

**DIGITAL IMPLEMENTATION OF AN INCREMENTAL
QUANTITY DIRECTIONAL PROTECTIVE RELAY
ELEMENT AND ITS APPLICATIONS USING AN
AUTOMATED TESTING PROCEDURE**

by

Deven Hymel, B.S.

A Thesis Presented in Partial Fulfillment
of the Requirements of the Degree
Master of Science

COLLEGE OF ENGINEERING AND SCIENCE
LOUISIANA TECH UNIVERSITY

May 2023

LOUISIANA TECH UNIVERSITY

GRADUATE SCHOOL

March 27, 2023

Date of thesis defense

We hereby recommend that the thesis prepared by

Deven Hymel, B.S.

entitled **Digital Implementation of an Incremental Quantity Directional**

Protective Relay Element and its Applications Utilizing an Automated Testing

Procedure

be accepted in partial fulfillment of the requirements for the degree of

Master of Science in Engineering, Electrical Engineering Concentration

Prashanna Bhattarai

Prashanna Bhattarai
Supervisor of Thesis Research

Jinyuan Chen

Jinyuan Chen
Head of Electrical Engineering

Thesis Committee Members:

Prashanna Bhattarai

Matthew Hartmann

Mickey Cox

Approved:

Hisham Hegab

Hisham Hegab
Dean of Engineering & Science

Approved:

Ramu Ramachandran

Ramu Ramachandran
Dean of the Graduate School

ABSTRACT

This thesis presents the design and analysis of a digital model of an incremental-quantity-based directional protective relay element using the electromagnetic transient program (EMTP). The digital model is based on a similar protective element present in commercially available time-domain-based relays and provides directional fault detection using incremental quantities. Such models are not readily available in transient study programs.

The element's operating principles are discussed in detail, and the digital design process is outlined. The digital model is designed in a simplified test system, and upon validation using a real-world time-domain-based relay, its performance is analyzed using the model of a real power transmission network. An automation script is developed to perform more robust testing. The development of the automation script is also discussed in detail within this thesis.

The digital model is intended to predict the performance of the same element in commercially available time-domain relays. Model performance was compared with a commercially available relay and the results were found to be similar.

APPROVAL FOR SCHOLARLY DISSEMINATION

The author grants to the Prescott Memorial Library of Louisiana Tech University the right to reproduce, by appropriate methods, upon request, any or all portions of this Thesis. It is understood that “proper request” consists of the agreement, on the part of the requesting party, that said reproduction is for his personal use and that subsequent reproduction will not occur without written approval of the author of this Thesis. Further, any portions of the Thesis used in books, papers, and other works must be appropriately referenced to this Thesis.

Finally, the author of this Thesis reserves the right to publish freely, in the literature, at any time, any or all portions of this Thesis.

Author _____

Date _____

TABLE OF CONTENTS

ABSTRACT	iii
APPROVAL FOR SCHOLARLY DISSEMINATION	iv
LIST OF FIGURES	viii
LIST OF TABLES	xi
ACKNOWLEDGMENTS	xiii
CHAPTER 1 INTRODUCTION	1
1.1 Faults and Transients	1
1.2 Protective Relaying	1
1.2.1 Overcurrent Protection	2
1.2.2 Differential Protection	3
1.2.3 Phasor-Based Line Protection	4
1.2.4 Disadvantages of Phasor-Based Protection and Alternatives	7
1.3 Software and Hardware Utilized in the Project	8
CHAPTER 2 INCREMENTAL QUANTITY DIRECTIONAL ELEMENT PRINCIPLES OF OPERATION	10
2.1 Operating Basics	10
2.1.1 Incremental and Incremental Replica Values	11
2.1.2 Extracting Incremental Values and Calculating Replica Values	13
2.1.3 Calculating Incremental Replica Current Values for Ground Loops	16
2.2 TD32 Circuit Representation	18

2.2.1	Determination of Fault Direction.....	21
2.2.2	Additional Element Security.....	23
CHAPTER 3 DIGITAL MODELING OF AN INCREMENTAL QUANTITY DIRECTIONAL ELEMENT		26
3.1	The Test System.....	26
3.2	Digital Modeling of the Element	28
3.2.1	EMTP Incremental Quantity Filter	30
3.2.2	Ground and Phase Loops	33
3.2.3	Fault Logic	36
3.3	Key Differences Between the Model and the Element.....	37
3.4	Preliminary Verification of the EMTP Model	38
CHAPTER 4 SIMULATION AUTOMATION WITHIN EMTP		45
4.1	Introduction.....	45
4.2	Basic Overview of Automation Scripting.....	45
4.3	Initializing Libraries and Their Functions	48
4.3.1	The “simoptions” Library	49
4.3.2	The “make_file_name” Library	50
4.3.3	The “run_emtp” Library	51
4.3.4	The “check_my_filename” Library	52
4.3.5	The “sw0_m” Library	52
4.3.6	Selecting Devices Within a Circuit Using Scripts	53
4.3.7	Changing Constant Parameter Line Lengths	55
4.4	Using Mplot For Automatic Data Extraction	57
4.4.1	Mplot Command And Output Files	60
4.5	Running EMTP Simulations and Data Extraction.....	61
4.5.1	Generating an Excel File for Results Storage.....	61

CHAPTER 5 TESTING OUTLINE	63
5.1 Introduction.....	63
5.2 The Test System.....	63
5.2.1 Model Settings	65
5.2.2 Testing Procedure	67
5.2.3 Adding Custom Outputs to the LCD Screen of the T401L	68
5.3 Test Outline.....	70
5.3.1 Test 1: Impacts of Fault Inception Angle	70
5.3.2 Test 2: Impacts of fault impedance	70
5.3.3 Test 3: Transition Point Along a Parallel Path.....	70
CHAPTER 6 TEST RESULTS	72
6.1 Test 1 – Impacts of Fault Inception Angle	72
6.2 Test 2 – Impact of Fault Impedance Results and Analysis.....	77
6.2.1 25Ω of Fault Impedance	77
6.2.2 50Ω of Fault Impedance	80
6.2.3 100Ω of Fault Impedance	83
6.2.4 250Ω, 500Ω, and 1000Ω of Fault Impedance	85
6.2.5 Summary.....	85
6.3 Test 3 – Transition Point Along a Parallel Path Results and Analysis	86
6.4 Summary.....	89
CHAPTER 7 CONCLUSIONS AND FUTURE WORK.....	90
7.1 Conclusions.....	90
7.2 Future Work.....	91
BIBLIOGRAPHY.....	92

LIST OF FIGURES

Figure 1-1: Impedance Relay Block and Trip Regions.	5
Figure 1-2: Modified Impedance Relay (mho relay) Block and Trip Regions.....	6
Figure 1-3: Implementing Several Relays Per Phase with Increasing Reach Points.....	7
Figure 2-1: Comparison of Currents in Steady State (top) and Transient (bottom) System.....	11
Figure 2-2: Application of the Superposition Principle to Derive the Fault-Generated Network Using the Pre-fault and the Faulted Networks.....	12
Figure 2-3: Delta Filter.	14
Figure 2-4: RL Fault Network	15
Figure 2-5: Zero Sequence Incremental Current.	16
Figure 2-6: Incremental Replica Current Filter.	17
Figure 2-7: Current Loop Quantity Calculation for A-G Loop.	17
Figure 2-8: TD32 Forward and Reverse Fault Conditions Positive Sequence	19
Figure 2-9: TD32 Operating Logic Diagram for A-G Loop.....	20
Figure 2-10: TD32 Torque Comparison For Forward Fault Conditions	22
Figure 2-11: TD32 Torque Comparison For Reverse Fault Condition.	23
Figure 2-12: TD32 Torque Calculations.....	24
Figure 2-13: TD32 Security Margin	25
Figure 3-1: EMTP Simple System Used For Development.	26
Figure 3-2: Parameters for CP Lines in EMTP Development System.	27
Figure 3-3: Source and Motor Impedances for EMTP Development System.	27
Figure 3-4: EMTP Line Block Model.....	28
Figure 3-5: EMTP Fault Block Model.....	28
Figure 3-6: Top Layer of the TD32 EMTP Model	29

Figure 3-7: Overview of EMTP TD32 Model	29
Figure 3-8: Top Layer of EMTP TD32 Loop Incremental Filter	30
Figure 3-9: Overview of EMTP Incremental Quantity Filter	31
Figure 3-10: EMTP Incremental Voltage Filter	31
Figure 3-11: EMTP Incremental Replica Current Filter.....	32
Figure 3-12: EMTP Weighted Sum Inputs.....	33
Figure 3-13: EMTP Incremental Replica Current Inputs in the T401L Mask.....	33
Figure 3-14: EMTP TD32 Loop Logic.....	34
Figure 3-15: Partial View of EMTP “Fault Logic” Block.....	37
Figure 3-16: Relay Starting Logic Wordbits, viewed in SEL SynchroWAVE.....	40
Figure 3-17: EMTP Model Starting Logic Wordbits.....	41
Figure 3-18: Local Relay EMTP Model Results for A-G Faults Along the Protected Line.	42
Figure 3-19: Remote Relay EMTP Model Results for A-G Faults Along the Protected Line.	43
Figure 3-20: TD32 Word Bit Overview as Seen in the T401L Instruction Manual [7]. .	44
Figure 4-1: Complete Flowchart For Automation Script.....	46
Figure 4-2: Script To Initialize Libraries in EMTP	49
Figure 4-3: Script for Modifying Simulation Options	50
Figure 4-4: EMTP Simulation Options, Basic Data	50
Figure 4-5: Script for Utilizing the “make_file_name” Library To Find The Full Path Of The Netlist File	51
Figure 4-6: Script for Extracting Values from Mplot Results File.....	52
Figure 4-7: Creating A Switch Object And Acting On It.....	53
Figure 4-8: Process used For Finding And Selecting Devices With Scripting.....	54
Figure 4-9: Script Used To Replace The Model And Form Data Of Constant Parameter Lines	56

Figure 4-10: The Format Of The “FormData” String.....	56
Figure 4-11: The Format of The “Model-Data” String.....	57
Figure 4-12: Script Used For Creating Mplot Button and Adjusting Settings.	57
Figure 4-13: Mplot Utility Settings.	58
Figure 4-14: Location Of The Mplot Button In EMTP’s Parts Library	59
Figure 4-15: Process For Generating An Mplot Command File Within An Automation Script.....	59
Figure 4-16: Example Mplot Results File.....	60
Figure 4-17: Example Simulation Results Extracted to Excel	62
Figure 5-1: EMTP Model System Utilized for Testing, Modeled Using Utility Provided Data.....	64
Figure 5-2: EMTP Digital Model Settings	66
Figure 5-3: Setting a timer in the SEL-T401L settings.	68
Figure 5-4: Setting up a latch in the SEL-T401L settings.	69
Figure 5-5: Setting up a custom LCD output for the SEL-T401L.....	69
Figure 6-1: Illustration of Fault Inception Angle on a Voltage Waveform.	72
Figure 6-2: Parallel Path of the Protected Line Highlighted in the EMTP System Model.	86

LIST OF TABLES

Table 3-1: Preliminary Data.....	39
Table 6-1: EMTP results for a L-G fault 60% down the protected line from the Remote relay.	74
Table 6-2: SEL-T401L Relay results for a L-G fault 60% down the protected line from the Remote relay.	74
Table 6-3: EMTP results for a L-L fault 60% down the protected line from the Remote relay.	75
Table 6-4: SEL-T401L Relay results for a L-L fault 60% down the protected line from the Remote relay.	75
Table 6-5: EMTP results for a 3PG fault 60% down the protected line from the Remote relay.	76
Table 6-6: SEL-T401L Relay results for a 3PG fault 60% down the protected line from the Remote relay.	76
Table 6-7: EMTP results for a L-G fault 60% down the protected line from the Remote relay with 25 Ω of fault impedance.	78
Table 6-8: SEL-T401L Relay results for a L-G fault 60% down the protected line from the Remote relay 25 Ω of fault impedance.	78
Table 6-9: EMTP results for a L-L fault 60% down the protected line from the Remote relay 25 Ω of fault impedance.	79
Table 6-10: SEL-T401L Relay results for a L-L fault 60% down the protected line from the Remote relay 25 Ω of fault impedance.	79
Table 6-11: EMTP results for a 3PG fault 60% down the protected line from the Remote relay 25 Ω of fault impedance.	80
Table 6-12: SEL-T401L Relay results for a 3PG fault 60% down the protected line from the Remote relay.	80
Table 6-13: EMTP results for a L-G fault 60% down the protected line from the Remote relay with 50 Ω of fault impedance.	81

Table 6-14: Relay results for a L-G fault 60% down the protected line from the Remote relay with 50Ω of fault impedance.	81
Table 6-15: EMTP results for a L-L fault 60% down the protected line from the Remote relay with 50Ω of fault impedance.	82
Table 6-16: SEL-T401L Relay results for a L-L fault 60% down the protected line from the Remote relay with 50Ω of fault impedance.	83
Table 6-17: SEL-T401L Relay results for a 3PG fault 60% down the protected line from the Remote relay with 50Ω of fault impedance.	83
Table 6-18: EMTP results for a L-G fault 60% down the protected line from the Remote relay with 100Ω of fault impedance.	84
Table 6-19: SEL-T401L Relay results for a L-G fault 60% down the protected line from the Remote relay with 100Ω of fault impedance.	84
Table 6-20: SEL-T401L Relay results for a L-L fault 60% down the protected line from the Remote relay with 100Ω of fault impedance.	85
Table 6-21: SEL-T401L Relay results for a 3PG fault 60% down the protected line from the Remote relay with 100Ω of fault impedance.	85
Table 6-22: EMTP results for a L-G fault along the parallel path.	87
Table 6-23: SEL-T401L Relay results for a L-G faults along the parallel path.	88

ACKNOWLEDGMENTS

I would like to thank Dr. Prashanna Bhattarai for his guidance and instruction as well as for acting as the head of my advisory committee. I would also like to extend thanks to Dr. Mickey Cox and Dr. Matthew Hartmann for agreeing to serve in my thesis committee. I would also like to thank Mr. Jeremy Blair at SEL for his helpful insights throughout my work on this thesis. Additionally, I would also like to thank Mr. Nolan Matthews and Mr. Toby Russell for getting me up to speed on the larger digital relay model, and for their contributions which made the work presented here possible.

CHAPTER 1

INTRODUCTION

1.1 Faults and Transients

When a system is under normal operating conditions, the root mean square (RMS) values of the voltages and currents do not vary with time, and the system is said to be operating in a steady state. Certain events, such as the sudden introduction of a new load, a rapid increase in demand from an existing one, short-circuit events, etc., may cause a disturbance to this equilibrium. The state of the system following the disturbance is known as a transient state and will last until the system settles down at a different steady state (assuming it does not lose stability).

Short-circuit events within a power system can expose the system to currents that are several times greater than those under normal operating conditions. If left unchecked, these high currents may cause damage to equipment or loss of life. Events such as these are referred to as faults.

1.2 Protective Relaying

Protective relaying refers to the equipment and design methods used to detect abnormal power system conditions, such as power system faults, and initiate corrective action as quickly as possible. This process is typically quick, automatic, and should incur limited impact to the portions of the system that are not directly impacted by the

disturbance. Relays themselves vary in complexity and methodology; however, they all serve the function of observing a respective portion of the power grid and determining when a fault has occurred. This is accomplished through the measurement of voltages and/or currents at the relay location. How the relay utilizes these measured voltages and currents can vary from relay to relay, but they all serve the same purpose of determining abnormal operating conditions. Some methods of system protection are briefly described in the following sections to provide a general overview of the concept.

1.2.1 Overcurrent Protection

Overcurrent schemes rely upon the assumption that faults typically produce currents that are of much greater magnitudes than those seen during normal operating conditions. The particulars of how this type of protection may function vary: some are electro-mechanical in nature and rely upon magnetic forces to operate, whereas others may be entirely digital, consisting of no moving parts. In addition to overcurrent elements, time-delay overcurrent elements allow for the response to be proportional to the magnitude of the input current, that is, the larger the fault current the quicker the element will operate [1].

Whatever form this version of protection takes, they all fundamentally do the same thing, which is ensuring that current greater than a certain predetermined threshold gets interrupted as soon as possible. This form of protection is simple, however, and it generally does not readily allow for the creation of protection zones, which makes its utility for protecting transmission lines marginal.

Time-delay overcurrent relays are especially used to protect radial systems. Adjustable time delays can be selected such that the breaker closest to the fault opens

first, while other upstream breakers with larger time delays remain initially closed and provide backup protection [1]. However, coordination between overcurrent relays is generally limited to five relays for a radial system otherwise the relay closest to the source begins to have an excessive time delay, rendering it ineffective [1]. Additionally, for more complex systems, coordinating multiple overcurrent elements to protect the system becomes cumbersome and, in many cases, unpractical.

1.2.2 Differential Protection

Differential protection schemes function by comparing the currents at each end of a protected portion of a system. They are typically used to protect buses, generators, and transformers [1]. Under normal operating conditions, the current entering a protected section should be the same (or proportional in the case of a transformer) as the current leaving the section. However, when a fault occurs within the protected section, these two currents become unequal. The difference between the current entering and leaving the protected section of the system is known as the difference current. This difference current then flows through a relay operating coil which may cause the relay to operate.

This type of protection generally requires the terminals of the protected portion of the system to be relatively close to function; for transformers and buses, this is generally not a concern as they are contained within one geographic location. However, this becomes a problem when trying to apply this type of protection to transmission lines that may be many miles in length. This problem is addressed using line protection schemes discussed in the next section.

1.2.3 Phasor-Based Line Protection

The challenge of protecting transmission lines is unique because unlike other assets, such as transformers and generators, the terminals of a line may be many miles apart from one another. Additionally, coordinating overcurrent relays can also be difficult once there are a large number of lines and buses within the system. Configuring overcurrent elements in systems with multiple sources also presents a challenge.

To address these issues, relays that respond to both voltage and current are utilized. The voltage-to-current ratio is more sensitive to faults than the current alone [1]. Relays that operate on a ratio of voltage-to-current are called impedance, distance, or ratio relays.

Consider **Eq. 1-1**, where Z is the voltage-to-current ratio seen by the relay at the bus. The values may be measured using PTs and CTs at the relay location. Next, Z is compared to a value Z_r , which is an adjustable relay setting that defines the border of the tripping region, shown in **Eq. 1-2**.

$$Z = \frac{V_{bus}}{I_{bus}} \quad \text{Eq. 1-1}$$

$$|Z| < |Z_r| \quad (TRIP) \quad \text{Eq. 1-2}$$

$$|Z| > |Z_r| \quad (BLOCK)$$

The setting Z_r defines a point in the complex plane through which the perimeter of a circle centered at the origin must pass. This circle can be thought of as the tripping region of the relay. For instance, during normal operating conditions, the value Z is outside of this area (a value for Z_r is selected to ensure this), but during fault conditions, the voltage measured at the bus typically will decrease and the current will increase,

causing Z to significantly decrease. This concept is illustrated in **Figure 1-1**, where Z under normal operating conditions is shown, and the tripping region is shaded in.

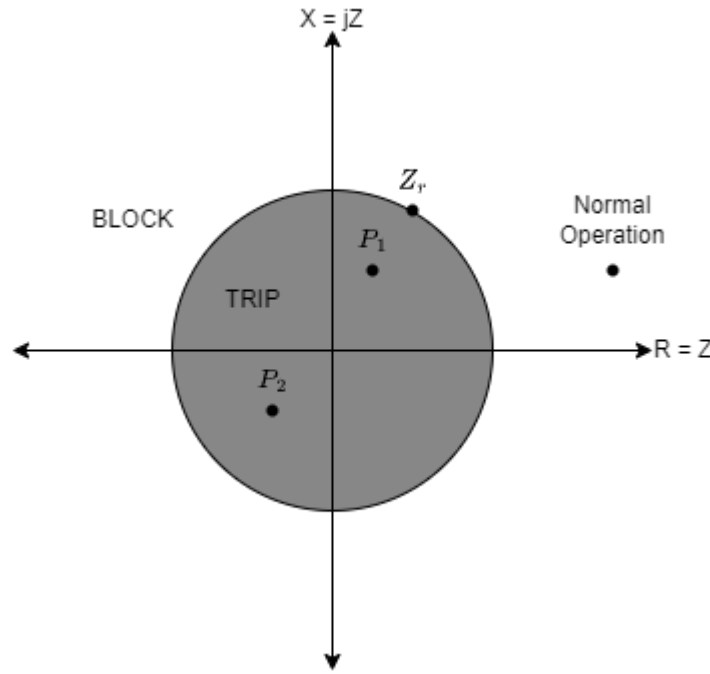


Figure 1-1: Impedance Relay Block and Trip Regions.

While impedance relays are useful for line protection and do not have the same inherent limitations of overcurrent and differential protection schemes, they do not have innate directional capabilities. For instance, a fault at P_1 in front of the relay and a fault at P_2 behind the relay will both cause the relay to operate.

To allow for directional capabilities, modified impedance relays (often called mho relays) are utilized. The key difference between an impedance relay and a mho relay is that a mho relay is directional [1]. A value of Z_r must be selected for mho relays as well, however, the difference is that the tripping region is no longer centered at the origin of the complex plane. Instead, the perimeter of the circle must now pass through both the origin and the selected point Z_r . This concept is illustrated in **Figure 1-2**, where the fault

in front of the relay (P1) still causes the relay to operate, but now the fault behind the relay (P2) is blocked.

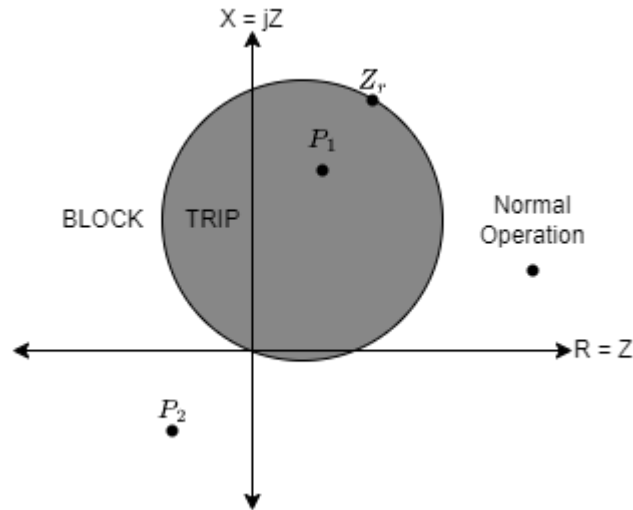


Figure 1-2: Modified Impedance Relay (mho relay) Block and Trip Regions.

Mho relays are typically used for line protection, with several relays being utilized per phase to create protection zones [1]. Implementing several relays per phase allows for the creation of backup protection for other zones. Values of Z_r are typically selected as 80%, 100%, and 120% of overall line impedance with a corresponding increase in time delay. This concept is illustrated in **Figure 1-3**, where the blocking regions of the respective zones corresponding to each selected Z_r are shown.

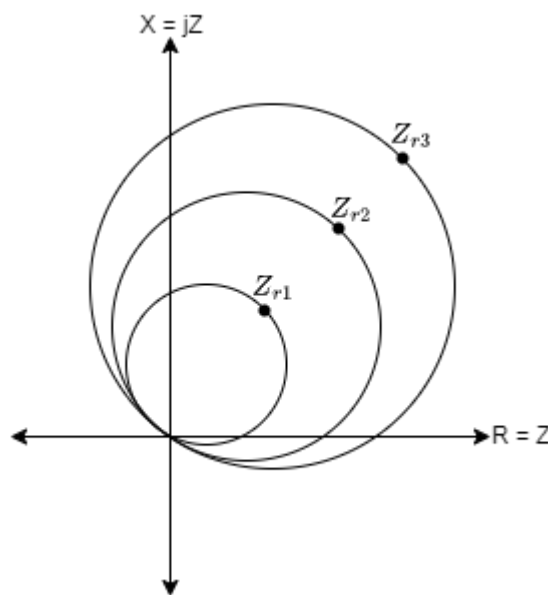


Figure 1-3: Implementing Several Relays Per Phase with Increasing Reach Points.

1.2.4 Disadvantages of Phasor-Based Protection and Alternatives

Traditional phasor-based line protection techniques have a minimum operating time of approximately one cycle (about 16.7 milliseconds), with best-case scenarios being about a half-cycle [2]. However, these fast operating times are achieved at the expense of reliability and security [3]. Newly developed time-domain-based relays (TDRs) are not limited by the inherent delays associated with phasor-domain-based protection systems. Instead of having to wait for one full cycle of data to determine abnormal conditions, time-domain elements can perform these calculations within a few milliseconds [4]. This is accomplished by analyzing the present and the one-cycle-old instantaneous values of currents and voltages.

One type of element available in TDRs are incremental-quantity-based elements. An incremental-quantity-based element utilizes values that represent the change due to a fault along with the magnitude of the line and system impedances to perform

calculations. The results of these calculations can be used to determine a variety of things depending on the purpose of the element within the relay, such as the distance of the fault from the relay or the direction of the fault with respect to the relay. The main focus of this thesis is the digital modeling and testing of an incremental quantity directional element using a transient analysis software package. The operating principles of this element are discussed in more detail in Chapter 2.

TDRs can detect faults much faster than phasor-based relays. One study has shown that for every one-cycle reduction in fault clearing time, up to 250 more megawatts of power can be transferred, or approximately 15 more mega-watts per millisecond of reduction for the system tested [2]. Decreasing fault-clearing times can also reduce damage to equipment and prevent potential loss of life. That being said, while time-domain elements have clear advantages in response time in comparison to traditional phasor-based elements, they typically still rely upon the dependability of traditional phasor-based elements to ensure system protection [5].

1.3 Software and Hardware Utilized in the Project

EMTP 4.2 and the software it is packaged with, Scopeview R2018b and Mplot, were utilized throughout this thesis for developing the system model as well as validating and testing. Note that Matlab Runtime version R2017b needs to be installed on the machine if Mplot is to be utilized. The EMTP protection toolbox was installed for the design and development process. Additionally, two SEL-T401L Ultra-High-Speed Line relays were utilized so that comparisons between the hardware and the digital model could be conducted. The relays were running on R100-V0 firmware. For interfacing with the relay, packaged SEL software was used such as AcSELERator (Ver. 7.0.0.7), SEL

Playback Filer Conversion Utility (Ver. 1.0.6.0), and SynchroWAVE Event Viewer (Ver. 1.8.1.36). Occasionally, Matlab (Ver. R2021b) and Mathcad Prime (Ver. 6.0.0.0) were utilized for calculations.

CHAPTER 2

INCREMENTAL QUANTITY DIRECTIONAL ELEMENT PRINCIPLES OF OPERATION

2.1 Operating Basics

An incremental quantity directional element (TD32) detects the direction in which a fault occurs relative to the position of the relay. It is used in a supervisory role for other elements such as the TD21 (incremental quantity distance element), the TD67 (incremental quantity directional overcurrent element), and in some instances, the TW87 (traveling wave differential protection element), within a TDR. The TD32 will assert a forward or reverse fault condition based on the relationship between the loop incremental replica currents (Δi_z) and incremental voltages (Δv), which are discussed in detail in the later sections of this chapter. The TD32 has a total of eight conditions it may assert during a fault scenario. These conditions correspond to forward and reverse faults for each phase, as well as a general forward and reverse condition that is asserted if any of the phases are faulted in the forward or reverse direction. The relay is coordinated with another relay placed at the remote end of the protected line, which is performing the same functions. This allows for the directional location of faults. For instance, if both the relays positioned at either end of the line assert a forward fault, then it can be inferred that a fault has occurred within the protected line.

2.1.1 Incremental and Incremental Replica Values

Relay elements that rely on the change between pre-fault and post-fault quantities are referred to as incremental quantity elements [3]. Under normal operating conditions, the incremental current and voltage are zero. However, these quantities will be non-zero during fault-induced transients because there will be a difference between the present and the one-cycle-old instantaneous values. The element then utilizes these values to determine an appropriate course of action. A simplistic comparison of currents in steady state and transient systems is depicted in **Figure 2-1**, where ΔI is the generated incremental quantity.

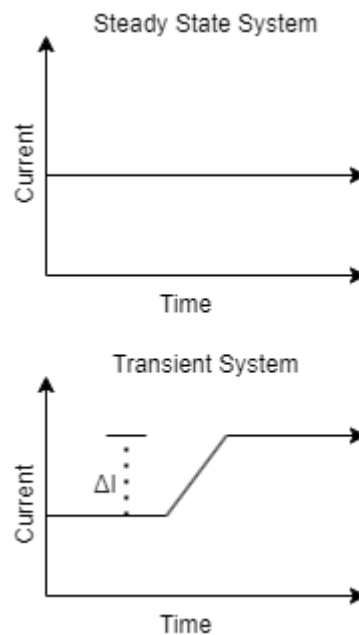


Figure 2-1: Comparison of Currents in Steady State (top) and Transient (bottom) System

As previously mentioned, fault-induced transients are typically what relays are monitoring for. A faulted network may be represented as the sum of the pre-fault network and the fault network using the superposition principle [6]. This concept is pictured in

Figure 2-2. The currents and voltages seen in the fault network of **Figure 2-2** are incremental quantities (i.e., they are the difference between the pre- and post-fault values). A delta (Δ) symbol is typically used to denote incremental quantities [6].

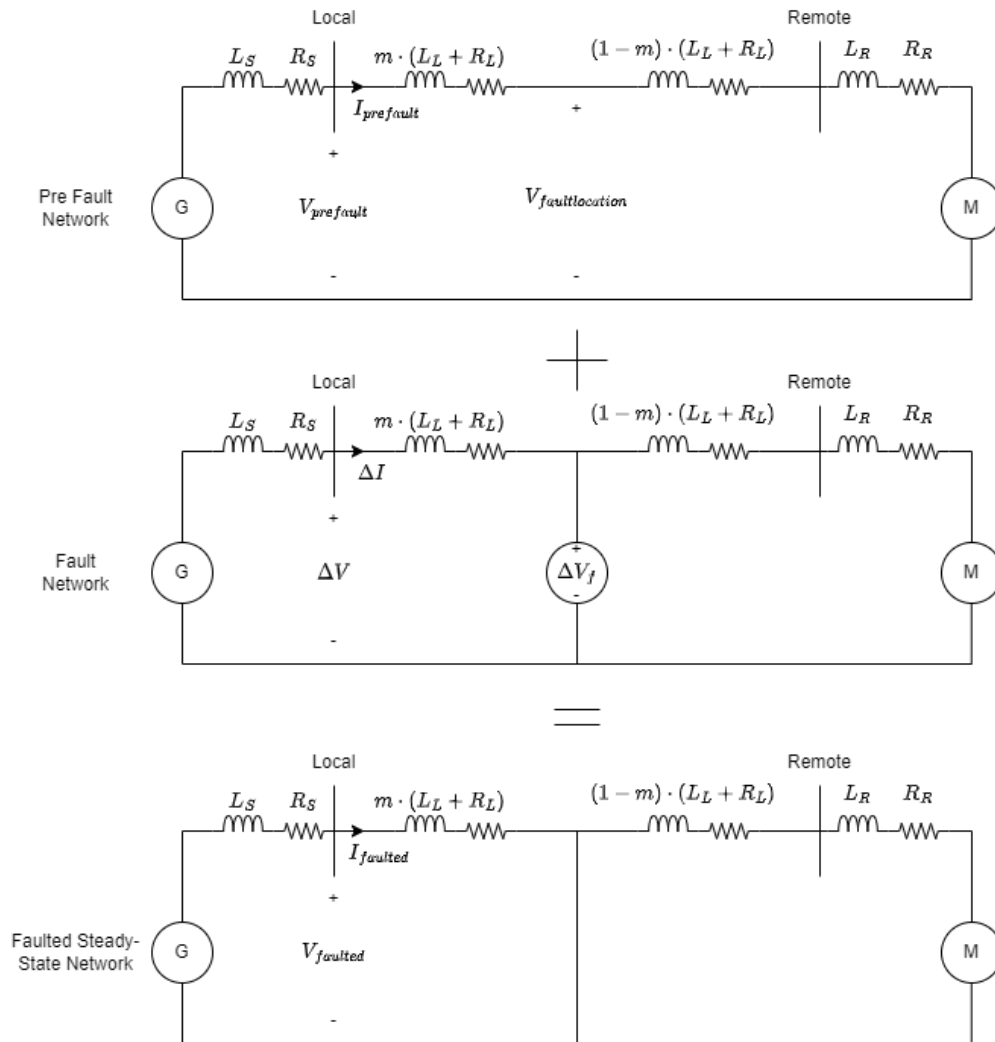


Figure 2-2: Application of the Superposition Principle to Derive the Fault-Generated Network Using the Pre-fault and the Faulted Networks

To avoid the differential equations that govern voltage and current in an RL network, incremental replica currents are introduced in the algorithms of the incremental-quantity-based elements. This effectively converts an RL network into a purely resistive

network. This is done so that the relationship between voltage and current becomes algebraic [6]. The derivation of an equation for incremental replica currents is outlined in the following section.

2.1.2 Extracting Incremental Values and Calculating Replica Values

This section briefly describes incremental values and incremental replica values and the methods of obtaining them as described in [6]. An incremental-quantity-based relay utilizes the incremental voltages, currents, and incremental replica currents (IRC) along with the magnitude of the line and system impedances to perform calculations. The results of these calculations can be used to determine a variety of things depending on the purpose of the element within the relay, such as the distance of the fault from the relay in the case of the incremental quantity distance element (TD21) or direction of the fault with respect to the relay in the case of the TD32. It is expected that these signals are filtered by a lowpass filter in real-world systems (with cut-off frequencies being dependent on the element in question) before the incremental values are extracted. However, such a digital filter, as discussed in Chapter 3, was found to have a negligible impact on the performance of the digital model, and was excluded from any extensive design process.

To begin performing the necessary calculations, the incremental voltage and replica current values must be extracted from the protected system. This is accomplished through the use of a delta filter, seen in **Figure 2-3**, which calculates the difference between the present instantaneous value and the time-delayed instantaneous value [5]. The time delay is equal to one power system cycle or approximately 16.667 ms for an operating frequency of 60 Hz.

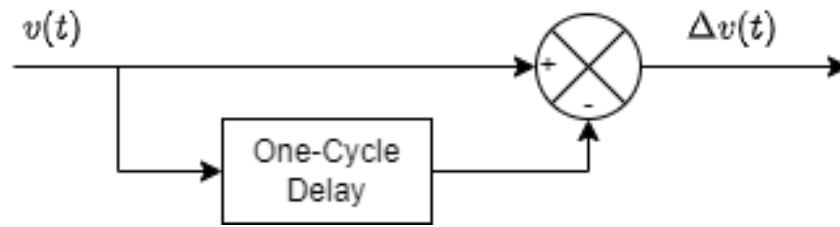


Figure 2-3: Delta Filter.

For incremental values, the extraction process is completed after the instantaneous input has passed through the delta filter pictured in **Figure 2-3** because they are simply the difference between the present and the one-cycle-old instantaneous values. However, the calculation of the IRC values is more involved as they are intended to replicate a voltage drop and are not merely a simple difference of values. The derivation of the equation used to calculate IRC values may be extracted from a faulted RL network as discussed below.

Consider the fault network pictured in **Figure 2-4**, where m is the fault location in terms of line percentage, R_L , and L_L represent the line resistance and inductance values respectively, and R_S , L_S , R_R , and L_R represent the source and remote system resistances and inductances respectively. The fault network is a representation of the change in the steady state current and voltage that arises due to a fault condition. Under normal operating conditions, the incremental values would be zero because there would be no difference between the one-cycle old value and the present instantaneous value.

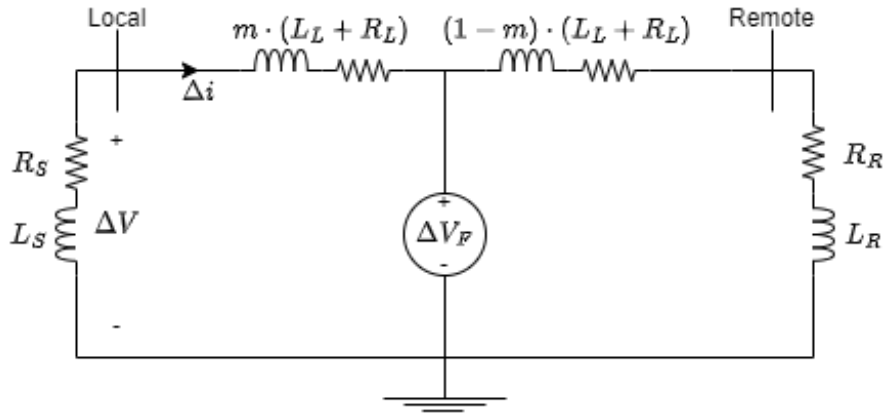


Figure 2-4: RL Fault Network

At the relay location, the incremental voltage and current are related by **Eq. 2-1**.

This equation may be scaled by a factor of one in the form of $\frac{|Z_S|}{|Z_S|}$. Doing so will produce

Eq. 2-2. The equation for incremental replica current may now readily be derived by dividing both sides of **Eq. 2-2** by $-|Z_S|$ to produce **Eq. 2-3**. An algebraic relationship may now be established between the incremental voltage and the incremental replica current, as seen in **Eq. 2-4**.

$$\Delta v = -(R_S \Delta i + L_S \frac{d}{dt}(\Delta i)) \quad \text{Eq. 2-1}$$

$$\Delta v = -|Z_S| \left(\frac{R_S}{|Z_S|} \Delta i + \frac{L_S}{|Z_S|} \frac{d}{dt}(\Delta i) \right) \quad \text{Eq. 2-2}$$

$$-\frac{\Delta v}{|Z_S|} = -\Delta i_Z = \frac{R_S}{|Z_S|} \Delta i + \frac{L_S}{|Z_S|} \frac{d}{dt}(\Delta i) \quad \text{Eq. 2-3}$$

$$-|Z_S| \cdot \Delta i_Z = R_S \Delta i + L_S \frac{d}{dt}(\Delta i) \quad \text{Eq. 2-4}$$

The derivation of **Eq. 2-4** allows for the establishment of the linear relationship between incremental voltage (Δv) and incremental replica current (Δi_Z) seen in **Eq. 2-5**. Again, this linear relationship is utilized to avoid the differential equations that arise from

RL circuits and is used to replicate the voltage drop associated with the circuit network shown in **Figure 2-4**.

$$\Delta v = -|Z_S| \cdot \Delta i_Z \quad \text{Eq. 2-5}$$

2.1.3 Calculating Incremental Replica Current Values for Ground Loops

Since IRC values are specific to each loop of the protected system, they must be calculated for all the phase-to-ground and phase-to-phase loops. This is accomplished by utilizing **Eq. 2-3**. However, before **Eq. 2-3** may be utilized, the relevant incremental currents and voltages must first be calculated by utilizing the previously discussed delta filter.

The process of calculating incremental current is displayed for the zero-sequence current in **Figure 2-5**, however, the process of calculating incremental current values is similar for the remaining three phase currents and is the same fundamental process as extracting incremental voltages (via the use of a delta filter). The zero-sequence current may be obtained by utilizing the standard equation, seen in **Eq. 2-6**.

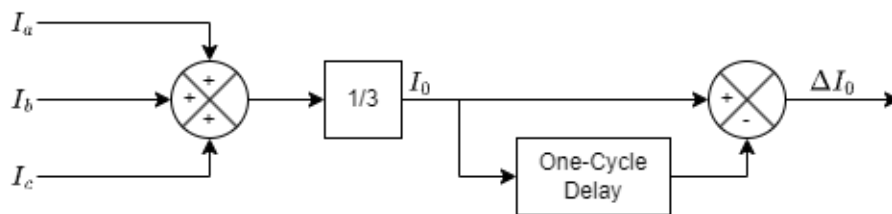


Figure 2-5: Zero Sequence Incremental Current.

$$I_0 = \frac{1}{3}(I_a + I_b + I_c) \quad \text{Eq. 2-6}$$

Once the three phase incremental currents (Δi_a , Δi_b and, Δi_c) have been extracted, the IRCs may be calculated utilizing **Eq. 2-3**. This process is shown in **Figure 2-6** with the input being the respective incremental current value.

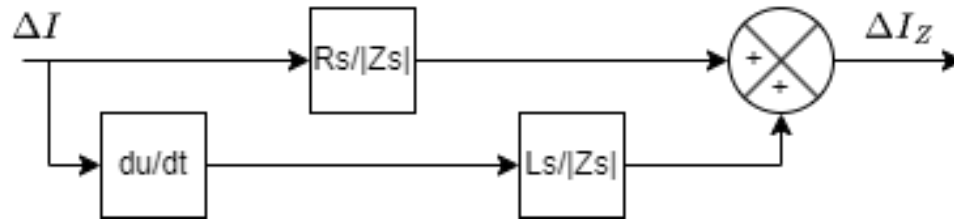


Figure 2-6: Incremental Replica Current Filter.

The output of **Figure 2-6** will be the respective IRC value (ΔI_{Za} , ΔI_{Zb} , ΔI_{Zc} , and ΔI_{Z0}). Once all of the IRC values have been calculated, the loop quantities utilized in the TD32 logic may then be obtained. This is accomplished by summing the IRC values of the phase and the zero-sequence IRC value for ground loops (i.e., A-G, B-G, C-G) as shown in **Figure 2-7** for the A-G loop. The ground loop values are ΔI_{ZAG} , ΔI_{ZBG} , and ΔI_{ZCG} . For phase loops, the difference of the incremental current values is utilized as the input of **Figure 2-6** and directly produces the phase-to-phase loop value. For instance, the difference of Δi_a and Δi_b when utilized as the input of **Figure 2-6** will produce the phase loop value ΔI_{ZAB} .

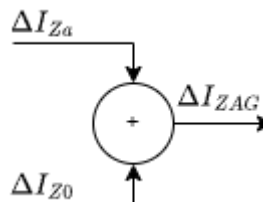


Figure 2-7: Current Loop Quantity Calculation for A-G Loop.

2.2 TD32 Circuit Representation

Figure 2-8 is the positive-sequence circuit representation of the basic operating principle of the TD32 element. Depending upon the location of the fault with respect to the relay, different impedance magnitudes and fault current polarities will be observed at the relay location. The quantities Z_{Is} , Z_{IL} , and Z_{IR} represent the positive-sequence impedances of the system source, line, and remote system (i.e., the system beyond the protected line) respectively. The TD32 utilizes the previously discussed incremental replica currents (Δi_z) along with the magnitude of the respective impedances to calculate the torques using equations **Eq. 2-7**, **Eq. 2-8**, and **Eq. 2-9**. The TD32 uses these calculated torque quantities to determine the relative direction of the fault. The torque is calculated as the product of the sign inverted incremental voltage and the IRC, meaning that the forward restraining torque will always be a positive quantity and the reverse restraining torque will always be a negative quantity [7]. The sign of the operating torque will vary depending upon the location of the fault relative to the relay due to current polarity.

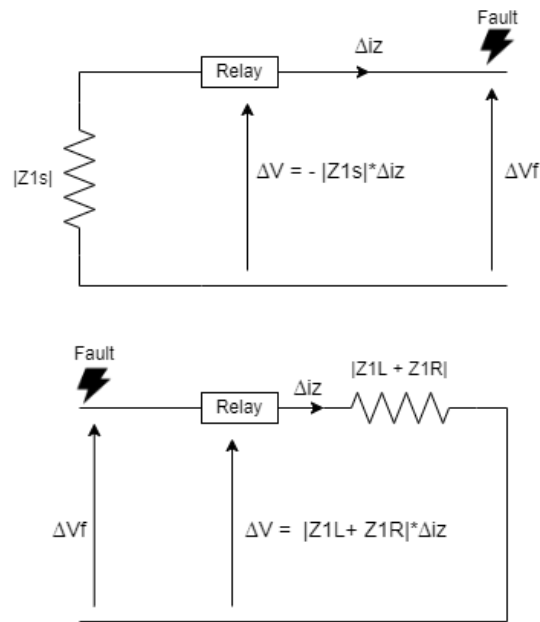


Figure 2-8: TD32 Forward and Reverse Fault Conditions Positive Sequence

$$|Z1s| * \Delta i_z^2 = \text{Forward Restraining Torque} \quad \text{Eq. 2-7}$$

$$-\Delta v * \Delta i_z = \text{Operating Torque} \quad \text{Eq. 2-8}$$

$$-|Z1L + Z1R| * \Delta i_z^2 = \text{Reverse Restraining Torque} \quad \text{Eq. 2-9}$$

Under normal operating conditions, no incremental quantities will be generated; thus, the IRC values will be zero as well. However, under transient conditions, the incremental voltage and IRC values will become non-zero. For a forward fault, we expect the operating torque to be greater than the forward restraining torque, and for a reverse fault, we expect the operating torque to be less than the reverse restraining torque. The calculation and comparison of torques are implemented in the element using the logic seen in **Figure 2-9**, shown for the A-G loop. Note, that these calculations and comparisons are made on a per-loop basis, that is, there is a set of logic for each of the L-

G and L-L loops. The input of the logic are the respective previously discussed loop quantities (ΔI_{ZAG} , ΔI_{ZAB} , and so on).

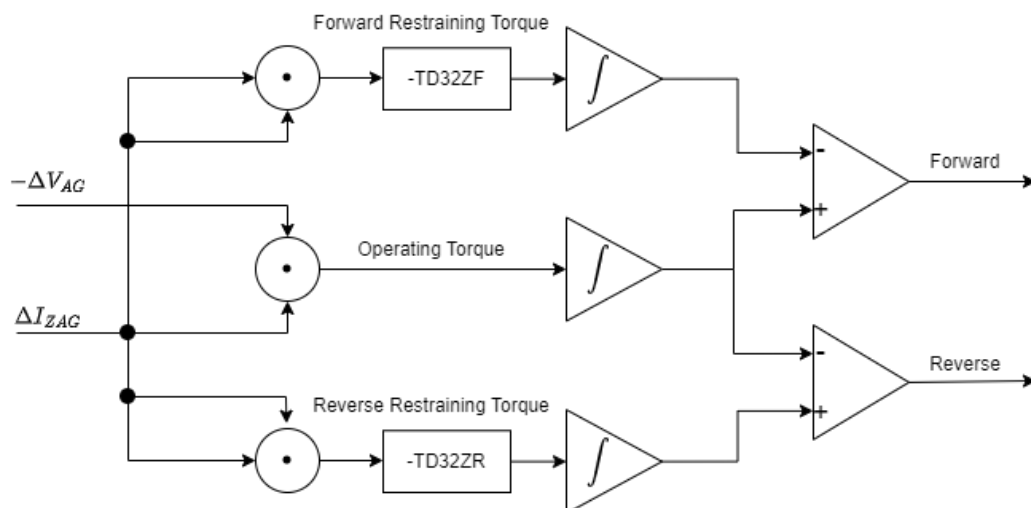


Figure 2-9: TD32 Operating Logic Diagram for A-G Loop

The TD32 has two configurable settings, namely TD32ZF and TD32ZR, which are the forward and reverse restraining impedances of the element, respectively, they constitute a percentage of the line and source impedances (Z_{ls} and Z_{lL}) and are typically set to 0.3Ω -secondary [7]. The TD32ZF setting is negative for consistency with other phasor-based directional element settings within the relay [7]. However, the logic utilizes the negative sign of both the TD32ZF and TD32ZR setting, which is what ultimately causes the forward restraining torque to be positive even though the TD32ZF setting itself is negative. The TD32ZR setting is positive, meaning that after the negative sign of it is taken by the logic, the reverse restraining torque will become negative. Recall that the forward restraining torque will always be a positive quantity and the reverse restraining torque will always be a negative quantity. All of the operating and restraining

torques are integrated to smooth them out and compared to one another to determine the direction of a fault. All of this is accomplished utilizing the logic shown in **Figure 2-9**.

2.2.1 Determination of Fault Direction

The TD32 operating logic utilizes the incremental replica current (Δi_z) and the incremental voltage (Δv) values produced by the incremental quantity filter along with the forward and reverse positive-sequence impedances to calculate the three torque values, pictured in **Figure 2-10**. The forward and reverse positive sequence impedances are configurable settings that must be provided and are respective to the system in which the TD32 is being implemented. The two values TD32ZF and TD32ZR are the TD32 forward and the reverse impedance thresholds, respectively. These two values are adjustable, with the default value being 0.3 Ω -secondary, which is adequate for most systems [7]. The operating torque is compared to the forward and reverse restraining torques via two comparators to determine the direction of the fault. This process is repeated for each ground and phase loop of the protected line. **Figure 2-10** shows the operating and two restraining torques after integration. Since an A-G fault has been introduced on the protected line, it is expected that the operating torque will be greater than the forward restraining torque. Similar waveforms can be seen on the corresponding relay on the other end of the protected line.

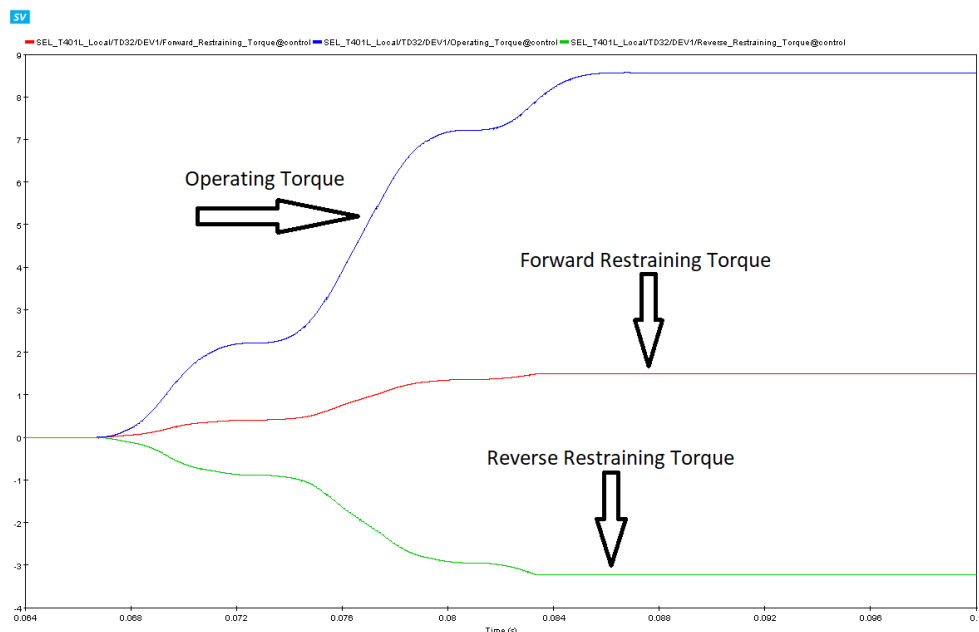


Figure 2-10: TD32 Torque Comparison For Forward Fault Conditions

Figure 2-11 shows the three torque values again for a fault introduced behind the protected line. In this scenario, it is expected that the operating torque will be less than the reverse restraining torque. However, the corresponding remote relay should see waveforms similar to those pictured in **Figure 2-10**, implying a forward fault condition from the “view” of the remote relay. As such, the local relay would assert a reverse fault and the remote relay would assert a forward fault, indicating that the fault is outside of the protected system. Note, that before such an assertion could be made, the two relays would need to communicate with one another, which would result in an operation delay.

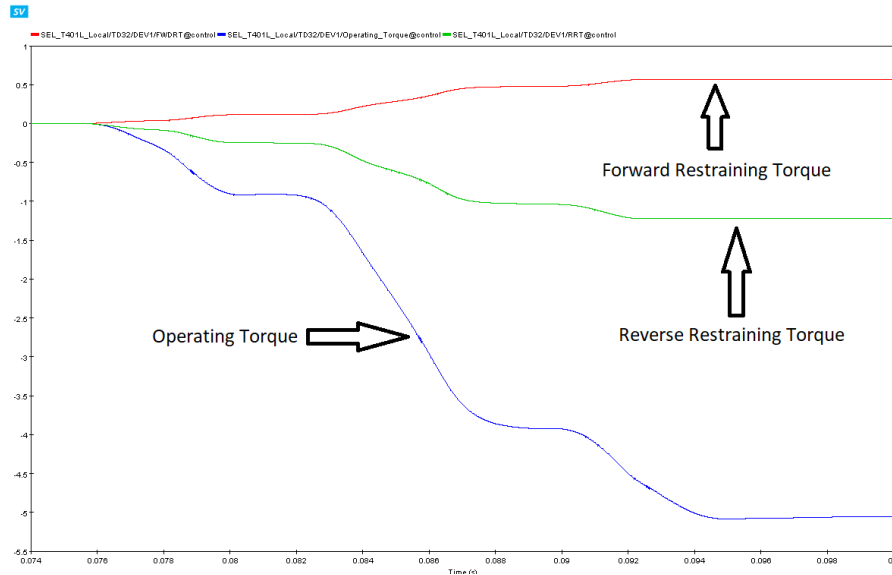


Figure 2-11: TD32 Torque Comparison For Reverse Fault Condition.

In conclusion, an operating torque is calculated for comparison with the forward and reverse restraining torques. In theory, just the polarity of the operating torque may be observed to determine the direction of a fault; however, utilizing this method would degrade the security of the element by making it overly sensitive [2]. This is why the additional step of calculating forward and reverse restraining torques is necessary.

2.2.2 Additional Element Security

To ensure the security (i.e., to ensure that the element does not operate when it should not) of the TD32, the incremental torque is applied to the outputs of the operating, forward restraining, and reverse restraining torques. This process is shown in **Figure 2-12**. The incremental torque is only applied for a fraction of a millisecond when initial transients are occurring [5]. The exact duration that the incremental torque is applied was not provided in literature and had to be determined experimentally through trial and error. This process provides an initial boost to the operating torque and one of the restraining torques following disturbance detection for added element security [5].

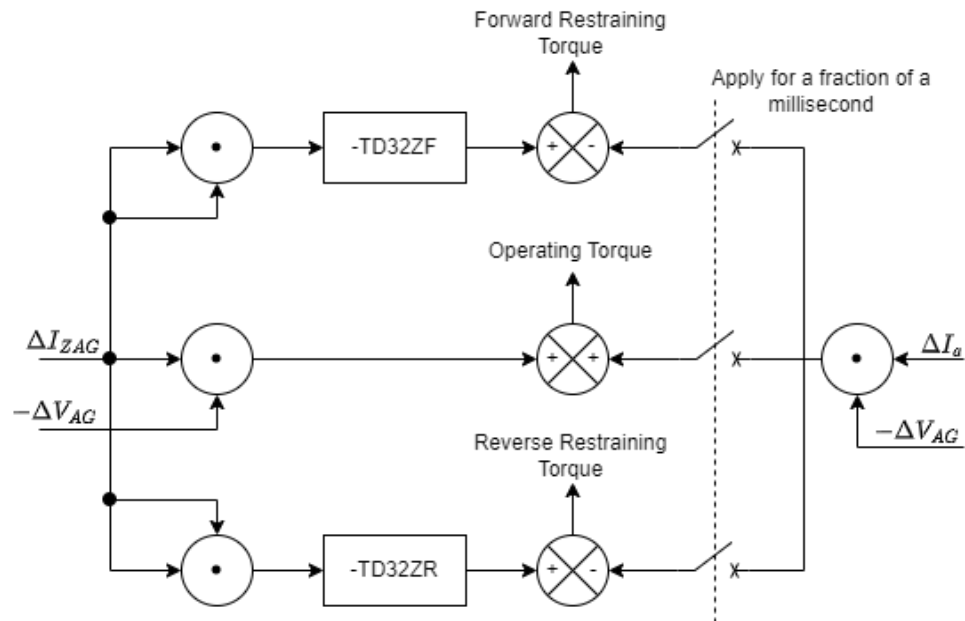


Figure 2-12: TD32 Torque Calculations

Additionally, a security margin is added to the outputs of the forward and reverse restraining torque integrators to offset them from zero. This process can be seen in **Figure 2-13**. This is done to ensure that any minuscule transients that may occur when the element is initiated do not cause the TD32 to falsely operate. The exact value of the security margin was not provided in literature and had to be determined experimentally through trial and error.

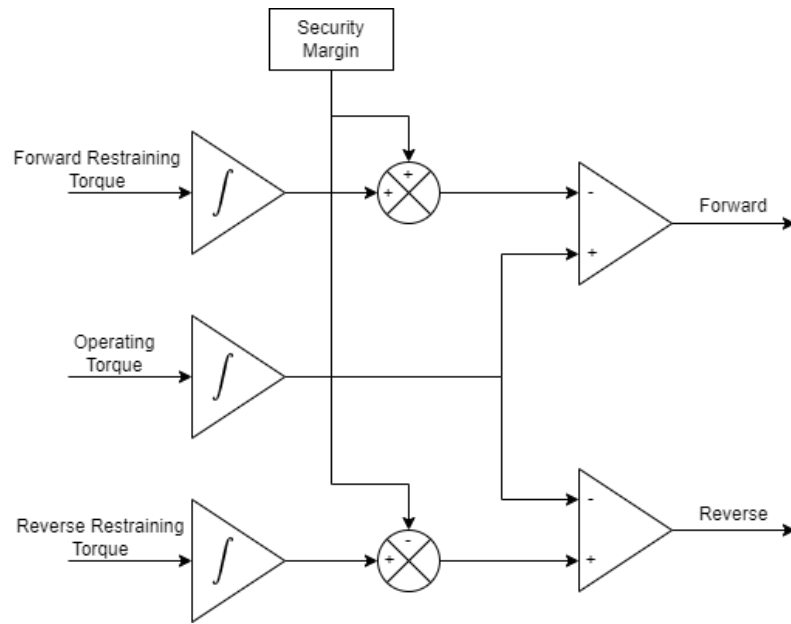


Figure 2-13: TD32 Security Margin

CHAPTER 3

DIGITAL MODELING OF AN INCREMENTAL QUANTITY DIRECTIONAL ELEMENT

3.1 The Test System

A simple test power system was utilized during the development of the digital model of the TD32 element to avoid any of the nuances that may arise from using a more complex system. The test system is shown below in **Figure 3-1** and consists of two synchronous machines and three line segments. The length of the protected line (Protected_Line) is 175km and the length of the two external line segments (ExternalLine_1 and External_Line_2) are 150km each. All lines are constant parameter lines with the characteristics shown in **Figure 3-2**. The percent impedance of the two machines is shown in **Figure 3-3**.

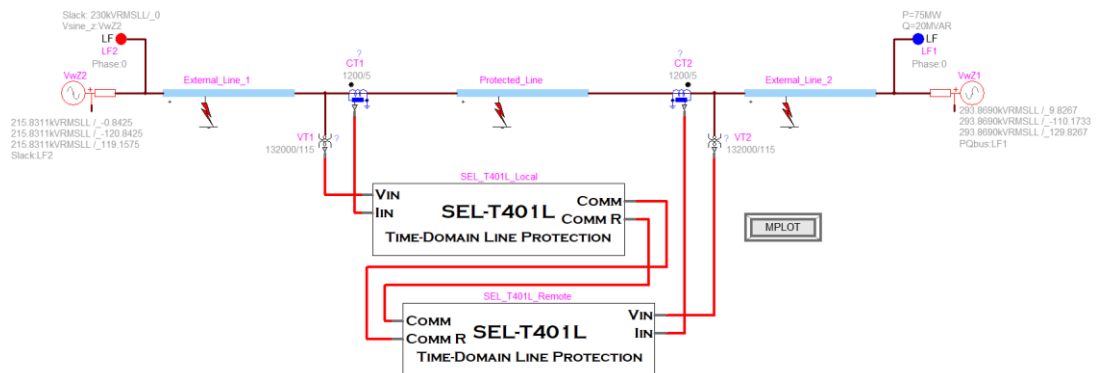


Figure 3-1: EMTP Simple System Used For Development.

Constant Parameter (CP) line/cable model (multiphase)

Load data from file (option)

Number of phases

Length

Select model

Distortionless Continuously transposed

Select type of data

R', L', C' R', Z₀, v (speed) R', Z₀, τ (delay)

Select Units

R' L' C'

Propagation mode data

Mode	R'	L'	C'
1	.3	3	.008
2	.02	.9	.0126

Figure 3-2: Parameters for CP Lines in EMTP Development System.

Source Impedance

Short-circuit data input Impedance data input

R' Impedance real part Sequence Data Input

	Zero	Positive	Negative
R	15	5	5

L' Impedance imaginary part Sequence Data Input

	Zero	Positive	Negative
L	100	20	20

Impedance data is automatically updated when OK is clicked or when moving to another tab.

Figure 3-3: Source and Motor Impedances for EMTP Development System.

The lines were modeled using the block seen in **Figure 3-4** where the line lengths on the right and left sides may be adjusted to move the fault location. The line segments on either end of the fault (DEV1) are using the constant parameters seen in **Figure 3-2**.

To introduce faults into the system, the fault block seen in **Figure 3-5** was used. It introduces single phase-to-ground, line-to-line, and three phase-to-ground faults into the system using the labeled “LG fault” and “LL fault” switches.

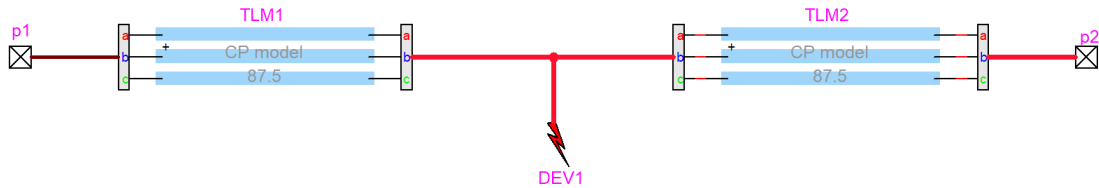


Figure 3-4: EMTP Line Block Model.

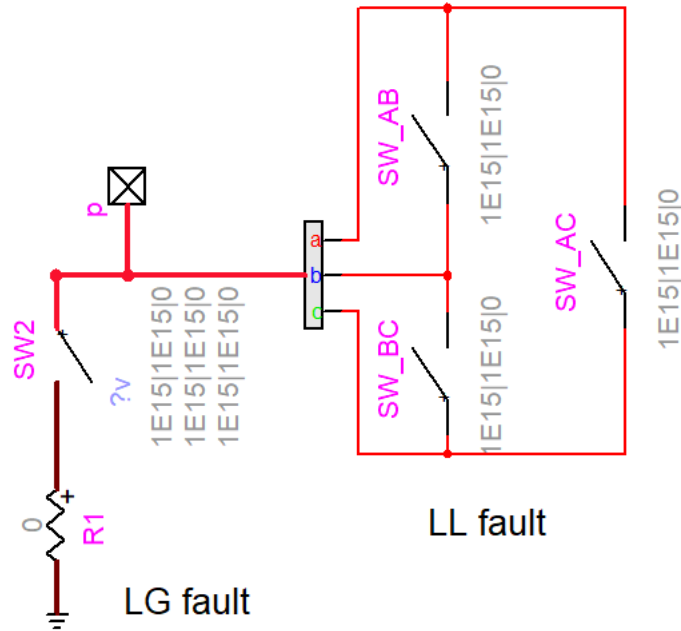


Figure 3-5: EMTP Fault Block Model.

3.2 Digital Modeling of the Element

Figure 3-6 is the top layer of the EMTP model that has been implemented and it shows the inputs and outputs of the model. **Figure 3-7** represents the first layer of the EMTP model. It consists of the incremental quantity filter, ground and phase loop logic

blocks, and the “Fault Logic Block.” All of these blocks are discussed in detail in the following text.

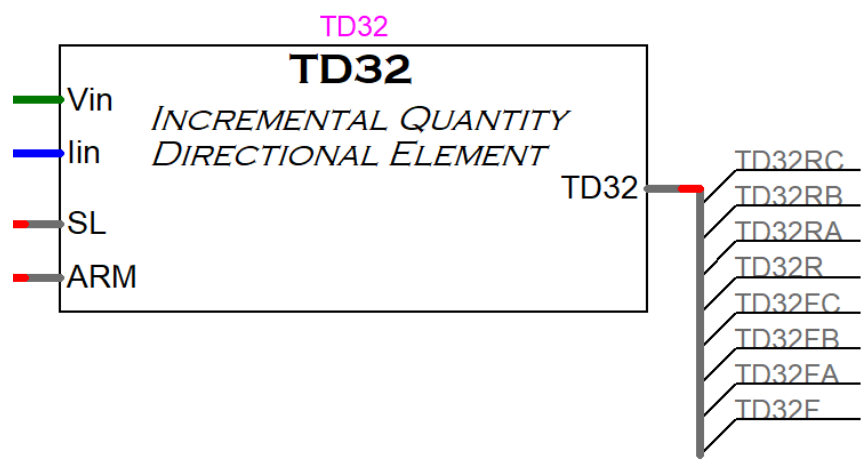


Figure 3-6: Top Layer of the TD32 EMTP Model

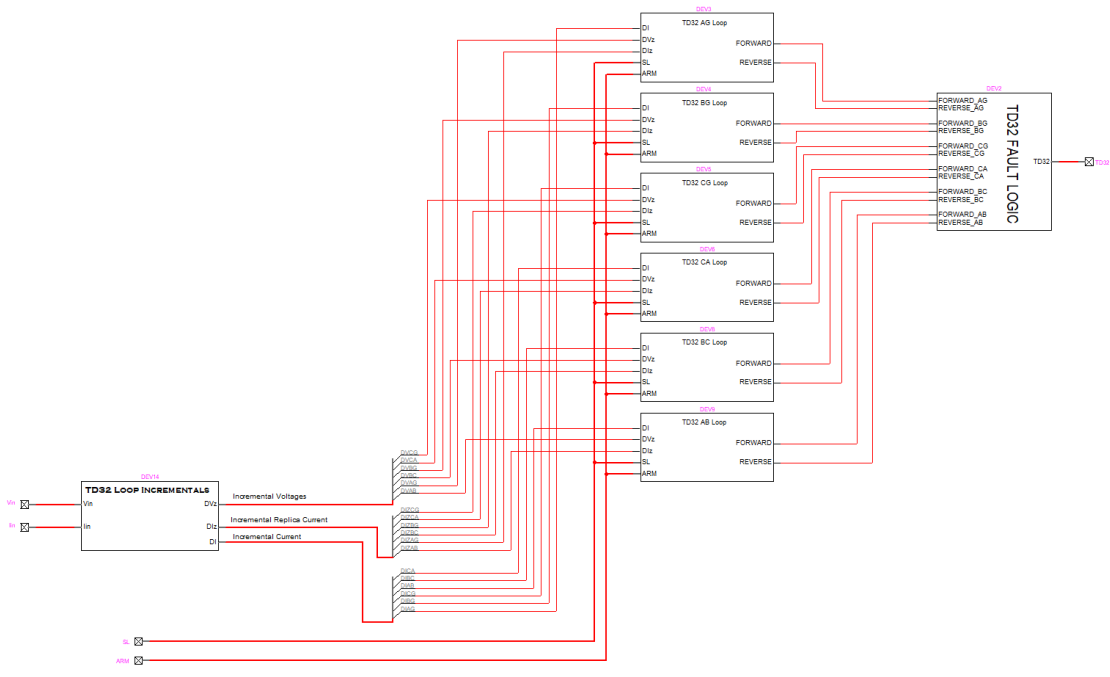


Figure 3-7: Overview of EMTP TD32 Model

3.2.1 EMTP Incremental Quantity Filter

The components of the incremental quantity filter are shown in **Figure 3-8** through **Figure 3-11**. **Figure 3-8** is an overview of the filter showing its inputs and outputs. The blocks shown in **Figure 3-9** are the individual components of the filter responsible for extracting incremental values and calculating incremental replica currents. **Figure 3-10** is the implementation of a simple incremental voltage filter. It is the difference between the present instantaneous voltage value and the one-cycle old instantaneous value. **Figure 3-11** represents the implementation of an incremental replica current filter within EMTP where the values of R_s , L_s , and Z_s have been provided as variables in the SEL-T401L mask within EMTP. The incremental current and incremental replica current values have both been extracted from the incremental current block for use in the loop logic blocks.

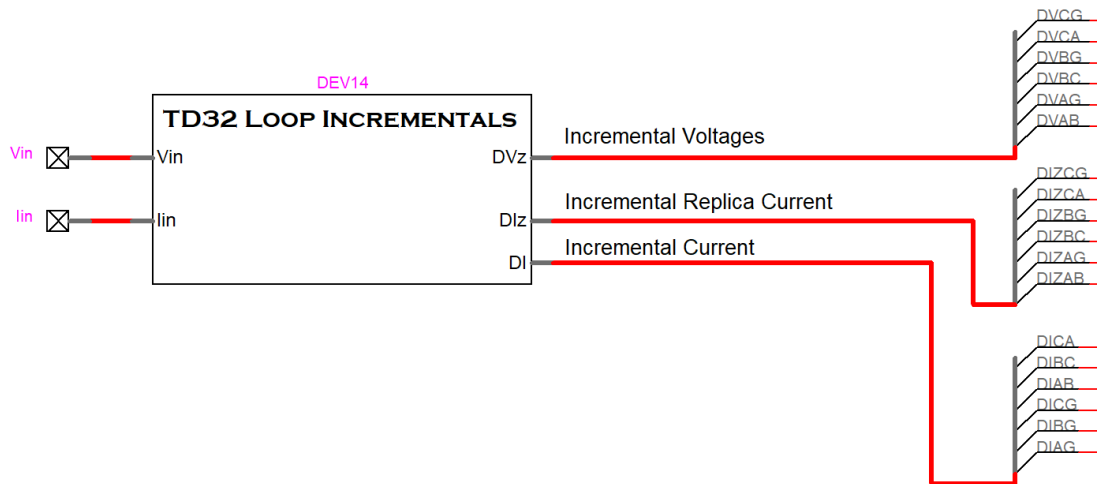


Figure 3-8: Top Layer of EMTP TD32 Loop Incremental Filter

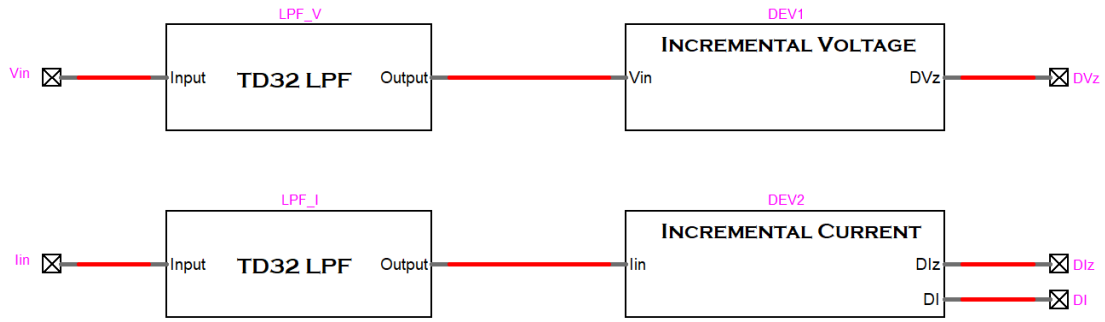


Figure 3-9: Overview of EMTP Incremental Quantity Filter

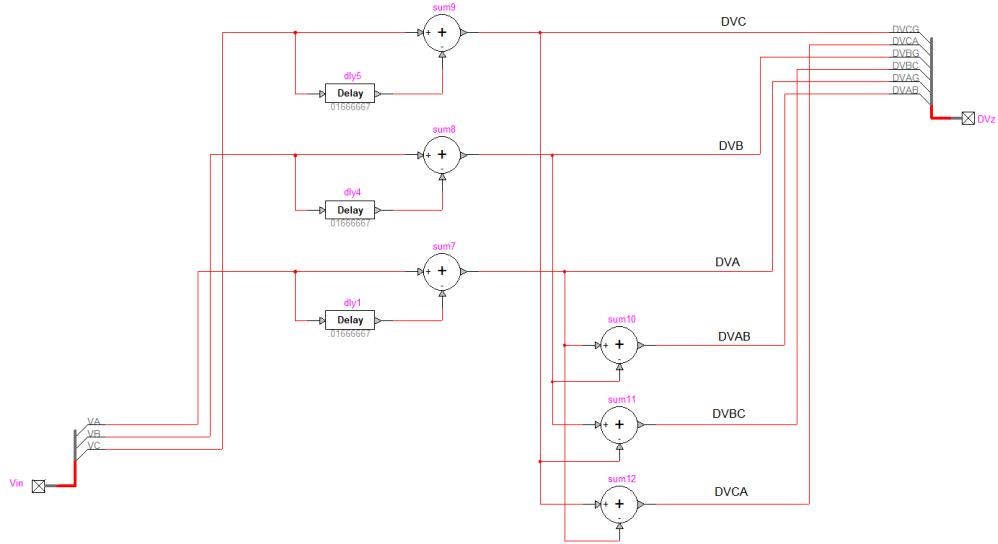


Figure 3-10: EMTP Incremental Voltage Filter

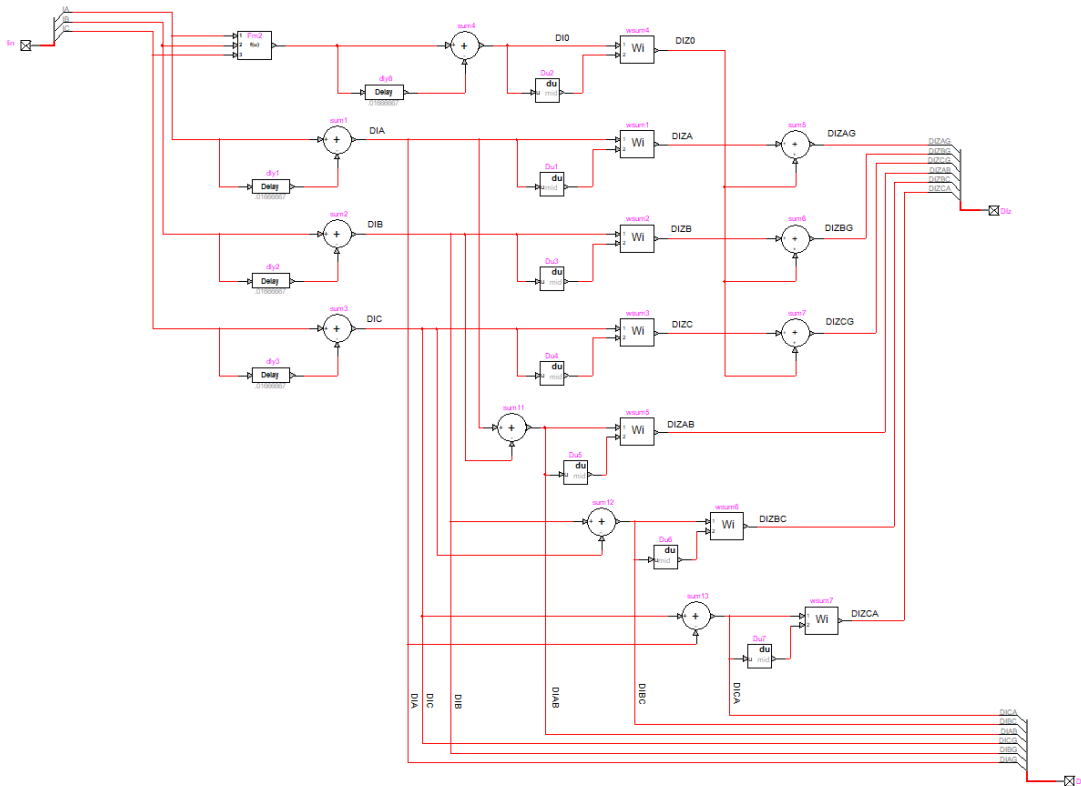


Figure 3-11: EMTP Incremental Replica Current Filter

EMTP weighted sum blocks are used in the incremental replica current block pictured in **Figure 3-11** which allows **Eq. 2-3** to be implemented in the EMTP model. **Figure 3-12** shows how the weighted sum blocks are configured, and **Figure 3-13** shows the calculations performed within EMTP to calculate the needed factors for **Eq. 2-3**, that is, the values of $\frac{R_S}{|Z_S|}$ and $\frac{L_S}{|Z_S|}$. EMTP has built-in derivative and delay blocks that are utilized where necessary in the incremental replica current filter.

Define input coefficients (w)

$$\text{output} = \text{gain} * (w_1 * \text{input}_1 + w_2 * \text{input}_2 + w_3 * \text{input}_3 + \dots)$$

Input coefficients (w_i)

#b1# #b2#

Gain

1

Figure 3-12: EMTP Weighted Sum Inputs.

```

b1 = cos(Z1ANG*(pi/180))
b2 = sin(Z1ANG*(pi/180))/(120*pi)
b3 = (((Z0MAG*cos(Z0ANG*(pi/180)))-(Z1MAG*cos(Z1ANG*(pi/180))))/Z1MAG)
b4 = (((Z0MAG*sin(Z0ANG*(pi/180)))-(Z1MAG*sin(Z1ANG*(pi/180))))/(Z1MAG*120*pi))
TWLPTr = 0.00001*TWLPT
LL = Line_Length*1.609345909

```

Figure 3-13: EMTP Incremental Replica Current Inputs in the T401L Mask.

3.2.2 Ground and Phase Loops

After the incremental quantity filter was developed, the six loop logics were then implemented into the model. The logic for the A-G loop can be seen in **Figure 3-14**. At this point, it is important to note that there are some differences here between what is shown in the SEL-T401L Ultra-High-Speed Line Relay (the TDR used in this study) manual and what has been implemented in EMTP.

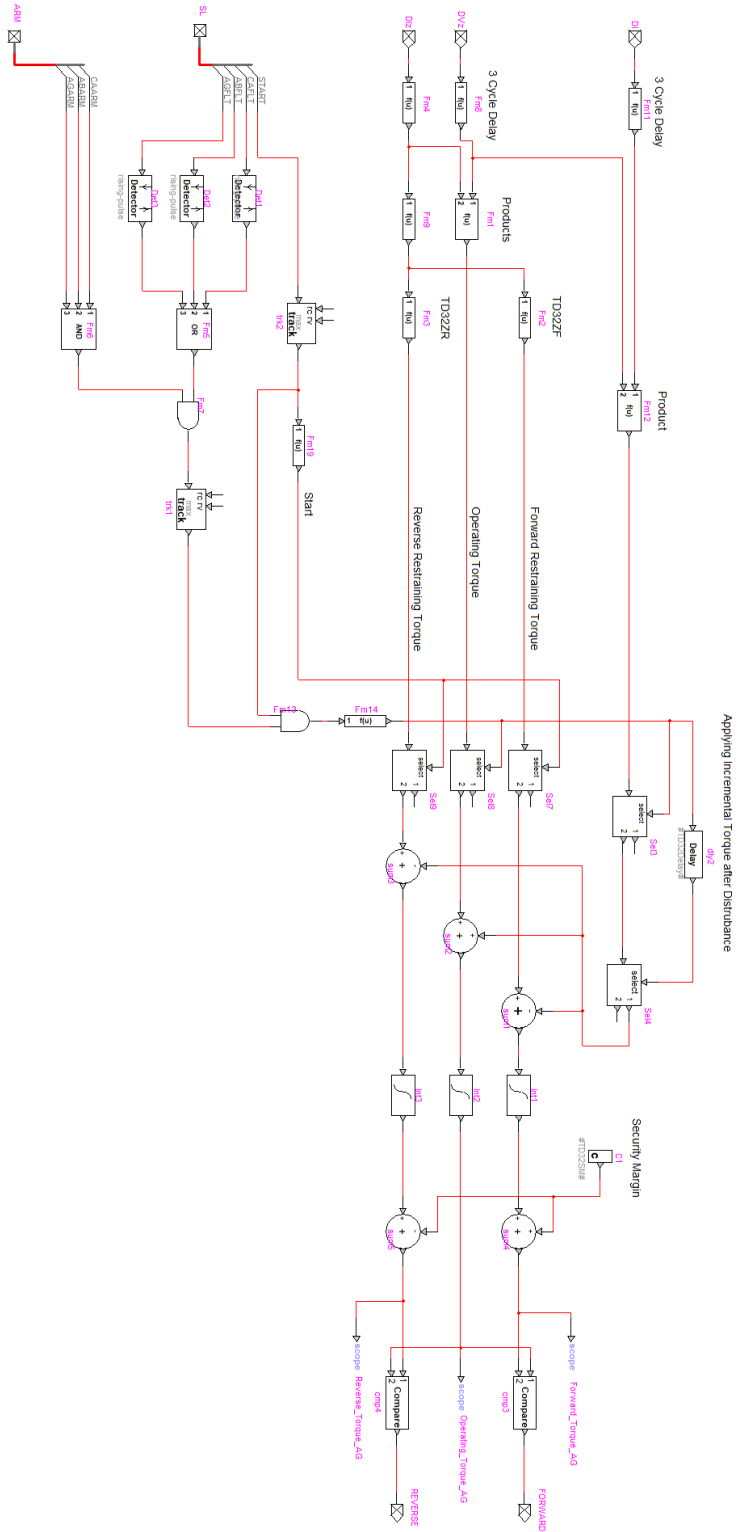


Figure 3-14: EMTP TD32 Loop Logic.

First, an AND gate has been included before the signal that allows the operating torque to be passed to the integrators. The inputs of this AND gate are the start signal, and a combination of the relevant fault and arm signals (from the Starting Logic block modeled in [8]). This was done because there were some instances where the operating torque would begin running erroneously. When this occurred, the element would, in some rare cases, falsely assert that a fault had occurred. Including this AND gate has not impacted the overall function of the loop logic in any way other than solving this issue.

Second, two additional pieces of logic have been included that were not mentioned in the SEL-T401L relay instruction manual, but were observed in other publications [5] related to the TD32 element. The incremental torque, which is defined as the product of the inverse sign incremental voltage and the incremental current (as opposed to the incremental replica current in the case of the operating and restraining torques), is applied for a fraction of a millisecond. This is done to provide a boost to the operating and restraining torques when the TD32 is initiated [5]. Additionally, a security margin was added to the output of the integration of the restraining torques. This was done to offset the forward and reverse restraining torques from zero, which in some cases would cause erroneous fault assertions during minor transients [5].

The loop logic for the other phases (i.e., B-G, C-G, A-B, etc.) is modeled similarly with their respective loop quantities and starting signals as inputs. The logic diagrams for the other ground and phase loops were not provided in the SEL-T401L relay instruction manual, so no indication was given that there are differences in logic between the loops.

During the implementation of the TD32 element using the simple test system, the forward and reverse restraining impedances (TD32ZF and TD32ZR) were both set to 0.3 Ω -secondary, as recommended in the SEL-T401L instruction manual [7]. The security margin was set to 0.2 and the incremental torque was applied for .75 ms. These settings were determined experimentally and may need to be readjusted on a system-by-system basis.

3.2.3 Fault Logic

The outputs of the loop logic blocks are passed to a “Fault Logic Block,” which is not shown in the SEL-T401L manual; however, it is implemented in the EMTP model as shown in **Figure 3-15** due to a lack of information on this portion of the relay. It is unlikely that this block will be the source of any deviation from the SEL-T401L relay performance due to its simplicity. The fault logic block, shown in **Figure 3-15**, was implemented to interpret the outputs of the previously discussed loop logic blocks. Based on the outputs of the loop logic blocks, the fault logic block will determine which word bits need to be asserted.

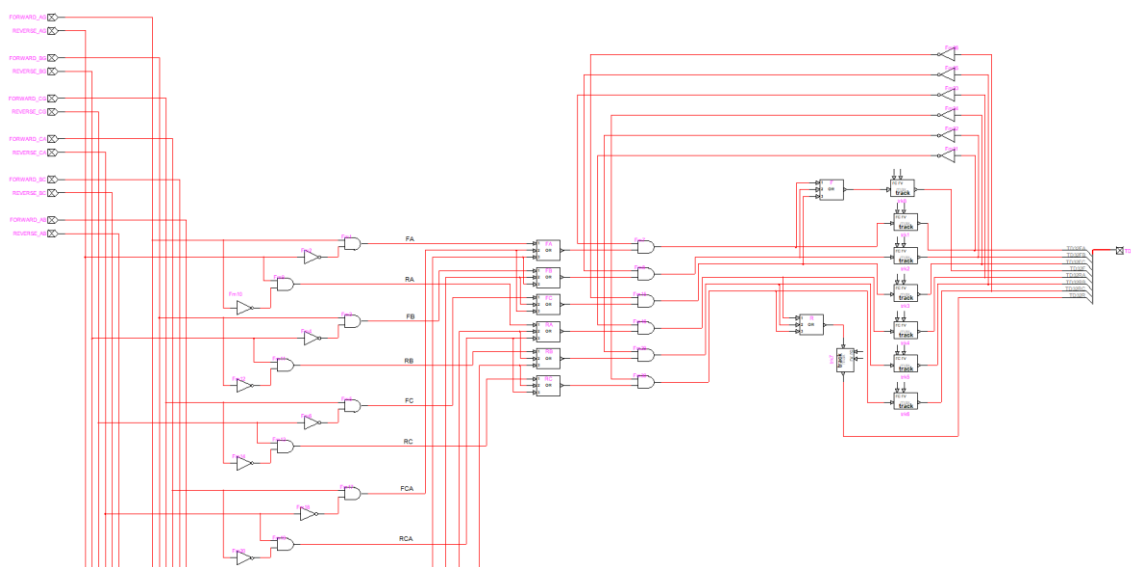


Figure 3-15: Partial View of EMTP “Fault Logic” Block

Due to the size of the fault logic block, not all the internal logic was able to be shown in **Figure 3-15**. However, the same logic implemented for the relay word bits TD32FA and TD32RA, TD32FB and TD32RB, and so on is the same for all the phase and ground loops. Again, there is no basis for this portion of the TD32 element in literature, however, it is necessary for the digital model for it to assert word bits.

3.3 Key Differences Between the Model and the Element

The security margin discussed in **Section 2.2.2** of the SEL-T401L relay was not provided in literature, and an estimation had to be made on initial experimental performance. A value of 0.2 was found to be sufficient in the cases tested, but no claim can be made about the accuracy of this setting.

The previously discussed incremental torque application time was also not explicitly provided in literature; however, an approximate value of a fraction of a

millisecond was provided in [5]. A value of .75 ms was found to be sufficient in the cases tested, but again no claim can be made about the accuracy of this setting.

Regarding the assertion of word bits, the SEL-T401L relay will reset its TD32 word bits after a few milliseconds, or if system conditions change [7]. However, due to the overall simplicity and scope of the model it was determined that the word bits could remain asserted until the model was reset for testing.

In the SEL-T401L the TD32 element is utilized just for directional fault detection, not for tripping. Its outputs are utilized in the relay's PILOT logic which can act as a directional comparison scheme (i.e., it compares the assertions of other directional elements within the relay) [7]. Because of this, there is a coordination delay between the local and remote relays. PILOT logic has not been implemented in the model as the TD32 is the only element under study in this paper. As such, the elements are allowed to act independently of one another, which may produce differing reaction times between the local and remote relays within the model. Additionally, it is expected that when the TD32 is utilized in the PILOT scheme it will typically assert in less than 1.5 ms [7].

3.4 Preliminary Verification of the EMTP Model

A set of preliminary tests were conducted within the simple model system to determine the validity of the digital TD32 element. Validity was determined by several parameters: correct fault direction assertion, element response time, and general performance across multiple fault conditions when compared with the SEL-T401L relay. This section is intended to act merely as a confirmation of the design process presented in this paper as opposed to an in-depth analysis of the element, which is conducted later in this thesis.

Testing was conducted on the simple system under L-G, L-L, and 3PG fault types at several locations both on and outside of the protected line. Step size was set to 1 μ s, which corresponds to a sampling frequency of 1 MHz, and the total simulation time was set to 200 ms. The methods presented in [9] were utilized to export EMTP simulation data from the model system's CTs and PTs for testing on the SEL-T401L relay. The collected data is presented in **Table 3-1**.

Table 3-1: Preliminary Data

Preliminary TD32 Model Testing														Local Response Time (ms)	Remote Response Time (ms)										
Fault Distance from Local (km)	Fault Type	Approximate Fault Inception Angle	Device	Local Relay Word Bits								Remote Relay Word Bits													
				TD32F	TD32FA	TD32FB	TD32FC	TD32R	TD32RA	TD32RB	TD32RC	TD32F	TD32FA			TD32FB	TD32FC	TD32R	TD32RA	TD32RB	TD32RC				
87.5	AG	90°	Model T401L	x	x									x	x							0.78	0.81		
			Model T401L	x	x									x	x								1.1	1.1	
	BC	90°	Model T401L	x		x	x							x		x	x						0.67	0.68	
			Model T401L	x		x	x							x		x	x						1.3	1.2	
	3PG	NA	Model T401L	x	x	x	x							x	x	x	x						0.65	0.65	
			Model T401L	x	x	x	x							x	x	x	x						1.1	1.1	
-75	AG	90°	Model T401L					x	x					x	x	x							0.77	1.43	
			Model T401L					x	x					x	x								1.1	1.3	
	BC	90°	Model T401L					x		x	x			x		x	x						0.65	1.24	
			Model T401L					x		x	x			x		x	x						1.4	1.2	
	3PG	NA	Model T401L					x	x	x	x			x	x	x	x						0.63	1.22	
			Model T401L					x	x	x	x			x	x	x	x						1.1	1.1	
250	AG	90°	Model T401L	x	x	x																	1.38	0.72	
			Model T401L	x	x																			1.3	1.3
	BC	90°	Model T401L	x		x	x																1.2	0.61	
			Model T401L	x		x	x																	1.3	1.3
	3PG	NA	Model T401L	x	x	x	x																1.12	0.59	
			Model T401L	x	x	x	x																	1.1	1.1

As previously noted, it is believed that there is a coordination delay present in the SEL-T401L relay that is not implemented in the model. For this reason, the response times of the SEL-T401L relays were within ± 0.2 ms of one another. If a similar delay were to be applied based on the results collected from the model, it can be observed that the response times are similar. For instance, when testing the 3PG fault located 250 km from the local relay it exhibited a response time of 1.12 ms whereas the SEL-T401L relay exhibited a response time of 1.1 ms. In the worst-case scenario, the model response time differed from the SEL-T401L relay by 0.45 ms in the case of a 3PG fault at the midpoint of the protected line.

There were two misoperations of the model for faults outside of the protected line, shown in **Table 3-1**, highlighted in orange. These two instances occurred when there was an A-G fault located at the midpoint of the external lines. The misoperation was in part due to the model's starting logic erroneously asserting an A-B fault condition. The SEL-T401L relay TD32 operated correctly when presented with the same test scenario. The SEL-T401L relay's relevant word bits are shown in **Figure 3-16** and the model's word bits are shown in **Figure 3-17**. ABFLT, BGFLT, and BCFLT are the three word bits that would have activated the TD32 element's logic for the B-G loop. None of these word bits were asserted for the SEL-T401L relay, however, the model's starting logic asserted ABFLT which activated the B-G and A-B loops for the EMTP model.

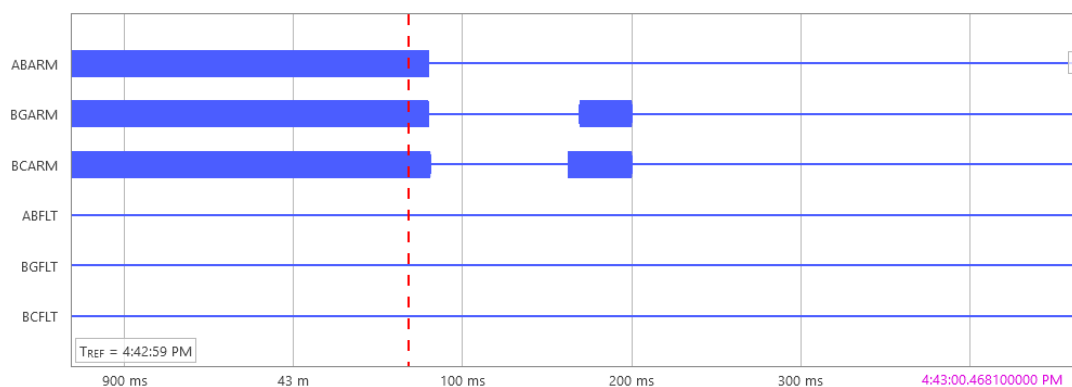


Figure 3-16: Relay Starting Logic Wordbits, viewed in SEL SynchroWAVE.

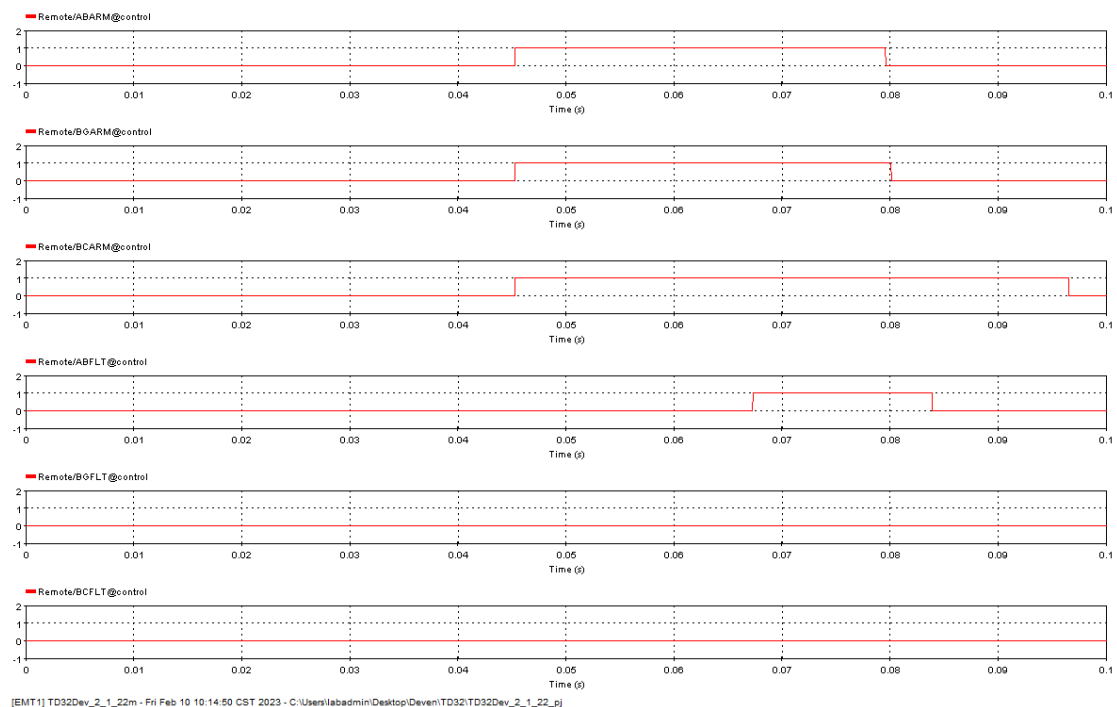


Figure 3-17: EMTTP Model Starting Logic Wordbits.

An automation script, discussed in greater detail in Chapter 4, and utilized extensively in Chapter 6 to gather the results presented there, was utilized to introduce A-G faults along the protected line in 2 km increments. This was done to further observe the impact of the starting logic on the TD32 model and to confirm whether or not the previously observed misoperations would ever cause an incorrect fault direction assertion.

The data collected from this test can be seen in **Figure 3-18** and **Figure 3-19**. Note that the x-axis of these figures corresponds to the distance from the local relay in km. The tests were performed in 2 km increments, so test 0 was performed at 2 km, test 10 was performed at 22 km, test 20 at 42 km, and so on from the local relay. The modeled element correctly determined the fault direction for each case tested and never asserted a reverse fault condition. However, for faults near the relay, the modeled element

would assert a fault on both phases A and B. Since the TD32 is a supervisory element, this misoperation is not critical as either the TD67 or TD21 would both need to operate to cause an erroneous trip.

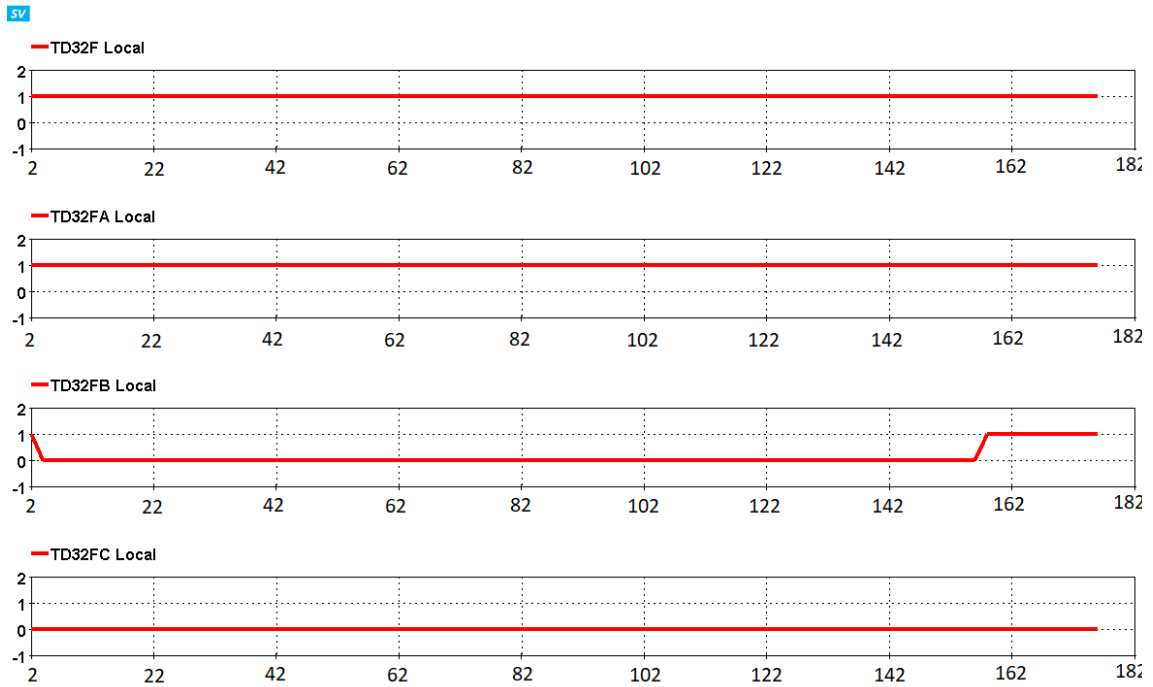


Figure 3-18: Local Relay EMTP Model Results for A-G Faults Along the Protected Line.

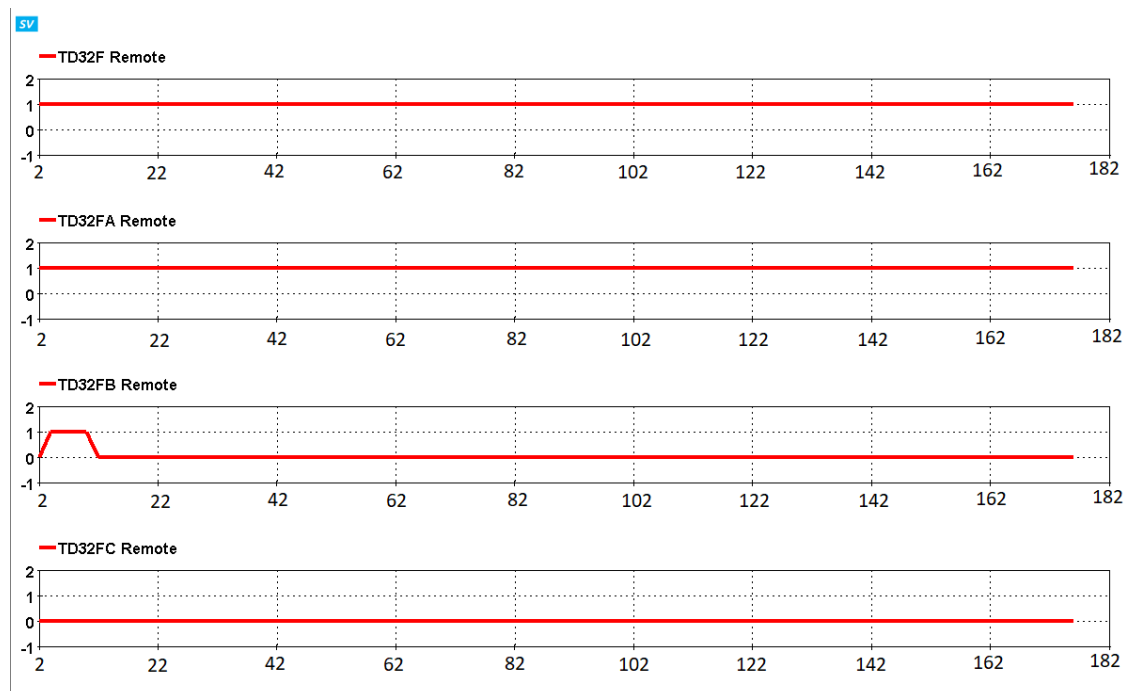


Figure 3-19: Remote Relay EMTP Model Results for A-G Faults Along the Protected Line.

In the current version of the model, the TD67 has not been implemented, so no tests can be performed at this time to determine whether a mis-trip would occur. However, the TD32 is specifically tuned for sensitivity and speed, so it is unlikely that the other elements which are responsible for tripping would also be impacted. Additionally, the only word bit utilized in initializing the TD67 is TD32F, where the other individual phases (TD32FA, TD32FB, and TD32FC) are utilized in the PILOT scheme as seen in **Figure 3-20**. For this reason, the model effectively operated correctly for each case tested.

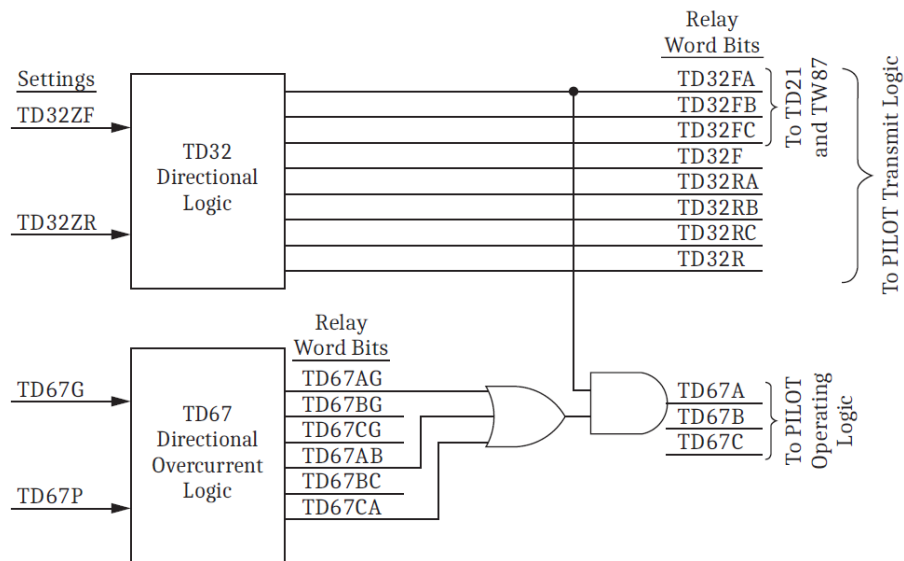


Figure 3-20: TD32 Word Bit Overview as Seen in the T401L Instruction Manual [7].

CHAPTER 4

SIMULATION AUTOMATION WITHIN EMTP

4.1 Introduction

In the previous chapter, the TD32 element was modeled in EMTP and validated using hardware. The operation of the element was verified using simple tests, such as introducing faults at the midpoints of the lines within the simple test system. After the general operation of the element was verified, the performance of the element needed to be analyzed in greater detail. This involves running a number of tests that wouldn't be feasible to do manually. To solve this problem, an automation script was developed so that a large number of tests could be run automatically, such as making changes to fault inception angle, fault location, etc.

4.2 Basic Overview of Automation Scripting

EMTP allows the user to create scripts using a scripting language based on Javascript syntax. The use of scripting allows the user to effectively automate the modification of parameters within a given circuit. It is a powerful feature that was heavily utilized throughout the development of the TD32 element presented in this thesis to run bulk simulations in EMTP during model verification and analysis.

Figure 4-1 is a complete flowchart of the automation script used throughout this thesis. Individual blocks and intricacies of the script are explained in more detail in the

following sections, with example code provided where necessary. In brief, the script will run multiple simulations and save the results from each respective simulation into a designated output file.

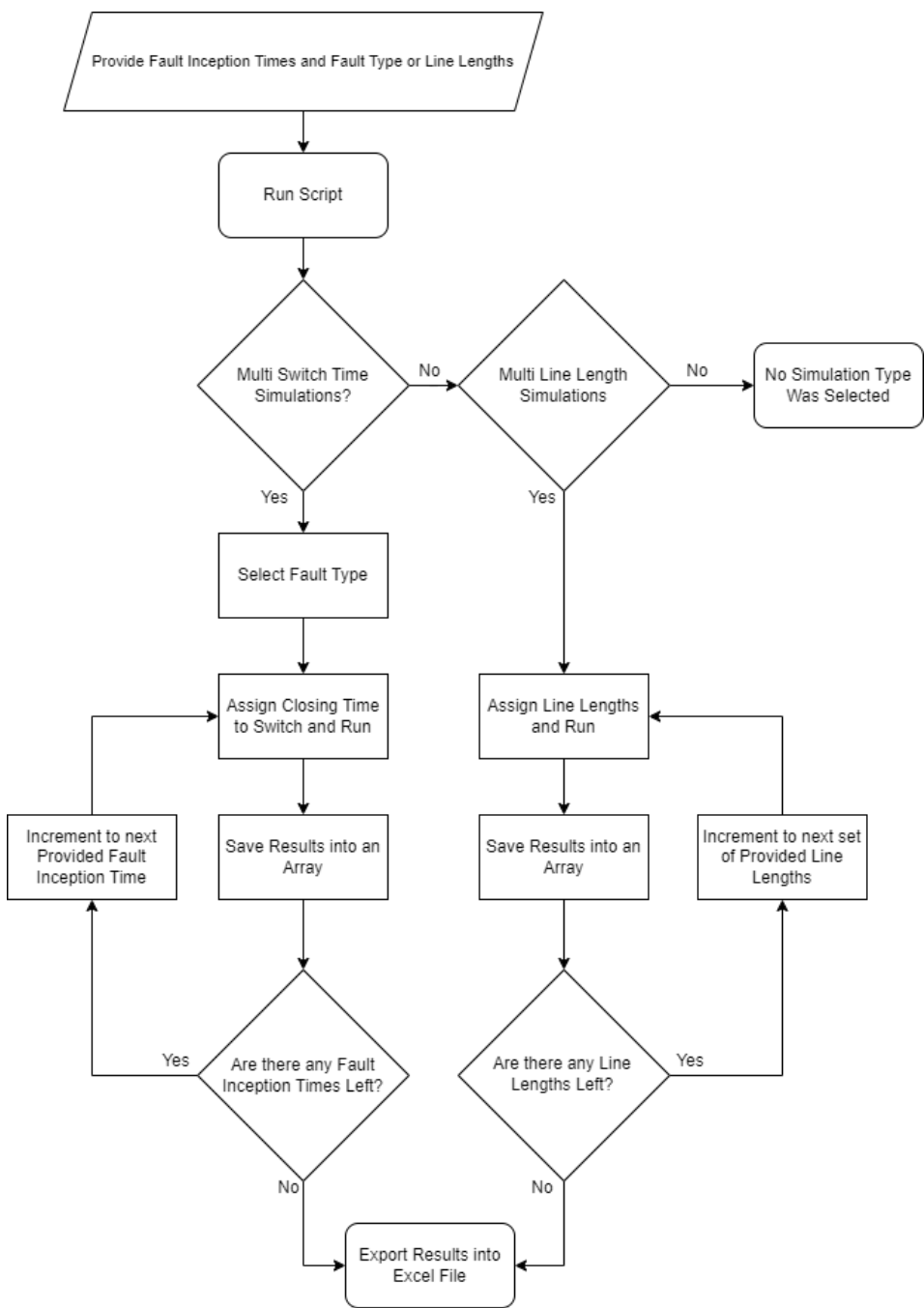


Figure 4-1: Complete Flowchart For Automation Script

Most of the functionality of the script comes from pre-built libraries provided by EMTP. Most importantly, the ability to interact with files on the machine where simulations are being run is provided by the “check_my_filename,” “find_data_in_object,” “make_file_name,” and the “SPFile” libraries. To interact with switches, the “sw0_m” library was used, which has built-in commands for changing switching times. However, there are no such libraries for changing the length of lines and this had to be done in a more roundabout way as discussed later. Starting EMTP simulations was accomplished through the “run_emtp” library and changing simulation parameters was accomplished via the “simoptions” library. These built-in resources are essential to the automation process.

For a simulation to be run, a netlist must first be generated. This is accomplished by utilizing the “Netlist_file_name();” command from the “make_file_name” library. This will return the full path to the netlist file and allow other functions to access it later. A new netlist file must be generated for every respective simulation, otherwise, the script would continually rerun the first case (i.e., it would simply reuse the same netlist file for each simulation). So, to generate a netlist file for every iteration, the “runExportToString” function from the built-in “DWStatic” library was used with its input being the built-in EMTP script used to generate netlist files. Once a new netlist file has been generated for a particular iteration, the “run_emtp()” command is used with its input arguments being the full path to the newly generated netlist file to allow for multiple simulations to be run.

To extract the necessary data from simulations, the script relies upon the “Mplot” utility which is also built into EMTP. Mplot is a program written entirely in Matlab that

can extract raw signal data from .mda and .m format files generated by EMTP and either parse or plot the data. Mplot is also able to be accessed through scripting via the “mplot_m” library and can accept commands in Matlab syntax. It has many functionalities, however, the only function utilized for this thesis was the “MINMAX” search command. As the name would suggest, it searches an assigned variable for the maximum and minimum value over a given interval. This is useful for determining fault locations and whether or not certain elements, mainly the TD32 in this case, have tripped. It will then copy this data into a .txt file which contains the maximum y-values, x-values, and the x-y coordinates of where the maximum y-value occurred. The resulting file can then be parsed and the “find_data_in_object” library can then be used to extract these values from the .txt file containing the results of the simulation into an array. Since we are only interested in the TD32, the only data needed from the set produced by Mplot is the maximum y-value for the scopes such as TD32FA, TD32FB, and so on. As stated, these data points are stored in respective arrays and then the array index is incremented to allow for the next set of data to be saved. The script is then able to move on to the next simulation assuming there are more switching times or line lengths provided.

4.3 Initializing Libraries and Their Functions

Initializing libraries within the EMTP scripting environment can be done by parsing the source script file which is included with EMTP. The full file path is not needed for libraries included with EMTP as such files should be located in certain positions when EMTP is installed. An example of initializing libraries in the scripting environment can be seen in **Figure 4-2**. After a library has been parsed, its functions can be used throughout the script wherever needed.

```

//Initializing Libraries
parseScriptFile('simoptions/object_creator.dwj'); //needed to interact with simulation settings
parseScriptFile('simoptions/open.dwj'); //needed to interact with simulation settings
parseScriptFile('simoptions/save.dwj'); //needed to interact with simulation settings
parseScriptFile('make_file_name.dwj'); //allows for creation of and interaction with files
parseScriptFile('run_emtp.dwj'); //always needed to run simulations
parseScriptFile('mplot_m.dwj'); //needed to interact with Mplot
parseScriptFile('check_my_filename.dwj'); //locates and/or generates file paths
parseScriptFile('find_data_in_object.dwj'); //for searching through data within objects
parseScriptFile('sw0_m.dwj'); //switch object creator

```

Figure 4-2: Script To Initialize Libraries in EMTP

4.3.1 The “simoptions” Library

The simulation options (“simoptions”) library contains three separate files that need to be parsed for the automation script, as seen in **Figure 4-2**. The names of these files are self-explanatory, the “object creator” file allows for the creation of an object which then can be acted upon with methods to change various simulation options, while the “open” and “save” files allow for simulations attributes to be changed and saved after being modified, respectfully.

Figure 4-3 shows the script necessary to create a simulation options object/device within the EMTP scripting environment. This object can then be acted upon by the various methods seen in the figure. Each of the methods is self-explanatory and corresponds to the simulation options menu within EMTP seen in **Figure 4-4**. The “SaveData()” function simply applies any options that have been modified prior in the script (equivalent to clicking “ok”).

```

//This section can be used to modify simulation parameters
oSimoptions = new oDevice_(cct);
oSimoptions.LoadFlow = 0;
oSimoptions.StartFromLoadFlow = 0;
oSimoptions.steadystate = 1;
oSimoptions.timedomain = 1;
oSimoptions.fscan = 0;
oSimoptions.Dt = 1;
oSimoptions.tmax = .1;
oSimoptions.SaveData();

```

Figure 4-3: Script for Modifying Simulation Options

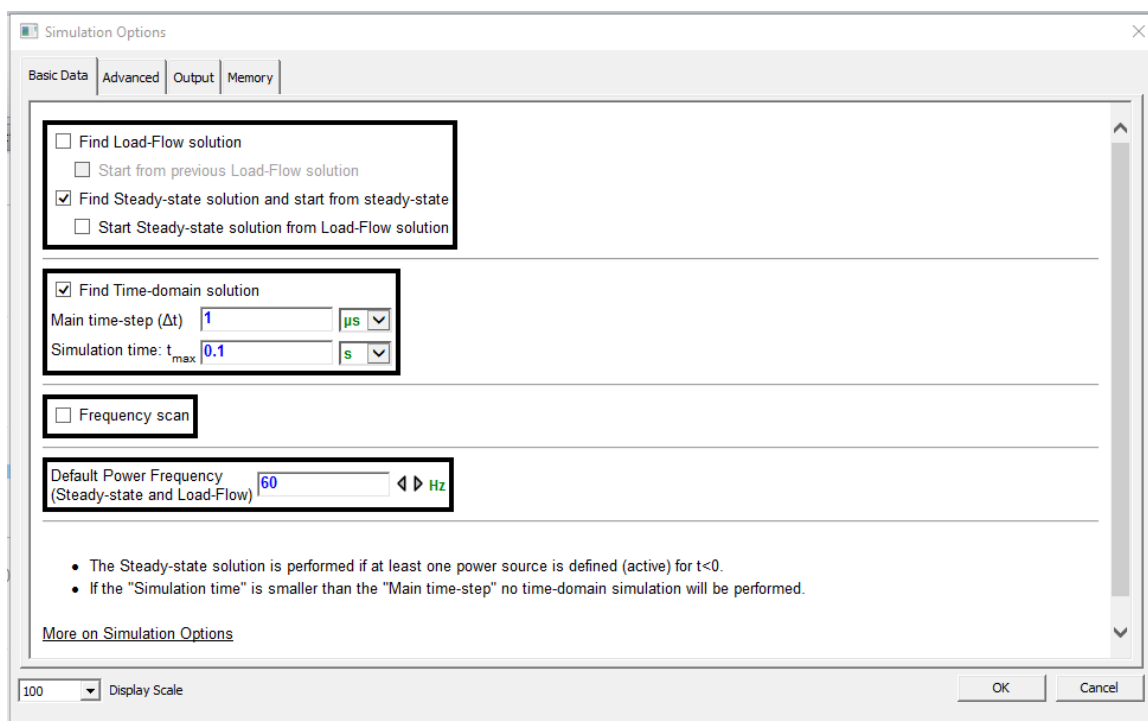


Figure 4-4: EMTP Simulation Options, Basic Data

4.3.2 The “make_file_name” Library

The “make_file_name” library allows the user to interact with files from the EMTP scripting environment. In the context of the automation script, it is used to retrieve the full path to the netlist file of the circuit for which multiple simulations are being run by using the “Netlist_file_name();” function. The full file path is used later in the script to

generate new netlist files for each respective simulation. See **Figure 4-5** for details on how this is implemented in the automation script.

```
//runs the simulation for each iteration
var netlist_file = Netlist_file_name();
str = runExportToString('report scripts/EMTP Netlist.rfm', null, 'NETLIST');
writeln('Starting simulation ', sim+1);
run_emtp(netlist_file, false);
writeln('Simulation complete');
```

Figure 4-5: Script for Utilizing the “make_file_name” Library To Find The Full Path Of The Netlist File

4.3.3 The “run_emtp” Library

The “run_emtp” library is simply a function that allows simulations to be run from a script. It is the equivalent of pressing the run button on the simulate tab of the EMTP software. It has two necessary arguments and two optional arguments. For the automation script, only the necessary arguments were provided. The first necessary argument is the full path to the netlist file. As explained in **subsection 0**, the “Netlist_file_name();” function is used to locate the full path to the netlist file. The second necessary argument is the visibility of the wait bar (effectively the simulation progress bar). For running multiple simulations the second argument must always be false, i.e., the wait bar must not be shown. Due to the way this library is configured the wait bar must be hidden when running multiple simulations otherwise the simulation will fail to run.

The two necessary inputs of this function can be seen in **Figure 4-5** as the full path to the netlist file and “false” for the option of displaying the wait bar.

4.3.4 The “check_my_filename” Library

The “check_my_filename” library is utilized for handling file names and relative paths. For the automation script, the “check_my_filename” function from this library was used to find the correct file name of the results file generated by Mplot. The script used to implement this is depicted in **Figure 4-6**, the “check_my_filename” function is used in conjunction with the “parseScriptFile();” function to extract the results vector V from the .txt file generated by Mplot after a simulation is completed.

```
//parses results file to extract results vector
parseScriptFile(check_my_filename(project_folder_name)+'\\results.txt');

//saving sim results to respective arrays
//these arrays are created earlier in the script
TD32FL[sim]      = V[0].maxy;
TD32FR[sim]      = V[1].maxy;
TD32FAL[sim]     = V[2].maxy;
TD32FAR[sim]     = V[3].maxy;
TD32FBL[sim]     = V[4].maxy;
TD32FBR[sim]     = V[5].maxy;
TD32FCL[sim]     = V[6].maxy;
TD32FCR[sim]     = V[7].maxy;
TD32RL[sim]      = V[8].maxy;
TD32RR[sim]      = V[9].maxy;
TD32RAL[sim]     = V[10].maxy;
TD32RAR[sim]     = V[11].maxy;
TD32RBL[sim]     = V[12].maxy;
TD32RBR[sim]     = V[13].maxy;
TD32RCL[sim]     = V[14].maxy;
TD32RCR[sim]     = V[15].maxy;
```

Figure 4-6: Script for Extracting Values from Mplot Results File.

4.3.5 The “sw0_m” Library

The “sw0_m” library is used to create a switch object and change its attributes, such as closing and opening times. This is the main method used to change switching times within the automation script. This library contains many more functions, but those used in **Figure 4-7** are the only methods utilized for the automation script. In the figure, the provided array of fault inception times is applied to phase A of the L-G fault switch and the switch is set to never open by providing a very large opening time.

```
var oswitchDevice = new oDevice_sw0(faultSwitch); //creates a device from switch
//changing switch close/open times for the LG switch
oswitchDevice.tclose_A = faultTimes[sim];
oswitchDevice.topen_A = '1E15';
oswitchDevice.SaveData();
```

Figure 4-7: Creating A Switch Object And Acting On It.

Before a switch object can be created, a switch device must first be selected within the circuit diagram in which multiple simulations are being run. This selection process is described in the following section.

4.3.6 Selecting Devices Within a Circuit Using Scripts

Selecting devices within EMTP is a process of searching layers of the circuit for devices and then pushing into the subcircuits of those devices until you reach the device you are attempting to interact with. This is accomplished through the use of the “DWCircuit” library and its functions. **Figure 4-8** is how this process is implemented in the EMTP scripting environment.

```

//Finding the fault location//
//search the circuit for object named {faultLocation}, this selects the line block where the fault will be.
//It will save all instances of {faultLocation} in an array, however there will SHOULD only be one instance
//of {faultLocation}, so it'll be an array with one element.

var faultLine = cct.devices('Name',faultLocation);

//This loop simply checks to make sure the things searched for actually exists
if(faultLine==null || faultLine.length==0){
    alert('The fault location does not exist or was entered incorrectly.');
```

```

    halt(); //Stop the script.
}

var faultBlock = faultLine[0];
var cctFaultBlock = faultBlock.subCircuit(); //selecting subcircuit of line block where fault where occur

var faultSwitch = cctFaultBlock.devices('Name', 'DEV1');

if(faultSwitch == null || faultSwitch.length == 0){
    alert('The fault switch does not exist!');
    halt();
}

var faultSwitchBlock = faultSwitch[0];
var cctFaultSwitchBlock = faultSwitchBlock.subCircuit();

var swc = cctFaultSwitchBlock.devices('Name',faultType);

if(swc==null || swc.length==0){
    alert('That switch does not exist within the fault block or was entered incorrectly.');
```

```

    halt();
}

var faultSwitch = swc[0];

var oswitchDevice = new oDevice_sw0(faultSwitch); //creates a device from switch

```

Figure 4-8: Process used For Finding And Selecting Devices With Scripting

To select a circuit object (such as a switch, line, or block), the top layer of the circuit must first be selected. One method of selecting the top layer of the circuit, and the method used in this paper, is by calling the “currentCircuit();” function of the “DWStatic” library and assigning it the variable “cct.” After this is complete, the variable “cct” will always refer to the topmost layer of the opened circuit.

The script pictured in **Figure 4-8** is specifically used to select the different switches within the fault block of the simple system, shown in **Figure 3-1**, **Figure 3-4**, and **Figure 3-5**. It uses the “devices();” function to search for devices within the currently selected circuit layer by name. Its input arguments are the parameter types used for the search (such as a name or value), and the particular parameter being searched for (what is

the device named, or the particular value being searched for). In this case, the variable “faultLocation” and “faultType” are simply strings that are equal to the name of the line block where the fault is occurring and the name of the switch corresponding to the particular fault type, respectively.

Once the “devices();” function finishes its search, it will output all results into an array. Though, there should only be one element within the said array as devices within the circuit layer should have unique names. EMTP will not allow you to have two devices named “Line1,” for instance. This single element is then assigned to a variable where the function “subCircuit();” is used to push into the subcircuit of this device. From here this process can repeat until the desired device is selected.

4.3.7 Changing Constant Parameter Line Lengths

EMTP does not contain any libraries that easily allow for the changing of line lengths like the libraries for switches and changing switching times, at the time of writing this thesis. Instead, line lengths must be modified manually. This process involves modifying the attribute list of the line object within EMTP by directly modifying the string for both the “FormData” and “ModelData” fields. Both of these data sets contain details that are used to generate the line before a simulation is run. This string is replaced for each respective simulation the script runs, similar to how a new netlist file must be generated for each respective simulation. This process is implemented in the EMTP scripting environment using the section of the script shown in **Figure 4-9**. The two string variables “formData1” and “formData2” are based on the string seen in **Figure 4-10**, and the string variables “modelData1” and “modelData2” are based on the string seen in

Figure 4-11. The reason there are two variables per line is because each line block is broken up into two sections to allow the fault location to be adjusted.

```
//creating strings to assign to line models
var formData1 = '3,3,,1,'+leftLength[sim]+' ,1,,,,mH,uF,,,,,kV, ';
var modelData1 = leftLength[sim]+' \t.3\t3\t.008\n'+leftLength[sim]+' \t.02\t.9\t.0126\n0\n0\n0';
var formData2 = '3,3,,1,'+rightLength[sim]+' ,1,,,,mH,uF,,,,,kV, ';
var modelData2 = rightLength[sim]+' \t.3\t3\t.008\n'+rightLength[sim]+' \t.02\t.9\t.0126\n0\n0\n0';

//assigning created strings to line models
TLM1.setAttribute('FormData',formData1);
TLM1.setAttribute('ModelData',modelData1);
TLM2.setAttribute('FormData',formData2);
TLM2.setAttribute('ModelData',modelData2);
```

Figure 4-9: Script Used To Replace The Model And Form Data Of Constant Parameter Lines

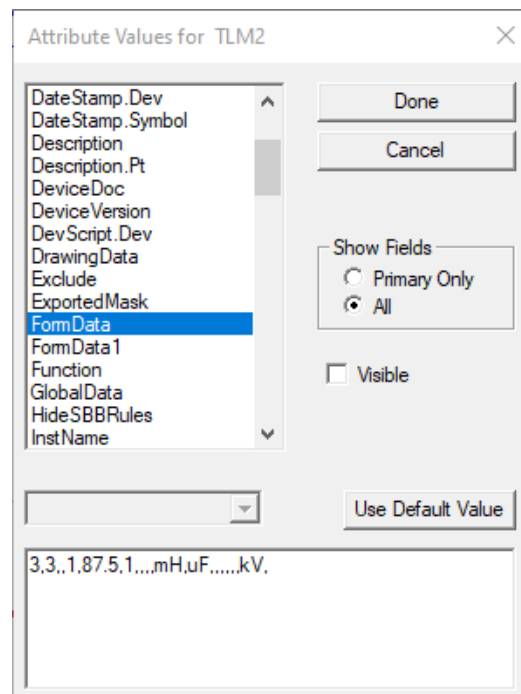


Figure 4-10: The Format Of The “FormData” String

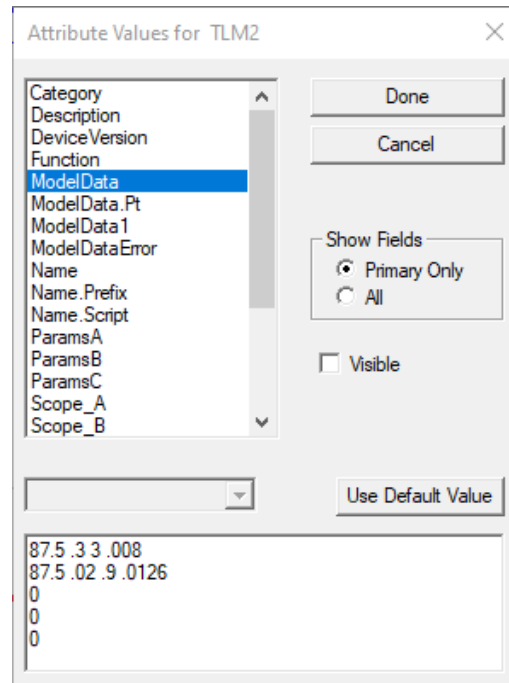


Figure 4-11: The Format of The “Model-Data” String

4.4 Using Mplot For Automatic Data Extraction

To interact with Mplot, an Mplot device must first be created in a manner similar to creating switch devices seen in **subsection 4.3.5**. The method used in this thesis can be seen in **Figure 4-12** along with the required settings needed to run multiple simulations. Similar to the simulation options seen in **subsection 4.3.1**, these settings correspond to Mplot settings seen in **Figure 4-13**.

```
//creating Mplot device and configuring settings to allow interaction through scripting
mplotButton = cct.devices('Script.Open.Dev','mplot_i.dwj');
mplotButton = mplotButton[0];
oMplot = new oDevice_mplot(mplotButton);
oMplot.waitmode = -1;
oMplot.ExtraTags = '0';
oMplot.ExtraFiles = '1';
oMplot.BaseCaseCheck='0';
```

Figure 4-12: Script Used For Creating Mplot Button and Adjusting Settings.

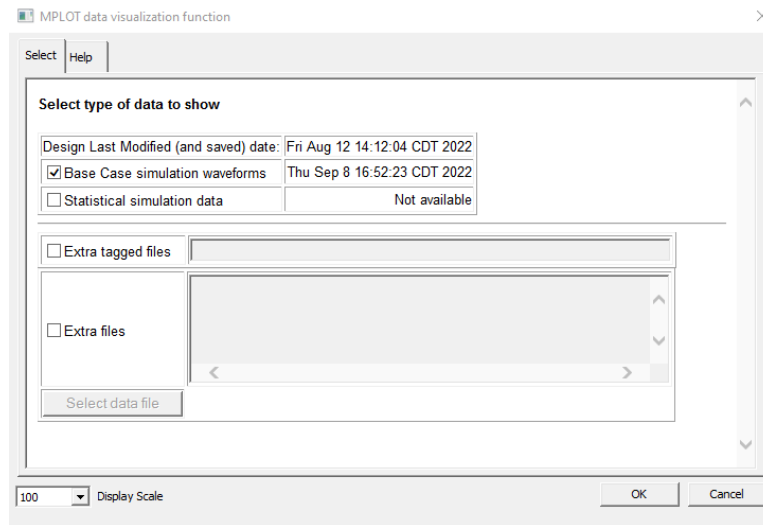


Figure 4-13: Mplot Utility Settings.

To issue commands to Mplot they have to be written into a .txt file in Matlab syntax. This .txt file will then be read and interpreted by Mplot into Matlab commands. This .txt file may be created manually, or automatically generated within the script itself by utilizing the previously discussed libraries. For the automation purposes of this paper, the Mplot command file was generated automatically within the automation script itself.

There are two things to be aware of when creating command files. First, the Mplot utility button must be on the top level of the circuit on which the script is being run on. For instance, the Mplot button may be seen on the top layer of the simple system in **Figure 3-1**. The Mplot button's location can be seen in **Figure 4-14**. Second, the command file generated by the automation script, seen in **Figure 4-15** must contain no spaces due to the way Mplot interprets the .txt file.

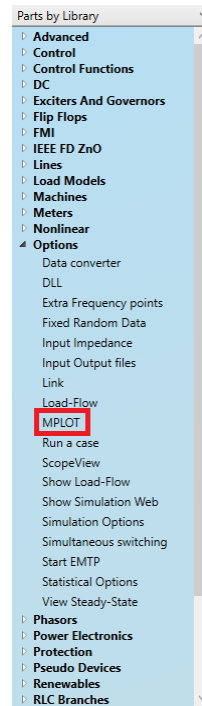


Figure 4-14: Location Of The Mplot Button In EMTP’s Parts Library

```

////////////////////////////////////Creating Mplot Command File////////////////////////////////////
commandFile = make_Newfile_name('MplotCommandFile.txt');
var matlabFile = make_file_name('m.m');

var mplotCommandFile = 'waitbar=0;\n'
    +'PlotCase(1).type='\MINMAX'\;\n'
    +'PlotCase(1).filename='\results.txt'\;\n'
    +'filn(1)=\'+matlabFile+'\;\n'
    +'PlotCase(1).var_y(1)=\SEL_T401L_Local/DETWFL@control@1'\;\n'
    +'PlotCase(1).var_y(2)=\SEL_T401L_Remote/DETWFL@control@1'\;\n'
    +'PlotCase(1).var_y(3)=\SEL_T401L_Local/TD21@control@1'\;\n'
    +'PlotCase(1).var_y(4)=\SEL_T401L_Remote/TD21@control@1'\;\n'
    +'PlotCase(1).var_y(5)=\SEL_T401L_Local/TW87@control@1'\;\n'
    +'PlotCase(1).var_y(6)=\SEL_T401L_Remote/TW87@control@1'\;\n'
    +'PlotCase(1).var_y(7)=\SEL_T401L_Local/DETWFL/TW_ARIV_TIME/TW_TOA/TW_MAG@control@1'\;\n'
    +'PlotCase(1).var_y(8)=\SEL_T401L_Remote/DETWFL/TW_ARIV_TIME/TW_TOA/TW_MAG@control@1'\;\n'
    +'PlotCase(1).var_y(9)=\SEL_T401L_Local/DETWFL_MODE@control@1'\;\n'
    +'PlotCase(1).var_y(10)=\SEL_T401L_Remote/DETWFL_MODE@control@1'\;\n'
    +'PlotCase(1).var_y(11)=\SEL_T401L_Local/TWDD@control@1'\;\n'
    +'PlotCase(1).var_y(12)=\SEL_T401L_Remote/TWDD@control@1'\;\n';


fid = SPFile(commandFile);
fid.openCreate();
fid.appendLine = mplotCommandFile;
fid.close();

```

Figure 4-15: Process For Generating An Mplot Command File Within An Automation Script.

4.4.1 Mplot Command And Output Files

Commands written for Mplot in a .txt file should have no spaces except for those between lines and in the comments. **Figure 4-15** uses the “MINMAX” data parsing command which will extract the values mentioned in **section 4.1** from the defined signals and export them into the designated results .txt file (in this instance literally called “results.txt”). Note here that the name of the file containing the results can be defined as anything and will be saved in the project folder along with the generated command file. A partial example of a results file can be seen in **Figure 4-16**. This is the results file that is later parsed through the EMTP scripting environment for each iteration utilizing the script seen in **Figure 4-6** to recover the vector “V” seen in **Figure 4-16**.



```

results.txt - Notepad
File Edit Format View Help
function Mplot_Data_Holder(){
this.name='';
this.miny=0;
this.maxy=0;
this.minx=0;
this.maxx=0;
this.interval_min=0;
this.interval_max=0;
}
filename = new Array();
var V= new Array();
filename[0]='c:/Users/labadmin/Desktop/Deven/Automation_Testing/AutomationTesting_pj/AutomationTestingm.m';
V[0]= new Mplot_Data_Holder();
V[0].name='SEL_T401L_Local/DETWFL@control@1';
V[0].miny=0;
V[0].maxy=69.9967713112;
V[0].minx=0;
V[0].maxx=0.066885;
V[0].interval_min=0;
V[0].interval_max=0;
V[1]= new Mplot_Data_Holder();
V[1].name='SEL_T401L_Remote/DETWFL@control@1';
V[1].miny=0;
V[1].maxy=105.003228689;
V[1].minx=0;
V[1].maxx=0.068175;
V[1].interval_min=0;
V[1].interval_max=0;
V[2]= new Mplot_Data_Holder();
V[2].name='SEL_T401L_Local/TD21AG@control@1';
V[2].miny=0;
V[2].maxy=1;
V[2].minx=0;
V[2].maxx=0.080698;
V[2].interval_min=0;
V[2].interval_max=0;
Ln 1, Col 1    100%    Windows (CRLF)    UTF-8

```

Figure 4-16: Example Mplot Results File

4.5 Running EMTP Simulations and Data Extraction

Once all libraries have been initialized, and the necessary devices have been created, the script is then able to run simulations and record the necessary data automatically. The purpose of this section is to explain this process in more detail.

4.5.1 Generating an Excel File for Results Storage

Before any iterations are started, an excel file must first be created to store any results that are generated. This can be done through the “make_file_name” library discussed in **subsection 0** and the built-in “SPFile” library within EMTP. First, the “make_Newfile_name();” function is called with its input argument being the desired name of the excel file in the form of a string. This file may then be targeted by the “SPFile” library where the “openCreate();” function is used to generate it. This method will search the circuit’s design folder for any files of the same name, and if none are found then the file is created; if a file is found of the same name, then that file is deleted and replaced by a new, empty file. This is done to ensure that results from previous runs of the automation script do not interfere with current results.

Results are added to the excel file by utilizing SPFile’s “.appendLine();” function. For excel files, this will cause any data, in this case a string containing the results of the respective iteration, to be written onto the file. The cursor will then move to the next row to prevent overwriting of data.

Once all the designated simulations have been run (either a series of fault times or line lengths) then the generated excel file containing the recorded information may be accessed. Once completed, the generated excel file containing the results will be like the one pictured in **Figure 4-17**.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
	Iteration	Fault Time	DETWFL Local	DETWFL Mode Local	TW87 Local	TW87 Operate Time Local	TD21 Local	TD21 Operate Time Local	TW Mag Local	DETWFL Remote	DETWFL Mode Remote	TW87 Remote	TW87 Operate Time Remote	TD21 Remote	TD21 Operate Time Remote	TW Mag Remote
2	Sim 1	0.079397	15.84998	5	1	0.081677	1	0.094864	1.367954	23.73993	5	1	0.083163	1	0.094943	0.859944
3	Sim 2	0.080323	15.84715	5	1	0.082872	1	0.095803	1.2893	23.74276	5	1	0.086141	1	0.095884	0.808715
4	Sim 3	0.081249	15.8472	5	1	0.085089	1	0.09677	1.053882	23.7427	5	1	0.088052	1	0.096858	0.661216
5	Sim 4	0.082175	15.84742	5	1	0.087219	1	0.097799	0.691451	23.74249	5	1	0.088227	1	0.097906	0.434902
6	Sim 5	0.082638	15.85051	5	1	0.087284	1	0.098393	0.475722	23.7394	5	1	0.088303	1	0.098529	0.300464
7	Sim 6	0.08273	15.85016	5	1	0.087275	1	0.098532	0.431189	23.73975	5	1	0.088316	1	0.098681	0.27235
8	Sim 7	0.082823	15.84949	5	1	0.087265	1	0.09869	0.385391	23.74042	5	1	0.088328	1	0.098865	0.243547
9	Sim 8	0.082916	15.84956	5	1	0.087255	1	0.098892	0.339244	23.74035	5	1	0.088338	1	0.099161	0.214558
10	Sim 9	0.083008	15.84965	5	1	0.087244	0	0	0.293184	23.74026	5	1	0.088344	0	0	0.18562
11	Sim 10	0.083101	15.84978	5	1	0.087234	1	0.099923	0.246264	23.74012	5	1	0.088347	1	0.099994	0.156139
12	Sim 11	0.083193	15.84998	5	1	0.087225	1	0.099842	0.199535	23.73993	5	1	0.088347	1	0.099936	0.126776
13	Sim 12	0.083286	15.85028	5	1	0.087215	1	0.099774	0.152045	23.73963	5	1	0.088343	1	0.099882	0.096933
14	Sim 13	0.083378	15.85085	5	1	0.087206	1	0.099722	0.10487	23.73906	5	1	0.088337	1	0.099837	0.067287
15	Sim 14	0.083471	0	5	0	0	1	0.099688	0.056861	0	0	0	0	1	0.099807	0
16	Sim 15	0.083564	0	5	0	0	1	0.099673	0.057024	0	0	0	0	1	0.099793	0
17	Sim 16	0.083656	0	5	0	0	1	0.099677	0.056932	0	0	0	0	1	0.099796	0
18	Sim 17	0.083749	0	5	0	0	1	0.099696	0.086274	0	0	0	0	1	0.099814	0
19	Sim 18	0.083841	15.84731	5	1	0.087213	1	0.099728	0.133609	23.7426	5	1	0.088345	1	0.099844	0.082572
20	Sim 19	0.083934	15.84777	5	1	0.087229	1	0.099771	0.181309	23.74214	5	1	0.088361	1	0.099885	0.112546
21	Sim 20	0.084027	15.84803	5	1	0.08725	1	0.099822	0.228801	23.74188	5	1	0.088381	1	0.099933	0.142388
22	Sim 21	0.084119	15.8482	5	1	0.087275	1	0.099878	0.275519	23.74171	5	1	0.088405	1	0.099987	0.171746
23	Sim 22	0.084212	15.8483	5	1	0.087304	1	0.099941	0.322417	23.74161	5	1	0.088434	0	0	0.201215
24	Sim 23	0.084304	15.8485	5	1	0.087337	0	0	0.36839	23.74141	5	1	0.088467	0	0	0.230158
25	Sim 24	0.084397	15.84951	5	1	0.087374	0	0	0.414463	23.7404	5	1	0.088503	0	0	0.259239
26	Sim 25	0.08449	15.84956	5	1	0.087415	0	0	0.460002	23.74035	5	1	0.088543	0	0	0.287884
27	Sim 26	0.084953	15.84688	5	1	0.087665	0	0	0.677454	23.74303	5	1	0.08879	0	0	0.423667
28	Sim 27	0.085878	15.84696	5	1	0.088337	0	0	1.042297	23.74295	5	1	0.089486	0	0	0.652268
29	Sim 28	0.086804	15.84703	5	1	0.089178	0	0	1.282392	23.74288	5	1	0.090467	0	0	0.803094
30	Sim 29	0.08773	15.84998	5	1	0.090174	0	0	1.367953	23.73993	5	1	0.091831	0	0	0.859943

Figure 4-17: Example Simulation Results Extracted to Excel

CHAPTER 5

TESTING OUTLINE

5.1 Introduction

In Chapter 3, a digital model of an incremental quantity directional element was developed and verified using a simple system. Once the functionality of the element was verified, the performance of the element needed to be analyzed. To accomplish this, a model of a real power system was developed in EMTP using the data provided by an electric utility company. The digital model of the relay was tested under a variety of different fault scenarios within this model system. Files generated from EMTP simulations were converted to the appropriate format using the methods discussed in [9] and played back into the TDRs. Variables such as fault inception angle, fault location (both on and outside the protected line), and fault impedance were varied for the tests. Metrics such as relay word bits and element assertions were collected for comparison purposes.

5.2 The Test System

As mentioned, to analyze the performance of the modeled TD32 element, tests were performed using a model of a portion of a real power system. The system was modeled using the data provided by a utility company, and as such, specifics cannot be

disclosed. However, an overview of the system is shown in **Figure 5-1** and a discussion of the basics is provided in the following text.

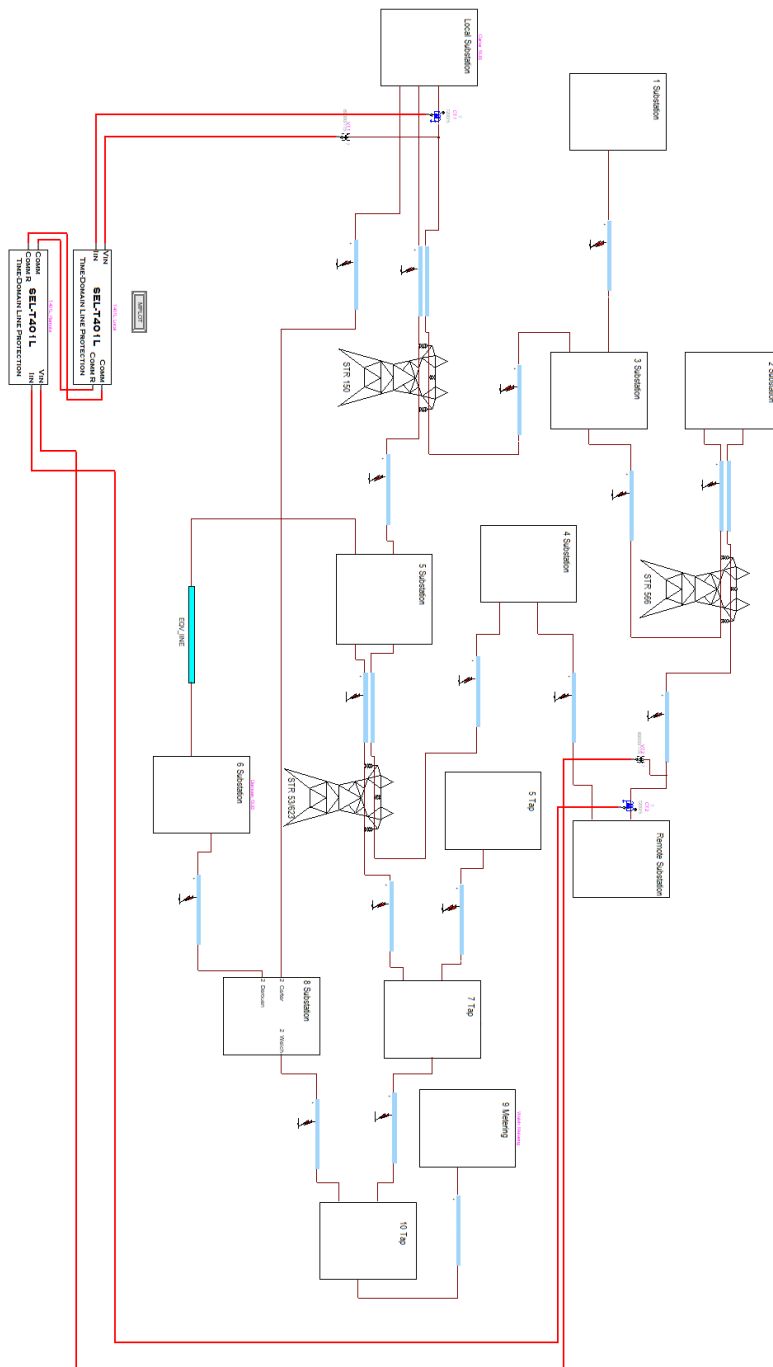


Figure 5-1: EMTP Model System Utilized for Testing, Modeled Using Utility Provided Data.

The system used for the study is a 69 kV line from the Local to Remote buses and the surrounding system two buses away from each station. The system contains 13 buses and 15 lines in total. The protected line is 39.59km (24.6 miles) in length and has multiple mutually coupled line segments and a parallel path. The PT ratio at both the Local and Remote Stations is 69,000:115 V and the CT ratio at both stations is 240:5. There are transformers at most of the buses, and each transformer has a load on the secondary that is equivalent to 75 percent of its rated MVA. EMTP's built-in frequency-dependent line model was utilized to model the system's transmission lines.

This system was chosen for testing because it presented many areas and scenarios in which the element could be verified against a real-world incremental quantity directional element.

5.2.1 Model Settings

Both the TD32 digital model and the TD32 element of the SEL-T401L relay have two configurable settings: TD32ZF and TD32ZR. These settings are discussed in greater detail in Section 2.2 of Chapter 2. Essentially, they represent a percentage of the line (TD32ZR) and source (TD32ZF) positive sequence impedances. The default values suggested in [7] of 0.3 Ω -secondary were utilized for testing within the model system. In addition, two settings were added to the TD32 digital model: TD32Delay and TD32SM, which are the incremental torque application time and security margin, respectively. These two additional settings are discussed in greater detail in Section 2.2.2 of Chapter 2. Note that the values for TD32Delay and TD32SM were determined experimentally (as discussed in Section 3.3 of Chapter 3), and as such are not generally meant to be configurable settings.

The other elements necessary for TD32 functionality (arming and starting logic) were previously modeled in [8]. The other settings are shown in **Figure 5-2** and were set as suggested by the utility company.

Initial Parameters-Values: Grid mode input			Insert row
	Parameter	Value	Definition/Comments
1	Z0MAG	21.45	Zero Sequence Line Impedance in [ohm-sec]
2	Z0ANG	73.6	Zero Sequence Line Impedance angle in [deg]
3	Z1MAG	7.35	Positive Sequence Line Impedance in [ohm-sec]
4	Z1ANG	71.5	Positive Sequence Line Impedance angle in [deg]
5	TD21MP	0.79	Reach Setting for TD21 Phase Loops
6	TD21MG	0.79	Reach Setting for TD21 Ground Loops
7	VNOM	115	PT Nominal voltage (L-L), [V-sec]
8	INOM	5	Relay Nominal Current (5A or 1A)
9	TWLPT	132.07	TW Line Propagation Time (us)
10	Line_Length	24.6	Line Length (miles)
11	TWTEST	0	TW87 Test Mode (ON:1, OFF:0)
12	XC	0	Series Capacitor Reactance (ohm-sec)
13	EXTSC	0	External Series Compensation (ON:1, OFF:0)
14	TD50P	0.3	TW87 Phase Overcurrent Supervision threshold [A-sec]
15	TD50G	0.3	TW87 Ground Overcurrent Supervision threshold [A-sec]
16	TW87BL1	.662	Blocking Location 1 (Line Length pu)
17	TW87BR1	0.02	Blocking Location 1 Radius (Line Length pu)
18	TW87BL2	10	Blocking Location 2 (Line Length pu)
19	TW87BR2	0	Blocking Location 2 Radius (Line Length pu)
20			
21	OP27	40	Open-Pole Detection Undervoltage Threshold (L-N) Default = 40
22	OPMODE	1	Open-Pole Detection Mode (Range:52,V),(Default: 52),(52:1, V:0)
23	OPDO	0.015	Open-Pole Security Dropout Time Delay (Range:0s-1s), (Default:0.015s)
24			
25	Vrst_SF	1.06	Scaling Factor for Vrst for Added Security
26	Vrst_MIN	50	Min Value for Restraining Voltage, ZC Protection
27	pi	3.141592654	
28			
29	TD32ZF	-(.3)	TD32 Forward Restraining Torque
30	TD32ZR	(.3)	TD32 Reverse Restraining Torque
31	TD32Delay	.00075	Incremental Torque Application Time
32	TD32SM	.2	TD32 Security Margin

Figure 5-2: EMTP Digital Model Settings

In **Figure 5-2**, ZOMAG, Z0ANG, Z1MAG, and Z1ANG are the secondary positive-sequence impedance magnitudes and angles for the zero-sequence and positive-sequence line impedance, respectively. TD21MP and TD21MG are the phase and ground

reach for the TD21 element. VNOM is the nominal line-to-line RMS voltage seen by the secondary of the PT. Likewise, INOM is the nominal current seen by the CT. TD50P and TD50G settings correspond to the incremental overcurrent supervision thresholds for the phase and ground loops, respectively.

TW87BL1, TW87BLR1, TW87BL2, and TW87BLR2 correspond to the blocking region settings for the TW87 element. They create a blocking region around tap points to prevent the relays from tripping for faults that occur on the tapped line. As shown in the system layout in **Figure 5-1** above, the protected line consists of one tapped line which is from substations 3 to 1. The TW87BL1 corresponds to the distance of the blocking location from the relay. Next, the TW87BLR1 corresponds to the radius, in terms of pu, of the blocking region around the blocking location.

Vrst_SF is a scaling factor for the TD21 restraining signal and is sometimes referred to as the “k” factor. Vrst_MIN is the minimum value for the TD21 restraining voltage based on default parameters obtained from the relay settings. The default value for the restraining voltage is 50 which is used for this thesis.

5.2.2 Testing Procedure

Tests were conducted in EMTPWorks 4.2 using a step size of $1 \mu s$ which corresponds to the sampling rate of 1MHz. The total duration of each simulation was set to 200 ms. This simulation duration and sampling rate were necessary to utilize the methods described in [9], which are also applicable to testing the TDRs used in the study. The methods described allow for the voltage and currents of the digital PTs and CTs in the model system to be extracted as a comtrade file which may then be converted into a “.ply” file utilizing SEL’s Playback File Converter software. The SEL-T401L has a

playback feature that utilizes “.ply” files to provide relay results based on provided CT and PT values. This is what allowed for comparisons of the digital model and the SEL-T401L.

5.2.3 Adding Custom Outputs to the LCD Screen of the T401L

By default, the SEL-T401L does not display the status of the TD32 element on the front panel LED indicators. However, the relay does allow the user to define custom outputs to the relay’s LCD screen based on user-defined logic. This feature was utilized to display the status of the TD32 on the LCD screen. This can be done by first creating a timer, as shown in **Figure 5-3**, in the SEL-T401L settings file which will track the status of a defined signal for a set interval of time. Since the TD32 typically reacts in approximately 4 ms, the time interval was designated to be between 0 and 15 ms to ensure the signal was captured.

Timers

Timer 01

T01IN Timer 01 Input SELogic Equation
Z4 OR 67G2 # EXTENDED REACH POTT KEY

T01PU Timer 01 Pickup (milliseconds) 133.3 Range = 0.0 to 60000.0

T01DO Timer 01 Dropout (milliseconds) 0.0 Range = 0.0 to 60000.0

Timer 02

T02IN Timer 02 Input SELogic Equation
TD32F

T02PU Timer 02 Pickup (milliseconds) 0.0 Range = 0.0 to 60000.0

T02DO Timer 02 Dropout (milliseconds) 15.0 Range = 0.0 to 60000.0

Figure 5-3: Setting a timer in the SEL-T401L settings.

Once a timer has been set, a latch can be designated to it to track the status of the signals in question. Once a latch has gone high, it will remain high until the reset conditions that have been defined for the latch have been met. This process is

demonstrated in **Figure 5-4**, where the latch is set to timer 2 (shown in **Figure 5-3**), and its reset condition is the target reset button found on the front panel of the SEL-T401L. This latch needs to be reset between each test performed on the SEL-T401L to ensure that the LCD is displaying the most recent results.

Latches

Latch 01

LT01S Latch 01 Set SELogic Equation
T02 ...

LT01R Latch 01 Reset SELogic Equation
TRGTR OR F_TRIG 3PO ...

Latch 02

LT02S Latch 02 Set SELogic Equation
T09 ...

LT02R Latch 02 Reset SELogic Equation
TRGTR OR F_TRIG 3PO ...

Figure 5-4: Setting up a latch in the SEL-T401L settings.

Once a latch has been designated, then a custom LCD output may finally be defined. This process is outlined in **Figure 5-5** where the TD32F will be displayed on the front panel LCD if latch one goes high.

Display Points and Aliases

DP01 Display Point 1
3PO, BREAKER, OPEN, CLOSED ...

DP02 Display Point 2
LT01,, TD32F ...

DP03 Display Point 3
LT02,, TD32R ...

Figure 5-5: Setting up a custom LCD output for the SEL-T401L.

5.3 Test Outline

This section contains a description of the tests performed in the model system and their intended purposes.

5.3.1 Test 1: Impacts of Fault Inception Angle

Faults were introduced at 60% of the protected line with respect to the Remote relay in the model system. The fault inception angle was varied from 90° to -90° in two degree increments to determine the impact on the performance of the TD32 element. It is not anticipated that the fault inception angle will have a major impact on the TD32's performance as it is an incremental quantity-based element. This test was conducted to compare performance between the SEL-T401L relay and the modeled element and to verify the extent of the impact that fault inception angle has on model performance.

5.3.2 Test 2: Impacts of fault impedance

Faults were introduced at 60% of the protected line with respect to the Remote relay in the model system (the same location as test 1). Fault impedance ranging from 25Ω to 1000Ω was introduced for each fault type to determine the impact of fault impedance on the modeled element. Results obtained in test 1 were used as a baseline for performance. It was expected that fault impedance would have a negative impact on the TD32 element's performance and that this negative impact would grow with respect to increasing fault impedance.

5.3.3 Test 3: Transition Point Along a Parallel Path

Faults were introduced in 10% increments along the parallel path of the protected line to observe the operating behavior of the modeled element. It was anticipated that there would be a transition point along the parallel path where either relay would switch

fault direction assertions. That is, if the relay initially determined the fault to be a forward fault, there would be a point on the parallel path after which the relay would assert a reverse fault.

CHAPTER 6

TEST RESULTS

6.1 Test 1 – Impacts of Fault Inception Angle

Fault inception angle refers to the angle of the voltage phasor at the time a fault occurs. This is illustrated in **Figure 6-1**. The fault corresponding to point P_1 is said to have a fault inception angle of 90° with respect to the voltage waveform. With the same reasoning, the point P_2 is said to have a fault inception angle of 0° with respect to the voltage waveform.

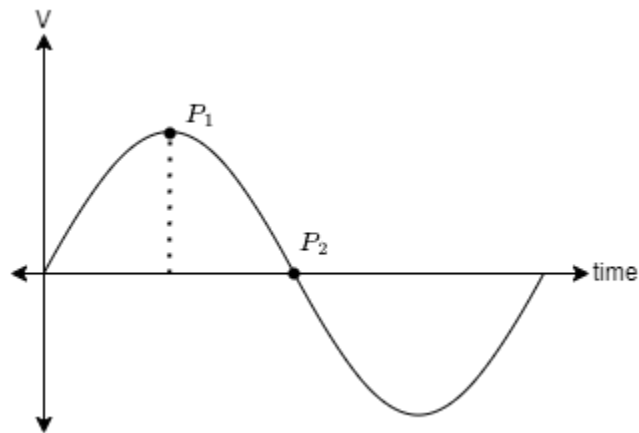


Figure 6-1: Illustration of Fault Inception Angle on a Voltage Waveform.

TDRs typically contain both incremental quantity (IQ) based elements and traveling wave (TW) based elements. TW-based elements utilize the surges and waves that are generated by faults [10]. However, the magnitude of the traveling wave that is

launched by a fault is itself dependent on the fault inception angle (point on the wave in which the fault occurs). Consequently, for faults occurring near the zero-crossing (around P_2 in **Figure 6-1**), the magnitude of the induced traveling wave will be very small, and will only diminish as the wave travels along the line [10]. This will degrade the performance of TW-based elements for faults around the zero crossing region.

IQ-based elements in theory are generally immune to the impacts of fault inception angle. Fundamentally, IQ-based elements are comparing one-cycle old data to the present instantaneous value, meaning that there will be some difference between the two regardless of fault inception angle during fault-induced transients. Albeit, for faults near the zero crossing the change will take a longer time to manifest causing a delay in the operating time. This test is intended to confirm the independence of the TD32's performance from the fault inception angle.

The results collected for A-G faults are shown in **Table 6-1** and **Table 6-2**, B-C faults in **Table 6-3** and **Table 6-4**, and 3PG faults in **Table 6-5** and **Table 6-6**. Results indicate that the fault inception angle had no impact on the TD32's performance for the cases tested. This is expected as the TD32 is an incremental quantity-based element and should not be severely impacted by the fault inception angle. Additionally, it may be noted that neither the SEL-T401L relay, nor the model, was significantly impacted by fault inception angle, in other words, the model mirrors relay performance in this regard.

Table 6-1: EMTP results for a L-G fault 60% down the protected line from the Remote relay.

Test 1 A-G			
Fault Location (km):	23.7539	Local	Remote
Fault Time (s)	Approximate Angle (deg)	TD32F	TD32F
0.066949	90	TRIP	TRIP
0.067875	70	TRIP	TRIP
0.068801	50	TRIP	TRIP
0.069727	30	TRIP	TRIP
0.070190	20	TRIP	TRIP
0.070282	18	TRIP	TRIP
0.070375	16	TRIP	TRIP
0.070468	14	TRIP	TRIP
0.070560	12	TRIP	TRIP
0.070653	10	TRIP	TRIP
0.070745	8	TRIP	TRIP
0.070838	6	TRIP	TRIP
0.070930	4	TRIP	TRIP
0.071023	2	TRIP	TRIP
0.071116	0	TRIP	TRIP
0.071208	-2	TRIP	TRIP
0.071301	-4	TRIP	TRIP
0.071393	-6	TRIP	TRIP
0.071486	-8	TRIP	TRIP
0.071579	-10	TRIP	TRIP
0.071671	-12	TRIP	TRIP
0.071764	-14	TRIP	TRIP
0.071856	-16	TRIP	TRIP
0.071949	-18	TRIP	TRIP
0.072042	-20	TRIP	TRIP
0.072505	-30	TRIP	TRIP
0.073430	-50	TRIP	TRIP
0.074356	-70	TRIP	TRIP
0.075282	-90	TRIP	TRIP

Table 6-2: SEL-T401L Relay results for a L-G fault 60% down the protected line from the Remote relay.

Test 1 A-G Relay Results			
Fault Location (km):	23.7539	Local	Remote
Fault Time (s)	Approximate Angle (deg)	TD32F	TD32F
0.069495	35	TRIP	TRIP
0.070097	22	TRIP	TRIP
0.070190	20	TRIP	TRIP
0.071949	-18	TRIP	TRIP
0.072042	-20	TRIP	TRIP

Table 6-3: EMTP results for a L-L fault 60% down the protected line from the Remote relay.

Test 1 B-C			
Fault Location (km):	23.7539	Local	Remote
Fault Time (s)	Approximate Angle (deg)	TD32F	TD32F
0.079397	90	TRIP	TRIP
0.080323	70	TRIP	TRIP
0.081249	50	TRIP	TRIP
0.082175	30	TRIP	TRIP
0.082638	20	TRIP	TRIP
0.082730	18	TRIP	TRIP
0.082823	16	TRIP	TRIP
0.082916	14	TRIP	TRIP
0.083008	12	TRIP	TRIP
0.083101	10	TRIP	TRIP
0.083193	8	TRIP	TRIP
0.083286	6	TRIP	TRIP
0.083378	4	TRIP	TRIP
0.083471	2	TRIP	TRIP
0.083564	0	TRIP	TRIP
0.083656	-2	TRIP	TRIP
0.083749	-4	TRIP	TRIP
0.083841	-6	TRIP	TRIP
0.083934	-8	TRIP	TRIP
0.084027	-10	TRIP	TRIP
0.084119	-12	TRIP	TRIP
0.084212	-14	TRIP	TRIP
0.084304	-16	TRIP	TRIP
0.084397	-18	TRIP	TRIP
0.084490	-20	TRIP	TRIP
0.084953	-30	TRIP	TRIP
0.085878	-50	TRIP	TRIP
0.086804	-70	TRIP	TRIP
0.087730	-90	TRIP	TRIP

Table 6-4: SEL-T401L Relay results for a L-L fault 60% down the protected line from the Remote relay.

Test 1 B-C Relay Results			
Fault Location (km):	23.7539	Local	Remote
Fault Time (s)	Approximate Angle (deg)	TD32F	TD32F
0.083008	12	TRIP	TRIP
0.083101	10	TRIP	TRIP
0.084027	-10	TRIP	TRIP
0.084119	-12	TRIP	TRIP

Table 6-5: EMTP results for a 3PG fault 60% down the protected line from the Remote relay.

Test 1 3PG			
Fault Location (km):	23.7539	Local	Remote
Fault Time (s)	Approximate Angle (deg)	TD32F	TD32F
0.066949	90	TRIP	TRIP
0.067875	70	TRIP	TRIP
0.068801	50	TRIP	TRIP
0.069727	30	TRIP	TRIP
0.070190	20	TRIP	TRIP
0.070282	18	TRIP	TRIP
0.070375	16	TRIP	TRIP
0.070468	14	TRIP	TRIP
0.070560	12	TRIP	TRIP
0.070653	10	TRIP	TRIP
0.070745	8	TRIP	TRIP
0.070838	6	TRIP	TRIP
0.070930	4	TRIP	TRIP
0.071023	2	TRIP	TRIP
0.071116	0	TRIP	TRIP
0.071208	-2	TRIP	TRIP
0.071301	-4	TRIP	TRIP
0.071393	-6	TRIP	TRIP
0.071486	-8	TRIP	TRIP
0.071579	-10	TRIP	TRIP
0.071671	-12	TRIP	TRIP
0.071764	-14	TRIP	TRIP
0.071856	-16	TRIP	TRIP
0.071949	-18	TRIP	TRIP
0.072042	-20	TRIP	TRIP
0.072505	-30	TRIP	TRIP
0.073430	-50	TRIP	TRIP
0.074356	-70	TRIP	TRIP
0.075282	-90	TRIP	TRIP

Table 6-6: SEL-T401L Relay results for a 3PG fault 60% down the protected line from the Remote relay.

Task 1 3PG Relay Results			
Fault Location (km):	23.7539	Local	Remote
Fault Time (s)	Approximate Angle (deg)	TD32F	TD32F
0.071116	0	TRIP	TRIP

6.2 Test 2 – Impact of Fault Impedance Results and Analysis

Although IQ-based elements should be generally immune to the impacts of fault inception angle, they are nonetheless impacted by fault impedance. This is because the introduction of fault impedance will directly decrease the amount of induced fault current. This in turn will lower the incremental current values. Since the operating principles of IQ elements are based on the fault network discussed in Chapter 2, its performance is linked to these incremental current values.

To observe the impact of fault impedance on the operation of the TD32, fault impedance was introduced to faults at the same location as Test 1. Starting at 25 Ω , fault impedance was increased up to a value of 1000 Ω . The results collected from these tests are shown below.

6.2.1 25 Ω of Fault Impedance

The A-G results collected for the EMTP model are shown in **Table 6-7** and the results collected for the SEL-T401L relay are shown in **Table 6-8**. For A-G faults with 25 Ω of fault impedance, the performance of the TD32 model was impacted on the Remote side whereas the Local side performance was not impacted except for one outlier case at a fault inception angle of 50°. The performance of the SEL-T401L relay was not impacted by the fault impedance for the cases tested.

Table 6-7: EMTP results for a L-G fault 60% down the protected line from the Remote relay with 25Ω of fault impedance.

Test 2 A-G w/ 25 Ohms of Fault Impedance			
Fault Location (km):	23.7539	Local	Remote
Fault Time (s)	Approximate Angle (deg)	TD32F	TD32F
0.066949	90	TRIP	TRIP
0.067875	70	TRIP	NT
0.068801	50	NT	NT
0.069727	30	TRIP	NT
0.070653	10	TRIP	NT
0.070745	8	TRIP	NT
0.070838	6	TRIP	NT
0.070930	4	TRIP	NT
0.071023	2	TRIP	NT
0.071116	0	TRIP	NT
0.071208	-2	TRIP	NT
0.071301	-4	TRIP	NT
0.071393	-6	TRIP	NT
0.071486	-8	TRIP	NT
0.071579	-10	TRIP	NT
0.072505	-30	TRIP	NT
0.073430	-50	TRIP	TRIP
0.074356	-70	TRIP	TRIP
0.075282	-90	TRIP	TRIP

Table 6-8: SEL-T401L Relay results for a L-G fault 60% down the protected line from the Remote relay 25Ω of fault impedance.

Test 2 A-G w/ 25Ω Fault Impedance Relay Results			
Fault Location (km):	23.7539	Local	Remote
Fault Time (s)	Approximate Angle (deg)	TD32F	TD32F
0.07000	24	TRIP	TRIP
0.07010	22	TRIP	TRIP
0.07204	-20	TRIP	TRIP
0.07213	-22	TRIP	TRIP

The results collected for B-C faults for the EMTP model are shown in **Table 6-9** and the results collected for the SEL-T401L relay are shown in **Table 6-10**. The results collected for 3PG faults for the EMTP model are shown in **Table 6-11** and the results collected for the SEL-T401L relay are shown in **Table 6-12**. Neither the performance of

the SEL-T401L relay nor the EMTP model was impacted by the fault impedance introduced for the cases tested.

Table 6-9: EMTP results for a L-L fault 60% down the protected line from the Remote relay 25 Ω of fault impedance.

Test 2 B-C Faults w/ 25 Ohms of Fault Impedance			
Fault Location (km):	23.7539	Local	Remote
Fault Time (s)	Approximate Angle (deg)	TD32	TD32
0.062760	90	TRIP	TRIP
0.063686	70	TRIP	TRIP
0.064612	50	TRIP	TRIP
0.065538	30	TRIP	TRIP
0.066464	10	TRIP	TRIP
0.066556	8	TRIP	TRIP
0.066649	6	TRIP	TRIP
0.066741	4	TRIP	TRIP
0.066834	2	TRIP	TRIP
0.066927	0	TRIP	TRIP
0.067019	-2	TRIP	TRIP
0.067112	-4	TRIP	TRIP
0.067204	-6	TRIP	TRIP

Table 6-10: SEL-T401L Relay results for a L-L fault 60% down the protected line from the Remote relay 25 Ω of fault impedance.

Test 2 B-C Faults w/ 25 Ω Fault Impedance Relay Results			
Fault Location (km):	23.7539	Local	Remote
Fault Time (s)	Approximate Angle (deg)	TD32F	TD32F
0.08301	12	TRIP	TRIP
0.08310	10	TRIP	TRIP
0.08412	-12	TRIP	TRIP
0.08421	-14	TRIP	TRIP

Table 6-11: EMTP results for a 3PG fault 60% down the protected line from the Remote relay 25Ω of fault impedance.

Test 2 3PG Faults w/ 25 Ohms of Fault Impedance			
Fault Location (km):	23.7539	Local	Remote
Fault Time (s)	Approximate Angle (deg)	TD32	TD32
0.066949	90	TRIP	TRIP
0.067874926	70	TRIP	TRIP
0.068800852	50	TRIP	TRIP
0.069726778	30	TRIP	TRIP
0.070652704	10	TRIP	TRIP
0.070745296	8	TRIP	TRIP
0.070837889	6	TRIP	TRIP
0.070930481	4	TRIP	TRIP
0.071023074	2	TRIP	TRIP
0.071115667	0	TRIP	TRIP
0.071208259	-2	TRIP	TRIP
0.071300852	-4	TRIP	TRIP
0.071393444	-6	TRIP	TRIP
0.071486037	-8	TRIP	TRIP
0.07157863	-10	TRIP	TRIP
0.072504556	-30	TRIP	TRIP
0.073430481	-50	TRIP	TRIP
0.074356407	-70	TRIP	TRIP
0.075282333	-90	TRIP	TRIP

Table 6-12: SEL-T401L Relay results for a 3PG fault 60% down the protected line from the Remote relay.

Test 2 3PG w/ 25 Ω Fault Impedance Relay Results			
Fault Location (miles):	23.7539	Local	Remote
Fault Time (s)	Approximate Angle (deg)	TD32F	TD32F
0.07112	0	TRIP	TRIP

6.2.2 50 Ω of Fault Impedance

The A-G results collected for the EMTP model are shown in **Table 6-13** and the results collected for the SEL-T401L relay are shown in **Table 6-14**. For A-G faults with

50 Ω of fault impedance, the model performance was impacted such that it no longer operated. The TD32 of the SEL-T401L relay was not impacted by this level of fault impedance and operated normally for all cases tested.

Table 6-13: EMTP results for a L-G fault 60% down the protected line from the Remote relay with 50 Ω of fault impedance.

Test 2 A-G Faults w/ 50 Ohms of Fault Impedance			
Fault Location (km):	23.7539	Local	Remote
Fault Time (s)	Approximate Angle (deg)	TD32F	TD32F
0.06695	90	NT	NT
0.06787	70	NT	NT
0.06880	50	NT	NT
0.06973	30	NT	NT
0.07056	12	NT	NT
0.07065	10	NT	NT
0.07075	8	NT	NT
0.07084	6	NT	NT
0.07093	4	NT	NT
0.07102	2	NT	NT
0.07112	0	NT	NT
0.07121	-2	NT	NT
0.07130	-4	NT	NT
0.07139	-6	NT	NT
0.07149	-8	NT	NT
0.07158	-10	NT	NT
0.07250	-30	NT	NT
0.07343	-50	NT	NT
0.07436	-70	NT	NT
0.07528	-90	NT	NT

Table 6-14: Relay results for a L-G fault 60% down the protected line from the Remote relay with 50 Ω of fault impedance.

Test 2 A-G Faults w/ 50 Ω Fault Impedance Relay Results			
Fault Location (km):	23.7539	Local	Remote
Fault Time (s)	Approximate Angle (deg)	TD32F	TD32F
0.08236	26	TRIP	TRIP
0.08245	24	TRIP	TRIP
0.08467	-24	TRIP	TRIP
0.08477	-26	TRIP	TRIP

The B-C results collected for the EMTP model are shown in **Table 6-15** and the results collected for the SEL-T401L relay are shown in **Table 6-16**. For B-C faults with 50Ω of fault impedance, neither the model nor the SEL-T401L relay performance was impacted, and both operated correctly for all cases tested.

Table 6-15: EMTP results for a L-L fault 60% down the protected line from the Remote relay with 50Ω of fault impedance.

Test 2 B-C Faults w/ 50Ω of Fault Impedance			
Fault Location (km):	23.7539	Local	Remote
Fault Time (s)	Approximate Angle (deg)	TD32F	TD32F
0.06276	90	TRIP	TRIP
0.06369	70	TRIP	TRIP
0.06461	50	TRIP	TRIP
0.06554	30	TRIP	TRIP
0.06646	10	TRIP	TRIP
0.06656	8	TRIP	TRIP
0.06665	6	TRIP	TRIP
0.06674	4	TRIP	TRIP
0.06683	2	TRIP	TRIP
0.06693	0	TRIP	TRIP
0.06702	-2	TRIP	TRIP
0.06711	-4	TRIP	TRIP
0.06720	-6	TRIP	TRIP
0.06730	-8	TRIP	TRIP
0.06739	-10	TRIP	TRIP
0.06832	-30	TRIP	TRIP
0.06924	-50	TRIP	TRIP
0.07017	-70	TRIP	TRIP
0.07109	-90	TRIP	TRIP

Table 6-16: SEL-T401L Relay results for a L-L fault 60% down the protected line from the Remote relay with 50Ω of fault impedance.

Test 2 B-C Faults w/ 50Ω Fault Impedance Relay Results			
Fault Location (km):	23.7539	Local	Remote
Fault Time (s)	Approximate Angle (deg)	TD32F	TD32F
0.08292	14	TRIP	TRIP
0.08301	12	TRIP	TRIP
0.08412	-12	TRIP	TRIP
0.08421	-14	TRIP	TRIP

For 3PG faults with 50 Ω of fault impedance, the model no longer operated for any of the cases tested. However, the performance on the SEL-T401L relay was not impacted and the testing results can be seen in **Table 6-17**.

Table 6-17: SEL-T401L Relay results for a 3PG fault 60% down the protected line from the Remote relay with 50Ω of fault impedance.

Test 2 3PG Faults w/ 50Ω Fault Impedance Relay Results			
Fault Location (km):	23.7539	Local	Remote
Fault Time (s)	Approximate Angle (deg)	TD32F	TD32F
0.07112	0	F	F

6.2.3 100Ω of Fault Impedance

The results collected from this test are shown in **Table 6-18** through **Table 6-21**. The TD32 of the SEL-T401L relay was not impacted by this level of fault impedance and operated normally for all cases tested. However, for faults with 100Ω of fault impedance, the model performance was impacted such that it no longer operated for the cases tested.

Table 6-18: EMTP results for a L-G fault 60% down the protected line from the Remote relay with 100Ω of fault impedance.

Test 2 A-G Faults w/ 100Ω of Fault Impedance			
Fault Location (km):	23.7539	Local	Remote
Fault Time (s)	Approximate Angle (deg)	TD32F	TD32F
0.06695	90	NT	NT
0.06787	70	NT	NT
0.06880	50	NT	NT
0.06973	30	NT	NT
0.07019	20	NT	NT
0.07029	18	NT	NT
0.07038	16	NT	NT
0.07047	14	NT	NT
0.07056	12	NT	NT
0.07065	10	NT	NT
0.07075	8	NT	NT
0.07084	6	NT	NT
0.07093	4	NT	NT
0.07102	2	NT	NT
0.07112	0	NT	NT
0.07121	-2	NT	NT
0.07130	-4	NT	NT
0.07139	-6	NT	NT
0.07149	-8	NT	NT
0.07158	-10	NT	NT
0.07167	-12	NT	NT
0.07176	-14	NT	NT
0.07186	-16	NT	NT
0.07250	-30	NT	NT
0.07343	-50	NT	NT
0.07436	-70	NT	NT
0.07528	-90	NT	NT

Table 6-19: SEL-T401L Relay results for a L-G fault 60% down the protected line from the Remote relay with 100Ω of fault impedance.

Test 2 A-G Faults w/ 100Ω Fault Impedance Relay Results			
Fault Location (km):	23.7539	Local	Remote
Fault Time (s)	Approximate Angle (deg)	TD32F	TD32F
0.06695	90	TRIP	TRIP
0.06963	32	TRIP	TRIP
0.06973	30	TRIP	TRIP
0.07241	-28	TRIP	TRIP
0.07250	-30	TRIP	TRIP
0.07528	-90	TRIP	TRIP

Table 6-20: SEL-T401L Relay results for a L-L fault 60% down the protected line from the Remote relay with 100Ω of fault impedance.

Task 2 B-C w/ 100Ω Fault Impedance Relay Results			
Fault Location (km):	23.7539	Local	Remote
Fault Time (s)	Approximate Angle (deg)	TD32F	TD32F
0.079397	90	TRIP	TRIP
0.082452556	24	TRIP	TRIP
0.082637741	20	TRIP	TRIP
0.082730333	18	TRIP	TRIP
0.082822926	16	TRIP	TRIP
0.082915519	14	TRIP	TRIP
0.083008111	12	TRIP	TRIP
0.083563667	0	TRIP	TRIP
0.084119222	-12	TRIP	TRIP
0.084211815	-14	TRIP	TRIP
0.084304407	-16	TRIP	TRIP
0.084397	-18	TRIP	TRIP
0.084489593	-20	TRIP	TRIP
0.084582185	-22	TRIP	TRIP
0.084674778	-24	TRIP	TRIP
0.084952556	-30	TRIP	TRIP
0.087730333	-90	TRIP	TRIP

Table 6-21: SEL-T401L Relay results for a 3PG fault 60% down the protected line from the Remote relay with 100Ω of fault impedance.

Task 2 3PG Faults w/ 100Ω Fault Impedance Relay Results			
Fault Location (km):	23.7539	Local	Remote
Fault Time (s)	Approximate Angle (deg)	TD32F	TD32F
0.07112	0	TRIP	TRIP

6.2.4 250 Ω , 500 Ω , and 1000 Ω of Fault Impedance

For fault impedances of 250Ω and above, both the model and the SEL-T401L relay did not function for any of the tested cases. For this reason, the data was not provided in tables for the sake of brevity.

6.2.5 Summary

Model performance differed from the SEL-T401L relay for faults with 25Ω of fault impedance or greater. The performance of the SEL-T401L relay was consistently

correct up until a fault impedance of 500Ω where it stopped functioning altogether. Considering the magnitude of the fault impedances involved, the SEL-T401L relay performance is commendable. Additionally, this test confirms that the introduction of fault impedance causes the relay model to differ from the performance of the SEL-T401L.

6.3 Test 3 – Transition Point Along a Parallel Path Results and Analysis

L-G Faults were introduced along the parallel path of the protected line in 10% increments to observe the impact on the TD32. The parallel path is shown highlighted in the system model in **Figure 6-2**.

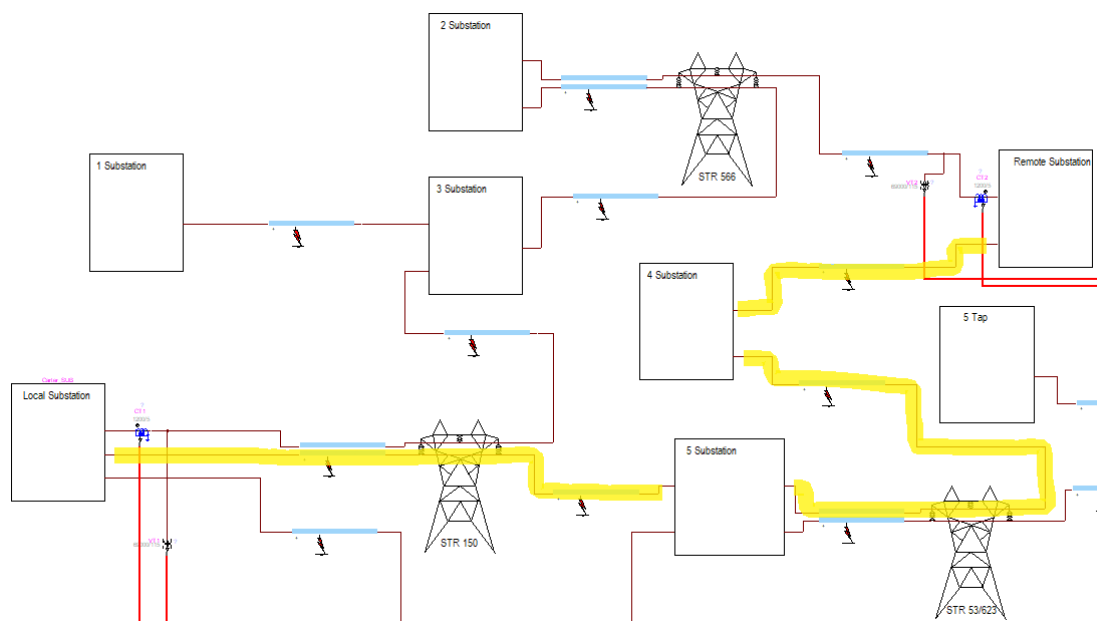


Figure 6-2: Parallel Path of the Protected Line Highlighted in the EMTP System Model.

The data collected for the EMTP model is shown in **Table 6-22** and the data collected for the SEL-T401L relay is shown in **Table 6-23**. The transition point of the EMTP model and the SEL-T401L relay was found to be similar. However, the cases

tested were determined by dividing the parallel line into 10% increments, so the accuracy of the transition point is not great.

Table 6-22: EMTP results for a L-G fault along the parallel path.

Test 3 Faults Along a Parallel Path Results				
Fault Type	Fault Location	Fault inception	Local	Remote
			TD32F	TD32F
AG	4.2488	0.066949	NT	TRIP
			NT	TRIP
			NT	TRIP
AG	8.4976	0.066949	NT	TRIP
			NT	TRIP
			NT	TRIP
AG	12.7464	0.066949	NT	NT
			NT	NT
			NT	NT
AG	16.9952	0.066949	TRIP	NT
			TRIP	NT
			TRIP	NT
AG	21.245	0.066949	TRIP	NT
			TRIP	NT
			TRIP	NT
AG	25.4928	0.066949	TRIP	NT
			TRIP	NT
			TRIP	NT
AG	29.7416	0.066949	TRIP	NT
			TRIP	NT
			TRIP	NT
AG	33.9904	0.066949	TRIP	NT
			TRIP	NT
			TRIP	NT
AG	38.2392	0.066949	TRIP	NT
			TRIP	NT
			TRIP	NT
AG	42.8883	0.066949	TRIP	NT
			TRIP	NT
			TRIP	NT

Table 6-23: SEL-T401L Relay results for a L-G faults along the parallel path.

Test 3 Faults Along a Parallel Path Relay Results				
Fault Type	Fault Location	Fault inception	Local	Remote
			TD32	TD32
AG	4.2488	0.06695	R	F
AG	8.4976	0.06695	R	F
AG	12.7464	0.06695	R	R
AG	16.9952	0.06695	F	R
AG	21.245	0.06695	F	R
AG	25.4928	0.06695	F	R
AG	29.7416	0.06695	F	R
AG	33.9904	0.06695	F	R
AG	38.2392	0.06695	F	R
AG	42.8883	0.06695	F	R

From the results, it can be noted that the transition point exists between 8.4976 km and 12.7464 km for the remote relay, and between 12.7464 km and 16.9952 km for the local relay. Both the digital model and the SEL-T401L relay indicate that the transition point of the TD32 is somewhere within this window.

6.4 Summary

The performance of the model was not impacted by the fault inception angle for the cases tested, and both the model and the SEL-T401L relay operated correctly. This confirms that the fault inception angle would have a negligible impact on the element.

When fault impedance was introduced, the modeled element began to deviate from the performance of the SEL-T401L relay. It can be noted that when 25Ω of fault impedance was introduced, the digital model performance became unreliable. When 50Ω of fault impedance was introduced, the digital model stopped operating entirely. However, this performance was not mirrored in the SEL-T401L relay as it continued to operate up until a fault impedance of 100Ω . Considering the magnitude of the fault impedances tested, the performance characteristics of the SEL-T401L relay are commendable, however, fault impedances over 50Ω are uncommon.

The digital model was also able to predict the transition point on a parallel path of the protected line. However, the accuracy of this test is lacking as it was performed by dividing the parallel line into 10% increments. Nonetheless, it was still able to determine the transition point within 4.2488 km (2.64 miles).

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

7.1 Conclusions

A digital model of the time-domain incremental quantity element was implemented, verified, and analyzed using EMTP in this thesis. Its performance was compared with the TD32 of the SEL-T401L Ultra-High-Speed line relay within a system modeled using data provided by a utility company. An automation script was developed to aid in the testing process. The automation script allowed for hundreds of simulations to be run with data extraction into Microsoft Excel.

The performance of the digital model was found to be comparable to the TD32 of the commercially available relay in the absence of fault impedance. However, once fault impedance was introduced into the system, the model performance began to differ from the SEL-T401L relay. Both the model and the TD32 of the commercially available relay were not impacted by fault inception angle for the cases tested. The model was also able to predict where the transition point in terms of directionality would be along the parallel path of the protected line. Overall, the model presents a good representation of the commercially available relay for bolted faults.

7.2 Future Work

Future work could include determining the cause of the disagreements between the model and the TD32 element of the SEL-T401L relay once fault impedance was introduced. This was the only major point of disagreement between the model and the relay.

In addition, the incremental quantity directional element (TD67) is closely linked to the TD32 and is responsible for tripping, so implementing that element is the logical next step in improving the completeness of the model.

BIBLIOGRAPHY

- [1] J. D. Glover, T. J. Overbye, and M. S. Sarma, “Power system analysis & design,” Cengage Learning, Boston, MA, 2017.
- [2] E. O. Schweitzer, B. Kasztenny, A. Guzman, V. Skendzic, and M. V. Mynam, “Speed of line protection - can we break free of phasor limitations?,” in *2015 68th Annual Conference for Protective Relay Engineers*, College Station, TX, USA, Mar. 2015, pp. 448–461. doi: 10.1109/CPRE.2015.7102184.
- [3] A. N. R. L. Sirisha and S. R. Bhide, “Incremental quantities based relays,” in *2014 International Conference on Power, Automation and Communication (INPAC)*, Amravati, India, Oct. 2014, pp. 27–32. doi: 10.1109/INPAC.2014.6981130.
- [4] F. V. Lopes *et al.*, “Time-Domain Relay Performance Evaluation Considering Brazilian Fault Cases,” presented at the The International Conference on Power Systems Transients (IPST), Perpignan, France, Nov. 2019.
- [5] E. O. Schweitzer, B. Kasztenny, and M. V. Mynam, “Performance of time-domain line protection elements on real-world faults,” in *2016 69th Annual Conference for Protective Relay Engineers (CPRE)*, College Station, TX, USA, Apr. 2016, pp. 1–17. doi: 10.1109/CPRE.2016.7914904.
- [6] B. Kasztenny, A. Guzmán, N. Fischer, M. V. Mynam, and D. Taylor, “Practical Setting Considerations for Protective Relays That Use Incremental Quantities and Traveling Waves,” in *Locating Faults and Protecting Lines at the Speed of Light: Time-Domain Principles Applied*, Schweitzer Engineering Laboratories, Inc, 2018.
- [7] *SEL-T401L Ultra-High-Speed Line Relay Instruction Manual*. Pullman, WA: Schweitzer Engineering Laboratories Inc., 2020.
- [8] T. Russell, “DESIGN AND PERFORMANCE ANALYSIS OF A COMPUTER MODEL OF AN INCREMENTAL QUANTITY DISTANCE ELEMENT IN PROTECTIVE RELAYS,” Louisiana Tech University, 2021.
- [9] T. Russel, P. Bhattarai, T. Field, and M. Quinteros, “Methods of Generating COMTRADE Files Using EMTP for Testing Traveling-Wave-Based Functions on SEL-T400L Relays,” presented at the EMTP User Conference, Held virtually, Oct. 2020. [Online]. Available: <https://www.youtube.com/watch?v=IIEvhBppXg0>

- [10]E. O. Schweitzer, A. Guzman, M. V. Mynam, V. Skendzic, B. Kasztenny, and S. Marx, "Locating faults by the traveling waves they launch," in *2014 67th Annual Conference for Protective Relay Engineers*, College Station, TX, USA, Mar. 2014, pp. 95–110. doi: 10.1109/CPRE.2014.6798997.