

**Performance Evaluation of a Wireless
Protocol for Mesh Networking Under the
Influence of Broadband Electromagnetic Noise**

by

Lily Lai Yam Woo

A Thesis submitted to the Faculty of Graduate Studies of
The University of Manitoba
in partial fulfilment of the requirements of the degree of

Master of Science

Department of Electrical and Computer Engineering
University of Manitoba
Winnipeg, Manitoba, Canada

Copyright © 2010 by Lily Lai Yam Woo

**PERFORMANCE EVALUATION OF
A WIRELESS PROTOCOL FOR MESH NETWORKING
UNDER THE INFLUENCE OF
BROADBAND ELECTROMAGNETIC NOISE**

by

Lily Lai Yam Woo

A Thesis submitted to the Faculty of Graduate Studies of
The University of Manitoba
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

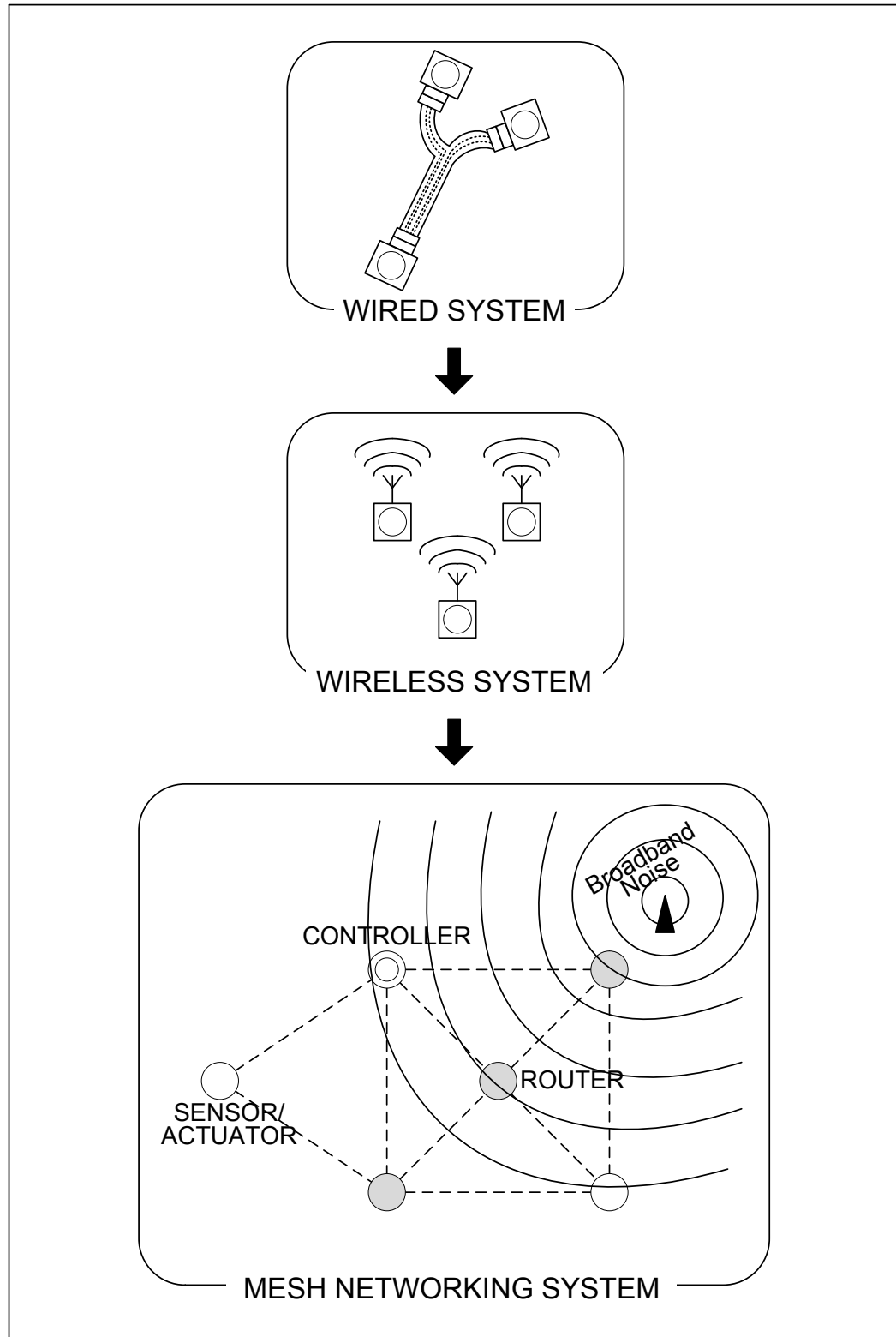
Department of Electrical and Computer Engineering
University of Manitoba
Winnipeg, Manitoba, CANADA

Thesis Advisors: W. Kinsner, Ph.D., P.Eng.
K. Ferens, Ph.D., P. Eng.

(xxiv + 199 + A1 + B1 + C55 + D1 + E1) = 282 pp.

© Lily Lai Yam Woo, February 12, 2010

Visual Abstract



Abstract

Migrating from a wired to a wireless implementation for communication system used in industrial applications is a logical move to avoid the many shortcomings associated with wires. When operated under harsh environments, those wires can break and could cause not only damage to the system but also endanger human lives. However, it is not well documented how well a wireless protocol can work under such harsh industrial environments. This thesis attempts to answer this research question in the point of view of gauging the performance of a wireless protocol under the influence of electromagnetic noise. More specifically, the type of noise signal that is the focus of this investigation is the random, pulsed type (e.g., discharges caused by sparking) that creates a hyperbolic broadband disturbance in the frequency domain. Consequently, a fractal noise model is used to study noise of this nature. The steps toward achieving this goal include: requirements gathering, wireless technology selection; noise modelling and synthesis; real noise capture and analysis to validate the chosen noise model; high-frequency fractal noise emulation in hardware; the use of a novel noise injection method for empirical work; and the conducting of a controlled synthetic noise-to-wireless node performance evaluation to obtain performance measure in the form of packet error rate (PER). Performance data in terms of PER versus signal-to-noise ratio (SNR) for various nodes separation have been collected. There were three significant findings: the obtained performance curves follow the standard 'S' trend; for a specific desired reliability (denoted by a certain PER), the SNR at the transmitter needs to be boosted as the correlation of the noise being present increases; and the maximum distance between nodes separation for a certain reliability to be achieved depends exponentially with the transmitter's SNR. The relationship in the third finding assists in placement of wireless nodes, which in turn can determine the minimum amount of wireless nodes required for an industrial system to reach the desired system reliability, thus boasting network cost saving.

Acknowledgments

I wish to acknowledge the following individuals/groups and thank them graciously for their guidance and assistance in providing me with:

Research and Funding Support:

- My advisors (Dr. W. Kinsner and Dr. K. Ferens)
- My colleague in the Delta Research Group (Dr. M. Potter)
- My funding company, TRILabs (Dr. Diamond, Jeff Rohne, Len Dacombe)
- My sponsor company, Parker Hannifin, previously Vansco Electronics (Kevin Dickson, Danica Pantner, and Kevin Robertson)

Access to Off-Road Vehicles:

- Vansco Electronics (Kevin Hickey)
- Buhler Versatile Inc. (John Vukelic)

Equipment Provision/Loan:

- Arbitrary waveform generators (33120A), electronic parts, power supplies, and cables: The technical staff of Dept. ECE, UofM (Gordon Toole, Sinisa Janjic, Mount-First Ng, Zoran Trajkoski, Brad Tabachnick, Ken Biegun, and Guy Jonatschick)
- Spectrum analyzer (HP 8546A) and RF anechoic chamber (with Faraday Cage): Dr. LoVetri and students (Colin Gilmore, Amer Zakaria, and Cameron Kaye)
- Spectrum analyzer (Anritsu MS2661C): Dr. Bridges, student (Terry Zhi Nong Weng), and CMC (James Dietrich)
- PLL controller module (sample part PLL400-2450A on Sirenza phase locked loop evaluation board): Sirenza (Matt Eckdahl, Chris Blum, and Jackie Silva) and their representative in Canada, Cain-Sweet, Co (Mark A. Sweet, Cliff Mandell, Nicholas Delis, and Wes Hansen)
- Buffer module (attenuator followed by an amplifier) suggestion: Wide Band Systems Inc. (Dennis Gable)

Technical Support:

- Freescale (Kenneth Doe, Wolfgang Pertold, Gonzalo Delgado Huitron, Brian Harold Robinson, JUAN, Rogelio, and Thomas)
- Agilent (Steve Minder, Pete Meyer, Tony Bruszewski, Andre Chartrand, and Nick Nunns)
- Tektronix (David M. McDonald)
- Mini-Circuits (Lu Chen and Michael Fainboym)

Mental Support:

- My parents (Thomas Woo and Molly Foo), my husband (Jonathan Wiens) and his family, my sister (Vivian Woo), my aunty (Susie Blume), Beulah Hasey, and my friends for believing in me, for supporting me with their encouraging words and countless prayers, and for their understanding and patience throughout the preparation and writing of this thesis.

Dedication

Dedicated to my parents.

In memory of my father, Thomas Woo C. K. (1948 - 2005) and
in honour of my mother, Molly Foo K. L.

Table of Contents

	Page
Visual Abstract	iii
Abstract	iv
Acknowledgments	v
Dedication	vi
List of Figures	xi
List of Tables	xvi
List of Abbreviations and Acronyms	xvii
List of Symbols	xxi
I INTRODUCTION	1
1.1 Motivation.....	1
1.2 Problem Definition and Approaches.....	2
1.2.1 Wireless Versus Wired	3
1.2.2 Choice of Wireless Technology	4
1.2.3 Choice of Performance Evaluation	5
1.2.4 Choice of Interfering Noise Signal	6
1.2.5 Choice of Noise Model	8
1.3 Thesis Statement, Thesis Objectives, and Thesis Outcome	9
1.4 Thesis Scope and Thesis Organization	10
1.5 Thesis Contributions	12
II BACKGROUND ON WIRELESS NETWORK PERFORMANCE EVALUATION	13
2.1 General Purpose of Performance Evaluation	14
2.2 General Approaches to Performance Evaluation	15
2.3 Pertinent Performance Evaluation Literature.....	16
2.4 Chapter Summary	20
III SELECTING WIRELESS TECHNOLOGY FOR CONTROL, MONITORING, AND SENSING	21
3.1 Requirements Specification	22
3.2 Criteria for Choosing a Wireless Protocol	25
3.3 A Description of ZigBee/802.15.4.....	30
3.3.1 ZigBee Network Topology	32
3.3.2 Name Origin	33
3.3.3 ZigBee Implementation: Hardware and Software	33

3.4 Practical Deployment Issues.....	34
3.4.1 Placement of Wireless Nodes	34
3.4.2 Placement of Wireless Network.....	36
3.5 Chapter Summary	37
IV MODELLING NOISE FROM INDUSTRIAL ENVIRONMENT	38
4.1 Electromagnetic Interference Mechanism	39
4.1.1 Noise Sources in General.....	40
4.1.2 Types of Noise Emission and Examples of Sources.....	41
4.1.3 What is Not of Interest and Why	43
4.1.4 What the Thesis Focus is and Why: Broadband EM Noise	44
4.2 Noise Model Selected/Employed: Fractal Noise	44
4.2.1 Commonly Used Noise Models in Communications.....	45
4.2.2 Why Employ Fractal Noise Model	47
4.2.3 Relevant Fractal Concepts Revealed	49
4.3 Validation of Noise Model (in the Field).....	57
4.3.1 Methodology for Characterizing Real Noise	59
4.3.2 Characteristics/Complexity of Real Noise	66
4.3.3 Contributions of Noise Validation	70
4.4 Fractal Noise Synthesis (in Software).....	71
4.4.1 Algorithms for Generating Time-Series of Fractal Noise.....	71
4.4.2 Spectral Filtering Algorithm	72
4.4.3 Care Taken when Coding.....	74
4.4.4 Verification of Synthesized Noise Data (Implementation Accuracy).....	77
4.4.5 Characteristics of Synthesized Noise Sequence.....	79
4.5 Fractal Noise Implementation: A Hardware Emulation System.....	80
4.5.1 Emulation Goals	80
4.5.2 Analog Noise Signals Involved: Baseband Fractal Signal Generation Module.....	81
4.5.3 Analog Noise Signals Involved: High-Frequency Fractal Signal Generation Module	82
4.5.4 Hardware Involved	88
4.6 Chapter Summary	89
V DESIGN OF PERFORMANCE EVALUATION	90

5.1 Experiment: Performance Evaluation under Synthetic Noise.....	91
5.1.1 Objective, Methodology, Outcomes, Benefits, Assumptions	92
5.1.2 System Description.....	95
5.1.3 System Revision: What Did Not Work and Improvement Done	100
5.1.4 Apparatus and Materials	106
5.1.5 Procedure	108
5.2 Experiment: Performance Evaluation under Physical Noise	147
5.2.1 Objective, Methodology, Outcomes, Benefits, Assumptions	147
5.2.2 System Description.....	148
5.2.3 Materials	149
5.2.4 Procedure	149
5.3 Chapter Summary	151
VI EXPERIMENTAL RESULTS AND DISCUSSION	152
6.1 Experimental Data	153
6.2 Background Theory	153
6.2.1 On Near-Field Effects.....	153
6.2.2 Inverse-Square Law (ISL).....	154
6.2.3 Friis Equation and Free-Space Path Loss (FSPL).....	154
6.3 Experimental Results & Discussion: PvS_{RX}	155
6.3.1 Calculations: How PER and SNR_{RX} are Obtained.....	156
6.3.2 Simplification / Assumption	158
6.3.3 Why PvS_{RX} Plot	158
6.3.4 Interpreting the PvS_{RX} Plot: Observations and Findings	159
6.4 Experimental Results & Discussion: MvS_{TX}	164
6.4.1 Calculations: How MD and SNR_{TX} are Obtained.....	165
6.4.2 Simplifications / Assumptions	169
6.4.3 Why MvS_{TX} Plot.....	169
6.4.4 Interpreting the MvS_{TX} Plot: Observations and Findings	171
6.5 Usage of PvS_{RX} and MvS_{TX}	173
6.5.1 Placement of Nodes: 1-D Methodology	173
6.6 Experimental Results & Discussion: For Experiment with Real Noise Source.....	180
6.7 Chapter Summary	180
VII CONCLUSIONS	182

7.1 Main Findings.....	182
7.2 Limitations.....	184
7.3 Recommendations.....	185
7.4 Contributions	186
References.....	190
APPENDIX A: CODE	[1 pp.]
A.1 Monofractal Noise Generation, Written in MATLAB.....	
A.2 PER Monitoring Software, Written in C.....	
A.3 GPIB-USB Trace Data Acquisition Software, Written in C#.....	
A.4 Compute Power Within Specified Bandwidth from Trace, Written in MATLAB ..	
A.5 Captured Noise Analysis: Spectrogram, Written in MATLAB	
A.6 Captured Noise Preprocessing: Convert .IQT to Time Series, Written in MATLAB	
A.7 Captured Noise Analysis: VFDT, Written in MATLAB.....	
A.8 Modulated Monofractal Noise Analyses, Written in MATLAB	
APPENDIX B: DATA.....	[1 pp.]
B.1 Capture Noise Data from Starter Source of Tractors.....	
B.2 Baseband Monofractal Noise Sequence or Datapoints	
B.3 Packet Error Rate Data	
B.4 Trace Data: ZigBee Spectrum and Noise Spectrum	
APPENDIX C: SUPPLEMENTARY TECHNICAL DOCUMENTATIONS	[55 pp.]
C.1 Wireless Design Alternatives.....	5 pp.
C.2 Voltage Regulator Circuits	3 pp.
C.3 Carrier Frequency Drifting Issue: Problem (Cause and Effect) and Solutions	3 pp.
C.4 Uploading Datapoints onto AWG using HyperTerminal	9 pp.
C.5 Uploading Datapoints onto AWG using IntuiLink.....	12 pp.
C.6 Performance Evaluation Setup and Noise Emulation Circuit Configuration...	2 pp.
C.7 Capturing Trace Data through GPIB Interface	15 pp.
C.8 Performance Evaluation Setup with Fully-Specified Connections of Cables..	2 pp.
C.9 Performance Evaluation Hardware and Software Listing.....	3 pp.
C.10 Determining Playback Frequency for Arbitrary Waveforms on AWG	1 pp.
APPENDIX D: RESULTS COMPILED IN TABULAR FORM.....	[1 pp.]
APPENDIX E: EXPERIMENTAL SNAPSHOTS.....	[1 pp.]

List of Figures

		Page
1.1	A wireless network replaces the wires and wire harnesses (dashed lines) to connect the end devices (sensors and actuators) to their respective input/output and control modules.	4
2.1	Layout of Chapter 2	13
3.1	Layout of Chapter 3	21
3.2	Wired connections in a typical communication system for a CMS application.	23
3.3	The ZigBee stack in comparison with the Open Systems Interconnection model.	32
3.4	A topological representation of a ZigBee mesh network (i.e., physical representation is not implied) illustrating its devices' network role and hardware type.	33
4.1	Layout of Chapter 4	39
4.2	Taking an instance of (top left) a Brownian motion process, and of (top left) a first-order autoregressive process (top right), and magnifying the region between the two dotted lines twice.	50
4.3	The unscaled (top row) and scaled (bottom row) versions of a self-affine signal belonging to (left column) a statistically scale-invariant stochastic process, the Brownian motion, and (right column) a morphologically scale-invariant deterministic process, defined by the Weierstrass-Mandelbrot function [BeLe80]. After [WPKF07].	51
4.4	Broadband frequencies versus band-limited frequencies, illustrated through (top) the spectrum of a power law process (brown noise), and (bottom) the characteristic of a filter's (low-pass) frequency response.	53
4.5	The time-domain versus spectral-domain representations for monofractal (top) white noise, (middle) pink noise, and (bottom) brown noise, where their spectral exponent, β , is 0, 1, and 2, respectively.	55
4.6	A snapshot of the noise acquisition setup.	59

4.7	A screenshot of one of the noise recordings viewed offline using RSAVu. The blue area is where the frequency mask is applied such that only captured signal with frequencies above -90 dBm will trigger to be stored/saved.	62
4.8	Variance fractal dimension trajectory (After [GrKi94]).	65
4.9	The time series of a sequence of different monofractal noises combined (top) and the result of passing the sequence to the VFDT (bottom).	66
4.10	The PSD (left, bottom) and spectrogram (right) of a captured noise time series (left, top).	67
4.11	Time series representing noise signals emanated by the starter system of a Buhler tractor measured at 3 metres (top, left), at 2 metres (top, right), and of a Ford tractor measured at 1 metre (bottom, left), and at 0.05 metres (bottom, right) away from the noise source....	68
4.12	Flowchart depicting the SFA for generating an fBm of the desired spectral exponent. ...	74
4.13	Flowchart for verifying the output of the MATLAB implementation of SFA.	78
4.14	Error in the computed spectral exponent corresponding to each noise colour.	79
4.15	Synthetic noise being added to the ZigBee receiver system: the modulating signal is N_{BB} (which represents the baseband noise signal that is scale-invariant, denoted by S), the carrier signal is C , the modulated signal is N_{HF} (which denotes high-frequency noise signal), the ZigBee transmitted signal is Z , and the interference-resultant signal going into the receiver system is R	84
4.16	Fractal modulation by Wornell, where the modulating signal is M (which denotes the message signal), and the carrier signal is B (which denotes a basic function signal), and the modulated signal is E (which represents the signal that has information embedded within itself on all temporal scales, hence a scale-invariant signal).	85
4.17	The frequencies that matter for the signals generated in 4.5.2.1 and in 4.5.3.1, respectively (After [WKFD07]).	88

5.1	Layout of Chapter 5	90
5.2	ZigBee-synthetic noise performance evaluation setup for PER measurements.....	96
5.3	ZigBee-noise performance evaluation setup for SNR measurements; (a) setup for measuring power of received ZigBee signal, and (b) setup for measuring power of noise signal.....	97
5.4	PLL connection diagram (after [Sire07a]).....	111
5.5	State Diagram for Transmitter Application.	119
5.6	Main Program Flowchart for Transmitter Application.....	122
5.7	State Diagram for Receiver Application.....	123
5.8	Main Program Flowchart for Receiver Application.	126
5.9	Supplementary Flowchart for RECEIVE PACKET state part of Receiver Application.	127
5.10	Supplementary Flowchart for GOOD PACKET state part of Receiver Application.....	128
5.11	Supplementary Flowchart for ECHO state part of Receiver Application.....	129
5.12	A HP 8546 spectrum analyzer screenshot showing two traces, 1) the spike of the 2.402 GHz PLL-generated carrier signal (in black shade), and 2) the spectrum of the modulated monofractal noise signal (in gray shade) originating from the baseband noise signal generated by AWG with configuration of 1.050 V _{pp} and using arbitrary waveform of spectral exponent 1.6 to induce 20% PER.....	133
5.13	The components of the performance evaluation under synthetic noise setup that reside outside of the anechoic chamber.....	134
5.14	The fully-assembled noise emulation circuit for generating high-frequency, analog fractal-based noise.	135
5.15	The components of the performance evaluation under synthetic noise setup that reside inside the anechoic chamber.....	136
5.16	Pseudocode of the trace data acquisition GUI software	138

5.17	Specific setup at the combiner module for enabling the capturing of ZigBee spectrum with spectrum analyzer (HP 8546A) and trace data acquisition GUI.....	142
5.18	Specific setup at the combiner module for enabling the capturing of input noise spectrum with spectrum analyzer (HP 8546A) and trace data acquisition GUI.....	143
5.19	A plot of the noise spectrum of one of the input noise signals captured by spectrum analyzer (HP 8546A) and trace data acquisition GUI software and with power within the 2 MHz frequency of interest (in red) calculated by compPowerBW.m and plotted by main_program_genpowerBW.m.....	145
5.20	A plot of the carrier signal spectrum (a spike) captured by spectrum analyzer (HP 8546A) and trace data acquisition GUI software.....	146
5.21	A plot of the ZigBee spectrum captured by spectrum analyzer (HP 8546A) and trace data acquisition GUI software and with its power within the 2 MHz frequency of interest (in red) calculated by compPowerBW.m and plotted by main_program_genpowerBW.m.	147
5.22	ZigBee-physical noise performance evaluation setup for PER measurements.....	148
5.23	The case 1 scenario where PER measurements are gathered for ZigBee transmissions without physical noise being present.	150
5.24	The case 3 scenario where PER measurements are gathered for ZigBee transmissions with physical noise (plasma globe switched on) being present.	151
6.1	Layout of Chapter 6.....	152
6.2	Performance of ZigBee communication in the presence of synthetic noise (modulated monofractal noise of spectral exponents 0.4, 0.8, 1.2, and 1.6, respectively) for nodes separation of 1 metre.....	160
6.3	Performance of ZigBee communication in the presence of synthetic noise (modulated monofractal noise of spectral exponents 0.4, 0.8, 1.2, and 1.6, respectively) for nodes separation of 2 metre.....	161

6.4	Performance of ZigBee communication in the presence of synthetic noise (modulated monofractal noise of spectral exponents 0.4, 0.8, 1.2, and 1.6, respectively) for nodes separation of 4 metre.....	162
6.5	Visual aid depicting the relationship between distance of nodes separation and transmitter output power to induce the same PER value at the respective receivers.	170
6.6	Maximum nodes separation as a function of signal-to-noise ratio at the transmitter for reliable communication (indicated by the value of PER tolerable), corresponding to each input noise type.....	172
6.7	The volume of the space for placement of router nodes between a source node (say, a sensor) and a destination node (say, the main controller) situated at d_0 distance apart from each other.....	174

List of Tables

 Page
1.1 Scope of thesis	11
3.1 Comparison highlighting specific ZigBee and Bluetooth characteristics.	29
4.1 Materials and equipment for real noise data acquisition.....	61
5.1 Table for recording the noise amplitude level tuned from the AWG that will induce the targeted PER%, the name of the files that stored the PER monitoring output and the trace data acquisition output, and the computed PER and SNR measurements for each input noise type for the ZigBee transmission between 2 nodes separated at the specified distance.	109
6.1 PER measurements for ZigBee transmission with physical noise source (plasma globe).	180

List of Abbreviations and Acronyms

AM	amplitude modulation (p. 83)
App Note	application note (p. 102)
ARQ	automatic repeat request (p. 177)
AWG	arbitrary waveform generator (p. 74)
AWGN	additive white Gaussian noise (p. 8)
BER	bit error rate (p. 14)
BNC	Bayonet Neill-Concelman connector (p. 108)
bps	bits per second (p. 26)
CAN	controller area network (p. 22)
CISPR	International Special Committee on Radio Interference (p. 41)
CMS	control, monitoring, and sensing (p. 1)
CRC	cyclic redundancy check (p. 29)
CW	CodeWarrior (p. 115)
DAC	digital-to-analog converter (p. 102)
DC	direct current (p. 47)
DSB-SC	double-sideband, suppressed-carrier (p. 83)
DSSS	direct sequence spread spectrum (p. 31)
EM	electromagnetic (p. 1)
EMC	electromagnetic compatibility (p. 180)
EMI	electromagnetic interference (p. 5)
ESD	electrostatic discharge (p. 139)
FCC	Federal Communications Commission (p. 180)
FFD	<i>full-function device</i> (p. 32)
FFT	Fast Fourier Transform (p. 77)

G	giga (p. 7)
GND	ground (p. 111)
GPIB	General Purpose Interface Bus (p. 104)
GUI	graphical user interface (p. 101)
Hz	hertz (p. 2)
IEEE	Institute of Electrical and Electronics Engineering (p. 4)
ISM	international, scientific, and medical (p. 7)
ISO	International Organization for Standardization (p. 42)
LED	light-emitting diode (p. 115)
LPF	low-pass filter (p. 100)
M	mega (p.2)
MAC	medium access control (p. 6)
MD	maximum distance of nodes separation (p. 164)
MvS_{TX}	plot of maximum distance versus signal-to-noise ratio at the transmitter (p. 159)
N connector	Type N connector (p. 108)
NS2	Network Simulator (p. 184)
OS	operating system (p. 104)
PC	personal computer (p. 115)
PER	packet error rate (p. 6)
PHY	physical (p. 11)
PLL REF	the reference frequency for the phase-locked loop module (p. 101)
PLL	phase-locked loop (p. 98)
PSD	power spectrum density (p. 8)
PvS_{RX}	plot of PER versus signal-to-noise ratio at the receiver (p. 155)
RAM	radiation absorbent material (p. 94)

REF UNLOCK	error message from SA model HP 8546A denoting ... (p. 101)
RF	<i>radio frequency</i> (p. 16)
RFD	<i>reduced-function device</i> (p. 32)
RFI	radio frequency interference (p. 5)
RNG	random number generator (p. 75)
RTSA	real-time spectrum analyzer (p. 60)
SA	spectrum analyzer (p. 101)
SMA	SubMiniature version A connector (p. 107)
SMAC	<i>Simple Media Access Controller</i> (p. 34)
SNR	signal-to-noise ratio (p. 6)
SNR_{RX}	signal-to-noise ratio at the receiver (p. 35)
SNR_{TX}	signal-to-noise ratio at the transmitter (p. 35)
sut	signal under test (p. 144)
USB	Universal Serial Bus (p. 104)
VCO	voltage-controlled oscillator (p. 99)
VDC	voltage direct current (p. 115)
VFD	variance fractal dimension (p. 63)
VFDT	variance fractal dimension trajectory (p. 63)
WBAm	wideband amplified (p. 98)
WBAt	wideband attenuator (p. 98)
Wi-Fi	wireless fidelity (p. 1)
WISENES	Wireless SEnsor NEtwork Simulator (p. 184)
WLAN	wireless local area network (p. 11)
WMAN	wireless metropolitan area network (p. 11)
WPAN	wireless personal area network (p. 11)

WRAN	wireless regional area network (p. 11)
ZC	ZigBee coordinator (p. 32)
ZED	ZigBee end device (p. 32)
ZR	ZigBee router (p. 32)

List of Symbols

%	percent (p. 64)
$\lfloor \cdot \rfloor$	floor function (p. 178)
β	a real-valued parameter denoting the exponent of $1/f^\beta$ noise, also known as spectral exponent that determines the time-correlation of the noise signal and, therefore, the noise colour (p. 8)
λ	carrier wavelength (p. 154)
Ω	ohm, the unit of electrical resistance (p. 107)
β^*	estimated spectral exponent (p. 77)
$\{ \}$	a set (p. 163)
A	area (p. 154)
abs	absolute value or modulus (p. 79)
$\hat{B}(m)$	Fourier coefficients of the desired fBm (p. 73)
c	propagation speed (p. 154)
C(t)	carrier signal (p. 99)
cm	centimetre (p. 154)
D	dimension (p. 56)
D	distance of nodes separation between the transmitter and the receiver (p. 130)
D	Fractal dimension (p. 56)
D	longest antenna dimension in metre (p. 154)
d	minimum far-field distance (p. 153)
D_σ	variance fractal dimension (p. 69)
d_0	distance between source and destination nodes (p. 174)
dB	the units of power in decibel (p. 31)

dBm	power measurement in decibel referenced to 1 mW (p. 27)
D_E	Euclidean dimension (also known as topological dimension) (p. 56)
d_{max}	maximum distance for nodes separation (p. 176)
d_x	distance for router nodes to be set apart from neighbouring nodes (p. 178)
f	frequency in hertz (p. 8)
f_c	carrier frequency (p. 88)
f_z	selected ZigBee channel center frequency (p. 88)
G_{RX}	receiving antenna gain (p. 154)
G_{TX}	transmitting antenna gain (p. 154)
H	Hurst exponent (p. 56)
i	counter (p. 176)
I	intensity (p. 154)
i	i -th experimental run (p. 157)
i	trial number (p. 178)
j	j -th trace (p. 157)
k	k -th trace (p. 158)
L	total number of lost packets in each experimental run (p. 157)
m	discrete frequency (p. 73)
m	$m = d_0/d_x$ (p. 178)
m	metre, used in the context of distance (p. 29)
m	milli, used as a prefix to units (p. 29)
max	the maximum function to find the largest number in the given list (p. 79)
m/s	meter per second (p. 154)
N	length of the desired fBm (p. 72)
n	path loss exponent (p. 154)

n	total number of captured traces (p. 157)
$N(j)$	digital noise sequence or datapoints (p. 98)
$N(t)$	analog baseband noise signal (p. 98)
$n(t)$	noise signal (p. 45)
$N^*(t)$	analog modulated noise signal (p. 99)
$N_{\text{NODES_MIN}}$	minimum number of wireless nodes in a network (p. 176)
N_{ROUTER}	number of router nodes (p. 176)
P	signal power (p. 154)
π or π	a constant with value 3.14159 (p. 76)
PL	path loss (p. 155)
P_N	input noise power (p. 92)
P_{NMAX}	maximum noise power (p. 175)
$PSD(.)$	power spectrum density function (p. 68)
P_{SRX}	received signal power (p. 92)
$P_{\text{SRX_EXP}}$	Received signal power expected (p. 176)
$P_{\text{SRX_MEAS}}$	Received signal power measured (p. 176)
P_{STX}	transmitter signal (output) power (p. 154)
$P_{\text{STX_AFF}}$	affordable transmitter output power (p. 175)
r	PER value (p. 179)
r	scaling (magnification/reduction) factor (p. 56)
R	total number of experimental runs (p. 157)
$r(t)$	received signal (p. 45)
$s(t)$	transmitted signal after having experienced path loss and attenuation as it propagates to the receiver (p. 45)
$SNR_{\text{TX_A}}$	actual SNR at the transmitter (p. 175)

$SNR_{TX_{dx}}$	value for signal-to-noise ratio corresponding to the d_x value extracted from the plot of MvS_{TX} for reliability governed by $r^0\%$ PER (p. 179)
$\text{sqrt}(\cdot)$	square root function (p. 76)
T	total number of packets transmitted in each experimental run (p. 157)
traceN	captured trace of the noise signal used (p. 156)
traceS	captured trace of the received signal (p. 156)
Trn	the power within 2 MHz bandwidth computed from the noise trace (p. 157)
Trs	the power within 2 MHz bandwidth computed from the signal trace (p. 158)
V	Volts (p. 112)
V+	supply voltage to these modules (attenuator and amplifier) (p. 111)
V_{cc}	supply voltage to voltage controlled oscillator module (p. 100)
V_{cont}	tuning voltage to the attenuator module (p. 111)
V_{pp}	Volt peak-to-peak (p. 75)
V_{rms}	Volt root mean square (p. 109)
V_{tune}	tuning voltage to voltage controlled oscillator module (p. 100)
W	Watt (p. 29)
$W(n)$	zero mean and unit variance white Gaussian noise discrete sequence of N data points (p. 73)
$\hat{W}(m)$	Fourier coefficients of $W(n)$ (p. 73)
x	number of intermediate nodes (p. 178)

Chapter 1

INTRODUCTION

1.1 Motivation

The ultimate goal motivating this thesis is the resilience of wireless networks in a real environment with *electromagnetic* (EM) noise. A wireless network is a network of nodes or devices that uses EM waves (i.e. radio waves) instead of wires for transmission and reception of data. Wireless communication in the network is achieved through the use of a standard wireless networking technology such as Wi-Fi [WiFi08], Bluetooth [Blue08a] or a proprietary protocol. The operating environment of a wireless network often contains noise interference. When EM radiation or noise signals interfere with the intended wireless signal, the performance of the wireless network may be affected. The effect could be degradation or even interruption of the network operation. Thus, a performance evaluation study is required to determine the extent of the wireless network survivability in the presence of such a disturbance.

This thesis is motivated by a need to develop a ZigBee network for an industrial application, and by the lack of published literature on wireless network performance under EM noise, in particular, ZigBee under broadband EM noise. The aforementioned type of wireless technology and source of interference were chosen specifically for the context of this work, which is wireless networking for industrial environment. The contending wireless network technologies discussed in this thesis for industrial *control, monitoring, and sensing* (CMS) systems includes Bluetooth and ZigBee. Bluetooth has been studied extensively for a period of time, but ZigBee is a new technology (its specification first ratified in 2004) and has not been evaluated in a real-world environment.

In terms of noise source, broadband EM noise (considered as the interfering signal) has not been emphasized in the past performance evaluation literature. Interference due to radio signals

from nearby wireless networks has been the common focus. In this thesis, broadband is a relative term and implies that the EM noise has a bandwidth that spans across or exceeds multiple narrowband wireless channels. Examples of narrowband noise include 60 Hz noise due to 60 Hz signal emissions from lights and computers, and spectra of active wireless channels due to nearby wireless transmission systems in operation, such as Wi-Fi (a 22 MHz receiver bandwidth), Bluetooth (a 1 MHz receiver bandwidth), and ZigBee (a 2 MHz receiver bandwidth), which are considered as unwanted signals that can cause interference to the wireless system under study. On the other hand, noises occurring from electronic devices such as white noise and pink noise, and transients in electrical systems are some examples of broadband noise.

Applications that may benefit from this work include industrial, home, utilities, medical and healthcare, agriculture, and logistics. In particular, this work has significance to any application that requires a wireless communication system to enable CMS, such as in planes, trains, ships, and storage or shipping containers.

1.2 Problem Definition and Approaches

The research problem posed in this thesis is: “How well does a wireless networking technology (in this case, ZigBee) perform under noise (in this case, broadband EM noise) from the operating environment of industrial CMS systems?” An understanding of ZigBee performance in noise can lead to more resilient wireless networks. This problem is worth answering because (i) no work has been done in terms of using broadband EM noise as the focal noise for interfering with the intended wireless signal, and (ii) this work is applicable across numerous application areas, such as industrial, medical, and commercial. In particular, the outcome of this work (i.e., performance data) will be useful to wireless system designers building networks for such application areas, as well as to the improvements of the wireless protocol specifications. For completeness, the other research question that is raised but not investigated in

this thesis is: “How many nodes can a ZigBee network accommodate in a real world implementation?” The addressing scheme in the ZigBee 2004 specification indicates that a network can support up to 2^{16} nodes [ZiAl04]. In reality, this theoretical maximum number of nodes may be restricted to a smaller number of nodes that the network can support before congestion occurs [HaSt06, Schw08].

1.2.1 Wireless Versus Wired

To-date, most machinery requires some form of data communication, which is normally achieved by interconnecting components (such as sensors, actuators, and controllers) using wires and wire harnesses. A wired implementation is prone to breaking for machines that are deployed in industrial environments. Inevitable factors from the operating environment, such as heat and vibration, tend to exhaust wires within the wire harnesses in the communication system, thus causing them to be either shorted, open, or have a high resistance. As a result, electrical malfunction may happen and could potentially cause damage to the machine and/or harm to human life, and may be very expensive in terms of time, money, and resources in repair and maintenance [TWHH03, Stew03]. Such a major shortcoming associated with any traditional wired communication systems can be avoided through a switch to wireless implementation. By going with a wireless solution, the stress and strain from the application’s operating environment will no longer have any impact on the communication system. Furthermore, a few major advantages would be a reduction of wired connections, ease of installation (i.e., addition and removal of devices), allowance for non-intrusive testing, and savings in terms of costs for materials and maintenance of the wireless communication system. A wireless solution is incomplete without the selection of a wireless technology that is best suited for the desired CMS application, which will be discussed next. The target application for this research work is a

wireless communication system for industrial machinery, such as excavators and tractors. Figure 1.1 shows a typical placement of a wireless network within the CMS system of an excavator.

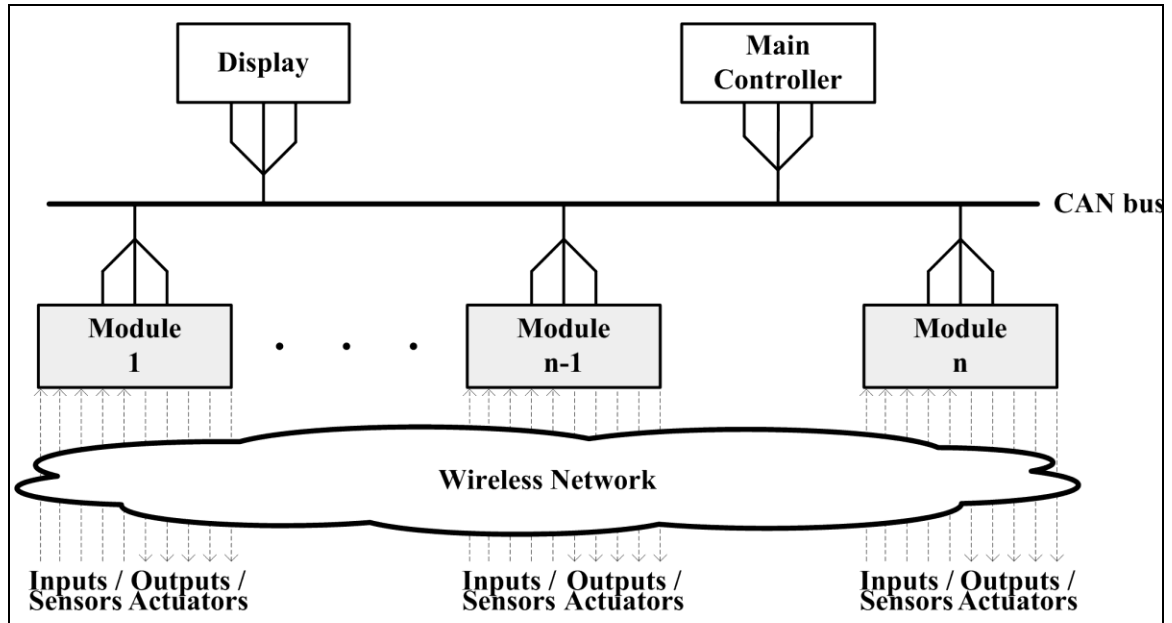


Fig. 1.1 A wireless network replaces the wires and wire harnesses (dashed lines) to connect the end devices (sensors and actuators) to their respective input/output and control modules.

1.2.2 Choice of Wireless Technology

ZigBee is proposed as a suitable wireless technology to replace the wired harness in sensor networks within industrial CMS systems because of the following features: low data rate, low power consumption, low cost, and reliability through mesh networking [WKFD06, WKFD07]. ZigBee [ZiA104] is a short-ranged wireless technology that is built upon the IEEE 802.15.4 [IEEE03] standard. Since the target application calls for a communication system that consists of sensors and actuators, the sensors play a big role in defining a suitable wireless technology. The passing of small-sized messages to and from the sensors [WoCu01], the use of batteries to power the sensor nodes, the expense involved in installing and maintaining a wired sensor network, the reliability of the communication system due to the stress and strain from the application's operating environment, are some of the issues that will directly benefit from ZigBee. For

example, by using a mesh topology [Poor03], nodes in the wireless network can not only be formed automatically, but can also self-heal through rerouting messages to the next available in-range node when a link is broken or when a node is down. Despite the short range specification of ZigBee (up to 100 meters [ZiAl08]), the distance of wireless coverage can be extended through the addition of nodes within range of others in the mesh as would be the case for a network within the cab of an excavator to have radio communicability reaching to the boom area.

However, having a communication system that employs wireless networking as a solution to prevent communication failure due to broken wires is not without any shortcomings. The downside is that wireless networks in general are susceptible to interference (i.e., *radio frequency interference*, RFI, and *electromagnetic interference*, EMI), whose presence is common to any industrial environment [Masi07]. RFI is induced by the various wireless communication systems that are deployed in an industrial site whose transmitters operate in the same frequency spectrum as the proposed wireless system. Meanwhile, EMI is due to the electromagnetic noise radiated from the machinery around or within the industrial CMS system. Both types of interference may cause any wireless networks' performance to degrade. Thus, the impact of noise upon the intended wireless communication signal needs to be investigated.

1.2.3 Choice of Performance Evaluation

The worst-case performance evaluation of ZigBee under broadband EMI is investigated empirically and is the main goal and focus of this research work in order to fill the lack of coverage in performance evaluation literature on the subject of examining the effects of such type of interference upon a wireless technology, and to gauge the practical limit or extent of the chosen wireless technology. Since this work does not involve any alterations/improvements to the ZigBee/802.15.4 protocol, the performance evaluation is not on quantifying the impact of the features in the protocol but on quantifying the impact of noise toward the protocol's network

performance, which means that the performance metrics for evaluating *medium access control* (MAC) features such as energy, latency, throughput, and delivery ratio [LuKR04] will be irrelevant to the context of this work. Bit error rate, packet loss, access delay, end-to-end delay, and goodput are some of the network performance metrics [Golm04]. This work emphasizes packet loss (a measure conducted after error correction) in order to provide a benchmark of the protocol's immunity to broadband interference. The term worst-case is in the sense that the performance evaluation is achieved through the addition of noise signal onto the received signal within the receiver's circuitry (i.e., over-the-wire) and through the use of *packet error rate* (PER) as the performance measure without the activation of retransmission, acknowledgement, and error correction in order to evaluate the protocol as is. Meanwhile, empirical work is favoured instead of simulation work so that the actual performance of real hardware employing the chosen protocol under the influence of the chosen noise type can be observed, where no speculation can take place. The empirical evaluation is achieved through a custom-designed experimental setup that includes the use of off-the-shelf ZigBee transceivers, emulation of broadband stochastic EM noise through fractal noise model, noise addition hardware, and software such as PER monitoring, trace capturing, and *signal-to-noise ratio* (SNR) computation programs.

Having covered briefly a description of the proposed scheme for performance evaluation experiments, the other part necessary to evaluate the performance of ZigBee or any wireless technology is the source of interference. In particular, a description of the reasoning and the modelling for the chosen type of noise signal is presented next.

1.2.4 Choice of Interfering Noise Signal

Broadband, stochastic, uncorrelated and correlated, electromagnetic (EM) noise is chosen as the noise signal type for this work because (i) such noise exists (i.e., is present and is real), (ii) there is lack of work in the literature that considers broadband correlated interference, and (iii) the

practical extent of a wireless communication system performance in an industrial environment can be gauged. In a vehicular application, electrical and electronic components nearby the potential wireless communication system (for example, starter motor, blower motor, wiper motor, and alternator) all pose as sources of interference. These noise sources emit stochastic, transient signals, which will impair the performance of the wireless communication system with its emission that spans over a broad spectrum of frequencies, i.e. broadband interference [ISO98, IEC02]. Performance evaluation literature [SiGr05, GoCR05, SBSL06] typically uses the approach where deterministic signals emitted by Wi-Fi, Bluetooth, or ZigBee devices are applied as the noise source to investigate either the issue of the chosen wireless protocol's performance in the face of RFI or the issue of coexistence between various wireless transceivers at the 2.4 GHz *international, scientific, and medical* (ISM) band, the license-free frequencies for global use. In the context of this research work, these noise sources are considered to emit *deterministic* (with its known shape of spectrum) and *narrowband* (since it causes narrowband interference to the wireless signal of interest) signals. To the best of the author's knowledge, there has been very little or no work that has been done in the area of performance evaluation under electromagnetic interference, more specifically stochastic, broadband uncorrelated and correlated interference (other than the simplest case of white noise) at the time of this research study. Broadband interference is more severe than narrowband interference in that the noise signal will most likely wash out wireless transmission and reception across the entire set of available channels. Because of that, the mitigation or removal of broadband noise is non-trivial, where filtering and channel switching are not viable methods. Due to the severity of such noise, it cannot be ignored and is thus the choice for a noise model to reflect the hostile operating environment of industrial applications.

Having determined the type of noise to use in the proposed performance evaluation experiments, the next step is to determine a noise model that can serve as a basis to generate such noise in order to produce broadband interference over ZigBee channels.

1.2.5 Choice of Noise Model

The noise model for this work is borrowed from fractal theory, more specifically monofractal noise signals (include white and coloured noises) are used as the input noise to conduct the performance evaluation experiments. Common traditional noise models used by researchers consist of white noise of specific time-domain statistical distributions (e.g., uniform or Gaussian), and spectra of various radio transmitters. White noise is a broadband noise (since its *power spectrum density* (PSD) is flat or evenly distributed across all frequencies), however, it is not sufficient to cover real, physical EM noise, which may have a sloped PSD. On the other hand, spectra from various radio transmitters are irrelevant as a noise model for this work as it will only enable the researcher to determine the impact of RFI and not EMI. EM noise occurs intermittently and is transient-like, which will produce a broadband disturbance [Kins02]. Moreover, the time-domain samples of physical noise signals are scale-invariant [PeJS92, Worn96, Kins04]. Furthermore, a recent work supports that noise emitted by the starter source from a tractor emit signals that exhibit changes in fractal complexity [WoKF08]. All of these call for a noise model that can model the stochastic (or random) noise process and the property of scale-invariance, and can produce noise signals of various degree of time-correlation or complexity to simulate the different gradient or inclination of PSD over a wide span of frequencies. A power law decay noise model (i.e., $1/f^\beta$ noise) with a long-tailed distribution fits such description with β being the spectral exponent that sets the level of PSD gradient [PeJS92, Kins04]. Employing $1/f^\beta$ noise model for natural physical processes or phenomenon is not a new idea as it has been used in the

following research areas: deoxyribonucleic acid sequence, speech, annual rainfall amount, and lightning strikes, to name a few [PeJS92, Worn96, GaCL03, Kins04].

1.3 Thesis Statement, Thesis Objectives, and Thesis Outcome

This thesis studies, quantifies, and evaluates empirically the worst-case packet loss performance of a wireless network employing the ZigBee/802.15.4 technology in an industrial setting through (i) the emulation and addition of broadband electromagnetic noise (modelled by monofractal noise signals) and (ii) the measurement of packet error rate at various noise power levels (to emulate the different signal-to-noise ratios) over various transmitter-receiver nodes' separation, at the receiving node.

The objectives used for this work include:

- (i) To determine a suitable wireless technology based on the target applications requirements,
- (ii) To model and emulate high-frequency broadband EMI based on fractals theory and fractal-based signal processing,
- (iii) To develop an experimental setup for controlled study of ZigBee-noise performance evaluation based on theories of communications, signal processing, and circuits and devices,
- (iv) To measure PER and obtain traces of ZigBee and noise spectra for each experiment.

The outcome of this thesis is performance data (i.e., PER versus SNR) stating the operational extent of ZigBee in industrial environment. The usefulness of the data is two-fold:

- (a) When the SNR at an industrial site/environment is known, the worst-case performance of ZigBee under such SNR can be looked up from the performance data.

- (b) The maximum distance for reliable communications between two ZigBee nodes can be determined by plotting distance versus SNR from data obtained for PER experiments repeated over different node separations.

1.4 Thesis Scope and Thesis Organization

This work calls for a multidisciplinary research which requires the following knowledge fields: fractals, signal processing, test and measurement, practical hands-on experimental work, programming design and development, technical communication, mathematics, communications, telecommunications, and antennas. In terms of the scope of this thesis, it is neither about prototype/product development nor on improving/changing a specific layer in the protocol. Table 1.1 depicts a more refined scope in terms of what to expect and what not to expect from this thesis. The context for this work is industrial environment, focusing on wireless industrial vehicular CMS system. This thesis references an excavator as an example of industrial machinery as a desired application requiring a wireless CMS system. This is also the reason for the values set for experimental parameters such as the noise input, the ZigBee operating frequencies, and the distance between nodes separation in the thesis.

Table 1.1 Scope of thesis.

On the subject of	This thesis IS about	This thesis is NOT about
Wireless technology	WPAN that includes ZigBee/802.15.4 only.	WLAN, WMAN, or WRAN. Yes, WPAN, but not on Wi-Fi (variants b or g) or Bluetooth.
Packet error evaluation technique	Experimental (empirical) measurements.	Analytical (mathematical) or simulation (computer) measurements.
Performance measures	One performance metric, i.e., packet loss in terms of packet error rate.	All performance metrics.
Issue examined	EMI.	Coexistence or RFI.
Noise type	Stochastic, broadband, uncorrelated and correlated signals.	Deterministic or narrowband.
Noise model	Fractal-based, specifically $1/f^\beta$ noise, with white noise and coloured noise inclusive.	Radio signals.
Antenna	Usage of microstrip patch antenna only.	In-depth antenna theory, design, and analysis.
Testing performed at protocol level	Physical (PHY) layer of ZigBee protocol.	Higher ZigBee layers.

This thesis is organized into seven chapters. Chapter 1 highlights the tradeoff between unreliable wired communication systems due to the harsh industrial operating environment factors and unreliable wireless communication systems due to interference, and the motivation for investigating the packet loss performance of ZigBee under the influence of broadband electromagnetic noise. Chapter 2 offers a background on related work in terms of performance evaluation techniques, performance measures, and the main players (wireless technology and noise model) involved. Chapter 3 describes the choice and reasoning for using ZigBee as the wireless technology for wireless CMS networks. Chapter 4 discusses the noise source of interest, its model, synthesis, implementation, and validation. Chapter 5 provides a description of and the reasoning behind the design of the setup for packet error experiments, and includes the procedure and materials required for the performance evaluation. Chapter 6 presents and analyzes the results

of these experiments. Finally, Chapter 7 reveals the major conclusions, contributions, and recommendations for future work.

1.5 Thesis Contributions

The major contributions of this thesis in each of the following categories are:

Science and Technology

- (a) Addresses the lack in performance evaluation research by examining the effects of broadband, stochastic electromagnetic (EM) noise (i.e. transient or pulsative in time-domain) on wireless technology.
- (b) Proves mathematically and empirically that there is a change in character of baseband input (monofractal) noise when it is amplitude modulated.
- (c) Determines the complexity of captured noise to be multifractal.

Research Community

- (a) Evaluates the performance of ZigBee system in industrial application (i.e. under industrial electromagnetic noise) through packet error rate measurement of ZigBee under synthesized broadband fractal noise and under real physical broadband noise (plasma globe), respectively.
- (b) Optimizes/improves the performance evaluation setup.
- (c) Discusses the use of monofractal noise as model for broadband, stochastic uncorrelated and correlated electromagnetic noise.
- (d) Proposes (not proven) that the minimum number of nodes in a wireless network for reliable operation may be achieved through the availability of these parameters (volume of space for nodes placement, reliability in PER, and the extraction of the maximum distance between nodes from PER vs. SNR at various node separation performance data).

A detailed description of the above contributions is included in Section 7.2.

Chapter 2

BACKGROUND ON WIRELESS NETWORK PERFORMANCE EVALUATION

The objectives of this chapter are to identify the past and current works or literature in the area of wireless network performance evaluation and to describe how the work presented in this thesis relates to and adds towards the previous studies. Performance evaluation is not a new subject area, and has a broad coverage. As such, this chapter can only provide coverage on a small portion of the literature on the subject that is pertinent towards the thesis goal and research methodology.

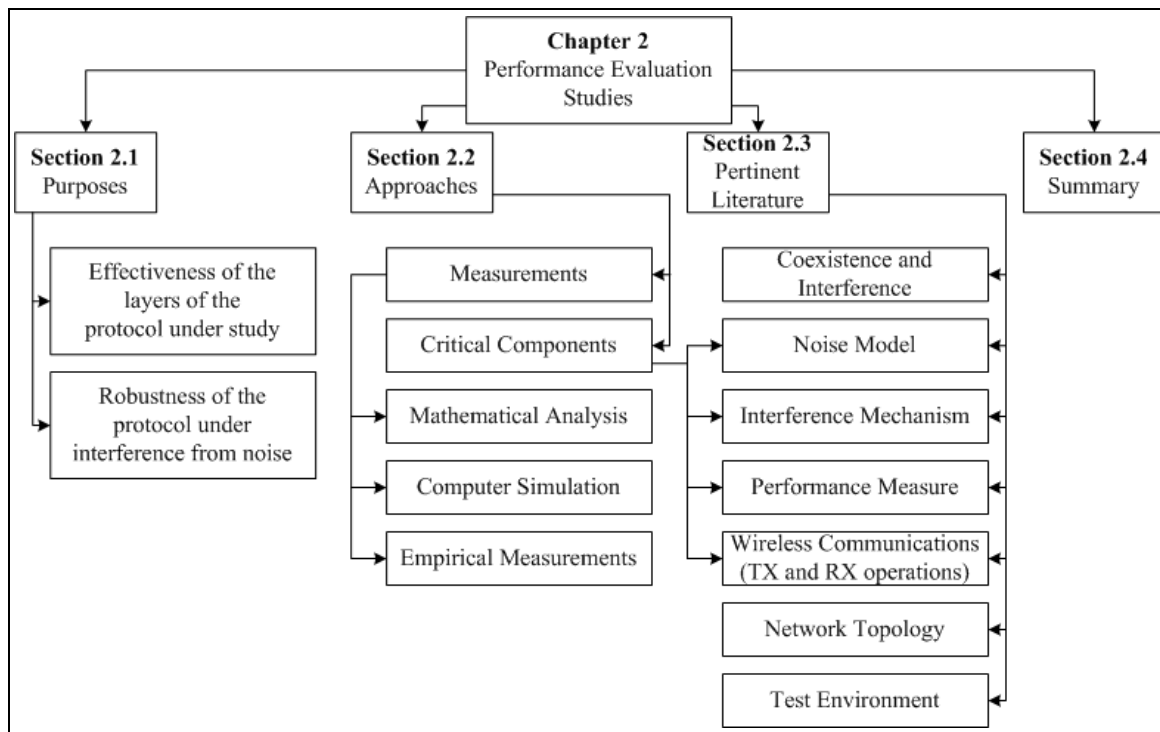


Fig. 2.1 Layout of Chapter 2.

2.1 General Purpose of Performance Evaluation

A review on performance evaluation literature shows that there are two general reasons for requiring a performance evaluation on wireless networking technology.

The first reason is to determine the effectiveness of the wireless protocol for an intended application through assessing the performance of its protocol layers. For instance, [LiBa07] evaluates the performance of IEEE 802.15.4 *medium access control* (MAC) layer by observing the effects of carrier sense multiple access with collision avoidance mechanism, number of network devices, sampling rates, and transmitting cycles on the transmission delay, end to end latency, and packet delivery rate performance measures. Meanwhile, [AISJ06] evaluates the performance of IEEE 802.15.4 *physical* (PHY) layer by observing the effects of data rate, channel noise power, and number of bits per symbol on *bit error rate* (BER) versus *signal-to-noise ratio* (SNR) performance measure.

The second reason is to gauge the wireless protocol in terms of its endurance or robustness in the face of interference from various noise sources present in the environment of an intended application. In other words, performance evaluation is done to quantify the effects of noise upon the wireless protocol performance.

For this thesis, emphasis shall be placed on the latter reason. The treatment of this latter subject can be broken down into involving the following two major components, (a) the wireless technology under assessment, and (b) the noise of interest. For discussion sake, component (a) shall be further broken down to either ZigBee or other non-ZigBee wireless technologies. On the other hand, component (b) can be emitted by either one of these sources; devices using similar or different wireless technology on the same frequency band as (a), or other noise sources, such as microwave oven or other electrical and electronic components.

2.2 General Approaches to Performance Evaluation

There are three common approaches in the assessment of wireless network performance under interference, namely, analytical (or mathematical), simulation, and experimental (or empirical) study [Golm06]. Some explanations shall now follow to distinguish between the terms “noise” and “interference”. A signal is considered as “noise” when it is unwanted (i.e., not the desired signal). The presence of noise may cause interference. When a noise signal interferes with the intended communication (i.e., transmission and reception), interference is said to have happened. Interference is an event or phenomenon that occurs due to a mixture of noise signal and the intended wireless signal. A consequence of interference is network performance degradation. How detrimental is the effects of noise upon the performance of a network adopting the wireless technology under assessment must then be investigated. To do so, performance evaluation will involve the modelling of A) noise, B) interference mechanism, and C) operations of the transmitter and receiver of the wireless technology under assessment, as well as defining some metrics to quantify the performance, known as D) performance measures.

An analytical study normally involves the use of mathematical equations to model elements A) to C). Analytical models are less realistic due to them being simplified by basing on assumptions. A simulation study is often carried out by modelling elements A) to C) in a computer program called a simulator. Parameters of the modelled components in a simulator can easily be modified to see their effects on the performance. An experimental study involves the use of actual hardware for elements A) to C) to measure performance in a real-world scenario. Depending on the implementation design, for example, whether parameters can be easily modified, and whether complete implementation details are known, the empirical results obtained may have restricted applicability, i.e., accuracy confined to the evaluation hardware, equipment, and environment used. Each evaluation approach has its own advantages and disadvantages.

Sometimes one approach may be more useful than the others depending on the requirements of the performance evaluation study. For example, if a fast performance evaluation is required, it may cost less to run a simulator to test for scalability issue than to actually develop, implement, and test many nodes. And at other times, more often is the case, a combination of approaches are used, e.g., a computer simulation is carried out following a mathematical analysis for the purpose of verifying results, such as done in [ShPK07].

According to [Golm06], some of the performance measures that are commonly used in evaluating the impact of noise on wireless system performance are the plot of BER versus SNR, packet loss, delay (sec), and throughput (bits/sec). The choice of performance measure to use depends on the protocol layer where the performance of the system is to be characterized, either at the PHY layer, or at higher layers (MAC and above).

2.3 Pertinent Performance Evaluation Literature

Research on wireless coexistence and interference issues have been extensively explored and published. An interference performance evaluation is a one-way or unidirectional assessment, where two systems are involved, namely the victim/intended signal and the interferer/noise signal. It is a study evaluating the performance of a wireless system in the presence of one or more interference sources. On the other hand, coexistence performance evaluation is where a two-way or mutual interference analysis is carried out. Basically, the role of the victim system and the interferer system in an interference performance evaluation would be switched or reversed, and usually some mitigation solutions/suggestions will accompany the coexistence results. Normally, a coexistence study is carried out to check whether a pair of *radio frequency* (RF) networks (dissimilar or similar in terms of wireless technology) can coexist in a shared spectrum space. The following sub-sections outlined the studies that have been done in covering

interference and coexistence issues, with a highlight on the similarity and difference in methodologies used in the literature and in this thesis.

In terms of a suitable noise model for use in performance evaluation, other wireless technologies (whether similar or different than the wireless technology under assessment) have been used extensively as the choice of noise source to assess the performance of the desired wireless system under interference. For example, the assessment of ZigBee/802.15.4 systems have been done with Wi-Fi/802.11 b or g [GoCR04, SiGr05, Rodr05, PRML06, ABFS07, ShPK07, SPCK07, MuTe08], Bluetooth/802.15.1 [SiGr05, ChGo05, Rodr05, SPCK07], and/or ZigBee/802.15.4 [GoCR04, Rodr05, HoKE07] as interferers. The reason is because other technologies operating within the same frequency band (e.g., the unlicensed 2.4 GHz *international, scientific, and medical* (ISM) band) as the system under study will have an impact on the system's performance. Needless to say, there has been a huge amount of literature that study coexistence or interference issue by bringing in other wireless technology to cause interference to the system under study. Because the coexistence and interference issues are applicable across any wireless protocol (i.e., performance evaluation would not be limited to assessing ZigBee technology alone), for completion purpose, I have listed here some non-ZigBee related publications [GoMo01, SBSL05].

However, there are very few studies and published work on the performance of a wireless system under interference due to electromagnetic noise emitted by unintentional transmitters or non-communication devices. Literature on this subject has covered mostly on radiation concerning microwave ovens. This may be because they present as quick sources to induce *electromagnetic* (EM) noise for carrying out empirical performance evaluation studies. For example, microwave oven has been used as an interferer to ZigBee [SiGr05] and to 802.11 [KaEr97]. However, microwave ovens only present one instance of unintentional transmitters that

will cause broadband¹ *electromagnetic interference* (EMI) to wireless communication systems. An industrial environment, for example, is an area surrounded with various unintentional transmitters such as motors and other electrical and electronic circuits or devices. Any wireless communication system placed within such an area is expected to operate reliably despite the occurrence of broadband EMI. Can a wireless system survive in an industrial setting? In particular, it is not mentioned in the ZigBee specification [ZiA104] nor are there sufficient studies² that show how well a ZigBee system will work in an industrial environment. This thesis contributes to and extends the performance evaluation literature in this aspect. It is critical to understand how ZigBee behaves under broadband EMI to assess its suitability for industrial applications.

The common interference mechanism in performance evaluation literature is done by having the victim system and the interferer system perform their own wireless transmission independently, yet concurrently within the same testing space/environment, thus allowing interference between the two signals to happen in the wireless medium, i.e., over-the-air, such as in [SiGr05], [PRML06], and [SBSL06]. Intuitively, this may be the way because the protocol under study is wireless, so must the medium that carries the noise. However, the author thinks that this method of interference does not provide the experimenter with much control, especially when SNR measurements were to be taken. The exact power of the noise signal interfering with the desired wireless signal would not be easily controlled or determined due to (i) the emanation

¹ Noise contributed by such devices is considered as broadband in the frequency-domain (with a frequency spectrum spanning across tens of MHz [KaEr97]) due to its signal being transient-like or of pulsed nature in the time-domain. In contrast, because the frequency spectrum of noise contributed by a transmitting wireless communication system is band-limited, such noise is considered as narrowband instead.

² The author has found [SGMP07] to be a similar study, which expresses the need to evaluate the impact of impulse noise (due to partial discharge) on ZigBee for deployment in electricity transmission substations. However, the publication only included the design and implementation of a ZigBee system for field trials and some preliminary proof-of-system results to show that the system worked according to its intended functionality.

of noise from a point source, and (ii) the occurrence of reflections within the testing environment. For full control over the noise addition, the interference mechanism employed in this thesis is at the wired circuit level, where interference happens in the wired portion of the receiver circuit through the use of a summing circuit to superimpose the noise signal and the received wireless signal.

To examine the throughput of ZigBee network under broadband EMI, this thesis employs the plot of *packet error rate* (PER) versus SNR as the performance measure. PER has been widely used as a performance metric for evaluating a wireless system under interference as can be seen in [GoCR04, SiGr05, PRML06, ABFS07, ShPK07, SPCK07]. My choice of using PER is in line with the packet-based communication of IEEE 802.15.4 system, as supported by the PER-based receiver sensitivity definition found in [IEEE03].

Owing to PER being the performance measure of interest, this thesis shall adopt the convention followed by publications like [AISJ06, SPCK07, MuTe08], where the terms ZigBee and IEEE 802.15.4 are used interchangeably. Based on the reasons that PER assesses the PHY layer performance of a wireless system (since packets in error are dropped or discarded at the PHY layer), and that the IEEE 802.15.4 standard contains the specification for ZigBee PHY layer, such convention is thus permitted.

According to [IEEE03], IEEE 802.15.4 supports star and peer-to-peer network topology. In line with that, majority of the wireless network performance evaluation literature, such as [SiGr05, PRML06, SPCK07, HoKE07, SABJ07], have used the basic network consisting of two nodes, a transmitter and a receiver, for both the victim system and the interferer system in their coexistence or interference studies. For this thesis, the 2-node topology is employed only for the wireless system under assessment.

Last but not least, with respect to empirical type of performance evaluation, the use of a real environment such as an office [SBSL06], a factory, or an industrial site [FuLa05] may consists of unaccounted-for variables, such as unknown third-party interference [Morr04]. As such, my approach of empirical analysis uses a laboratory simulation that involves real hardware, signal generators, and measurement instruments with connectivity through cables. Such an arrangement boasts control over the testing environment.

2.4 Chapter Summary

This section provides a review of the pertinent past literature on the impact of interference and coexistence on wireless systems, with an emphasis on ZigBee-related studies. The important components for a performance evaluation study have been highlighted with respect to past work and in relation to this thesis. The components are the wireless technology under assessment, the noise model, the interference mechanism, the performance measure(s), and the evaluation approach (analytical, simulation, or empirical). The issue that this thesis is interested in is interference, not coexistence. The focus is not to improve the protocol under study by examining its features, but to test the protocol as is for application in industrial environments. There exist various sources that may cause broadband EMI to the wireless protocol under study in such environments. The severity of its impact on the wireless network performance is judged/measured/quantified through PER.

Chapter 3

SELECTING WIRELESS TECHNOLOGY FOR CONTROL, MONITORING, AND SENSING

This chapter describes the thought process involved in choosing a wireless technology that is best suited for enabling wireless communications in the control, monitoring, and sensing (CMS) application of interest, i.e., industrial machinery. The thought process begins by first specifying the requirements of the desired system. Then, by going through a list of protocol characteristics, the available technologies considered are shortlisted. The protocol with the closest match to the desired characteristics is chosen. The desired protocol characteristics are influenced by the application's system requirements.

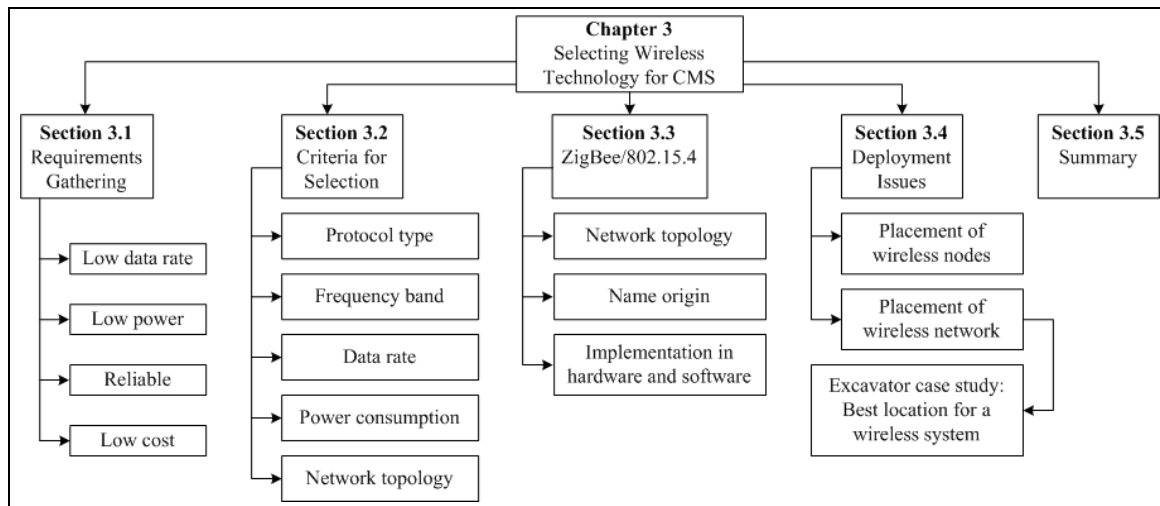


Fig. 3.1 Layout of Chapter 3.

In addition, this chapter provides an overview of ZigBee (the chosen protocol) with an emphasis on the IEEE 802.15.4 standard (which defines the bottom two layers of ZigBee), the backbone that enables low data rate, low-power, wireless data communications. Furthermore, some practical wireless network deployment issues that have direct relevance to the thesis

objectives, along with a case study on a CMS example application, are also presented in the chapter.

3.1 Requirements Specification

There are many types of applications that require the need of a reliable communication, such as tracking, event detection, periodic measurements, automation, control or surveillance with usage/demands in battlefield, medicine and health care, environmental, structural, and vehicular applications. The communication systems used for these applications are mostly wire-implemented. By looking closer at what each such communication system is composed of, some of the major requirements that may aid in the selection of a suitable wireless protocol may be identified. As depicted by Fig. 3.2, for a typical communication system to function, it contains at minimum a main controller, a display mechanism, and several other control modules, all connected through a bus system, such as the controller area network (CAN). These modules in turn are connected through wires or wire harnesses to end devices such as sensors and actuators. The main controller and control modules are the components that are responsible for control and monitoring, while the sensors and actuators are the components that provide sensing and trigger output mechanism, respectively.

Since the sensors play an important role in a CMS application, their needs shall shape part of the system requirements. The numerous sensors that may be present in a communication system include temperature, pressure, vibration, position, speed, rotation, or level. Because a sensor is a simple and dedicated device that measures a specific physical quantity or triggers a high or low value for on/off signalling, it outputs a small or limited amount of data. For instance, a temperature sensor may output a signal to indicate the measured temperature or the rate of change of temperature over a period of time. Thus, the sensor application requires at most several bytes of data, e.g., 10 bytes in total (pg. 206 of [Call03], and pg. 1 of [WoCu01]), to be transferred on

the transmission medium (wire or air). This implies that a sensor application will require a wireless protocol with a low data rate as justified by the size of the sensor output data.

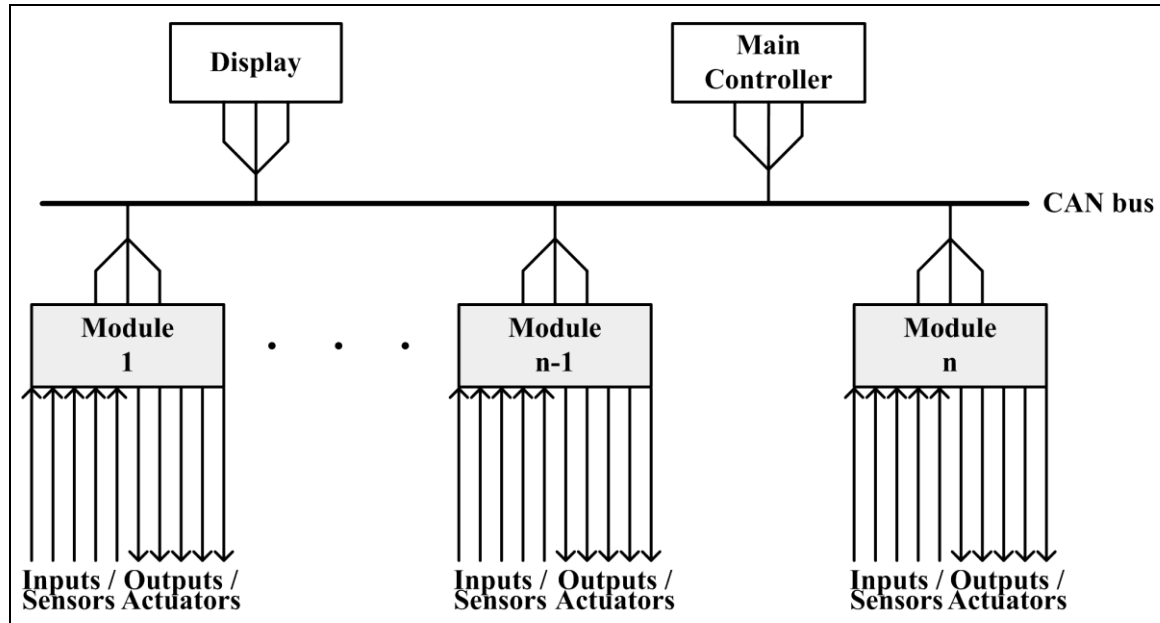


Fig. 3.2 Wired connections in a typical communication system for a CMS application.

Wireless sensor nodes are typically powered by batteries. The sensor nodes are left unattended, and as such, they are expected to work for long periods of time. Furthermore, depending on the application, the sensors may be queried by its respective controllers at regular intervals (i.e., either continuously or periodically for monitoring purpose) or at event-driven basis (i.e., intermittently for alerting or triggering purpose). Therefore, depending on the usage (e.g., frequency of the queries), this may cause the battery in the sensor to drain its power sooner. Since sensors are expected to operate for a long time (perhaps for months or for years before its batteries are replaced), a power-efficient wireless transmission protocol is thus required and preferred for optimizing (thus prolonging) the battery life to power the sensors. Another reason for requiring a low power sensor node is so that batteries that power the sensors do not need to be replaced that frequently. Replacement of battery may be a difficult task in itself. To illustrate, depending on the positioning of the communication system, say around the engine part in an

excavator for example, some machinery may need to be taken apart in order to reach the sensor nodes before replacement of batteries can be done. Thus, a power-efficient protocol would lead to savings in terms of costs for battery replacement and costs for system maintenance.

Another important requirement will be reliable data transfer. CMS systems are critical applications. First, the information returned by the sensors may require prompt actions. Second, human lives may be involved during the operation of the CMS system (in this case, the operator of the industrial machinery and on-site workers or passersby). Thus, data collection and reporting must be timely and accurate. Apart from the application needs and safety of human, the hostile operating environment of the system is another factor that calls for data transmissions with high reliability. The operating environment of a CMS system is harsh in the sense that its internal electrical and electronic circuits and devices and its metal structures (e.g., steel structures surrounding the cab of an excavator) may cause interference to the communication system due to noise emission and reflections, respectively. Hence, a wireless technology that can build reliable communication networks is required so as to provide data timeliness and accuracy, to ensure safety to human during operation, and to survive under harsh conditions.

Last but not least, a requirement that does not contribute to the selection process but is still a major consideration for all applications (whether wired or wireless) is cost. To implement any system, costs are involved in the purchasing of devices, and in the installation and maintenance of the system, to name a few. Moreover, these expenses will increase when more than one such system are being built. Thus, a cost-effective (i.e. low cost) wireless technology can be one means to achieve this requirement.

To conclude, some crucial requirements in terms of a wireless technology for building a wireless CMS system in industrial machinery have been identified. Specifically, a wireless technology that meets the requirements of low data rate, low power, reliable, and low cost is

required. These requirements are sufficiently general to hold true for any CMS system. More importantly, these requirements become the design parameters that are desired when choosing a wireless protocol that is right for use in CMS applications, as will be evident in the next section.

3.2 Criteria for Choosing a Wireless Protocol

In determining a technology that will meet the requirements of the desired CMS application, one can begin short-listing technologies as one goes through the list of wireless protocol characteristics and available options in the following order:

- i. Protocol type: standards-based or proprietary-based,
- ii. Frequency band: licensed or unlicensed,
- iii. Data rate: low or high
- iv. Power consumption: low or high;
- v. Network topology: star or point-to-point (tree or mesh)

The choice to go with for each characteristic depends on the desired design parameters.

For reasons of low cost and interoperability, standards-based protocol is preferred over proprietary protocol. By adopting standards-based technology, companies may save on production costs when making wireless systems in volumes. But if that is not the case, another benefit is that a solution developed from standards-based protocol is ensured to be interoperable. This means that independent of the manufacturer, any test equipment or devices will work with the current solution as long as they all use the same standards-based technology. Moreover, companies may often find a vendor with a lower priced wireless product due to the competition amongst standards-based products manufacturers. With regards to wireless data communications networks, we shall focus on standards defined by the Institute of Electrical and Electronic Engineers (IEEE) standards organization. In particular, the IEEE 802.11 Wireless Local Area

Networks (WLAN) standards and the IEEE 802.15 Wireless Personal Area Networks (WPAN) standards shall be considered for the case of wireless networking in industrial machinery.

For the reason of low cost, unlicensed (or license-free) frequency band is chosen. Wireless communication systems that use a license-free frequency band such as one of the international, scientific, and medical (ISM) bands, can transmit within the limited spectrum of frequencies without needing to pay a license fee. This, of course, comes at the cost of having no protection against interference from licensed and/or unlicensed users sharing the same spectrum since the spectrum is free for all. In particular, the 2.4 GHz ISM band (also known as 2.45 GHz ISM, due to the center frequency of the band's bandwidth) is preferred due to its worldwide availability. The wireless networking protocols that are available at this band are IEEE 802.11 variants b and g, IEEE 802.15.1, and IEEE 802.15.4, where each of these protocols defines the specification for the lowest two layers of the Wi-Fi, Bluetooth, and ZigBee wireless networking technology, respectively.

For the reason of sensor message size in CMS application, wireless protocols with a low data rate are sufficient. From the shortlisted four IEEE 802 standards, IEEE 802.11b and IEEE 802.11g are eliminated because their data rates (with a maximum data rate of 11 Mbps and 54 Mbps, respectively) are unnecessary and considered as overkill for the intended application. The maximum or peak data rates for the two remaining shortlisted protocols are: IEEE 802.15.1 (1 Mbps), and IEEE 802.15.4 (250 kbps).

For the reasons of low power consumption provision and reliable communication through the structure of network topology, ZigBee (IEEE 802.15.4) prevails over Bluetooth (IEEE 802.15.1) as the choice of wireless communications technology for a vehicular CMS application.

In terms of power consumption, the communication protocols for ZigBee and for Bluetooth are designed to consume very low power. However, ZigBee is considered to consume lesser

power than Bluetooth in the following regards: ZigBee power profile is to have battery-operated devices to sleep most of the time during inactive periods and only waking up periodically to check if there is pending messages as oppose to Bluetooth “always on” power profile; most end devices (e.g., battery-powered sensor nodes) use the type of ZigBee hardware called “reduced-function devices”, which allow for further power-saving through the reduced complexity of the ZigBee hardware components; ZigBee low data rate allows for low power transmission (its transmit power ranges between -3 dBm and 10 dBm).

In terms of network topology, ZigBee can support a mesh network, whereas Bluetooth could only support a star topology (known as a piconet) or a cluster of piconets (known as a scatternet). A mesh topology is constructed when each node in the network is connected to two or more other nodes within the network. A piconet can have at most 8 devices (also called nodes), where each of the 7 slave nodes is connected to a central node, called the master node. Meanwhile, a scatternet is formed when one of the nodes in one piconet participates as a slave in another piconet. This network structure supports a communication of more than 8 nodes since several piconets are interconnected. The main disadvantage of a star topology is that the master node is the central point of failure. The network becomes completely unreliable when the master node fails, thus causing all its attaching slave nodes to no longer have communicability. On the other hand, the failure of a single node in a mesh network will not make the network unreliable since the connectivity of one node to at least two or more other nodes (i.e., the notion of mesh) provides redundant paths for the data to take on to reach its destination. As a consequence, a mesh network is said to be very reliable as it can continue to operate despite the link’s or node’s failure. Aside from the ability to self-heal (i.e., reroute data in the event of a broken link/connection or a failed node), a mesh network can also self-form (i.e., automatically discovers and associates newly added nodes or disassociates nodes that have been removed). Furthermore, a

mesh network topology can offer range extension to interconnect nodes that are out of the range of ZigBee's maximum point-to-point transmission distance. This is achieved by adding additional in-range nodes in between the two out-of-range nodes to act as routers to enable the data to reach its destination over several hops (i.e., by going through several intermediate nodes).

Therefore, Bluetooth is eliminated leaving ZigBee as the best wireless communication technology candidate for CMS applications. Table 3.1 provides a comparison between the said technologies in terms of their protocol characteristics.

Table 3.1 Comparison highlighting specific ZigBee and Bluetooth characteristics.

Characteristics	ZigBee (following IEEE Std 802.15.4 - 2003) [ZiA104, ZiA108, IEEE03]	Bluetooth [Blue08b]
Standard	IEEE 802.15.4	IEEE 802.15.1
Single-hop transmission range	1 - 100 m	1- 100 m (power-class dependent)
Price per chip	USD\$ 2	USD\$ 3
Frequency Band	868/915 MHz PHY <ul style="list-style-type: none"> • 868-868.6 MHz • 902-928 MHz 2450 MHz PHY <ul style="list-style-type: none"> • 2400-2483.5 MHz 	2.4 GHz ISM band <ul style="list-style-type: none"> • 2400-2483.5 MHz
Number of Channels	27 channels (numbered 0 to 26) <ul style="list-style-type: none"> • 1 channel @ 868 MHz band • 10 channels @ 915 MHz band • 16 channels @ 2450 MHz band 	79 channels (numbered 0 to 78)
Peak Data Rate	20 kbps @ 868 MHz band 40 kbps @ 915 MHz band 250 kbps @ 2450 MHz band	1000 kbps (for Basic Rate) 3000 kbps (for Enhanced Data Rate)
Power <ul style="list-style-type: none"> • Consumption • Maximum Output or Transmit Power and Expected Range • Battery Life 	60 mW active @ 0 dBm 5-2000 mW sleep (mode dependent) <p>Maximum output power is conformed to local regulations:</p> 868 MHz band Europe: 25 mW 915 MHz band United States: 1 W 2450 MHz band United States/Canada: 1 W Japan: 10 mW/MHz Expected range to be dependent on the environment, antenna, and operating frequency band.	80 mW active @ 0 dBm 100 mW sleep [Holl04] <p>Maximum output power is dependent on device power class: Class 1: 100 mW Class 2: 2.5 mW Class 3: 1 mW Expected range to be 100 m, 10 m, and 1 m, respectively</p> Days
Modulation <ul style="list-style-type: none"> • Digital • Spread Spectrum 	Binary phase-shift keying @ 868/915 MHz PHY Offset quadrature phase-shift keying @ 2450 MHz PHY Direct Sequence	Gaussian frequency-shift keying (for Basic Rate) Gaussian frequency-shift keying and Phase-shift keying (for Enhanced Data Rate) Frequency Hopping or Adaptive Frequency Hopping
Error Detection	Cyclic redundancy check (CRC)	Header error check, CRC
Error Correction	None	Forward error correction
Topology	Star and peer-to-peer (the latter allows for mesh networking)	Piconet
Number of nodes per network	$2^{16} = 65536$ [ZiA108, FAQ 24]	$2^3 = 8$ (1 master, 7 slaves)
Receiver bandwidth	2 MHz	1 MHz
System resource/ Stack size	4 - 32 KB [ZiA108, FAQ 17]	250 KB

3.3 A Description of ZigBee/802.15.4

The ZigBee protocol stack consists of a 4-layered architecture (Fig. 3.3). The bottom two layers, i.e., the *physical* (PHY) and the *medium access control* (MAC) layers adopt the IEEE 802.15.4 standard (which was approved in May 2003), whereas the top two layers, i.e., the networking, which includes security, and the application layers follow the specification defined by ZigBee Alliance (which was first ratified in December 2004). ZigBee Alliance is a body comprised of companies that joined forces to define and maintain the ZigBee specification, and to provide interoperability and conformance testing specifications. Although there has been an updated version to the mentioned two specifications, namely IEEE 802.15.4-2006 (published in September 2006) and the ZigBee-2006 (published in December 2006) and ZigBee-2007 (published in January 2008), this thesis shall follow the description found in the first version of both specifications [ZiA104, IEEE03] in lieu of the start date (August 2005) of the research work.

The responsibilities of each layer on the ZigBee stack are described briefly in a top-down manner as follows:

- The application layer consists of an application framework within which end manufacturer-defined application objects are hosted, a ZigBee device object, and an application support sublayer. The ZigBee device object is responsible for network device role definition, binding requests initiation and/or response, establishment of secure relationship between network devices, network device discovery, and service discovery (i.e., determination of features and services supported by devices on the network). The application support sublayer is responsible for maintenance of table for binding (i.e., the ability to communicate with other devices of the same services and needs in the network) and message forwarding between bound devices.

- The network layer is responsible for providing two services, a data transmission service (which includes preparation of network frame; application of security to frames; and routing of frames) and a management service (which includes device discovery and configuration; network formation, joining and leaving; discovery and maintenance of routes between devices; discovery of one-hop neighbours and storage of relevant neighbour information; and control of receiver activation and reception duration).
- The medium access control layer is responsible for controlling access to the radio channel, where carrier sensing multiple access collision avoidance mechanism is employed. It also provides MAC layer security, flow control (through acknowledgement and data packets retransmission), frame validation, and network synchronization. In terms of security, there is currently no forward error correction in the specification, only error detection via 16-bit cyclic redundancy check (CRC).
- The physical layer is responsible for radio transceiver activation and deactivation, receiver energy detection, received packet link quality indication, clear channel assessment, selection of channel frequency, and data transmission and reception, besides interfacing between the physical radio channel and the MAC layer. There are two PHY layers that correspond to two ranges of operating frequency; 868/915 MHz and 2.4 GHz. The modulation technique that it uses is *direct sequence spread spectrum* (DSSS) using binary phase-shift keying for 868/915 MHz PHY, and DSSS using offset quadrature phase-shift keying for 2.4 GHz PHY. Since the 2.4 GHz PHY is available for global use, when we refer to ZigBee, we are referring to the 2.4 GHz ZigBee signal, that can transmit in one of the 16 channels (centered starting from 2405 MHz, with channel separation of 5 MHz), and a 6 dB bandwidth of 2 MHz.

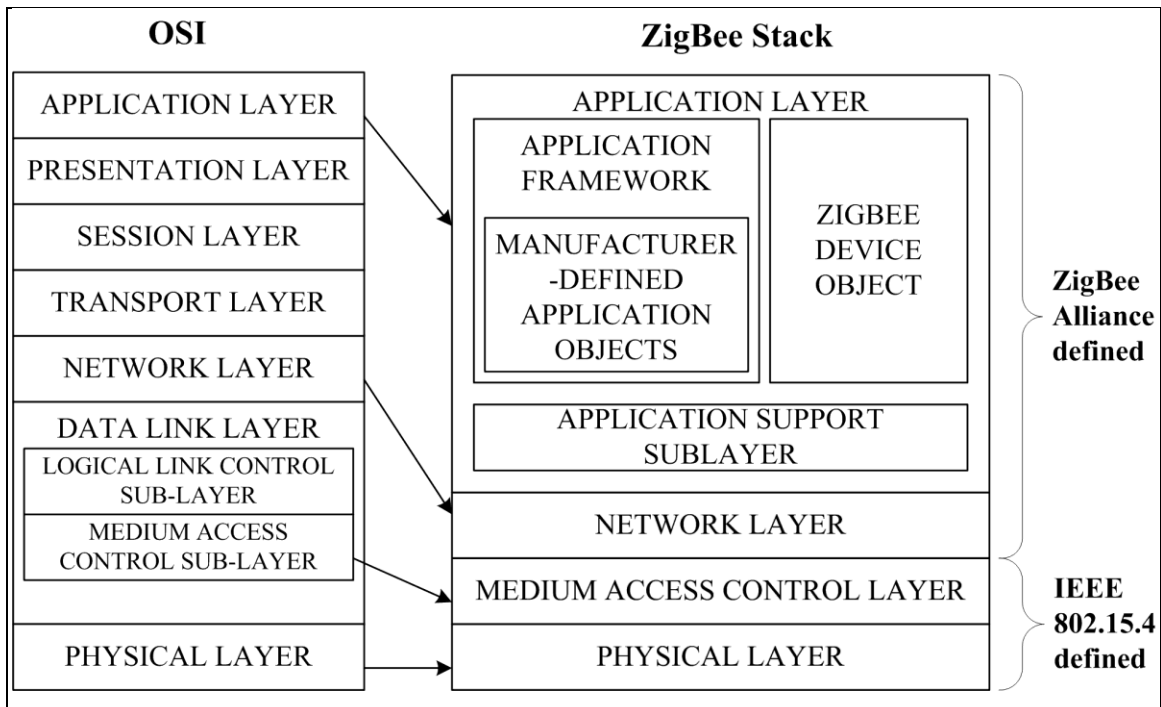


Fig. 3.3 The ZigBee stack in comparison with the Open Systems Interconnection model.

3.3.1 ZigBee Network Topology

A ZigBee network can be constructed in various configurations, taking the form of either one of the following network topologies: star, mesh, or a combination of the two, known as cluster-tree. In terms of physical devices, ZigBee defines the device types based on hardware complexity using the notion of *reduced-function device* (RFD) and *full-function device* (FFD). Furthermore, ZigBee defines the device types based on their network roles and responsibilities: a ZigBee coordinator (ZC) starts up a network and assigns addresses to newly attached devices; a ZigBee router (ZR) routes messages as well as distributes addresses to its attaching end devices; and a ZigBee end device (ZED), basically a sensing device is normally found at the edge of a network, as shown in Fig. 3.4. Both ZC and ZR are FFDs, while ZED can be either an FFD or an RFD.

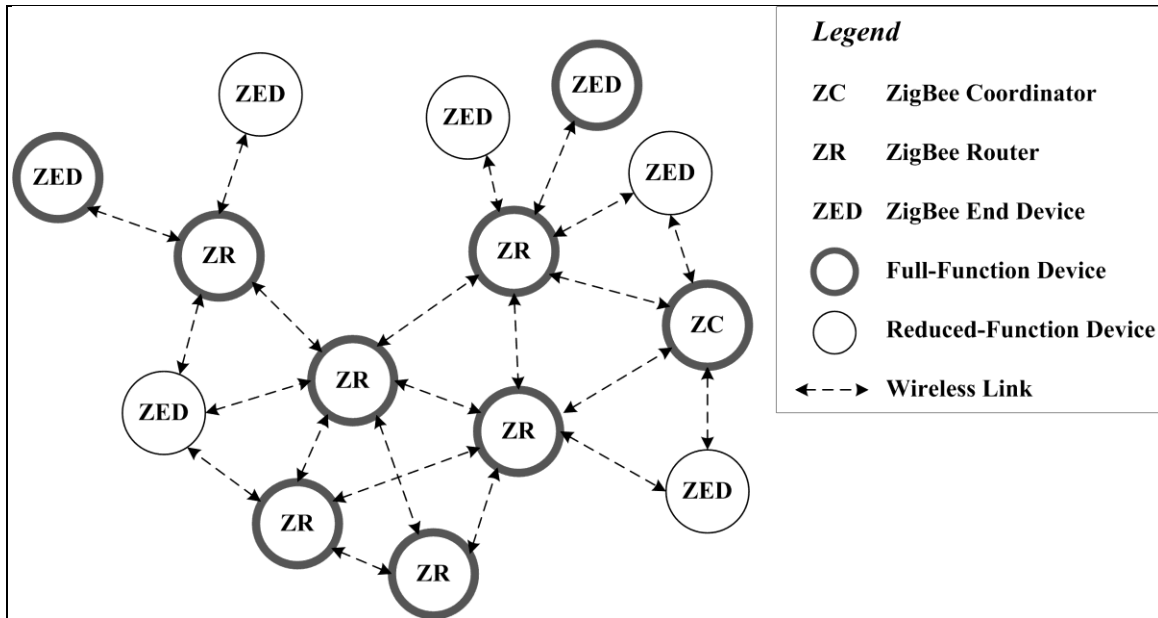


Fig. 3.4 A topological representation of a ZigBee mesh network (i.e., physical representation is not implied) illustrating its devices' network role and hardware type.

3.3.2 Name Origin

The name ZigBee was chosen at random to be one that does not infringe on established trademarks, contrary to the explanation once circulated by ZigBee Alliance on its frequently-asked questions section that it was derived from a term known as the “ZigBee Principle”, which denotes the silent form of communication (i.e., the zig-zag dance pattern) used by bees to relay information (such as location, direction, and distance) of newly-discovered food sources to other bees. Although such principle was thought to be analogous to the invisible links in a wireless network, it is purely a myth as such principle is non-existent in apiology, the scientific study of honeybees [Rupe04].

3.3.3 ZigBee Implementation: Hardware and Software

A wireless sensor node consists of components such as the sensing/actuator unit, the processing unit, the transceiver unit, and the power supply unit. When employing a radio technology, say ZigBee, the processing unit and the transceiver unit are the units that contain the

ZigBee software stack and radio firmware respectively. From the hardware perspective, ZigBee products have evolved from evaluation board with separate transceiver and microcontroller chips to a system-on-chip solution where the transceiver and microcontroller resides on a single chip. Meanwhile, from the software perspective, there are 802.15.4-compliant products as well as products that contain a fully-certified ZigBee stack, which normally requires a costly license to be purchased by the user.

Numerous proprietary products have emerged as a consequence of the introduction of ZigBee. Proprietary platforms range from product that provides simple wireless connectivity with a reduced number of protocol layers (e.g., Freescale's *Simple Media Access Controller (SMAC)*) to full ZigBee-similar stack product (e.g., Ember Corporation's *EmberZNet PRO* [Embe09], or Microchip's *MiWi* [Micr09]).

3.4 Practical Deployment Issues

The author has identified two deployment issues that would require consideration prior to deploying wireless communication system within any CMS application: 1) placement of wireless nodes, and 2) placement of wireless network. The mentioned two deployment issues matter due to real world constraints that exist, such as electromagnetic noise that may severely degrade wireless system performance or damage electrical devices, and physical restrictions such as limited available space for system installation that may affect size of nodes separation, and metal blockages/obstructions that cause redirection of waves.

3.4.1 Placement of Wireless Nodes

At the very least, the first issue concerns with knowing how many and how far apart wireless nodes are to be placed in order for the communication system to be effective. This is a crucial and practical deployment issue as it involves a trade-off between cost and reliability, the two being the rationale for putting wireless nodes apart. When wireless nodes are put far apart, the number of

nodes within the allotted system space³ reduces, and in turn lowers cost involved for purchasing nodes. However, due to the large distance between nodes, reliability of the communication system may be sacrificed. On the other hand, when wireless nodes are densely populated within the given system space, reliability of the system will definitely be enhanced. However, this comes at the price of having to increase the cost to purchase the large number of wireless nodes.

The author proposes a method to balance the trade-off associated with the placement-of-nodes issue. To reach a state where cost is minimized and yet reliability is not compromised, the idea is to determine the minimum number of nodes within the system space that can achieve the amount of reliability desired. The author proposes that if the maximum *nodes separation*⁴ associated with the desired reliability (in terms of a tolerable packet error rate) can be known, then the mentioned goal can be achieved. Furthermore, the author proposes to exploit the knowledge of performance plots that associate PER and SNR_{RX} ⁵ for various nodes separation. More precisely, the maximum nodes separation can be ascertained given (i) the range of adjustable transmit power for the wireless node, (ii) the amount of noise present within the environment of the communication system, and (iii) the plot of nodes separation (or distance between two nodes) versus SNR_{TX} ⁶ that is valid for the amount of reliability desired. The plot that relates distance between nodes and SNR_{TX} is derived from at least three plots of PER versus SNR_{RX} , each gathered over a different nodes separation, in order to give at least three points to draw the plot. Finally, by counting the number of wireless nodes that fill the system space, where each node has been placed at the maximum distance apart from each other, the minimum number

³ The term “system space” is used to mean the amount of space (volume) available within the communication systems where wireless nodes can be fitted.

⁴ The term “nodes separation” is used to mean the distance where nodes are placed apart from each other.

⁵ The term “ SNR_{RX} ” is used to imply the signal-to-noise ratio measured at the receiver node.

⁶ The term “ SNR_{TX} ” is used to denote the signal-to-noise ratio occurring at the transmitter node.

of nodes can be found. In this way, a bound of the actual number of nodes possible in a factory floor due to such type of noise can be established.

3.4.2 Placement of Wireless Network

When converting a communication system from wired to wireless, a deployment issue to consider is the placement of wireless network. Depending on what requirements are specified for the new wireless system, none or some of the system may be preserved. Some of the requirements that affect the part or parts of the system that will have their wires or wire harnesses removed include reusability, redundancy, and minimal or zero user intervention.

3.4.2.1 Excavator case study: Best location to place a ZigBee network

For this case study, the excavator, an off-road vehicle is chosen because (i) it is an example of a vehicular type of CMS application operating in industrial environment, (ii) it can be easily accessible, as they are used all around the world in construction sites, and (iii) due to the previous reason, it is also a source for collecting noise samples. Referring to Fig. 3.2, the communication system for an excavator would consist of modules such as display, front and rear input/output modules, boom control, throttle control, and drive control that are commonly interconnected through a controller area network (CAN).

Figure 3.2 shows that most of the wired connections are between the end devices (input and output components such as sensors and actuators) and the various modules present within the industrial machinery. The wires from sensors are routed and may be force-fitted around the vehicle to its respective controller modules. Since the cause for failures in the vehicular system is attributed to wires, which can break under the harsh operating conditions or over time depending on the wire and wiring quality, it is best to place a wireless network between the modules and its input/output components to replace such wires and wire harnesses, as depicted in Fig. 1.1 in Chapter 1. This placement is ideal as the existing CAN vehicular network system can be

preserved, which means the capital investment of the current design can be reused when switching over to a ZigBee wireless system. To incorporate ZigBee into such existing network structure, the author suggests (though not explored in this thesis) to use CAN over ZigBee, which could be implemented through (a) an addition of a software (ZigBee) stack within the microprocessor employing CAN, or (b) a protocol convertor. The first method would require a software redesign and is thus a partial module-reuse solution, whereas the latter method is meant to be a plug-and-play device and is thus a total module-reuse solution. Diagrams depicting other possible placements of a wireless network to the mentioned communication system, together with a brief highlight of their respective advantages and disadvantages, can be found in Appendix C.1.

3.5 Chapter Summary

There are many wireless protocols available to choose from, but the most suitable protocol can only be selected through performing an engineering analysis of the requirements of the system/application of interest.

Chapter 4

MODELLING NOISE FROM INDUSTRIAL ENVIRONMENT

Noise exists in real world. One may ask why there is a need to study noise. In my case, I would like to find out how severe noise affects the intended signal transmission. Noise affecting both analog and digital system operations may have either temporary consequence (e.g., causing errors or flipping of bits) or permanent consequence (e.g., destroying electrical or electronic components). Furthermore, such studies can also aid in the creation of noise control methods (author's note: this area is not explored as it is not part of the thesis objective) to suppress or eliminate the negative noise effects.

The objectives of this chapter are to define a subset of the set of noise signals, that is of interest to this research work, and to provide a progression showing the different tasks that are involved, that leads towards the type of analog noise signals used in my research to empirically gauge the effects of broadband electromagnetic interference on a ZigBee system. ZigBee is the wireless protocol proposed for vehicular applications, as determined in the previous chapter. The steps leading to that include selection of a model that best represents the noise of interest, noisy channel characterization involving measurement and analysis of one type of real noise emission from vehicular electronic and electrical component, software synthesis and hardware emulation of the selected type of noise signals, and finally analysis of the synthesized noise signal to ensure conformance to the selected noise model.

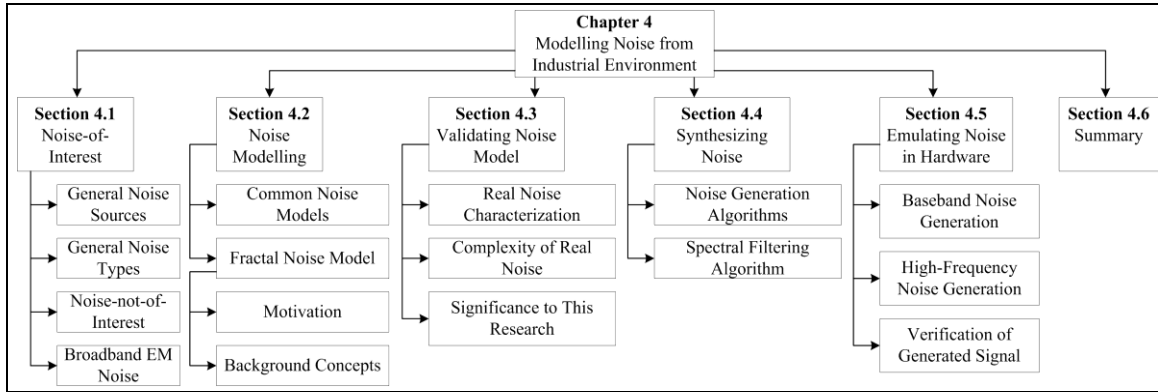


Fig. 4.1 Layout of Chapter 4.

4.1 Electromagnetic Interference Mechanism

Noise generally refers to fluctuations or unexpected deviation of a quantity of a particular variable. The term ‘noise’ can have a different meaning depending on the context of where it is being used. For example, when dealing with, say, vibration and sound, noise in this context means acoustic noise or audible noise; whereas in image processing, noise refers to pixels of the image that have been corrupted either during its acquisition, transmission, or reproduction. In this thesis, noise refers to “unwanted signal” that interferes with the communication (transmission and reception) of the intended wireless signal.

More specifically, the sources that contribute to such radio frequency noise in the operating environment of vehicular systems will first be discussed, followed by discussing the noise emissions by the sources from the point of view of being either narrowband or broadband. Finally, the reasoning for excluding narrowband noise and for concentrating on noise that causes broadband electromagnetic interference is provided.

4.1.1 Noise Sources in General

It is crucial to be aware of the sources of radio frequency noise as knowing them will allow:

- i. better placement of wireless sensor nodes. Wireless system designers can plan for a proper arrangement of nodes having known what noise source is present where.
- ii. determining which noise removal methods will be more effective. If a noise source is known to present high enough concentration of noise to affect the intended signal transmission and reception, it should be filtered or reduced.
- iii. capturing solely the noise signal for analysis/assessment/modeling purpose. This can be achieved by turning the specific noise source on.

According to [Anri03, BiHa05, HoRo06, Kins02, Tekt01], factors contributing to wireless signal degradation/weakening/fading (effects of interference), which affect performance of wireless networks in terms of possible loss of data, are:

A. Interference from “radiating sources”. Types of radiating sources include:

- i. Man-made sources that intentionally emit data-carrying signals (with known/definitive spectral profile), whether in-band (on-frequency sources or off-frequency sources that produce on-frequency effects), out-of-band (off-frequency sources): Meaningful transmissions such as those coming from radio broadcast stations, satellite television broadcast, garage door opener, and amateur radio. In addition, wireless communications devices and systems such as cellular and cordless phones, electronic toll collection systems, radio frequency identification systems, wireless Internet service provider systems, wireless (keyless) entry system, and wireless systems such as Wi-Fi, ZigBee, and Bluetooth;
- ii. Man-made sources that unintentionally emit non-data-carrying signals (with unknown/not definitive spectral profile): Electrical and electronic devices in household tools or

- appliances such as a table saw and a microwave oven, computers, medical equipment, vehicles, and industrial machinery such as ignition systems and powerful motors;
- iii. Natural sources that contribute to electromagnetic activity in terms of background noise: Natural phenomena includes lightning, earthquakes, and sunspots;
 - iv. Other sources: First is where the source is unknown since the produced signal is a product of two or more radio frequency signals being mixed and transmitted unintentionally, and second is a source of malicious nature intentionally transmitting signals to jam another.
- B. Interference due to “channel structure” affecting signal propagation:
- i. physical obstructions/blockages from indoor, outdoor, and specialized environments (whether natural or man-made, static or mobile) and their material/structural properties such as density, thickness, and spacing;
 - ii. the level of moisture during weather or atmospheric conditions such as rainfall and fog.

4.1.2 Types of Noise Emission and Examples of Sources

Since the thesis focuses on vehicular system as the target application, this section shall establish the different types of noise emission that have relevance in such context, with the discussion being centered around electromagnetic noise and not acoustical noise.

Vehicular systems include tractors, excavators, and other off- or on-the-road machinery. These systems have been designed to operate within a certain degree of electromagnetic compatibility (the ability of the electrical or electronic equipment under test to reduce its own unwanted emission and to endure the effects of interference caused by others) as regulated by national and international regulatory and standards agencies. Examples of the latter include the International Special Committee on Radio Interference (CISPR), which is a technical committee of the International Electrotechnical Commission, and the International Organization for

Standardization (ISO). In particular, I have a copy of CISPR 25 and ISO 14982, courtesy from my industrial sponsor.

The CISPR 25 [IEC02] standard describes the measurement methods and limits for conducted and radiated electromagnetic emissions from vehicular electrical and electronic components for the purpose of protecting on-vehicle receivers and systems. Meanwhile, the ISO 14982 [ISO98], a conformance standard, specifies the measurement methods for electromagnetic compatibility evaluation applicable to machinery in the fields of agriculture and forestry. Both standards characterize noise emissions or radio disturbance from vehicular electrical and electronics components to fall into two categories: narrowband emission and broadband emission. Such categories are helpful when assessing electromagnetic interference effects.

A noise emission is termed *narrowband* when the bandwidth of its frequency spectrum span less than that of the intended receiver. Sources in a vehicle that contribute to narrowband emissions include devices that incorporate oscillators or clock generators [ISO98, IEC02]. On the other hand, a noise emission is termed *broadband* when the bandwidth of its frequency spectrum span greater than the operating channel width or span across multiple channels of the receiver. Some of the vehicular sources that emanate broadband emissions are ignition system, alternator, and dc motors such as for blower, wiper, and washer pump, and power seat [ISO98, IEC02].

4.1.3 What is Not of Interest and Why

This section shall establish the noise signals that are not of interest to my research yet exist in the working environment of vehicular machineries. The noise signals are namely acoustic noise, signals from other wireless transmitters, and narrowband electromagnetic noise.

Acoustic noise [Bell82] is referring to unwanted sound waves emission from a device or machinery. An example of a context where such noise is of interest is to the mechanical engineering studies of a system, which is not the scope of this thesis.

Signals from other wireless transmitters are useful for their own intended systems, however, may be seen as noise to the intended signal transmission when they are present within the same frequency band. In the event of crosstalk, such signals can contribute to either co-channel interference or adjacent channel interference. These signals are not of interest because their interference effects upon the intended signal transmission have been researched (as mentioned in Chapter 2).

Now, with respect to the type of electromagnetic noise that matters for this thesis, narrowband noise is not of interest. This is because such noise may be eliminated through the proper application of filters, provided the shape of the narrowband noise spectrum and its frequencies are known. Instead of filtering the noise, another alternative would be to switch the operating channel of the wireless communication system to a different band, one that is clear from the detected narrowband noise. In practical situations however, a band does not always stay clear from noise. In such event, an adaptive changing of frequency band algorithm may be employed.

4.1.4 What the Thesis Focus is and Why: Broadband EM Noise

This thesis focuses on *electromagnetic noise*, EM. More precisely, electromagnetic noise that is of broadband nature is of utmost interest. Such distinction has to be made clear because EM noise is inclusive of narrowband noise. An example of narrowband emission is the single frequency 60 Hz emitted by light fixtures coming from being lit by a 60 Hz alternating current power supply. In relation to the noise sources introduced in Section 4.1.1, the broadband EM noise that this thesis focuses on would be emitted by those stated in A(ii), i.e., sources that radiate unintentional non-data-carrying signals.

Broadband EM noise is the central focus because it has more severe consequences compared to narrowband noise. It is more severe in the sense of noise control, and in the sense of noise effects. In terms of noise control, filters will not completely eliminate broadband noise because of a filter's finite bandwidth (determined by the cut-off skirt). Also, switching to a clearer channel is not always feasible as broadband noise may wipe up all available frequency channels for the wireless system. In terms of noise effects, broadband emission causes errors that occur in bursts and not at random bits, thus rendering error correction harder to perform.

4.2 Noise Model Selected/Employed: Fractal Noise

This section provides a brief account on some of the more commonly used models for representing interfering signals that have been used by researchers in the field wireless communications. The primary goal of the section is to introduce a simple model, based on fractal concepts, for broadband electromagnetic noise. However, such introduction would not be complete without the inclusion of justifications for selecting such noise model, the identification of what is lacking in the common noise models, and the language and notations for understanding the fractal-based model.

4.2.1 Commonly Used Noise Models in Communications

In a communication system, there are different elements that can be modelled. Examples of the elements include the transmitter, the receiver, and the channel. The element that I am interested in is the channel. With respect to the communication channel, there are two factors that provoke distortion to the original character of the wirelessly transmitted signal as it arrives at the receiver, namely congestion and link corruption [CoSI05]. Errors due to congestion are due to channel bandwidth contention and happen at the network layer of the wireless protocol. Modelling the network behaviour of the wireless protocol is not part of the thesis focus and thus is mentioned here only for completion. On the other hand, errors due to link corruption are due to namely wave propagation phenomena (such as reflection, diffraction, scattering, and fading) and noise within the channel [KaWi05] and happen at the physical layer of the wireless protocol.

Wave propagation phenomena cause loss of power (path loss) and attenuation, whereas noise causes interference to the transmitted signal (pg. 93-94 of [KaWi05]). The distortion caused by the link-corruption-related factors is in the form of bit errors in the received signal, either with a random or a bursty error pattern. The received signal, $r(t)$ can thus be represented as follows:

$$r(t) = s(t) + n(t)$$

where $s(t)$ refers to the transmitted signal after having experienced path loss and attenuation as it propagates to the receiver, and $n(t)$ refers to the noise signal. The two components within a communication channel that can be modelled are thus the signal propagation and the noise. So, in channels where multipath and fading is the source of distortion in the transmitted signal, the signal propagation is modelled using either the Rayleigh or the Rician distribution [JeBS02, KaWi05]. For characterizing errors due to noise in communication channels, a common model used is the *additive white Gaussian noise* (AWGN). Because the nature of the noise is such that it exists independent of the wireless signal, the noise contribution is thus added to the signal, hence

the term additive in AWGN (pg. 70 of [Seyb05]). AWGN is a stochastic or random noise signal whose amplitude levels follow a Gaussian distribution in the time-domain. Meanwhile, its *power spectrum density*, PSD is constant and spans across a wide bandwidth in the frequency-domain. This is due to the noise samples in time that fluctuate without any correlations or that they are time-independent (thus the term white).

The focus of this section (in line with the thesis) is on models to represent the noise signal component, and not on models to represent a mixture of noise signals. Since the simulation for evaluating the performance of the chosen wireless protocol takes place in an anechoic chamber and without any obstacles in between the pair of transmitter and receiver nodes, the $s(t)$ component's path loss can be represented by the Friis free-space equation (pg. 93 of [KaWi05]) and the attenuation (if any) due to the cable used between the receiver antenna and the receiver board can be accounted for through measurement.

The nature of the noise process needs to be known when modelling noise (pg. 10 of [JeBS02]). Furthermore, because of the apparent randomness of physical noise, this signal can be well-modelled by a random process. As such, stochastic-based models will be used to represent the complexity of the noise process. Based on these ideas, the common noise models fall under two general categories based on the shape of their power spectrum density: namely, uncorrelated noise models or noises with a flat PSD, and correlated noise models or noises with a non-flat PSD. Correlated noise, on the other hand, can have a short tail or a long tail power spectrum.

Real noise is not white (if stated otherwise, this would mean that the noise signal has infinite energy, which is unrealistic in a real-world setting) [Kosk06]. Real noise is more abrupt or impulsive, and thus its characteristics are not well described by the common AWGN model or any other white noise model with a different amplitude distribution. Such nature is better characterized by a correlated noise model (one such being fractal with power law and long term

dependencies) than modelling after uncorrelated noise alone. Furthermore, according to [CoSI05], frame errors in real-world wireless channels are mostly time-correlated (i.e., correlated and non-stationary⁷ locally), which render correlated channel models as more suitable for representing link corruption errors. The correlated noise models from fractal theory, such as monofractal and multifractal noises, have the benefit that they are no more complex than and at least as manageable computationally as the uncorrelated noise model [Worn96].

Last but not least, I have to emphasize that the common noise models can still be used and are still being used, but when it comes to the type of noise (impulsive) that is of interest to my thesis, it is critical to depart from the traditional models for a better and more accurate representation.

4.2.2 Why Employ Fractal Noise Model

The following consists of evidence (from nature) and indications that point to why I think that the noise of interest (broadband EM industrial noise) is fractal, or in other words, models from fractal theory are suitable:

- i. The dominant idea that drives or starts my research came from observing the effects from sparking event. The event (e.g., sparks from within a DC motor) is a random or stochastic process, where sparks produce transients (the unwanted signal) that cause electromagnetic radiation (the broadband noise). Transients composed of succession of impulses (pulsative) that occur at different time intervals. Owing to that, noise frequency content is an exponential decay, in other words, a broadband power law distribution. Since each spark is different, the frequency content has a different power law fall off or slope. Evidence of sparks producing wide spectrum electromagnetic radiation can be recalled from the history of wireless communication, where Marconi

⁷ Strict-sense stationarity (sss) describes a process whose probability density function does not change with time/space. All the statistical moments must be the same. Wide-sense stationarity (wss) describes a process where only the first two moments must be constant (mean, μ , standard deviation, σ , auto-correlation, ϕ_{XX} , and cross-correlation, γ_{XY}).

had used electromagnetic waves (with the use of an improved spark gap transmitter), to transmit a message wirelessly across the Atlantic ocean in 1901 (pg. 436 of [BuHe04]).

- ii. Electrical noise, also known as intrinsic electronic noise found in acquisition and distribution electronic system (e.g., vehicular electronic components) consists of thermal (white) noise, shot (white) noise, and flicker ($1/f$ or pink) noise [Kins02, Kins04].

Due to the above characteristics of noise in nature, fractal-based and wavelet-based noise models are suitable to model physical broadband EM noise emitted by electrical and electronics components in a vehicular system because they can cover both uncorrelated and correlated form of noises, as well as noises with single or multiple fractal dimension. For this thesis, the fractal-based model that I use focuses on using stochastic, statistical, self-affine fractals such as monofractal noises to produce synthetic broadband EM noise.

This thesis uses two methods to arrive at a model for modeling the noise of interest. The first method is analytical-based (with details elaborated in the previous paragraph). It provides real-world evidence that point to fractal model being suitable for modeling the noise of interest as well as the advantages of using such model. The second method is empirical-based where real noise signals have been captured and analyzed. The results of the captured real noise signals analysis show that the character of the noise signal is multifractal in time. As to whether the proposed model can model noise signal naturally, the author will let the results from the real noise capture experiment to speak for themselves. Please refer to sections 4.3.2 and 4.3.3 (pages 66-70) for the results of the empirical-based method and its significance to the type of noise signals used in the performance evaluation experiments of the thesis.

4.2.3 Relevant Fractal Concepts Revealed

The following reveals the relevant fractal terms or concepts and their descriptions to aid in understanding the contents of this chapter and the core of the thesis.

4.2.3.1 Time-Domain Signal Properties or Features

The following shows some of the time-domain properties of a fractal signal:

In terms of signal type, a deterministic signal or a stochastic signal can be a fractal.

Description: A *deterministic* signal is one where its future values can be known exactly (i.e., with probability = 1), provided there is sufficient information regarding its past [Bruc01].

Description: A *stochastic* signal is one where its future values cannot be known exactly even if its past information is available [Bruc01].

Furthermore, if such type of signal exhibits the property of scale invariance, then the signal is definitely a fractal.

Description: A *scale-invariant* signal implies that the signal is invariant under linear transformations (i.e., magnification or reduction). In other words, scale invariance means scale independence, where the statistics of the signal or process do not change whether on short or long time scales [PeSa88, Worn96, Turc97, Kins04].

Figure 4.2 shows that scale invariance is the key to distinguish a fractal signal from a non-fractal signal even though the origin of each signal is a random process. Unlike the brown noise signal, the first-order autoregressive signal is not a fractal since it does not scale no matter what scaling factor is used.

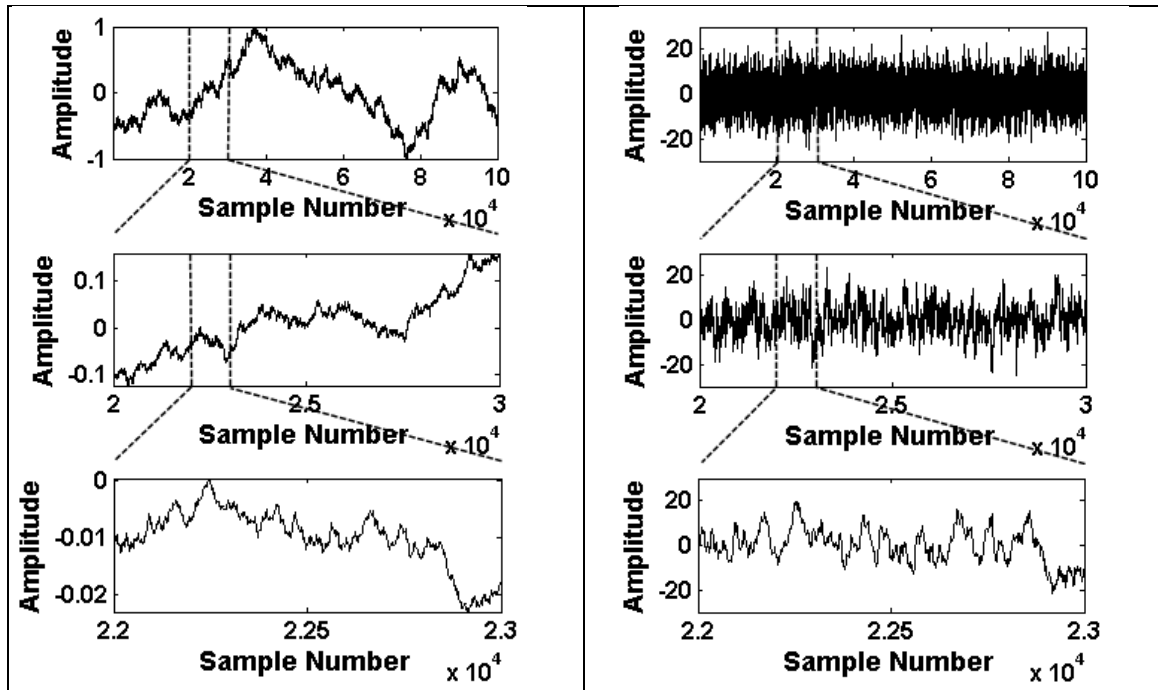


Fig. 4.2 Taking an instance of (top left) a Brownian motion process, and of (top left) a first-order autoregressive process (top right), and magnifying the region between the two dotted lines twice.

There are two types of scaling that may bring out the property of scale invariance in a signal, namely self-similar scaling and self-affine scaling.

Description: *Self-similar* scaling [PeSa88, Worn96, Turc97, Kins04] is achieved by applying a single scaling factor (for magnification or reduction purposes) in all directions. On a one-dimensional signal, i.e., a time series, this implies that each of the time and amplitude axes uses the same scaling.

Description: *Self-affine* scaling [PeSa88, Worn96, Turc97, Kins04] is achieved by applying different scaling factors in all directions. On a 1-dimensional signal, this implies that each of the time and amplitude axes uses a different scaling.

For a signal that is a fractal, the original signal can be reproduced only after the application of one of the mentioned scaling type. The shape of the reproduced signal may be morphologically or statistically similar.

Description: The term *morphological* is used when the scaled version is an exact copy or a perfect replica of the whole of the original signal [PeSa88, Worn96, Turc97, Kins04].

Description: The term *statistical* is used when the scaled version is not an exact copy, yet its statistical properties, structure, and complexity are similar to that of the original waveform [PeSa88, Worn96, Turc97, Kins04].

Figure 4.3 attempts to sum up the mentioned time-domain signal properties through illustrating their relevance/usage in describing fractal time series.

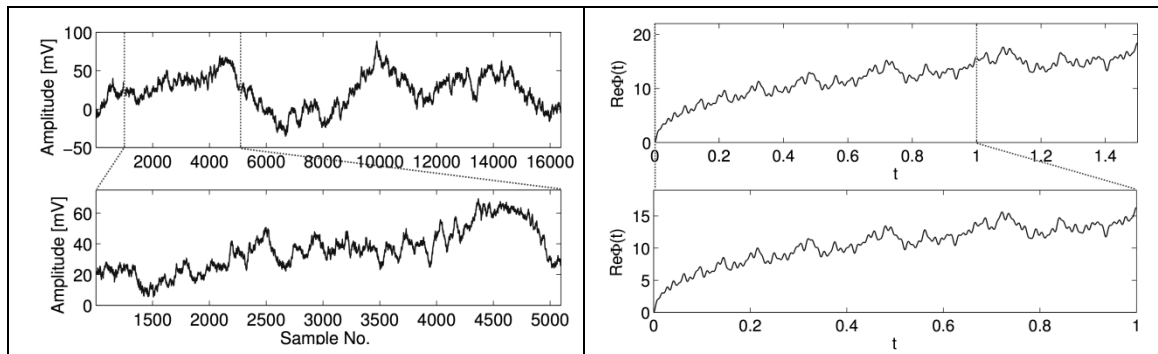


Fig. 4.3 The unscaled (top row) and scaled (bottom row) versions of a self-affine signal belonging to (left column) a statistically scale-invariant stochastic process, the Brownian motion, and (right column) a morphologically scale-invariant deterministic process, defined by the Weierstrass-Mandelbrot function [BeLe80]. After [WPKF07].

This thesis deals with monofractal noise models to synthesize self-affine signals from the class of statistically scale-invariant stochastic processes known as $1/f$ processes [PeSa88, Worn96, Turc97, Kins04], as the noise signals to induce radio frequency interference in my performance evaluation experiments.

4.2.3.2 Frequency-Domain Signal Properties or Features

Since fractal signals such as monofractal noises, which are $1/f$ processes, have been proposed (Section 4.2.2) for modelling broadband electromagnetic noise signals, the frequency-domain signal properties shall be discussed in relation to the chosen noise model.

The broadband nature of these noises makes them useful in modelling the noise of interest. At this point, it is useful to make a distinction on what is meant when the term broadband is used. The key to distinguish is from the point-of-view of the signal's cut-off frequency. Unlike traditional filter functions that have a band-limiting skirt defined by a 3-dB cut-off frequency (an example of a filter function is shown at the bottom panel of Fig. 4.4), the cut-off frequency for the spectrum of $1/f$ processes appears at the knee frequency, which is the frequency where the monofractal spectrum meets the noise floor or background noise. Since the frequency spectrum of a monofractal noise is depicted by an infinitely (theoretically-speaking) long tail distribution (top panel of Fig. 4.4), the cut-off occurs only after a broad or long range of frequencies, thus the logic for using the term "broadband". On a side note, real-world realizations of $1/f$ processes, as synthesized in software, will have a finite tail. Section 4.4 provides the synthesis details.

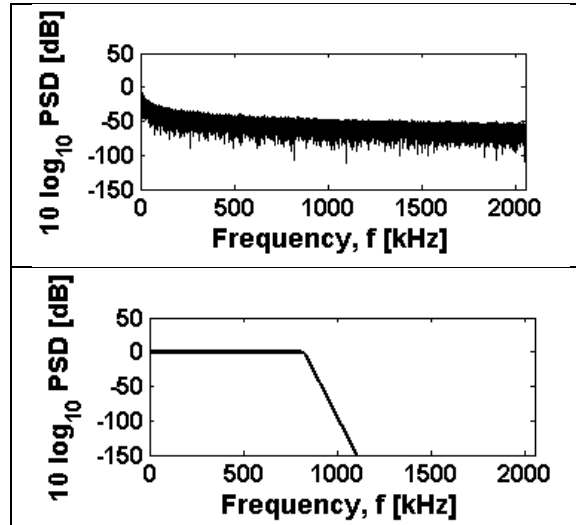


Fig. 4.4 Broadband frequencies versus band-limited frequencies, illustrated through (top) the spectrum of a power law process (brown noise), and (bottom) the characteristic of a filter's (low-pass) frequency response.

Power spectrum density, PSD, is the signal processing tool used to characterize fractal signals. PSD is computed by squaring the magnitude of the signal's Fourier transform. The PSD of $1/f$ processes has a power law dependence on frequency such that:

$$P(f) \propto \frac{1}{f^\beta} \quad (1)$$

where f denotes frequency, and β is the positive exponent of the indicated power law function, also known as the spectral exponent, and the symbol \propto represents proportionality [Turc97]. The spectral exponent can take either an integer or a fractional value and links the function to a shade of colour as the naming convention for the noise. When β is integer-valued, it represents a subset of monofractal noises, such as white noise ($1/f^0$), pink noise ($1/f^1$), brown noise ($1/f^2$) and black noise ($1/f^3$). Fractional-valued β represents the subset of monofractal noises called fractional Brownian motion, fBm, also known as fractional noises, with β being a real number. The spectral exponent, β is not limited to zero and positive value (i.e., white to brownish or black noise), but

negative value too (i.e., blue noise and violet noise) with respect to the zero-slope of the white noise.

The straight line or linear relationship is evident only through applying linear regression on the frequency content of the power law in a log-log plot. A graph with logarithmic scales on both its axes is a better representation for power law as such graph stretches the lower frequencies and shrinks the higher frequencies on each axis, compared to a linearly-scaled graph. A semi-log graph (top panel of Fig. 4.4) only shows the exponential nature of the power law.

As depicted by Fig. 4.5, the slope or gradient of the line on the log-log plot of a power law gives the β value of the corresponding fractal signal. A single beta will be found for a signal that has a complexity of a monofractal. The distinction between multifractal and monofractal is given in Section 4.2.3.3.

The steepness of the constant slope in the log-log plot of a monofractal noise power spectrum can reveal the degree of persistence of the noise time samples. Referring to Fig. 4.5, as the slope in the spectral domain gets steeper, the noise signal in the time-domain gets smoother, which means that the noise time samples become more correlated. In other words, the trend as β increases is that the degree of persistence in the time series increases too, from the totally-uncorrelated white noise time samples to the more correlated time samples of brown noise. Persistence implies the tendency of each spike to follow the trend or direction that the previous spike has taken on (up is followed by up, and down is followed by down) [Kins04]. To complete the picture, there exists also the notion of anti-persistence. For e.g., the time series when β is -0.5 or -1.0 are considered anti-persistent because the adjacent values in its time samples are less correlated than the uncorrelated white noise ($\beta = 0$) [Turc97]. The degree of persistence (or correlation) in the time series is related to the rate of the PSD decay. The smoother the signal observed in the time-domain, the steeper the decay observed in the spectral-domain.

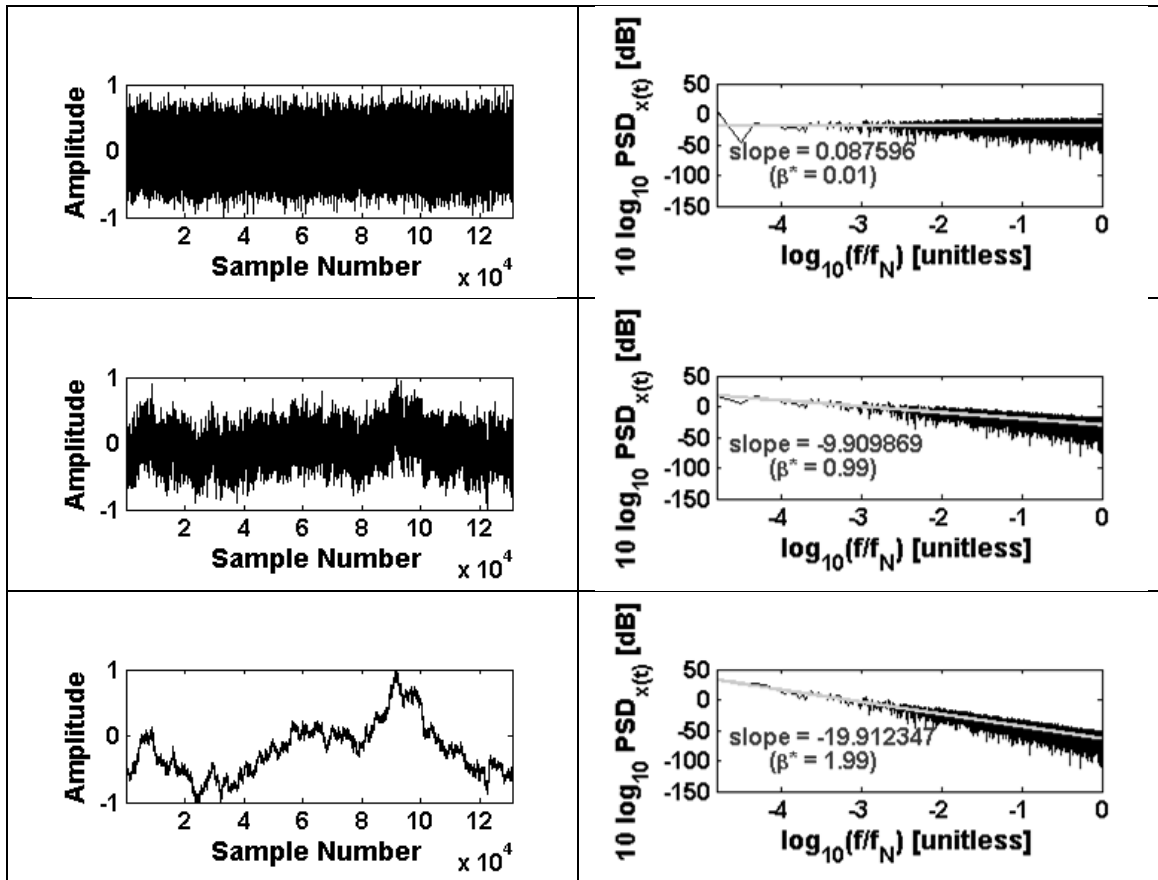


Fig. 4.5 The time-domain versus spectral-domain representations for monofractal (top) white noise, (middle) pink noise, and (bottom) brown noise, where their spectral exponent, β , is 0, 1, and 2, respectively.

One constraint to note is that power law exists only around the direct current (DC) component. As will be seen in Section 4.5, this fact is crucial as it justifies the observation from the empirical work on PSD of modulated monofractal noise, that a power law function becomes non-power law after being amplitude-modulated.

4.2.3.3 Measure of Signal Complexity

Signal in this sense is referring to the noise signal. So, this section is more aptly named as ‘Measure of Noise Complexity’.

Euclidean dimension (also known as topological dimension), D_E , describes the dimension of simple objects in terms of a non-negative integer value, such as a point ($D_E = 0$), line or curve ($D_E = 1$), area or surface ($D_E = 2$), volume ($D_E = 3$), and time-space ($D_E = 4$).

Fractal dimension, D is the non-negative fractional dimension that describes the complexity or irregularity of a fractal object to fill space. For example, an object with fractal dimension of 1.8 has a complexity such that it fills up a space amounting to more than a line but less than an area (as it approaches but not exactly a dimension of 2). In this case, dimension of 1 is termed as the embedding dimension (or embedded Euclidean space).

The measure of complexity for a fractal time series can be described in terms of H , the Hurst exponent [PeSa88, Kins04] or β , the spectral exponent because they are related to the fractal dimension, D , in the following manner:

$$D = 2 - H = \frac{5 - \beta}{2} \quad (2)$$

Furthermore, the range for each complexity measure parameter is restricted to: $1 < \beta < 3$, $0 < H < 1$, and $1 < D < 2$, respectively [PeSa88]. The Hurst exponent gives the degree of correlation. The power law exponent provides the same information as the Hurst exponent. This relationship is utilized in relating the variance fractal dimension and Hurst exponent to spectral fractal dimension and spectral exponent [Kins04] for results in section 4.3.2.2.

The H (hence β) plays a role in scaling a self-affine fractal time series. As mentioned in Section 4.2.3.1, for a scaled 1-D-embedded signal to be indistinguishable from the original signal, the appropriate type of scaling has to be applied. In particular, if x-axis is magnified by r , then $y = rx$ is similar to $y = x$ for a self-similar time series. However, if x-axis is magnified by r , then $y =$

$r^H x$ is similar to $y = x$ for a self-affine signal. The signals in Fig. 4.3 are self-affine since different magnification factor was applied to the two axes.

There are many different functions to compute fractal dimension. These functions can be generalized to fall under the category of morphological-based, entropy-based, and transform based [Kins05]. The transform-based function called variance fractal dimension will be used in Section 4.3 to provide insight to the properties of the time series of real noise capture. In terms of the type of fractality or complexity for an object (in our case, the object is a series in time), it may be monofractal or multifractal. An object whose complexity is consistent throughout the entire object results in a single fractal dimension (hence a single scaling or a single spectral exponent), and as such is thus called monofractal. In other words, the various fractal dimension functions when computed for such object, should all amount to the same value. Multifractal, on the other hand, consists of a “mixture” of monofractals (hence, multiple scaling relationships or consisting of multiple spectral exponents). This implies that such object has a different dimension when computed at different instance of the object. An ‘object’ can be 1-D, 2-D, or more dimensional. If the object is 1-D, the term “instance” denotes the point, spot, segment, or window in time where the fractal dimension is computed for. The instance where a dimension is computed matters for a multifractal object, because the object will have varying dimensions over time. For a monofractal object on the other hand, the instance also matters because one can tell that the object is monofractal when the same dimension value is obtained at different instance of time, i.e., constant dimension over time (the notion for single fractal or mono-fractal as opposed to multifractal).

4.3 Validation of Noise Model (in the Field)

This section brings forth the findings of the analyses that extract characteristics of real vehicular noise data for the purpose of validating the noise model (see Section 4.2) that I chose to

represent broadband EM noise in industrial operating environment. A comparison between real noise character and the character represented by the fractal noise model is required in order to validate the chosen noise model. However, the extraction of real noise signature requires the input of real noise time series. To the best of my knowledge, there seems to be no published data on industrial vehicular noise available for use. As such, I took the initiative to collect noise data from the field. The measurement/collection of noise data from the source of the starter system of industrial vehicles took place in the Winnipeg facilities of Vansco and Buhler Versatile on February 1, 2007. The noise measurements were done by myself and assisted by Dr. Ferens. Figure 4.6 shows a snapshot taken on the day I conducted my noise measurements. For more information on the details of the vehicular noise analysis, please refer to my published paper [WoKF08].



Fig. 4.6 A snapshot of the noise acquisition setup.

4.3.1 Methodology for Characterizing Real Noise

To not take away the thesis focus, this section only touches on the important ideas surrounding the three steps involved (i.e., noise identification, acquisition, and analysis) in achieving the goal of obtaining real noise character. Again, for implementation-specific details, please see the published work [WoKF08].

4.3.1.1 Real Noise Source Identification / Identification of Real Noise Sources in Vehicle

The references that I used to help me in identifying the vehicular electrical and electronics components that give out broadband EM noise are [IEC02], [ISO98], and [DALP97]. Some examples of the sources in vehicles that may cause broadband disturbance are motors (such as starter, wiper, and blower), lightings, and alternators.

The noise data used for the analyses [WoKF08] were recorded from the starter system as the noise source. This is the only source that successfully provides noise readings within the frequency range of interest. Attempts to collect noise data from the lighting and alternator components in a tractor did not result in any noticeable EM noise other than the background noise.

It is crucial that only one noise source is on at the time of the measurement. Noises that are internal to the operations of the tractor can be eliminated by switching only one source to be on at a time. For instance, Mr. John Vukelic (Buhler Versatile representative) had kindly hard-wired a mechanism to one of the company's tractor so that only the starter motor is triggered to run at the press of a button. As for noise external to the tractor, I have selected a spectrum analyzer for use in my noise signal acquisition that has the ability to mask out certain noises that appear within the frequency range of interest.

4.3.1.2 Real Noise Data Acquisition / Acquisition of Real Noise Data

A real-time spectrum analyzer, RTSA, was rented to acquire recordings of time samples that capture the transient or bursty nature of vehicular noise for the 15 MHz span of frequencies centered at 2.405 GHz. As I am interested in determining if broadband EM noise exists to possibly interfere with ZigBee operation, the spectrum analyzer and the receiver antenna were set for a center frequency and a resonant frequency at 2.405 GHz (i.e., the first channel in ZigBee 2.4 GHz operation), respectively.

Table 4.1 Materials and equipment for real noise data acquisition.

Equipment / Materials	Justification
Real-time spectrum analyzer (model: Tektronix RSA 3303A with Option 02 [Tekt08a])	<ul style="list-style-type: none"> • real-time, i.e., time and frequency information; • met requirement: ranges from DC up to 3 GHz; • trigger in time (Power Trigger) and in frequency (15 MHz bandwidth Frequency Mask Trigger) domains
Receiver antenna (custom-built microstrip patch antenna, from [BeKZ06])	<ul style="list-style-type: none"> • to receive signals at its set frequencies; • no cost, re-use antenna from PER experiments.
Table/cart	RTSA requires a stable platform.
Power bar	To attach to nearby electrical outlet to power RTSA.
Anti-electrostatic wrist strap	As the name implies.
Keyboard and mouse	These are required to manoeuvre around the different folders and directories in the RTSA operating system (RTSA is running on Windows Vista) and to type in filenames for the captured noise traces.
Coaxial (RG-58) cable	To be attached between RTSA and the receiver antenna so that the antenna reaches the capture point, which is at a particular distance away from the noise source.

Table 4.1 provides a complete list of equipment and materials that I used and would recommend for this stage, plus justification for their use. The recordings of noise signals emitted by the starter system of a tractor are available on the attached CD-ROM (Appendix B.1).

The following outlines my industrial noise capture procedure:

1. Obtain baseline noise when tractor is not running. Baseline is to capture the ambient noise to see what noise activity (if any) is present and also to know the noise floor.
2. Apply frequency mask (Fig. 4.7) to mask out noise floor, lesser strength signals, and unwanted signals.
3. Hold antenna at a distance away from the noise source. Turn noise source on. The RTSA will capture emitted noise to save on trigger according to the shape of frequency mask set.
4. Repeat step 3, this time by holding the antenna at a different distance. The distances used were 0.05, 0.3, 1, 2, and 5 metres.

5. Repeat steps 3 and 4 for a different noise source. Noise sources attempted were lighting, alternator, and starter motor.
6. Repeat steps 1-5 for a different machine. Two brands of tractors were used, Ford and Buhler Versatile.

EM noise does exist in the 2.4 GHz range as is evident from Fig. 4.7. With the frequency mask trigger feature (included only when option 02 is installed on the spectrum analyzer), regions of unwanted signals (such as noise floor and other existing transmitters' spectra) can be masked off, so that the only signal captured is the noise emitted from the noise source of interest.

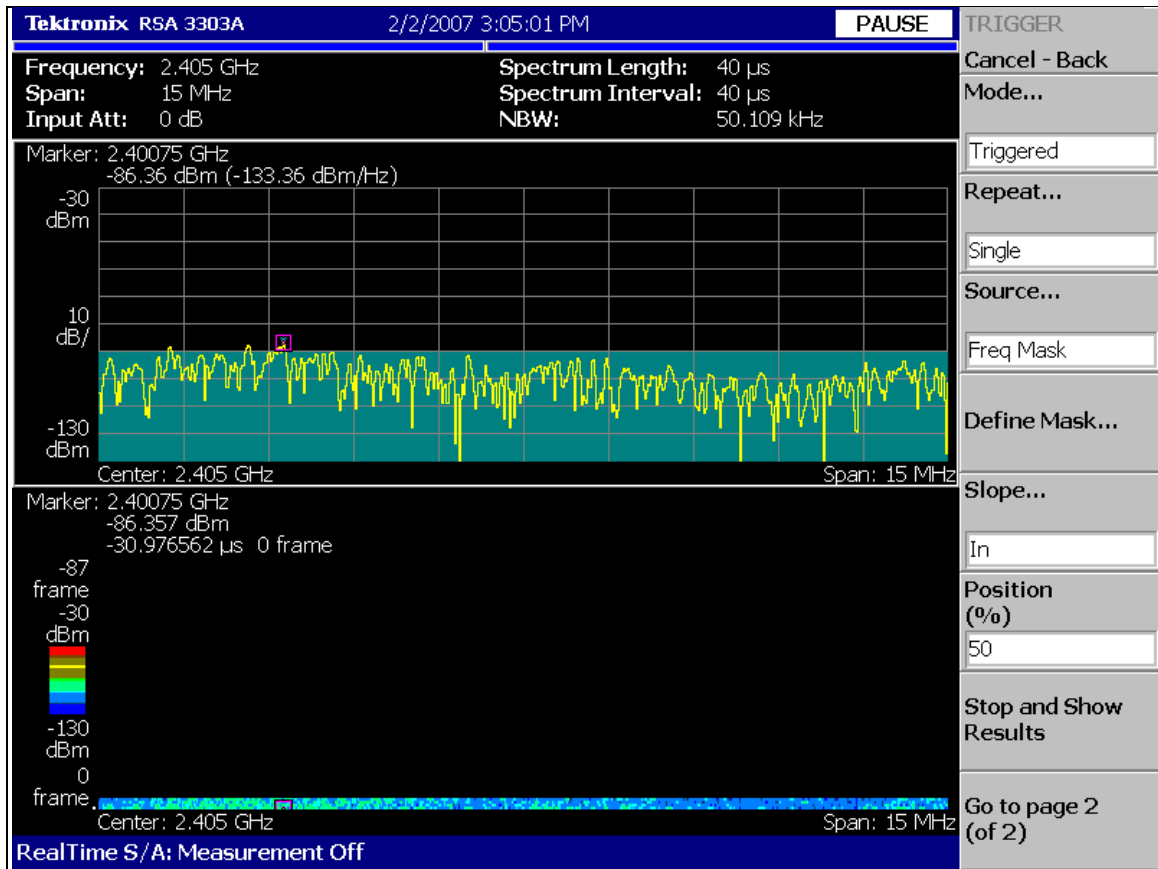


Fig. 4.7 A screenshot of one of the noise recordings viewed offline using RSAVu. The blue area is where the frequency mask is applied such that only captured signal with frequencies above -90 dBm will trigger to be stored/saved.

4.3.1.3 Real Noise Data Analysis / Analysis of Real Noise Data

The pre-processing stage is an artefact due to the format of the stored noise trace. The RTSA stores the captured noise signal in the .iqt (in-phase and quadrature versus time) format. So, this stage involves obtaining time series (power versus time) information from the raw noise data. The program to do this is provided by Tektronix [Simm06] of which I have rewritten (see `plottimeseries_v12.m` in Appendix A.6) to work for my captured noise data. The implementation of this stage has been verified by comparing the obtained time series obtained to that displayed on RSAVu, an offline analysis program [Tekt08b].

To analyze the captured noise time series, two analyses were attempted: first is a frequency-domain analysis. First, the spectral analysis of the signal under test was performed and information were displayed in a log-log plot to see how the signal's spectral density varies with frequency, and a spectrogram to see how the signal's spectral density varies with time (see `noisecaptureSFD_v11.m` in Appendix A.5); second is a time-domain analysis by using a multi-scale analysis tool known as variance fractal dimension trajectory, VFDT, where a constant trajectory would mean the signal under test is monofractal in time, otherwise, the signal is multifractal in time.

I proceed to using the second approach because PSD does not exist for a time-varying spectral density, which is the case for the starter noise burst portion. Furthermore, the multi-scale approach used by the variance fractal dimension algorithm allows for much richer spectrum of information of the signal's complexity to be revealed, compared to single-scale approach (such is the case for spectral analysis).

In general, VFDT is computed by passing windowed segments (with the option of overlapping or non-overlapping windows) consecutively to be processed by the VFD algorithm. VFD uses a multi-scale approach to extract the complexity of the portion of signal that it

received, where a fractal dimension is computed based on the slope of the log-log plot of multiple variances corresponding to different time scales. Thus, each point in the trajectory is essentially the fractal dimension for the respective windowed segment. The flowchart showing the operations involved for the VFDT and the VFD is shown in Fig. 4.8. The MATLAB® code that I wrote can be found in Appendix A.7. The VFD code section has been verified by using simple test signals (such as a line, a sine wave, and a triangle wave) and complex test signals (by using the different monofractal noise signals). Correct implementation is indicated by the VFD returning a dimension of 1 for test waveforms in the first case, and a dimension between 1 and 2 for monofractal with spectral exponent of $1 < \beta < 3$ for the test waveforms in the latter case. The two parameters that control the amount of details seen in the trajectory are the window size, and the window overlapping amount used in the VFDT. In other words, the shape of the computed trajectory will change when a different choice of value is used for each of the parameters. For each of my captured noise time series, I find through experimenting that its features or signature is adequately extracted by using the following design choice: 30 consecutive windows, each spaced apart by a 50% overlapping. On a different note, the trajectory (output of VFD) is dependent on the quality of the input signal. Thus, factors such as the sampling frequency, capture span, type and orientation of receiver antenna, and the degree of isolation of noise sources, all may affect the recorded wave shape (hence the shape of the computed trajectory).

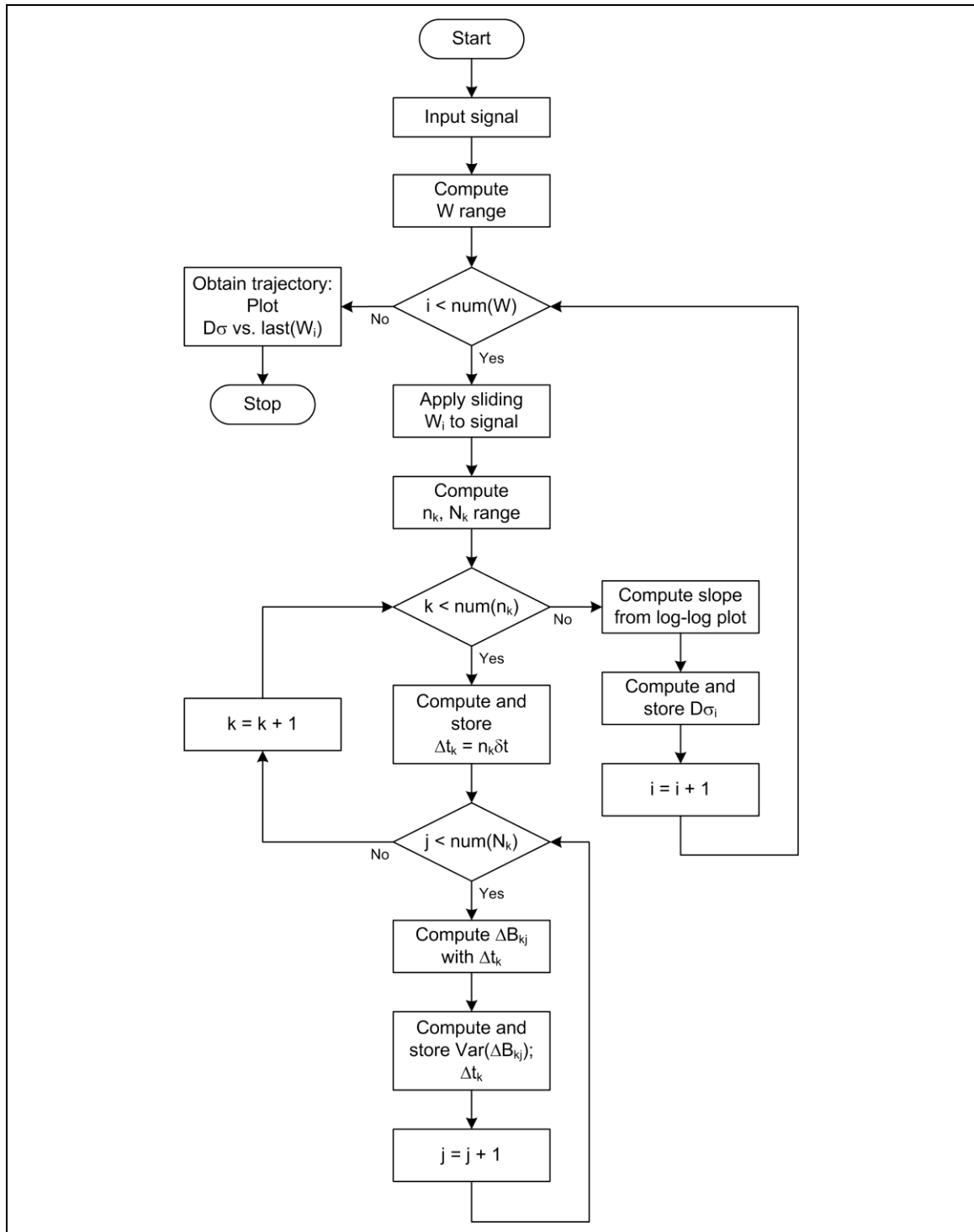


Fig. 4.8 Variance fractal dimension trajectory (After [GrKi94]).

Figure 4.9 shows the multi-scale tool in action where VFDT is applied to a time series made up by stringing white noise, pink noise, brownish noise, and white noise with a gain, one after the

other. Due to the monofractality of the sequence, the trajectory returned showed constant dimension for each monofractal noise. The abruptness due to the end of one monofractal noise being attached to the start of the next monofractal noise explains the delay seen before the trajectory settle to the expected dimension for each monofractal noise type.

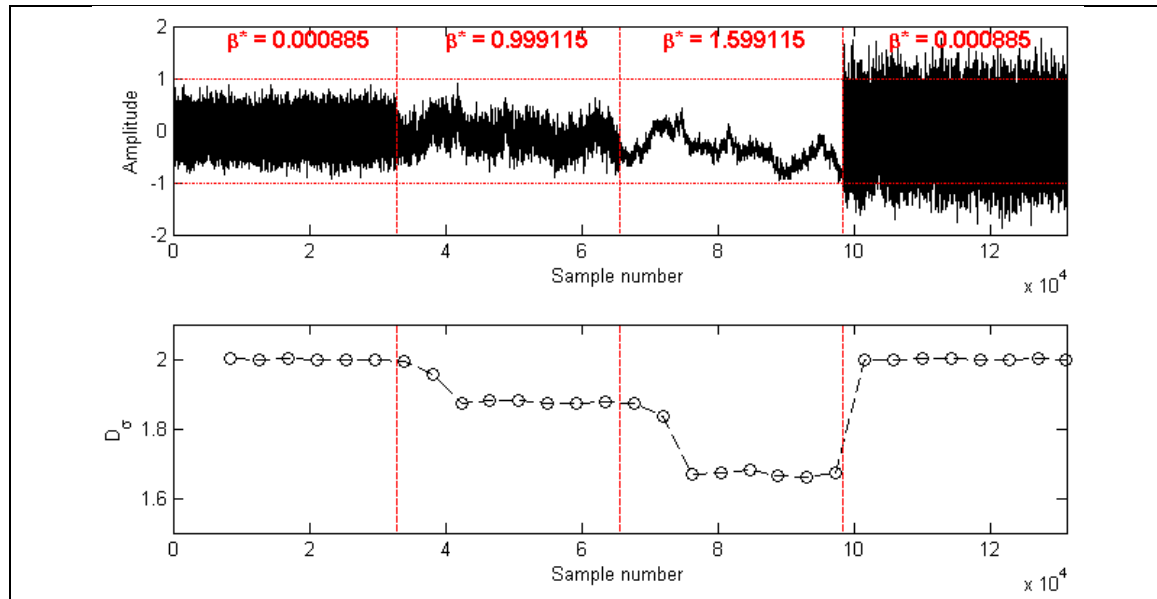


Fig. 4.9 The time series of a sequence of different monofractal noises combined (top) and the result of passing the sequence to the VFDT (bottom).

4.3.2 Characteristics/Complexity of Real Noise

The resulting plots from the single-scale and the multi-scale analyses are presented and discussed in this section to draw conclusions about the complexity of real noise.

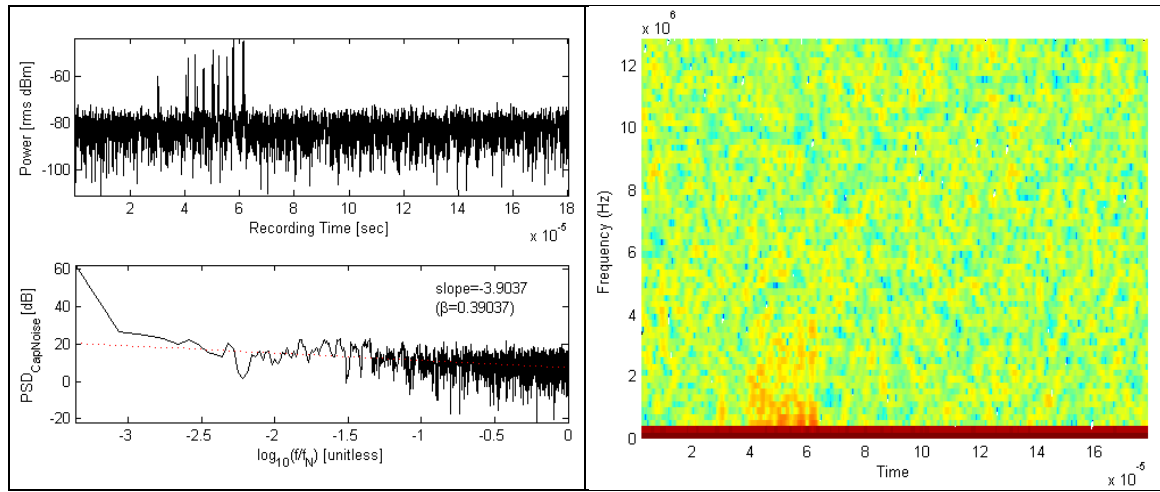


Fig. 4.10 The PSD (left, bottom) and spectrogram (right) of a captured noise time series (left, top).

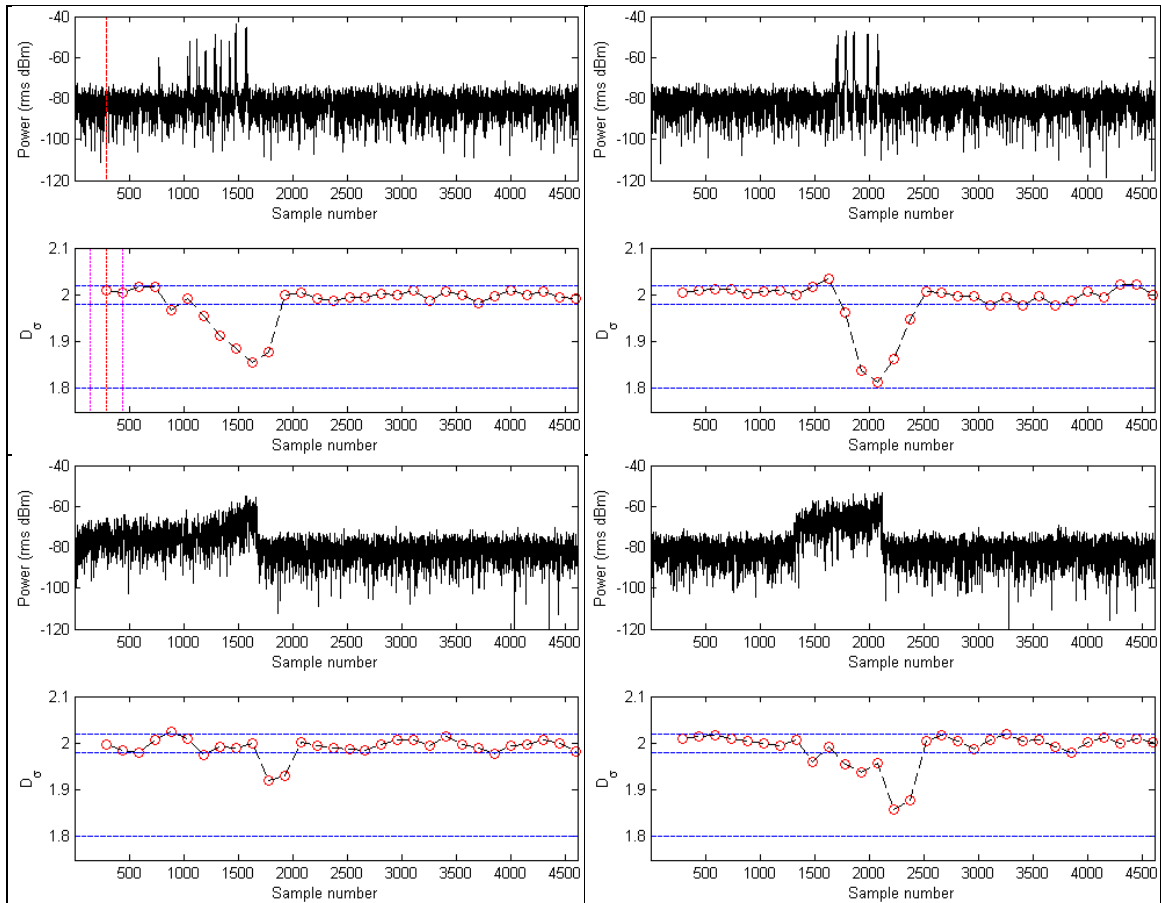


Fig. 4.11 Time series representing noise signals emanated by the starter system of a Buhler tractor measured at 3 metres (top, left), at 2 metres (top, right), and of a Ford tractor measured at 1 metre (bottom, left), and at 0.05 metres (bottom, right) away from the noise source.

4.3.2.1 Complexity of Real Noise Gathered from Single-Scale Analysis

Figure 4.10 (left, bottom) shows that the value of the spectral exponent derived from the slope of the log-log plot of PSD(captured signal) is approximately zero. This may be due to the limited capture span of the spectrum analyzer (15 MHz), such that any captured signal would then inadvertently appear to be white. However, a further check (the spectrogram on Fig. 4.10 (right)) verifies that the captured signal as a whole is non-stationary. Due to the non-identical frequencies being observed over the length of time examined, the act of taking the average (since the PSD operation is averaging the magnitude of frequency) only serves to destroy critical information since there is no way of distinguishing which frequencies belong to which portion of the captured

signal (the noise burst portion or the background noise portion). In a nutshell, all the findings point to PSD (a single-scale analysis) being a wrong tool to analyze the captured noise traces, with PSD being meant for analyzing stationary signal.

4.3.2.2 Complexity of Real Noise Gathered from Multi-Scale Analysis

Each recording (shown in top panel of each cell on Fig. 4.11) was 200 μ sec (4608 samples in total, after having deducted 512 samples at the pre-processing stage) in length and sampled at 25.6 Msamples/s. The horizontal axis on each recording's time series and VFDT plot use the same label (i.e., both be stated either in sample number or in time) for ease of matching the returned fractal dimensions to the recorded wave shape. Note that each point on the trajectory denotes the returned dimension for each windowed segment, plotted at the end point of the window.

The computed trajectory for each noise recording in Figure 4.11 is not constant. First interpretation would be that the entire time series denotes an object that is multifractal over time. However, because there are two noticeable signals present in the recording, the trajectory is best analyzed by splitting it into two distinct parts, namely the background noise portion (consistent noisiness with power concentration/strength/intensity between -90 and -75 rms dBm) and the starter noise portion (abrupt, transient-like spikes that overpowered the background noise).

The background noise portion has an almost constant trajectory (valued at $D_{\sigma} \approx 2$), which indicates that its fractality is temporal monofractality (or monofractal in time). The slight variation (a departure from the expected no variation) in the returned dimensions can be attributed to the nature of a real background noise signal as opposed to a synthesized white noise as in Fig. 4.9. The starter noise portion has a varying (dip/trough shaped) trajectory ($1.5 < D_{\sigma} < 2.0$), which can be restated as $0 < H < 0.5$. This range of Hurst exponent indicates that the signal has a negative correlation or anti-persistence behaviour [Turc97]. The implication is that the trend of its

amplitude increments is such that an increase is most likely followed by a decrease, and vice-versa [Kins04]. Furthermore, the complexity of the noise is a mixture of monofractals over time, consisting of components with nature between white/pink noise ($H = 0$) and brown noise ($H = 0.5$).

For future work, the few limitations/shortcomings associated with my work on real noise characterization can be improved by considering more noise sources, and by obtaining noise recordings at higher sampling frequency.

4.3.3 Contributions of Noise Validation

This section relates the noise characterization findings to their effects/significance upon my ZigBee-noise performance evaluation experiments.

First, the real noise capture analysis showed that the starter noise burst portion is multifractal in time supports (thus validates) my choice of a fractal-based noise model to represent real transient-like (broadband) EM noise. Second, the finding that the starter noise burst portion has a complexity consisting of a mixture of anti-persistent monofractal components can be utilized to influence the choice of monofractals used as the input noise in my experiments so as to better reflect noise in reality. More specifically, by restating the returned range of dimensions for the starter noise burst portion in terms of a range of spectral exponents, I can fine tune the noise model to use say, monofractals with exponents 0.4, 0.8, 1.2 and 1.6, respectively, as fractals that best fit the property of the captured noise, to study its effects on ZigBee transmission.

4.4 Fractal Noise Synthesis (in Software)

This section provides a description of the synthesis and verification procedure of the algorithm used to generate discrete sequence of the subset of monofractal noise called fractional noise, the few precautions to note in the MATLAB implementation of the algorithm to prepare the discrete sequence for conversion to analog signal, and a highlight of the distinct character of the synthesized sequence.

The conjecture that broadband electromagnetic industrial noise is better modelled by fractal because it has the following properties: long term dependencies and power law, leads to monofractal noises being used as a starting point in achieving a model for the noise of interest. A process with long-term correlation will have a power spectrum density that follows a power law distribution. Furthermore, the spectral exponent (slope of the PSD) of a power law process can take on any value. As such, I have to be concerned with fractional noise, rather than approximating a fractional-valued power law to an integer noise. Both integer and fractional noises are objects with single fractal dimension, thus the generalized term “monofractal noise” is being used. Monofractal noise can model the different types of error in received bits. For instance white noise models random error, while coloured (fractional) noise models burst errors with its varying degree of correlation.

4.4.1 Algorithms for Generating Time-Series of Fractal Noise

There are several methods to synthesize synthetic fractional noise sequence. Among the more common fractional Brownian motion generation methods are:

- i. Spectral Filtering Algorithm, SFA [PeSa88, GMHG94, Turc97, GCTH07]
- ii. Simplified SFA (also known as Saupe’s method) [PeSa88, Kins04]
- iii. Random mid-point displacement [PeSa88, GCTH07]
- iv. Successive random addition method [PeSa88, GMHG94, Turc97, GCTH07]

- v. Weierstrass-Mandelbrot function-based method [PeSa88, GMHG94, GCTH07]
- vi. Wavelet-based method [GCTH07]

For more details about the above-mentioned approaches, the references indicated in square brackets are suggested to interested readers.

4.4.2 Spectral Filtering Algorithm

The fractional noise signals used in this thesis have all been synthesized using the spectral filtering algorithm, *SFA*, because

- i. The algorithm is simple and straightforward to implement;
- ii. The algorithm can produce the desired sequence in one run;
- iii. The algorithm's output sequence is an accurate estimation of the desired fractional noise. According to Voss (pg. 50 of [PeSa88]), this advantage is brought by the application of fast Fourier transform, which covers a full range of frequency components for the input time interval.
- iv. The algorithm operates in the spectral domain (i.e., loglog-domain) in estimating the fractal measures (β and H). This domain is where these measures have linear relationship, which otherwise they have non-linear relationship in the linear domain.
- v. The algorithm can synthesize any fractional noises [Kins04]

The input parameters to the SFA are the respective value for

- i. the length of the desired fBm, N ,
- ii. the seed value to initialize the starting point of the pseudorandom number generator in step (i) of the SFA, $seedval$, and
- iii. the value of the desired fBm's spectral exponent, β . My interest is monofractal noises with spectral exponent in the range between 0 and 3.

The SFA (pg. 49-50 of [PeSa88], pg. 149-151 of [Turc97], pg. 63-65 of [Kins04]) consists of four steps:

- i. Generate a zero mean and unit variance white Gaussian noise discrete sequence of N data points, $W(n)$;
- ii. Obtain the Fourier coefficients of $W(n)$, $\hat{W}(m)$, by taking the discrete Fourier transform of the white noise sequence from step (i);
- iii. Obtain the Fourier coefficients of the desired fBm, $\hat{B}(m)$, by multiplying the Fourier coefficients from step (ii) with a transfer function of the form $f^{-\left(\frac{\beta}{2}\right)}$ where $f = \frac{m}{N-1}$ for $0 \leq m \leq N-1$. The exponent of the transfer function being $-\frac{\beta}{2}$ is simply because the PSD of the desired fBm that gives $f^{-\beta}$ is achieved by taking the squared magnitude of the Fourier transform of the fractional noise sequence;
- iv. Obtain an N -point sequence of the desired fBm, $B(n)$ by taking the inverse discrete Fourier transform of the filtered coefficients from step (iii).

Figure 4.12 shows a flow diagram of the SFA. My implementation of SFA in MATLAB® can be found in Appendix A.1. The four noise sequences used for my PER experiments can be found in Appendix B.2 on the attached CD_ROM.

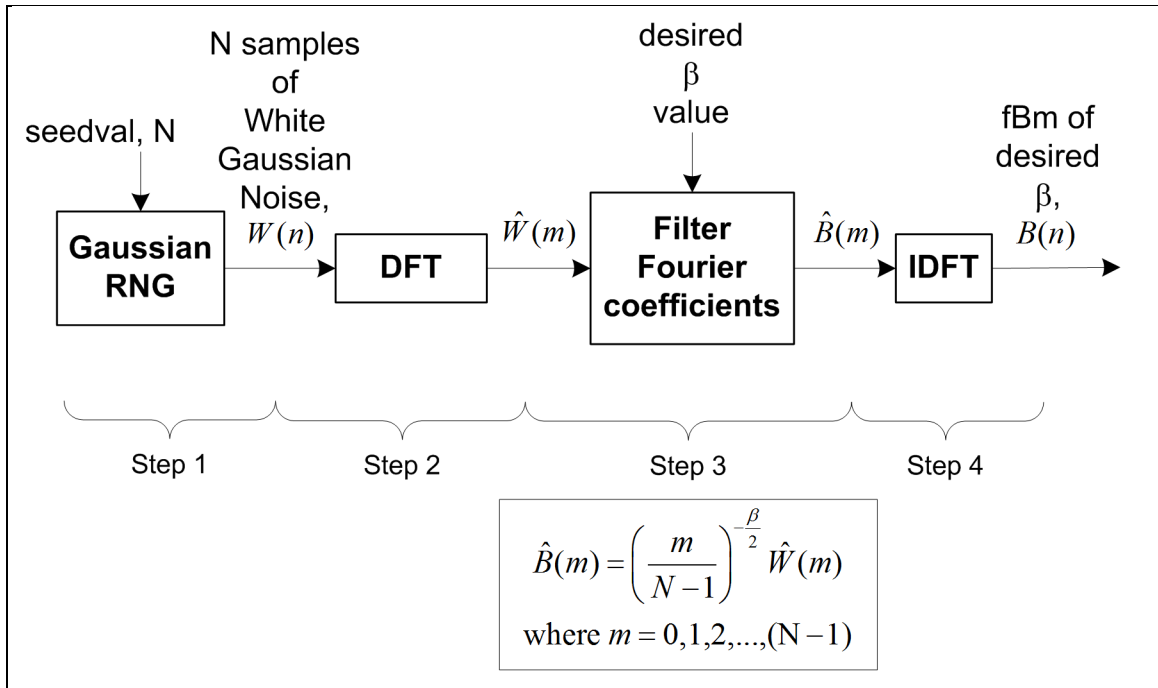


Fig. 4.12 Flowchart depicting the SFA for generating an fBm of the desired spectral exponent.

4.4.3 Care Taken when Coding

This section highlights the restricting elements to be cautious and to take note of when proceeding to generate discrete fractal noise sequence through SFA in MATLAB® for feeding into a function generator (specifically, Agilent 33120A arbitrary waveform generator [Agil02]) to produce the analog version of the synthetic noise. The restricting elements are the constraints imposed by the software (i.e., MATLAB®) and the hardware (i.e., AWG) used.

4.4.3.1 Limitation/Restrictions Imposed by Hardware

The MATLAB computation of fractal noise sequence for use in my PER experiments was implemented such that the preference specified by the machine (AWG) that accepts the discrete noise data points is followed closely (see Appendix A.1 for all the participating codes that make up the MATLAB program for fractal noise synthesis), namely:

- The waveform to be uploaded should consist of the maximum amount of points acceptable by the function generator, i.e., 16,000 samples;

- ii. The waveform to be uploaded should be made up of floating-point values between -1 and +1 (to represent the 12-bit discrete voltage levels for the digital-to-analog convertor), where the values -1 and +1 correspond to the peak values at the negative and at the positive extremities corresponding to the peak-to-peak Voltage, V_{pp} , value set for the AMPLITUDE parameter. The actual value set for the AMPLITUDE parameter is influenced by the performance evaluation experimental procedure. The procedure requires that the AMPLITUDE value be tuned from the knob on the function generator to one that will induce a certain percentage of PER (a desired value between 0% and 100% PER, inclusive) when one instance of ZigBee 1000-packet transmission is executed. So, for X number of desired PER values, there should be X different amplitude values found; and
- iii. The FREQUENCY parameter is set to be the maximum output (playback) frequency in order to prevent dropping or skipping of data points per each cycle of the playback waveform. The derivation can be found in Appendix C.10 (with reference to pg. 275-276, Chapter 7 Tutorial in [Agil02]).

The machine-specific rules stated in the first two bulleted items can be found in (pg. 174/178 Chapter 4 Remote Interface Reference in [Agil02]). To conform to the rule in (ii), I have coded a normalization routine in MATLAB (normalize.m code in Appendix A.1), which is to be used when generating the fractal noise sequence required for noise uploading onto AWG.

4.4.3.2 Limitation/Restrictions Imposed by Software

A note on the type and length of the random number generator, RNG, used for generating the N sequence of Gaussian distributed random numbers (step 1 of SFA algorithm, calls the function GaussRV2.m, Appendix A.1). I used the MATLAB® RANDN function, specifically, randn('seed', seedval), where 'seed' calls the polar algorithm (a default in MATLAB version 4)

as the generator algorithm used by the random number stream with a period approximate to $(2^{31}-1)*(\pi/8)$, seedval is the value that the state of the generator is initialized to and takes a scalar integer value between 0 and $2^{31}-2$, inclusive, and the letter 'n' in the function name denotes returning pseudorandom values that follows a standard normal distribution. Compared to the period of the generated stream, the actual length used (16,000 points $\approx 2^{14}$) in each sequence of the baseband noise signals in my research did not exceed the number of samples in one cycle of the stream, which ensures that the pseudo-randomness of the generated sequence is sustained since the sequence does not repeat itself. The final step required to compute a white Gaussian noise (WGN) sequence of the desired mean and variance from the generated m-by-n matrix of normally-distributed random numbers rand(m,n) is by setting $WGN = \text{mean} + \text{sqrt}(\text{variance}) * \text{randn}(m,n)$.

A note on the value of the seed used in rand(). First, to reproduce the same sequence of monofractal noises that are used in this thesis, the same seed choice as the author should be used when running the MATLAB implementation of SFA. Second, care was taken such that the chosen seed value is one that gives the least over/under -estimation of beta, determined from seeing the result for a seed value ranging between 1 and 50. The pool being the first 50 integer values is arbitrarily selected.

For step 3 of the SFA algorithm, to ensure proper and accurate filtering of the Fourier coefficients of the white noise sequence from step 2 of SFA algorithm, the trick is to first create a filter function solely for the first half (this value is calculated using E_pt.m, Appendix A.1) of the sequence of points corresponding to the expected length of the noise spectrum's Nyquist frequency. Second, the filter function corresponding to the sequence of points for the remaining half of the noise spectrum length is created by flipping the sequence produced in the previous step. Third, the two sequences are appended consecutively. Such manoeuvre is required to be in

line with the symmetry (a feature common in the output sequence returned by the application of MATLAB® FFT function) found in the Fourier coefficients of the white noise sequence. Fourth, to avoid the existence of a singularity caused by a division by zero in the filter function when $m = 0$, the first value in the newly-created filter function is reset to zero before the filter function can be multiplied onto the Fourier coefficients of the white noise sequence. Because of this, I have aptly named this region in the computed PSD of the generated noise sequence as the “undefined” region (see Fig. 4.17) so that the region is never considered or is omitted in any signal analysis. Furthermore, when plotting the log-log plot, the first point of the PSD sequence should be omitted, since $\log 0$ is undefined. Steps 2 to 4 of SFA is implemented in function `spec_filter.m`, Appendix A.1.

4.4.4 Verification of Synthesized Noise Data (Implementation Accuracy)

I have created a MATLAB code (`verifyCN.m` in Appendix A.1) to verify that my implementation of SFA is correct. The verification uses the idea presented in section 4.2.3.2, i.e., the gist is to apply logarithm to the PSD of the output sequence and estimate its spectral exponent from the slope of the log-log plot. The step-by-step flow diagram for the verification procedure can be seen in Fig. 4.13. On a specific note, since $10 \log_{10} PSD$ versus $\log_{10} f$ is being plotted (Fig. 4.5 of section 4.2.3.2), the value of the estimated spectral exponent of the output (the synthesized sequence), β^* , can be found using $-slope/10$.

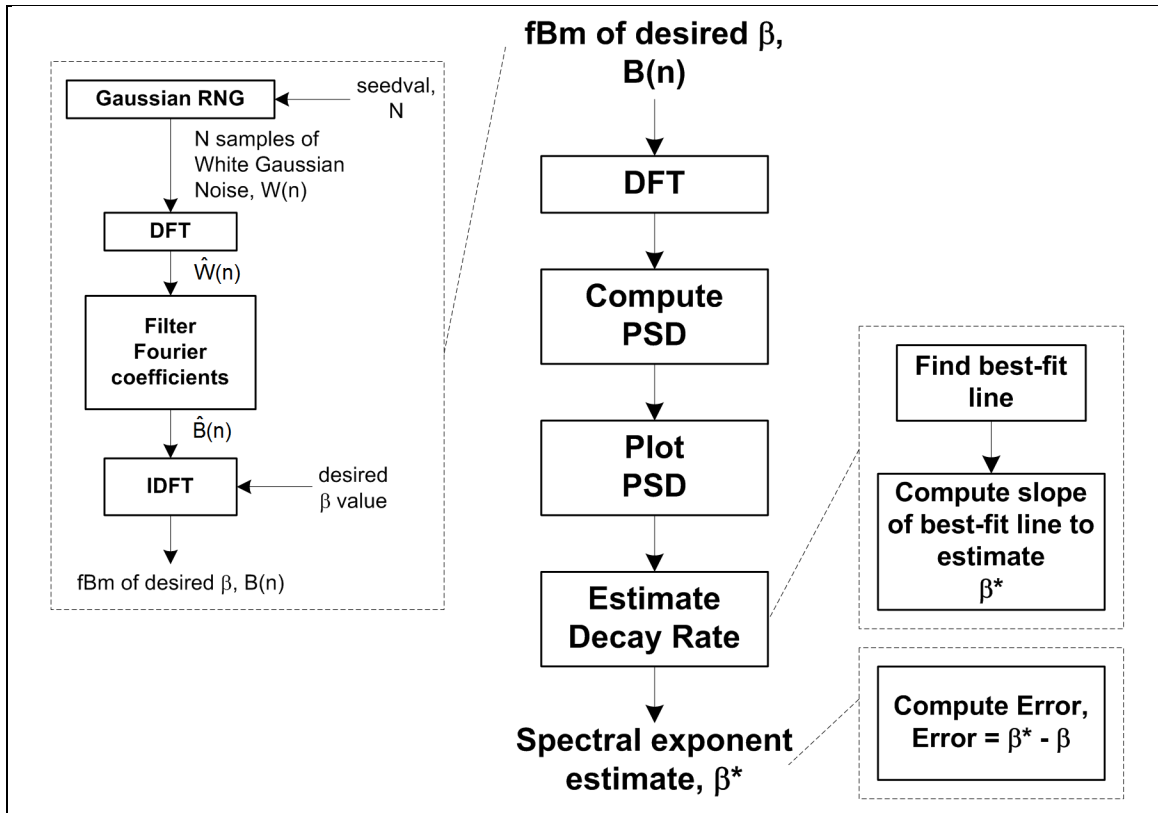


Fig. 4.13 Flowchart for verifying the output of the MATLAB implementation of SFA.

The accuracy of the algorithm can be judged from the amount of the computed error value. The error value is calculated by taking the difference between the estimated spectral exponent of the output (the synthesized sequence), β^* , and the expected spectral exponent of the desired sequence, β_{exp} , i.e., by computing $\beta^* - \beta_{exp}$. When the error is greater than the expected value, the estimated spectral exponent for the monofractal under test is said to be over-estimated, and likewise, when the error is lesser than the expected value, the estimated spectral exponent for the monofractal under test is said to be under-estimated. Figure 4.14 shows the computed error values corresponding to the different monofractals generated from my MATLAB implementation with expected spectral exponent of 0, 1, 1.2, 1.6, 2, 2.4, 2.8, respectively, for a specific seed value. The magnitude of the error values in the SFA-generated sequences is considered acceptable since they range from 0.004 to 0.012. Furthermore, if need be, an upper-bound error can be computed using

$\text{abs}(\max(\beta^*) - \beta_{exp}) / \beta_{exp}$. I have used such chart (Fig. 4.14) to my advantage where I have repeated the procedure to compute the error values for the same desired monofractals but generated from a different seed value. After 50 repetitions, I pick the seed that offers the least magnitude of error across the tested monofractals. In other words, this seed will generate monofractal sequences with the least error compared to the other 40 seeds. Thus, this shows the error in the synthesized sequence can be minimized by the right value of seed supplied to the algorithm. It is my design choice to use a seed that gives the least over- and under-estimation for the synthesized monofractal signals.

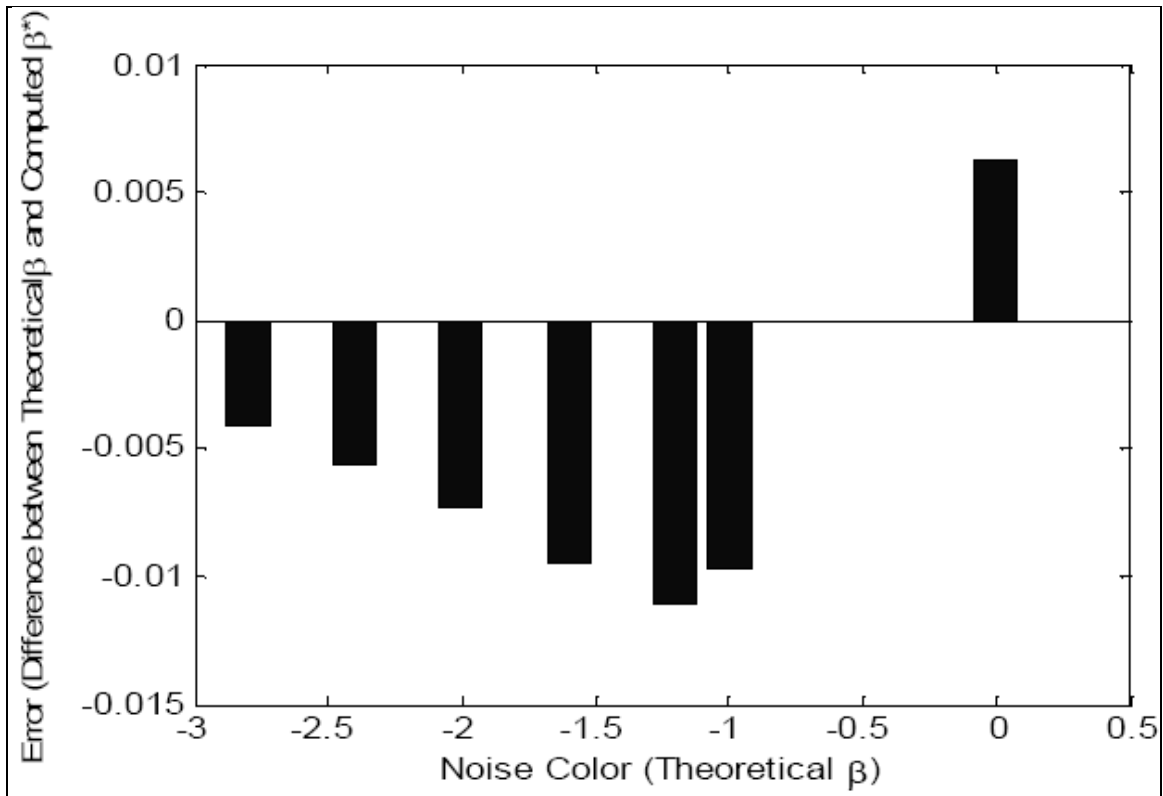


Fig. 4.14 Error in the computed spectral exponent corresponding to each noise colour.

4.4.5 Characteristics of Synthesized Noise Sequence

There are three distinct characteristics that I can gather from the log-log plot generated for the SFA-synthesized noise sequence.

First is that their PSD plots have an asymmetry characteristic. The asymmetry characteristic on those plots was due to the type of plot itself – a log-log plot. Since the human brain thinks in a linear fashion, the asymmetry appearance is really only what the human eyes perceived. The log-log plot (right column of Fig. 4.5) has scales with logarithmic distances, which explains why the curve at lower frequencies is spread or stretched, while the curve at higher frequencies is compressed. Since the curve is drawn on a log-log plot, this characteristic is true on both axes.

Second is that their PSD plots have a straight line characteristic. When the PSD is plotted with linear scales on both axis, an exponentially decreasing curve (i.e. the curve follows power law, $y = bx^a$) is obtained. By taking log on both sides of the power law equation, we get $\log y = \log b + a \log x$ (this is our log-log plot which follows the form of a straight line equation, $y = b + ax$).

Third is the common characteristic of the first point in each PSD plot. To explain the characteristic of the first point in the PSD plot, the term singularity is used. The first point has an infinite value (due to a division by zero in computing the Fourier coefficient for this point), which makes it impossible to plot such point on the graph. As such, in the MATLAB program, a value was assigned to the point in order to not ignore the existence of the first point.

4.5 Fractal Noise Implementation: A Hardware Emulation System

The system consists of two analog signal generation modules: the baseband stage and the high-frequency band stage. This section will describe the implementation of the system to be within the confines of the goals of the emulation system.

4.5.1 Emulation Goals

The noise emulation hardware system should be implemented with the following goals/purposes in mind, i.e., to:

- 1) enable control (thus adding flexibility), through the tuning of
 - i. signal power/amplitude;
 - ii. signal type;
 - iii. signal frequency.
- 2) enable worst-case corruption, through applying maximum concentration of noise
 - i. in-wire within the receiver module;
 - ii. on the protocol's operating channel's center frequency, e.g., ZigBee $f_{\text{channel}\#12} = 2.41$ GHz;
 - iii. along the 3-dB bandwidth, e.g. for ZigBee is 2 MHz.
- 3) enable accurate noise power measurement through in-wire addition of noise instead of over-the-air because in doing so, this would
 - i. eliminate the issue of signal emanation;
 - ii. eliminate the issue of signal reflection (if any, it will be minor);
 - iii. allow the use of a tool (spectrum analyzer) to provide direct measurement;
 - iv. allow the measurement and deduction of cable attenuation, if present, from the measured noise power.

4.5.2 Analog Noise Signals Involved: Baseband Fractal Signal Generation Module

The need for this stage in my noise emulation hardware system is due to the current state of technology in function generator hardware, which does not allow production of high-frequency fractal noise in one step. This is the first step.

4.5.2.1 Module Design

The following are the crucial elements of the baseband fractal signal generation stage of the noise emulation hardware system:

- 1) Module: A function generator. In particular, I used an arbitrary waveform generator, *AWG*.
- 2) Inputs: The discrete noise sequence (from subsection 4.4), and the amplitude and frequency information were uploaded onto the function generator so that it can generate a user-defined waveform.
- 3) Output: Analog baseband fractal signal, with some periodicity, an artefact due to the playback frequency of the *AWG*.
- 4) Justification for module: To allow for analog signal generation of user-defined input signal.
- 5) Purpose of signal creation: To serve as input to the next stage in the fractal noise implementation system.

4.5.2.2 Verification of Module Output

The verification procedure requires the output of the module to be fed into an oscilloscope. The oscilloscope output is then visually compared against the time-domain time sequence display of the discrete sequence of fractal noise in MATLAB®. The module is correctly implemented when both displays match one another.

4.5.3 Analog Noise Signals Involved: High-Frequency Fractal Signal Generation Module

The need for this stage in my noise emulation hardware system is due to the current state of technology in function generator hardware, which limits the direct production of high-frequency fractal noise one step. This is the additional step required.

4.5.3.1 Module Design

The following are the crucial elements of the high-frequency fractal signal generation stage of the noise emulation hardware system:

- 1) Module: Multiplier.
- 2) Inputs: 1) The analog signal output from the module in sub-subsection 4.5.2, and 2) the carrier signal.

- 3) Output: Analog high-frequency signal. The word “fractal” is deliberately not attached to the naming of the output. This is because additional manoeuvre (see 4.5.3.3 to 4.5.3.6) is required to force the signal to be an analog high-frequency fractal signal.
- 4) Justification for module: To perform amplitude modulation, *AM*, using the double sideband suppressed carrier, *DSB-SC*, scheme in order to shift frequencies of the baseband noise signal to the desired frequency.
- 5) Purpose of signal creation: To be the noise source that will cause broadband EM interference during the time period of ZigBee packets transmission in my performance evaluation experiments.

4.5.3.2 Comparison to Related Work

The fractal modulation scheme by Wornell (Chapter 6, [Worn96]) is quoted here because to my best knowledge, this may be the first treatment of fractal signal with respect to / in relation with modulation in the fields of communications.

The following are the crucial elements of the fractal modulation scheme for a transmitter:

- 1) Module: Multiplier
- 2) Inputs: 1) message or data, 2) an expansion in terms of wavelet-based orthonormal self-similar basis of degree H ;
- 3) Output: a bihomogeneous waveform of degree H , a signal with message embedded within itself at all time scales, is the signal to be transmitted to a fractal modulation receiver.
- 4) Justification for module: the scale-invariant property of the output signal allows recovery of information from a small segment of the received signal [pg. 4, Worn96]
- 5) Purpose of signal creation: To enable a secure and reliable communication system. More specifically, so that the transmitted signal in such communication system would contain the information carried by the data at all time scales.

When comparing my work to Wornell's, the similarity is in the field of application, i.e., both work are for use in communication system. Meanwhile the dissimilarity is in the purpose of the fractal modulation application, and hence its technicality, i.e., its inputs and output: Wornell transmits data embedded in fractal for secure communication (see Fig. 4.16); I transmit fractal as high-frequency noise signal for inducing broadband radio frequency interference (see Fig. 4.15). Even though the treatment of fractal modulation by Wornell is of a different nature than mine, it has been presented here to show a different take/approach on the matter and to support that what I did was indeed necessary.

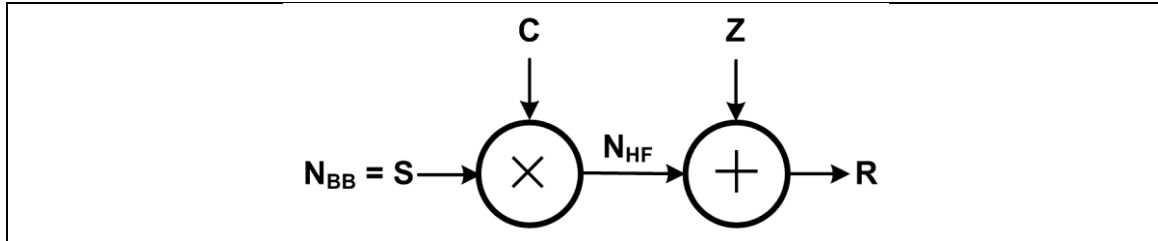


Fig. 4.15 Synthetic noise being added to the ZigBee receiver system: the modulating signal is N_{BB} (which represents the baseband noise signal that is scale-invariant, denoted by S), the carrier signal is C , the modulated signal is N_{HF} (which denotes high-frequency noise signal), the ZigBee transmitted signal is Z , and the interference-resultant signal going into the receiver system is R .

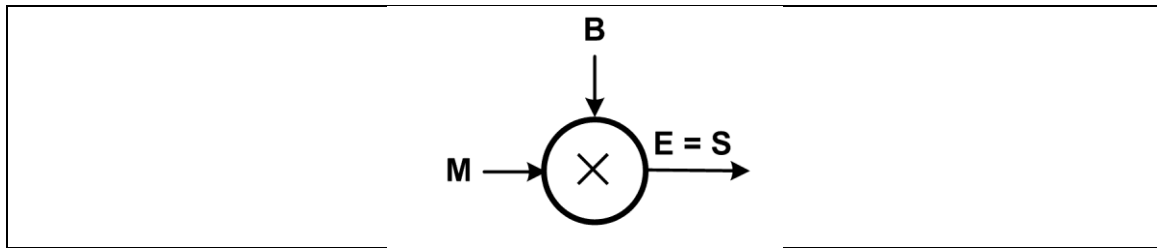


Fig. 4.16 Fractal modulation by Wornell, where the modulating signal is M (which denotes the message signal), and the carrier signal is B (which denotes a basic function signal), and the modulated signal is E (which represents the signal that has information embedded within itself on all temporal scales, hence a scale-invariant signal).

4.5.3.3 Verification of Module Output

Since amplitude modulation is the core element in producing the desired analog noise signal, it is best to determine if there are any side-effects that come from its employment. By viewing the AM process from the angle/perspective of its

- i. operator, a multiplier, then AM is non-linear. A multiplication operation is non-linear [pg. 121, Boga83; pg. 219, Carr99; pg. 74, HuSi01]
- ii. inputs and output, then AM is linear. This is because the same frequency spectrum (unchanging in terms of its relative amplitudes and phases) is produced at the output, as a result of a linear shift of frequencies of the original spectrum. [pg. 220, Tirr93; pg. 116, Gaba97; pg. 169, PrSa05]

It is the non-linearity of the amplitude modulation process that raises the question of whether the characteristics of the input signal, a baseband signal with a power law frequency distribution, has changed after amplitude modulation.

Will the output signal have the same characteristics as that of the modulating signal or will it be of a totally different nature? To answer such uncertainty, I have performed two types of analyses. Both analyses involve examining the upper-sideband of the PSD of the modulated signal in the positive frequencies. The reason is because this region has a decreasing exponential spectrum distribution, thus causing it to be the region where a power law distribution (i.e.,

indication of monofractal) is most likely to be found. To reveal its spectral characteristics, the first analysis applies binomial expansion to the function of interest, whereas the second analysis attempts to fit the function of interest to a single power law. The descriptions of the derivation steps involved in the mentioned analyses together with their results and interpretations have been presented in [WPKF07]. The MATLAB codes used for the respective analyses can be found in Appendix A.8.

4.5.3.4 Significant Findings of Verification: On Modulated Noise Character

Through performing the mentioned analysis, the modulated noise signal is found to be:

- i. multifractal over the upper-sideband of its positive frequency spectrum, and
- ii. monofractal within a small frequency interval in the upper-sideband of its positive frequency spectrum. The region of monofractality can be determined through the application of an experimentally-determined ratio. Furthermore, the spectral exponent of the power law that describes the monofractal region differs by a significant increase from that of the modulating signal. In other words, the modulated signal is much more correlated than the modulating signal.

Because the objective is to produce high-frequency fractal noise signal, some tweaking (see section 4.5.3.5) is thus required to ensure the generated signal at the output matches the property of the signal at the input as close as possible. Had the verification not been done, (i) I would have added the wrong noise signal, and (ii) the performance study would be inaccurate as the results would be attributed to the wrong noise type than intended/mentioned. Last but not least, the increased value of spectral exponent of the modulated signal point towards the realization that realistic or practical high-frequency fractal noise is not best achieved through this method.

4.5.3.5 Module Design Refinement

The module refinement takes the form of stating the exact positioning of (i) the carrier signal frequency and (ii) the region of the noise of interest for superpositioning onto the selected ZigBee channel.

The following lists the various constraints that affect the above decision.

- i. Constraint due to the internal filter (a 10 MHz, 7-th order Bessel filter (pg. 276, Chapter 7 Tutorial in [Agil02]) of the hardware used for the module in section 4.5.2.1:
 - The upper-sideband of the modulated signal's positive frequency spectrum is only 10 MHz-wide. In Fig. 4.17, this value is denoted by AWG_{cutoff} .
- ii. Constraint due to the analytical work from sections 4.5.3.3 to 4.5.3.4:
 - The fractal property of the modulated noise signal is not monofractal. Since the goal of my performance evaluation experiments is to gauge the effects of monofractal noise upon ZigBee packet transmission, the region within the modulated signal where monofractality can be observed is the region that matter. The frequency setting for the carrier signal and the region for interfering with ZigBee must then follow the experimentally-determined ratio of 4:1. The 4:1 denotes the ratio of the carrier offset from the center frequency of the selected ZigBee channel, f_z , to the 6-dB bandwidth of the ZigBee channel.
- iii. Considering (i) and (ii) above, and constraint due to the carrier signal component from the module in section 4.5.3.1:
 - Because the effective frequency range that the frequency synthesizer [Sire07b] covers is between 2.4 GHz and 2.5 GHz, the first possible channel for evaluating

fractal noise interference upon ZigBee system is the second channel in the 2.4GHz PHY, ZigBee channel 12, i.e., 2410 MHz.

With restrictions set by the abovementioned constraints and referring to Fig. 4.17, the analog high-frequency fractal noise signal that I synthetically generate for my performance evaluation experiments takes the following values: the carrier frequency, $f_c = 2402$ MHz, and the region of noise of interest is centered at the selected ZigBee channel center frequency, i.e., $f_z = 2410$ MHz. The superposition of synthetic noise upon ZigBee happens within the range of frequencies from 2409 to 2411 MHz (indicated by the shaded area in Fig. 4.17). The undefined region is noted here to caution readers against using this range of frequencies (whether in the baseband or high-frequency band) in the generated signal.

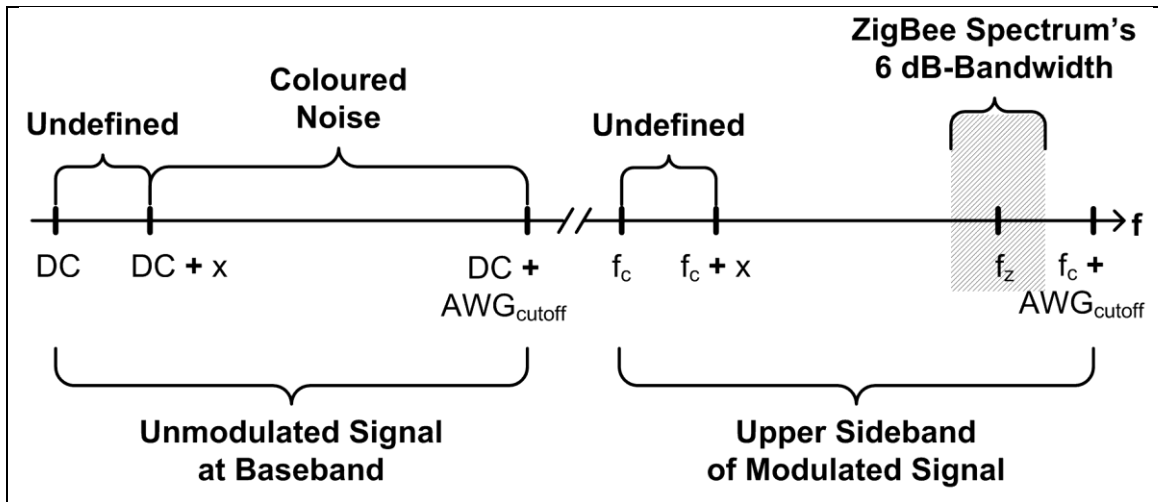


Fig. 4.17 The frequencies that matter for the signals generated in 4.5.2.1 and in 4.5.3.1, respectively (After [WKFD07]).

4.5.4 Hardware Involved

Specific hardware components used for the implementation of high-frequency fractal noise is covered in Section 5.

4.6 Chapter Summary

By characterizing real noise (through knowledge presented in sub-sections 4.1, 4.2, and 4.3) and simulating noise (through knowledge presented in sub-sections 4.4 and 4.5), the complexity of real noise can be understood and synthetic noise that resembles closer to real noise can be generated. The benefit of the using fractal-based noise model lies on its ability to easily model the varying fractal complexity aspect of broadband EM noise, something that was not possible with the existing common noise models. As a consequence, synthetic noises that closely follow the characteristic of real vehicular communication channels can be generated.

Chapter 5

DESIGN OF PERFORMANCE EVALUATION

SYSTEM AND EXPERIMENTS

The focus of this research work is to analyze the performance of the selected wireless technology, ZigBee, under noisy conditions from the operating environment of industrial applications. Since an empirical approach is decided upon to achieve this goal, a system (an experimental setup) and experiments for measuring the performance of the ZigBee technology under the selected noise signals is required and has thus been designed and developed.

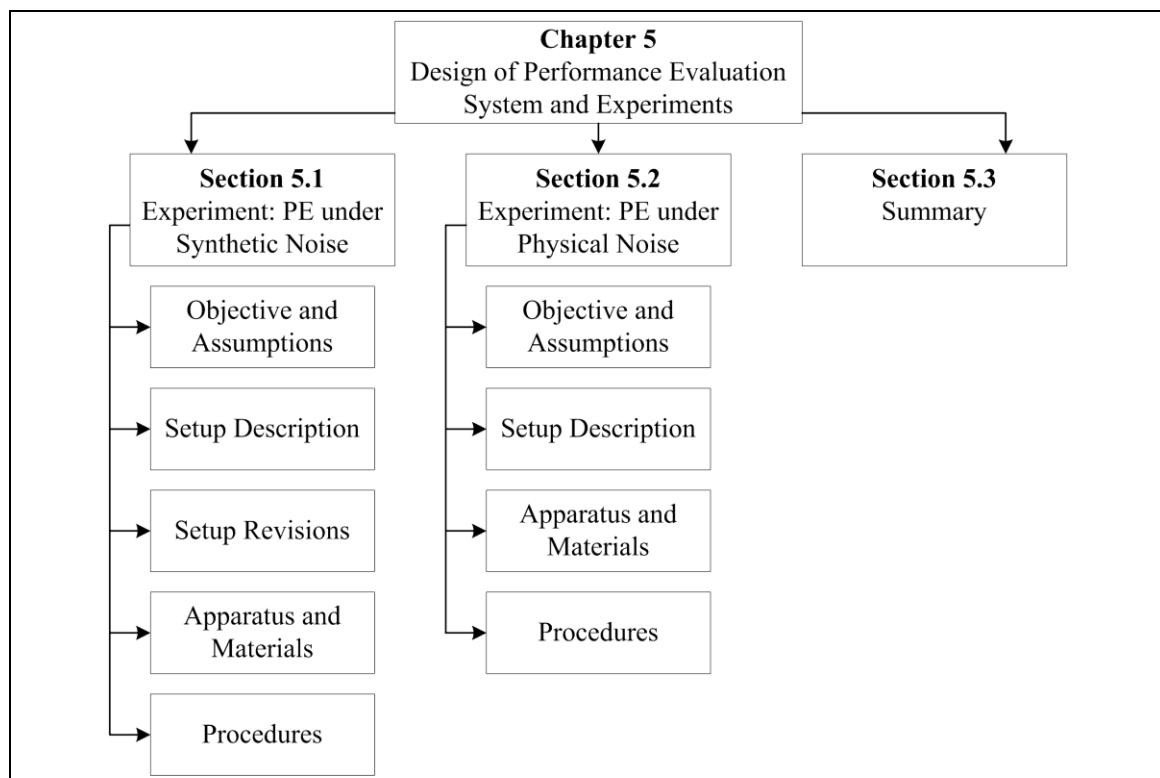


Fig. 5.1 Layout of Chapter 5.

5.1 Experiment: Performance Evaluation under Synthetic Noise

The performance evaluation setup consists of hardware and software components for a 2-node network that enable (i) the transmission and reception of ZigBee packets for PER measurements, (ii) the emulation of high-frequency noise signal as source of interference, and (iii) the capturing of ZigBee and noise traces to calculate their respective powers for SNR computations. The experiments involve measuring PER and SNR for ZigBee transmission under the respective input noises at different nodes' separation. This section covers the specification and implementation of such a system for gathering performance data.

The general design of the PER performance evaluation setup was suggested by Dr. Ferens. The hardware components for this original setup were sourced together with Dr. Ferens and the undergraduate design group [BeKZ06]. The original setup design underwent two major modifications. The first modification was needed after I realized that the PER measurements collected using the original setup components were inconsistent and irreproducible. I traced the issue to be due to the drifting of the carrier frequency. A second modification was warranted when I realized that the performance curves (PER versus SNR) for different nodes' separations (distances) were overlapping onto each other, suggesting that the different nodes separations have the same effect on system performance. The two modifications mentioned are the major ones among the other modifications that I had to do. The first modification is critical as it impacts the noise power measurements. The second modification is critical as it impacts the performance evaluation results. The solutions to the two issues that called for major modifications as well as other improvements made to the performance evaluation system will be presented within this section (see subsection 5.1.3).

5.1.1 Objective, Methodology, Outcomes, Benefits, Assumptions

The objective of this experiment is to study (measure and evaluate/analyze) the performance of the selected protocol, ZigBee, under noisy conditions. More specifically:

- A. To determine the effect of different broadband noise types on ZigBee receiver's packet loss performance, repeated over several nodes separations.
- B. To determine the minimum number of nodes in a ZigBee network under the different broadband noise types.

An elaboration of the methodology used is presented below:

- where noisy conditions are simulated by a range of monofractal noises that emulates a broadband physical noise (e.g. emitted by starter source), fed into the performance evaluation system one noise type at a time. Each noise type being injected at various power levels to affect 0% to 100% PER.
- where performance of ZigBee is in terms of ZigBee receiver's packet loss performance versus SNR at the receiver.
- where different SNRs are simulated through (i) P_{SRX} is the received ZigBee signal from the transmission of fixed-sized ZigBee packet from a transmitter with fixed output power level to a receiver positioned at a fixed, specific distance away, and (ii) P_N is the noise signal injected between the external receiving antenna and the receiver's circuitry, where a different power level of the specific emulated noise type is combined with P_{SRX} for the entire duration of each experiment (i.e., one experiment amounting to 10 separate runs, where each run consists of a 1000-packet transmission) OR for the specific emulated noise type, a different power level of the noise (the P_N) is being added to the received ZigBee signal between the external receiving antenna and the receiver's circuitry per 10 runs of 1000 packets transmission.

- where the effect of distance is simulated by repeating PER vs. SNR at a different nodes separation.
- where 1 experiment = 10 runs (1000-packet transmission per run)
- where number of experiments = [(with noise = 4 noise types \times 7 noise levels) + (without noise = 2 trials)] \times 3 distances = 90 experiments, where noise type = monofractal noise with spectral exponent 0.4, 0.8, 1.2, and 1.6, respectively; noise level = PER 0%, 10%, 20%, 40%, 60%, 80%, and 100%, respectively; distance = 1 m, 2 m, and 4m, respectively.
- number of tasks = 90 experiments \times (2 tasks) = 180 tasks, where task 1 = obtain PER and task 2 = obtain trace (offline)

The outcomes of the experiment are performance data in terms of:

- (a) PER versus SNR at the receiver for each noise type and at various distances;
- (b) maximum distance between two ZigBee nodes for reliable communications (say X% is the tolerable PER) versus SNR at the transmitter for each noise type.

The following shows the benefits that can be gained from the output:

- (i) Performance of ZigBee under the influence of each noise type. In other words, using (a), with SNR at a particular site known, one can find out the worst case error rates to expect under a specific noise type when nodes are separated at the specified distances apart.
- (ii) Number of ZigBee nodes to be present within a given space. In other words, using (b), with SNR at the transmitter end for a particular site known, one can extract the maximum distance for nodes separation that will achieve the desired level of reliability under the specific noise type to assist in nodes placement that will lead to a minimum number of nodes in a given volume of space at the site.

The following lists the assumptions and constraints for the experimental setup design:

- A. Communication channel is free from error.

- B. Noise is added one noise type at a time, i.e. not a mixture of noises.
- C. Propagation of packets or signal transmission from the transmitter to the receiver is line of sight, i.e., no obstruction.
- D. Worst-case performance.
- E. No retransmission allowed.

To satisfy A, I can use a Faraday cage as the communication channel. Such a cage blocks radio waves from coming into the cage. However, its shortcoming is multipath can happen within the cage, causing the channel to not be free from error. To prevent multipath, I can use an anechoic chamber as the communication channel, whose function is to absorb any reflection of waves within the chamber. However, its downside is that the chamber is not shielded from ambient radio waves, again causing the channel to not be free from error. By considering the mentioned factors, I have thus decided to use a special chamber that is capable of performing the two functions, which are (1) acts as a Faraday cage to prevent external waves from coming into the chamber, and at the same time (2) acts as an anechoic chamber to absorb internally reflected waves. I have used such a chamber for the PER experiments. The special chamber was located in the Ground Penetrating Radar laboratory at the University of Manitoba and was used with permission from Dr. LoVetri. This particular chamber is semi-lined with radiation absorbent material (RAM). The term semi-lined is in the sense that not all internal surfaces of the chamber are covered with RAM. In this case, the uncovered surface is the floor. With the limited amount of loose RAM available for use, I have placed them on the floor (see Fig. 5.15) along the transmission path to minimize reflection from ground.

To satisfy C, both the transmitter and the receiver nodes have been elevated to the same height (0.5 metres) and no obstructions were placed in between them. Furthermore, I have placed the two ZigBee nodes such that they are aligned along the center line of the rectangular chamber.

5.1.2 System Description

The ZigBee-noise performance evaluation setup can be categorized into three distinct parts, namely the communication network circuit, the noise emulation circuit, and the trace acquisition circuit. Figure 5.2 shows the hardware and software components for the first two circuits, whereas the hardware components pertaining to the third circuit can be seen in Fig. 5.3. Apart from that, a special chamber that is capable of absorbing any internal radio waves (an anechoic chamber) and shielding against any ambient radio waves (a Faraday cage) is used to provide control over the transmission channel. Thus, the communication network circuit is placed inside the chamber, whereas the noise emulation circuit is placed out of the chamber. The trace acquisition circuit is located outside the chamber, but it does require access to some of the components inside the chamber. The setup shown in Fig. 5.2 is the third and finalized (incorporating the second revision) version. The original version [WKFD06] of the setup can be found in Appendix C.6 (Fig. 1), and the second version (incorporating the first revision) can be found in Appendix C.6 (Fig. 2) or [WKFD07].

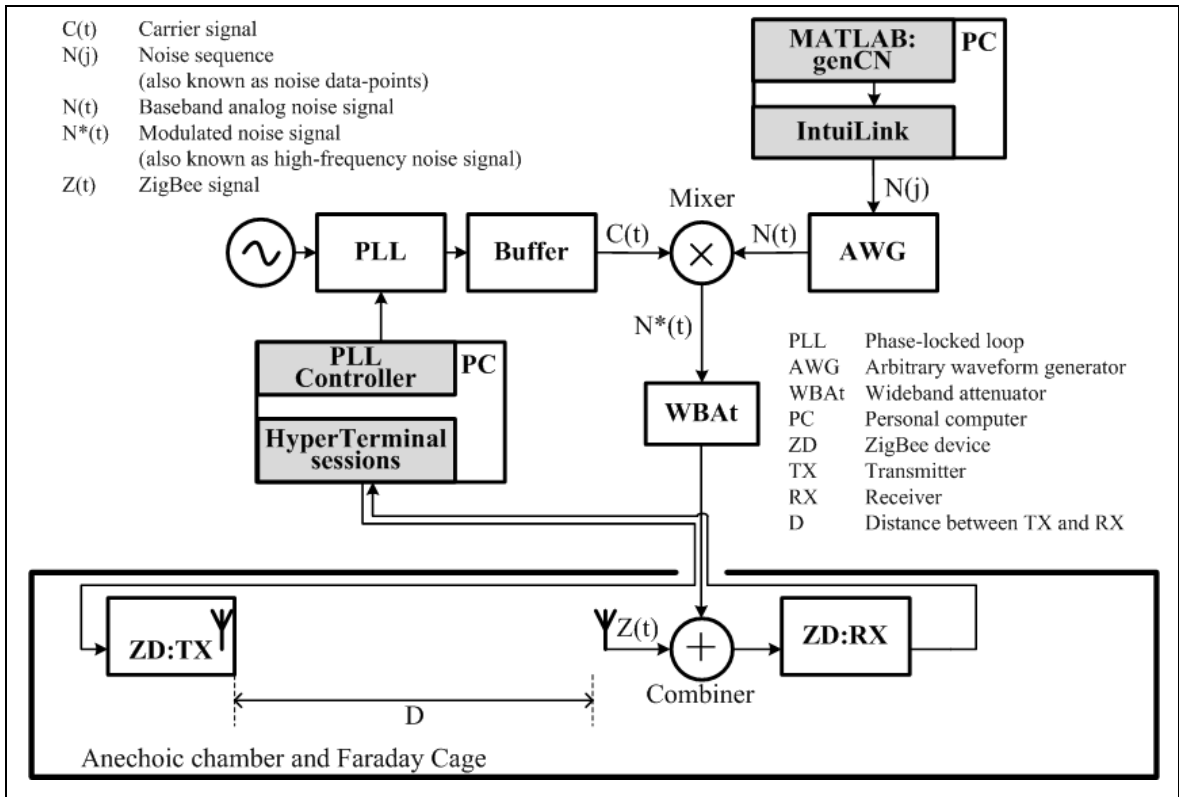
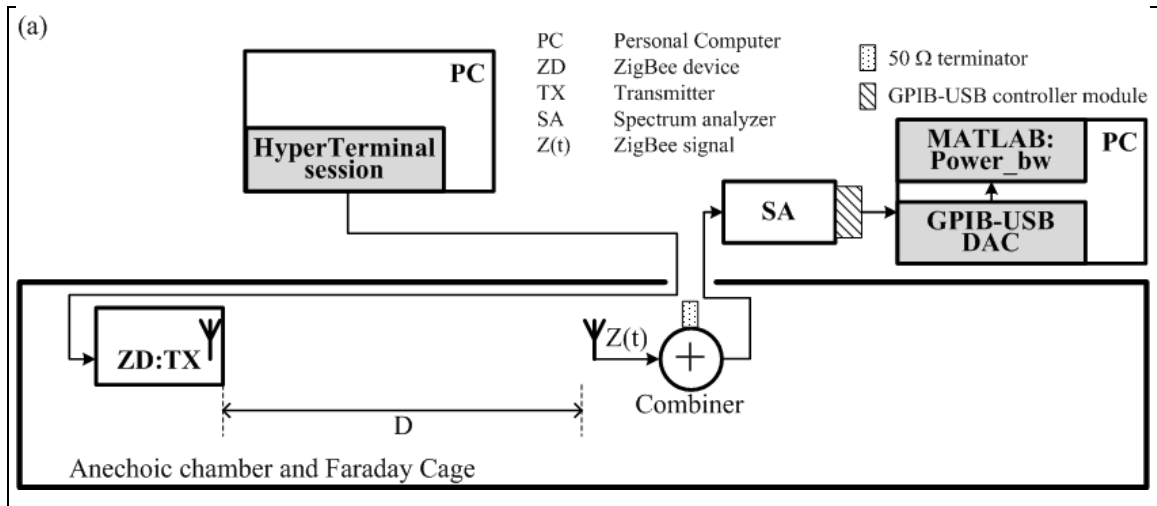


Fig. 5.2 ZigBee-synthetic noise performance evaluation setup for PER measurements.



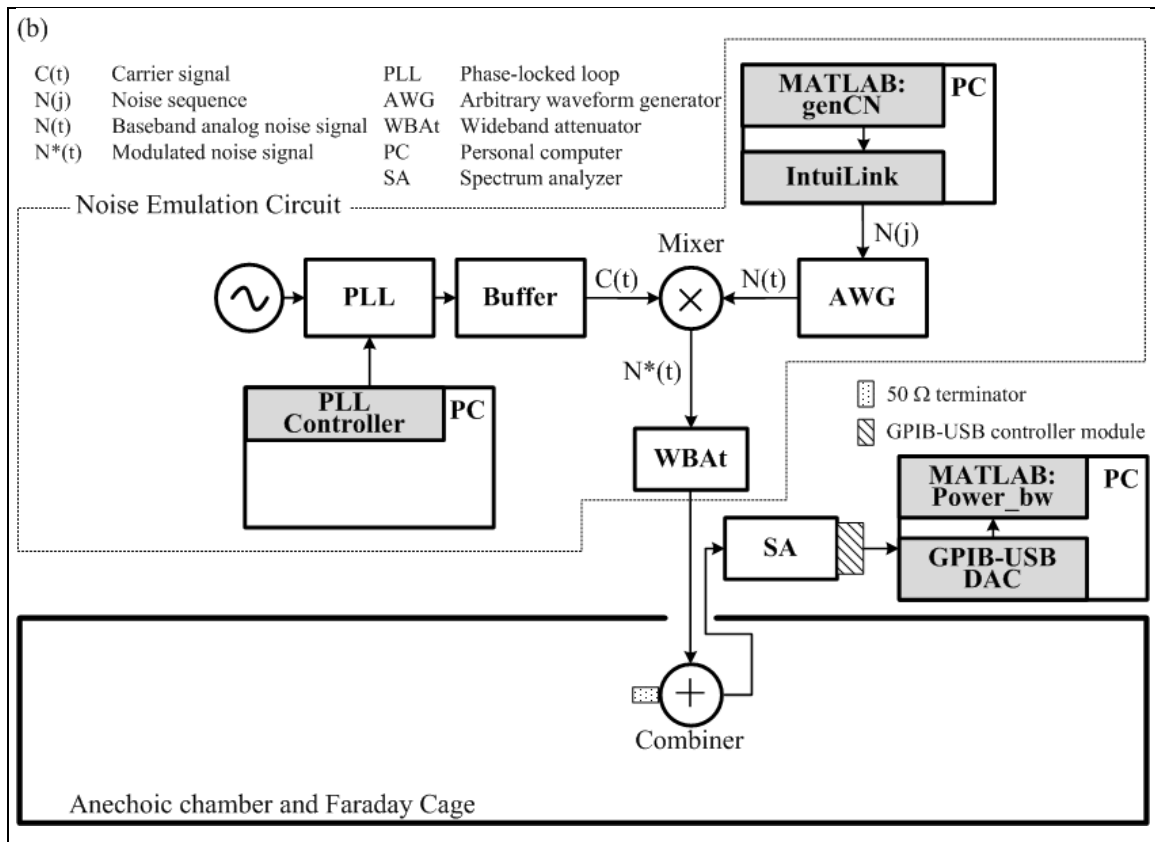


Fig. 5.3 ZigBee-noise performance evaluation setup for SNR measurements; (a) setup for measuring power of received ZigBee signal, and (b) setup for measuring power of noise signal.

The communication network consists of two nodes, a ZigBee transmitter and a ZigBee receiver. This configuration is the most basic of all the topologies (star and peer-to-peer) that ZigBee supports. Such configuration is chosen because it is the simplest case that can provide a basis for understanding the operation of ZigBee and presents a basis for extending to a network consisting of more nodes. The ZigBee transmitter is placed at various distances. The ZigBee transmitter will use its built-in F-antenna, whereas the ZigBee receiver will use an external antenna to enable the superposition of noise signal onto the received ZigBee signal.

The noise emulation circuit consists of an arbitrary waveform generator (AWG), a phase-locked loop (PLL), a buffer constructed with a wideband attenuator (WBAt) followed by a wideband amplifier (WBAm), a mixer, and another wideband attenuator.

In general, the AWG serves as a baseband noise source to the performance evaluation setup. In other words, the AWG can output noise to match the diversity of noise classes found in the operating environment of industrial applications, provided it gets the appropriate noise samples as its input. In particular, samples from a fractal noise sequence (refer to Ch. 4.4 for its synthesis) are fed or uploaded to the AWG. Let the noise data-points be denoted as $N(j)$. The model of AWG that I used (33120A) can accept up to four user-defined waveforms (in this case, four noise sequences) in its non-volatile memory. The purpose of the AWG is to produce an analog version of the supplied noise sequence. Let the analog noise signal be denoted as $N(t)$. The generated analog noise signal is a low-pass signal that has a maximum bandwidth of 10 MHz. $N(t)$ is band-limited because of a constraint imposed by the equipment (i.e. the cut-off frequency of the internal filter⁸ of the AWG).

In order for $N(t)$ to interfere with the ZigBee transmission, the noise signal baseband frequencies will need to be shifted to gigahertz frequencies so that the noise spectrum is in a region that will influence or affect the ZigBee spectrum. The double-sideband, suppressed-carrier (DSB-SC) type of amplitude modulation, AM, is employed to produce a high-frequency noise signal. Let the modulated noise signal (i.e. the high-frequency noise signal) be denoted as $N^*(t)$. The DSB-SC AM is chosen because of its simple implementation, which is achieved by multiplying the modulating signal, in this case $N(t)$, with a carrier signal, $C(t)$. The PLL is used to generate a carrier signal with the desired *carrier frequency*, f_c . The mixer is used as the multiplication operator. A buffer is placed between the output of the PLL and its load (the mixer)

⁸ 10 MHz, 7th-order Bessel filter (Agilent 33120A User's Guide [Agil02], pp. 276)

to isolate the PLL from varying load conditions. For example, when the PLL changes its oscillation frequency in response to a variation in its load impedance, this effect is known as frequency pulling. The implementation of a buffer to mitigate the frequency pulling effect experienced by the PLL is done by using an attenuator followed by an amplifier [Maxi03, Mini08].

In the original setup design, a voltage controlled oscillator (VCO), was used as the frequency synthesizer component. However, it had been replaced with a PLL and a buffer because a closed-loop system is a better means for producing stable and accurate output frequency. A frequency synthesizer with open-loop architecture such as the VCO, had been found to produce output frequency that would drift. As a consequence, the modulated noise spectrum would drift. Since shifts in carrier frequency occurred during the transmission of packets and at a random fashion, this caused the noise power affecting ZigBee spectrum to be inconsistent. A description of the causes, suggested counter measures, and resolutions that I attempted to reduce the VCO output frequency instability can be found in Appendix C.3.

Finally, a WBA_t is placed at the output of the mixer to reduce the power of the emulated high-frequency noise signal before it is added to the ZigBee receiver circuit to impact errors in the received packets. Noise is added one at a time. Monofractal noise with spectral exponent 0.4, 0.8, 1.2, and 1.4 were used to measure PER versus SNR for ZigBee communication under synthetic noise. These monofractal noise signals were selected because they matched the complexity of samples of real noise recordings obtained from starter source of tractors [WoKF08].

5.1.3 System Revision: What Did Not Work and Improvement Done

From the basic experimental setup design (in Appendix C.6 (Fig. 1)), I have contributed the following revisions to making the experimental setup to evolve to the current design (Figs. 5.2 and 5.3) and the current experimental procedure (see section 5.1.5).

The original setup (version 0). Please refer to Appendix C.6 (Fig. 1).

Problems encountered at the time of this version include:

- (i) Inaccurate PER due to drifting of carrier frequency.
- (ii) Injected noise spectrum does not follow noise model due to low-pass filter, LPF.
- (iii) Inefficient noise datapoints uploading.

The respective problems were caused by:

- (i) The sensitivity of VCO, an open-loop system for synthesizing carrier frequency.
- (ii) Contribution from the undefined region (Fig. 4.17) were doubled and centered on ZigBee center frequency due to output of 1 MHz-LPF being doubled after amplitude modulation.
- (iii) Slow serial uploading of noise datapoints onto AWG due to HyperTerminal software.

Solutions to the mentioned problems:

- (i) The first attempt was to improve the stability at the input of the VCO (by adding bypass capacitors to the power source, and the input pins (V_{cc} and V_{tune}) of the VCO being supplied by adjustable voltage regulators with ripple rejection). However, the synthesized carrier frequency still drifts. The second attempt that worked was to replace the VCO module with a closed-loop device (i.e., a phase-locked loop, PLL) and a buffer between the PLL and its load to increase stability in the input and output of the PLL. Please refer to Appendix C.2 for the circuit of the improved adjustable voltage regulator, and to

Appendix C.3 for my notes on the findings, discussion and conclusions reached during my investigation on the carrier frequency drifting issue.

- (ii) Do without the LPF.
- (iii) Use IntuiLink (40 sec uploading time) instead of HyperTerminal (45 minutes). For uploading datapoints onto AWG procedure, please refer to Appendix C.4 for the HyperTerminal method and to Appendix C.5 for the IntuiLink method.

The improved setup (version 1). Please refer to Appendix C.6 (Fig. 2).

Problems encountered at the time of this version include:

- (i) Inaccurate measurement by spectrum analyzer, SA, indicated by REF UNLOCK due to having applied the SA's 10 MHz source to PLL REF.
- (ii) Inaccurate analog generation of baseband.
- (iii) Inaccurate PER due to the GUI of the ZigBee Radio Test Script of Freescale Test Tool Application (from the evaluation kit v4.0 CD that came with the Freescale 13193 Evaluation Kit [Free05a]) that recorded packets received as packet dropped.
- (iv) Inaccurate PER or channel noise.
- (v) Invalid PER data being returned.
- (vi) Injected noise spectrum does not follow noise model.
- (vii) Selection of input noises may not resemble real broadband physical noise.
- (viii) Inaccurate SNR due to noise injected at the transmitter.
- (ix) Inaccurate SNR due to bad measurement method.

The respective problems were caused by:

- (i) The connection diagram (which shows the same spectrum analyzer being used for providing the 10 MHz sinusoidal source as well as for measurement) provided in App Note 101 [Sire07a] for the PLL module is not feasible in reality.
- (ii) Two causes, namely (a) amplitude quantization error from not using the full range 0-4095 DAC codes or full range floating-point values between -1 and 1, and phase truncation error from using a non-16000 points sequence; (b) input noise does not follow noise data points due to setting the AWG playback frequency to a wrong value.
- (iii) This scenario only happens when the GUI screen is moved. It appears that movement of GUI screen by mouse causes the GUI program to focus on redrawing instead of multi-tasking (i.e., redraw and also capture transmitted packets).
- (iv) Due to presence of (a) human, and (b) location of the noise emission equipment.
- (v) Due to a software bug in PER_RX code.
- (vi) Due to amplitude modulation, says finding from analysis (see sub-subsections 4.5.3.3 and 4.5.3.4).
- (vii) Arbitrary choice of spectral exponent when generating synthetic noise sequence.
- (viii) The values used for calculating SNR were all obtained at the transmitter end. So, the calculated SNR is actually SNR at the transmitter, despite the different nodes separation used.
- (ix) Any trace data output from the spectrum analyzer (model HP 8546A [Agil94]) is stored in a format (.trc) that is not readily readable since the dedicated software for doing so is unavailable and that technical support is no longer available since the particular model of spectrum analyzer has become obsolete. As such, capturing of the noise and the ZigBee spectra would need to be done using a different spectrum analyzer, causing the measurements to be "offline", i.e., done after the PER experiments. Since the spectrum

analyzer (Anritsu 9 kHz to 3GHz MS2661C [Anri]) that has a spectrum capture software is situated at a different lab than where the PER experiments were performed, the noise emulation circuit will need to be resetup, which may lead to the accuracy of the measurements being compromised.

Solutions to the mentioned problems:

- (i) Use an AWG dedicated for supplying 10 MHz reference signal.
- (ii) To prevent cause (a), according to the highlighted constraints on the arbitrary waveform input found in pages 178 and 278 of Agilent 33120A User's Guide [Agil02], I should synthesize noise to have these configurations: 16000 points, and normalized between -1 and 1. To prevent cause (b), according to pg 276 [Agil02], I should synthesize noise using 2.5 kHz playback frequency.
- (iii) Get rid of TestTool GUI. Since then, Freescale's basic PER test [Free08a], an SMAC [Free08b] application, has been used and modified to suit my experiments.
- (iv) For (a), automate PER monitoring system. For (b), two measures were taken, namely, the equipment is brought outside of anechoic chamber, and the noise injection circuit is covered with a metal casing.
- (v) Remove hard reset in code.
- (vi) Apply monofractal noise by a 4:1 carrier offset to monofractal noise region ratio.
- (vii) Create monofractal noise signals with its range of spectral exponents matching that of real noise emitted from starter source of tractors.
- (viii) Inject noise at the receiver.
- (ix) Perform spectrum capture of the ZigBee and the noise signals from the spectrum analyzer used in the PER experiments, this time using a custom-written trace data acquisition software that extracts the data from the general purpose interface bus (GPIB) of the SA

through a GPIB-USB controller module and stores the trace data onto a personal computer in a text file format with just amplitude and frequency information of the trace data. Please see Appendix C.7 for the procedure and the code for the data acquisition through GPIB C# application that I wrote.

The improved setup (version 2). Please refer to Fig. 5.2.

Problems encountered at the time of this version include:

- (i) Replaced the kaput laptop with a new laptop, but PLL software cannot be run.
- (ii) Laptop Vista OS does not read serial data returned from PER_RX board.
- (iii) Potential problem: reflection may cause cancellation of signals thus creating more bit error.
- (iv) Laptop overheating.
- (v) No packets received at the receiver.
- (vi) Inconsistent baseline PER.
- (vii) Inaccurate SNR.
- (viii) SA does not show spectrum of received ZigBee packets.
- (ix) Unable to capture ZigBee spectrum.
- (x) Input noise originating from baseband monofractal noise with spectral exponents ≥ 2 has no effect on ZigBee packets transmission.

The respective problems were caused by:

- (i) New laptop does not have parallel port.
- (ii) Serial-to-USB driver from new SMAC code.
- (iii) Ground reflection or reflection from floor and metal strip panels on floor.
- (iv) Due to running many program for the experiments.

- (v) Carrier noise (not even noise itself) overpowers the packet transmission.
- (vi) Transmit and receive antennas' off-alignment.
- (vii) Due to the use of a second antenna to capture ZigBee trace, while noise at combiner.
- (viii) The SA's settings were incorrectly configured.
- (ix) Spectrum of received ZigBee packet was intermittent.
- (x) Due to the 4:1 ratio configuration for input noise signal and ZigBee signal (see Fig. 4.16) and the steepness of the baseband monofractal spectral exponent, the noise power at the tail of the input noise spectrum (i.e., the region where noise impacts ZigBee) is too low (looks more like noise floor compared to the input noise steep PSD at lower frequencies) to cause any packet error during the ZigBee transmission.

Solutions to the mentioned problems:

- (i) Add an express Card that provides a parallel port, care must be taken if OS is not windows 2000 or XP, for e.g. with Vista, manually enter parallel port address to have PLL Controller work on this OS.
- (ii) Expect for serial data from serial port instead of USB port.
- (iii) Overcome that by padding the transmit path and metal strips in the anechoic chamber with RAMs.
- (iv) Two solutions, namely (a) bought a USB cooler fan and aluminum cooler panel for laptop, solved laptop overheating issue - can dissipate heat, and (b) brought in a fan.
- (v) Add an attenuator at mixer's output.
- (vi) Adjust antenna to ensure orientation provides maximum gain and sensitivity (when aligned).

- (vii) Still after-the-fact solution, but without needing to resetup equipment where a lot of configurations may change, both ZigBee and noise measured at the output of combiner provides consistency in terms of attenuation.
- (viii) Two measures taken, namely (a) checked and verified that cable was not broken, and (b) the EMI receiver ATN is set to MANual (instead of AUTO) and to 0 dB, by putting to 0 dB compared to 10 dB (from auto settings), the noise floor is lowered by 10 dB.
- (ix) The MAX-HOLD command is recommended in the HP 8546A user's guide [Agil94] for viewing unstable input signal.
- (x) Note to self that with the current performance evaluation system and procedure, PER experiments cannot be carried out for input noise originating from monofractal noise with spectral exponent ≥ 2 .

5.1.4 Apparatus and Materials

Below is a list of experimental setup apparatus and materials (hardware and software, inclusive) that I used for my performance evaluation experiments. For results reproducibility purpose, the indicated tools are recommended. See Figs. 5.2 and 5.3 for the experimental setup layout or Appendix C.8. See Appendix C.9 for a complete listing of the hardware and software required for this experiment.

- For test and measurement: one oscilloscope (Tektronix TDS 210), two arbitrary waveform generators (both are Agilent 33120A) where AWG#1 is dedicated for storing baseband noise signals and AWG#2 is for supplying a 10 MHz reference signal to the PLL evaluation board, two spectrum analyzers that range up to 3 GHz (HP 8546A EMI Receiver and MetaGeek 2.4 GHz Wi-Spy with Chanalyzer software), one multimeter (Fluke 112 true RMS digital multimeter), MATLAB editor (R2007b), and one personal computer (Dell laptop).

- For noise emulation circuit: one frequency synthesizer (with PLL400-2450A part on a PLL evaluation board), two attenuators (Mini-Circuit ZX73-2500), one amplifier (Mini-Circuit ZX60-4016E), one mixer (Mini-Circuit ZX05-30W), one external parallel port (Quatech SPPXP-100 ExpressCard IEEE-1284 parallel adapter), Sirenza PLL controller software, Agilent IntuiLink Waveform Editor, Agilent IO Libraries Suite, text editor (TextPad), the author's monofractal noise generation MATLAB program (in Appendix A.1).
- For powering and tuning PLL, its buffer and attenuator: three sets of adjustable voltage regulators (two with improved ripple rejection and one with minimum program current, see Appendix C.2 for circuit diagrams and recommended circuit components).
- For ZigBee communication: Two ZigBee nodes (both use Freescale MC13192-EVB [Free07] boards. The difference is that the transmitter board uses its internal antenna, whereas the receiver board uses an external antenna), one combiner (Mini-Circuit ZN2PD2-50), one SMA microstrip patch antenna (custom-made to resonate at 2.405 GHz), one device for flashing code (P&E Microcomputer Systems Multilink BDM Debugger, USB HCS08/HCS12), one USB hub (Belkin 4-port USB hub on self-powered mode), the author's PER monitoring software (in Appendix A.2), Freescale CodeWarrior for Microcontroller, Freescale BeeKit, terminal emulation program (Hilgraeve HyperTerminal Private Edition), and Microsoft Excel.
- For SNR measurements: GPIB-USB adapter (Prologix GPIB-USB controller module), the author's trace data acquisition GUI C# program (in Appendix A.3), the author's trace signal power calculation MATLAB program (in Appendix A.4), one 50 Ω terminator (Wiltron 28S50-2, SMA BB), and Visual Studio C#.
- Others: cables (see listing in Appendix C.9 (Table 1)), adapters (three SMA-SMA barrel adapters, three BNC-SMA adapters, one N-BNC adapter, one N-SMA adapter), DC power

supplies (+5 V DC Voltage, and +11 V DC Voltage), one physical noise source (Dream Cheeky USB plasma globe), three bypass capacitors (4.7 μ F electrolytic, 0.1 μ F ceramic disc, 0.01 μ F ceramic disc), one roll of foil tape, one screwdriver, one breadboard, one metal box with cover, three power bars and a bunch of jumper wires.

5.1.5 Procedure

The procedure for the performance evaluation of ZigBee under synthetic noise experiment consists of three parts, namely the preliminary settings and configurations, the PER measurements, and the SNR measurements. See Appendix C.8 for the diagrams of the setup used for the PER measurements and the SNR measurements with full cable-connectivity shown.

5.1.5.1 Preliminary

1. Gather power supplies, test and measurement equipment, cables, connectors and adapters, circuit components and jumper wires. Use the hardware listing in Appendix C.9 (Table 1) and the diagrams in Appendix C.8.
2. Open a new Excel file. Prepare three excel sheets for recording data, one sheet for each distance. A suggested format for data recording is given in Table 5.1. Each of the created sheets contains five copies of Table 5.1, where one table is for recording the ZigBee transmission without noise experiment and the remaining tables are for recording the ZigBee transmission for each of the four different noise types experiment, respectively. Save the Excel file. Note: The Excel file that contains both the recorded and the calculated data for my performance evaluation under synthetic noise experiment can be found in Appendix D.

Table 5.1 Table for recording the noise amplitude level tuned from the AWG that will induce the targeted PER%, the name of the files that stored the PER monitoring output and the trace data acquisition output, and the computed PER and SNR measurements for each input noise type for the ZigBee transmission between 2 nodes separated at the specified distance.

Distance = m		ZigBee transmission with noise (Y/N) =		If Y, input noise type, β =		Computed SNR (dB)
Noise Amplitude		PER (%)		Filename		
mV_{pp}	mV_{rms}	Targeted	Measured	PER Data (.txt)	Trace Data (.txt)	

3. Install programs and drivers on the personal computer (in my case, a laptop). See software listing in Appendix C.9 (Table 2).
4. Measure cable attenuation and label cables.
 - a. METHOD: Measure cable attenuation using tracking generator and EMI receiver.
 - b. STEPS:
 - 1) Set the center frequency, span, and REF level. I have used 2.405 GHz, 15 MHz, and 10 dBm, respectively. This is because the attenuation or gain measured is for that range of frequencies.
 - 2) Connect cable-under-test between the tracking generator output and input 2 on HP 8546A EMI Receiver.
 - 3) Press the TRACK GEN button.
 - 4) Apply source power by pressing the SRC POWER softkey to ON, and then adjusting the knob to set SRC POWER to the desired value in dBm. I have used 0 dBm and 10 dBm.
 - 5) Read marker value at center frequency. To determine whether it is a gain or attenuation, use MKR - SRC POWER, e.g. MKR = -8.87 dBm, SRC POWER = -10

dBm, MKR - SRC POWER = +1.13 dBm (GAIN). A positive value denotes GAIN, whereas a negative value denotes ATTENUATION.

- 6) Label the cable with its specific measured attenuation or gain value.
- 7) Repeat steps 1) to 6) for another cable.
- 8) Categorize the cables into “cables without attenuation” and “cables with attenuation”

NOTE: For consistency, when taking measurements, always use the same REF level and SRC power across all measurements.

5. Upload noise datapoints onto AWG #1 using IntuiLink (see Appendix C.5 for the step-by-step instruction). Justification: Uploading datapoints through IntuiLink takes only 40 seconds per noise signal, instead of 40 minutes through HyperTerminal, which was the method that I originally used. If IntuiLink is not available, the upload procedure using HyperTerminal can be used (see Appendix C.4).
 - a. uploaded input noise datapoints (the first four out of these: beta = -0.4, -0.8, -1.2, -1.6, 0.0, -1.0, -2.0, -3.0) onto the AWG
 - 1) with IntuiLink 1.5 on windows Vista and USB to serial convertor cable and RS-232 cable between the laptop and the AWG, cannot detect the port that the instrument is connected to
 - 2) have to launch Agilent Connection Expert from Agilent IO Control -> refresh all -> add instrument to the port that the USB to serial convertor is attached to (do this only once) -> connect to instrument, then only launch IntuiLink -> Connection -> Connect, Open File ->Send Waveform (here pretty much follows the uploading datapoints to AWG using IntuiLink guidelines that I created)
6. Assemble voltage regulator circuits on breadboard. See Appendix C.2.

- a. To filter out electrical noise picked up by the jumper wires, thus affecting the voltage regulator tuned output, I have used bypass capacitors at the power supply end to my voltage regulators.
7. Solder jumper wires to attenuators ($V+$, V_{cont} , and two GND pins) and amplifier ($V+$ and one GND pins).
 8. Assemble and configure the noise injection circuit. Refer to Appendix C.6 (pg. 2) for the voltage and power configuration while doing the following:
 - a. Set carrier frequency. Figure 5.4 shows the frequency synthesizer module with its inputs and output.

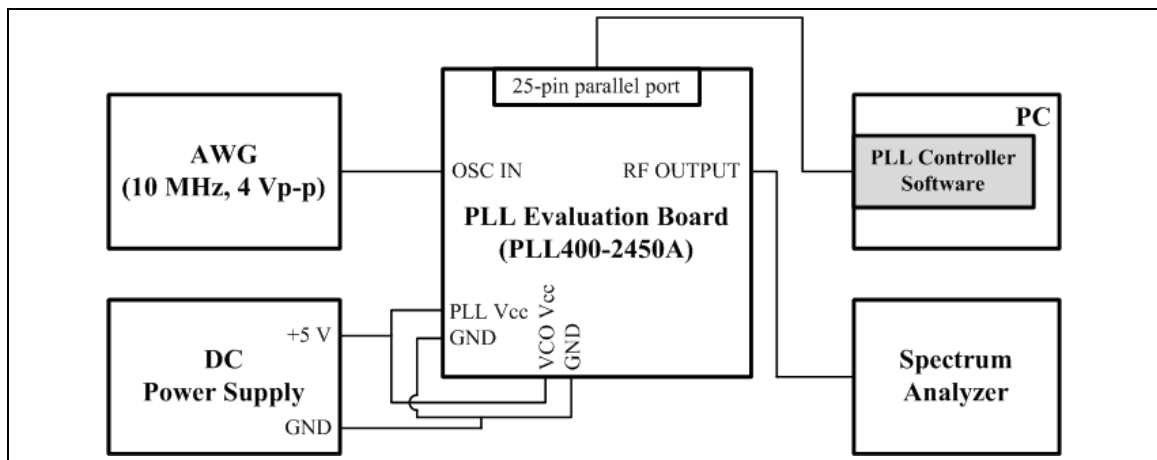


Fig. 5.4 PLL connection diagram (after [Sire07a]).

- 1) Using jumper wires, connect power supply (+5V) to PLL V_{cc} and VCO V_{cc} pins, and connect power supply (GND) to PLL GND and VCO GND pins.
- 2) Turn on AWG #2. Set its FREQ and AMPL to 10 MHz and 4 V_{pp} . Justification: According to App Note 101, the amplitude of the reference signal should be between 2 and 4 V_{pp} for a +5 V operation of PLL integrated circuit. To verify the AWG output frequency and amplitude, connect AWG #2 output to an oscilloscope with a coaxial cable with BNC-M connectors.

- 3) Connect AWG #2 output to the PLL REF input with a coaxial cable with BNC-M connectors and an SMA-M to BNC-F adapter. Justification: Sirenza Microdevices' App Note 101 suggested to use the 10 MHz reference from the back of a currently used for measurement spectrum analyzer. I have found the suggestion to be unfeasible as doing so will produce a REF UNLOCK warning message of the spectrum analyzer and will affect the measured values to be inaccurate due to an unlocked measurement. As such, a separate AWG is dedicated to solely generate a 10 MHz reference signal for the PLL.
- 4) Connect the PLL parallel port to laptop parallel port using a 25-pin printer cable (I used two such cables, one with M-F connectors and the other with M-M connectors).
- 5) Load PLL controller software on laptop. For a laptop that does not have a parallel port, an external parallel port device was used. Doing so will require a manual input of the parallel port address used by the PLL Controller software. The parallel port address can be found by going to System->Device Manager->Ports (COM & LPT), and right click on Quatech SPPXP-100 ExpressCard IEEE-1284 Parallel Adapter (LPT1) to go to Properties->Resources->I/O Range->Setting. Say CFF8-CFFF is being indicated. Now, go to "Tools->Options" within the PLL Controller software and enter CFF8 (the first four bytes of the I/O range setting found in step 3) as the parallel port address.
- 6) On the PLL controller software menu bar, click on File->Open App Note... and select the appropriate application note. In my case, appnote113-.van was chosen, as indicated in the product specification [Sire07b].
- 7) On the PLL controller software menu bar, enter all the required configurations (option, reference input, channel spacing, and output frequency). If unsure, refer to

App Note 113 for configuration values. I have used 207000 for option, 10 MHz for reference input, 1000 MHz for channel spacing (i.e. step size) and 2402 MHz as the desired output frequency. Click on “Calculate Pattern” button. Justification: The output frequency 2402 MHz was (the best choice; cannot even try out other channels) chosen due to restrictions from (i) the microstrip patch antenna was made to resonate at 2405 MHz, (ii) the range of frequencies that the PLL can synthesize is between 2400 MHz and 2500 MHz in steps of 1 MHz, (iii) bandwidth of AWG-generated analog baseband noise is 10 MHz due to its internal filter cut-off frequency, (iv) avoid contribution from the undefined region of the modulated signal PSD, (v) ZigBee channel 12 (2410 MHz), and (vi) the 4:1 carrier-to-selected-ZigBee-channel-frequency interval offset to ZigBee 6 dB channel width ratio.

- 8) Power on the spectrum analyzer. Have it set to use the 20 MHz - 2.9 GHz input. Plug in an N-SMA connector on the mentioned input. Set the start and stop frequencies to 2.4015 GHz and 2.4115 GHz, respectively (i.e. a 10 MHz span).
 - 9) Connect the PLL output to the spectrum analyzer with a coaxial cable with SMA-M connectors.
 - 10) Turn the power supply +5 V connected to the PLL evaluation board on.
 - 11) Click the “Send All” button on the PLL controller software. Doing so will reprogram the PLL to lock onto the desired output frequency.
 - 12) Observe PLL output frequency on the spectrum analyzer to verify its output. At 0 dBm REF level on the spectrum analyzer, the PLL output signal peak amplitude was measured to be -1 dBm. When done, unplug the spectrum analyzer.
- b. Set the buffer and the mixer.

- 1) Attach an attenuator to the PLL output using a coaxial cable with SMA-M connectors. Power the attenuator. Adjust the attenuator V_{cont} (by adjusting the potentiometer of its voltage regulator) so that -10 dBm is achieved at the attenuator output.
 - 2) Attach an amplifier at the attenuator output using an SMA-M to SMA-M barrel adapter. Power the amplifier. Adjust the attenuator V_{cont} so that +7 dBm is achieved at the attenuator output.
 - 3) Attach the amplifier output to the mixer's LO port using an SMA-M to SMA-M barrel adapter. The mixer's output power as seen from its RF port should measure approximately -28 dBm.
 - 4) Attach an attenuator to the mixer's RF port using an SMA-M to SMA-M barrel adapter. Power the attenuator. Adjust the attenuator V_{cont} (by adjusting the potentiometer of its voltage regulator) so that -60 dBm is achieved at the attenuator output.
- c. Set AWG #1 (the AWG has been powered on since step 5 in the preliminary procedure).
- 1) Attach AWG#1 output to the mixer's IF port using a coaxial cable with BNC-M connectors and an SMA-M to BNC-F adapter.
 - 2) Set the AWG to output arbitrary waveform (N1_0DOT4), amplitude = 50 mV_{pp} (minimum), and frequency = 2.5 kHz.
9. Measured distances and put markers at 1 m, 2 m, 3 m, and 4 m on the anechoic chamber floor; accuracy $\pm 3\text{cm}$, measurements convert from inches to metres, and using tape on floor (by eyeing)
10. Upload PER_TX and PER_RX code to the transmitter and the receiver boards, respectively. Keep references ([Free08a] and [Pemi]) handy while doing the following:

- a. The steps or instructions for flashing Freescale board, for e.g. with PER_RX application
 - 1) Load CodeWarrior (CW), and open PER_RX.mcp
 - 2) On CW: Press the “Compile” button
 - 3) If no errors encountered during build process, make sure target power is OFF and Multilink is not connected to either the target or the PC
 - if powering target through USB: for target to be OFF, flip S106 switch to VDC, and connect USB cable between PC and target
 - if powering target through battery: for target to be OFF, flip S106 switch to USB, and connect battery to J105
 - 4) Connect Multilink's ribbon cable to the target. Make sure the red stripe on the ribbon cable is on pin 1 of board
 - 5) Connect Multilink to PC via USB cable. The Blue LED on Multilink should lit up
 - 6) Turn target power on. The Yellow LED on Multilink should lit up
 - if powering target through USB: for target to be ON, flip S106 switch to USB
 - if powering target through battery: for target to be ON, flip S106 switch to VDC
 - 7) On CW: Press the “Debug/Download” button
 - 8) On the “ICD Connection Manager” window that popped up, click the “Connect” button
 - 9) On the “Erase and Program Flash” window that popped up, click the “Yes” button
 - 10) After the flashing is completed, turn target power off
 - if powering target through USB: for target to be OFF, flip S106 switch to VDC, and leave USB cable connected between PC and target
 - if powering target through battery: for target to be OFF, flip S106 switch to USB, and disconnect battery to J105

- 11) Disconnect the USB B-M cable from the Multilink
 - 12) Disconnect the ribbon cable of the Multilink from the board
 - 13) Close the “True-Time Simulator & Real-Time Debugger” window that popped up
 - 14) Repeat steps 1) to 13), but this time for PER_TX.mcp
11. Configure the HyperTerminal sessions to assess the receiver board and the transmitter board, respectively.
- 1) Connect USB cable between PC and target, if it's not already connected. Say, the target is the receiver board. The target is powered through USB. Justification: I have decided to power the transmitter and the receiver nodes by USB of laptop instead of by 9V-battery in order to have a consistent (constant 5V DC) power supply so that there will not be any transmitter output power drop to affect the PER and SNR results.
 - 2) Go to control panel->system->device manager->Ports (COM & LPT), and check which COM number is belonging to which board (the transmitter or the receiver). For each board, connect and then disconnect the USB cable to the board to check which port number disappears under Ports (COM & LPT). It should be a number between 1 and 10
 - 3) Use a PC terminal program (such as HyperTerminal) and set the correct baud rate, data bits, parity, stop bits, and flow control. (38400, 8, None, 1, None) and COM port (as found out from previous step).
 - 4) In the PC terminal program, set the properties in the Properties->Settings tab->ASCII Setup... to have a check mark on “Send line ends with line feeds”, “Echo typed characters locally”, “Append line feeds to incoming line ends”, and “Wrap lines that exceed terminal width”

- 5) Repeat steps 1) to 4) for another target. Say, the target is the transmitter board.
12. At this stage, all the test and measurement equipment, and the noise emulation circuit have been powered up. Allow 30 minutes (warm up period) to lapse before proceeding with PER measurements and SNR measurements.

5.1.5.2 PER Measurements

PER Monitoring Software

The ZigBee/802.15.4 packets transmission is simulated at the physical layer using SMAC-based PER monitoring software. The PER monitoring software consists of two parts, the PER_TX for the transmitter application and the PER_RX for receiver application. These define the firmware to be uploaded onto the respective MC13192 evaluation boards, one board for transmitting packets, and one board for receiving packets.

Using the basic PER test program from Freescale (pg. 11 to 17 of AN3231 [Free08a]) as the base template, I have customised the basic program by adding the following features:

- (i) Automation capability, (i.e., enabled command input from keyboard to be sent over serial communication instead of physical press of onboard buttons).
- (ii) Transmitter labels transmit packets according to their transmission order within the 1000 packets to be transmitted.
- (iii) Receiver extracts labels of received packets and use that to (a) monitor the order of received packets, (b) compute labels of dropped packets, and (c) determine whether the dropped packet pattern is random or in bursts.

Since the basic code was SMAC-based, the reference needed when coding is [Free08b].

Modification to the original code is required:

- (i) To avoid using manual button commands so that the presence of human during the PER measurements experiment is not required. By doing this, the possibility of packet error due to presence of human (needed to press button on board) can be prevented.
- (ii) For ease of use. Experimenter does not need to access (open and close the door) the anechoic chamber to reset the receiver and the transmitter each time.
- (iii) To better understand the effects of the input noise type upon the ZigBee transmission.

The basic program's functionality includes controlling of the starting of each application (the receiver listening on the set channel for incoming packets, and the transmitter transmitting 1000 fixed-size-and-content packets to the receiver, respectively) and the changing of the communication channel frequency all done through the press of specific buttons on-board. In addition to that, the receiver application output some information pertaining to the received packet such as displaying the number of packets received thus far, and the overall number of received packets upon receiving a done packet signalling completion of transmission from the transmitter.

The basic program caters for asynchronous receiving and transmitting. Hence, with respect to the activating sequence for the two wireless nodes, the experimenter has to enforce synchronization by first starting the receive board first and then only starting the transmit board. Otherwise, inaccurate PER reading will be resulted. In the event of one or both boards stalling or not functioning, a hard reset is required and is done through powering off and repowering the board (unplug the board attached USB cable from the USB hub) from the USB hub from outside the anechoic chamber.

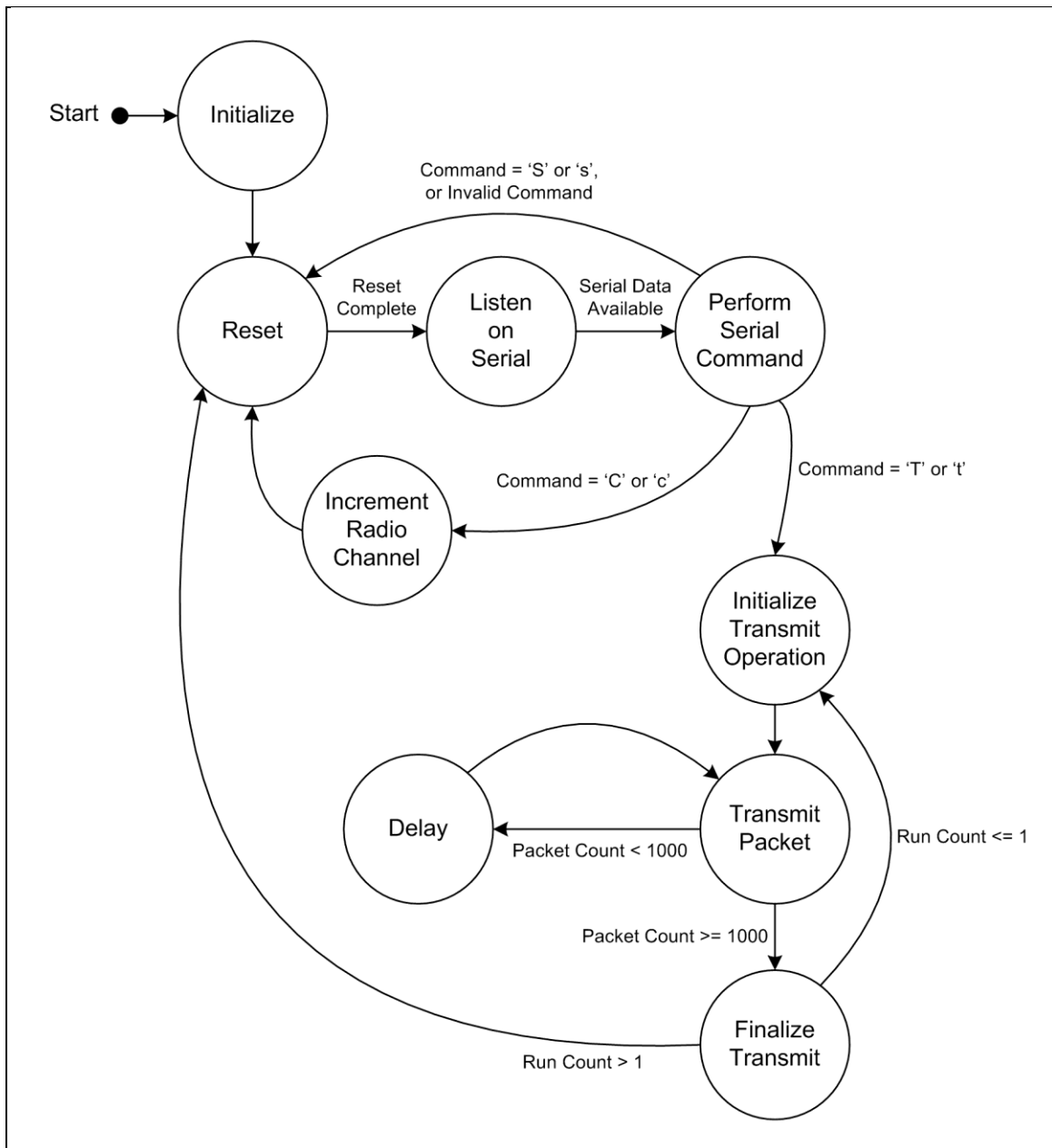


Fig. 5.5 State Diagram for Transmitter Application.

The C code written (PER_TX.c in Appendix A.2) for the transmit operation is done in terms of states. As such, Fig. 5.5 is included to reflect such representation. The following states (*Initialize*, *Reset*, *Initialize Transmit Operation*, *Transmit Packet*, *Delay*, and *Finalize Transmit*)

from the original code have been modified to my needs. Meanwhile, the newly added states are *Listen on Serial* and *Perform Serial Command*, and *Increment Radio Channel*.

The functionality of each state and the event that signals the transition to the next for the transmitter application depicted in Fig. 5.5 is described in the following:

- Initialize: This is where the microcontroller (enabling serial communication and interrupt from keyboard) and the radio (setting the radio to default output power, i.e., 0 dBm, and default channel number, i.e., 0 denoting 2.405 GHz) on the evaluation board that acts as a transmitter is initialized. Then, the state goes to *Reset*.
- Reset: Sets *run* counter to default value (one). Then, the state goes to *Listen on Serial*. An additional step (resetting channel number to the default value) is executed when user input ‘S’ or ‘s’.
- Listen on Serial: Prompts the user to input a 1-letter command (need not hit the Return key afterwards). The letter can either be a small letter or a capital letter, where ‘S’ signifies start reset, ‘C’ signifies change channel, and ‘T’ signifies transmit packet. Then, the state goes to *Perform Serial Command*.
- Perform Serial Command: When ‘S’ is pressed, the state goes to *Reset*. When ‘C’ is pressed, the state goes to *Increment Radio Channel*. When ‘T’ is pressed, the state goes to *Initialize Transmit Operation*. When any other character is pressed, an error message is displayed prompting user to re-enter command.
- Increment Radio Channel: The channel number is incremented. If channel number exceeds maximum value, channel number is reset to the default value. Then, the state goes to *Reset*.
- Initialize Transmit Operation: The data portion (18 bytes) of the packet to be transmitted is assigned a fixed content. After that, the *packet count* counter is reset to zero. Then, the state goes to *Transmit Packet*.

- Transmit Packet: The *packet count* counter is incremented. After that, the last two bytes of the packet data portion are assigned the *packet count* value. Then, the packet is transmitted. Next, if *packet count* is at maximum value, the state goes to *Finalize Transmit*, else the state goes to *Delay*. Note that the hard-coded maximum value (1000) in this case sets the limit to the number of packets to be transmitted per run.
- Delay: Set the amount of delay before transmitting the next packet. Then, the state goes to *Transmit Packet*.
- Finalize Transmit: Transmits twenty DONEDONE packets to tell receiver that the 1000-packet transmission has ended. Twenty is just a safety measure so that in the event of interference (where the transmitted DONEDONE packets could be lost) from noise, it is hoped that at least one DONEDONE packet will reach the receiver. After that, the *run* counter is incremented. If *run* exceeds maximum value, the state goes to *Reset*, else the state goes to *Initialize Transmit Operation*. Note that the hard-coded maximum value (1) in this case sets the limit to the number of runs of the 1000-packet transmission.

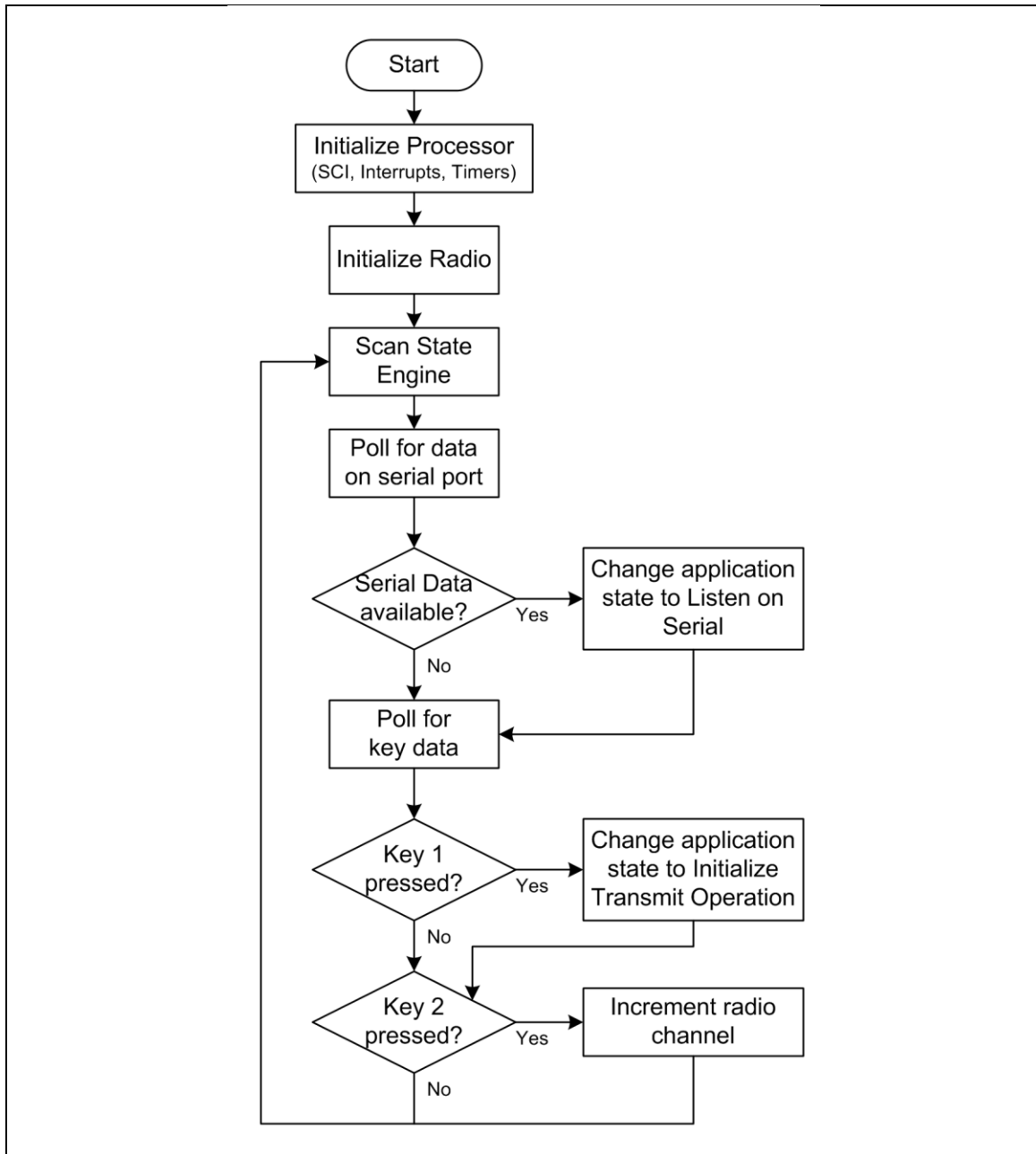


Fig. 5.6 Main Program Flowchart for Transmitter Application.

Fig. 5.6 shows the overall functionality of the PER_TX code, where the “Scan State Engine” block’s functionality is represented by Fig. 5.5.

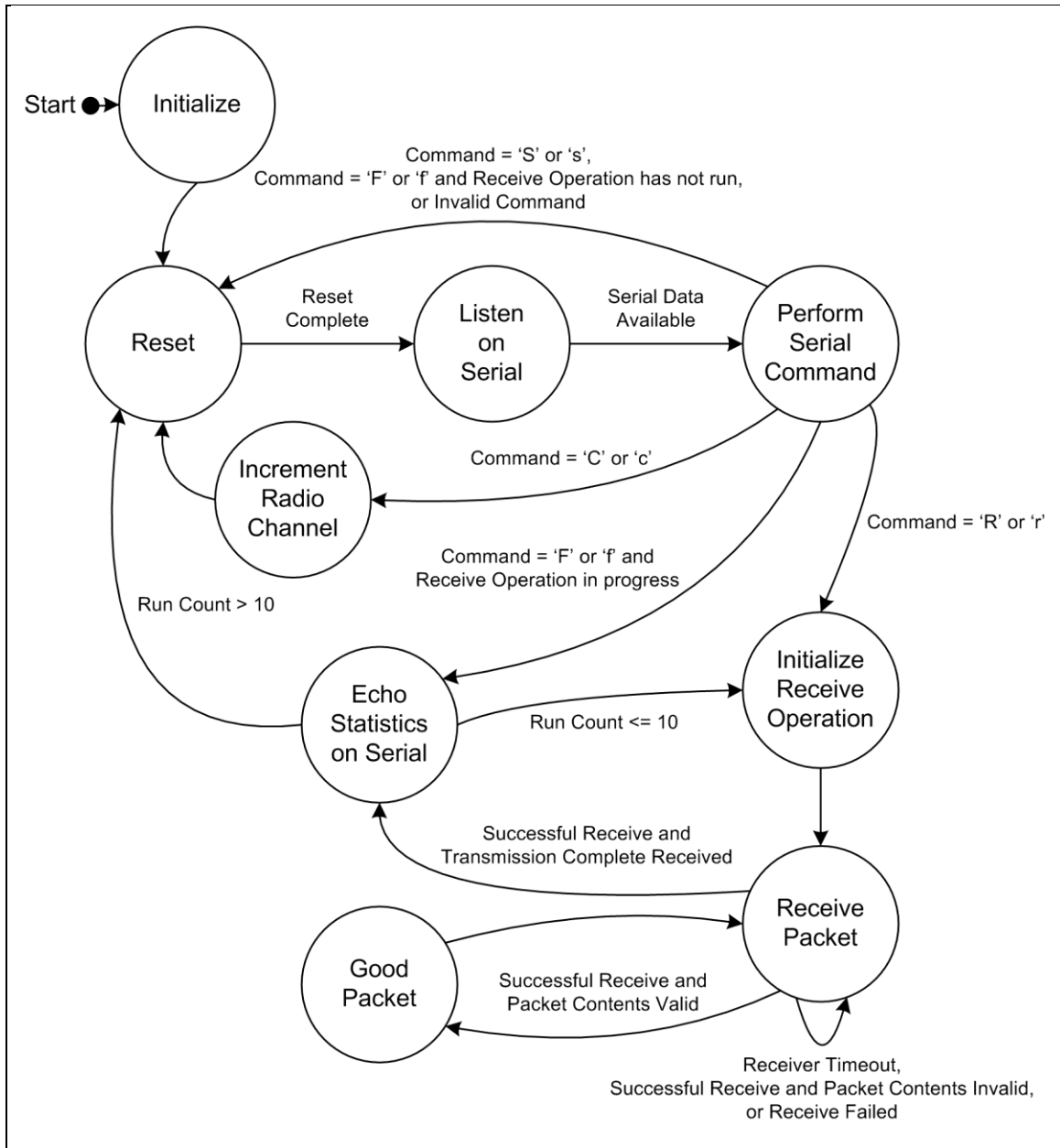


Fig. 5.7 State Diagram for Receiver Application.

The C code written (PER_RX.c in Appendix A.2) for the receive operation is done in terms of states. As such, Fig. 5.7 is included to reflect such representation. The following states (*Initialize*, *Reset*, *Initialize Receive Operation*, *Receive Packet*, *Good Packet*, and *Echo Statistics*

on Serial) from the original code have been modified to my needs. Meanwhile, the newly added states are *Listen on Serial* and *Perform Serial Command*, and *Increment Radio Channel*.

The functionality of each state and the event that signals the transition to the next for the receiver application depicted in Fig. 5.7 is described in the following:

- Initialize: This is where the microcontroller (enabling serial communication and interrupt from keyboard) and the radio (setting the radio to default output power, i.e., 0 dBm, and default channel number, i.e., 0 denoting 2.405 GHz) on the evaluation board that acts as a receiver is initialized. Next, the expected content of a received packet is setup. Then, the state goes to *Reset*.
- Reset: Sets *run* counter to default value (one). Then, the state goes to *Listen on Serial*. An additional step (resetting channel number to the default value) is executed when user input 'S' or 's'.
- Listen on Serial: Prompts the user to input a 1-letter command (need not hit the Return key afterwards). The letter can either be a small letter or a capital letter, where 'S' signifies start reset, 'C' signifies change channel, 'R' signifies receive packet, and 'F' signifies force statistics printing. Then, the state goes to *Perform Serial Command*. Note that the command 'F' is used when the receiver is stuck at *Receive Packet* state (due to no DONEDONE packet is being received for the current run of receive operation) to force the state to change to *Echo Statistics on Serial*.
- Perform Serial Command: When 'S' is pressed, the state goes to *Reset*. When 'C' is pressed, the state goes to *Increment Radio Channel*. When 'R' is pressed, the state goes to *Initialize Receive Operation*. When 'F' is pressed and Receive Operation has not run, the state goes to *Reset*, else the state goes to *Echo Statistics on Serial*. When any other character is pressed, an error message is displayed prompting user to re-enter command.

- Increment Radio Channel: The channel number is incremented. If channel number exceeds maximum value, channel number is reset to the default value. Then, the state goes to *Reset*.
- Initialize Receive Operation: The holder for storing a received packet is initialized. After that, the counters (*packet received good*, *lost packet count*, *previous received packet number*, *current received packet number*, and *total packets lost*) are reset to zero. Then, the state goes to *Receive Packet*.
- Receive Packet: This is where the radio listens on its set channel to await for packets. If the listening times out, the state goes to *Receive Packet*. In the event that a packet is successfully received, the packet's content is checked, else the state goes to *Receive Packet*. If the received packet's content is DONEDONE, the state goes to *Echo Statistics on Serial*. If the received packet's content matches the expected content, the state goes to *Good Packet*. Otherwise, the state goes to *Receive Packet*. Note that any packet with garbled content would have been dropped at the physical layer, and thus will not reach this state.
- Good Packet: This is where the information on both the received packet and the lost packets prior to the currently received packet are being computed and printed to serial. For more details on the computation, please refer to Fig. 5.10.
- Echo Statistics on Serial: The information on the lost packets prior to the last received packet are being computed and printed to serial. Next, the current run statistics (*packet received good*, *lost packet count*, and packet error rate) is computed and printed to serial. After that, the *run* counter is incremented. If *run* exceeds maximum value, the state goes to *Reset*, else the state goes to *Initialize Receive Operation*. Note that the hard-coded maximum value (10) in Fig. 5.7 sets the limit to the number of runs of the listening for the expected 1000-packet transmission.

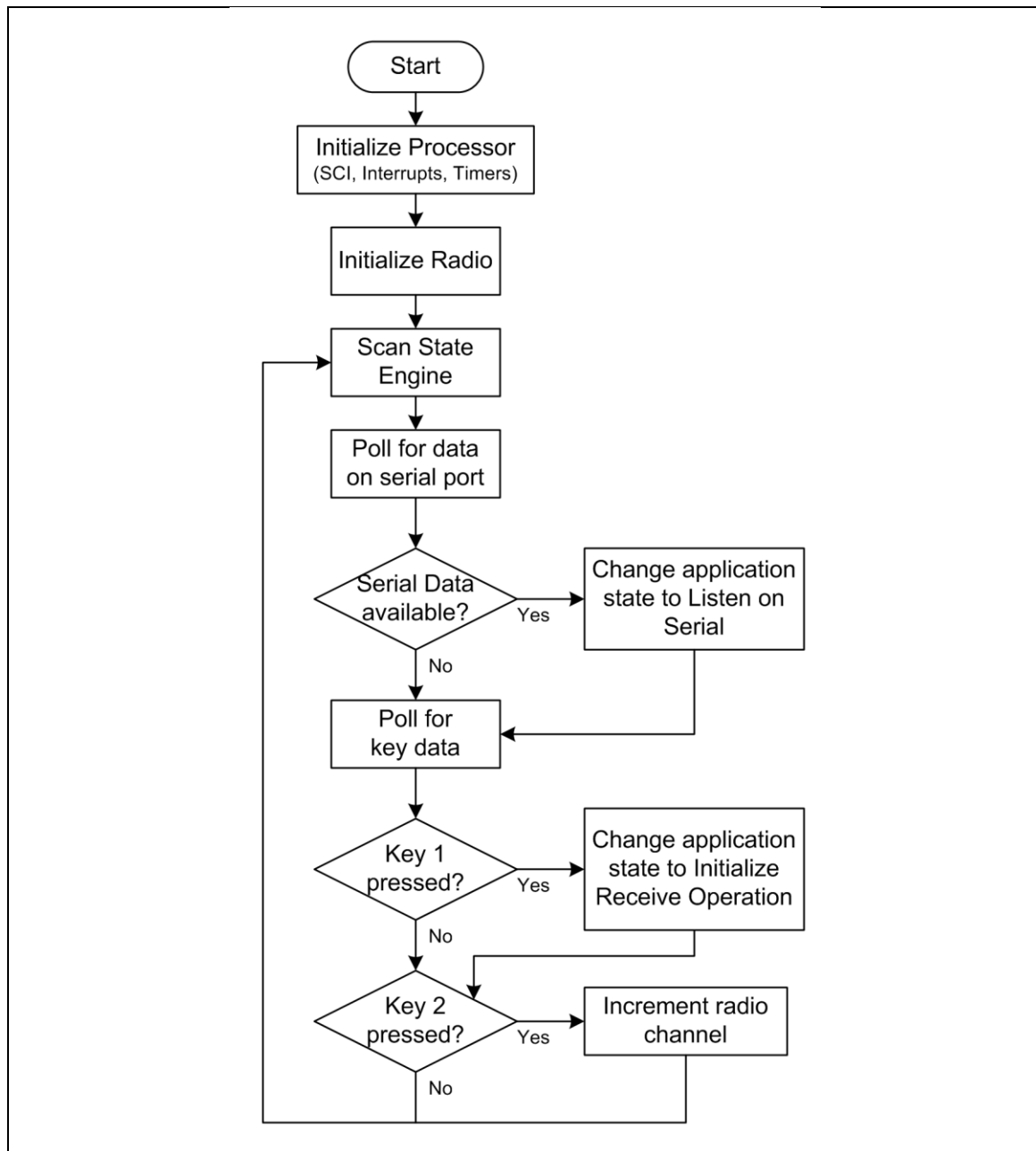


Fig. 5.8 Main Program Flowchart for Receiver Application.

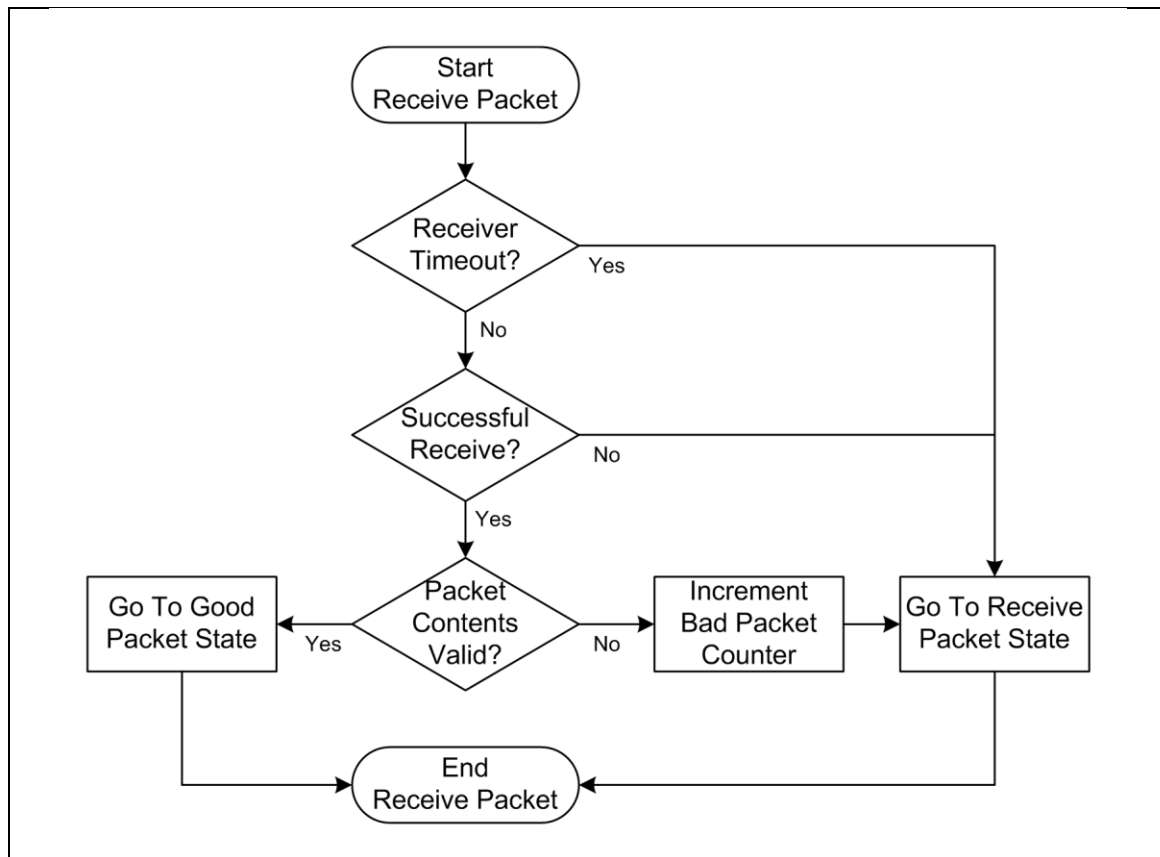


Fig. 5.9 Supplementary Flowchart for RECEIVE PACKET state part of Receiver Application.

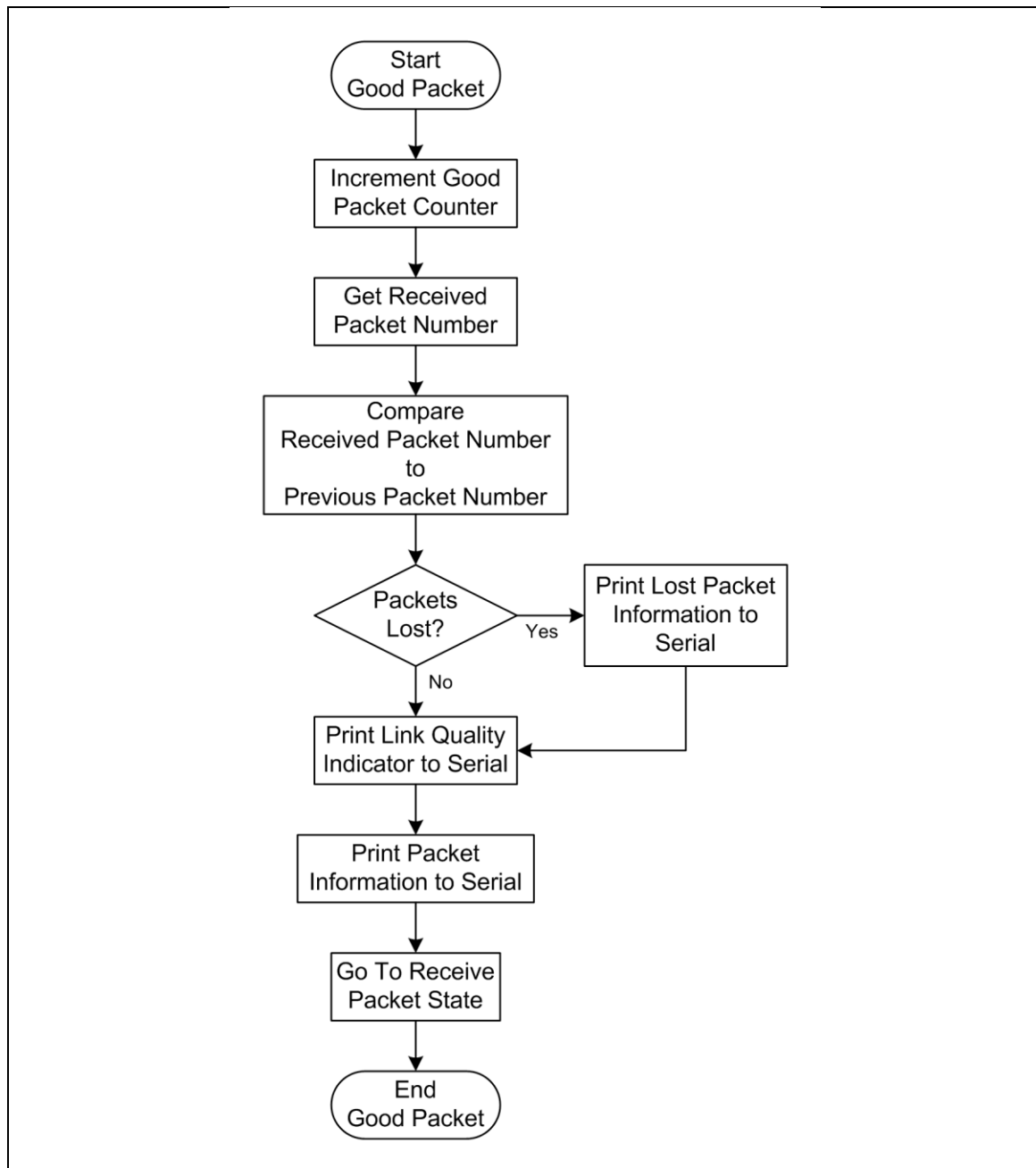


Fig. 5.10 Supplementary Flowchart for GOOD PACKET state part of Receiver Application.

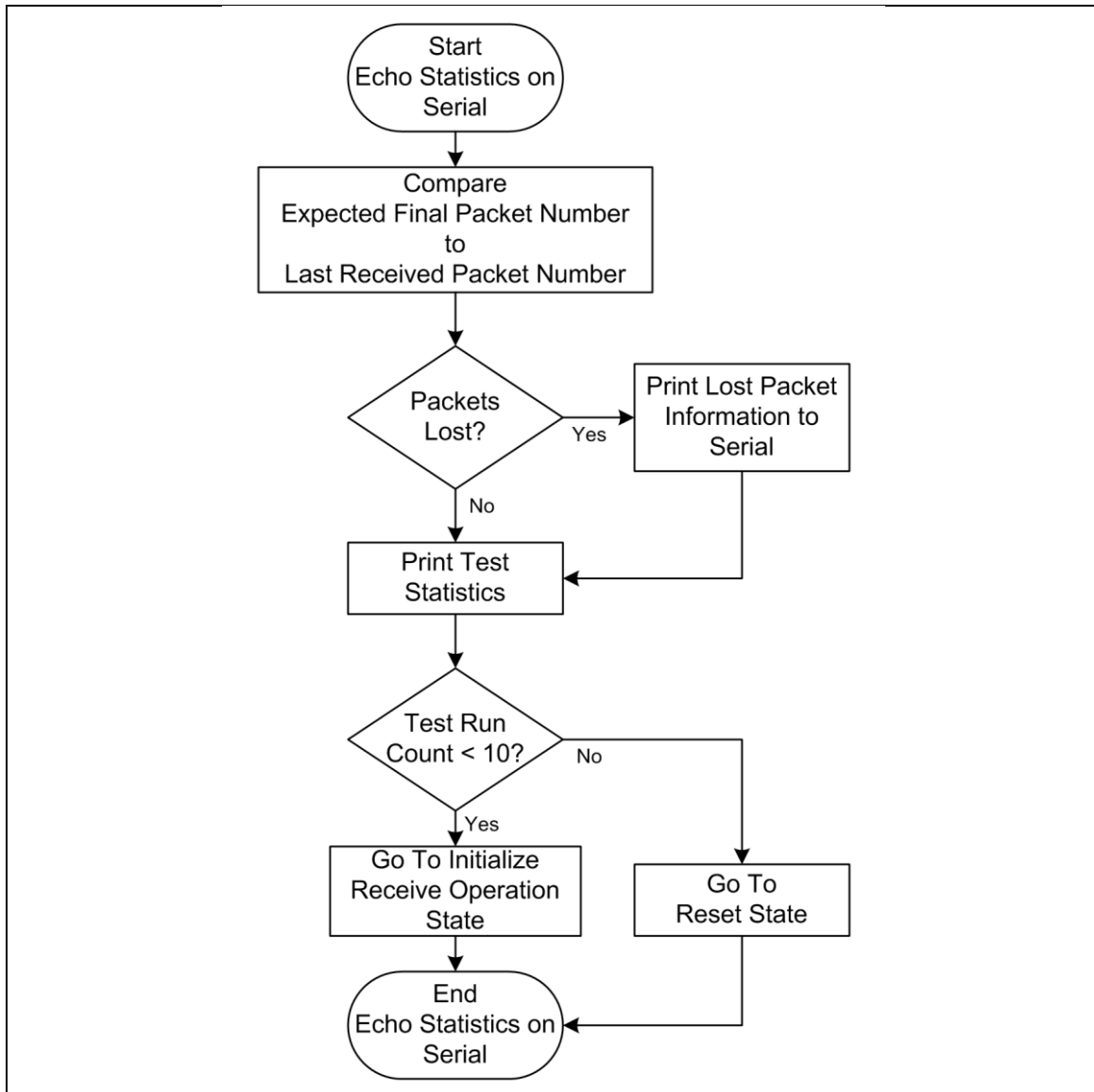


Fig. 5.11 Supplementary Flowchart for ECHO STATISTICS ON SERIAL state part of Receiver Application.

Furthermore, unlike the PER_TX program, the PER_RX program is much more complicated that the full functionality of the program could not be captured in a state diagram (Fig. 5.7) alone. Thus, additional flowcharts (Figs. 5.9, 5.10, and 5.11) have been presented to complete the program description.

PER Gathering Procedure

Procedure for gathering PER measurements:

1. Fasten a 6-ft coaxial cable with SMA-M connectors between the noise emulation circuit's end module (the attenuator) and the combiner's port 1. The output of the noise emulation circuit's AWG module should be a baseband monofractal noise signal with spectral exponent 0.4. This is due to the setting in preliminary procedure step (8-c-2): arbitrary noise = N1_0DOT4, amplitude = 50 mV_{pp} (minimum), and frequency = 2.5 kHz.
2. Attach an external antenna (i.e., the microstrip patch antenna) to the combiner's port 2 using a short coaxial cable with SMA-M connectors. Then, attach the combiner's SUM port to the receiver board's SMA RF connector using a short coaxial cable with SMA-M connectors. Note: The 13192-EVB board that acts as the receiver had been modified to use an external antenna instead of its default F-antenna. The modification involved extracting the resistor closest to the SMA RF connector on the board and then resoldering it after having turned it by 90° anti-clockwise.
3. Power up the USB hub, laptop, and SA, if they are not already switched on. Switch USB hub to the self-powered mode, if it is not already in that mode. The circuit for PER measurements must have been fully assembled as shown in Appendix C.8 (Fig. 1). Figures 5.13, 5.14, and 5.15 show snapshots of the assembled and working PER measurements circuit.
4. Launch HyperTerminal connection to the transmitter board COM and the receiver board COM (see procedure Preliminary step 11 for settings). In both screen, press 'S' once to reset board. Then press 'C' once to change channel to CH2 = 2410 MHz.
5. Place the transmitter at $D = D1$ away from the receiver. Pad all the metal strips in the chamber. Pad the transmit path in the chamber. Orient the transmitter board so that its built-in F-antenna is aligned (in the direction) and faces the receiver board's external antenna. Open

- the Excel file that was created in Preliminary step 2. On the first Excel sheet, fill in the distance value as the nodes separation used.
6. Turn chamber light off and close shut chamber door.
 7. Press 'R' once on the HyperTerminal screen for the receiver to have the receiver board to start listening. Press 'T' once on the HyperTerminal screen for the transmitter to have the transmitter board to perform 1 run of 1000 packets transmission.
 8. Observe PER. If nothing is received or too many PER recorded, adjust one of the antennas (whichever more convenient, in my case the receiver's external antenna). Repeat steps 7 and 8 until achieve an arrangement that gives the least PER (approximately 0%).
 9. For each of the input noise types on the opened Excel table, fill in the column on Noise Amplitude (mV_{pp} , mV_{rms}) while doing the following steps:
 - a. Do step 7.
 - b. Turn the AWG amplitude (AMPL) knob until packet starts to drop a lot.
 - c. Record the amplitude in mV_{pp} and in mV_{rms} . To toggle between mV_{rms} and mV_{pp} : Enter number->down arrow key and Enter number->up arrow key
 - d. Repeat steps a to c to fill in the amplitude that gives a targeted PER of 100, 80, 60, 40, 20, 10, and 0%, respectively.
 10. Perform PER measurements for the indicated noise amplitudes (7) and noise types (4) by following the steps below:
 - a. Select a noise type from AWG#1.
 - b. Referring to the Excel table from step 9d, set the AWG#1's front panel to one of the amplitude (mV_{pp}) values for that noise type.
 - c. Repeat step 7 three times. For each time, the output from the receiver's HyperTerminal session is saved and the filename is written in the Excel sheet.
-

- d. Repeat steps b to c for a different amplitude value. Do this until all the amplitude values corresponding to the targeted PER values (0, 10, 20, 40, 60, 80, and 100%) have its corresponding PER data.
 - e. Repeat steps a to d for a different noise type. Do this until there are PER data for all four type of noise (with $\beta = 0.4, 0.8, 1.2,$ and $1.6,$ respectively).
11. Perform PER measurement for ZigBee spectrum (i.e. ZigBee transmission alone). Do this by unplugging noise cable (and aim its end towards a RAM absorber) and plugging a 50Ω terminator into the combiner's port 1. Close the chamber door and run the PER program twice so that there would be two PER readings. Each reading consists of the PER values for ten 1000-packet transmission.
 12. Perform trace capture for the ZigBee transmission without noise using section 5.1.5.3 procedure for A. Then, perform trace capture for the indicated noise amplitudes (7) and noise types (4) using section 5.1.5.3 procedure for B.
 13. Repeat steps 2 to 12 for another nodes separation, D. Do this until all the different nodes separations ($D1 = 1$ metre, $D2 = 2$ metre, and $D3 = 4$ metres) have been covered.

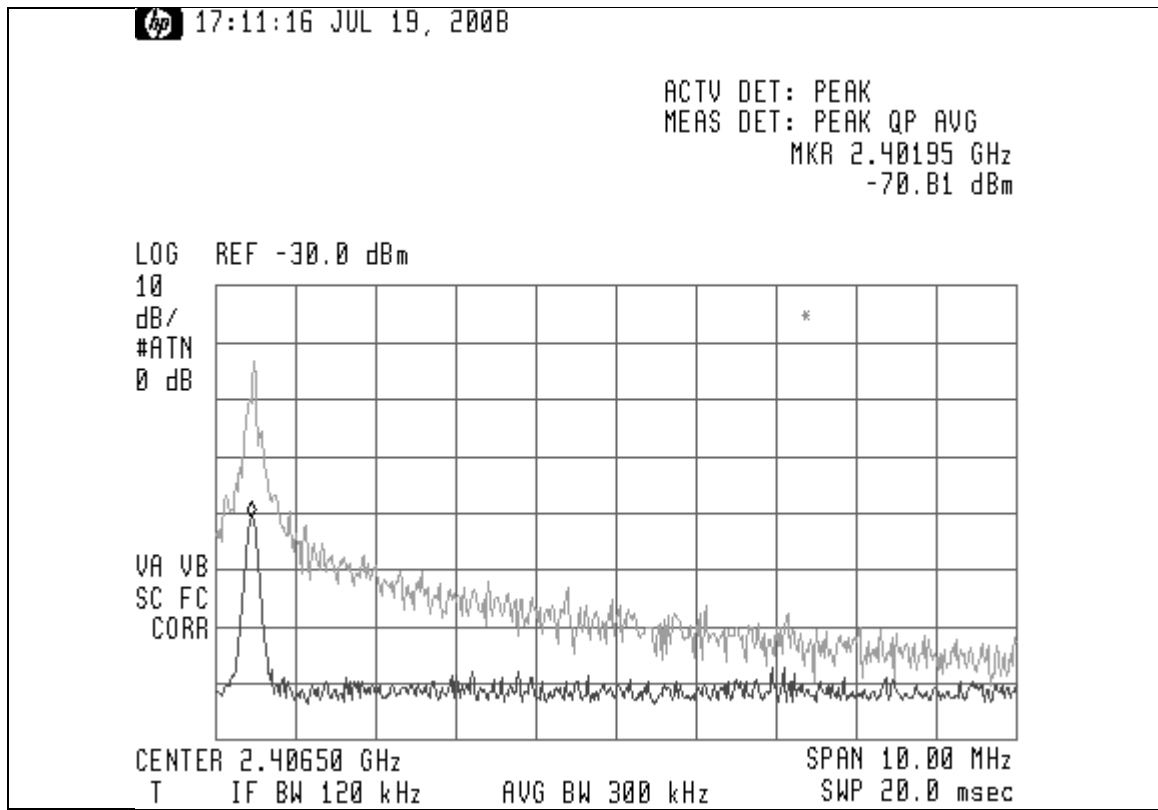


Fig. 5.12 A HP 8546 spectrum analyzer screenshot showing two traces, 1) the spike of the 2.402 GHz PLL-generated carrier signal (in black shade), and 2) the spectrum of the modulated monofractal noise signal (in gray shade) originating from the baseband noise signal generated by AWG with configuration of 1.050 V_{pp} and using arbitrary waveform of spectral exponent 1.6 to induce 20% PER.

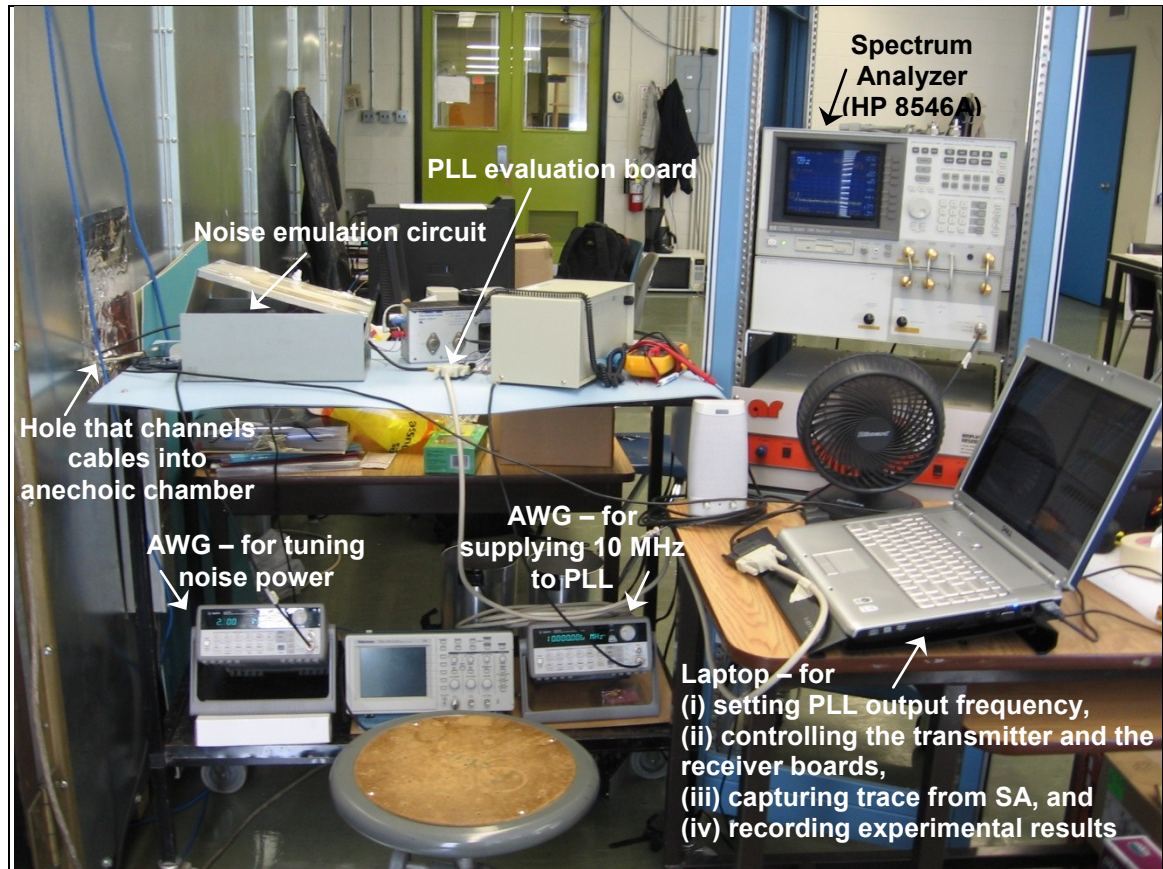


Fig. 5.13 The components of the performance evaluation under synthetic noise setup that reside outside of the anechoic chamber.

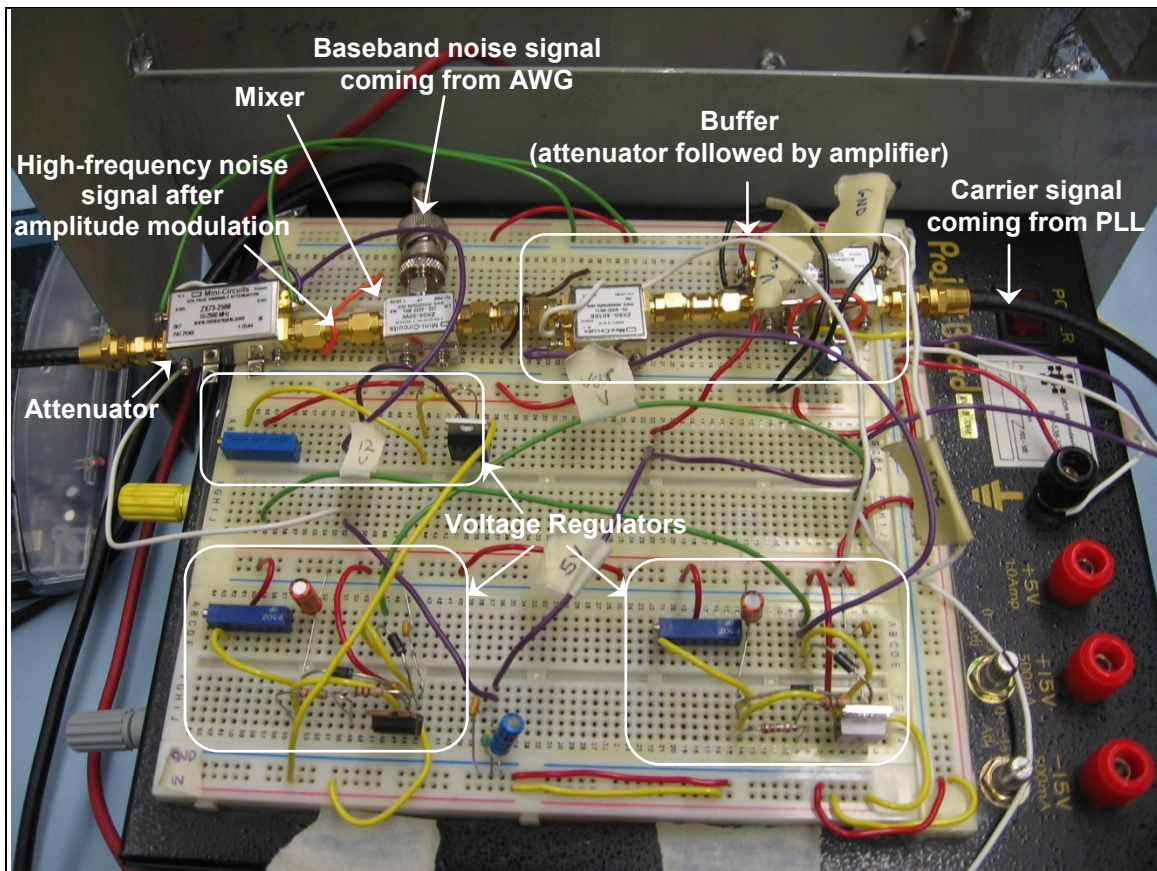


Fig. 5.14 The fully-assembled noise emulation circuit for generating high-frequency, analog fractal-based noise.

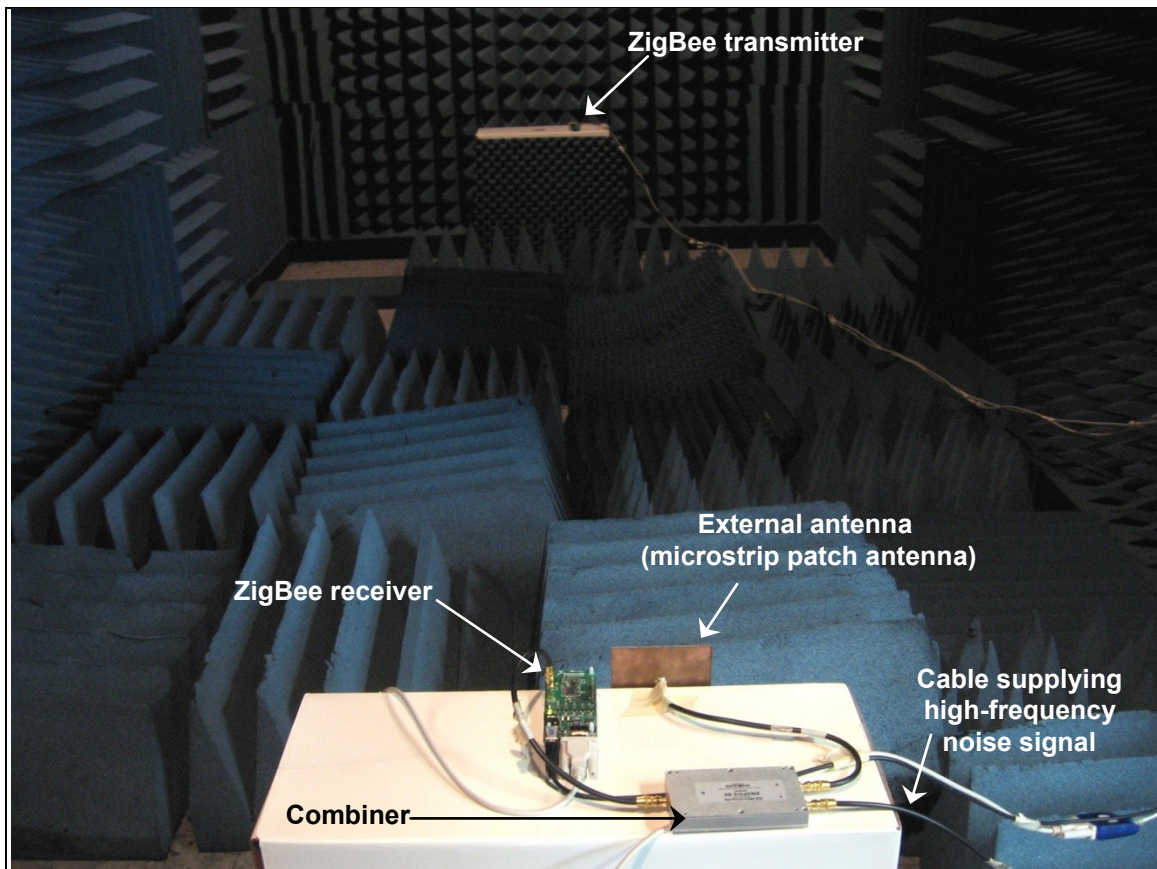


Fig. 5.15 The components of the performance evaluation under synthetic noise setup that reside inside the anechoic chamber.

5.1.5.3 SNR Measurements

Trace Data Acquisition Software

The software for saving the display from HP 8546A spectrum analyzer as a trace file is not available. As such, I have written a graphical user interface in C# that extracts the trace of the spectrum displayed on HP 8546A with a USB-GPIB controller module via the spectrum analyzer's GPIB interface and stores the trace information in terms of pairs of amplitude (in dBm) value and its corresponding frequency (in Hz) value in a text file. The pseudocode for my trace data acquisition software is presented in Fig. 5.16. The important events handled by the trace data acquisition GUI program are the init event, the connect event, the capture event, the

disconnect event, and the exit event. Please cross reference Appendix A.3 for the actual code written in C# and Appendix C.7 for the documentation on using the software.

```

<Init Event>
CREATE a new serial port object
GET a list of available COM ports
SHOW the available COM ports in a drop down list on the GUI
// INITIALIZE the GUI elements
    CLEAR GUIStatus
    ENABLE the Connect button
    DISABLE the Disconnect and the CaptureTrace buttons, and the CaptureOneTime,
CaptureTenTimes, CaptureTwentyTimes check boxes
SET the default pathname or directory for storing trace data

<Connect Event>
GET the user selected COM port number
SET selectedPort to the user selected COM port number
// INITIALIZE the serial port object
    SET PortName to selectedPort
    SET BaudRate to 115200
    SET Parity to None
    SET DataBits to 8
    SET StopBits to 1
SET RTS/CTS handshaking
SET Error handling
SET the read and the write timeouts
OPEN the serial port
// UPDATE the GUI elements
    CLEAR GUIStatus
    SET GUIStatus to "Connected" to show port connection has been established
between pc and instrument
    ENABLE the Disconnect and the CaptureTrace buttons, and the CaptureOneTime,
CaptureTenTimes, CaptureTwentyTimes check boxes
    DISABLE the Connect button
// CONFIGURE the GPIB address of the GPIB_USB module to that of the instrument being
controlled
    GET the instrument GPIB address, currAddr
    SEND the GPIB command with the value denoted by currAddr appended to the string
'++addr ' to the GPIB-USB module

<CaptureTrace Event>
SET numSnapshots to 0
SET amplitude unit, trace data format, and enable single sweep
GET center frequency, span, start frequency, stop frequency, and reference level
SET centerFreq, spanFreq, startFreq, stopFreq, and refLevel
// UPDATE the GUI elements
    CLEAR GUIStatus
    SET GUIStatus to "Busy: Capturing trace ..." // to show trace capture has
started
    ENABLE the Disconnect button
    DISABLE the Connect button and the CaptureTrace button
IF CaptureOneTime is checked THEN
    SET numSnapshots to 1
ELSE-IF CaptureTenTimes is checked THEN
    SET numSnapshots to 10
ELSE // for the case when CaptureTwentyTimes is checked
    SET numSnapshots to 20
END IF
FOR each numSnapshots
    CALL processFile function // to process incoming data
END FOR
SEND the GPIB command '++loc' to return control back to front panel of instrument

```

```

// UPDATE the GUI elements
    CLEAR GUIStatus
    SET GUIStatus to "Idle: Completed trace capture" to show trace capture has ended
    ENABLE the Disconnect and the CaptureTrace buttons
    DISABLE the Connect button
    UNCHECK the CaptureOneTime, CaptureTenTimes, CaptureTwentyTimes check boxes

<Disconnect Event>
CLOSE the serial port
// UPDATE the GUI elements
    CLEAR GUIStatus
    SET GUIStatus to "Disconnected" to show port connection has been ended between
pc and instrument
    ENABLE the Connect button
    DISABLE the Disconnect and the CaptureTrace buttons, and the CaptureOneTime,
CaptureTenTimes, CaptureTwentyTimes check boxes

<Exit Event>
CLOSE the serial port
CLOSE the GUI window

<ProcessFile Function>
GET filename
IF filename exists THEN
    SHOW file error message
    RETURN
END IF
OPEN the file
GET incoming trace data
// FORMAT data before writing
    COUNT the number of amplitudes returned from the comma-separated incoming trace
data
    FOR each amplitude
        COMPUTE the frequency corresponding to the amplitude
        APPEND the computed frequency to the amplitude value separated with a
comma
        WRITE formatted string to file
    END FOR
CLOSE the file

```

Fig. 5.16 Pseudocode of the trace data acquisition GUI software.

Trace Data Capture Procedure

There are two types of trace data to capture, namely, A) ZigBee spectrum and noise floor, and B) the spectrum of each input noise type's respective noise levels. The transmission of ZigBee packets appear on the SA (model HP 8546A) as random spectrum spikes that occur intermittently across the set frequency span. Thus, to determine the shape and amount of frequencies taken by the packets transmission, the MAX HOLD (pg. 6-16 of [Agil94]) feature of the SA is used. For consistency, MAX HOLD is used also for capturing the trace of the input noise, despite its

spectrum being of stable appearance on the SA. Using MAX HOLD enables the capture of the maximum envelope of the signal-under-test. Thus, the captured spectra (ZigBee and noise) would contribute towards the worst case SNR computation. Note that when handling sensitive equipment and/or circuits, such as the EMI Receiver (HP 8546A), the noise emulation circuit, and the AWG (33120A), ESD protection wristband is worn by the author at all times.

For A: ZigBee spectrum with Max Hold

- (1) Unplug noise injection cable from combiner's port 1 (and aim its end towards a RAM absorber) and fasten the 50 Ω terminator module on the combiner's port 1. See Fig. 5.17.
- (2) SA is powered on and set to these configurations (SP, FA, FB, ATN, REF, IF, AVG BW, and SP values as stated in Appendix D). Connect cable from SA to the combiner's output i.e., the (S)um port. Press 'Trace' on SA.
- (3) Launch HyperTerminal connection to the transmitter board. Press 'S' once to software reset the state and the counters of the transmitter application. Then, press 'C' once to change to channel 2 = 2410 MHz.
- (4) Launch Data Acquisition GUI. Select COM port and click 'Connect'.
- (5) On Data Acquisition GUI, type in the respective filename. Select 'Once'. Fill in this filename on the respective row on the Trace Data column in the opened Excel sheet.

- (6) On HyperTerminal connection to the transmitter board, press 'T' and at the same time, on the SA, press 'MAX HOLD A'. The HyperTerminal screen will say:

```
Run #:1
The TX radio is now transmitting...on channel 2...
```

- (7) Watch the screen of the HyperTerminal connection to the transmitter board. When it says “End of transmission”, on the SA, quickly press 'VIEW A'. Doing this stores the current state of MaxHold values to Trace A to ensure that no signals after the ZigBee transmission had ended will be able to alter the values in Trace A.
- (8) Then, on the Data Acquisition GUI, press 'Capture'. After that, on the SA, press 'CLEAR WRITE A'. One of the captured traces is shown in Fig. 5.21.
- (9) Repeat steps (5) to (8) for nine times to get altogether 10 captures of ZigBee spectrum (with highest amplitude values).

For B: Noise spectrum with Max Hold

- (1) Unfasten the 50 Ω terminator module from combiner’s port 1 and plug in noise injection cable to combiner’s port 1. Then, unplug cable attached to the receiver’s external antenna from combiner’s port 2 (and leave it as it is on the box) and fasten the 50 Ω terminator module on the combiner’s port 2. See Fig. 5.18.
- (2) SA is powered on and set to these configurations (SP, FA, FB, ATN, REF, IF, AVG BW, and SP values as stated in Appendix D). Connect cable from SA to the combiner’s output i.e., the (S)um port. Press 'Trace' on SA.
- (3) Noise circuit is powered on. PLL is set to lock onto 2402 MHz (when chosen ZigBee channel is with center frequency 2410 MHz).
- (4) Launch Data Acquisition GUI. Select COM port and click 'Connect'.

- (5) On Data Acquisition GUI, type in the respective filename. Select 'Once'. Fill in this filename on the respective row on the Trace Data column in the opened Excel sheet.
- (6) On AWG, set noise to one of the input noise types. Then set the AWG amplitude to one of the noise amplitude levels that was recorded in the Excel sheet from step 9 of PER measurements procedure.
- (7) On the SA, press 'MAX HOLD A' and watch the SA screen. When there is least movement in terms of displayed spectrum, press 'VIEW A'.
- (8) Then, on the Data Acquisition GUI, press 'Capture'. When GUI status says "Idle: Completed trace capture", on the SA, press 'CLEAR WRITE A'. One of the captured traces is shown in Fig. 5.19.
- (9) Repeat steps (5) to (8) for two times to get altogether 3 captures of noise spectrum (with highest amplitude values i.e., upper envelope of noise fuzzy PSD).
- (10) Repeat steps (5) to (9) for another noise amplitude level. Do this until all noise amplitude levels have been covered.
- (11) Repeat steps (5) to (10) for another input noise type. Do this until all noise types have been covered.

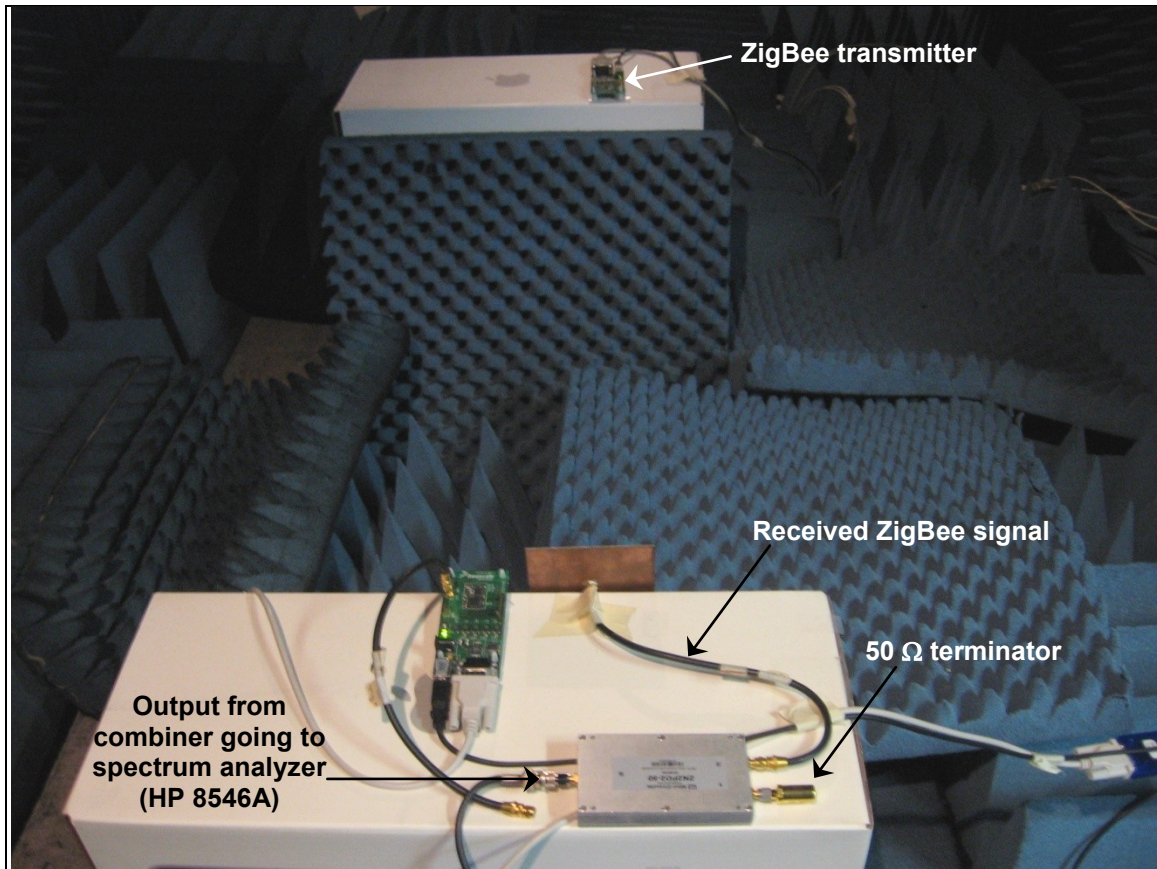


Fig. 5.17 Specific setup at the combiner module for enabling the capturing of ZigBee spectrum with spectrum analyzer (HP 8546A) and trace data acquisition GUI.

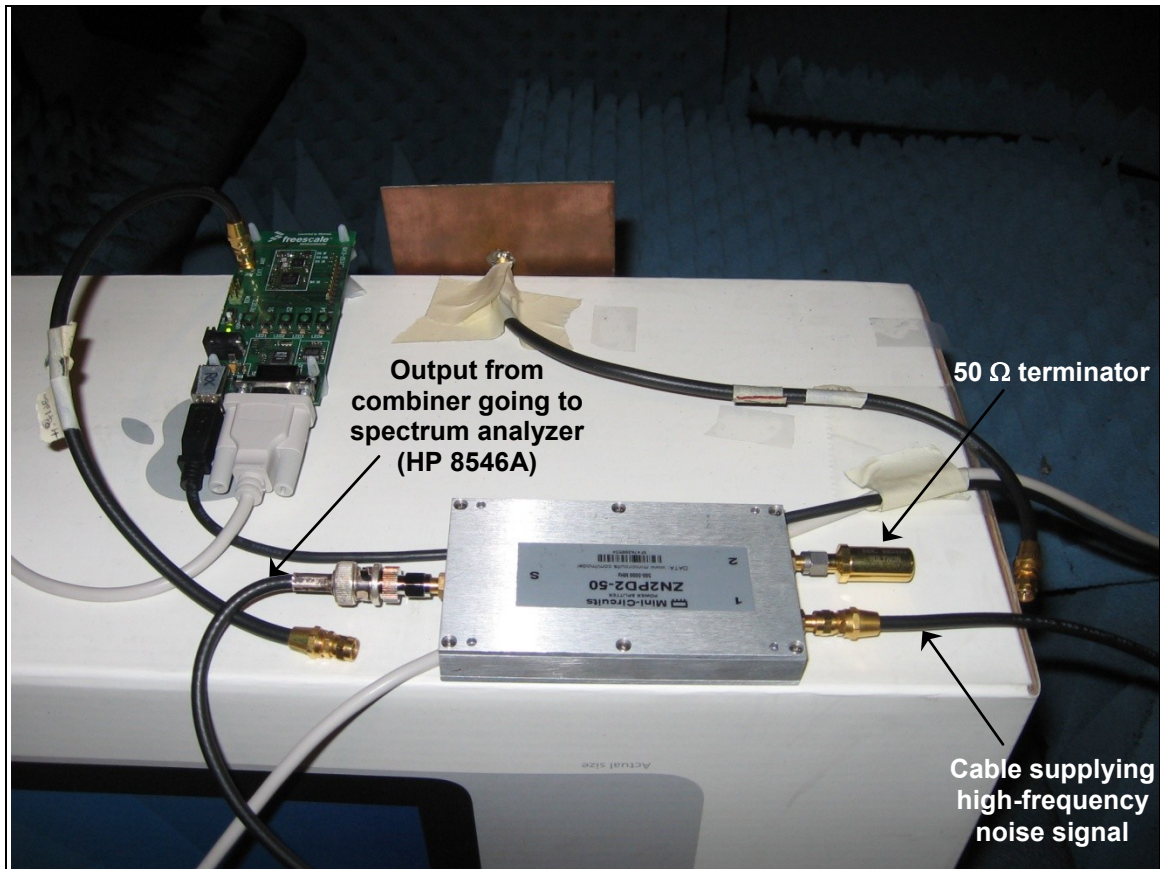


Fig. 5.18 Specific setup at the combiner module for enabling the capturing of input noise spectrum with spectrum analyzer (HP 8546A) and trace data acquisition GUI.

Compute Power Measurements for SNR Computation

The captured traces of ZigBee signal and noise signal will first need to be processed to measure their signal power before SNR calculation can be performed. Please see Appendix A.4 for the MATLAB code (`computePowerBW.m`) that I written to calculate the total power of the captured trace between the frequencies 2.409 and 2.411 GHz. This specific 2 MHz range is the 6-dB bandwidth of the ZigBee spectrum where ZigBee experienced packet loss due to the presence of noise.

The type of spectrum represented by the trace data captured from HP 8546A affects the calculation of total power within a specified bandwidth. With respect to the spectrum analyzer

display, the amplitude axis gives the signal level (in terms of power in units of dBm) at each frequency component of the signal under test (sut). If the amplitudes denote the Fourier spectrum (i.e., the magnitude of the Fourier coefficients) of the sut, a factor of two is to be multiplied to the amplitudes of the captured trace. If the amplitudes denote the power spectrum of the sut, then a factor of 2 is not required. For computation of SNR, whether or not there is a factor of 2 is not crucial anyway because the factor of 2 if present will be cancelled out. Thus, the calculation for signal power within a specified bandwidth has been taken as the sum of the linear measured amplitudes (in terms of power in units of Watts) across the specified interval (in my case, the interval of interest is the 2 MHz frequency range relating to the 6-dB bandwidth of the ZigBee signal centered at its channel's center frequency). Also, the discrepancy between the measured and the expected amplitudes will not matter when it comes to SNR computation because both the ZigBee signal and the noise signal will have the same discrepancy offset and the ratio is maintained. Examples of the computed power within the specified 2 MHz bandwidth for the trace of a noise signal trace and the trace of a ZigBee signal trace can be found in Figs. 5.19 and 5.21, respectively. In terms of accuracy of the captured trace with that displayed on the HP 8546A spectrum analyzer, a visual verification can be performed by comparing the amplitudes and their corresponding frequencies for the trace of the modulated signal in Fig. 5.19 and the trace of the carrier signal in Fig. 5.20 to the actual spectrum analyzer screenshot in Fig. 5.12.

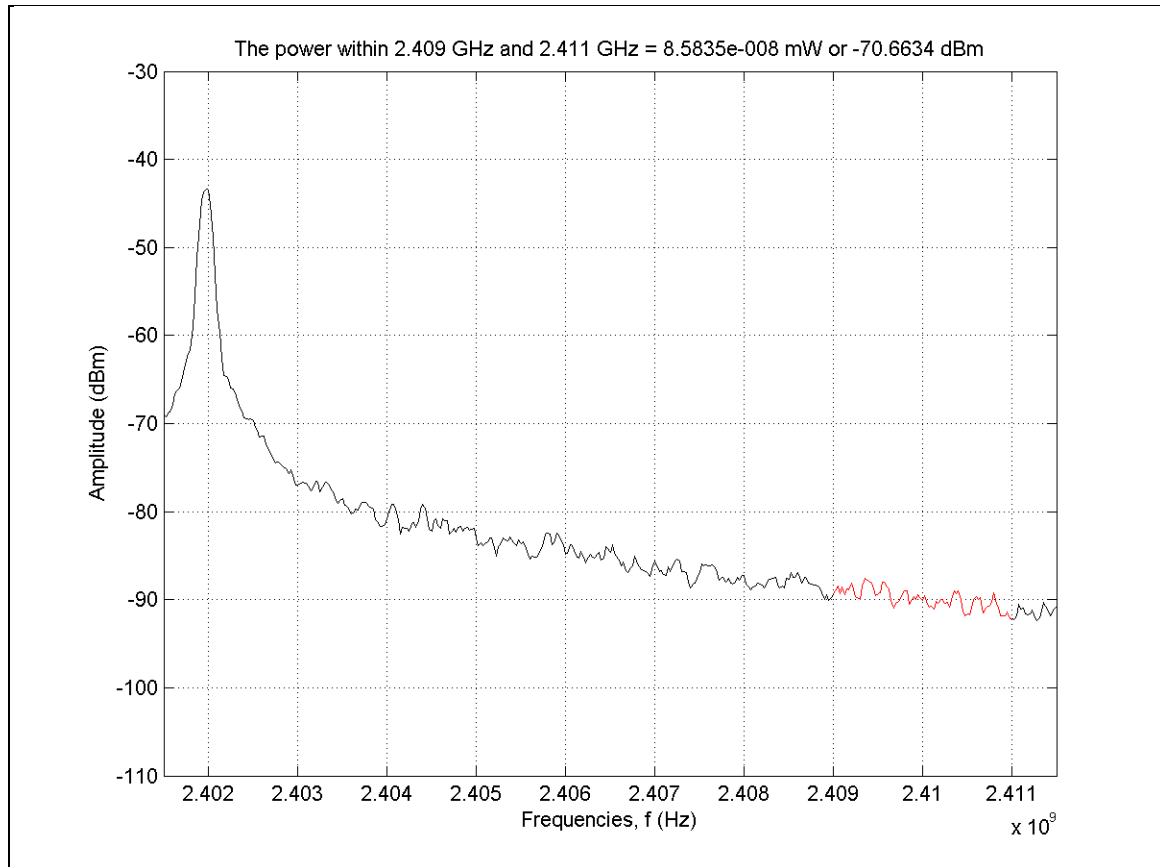


Fig. 5.19 A plot of the noise spectrum of one of the input noise signals captured by spectrum analyzer (HP 8546A) and trace data acquisition GUI software and with power within the 2 MHz frequency of interest (in red) calculated by compPowerBW.m and plotted by main_program_genpowerBW.m.

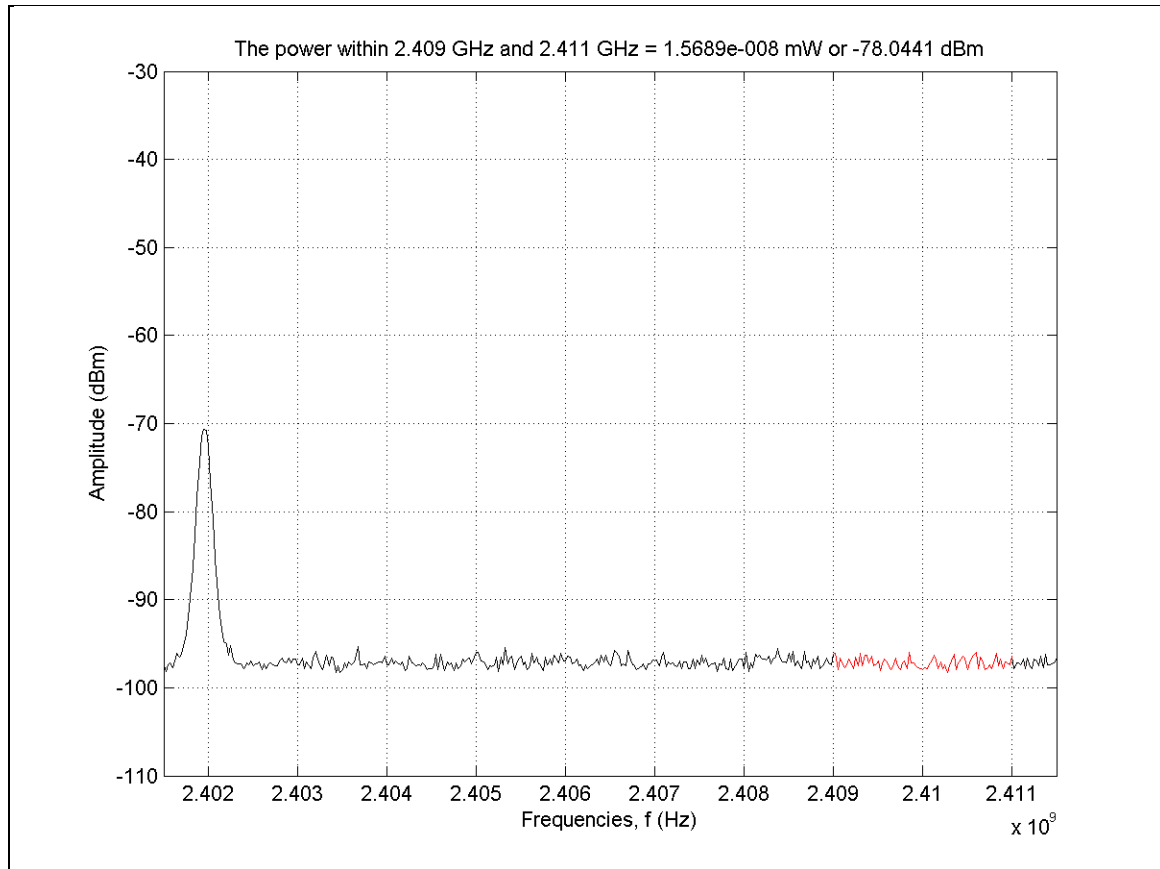


Fig. 5.20 A plot of the carrier signal spectrum (a spike) captured by spectrum analyzer (HP 8546A) and trace data acquisition GUI software.

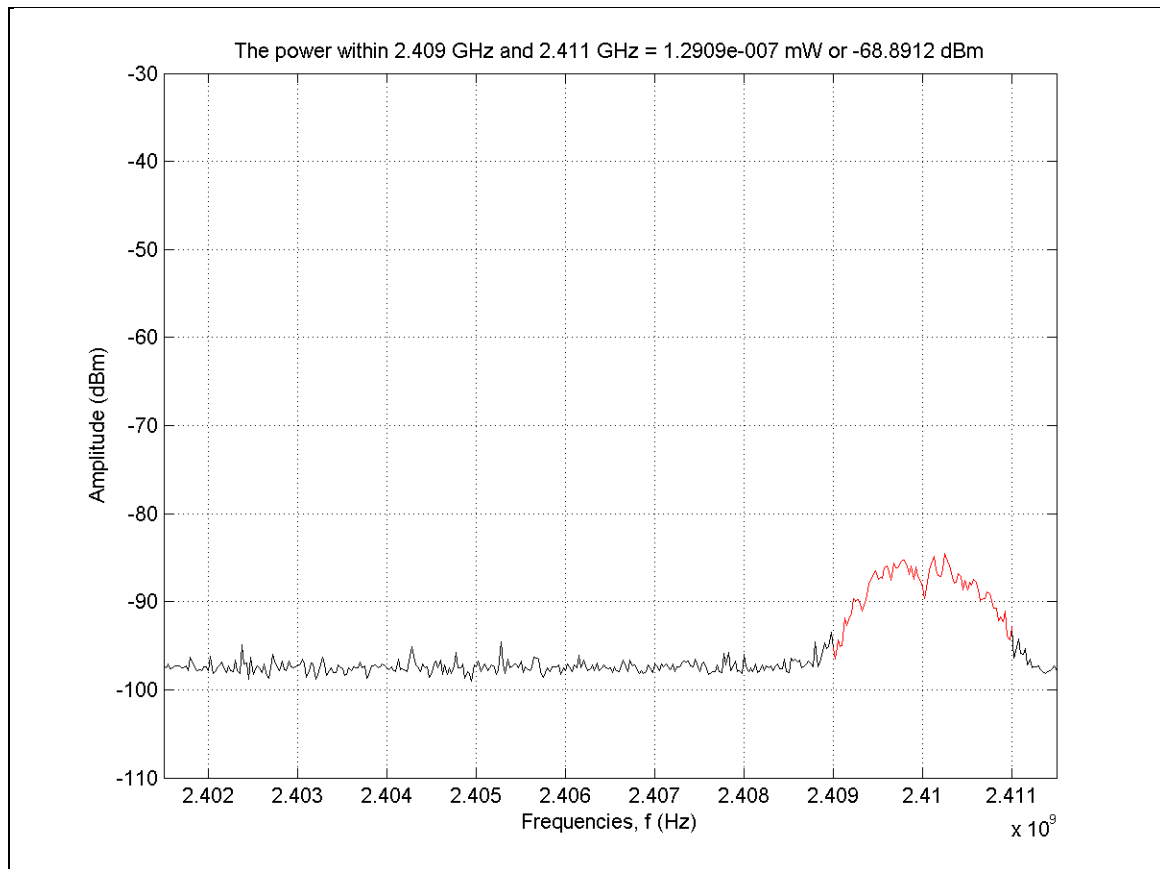


Fig. 5.21 A plot of the ZigBee spectrum captured by spectrum analyzer (HP 8546A) and trace data acquisition GUI software and with its power within the 2 MHz frequency of interest (in red) calculated by `compPowerBW.m` and plotted by `main_program_genpowerBW.m`.

5.2 Experiment: Performance Evaluation under Physical Noise

5.2.1 Objective, Methodology, Outcomes, Benefits, Assumptions

The goal of this experiment is to evaluate the performance of ZigBee in physical environment. In other words, the experiment is done to check the impact of physical broadband noise (instead of synthetic noise) upon ZigBee communication. Due to resource constraints, the experiment was conducted by bringing in a physical noise source into the test environment (where the communication network is setup in anechoic chamber), instead of bringing the setup to a physical environment. The advantage of this is isolation of noise source (i.e., no other unknown noise sources will be present to influence the experimental result (i.e., the PER

measurements). The physical noise source used is a plasma globe because its operation (spark discharges) can emit broadband electromagnetic radiation that may be sufficient to affect the operation of nearby electrical devices.

5.2.2 System Description

Fig. 5.22 shows a 2-node network that can be brought to any site (say the operating environment of an excavator) and allow for PER measurements to be gathered to check the wireless protocol performance under the influence of physical noise specific to the environment of the target application, no longer injected noise. There will not be any SNR measurements as the physical noise is emanated and not injected.

For ease of accessibility reasons, instead of going out to a site, I brought in physical noise sources, such as Jacob's ladder and plasma globe, which are devices that generate sparks to simulate broadband interference.

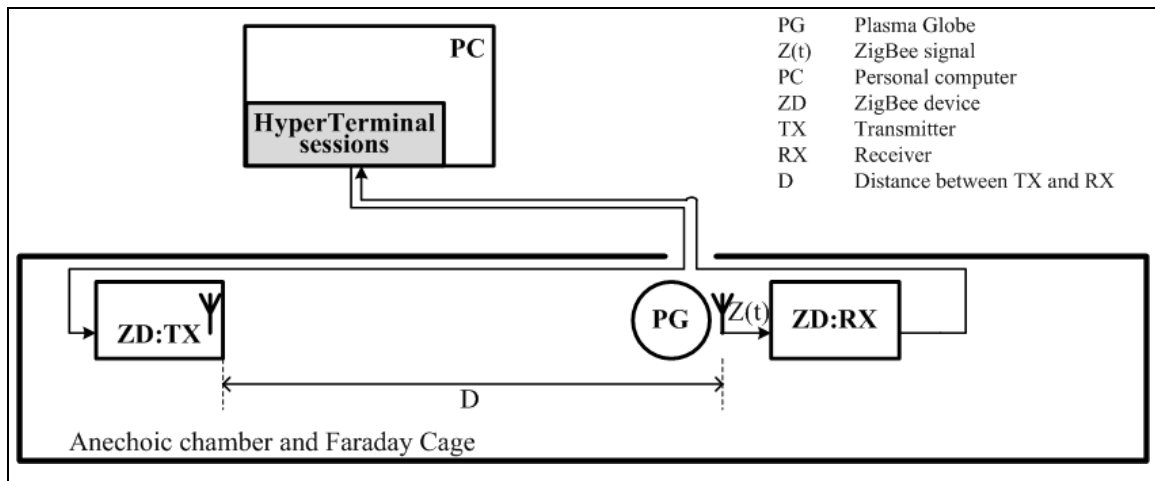


Fig. 5.22 ZigBee-physical noise performance evaluation setup for PER measurements.

5.2.3 Materials

The physical noise source could be a Tesla coil, Jacob's ladder, plasma lamp (also known as a plasma globe), Wimshurst machine, or Van de Graaff generator. A plasma globe (powered from USB) was used in my experiment.

5.2.4 Procedure

The setup (Fig. 5.22) is deliberately designed in a way to portray the worst case scenario by having:

- (i) the physical noise source placed closest to the receiving antenna (in front of the receiving antenna but apart by 15 inches), and
- (ii) the ZigBee signal is the weakest since the farthest node separation was used, i.e., $D = 4$ metres.

Apart from that, the constants in the setup are that the Wi-Fi setting on the laptop is turned off, the USB hub and the power bar for powering the plasma globe reside inside the anechoic chamber but are carefully placed underneath the RAMs, and the anechoic chamber door is closed throughout the entire PER measurements experiment.

PER measurements were gathered for the following three cases:

- Case 1: Perform ZigBee packets transmission without the presence of plasma globe, and with power bar unplugged from mains, as shown by a screenshot in Fig. 5.23;
- Case 2: Perform ZigBee packets transmission with the presence of plasma globe located on a chair, and with USB hub attached but power bar unplugged from mains (i.e., the physical noise source is at OFF state); and
- Case 3: Perform ZigBee packets transmission with the presence of plasma globe located on a chair, and with USB hub attached and power bar plugged in to mains (i.e., the physical noise source is at ON state), as shown by a screenshot in Fig. 5.24.

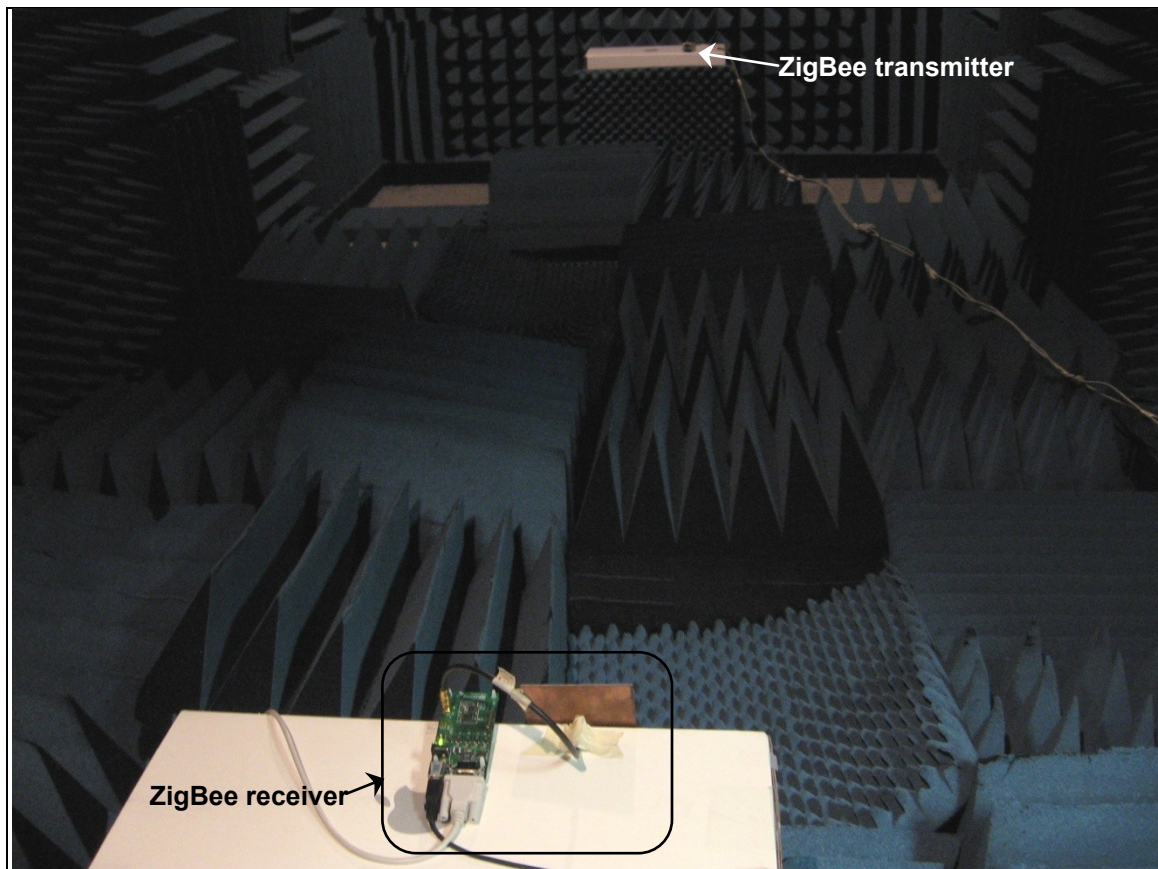


Fig. 5.23 The case 1 scenario where PER measurements are gathered for ZigBee transmissions without physical noise being present.

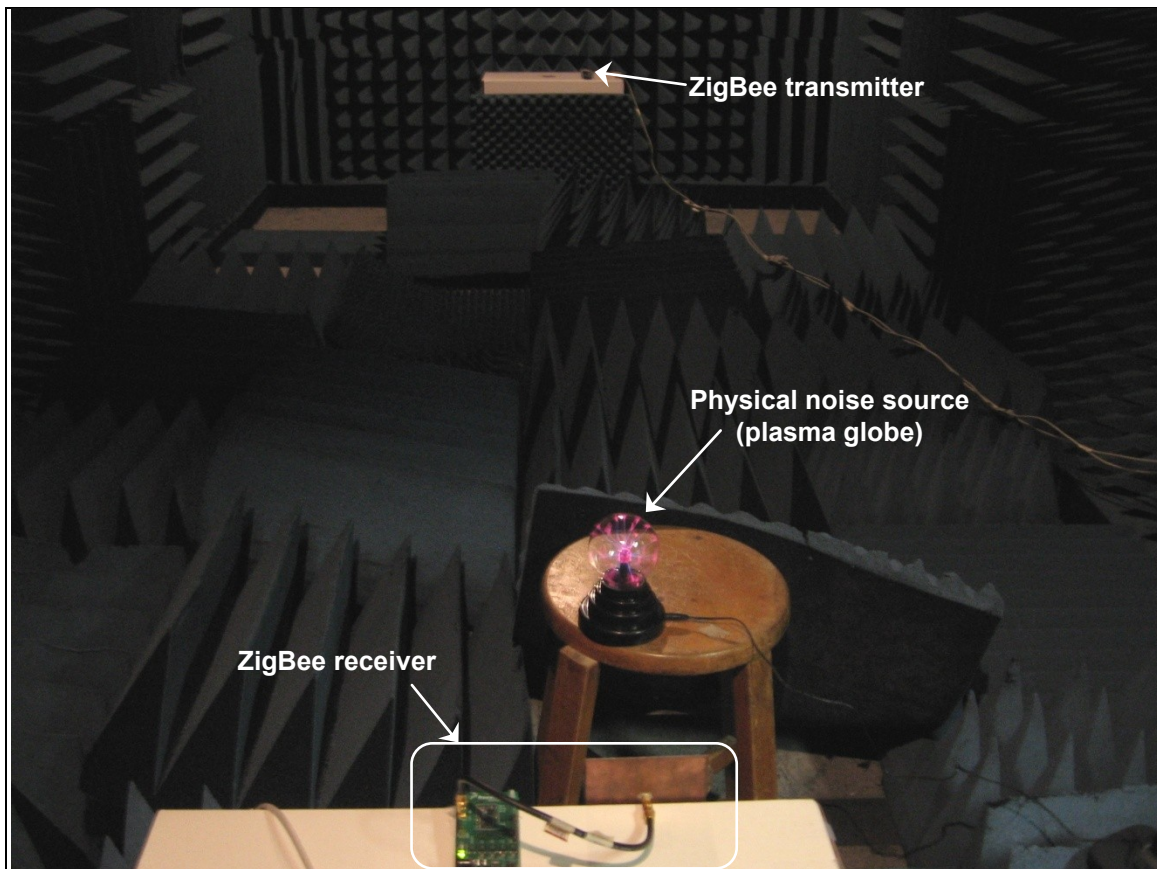


Fig. 5.24 The case 3 scenario where PER measurements are gathered for ZigBee transmissions with physical noise (plasma globe switched on) being present.

5.3 Chapter Summary

This chapter has presented a description covering both the hardware aspect and the software aspect of the design for the performance evaluation under synthetic noise experiment. Also included are snapshots of the implemented experimental setup that produced the results discussed in the next chapter, and identification of problems and implemented solutions that led to the final design of the setup. Finally, a description of a verification experiment using the same communication network but this time the evaluation is to see the impact with an actual physical noise source, has also been presented.

Chapter 6

EXPERIMENTAL RESULTS AND DISCUSSION

This section presents the results and discusses the observations from the results in the manner as presented in Fig. 6.1.

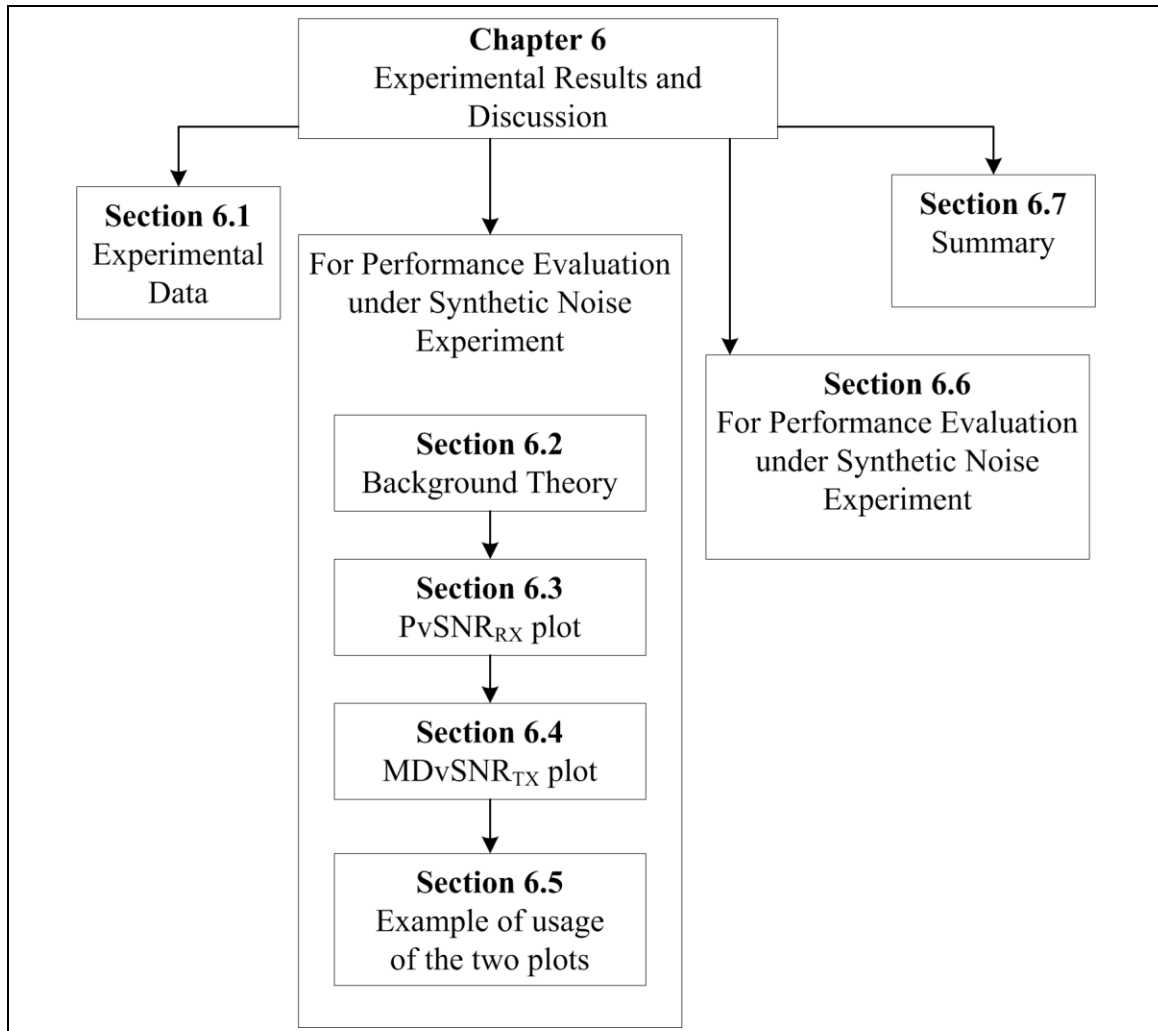


Fig. 6.1 Layout of Chapter 6.

6.1 Experimental Data

The parameters encapsulated by the gathered data are as follows:

- (i) Packet error rate, PER
- (ii) Distance between nodes separation (i.e., between the transmitter device and receiver device), D
- (iii) Power of the signal arrived at the receiver (i.e., the received signal power), P_{S_RX}
- (iv) Power of the input noise (i.e., of the noise added to P_{S_RX}), P_N
- (v) The input noise type (in terms of the spectral exponent of the baseband monofractal noise used to create the high-frequency analog noise signal), β

The experimental data that I collected from my performance evaluation experiments (Chapter 5) are available in the attached CD-ROM in the Appendix section. For the PER data over different nodes separation, please refer to Appendix B.3. For the traces of ZigBee signal and noise signal, respectively, please refer to Appendix B.4.

6.2 Background Theory

This section provides the concepts in the theory of radio frequency propagation that were utilized within this chapter in terms of their definition and along with how they are used in the thesis.

6.2.1 On Near-Field Effects

For normal propagation physics to apply, the receiving antenna must be in the far field (i.e., the receiving antenna must be far apart from the transmitting antenna such that the wave is planar). The following formula [pp. 58-59, Morr04] is used to calculate minimum far-field distance, d (in metres):

$$d > \frac{2D^2}{\lambda}$$

where D is the longest antenna dimension (in metres), the carrier wavelength, $\lambda = c/f$, the propagation speed, $c = 3 \times 10^8$ m/s (in general) but to be more accurate, c is affected by permittivity of material that antenna is made of; and f is the frequency in Hertz.

Since $D = 5.5$ cm = 0.055 metres (measured to be the widest length of my antenna), and $f = 2410$ MHz, the far-field of my receiving antenna (microstrip patch antenna) is or begins at 0.0486 m (roughly at 5 cm apart from the transmitting antenna (F-antenna). So, for my experiments, using distances such as 1, 2, and 4 metres should be good in the sense that I do not have to be concerned with the results of the experiments being influenced by near-field effects.

6.2.2 Inverse-Square Law (ISL)

With regards to received signal strength, the inverse-square law is used. The law states that the strength or intensity of signal varies inversely with the square of the distance between the transmitting and the receiving antenna. [pp. 60, Morr04].

This law is utilized in the Friis equation and the free-space path loss equation (as shown in the next sub-section).

6.2.3 Friis Equation and Free-Space Path Loss (FSPL)

From ISL, $I \propto \frac{1}{d^2}$ and $I = \frac{P}{A}$

$$P_{S_{RX}} = I_d \times A_{RX} \text{ where } I_d = \frac{P_{S_{TX}} G_{TX}}{4\pi d^2} \text{ and } A_{RX} = \frac{G_{RX} \lambda^n}{4\pi}$$

$$P_{S_{RX}} = \frac{P_{S_{TX}} G_{TX} G_{RX} \lambda^n}{(4\pi d)^2} \text{ where } d > \frac{2D^2}{\lambda} \text{ and the wavelength, } \lambda = \frac{c}{f} \text{ and the path-loss exponent,}$$

$n = 2$ (for path loss in free-space)

$P_{S_{RX}(dB)} = P_{S_{TX}(dB)} + G_{TX(dB)} + G_{RX(dB)} + 20 \log\left(\frac{1}{4\pi d}\right) + 10n \log(\lambda)$ is known as the Friis equation [KaWi05, Seyb05].

Assuming unity gain for both the transmit and the receive antennas,

$PL_{(dB)} = 20 \log \left(\frac{4\pi d}{\lambda} \right)$ is known as the Free-space path loss (FSPL)

$$PL_{(dB)} = P_{S_{TX}(dB)} - P_{S_{RX}(dB)}$$

To obtain maximum distance versus SNR_{TX} plot, we first need to find out what $P_{S_{TX}}$ is. $P_{S_{TX}}$ can be found by applying the following formula (from Friis equation and FSPL):

$$P_{S_{TX}(dB)} = PL_{(dB)} + P_{S_{RX}(dB)}$$

Thus, in section 6.4.1.1, $P_{S_{TX}}$ is computed using the above formula.

6.3 Experimental Results & Discussion: PvS_{RX}

The first experimental result consists of the plot of packet error rate, PER, versus signal-to-noise ratio at the receiver, SNR_{RX} , which shall be denoted by PvS_{RX}. There will be three such plots to be computed, namely PvS_{RX} at 1 m, and respectively at 2 m, and 4 m. Each plot makes use of the following data:

- (i) A specific nodes separation, D , where $D = d_i$, where $i = 1, 2$, and 3 , such that $d_1 = 1$ m, $d_2 = 2$ m, and $d_3 = 4$ m. These distances were chosen with consideration of the typical dimensions (width and length) of the structures (such as the cab and the arm) in the target application (excavator) where wireless nodes can be placed apart from one another. A quick verification (done by referring to the specifications of a compact excavator such as [Bobc09]) confirms that 4 metres is sufficient to account for the length of the cab. In other words, wireless nodes can be placed at most a distance of 4 metres apart.
- (ii) Range of PER, where each PER point on the 'PvS_{RX} ($D = d_i$)' plot is computed as the average of the resulting PER values obtained from ten experimental runs performed at a particular nodes separation, D . Each run consists of recording the number of packets dropped due to the effects of turning the input noise on (at an amount of P_N at the receiver device to corrupt the received transmitted-signal before the received signal

enters the demodulation stage) for the entire duration of a 1000-packet transmission from a transmitter to a receiver.

- (iii) Range of SNR_{RX} , where each SNR_{RX} point on the 'Pv S_{RX} ($D = d_i$)' plot is calculated from computing the received signal power and the noise power from the offline captured traces. The received signal power used is the average value of the power values computed from ten traces of the received signal captured during a 1000-pkt transmission when no noise was present. Note that the same ten traces of received signal captured were used across the different noise types per nodes separation. The noise power for each of the levels used in (ii) is the average value of the power values computed from three traces of the noise signal captured at the receiver when there is no signal transmission (i.e., when the transmitter is not transmitting packets). The term "offline" is used because both the signal and the noise traces were obtained "after-the-fact", that is after the PER measurements. Hence, the SNR_{RX} values are considered as offline rather than real-time measurements. The offline measurements are not ideal but should be accurate as the configurations of the equipment involved in the PER measurements have not changed since the signal and the noise traces were captured immediately after the PER data for each distance have been gathered.

6.3.1 Calculations: How PER and SNR_{RX} are Obtained

For each noise power of an injected noise type, the following performance parameters are calculated based on the collected data. The data for each noise type consists of seven records, each record consisting of three elements, namely the number of packets dropped or lost (L), and the captured trace of the noise signal used ($trace_N$), and the captured trace of the received signal ($trace_S$) that resulted in L. Thus, the data for a specific input noise type = (L, $trace_S$, $trace_N$)_{n,D},

where $n = 1..7$. The signal power is a constant value despite the noise power level since the same transmitter output power was used throughout experiments for different nodes separation.

To obtain packet error rate, PER (in percentage):

$$\text{PER} = \sum_{i=1..R} [(L_i/T) \times 100\%]/R, \text{ where}$$

T = total number of packets transmitted in each experimental run. In this case, 1000 packets were transmitted per experimental run, therefore $T=1000$.

R = total number of experimental runs. In this case, for each noise power level being injected, ten runs of 1000-packet transmission were performed, therefore $R=10$.

i = i -th experimental run.

L = total number of lost packets in each experimental run, where $L = T -$ total number of packets received (as reported by the PER monitoring software).

To obtain SNR_{RX} (in dB):

$$\text{SNR}_{\text{RX}} = 10 \times \log_{10}(P_{\text{SRX}}(\text{mW})/P_{\text{N}}(\text{mW})), \text{ where}$$

To obtain noise power (in mW), the following calculation is performed. The captured trace is that of the high frequency noise (more accurately, the input noise is the product of amplitude modulating a baseband monofractal noise) output by the noise injection circuit.

$$P_{\text{N}} = [\sum \text{Trn}_{j=1..n}/n], \text{ where}$$

j = j -th trace.

n = total number of captured traces. In this case, three traces were captured, therefore $n=3$.

Trn = the computed power (in mW) within 2 MHz bandwidth (between 2409 MHz and 2411 MHz for ZigBee channel 12, which is centered at 2410 MHz) of each captured noise trace. The power value was computed by feeding the trace to `compPowerBW.m` (see Appendix A.4) function in MATLAB.

To obtain signal power (in mW), the following calculation is performed. The captured trace this time is however that of the packets transmission.

$P_{SRX} = [\sum Trs_{k=1..n}/n]$, where

$k = k$ -th trace.

$n =$ total number of captured traces. In this case, ten traces were captured, therefore $n=10$.

$Trs =$ the computed power (in mW) within 2 MHz bandwidth (between 2409 MHz and 2411 MHz for ZigBee channel 12, which is centered at 2410 MHz) of each captured ZigBee signal trace. The power value was computed by feeding the trace to `compPowerBW.m` (see Appendix A.4) function in MATLAB.

6.3.2 Simplification / Assumption

The assumption and/or simplification that holds for the analysis of this experimental result is that the input noise is uniformly-distributed in time and in space during each transmission of 1000 packets from the transmitter to the receiver.

6.3.3 Why PvS_{RX} Plot

6.3.3.1 On Why PER vs. SNR instead of BER vs. SNR

The usual performance plot consists of datapoints of BER versus SNR_{RX} . One such example is seen in (Fig. E.2 on pg. 644 of [IEEE03]). However, I have used PER versus SNR_{RX} . This is because:

- (i) The PER monitoring software, which provides the firmware for the transmitter and receiver boards, is PER-based. In other words, the received data is in terms of the number of received packets.

- (ii) The receiver sensitivity definition as stated in the IEEE 802.15.4 2003-standard specification [IEEE03] is PER-based, as pointed out by a datasheet (pg. 58 of [Texa07]), which advocates PER measurements instead of BER.

Although not part of this thesis, BER measurements can be performed with a few manoeuvres following the suggestions in (pg. 207 of [Call03], and pg. 58 of [Texa07]), but care must be taken to ensure that the behaviour of the transceiver operation still reflects the norm.

6.3.3.2 On the Usefulness of PvS_{RX} Plot

In terms of usage of the PvS_{RX} plot, the plot of maximum distance versus SNR_{TX} (MvS_{TX}) can be achieved through the PvS_{RX} plots for several different nodes separation and together with the collected data. This idea will be clearer from the calculations presented in section 6.4.1.

Furthermore, the PvS_{RX} plot can benefit wireless network designer since it tells the operational conditions of ZigBee in industrial environment.

6.3.4 Interpreting the PvS_{RX} Plot: Observations and Findings

The outcome of the PER empirical experiments (i.e., the gathered data) resulted in the plots shown in Figs. 6.2 thru 6.4. Each of these plots signifies the PER versus SNR_{RX} recorded for ZigBee packets transmission under each of the four noise types, repeated over 3 trials, where each trial is at a different transmitter-receiver nodes separation.

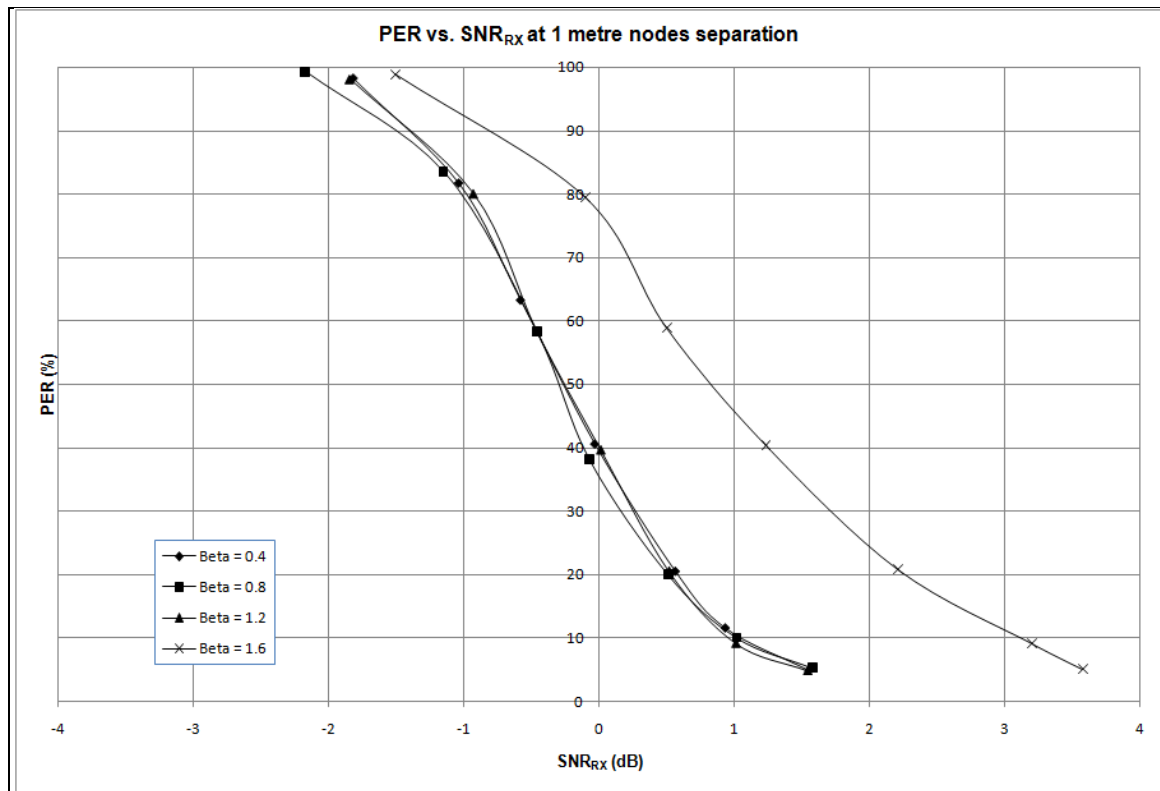


Fig. 6.2 Performance of ZigBee communication in the presence of synthetic noise (modulated monofractal noise of spectral exponents 0.4, 0.8, 1.2, and 1.6, respectively) for nodes separation of 1 metre.

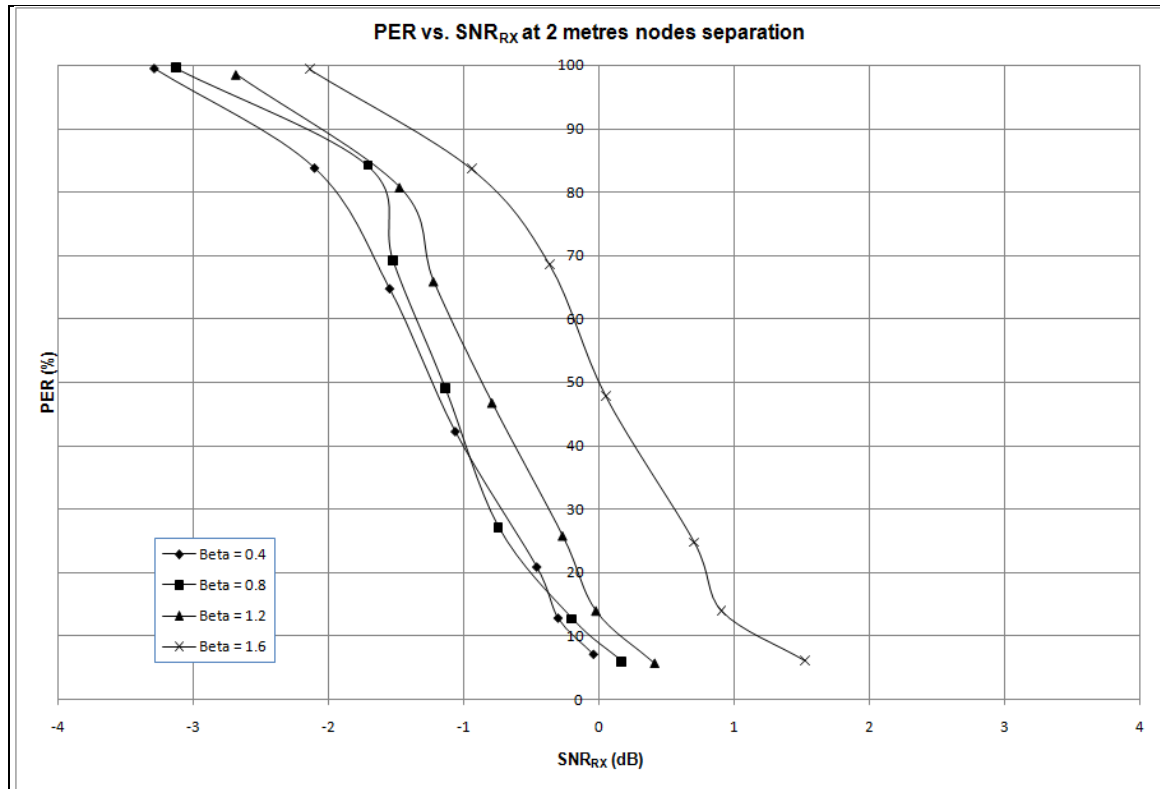


Fig. 6.3 Performance of ZigBee communication in the presence of synthetic noise (modulated monofractal noise of spectral exponents 0.4, 0.8, 1.2, and 1.6, respectively) for nodes separation of 2 metre.

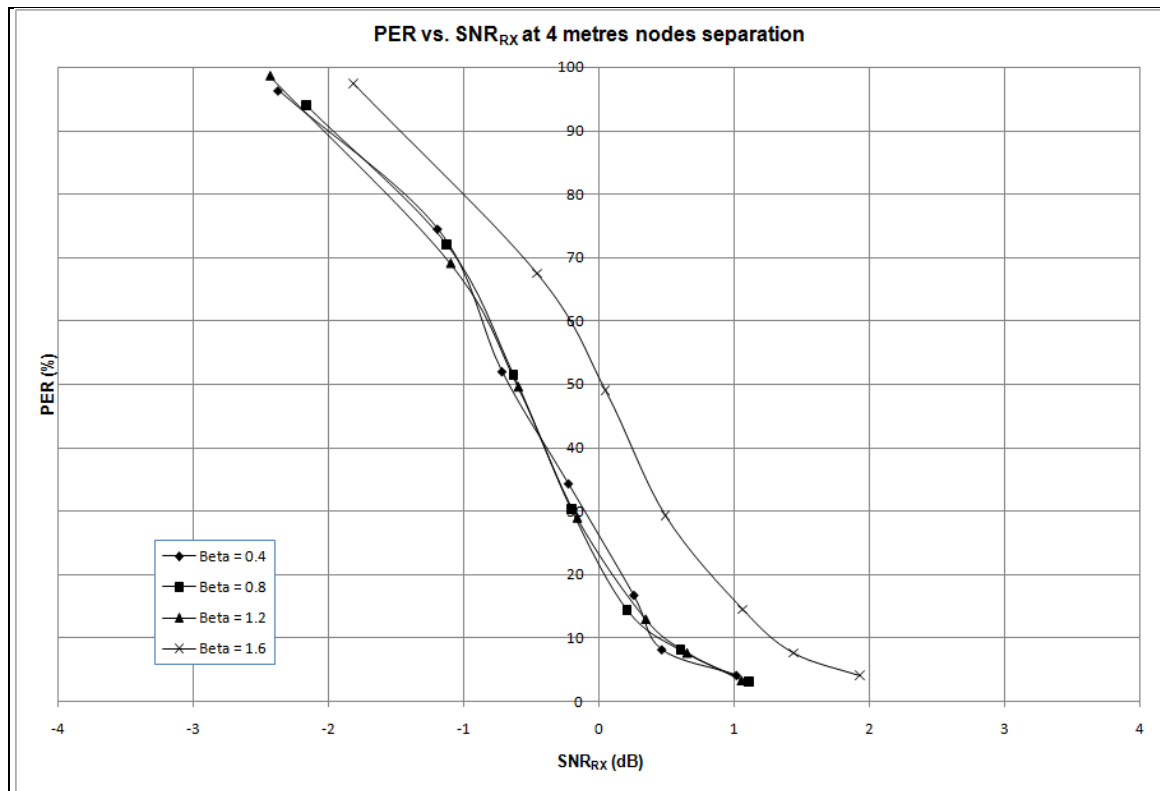


Fig. 6.4 Performance of ZigBee communication in the presence of synthetic noise (modulated monofractal noise of spectral exponents 0.4, 0.8, 1.2, and 1.6, respectively) for nodes separation of 4 metre.

Observations

- (i) Trend in each curve on each figure: As SNR_{RX} was increased (i.e., as modulated noise power becomes weaker compared to the ZigBee signal power), PER was decreased. Same observation across all the four types of input noise in each figure.
- (ii) Vertical observation of each figure: At each SNR_{RX} value, the average PER measured for input noise originating from baseband noise with spectral exponent of 0.4, 0.8, and 1.2, respectively, is much lesser than the average PER measured for input noise originating from baseband noise with spectral exponent of 1.6. Figure 6.4 for example, the offset ranges between 10% and 20% PER.

(iii) Horizontal observation of each figure: At each PER value, a smaller amount of noise power was needed (since larger SNR_{RX}) for noise type originating from baseband noise with spectral exponent of 1.6 compared to the noise power required by noise types originating from baseband noise with spectral exponent of 0.4, 0.8, and 1.2, respectively, to induce the same value of PER.

(iv) Trend across figures (i.e., trend across nodes separation): see section 6.3.4.1.

Interpretation

Observation (i) indicates that all the PER vs. SNR_{RX} plots (Figs. 6.2 to 6.4) follow the typical trend of performance curve, i.e., the S-shape. This in turn verifies that the experimental setup (which includes the communication network module; the noise modulation and injection module; and the trace acquisition module for power measurements at receiver) is functioning correctly.

Observations (ii) and (iii) indicates that higher correlated input noise (i.e., noise modulated from higher correlated baseband monofractal) causes a worsen impact in terms of the loss of packets during a ZigBee/802.15.4 packets transmission.

6.3.4.1 Trend of the PvS_{RX} Plots across Different Nodes Separation

The trend of the PvS_{RX} plots across different nodes separation is expected to be identical. In other words, the plots (Figs 6.1 (a) to (c)) should overlap onto each other perfectly. The explanation for this is as follows: The transmitter output power is fixed at the same value for all nodes separation. Due to the inverse square law, the transmitted signal deteriorates over distance such that the received signal power observed at a receiver d_1 metres away from the transmitter will be greater than that observed at a receiver d_2 metres away from the transmitter, i.e., $P_{s_{rx_d1}} > P_{s_{rx_d2}}$, given $d_1 < d_2$. As a consequence, the set of noise power values that will impact the received signal to cause a PER ranging from 0% to 100% for the two distances would be: $\{P_{n_{rx_d1}}\} > \{P_{n_{rx_d2}}\}$. But, because each PER value corresponds to one unique SNR_{RX} value, when

calculating $\{\text{SNR}_{\text{rx}_d1}\}$ and $\{\text{SNR}_{\text{rx}_d2}\}$, they will be equivalent, i.e., $\{\text{SNR}_{\text{rx}_d1}\} = \{\text{SNR}_{\text{rx}_d2}\}$. In other words, the same set of ratios (the SNR_{RX} values) that correspond to the range of PER from 0% to 100% will be observed despite the different nodes separations. Thus, $\text{PvS}_{\text{rx}_d1}$ plot = $\text{PvS}_{\text{rx}_d2}$ plot. This is a feature of the PER measurement procedure for the tuning of noise power amount to be added to the received signal at the receiver, where the procedure requires that the noise power be tuned to one that will impact a certain PER value (with the targeted set of PER values being 0%, 10%, 20%, 40%, 60%, 80%, and 100%).

The plots (Figs 6.1 (a) to (c)) did not overlap onto each other perfectly. The explanation for this is as follows: This may be due to the sensitivity of the receiving antenna that changes after every set of experiments or trial, where the transmitter node and the receiver node are placed at a different node separation for each trial. At each trial, the transmitter node is manually moved to a different distance apart from the receiver node to alter their nodes separation. So, any offset that exists in the alignment between the transmitting and the receiving antennas will affect the receiving antenna's directivity, hence its gain (pg. 39 and pg. 63 of [Seyb05]). This in turn will affect the amount of transmitted signal arriving at the receiver. Thus, the measured values of received signal power across different nodes separations will not follow the expected values of received signal power calculated using the transmitter output power and the basic Friis equation. And due to the PER measurement procedure, the noise power measurements are affected accordingly.

6.4 Experimental Results & Discussion: MvS_{TX}

The second experimental result consists of the plot of maximum distance for nodes separation, MD, versus signal-to-noise ratio at the transmitter, SNR_{TX} , which shall be denoted by MvS_{TX} . There will be one such plot to be computed per desired reliability in terms of packet error

rate tolerable. For example, I have computed MvS_{TX} for a reliability of 20% tolerable PER (Fig. 6.6). The plot makes use of the following data:

- (i) A specific PER. This value is specified by the user and denotes the desired reliability.
- (ii) Range of nodes separation. These values follow that used in the PER experiments.
- (iii) Range of SNR_{TX} . See section 6.4.1 on how these values are obtained.

6.4.1 Calculations: How MD and SNR_{TX} are Obtained

6.4.1.1 On How to Obtain MvS_{TX} Plot

First, a table with nine columns (hence, nine components) will need to be constructed. These components are calculated as follows:

- (i) The spectral exponent, β , of the monofractal noise that led to the input noise. There are four such values, 0.4, 0.8, 1.2, and 1.6. These values are computed as within the range of Hurst exponents that describes the complexity or signature of starter motor noise, as found in section 4.3 (in particular 4.3.2.2).
- (ii) Desired reliability in terms of packet error rate tolerable, PER^* . I have arbitrarily chosen 20% PER to plot a sample of the figure that denotes the second experimental result.
- (iii) Nodes separation, D^* . The values used in the PER experiments were 1, 2, and 4 metres.
- (iv) Path loss for each nodes separation, PL_{D^*} . This value is computed using the following formula:

$$PL_{D^*}(dB) = 20 \log_{10} \left(\frac{4\pi D^*}{\lambda} \right)$$

where the wavelength, $\lambda = c/f$, the propagation speed, $c = 3 \times 10^8$ m/s, and frequency, $f = 2.41$ GHz = 2.41×10^9 /s.

- (v) The value of signal-to-noise ratio at the receiver when packet error rate is PER^* valued, $SNR_{RX_PER^*}$. This value is extracted from the PvS_{RX} plot for 1 m, 2 m, and 4 m nodes

separation, respectively, for each input noise type. With four different input noise signals and three different nodes separation, there will be in total twelve values for $SNR_{RX_PER^*}$ being extracted. The unit for $SNR_{RX_PER^*}$ is in dB.

- (vi) The power of the received signal that led to packet error rate being PER^* valued, $P_{SRX_PER^*}$. This value is equal to the received signal power recorded when node separation is D^* valued, $P_{SRX_D^*}$ (in mW), which is extracted from the collected data. With fixed transmitter output power and three different nodes separation, there will be in total three values for $P_{SRX_PER^*}$ being extracted. To state $P_{SRX_PER^*}$ in dBm, simply compute:

$$10 \log_{10} \left(\frac{P_{SRX_PER^*} (mW)}{1 mW} \right)$$

- (vii) The power of the input noise signal injected at the receiver that led to packet error rate being PER^* valued, $P_{NRX_PER^*}$. This value is computed using the information from (v) and (vi). The unit for $P_{NRX_PER^*}$ is stated in mW and in dBm. For $P_{NRX_PER^*}$ in mW, the following formula is used:

$$P_{NRX_PER^*} (mW) = P_{SRX_PER^*} / 10^{(SNR_{RX_PER^*}/10)}$$

Then, a conversion from mW to dBm is done using:

$$10 \log_{10} \left(\frac{P_{NRX_PER^*} (mW)}{1 mW} \right)$$

- (viii) The power of the transmitted signal that led to packet error rate being PER^* valued in the ZigBee-noise performance evaluation experiment, $P_{STX_PER^*}$. This value is computed using the information from (iv) and (vi). The unit for $P_{STX_PER^*}$ is stated in mW and in dBm. For $P_{STX_PER^*}$ in dBm, the following formula is used:

$$P_{S_{TX_PER^*}}(dBm) = PL_{D^*}(dBm) + P_{S_{RX_PER^*}}$$

Then, a conversion from dBm to mW is done using:

$$P_{S_{TX_PER^*}}(mW) = 10^{(P_{S_{TX_PER^*}}(dBm)/10)} \times 1 mW$$

- (ix) The value of signal-to-noise ratio at the transmitter when packet error rate is PER^* valued, $SNR_{TX_PER^*}$. This value is computed using the information from (viii). The unit for $SNR_{TX_PER^*}$ is in dB. The value can be obtained using either

$$SNR_{TX_PER^*}(dB) = 10 \log_{10} \left(\frac{P_{S_{TX_PER^*}}(mW)}{P_{N_{RX_PER^*}}(mW)} \right) \text{ or}$$

$$SNR_{TX_PER^*}(dB) = P_{S_{TX_PER^*}}(dBm) - P_{N_{RX_PER^*}}(dBm)$$

Note that it is alright to use the $P_{N_{RX_PER^*}}$ value, which is measured at the receiver, in computing $SNR_{TX_PER^*}$ at the transmitter, because of the assumption that input noise is uniformly-distributed in time and in space.

Then, the values from (iii) and (ix) are plotted to produce Fig. 6.6. There will be four curves on the MvS_{TX} plot, each corresponding to a different input noise. Refer to Appendix D for the constructed table.

6.4.1.2 Possible Reasoning on Why the Computed P_{STX} was Not as Expected

Problem: The entire PER experiments were carried out with the transmitter node transmitting packets at a constant output power, which was hard-coded to 0 dBm (the typical transmit power used by IEEE 802.15.4 devices [IEEE03]) in the transmitter's firmware. The computed P_{STX} (calculated as stated in bullet item (viii) of 6.4.1.1) for the column of the table found in Appendix D is thus expected to show the same power value for all different nodes separation. However, inconsistent transmit power values (i.e., $P_{STX_{1m}} \neq P_{STX_{2m}} \neq P_{STX_{4m}}$) were computed.

Reasoning: This could be due to two possible reasons; inaccurate mapping function, f , and/or inaccurate measured value, P_{SRX} .

The first may be due to the assumptions taken when using the mapping function (Friis equation in section 6.2.3). For example, since the type of antenna at the transmitter is not similar to that at the receiver (i.e., the receiving antenna is a microstrip patch antenna (directional), whereas the transmitting antenna is the board's built in F-antenna (omnidirectional)), the assumption that both antenna gains are unity is oversimplified.

The possible causes for the latter may be:

- (i) This discrepancy could be attributed to the sensitivity of the “directional” receive antenna having changed its gain when the antenna was reoriented (moved and re-aligned to the “omni-directional” transmit antenna by eye) for the experiment to be repeated for a different nodes separation. A different antenna gain (due to a change in antennas alignment) affects the recorded power for the noise and the ZigBee signals, respectively.
- (ii) The discrepancy may also be due to errors in the measured PS_{RX} values. The errors may be contributed from having averaged X number of consecutive traces of the measured signal at no specific instance. The instance may not be long enough time-wise to represent the length of the packets transmission, which is an effect of offline or non-real time measurements.
- (iii) Another factor contributing to the discrepancy in signal measurements could be due to the anechoic chamber being not 100% reflection-proof. Reflection will add or subtract from the transmitted signal. There may be remnants of reflected signals due to the radiation absorption materials (RAMs) not absorbing the reflected signals completely.

Note: The MvS_{TX} plot (derived from the computed $P_{S_{TX}}$) is still relatively accurate, with an offset that is systematic throughout, i.e., the error that is present is systematic across all the data and

plots. This is because the different variability that exists is consistent throughout the time of the experiments.

6.4.2 Simplifications / Assumptions

Following the explanation in 6.4.3.1, $P_{S_{TX}}$ values are required to plot MvS_{TX} . However, because I did not collect SNR_{TX} data from my performance evaluation experiments, I obtained such data by reverse engineering from the gathered SNR_{RX} measurements using the well-known, basic Friis equation (i.e., in its simplest form which holds for ideal conditions). It makes sense to use such form of the equation to figure out $P_{S_{TX}}$ from $P_{S_{RX}}$ considering my measurements for the entire performance evaluation experiments were performed within an anechoic chamber. But, because the anechoic chamber is semi-lined, reflections may exist. Despite that, errors if exist is systematic and consistent throughout. Since this is the best I can do, the computed $P_{S_{TX}}$ values are considered the best estimate to the reality. In computing SNR_{TX} , the $P_{N_{TX}}$ is also required. The $P_{N_{TX}}$ value is assumed to be equivalent to $P_{N_{RX}}$. This is made possible because of the equal noise distribution assumption, where noise is not local but same concentration at all points in space and in time.

6.4.3 Why MvS_{TX} Plot

6.4.3.1 On Why MvS_{TX} and Not MvS_{RX}

The following description will show that it is the SNR_{TX} (not the SNR_{RX}) that matters in relation to nodes separation (i.e., the distance between the transmitter node and the receiver node).

Assumption:

- (i) Noise power is equally distributed in space. In other words, the noise power at every point in distance is constant.

Procedure:

1. For fixed transmit output power, $P_{STX_1m} = P_{STX_2m} = P$. By ISL, $P_{SRX_1m} > P_{SRX_2m}$. Assuming noise is constant at all nodes separation, then $PER_{RX_2m} > PER_{RX_1m}$.
2. Increase P_{STX_2m} such that $PER_{RX_2m} = PER_{RX_1m}$. This in turn ensures $SNR_{RX_1m} = SNR_{RX_2m}$ and $SNR_{TX_1m} < SNR_{TX_2m}$.

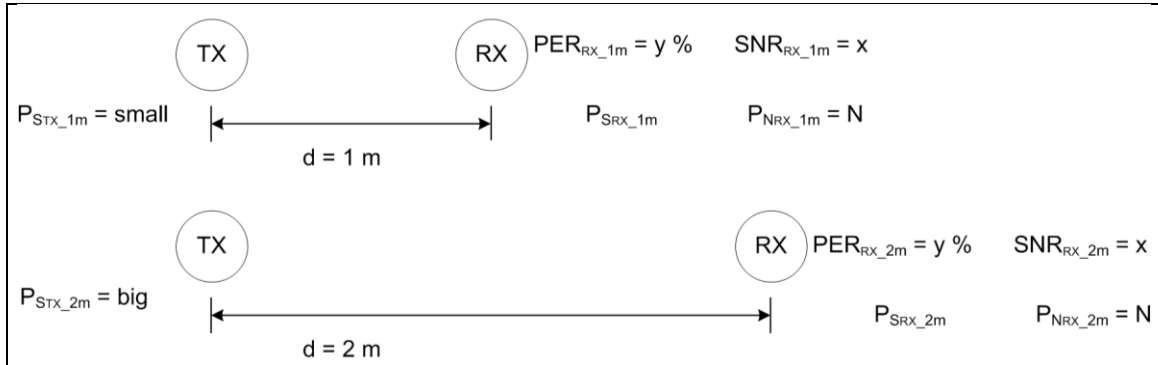


Fig. 6.5 Visual aid depicting the relationship between distance of nodes separation and transmitter output power to induce the same PER value at the respective receivers.

Consequences:

- 1) As nodes separation is increased, SNR_{TX} is required to increase to achieve the same PER value. This scenario is depicted in Fig. 6.5.
- 2) Following the equal noise distribution assumption, $P_{NTX} = P_{NRX}$. Thus, can reuse PER vs. SNR_{RX} plots (such as in Figs. 6.2 to 6.4) to produce maximum distance vs. SNR_{TX} for a desired reliability plot (Fig. 6.6).

6.4.3.2 On Usefulness of MvS_{TX} Plot

In terms of usage of the MvS_{TX} plot, it can benefit wireless network designer in terms of assisting in the placement of nodes and in determining the minimum number of nodes in a network that will ensure a certain reliability is achieved given the SNR at the transmitter. Section 6.4 provides the procedure on how the MvS_{TX} is used to determine the latter.

6.4.4 Interpreting the MvS_{TX} Plot: Observations and Findings

This section presents the observations and findings pertaining to Fig. 6.6, where each curve on the plot corresponds to the maximum distance vs. SNR_{TX} for reliability with 20% PER for a specific input noise type. Note that PER and SNR_{RX} have a 1:1 mapping, which means that 20% PER is associated to a specific SNR_{RX} value (hence, a specific P_{SRX} value). Essentially, Fig. 6.6 provides the answer to the question of what the value for P_{STX} should be in order for a specific P_{SRX} value to be observed at X distance away from the transmitter. In other words, how much transmit output power should be supplied to obtain P_{SRX} at X distance away from the transmitter. Furthermore, the MvS_{TX} plot (Fig. 6.6) is unique in the sense that it reflects the character of the input noise type, i.e., the high-frequency fractal noise types. To show an example on how to read Fig. 6.6; at nodes separation of 3 metres, SNR_{TX} is approximately 50 dB. This is saying that the ZigBee signal must overpower the noise signal by 10^5 times.

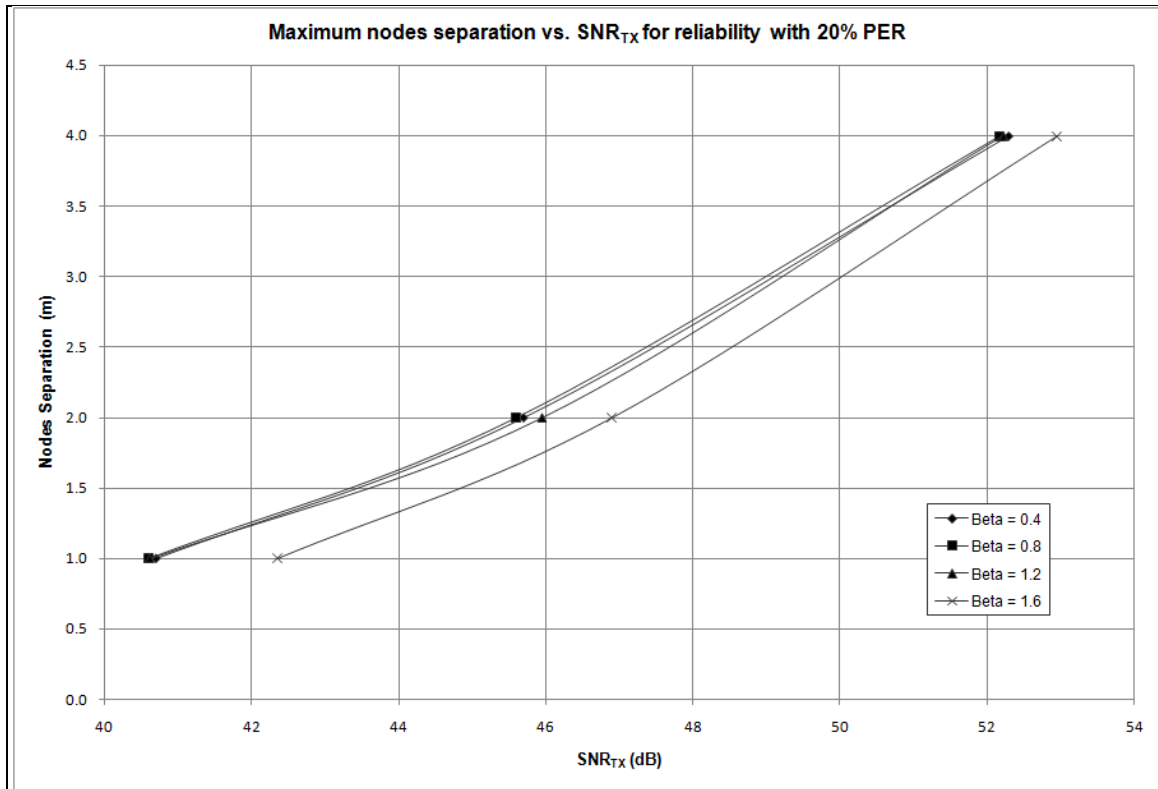


Fig. 6.6 Maximum nodes separation as a function of signal-to-noise ratio at the transmitter for reliable communication (indicated by the value of PER tolerable), corresponding to each input noise type.

Observations and Their Interpretation/Significance

The trend of each curve in Fig. 6.6 is an exponential growth. It cannot be a straight line. This is as expected because signal strength away from a point source is a function of the inverse of distance squared, i.e., follows the inverse square law. So, as nodes separation increases, SNR_{TX} (hence, P_{STX}) has to be increased in order to affect the desired P_{SRX} value (hence, a specific SNR_{RX} to induce 20% PER) at the receiver.

Because zero or negative distance between two nodes does not exist, when the curves in Fig. 6.6 are extrapolated, the trend to the left-hand-side (LHS) of the plot is expected to saturate and approach zero metre. This is because the distance between two nodes can never be zero, otherwise the two nodes will become one. The cause for the saturation may be due to near-field

effect. As shown in section 6.3.4.1, the near-field effect dominates when the nodes are separated no greater than 0.055 metre. The trend to the right-hand-side (RHS) of the plot is expected to be bounded by the maximum affordable transmit output power for the chosen wireless protocol. The LHS and RHS are the anticipated extremities of the wireless system.

The noise type affects the relationship between nodes separation and SNR_{TX} as follows: For a certain SNR_{TX} value, nodes separation increases as correlation in noise samples reduces. On the other hand, for a certain nodes separation, more transmit power is required for increasing correlation in noise samples. The first (i.e., referring to the vertical observation) is saying that higher correlated noise effects are more severe than lower correlated noise and as such would require the nodes to be brought closer in distance. The latter (i.e., referring to the horizontal observation) is saying that higher correlated noise effects are more severe than lower correlated noise and as such would require the system to increase or boost its transmit output power.

6.5 Usage of PvS_{RX} and MvS_{TX}

This section presents one usage of the MvS_{TX} plot, which is generated from the plots of PvS_{RX} for various nodes separation. The usage is that the minimum number of wireless nodes within a volume of application space can be determined. An example of the usage is presented, where the methodology for placement of nodes from the perspective of a one-dimension case is discussed. The term one-dimension is used to indicate the formation of a network based on a single, direct, one-line path of nodes distribution. Extension of the methodology for placement of nodes to more than 1 dimension is possible, however is not investigated in this thesis.

6.5.1 Placement of Nodes: 1-D Methodology

There are two methods being suggested for determining the minimum number of wireless nodes that achieves a desired tolerable packet error rate, given a volume where the source node and the destination node are situated at the farthest corner from each other, as shown in Fig. 6.7.

The basic concept underlying the two methods is that minimum number of nodes is achieved when setup using the maximum nodes separation obtained from an MvS_{TX} plot for a specific input noise. The choice of method to use depends on the meaning of or the goal for setting the desired tolerable PER (denoted as $G\%$). The first method defines $G\%$ as the set tolerable PER as seen by each receiver wireless node. The second method defines $G\%$ as the set tolerable PER as seen by the last wireless node, i.e., the destination node. In other words, the first denotes $G\%$ as the set tolerable PER per node, whereas the latter denotes $G\%$ as the set tolerable PER per system.

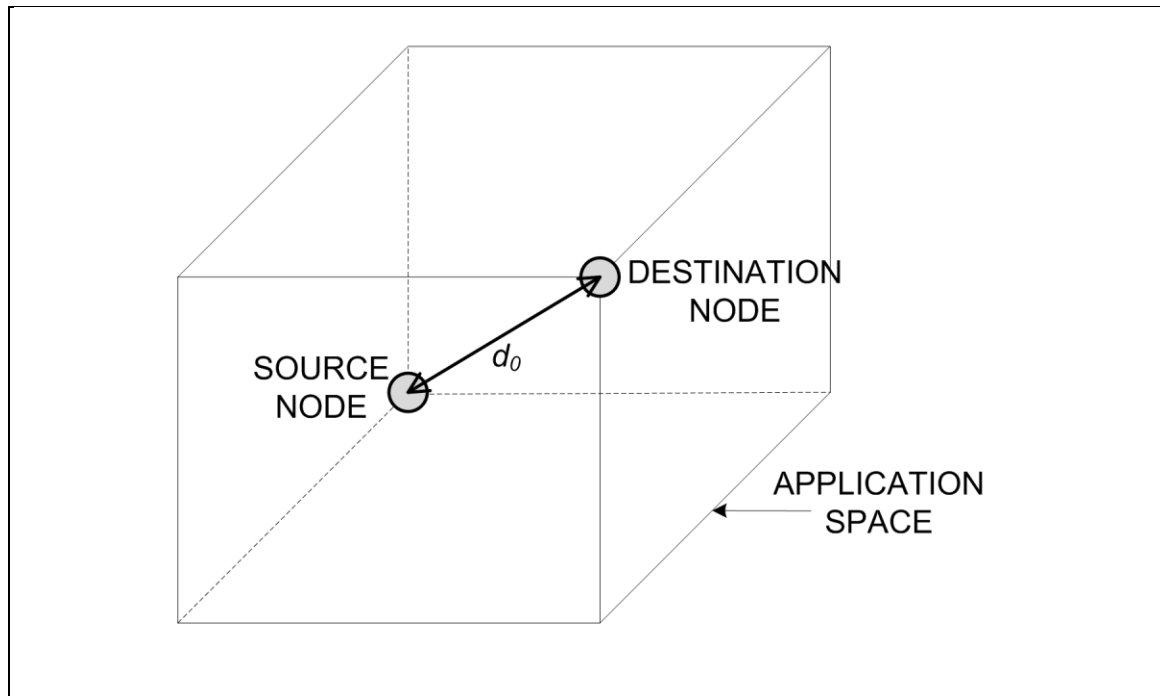


Fig. 6.7 The volume of the space for placement of router nodes between a source node (say, a sensor) and a destination node (say, the main controller) situated at d_0 distance apart from each other.

The following presents the assumptions, known parameters, and unknown parameters that are common to both the methods:

Assumptions:

- The placement of router nodes is in a straight line path (i.e., one-dimension, 1-D). Such path leads to the number of wireless router nodes being the minimum since it is the shortest path between the source node and the destination node.
- Each router node is placed at the same distance apart from the previous node.
- Equal noise distribution assumption, i.e., same noise power experienced by each receiver node.

Known parameters:

- The distance between source and destination nodes, d_0 .
- The desired tolerable PER, $G\%$. This is the design goal set by the wireless system designer.
- The actual SNR at the transmitter, SNR_{TX_A} , computed using:

$$SNR_{TX_A}(dB) = 10 \log_{10} [P_{STX_AFF}(mW)/P_{NMAX}(mW)] \text{ or } SNR_{TX_A}(dB) = P_{STX_AFF}(dBm) - P_{NMAX}(dBm), \text{ where}$$

- P_{STX_AFF} = the affordable transmitter output power; and
- P_{NMAX} = the maximum noise power measured at the site. See note for a suggestion on how to measure the value for this parameter. The maximum noise power derivation method suggested here is an estimation to simplify the real scenario. It is difficult to have a mathematical model (will end up being too complex) to encompass all scenarios.

Note: The computation of the maximum noise power, P_{NMAX} , value would require:

- a repeated transmission of a beacon signal for a period of time
- a continuous recording of $|P_{SRX_EXP} - P_{SRX_MEAS}|$ for a period of time, where
 - Received signal power measured, P_{SRX_MEAS} , value is obtained from measurement, where P_{SRX_MEAS} = signal power calculated from captured trace of the received signal with a RTSA;

- Received signal power expected, $P_{\text{SRX_EXP}}$ value is obtained from calculation, where $P_{\text{SRX_EXP}}$ = computed from the Friis equation.
- a peak detector, such that $P_{\text{NMAX}} = \max(\{|P_{\text{SRX_EXP}} - P_{\text{SRX_MEAS}}|_i\})$. In other words, the peak detector is used to detect the highest (i.e., worst) noise power from the set of noise power instances (with each instance being referred by the counter i) computed for the period of time of the received signals measurements.

Unknown (need to find out) parameter:

- The maximum distance for nodes separation, d_{max} , to place wireless node(s) apart from the previous wireless node.
- The number of router nodes, N_{ROUTER} , required between the source node and the transmitter node, so that the minimum number of wireless nodes in a network, $N_{\text{NODES_MIN}}$, can be computed as follows: $N_{\text{NODES_MIN}} = 2 + N_{\text{ROUTER}}$, where the source node and the destination node account for the constant 2 in the equation.

6.5.1.1 Method 1 – The Resend Method for Lossless Case

This method is named as the resend method since the method (with retransmission of packets) leads to a lossless communication system.

Disadvantage:

Slow transmission rate, because:

- Time spent by transmitting node will increase because of retransmissions (i.e., the transmitting node will retransmit up to a maximum number of retransmissions, unless it receives an acknowledgement from the receiving node before timing out) to be done at each hop before sending the next packet;
- Time spent by receiving node = time to wrap and send resend request + time to obtain all the requested resent packets + time to accept incoming packets (a new sequence of packets).

Advantage:

Only require one MvS_{TX} plot.

Assumptions:

- The entire system (from the source node to the destination node) is lossless.
- The system employs some form of automatic repeat request (ARQ) for data transmission.
- Each receiver node has the resources (memory and processing power) to handle a maximum of G% PER.
- Wait to receive all the G% packets that were dropped or lost before transmitting to the next node.

Procedure:

1. Set design goal, say G% PER. This is the set tolerable PER per node.
2. Given the volume of the application space, determine the distance between the source node and the destination node, say d_0 metres.
3. Given the affordable transmitter output power, P_{STX_AFF} and the measured maximum noise power, P_{NMAX} , determine the actual signal-to-noise ratio at the transmitter, SNR_{TX_A} . Use $SNR_{TX_A}(dB) = 10 \log_{10} [P_{STX_AFF}(mW)/P_{NMAX}(mW)]$ or $SNR_{TX_AFF}(dB) = P_{STX_AFF}(dBm) - P_{NMAX}(dBm)$.
4. From the plot of MvS_{TX} for reliability governed by G% PER for a specific input noise (i.e., use a spectral exponent from the experimental data that matches closely to the fractality of the noise at the site), extract the value for distance corresponding to the SNR_{TX_AFF} value. Let this distance be denoted as d_x .
5. Compute $m = d_0/d_x$, such that if m evaluates to a fractional number, set $N_{ROUTER} = \lfloor m \rfloor$, and if m evaluates to an integer number, set $N_{ROUTER} = m - 1$.

6. Place N_{ROUTER} number of router nodes in a straight line between the source node and destination node, such that each router node is at maximum distance of $d_{\text{max}} = d_x$ apart from the previous node. This signifies the shortest path.
7. For redundancy, add extra paths between the source node and the destination node. However, nodes must be placed at d_{max} apart from its surrounding nodes. This step will produce number of nodes more than the minimum required, however is recommended in case of node failure.

6.5.1.2 Method 2 – The No-Resend Method for Lossy Case

This method is named as the no-resend method since the method (with no retransmission of packets) leads to a lossy communication system.

Disadvantage:

Require more than one MvS_{TX} plot.

Advantage:

Guaranteed total number of packets in error from the source node to the destination node despite the number of intermediate router nodes between them.

Assumption:

The maximum number of packets the entire system can lose is G%.

Procedure:

1. Set design goal, say G% PER. This is the set tolerable PER per system.
2. Given the volume of the application space, determine the distance between the source node and the destination node, say d_0 metres.
3. Given the affordable transmitter output power, $P_{\text{STX_AFF}}$ and the measured maximum noise power, P_{NMAX} , determine the actual signal-to-noise ratio at the transmitter, $\text{SNR}_{\text{TX_A}}$. Use $\text{SNR}_{\text{TX_A}}(\text{dB}) = 10 \log_{10} [P_{\text{STX_AFF}}(\text{mW})/P_{\text{NMAX}}(\text{mW})]$ or $\text{SNR}_{\text{TX_AFF}}(\text{dB}) = P_{\text{STX_AFF}}(\text{dBm}) - P_{\text{NMAX}}(\text{dBm})$.

4. The trial number, i , and the number of intermediate nodes, x , are integer values. Start with $i = 1$, and $x = 0$.
 - 4.1 Add x intermediate nodes between the source node and the destination node.
 - 4.2 Determine the distance between nodes, d_x , using $d_x = d_0/i$.
 - 4.3 Determine the PER value, r , for each intermediate node that after i number of hops will yield a cumulative PER at the end receiver (i.e., the destination node) equivalent to the design goal, i.e., $G \pm 1$ PER. Compute the value for r to one decimal place using $r = G/i$.
 - 4.4 Create the plot of MvS_{TX} for reliability governed by $r\%$ PER for a specific input noise (i.e., use a spectral exponent from the experimental data that matches closely to the fractality of the noise at the site), by proceeding as described in 6.4.1.1.
 - 4.5 From the plot created in 4.4, extract the value for signal-to-noise ratio corresponding to the d_x value. Let this signal-to-noise ratio be denoted as SNR_{TX_dx} .
 - 4.6 If $SNR_{TX_dx} \leq SNR_{TX_A}$, set $d_{max} = d_x$ and $N_{ROUTER} = x$ and go to step 5. Else, increment i and increment x and go to 4.1.
5. Place N_{ROUTER} number of router nodes in a straight line between the source node and destination node, such that each router node is at maximum distance of $d_{max} = d_x$ apart from the previous node. This signifies the shortest path.
6. For redundancy, add extra paths between the source node and the destination node. However, nodes must be placed at d_{max} apart from its surrounding nodes. This step will produce number of nodes more than the minimum required, however is recommended in case of node failure.

6.6 Experimental Results & Discussion: For Experiment with Real Noise Source Results

Table 6.1 PER measurements for ZigBee transmission with physical noise source (plasma globe).

Case	PER measurements per trial (where each trial signifies a 1000-pkt transmission under the specified case)										Average PER (%)
	1	2	3	4	5	6	7	8	9	10	
No globe present	3	5	0	4	3	0	3	0	3	1	0.22
Globe OFF state	4	1	3	4	4	1	3	2	4	2	0.28
Globe ON state	28	27	11	25	27	25	27	21	23	25	2.29

Observation

Despite the powered on plasma globe being placed very close (15 inches away) to the receiving antenna, the percentage of packet error recorded for the 2-node communication network is very low (less than 3% PER, as seen in Table 1).

Interpretation/Indication

The plasma globe emits very low radiation.

Justification

First, the plasma globe is a USB device that has to comply with the electromagnetic compatibility (EMC) requirements defined by the Federal Communications Commission (FCC), which may account for the reduced amount of leakage of radiation or noise to interfere with another device. Second, the plasma globe may have some form of shield that contains the radiation within the globe.

6.7 Chapter Summary

This chapter has presented and analyzed the results for the performance evaluation experiments for monitoring the packet error for a 2-node ZigBee network under synthetic noise and also under physical noise. The data collected from the performance evaluation experiments have been presented in terms of the plots, the relationship between PER and SNR_{RX} for each

input noise type at various nodes separation, and the relationship between the maximum distance between two nodes and SNR_{TX} to achieve a specific reliability.

The first plot draws two conclusions; that the performance evaluation system and procedure functioned as intended since the typical 'S'-shaped performance curve was obtained for the performance of ZigBee under each noise type, and that the higher the correlation of the input noise, the higher is the observed packet error rate.

The conclusion drawn from the second plot is that as noise correlation increases, SNR_{TX} increases for the same PER. This observation would not have been revealed had the input noise was based on uncorrelated noise alone. Thus, systems that have been simulated by uncorrelated noise model in the past may now utilize the fractal-based noise model to reveal more interesting effects to better understand effects due to the point of view of broadband electromagnetic noise. The second plot finds its use in terms of determining the placement that gives the minimum number of nodes, in which two different procedures for utilizing the second plot have been presented.

Chapter 7

CONCLUSIONS

This thesis has presented an empirical study to determine the robustness of ZigBee wireless networks to broadband electromagnetic industrial noise. ZigBee has been chosen as the wireless protocol that best matched the requirements for an industrial control, monitoring, and sensing data communication system. A fractal-based noise model has been chosen to simulate and synthesize the broadband noise source. An industrial noise capture and analysis have been carried out and its findings have validated the choice of the noise model used. An experimental setup for measuring the packet error rate performance of ZigBee protocol under synthetic noise has been designed and implemented. The synthetic noise complexity is tuned to match the complexity of the real noise emitted from the starter system of tractors. The interference is induced by adding noise to the received signal in the wired portion of the receiver before demodulation. The data collected from the conducted laboratory experiments under synthetic noise and under physical noise have been analyzed. The obtained performance plots (PER versus SNR_{TX} , and maximum nodes separation versus SNR_{RX}) have led to the following main findings.

7.1 Main Findings

Listed below are the outcomes drawn from this research work:

- (i) For each noise type, the relationship between packet error rate and SNR at the receiver follows the typical ‘S’-shaped trend of a performance curve, where PER decreases as SNR_{RX} increases. This observation verifies that the performance evaluation experimental setup has been correctly implemented. The impact of different noise types on packet loss is that the higher the correlation of the noise signal, the larger the amount of dropped or lost packets. In other words, the impact due to correlated noise is more severe than that

due to uncorrelated noise. Therefore, ZigBee's PER/SNR performance depends on the power level and the type of broadband EM noise.

- (ii) For each noise type, the relationship between maximum nodes separation and SNR at the transmitter (i.e., MvS_{TX}) that ensures a specific reliability is achieved by the communication network, can be derived from the performance plots (PER vs. SNR at the receiver) gathered over several different nodes separation for that noise type. The impact of different noise types on the maximum nodes separation is that the higher the correlation in the noise signal, the closer the distance between nodes. The impact of different noise types on the SNR_{TX} is that the higher the correlation in the noise signal, the greater the transmit output power, P_{STX} . In other words, the expense for setting a network at a site that experiences interference from correlated noise signals would be costly, since more nodes will be needed, and a higher transmit output power will be required.
- (iii) From section 6.5.1, to achieve the minimum number of ZigBee nodes per network for the network to be operating at or within a specific desired reliability in a given space under broadband noise, the distance between nodes is limited to the maximum nodes separation value, which is governed by the SNR at the transmitter (i.e., the noise power present in the space, and the affordable transmitter output power). An insight to this is that the wireless system designer does not have the luxury of placing wireless nodes apart by the maximum distance in the single hop transmission range as stated in the ZigBee specification (see Table 3.1). This is because a node can communicate directly with another node "in range", however when a specific noise power, P_N is present in the operating environment, the transmission range has to be reduced to the maximum nodes separation for the specific transmit output power (obtained from the MvS_{TX} plot) in order

to achieve the desired reliability. This reduced range is the effective transmission range for the said operating environment.

- (iv) Performance evaluation under physical broadband noise has been carried out using the noise emitted from a plasma globe. As expected, during ZigBee transmission, packets were dropped when the noise source is turned on, and there was negligible packets loss when the noise source is not turned on.

7.2 Limitations

Some of the limitations identified within this research work include:

- (i) Analog, high-frequency monofractal noise signal is not best produced using the method specified in this thesis. The first reason is because the output beta is non-specific since the spectral exponent of the modulated signal differs by a significant increase from the spectral exponent of the modulating signal. The second reason is because the region where monofractality is observed is small compared to the frequency range of the modulated noise spectrum. Refer to section 4.5.3.4 for more details.
- (ii) Anechoic-chamber is not fully-lined but semi-lined, thus not 100% reflection-free. Justification is that any realization of anechoic chamber in real world implementation is not 100% error free. Thus, this limitation is neutralized somewhat as ideal case can never be attained.
- (iii) Measurements may be affected by not simulating noise and evaluating ZigBee performance all in software. For example, the use of external antenna may affect measurements due to the impedance of its probes possibly altering the network impedance parameters. However, the offset in measurements, if any, would be consistent throughout, and thus the error would be systematic.

7.3 Recommendations

The following lists a few suggestions on possible extensions to the current work:

- (i) With respect to performance evaluation, the core elements in my thesis involve *wireless network* of a specific protocol and *noise in hardware simulation*. An improvement is to verify my work by doing wireless network and noise in *software simulation*. This work covers the baseline case, i.e., a two-node network, the basis for all topologies. To verify my empirical simulation and to go beyond the basic 2-node case to add perspective on larger network, use computer simulation to simulate the network. Available simulators include WIREless SEnsor NETwork Simulator (WISENES) that implements the MAC and NETWORKING layers [WISE], Network Simulator (NS2) with 802.15.4 extension that implements the PHY and MAC layers [ZhLe06], and MATLAB/Simulink-based simulator TRUETIME that implements the MAC layer [CeHO09] of ZigBee. Because each wireless network simulator emulates one or more specific protocol layers, the choice of which simulator to use depends on which protocol layer the problem-at-hand is to be investigated in.
- (ii) With respect to performance evaluation, a more advanced stage is to analyze performance of wireless network for CMS system under real-time control system conditions [CoSI05]. In other words, the next move is *wireless network* and *control system*, and then to wireless network, control system and noise, in hardware simulation and in software simulation. This will be beneficial both to a control system designer and a wireless system designer.
- (iii) With respect to the modulated monofractal noise analysis (in section 4.5.3.3), I have thus far performed mathematical and computer-simulated analyses. By doing an empirical analysis in hardware, the modulated monofractal noise analysis will then be complete.

The findings from the three analyses will give a fuller picture as to the characteristic of the modulated noise signal.

- (iv) With respect to reducing broadband noise effects, the only method for noise reduction or removal is through denoising (removal of noise from a signal) because of the complexity of the broadband noise type.
- (v) With respect to correcting broadband noise effects, one method is to apply error correction for ZigBee targeting burst error pattern.
- (vi) With respect to improving the ZigBee protocol to deal with the broadband noise effects, one method is to investigate into adaptive channel (not frequency) hopping scheme to continually check for and move to a new clearer channel on both the transmitter and the receiver to allow for increased throughput or more data transmission capability (source from Kenneth Doe, Freescale application engineer, of his research).

7.4 Contributions

The contributions made by this thesis are listed below:

- (i) Wireless solution using the ZigBee protocol is recommended to replace the wired solution for CMS applications (e.g., the communication system within an excavator). The determination of a suitable wireless protocol for the indicated application was done by first short-listing the list of available standard-based wireless protocols to those that operate in the 2.4 GHz frequency band. Next, the process of elimination was done by comparing the features within each of the protocol in the shortlist against the application requirements. Wi-Fi (802.11) was eliminated because its transfer rate is overkill, and Bluetooth (802.15.1) was eliminated due to its limited number of nodes due to its star topology, leaving ZigBee meeting the requirements of low cost, low data, low power, and reliable through mesh topology as the best choice.

- (ii) The steps taken for emulating a noisy radio channel involved noise sources identification, noise modelling, noise validation, noise synthesis in software, and high-frequency noise implementation in hardware.
- (iii) The noise of interest is broadband EM noise. This is because the impact of broadband noise is more severe and not trivial to overcome (reduce or eliminate) compared to narrowband noise. Some examples of broadband noise sources in a vehicular application are ignition system, alternator, and motors such as blower, wiper, and washer.
- (iv) Fractal-based noise models (specifically monofractal noises) are used to represent the noise of interest. The reason is because such model can represent both uncorrelated noise signals (such as white noise) and correlated noise signals (such as coloured noises).
- (v) Field measurements of real industrial noise have been conducted. This is because there are no accessible electromagnetic industrial noise data available for use.
- (vi) The captured industrial noise traces have been analyzed using a multiscale signal analysis method. From section 4.3, the characteristic of the time samples in the recorded instances of noise emitted by a real noise source (the starter system in a tractor) is found to be multifractal (i.e., a mixture of noises with different sloped PSD). This supports our work that advocates using a fractal-based noise model, one that departs from the commonly-used white noise model with a flat PSD, as a more realistic representation of broadband EM noise. Furthermore, having understood the complexity of the noise of interest, methods for noise removal can be devised to enhance ZigBee operations.
- (vii) The synthesis of monofractal noise sequence is done using the spectral filtering algorithm because it is easy to implement and can produce accurate estimation of the desired monofractal noise in one run.

- (viii) The analog fractal noise generation is a 2-step process. First, the analog baseband signal is produced from a hardware (an arbitrary waveform generator), where the different noise types came from having stored the respective digital noise sequences that was synthesized in software. Second, the baseband signal is amplitude modulated in hardware in order to produce a high-frequency analog noise signal.
- (ix) The characteristic of the signal produced from amplitude modulating a monofractal noise signal is found by analysis to be multifractal. However, by following an experimentally-determined ratio, a small region of fractality can be obtained in the upper-sideband of the positive frequency spectrum of the modulated noise signal. This is the method used in the thesis to produce high-frequency monofractal noise signal.
- (x) An experimental setup for assessing the packet loss performance of wireless protocol under the influence of noise, with the flexibility of arbitrary choice for the wireless protocol, the noise (type and power levels) and the distance of separation between nodes, has been designed, developed, and implemented.
- (xi) The original setup design has undergone two major revisions. The first revision was required to eliminate the issue of unstable carrier signal causing non-repeatable PER measurements since the superposition of the noise spectrum was not fixed at ZigBee bandwidth of interest at all times. The second revision was required to enable proper SNR at the receiver measurements.
- (xii) The data collected from the implemented performance evaluation system give a lower bound (i.e., worst-case) performance. This is expected as can be seen from the design of the system where noise is injected in-wire and superimposed entirely onto the ZigBee frequency band of interest (i.e., the 6-dB bandwidth centred on the ZigBee operating channel). Furthermore, retries or retransmissions are not enabled.

- (xiii) An attempt/initiative to fill gap in performance evaluation literature by showing effects due to correlated noise as interfering signal type (current lit: uncorrelated, i.e., using conventional AWGN).
- (xiv) Results advocate that there is a need to include practical measures of SNR and number of nodes for real environment, and information about the effects of correlated noise (with different degrees of correlation) in the ZigBee specification.
- (xv) Creating awareness and dispensing of information to the public, science and technology, and research community through five (5) published and one (1) accepted peer-reviewed IEEE conference papers. [WKFD06], [WKFD07], and [FeWK09] were on research progress and latest results, [WoKF08] was on real noise capture and its characteristics, [WPKF07] was on characteristics of modulated monofractal noise, and [WoFK10] was written as the final paper in the series to wrap up the research.

References

- [ABFS07] L. Angrisani, M. Bertocco, D. Fortin, and A. Sona, "Assessing coexistence problems of IEEE 802.11b and IEEE 802.15.4 wireless networks through cross-layer measurements," in *IEEE Instrum. and Meas. Technol. Conf. Proc. (IMTC) 2007* [Online], May 1–3, 2007, pp. 1–6. Available: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?tp=&arnumber=4258508
- [Agil02] *Agilent 33120A 15 MHz Function / Arbitrary Waveform Generator: User's Guide*, 6th ed., Publication No. 33120-90006 [Online], Agilent Technologies, Mar. 2002. Available: http://www.home.agilent.com/upload/cmc_upload/All/6C0633120A_USERSGUIDE_ENGLISH.pdf
- [Agil07] IntuiLink Waveform Editor for 33120A, 33220A, 33250A Function Generator [Online], ver. 1.4, Agilent Technologies. Available: <http://www.home.agilent.com/agilent/redirector.jsp?action=doc&lc=eng&cc=US&id=220465<ype=External%20File>
- [Agil94] *EMI Receiver Series: HP 8546E/HP 8546A EMI Receiver, HP 85422E/HP 85462A Receiver RF Section: User's Guide*, HP Part No. 5962-5081 [Online], Agilent Technologies, Aug. 1994. Available: <http://cp.literature.agilent.com/litweb/pdf/5962-5081.pdf>
- [AISJ06] M. Alnuaimi, K. Shuaib, and I. Jawhar, "Performance evaluation of IEEE 802.15.4 physical layer using MATLAB/Simulink," in *Int. Conf. Innovations in Inf. Technol. 2006* [Online], Nov. 2006, pp. 1–5. Available: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?tp=&arnumber=4085420
- [Anri] *MS2661C Spectrum Analyzer – 9kHz to 3 GHz*, Catalogue no. MS2661C-E-A-1-(6.00) [Online], Anritsu, May 2003. Available: http://www.eu.anritsu.com/files/MS2661C_EA1600.pdf
- [Anri03] *Fundamentals of Interference in Wireless Networks*, Application Note [Online], Spectrum Master MS2771B, Anritsu, May 2003. Available: <http://www.eu.anritsu.com/files/11410-00302.pdf>
- [BeKZ06] M. Becker, J. Kurtas, and K. Zelickson, "ZigBee networks robustness to noise in industrial environments," 24.400 Group design project, Dept. of Electrical and Computer Engineering, University of Manitoba, Winnipeg, MB, Mar. 2006.
- [BeLe80] M.V. Berry, and Z.V. Lewis, "On the Weierstrass–Mandelbrot fractal function," in *Proc. Roy. Soc. Lond.* [Online], ser. A, Mathematical, Physical & Engineering Sciences, vol. 370, no. 1743, Apr. 24, 1980, pp. 459–484. Available: <http://rspa.royalsocietypublishing.org/content/370/1743/459.full.pdf+html>
- [Bell82] Lewis H. Bell, *Industrial Noise Control: Fundamentals and Applications*, New York, NY: Marcel Dekker, Inc, Sept. 1982, 572 pp., ISBN 0-8247-1787-2.

- [BiHa05] Drew Bird, and Mike Harwood, "Introduction to networking - Factors affecting wireless signals," chap. 1 in *EXAM CRAM™ 2 Network +*, Pearson Certification, June 2005.
- [Blue08a] Bluetooth Special Interest Group (SIG), Available:
<http://www.bluetooth.com/bluetooth/>
- [Blue08b] Bluetooth Special Interest Group, "How Bluetooth Technology Works," Available:
<http://bluetooth.com/Bluetooth/Technology/Works/>
- [Bobc09] Bobcat, "430 Compact Excavator specifications," Available:
<http://www.bobcat.com/excavators/models/430>
- [Boga83] Theodore F. Bogart, *Linear Integrated Circuits: Applications and Experiments*, John Wiley & Sons Inc., 1983, 245 pp.
- [Bruc01] Eugene N. Bruce, *Biomedical Signal Processing and Signal Modeling*, John Wiley & Sons, 2008, 528 pp., ISBN 978-0-471-34540-4, 978-1-60119-547-0.
- [BuHe04] Bryan H. Bunch, and Alexander Hellemans, *The History of Science and Technology: A Browser's Guide to the Great Discoveries, Inventions, and the People Who Made Them, from the Dawn of Time to Today*, Houghton Mifflin Harcourt, 2004, 776 pp., ISBN 0618221239, 9780618221233.
- [Call03] Edgar H. Callaway, *Wireless Sensor Networks: Architectures and Protocols*, vol. 3 of *Internet and Communications*, CRC Press, 2003, 342 pp., ISBN 0849318238, 9780849318238.
- [Carr99] Joseph Carr, *Practical Radio Frequency Test and Measurement: A Technician's Handbook*, Newnes, 1999, 360 pp.
- [CeHO09] Anton Cervin, Dan Henriksson, and Martin Ohlin: "TrueTime 2.0 beta 1—Reference manual," Dept. of Autom. Control, Lund Univ., Sweden, Jan. 2009.
- [ChGo05] Nicolas Chevrollier, and Nada Golmie, "On the use of wireless network technologies in healthcare environments," in *Proc. 5th IEEE Workshop Applicat. and Services in Wireless Netw. (ASWN 2005)* [Online], Paris, France, June 2005, pp. 147–152. Available:
<http://w3.antd.nist.gov/pubs/aswn05.pdf>
- [CoSI05] Jeremy Colandairaj, William Scanlon, and George Irwin, "Understanding wireless networked control systems through simulation," *IEE Comput. & Control Eng. J.* [Online], vol. 16, no. 2, Apr.–May 2005, pp. 26–31. Available:
http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1454282
- [DALP97] R. Dalke, R. Achatz, Y. Lo, P. Papazian, and G. Hufford, "Measurement and analysis of man-made noise in VHF and UHF bands," in *1997 Proc. Wireless Commun. Conf.* [Online], Boulder, CO, Aug. 11–13, 1997, pp. 229–233. Available:
http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=622284

- [Embe09] Ember Corporation, “ZigBee Software - EmberZNet PRO,” Available: http://www.ember.com/products_zigbee_software.html
- [FeWK09] Ken Ferens, Lily Woo, and Witold Kinsner, “Performance of ZigBee networks in the presence of broadband electromagnetic noise,” in *IEEE 2009 Proc. Can. Conf. Electr. and Comput. Eng. (CCECE 2009)*, St. John’s, NL, May 3-6, 2009, pp. 407-410. Available: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=5090164
- [Free05a] *13193EVK Evaluation Kit: User’s Guide*, rev. 1.0, Document no. 802154EVKUG [Online], Freescale Semiconductor, June 2005. Available: http://www.freescale.com/files/rf_if/doc/user_guide/802154EVKUG.pdf
- [Free05b] A. Asmussen, R. Rodriguez, L. Roshak, *MC1319x Range Performance*, rev. 1.2, Document no. AN2902 [Online], Application Note, Freescale Semiconductor, Oct. 2005. Available: http://www.freescale.com/files/rf_if/doc/app_note/AN2902.pdf
- [Free07] *13192EVK Evaluation Kit: User’s Guide*, rev. 1.3, Document no. 802154EVKUG [Online], Freescale Semiconductor, June 2007. Available: http://www.freescale.com/files/rf_if/doc/user_guide/802154EVKUG.pdf
- [Free08a] *MC1319x/MC1320x/MC1321x Demonstration Operation: Running SMAC Based Demonstration Applications*, rev. 1.2, Document no. AN3231 [Online], Application Note, Freescale Semiconductor, Mar. 2008. Available: http://www.freescale.com/files/rf_if/doc/app_note/AN3231.pdf
- [Free08b] *Simple Media Access Controller (SMAC): User’s Guide*, rev. 1.5, Document no. SMACRM [Online], Freescale Semiconductor, Mar. 2008. Available: http://www.freescale.com/files/rf_if/doc/user_guide/SMACRM.pdf
- [FuLa05] Peter Fuhr, and Robert Lau, “Mesh radio network performance in cargo containers,” *Sensors Online* [Online], Mar. 1, 2005. Available: <http://www.sensormag.com/sensors/content/printContentPopup.jsp?id=187126>
- [GaCL03] J. B. Gao, Yinhe Cao, and Jae-Min Lee, “Principal component analysis of $1/f^\alpha$ noise,” *Phys. Lett. A* [Online], vol. 314, no. 5–6, Aug. 11, 2003, pp. 392–400. Available: http://www.gao.ece.ufl.edu/gao_pca_03.pdf
- [GCTH07] Jianbo Gao, Yinhe Cao, Wen-wen Tung, and Jing Hu, *Multiscale Analysis of Complex Time Series: Integration of Chaos and Random Fractal Theory, and Beyond*, Hoboken, New Jersey: John Wiley & Sons, Inc., 2007, 352 pp., ISBN 0471654701, 9780471654704.
- [GMHG94] John C. Gallant, Ian D. Moore, Michael F. Hutchinson, and Paul Gessler, “Estimating fractal dimension of profiles: A comparison of methods,” *J. Math. Geol.* [Online], Netherlands: Springer, vol. 26, no. 4, May 1994, pp. 455–481. Available: <http://www.springerlink.com/content/k828rw2361g76125/fulltext.pdf>

- [GoCR04] Nada Golmie, David Cypher, and Olivier Rebala, "Performance evaluation of low rate WPANs for medical applications," in *IEEE Mil. Commun. Conf. (MILCOM) 2004* [Online], vol. 2, Oct. 31–Nov. 3, 2004, pp. 927–933. Available: http://w3.antd.nist.gov/pubs/milcom04_golmie.pdf
- [GoCR05] Nada Golmie, David Cypher, and Olivier Rebala, "Performance analysis of low rate wireless technologies for medical applications," *Computer Communications* [Online], Elsevier, vol. 28, no. 10, June 2005, pp. 1255–1275. Available: http://w3.antd.nist.gov/pubs/com04_golmie.pdf
- [Golm04] Nada Golmie, "Coexistence in unlicensed bands: Challenges and solutions," Tutorial #3, *IEEE 802 Plenary Meeting* [Online], Portland OR, July 13, 2004. Available: http://www.ieee802.org/802_tutorials/04-July/802CoexistenceTutorialJuly04a.pdf
- [Golm06] Nada Golmie, "Coexistence in Wireless Networks: Challenges and System-Level Solutions in the Unlicensed Bands," United Kingdom, UK: Cambridge University Press, 2006, 144 pp.
- [GoMo01] Nada Golmie, and Frederic Mouveaux, "Interference in the 2.4 GHz ISM band: Impact on the Bluetooth access control performance," in *IEEE Proc. of the 18th Int. Conf. Commun. (ICC'01) 2001* [Online], vol. 8, Helsinki, Finland, June 11–14, 2001, pp. 2540–2545. Available: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=936608 or <http://www.antd.nist.gov/pubs/golmieicc.pdf>
- [GrKi94] W. Grieder, and W. Kinsner, "Speech segmentation by variance fractal dimension," in *IEEE 1994 Proc. Can. Conf. Electr. and Comput. Eng. (CCECE 1994)* [Online], vol. 2, Halifax, NS, Sept. 25–28, 1994, pp. 481–485. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=405793
- [HaSt06] Mats Skogholt Hansen, and Stig Støa, "Practical evaluation of IEEE 802.15.4/ ZigBee medical sensor networks," Master's thesis [Online], Norwegian University of Science and Technology, June 2006. Available: http://www.diva-portal.org/diva/getDocument?urn_nbn_no_ntnu_diva-1283-1__fulltext.pdf
- [HoKE07] M. Howlader, C.J. Kiger, and P.D. Ewing, "Coexistence assessment of industrial wireless protocols in the nuclear facility environment," *United States Nuclear Regulatory Commission (U.S.NRC)*, NUREG/CR-6939 [Online], July 2007. Available: <http://www.nrc.gov/reading-rm/doc-collections/nuregs/contract/cr6939/cr6939.pdf>
- [Holl04] Eric Holland, "Understanding your wireless options," *Intelligent Systems, Sensor Magazine* [Online], 2004. Available: <http://archives.sensorsmag.com/articles/1204/41/>
- [HoRo06] Cindy Hood, and Dennis Roberson, "Management of interference in wireless networks," Guest Editorial, *Int. J. Netw. Manage.* [Online], Wiley InterScience, vol. 16, no. 2, Mar./Apr. 2006, pp. 85–87. Available: <http://www3.interscience.wiley.com/cgi-bin/fulltext/112474056/PDFSTART>

- [HuSi01] Gurdeep S. Hura, and Mukesh Singhal, *Data and Computer Communications: Networking and Internetworking*, CRC Press, 2001, 1168 pp.
- [IEC02] CISPR-25, *Radio Disturbance Characteristics for the Protection of Receivers Used on board Vehicles, Boats, and on Devices—Limits and Methods of Measurement*, 2nd ed., 2002-08, Geneva, Switzerland: International Electrotechnical Commission, 123 pp. (Courtesy of Vansco Electronics Ltd.).
- [IEEE03] IEEE Std 802.15.4-2003, *IEEE Standard for Information Technology-Telecommunications and information exchange between systems-Local and metropolitan area networks specific requirements—Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs)*.
- [ISO98] ISO 14982:1998(E), *Agricultural and Forestry Machinery—Electromagnetic Compatibility—Test Methods and Acceptance Criteria*, 1st ed., 1998-07-01, Geneva, Switzerland: International Organization for Standardization, 37 pp. (Courtesy of Vansco Electronics Ltd.).
- [JeBS02] Michel C. Jeruchim, Philip Balaban, and K. Sam Shanmugan, *Simulation of Communication Systems: Modeling, Methodology, and Techniques*, Information Technology Series: Transmission, Processing, and Storage, 2nd ed., Springer, 2000, 907 pp., ISBN 0306462672, 9780306462672.
- [KaEr97] A. Kamerman, and N. Erkocevic, “Microwave oven interference on wireless LANs operating in the 2.4GHz ISM band,” in *8th IEEE Int. Symp. Personal, Indoor and Mobile Radio Commun.* [Online], 1997 ‘Waves of the Year 2000’ (PIMRC’97), vol. 3, Sept. 1–4, 1997, pp. 1221–1227. Available: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=627080
- [KaWi05] Holger Karl, and Andreas Willig, *Protocols and Architectures for Wireless Sensor Networks*, West Sussex, UK: John Wiley and Sons, Ltd, 2005, 497 pp., ISBN 0470095105, 9780470095102.
- [Kins02] Witold Kinsner, “Appendix A: Noise in electronic circuits,” chap. 4 in *Microprocessor Interfacing Course Notes*, Winnipeg, MB, CA: Department of Electrical and Computer Engineering, University of Manitoba, 1985–2002, 62 pp.
- [Kins04] Witold Kinsner, *Fractal and Chaos Engineering Course Notes*, Winnipeg, MB: Department of Electrical and Computer Engineering, University of Manitoba, 2004.
- [Kins05] W. Kinsner, “A unified approach to fractal dimensions,” in *4th IEEE Conf. Cognitive Informat. (ICCI 2005)*, Aug. 8–10, 2005, pp. 58–72.
- [Kosk06] Bart Kosko, *Noise*, Viking, 2006, 252 pp., ISBN 0670034959, 9780670034956.

- [LeCL07] Hyung Jung Lee, Alberto Cerpa, and Philip Levis, "Improving wireless simulation through noise modeling," in *Proc. 6th Int. Conf. Inf. Process. Sensor Netw. 2007 (IPSN '07)*, New York, NY: ACM, Cambridge, MA, Apr. 25–27, 2007, pp. 21–30.
- [LiBa07] Xuedong Liang, and Ilanko Balasingham, "Performance analysis of the IEEE 802.15.4 based ECG monitoring network," in *Proc. 7th Int. Conf. Wireless and Optical Commun. (IASTED)* [Online], Montreal, May 30–June 1, 2007. Available: <http://old-www.cwi.nl/projects/credo/publications/565-087.pdf>
- [LuKR04] Gang Lu, Bhaskar Krishnamachari, and Cauligi S. Raghavendra, "Performance evaluation of the IEEE 802.15.4 MAC for low-rate low-power wireless networks," in *IEEE Int. Conf. Performance, Comput., and Commun.* [Online], Apr. 2004, pp. 701–706. Available: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?tp=&arnumber=1395158
- [Masi07] Ken Masica, "Recommended practices guide for securing ZigBee wireless networks in process control system environments (Draft)," Control Systems Security Program (CSSP), *United States Computer Emergency Readiness Team (US-CERT)*, Department of Homeland Security, [Online], Apr. 2007. Available: <http://csrp.inl.gov/Documents/Securing%20ZigBee%20Wireless%20Networks%20in%20Process%20Control%20System%20Environments.pdf> through http://www.us-cert.gov/control_systems/csarchive.html
- [Maxi03] *Buffer Amplifiers Solve VCO Problems*, Application note 2019 [Online], Maxim Integrated Products, May 11, 2003. Available: http://www.maxim-ic.com/appnotes.cfm/appnote_number/2019
- [Micr09] Microchip, "MiWi Stack," ZigBee Protocol Connectivity Solutions, [Online]. Available: http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=2113¶m=en520414
- [Mini07] Minicircuits, Selected Modules' DataSheet [Online]. Available: <http://minicircuits.com/>
- [Mini08] *How VCO Parameters Affect Each Other*, Application Note AN95-005 [Online], Mini-Circuits. Available: <http://www.minicircuits.com/pages/pdfs/an95005.pdf>
- [Morr04] Robert Morrow, *Wireless Network Coexistence*, 1st ed., New York, NY: McGraw-Hill Professional, Aug. 2004, 444 pp.
- [MoSL01] Michael Moore, Stephen Smith, and Kang Lee, "The next step – Wireless IEEE 1451 smart sensor networks," Intelligent Systems, *Sensor Magazine* [Online], 2001. Available: <http://archives.sensormag.com/articles/0901/35/main.shtml>
- [MuTe08] R. Muşaloiu-E., and A. Terzis, "Minimising the effect of WiFi interference in 802.15.4 wireless sensor networks," *Int. J. Sensor Networks* [Online], vol. 3, no. 1, 2008, pp.43–54. Available: <http://www.cs.jhu.edu/~razvanm/ijsn2008interference.pdf>

- [PeJS92] H.-O. Peitgen, H. Jürgens, and D. Saupe, *Chaos and Fractals: New Frontiers of Science*, New York: Springer-Verlag, 1992, 984 pp.
- [Pemi] *USB HCS08-HCS12 Multilink Rev B Technical Summary*, ver. 1.01, Document no. PE3336 [Online], P& E Microcomputer Systems. Available:
http://www.pemicro.com/downloads/main_downloads_temp/200912140200312362033/PE3336-%20Technical%20summary%20for%20USB-ML-12%20%20Rev%20B.pdf
- [PeSa88] H. Peitgen, and D. Saupe, eds. *The Science of Fractal Images*, Springer-Verlag New York, Inc., 1988, pp. 312, ISBN:0-387-96608-0.
- [Poor03] Robert Poor, “Wireless mesh networks,” *Intelligent Systems, Sensors Magazine* [Online], Questex Media Group, Inc., Feb. 2003. Available:
<http://www.sensormag.com/articles/0203/38/main.shtml>
- [PRML06] M. Petrova, J. Riihijarvi, P. Mahonen, and S. LaBell, “Performance study of IEEE 802.15.4 using measurements and simulations,” in *IEEE Wireless Commun. and Netw. Conf. (WCNC) 2006* [Online], vol. 1, pp. 487–492. Available:
http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=1683512
- [Rodr05] R. Rodriguez, *MC1319x Coexistence*, rev. 1.2, Document no. AN2935 [Online], Application Note, Freescale Semiconductor, July 2005. Available:
http://www.freescale.com/files/rf_if/doc/app_note/AN2935.pdf
- [Rupe04] Rupert Goodwins, “Rupert Goodwins' Diary,” blog [Online], Oct. 15, 2004. Available:
<http://community.zdnet.co.uk/blog/0,1000000567,10003969o-2000331777b,00.htm>
- [SABJ07] Khaled Shuaib, Maryam Alnuaimi, Mohamed Boulmalf, Imad Jawhar, Farag Sallabi, and Abderrahmane Lakas, “Performance evaluation of IEEE 802.15.4: Experimental and simulation results,” *J. Commun.* [Online], Oulu, FI: Academy Publisher, vol. 2, no. 4, June 2007, pp. 29–37. Available:
<http://www.academypublisher.com/jcm/vol02/no04/jcm02042937.pdf>
- [SBSL05] K. Shuaib, M. Boulmalf, F. Sallabi, and A. Lakas, “Performance analysis: Co-existence of IEEE 802.11g with Bluetooth,” in *2nd IFIP Int. Conf. Wireless and Optical Commun. Netw. (WOCN) 2005* [Online], Mar. 6–8, 2005, pp. 40–44. Available:
http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?tp=&arnumber=1435985
- [SBSL06] K. Shuaib, M. Boulmalf, F. Sallabi, and A. Lakas, “Co-existence of ZigBee and WLAN - A performance study,” in *2006 IFIP Int. Conf. Wireless and Optical Commun. Netw.* [Online], Apr. 11–13, 2006, 5 pp. Available:
http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?tp=&arnumber=1666534
- [Schw08] John Schwartz, “Demystifying ZigBee and 802.15.4,” *Electronic Component News* [Online], Jan. 2008. Available: <http://www.ecnmag.com/Article-Demystifying-ZigBee-and-802-15-4.aspx>

- [Seyb05] John S. Seybold, *Introduction to RF Propagation*, Hoboken, NJ: John Wiley & Sons, Inc., 2005, 330 pp., ISBN 0471655961, 9780471655961.
- [SGMP07] Qingshan Shan, I. A. Glover, P.J. Moore, I. Portugues, R.J. Watson, and R. Rutherford, "Performance of ZigBee in electricity supply substations," in *Int. Conf. Wireless Commun., Netw. and Mobile Comput. (WiCom) 2007* [Online], Sept. 21–25, 2007, pp. 3871–3874. Available: <http://ieeexplore.ieee.org/Xplore/login.jsp?url=/stamp/stamp.jsp?arnumber=4340732&isnumber=4339775>
- [ShPK07] Soo Young Shin, Hong Seong Park, and Wook Hyun Kwon, "Mutual interference analysis of IEEE 802.15.4 and IEEE 802.11b," *Comp. Netw.*, vol. 51, no. 12, Aug. 2007, pp. 3338–3353.
- [SiGr05] A. Sikora, and V.F. Groza, "Coexistence of IEEE802.15.4 with other systems in the 2.4 GHz-ISM-band," in *Proc. IEEE Instrum. and Meas. Technol. Conf. (IMTC) 2005* [Online], vol. 3, May 16–19, 2005, pp. 1786–1791. Available: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=1604479
- [Simm06] J. Simmons, IQTRead: .IQT conversion software, Tektronix, May 2006.
- [Sire07a] *AN-101 - PLL Evaluation Board Application Note* [Online], rev. C, Sirenza [cited Jan. 12, 2007]. Available: [http://www.sirenza.com/documents/products/2599/PLL400-2450A\(Y\)_Evaluation_Board_AN.pdf](http://www.sirenza.com/documents/products/2599/PLL400-2450A(Y)_Evaluation_Board_AN.pdf)
- [Sire07b] *PLL400-2450A Datasheet - PLL Product Specification* [Online], rev. P3, Sirenza [cited Jan. 12, 2007]. Available: [http://www.sirenza.com/documents/products/3659/PLL400-2450A\(Y\)_Datasheet.pdf](http://www.sirenza.com/documents/products/3659/PLL400-2450A(Y)_Datasheet.pdf)
- [Sire07c] *AN-113 - PLL Programming Note* [Online], rev. A, Sirenza [cited Jan. 12, 2007]. Available: [http://www.sirenza.com/documents/products/2601/PLL400-2450A\(Y\)_Application_Note.pdf](http://www.sirenza.com/documents/products/2601/PLL400-2450A(Y)_Application_Note.pdf)
- [Sire07d] PLL Controller Software [Online], Sirenza Microdevices. Available: [http://www.sirenza.com/documents/products/2600/PLL400-2450A\(Y\)_Controller_Software.zip](http://www.sirenza.com/documents/products/2600/PLL400-2450A(Y)_Controller_Software.zip)
- [SPCK07] Soo Young Shin, Hong Seong Park, Sunghyun Choi, and Wook Hyun Kwon, "Packet error rate analysis of ZigBee under WLAN and Bluetooth interferences," *IEEE Trans. Wireless Commun.* [Online], vol. 6, no. 8, Aug. 2007, pp. 2825–2830. Available: <http://ieeexplore.ieee.org/Xplore/login.jsp?url=/stamp/stamp.jsp?arnumber=4290024&isnumber=4290009>
- [Stew03] L. Stewart, "Troubleshoot to get the most out of black-box magic," *Construction Equipment* [Online], Jan. 2003. Available: <http://www.constructionequipment.com/article/CA471330.html>

- [Tek01] *Fundamentals of Interference in Mobile Networks*, Application Note [Online], Tektronix, 2001. Available: http://www.tek.com/Measurement/App_Notes/2G_14758/eng/2GW_14758_0.pdf
- [Tek08a] *Real-Time Spectrum Analyzers: RSA3303A, RSA3308, WCA230A, and WCA280A, Data sheet* [Online], Tektronix. Available: http://www2.tek.com/cmsreplive/psrep/13394/37W_18864_2_2008.05.21.14.32.42_13394_EN.pdf
- [Tek08b] RSAVu offline analysis software, ver. 3.31.000, Tektronix, [obtained as an attachment in an email correspondence with Freescale technical support].
- [Tesa07] *CC2420 Data Sheet: 2.4 GHz IEEE 802.15.4/ZigBee-Ready RF Transceiver*, rev. B, Chipcon SWRS041B [online], Texas Instruments, Mar. 19, 2007, 89 pp. Available: <http://focus.ti.com/lit/ds/symlink/cc2420.pdf>
- [Turc97] Donald Lawson Turcotte, *Fractals and Chaos in Geology and Geophysics*, 2nd ed., Cambridge, UK: Cambridge University Press, 1997, 398 pp., ISBN 0-521-56733-5.
- [Tuzl02] Vyacheslav Petrovich Tuzlukov, *Signal Processing Noise*, The Electrical Engineering and Applied Signal Processing Series, CRC Press, 2002, 663 pp., ISBN 0849310253, 9780849310256.
- [TWHH03] W.E. Turner, H.J. Wilkes, E.D. Hennen, and J. Hackworth, "Report of investigation: Fatal powered haulage accident," [Online], United States Department of Labor, Mine Safety and Health Administration, Nov. 19, 2003. Available: <http://www.msha.gov/FATALS/2003/FTL03M26.HTM>
- [Vasi05] Gabriel Vasilescu, *Electronic Noise and Interfering Signals: Principles and Applications*, Signals and Communication Technology Series, Heidelberg, Germany: Springer-Verlag, 2005, 709 pp., ISBN 3540407413, 9783540407416.
- [WiFi08] Wi-Fi Alliance, Available: <http://www.wi-fi.org/>
- [WISE] DACI (Design, Applications, Communications, and Implementations) Research Group, Wireless Sensor Network Simulator (WISENES) [Online], Tampere University of Technology. Available: http://www.tkt.cs.tut.fi/research/daci/daci_wsn_wisenes.html
- [WKFD06] Lily Woo, Witold Kinsner, Ken Ferens, and Jeff Diamond, "Modelling and emulation of multifractal noise in performance evaluation of mesh networks," in *IEEE 2006 Proc. Can. Conf. Electr. and Comput. Eng. (CCECE 2006)* [Online], Ottawa, ON, May 7–10, 2006, pp. 1646–1651. Available: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=4055071
- [WKFD07] Lily Woo, Witold Kinsner, Ken Ferens, and Jeff Diamond, "Performance results and analysis of ZigBee networks in the presence of multifractal noise," in *IEEE 2007 Proc. Can. Conf. Electr. and Comput. Eng. (CCECE 2007)* [Online], Vancouver, BC, Apr. 22–25, 2007, pp. 924–927. Available: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=4232895

- [WoCu01] A. Woo, and D.E. Culler, "A transmission control scheme for media access in sensor networks," in *Proc. 7th Annual Int. Conf. Mobile Comput. and Netw. (MobiCom '01)* [Online], New York, NY: ACM Press, Rome, Italy, 2001, pp. 221–235. Available: <http://doi.acm.org/10.1145/381677.381699>
- [WoFK10] Lily Woo, Ken Ferens, and Witold Kinsner "Reliability of ZigBee networks under broadband electromagnetic noise interference," in *IEEE 2010 Proc. Can. Conf. Electr. and Comput. Eng. (CCECE 2010)*, Calgary, AB, May 2-5, 2010 (accepted).
- [WoKF08] Lily Woo, Witold Kinsner, and Ken Ferens, "An analysis of captured industrial vehicular noise signals for ZigBee communications," in *IEEE 2008 Proc. Can. Conf. Electr. and Comput. Eng. (CCECE 2008)* [Online], Niagara Falls, ON, May 4–7, 2008, pp. 1423–1428. Available: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=4564776
- [Worn96] Gregory W. Wornell, *Signal Processing with Fractals: A Wavelet-Based Approach*, 1st ed., Upper Saddle River, NJ: Prentice Hall Press, 1996, 177 pp.
- [WPKF07] Lily Woo, Michael Potter, Witold Kinsner, and Ken Ferens, "Analysis of Modulated Monofractal Noise," in *IEEE 2007 Proc. Can. Conf. Electr. and Comput. Eng. (CCECE 2007)*, Vancouver, BC, Apr. 22–25, 2007, pp. 884–887. Available: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=4232885
- [ZhLe06] J. Zheng, and M. J. Lee, "A comprehensive performance study of IEEE 802.15.4," chap. 4 in *Sensor Network Operations*, IEEE Press, Wiley Interscience, 2006, pp. 218–237, ISBN 0-471-71976-5.
- [ZiA104] *ZigBee Specification*, ZigBee Document 053474r06 [Online], version 1.0, ZigBee Alliance, 378 pp., Dec. 2004. Available: <http://www.zigbee.org>
- [ZiA108] ZigBee Alliance, "FAQ," Available: <http://www.zigbee.org/About/FAQ/tabid/192/Default.aspx>

APPENDIX A: CODE

The code listed below can be found in the attached CD-ROM under the Appendices\Code directory.

A.1 Monofractal Noise Generation, Written in MATLAB

A.2 PER Monitoring Software, Written in C

A.3 GPIB-USB Trace Data Acquisition Software, Written in C#

A.4 Compute Power Within Specified Bandwidth from Trace, Written in MATLAB

A.5 Captured Noise Analysis: Spectrogram, Written in MATLAB

A.6 Captured Noise Preprocessing: Convert .IQT to Time Series, Written in MATLAB

A.7 Captured Noise Analysis: VFDT, Written in MATLAB

A.8 Modulated Monofractal Noise Analyses, Written in MATLAB

APPENDIX B: DATA

The data listed below can be found in the attached CD-ROM under the Appendices\Data directory.

B.1 Capture Noise Data from Starter Source of Tractors

B.2 Baseband Monofractal Noise Sequence or Datapoints

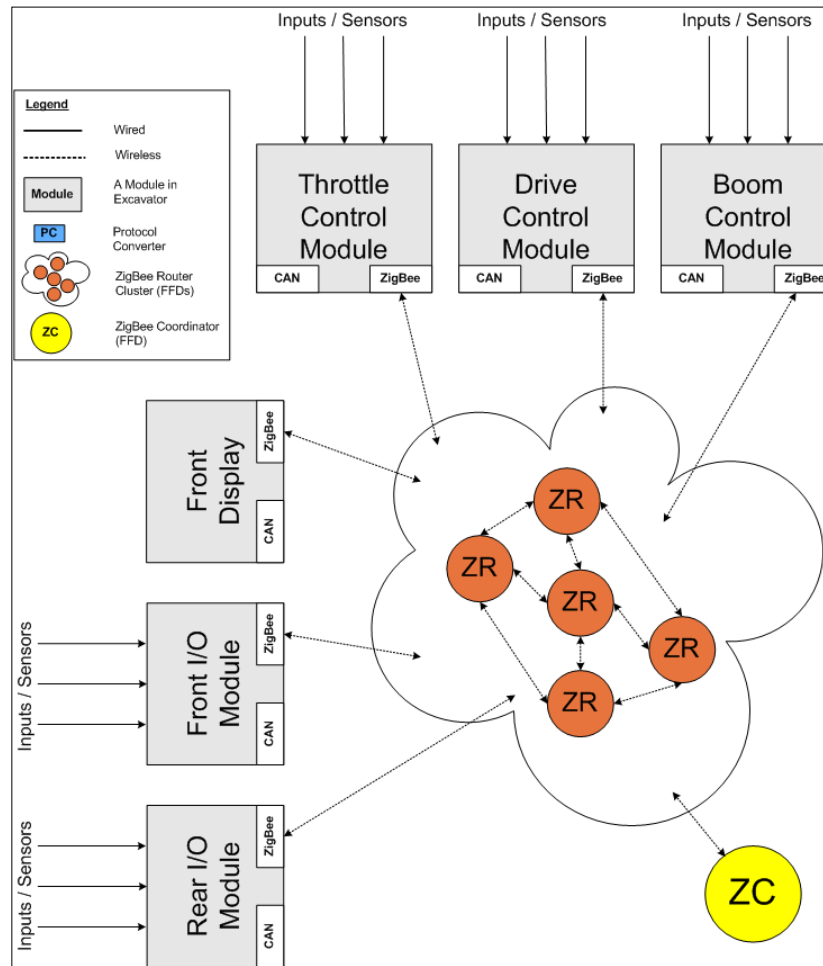
B.3 Packet Error Rate Data

B.4 Trace Data: ZigBee Spectrum and Noise Spectrum

APPENDIX C: SUPPLEMENTARY TECHNICAL DOCUMENTATIONS

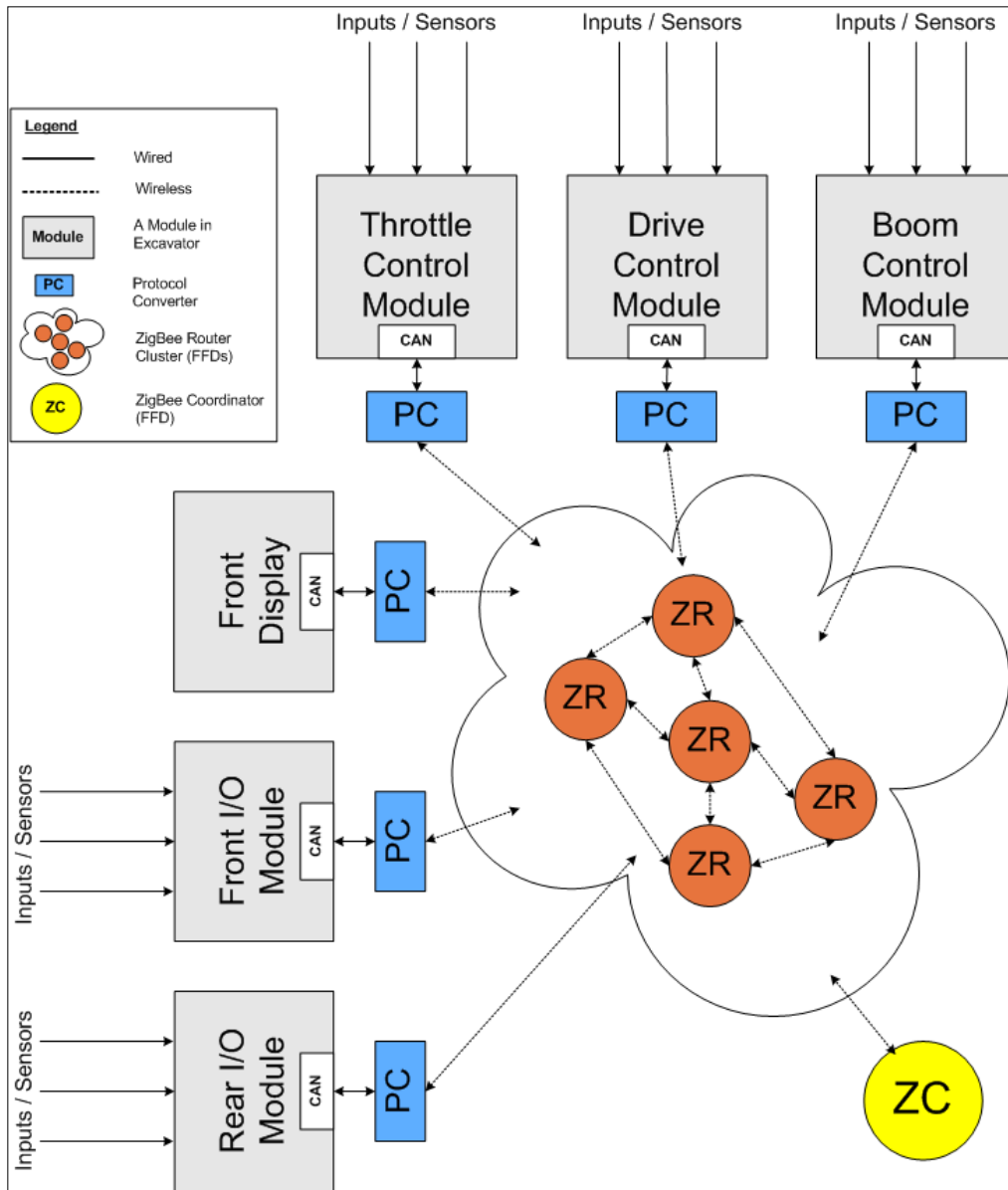
C.1 Wireless Design Alternatives

DESIGN 01a: Single Sensors, Single Modules



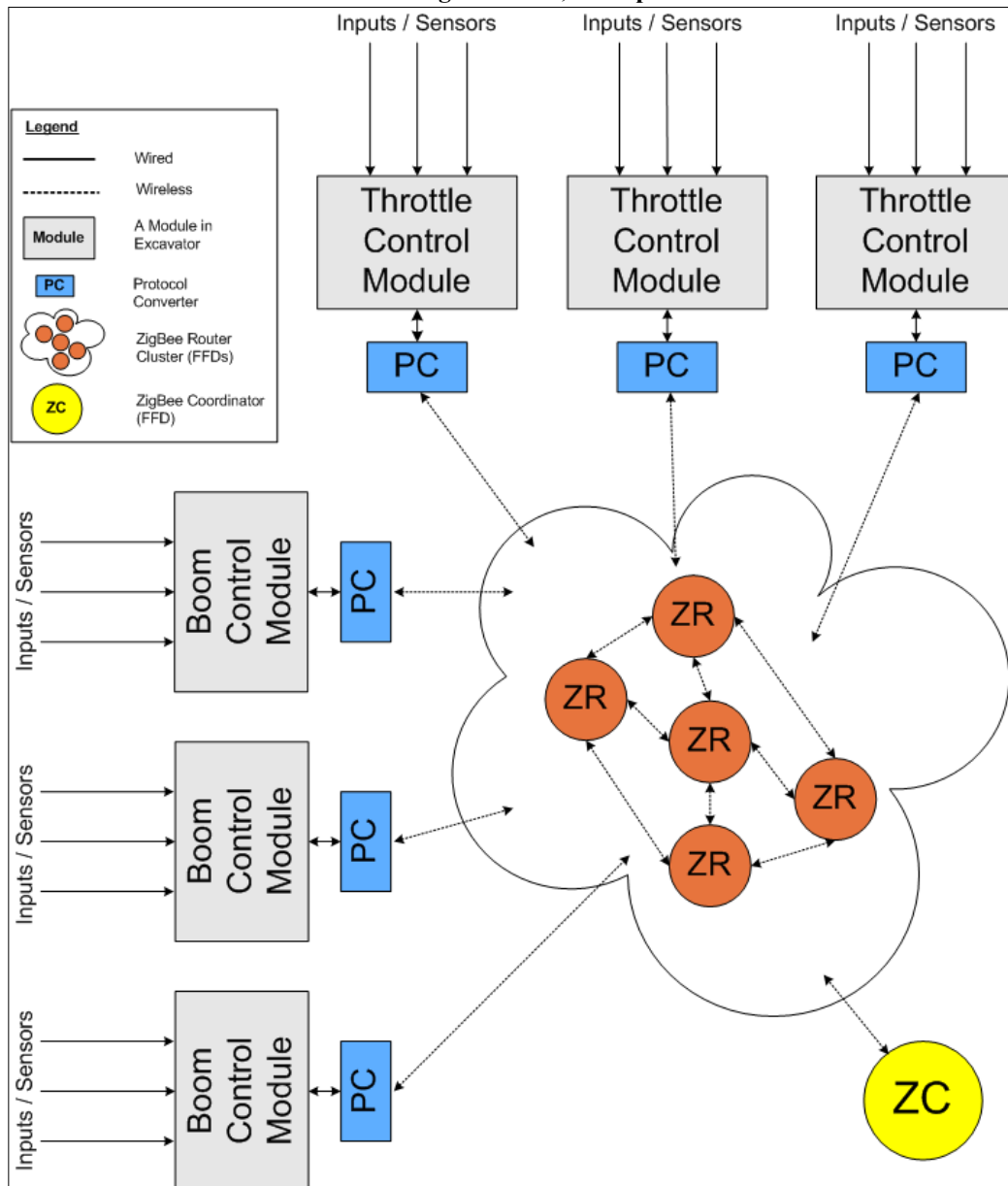
Pros	Cons
<ul style="list-style-type: none"> ▪ Wired Harness Removed (default) ▪ Partial Module Reuse ▪ Partial Healing since operator is aware of damaged module/sensors ▪ Easily add/remove modules ▪ System remains the same as before in terms of the number of sensors/modules, except that main harness are removed ▪ Capable of switching between CAN and ZigBee communication mode 	<ul style="list-style-type: none"> ▪ Reduced Reliability <ul style="list-style-type: none"> ○ No redundancy on both modules and sensors ▪ Operator intervention required for replacement of damaged module

DESIGN 01b: Single Sensors, Single Modules



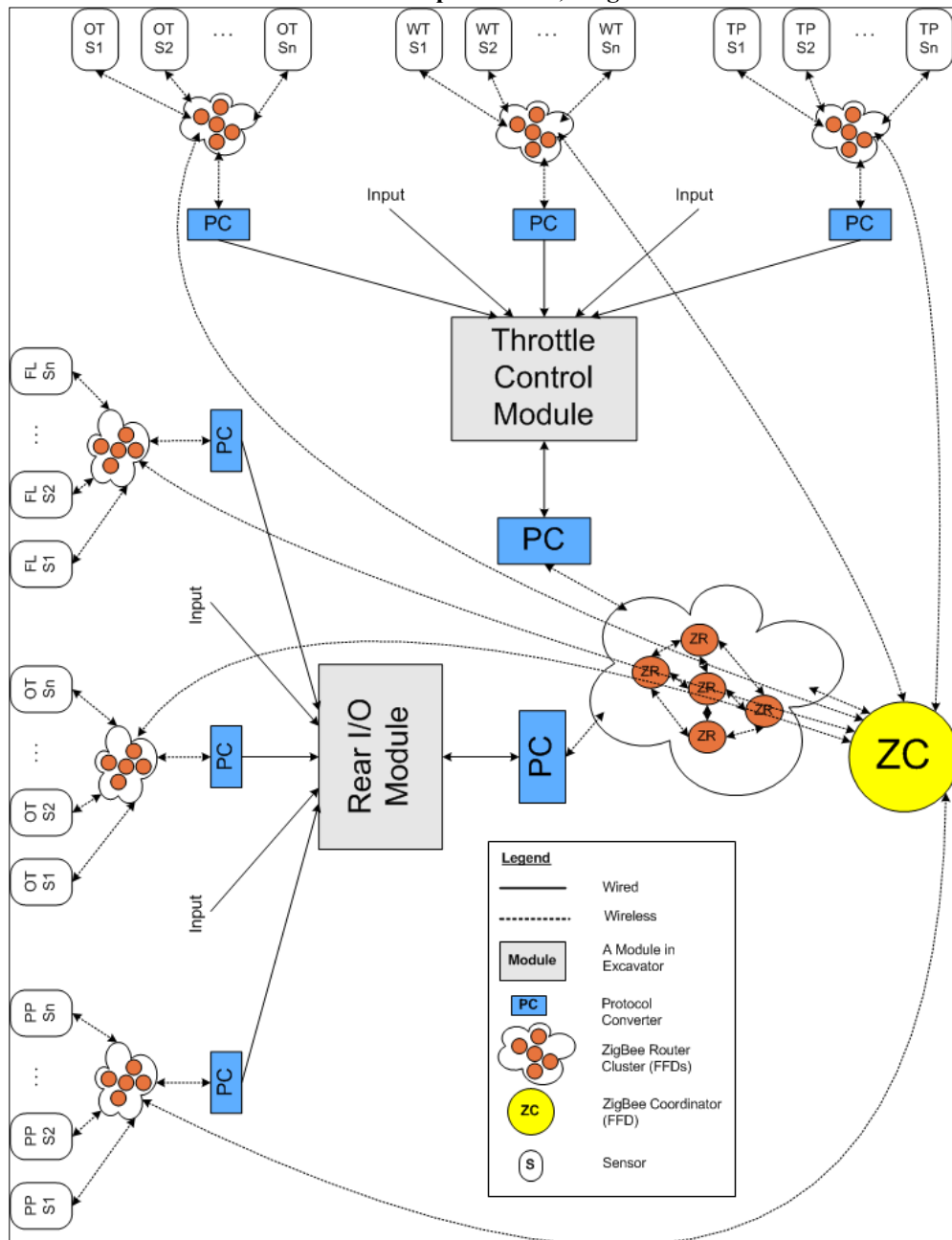
Pros	Cons
<ul style="list-style-type: none"> ▪ Wired Harness Removed (default) <ul style="list-style-type: none"> ○ No longer concern about connectors being faulty since they are no longer there ▪ Total Module Reuse ▪ Partial Healing since operator is aware of damaged module/sensors ▪ Easily add/remove modules ▪ System remains the same as before in terms of the number of sensors/modules, except that main harness are removed 	<ul style="list-style-type: none"> ▪ Reduced Reliability <ul style="list-style-type: none"> ○ No redundancy on both modules and sensors ▪ Operator intervention required for replacement of damaged module

DESIGN 02: Single Sensors, Multiple Modules



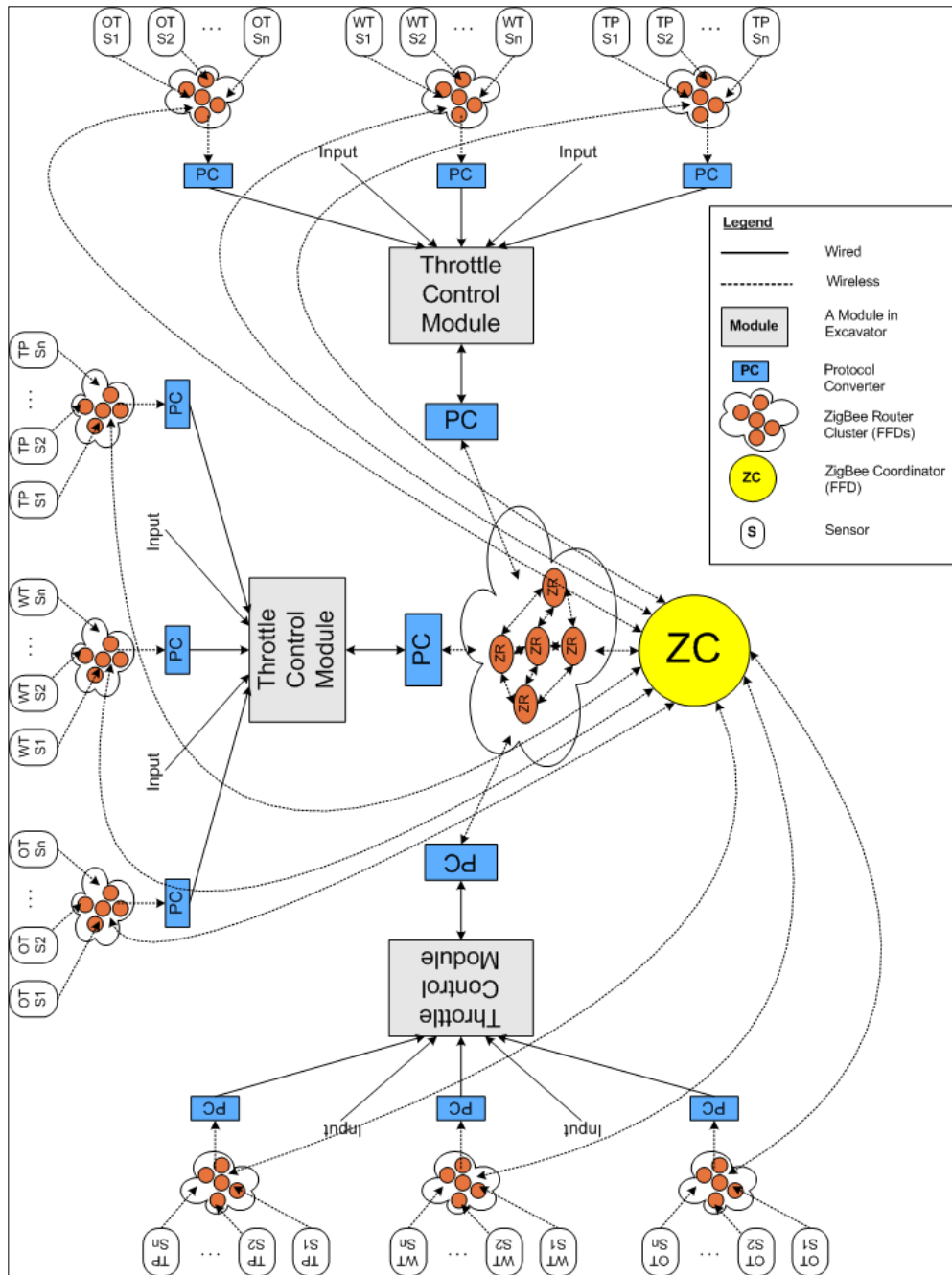
Pros	Cons
<ul style="list-style-type: none"> ▪ Wired Harness Removed (default) ▪ Module Reuse (default) ▪ Modules Self Healing <ul style="list-style-type: none"> ○ homogeneous redundancy ○ higher reliability ▪ Easily add/remove modules ▪ Redundant sensors not required ▪ No operator intervention 	<ul style="list-style-type: none"> ▪ May not have enough room to support multiple modules and its respective ▪ No Sensors Self Healing <ul style="list-style-type: none"> ○ No redundancy on sensors <p>Note: hence, even when only one sensor goes down, entire module is considered down.</p>

DESIGN 03: Multiple Sensors, Single Modules



Pros	Cons
<ul style="list-style-type: none"> ▪ Wired Harness Removed (default) ▪ Module Reuse (default) ▪ Sensors Self Healing <ul style="list-style-type: none"> ○ homogeneous redundancy ○ higher reliability ▪ Easily add/remove sensors/modules ▪ Partial Module Healing since operator is aware of damaged module 	<ul style="list-style-type: none"> ▪ May not have enough room to support multiple sensors of a kind (per module) ▪ No Modules Self-Healing <ul style="list-style-type: none"> ○ Operator intervention required for replacement of damaged module ○ No module redundancy ○ Single point of failure

DESIGN 04: Multiple Sensors, Multiple Modules



Pros	Cons
<ul style="list-style-type: none"> ▪ Wired Harness Removed (default) ▪ Module Reuse (default) ▪ Both Sensors and Modules Self Healing <ul style="list-style-type: none"> ○ homogeneous redundancy ○ higher reliability ▪ Easily add/remove modules/sensors ▪ No operator intervention 	<ul style="list-style-type: none"> ▪ May not have enough room to support multiple modules and multiple sensors

C.2 Voltage Regulator Circuits

Circuit to supply voltage to V_{cont} of Attenuator

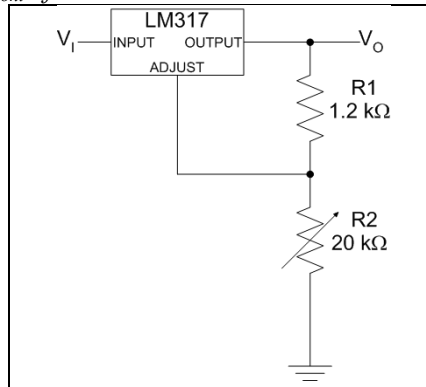


Fig. 1: 1.2V-20V Regulator with Minimum Program Current (after [2])

Circuit to supply voltage to buffer module, specifically V^+ of Amplifier and V_{cont} of Attenuator

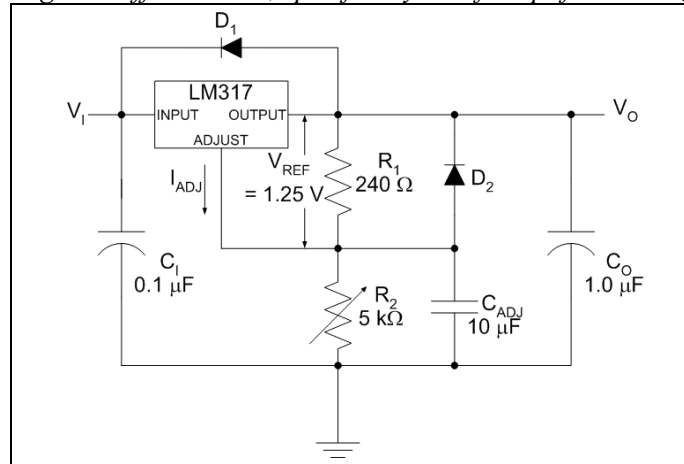


Fig. 2: Adjustable Regulator with Improved Ripple Rejection (after [2])

Details of the specific components used

Circuit components from Figure 1	Component used (available from ECE Tech Shop)
R_1	1.2 k Ω resistor
R_2	20 k Ω potentiometer
V_1	+15 VDC power supply
3-terminal adjustable regulator	LM317
Circuit components from Figure 2	Component used (available from ECE Tech Shop)
D_1 and D_2	1N4005
R_1	Two 120 Ω resistor (placed in series to make 240 Ω)
R_2	5 k Ω potentiometer (CT20P502)
C_1	0.1 μ F ceramic disc
C_{ADJ}	10 μ F electrolytic
C_O	1.0 μ F solid tantalum
V_1	+15 VDC power supply
3-terminal adjustable regulator	LM317

Purpose for the components in Figure 2

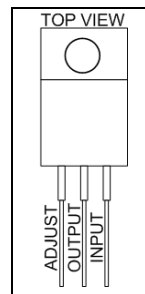
Component Type	Specific component	Purpose
Capacitor	C ₁ , Input terminal bypass capacitor	Recommended for the case where the regulator is located far from the power-supply filter capacitors.
	C _{ADJ} , Adjustment terminal bypass capacitor	To improve ripple rejection.
	C _O , Output terminal bypass capacitor	To swamp the effect of excessive ringing and to insure stability.
Protection Diode	D ₁	D ₁ prevents C _O from discharging into output of the regulator if input is shorted to ground, by providing low impedance discharge path for C _O .
	D ₂	D ₂ discharges C _{ADJ} if input/output is shorted to ground, by providing a low impedance discharge path for C _{ADJ} .

Calculation of V_O in Figure 2

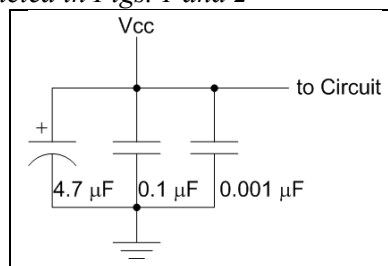
$$V_O = V_{REF} (1 + R_2/R_1) + I_{ADJ} (R_2);$$

where V_{REF} for LM317 = 1.25V, and I_{ADJ} is negligible in most applications because it is typically 50µA.

LM317 legs configurations

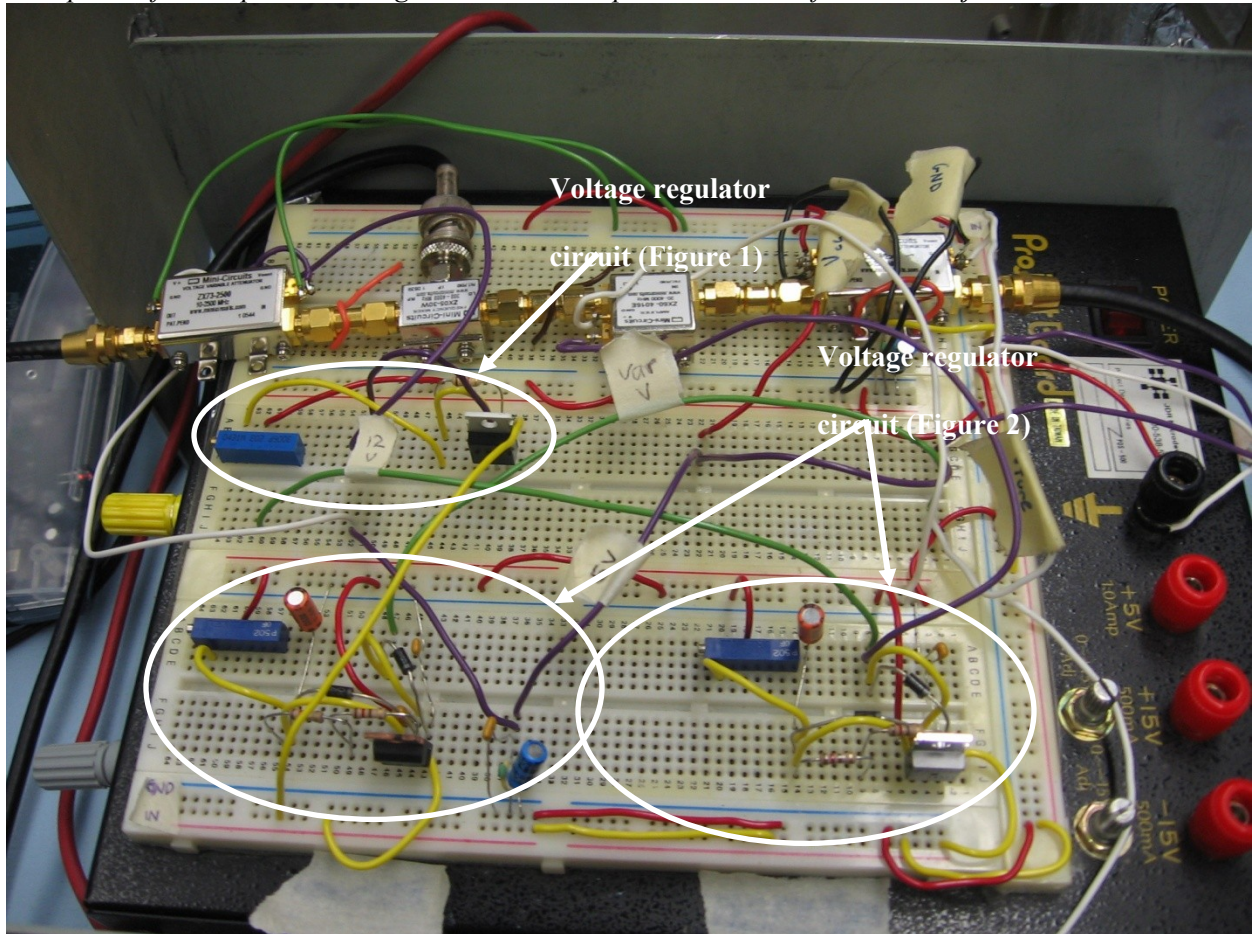


Bypass capacitors connected to power source V_{cc} and to ground, where V_{cc} is the power supply supplying V_I of the circuits depicted in Figs. 1 and 2



Bypass capacitors values	Type used (available from ECE Tech Shop)
4.7µF	Electrolytic (with voltage rating = 25 V)
0.1µF	Ceramic disc (capacitor code 103)
0.01µF	Ceramic disc (capacitor code 104)
V _{cc}	Power supply (+15 VDC)

Snapshot of the implemented regulator circuits to power modules of the noise injection circuit



References

- [1] National Semiconductor, "LM117/LM317A/LM317: 3-Terminal Adjustable Regulator," June 2006. (Online) <http://cache.national.com/ds/LM/LM117.pdf>
- [2] Texas Instruments, "LM317: 3-Terminal Adjustable Regulator," Rev. U, April 2008. (Online) <http://focus.ti.com/lit/ds/symlink/lm317.pdf>
- [3] Digi-Key Corporation, "Specification for CT20P502 Part," Accessed July 2008. (Online) <http://search.digikey.com/scripts/DkSearch/dksus.dll?Detail?name=CT20P502-ND>
- [4] Ross, Kevin, "Basic Circuits – Bypass Capacitors," Accessed July 2008. (Online) <http://www.seattlerobotics.org/encoder/jun97/basics.html>
- [5] van Roon, Tony, "Capacitors Tutorial," Accessed August 2008. (Online) <http://www.uoguelph.ca/~antoon/gadgets/caps/caps.html>

C.3 Carrier Frequency Drifting Issue: Problem (Cause and Effect) and Solutions

Question: Is drifting a problem?

Answer: Yes, for our intended experimentation. When the carrier frequency drifts, the noise spectrum drifts too, causing the noise power that affects the ZigBee spectrum to be inconsistent, should the drift occur during the transmission of packets. The recorded Packet Error Rate (PER) and Signal-to-Noise Ratio (SNR) will not be accurate.

Experiments Conducted

Since VCO is an input-output (I/O) module, the output is affected by its input to some degree. Table 1 shows the experiments conducted (with the mentioned improvements) with respect to the input of the VCO to check if the VCO output frequency still drifts. The experiment is simply to view the output frequency at the RF OUT pin of the VCO (i.e. testing the VCO unit in isolation) using a spectrum analyzer and measuring the output frequency at the start and at the end of the experiment and then taking the difference to obtain how much the output frequency had drifted. The output voltage value of the +12V power supply (to V_{cc}), the voltage regulator (to V_{tune}), and the +15V power supply are also recorded at the start and at the end of the experiment to monitor if any of them drifted. These recorded value are accurate only to the resolution (0.01V for range between 6V and 60V) of the digital multimeter (Fluke 112) used.

Table 1: Improvements made at the input side of the VCO to counter the output frequency drift issue and their respective motivation.

Improvements Done towards Reducing the VCO Output Frequency Drift	Purpose
Improved voltage regulator circuit that supplies to V_{tune} of VCO, however still using breadboard's power supply to supply to V_{in} of voltage regulator	To ensure voltage regulator's output voltage is stable.
Changed power supply (instead of using the voltage regulator to a fixed +12V power supply unit) for V_{cc} of VCO	To ensure a stable V_{cc} .
Changed power supply (instead of breadboard's to a fixed +15V power supply unit) for V_{in} of voltage regulator	To ensure voltage regulator's output voltage is stable.
Mounted VCO module on PCB board. Grounded the entire board (except for V_{cc} and V_{tune} traces) to the ground of the +12V power supply. Placed bypass capacitors on V_{cc} trace, close to V_{cc} pin of VCO.	To ensure proper grounding, bypassing and that module is stable in its position.
Used a voltage reference (LM336BLP-2-5, a 2.5V reference) to supply V_{tune}	To ensure a stable V_{tune} .
Used batteries as power supplies to V_{cc} and V_{tune}	To ensure a stable V_{tune} and V_{cc} .

Observation

The VCO output frequency still drifts.

Parameters identified that would have impacted the VCO output frequency to drift.

Table 2: The causes and suggested counter measures for VCO output frequency drift or instability

	VCO output frequency will vary with:	Suggested counter measures
From the input side of VCO	Supply voltage changes (the effect is termed as frequency pushing)	- Use a more stable power source. - Use bypass capacitors.
	Tuning voltage changes (relates to tuning sensitivity, e.g. 40 MHz/V)	- Use a more stable power source. - Use low noise components for the adjustable voltage regulator. - Use bypass and output capacitors.
From the output side of VCO	Load impedance changes (the effect is termed as frequency pulling)	- Place buffer amplifier between VCO and its load. - Place a pad / attenuator between VCO and its load.
The VCO circuitry	EM and RF coupled directly or indirectly to the VCO	- Mount all components on a PCB board. - Use shielded wire connections. - Shortened wires wherever possible.
	Mechanical vibration	- Secure the VCO.
	Temperature (ambient)	- Cover VCO with a box to keep temperature balanced.

Note:

- (i) The VCO will need to be warmed up for about 30 minutes before use.
- (ii) Tuning voltage is applied to adjust the VCO to output a sinusoid with the desired frequency.
- (iii) Supply voltage is required to power up the VCO for use.
- (iv) Grounding, shielding and routing of traces near the VCO can also affect the VCO's operation.
- (v) Covering the VCO with a box to keep the temperature balanced may not be sufficient as a proper temperature management requires temperature sensing and control.
- (vi) Placing a buffer amplifier between the VCO output and its load provides output isolation from the load. In other words, the buffer amplifier acts as an impedance transformer to isolate the output of the VCO from any impedance variability.
- (vii) Placing a pad/attenuator in between the VCO output and its load also provides isolation between the two circuit stages.
- (viii) Things to consider when choosing bypass capacitors: proper capacitor value, dielectric material, geometry, and location of the capacitor in relation to the integrated circuit/device/module.
- (ix) Passive components (e.g. resistors, capacitors and inductors) within the voltage regulator/power supply circuit can affect the output value of the circuit. For example, the value of capacitance changes when the molecules of the capacitor change. This accounts for the short term drift in the V_O of the voltage regulator to the VCO's V_{tune} , which in turn affects the short-term output frequency drift of the VCO.
- (x) Long-term drift is caused by temperature changes, whereas short-term drift can be caused by EM/RF, people nearby the circuit, and switching circuits.

Conclusion and Discussion

Due to the many parameters that affect the output frequency of the VCO (since the VCO is basically a sensitive device due to its open-loop architecture), rather than trying to correct each of these parameters, a better architecture to consider is a phase-locked loop (closed-loop) which helps to stabilize the VCO output frequency. As a safety measure, it is always desirable to have at least 10 - 12 dB of isolation between the output of the PLL and its load to isolate the PLL from load impedance variation. This we can achieve with either a buffer amplifier or a pad (6 dB) placed between the PLL and its load (in our case, the mixer). We need a complete PLL frequency synthesizer which includes a PLL IC, a VCO, a reference frequency source, and a loop filter on an evaluation board with SMA output to replace our current frequency source, which is the VCO. In order to achieve higher accuracy and stability, a digital PLL is preferable. As an advantage, no tuning (which is time-consuming) is required to set the PLL to the desired output frequency. However, some programming is required to obtain the desired output frequency.

References

- [1] Maxim Integrated Products, "Application Note 2019: Buffer Amplifiers Solve VCO Problems," May 2003. (Online) http://www.maxim-ic.com/appnotes.cfm/appnote_number/2019
- [2] Mini-Circuits, "AN-95-005: How VCO Parameters Affect Each Other," (Online) <http://www.minicircuits.com/appnote/an95005.pdf>
- [3] Mini-Circuits, "AN-95-006: Optimizing VCO/PLL Evaluations & PLL Synthesizer Designs," (Online) <http://www.minicircuits.com/appnote/an95006.pdf>

C.4 Uploading Datapoints onto AWG using HyperTerminal

1. Objective

This documentation outlines the basic configuration for setting up the Windows® HyperTerminal program (on a Windows O.S.) to talk to the Agilent 33120A Arbitrary Waveform Generator (AWG) over RS-232 interface, the causes for errors encountered when trying to send more than 8 data points through a connection between the HyperTerminal and the AWG and their respective solutions, and last but not least a step-by-step procedure on downloading data points onto the volatile memory of AWG.

2. Preliminaries

*NOTE: The entire document is written for use with the Agilent 33120A 15MHz Function /Arbitrary Waveform Generator and the HyperTerminal program on machine using Windows® operating system. Use this particular AWG manual (33120-90104 manual, Publication Number 33120-90006, Edition 6, March 2002) when referring to the recommended pages. The manual can be downloaded from:
http://www.home.agilent.com/upload/cmc_upload/All/6C0633120A_USERSGUIDE_ENGLISH.pdf*

- An arbitrary waveform generator is used to accept and store data points of a desired signal waveform either on volatile or non-volatile memory so that the generator can generate and display the input waveform.
- By default, the AWG is set to use GPIB interface.
- Agilent 33120A Function Generator/Arbitrary Waveform Generator:
 - Baud Rate : 9600 (selectable)
 - Data bits : 8 (selectable; use this setting to download arbitrary waveforms)
 - Parity : None (selectable; use this setting to download arbitrary waveforms)
 - Start bits : 1 (fixed)
 - Stop bits : 2 (fixed)
 - Flow control : DTR/DSR (fixed)
- The AWG can only store up to four user-defined arbitrary waveforms.
- Refer to page 139 of the AWG manual for minimum and maximum output frequency.
- The number of data points required for downloading is at least 8 points (maximum 16,000 points).
- Press SHIFT followed by < for quick access to error code when error indicator is turned on. Refer to pages 229-241 for the error message denoted by a specific error code.

3. Configuration for Setting Up Communication between AWG and HyperTerminal over RS-232

3.1 Choosing and Attaching the Correct RS-232 Cable

In order to use RS-232 interface, a DTE-to-DTE interface cable is required. This cable is also known as a null-modem, modem-eliminator or crossover cable. Depending on your machine's serial port configuration, either a cable with 9 pins or 25 pins connectors is needed. To be certain of the cable, check the connectivity of the pins from both ends of the cable against the diagram below (Figure 1) and with the use of a multimeter.

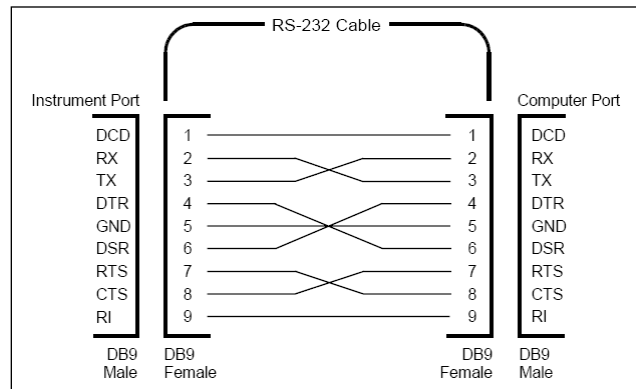


Figure 1: Typical RS-232 cable pin diagram

3.2 Setting Up the Agilent 33120A Arbitrary Waveform Generator

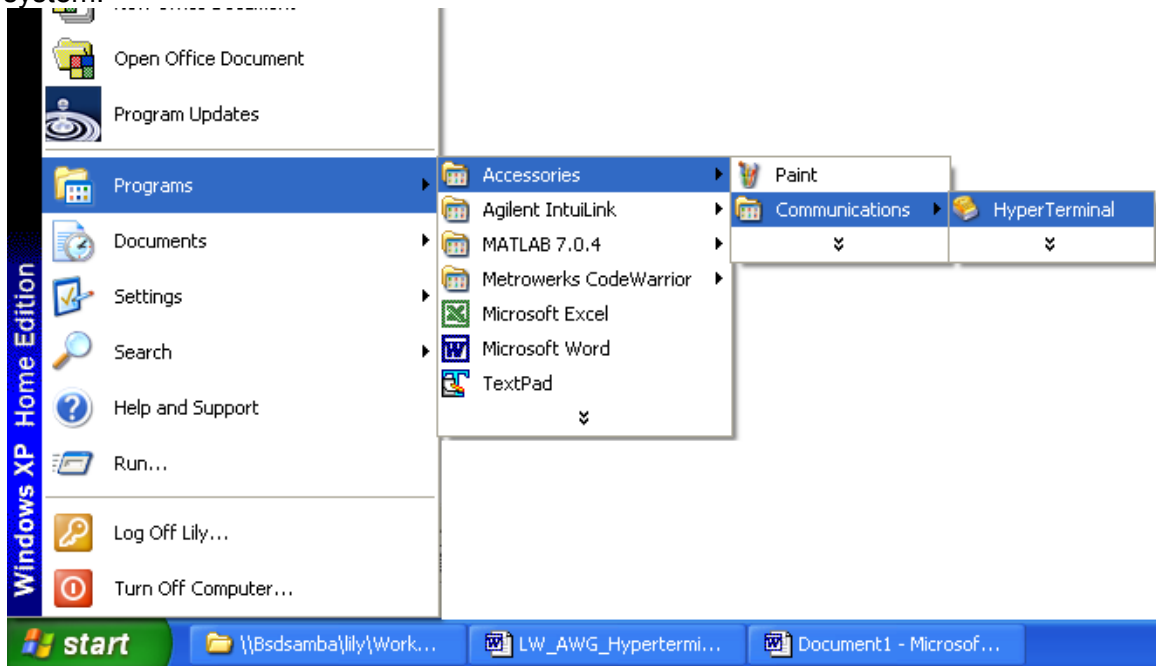
To enable the RS-232 interface:

Press SHIFT and MENU ON/OFF to turn on the menu. Press < twice to reach I/O MENU. Press ∨ once and > once to reach the INTERFACE option. Press ∨ once to reach the PARAMETER level. Use < or > to view the interface choices (GPIB or RS-232). Press ENTER to save the change and to turn off the menu. The interface selection is stored in non-volatile memory.

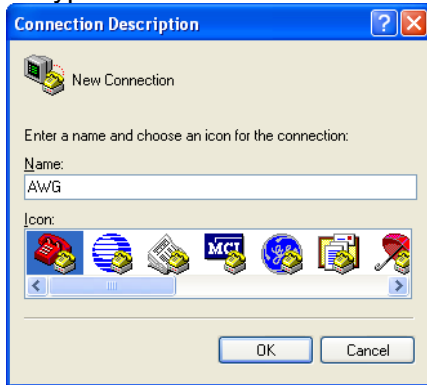
(Refer to pages 115-116, 195, 218-220 for more details)

3.3 Setting Up Windows® HyperTerminal

1. Launch HyperTerminal program. The following selection is applicable to Windows XP only (Click *Start->Programs->Accessories->Communications->HyperTerminal*). The location of HyperTerminal launch icon may differ on other versions of Windows operating system.



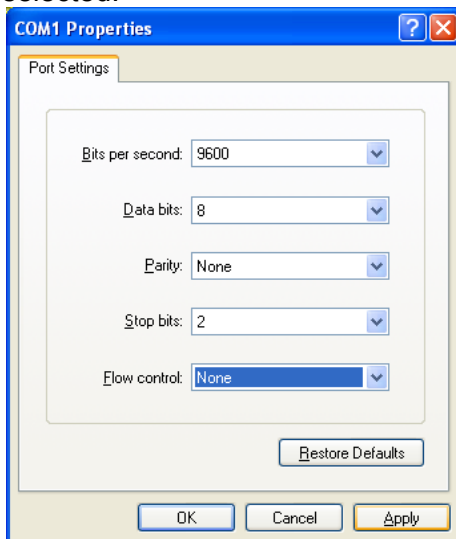
2. Type in a connection name and then click OK.

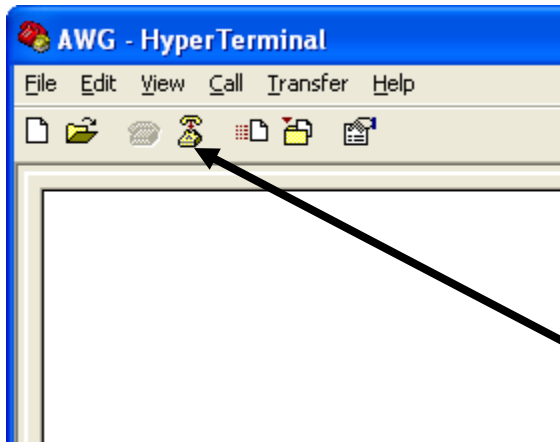


3. Select the COM port where the Null modem cable is plugged into and click OK.



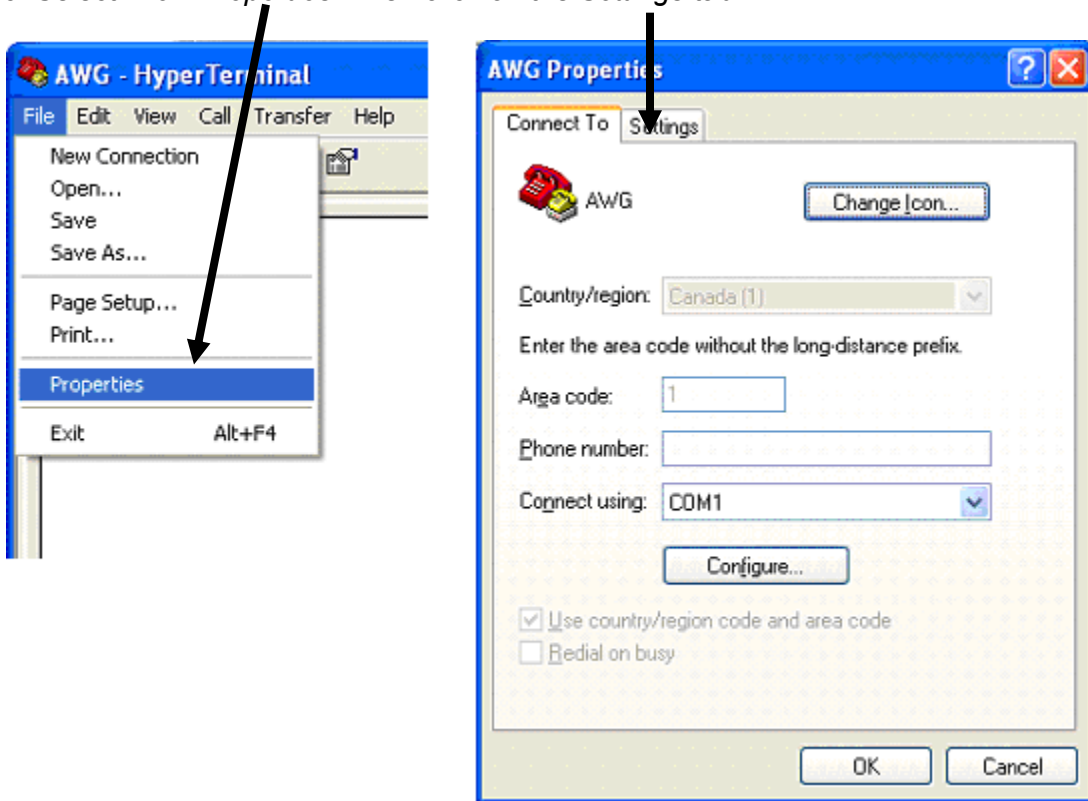
4. Input the COM port parameters and then click OK. The following values match the 33120A's default configuration. The illustration below assumes that COM1 port is selected.



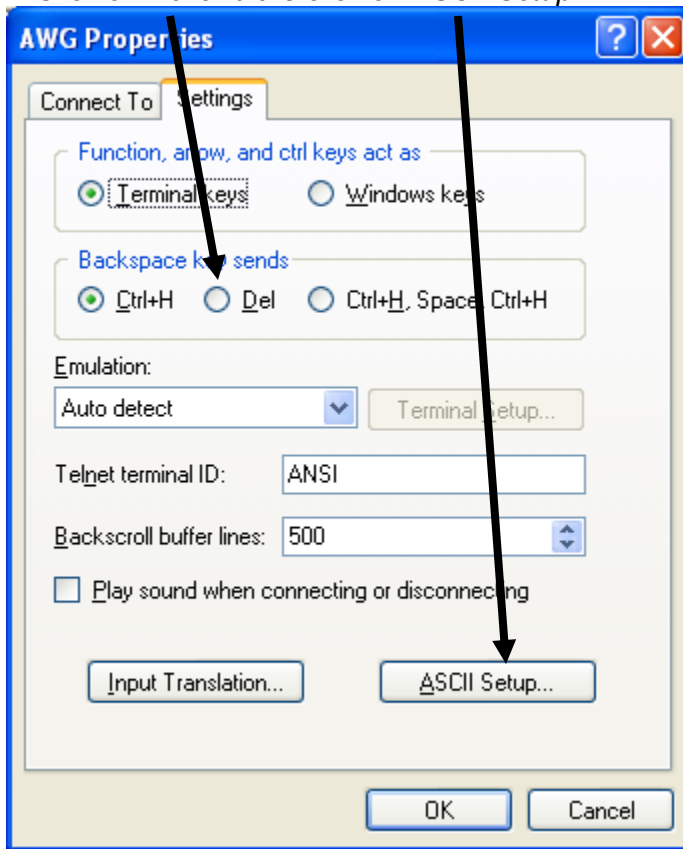


5. Click here to disconnect the connection.

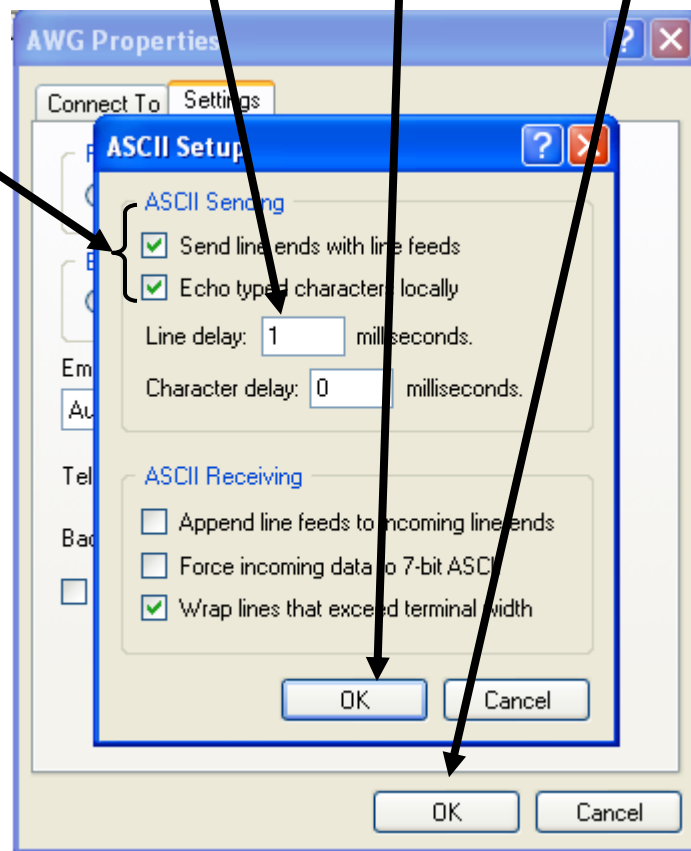
6. Select *File->Properties*. Then click on the *Settings* tab.



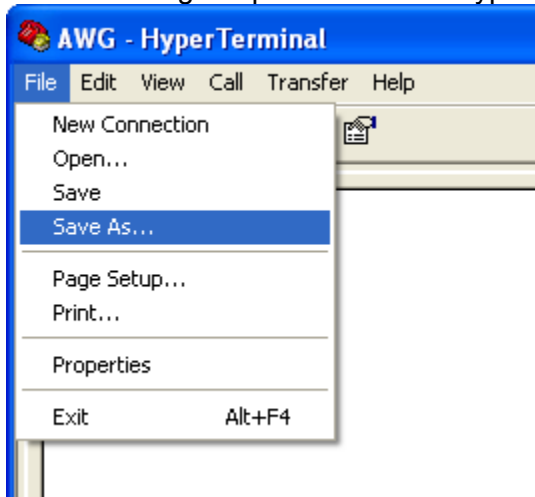
7. Click on *Del* and then click on *ASCII Setup....*



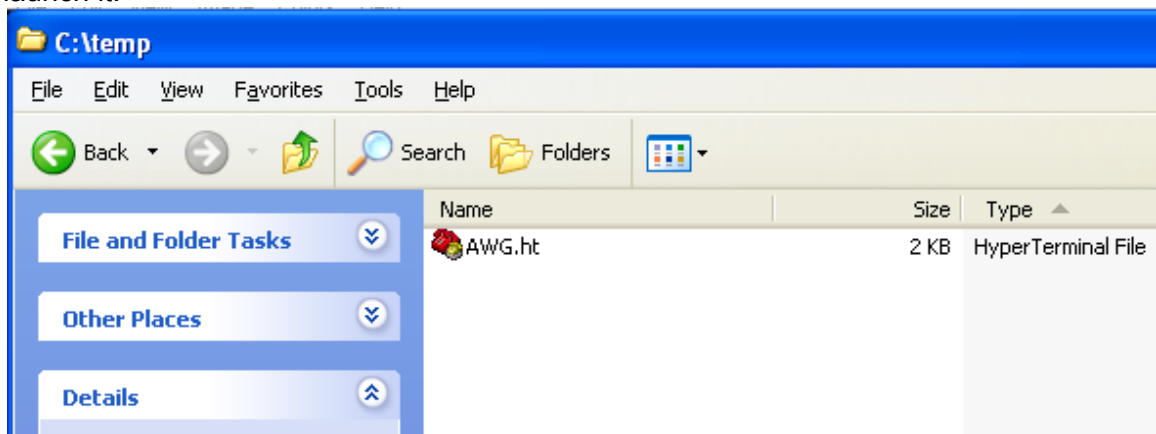
8. Check the first two boxes. Then type 1 here. After that, click OK and then click OK again.



9. Click on *File->Save As...* to save the current HyperTerminal configuration. Then click X on the far right top corner to exit HyperTerminal.



10. Each time you want to access your stored HyperTerminal settings, just locate the directory where your HyperTerminal file was stored in and double-click at its icon to launch it.



4. Some Causes for the Errors Encountered on AWG and Their Respective Solution

The AWG failed to display the input waveform when:

- (i) The data file being transferred from HyperTerminal does not end with a carriage return character; and/or
- (ii) Character delay on HyperTerminal's Setting->ASCII Setup is 0ms; and/or
- (iii) A data point consists of three or more consecutive digits of the same number. E.g. 0.44412, 0.122227 and etc.

Resolved each of the above as follows:

- (i) Added a carriage return character (by hitting the Enter or Return key on the keyboard) at the End-Of-File (EOF) of the data file.
- (ii) Reset character delay to 1ms.
- (iii) Changed the sequence of the AWG commands where SYST:REM has to come before the CLS and the RST commands.

Reasoning for each of the above solutions:

- (i) Each carriage return character at the end of a line provides a line feed for the line sent so that the AWG will process the line as one command. Should there be a missing carriage return character at the EOF, then the last command in the sequence of AWG commands will not be processed. Your request will not be performed since the AWG has not received the last command yet.
- (ii) The HyperTerminal is sending each character at a speed faster than can be handled by the AWG. This causes the 521 (i.e. input buffer overflow) error.
- (iii) Agilent technical support person suggested the sequence of AWG commands where CLS and RST come before SYST:REM in an earlier email correspondence. In addition, we had a hard time resolving this problem because the suggested sequence actually worked. However, we found out that the sequence of AWG commands only worked for as long as there was no data point with three or more consecutive digits of the same number.

Note: For the 1st cause and the 3rd cause, the errors returned to the AWG screen were 101 (Invalid Character) and 785 (Specified Waveform Does Not Exist). The second error occurred because of the first error since a complete 8-data point waveform has not been completely received.

5. A Template of a Working Sequence of SCPI (Standard Commands for Programmable Instruments) Commands for Uploading Datapoints onto the Volatile Memory of the AWG

Type the following and save it as a .txt file.

```
SYST:REM
*CLS
*RST
APPL:USER 0.5 KHZ, 2.0 VPP, 0 V
DATA VOLATILE, <floating point data points between -1 and 1, each
separated by a comma except for the very last data point entry>
FUNC:USER VOLATILE
FUNC:SHAP USER
```

Meaning of each line of command:

Line 1: Set the AWG in remote mode

Line 2: Clear the error queue.

Line 3: Reset the function generator to its default state.

Line 4: Specify the frequency, amplitude and offset of the arbitrary waveform.

Line 5: Specify the arbitrary waveform's data points to be transferred to the volatile memory of the AWG.

E.g. DATA VOLATILE,

0.0089277,0.0089494,0.008938,0.0089243,0.0089238,0.0088884,0.0088603,0.0088547

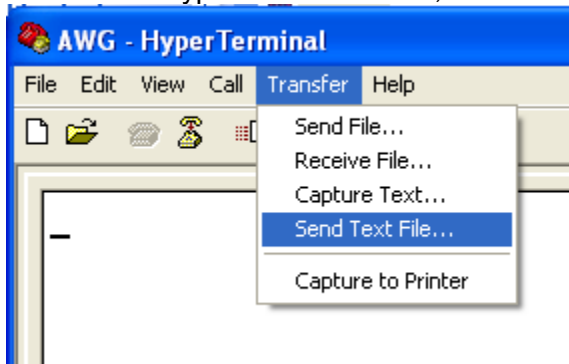
Line 6: Select the waveform currently stored in volatile memory.

Line 7: Output the currently selected arbitrary waveform on the AWG screen.

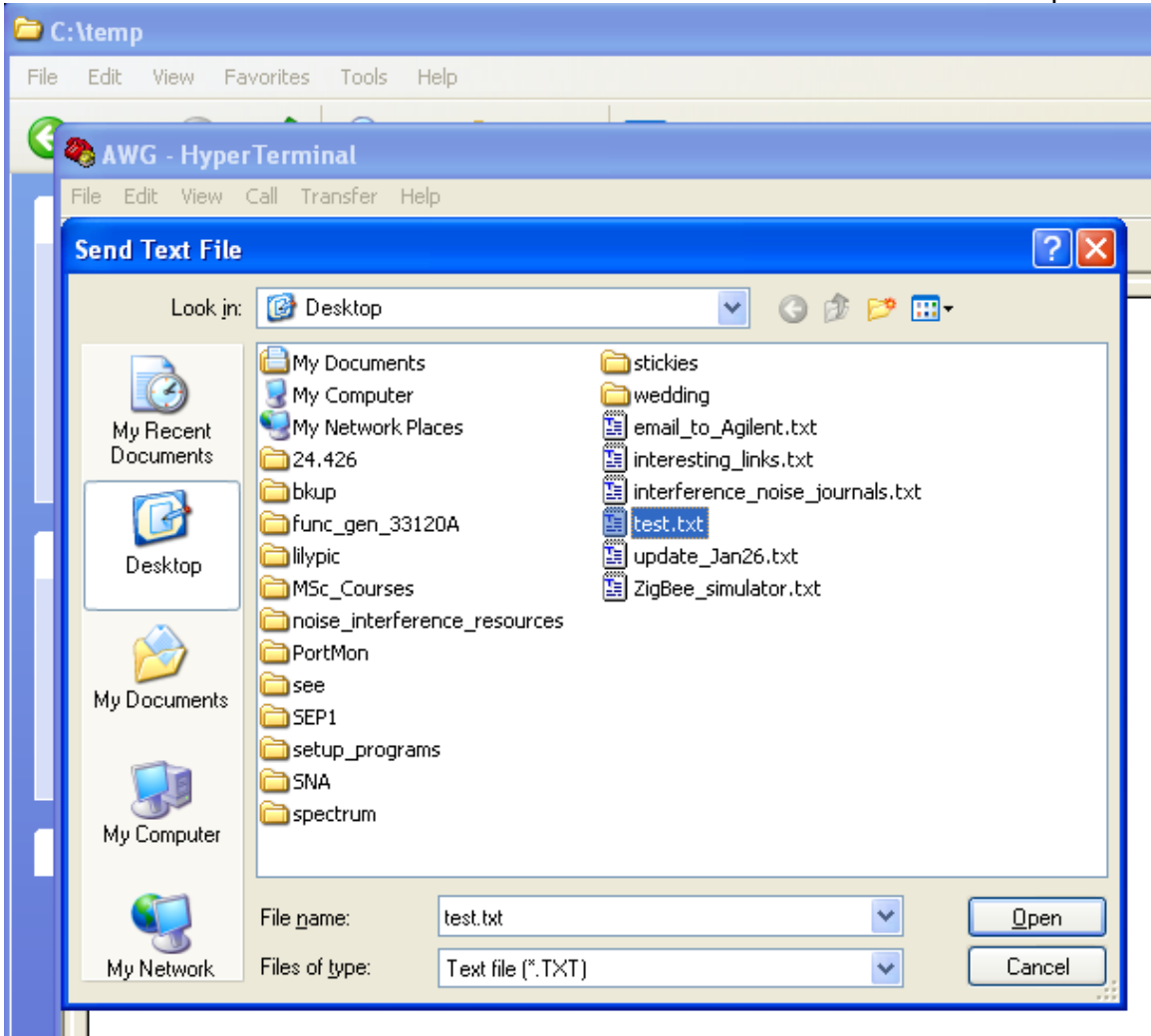
NOTE: Refer to 174-185 of the manual for Arbitrary Waveform commands.

6. Steps for AWG to Download Arbitrary Waveform

1. Make sure RS-232 cable is attached to the right serial port on the pc and the AWG.
2. Power on the AWG.
3. Open a HyperTerminal session (refer to step 10 in Section 3).
4. From the HyperTerminal menu, select *Transfer->Send Text File*

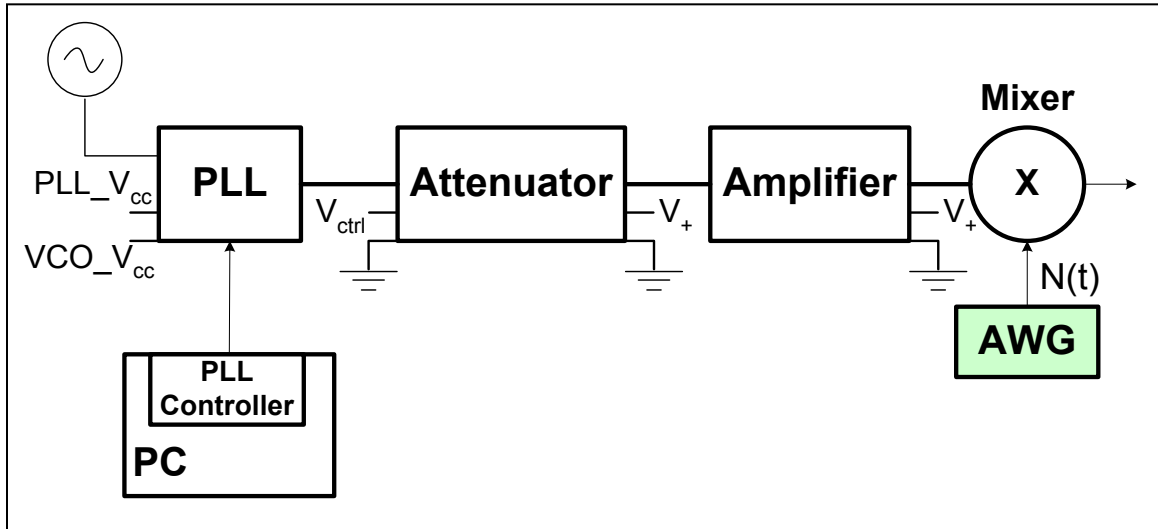


5. Select the text file that contains the commands from Section 5 and then click Open.

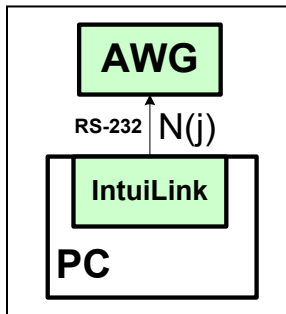


C.5 Uploading Datapoints onto AWG using IntuiLink

Setup for modulating noise signal, $N(t)$.



Using IntuiLink as a fast method for loading datapoints of a noise waveform, $N(j)$ over RS-232 onto the Arbitrary Waveform Generator, AWG.



Purpose: Main purpose of this documentation is to provide the necessary procedure for the purpose of loading an arbitrary waveform onto the AWG for storage. You may load up to 4 user-defined waveforms in its non-volatile memory.

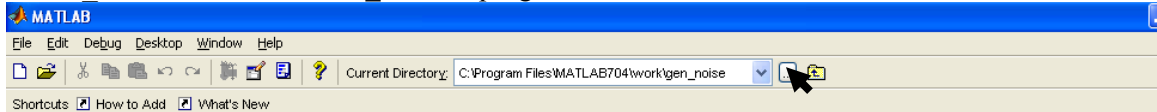
Assumption: The following procedure is based upon the assumption that *MATLAB*, *TextPad*, and *Agilent IntuiLink Waveform Editor* have been installed, and *color_noise.m* file is available on the Windows operating machine (we used *Windows XP* in all the illustrations/screenshots), to be used for the purpose of transferring datapoints to the Agilent 33120A 15 MHz Function / Arbitrary Waveform Generator via RS-232. One RS-232 serial cable, specifically a DTE-to-DTE interface (also known as null-modem, modem-eliminator, or crossover) cable with DB-9 (9 pins) female connector on each end, is required. For reference purposes, a copy of the AWG manual can be downloaded from http://www.home.agilent.com/upload/cmc_upload/All/6C0633120A_USERSGUIDE_ENGLISH.pdf.

Procedure: We can divide the process of downloading MATLAB generated datapoints onto the AWG over RS-232 using IntuiLink into two major procedures:

- (A) Preparing the colour noise text file.
- (B) Using IntuiLink to load the colour noise datapoints.

A. Preparing the Colour Noise Text File

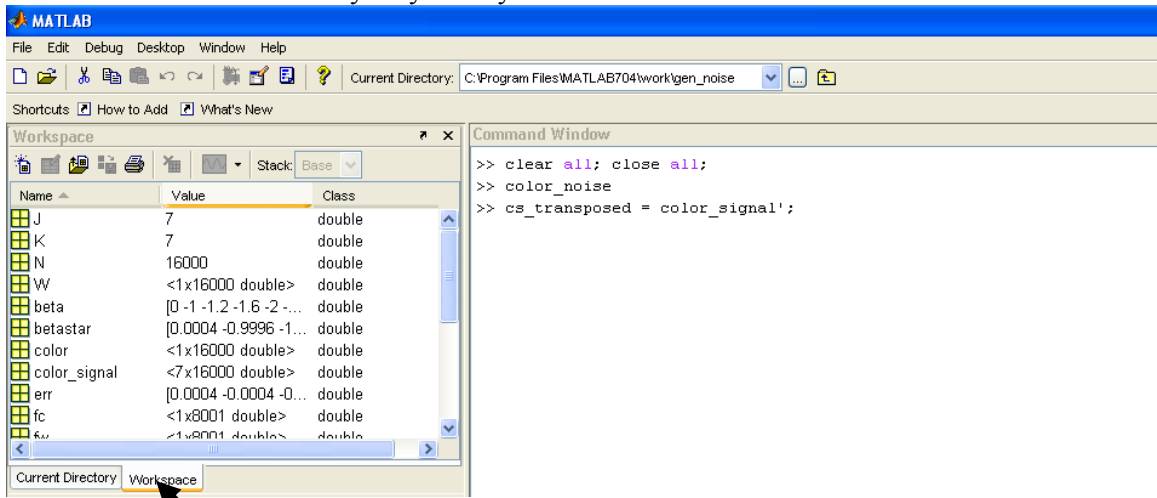
(i) Set MATLAB's Current Directory to the folder where the Spectral Filtering MATLAB code is located. Run the code by typing the program name (without the .m extension), e.g. use type `color_noise` to run the `color_noise.m` program.



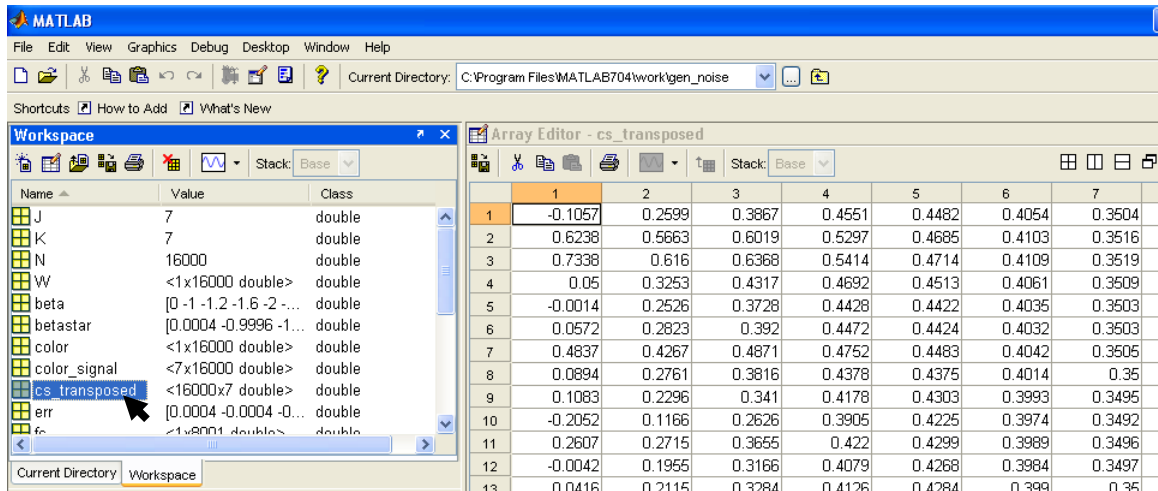
(ii) Click on the Workspace tab on the top left window of MATLAB. On the Command Window, type:

```
cs_transposed = color_signal';
```

and hit Enter or the Return key on your keyboard.

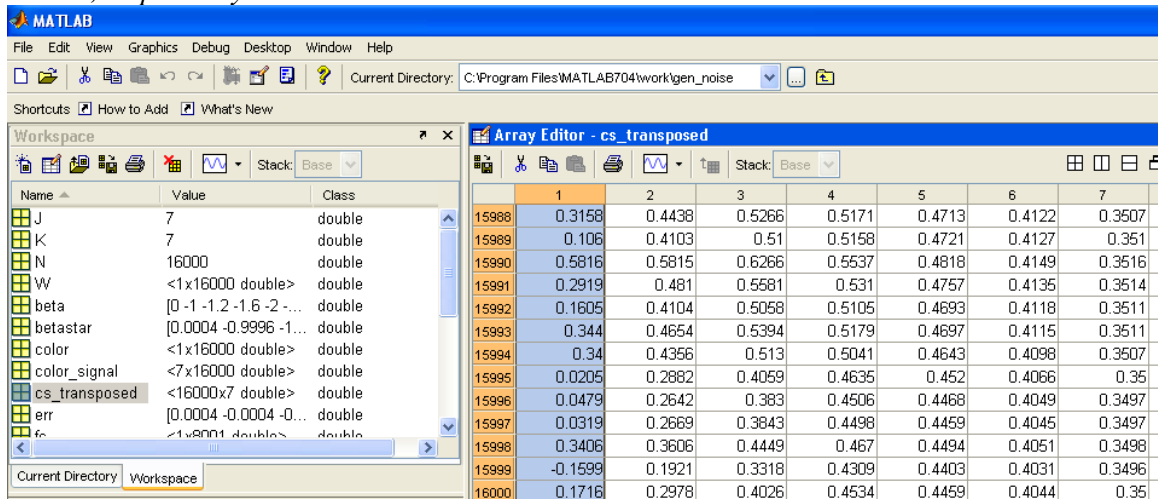


(iii) Double click on `cs_transposed` array name within MATLAB's Workspace window to bring up the Array Editor window.

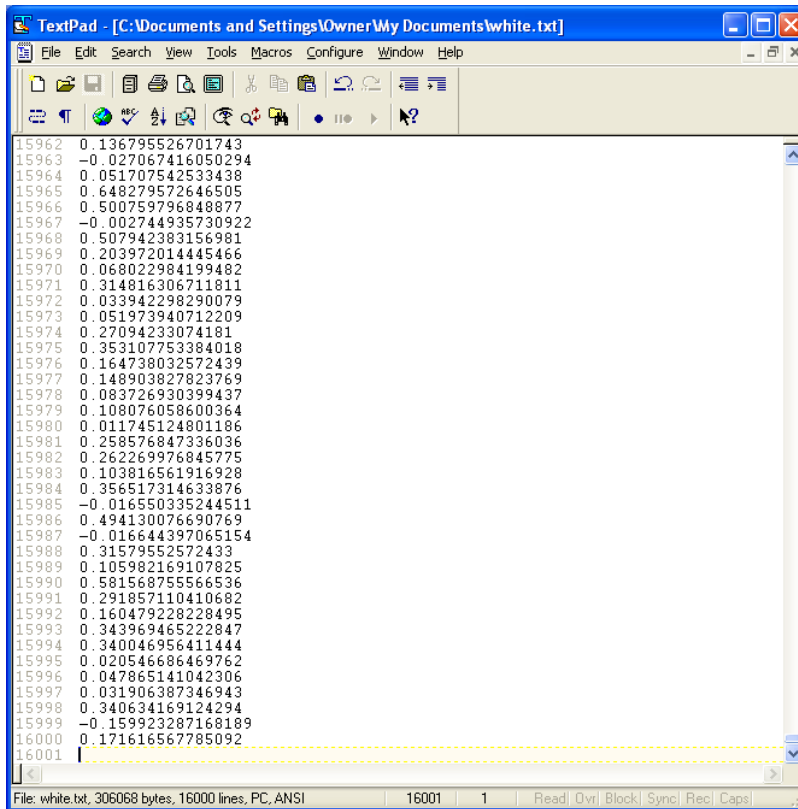


(iv) Highlight the appropriate* column within the cs_transposed array. Press Ctrl-C or right-click on the highlighted area, and select Copy.

*: By appropriate, we meant to emphasize choosing the right column to copy because each filled column consists of 16,000 rows of values corresponding to the 16,000 datapoints of a noise waveform associated with a specific beta. Columns 1- 7 relate to betas 0.0, 1.0, 1.2, 1.6, 2.0, 2.4, and 2.8, respectively.

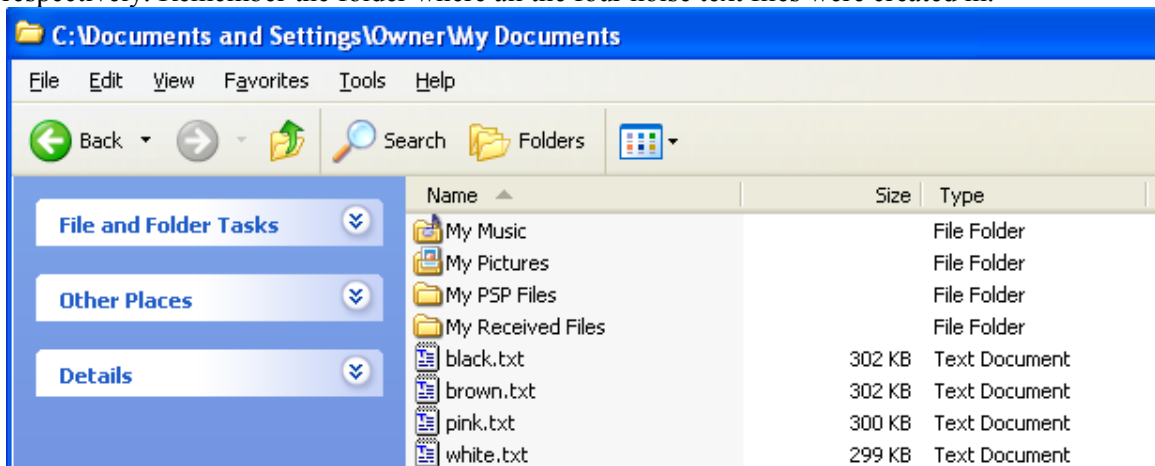


(v) Paste the elements of the selection onto a new file within a text editor (TextPad is recommended). Click File->Save As... on the Menu bar or press F12 to save the file as a text file (.txt file), such as white.txt (since beta=0.0 denotes the spectral exponent of white noise).



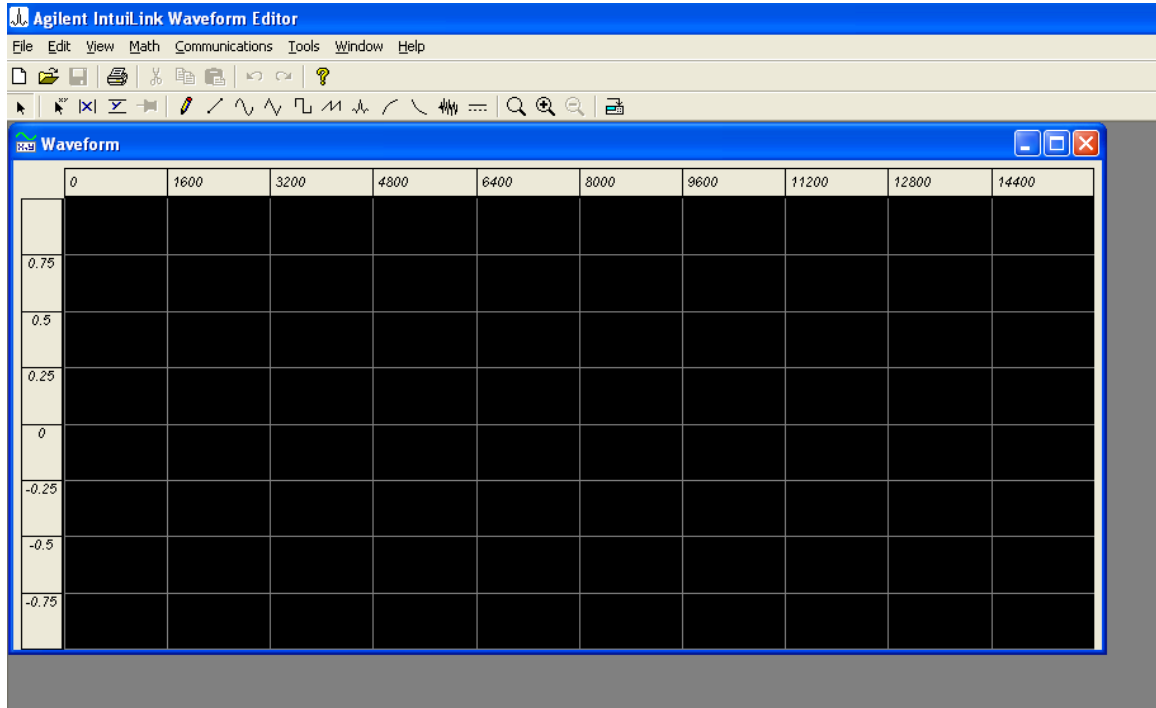
(vi) Close the file and exit TextPad.

(vii) Repeat steps (iv) to (vi), to create three other text files with three other noise colours. For example, I have used betas 1.0, 1.6, and 2.4 to create pink.txt, brown.txt and black.txt, respectively. Remember the folder where all the four noise text files were created in.



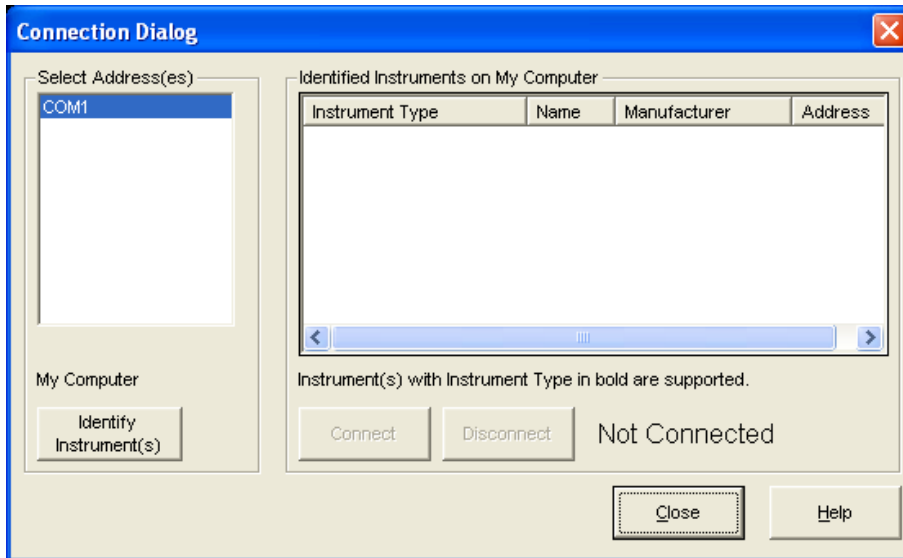
B. Using IntuiLink to Load the Colour Noise Datapoints

- (i) Connect RS-232 serial cable between your pc/laptop and the AWG.
- (ii) Then, launch the Agilent IntuiLink Waveform Editor by going to Start->Programs->Agilent IntuiLink->Waveform Generator->Waveform Editor.



(iii) Next, click Communications->Connection ... on the Menu bar. The Connection Dialog window will appear.

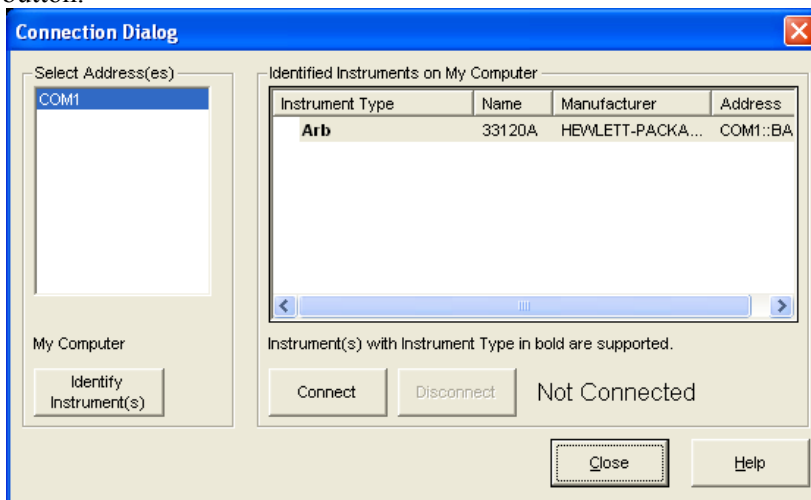
(iv) Now, either double-click on COM1 or the Identify Instrument(s) button. The respective COM's settings box will appear.



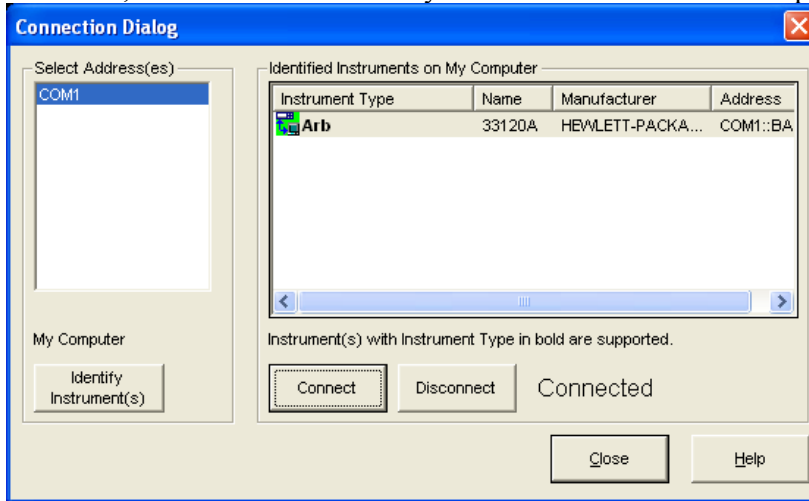
(v) Type in the following values as the selected COM's settings (these values follow the factory setting found on page 195 of the AWG manual) and hit OK.




(vi) Once the arb instrument is identified, its properties will appear as a line within the "Identified Instruments on My Computer" section of the Connection Dialog window. Now, click the Connect button.



(vii) Once you see the word “Connected” above the Close button, you may now hit Close. The Connection Dialog window should go away / disappear without returning any message. Otherwise, check the connection of your RS-232 serial cable and repeat steps (i)-(vii).

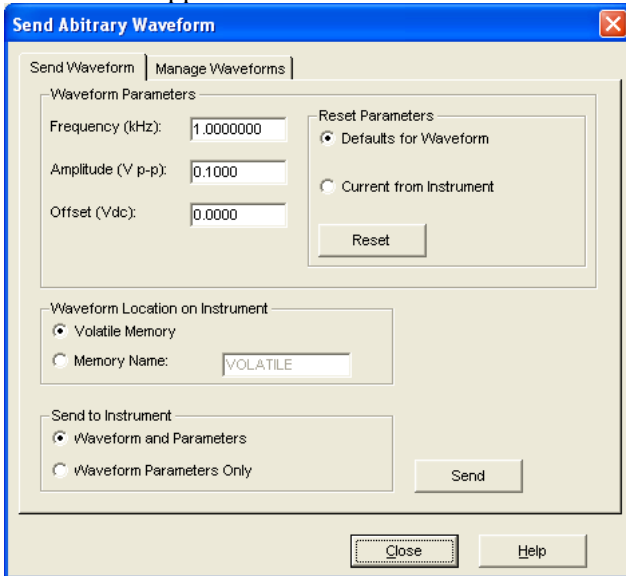


(viii) Press Ctrl+O or click File->Open ... on the Menu bar or click on the open file icon , and locate the folder where the four noise text files were stored under. Select a noise text file, say white.txt and click Open.

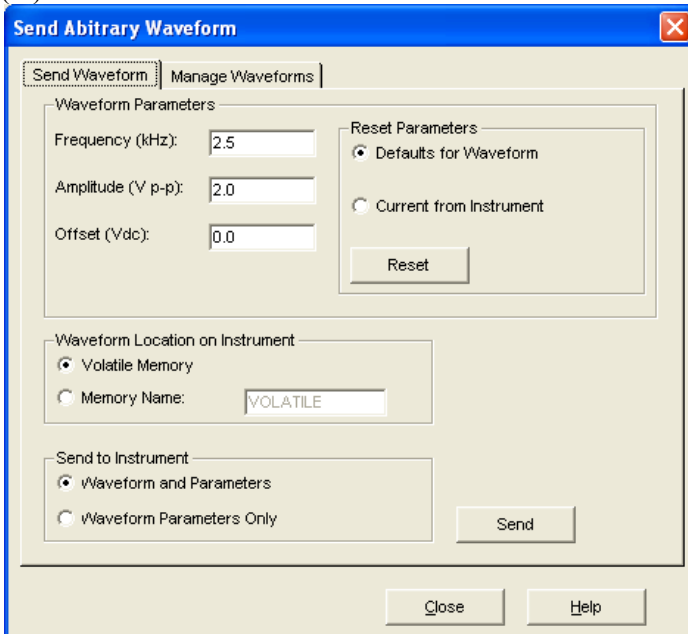
(ix) Repeat step (viii) three more times, each time to open one of the remaining three colour noise datapoints text files. After that, you'd see a screen looking like the following.



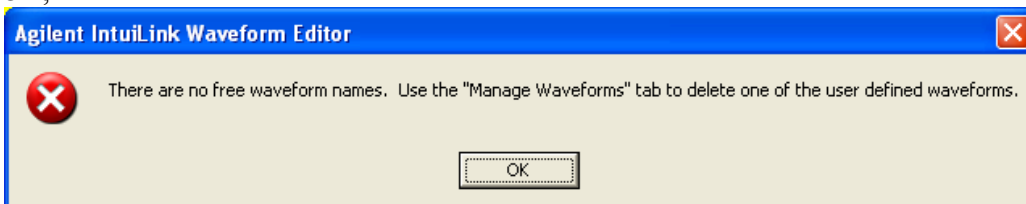
(x) Click anywhere on a colour noise waveform, say the white.txt waveform edit window to make it active. Then, click Communications-> Send Waveform ... on the Menu bar. The following window will appear.



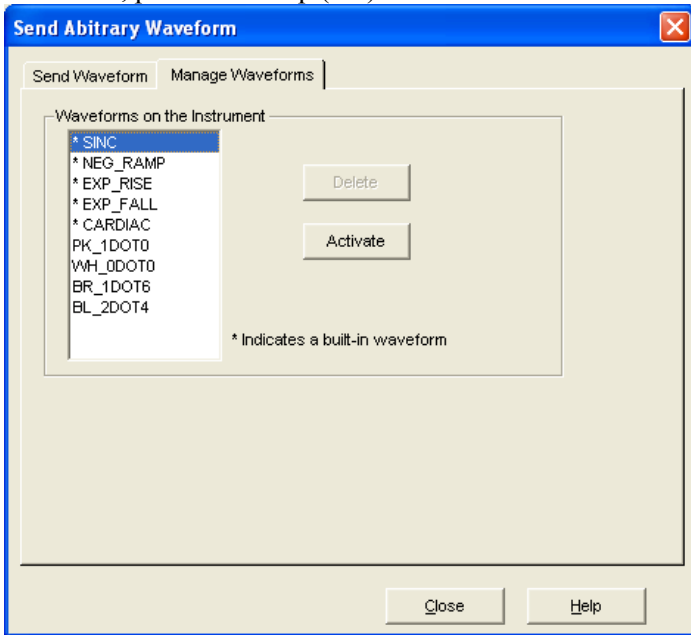
(xi) Now set the Waveform Parameters with the values as seen here.



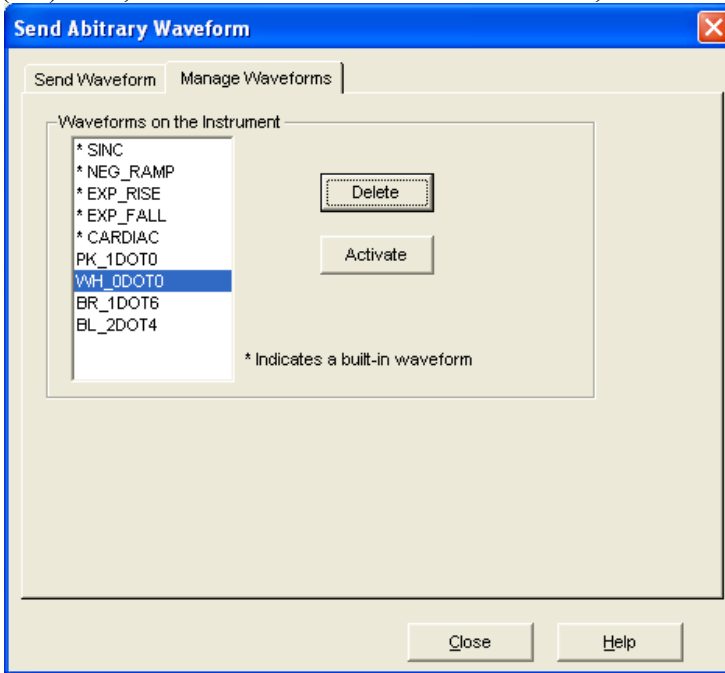
(xii) Then click on "Memory Name:" radio button. If you get the following error message, just hit OK,



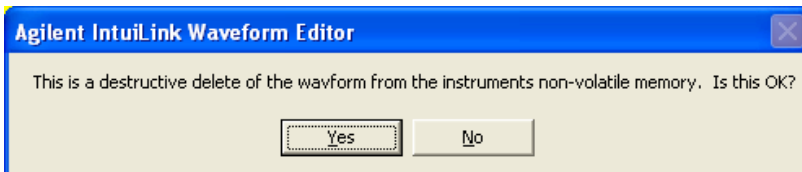
and click on the Manage Waveforms tab within the Send Arbitrary Waveform window. Otherwise, proceed to step (xvi).



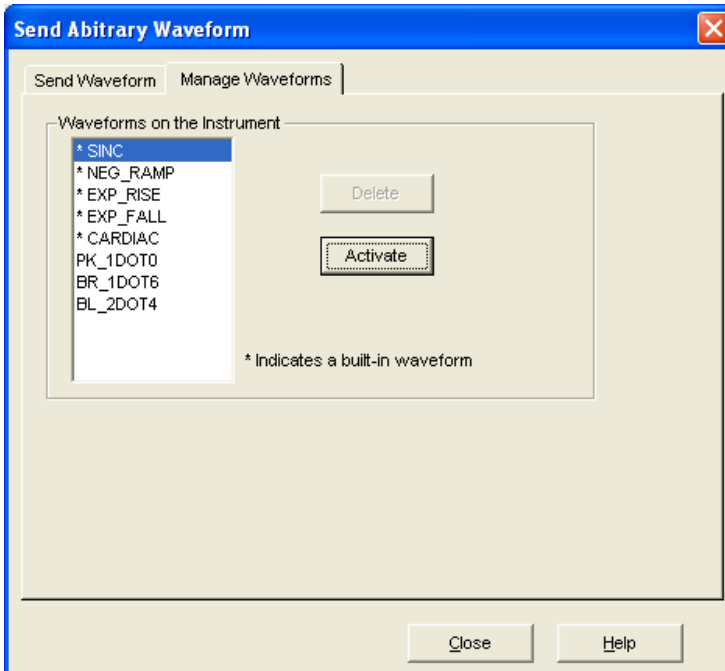
(xiii) Now, select a waveform to delete. Hit Delete,



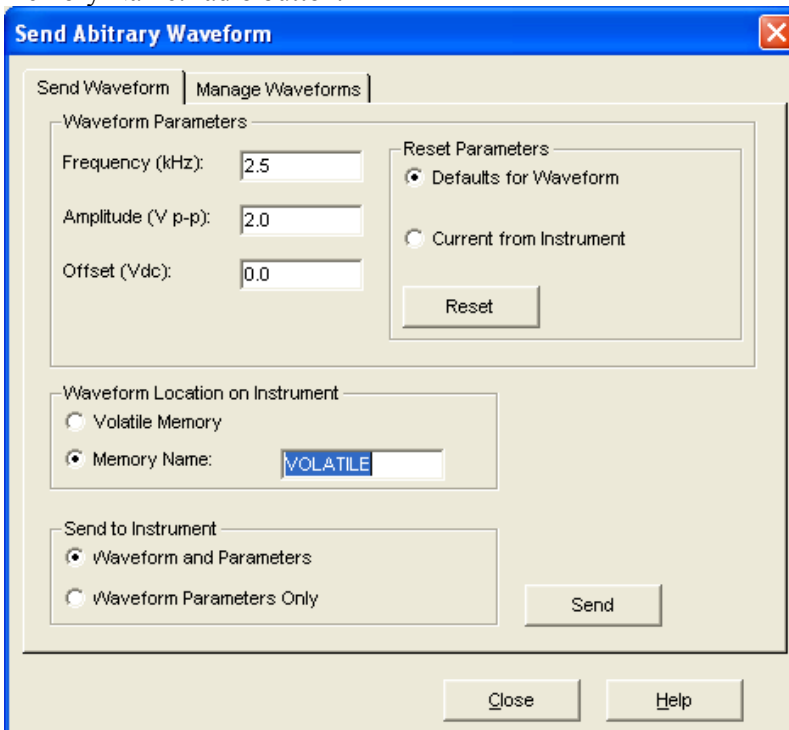
and then hit Yes.



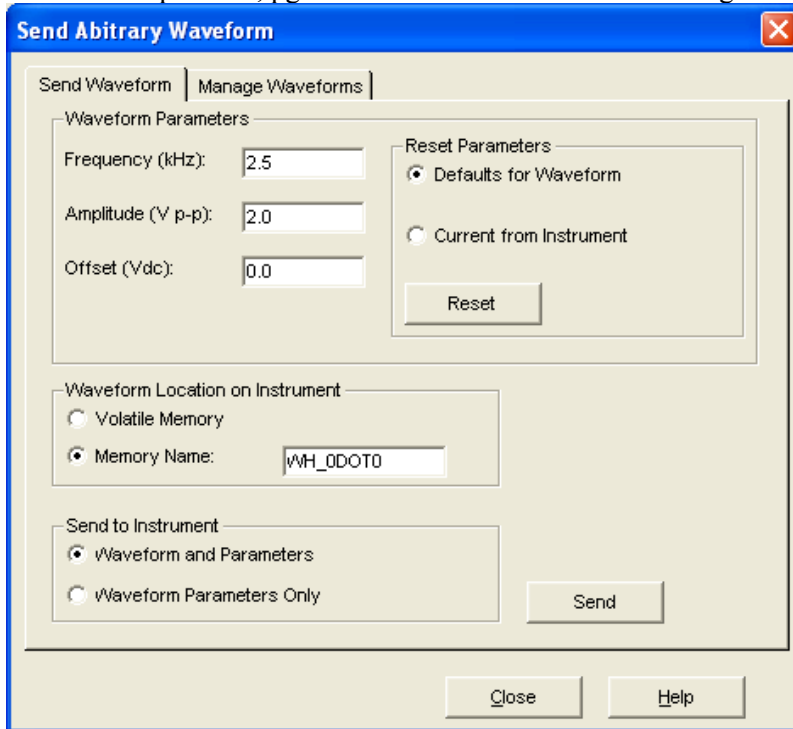
(xiv) This is the screen that you'd get. Notice the selected waveform "WH_0DOT0" is no longer on the list.



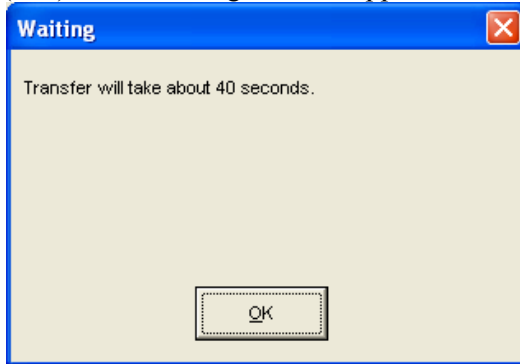
(xv) Click the Send Waveform tab within the Send Arbitrary Waveform window. Now select the Memory Name: radio button.



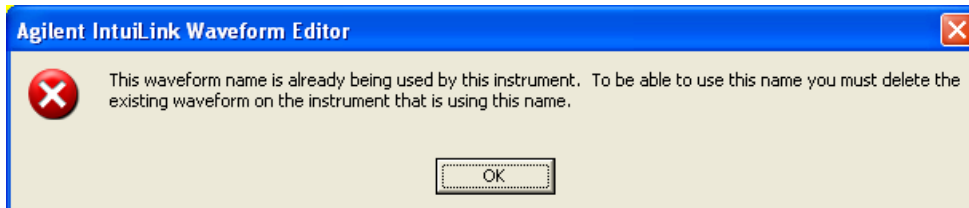
(xvi) Type in a name for your waveform. In this case, I used WH_0DOT0 to denote white noise. Then hit the Send button. Note: You are only allowed 8 characters (refer to bullet point #5, pg. 176 or bullet point #1, pg 182 of the AWG manual for naming restrictions.)



(xvii) The following box will appear. Just hit OK.

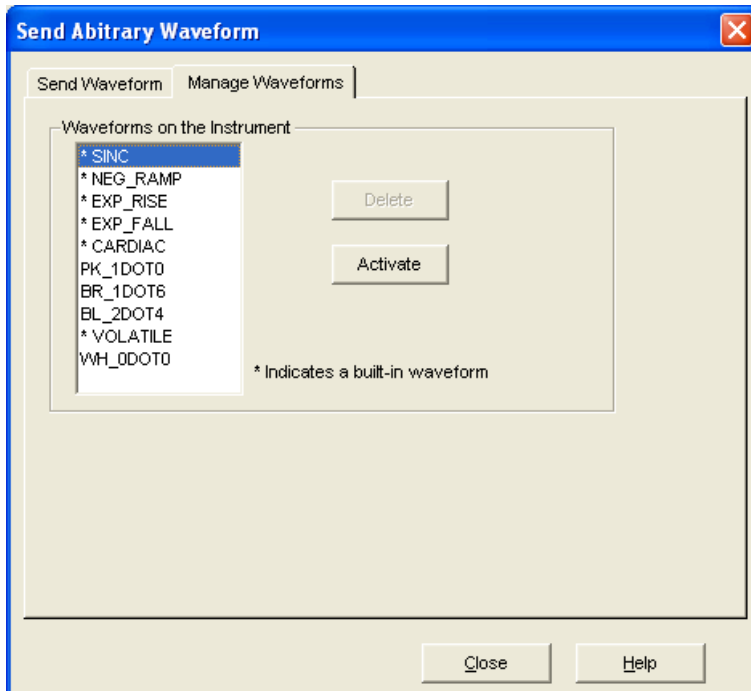


(xviii) When the file transfer has completed, you may do the following check. To be sure if your waveform was successfully stored in non-volatile memory, hit the Manage Waveform tab within the Send Arbitrary Waveform window. You'd get the following error message to indicate that the waveform exists.



Just hit OK.

(xix) If you are still curious, clear the Memory Name: content and hit the Manage Waveform tab. You'd then get this screen to indicate that WH_0DOT0 existed and the same datapoints also existed as VOLATILE.



Now, just hit close.

(xx) Repeat steps (x) to (xvii) three more times to upload the remaining three colour noise waveforms. You may skip steps (xiii) to (xv) in each of your repeats should you have chosen to delete all the four available user-defined waveform in the non-volatile memory one at a time in order to free up the memory for your new waveforms back in step (xiii) itself.

Note: There is only enough memory to store 4 user-defined arbitrary waveforms.

(xxi) Finally, close all the open edit waveform windows, exit the Agilent IntuiLink Waveform Editor application and unplug the cable.

C. Steps to Choose and Output the Desired Arbitrary Waveform from the AWG

(i) Hit the Shift button on the AWG front panel to bring the machine out from remote mode into local mode.

(ii) Hit the Shift button again, then the ARB button and keep pressing the > button or the < button to view all the available arbitrary waveforms. A waveform by the name of VOLATILE will remain there until you shut down the AWG, after which all that was stored in volatile memory will be gone.

(iii) To select a waveform, repeat the previous step by going to the waveform that you want. Stop progressing to the next waveform and hit Enter. The AWG will now be playing back the selected arb waveform.

C.6 Performance Evaluation Setup and Noise Emulation Circuit Configuration

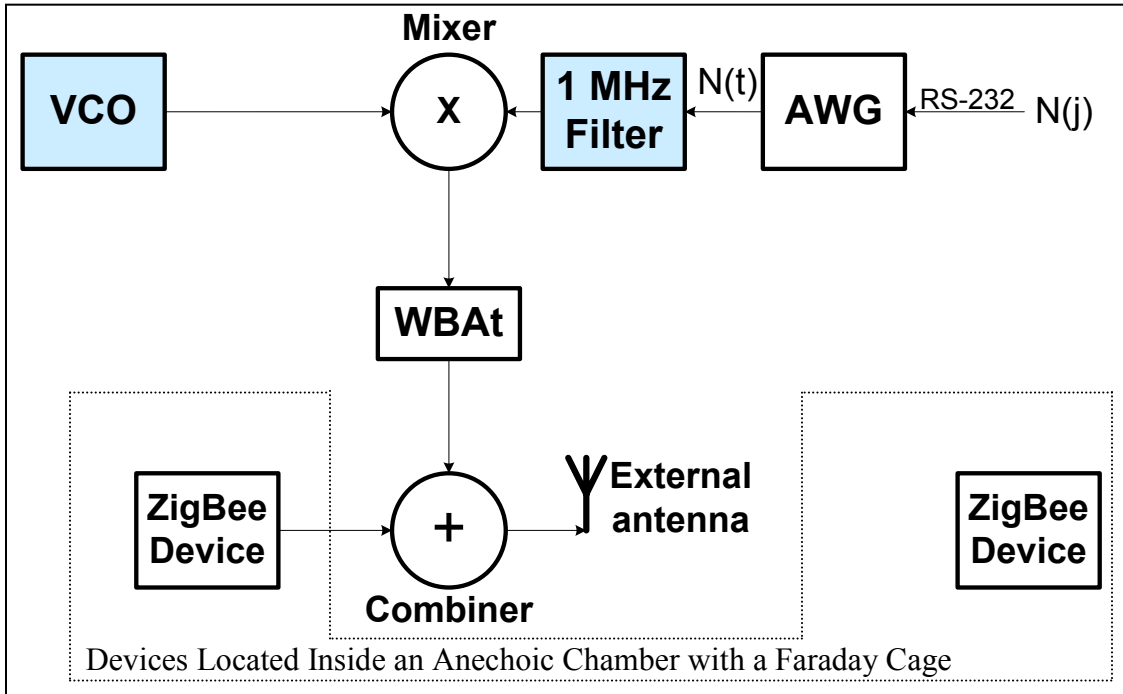


Fig. 1 Module Layout of the Initial Experimental Setup Design

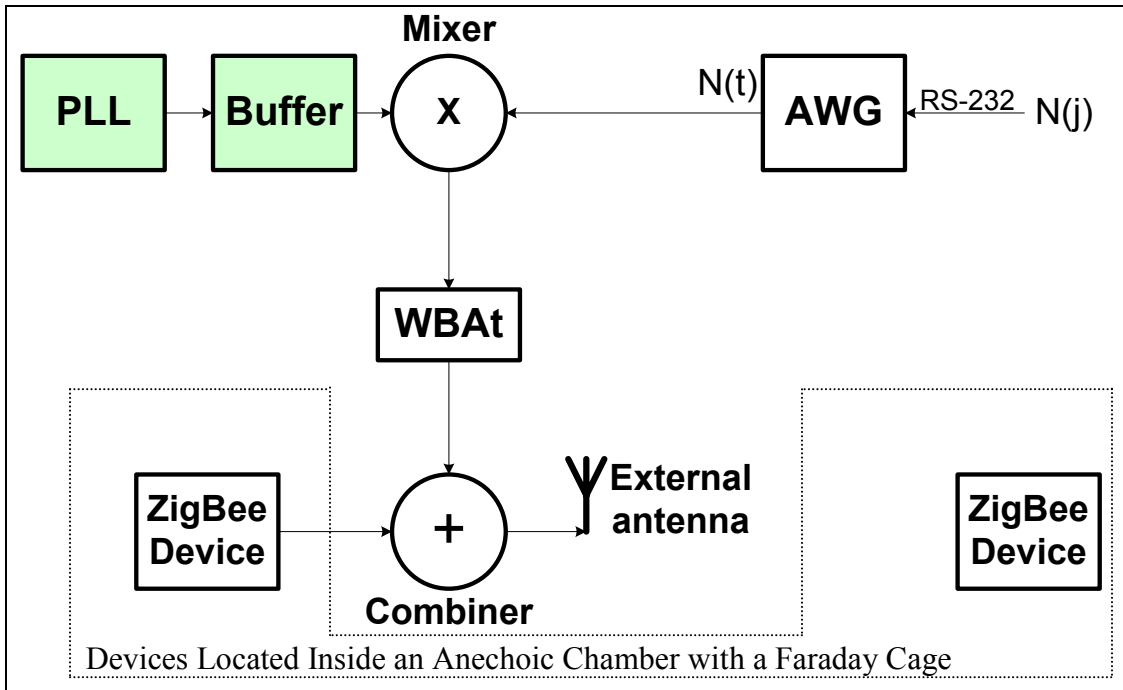


Fig. 2 Module Layout of the Revised Experimental Setup Design

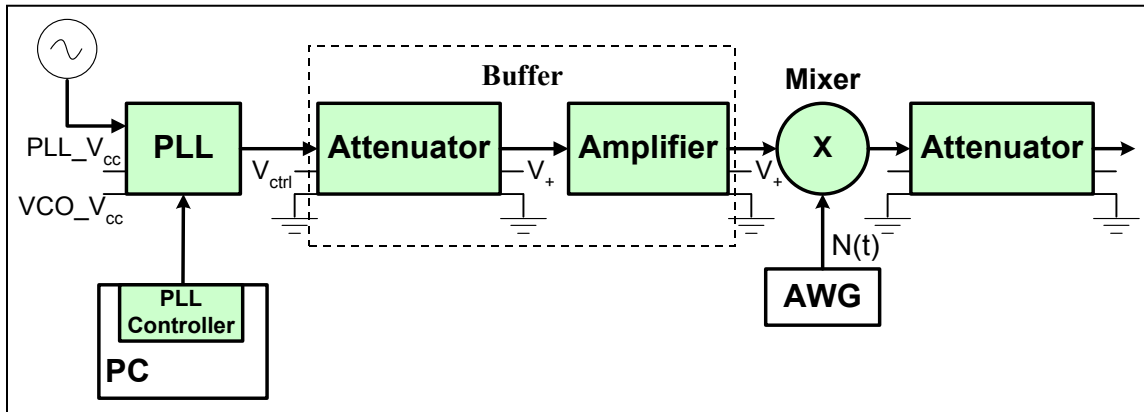


Fig. 3 Setup configuration for PLL, Attenuator, Amplifier and Mixer

To drive the mixer, we need a power level of 7 dBm. The following voltage values were used:

PLL

PLL $V_{cc} = VCO V_{cc} = 4.885V$ (where the voltage expected is in the range of 4.75 V– 5.25 V) (A reference frequency with the settings (10 MHz, 4 V_{pp}) was fed from an arbitrary waveform generator. The PLL output frequency was set to be 2.402 GHz through the PLL Controller. The measured PLL output power was approximately –1 dBm. The PLL specification says that the output power is expected to be between -2 and 4 dBm (typically, 1 dBm).)

Attenuator

$V_{cont} = 6.74 V$ (To obtain an attenuation of approximately 9 dB at this control voltage.)
 $V_+ = 5.005 V$ (expected +5V)

Amplifier

$V_+ = 11.99 V$ (expected 12 V)
 (Gain obtained to be approximately 17 dB over the frequency of 2.402 GHz.)

Mixer

No voltage values needed since device does not have any voltage or ground pins.
 (The mixer's output power was measured to be approximately –27.5 dBm. Since the mixer's specification indicated an LO-RF isolation of 35 dB typically, we can account for the measured output power by performing this calculation: 7 dBm (input) – 35 dB (LO-RF isolation) = –28 dBm (expected output power))

Attenuator

$V_{cont} = 1.392 V$ (To obtain an output power of approximately -60 dBm.)
 $V_+ = 5.005 V$ (expected +5V)

Note: V_{cont} for Attenuator will need to be retuned should the PLL output frequency differ from 2.402 GHz. The amount of attenuation will be dependent upon the amount of amplifier gain achieved over this new PLL output frequency in order to deliver an LO power of 7dBm to the mixer from the PLL.

C.7 Capturing Trace Data through GPIB Interface

Materials:

Hardware	
Spectrum analyzer	HP 8546A
GPIB-USB module	Prologix GPIB-USB Controller
Cable	USB A-B
Computer	Laptop
Software	
Integrated development environment (IDE)	Microsoft Visual Studio 2005 .NET framework 2.0
Serial terminal	HyperTerminal Private Edition, ver. 6.3 (for testing and determining the relevant controller and instrument commands to use in the testApp code) Data Acquisition GUI execution file (testApp.exe)
Operating system	Windows Vista

Table 1. Hardware and software components involved to enable data acquisition from spectrum analyzer

Procedure for Acquiring Trace Data Off HP 8546A Spectrum Analyzer:

1. Connect the hardware components listed in Table 1 according to the order depicted in Fig. 1. Plug the type A end of the USB cable to the laptop and plug the type B end of the USB cable to the Prologix GPIB-USB controller. Then plug the controller directly to the GPIB connector at the back of the spectrum analyzer.
2. Power on the laptop.
3. Power on the spectrum analyzer.
4. Launch testApp.exe.
5. From the **Port Number** pull-down list, select the virtual COM port created by the USB driver. In order to know which COM port to select, go to *Start->Control Panel->System->Device Manager*, look under *Ports (COM & LPT)* for the COM port associated with a USB serial port. In our case, it was COM 11. Then, click the **Connect** button on the data acquisition GUI window once. As illustrated in Fig. 2, once the selected port had been successfully connected, the **Status** box would display the message *Connected*.
6. Using the front panel buttons on the spectrum analyzer, configure the spectrum analyzer to display the appropriate spectrum.
7. In order to capture the data of such display, from the data acquisition GUI window, do the following: First, type in the appropriate filename under the **Filename** textbox. Then, select the appropriate check box to denote the number of traces to be captured. Finally, click the **Capture** button on the data acquisition GUI window once.
8. Repeat steps 6 and 7 for another capture.
9. To close the serial port and to allow the data acquisition GUI window to remain open, click the **Disconnect** button once. Otherwise, click the **Exit** button once to close the serial port as well as to end the data acquisition GUI program.

References:

- [1] Prologix.biz, Prologix GPIB-USB Controller User Manual, ver. 4.65, July 9, 2007, available online:
<http://prologix.googlepages.com/PrologixGpibUsbManual.pdf>
- [2] Prologix.biz, GPIB-USB Controller 4.2 Frequently Asked Questions, available online:
<http://prologix.googlepages.com/faq2>
- [3] Agilent, EMI Receiver Series Programmer's Guide (part number 5962-5083), available online:
<http://cp.literature.agilent.com/litweb/pdf/5962-5083.pdf>
- [4] Agilent, EMI Receiver Series User's Guide (part number 5962-5081), available online:
<http://cp.literature.agilent.com/litweb/pdf/5962-5081.pdf>

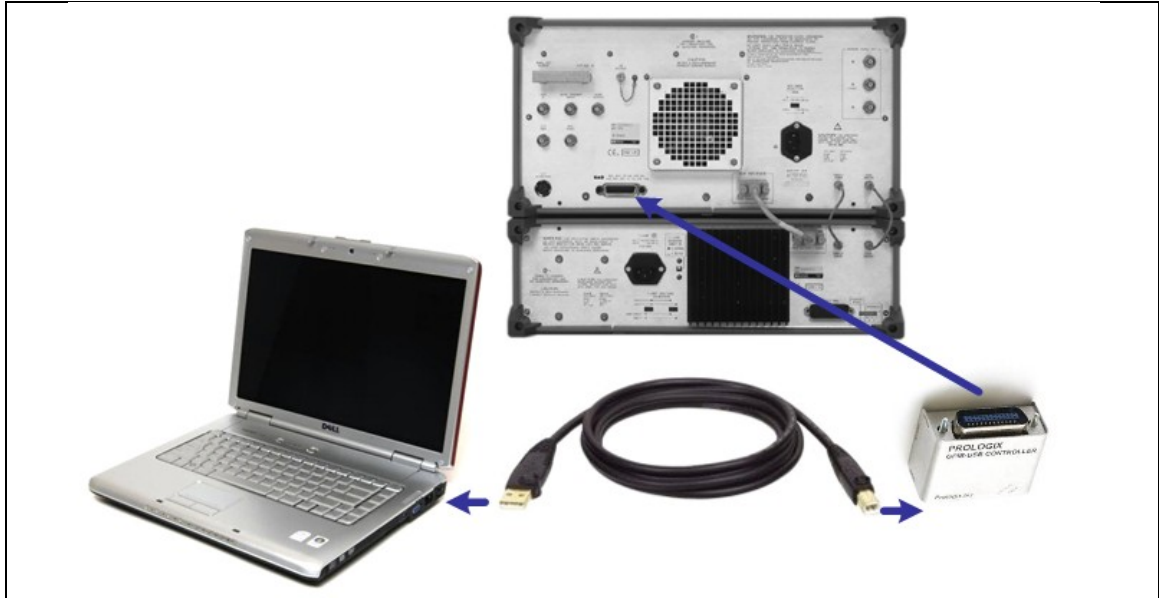


Fig. 1 Hardware setup for data acquisition through GPIB interface

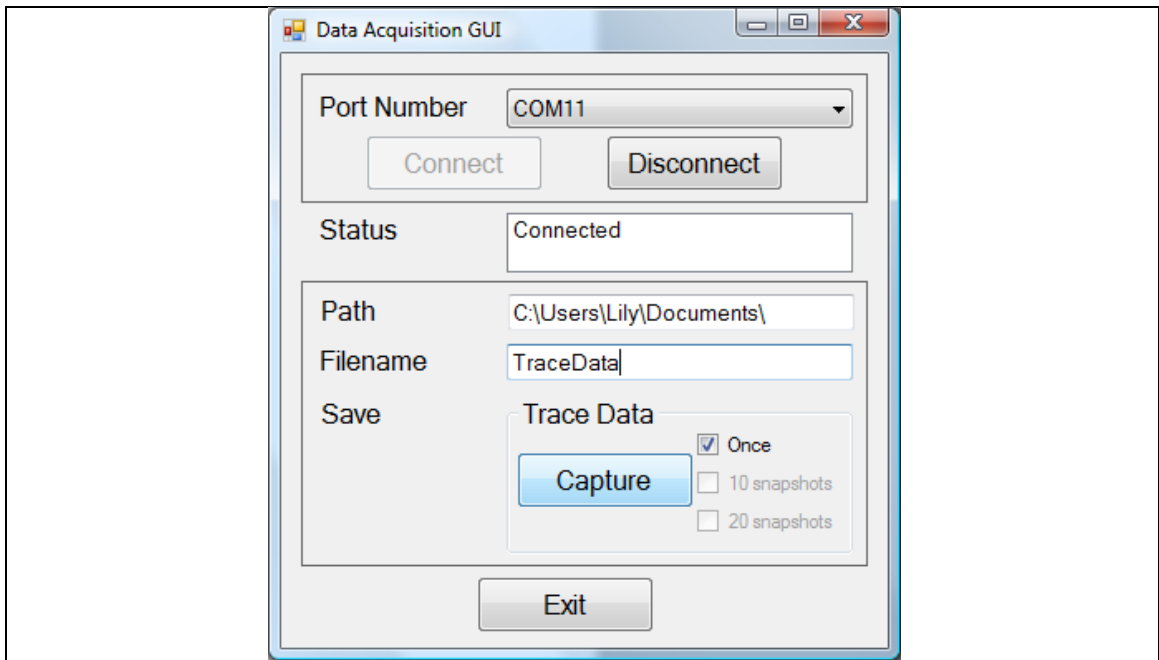


Fig. 2 A screenshot of the GUI window of the GPIB data acquisition program (developed using Microsoft Visual C#)

Note:

Reference [1] tells:

- (i) that the controller mode (as oppose to device mode) is used to remotely control instruments. For the module to be in controller mode, the query `++mode` has to return 1, if not, set it to 1 by issuing `++mode 1`
- (ii) that any USB input that starts with the “++” character sequence is a controller command and is not transmitted over GPIB. For e.g. `++addr` (queries the attached instrument’s GPIB address).
- (iii) to issue `++addr 18` to set the GPIB address to 18.
- (iv) to issue `++loc` in order to return control back to the front panel of the instrument.

Reference [2] tells:

(i) the ASCII setup settings for HyperTerminal with Prologix GPIB-USB controller. Otherwise, these are the settings:

- (a) Connect USB cable. Start HyperTerminal.
- (b) In "File" | "Properties" dialog, select the virtual COM port created by USB driver. Click "OK", then select "Settings" tab and click "ASCII setup...". In the dialog, check "Send line ends with line feeds" and "Echo typed characters locally". Click "OK" twice.
- (c) You are now ready to communicate with the controller. Enter `++ver` command in the HyperTerminal window to verify communication with the controller.

(ii) the PORT settings can be found in <http://prologix.googlepages.com/resources> under C# sample.

Otherwise, these are the settings:

- (a) baudrate or bits per second = 115299
- (b) databits = 8
- (c) parity = none
- (d) stopbits = 1
- (e) flow control = none

Reference [3] tells:

(i) the instrument commands to use:

- (a) preset instrument: `IP`;
- (b) set center frequency in MHz: `CF 2405MZ`;
- (c) set span in MHz: `SP 2MZ`;
- (d) define amplitude unit for amplitude values sent from spectrum analyzer to dBm: `AUNITS DBM`;
- (e) set trace data format to real number format: `TDF P`;
- (f) enable single sweep: `SNGLS`;
- (g) enable continuous sweep: `CONTS`;
- (h) take sweep: `TS`;
- (i) query trace data: `TRA?`;

(ii) to use a semicolon, i.e., ‘;’ to separate each instrument command.

(iii) to use question mark, i.e., ‘?’ to return a query response.

Reference [4] tells:

(i) how to use the front panel keys and the softkeys to perform the desired spectrum analyzer operations.

Pseudocode for Data Acquisition GUI:

<Init Event>

```
CREATE a new serial port object
GET a list of available COM ports
SHOW the available COM ports in a drop down list on the GUI
// INITIALIZE the GUI elements
    CLEAR GUIStatus
    ENABLE the Connect button
    DISABLE the Disconnect and the CaptureTrace buttons, and the CaptureOneTime,
    CaptureTenTimes, CaptureTwentyTimes check boxes
SET the default pathname or directory for storing trace data
```

<Connect Event>

```
GET the user selected COM port number
SET selectedPort to the user selected COM port number
// INITIALIZE the serial port object
    SET PortName to selectedPort
    SET BaudRate to 115200
    SET Parity to None
    SET DataBits to 8
    SET StopBits to 1
SET RTS/CTS handshaking
SET Error handling
SET the read and the write timeouts
OPEN the serial port
// UPDATE the GUI elements
    CLEAR GUIStatus
    SET GUIStatus to "Connected" to show port connection has been established between
    pc and instrument
    ENABLE the Disconnect and the CaptureTrace buttons, and the CaptureOneTime,
    CaptureTenTimes, CaptureTwentyTimes check boxes
    DISABLE the Connect button
// CONFIGURE the GPIB address of the GPIB_USB module to that of the instrument being
controlled
    GET the instrument GPIB address, currAddr
    SEND the GPIB command with the value denoted by currAddr appended to the string
    '++addr ' to the GPIB-USB module
```

<CaptureTrace Event>

```
SET numSnapshots to 0
SET amplitude unit, trace data format, and enable single sweep
GET center frequency, span, start frequency, stop frequency, and reference level
SET centerFreq, spanFreq, startFreq, stopFreq, and refLevel
// UPDATE the GUI elements
    CLEAR GUIStatus
    SET GUIStatus to "Busy: Capturing trace ..." // to show trace capture has started
    ENABLE the Disconnect button
    DISABLE the Connect button and the CaptureTrace button
IF CaptureOneTime is checked THEN
    SET numSnapshots to 1
ELSE-IF CaptureTenTimes is checked THEN
    SET numSnapshots to 10
ELSE // for the case when CaptureTwentyTimes is checked
    SET numSnapshots to 20
END IF
FOR each numSnapshots
    CALL processFile function // to process incoming data
END FOR
SEND the GPIB command '++loc' to return control back to front panel of instrument
// UPDATE the GUI elements
    CLEAR GUIStatus
    SET GUIStatus to "Idle: Completed trace capture" to show trace capture has ended
    ENABLE the Disconnect and the CaptureTrace buttons
    DISABLE the Connect button
    UNCHECK the CaptureOneTime, CaptureTenTimes, CaptureTwentyTimes check boxes
```

```

<Disconnect Event>
CLOSE the serial port
// UPDATE the GUI elements
    CLEAR GUIStatus
    SET GUIStatus to "Disconnected" to show port connection has been ended between pc
and instrument
    ENABLE the Connect button
    DISABLE the Disconnect and the CaptureTrace buttons, and the CaptureOneTime,
    CaptureTenTimes, CaptureTwentyTimes check boxes

<Exit Event>
CLOSE the serial port
CLOSE the GUI window

<ProcessFile Function>
GET filename
IF filename exists THEN
    SHOW file error message
    RETURN
END IF
OPEN the file
GET incoming trace data
// FORMAT data before writing
    COUNT the number of amplitudes returned from the comma-separated incoming trace
data
    FOR each amplitude
        COMPUTE the frequency corresponding to the amplitude
        APPEND the computed frequency to the amplitude value separated with a
comma
        WRITE formatted string to file
    END FOR
CLOSE the file

```


Code for Data Acquisition GUI:

Form1.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.IO;
using System.IO.Ports;
using System.Text;
using System.Windows.Forms;

namespace testApp
{
    public partial class Form1 : Form
    {
        // Class Variables
        private static SerialPort _serialPort;
        private static string defaultAddr="";
        private static decimal centerFreq=0;
        private static decimal spanFreq=0;
        private static decimal startFreq=0;
        private static decimal stopFreq=0;
        private static decimal refLevel=0;

        public Form1()
        {
            InitializeComponent();
        }

        private void btnCaptureTrace_Click(object sender, EventArgs e)
        {
            int numSnapshots = 0;

            // Query center frequency
            _serialPort.WriteLine("CF?");
            string centerFreqStr = _serialPort.ReadLine();
            centerFreqStr = centerFreqStr.Trim();
            centerFreq = decimal.Parse(centerFreqStr);

            //Query span
            _serialPort.WriteLine("SP?");
            spanFreq = decimal.Parse((_serialPort.ReadLine()).Trim());

            //Query start frequency
            _serialPort.WriteLine("FA?");
            startFreq = decimal.Parse((_serialPort.ReadLine()).Trim());

            //Query stop frequency
            _serialPort.WriteLine("FB?");
            stopFreq = decimal.Parse((_serialPort.ReadLine()).Trim());

            //Query reference level
            _serialPort.WriteLine("RL?");
            refLevel = decimal.Parse((_serialPort.ReadLine()).Trim());

            // Set amplitude unit; set trace data format;
            _serialPort.WriteLine("AUNITS DBM;TDF P;SNGLS;");

            // Update GUI -- to signal capture has started
            btnConnect.Enabled = false; // grey out the Connect button
            btnDisconnect.Enabled = true; // enable the Disconnect button
            btnCaptureTrace.Enabled = false; // grey out the Capture trace button
            lbStatus.Items.Clear(); // clear the status box
            lbStatus.Items.Insert(0, "Busy: Capturing trace ...");
        }
    }
}
```

```

true))
    if ((cbCaptureTenTimes.Checked == true) || (cbCaptureTwentyTimes.Checked ==
true))
    {
        if (cbCaptureTenTimes.Checked == true)
        {
            numSnapshots = 10;
        }
        else
        {
            numSnapshots = 20;
        }
        for (int i = 0; i < numSnapshots; i++)
        {
            processFile(i.ToString());
        }
    }
    else // default case for cbCaptureOneTime.Checked == true or if none of the
checkboxes is checked
    {
        // Create file, read data till EOF, close file
        processFile("");
    }

    // Set to continuous stream and return control back to front panel (equiv. to
pressing LOCAL soft button)
    _serialPort.WriteLine("CONTS;");
    _serialPort.WriteLine(++loc);

    // Update GUI -- to signal capture has completed
    guiInit2();
    cbCaptureOneTime.CheckState = CheckState.Unchecked; // uncheck the Once
check box
    cbCaptureTenTimes.CheckState = CheckState.Unchecked; // uncheck the 10
snapshots check box
    cbCaptureTwentyTimes.CheckState = CheckState.Unchecked; // uncheck the 20
snapshots check box
    lbStatus.Items.Clear(); // clear the status box
    lbStatus.Items.Insert(0, "Idle: Completed trace capture.");
}

private void btnConnect_Click(object sender, EventArgs e)
{
    // Get the selected COM port
    string selectedPort = cbSerialPortNum.SelectedItem.ToString();

    // Initialize the serial port object
    _serialPort.PortName = selectedPort;
    _serialPort.BaudRate = 115200;
    _serialPort.Parity = Parity.None;
    _serialPort.DataBits = 8;
    _serialPort.StopBits = StopBits.One;

    // RTS/CTS handshaking
    _serialPort.Handshake = Handshake.RequestToSend;
    _serialPort.DtrEnable = true;

    // Error handling
    _serialPort.DiscardNull = false;
    _serialPort.ParityReplace = 0;

    // Set the read/write timeouts
    _serialPort.ReadTimeout = 10000; // this is how long a read may take before
timing out
    _serialPort.WriteTimeout = 500;

    // Open the serial port
    _serialPort.Open();
}

```

```

// Update GUI elements
guiInit2();
lbStatus.Items.Clear(); // clear the status box
lbStatus.Items.Insert(0, "Connected");
// Send initialization and/or setup commands to the instrument
//-----

// Query current GPIB address
_serialPort.WriteLine("++addr");
defaultAddr = _serialPort.ReadLine();

// Set GPIB address to current
StringBuilder gpibAddr = new StringBuilder();
gpibAddr.Append("++addr ");
gpibAddr.Append(defaultAddr.Trim());
_serialPort.WriteLine(gpibAddr.ToString());

// Preset instrument; set center freq; set span;
//_serialPort.WriteLine("IP;CF 2405MZ;SP 2MZ;"); // commented out to allow
user configure his/her own settings
}

private void btnDisconnect_Click(object sender, EventArgs e)
{
// Close the serial port
_serialPort.Close();

// Update GUI elements
guiInit1();
lbStatus.Items.Clear(); // clear the status box
lbStatus.Items.Insert(0, "Disconnected");
}

private void btnExit_Click(object sender, EventArgs e)
{
// Close the serial port
_serialPort.Close();

// Close GUI window
Close();
}

private void cbCaptureOneTime_CheckedChanged(object sender, EventArgs e)
{
if (cbCaptureOneTime.CheckState == CheckState.Checked)
{
cbCaptureTenTimes.Enabled = false;
cbCaptureTwentyTimes.Enabled = false;
}
else if (cbCaptureOneTime.CheckState == CheckState.Unchecked)
{
cbCaptureTenTimes.Enabled = true;
cbCaptureTwentyTimes.Enabled = true;
}
}

private void cbCaptureTenTimes_CheckedChanged(object sender, EventArgs e)
{
if (cbCaptureTenTimes.CheckState == CheckState.Checked)
{
cbCaptureOneTime.Enabled = false;
cbCaptureTwentyTimes.Enabled = false;
}
else if (cbCaptureTenTimes.CheckState == CheckState.Unchecked)
{
cbCaptureOneTime.Enabled = true;
cbCaptureTwentyTimes.Enabled = true;
}
}

```

```

}

private void cbCaptureTwentyTimes_CheckedChanged(object sender, EventArgs e)
{
    if (cbCaptureTwentyTimes.CheckState == CheckState.Checked)
    {
        cbCaptureOneTime.Enabled = false;
        cbCaptureTenTimes.Enabled = false;
    }
    else if (cbCaptureTwentyTimes.CheckState == CheckState.Unchecked)
    {
        cbCaptureOneTime.Enabled = true;
        cbCaptureTenTimes.Enabled = true;
    }
}

private void Form1Load(object sender, EventArgs e)
{
    // Initialize the serial port object
    _serialPort = new SerialPort();

    // Initialize the list of available COM ports
    cbSerialPortNum.Items.Clear();
    string[] availablePorts = SerialPort.GetPortNames();
    foreach(string availablePort in availablePorts)
    {
        cbSerialPortNum.Items.Add(availablePort);
    }
    if (cbSerialPortNum.Items.Count > 0)
    {
        cbSerialPortNum.SelectedIndex = 0;
    }
    // Initialize the GUI elements
    guiInit1();
    lbStatus.Items.Clear(); // clear the status box
    // Set the default directory and file name
    StringBuilder defaultPath = new StringBuilder();
    defaultPath.Append(Environment.GetFolderPath(Environment.SpecialFolder.Personal));
    defaultPath.Append(Path.DirectorySeparatorChar);
    edPath.Text = defaultPath.ToString();

    edFilename.Text = "TraceData";
}

private void guiInit1()
{
    btnConnect.Enabled = true; // enable the Connect button
    btnDisconnect.Enabled = false; // grey out the Disconnect button
    btnCaptureTrace.Enabled = false; // grey out the Capture trace button
    cbCaptureOneTime.Enabled = false; // grey out the Once check box
    cbCaptureTenTimes.Enabled = false; // grey out the 10 snapshots check
box
    cbCaptureTwentyTimes.Enabled = false; // grey out the 20 snapshots check
box
}

private void guiInit2()
{
    btnConnect.Enabled = false; // grey out the Connect button
    btnDisconnect.Enabled = true; // enable the Disconnect button
    btnCaptureTrace.Enabled = true; // enable the Capture trace button
    cbCaptureOneTime.Enabled = true; // enable the Once check box
    cbCaptureTenTimes.Enabled = true; // enable the 10 snapshots check box
    cbCaptureTwentyTimes.Enabled = true; // enable the 20 snapshots check box
}

private void processFile(string count)
{

```

```

string incomingData;
StringBuilder fileName = new StringBuilder();
fileName.Append(edPath.Text);
fileName.Append(edFilename.Text);
if (count.Length != 0)
{
    fileName.Append("_");
    fileName.Append(count);
}
fileName.Append(".txt");
// Test if the file already exists
if (File.Exists(fileName.ToString()))
{
    MessageBox.Show("A file with the specified name already exists", "File
Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
    return;
}
// Open the file
using (StreamWriter wrtFS = File.CreateText(fileName.ToString()))
{
    incomingData = receiveSerialData();
    if (incomingData.Length == 0)
    {
        return;
    }
    else
    {
        // Stream writing
        // If need be, uncomment these lines to save these info in the file
        //wrtFS.WriteLine(centerFreq.ToString());
        //wrtFS.WriteLine(spanFreq.ToString());
        //wrtFS.WriteLine(startFreq.ToString());
        //wrtFS.WriteLine(stopFreq.ToString());
        //wrtFS.WriteLine(refLevel.ToString());
        decimal factor = spanFreq / 400;
        decimal corrFreqValue;
        // format data before writing to have
        // comma separated values, i.e. amplitude, frequency rows in file
        string[] datumValue = incomingData.Split(new Char[] { ',' });
        for (int j = 0; j < datumValue.GetLength(0); j++)//0-400 = 401 points
        {
            corrFreqValue = startFreq + factor * (j);
            datumValue[j] = datumValue[j] + "," +
corrFreqValue.ToString("F2");
            wrtFS.WriteLine(datumValue[j]);
        }
        wrtFS.Close();
    }
}

private string receiveSerialData()
{
    try
    {
        string receivedData;
        _serialPort.WriteLine("TS;TS;TRA?;"); // Send commands (obtain trace) to
instrument
        receivedData = (_serialPort.ReadLine()).Trim();
        return receivedData;
    }
    catch (Exception ex)
    {
        return "";
    }
}
}
}

```

Code for Data Acquisition GUI (continued):

Form1.Designer.cs

```
namespace testApp
{
    partial class Form1
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be disposed;
        otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            this.cbSerialPortNum = new System.Windows.Forms.ComboBox();
            this.label1 = new System.Windows.Forms.Label();
            this.label3 = new System.Windows.Forms.Label();
            this.lbStatus = new System.Windows.Forms.ListBox();
            this.label4 = new System.Windows.Forms.Label();
            this.label5 = new System.Windows.Forms.Label();
            this.btnConnect = new System.Windows.Forms.Button();
            this.edFilename = new System.Windows.Forms.TextBox();
            this.groupBox2 = new System.Windows.Forms.GroupBox();
            this.cbCaptureTwentyTimes = new System.Windows.Forms.CheckBox();
            this.btnCaptureTrace = new System.Windows.Forms.Button();
            this.cbCaptureTenTimes = new System.Windows.Forms.CheckBox();
            this.cbCaptureOneTime = new System.Windows.Forms.CheckBox();
            this.btnExit = new System.Windows.Forms.Button();
            this.btnDisconnect = new System.Windows.Forms.Button();
            this.panel1 = new System.Windows.Forms.Panel();
            this.panel2 = new System.Windows.Forms.Panel();
            this.edPath = new System.Windows.Forms.TextBox();
            this.label6 = new System.Windows.Forms.Label();
            this.groupBox2.SuspendLayout();
            this.panel2.SuspendLayout();
            this.SuspendLayout();
            //
            // cbSerialPortNum
            //
            this.cbSerialPortNum.DropDownStyle =
System.Windows.Forms.ComboBoxStyle.DropDownList;
            this.cbSerialPortNum.Font = new System.Drawing.Font("Microsoft Sans Serif",
9.75F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte) 0));
            this.cbSerialPortNum.FormattingEnabled = true;
            this.cbSerialPortNum.Location = new System.Drawing.Point(136, 21);
        }
    }
}
```

```

this.cbSerialPortNum.Name = "cbSerialPortNum";
this.cbSerialPortNum.Size = new System.Drawing.Size(209, 24);
this.cbSerialPortNum.TabIndex = 0;
//
// label1
//
this.label1.AutoSize = true;
this.label1.Font = new System.Drawing.Font("Microsoft Sans Serif", 12F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)0));
this.label1.Location = new System.Drawing.Point(19, 22);
this.label1.Name = "label1";
this.label1.Size = new System.Drawing.Size(98, 20);
this.label1.TabIndex = 1;
this.label1.Text = "Port Number";
//
// label3
//
this.label3.AutoSize = true;
this.label3.Font = new System.Drawing.Font("Microsoft Sans Serif", 12F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)0));
this.label3.Location = new System.Drawing.Point(7, 69);
this.label3.Name = "label3";
this.label3.Size = new System.Drawing.Size(45, 20);
this.label3.TabIndex = 5;
this.label3.Text = "Save";
//
// lbStatus
//
this.lbStatus.Font = new System.Drawing.Font("Microsoft Sans Serif", 9.75F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)0));
this.lbStatus.FormattingEnabled = true;
this.lbStatus.ItemHeight = 16;
this.lbStatus.Location = new System.Drawing.Point(136, 96);
this.lbStatus.Name = "lbStatus";
this.lbStatus.Size = new System.Drawing.Size(209, 36);
this.lbStatus.TabIndex = 6;
//
// label4
//
this.label4.AutoSize = true;
this.label4.Font = new System.Drawing.Font("Microsoft Sans Serif", 12F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)0));
this.label4.Location = new System.Drawing.Point(19, 96);
this.label4.Name = "label4";
this.label4.Size = new System.Drawing.Size(56, 20);
this.label4.TabIndex = 7;
this.label4.Text = "Status";
//
// label5
//
this.label5.AutoSize = true;
this.label5.Font = new System.Drawing.Font("Microsoft Sans Serif", 12F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)0));
this.label5.Location = new System.Drawing.Point(19, 175);
this.label5.Name = "label5";
this.label5.Size = new System.Drawing.Size(74, 20);
this.label5.TabIndex = 8;
this.label5.Text = "Filename";
//
// btnConnect
//
this.btnConnect.Font = new System.Drawing.Font("Microsoft Sans Serif", 12F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)0));
this.btnConnect.Location = new System.Drawing.Point(51, 49);
this.btnConnect.Name = "btnConnect";
this.btnConnect.Size = new System.Drawing.Size(107, 34);
this.btnConnect.TabIndex = 9;
this.btnConnect.Text = "Connect";

```

```

        this.btnConnect.UseVisualStyleBackColor = true;
        this.btnConnect.Click += new System.EventHandler(this.btnConnect_Click);
        //
        // edFilename
        //
        this.edFilename.Font = new System.Drawing.Font("Microsoft Sans Serif", 9.75F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte) 0));
        this.edFilename.Location = new System.Drawing.Point(136, 175);
        this.edFilename.Name = "edFilename";
        this.edFilename.Size = new System.Drawing.Size(209, 22);
        this.edFilename.TabIndex = 10;
        //
        // groupBox2
        //
        this.groupBox2.Controls.Add(this.cbCaptureTwentyTimes);
        this.groupBox2.Controls.Add(this.btnCaptureTrace);
        this.groupBox2.Controls.Add(this.cbCaptureTenTimes);
        this.groupBox2.Controls.Add(this.cbCaptureOneTime);
        this.groupBox2.Font = new System.Drawing.Font("Microsoft Sans Serif", 12F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte) 0));
        this.groupBox2.Location = new System.Drawing.Point(136, 207);
        this.groupBox2.Name = "groupBox2";
        this.groupBox2.Size = new System.Drawing.Size(209, 95);
        this.groupBox2.TabIndex = 13;
        this.groupBox2.TabStop = false;
        this.groupBox2.Text = "Trace Data";
        //
        // cbCaptureTwentyTimes
        //
        this.cbCaptureTwentyTimes.AutoSize = true;
        this.cbCaptureTwentyTimes.Font = new System.Drawing.Font("Microsoft Sans
Serif", 8.25F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
((byte) 0));
        this.cbCaptureTwentyTimes.Location = new System.Drawing.Point(115, 67);
        this.cbCaptureTwentyTimes.Name = "cbCaptureTwentyTimes";
        this.cbCaptureTwentyTimes.Size = new System.Drawing.Size(89, 17);
        this.cbCaptureTwentyTimes.TabIndex = 17;
        this.cbCaptureTwentyTimes.Text = "20 snapshots";
        this.cbCaptureTwentyTimes.UseVisualStyleBackColor = true;
        this.cbCaptureTwentyTimes.CheckedChanged += new
System.EventHandler(this.cbCaptureTwentyTimes_CheckedChanged);
        //
        // btnCaptureTrace
        //
        this.btnCaptureTrace.Font = new System.Drawing.Font("Microsoft Sans Serif",
12F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte) 0));
        this.btnCaptureTrace.Location = new System.Drawing.Point(6, 33);
        this.btnCaptureTrace.Name = "btnCaptureTrace";
        this.btnCaptureTrace.Size = new System.Drawing.Size(107, 34);
        this.btnCaptureTrace.TabIndex = 14;
        this.btnCaptureTrace.Text = "Capture";
        this.btnCaptureTrace.UseVisualStyleBackColor = true;
        this.btnCaptureTrace.Click += new
System.EventHandler(this.btnCaptureTrace_Click);
        //
        // cbCaptureTenTimes
        //
        this.cbCaptureTenTimes.AutoSize = true;
        this.cbCaptureTenTimes.Font = new System.Drawing.Font("Microsoft Sans Serif",
8.25F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte) 0));
        this.cbCaptureTenTimes.Location = new System.Drawing.Point(115, 44);
        this.cbCaptureTenTimes.Name = "cbCaptureTenTimes";
        this.cbCaptureTenTimes.Size = new System.Drawing.Size(89, 17);
        this.cbCaptureTenTimes.TabIndex = 16;
        this.cbCaptureTenTimes.Text = "10 snapshots";
        this.cbCaptureTenTimes.UseVisualStyleBackColor = true;
        this.cbCaptureTenTimes.CheckedChanged += new
System.EventHandler(this.cbCaptureTenTimes_CheckedChanged);

```



```

//
// cbCaptureOneTime
//
this.cbCaptureOneTime.AutoSize = true;
this.cbCaptureOneTime.Font = new System.Drawing.Font("Microsoft Sans Serif",
8.25F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte) 0));
this.cbCaptureOneTime.Location = new System.Drawing.Point(115, 21);
this.cbCaptureOneTime.Name = "cbCaptureOneTime";
this.cbCaptureOneTime.Size = new System.Drawing.Size(52, 17);
this.cbCaptureOneTime.TabIndex = 15;
this.cbCaptureOneTime.Text = "Once";
this.cbCaptureOneTime.UseVisualStyleBackColor = true;
this.cbCaptureOneTime.CheckedChanged += new
System.EventHandler(this.cbCaptureOneTime_CheckedChanged);
//
// btnExit
//
this.btnExit.Font = new System.Drawing.Font("Microsoft Sans Serif", 12F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte) 0));
this.btnExit.Location = new System.Drawing.Point(118, 315);
this.btnExit.Name = "btnExit";
this.btnExit.Size = new System.Drawing.Size(107, 34);
this.btnExit.TabIndex = 14;
this.btnExit.Text = "Exit";
this.btnExit.UseVisualStyleBackColor = true;
this.btnExit.Click += new System.EventHandler(this.btnExit_Click);
//
// btnDisconnect
//
this.btnDisconnect.Font = new System.Drawing.Font("Microsoft Sans Serif",
12F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte) 0));
this.btnDisconnect.Location = new System.Drawing.Point(196, 49);
this.btnDisconnect.Name = "btnDisconnect";
this.btnDisconnect.Size = new System.Drawing.Size(107, 34);
this.btnDisconnect.TabIndex = 15;
this.btnDisconnect.Text = "Disconnect";
this.btnDisconnect.UseVisualStyleBackColor = true;
this.btnDisconnect.Click += new
System.EventHandler(this.btnDisconnect_Click);
//
// panel1
//
this.panel1.BorderStyle = System.Windows.Forms.BorderStyle.FixedSingle;
this.panel1.Location = new System.Drawing.Point(12, 12);
this.panel1.Name = "panel1";
this.panel1.Size = new System.Drawing.Size(342, 78);
this.panel1.TabIndex = 16;
//
// panel2
//
this.panel2.BorderStyle = System.Windows.Forms.BorderStyle.FixedSingle;
this.panel2.Controls.Add(this.edPath);
this.panel2.Controls.Add(this.label6);
this.panel2.Controls.Add(this.label3);
this.panel2.Location = new System.Drawing.Point(12, 137);
this.panel2.Name = "panel2";
this.panel2.Size = new System.Drawing.Size(342, 172);
this.panel2.TabIndex = 17;
//
// edPath
//
this.edPath.Font = new System.Drawing.Font("Microsoft Sans Serif", 9.75F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte) 0));
this.edPath.Location = new System.Drawing.Point(124, 7);
this.edPath.Name = "edPath";
this.edPath.Size = new System.Drawing.Size(209, 22);
this.edPath.TabIndex = 19;
//

```

```

        // label6
        //
        this.label6.AutoSize = true;
        this.label6.Font = new System.Drawing.Font("Microsoft Sans Serif", 12F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)0));
        this.label6.Location = new System.Drawing.Point(7, 7);
        this.label6.Name = "label6";
        this.label6.Size = new System.Drawing.Size(42, 20);
        this.label6.TabIndex = 18;
        this.label6.Text = "Path";
        //
        // Form1
        //
        this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
        this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
        this.ClientSize = new System.Drawing.Size(366, 358);
        this.Controls.Add(this.btnDisconnect);
        this.Controls.Add(this.btnExit);
        this.Controls.Add(this.groupBox2);
        this.Controls.Add(this.edFilename);
        this.Controls.Add(this.btnConnect);
        this.Controls.Add(this.label5);
        this.Controls.Add(this.label4);
        this.Controls.Add(this.lbStatus);
        this.Controls.Add(this.label1);
        this.Controls.Add(this.cbSerialPortNum);
        this.Controls.Add(this.panel1);
        this.Controls.Add(this.panel2);
        this.Name = "Form1";
        this.Text = "Data Acquisition GUI";
        this.Load += new System.EventHandler(this.Form1Load);
        this.groupBox2.ResumeLayout(false);
        this.groupBox2.PerformLayout();
        this.panel2.ResumeLayout(false);
        this.panel2.PerformLayout();
        this.ResumeLayout(false);
        this.PerformLayout();

    }

#endregion

private System.Windows.Forms.ComboBox cbSerialPortNum;
private System.Windows.Forms.Label label1;
private System.Windows.Forms.Label label3;
private System.Windows.Forms.ListBox lbStatus;
private System.Windows.Forms.Label label4;
private System.Windows.Forms.Label label5;
private System.Windows.Forms.Button btnConnect;
private System.Windows.Forms.TextBox edFilename;
private System.Windows.Forms.GroupBox groupBox2;
private System.Windows.Forms.CheckBox cbCaptureTwentyTimes;
private System.Windows.Forms.Button btnCaptureTrace;
private System.Windows.Forms.CheckBox cbCaptureTenTimes;
private System.Windows.Forms.CheckBox cbCaptureOneTime;
private System.Windows.Forms.Button btnExit;
private System.Windows.Forms.Button btnDisconnect;
private System.Windows.Forms.Panel panel1;
private System.Windows.Forms.Panel panel2;
private System.Windows.Forms.TextBox edPath;
private System.Windows.Forms.Label label6;
}
}

```

C.8 Performance Evaluation Setup with Fully-Specified Connections of Cables

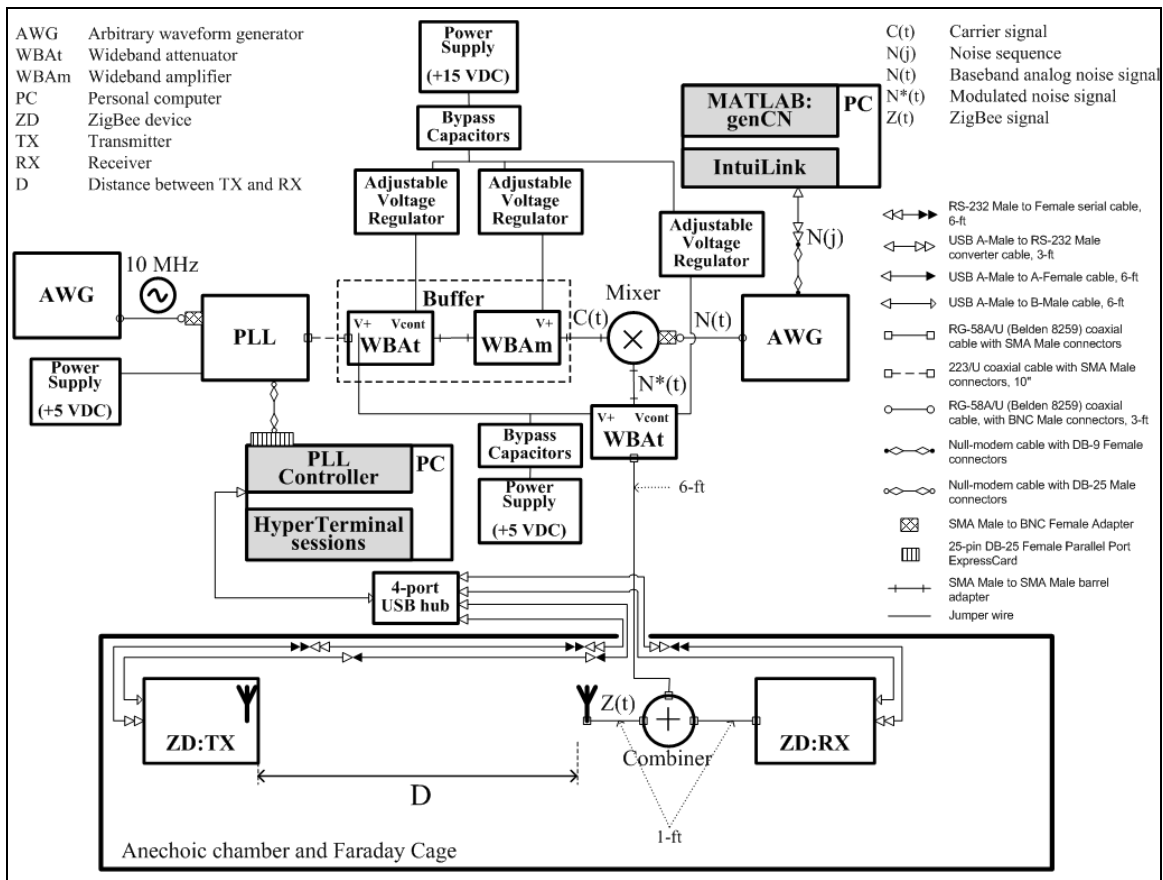


Fig. 1 Supplementary to Fig. 5.2 with types of cables used shown

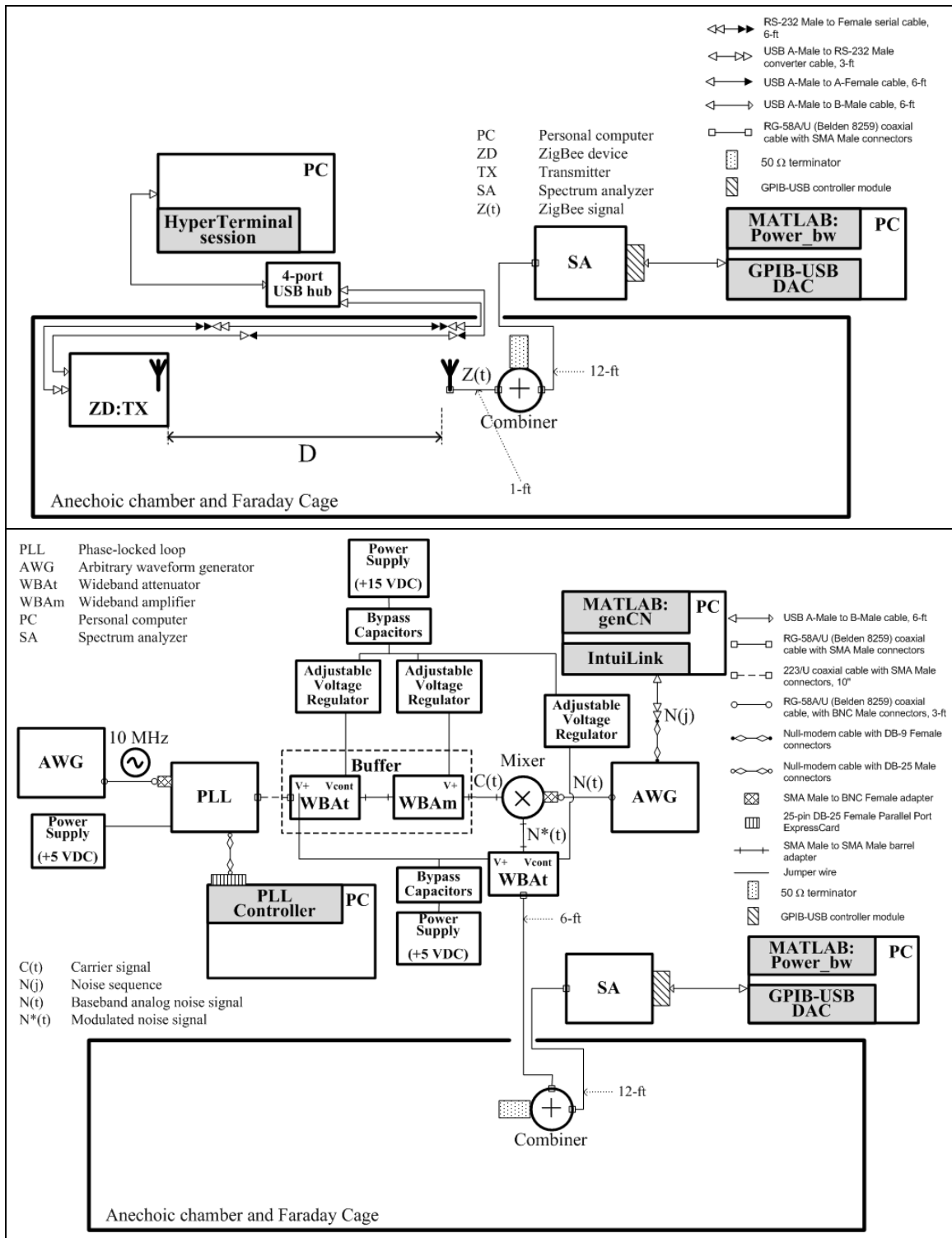


Fig. 2 Supplementary to Fig. 5.3 with types of cables used shown

C.9 Performance Evaluation Hardware and Software Listing

Purpose	Quantity	Name	Manufacturer
Cables	4 2 2 1 1 1 2 1 1 1 3 3	- USB 2.0 A-M to B-M, 6-ft - USB 2.0 A-M to A-F, 6-ft - RG-58A/U (Belden 8259) coaxial cable with SMA Male connectors, 1-ft - RG-58A/U (Belden 8259) coaxial cable with SMA Male connectors, 6-ft - RG-58A/U (Belden 8259) coaxial cable with SMA Male connectors, 12-ft - 223/U coaxial cable with SMA Male connectors, 10-in - RG-58A/U (Belden 8259) coaxial cable with BNC Male connectors, 3-ft - RS-232 F-F, 6-ft, null-modem cable with DB-9 Female connectors - RS-232 F-F, 6-ft, null-modem cable with DB-25 Female connectors - RS-232 M-M, 10-ft, null-modem cable with DB-25 Male connectors - RS-232 M-F serial cable, 6-ft - USB A-M to RS-232 Male converter, 3-ft	All supplied by ECE tech shop
Connectors or Adapters	many 3 2 1 1	- Jumper wires - SMA-M to SMA-M barrel adapter - SMA-M to BNC-F adapter - N-BNC (used at HP 8546A) - N-SMA (used at HP 8546A)	All supplied by ECE tech shop
Test Equipments	1 2 1 1 1	- Oscilloscope (TDS 210) - Arbitrary Waveform Generator (33120A) - Spectrum Analyzer (HP 8546A) - Multimeter – True RMS Multimeter (112) - Wi-Spy Spectrum Analyzer	Tektronix Agilent Agilent FLUKE MetaGeek
Modules	1 1 1 1 1 1 1 2 1 2 1 1 1	- Laptop - 25-pin DB-25 Female Parallel Port ExpressCard (SPPXP-100 ExpressCard IEEE-1284 Parallel Adapter) [Quat] - PLL Evaluation Board (PLL400-2450A) [Sire] - Buffer (1 Attenuator: ZX73-2500, followed by 1 Amplifier: ZX60-4016E) [Mini] - Mixer (ZX05-30W) [Mini] - Attenuator (ZX73-2500) [Mini] - Combiner (ZN2PD2-50) [Mini] - Transceiver Boards (13192-EVB) [Free] - BDM Debugger – USB HCS08/HCS12 Multilink, Rev. B 0105 (USB-ML-12) [PEMi] - Antenna (Microstrip Patch), 2 MHz-span, $f_{\text{resonant}} = 2.405$ GHz - USB hub, 7-port, self-powered - 50 Ω Terminator 28S50-2, SMA BB (Ser. 805852) - GPIB-USB Controller Module, v4.2 [Prol]	Dell Quatech RFMD, previously Sirenza Mini-Circuits Mini-Circuits Mini-Circuits Mini-Circuits Freescale P&E Microcomputer Systems Custom-made [BeKZ06] Belkin Wiltron Prologix
Power Supply	1 1	- DC Voltage (+5V) - DC Voltage (+11V)	All supplied by ECE tech shop
Voltage Regulator	2 1	- With Improved Ripple Rejection (Figures 1 and 3 of [T1]) - With Minimum Program Current (Figure 6 of [T1])	Circuits put together by author
Physical Noise Source	1	- Plasma Globe [Dream]	Dream Cheeky, with 4-ft of USB cable
Miscellaneous	1 3	- Metal Box - Power Bar	All supplied by ECE tech shop

Table 1 Listing of Recommended Hardware for PE under synthetic noise and under physical noise.

Purpose	Type	Name	Manufacturer
Captured noise experiments: beta for noise to generate	Proprietary	- IQTRead Distribution May 2006.zip [Simm06] - MATLAB software	Tektronix The MathWorks, Inc.
	Created by author	- convertIQTtoTimeSeries - VFD function - VFDT function - Main program	Written in MATLAB Written in MATLAB Written in MATLAB Written in MATLAB
PER experiments: analog baseband noise	Proprietary	- IntuiLink software for Agilent 33120A Waveform Generator, v1.5 - Agilent IO Libraries Suite 15.0 [Agil] - Agilent IO Libraries Suite 15.0 Patch 1 [Agil] - MATLAB software - TextPad, v5.2.0, c1993 (for viewing values of noise sequence samples)	Agilent Agilent Agilent The MathWorks, Inc. Helios Software Solutions
	Created by author	- monofractalNoiseGeneration function - Main program	Written in MATLAB Written in MATLAB
PER experiments: analog high-frequency noise	Proprietary	- PLL controller software, v4.0, c2002	Vari-L Company, Inc.
PER experiments: PER measurements	Proprietary	- CodeWarrior for Microcontroller, evaluation version, v6.2 - BeeKit (use SMAC codebase) - P&E hardware drivers, v9.02 [P&E] - HyperTerminal (for communication over serial COM ports), v6.3, Private Edition - Excel (for entering results)	Freescale Freescale P&E Microcomputer Systems Hilgraeve, Inc. Microsoft Corporation
	Created by author	- PER monitoring program	Written in C
PER experiments: SNR measurements	Proprietary	- Visual Studio C#	Microsoft Corporation
	Created by author	- GPIB-USB trace data acquisition program - Power within bandwidth	Written in C# Written in MATLAB
Operating system on laptop	Proprietary	- Windows Vista	Microsoft Corporation

Table 2 Listing of Recommended Software for PE under synthetic noise.

Reference

[Quat] For purchasing parallel ExpressCard, Online:

<http://www.sirenta.com/default.aspx?pageid=31&categoryid=92&productid=PLL400-2450A>

[Sire] For purchasing PLL evaluation board, Online: http://www.quatech.com/catalog/expresscard_parallel.php

[Mini] For purchasing Mini-Circuits products, Online: <http://www.minicircuits.com/>

[Free] For purchasing Freescale 13192 transceiver board, Online:

http://www.freescale.com/webapp/sps/site/prod_summary.jsp?code=MC13192&nodeId=01J4Fs25658166

[PEMi] For purchasing background debugger mode module, Online: www.pemicro.com

[Pro] For purchasing GPIB-USB controller module, Online: <http://store.prologix.biz/gpco.html>

[TI] Texas Instrument, LM 317, data sheet, Online: <http://focus.ti.com/lit/ds/symlink/lm317.pdf>

[Dream] For purchasing plasma globe, Online: www.dreamcheeky.com

[Agil] For downloading Agilent IO Libraries Suite 15.0 and its patch, Online:

<http://www.home.agilent.com/agilent/editorial.jsp?cc=US&lc=eng&ckey=1184883&nid=-11143.0.00&id=1184883>

[P&E] For downloading P&E hardware drivers, Online:

http://www.pemicro.com/products/product_viewDetails.cfm?product_id=33&menu_id=details&CFID=315131&CFTOKEN=62565445

[Simm06] J. Simmons, "IQTRead: .IQT conversion software," Tektronix, May 2006.

C.10 Determining Playback Frequency for Arbitrary Waveforms on AWG

Objective:

Determine the maximum playback frequency for arbitrary waveforms on Agilent 33120A in order to avoid dropping/skipping arbitrary waveform data points.

Calculation:

The formula for computing maximum playback frequency for arbitrary waveforms is:

$$F_{out} = \frac{F_{clk}}{Points}$$

Note: According to User's Guide, pg 276 (pdf-pg 278) [1], this formula is for calculating the maximum output frequency aka maximum useful output frequency.

Given:

- Sample Rate for Arbitrary Waveforms = 40 MSamples/sec
- Number of waveform points = 16,000

Note: Since according to User's Guide, pg 278 (pdf-pg 280) [1]:

“In fact, it is generally best to create arbitrary waveforms which use all available data (16,000 points long and the full range from 0 to 4,095 DAC codes).”

Thus:

$$F_{out} = \frac{40,000,000 \text{ Samples / sec}}{16,000 \text{ Samples}}$$
$$F_{out} = 2500 / \text{sec} = 2.5 \text{ k/sec} = 2.5 \text{ kHz}$$

Relevance to Research Work (i.e. Why Should I Care about This?):

$F_{out}=2.5$ kHz is the maximum playback (output) frequency for the 16000 datapoints of our monofractal noise. This tells us that if we set the function generator output frequency beyond the maximum playback frequency (i.e. 2.5 kHz in our case), the number of output samples per waveshape cycle will decrease (i.e. some data points get dropped/skipped). The result is different groups of points being output on successive waveform cycles, which means that the output signal is no longer the intended input noise signal. Therefore, to ensure that when we input say, data point of a brown noise (monofractal with spectral exponent, $\beta=2$), that the output frequency remains a brown noise analog signal, the choice of output frequency to use ranges between 100 μ Hz (i.e., 0.0001 Hz) and maximum playback frequency (i.e. 2.5 kHz) , inclusive. In other words, the calculated maximum output frequency ensures that every waveshape point in RAM is output for every waveform cycle. Note: According to User's Guide, pg 298 (pdf-pg 300) [1], the frequency range for 12,288 to 16,000 points for arbitrary waveforms is 100 μ Hz – 200 kHz.

References:

- [1] Agilent 33120A 15 MHz Function / Arbitrary Waveform Generator User's Guide, publication number 33120-90006, ed. 6, March 2002, available online:
http://www.home.agilent.com/upload/cmc_upload/All/6C0633120A_USERSGUIDE_ENGLISH.pdf

APPENDIX D: RESULTS COMPILED IN TABULAR FORM

The tables that were used to graph the PvS_{RX} and MvS_{TX} plots can be found in the Excel file in the attached CD-ROM under the Appendices\Results directory.

APPENDIX E: EXPERIMENTAL SNAPSHOTS

The snapshots taken during the running of the performance evaluation experiments can be found in the attached CD-ROM under the Appendices\Photos directory.