been extracted from *ModelSim*. These graphs present the input QPSK signal, the reproduced baseband signals (i and q) with both alternating and random input bit streams, the step-phase response of the carrier recovery system and the generated local carriers, the step-phase response of the timing recovery system and the generated clock for detection timing, as well as the reproduced bit streams on both i - and q -channels.

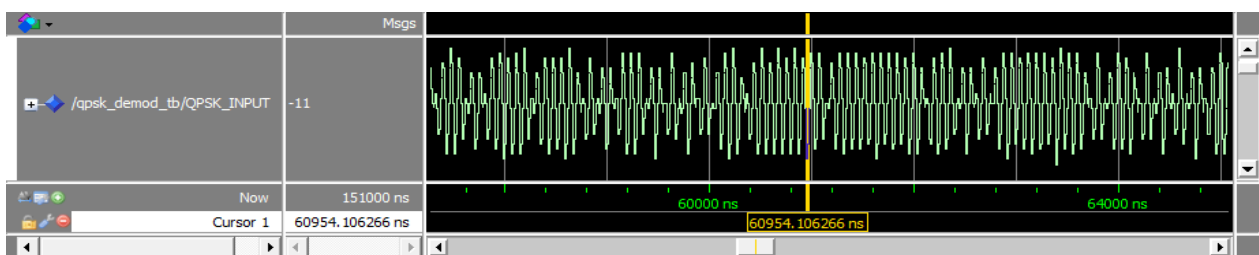
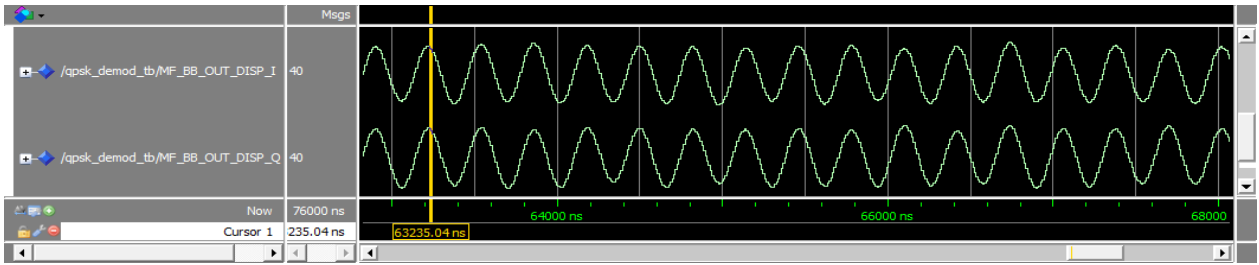
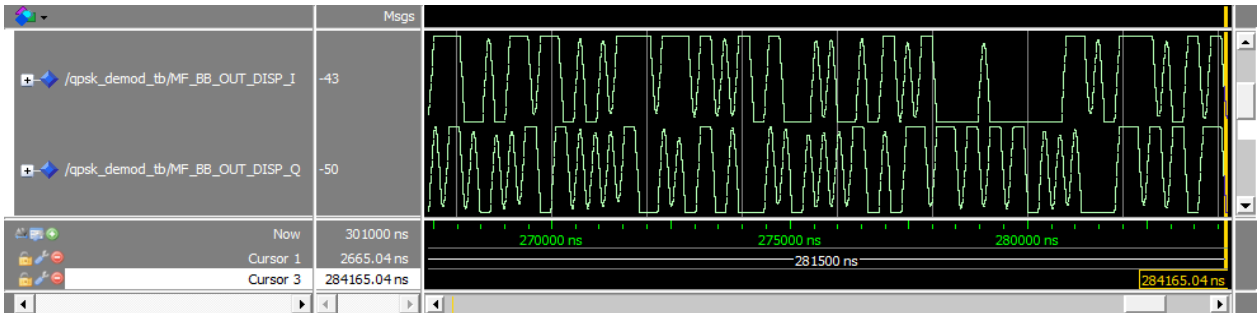


Figure 6.3: Test-bench input QPSK signal



(a) Baseband signals reproduced with alternating bits stream



(b) Baseband signals reproduced with random bits stream

Figure 6.4: Baseband signals reproduced by the correlator blocks

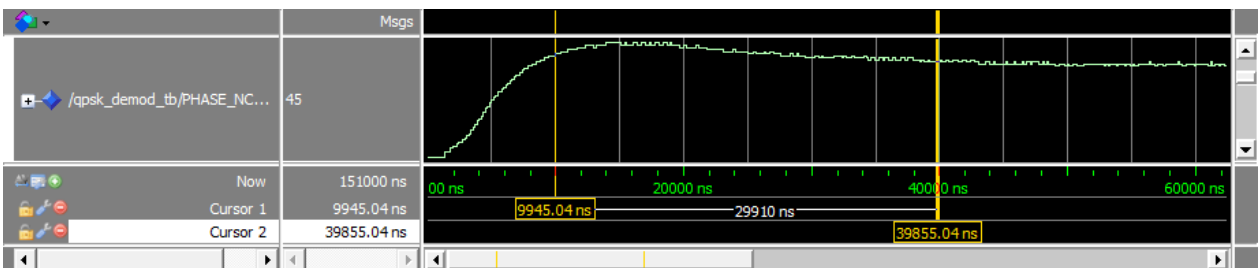


Figure 6.5: Simulated carrier recovery phase-step response

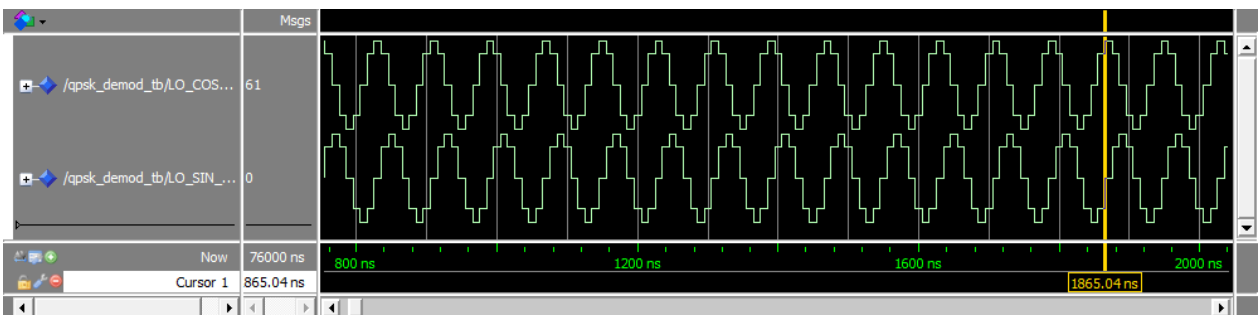


Figure 6.6: Generated local carrier signals from the carrier recovery system

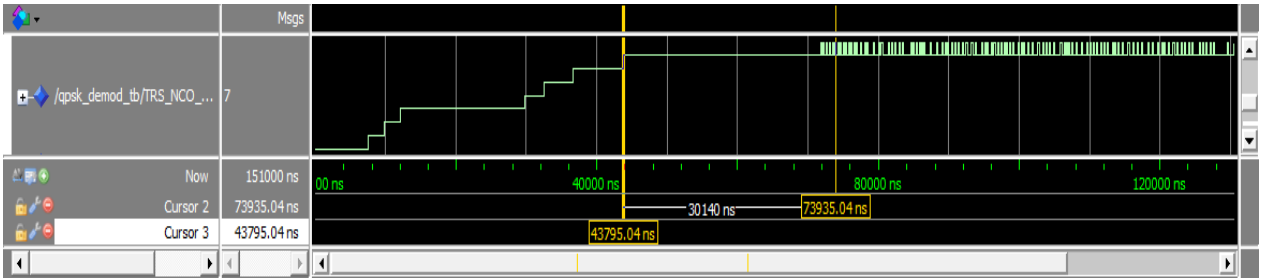


Figure 6.7: Simulated timing recovery phase-step response

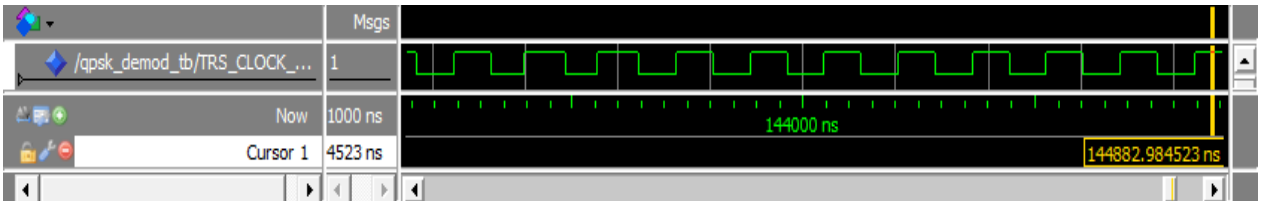


Figure 6.8: Clock signal generated by the timing recovery system for bit detection by the DDB

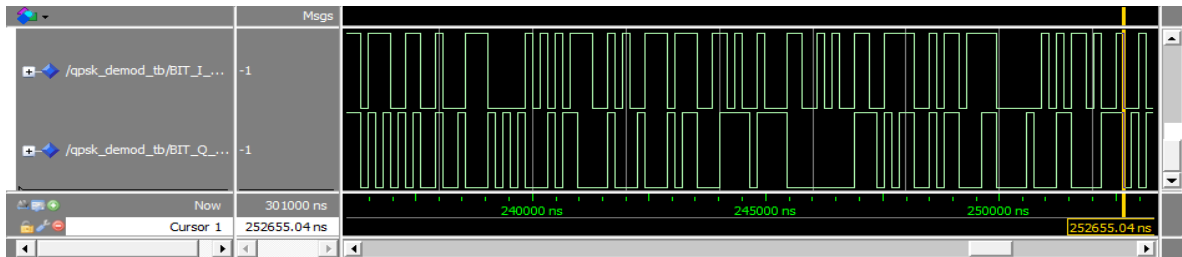


Figure 6.9: Reproduced bit stream i -(top) and q -(bottom) channels

For every symbol received, the captured amplitudes of the i - and q -bits can be used to plot the phase constellation diagram of the received QPSK signal, as shown in Figure 6.10. Also, the bit stream reproduced after each test is used to evaluate the bit-error-rate of the system for that particular signal-to-noise ratio of the input signal. The resulting BER versus SNR of the overall system is shown in Figure 6.11. Careful observation of graphs in Figure 5.8 and Figure 6.10 shows that simulation results obtained with matlab compare well with the FPGA emulation results obtained from the Test-Bench.

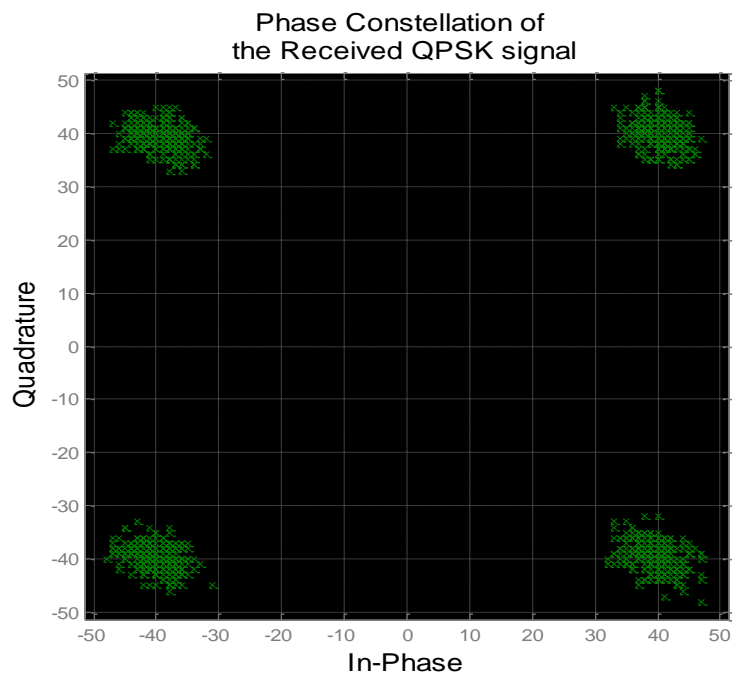
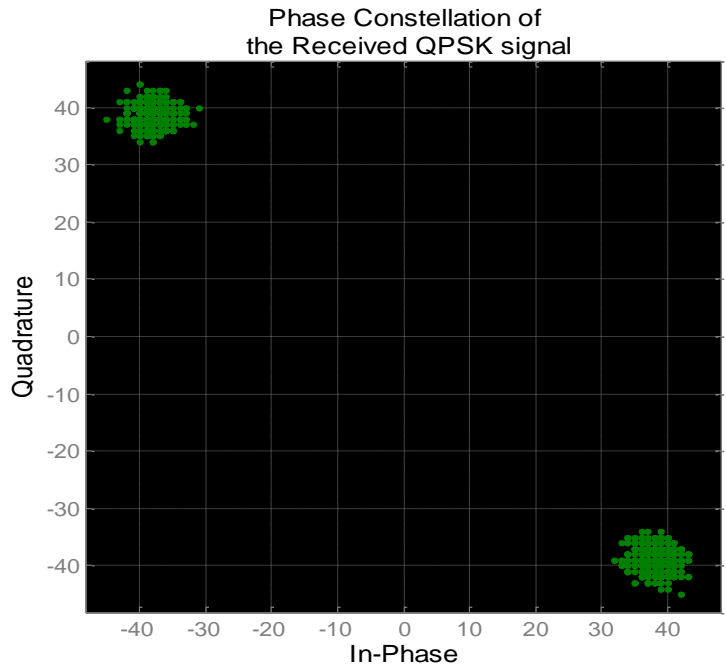


Figure 6.10: Phase constellation diagrams with (a) alternating and (b) random bit streams with an input SNR of 12 dB

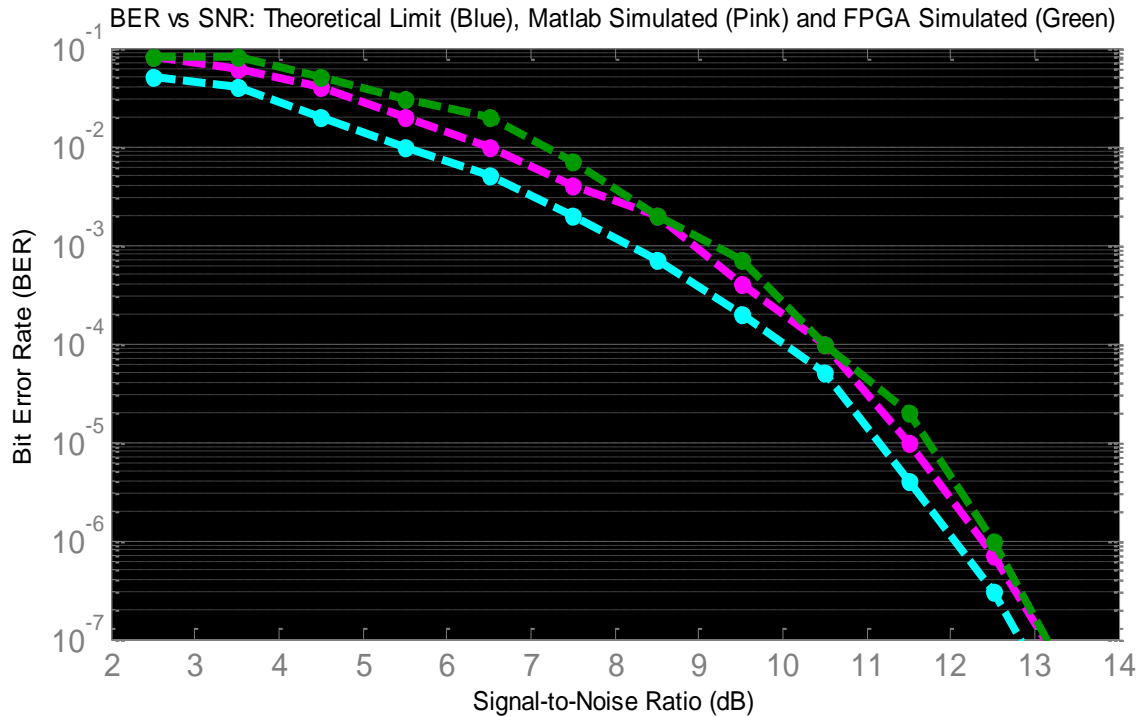


Figure 6.11: Simulated BER versus SNR plot

6.6. Interpretation and evaluation of results

6.6.1. Resource utilisation

Memory utilisation as demanded by the system can be accommodated by selecting an appropriate chip, but is generally not considered a major consideration as chips with larger memory capacity can readily be found.

Analysis of the *power consumption* has shown that the static power consumption of the full system is about 12 mW, which is acceptably low.

Timing analyses have shown that the maximum clock frequency that can be used for the QPSK demodulator is about $f_{\text{clk-max}} = 106$ MHz. Given that the FPGA clock frequency should be equal to, or higher than, the ADC sampling clock, and since all the subsystems and sub-blocks run with the same clock, the FPGA clock frequency has been chosen to be equal to the ADC sampling frequency. Therefore, the maximum sampling frequency is 106 MHz. It was stipulated in the theoretical design section that the number of samples-per-symbol should be 8 or more for the accurate modeling of the digital matched-filter. This design is indeed flexible in terms of the number of samples-per-symbol; it can be 8 or any multiple of 8. Consequently, when the system

is implemented with 8 samples-per-symbol, the maximum achievable symbol rate would be $106/8 = 13.25$ Msps, which results in a maximum data rate of $R_{d\text{-max}} = 26.5$ Mbps. When the system is implemented with 16 samples-per-symbol, the maximum achievable symbol rate would be $106/16 = 6.665$ Msps, which results in a maximum data rate of $R_{d\text{-max}} = 13.25$ Mbps. *In either of these two cases, the achievable data rate is greater than the 10 Mbps specification set.*

6.6.2. Operational results

The results presented in Figure 6.4 show that the mixers and matched-filters are working well; as a result the generated baseband signals are of good quality despite a moderate input signal-to-noise ratio of 12 dB.

Figure 6.5 shows the phase-step response of the carrier recovery system where the phase and frequency of the local carriers lock perfectly within 1° accuracy to the phase of the input carrier. The response also shows that the carrier acquisition time is 9945 ns. With the frequency of the main clock used, set to 80 MHz (for 10 Mbps), the clock period is 12.5 ns. Subsequently, with a symbol length of 16 samples-per-symbol, the symbol period is $16 \times 12.5 = 200$ ns. Therefore, a carrier acquisition time of 9945 ns corresponds to $9945 / (16 \times 12.5) = 50$ symbols; and results in a loop bandwidth of $B_L = 2.5\% R_s \approx 125$ kHz. *Achieving such a short acquisition time, coupled with a high system accuracy, is quite an achievement. This also reflects that the loop bandwidth is at its optimum value in terms of trade-off between being too large for poor accuracy and short acquisition time, and being too narrow for high accuracy and long acquisition time.* Figure 6.6 above shows the local carrier signals generated by the carrier recovery system, in which the phase adjustment process can be observed.

In a similar manner, the timing recovery system's phase-step response given in Figure 6.7 shows the local timing locking accurately to the peak timing of the incoming basebands signal to within 1 sample accuracy. The response also shows an acquisition time of 43795 ns, which corresponds to $43795 / (16 \times 12.5) = 218$ symbols, and results in a loop bandwidth of $B_L = 0.6\% R_s \approx 30$ kHz. *The longer acquisition time here is a result of the time taken by the carrier recovery system to first achieve the carrier acquisition.* However, the acquisition and the loop bandwidth are still acceptable as justified by the accurate results obtained. Figure 6.8 shows the clock generated by the timing recovery system for the detection block.

The phase-constellation diagram in Figure 6.10, plotted for an input SNR of 12 dB, shows well-concentrated symbols around the fixed decisions points, which verifies the good quality of the

demodulator implementation. A further verification of the good quality of the implemented QPSK demodulator can be observed from the BER versus SNR plot in Figure 6.11, which shows excellent agreement with the theoretical expectation. For example, with 12 dB SNR, the theoretical limit of the BER is 10^{-6} , and the simulated BER for the FPGA-based demodulator is about 5×10^{-6} . The simulated BER is therefore within a factor 5 of the theoretical BER limit. *This, together with the high-speed capability of the implemented demodulator, is note-worthy.*

6.7. Chapter summary

The implementation of the QPSK demodulator on an FPGA has been presented in this chapter. The FPGA has been selected for having the best trade-off between operating speed, flexibility, size and power consumption. Subsequently, the Microsemi family has been selected due to its robustness against space radiation. The IGLOO2 FPGA from the Microsemi family has been selected for its extremely low power consumption features. *Libero* has been used for the VHDL implementation of the demodulator on the FPGA.

In order to evaluate the speed capability and power consumption of the digital QPSK demodulator fully developed on FPGA, the system has been analysed with various tools in the integration and development environment. The results show that the maximum acceptable clock frequency of the system is 106 MHz, which means that the maximum achievable data rate would be 13.25 Mbps if an ADC sampling rate of 16 samples-per-symbol is used, and 26.50 Mbps if an ADC sampling rate of 8 samples-per-symbol is used. In either case, the achievable data rate is greater than the 10 Mbps specified. The results also show that the power consumption of the system when running at the maximum frequency (106 MHz) is less than 15 mW, which is adequately low.

Similarly, in order to further evaluate its practical performance, the demodulator has been tested using a test-bench and a QPSK signal with -60 dBm of AWGN noise power and various possible values of SNR from 0 dB up to the maximal acceptable SNR value for a 12 bit ADC. However, only the results achieved with one value of SNR (in this case 12 dB) have been presented in this chapter (Figure 6.3 though Figure 6.11) solely due to space constraints of this document. The results obtained with both alternating and random bit streams, and for different signal-to-noise ratios of the input signals, show that the good quality baseband signals reproduced by the correlator block are the result of good match-filtering and good carrier recovery. The test-bench results also reveal a well-functioning carrier recovery system with a short acquisition time of $50 \times T_s$, and a phase-lock accuracy of 1° that is a result of an excellent

loop bandwidth of the order 2.5% of R_s . In addition, the results also show a well-functioning timing recovery system with a longer acquisition time of $218 \times T_s$, but a phase-lock accuracy of 1 sample that is a result of a narrow loop bandwidth of $0.6\% R_s$.

Finally, the resulting phase-constellation diagramme of the received QPSK symbols with a 12 dB SNR reflects a very good quality QPSK demodulator. This has also been confirmed through the BER versus SNR plots for the overall system, which has proven to be very close to the theoretical expectation, in such a way that the simulated BER values are within a factor of 5 of the theoretical BER limits.

CHAPTER 7: CONCLUSION AND RECOMMENDATIONS

7.1. Introduction

This project was conceived from the perspective that the nanosatellites industry is fast growing. Nanosatellites are destined to provide services, such as Earth observation and real-time point-to-point communication, which require high-speed communication links. The design and implementation of a high data rate QPSK demodulator for nanosatellites, as have been presented in this work, form part of a broader programme at CPUT to develop such high-speed communications links.

7.2. Meeting the research objectives

It is recalled that the objectives defined for this research project include:

- Design and implement a QPSK demodulator that meets the following specifications:
 - support data rates up to 10Mbps;
 - operate within available amateur VHF-, UHF-, L-, S-, C-, or X- bands.
 - be of a reasonably good quality, that is, $BER_{\text{simulated}}$ within a factor of 10 of the ideal $BER_{\text{theoretical}}$;
 - comply with the PC104 standard that CubeSats generally deploy;
 - consume less than 100 mW at 10 Mbps;
 - be flexible and reconfigurable for the purpose of integration with software-defined-radio (SDR) platforms; and
 - present some level of robustness against space radiation.
- Understand the technical difference between different digital modulation schemes and be able to motivate the selection of one for a given application.
- Investigate which implementation approach of the all-analog, all-digital or hybrid of both would be the most appropriate to implement the desired QPSK demodulator.
- Identify, for each subsystem of the QPSK demodulator, existing techniques used to implement the subsystem; then select the one that would be most suitable to accomplish the expected task while complying with the constraints set.
- Attempt to be creative through the design and development of the various subsystems in order to be able to either come up with a novel implementation technique of a subsystem or produce a novel overall system output.

The assessment of the QPSK demodulator implementation against each of these objectives is briefly discussed below.

7.2.1. Performance of the developed QPSK demodulator system

The desired QPSK demodulator was successfully developed fully on the *IGLOO2* FPGA. Its performance as per the tests performed, are summarised below.

7.2.1.1. Maximum supported data rate

From the measurements presented in Sections 6.5 as well as the subsequent discussions in Section 6.6.1, data rates of up to 13.25 Mbps can be supported if a sampling rate of 16 samples-per-symbol is applied; and up to 26.50 Mbps if a sampling rate of 8 samples-per-symbol is applied. This means that the developed QPSK demodulator meets and exceeds the required data rate of 10 Mbps.

7.2.1.2. Operating frequency spectrum band

Applying a root-raised-cosine pulse shaping-filter with $\alpha = 0.36$ requires a channel bandwidth of at least 7 MHz to achieve the 10 Mbps data rate. This implies that S-, C- or X-bands should be utilised. However, if lower data rates are required (≤ 1 MHz), lower bands (VHF or UHF) could be used.

7.2.1.3. BER performance of the QPSK demodulator

The demodulator's performance test results as presented in Section 6.5.2 show that the BER-SNR curves are in close agreement with the theoretical limits of the QPSK modulation scheme, without error correction. The demodulator achieves a BER of about 5×10^{-6} for a 12 dB SNR, whereas the theoretical BER is 10^{-6} at the same SNR.

Note that once error correction has been implemented, the overall BER of the receiver will improve from its current status. Also note that a way of improving on this performance would involve further optimisation of the design of the subsystems, such as the matched-filters.

7.2.1.4. Adherence to PC104 standard

The QPSK demodulator has been implemented totally on a single FPGA chip; hence, its small size allows it to be easily integrated with other hardware on a PC104 size printed circuit board.

7.2.1.5. Power consumption

The measurements presented in Section 6.5 show that the maximum power consumption of the resulting FPGA-based QPSK demodulator is of the order 15 mW when operating at the clock frequency of 106 MHz. Note that the achievable data rate is 13.25 Mbps if a sampling rate of 16 samples-per-symbol is applied, and 26.50 Mbps if a sampling rate of 8 samples-per-symbol is applied. This means that for both cases, the achievable data rate with a clock frequency of 106 MHz is more than 10 Mbps. Therefore, the FPGA-based QPSK demodulator meets the power consumption requirement set of less than 100 mW at a data rate of 10 Mbps.

7.2.1.6. Flexibility and reconfigurability

In order to comply with the software-defined-radio (SDR) requirements, a technical decision was made to implement all the subsystems of the QPSK demodulator fully digitally. It has been shown that the fully digital implementation is able to achieve the high data rates required, despite the speed limitations associated with the digital implementation due to the ADC maximum sampling frequency as well as the maximum possible clock frequency. The fully digital implementation has the advantage that different modulation or demodulation protocols (schemes) can be implemented on the same chip, without increasing the mass or size of the receiver/transmitter, as is the case with analogue implementations. Another advantage of the all-digital implementation is that it is very flexible, as everything is set up in software. As a result of these two stated advantages, the developed system can be remotely modified, reconfigured or even isolated without necessarily adding or removing any hardware. That is the essence of SDRs, which this FPGA-based QPSK demodulator complies with.

7.2.1.7. Mitigation of the effects of space radiation at design and development level

In terms of ensuring that the system has some robustness against space radiation, the FPGA was selected based on its reported performance in a radiation environment. An *anti-fused-based* FPGA from Microsemic has, therefore, been selected during the development phase. Anti-fused-based FPGAs are manufactured to be more immune to space radiation than other FPGA technologies, such as the *flashed-based* (Xilinx) and *SRAM-based* (Altera) FPGAs.

7.2.2. Understanding the performance parameters of digital modulation schemes

There exist many families of digital modulation techniques, including amplitude shift keying (ASK), frequency shift keying (FSK), phase shift keying (PSK) and possible hybrid of ASK and

PSK. These families comprise modulation techniques that employ the principle of coherent or non-coherent detection. The technical difference between digital modulation schemes lies in their *spectral efficiency*, *power efficiency* and the *complexity to implement* the respective modulation schemes. Therefore, the selection of a modulation scheme for a particular application considers the available power and spectrum, the desired data rates as well as the expertise of the designer. The best choice is often the scheme that provides the best trade-off between the three limitations.

7.2.3. Investigation of the most adequate implementation approach

The difference between implementation approaches has been discussed in Section 4.2. From that discussion, the *all-analog implementation* is capable of operating at extremely high speeds; but at the cost of power and size. In addition, it will be very complex to implement, and will not be flexible or reconfigurable. The *hybrid implementation* will provide high-speed communications and reduced size, but with limited flexibility. In addition, it will be quite complex to implement and be costly in terms of power. The *all-digital implementation*, however, is flexible and reconfigurable, while also having small power and size requirements. Its major drawback is that it has a reduced capability in terms of operating speed.

7.2.4. Understanding the various techniques used for the design of subsystems

Various techniques for the design of each of the subsystems of the QPSK demodulator have been described in Chapter 4. The most appropriate techniques to implement the automatic gain control (AGC), correlator blocks, carrier recovery system (CRS), timing recovery system (TRS) and detection and decision block (DDB), which constitute the QPSK demodulator, have been described.

7.2.5. Contribution and novelty of the research

The primary contribution of this research is that the developed digital QPSK demodulator contributes to the completion of the high-speed receiver for nanosatellites under development at F'SATI. This high-speed receiver, once completed, will be combined with existing nanosatellite high-speed transmitters, to form high-speed transceivers that will be available off-the-shelf for F'SATI/CPUT nanosatellites.

The secondary contribution of this research is that the SDR-based high-speed receiver that supports high data-rate communications up to 10 Mbps, will be made available to the community of nanosatellites developers. As discussed in Section 1.8, this contribution is significant to the nanosatellite industry for a few reasons. On one hand, CubeSat developers such as Planet-Labs and Spire-Inc, who currently have advanced communication technologies, capable of achieving data rates above 10 Mbps with their nanosatellites receivers, do not make their products available to the community (Planet Labs, 2017; Spire, 2017). On the other hand, all of the nanosatellites communication products available to date, no CubeSat receiver can support uplink communications beyond 5 Mbps (Clyde-Space, 2017; F'SATI, 2017; Gomspace, 2017; ISIS, 2017; Syrlinks, 2017; Tethers Unlimited, 2017). Therefore, this product will be a relief to nanosatellites developers who wish to achieve applications, such as *in-situ monitoring*, *instant point-to-point communication*, *space weather monitoring* and *high-quality Earth observation*; which require high speed communications from ground-to-satellite or between satellites, and which therefore need a high speed receiver for such tasks.

7.3. Recommendations

- The FPGA-based QPSK demodulator has been emulated in this work. It is recommended to implement the demodulator practically, and productise it for future deployment in the nanosatellite missions of CPUT.
- Furthermore, higher data rates can be achieved through a hybrid implementation. However, that will lead to increased complexity and power consumption.

REFERENCES

- Agrell, E., Lassing, J., Strom, E.G. & Ottosson, T. 2007. Gray Coding for Multilevel Constellations in Gaussian Noise. *IEEE Transaction Information Theory*, pp.1-13.
- Ahamed, S. V. & Lawrence, V.B. 1997. *Design and engineering of intelligent communication systems*. Massachusetts: Kluwer Academic.
- Alexander, G.M. & Henrique, J.A.S. 1996. Optimization of the QPSK demodulator for digital satellite radio. *IEEE proceedings of the ISCAS, 7-9 May 1996*. pp1970-1973.
- Ashok, A. 2007. *Digital signal processing: A modern introduction*. Canada: Nelson, Inc.

- Bae, K.T., Jeon, W.G. & Cho, Y.S. 1997. Convergence analyses of timing and carrier recovery loops for digital communication channel with adaptive equalizer. *IEEE Proceedings of the 40th Midwest Symposium on Circuits and Systems, August 1997*. pp.1366-1369.
- Berner, M. 2005. Postgraduate Course in Radio Communications: Timing and carrier recovery. Class notes, IKI, Finland.
- Booyesen, S. 2009. The design of a high speed topology for a QPSK demodulator with emphasis on the synchronization algorithms needed for demodulation. MSc thesis, University of Stellenbosch, South Africa.
- Bozic, S.M. 1979. *Digital and Kalman filtering*. London: Edward Arnold Ltd.
- Brillant, A. 2008. *Digital and analog fiber optic communications for CATV and FTTx applications*. New Jersey: John Wiley & Sons, Inc.
- Cagri, A.U. & Schumacher, H. 2011. A system-on-package analog synchronous QPSK demodulator for ultra-high rate 60 GHz wireless communications. *IEEE proceedings of the International Microwave Symposium digest, July 2011*. pp1-4.
- Calcutt, D. & Tetley, L. 1994. *Satellite communications: principles and applications*. London: Edward Arnold.
- Carnahan, J. 2013. CubeSat standard updates: A presentation at the cubesat developers' workshop. Technical notes, California Polytechnic State University, USA.
- Chan, H.A. & Li, Y. 2007. Electronic Communications: Communication networks and systems fundamentals. Class notes, University of Cape Town, South Africa.
- Chattopadhyay, S. & Sanyal, S. K. 2009. Comparison of Performance Metrics for QPSK and OQPSK Transmission Using Root Raised Cosine & Raised Cosine Pulse-shaping Filters for Applications in Mobile Communication. *International Journal of Computer Sciences and Information Security*, 6(2), pp.106-112, .
- Chen, J. 2004. Carrier recovery in burst-mode 16-QAM. MSc thesis, University of Saskatchewan, China.
- Chen, S. No date. Radio Communication Networks and Systems: Carrier and timing recovery techniques. Class notes, University of Southampton, England.

Chitode, J.S. 2009. *Principles of Communication*. 1st ed. Pune (India): Technical Publication Pune.

Clyde Space, 2017. CubeSats products. Website: <https://www.clyde.space/products> [27 July 2017]

Cruz, M.M.C. 2011. *Handbook of Research on Mobility and Computing: Evolving Technologies and Ubiquitous Impacts*. Vol1. United State of America: IGI Global.

Deng, S., Hu, Y. & Sawan, M. 2006. A high data rate QPSK demodulator for inductively powered electronics implants. *IEEE Proceedings of the ISCAS, 1-3 June 2006*. pp:2577-2280.

Dick, C., Harris, F. & Rice, M. 2004. FPGA implementation of carrier synchronization for QAM receivers. *Journal of VLSI Signal Processing*, 36, pp.57-71.

Edward, A.L. & David, G.M. 1994. *Digital communication*. 2nd ed. Massachussetts: Kluwer academic publisher.

Faruque, S. 2015. *Radio Frequency Propagation Made Easy*. United State of America: Springer International Publisher.

Fitton, M. 2012. Principles of digital modulation. Technical notes, Toshiba telecommunications research lab, France.

Forouzan, B.A. 2007. *Data communication and networking*. 4th ed. New York: McGraw-Hill.

French South African Institute of Technology, 2017. CubeSats products. Website: <http://www.cput.ac.za/blogs/fsati/products/> [27 July 2017]

Gentile, K. 2002. Mixed signal analysis: the care and feeding of digital pulse-shaping filters. Technical notes, RF Design Inc., USA.

Gentile, K. 2007. *Digital pulse shaping filter basics*. Technical notes, Analog Devices, USA

Geza, K., Jako, Z. & Kis, G. 2002. Chotic communication with correlator receivers: Theory and performance limits. *IEEE Proceedings*. 90(5), pp.711-732, May.

Glover, I. & Grant, P. 1998. *Digital communications*. Great Britain: Prentice Hall.

Goldfarb, G. 2008, Digital signal processing techniques for coherent optical communication. PhD thesis, University of Central Florida, USA.

- GOMspace, 2017. CubeSat communication subsystems. Website: <https://gomspace.com/Shop/subsystems/communication/nanocom-sr2000.aspx> [27 July 2017]
- Hart, D. 2000. Satellite communications. Class notes, Ohio State University, USA.
- Hasan, M., Akhter, F., Han Khan, MM. 2010. Design and implementation of a QPSK demodulator. BSc thesis, BRAC University, Bangladesh.
- Haykin, S. 2000. *Communication systems*. 4th ed. Toronto: John Wiley & Sons.
- Haykin, S. 2014. *Digital communication systems*. USA: John Wiley & Sons.
- Hranac, R. & Currivan, B. Carrier-to-Noise, Signal-to-Noise & Modulation Error Ratio. Technical notes, Cisco Systems Inc., USA.
- Hranac, R. 2001. QPSK and 16-QAM digital modulation. Technical notes, Cisco systems, USA.
- Inovative Solutions In Space, 2017. Communication Systems for CubeSats. Websites: <https://www.isispace.nl/product-category/satellite-products/communication-systems/> [24 July 2017]
- Ippolito, L.J. 2008. *Satellite communications systems engineering: Atmospheric effects, Satellite link design and System performances*. 1st ed. UK: John Wiley & Sons.
- Jain, P.K., Pal, S. & Pandharipande, V.M. 2003. Satellite Communications: Analysis and implementation of carrier recovery process in a multi-mission remote sensing satellite ground station. Technical notes, RF Design, inc., USA.
- Jalali, S. 2012. Wireless channel equalization in digital communication systems. PhD thesis, Claremont Graduate University and California State University, France and USA.
- JE9PEL, 2017. All satellites frequency list update. Website: <http://www.ne.jp/asahi/hamradio/je9pel/satslist.htm>, [31 January 2017].
- Johnson, J. 2011. List and comparison of FPGA companies. Website: <http://www.fpgadeveloper.com/2011/07/list-and-comparison-of-fpga-companies.html>, [14 February 2016]
- Jooste, C., Quibell, J. & Visser, D.F. 2014. CubeSat product user manual: S-band transmitter (STX). Technical notes, Cape Peninsula University of Technology, South Africa.

Kapadia, R.J. 2012. *Digital Filters, Theory, Application and Design of Modern Filters*. Weinheim: Wiley-VCH.

Katiar, M. & Anju. 2012. Performance evaluation of mean square error of Butterworth and Chebyshev filter with Matlab. *International Journal of Engineering Research & Technology*, 1(3), pp.1-5, May.

Keane, J.P. & Hurst, P.J. No date. Timing recovery for the magnetic recording channel using the wave difference method. Unpublished journal article.

<http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.131.4862&rep=rep1&type=pdf>

[1 June 2017]

Ke-Lin, D. & Swamy, M.N.S. 2010. *Wireless communication systems*. New York: Cambridge University press.

Lan, W., Toorian, A., Hutmuranasin, A. & Lee, S. 2007. CubeSat Design Specification, revision 10. Technical notes, California Polytechnic State University, USA.

Larson, W.J. & Wertz, J.R. 1999. *Space mission analysis and design*. 3rd ed. London: Kluwer Academic.

Lathi, B.P. & Ding, D. 2010. *Modulations: digital and analog communication systems*. 4th ed. New York: Oxford University Press, Inc.

Lau, F.C.M., Tse, C.K., Tam, W.M. & Hau, S.F. 2001. Optimum design for correlator-type receivers in chaos-based digital communication systems. *NOLTA2001 Proceedings*, pp. 565-568.

Le-Ngoc, T. & Feher, K. 1980. A digital approach of symbol timing recovery systems. *IEEE Transaction on communications*, 28(12), pp.1994-2000, December.

Lenz, C., Friedman, A., Underhill, B., Ferring, S., Rademacher, J. & Reed, H. 2002. Arizona state university satellite 1 (ASUSat1): Low-cost student-designed nanosatellite. *Journal of spacecraft and rockets*, 39(5), September.

Litwin, L. 2001. Time and frequency analysis of a communication system: Matched Filtering and timing recovery in digital receivers. Technical notes, RF Design, Inc., USA.

Liu, X., Chandrasekhar, S. & Leven, A. 2008. Digital self-coherent detection. *Optical society of America*, 16(2), pp.1-12, January.

Mahlab, U. 2015. Digital Communication: digital carrier modulation schemes. Class notes, Holon Institute of Technology, Israel.

Maini, A.K. & Agrawal, V. 2007. *Satellite technology: principles and applications*. Chichester: John Wiley & Sons.

Makolomakwe, K. 2013. A baseband modem for CubeSat UHF transceiver. MTech thesis, Cape Peninsula University of Technology, South Africa.

Man, C. 2012. Quantization noise: A derivation of the functional equation. Technical notes, Analog Devices, USA.

Maral, G. & Bousquet, M. 2009. *Satellite communications systems: systems, techniques and technology*. 5th ed. Chichester: John Wiley & Sons.

Maral, G., Bousquet, M. 1998. *Satellite communications systems: systems, techniques and technology*. 3rd ed. Chichester: John Wiley & Sons.

McNutt, C.J., Vick, R., Whiting, H. & Luke, J. 2009. *Modular nanosatellites-Plug and pay (PNP) Cubesat*. AIAA-7th responsive space conference, pp.1-14, April.

Memon T.D., Ghangro W., Chowdhry B.S. & Sheikh, A.S. No-date. Quadrature phase shift keying Modulator & Demodulator for wireless modem. Unpublished journal article, Mehran University of Engineering and Technology, Pakistan.

Mentor Graphics Corporation. 2012. ModelSim Tutorial, Software version 10.1c. USA.

Michael, T.J. 2002. A fixed-point phase lock loop in software defined radio. MSc thesis, Naval Postgraduate School, California, USA.

Micheli, D. No date. Digital Modulation techniques. Class Notes, Universita Politecnica delle Marche, Italia.

Microsemi Corporation, 2017. List of Microsemi FPGAs. Website: <https://www.microsemi.com/products/fpga-soc/fpgas> [22 June 2017]

Microsemi Corporation. 2016. IGLOO2 Datasheet. Microsemi Power Matter Inc., USA.

Mishra, A., Sharma, D.K. & Saxena, R. 2010. Analog & digital modulation techniques: an overview. *International Journal of Computing Sciences and Communication Technology*, 3(1), pp.551-561, July.

Mitra, M. 2008. *Satellite communication*. New Delhi: Printice-Hall of india private limited.

Mitra, S.I. 2006. *Digital signal processing: a computer based approach*. 3rd ed. New Delhi: McGraw-Hill.

Mulally, D.J. & Lefevre, D.K. No date. A comparison of digital modulation methods for small satellite data links. Technical notes, Cynetics Corporation, Rapid City.

Munakata, R. 2009. CubeSat Design Specification, revision 12. Technical notes, California Polytechnic State University, USA.

Nicoloso, S.P. 1997. An investigation of carrier recovery techniques for PSK modulated signals in CDMA and multipath mobile environments. MSc thesis, Virginia Polytechnic Institute and State University, USA.

Pasupathy, S. 1979. Minimum shift keying: a spectrally efficient modulation. *IEEE Communications magazine*, July 1979. pp.14-22.

Peebles, P.Z. Jr., 1987. *Digital communication systems*. New Jersey: Prentice-Hall, Inc.

Planet-Labs, 2017. CubeSat Radios. Websites: <https://directory.eoportal.org/web/eoportal/satellite-missions/d/dove> [18 July 2017]; and <https://www.itu.int/en/ITU-R/space/workshops/2015-prague-small-sat/Presentations/Planet-Labs-Safyan.pdf> [18 July 2017]

Popescu S.O., Gontean A.S. & Ianchis D. 2011. Implementation of a QPSK system of FPGA. *IEEE Proceedings of the ISISIS, Subotica, 8-10 September 2011*. pp.365-370.

Pozar, D.M. 2005. *Microwave engineering*. 3rd ed. USA: John Wiley & Sons, Inc.

Pratt, T., Bostian, C. & Alluntt, J. 2003. *Satellites communications*. 2nd ed. USA: John Wiley & Sons.

Proakis, J.G. 1995. *Digital communications*. 3rd ed. Singapore: McGraw-Hill, Inc.

Pursley, M.B. 2005. *Introduction to digital communications*. New Jersey: Prentice Hall

- Rosu, I. 2017. Automatic Gain Control (AGC) in Receivers. Class notes, University POLITEHNICA of Bucharest, Romania.
- Royi, N. 2011. Ultra high frequency (UHF) frequency modulation (FM) telemetry transmitter for CubeSats. MTech thesis, Cape Peninsula University of Technology, South Africa.
- Sackey, E.N.O. 2006. Performance evaluation of M-ary frequency shift keying radio modems via measurements and simulations. MSc thesis, Blekinge Institute of Technology, Sweden.
- Saramaki, T. 1993. *Handbook for digital signal processing*. Finland: John Wiley & Sons, Inc.
- Sarkar, A. 2012. Electronic Communication Engineering: Digital communication, Equalization. Class notes, Jalpaiguri Government Engineering College, Bengal.
- Schlichter, T.J. 1999. Digital filter design using Matlab. MSc thesis, Mississippi State University, USA.
- Selva, D. & Krejci, D. 2011. A survey and assessment of the capabilities of CubeSats for Earth observation. *Elsevier-Acta Astronautica*, 74, pp.50-68, December.
- Simon, M.K. 2001. Bandwidth-efficient digital modulation with application to deep-space communications. Technical report, Jet Propulsion Laboratory, California Institute of Technology, USA.
- Singarayar, F., Reinhard, R., Asma, C., Bernal, C., Weggelaar, W. & Kataria, D. 2013. The QB50 mission: system requirements and recommendations. Issue 3. Technical notes.
- Sklar, B. 2008. *Digital communications: fundamentals and applications*. London: Prentice Hall International.
- Song, W. & Yao, Q. no date. Design and implementation of QPSK modem based on FPGA. Unpublished journal article, North China Electric Power University, China.
- Spalvieri, A. 2013. Digital communications: An overview of timing recovery. Class notes, Polytecnico di Milano, Italia.
- Spire, 2017. CubeSat missions. Website: <http://spacenews.com/spire-40-cubesats-in-orbit-competing-more-directly-in-space-based-ship-tracking-market/> [18 July 2017]

- Sreekanth, P.V. 2003. *Digital microwave communication systems: with selected topics in mobile communications*. India: University Press Private Limited.
- Stark, J., Fortescue, P. & Swinerd, G. 2003. *Space systems engineering*. 3rd ed. Chichester: John Wiley & Sons.
- Steber, J.M. 2001. Digital Communication: PSK demodulation, Part1. Technical notes, WJ Communications Inc., USA.
- Steven, J.O. 2011. Modular FPGA-BASED Software Defined Radio for Cubesats. MSc thesis, Worcester Polytechnic Institute, United State of America.
- Stevens, A.E. 1995. An integrate-and-dump receiver for fiber optic networks. PhD thesis, Graduate School of Art and Sciences, Columbia University, USA.
- Stewartwout, M. 2017. *All CubeSat launched up to January 2015*. Website: <https://sites.google.com/a/slu.edu/swartwout/home/cubesat-database> [30 August 2017].
- Syrlinks, 2017. CubeSat transmitter and receiver. Website: <http://www.syrlinks.com/en/space/hdr-transmitters.html> [24 July 2017]
- Tao, M. 2013. Principles of communications: digital modulation techniques. Class notes, Shanghai Jiao Tong University, Shanghai.
- Tatu, S.O., Moldovan, E. & Wu, K. 2002. A new carrier recovery method for a six-port millimetre wave receiver. *IEEE Transaction on Microwave Theory and Techniques*, 50(11), pp.2436-2442, November.
- Tether Unlimited Inc., 2017. Software Defined Radios for CubeSats. Website: <http://www.tethers.com/SWIFT.html> [24 July 2017]
- Thede, L. 1996. *Analog and digital filter design using C*. New Jersey: Prentice Hall.
- Tipper, D. 2015. Digital Communications: Digital modulation. Class notes, University of Pittsburgh, USA.
- Torlak, M. 2013. Telecommunication Switching and Transmission: Introduction to digital modulation. Class notes, University of Texas at Dallas, USA.

- Torres, V., Perez-Pcual, A., Sansaloni, T. & Valls, J. 2006. *Design of high performance timing recovery loops for communication applications. IEEE Proceedings of the Workshop on signal processing systems design and implementation.* pp:1-4.
- Tytgat, M., Steyaert, M. & Reynaert, P. No date. Time domain model for Costas loop based QPSK receiver. Unpublished journal article, K.U Leuven ESAT-MICAS, Belgium.
- Vadhavkar, P.R. 2007. Mapping controllers from the s-domain to the z-domain using magnitude invariance and phase invariance methods. MSc thesis, Wichita State University, Pune, India.
- Vertat, I. & Mraz, J. 2013. Hybrid M-FSK/DQPSK Modulations for CubeSat Picosatellites. *Radioengineering journal*, 22(1), pp.389-393, April.
- Wang, J. 2002. Timing recovery techniques for digital recording systems. PhD thesis, National University of Singapore, Singapore.
- Webber, J. & Dahnoun, N. 1996. Implementing a $\pi/4$ shift D-QPSK baseband modem using TMS320C50. Technical Notes, Texas Instrument & ESIEE-Paris, France.
- Widrow, B., Kollar, I. & Lui, M.C. 1996. Statistical theory of quantization. *IEEE Transaction Information Theory*, pp.353-361, April.
- Wong, T. & Lok, J. 2014. Digital Communication: theory of digital communications. Class notes, University of Florida, USA.
- Yang, C.S., Kao S.P., Lee, F.B. & Hung, P.S. No date. Twelve different interpolation methods: a case study of surfer 8.0. Unpublished journal article.
- Yoshida, S. 1995. A digital coherent receiver suitable for land-mobile satellite communications. *IEEE transaction on vehicular technology*, 44(4), pp.771-778, November.
- Zafar, A. & Farooq, S.Z. 2008. Implementation and analysis of QPSK and 16GAM Modulator & Demodulator. *IEEE Proceedings of the ICAST, Pakistan, 29-30 November 2008.* pp:64-68.

APPENDICES

Appendix-A:

The Matlab Code of the QPSK demodulator

```
%% *****
%
%           Name: Joel Scientific BIYOGHE
%           Type: QPSK Demodulator
%
% NB: THIS SCRIPT ONLY PRESENTS A SIMPLIFIED VERSION OF THE "TOP-LEVEL-CODE"
%     ie. THE CODE OF THE TOP-BLOCKS OF THE QPSK DEMODULATOR. THE CODES FOR
%     ALL THE SUB-PROCESSES ARE NOT PRESENTED DUE TO SPACE IN THIS DOCUMENT,
%     AND ARE GIVEN IN THE CD-ROM OF THIS PROJECT!
%
% *****

while ((sample + sample_offset_pts) <= total_simulation_pts)
    %% ****THE ADC PROCESS PROCESS****
    data = qpsk_sampled_data(sample+sample_offset_pts); %the "sampling_offset"
    %% THE LCG PROCESS
    i_car = i_lo(count_in_symbol);
    q_car = q_lo(count_in_symbol);
    %% *****THE CORRELATOR*****
    %% THE MIXERS PROCESS
    i_bpsk = data*i_car;
    q_bpsk = data*q_car;
    %SCALL DOWN THE I/Q MIXER DATA
    scaling_factor = scal_factor_IQ;
    value_to_scall_down = i_bpsk;
    scaled_i_bpsk =
Scaling_Down_function(scaling_factor,value_to_scall_down);
    value_to_scall_down = q_bpsk;
    scaled_q_bpsk =
Scaling_Down_function(scaling_factor,value_to_scall_down);
    %% THE MATCHED FILTER PROCESS
    Xin = scaled_i_bpsk;
    for k = 1:length(h_coef_mf)
        interm_sums_mf_i(k) = h_coef_mf(k)*Xin + interm_sums_mf_i(k+1);
    end
    mfi_scal = interm_sums_mf_i(1);
    Xin = scaled_q_bpsk; %scaled_af_q; % AF_output_q; %
    for k = 1:length(h_coef_mf)
        interm_sums_mf_q(k) = h_coef_mf(k)*Xin + interm_sums_mf_q(k+1);
    end
    mfq_scal = interm_sums_mf_q(1);
    %SCALL DOWN THE MF FILTERED DATA
    scaling_factor = scal_factor_MF;
    value_to_scall_down = mfi_scal;
    scaled_mf_i = Scaling_Down_function(scaling_factor,value_to_scall_down);
    value_to_scall_down = mfq_scal;
    scaled_mf_q = Scaling_Down_function(scaling_factor,value_to_scall_down);

    %% *****TIMING RECOVERY SYSTEM (TRS)*****
    %% THE TED PROCESS
    %%1st clk
    %Define the new "cap-time" and the two adjacent sample pts.
```

```

if(n==1)
    if((trs_nco_out_cap_adr - symb_length/2)== -16)
        b = -16 +1;
    else
        b = trs_nco_out_cap_adr - symb_length/2;
    end
    a = trs_nco_out_cap_adr + symb_length/2;
    r = trs_nco_out_cap_adr;
end
%Fill the TRS array
if(n==1)
    %transfer the new symbol stored into the search-array "mfis_2symb"
    mfis_2symb(1:16) = mfis_2symb(17:32);
    mfis_2symb(17:32)= mfis_symb;
    mfis_symb(n) = scaled_mf_i;
    mfqs_2symb(1:16) = mfqs_2symb(17:32);
    mfqs_2symb(17:32)= mfqs_symb;
    mfqs_symb(n) = scaled_mf_q;
    n=n+1;
elseif(n==16)    %bring "n" back to 1 when it reaches 16
    mfis_symb(n) = scaled_mf_i;
    mfqs_symb(n) = scaled_mf_q;
    n = 1;
else
    mfis_symb(n) = scaled_mf_i;
    mfqs_symb(n) = scaled_mf_q;
    n=n+1;
end
%%2nd clk
%Capture the three samples
if(n==2)
    xi_r = mfis_2symb(r + symb_length); %NB: + symb_length because 17-32
    xi_b = mfis_2symb(b + symb_length); %NB: +1 to avoid case of 0
    xi_a = mfis_2symb(a + symb_length); %NB: -1 to avoid case of 33
    xq_r = mfqs_2symb(r + symb_length);
    xq_b = mfqs_2symb(b + symb_length);
    xq_a = mfqs_2symb(a + symb_length);
end
%%3rd clk
%Upper limit max value MF of samples to "peak_max_acceptable"
if(n==3)
    if(xi_b>+peak_max_acceptable)
        xi_b_lim = +peak_max_acceptable;
    elseif(xi_b<-peak_max_acceptable)
        xi_b_lim = -peak_max_acceptable;
    else
        xi_b_lim = xi_b;
    end
    if(xi_a>+peak_max_acceptable)
        xi_a_lim = +peak_max_acceptable;
    elseif(xi_a<-peak_max_acceptable)
        xi_a_lim = -peak_max_acceptable;
    else
        xi_a_lim = xi_a;
    end
    if(xi_r>+peak_max_acceptable)
        xi_r_lim = +peak_max_acceptable;

```



```

elseif(xi_r<-peak_max_acceptable)
    xi_r_lim = -peak_max_acceptable;
else
    xi_r_lim = xi_r;
end
end
%Det sign() of xi_r
if(n==3)
    if(xi_r>0)
        xi_r_signum = +1;
    elseif(xi_r<0)
        xi_r_signum = -1;
    else
        xi_r_signum = 0;
    end
end
end
%%4th clk
%Do the E/L Algo
if(n==4)
    if(xi_a_lim>0 && xi_b_lim>0)                %Case of 2 positive bits
        err_el = 0;
    elseif(xi_a_lim >two_bits_id_lim && xi_r_lim >two_bits_id_lim) %2+ve
        err_el = 0;
    elseif(xi_b_lim >two_bits_id_lim && xi_r_lim >two_bits_id_lim) %2+ve
        err_el = 0;
    elseif(xi_a_lim<0 && xi_b_lim<0)                %2-ve
        err_el = 0;
    elseif(xi_a_lim <-two_bits_id_lim && xi_r_lim <-two_bits_id_lim)%2-ve
        err_el = 0;
    elseif(xi_b_lim <-two_bits_id_lim && xi_r_lim <-two_bits_id_lim)%2-ve
        err_el = 0;
    else                %Case of 2 transition bits
        err_el = xi_r_signum*(xi_a_lim - xi_b_lim);
    end
end
end
%% THE LF PROCESS FOR THE TRS
if(n==5)
    trs_lp_out = (k1_trs*err_el) + (k2_trs*(err_el + lp_2_trs));
    lp_2_trs = err_el + lp_2_trs;
end
%SCALL DOWN THE LF INTEGERS DATA
if(n==6)
    scalling_factor = scal_factor_LF_TRS;
    value_to_scall_down = trs_lp_out;
    scaled_lp_trs
Scaling_Down_function(scalling_factor,value_to_scall_down);
end
%% THE TRS NCO PROCESS
% THE INTEGRATION AND LIMIT OUTPUT TRS-NCO INT TO 8192=8ko-trs = 8*1024!
if(n==7)
    if(trs_nco_integrator_out >= 8192 && scaled_lp_trs > 0)
        trs_nco_integrator_out = 8192;
    elseif(trs_nco_integrator_out <= -8192 && scaled_lp_trs < 0)
        trs_nco_integrator_out = -8192;
    else
        trs_nco_integrator_out = scaled_lp_trs + trs_nco_integrator_out;
    end
end

```

```

end
% NCO GAIN BLOCK "PHASE NCO DETERMIANTION"
if(n==8)
    a_trs = ko_trs_num*trs_nco_integrator_out;
    b_trs = ko_trs_den;
    c_trs = rem(a_trs,b_trs);
    d_trs = a_trs-c_trs;
    trs_nco_gain_out = d_trs/b_trs; % aka: phase_dif_nco_trs;
end
%TRANSFERT OF INFO FROM TRS TO DDB B4 END-OF-CURRENT-SYMBOL
if(n==14)
    if(trs_nco_gain_out>8)
        trs_nco_out_cap_adr = +8;
    elseif(trs_nco_gain_out<-8)
        trs_nco_out_cap_adr = -8;
    else
        trs_nco_out_cap_adr = trs_nco_gain_out;
    end

    activate_ddm_detection=1;
    sample_to_ddb_i = xi_r;
    sample_to_ddb_q = xq_r;
else
    activate_ddm_detection=0;
end

%% *****THE DETECTION AND DECISION BLOCK (DDB)*****
%% Detection Block
if(activate_ddm_detection==1) %THIS WILL HAPPEN WHEN "n=15"
    peak_adress = trs_nco_out_cap_adr;
    peak_amplitude_i = sample_to_ddb_i;
    peak_amplitude_q = sample_to_ddb_q;
    activate_ddm_decision = 1;
else
    activate_ddm_decision = 0;
end
%% Decision Block %THIS WILL HAPPEN WHEN "n=16"
if(activate_ddm_decision==1)
    %make the decision
    if(peak_amplitude_i > comp_lim)
        bit_i = +1;
    elseif(peak_amplitude_i < -comp_lim)
        bit_i = -1;
    else
        bit_i = 0;
    end
    if(peak_amplitude_q > comp_lim)
        bit_q = +1;
    elseif(peak_amplitude_q < -comp_lim)
        bit_q = -1;
    else
        bit_q = 0;
    end
end
end

```

```

%% *****THE CARRIER RECOVERY SYSTEM (CRS)*****
%% THE PED "COSTAS'ERROR COMPUTATION ALGORITHM"
%The Limiter's output in the I-channel and Q-channel.
lim_out_i_scall = sign(scaled_mf_i_crs);
lim_out_q_scall = sign(scaled_mf_q_crs);
%Erreur on I-channel and Q-channel
erreur_i_scall = lim_out_i_scall*scaled_mf_q_crs;
erreur_q_scall = lim_out_q_scall*scaled_mf_i_crs;
erreur_g_scall = erreur_i_scall - erreur_q_scall;
%% THE LF PROCESS
lp_in = erreur_g_scall;
lp_1_crs = lp_in;
lp_2_crs = lp_in + lp_2_crs;
lp_out = k1_crs*lp_1_crs + k2_crs*lp_2_crs;
%SCALL DOWN THE LPF FILTERED DATA
scalling_factor = scal_factor_LF_CRS;
value_to_scall_down = lp_out;
scaled_lp = Scalling_Down_function(scalling_factor,value_to_scall_down);
%% THE NUMERICALLY CONTROLLED OSCILATOR (NCO)
% THE INTEGRATION
nco_in = scaled_lp;
nco_integrator_out = nco_in + nco_integrator_out;
nco_integrator_out_array(sample) = nco_integrator_out;
% THE NCO GAIN BLOCK "PHASE NCO DETERMIANTION"
a = ko_crs_num*nco_integrator_out;
b = ko_crs_den;
c = rem(a,b);
d = a-c;
e = d/b;
nco_gain_out = e; % aka: phase_dif_nco;
% THE NCO CARRIER GENERATION PROCESS AFTER Z-1.
if(count_in_symbol == symb_length) %here we aquire the new cycles
    phase_nco = - nco_gain_out; %NB: THE (-) IS DUE TO (FEEDBACK).

    lo_cos_sin =
nco_function(lut_coefs,phase_nco,fsa,adc_samp_rate,modulation_factor,num_symb_
nco_at_time);
    i_lo = lo_cos_sin(1,:);
    q_lo = lo_cos_sin(2,:);

    %register the generated cycles in an array "JUST FOR DISPLAY".
    lo_cos(((symb-1)*symb_length +1): symb*symb_length) = i_lo;
    lo_sin(((symb-1)*symb_length +1): symb*symb_length) = q_lo;
    %%SETUP FOR NEXT SAMPLE ITERATION
    symb
    phase_nco
    count_in_symbol = 1;
    symb = symb+1;
else %here the generated cycles are used, till whole symbol is processed
    i_lo = i_lo;
    q_lo = q_lo;
    count_in_symbol = count_in_symbol + 1;
end
sample = sample+1;
end

```

Appendix-B: The VHDL Code of the QPSK demodulator

```

-----
-- NB: THIS SCHIPT ONLY PRESENTS A SIMPLIFIED VERSION OF THE "TOP-LEVEL-CODE" --
-- ie. THE CODE OF THE TOP-BLOCKS OF THE QPSK DEMODULATOR. THE CODES FOR --
-- ALL THE SUB-PROCESSES ARE NOT PRESENTED DUE TO SPACE IN THIS DOCUMENT, --
-- AND ARE GIVEN IN THE CD-ROM OF THIS PROJECT --
-----

```

```

-----
--                               Code Developed by: Joel Scientifiq BIYOGHE --
--                               Code Type:          QPSK Demodulator --
--                               Device:            IGLOO2 FPGA --
--                               Language:         VHDL --
-----

```

```

library IEEE;
use IEEE.std_logic_1164.all;
use ieee.numeric_std.all;

```

```
entity QPSK_DEMOD is
```

```

  generic(
    symb_length      : integer := 16;
    qpsk_max         : integer := 15; --
    lo_max           : integer := 63; -- ie: lut coefs max, as specified on matlab.
    phase_max        : integer := 360; -- ie max phase is in fact it is +/-80°.
    MF_scaling_out_max : integer := 127; -- ie: "mf-scaling-block output-max".
    NCO_out_max      : integer := 32 -- ie: for ko = 4096/LF_SF. -32 to +32
  );

```

```

  port(
    clk, rst      : in std_logic;
    qpsk_input    : in std_logic_vector(4 downto 0); -- qpsk-data: ie: 5bits(4 + 1);
    phase_nco_disp : out std_logic_vector(9 downto 0);
    lo_cos_disp   : out std_logic_vector(6 downto 0); -- LO_max = 63, ie: 7=(6+1)
    lo_sin_disp   : out std_logic_vector(6 downto 0); -- LO_max = 63, ie: 7=(6+1)
    MF_bb_out_disp_i : out std_logic_vector(7 downto 0); -- MF_scaling_out_max = 127,
    MF_bb_out_disp_q : out std_logic_vector(7 downto 0); -- MF_scaling_out_max = 127,
    trs_nco_cap_time_disp : out std_logic_vector(5 downto 0); -- NCO_out_max = +/-32 => 6,
    trs_clock_disp  : out std_logic;
    MF_bb_cap_i_disp : out std_logic_vector(7 downto 0);
    MF_bb_cap_q_disp : out std_logic_vector(7 downto 0);
    bit_i_disp      : out std_logic_vector(1 downto 0);
    bit_q_disp      : out std_logic_vector(1 downto 0)
  );

```

```
end QPSK_DEMOD;
```

```
architecture qpsk_demodualation of QPSK_DEMOD is
```

```
--SUBTYPES
```

```
  subtype range_std_logic is std_logic;
```

```
  subtype range_phs_nco is integer range -phase_max to +phase_max;
```

```

subtype range_qpsk_data is std_logic_vector(4 downto 0);
subtype range_lo_data is integer range -LO_max to +LO_max;
subtype range_scaled_MF_data is integer range -MF_scaling_out_max to
+MF_scaling_out_max;
subtype range_nco_capture_time is integer range -NCO_out_max to +NCO_out_max;
subtype range_signed_binary is integer range -1 to +1;

--SIGNALS
signal phase_nco_per_symb_signal : range_phs_nco;
signal lo_cosinus,lo_sinus : range_lo_data; --
signal MF_i_bb_signal : range_scaled_MF_data;
signal MF_q_bb_signal : range_scaled_MF_data;
signal trs_nco_cap_time_signal : range_nco_capture_time;
signal trs_clock_signal : range_std_logic;
signal activate_ddb_signal : range_std_logic;
signal MF_bb_captured_i_signal : range_scaled_MF_data;
signal MF_bb_captured_q_signal : range_scaled_MF_data;

signal bit_i_signal : range_signed_binary;
signal bit_q_signal : range_signed_binary;

--COMPONENTS
component CAR_RECOV_SYSTEM
port(
    clk,rst : in range_std_logic;
    qpsk_input : in range_qpsk_data;
    phase_nco_per_symb : out range_phs_nco; -- Just for display
    lo_cos : out range_lo_data; -- For use in the cor
    lo_sin : out range_lo_data; -- For use in the cor
);
end component;

component CORRELATOR
port(
    rst,clk : in range_std_logic;
    qpsk_input : in range_qpsk_data;
    lo_input : in range_lo_data;
    MF_bb_output : out range_scaled_MF_data
);
end component;

component TIM_RECOV_SYSTEM
port(
    clk,rst : in range_std_logic;
    MF_bb_i : in range_scaled_MF_data;
    MF_bb_q : in range_scaled_MF_data;
    trs_nco_cap_time : out range_nco_capture_time;
    trs_clock : out range_std_logic;
    activate_ddb : out range_std_logic;
    MF_bb_captured_i : out range_scaled_MF_data;
    MF_bb_captured_q : out range_scaled_MF_data
);
end component;

```

```

    );
end component;

component DET_DEC_BLOCK
port(
    clk, rst                : in range_std_logic;
    MF_bb_i                 : in range_scaled_MF_data; --from cor (will not be used here)
    MF_bb_q                 : in range_scaled_MF_data; --from cor (will not be used here)
    trs_nco_cap_time       : in range_nco_capture_time; --from trs (will not be used),
    MF_bb_i_cap            : in range_scaled_MF_data;
    MF_bb_q_cap            : in range_scaled_MF_data;
    activate_ddb           : in range_std_logic;
    bit_i                   : out range_signed_binary;
    bit_q                   : out range_signed_binary
);
end component;

```

```

begin
LOCAL_COS_SIN_CAR      : CAR_RECOV_SYSTEM
port map(
    rst                => rst,
    clk                => clk,
    qpsk_input         => qpsk_input,
    phase_nco_per_symb => phase_nco_per_symb_signal, --just for display
    lo_cos             => lo_cosinus,                --for use in cor
    lo_sin             => lo_sinus,                  --for use in cor
);

    phase_nco_disp <= std_logic_vector(to_signed(phase_nco_per_symb_signal,10));
    lo_cos_disp    <= std_logic_vector(to_signed(lo_cosinus,7)); --for display
    lo_sin_disp    <= std_logic_vector(to_signed(lo_sinus,7)); --for display

```

```

CORRELATOR_I          : CORRELATOR
port map(
    rst                => rst,
    clk                => clk,
    qpsk_input         => qpsk_input,
    lo_input           => lo_cosinus,
    MF_bb_output       => MF_i_bb_signal
);

```

```

CORRELATOR_Q          : CORRELATOR
port map(
    rst                => rst,
    clk                => clk,
    qpsk_input         => qpsk_input,
    lo_input           => lo_sinus,
    MF_bb_output       => MF_q_bb_signal
);

```

```

MF_bb_out_disp_i <= std_logic_vector(to_signed(MF_i_bb_signal,8)); -- for display

```

```
MF_bb_out_disp_q <= std_logic_vector(to_signed(MF_q_bb_signal,8)); -- for display
```

```
TIM_REC_SYS : TIM_RECOV_SYSTEM
```

```
port map(
```

```
clk           => clk,  
rst           => rst,  
MF_bb_i       => MF_i_bb_signal,  
MF_bb_q       => MF_q_bb_signal,  
trs_nco_cap_time => trs_nco_cap_time_signal, -- to ddb and for display  
trs_clock     => trs_clock_signal,  
activate_ddb  => activate_ddb_signal, -- to ddb  
MF_bb_captured_i => MF_bb_captured_i_signal, -- to ddb and for display  
MF_bb_captured_q => MF_bb_captured_q_signal -- to ddb and for display  
);
```

```
trs_nco_cap_time_disp <= std_logic_vector(to_unsigned(trs_nco_cap_time_signal,6));  
trs_clock_disp        <= trs_clock_signal;  
MF_bb_cap_i_disp     <= std_logic_vector(to_signed(MF_bb_captured_i_signal,8));  
MF_bb_cap_q_disp     <= std_logic_vector(to_signed(MF_bb_captured_q_signal,8));
```

```
DET_DEC : DET_DEC_BLOCK
```

```
port map(
```

```
clk           => clk,  
rst           => rst,  
MF_bb_i       => MF_i_bb_signal,  
MF_bb_q       => MF_q_bb_signal,  
trs_nco_cap_time => trs_nco_cap_time_signal,  
activate_ddb  => activate_ddb_signal,  
MF_bb_i_cap   => MF_bb_captured_i_signal,  
MF_bb_q_cap   => MF_bb_captured_q_signal,  
bit_i         => bit_i_signal,  
bit_q         => bit_q_signal  
);
```

```
bit_i_disp <= std_logic_vector(to_signed(bit_i_signal,2)); --for display  
bit_q_disp <= std_logic_vector(to_signed(bit_q_signal,2)); --for display
```

```
end qpsk_demodulation;
```

Appendix-C: The System's Resource Analysis Reports

The screenshot shows the Libero IDE interface with the 'Reports' window open, displaying the 'Resource Usage' report for the 'QPSK_DEMOD' project. The report is organized into a tree view on the left and a main content area on the right. The main content area contains two tables: 'Resource Usage' and 'Detailed Logic Resource Usage'.

Resource Usage

Type	Used	Total	Percentage
4LUT	6284	12084	52.00
DFF	3960	12084	32.77
I/O Register	0	693	0.00
User I/O	73	231	31.60
-- Single-ended I/O	73	231	31.60
-- Differential I/O Pairs	0	115	0.00
RAM64x18	0	22	0.00
RAM1K18	0	21	0.00
MACC	20	22	90.91
Chip Global	2	8	25.00
CCC	0	2	0.00
RCOSC_25_50MHZ	0	1	0.00
RCOSC_1MHZ	0	1	0.00
XTLOSC	0	1	0.00
SERDESIF Blocks	0	1	0.00
-- SERDESIF Lanes	0	4	0.00
HPMS	0	1	0.00

Detailed Logic Resource Usage

Type	4LUT	DFF
Fabric Logic	5564	3240
RAM64x18 Interface Logic	0	0

Libero - F:\Desktop\Joel_MTech_2014\5_Vhdl_Work_Chap5\Vhdl_Implementation\QPSK_DEM\QPSK_DEM.prjx

Project File Edit View Design Tools Help

Design Flow

QPSK_DEMOD

Tool

- Create Design
 - System Builder
 - Create SmartDesign
 - Create HDL
 - Create SmartDesign Testbench
 - Create HDL Testbench
- Verify Pre-Synthesized Design
- Create Constraints
 - I/O Constraints
 - Timing Constraints
 - Floorplan Constraints
- Implement Design
- Synthesize
 - Verify Post-Synthesis Implementation
 - Simulate
 - Compile
 - Configure Flash*Freeze
 - Place and Route
 - Edit Constraints
 - I/O Constraints
 - Timing Constraints
 - Floorplan Constraints
 - Verify Post Layout Implementation
 - Generate Back Annotated Files
 - Simulate
 - Verify Timing
 - Verify Power
 - Edit Design Hardware Configuration
 - Programming Connectivity and Interface
 - Programmer Settings
 - Device I/O States During Programming
 - Configure Security and Programming Opti...

Reports

QPSK_DEMOD reports

- QPSK_DEMOD
 - QPSK_DEMOD_pinrpt...
 - QPSK_DEMOD_pinrpt...
 - QPSK_DEMOD_bankrpt...
 - QPSK_DEMOD_jeff.xml
- Synthesize
 - synplifylog
 - QPSK_DEMOD.ssr
 - run_options.txt
- Compile
 - QPSK_DEMOD_rwneti...
 - QPSK_DEMOD_co...
 - QPSK_DEMOD_compil...
 - QPSK_DEMOD_combin...
- Place and Route
 - QPSK_DEMOD_glb_net...
 - QPSK_DEMOD_lay...
- Verify Timing
 - QPSK_DEMOD_maxdel...
 - QPSK_DEMOD_mindel...
 - QPSK_DEMOD_maxdel...
 - QPSK_DEMOD_mindel...

Timing Report Max Delay Analysis

SmartTime Version 11.4.0.112
Microsemi Corporation - Microsemi Libero Software Release v11.4 (Version 11.4.0.112)
Date: Mon Jan 09 01:27:03 2017

Design	QPSK_DEMOD
Family	IGLOO2
Die	M2GL010T
Package	484 FBGA
Temperature	COM
Voltage	COM
Speed Grade	STD
Design State	Post-Layout
Data source	Production
Min Operating Condition	BEST
Max Operating Condition	WORST
Scenario for Timing Analysis	Primary

Summary

Clock Domain	Period (ns)	Frequency (MHz)	Required Period (ns)	Required Frequency (MHz)	External Setup (ns)	External Hold (ns)	Min Clock-To-Out (ns)	Max Clock-To-Out (ns)
clk	9.458	105.731	N/A	N/A	3.589	-0.190	4.173	10.205

	Min Delay (ns)	Max Delay (ns)
Input to Output	N/A	N/A

Messages Errors Warnings Info

01:29 AM 2017/01/09

Libero - F:\Desktop\Joel_MTech_2014\5_Vhdl_Work_Chap5\Vhdl_Implementation\QPSK_DEM\QPSK_DEM.prjx

Project File Edit View Design Tools Help

Design Flow

QPSK_DEMOD

Tool

- Create Design
 - System Builder
 - Create SmartDesign
 - Create HDL
 - Create SmartDesign Testbench
 - Create HDL Testbench
- Verify Pre-Synthesized Design
- Create Constraints
 - I/O Constraints
 - Timing Constraints
 - Floorplan Constraints
- Implement Design
- Synthesize
 - Verify Post-Synthesis Implementation
 - Simulate
 - Compile
 - Configure Flash*Freeze
 - Place and Route
 - Edit Constraints
 - I/O Constraints
 - Timing Constraints
 - Floorplan Constraints
 - Verify Post Layout Implementation
 - Generate Back Annotated Files
 - Simulate
 - Verify Timing
 - Verify Power
 - Edit Design Hardware Configuration
 - Programming Connectivity and Interface
 - Programmer Settings
 - Device I/O States During Programming
 - Configure Security and Programming Opti...

Reports

QPSK_DEMOD reports

- Synthesize
 - synplifylog
 - QPSK_DEMOD.ssr
 - run_options.txt
- Compile
 - QPSK_DEMOD_rwneti...
 - QPSK_DEMOD_co...
 - QPSK_DEMOD_compil...
 - QPSK_DEMOD_combin...
- Place and Route
 - QPSK_DEMOD_glb_net...
 - QPSK_DEMOD_lay...
- Verify Timing
 - QPSK_DEMOD_maxdel...
 - QPSK_DEMOD_mindel...
 - QPSK_DEMOD_maxdel...
 - QPSK_DEMOD_mindel...
- Verify Power
 - QPSK_DEMOD_power...

Power Report for design QPSK_DEMOD with the following settings:

Vendor:	Microsemi Corporation
Program:	Microsemi Libero Software, Release v11.4 (Version 11.4.0.112)
Copyright (C):	1989-
Date:	Mon Jan 09 01:33:31 2017
Version:	3.0

Design:	QPSK_DEMOD
Family:	IGLOO2
Die:	M2GL010T
Package:	484 FBGA
Temperature Range:	COM
Voltage Range:	COM
Operating Conditions:	Typical
Operating Mode:	Active
Process:	Typical
Data Source:	Preliminary

Power Summary

	Power (mW)	Percentage
Total Power	14.060	100.0%
Static Power	14.060	100.0%
Dynamic Power	0.000	0.0%

Messages Errors Warnings Info

01:35 AM 2017/01/09

Appendix-D:
The Test-Bench for emulation of the FPGA-based QPSK demodulator

```
-----
--                               Code Developed by: Joel Scientific BIYOGHE           --
--                               Code Type: TEST BENCH FOR THE QPSK Demodulator       --
--                               Device: IGLOO2 FPGA, Die: M2GL010T, Pacakge: 484FBGA  --
--                               Language:          VHDL                             --
-----
```

```
library ieee;
library std;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
use ieee.numeric_std.ALL;
use std.textio.all;
--use ieee.std_logic_textio.all;
```

```
entity QPSK_DEMOD_TB is
end QPSK_DEMOD_TB;
```

```
architecture behavioral of QPSK_DEMOD_TB is
```

```
--Component
```

```
component QPSK_DEMOD
```

```
-- ports
```

```
port(
```

```
-- Inputs
```

```
clk           : in std_logic;
rst           : in std_logic;
qpsk_input   : in std_logic_vector(4 downto 0);
```

```
-- Outputs
```

```
phase_nco_disp      : out std_logic_vector(9 downto 0);
lo_cos_disp         : out std_logic_vector(6 downto 0);
lo_sin_disp         : out std_logic_vector(6 downto 0);
MF_bb_out_disp_i    : out std_logic_vector(7 downto 0);
MF_bb_out_disp_q    : out std_logic_vector(7 downto 0);
trs_nco_cap_time_disp : out std_logic_vector(5 downto 0);
trs_clock_disp      : out std_logic;
MF_bb_cap_i_disp    : out std_logic_vector(7 downto 0);
MF_bb_cap_q_disp    : out std_logic_vector(7 downto 0);
bit_i_disp          : out std_logic_vector(1 downto 0);
bit_q_disp          : out std_logic_vector(1 downto 0);
);
```

```
end component;
```

```
--constants and signals
```

```
constant CLK_PERIOD : time := 12.5 ns;
```

```
constant No_symb_pro: integer := 2000026; -- should be = core_tx + crs_trs_fe + 0_data
```

```

constant symb_length: integer := 16;

signal CLK          : std_logic := '0';
signal RST          : std_logic := '0';
signal QPSK_INPUT   : std_logic_vector(4 downto 0) := (others => '0');
signal PHASE_NCO_DISP : std_logic_vector(9 downto 0);-- := (others => '0');
signal LO_COS_DISP  : std_logic_vector(6 downto 0);-- := (others => '0');
signal LO_SIN_DISP  : std_logic_vector(6 downto 0);-- := (others => '0');
signal MF_BB_OUT_DISP_I : std_logic_vector(7 downto 0);--
signal MF_BB_OUT_DISP_Q : std_logic_vector(7 downto 0);--
signal TRS_NCO_CAP_TIME_DISP : std_logic_vector(5 downto 0);
signal TRS_CLOCK_DISP : std_logic;
signal MF_BB_CAP_I_DISP : std_logic_vector(7 downto 0);
signal MF_BB_CAP_Q_DISP : std_logic_vector(7 downto 0);
signal BIT_I_DISP      : std_logic_vector(1 downto 0);
signal BIT_Q_DISP      : std_logic_vector(1 downto 0);

file          file_in_qpsk:text          open          read_mode          is
"C:\Joel_MTech\5_Vhdl_Work_Chap5\Matlab_QpskSamGen_FurBitPro\qpsk_samples.txt";

begin

--Define "reset" input
process
    variable vhdl_initial : BOOLEAN := TRUE;
begin
    if ( vhdl_initial ) then
        -- Assert Reset
        RST <= '0';
        wait for (CLK_PERIOD*10);

        RST <= '1';
        wait;
    end if;
end process;

-- Define the "clk" input :10MHz Clock Driver
CLK <= not CLK after(CLK_PERIOD/2.0);

--Define the "in_data" input data process: the input comes sample-per sample!
process
    variable IN_LINE_QPSK : line;
    variable data_in_qpsk : integer range -15 to +15;
begin
    wait for (CLK_PERIOD*(10)); -- wait the same time rest is 0.

    for i in 1 to (No_symb_pro*symb_length) loop
        wait until rising_edge(CLK);

        --inputing process
        readline(file_in_qpsk,IN_LINE_QPSK); --get the line number from the data file "file-in"
    end loop;
end process;

```

```
    read(IN_LINE_QPSK,data_in_qpsk); -- "input data" extracts the input-data in the line.  
    QPSK_INPUT <= std_logic_vector(to_signed(data_in_qpsk,5)); -- the input-data  
obtained is transfer to the signal "IN_DATA" to go to input port.
```

```
end loop;  
end process;
```

```
-- Instantiate Unit Under Test: QPSK_DEMODULATOR
```

```
QPSK_DEMOD_0 : QPSK_DEMOD
```

```
-- port map
```

```
port map
```

```
(
```

```
-- Inputs
```

```
clk      => CLK,
```

```
rst      => RST,
```

```
qpsk_input => QPSK_INPUT,
```

```
-- Outputs
```

```
phase_nco_disp => PHASE_NCO_DISP,
```

```
lo_cos_disp    => LO_COS_DISP,
```

```
lo_sin_disp    => LO_SIN_DISP,
```

```
MF_bb_out_disp_i => MF_BB_OUT_DISP_I,
```

```
MF_bb_out_disp_q => MF_BB_OUT_DISP_Q,
```

```
trs_nco_cap_time_disp => TRS_NCO_CAP_TIME_DISP,
```

```
trs_clock_disp  => TRS_CLOCK_DISP,
```

```
MF_bb_cap_i_disp => MF_BB_CAP_I_DISP,
```

```
MF_bb_cap_q_disp => MF_BB_CAP_Q_DISP ,
```

```
bit_i_disp      => BIT_I_DISP,
```

```
bit_q_disp      => BIT_Q_DISP
```

```
);
```

```
end behavioral;
```