MODELING SWITCHING NETWORKS USING BOND GRAPH TECHNIQUE

By

Ali Shiva

A Thesis Submitted to the Faculty of the

DEPARTMENT OF AEROSPACE AND MECHANICAL ENGINEERING

In Partial Fulfillment of the Requirements for the Degree of

MASTERS OF SCIENCE

WITH A MAJOR IN MECHANICAL ENGINEERING

In the Graduate College

University of Arizona

Tucson, Arizona

2 0 0 4

STATEMENT BY AUTHOR

This thesis has been submitted in partial fulfillment of requirements for an advanced degree at The University of Arizona and is deposited in the University Library to be made available to borrowers under rules of the Library.

Brief quotations from this thesis are allowable without special permission, provided that accurate acknowledgment of source is made. Requests for permission for extended quotation from or reproduction of this manuscript in whole or in part may be granted by the head of the major department or the Dean of the Graduate College when in his or her judgment the proposed use of the material is in the interests of scholarship. In all other instances, however, permission must be obtained from the author.

SIGNED: _____

APPROVAL BY THESIS DIRECTOR

This thesis has been approved on the date shown below:

_____ _____

Parviz E. Nikravesh
Professor

Date

Acknowledgments

"Wake up; you have an email from the University of Arizona." It is far from possible to squeeze in one page all the joy and excite I experienced from the time when my brother woke me up with this sentence, till now that I see myself typing these "final" phrases which bring this thesis and consequently my Masters program to a closure; marking the end of two years of my life in Arizona. It was fun. This single page is for paying tribute to that.

I remember Dr. Cellier once saying in ECE542 that he will teach Jury's stability test not only for its computational efficiency, but also because "he was a nice guy". Indeed, Dr. Cellier himself is one of the nicest men I've ever met. I was inspired by his knowledge, dedication, and above all, his exceptionally pleasant attitude. He redefined the meaning of "patience". ECE542 was fun, ECE501 was astounding. Students like myself with the most primitive questions knocking on the door of his office were always welcome and treated with respect, attention, and a fatherly smile.

Dr. Nikravesh supported me academically as well as personally, which I am truly grateful of. The systematic way which AME553 was presented made it an exciting and memorable journey in the world of computational dynamics, and I learned a lot out of it. Recalling those thrilling moments of working on the computer project is ever filling with joy and delight.

Due to my devotion to Controls, at course registration time I approached AME550 with a bit of reluctance, only to realize before long that it was indeed a "must" course and I couldn't have missed it for anything. I learned, and enjoyed learning it. Thanks to Dr. Arabyan.

I also want to thank Dr. Wang from the SIE dept., for his golden sentence_finally engraved in my head after an interesting series of debates_"…you don't need feedback."

As for the folks I met at CAEL: Lin, Thorsten, Jens, Sudareshan, Xiang, Michael, Ananth, and Juan, Adrijan, Chris, Michael and Aijaz, which most of them by the time of defending this thesis have left U of A to seek their future in life, I wish all the best. What better an appreciation than the pleasant recollections which will linger on. From the night of working on AME555's final project to the night of working on AME550's final project, from my first class with Dr. Ganapol to my last class with Dr. Arabyan, all will be filed in the "U of A archives" of my mind.

While browsing through other theses, I came across one in which the author had written a "thank you" sentence for the custodial staff too. Quite a few of these folks may never have the luxury to take classes at the U. I thank them, for their hard work, and their sacrifice.

For someone like me who always loved school, U of A and Tucson was a fascinating experience; from the very beginning till the final moments of my stay. Once again the joy of coming across classrooms, text books and lecture notes, in addition to the lovely campus, the student union, the bookstore, the hanging bridges of AME, the friendly little cubicle in CAEL, the "TA"ing, the X stroke on the calendar for each day, the last week of classes, the pedaling, the cooking, the mosque, the nightly strolls, the sunsets, my peaceful little rooms at 1177 East Lester St., … all and all shaped a special snapshot of time in my life. Special thanks go to Dr. Raza and Mrs. Zeba Hashim, Dr. Behrouz Dehdashti, and Mrs. Esther Bryan, who all made it possible.

Deepest thanks to my father, mother and brothers; for their financial and moral support, the life-saving phone calls, and honestly every single thing in my life. Mother, Father, I thank you.

My wife demonstrated unconditional love and amazing patience especially throughout this seemingly-endless departure. I've already learned from her, and look forward to learning more. Whenever I hear a Nokia cell phone ring, I'm reminded of an all too familiar tingle in my heart.

My heartfelt blessings and prayers to Karnough and his maps. To him I owe the existence of this thesis, and consequently much more. Wherever it is, may his soul rest in peace.

And, I humbly thank the "Teacher", who ceaselessly whispers in our heart and soul.

To

The Awaited.

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

In this thesis a novel approach to modeling a switching electrical/mechanical network is introduced. The Bond Graph technique, the modeling tool in this thesis, is explained and some examples are presented. The switching element and its causality are discussed. The gearbox system is taken as a mechanical switching network and after a brief description is modeled into a logical switching network by means of the Karnough diagram, and thereafter translated into Bond Graph representation. Further on, the derived algorithm is applied to model a pilot ejection scenario. Additional issues regarding existing analogies between electrical and mechanical systems are discussed. After a short introduction to the modeling software DYMOLA™, the generated models are simulated and the validity of the results is analyzed.

PREFACE: THESIS OUTLINE

This thesis is subdivided into five chapters:

Chapter 1 is the introductory chapter, with two sections. The first section explains the problem; switching networks in Bond graphs, its significance in modeling, and a brief sketch of how it is tackled. The second section contains a passing survey on the existing literature regarding this topic and shortly describes selected previous research by others. Chapter 2 contains the technical background; divided in two sections: first, Bond graph modeling is concisely explained since it is assumed that the typical reader with a mechanical engineering background is unlikely to be familiar with this method; however for a more in depth study, adequate references are pointed out to the reader. The second part briefly explains the essence of Karnough maps; the tool used to attain an electric switching diagram from logical expressions. Chapter 3 describes in detail what was done to solve the problem, consisting of two main parts. In the first part, the automotive gearbox is introduced as a mechanical switching network and thereafter modeled into Bond graphs, describing the difficulties encountered and the solutions proposed. The second part illustrates modeling a simple pilot ejection scenario based upon the technique obtained from the previous section, including an exploration of further details regarding the analogy of mechanical and electrical systems. In chapter 4, the derived models are simulated using Dymola^{TM}, while briefly exposing the reader to the basics of the software structure. Chapter 5 summarizes the work done, emphasizing some of the benefits and indicating some possible detriments, along with potential future research. A complete list of references used is included at the end, wrapping up this thesis.

1. *INTRODUCTION*

1.1. *PROBLEM STATEMENT*

In one word: This thesis is about “modeling”. The goal is to develop a systematic algorithm for modeling switching networks (with an emphasis on mechanical/electrical systems) with Bond graphs.

“The Bond graph technique, originated by Paynter [24], presents a tool for continuous system modeling in a graphical sense, by generalizing the voltage/current relationship of electrical circuits to other physical phenomena such as mechanical and chemical interactions. Bond graph models offer a direct description of differential equations; however the dependencies between the system variables are approached indirectly. For example, current can be considered to cause voltage across a resistor or the other way around, depending on whether the energy is coming from a current or a voltage source. By fixing which variables are given (input) and which variables of the systems should be computed (state and output); a particular differential equation system can be derived from a set of possibilities.” [27]

Bond graphs do take into consideration the causal relation between the system elements. This feature often provides the modeler with a deeper insight regarding the underlying physical phenomena. Of interest is the issue of modeling networks which are set to operate in dissimilar consecutive stages, thus “switching”. The act of switching itself results in a discontinuous operational behavior demanded from the system by its surroundings: a discrete event takes place that changes the structure of the model, as a

consequence of which the system behavior changes. It should be noted however that if a mechanical component is modeled detailed enough, there usually will be no discontinuities in the system. Discontinuities appear when the modeler neglects some "fast" dynamics, in order to reduce modeling effort and simulation time.

In Bond graph modeling, traditionally there have been two methods for dealing with switching occurrences: either using non-ideal switches instead of ideal switches, or generating a separate Bond graphic model for either of the switch positions for the entire system, resulting in an extensive number of independent models. We employ a novel method to tackle this issue, by using digital logic as the backbone algorithm. Further on, taking care of residual flows/efforts, in addition to important considerations regarding energy storage elements subject to sudden change in flow/effort variables are discussed. We first describe the switching plan in terms of logical phrases. In the next step, these logical phrases are mapped into an electrical switching network by means of digital logic tools, namely the Karnough maps or the Quine-McCluskey method for problems of greater dimensions. Thereafter, the acquired circuit diagram of the switching scheme is translated into Bond graphs. However the work is not complete since the functionality of an ideal switch, i.e. attaining two different positions, can result in certain complexities in the network such as causality conflicts and undetermined values for efforts/flows. This obscurity was dealt with by employing additional switches which are activated inversely relative to the main switches, guiding the undetermined flows/efforts towards the ground hence generating a logically proper final model which can be simulated.

This method provides the modeler with a systematic algorithm for modeling

discontinuities in Bond graphs, as a compelling alternative to constructing case dependant models by unavoidable use of heuristics, or even being forced to undesirably deal with separate models for different stages of operation with the consequence of obvious complexity growth as the model inflates.

As an outcome of exploring this phenomenon with Bond graphs, interesting conclusions are drawn regarding modeling mechanical and electrical systems along with some fundamental concepts with respect to existing analogies between the two domains.

1.2. *SELECTED REVIEW OF PRECEDING RESEARCH*

As mentioned earlier, as of the time of writing this thesis there does not exist a systematic algorithm for constructing Bond graphic switching networks in the literature. Glaser et al. in [15] discuss modeling a switching power converter using Dymola. In this paper, the authors draw a comparison between Dymola and other simulation platforms and point out the benefits of the former, namely its ability to handle time and state events in an object-oriented manner, in addition to its capability in formulating models as differential algebraic equations. The authors mention the difficulties regarding simulating switching power converters, which operate by using switches to change the configuration of inductor-capacitor networks. Therefore each switching event in an ideal case results in a discontinuity. Since power converter simulations can run for thousands of cycles, the lack of explicit event handling in a simulating software could cause problems such as unnecessary simulation length and generation of large amounts of superfluous data.

Thereafter, the switching converters are modeled with linear electric components and the ideal switch is treated as a controlled resistor which has either infinite or zero resistance depending on the switch being open or closed. The authors indicate the causality dilemma due to the bi-positional nature of an ideal switch and propose the use of non ideal switches instead, and avoid stiffness in the system by running the converter in "continuous conduction mode" meaning that there is no interval during which all of the switches are off, however they mention the difficulties of running in "discontinuous conduction mode". The converter model is constructed and the model is simulated with Dymola and SPICE3 which demonstrate the consistency of the results.

In [11] Elmqvist et al. discuss basic mechanisms in object-oriented modeling of hybrid systems (being a mixture of continuous and discrete components) in Dymola. The causality in Newtonian laws is mentioned as numerical artifacts which are rendered into assignment statements by the simulating software. The authors mention that "events" are low level elements useful for the control of integrators but not appropriate for comfortable definition of hybrid models. Instead, Dymola offers high level elements for the definition of discontinuities and instantaneous equations controlled by Boolean variables and expressions. Instantaneous equations are conditional equations evaluated only at the instant when their Boolean condition becomes true. The switch element and its causality are introduced and it is pointed out that a switch must always be placed in such a fashion that both of its causalities must be compatible with the circuit environment. "Dry friction" is mentioned as a variable structure model (i.e. with altering causality) which differs from the electrical switch in the sense that the electrical switch,

switches between two different equations whereas the “friction element” adds or removes equations or variables depending on the value for the velocity. Eventually the authors construct a state transition diagram for the five regions of the friction model (stuck; start forward; forward; start backward; backward), and conclude that discontinuities and instantaneous equations controlled by Boolean variables which are based on explicit definitions of events and event actions are superior to the traditional “event-oriented” approach, by sorting all equations together generating automatic synchronization of all events.

Cellier et al. in [6] address the issue of modeling variable structure systems in Bond graphs; a new approach in the field, given that Bond graphs where traditionally used to describe continuous-time physical processes with fixed structures. This paper claims that contrary to the common belief at that time, the introduction of causality strokes in the Bond graph literature was not only a victory but a major setback, since fixed causalities make it impossible to model variable structures. The authors exploit acausal Bond graphs to tackle this issue. The ideal switch and its equations are discussed; showing a “division by zero” for flow or effort when a switch with predetermined causality is open or closed respectively, stating that a switch must be placed in an “algebraic loop” in order to allow variable causality for the switch. Change of model order due to switching is described, and the derivative of flow is set to zero in the switch equation when the switch is open to preserve the switch equation inside an algebraic loop. Due to difficulties associated with ideal switches, it is mentioned that using non-ideal switches instead has been the major preference of researchers, with the drawback of creating artificially stiff systems; a

problem which the authors mention to endure.

In [17] Kim and Dan Cho develop an automatic transmission model for vehicle control in Bond graphs, mentioning that improving both "shift feel" and "fuel efficiency" is a significant aspect in automatic transmissions. After a brief explanation of the mechanism, the kinematic and dynamic constraints and equations are then developed under two main categories: upshift models and downshift models. The upshift model is partitioned into four sections: shift from 1^{st} to 2^{nd}, from 2^{nd} to 3^{rd}, from 3^{rd} to 4^{th}, and reverse gear. The downshift models are derived for three states: 4^{th} to 3^{rd}, 3^{rd} to 2^{nd}, and 2^{nd} to 1^{st}. For each shifting phase two models have been derived: the torque phase model and the speed phase model. However, the obtained Bond graph models are simulated separately and the authors do not integrate them into one overall model, hence the paper ends with a set of independent models of a physically intact gearbox.

Pastravanu in [23] employs Bond graphs with switching elements for quantitative analysis of hybrid dynamical systems; addressing a class of hybrid systems which is linear time-invariant with constant or harmonic effort/flow sources. The control action is based on discrete events, and the output vector comprises continuous-time signals. The author formulates the global and local state vectors and the augmented state equation, and then constructs a closed-form expression for an arbitrary sequence of modes with two different subsequences: time instants corresponding to the occurrence of either external or internal discrete events. Presented as an example is the simulation of a circuit containing 6 diodes, 3 effort sources, 2 transistors, an inductor and a capacitor; where the control action is responsible for switching the transistors, the capacitor loading current is

the output and the communications of the diodes are the uncontrolled discrete events. This circuit is modeled with Bond graphs resulting in four distinct Bond graph topologies. The subject attended here however does not challenge the problem of causality conflicts in switching Bond graphs or does not propose how to generate a topologically correct switching network.

In [7] Coudert et al. implement Bond graph modeling in designing an automatic gearbox, with an emphasis on reducing the peak of pressure variation on output torque during a gear shift. In this paper two models are produced: a complete one including the entire driveline following the gearbox, and a simplified one which according to the authors can be used for automatic control in further studies. Here the simple model is used to test different pressure modulations in order to reduce the jerk felt by the passengers in acceleration. In the complete model, the engine block, torque converter, gearbox, decision block (containing the shift schedule), and the driveline behind the gearbox are explained and modeled. However in order to avoid causality problems, a slight slip is allowed in the gearbox model even when the clutch is closed. Moreover, the authors explicitly state that "a simplified model is made and used for the study of only one shift… you must change the elements' values when you change the shift studied"; hence avoiding the complexities of integrating sub-models into a complete Bond graphic representation. Two independent clutches are used in order to control two pressures. In the end the authors conclude that although the peak on the output torque during a shift seems to be inescapable, independent control of the two pressures can reduce the peak.

Edström et al. in [9] discuss two different ways to model mode-switching systems;

i.e. dynamic systems undergoing fast transitions between continuous modes of operation. One method is to handle the fast transitions by means of stiff solvers with the advantage of avoiding further complexity of the modeling process. Another approach is to approximate fast transitions by ideal instantaneous mode-transitions. This is modeled by a set of continuous dynamic mode-models and a set of discrete mode-transitions. Each mode transition is further associated with a transition condition; i.e. a Boolean combination of relations over variables in continuous mode-models. The emerging problem is that the number of mode-models grows very rapidly with the number of switches in the system. The authors then describe the Bond graphic switch element and the importance of causality. To simulate a system, its model is translated into an acceptable form for a numerical solver in two steps: first the Bond graph is translated into a solver-independent mathematical form; subsequently this intermediate model is interpreted into solver-specific code. The intermediate structure is a hybrid system since it combines continuous and discrete mode-transitions. Thereafter the authors mention that in order to avoid more detailed discussion, it is assumed that no causality conflict exists in any mode. Later on, the intermediate structure is explained in more detail and their composition is described by means of the “interleaved composition operator”, in which only one switch is allowed to switch at a time.

In [20] Lumkes and Fronczak focus on a hydraulic axial motor, switching intermittently as a pump for dynamic braking in a hydraulic transmission system, using four high speed on-off poppet valves. The authors claim that this system provides regenerative braking, engine and road load decoupling, independent torque control at

each drive wheel and higher urban fuel economy. In the model development, only two valves out of four are modeled due to symmetry, and the vehicle direction is assumed constant during the switch from acceleration to braking and vice versa. However it is mentioned that incorrect command from the adaptive controller resulting in both valves to be closed or opened causes either undesirably high or low fluid pressure. The Bond graph state equations are integrated using a variable step fourth order Runge-Kutta algorithm. At the end the authors mention that proper valve commands can only be determined after the valve characteristics are known. In the simplest form, the system is analogous to a spring (hose and oil compliance) that must be compressed to the proper force (pressure) and at that point the operating mode of the system must be quickly switched; pump to motor or vice versa.

Garcia et al. in [13] apply Bond graph modeling to semiconductor switching devices, with the aim of coming up with a single Bond graph representation for all switch state combinations that can physically occur. The authors mention the historic difficulty of modeling power electronic switching converters, and explain the two approaches for modeling discontinuities with Bond graphs in power electronics; the difference between the methods being the Bond graphic causality assignment. The first approach suggests free causalities for the circuit elements due to different states of the switches, while the second approach conserves causality assignments regardless of the switch state. The authors claim that although applying the first method benefits from utilizing ideal switches, however 2^n Bond graph models are required for a device containing n switches. Hence the authors choose the second approach in which non-ideal switches are used with

fixed causality assignments despite the switch being open or closed. Later on, the physical details of a semiconductor switching device is explained, and "pseudo Bond graphs" (in which the product of the flow and effort is not "power") are used for modeling since "temperature" has been chosen as the effort and "heat flow" as the flow; whereby "heat flow" is already a "power" variable. The authors conclude that the single Bond graph model allows the simulation of temperature distribution of the device from the chip surface through the package and heat sink as well as the currents and voltages within the network, and the power dissipated in the semiconductor is used by the Bond graph thermal model to determine the dynamic temperature in the circuit.

In [22] Otter et al. discuss hybrid modeling and its implementation in the object-oriented modeling language Modelica, based on a synchronous data flow principle which states that at every time instant the active equations of a hybrid model express relations between variables that have to be fulfilled concurrently. For this purpose, all equations are sorted by block-lower-triangular partitioning under the assumption that all equations are active. The order of equations is determined by data flow analysis resulting in an automatic synchronization of continuous and discrete equations. The authors mention the disadvantage of this method to be a restriction on the types of systems that can be modeled. In a switching system, the requirement for the equations to be continuous and differentiable is often violated by "if" clauses. The proposed solution is by detecting the switching points with a prescribed bound, halting the integration, selecting the corresponding new branch, and restarting the integration. Since it is not possible to determine beforehand whether a specific relation will lead into a discontinuity or not, it is

assumed that every relation will potentially experience a discontinuity or non-differentiable point. Later on in the paper, a "parameterized curve description" is explained by means of an ideal diode and Coulomb friction as examples, presenting the equations for the latter and making use of the derivative of the velocity in order to construct a switching structure. The authors conclude with simulating a two block system coupled by Coulomb friction.

Elmqvist et al. in [12] focus on noteworthy considerations in object-oriented modeling of switching power electronic circuits using Dymola. The modulation/regulation of power signals typically incorporates high frequency switching devices. It is mentioned that modern integration routines handle discontinuities utilizing "indicator functions"; whenever an indicator function crosses through zero it indicates to the integration algorithm that a discrete event is about to occur. Instead of decreasing the step size, the integration routine interpolates the indicator function in order to locate the zero crossing. After the discrete event takes place, the integration algorithm is restarted. In Dymola, discontinuities are described using "if" expressions and "when" clauses, giving it the ability to cope with variable structures. As an example, a power converter entailing four ideal switches is presented and different switching combinations are discussed, along with four different problematic situations due to different combinations of the switches. The solution proposed by the authors is the replacement of ideal elements with non-ideal ones, depending on the cause of error. Towards the end, an AC-DC converter with pulse-width modulation control is modeled and simulation results are presented.

2. *BACKGROUND*

This chapter offers a short introduction to the technical background employed in this text; arranged in two sections: in the first section, the essence of Bond graph modeling is reviewed. The second section briefly describes some necessary aspects of digital logic, which serve as the backbone of the algorithm developed in this thesis.

2.1. *BOND GRAPH MODELING*

2.1-1. *Why Bond Graphs?*

Three graphical representation schemes are prominent for modeling engineering systems: Block diagrams, Signal Flow graphs, and Circuit diagrams. Block diagrams consist of blocks which represent transducers, interconnected with paths which represent signals. In Signal Flow graphs, each path in a Block Diagram is replaced by a node and each box is converted to a path, hence it could be considered to be the dual representation of Block diagrams. These two metaphors have the same advantages and disadvantages: they capture the computational structure of the system, while not preserving the topological structure. If a slight change in the physical system occurs, it might require that the entire computational structure be rearranged, and therefore its corresponding diagram may change drastically. An element in a power-flowing circuit has simultaneously two variables associated with it: one *across* variable; the potential v, and one *through* variable; the current i. In Block diagrams and Signal Flow graphs, these two variables get separated from each other, and it is this fact that tears down the analogy

between the topological structure and the computational structure [3].

In contrast to block diagrams and signal flow graphs which preserve the computational structure of a system solely, circuit diagrams are only able to reflect the topological structure, in addition to the fact that their application is restricted to electrical systems.

2.1-2. *Power Bonds*

With the major drawback mentioned above, the necessity of coming up with yet another method seemed unavoidable; a method which would not only overcome the stated disadvantage but could also be applied to all kinds of physical systems. All physical systems have in common the conservation laws for mass and energy. Bond graphs, originated by Paynter [24], concern themselves with the conservation of energy. Presented here is a short introduction to Bond graphs. Further details can be found in [3], [4] and [24].

Energy can be transported from one place to another, can be stored and can be converted into different forms, but cannot be dissipated. Hence if the amount of energy is changing at a certain location, either additional energy is flowing in, or some of the initial energy is flowing out. In both cases, there exists a rate of change of energy in time; defined as power, P:

$$P = dE / dt \tag{2.1}$$

Power flow can be expressed as the product of two adjugate variables: an *extensive* variable (proportional to the amount), and an *intensive* variable (independent of the amount). In an electrical system, for instance:

$$P = v \cdot i \tag{2.2}$$

where the power is a product of *voltage* and *current*. In a mechanical system:

$$P = F \cdot V \tag{2.3}$$

with the power being a product of *force* and *velocity*. In Bond graphs, the energy flows are represented as directed harpoons between two pins. The two variables are annotated *above* (intensive: *effort* variable, "e") and *below* (extensive: *flow* variable, "f") the harpoon. Voltage, temperature, force, torque, and pressure are examples of effort variables; while current, entropy flow, velocity, angular velocity, and fluid flow are examples of flow variables. In our convention, *above* is the side where the hook is, as depicted in Fig. 2.1. These bonds are connected to junctions, and system elements.

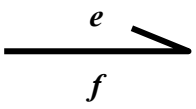
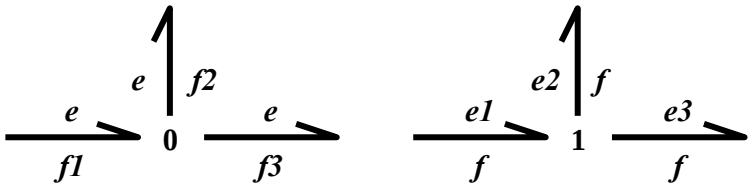


Fig. 2.1: A bond, with effort and flow.

2.1-2.1. *Junctions*

Two types of junctions exist: the "0-junction" and the "1-junction"; shown in Fig. 2.2.



$$\sum f_i = 0 \qquad e_1 = e_2 = \ldots = e_n \tag{2.4}$$

$$\sum e_i = 0 \qquad f_1 = f_2 = \ldots = f_n \tag{2.5}$$

Fig. 2.2: Type 0 and type 1 junctions.

In a type 0 junction, all effort variables are equal, while all flow variables add up to zero, thus a 0-junction is equivalent to a node in an electrical circuit. In a type 1 junction, all flow variables are equal and all effort variables add up to zero. The 0-junction and 1-junction represent Kirchhoff's current and voltage laws in linear circuit theory, respectively. At least three bonds are required to form a junction since with only two bonds either of the junctions results in two identical equations. Moreover, neighboring junctions of the same type can be combined into a single junction.

2.1-2.2. *System Elements*

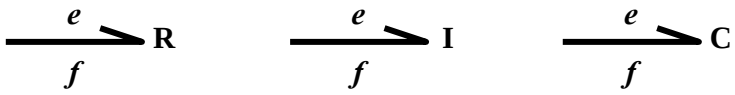
We shall concern ourselves with four types of system elements: **sources**, **passive elements**, **reversible energy converters**, and **switches**.

Sources fall into two categories: *effort* sources and *flow* sources, demonstrated in figure 2.3. If the effort and flow variables of a source have opposite directions, energy is added to the system.

Fig. 2.3: Effort and flow sources.

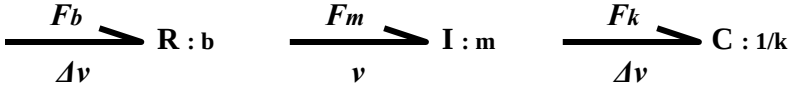
Three types of **passive elements** are discussed in this text: *resistors* (which dissipate energy, i.e. transform it into heat irreversibly), *capacitors* and *inductors* (which both store energy). Inductors and capacitors, depending on the relative direction of their flow and effort variables can either discharge energy into the system or draw it out towards themselves, whereas resistors always drain energy out of the system. In each element, the relation between the effort, flow, and the element's characteristic parameter is governed

by a single equation. Figure 2.4 demonstrates the elements for electrical systems, and figure 2.5 illustrates the same elements for mechanical systems; with the resistance being equivalent to the damper, the inductance denoting the mass, and the capacitance functioning as the reciprocal of the spring constant. In equations 2.9 and 2.11, Δv is the velocity *difference* of the two ends of the damper and the spring, respectively.



$$\begin{aligned} e &= R \cdot f \\ f &= R / e \end{aligned} \quad (2.6) \qquad df/dt = e/I \quad (2.7) \qquad de/dt = f/C \quad (2.8)$$

Fig. 2.4: Passive elements in electrical circuitry.



$$F_b = b \cdot \Delta v \quad (2.9) \qquad dv/dt = F_m / m \quad (2.10) \qquad dF_k / dt = \Delta v \cdot k \quad (2.11)$$

Fig. 2.5: Passive elements in mechanical circuitry.

Whereas the conventional representation of mechanical systems makes use of the absolute motions of the masses (position and velocity) as its state variables, the Bond graph representation selects the absolute velocities of masses as one type of state variable, and the spring forces as the other. As an example, consider a simple two-body mechanical system connected by a spring and a damper in parallel, positioned on a surface with friction, under the effect of a single external force. Let us assume that the two bodies have different frictional characteristics. An illustration of this system is

presented in Fig. 2.6.

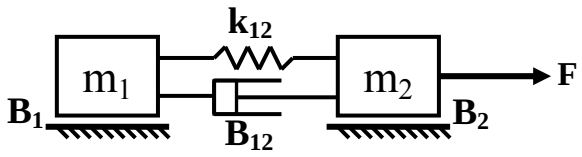


Fig. 2.6: A two-body mechanical system.

The D'Alembert principle is formulated for each mass by grouping all forces that act on that particular mass around a junction of type 1. Each of the connecting elements between m_1 and m_2 (being a spring and a damper here) through which the two bodies interact is represented by a type 0 junction placed between the two masses, resulting in the Bond graph metaphor shown in Fig. 2.7. The effort source in this figure denotes the external force acting on m_2. The perpendicular strokes at either side of the bonds depicted in Fig. 2.7 are explained in the next section on causal Bond graphs.

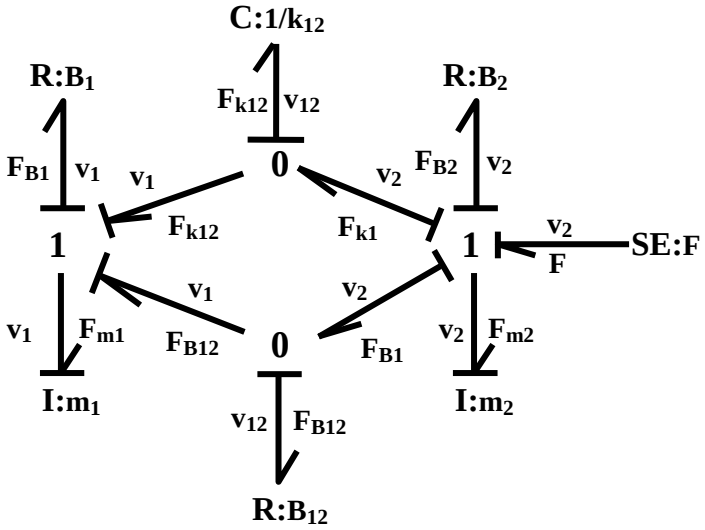
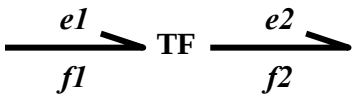


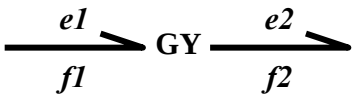
Fig. 2.7: A mechanical system in Bond graph representation.

Reversible energy converters mentioned here are *transformers* and *gyrators*, both

possessing an incoming and an outgoing bond; demonstrated in Fig. 2.8 with their governing equations. Mechanical gears, electric transformers and hydraulic shock absorbers are examples of transformers. A DC motor is an example of a gyrator.



$$e1 = me2 \qquad f2 = mf1 \tag{2.12}$$



$$e1 = rf2 \qquad e2 = rf1 \tag{2.13}$$

Fig. 2.8: Transformer and gyrator.

A **switch** is a two-pin element, just like all the other traditional passive linear circuit elements, but with two situations: *closed*, in which the flow through the switch is determined from the outside and the effort across the switch is zero (meaning that the effort doesn't change from one side to the other in an ideal closed switch); and *open*, in which the flow through the switch is zero, and the two sides of the switch have independent effort values. Whereas resistors, inductors, capacitors, and sources are all modeled with a single equation, the switch element is described by two equations, one for each of the two possible switch positions. However, these two equations can be combined in a single conditional statement:

$$\mathbf{0} = \text{if } \boldsymbol{Openswitch} \text{ then } \boldsymbol{flow} \text{ else } \boldsymbol{effort} \tag{2.14}$$

Openswitch is a *Boolean variable* with the two possible values *true* and *false* corresponding to the *open* and *close* positions, respectively. For the purpose of an equation solver, this equation can be re-written in a more useful form as:

$$\mathbf{0} = \boldsymbol{Openswitch\ .\ flow} + (1 - \boldsymbol{Openswitch})\ \boldsymbol{.\ effort} \qquad (2.15)$$

In this equation, ***Openswitch*** is a discrete variable with the two possible values 1 and 0, representing the open and close positions respectively, which is a signal transmitted to the switch depending on the operational requirements of the circuit.

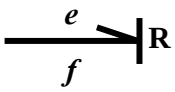
2.1-3. *Causal Bond Graphs*

Since each bond incorporates two variables, the effort and the flow, consequently two equations are required to solve for these two variables. One equation is provided by the governing equation of the element. It turns out that it is always possible to compute one of the two variables at each side of the bond. A vertical bar placed at one side of the bond indicates the side where the flow is being computed. The causality of the discussed elements is depicted in the following figures.

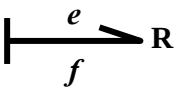
Sources have fixed causality bars. The effort source computes the effort; therefore the flow is computed on the *other side*. The flow source computes the flow and consequently has the causality stroke at the *source end* of the bond, as shown in Fig. 2.9.

Fig. 2.9: Causality for sources.

Since there is no casual preference for the **Resistors,** therefore they have free causality. The causality stroke is placed depending on where the flow is being computed, as shown in Fig. 2.10.



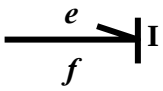
$$f = (1/R)\cdot e \quad (2.16)$$



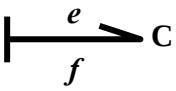
$$e = R\cdot f \quad (2.17)$$

Fig. 2.10: Free causality in resistors.

The causality of the **storage elements** is determined by the desire to use integrators instead of differentiators; however, this is a preference, not a requirement. Figure 2.11 shows the preferred causal bars for these elements.



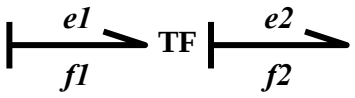
$$df/dt = e/I \quad (2.18)$$



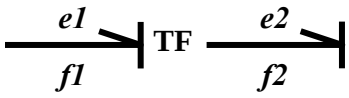
$$de/dt = f/C \quad (2.19)$$

Fig. 2.11: Preferred causalities in storage elements.

Since there exists exactly one equation for the efforts and another for the flows, it is mandatory that the **transformer** compute one effort variable and one flow variable. Hence there is one causality stroke at the TF element. There are two possible causality allocations for the transformer element, depending on its location in a system, both shown in Fig. 2.12.



$$\begin{aligned} e1 &= m\cdot e2 \\ f2 &= m\cdot f1 \end{aligned} \quad (2.20)$$

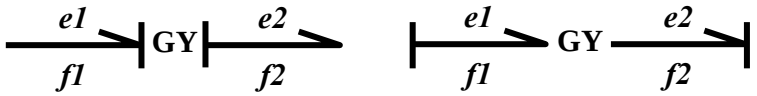


$$\begin{aligned} e2 &= (1/m)\cdot e1 \\ f1 &= (1/m)\cdot f2 \end{aligned} \quad (2.21)$$

Fig. 2.12: Transformer in its two possible causal configurations.

As we must compute one equation at the left and the other at the right of the **gyrator**,

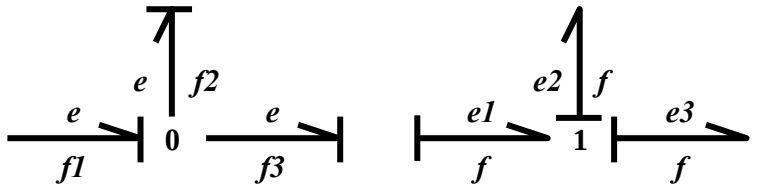
the equations may either be solved for the two effort variables or for the two flow variables. Similar to the transformer, the causality for the gyrator can attain two different situations; as depicted in Fig. 2.13.



$$f2 = (1/r)\cdot e1 \qquad e1 = r\cdot f2 \quad (2.22)$$
$$f1 = (1/r)\cdot e2 \qquad e2 = r\cdot f1 \quad (2.23)$$

Fig. 2.13: Gyrator in its two possible causal configurations.

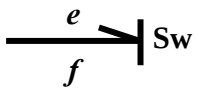
Junctions of type 0 have only one flow equation therefore exactly one causality bar must be at the junction. Junctions of type 1 have only one effort equation, resulting in exactly (n-1) causality bars at the junction. This property was also demonstrated in figure 2.7, where the two connected masses where taken as an example for Bond graph representation. Figure 2.14 shows this property for the two junctions in a simple 3-bond case.



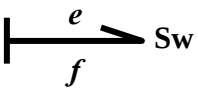
$$f1 = f2 + f3 \quad (2.24) \qquad e1 = e2 + e3 \quad (2.25)$$

Fig. 2.14: Causality in junctions.

The causality of the **switch** depends on the switch's state; an open switch necessitates a "flow = 0" for its relative junction and a closed switch inflicts an "effort = 0". Figure 2.15 shows both cases.



$$f = 0; open \quad (2.26)$$



$$e = 0; closed \quad (2.27)$$

Fig. 2.15: Open and closed switches.

As a direct (and important) result of two possible states for the switch, its causality must not be dictated by its surrounding circuit, but must be merely a function of the independent discrete variable ***Openswitch*** from equation 2.15. This result and its ramifications will be exploited in the next chapter.

At the end it should be noted that the Bond graph representation is not unique: several different-looking bond graphs may represent identical equation systems; see [3], [4].

2.2. *DIGITAL LOGIC: THE KARNOUGH MAP*

The second section of this chapter is on digital logic. Karnough maps are a method for simplifying and graphically representing Boolean functions. A Boolean function $f(x_1, x_2, x_3, ..., x_n)$ is a function of *n* individual statements $x_1, x_2, x_3, ..., x_n$ combined by AND, OR, and NOT operations. The statements $x_1, x_2, x_3, ..., x_n$ are known as Boolean variables which can either be true or false, corresponding to a 1 or 0 in Boolean algebra. For instance,

$$f(X, Y, Z) = X\overline{Y} + XY(Z + \overline{X}) + \overline{Z} \quad (2.28)$$

is a Boolean function composed of three variables *X*, *Y* and *Z*; with the product as AND, sum as OR, and the over bar denoting the complement, NOT.

Boolean functions can be simplified into sums of products or products of sums. For example, expressing equation 2.28 as the sum of products results in:

$$f(X,Y,Z) = X\overline{Y} + XYZ + XY\overline{X} + \overline{Z} \tag{2.29}$$

with the third term simply being 0, hence canceling out.

For any Boolean function there exists a standard or canonical form in which each term contains all of the variables, either complemented or uncomplemented (Note that a variable *a* and its complement $\overline{a}$ are considered to be the same "variable", but different "literals"). A product possessing this property in known as a "minterm", whereas a sum with this property is known as a "maxterm". For converting equation 2.29 into this canonical form, we start out by multiplying each term with the "sum of the missing variable and its complement";

$$f(X,Y,Z) = X\overline{Y}(Z + \overline{Z}) + XYZ + \overline{Z}(X + \overline{X})(Y + \overline{Y}) \tag{2.30}$$

which eventually simplifies as:

$$f(X,Y,Z) = XYZ + X\overline{Y}Z + XY\overline{Z} + X\overline{Y}\,\overline{Z} + \overline{X}Y\overline{Z} + \overline{X}\,\overline{Y}\,\overline{Z} \tag{2.31}$$

Equation 2.31 expresses $f(X,Y,Z)$ by its minterms. Similarly, an expression composed of maxterms can be obtained for $f(X,Y,Z)$, which we shall not attend. A Boolean function of *n* variables has 2*n* minterms and 2*n* maxterms. For the sake of simplicity, minterms are usually coded in decimal numbers. In each term we assign a 1 to a variable and 0 to a complemented variable. As an example, $X\overline{Y}Z$ would be 101; the binary equivalent of 5 which is usually denoted as m_5. Hence we could write equation 2.31 as:

$$f(X,Y,Z) = m_7 + m_5 + m_6 + m_4 + m_2 + m_0 \tag{2.32}$$

or in the "minterm list form" as:

$$f(X,Y,Z) = \sum m(0,2,4,5,6,7) \tag{2.33}$$

For functions up to six variables, a Karnough map can be constructed out of 2^n possible minterms of the n variables, arranged separately in a rectangle split into 2^n cells. Each cell is associated with a number that corresponds to the decimal value of each minterm. Consider the following examples:

$$f_1(X,Y) = \overline{X}Y \tag{2.34}$$

with the corresponding Karnough map in Fig 2.16.

	$\overline{Y}$	Y
$\overline{X}$	0 **0**	1 **1**
X	2 **0**	3 **0**

Fig. 2.16: Karnough map for equation 2.34.

And for:

$$f_2(X,Y,Z) = X\overline{Y}\overline{Z} + \overline{X}YZ + X\overline{Y}Z + \overline{X}Y\overline{Z} \tag{2.35}$$

the Karnough map is shown in Fig. 2.17.

	$\overline{Y}\overline{Z}$	$\overline{Y}Z$	YZ	$Y\overline{Z}$
$\overline{X}$	0 **0**	1 **0**	3 **1**	2 **1**
X	4 **1**	5 **1**	7 **0**	6 **0**

Fig. 2.17: Karnough map for equation 2.35.

The main feature of a Karnough map is that cells that are physically adjacent are also logically adjacent. In other words, physically adjacent minterms differ from each other by a single variable. Consider the following example:

$$f(A,B,C,D) = \Sigma m(0,2,3,5,9,11) \tag{2.36}$$

A Karnough map can be for constructed equation 2.36 as the table depicted in Fig. 2.18, leaving blank the contents of the cells with zeros.

	$\overline{C}\overline{D}$	$\overline{C}D$	CD	$C\overline{D}$
$\overline{A}\overline{B}$	0 **1**	1	3 **1**	2 **1**
$\overline{A}B$	4	5 **1**	7	6
AB	12	13	15	14
$A\overline{B}$	8	9 **1**	11 **1**	10

Fig. 2.18: Karnough map for equation 2.36.

Adjacent cells containing a 1 have been grouped in Fig. 2.18, indicating the minterms that can be combined to produce simpler terms. Notice that cells 3 and 11 are adjacent, so

are cells 2 and 0. Minterm 5 cannot be grouped with any other cell since there isn't present a cell containing a 1 in its neighborhood. In order to simplify equation 2.36, the variable that changes from 1 to 0 or vice versa in the Karnough map is eliminated from the minterm expression of grouped cells. For example in the grouping of minterms 9 and 11, we can observe by inspection that *C* changes value, therefore it is left out while combining the two cells. As a result of carrying out this procedure for equation 2.36, five of the six minterms are combined into three pairs, as demonstrated in equations 2.37-39:

$$m_0 + m_2 = \overline{A}\,\overline{B}\,\overline{C}\,\overline{D} + \overline{A}\,\overline{B}C\,\overline{D} = \overline{A}\,\overline{B}(C + \overline{C})D = \overline{A}\,\overline{B}D \tag{2.37}$$

$$m_3 + m_{11} = \overline{A}\,\overline{B}CD + A\overline{B}CD = (\overline{A} + A)\overline{B}CD = \overline{B}CD \tag{2.38}$$

$$m_9 + m_{11} = A\overline{B}\,\overline{C}D + A\overline{B}CD = A\overline{B}(\overline{C} + C)D = A\overline{B}D \tag{2.39}$$

And equation 2.36 simplifies to:

$$f(A,B,C,D) = \overline{A}\,\overline{B}\,\overline{D} + \overline{B}CD + A\overline{B}D + \overline{A}\overline{B}CD \tag{2.40}$$

which can be employed in constructing the switching diagram in Fig. 2.19.

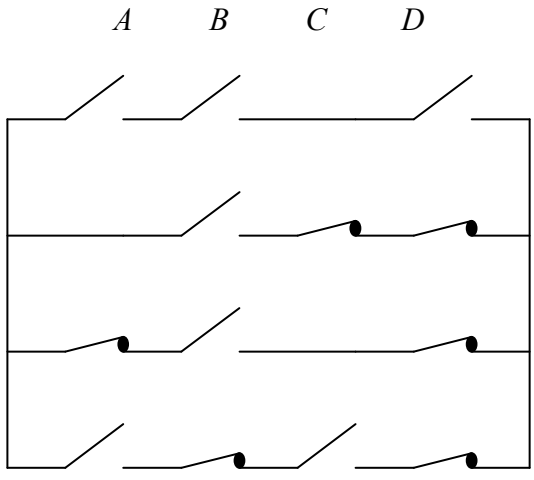


Fig. 2.19: Switching diagram for equation 2.40.

Each Boolean variable is represented by a set of switches placed in the same column which operate simultaneously. For instance, there are two switches for C (in the second and fourth row) which have inverse functionality, since equation 2.40 contains C and $\overline{C}$. By this method we were able to reduce the number of switches from 24 to 13.

At the end it should be mentioned that we only concerned ourselves with minterms and grouping the 1 elements, and our example only portrayed grouping two adjacent cells. Larger number of adjacent cells can be grouped, provided the number of cells in a group is a power of 2. Also a cell may participate in any number of pairings desired. Furthermore it is possible to exploit grouping the maxterms by a similar (dual) method.

Karnough maps are not suitable for functions with more than six variables. For these cases, there exists another approach named the Quine-McCluskey method. Details regarding this technique can be found in any standard digital logic text, such as [16] and [19].

3. *SYSTEMATIC ALGORITHM FOR SWITCHING NETWORKS*

This chapter is the heart of this thesis. The main method for developing a structured algorithm for deriving switching networks in Bond graphs is provided here in full detail. This chapter consists of two sections. The first section is on modeling a hypothetical automotive gearbox, including a brief description followed by Bond graph modeling and analysis. The second section looks into a pilot ejection scenario and its representation in Bond graphs utilizing the main technique, accompanied by further discussion on analogous mechanical and electrical elements.

3.1. *GEARBOX MECHANISM*

3.1-1. *Basic Description*

The forces required for locomotion in an automobile are provided by the power train. These forces are generated by the conversion of the chemical energy of liquid fuels (or the electrical energy of an electric motor) into the energy of motion (kinetic energy) within the internal combustion engine. The torque and angular velocity generated by the engine are converted to different ratios in the transmission system, governed by the following equation:

$$P = T \cdot \omega \tag{3.1}$$

were P is the power, T is the torque and ω is the angular velocity. Hence under constant power generation, T and ω are related reciprocally: the higher the torque (more “force” in the wheels), the lower the angular velocity (less car speed), and vice versa. This selection

of torque and angular velocity is made possible by means of the gearbox, according to the tractive-power demand of the vehicle. Compound reverted gear trains are commonly used in manual automotive transmissions to provide user-selectable ratios between the engine and the drive wheels for the torque multiplication. These gearboxes usually have from three to six forward speeds and one reverse. Overall conversion takes place in a manually shifted or automatic transmission with variable transmission ratios, in addition to a final drive with a constant transmission ratio. The drive train layout varies in automobiles, depending on the position of the engine and the driven axle. Conventional manual transmissions are usually composed of a gearbox which entails three shafts with gears mounted on them: the input shaft, the idler shaft, and the output shaft. The input shaft transmits the power from the engine to the gearbox. On the engine side, the input shaft ends up in the clutch disk, and on the gearbox side, it ends up in a fixed gear mounted on it, which transmits the power to the idler shaft. This input gear is always in mesh with the left most gear on the countershaft (idler) at the bottom, as demonstrated in Fig. 3.1.

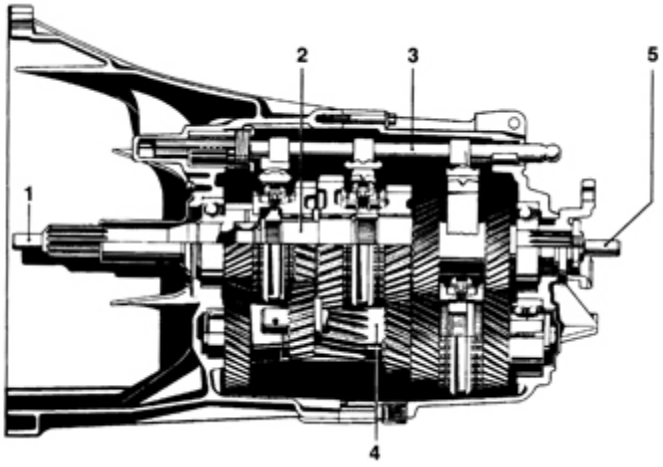


Fig. 3.1: Five Speed Transmission: *1 Input Shaft, 2 Main Shaft, 3 Selector Rail, 4 Idler, 5 Output Shaft.*

The countershaft has several fixed gears integral with it, each of which meshes with a different output gear that is freewheeling on the output shaft. The output shaft is concentric with the input shaft, making this a reverted train, but the input and output shafts only connect through the gears on the countershaft except in "top gear" (usually fourth or fifth speed), for which the input and output shafts are directly coupled together with a synchromesh clutch for 1:1 ratio. Therefore the power is transmitted via the input shaft to the idler, and from the idler to the output shaft, and from there onwards to the following elements in the driveline.

The mechanical gearbox could be viewed as a mechanical switching network with the same functionality of "mutual exclusivity", i.e. one and only one switch is closed (operating) at each time (If the automobile is cruising with the 1st gear, then it is NOT cruising with the 2nd, 3rd and 4th gear, and so on). Consequently if we model a theoretical gearbox with four stages, naming switch *A* for the 1st gear, *B* for 2nd, and so on, the logical representation would be:

$$\bar{A}\bar{B}\bar{C}\bar{D} + \bar{A}B\bar{C}\bar{D} + \bar{A}\bar{B}C\bar{D} + \bar{A}\bar{B}\bar{C}D \tag{3.2}$$

which could be represented with the decision tree shown in Fig. 3.2.

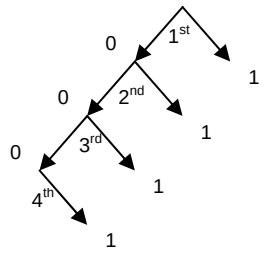


Fig. 3.2: Four switch decision tree.

We now attempt to translate this expression into an electrical switching network using the Karnough map in Fig. 3.3. As it can be seen, there are no adjacent cells with a 1 element, hence equation 3.2 is the minimal expression.

	$\overline{C}\,\overline{D}$	$\overline{C}D$	CD	$C\overline{D}$
$\overline{A}\,\overline{B}$	0	1 **1**	3	2 **1**
$\overline{A}B$	4 **1**	5	7	6
AB	12	13	15	14
$A\overline{B}$	8 **1**	9	11	10

Fig. 3.3: Karnough map for gearbox.

The corresponding switching circuit is depicted in Fig. 3.4.

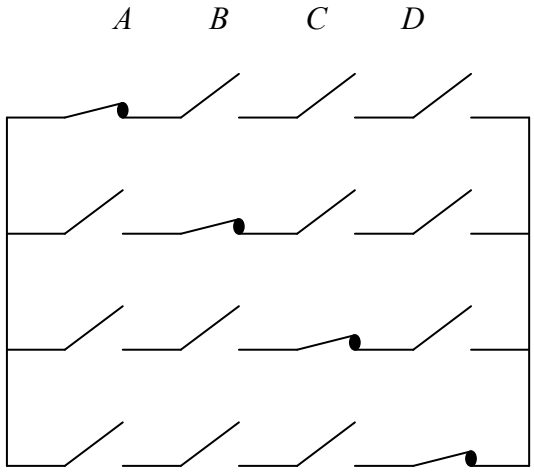


Fig. 3.4: Switching network for gearbox.

3.1-2. *Bond Graph Modeling*

The network in Fig. 3.4 can now be presented via Bond graphs shown in Fig. 3.5. In this diagram, the switching network of Fig. 3.4 has been rotated 90º CCW. Each row containing the four switches from Fig. 3.4 has been replaced with a type 1 junction, surrounded by six bonds, four of which are connected to switch elements.

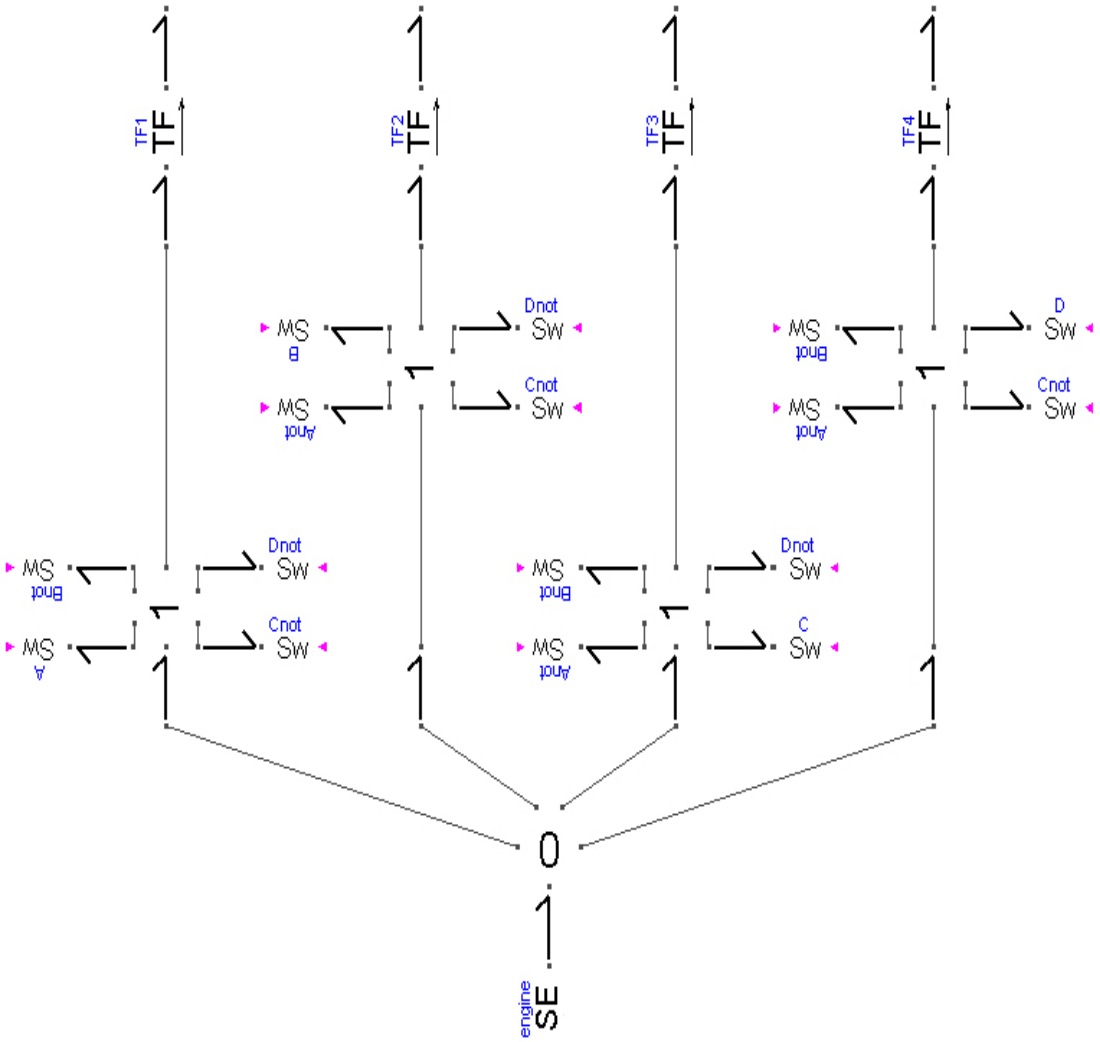


Fig. 3.5: Gearbox in Bond graphs; first attempt.

The effort source represents the torque generated by the engine, streaming power into a type 0 junction which is the entering node for the switching network. The four departing lines from this node correspond to four power paths, each one passing through its relevant series of switches and eventually arriving at a transformer which is responsible for the torque/angular velocity conversion similar to a gear pair. Notice that the model in Fig. 3.5 is only a partial model, mainly to demonstrate the Bond graphic arrangement of the switching network. For completion, a triggering device for the switch elements needs to be implemented; also the bonds departing the transformers have to be connected to the other components of the driveline depending on the modeler's desire.

According to our design in Fig. 3.5, we expect that by carrying out a switching plan resulting in any of the "true" combinations ($A\overline{B}\overline{C}\overline{D}$, $\overline{A}B\overline{C}\overline{D}$, $\overline{A}\overline{B}C\overline{D}$ or $\overline{A}\overline{B}\overline{C}D$), the engine power would flow through the 1st, 2nd, 3rd or 4th path respectively. However, a closer examination shows that the "causality bar rule" for the type 1 junction is violated if more than one switch in a line is opened; i.e. considering the bonds around a type 1 junction, more that one causality bar would rest away from the junction itself (previous chapter). Whereas in forming the switching diagram in Fig. 3.4, apparently there were no restrictions whatsoever in opening or closing any particular switch, as long as the consistency is preserved (meaning that if anywhere we open a switch, we close its complement and vice versa). Is the Bond graph model in Fig. 3.5 wrong?

This problem emerges from the fact that in the electrical network, if two consecutive switches are open, the wire in between would have "floating voltage", i.e. the voltage in that section becomes indeterminate. Consider the circuit in Fig. 3.6, with two ideal

switches in series, a voltage source and a resistor. As depicted in this figure, if both switches *A* and *B* are open, the voltage in the intermediate wire cannot be determined.

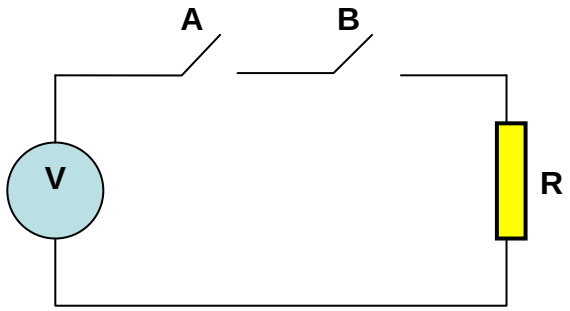


Fig. 3.6: Two switches in a series.

A similar phenomenon occurs in the dual problem: consider the electrical circuit in Fig. 3.7, with ideal switches in parallel.

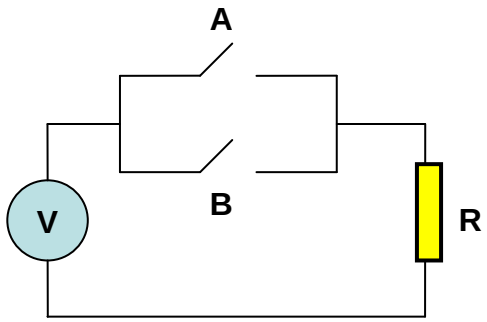


Fig. 3.7: Two parallel switches.

If either of the switches *A* or *B* is closed while the other is kept open, naturally the current will flow through the closed switch and thereafter through the resistor R without any problems. But if both switches are closed simultaneously, then we cannot determine the current in either of the branches, as we assumed ideal wires and hence no resistance in them. This is the very problem that propagates to the Bond graph scheme of Fig. 3.5, resulting in the causality conflict for the type 1 junctions, which further infects the type 0

junction too.

One way to cope with this problem is to use “non-ideal” switches. For instance, let us take a closer look at a real world example: In the automotive gearbox we know that all gears are in contact with their conjugate on the countershaft, but only one gear is actually transmitting power, while the other gears are rotating freely. Gear shifting is performed using either disengagement of power transmission (positively interlocking mechanism) in manual and semi-automatic gear boxes, or under load by a friction mechanism in automatic transmissions. The force required for gear selection is transmitted via shaft linkage rods or cables. Gears on the output shaft are not moved into and out of engagement when shifting from one speed to another except for reverse. Rather, the desired ratio gears are selectively locked to the output shaft by synchromesh mechanisms. Before a shift can take place, it is necessary to synchronize the rotating velocities of the transmission elements being joined. Synchronizer assemblies include a friction coupling for initial equalization of rotating speed and a lockout mechanism to prevent positive gear engagement prior to the completion of the synchronization process. Figure 3.8 shows this mechanism for manual transmissions.

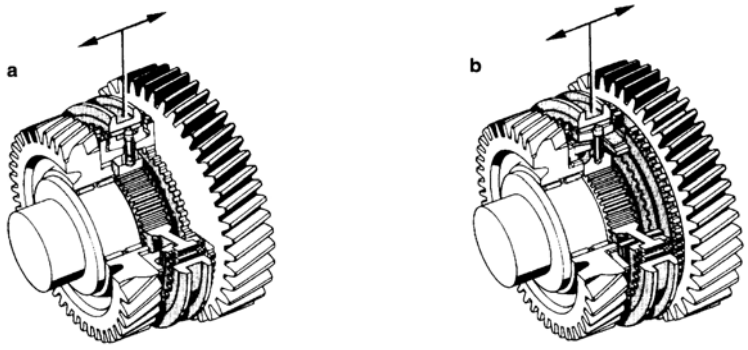


Fig. 3.8: Manually shifted transmission; a) Single-band synchromesh, b) Dual-band synchromesh. [1]

The synchromesh clutches are beside each gear on the output shaft and are partially hidden by the shifting collars which move them left and right in response to the driver's hand on the shift lever. These clutches act to lock one gear to the output shaft at a time to provide a power path from input to output of a particular ratio. When the gear is changed, the operating lever shifts the synchromesh gear, which initially synchronizes the angular velocity of this newly chosen gear, and then locks it. Thus there exists a transition state, in which the angular velocity of the newly chosen gear is synchronized with the angular velocity of the shaft which it is mounted on, indicating that the "switching" is not ideal. It is impossible to "jump" from one state to another arbitrarily chosen state. For example, if a car is at top speed in the 4^{th} gear, it is not possible to suddenly shift to the 1^{st} gear.

In order to implement non-ideal switches, one method is to insert a high resistance element which would allow a very little amount of current bypass, as shown in Fig. 3.9.

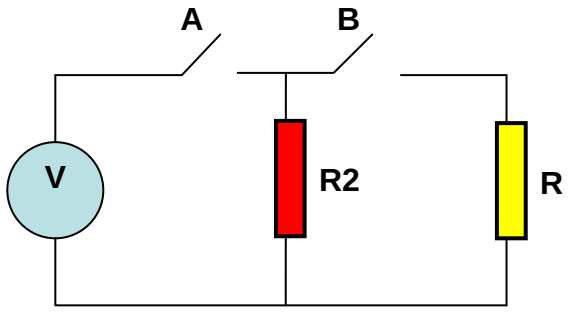


Fig. 3.9: Two switches with high resistance in between.

When both switches are closed, the circuit would operate in almost the same manner as the circuit in Fig. 3.6 would, only that now a little amount of flow would pass through the new route since R2 is of a very high resistance. The improvement of our model becomes apparent when both of the switches are opened, in which now there will not be

any undetermined voltages/currents since all of the components in the circuit are interconnected.

Another method to implement non-ideal switching is by introducing a supplementary high-resistance element parallel to each switch, as depicted in Fig. 3.10. Even if both switches are open at the same time, electrical current exists in the bypass lines, and no undetermined voltages/currents will show up. In this sense, each switch accompanied with its parallel resistor is defined as a non-ideal switch element.

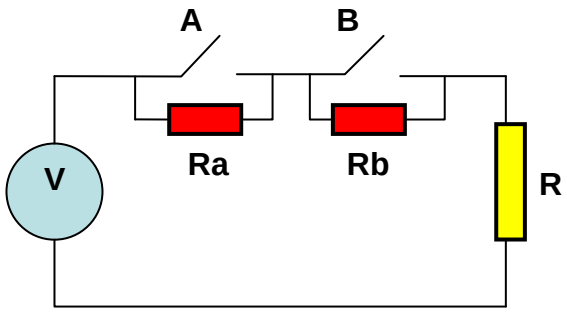


Fig. 3.10: Two switches with high resistances parallel to each switch; Non-ideal switches.

The techniques mentioned above, however, get rid of the problem by adding non-idealness to the system, which is something we do not intend to do. Our purpose is to establish an algorithm for modeling "ideal" switching networks, and from then on introducing non-ideal elements into the system is left to the modeler depending on what compromises s/he decides to make or what factors s/he chooses to give more importance to. A possible way to tackle this issue in the electrical network is to connect the intermediate wire between two switches to the ground, with a wire that contains yet another switch. Whenever the switch to the left on the main wire is closed, the switch on the ground wire is opened; but whenever the switch on the main wire is opened, this

ground wire switch is closed in order to attach the intermediate wire to the ground and hence allow it to dump its voltage. These modifications are shown in Fig. 3.11.

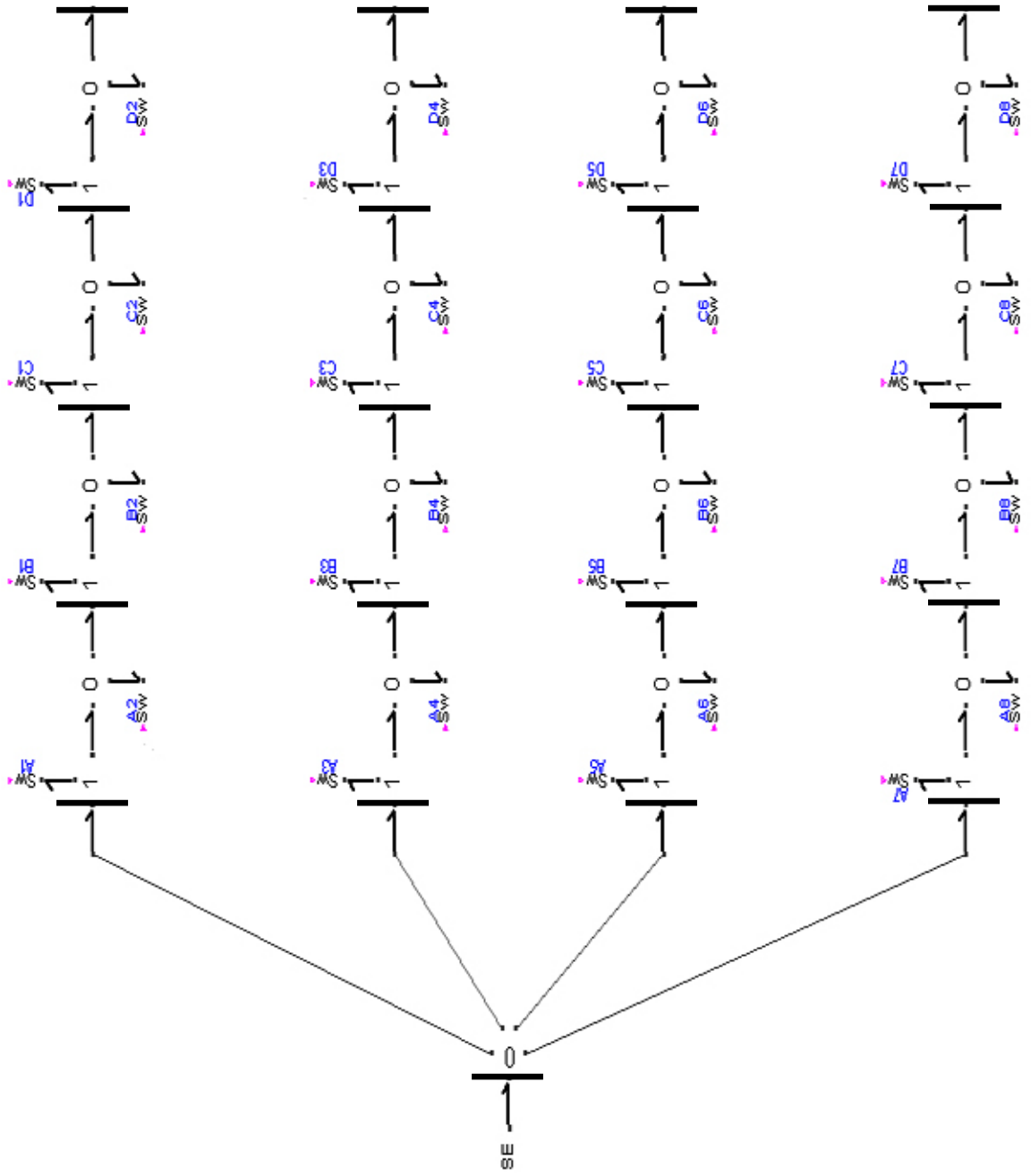


Fig. 3.11: Modified switchbox; Bond graph representation.

In the Bond graph diagram depicted in Fig. 3.11, the type 1 junction with four switch harpoons in each power path from Fig. 3.5 has been fractured into four 1 type junctions with four type 0 junctions inserted between them. In this case, each type 1 junction will have only one departing bond for a single switch, and another bond leaving towards a neighboring type 0 junction. Departing from each type 0 junction not only is the bond to the next type 1 junction, but also a bond towards a newly introduced switch which operates inversely with respect to the preceding switch connected to the type 1 junction. This newly introduced switch is responsible for guiding the residual effort towards the ground. For example, let us consider the line farthest to the left in Fig. 3.11. This line represents $A\overline{B}\overline{C}\overline{D}$, therefore switch A1 is A, B1 is $\overline{B}$, C1 is $\overline{C}$ and D1 is $\overline{D}$. If at the time of consideration this is the line that transmits power, then $A\overline{B}\overline{C}\overline{D}$ must be "true" hence A1, B1, C1, D1 are closed and A2, B2, C2, D2 (attached to the type 0 junctions) are open, keeping disconnected any path towards the ground in this line. As soon as a gear shift takes place, for instance we shift to the second gear, then power has to flow through the second line and the second line only, thus $\overline{A}B\overline{C}\overline{D}$ becomes "true"; requiring $\overline{A}$ being "true" as a necessary condition which in turn means that switch A1 in the first line must open and other switches in that line remain untouched. Since B1 continues to remain open, therefore there exits a segment between two open switches A1 and B1 which previously transmitted power and now suddenly has been disconnected from everywhere. But our recent supplement now takes care of this issue. A2 now closes and links this segment to the ground, therefore "zeroing" the leftover effort. Note that since without a loss of generality we can assume the ground to have zero voltage, the "ground element"

and the bond connected to it have been erased. The causality bars which do not change during the switching procedure have been depicted. Therefore according to this arrangement, the causality bars for the type 0 junction located between the engine and the switchbox have to be fixed. Hence we have been able to generate a logically correct switching network with Bond graphs.

At the end, it is interesting to remark that Multi-step shiftable toothed gear transmissions operate with a very high efficiency of $\eta \geq 0.98$. The helical gear-set is less efficient than the spur gear-set due to sliding friction along the helix angle, but is used in modern transmissions for quiet operation.

3.1-3. *Further Considerations in Modeling*

To this point, we mostly talked about the switch element itself. Other issues regarding the analogy of mechanical and electrical elements must be taken into consideration. For example, let's look at two electrical circuits: the first circuit consists of an ideal capacitor, an effort source and a switch, as in Fig. 3.12. If the switch is closed, the simulating software would complain about a problem.

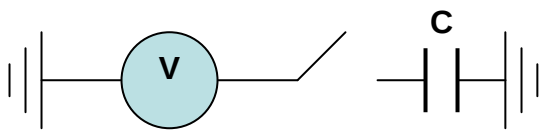


Fig. 3.12: Effort source and an ideal capacitor.

From basic circuitry we know that when we close the switch, we are essentially imposing a finite amount of voltage (effort) to go across the capacitor in an *infinitesimal* amount of time, or actually in *no time*. This would require an *infinitely large* amount of

current (flow), and it will burn the capacitor. Analogous to this case in a mechanical system, if we have a mass-less ideal spring (i.e. no damping) with one side fixed and we suddenly exert a finite force (effort), it means that we require the spring to suddenly stretch from l_0 to l_1 in *zero amount of time*, and that consequently means its velocity (flow) of deformation has to be *infinitely large*; since we assumed no resistance to deformation.

In reality neither a non-resistive spring nor a non-resistant capacitor exists. For charging a capacitor, the voltage is gradually increased, in addition to the *non-idealness* nature of the capacitor; meaning that even the most ideal capacitor has a little amount of resistance, as demonstrated in Fig. 3.13.

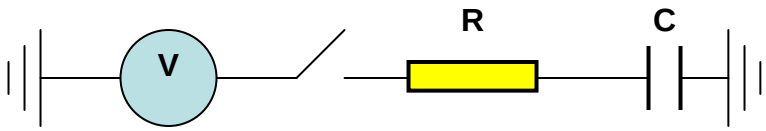


Fig. 3.13: Effort source and a non-ideal capacitor.

By introducing an initial small amount of resistance to avoid the sudden "jump" that we discussed earlier, such a circuit implemented in Dymola is shown in Fig. 3.14, with C = 0.001 F, R = 0.0001 Ω, and the initial effort to be 100 V. Initially the switch is closed, and a Boolean step function opens the switch at $t = 1.5$ s. In Fig. 3.14, the causality strokes have been depicted for the elements that have fixed causality. The resistor must have free causality in order to maintain the causality rule for the type 1 junction when the switch changes position.

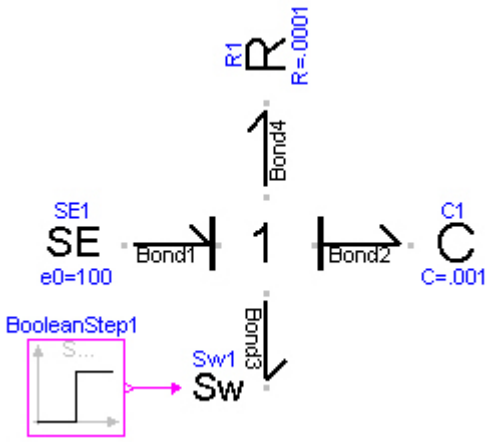


Fig. 3.14: Effort source and a non-ideal capacitor in Dymola in a switching circuit.

The simulation result shows an initial ramp in the capacitance effort, which is magnified in Fig. 3.15 for clarity. Notice that the ramp exists only for a very short period of time during $0 < t < 0.004\,\text{s}$.

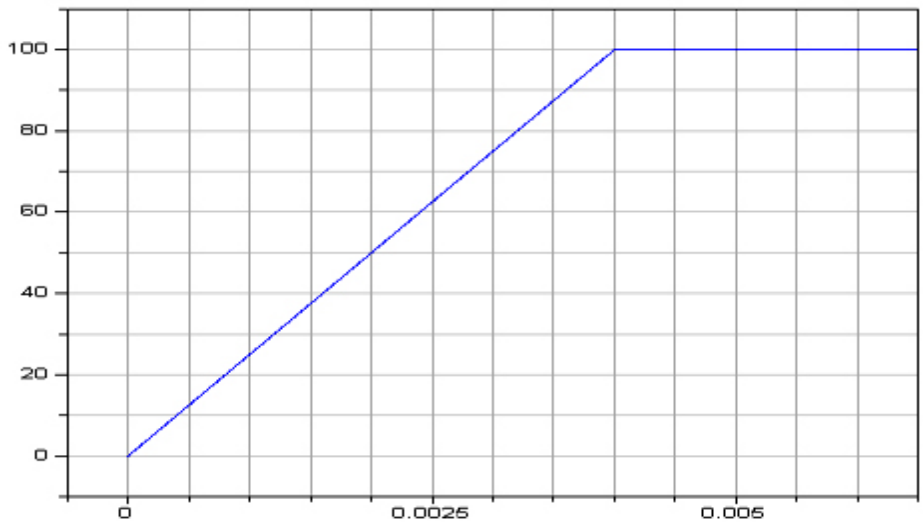


Fig. 3.15: Effort variable of small resistance in a non-ideal capacitor.

Also notice that there is no oscillatory behavior: in the absence of mass, no oscillation is possible, even with a spring (which is assumed to be mass less). One important issue

that must be realized is that although the configuration in Fig. 3.14 is considered to be a "series" configuration in the electrical circuit, nevertheless is "parallel" in its mechanical equivalent, since the capacitance C is analogous to 1/*k*. Had we assembled the ideal spring and damper in series as Fig. 3.16 displays, still the aforementioned problem would exist.

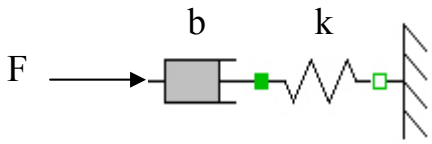


Fig. 3.16: Effort source (*force*), damper and spring in a "series" mechanical configuration.

With a finite force applied to the system in zero time, the damper would initially behave like a rigid body, fully transmitting the applied force to the spring in zero time, again resulting in a demand for an infinitely fast deformation in the spring. As a matter of fact, it is this very quality of the damper which we harness by placing it in parallel with the spring: its resistance to deformation.

The second circuit of interest is composed of an ideal inductance with a switch connected to an effort source, depicted in Fig. 3.17.

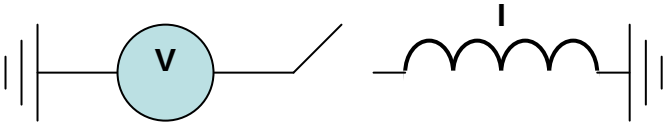


Fig. 3.17: Effort source and an ideal inductor.

If the model is constructed initially with a closed switch, again from basic circuitry we know that as soon as an attempt is made to open the switch, the simulating software

prompts an error message; because abrupt opening of the switch means we are demanding the current (flow) in the inductor to reach zero in *zero time*. Since this is not possible, an electric arc will be drawn as the switch is opened, burning the switch. Equivalently in a mechanical system, this corresponds to a case where an external force is being exerted on a mass in a frictionless environment (since we included no resistance), then by suddenly removing the force, we demand the velocity (flow) of the mass to come to zero in *zero time*, which consequently requires a force (effort) *infinitely large*. This problem could be resolved by adding an appropriately placed resistance in the circuit as illustrated in Fig. 3.18. The mechanical interpretation of this circuit denotes that after the force is suddenly removed, the velocity (flow) of the mass gradually diminishes due to the friction (resistance).

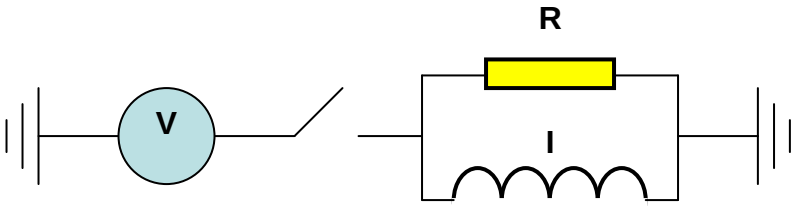


Fig. 3.18: Effort source and a non-ideal inductor.

As an example entailing all the elements, let us model a spring and a damper in series configuration, connected to a mass. The resistor represents the damper, with the damping value set to 5 kg/s. The capacitance is the spring with the stiffness of 1000 N/m, inverted in the electrical system would be 0.001 F. The inductance characterizes the mass of 10 kg. The source of effort here is the weight which is active due to the presence of gravity. With the gravitational constant set to be 9.8 m/s^2, the weight is 98 N. The model in

Dymola would be as in Fig. 3.19.

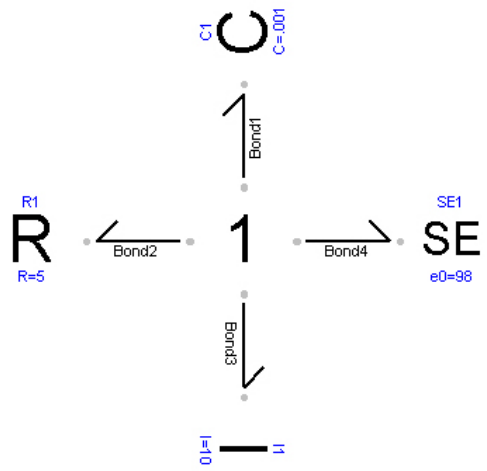


Fig. 3.19: Spring-mass-damper system with gravity, Dymola representation.

The initial conditions are arbitrarily set to differ from those of static equilibrium; therefore obviously the system will oscillate with an exponentially governed envelope in a declining sinusoidal motion.

Now we desire to run a simulation in which the gravity is included for some part of the time, and then the gravity is suddenly discarded, i.e. “switched off”. By simply implementing a switch, we obtain the configuration in Fig. 3.20. A Boolean step function is attached to the switch in order to open the switch at $t = 5\,\mathrm{s}$.

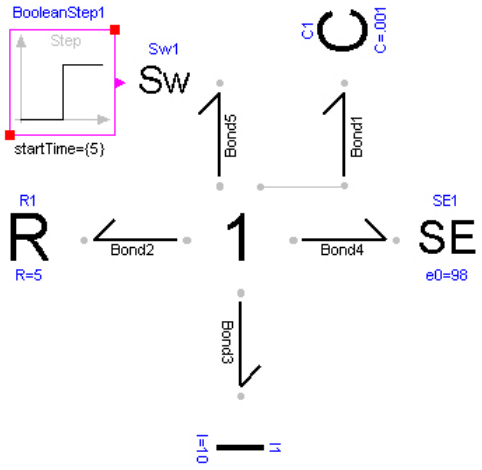


Fig. 3.20: Spring-mass-damper system with switching gravity, Dymola representation.

Though at the first glance this configuration appears to be flawless, it is not so. Dymola will generate an error message, occurring at the 5th second. In order to better visualize the error, let us translate this model into a circuit diagram sketched in Fig. 3.21.

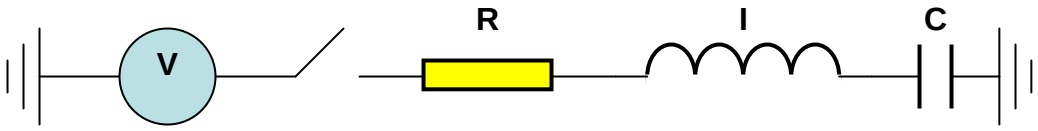


Fig. 3.21: Circuit diagram with floating elements.

According to our previous discussions, the mystery of the error is unraveled. When the switch is opened, the inductance on the right side of the switch will be floating. One way to resolve this defect is by introducing another switch connected to the ground placed between the existing switch and the resistor with open/close states opposite to the inline switch. By doing so, the problem of the floating element is resolved. The

realization in Dymola will be as demonstrated in Fig. 3.22.

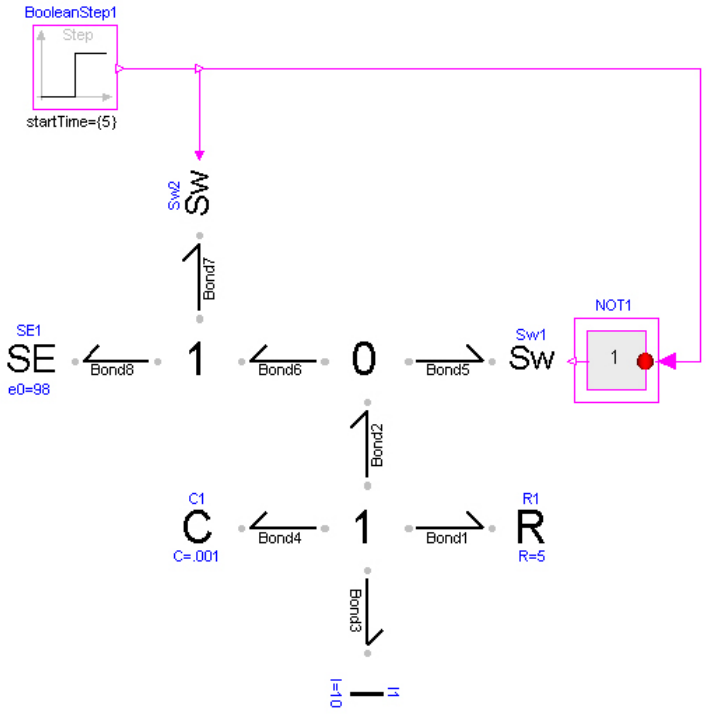


Fig. 3.22: Corrected model in Dymola representation.

Hence we conclude that in modeling even a mechanical system, we do not have complete freedom regarding the topological arrangement of our desired components. Residual voltage/currents of the model's dual representation using electrical elements must be taken care of.

3.2. *PILOT EJECTION SCENARIO*

We wish to look into yet another scenario, encompassing further details in Bond graph modeling of switching systems. The example discussed here is a simplified pilot

ejection model in a two dimensional motion, composed of four stages: in the first stage, the pilot seat is placed upon a pre-stressed spring in an intact configuration with the airplane. The second stage describes the time span when the spring is released, pushing the seat upwards along the guiding rail of the seat. In the third stage the contact between the spring and the seat has ended, yet the seat is still traveling upwards along the rail. In the last stage, the seat has left the airplane, undergoing free motion.

The difference between this case and the gearbox structure is that here the act of switching between separate stages represents a non-reversible shift in time for the same system, therefore causing "time" to be an explicit variable, whereas in the case of the gearbox, there always exists a physically different line of power flow regardless of being operational or not, with "time" being an implicit variable in the process of the system.

3.2-1. *Pilot Ejection: Phase One*

During the first phase, the seat is attached to the plane while pressurizing the spring underneath it. We assign the *x* direction to be the horizontal direction of motion and *z* to be the vertical. With t_1 being the time when the seat ejection system is triggered, the equations of motion prior to ejection are:

$t < t_1$:

$$(M+m)\cdot \ddot{x}_M = F_x - B_x \cdot \dot{x}_M^2 - \overline{F_z} \tag{3.3}$$

$$(M+m)\cdot \ddot{z}_M = F_z - B_z \cdot \dot{z}_M^2 - (M+m)\cdot g \tag{3.4}$$

$$x_m = x_M \qquad\qquad z_m = z_M - z_0 \tag{3.5}$$

where F_x is the aircraft thrust generated by the engines; F_z represents the aerodynamic

lift, which results from the partial conversion of engine thrust into a vertical force due to the wings' geometry, therefore originally caused by a horizontal force symbolized with $\overline{F_z}$. Since the lift force (an effort variable) stems from the horizontal velocity (a flow variable), we implement a gyrator in the model to demonstrate this phenomenon. B_x and B_z are the aerodynamic friction coefficients. The vertical coordinates of the aircraft and pilot seat are z_M and z_m respectively, and the spring's free length is assigned to be z_0. Modeled in Bond graphs, hands in the schematic in Fig. 3.23.

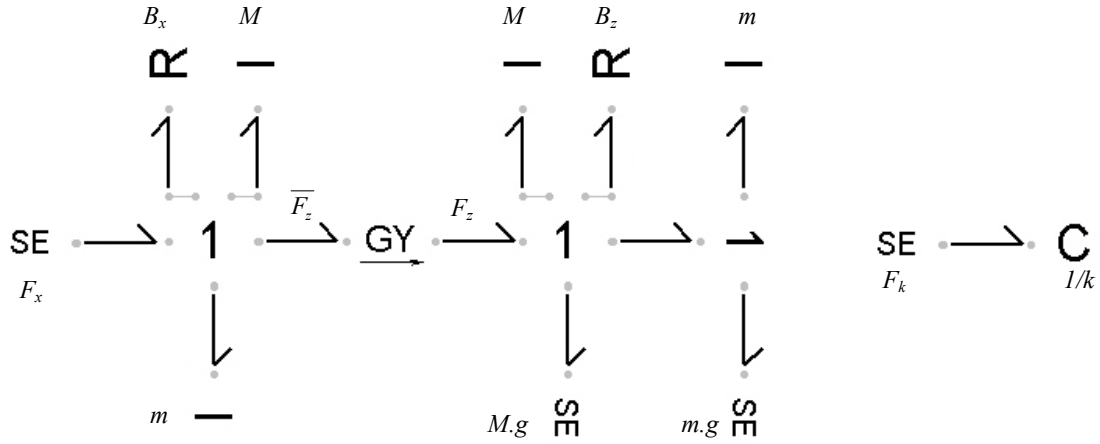


Fig. 3.23: Pilot ejection first phase, Bond graph representation.

Notice that the Bond graph on the right with the effort source and capacitor is separate, expressing the spring and the initial force for pressurizing it. The Bond graph on the left is composed of two segments: To the left of the gyrator is a type 1 junction surrounded by the bonds and elements which conceptualize the forces acting on the airplane in the *x* direction; namely the effort source as the engine thrust, the resistance as the horizontal friction, and two inductors corresponding to the mass of the airplane and

the mass of the pilot seat. On the right side of the gyrator there are two type1 junctions which logically can be amalgamated into one; the reason for using two junctions is solely for visual clarity. Forces acting in the z direction are located around these type 1 junctions: two effort sources representing the weight of the airplane and pilot seat, two masses for the airplane and pilot seat due to their involvement in the vertical direction of motion, and the vertical friction coefficient denoted by the resistance element.

3.2-2. *Pilot Ejection: Phase Two*

The second phase starts when the spring underneath the pilot seat is released, enforcing an upward motion on the seat along the guiding rail. The equations for this phase are:

$t \in [t_1, t_2]$:

$$(M+m)\cdot \ddot{x}_M = F_x - B_x \cdot \dot{x}_M^2 - \overline{F_z} \tag{3.6}$$

$$M \cdot \ddot{z}_M = F_z - B_z \cdot \dot{z}_M^2 - M \cdot g - k \cdot (z_M - z_m) \tag{3.7}$$

$$x_m = x_M \tag{3.8}$$

$$m \cdot \ddot{z}_m = k \cdot (z_M - z_m) - b_z \cdot \dot{z}_m^2 - m \cdot g \tag{3.9}$$

where b_z denotes the friction coefficient between the seat and its rail. The Bond graph for this phase is depicted in Fig. 3.24. As in the previous graph in Fig. 3.23, the mass of the seat and aircraft have the same motion in the x direction, hence the inductors representing the horizontal contribution of these masses are located around the same type 1 junction. The difference with the preceding stage is in the z direction, where the seat has an additional upwards movement relative to the aircraft trajectory, accompanied with

a newly introduced friction along the guiding rail.

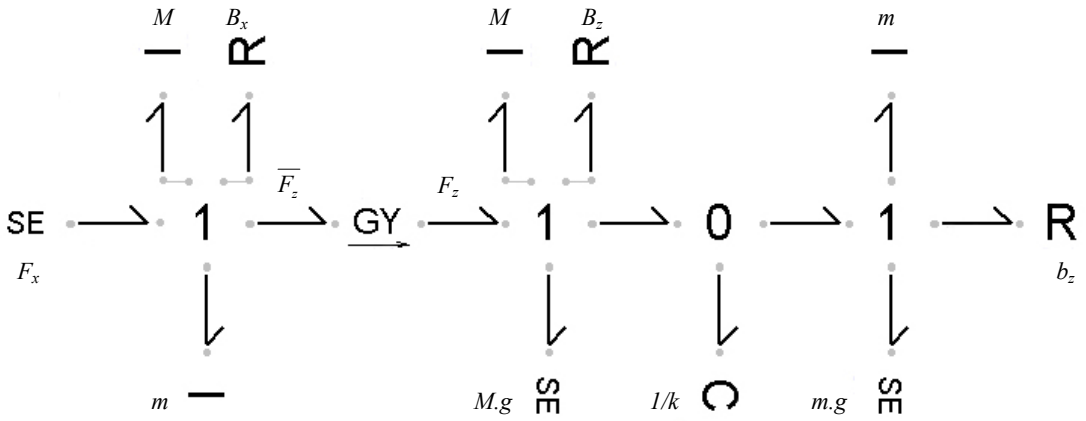


Fig. 3.24: Pilot ejection second phase, Bond graph representation.

The type 0 junction demonstrates the connection between the seat and the aircraft through the spring. As long as $k \cdot (z_M - z_m) > 0$, the spring and seat maintain contact.

3.2-3. *Pilot Ejection: Phase Three*

In the third phase, the contact between seat and spring has ended, while the seat continues its upward motion along the guiding rail. The only forces acting on the seat are its weight and the friction caused by sliding along the rail in the z direction. The equations of motion are:

$t \in [t_2, t_3]$:

$$(M+m) \cdot \ddot{x}_M = F_x - B_x \cdot \dot{x}_M^2 - \overline{F_z} \tag{3.10}$$

$$M \cdot \ddot{z}_M = F_z - B_z \cdot \dot{z}_M^2 - M \cdot g \tag{3.11}$$

$$x_m = x_M \tag{3.12}$$

$$m \cdot \ddot{z}_m = -b_z \cdot \dot{z}_m^2 - m \cdot g \tag{3.13}$$

As in the first phase, this stage also requires two separate Bond graphs, as shown in figure 3.25. The Bond graph on the right describes the seat motion in the z direction.

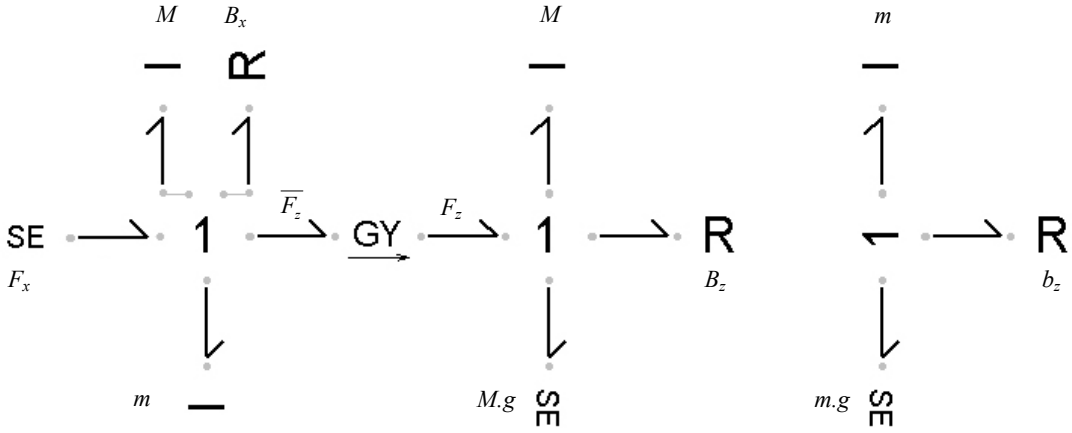


Fig. 3.25: Pilot ejection third phase, Bond graph representation.

Notice that due to the exceedingly minor effect which the seat-rail friction inflicts on the vertical motion of the aircraft, this connection has been neglected and the vertical motion of the two bodies is considered totally disjoined, therefore the vertical friction due to the seat's upward motion along the rail has only been included for the seat model; denoted by the resistor. The mass of the seat is still included in the type 1 junction to the left of the gyrator due to its effect on the aircraft's horizontal motion.

3.2-4. *Pilot Ejection: Phase Four*

Finally, the upward shift of the seat results in its total disengagement from the plane, and consequently exhibits free motion. The airplane and seat motions are described by the following set of equations:

$t > t_3$:

$$M \cdot \ddot{x}_M = F_x - B_x \cdot \dot{x}_M^2 - \overline{F_z} \tag{3.14}$$

$$M \cdot \ddot{z}_M = F_z - B_z \cdot \dot{z}_M^2 - M \cdot g \tag{3.15}$$

$$m \cdot \ddot{x}_m = -b_x \cdot \dot{x}_m^2 - \overline{f_z} \tag{3.16}$$

$$m \cdot \ddot{z}_m = f_z - b_z \cdot \dot{z}_m^2 - m \cdot g \tag{3.17}$$

The two-dimensional motion of the seat is portrayed by the Bond graph on the right side in Fig. 3.26. Similar to the aircraft model, the seat model is composed of two type 1 junctions (each for dealing with one dimension of motion) interconnected by a gyrator.

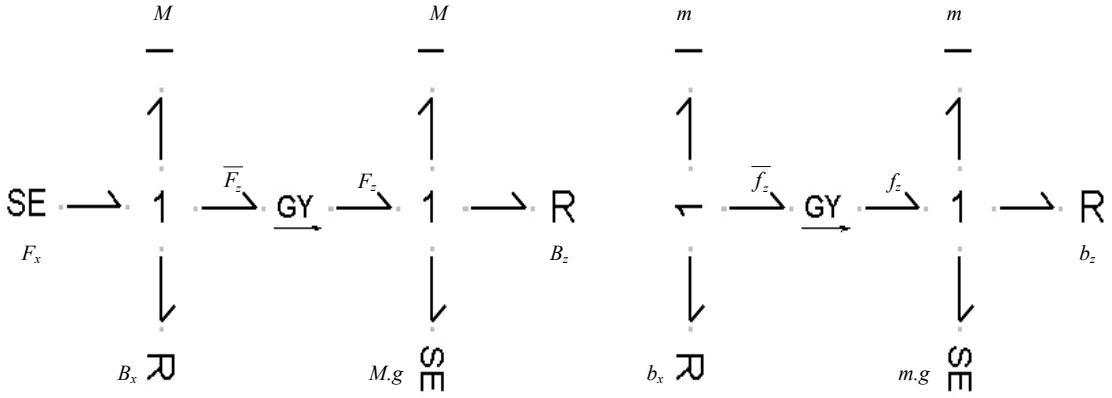


Fig. 3.26: Pilot ejection fourth phase, Bond graph representation.

In this stage, b_z is the frictional factor for the seat motion in air. Contrary to the prior stages, it is plainly visible that the entire dynamic properties of the seat and the aircraft are decoupled. The horizontal motion of the seat has very little contribution to the vertical motion, which can easily be taken care of by the conversion factor of the newly implemented gyrator.

3.2-5. *Original Subsystems Combined via Switching Algorithm*

Now we attempt to generate one integrated model instead of dealing with four separate models. We employ the derived automatic method for modeling this problem. It seems clear that without a consistent algorithm, the integration of different operational stages of a model requires a lot of intuition, involving a time consuming effort in a heuristical manner with the obvious possibility of errors, especially as the dimensionality of the problem grows. In this section we shall witness the versatility of the proposed method by overcoming the drawbacks mentioned.

This problem is made up of four stages which operate consequentially thus making it similar to the gearbox problem in the logical sense; combined of four power paths which come into operation one at a time. The entire model is sketched in Fig. 3.27. The farthest power line to the left is the first stage, followed by the subsequent stages in their order of activation. The engine's force is represented by the effort source at the bottom of the figure, connected to a type 0 junction which branches of into the four power lines, directing power into each of the paths at a time by an appropriate switching plan. Notice that again this model is not ready to be simulated since the triggering devices for the switches have been left out due to space limitation; more importantly some issues regarding the initial conditions need additional clarification, which will be explained in the next chapter.

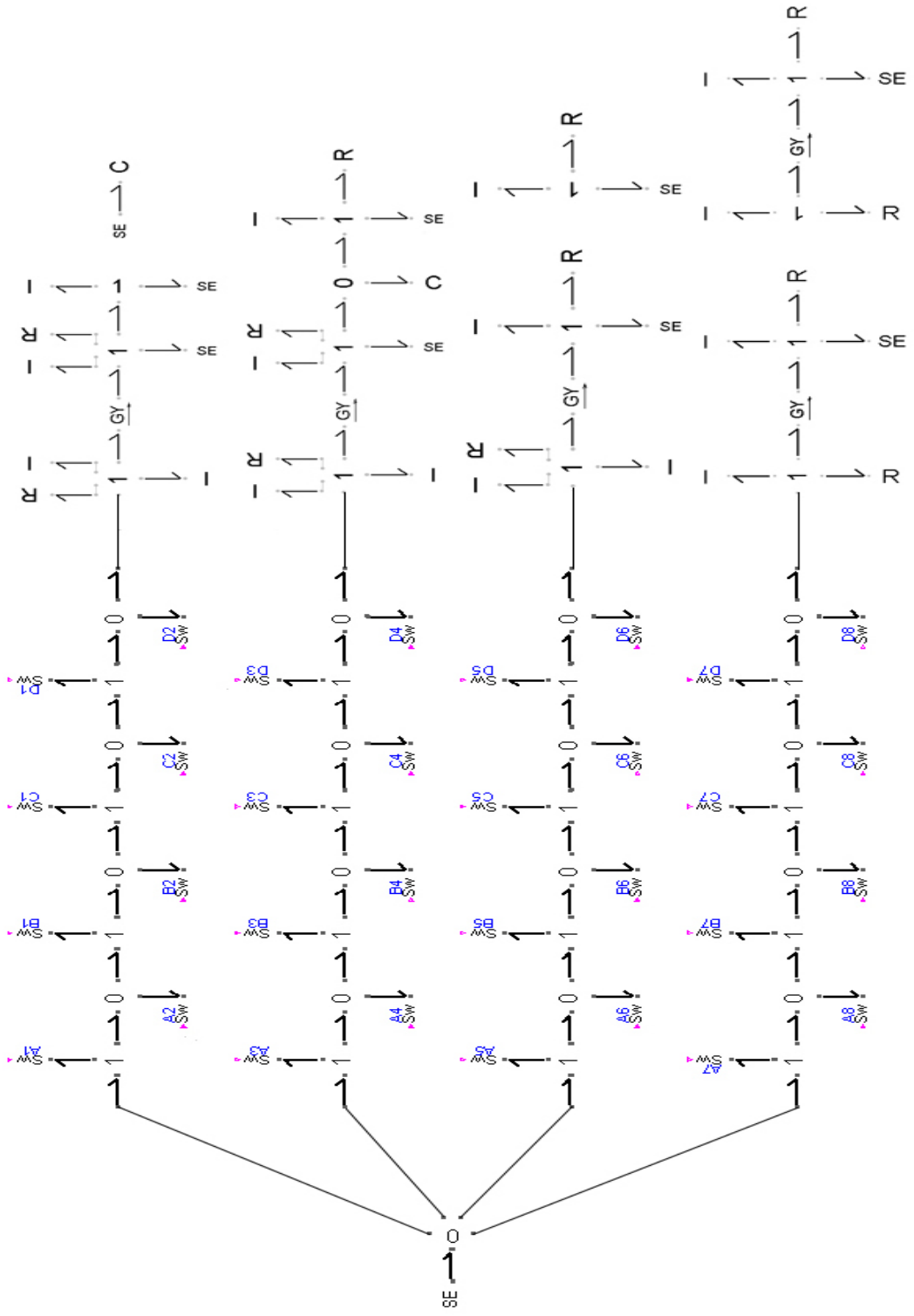


Fig. 3.27: Pilot ejection complete model, Bond graph representation.

In this chapter we were able to develop a systematic method for modeling switching systems in Bond graphs. We presented two examples which incorporated mutually exclusive switches, thus resulting in the maximum number of switch elements. In the next chapter, the complete simulating scheme for both models, gearbox and pilot ejection, will be discussed, followed by additional conclusions.

4. *SIMULATION AND ANALYSIS*

The previous chapter presented a practical method for modeling switching networks with Bond graphs by means of two examples. As we mentioned there, those models were not entirely ripe for simulation yet, and required some additional tweaks. This chapter is organized in three sections. The first section is a short introduction to the simulating platform; Dymola. In the second and third sections, the four stage gearbox and the pilot ejection scenario from the previous chapter are regulated and thereafter simulated.

4.1. *INTRODUCTION TO DYMOLA* [TM 1]

Dymola [2, 8, and 28] is a general-purpose object-oriented language for modeling physical hybrid systems, initially coded by Elmqvist [10]; which has now been developed into a powerful commercial software with abilities such as 3D animation and real-time simulation; open for user defined model components and interfaces to other programs.

Dymola uses hierarchical object-oriented modeling to describe in detail the systems, subsystems and components. This facilitates testing of coordinated control systems against integrated models. The models are written in the open, object-oriented modeling language Modelica, and are hierarchically decomposed into sub models; usually built graphically by dragging component models (from the Modelica standard library, from other free libraries, or component models developed by the user) and connecting them. Model libraries are available for electronics, rotational, translational and 3D mechanics,

[1] Dymola is a registered trademark of Dynasim, Inc.

thermodynamics, hydraulics, control, etc. The libraries range from basic components to more specialized domains such as the Power Train library, and can be expanded with user-written model components. Models are either composed of other more primitive components or described by equations at the lowest level, making possible true reuse of models in different contexts. The equation-based nature of Modelica is essential for enabling truly reusable libraries. Reuse of modeling knowledge is supported by use of libraries containing model classes and by using inheritance. Connections between sub-models are conveniently described by defining connectors that model physical couplings. Measurement data and model parameters cover additional model aspects. Mass and inertia of 3D mechanical bodies can be imported from CAD packages, and visual properties may be imported in DXF and STL format. The icons of model components are defined either by drawing shapes in Dymola, or by importing graphics from other tools in bitmap format. Dymola also offers a matrix manipulation language for matrix and vector operations by a command line interface that allows the user to control the Dymola compiler.

Model details are given by ordinary differential equations and algebraic equations; DAEs (Differential-Algebraic Equations). The user need not convert the equations to assignment statements. Matrix equations facilitate convenient modeling of 3D mechanical systems, control systems, etc. The Dymola language is not a procedural language, but rather “non-causal”; however user-defined Modelica functions also support procedural modeling.

Symbolic processing is used to make simulations efficient. Dymola converts the

differential-algebraic system of equations symbolically to state-space form, i.e. solves for the derivatives. The usual need for manual conversion of equations to a Block diagram is removed by the use of automatic formula manipulation. Efficient graph-theoretical algorithms are used to determine which variable to solve for in each equation, and to find minimal systems of equations, using tearing, that need to be solved simultaneously (algebraic loops). If the solution is non-linear, a non-linear equation solver will automatically be called upon. The equations are then, if possible, solved symbolically. Otherwise code for efficient numeric solution is generated. Higher index DAEs, typically obtained because of constraints between sub-models, are handled by symbolically differentiating equations. Dymola automatically generates the needed time and state events to handle instantaneous equations. Hence the modeler is relieved from gathering all the model's equations manually.

Dymola transforms a declarative, equation-based, model description into efficient simulation code. Advanced symbolic manipulation (computer algebra) is used to handle very large sets of equations. Dymola provides a complete simulation environment, but can also export code for simulation in Simulink. In addition to the usual offline simulation, Dymola can generate code for specialized Hardware-in-the-Loop (HIL) systems. Experiments are controlled with a Modelica-based scripting language, which combines the expressive power of Modelica with access to external C libraries; e.g., LAPACK. The built-in plotting and animation features of Dymola provide the basis for visualization and analysis of simulation data. Experiments are automatically documented with logs of all operations in HTML format, including animations in VRML (Virtual

Reality Modeling Language) and images.

Discontinuous equations are automatically handled by introducing discrete events as required by numerical integration routines. Modelica also supports instantaneous equations to model friction, impacts, difference equations, etc. When Dymola encounters a state variable, the compiler integrates the state variable using any integration method selected by the user. Discrete events have an effect on the nature of the integration, in the sense that a discrete event placed in the routine is viewed by the compiler as a set of new initial conditions, from where the integration is carried on accordingly hence forth. However if this does not occur, then the step size control must make the step size small enough to capture the discontinuity.

The Dymola Bond graph library was introduced by Cellier as a part of the graphical modeling environment. This library was created using the Dymola language itself for producing new "models"; not by adding an extra paradigm to the underlying language. Hence Dymola doesn't know anything about Bond graphs; it only deals with these newly defined "models". Since Dymola is perfectly capable of determining the correct causalities, there is no need to use causal bonds, however this feature exists. For further information, the reader is referred to [5].

4.2. *GEARBOX SIMULATION*

4.2-1. *Preliminary Test: Things to Be Careful About*

In the first attempt to test the validity of the developed Bond graph switching

network, an experimental model presented in Fig. 4.1 was initially assembled in Dymola (triggering devices for the switches have been left out of the picture for more clarity). It can be seen in this figure that the power paths ultimately lead to a terminal type 1 junction linked to a common element; here being a resistor for simplicity. The switches are set to operate consecutively; i.e. at the same time instant which the preceding switch opens, the next switch closes, leaving no operational dead zones.

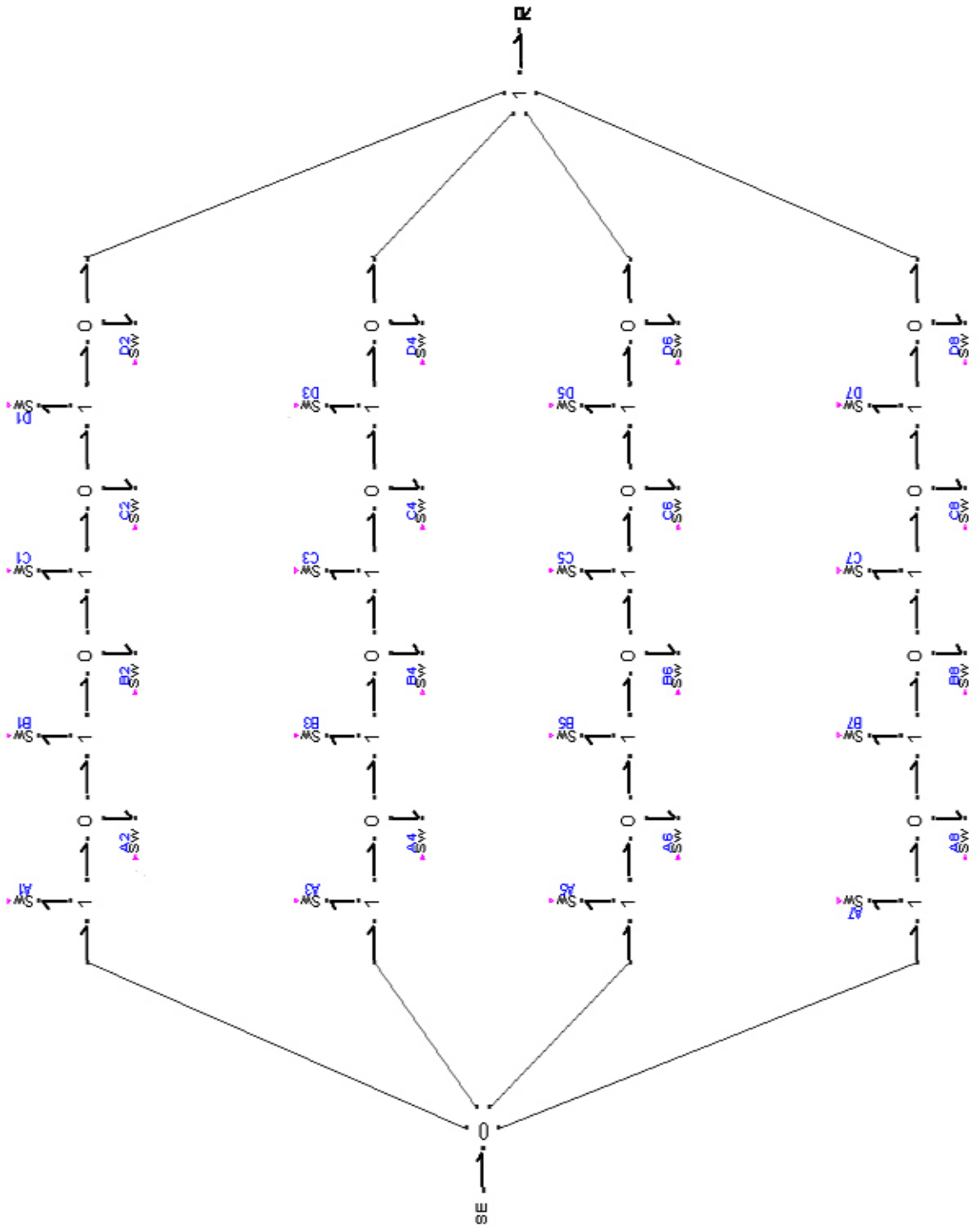


Fig. 4.1: Test with common resistor.

As a second configuration, the terminal type 1 junction with the common resistor is removed and instead, a resistor is implemented separately for each power path, as partially sketched in Fig. 4.2.

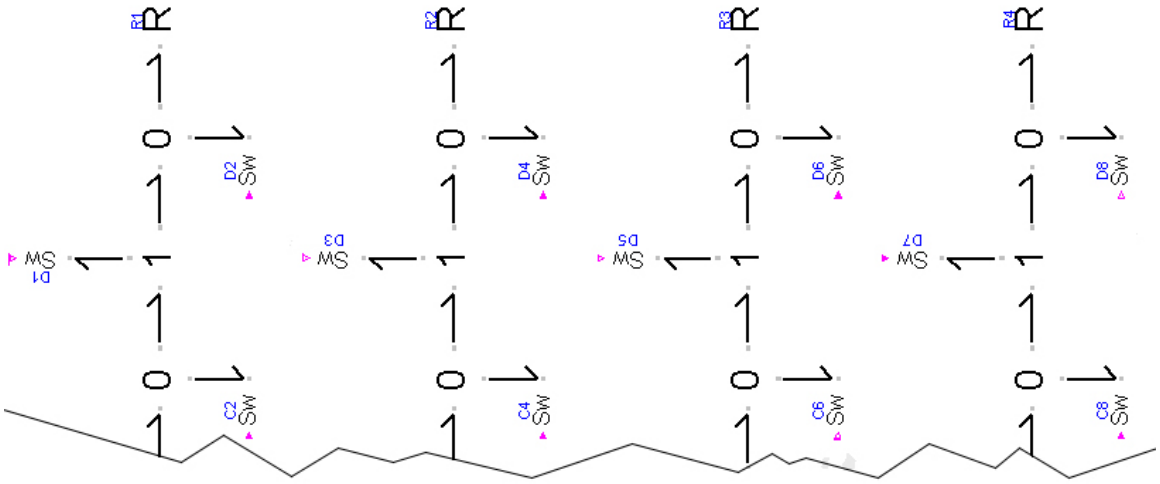


Fig. 4.2: Test with terminally decoupled lines.

Although by carefully defining the switching scheme, we seemed to have avoided a logical superimpose of line activity in Fig. 4.1, this may not be entirely true from the numerical aspect (depending on the simulation software that the modeler employs), since the model is being simulated digitally; hence with discrete "time steps". In the first model, if two (or more) power paths are simultaneously active even for the slightest amount of time, the flows in the relative paths become indeterminate since all of the lines have identical characteristics, therefore leaving no preference for the current (flow). Consequently if there exists a time span, no matter how small, in which the succeeding line has been activated while the preceding line hasn't been disengaged yet, the simulation will fail. This problem can be resolved with the second model by decoupling

the power paths and making them independent, since topologically both descriptions in Fig. 4.1 and 4.2 are the same. However both configurations were simulated with Dymola without any errors, due to the software's capability of handling these events.

4.2-2. *Simulation and Results*

The complete model used for simulation is illustrated in Fig. 4.4. Each power line ends in an inductance representing the mass of the leading gear in mesh, followed by a transformer; as partially seen in Fig. 4.3. The four transformers employed in the model denote four conversion ratios for the torque and angular velocity, with the ratios: 0.25, 0.5, 0.75 and 1. After the transformer, an effort source is placed symbolizing a 500 N-m load along with another inductance for the following gear in mesh, and finally a resistor which takes into account minor damping effects. All inertias are set to be 2 kg.m^2. The engine is assumed to produce a constant 10,000 N-m effort.

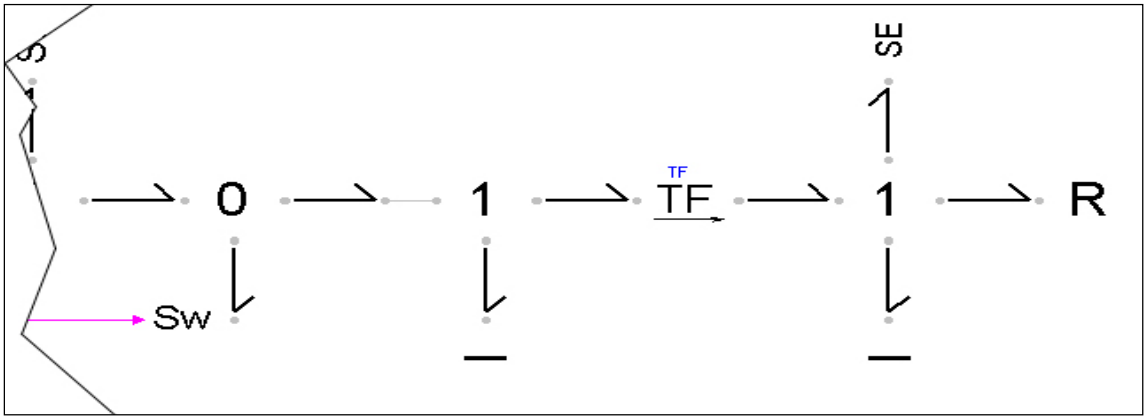


Fig. 4.3: Partial configuration in each power path.

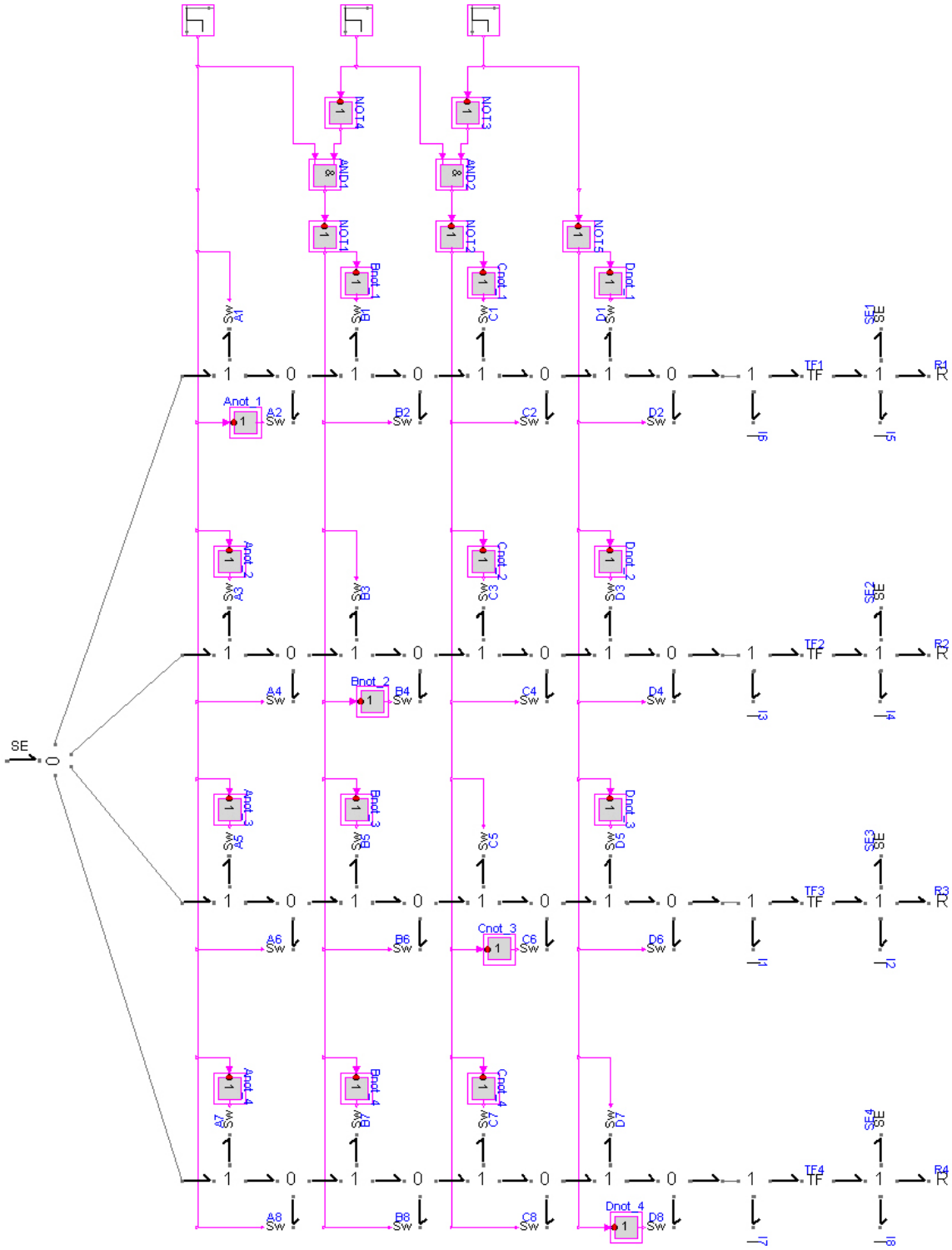


Fig. 4.4: Simplified gearbox.

Note that in the present configuration, the engine has been modeled merely with an effort source, which only corresponds to the engine's torque. A more complete model entailing the engine's inertia and friction coupling will be introduced later on; left out here due to the priority of obtaining a basic topologically-correct model.

The model is simulated for 8 seconds. Gear shifting takes place in the 2nd, 4th and 6th seconds. Dymola provides the capability to plot the flow and effort for every element in the model. For instance, the angular velocity (flow) for the engine (effort source) vs. time is depicted in Fig. 4.5.

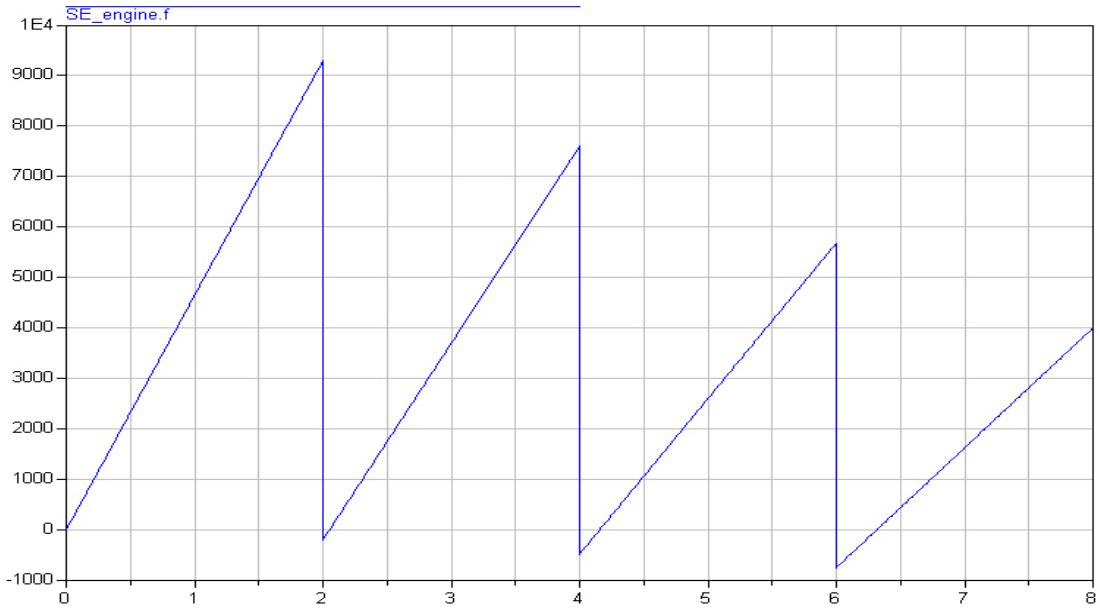


Fig. 4.5: Engine (effort source only): angular velocity (flow variable) vs. time; similar loads.

As it can be seen, the angular velocity starts off from a zero value and increases till the next switching point, at which time it abruptly drops and starts a new incline during the following gear stage. This behavior can be attributed to the fact that the driving effort source only represents the engine torque. In a real engine, the power cannot be instantaneously set back to zero due to the inertia associated with the rotating mass of the

engine assembly. In our the engine model, at each switching occurrence we would expect the angular velocity to sharply drop down to zero, simply because there is no "memory" (inertia) associated with it. However, the graph in Fig. 4.5 clearly shows a declining envelope for the minimum points (i.e. the switching instances); resulting in the angular velocity to be less that the expected zero value at these points. In the next test, we increase the loads to 4000, 3200, 2600 and 1800 N-m for the four power lines, respectively. Figure 4.6 demonstrates the graph for the engine's angular velocity (flow) vs. time under new load conditions.

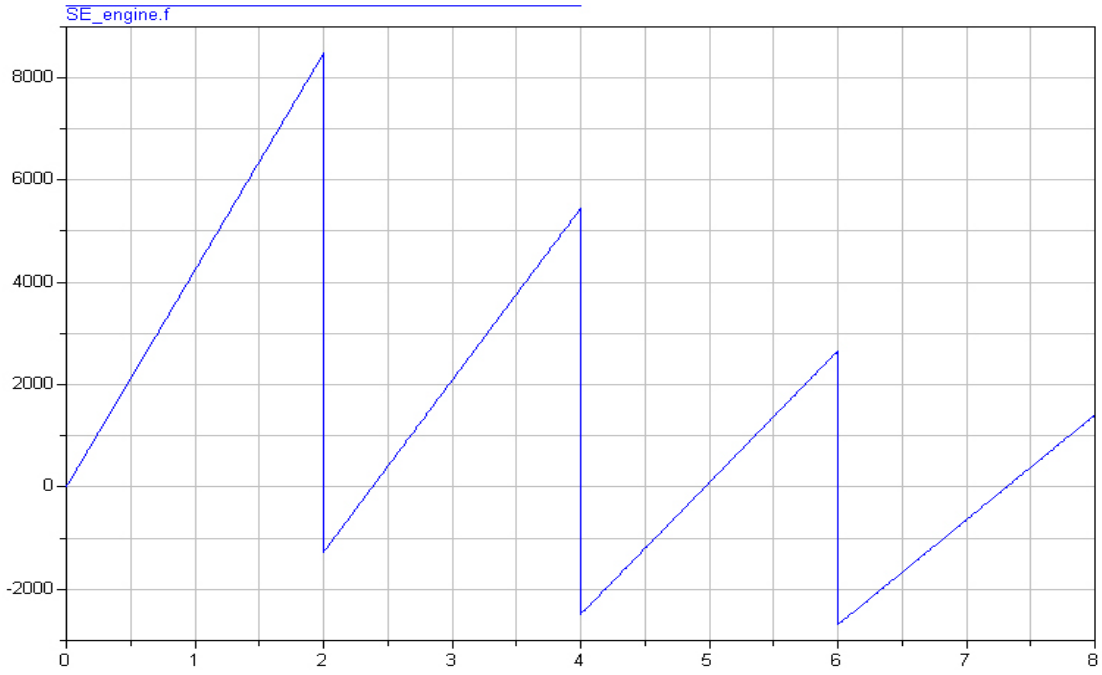


Fig. 4.6: Engine (effort source only): angular velocity (flow variable) vs. time; dissimilar loads.

It is evident that the minimum values have decreased even more. This phenomenon can be traced back to the effort sources acting as loads. For each power line, this load exists and is operational, even prior to the switching time and engagement of that particular line. The results depicted in Fig. 4.7 confirm this hypothesis, where all the

loads are set to be zero; hence the minimum points occurring at the switching instances have all attained a zero value.

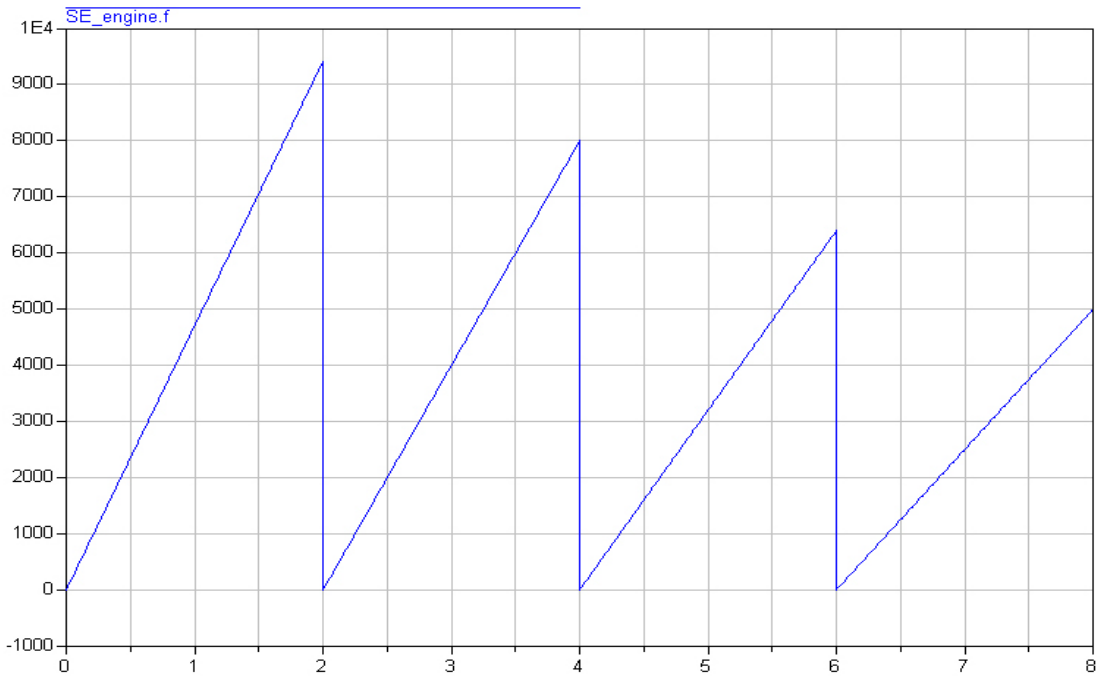


Fig. 4.7: Engine (effort source only): angular velocity (flow variable) vs. time; zero loads.

One way to tackle this problem is by entering the loads into the switching program; i.e. switching them on and off in the model according to the engagement time of their relative gear. Another way is to directly manipulate them through the source code, by setting new initial conditions using "if" expressions or "when" clauses. The next section on the pilot ejection problem discusses the latter approach in full detail; therefore we shall now explore the former method.

The four effort sources acting as loads at the end of each power path in Fig. 4.4 are removed in order to be replaced by a common load for all power lines. We employ a time varying effort source with a sinusoidal effort property. In order to counteract the flaw mentioned earlier; i.e. the undesired effect of the load effort source on a particular power

line prior to its engagement, it is necessary to temporarily “isolate” each power path from the presence of all sources (flow and effort) during its relative disengagement time. Temporary isolation from the effort source denoting the engine has already been taken care of by the original switching network. But we also need to isolate the elements of the currently inactive power paths with respect to the effort source that represents the load. However this task is much simpler than designing the original switching network: all that needs to be done is to implement two switches with an inverse operation scheme at the end of each power line prior to the common load; one switch for grounding the elements during inactivity, and another to establish connection with the load when needed. A complete layout is shown in Fig. 4.8.

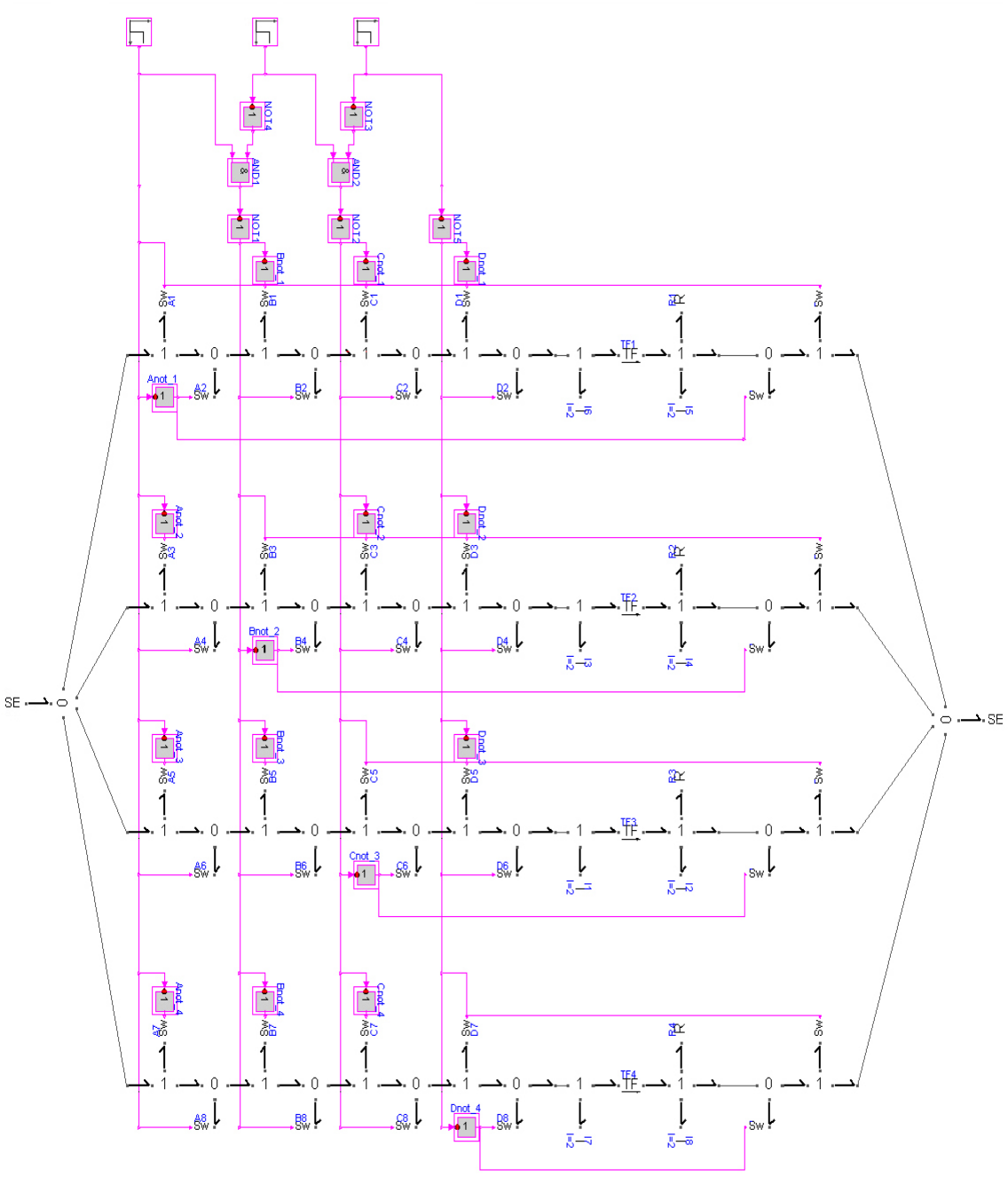


Fig. 4.8: Gearbox with common engine and load.

There are two additional switches at the end of each power path; the first being attached to a type 0 junction (for ground connection) and the second to a type 1 junction

(for load connection). During the disengagement time of a particular power line, the first switch remains closed, connecting the relative elements to the ground. The second switch is open, keeping the elements isolated from the load effort source. When this particular power path becomes active, both switches change position: the first switch opens and the second switch closes, disconnecting that power line from ground and connecting the elements in that line to the load. The angular velocity (flow) of the engine (effort source) and the torque generated by the load are plotted vs. time in Fig. 4.9, with the latter being shown as the dashed curve.

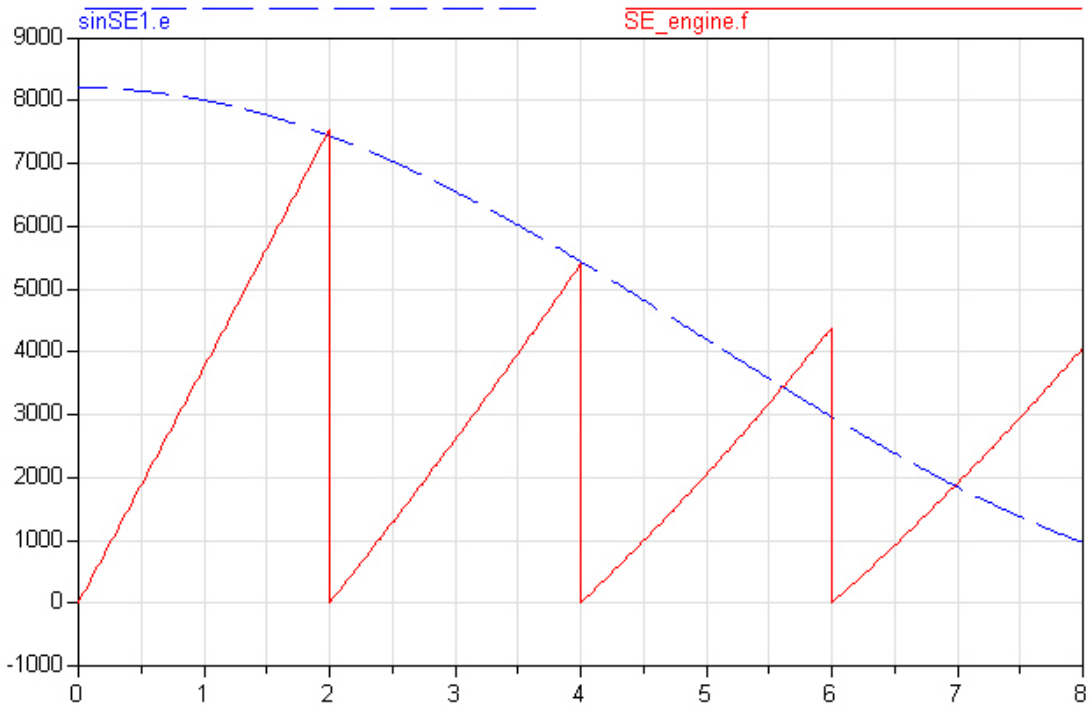


Fig. 4.9: Engine (effort source only) angular velocity (flow) and load torque (dashed) vs. time.

As it can be seen, the minimum values for the engine angular velocity at each switching occurrence have all attained a value of zero due to the two newly introduced switches placed at the end of each power line prior to the load effort source; isolating the

elements of the power paths during disengagement time while grounding the flows/efforts.

Now that we were able to establish an error-free model, we attempt to proceed a step further for rendering the model to be more realistic, by including inertia as part of the engine. As an initial design, an inductor is placed after the engine effort source; connected to a type 1 junction followed by the rest of the gearbox unaltered; as partially shown in Fig. 4.10. Dymola, however, prompted an error message and consequently the simulation failed.

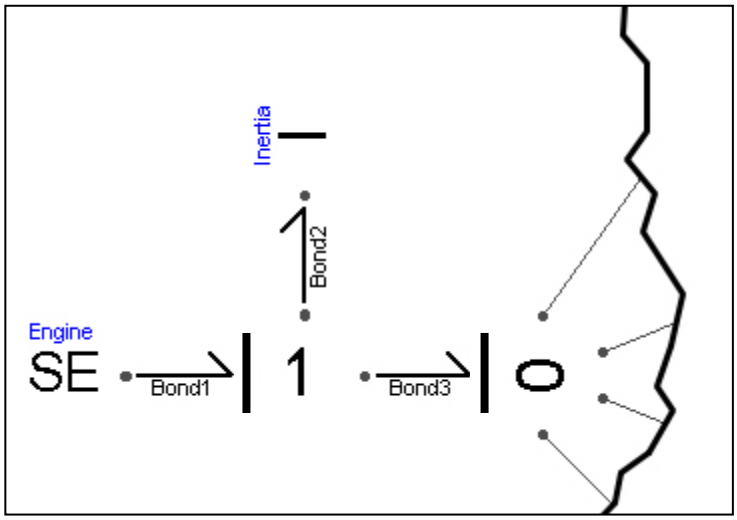


Fig. 4.10: New engine model with inertia included; initial attempt.

In section "3.1-3. *Further Considerations in Modeling*", it was discussed that not any arbitrary arrangement of elements will essentially lead to a logically proper model; an issue which reveals itself in Bond graphs under the form of "causality conflict". After demonstrating the causality bars that remain invariable in Fig. 3.11, we concluded that the type 0 junction which streams power into the branching paths must have fixed

causality bars; namely, the four causality bars on the switch box side must rest away from this type 0 junction, dictating the remaining causality bar on the engine side to be placed at the junction itself. The arrangement in Fig. 4.10 "somewhat" violates this conclusion. In this figure, the causality bar for "Bond1" imparting the effort source rests at the type 1 junction, and as before, we require that the causality stroke for the type 0 junction at "Bond3" to rest at the side of the 0 junction itself. The causality *preference* for the inductor means dragging the causality bar for "Bond2" away from the type 1 junction and placing it on the inductor side, but since the type 1 junction already has one of its causality bars placed away from the junction itself therefore we must compromise our *preference* in order to not have the causality *rule* of the type 1 junction violated. It turns out that giving up this *preference* results in an initially-hidden error, similar to those disclosed in section 3.1-3.

According to the layout in Fig. 4.10, by starting the simulation, the inductor's flow variable which now corresponds to the angular velocity of the engine assembly's inertia (let us call it the "flywheel" for brevity, though we are not being very accurate in using the term) starts to increase as the engine power streams into the first power line; i.e. the "first gear". However, at a switching instance, we are abruptly connecting an already rotating element (the flywheel represented by the inductor) to a series of stationary (or even rotating) elements; i.e. the gear pair in the subsequent power line. Therefore we are imposing a finite jump on the angular velocities of both the flywheel and the gear pair by trying to match the two different angular velocities in *zero time*, which of course is not possible. In an actual automotive driveline system, for instance, we know that at each

switching occurrence (changing gears), there exists a short *slipping* phase, which is made possible by means of the clutch disk in manual transmissions or hydrodynamic coupling in automatic transmissions. In either case, the difference between the angular velocities of the engine and the gearbox is dissipated via irreversible conversion into heat, resulting in a swift synchronization between the two velocities due to *friction*.

The model in Fig. 4.10 is modified by adding a resistor to the engine model, as shown in Fig. 4.11. Included are the causality strokes for the elements in the figure. As it can be seen, we have achieved the desired causalities for all elements.

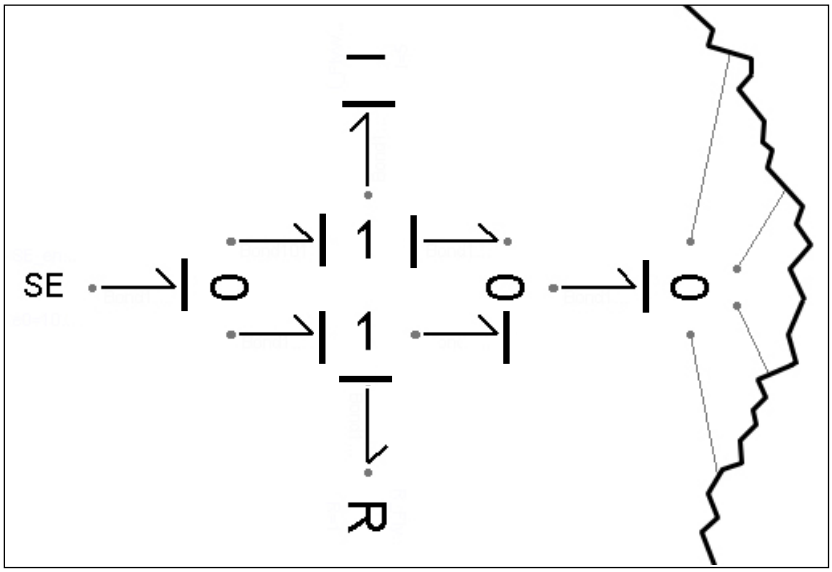


Fig. 4.11: New engine model with inertia and friction included.

With the new engine configuration implemented we simulate the model giving a value of 5 kg.m^2 for the flywheel inertia and 1 N-m.s for the torsional friction. The angular velocity graph of the engine assembly is shown in Fig. 4.12. The curve demonstrates a more realistic behavior compared to the previously obtained saw tooth graph (which still exists for the effort source).

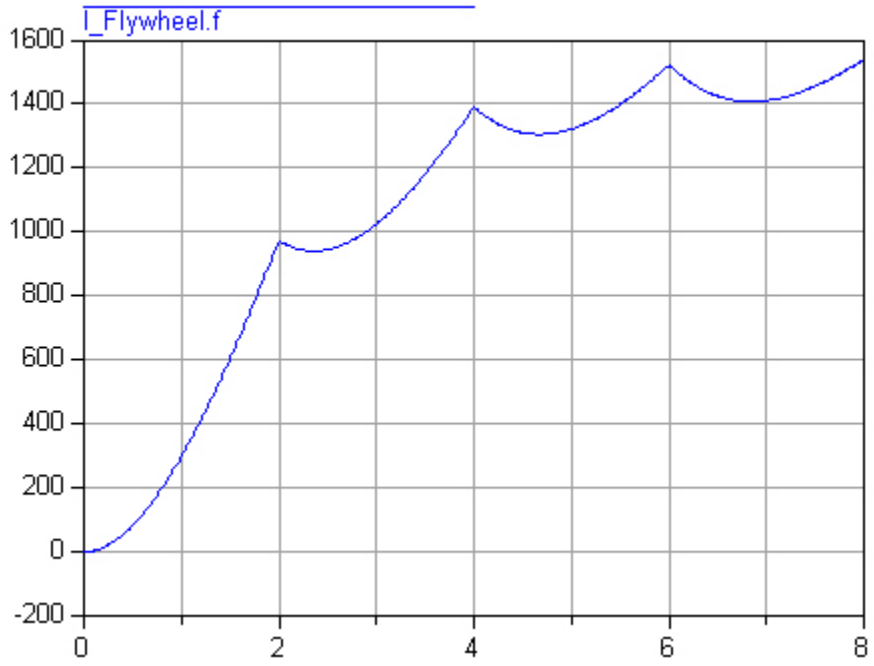


Fig. 4.12: Angular velocity of engine assembly vs. time; with R=1 N-m.s.

In Fig. 4.12 it is visible that by starting the simulation, the angular velocity of the engine assembly increases up to the point where the system experiences a switching event; i.e. the following gear stage becomes active. At this point, contrary to the previous case, the angular velocity does not drop to zero, but continues its trajectory from the same point due to the presence of inertia. However there is a local decline exactly after each switching occurrence. The leading gear in the subsequent power line has a different angular velocity (zero in this case) and therefore a different angular momentum. These local declines indicate the *slipping* phase, where the two different angular velocities of the engine assembly and the leading gear of the subsequent power line are being synchronized.

It is interesting to note that by increasing the resistive value, the graph in Fig. 4.12

and the saw tooth graph in Fig. 4.9 start converging towards each other. For instance, by increasing R to 50 N-m.s, the flow for the engine effort source and the flow (angular velocity) for the inductor (flywheel) show a lot of resemblance, as depicted in Fig. 4.13; with the latter sketched as the dashed graph.

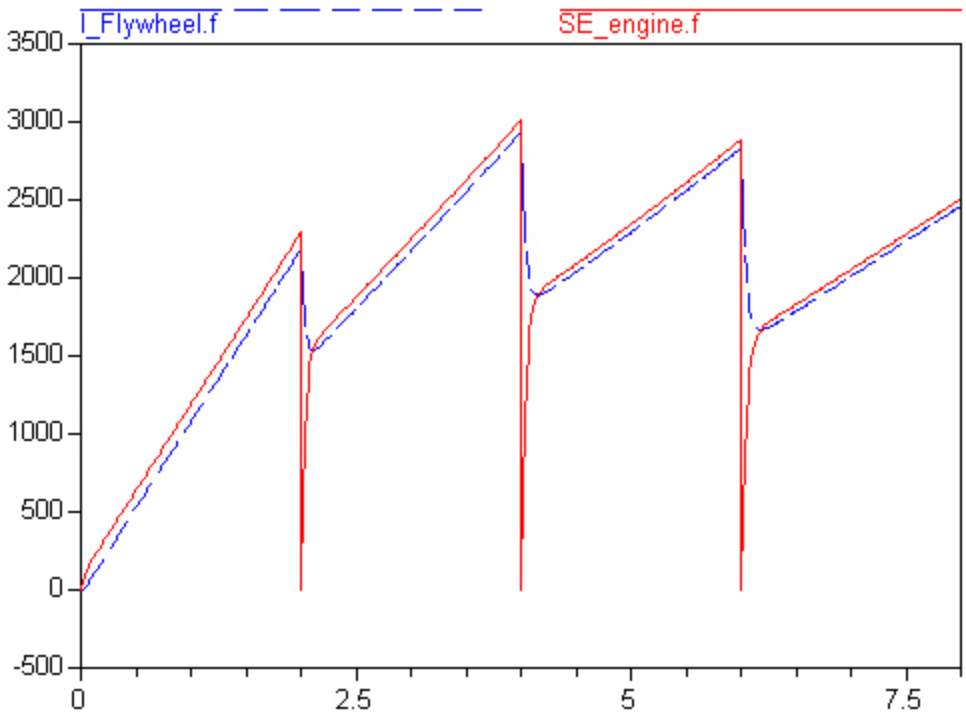


Fig. 4.13: Angular velocity of flywheel (dashed) and flow of effort source vs. time; with R=50 N-m.s.

Greater values for the torsional damping coefficient mean a more "stiff" coupling; a less deformable damper. If the torsional damping coefficient is increased to 5000 N-m.s, the two graphs become almost identical except for the sharp drops in the flow curve for the effort source, which occur at the switching instances. For this new value of R, the flywheel angular velocity is displayed in Fig. 4.14. This graph is comparable to the one presented in [2], where Brück et al. discuss some features of Dymola by modeling a

vehicle engine.

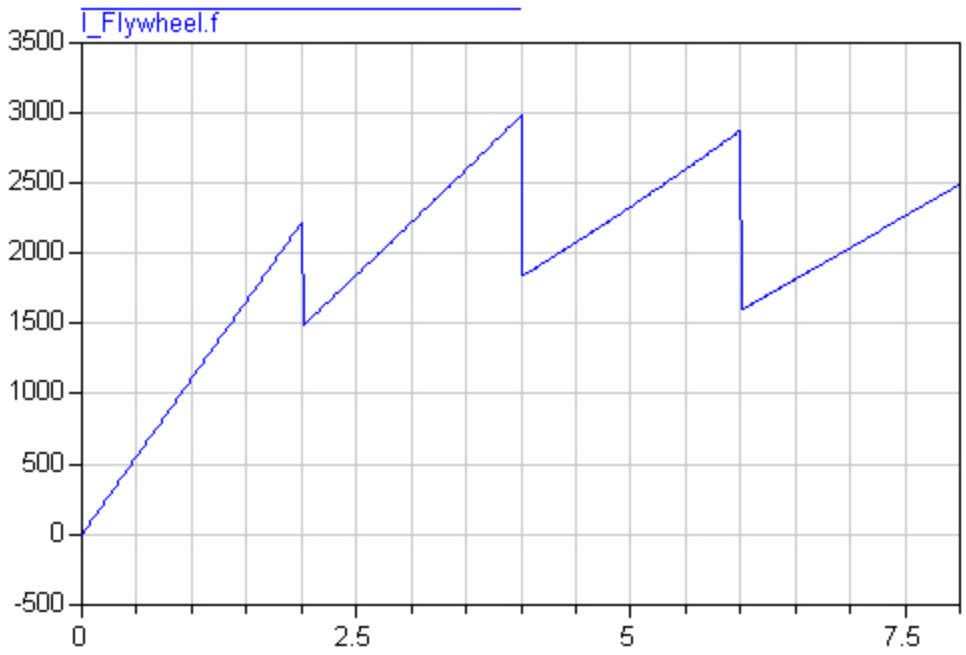


Fig. 4.14: Angular velocity of flywheel vs. time; with R=5000 N-m.s.

Other graphs of interest are the effort (torque) and the flow (angular velocity) of a particular gear in the system. For example, the angular velocity of the leading gear in the second line is demonstrated in Fig. 4.15. The torsional damping coefficient has been set back to 1 N-m.s for more visual clarity.

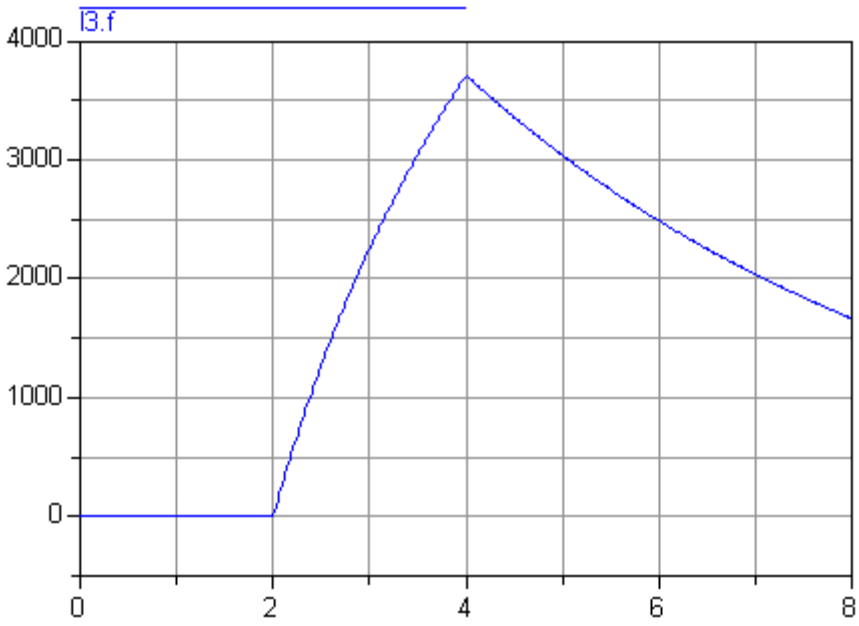


Fig. 4.15: Angular velocity of second leading gear vs. time.

From this graph it can be seen that before the engagement time of the second gear stage, the angular velocity of the gear pair in the second power path is zero. When the system shifts to the second gear at $t = 2$, the angular velocity of the leading gear in the second power path starts to rise and continues its ascend till the next switching event at $t = 4$, where the system shifts to the third gear. At that point, the angular velocity of the leading gear in the second power path halts its increase, and starts a decline on a new trajectory which will eventually reach zero. Figure 4.16 shows the effort (torque) for the same gear.

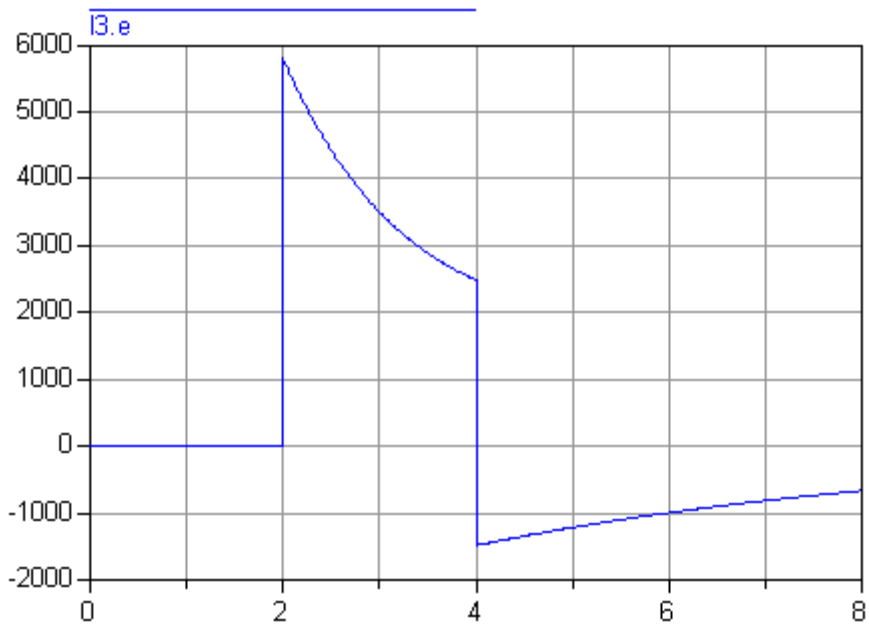


Fig. 4.16: Torque of second leading gear vs. time.

Similar to Fig. 4.15, before shifting into the second gear, the gears in the second power path are inactive, thus maintaining zero torque. At $t = 2$, this gear pair is suddenly subjected to the engine power and the load simultaneously, therefore we witness the abrupt rise in Fig. 4.16 at this point. As the angular velocity of the leading gear in the second power path increases after engagement (as shown in Fig. 4.15), the torque on this gear decreases in addition to the fact that the sinusoidal load is also declining. At $t = 4$, the second power path is disengaged and both the engine torque and the load are removed simultaneously, however this gear is still subjected to the angular momentum of its conjugate, hence the torque on this gear suddenly drops to not even zero but a negative value. As time proceeds, the angular velocities of both gears diminish due to friction, resulting in a decreasing angular momentum for the conjugate gear which in turn leads to less torque on the leading gear, which will eventually reach zero.

4.3. *PILOT EJECTION SIMULATION*

4.3-1. *Things to Be Careful About*

The main difference between this case and the gearbox model is that although similar to the gearbox, this model is also composed of separate power paths; however we must keep in mind that only one of the power lines is physically meaningful at a time, whereas in the gearbox model, all of the power lines coexist in the real world (except for the load element). This fact results in an important concern: synchronizing the initial conditions of all the elements in the currently active power path with the final values of the corresponding elements from the previously active power line. In the gearbox model, the issue of resetting the initial conditions was merely confined to the load elements; and even in that case, they needed only to be set back to zero. The problem here is feeding back the final values of evolving state trajectories to the newly engaged line at each switching occurrence.

4.3-2. *Position Sensors*

We know that Bond graphs directly deal with efforts and flows. However it is a very common practice that the modeler should be interested in the integral of the flow variable; which in our case is the “position”. For this purpose, sensor elements have been implemented in the Bond graph library. We shall concern ourselves with two types of sensors: the “e-sensor”, and the “q-sensor”.

An e-sensor is essentially a flow source with zero flow. Connected to a type 0 junction, an e-sensor will read out the effort value at that junction without having any

effect on the flow variables, since we know that all efforts around a type 0 junction have the same value. Similarly, a q-sensor can be viewed as an effort source with zero effort. By connecting it to a type 1 junction, we can read out the flow value of that junction which is equal for all of the surrounding elements, without affecting the effort variables. These two sensors are depicted in Fig. 4.17.

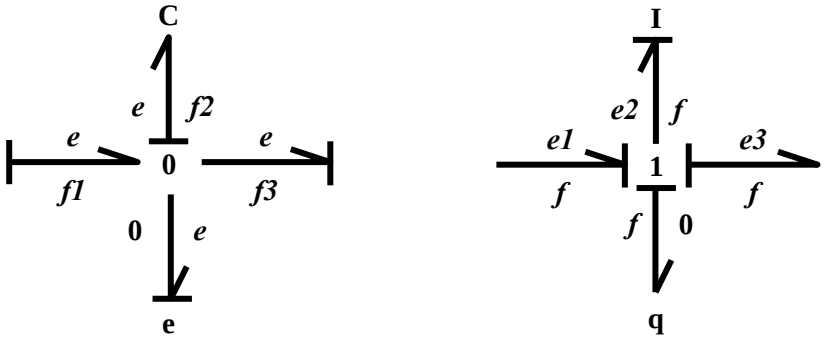


Fig. 4.17: e and q sensors.

This figure shows an e-sensor connected to a type 0 junction, along with a capacitor denoting a spring. In this case, the effort would be the force in the spring; $k \cdot \Delta x$, which is the value read by the e-sensor. Hence we can easily compute the positional difference between the two ends of the spring by simply dividing the output value of the sensor by *k*. In the right graph of Fig. 4.17, a q-sensor is attached to a type 1 junction which on the other side is connected to an inductance representing a mass, therefore the flow variable for this junction would be the velocity of the mass, *v*. The q-sensor directly reads out the position of the mass by integrating the flow variable once.

4.3-3. *Simulation and Results*

We now attempt to complete the pilot ejection model from the previous chapter and

launch it for simulation. As mentioned earlier, since each branch of the model is meant to dynamically follow its forerunner, at each switching occurrence we require that the dynamic properties of all elements should be set equal to the final value of the corresponding elements in the preceding power line. To facilitate this task, we implement a q-sensor for each available mass in the model for both directions of motion in order to fully track the two dimensional trajectory of each mass. Figure 4.18 shows the first phase of the pilot ejection model with the q-sensors implemented. Each inductance has been replaced by an additional type 1 junction with two bonds streaming out; one bond connects to the original inductance, and another bond connects to the q-sensor. The models of the other three stages are modified in the same way.

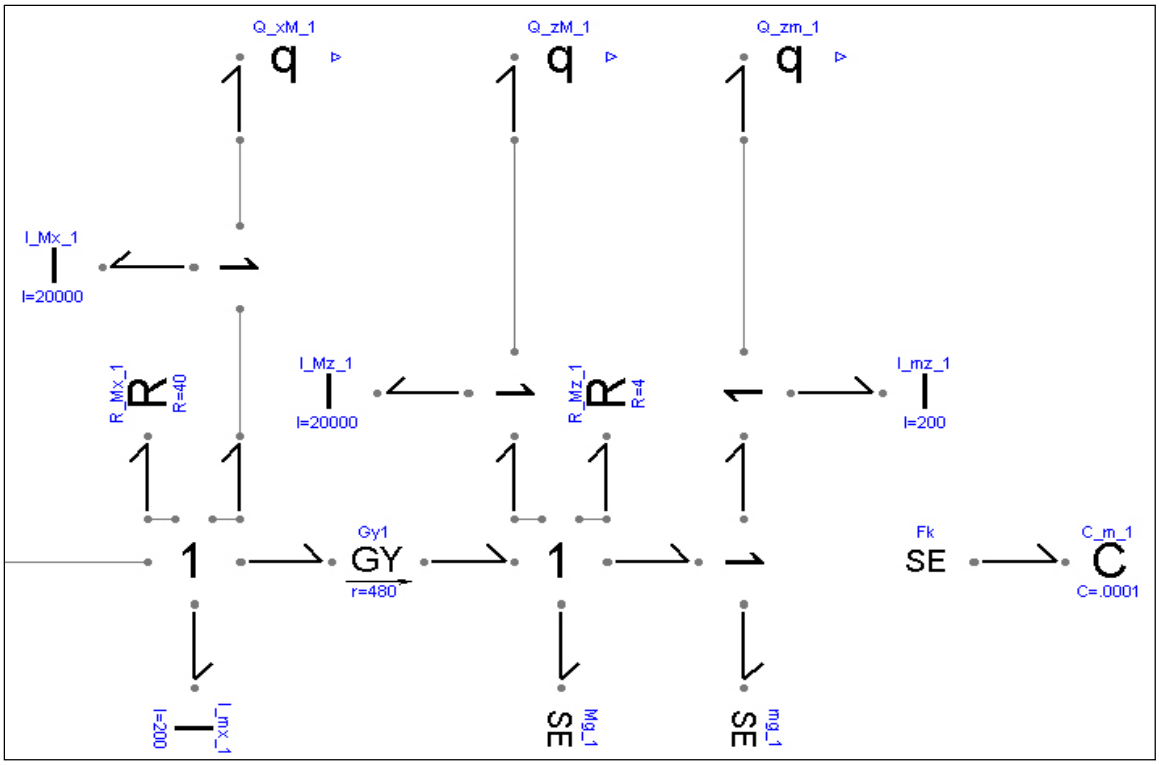


Fig. 4.18: Pilot ejection first phase with sensors.

In addition to the q-sensors, the model for the second phase in Fig. 4.19 contains an e-sensor linked to the type 0 junction where the capacitor (spring) is located. Since the effort associated with this junction is $k \cdot (z_M - z_m)$, the output of this sensor delivers the elevation difference of the aircraft and seat, multiplied by the spring constant. A gain of $1/k$ needs to be installed after the sensor to hand in the precise value for $(z_M - z_m)$.

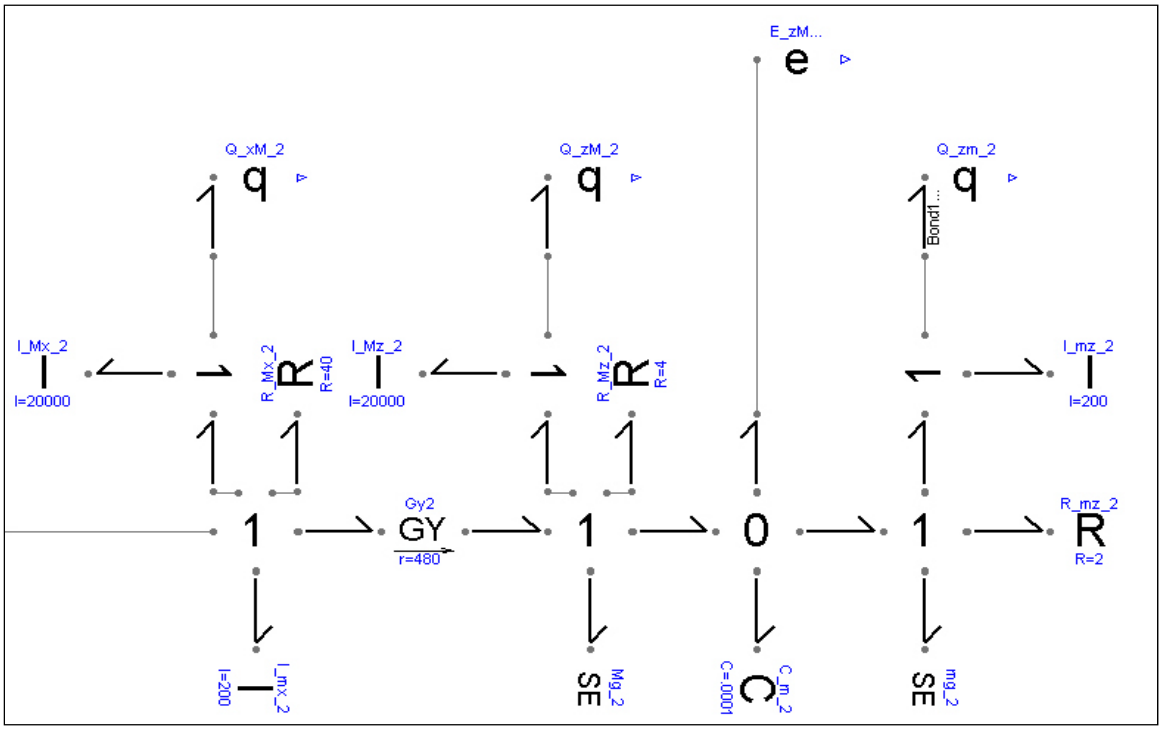


Fig. 4.19: Pilot ejection second phase with sensors.

Figures 4.20 and 4.21 depict the necessary modifications for the third and the fourth phase, respectively.

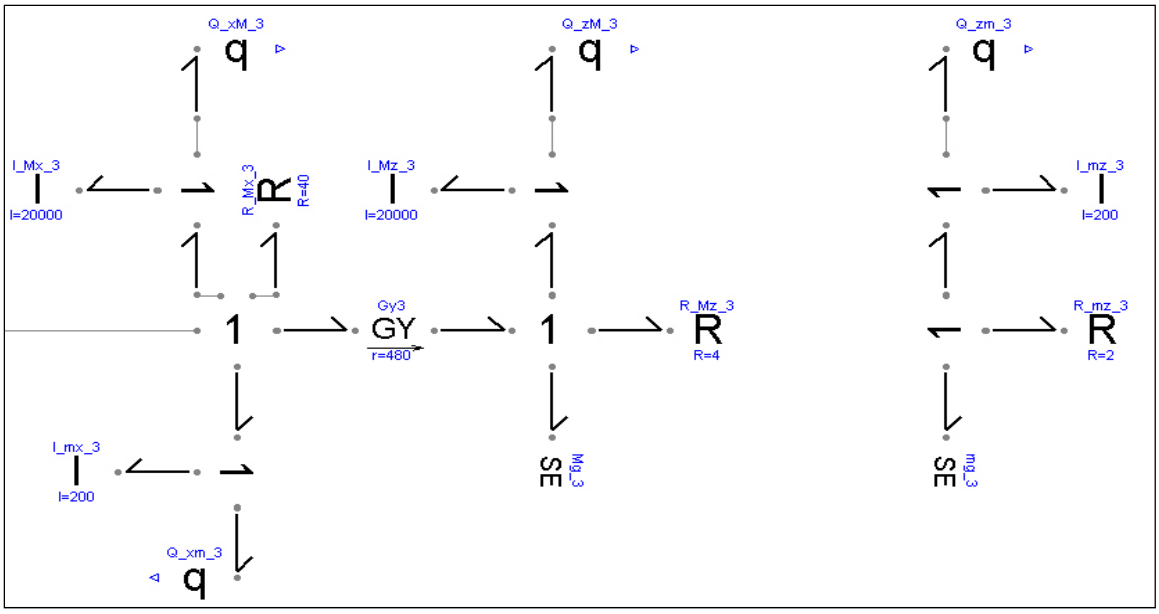


Fig. 4.20: Pilot ejection third phase with sensors.

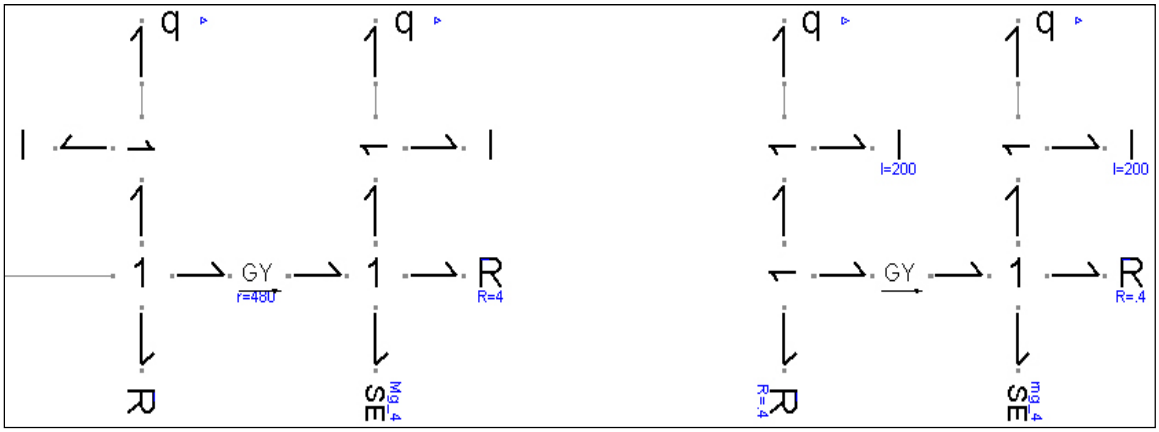


Fig. 4.21: Pilot ejection fourth phase with sensors.

Figure 4.22 shows the overall layout of the complete pilot ejection model in a rather compact form, which is a 90º CW rotation of Fig. 3.27 from chapter 3.

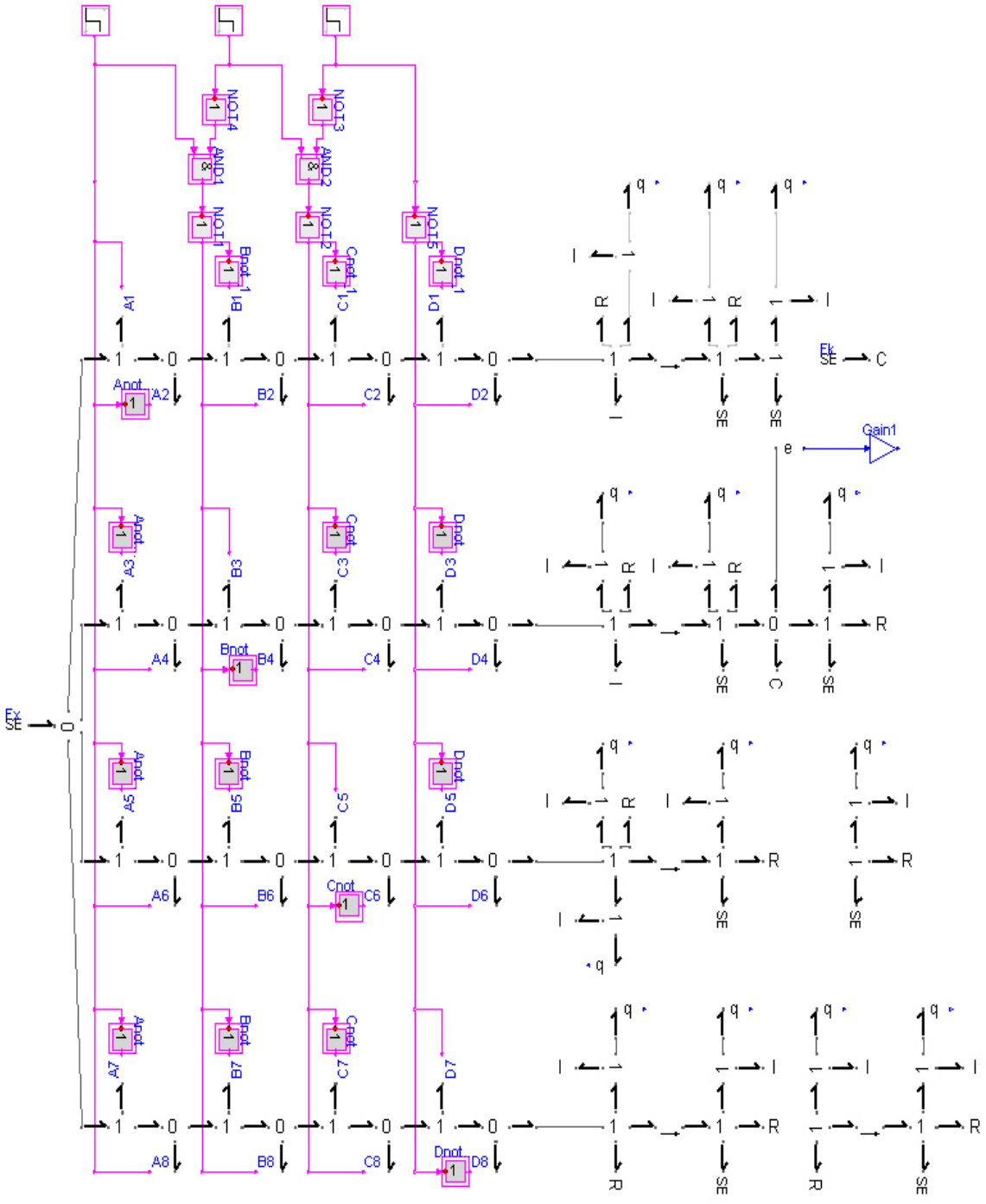


Fig. 4.22: Complete pilot ejection model.

For simulating the model, the engine thrust is set to be 100,000 N. The airplane mass

and seat mass are 20,000 kg and 200 kg respectively, and the spring constant is 10,000 N/m. The damping coefficients representing the air resistance for the plane are 40 and 4 kg/s in the x and z direction respectively, and 0.4 kg/s for the seat in both directions during free motion. However for the period of time when the seat travels along the rail, the resistive coefficient is assigned a value of 2 kg/s in the z direction. Without the loss of generality we could initiate the simulation with $z_M = 0.5$ m for the aircraft and $z_m = 0$ m for the seat; the difference being equal to the span which the seat and spring maintain contact. The overall structure of the program code has the following form:

```
model modelname
      parameter
      .
      .
      .

      initial equation
            Initial equations
      equation
            when conditions
                  equations
            end when
            equations
            .
            .
            .
end modelname
```

The entire program code is placed between a “model-end” clause. Coding starts with

defining the constant parameters, followed by all the elements' specifications and their relation. The latter part, however, was automatically completed by the compiler when the modeler finished constructing the model simply using the graphic-user interface. In fact, while displaying the source code, Dymola leaves this part out by just showing a tiny symbol on the screen to avoid unnecessary expansion. The modeler can easily access the code to this section by clicking on the mentioned symbol and expanding it.

Two initial conditions are declared at the beginning of the code: first, the initial value for the aircraft velocity; set to be 420 m/s in the *x* direction. There is no need to define this as an initial condition for the seat too, since it assumes the same initial value as the airplane. The second initial condition is the position of the aircraft's center of mass. We set this value to be 0.5 m by assigning it to the output of the q-sensor connected to the inductor representing the aircraft's mass in the *z* direction within the first power path.

There are three switching occurrences, with the first being arbitrarily chosen at $t = 1$; firing the ejection cycle. For each of these switching occurrences there is a "when" clause in the program code that becomes "true" accordingly; enclosing a series of "reinit" commands for re-initializing the state variables of the next power path elements to be consistent with the final values of the matching elements from the former power line. These state variables are: i) "flows" of the inductances that represent the masses of the aircraft and seat; and ii) the output of the q-sensors that measure the vertical and horizontal positions of these two masses. Also in the first "when" clause (meant for the second stage), the "effort" of the capacitor (spring) is a state variable and therefore included in the re-initialization sequence.

The simulation was executed for 3 seconds. Switching occurs at $t = 1$ (arbitrarily chosen; starting the second stage), $t = 1.064$ (contact with spring ends; starting the third stage), and $t = 1.17$ (travel along the rail ends; starting the fourth stage i.e. free motion).With the commencement of the simulation, all of the power paths start running, although as mentioned before, only one path at a time has physical meaning. However due to the re-initialization plan in the program code, when the right time comes for each power line to be active, its state variables are appropriately synchronized, by capturing the trajectory of their corresponding variable in the previous line, and continuing to evolve from there on. Figure 4.23 shows the *z* coordinate evolution in time for both the aircraft (dashed curve) and the seat during the first stage; i.e. before triggering the ejection cycle. Therefore the trajectories are only valid up to the first second.

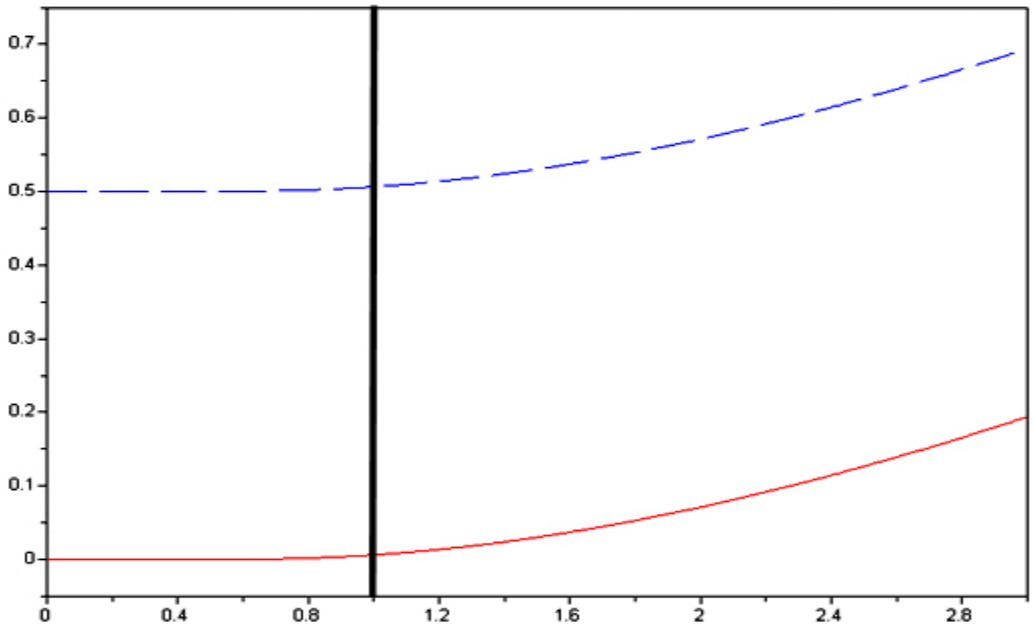


Fig. 4.23: *z* coordinates for aircraft (dashed) and seat vs. time; first phase.

As it can be seen, the elevation difference remains preserved had the ejection cycle

not been fired, according to the curves beyond the first second.

Figure 4.24 depicts the z coordinate vs. time for both the aircraft and the seat during the second stage; where the ejection system has been triggered. Only the trajectories between $t = 1$ and $t = 1.064$ are valid in our simulation. During this time span, it is visible from the figure that the seat starts its upward motion until its reaches the z coordinate of the aircraft's center of mass.

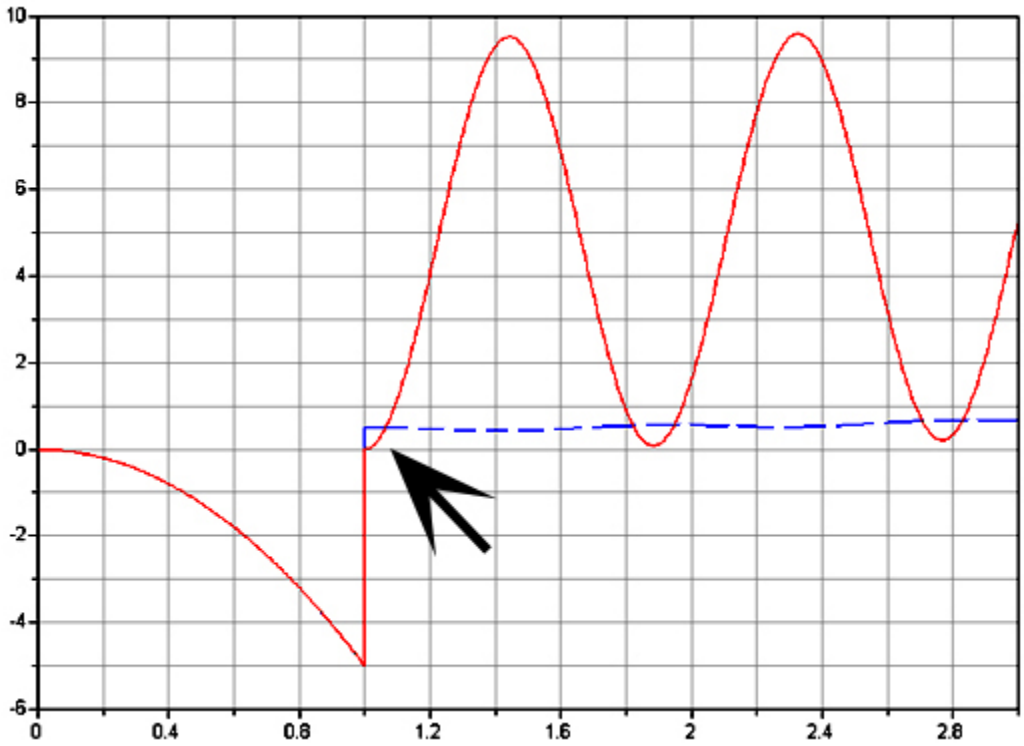


Fig. 4.24: z coordinates for aircraft (dashed) and seat vs. time; second phase.

These curves are generated by the second power path. Since all power lines, including the second, are active throughout the simulation, therefore the unnecessary sections must be discarded. However, it is visible in this figure that how the curves have abruptly adjusted themselves to the correct trajectory at $t = 1$. The oscillatory curve with greater

magnitude belongs to the seat due to the continuation of the spring's effect. This graph also shows the minor fluctuations in the aircraft altitude in the dashed curve, for the same reason. The fact that the valid time span for the second phase is so short is quite natural, considering the functionality of the system, i.e. launching the pilot seat as fast as possible.

Figure 4.25 demonstrates the same values for the third phase, where the contact between the spring and seat has ended, yet the seat is still traveling along the guiding rail. The only forces acting on the seat are the rail friction force, and gravity. The valid segments of both curves are in the time range of $1.064 < t < 1.17$.

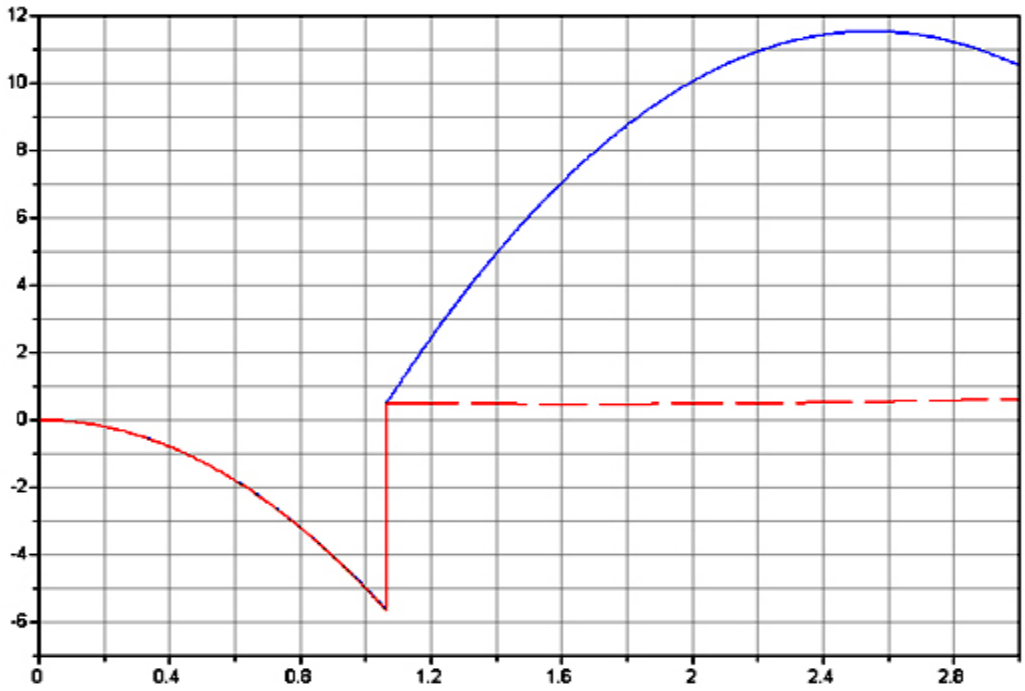


Fig. 4.25: *z* coordinates for aircraft (dashed) and seat vs. time; third phase.

Finally, the seat leaves the plane and continues its motion freely, exposed to air friction and gravity. The curves for this stage are demonstrated in Fig. 4.26.

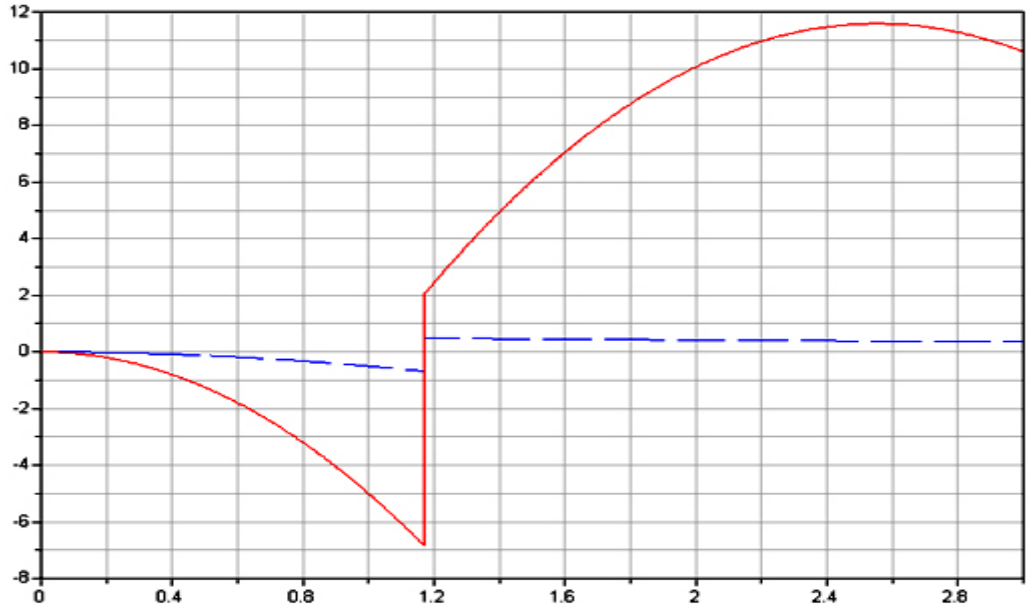


Fig. 4.26: *z* coordinates for aircraft (dashed) and seat vs. time; fourth phase.

Hence, combining all the graphs for the seat in the *z* direction of motion and eliminating the irrelevant segments of the curves, hands in the curve shown in Fig. 4.27:

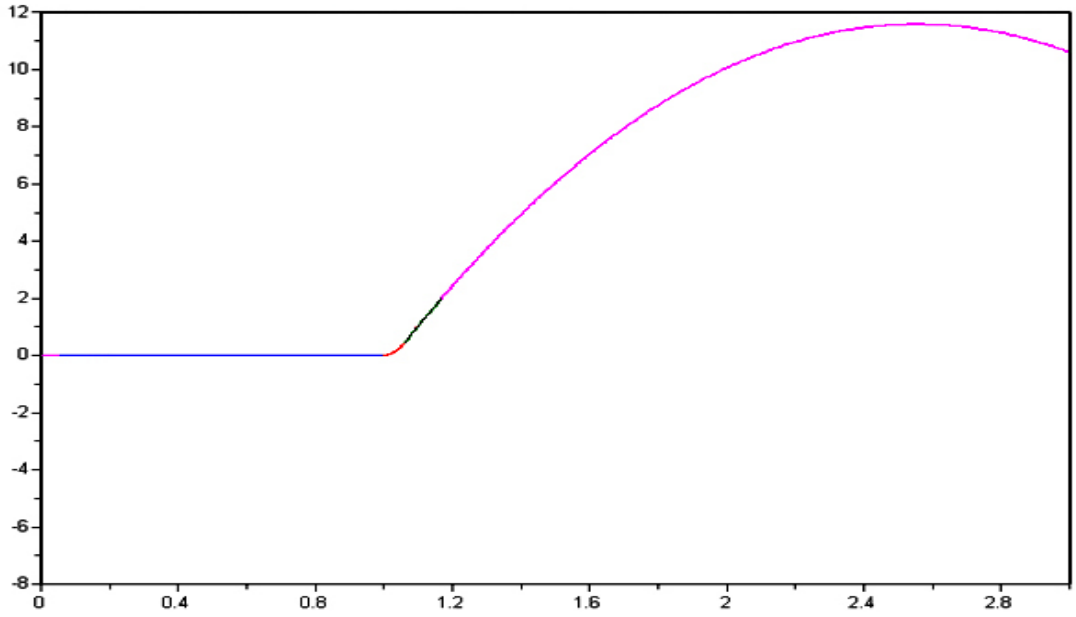


Fig. 4.27: *z* coordinate of seat vs. time; entire trajectory.

As mentioned before, Dymola offers the possibility to generate plots for all the flow and effort variables of a model. Presented here was the relative altitude of the pilot seat plotted versus time to demonstrate the functionality of our model. Additional information of the system's dynamics can be obtained by producing further plots; which is left out here for brevity. However, the results of this simulation along with the results from simulating the gearbox model demonstrate the fact that proposed method for modeling switching networks indeed provides a systematic approach to solving the long-lingered problem of describing discontinuous models using Bond graphs.

5. *SUMMARY, CONCLUSION, FUTURE WORK*

This thesis presented a novel approach to developing a switching algorithm for Bond graph modeling. A four-stage gearbox was taken as a mechanical switching network and after a general introduction was translated into a swtiching electrical circuit diagram by means of digital logic techniques, and in the next step was mapped into a Bond graph metaphor. After the first modeling attempt it was observed that the causality rule for the type 1 junctions was violated; propagating through the entire network with the consequence of infecting the causality strokes at the type 0 junctions too. Results beautifully match the real world phenomena: abrupt switching from one state to any other arbitrary state is not possible. A transition phase always comes into play, in order to synchronize between the previous and future effort(s)/flow(s), which has to be taken into consideration even when modeling mechanical systems.

After a closer examination of switching properties in electrical circuits, it was then concluded that either the switches have to be defined in a non-ideal fashion, or additional switches are needed in order to solve issues such as dumping the residual voltage from the intermediate sections. Therefore by introducing auxiliary switches into the developed Bond graph, the floating efforts (voltages) were allowed to escape; the causality rules for the junctions were amended; and a conflict-free Bond graph representation was generated, incorporating only ideal switches. The algorithm developed was thereafter applied to another mechanical example; the pilot ejection problem. Further conclusions were made regarding modeling mechanical systems and their analogy to their dual electric representation. After verifying the versatility of the proposed algorithm with this

problem too, we concluded that this method could be generalized to other problems of interest with the same nature.

The advantage of this method becomes more evident when dealing with complex systems incorporating several switching mechanisms, each acting as a single component of the entire system. For example consider a vast production line containing numerous subsystems, including an *n*-stage gearbox in one branch and an electrical switching board with *m* positions implemented elsewhere. In the case of "mutual exclusivity" (which will result in the maximum number of switches), now we only need $2 \times n^2$ *switches* for the gearbox and $2 \times m^2$ *switches* for the electrical board; with the obvious benefit of all being interconnected in a single integrated model. Even if a switching subsystem needs to be "isolated" (chapter four) from the effect of the subsequent elements in the line, an additional $2 \times$*(number of stages)* more switches will be needed for that element, which is still much better than the previous approach in the field; i.e. constructing $n \times m$ separate *models* for all different possible switch configurations of the *entire system*. As the number of switching subsystems increases, we could even reach a point where the number of separate models will exceed the number of switches proposed in this text.

The algorithm developed in this thesis indeed hands in a systematic way for presenting complicated switching systems via Bond graphs; however further domains for exploration could be looking into a minimizing (optimization) algorithm for the number of switches. Also of interest could be investigating the necessity/usefulness of introducing a new Bond graph element in order to handle multiple switching. Synchronization of the state variables of identical elements that appear more than once in

the model was dealt with manually by inspection; however the room for research is open to form a method for automatically handling this task, which is left out from this work due to our focus on solving the main problem.

The voltage/current metaphor is indeed very useful in designing and analyzing systems with energetic components since it demonstrates the energy (power) flow in the interconnected elements of a system in a topological fashion, though under the umbrella of the present configuration it seems hard or even impossible to implement Bond graphs in domains were the cause-effect relations between the building blocks of the system are not clear and mixed with some degree of vagueness. This issue restricts the application of Bond graphs to physical systems, and even only to those for which the governing principles are fully understood. Hence, switching mechanisms in other types of systems will have to be dealt with differently.

REFERENCES

[1] BOSCH GmbH., R.: ***Automotive Handbook***, VDI Verlag, 2nd ed.,1986.

[2] BRÜCK, D.; ELMQVIST, H.; OLSSON, H.; MATTSSON, S.E.: ***Dymola for Multi-Engineering Modeling and Simulation***, 2nd Intl. Modelica Conf. Proc., Oberpfaffenhofen, Germany, Mar.2002, pp.55_1-55_8.

[3] CELLIER, F. E.: ***Continuous System Modeling***, Springer-Verlag, 1991.

[4] CELLIER, F. E.: ***ECE449/549 Lecture Notes***, University of Arizona, Tucson AZ, U.S.A., 2003.

[5] CELLIER, F.E.; McBRIDE, R.T.: ***Object-oriented Modeling of Complex Physical Systems Using the Dymola Bond Graph Library***, Proc. ICBGM'03, Intl. Conf. Bond Graph Modeling and Simulation, Orlando, FL, 2003, pp.157-162.

[6] CELLIER, F. E.; OTTER, M.; ELMQVIST, H.: ***Bond Graph Modeling of Variable Structure Systems***, Proc. ICBGM' 95, 2nd SCS Intl. Conf. on Bond Graph Modeling and Simulation, Las Vegas, NV, pp.49-55.

[7] COUDERT, N.; DAUPHIN-TANGUY, G.; RAULT, A.: ***Mechatronic Design of an Automatic Gear Box using Bond Graphs***, IEEE Systems, Man and Cybernetics, Conf. Proc. 'Systems Engineering in the Service of Humans', Oct.1993, pp.216 -221, vol.2.

[8] ***Dymola5 User Manual***, © 1992-2002 by Dynasim AB., Research Park Ideon, SE-223 70 Lund, Sweden.

[9] EDSTRÖM, K.; STRÖMBERG, J.; TOP, J.: ***Aspects on Simulation of Switched Bond Graphs***, Proc. 35th Conf. on Decision and Control, Kobe, Japan, 1996.

[10] ELMQVIST, H.: ***A Structured Model Language for Large Continuous Systems***, PhD Dissertation, Dept. of Automatic Control, Lund Institute of Technology, Lund, Sweden, 1978.

[11] ELMQVIST, H.; CELLIER, F. E.; OTTER, M.: ***Object Oriented Modeling of Hybrid Systems***, Proc. ESS' 93, SCS European Simulation Symposium, Delft, Netherlands, 1993.

[12] ELMQVIST, H.; CELLIER, F. E.; OTTER, M.: ***Object Oriented Modeling of Power Electronic Circuits using Dymola***, Proc. CISS' 94, First Joint Conference of International Simulation Societies, Zurich, Switzerland, 1994, pp. 156-161.

[13] GARCIA, J.; DAUPHIN-TANGUY, G.; ROMBAUT, C.: ***Electrothermal Bond Graph Model for Semiconductor Switching Devices***, Applied Power Electronics Conference and Exposition, APEC ' 96. Conf. Proc., 11th Annual, Mar.1996, pp.258-263, vol.1.

[14] GILLESPIE, T.: ***Fundamentals of Vehicle Dynamics***, SAE Publications, 1992.

[15] GLASER, J.S.; CELLIER F.E.; WITULSKI, A.F.: ***Object Oriented Switching Power Converter Modeling using Dymola with Event-handling***, Proc. OOS' 95, SCS Object-Oriented Simulation Conference, Las Vegas, NV, 1995, pp.141-146.

[16] KATZ, R. H.: ***Contemporary Logic Design***, Benjamin/Cummings, 1994.

[17] KIM, J.; DAN CHO, D.: ***An Automatic Transmission Model for Vehicle Control***, ITSC 97 IEEE Conf. on Intelligent Transportation System, 9-12 Nov.1997, pp.759 -764.

[18] KREBS, M.: ***Modeling of Conditional Index Changes***, Masters Thesis, ECE Dept., University of Arizona, Tucson AZ, U.S.A., 1997.

[19] LALA, P. K.: ***Practical Digital Logic Design and Testing***, Prentice-Hall, 1996.

[20] LUMKES, J. JR.; FRONCZAK, F. J.: ***Design, Simulation and Validation of a Bond Graph Model and Controller to Switch between Pump and Motor Operation using Four On/Off Valves With a Hydraulic Axial Piston Pump/Motor***, American Control Conf., 28-30 June 2000, pp.3605 -3609 vol.5.

[21] NORTON, R. L.: ***Design of Machinery***, McGraw-Hill, 1992.

[22] OTTER, M.; ELMQVIST, H.; MATTESON, S. E.: ***Hybrid Modeling in Modelica based on the Synchronous Data Flow Principle***, CACSD'99, U.S.A, 1999.

[23] PASTRAVANU, O. C.: ***Switched Bond Graphs in Computer-Aided Analysis of Hybrid Dynamical Systems***, Proc. of the 1998th IEEE International Conf. on Control Applications, Trieste, Italy, 1998.

[24] PAYNTER, H. M.: ***Analysis and Design of Engineering Systems***, MIT press, 1961.

[25] SCHWEISGUTH, M.: ***Semiconductor Modeling with Bond Graphs***, Masters Thesis, ECE Dept., University of Arizona, Tucson AZ, U.S.A., 1997.

[26] SHIM, T.: ***Introduction to Physical System Modeling Using Bond Graphs***, Lecture Notes, University of Michigan, Ann Arbor MI, U.S.A.

[27] ZEIGLER, B. P.; PRAEHOFER, H.; KIM, T. K.: ***Theory of Modeling and Simulation***, Academic Press, 2000.

Further References

[28] http://www.dynasim.com/
Dynasim AB, Research Park Ideon, SE-223 70 Lund, Sweden.