MODELING AND CUSTOM DATASET GENERATION OF THE DNP3 SCADA PROTOCOL

BY

ARTURO CUEVAS

THESIS

Submitted in partial fulfillment of the requirements

for the degree of Master of Science in Electrical and Computer Engineering

in the Graduate College of the

University of Illinois Urbana-Champaign, 2023

Urbana, Illinois

Adviser:

Professor David M. Nicol

ABSTRACT

The research objectives of this study are two-fold. First, we aim to gain a deeper understanding of the normal traffic patterns of the DNP3 protocol. Second, we aim to evaluate the efficacy of the cycle detection and DTMC methods in generating synthetic traffic that is representative of these normal patterns.

This thesis focuses on the generation of synthetic DNP3 traffic that is representative of real-world scenarios. It employs two methods: cycle detection and Discrete Time Markov Chains (DTMC).. By harnessing these methods, the aim is to create synthetic traffic that mirrors the behavior of actual DNP3 traffic.

The need for realistic synthetic DNP3 traffic extends beyond the scope of cybersecurity. It also plays an instrumental role in network management, aiding in the development of efficient network architectures, and in performance evaluation, helping to improve the Quality of Service (QoS) in SCADA systems.

SCADA (Supervisory Control and Data Acquisition) systems form the backbone of many critical infrastructures including power, water, and gas utilities, to name a few.

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my thesis advisor Professor David M. Nicol for his invaluable guidance and support in my graduate studies. Professor Nicol displayed immense patience, consideration, and kindness to me during the completion of my degree that I hope I can repay to the future generation of researchers.

As well I would to thank the support of my fellow research group members at the Information Trust Institute: David Emmerich, Matthew Needham, and Logan Marlow, for their support and expertise in the completion of my thesis. I also would like to thank the endless support from my family and friends who made this possible.

TABLE OF CONTENTS

[CHAPTER 1 INTRODUCTION 1](#_Toc140061204)

[CHAPTER 2 BACKGROUND 3](#_Toc140061205)

[2.1 Industrial Control System (ISC) 3](#_Toc140061206)

[2.2 ICS Communication Network 4](#_Toc140061207)

[2.2.1 DNP3 Protocol 6](#_Toc140061208)

[2.3 Previous Methods for Traffic Generation 8](#_Toc140061209)

[2.3.1 DTMC 10](#_Toc140061210)

[2.3.2 Cycle Detection 11](#_Toc140061211)

[2.4 Role of Scapy in Traffic Analysis 11](#_Toc140061212)

[CHAPTER 3 MOTIVATION 11](#_Toc140061213)

[3.1 Importance of DNP3 Traffic Analysis 12](#_Toc140061214)

[3.2 Current Challenges in DNP3 Traffic Analysis 12](#_Toc140061215)

[3.3 Proposed Methodology for Analysis 12](#_Toc140061216)

[CHAPTER 4 CUSTOM DNP3 LIBRARY 13](#_Toc140061217)

[4.1 Building the Custom Scapy Library 13](#_Toc140061218)

[4.2 Utility of Scapy in DNP3 Traffic Analysis 13](#_Toc140061219)

[4.3 Benefits and Limitations of the Custom Library 13](#_Toc140061220)

[CHAPTER 5 SYNTHETIC TRAFFIC GENERATION 13](#_Toc140061221)

[5.1 Cycle Detect Method for Traffic Generation 13](#_Toc140061222)

[5.2 DTMC Method for Traffic Generation 13](#_Toc140061223)

[5.3 Comparative Analysis of the Two Methods 13](#_Toc140061224)

[CHAPTER 6 CASE STUDY: 13](#_Toc140061225)

[6.1 Evaluation Criteria 13](#_Toc140061226)

[6.2 Performance Analysis 13](#_Toc140061227)

[6.1 Experiment Setup 13](#_Toc140061228)

[6.2 DNP3 Traffic Analysis using Cycle Detection 13](#_Toc140061229)

[6.3 DNP3 Traffic Analysis using DTMC 14](#_Toc140061230)

[6.4 Comparative Analysis of Both Methods 14](#_Toc140061231)

[6.5 Discussion on Results 14](#_Toc140061232)

[CHAPTER 7 CASE STUDY/DISCUSSION 17](#_Toc140061233)

[CHAPTER 8 CONCLUSION 17](#_Toc140061234)

[8.1 Summary of Findings 17](#_Toc140061235)

[8.2 Implications and Applications 17](#_Toc140061236)

[8.3 Recommendations for Future Research 17](#_Toc140061237)

[RERFERENCES 17](#_Toc140061238)

# CHAPTER 1 INTRODUCTION

Our modern society’s critical infrastructure is heavily dependent on Supervisory Control and Data Acquisition (SCADA) systems. These systems use protocols that are designed to provide reliable and secure communication across various sectors of critical infrastructure such as utilities, water treatment, oil, and gas. These SCADA protocols have become the main communication medium of our society’s most valuable assets for over thirty years, and their security is constantly in question as nation’s are continuing to experience cyber-attacks. Malicious actors have been able to exploit both human and technical weakness to cut off access or damage critical infrastructure.

Cyber-attacks such as Stuxnet (2010) and Ukraine Power Grid (2015) were catalysts for a boom in research dedicated to understanding, predicting, and mitigating potential attacks on SCADA systems. A large component of this research is dedicated to the design of intrusion detection systems, stress testing, and training of various SCADA protocols.

A vital resource in conducting research on the security of SCADA protocols is the availability of realistic SCADA traffic. However, this traffic is scarce as the deployment of real SCADA systems, such as a power grid substation, are expensive, time consuming, or heavily restricted. Also, the access to network traffic from active SCADA systems is often not allowed for privacy reasons. This has grown the need for simulation/emulation systems such as CORE or Mini-mega which simulate real SCADA components and traffic. Thus, the synthetic generation of realistic network traffic is vital for critical infrastructure research.

One of the most popular SCADA protocols is the Distributed Network Protocol 3 (DNP3) which provides communication for utilities such as water and electricity. However, given its popularity the amount of research on generating syntenic traffic is extremely limited. Research in other SCADA protocols such as Modbus, have explored utilizing various statistical methods for generating traffic.

This thesis aims to contribute to the synthetic generation of SCADA traffic that is realistic, inexpensive, and adjustable. A focus on DNP3 traffic is made and is impactful as DNP3 is the leading SCADA protocol for power utilities across the United States. Two distinct methods for creating synthetic DNP3 traffic are explored: cycle detection and Discrete Time Markov Chain (DTMC). These methods were implemented using a custom-built library that is an extension of Scapy, an open-source Python-based packet manipulation program and library.

This thesis makes the following contributions:

* A custom DNP3 library called DNP3 Dissector that is an extension to the Scapy library written in Python and is capable of both reading previously captured network traces and creating new network traces.
* A synthetic DNP3 traffic generator that is based on detecting cycles in the traffic by utilizing various methods such as K-Means clustering.
* A second synthetic DNP3 traffic generator that is based on using Discrete Time Markov Chains (DTMC) that makes predictions based on a transition matrix.

The thesis is structured as follows:

* Chapter 2 provides an overview of the topics and existing literature needed to understand the work.
* Chapter 3 showcases the motivation for this work and its potential impact.
* Chapter 4 goes over the customer DNP3 library necessary to conduct analysis on DNP3 traffic.
* Chapter 5 discusses the design and implementation of both traffic generators.
* Chapter 6 presents the results of our experiments, including a comparison of the two methods based on several metrics.
* Chapter 7 provides a conclusion and suggestion for future work.

# CHAPTER 2 BACKGROUND

## 2.1 Industrial Control System (ISC)

Industrial Control Systems (ISC) are an integral piece of our modern-day world as they provide the access to critical resources like water, electricity, and gas. At their core an *ICS* is as a group of devices that are strung together with the goal of monitoring and automating key industrial processes. These processes are often physical like the closing and opening of a breaker or verifying the temperature of a power plant. These systems are composed of both software and hardware components that communicate with other devices or to a Human Machine Interface (HMI).

A popular type of ICS are Supervisory Control and Data Acquisition (SCADA) systems, which are used in various industries like energy, water treatment, and transportation. At a high-level SCADA systems are composed of a supervisor that controls the local operations of field devices. In SCADA systems, the distances between these devices are often quite long and can consist of multiple layers of primary and secondary components. For example, power grid substations have a Remote Terminal Unit (RTU) that controls multiple breaker relays, which in turn monitor the status of the electrical line. Commands and data collected are then intergraded into an HMI for maintenance.

A diagram of a network

Description automatically generated

## 2.2 ICS Communication Network

The communication network of ICS can be spilt into two components, a **Supervisory Control Network**, and a **Field Communications Network**. The Supervisory Network is the highest layer and consists of computer-based controllers like Remote Terminal Units (RTUs), Intelligent Electronic Devices (IEDs), or Programmable Logic Controllers (PLCs). These devices then communicate with a higher layer of supervisory devices such as Master Terminal Units (MTUs) or HMIs. The Field Communications Network consists of sensors and actuators (breakers, values, etc.) that communicate with the controller devices (RTU, IED, PLC) with time sensitive field communication protocols.

ICS protocols have a major separation in design from traditional network protocols due to system requirements. The three pillars of data security (Confidentiality, Availability, and Integrity) are often used to define a system’s operational priorities. In traditional network protocols, like HTTP, the priorities are Confidentiality (C), Integrity (I), and then Availability (A). However, due to the critical services ICS systems provide Availability (A) is a top focus and following is the integrity (I) of the data. Confidentiality (C) for ICS protocols were a low priority item and encryption schemes were only added after their conception.

For much of their history ICS networks where isolated and used serial and analog channels. However, with the advent of fast and reliable telecommunications, current ICS networks utilize advanced communication channels like ethernet or TCP. Due to the open standard of these telecommunication mediums, ICS systems like SCADA have seen a fragmented market of protocols.

In the manufacturing sector, Modbus is a common protocol that enables communication among many devices connected to the same local network. For instance, a ICS measuring temperature and humidity in a production area. In the utilities sector, DNP3 provides reliable communications between various types of data acquisition and control equipment. It allows for effective control over processes such as water treatment and power distribution. The power systems sector often uses the IEC 61850 protocol, which is specifically designed for substation automation, enabling intercommunication between IEDs (Intelligent Electronic Devices) from different manufacturers. Finally, the automation industry commonly employs the OPC (OLE for Process Control) protocol, which is a series of standards and specifications for industrial telecommunication. OPC allows real-time data from control devices to be easily shared between different types of hardware and software.

The effectiveness and security of SCADA systems rely heavily on the communication protocol that facilitates the data transfer between the supervisory and field networks. Consequently, the DNP3 protocol, given its widespread use, becomes a crucial point of interest when considering the cybersecurity of power distribution.

### 2.2.1 DNP3 Protocol

The Distributed Network Protocol (DNP3) is an open, standards-based communications protocol that is utilized extensively in utilities such as water, power grid distribution, and gas. Developed in the 1990’s, DNP3 was one of the first public standards in a time when most protocols where private and vendor defined. DNP3 was designed for long distance wide area networks like power grids which can be on the order of miles. This is in contrast with other SCADA protocols such as Modbus which were developed for local area networks inside factories.

DNP3 was designed to facilitate communication between various types of data acquisition and control instrumentation. In the beginning, DNP3 was designed for serial links such as radio. This enabled the protocol to have reliable and efficient data transmission even over potentially unreliable communication channels. By 1998, DNP3 supported communication over TCP/IP and UDP/IP. This update came as a double-edged sword as DNP3 experienced the benefits of a state-of-the-art telecommunication protocol but was also exposed to their vulnerabilities and limitations.

A diagram of a diagram of a machine

Description automatically generated

The DNP3 protocol is divided into three layers: **Data Link Layer**, **Transport Control Layer**, and **Application Layer**. In an TCP/IP network, the Data Link Layer lies between the transport layer (TCP) and the Transport Control Layer. This interface allows for management of flow control, error handling, frame synchronization, and link status. The structure of the Data Link Layer consists of a header block and then optional data block(s) with a maximum size of 16 bytes. The head is made of 5 fields: START, LENGTH, CONTROL, DESTINATION, and SOURCE. The START field is a fixed 2-byte sequence (0x05 and 0x64) that indicates the start of a DNP3 packet. The CONTROL field provides information on packet direction, transaction indication, error code, flow control, and function type. The DESTINATION and SOURCE fields represent DNP3 addresses and are on the range from 1 to the maximum number of relays. At the end of the header block is a CRC (Cyclic Redundancy Check) checksum for error checking. After the header are the data blocks which constitute the payload of the Data Link Layer. Each data block has a 2-byte CRC that verifies the integrity of the block.

The Transport Control Layer acts as an interface to assemble fragments from the Data Link Layer and dissemble fragments from the lower Application Layer. The layer controls these segments by having three fields: FIR (for first segment), FIN (for final segment), and SEQUENCE (identification used to determine segment order).

The Application Layer is the highest layer in the protocol and interacts directly with application processes. It defines the format of the data packets exchanged and controls the types and orders of messages. This layer handles data segmentation and reassembly, and it ensures the reliability of communication. The functions supported at an application level include data read, data write, time synchronization, and system restart operations. It also manages the control codes used for communication between DNP3 master and DNP3 outstation, such as confirming received data, requesting data, and responding to data requests.

Other key features of DNP3 include time synchronization, which is crucial for processes that need coordinated real-time control (RTC). Also, the DNP3 standard has gone through many iterations to add encryption over the protocol and is settling on a form of Diffie-Hellman Public Key Exchange. One of the unique characteristics of DNP3 is its ability to send data without being requested by the controlling computer, a feature called unsolicited reporting. This allows for more efficient communication and faster response times. DNP3 also includes robust error checking and correction mechanisms that increase the reliability of data transmissions.

## 2.3 Previous Methods for Traffic Generation

Synthetic network traffic generation has been a vital part of network management and research since the advent of digital networking. Historically, as networks grew more complex and the number of connected devices increased, it became necessary to understand and quantify network performance and security under various conditions. Thus, the need for synthetic network traffic generation arose. According to [], *synthetic traffic* refers to artificially created network data intended to mimic real-world traffic patterns. The ability to generate and analyze this kind of traffic enables researchers and network administrators to stress-test networks, model potential usage scenarios, and understand how changes in network configurations might impact performance.

Although ICS network traffic makes a small share of total network traffic and the traffic generation research space, it also plays a critical role in the testing and management of SCADA systems []. Given the critical nature of many SCADA systems, the ability to accurately generate representative SCADA network traffic for testing and simulation purposes is essential for ensuring system reliability and security.

A successful synthetic network traffic generation should accurately reflect real-world conditions, traffic patterns, and user behaviors. The generated traffic should vary in size, frequency, and other characteristics to adequately describe all aspects of the network []. Consequently, many traffic generation methods exist and can vary widely. Approaches range from simple repeated messages based on flow averages to complex statical models that account for user behavior, time of day, and even varying packet sizes.

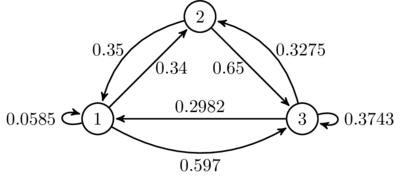
There are several modeling techniques used in generating synthetic network traffic and each carry unique characteristics and benefits. Deterministic models, like Discrete Markov Chains (DTMC) or Deterministic Finite Automata (DFA) provide a systematic approach and base transitions based on states that follow a fixed probability distribution. Machine learning models are an advanced method that utilized neural networks based on the intricate relations and dependencies in the network traffic. These models can capture implicit traffic patterns but lack the sense of a structure network pattern []. In flow-based generation, the traffic is looked at a flow level between a source and destination. Generation is accomplished by considering aspects like flow duration, packers per flow, and bytes per packet []. The modeling techniques explored in this thesis are expanded below.

### 2.3.1 DTMC

Discrete-Time Markov Chains (DTMC) provide a simple and systematic approach to synthetic traffic generation, including the creation of traffic for SCADA networks. A DTMC is a sequence of random variables where the value at each time step depends only on the state at the previous time step. This “memoryless" property makes DTMCs an excellent choice for modeling scenarios where the future state depends solely on the current state and not on the sequence of events that preceded it [].

In the context of synthetic SCADA network traffic generation, DTMCs can be utilized to model the sequences of commands and responses that characterize SCADA communication patterns. For example, a DTMC might represent the states of a SCADA device (idle, waiting for command, processing command, sending response) and the transitions between those states. The probabilities of these transitions can be estimated from real-world traffic data to create a model that realistically mimics SCADA network communication.

A particularly relevant application of DTMCs in SCADA traffic generation was described in [], which employed DTMCs for creating synthetic Modbus traffic. The authors of [] used DTMCs to model the sequences of Modbus function codes seen in real-world network traffic. The DTMC was trained on a dataset of network traffic, with states representing different Modbus function codes and the transitions between them reflecting the likelihoods of those function codes following each other in the network traffic. This approach allowed the authors to generate synthetic Modbus traffic that closely mirrored the statistical characteristics of the real-world Modbus traffic on which the model was trained.



### 2.3.2 Cycle Detection

The nature of SCADA traffic, specifically DNP3, is quite cyclic in nature. The traffic centers around a polling scheme from the DNP3 master device (RTU) to a DNP3 slave device (PLC). For this reason, researchers [] have attempted to rely on the strong cyclic patterns of Modbus traffic to produce synthetic traffic and an intrusion detection system (IDS). The basis of this modeling technique is called cycle detection and is a type of flow-based modeling that attempts to discover the cycles of messages within short- or long-lived connection.

The model tool created in [] is called PeriodAnalyzer and starts analyzing real-world Modbus traffic. The researchers considered multiple factors based on what is expected of normal Modbus network traffic. For instance, there can be multiple cycles per flow, and each with different periodicity levels.

The Period Analyzer tool is divided into three independent modules. First, the **Multiplexer Module** is tasked with filtering out non-Modbus packets from the traffic and sorting each packet by their flow. The authors described a flow as the tuple (Server Address, IP Protocol, Server Port, Client Address, Client Port), and make note of whether the flow is short- or long-lived connection. In their experimentation, a threshold of one second was enough to divide the two types of connections. Second, the **Tokenizer Module** takes in the list packets group by each flow from the multiplexer module and adds relevant tokens to identify pairs of messages. These pair tokens identify the communication sequence of a RTU requesting data from a relay and the RTU responding back. Lastly, the Learner Module

Due to the regular transmission of specific types of data packets in Modbus systems, their model was able to evaluated against real-world Modbus traffic as an IDS. This system raises alarms when it detects new cycles or when current cycles begin to display abnormal behavior.

it considers traffic at the macro level rather than individual packets.

## 2.4 Role of Scapy in Traffic Analysis

For the implementation of these methods and the analysis of network traffic, this study leverages Scapy, a powerful Python-based interactive packet manipulation program and library. Scapy's flexibility and extensive features make it an ideal choice for custom traffic analysis and generation. In order to enhance Scapy's functionality in the context of DNP3 traffic, a custom library was built.

Scapy is a powerful Python-based network tool that allows users to send, sniff, dissect, and forge network packets. As an interactive packet manipulation program, it enables packet construction through a high-level and flexible interface. It can handle tasks ranging from network scanning to probing, tracerouting to unit testing, and network discovery to attack detection.

In the context of SCADA (Supervisory Control and Data Acquisition) systems, Scapy's capacity to construct and manipulate network packets makes it a valuable tool in the generation of synthetic SCADA network traffic. Using Scapy, researchers and engineers can construct network packets that mimic the patterns found in real-world SCADA traffic.

One notable application of Scapy in this domain is detailed in a paper where it was employed to generate synthetic SCADA network traffic based on an analysis of real, normal network traffic. The authors utilized Scapy to dissect real-world SCADA network traffic, extracting key characteristics and patterns. They then used Scapy to generate synthetic network packets that maintained these patterns, effectively creating a synthetic traffic flow that closely mimicked the behavior of the analyzed real-world SCADA traffic.

Through such use cases, Scapy serves as a versatile and effective tool for studying SCADA systems, allowing researchers to create realistic synthetic network traffic for testing and analysis. Its flexibility and functionality provide a strong foundation for ongoing efforts to improve the robustness and security of SCADA systems.

# CHAPTER 3 MOTIVATION

To ensure the robustness of these systems, it's crucial to understand, monitor, and test DNP3 network traffic. However, generating synthetic, yet representative DNP3 traffic for analysis and testing poses significant challenges. Most synthetic traffic lacks the complexity and realistic nature of actual network traffic, thereby reducing the efficacy of the testing process.

They have made it possible to manage large-scale, geographically distributed operations from a centralized location, providing real-time data and control necessary for the smooth functioning of these essential services.

utilities in the United States over 70% of the market share

Despite its benefits, DNP3, like any other communication protocol, is not immune to cyber threats. The very features that make SCADA systems advantageous, such as centralized control and the ability to communicate over long distances, also make them vulnerable. Successful cyber attacks can lead to loss of control over physical processes, disruption of services, and even potential physical damage and hazards.

As the reliance on SCADA systems and the DNP3 protocol increases, so does the need for robust security measures. Understanding the normal traffic patterns of DNP3, being able to generate synthetic yet realistic traffic for testing and analysis, and detecting anomalies that may signal a cyber threat are all crucial aspects of maintaining the cybersecurity of SCADA systems. This forms the basis for the research presented in this thesis.

The successful use of DTMCs in applications like the one described in [] underscores the potential of this approach for generating synthetic SCADA network traffic. By accurately reflecting the sequences of commands and responses that typify SCADA network communication, DTMCs offer a promising avenue for creating realistic, high-fidelity synthetic traffic for SCADA systems.

An extensive search of the literature reveals that the automated approach presented in this paper is the first to directly exploit periodicity in industrial control network traffic. Message repetition and timing information are used to detect periodic cycles in the traffic; this addresses the limitations of existing approaches. The approach is validated using traffic traces collected from operational industrial control networks – two water treatment plant networks and one electric-gas utility network.

3.1 Importance of DNP3 Traffic Analysis

3.2 Current Challenges in DNP3 Traffic Analysis

3.3 Proposed Methodology for Analysis

The ensuing sections of this paper will delve into the intricacies of the methods employed, their implementation, and the results obtained from the experiments conducted. Through comparative analysis, it aims to shed light on the effectiveness of each method in creating synthetic and realistic DNP3 traffic. This work aspires to contribute to the existing body of knowledge and tools in the field of SCADA cybersecurity, providing a solid foundation for further research and improvements.

# CHAPTER 4 CUSTOM DNP3 LIBRARY

4.1 Building the Custom Scapy Library

4.2 Utility of Scapy in DNP3 Traffic Analysis

4.3 Benefits and Limitations of the Custom Library

# CHAPTER 5 SYNTHETIC TRAFFIC GENERATION

5.1 Cycle Detect Method for Traffic Generation

5.2 DTMC Method for Traffic Generation

5.3 Comparative Analysis of the Two Methods

# CHAPTER 6 CASE STUDY:

6.1 Evaluation Criteria

6.2 Performance Analysis

6.1 Experiment Setup

6.2 DNP3 Traffic Analysis using Cycle Detection

6.3 DNP3 Traffic Analysis using DTMC

6.4 Comparative Analysis of Both Methods

6.5 Discussion on Results

A diagram of a network

Description automatically generated

A graph of a number of bars

Description automatically generated

A graph of a packet size distribution

Description automatically generated

A graph of blue and orange bars

Description automatically generated

A graph of blue and orange bars

Description automatically generated

A graph with blue and orange bars

Description automatically generated

A graph with blue and orange bars

Description automatically generated

# CHAPTER 7 CASE STUDY/DISCUSSION

# CHAPTER 8 CONCLUSION

8.1 Summary of Findings

8.2 Implications and Applications

8.3 Recommendations for Future Research

# RERFERENCES

If you have many references, it’s best to use a software package such as RefWorks or OneNote (or BibTeX for LaTeX users). For just a few, following IEEE style, the formatted list below will serve; just replace the information with that from your sources and update the numbering manually. Several different kinds of sources are listed here, so pick the “template” for the kind of source you wish to cite. Number the list according to (1) the order of citation in the main text, or (2) author last name. Note that line spacing for references is narrow. Your advisor may prefer that you use a reference style other that IEEE; for instance, in remote sensing it is customary to use Radio Science (AGU) style, and researchers in microelectronics often use AIP. The software packages mentioned above will output any style. If you are doing it manually, consult the reference style guides provided on the ECE Thesis Check wiki at: <https://wiki.engr.illinois.edu/display/ECEThesisReview/> .

[1] *Motorola Semiconductor Data Manual,* Motorola Semiconductor Products, Inc., Phoenix, AZ, 2007.

[2] *Double Data Rate (DDR) SDRAM,* datasheet, Micron Technology, Inc., 2000. Available at: <http://download.micron.com/pdf/datasheets/dram/ddr/512MBDDRx4x8x16.pdf>

[3] Linx Technologies LT Series, web page. Available at: <http://www.linxtechnologies.com/products/rf-modules/lt-series-transceiver-modules/>. Accessed January 2012.

[4] J. A. Prufrock, *Lasers and Their Applications in Surface Science and Technology,* 2nd ed. New York, NY: McGraw-Hill, 2009.

[5] W. P. Mondragon, “Principles of coherent light sources: Coherent lasers and pulsed lasers,” in *Lasers and Their Applications in Surface Science and Technology,* 2nd ed., J. A. Prufrock, Ed. New York, NY: McGraw-Hill, 2009, pp. 117-132.

[6] G. Liu, “TDM and TWDM de Bruijn nets and shufflenets for optical communications,” *IEEE Transactions on Computers*, vol. 59, no. 1, pp. 695-701, June 2011.

[7] S. Al Kuran, “The prospects for GaAs MESFET technology in dc–ac voltage conversion,” in *Proceedings of the Fourteenth Annual Portable Design Conference*, 2010, pp. 137-142.

[8] K. E. Elliott and C. M. Greene, “A local adaptive protocol,” Argonne National Laboratory, Argonne, IL, Tech. Rep. 916-1010-BB, 2006.

[9] J. Groeppelhaus, “Java 5.7 tutorial: Design of a full adder,” class notes for ECE 290, Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, 2011.

Appendix A Title

If you have an appendix or appendices, place *before* the References *if they contain reference citations*. Do not start a new sequence of page numbers.