Analysis and Design of a Linear Motor for High Speed Applications

by

Pablo Rodriguez

Submitted to the Department of Mechanical Engineering in partial fulfillment of the requirements for the degree of

Master of Science in Mechanical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 1993

© Massachusetts Institute of Technology 1993. All rights reserved.

Author

Department of Mechanical Engineering

May 18, 1993

Certified by

Kamal Youcef-Toumi
Associate Professor
Thesis Supervisor

Accepted by

Dr. Ain A. Sonin
Chairman, Departmental Committee on Graduate Students

ARCHIVES

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

AUG 10 1993
Analysis and Design of a Linear Motor for High Speed Applications 

by 

Pablo Rodriguez

Submitted to the Department of Mechanical Engineering on May 18, 1993, in partial fulfillment of the requirements for the degree of Master of Science in Mechanical Engineering

Abstract

In this thesis, a unified method of modeling various configurations of the Sawyer linear motor is developed. Two separate linear motor structures were choose in the modeling, including a simplified structure commonly depicted in literature as well as the structure employed by currently available linear motors. Through the use of bond graphs, the state equations of both systems are derived and transformed into a form that can easily be implemented in a computer simulation. The systems representing the various motor structures were placed into SIMULINK® , an extension of the numerical software MATLAB® used to simulate dynamic systems. Various observations about the interaction of various energy domains were noted.

As a method of simulation verification, experiments were performed on the Normag 4XY2504-20 linear motor. Values of maximum holding force, motor stiffness, damping ratio, and natural frequency were obtained and compared. It was found that the simulation can predict the force characteristics of the linear motor quite well, while some of the system characteristics were off by a factor of two. In addition, the simulations revealed several key characteristics of the Sawyer linear motor, including the limiting factors affecting force generation and the inherent underdamped behavior the motor.

Thesis Supervisor: Kamal Youcef-Toumi
Title: Associate Professor
Contents

1 Introduction .................................................. 11
   1.1 Preliminaries ........................................ 11
   1.2 Thesis Content ...................................... 12

2 Background on Stepper Motors .......................... 14
   2.1 Variable Reluctance Motor ......................... 14
   2.2 Permanent Magnet Motor ............................ 18
   2.3 Hybrid Motor ........................................ 22
   2.4 Linear Hybrid Stepper Motor ....................... 24

3 Basic Magnetic Components ............................. 29
   3.1 Magnetic Source Elements ......................... 29
       3.1.1 Electromagnets ................................ 30
       3.1.2 Permanent Magnets ............................ 32
   3.2 Magnetic Permeance ................................ 34
   3.3 Force Generation ................................... 39
   3.4 Magnetic System Losses ............................ 43
       3.4.1 Hysteresis Losses ............................... 43
       3.4.2 Eddy Current Losses ......................... 48

4 System Modeling ............................................ 52
   4.1 Modeling Assumptions ............................... 53
   4.2 Simplified Linear Motor ............................ 56
   4.3 A More Practical Linear Motor ..................... 63
5 Motor Simulations

5.1 Simple Motor Simulation Structure ........................................... 72
5.2 Simple Motor Simulation Results ............................................. 76
  5.2.1 Simulation Results to Velocity Command ................................. 77
  5.2.2 Static Force Simulation Results ........................................... 80
  5.2.3 Simulation Results to Step Input ......................................... 86
  5.2.4 Simulation Results Using Micro-steps ................................ 88
5.3 Practical Motor Simulation Structure ........................................ 89
5.4 Practical Motor Simulation Results ........................................... 92
  5.4.1 Simulation Results to Velocity Command ................................. 92
  5.4.2 Simulation Results to Static Force Measurements ..................... 93
  5.4.3 Simulation Results to Step Input ......................................... 99

6 Experimental Results .............................................................. 104

6.1 Response to Velocity Command ................................................. 104
6.2 Response to Step Input .......................................................... 105
6.3 Static Force Measurement ....................................................... 107

7 Conclusions and Recommendations ............................................. 110

7.1 Comparison of Results .......................................................... 110
7.2 Discussion of Simulators ....................................................... 111
7.3 Recommendations for Future Work .......................................... 112

A Detail Drawings ........................................................................... 113

B Calculation of Reluctance Constants .......................................... 116

C Permanent Magnet Parameters ..................................................... 120

D Magnetic Losses .......................................................................... 122
  D.1 Hysteresis Parameters ............................................................ 122
  D.2 Eddy Current Parameters ....................................................... 125
  D.3 Saturation Parameters ........................................................... 126
## List of Figures

2-1 Basic structure of a 3-phase variable reluctance motor. .......... 15
2-2 Commutation sequence of a 3-phase variable reluctance motor. ... 15
2-3 Cross section of an alternate variable reluctance motor. .......... 17
2-4 Typical arrangement for a linear variable reluctance motor. ...... 18
2-5 Cross-sectional view of a 2-phase permanent magnet motor. ...... 19
2-6 Driving sequence of a permanent magnet motor. .................. 19
2-7 Micro-stepping sequence for a permanent magnet motor. .......... 21
2-8 Configuration of a linear permanent magnet motor. .............. 22
2-9 Cross section of a rotary hybrid motor. ......................... 23
2-10 Rotor structure of a rotary hybrid motor. ....................... 23
2-11 Simplified 2-phase linear stepper motor. ....................... 24
2-12 Driving sequence for Sawyer motor. ............................ 26
2-13 Various forcer configurations of a Sawyer motor. ............... 28

3-1 Consideration of an electromagnet. ............................. 30
3-2 A bond graph model of an electromagnet. ....................... 33
3-3 Typical demagnetization curve for several permanent magnets. ... 34
3-4 Typical operating curve for neodymium-iron-boron permanent magnet. 35
3-5 Decomposition of permanent magnet operating curve. ............ 35
3-6 Bond graph model of a permanent magnet. ........................ 35
3-7 Ferromagnetic material properties. ............................... 36
3-8 B-H curve for stack of silicon and cast steel. .................. 38
3-9 Bond graph notation for magnetic permeance. .................... 39
3-10 Tooth structure geometry for hybrid linear motor. 40
3-11 Bond graph notation for permeance due to air gap. 43
3-12 Hysteresis curve of silicon steel. 44
3-13 Analogy of magnetic and mechanical parameters. 45
3-14 Eddy currents formed on lamination due to time varying flux. 48

4-1 Air gap reluctance as a function of pitch position. 54
4-2 Simplified 2-phase linear stepper motor. 56
4-3 Skeleton of bond graph modeling simplified linear motor. 57
4-4 Multiple reluctance terms contributing to force generation. 57
4-5 Assembled and augmented bond graph for a simplified linear motor. 58
4-6 A more practical approach to linear motor. 63
4-7 Structure of magnetic circuit for the practical linear motor. 65
4-8 Augmented bond graph for the practical linear motor. 66

5-1 Skeleton structure of simple motor simulator. 72
5-2 Details within the Reluctance block. 73
5-3 Details within the Flux Generation block. 74
5-4 Details within the Force Generation block. 75
5-5 Details within the Mechanical Domain block. 76
5-6 Simple structure position response to velocity command of 50 [Hz]. 77
5-7 Total force generated by simple structure in response to velocity command of 50 [Hz]. 78
5-8 Force generated by individual poles in response to velocity command of 50 [Hz]. 78
5-9 Flux through air gaps found in response to velocity command of 50 [Hz]. 79
5-10 Current inputs $I_A$ and $I_B$ to generate velocity command of 50 [Hz]. 79
5-11 Total force generated while holding winding currents constant. 81
5-12 Force generated by individual poles while holding winding currents constant. 82
5-13 Magnetomotive force across poles using constant winding currents. 82
5-14 Flux flowing through individual poles using constant winding currents. 83
5-15 Scaled total force generated while holding winding currents constant. 84
5-16 Scaled force generated by individual poles while holding winding currents constant. 85
5-17 Flux flowing through individual poles using constant winding currents and scaling of the flux through each pole. 85
5-18 Time response of position due to a current step input. 86
5-19 Time response of total force generated. 87
5-20 Force generated by individual poles. 87
5-21 Flux flowing through each pole. 88
5-22 Normalized time response to step input. 89
5-23 Position response to micro-stepping sequence. 90
5-24 Total force generated by simple motor structure using micro-stepping sequence. 90
5-25 Skeleton structure of practical motor simulator. 91
5-26 Practical motor structure position response to velocity command of 50 [Hz]. 92
5-27 Total force generated by the practical structure to velocity command of 50 [Hz]. 93
5-28 Force generated by individual poles to velocity command of 50 [Hz]. 94
5-29 Flux through air gaps found in response to velocity command of 50 [Hz]. 95
5-30 Current inputs $I_A$ and $I_B$ to generate velocity command of 50 [Hz]. 96
5-31 Total force generated holding winding currents constant. 96
5-32 Force generated by individual poles holding winding currents constant. 97
5-33 Flux flowing through individual poles using constant winding currents. 98
5-34 Position time response to a current step input. 99
5-35 Time response of total force generated. 100
5-36 Force generated by individual forcers. 101
5-37 Flux flowing through each pole. 102
5-38 Normalized time response to step input. 103
6-1 Experimental setup using the NorMag 4XY2504-20 linear motor.

6-2 Position response of NorMag 4XY2504-20 linear motor to velocity command.

6-3 Normalized step response of NorMag 4XY2504-20 linear motor.

6-4 Experimental setup to measure stiffness of NorMag 4XY2504-20.

6-5 Force versus displacement curve of NorMag 4XY2504-20 motor.

6-6 Scaled force versus displacement curve to determine stiffness of force.

A-1 Detail drawing of electromagnet core used by Normag linear Motor.

A-2 Detail drawing of platen used with Normag linear Motor.

A-3 Detail drawing of permanent magnets.

A-4 Assembled Normag linear motor configuration.

B-1 Detail pole geometry of linear motor.

C-1 Data collected for the rare-earth magnet neodymium-iron-boron.

C-2 $M$ vs. $\phi$ plot use to approximate $R_m$ and $M_c$.

D-1 $B$-$H$ hysteresis curve for Si-steel lamination stack.

D-2 Approximation of the area inside the hysteresis loop.

D-3 Iron volume considered for hysteresis losses.

D-4 Minimal cross sectional area of iron path being considered for saturation.
List of Tables

3.1 Generalized capacitance for various energy domains. 37

7.1 Summary of parameters measured in simulations and from experimentation. 111
Chapter 1

Introduction

1.1 Preliminaries

Like their rotary counterparts, linear motors consist of a "stator", called the "platen", and a "rotor", referred to as the "forcer", that have been unrolled and extended over a desired length. The distinct difference, however, lies in the fact that a rotary motor requires a mechanical transmission to transform the rotary motion into linear. Any mechanical interface has added cost, requires additional space, and in general, introduced complexity while reducing the reliability of the system. A linear motor, on the other hand, already produces linear motion, avoiding the need for any intermediate interface between the motor and its payload.

Although there are various types of linear motor, the most commonly employed linear motor is the Sawyer stepper. In combination with the inherent advantages of a linear motor, the Sawyer linear stepper has the capacity for long, nearly frictionless motion with precise open-loop positioning. In addition, the 2-D linear motor has a cycle path configuration and size that is only limited by platen size. The combination of these qualities has made the Sawyer motor the ideal choice for applications requiring linear motion, precise positioning, high reliability and low maintenance.

Since the invention of the Sawyer linear motor in 1969, the use of linear stepping motor systems have been applied in a limited number. Various factors have contributed to their limited use, including the patent control and subsequent high cost
due to limited availability. Currently, only one US company produces the 2-D linear motors, Northern Magnetics, Inc. (NorMag). With the market control, linear motors have not been cost effective to most other comparable motion technologies.

Various attempts have been made to fabricate 2-D linear motors. Reference [3, 6] have addressed several key issues involved in designing a linear stepping motor. The theoretical development, however, is often complex and obscure. Similar literature on linear motors reveals that the analysis required to design the forcers of linear motor is anything but trivial. The combination of terminology and electro-magnetic concepts makes the entire process unapproachable to someone outside the field.

Instead of following this norm, the entire concept of electro-magnetic fields must be generalized to comprehensible concepts, thereby allowing the designer to visualize how the system components behave in a complex system. A unified approach to system dynamics can be taken to examine how the various system parameters interact with one another to produce useful work.

1.2 Thesis Content

The objective of this thesis is to develop the necessary tools required to analyze the electro-magnetic system involved with linear motors. To achieve this goal, Chapter 2 begins with a qualitative approach on how rotary and linear motors function. Discussion in this chapter is limited to the three basic types of motors: the variable reluctance, permanent magnet, and hybrid stepper motors.

As all complex systems are composed of smaller, simpler components, Chapter 3 is used to explore the components that are combined to make a motor. The components discussed in this chapter include the electromagnet, the permanent magnet, magnetic permeance, as well as common energy dissipative elements in the magnetic domain. The emphasize of the chapter is to draw analogies of the components in the magnetic domain to other, more comprehensible, energy domains. This chapter forms the basis of the unified system dynamics approach taken to model the linear motor.

As an extension, the basic elements detailed in Chapter 3 are combined in the
manner used by the linear motor. The resulting system is presented in Chapter 4. From this arrangement, the physical system is extracted into bond graph notation where the system of equations are readily extracted. As will be observed from in chapter, bond graph notation is particularly useful in systems where interactions between components are not easily understood. In addition, its step by step operations and consistency follow closely that of an electrical network, making the process insightful as well as appealing.

In Chapter 4, the bond graph modeling technique is employed for two separate linear motor configurations: the simplified hybrid linear motor structure and a more practical motor structure. Through these two separate developments, the reader will be shown how the concepts are applied to the simplified version of the Sawyer motor, commonly depicted in the literature because of its simplistic structure. In practice, the fabrication of this simplified motor structure is not practical and thus the alternate structure is also developed, extending the principles employed in the first model.

The system of equations obtained from both the simplified and practical motor structures are then implemented in a computer simulation. Both the structure and the simulation results obtained are presented in Chapter 5. In order to make the user interface to the simulation as friendly as possible, the systems were placed into SIMULINK™, an extension of the numerical software MATLAB™. This software package was chosen to aid the user in concentrating in the system structure versus the programming syntax that commonly appear in dynamic simulations.

As a method of verifying the results obtained from the simulation, Chapter 6 presents some experimental results obtained with the NorMag 4XY2504-20 2-D linear motor. Three basic experiments were performed on the motor, including the response to: velocity command, step input, and static force measurement.

In Chapter 7, the results obtained from the simulation are compared with those obtained experimentally. A discussion on parameter discrepancies is presented, although the results from the simulation were very close to those observed with the NorMag 4XY2504-20 2-D linear motor. Suggestions for future work are also presented in this chapter.
Chapter 2

Background on Stepper Motors

We will begin our understanding of linear motors by going to the roots of all non-contact commutation motors, the rotary stepper. Within this field, there are a number of different motor designs that could easily take a few chapters to discuss, but rather than going into all possible designs, this chapter will focus on the three main types of motors: variable reluctance, permanent magnet, and hybrid stepper. Our understanding of these three basic types of motors will provide the basis of understanding on how a combination of electromagnets and permanent magnet can be arranged to produce useful work.

Sections 1 through 3 are used to describe each of these motors and their typical characteristics and applications. Through these descriptions, the reader is encouraged to consider how the rotary motors can be “unrolled and extended” to the comparable linear case. Section 4 will then make a direct transition to the linear motor of our particular interest: the linear hybrid stepper.

2.1 Variable Reluctance Motor

The birth of steppers can be traced to the introduction of the variable reluctance motor in the early 1900’s. One of the easiest stepper motors to understand, the rotary variable reluctance motor consists of a soft iron rotor and stator structure with electromagnets as illustrated in Figure 2-1.
Figure 2-1: Basic structure of a 3-phase variable reluctance motor.

Figure 2-2: Commutation sequence of a 3-phase variable reluctance motor.

For the sake of simplicity, we will consider the rotation of the rotor as we energize each of the windings in the motor. Namely, when we energize winding A and hold the other windings off, a magnetic flux is generated through the stator poles that attract the rotor poles until they align as in Figure 2-2a. The majority of the magnetic flux generated by the electromagnet of winding A will flow as illustrated in the figure with the dashed lines.

Effectively, when the magnetic flux is generated by the winding of the electromagnet, a magnetic circuit will be formed comparable to an electrical network. As nature seeks to minimize the energy of the magnetic circuit, the motor will tend to
seek the position where the “reluctance” for the flux to flow through is minimized. This will naturally occur when the distance that the flux has to pass through the air is minimized—when the poles of the rotor are aligned with the poles of the stator. Thus, the reluctance of the magnetic circuit will change, hence the name “variable reluctance,” and the energy in the circuit will be minimized. A more detailed description of this circuit will be discussed in the upcoming chapters, but it becomes easier to understand once an analogy with an equivalent electrical circuit has been made.

To advance the rotor one step, we now allow winding $B$ to be energized and hold the other windings off. A new magnetic flux is generated through the corresponding stator poles that attract the rotor poles to alignment as shown Figure 2-2b. Note that the rest position with winding $B$ energized has gone through a rotation of 30 degrees. This rotation corresponds to the “step angle” and is typically only a function of the geometry of the motor stator and rotor teeth. Variations on this tooth arrangement are therefore possible without disturbing the functionality of the motor.

The subsequent rest position, when winding $C$ is energized, is illustrated in Figure 2-2c. Note that the step angle is the same as before, 30 degrees, and our total rotation is now 60 degrees. To continue advancing the motor, the sequence $A-B-C$ can be repeated. Thus, a full revolution would require the sequence $A-B-C-A-B-C$ to be executed four consecutive times.

In our sequence of motion, we have considered energizing windings $A$, $B$, and $C$, in that particular order. For this sequence, $A-B-C-A-B-C$, we have indicated that the motor has gone through a rotation of 360 degrees counter-clockwise. If the sequence was reversed by switching the energizing of windings $B$ and $C$, the new sequence, $A-C-B-A-C-B$, would produce an equal rotation in the clockwise direction. Thus, the minimal number of separate windings required for this motor is three so that both magnitude and direction of travel can be specified.

The variable inductance motor discussed so far was one of the first step motors developed. As noted earlier, the step angle of this structure only provides stable positions every 30 degrees, a rather coarse positioning step motor for applications requiring position control. Another draw back of this particular arrangement is the
torque producing characteristics of the motor. Namely, the maximum holding torque the motor generates corresponds to when the rotor and stator poles are not aligned. Because we only have four stable positions per one motor revolution, the average torque produced per revolution drops. As possible alteration to the motor described so far, the tooth structure can be altered to produce finer position steps and a more continuous torque production. Figure 2-3 illustrates this other implementation of a variable reluctance step motor [11].

Note that Figure 2-3 also illustrates an additional alteration to our original variable reluctance motor. Namely, instead of using the minimum number of windings required, additional windings have been added to provide additional performance enhancement. Because these modifications provide the motor with significantly more detent positions and better performance, they are both incorporated extensively.

The principles of operation detailed for the rotary variable reluctance motor can equally be applied to the linear case. Figure 2-4 illustrates the general structure of the linear variable reluctance motor. The linear motor consists of coils wound on U-shaped cores that form the stator of the motor while the toothed steel rack form the slider. Like the rotary motor, the linear motor produces attractive forces that cause relative motion between the stator and the slider by energizing the separate windings in a particular sequence. This motor also requires a minimum of three phases to determine both magnitude of travel and direction.

Sources [5, 14, 18] detail various applications where linear variable reluctance
motors are used. Smaller pitch sizes than those illustrated in Figure 2-4 as well as additional winding phases are employed for finer position resolution and continuous force generation. Static thrust up to $8 \, N/cm^2$ of pole area is commonly achieved. Dynamically, a typical $1\, kW$ unit can develop a thrust of $125\, N$ at $5\, m/s$.

2.2 Permanent Magnet Motor

Unlike variable reluctance motors, permanent magnet steppers employ a permanent magnet in the rotor. Figure 2-5 illustrates an example of a basic two phase permanent magnet motor. In this stepper, a cylindrical magnet is used in the rotor, with the poles running along the length of the cylinder, opposite poles 180 degrees around the circumference. The stator consists of a series of coils wound around stator poles.

This particular arrangement of electromagnets and permanent magnets provides relative motion as follows. If we energize winding $A$ with a positive current and have winding $B$ off, the stator teeth near the rotor surface will produce a $N$ and $S$ polarity as shown in Figure 2-6a. The rotor magnet will therefore align in such a way as to align the $S$ pole of the rotor with the $N$ pole of the stator and the $N$ pole of the rotor with the $S$ pole of stator. (The tendency of magnets is to attract the opposite pole.)

We can now energize winding $B$ with a positive current and turn off winding $A$ to force the motor to move a full step, 90 degrees rotation, as illustrated in Figure 2-6b. Again, the $N/S$ poles of the rotor/stator, respectively, will align as shown. To continue the rotation, we now energize winding $A$ with a negative current. This will
Figure 2-5: Cross-sectional view of a 2-phase permanent magnet motor.

Figure 2-6: Driving sequence of a permanent magnet motor.
reverse the magnetic polarity of the stator teeth of winding $A$ and force the rotor to the next stable position. Figure 2-6d illustrates the last step of our rotation, where winding $B$ is now energized with a negative current.

The complete sequence, for rotation in the clockwise direction, would be $A-B-A'-B'$, where $A$ signifies winding $A$ energized with a positive current and $A'$ signifies the winding is energized with a negative current. Effectively, by having a permanent magnet in the rotor, the permanent magnet motor structure provides two stable and different rest positions, depending on the direction of the current. Thus, this motor only requires a minimum of two windings to specify both magnitude and direction of travel.

An additional feature of the permanent magnet motor is that the motor comes to rest even if the excitation of the windings ceases. Namely, even if the current in windings $A$ and $B$ are completely turned off, the motor will tend to line the permanent magnet poles with part of the poles of the soft-iron stator. These predetermined positions are referred to as “detent positions”.

The step angle for our simple motor thus far is rather large, 90 degrees. To produce finer positions, modifications to tooth structure of the stator can be made similar to those made to the variable reluctance motor. The rotor, in turn, is typically modified by adding multiple permanent magnets around the cylinder of the rotor. In addition, this motor structure can take advantage of a different driving technique than the simple sequence described earlier.

From our earlier motor discussion, recall that the sequence of winding energizing was to have each of the windings energized fully, with either positive or negative current, while the others where held off. This method of driving is called “full stepping”, since the motor can only be held at an integer value of the step angle. The “micro-stepping” technique does not completely turn on and off the windings in the abrupt sense that full stepping does to travel from one pole alignment to the other. Instead, it energizes multiple windings to provide an intermediate set of stable positions. Figure 2-7 shows the sequence of micro-stepping our simple permanent magnet motor through one full step. For simplicity, we examine the micro-stepping of the
permanent magnet with only "half-steps".

In the micro-stepping sequence illustrated in Figure 2-7, a full step actually occurs with two intermediate micro-steps. As before, we begin the process of taking a step by energizing winding A with positive current and having winding B off. To advance one half step, both windings A and B are both energized with a positive current. The rotor magnet will then align to a position that corresponds to half the step size, 45 degrees, as illustrated in Figure 2-7b. We complete our full step by turning winding A off completely and letting the motor move to align with the poles of winding B. See Figure 2-7c.

With the availability of semiconductor motor drivers, a full step can typically be composed of 128 micro-steps. (For a complete driving sequence of 8 micro-steps, refer to [2, 9, 11].) It is also possible to drive the motor completely sinusoidally by driving each of the windings with a sinusoidal input, each winding having a separate phase angle depending on tooth configuration. This technique is commonly employed by AC motors as well as a variety of other motors.

There are two major drawbacks of using permanent magnets to produce relative motion. First, the magnets are costly compared to the simple soft-iron rotor of the variable reluctance motor. And secondly, the maximum flux density level is limited by the level of magnetic remanence of the magnet. To increase the flux, the current trend is to move away from ferrite magnets and into rare-earth magnets, adding to the cost of the permanent magnet stepper motor.
The configuration of a linear permanent magnet motor is fairly similar to that of the rotary one. Figure 2-8 illustrates the basic configuration of a permanent magnet linear motor. The windings are unrolled to form a track and the permanent magnets are mounted on the bottom face of the slider. Reference [14] gives a detailed description of linear stepper motors.

2.3 Hybrid Motor

The hybrid stepper motor combines some features of the small angle variable reluctance motor and permanent magnet motor described thus far. A cross sectional view of the hybrid motor is shown in Figure 2-9.

The rotor of the hybrid step motor is constructed from a cylindrically shaped permanent magnet that is magnetized along the axis of the rotor. The poles of the magnet are then covered by soft iron cups that conduct the flux of a permanent magnet to the air gap between rotor and stator. These cups are magnetically isolated, with the only contact being made through the permanent magnet. In addition, the teeth on the two cups are offset with respect to each other by 1/2 the pitch size. Refer to Figure 2-10 [9].

The stator core structure is very similar to that of the variable reluctance motor shown in Figure 2-3, with the exception of the winding connections being different. In the variable reluctance motor, only one of the two coils of one phase is wound on the pole, while in the four-phase hybrid motor, coils of two different phases are wound
on the same pole as shown in the following figure.

In this type of motor, torque is created by the interaction of two types of magnetic fields in air gaps of the rotor/stator toothed structure. The magnetic fields generated by the windings alone generate a torque similar to the variable reluctance motor. In addition, the fields generated by the permanent magnet are superimposed or cancel those of the electromagnets. For now, this simple explanation will do to explain how a hybrid motor generates a torque. The exact driving sequence and details on how these fields interact will be discussed in the following section for the linear case. References [2, 9, 11] expand more on rotary hybrid motors.

The most popular hybrid stepper is the four-phase motor with a step angle of 1.8 degrees. Drive electronics similar to those described for permanent magnet motors are
also employed to micro-step the motors, providing accuracies of up to 50 arc-seconds. In applications requiring larger torque than that produced by the single stack motor illustrated in Figure 2-9, multi-stack, or cascade type, hybrid motors can be employed. This arrangement is a mere duplication of that illustrated in Figure 2-9, only with additional permanent magnets and toothed cups along the axis of the motor.

2.4 Linear Hybrid Stepper Motor

The linear hybrid permanent magnet motor, like its rotary counterpart, generates mechanical force by the interaction of magnetic fields of permanent magnets and electromagnets. Also referred to as the Sawyer linear motor, this stepper has the capability of continuous linear motion without the wear of any mechanical transmission required to transfer rotary motion into linear. Employed with an air bearing, most hybrid stepper motors enjoy non-contact force generation, adding significant reliability to most systems. With the added benefit of high position accuracy without any type of sensory feedback controller, the Sawyer motor is in many ways the ideal motor for automated systems where precise positioning and long life are required.

A simplified structure of the stepper is presented in Figure 2-11. The basic components of the motor include the “forcer” or “slider” which moves about a linear
toothed track (or table, in the case of 2-D linear motor) called the “platen”. The
forcer consists of a minimum of two electromagnets separated by a permanent magnet
as illustrated in the figure. The electromagnets are typically fabricated of laminated
C-cores, where various configurations have been utilized for ease of manufacturing.
The platen is fabricated, of any length or size, from high permeability ferromagnetic
material. Ideally, the platen should also be fabricated from laminations to reduce
magnetic losses, but current manufacturing techniques make it prohibitive.

Consider the steps required to move the motor a one pitch displacement as il-
lustrated in Figure 2-12. Although the motor will experience motions in both the
horizontal and vertical directions, our immediate discussion will only consider trans-
lational motions in the horizontal—the direction in which the motor is designed to
move. We begin by having winding A fully excited and winding B turned off as
illustrated in Figure 2-12a.

In this configuration, the magnetic flux of the left core concentrates into the left
tooth, pole 1, bringing the flux density in this tooth to a maximum, while that in
the other tooth is reduced to a negligible value. On the right core, an equal amount
of flux flows through poles 3 and 4 and each pole responds to this flux by trying to
line the pole faces with the platen. Namely, pole 3 will generate a force that will
attempt to move the forcer to the left, while pole 4 will generate a force to move the
motor right. As we shall see in the theoretical sections, this force is dependent on
the flux flowing through the pole. In this case, poles 3 and 4 each carry an equal
amount of flux, and thus generate an equal amount of force. The resultant force,
adding the force term for each of the poles, is canceled and thus the motor is in a
stable configuration.

As we begin to decrease the current in winding A and increase that in winding B,
the electromagnets will commutate the flux so that the flux density running through
pole 4 will increase while that in pole 3 will decrease. As this begins to occur, the
force generated by pole 4 will begin to dominate, producing an acceleration to the
right. By de-energizing winding A and energizing B, the forcer will be driven to
the right a quarter tooth pitch to bring pole 4 in full alignment with the platen as
Figure 2-12: Driving sequence for Sawyer motor.
illustrated in Figure 2-12b.

The sequential steps of the hybrid motor follow similar logic. In Figure 2-12c, the negative current applied to winding A prevents any of the flux from the permanent magnet to flow through the winding, forcing the flux to flow through pole 2. This concentration of flux brings the slider to another stable position at half a tooth pitch. Likewise, Figure 2-12d illustrates how winding B prevents any flux from flowing through pole 4, forcing the slider to move another quarter of a pitch.

The sequence of current excitations described so far can now be repeated, each time the slider assuming a series of new stable positions along the platen. The direction of travel can be reversed by, instead of energizing winding B with a positive current as in Figure 2-12b, we begin the sequence with Figure 2-12d. Similar effects of motion opposite direction displacements can be accomplished by changing the pole spacing between two adjacent poles by half the pitch length. For example, poles 3 and 4 could be swapped to produce displacements to the left when subjected to the driving sequence depicted in Figure 2-12.

In addition, the full current excitation switchings depicted in Figure 2-12 can be replaced by two sinusoidal waveforms that are 90 degrees out of sync. Most linear Sawyer motors do not run with analog sinusoidal waveforms, but rather by micro-stepped sinusoidal waveforms described earlier. By driving the currents in the two electromagnets in this arrangement, the slider assumes a series of new positions along the platen that effectively produce continuous linear motion.

It should also be mentioned that the core configuration depicted in Figure 2-11 are not a unique arrangement. Figure 2-13 depict several other winding and electromagnet arrangements that could equally produce the linear motion illustrated in Figure 2-12. Additional configurations are also available in [9]. The choice on which of these arrangements is most suitable for a Sawyer motor is often left to the designer, who takes the responsibility of considering manufacturability, system losses, etc.
Figure 2-13: Various forcer configurations of a Sawyer motor.
Chapter 3

Basic Magnetic Components

As discussed earlier, rotary and linear stepper motors can be broken down into three basic categories: variable reluctance, permanent magnet, and hybrid. The force generation within any of these three categories is similar for both the linear and rotary case but varies from category to category. To fully understand how such a simple arrangement of magnetic components can produce useful work, it is important to start with the fundamental elements of a magnetic circuit. This chapter is dedicated to describing these elements as well as their dynamic behavior.

Section 1 begins with a brief introduction of magnets; both the electromagnet and permanent magnet characteristics are discussed. This section also serves as the introduction for terminology used throughout the rest of this chapter and more importantly, for Chapter 4. Section 2 introduces the concept of magnetic permeance, the key element involved in the generation of useful work from a magnetic circuit. This section serves as an introduction to Section 3 where the actual mechanism for force generation is discussed. Finally, we close this chapter by exploring dissipative elements in the magnetic domain, including hysteresis and eddy current losses.

3.1 Magnetic Source Elements

In our earlier discussion on variable inductance motors, a qualitative approach was used for explaining the force generated by the Sawyer motor. In order to effectively
design such a motor, a more rigorous approach is required. Namely, an appropriate mathematical model for the complete system must be developed to understand how the system behaves, both statically and dynamically. As with all complex systems, individual elements can be extracted and understood separately before attempting to understand the complete complex system. An appropriate element to start with is the magnet since it is extensively studied and well understood.

There are two basic types of magnets commonly used: the permanent magnet and the electromagnet. The following sections will explore each of these elements and generate some general characteristics which can be utilized to model the hybrid linear motor.

3.1.1 Electromagnets

The electromagnet configuration shown in Figure 3-1 can serve as the start of our understanding of magnetics [8]. By having the core of the electromagnet made from high permeability ferromagnetic material, ie soft iron, we can refer to an introductory book on magnetic physics to find that the device will behave, on the electrical port, as an inductor. We would expect the device to have a linear relationship between current, $i$, and the magnetic flux linkage, $\lambda$. In addition, we could readily find the current and voltage established on the electrical port, but would otherwise be unaware of the effects inside the core.

Whenever current flows through a coil, a magnetic flux is induced around the coil. If we assume each of the $N$ turns of the electromagnet winding links an equal amount
of flux in the core, then the total flux, \( \phi \), can easily be related to the flux linkage by:

\[
\lambda = N\phi
\]

(3.1)

In practice, not all of the physical number of turns of wire will link an equal amount of flux, particularly when multiple layers of wire are wound around the core. This phenomena is commonly referred to as flux “leakage”. Equation (3.1) is still valid, except that \( N \) now represents the effective number of turns rather than the actual number.

We now differentiate Equation (3.1) with respect to time and arrive at the expression:

\[
\dot{\lambda} = N\dot{\phi}
\]

(3.2)

The above expression relates the rate of change of flux linked to the rate of change of total magnetic flux. Applying Faraday's law to the coil, we can find the relationship between the electrical voltage, \( V \), and the rate of change of flux in the core. Namely,

\[
V = \frac{d}{dt}(\text{Flux}_{\text{coil}}) = N\dot{\phi}
\]

(3.3)

In order to drive the flux through the core, a magnetomotive force, \( MMF \), is established across the core much like an electromotive force in an electrical circuit. The analogy consists of considering the magnetic flux lines comparable to the displacement vector of an electric charge. In the electrical system, there is an element, referred to as the electromotive force, in the circuit loop where the electric charge must travel “uphill”, from a lower to higher potential, despite the fact that the electrostatic force is trying to push it from higher to lower potential. In a similar fashion, the magnetic circuit has a driving force that influences the flux to move from lower to higher potential to continue to circulate through the circuit. Reference [8] states that this \( MMF \) is proportional to both the number of turns of the conductor and to
the current flowing in the coil by the relationship:

\[ MMF = Ni \]  \hspace{1cm} (3.4)

Equations (3.3) and (3.4) tie the relationship between the electrical port to the magnetic port in a consistent manner. Namely, the electromagnet can be viewed as a simple 2-port element relating the electrical power variables, voltage and current, to some variables in the magnetic domain by a "transmission ration" of \( N \), the effective number of turns.

References [1, 8] show that using magnetomotive force variable, \( MMF \), as the the effort quantity and the rate of change of flux, \( \dot{\phi} \), as the flow quantity leads to a consistency with bond graph notation. Thus, the power variables in the magnetic domain are \( MMF \), the effort quantity that supplies a driving force, and \( \dot{\phi} \), the flow quantity that represents the rate of change of flux in the circuit. In addition to these magnetic variables, it is often useful to define the variables \textit{magnetic field strength}, \( \vec{H} \), and \textit{magnetic flux density}, \( \vec{B} \). Field strength is simply the magnetomotive force per unit length and the flux density vector corresponds with the amount of flux passing through an area element normal to the flux. Note that \( \vec{B} \) is actually a vector with a direction aligned with the flux lines, whereas the quantity \( B \) is simply the magnitude of \( \vec{B} \). In the discussions that follow, the quantities \( B \) and \( H \) will be used with an implied reference to their direction as vectors.

Having made these definitions for the magnetic domain, it becomes clear that the two port element described as the electromagnet is merely a gyrator that relates an effort variable in one domain to a flow variable in another and vice versa. The electromagnet will therefore be represented in bond graph notation as illustrated in Figure 3.2.

3.1.2 Permanent Magnets

By definition, a permanent magnet is any material that retains its magnetization. We are typically only concerned with materials that can be strongly magnetized,
Figure 3-2: A bond graph model of an electromagnet.

namely ferromagnetic materials. In these materials, there exists a special force that couples the spins of the electrons in adjacent atoms in a crystal, a force created by a quantum-mechanics effect that can be reviewed in [12].

Ferromagnetic materials, in their free state, are made up of small “domains” where the magnetic dipole is aligned. On the macroscopic level, the effective poles from domain to domain do not align, settling into a state of least magnetic energy. However, if the material is immersed in an external magnetic field, all dipoles tend to align along the field. By removing the material from the establishing magnetic field, the domains will retain their magnetic dipole alignment and hence retain a magnetic field capable of generating a magnetic flux. For a more extensive review on magnetic materials, refer to [12, 17].

Typical specifications for permanent magnets are given by demagnetization curves as illustrated in Figure 3-3 [8]. By convention, these curves are plotted as flux density versus field intensity, $B$ vs. $H$, and are commonly referred to as $B$-$H$ curves. Typical linear hybrid motors employ the high-energy-product rare earth magnets, denoted by the heavy curve in Figure 3-3.

A reploting of these curves for the rare-earth magnet neodymium-iron-boron manufactured by Tridus International is provided in Figure 3-4. (Note the correct axis plot, as well as the conversion to SI units.) If we consider the operating curve for normal motor operations, 80 °C, a superposition of two curves can be arrived on as illustrated in Figure 3-5. The first component provides constant magnetomotive force independent of the flux and can therefore be considered as an ideal effort source. This component is analogous to a electromotive source, such as a battery or constant
voltage supply, in the electrical domain.

The second component of a permanent magnet operating curve will be discussed in the following section. For the time being, it will suffice that this term is being modeled as magnetic capacitance, or permeance. The equivalent model for a permanent magnet in bond graph form can therefore be represented as shown in Figure 3-6. A method for obtaining the constants, $R_m$ and $S_{pm}$ is presented in Appendix C.

### 3.2 Magnetic Permeance

A typical relation for a ferromagnetic material core is sketched in Figure 3-7. [8] As noted earlier, the rate of change of flux through this element, $\dot{\phi}$, is considered the generalized flow quantity. If we integrate $\dot{\phi}$ with respect to time, we arrive at $\phi$, the generalized displacement term for the magnetic domain. Thus, Figure 3-7 shows that the effort quantity, $MMF$, across ferromagnetic materials is a function of the generalized displacement term, $\phi$ [8]. As we shall see later, the area under this curve is the potential energy stored by the element, while the coenergy is the area to the
Figure 3-4: Typical operating curve for neodymium-iron-boron permanent magnet.

Figure 3-5: Decomposition of permanent magnet operating curve.

Figure 3-6: Bond graph model of a permanent magnet.
left of the curve.

From a system dynamics text, a dynamic element that stores potential energy is considered a "generalized capacitor". In the electrical domain, the voltage drop across a capacitor is a function of the electric charge across the capacitor plates—an effort quantity is a function of generalized displacement. This has an analogy in the mechanical domain, where the force across a spring is a function of the displacement of the spring. Figure 3-7 indicates that an appropriate model for a ferromagnetic core would therefore be a capacitance.

Several sources, [1, 8, 10], prefer to work with the reluctance of the magnetic circuit instead of the permeance. The relationship between capacitance and reluctance is merely that they are inverses of one another, much like the stiffness term in the mechanical domain is merely the inverse of compliance. As a summary, Table 3.1 presents the constitutive relations for capacitance in the three major domains: electrical, mechanical, and magnetic.

It is noteworthy to point out, however, that by using either magnetic reluctance or permeance, the magnetic core is modeled as a energy storage element. Historically, the reluctance in the magnetic path was sometimes thought of as being analogous to electrical resistance [12], so that the magnetic flux flowing through a typical magnetic circuit was analogous to the current flowing through resistors and the comparable circuit could be solved readily. As pointed out in [8], this resistance analogy breaks down for theory of magnetic energy and coenergy since the electrical resistor dissipates
Table 3.1: Generalized capacitance for various energy domains.

<table>
<thead>
<tr>
<th>Energy Domain</th>
<th>Effort Quantity</th>
<th>Flow Quantity</th>
<th>Generalized Displacement</th>
<th>Capacitance Term</th>
</tr>
</thead>
<tbody>
<tr>
<td>Electrical</td>
<td>Electromotive force ($e$)</td>
<td>Current ($i$)</td>
<td>Charge ($q$)</td>
<td>Capacitance ($C$)</td>
</tr>
<tr>
<td>Magnetic</td>
<td>Magnetomotive force ($M$)</td>
<td>Flux rate ($\dot{\phi}$)</td>
<td>Flux ($\phi$)</td>
<td>Permeance ($P$)</td>
</tr>
<tr>
<td>Mechanical</td>
<td>Force ($F$)</td>
<td>Velocity ($\dot{x}$)</td>
<td>Distance ($x$)</td>
<td>Compliance ($1/k$)</td>
</tr>
</tbody>
</table>

power while a magnetic element exhibiting reluctance in reality stores energy. Hence, the capacitance model presented is more appropriate for lump parameter modeling.

The sketch presented in Figure 3-7 can now be supplemented with actual data. Figure 3-8 presents a $B-H$ curve for a stack of silicon steel laminations [12]. For comparison, the $B-H$ curve for cast steel is also presented. Note that these are magnetization curves for the virgin materials—materials that have not been subjected to magnetic fields. We will consider the demagnetization curves later in this chapter. As illustrated in the figure, $MMF$ is seen to be some function of $\phi$ that can be expressed as:

$$MMF = \psi(\phi)$$

(3.5)

From Figure 3-8, one can see that for an appropriate range of $\phi$, a linear relationship can be found between the $MMF$ and the flux running through the core. This slope, $R$, is defined as the reluctance and is inversely related to the permeance, $P$. Mathematically,

$$MMF = R \cdot \phi = \frac{\phi}{P}$$

(3.6)

In order to understand the permeance term, we again turn to the electrical capacitor for an analogy. It is well known that for two plates of area $A$, separated a
distance $z$, by a dielectric material of value $k$, the capacitance, $C$, can be found by:

$$C = k\varepsilon_0 \frac{A}{z}$$ \hspace{1cm} (3.7)

where $\varepsilon_0$ is called the permittivity constant of free space. Similarly, for magnetic material, the permeability constant is dependent on physical properties of the magnetic material as well as the geometry of the flux path. For magnetic materials, the permeance is related to the cross sectional area of the magnetic flux path, $A$, the length of the path that the flux passes through, $z$, and the relative permeability of the material, $\kappa$, by the relationship:

$$P = \kappa\mu_0 \frac{A}{z}$$ \hspace{1cm} (3.8)

where $\mu_0$ is called the permeability constant of free space.

In bond graph notation, magnetic permeance can be represented as illustrated in Figure 3-9.

As a final comment on magnetic capacitance, the discussion on cores can be ex-
\[
\begin{align*}
\frac{MMF}{\dot{\phi}} & \rightarrow C
\end{align*}
\]

Figure 3-9: Bond graph notation for magnetic permeance.

tended to other magnetic materials that exhibit similar characteristics. Namely, the permeance term in our development of the permanent magnet can now be fully understood. Recall that the operating curve for the permanent magnet was composed of two components, one consisting of a constant magnetomotive force and the other of a curve whose \(MMF\) was dependent on \(\phi\). (Refer to Figure 3-5.) As stated earlier, relationships where magnetomotive force is a function of flux correspond to capacitances. Thus, expression (3.8) is fully applicable for the permanent magnet, with the exception that the relative permeability, \(\kappa\), of our rare-earth magnet is now different from that of silicon steel. In most cases, this value is obtained empirically—directly from the operating curve of the permanent magnet such as Figure 3-4.

Yet another magnetic capacitor to consider would be the one present when magnetic fields flow through air as in Figure 3-10. Here, too, equation (3.8) is valid, and the geometric parameters of area and length are equally applicable to those described for the electrical capacitor. The only distinction is in the relative permeability term, \(\kappa\), which is now that of air. Most references list this value to be very close to that of free space and it is commonly approximated with unity.

### 3.3 Force Generation

Thus far, we have seen how electrical energy is converted to magnetic as well as how this energy can be stored in magnetic fields. We are, however, interested in the mechanical domain, where we can convert this magnetic energy to useful mechanical work. This work is generated solely by the reluctance term associated with the air gap. The relationships are all nonlinear and will require the development based on energy considerations presented in this section.

Before we consider the theory, the introduction of a few geometric variables are
Figure 3-10: Tooth structure geometry for hybrid linear motor.

in order. Figure 3-9 provided a detailed sketch of the air gap across our hybrid stepper motor discussed in Chapter 1. Some geometrical parameters considered in this discussion will be:

\[
\begin{align*}
 h &= \text{air gap width} \\
 z &= \text{tooth engagement} \\
 p &= \text{pitch} \\
 x &= \text{tooth engagement} \\
 z &= \text{air gap height}
\end{align*}
\]

It will be convenient to also talk about the air gap area, \(A_{\text{gap}}\), and the air gap volume, \(V_{\text{gap}}\). These quantities are related as follows:

\[
V_{\text{gap}} = h \cdot x \cdot z = A_{\text{gap}} \cdot z \quad (3.9)
\]

Having taken care of all the preliminaries, we now focus on the tangential force generation from the magnetic domain. As stated earlier, the reluctance of the air gap stores potential energy much like a mechanical spring. The rate of change of energy, or the power \(P\), of our magnetic reluctance is simply the product of the effort and flow quantities in that element. This quantity can be expressed by:

\[
P \equiv \text{effort} \cdot \text{flow} = M \cdot \dot{\phi} \quad (3.10)
\]
The change in energy inside the air gap is simply the integral of power with respect to time. Namely,

\[ \Delta E \equiv \int P \, dt = \int M \, d\phi \quad (3.11) \]

We can apply equations (3.6) from the previous section to the reluctance across the air gap. Namely, the magnetomotive force term is replaces as follows:

\[ \Delta E = \int R_{\text{gap}} \, \phi \, d\phi \quad (3.12) \]

Noting that the reluctance term of the air gap is not a function of the flux flowing through the element, but rather only a function of geometry, we can easily integrate expression (3.12) to arrive at:

\[ \Delta E = \frac{R_{\text{gap}}}{2} \phi^2 \quad (3.13) \]

This expression is true for all capacitance terms. Therefore, the potential energy for a mechanical spring and an electrical capacitor would be expressed as:

\[ \Delta E_{\text{spring}} = \frac{K_{\text{spring}}}{2} x^2 \]
\[ \Delta E_{\text{capacitor}} = \frac{1}{2 \, C_{\text{electrical}}} q^2 \]

We are now ready to arrive at the force generation characteristic of the magnetic capacitor. As detailed in Chapter 2, the reluctance of the air gap will seek to minimize its potential energy. For a given flux flowing through the air gap, expression (3.13) can only be minimized by changing the reluctance of the magnetic field. Namely, the geometry of the air gap, if allowed, will change to minimize potential energy stored by the reluctance. From expression (3.8), available parameters include the area and height of the air gap. We will consider both possibilities individually.

Recall that the area term of the gap involved the air gap width and the tooth engagement. Variations in the width are not allowed, since this would mean the core width changes. Thus, if the reluctance of the air gap is to change, the only flexibility
allowed in the area would come from the tooth engagement, \( x \). As the reluctance of
the air gap is minimized, a tangential force will be produced when the stored magnetic
energy is transformed into mechanical work by a relative displacement of the poles.
This form of energy transfer is called virtual work and is expressed by:

\[
- \Delta E_{\text{gap}} = F_{\text{tangential}} \, dx
\]  

(3.14)

The negative sign in the above expression can be translated into layman terms as: a
force will be generated in such a way as to reduce the overall energy of the system.
For additional information on virtual work, refer to [4, 8].

For our relationship of potential energy in the air gap, the virtual work expression
of equation (3.14) would be rewritten as follows.

\[
F_{\text{tangential}} = -\frac{\partial \Delta E}{\partial x} = -\frac{1}{2} \frac{\partial R}{\partial x} \phi^2
\]  

(3.15)

If we assume that the relationship between tooth engagement and reluctance is in-
versely proportional, as in expression (3.8), we can arrive at the tangential force
generated as:

\[
F_{\text{tangential}} = \frac{z}{2\kappa \mu_0 h x^2} \phi^2
\]  

(3.16)

Using the concept of virtual work, a similar derivation can be arrived for the
attractive force. By allowing displacements in the air gap height, we can find:

\[
F_{\text{attractive}} = -\frac{\partial \Delta E}{\partial z} = -\frac{1}{2} \frac{\partial R}{\partial z} \phi^2
\]  

(3.17)

And again, if we use the linear relationship for the reluctance in the air gap, we can
find the attractive force by:

\[
F_{\text{attractive}} = -\frac{1}{2\kappa \mu_0 h x} \phi^2
\]  

(3.18)

Hence, the bond graph model of the air gap must be modified from that illustrated
in Figure 3-9. Namely, this reluctance term is no longer just a one port element, but
Figure 3-11: Bond graph notation for permeance due to air gap.

rather a two port element linking the magnetic with the mechanical domain. Figure 3-11 shows the appropriate bond graph representation for the air gap reluctance.

For the development of the simulation, we will be concerned with the tangential force generation component, equation (3.16). It will suffice to say that the attractive force component will be compensated for by the air bearing of the motor. As pointed out by [1, 6, 13], the air bearing provides a stiff spring in the vertical direction of the motor, so that the attractive force in combination with the air bearing maintain the air gap height fairly constant. Any dynamics associated with the air gap height will therefore be small and neglected in the simulation.

It should be noted that, although for our application we are primarily concerned with the tangential force generation, the attractive force produced by the air gap is also extensively used for other actuators. [4, 8, 13] provide some of the typical dynamics and applications associated with the attractive force component.

3.4 Magnetic System Losses

The discussion of magnetic circuits has thus ignored losses in the system. As with all energy domains, magnetic circuits experience energy losses in a variety of ways. In this section, we will concentrate the two primary methods, including hysteresis and eddy current losses.

3.4.1 Hysteresis Losses

As we saw earlier in Figure 3-7, the portion of the curve that passes through the origin was obtained by subjecting a piece of silicon to an increasing magnetic field.
As we continue to increase the field intensity to produce larger magnetic fields, there eventually comes a point where the laminations become saturated, corresponding with the material domains having been aligned with respect to the applied $H$ field. Once saturation of the iron path begins to take place, the law of diminishing returns takes over and the iron will not carry additional flux. Refer to Figure 3-12.

As this magnetic field is decreased back to zero, the magnetization curve is not retracted, leaving the core partially magnetized. The induction of a magnetic field in the opposite direction eventually reduces the residual magnetic field free state. Continuing to increase the $H$ field in this direction will now induce the flow of magnetic flux in the opposite direction until we reach the saturation state where the material domains have once again aligned with the applied $H$ field. The $H$ field can now be reversed and the process will be reversed until the saturation point is reached again. In this manner, the iron retains some memory of the magnetic field to which it was exposed. This dependency of previous states is called hysteresis and the path that has been described is referred to as a hysteresis loop.

Work done by [8, 7] have developed an analogy of the saturation effects and hysteretic behavior of magnetic material to load bearing capacities of mechanical com-
Figure 3-13: Analogy of magnetic and mechanical parameters.

components. As stated earlier, the magnetic reluctance is comparable to the mechanical spring, where mechanical force goes like magnetomotive force, $M$, and displacement goes like flux, $\phi$. In the analogy, stress and strain properties of a mechanical sample under load are compared to $H$ and $B$ of a magnetic sample under a magnetomotive force. Figure 3-13 illustrates the analogy [8].

Hysteresis loops in the mechanical elements are due to plastic deformation of the material and limit the force carrying capabilities of the component. The magnetic element behaves differently, where the saturation that is experienced limits the flux carrying capabilities. As illustrated in Figure 3-13, the general shape of the hysteresis loops are quite different, but express one general concept: energy must be expended in tracing a hysteresis loop since losses are incurred in the non-reversible movements. The energy lost per cycle per unit volume corresponds to the area inside the loop.

Detailed modeling of hysteresis in the magnetic domain is quite complicated since the memory effects must be incorporated into the model. Reference [7] discusses the procedure for developing a model. For the purpose of our modeling, however, the hysteresis effect will be modeled by a single dissipative element. This approximation
will be sufficient, considering that one cycle of the hysteresis loop corresponds to the electromagnet being cycled through one sinusoidal wave of current input.

The correlation can further be associated with the velocity of the forcer as follows. From the hysteresis curve for the path of magnetic flux in the iron, we can calculate the area inside the loop to find the energy dissipated $E_p$ per cycle. Note that this energy is per unit volume, and thus requires the entire volume through which the flux penetrates to calculate the total energy loss, $E_{loss}$. Thus,

$$E_{total} = E_{cycle} \cdot \frac{x}{pitch} \quad (3.19)$$

where $E_{cycle}$ is the total energy loss per cycle given by:

$$E_{cycle} = E_p \cdot V_{iron} \quad (3.20)$$

Refer to Appendix D for considerations given to the calculation of the constant term $E_{cycle}$.

The power loss associated with this element is related to the total energy loss by:

$$P_{loss} \equiv \frac{dE_{total}}{dt} = E_{cycle} \cdot \frac{\dot{x}}{pitch} \quad (3.21)$$

If such an element is to be reflected to the mechanical side, then the power loss in the mechanical side should be:

$$P_{loss} = \text{force} \cdot \text{velocity}$$

Define the term $F_{hysteresis}$ to be the force across an equivalent element in the mechanical domain that dissipates the hysteresis energy of expression (3.21). Hence,

$$F_{hysteresis} \cdot \dot{x} = E_{cycle} \cdot \frac{\dot{x}}{pitch} \quad (3.22)$$
Canceling out the velocity terms, we find

\[ F_{\text{hysteresis}} = \frac{E_{\text{cycle}}}{\text{pitch}} \]  

(3.23)

Expression (3.23) implies that hysteresis losses can simply related to a constant force that dissipates energy. An analogy of this phenomena can be drawn out to systems that experience a constant kinetic friction loss, such as a block sliding on a table, commonly derived in Newtonian physics. In such problems, the sum of the forces includes a dissipative term that is constant for all velocities and is given by:

\[ F_{\text{friction}} = \mu_k F_{\text{normal}} \]  

(3.24)

where \( \mu_k \) is the coefficient of kinetic friction and \( F_{\text{normal}} \) is the normal force between the two bodies in relative motion.

Hence, the model developed in the next chapter will include a mechanical friction term—an element that dissipates energy independent of motor velocity and whose coefficient will depend on the area of the hysteresis loop of the laminations shown in Figure 3-12. Parameters associated with this dissipative element are presented in Appendix D.

As a final comment about modeling the hysteresis losses, note that expression (3.23) does not indicate direction of force. As with the mechanical equivalent, the sign indicating the direction of the hysteresis force is obtained from the direction of travel. Thus, if the forcer is moving with positive velocity, \( F_{\text{hysteresis}} \) will be in the negative direction and vice-versa. In addition, this term becomes zero whenever the forcer is not moving, since no losses are incurred when the motor is stationary. In summary, losses due to hysteresis will be modeled by an equivalent force term in the mechanical domain with the properties:

\[
F_{\text{hysteresis}} \begin{cases} 
0 & \text{if } \dot{x} = 0 \\
-\text{sign}(\dot{x}) \cdot \frac{E_{\text{cycle}}}{\text{pitch}} & \text{if } |\dot{x}| > 0
\end{cases}
\]  

(3.25)
Figure 3-14: Eddy currents formed on lamination due to time varying flux.

3.4.2 Eddy Current Losses

The field of eddy current losses is relatively old and well understood discipline in the electrical engineering community. The concern for understanding stems from AC equipment, including transformers, generators, motors, etc., that link their main energy losses to eddy currents. Simply stated, eddy current losses are produced by the current in the magnetic material, such as the electromagnet core or the platen, that are formed when time varying fluxes are induced by electromotive forces.

To illustrate the conditions typical of those that occur in an electromagnet lamination, consider the lamination shown in Figure 3-14. The alternating flux, \( \phi \), induced in the cores as the flux is commutated from one pole to another, permeates the core laminations as illustrated in the figure. Applying Faraday’s law of induction around the boundary \( abcdap \), an electromotive force \( e \) will be induced as expressed in equation (3.3). In our case, the expression can be re-written as:

\[
e = -\frac{d\phi}{dt}\tag{3.26}
\]

Because of the rise of this electromotive force, a current \( i \) will circulate around the boundary. The direction of this current is specified by Lenz’s law, which states that the current will flow in a direction so as to oppose the change in magnetic field that produced it. The effect of such current is to shield the material from the flux, bringing about a smaller flux density near the center of the lamination than at the surface.

The current formed by the time varying flux can be readily calculated knowing electrical resistance, \( r \), of the lamination piece. The relationship is found using Ohm’s
law and is expressed by:

\[ i = \frac{e}{r} = -\frac{1}{r} \dot{\phi} \]  (3.27)

and the instantaneous power loss can be found by:

\[ P_{\text{inst.loss}} = i^2 r = \frac{\dot{\phi}^2}{r} \]  (3.28)

The flux, as a function of time, will be sinusoidal and have a form expressed by:

\[ \phi = \phi_{\text{max}} \cos(\omega t) \]  (3.29)

were \( \omega \) is the current driving frequency. The rate of change of flux will therefore be:

\[ \dot{\phi} = -\omega \cdot \phi_{\text{max}} \sin(\omega t) \]  (3.30)

so that the instantaneous power loss from equation (3.28) can be expressed as:

\[ P_{\text{inst.loss}} = \frac{[\omega \phi_{\text{max}} \sin(\omega t)]^2}{r} \]  (3.31)

In our development, we have assumed that the current produced by the time varying flux, as well as the flux in the magnetic material, was uniform across the lamination. More accurate analysis done by several researchers involved in the field of electrical transformers take into account these effects. The correct instantaneous power loss per unit volume, according to [15], is:

\[ P_{\text{inst.loss}} = \frac{[\tau \omega B_{\text{max}} \sin(\omega t)]^2}{12 \rho} \]  (3.32)

where \( \tau \) is the lamination thickness, \( \rho \) is the volume resistivity of the core and \( B_{\text{max}} \) is the maximum flux density seen by the core. Note that expression (3.32) is based on SI units, where the power will be obtained in Watts provided that all lengths are in meter, \( \rho \) is in \( \Omega \) per cubic meter, and \( B_{\text{max}} \) is expressed in Tesla.

Taking into account the total iron volume \( V_{\text{iron}} \) and averaging the instantaneous
power loss over the period of the frequency, the total power can be found by:

\[ P_{\text{loss}} = \left( \frac{\tau \pi f B_{\text{max}}}{6 \rho} \right)^2 V_{\text{iron}} \]  \hspace{1cm} (3.33)

Noting that the velocity of the slider is proportional to the driving frequency of the current inputs, the expression

\[ f = \frac{\dot{x}}{\text{pitch}} \]  \hspace{1cm} (3.34)

is valid if the motor does not “skip”. The geometrical parameters and constants can be extracted from expression (3.33) and in combination with equation (3.34), producing the expression

\[ P_{\text{loss}} = b \cdot \dot{x}^2 \]  \hspace{1cm} (3.35)

where the constant \( b \) contains the parameters \( t, B_{\text{max}}, \text{pitch} \), and is given by:

\[ b = \left( \frac{\tau \pi B_{\text{max}}}{\text{pitch}} \right)^2 \frac{V_{\text{iron}}}{6 \rho} \]  \hspace{1cm} (3.36)

As derived for the hysteresis losses, an equivalent element can be reflected in the mechanical domain that considered the energy dissipated by eddy currents. As noted earlier, the power loss in the mechanical domain of this element should be:

\[ P_{\text{loss}} = \text{force} \cdot \text{velocity} \]

Define the term \( F_{\text{eddy}} \) to be the force across an equivalent element in the mechanical domain that dissipates the eddy current energy of expression (3.35). Hence,

\[ F_{\text{eddy}} \cdot \dot{x} = b \cdot \dot{x}^2 \]  \hspace{1cm} (3.37)

Canceling out the velocity terms, we find that

\[ F_{\text{eddy}} = b \cdot \dot{x} \]  \hspace{1cm} (3.38)

The derivation considered for eddy currents thus far tend to indicate that the
effect can be modeled by a mechanical damper whose coefficient $b$ depends on several parameters expressed in equation (3.36). Considerations for the calculation of this damping constant are detailed in Appendix D.
Chapter 4

System Modeling

Now that the basic elements of a magnetic circuit are understood, the overall system is considered. In this chapter, the bond graph model for two separate versions of the linear motor will be developed. The first development considers the simplified version of the Sawyer motor most commonly depicted in the literature because of its simplistic structure. In practice, the fabrication of this simplified motor structure is not practical and thus an alternate motor structure design is used. The model for this alternate structure is also developed. In either case, the state equations are derived and altered to an acceptable form that can readily be implemented in a computer simulation.

We open this chapter by introducing the assumptions made in modeling the system. In any dynamic system model, these preliminary assumptions must be made for two reasons. First, the state equations describing the behavior of the system must be obtained analytically. For nonlinear elements, the mathematical expression that describes the nonlinear behavior is typically not easily obtained and even if "complete" nonlinear effects can be expressed mathematically, solving for the exact solution becomes far too complicated, requiring some approximations. Second, by making some simplifications in the model, the general system behavior will be understood without the complication of the minor details that do not influence the general response. Having reproduced this general behavior, the model can then be refined to explore one particular area of interest. This learning approach is more efficient and easier.
Section 2 proceeds by describing the simplified linear motor illustrated in the introduction to the linear motor, Figure 2-11. From the physical system, a bond graph representation of the elements in the magnetic circuit is developed. From the bond graph model, the state equations of the system are then derived. In a similar fashion, Section 3 discusses the bond graph construction as well as the development of the state equations for a more practical motor structure.

4.1 Modeling Assumptions

The motto a model is only as good as its assumptions will be as applicable in the development of the linear motor model as in any system. This section is dedicated to explaining some of the simplifications made in arriving at the state equations of the linear motor, both of the simplified and the more practical version.

For the purpose of simplifying the analysis, the reluctance in the iron path will be assumed to be low, so that only the reluctance in the air gap and permanent magnet will contribute to the dynamics of the system. The model does not take into account any flux leakage in the flux path, thereby assuming that the flux concentrates solely through the air gap volume between the poles of the motor and those of the platen.

The reluctance across the air gap, as the motor is displaced relative to the platen, will be assumed to vary sinusoidally instead of having the inversely proportional relationship expressed in equation (3.8). This will serve two purposes. First, the development of the state equations will be greatly simplified, while still retaining a first order approximation of how the reluctance varies as the slider moves across the platen. The second benefit comes from the numerical calculations required in the simulation, whereby this assumption allows for faster computations as well as more reliable predictions of motor behavior. As will be discussed in the simulation chapter, the reliability of simulation results depends on the numerical stability of this simplification.

In addition, the air bearing that opposes the attractive force will be assumed to be static and thus it supports the slider rigidly away from the platen. References
[1, 6, 13] have shown that the forces involved in the air bearing are actually not static but introduce complex dynamics of their own. We are, however, primarily concerned with the tangential force as the motor moves across the platen and will therefore limit our development to within this area. These simplifications will allow the reluctance of the air gap to be modeled by the sinusoidal wave depicted in Figure 4-1. The reluctance of the air gap will therefore be expressed by:

$$ R = R_{\text{offset}} - R_{\text{amplitude}} \cdot \cos(\alpha x + \beta) $$  \hspace{1cm} (4.1)

where the constants $R_{\text{offset}}$, $R_{\text{amplitude}}$, and $\alpha$ are functions of pole geometry and are constant from pole to pole. These values are derived in Appendix B. The phase angle $\beta$ again depends on the geometry of the forcer poles and, as discussed in the appendix, will vary from pole to pole.

In addition, all losses in the magnetic domain, such as hysteresis and eddy currents, are reflected across to the mechanical domain. As a first order approximation, these losses will be modeled by a linear dashpot and a constant kinetic friction term, where all power losses will be assumed to be dissipated through heat generation. The previous chapter details how these power losses are reflected and Appendix D examines the parameters involved.

Other nonlinear effects, such as core flux saturation, will be neglected. Note, however, that in a more accurate model core saturation is the key component that limits the force produced by the motor and therefore it must be included. With this
consideration in mind, the point of diminishing returns for increasing flux density for the actual system would give a maximum force for a particular winding/core arrangement, whereas the proposed model would indicate that additional force can be generated for the same considerations.

Additionally, all assumptions made for the individual magnetic components in the previous chapter will be incorporated in this discussion. Namely, the magnetomotive force potential across the air gap reluctance will be assumed to have a linear relationship with the flux flowing through the poles, as expressed in equation (3.6). The assumptions made for the permanent magnet will also be used; the permanent magnet will be modeled by the combination of a constant effort source and a linear reluctance element.

In the electrical domain, several simplifications have also been made. First, the motor drive circuit is assumed to be an ideal current source. Although the assumption is reasonable for a range of currents, the driver amplifiers will produce a sourcing current up to a limited voltage. In addition, the sourcing current from switching power amplifiers will also depend on the system impedance and the maximum power rating of the power supply. In the models being developed, these effects will be neglected and the current source will be capable of providing infinite voltage for any driving frequency.

Secondly, the electrical resistance of the windings has been neglected. This term was found to be small compared with other energy dissipative elements, thereby making it possible for it to be simply neglected or be incorporated in the mechanical damping term. And thirdly, the coil windings are assumed to link all the flux in the iron core and to have equal flux densities. As discussed in the electromagnet section of the previous chapter, this assumption is a good first order approximation of the physical system.
4.2 Simplified Linear Motor

We begin the construction of our bond graph model with the physical system representation of the linear motor. Figure 4-2 illustrates the simplified structure of the linear motor described in Chapter 2.

The bond graph development for the magnetic circuit follows closely that of an electrical circuit. Reference [8] describes this approach. Although this method is not unique to the development of bond graphs, its step by step operations and consistency are very appealing, especially for the electro-magnetic domains, where the complex system interactions are not easily grasped. Hence, this process is suggested for any magnetic circuit. The five steps to developing the bond graph representation of the system include:

1. For each node in the circuit with a distinct potential, write a 0-junction.

2. Insert each 1-port circuit element by adjoining it to a 1-junction and inserting the 1-junction between the appropriate pair of 0-junctions.

3. Assign power direction to all bonds, using a through-power convention for the 0-1-0 sections.

4. If the circuit has an explicit ground potential, delete that 0-junction and its bond from the graph. If no explicit ground potential is shown, choose any 0-junction and delete it.

5. Simplify the resulting bond graph by replacing 2-port 0 and 1-junctions that have through powers by single bonds.
Figure 4-3: Skeleton of bond graph modeling simplified linear motor.

Figure 4-4: Multiple reluctance terms contributing to force generation.

Following this convention, we begin by assigning each distinct magnetomotive force potential a zero junction into the basic motor structure and including the magnetic elements in the circuit. Figure 4-3 illustrates the skeleton of our bond graph after the above steps 1 and 2 have been executed. In this figure, note that each of the reluctance terms associated with air gaps contribute to a force generation, where the magnitude of the force can vary from pole to pole, but all have a common velocity. Thus, they are linked together at a 1-junction to the mass of the forceer. The linkage to the mechanical domain is illustrated in Figure 4-4. The damping term associated with system losses is also added to this 1-junction.
Figure 4-5: Assembled and augmented bond graph for a simplified linear motor.

The subsequent steps in simplifying the bond graph are depicted in Figure 4-5. As no explicit ground potentials are available in the circuit, note that the $M_o$ associated with the magnetomotive force potential in the platen was chosen as the arbitrary ground. As will be explored in the following section, this potential is the most appropriate ground reference.

Also note that in bond graph form, the bilateral interaction between two systems must be explicitly expressed using causality strokes. The convention illustrated in Figure 4-5 follows that of reference [8], where the causality stroke next to an element signifies that the element will react to the imposed effort quantity by a flow quantity. Similarly, the causality stroke away from the element signifies that the element will respond with an effort quantity to the imposed flow by some constitutive relationship describing the behavior of the element. For additional information on causality, refer to [8], pp. 53-61.

From the bond graph model of the linear motor, the system of equations describing the system can be derived as follows. For the 0-junctions, all efforts leading into the junction all have the same magnetomotive force since, by definition, a 0-junction
represents a circuit node with a distinct potential. The flows, on the other hand, will sum so that the net flow into the node will equal the net flow out, as expressed in electrical terms by Kirchhoff’s current law. Thus, for the 0-junctions, we have 2 equations, one for each junction, that are:

\[ \dot{\phi}_1 = \dot{\phi}_m + \dot{\phi}_2 \]  
\[ \dot{\phi}_4 = \dot{\phi}_m - \dot{\phi}_3 \]  

(4.2)  
(4.3)

For the time being, we will only consider the 1-junctions associated with the magnetic domain. These junctions represent a magnetic equivalent to Kirchhoff’s voltage law written along a loop in which a flux flows and experiences multiple magnetomotive force drops. Taking the 1-junctions of the circuit, we find the following 3 expressions:

\[ M_2 = NI_A - M_1 \]  
\[ M_m = S_{pm} + M_2 - M_3 \]  
\[ M_3 = M_4 - NI_B \]  

(4.4)  
(4.5)  
(4.6)

Note that \( M_m \) is the magnetomotive force across the reluctance term associated with the permanent magnet while \( S_{pm} \) is the effort source quantity supplied by the permanent magnet.

The above set of expressions represent the compatibility and continuity expressions that relate one element to another. The constitutive relations for the system are also required. These equations are as follows:

\[ R_1 : \ M_1 = R_1(x)\phi_1 \]  
\[ R_2 : \ \phi_2 = \frac{M_2}{R_2(x)} \]  
\[ R_3 : \ \phi_3 = \frac{M_3}{R_3(x)} \]  
\[ R_4 : \ M_4 = R_4(x)\phi_4 \]  
\[ R_m : \ \phi_m = \frac{M_m}{R_m} \]  

(4.7)  
(4.8)  
(4.9)  
(4.10)  
(4.11)
As we examine the number of states associated with the magnetic domain, we consider only the energy storage elements associated with the air gap reluctance. A first glance at Figure 4-5 would indicate that 6 states can be found, one from each reluctance in integral casuality. Namely, \( R_1 \) and \( R_4 \) are presumed to have two states each, while \( R_2 \) and \( R_3 \) are presumed to have one state each. It can be shown, however, that only two unique states exist, those associated with the integral casuality of \( R_1 \) and \( R_4 \) on the magnetic domain. Refer to Appendix E.

As detailed in Appendix E, the two states of the magnetic circuit are \( M_1 \) and \( M_4 \). Note that, in addition, a third state \( \dot{z} \) is found in the inertia mass of the forcer, which will be dealt with shortly. For the time being, we can solve for the two states in the magnetic domain using equations (4.2) through (4.11). (Refer to Appendix F.) These are:

\[
\frac{1}{R_m} \dot{M}_4 + \left( \frac{1}{R_1} + \frac{1}{R_2} + \frac{1}{R_m} \right) \dot{M}_1 = \left( \frac{\dot{R}_1}{R_1^2} + \frac{\dot{R}_2}{R_2^2} \right) M_1 - \frac{N}{R_2} \dot{I}_A \\
\quad + \left( \frac{N}{R_m} + \frac{N}{R_2} \right) \dot{I}_A + \frac{N}{R_m} \dot{I}_B \tag{4.12}
\]

\[
\frac{1}{R_m} \dot{M}_1 + \left( \frac{1}{R_3} + \frac{1}{R_4} + \frac{1}{R_m} \right) \dot{M}_4 = \left( \frac{\dot{R}_3}{R_3^2} + \frac{\dot{R}_4}{R_4^2} \right) M_4 - \frac{N}{R_4} \dot{I}_B \\
\quad + \left( \frac{N}{R_m} + \frac{N}{R_3} \right) \dot{I}_B + \frac{N}{R_m} \dot{I}_A \tag{4.13}
\]

This system of equations can be expressed in matrix form as follows:

\[
A \frac{d\vec{M}}{dt} = B \vec{M} + C \vec{I} + D \frac{d\vec{I}}{dt} \tag{4.14}
\]

where,

\[
A = \begin{bmatrix}
\left( \frac{1}{R_1} + \frac{1}{R_2} + \frac{1}{R_m} \right) & \left( \frac{1}{R_m} \right) \\
\left( \frac{1}{R_m} \right) & \left( \frac{1}{R_3} + \frac{1}{R_4} + \frac{1}{R_m} \right)
\end{bmatrix} \tag{4.15}
\]

\[
B = \begin{bmatrix}
\left( \frac{\dot{R}_1}{R_1^2} + \frac{\dot{R}_2}{R_2^2} \right) & 0 \\
0 & \left( \frac{\dot{R}_3}{R_3^2} + \frac{\dot{R}_4}{R_4^2} \right)
\end{bmatrix} \tag{4.16}
\]
\[
C = \begin{bmatrix}
-N \left( \frac{1}{R_3} \right) & 0 \\
0 & -N \left( \frac{1}{R_3} \right)
\end{bmatrix}
\] (4.17)

\[
D = \begin{bmatrix}
N \left( \frac{1}{R_m} + \frac{1}{R_3} \right) & N \left( \frac{1}{R_m} \right) \\
N \left( \frac{1}{R_m} \right) & N \left( \frac{1}{R_m} + \frac{1}{R_3} \right)
\end{bmatrix}
\] (4.18)

and the vector \( \vec{\tilde{M}} \) contains the states \( M_1 \) and \( M_4 \) and the vector \( \vec{\tilde{I}} \) contains the input currents \( I_A \) and \( I_B \). They can be expressed by:

\[
\vec{\tilde{M}} = \begin{bmatrix}
M_1 \\
M_4
\end{bmatrix}
\] (4.19)

\[
\vec{\tilde{I}} = \begin{bmatrix}
I_A \\
I_B
\end{bmatrix}
\] (4.20)

Note that the matrices \( A, B, C, \) and \( D \) are time varying. In addition, note that the system represented in equation (4.14) does not explicitly show the derivative of the states, as denoted by the \( A \) matrix. For the computer simulation, this system must be reduced to the explicit form by taking the inverse of the matrix \( A \) and solving for the rate of change of the states alone. Namely, we wish to arrive at:

\[
\frac{d\vec{\tilde{M}}}{dt} = A^{-1} \left( B\vec{\tilde{M}} + C\vec{\tilde{I}} + D\frac{d\vec{\tilde{I}}}{dt} \right)
\]

\[
= B'\vec{\tilde{M}} + C'\vec{\tilde{I}} + D'\frac{d\vec{\tilde{I}}}{dt}
\] (4.21)

The matrix \( A^{-1} \) is found to be:

\[
A^{-1} = \frac{1}{\text{det}[A]} \begin{bmatrix}
\left( \frac{1}{R_3} + \frac{1}{R_4} + \frac{1}{R_m} \right) & \left( -\frac{1}{R_m} \right) \\
\left( -\frac{1}{R_m} \right) & \left( \frac{1}{R_1} + \frac{1}{R_2} + \frac{1}{R_m} \right)
\end{bmatrix}
\] (4.22)
The new matrices that solve for the system in standard form are as follow:

\[
B' = \begin{bmatrix}
\left(\frac{1}{R_1} + \frac{1}{R_3} + \frac{1}{R_m}\right) \left(\frac{R_4}{R_3^2} + \frac{R_4}{R_3^4}\right) & -\frac{1}{R_m} \left(\frac{R_4}{R_3^2} + \frac{R_4}{R_3^4}\right) \\
-\frac{1}{R_m} \left(\frac{R_4}{R_3^2} + \frac{R_4}{R_3^4}\right) & \left(\frac{1}{R_1} + \frac{1}{R_3} + \frac{1}{R_m}\right) \left(\frac{R_4}{R_3^2} + \frac{R_4}{R_3^4}\right)
\end{bmatrix}
\]

\[
det [A]
\]

\(4.23\)

\[
C' = \begin{bmatrix}
-N \left(\frac{R_4}{R_3^2}\right) \left(\frac{1}{R_3} + \frac{1}{R_4} + \frac{1}{R_m}\right) & N \frac{1}{R_m} \left(\frac{R_4}{R_3^2}\right) \\
N \frac{1}{R_m} \left(\frac{R_4}{R_3^2}\right) & -N \left(\frac{R_4}{R_3^2}\right) \left(\frac{1}{R_3} + \frac{1}{R_4} + \frac{1}{R_m}\right)
\end{bmatrix}
\]

\[
det [A]
\]

\(4.24\)

\[
D' = \begin{bmatrix}
N \left(\frac{1}{R_3} + \frac{1}{R_4} + \frac{1}{R_m}\right) \left(\frac{1}{R_1} + \frac{1}{R_3} + \frac{1}{R_m}\right) - \frac{N}{R_m} & N \frac{1}{R_1} \frac{1}{R_m} \\
N \frac{1}{R_1} \frac{1}{R_m} & N \left(\frac{1}{R_1} + \frac{1}{R_3} + \frac{1}{R_m}\right) \left(\frac{1}{R_3} + \frac{1}{R_m}\right) - \frac{N}{R_m}
\end{bmatrix}
\]

\[
det [A]
\]

\(4.25\)

where the determinant of the "coupler matrix" is given by:

\[
det [A] = \left(\frac{1}{R_1} + \frac{1}{R_2}\right) \left(\frac{1}{R_3} + \frac{1}{R_4}\right) + \frac{1}{R_m} \left(\frac{1}{R_1} + \frac{1}{R_2} + \frac{1}{R_3} + \frac{1}{R_4}\right)
\]

\(4.26\)

The state equations can now be used to find the flux running through each of the reluctances. The output matrix goes as follows:

\[
\tilde{\phi} = \begin{bmatrix}
\phi_1 \\
\phi_2 \\
\phi_3 \\
\phi_4
\end{bmatrix} = \begin{bmatrix}
\frac{1}{R_1} & 0 \\
-\frac{1}{R_2} & 0 \\
0 & \frac{1}{R_3} \\
0 & \frac{1}{R_4}
\end{bmatrix} \tilde{M} + \begin{bmatrix}
0 & 0 \\
0 & N \frac{1}{R_3} \\
0 & 0 \\
0 & 0
\end{bmatrix} \tilde{I}
\]

\(4.27\)

We can easily verify that the equations represented by expression (4.21) are independent by using the determinant of \(B'\). The determinant is found to be:

\[
det [B'] = \frac{\left(\frac{1}{R_1} + \frac{1}{R_3}\right) \left(\frac{R_4}{R_3^2} + \frac{R_4}{R_3^4}\right) \left(\frac{1}{R_1} + \frac{1}{R_3}\right) \left(\frac{R_4}{R_3^2} + \frac{R_4}{R_3^4}\right)}{det [A]}
\]

\(4.28\)

which is not zero unless the reluctance is not moving, i.e. the motor is stationary.

Up until now, we have only given consideration to the magnetic domain dynamics.

62
We now consider those of the mechanical domain. Recall from the bond graph model of the simplified motor that the flux running through the air gaps produced a force at each pole. From equation (3.15), the tangential force produced by the air gap reluctance was found to be:

$$ F_i = -\frac{1}{2} \frac{\partial R_i}{\partial x} \phi_i^2 $$

(4.29)

for $i = 1, 2, 3, 4$, corresponding with each of the four poles.

The sum of the forces acting on the mass will thus produce an acceleration. Taking into account the four force terms associated with the air gap reluctances, as well as other terms associated with energy dissipation in the system, the equations of motion for the mass are:

$$ \ddot{x} = \frac{1}{\text{mass}} \left[ -\frac{1}{2} \left( \frac{\partial R_1}{\partial x} \phi_1^2 + \frac{\partial R_2}{\partial x} \phi_2^2 + \frac{\partial R_3}{\partial x} \phi_3^2 + \frac{\partial R_4}{\partial x} \phi_4^2 \right) - b \dot{x} - F_{\text{hysteresis}} \right] $$

(4.30)

### 4.3 A More Practical Linear Motor

Consider the motor construction illustrated in Figure 4-6. As discussed earlier, motors using this structure are currently being manufactured for commercial use, as they are easier to fabricate than those described in the previous section. Though many similarities exist between this configuration and the one discussed earlier, the system of equations describing this motor become considerably larger due to the addition of the second pair of cores.

We begin the development of the state equations by transferring the physical motor
configuration to an equivalent bond graph notation as we did in the previous section. Namely, we begin by writing the 0-junctions where distinct magnetomotive force potentials exist and write any connecting elements through a 1-junction. The skeletal bond graph for the magnetic circuit is illustrated in Figure 4-7.

The next consideration is given to deleting a the ground potential. Note that in this case, the bond graph would indicate that either $M_{0a}$ or $M_{0b}$ can be removed. In actuality, these potentials are the same, as illustrated in Figure 4-6. This is due to the fact that if these two magnetomotive force potentials were distinct, a magnetic flux would be induced in the platen. Our assumptions were, however, that the reluctance of the platen was negligible, so that for a given flux, there can not exist a potential difference. Thus, both $M_{0a}$ and $M_{0b}$ are removed as ground potentials.

Figure 4-7 can be simplified and reduced to the form presented in Figure 4-8. Also included in this figure are the causality strokes required in bond graph notation.

We can now begin writing the equations that describe this magnetic system. For the 0-junctions, we note that:

\[
\frac{v_a}{N} = \dot{\phi}_{2a} + \dot{\phi}_{3a} \tag{4.31}
\]
\[
\dot{\phi}_{2a} = \dot{\phi}_{1a} + \dot{\phi}_{ma} \tag{4.32}
\]
\[
\dot{\phi}_{3a} = \dot{\phi}_{4a} - \dot{\phi}_{ma} \tag{4.33}
\]
\[
\frac{v_b}{N} = \dot{\phi}_{2b} + \dot{\phi}_{3b} \tag{4.34}
\]
\[
\dot{\phi}_{2b} = \dot{\phi}_{1b} + \dot{\phi}_{mb} \tag{4.35}
\]
\[
\dot{\phi}_{3b} = \dot{\phi}_{4b} - \dot{\phi}_{mb} \tag{4.36}
\]

where $v_a$ and $v_b$ are the voltage drops across the windings and are the voltages required of the amplifiers to generate the current input. In general, we will not be interested in these quantities and equations (4.31) and (4.34) are not used.

For the 1-junctions, excluding that associated with the mechanical domain, we obtain:

\[
M_{1a} = NI_A - M_{2a} \tag{4.37}
\]
Figure 4.7: Structure of magnetic circuit for the practical linear motor.
Figure 4-8: Augmented bond graph for the practical linear motor.
\[ M_{4a} = NI_A - M_{3a} \quad (4.38) \]
\[ M_{ma} = S_{pm} + M_{1a} - M_{4a} \quad (4.39) \]
\[ M_{1b} = NI_B - M_{2b} \quad (4.40) \]
\[ M_{4b} = NI_B - M_{3b} \quad (4.41) \]
\[ M_{mb} = S_{pm} + M_{1b} - M_{4b} \quad (4.42) \]

The constitutive relationships for the elements in the magnetic circuit are:

\[ R_{1a} : \phi_{1a} = \frac{M_{1a}}{R_{1a}(x)} \quad (4.43) \]
\[ R_{2a} : M_{2a} = R_{2a}(x)\phi_{2a} \quad (4.44) \]
\[ R_{3a} : M_{3a} = R_{3a}(x)\phi_{3a} \quad (4.45) \]
\[ R_{4a} : \phi_{4a} = \frac{M_{4a}}{R_{4a}(x)} \quad (4.46) \]
\[ R_{ma} : \phi_{ma} = \frac{M_{ma}}{R_{ma}} \quad (4.47) \]
\[ R_{1b} : \phi_{1b} = \frac{M_{1b}}{R_{1b}(x)} \quad (4.48) \]
\[ R_{2b} : M_{2b} = R_{2b}(x)\phi_{2b} \quad (4.49) \]
\[ R_{3b} : M_{3b} = R_{3b}(x)\phi_{3b} \quad (4.50) \]
\[ R_{4b} : \phi_{4b} = \frac{M_{4b}}{R_{4b}(x)} \quad (4.51) \]
\[ R_{mb} : \phi_{mb} = \frac{M_{mb}}{R_{mb}} \quad (4.52) \]

Using power variables, we note that only the elements \( R_2 \)'s and \( R_3 \)'s are in integral causality in the magnetic domain. As noted earlier, the integral causality on the mechanical side of these elements does not introduce additional states, as the determinant of the matrix expressed in equation (E.5) is zero for all reluctances. Thus, the states are \( M_{2a}, M_{3a}, M_{2b}, \) and \( M_{3b}. \) In a similar manner to the previous section, these state equations can be put in matrix form as follows:

\[ A \frac{d\vec{M}}{dt} = B\vec{M} + C\vec{I} + D\frac{d\vec{I}}{dt} \quad (4.53) \]
where the matrices $A$, $B$, $C$, and $D$ have the form:

$$A = \begin{bmatrix} a_{11} & a_{12} & 0 & 0 \\ a_{21} & a_{22} & 0 & 0 \\ 0 & 0 & a_{33} & a_{34} \\ 0 & 0 & a_{43} & a_{44} \end{bmatrix}$$  \hfill (4.54)

$$B = \begin{bmatrix} b_{11} & 0 & 0 & 0 \\ 0 & b_{22} & 0 & 0 \\ 0 & 0 & b_{33} & 0 \\ 0 & 0 & 0 & b_{11} \end{bmatrix}$$  \hfill (4.55)

$$C = \begin{bmatrix} c_{11} & 0 \\ c_{21} & 0 \\ 0 & c_{32} \\ 0 & c_{42} \end{bmatrix}$$  \hfill (4.56)

$$D = \begin{bmatrix} d_{11} & 0 \\ d_{21} & 0 \\ 0 & d_{32} \\ 0 & d_{42} \end{bmatrix}$$  \hfill (4.57)

Coefficients of these matrices can be found in Appendix G. The vector $\tilde{M}$ is now composed of the new states,

$$\tilde{M} = \begin{bmatrix} M_{2a} \\ M_{3a} \\ M_{2b} \\ M_{3b} \end{bmatrix}$$  \hfill (4.58)

The state equations can now be re-written in standard form:

$$\frac{d\tilde{M}}{dt} = B'\tilde{M} + C'\bar{I} + D'd\bar{I}$$  \hfill (4.59)
where the new matrices $B'$, $C'$, and $D'$ are represented by:

$$
B' = \begin{bmatrix}
bb_{11} & bb_{12} & 0 & 0 \\
bb_{21} & bb_{22} & 0 & 0 \\
0 & 0 & bb_{33} & bb_{34} \\
0 & 0 & bb_{43} & bb_{44}
\end{bmatrix}
$$

(4.60)

$$
C' = \begin{bmatrix}
cc_{11} & 0 \\
cc_{21} & 0 \\
0 & cc_{32} \\
0 & cc_{42}
\end{bmatrix}
$$

(4.61)

$$
D' = \begin{bmatrix}
dd_{11} & 0 \\
dd_{21} & 0 \\
0 & dd_{32} \\
0 & dd_{42}
\end{bmatrix}
$$

(4.62)

Coefficients for these matrices are also found in Appendix G.

The state equations can now be used to find the flux running through each of the reluctances. The output matrix is as follows:

$$
\vec{\phi} \equiv \begin{bmatrix}
\phi_{1a} \\
\phi_{2a} \\
\phi_{3a} \\
\phi_{4a} \\
\phi_{1b} \\
\phi_{2b} \\
\phi_{3b} \\
\phi_{4b}
\end{bmatrix} = \begin{bmatrix}
-\frac{1}{R_{1a}} & 0 & 0 & 0 \\
\frac{1}{R_{2a}} & 0 & 0 & 0 \\
0 & \frac{1}{R_{3a}} & 0 & 0 \\
0 & -\frac{1}{R_{4a}} & 0 & 0 \\
0 & 0 & -\frac{1}{R_{1b}} & 0 \\
0 & 0 & \frac{1}{R_{2b}} & 0 \\
0 & 0 & 0 & \frac{1}{R_{3b}} \\
0 & 0 & 0 & -\frac{1}{R_{4b}}
\end{bmatrix} \vec{M} + \begin{bmatrix}
N \frac{1}{R_{1a}} & 0 \\
0 & 0 \\
0 & 0 \\
0 & N \frac{1}{R_{4a}} & 0 \\
0 & 0 & N \frac{1}{R_{1b}} \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & N \frac{1}{R_{4b}}
\end{bmatrix} \vec{I}
$$

(4.63)

Having found the flux flowing through the air gaps, the force generated by the motor can now be found. Using equation (4.29), the total force produced by the
reluctances is the sum of the force generated by each pole. Taking the sum of all
the forces acting on the forcer mass, including the damping term, the equations of
motion for the motor can be derived. In a similar manner to the simplified motor,
the differential equation of the forcer is:

\[
\ddot{x} = \frac{1}{\text{mass}} \left[ -\frac{1}{2} \sum_{i=1}^{4} \left( \frac{\partial R_{ia}}{\partial x} \phi_{ia}^2 + \frac{\partial R_{ib}}{\partial x} \phi_{ib}^2 \right) - \kappa \dot{x} - F_{\text{hysteresis}} \right]
\]  

(4.64)

As a final note on the derivation of the state equations of this more practical motor,
it should be noted that the state equations expressed in equation (4.59) exhibit no
coupling between the states of the left forcer poles and those of the right poles.
Namely, we can write the equations of the two separate sets of cores independently
as:

\[
\begin{bmatrix}
\dot{\phi}_{2a} \\
\dot{M}_{3a}
\end{bmatrix} =
\begin{bmatrix}
bb_{11} & bb_{12} \\
bb_{21} & bb_{22}
\end{bmatrix}
\begin{bmatrix}
M_{2a} \\
M_{3a}
\end{bmatrix} +
\begin{bmatrix}
cc_{11} \\
cc_{21}
\end{bmatrix} I_A
\]  

(4.65)

\[
\begin{bmatrix}
\dot{M}_{2b} \\
\dot{M}_{3b}
\end{bmatrix} =
\begin{bmatrix}
bb_{33} & bb_{34} \\
bt_{43} & bb_{44}
\end{bmatrix}
\begin{bmatrix}
M_{2b} \\
M_{3b}
\end{bmatrix} +
\begin{bmatrix}
cc_{32} \\
cc_{43}
\end{bmatrix} I_B
\]  

(4.66)

Note that the output equation, expression (4.63), is still an adequate representa-
tion of the output states, and thus, the subsystems represented by the above set of
matrices is an accurate replacement of expression (4.59). In the subsequent discus-
sion, the model expressing the states of the more practical motor are implemented in
the computer simulation of the motor, allowing the simulation to be easily divided
into subsystems that can be implemented and debugged independently.
Chapter 5

Motor Simulations

The system of equations describing both the simplified and more practical linear motor structures are now ready to be implemented in a computer simulation. In order to make the user interface to the simulation as friendly as possible, the systems were placed into SIMULINK\textsuperscript{TM}, an extension of the numerical software MATLAB\textsuperscript{TM}. This software was chosen for two primary reasons. First, SIMULINK\textsuperscript{TM} uses a block diagram approach to simulate dynamic systems, aiding the user in concentrating on the system structure versus the programming syntax. And secondly, the software provided extensive flexibility in readily implementing several simulation parameters, including the type of numerical integrator to be used. In addition to these key features, additional flexibility is offered in the software to allow modeling of the nonlinear dynamics associated with the linear motor with relative ease.

In this chapter, the dynamic simulator structure and the results obtained from the simulations will be discussed. Section 1 describes the layout used to implement the simplified motor structure. Results obtained from this simulation, as well as parameter verification, are presented in Section 2. Sections 3 and 4 follow a similar pattern to describe the simulation of the more practical motor.
5.1 Simple Motor Simulation Structure

The structure used in simulating the motors was intentionally chosen to aid the user in understanding how the causality of the physical system will flow. Consider the global structure of the simulator presented in Figure 5-1. The skeleton of the structure consists of five major components, including the Input, Reluctance, FluxGeneration, Force Generation, and Mechanical Domain blocks. These major blocks serve to group the functionality of various other smaller components detailed below.

The user of a linear stepper motor designates the input currents denoted by $I_A$ and $I_B$. Although the user is free to choose the value and time history of these inputs, the simulator assumes that currents will be sinusoidal and 90 degrees out of phase, denoted by the cosine and sine associated with the $I_A$ and $I_B$ blocks, respectively. Once the user has specified these inputs, the system response can be observed.

The plant of the linear motor begins with the Reluctance block. Various terms are calculated within this block, as illustrated in Figure 5-2. This block is used to calculate the reluctance terms associated with each air gap, as well as their time derivatives. In addition, the permanent magnet reluctance, assumed to be constant, as well as the time derivatives of the current inputs are calculated. The combination of all these values are multiplexed into a single line and linked to the Flux Generation block.

Perhaps the hardest component to understand, the Flux Generation block is used
Figure 5-2: Details within the *Reluctance* block.
Figure 5-3: Details within the *Flux Generation* block.

to implement the second order system describing the magnetic domain as expressed in equation (4.21). The easiest interpretation of the blocks within this element can be seen by working backwards, from left to right, instead of forwards. As expressed in (4.27), the output vector from the magnetic circuit consists of the flux associated with each reluctance. Hence the output blocks \( i = 1, 2, 3, 4 \) are linked to the flux through the \( R_i \) reluctance term. Refer to Figure 5-3.

Proceeding left, consider the integrator block \( \frac{1}{s} \) denoted \( dM_1 \). Prior to being integrated, the signal coming into this block is the \( \dot{M}_1 \) component of the state matrix in (4.21). As noted in expression (4.21), the \( \dot{M}_1 \) term is associated with the sum of terms involving the current inputs, both the \( \bar{I} \) and \( \frac{d}{dt} \bar{I} \) terms, as well as the instantaneous
values of the states themselves, $M_i$ and $M_4$. Note that in addition, the inverse of the $\text{det} [A]$ is taken into account by simply multiplying after the summer. A similar structure follows for the $\dot{M}_4$ term.

Having solved for the two state variables of the system, the additional output variables $\phi_2$ and $\phi_3$ can be found by the relations stated in expression (4.27). All together, four flux terms are calculated and linked to the Force Generation block.

The Force Generation block, as illustrated in Figure 5-4, is fairly straightforward. Having the $\phi_i$'s for each reluctance, the force generated by each pole is calculated using expression (4.29). It is noteworthy to point out that in addition to the flux values obtained in the Flux Generation block, the force generated by each pole also depends on the calculation of the rate of change of the reluctance with respect to position. This computation the explicit value of $x$, and hence an additional input line. The total force generated by the motor is simply the sum of each reluctance component, as implemented by the summing block.

The Mechanical Domain block, shown in Figure 5-5, includes all the mechanical terms associated with the forcer. The summer block represents the sum of the forces acting on the mass of the forcer, including the force generated by the reluctance, as
well as those associated with energy losses. Dividing the sum by the mass of the forcer, we arrive at the evaluation of expression (4.30). This term is associated with the acceleration of the forcer, from which the first integrator block produces velocity while the second produces the position of the forcer. The output of this block, $x$, is then tied back to any blocks requiring position information, including the Reluctance and Force Generation blocks.

Throughout the block diagram layout, various storage blocks are used to collect particular values of interest. The primary values include:

\[
X \quad \text{position of the motor} \\
F_t \quad \text{total force generated by the motor} \\
F_i \quad \text{force generated by the } i^{th} \text{ pole} \\
flux_i \quad \text{denoting the flux flowing through the } i^{th} \text{ pole}
\]

Similar blocks have also been used to store information about time as well as the input currents.

### 5.2 Simple Motor Simulation Results

In order to verify the validity of the simulation, several tests were performed. Detailed below are conditions and results specified for three elementary simulation evaluations
Figure 5-6: Simple structure position response to velocity command of $50 \text{ Hz}$.

performed. These evaluations include: response to velocity command, static force estimation, and response to step input command. In addition, the motor response to a discretized sinusoidal current input, the commonly practiced “micro-stepping” technique, is also provided.

5.2.1 Simulation Results to Velocity Command

Several tests can be performed with the simulation structure described above, but perhaps the most interesting will be the simulation of the motor moving. In this simulation, a command frequency of $50 \text{ Hz}$ was used to drive the sinusoidal current inputs, corresponding to a forcer velocity of approximately $0.05 \text{ m/sec}$. The results for this simulation are presented in Figures 5-6 through 5-10.

On close examination of this data, there are some interesting observations to note. First, Figure 5-6 illustrates how the motor accelerates in response to the command input in going from $\dot{x} = 0$ to $\dot{x} = 0.05 \text{ m/sec}$. The oscillations observed during the first 40 msec indicates the underdamped characteristics of the forcer, as will be discussed
Figure 5-7: Total force generated by simple structure in response to velocity command of 50 [Hz].

Figure 5-8: Force generated by individual poles in response to velocity command of 50 [Hz].
Figure 5-9: Flux through air gaps found in response to velocity command of 50 [Hz].

Figure 5-10: Current inputs $I_A$ and $I_B$ to generate velocity command of 50 [Hz].
later in the step response of the motor. In essence, the motor generates a relatively large force when it becomes “misaligned” with the magnetic flux flowing through the poles, overshooting the forcer past the instantaneous stable position. A negative force is then created to move the motor in the opposite direction. This process continues as illustrated in Figure 5-7 due to the fact that very little damping occurs.

The second observation to note concerns Figure 5-9. Although the first sinusoidal period of the wave is somewhat distorted due to the acceleration of the forcer, these plots illustrate how the flux is commutated from pole to pole. Initially, when the motor is aligned with Pole1, the majority of the flux concentrates through this pole, while Pole2 receives a minimal amount. The flux flows through the platen and re-enters the forcer via Pole3 and Pole4, each pole receiving approximately half of that flowing through Pole1.

As the forcer advances, Pole1 begins to be misaligned with the platen teeth, while Pole2, on the other hand, begins to align. These relative alignments, in combination with the reduced amount of current running through winding A, allow the flux through Pole1 to decrease as the flux through Pole2 increases. A similar case is illustrated in Pole3 and Pole4. This commutation of flux allows the advancing poles, Pole2 and Pole4 to generate larger forces to accelerate the motor in the desired direction of travel.

In general, the winding current commutates the flux flowing through the poles in such a way as to provide the correct amount of flux to the poles that will advance the motor in the correct direction. This sequence of flux commutation was depicted earlier, in Chapter 2, to qualitatively explain how the motor moves. (Refer to Figure 2-12.) The simulation illustrates this swapping of flux in the precise matter we had assumed it would flow.

5.2.2 Static Force Simulation Results

Thus far, the model has predicted appropriate behavior for the linear motor. The next measure of the simulation’s accuracy involves determining the force generated by the simulation. The simplest case comes from the static force generated by the
Figure 5-11: Total force generated while holding winding currents constant.

forcer. In this simulation, the current through winding A is held at the maximum current allowed, \( I_A = I_{max} \), while winding B was held off. With these given currents, a displacement \( x \) was specified and the force generated by the forcer was observed, along with various other parameters. The results of this simulation are listed in Figures 5-11 through 5-14.

As indicated in Figure 5-11, the theoretical maximum force that this forcer is capable of generating is approximately 23 \([N]\), or \( \sim 5.2\ [lbf] \). Although the value does not appear to be exaggerated, the simulation considers only an individual forcer. If we were to consider the force generated by 8 similar forcers, the net force would be in the order of 184 \([N]\), or \( \sim 41.6\ [lbf] \), where as a comparable motor, such as the NorMag 4XY2504-20, is rated at 26-30 \([lbf] \). The simulation overpredicted the total force generation capability. An observation into the flux running through each pole reveals a possible explanation for the discrepancy.

On closer examination of the data provided by the simulation, consider Figure 5-14. The simulation calculated that the maximum flux flowing through an individual
Figure 5-12: Force generated by individual poles while holding winding currents constant.

Figure 5-13: Magnetomotive force across poles using constant winding currents.
Figure 5-14: Flux flowing through individual poles using constant winding currents. A pole was on the order of $2.7 \times 10^{-4}$ [Weber]. Recall that the simulation had neglected saturation effects in the cores, thereby allowing all the flux to be used in the force generation. A first order approximation of the saturation flux allowed through the poles reveals that this flux is in excess of the saturation flux by a factor of 1.4. Although the discrepancy might seem insignificant, consider the fact that the force generated goes per pole as $\phi^2$.

A more accurate model would consider such saturation effects in more detail. For the time being, the easiest way of overcoming this discrepancy is to include a "saturation" block in the simulation. There are several ways of implementing this block in SIMULAB™, including a built in block where the saturation levels in both the positive and negative direction are specified. It was observed, however, that this block did not function correctly in the simulation, since the poles were predominantly in saturation as illustrated in Figure 5-14.

Instead, a simple scaling function was used to take into account saturation effects.
Figure 5-15: Scaled total force generated while holding winding currents constant.

Namely,

\[
\phi_i^{use} = \frac{\phi_{sat}}{\phi_{max}} \phi_i^{calculated}
\]

was used to scale the flux calculated by the simulation so as not to exceed the saturation level. The parameter \(\phi_{sat}\) was calculated directly from the \(B_{sat}\) level of silicon steel and represents the maximum flux, corresponding to saturation, capable of flowing through the cores. (Refer to Appendix D.) \(\phi_{max}\) corresponds to the maximum flux in the simulation, pertaining to the value of flux flowing through a pole that is aligned with the plten teeth.

The results using this scaling factor are illustrated in Figures 5-15 through 5-18. These results are significantly closer to the estimated values. The force generated is approximately \(13 [N] \sim 2.9 [lb]\), per forcer, or \(\sim 23.2 [lb]\) total for a motor using 8 forceps.
Figure 5-16: Scaled force generated by individual poles while holding winding currents constant.

Figure 5-17: Flux flowing through individual poles using constant winding currents and scaling of the flux through each pole.
5.2.3 Simulation Results to Step Input

Another simulation performed was that of a step input. The simulation created a position step of $\frac{1}{4}$ pitch by having the current inputs vary for winding $A$ from $I_A = I_{\text{max}}$ to $I_A = \frac{\sqrt{2}}{2} \cdot I_{\text{max}}$ and for winding $B$ going from $I_B = 0$ to $I_B = \frac{\sqrt{2}}{2} \cdot I_{\text{max}}$. The results are illustrated in Figure 5-18 through 5-21. Note that, in addition, the scaling factor expressed in equation (5.1) was also implemented.

In order to determine typical parameters that describe a step response, Figure 5-19 was normalized and replotted as shown in Figure 5-22. Key parameters to observe include:

- Maximum Percent Overshoot $= 78.7$ [percent]
- Delay Time $= 1.79$ [msec]
- Rise Time $= 2.51$ [msec]
- Peak Time $= 4.46$ [msec]
Figure 5-19: Time response of total force generated.

Figure 5-20: Force generated by individual poles.
Figure 5.21: Flux flowing through each pole.

from which the characteristics of the system can be deduced as follows:

\[
\begin{align*}
\text{Damping Ratio} & \quad \zeta = 7.6 \times 10^{-2} \\
\text{Damped Natural Frequency} & \quad \omega_d = 70.4 \left[ \frac{\text{rad}}{\text{sec}} \right] \\
\text{Undamped Natural Frequency} & \quad \omega_n = 706 \left[ \frac{\text{rad}}{\text{sec}} \right]
\end{align*}
\]

As the calculations reveal, the system is highly underdamped. Ideally, we would wish our systems to have damping ratios \( \zeta \) between 0.5 and 0.8, since they approach the final value more rapidly than a critically damped or overdamped system. The simulation predicts, however, that the system will oscillate for a considerable amount of time before settling down to the commanded position.

### 5.2.4 Simulation Results Using Micro-steps

As a final test on the simulator for the simple motor structure, the motor was commanded to move at a constant velocity using micro-steps. This simulation is more
Figure 5-22: Normalized time response to step input.

realistic of the open loop control of a linear stepper motor, since the sinusoidal waves used thus far are not continuous.

The results of this simulation are illustrated in Figure 5-23. Note that the following parameters were specified in this simulation include:

- Reference Velocity \( \frac{m}{sec} \) = .01
- No. of Microsteps = 16 \( \frac{micro - step}{pitch} \)

5.3 Practical Motor Simulation Structure

Like the simple motor structure, the simulation for the practical motor was implemented in SIMULINK\textsuperscript{TM} and therefore follows closely to that outlined in the previous section.

The global structure of the simulator is illustrated in Figure 5-25. For the sake
Figure 5-23: Position response to micro-stepping sequence.

Figure 5-24: Total force generated by simple motor structure using micro-stepping sequence.
of brevity, the reader is asked to extend the details of the simulator blocks from the previous sections to those illustrated in Figure 5-25. It should suffice to mention that the simulator block for forcers A and B are identically the same in structure. The difference between the two arises only from the reluctance of the poles within each forcer.

Note that this simplification of forcers can be achieved by noting that equation (4.59) exhibit no coupling between the states of the forcers A and B. As detailed in the previous chapter, this observation allows the state equations of each forcer to be solved for independently. The force produced by each forcer is then summed to produce a net force on the mass of the motor.
Figure 5-26: Practical motor structure position response to velocity command of 50 [Hz].

5.4 Practical Motor Simulation Results

Similar simulations as those of the simple motor structure where run with the practical motor simulator. Detailed below are the results from the simulations, including: response to velocity command, static force estimation, and response to step input command.

5.4.1 Simulation Results to Velocity Command

The practical motor model responded in much the same way as the simple motor model responded to the frequency command of 50 [Hz]. The results for this simulation are presented in Figures 5-26 through 5-30.

From Figure 5-29, it can be observed that the practical motor works in exactly the same manner that the simple motor structure works. Namely, in order to advance, windings $A$ and $B$ are actuated in such a way so as to commutate the flux flowing through the poles. To move in either direction, the flux through the advancing pole
Figure 5-27: Total force generated by the practical structure to velocity command of 50 [Hz].

is increased while that of the trailing pole is decreased.

5.4.2 Simulation Results to Static Force Measurements

The simulation results for the static force measurement are provided in Figures 5-31 through 5-33.

Perhaps the most interesting observation to make from these results is the fact that the practical motor generates exactly twice as much force as that of the simple motor. This observation can be confirmed directly from Figure 5-11 for the simple motor structure and from Figure 5-31 for the practical motor.

A closer observation of the flux flowing through the poles of the practical motor forcers reveals how this motor structure can produce twice as much force compared to that of the simple motor. Recall that within each forcer, two poles are aligned such that the flux flowing through one pole is nearly the same as the other. (For example, consider poles 1a and 3a.) In addition, consider Figures 5-14 and 5-33. The flux for
Figure 5-28: Force generated by individual poles to velocity command of $50\,[Hz]$. 
Figure 5-29: Flux through air gaps found in response to velocity command of 50 [Hz].
Figure 5-30: Current inputs $I_A$ and $I_B$ to generate velocity command of 50 [Hz].

Figure 5-31: Total force generated holding winding currents constant.
Figure 5-32: Force generated by individual poles holding winding currents constant.
Figure 5-33: Flux flowing through individual poles using constant winding currents.
each motor structure reveal that the magnitude of the flux through corresponding poles is the same. And thus, by having two poles with the same flux, the practical motor structure generates twice as much force.

Note that as before, the flux running through the poles of the practical motor structure are also in saturation. Refer to Figure 5-33. As was the case with the simpler model, an appropriate method of dealing with the saturation is required so as to prevent the model from predicting excessive force generation.

5.4.3 Simulation Results to Step Input

The results of the step input simulation are presented in Figures 5-34 through 5-37.

Normalizing Figure 5-34, as illustrated in Figure 5-35, the following key parameters are observed:

\[
\text{Maximum Percent Overshoot} \quad \text{\(= 80.2 \, \text{[percent]} \)}
\]

\[
\text{Delay Time} \quad \text{\(= 1.76 \, \text{[msec]} \)}
\]
Figure 5.35: Time response of total force generated.

Rise Time \(-\quad 2.60\,[msec]\)

Peak Time \(-\quad 4.84\,[msec]\)

from which the characteristics of the system can be deduced as follows:

Damping Ratio \(\zeta\quad 7.0 \times 10^{-2}\)

Damped Natural Frequency \(\omega_d\quad 649\,\text{rad/sec}\)

Undamped Natural Frequency \(\omega_n\quad 651\,\text{rad/sec}\)

Note that although the characteristics of this more practical motor are not exactly those of the simpler one, the values are significantly close. Variations between the two structures were introduced primarily in the scaling of the flux so as not to exceed saturation. Namely, by changing the effective force the motor can generate, the stiffness of the system was modified, leading to slightly characteristics as note above.
Figure 5-36: Force generated by individual forcers.
Figure 5-37: Flux flowing through each pole.
Figure 5-38: Normalized time response to step input.
Chapter 6

Experimental Results

In order to verify the results obtained from the simulation, a series of key parameters on existing linear motors where measured by experimentation. At the time of writing, the NorMag 4XY2504-20 linear motor, in combination with a Motion Science stepper motor driver, was available for testing. This combination of motor and driver were used to obtain the characteristics of linear motors in response to: velocity command, step input, and static force measurement.

Briefly, the experimental setup for these series of tests consisted of the NorMag motor configured as illustrated in Figure 6-1. Initially, a host computer was used to generate the pulse(s) required to drive the NorMag linear motor. This signal was amplified by the motor driver so that sinusoidal current inputs could be supplied to the windings in the motor.

The position of the motor was then measured using a linear variable differential transformer (LVDT). The host computer served as a data collection center by digitizing, via Keithley/MetraByte DAS20 data acquisition board, the position signal and storing it for post-processing.

6.1 Response to Velocity Command

To determine the time response of the motor when commanded a constant velocity, a Motion Science driver card was used to generate a series of pulses that corresponded to
a motor linear velocity of $2 \left[ \frac{in}{sec} \right]$. The position response was digitized and normalized to multiples of a pitch. (Recall that $pitch = .040 \left[ in \right]$.) The results are shown in Figure 6-2.

From Figure 6-2, note that even at a constant velocity command, the position of the motor varies within each pitch. Namely, the period of oscillations observed in the figure corresponds to the pitch of the forcer.

### 6.2 Response to Step Input

To determine the characteristics of a typical linear motor, the NorMag motor was commanded to take a position step corresponding to $1/16$ of the pitch. (Note that the step response of $1/4$ pitch would have been preferable, as was done in the simulation, but the largest step generated by the driver was only $1/16$ of a pitch.) The position response of the motor was digitized using the DAS20 data acquisition board. The resulting data points were then normalized and plotted as shown in Figure 6-3.

The following system parameters were observed using Figure 6-3:

Maximum Percent Overshoot $\quad 73.4\ [\text{percent}]$
Figure 6-2: Position response of NorMag 4XY2504-20 linear motor to velocity command.

Figure 6-3: Normalized step response of NorMag 4XY2504-20 linear motor.
Figure 6-4: Experimental setup to measure stiffness of NorMag 4XY2504-20.

Delay Time $-$ 3.3 [msec]
Rise Time $-$ 4.8 [msec]
Peak Time $-$ 8.5 [msec]

The characteristics of the system were then deduced as follows:

$$\zeta = 9.8 \times 10^{-2}$$
$$\omega_d = 370.0 \frac{\text{rad}}{\text{sec}}$$
$$\omega_n = 371.4 \frac{\text{rad}}{\text{sec}}$$

6.3 Static Force Measurement

The static force measurement was used to determining two principle parameters: the "stiffness" of the motor, and the maximum holding force. Note that during these measurements, the motor was configured as shown in Figure 6-4. A known mass was hung to the motor via a cable and pulley as illustrated in the figure, thereby applying a tangential load on the motor.

In the stiffness test, the force versus displacement curve was obtained by applying
Figure 6-5: Force versus displacement curve of NorMag 4XY2504-20 motor.

increasing loads on the motor and measuring the displacement. The readings from
the LVDT were manually recorded, as were the values of weights used. The data was
compiled and plotted as shown in Figure 6-5.

A linear curve fit can be made through the data points in Figure 6-5 but note that,
for comparison with the simulation results, the force generation of the motor must
be divided by the number of forcers. Namely, the simulation modeled the behavior
of an individual forcer, while the measurements made in this section included eight
such forcers. Thus, Figure 6-5 was scaled accordingly and re-plotted in Figure 6-6.
A linear curve fit through this set of data points reveal that the stiffness of a single
forcer, corresponding to the slope of the curve fit, was found to be $8.9 \times 10^4 \frac{N}{m}$.

In a similar manner, the maximum holding force generated by the motor was
obtained by increasing the weights until the motor slipped. The approximate holding
force obtained using this technique was 25 [lbf], or 111 [N]. Taking into account
that eight forcers produced this holding force, the average holding force per forcer is
approximately 13.9 [N].
Figure 6-6: Scaled force versus displacement curve to determine stiffness of forcer.
Chapter 7

Conclusions and Recommendations

In this chapter, we will examine the accuracy of the simulation results. In addition, some general advice is given about the use of the simulators. Recommendations for future work are also presented.

7.1 Comparison of Results

Now that all the data is compiled, a comparison of the parameters obtained from both theory and experimentation is in order. Table 7.1 summarizes the parameters measured. Note that the table has been normalized to values obtained by a single forcer in each case.

As observed from the table, the values obtained for the damping ration in all three cases agree closely. The results indicate that the system is highly underdamped. Although precaution was taken to determine the correct values of the damping coefficient and other energy dissipative components, the simulations the simulators predicted slightly smaller than those measured in the NorMag motor.

It should be noted that the values obtained from both the simple and practical motor structures were very close in the damping ratio, although the simple motor one included the saturation effects while the more practical structure did not. The
maximum holding force of the forcer, however, was greatly effected by the saturation of the cores. As observed in Figures 5-11 and 5-31, the force generation characteristics of both structures is identical, ie both motors generate the same amount of force for a given number of forcers. The saturation effects considered in the simple motor structure indicate how limiting the effects of saturation really are.

In addition, note how the stiffness of the system is effected by the saturation effect. Namely, by reducing the force generation capability of the simple motor structure, the stiffness of that system was reduced by half, as indicated by the results obtained from the practical motor model. By making note of the fact that the force and stiffness values approximate the values measured in the NorMag motor, it can be deduced that larger force and stiffness characteristics can be obtained by overcoming the flux saturation observed in the flux path.

Although the simulation was useful in estimating the force characteristics of the linear motor, note the large discrepancy in the values of the natural frequency. A closer investigation of the saturation effects might bring the discrepancy to an acceptable level.

### 7.2 Discussion of Simulators

The results obtained from both simulators were nearly identical. Disregarding the saturation effects introduced into the simple motor structure, one can reasonably
conclude that the practical motor structure is simply a scaled version of the less complicated model. The only distinction that can be observed from the two simulators is the computational time; the practical motor model runs approximately 10 to 50 times slower. For example, the step response recorded for the simpler model was approximately 15 minutes long, while the practical structure took 8 hours! Thus, given that both simulations provide the same results, it is un-advisable to use the practical motor structure because the required computation time.

In general, it was also observed that the discontinuous current inputs required significantly small step sizes. Simulation step sizes of $1 \times 10^{-5} \text{[sec]}$ were typical for continuous sinusoidal inputs. The step response simulations, however, required that the simulation step size be in the order of $1 \times 10^{-6} \text{[sec]}$ for the simple motor structure, or $1 \times 10^{-7} \text{[sec]}$ for the practical motor model. Data collected from these simulations tended to be excessive, often requiring the use of only 1 of 20 or 1 of 100 points of the data recorded.

7.3 Recommendations for Future Work

The need for a better understanding of saturation effects is essential for the development of a better linear motor model. This area is perhaps the most impactful in quantifying the force generation capabilities of linear motors and a more thorough investigation must be carried out.

In addition, the magnetic energy dissipative elements mirrored to the mechanical domain would be more useful being incorporated into the correct energy domain. Although they were implemented in this manner to alleviate the complexity of the system of equations obtained in the magnetic domain, their physical representation soon looses perspective.
Appendix A

Detail Drawings

Provided in this appendix are detailed drawings of the linear motor components, including cores, permanent magnets, and the platen.
NOTES:
1. LAMINATION STACK MADE FROM 26 GAGE (PREFERRED)
   M19 GRADE SILICON STEEL OR 24 GAGE (ACCEPTABLE).
2. ALL DIMENSIONS ARE IN INCHES.

Figure A-1: Detail drawing of electromagnet core used by Normag linear Motor.

Figure A-2: Detail drawing of platen used with Normag linear Motor.
REFERENCES

1. MAGNETIC NORTH-SOUTH ALONG .075" WIDTH.
2. TO BE ARRANGED TO FORM 2.5" PM MAGNET AS ILLUSTRATED IN REFERENCE.
3. ALL DIMENSIONS ARE IN INCHES.

Figure A-3: Detail drawing of permanent magnets.

Figure A-4: Assembled Normag linear motor configuration.
Appendix B

Calculation of Reluctance Constants

In order to approximate the reluctance in the air gap as a function of tooth engagement, consider equation (3.7) from Chapter 3. In this expression, the permeance of the air gap was found to be:

\[ P = \kappa \mu_o \frac{A}{z} \]  \hspace{1cm} (B.1)

where \( A \) is the area of overlap and \( z \) is the distance that the flux has to pass through the air. The reluctance is simply the inverse of expression (B.1) and is expressed:

\[ \frac{1}{R} = \kappa \mu_o \frac{A}{z} \]  \hspace{1cm} (B.2)

The constant \( \kappa \) is the relative permeability of the material through which the flux is passing and \( \mu_o \) is called the permeability constant of free space. As discussed earlier, these two constants for the air gap are:

\[ \kappa \approx 1 \]

\[ \mu_o = 4\pi \times 10^{-7} \left[ \frac{\text{Henry}}{\text{meter}} \right] \]

The geometry of a typical electromagnet core for the forcer of the linear motor is presented in Figure A-1 of Appendix A. This drawing was compiled from approx-
imating the dimensions of the Normag linear motor. Using this geometry, consider the tooth/platen structure of one of the force poles presented in Figure B-1.

When the forcer pole teeth are aligned with the platen teeth, the flux flowing from the pole to the platen will experience the minimum reluctance provided that no displacements in the $z$ are allowed. From equation B.2, this minimal reluctance $R_{\text{min}}$ is simply the sum of all the area overlaps. Namely,

$$\frac{1}{R_{\text{min}}} = \mu_0 \left[ \sum_i \frac{A_i}{z_i} \right]$$

(B.3)

For the given geometry, we will consider the terms:

$$\frac{1}{R_{\text{min}}} \approx \mu_0 \left[ 4 \left( \frac{2.5 \cdot 0.02}{0.008} + \frac{2.5 \cdot 0.02}{0.0208} + \frac{2.5 \cdot 0.005}{0.0208} \right) + 3 \left( \frac{2.5 \cdot 0.015}{0.0408} \right) \right] \cdot 0.0254$$

which reduces to:

$$R_{\text{min}} = 2.32 \times 10^5 \left[ \frac{1}{H} \right]$$

Now consider when the forcer teeth are not aligned with the platen teeth. This configuration will produce the maximum reluctance, $R_{\text{max}}$, which can also be found using expression ( B.3). Using the geometry presented in Figure B-1, $R_{\text{max}}$ can be
approximated by:

\[
\frac{1}{R_{\text{max}}} \approx \mu_0 \left[ 4 \left( \frac{2.5 \cdot 0.005}{0.008} + \frac{2.5 \cdot 0.005}{0.0208} + 2.5 \cdot 0.02 \right) + 3 \left( \frac{2.5 \cdot 0.015}{0.0208} + \frac{2.5 \cdot 0.015}{0.0408} \right) \right] \cdot 0.0254
\]

which reduces to:

\[
R_{\text{max}} = 6.79 \times 10^6 \left[ \frac{1}{H} \right]
\]

Having the constants \( R_{\text{min}} \) and \( R_{\text{max}} \), the constant coefficients of equation (4.1), \( R_{\text{offset}} \) and \( R_{\text{amplitude}} \), can be found as follows:

\[
R_{\text{offset}} = \frac{1}{2} \left[ R_{\text{max}} + R_{\text{min}} \right] \quad \text{(B.4)}
\]

and

\[
R_{\text{amplitude}} = \frac{1}{2} \left[ R_{\text{max}} - R_{\text{min}} \right] \quad \text{(B.5)}
\]

Using these expressions and the calculated values of \( R_{\text{min}} \) and \( R_{\text{max}} \), we can find the numerical values of \( R_{\text{offset}} \) and \( R_{\text{amplitude}} \). These are:

\[
R_{\text{offset}} = 4.55 \times 10^5 \left[ \frac{1}{H} \right]
\]

\[
R_{\text{amplitude}} = 2.23 \times 10^5 \left[ \frac{1}{H} \right]
\]

Expression (4.1) gives the reluctance across the air gap as the function

\[
R = R_{\text{offset}} - R_{\text{amplitude}} \cdot \cos(\alpha x + \beta) \quad \text{(B.6)}
\]

As discussed in Chapter 4, the constant terms \( \alpha \) and \( \beta \) are a function of the pole geometry and are found to be:

\[
\alpha = \frac{2 \pi}{\text{pitch}} \quad \text{(B.7)}
\]
\[ \beta = \begin{cases} 
0 & \text{for pole 1,} \\
\pi & \text{for pole 2,} \\
\frac{1}{2}\pi & \text{for pole 3,} \\
\frac{3}{2}\pi & \text{for pole 4,} 
\end{cases} \] (B.8)

Note that the \( \beta \) term associated with each pole varies from pole to pole. The terms expressed in equation (B.8) are provided for the pole geometry illustrated in Figure 4-2 which is for the simplified linear motor. Similar terms can be found for the more practical motor construction.
Appendix C

Permanent Magnet Parameters

As described in Chapter 3, the permanent magnet operating curve can be decomposed into two elements, one as constant effort source and the other as a reluctance term. To obtain the parameters for these elements, data was collected from the 80 °C operating B-H curve for the rare-earth magnet neodymium-iron-boron illustrated in Figure 3-5. The data collected is shown in Figure C-1.

We are interested in obtaining a magnetomotive force versus flux plot from the collected data. The following steps were taken in the conversion process. First, the data collected was converted to SI units from which the $H$ vector of data was converted to a magnetomotive force by:

$$MMF = Hl_{magnet}$$  \hspace{1cm} (C.1)

where $l_{magnet}$ is the length of the path through which the flux flows in the permanent magnet. From Appendix A, Figure A-3, $l_{magnet} = .075$[in]. Likewise, the $B$ vector of data was converted to magnetic flux by noting:

$$\phi = BA_{magnet}$$  \hspace{1cm} (C.2)

where $A_{magnet}$ is the area through which the flux flows across the permanent magnet. From Appendix A, Figure A-3, $A_{magnet} = .150 \times 2.500$[in$^2$].

120
Figure C-1: Data collected for the rare-earth magnet *neodymium-iron-boron*.

Figure C-2: *M* vs. *ϕ* plot used to approximate *R_m* and *M_c*.

The resulting data was then plotted, magnetomotive force versus flux, as shown in Figure C-2. From these data points, a least squares fit was found for the equation

\[ MMF = R_m \cdot \phi + S_{pm} \]  \hspace{1cm} (C.3)

where the slope *R_m* corresponds to the reluctance term and *S_{pm}* is typically referred to as the coercive force and corresponds with the axis intercept. For the rare-earth magnet being considered, these values were found to be:

\[ R_m = 5.43 \times 10^6 \left[ \frac{\text{Am} \cdot m^2}{\text{Tesla}} \right] \]
\[ S_{pm} = -1510 \left[ \text{Am} \right] \]
Appendix D

Magnetic Losses

Parametric constants associated with energy dissipation in the magnetic domain are presented in this appendix.

D.1 Hysteresis Parameters

As detailed in Chapter 3, the energy loss associated with hysteresis in the iron path is simply the area enclosed by the $B$-$H$ hysteresis curve of the particular iron being used. A typical curve for silicon steel motor laminations is illustrated in Figure D-1. The curve depicts typical values for 26T265 Silicon Steel based on ASTM A773-91 testing standard. (The data has been provided courtesy of KJS Associates, Inc., Dayton, Ohio.)

As an approximation to the area enclosed by the hysteresis, consider the polygon in Figure D-2. Using conventional calculating techniques, the area of this polygon was found to be:

$$E_{p} \approx 15.0 \ [Oe \cdot K Gauss]$$

Noting the unit conversions, we find the energy inside the hysteresis curve to be:

$$E_{p} \approx 119.4 \ \left[ \frac{\text{Joule}}{m^3} \right]$$
Figure D-1: $B$-$H$ hysteresis curve for Si-steel lamination stack.

Figure D-2: Approximation of the area inside the hysteresis loop.
Note that this expression is an energy density. The total energy lost per cycle by the iron path is simply the volume times this density. Mathematically, we express the total energy loss per cycle, \( E_{\text{cycle}} \), as

\[
E_{\text{cycle}} = E_p \cdot V_{\text{iron}} \tag{D.1}
\]

where \( V_{\text{iron}} \) is the iron volume through that the magnetic flux penetrates.

The volume of iron through which the magnetic flux penetrates will be approximated by taking the total volume of the cores and that of the platen. Although in practice the average volume of core is smaller, an over estimation in this parameter will be compensated by the fact that the platen is not constructed from silicon lamination steel but from other grades of steel that are likely to contribute larger losses.

From Figure A-4, the volume per force can be approximated by that illustrated in Figure D-3. The total volume can be found to be:

\[
V_{\text{iron}} \approx 9.66 \times 10^{-6} \text{ [m}^3\text{]}
\]

Using expression (D.1), the total energy loss per hysteresis cycle is therefore:

\[
E_{\text{cycle}} = 1.15 \times 10^{-3} \text{ [J]}
\]

The hysteresis force expressed in equation (3.23) is now calculated by:

\[
F_{\text{hysteresis}} = \frac{E_{\text{cycle}}}{\text{pitch}} \tag{D.2}
\]

and is found to be:

\[
F_{\text{hysteresis}} = 1.13 \text{ [N]}
\]
D.2 Eddy Current Parameters

The damping coefficient used to approximate the eddy current losses, equation (3.36), was

\[ b = \left( \frac{\tau \pi B_{\text{max}}}{\text{pitch}} \right)^2 \frac{V_{\text{iron}}}{\delta \rho} \]  

(D.3)

The values used in calculating this coefficient are as follows:

- \( \tau \) – lamination thickness
- \( \text{pitch} \) – pitch of the motor
- \( \phi_{\text{max}} \) – maximum flux through pole
- \( \rho \) – resistivity of iron
- \( V_{\text{iron}} \) – iron volume of flux path

The geometrical parameters, including \( V_{\text{iron}}, \tau, \) and \( \text{pitch}, \) can all be found using the detail drawings provided in Appendix A. As noted in the previous section, the volume of iron is approximated by that illustrated in Figure D-3 and was found to be:

\[ V_{\text{iron}} \approx 9.66 \times 10^{-6} \text{[m}^3\text{]} \]

Refering to Appendix A, Figure A-1, the thickness 26 gauge laminations is \( .0185[\text{in}]. \) Parameters involving the material properties of the cores can be obtained from a number of references. The resistivity of silicon steel, according to reference [16], \(^{13}\)
\[ \approx 24 \times 10^{-8} \Omega \cdot m \]. For the \( B_{\text{max}} \), the hysteresis curve shown in Figure D-1 indicates that the laminated cores saturate at approximately 12\([KG]\).

Making all the appropriate conversions, the damping coefficient is found to be:

\[ b = \left[ \frac{\pi \cdot 0.4699 \cdot 1.2}{1.016} \right]^2 \frac{966}{6 \cdot 24} \approx 20.4 \left[ \frac{N \text{ sec}}{m} \right] \]

### D.3 Saturation Parameters

As was noted in Chapter 3, magnetic materials exhibit a maximum \( B \) field. This corresponds to the saturation level at which point it becomes increasingly harder to generate magnetic flux. For the iron path used by the linear motor, consider the \( B-H \) curve on Figure D-1. From this curve, it can be observed that \( B_{\text{sat}} \) is approximately 12\([KG]\). From this value, we can obtain the maximum flux \( \phi_{\text{max}} \) that the motor structure will be able to handle by noting that:

\[ \phi_{\text{sat}} = B_{\text{sat}} \cdot A_{\text{min}} \quad \text{(D.4)} \]

where \( A_{\text{min}} \) is the smallest cross-sectional area through which the flux must cross.

The minimum area encountered in the flux path corresponds to that of the platen teeth, where the area being considered for one pole is illustrated in Figure D-4. This area is found to be \( .1008 [in^2] \). In combination with the saturation \( B_{\text{sat}} \), the maximum flux is found to be:

\[ \phi_{\text{sat}} = \frac{12}{10} \cdot (.1008) \cdot (.0254^2) \approx 7.8 \times 10^{-5} \left[ \text{Tesla} \cdot m^2 \right] \]
Figure D-4: Minimal cross sectional area of iron path being considered for saturation.
Appendix E

Number of States for Simple Motor

When using power variables to describe a system, the effort quantity will be the state variable. Thus, a reluctance term in integral causality in the magnetic domain will have $M_i$ as a state variable while a reluctance term with integral causality in the mechanical domain will have $F_i$ as a another state variable.

For each reluctance term, $i = 1, 2, 3, 4$, the magnetomotive force is related to the flux by:

$$M_i = R_i(x) \cdot \phi_i$$  \hspace{1cm} (E.1)

as stated in expression (3.6). The force generated by the flux through the air gap was found to be:

$$F_i = -\frac{1}{2} \frac{\partial R_i(x)}{\partial x} \phi_i^2$$  \hspace{1cm} (E.2)

as stated in expression (3.15). This pair of equations describes the state of the reluctance in both the mechanical and magnetic domains.

To determine if the states of the each reluctance term are unique, consider the perturbation of the system about some nominal operating point:

$$\delta M_i \equiv \frac{\partial M_i}{\partial \phi_i} \delta \phi_i \cdot \frac{\partial M_i}{\partial x} \delta x = R_i \delta \phi_i \cdot \frac{\partial R_i}{\partial x} \phi_i \delta x$$  \hspace{1cm} (E.3)
\[ \delta F_i = \frac{\partial F_i}{\partial \phi_i} \delta \phi_i + \frac{\partial F_i}{\partial x} \delta x = -\frac{\partial R_i}{\partial x} \phi_i \delta \phi_i - \frac{1}{2} \frac{\partial^2 R_i}{\partial x^2} \phi^2_i \delta x \]  

(E.4)

where \( R_i = R_i(x) \). Expressions (E.4) and (E.4) can be expressed in matrix form:

\[
\begin{bmatrix}
\delta M_i \\
\delta F_i
\end{bmatrix}
= \begin{bmatrix}
R_i & \frac{\partial R_i}{\partial x} \phi_i \\
-\frac{\partial R_i}{\partial x} \phi_i & -\frac{1}{2} \frac{\partial^2 R_i}{\partial x^2} \phi^2_i
\end{bmatrix}
\begin{bmatrix}
\delta \phi_i \\
\delta x
\end{bmatrix}
\]  

(E.5)

If the states of each reluctance term are to be independent, expressions (E.4) and (E.4) must be linearly independent. This implies that the determinant of expression (E.5) must not be zero. This can easily be verified by finding the determinant

\[ \det \begin{bmatrix}
R_i & \frac{\partial R_i}{\partial x} \phi_i \\
-\frac{\partial R_i}{\partial x} \phi_i & -\frac{1}{2} \frac{\partial^2 R_i}{\partial x^2} \phi^2_i
\end{bmatrix} = -\frac{1}{2} R_i \frac{\partial^2 R_i}{\partial x^2} \phi^2_i + \left( \frac{\partial R_i}{\partial x} \right)^2 \phi^2_i 
\]  

(E.6)

The reluctance term was found to be a function of position by the relationship:

\[ R = \frac{1}{P} = \frac{1}{\kappa \mu_o \frac{z}{h \cdot x}} \]  

(E.7)

as stated in equation (3.8). Carrying out the differentiation of equation (E.6), we find the determinant to be:

\[ -\frac{1}{2} R_i \frac{\partial^2 R_i}{\partial x^2} \phi^2_i + \left( \frac{\partial R_i}{\partial x} \right)^2 \phi^2_i = \left[ -\frac{z^2}{\kappa^2 \mu_o^2 h^2 x^4} + \frac{z^2}{\kappa^2 \mu_o^2 h^2 x^4} \right] \phi^2_i = 0 
\]  

(E.8)

The implications of expression (E.8) are obvious. The system of equations representing the two states of a reluctance term are not independent. Thus, the integral causality of each reluctance does not introduce an additional state. Using the convention of power variables, the two states of the magnetic circuit are \( M_1 \) and \( M_4 \). Note that in addition, a third state is found in the inertia mass of the forcer, \( \dot{x} \).
Appendix F

Derivation of State Equations for Simple Motor

In this appendix, we will concern ourselves with deriving the magnetic circuit state equations for the simple motor structure. Recall that the states used were $M_1$ and $M_4$.

From Section 4.2, the equations describing the simple motor were found to be:

\[
\begin{align*}
\dot{\phi}_1 &= \dot{\phi}_m + \dot{\phi}_2 \\ 
\dot{\phi}_4 &= \dot{\phi}_m - \dot{\phi}_3 \\ 
M_2 &= NI_A - M_1 \\ 
M_m &= S_{pm} + M_2 - M_3 \\ 
M_3 &= M_4 - NI_B \\ 
M_1 &= R_1(x)\phi_1 \\ 
\phi_2 &= \frac{M_2}{R_2(x)} \\ 
\phi_3 &= \frac{M_3}{R_3(x)} \\ 
M_4 &= R_4(x)\phi_4 \\ 
\phi_m &= \frac{M_m}{R_m}
\end{align*}
\]
Taking equation (F.1) and using equations (F.10) and (F.7) to find \( \dot{\phi}_m \) and \( \dot{\phi}_2 \), respectively, we find:

\[
\dot{\phi}_1 = \dot{\phi}_m + \dot{\phi}_2 = \frac{d}{dt} \left[ \frac{M_m}{R_m} \right] + \frac{d}{dt} \left[ \frac{M_2}{R_2} \right] \tag{F.11}
\]

Using equations (F.4), the \( M_m \) term is expanded in expression (F.11) so that:

\[
\dot{\phi}_1 = \frac{d}{dt} \left[ \frac{S_{pm} + M_2 - M_3}{R_m} \right] + \frac{d}{dt} \left[ \frac{M_2}{R_2} \right] \tag{F.12}
\]

and substituting (F.3) and (F.5) for \( M_2 \) and \( M_3 \), we arrive at

\[
\dot{\phi}_1 = \frac{d}{dt} \left[ \frac{S_{pm} + (NI_A - M_1) - (M_4 - NI_B)}{R_m} \right] + \frac{d}{dt} \left[ \frac{NI_A - M_1}{R_2} \right] \tag{F.13}
\]

From expression (F.6), \( \phi_1 \) is related to \( M_1 \) by:

\[
\phi_1 = \frac{M_1}{R_1} \tag{F.14}
\]

which implies that

\[
\dot{\phi}_1 = \frac{d}{dt} \left[ \frac{M_1}{R_1} \right] \tag{F.15}
\]

This \( \dot{\phi}_1 \) term can now be substituted into expression (F.13) so that

\[
\frac{d}{dt} \left[ \frac{M_1}{R_1} \right] = \frac{d}{dt} \left[ \frac{S_{pm} + NI_A + NI_B - M_1 - M_4}{R_m} \right] + \frac{d}{dt} \left[ \frac{NI_A - M_1}{R_2} \right] \tag{F.16}
\]

Note that \( R_1 \) and \( R_2 \) are time varying while \( S_{pm} \) and \( R_m \) are assumed to be constant.

Carrying out the time derivatives of expression (F.16) and rearranging the results, we arrive at:

\[
\frac{1}{R_m} \dot{M}_4 + \left( \frac{1}{R_1} + \frac{1}{R_2} + \frac{1}{R_m} \right) \dot{M}_1 = \left( \frac{\dot{R}_1}{R_1^2} + \frac{\dot{R}_2}{R_2^2} \right) M_1 - \frac{N \dot{R}_2}{R_2^2} I_A \\
+ \left( \frac{N}{R_m} + \frac{N}{R_2} \right) I_A + \frac{N}{R_m} \dot{I}_B \tag{F.17}
\]

Thus, the first equation of the system is derived, with only the terms associated with
the states and inputs being used.

Expression (F.2) is used in conjunction with equations (F.10) and (F.8) to arrive at:

\[
\dot{\phi}_4 = \dot{\phi}_m - \dot{\phi}_3 = \frac{d}{dt} \left[ \frac{M_m}{R_m} \right] + \frac{d}{dt} \left[ \frac{M_3}{R_3} \right]
\]  

(F.18)

Making use of equation (F.4) once again, expression (F.18) can be re-written as:

\[
\dot{\phi}_4 = \frac{d}{dt} \left[ \frac{S_{pm} + M_2 - M_3}{R_m} \right] - \frac{d}{dt} \left[ \frac{M_3}{R_3} \right]
\]  

(F.19)

and substituting (F.3) and (F.5) for \(M_2\) and \(M_3\), we arrive at

\[
\dot{\phi}_4 = \frac{d}{dt} \left[ \frac{S_{pm} + (NI_A - M_1) - (M_4 - NI_B)}{R_m} \right] - \frac{d}{dt} \left[ \frac{M_4 - NI_B}{R_3} \right]
\]  

(F.20)

From expression (F.9), \(\phi_4\) is related to \(M_4\) by:

\[
\phi_4 = \frac{M_4}{R_4}
\]  

(F.21)

which implies that

\[
\dot{\phi}_4 = \frac{d}{dt} \left[ \frac{M_4}{R_4} \right]
\]  

(F.22)

In a similar manner to the previous state equation, (F.20) and (F.22) are equated so that

\[
\frac{d}{dt} \left[ \frac{M_4}{R_4} \right] = \frac{d}{dt} \left[ \frac{S_{pm} + NI_A + NI_B - M_1 - M_4}{R_m} \right] - \frac{d}{dt} \left[ \frac{M_4 - NI_B}{R_3} \right]
\]  

(F.23)

Carrying out the time derivatives of expression (F.16) and rearranging the results, we arrive at:

\[
\frac{1}{R_m} \ddot{M}_1 + \left( \frac{1}{R_3} + \frac{1}{R_4} + \frac{1}{R_m} \right) \ddot{M}_4 = \left( \frac{\dot{R}_3}{R_3^2} + \frac{\dot{R}_4}{R_4^2} \right) M_4 - \frac{N \dot{R}_3}{R_3^2} I_B
\]

\[
+ \left( \frac{N}{R_m} + \frac{N}{R_3} \right) \dot{I}_B + \frac{N}{R_m} \dot{I}_A
\]  

(F.24)

Expression (F.24) becomes the second state equation.
The system of equations, including (F.17) and (F.24), can be written in matrix form:

\[ \mathbf{A} \frac{d\mathbf{M}}{dt} = \mathbf{B} \mathbf{M} + \mathbf{C} \mathbf{I} + \mathbf{D} \frac{d\mathbf{I}}{dt} \]  \hspace{1cm} (F.25)

where,

\[ \mathbf{A} = \begin{bmatrix} \left( \frac{1}{R_1} + \frac{1}{R_2} + \frac{1}{R_m} \right) & \left( \frac{1}{R_m} \right) \\ \left( \frac{1}{R_m} \right) & \left( \frac{1}{R_3} + \frac{1}{R_4} + \frac{1}{R_m} \right) \end{bmatrix} \]  \hspace{1cm} (F.26)

\[ \mathbf{B} = \begin{bmatrix} \left( \frac{\dot{R}_1}{R_1^2} + \frac{\dot{R}_2}{R_2^2} \right) & 0 \\ 0 & \left( \frac{\dot{R}_3}{R_3^2} + \frac{\dot{R}_4}{R_4^2} \right) \end{bmatrix} \]  \hspace{1cm} (F.27)

\[ \mathbf{C} = \begin{bmatrix} -N \left( \frac{\dot{R}_2}{R_2^2} \right) & 0 \\ 0 & -N \left( \frac{\dot{R}_1}{R_1^2} \right) \end{bmatrix} \]  \hspace{1cm} (F.28)

\[ \mathbf{D} = \begin{bmatrix} N \left( \frac{1}{R_m} + \frac{1}{R_2} \right) & N \left( \frac{1}{R_m} \right) \\ N \left( \frac{1}{R_m} \right) & N \left( \frac{1}{R_m} + \frac{1}{R_3} \right) \end{bmatrix} \]  \hspace{1cm} (F.29)

as expressed in equations (4.15) through (4.18).
Appendix G

Matrix Coefficients

The expanded form of the matrix coefficients discussed in Section 4.3 are presented in this appendix.

\[
\begin{align*}
a_{11} &= \frac{1}{R_{1a}} + \frac{1}{R_{2a}} + \frac{1}{R_{ma}} \\
a_{12} &= -\frac{1}{R_{ma}} \\
a_{21} &= -\frac{1}{R_{ma}} \\
a_{22} &= \frac{1}{R_{3a}} + \frac{1}{R_{4a}} + \frac{1}{R_{ma}} \\
a_{33} &= \frac{1}{R_{1b}} + \frac{1}{R_{2b}} + \frac{1}{R_{mb}} \\
a_{34} &= \frac{1}{R_{mb}} \\
a_{43} &= \frac{1}{R_{mb}} \\
a_{44} &= \frac{1}{R_{3b}} + \frac{1}{R_{4b}} + \frac{1}{R_{mb}} \\
b_{11} &= \frac{R_{1a}}{R_{1a}^2} + \frac{R_{2a}}{R_{2a}^2} \\
b_{22} &= \frac{R_{3a}}{R_{3a}^2} + \frac{R_{4a}}{R_{4a}^2} \\
b_{33} &= \frac{R_{1b}}{R_{1b}^2} + \frac{R_{2b}}{R_{2b}^2}
\end{align*}
\]
\[ b_{44} = \frac{R_{3b}}{R_{3b}^2} + \frac{R_{4b}}{R_{4b}^2} \]
\[ c_{11} = -N \frac{R_{1a}}{R_{1a}^2} \]
\[ c_{21} = -N \frac{R_{4a}}{R_{4a}^2} \]
\[ c_{32} = -N \frac{R_{1b}}{R_{1b}^2} \]
\[ c_{42} = -N \frac{R_{4b}}{R_{4b}^2} \]
\[ d_{11} = N \frac{1}{R_{1a}} \]
\[ d_{21} = N \frac{1}{R_{4a}} \]
\[ d_{32} = N \frac{1}{R_{1b}} \]
\[ d_{42} = N \frac{1}{R_{4b}} \]
\[ b_{11} = \frac{a_{22} b_{11}}{a_{11} a_{22} - a_{21} a_{12}} \]
\[ b_{12} = \frac{-a_{12} b_{22}}{a_{11} a_{22} - a_{21} a_{12}} \]
\[ b_{21} = \frac{-a_{21} b_{11}}{a_{11} a_{22} - a_{21} a_{12}} \]
\[ b_{22} = \frac{a_{11} b_{22}}{a_{11} a_{22} - a_{21} a_{12}} \]
\[ b_{33} = \frac{a_{44} b_{33}}{a_{33} a_{44} - a_{34} a_{43}} \]
\[ b_{34} = \frac{-a_{34} b_{44}}{a_{33} a_{44} - a_{34} a_{43}} \]
\[ b_{43} = \frac{-a_{43} b_{33}}{a_{33} a_{44} - a_{34} a_{43}} \]
\[ b_{44} = \frac{a_{33} b_{44}}{a_{33} a_{44} - a_{34} a_{43}} \]
\[ c_{11} = \frac{a_{22} c_{11}}{a_{22} c_{11} - a_{12} c_{21}} \]
\[ a_{11} a_{22} - a_{21} a_{12} \]
\[ c_{21} = \frac{a_{11} c_{21}}{a_{11} a_{22} - a_{21} a_{12}} \]
\[ a_{11} c_{21} - a_{21} c_{11} \]
\[ c_{32} = \frac{a_{44} c_{32}}{a_{33} a_{44} - a_{34} a_{43}} \]
\[ a_{33} a_{44} - a_{34} a_{43} \]
\[ c_{42} = \frac{a_{33} c_{42}}{a_{33} a_{44} - a_{34} a_{43}} \]
\[ \begin{align*}
\dd_{11} &= \frac{a_{22}d_{11} - a_{12}d_{21}}{a_{11}a_{22} - a_{21}a_{12}} \\
\dd_{12} &= \frac{a_{11}d_{21} - a_{21}d_{11}}{a_{11}a_{22} - a_{21}a_{12}} \\
\dd_{32} &= \frac{a_{44}d_{32} - a_{34}d_{42}}{a_{33}a_{44} - a_{34}a_{43}} \\
\dd_{42} &= \frac{a_{33}d_{42} - a_{43}d_{32}}{a_{33}a_{44} - a_{34}a_{43}}
\end{align*} \]
Appendix H

Simulation Code for Simple Motor Model

H.1 Calculation of Initial Conditions

Before commanding the motor to move, the system is assumed to of have reached some steady state value. These steady state values remove any dynamics associated with "turning on" the motor, thereby minimizing the computer time to calculate the values of interest.

To calculate correct initial values for our two states, $M_1$ and $M_4$, consider equations (4.2) through (4.11). If we are only considering the steady state term of the system, all time derivatives expressions can be reduced to their static condition. Namely, expressions (4.2) and (4.3) are reduced to the following:

$$\phi_1 = \phi_m + \phi_2 \quad \text{(H.1)}$$
$$\phi_4 = \phi_m - \phi_3 \quad \text{(H.2)}$$

In the simulation, the motor currents are initialized with current $I_A$ being fully energized and $I_B$ being off. This forces the motor to align pole 1, leaving the reluctance
of the poles to be as follows:

\[ R_1 = R_{\text{min}} \]  \hspace{1cm} (H.3)
\[ R_2 = R_{\text{max}} \]  \hspace{1cm} (H.4)
\[ R_3 = \frac{1}{2}(R_{\text{min}} + R_{\text{max}}) \]  \hspace{1cm} (H.5)
\[ R_4 = \frac{1}{2}(R_{\text{min}} + R_{\text{max}}) \]  \hspace{1cm} (H.6)

The system can now be determined by the system of equations:

\[
\begin{bmatrix}
1 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 \\
R_1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\
0 & R_2 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\
0 & 0 & R_3 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\
0 & 0 & 0 & R_4 & 0 & 0 & 0 & 0 & -1 & 0 \\
0 & 0 & 0 & 0 & R_m & 0 & 0 & 0 & 0 & -1
\end{bmatrix}
\begin{bmatrix}
\phi_1 \\
\phi_2 \\
\phi_3 \\
\phi_4 \\
\phi_m \\
M_1 \\
M_2 \\
M_3 \\
M_4 \\
M_m
\end{bmatrix}
= 
\begin{bmatrix}
0 \\
N I_A \\
S_{pm} \\
-N I_B
\end{bmatrix}
\]  \hspace{1cm} (H.7)

The problem presented in expression (H.7) is the simple linear system of equations:

\[ A \vec{X} = \vec{b} \]

commonly solved in any linear algebra problem. The solution to the system of equations is given by:

\[ \vec{X} = A^{-1} \vec{b} \]

which can be solved either numerically or symbolically. Note that our interests lie in the initial conditions of the states \( M_1 \) and \( M_4 \), which have been solved symbolically using expression (H.7) and are found to be:

\[ M_{1o} = \frac{R_1 [NI_A (R_4 R_3 + R_2 R_3 + R_m R_3 + R_4 R_2 + R_4 R_m) + S_{pm} R_3 (R_3 + R_4) + R_3 R_2 N I_6]}{R_4 R_3 R_1 + R_3 R_2 R_1 + R_m R_2 R_1 + R_4 R_2 R_1 + R_4 R_m R_1 + R_3 R_2 R_4 + R_3 R_m R_2 + R_m R_2 R_4} \]  \hspace{1cm} (H.8)

138
\[ M_{4o} = \frac{R_4 (N_1 I_2 R_2 + S_{pm} R_3 (R_2 + R_1) + NI_3 (R_1 R_3 + R_2 R_3 + R_m R_m + R_2 R_2 + R_m R_2))}{R_4 R_3 R_1 + R_3 R_2 R_1 + R_m R_3 R_1 + R_4 R_2 R_1 + R_4 R_4 R_1 + R_3 R_2 R_4 + R_3 R_2 R_2 + R_m R_2 R_4} \]  

Noting the above constraints on the reluctances and the currents, the calculated values of the initial magnetomotive forces are:

\[ M_{1o} = 59.75 \text{ [Aturns]} \]
\[ M_{4o} = 58.57 \text{ [Aturns]} \]

Note also that the initial conditions for the mechanical side must also be specified. These initial conditions are:

\[ x_o = 0 \text{ [m]} \]
\[ \dot{x}_o = 0 \text{ [m/sec]} \]

### H.2 Definition of Parameter Inputs

Before the simulation can be used, the initialization file inits.m must be run in MATLAB™. In this file, the following parameters are specified:

- \( max \) — maximum number of points to store
- \( T_{final} \) — final simulation time
- \( S_{max} \) — maximum integration step
- \( S_{min} \) — minimum integration step
- \( error \) — error tolerance for numerical integrator
- \( pitch \) — pitch of the motor
- \( gap \) — air gap height
- \( mass \) — mass of the motor
- \( I_{max} \) — maximum current
- \( f \) — driving frequency
\( R_{\text{min}} \) – minimum calculated reluctance
\( R_{\text{max}} \) – maximum calculated reluctance
\( R_m \) – permanent magnet reluctance term
\( S_{\text{pm}} \) – permanent magnet magnetomotive source
\( \text{damp} \) – damping coefficient for eddy currents
\( F_{\text{hys}} \) – force term for hysteresis
\( x_0 \) – initial position
\( v_0 \) – initial velocity

Within this file, note that the initial conditions for the magnetic circuit are calculated, \( M_{1o} \) and \( M_{4o} \), as well as other parameters used in the simulation.

H.3  Code Listing
FILE: inits.m

max = 50000;  % Maximum number of points to store for each variable
              % This variable is used to find out how big other
              % variables will be stored in the simulation
Tfinal= .01;   % The final time for the simulation = Ttotal
Smax = 1e-5;   % Maximum step allowed in the simulation.
Smin = 1e-6;   % Minimum step allowed in the simulation.
error = 1e-5;  % Error tolerance for RK-4/5 integrator.

pitch = .04*.0254; % The physical length of the pitch, .04".
N = 60;          % Number of turns in the windings.
gap = 0.0008*.0254; % Ideal gap size, as specified by Normag.
xo = 0;          % Initial position.
vo = 0;          % Initial velocity.
mass = .255;     % Physical mass of the forcer, in kg.
Imax = 1;        % Maximum current sent to windings, in Amps.
f = 50;          % An arbitrary driving frequency, Hertz.
damp = 20.4;     % Damping coefficient of eddy currents, N sec/m.
Phys = 1.13;     % Constant force term of hysteresis losses, N.

Rmin = 2.32e5;   % Minimum reluctance of the air gap, 1/Henry.
Rmax = 6.79e5;   % Maximum reluctance of the air gap, 1/Henry.
Roff = (Rmax+Rmin)/2;  % Combinations of the above two values used
Ramp = (Rmax-Rmin)/2;  % Makes the simulation blocks easier to read.
alpha = 2*pi/pitch;  % Terms used to calculate reluctance of poles.
betal = 0;  beta2 = pi; beta3 = pi/2; beta4 = 3*pi/2;

Rm = 5.86e6;     % Permanent magnet reluctance term, 1/Henry.
Spm = 1550;      % Permanent magnet MMF source, Amp-turns.

% % Calculation of initial conditions.
% Assumptions taken:
%  * start motor with winding A = Imax, winding B = 0
%  * start motor with x = 0, meaning that pole 1 is aligned
%
Ia = Imax;
R1 = Rmin;  R2 = Rmax;  R3 = (Rmin + Rmax)/2;  R4 = (Rmin + Rmax)/2;

det = R1*R3*R4 + R1*R2*R3 + R1*R3*Rm + R1*R2*R4 + R1*R4*Rm
     + R2*R3*R4 + R2*R3*Rm + R2*R4*Rm;

mmflo = R1*(N*Ia*(R3*R4 + R2*R3 + R3*Rm + R2*R4 + R4*Rm)
           + Sm*R2*(R3 + R4))/det;
mmf4o = R4*(N*Ia*R2*R3 + Sm*R3*(R1 + R2))/det;

Psat = 1.94e-4; % Data using Bsat X Amin to calculate flux saturation
Pmax = mmflo/R1; % Find maximum flux system thinks it has

clear det; clear R1; clear R2; clear R3; clear R4; clear Ia;
FILE: simple.m

function [ret,x0,str]=simple(t,x,u,flag);
%SIMPLE is the M-file description of the SIMULINK system named SIMPLE.
% The block-diagram can be displayed by typing: SIMPLE.
%
% SYS=SIMPLE(T,X,U,FLAG) returns depending on FLAG certain system values given time point, T, current state vector, X,
% and input vector, U.
% FLAG is used to indicate the type of output to be returned in SYS.
%
% Setting FLAG=1 causes SIMPLE to return state derivatives, FLAG=2 discrete states, FLAG=3 system outputs and FLAG=4 next sample time. For more information and other options see SFUNC.
%
% Calling SIMPLE with a FLAG of zero:
% [SIZES]=SIMPLE([],[],[],0), returns a vector, SIZES, which contains the sizes of the state vector and other parameters.
% SIZES(1) number of states
% SIZES(2) number of discrete states
% SIZES(3) number of outputs
% SIZES(4) number of inputs.
% For the definition of other parameters in SIZES, see SFUNC.
% See also, TRIM, LINMOD, LINSIM, EULER, RK23, RK45, ADAMS, GEAR.
%
% Note: This M-file is only used for saving graphical information;
% after the model is loaded into memory an internal model representation is used.

% the system will take on the name of this mfile:
sys = mfilename;
new_system(sys)
simver(1.2)
if(0 == (nargin + nargout))
    set_param(sys,'Location',[68,428,1048,935])
    open_system(sys)
end;
set_param(sys,'algorithm', 'RK-45')
set_param(sys,'Start time', '0.0')
set_param(sys,'Stop time', 'Tfinal')
set_param(sys,'Min step size', 'Smin')
set_param(sys,'Max step size', 'Smax')
set_param(sys,'Relative error', '1e-3')
set_param(sys,'Return vars', '')
add_block('built-in/Fcn', [sys,'/','\10'])
set_param([sys,'/','\10'],...
    'Expr','u[1]/10',...
    'position',[225,120,265,140])
add_block('built-in/Constant', [sys,'/','Constant'])
set_param([sys,'/','Constant'],...
    'Value','0',...
    'position',[30,315,50,335])
142
add_block('built-in/Fcn', [sys, '/', 'Ib - SINE'])
set_param([sys, '/', 'Ib - SINE'], ...
  'Expr', 'Imax*sin(2*pi*f*k[1])', ...
  'position', [220, 320, 260, 340])

% Subsystem 'Force_Generation'.

new_system([sys, '/', 'Force_Generation'])
set_param([sys, '/', 'Force_Generation', 'Location', [394, 102, 821, 626])

add_block('built-in/To Workspace', [sys, '/', 'Force_Generation/F4'])
set_param([sys, '/', 'Force_Generation/F4'], ...
  'mat-name', 'F4', ...
  'buffer', 'max', ...
  'position', [330, 361, 355, 379])

add_block('built-in/To Workspace', [sys, '/', 'Force_Generation/F3'])
set_param([sys, '/', 'Force_Generation/F3'], ...
  'mat-name', 'F3', ...
  'buffer', 'max', ...
  'position', [335, 271, 360, 289])

add_block('built-in/To Workspace', [sys, '/', 'Force_Generation/F2'])
set_param([sys, '/', 'Force_Generation/F2'], ...
  'mat-name', 'F2', ...
  'buffer', 'max', ...
  'position', [330, 186, 355, 204])

add_block('built-in/To Workspace', [sys, '/', 'Force_Generation/F1'])
set_param([sys, '/', 'Force_Generation/F1'], ...
  'mat-name', 'F1', ...
  'buffer', 'max', ...
  'position', [330, 101, 355, 119])

add_block('built-in/Inport', [sys, '/', 'Force_Generation/in_5'])
set_param([sys, '/', 'Force_Generation/in_5'], ...
  'Port', '5', ...
  'position', [20, 405, 40, 425])

add_block('built-in/Inport', [sys, '/', 'Force_Generation/in_4'])
set_param([sys, '/', 'Force_Generation/in_4'], ...
  'Port', '4', ...
  'position', [20, 80, 40, 100])

add_block('built-in/Inport', [sys, '/', 'Force_Generation/in_3'])
set_param([sys, '/', 'Force_Generation/in_3'], ...
  'Port', '4', ...
  'position', [20, 320, 40, 340])

add_block('built-in/Outport', [sys, '/', 'Force_Generation/Ftotal'])
set_param([sys, '/', 'Force_Generation/Ftotal'], ...
  'Port', '1', ...
  'position', [470, 270, 490, 290])
set_param([sys,'"Force_Generation/(Flux^2)3"],...
   'Expr',u[1]*u[1]/2,...
   'position',[105,320,145,340])

add_block('built-in/Fcn',[sys,'"Force_Generation/(dR/dx)4")
set_param([sys,'"Force_Generation/(dR/dx)4"],...
   'Expr','-Ramp*alpha*sin(alpha*u[1]+beta4)',...
   'position',[205,380,245,400])

add_block('built-in/Product',[sys,'"Force_Generation/Product3"]
set_param([sys,'"Force_Generation/Product3"],...
   'hide name',0,...
   'inputs','2',...
   'position',[270,377,285,428])

add_block('built-in/Fcn',[sys,'"Force_Generation/(Flux^2)4")
set_param([sys,'"Force_Generation/(Flux^2)4"],...
   'Expr',u[1]*u[1]/2,...
   'position',[105,405,145,425])

add_block('built-in/Sum',[sys,'"Force_Generation/Sum"]
set_param([sys,'"Force_Generation/Sum"],...
   'inputs','++++',...
   'position',[385,103,400,452])

add_line([sys,'"Force_Generation"],[405,280;465,280])
add_line([sys,'"Force_Generation"],[295,320;380,320])
add_line([sys,'"Force_Generation"],[315,320;315,280;330,280])
add_line([sys,'"Force_Generation"],[290,405;380,405])
add_line([sys,'"Force_Generation"],[315,405;315,370;325,370])
add_line([sys,'"Force_Generation"],[295,235;380,235])
add_line([sys,'"Force_Generation"],[310,235;310,195;325,195])
add_line([sys,'"Force_Generation"],[295,150;380,150])
add_line([sys,'"Force_Generation"],[310,150;310,110;325,110])
add_line([sys,'"Force_Generation"],[45,90;175,90;175,390;200,390])
add_line([sys,'"Force_Generation"],[175,135;205,135])
add_line([sys,'"Force_Generation"],[175,220;205,220])
add_line([sys,'"Force_Generation"],[175,305;205,305])
add_line([sys,'"Force_Generation"],[150,330;270,330])
add_line([sys,'"Force_Generation"],[45,415;100,415])
add_line([sys,'"Force_Generation"],[45,330;100,330])
add_line([sys,'"Force_Generation"],[45,245;100,245])
add_line([sys,'"Force_Generation"],[45,160;100,160])
add_line([sys,'"Force_Generation"],[255,135;270,135])
add_line([sys,'"Force_Generation"],[150,160;270,160])
add_line([sys,'"Force_Generation"],[255,220;270,220])
add_line([sys,'"Force_Generation"],[150,245;270,245])
add_line([sys,'"Force_Generation"],[255,305;270,305])
add_line([sys,'"Force_Generation"],[250,390;265,390])
add_line([sys,'"Force_Generation"],[150,415;265,415])

% Finished composite block 'Force_Generation'.

set_param([sys,'"Force_Generation"],...
   'position',[585,162,665,358])
145
% Subsystem 'Flux_Generation'.

new_system(['sys','/','Flux_Generation'])
set_param(['sys','/','Flux_Generation'],'Location',[67,58,996,740])

add_block('built-in/To Workspace',['sys','/','Flux_Generation/save8'])
set_param(['sys','/','Flux_Generation/save8'],
  'mat-name','m3',
  'buffer','max',
  'position',[710,590,740,610])

add_block('built-in/To Workspace',['sys','/','Flux_Generation/save7'])
set_param(['sys','/','Flux_Generation/save7'],
  'mat-name','m4',
  'buffer','max',
  'position',[710,460,740,480])

add_block('built-in/To Workspace',['sys','/','Flux_Generation/save6'])
set_param(['sys','/','Flux_Generation/save6'],
  'mat-name','m1',
  'buffer','max',
  'position',[725,220,755,240])

add_block('built-in/To Workspace',['sys','/','Flux_Generation/save5'])
set_param(['sys','/','Flux_Generation/save5'],
  'mat-name','m2',
  'buffer','max',
  'position',[695,55,725,75])

add_block('built-in/Fcn',['sys','/','Flux_Generation/use dl'])
set_param(['sys','/','Flux_Generation/use dl'],
  'Drop Shadow',4,
+ (1/(u[9]*u[3]))*u[11])',
  'position',[115,580,155,600])

add_block('built-in/To Workspace',['sys','/','Flux_Generation/save1'])
set_param(['sys','/','Flux_Generation/save1'],
  'mat-name','flx1',
  'buffer','max',
  'position',[865,310,895,330])

add_block('built-in/To Workspace',['sys','/','Flux_Generation/save4'])
set_param(['sys','/','Flux_Generation/save4'],
  'mat-name','flx4',
  'buffer','max',
  'position',[865,545,895,565])

add_block('built-in/To Workspace',['sys','/','Flux_Generation/save3'])
set_param(['sys','/','Flux_Generation/save3'],
  'mat-name','flx3',
  'buffer','max',
  'position',[870,670,900,690])

146
add_block('built-in/To Workspace',[sys,'/','Flux_Generation/save2'])
set_param([sys,'/','Flux_Generation/save2'],
    'mat-name','flx2',
    'buffer','max',
    'position',[865,135,895,155])

add_block('built-in/Product', [sys,'/','Flux_Generation/Prod6'])
set_param([sys,'/','Flux_Generation/Prod6'],
    'hide name',0,
    'inputs',2,
    'position',[765,623,775,667])

add_block('built-in/Fcn', [sys,'/','Flux_Generation/C3'])
set_param([sys,'/','Flux_Generation/C3'],
    'Expr','1/u[5]',
    'position',[700,645,740,665])

add_block('built-in/Product', [sys,'/','Flux_Generation/Prod5'])
set_param([sys,'/','Flux_Generation/Prod5'],
    'hide name',0,
    'inputs',2,
    'position',[765,493,775,537])

add_block('built-in/Fcn', [sys,'/','Flux_Generation/C4'])
set_param([sys,'/','Flux_Generation/C4'],
    'Expr','1/u[7]',
    'position',[700,515,740,535])

add_block('built-in/Product', [sys,'/','Flux_Generation/Prod4'])
set_param([sys,'/','Flux_Generation/Prod4'],
    'hide name',0,
    'inputs',2,
    'position',[765,258,775,302])

add_block('built-in/Fcn', [sys,'/','Flux_Generation/C1'])
set_param([sys,'/','Flux_Generation/C1'],
    'Expr','1/u[1]',
    'position',[700,280,740,300])

add_block('built-in/Product', [sys,'/','Flux_Generation/Prod3'])
set_param([sys,'/','Flux_Generation/Prod3'],
    'hide name',0,
    'inputs',2,
    'position',[765,83,775,127])

add_block('built-in/Fcn', [sys,'/','Flux_Generation/C2'])
set_param([sys,'/','Flux_Generation/C2'],
    'Expr','1/u[3]',
    'position',[700,105,740,125])

add_block('built-in/Outport', [sys,'/','Flux_Generation/Flux4'])
set_param([sys,'/','Flux_Generation/Flux4'],
    'Port','4',
    'position',[865,505,885,525])
add_block('built-in/Outport', ['sys', '/', 'Flux_Generation/Flux1'])
set_param(['sys', '/', 'Flux_Generation/Flux1'],
    'Port', '1',
    'position', [865, 270, 885, 290])

add_block('built-in/Outport', ['sys', '/', 'Flux_Generation/Flux3'])
set_param(['sys', '/', 'Flux_Generation/Flux3'],
    'Port', '3',
    'position', [870, 635, 890, 655])

add_block('built-in/Inport', ['sys', '/', 'Flux_Generation/in_1'])
set_param(['sys', '/', 'Flux_Generation/in_1'],
    'Port', '1',
    'position', [15, 375, 35, 395])

add_block('built-in/Outport', ['sys', '/', 'Flux_Generation/Flux2'])
set_param(['sys', '/', 'Flux_Generation/Flux2'],
    'Port', '2',
    'position', [865, 95, 885, 115])

add_block('built-in/Integrator', ['sys', '/', 'Flux_Generation/dM1'])
set_param(['sys', '/', 'Flux_Generation/dM1'],
    'Initial', 'mmf1o',
    'position', [535, 247, 560, 293])

add_block('built-in/Fcn', ['sys', '/', 'Flux_Generation/using M1'])
set_param(['sys', '/', 'Flux_Generation/using M1'],
    'Drop Shadow', 4,
    'position', [110, 180, 150, 200])

add_block('built-in/Fcn', ['sys', '/', 'Flux_Generation/1/\det[A]'])
set_param(['sys', '/', 'Flux_Generation/1/\det[A]'],
    'Drop Shadow', 4,
    'position', [110, 375, 150, 395])

add_block('built-in/Sum', ['sys', '/', 'Flux_Generation/Sum'])
set_param(['sys', '/', 'Flux_Generation/Sum'],
    'inputs', '++++',
    'position', [435, 181, 455, 339])

add_block('built-in/Integrator', ['sys', '/', 'Flux_Generation/dM4'])
set_param(['sys', '/', 'Flux_Generation/dM4'],
    'Initial', 'mmf4o',
    'position', [540, 486, 565, 524])

add_block('built-in/Fcn', ['sys', '/', 'Flux_Generation/using M4'])
set_param(['sys', '/', 'Flux_Generation/using M4'],
    'Drop Shadow', 4,
    'position', [110, 220, 150, 240])

add_block('built-in/Product', ['sys', '/', 'Flux_Generation/Product1'])
set_param([sys,'/','Flux_Generation/Product1'...]
  'hide name',0,...
  'inputs',2,...
  'position',[245,158,255,202])
add_block('built-in/Product',[sys,'/','Flux_Generation/Product3'])
set_param([sys,'/','Flux_Generation/Product3'...]
  'hide name',0,...
  'inputs',2,...
  'position',[245,218,255,262])
add_block('built-in/Product',[sys,'/','Flux_Generation/Prod1'])
set_param([sys,'/','Flux_Generation/Prod1'...]
  'inputs',2,...
  'position',[505,248,515,292])
add_block('built-in/Fcn',[sys,'/','Flux_Generation/using dl'])
set_param([sys,'/','Flux_Generation/using dl'...]
  'Drop Shadow',4,...
  'position',[110,310,150,330])
add_block('built-in/Fcn',[sys,'/','Flux_Generation/using I'])
set_param([sys,'/','Flux_Generation/using I'...]
  'Drop Shadow',4,...
  'position',[110,270,150,290])
add_block('built-in/Sum',[sys,'/','Flux_Generation/Sum1'])
set_param([sys,'/','Flux_Generation/Sum1'...]
  'inputs',+++++,...
  'position',[435,436,455,594])
add_block('built-in/Product',[sys,'/','Flux_Generation/Prod2'])
set_param([sys,'/','Flux_Generation/Prod2'...]
  'inputs',2,...
  'position',[505,483,515,527])
add_block('built-in/Fcn',[sys,'/','Flux_Generation/use M4'])
set_param([sys,'/','Flux_Generation/use M4'...]
  'Drop Shadow',4,...
  'position',[115,495,155,515])
add_block('built-in/Fcn',[sys,'/','Flux_Generation/use M1'])
set_param([sys,'/','Flux_Generation/use M1'...]
  'Drop Shadow',4,...
  'position',[115,455,155,475])
add_block('built-in/Product',[sys,'/','Flux_Generation/Product2'])
set_param([sys,'/','Flux_Generation/Product2'...]
  'hide name',0,...
add_block('built-in/Product', [sys, '/', 'Flux_Generation/Product6'])
set_param([sys, '/', 'Flux_Generation/Product6'],
'hide name', 0,
'inputs', '2',
'position', [250, 493, 260, 537])

add_block('built-in/Fcn', [sys, '/', 'Flux_Generation/use 1'])
set_param([sys, '/', 'Flux_Generation/use 1'],
'Drop Shadow', 4,
+(u[4]/(u[9]*u[3]^2))*u[10]',
'position', [115, 545, 155, 565])

add_block('built-in/Fcn', [sys, '/', 'Flux_Generation/Nla'])
set_param([sys, '/', 'Flux_Generation/Nla'],
'Expr', 'N*u[10]',
'position', [110, 70, 150, 90])

add_block('built-in/Sum', [sys, '/', 'Flux_Generation/M2'])
set_param([sys, '/', 'Flux_Generation/M2'],
'inputs', '+',
'position', [610, 63, 630, 127])

add_block('built-in/Fcn', [sys, '/', 'Flux_Generation/Nlb'])
set_param([sys, '/', 'Flux_Generation/Nlb'],
'Expr', 'N*u[12]',
'position', [115, 640, 155, 660])

add_block('built-in/Sum', [sys, '/', 'Flux_Generation/M3'])
set_param([sys, '/', 'Flux_Generation/M3'],
'inputs', '+',
'position', [610, 603, 630, 667])

add_block('built-in/Fcn', [sys, '/', 'Flux_Generation/sat2'])
set_param([sys, '/', 'Flux_Generation/sat2'],
'Expr', 'Psat*u[1]/Pmax',
'position', [795, 94, 820, 116])

add_block('built-in/Fcn', [sys, '/', 'Flux_Generation/sat1'])
set_param([sys, '/', 'Flux_Generation/sat1'],
'Expr', 'Psat*u[1]/Pmax',
'position', [800, 269, 825, 291])

add_block('built-in/Fcn', [sys, '/', 'Flux_Generation/sat3'])
set_param([sys, '/', 'Flux_Generation/sat3'],
'Expr', 'Psat*u[1]/Pmax',
'position', [800, 634, 825, 656])

add_block('built-in/Fcn', [sys, '/', 'Flux_Generation/sat4'])
set_param([sys, '/', 'Flux_Generation/sat4'],
'Expr', 'Psat*u[1]/Pmax',
'position', [800, 504, 825, 526])
% Finished composite block 'Flux_Generation'.

set_param(['sys','/','Flux_Generation'],
    'position',[460,202,515,358])

add_block('built-in/To Workspace',['sys','/','Time Keeper'])
set_param(['sys','/','Time Keeper'],
    'mat-name','time',
    'buffer','max',
    'position',[220,156,260,174])

add_block('built-in/Fcn',['sys','/','Ia - COS'])
set_param(['sys','/','Ia - COS'],
    'Expr','Ia max*cos(2*pi*f*u[1])',
    'position',[220,270,260,290])

add_block('built-in/Clock',['sys','/','Clock'])
set_param(['sys','/','Clock'],
    'position',[95,155,115,175])

% Subsystem 'Misc. Stuff'.

new_system(['sys','/','Misc. Stuff'])
set_param(['sys','/','Misc. Stuff'],'Location',[119,506,625,962])
open_system(['sys','/','Misc. Stuff'])

add_block('built-in/To Workspace',['sys','/','Misc. Stuff/Save Ia'])
set_param(['sys','/','Misc. Stuff/Save Ia'],
    'mat-name','Ia',
    'buffer','max',
    'position',[150,450,190,470])

add_block('built-in/To Workspace',['sys','/','Misc. Stuff/Save Ib'])
set_param(['sys','/','Misc. Stuff/Save Ib'],
    'mat-name','Ib',
    'buffer','max',
    'position',[155,540,195,560])

add_block('built-in/Derivative',['sys','/','Misc. Stuff/dIb'])
'Value', 'Rm', ...
'position', [195, 385, 215, 405])

add_block('built-in/Fcn', [sys, '/', 'Misc. Stuff/R2'])
set_param([sys, '/', 'Misc. Stuff/R2'], ...
'Expr', 'Roff-Ramp*cos(alpha*u[1]+beta2)', ...
'position', [185, 145, 225, 165])

add_block('built-in/Fcn', [sys, '/', 'Misc. Stuff/R3'])
set_param([sys, '/', 'Misc. Stuff/R3'], ...
'Expr', 'Roff-Ramp*cos(alpha*u[1]+beta3)', ...
'position', [185, 225, 225, 245])

add_block('built-in/Fcn', [sys, '/', 'Misc. Stuff/R4'])
set_param([sys, '/', 'Misc. Stuff/R4'], ...
'Expr', 'Roff-Ramp*cos(alpha*u[1]+beta4)', ...
'position', [185, 305, 225, 325])

add_line([sys, '/', 'Misc. Stuff'], [95, 515; 345, 515])
add_line([sys, '/', 'Misc. Stuff'], [95, 345; 130, 350; 150, 550])
add_line([sys, '/', 'Misc. Stuff'], [125, 435; 125, 460; 145, 460])
add_line([sys, '/', 'Misc. Stuff'], [125, 435; 125, 460; 145, 460])
add_line([sys, '/', 'Misc. Stuff'], [295, 555; 345, 555])
add_line([sys, '/', 'Misc. Stuff'], [241, 515; 241, 555; 255, 555])
add_line([sys, '/', 'Misc. Stuff'], [240, 435; 240, 475; 255, 475])
add_line([sys, '/', 'Misc. Stuff'], [295, 475; 345, 475])
add_line([sys, '/', 'Misc. Stuff'], [95, 75; 180, 75])
add_line([sys, '/', 'Misc. Stuff'], [150, 75; 150, 155; 180, 155])
add_line([sys, '/', 'Misc. Stuff'], [150, 155; 150, 155; 180, 235])
add_line([sys, '/', 'Misc. Stuff'], [150, 235; 150, 235; 180, 155])
add_line([sys, '/', 'Misc. Stuff'], [295, 355; 345, 355])
add_line([sys, '/', 'Misc. Stuff'], [295, 275; 245, 275])
add_line([sys, '/', 'Misc. Stuff'], [230, 315; 345, 315])
add_line([sys, '/', 'Misc. Stuff'], [240, 315; 240, 355; 255, 355])
add_line([sys, '/', 'Misc. Stuff'], [240, 235; 245, 235])
add_line([sys, '/', 'Misc. Stuff'], [240, 235; 245, 275; 255, 275])
add_line([sys, '/', 'Misc. Stuff'], [295, 195; 345, 195])
add_line([sys, '/', 'Misc. Stuff'], [230, 155; 345, 155])
add_line([sys, '/', 'Misc. Stuff'], [240, 155; 240, 195; 255, 195])
add_line([sys, '/', 'Misc. Stuff'], [230, 75; 345, 75])
add_line([sys, '/', 'Misc. Stuff'], [240, 75; 240, 115; 255, 115])
add_line([sys, '/', 'Misc. Stuff'], [295, 115; 345, 115])
add_line([sys, '/', 'Misc. Stuff'], [370, 315; 425, 315])
add_line([sys, '/', 'Misc. Stuff'], [220, 395; 345, 395])

% Finished composite block 'Misc. Stuff'.

set_param([sys, '/', 'Misc. Stuff'], ...
'position', [335, 204, 390, 356])

% Subsystem 'Mechanical'.

new_system([sys, '/', 'Mechanical'])
set_param([sys, '/', 'Mechanical'], ['Location', [598, 285, 996, 584]])
add_block('built-in/Fcn',[sys,'/','Mechanical/Fcn'])
set_param([sys,'/','Mechanical/Fcn'],
    'orientation',2,...
    'Expr','Phys*u[1]',...
    'position',[155,40,195,60])

add_block('built-in/MATLAB Fcn',[sys,'/','Mechanical/MATLAB Fcn'])
set_param([sys,'/','Mechanical/MATLAB Fcn'],
    'orientation',2,...
    'MATLAB Fcn','sign',...
    'Output Width','-1',...
    'position',[225,35,275,65])

add_block('built-in/To Workspace',[sys,'/','Mechanical/Save Ft'])
set_param([sys,'/','Mechanical/Save Ft'],
    'mat-name','Ft',...
    'buffer','max',...
    'position',[230,160,260,190])

add_line([sys,'/','Mechanical'],[435,120;495,120])
add_line([sys,'/','Mechanical'],[460,120;460,170;490,170])
add_line([sys,'/','Mechanical'],[350,120;395,120])
add_line([sys,'/','Mechanical'],[370,120;370,215;295,215])
add_line([sys,'/','Mechanical'],[205,120;225,120])
add_line([sys,'/','Mechanical'],[285,120;310,120])
add_line([sys,'/','Mechanical'],[235,215;145,215;145,250;165,250])
add_line([sys,'/','Mechanical'],[145,215;145,140;165,140])
add_line([sys,'/','Mechanical'],[370,120;370,50;395,50])
add_line([sys,'/','Mechanical'],[370,50;280,50])
add_line([sys,'/','Mechanical'],[220,50;200,50])
add_line([sys,'/','Mechanical'],[110,120;165,120])
add_line([sys,'/','Mechanical'],[150,50;150,100;165,100])
add_line([sys,'/','Mechanical'],[205,120;210,175;225,175])

% Finished composite block 'Mechanical'.

set_param([sys,'/','Mechanical'],
    'position',[750,161,840,359])

add_block('built-in/Step Fcn',[sys,'/','Step Fcn'])
set_param([sys,'/','Step Fcn'],
    'Time','10*Smax',...
    'Before','0',...
    'After','.1/(f*4)',...
    'position',[110,270,130,290])

add_line(sys,[120,165;170,165;170,130;220,130])
add_line(sys,[170,165;215,165])
add_line(sys,[520,220;580,220])
add_line(sys,[520,260;580,260])
add_line(sys,[520,300;580,300])
add_line(sys,[520,340;580,340])
add_line(sys,[395,280;455,280])
add_line(sys,[265,330;330,330])
add_line(sys,[670,260;745,260])
add_line(sys,[265,280;330,280])
add_line(sys,[845,260;880,260;880,115;305,115;305,230;330,230])
add_line(sys,[550,115;550,180;580,180])
add_line(sys,[135,280;215,280])
add_line(sys,[170,280;170,330;215,330])
% Return any arguments.
if (nargin > nargout)
    % Must use feval here to access system in memory
    if (nargin > 3)
        if (flag == 0)
            eval(['[ret,x0,str]=',sys,'(t,x,u,flag);'])
        else
            eval(['ret =', sys,'(t,x,u,flag);'])
        end
    else
        [ret,x0,str] = feval(sys);
    end
end
Appendix I

Simulation Code for Practical Motor Model

I.1 Calculation of Initial Conditions

To eliminate any initial condition transients, the following of the initial magnetomotive forces across $R_{2a,b}$ and $R_{3a,b}$ are calculated in this appendix. Note that $R_{ia,b}$ is used to define the reluctance of the $i^{th}$ pole in forcer $a$ or $b$. These values will be different, by the generalization of these terms will allow us to write the initial condition matrix once for both sets of forcers.

As was done in Appendix H, consider the steady state terms of the system, where all time derivatives are reduced to their static condition. Namely, expressions (4.32), (4.33), (4.35), and (4.36) are reduced to the following:

\[
\begin{align*}
\phi_{2a} &= \phi_{1a} + \phi_{ma} \\
\phi_{3a} &= \phi_{4a} - \phi_{ma} \\
\phi_{2b} &= \phi_{1b} + \phi_{mb} \\
\phi_{3b} &= \phi_{4b} - \phi_{mb}
\end{align*}
\] (I.1) (I.2) (I.3) (I.4)
Define the terms:

\[ R_{i,a,b} \quad \text{reluctance across the } i^{th} \text{ pole in forcer } a \text{ or } b \]  
\[ \phi_{i,a,b} \quad \text{flux through } i^{th} \text{ pole in forcer } a \text{ or } b \]  
\[ M_{i,a,b} \quad \text{magnetomotive force across } i^{th} \text{ pole in forcer } a \text{ or } b \]

These definitions allow us to write the \( 20 \times 20 \) system of equations describing the steady state system into two \( 10 \times 10 \) systems that only need to be solved once. Namely, equations (4.37) through (4.52), in combination with (I.1) through (I.4), can be written as:

\[
\begin{bmatrix}
1 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 & 1 \\
R_{11,a,b} & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\
0 & R_{21,a,b} & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\
0 & 0 & R_{31,a,b} & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\
0 & 0 & 0 & R_{41,a,b} & 0 & 0 & 0 & 0 & 0 & -1 \\
0 & 0 & 0 & 0 & R_{m1,a,b} & 0 & 0 & 0 & 0 & -1
\end{bmatrix}
\begin{bmatrix}
\phi_{11,a,b} \\
\phi_{21,a,b} \\
\phi_{31,a,b} \\
\phi_{41,a,b} \\
\phi_{m1,a,b} \\
I_{11,a,b} \\
I_{21,a,b} \\
I_{31,a,b} \\
I_{41,a,b} \\
I_{m1,a,b}
\end{bmatrix}
= 
\begin{bmatrix}
0 \\
0 \\
N_{1,a,b} \\
N_{2,a,b} \\
\Phi_{m1,a,b} \\
S_{pm} \\
0 \\
0 \\
0 \\
0 \\
0
\end{bmatrix}
\]

where the two \( 10 \times 10 \) system of equations comes from using either the \( a \) or \( b \) term associated with the subscript \( a, b \).

The solution for the initial condition of the states \( M_{21,a,b} \) and \( M_{31,a,b} \) have been carried out simbolictly. These are:

\[
M_{21,a,b} = \frac{R_3 \left[ N_{I,a,b} (R_1 R_3 + R_3 R_m + R_3 R_4 + R_4 R_m) + S_{pm} R_1 (R_3 + R_4) \right]}{R_4 R_3 R_1 + R_3 R_2 R_1 + R_4 R_2 R_1 + R_4 R_3 R_1 + R_4 R_3 R_4 + R_3 R_2 R_4 + R_2 R_m R_2 + R_m R_2 R_4}
\]

\[
M_{31,a,b} = \frac{R_3 \left[ N_{I,a,b} (R_1 R_3 + R_1 R_m + R_2 R_m + R_3 R_4) - S_{pm} R_4 (R_1 + R_2) \right]}{R_4 R_3 R_1 + R_3 R_2 R_1 + R_4 R_2 R_1 + R_4 R_3 R_1 + R_4 R_3 R_4 + R_3 R_2 R_4 + R_2 R_m R_2 + R_m R_2 R_4}
\]

where the term \( R_i \) is associated with \( R_{i1,a,b} \) for \( i = 1, 2, 3, 4 \).

At the initial state, the following conditions are specified. For forcer \( a \):

\[
I_a = I_{max}
\]

\[
S_{pm} = -S_{pm}
\]

\[
R_{1a} = R_{min}
\]

159
\[ R_{2a} = R_{\text{max}} \]  
(1.14)  
\[ R_{3a} = R_{\text{min}} \]  
(1.15)  
\[ R_{4a} = R_{\text{max}} \]  
(1.16)  
\[ R_{4a} = R_{\text{max}} \]  
(1.17)

so that the initial conditions for \( M_{2a} \) and \( M_{3a} \) are:

\[ M_{2a} = 0 \text{[Aturns]} \]  
(1.18)  
\[ M_{3a} = 60 \text{[Aturns]} \]  
(1.19)

For force \( b \), all poles are aligned and no current is running, so:

\[ I_b = 0 \]  
(1.20)  
\[ S_{pm} = +S_{pm} \]  
(1.21)  
\[ R_{1b} = \frac{(R_{\text{min}} + R_{\text{max}})}{2} \]  
(1.22)  
\[ R_{2b} = \frac{(R_{\text{min}} + R_{\text{max}})}{2} \]  
(1.23)  
\[ R_{3b} = \frac{(R_{\text{min}} + R_{\text{max}})}{2} \]  
(1.24)  
\[ R_{4b} = \frac{(R_{\text{min}} + R_{\text{max}})}{2} \]  
(1.25)  
\[ R_{4b} = \frac{(R_{\text{min}} + R_{\text{max}})}{2} \]  
(1.26)

leading to initial conditions for force \( b \):

\[ M_{2b} = 56.8 \text{[Aturns]} \]  
(1.27)  
\[ M_{3b} = -56.8 \text{[Aturns]} \]  
(1.28)

These values are calculated in the initialization file to be discussed in the next section.
I.2 Definition of Parameter Inputs

Before the simulation can be used, the initialization file initp.m must be run in MATLAB\textsuperscript{TM}. In this file, the following parameters are specified:

\begin{itemize}
  \item \textit{max} – maximum number of points to store
  \item \textit{T\textsubscript{final}} – final simulation time
  \item \textit{S\textsubscript{max}} – maximum integration step
  \item \textit{S\textsubscript{min}} – minimum integration step
  \item \textit{error} – error tolerance for numerical integrator
  \item \textit{pitch} – pitch of the motor
  \item \textit{gap} – air gap height
  \item \textit{mass} – mass of the motor
  \item \textit{I\textsubscript{max}} – maximum current
  \item \textit{f} – driving frequency
  \item \textit{R\textsubscript{min}} – minimum calculated reluctance
  \item \textit{R\textsubscript{max}} – maximum calculated reluctance
  \item \textit{R\textsubscript{m}} – permanent magnet reluctance term
  \item \textit{S\textsubscript{pm}} – permanent magnet magnetomotive source
  \item \textit{damp} – damping coefficient for eddy currents
  \item \textit{F\textsubscript{hys}} – force term for hysteresis
  \item \textit{x\textsubscript{o}} – initial position
  \item \textit{v\textsubscript{o}} – initial velocity
\end{itemize}

Within this file, note that the initial conditions for the magnetic circuit are calculated, \(M_{2a}, M_{3a}, M_{2b},\) and \(M_{3b},\) as well as other parameters used in the simulation.
I.3 Code Listing
max = 50000; % Maximum number of points to store for each variable
% This variable is used to find out how big other
% variables will be stored in the simulation
Tfinal = .01; % The final time for the simulation = TTotal
Smax = 1e-5; % Maximum step allowed in the simulation.
Smin = 1e-6; % Minimum step allowed in the simulation.
error = 1e-5; % Error tolerance for RK-4/5 integrator.
pitch = .04*.0254; % The physical length of the pitch, .04".
N = 60; % Number of turns in the windings.
gap = 0.0008*.0254; % Ideal gap size, as specified by Normag.
x0 = 0; % Initial position.
vo = 0; % Initial velocity.
mass = 2*.255; % Physical mass of the forcer, in kg.
Imax = 1; % Maximum current sent to windings, in Amps.
f = 50; % An arbitrary driving frequency, Hertz.
damp = 2*20.4; % Damping coefficient of eddy current, N sec/m.
Fphys = 2*1.13; % Constant force term of hysteresis losses, N.

Rmin = 2.32e5; % Minimum reluctance of the air gap, 1/Henry.
Rmax = 6.79e5; % Maximum reluctance of the air gap, 1/Henry.
Roff = (Rmax+Rmin)/2; % Combinations of the above two values
Ramp = (Rmax-Rmin)/2; % Makes the simulation blocks easier to read.
alpha = 2*pi/pitch; % Terms used to calculate reluctance of poles.
beta1 = 0;
beta2 = pi;
beta3 = pi/2;
beta4 = 3*pi/2;

Rm = 5.8e6; % Data for the permanent magnet reluctance te:m, 1/H.
Spm = 1560; % Data for the permanent magnet MMF source, Amp-turns.

% Calculation of initial conditions.
% Assumptions taken:
% * start motor with winding A = Imax, winding B = 0
% * start motor with x = 0, meaning that pole 1a is aligned
%
Ia = Imax;
Spm = -Spm;
R1 = Rmin; R2 = Rmax; R3 = Rmin; R4 = Rmax;

det = R1*R3*R4 + R1*R2*R3 + R1*R3*Rm + R1*R2*R4 + R1*R4*Rm
      + R2*R3*R4 + R2*R3*Rm + R2*R4*Rm;

mmf2a = R2*(N*Ia*(R3*R4 + R1*R3 + R3*Rm + R4*Rm)
         + Spm*R1*(R3 + R4))/det;
mmf3a = R3*(N*Ia*(R1*R2 + R1*Rm + R2*Rm + R2*R4)
        - Spm*R4*(R1 + R2))/det;
Ib = 0;
Spm = -Spm;
R1 = (Rmax + Rmin)/2;
R2 = (Rmax + Rmin)/2;
R3 = (Rmax + Rmin)/2;
R4 = (Rmax + Rmin)/2;
det = R1*R3*R4 + R1*R2*R3 + R1*R3*Rm + R1*R2*R4 + R1*R4*Rm
    + R2*R3*R4 + R2*R3*Rm + R2*R4*Rm;

mmf2b = R2*(N*Ib*(R3*R4 + R1*R3 + R3*Rm + R4*Rm)
    + Spm*R1*(R3 + R4))/det;
mmf3b = R3*(N*Ib*(R1*R2 + R1*Rm + R2*Rm + R2*R4)
    - Spm*R4*(R1 + R2))/det;

Psat = 1.94e-4; % Data using Bs at x Amin to calculate flux saturation
Pmax = mmf3a/R1; % Find maximum flux system thinks it has

clear det; clear R1; clear R2; clear R3; clear R4; clear Ia; clear Ib;
function [ret, x0, str] = practical(t, x, u, flag);

% The block-diagram can be displayed by typing: PRACTICAL.

% SYS = PRACTICAL(T, X, U, FLAG) returns depending on FLAG certain
% system values given time point, T, current state vector, X,
% and input vector, U.
% FLAG is used to indicate the type of output to be returned in SYS.

% Setting FLAG=1 causes PRACTICAL to return state derivatives, FLAG=2
% discrete states, FLAG=3 system outputs and FLAG=4 next sample
% time. For more information and other options see SFUNC.

% Calling PRACTICAL with a FLAG of zero:
% [SIZES] = PRACTICAL([], [], [], 0), returns a vector, SIZES, which
% contains the sizes of the state vector and other parameters.
% SIZES(1) number of states
% SIZES(2) number of discrete states
% SIZES(3) number of outputs
% SIZES(4) number of inputs.
% For the definition of other parameters in SIZES, see SFUNC.
% See also, TRIM, LINMOD, LINSIM, EULER, RK23, RK45, ADAMS, GEAR.

% Note: This M-file is only used for saving graphical information;
% after the model is loaded into memory an internal model
% representation is used.

% the system will take on the name of this mfile:
sys = mfilename;
new_system(sys)
simver(1.2)
if(0 == (nargin + nargout))
  set_param(sys, 'Location', [9, 179, 996, 719])
  open_system(sys)
end;
set_param(sys, 'algorithm', 'RK-45')
set_param(sys, 'Start time', '0.0')
set_param(sys, 'Stop time', 'Tfinal')
set_param(sys, 'Min step size', 'Smax')
set_param(sys, 'Max step size', 'Smin')
set_param(sys, 'Relative error', 'error')
set_param(sys, 'Return vars', '')

% Subsystem 'Mechanical Domain'.

new_system([sys, '/','Mechanical Domain'])
set_param([sys, '/','Mechanical Domain'], 'Location', [598, 285, 996, 584])

add_block('built-in/To Workspace', [sys, '/','Mechanical Domain/Save Ft'])
set_param([sys, '/','Mechanical Domain/Save Ft'], ...
  'mat-name', 'Ft', ...
  'buffer', 'max', ...
  'position', [230, 180, 260, 210])
add_block('built-in/MATLAB Fcn', [sys, '/', 'Mechanical Domain/MATLAB Fcn'])
set_param([sys, '/', 'Mechanical Domain/MATLAB Fcn'],
    'orientation', 2, ...
    'MATLAB Fcn', 'sign', ...
    'Output Width', '-1', ...
    'position', [225, 35, 275, 65])

add_block('built-in/Fcn', [sys, '/', 'Mechanical Domain/Fcn'])
set_param([sys, '/', 'Mechanical Domain/Fcn'],
    'orientation', 2, ...
    'Expr', 'Phys*u[1]', ...
    'position', [155, 40, 195, 60])

add_block('built-in/To Workspace', [sys, '/', 'Mechanical Domain/Save Fd'])
set_param([sys, '/', 'Mechanical Domain/Save Fd'],
    'mat-name', 'Fd', ...
    'buffer', 'max', ...
    'position', [170, 270, 200, 300])

add_block('built-in/To Workspace', [sys, '/', 'Mechanical Domain/Save Vt'])
set_param([sys, '/', 'Mechanical Domain/Save Vt'],
    'mat-name', 'Vt', ...
    'buffer', 'max', ...
    'position', [400, 35, 430, 65])

add_block('built-in/Outport', [sys, '/', 'Mechanical Domain/out_1'])
set_param([sys, '/', 'Mechanical Domain/out_1'],
    'Port', '1', ...
    'position', [500, 110, 120, 130])

add_block('built-in/Inport', [sys, '/', 'Mechanical Domain/in_1'])
set_param([sys, '/', 'Mechanical Domain/in_1'],
    'Port', '1', ...
    'position', [85, 110, 105, 130])

add_block('built-in/Integrator', [sys, '/', 'Mechanical Domain/Velocity'])
set_param([sys, '/', 'Mechanical Domain/Velocity'],
    'Initial', 'vo', ...
    'position', [315, 99, 345, 141])

add_block('built-in/Sum', [sys, '/', 'Mechanical Domain/Sum'])
set_param([sys, '/', 'Mechanical Domain/Sum'],
    'inputs', '++-', ...
    'position', [170, 92, 200, 148])

add_block('built-in/Gain', [sys, '/', 'Mechanical Domain/1//MASS'])
set_param([sys, '/', 'Mechanical Domain/1//MASS'],
    'Gain', '1/mass', ...
    'position', [230, 95, 280, 145])

add_block('built-in/Gain', [sys, '/', 'Mechanical Domain/DAMPING'])
set_param([sys, '/', 'Mechanical Domain/DAMPING'],
    'orientation', 2, ...
add_block('built-in/Integrator',[sys,'/','Mechanical Domain/Position'])
set_param([sys,'/','Mechanical Domain/Position'],
          'Initial',xo,...
          'position',[400,99,430,141])
add_line([sys,'/','Mechanical Domain'],[205,120;225,120])
add_line([sys,'/','Mechanical Domain'],[215,120;215,195;225,195])
add_line([sys,'/','Mechanical Domain'],[150,50;150,100;165,100])
add_line([sys,'/','Mechanical Domain'],[110,120;165,120])
add_line([sys,'/','Mechanical Domain'],[220,50;200,50])
add_line([sys,'/','Mechanical Domain'],[350,120;395,120])
add_line([sys,'/','Mechanical Domain'],[370,120;370,50;395,50])
add_line([sys,'/','Mechanical Domain'],[370,50;280,50])
add_line([sys,'/','Mechanical Domain'],[235,250;145,250;145,285;165,285])
add_line([sys,'/','Mechanical Domain'],[145,250;145,140;165,140])
add_line([sys,'/','Mechanical Domain'],[285,120;310,120])
add_line([sys,'/','Mechanical Domain'],[370,120;370,250;295,250])
add_line([sys,'/','Mechanical Domain'],[350,120;495,120])

% Finished composite block 'Mechanical Domain'.

set_param([sys,'/','Mechanical Domain'],
          'position',[765,196,855,394])
add_block('built-in/Fcn',[sys,'/','Ib - SINE'])
set_param([sys,'/','Ib - SINE'],
          'Expr','Imax*sin(2*pi*f*u[1])',
          'position',[180,445,220,465])

add_block('built-in/Fcn',[sys,'/','Ia - COS'])
set_param([sys,'/','Ia - COS'],
          'Expr','Imax*cos(2*pi*f*u[1])',
          'position',[180,220,220,240])

add_block('built-in/Constant',[sys,'/','Constant'])
set_param([sys,'/','Constant'],
          'Value','0',
          'position',[45,220,65,240])

add_block('built-in/To Workspace',[sys,'/','Current Ia'])
set_param([sys,'/','Current Ia'],
          'orientation',2,
          'mat-name','Ia',
          'buffer','max',
          'position',[180,175,220,195])

add_block('built-in/To Workspace',[sys,'/','Current Ib'])
set_param([sys,'/','Current Ib'],
          'orientation',2,
          'mat-name','Ib',
          'buffer','max',
          'position',[180,175,220,195])
% Subsystem 'Reluctance A'.

new_system([sys,'/','Reluctance A'])
set_param([sys,'/','Reluctance A'], 'Location', [35, 82, 541, 538])

add_block('built-in/Fcn', [sys,'/','Reluctance A/R4a'])
set_param([sys,'/','Reluctance A/R4a'], ...
    'Expr', 'Roff-Ramp*cos(alpha*u[1]+beta2)', ...
    'position', [180, 305, 220, 325])

add_block('built-in/Fcn', [sys,'/','Reluctance A/R2a'])
set_param([sys,'/','Reluctance A/R2a'], ...
    'Expr', 'Roff-Ramp*cos(alpha*u[1]+beta2)', ...
    'position', [180, 145, 220, 165])

add_block('built-in/Fcn', [sys,'/','Reluctance A/R3a'])
set_param([sys,'/','Reluctance A/R3a'], ...
    'Expr', 'Roff-Ramp*cos(alpha*u[1]+beta1)', ...
    'position', [180, 225, 220, 245])

add_block('built-in/Fcn', [sys,'/','Reluctance ::/R1a'])
set_param([sys,'/','Reluctance A/R1a'], ...
    'Expr', 'Roff-Ramp*cos(alpha*u[1]+beta1)', ...
    'position', [185, 65, 225, 85])

add_block('built-in/Constant', [sys,'/','Reluctance A/Rm'])
set_param([sys,'/','Reluctance A/Rm'], ...
    'Value', 'Rm', ...
    'position', [195, 385, 215, 405])

add_block('built-in/Mux', [sys,'/','Reluctance A/Mux'])
set_param([sys,'/','Reluctance A/Mux'], ...
    'inputs', '11', ...
    'position', [350, 53, 365, 497])

add_block('built-in/Outport', [sys,'/','Reluctance A/out_1'])
set_param([sys,'/','Reluctance A/out_1'], ...
    'Port', '1', ...
    'position', [430, 265, 450, 785])

add_block('built-in/Derivative', [sys,'/','Reluctance A/dR1a'])
set_param([sys,'/','Reluctance A/dR1a'], ...
    'position', [260, 105, 290, 125])

add_block('built-in/Derivative', [sys,'/','Reluctance A/dR2a'])
set_param([sys,'/','Reluctance A/dR2a'], ...
    'position', [260, 185, 290, 205])

add_block('built-in/Derivative', [sys,'/','Reluctance A/dR3a'])
set_param([sys,'/','Reluctance A/dR3a'], ...
    'position', [260, 265, 290, 285])

168
add_block('built-in/Derivative', [sys, '/', 'Reluctance A/dR4a'])
set_param([sys, '/', 'Reluctance A/dR4a'],
  'position', [260, 345, 290, 365])

add_block('built-in/Inport', [sys, '/', 'Reluctance A/Position X'])
set_param([sys, '/', 'Reluctance A/Position X'],
  'Port', '1',
  'position', [70, 65, 90, 85])

add_block('built-in/Inport', [sys, '/', 'Reluctance A/Current Ia'])
set_param([sys, '/', 'Reluctance A/Current Ia'],
  'Port', '2',
  'position', [70, 425, 90, 445])

add_block('built-in/Derivative', [sys, '/', 'Reluctance A/dIa'])
set_param([sys, '/', 'Reluctance A/dIa'],
  'position', [260, 465, 290, 485])
add_line([sys, '/', 'Reluctance A'], [220, 395; 345, 395])
add_line([sys, '/', 'Reluctance A'], [370, 275; 425, 275])
add_line([sys, '/', 'Reluctance A'], [295, 115; 345, 115])
add_line([sys, '/', 'Reluctance A'], [75, 75; 345, 75])
add_line([sys, '/', 'Reluctance A'], [240, 75; 240, 115; 255, 115])
add_line([sys, '/', 'Reluctance A'], [225, 155; 345, 155])
add_line([sys, '/', 'Reluctance A'], [240, 155; 240, 195; 255, 195])
add_line([sys, '/', 'Reluctance A'], [295, 195; 345, 195])
add_line([sys, '/', 'Reluctance A'], [225, 235; 345, 235])
add_line([sys, '/', 'Reluctance A'], [240, 235; 240, 275; 255, 275])
add_line([sys, '/', 'Reluctance A'], [225, 315; 345, 315])
add_line([sys, '/', 'Reluctance A'], [240, 315; 240, 355; 255, 355])
add_line([sys, '/', 'Reluctance A'], [295, 275; 345, 275])
add_line([sys, '/', 'Reluctance A'], [295, 355; 345, 355])
add_line([sys, '/', 'Reluctance A'], [95, 75; 180, 75])
add_line([sys, '/', 'Reluctance A'], [150, 75; 150, 155; 175, 155])
add_line([sys, '/', 'Reluctance A'], [150, 155; 150, 235; 175, 235])
add_line([sys, '/', 'Reluctance A'], [150, 235; 150, 315; 175, 315])
add_line([sys, '/', 'Reluctance A'], [295, 475; 345, 475])
add_line([sys, '/', 'Reluctance A'], [95, 435; 345, 435])
add_line([sys, '/', 'Reluctance A'], [240, 435; 240, 475; 255, 475])

% Finished composite block 'Reluctance A'.

set_param([sys, '/', 'Reluctance A'],
  'position', [290, 117, 345, 268])

add_block('built-in/Clock', [sys, '/', 'Clock'])
set_param([sys, '/', 'Clock'],
  'position', [55, 105, 75, 125])

add_block('built-in/To Workspace', [sys, '/', 'Time Keeper'])
set_param([sys, '/', 'Time Keeper'],
  'mat-name', 'time',
  'buffer', 'max',
  'position', [180, 106, 220, 124])

169
new_system(['sys','/','Flux Generation A'])
set_param(['sys','/','Flux Generation A'], 'Location', [234, 82, 1163, 801])

add_block('built-in/Sum', ['sys','/','Flux Generation A/M4'])
set_param(['sys','/','Flux Generation A/M4'],
          'inputs', '+-',
          'position', [610, 603, 630, 667])

add_block('built-in/Fcn', ['sys','/','Flux Generation A/N*Ia'])
set_param(['sys','/','Flux Generation A/N*Ia'],
          'Expr', 'N*u[10]',
          'position', [115, 640, 155, 660])

add_block('built-in/Sum', ['sys','/','Flux Generation A/M1'])
set_param(['sys','/','Flux Generation A/M1'],
          'inputs', '+-',
          'position', [610, 63, 630, 127])

add_block('built-in/Fcn', ['sys','/','Flux Generation A/N*Ia'])
set_param(['sys','/','Flux Generation A/N*Ia'],
          'Expr', 'N*u[10]',
          'position', [110, 70, 150, 90])

add_block('built-in/Fcn', ['sys','/','Flux Generation A/use Ia'])
set_param(['sys','/','Flux Generation A/use Ia'],
          'Drop Shadow', 4,
          'position', [115, 554, 155, 565])

add_block('built-in/Product', ['sys','/','Flux Generation A/Product6'])
set_param(['sys','/','Flux Generation A/Product6'],
          'hide name', 0,
          'inputs', '2',
          'position', [250, 493, 260, 537])

add_block('built-in/Product', ['sys','/','Flux Generation A/Product2'])
set_param(['sys','/','Flux Generation A/Product2'],
          'hide name', 0,
          'inputs', '2',
          'position', [250, 433, 260, 477])

add_block('built-in/Fcn', ['sys','/','Flux Generation A/use M2'])
set_param(['sys','/','Flux Generation A/use M2'],
          'Drop Shadow', 4,
          'position', [115, 455, 155, 457])

add_block('built-in/Fcn', ['sys','/','Flux Generation A/use M3'])
set_param(['sys','/','Flux Generation A/use M3'],
          'Drop Shadow', 4,
add_block('built-in/Product', [sys, '/','Flux Generation A/Prod2'])
set_param([sys, '/','Flux Generation A/Prod2'],
  'inputs', '2',
  'position', [505, 483, 515, 527])

add_block('built-in/Sum', [sys, '/','Flux Generation A/Sum1'])
set_param([sys, '/','Flux Generation A/Sum1'],
  'inputs', '++++',
  'position', [435, 436, 455, 594])

add_block('built-in/Fcn', [sys, '/','Flux Generation A/using Ia'])
set_param([sys, '/','Flux Generation A/using Ia'],
  'position', [110, 270, 150, 290])

add_block('built-in/Fcn', [sys, '/','Flux Generation A/using dIa'])
set_param([sys, '/','Flux Generation A/using dIa'],
  'position', [110, 310, 150, 330])

add_block('built-in/Product', [sys, '/','Flux Generation A/Prod1'])
set_param([sys, '/','Flux Generation A/Prod1'],
  'inputs', '2',
  'position', [505, 248, 515, 292])

add_block('built-in/Product', [sys, '/','Flux Generation A/Product3'])
set_param([sys, '/','Flux Generation A/Product3'],
  'hide name', 0,
  'inputs', '2',
  'position', [245, 218, 255, 262])

add_block('built-in/Product', [sys, '/','Flux Generation A/Product1'])
set_param([sys, '/','Flux Generation A/Product1'],
  'hide name', 0,
  'inputs', '2',
  'position', [245, 158, 255, 202])

add_block('built-in/Fcn', [sys, '/','Flux Generation A/using M3'])
set_param([sys, '/','Flux Generation A/using M3'],
  'position', [110, 220, 150, 240])

add_block('built-in/Integrator', [sys, '/','Flux Generation A/dM3'])
set_param([sys, '/','Flux Generation A/dM3'],
  'Initial', 'mmp3a',
  'position', [545, 490, 570, 520])
add_block('built-in/Sum', [sys,'/','Flux Generation A/Sum'])
set_param([sys,'/','Flux Generation A/Sum'],...
  'inputs','++++',...
  'position',[435,181,455,339])

add_block('built-in/Fcn', [sys,'/','Flux Generation A/1/det[A]'])
set_param([sys,'/','Flux Generation A/1/det[A]'],...
  'Drop Shadow',4,...
  +u[1]*u[3]*u[7]+u[1]*u[3]*u[5])',...
  'position',[110,375,150,395])

add_block('built-in/Fcn', [sys,'/','Flux Generation A/using M2'])
set_param([sys,'/','Flux Generation A/using M2'],...
  'Drop Shadow',4,...
  u[4]/u[3]^2)',...
  'position',[110,180,150,200])

add_block('built-in/Integrator', [sys,'/','Flux Generation A/dM2'])
set_param([sys,'/','Flux Generation A/dM2'],...
  'Initial','mmp2a',...
  'position',[535,253,560,287])

add_block('built-in/Outport', [sys,'/','Flux Generation A/Flux1'])
set_param([sys,'/','Flux Generation A/Flux1'],...
  'Port','1',...
  'position',[820,95,840,115])

add_block('built-in/Inport', [sys,'/','Flux Generation A/in_1'])
set_param([sys,'/','Flux Generation A/in_1'],...
  'Port','1',...
  'position',[15,375,35,395])

add_block('built-in/Outport', [sys,'/','Flux Generation A/Flux4'])
set_param([sys,'/','Flux Generation A/Flux4'],...
  'Port','4',...
  'position',[825,635,845,655])

add_block('built-in/Outport', [sys,'/','Flux Generation A/Flux2'])
set_param([sys,'/','Flux Generation A/Flux2'],...
  'Port','2',...
  'position',[820,270,840,290])

add_block('built-in/Outport', [sys,'/','Flux Generation A/Flux3'])
set_param([sys,'/','Flux Generation A/Flux3'],...
  'Port','3',...
  'position',[620,505,840,525])

add_block('built-in/Fcn', [sys,'/','Flux Generation A/C1'])
set_param([sys,'/','Flux Generation A/C1'],...
set_param([sys,'/','Flux Generation A/save3'],...
            'mat-name','flx3a',...
            'buffer','max',...
            'position',[820,545,850,565])

add_block('built-in/To Workspace',[sys,'/','Flux Generation A/save2'])
set_param([sys,'/','Flux Generation A/save2'],...
            'mat-name','flx2a',...
            'buffer','max',...
            'position',[820,310,850,330])

add_block('built-in/Fcn',[sys,'/','Flux Generation A/use dIa'])
set_param([sys,'/','Flux Generation A/use dIa'],...
           'Drop Shadow',4,...
           'position',[115,580,155,600])
add_line([sys,'/','Flux Generation A'],[780,105;815,105])
add_line([sys,'/','Flux Generation A'],[795,105;725,145;815,145])
add_line([sys,'/','Flux Generation A'],[575,505;585,505;585,420;190,420;190,250;240,250])
add_line([sys,'/','Flux Generation A'],[585,505;760,505])
add_line([sys,'/','Flux Generation A'],[585,505;585,620;605,620])
add_line([sys,'/','Flux Generation A'],[160,650;605,650])
add_line([sys,'/','Flux Generation A'],[565,270;575,270;575,110;205,110;205,170;240,170])
add_line([sys,'/','Flux Generation A'],[575,110;605,110])
add_line([sys,'/','Flux Generation A'],[155,80;605,80])
add_line([sys,'/','Flux Generation A'],[40,385;105,385])
add_line([sys,'/','Flux Generation A'],[70,385;70,190;105,190])
add_line([sys,'/','Flux Generation A'],[70,190;70,80;105,80])
add_line([sys,'/','Flux Generation A'],[190,420;190,525;245,525])
add_line([sys,'/','Flux Generation A'],[70,385;70,590;110,590])
add_line([sys,'/','Flux Generation A'],[70,465;110,465])
add_line([sys,'/','Flux Generation A'],[70,505;110,505])
add_line([sys,'/','Flux Generation A'],[70,555;110,555])
add_line([sys,'/','Flux Generation A'],[70,320;105,320])
add_line([sys,'/','Flux Generation A'],[70,280;105,280])
add_line([sys,'/','Flux Generation A'],[70,230;105,230])
add_line([sys,'/','Flux Generation A'],[160,590;290,590;330,575;430,575])
add_line([sys,'/','Flux Generation A'],[160,555;290,555;325,535;430,535])
add_line([sys,'/','Flux Generation A'],[265,515;280,515;300,495;430,495])
add_line([sys,'/','Flux Generation A'],[160,505;245,505])
add_line([sys,'/','Flux Generation A'],[160,465;245,465])
add_line([sys,'/','Flux Generation A'],[265,455;430,455])
add_line([sys,'/','Flux Generation A'],[155,385;480,385;480,280;500,280])
add_line([sys,'/','Flux Generation A'],[480,385;480,495;500,495])
add_line([sys,'/','Flux Generation A'],[460,515;500,515])
add_line([sys,'/','Flux Generation A'],[520,505;540,505])
add_line([sys,'/','Flux Generation A'],[155,320;430,320])
add_line([sys,'/','Flux Generation A'],[155,280;430,280])
add_line([sys,'/','Flux Generation A'],[460,260;500,260])
add_line([sys,'/','Flux Generation A'],[520,270;530,270])
add_line([sys,'/','Flux Generation A'],[155,190;240,190])
add_line([sys,'/','Flux Generation A'],[155,230;240,230])
add_line([sys,'/','Flux Generation A'],[260,240;430,240])
add_line([sys,'/','Flux Generation A'],[260,180;275,180;295,200;430,200])
add_line([sys,'/','Flux Generation A'],[745,115;760,115])
add_line([sys,'/','Flux Generation A'],[635,95;760,95])
add_line([sys,'/','Flux Generation A'],[745,290;760,290])
add_line([sys,'/','Flux Generation A'],[565,270;575,270;575,355;220,355;220,445;245,445])
add_line([sys,'/','Flux Generation A'],[575,270;760,270])
add_line([sys,'/','Flux Generation A'],[745,525;760,525])
add_line([sys,'/','Flux Generation A'],[745,655;760,655])
add_line([sys,'/','Flux Generation A'],[635,635;760,635])
add_line([sys,'/','Flux Generation A'],[70,590;70,690;675,690;675,655;695,655])
add_line([sys,'/','Flux Generation A'],[675,655;675,115;695,115])
add_line([sys,'/','Flux Generation A'],[675,525;695,525])
add_line([sys,'/','Flux Generation A'],[675,290;695,290])
add_line([sys,'/','Flux Generation A'],[780,645;820,645])
add_line([sys,'/','Flux Generation A'],[805,645;805,680;820,680])
add_line([sys,'/','Flux Generation A'],[780,515;815,515])
add_line([sys,'/','Flux Generation A'],[800,515;800,555;815,555])
add_line([sys,'/','Flux Generation A'],[780,280;815,280])
add_line([sys,'/','Flux Generation A'],[800,280;800,320;815,320])
add_line([sys,'/','Flux Generation A'],[70,650;110,650])

% Finished composite block 'Flux Generation A'.

set_param([sys,'/','Flux Generation A'],...
  'position',[415,117,470,273])

% Subsystem 'Force Generation A'.

new_system([sys,'/','Force Generation A'])
set_param([sys,'/','Force Generation A'],'Location',[385,40,812,564])

add_block('built-in/Fcn',[sys,'/','Force Generation A/(dR//dx)4a'])
set_param([sys,'/','Force Generation A/(dR//dx)4a'],...
  'Expr','-Ramp*alpha*sin(alpha*u[1]+beta2)',...
  'position',[205,380,245,400])

add_block('built-in/Fcn',[sys,'/','Force Generation A/(dR//dx)3a'])
set_param([sys,'/','Force Generation A/(dR//dx)3a'],...
  'Expr','-Ramp*alpha*sin(alpha*u[1]+beta1)',...
  'position',[210,295,250,315])

add_block('built-in/Fcn',[sys,'/','Force Generation A/(dR//dx)2a'])
set_param([sys,'/','Force Generation A/(dR//dx)2a'],...
  'Expr','-Ramp*alpha*sin(alpha*u[1]+beta2)',...
add_block('built-in/Fcn', [sys, '/','Force Generation A/(dR//dx)la'])
set_param([sys, '/','Force Generation A/(dR//dx)la'],
    'Expr', '-Ramp*alpha*sin(alpha*u[1]+betal)',
    'position', [205, 125, 245, 145])

add_block('built-in/Sum', [sys, '/','Force Generation A/Sum'])
set_param([sys, '/','Force Generation A/Sum'],
    'inputs', '++++',
    'position', [320, 103, 335, 452])

add_block('built-in/Fcn', [sys, '/','Force Generation A/(Flux^2)4'])
set_param([sys, '/','Force Generation A/(Flux^2)4'],
    'Expr', 'u[1]*u[1]/2',
    'position', [105, 405, 145, 425])

add_block('built-in/Product', [sys, '/','Force Generation A/Product3'])
set_param([sys, '/','Force Generation A/Product3'],
    'hide name', 0,
    'inputs', '2',
    'position', [270, 377, 285, 428])

add_block('built-in/Fcn', [sys, '/','Force Generation A/(Flux^2)3'])
set_param([sys, '/','Force Generation A/(Flux^2)3'],
    'Expr', 'u[1]*u[1]/2',
    'position', [105, 320, 145, 340])

add_block('built-in/Product', [sys, '/','Force Generation A/Product2'])
set_param([sys, '/','Force Generation A/Product2'],
    'hide name', 0,
    'inputs', '2',
    'position', [275, 292, 290, 343])

add_block('built-in/Fcn', [sys, '/','Force Generation A/(Flux^2)2'])
set_param([sys, '/','Force Generation A/(Flux^2)2'],
    'Expr', 'u[1]*u[1]/2',
    'position', [105, 235, 145, 255])

add_block('built-in/Product', [sys, '/','Force Generation A/Product1'])
set_param([sys, '/','Force Generation A/Product1'],
    'hide name', 0,
    'inputs', '2',
    'position', [275, 207, 290, 258])

add_block('built-in/Fcn', [sys, '/','Force Generation A/(Flux^2)1'])
set_param([sys, '/','Force Generation A/(Flux^2)1'],
    'Expr', 'u[1]*u[1]/2',
    'position', [105, 150, 145, 170])

add_block('built-in/Product', [sys, '/','Force Generation A/Product'])
set_param([sys, '/','Force Generation A/Product'],
    'hide name', 0,
    'inputs', '2',
    'position', [275, 122, 290, 173])
'mat-name', 'F4a', ...
'buffer', 'max', ...
'position', [360, 410, 400, 430]
add_line([sys, ' ', 'Force Generation A'], [150, 415; 265, 415])
add_line([sys, ' ', 'Force Generation A'], [250, 390; 265, 390])
add_line([sys, ' ', 'Force Generation A'], [255, 305; 270, 305])
add_line([sys, ' ', 'Force Generation A'], [150, 245; 270, 245])
add_line([sys, ' ', 'Force Generation A'], [255, 220; 270, 220])
add_line([sys, ' ', 'Force Generation A'], [150, 160; 270, 160])
add_line([sys, ' ', 'Force Generation A'], [250, 135; 270, 135])
add_line([sys, ' ', 'Force Generation A'], [45, 160; 100, 160])
add_line([sys, ' ', 'Force Generation A'], [45, 245; 100, 245])
add_line([sys, ' ', 'Force Generation A'], [340, 280; 360, 280])
add_line([sys, ' ', 'Force Generation A'], [45, 330; 100, 330])
add_line([sys, ' ', 'Force Generation A'], [45, 415; 100, 415])
add_line([sys, ' ', 'Force Generation A'], [150, 330; 270, 330])
add_line([sys, ' ', 'Force Generation A'], [45, 90; 175, 90; 175, 390; 200, 390])
add_line([sys, ' ', 'Force Generation A'], [175, 305; 205, 305])
add_line([sys, ' ', 'Force Generation A'], [175, 220; 205, 220])
add_line([sys, ' ', 'Force Generation A'], [175, 135; 200, 135])
add_line([sys, ' ', 'Force Generation A'], [295, 150; 315, 150])
add_line([sys, ' ', 'Force Generation A'], [305, 150; 305, 135; 355, 135])
add_line([sys, ' ', 'Force Generation A'], [290, 405; 315, 405])
add_line([sys, ' ', 'Force Generation A'], [305, 405; 305, 420; 355, 420])
add_line([sys, ' ', 'Force Generation A'], [295, 320; 315, 320])
add_line([sys, ' ', 'Force Generation A'], [305, 320; 305, 340; 355, 340])
add_line([sys, ' ', 'Force Generation A'], [295, 235; 315, 235])
add_line([sys, ' ', 'Force Generation A'], [305, 235; 305, 220; 355, 220])

% Finished composite block 'Force Generation A'.

set_param([sys, ' ', 'Force Generation A'], ...
'position', [540, 77, 620, 273])

% Subsystem 'Reluctance B'.

new_system([sys, ' ', 'Reluctance B'])
set_param([sys, ' ', 'Reluctance B'], 'Location', [35, 82, 541, 538])

add_block('built-in/Fcn', [sys, ' ', 'Reluctance B/R4b'])
set_param([sys, ' ', 'Reluctance B/R4b'], ...
'Expr', 'Roff-Ramp*cos(alpha*u[1]+beta4)', ...
'position', [180, 305, 220, 325])

add_block('built-in/Fcn', [sys, ' ', 'Reluctance B/R2b'])
set_param([sys, ' ', 'Reluctance B/R2b'], ...
'Expr', 'Roff-Ramp*cos(alpha*u[1]+beta4)', ...
'position', [180, 145, 220, 165])

add_block('built-in/Fcn', [sys, ' ', 'Reluctance B/R3b'])
set_param([sys, ' ', 'Reluctance B/R3b'], ...
'Expr', 'Roff-Ramp*cos(alpha*u[1]+beta3)', ...
'position', [180, 225, 220, 245])
% Finished composite block 'Reluctance B'.
set_param([sys,'/','Reluctance B'],
    'position',[290,342,345,493])

% Subsystem 'Flux Generation B'.
new_system([sys,'/','Flux Generation B'])
set_param([sys,'/','Flux Generation B'],'Location',[234,82,1163,801])

add_block('built-in/Sum',[sys,'/','Flux Generation B/M4'])
set_param([sys,'/','Flux Generation B/M4'],
    'inputs','+-',
    'position',[610,603,630,667])

add_block('built-in/Fcn',[sys,'/','Flux Generation B/N*Ib'])
set_param([sys,'/','Flux Generation B/N*Ib'],
    'Expr','N*u[10]',
    'position',[115,640,155,660])

add_block('built-in/Sum',[sys,'/','Flux Generation B/M1'])
set_param([sys,'/','Flux Generation B/M1'],
    'inputs','+-',
    'position',[610,63,630,127])

add_block('built-in/Fcn',[sys,'/','Flux Generation B/N*Ib'])
set_param([sys,'/','Flux Generation B/N*Ib'],
    'Expr','N*u[10]',
    'position',[110,70,150,90])

add_block('built-in/Fcn',[sys,'/','Flux Generation B/use Ib'])
set_param([sys,'/','Flux Generation B/use Ib'],
    'Drop Shadow',4,
    'position',[115,545,155,565])

180
add_block('built-in/Product', [sys, '/','Flux Generation B/Product6'])
set_param([sys, '/','Flux Generation B/Product6'],
  'hide name', 0,...
  'inputs', '2',...
  'position', [250, 493, 260, 537])

add_block('built-in/Product', [sys, '/','Flux Generation B/Product2'])
set_param([sys, '/','Flux Generation B/Product2'],
  'hide name', 0,...
  'inputs', '2',...
  'position', [250, 433, 260, 477])

add_block('built-in/Fcn', [sys, '/','Flux Generation B/use M2'])
set_param([sys, '/','Flux Generation B/use M2'],
  'Drop Shadow', 4,...
  'Expr', 'u(2)/u(1)^2 + u(4)/u(3)^2)/u(9)',...
  'position', [115, 455, 155, 475])

add_block('built-in/Fcn', [sys, '/','Flux Generation B/use M3'])
set_param([sys, '/','Flux Generation B/use M3'],
  'Drop Shadow', 4,...
  'Expr', '(1/u(1)+1/u(3)+1/u(9))*u(6)/u(5)^2+u(8)/u(7)^2)',...
  'position', [115, 490, 155, 495])

add_block('built-in/Product', [sys, '/','Flux Generation B/Prod2'])
set_param([sys, '/','Flux Generation B/Prod2'],
  'inputs', '2',...
  'position', [505, 483, 515, 527])

add_block('built-in/Sum', [sys, '/','Flux Generation B/Sum1'])
set_param([sys, '/','Flux Generation B/Sum1'],
  'inputs', '+:++',...
  'position', [435, 436, 455, 594])

add_block('built-in/Fcn', [sys, '/','Flux Generation B/using Ib'])
set_param([sys, '/','Flux Generation B/using Ib'],
  'Drop Shadow', 4,...
  'position', [100, 2, 0, 390])

add_block('built-in/Fcn', [sys, '/','Flux Generation B/using dIb'])
set_param([sys, '/','Flux Generation B/using dIb'],
  'Drop Shadow', 4,...
  'position', [100, 310, 150, 330])

add_block('built-in/Product', [sys, '/','Flux Generation B/Prod1'])
set_param([sys, '/','Flux Generation B/Prod1'],
  'inputs', '2',...
  'position', [505, 248, 515, 292])
add_block('built-in/Product', [sys, '/', 'Flux Generation B/Product3'])
set_param([sys, '/', 'Flux Generation B/Product3'],
  'hide name', 0,...
  'inputs', '2',...
  'position', [245, 218, 255, 262])

add_block('built-in/Product', [sys, '/', 'Flux Generation B/Product1'])
set_param([sys, '/', 'Flux Generation B/Product1'],
  'hide name', 0,...
  'inputs', '2',...
  'position', [245, 158, 255, 202])

add_block('built-in/Fcn', [sys, '/', 'Flux Generation B/using M3'])
set_param([sys, '/', 'Flux Generation B/using M3'],
  'Drop Shadow', 4,...
  'position', [110, 220, 150, 240])

add_block('built-in/Integrator', [sys, '/', 'Flux Generation B/dM3'])
set_param([sys, '/', 'Flux Generation B/dM3'],
  'Initial', 'mmf3b',...
  'position', [545, 485, 565, 521])

add_block('built-in/Sum', [sys, '/', 'Flux Generation B/Sum'])
set_param([sys, '/', 'Flux Generation B/Sum'],
  'inputs', '+++',...
  'position', [435, 181, 455, 339])

add_block('built-in/Fcn', [sys, '/', 'Flux Generation B/1//det[A]'])
set_param([sys, '/', 'Flux Generation B/1//det[A]'],
  'Drop Shadow', 4,...
  +u[1]*u[3]*u[7]+u[1]*u[3]*u[5])',
  'position', [110, 375, 150, 395])

add_block('built-in/Fcn', [sys, '/', 'Flux Generation B/using M2'])
set_param([sys, '/', 'Flux Generation B/using M2'],
  'Drop Shadow', 4,...
  u[4]/u[3]^2)',...
  'position', [110, 180, 150, 200])

add_block('built-in/Integrator', [sys, '/', 'Flux Generation B/dM2'])
set_param([sys, '/', 'Flux Generation B/dM2'],
  'Initial', 'mmf2b',...
  'position', [535, 252, 555, 288])

add_block('built-in/Outport', [sys, '/', 'Flux Generation B/Flux1'])
set_param([sys, '/', 'Flux Generation B/Flux1'],
  'Port', '1',...
  'position', [820, 95, 840, 115])
add_block('built-in/Inport', [sys, '/', 'Flux Generation B/in_1'])
set_param([sys, '/', 'Flux Generation B/in_1'],
  'Port', '1',
  'position', [15, 375, 35, 395])

add_block('built-in/Outport', [sys, '/', 'Flux Generation B/Flux4'])
set_param([sys, '/', 'Flux Generation B/Flux4'],
  'Port', '4',
  'position', [825, 635, 845, 655])

add_block('built-in/Outport', [sys, '/', 'Flux Generation B/Flux2'])
set_param([sys, '/', 'Flux Generation B/Flux2'],
  'Port', '2',
  'position', [820, 270, 840, 290])

add_block('built-in/Outport', [sys, '/', 'Flux Generation B/Flux3'])
set_param([sys, '/', 'Flux Generation B/Flux3'],
  'Port', '3',
  'position', [820, 505, 840, 525])

add_block('built-in/Fcn', [sys, '/', 'Flux Generation B/C1'])
set_param([sys, '/', 'Flux Generation B/C1'],
  'Expr', '1/u[1]',
  'position', [700, 105, 740, 125])

add_block('built-in/Product', [sys, '/', 'Flux Generation B/Prod3'])
set_param([sys, '/', 'Flux Generation B/Prod3'],
  'hide name', 0,
  'inputs', '2',
  'position', [765, 83, 775, 127])

add_block('built-in/Fcn', [sys, '/', 'Flux Generation B/C2'])
set_param([sys, '/', 'Flux Generation B/C2'],
  'Expr', '1/u[3]',
  'position', [700, 280, 740, 300])

add_block('built-in/Product', [sys, '/', 'Flux Generation B/Prod4'])
set_param([sys, '/', 'Flux Generation B/Prod4'],
  'hide name', 0,
  'inputs', '2',
  'position', [765, 258, 775, 302])

add_block('built-in/Fcn', [sys, '/', 'Flux Generation B/C3'])
set_param([sys, '/', 'Flux Generation B/C3'],
  'Expr', '1/u[5]',
  'position', [700, 515, 740, 535])

add_block('built-in/Product', [sys, '/', 'Flux Generation B/Prod5'])
set_param([sys, '/', 'Flux Generation B/Prod5'],
  'hide name', 0,
  'inputs', '2',
  'position', [765, 493, 775, 537])

add_block('built-in/Fcn', [sys, '/', 'Flux Generation B/C4'])
set_param([sys, '/', 'Flux Generation B/C4'],...
add_line([sys,'/','Flux Generation B'],[70,465;110,465])
add_line([sys,'/','Flux Generation B'],[70,505;110,505])
add_line([sys,'/','Flux Generation B'],[70,555;110,555])
add_line([sys,'/','Flux Generation B'],[70,320;105,320])
add_line([sys,'/','Flux Generation B'],[70,280;105,280])
add_line([sys,'/','Flux Generation B'],[70,230;105,230])
add_line([sys,'/','Flux Generation B'],[160,590;290,590;330,575;430,575])
add_line([sys,'/','Flux Generation B'],[160,555;290,555;325,535;430,535])
add_line([sys,'/','Flux Generation B'],[265,515;280,515;300,495;430,495])
add_line([sys,'/','Flux Generation B'],[160,505;245,505])
add_line([sys,'/','Flux Generation B'],[160,465;245,465])
add_line([sys,'/','Flux Generation B'],[265,455;430,455])
add_line([sys,'/','Flux Generation B'],[155,385;480,385;480,280;500,280])
add_line([sys,'/','Flux Generation B'],[480,385;480,495;500,495])
add_line([sys,'/','Flux Generation B'],[460,515;500,515])
add_line([sys,'/','Flux Generation B'],[520,505;540,505])
add_line([sys,'/','Flux Generation B'],[155,320;430,320])
add_line([sys,'/','Flux Generation B'],[155,280;430,280])
add_line([sys,'/','Flux Generation B'],[460,260;500,260])
add_line([sys,'/','Flux Generation B'],[520,270;530,270])
add_line([sys,'/','Flux Generation B'],[155,190;240,190])
add_line([sys,'/','Flux Generation B'],[155,230;240,230])
add_line([sys,'/','Flux Generation B'],[260,240;430,240])
add_line([sys,'/','Flux Generation B'],[260,180;275,180;295,200;430,200])
add_line([sys,'/','Flux Generation B'],[745,115;760,115])
add_line([sys,'/','Flux Generation B'],[635,95;760,95])
add_line([sys,'/','Flux Generation B'],[745,290;760,290])
add_line([sys,'/','Flux Generation B'],[560,270;575,270;575,355;220,355;220,445;245,445])
add_line([sys,'/','Flux Generation B'],[575,270;760,270])
add_line([sys,'/','Flux Generation B'],[745,525;760,525])
add_line([sys,'/','Flux Generation B'],[745,655;760,655])
add_line([sys,'/','Flux Generation B'],[635,635;760,635])
add_line([sys,'/','Flux Generation B'],[70,590;70,690;675,690;675,655;695,655])
add_line([sys,'/','Flux Generation B'],[675,655;675,115;695,115])
add_line([sys,'/','Flux Generation B'],[675,525;695,525])
add_line([sys,'/','Flux Generation B'],[675,290;695,290])
add_line([sys,'/','Flux Generation B'],[780,645;820,645])
add_line([sys,'/','Flux Generation B'],[680,645;805,680;820,680])
add_line([sys,'/','Flux Generation B'],[780,515;815,515])
add_line([sys,'/','Flux Generation B'],[800,515;800,555;815,555])
add_line([sys,'/','Flux Generation B'],[780,280;815,280])
add_line([sys,'/','Flux Generation B'],[800,280;800,320;815,320])
add_line([sys,'/','Flux Generation B'],[680,650;70,650;110,650])

% Finished composite block 'Flux Generation B'.

set_param([sys,'/','Flux Generation B'],...
add_block('built-in/Fcn', [sys, '/', 'Force Generation B/(Flux^2)1'])
set_param([sys, '/', 'Force Generation B/(Flux^2)1'),...
  'Expr', 'u[1]*u[1]/2',...
  'position', [105, 235, 145, 255])

add_block('built-in/Product', [sys, '/', 'Force Generation B/Product1'])
set_param([sys, '/', 'Force Generation B/Product1'),...
  'hide name', 0, ...
  'inputs', '2', ...
  'position', [275, 207, 290, 258])

add_block('built-in/Fcn', [sys, '/', 'Force Generation B/(Flux^2)1'])
set_param([sys, '/', 'Force Generation B/(Flux^2)1'),...
  'Expr', 'u[1]*u[1]/2',...
  'position', [105, 150, 145, 170])

add_block('built-in/Product', [sys, '/', 'Force Generation B/Product'])
set_param([sys, '/', 'Force Generation B/Product'),...
  'hide name', 0, ...
  'inputs', '2', ...
  'position', [275, 122, 290, 170])

add_block('built-in/Inport', [sys, '/', 'Force Generation B/in_1'])
set_param([sys, '/', 'Force Generation B/in_1'),...
  'Port', '2', ...
  'position', [20, 150, 40, 170])

add_block('built-in/Inport', [sys, '/', 'Force Generation B/in_2'])
set_param([sys, '/', 'Force Generation B/in_2'),...
  'Port', '3', ...
  'position', [20, 235, 40, 255])

add_block('built-in/Outport', [sys, '/', 'Force Generation B/Fb'])
set_param([sys, '/', 'Force Generation B/Fb'),...
  'Port', '1', ...
  'position', [365, 270, 385, 290])

add_block('built-in/Inport', [sys, '/', 'Force Generation B/in_3'])
set_param([sys, '/', 'Force Generation B/in_3'),...
  'Port', '4', ...
  'position', [20, 320, 40, 340])

add_block('built-in/Inport', [sys, '/', 'Force Generation B/in_4'])
set_param([sys, '/', 'Force Generation B/in_4'),...
  'Port', '1', ...
  'position', [20, 80, 40, 100])

add_block('built-in/Inport', [sys, '/', 'Force Generation B/in_5'])
set_param([sys, '/', 'Force Generation B/in_5'),...
  'Port', '5', ...
  'position', [20, 405, 40, 425])

add_block('built-in/To Workspace', [sys, '/', 'Force Generation B/Save F2b
')
set_param([sys,'/','Force Generation B/Save F2b '],
  'mat-name','F2b',
  'buffer','max',
  'position',[355,205,395,225])

add_block('built-in/To Workspace',[sys,'/','Force Generation B/Save F1b'])
set_param([sys,'/','Force Generation B/Save F1b '],
  'mat-name','F1b',
  'buffer','max',
  'position',[355,120,395,140])

add_block('built-in/To Workspace',[sys,'/','Force Generation B/Save F4b'])
set_param([sys,'/','Force Generation B/Save F4b '],
  'mat-name','F4b',
  'buffer','max',
  'position',[360,415,400,435])

add_block('built-in/To Workspace',[sys,'/','Force Generation B/Save F3b'])
set_param([sys,'/','Force Generation B/Save F3b '],
  'mat-name','F3b',
  'buffer','max',
  'position',[360,335,400,355])

add_line([sys,'/','Force Generation B '],[150,415;265,415])
add_line([sys,'/','Force Generation B '],[250,390;265,390])
add_line([sys,'/','Force Generation B '],[250,305;270,305])
add_line([sys,'/','Force Generation B '],[150,245;270,245])
add_line([sys,'/','Force Generation B '],[250,220;270,220])
add_line([sys,'/','Force Generation B '],[150,160;270,160])
add_line([sys,'/','Force Generation B '],[255,135;270,135])
add_line([sys,'/','Force Generation B '],[45,160;100,160])
add_line([sys,'/','Force Generation B '],[45,245;100,245])
add_line([sys,'/','Force Generation B '],[340,280;360,280])
add_line([sys,'/','Force Generation B '],[45,330;100,330])
add_line([sys,'/','Force Generation B '],[45,415;100,415])
add_line([sys,'/','Force Generation B '],[150,330;270,330])
add_line([sys,'/','Force Generation B '],[45,90;175,90;175,390;200,390])
add_line([sys,'/','Force Generation B '],[175,305;200,305])
add_line([sys,'/','Force Generation B '],[175,220;200,220])
add_line([sys,'/','Force Generation B '],[175,135;205,135])
add_line([sys,'/','Force Generation B '],[295,235;315,235])
add_line([sys,'/','Force Generation B '],[305,235;305,215;350,215])
add_line([sys,'/','Force Generation B '],[295,150;315,150])
add_line([sys,'/','Force Generation B '],[305,150;305,130;350,130])
add_line([sys,'/','Force Generation B '],[290,405;315,405])
add_line([sys,'/','Force Generation B '],[305,405;305,425;355,425])
add_line([sys,'/','Force Generation B '],[295,320;315,320])
add_line([sys,'/','Force Generation B '],[305,320;305,345;355,345])

% Finished composite block 'Force Generation B'.

188
set_param(sys,'/','Force Generation B'),...
    'position',[540,302,620,498])

add_block('built-in/Sum',[sys,'/','Sum'])
set_param([sys,'/','Sum'],...
    'inputs','++',...
    'position',[705,274,730,316])

add_block('built-in/To Workspace',[sys,'/','Position'])
set_param([sys,'/','Position'],...
    'mat-name','X',...
    'buffer','max',...
    'position',[555,27,585,53])

add_block('built-in/To Workspace',[sys,'/','FORCEa'])
set_param([sys,'/','FORCEa'],...
    'mat-name','Fa',...
    'buffer','max',...
    'position',[685,160,715,190])

add_block('built-in/To Workspace',[sys,'/','FORCEb'])
set_param([sys,'/','FORCEb'],...
    'mat-name','Fb',...
    'buffer','max',...
    'position',[700,385,730,415])

add_block('built-in/Fcn',[sys,'/','Ia - COS1'])
set_param([sys,'/','Ia - COS1'],...
    'Expr','u[1]/10',...
    'position',[180,65,220,85])
add_line(sys,[735,295;760,295])
add_line(sys,[350,420;410,420])
add_line(sys,[350,195;410,195])
add_line(sys,[475,255;535,255])
add_line(sys,[475,215;535,215])
add_line(sys,[475,175;535,175])
add_line(sys,[475,135;535,135])
add_line(sys,[70,230;175,230])
add_line(sys,[95,230;95,455;175,455])
add_line(sys,[475,480;535,480])
add_line(sys,[475,440;535,440])
add_line(sys,[475,400;535,400])
add_line(sys,[475,360;535,360])
add_line(sys,[225,455;285,455])
add_line(sys,[250,455;250,410;225,410])
add_line(sys,[225,230;285,230])
add_line(sys,[250,230;250,185;225,185])
add_line(sys,[225,75;505,75;505,95;535,95])
add_line(sys,[270,75;270,155;285,155])
add_line(sys,[270,155;270,380;285,380])
add_line(sys,[270,320;535,320])
add_line(sys,[625,175;660,175;660,285;700,285])
add_line(sys,[660,175;680,175])
add_line(sys,[625,400;660,400;660,305;700,305])
add_line(sys,[660,400;695,400])
add_line(sys,[80,115;175,115])
add_line(sys,[150,115;150,75;175,75])
add_line(sys,[505,75;505,40;550,40])

% Return any arguments.
if (nargin | nargout)
    % Must use feval here to access system in memory
    if (nargin > 3)
        if (flag == 0)
            eval(['[ret,x0,xstr]=',sys,'(t,x,u,flag);'])
        else
            eval(['ret =', sys,'(t,x,u,flag);'])
        end
    else
        [ret,x0,str] = feval(sys);
    end
end
Appendix J

Listing of Data Collection Code

The code use to collect the data using the Metrabyte DAS20 board.
/*
Through the use of the DAS20 data acquisition board, this program is used to collect data from any analog signal. Sampling frequency can be set via an external oscillator connected to CNTRL SRC pin. (Maximum sampling frequency is 100 kHz, as specified by the manufacturer of the board.) The data is then stored in the specified filename. Plots can be generated with MATLAB.

The code is self-explanatory, with the exception of the modes specified when using the DAS20 library files. Refer to manual.

This program was created in combination by: Yuri Bendana, Manuel Madrigal, & Pablo Rodriguez, working under the supervision of KYT, Laboratory for Manufacturing and Productivity.
*/

#include <stdio.h>
#include <alloc.h>
#include <conio.h>
#include <math.h>
#include <dos.h>
#include <float.h>
#include <string.h>
#include <stdlib.h>

#define ASIZE 5120
#define FREQUENCY 1000.0
#define TOTALCONV 2000
#define MOPRO (1<<4)
#define OFF 0
#define ONN 1

int DAS20(int mode, int data[]);
void checkit(int mode, int flag);
void stepit(void);
float volt;

main()
{
    int flag, mode, noc;
    int j, data[10], darray[ASIZE], filenum;
    int far *buffer;
    char filename[30], filenumstr[5];
    FILE *fp;

    clrscr();

    noc = TOTALCONV;
    printf("\n\nHit any key to start ...\n");
    while(!kbhit()) {}
/* -------------------Initialize DAS20 Board------------------- */
mode = 0;
data[0] = 0x300;    /* Base Address */
data[1] = 5;        /* Interrupt Level */
data[2] = 1;        /* DMA Level */
flag = DAS20(mode, data);
checkit(mode, flag);

mode=15;
data[0] = MOPRO;
flag = DAS20(mode, data);
checkit(mode, flag);

/* -------------------Allocate buffer------------------- */

printf("Available space = %lu\n", farcoreleft());

if ((buffer = (int far *)farmalloc(50000L)) == NULL)
{
    printf("\n\ntHolly cow, Batman, we can't allocate buffer\n");
    printf("\n\n\nt Hit any key to escape...\n");
    getch();
    return(0);
}

/* -------------------Load DMA Que------------------- */
mode = 1;
data[0] = 0;        /* Channel number */
data[1] = 3;        /* Gain Range */
data[2] = 1;        /* End of queue */
flag = DAS20(mode, data);
checkit(mode, flag);

/* -------------------Start Collecting Data------------------- */
mode = 6;
data[0] = (noc<ASIZE)?noc:ASIZE; /* # of Conversions */
data[1] = FP_SEG(buffer);        /* Address of Buffer */
data[2] = 0;                    /* 0 = external clock */
    data[3] = 1;                    /* 1 = single cycle */
flag = DAS20(mode, data);
checkit(mode, flag);

/* stepit(); */

/* -------------------Monitor Status------------------- */
data[1] = 1;
while(data[1] != 0)
{
    mode = 12;
    flag = DAS20(mode, data);
    checkit(mode, flag);
    printf("Conversion # = %u\n", data[2]);
}
/* ----------------Transfer Data to Array---------------- */
mode = 13;
data[0] = (noc<ASIZE)?noc:ASIZE;
data[1] = FP_SEG(buffer);
data[2] = 0;
data[3] = FP_OFF(darray);
data[4] = 0;
data[5] = 1;
data[6] = 0;
flag = DAS20(mode,data);
checkit(mode,flag);

/* ----------------Print Data to Screen/File---------------- */
fp = fopen("c:\Prod\Data\param.dat","r");
fgets(filenumstr, 5,fp);
close(fp);
filenum = atoi(filenumstr);
filenum++;
sprintf(filename, "%s%03s", "c:\Prod\Data\run",filenum, ".dat");
fp = fopen(filename,"w");
printf("Writing to file %s\n",filename);
for(j =0; j<noc; j++)
{
    volt = (darray[j]*10.0)/4096.0;
    fprintf(fp,"\n%f ",volt);
}
close(fp);
fp = fopen("c:\Prod\Data\param.dat","w");
sprintf(filenumstr, "%i", filenum);
fprintf(fp,"%s\n",filenumstr);
fprintf(fp,"%i\n",noc);
fprintf(fp,"%f\n",FREQUENCY);
close(fp);

/* Free up the memory we allocated and exit the program */
farfree(buffer);
return(0);
}
void checkit(int mode, int flag)
{
    if (flag != 0)
    {
        printf("\n\nERROR DETECTED, MODE: %d, FLAG: %d", mode, flag);
        exit(0);
    }
}

void stepit(void)
{
    int flag, mode, data[10], use;

    mode = 15; /* Turn on LED to indicate MOPRO is on */
    use = MOPRO;
    data[0] = use;
    flag = DAS20(mode, data);
    checkit(mode, flag);

    mode = 15; /* Get ready to take a step */
    use = use | (ONN<<0) | (OFF<<1) | (ONN<<3);
    data[0] = use;
    flag = DAS20(mode, data);
    checkit(mode, flag);

    mode = 15; /* Turn motor power on */
    use = use & ~(ONN<<2);
    data[0] = use;
    flag = DAS20(mode, data);
    checkit(mode, flag);

    mode = 14; /* Check to see if everything is ok */
    flag = DAS20(mode, data);
    data[0] &= 1;
    printf("\n\nCHECKING TO See IF EVERYTHING IS OKAY: %d", data[0]);
    if (data[0] == 1)
    {
        printf("\n\n\nERROR HAS BEEN DETECTED IN MOTOR POWER!");
        exit(0);
    }

    mode = 15; /* Make the motor take a step */
    use = use & (OFF<<0);
    data[0] = use;
    flag = DAS20(mode, data);
    checkit(mode, flag);

    mode = 15;
    use = use & ~MOPRO;
    data[0] = use;
    flag = DAS20(mode, data);
    checkit(mode, flag);
}
Bibliography


