Modeling of Chemical Reaction Networks Using Bond Graphs

by
cand. verf. Michael Amrhein

A Report submitted to the

Electrical and Computer Engineering Department

University of Arizona,
Tucson

In partial Fulfillment of the Requirements
for the Degree

Diplomingenieur

Fachrichtung Verfahrenstechnik
at the
Universität Stuttgart

Modellierung von chemischen Reaktionssystemen mit Hilfe von Bond–Graphen

Modeling of Chemical Reaction Networks Using Bond Graphs

Michael Amrhein (cand. verf.)

Studienarbeit Regelungstechnik

Prof. Dr.-Ing. M. Zeitz

Betreuer: Prof. Dr.-Ing. F.E. Cellier

University of Arizona—Tucson, USA

Institut für Systemdynamik und Regelungstechnik

Prof. Dr.-Ing. E.D. Gilles

Universität Stuttgart

1990

Kurzfassung

Diese Arbeit entstand 1989/90 im Rahmen eines Integrierten Auslandsstudiums zwischen dem Institut für Energiewirtschaft und Rationelle Energieanwendung der Universität Stuttgart und dem Department of Nuclear and Energy Engineering der University of Arizona—Tucson, USA. Die Programmleiter waren in Stuttgart Prof. Dr.-Ing. Voß und in Tucson Prof. Dr. Seals. Die ersten neun Monate dieses Austauschprogramms wurden vom Deutschen Akademischen Austauschdienst (DAAD) mit einem Stipendium gefördert. Die vorliegende Arbeit wurde von Prof. Dr.-Ing. F.E. Cellier, Associate Professor am Department of Electrical and Computer Engineering, betreut. Prof. Celliers Forschungsschwerpunkt liegt seit einigen Jahren in der Entwicklung neuer Verfahren zur Modellbildung und Simulation, sowie im Softwaredesign in der Simulationstechnik, der computergestützten Modellierung und dem CAD.

Die Idee zum Thema dieser Arbeit *"Modellbildung von Chemischen Reaktionsnetzwerken mit Hilfe von Bond Graphen"* muß man zusammen mit dem Ziel sehen, ein allgemeingültiges Verfahren zur dynamischen Beschreibung gemischter Energiesysteme zu entwickeln. Dabei hatte Prof. Cellier schon früh erkannt, daß

das graphische Konzept der Bond–Graphen sich als vorteilhaft und leistungsstark erweist, komplexe und verkoppelte Systeme verschiedener physikalischer Disziplinen modul–hierarchisch zu modellieren. Einige Ergebnisse dieser Arbeit verwendet Prof. Cellier im neunten Kapitel seines ersten von zwei Büchern, das verschiedene Verfahren der Modellbildung umfassend erklären wird. Der erste Band erscheint im Winter 1990 im Springer–Verlag, New York.

Wie gelangt man vom System chemischer Reaktionen zu deren Simulation? Man stellt zuerst Modellgleichungen eines Systems auf, die man mit Hilfe der Methode der Bond–Graphen in einem Bond–Graph–Modell beschreibt; dieses graphische Modell wird in einer Modellierungssprache, in dieser Arbeit der Modellierungssprache DYMOLA, codiert, welche dann die Zustandsgleichungen des Modells generiert. DYMOLA kompiliert anschließend das DYMOLA–Programm in den Code der Simulationssprache DESIRE. Damit ist es dann möglich, das Modell mit einem Rechner zu simulieren.

Zuerst werden in der vorliegenden Arbeit die Methode der Bond–Graphen und die Modellierungssprache DYMOLA allgemein eingeführt und anhand eines Beispiels eines elektrischen Stromkreises veranschaulicht. Anschließend folgt die Anwendung der Methode der Bond Graphen auf chemische Reaktionen unter der Annahme von geschlossenem System und idealer Gase.

Das Konzept der Bond Graphen basiert auf der Betrachtung von Energieflüssen und dem Aufstellen von Leistungsbilanzen. So kann jedes physikalische System durch Bond Graphen durch Leistungsbetrachtung analog beschrieben und verschiedene physikalische Systeme durch das Aufstellen von Leistungsbilanzen verkoppelt werden. Zum Beispiel stellt ein Bleiakkumulator im Auto einen elektro–chemischen Koppler dar. Bond–Graph–Modelle sind modular strukturiert und bewahren gleichzeitig die topologische wie auch die rechentechnische Stuktur. Strukturelle Singularitäten und algebraische Schleifen können im Bond Graph Modell aufgedeckt werden.

Die Modellierungssprache DYMOLA hat sich unter verschiedenen anderen Sprachen wie ENPORT–7, THTSIM, CAMAS und CAMP als die leistungsstarke Sprache erwiesen, Bond–Graph–Modelle in einen Computercode zu übertragen. DYMOLA besitzt selbst keinen eigenen Simulationskern sondern stellt einen Preprozessor dar, der hierarchisch modular strukturierte DYMOLA–Codes in Codes von flachen Simulationssprachen wie DESIRE, SIMNON oder FORTRAN umwandelt.

Zur dynamischen Beschreibung von chemischen Reaktionen idealer Gase im geschlossenen System benötigt man vier Bestimmungsgleichungen neben zwei thermodynamischen Zwängen (z.B. konstante Temperatur und konstanter Druck) zum Bestimmen der sechs Unbekannten in der Gibbs'schen Fundamentalgleichung, die als Basisgleichung des Modells dient. Die vier Gleichungen sind: Die Molenbilanz, die man aus der chemischen Reaktionskinetik erhält, eine Zustandsgleichung — in dieser Arbeit wird die ideale Gasgleichung verwendet — und zwei Leistungsbilanzen, die sich beide aus der Gibbs'schen Fundamentalgleichung herleiten lassen und von denen sich eine als Gibbs–Duhem–Gleichung herausstellt.

Zur Darstellung der Modellgleichungen in Bond–Graphen verwendet man sowohl Basiselemente der Bond–Graph–Methode (dies sind drei Verbindungselemente der Bond–Graphen: der Molenbilanz bzw. die Bilanz der chemischen Leistung werden in der sogenannten "0–junction" bzw. "1–junction" aufgestellt und die stöchiometrischen Koeffizienten der Reaktionsgleichungen werden in den "Transformern" berücksicht) und neue Bond–Graph–Elemente, die den verschiedenen chemischen Reaktionen angepaßt sind und die Berechnung der Leistungsbilanzen und der Zustandsgleichung enthalten.

Als Beispiel einer chemischen Reaktion wird die Brom–Wasserstoff–Reaktion behandelt, die nach Herzfeld und Polanyi durch fünf Reaktionsgleichungen beschrieben wird. Diese Reaktion wird als Bond–Graph–Modell für drei verschiedene Fälle dargestellt: den isothermen–isochoren Fall, den isothermen–isobaren Fall und den isobaren–isentropen Fall. Für jedes einzelne Bond–Graph–Modell wird ein DYMOLA–Hauptprogramm erstellt, das die DYMOLA–Module für die Bond–Graph–Elemente wie einzelne Bausteine verknüpft. Anschließend wird das DYMOLA–Programm in ein DESIRE–Programm compiliert und simuliert.

Das Fazit dieser Arbeit ist: Bond–Graphen haben sich als graphisches Verfahren bewährt, das dynamische Verhalten physikalischer Systeme hierarchisch modular zu beschreiben. Die Bond–Graph–Modelltopologie entspricht der Systemtopologie, die rechentechnische Struktur bleibt gewahrt und es werden algebraische Schleifen und strukturelle Singularitäten aufgedeckt. Wie bei jedem graphischen Verfahren wird auch das Bond–Graph–Modell schnell für große Systeme unübersichtlich. Dem kann man aber durch Bildung von neuen modularen Bond–Graph–Elementen entgegenwirken.

Das Modell der betrachteten chemischen Reaktion wurde unter strengen Annahmen aufgestellt: abgeschlossenes System, ideale Gase, dynamische Verwendung der idealen Gasgleichung, die eigentlich nur in der Umgebung des Gleichgewichts gültig ist.

DYMOLA erwies sich leistungsstark, Bond–Graphen–Modelle zu codieren. DYMOLA setzt automatisch die Kausalitäten der Modellgleichungen und ist fähig, nach einer beliebigen Variablen einer Gleichung aufzulösen, so daß Modulduplizierungen vermieden werden. Daneben hat DYMOLA noch programmtechnische Schwächen, deren Beseitigung einigen Aufwand erfordern werden.

DESIRE zeichnet sich durch eine schnelle Kompilierung aus, reagiert aber nicht robust auf Variationen von Simulationsparametern. Prof. Cellier plant, die lang erprobte und bewährte Simulationssprache ACSL DYMOLA nachzuschalten.

Diese Arbeit sollte als Motivation dienen, die Technik der Bond–Graphen weiter zu entwickeln; ein zukünftige Anwendung besteht, zum Beispiel,darin, Apparate der Verfahrenstechnik in einer Bibliothek von Bond Graph Elementen abzulegen, die man beim Anlagendesign nach dem Baukastenprinzip miteinander verschalten müßte.

Abstract

The aim of this research work is to develop a general methodology for the dynamical description of mixed energy systems. This problem will be studied through the analysis of chemical reaction kinetics and chemical thermodynamics. The Bond graph concepts will prove to be very convenient and powerful to describe complex chemical reactions. A new software tool "DYMOLA" will be introduced which can be used for computer aided modeling using Bond graphs. The methodology will be substantiated by means of the bromine–hydrogen reaction.

Acknowledgments

The research described in this report has been performed in close co–operation with my advisor Dr. François E. Cellier, Professor of the Department of Electrical and Computer Engineering at the University of Arizona. I wish to express my great appreciation to him for his invaluable help, guidance, support, and encouragement.

The sentences of my project's documentation have been disentangled and transformed into correct English by Glenn Farrenkopf and Dr. François E. Cellier, for whose hard work I express my sincere thanks.

I am deeply endebted to my parents, whose support inspired me to put forth the effort to come to the U.S.A. for absolving a part of my studies.

I also wish to acknowledge my sincere gratitude to many other people for their valued company and moral support.

My stay, and thus, this research has been supported by sponsorship of the German Academic Exchange Service.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

One task of an engineer is to invent new means which are likely to lead to simpler solutions of his problems. To be capable to solve problems, an engineer has to better understand his physical system at first in order to manipulate the systems' behavior thereafter. For this, he will derive a model of a real system based on data extracted from experiments applied to it. This model may contain, of course, many simplifying assumptions of the real system that have to be validated in different experimental simulation runs. Finally, he draws conclusions from the final simulation data that must be further validated again in simulation runs.

This project applies a modeling and validation procedure to chemical reaction systems to provide a deeper understanding of how they macroscopically work.

Traditionally, there are two types of chemists. The "kineticists" consider the change of matter dynamically in terms of differential equations by totally ignoring the flow of energy, whereas the "chemical thermodynamicists" examine chemical reactions near equilibrium rather than looking at the change of matter and at the change of energy.

However, most chemical reactions either absorb heat energy (endothermic) or generate heat energy (exothermic) and with this, the temperature, the pressure and/or the volume of the total chemical system are likely to vary. Thus, it is necessary to examine a reaction, chemical kinetics, thermodynamics, and pneumatic/hydraulic flows in unison, since, further, chemical reactions are studied farther away from their equilibrium than traditional approaches allow . For this study, the concepts of Bond graph modeling have proven to be a very convenient way to gain an improved understanding of what is happening in these highly complex and non–linear systems. The software tool DYMOLA will be introduced to implement these hierarchical Bond graphs.

Never before in the past have Bond graphs been used in detail to describe chemical reaction systems. Bond graphs were invented in 1961 by Henry Paynter[12], a professor at M. I. T. This invention has been used to describe the dynamical behavior of systems from different disciplines of physical sciences in a uniform manner. It has been known for a long time that the power of any physical system can be expressed by two variables: a *through variable* and an *across variable.* These two quantities are carried along the entire Bond graph so that the balance of energy at any point of the Bond graph can be found. This makes it easier to deal with interfaces between different types of physical systems as well as determining the energy within the system. A lead accumulator, for example, — usually used as a starter battery in a car — represents such an interface, a chemical–electrical coupler. The Bond graph methodology provides for a more systematic technique in handling complex systems. Further, Bond graphs preserve both the topological and computational structure of all kinds of physical systems, thus, they constistute a more general graphical representation than, for example, either signal flow graphs or Block diagrams that are commonly applied by Control Engineers. Chapter 2 provides for more information about the Bond graph modeling technique in general and contains a step–by–step description (with figures) which illustrates the concept of using Bond graphs.

As mentioned above, DYMOLA (DYnamic MOdeling LAnguage) is exploited to implement Bond graphs in a computer program by mapping Bond graphs into a state-space description. This application of DYMOLA is new and has exclusively been used by some of Dr. François Cellier's Masters and Ph. D.-students at the University of Arizona. DYMOLA was designed at the Lund Institute of Technology in Sweden in 1979 by Hilding Elmqvist in his Ph. D. dissertation.[18]. Large scale systems, for example, chemical systems, can be considered, that can be handled conveniently by DYMOLA by being modeled in a modularly hierarchical manner. Chapter 3 provides for the general introduction of the properties of DYMOLA — the *cut* concept will be explained in detail, an application of DYMOLA to the example introduced in the second chapter will be shown. In this example, DYMOLA is demonstrated to be a very powerful tool, able to automatically assign causalities of the model's equations, produce a state–space description and, finally, transfer the DYMOLA–program into a simulation language code. The currently available codes are DESIRE [23] and SIMNON [17], two direct executing flat continuous–system simulation languages, that are available on both PC–compatibles, VAX/VMS-systems. Some rules to be followed about the transition of Bond graphs to the DYMOLA description are shown at the end of chapter 3 as well.

Chapter 4 contains the case–study of this project. At the beginning of chapter 4, basic equations of chemical kinetics and thermodynamics will be introduced since they are needed to derive the algebraic and differential equations for modeling chemical systems. The sets of differential and algebraic equations will be evaluated in both existing and newly invented Bond graph elements in order to hierarchically describe the dynamical behavior of chemical reactions. The design of modular Bond graph elements is demonstrated in the Hydrogen–bromine reaction representing a complex chemical reaction. This project introduces a method for modeling chemical reactions, which is applied to the Hydrogen–bromine reaction under certain input constraints. First the isochoric (constant volume) and isothermic (constant temperature) is analyzed, and then the isobaric (constant pressure) and isothermic case is analyzed. For the model validation there will be used another program, named "Computer Program for the Calculation of Complex Chemical Equilibrium Compositions, Rocket Performance, Incident and Reflected Shocks, and Chapman–Jouget Detonations" [59].

Chapter 5 concludes this project by summarizing the results and suggesting future research topics.

Chapter 2

Bond Graphs

2.1 Preview

In this chapter the concepts of Bond graphs are introduced in general. The Bond graph methodology will prove to be a very powerful tool to represent any kind of physical systems.

2.2 Introduction to the Basic Principles of Bond Graphs

For a long time it has been known that the dynamical behavior of systems of different types of physical sciences has mathematical structures in common. Thermodynamic systems, for example, have a fluid mechanical analogon, i.e. the temperature and the heat energy flow can be treated as velocity and shear tension, respectively. Another example is the mechanical system and its electrical system counterpart, since the mass, the spring and the friction represent the inductor,the capacitor and the resistor, respectively.

The need to both formalize the knowledge of a common structure of physical systems and simultaneously provide both their computational and topological structure, led to the invention of Bond graphs. This relatively new and very powerful tool can be applied to most physical systems including chemical reaction networks, electrical networks, mechanical rigid bodies, and thermal and hydraulic/pneumatic transport phenomena. See the reference list for further detailed information about the bibliography of Bond graphs **[1–15]**, especially there is Dr. Cellier's book mentioned **[2]**.

The real strength of the Bond graph is the ease by which different types of physical systems can be interfaced using the transducer, as will be shown later. This is only possible since the power in the sytem is carried along in the Bond graphs. Hence, the balance of power can be made at any point in the graph and thus the connection to another system can be easily made since the law of energy conservation is true for any point inside the graph describing a physical system. It should be noted that using power instead of energy is more convenient since one of the main tasks of Bond graphs is to represent dynamical systems and, thus, the change of energy over time is important to be considered which, of course, is equal to power.

The engineering world always needs graphical methods to quickly illustrate physical problems. A control engineer, for example, traditionally uses the Block diagram whose disadvantages will be described. This shall help to clarify the importance of Bond graphs and how they overcome the Block diagram's disadvantages.

Since the Block diagram preserves the computational structure of a control system, there is no problem to read out its set of differential and algebraic equations directly. But in order to draw the Block diagram for a physical system, it is important to derive its complete set of equations first. There is yet another disadvantage as mentioned above. Even though the Block diagram preserves the

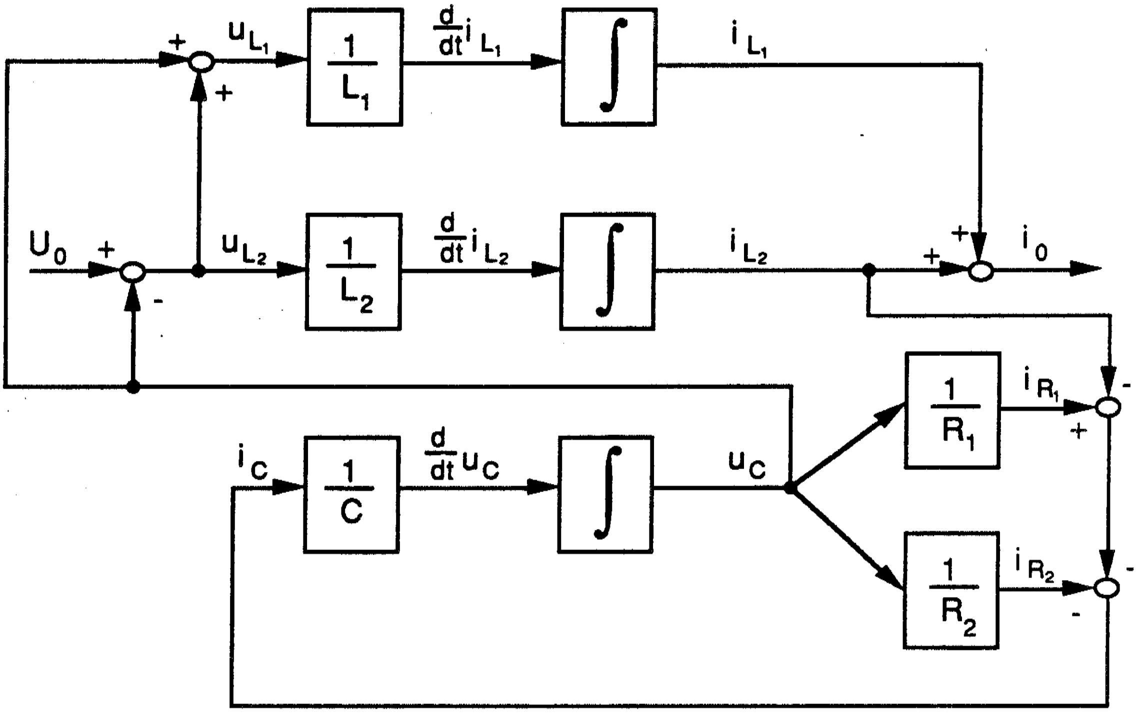


Figure 2.1: Block diagram of a simple RLC circuit

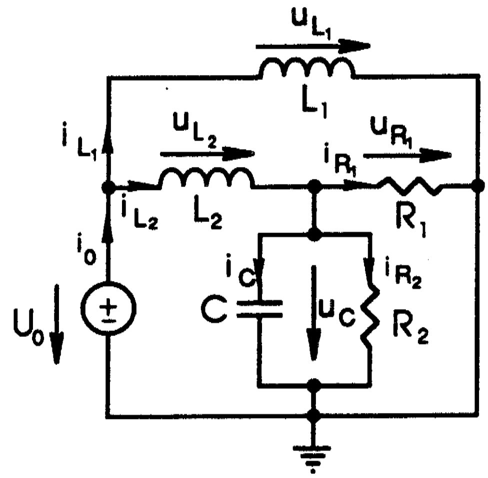


Figure 2.2: RLC circuit of the above Block diagram

computational structure it does not preserve the topological structure. This can be seen from the Block diagram of a simple electrical circuit in Fig.2.1.

It is almost impossible to reconstruct the physical system shown in Fig.2.2 from its Block diagram. This is because the voltages and and the currents associated with the voltages are totally separated in the Block diagram representation. It should be noted that in this chapter an example of an electrical circuit will be used since electrical circuits can be conveniently used as examples when explaining the application of graphical methods.

In the following the algebraic-differential equations of the electrical network are given:

$$\frac{di_{L_1}}{dt} = \frac{u_{L_1}}{L_1} \tag{2.1}$$

$$\frac{di_{L_2}}{dt} = \frac{u_{L_2}}{L_2} \tag{2.2}$$

$$\frac{du_C}{dt} = \frac{i_C}{C} \tag{2.3}$$

$$i_{R_1} = \frac{u_{R_1}}{R_1} \tag{2.4}$$

$$i_{R_2} = \frac{u_C}{R_2} \tag{2.5}$$

$$u_{L_1} = u_{L_2} + u_{R_1} \tag{2.6}$$

$$u_{L_2} = U_0 - u_C \tag{2.7}$$

$$u_{R_1} = u_C \tag{2.8}$$

$$i_0 = i_{L_1} + i_{L_2} \tag{2.9}$$

$$i_C = -i_{R_1} - i_{R_2} + i_{L_2} \tag{2.10}$$

The complex kinetical and thermodynamical properties of chemical reaction networks can be decomposed into smaller modules. If there is furthermore an electrical system connected, as the lead accumulator represents a chemical–electrical linkage as mentioned, the Bond graph will prove to be quite helpful to achieve this coupling.

Modeling physical systems with Bond Graphs is a simplified abstract construction which will be introduced in this chapter. Thereby the Bond Graph modeling will be explained in general, and in the end of this chapter the concept of Bond Graphs will be applied to the previously shown electrical circuit.

2.3 Basic Definitions

This section demonstrates the basic definitions of the Bond graph methodology containing the multiport elements (Sec.2.3.1 and 2.3.4) which build the "network" of the Bond graphs. The through and across variables (Sec.2.3.2) are carried along the Bond graphs, determining the value for the power, displacement,

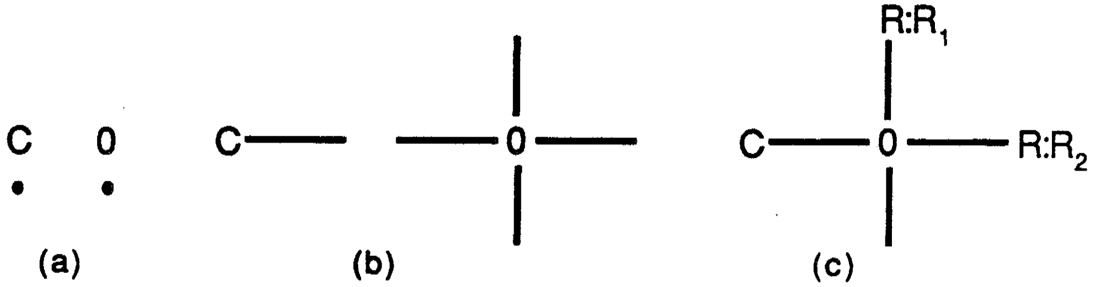


Figure 2.3: (a) Elements, (b) Elements with ports, (c) formation of bonds and momentum (Sec.2. 3. 3).

2.3.1 Multiport Elements

In analyzing systems the first step will be to isolate *components* or *elements* and to define *variables* which will be used to characterize them. In the example in Section 2. 2 the capacitor, inductor, resistor and the voltage source have been the elements while the voltage u and the current i represented the two variables. Such components are realized in the graph's nodes by alpha-numeric letters (for example 0 and C in Fig.2.3a).

The connection of an element or a subsytem with its environment is called and is *port* indicated by line segments incident to the end of the component. For example,one port belongs to the capacitor C in Fig.2.3b while four ports are linked to the 0-element. Since elements and their ports are not isolated in a real-world system the *bond* is introduced being simply a combination of pairs of ports. This is illustrated in Fig.2.3c where two resistors and one inductance representing one-port elements are combined with the four-port element of a so called *0-junction.*

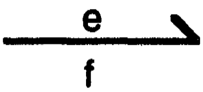


Figure 2.4: Bond Graph with through and across variable

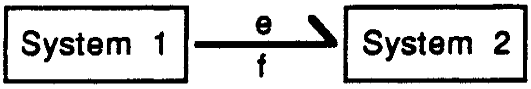


Figure 2.5: Power flow convention

2.3.2 Bond Graphs, Through and Across Variables

A group of multiport components combined is a so called *Bond Graph* whereby *nodes* symbolize the multiport elements attached by branches. The *bond* can also be described as a bold half arrow connecting two variables — one *across variable*, which in the Bond Graph language is usually referred as the *effort "e"*, and one *through variable* called the *flow "f"*.

As it turned out the power of any physical sytem can be always split into these two quantities. Fig.2.4 presents such a bond. Such a bond is normalized, i.e. the bond's arrow half and the across variable are always on the left side and the through variable is on the opposite side. The direction of the power flow along the Bond graph still has to be considered. The bond connecting system S1 and system S2 in Fig.2.5 points to the system S2 and thus the power will flow from S1 to S2 if the product of $P(t) = e(t) \cdot f(t)$ has positive sign. On the other hand, the power flows from S2 to S1 with P (t) having opposite sign.

2.3.3 Direct and Indirect Integral Quantities

Bond graphs are based on three direct and three indirect integral quantities. The direct quantities are represented by the two variables in ***effort*** *e* and ***flow*** *f* and their product as seen above being a ***power*** *P* or a conducted energy flux.

$$P(t) = e(t) \cdot f(t) \tag{2.11}$$

The ***displacement*** *q (t)* equal to the time integral of the flow, and the ***momentum*** *p (t)*, equal to the time integral of the effort, comprise two of the three indirect integral quantities. The third one is the ***energy*** *E (t)*, the time integral of the power over the interval $[t_0, t]$.

$$\begin{aligned} q(t) &= q(t_0) + \int_{t_0}^{t} f(\tau)d\tau && (2.12) \\ p(t) &= p(t_0) + \int_{t_0}^{t} e(\tau)d\tau && (2.13) \\ E(t) &= E(t_0) + \int_{t_0}^{t} P(\tau)d\tau && (2.14) \end{aligned}$$

The four generic variables — the effort, flow, displacement and momentum — are used in most of the common types of physical sytems as demonstrated in Table 2. 1.

2.3.4 Basic Multiport Elements

In the Bond graph methodology nine basic multiport elements can be found which are separated into four categories. The first three categories are considered the two *sources* (the effort source SE and the flow source SF), the *dissipation* (the resistor R), and the two *storages* (the inductance I and the capacitance C), and are completely characterized by 1-port elements. The fourth group formed by the *transducers* (the gyrator GY and the transformer TF) and the *junctions* (the 0– and the 1–junction) are 2– or at least 3-port elements, respectively. In Table 2. 2 these elements are given together with their mathematical definition and an example.

The little arrows pointing to the effort and flow source mean a signal path. The value of the signal is determined from outside of the system. Since one only considers a system inherent in another system the sytem considered has to be attached to outer systems. The universe is the largest system and is an exception to this rule. The sources isolated do not have any physical meaning since energy cannot be generated, but only transformed.

The resistor represents a special element in the sense that it is actually connected with its environment since its dissipated energy does not disappear from the system, but is transformed into heat energy flowing into the environment.

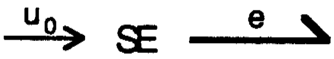
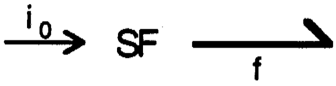
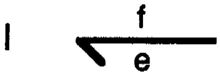
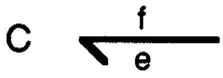
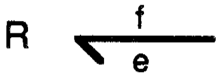
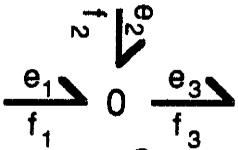
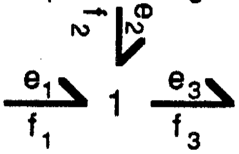
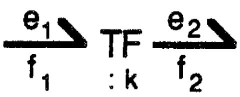
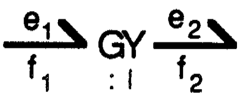
The capacitor and the inductor store energy flowing into them.

The most powerful strength of the Bond graph can be considered the ease with which different types of physical sytems can be attached. If systems are to be connected, the law of energy conservation has to be satisfied. Since the power is carried all the way along the Bond graphs of each system the link between can be easily made by introducing a transducer. In reality, the transducer and resistor "lose" energy, but this energy is actually transformed into heat.

Table 2.1: Power–(e,f) and energy–(p,q) variables [2]

	Effort	Flow	Generalized Momentum	Generalized Displacement
	e	f	p	q
Electrical	voltage u [V]	current i [A]	flux Φ [V·s]	charge q [A·s]
Translational	force F [N]	velocity v [m·s^{-1}]	momentum I [N·s]	displacement x [m]
Rotational	torque T [Nm]	angular velocity ω [rad·s^{-1}]	twist τ [Nm·s]	angle ϕ [rad]
Hydraulic	pressure p [N·m^{-2}]	volume flow ϕ_v [m^3·s^{-1}]	pressure rate Γ [N·m^{-2}·s]	volume V [m^3]
Chemical	chemical potential μ [J·mol^{-1}]	molar flow $\frac{dN}{dt}$ [mol·s^{-1}]	–	molar mass N [mol]
Thermodynamical	temperature T [°K]	entropy flow $\frac{dS}{dt}$ [W·°K^{-1}]	–	entropy S [J·°K^{-1}]

Table 2.2: Nine basic Bond graph elements

Name	Symbol	Example
Effort source		electrical: voltage source E = U_0 mechanical: force E = F
Flow source		el.: F = I_0 mech.: F = v (speed: wheels moved by road)
Inductance		el.: inductance L mech.: mass m
Capacitance		el.: conductor C mech.: spring C=1/k ; k spring constant
Resistance		el.: resistor R mech.: friction B
0–junction		el.: node of Kirchhoff's current law mech.: node with equal forces
1–junction		el.: loop of Kirchhoff's voltage law mech.: a mass with connected forces, thus, equal velocities
Transformer		el.: electrical transformer mech.: gear
Gyrator		el.–mech. : DC–motor

Defining the force of the mechanical system as a flow instead of an effort variable, and the velocity as effort variable, the gyrator of the electro-mechanical DC-motor would simply become a transformer. Thus these two types of transducers are really the same depending on the definition of the direct integral quantities.

All the efforts are equal at the 0-junction while the flows add up to zero. Identifying the voltage of the electrical system as effort and the current as flow the node of the Kirchhoff's current law is the analogon to the 0-junction. Since the 1-junction is the dual of the 0-junction, its properties are straightforward and are also summarized in Table 2. 2.

2.4 Causality, Singularities, Algebraic Loops

The concept of Bond graphs provides an easy way to translate from a physical system to a graphical representation while preserving the topological structure. But if this were the whole story the Bond graph could be only considered another variant to be added to the wide range of techniques already available. There is yet an additional feature making the Bond graph into a most potent tool for displaying the properties of a system and directing its analysis. This is the assignment of causalities to the bonds along with the power direction.

Each bond is connected with two structure elements, and thus is associated with two equations: one side determines its effort *"e"* and the other side determines its flow *"f"*. The decision whether the equation at the end of each bond has to be solved for either the effort "e" or the flow "f" is indicated by a small perpendicular stroke at one end of the bond as demonstrated in Fig.2.6 for a resistor.

If the stroke is attached to the end of the bond, this symbolizes the associate

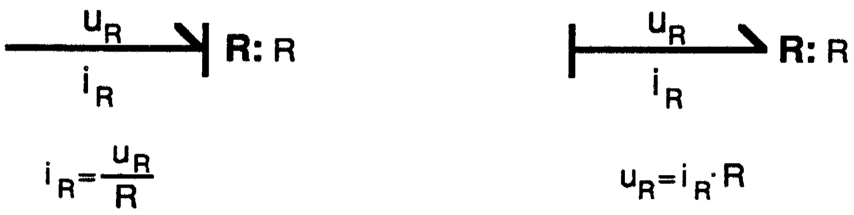


Figure 2.6: Causality of the resistor and the appropriate equations

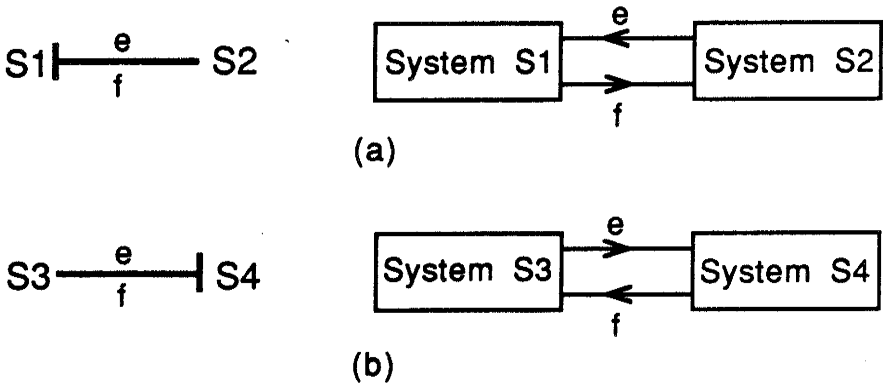


Fig. 2. 7 a.) effort e is input of system S1 and output of system S2
flow f is input of system S2 and output of system S1
b.) effort e is input of system S4 and output of system S3
flow f is input of system S3 and output of system S4

equation to be solved for the flow variable, and vice versa, if the stroke positioned at the beginning of the bond , then the associate equation has to be evaluated for the effort variable as illustrated in Fig.2.7. From this figure follows that the stroke denotes the end where the flow variable is computed.

This computational structure allows the state equations to be conveniently derived from the Bond graph representation, and then to be resolved for the causal variable. Table 2. 3 shows the nine basic elements from the previous section, but now their causal strokes have been added.

Most of these elements have mandated causalities. It comes as no surprise that the flow of a flow source is determined at the flow source, and thus the effort must be determined at the other end. Consequently the little bar is at the source. For the effort source, it is just the opposite.

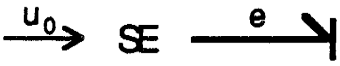
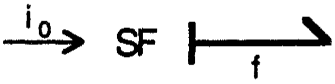
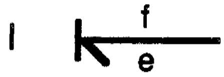
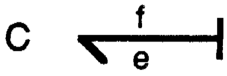
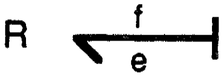
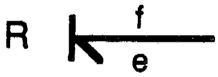
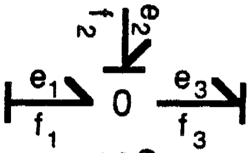
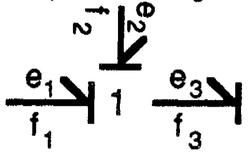
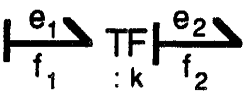
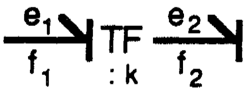
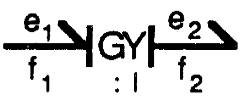
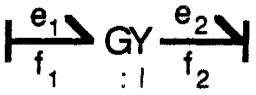
Since the goal is to derive state equations from the bond graphs the causalities of the inductance and the capacitor are recommended. Thinking in terms of electrical sytems the current i, for example, will be the state variable of the inductor L

$$d\, i_L/dt = u_L/L$$

and thus this equation will be evaluated for the current, or in Bond graph terminology the flow. If the inductor's stroke was placed the other way around, this would mean that the integral equation should be solved for u_L and the variable i_L should be computed from somewhere else in the circuit, and then would have to be differentiated in above equation.

Since the 0–junction connects all bonds with equal efforts and adds up all flows to zero there is only one equation generated that can be solved for one of these flow variables, indicated by exactly one stroke at this 0–junction. Since the 1–junction is the dual of the 0–junction, all the bonds attached but one have their strokes at the side of the 1–junction.

Table 2.3: Nine basic Bond graph elements and their assigned causality

Name	Causal Form	Causal Relation
Effort source		$e\,(t) = E\,(t)$ f arbitrary
Flow source		$f\,(t) = F\,(t)$ e arbitrary
Inductance		$f = \Phi_I^{-1}\left(\int^t e\;(\tau)\,d\,\tau\right)$
Capacitance		$e = \Phi_C^{-1}\left(\int^t f\;(\tau)\,d\,\tau\right)$
Resistance	 	$e = \Phi_R\,(f)$ $f = \Phi_R^{-1}\,(e)$
0–junction		$e_1 = e_2 = e_3$ $f_2 = f_3 - f_1$
1–junction		$f_1 = f_2 = f_3$ $e_3 = e_1 + e_2$
Transformer	  e1 TF e2 f1 : k f2 e1 TF e2 f1 : k f2	$e_1 = k\cdot e_2$ $f_2 = k\cdot f_1$ $e_2 = e_1/k$ $f_1 = f_2/k$
Gyrator	  e1 GY e2 f1 : l f2 e1 GY e2 f1 : l f2	$f_1 = e_2/l$ $f_2 = e_1/l$ $e_1 = l\cdot f_2$ $e_2 = l\cdot f_1$

The transformer transmits causality straight through itself with no change in direction, and thus has the causal strokes either both at the end or both at the start of its attached bonds. The gyrator, on the other hand, inverts the direction of causality: i.e., a flow causality into the gyrator results in an effort causality at the output. The resistor *"R"* has an arbitrary causality which can be used to adjust the causality to the system's requirements.

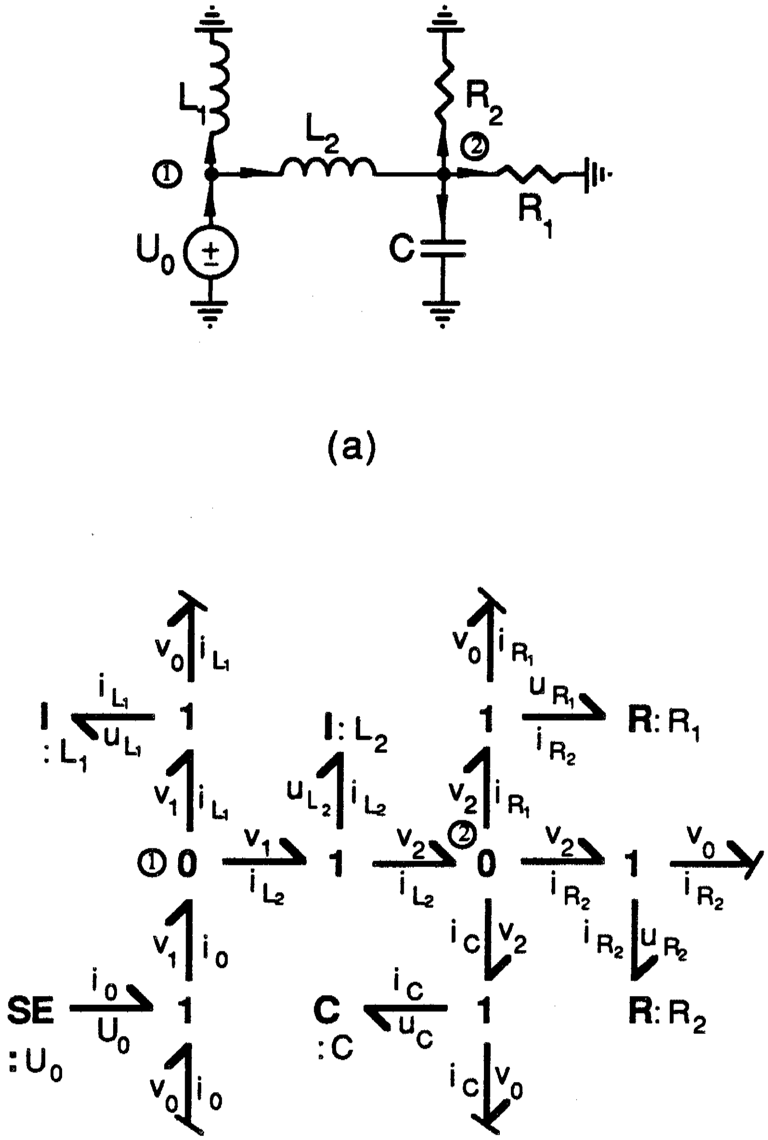
2.5 Fields

Sometimes it is quite useful to split a system into identifiable parts as then the structure can be more simply understood. In this context the term *"field"* is used as a discription of Bond graph elements which are grouped together according to their properties. In Chapter 4, the CS–field will be introduced which determines the change of a compound's mass influenced by all the reactions of a certain chemical system. There also exist different types of fields which are described in the book of *Blundell* [4].

2.6 An Example and Rules in Bond Graphs

Since electrical networks include all the elements and structures of the Bond graph methodology, Bond graph application can be shown using the RLC circuit introduced above. It should be noted that the chemical networks are not suitable for general examples. This is because chemical networks represent a special case for which new Bond graph elements have been developed in this project.

One of the great advantages of using Bond graphs is that they keep the topology of their physical system. This is shown in Fig.2.8 with the modified RLC circuit.



(a)

(b)

Figure 2.8: (a) Modified RLC circuit, (b) the detailed Bond graph

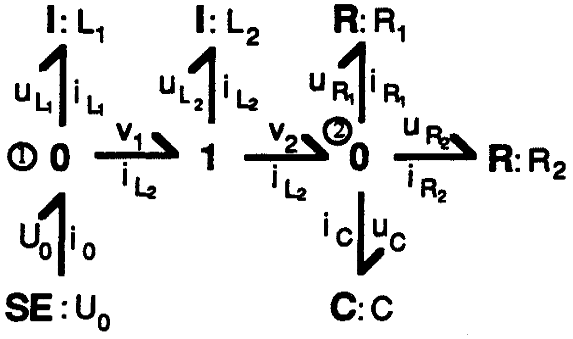


Figure 2.9: Simplified Bond graph of the RLC circuit

The rules for constructing the Bond graph of electrical circuits are very simple. Each node of the circuit is represented by one 0–junction since according to Kirchhoff's current law all voltages — the efforts — are equal and the currents — the flows — add up to zero. They are indicated by (1) and (2) in Fig.2.8. The 1–junctions represent the loops or meshes of the electrical circuit. Each branch has to be translated into the Bond graph language as two bonds connecting two 0–junctions with one 1–junction in between. From the sign convention of the power, it follows directly that the harpoons point in the same direction that was picked for the branch currents. This means that the harpoons are directed away from the sources and towards the passive elements R, L and C.

Fig.2.8 can be simplified after finding nodes which have zero voltages. This is the case for the reference node v_0, normalized to zero, by convention: thus, its power can be considered zero. Consequently, these harpoons are redundant and can be eliminated. Now all the 1–junctions except one have only two bonds attached. Whenever in either the 0–junction or the 1–junction two bonds are connected which point in the same direction they can be cancelled out. This cancellation of these junctions simply means that one redundant power balance with same input and output of the junctions was eliminated. Thus, the two bonds

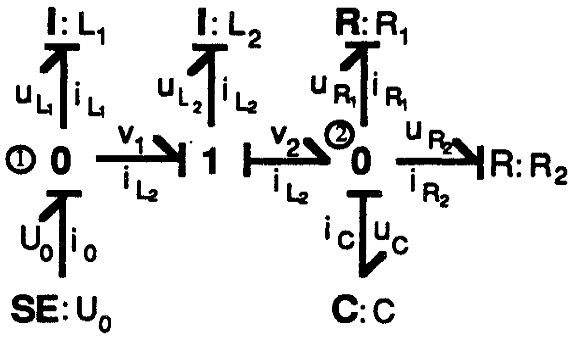


Figure 2.10: Bond graph of the RLC circuit with assigned causality

will merge to one. If it happens that two junctions of the same type are adjacent to each other, these two junctions can be combined into one. As a result, 1– and 0–junctions always toggle in any Bond graph model. Fig.2.9 illustrates the simplified Bond graph.

Setting the causal strokes is an easy procedure, and is illustrated in Fig. 2.10. Since the effort source *SE* has its causality at the 0–junction (1), the causality of the bond between this 0–junction and the 1–junction is fixed at its end. The inductor L_1 then causes no conflict with the causal rules. As a result of the inductor L_2 setting the stroke away from the 1–junction, the assigment of the causality of this bond inbetween the 1–junction and the rightmost 0–junction (2) is at this 1–junction. The capacitor C determines the position of the resistors' causality strokes since, as previously noted, 0–junctions can only have one causal stroke attached. All the causal rules are satisfied and thus this electrical circuit problem can be solved and a unique solution obtained.

The Bond graph causality property is a powerful instrument which can be used to reveal singularities and algebraic loops in a given physical system. This will be shown in the following paragraphs.

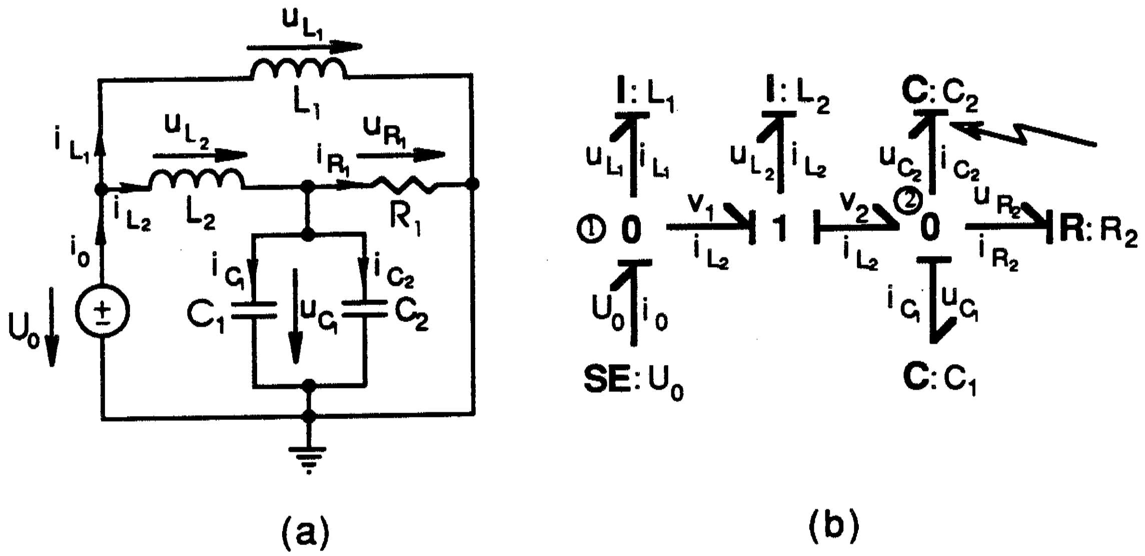


Figure 2.11: (a) RLC circuit with singularity, (b) its Bond graph

Fig. 2.11 is almost identical with the RLC circuit from above but the resistor R_2 is replaced by the capacitor C_2. The solution of this circuit is *degenerate* since the two capacitor in parallel yield a third order– instead of the expected fourth order system. The *singularity* can be eliminated, however, by combining the two capacitors into one.

To illustrate how an *algebraic loop* can be revealed the above circuit has been modified with the capacitor C being replaced by a third resistor R_3 shown in Fig. 2.12. This algebraic loop gives the choice of assigning causality which is to be ascertained by the equations for the RLC circuit.

$$\frac{d[i_{L_1}]}{dt} = \frac{u_{L_1}}{L_1} \tag{2.15}$$

$$\frac{d[i_{L_2}]}{dt} = \frac{u_{L_2}}{L_2} \tag{2.16}$$

$$i_{R_1} = \frac{u_{R_1}}{R_1} \tag{2.17}$$

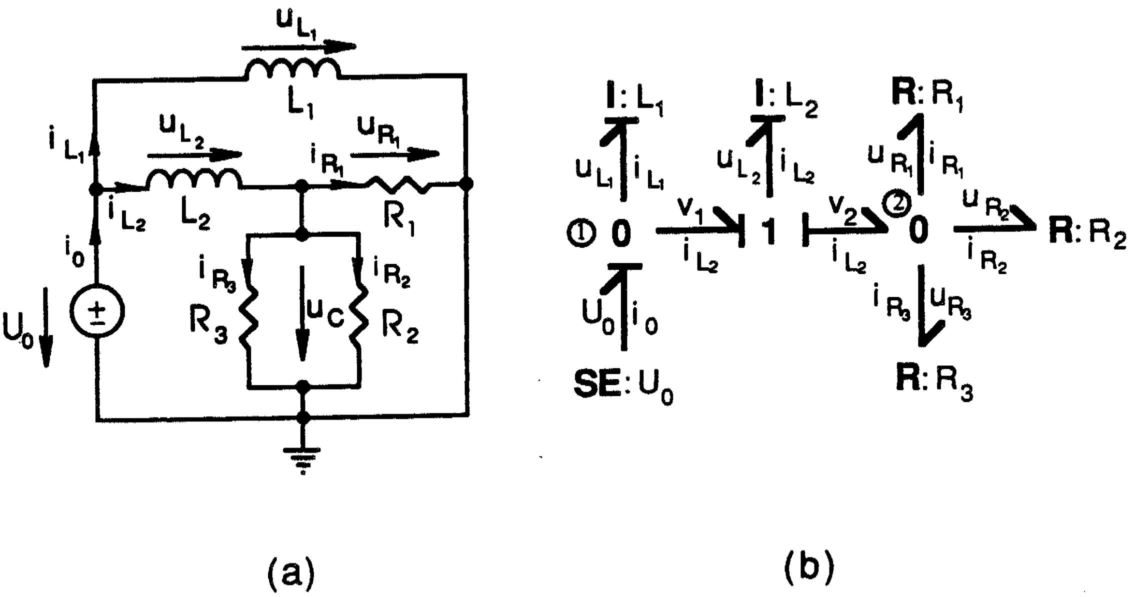


Figure 2.12: (a) RLC circuit with algebraic loop, (b) its Bond graph

$$i_{R_2} = \frac{u_{R_3}}{R_2} \tag{2.18}$$

$$i_{R_3} = \frac{u_{R_3}}{R_3} \tag{2.19}$$

$$[u_{L_2}] = U_0 - u_{R_3} \tag{2.20}$$

$$[u_{L_1}] = u_{L_2} + u_{R_1} \tag{2.21}$$

$$i_{R_3} = -i_{R_1} - i_{R_2} + i_{L_2} \tag{2.22}$$

The voltage u_{L_1} has to be determined by equ. (2.21) according to equ. (2.16). Thus, the voltages u_{L_2} and u_{R_1} must be evaluated either from equ. (2.20) or from equ. (2.22) where in both cases the voltage u_{R_3} must be known. Equ. (2.23) is derived by plugging equ. (2.18–20) and equ. (2.22) into equ. (2.17). The current i_{L_2} is assumed to be known since i_{L_2} is a state variable according to equ. (2.16).

$$u_{R_3} = R \cdot i_{L_2} \quad \text{where} \quad \frac{1}{R} = \frac{1}{R_1} + \frac{1}{R_2} + \frac{1}{R_3} \tag{2.23}$$

However, with equ. (2.16) an algebraic loop occurs since this equation assumes the unknown voltage u_{L_2} is known, u_{L_2} has to be computed somewhere else, but there is no further equation available to evaluate u_{L_2}.

Note that the computer will try to assign causalities illustrated by the brackets in the equations, but then will give an error message. In this case the algebraic loop can be solved be hand by simply eliminating the current i_{R_1} from the set of differential equations.

2.7 The Dual Principle in Bond Graphs

Any Bond graph model representing a physical system has a dual. Sometimes it is quite helpful to apply this dual principle especially when there are more 1–junctions than 0–junctions in the model. This will become clear when the Bond graph model is coded in the modeling language DYMOLA (Chapter 3). Fig. 2.13 illustrates most of the commonly used Bond graph elements and their duals. The 1–junction will be transformed into the 0–junction (which is represented conveniently by the *node* in DYMOLA) since the 1–junction does not have a direct translation into DYMOLA.

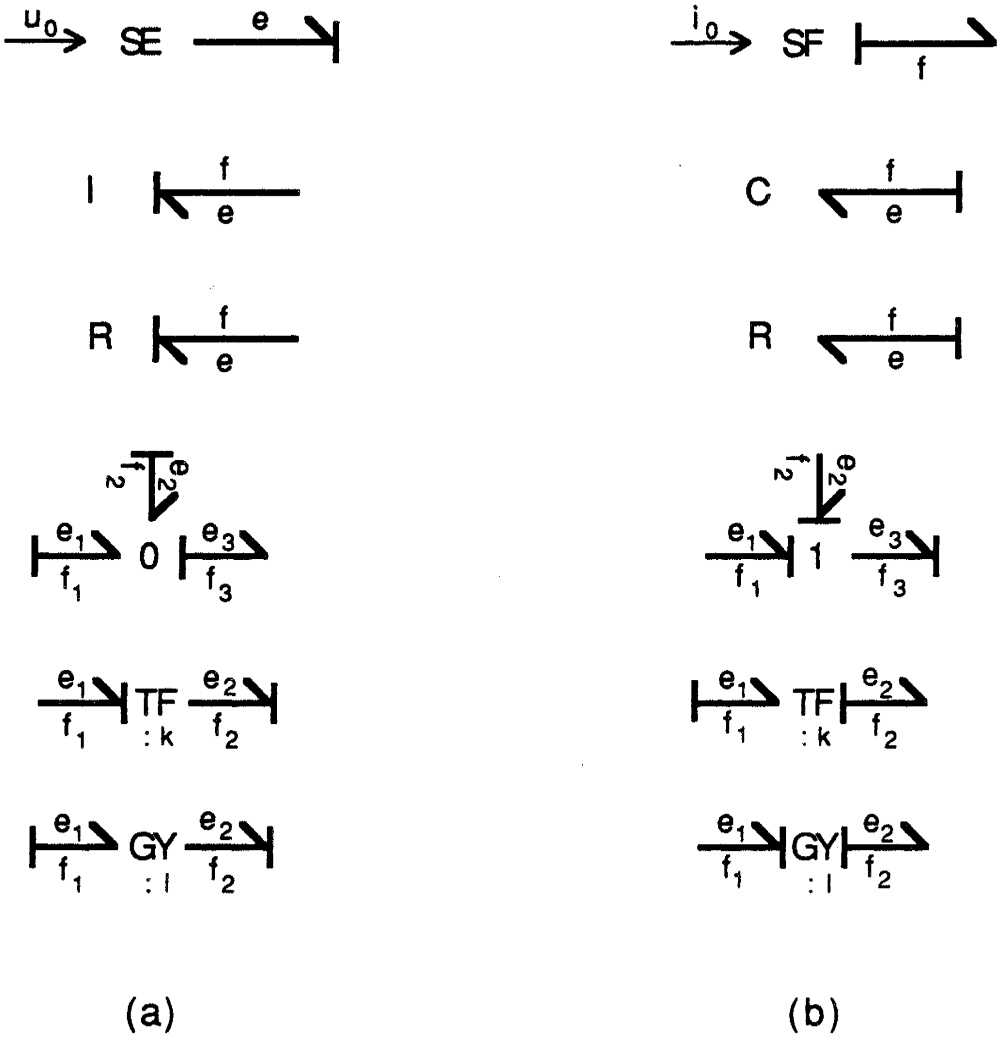


Figure 2.13: a.) Basic Bond graph elements and b.) their dual

Chapter 3

DYMOLA

This chapter introduces DYMOLA (DYnamic MOdeling LAnguage). The focus will not be on of detail software aspects, but rather on the main features, properties, unsolved problems, and the application of DYMOLA. Illustrative examples are included which show how DYMOLA works. It is also shown how the DYMOLA compiler processes DYMOLA source code into the simulation language DESIRE (DIrect Executing Simulation in REal time) [23] by means of an example. For further information concerning the software aspects of DYMOLA, Qingsu Wang's master thesis [30] can be consulted. This description of DYMOLA is closely related to Dr. François Cellier's book [1], Qingsu Wang's master thesis, and Hilding Elmqvist's PhD. dissertation [18].

"The first bond graph simulation language written in the early seventies was ENPORT [27,28]. This software used an approach similar to SPICE, i.e., it did not request causalities to be specified, and it transformed the topological input description into a branch admittance matrix which could then be solved by employing similar techniques to those used in SPICE. Consequently, ENPORT was able to handle structurally singular problems. The current version of the code, ENPORT–7 [28], offers an alphanumerical topological input language which, however, is not yet very user–friendly. A full–fledged graphical window system is currently under development. ENPORT–7 runs on various mainframe computers, but a slightly reduced version, ENPORT/PC, exists for IBM PC's and compatibles. ENPORT offers also a macro capability (somewhat comparable to the subcircuits in SPICE) which is, however, rather clumsy, and does not provide for fully hierarchical decomposition capabilities.

In the late seventies, another bond graph simulation language was developed at Twente University in the Netherlands, called THTSIM in Europe, and TUTSIM in the United States [8]. TUTSIM translates bond graphs into a state–space representation. The user is required to specify the causalities, and structurally singular systems cannot be handled. TUTSIM's simulation engine is somewhat poor in comparison with other state–space solvers such as ACSL [26]. The same research group is currently prototyping a new bond graph modeling system, CAMAS [16], which runs on SUN's, has nice graphics capabilities, and is able to handle algebraic loops. CMAS employs an object–oriented language (SIDOPS) for the model description which has similar properties as DYMOLA. Once available, this might become a good product.

The third product on the market is CAMP [20,21], a preprocessor to ACSL which translates bond graphs into ACSL programs. CAMP has the same limitations as TUTSIM, i.e., it does not handle algebraic loops or structural singularities, but it has the better simulation engine (ACSL). It is not truly flexible with respect to handling non–standard network elements. Non–linear elements need to be edited manually into the generated ACSL program which is very clumsy. There exists meanwhile also a graphical frontend for CAMP [22]. However, as in the case of ENPORT–7, the graphics editor is menu–driven rather than window–operated.

With exception of the unfinished CAMAS system, none of the above products is able to handle hierarchically structured models in a general fashion which is essential for the analysis of complex systems."[1]

For these reasons, it has been decided to use a different software tool. DYMOLA, developed by Hilding Elmqvist (1979) and enhanced by Qingsu Wang (1989), represents an interface between a hierarchical system modeling language and flat continuous simulation languages. Since DYMOLA does not provide for a simulation engine, it cannot be used as a simulation language but serves as a front–end to several simulation languages. Most of the CSSL–type languages sort the equations of a model but do not provide for the assignment of the causality to these equations. For example, the equation of Ohm's law

$$u = R \cdot i$$

inside a model cannot be solved for the current i:

$$i = \frac{u}{R}$$

DYMOLA provides this feature.

Currently, DYMOLA can be run on either UNIVAC computers — this is the SIMULA version of DYMOLA — or on PC compatibles and VAX/VMS machines, where DYMOLA is coded in PASCAL. DESIRE and SIMNON (another direct executing language), and FORTRAN represent the languages which are supported by DYMOLA. It would be a fairly simple task to enhance to another language, for instance the well–known ACSL (Advanced Continuous Simulation Language).

DYMOLA preserves the possibility for the user to code continuous systems modularly and hierarchically structured such that programs are better readable and understandable. For this, DYMOLA offers the *submodel*–concept as well as the *cut*–concept which will be explained in detail by means of figures and programs later in this chapter.

3.1 Some Selected Features of DYMOLA

In this section DYMOLA's *cut*–concept and *submodel*–concept are introduced.

3.1.1 Some Properties of DYMOLA

Since there already exist a number of documents on DYMOLA ([**18,19**], Dr. François Cellier's book, and Qingsu Wang's thesis), parts of their work are extracted by paraphrasing and quoting in certain paragraphs.

The following items demonstrate the different types of constants and variables used in DYMOLA, and how DYMOLA expressions are transformed by the DYMOLA equation handler.

1. Constants and Variables

   (a) DYMOLA destinguishes between two types of variables:
   The *terminal*–type variables act like global variables and are thus connected to variables or constants outside of the model. The domain of *local*–type variables is limited to inside the model.

   (b) *Terminal*–type variables can work either as inputs or as outputs. The user can explicitly declare *terminal*–variables to be either of *input*–type or of *output*–type. In case that a default value is assigned to terminal–variables they do not have to be attached to the outside of the model anymore; otherwise they do.

   (c) Constant variables of a model are defined as *constant*– or *parameter*–type. If constants are to be reassigned in simulations then the *parameter*–type will be used. Unless the value of the constant parameter is already assigned from outside of the model, a default value has to be declared. It is only possible to change the value in between simulation runs, but never during a simulation run.

Constants, as their name implies, obtain constant values that never change.

"*Externals* are similar to parameters, but they provide for an implicit rather than explicit data exchange mechanism. In this respect, they are similar to COMMON variables in a Fortran program. Externals are used to simplify the utilization of global constants or global parameters. For security reasons, the calling program must acknowledge its awareness of the existence of these globals, by specifying them as *internal*. Notice, however, that *"internal"* is not a declaration but only a provision for redundancy, i. e., all *"internal"* variables must be declared as something else also." [2]

2. Declaring Derivatives and Initial Conditions

   (a) In DYMOLA the *dot* notation is used as in DARE–P. The first and second derivative of a variable can be expressed by the **der(.)** and **der(.)**–operator, respectively, or abbreviated by the prime (′) or (″). These operators are allowed to be placed in any position of an equation, i.e., both on the left side and on the right side.

   (b) If initial conditions differ from the zero default value, they have to be assigned from outside of the model. The use of initial conditions is described in more detail in Section 3. 1. 3.

3. Transforming and Solving equations in DYMOLA:

   (a) Since in DYMOLA, equations with the syntax *expression* = *expression* are used, and the equations are solved for the appropriate variable, it does not matter whether an expression of "der*(chemical*

potential)" and the variable "***chemical potential***" is found on the left side of the equal sign.

(b) If there is a term of an equation multiplied by a zero parameter, then this term will be automatically cancelled out during the model expansion — the DYMOLA procedure to receive all the connected equations from the main program and its submodels .

For instance, if there is

$$C_a = 0.0 \tag{3.1}$$

and the model equation is to be

$$C_a \cdot \mathbf{der}(u_a) = i_a - i_b + \frac{u_a}{R_a} \tag{3.2}$$

then the equation handler sets the left term of equ.(3. 2) to zero. Equation (3. 2) becomes

$$0.0 = i_a - i_b + \frac{u_a}{R_a} \tag{3.3}$$

resulting in one of the three equations:

$$i_a = i_b - \frac{u_a}{R_a} \tag{3.4}$$

$$i_b = i_a + \frac{u_a}{R_a} \tag{3.5}$$

$$u_a = (i_b - i_a) \cdot R_a \tag{3.6}$$

depending on the causality assigned during the model expansion. If C_a is not equal to zero, equ.(3. 2) is transformed to

$$\mathbf{der}(u_a) = \frac{1}{C_a} \cdot (i_a - i_b + \frac{u_a}{R_a}) \tag{3.7}$$

Note that this feature is currently not yet imlemented.

(c) The concept of eliminating terms which are multiplied by a zero parameter has some problems. Sometimes it is quite useful to run a simulation, for example, in DESIRE, with different values assigned to one parameter. However, if the equation including the zero parameter is eliminated in the compiling process of the DYMOLA program to the DESIRE code, it is no longer possible to interactively replace the zero parameter by a non–zero value in DESIRE directly. The parameter has to be returned to DYMOLA in this case. However, the advantage of the process of eliminating zero parameters predominates the disadvantage of not being flexible to parameter change, since due to this elimination feature an entire class of structural singularities has been avoided.

3.1.2 The "Cut"–Concept and its Connections

Subsystems are introduced in most CSSLs (Continuous System Simuation Languages) in order to keep a structure hierarchical. The number of parameters and variables which are exchanged by the submodels will grow as the level of hierarchy is increased.

The mechanism which groups together variables exchanged by submodels is called the *"cut"*. One can imagine the procedure of using cuts as a complex connection mechanism of physical systems, for instance, to group individual electrical wires into buses or cables and cables into trunks." A cut is like a plug or a socket. It defines an interface to the outside world."[2].

As an example, two pressure tanks $S1$ and $S2$ are to be considered which are connected by a pipe where gas flows through (flow rate Φ, pressure p, temperature T), depicted in Fig.3.1.

The connection of the two pressure tanks is split to describe the two subsystems independently of each other. The connection of the two subsystems will be achieved by declaring the variables Φ, p, T in *cuts*:

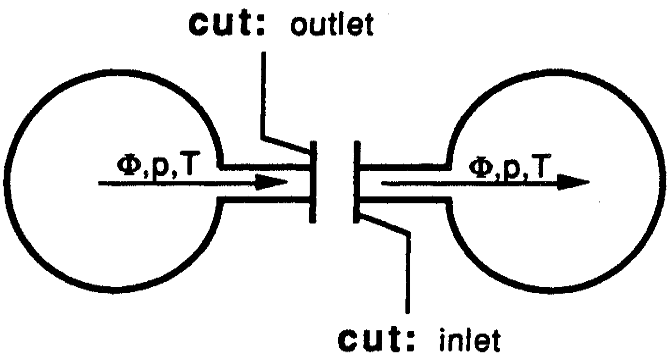


Figure 3.1: Pressure system cut to be descibed seperately

cut *inlet (Φ, p, T)*

cut *outlet (Φ, p, T)*

The variables could be attached from outside of the subsystems using:

$$\begin{aligned} S1.0 &= S2.0 \\ S1.p &= S2.p \\ S1.T &= S2.T \end{aligned}$$

However, it is more convenient to use the *connect*–statement in DYMOLA to connect an entire set of variables between subsystems:

connect *S1:outlet* **at** *S2:inlet*

The use "of named parameters instead of positional parameters upon invocation of a DYMOLA model helps with long parameter lists." [2]. The variables which are defined in cuts as either inputs or outputs are explained later in this section.

In summary, the declaration of cuts in submodels and the *connect*–statement can generally be described as follows:

cut *cut_name (cut_variables)*

connect *model1_spec:cut1_name* **at** *model2_spec:cut2_name*

In the cuts all the corresponding variables are set equal.

If this were the whole story of the cuts, they would be nothing but a tool to conveniently handle variables which are exchanged between submodels. However, this could also be done by simply declaring these variables as *terminals*. The two models describing both the mechanical friction are shown in Fig.3.2. The force F is chosen to be output in the first model while in the second model the causality is assigned to the velocity v, since v is chosen as the output.

These two models are almost identical except that the inputs and outputs are interchanged. DYMOLA overcomes the problem of duplicated models by distinguishing between the well-known *across*– and *through–variables* in the *cut*–statement itself. The connection of such cuts can be considered as an outlet of an electrical wire attached to an inlet transmitting both the voltage V (the across–variable) and the current I (the through–variable). Such a transmission of the two variable types is guaranteed by the following cut–statement:

cut *cut_name (across_variables/through_variables)*

Fig.3.3 shows the mechanical friction symbolized by an electrical resistor with two ports, X and Y.

Such a submodel can be coded as shown in Fig.3.4:

Three submodels *SM1, SM2, SM3* of the mechanical friction model (Fig.3.3) are to be connected in the *port* X as shown in Fig.3.5.

By connecting the three submodels at *cut* X,

model name : *friction1*

input : v

ouput : F

parameter : B

equations : $F = B \cdot v$

or

model name : *friction2*

input : F

ouput : v

parameter : B

equations : $v = F/B$

Figure 3.2: Two Models of a mechanical friction

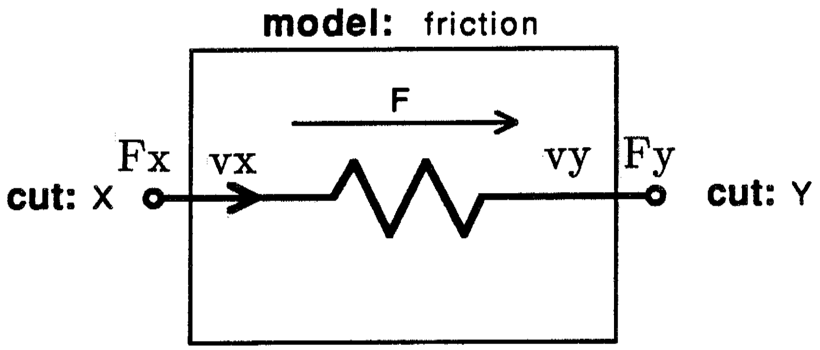


Figure 3.3: Mechanical friction symbolized by an an electrical resistor

```
model name: friction
        cut :         X(Fx/vx), Y(Fy/-vy)
        local :       F
        parameter B
        equations :vx = vy
                   vx = vy
                   v  = vx
                   F  = Fa - Fb
                   F  = v * B
end
```

Figure 3.4: Model of a mechanical friction

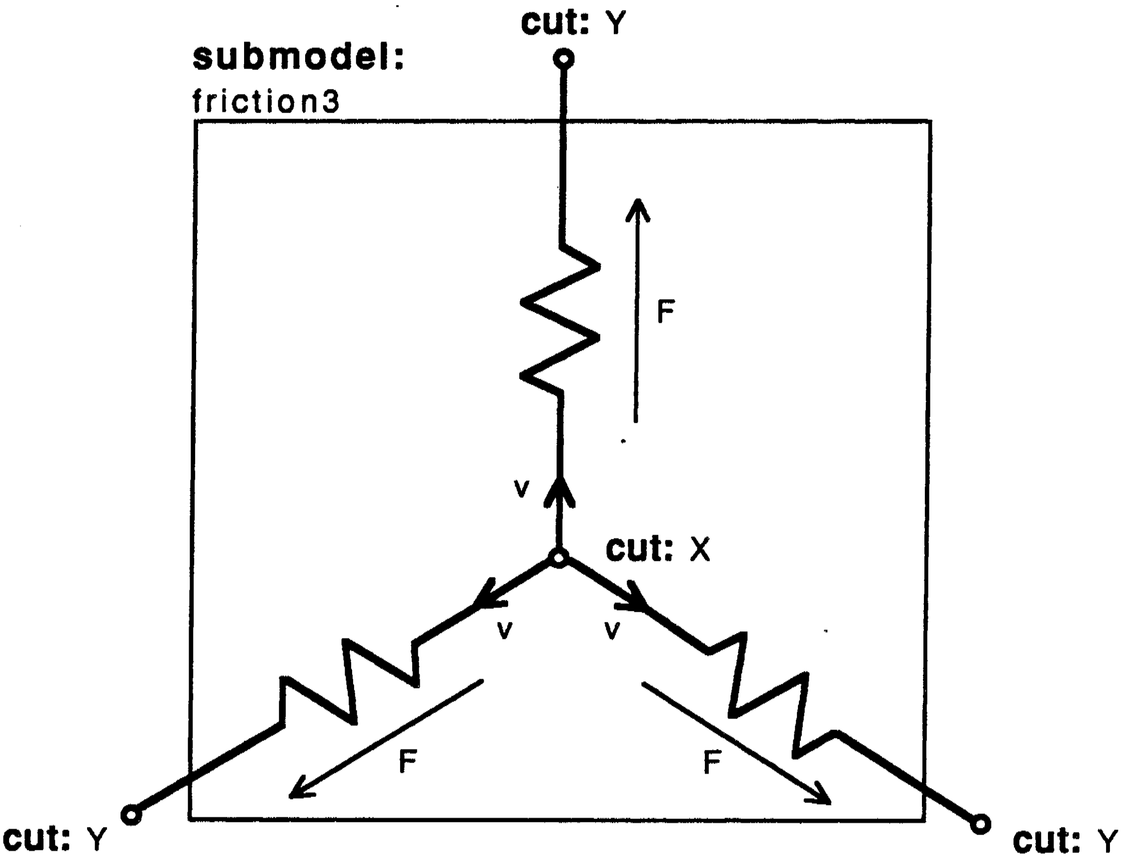


Figure 3.5: Three friction submodels connected in one *port X*

connect *SM1:X* **at** *SM2:X* **at** *SM3:X*

or abbreviated:

connect *SM1:X=SM2:X=SM3:X*

the following equations will be generated automatically:

$$\begin{aligned} SM1.Fa &= SM2.Fa \\ SM2.Fa &= SM3.Fa \\ 0.0 &= SM1.v + SM2.v + SM3.v \end{aligned}$$

This is no surprise since the *port X* represents the 0–junction in the Bond graph methodology where all the flow–variables (here the velocities) have to be added up to zero while the effort variables (here the forces) are equal to each other.
It should be noticed that all the flow–variables at cuts are *normalized* to point towards the subsystem. If a flow–variable is directed into the opposite direction, it must take a minus sign on the cut definition as already shown in Fig.3.4 of the *Y–cut.*

"Cuts" can be grouped hierarchically together as will be demonstrated using the friction model:

cut *X(Fx/vx) Y(Fy/-vx)*

1. Declaring a hierarchical cut:

   cut *Z[X,Y]*

   and connecting two submodels *SM1, SM2* with common cut Z:

   connnect *SM1:Z* **at** *SM2:Z*

generates the same result in DYMOLA as using:

```
connect SM1:X at SM2:X
connect SM1:Y at SM2:Y
```

2. "One cut can be declared as ***main cut***. The main cut is the default cut in a connection, i. e. it suffices to specify the model name to connect the main cut of a submodel."[2] For instance, the friction–model is to be connected to a system S with the same main cut:

```
model friction
    cut X(Fx/vx) Y(Fx/vx)
    main cut M[X,Y]
    ⋮
end
connect friction at S
```

3. There are often inherent connections in models, especially if the cut variables are to be appointed from a source to a destination. This is shown by the friction model of Fig.3.4, where its equations $F = Fx - Fy$ and $vx = vy$ will disappear by introducing a directed path from the input cut to the output cut declared in the model below:

```
model friction
    cut X(Fx/vx) Y(Fy/-vy)
    path trans<X—Y>
    ⋮
end
```

4. Two friction–submodels *SM1, SM2* can be connected inside of a model *MS* by simply writing:

connect *(trans) SM1* **to** *SM2*

which can be abbreviated as:

connect *(trans) SM1—SM2*

If a parallel connection of two friction-models *SM3, SM4* was desired in a model *MP*, then the code would appear as follows:

connect *(trans) SM3* **par** *SM4*

or abbriaviated:

connect *(trans) SM3* || *SM4*

The "\" operator denotes the reversed order in the parallel connection.

Fig. 3.6 illustrates the subsystems *SM1* and *SM2* both coupled in parallel and in series.

In summary, due to the existence of the cut-concept DYMOLA provides for the following features:

1. As seen from the friction-model, additional description of models caused by variation of I/O variables can be avoided. This is because variables that are declared as "terminals" or in "cuts", are indirected. Thus, for example, the two "friction-models" of Fig. 3. 2 with the input and outputs interchanged can be united in the one "friction-model" of Fig. 3. 4 with the input and output undirected.

2. By using the cut-concept, the physical laws which describe the static and dynamic properties of a model are seperated from the physical properties which dominate at connecting points of several subsystems.

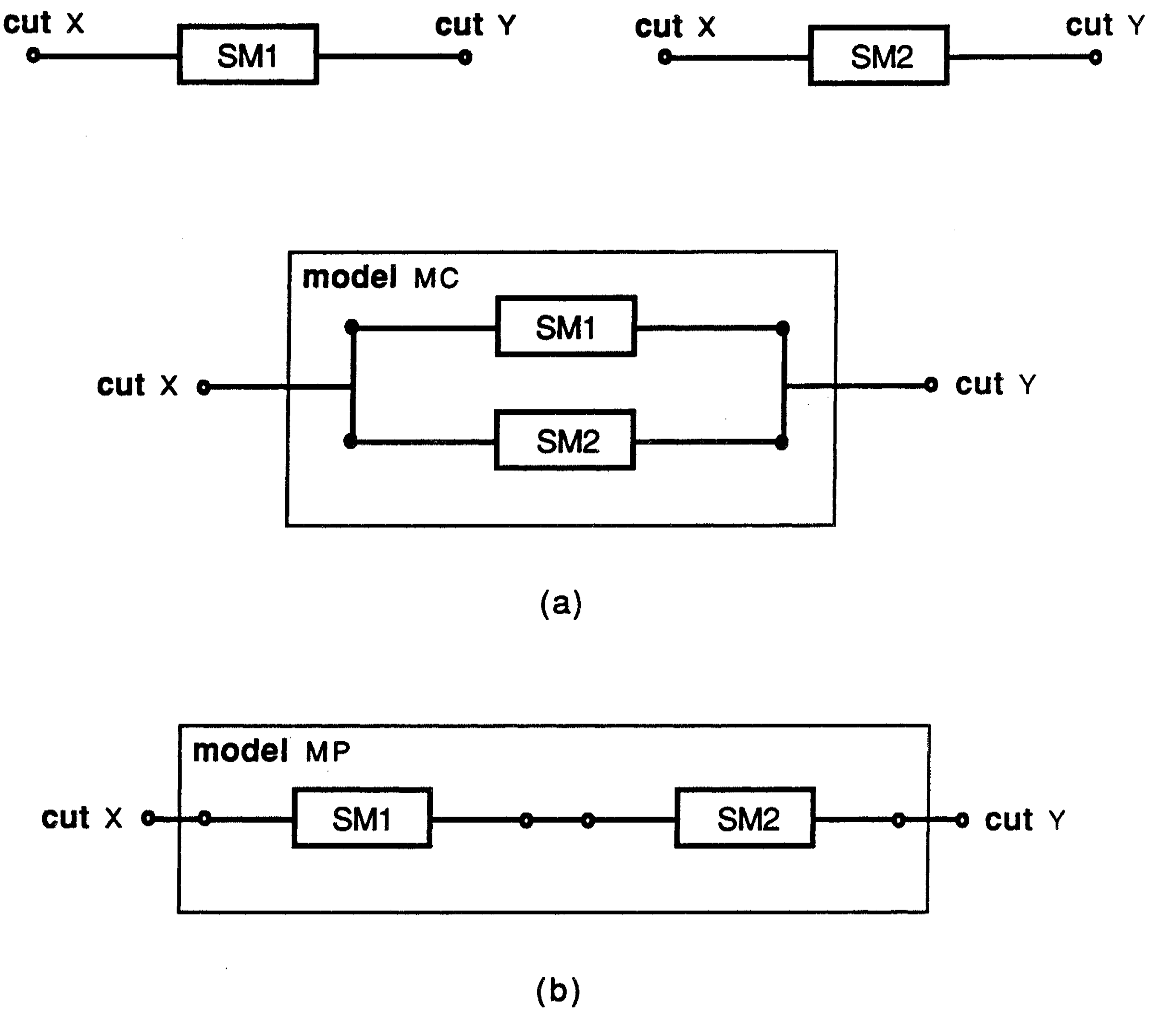


Figure 3.6: Coupled models in DYMOLA: (a) parallel, (b) series connection

3. Models in DYMOLA can be defined in proper modular form. This means that models can be developed as a set of independent modules being atomic or generic.

Finally, the use of cuts is necessary to exploit DYMOLA for implementing Bond graph models since DYMOLA comes up with an equivalence to the 0–junction (the DYMOLA–node) and the possibility to distinguish between across– and through–variables.

3.1.3 Hierarchical Model Structures, Submodels and Nodes in DYMOLA

As discussed above, the CSSL-languages together with DYMOLA use the concept of submodels to achieve hierarchical structures in programs.

This concept is going to be demonstrated by means of an example: A main system *"MS"* is decoupled into three subsytems: *SS1, SS2,* and *SS3*. Further, the subsystem *SS1* includes two additional subsystems *SSS1,* and *SSS2* shown in Fig.3.7. This hierarchical system can be coded in DYMOLA as illustrated in Fig.3.8.

But this description of a hierarchically structured system has a serious disadvantage. For instance, even if two subsystems are to be identical, their description has to be replicated in a program if they are to be called several times by the main program. The user can avoid such a replication by declaring subsystems as *"model type"*s in DYMOLA.

These are, for example, an electrical resistor and inductance that can be defined as generic, atomic *model types* shown in Fig.3.9.

The description, in general, is given by Fig.3.10.

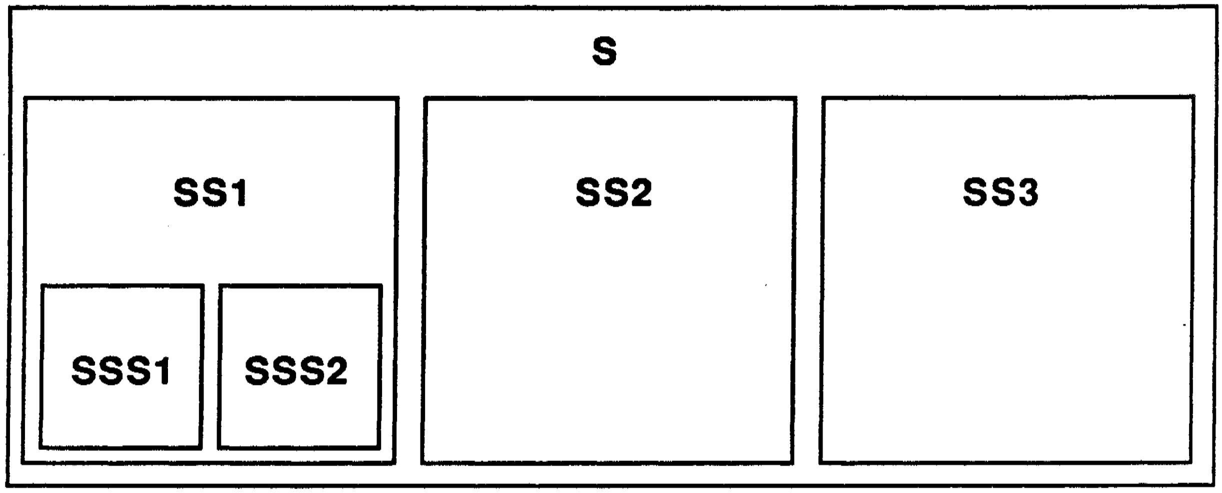


Figure 3.7: A hierarchically structured system in DYMOLA

The terms being in square brackets of Fig.3.10 are optional. Two of these optional terms need to be explained in more detail:

1. **Node:** "Sometimes it is useful to allow connections to take place inside a model instead of across model boundaries. For this purpose, DYMOLA provides a node declaration" [2] besides the *path* declaration introduced above. *Cuts* can be attached to the *nodes*. They then automatically assume the same hierarchical structure as the cuts that are attached to them.

2. **Submodel:** With the *submodel*-statement, the subsystems declared as *model types*, are called. The syntax of the *submodel* –statement can be described, in general, as follows:

   submodel [*(model_type_name)*] {*model_identifier*}
   [(*parameter_list*)] [(**ic** *initial_list*)]

   where *parameter_list* = {*number*} or {*parameter* = *number*}

```
model S
    model SS1
        model SSS1
        ...
        end
        model SSS2
        ...
        end
    end
    model SS2
    ...
    end
    model SS3
    ...
    end

...
end
```

Figure 3.8: Description of the hierarchical structure of a system

```
model type resistor
    cut X(V/I) Y(V/-I)
    path Z<X—Y>
    parameter R = 1.0
    R * I = V
end


model type inductance
    cut X(V/I) Y(V(-I)
    path Z<X—Y>
    parameter L = 1.0
    der(I) * L = V
end
```

Figure 3.9: Submodels *model types* resistor and inductance

```
model [type] model_name
    declaration_variable_type
    [submodel submodel_name]
    [node node_name]
    [connect variable_name]
    model_equations
end
```

Figure 3.10: General description of a DYMOLA *model* [*type*]

The curly brackets denote a repetition of one or more times.

If no *model_type_name* is given it is assumed to be the same as the *model_identifier.*

The *parameter_list* is used to change default values of the subsystems (*model types*). This can be done by either explicitly denoting the names of the parameters or by simply giving them in the same order as they are declared in the *model type.*

In the *initial_list* there are the initial conditions that are to be changed from the zero default value. Further, the *initial_list* can be treated like the *parameter_list* from above.

An example will be considered in which both the *submodel–* and the *node–*

statements are illustrated. Refer to Section 3. 3 for further information about these statement, where the RLC–circuit description of Chapter 2 is coded in DYMOLA.

By coupling the two atomic elements, ***model type resistor*** and ***inductor*** of Fig.3.9, a coupled model "pri" is generated. "Pri" stands for "parallel connected resistor and inductor". Its symbolic description as well as its code in DYMOLA are shown in Fig.3.11.

The "model type pri" of Fig.3.12 can be rewritten by using the *node*–statement as shown in Fig.3.12.

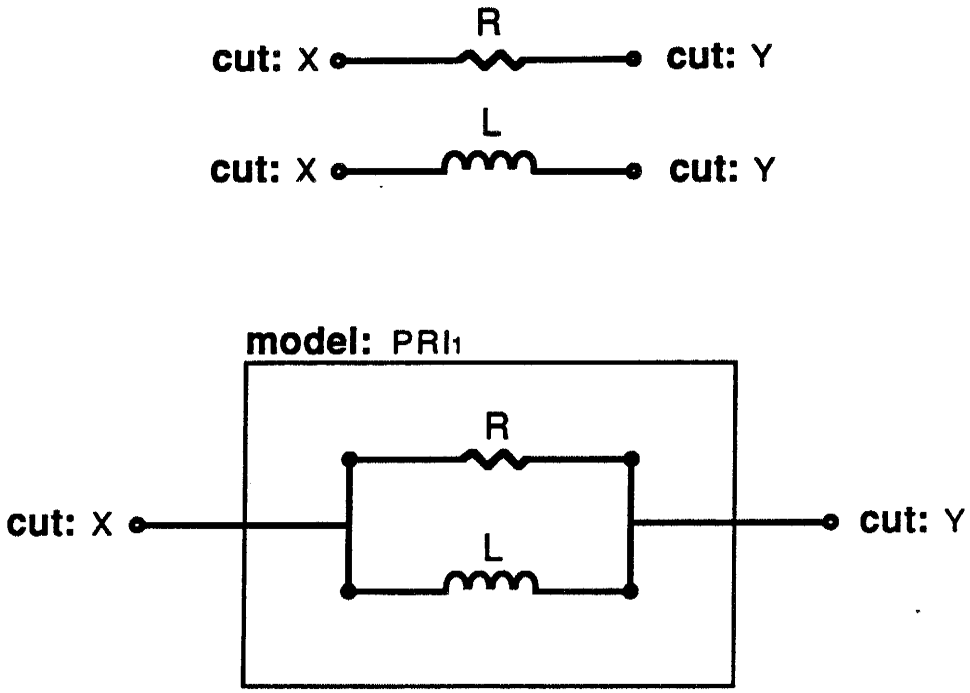
With the "model type pri" of Fig.3.11, using the *node*–concept, it is possible to connect the submodels more clearly to the "pri–model". Remember that both the "resistor–" and the "inductor–model types" are provided with a DYMOLA path. Thus, both the resistor and the capacitor can be connected from the "node N1" to the "node N2" by using the "from" and "to"-statements. Therefore, the direction of the model types have been considered automatically.

Using nodes is a convenient way to make several connections, like in the case of a power distributor.

3.2 Bond Graph Modeling in DYMOLA

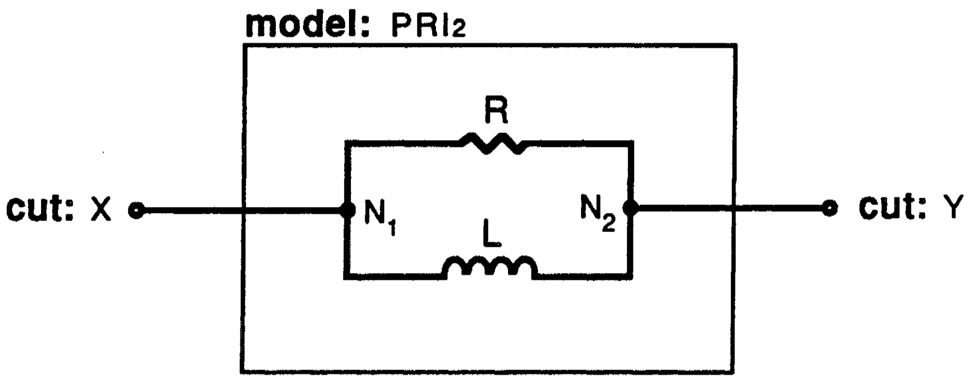
Bond graphs have been discussed in Chapter 2 and in this chapter it has been explained how an input file for DYMOLA has to be prepared. This section is dedicated to the use of Bond graphs in performing actual simulation runs.

For this, several rules are to be given to implement Bond graph models in DYMOLA programs where the RLC-circuit introduced in Chapter 2 is used to illustrate them. The approach is actually straightforward:



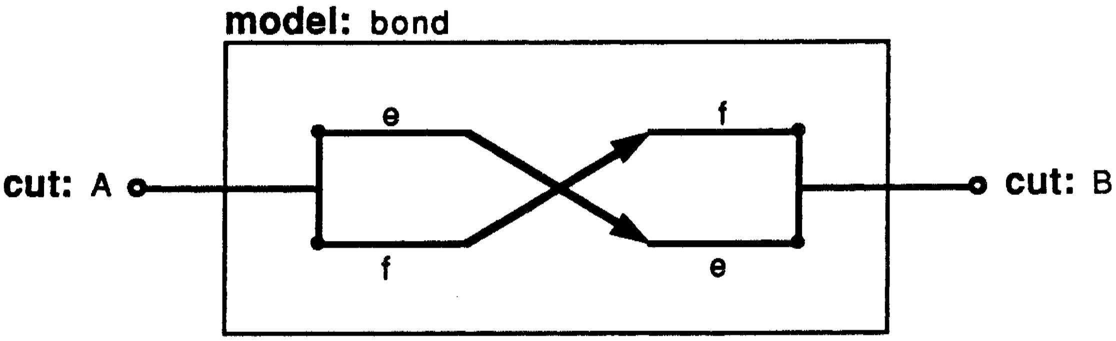
```
model type pri
        cut X(Vx/I)
        cut Y(Vy/-I)
        submodel resistor (R = 30)
        submodel inductor (L = 1.0E-3) (ic I = 1.0)

        connect resistor:X at inductor:X
        connect resistor:Y at inductor:Y
        connect X at resistor:X
        connect resistor:Y at Y
end
```

Figure 3.11: Model type pri using the *cut*–concept



```
model type pri
    cut X(Vx/I) Y(Vy/-I)
    submodel resistor (R = 30)
    submodel inductor (L = 1.0E - 3) (ic I = 1.0)
    node N1, N2

    connect resistor from N1 to N2, - >
            inductor from N1 to N2, - >
            X at N1, - >
            N2 at Y
end
```

Figure 3.12: Model type pri using the *node*–concept



```
model type bond
        cut A(x/y) B(y/-x)
        main cut C [A B]
        main path P<A—B>
end
```

Figure 3.13: Model type "bond" to exchange the effort and flow variable

1. The *0-junction* of the Bond graph terminology is equivalent to the *DYMOLA node*.

   There is no DYMOLA equivalent to the *1-junction*. However, as explained above, 1–junctions are the same as 0–junctions with the effort and flow variables interchanged. This circumstance can be used to create a model type "bond" to simply exchange the effort and flow variables as shown in Fig 3.13.

   Since in all the Bond graph models the junction–types toggle between 0– and 1–junctions, they can be both defined by regular DYMOLA "nodes" if they are connected with a model type "bond".

2. In order to avoid duplication of the different types of Bond graph elements R,C, L, TF, and GY, there is one additional rule to be considered:
   All the atomic model types of the Bond graph elements (except for the "bond") have to only be attached to 0–junctions. If they need to be attached to 1–junctions, then a "bond" has to be placed in between.

It should be noticed that DYMOLA is *case–sensitive.*

Fig. 3.14a shows again the Bond graph model of the RLC–circuit of Chapter 2 while Fig. 3.14b presents the expanded Bond graph according to the rules given above, i. e. all the elements are attached to 0–junctions. The causalities of the Bond graph were not marked "since DYMOLA is perfectly able to handle the causality assignment by itself (although no structural singularities yet)."[2]

In Appendix A, the DYMOLA model type library is given that describes the basic Bond graph elements. These model types are used to code the RLC–circuit in the DYMOLA program shown in Fig. 3.15, where $L1 = 1.0\ H, L2 = 2.0\ H, C = 2.0E - 1\ F, R1 = 8.0\ \Omega, R2 = 7.0\ \Omega$, and $U0 = \sin(0.5 * t)\ V$.

Following the rules given in Section 3. 2 the program in Fig. 3.15 is quite self–explanatory. Since there are four junctions in the Bond graph model and one 1–junction with three bonds attached, there are also four nodes that have to be declared in DYMOLA, and the "model type bond" has to be called three times. The arrow ($- >$) indicates contination lines.

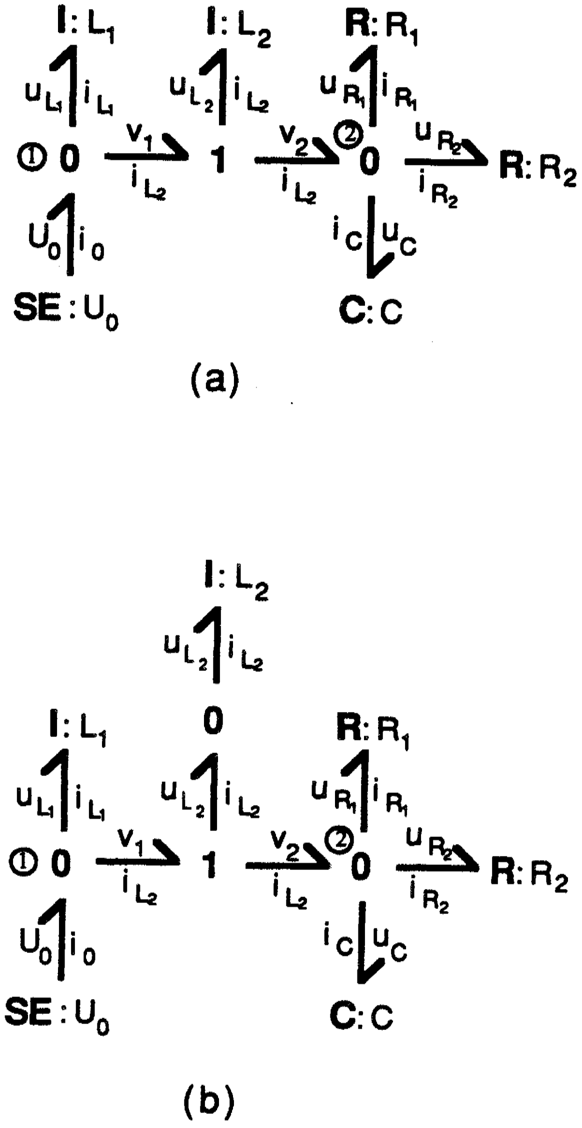


Figure 3.14: (a) The Bond graph of RLC–circuit of Chapter 2, (b) its DYMOLA expanded Bond graph

```
model RLC
   submodel (SE) U0
   submodel (R) R1 (R = 8.0), R2 (R = 7.0)
   submodel (I) L1 (I = 1.0), L2 (I = 2.0)
   submodel (C) C1 (C = 2.0E - 1)
   submodel (bond) B1, B2, B3

   node v1, iL2, uL2, v2
   input u
   output y1, y2

   connect U0 at v1, - >
             L1 at v1, - >
             L2 at uL2, - >
             C at v2, - >
             R1 at v2, - >
             R2 at v2, - >
             B1 from v1 to iL2, - >
             B2 from iL2 to uL2, - >
             B3 from iL2 to v2, - >
   U0.E0 = u
   y1 = u
   y2 = C1.e
end
```

Figure 3.15: The main program of the RLC circuit in DYMOLA

3.3 Generating DESIRE Programs, the Simulation Control Model

The objective of this section is to show how to generate a DESIRE program from a source code in DYMOLA. The DYMOLA preprocessor compiles DYMOLA code into DESIRE code. The individual steps of the DYMOLA preprocessor are demonstrated in Section 3.4, which is helpful if there is a DYMOLA program to debug. DESIRE has been chosen as simulation language since it is a direct executing language, and thus, very fast compiling programs.

Before explaining how to run the DYMOLA preprocessor, the *simulation control portion* (i. e., the experiment description) of the DYMOLA program will be introduced.

The basic information for the simulation control such as simulations step, communication points, and simulation time are required in order to run a DYMOLA program describing a continuous system. The syntax of the *control program* starts with "cmodel" and it normally will be stored under the same name as the one of the DYMOLA program, but now with the extension "ctl". The cmodel includes the following three parts:

1. basic part
2. run control block
3. output block

Sometimes it is desirable to store the output tabulated with respect to time in an ASCII–file. This can be done very easily by changing the output–channel from the screen into a file, shown in Fig.3.16.

The *basic part* contains the following information:

```
cmodel
    simutime 30
    step 1.0E - 2
    commupoints 1000
    input 1,u(depend,sin(0.5 * t))

    ctblock
        {GEAR-Integration Algorithm}
        irule 15
        ERMAX = 1.0E - 3
        DTMAX = 0.01
        disconnect 1 | connect 'RLC.tab' as output 1
        scale = 1.5
        XCCC = 1
        label TRY
        drun | if XCCC < 0 then XCCC = -XCCC | scale = 2 * scale | go to TRY
                  else proceed
        disconnect 1 | connect 'user' as output 1
    ctend

    outblock
        OUT
        type y1, y2
    outend

end
```

Figure 3.16: Simulation Control Program of the RLC–circuit

1. simulation time
2. simulation step size
3. number of communication points
4. inputs(outputs)

Qingsu Wang's thesis contains more information about the format of the basic part.

If there are *inputs* in the DYMOLA program, then there must be the same number of *inputs* in the control program.

The *run control block* is coded in DESIRE, involving DESIRE run control statements in the *run time output part* of DESIRE. Inside the run control block, the *integration rule* can be chosen by the statement "irule". If an integration method with *variable step size* is used, then the *absolute error* of the state variables and the *maximum step size* can be specified using the statements "ERMAX" and "DT-MAX", respectively. Refer to Granino Korn's book [23] for further information.

The feature of defining the *tabular function* in DYMOLA models has to be in a special format and can be looked up both in Qingsu Wang's thesis and Granino Korn's book [23,30].

The *output block* contains the simulation output requirements. There are four output statements "disptxy, dispt, type, and stash". Only two of them, "dispt" and "type", are to be described in more detail in this section. For further information about the other two statements, refer to Qingsu Wang's thesis.

In order to receive a graph of the simulation with respect to time on the screen, the output statement "dispt" has to be chosen. The "type"- statement causes an output in tabulated form with respect to time either on the screen, or, if the output channel has been changed to a file, then to this file.

The two lines inside the "ctblock" starting with the ***"disconnect"***–statement force the program to store the output in a file. Note that the ***"connect"***–statement used in the "ctblock", which is written in the DESIRE language, has a completely different meaning from DYMOLA's ***connect***–statement, namely to connect the output to the file "RLC.tab" or "user", respectively.

The portion in between the two "disconnect–lines" automatically scales the run–time display if the "dispt"–statement has been chosen. "**XCCC** is a DESIRE variable which is set −1 whenever the DESIRE program is interrupted with an "overflow". This happens when one of the displayed variables hits either the top or the bottom of the displayed window. At this time, the plot is simply rescaled, and the simulation is rerun by a new **drun** statement. "Since DESIRE is so fast, it is not worth the effort to store the results of the previous attempt, instead, the entire simulation is simply rerun." [2]

How the DYMOLA compiler processes DYMOLA programs together with the simulation control block is now looked at. This is illustrated by using the DYMOLA program of the RLC–circuit from above.

The command sequence invoking the DYMOLA preprocessor which immediately reads in the model definition, is shown:

```
$ dymola
  > enter model
   — @bond.lib
   — @SE.lib
   — @R.lib
   — @I.lib
   — @C.lib
   — @RLC.dym
  >
```

Being at the operating system prompt '$', the DYMOLA preprocessor is called which enters into an interactive mode indicated by its own prompt '>'. The statement "enter model" instructs DYMOLA to read a model pointed out by the next level prompt '–' as long as the model specifications are read in. It is possible to enter all the equations of the DYMOLA model directly, but it is more practical to enter them indirectly by using the add–sign '@' connected with the file names of the model types stored in the DYMOLA model type library. Finally, the specific RLC–circuit application model is loaded into the DYMOLA preprocessor intializing the first step of the compilation, i. e. all *submodel* references both of the main program and the model types, and the *connect*–statements are replaced by their model definitions and by the couplings equations, respectively. In Appendix A, the result of using the following two commands is shown :

```
> outfile RLC.eq1
> output equations
```

The "*outfile*"–statement commands DYMOLA to redirect the output to the file "RLC. eq1".

If it is desired to compile a DYMOLA program directly into a DESIRE program, the previous two commands do not have to be entered, but the following one does:

```
> partition eliminate
```

This manipulates all the equations above in the following way:

1. The causality to each equation is assigned.

2. The equations are sorted into an executable sequence.

3. The equations which have to be solved simultaneously are partitioned into smaller sets of equations.

4. Redundant equations are eliminated.

5. The equations are solved for the causal variables if possible.

The commands:

```
> enter experiment
— @RLC.ctl
```

loads the simulation control program and

```
> outfile RLC.des
> output desire program
```

will generate the DESIRE program and save it in the file "RLC.des" which is given in the Appendix A. This process will take a little longer since all the variables of the DYMOLA program have to be converted into DESIRE variables of eight characters at most.

This DESIRE program can now be executed by entering the following sequence:

```
  > stop
$ desire
   > load 'RLC.des'
   > run
```

which will generate the file "RLC.tab" with the desired output variables uc and i_{R_1} tabulated with respect to time as declared by the "disconnect"-statement of the control block.

3.4 Helpful Commands to Debug a DYMOLA–Model

Very often the DYMOLA compiler gives error messages or even worse the result of the simulation does not represent the expected result. There are some statements presented to follow the preprocessor and discover how it manipulates the equations of the DYMOLA model step by step. This is one way to debug a DYMOLA model.

As in Section 3. 3, both the model types of the DYMOLA model library and the specific DYMOLA model have to be entered.

Assigning the causality to the model equations, sorting them, and saving the result in the file "RLC.sor" is obtained by entering the following sequence:

```
> partition
> outfile RLC.sor
> output sorted equations
```

The file "RLC.sor" is given in the Appendix A. "The variables enclosed between square brackets "[]"are the variables for which each equation must be solved. This set of equations contains many trivial equations of the type $a = b$. DYMOLA is capable of throwing those out." [2] This is achieved by the following commands:

```
> eliminate equations
> eliminate parameters
```

The algorithm which is now invoked performs the following tasks along with several other ones which can be looked up in Dr. François Cellier's book:

All the equations of the type $a = b$ are eliminated by replacing all occurances of the variable a in all other equations by the variable b. " There is one exception to the rule: the eliminate operation will never eliminate a variable that was declared as *output*. If a is an output variable , it will throw the equations away as well, but in this case, all occurences of b are replaced by a. If both a and b are declared as *output*-variables, the equation will not be eliminated at all."[2]

It should be noticed that both commands *eliminate equations* and *eliminate parameters* are not currently available in DYMOLA, but are planned for the near future. The following set of instructions achieves a much reduced set of equations compared to the ones saved in the file "RLC.sor" and, in addition, the equations will be solved for their causal variables.

```
> partition eliminate
> outfile RLC.sov
> output solved equations
```

The file "RLC.sov" is presented in Appendix A.

This section is not intended to replace the comprehensive paragraphs in Dr. François Cellier's book, but is intended to represent a partial summary of the most important commands.

3.5 Features of DYMOLA of the Future

Since DYMOLA is currently in the development stage, there are still some problems with the language. A fair amount of research work is needed to make DYMOLA a comprehensive language and, thus, turn DYMOLA into production code. Some suggestions for the future extracted of Dr. François Cellier's book and Qingsu Wang's thesis follow:

1. DYMOLA is currently able to get rid of variables in equations of type $x = y$. However, it should be capable to eliminate variables in equations of type $x \pm y = 0.0$, too.

2. "DYMOLA should be able to recognize equations that have been specified twice, and eliminate the duplication automatically to avoid redundant equations."[2] This property is very important in achieving hierarchically connected submodels.

3. "DYMOLA should be able to handle superfluous connections, if we specify that $\phi_1 = -\phi_2$, it is obvious that also $\omega_1 = \omega_2$"(ω is the angular velocity and ϕ its corresponding angle). "However, DYMOLA will not let us specify this additional connection at the current time. Superfluous connections should simply be eliminated during model expansion."[2] This elimination can help to get rid of a whole group of structural singularities.

4. DYMOLA currently extracts the equations which cannot be solved individually, by grouping them together in a set of equations. DYMOLA should be able to transform this set of equations, of course, if they are linear, into matrix form. DESIRE could take care of the generated matrix representation to solve the set of equations by simply inverting the matrix. (In the near future DESIRE will provide algorithms which invert matrices.)

5. DYMOLA can solve linear equations. However, nonlinear equations like

$$A^2 + B^2 + 2 * C + 10.0 = 0.0$$

cannot be solved for A or B. Several possible solutions to handle nonlinear equations are proposed by Dr. François Cellier [**3**].

6. While DYMOLA can handle continuous systems it cannot handle discrete time models. This feature would make sense, since DESIRE allows discrete time models.

7. DYMOLA needs more powerful user interfaces. For instance, a graphic preprocessor of DYMOLA, called "*HIBLITZ*" [**19**], supporting all the modeling concepts of DYMOLA is already available on "Silicon Graphics (Iris)" but not on 386–compatibles yet.

Chapter 4

Chemical Reaction Networks Modeled by Bond Graphs

4.1 Preview

The main emphasis of the previous chapters has been the Bond graph methodology and its implementation in the modeling language DYMOLA, having been introduced by means of an example, an RLC circuit. The goal of this chapter is to apply Bond graphs to chemical reaction networks described by a set of chemical reactions. Before doing this, however, some basic equations have to be derived which are necessary to describe and to fully understand the Bond graph model of chemical reactions. See the reference list for further detailed information about chemical thermodynamics and kinetics [31–50,60–65]. It will be necessary to introduce some equations which represent simultaneously occurring energy transitions, but the intention of this chapter still represents the modular modeling of chemical systems.

4.2 Chemical Reaction Kinetics

The intention of this section is to describe equations for chemical reactions by looking at the macroscopic rather than the microscopic behavior of an occurring reaction. The chemical properties of matter, such as color, taste, smell, etc. will also not be considered but, instead, the calculation of the physical properties, i.e. the temperature, volume, pressure, concentration, etc. The big advantage of Bond graphs is their capability of very easily connecting different types of physical systems. Chemical power can be expressed as follows:

$$P_{chem1} = \dot{\mu} \cdot n \qquad\qquad P_{chem2} = \mu \cdot \dot{n} \tag{4.1}$$

where the chemical potential μ and the number of moles n can be declared as the across variables e while the time derivative of the chemical potential $\dot{\mu}$ and the molar flow rate $\dot{n}$ can be chosen to be the through variable f. One interesting point associated with this result is that the power is separated into two different forms as it has been suggested by declaring two flow– and two effort–variables. The reasons for this will be explained below.

Table 2. 1 shows that chemical besides thermodynamical systems do not have a generalized momentum. This is due to the fact that these systems lack an inertia element, one of the two storage elements existing in both electrical and mechanical systems, for example. With this information Dr. François Cellier concluded in his forthcoming book that the missing inertia element is the reason why it is possible to decouple the chemical kinetics from the chemical thermodynamics. This was exactly the decomposition property that has led the chemical sciences to split into two camps. The "kineticists" research reactions by dealing with the change of matter. Thereby they totally ignore the variation of chemical power. The "thermodynamicists" mainly consider chemical reactions near chemical equilibrium. Thus, they look neither at the change of mass nor at the power flow but at the statics of energy around the steady-state.

Even though the decoupling property seems to be very convenient in modeling chemical reactions, it will cause difficulties if other physical systems are to be linked to chemical systems, since two forms of energy have to be transmitted at the same time. Mostly chemical reactions occur endothermically or exothermically under definite conditions, i. e., substances react isobarically, isothermically, isochorically, etc. The hydraulics/pneumatics and thermodynamics of chemical system reactions have to be considered simultaneously in this cases. This idea is explored in this chapter.

In the following sections the kinetic equations will be derived before continuing with chemical thermodynamics.

4.2.1 Basics of Chemical Reactions

The following chemical reaction shall be considered using appropriate notation:

$$C_3H_8 + 5O_2 \rightleftharpoons 3CO_2 + 4H_2O \tag{4.2}$$

or

$$C_3H_8 + 5O_2 \xrightarrow{k1} 3CO_2 + 4H_2O \tag{4.3}$$

$$3CO_2 + 4H_2O \xrightarrow{k2} C_3H_8 + 5O_2 \tag{4.4}$$

or more generally:

$$\sum_{i=1}^{I} \nu_i^R \cdot A_i \rightleftharpoons \sum_{i=1}^{I} \nu_i^P \cdot A_i \tag{4.5}$$

where A_i represents the chemical symbol for the ith chemical species while ν_i^R and ν_i^P are the non-negative integer coefficients in the chemical reaction. By convention, chemical species on the right-hand side of equ.(4. 1) are called products of the reactants standing on the left–hand side of the right–left–harpoons. These harpoons mean that this reaction actually represents an equilibrium reaction, i.e., after a considerably long time the composition of the reaction substances will not change macroscopically any more. The reaction will not totally react either to the right or to the left side but the equilibrium composition will be something in between the two extremes. However, in reality this equilibrium will never be static but dynamic, or, in other words, this equilibrium is actually a flow equilibrium. This means the same number of reactants will react to the right side as the number of products being split again into reactants.

Due to the law of mass conservation, the sum of individual atoms of the reactants must be the same as the sum of individual atoms on the right–hand side. The act of balancing the atoms on both sides is called the stoichiometry. Equ.(4. 5) can be written differently as following:

$$\sum_{i=1}^{I} \nu_{i,k_i} \cdot A_i = 0.0 \quad \text{or} \quad \vec{\nu}_{k_i}^{T} \cdot \vec{A} = 0.0 \tag{4.6}$$

where

$$\nu_{i,k_i} = \nu_{i,k_i}^{P} - \nu_{i,k_i}^{R} \tag{4.7}$$

is called the stoichiometric coefficient.

Traditionally, ν_{i,k_i} is negative for reactants and positive for products and can be either positive or negative for species being both products and reactants at the same time. The non–negative coefficients ν_{i,k_i}^{R} and ν_{i,k_i}^{P} are called the reactant and product stoichiometric coefficients, respectively. It should be noted that the second subscript k_i of the stoichiometric coefficients stands for K different reactions occurring simultaneously. If there is a set of chemical reactions then the stoichiometric coefficients will build each a matrix of dimension i times k_i. In the simple case of one chemical reaction the stoichiometric coefficients are written in terms of a vector. As an example, the vectors of the stoichiometric coefficients of the above reaction are given as follows:

$$\begin{aligned} \vec{\nu}^R &= [1,5,0,0]^T && (4.8)\\ \vec{\nu}^P &= [0,0,3,4]^T && (4.9)\\ \vec{\nu} &= [-1,-5,3,4]^T && (4.10) \end{aligned}$$

This reaction of carbonhydrogens can also be said to be the reaction of one *mole* of propane C_3H_8 with *five moles* of oxygen O_2 to *three moles* of carbondioxide CO_2 and *four moles* of water H_2O. In the case of introducing the concept of a *"mole"* for describing a chemical reaction macroscopically the balance of atoms is precisely guaranteed.

One mole of any chemical substance exactly contains the same number of molecules as 12grams of C^{12}, namely $6.023 \cdot 10^{23}$. This number is mostly referred as *Avogadro* or *Loschmidt number*. Further, the relation between the *mass* m of a species and its number of moles n is expressed by the following equation:

$$m = n \cdot M \tag{4.11}$$

where M represents the *molar mass* of a chemical species. The molar mass can be succinctly expressed as the product of the *molecular mass* and the fraction $1 gram \cdot mol^{-1}$. The molecular mass of a molecule is roughly explained as the sum of the molecular masses of all the species building this molecule. The molecular mass of an atom is evaluated from its number of heavy elementary particles (protons and neutrons) multiplied by the mass of the heavy particles. Values of molecular mass are basic data of the chemical periodic system.

With the concept of moles a *normalized* form of the mass of every species taking part at a reaction has been obtained providing for the necessary condition of balancing the number of atoms. The number of moles of any pure substance is measured in *mole*.

In chemical laboratories, the ***molar concentration c*** is very often applied to express the amount of a certain substance in a liquid or in a gas phase. The molar concentration is the ratio of the number of moles n and the the volume V occupied by this quantity.

$$c_i = \frac{n_i}{V} \tag{4.12}$$

With this normalization the influence of a possibly variable volume is eliminated. The unit of the molar concentration is $mole \cdot m^{-3}$.

If the amounts of chemical species in a single reaction are measured in moles or in moles per volume then in the course of this reaction the number of moles will change. Mathematically, the ***molar flow rate*** ξ_i is described as the derivative of the number of moles with respect to time, and is measured in $mole \cdot sec^{-1}$.

$$\xi_i = \frac{dn_i}{dt} \tag{4.13}$$

It should be noted that in some textbooks the letter ξ is used for the reaction extent or the degree of advancement whose derivative is defined as

$$\frac{1}{V}\frac{d\xi_{k_i}}{dt} = \frac{1}{\nu_{i,k_i}}\frac{c_i}{dt} \tag{4.14}$$

For fixed volume this equation can be simplified to

$$\frac{d\xi_{k_i}}{dt} = \frac{1}{\nu_{i,ki}}\frac{dn_i}{dt} \tag{4.15}$$

which is well known as the ***"law of definite properties"***. In this project the first application of the letter ξ will prove to be more convenient for describing the dynamic behavior of chemical reactions also with changing volume.

The ***amount of mixture*** x_i is introduced as the number of moles of each substance i divided by the sum of the number of moles of all pure substances contained in the mixture.

$$x_i = \frac{n_i}{n} \quad \text{where} \quad n = \sum_{i=1}^{I} n_i \tag{4.16}$$

The *reaction rate* directly depends on the concentration of reactants and on the *reaction constant* k_i. In order to get their mathematical relation the microscopic behavior of the reaction has to be considered. A reaction will only take place if all the involved partner molecules come together at the same time at the same place. The probability of such an event is directly proportional to the density of every involved partner molecule. On the other hand, the density in a unity volume is nothing but the concentration. One can assume that the probability of any molecule to be at a specific time at a certain place is statistically independent of the probability of the other molecule to be at the same time in the same volume element. Consequently, the product of the reactants' concentration represents their joint probability to react with each other. In the propane–oxygen reaction, for example,

$$\begin{aligned} r_1 &= k_1 \cdot c_{C_3H_8} \cdot c_{O_2}^5 && (4.17) \\ r_2 &= k_2 \cdot c_{CO_2}^3 \cdot c_{H_2O}^4 && (4.18) \end{aligned}$$

are the reaction rates of this equilibrium reaction.
In general, the reaction rates are given as:

$$r_{k_i} = k_i \prod_{i=1}^{I} c_{A_i}^{\nu_{i,k_i}^R} \qquad (4.19)$$

The change of one reaction participant's concentration in a set of chemical reactions is the sum of all the reaction rates of these reactions in which the participant is involved minus the sum of all the reaction rates of those reactions in which it is eliminated. This can be expressed using the stoichiometric coefficients as follows:

$$\begin{aligned} \frac{dc_i}{dt} &= \sum_{k_i=1}^{K} \nu_{i,k_i}^P \cdot r_{k_i} - \sum_{k_i=1}^{K} \nu_{i,k_i}^R \cdot r_{k_i} \\ &= \vec{\nu}_i^T \cdot \vec{r} && (4.20) \end{aligned}$$

Using this notation, the equations of the above chemical reaction become:

$$\frac{d}{dt} c_{C_3H_8} = r_2 - r_1 \qquad (4.21)$$

$$\frac{d}{dt}c_{O_2} = 5r_2 - 5r_1 \tag{4.22}$$

$$\frac{d}{dt}c_{CO_2} = 3r_1 - 3r_2 \tag{4.23}$$

$$\frac{d}{dt}c_{H_2O} = 4r_1 - 4r_2 \tag{4.24}$$

However, these equations do not perfectly describe the reaction system since the chance of seven reactants in equ.(4. 18) colliding in the same instant and at the same place is highly unlikely, and thus, may never occur. Chemical experiments have shown that the maximum of three or four participants involved in one reaction step will never be exceeded and, therefore, the above brutto reaction has to be split into a set of chemical reactions as will be done in the case of the hydrogen–bromine reaction in the following section. The propane–oxygen reaction is actually a complex chemical reaction with radicals involved.

4.2.2 The Temperature Dependence of the Reaction Rate Constants

In the previous sections the kinetic equations together with the reaction rate constants have been considered to be completely decoupled from the thermic system. However, heat energy, which is either absorbed or generated in a reaction, is fed back to the chemical system again. This leads either to cooling or heating of the system in which the temperature varies. Experiments have shown that most of the chemical reaction rates generally increase very fast as the temperature rises. This temperature dependence can only be explained as a temperature dependence of the reaction rate constants, which increases exponentially with linearly increasing temperature. The following empirical relation between the temperature and the reaction rate coefficient has been explored

$$k = A \cdot e^{\frac{-b}{T}} \tag{4.25}$$

where b and A denote empirically determined constants.

In order to interpret equ.(4.25), Arrhenius (1889) assumed these particles to react in a collision with each other if their energy exceeds a certain amount of energy E_a. For an ideal gas, the number of particles fulfilling this condition was found by Boltzmann to be:

$$\frac{c_{E_a}}{c} = e^{\frac{-E_a}{RT}} \tag{4.26}$$

where c denotes the total concentration of a reaction substance.

The reaction rate r of a bimolecular reaction is assumed to be proportional to the concentration c_{E_a} of the "activated" particles. Further, the reaction rate increases with the growing number of collisions (CO) being by itself proportional to the total concentration c.

$$r = a \cdot c_{E_a} \cdot CO = a \cdot e^{\frac{-E_a}{RT}} \cdot c \cdot CO \tag{4.27}$$

or

$$r = a \cdot e^{\frac{-E_a}{RT}} \cdot c \cdot b \cdot c \tag{4.28}$$

With $A = a \cdot b$, equ.(4.28) becomes:

$$r = A \cdot e^{\frac{-E_a}{RT}} \cdot c^2 \tag{4.29}$$

Equ.(4. 29) is usually referred to as *Arrhenius' equation* with

$$k = k_0 \cdot T^m \cdot e^{\frac{-E_a}{RT}} \tag{4.30}$$

where m can vary between zero and four. Usually $m \approx 0.5$.

The constant b of the empirical equ.(4.25) can be identified as the factor E_a/R where E_a denotes the *activation energy* by Arrhenius. The reaction will take place if the temperature is sufficiently high so that the collision energy exceeds the activation energy. In the exothermic case, the temperature of the system grows such that the reaction is accelerated, which can finally lead to an explosion in which the energy stored in the matter of the reactants is mainly transformed into heat energy and pressure/volume work.

4.2.3 An Example for a Complex Chemical Reaction: The Hydrogen–Bromine Reaction

One might expect the hydrogen–bromine reaction with its chemical brutto reaction:

$$H_2 + Br_2 \rightleftharpoons 2HBr \tag{4.31}$$

or equivalently,

$$H_2 + Br_2 \xrightarrow{k_1} 2HBr \tag{4.32}$$

$$2HBr \xrightarrow{k_2} H_2 + Br_2 \tag{4.33}$$

to have a steady–state of:

$$\frac{k_1}{k_2} = \frac{c^2_{HBr}}{c_{H_2} \cdot c_{Br_2}} \tag{4.34}$$

where equ.(4. 34) was found from the set of differential equations with zero derivatives. But experiments of Bodenstein and Lind showed in 1906 that in the gas phase, over the temperature range of $200 - 300°C$, the empirical equation

$$\frac{d}{dt} c_{HBr} = \frac{k \cdot c_{H_2} \cdot c^{1/2}_{Br_2}}{1 + k' \cdot c_{HBr}/c_{Br2}} \tag{4.35}$$

where $k' = 0.1$ and $k \propto e^{\frac{E}{RT}}$ and $E = 175kJ \cdot mol^{-1}$, was approximately followed for the concentration change HBr. Equ.(4. 34) cannot be true since the concentration of bromine in equ.(4. 35) has some influence on the steady–state concentration of hydrogen–bromine. The concentration coefficient of bromine has an exponent of 1/2 rather than 1, as in equ.(4. 34).

Christiansen, Herzfeld, and Polanyi proposed the following set of chemical equations thirteen years later to explain what actually happens in reality:

$$Br_2 \xrightarrow{k_1} 2Br^\bullet \tag{4.36}$$

$$2Br^\bullet \xrightarrow{k_2} Br_2 \tag{4.37}$$

$$Br^{\bullet} + H_2 \xrightarrow{k_3} HBr + H^{\bullet} \tag{4.38}$$

$$HBr + H^{\bullet} \xrightarrow{k_4} Br^{\bullet} + H_2 \tag{4.39}$$

$$Br_2 + H^{\bullet} \xrightarrow{k_5} HBr + Br^{\bullet} \tag{4.40}$$

The hydrogen–bromine reaction was in fact found to contain a chain reaction, whereby the reaction (4. 36) represents the initiation step forming one of the so–called ***"chain carriers"***; in this case, the bromine radical $Br^{\bullet}$. Reaction (4. 38) and (4. 40) denote the propagation step, that is the chain carriers $Br^{\bullet}$ and $H^{\bullet}$ stay in these two reaction types by catalyzing them, until the bromine radical is removed in the termination step of reaction (4. 37). Reaction (4. 39) represents an inhibition step since the hydrogenbromine, the product of the entire reaction set, is split again.

The typical reaction rate constants were measured years later as, [**43,etc.**],

$$k_1 = 1.39 \cdot 10^8 \cdot T^{1/2} \cdot \left(\frac{189243}{R \cdot T}\right)^{1.97} \cdot e^{\frac{-189243}{R \cdot T}} \tag{4.41}$$

$$k_2 = \frac{k_1}{K(T)} \tag{4.42}$$

$$k_3 = 10^{11.43} \cdot e^{\frac{-82400}{R \cdot T}} \tag{4.43}$$

$$k_4 = \frac{k_5}{10} \tag{4.44}$$

$$k_5 = \frac{k_3}{10^{-0.54} \cdot e^{\frac{67400}{R \cdot T}}} \tag{4.45}$$

where $K(T)$ is the equilibrium constant of reaction number 3. These constants strongly depend on temperature. This dependency has been analyzed in detail in Section (4.2.1.1) where the Arrhenius equation has been introduced.

The concentration rates break down to the following set of equations:

$$\frac{d}{dt} c_{Br_2} = -r_1 + r_2 - r_5 \tag{4.46}$$

$$\frac{d}{dt} c_{Br^{\bullet}} = 2r_1 - 2r_2 - r_3 + r_4 + r_5 \tag{4.47}$$

$$\frac{d}{dt} c_{H_2} = -r_3 + r_4 \tag{4.48}$$

$$\frac{d}{dt}c_{H\bullet} = r_3 - r_4 - r_5 \tag{4.49}$$

$$\frac{d}{dt}c_{HBr} = r_3 - r_4 + r_5 \tag{4.50}$$

where the reaction rates are:

$$r_1 = k_1 \cdot c_{Br_2} \tag{4.51}$$

$$r_2 = k_2 \cdot c^2_{Br\bullet} \tag{4.52}$$

$$r_3 = k_3 \cdot c_{H_2} \cdot c_{Br\bullet} \tag{4.53}$$

$$r_4 = k_4 \cdot c_{HBr} \cdot c_{H\bullet} \tag{4.54}$$

$$r_5 = k_5 \cdot c_{H\bullet} \cdot c_{Br_2} \tag{4.55}$$

Since Bond graphs applied to chemical reactions require computation in moles, equs.(4.46–50) must be written in terms of number of moles and of molar flow rates rather than in terms of concentrations, in which the kineticists prefer to formulate the kinetic equations.

$$\frac{d}{dt}\left(\frac{n_{Br_2}}{V}\right) = -k_1 \cdot \frac{n_{Br_2}}{V} + k_2 \cdot \left(\frac{n_{Br\bullet}}{V}\right)^2 - k_5 \cdot \frac{n_{H\bullet}}{V} \cdot \frac{n_{Br_2}}{V} \tag{4.56}$$

$$\begin{aligned} \frac{d}{dt}\left(\frac{n_{Br\bullet}}{V}\right) &= 2k_1 \cdot \frac{n_{Br_2}}{V} - 2k_2 \cdot \left(\frac{n_{Br\bullet}}{V}\right)^2 - k_3 \cdot \frac{n_{H_2}}{V} \cdot \frac{n_{Br\bullet}}{V} \\ &\quad + k_4 \cdot \frac{n_{HBr}}{V} \cdot \frac{n_{H\bullet}}{V} + k_5 \cdot \frac{n_{H\bullet}}{V} \cdot \frac{n_{Br_2}}{V} \end{aligned} \tag{4.57}$$

$$\frac{d}{dt}\left(\frac{n_{H_2}}{V}\right) = -k_3 \cdot \frac{n_{H_2}}{V} \cdot \frac{n_{Br\bullet}}{V} + k_4 \cdot \frac{n_{HBr}}{V} \cdot \frac{n_{H\bullet}}{V} \tag{4.58}$$

$$\frac{d}{dt}\left(\frac{n_{H\bullet}}{V}\right) = k_3 \cdot \frac{n_{H_2}}{V} \cdot \frac{n_{Br\bullet}}{V} - k_4 \cdot \frac{n_{HBr}}{V} \cdot \frac{n_{H\bullet}}{V} - k_5 \cdot \frac{n_{H\bullet}}{V} \cdot \frac{n_{Br_2}}{V} \tag{4.59}$$

$$\frac{d}{dt}\left(\frac{n_{HBr}}{V}\right) = k_3 \cdot \frac{n_{H_2}}{V} \cdot \frac{n_{Br\bullet}}{V} - k_4 \cdot \frac{n_{HBr}}{V} \cdot \frac{n_{H\bullet}}{V} + k_5 \cdot \frac{n_{H\bullet}}{V} \cdot \frac{n_{Br_2}}{V} \tag{4.60}$$

Defining molar flowrate as:

$$\xi = \frac{d}{dt}n \tag{4.61}$$

and the derivative of the total volume V with respect to time as:

$$\Phi = \frac{d}{dt}V \tag{4.62}$$

equs.(4. 56–60) can be rewritten as:

$$\begin{aligned}
\xi_{Br_2} &= -k_1 \cdot n_{Br_2} + k_2 \cdot \left(\frac{n^2_{Br^\bullet}}{V}\right) - k_5 \cdot \left(\frac{n_{H^\bullet} \cdot n_{Br_2}}{V}\right) + \Phi \cdot \left(\frac{n_{Br_2}}{V}\right) && (4.63)\\
\xi_{Br^\bullet} &= 2k_1 \cdot n_{Br_2} - 2k_2 \cdot \left(\frac{n^2_{Br^\bullet}}{V}\right) - k_3 \cdot \left(\frac{n_{H_2} \cdot n_{Br^\bullet}}{V}\right) \\
&\quad + k_4 \cdot \left(\frac{n_{HBr} \cdot n_{H^\bullet}}{V}\right) + k_5 \cdot \left(\frac{n_{H^\bullet} \cdot n_{Br_2}}{V}\right) + \Phi \cdot \left(\frac{n_{Br^\bullet}}{V}\right) && (4.64)\\
\xi_{H_2} &= -k_3 \cdot \left(\frac{n_{H_2} \cdot n_{Br^\bullet}}{V}\right) + k_4 \cdot \left(\frac{n_{HBr} \cdot n_{H^\bullet}}{V}\right) + \Phi \cdot \left(\frac{n_{H_2}}{V}\right) && (4.65)\\
\xi_{H^\bullet} &= k_3 \cdot \left(\frac{n_{H_2} \cdot n_{Br^\bullet}}{V}\right) - k_4 \cdot \left(\frac{n_{HBr} \cdot n_{H^\bullet}}{V}\right) \\
&\quad - k_5 \cdot \left(\frac{n_{H^\bullet} \cdot n_{Br_2}}{V}\right) + \Phi \cdot \left(\frac{n_{H^\bullet}}{V}\right) && (4.66)\\
\xi_{HBr} &= k_3 \cdot \left(\frac{n_{H_2} \cdot n_{Br^\bullet}}{V}\right) - k_4 \cdot \left(\frac{n_{HBr} \cdot n_{H^\bullet}}{V}\right) \\
&\quad + k_5 \cdot \left(\frac{n_{H^\bullet} \cdot n_{Br_2}}{V}\right) + \Phi \cdot \left(\frac{n_{HBr}}{V}\right) && (4.67)
\end{aligned}$$

By introducing the *reaction flow rates*:

$$\begin{aligned}
\xi_{k1} &= k_1 \cdot n_{Br_2} && (4.68)\\
\xi_{k2} &= k_2 \cdot \left(\frac{n^2_{Br^\bullet}}{V}\right) && (4.69)\\
\xi_{k3} &= k_3 \cdot \left(\frac{n_{H_2} \cdot n_{Br^\bullet}}{V}\right) && (4.70)\\
\xi_{k4} &= k_4 \cdot \left(\frac{n_{HBr} \cdot n_{H^\bullet}}{V}\right) && (4.71)\\
\xi_{k5} &= k_5 \cdot \left(\frac{n_{H^\bullet} \cdot n_{Br_2}}{V}\right) && (4.72)
\end{aligned}$$

equs.(4. 63–67) become:

$$\begin{aligned}
\frac{d}{dt} n_{Br_2} &= -\xi_{k1} + \xi_{k2} - \xi_{k5} + \Phi \cdot \left(\frac{n_{Br_2}}{V}\right) && (4.73)\\
\frac{d}{dt} n_{Br^\bullet} &= 2\xi_{k1} - 2\xi_{k2} - \xi_{k3} + \xi_{k4} + \xi_{k5} + \Phi \cdot \left(\frac{n_{Br^\bullet}}{V}\right) && (4.74)\\
\frac{d}{dt} n_{H_2} &= -\xi_{k3} + \xi_{k4} + \Phi \cdot \left(\frac{n_{H_2}}{V}\right) && (4.75)\\
\frac{d}{dt} n_{H^\bullet} &= \xi_{k3} - \xi_{k4} - \xi_{k5} + \Phi \cdot \left(\frac{n_{H^\bullet}}{V}\right) && (4.76)\\
\frac{d}{dt} n_{HBr} &= \xi_{k3} - \xi_{k4} + \xi_{k5} + \Phi \cdot \left(\frac{n_{HBr}}{V}\right) && (4.77)
\end{aligned}$$

The reaction described by these equations will be illustrated with an example. Reactants of a certain amount are put in a tank of known volume with a piston at one end. After steady–state is reached, the piston is to be moved in, decreasing the total volume. From equs.(4. 73–77), it follows that the number of moles increases, which cannot be true since in a system without in– or outflow of mass, the total number of moles does not change. If the term with the Φ in equs.(4. 73–77) were cancelled out, the change in the number of moles of one species would only depend on the reaction flow rates of those reactions in which the species is involved. Consequently, a surplus term will be added to the set of differential equations (equs. (4. 46–50)). to make the two sets of differential equations consistent:

$$\begin{aligned}
\frac{d}{dt}c_{Br_2} &= -k_1 \cdot c_{Br_2} + k_2 \cdot c^2_{Br\circ} - k_5 \cdot c_{H\bullet} \cdot c_{Br_2} - \frac{\Phi}{V} \cdot c_{Br_2} & (4.78)\\
\frac{d}{dt}c_{Br\bullet} &= 2k_1 \cdot c_{Br_2} - 2k_2 \cdot c^2_{Br\bullet} - k_3 \cdot c_{H_2} \cdot c_{Br\bullet} \\
&\quad + k_4 \cdot c_{HBr} \cdot c_{H\bullet} + k_5 \cdot c_{H\bullet} \cdot c_{Br_2} - \frac{\Phi}{V} \cdot c_{Br\bullet} & (4.79)\\
\frac{d}{dt}c_{H_2} &= -k_3 \cdot c_{H_2} \cdot c_{Br\bullet} + k_4 \cdot c_{HBr} \cdot c_{H\bullet} - \frac{\Phi}{V} \cdot c_{H_2} & (4.80)\\
\frac{d}{dt}c_{H\bullet} &= k_3 \cdot c_{H_2} \cdot c_{Br\bullet} - k_4 \cdot c_{HBr} \cdot c_{H\bullet} - k_5 \cdot c_{H\bullet} \cdot c_{Br_2} - \frac{\Phi}{V} \cdot c_{H\bullet} & (4.81)\\
\frac{d}{dt}c_{HBr} &= k_3 \cdot c_{H_2} \cdot c_{Br\bullet} - k_4 \cdot c_{HBr} \cdot c_{H\bullet} + k_5 \cdot c_{H\bullet} \cdot c_{Br_2} - \frac{\Phi}{V} \cdot c_{HBr} & (4.82)
\end{aligned}$$

The concentration of the species will now increase if the volume decreases since the volume flow rate will get a negative sign, and therefore making the last term of equs.(4. 78–81) positive. Only with the assumption of constant volume can the last term be deleted. In many textbooks of chemical kinetics this last term is erroneously omitted for the variable volume case as well.

4.3 Chemical Thermodynamics

Since the goal of this chapter is to derive equations balancing the energy in a chemical system it is necessary to consider the thermochemical as well as the kinetic behavior which has given the relation between the number of moles in the reaction in the previous section.

4.3.1 Basic Definitions

The basic thermodynamics terms are to be explained first. Most chemical reactions generate heat, and are thus exothermic. Reactions which absorb heat are endothermic. These reactions have some thermodynamical constraints and occur under two constraints at the most, for example, isothermic (under constant temperature), isochoric (with constant volume), or isobaric (under constant pressure), respectively. The case of reactions without the in– or outflux of heat energy is called adiabatic while reactions with constant total entropy are called isentropic.

4.3.2 The Chemical Potential and the Two Power Balance Equations

Assuming that only pressure–volume work is possible the first and second law of thermodynamics give the following equation, found in many textbooks [**29**,**36**,**43**]:

$$\dot{U} = T \cdot \dot{S} - p \cdot \dot{V} + \sum_{i=1}^{I} \mu_i \cdot \dot{n}_i \tag{4.83}$$

where U represents the *internal energy*, T is the *temperature*, S stands for the *entropy*, p denotes the *pressure* and V the *volume*. $\dot{n}_i$ represents the *molar flow* of a pure substance i — in the Bond graph terminology it is symbolized by a ***through*** or ***flow*** variable —, μ_i characterizes the ***chemical potential*** of a pure substance and is, in Bond graphs terms the ***across*** or ***effort*** variable. The existence of a chemical potential comes as no surprise since in any chemical system mass flow is denoted by the molar flow rate $\dot{n}$ carrying chemical power along the Bond graphs. This power can be evaluated from the product of the molar flow rate and its adjugate variable —this is exactly the chemical potential— measured in $Joule \cdot mole^{-1}$.

Equ.(4. 83) shows three possible ways to simultaneously store energy in a chemical reaction. These are the ***thermic storage*** (expressed through the storage of heat energy), the ***hydraulic/pneumatic storage*** and finally the ***chemical storage.*** The existence of a difference in chemical potential is the driving force in a chemical reaction necessary for the transfer of matter while the difference in temperature and pressure are necessary for the transfer of heat energy and pressure–volume work.

The variables used in equ.(4. 83) can be subdivided into two categories. One category includes the property of pressure, temperature, and chemical potential remaining constant regardless of whether or not one were to describe all the matter or only a part of it. Since this category is independent of matter, it is called ***intensive.*** On the other hand, those properties, for example, the volume and the entropy, which vary directly with the changing mass, are called *extensive.* Besides the entropy and the volume, the internal energy and the molar mass of any pure substance fall into this category. If the extensive properties are divided/***normalized*** by the mass, then they certainly become intensive since they do not depend on matter any longer. By doing this, equ.(4. 83) becomes

$$\frac{d}{dt}(n \cdot u) - T \cdot \frac{d}{dt}(n \cdot s) + p \cdot \frac{d}{dt}(n \cdot v) - \sum_{i=1}^{I} \mu_i \cdot \frac{d}{dt}(n \cdot x_i) = 0.0 \qquad (4.84)$$

where u denotes the *molar internal energy,* s is called the *molar entropy,* and x_i represents the *molar fraction.*

Evaluating equ.(4. 83) gives:

$$n\left(\frac{du}{dt} - T\cdot\frac{ds}{dt} + p\cdot\frac{dv}{dt} - \sum_{i=1}^{I}\mu_i\cdot\frac{dx_i}{dt}\right) + \frac{dn}{dt}\left(u - T\cdot s + p\cdot v - \sum_{i=1}^{I}\mu_i\cdot x_i\right) = 0.0$$

Since the molar mass and the molar flow rate are independent of each other, the two terms between the parentheses have to be zero:

$$\frac{du}{dt} = T\cdot\frac{ds}{dt} - p\cdot\frac{dv}{dt} + \sum_{i=1}^{I}\mu_i\cdot\frac{dx_i}{dt} \tag{4.85}$$

$$u = T\cdot s - p\cdot v + \sum_{i=1}^{I}\mu_i\cdot x_i \tag{4.86}$$

Equ.(4. 86) again shows that chemical systems actually have three possibilities to store energy.

It proved to be convenient to use the *Gibbs free energy G* or its intensive equivalent g and the *enthalpy H*, or the *molar enthalpy h* to express chemical thermodynamic equations.

$$H = U + p\cdot V = T\cdot S + \sum_{i=1}^{I}\mu_i\cdot n_i \tag{4.87}$$

$$G = H - T\cdot S = \sum_{i=1}^{I}\mu_i\cdot n_i \tag{4.88}$$

or

$$h = u + p\cdot v = T\cdot s + \sum_{i=1}^{I}\mu_i\cdot x_i \tag{4.89}$$

$$g = h - T\cdot s = \sum_{i=1}^{I}\mu_i\cdot x_i \tag{4.90}$$

The *molar Gibbs free energy g* measures the total chemical energy stored in one mole of a pure substance. The Gibbs free energy G can also be written as

$$G = U + p\cdot V - T\cdot S \tag{4.91}$$

or by taking its derivative with respect to time:

$$\dot{G} = \dot{U} + p\cdot\dot{V} + \dot{p}\cdot V - \dot{T}\cdot S - T\cdot\dot{S} \tag{4.92}$$

Replacing the time derivative of the internal energy $\dot{U}$ by equ.(4. 83) this becomes:

$$\dot{G} = \dot{p} \cdot V - \dot{T} \cdot S + \sum_{i=1}^{I} \mu_i \cdot \xi_i \tag{4.93}$$

Using equ.(4. 88) again, equ.(4. 92) can be rewritten as:

$$\dot{G} = \sum_{i=1}^{I} \dot{\mu}_i \cdot n_i + \sum_{i=1}^{I} \mu_i \cdot \xi_i \tag{4.94}$$

With equ.(4. 94) compared to equ.(4. 93) it follows:

$$\sum_{i=1}^{I} \dot{\mu}_i \cdot n_i = \dot{p} \cdot V - \dot{T} \cdot S \tag{4.95}$$

Under the assumption of constant temperature and pressure this equation boils down to

$$\sum_{i=1}^{I} \dot{\mu}_i \cdot n_i = 0.0 \tag{4.96}$$

which represents the well–known *Gibbs–Duhem equation.*

In the following sections a closed system with total constant energy is assumed. By neglecting any other work except for pressure–volume work, the internal energy can be considered as constant:

$$U = constant$$

From equ.(4. 83) it follows

$$\dot{U} = T \cdot \dot{S} - p \cdot \dot{V} + \sum_{i=1}^{I} \mu_i \cdot \dot{n}_i = 0.0 \tag{4.97}$$

With equ.(4. 95) and equ.(4. 97) two separate power balance equations have been found helping to build the *"Chemical Reactor"* of the Bond graph model.

$$\begin{aligned} p \cdot \dot{V} &= T \cdot \dot{S} + \sum_{i=1}^{I} \mu_i \cdot \xi_i & (4.98) \\ \dot{p} \cdot V &= \dot{T} \cdot S + \sum_{i=1}^{I} \dot{\mu}_i \cdot n_i & (4.99) \end{aligned}$$

4.3.3 The Equation of State

The equation of state for liquids, simple solids and for an ideal gas will be shown in this section, as will the use of the equation of state for an ideal gas.

4.3.3.1 The Purpose of an Equation of State

Even though in the last section two power balances have been found, there are still four additional equations yielding a total of six unknowns to determine. Three of them were already determined, i.e., the two inputs from the environment and the one set of kinetic equations. The objective of this section will be to derive the sixth and last equation. For this purpose, a fundamental postulate of the thermodynamic state principle is cited:

"For any real system there exists a positive integer z such that if z intensive thermodynamic state functions are fixed, then all other intensive thermodynamic state functions are fixed."

This state principle must be regarded to be empirical since it is based on experimental observations near equilibrium.

The concept of an *equation of state* is determined as a direct consequence of this principle, whereby one intensive variable depends on the linear combination of the other intensive variables. If the value of one extensive variable is given, then the value of the mass can be evaluated.

Especially for an ideal gas the equation of state is:

$$p \cdot V = n \cdot R \cdot T \tag{4.100}$$

where V represents the ***volume***, p is the ***pressure***, T denotes the ***absolute temperature***, R is the ***ideal gas constant*** ($R = 8.314 Joule \cdot {}^{o}K^{-1} \cdot mole^{-1}$), and n represents its ***total number of moles***.

In order to illustrate Dalton's law a container with a fixed volume containing gaseous compounds is considered. All compounds are completely independent of each other and their measured pressure represent the partial gas pressure. Equ.(4.101) can be written for the individual component i:

$$p_i \cdot V = n_i \cdot R \cdot T \tag{4.102}$$

In the second case the ***partial volume*** V_i is given by the modified equation of state under the condition of constant temperature and pressure:

$$p \cdot V_i = n_i \cdot R \cdot T \tag{4.103}$$

according to ***Avogadro's Law*** which states that one mole of an ideal gas under constant pressure and temperature exactly occupies the same volume as a mole of any other gas with the same pressure and temperature (every gas under normal conditions, i. e. with $p = 1.023\ bar$ and $T = 273.15\ ^\circ K$, has a volume of $V = 22.4\ liter = .0224\ m^3$). Since the time derivative of the partial volume is needed for the power balance of equs.(4.98,99), equ.(4.103) has to be rewritten as follows:

$$\Phi_i = \frac{\xi_i \cdot R \cdot T}{p} \quad \text{where} \quad \xi_i = \sum_{k_i=1}^{K} \nu_{i,k_i} \cdot \xi_{k_i} \tag{4.104}$$

Application of equ.(4.104) to the hydrogen– bromine reaction yields:

$$\begin{aligned}
\Phi_{Br_2} &= \xi_{Br_2} \cdot \left(\frac{R \cdot T}{p}\right) = (-\xi_{k1} + \xi_{k2} - \xi_{k5}) \cdot \left(\frac{R \cdot T}{p}\right) && (4.105)\\
\Phi_{Br\bullet} &= \xi_{Br\bullet} \cdot \left(\frac{R \cdot T}{p}\right) = (2\xi_{k1} - 2\xi_{k2} - \xi_{k3} + \xi_{k4} + \xi_{k5}) \cdot \left(\frac{R \cdot T}{p}\right) && (4.106)\\
\Phi_{H_2} &= \xi_{H_2} \cdot \left(\frac{R \cdot T}{p}\right) = (-\xi_{k3} + \xi_{k4}) \cdot \left(\frac{R \cdot T}{p}\right) && (4.107)\\
\Phi_{H\bullet} &= \xi_{H\bullet} \cdot \left(\frac{R \cdot T}{p}\right) = (\xi_{k3} - \xi_{k4} - \xi_{k5}) \cdot \left(\frac{R \cdot T}{p}\right) && (4.108)\\
\Phi_{HBr} &= \xi_{HBr} \cdot \left(\frac{R \cdot T}{p}\right) = (\xi_{k3} - \xi_{k4} + \xi_{k5}) \cdot \left(\frac{R \cdot T}{p}\right) && (4.109)
\end{aligned}$$

The total volume flow is the sum of all the component volume flows:

$$\Phi = \sum_{i=1}^{I} \Phi_i = \xi \cdot \left(\frac{R \cdot T}{p}\right) = (\xi_{k_1} - \xi_{k_2}) \cdot \left(\frac{R \cdot T}{p}\right) \tag{4.110}$$

If the volume flow of each reaction is desired rather than of the individual compounds, then the above equation is usefully split into:

$$\Phi_{k_i} = \xi_{e_{k_i}} \cdot \left(\frac{R \cdot T}{p}\right) \quad \text{where} \quad \xi_{e_{k_i}} = \sum_{i=1}^{I} \nu_{i,k_i} \cdot \xi_{k_i} \tag{4.111}$$

and applied to the hydrogen–bromine reaction:

$$\Phi_{k1} = (2-1) \cdot \xi_{k1} \cdot \frac{R \cdot T}{p} \tag{4.112}$$

$$\Phi_{k2} = (1-2) \cdot \xi_{k2} \cdot \frac{R \cdot T}{p} \tag{4.113}$$

$$\Phi_{k3} = (2-2) \cdot \xi_{k3} \cdot \frac{R \cdot T}{p} \tag{4.114}$$

$$\Phi_{k4} = (2-2) \cdot \xi_{k4} \cdot \frac{R \cdot T}{p} \tag{4.115}$$

$$\Phi_{k5} = (2-2) \cdot \xi_{k5} \cdot \frac{R \cdot T}{p} \tag{4.116}$$

Note that the numbers between the parentheses denote the difference of the number of gaseous product particles and the number of gaseous reactant particles according to the definition of the stoichiometric coefficient.

In the third, fourth and fifth reaction, two gaseous reactants are traded for two other gaseous products. Consequently, the volume does not change, in full accordance with Avogadro's Law, and thus, hydraulic/pneumatic power is neither produced nor absorbed. This is contrary to the first and second reactions, where one mole of bromine–molecules Br_2 is split into two moles of bromide-radicals $Br^\bullet$ or generated by two moles of $Br^\bullet$, respectively. As a result, the first two reactions produce hydraulic/pneumatic power ($p \cdot \Phi_{k1}$ and $p \cdot \Phi_{k2}$). The sum of the stoichiometric coefficients multiplied by the reaction flow rate ξ_{k_i} is called the *effective flow rate* $\xi_{e_{k_i}}$ of each reaction.

4.3.4 The Initial Conditions

The two power balances, the reaction kinetics and the derived modified equation of state for ideal gases are without exception of dynamic type. I.e., they depend on time and therefore they have to be integrated over time to solve this highly nonlinear problem. Thus, the initial conditions for the dynamic equations have to be evaluated. This process is covered in the next two sections.

4.3.4.1 Initial Conditions of the Kinetics and the Equation of State

There is usually no problem to get the initial conditions of the reaction kinetics and the equation of state. The reactants' number of moles n_{0i} are normally measured before the reaction takes place and, thus the set of initial conditions of the reaction kinetics are directly given. The equation of state requires either the temperature T_0, the pressure p_0, or the volume V_0 as initial values depending on how the dynamic variables have been chosen.

The two initial values of the power balance equations have to be computed from thermodynamic data which is tabulated in appropriate literature [47–54]. This calculation is to be the subject of the following section.

4.3.4.2 Initial Conditions of the Two Power Balance Equations

In any case it is necessary to determine the initial value of the chemical potential μ_{0i} from equ.(4.86) shown below under isobaric and isothermic conditions:

$$U_i = -p \cdot V_i + T \cdot S_i + \mu_i \cdot n_i \tag{4.117}$$

There exist various equations of state for all kinds of properties of the matter, which can be found in the literature [40,48].

As an example of another form of the equation of state, liquids or simple solids are to be looked at. Assuming fluids and simple solids to be nearly incompressible, their volume boils down to be directly proportional to the number of moles.

$$V \propto n$$

Actually, the intention of this project was to derive dynamic equations in order to simulate chemical reactions far away from their steady-state. But this plan fails with the application of the modified equation of state since any equation of state is in some way empirical and only valid in the vicinity of equilibrium. Unfortunately, chemical thermodynamics does not provide such an equation of state that can correctly be used in dynamic simulation.

4.3.3.2 The Application of the Equation of State to Ideal Gases

The equation of state is normally applied to compute one of the hydraulic/pneumatic variables p or V. It would be useful if a value of the pneumatic/ hydraulic energy $p \cdot V$ of any gaseous compound i in a ideal mixture of ideal gases were found. For this, two different cases have to be considered:

In the first case, the pressure of each compound in an isochoric and isothermic reaction is to be evaluated depending on the constant temperature and volume and its number of moles. Therefore, the ***partial gas pressure*** p_i will be introduced as well as ***Dalton's law***. According to this law, the sum of all partial pressures in an ideal gas is identically equal to the partial pressures of any compound of the gas mixture. Equ.(4.100) can be rewritten as:

$$\left(\sum_{i=1}^{I} p_i\right) \cdot V = \left(\sum_{i=1}^{I} n_i\right) \cdot R \cdot T \tag{4.101}$$

Rearranging terms, it follows:

$$H_i = U_i + p \cdot V_i = T \cdot S_i + \mu_i \cdot n_i \tag{4.118}$$

If the enthalpy H_i and the entropy S_i were known, it would be very easy to compute the initial values of μ_i and U_i.

The Heat of Formation The origin of the ***heat of formation*** will be determined by means of a carbon with oxygen to carbondioxide reaction:

$$C + O_2 \xrightarrow{k_1} CO_2 \tag{4.119}$$

If this reaction occurs stiochiometrically, i.e., if, for example, the reactants are completely converted into carbondioxide, then the measured heat energy is the same as the ***reaction enthalpy*** ΔH_{k_i} when ***mixture enthalpies*** ΔH_m (being zero for ideal compositions) are neglected. This means that the reaction substances will not influence each other neither micro– nor macroscopically.

$$\Delta Q = \Delta H_{k_i} + \Delta H_m \quad \text{where} \quad H_m = 0.0 \tag{4.120}$$

This relation can also easily be derived from the first law of thermodynamics by neglecting any kinetic, potential or mechanical work.

Since the above observed reaction, which is considered to be completely stoichiometrically converted, occurs under normal conditions (i.e. at room temperature $T^* = 298.15° \, K$ and the pressure $p^* = 1.0132 \, bar$), the heat energy measured is called te ***standard heat of formation*** of the molecule CO_2. The heat of formation is normalized to this ***standard molar heat of formation*** of the pure substance:

$$h_i(p^*, T^*) = \frac{H_(p^*, T^*)}{n_i} = \Delta h^*_{k_i} \tag{4.121}$$

for which data is tabulated in most of the common literature [51,56,57]. By convention, the standard heat of formation of pure elements, for example of hydrogen H_2 and carbon C, is normalized to zero.

For more complex reactions under certain conditions (p, T)

$$\nu_1^R \cdot A + \nu_2^R \cdot B \xrightarrow{k_1} \nu_3^P \cdot C + \nu_4^P \cdot D \tag{4.122}$$

the *molar reaction enthalpy* is found as follows:

$$\begin{aligned}\Delta h(p,T) &= \nu_3^P \cdot h_C(p,T) + \nu_4^P \cdot h_D(p,T) - \nu_1^R \cdot h_A(p,T) - \nu_2^R \cdot h_B(p,T) \\ &= \sum_{i=1}^{I} \nu_i \cdot h_i(p,T) \end{aligned} \tag{4.123}$$

where ν_i represents the stoichiometric coefficient and $h_i(p,T)$ denotes the *molar heat of formation* which can be written as the differential of the intensive variables p and T and the molar standard heat of formation h_i^* as following:

$$h_i(p,T) = \underbrace{h_i(p^*,T^*)}_{a} + \underbrace{\int_{T^*}^{T} \left.\frac{\partial h_i(p^*,T)}{\partial T}\right|_p dT}_{b} + \underbrace{\int_{p^*}^{p} \left.\frac{\partial h_i(p,T^*)}{\partial p}\right|_T dp}_{c} \tag{4.124}$$

The term a of equ.(4. 124) is as said the molar standard heat of formation of the pure substance i. This value is usually found in existing tables in literature, for example in Perry and Chilton [46].

The integrand of the *temperature correction term* b can be identified as the *specific heat capacity* c_{p_i}:

$$c_{p_i}(T,p) = \left.\frac{\partial h_i}{\partial T}\right|_p \tag{4.125}$$

The specific heat capacity under constant pressure is equal to the specific heat capacity under constant volume for ideal gases, and also equals the partial derivative of the molar enthalpy h_i of the pure compound i with respect to temperature. For ideal gases in the vicinity of atmospheric pressure, the specific heat capacity can be approximated by a polynomial of the temperature variable:

$$c_{p_i}(T,p) = \alpha_i + \beta_i \cdot T + \gamma_i \cdot T^2 (+\delta_i \cdot T^3) \tag{4.126}$$

where the coefficients α_i, β_i and γ_i are also tabulated in the aforementioned literature. Equ.(4. 126) should only be used for the gaseous phase. It will not be valid any more if the phase changes from the gaseous phase to the liquid one since the value of the molar heat capacity will jump from a low value to a very high one.

Under the assumption of an ideal gas, the *pressure correction term c* can be neglected. Thermodynamical data for this correction term can also be looked up in the appropriate literature for real gases.

In summary, the heat of formation of an ideal gas is:

$$\begin{aligned} h_i(p,T) &= h_i(p^*,T^*) + \int_{T^*}^{T} c_{p_i}(p,T)dT \qquad (4.127)\\ &= h_i(p^*,T^*) + \alpha_i \cdot (T - T^*) + \frac{\beta_i}{2} \cdot (T^2 - T^{*2}) + \frac{\gamma_i}{3} \cdot (T^3 - T^{*3}) \end{aligned}$$

Since the objective of this section is to compute the initial chemical potential with the entropy besides the heat of formation, the entropy has also to be determined. It turns out that it can be evaluated by the same method applied to the heat of formation. This is because both the enthalpy and the entropy are state functions.

$$s_i(p,T) = \underbrace{s_i(p^*,T^*)}_{a} + \underbrace{\int_{T^*}^{T} \left.\frac{\partial s_i(p^*,T)}{\partial T}\right|_p dT}_{b} + \underbrace{\int_{p^*}^{p} \left.\frac{\partial s_i(p,T^*)}{\partial p}\right|_T dp}_{c} \qquad (4.128)$$

Term *a* can be normalized to zero, and thus can be cancelled out if the initial temperature is chosen to be the standard room temperature; otherwise its value must be found from tabulated data in the literature.

The integrand of the term *b* depends on the specific heat capacity as well as the heat of formation, but now, by definition of the entropy, divided by the temperature:

$$\frac{c_{p_i}}{T} = \left.\frac{\partial s_i(p^*,T)}{\partial T}\right|_p \qquad (4.129)$$

This is valid for systems of one–phase ideal gases.

Unfortunately, term c cannot be eliminated as in the case of the heat of formation. For this, one has to consider one of the *Maxwell equations*:

$$\frac{\partial S}{\partial p} = -\frac{\partial V}{\partial T} \tag{4.130}$$

Taking equ.(4.130) and the equation of state for ideal gases it follows that:

$$\int_{p^*}^{p} \left.\frac{\partial S_i(p)}{\partial p}\right|_T dp = -\int_{p^*}^{p} \left.\frac{\partial V_i(p)}{\partial T}\right|_p dp = -n_i \cdot R \int_{p^*}^{p} \frac{1}{p} dp = -n_i \cdot R \cdot \ln\left(\frac{p}{p^*}\right) \tag{4.131}$$

Normalizing the pressure correction term yields:

$$\Delta s_i(p, T^*) = -x_i \cdot R \cdot \ln\left(\frac{p}{p^*}\right) \tag{4.132}$$

In the literature, data is only listed for the first two terms of the molar entropy. These two terms are abbreviated as follows:

$$s_i^t(p^*, T) = s_i(p^*, T^*) + \int_{T^*}^{T} \left.\frac{\partial s_i(p^*, T)}{\partial T}\right|_p dT \tag{4.133}$$

where t stands for *"tabulated"*, and with this abbreviation the molar entropy can be rewritten as:

$$s_i(p, T) = s_i^t(p^*, T) - x_i \cdot R \cdot \ln\left(\frac{p}{p^*}\right) \tag{4.134}$$

With the free Gibbs energy of a pure compound i, equ.(4.88) can be written as:

$$G_i = H_i - T \cdot S_i \tag{4.135}$$

The value of the free Gibbs energy exactly represents the chemical energy stored in the compound i. By normalizing equ.(4.133) it follows:

$$g_i = h_i - T \cdot s_i \tag{4.136}$$

Interestingly, the molar Gibbs free energy is just another name of the chemical potential μ_i. The chemical potential can hence be expressed as follows:

$$\mu_i(p, T) = g_i(p, T) = h_i - T \cdot s_i = h_i - T \cdot s_i^t + x_i \cdot \ln\left(\frac{p}{p^*}\right) \tag{4.137}$$

With this equation it is possible to compute the initial value of the chemical potential. The second power balance equation under isothermic and isobaric conditions is not violated by equ.(4. 137) since the sum over all the μ_i is again a constant:

$$\sum_{i=1}^{I} x_i = 1 \tag{4.138}$$

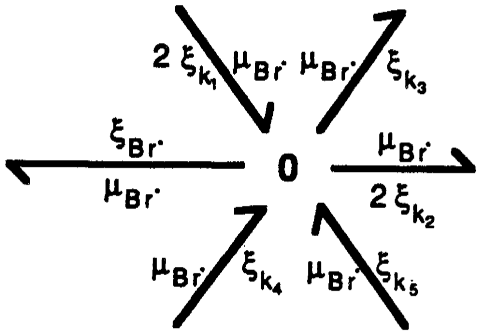
Note that the decoupling of the kinetics and the thermodynamics can only be assumed if the chemical is considered to be isolated from the surrounding systems. But as soon as the chemical system is attached to the thermodynamic or hydraulic/pneumatic environment the decoupling effect will be eliminated and thus, the chemical reaction kinetics will depend on the variation of temperature as mentioned.

4.4 Chemical Reaction Bond Graphs

In the previous sections, all the definitions have been derived because they serve as a basis for the Bond graphs applied to chemical reaction kinetics and chemical thermodynamics. For this purpose, two basic examples of the previously introduced hydrogen–bromine reaction are to be shown in the Bond graph model, and in their implementation in the modeling language DYMOLA. The hydrogen–bromine reaction will be considered under isothermic and isobaric, and isothermic and isochoric conditions.

4.4.1 Basic Elements in Chemical Bond Graphs

According to the general definition of the *0–junction* the flow variables have to add up to zero. In the case of chemical systems, the flow variables are represented by



$$\xi_{Br^\bullet} = 2\xi_{k_1} - 2\xi_{k_2} - \xi_{k_3} + \xi_{k_4} + \xi_{k_5} \tag{4.139}$$

Figure 4.1: The mass balance of the bromine radical $Br^\bullet$

the molar flow rates. Since the mass balance has to be satisfied, the molar change ξ_i of one pure substance of a reaction is balanced by the reaction flow rates ξ_{k_i} of those reactions in which this pure substance is involved.

As an example, the change of moles of the bromine radical $\xi_{Br^\bullet}$ depends on all five reactions illustrated in Fig.4.1 including the bond graph and its appropriate equation of the mass balance. The direction of the bonds follows directly from the pure substances being either a product (then the arrow will point towards the 0–junction), or a reactant (then the arrow will point away from the 0–junction).

Since in the first and second reaction two moles of the bromine radical $Br^\bullet$ are generated or absorbed, respectively, it should be clear to multiply these reactions by two. In general, the mass balance can be written as:

$$\xi = \sum_{i=1}^{I} \nu_{i,k_i} \cdot \xi_{k_i} \tag{4.140}$$

where ν_{i,k_i} represents the stoichiometric coefficient as usual.

The mass balance at 0-junctions perfectly satisfies the power balance which has to be valid at the 0-junction as well as in any point of the Bond graph model.

$$\mu_{Br^\bullet} \cdot \xi_{Br^\bullet} = \mu_{Br^\bullet} \cdot 2\xi k_1 - \mu_{Br^\bullet} \cdot 2\xi_{k_2} - \mu_{Br^\bullet} \cdot \xi_{k_3} + \mu_{Br^\bullet} \cdot \xi_{k_4} + \mu_{Br^\bullet} \cdot \xi_{k_5} \quad (4.141)$$

The other type of junction, the *1-junction*, symbolizes one chemical reaction. Thus, its reaction flow rate ξ_{k_i} has to be constant while the chemical potentials representing the ***across variables*** add up to zero. The general formula of this balance of the chemical energy boils down to:

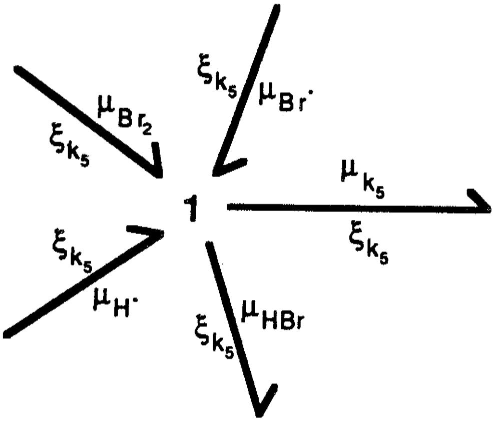
$$\mu_{k_i} = \sum_{i=1}^{I} \nu_{i,k_i} \cdot \mu_i \quad (4.142)$$

and its approriate power balance:

$$\mu_{k_i} \cdot \xi_{k_i} = \left(\sum_{i=1}^{I} \nu_{i,k_i} \cdot \mu_i \right) \cdot \xi_{k_i} \quad (4.143)$$

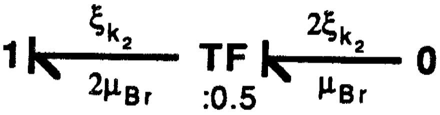
However, the chemical potential of a reaction evaluated at the 1–junction is nothing but the balance of chemical power into or out of the reaction with other types of power such as the thermodynamic and the hydraulic/ pneumatic power. An example of the equ.(4. 142) or equ.(4. 143) is demonstrated in Fig.4.2, representing the third reaction.

The transformer is introduced in the Bond graph model in order to take the stoichiometric coefficients into account. The transformer necessary in the second reaction serves as an example: With the mass of the bromine radical $Br^\bullet$ balanced at the 0–junction (which is shown in Fig. 4.1), there is yet one outflux of power, being the product of the reaction rate $2\xi_{k_2}$ and the common effort variable $\mu_{Br^\bullet}$ of the 0–junction. This power has to be converted into a differently composed power since it flows with a reaction rate ξ_{k_2} into the second reaction at the 1–junction. However, since the power balance always has to be preserved, the chemical potential going along with the power flux into the 1–junction must have a value of $2\mu_{Br^\bullet}$. This convertion of power is shown in Fig. 4.3.



$$\mu_{k_5} = \mu_{Br_2} + \mu_{Br^\bullet} + \mu_{H^\bullet} - \mu_{HBr} \tag{4.144}$$

Figure 4.2: The chemical power balance at the 1–junction



$$\mu_{Br^\bullet} \cdot (2\xi_{k_2}) = (2\mu_{Br^\bullet}) \cdot \xi_{k_2} \tag{4.145}$$

Figure 4.3: The convertion of power at the transformer

```
model type TF
   cut A(mu1/xi1), B(mu2/ - xi2)
   main cut C[A B]
   main path P<A-B>
   parameter k = 1

   k * xi2 = xi1
   k * mu1 = mu2
end
```

Figure 4.4: The DYMOLA–program of the transformer

The knowledge of the Bond graphs, being a unified methodology for modeling complex physical systems enables the reuse of the same transformer as the one implemented for electrical systems in DYMOLA in Chapter 3.

The mass of any compound of the reaction is accumulated in the *CS–element*, a new storage element used to treat chemical systems conveniently. This element works like a ***capacitive source*** and is attached to all the 0–junctions of the chemical systems since the mass flow, which is balanced in the 0–junction, is either accumulated in it or is absorbed from it. In addition, the CS–element is used to compute the chemical potential μ_i of any pure substance by using the initial condition, the molar free energy μ in equ. (4. 137). The initial condition will be evaluated in the main program in DYMOLA and then transmitted by the "submodel–call" into the CS–model, as will be explained in more detail at hand of two examples later in this Section((4. 4. 2)and (4. 4. 3)).

At this point of the report it is appropriate to leave the introduction into the general Bond graph elements of chemical reactions but to consider special variations of the two power balances. Under certain thermodynamical conditions, for instance, the isochoric–isothermic case, the two power balance equations are simplified but different for every case. Therefore, new Bond graph elements are developed for differing input–cases and even for one specific input–case. For example, the CS–element which is to compute the chemical potential by using the second power balance equation cannot be generic both in the case of isobaric–isothermic and of isochoric–isothermic conditions, whose treatment is to be the subject of the rest of this chapter.

4.4.2 The Isobaric–Isothermic Case

Under the assumption of isobaric–isothermic, conditions equs.(4. 98) and (4. 99) lead to:

$$p \cdot \dot{V} = T \cdot \dot{S} + \sum_{i=1}^{I} \mu_i \cdot \xi_i \tag{4.146}$$

$$0.0 = \sum_{i=1}^{I} \dot{\mu}_i \cdot n_i \tag{4.147}$$

The pressure p and the temperature T are given as inputs in equ.(4. 146), the chemical potential μ_i is calculated in the *CS*–element as mentioned above, and the reaction flow rate ξ_i is determined by the kinetic equations. Thus, there are only the volume flow $\dot{V}$ and the entropy flow $\dot{S}$ left to be computed from equ.(4. 146). For this, another new element, the so–called *chemical reactor ChR* is introduced which is attached at each 1–junction. This chemical reactor performs the conversion of the chemical power balanced in the 1–junction into other forms of energy, namely thermic and hydraulic/pneumatic energy. Thus, the chemical reactor is nothing but an ideal energy transducer converting the power at its ports and neither storing power nor dissipating power.

It proved to be convenient to use the first power balance in the *ChR*–element to compute the entropy flow. Since the chemical potential μ_{k_i} of the reaction is balanced at the 1–junction, and thus, serves as an input of the chemical reactor and hence, as input of the first power balance equation, it is necessary to express the whole power balance in terms of the partial volume flow $\dot{V}_{k_i}$, the partial entropy flow $\dot{S}_{k_i}$, and the molar flow rate ξ_{k_i} of component reactions. For this, the term of the chemical power in the power balance has to be rewritten in terms of component reactions.

Before showing this conversion, the mass balance expressed by the molar flow rate ξ_i of the pure substance i and the molar flow rate ξ_{k_i} of the component reactions in matrix notation will be shown using equ.(4. 140):

$$\xi_i = \sum_{k_i=1}^{K} \nu_{i,k_i} \cdot \xi_{k_i} \tag{4.148}$$

or

$$\begin{pmatrix} \xi_{Br_2} \\ \xi_{Br\bullet} \\ \xi_{H_2} \\ \xi_{H\bullet} \\ \xi_{HBr} \end{pmatrix} = \begin{pmatrix} -1 & 1 & 0 & 0 & -1 \\ 2 & -2 & -1 & 1 & 1 \\ 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 1 & -1 & -1 \\ 0 & 0 & 1 & -1 & 1 \end{pmatrix} \cdot \begin{pmatrix} \xi_{k_1} \\ \xi_{k_2} \\ \xi_{k_3} \\ \xi_{k_4} \\ \xi_{k_5} \end{pmatrix} \tag{4.149}$$

or

$$\vec{\xi_s} = \mathbf{N} \cdot \vec{\xi_k} \tag{4.150}$$

Note that the computation of the molar flow rate ξ_i of the pure compound i from the molar flow rate ξ_{k_i} of the component reaction is possible but not vice versa since the matrix N is singular. Dr. François Cellier explored an interesting symmetry between this mass balance and the power balance at the 1–junction represented by the chemical potentials known by equ.(4. 142):

$$\mu_{k_i} = \sum_{i1}^{I} \nu_{i,k_i} \cdot \mu_i \tag{4.151}$$

and in matrix notation:

$$\begin{pmatrix} \mu_{k_1} \\ \mu_{k_2} \\ \mu_{k_3} \\ \mu_{k_4} \\ \mu_{k_5} \end{pmatrix} = \begin{pmatrix} -1 & 2 & 0 & 0 & 0 \\ 1 & -2 & 0 & 0 & 0 \\ 0 & -1 & -1 & 1 & 1 \\ 0 & 1 & 1 & -1 & -1 \\ -1 & 1 & 0 & -1 & 1 \end{pmatrix} \cdot \begin{pmatrix} \mu_{Br_2} \\ \mu_{Br^\bullet} \\ \mu_{H_2} \\ \mu_{H^\bullet} \\ \mu_{HBr} \end{pmatrix} \tag{4.152}$$

or in the more compact way:

$$\vec{\mu}_k = \mathbf{M} \cdot \vec{\mu}_s \tag{4.153}$$

where obviously:

$$\mathbf{N} = \mathbf{M}^T \tag{4.154}$$

Since the matrix **M** is the transpose of the matrix **N**, it is straightforward that the matrix **M** is singular as well, and thus, the chemical potentials μ_i of the pure compounds i cannot be computed from the chemical potential μ_{k_i} of the component reactions k_i.

The proof for the convertion of chemical power in terms of the pure substances i into the chemical power in terms of the component reactions k_i is as follows:

$$\begin{aligned} \sum_{i=1}^{I} (\mu_i \cdot \xi_i) &= \mu_{Br_2}\xi_{Br_2} + \mu_{Br^\bullet}\xi_{Br^\bullet} + \mu_{H_2}\xi_{H_2} + \mu_{H^\bullet}\xi_{H^\bullet} + \mu_{HBr}\xi_{HBr} \\ &= \mu_{Br_2}(-\xi_{k_1} + \xi_{k_2} - \xi_{k_5}) \\ &\quad + \mu_{Br^\bullet} \cdot (2\xi_{k1} - 2\xi_{k_2} - \xi_{k_3} + \xi_{k_4} + \xi_{k_5}) \\ &\quad + \mu_{H_2} \cdot (-\xi_{k_3} + \xi_{k_4}) \\ &\quad + \mu_{H^\bullet} \cdot (\xi_{k_3} - \xi_{k_4} - \xi_{k_5}) \\ &\quad + \mu_{HBr} \cdot (\xi_{k_3} - \xi_{k_4} + \xi_{k_5}) \end{aligned}$$

$$
\begin{aligned}
&= (-\mu_{Br_2} + 2\mu_{Br^{\bullet}}) \cdot \xi_{k_1} \\
&\quad + (\mu_{Br_2} - 2\mu_{Br^{\bullet}}) \cdot \xi_{k_2} \\
&\quad + (-\mu_{Br^{\bullet}} - \mu_{H_2} + \mu_{H^{\bullet}} + \mu_{HBr}) \cdot \xi_{k_3} \\
&\quad + (\mu_{Br^{b}ullet} + \mu_{H_2} - \mu_{H^{\bullet}} - \mu_{HBr}) \cdot \xi_{k_4} \\
&\quad + (-\mu_{Br_2} + \mu_{Br^{\bullet}} - \mu_{H^{\bullet}} + \mu_{HBr}) \cdot \xi_{k_5} \\
&= \mu_{k1}\xi_{k_1} + \mu_{k2}\xi_{k_2} + \mu_{k3}\xi_{k_3} + \mu_{k4}\xi_{k_4} + \mu_{k5}\xi_{k_5} \\
&= \sum_{i=1}^{I} (\mu_{k_i} \cdot \xi_{k_i}) \qquad (4.155)
\end{aligned}
$$

This proof can be written more compactly as:

$$\vec{\mu}_s^T \cdot \vec{\xi}_s = \vec{\mu}_s^T \cdot \left(\mathrm{N} \cdot \vec{\xi}_k\right) = \vec{\mu}_s^T \cdot \left(\mathrm{M}^T \cdot \vec{\xi}_k\right) = \left(\vec{\mu}_s^T \cdot \mathrm{M}^T\right) \cdot \vec{\xi}_k = (\mathrm{M} \cdot \vec{\mu}_s)^T \cdot \vec{\xi}_k = \vec{\mu}_k^T \cdot \vec{\xi}_k \qquad (4.156)$$

The power balance (equ.(4. 146)) can be either decomposed in terms of reaction compounds k_i:

$$p \cdot \sum_{i=1}^{I} \Phi_i = T \cdot \sum_{i=1}^{I} \dot{S}_i + \sum_{i=1}^{I} \mu_i \cdot \xi_i \qquad (4.157)$$

or in terms of substance compounds i:

$$p \cdot \sum_{k=1}^{K} \Phi_{k_i} = T \cdot \sum_{k=1}^{K} \dot{S}_{k_i} + \sum_{k=1}^{K} \mu_{k_i} \cdot \xi_{k_i} \qquad (4.158)$$

Considering the power balance at a single chemical reactor *ChR*, equ.(4. 158) boils down to:

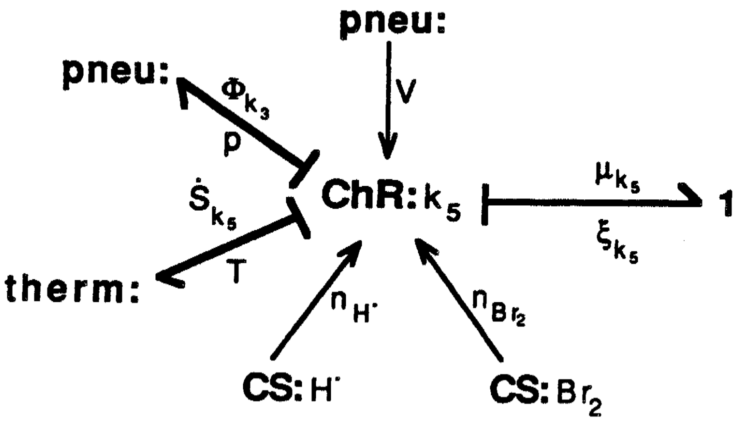
$$p \cdot \Phi_{k_i} = T \cdot \dot{S}_{k_i} + \mu_{k_i} \cdot \xi_{k_i} \qquad (4.159)$$

Nature tries to minimize its energy as well as in chemical reactions, for example, where the co–energy G is to reach minimal energy. Since the sum $\dot{G}$ of the co–energy's derivative equals zero, and since it does not make much physical sense if the single co–powers $\dot{G}_i$ of the component substances change in time — why should nature change any quantity if this does not contribute to energy minimization ? —, the time–derivative of the chemical potential $\dot{\mu}_i$ can be set to zero. Thus, the chemical potentials μ_i stay constant over time and are evaluated from equ.(4. 137).

4.4.2.1 The Bond Graph Elements

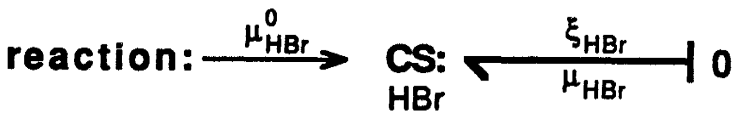
The partial volume flow Φ_{k_i} of equ.(4.159) in the chemical reactor *ChR* has to be computed from the dynamic version of the equation of state for ideal gases (4.111) which is only valid in the isobaric–isothermic case. The chemical reactor *ChR* also includes the kinetic equations (4.68–72) that evaluate the molar reaction flow rates ξ_{k_i}. Fig.4.5 shows the chemical reactor *ChR* of the fifth reaction k_5 in which the signal paths coming from the CS–Br_2 and $H^{\bullet}$–elements (explained later) indicate the modulation of the number of moles n_{Br_2} and $n_{H^{\bullet}}$, respectively. These numbers of moles have to be known for the kinetic equations and are declared as *terminals* in the DYMOLA *model type* of the chemical reactor *ChR*. The modulation signal of the total volume V coming from the *Pneumatic*–element (later explained), is needed for the first power balance (equ.(4.159)) and is defined as *terminal*–variable as well. In the case of the fifth equation, the effective molar flow rate ξ_{ek_5} will be zero. Cuts in a model represent an interface to outer modules like the *chem*–, the *pneuk5*–, and the *thermk5*–cuts of the chemical reactor *ChR*–element with the approriate cuts in the *CS*–, the *Thermic*–, and the *Pneumatic*–elements, respectively. Due to the fact that the chemical reactor *Chr* of each reaction has different modulation signals it was necessary to create one such model for each reaction in the system.

The *CS*–model is sufficiently simple to be described as an atomic element which can be reused for all *CS*–elements. Under isobaric–isothermic conditions, the chemical potential μ_i of any substance i is constant, and thus, the chemical potential μ_i equals the molar Gibbs free energy μ_{i0} evaluated from equ.(4.137) in the main program. The CS_i–elements compute the numbers of moles of any substance i from the molar flow rates ξ_i which are evaluated at the 0–junctions of the Bond graph. Fig.4.6 depicts the generic *CS*–element.



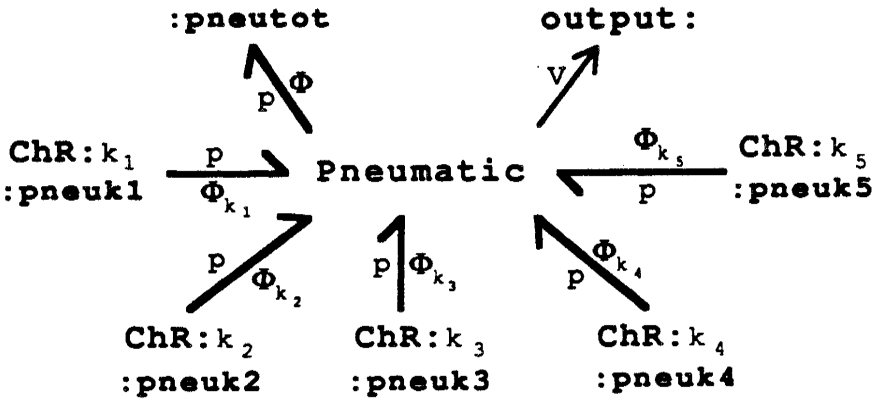
```
model type ChRk5
   cut chem(muk5/ - xik5), thermk5(Temp/ - Sdotk5)
   cut pneumk5(p/Phik5)
   main cut C [chem]
   terminal nBr2, nH, V
   parameter R = 8.314
   local k5, xiek5

   k5 = 10.0**11.97 * exp(-149800.0/(R * Temp))
   0.0 = Temp * Sdotk5 + muk5 * xik5k
   xiek5 = 0.0
   p * Phik5 = T * Sdotk5 + muk5 * xik5
   p * Phik5 = xiek5 * R * Temp
   xik5 = k5 * nBr2 * nH/V
end
```

Figure 4.5: The ChR–element in a Bond graph and coded in DYMOLA



```
model type CS
   main cut chem(mu/xi)
   terminal n, mu0

   mu = mu0
   der(n) = xi
end
```

Figure 4.6: The generic CS–element in a Bond graph and coded in DYMOLA



```
model type Pneumatic
   main cut pneutot(p/Phi)
   cut pneuk1(p/-Phik1), pneuk2(p/-Phik2)
   cut pneuk3(p/-Phik3), pneuk4(p/-Phik4)
   cut pneuk5(p/-Phik5)
   terminal V

   Phi = Phik1 + Phik2 + Phik3 + Phik4 + Phik5
   der(V) = Phi
end
```

Figure 4.7: The *Pneumatic*–element in a Bond graph and coded in DYMOLA

The *Pneumatic*–element adds up all the partial volume flows Φ_{k_i} that are computed in the *ChR*–elements, and integrates the total flow to the total volume V, on the other hand, needed as modulation signal of the *ChR*–elements. This element is shown in Fig.4.7.

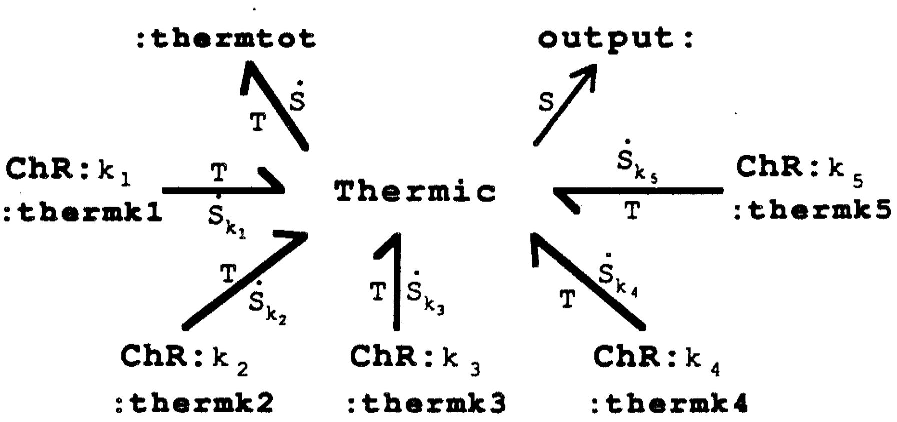
The *Thermic*–element is needed if one wants to know the total entropy — the integral of the sum of partial entropies $\dot{S}_{k_i}$ — S which, actually, is not used in any of the model equations to compute the kinetics and thermodynamics. But if the internal energy U is to be computed to validate the model as shown later, the total entropy has to be known. Fig.4.8 shows the *Thermic*–element.

Since the isochoric–isothermic case will prove to be the more general case than the isobaric–isothermic case, the *main program* is not shown for the isobaric–isothermic case, but it is referred to the next section where the structure of the main program of the isochoric–isothermic case will be explained in detail, or to Appendix B. Fig.4.9 shows the Bond graph model of the hydrogen–bromine reaction under isobaric–isothermic conditions. The forth reaction has been left out of this graph to keep the graph planar. The signal paths are also not drawn in order to enhance the clarity of the graph. The main program's generated DESIRE program is given in Appendix B as well.

The simulation run of the DESIRE program generates the graph of the number of moles n_i changing over time which is shown in Fig.4.10.

4.4.3 The Isochoric–Isothermic Case

Due to the complexity of the modulation of chemical reactions — the number of moles in the reaction rate is variable! — it was necessary to come up with some Bond graph elements that describe the isochoric, isothermic case separately. But



```
model type Thermic
   main cut thermtot(T/ – Sdot)
   cut thermk1(T/Sdotk1), thermk2(T/-Sdotk2)
   cut thermk3(T/Sdotk3), thermk3(T/-Sdotk4)
   cut thermk5(T/Sdotk5)
   terminal S

   Sdot = Sdotk1 + Sdotk2 + Sdotk3 + Sdotk4 + Sdotk5
   der(S) = Sdot
end
```

Figure 4.8: The *Thermic*–element in a Bond graph and coded in DYMOLA

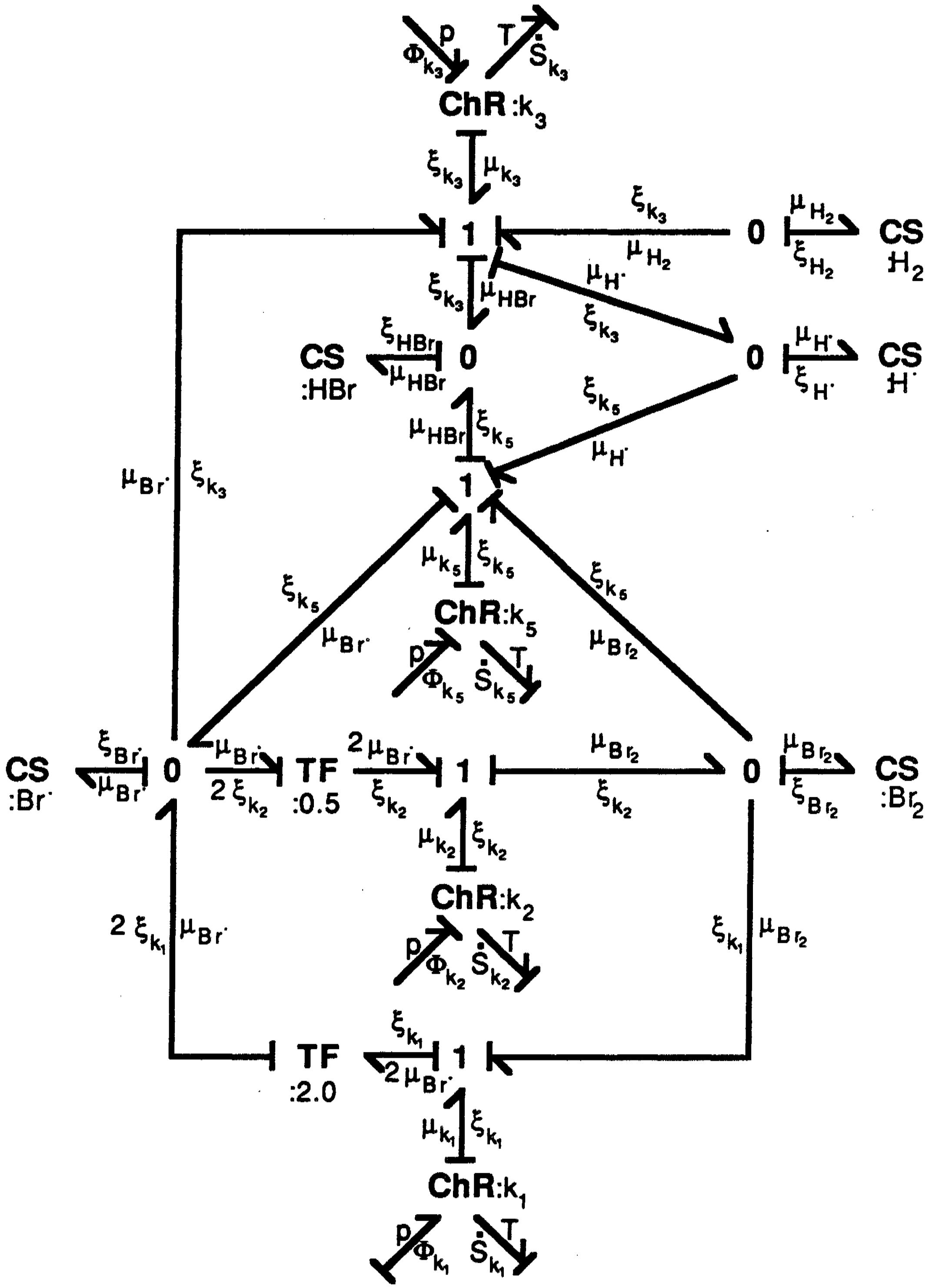


Figure 4.9: Bond graph of the hydrogen–bromine reaction, p=const., T=const.

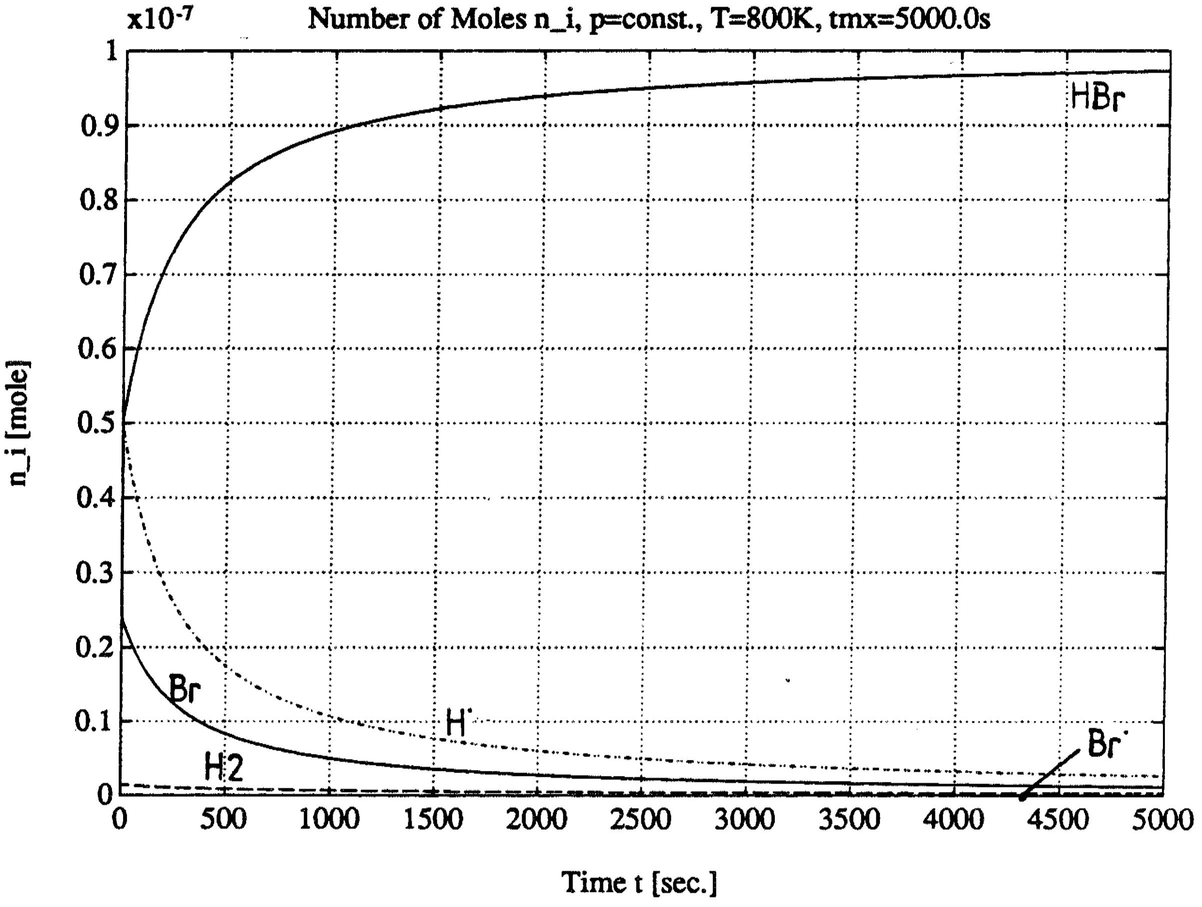


Figure 4.10: The Number of moles changing in time for $p = const., T = const.$

before doing this, the two power balances are to be considered:

$$0.0 = T \cdot \dot{S} + \sum_{i=1}^{I} \mu_i \cdot \xi_i \tag{4.160}$$

$$\dot{p} \cdot V = \sum_{i=1}^{I} \dot{\mu}_i \cdot n_i \tag{4.161}$$

As known from the isobaric–isothermic case, equ.(4.160) can be expressed in terms of component reactions k_i. Thus, equ.(4.160) is used in the *chemical reactor*. On the other hand, this second power balance reveals that the chemical potential has to be evaluated dynamically. The question of whether or not this equation can be decomposed into component reactions rather than component substances is answered in the proof below:

The time derivative of the chemical potentials decomposed into component reactions depend as shown in the previous section on those which are decomposed into component substances:

$$\vec{\dot{\mu}}_k = \mathbf{M} \cdot \vec{\dot{\mu}}_S \tag{4.162}$$

while the number of moles of any pure substance i is determined by integrating the mass balance with the initial condition $\vec{n}_0$:

$$\vec{n}_S = \mathbf{N} \cdot \vec{n}_k + \vec{n}_0 \tag{4.163}$$

However, the matrix N does not have full rank and in the more general case of complex reactions, is not even square, but is of dimension $K \times I$, where K denotes the number of chemical reactions and I represents of component gases. Thus, N cannot be inverted to receive the number of moles of component reactions from equ.(4.163). There is only one other chance left to assign the causality of equ.(4.163) in the number of moles of component reactions; that is to normalize all the initial conditions of $\vec{n}_k$ to zero, and to set the constant vector $\vec{n}_0$ equal to the vector $\vec{n}_0$ of the given initial conditions:

$$\vec{n}_0 \equiv \vec{n}_S(t = 0.0) \quad , \quad \vec{n}_k \equiv 0.0 \tag{4.164}$$

The time derivative of the free Gibbs energy now becomes:

$$\begin{aligned}
\dot{G} &= \vec{\mu}_S^T \cdot \vec{\xi}_S + \dot{\vec{\mu}}_S^T \cdot \vec{n}_S \\
&= \vec{\mu}_S^T \cdot \left(\mathbf{N} \cdot \vec{\xi}_k\right) + \dot{\vec{\mu}}_S^T \cdot (\mathbf{N} \cdot \vec{n}_k + \vec{n}_0) \\
&= \left(\vec{\mu}_S^T \cdot \mathbf{M}\right) \cdot \vec{\xi}_k + \left(\dot{\vec{\mu}}_S^T \cdot \mathbf{M}\right) \cdot \vec{n}_k + \dot{\vec{\mu}}_S^T \cdot \vec{n}_0 \\
&= \vec{\mu}_k^T \cdot \vec{\xi}_k + \dot{\vec{\mu}}_k^T \cdot \vec{n}_k + \dot{\vec{\mu}}_S^T \cdot \vec{n}_0
\end{aligned}$$

and therefore:

$$\dot{\vec{\mu}}_S^T \cdot \vec{n}_S \neq \dot{\vec{\mu}}_k^T \cdot \vec{n}_k \tag{4.165}$$

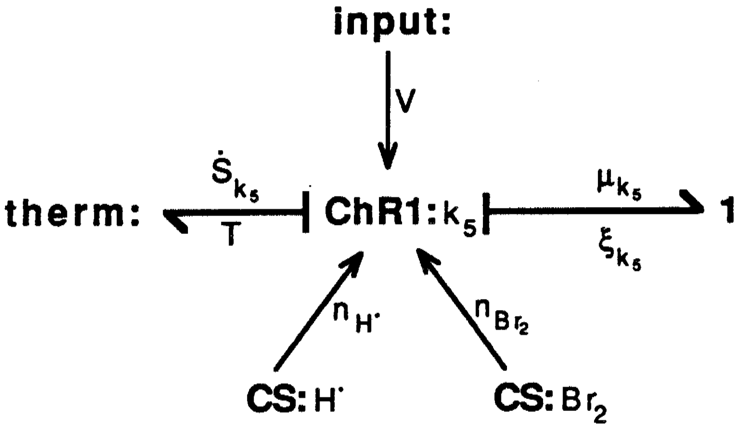
From this proof it follows that the second power balance cannot be decomposed into component reactions, and thus, the power balance equations now become:

$$0.0 = T \cdot \dot{S}_{k_i} + \mu_{k_i} \cdot \xi_{k_i} \tag{4.166}$$

$$\dot{p}_i \cdot V = \dot{\mu}_i \cdot n_i \tag{4.167}$$

4.4.3.1 The Bond Graph Elements

The above results require *two chemical reactors ChR1* and *ChR2* to compute the two power balance equations separately. In the *ChR1-module* attached to the 1-junction, the first power balance equation will be used to evaluate the partial entropy $\dot{S}_{k_i}$ of component reactions and to solve the kinetic equations for the reation flow rates ξ_{k_i}. With this process the causalities and the signal paths of the Bond graphs are fixed as well as for the chemical reactor in the isobaric–isothermic case. The *ChR2-element* of the fifth reaction in the Bond graph model and its code in DYMOLA are shown in Fig. 4.11.



```
model type ChR1k5
   cut chem1(muk5/ - xik5), thermk5(Temp/ - Sdotk5)
   main cut C [chem1]
   terminal nBr2, nH, V
   parameter R = 8.314
   local k5

   k5 = 10.0**11.97 * exp(-149800.0/(R * Temp))
   0.0 = Temp * Sdotk5 + muk5 * xik5
   xik5 = k5 * nBr2 * nH/V
end
```

Figure 4.11: The ChR1–element in a Bond graph and coded in DYMOLA

Under isochoric–isothermic conditions, the total chemical power P_{chem} depends on two different kinds of chemical power:

$$P_{chem} = \dot{\mu}_i \cdot n_i + \mu_i \cdot \xi_i \tag{4.168}$$

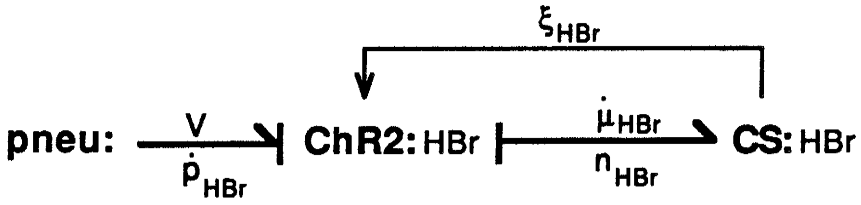
The second power balance relating to the second term of equ.(4.168) has to be evaluated in the chemical reactor *ChR2* of any component substance. Together with the chemical reactor *ChR1*, the *ChR2*–elements correspond to the chemical reactor *ChR* in the isobaric–isothermic case which was actually only a special case where the dynamic character of the second power balance equation disappeared. For this separation, the ***pneu–cut*** of the *ChR*– element was transferred into the *ChR2*–element rather than into the *ChR1*–element since the partial pressure p_i of any pure substance is to be evaluated from the second balance equation and is needed in the *Pneumatic*–element to calculate the total volume V. Both V and p_i are transferred through the ***pneu–cut*** of the *ChR2*– and the *Pneumatic*–element. Further, since the *ChR2*–elements are to compute the number of moles n_i, they have to have the molar flow rates ξ_i as input or as terminal of the *CS*–element connected to the *ChR2*–element. These assumptions lead to the entire Bond graph model of the *ChR2*–module and its implementation in DYMOLA given in Fig. 4.12 for the *HBr*-gas:

There is only the equation of state left which is utilized to compute the chemical potential μ_i of component gases in the *CS*–element. Since the reaction occurs under constant volume the equation of state is exploited conveniently to compute the partial pressure change p_i of component gases as follows:

$$\dot{p}_i = \frac{\xi_i \cdot R \cdot T}{V} \tag{4.169}$$

Equ.(4.166) plugged into the second power balance equation (4.161) leads to the formula to evaluate the chemical potential μ_i:

$$\dot{\mu}_i = \left(\frac{\xi_i}{n_i}\right) \cdot R \cdot T = \left(\frac{\dot{x}_i}{x_i}\right) \cdot R \cdot T \tag{4.170}$$



```
model type ChR2HBr
   cut chem2(mudotHBr/ - nHBr), pneu5(pdotHBr/V)
   main cut C [chem2]
   terminal xiHBr

   pdotHBr * V = mudotHBr * nHBr
   der(nHBr) = xiHBr
end
```

Figure 4.12: The ChR2–element in a Bond graph and coded in DYMOLA

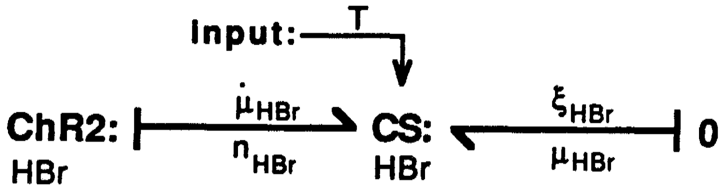
There is still a problem to be solved in equ.(4.170). If the number of moles n_i equals zero, for example, by starting the chemical reaction with the initial conditions of n_{HBr}, $n^{\bullet}_{H}$, and $n^{\bullet}_{Br}$ being zero, equ.(4.170) exhibits a singularity. This singularity physically means a discontinuity in the chemical potential. Since no matter has no chemical potential, but the smallest amount of matter has chemical potential equal to the the free energy of formation, the value of the chemical potential jumps from zero at $n_i = 0.0$ to the value of the free energy of formation. Practically, this discontinuity does not bother any further since the free energy of formation will be assigned as an initial condition of the chemical potential of component gases even before the first gas molecule appears in the reaction. Thus, the discontinuity is numerically avoided best by adding a small number ϵ to the denominator of equ.(4.170). This number shall be chosen at least four orders of magnitude smaller than the smallest number n_i of moles expected.

Interestingly, if equ.(4.170) is integrated to:

$$\mu_i = \mu_i^* + (R \cdot T)\ln(x_i) \tag{4.171}$$

then this equation looks pretty much the same as the equation used in most of the textbooks of chemical thermodynamics [**33,40,47,48**] to compute the chemical potential of ideal gases. But equ.(4.171) has only been considered in a very special case, namely the isochoric–isothermic case, and is not valid for the variety of conditions many textbooks assume.

Back to the *CS*–element. Since in the *CS*– element the time derivative of the chemical potential $\dot{\mu}_i$ is needed in the chemical reactor *ChR2*, and the number of moles n_i evaluated within this reactor which is connected to the *CS*–element, has to be known as well. It comes as no surprise that the *CS*– and the *ChR2*–elements are connected by the *chem2–cut* that transfers both $\dot{\mu}_i$ and n_i. The equivalent reason breaks down for the *chem1–cut* introduced between the *CS*–element and the *ChR2*–reactor.



```
model type CS
   cut chem1(mu/xi), chem2(mudot/n)
   main cut C[chem1]
   terminal Temp
   parameter R = 8.314, eps = 1.0E−15

   mudot * (n + eps) = xi * R * Temp
   der(mu) = mudot
end
```

Figure 4.13: The generic CS–element in a Bond graph and coded in DYMOLA

Fig. 4.13 shows the generic model type of the *CS*–element with the temperature *T* as modulation signal for the *HBr*–gas.

If one wants to evaluate either the total entropy, or the total number of moles, or the Gibbs' free energy, then the model types *Thermic, Chemical, and Chpower*, have to be added to the DYMOLA program shown in Appendix B.

4.4.3.2 The Main Program and the Bond Graph of the Complete Reaction

The complete hydrogen–bromine reaction is shown in Fig.4.14 whereby the fourth reaction being the least important one is left out to keep the Bond graph planar. The signal paths are also not drawn to make the Bond graph less cluttered.

Since the Bond graph model is already rather extensive, the modified Bond graph model which could be directly translated into the DYMOLA language will not be shown. But instead, the chemical system will be analyzed by counting the ***bond–model types*** and the ***nodes*** needed to appropriately describe the Bond graph model in the modeling language. Presently there are five 1–junctions corresponding to five chemical reactions with ***21 bonds*** attached to, which can be split into two categories. The first category of bonds represents the connection of 1–junctions with 0–junctions. Ten of this type can be read out from Fig.4.14 plus four of the fourth reaction shown in Fig.4.15.

The second category describes the connection between 1–junctions and multiport– elements: For instance, the transformer *TF* and the chemical reactors ***ChR1***. In this category of Bond graphs, normally, additional 0–junction should be inserted since according to the rule of Chapter 3, any model–type–element has to be embedded in between either two nodes or at least one node and another model–type–element (due to the absence of the 1–junction in the DYMOLA– language). One can count seven bonds of this category. The same number of nodes which

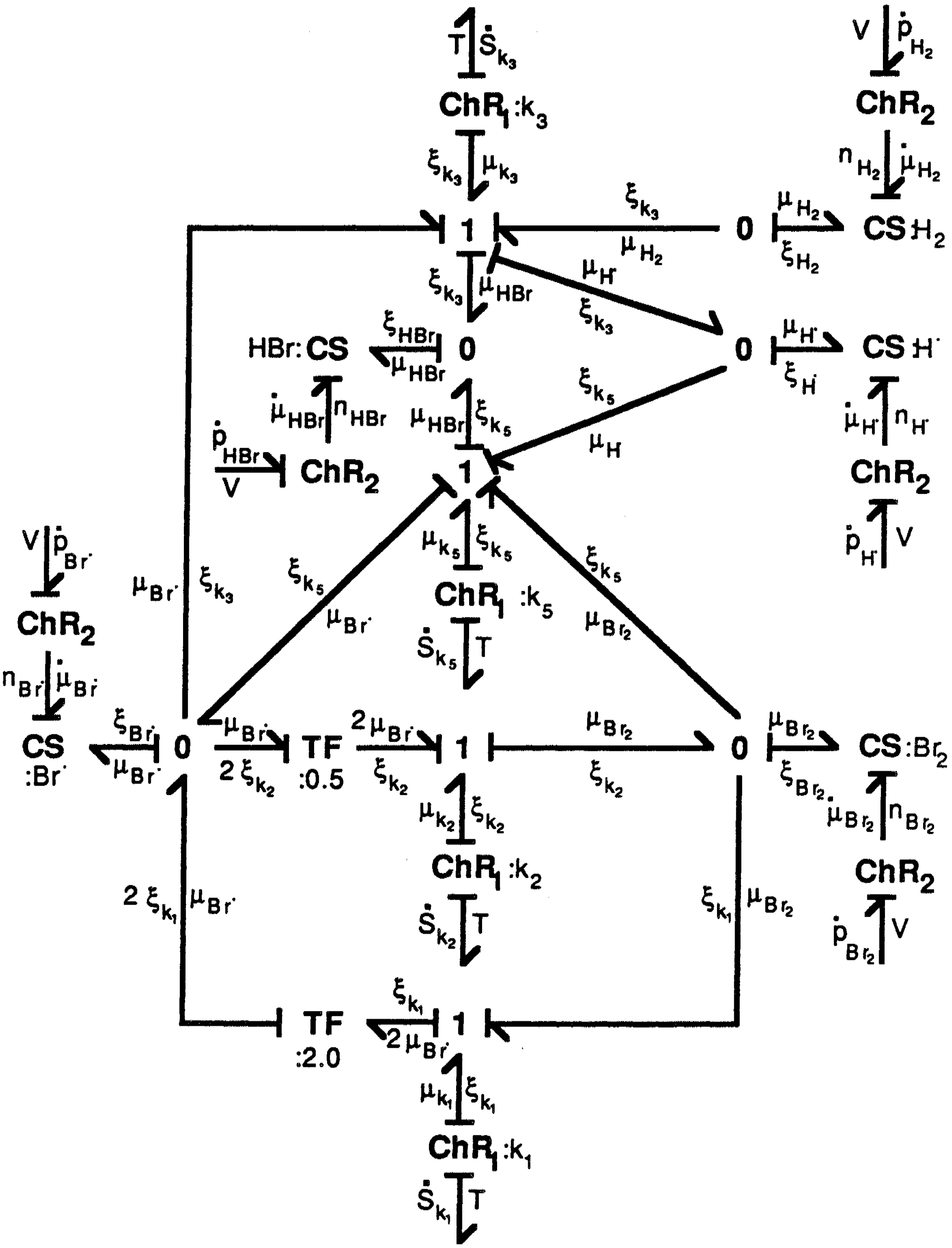


Figure 4.14: Bond graph model of the isochoric–isothermic H_2–Br_2–reaction

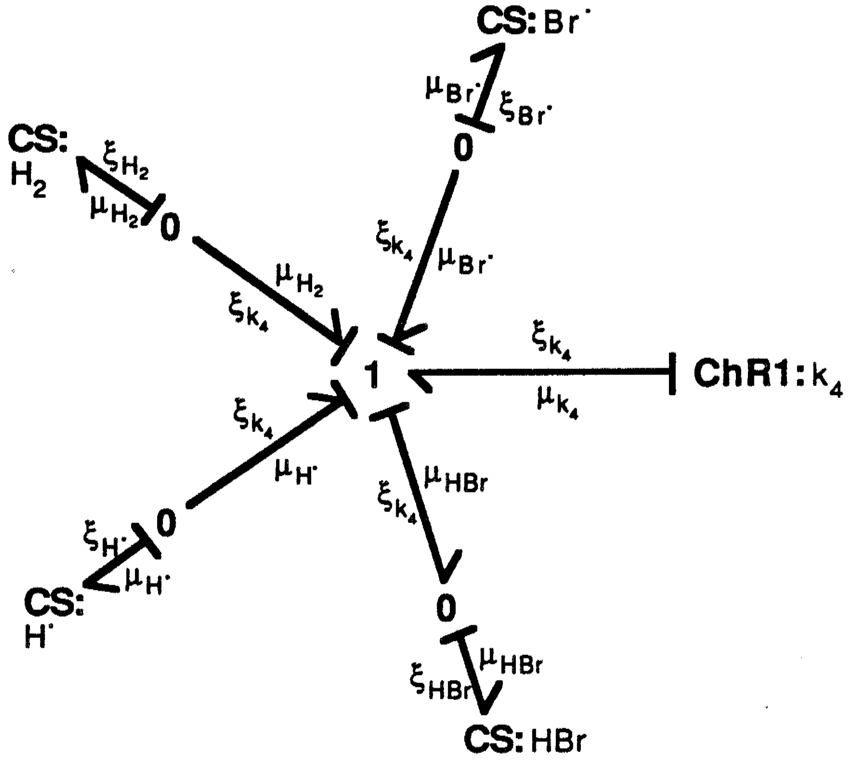


Figure 4.15: The fourth reaction illustrated in Bond graphs

represent the newly added 0–junctions are to be introduced plus the number of 0–junctions already existing. This is equal to the number of pure substances i, i.e., five of them, and the number of 1–junctions (five of them also). These are 17 altogether.

The construction of the main program follows, as described in Chapter 3:

1. Loading the model–types:

```
@bond.lib
@TF.lib
@Thermic.ct
@Pneumati.ct
@Chemical.ct
@Chpower.ct
@CS.ct
@ChR1k1.ct
@ChR1k2.ct
@ChR1k3.ct
@ChR1k4.ct
@ChR1k5.ct
@ChR2Br2.ct
@ChR2Br.ct
@ChR2H2.ct
@ChR2H.ct
@ChR2HBr.ct
```

2. Calling the submodels:

```
submodel ChR1k1
submodel ChR1k2
submodel ChR1k3
submodel ChR1k4
submodel ChR1k5
submodel ChR2B2  (ic nBr2 = 0.5E-7)
submodel ChR2Br  (ic nBr = 0.0)
submodel ChR2H2  (ic nH2 = 0.5E-7)
submodel ChR2H   (ic nH = 0.0)
submodel ChR2HB  (ic nHBr = 0.0)
submodel (CS)        CSBr2 (ic mu1 = muBr2) ->
                     CSBr  (ic mu1 = muBr)  ->
                     CSH2  (ic mu1 = muH2) ->
                     CSH   (ic mu1 = muH)  ->
                     CSHBr (ic mu1 = muHBr)

submodel Pneumatic   (ic p = 1.0E+5)
submodel Thermic     (ic S = S0)
submodel Chpower     (ic G = 0.0)
submodel Chemical
submodel (TF)       TF1(k = 0.5), TF2(k = 2.0)
submodel (bond)     B11, B1Br2, B1Br, B21, B2Br2, B2Br, B31
submodel (bond)     B3H2, B3H, B3HBr, B3Br, B41, B4H, B4HBr
submodel (bond)     B4Br, B4H2, B51, B5Br, B5Br2, B5H, B5HBr
```

Note that in the current version of DYMOLA it is not possible to declare the initial conditions as variables evaluated in the main program itself. But this capability would make sense since it is very often necessary to run a program several times, for instance in the case of chemical reactions, with different values of temperature which directly influences the free energy of formation. However, this energy composed by the temperature dependent heat of formation and the molar entropy of component substances i:

$$\mu_0 = h(T) - T \cdot s(T) \tag{4.172}$$

represents the initial condition of the chemical potential evaluated in the *CS*–element. Since the reference pressure p_0 has been made equal to the operating pressure p in order to make the simulation easier, the molar entropy S_i evaluated from equ.(4. 134) will actually depend only on the temperature as correctly shown in equ.(4.172).

The initial condition S_0 of the *Thermic*–element can be computed from the sum of the single entropies $S0_i$ of n_{Br_2} and n_{H_2} at the temperature $T = 800^{\circ}K$, since at the initial time $t = 0.0s$ the two gases can be considered to be separate and in chemical equilibrium. This assumption is valid for all isothermic cases which have been analyzed above.

Also, the current DYMOLA version does not allow negative initial conditions to be directly specified in a submodel–call.

In Appendix B the main program which actually generates the DESIRE program appears. Some further remarks about this main program explaining the practical implementation of negative initial conditions follow. For this purpose, the sc–variable, being the variable of the switch–function in the *CS*–element is introduced. If the initial condition has a negative sign then sc equals 0.5 and the switch–function is $sgn(0.5 - 1) = -1$. For the initial condition being positive sc is 1.5 and the switch–function is $sgn(1.5 - 1) = +1$.

One might erroneously conclude that the inability of the current version of DYMOLA to allow the declaration of variable initial conditions represents a serious problem when a chemical reaction occurs under non–isothermic conditions, where the free energy of formation μ_0 depends on the temperature. But since the free energy of formation will only be considered once during a complete simulation, namely as the initial value, the change of its value is of no interest any more for any time greater than zero.

3. Computation of the free energy of formation and reading the equilibrium constant K:

```
func K900   = KBr2Bt(Temp0)
func HiBr2  = HiBr2t(Temp0)
func HiBr   = HiBrt(Temp0)
func HiH2   = HiH2t(Temp0)
func HiH    = HiHt(Temp0)
func HiHBr  = HiHBrt(Temp0)
func SiBr2  = SiBr2t(Temp0)
func SiBr   = SiBrt(Temp0)
func SiH2   = SiH2t(Temp0)
func SiH    = SiHt(Temp0)
func SiHBr  = SiHBrt(Temp0)

muBr2  = HiBr2  - Temp0*SiBr2
muBr   = HiBr   - Temp0*SiBr
muH2   = HiH2   - Temp0*SiH2
muH    = HiH    - Temp0*SiH
muHBr  = HiHBr  - Temp0*SiHBr

S0 = SiBr2 + SiH2
```

   These values of the free energy of formation are used as initial conditions of the *CS*–element as mentioned above.

4. Declaration of the input and output

5. Declaration of the variable type

6. Declaration of the nodes

7. Connections of the Bond graph elements

8. Setting the modulation signals

9. Connection of the n_i–state variables with the outputs

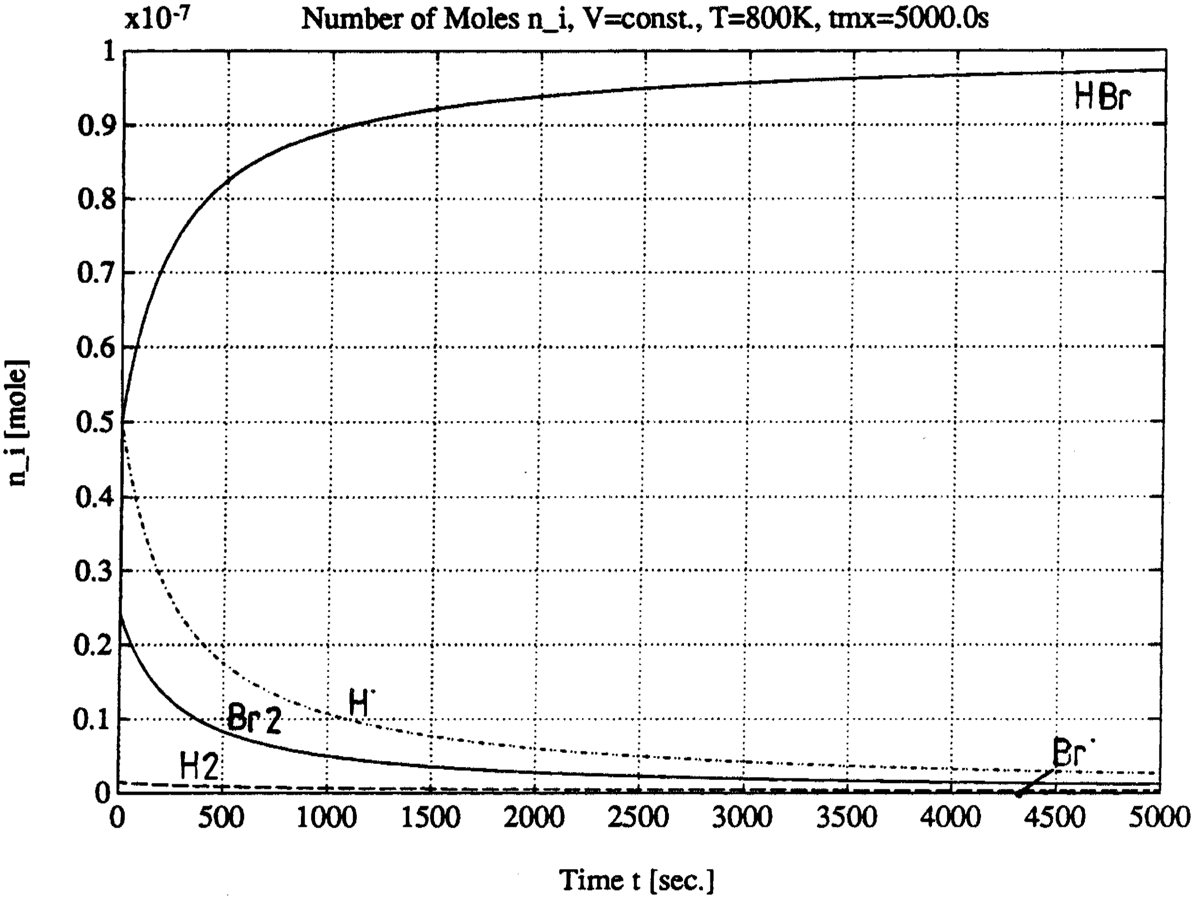


Figure 4.16: Change of the number of moles in the $H_2 - -Br_2$ reaction, $T = 800^oK, V = 1.0E - 3\,m^3$

The last six items are thoroughly explained in Chapter 3, and thus, their code is not given now but is included in Appendix B. The DYMOLA code is compiled into the simulation language DESIRE. By executing the DESIRE program, the graph of the number of moles n_i changing over time and plotted by the graphic module of MATLAB, [25], is achieved. This is shown in Fig. 4.16.

This graph will be analyzed in more detail in Section 4.5.

4.4.4 The Isentropic–Isobaric Case

Mainly, the isentropic–isobaric case of the hydrogen–bromine reaction can be considered equivalently to the isochoric–isothermic case concerning the number of model types needed in the Bond graph. Thus, this case will not be described as much in detail as the isochoric–isothermic case but each model–element will be succinctly shown. The reason why the isentropic–isobaric case is explained nevertheless, is that it needs to be analyzed how the changing temperature influences the behavior of the chemical system. Remember, there is a large temperature dependency of the reaction constants k_i according to Arrhenius' Law.

This time, it has proved to be more convenient to operate with ***partial temperature flows*** $\dot{T}_i$ rather than with ***partial entropy flows*** in the isothermic cases. These "partial temperature flows" do not exactly have a physical meaning, but they work mathematically just fine. The two power balance equations boil down to:

$$\begin{aligned} p \cdot \Phi_{k_i} &= \mu_{k_i} \cdot \xi_{k_i} && (4.173) \\ 0.0 &= \dot{T}_i \cdot S + \dot{\mu}_i \cdot n_i && (4.174) \end{aligned}$$

Since three quantities of the state of equation vary, the equation of state looks different from the isothermic cases for component substances i:

$$p \cdot \Phi_i = \xi \cdot R \cdot T_i + n \cdot R \cdot \dot{T}_i \qquad (4.175)$$

where ξ represents the total molar flow rate and n denotes the total number of moles. When plugging equ.(4. 173) into equ.(4. 175), it follows that:

$$\mu_i \cdot \xi_i = \xi \cdot R \cdot T_i + n \cdot R \cdot \dot{T}_i \qquad (4.176)$$

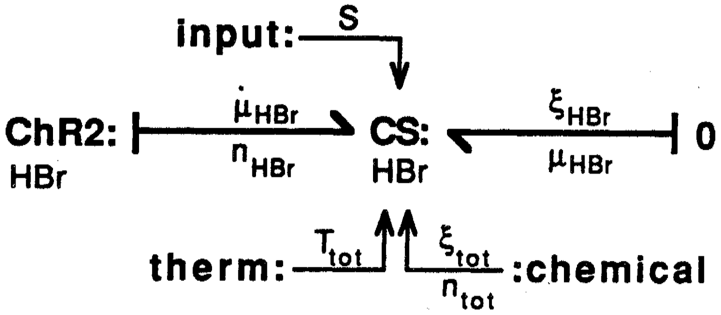
Expanding equ.(4. 176) by S, and plugging equ.(4. 174) into equ(4. 176), it can be seen that:

$$\dot{\mu}_i \cdot n_i \cdot n \cdot R = \xi \cdot R \cdot T_i \cdot S - \mu_i \cdot \xi_i \cdot S \qquad (4.177)$$

The appropriate *CS*–element for *HBr* is illustrated in Fig.4.17. Note that, this time, the *CS*–element is modulated by the input ***entropy S***, the ***total temperature T_{tot}*** evaluated in the *Thermic*–element, the ***total flow of moles ξ_{tot}***, and ***the total number of moles n_{tot}***, both computed in a new element, called *Chemical*. The *Chemical*–element is connected with the *CS*–elements by terminals rather than by means of power that would have been achieved by cuts. Fig.4.18 shows the *ChR1*–element for the fifth equation, and Fig.4.19 depicts the *ChR2*–element for the *HBr*–molecule. Note that the thermic and pneumatic/hydraulic ports have switched the places in between the chemical reactors ChR_i in comparison to the isochoric–isothermic case. The *Pneumatic*–element is the same as in the isobaric–isothermic case. Below the DYMOLA codes of the *Thermic*– and the *Chemical*–elements are given:

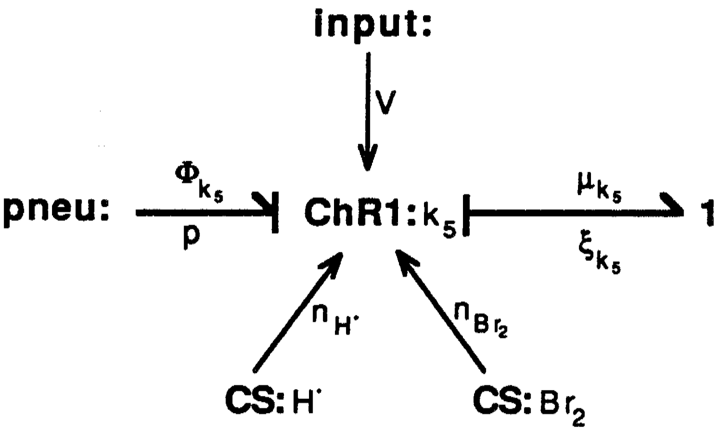
```
model type Thermic
   main cut thermtot(Tdot/-S)
   cut thermBr2(TdotBr2/-S), thermBr(TdotBr/-S)
   cut thermH2(TdotH2/-S), thermH(TdotH/-S)
   cut thermHBr(TdotHBr/-S)

   Tdot = TdotBr2 + TdotBr + TdotH2 + TdotH + TdotHBr
   der(T) = Tdot
end
```

```
model type Chemical
   terminal xiBr2, xiBr, xiH2, xiH, xiHBr
   terminal ntot, xitot

   xitot = xiBr2, xiBr + xiH2 + xiH + xiHBr
   der(ntot) = xitot
end
```



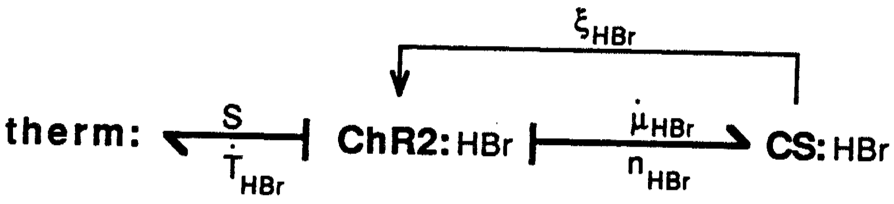
```
model type CS
   cut chem1(mu/xi), chem2(mudot/n)
   main cut C[chem1]
   terminal Ttot, S, ntot, xitot
   parameter R = 8.314, eps = 1.0E-15

   mudot * (n + eps) * ntot * R = xitot * R * Ttot * S - mu * xi * S
   der(mu) = mudot
end
```

Figure 4.17: The CS–element in a Bond graph and coded in DYMOLA



```
model type ChR1k5
   cut chem1(muk5/ - xik5), pneuk5(p/Phik5)
   main cut C [chem1]
   terminal nBr2, nH, V
   parameter R = 8.314
   local k5

   k5 = 10.0**11.97 * exp(-149800.0/(R * Temp))
   p * Phik5 = muk5 * xik5
   xik5 = k5 * nBr2 * nH/V
end
```

Figure 4.18: The ChR1–element in a Bond graph and coded in DYMOLA



```
model type ChR2HB
   cut chem2(mudotHBr/ - nHBr), thermHBr(TdotHBr/ - S)
   main cut C[chem2]
   terminal xiHBr

   0.0 = TdotHBr * S + mudotHBr * nHBr
   der(nHBr) = xiHBr
end
```

Figure 4.19: The ChR2–element in a Bond graph and coded in DYMOLA

Fig.4.20 shows the entire Bond graph of the isentropic–isobaric hydrogen–bromine reaction with the same simplifications done as in the isothermic cases. Appendix B includes all the programs of the model types, the main program, the DESIRE program, and Appendix D includes graphs of the number of moles changing with time as well as graphs of the changing total temperature, volume, the change of Gibbs' free enthalpy, the reaction enthalpy, and the change of the mole fraction. The graphs were faster to be generated by ACSL programs than by the DESIRE programs. Appendix E includes the ACSL–programs of the three cases.

4.4.5 Further Cases for the Hydrogen–Bromine–Reaction

The *isentropic–isochoric* case of the hydrogen–bromine reaction turns out to be more difficult than the previous three cases. Looking at the two power balances:

$$\dot{p} \cdot V = \dot{T} \cdot S + \sum_{i=1}^{I} \dot{\mu}_i \cdot n_i \tag{4.178}$$

$$0.0 = \sum_{i=1}^{I} \mu_i \cdot \xi_i \tag{4.179}$$

the second power balance equation (4.179) boils down to be unsolvable for individual components i. E.g., since the molar flow rates ξ_i are evaluated from the kinetic equations, exactly one of the chemical potentials μ_i can be computed from equ.(4.179). The first power balance equation (4.178) can be decomposed into component substances i:

$$\dot{p}_i \cdot V = \dot{T}_i \cdot S + \dot{\mu}_i \cdot n_i \tag{4.180}$$

However, equ.(4.180) requires all the derivatives of the chemical potentials $\dot{\mu}_i$ to be known, thus, equ.(4.179) and equ.(4,180) are structurally singular.

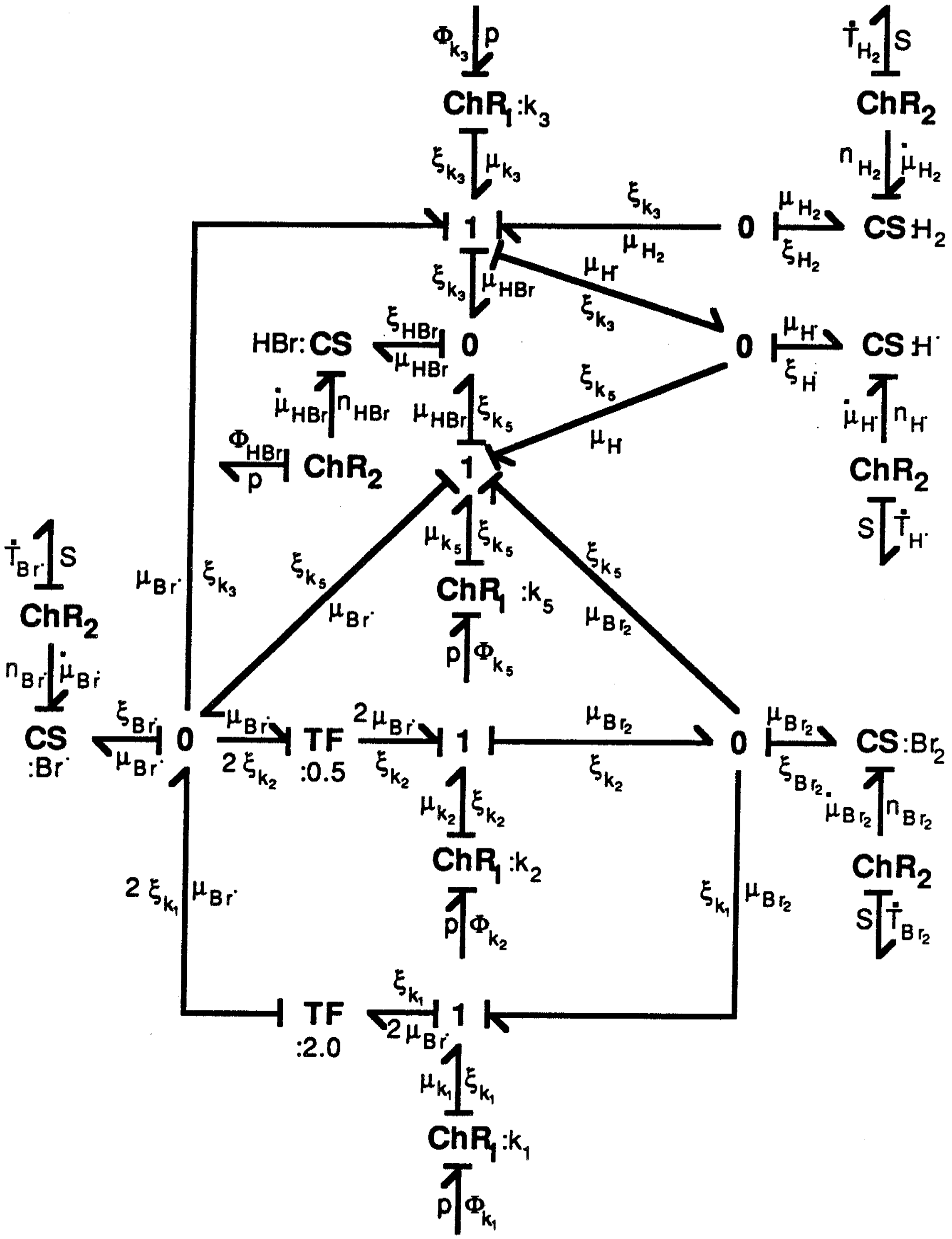


Figure 4.20: Bond graph model of the hydrogen–bromine reaction, S=const., p=const.

Finally, the ***adiabatic–isobaric***–case will be considered. Since the total thermic power adds up to zero, the external condition can be written as:

$$\sum_{i=1}^{I}\left(T\cdot\dot{S}_i+\dot{T}\cdot S_i\right)=0.0 \tag{4.181}$$

If it is decided to operate on partial entropy flows of component substances S_i, the power balance equations both written in component substances, turn into:

$$\begin{aligned} p\cdot\Phi_i &= T\cdot\dot{S}_i+\mu_i\cdot\xi_i & (4.182)\\ 0.0 &= \dot{T}\cdot S_i+\dot{\mu}_i\cdot n_i & (4.183)\end{aligned}$$

The equation of state for ideal gases can be written as:

$$p\cdot\Phi_i=n_i\cdot R\cdot\dot{T}+\xi_i\cdot R\cdot T \tag{4.184}$$

Equ.(4. 181) is to evaluate $\dot{T}$, equ.(4. 182) computes $\dot{S}_i$, equ.(4. 183) is used to find $\dot{\mu}_i$, and equ.(4. 184) determines Φ_i. Since each of the four equations (4. 181–184) includes furthermore one of the unkown variables $\dot{T},\dot{S}_i,\dot{\mu}_i,\Phi_i$ — besides the causal variable —, this set of equations turns out to contain an algebraic loop expanding over all these equations.

Of course, there are further possibilities to choose external constraints for the hydrogen-bromine reaction, but then, the received set of equations could be treated similarly to one of the cases previously considered. The following section includes the validation of the chemical reactions' model, i.e. the simulation results of the different cases will be compared to physical equations as well as to results of another simulation program.

4.5 Verifying the Model of a Chemical Reaction

One of the main objectives of an engineer modeling physical systems represents the model validation of his abstract mathematical construction. For this, he will compare his simulation results to data received by experiments or by other simulations in which the data has been validated.

4.5.1 The Balances of Moles, Internal Energy, and the Steady–state of Moles, and the Reaction Enthalpy

In this process it is of interest to consider the total number of Br and H atoms, which have to be constant over time since the chemical system is closed. Also the internal energy U has to stay constant. The total numbers of Br and H–atoms — aBr and aH — can be added up stoichiometrically from the individual component gases:

$$aBr = 2 \cdot n_{Br_2} + n_{Br\bullet} + n_{HBr} = const. = 2 \cdot n^0_{Br_2} \qquad (4.185)$$

$$aH = 2 \cdot n_{H_2} + n_{H\bullet} + n_{HBr} = const. = 2 \cdot n^0_{H_2} \qquad (4.186)$$

They have to be constants equal to twice the initial condition of the number of moles n^0_i since mass is neither added nor substracted.

The simulation runs generate constant a_i for all three cases as desired from the above equation. The graphs are given in Appendix D. The internal energy U stays constant over time as it can easily be shown in simulation runs.

The steady–state of the number of moles n_i and the difference of the chemical energy's steady–state G and the initial condition G_0 can be compared to values calculated by hand. Equ.(4. 148)

$$\xi_i = \sum_{k_i=1}^{K} \nu_{i,k_i} \cdot \xi_{k_i}$$

applied to the hydrogen–bromine reaction leads to equs.(4. 187–191):

$$\frac{d}{dt} n_{Br_2} = -\xi_{k1} + \xi_{k2} - \xi_{k5} \qquad (4.187)$$

$$\frac{d}{dt}n_{Br\bullet} = 2\xi_{k1} - 2\xi_{k2} - \xi_{k3} + \xi_{k4} + \xi_{k5} \tag{4.188}$$

$$\frac{d}{dt}n_{H_2} = -\xi_{k3} + \xi_{k4} \tag{4.189}$$

$$\frac{d}{dt}n_{H\bullet} = \xi_{k3} - \xi_{k4} - \xi_{k5} \tag{4.190}$$

$$\frac{d}{dt}n_{HBr} = \xi_{k3} - \xi_{k4} + \xi_{k5} \tag{4.191}$$

The time derivatives of the molar flow rates ξ_i are set to zero in the above equations if the steady–state of the number of moles n_i is to be considered. Solving this linear system of singular equations for the molar flow rates of component reactions ξ_{k_i}, the following set of equations results:

$$\xi_{k_1} = \xi_{k_2} \tag{4.192}$$

$$\xi_{k_3} = \xi_{k_4} \tag{4.193}$$

$$\xi_{k_5} = 0.0 \tag{4.194}$$

Plugging equs.(4. 68–72) into equs.(4. 192–194), the set of equations boils down to a nonlinear system of equations in terms of the number of moles n_i:

$$k_1 \cdot n_{Br_2} = k_2 \cdot \frac{n_{Br}^2}{V} \tag{4.195}$$

$$k_3 \cdot \frac{n_{Br} \cdot n_{H_2}}{V} = k_4 \cdot \frac{n_{HBr} \cdot n_H}{V} \tag{4.196}$$

$$k_5 \cdot \frac{n_H \cdot n_{Br_2}}{V} = 0.0 \tag{4.197}$$

There are five unknowns n_i and five equations if the equs.(4. 185,186) are taken into account. Since all the equations for the computation of energy are only valid for the chemical reaction being stoichiometric, the initial conditions of n_{Br_2} and n_{H2} must be set equal:

$$n_{Br_2}^0 \equiv n_{H_2}^0 \tag{4.198}$$

Solving the above set of equations by taking note of equ.(4. 198), the following single solution is obtained:

$$n_{Br_2} = 0.0 \tag{4.199}$$

$$n_{Br^\bullet} = 0.0 \tag{4.200}$$

$$n_{H_2} = 0.0 \tag{4.201}$$

$$n_{H^\bullet} = 0.0 \tag{4.202}$$

$$n_{HBr} = 2 \cdot n^0_{Br_2} = 2 \cdot n^0_{H_2} \tag{4.203}$$

One must be careful not only to consider the first seconds of the simulation of the hydrogen–bromine reaction but simulate the reaction over a long time, for example, $5000sec.$. Fig.4.21 which shows the changing number of moles in the isochoric–isothermic hydrogen–bromine reaction simulated over $3.3sec$, might provoke the following steady–states assuming:

$$\begin{aligned} n_{Br_2} &\approx 0.5 \cdot n^0_{Br_2} \\ n_{Br^\bullet} &\approx 0.0 \\ n_{H_2} &\approx 0.0 \\ n_{H^\bullet} &\approx n^0_{Br_2} \\ n_{HBr} &\approx n^0_{Br_2} \end{aligned}$$

Fig.4.22 shows the actual steady–states of the number of moles which are equivalent to those received by equ.(4. 199–203).

The *reaction enthalpy* can be theoretically calculated using the following equations:

$$g_i(T) = h_i(T) - T \cdot s_i(T) \tag{4.204}$$

$$\Delta g(T) = \sum_{i=1}^{I} \nu_i \cdot g_i(T) \tag{4.205}$$

The model type which is needed in the simulation to gain this additional information is shown in Fig. 4.23.

It is apperent that the terminals of the chemical potentials μ_i and their time derivative $\dot{\mu}_i$, and the molar flow rates ξ_i have to be connected to the appropriate

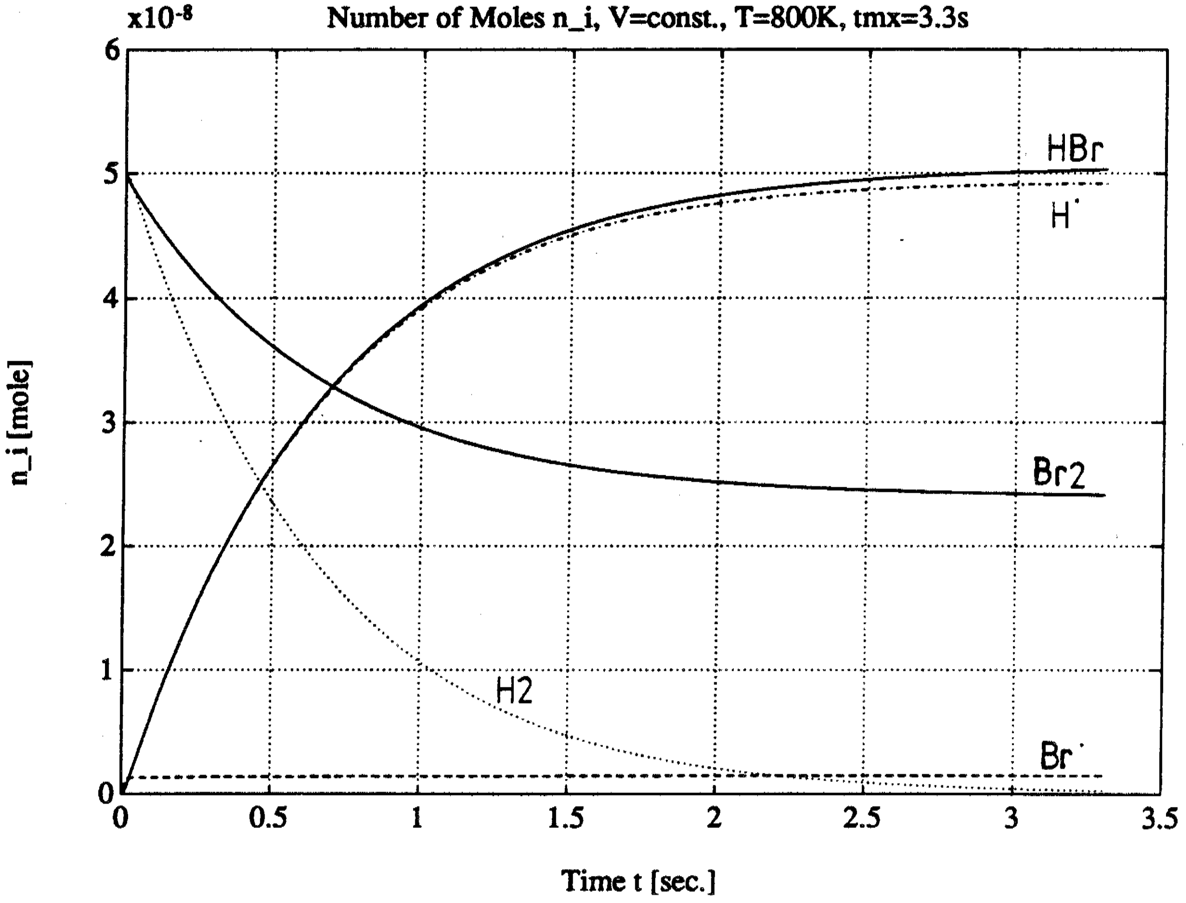


Figure 4.21: Number of Moles in the hydrogen–bromine reaction; simulation time 3.3$sec.$, $V = const., T = const.$

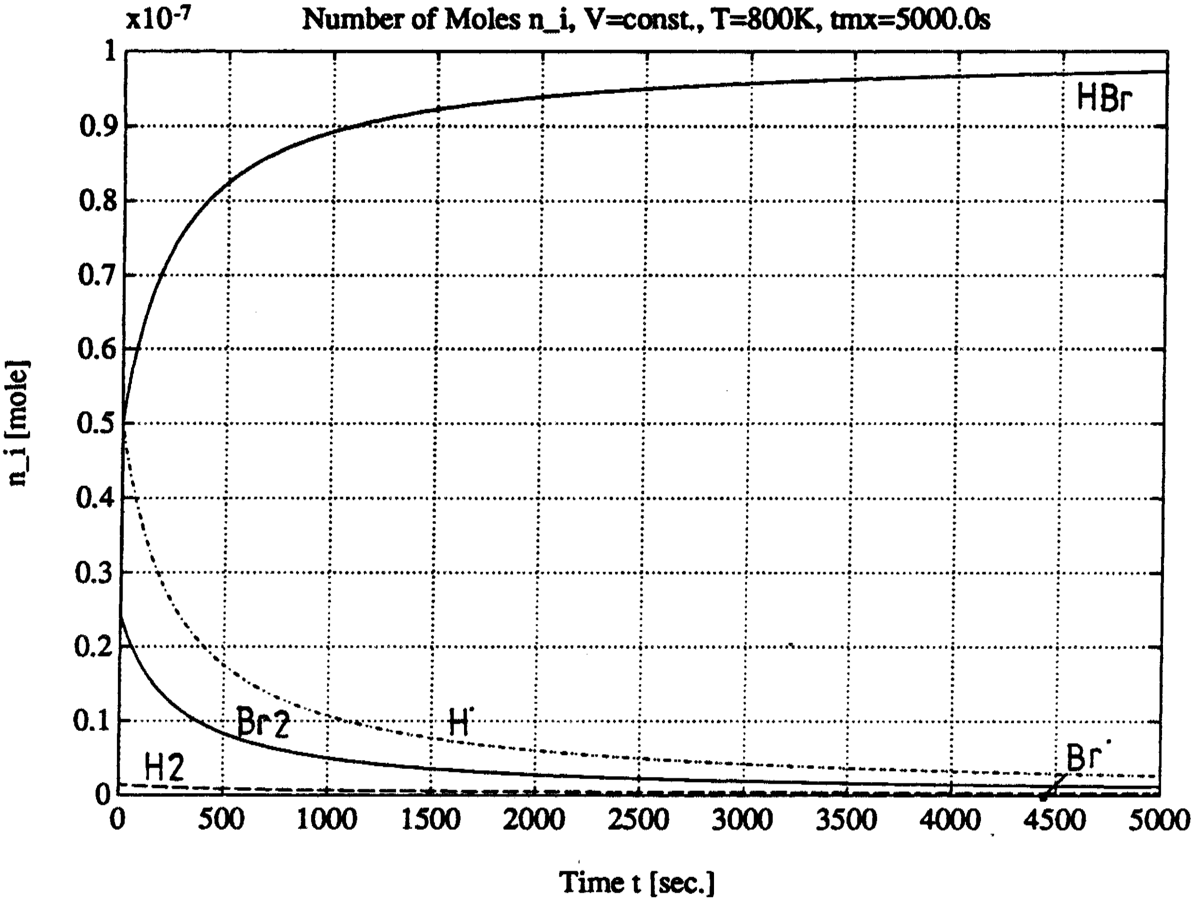


Figure 4.22: Number of Moles in the hydrogen–bromine reaction; simulation time $5000 sec., V = const., T = const.$

```
model type Chemical
    terminal nBr2, nBr, nH2, nH, nHBr
    local ntot

    ntot = nBr2 + nBr + nH + nH2 + nHBr
end
```

Figure 4.23: Model type to compute the Chemical Energy/Power

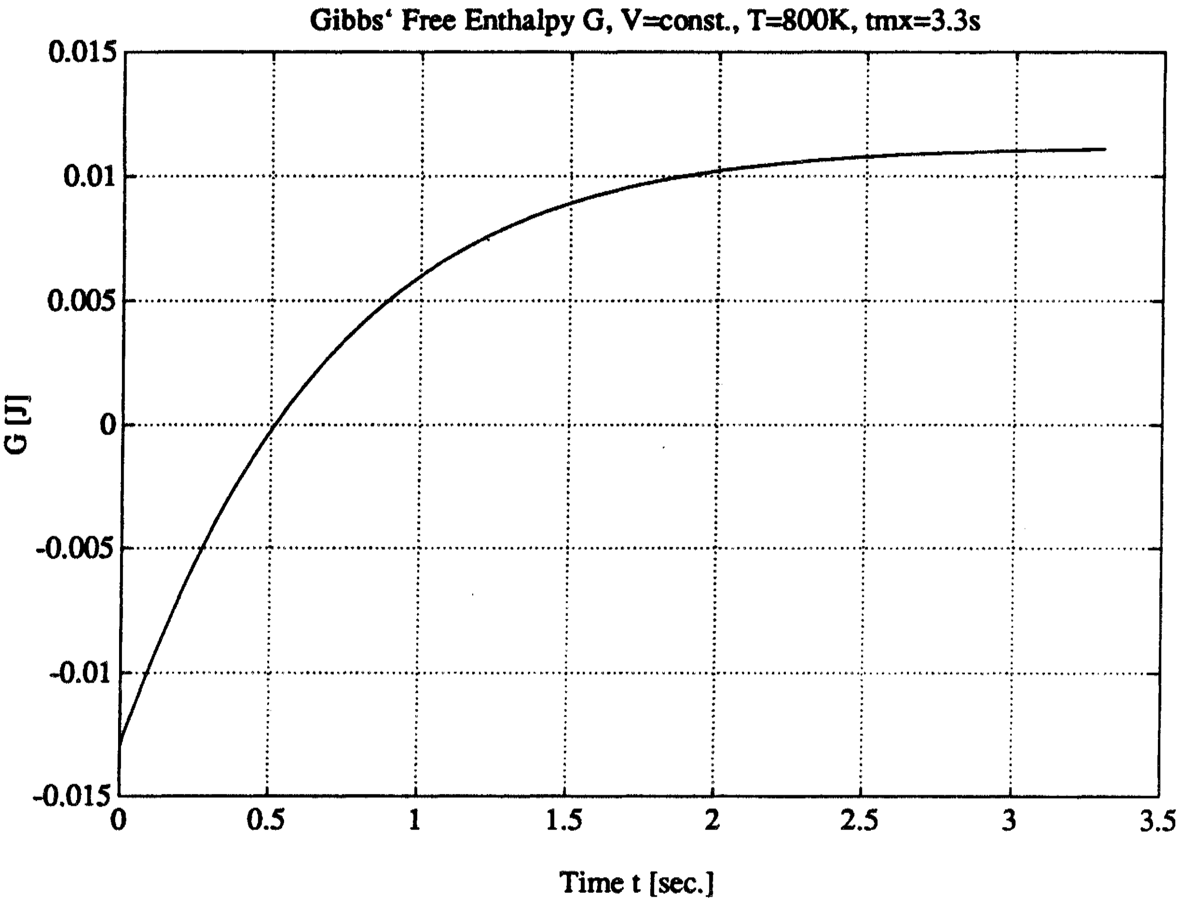


Figure 4.24: Gibbs' free energy in the hydrogen–bromine reaction; simulation time 3.3*sec.*, $V = const., T = const.$

CS_i–modules while the terminal of the number of moles n_i needs to be connected to the $ChR2_i$–element. As usually, the connections used in the *Chpower-model type* have to be set in the main program shown in the Appendix B.

The graphs which illustrate the change of the total chemical energy G constructed from the simulation data of the isothermic– isochoric hydrogen–bromine reaction are depicted below. For the *Gibbs' free energy*, the simulation also has to run over a long time as the comparison of Fig.4.24 which illustrates the simulation run over 3.3*sec.*, to Fig.4.25 — the simulation over 5000*sec.* — shows. The initial value of the Gibbs' free energy G has to be read out of Fig.4.24 since Fig.4.25 does not provide it. G jumps in the first period of the simulation to a high value seen in Fig.4.25 for zero time which leads to the supposed initial value $\approx -0.0085J$. The reaction enthalpy ΔG_r can be computed from the simulation run as the difference

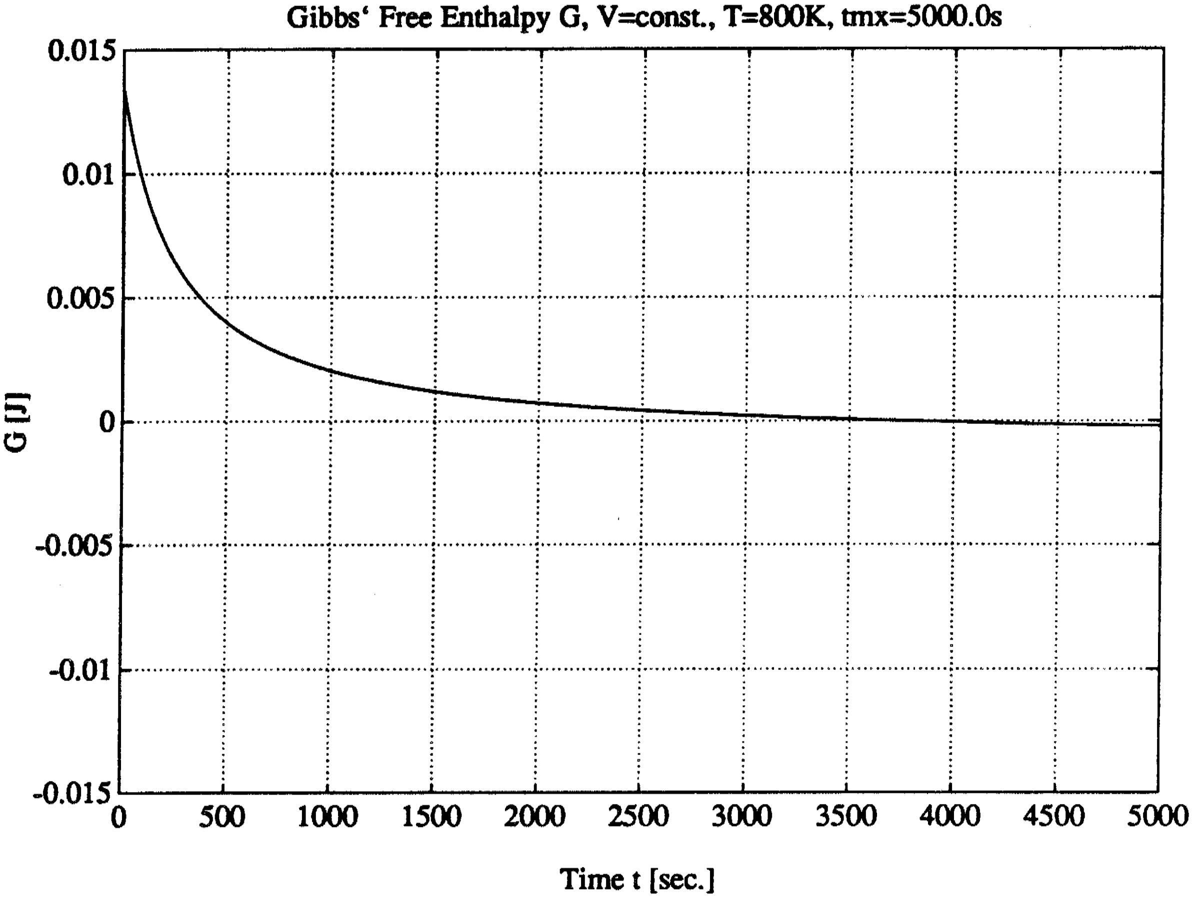


Figure 4.25: Gibbs' free energy in the hydrogen–bromine reaction; simulation time 5000.0*sec.*, $V = const., T = const.$

of the steady–state of the Gibbs' free energy and its initial value:

$$\Delta G_r = G(t \to \infty) - G(t = 0) \tag{4.206}$$

The following table 4.1 shows the reaction enthalpies of the isobaric–isothermic, isochoric–isothermic, and isentropic–isobaric hydrogen–bromine reaction in comparison to the theoretically calculated value. The enthalpies and entropies of equ.(4.204) have been looked up in the table of thermodynamic data [51].:

Table 4.1: Model validation by comparing the molar Gibbs free energy calculated by hand and evaluated in the simulation runs of the isobaric–isothermic, isochoric–isothermic, and isentropic–isobaric cases

Equ.(4.205)	$\Delta g(T = 800^oK) = -0.593E - 2J$
(p=const., T=800K)	$\approx -0.6E - 2J$
(V=const., T=800K)	$\approx +0.6E - 2J$
(S=const., p=const.)	$\approx -0.6E - 2J$

Since only the temperature varies, the change of the free Gibbs' energy $\Delta g(T)$ should be the same for the isobaric–isothermic and isocoric–isothermic cases according to equ.(4.205). But this is only true for the isobaric–isothermic and the isentropic–isobaric cases. The isochoric–isothermic reaction seems to be highly endothermic in the beginning (graphs are included in Appendix D), but the steady–state of G has a smaller value than its peak at the reaction's beginning. This is plausible, since in the beginning energy will be absorbed expressed by a positive value of ΔG until its activation energy is exceeded. Then, the reaction starts to occur and energy will be generated expressed by the ΔG–value which decreases again. The steady–state of ΔG is positive. This result is not in accordance to the result obtained by equ.(4.205) which has to be explained in future research. Note that the results of the validation for the isobaric–isothermic and isentropic–isobaric cases have been shown, of course, but not their model types for the model validation since these model types are similar to those of the isochoric–isothermic case which can be looked up in Appendix B.

4.5.2 Comparison to Another Simulation Program

There is yet another simulation program, the "Computer Program for Calculation of Complex Chemical Equilibrium Compositions, Rocket Performance, Incident and Reflected Shocks, and Chapman–Jouget Detonations", [59], which can be used to simulate the combustion in chemical reactions, and especially in this process to simulate the hydrogen–bromine reaction. This program, as the name implies, is used for computations involving chemical equilibria in complex systems using a free–energy minimization technique. "The program permits calculations such as (1) chemical equilibrium for assigned thermodynamic states (T,p), (H,p), (S,p), (T,V), (U,V), or (S,V), (2) theoretical rocket performance for both equilibrium and frozen compositions during expansion, (3) incident and reflected shock properties, and (4) Chapman–Jouget detonation properties. The program considers condensed species as well as gaseous species."[59] In this project only the first feature has been used for the hydrogen–bromine reaction.

For numerical reasons, the free–energy minimization formulation is used instead of the equilibrium constant method. "The condition for equilibrium may be stated in terms of several thermodynamic functions such as the minimization of the Gibbs free energy ($g = h - T \cdot s$) or Helmholtz free energy ($f = g - p \cdot v$) or the maximization of entropy. If one wishes to use temperature and pressure to characterize a thermodynamic state, the Gibbs free energy is most easily minimized inasmuch as temperature and pressure are its natural variables. Similarily, the Helmholtz free energy is most easily minimized if the thermodynamic state is characterized by temperature and volume (or density)."[**59**] The equations for this minimization problem as well as their numerical solution are not shown in these studies but can easily be looked up in [**59**].

This program package is introduced by being applied to the hydrogen–bromine reaction: The ***program input data*** includes all information in order to run the program. This can be discussed in four categories. One is optional and three of them are required. The four categories are illustrated by the program input data of the isochoric–isothermic hydrogen–bromine reaction:

1. Library of thermodynamic data for reaction products (THERMO data):

```
thermo                                                     *
```

   Since all the thermodynamic data of the reaction's reactants and products are included in the data base stored in the file "chemthrm.dat", there are no further data necessary to be implicitly written in the input file. Additional information about the "THERMO data" are found in [**59**].

2. Data pertaining to reactants (REACTANTS cards)

```
reactants                                                    *
h 2.                                   100.  3512.0   g800.0 f
br 2.                                  100.  11834.   g800.0 o
```

   There is a special input format to be followed which is shown in Table 1 in Appendix C.

   The first card in the set contains the word "reactants" ("h 2." and "br 2.") punched in card columns 1 to 9. The value "100." describes the relative weight of fuel in total fuel (the hydrogen has been chosen to be the fuel, thus "f" at the end of its card) or oxidant in total oxidant (bromine is oxidant, thus "o" at the end of its card) . Since in this case the reaction is considered to be stoichiometrical, these values will always be 100%.

The next column provides for the molar enthalpies of H_2 and Br_2. The unit is *cal/mole*.

Since H_2 and Br_2 are in gaseous state, this is indicated by "g". "800.0" descibes the temperature associated with the molar enthalpies from above.

3. Chemical formulae of species which are singled out for special purposes (optional)

```
omit        br2(l)                              *
```

which excludes liquid bromine to be considered by the reaction simulation. The names of from one to four species start in columns 16, 31, 46, and 61.

4. Namelist data which include the type of problem, required schedules, and options (NAMELISTS inputs) always require an "inpt2"–input.

```
namelists                                                    *
& inpt2 tv t nsqm t v 1.0E-3 of t mix 79.904 trace 1.0E-12
            temper 800 kase 50 eta 1.0
& end
stop                                                         *
```

Table 2 in Appendix C indicates which variables are reqired and which are optional for the various types of problems. Table 3 gives a brief definition of each variable. "tv" which is set to be true "t", expresses the thermodynamic state function to be isochoric and isothermic. If the pressure is desired to be in units of N/m^2, then "nsqm" has to be set true. The next expression sets the constant volume equal to $1.0E-3m^3$. "of" indicates that in "mix" the oxidant to fuel weight ratios is given. The value for "mix" is calculated by the following equation:

$$mix = \frac{molecular\ mass\ of\ oxidant\ (Br_2)}{molecular\ mass\ of\ fuel\ (H_2)} = 79.904$$

If compositions of species are to be printed if they are smaller than $5 \cdot 10^{-6}$, then "trace" has to be set equal to the desired value. The temperature "temper" equals 800^oK. "kase 50" represents an optional assigned number associated with the considered case. The efficiency coefficient "eta" always has a value of 1.0 if one is interested in receiving reaction equilibrium compositions.

Note that the program input data must be stored in a file named "cheminfi.ipt'.

The output file ("chemprnt.out") of the simulation provides the equilibrium composition of the compounds of the chemical reaction as well as its appropriate thermodynamic data. The essential part of the results of the isobaric–isothermic, isochoric–isothermic, and isentropic–isobaric cases, i.e. the equilibrium compositions (mole fraction x_i) and the values of entropy and enthalpy and its associated value of the Gibbs' free enthalpy are presented in the following table 4.2. The complete results' files of the three cases can be looked up in Appendix C.

The values of the equilibrium compositions are similar to the steady–state values of the number of moles gained by DESIRE simulation runs. However, the values of the Gibbs' free enthalpy differ from the values computed from equ.(4.205). It is still open research to explain these divergences.

Finally, the advantages and disadvantages of the combustion program which has been used without exception to generate data of chemical reactions under

Table 4.2: Essential results received by the "Combustion Program"

case	x_{Br_2}	$x_{Br\bullet}$	x_{H_2}	$x_{H\bullet}$
(p,T)	$3.9328E-3$	$1.9007E-5$	$4.4624E-6$	$< 1.0E-12$
(V,T)	$3.9424E-3$	$7.3788E-9$	$4.4517E-6$	$< 1.0E-12$
(S,T)	$3.9379E-3$	$< 1.0E-12$	$< 1.0E-12$	$< 1.0E-12$

Table 4.3: Essential results received by the "Combustion Program" (Table 2 continued)

case	x_{HBr}	entropy [J/K]	enthalpy [J]	G [J]
(p,T)	$9.9604E-1$	$2.276E-5$	$-2.114E-3$	$-2.032E-2$
(V,T)	$9.9605E-1$	$9.745E-6$	$-2.114E-3$	$-9.910E-3$
(S,T)	$9.9606E-1$	0.0	$-4.475E-3$	$-4.475E-3$

certain thermodynamic conditions are to be treated. In contrast to the Bond graph model the mass balance is coupled to the energy balance, i.e. the number of moles are computed simultaneously from equations involving both the Gibbs/Helmholtz free energy. Since it is not necessary to know the intermediate reaction steps but only the global reactions, one can consider highly complicated reactions. Also, there is no need to know the reaction coefficients k_i which are often not available. This program only provides the equilibrium state of a reaction rather that the dynamic case which clearly represents a big disadvantage. This program does not distiguish between the different forms of energy as the Bond graph model certainly does. In summary, this program package can only help to validate the equilibrium composition of reactions which, of course, has already been done in Section 4.5.1, and thus, does not provide new knowledge.

A program package to validate the dynamic data has not been found.

Chapter 5

Conclusions

The Bond graph methodology that has proven to be a very powerful tool to model the dynamical behavior of all kinds of physical systems, its implementation into the modeling language DYMOLA, and finally the compilation of DYMOLA programs to the simulation language DESIRE have been demonstrated in this project.

The modeling of chemical reaction systems has been presented using a new application of the Bond graph technique. Thereby, the two power balance equations, the kinetic equations, the state equation of ideal gases, and the input constraints — as the case–study of this project — of either constant volume and temperature, constant pressure and temperature, or constant entropy and pressure have served as basis for the design of a Bond graph model using both existing and newly designed Bond graph elements.

The absence of the inertia element in chemical reaction systems can be considered one of the reasons why the chemical sciences have divided into two camps, the chemical kinetics and the chemical thermodynamics. This interpretation has been given by my advisor, Dr. François Cellier.

Interestingly, the second power balance equation (equ.(4. 99) could not be evaluated in terms of component reactions due to the fact that the matrix M of equation (4. 152) happened to be singular.

There are some problems unsolved:

1. The Gibbs' free energy G of the isochoric–isothermic case differs from the

value obtained in equ.(4.205).

2. The values of the Gibbs' free energy G computed from the "Computer Program for Calculation of Complex Chemical Equilibrium Compositions, Rocket Performance, Incident and Reflected Shock, and Chapman–Jouget Detonation" [59] differ from the value obtained in equ.(4.205) and they are not equal to each other.

3. If in the isentropic–isobaric case the stationary value of the entropy S is different from zero value, the simulation will be unstable.

4. Considering the equation of state for the initial condition of the total volume V, the total volume can be calculated for zero time in the isobaric–isothermic and isochoric–isothermic cases as $V_{tot} = (n_{tot} \cdot R \cdot T)/p$. Simulation results are included in Appendix F. The reaction enthalpy ΔG differs from the value received from equ.(2.205), too.

This project can be used as a basis for future research on the application of Bond graphs to complex chemical reactions as well as to the combination of chemical sytems with all kinds of physical systems. The following research topics are suggested for the future:

1. Modeling of chemical reactions under all kinds of input constraints.

2. Modeling chemical reactions in the borderline cases of electrochemistry, photochemistry, etc.

3. Designing a library of Bond graph elements describing all types of instruments that are used in chemical engineering, such as elements for destillation columns, chemical reactors (for example: CSTR and tubular reactor), pipes, etc. Thus, for any application in chemical engineering elements of this library simply would have to be plugged together like a modular construction system.

   Currently, one of Dr. François Cellier's Ph. D. students is studying this topic.

References

[1] Cellier, François E. Cellier (1990): "Hierarchical Non–Linear Bond Graphs: A Unified Methodology for Modeling Complex Physical Systems", *Keynote address to "European Simulation Multiconference"*, Erlangen–Nuremberg, FRG.

[2] Cellier, François E. Cellier (1990): "Continuous System Modeling and Simulation, *ECE472/572 class notes*, Dept. of Electrical and Computer Engineering, University of Arizona, Tucson.

[3] Cellier, François E. (1986): "Combined Continuous/Discrete System Simulation: Applications, Techniques, and Tools", *Proceedings of Winter Simulation Conference*, edited by J. R. Wilson, *et al.*, pp. 858–865.

Bond Graphs:

[4] Blundell, Alan (1982): *Bond Graphs for Modelling Engineering Systems*, Ellis Horwood Publishers, Chichester, United Kingdom, and Halsted Press, New York.

[5] Bos, A.M.; and Breedveld, Pieter C. (1985): "Update of the Bond Graph Bibliography", *J. Franklin Institute*, **319**(1/2), pp.269–286.

[6] Breedveld, Pieter C. (1982): "Thermodynamic Bond Graphs and the Problem of Thermal Inertance", *J. Franklin Institute*, **314**(1), pp.15–40.

[7] Breedveld, Pieter C. (1984): ***Physical Systems Theory in Terms of Bond Graphs***, Ph.D. Dissertation, Technical University Twente, Enschede, The Netherlands.

[8] Dixhoorn, Jan J. van (1982): "Bond Graphs and the Challenge of a Unified Modelling Theory of Physical Systems", in: ***Progress in Modelling and Simulation***, (F. E. Cellier, ed.), Academic Press, London, pp.207–245.

[9] Gebben, Vernon D. (1979): "Bond Graph Bibliography", *J. Franklin Institute*, **308**(3), pp.361–369.

[10] Karnopp, Dean C.; and Ronald C. Rosenberg (1974): *System Dynamics; A Unified Approach*, John Wiley, New York.

[11] Oster, George F.; and Perelson, Alan S. (1974): "Chemical Reaction Networks", *IEEE Trans. Circuits and Systems*, **CAS–21**(6), pp.709–721.

[12] Paynter, Henry M. (1961): *Analysis and Design of Engineering Systems*, M.I.T. Press, Cambridge, MA.

[13] Thoma, Jean U. (1975): "Entropy and Mass Flow for Energy Conversion", *J. Franklin Institute*, **299**(2), pp.89–96.

[14] Thoma, Jean (1976): "Simulation, Entropy Flux and Chemical Potential", *BioSystems*, **8**, pp. 1–9.

[15] Thoma, Jean; and Atlan, Henri (1985): "Osmosis and Hydraulics by Network Thermodynamics and Bond Graphs", *J. Franklin Institute*, **319**(1/2), pp. 217–226.

[16] Wellstead, P.E (1979): "Introduction to Physical System Modelling" Academic Press, 1979.

Majuda
ISBN 0-12-744380-0

Software Packages, DYMOLA, DESIRE:

[16] Broenink, Jan F. (1984): ***Computer-Aided Physical-Systems Modeling and Simulation: A Bond-Graph Approach***, Ph. D. Dissertation, Universiteit Twente, Enschede, The Netherlands.

[17] Elmqvist, Hilding (1975): ***SIMNON - An Interactive Simulation Program for Non-linear Systems — User's Manual***, Report CODEN: LUTFD2/(TFRT-7502), Dept. of Automatic Control, Lund Institute of Technology, Lund, Sweden.

[18] Elmqvist, Hilding (1978): ***A Structured Model Language for Large Continuous Systems***, Ph.D. Thesis, Report CODEN: LUTFD2/(TRFT-1015), Dept. of Automatic Control, Lund Institute of Technology, Lund, Sweden.

[19] Elmqvist, Hilding (1982): "A Graphical Approach to Documentation and Implementation of Control Systems", ***Proc. 3rd IFAC/IFIP Symposium on Software for Computer Control (SOCOCO'82)***, Madrid, Spain, Pergamon Press, Oxford, U.K.

[20] Granda, Jose J. (1982): ***Computer Aided Modeling Program (CAMP): A Bond Graph Preprocessor for Computer Aided Design and Simulation of Physical Systems Using Digital Simulation Languages***, Ph.D. Dissertation, Dept. of Mechanical Engineering, University of California, Davis.

[21] Granda, Jose J. (1985): "Computer Generation of Physical System Differential Equations Using Bond Graphs", ***J. Franklin Institute***, **319**(1/2), pp.243–255.

[22] Granda, Jose J.; and Pourrahimi, F. (1985): "Computer Graphic Techniques for the Generation and Analysis of Physical System Models", in: ***Artificial Intelligence, Graphics, and Simulation***, Proceedings of the Western Simulation MultiConference, (G. Birtwistle, ed.), SCS Publishing, pp. 70–75.

[23] Korn, Granino A. (1989): ***Interactive Dynamic System Simulation***, McGraw–Hill, New York.

[24] Korn, Granino A. (1989): ***NEUNET/DESIRE User's Manual***, 6801 E. Opata Rd., Tucson, AZ 85715.

[25] The MathWorks, Inc. (1988): it Pro–Matlab with System Identification Toolbox and Control Toolbox - User Manual, Version 3.30, 21 Eliot St., South Natick, MA 01760.

[26] Mitchell, Edward E.L.; and Gauthier, Joe S. (1986): ***ACSL: Advanced Continuous Simulation Language — User Guide/Reference Manual***, Mitchell & Gauthier Assoc., P.O.Box 685, Concord, MA 01742.

[27] Ronald C. Rosenberg (1974): ***A User's Guide to ENPORT-4***, John Wiley, New York.

[28] RosenCode Associates, Inc. (1989): ***The ENPORT Reference Manual***, RosenCode Associates, Inc., 200 N. Capitol Bldg., Lansing, MI 48933.

[29] Systems Control Technology, Inc. (1985): ***CTRL–C, A Language for the Computer–Aided Design of Multivariable Control Systems, User's Guide***, 2300 Geng Rd., P.O.Box 10180, Palo Alto, CA 94303.

[30] Wang, Qingsu (1989): ***Management of Continuous System Models in DEVS–Scheme: Time Windows for Event Based Control***, M. Thesis, Dept. of Electrical and Computer Engineering, The University of Arizona, Tucson.

Chemical Kinetics and Thermodynamics:

[31] Balzhiser, Richard. R.; Samuels, Michael R.; and Eliassen, John D. (1972): ***Chemical Engineering Thermodynamics***, Prentice Hall, New Jersey.

[32] Carver, Mike B.; Hanley, D. H.; and Chaplin, K. R. (1978): ***MACKSIM: Mass Action Chemical Kinetics Simulation User's Manual***, Report: AECL–6413, Atomic Energy of Canada, Ltd., Chalk River Nuclear Laboratories, Chalk River, Ontario, Canada.

[33] Christen, Hans Rudolph (1979): ***Thermodynamik und Kinetik chemischer Reaktionen***, Studienbücher Chemie, Diesterweg; Salle; and Sauerländer Verlag, Frankfurt a. M., FRG; Aarau, Switzerland.

[34] Deuflhard, Peter; Bader, G.; and Nowak, U. (1980): ***LARKIN — A Software Package for the Numerical Simulation of LARge Systems Arising in Chemical Reaction KINetics***, Report 100, University of Heidelberg, Institut für Angewandte Mathematik, Im Neuheimer Feld 293, D–6900 Heidelberg 1, FRG.

[35] Eigenberger, G. (1989): "Chemiereaktortechnik I", ***class notes***, Institut für Chemische Verfahrenstechnik, Universität Stuttgart, Stuttgart, FRG.

[36] Emanuel, N.M.; and Knorre, D.G. (1973): ***Chemical Kinetics: Homogeneous Reactions***, John Wiley & Sons, Chichester.

[37] Hill, Charles G., Jr. (1977): ***An Introduction to Chemical Engineering Kinetics & Reactor Design***, John Wiley & Sons, N.Y.

[38] Hoppensteadt, Frank C. P.; Alfeld; and Aiken, Richard (1981): "Numerical Treatment of Rapid Chemical Kinetics by Perturbation and Projection Methods", in: ***Modelling of Chemical Reaction Systems***, (Klaus H. Ebert, Peter Deuflhard, and Willi Jäger, eds.), Springer Verlag, Berlin, Series in Chemical Physics, **18**, pp. 31–37.

[39] Klotz, Irving M.; and Rosenberg, Robert M. (1986), ***Chemical Thermodynamics: Basic Theory and Methods, 4th Ed.***, Benjamin/Cummings Publ., Menlo Park, CA.

[40] Laidler, Keith J. (1987): ***Chemical Kinetics, 3rd Ed.***, Harper & Row, New York.

[41] Latham, Joseph L. (1962): ***Elementary Reaction Kinetics***, Butterworth & Co., London, U.K.

[42] Mahan, Bruce H. (1965): ***University Chemistry***, Addison–Wesley, Reading, MA.

[43] Moore, John W.; Pearson, Ralph G.;, and Frost, Arthur A. (1981): ***Kinetics and Mechanism***, John Wiley & Sons, New York.

[44] Nicholas, John (1976): ***Chemical Kinetics: A modern Survey of Gas Reactions***, Harper & Row, London, U.K.

[45] Patterson, Gary K.; and Rozsa, R. B. (1978): ***DYNSYL: A General–Purpose Dynamic Simulator for Chemical Processes***, Report: UCRL–52561, Lawrence Livermore Laboratory, Livermore, CA 94550.

[46] Perry, Robert H.; and Chilton, Cecil H. (eds.) (1973): ***Chemical Engineers' Handbook***, 5th Edition, McGraw–Hill, New York.

[47] Rock, Peter A. (1983): ***Chemical Thermodynamics***, University Science Books, Mill Valley, CA.

[48] Smith, Joseph M. (1981): ***Chemical Engineering Kinetics, 3rd Ed.***, McGraw–Hill, New York.

[49] Steinfeld, Jeffrey I.; Francisco, Joseph S.; and Hase, William L. (1989): ***Chemical Kinetics and Dynamics***, Prentice Hall, Englewood Cliffs, N.J.

[50] Westererp, K.A.; Van Swaaij, W.P.M.; and Beenackeers, A.A.C.M. (1984): *Chemical Reactor Design and Operation*, John Wiley & Sons, N.Y.

Data for Hydrogen–Bromine Reaction:

[51] Barin, Ihsan; Knacke, Ottmar (1973): *Thermochemical Properties of Inorganic Substances*, Springer Verlag, Berlin, FRG.

[52] Bawn, C.E.H.; and White, A.G. (1951): "Conditions: in Aqueous Solution at Room Temperature (23°C)", *J. Chem. Soc.*, p.339.

[53] Bodenstein, M.; and Luetkemeyer, H. (1924): *Z.Physik.Chem, 114*, p.208.

[54] Bodenstein, M.; and Jung, G. (1926): *Z.Physik.Chem, 121*, p.127.

[55] Chase, M.W., Jr.; Davies, C.A.; Downey, J.R., Jr.; Frurip, D.J.; McDonald, R.A.; and Syverud, A.N.(1985): "JANAF Thermochemical Tables, Trd. ed.; Part I, Al–Co", *J. Physical and Chemical Reference Data*, Dow Chemical Co., Midland, Michigan.

[56] Dean, John A. *ed.* (1973): *Lange's Handbook of Chemistry* , McGraw–Hill, New York.

[57] Kerr, J. Alistair; and Moss, Stephen J. (1981): *Handbook of Bimolecular and Thermolecular Gas Reactions*, CRC Press, Boca Raton, FL.

[58] Jost, W.(1929): *Z.Physik.Chem., B3*, p.95.

Computer Program to Evaluate Chemical Equilibrium

[59] Gordon, Sanford; and McBride, Bonnie J. (1971): ***Computer Program for Calculation of Complex Chemical Equilibrium Compositions, Rocket Performance, Incident and Reflected Shocks, and Chapman–Jouget Detonations***, NASA SP–273.

General Thermodynamics

[60] Lavenda, Bernard H. (1985): ***Nonequilibrium Statistical Thermodynamics***, John Wiley, New York.

[61] Lawden, Derek F. (1987): ***Principles of Thermodynamics and Statistical Mechanics***, John Wiley, New York.

[62] Prigogine, Ilya (1967): ***Thermodynamics of Irreversible Processes***, 3rd Edition, John Wiley Interscience, New York.

[63] Stowe, Keith S. (1984): ***Introduction to Statistical Mechanics and Thermodynamics***, John Wiley, New York.

[64] Truesdell, Clifford (1984): ***Rational Thermodynamics***, 2nd Edition, Springer, New York.

[65] Yao, Y. L. (1981): ***Irreversible Thermodynamics***, Science Press, Beijing, distributed by: Van Nostrand Reinhold, New York.

Appendix A

Programs for chapter 3

This appendix describes all programs to which chapter 3 refers. It includes the DYMOLA model library, the DYMOLA and DESIRE programs and intermediate results of the DYMOLA preprocessor of the RLC–circuit, and a simulation graph of the RLC–circuit.

A.1 The DYMOLA Model Library of the Basic Bond Graphs

```
model type SE

    cut A(e/.)
    terminal E0
    E0 = e
end

model type SF
    cut A(./ - f)
    terminal F0
    F0 = f
end

model type R
    cut A(e/f)
    parameter R = 1.0
    R * f = e
end

model type C
    cut A(e/f)
    parameter C = 1.0
    C*der(e) = f
end

model type I
    cut A(e/f)
    parameter I = 1.0
    I*der(f) = e
end

model type TF
    cut A(e1/f1) B(e2/ - f2)
    main cut C[A B]
    main path P < A - B >
    parameter m = 1.0
    e1 = m * e2
    f2 = m * f1
```

```
end

model type GY
      cut A(e1/f1) B(e2/ - f2)
      main cut C[A B]
      main path P < A - B >
      parameter r = 1.0
      e1 = r * f2
      e2 = r * f1
end
```

A.2 The Main Program of the RLC–Circuit

```
{ RLC - circuit }

@r.lib
@c.lib
@i.lib
@se.lib
@bond.lib

{ ***************************************************************** }

model RLC

   submodel (SE) U0
   submodel (R) R1(R = 8.0), R2(R = 7.0)
   submodel (I) L1(I = 1.0), L2(I = 2.0)
   submodel (C) C1(C = 2.0E-1)
   submodel (bond) B1,B2,B3

   node v1,iL2,uL2,v2
   input u
   output y1,y2
   parameter dirty = 1.0

   connect U0 at v1, ->
           L1 at v2, ->
           L2 at uL2, ->
           C1 at v2, ->
           R1 at v2, ->
           R2 at v2, ->
           B1 from v1 to iL2, ->
           B2 from iL2 to uL2, ->
           B3 from iL2 to v2

   U0.E0 = u
   dirty*y1 = u
   dirty*y2 = C1.e
end
```

A.3 The Simulation Control Program of the RLC–Circuit

```
{ RLC - circuit }

cmodel

   simutime 30.0
   step 1.0E-2
   commupoints 1000

   input 1,u(depend, sin(0.5*t))

   ctblock
        {Gear-Integration Algorithm}
        irule 15
        ERMAX = 1.0E-3
        DTMAX = 0.01

        disconnect 1 | connect 'RLC.tab' as output 1

        scale = 1.5
        XCCC = 1
        label TRY
        drun | if XCCC < 0 then XCCC=-XCCC | scale = 2*scale | go to TRY
                                             else proceed

        disconnect 1 | connect 'user' as output 1
   ctend

   outblock
       OUT
       type y1, y2
   outend

   end
```

A.4 (a) RLC.eq1, (b) RLC.sor, (c) RLC.sov

```
R1                  R*f = e
C1                  C*dere = f
L1                  I*derf = e
U0                  dirty*E0 = e
R2                  R*f = e
L2                  I*derf = e
RLC                 U0.E0 = u
                    dirty*y1 = u
                    dirty*y2 = C1.e
                    U0.e = B1.x
                    L2.e = B2.y
                    L2.f = B2.x
                    B2.x = B3.x
                    B1.y = B2.x
                    B3.y + B2.y = B1.x
                    R2.e = B3.y
                    R1.e = R2.e
                    C1.e = R1.e
                    L1.e = C1.e
                    R2.f + R1.f + C1.f + L1.f = B3.x
```

(a)

```
RLC                 C1.e = [R1.e]
R1                  R*[f] = e
RLC                 R1.e = [R2.e]
R2                  R*[f] = e
RLC                 L2.f = [B2.x]
                    B2.x = [B3.x]
                    R2.f + R1.f + [C1.f] + L1.f = B3.x
C1                  C*[dere] = f
RLC                 [L1.e] = C1.e
L1                  I*[derf] = e
RLC                 [U0.E0] = u
U0                  dirty*E0 = [e]
RLC                 U0.e = [B1.x]
                    [B1.y] = B2.x
                    R2.e = [B3.y]
                    B3.y + [B2.y] = B1.x
                    [L2.e] = B2.y
L2                  I*[derf] = e
RLC                 dirty*[y1] = u
                    dirty*[y2] = C1.e
```

(b)

```
R1                  f = C1.e/R
R2                  f = C1.e/R
RLC                 C1.f = L2.f - (R2.f + R1.f + L1.f)
C1                  dere = f/C
L1                  derf = C1.e/I
U0                  B1.x = dirty*RLC.u
RLC                 B2.y = B1.x - C1.e
L2                  derf = B2.y/I
RLC                 y1 = u/dirty
                    y2 = C1.e/dirty
```

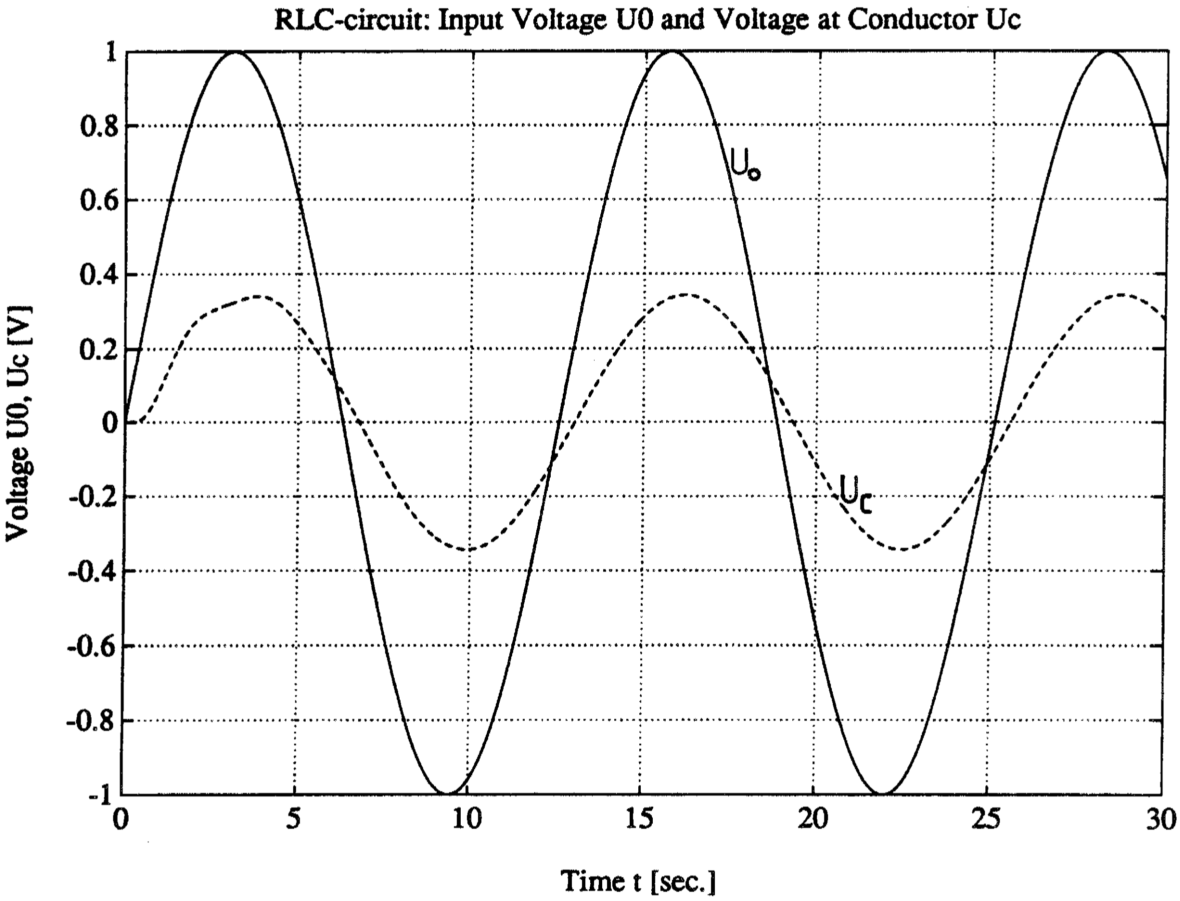
(c)

A.5 DESIRE Program of the RLC–Circuit

```
-----------------------------------------
-- CONTINUOUS SYSTEM RLC
-----------------------------------------
-- STATE C1$e L1$f L2$f
-- DER dC1$e  dL1$f  dL2$f
-- OUTPUT y1 y2
-- INPUT u
-- PARAMETERS and CONSTANTS:
R1$R=8.0
C=2.0E-1
L1$I=1.0
U0$diy=1.0
R2$R=7.0
L2$I=2.0
RC$diy=1.0
-- INITIAL VALUES OF STATES:
C1$e=0
L1$f=0
L2$f=0

-----------------------------------------
TMAX=30.0 | DT=1.0E-2 | NN=1000
        irule 15
        ERMAX = 1.0E-3
        DTMAX = 0.01

        disconnect 1 | connect 'RLC.tab' as output 1

        scale = 1.5
        XCCC = 1
        label TRY
        drun | if XCCC < 0 then XCCC=-XCCC | scale = 2*scale | go to TRY
                                              else proceed

        disconnect 1 | connect 'user' as output 1
-----------------------------------------
DYNAMIC
-----------------------------------------
u = sin(0.5*t)
-- Submodel: R1
R1$f = C1$e/R1$R
-- Submodel: R2
R2$f = C1$e/R2$R
-- Submodel: RLC
C1$f = L2$f - (R2$f + R1$f + L1$f)
-- Submodel: C1
d/dt C1$e = C1$f/C
-- Submodel: L1
d/dt L1$f = C1$e/L1$I
```

```
-- Submodel: U0
B1$x = U0$diy*u
-- Submodel: RLC
B2$y = B1$x - C1$e
-- Submodel: L2
d/dt L2$f = B2$y/L2$I
-- Submodel: RLC
y1 = u/RC$diy
y2 = C1$e/RC$diy
----------------------------------------
       OUT
       type y1, y2
----------------------------------------
/--
/PIC 'rlc.PRC            '
/--
```

A.6 Graph of a Simulation of the RLC–Circuit



Appendix B

Programs for Chapter 4

This appendix contains all programs to which Chapter 4 refers, excluding the programs of the "combustion program" which are shown in Appendix C.

B.1 Hydrogen–Bromine Reaction: Isochoric-isothermic Case

Following, the DYMOLA main program "*CT.dym*" including all the model types, the simulation control program "*CT.ctl*", and its generated DESIRE–program "*CT.des*" of the hydrogen–bromine reaction under isochoric–isothermic conditions are listed. Additionally, there are the DYMOLA main program "*CTwish.dym*" and the simulation control program "*CTwish.ctl*" that are listed presenting the desirable programs of DYMOLA in the future.

B.1.1 DYMOLA Main Program "*CT.dym*" Including All the Model Types

```
{ H2 -Br2 reaction isochoric and isothermic }

{ Bond Graph bond }

model type bond

  cut A (x / y) B (y / -x)
  main cut C [A B]
  main path P <A - B>

end


{ H2 - Br2 Reaction isochoric and isothermic }

model type Thermic

  cut thermtot( Temp/-Sdot )
  cut thermk1( Temp/Sdotk1 ), thermk2( Temp/Sdotk2 )
  cut thermk3( Temp/Sdotk3 ), thermk4( Temp/Sdotk4 )
  cut thermk5( Temp/Sdotk5 )
  terminal S
  parameter dirty = 1.0

  Sdot     = Sdotk1 + Sdotk2 + Sdotk3 + Sdotk4 + Sdotk5
  dirty*der(S) = Sdot

end

{ H2 - Br2 Reaction isochoric and isothermic }

model type Chemical

     terminal nBr2, nBr, nH2, nH, nHBr
     local ntot

     ntot  = nBr2 + nBr + nH + nH2 + nHBr

end


{ H2 - Br2 Reaction isochoric and isothermic }

model type Pneumatic

  cut pneutot( pdot/V )
  cut pneu1( pdotBr2/-V ), pneu2( pdotBr/-V )
  cut pneu3( pdotH2/-V ), pneu4( pdotH/-V )
```

```
   cut pneu5( pdotHBr/-V )
   terminal p
   parameter dirty = 1.0

   pdot          = pdotBr2 + pdotBr + pdotH2 + pdotH + pdotHBr
   dirty*der(p) = pdot

 end


 { H2 - Br2 Reaction isochoric and isothermic }

 model type Chpower

    terminal mudBr2, mudBr, mudH2, mudH, mudHBr
    terminal muBr2, muBr, muH2, muH, muHBr
    terminal xiBr2, xiBr, xiH2, xiH, xiHBr
    terminal nBr2, nBr, nH2, nH, nHBr
    terminal Gdot, G
    parameter dirty = 1.0

    Gdot = mudBr2*nBr2 + mudBr*nBr + mudH2*nH2 + mudH*nH + mudHBr*nHBr ->
          + muBr2*xiBr2 + muBr*xiBr + muH2*xiH2 + muH*xiH + muHBr*xiHBr
    dirty*der(G) = Gdot

 end


 { H2 - Br2 reaction isochoric and isothermic }

 model type TF

   cut A( mu1/xi1 ), B( mu2/-xi2 )
   main cut C [A B]
   main path P < A-B >
   parameter k = 1.0

   k*xi2 = xi1
   k*mu1 = mu2

 end


 { H2 - Br2 Reaction isochoric and isothermic }

 model type CS

   cut chem1(mu / xi), chem2(mudot / n)
   terminal Temp
   local mu1
   parameter R = 8.314, eps = 1.0E-15, dirty = 1.0, sc = 1.5

   sgn( sc - 1 )*mu1 = mu
   mudot*( n + eps ) = xi*R*Temp
   dirty*der(mu1) = sgn( sc - 1 )*mudot

 end
```

```
{ H2 - Br2 Reaction, isochoric and isothermic }

model type ChR1k1

  cut chem1( muk1/-xik1 ), thermk1( Temp/-Sdotk1 )
  main cut C[ chem1 ]
  terminal nBr2, V
  parameter R=8.314
  local k1, ak1

  ak1    = 1.39E+8*Temp**0.5*( 189243/( R*Temp ) )**1.97
  k1     = ak1*exp( -189243/( R*Temp ) )
  0.0    = Temp*Sdotk1 + muk1*xik1
  xik1   = k1*nBr2

end


{ H2 - Br2 Reaction, isochoric and isothermic }

model type ChR1k2

  cut chem1( muk2/-xik2 ), thermk2( Temp/-Sdotk2 )
  main cut C[ chem1 ]
  terminal nBr, K, V
  parameter R=8.314
  local k1, ak1, k2

  ak1     = 1.39E+8*Temp**0.5*( 189243/( R*Temp ) )**1.97
  k1      = ak1*exp( -189243/( R*Temp ) )
  k2      = k1/K
  0.0     = Temp*Sdotk2 + muk2*xik2
  xik2    = k2*nBr*nBr/V

end


{ H2 - Br2 Reaction, isochoric and isothermic }

model type ChR1k3

  cut chem1( muk3/-xik3 ), thermk3( Temp/-Sdotk3 )
  main cut C[chem1]
  terminal nH2, nBr, V
  parameter R=8.314
  local k3

  k3     = ( 10**11.43 )*exp( -82400/( R*Temp ) )
  0.0    = Temp*Sdotk3 + muk3*xik3
  xik3   = k3*nH2*nBr/V

end
```

```
{ H2 - Br2 Reaction, isochoric and isothermic }

model type ChR1k4

  cut chem1( muk4/-xik4 ), thermk4( Temp/-Sdotk4 )
  main cut C[chem1]
  terminal nHBr, nH, V
  parameter R=8.314, m=10.0
  local k4, k5

  k5     = 10.0**11.97*exp( -149800.0/( R*Temp ) )
  k4     = k5/m
  0.0    = Temp*Sdotk4 + muk4*xik4
  xik4   = k4*nHBr*nH/V

end


{ H2 - Br2 Reaction, isochoric and isothermic }

model type ChR1k5

  cut chem1( muk5/-xik5 ), thermk5( Temp/-Sdotk5 )
  main cut C[chem1]
  terminal nBr2, nH, V
  parameter R=8.314
  local k5

  k5     = 10.0**11.97*exp( -149800.0/( R*Temp ) )
  0.0    = Temp*Sdotk5 + muk5*xik5
  xik5   = k5*nBr2*nH/V

end


{ H2 - Br2 Reaction, isochoric and isothermic }

model type ChR2B2

  cut chem2( mudotBr2/-nBr2 ), pneul( pdotBr2/V )
  main cut C[ chem2 ]
  terminal xiBr2
  parameter dirty = 1.0

  pdotBr2*V = mudotBr2*nBr2
  dirty*der(nBr2) = xiBr2

end
```

```
{ H2 - Br2 Reaction, isochoric and isothermic }

model type ChR2Br

  cut chem2( mudotBr/-nBr ), pneu2( pdotBr/V )
  main cut C[ chem2 ]
  terminal xiBr
  parameter dirty = 1.0

  pdotBr*V  = mudotBr*nBr
  dirty*der(nBr) = xiBr

end


{ H2 - Br2 Reaction, isochoric and isothermic }

model type ChR2H2

  cut chem2( mudotH2/-nH2 ), pneu3( pdotH2/V )
  main cut C[ chem2 ]
  terminal xiH2
  parameter dirty = 1.0

  pdotH2*V   = mudotH2*nH2
  dirty*der(nH2) = xiH2

end


{ H2 - Br2 Reaction, isochoric and isothermic }

model type ChR2H

  cut chem2( mudotH/-nH ), pneu4( pdotH/V )
  main cut C[ chem2 ]
  terminal xiH
  parameter dirty = 1.0

  pdotH*V   = mudotH*nH
  dirty*der(nH) = xiH

end


{ H2 - Br2 Reaction, isochoric and isothermic }

model type ChR2HB

  cut chem2( mudotHBr/-nHBr ), pneu5( pdotHBr/V )
  main cut C[ chem2 ]
  terminal xiHBr
  parameter dirty = 1.0
  pdotHBr*V = mudotHBr*nHBr
  dirty*der(nHBr) = xiHBr

end
```

```
{--------------------------------------------------------------------------
{ *********************  MAIN PROGRAM  *************************************
{--------------------------------------------------------------------------

model reaction1

{ ******  Calling the Submodels  ****** }

  submodel ChR1k1
  submodel ChR1k2
  submodel ChR1k3
  submodel ChR1k4
  submodel ChR1k5
  submodel ChR2B2   (ic nBr2 = 0.5E-7)
  submodel ChR2Br   (ic nBr = 0.0)
  submodel ChR2H2   (ic nH2 = 0.5E-7)
  submodel ChR2H    (ic nH = 0.0)
  submodel ChR2HB   (ic nHBr = 0.0)
  submodel (CS)          CSBr2 (sc = 0.5) (ic mu1 = 204493.0) ->
                         CSBr  (sc = 0.5) (ic mu1 = 53772.8)  ->
                         CSH2  (sc = 0.5) (ic mu1 = 129023.0) ->
                         CSH   (sc = 1.5) (ic mu1 = 106772.0)  ->
                         CSHBr (sc = 0.5) (ic mu1 = 226828.0)

  submodel Pneumatic   (ic p = 1.0E+5)
  submodel Thermic     (ic S = 441.51)
  submodel Chpower     (ic G = 0.0)
  submodel Chemical
  submodel (TF)        TF1(k = 0.5), TF2(k = 2.0)
  submodel (bond)      B11, B1Br2, B1Br, B21, B2Br2, B2Br, B31
  submodel (bond)      B3H2, B3H, B3HBr, B3Br, B41, B4H, B4HBr
  submodel (bond)      B4Br, B4H2, B51, B5Br, B5Br2, B5H, B5HBr


{ ****** I/O Declaration ****** }

  input   Temp0, V0
  output  y1, y2, y3, y4, y5


{ ****** Declaration of the Variable Type ****** }

  parameter dirty = 1.0
  local   K900


{ ****** Declaration of the nodes ****** }

  node    n11, n21, n31, n41, n51, n1Br, n2Br, nBr2, nBr, ->
          nH2, nH, nHBr, n1, n2, n3, n4, n5
```

```
{ +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }
{                   CONNECTIONS OF THE BOND GRAPH ELEMENTS                 }
{ +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }

{ ****** Connections of the ChR1-elements and of the bonds }
        { attached to either the 0-junctions or the 1-junctions ****** }


  connect ChR1k1:chem1 at   n11,     ->
          B11   from n11  to n1,   ->
          B1Br2 from nBr2 to n1,   ->
          B1Br  from n1   to n1Br, ->
          TF1   from n1Br to nBr

  connect ChR1k2:chem1 at   n21,     ->
          B21   from n21  to n2,   ->
          B2Br  from n2Br to n2,   ->
          B2Br2 from n2   to nBr2, ->
          TF2   from nBr  to n2Br

  connect ChR1k3:chem1 at   n31,     ->
          B31   from n31  to n3,   ->
          B3HBr from n3   to nHBr, ->
          B3H   from n3   to nH,  ->
          B3H2  from nH2  to n3,   ->
          B3Br  from nBr  to n3

  connect ChR1k4:chem1 at   n41,     ->
          B41   from n41  to n4,   ->
          B4H   from nH   to n4,   ->
          B4HBr from nHBr to n4,   ->
          B4Br  from n4   to nBr,  ->
          B4H2  from n4   to nH2

  connect ChR1k5:chem1 at   n51,     ->
          B51   from n51  to n5,   ->
          B5Br  from n5   to nBr,  ->
          B5Br2 from nBr2 to n5,   ->
          B5H   from nH   to n5,   ->
          B5HBr from n5   to nHBr


{ ****** Connection of the ChR2-elements to the CS-elements ****** }

 connect ChR2B2:chem2  at CSBr2:chem2, ->
         ChR2Br:chem2  at CSBr:chem2,  ->
         ChR2H2:chem2  at CSH2:chem2,  ->
         ChR2H:chem2   at CSH:chem2,   ->
         ChR2HB:chem2  at CSHBr:chem2
```

```
{ ****** Connection of the CS-elements to the 0-junctions ****** }

 connect CSBr2:chem1 at nBr2, ->
         CSBr:chem1  at nBr,  ->
         CSH2:chem1  at nH2,  ->
         CSH:chem1   at nH,   ->
         CSHBr:chem1 at nHBr

{ ****** Connection of the ChR2-elements to the Pneumatic-element ****** }

  connect ChR2B2:pneu1 at Pneumatic:pneu1, ->
          ChR2Br:pneu2 at Pneumatic:pneu2, ->
          ChR2H2:pneu3 at Pneumatic:pneu3, ->
          ChR2H:pneu4 at Pneumatic:pneu4, ->
          ChR2HB:pneu5 at Pneumatic:pneu5

{ ****** Connection of the ChR1-elements to the Thermic-element ****** }

  connect ChR1k1:thermk1 at Thermic:thermk1, ->
          ChR1k2:thermk2 at Thermic:thermk2, ->
          ChR1k3:thermk3 at Thermic:thermk3, ->
          ChR1k4:thermk4 at Thermic:thermk4, ->
          ChR1k5:thermk5 at Thermic:thermk5

{ +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }
{             Computation of the Free Energy of Formation   and             }
{             Reading out the equilibrium constant K                        }
{ +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }

     func K900  = KBr2Bt(Temp0)


{ +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }
{                     Setting the Modulating Signals                        }
{ +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }


{ ****** V - Terminal of the ChR1-elements and the V - input ****** }

  ChR1k1.V = V0
  ChR1k2.V = V0
  ChR1k3.V = V0
  ChR1k4.V = V0
  ChR1k5.V = V0
```

```
{ ****** ni - Terminal of the ChR1-elements and the ChR2-elements ****** }

  ChR1k1.nBr2   = ChR2B2.nBr2

  ChR1k2.nBr    = ChR2Br.nBr
  ChR1k2.K      = K900

  ChR1k3.nH2   = ChR2H2.nH2
  ChR1k3.nBr   = ChR2Br.nBr

  ChR1k4.nHBr = ChR2HB.nHBr
  ChR1k4.nH   = ChR2H.nH

  ChR1k5.nBr2 = ChR2B2.nBr2
  ChR1k5.nH   = ChR2H.nH


{ ****** xi-Terminal of the ChR2-elements and the CS-elements ****** }

  ChR2B2.xiBr2 = CSBr2.xi
  ChR2Br.xiBr  = CSBr.xi
  ChR2H2.xiH2  = CSH2.xi
  ChR2H.xiH    = CSH.xi
  ChR2HB.xiHBr = CSHBr.xi


{ ****** T - Terminal of the ChR2-elements and the T - input ****** }

  CSBr2.Temp = Temp0
  CSBr.Temp  = Temp0
  CSH2.Temp  = Temp0
  CSH.Temp   = Temp0
  CSHBr.Temp = Temp0


{ ****** mudoti - Terminal of the Chpower-element}
        { and the CS-elements ****** }

  Chpower.mudBr2  = CSBr2.mudot
  Chpower.mudBr   = CSBr.mudot
  Chpower.mudH2   = CSH2.mudot
  Chpower.mudH    = CSH.mudot
  Chpower.mudHBr  = CSHBr.mudot


{ ****** mui - Terminal of the Chpower-element}
        { and the CS-elements ****** }

  Chpower.muBr2 = CSBr2.mu
  Chpower.muBr  = CSBr.mu
  Chpower.muH2  = CSH2.mu
  Chpower.muH   = CSH.mu
  Chpower.muHBr = CSHBr.mu
```

```
{ ****** xii - Terminal of the Chpower-element}
        { and the CS-elements ****** }

  Chpower.xiBr2 = CSBr2.xi
  Chpower.xiBr  = CSBr.xi
  Chpower.xiH2  = CSH2.xi
  Chpower.xiH   = CSH.xi
  Chpower.xiHBr = CSHBr.xi


{ ****** ni - Terminal of the Chpower-element}
        { and the ChR2-elements ****** }

  Chpower.nBr2 = ChR2B2.nBr2
  Chpower.nBr  = ChR2Br.nBr
  Chpower.nH2  = ChR2H2.nH2
  Chpower.nH   = ChR2H.nH
  Chpower.nHBr = ChR2HB.nHBr


{ ****** ni - Terminal of the Chemical-element}
        { and the ChR2-elements ****** }

  Chemical.nBr2 = ChR2B2.nBr2
  Chemical.nBr  = ChR2Br.nBr
  Chemical.nH2  = ChR2H2.nH2
  Chemical.nH   = ChR2H.nH
  Chemical.nHBr = ChR2HB.nHBr


{ ****** T, V -Terminal of the Thermic-, }
       { Pneumatic-element and the T, V -inputs ****** }

  Thermic.Temp = Temp0
  Pneumatic.V  = V0


{ +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }
{          Connection of the ni - state variables with the outputs          }
{ +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }

  dirty*y1 = ChR2B2.nBr2/Chemical.ntot
  dirty*y2 = ChR2Br.nBr/Chemical.ntot
  dirty*y3 = ChR2H2.nH2/Chemical.ntot
  dirty*y4 = ChR2H.nH/Chemical.ntot
  dirty*y5 = ChR2HB.nHBr/Chemical.ntot

end
```

B.1.2 Simulation Control Program "*CT.ctl*"

```
{ H2 - Br2 reaction isobaric and isothermic }

{ ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }
{      Control Block specifying the Parameters of the Simulation         }
{              ( coded in the DYMOLA - Terminology )                     }
{ ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }


cmodel

  simutime    3.3
  step        1.0E-16
  commupoints 2001


{ ****** Specifying of the Simulation Inputs ****** }

  input 2, Temp0(depend, 800.0),  ->
           V0(depend, 1.0E-3)


{ ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }
{           Control - Block coded in the DESIRE - language               }
{ ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }

  ctblock


{ ****** Specifying the Integration  Algorithm, largest }
      { Time Step, and largest absolute Error ****** }

    { Gear-Algorithm }
    irule 15

    DTMAX=0.009
    ERMAX=1.0E-6


{ ****** Table of the Data to compute the free energy of formation ****** }

      dimension KBr2Bt[38]

      data      298.15, 300.0, 400.0, 500.0, 600.0, 700.0
      data      800.0, 900.0, 1000.0, 1100.0, 1200.0, 1300.0
      data      1400.0, 1500.0, 1600.0, 1700.0, 1800.0, 1900.0
      data      2000.0
      data      3.1623E-29, 7.7446E-29, 1.9543E-20, 2.2182E-15
      data      5.2844E-12, 1.3867E-9,  9.0782E-8,  2.3768E-6
      data      3.2509E-5,  2.7861E-4,  1.6788E-3,  7.6913E-3
      data      2.8510E-2,  8.8716E-2,  2.4044E-1,  5.8344E-1
      data      1.7947   ,  2.6061   ,  4.9431

      read KBr2Bt
```

```
{ ****** Routine to restart the simulation by doubling the scale factor }
       { after its stop due to the bounds exceeded ****** }

    scale = 1.0E-7
    XCCC  = 1
    label TRY
    drunr | if XCCC < 0 then XCCC = -XCCC | scale = 2*scale | go to TRY
                      else proceed
  ctend

{ ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }
{                         The Output Block                          }
{ ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }

  outblock
    OUT
    dispt y1, y2, y3, y4, y5
  outend

end
```

B.1.3 Generated DESIRE–Program "*CT.des*"

```
----------------------------------------
-- CONTINUOUS SYSTEM reaction1
----------------------------------------
-- STATE S G p CSr2$mu1 Ch2$nBr2 ChBr$nBr ChH2$nH2 ChR2H$nH
-- DER dS  dG  dp  dCSr$mu1  dCh2nBr2  dChB$nBr  dChH$nH2  dChRH$nH
-- STATE ChB$nHBr Cr$CBu11 CSH2$mu1 CSH$mu1 Cr$CBu12
-- DER dChBnHBr  dCr$Bu11  dCSH$mu1  dCSH$mu1  dCr$Bu12
-- OUTPUT y1 y2 y3 y4 y5
-- INPUT Temp0 V0
-- PARAMETERS and CONSTANTS:
Tc$dirty=1.0
Cr$C$ty1=1.0
Pc$dirty=1.0
TF1$k=0.5
CSBr2$R=8.314
CSr2$eps=1.0E-15
C2$C$ty1=1.0
CSBr2$sc=0.5
ChR1k1$R=8.314
ChR1k2$R=8.314
ChR1k3$R=8.314
ChR1k4$R=8.314
m=10.0
ChR1k5$R=8.314
C2$C$ty2=1.0
Cr$C$ty2=1.0
C2$C$ty3=1.0
CH$C$ty1=1.0
CB$dirty=1.0
CSBr$R=8.314
```

```
Cr$CBps1=1.0E-15
Cr$C$ty3=1.0
CSBr$sc=0.5
CSH2$R=8.314
CSH2$eps=1.0E-15
C2$C$ty4=1.0
CSH2$sc=0.5
CSH$R=8.314
CSH$eps=1.0E-15
CH$C$ty2=1.0
CSH$sc=1.5
CSHBr$R=8.314
Cr$CBps2=1.0E-15
Cr$C$ty4=1.0
CSHBr$sc=0.5
TF2$k=2.0
r1$dirty=1.0
-- INITIAL VALUES OF STATES:
S=441.51
G=0.0
p=1.0E+5
CSr2$mu1=204493.0
Ch2$nBr2=0.5E-7
ChBr$nBr=0.0
ChH2$nH2=0.5E-7
ChR2H$nH=0.0
ChB$nHBr=0.0
Cr$CBu11=53772.8
CSH2$mu1=129023.0
CSH$mu1=106772.0
Cr$CBu12=226828.0

----------------------------------------
TMAX=3.3 | DT=1.0E-16 | NN=2001
    irule 15
    DTMAX=0.009
    ERMAX=1.0E-6
      dimension KBr2Bt[38]
      data      298.15, 300.0, 400.0, 500.0, 600.0, 700.0
      data      800.0, 900.0, 1000.0, 1100.0, 1200.0, 1300.0
      data      1400.0, 1500.0, 1600.0, 1700.0, 1800.0, 1900.0
      data      2000.0
      data      3.1623E-29, 7.7446E-29, 1.9543E-20, 2.2182E-15
      data      5.2844E-12, 1.3867E-9,  9.0782E-8,  2.3768E-6
      data      3.2509E-5,  2.7861E-4,  1.6788E-3,  7.6913E-3
      data      2.8510E-2,  8.8716E-2,  2.4044E-1,  5.8344E-1
      data      1.7947    , 2.6061    , 4.9431
      read KBr2Bt
    scale = 1.0E-7
    XCCC  = 1
    label TRY
    drunr | if XCCC < 0 then XCCC = -XCCC | scale = 2*scale | go to TRY
                      else proceed
----------------------------------------
DYNAMIC
----------------------------------------
Temp0 = 800.0
V0 = 1.0E-3
```

```
-- Submodel: CSBr
CSBr$mu = sgn(CSBr$sc - 1)*Cr$CBul1
-- Submodel: TF1
TF1$mu1 = CSBr$mu/TF1$k
-- Submodel: CSBr2
CSBr2$mu = sgn(CSBr2$sc - 1)*CSr2$mu1
-- Submodel: reaction1
B11$x = TF1$mu1 - CSBr2$mu
-- Submodel: ChR1k1
Chk1$ak1 = 1.39E+8*Temp0^0.5*(189243/(ChR1k1$R*Temp0))^1.97
ChRk1$k1 = Chk1$ak1*exp(-189243/(ChR1k1$R*Temp0))
B1Br$x = ChRk1$k1*Ch2$nBr2
C1$Sdok1 = B11$x*B1Br$x/(-Temp0)
-- Submodel: TF2
TF2$mu2 = TF2$k*CSBr$mu
-- Submodel: reaction1
B21$x = CSBr2$mu - TF2$mu2
-- Submodel: ChR1k2
Chk2$ak1 = 1.39E+8*Temp0^0.5*(189243/(ChR1k2$R*Temp0))^1.97
ChRk2$k1 = Chk2$ak1*exp(-189243/(ChR1k2$R*Temp0))
-- Submodel: reaction1
func K900 = KBr2Bt,Temp0
-- Submodel: ChR1k2
k2 = ChRk2$k1/K900
TF2$xi2 = k2*ChBr$nBr*ChBr$nBr/V0
C2$Sdok2 = B21$x*TF2$xi2/(-Temp0)
-- Submodel: CSH
CSH$mu = sgn(CSH$sc - 1)*CSH$mu1
-- Submodel: CSHBr
CSHBr$mu = sgn(CSHBr$sc - 1)*Cr$CBul2
-- Submodel: CSH2
CSH2$mu = sgn(CSH2$sc - 1)*CSH2$mu1
-- Submodel: reaction1
B31$x = CSH$mu + CSHBr$mu - (CSBr$mu + CSH2$mu)
-- Submodel: ChR1k3
k3 = 10^11.43*exp(-82400/(ChR1k3$R*Temp0))
B3Br$y = k3*ChH2$nH2*ChBr$nBr/V0
C3$Sdok3 = B31$x*B3Br$y/(-Temp0)
-- Submodel: reaction1
B41$x = CSH2$mu + CSBr$mu - (CSHBr$mu + CSH$mu)
-- Submodel: ChR1k4
ChRk4$k5 = 10.0^11.97*exp(-149800.0/(ChR1k4$R*Temp0))
k4 = ChRk4$k5/m
B4H2$x = k4*ChB$nHBr*ChR2H$nH/V0
C4$Sdok4 = B41$x*B4H2$x/(-Temp0)
-- Submodel: reaction1
B51$x = CSHBr$mu + CSBr$mu - (CSH$mu + CSBr2$mu)
-- Submodel: ChR1k5
ChRk5$k5 = 10.0^11.97*exp(-149800.0/(ChR1k5$R*Temp0))
B5HBr$x = ChRk5$k5*Ch2$nBr2*ChR2H$nH/V0
C5$Sdok5 = B51$x*B5HBr$x/(-Temp0)
-- Submodel: Thermic
Sdot = C1$Sdok1 + C2$Sdok2 + C3$Sdok3 + C4$Sdok4 + C5$Sdok5
d/dt S = Sdot/Tc$dirty
-- Submodel: Chemical
ntot = Ch2$nBr2 + ChBr$nBr + ChR2H$nH + ChH2$nH2 + ChB$nHBr
-- Submodel: reaction1
CSBr2$xi = TF2$xi2 - (B5HBr$x + B1Br$x)
-- Submodel: CSBr2
mudotBr2 = CSBr2$xi*CSBr2$R*Temp0/(Ch2$nBr2 + CSr2$eps)
```

```
-- Submodel: TF2
TF2$xi1 = TF2$k*TF2$xi2
-- Submodel: TF1
TF1$xi2 = B1Br$x/TF1$k
-- Submodel: reaction1
CSBr$xi = B5HBr$x + B4H2$x + TF1$xi2 - (B3Br$y + TF2$xi1)
-- Submodel: CSBr
mudotBr = CSBr$xi*CSBr$R*Temp0/(ChBr$nBr + Cr$CBps1)
-- Submodel: reaction1
CSH2$xi = B4H2$x - B3Br$y
-- Submodel: CSH2
mudotH2 = CSH2$xi*CSH2$R*Temp0/(ChH2$nH2 + CSH2$eps)
-- Submodel: reaction1
CSH$xi = B3Br$y - (B5HBr$x + B4H2$x)
-- Submodel: CSH
mudotH = CSH$xi*CSH$R*Temp0/(ChR2H$nH + CSH$eps)
-- Submodel: reaction1
CSHBr$xi = B5HBr$x + B3Br$y - B4H2$x
-- Submodel: CSHBr
mudotHBr = CSHBr$xi*CSHBr$R*Temp0/(ChB$nHBr + Cr$CBps2)
-- Submodel: Chpower
Gdot = mudotBr2*Ch2$nBr2 + mudotBr*ChBr$nBr + mudotH2*ChH2$nH2 + mudotH*
  ChR2H$nH + mudotHBr*ChB$nHBr + CSBr2$mu*CSBr2$xi + CSBr$mu*CSBr$xi +
  CSH2$mu*CSH2$xi + CSH$mu*CSH$xi + CSHBr$mu*CSHBr$xi
d/dt G = Gdot/Cr$C$ty1
-- Submodel: ChR2B2
C2$pdor2 = mudotBr2*Ch2$nBr2/V0
-- Submodel: ChR2Br
Cr$pdoBr = mudotBr*ChBr$nBr/V0
-- Submodel: ChR2H2
C2$pdoH2 = mudotH2*ChH2$nH2/V0
-- Submodel: ChR2H
CH$pdotH = mudotH*ChR2H$nH/V0
-- Submodel: ChR2HB
CB$pdoBr = mudotHBr*ChB$nHBr/V0
-- Submodel: Pneumatic
pdot = C2$pdor2 + Cr$pdoBr + C2$pdoH2 + CH$pdotH + CB$pdoBr
d/dt p = pdot/Pc$dirty
-- Submodel: CSBr2
d/dt CSr2$mu1 = sgn(CSBr2$sc - 1)*mudotBr2/C2$C$ty1
-- Submodel: ChR2B2
d/dt Ch2$nBr2 = CSBr2$xi/C2$C$ty2
-- Submodel: ChR2Br
d/dt ChBr$nBr = CSBr$xi/Cr$C$ty2
-- Submodel: ChR2H2
d/dt ChH2$nH2 = CSH2$xi/C2$C$ty3
-- Submodel: ChR2H
d/dt ChR2H$nH = CSH$xi/CH$C$ty1
-- Submodel: ChR2HB
d/dt ChB$nHBr = CSHBr$xi/CB$dirtv
-- Submodel: CSBr
d/dt Cr$CBu11 = sgn(CSBr$sc - 1)*mudotBr/Cr$C$ty3
-- Submodel: CSH2
d/dt CSH2$mu1 = sgn(CSH2$sc - 1)*mudotH2/C2$C$ty4
-- Submodel: CSH
d/dt CSH$mu1 = sgn(CSH$sc - 1)*mudotH/CH$C$ty2
-- Submodel: CSHBr
d/dt Cr$CBu12 = sgn(CSHBr$sc - 1)*mudotHBr/Cr$C$ty4
```

```
-- Submodel: reaction1
y1 = Ch2$nBr2/ntot/r1$dirty
y2 = ChBr$nBr/ntot/r1$dirty
y3 = ChH2$nH2/ntot/r1$dirty
y4 = ChR2H$nH/ntot/r1$dirty
y5 = ChB$nHBr/ntot/r1$dirty
----------------------------------------
    OUT
    dispt y1, y2, y3, y4, y5
----------------------------------------
/--
/PIC 'h9_ctlas.PRC           '
/--
```

B.1.4 Desirable DYMOLA Main Program

```
{ H2 -Br2 reaction isochoric and isothermic }


{ +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }
{             Loading the Model - Type Programs                       }
{ +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }

@bond.lib
@TF.lib
@Thermic.ct
@Pneumati.ct
@Chemical.ct
@Chpower.ct
@CS.ct
@ChR1k1.ct
@ChR1k2.ct
@ChR1k3.ct
@ChR1k4.ct
@ChR1k5.ct
@ChR2Br2.ct
@ChR2Br.ct
@ChR2H2.ct
@ChR2H.ct
@ChR2HBr.ct



{---------------------------------------------------------------------------}
{ ********************  MAIN PROGRAM  ************************************ }
{---------------------------------------------------------------------------}

model reaction1

{ ******  Calling the Submodels  ****** }

  submodel ChR1k1
  submodel ChR1k2
  submodel ChR1k3
  submodel ChR1k4
  submodel ChR1k5
  submodel ChR2B2  (ic nBr2 = 0.5E-7)
  submodel ChR2Br  (ic nBr = 0.0)
  submodel ChR2H2  (ic nH2 = 0.5E-7)
```

```
   submodel ChR2H    (ic nH = 0.0)
   submodel ChR2HB   (ic nHBr = 0.0)
   submodel (CS)          CSBr2 (ic mu1 = muBr2) ->
                          CSBr  (ic mu1 = muBr)  ->
                          CSH2  (ic mu1 = muH2) ->
                          CSH   (ic mu1 = muH)   ->
                          CSHBr (ic mu1 = muHBr)

   submodel Pneumatic    (ic p = 1.0E+5)
   submodel Thermic      (ic S = S0)
   submodel Chpower      (ic G = 0.0)
   submodel Chemical
   submodel (TF)         TF1(k = 0.5), TF2(k = 2.0)
   submodel (bond)       B11, B1Br2, B1Br, B21, B2Br2, B2Br, B31
   submodel (bond)       B3H2, B3H, B3HBr, B3Br, B41, B4H, B4HBr
   submodel (bond)       B4Br, B4H2, B51, B5Br, B5Br2, B5H, B5HBr


 { ****** I/O Declaration ****** }

   input   Temp0, V0
   output  y1, y2, y3, y4, y5
{ ****** Declaration of the Variable Type ****** }

  local   K900, HiBr2, HiBr, HiH2, HiH, HiHBr
  local   SiBr2, SiBr, SiH2, SiH, SiHBr
  local   muBr2, muBr, muH2, muH, muHBr


{ ****** Declaration of the nodes ****** }

  node    n11, n21, n31, n41, n51, n1Br, n2Br, nBr2, nBr, ->
          nH2, nH, nHBr, n1, n2, n3, n4, n5


{ +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
{                   CONNECTIONS OF THE BOND GRAPH ELEMENTS
{ +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

{ ****** Connections of the ChR1-elements and of the bonds }
        { attached to either the 0-junctions or the 1-junctions ****** }

  connect ChR1k1:chem1 at   n11,      ->
          B11   from n11  to n1,    ->
          B1Br2 from nBr2 to n1,    ->
          B1Br  from n1   to n1Br, ->
          TF1   from n1Br to nBr

  connect ChR1k2:chem1 at   n21,      ->
          B21   from n21  to n2,    ->
          B2Br  from n2Br to n2,    ->
          B2Br2 from n2   to nBr2, ->
          TF2   from nBr  to n2Br

  connect ChR1k3:chem1 at   n31,      ->
          B31   from n31  to n3,    ->
          B3HBr from n3   to nHBr, ->
          B3H   from n3   to nH,   ->
          B3H2  from nH2  to n3,    ->
          B3Br  from nBr  to n3
   connect ChR1k4:chem1 at   n41,     ->
           B41   from n41  to n4,   ->
           B4H   from nH   to n4,   ->
           B4HBr from nHBr to n4,   ->
           B4Br  from n4   to nBr,  ->
           B4H2  from n4   to nH2

   connect ChR1k5:chem1 at   n51,     ->
           B51   from n51  to n5,   ->
           B5Br  from n5   to nBr,  ->
           B5Br2 from nBr2 to n5,   ->
           B5H   from nH   to n5,   ->
           B5HBr from n5   to nHBr

  { ****** Connection of the ChR2-elements to the CS-elements ****** }

   connect ChR2B2:chem2  at CSBr2:chem2, ->
           ChR2Br:chem2  at CSBr:chem2,  ->
           ChR2H2:chem2  at CSH2:chem2,  ->
           ChR2H:chem2   at CSH:chem2,   ->
           ChR2HB:chem2  at CSHBr:chem2

 { ****** Connection of the CS-elements to the 0-junctions ****** }

  connect CSBr2:chem1 at nBr2, ->
          CSBr:chem1  at nBr,  ->
          CSH2:chem1  at nH2,  ->
          CSH:chem1   at nH,   ->
          CSHBr:chem1 at nHBr

 { ****** Connection of the ChR2-elements to the Pneumatic-element ****** }

   connect ChR2B2:pneu1 at Pneumatic:pneu1, ->
           ChR2Br:pneu2 at Pneumatic:pneu2, ->
           ChR2H2:pneu3 at Pneumatic:pneu3, ->
           ChR2H:pneu4 at Pneumatic:pneu4, ->
           ChR2HB:pneu5 at Pneumatic:pneu5

 { ****** Connection of the ChR1-elements to the Thermic-element ****** }

   connect ChR1k1:thermk1 at Thermic:thermk1, ->
           ChR1k2:thermk2 at Thermic:thermk2, ->
           ChR1k3:thermk3 at Thermic:thermk3, ->
           ChR1k4:thermk4 at Thermic:thermk4, ->
           ChR1k5:thermk5 at Thermic:thermk5
```

```
{ ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }
{            Computation of the Free Energy of Formation   and               }
{            Reading out the equilibrium constant K                          }
{ ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }

     func K900  = KBr2Bt(Temp0)
     func HiBr2 = HiBr2t(Temp0)
     func HiBr  = HiBrt(Temp0)
     func HiH2  = HiH2t(Temp0)
     func HiH   = HiHt(Temp0)
     func HiHBr = HiHBrt(Temp0)
     func SiBr2 = SiBr2t(Temp0)
     func SiBr  = SiBrt(Temp0)
     func SiH2  = SiH2t(Temp0)
     func SiH   = SiHt(Temp0)
     func SiHBr = SiHBrt(Temp0)

     muBr2 = HiBr2 - Temp0*SiBr2
     muBr  = HiBr  - Temp0*SiBr
     muH2  = HiH2  - Temp0*SiH2
     muH   = HiH   - Temp0*SiH
     muHBr = HiHBr - Temp0*SiHBr

     S0 = SiBr2 + SiH2


{ ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }
{                 Setting the Modulating Signals                             }
{ ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }


{ ****** V - Terminal of the ChR1-elements and the V - input ****** }

  ChR1k1.V = V0
  ChR1k2.V = V0
  ChR1k3.V = V0
  ChR1k4.V = V0
  ChR1k5.V = V0


{ ****** ni - Terminal of the ChR1-elements and the ChR2-elements ****** }

 ChR1k1.nBr2  = ChR2B2.nBr2

 ChR1k2.nBr   = ChR2Br.nBr
 ChR1k2.K     = K900

 ChR1k3.nH2  = ChR2H2.nH2
 ChR1k3.nBr  = ChR2Br.nBr

 ChR1k4.nHBr = ChR2HB.nHBr
 ChR1k4.nH   = ChR2H.nH

 ChR1k5.nBr2 = ChR2B2.nBr2
 ChR1k5.nH   = ChR2H.nH
```

```
{ ****** xi-Terminal of the ChR2-elements and the CS-elements ****** }

  ChR2B2.xiBr2 = CSBr2.xi
  ChR2Br.xiBr  = CSBr.xi
  ChR2H2.xiH2  = CSH2.xi
  ChR2H.xiH    = CSH.xi
  ChR2HB.xiHBr = CSHBr.xi


{ ****** T - Terminal of the ChR2-elements and the T - input ****** }

  CSBr2.Temp = Temp0
  CSBr.Temp  = Temp0
  CSH2.Temp  = Temp0
  CSH.Temp   = Temp0
  CSHBr.Temp = Temp0


{ ****** mudoti - Terminal of the Chpower-element}
        { and the CS-elements ****** }

  Chpower.mudBr2 = CSBr2.mudot
  Chpower.mudBr  = CSBr.mudot
  Chpower.mudH2  = CSH2.mudot
  Chpower.mudH   = CSH.mudot
  Chpower.mudHBr = CSHBr.mudot


{ ****** mui - Terminal of the Chpower-element}
        { and the CS-elements ****** }

  Chpower.muBr2 = CSBr2.mu
  Chpower.muBr  = CSBr.mu
  Chpower.muH2  = CSH2.mu
  Chpower.muH   = CSH.mu
  Chpower.muHBr = CSHBr.mu


{ ****** xii - Terminal of the Chpower-element}
        { and the CS-elements ****** }

  Chpower.xiBr2 = CSBr2.xi
  Chpower.xiBr  = CSBr.xi
  Chpower.xiH2  = CSH2.xi
  Chpower.xiH   = CSH.xi
  Chpower.xiHBr = CSHBr.xi


{ ****** ni - Terminal of the Chpower-element}
        { and the ChR2-elements ****** }
  Chpower.nBr2 = ChR2B2.nBr2
  Chpower.nBr  = ChR2Br.nBr
  Chpower.nH2  = ChR2H2.nH2
  Chpower.nH   = ChR2H.nH
  Chpower.nHBr = ChR2HB.nHBr


{ ****** ni - Terminal of the Chemical-element}
         { and the ChR2-elements ****** }

  Chemical.nBr2 = ChR2B2.nBr2
  Chemical.nBr  = ChR2Br.nBr
  Chemical.nH2  = ChR2H2.nH2
  Chemical.nH   = ChR2H.nH
  Chemical.nHBr = ChR2HB.nHBr


{ ****** T, V -Terminal of the Thermic-, }
        { Pneumatic-element and the T, V -inputs ****** }

  Thermic.Temp = Temp0
  Pneumatic.V  = V0


{ +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }
{          Connection of the ni - state variables with the outputs          }
{ +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }

  y1 = ChR2B2.nBr2/Chemical.ntot
  y2 = ChR2Br.nBr/Chemical.ntot
  y3 = ChR2H2.nH2/Chemical.ntot
  y4 = ChR2H.nH/Chemical.ntot
  y5 = ChR2HB.nHBr/Chemical.ntot

end
```

B.1.5 Desirable Simulation Control Program

```
{ H2 - Br2 reaction isobaric and isothermic }

{ +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }
{      Control Block specifying the Parameters of the Simulation    }
{              ( coded in the DYMOLA - Terminology )                }
{ +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }


cmodel

  simutime    5000.0
  step        1.0E-5
  commupoints 10001


{ ****** Specifying the Inputs of the Simulation ****** }

  input 2, Temp0(depend, 800.0),   ->
           V0(depend, 1.0E-3)


{ +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }
{          Control - Block coded in the DESIRE - language           }
{ +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }

  ctblock


{ ****** Specifying the Integration  Algorithm, largest }
       { Time Step, and largest absolute Error ****** }

    { Runge-Kutta (4/2) }
    irule 4

    DTMAX=0.1
    ERMAX=1.0E-4


{ ****** Table of the Data to compute the free energy of formation ****** }

      dimension KBr2Bt[38], SiBr2t[38], HiBr2t[38]
      dimension SiBrt[38], HiBrt[38], SiH2t[38], HiH2t[38]
      dimension SiHt[38], HiHt[38], SiHBrt[30], HiHBrt[30]
      data      298.15, 300.0, 400.0, 500.0, 600.0, 700.0
      data      800.0, 900.0, 1000.0, 1100.0, 1200.0, 1300.0
      data      1400.0, 1500.0, 1600.0, 1700.0, 1800.0, 1900.0
      data      2000.0
      data      3.1623E-29, 7.7446E-29, 1.9543E-20, 2.2182E-15
      data      5.2844E-12, 1.3867E-9,  9.0782E-8,  2.3768E-6
      data      3.2509E-5,  2.7861E-4,  1.6788E-3,  7.6913E-3
      data      2.8510E-2,  8.8716E-2,  2.4044E-1,  5.8344E-1
      data      1.7947   ,  2.6061   ,  4.9431
```

S_{Br_2}

```
data      298.15,   300.0,   400.0,   500.0,   600.0,   700.0
data      800.0,    900.0, 1000.0, 1100.0, 1200.0, 1300.0
data      1400.0,   1500.0, 1600.0, 1700.0, 1800.0, 1900.0
data      2000.0
data      245.510, 245.731, 256.220, 264.463, 271.246, 277.007
data      282.014, 286.444, 290.413, 294.014, 297.300, 300.336
data      303.145, 305.766, 308.224, 310.535, 312.716, 314.780
data      316.744
```

H_{Br_2}

```
data      298.15,   300.0,   400.0,   500.0,   600.0,   700.0
data      800.0,    900.0, 1000.0, 1100.0, 1200.0, 1300.0
data      1400.0,   1500.0, 1600.0, 1700.0, 1800.0, 1900.0
data      2000.0
data      30927.892, 30994.880, 34641.583, 38334.341, 42056.406
data      45795.218, 49546.591, 53306.338, 57074.458, 60846.764
data      64631.632, 68420.686, 72213.926, 76011.354, 79817.155
data      83627.143, 87441.318, 91263.866, 95090.602
```

S_{Br}

```
data      298.15,   300.0,   400.0,   500.0,   600.0,   700.0
data      800.0,    900.0, 1000.0, 1100.0, 1200.0, 1300.0
data      1400.0,   1500.0, 1600.0, 1700.0, 1800.0, 1900.0
data      2000.0
data      175.017, 175.146, 181.117, 185.752, 189.553, 192.785
data      195.599, 198.098, 200.347, 202.394, 204.278, 206.020
data      207.644, 209.168, 210.600, 211.957, 213.242, 214.469
data      215.637
```

H_{Br}

```
data      298.15,   300.0,   400.0,   500.0,   600.0,   700.0
data      800.0,    900.0, 1000.0, 1100.0, 1200.0, 1300.0
data      1400.0,   1500.0, 1600.0, 1700.0, 1800.0, 1900.0
data      2000.0
data      111913.164, 111950.845, 114027.498, 116104.151
data      118189.177, 120286.764, 122392.724, 124515.432
data      126650.700, 128798.528, 130963.104, 133140.240
data      135334.123, 137540.567, 139759.571, 141995.322
data      144247.820, 146512.879, 148794.685
```

S_{H_2}

```
data      298.15,   300.0,   400.0,   500.0,   600.0,   700.0
data      800.0,    900.0, 1000.0, 1100.0, 1200.0, 1300.0
data      1400.0,   1500.0, 1600.0, 1700.0, 1800.0, 1900.0
data      2000.0
data      130.670, 130.850, 139.148, 145.625, 150.959, 155.510
data      159.496, 163.046, 166.254, 169.189, 171.893, 174.405
data      176.758, 178.969, 181.058, 183.030, 184.931, 186.731
data      188.460
```

```
{ ****** Routine to restart the simulation by doubling the scale factor }
       { after its stop due to the bounds exceeded ****** }

    scale = 1.0E-7
    XCCC  = 1
    label TRY
    drunr | if XCCC < 0 then XCCC = -XCCC | scale = 2*scale | go to TRY
                      else proceed
  ctend

{ ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }
{                         The Output Block                         }
{ ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }

  outblock
    OUT
    dispt y1, y2, y3, y4, y5
  outend

end
```

```
data      298.15,   300.0,   400.0,   500.0,   600.0,   700.0        H H2
data      800.0,    900.0, 1000.0, 1100.0, 1200.0, 1300.0
data      1400.0,  1500.0, 1600.0, 1700.0, 1800.0, 1900.0
data      2000.0
data      0.0,        54.428,     2939.134,    5840.586,  8767.159
data      11723.04,   14704.052, 17718.538, 20766.528, 23843.826
data      26950.432, 30094.718, 33264.126, 36471.215, 39711.798
data      42981.689, 46285.074, 49617.767, 52988.141

data      298.15,   300.0,   400.0,   500.0,   600.0,   700.0
data      800.0,    900.0, 1000.0, 1100.0, 1200.0, 1300.0
data      1400.0,  1500.0, 1600.0, 1700.0, 1800.0, 1900.0
data      2000.0
data      114.681, 114.810, 120.793, 125.4365, 129.226, 132.433
data      135.2085, 137.662, 139.852, 141.836, 143.645, 145.307
data      146.852, 148.284, 149.628, 150.888, 152.077, 153.203
data      154.271

data      298.15,   300.0,   400.0,   500.0,   600.0,   700.0
data      800.0,    900.0, 1000.0, 1100.0, 1200.0, 1300.0
data      1400.0,  1500.0, 1600.0, 1700.0, 1800.0, 1900.0
data      2000.0
data      218149.027, 218186.208, 220267.548, 222348.388
data      224429.227, 226505.880, 228586.720, 230667.559
data      232748.399, 234829.238, 236905.891, 238986.731
data      241067.570, 243148.410, 245229.250, 247305.902
data      249386.742, 251467.582, 253548.421

data      298.15,   300.0,   400.0,   500.0,   600.0,   700.0        S HBr
data      800.0,    900.0, 1000.0, 1100.0, 1200.0, 1300.0
data      1400.0,  1500.0, 1600.0
data      198.718, 198.898, 207.276, 213.824, 219.250, 223.910
data      228.013, 231.702, 235.055, 238.145, 241.017, 243.701
data      246.234, 248.629, 250.9065

data      298.15,   300.0,   400.0,   500.0,   600.0,   700.0        H HBr
data      800.0,    900.0, 1000.0, 1100.0, 1200.0, 1300.0
data      1400.0,  1500.0, 1600.0
data      -36404.226, -36349.798, -33435.785, -30500.838
data      -27528.210, -24505.340, -21428.042, -18296.316
data      -15110.161, -11869.578,  -8570.380,  -5212.566
data      -1800.324, 1670.533, 5200.006

read KBr2Bt
read SiBr2t
read HiBr2t
read SiBrt
read HiBrt
read SiH2t
read HiH2t
read SiHt
read HiHt
read SiHBrt
read HiHBrt
```

B.2 Hydrogen–Bromine Reaction: Isentropic-isobaric Case

Following, the DYMOLA main program "*PS.dym*" including all the model types, the simulation control program "*PS.ctl*", and its generated DESIRE–program "*PS.des*" of the hydrogen–bromine reaction under isentropic–isobaric conditions are listed.

B.2.1 DYMOLA Main Program "*PS.dym*" Including All the Model Types

```
{ Bond Graph bond }

model type bond

  cut A (x / y) B (y / -x)
  main cut C [A B]
  main path P <A - B>

end

{ H2 - Br2 reaction isobaric and isentropic }

model type TF

  cut A( mu1/xi1 ), B( mu2/-xi2 )
  main cut C [A B]
  main path P < A-B >
  parameter k = 1.0

  k*xi2 = xi1
  k*mu1 = mu2

end


{ H2 - Br2 Reaction isobaric and isentropic }

model type Thermic

  cut thermtot( Tdot/-S )
  cut therm1( TdotBr2/S ), therm2( TdotBr/S )
  cut therm3( TdotH2/S ), therm4( TdotH/S )
  cut therm5( TdotHBr/S )
  terminal Temp
  parameter dirty = 1.0

  Tdot = TdotBr2 + TdotBr + TdotH2 + TdotH + TdotHBr
  dirty*der(Temp) = Tdot

end
```

```
{ H2 - Br2 Reaction isobaric and isentropic }

model type Pneumatic

  cut pneutot( p/Phi )
  cut pneu1( p/-Phik1 ), pneu2( p/-Phik2 )
  cut pneu3( p/-Phik3 ), pneu4( p/-Phik4 )
  cut pneu5( p/-Phik5 )
  terminal V
  parameter dirty = 1.0

  Phi           = Phik1 + Phik2 + Phik3 + Phik4 + Phik5
  dirty*der(V)  = Phi

end


{ H2 - Br2 Reaction isobaric and isentropic }

model type Chemical
     terminal xiBr2, xiBr, xiH2, xiH, xiHBr
     terminal nBr2, nBr, nH2, nH, nHBr
     terminal ntot, xitot

     xitot = xiBr2 + xiBr + xiH2 + xiH2 + xiHBr
     ntot  = nBr2 + nBr + nH2 + nH2 + nHBr

end


{ H2 - Br2 Reaction, isobaric and isentropic }

model type CS

   cut chem1(mu/xi), chem2(mudot/n)
   main cut C[chem1]
   local Temp, mu1
   terminal Ttot, S, ntot, xitot
   parameter R = 8.314, eps = 1.0E-15, sc = 1.5

   Temp = Ttot*n/ntot
   mudot*(n + eps)*ntot*R = xitot*R*Temp*S - mu*xi*S
   sgn( sc - 1.0 )*der(mu1) = mudot
   mu1*sgn( sc -1.0 ) = mu

end


{ H2 - Br2 Reaction, isobaric and isentropic }

model type ChR1k1

  cut chem1( muk1/-xik1 ), pneu1( p/Phik1 )
  main cut C[ chem1 ]
  terminal nBr2, Temp
  parameter R=8.314
  local k1, ak1
  ak1    = 1.39E+8*Temp**0.5*( 189243/( R*Temp ) )**1.97
  k1     = ak1*exp( -189243/( R*Temp ) )
  p*Phik1 = muk1*xik1
  xik1   = k1*nBr2

end


{ H2 - Br2 Reaction, isobaric and isentropic }

model type ChR1k2

  cut chem1( muk2/-xik2 ), pneu2(p/Phik2)
  main cut C[ chem1 ]
  terminal nBr, K, V, Temp
  parameter R=8.314
  local k1, ak1, k2

  ak1     = 1.39E+8*Temp**0.5*( 189243/( R*Temp ) )**1.97
  k1      = ak1*exp( -189243/( R*Temp ) )
  k2      = k1/K
  p*Phik2  = muk2*xik2
  xik2    = k2*nBr*nBr/V

end


{ H2 - Br2 Reaction, isobaric and isentropic }

model type ChR1k3

  cut chem1( muk3/-xik3 ), pneu3(p/Phik3)
  main cut C[chem1]
  terminal nH2, nBr, V, Temp
  parameter R=8.314
  local k3

  k3    = ( 10**11.43 )*exp( -82400/( R*Temp ) )
  p*Phik3 = muk3*xik3
  xik3  = k3*nH2*nBr/V

end


{ H2 - Br2 Reaction, isobaric and isentropic }

model type ChR1k4

  cut chem1( muk4/-xik4 ), pneu4(p/Phik4)
  main cut C[chem1]
  terminal nHBr, nH, V, Temp
  parameter R=8.314, m=10.0
  local k4, k5

  k5    = 10.0**11.97*exp( -149800.0/( R*Temp ) )
  k4    = k5/m
  p*Phik4 = muk4*xik4
  xik4  = k4*nHBr*nH/V

end
```

```
{ H2 - Br2 Reaction, isobaric and isentropic }

model type ChR1k5

  cut chem1( muk5/-xik5 ), pneu5(p/Phik5)
  main cut C[chem1]
  terminal nBr2, nH, V, Temp
  parameter R=8.314
  local k5

  k5    = 10.0**11.97*exp( -149800.0/( R*Temp ) )
  p*Phik5 = muk5*xik5
  xik5  = k5*nBr2*nH/V

end


{ H2 - Br2 Reaction, isobaric and isentropic }

model type ChR2B2

  cut chem2( mudotBr2/-nBr2 ), therm1( TdotBr2/-S)
  main cut C[ chem2 ]
  terminal xiBr2
  parameter dirty = 1.0, eps = 1.0E-15

  0.0 = TdotBr2*(S + eps) + mudotBr2*nBr2
  dirty*der(nBr2) = xiBr2

end


{ H2 - Br2 Reaction, isobaric and isentropic }

model type ChR2Br

  cut chem2( mudotBr/-nBr ), therm2(TdotBr/-S)
  main cut C[ chem2 ]
  terminal xiBr
  parameter dirty = 1.0, eps = 1.0E-15

  0.0 = TdotBr*(S + eps) + mudotBr*nBr
  dirty*der(nBr) = xiBr

end


{ H2 - Br2 Reaction, isobaric and isentropic }

model type ChR2H2

  cut chem2( mudotH2/-nH2 ), therm3(TdotH2/-S)
  main cut C[ chem2 ]
  terminal xiH2
  parameter dirty = 1.0, eps = 1.0E-15
```

```
   0.0 = TdotH2*(S + eps) + mudotH2*nH2
   dirty*der(nH2) = xiH2

 end


 { H2 - Br2 Reaction, isobaric and isentropic }

 model type ChR2H

   cut chem2( mudotH/-nH ), therm4(TdotH/-S)
   main cut C[ chem2 ]
   terminal xiH
   parameter dirty = 1.0, eps = 1.0E-15

   0.0 = TdotH*(S + eps) + mudotH*nH
   dirty*der(nH) = xiH

 end


 { H2 - Br2 Reaction, isobaric and isentropic }

 model type ChR2HB

   cut chem2( mudotHBr/-nHBr ), therm5(TdotHBr/-S)
   main cut C[ chem2 ]
   terminal xiHBr
   parameter dirty = 1.0, eps = 1.0E-15

   0.0 = TdotHBr*(S + eps) + mudotHBr*nHBr
   dirty*der(nHBr) = xiHBr

 end



 {----------------------------------------------------------------------------}

 model reaction1

   submodel ChR1k1
   submodel ChR1k2
   submodel ChR1k3
   submodel ChR1k4
   submodel ChR1k5
   submodel ChR2B2   (ic nBr2 = 0.5E-7)
   submodel ChR2Br   (ic nBr = 0.0)
   submodel ChR2H2   (ic nH2 = 0.5E-7)
   submodel ChR2H    (ic nH = 0.0)
   submodel ChR2HB   (ic nHBr = 0.0)
   submodel (CS)          CSBr2 (sc = 0.5) (ic mu1 = 204493.0) ->
                          CSBr  (sc = 0.5) (ic mu1 = 53772.8)  ->
                          CSH2  (sc = 0.5) (ic mu1 = 129023.0) ->
                          CSH   (sc = 1.5) (ic mu1 = 106772.0)  ->
                          CSHBr (sc = 0.5) (ic mu1 = 226828.0)
```

```
submodel Pneumatic  (ic V = 1.0E-3)
submodel Thermic    (ic Temp = 800.0)
submodel Chemical
submodel (TF)        TF1(k = 0.5), TF2(k = 2.0)
submodel (bond)      B11, B1Br2, B1Br, B21, B2Br2, B2Br, B31
submodel (bond)      B3H2, B3H, B3HBr, B3Br, B41, B4H, B4HBr
submodel (bond)      B4Br, B4H2, B51, B5Br, B5Br2, B5H, B5HBr

input   p0, S0
output  y1, y2, y3, y4, y5
local   K900, Temp0
parameter dirty = 1.0
node    n11, n21, n31, n41, n51, n1Br, n2Br, nBr2, nBr, ->
        nH2, nH, nHBr, n1, n2, n3, n4, n5


connect ChR1k1:chem1 at   n11,      ->
        B11   from n11  to n1,   ->
        B1Br2 from nBr2 to n1,   ->
        B1Br  from n1   to n1Br, ->
        TF1   from n1Br to nBr

connect ChR1k2:chem1 at   n21,      ->
        B21   from n21  to n2,   ->
        B2Br  from n2Br to n2,   ->
        B2Br2 from n2   to nBr2, ->
        TF2   from nBr  to n2Br

connect ChR1k3:chem1 at   n31,      ->
        B31   from n31  to n3,   ->
        B3HBr from n3   to nHBr, ->
        B3H   from n3   to nH,  ->
        B3H2  from nH2  to n3,   ->
        B3Br  from nBr  to n3

connect ChR1k4:chem1 at   n41,      ->
        B41   from n41  to n4,   ->
        B4H   from nH   to n4,   ->
        B4HBr from nHBr to n4,   ->
        B4Br  from n4   to nBr,  ->
        B4H2  from n4   to nH2

connect ChR1k5:chem1 at   n51,      ->
        B51   from n51  to n5,   ->
        B5Br  from n5   to nBr,  ->
        B5Br2 from nBr2 to n5,   ->
        B5H   from nH   to n5,   ->
        B5HBr from n5   to nHBr

{ ****** Connection of the ChR2-elements to the CS-element ****** }

  connect ChR2B2:chem2  at CSBr2:chem2, ->
          ChR2Br:chem2  at CSBr:chem2,  ->
          ChR2H2:chem2  at CSH2:chem2,  ->
          ChR2H:chem2   at CSH:chem2,   ->
          ChR2HB:chem2  at CSHBr:chem2


{ ****** Connection of the CS-elements to the 0-junctions ****** }

  connect CSBr2:chem1 at nBr2, ->
          CSBr:chem1  at nBr,  ->
          CSH2:chem1  at nH2,  ->
          CSH:chem1   at nH,   ->
          CSHBr:chem1 at nHBr


{ ****** Connection of the ChR2-elements to the Pneumatic-element ****** }

  connect ChR1k1:pneu1 at Pneumatic:pneu1, ->
          ChR1k2:pneu2 at Pneumatic:pneu2, ->
          ChR1k3:pneu3 at Pneumatic:pneu3, ->
          ChR1k4:pneu4 at Pneumatic:pneu4, ->
          ChR1k5:pneu5 at Pneumatic:pneu5


{ ****** Connection of the ChR1-elements to the Thermic-element ****** }

  connect ChR2B2:therm1 at Thermic:therm1, ->
          ChR2Br:therm2 at Thermic:therm2, ->
          ChR2H2:therm3 at Thermic:therm3, ->
          ChR2H:therm4 at Thermic:therm4,  ->
          ChR2HB:therm5 at Thermic:therm5

          func K900   = KBr2Bt(Temp0)

          Temp0 = Thermic.Temp

          ChR1k2.V = Pneumatic.V
          ChR1k3.V = Pneumatic.V
          ChR1k4.V = Pneumatic.V
          ChR1k5.V = Pneumatic.V


          ChR1k1.nBr2  = ChR2B2.nBr2

          ChR1k2.nBr   = ChR2Br.nBr
          ChR1k2.K     = K900

          ChR1k3.nH2  = ChR2H2.nH2
          ChR1k3.nBr  = ChR2Br.nBr

          ChR1k4.nHBr = ChR2HB.nHBr
          ChR1k4.nH   = ChR2H.nH

          ChR1k5.nBr2 = ChR2B2.nBr2
          ChR1k5.nH   = ChR2H.nH


          ChR1k1.Temp = Thermic.Temp
          ChR1k2.Temp = Thermic.Temp
          ChR1k3.Temp = Thermic.Temp
          ChR1k4.Temp = Thermic.Temp
          ChR1k5.Temp = Thermic.Temp
          ChR2B2.xiBr2  = CSBr2.xi
          ChR2Br.xiBr   = CSBr.xi
          ChR2H2.xiH2   = CSH2.xi
          ChR2H.xiH     = CSH.xi
          ChR2HB.xiHBr  = CSHBr.xi

          CSBr2.Ttot = Thermic.Temp
          CSBr.Ttot  = Thermic.Temp
          CSH2.Ttot  = Thermic.Temp
          CSH.Ttot   = Thermic.Temp
          CSHBr.Ttot = Thermic.Temp

          CSBr2.S = S0
          CSBr.S  = S0
          CSH2.S  = S0
          CSH.S   = S0
          CSHBr.S = S0

          CSBr2.xitot = Chemical.xitot
          CSBr.xitot  = Chemical.xitot
          CSH2.xitot  = Chemical.xitot
          CSH.xitot   = Chemical.xitot
          CSHBr.xitot = Chemical.xitot

          CSBr2.ntot = Chemical.ntot
          CSBr.ntot  = Chemical.ntot
          CSH2.ntot  = Chemical.ntot
          CSH.ntot   = Chemical.ntot
          CSHBr.ntot = Chemical.ntot


          Chemical.xiBr2 = CSBr2.xi
          Chemical.xiBr  = CSBr.xi
          Chemical.xiH2  = CSH2.xi
          Chemical.xiH   = CSH.xi
          Chemical.xiHBr = CSHBr.xi

          Chemical.nBr2 = ChR2B2.nBr2
          Chemical.nBr  = ChR2Br.nBr
          Chemical.nH2  = ChR2H2.nH2
          Chemical.nH   = ChR2H.nH
          Chemical.nHBr = ChR2HB.nHBr

          Thermic.S = S0
          Pneumatic.p = p0

          dirty*y1 = CSBr2.n
          dirty*y2 = CSBr.n
          dirty*y3 = CSH2.n
          dirty*y4 = CSH.n
          dirty*y5 = CSHBr.n

end
```

B.2.2 Simulation Control Program "*PS.ctl*"

```
{ H2 - Br2 reaction isobaric and isentropic }

{ ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }
{      Control Block specifying the Parameters of the Simulation    }
{              ( coded in the DYMOLA - Terminology )                }
{ ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }


cmodel

  simutime    5000.0
  step        1.0E-12
  commupoints 2001


{ ****** Specifying the Inputs of the Simulation ****** }

  input 2, p0(depend, 1.013E+5),  ->
           S0(depend, 0.0)


{ ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }
{          Control - Block coded in the DESIRE - language          }
{ ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }

  ctblock


{ ****** Specifying the Integration  Algorithm, largest }
      { Time Step, and largest absolute Error ****** }

    { Runge-kutta (4/2) }
    irule 4

    DTMAX=0.1
    ERMAX=1.0E-6


{ ****** Table of the Data to compute the free energy of formation ****** }

      dimension KBr2Bt[38]

      data      298.15, 300.0, 400.0, 500.0, 600.0, 700.0
      data      800.0, 900.0, 1000.0, 1100.0, 1200.0, 1300.0
      data      1400.0, 1500.0, 1600.0, 1700.0, 1800.0, 1900.0
      data      2000.0
      data      3.1623E-29, 7.7446E-29, 1.9543E-20, 2.2182E-15
      data      5.2844E-12, 1.3867E-9,  9.0782E-8,  2.3768E-6
      data      3.2509E-5,  2.7861E-4,  1.6788E-3,  7.6913E-3
      data      2.8510E-2,  8.8716E-2,  2.4044E-1,  5.8344E-1
      data      1.7947    ,  2.6061   ,  4.9431

      read KBr2Bt
```

```
{ ****** Routine to restart the simulation by doubling the scale factor }
       { after its stop due to the bounds exceeded ****** }

    scale = 1.0E-7
    XCCC  = 1
    label TRY
    drunr | if XCCC < 0 then XCCC = -XCCC | scale = 2*scale | go to TRY
                          else proceed
  ctend
{ +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }
{                          The Output Block                           }
{ +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ }

  outblock
    OUT
    dispt y1, y2, y3, y4, y5
  outend

end
```

B.2.3 Generated DESIRE–Program "*PS.des*"

```
----------------------------------------
- CONTINUOUS SYSTEM reaction1
----------------------------------------
- STATE Thc$Temp Pnetic$V CSr2$mu1 Ch2$nBr2 ChBr$nBr ChH2$nH2 ChR2H$nH ChB$nHB

- DER dThcTemp  dPneic$V  dCSr$mu1  dCh2nBr2  dChB$nBr  dChH$nH2  dChRH$nH  dC
nHBr
- STATE Cr$CBu11 CSH2$mu1 CSH$mu1 Cr$CBu12
- DER dCr$Bu11  dCSH$mu1  dCSH$mu1  dCr$Bu12
- OUTPUT y1 y2 y3 y4 y5
- INPUT p0 S0
- PARAMETERS and CONSTANTS:
F1$k=0.5
c$dirty=1.0
c$dirty=1.0
SBr2$R=8.314
Sr2$eps=1.0E-15
SBr2$sc=0.5
hR1k1$R=8.314
hR1k2$R=8.314
hR1k3$R=8.314
hR1k4$R=8.314
=10.0
hR1k5$R=8.314
2$C$ty1=1.0
hB2$eps=1.0E-15
r$dirty=1.0
hBr$eps=1.0E-15
2$C$ty2=1.0
hH2$eps=1.0E-15
H$dirty=1.0
h2H$eps=1.0E-15
B$dirty=1.0
hHB$eps=1.0E-15
SBr$R=8.314
```

```
 Cr$CBps1=1.0E-15
 CSBr$sc=0.5
 CSH2$R=8.314
 CSH2$eps=1.0E-15
 CSH2$sc=0.5
 CSH$R=8.314
 CSH$eps=1.0E-15
 CSH$sc=1.5
 CSHBr$R=8.314
 Cr$CBps2=1.0E-15
 CSHBr$sc=0.5
 TF2$k=2.0
 r1$dirty=1.0
 -- INITIAL VALUES OF STATES:
 Thc$Temp=800.0
 Pnetic$V=1.0E-3
 CSr2$mu1=204493.0
 Ch2$nBr2=0.5E-7
 ChBr$nBr=0.0
 ChH2$nH2=0.5E-7
 ChR2H$nH=0.0
 ChB$nHBr=0.0
 Cr$CBu11=53772.8
 CSH2$mu1=129023.0
 CSH$mu1=106772.0
 Cr$CBu12=226828.0

 ----------------------------------------
 TMAX=5000.0 | DT=1.0E-12 | NN=2001
     irule 4
     DTMAX=0.1
     ERMAX=1.0E-6
       dimension KBr2Bt[38]
       data      298.15, 300.0, 400.0, 500.0, 600.0, 700.0
       data      800.0, 900.0, 1000.0, 1100.0, 1200.0, 1300.0
       data      1400.0, 1500.0, 1600.0, 1700.0, 1800.0, 1900.0
       data      2000.0
       data      3.1623E-29, 7.7446E-29, 1.9543E-20, 2.2182E-15
       data      5.2844E-12, 1.3867E-9,  9.0782E-8,  2.3768E-6
       data      3.2509E-5,  2.7861E-4,  1.6788E-3,  7.6913E-3
       data      2.8510E-2,  8.8716E-2,  2.4044E-1,  5.8344E-1
       data      1.7947    ,  2.6061   ,  4.9431
       read KBr2Bt
     scale = 1.0E-7
     XCCC  = 1
     label TRY
     drunr | if XCCC < 0 then XCCC = -XCCC | scale = 2*scale | go to TRY
                         else proceed
----------------------------------------
DYNAMIC
----------------------------------------
p0 = 1.013E+5
S0 = 0.0
-- Submodel: CSBr
CSBr$mu = Cr$CBu11*sgn(CSBr$sc - 1)
-- Submodel: TF1
TF1$mu1 = CSBr$mu/TF1$k
-- Submodel: CSBr2
CSBr2$mu = CSr2$mu1*sgn(CSBr2$sc - 1)
```

```
-- Submodel: reaction1
B11$x = TF1$mu1 - CSBr2$mu
-- Submodel: ChR1k1
Chk1$ak1 = 1.39E+8*Thc$Temp^0.5*(189243/(ChR1k1$R*Thc$Temp))^1.97
ChRk1$k1 = Chk1$ak1*exp(-189243/(ChR1k1$R*Thc$Temp))
B1Br$x = ChRk1$k1*Ch2$nBr2
-- Submodel: TF1
TF1$xi2 = B1Br$x/TF1$k
-- Submodel: Chemical
Ch1$ntot = Ch2$nBr2 + ChBr$nBr + ChH2$nH2 + ChH2$nH2 + ChB$nHBr
-- Submodel: ChR1k5
ChRk5$k5 = 10.0^11.97*exp(-149800.0/(ChR1k5$R*Thc$Temp))
B5HBr$x = ChRk5$k5*Ch2$nBr2*ChR2H$nH/Pnetic$V
-- Submodel: ChR1k2
Chk2$ak1 = 1.39E+8*Thc$Temp^0.5*(189243/(ChR1k2$R*Thc$Temp))^1.97
ChRk2$k1 = Chk2$ak1*exp(-189243/(ChR1k2$R*Thc$Temp))
-- Submodel: reaction1
func K900 = KBr2Bt,Thc$Temp
-- Submodel: ChR1k2
k2 = ChRk2$k1/K900
TF2$xi2 = k2*ChBr$nBr*ChBr$nBr/Pnetic$V
-- Submodel: reaction1
CSBr2$xi = TF2$xi2 - (B5HBr$x + B1Br$x)
-- Submodel: ChR1k3
k3 = 10^11.43*exp(-82400/(ChR1k3$R*Thc$Temp))
B3Br$y = k3*ChH2$nH2*ChBr$nBr/Pnetic$V
-- Submodel: TF2
TF2$xi1 = TF2$k*TF2$xi2
-- Submodel: ChR1k4
ChRk4$k5 = 10.0^11.97*exp(-149800.0/(ChR1k4$R*Thc$Temp))
k4 = ChRk4$k5/m
B4H2$x = k4*ChB$nHBr*ChR2H$nH/Pnetic$V
-- Submodel: reaction1
CSBr$xi = B5HBr$x + B4H2$x + TF1$xi2 - (B3Br$y + TF2$xi1)
CSH2$xi = B4H2$x - B3Br$y
CSHBr$xi = B5HBr$x + B3Br$y - B4H2$x
-- Submodel: Chemical
C1$xitot = CSBr2$xi + CSBr$xi + CSH2$xi + CSH2$xi + CSHBr$xi
-- Submodel: CSBr2
C2$C2mp1 = Thc$Temp*Ch2$nBr2/Ch1$ntot
mudotBr2 = (C1$xitot*CSBr2$R*C2$C2mp1*S0 - CSBr2$mu*CSBr2$xi*S0)/((Ch2$nBr2
  CSr2$eps)*Ch1$ntot*CSBr2$R)
-- Submodel: ChR2B2
C2$Tdor2 = mudotBr2*Ch2$nBr2/(-(S0 + ChB2$eps))
-- Submodel: CSBr
Cr$Crmp1 = Thc$Temp*ChBr$nBr/Ch1$ntot
mudotBr = (C1$xitot*CSBr$R*Cr$Crmp1*S0 - CSBr$mu*CSBr$xi*S0)/((ChBr$nBr +
  Cr$CBps1)*Ch1$ntot*CSBr$R)
-- Submodel: ChR2Br
Cr$TdoBr = mudotBr*ChBr$nBr/(-(S0 + ChBr$eps))
-- Submodel: CSH2
C2$C2mp2 = Thc$Temp*ChH2$nH2/Ch1$ntot
CSH2$mu = CSH2$mu1*sgn(CSH2$sc - 1)
mudotH2 = (C1$xitot*CSH2$R*C2$C2mp2*S0 - CSH2$mu*CSH2$xi*S0)/((ChH2$nH2 +
  CSH2$eps)*Ch1$ntot*CSH2$R)
-- Submodel: ChR2H2
C2$TdoH2 = mudotH2*ChH2$nH2/(-(S0 + ChH2$eps))
-- Submodel: CSH
CSH$Temp = Thc$Temp*ChR2H$nH/Ch1$ntot
CSH$mu = CSH$mu1*sgn(CSH$sc - 1)
```

```
-- Submodel: reaction1
CSH$xi = B3Br$y - (B5HBr$x + B4H2$x)
-- Submodel: CSH
mudotH = (C1$xitot*CSH$R*CSH$Temp*S0 - CSH$mu*CSH$xi*S0)/((ChR2H$nH +
  CSH$eps)*Ch1$ntot*CSH$R)
-- Submodel: ChR2H
CH$TdotH = mudotH*ChR2H$nH/(-(S0 + Ch2H$eps))
-- Submodel: CSHBr
Cr$Crmp2 = Thc$Temp*ChB$nHBr/Ch1$ntot
CSHBr$mu = Cr$CBu12*sgn(CSHBr$sc - 1)
mudotHBr = (C1$xitot*CSHBr$R*Cr$Crmp2*S0 - CSHBr$mu*CSHBr$xi*S0)/((ChB$nHBr +
  Cr$CBps2)*Ch1$ntot*CSHBr$R)
-- Submodel: ChR2HB
CB$TdoBr = mudotHBr*ChB$nHBr/(-(S0 + ChHB$eps))
-- Submodel: Thermic
Tdot = C2$Tdor2 + Cr$TdoBr + C2$TdoH2 + CH$TdotH + CB$TdoBr
d/dt Thc$Temp = Tdot/Tc$dirty
-- Submodel: ChR1k1
Pc$Phik1 = B11$x*B1Br$x/p0
-- Submodel: TF2
TF2$mu2 = TF2$k*CSBr$mu
-- Submodel: reaction1
B21$x = CSBr2$mu - TF2$mu2
-- Submodel: ChR1k2
Pc$Phik2 = B21$x*TF2$xi2/p0
-- Submodel: reaction1
B31$x = CSH$mu + CSHBr$mu - (CSBr$mu + CSH2$mu)
-- Submodel: ChR1k3
Pc$Phik3 = B31$x*B3Br$y/p0
-- Submodel: reaction1
B41$x = CSH2$mu + CSBr$mu - (CSHBr$mu + CSH$mu)
-- Submodel: ChR1k4
Pc$Phik4 = B41$x*B4H2$x/p0
-- Submodel: reaction1
B51$x = CSHBr$mu + CSBr$mu - (CSH$mu + CSBr2$mu)
-- Submodel: ChR1k5
Pc$Phik5 = B51$x*B5HBr$x/p0
-- Submodel: Pneumatic
Phi = Pc$Phik1 + Pc$Phik2 + Pc$Phik3 + Pc$Phik4 + Pc$Phik5
d/dt Pnetic$V = Phi/Pc$dirty
-- Submodel: CSBr2
d/dt CSr2$mu1 = mudotBr2/sgn(CSBr2$sc - 1)
-- Submodel: ChR2B2
d/dt Ch2$nBr2 = CSBr2$xi/C2$C$ty1
-- Submodel: ChR2Br
d/dt ChBr$nBr = CSBr$xi/Cr$dirty
-- Submodel: ChR2H2
d/dt ChH2$nH2 = CSH2$xi/C2$C$ty2
-- Submodel: ChR2H
d/dt ChR2H$nH = CSH$xi/CH$dirty
-- Submodel: ChR2HB
d/dt ChB$nHBr = CSHBr$xi/CB$dirty
-- Submodel: CSBr
d/dt Cr$CBu11 = mudotBr/sgn(CSBr$sc - 1)
-- Submodel: CSH2
d/dt CSH2$mu1 = mudotH2/sgn(CSH2$sc - 1)
-- Submodel: CSH
d/dt CSH$mu1 = mudotH/sgn(CSH$sc - 1)
-- Submodel: CSHBr
d/dt Cr$CBu12 = mudotHBr/sgn(CSHBr$sc - 1)
```

```
-- Submodel: reaction1
y1 = Ch2$nBr2/r1$dirty
y2 = ChBr$nBr/r1$dirty
y3 = ChH2$nH2/r1$dirty
y4 = ChR2H$nH/r1$dirty
y5 = ChB$nHBr/r1$dirty
------------------------------------------
    OUT
    dispt y1, y2, y3, y4, y5
------------------------------------------
/--
/PIC 'h9_ps.PRC            '
/--
```

B.3 Hydrogen–Bromine Reaction: Isobaric-isothermic Case

Following, the DYMOLA main program "*PT.dym*" including all the model types, the simulation control program "*PT.ctl*", and its generated DESIRE–program "*PT.des*" of the hydrogen–bromine reaction under isobaric–isothermic conditions are listed.

B.3.1 DYMOLA Main Program "*PT.dym*" Including All the Model Types

```
{ H2 -Br2 reaction isobaric and isothermic }

{ Bond Graph bond }

model type bond

  cut A (x / y) B (y / -x)
  main cut C [A B]
  main path P <A - B>

end
```

```
{ H2 - Br2 Reaction isobaric and isothermic }

model type Thermic

  main cut thermtot( Temp/-Sdot )
  cut thermk1( Temp/Sdotk1 ), thermk2( Temp/Sdotk2 )
  cut thermk3( Temp/Sdotk3 ), thermk4( Temp/Sdotk4 )
  cut thermk5( Temp/Sdotk5 )
  parameter dirty=1.0
  terminal S

  Sdot      = Sdotk1 + Sdotk2 + Sdotk3 + Sdotk4 + Sdotk5
  dirty*der(S) = Sdot

end


{ H2 - Br2 Reaction isobaric and isothermic }

model type Pneumatic

  main cut pneutot( p/phi )
  cut pneuk1( p/-phik1 ), pneuk2( p/-phik2 )
  cut pneuk3( p/-phik3 ), pneuk4( p/-phik4 )
  cut pneuk5( p/-phik5 )

  parameter dirty = 1.0
  terminal V

  phi       = phik1 + phik2 + phik3 + phik4 + phik5
  dirty*der(V) = phi

end

{ H2 - Br2 Reaction isobaric and isothermic }

model type Chpower

   cut chem1(muBr2/xiBr2), chem2(muBr/xiBr), chem3(muH2/xiH2)
   cut chem4(muH/xiH), chem5(muHBr/xiHBr)
   terminal Gdot, G
   parameter dirty = 1.0

   Gdot = muBr2*xiBr2 + muBr*xiBr + muH2*xiH2 + muH*xiH + muHBr*xiHBr
   dirty*der(G) = Gdot

end


{ H2 - Br2 reaction isobaric and isothermic }
```

```
model type TF

  cut A( mu1/xi1 ), B( mu2/-xi2 )
  main cut C [A B]
  main path P < A-B >
  parameter k = 1.0

  k*xi2 = xi1
  k*mu1 = mu2

end


{ H2 - Br2 Reaction isobaric and isothermic }

model type CS

  cut chem( mu/xi ), chem1(mu/xi)
  parameter dirty=1.0
  terminal n, mu0

  mu       = mu0
  dirty*der(n) = xi

end


{ H2 - Br2 Reaction, isobaric and isothermic }

model type ChRk1

  cut chem( muk1/-xik1 ), thermk1( Temp/-Sdotk1 ), pneuk1( p/phik1 )
  main cut C[ chem ]
  terminal nBr2, V
  parameter R=8.314
  local k1, xiek1, ak1

  ak1    = 1.39E+8*Temp**0.5*( 189243/( R*Temp ) )**1.97
  k1     = ak1*exp( -189243/( R*Temp ) )
  xiek1 = xik1
  p*phik1 = Temp*Sdotk1 + muk1*phik1
  p*phik1 = xiek1*R*Temp
  xik1   = k1*nBr2

end
```

```
{ H2 - Br2 Reaction, isobaric and isothermic }

model type ChRk2

  cut chem( muk2/-xik2 ), thermk2( Temp/-Sdotk2 ), pneuk2( p/phik2 )
  main cut C[ chem ]
  terminal nBr, V, K
  parameter R=8.314
  local k1, ak1, k2, xiek2

  ak1    = 1.39E+8*Temp**0.5*( 189243/( R*Temp ) )**1.97
  k1     = ak1*exp( -189243/( R*Temp ) )
  k2     = k1/K
  xiek2  = -xik2
  p*phik2  = Temp*Sdotk2 + muk2*xik2
  p*phik2  = xiek2*R*Temp
  xik2   = k2*nBr*nBr/V

end
```

```
{ H2 - Br2 Reaction, isobaric and isothermic }

model type ChRk3

  cut chem( muk3/-xik3 ), thermk3( Temp/-Sdotk3 ), pneuk3( p/phik3 )
  main cut C[chem]
  terminal nH2, nBr, V
  parameter R=8.314
  local k3, xiek3

  k3    = ( 10**11.43 )*exp( -82400/( R*Temp ) )
  xiek3 = 0.0
  p*phik3 = Temp*Sdotk3 + muk3*xik3
  p*phik3 = xiek3*R*Temp
  xik3  = k3*nH2*nBr/V

end
```

```
{ H2 - Br2 Reaction, isobaric and isothermic }

model type ChRk4

  cut chem( muk4/-xik4 ), thermk4( Temp/-Sdotk4 ), pneuk4( p/phik4 )
  main cut C[chem]
  terminal nHBr, nH, V
  parameter R=8.314, m=10.0
  local k4, k5, xiek4

  k5      = 10.0**11.97*exp( -149800.0/( R*Temp ) )
  k4      = k5/m
  xiek4 = 0.0
  p*phik4 = Temp*Sdotk4 + muk4*xik4
  p*phik4 = xiek4*R*Temp
  xik4    = k4*nHBr*nH/V

end


{ H2 - Br2 Reaction, isobaric and isothermic }

model type ChRk5

  cut chem( muk5/-xik5 ), thermk5( Temp/-Sdotk5 ), pneuk5( p/phik5 )
  main cut C[chem]
  terminal nBr2, nH, V
  parameter R=8.314
  local k5, xiek5

  k5      = 10.0**11.97*exp( -149800.0/( R*Temp ) )
  xiek5 = 0.0
  p*phik5 = Temp*Sdotk5 + muk5*xik5
  p*phik5 = xiek5*R*Temp
  xik5    = k5*nBr2*nH/V

end
```

```
{------------------------------------------------------------------------}

model reaction1

  submodel ChRk1
  submodel ChRk2
  submodel ChRk3
  submodel ChRk4
  submodel ChRk5
  submodel (CS)         CSBr2 (ic n = 0.5E-7), ->
                        CSBr  (ic n = 0.0),    ->
                        CSH2  (ic n = 0.5E-7), ->
                        CSH   (ic n = 0.0),    ->
                        CSHBr (ic n = 0.0)
  submodel Pneumatic    (ic V = 1.0E-3)
  submodel Thermic      (ic S = 0.0)
  submodel Chpower      (ic G = 0.0)
  submodel (TF)         TF1(k = 0.5), TF2(k = 2.0)
  submodel (bond)       B11, B1Br2, B1Br, B21, B2Br2, B2Br, B31
  submodel (bond)       B3H2, B3H, B3HBr, B3Br, B41, B4H, B4HBr
  submodel (bond)       B4Br, B4H2, B51, B5Br, B5Br2, B5H, B5HBr


  input    Temp0, p0
  output   y1, y2, y3, y4, y5
  parameter dirty = 1.0
  local    K900
  node     n11, n21, n31, n41, n51, n1Br, n2Br, nBr2, nBr, ->
           nH2, nH, nHBr, n1, n2, n3, n4, n5


  connect ChRk1:chem at   n11,      ->
          B11   from n11  to n1,    ->
          B1Br2 from nBr2 to n1,    ->
          B1Br  from n1   to n1Br,  ->
          TF1   from n1Br to nBr

  connect ChRk2:chem at   n21,      ->
          B21   from n21  to n2,    ->
          B2Br  from n2Br to n2,    ->
          B2Br2 from n2   to nBr2,  ->
          TF2   from nBr  to n2Br
```

```
connect ChRk3:chem at   n31,       ->
        B31   from n31  to n3,     ->
        B3HBr from n3   to nHBr,   ->
        B3H   from n3   to nH,     ->
        B3H2  from nH2  to n3,     ->
        B3Br  from nBr  to n3

connect ChRk4:chem at   n41,       ->
        B41   from n41  to n4,     ->
        B4H   from nH   to n4,     ->
        B4HBr from nHBr to n4,     ->
        B4Br  from n4   to nBr,    ->
        B4H2  from n4   to nH2

connect ChRk5:chem at   n51,       ->
        B51   from n51  to n5,     ->
        B5Br  from n5   to nBr,    ->
        B5Br2 from nBr2 to n5,     ->
        B5H   from nH   to n5,     ->
        B5HBr from n5   to nHBr


connect CSBr2 at nBr2, ->
        CSBr  at nBr,  ->
        CSH2  at nH2,  ->
        CSH   at nH,   ->
        CSHBr at nHBr

connect ChRk1:pneuk1 at Pneumatic:pneuk1, ->
        ChRk2:pneuk2 at Pneumatic:pneuk2, ->
        ChRk3:pneuk3 at Pneumatic:pneuk3, ->
        ChRk4:pneuk4 at Pneumatic:pneuk4, ->
        ChRk5:pneuk5 at Pneumatic:pneuk5

connect ChRk1:thermk1 at Thermic:thermk1, ->
        ChRk2:thermk2 at Thermic:thermk2, ->
        ChRk3:thermk3 at Thermic:thermk3, ->
        ChRk4:thermk4 at Thermic:thermk4, ->
        ChRk5:thermk5 at Thermic:thermk5
```

```
  connect Chpower:chem1 at CSBr2:chem1, ->
          Chpower:chem2 at CSBr:chem1,  ->
          Chpower:chem3 at CSH2:chem1,  ->
          Chpower:chem4 at CSH:chem1,   ->
          Chpower:chem5 at CSHBr:chem1

  func K900   = KBr2Bt(Temp0)

  ChRk1.nBr2 = CSBr2.n
  ChRk1.V    = Pneumatic.V


  ChRk2.nBr  = CSBr.n
  ChRk2.V    = Pneumatic.V
  ChRk2.K    = K900

  ChRk3.nH2  = CSH2.n
  ChRk3.nBr  = CSBr.n
  ChRk3.V    = Pneumatic.V

  ChRk4.nHBr = CSHBr.n
  ChRk4.nH   = CSH.n
  ChRk4.V    = Pneumatic.V

  ChRk5.nBr2 = CSBr2.n
  ChRk5.nH   = CSH.n
  ChRk5.V    = Pneumatic.V

  CSBr2.mu0 = -204493.0
  CSBr.mu0  = -53772.8
  CSH2.mu0  = -129023.0
  CSH.mu0   = 106772.0
  CSHBr.mu0 = -226828.0

  Thermic.Temp = Temp0
  Pneumatic.p  = p0

  dirty*y1 = CSBr2.n
  dirty*y2 = CSBr.n
  dirty*y3 = CSH2.n
  dirty*y4 = CSH.n
  dirty*y5 = CSHBr.n

end
```

B.3.2 Simulation Control Program "*PT.ctl*"

```
 { H2 - Br2 reaction isobaric and isothermic }

 cmodel

   simutime    5000.0
   step        1.0E-5
   commupoints 5001


   input 2, Temp0(depend, 800.0),  ->
            p0(depend, 1.0E+5)

   ctblock

     { Runge-Kutta (4/2) }
     irule 4
     DTMAX=0.1
     ERMAX=1.0E-10
     disconnect 1 | connect 'h9ptgd.gr' as output 1

     dimension KBr2Bt[38]

    data      298.15, 300.0, 400.0, 500.0, 600.0, 700.0
    data      800.0, 900.0, 1000.0, 1100.0, 1200.0, 1300.0
    data      1400.0, 1500.0, 1600.0, 1700.0, 1800.0, 1900.0
    data      2000.0
    data      3.1623E-29, 7.7446E-29, 1.9543E-20, 2.2182E-15
    data      5.2844E-12, 1.3867E-9,  9.0782E-8,  2.3768E-6
    data      3.2509E-5,  2.7861E-4,  1.6788E-3,  7.6913E-3
    data      2.8510E-2,  8.8716E-2,  2.4044E-1,  5.8344E-1
    data      1.7947   ,  2.6061   ,  4.9431

    read KBr2Bt

    scale = 1.0E-7
    XCCC  = 1
    label TRY
    drunr | if XCCC < 0 then XCCC = -XCCC | scale = 2*scale | go to TRY
                      else proceed
    disconnect  1 | connect 'user' as output 1

  ctend

  outblock
    OUT
    dispt y1, y2, y3, y4, y5
  outend

end
```

B.3.3 Generated DESIRE–Program "*PT.des*"

```
----------------------------------------
-- CONTINUOUS SYSTEM reaction1
----------------------------------------
-- STATE S Pnetic$V G CSBr2$n CSBr$n CSH2$n CSH$n CSHBr$n
-- DER dS  dPneic$V  dG  dCSBr2$n  dCSBr$n  dCSH2$n  dCSH$n  dCSHBr$n
-- OUTPUT y1 y2 y3 y4 y5
-- INPUT Temp0 p0
-- PARAMETERS and CONSTANTS:
Tc$dirty=1.0
Pc$dirty=1.0
Cr$C$ty1=1.0
TF1$k=0.5
C2$C$ty1=1.0
ChRk1$R=8.314
ChRk2$R=8.314
ChRk3$R=8.314
ChRk4$R=8.314
m=10.0
ChRk5$R=8.314
Cr$C$ty2=1.0
C2$C$ty2=1.0
CH$dirty=1.0
Cr$C$ty3=1.0
TF2$k=2.0
r1$dirty=1.0
-- INITIAL VALUES OF STATES:
S=0.0
Pnetic$V=1.0E-3
G=0.0
CSBr2$n=0.5E-7
CSBr$n=0.0
CSH2$n=0.5E-7
CSH$n=0.0
CSHBr$n=0.0
```

```
-----------------------------------------
TMAX=5000.0 | DT=1.0E-5 | NN=5001
    irule 4
    DTMAX=0.1
    ERMAX=1.0E-10
    disconnect 1 | connect 'h9ptgd.gr' as output 1
    dimension KBr2Bt[38]
    data      298.15, 300.0, 400.0, 500.0, 600.0, 700.0
    data      800.0, 900.0, 1000.0, 1100.0, 1200.0, 1300.0
    data      1400.0, 1500.0, 1600.0, 1700.0, 1800.0, 1900.0
    data      2000.0
    data      3.1623E-29, 7.7446E-29, 1.9543E-20, 2.2182E-15
    data      5.2844E-12, 1.3867E-9,  9.0782E-8,  2.3768E-6
    data      3.2509E-5,  2.7861E-4,  1.6788E-3,  7.6913E-3
    data      2.8510E-2,  8.8716E-2,  2.4044E-1,  5.8344E-1
    data      1.7947    ,  2.6061   ,  4.9431
    read KBr2Bt
    scale = 1.0E-7
    XCCC  = 1
    label TRY
    drunr | if XCCC < 0 then XCCC = -XCCC | scale = 2*scale | go to TRY
                          else proceed
    disconnect  1 | connect 'user' as output 1
-----------------------------------------
DYNAMIC
-----------------------------------------
Temp0 = 800.0
p0 = 1.0E+5
-- Submodel: reaction1
muBr = -53772.8
-- Submodel: TF1
```

```
TF1$mu1 = muBr/TF1$k
-- Submodel: reaction1
muBr2 = -204493.0
B11$x = TF1$mu1 - muBr2
-- Submodel: ChRk1
Chk1$ak1 = 1.39E+8*Temp0^0.5*(189243/(ChRk1$R*Temp0))^1.97
ChRk1$k1 = Chk1$ak1*exp(-189243/(ChRk1$R*Temp0))
B1Br$x = ChRk1$k1*CSBr2$n
Pc$phik1 = B1Br$x*ChRk1$R*Temp0/p0
C1$Sdok1 = (p0*Pc$phik1 - B11$x*Pc$phik1)/Temp0
-- Submodel: ChRk2
Chk2$ak1 = 1.39E+8*Temp0^0.5*(189243/(ChRk2$R*Temp0))^1.97
ChRk2$k1 = Chk2$ak1*exp(-189243/(ChRk2$R*Temp0))
-- Submodel: reaction1
func K900 = KBr2Bt,Temp0
-- Submodel: ChRk2
k2 = ChRk2$k1/K900
TF2$xi2 = k2*CSBr$n*CSBr$n/Pnetic$V
xiek2 = -TF2$xi2
Pc$phik2 = xiek2*ChRk2$R*Temp0/p0
-- Submodel: TF2
TF2$mu2 = TF2$k*muBr
-- Submodel: reaction1
B21$x = muBr2 - TF2$mu2
-- Submodel: ChRk2
C2$Sdok2 = (p0*Pc$phik2 - B21$x*TF2$xi2)/Temp0
-- Submodel: ChRk3
xiek3 = 0
Pc$phik3 = xiek3*ChRk3$R*Temp0/p0
-- Submodel: reaction1
muH = 106772.0
muHBr = -226828.0
muH2 = -129023.0
B31$x = muH + muHBr - (muBr + muH2)
```

```
-- Submodel: ChRk3
k3 = 10^11.43*exp(-82400/(ChRk3$R*Temp0))
B3Br$y = k3*CSH2$n*CSBr$n/Pnetic$V
C3$Sdok3 = (p0*Pc$phik3 - B31$x*B3Br$y)/Temp0
-- Submodel: ChRk4
xiek4 = 0
Pc$phik4 = xiek4*ChRk4$R*Temp0/p0
-- Submodel: reaction1
B41$x = muH2 + muBr - (muHBr + muH)
-- Submodel: ChRk4
ChRk4$k5 = 10.0^11.97*exp(-149800.0/(ChRk4$R*Temp0))
k4 = ChRk4$k5/m
B4H2$x = k4*CSHBr$n*CSH$n/Pnetic$V
C4$Sdok4 = (p0*Pc$phik4 - B41$x*B4H2$x)/Temp0
-- Submodel: ChRk5
xiek5 = 0
Pc$phik5 = xiek5*ChRk5$R*Temp0/p0
-- Submodel: reaction1
B51$x = muHBr + muBr - (muH + muBr2)
-- Submodel: ChRk5
ChRk5$k5 = 10.0^11.97*exp(-149800.0/(ChRk5$R*Temp0))
B5HBr$x = ChRk5$k5*CSBr2$n*CSH$n/Pnetic$V
C5$Sdok5 = (p0*Pc$phik5 - B51$x*B5HBr$x)/Temp0
-- Submodel: Thermic
Sdot = C1$Sdok1 + C2$Sdok2 + C3$Sdok3 + C4$Sdok4 + C5$Sdok5
d/dt S = Sdot/Tc$dirty
-- Submodel: Pneumatic
phi = Pc$phik1 + Pc$phik2 + Pc$phik3 + Pc$phik4 + Pc$phik5
d/dt Pnetic$V = phi/Pc$dirty
-- Submodel: reaction1
CSBr2$xi = TF2$xi2 - (B5HBr$x + B1Br$x)
xiBr2 = -CSBr2$xi
```

```
-- Submodel: TF2
TF2$xi1 = TF2$k*TF2$xi2
-- Submodel: TF1
TF1$xi2 = B1Br$x/TF1$k
-- Submodel: reaction1
CSBr$xi = B5HBr$x + B4H2$x + TF1$xi2 - (B3Br$y + TF2$xi1)
xiBr = -CSBr$xi
CSH2$xi = B4H2$x - B3Br$y
xiH2 = -CSH2$xi
CSH$xi = B3Br$y - (B5HBr$x + B4H2$x)
xiH = -CSH$xi
CSHBr$xi = B5HBr$x + B3Br$y - B4H2$x
xiHBr = -CSHBr$xi
-- Submodel: Chpower
Gdot = muBr2*xiBr2 + muBr*xiBr + muH2*xiH2 + muH*xiH + muHBr*xiHBr
d/dt G = Gdot/Cr$C$ty1
-- Submodel: CSBr2
d/dt CSBr2$n = CSBr2$xi/C2$C$ty1
-- Submodel: CSBr
d/dt CSBr$n = CSBr$xi/Cr$C$ty2
-- Submodel: CSH2
d/dt CSH2$n = CSH2$xi/C2$C$ty2
-- Submodel: CSH
d/dt CSH$n = CSH$xi/CH$dirty
-- Submodel: CSHBr
d/dt CSHBr$n = CSHBr$xi/Cr$C$ty3
-- Submodel: reaction1
y1 = CSBr2$n/r1$dirty
y2 = CSBr$n/r1$dirty
y3 = CSH2$n/r1$dirty
y4 = CSH$n/r1$dirty
y5 = CSHBr$n/r1$dirty
----------------------------------------
    OUT
    dispt y1, y2, y3, y4, y5
----------------------------------------
/--
/PIC 'h9_pt.PRC          '
/--
```

Appendix C

Programs for the Model Validation

This appendix contains useful tables of the "Computer Program for Calculation of Complex Chemical Equilibrium Compositions, Rocket Performance, Incident and Reflected Shocks, and Chapman–Jouget Detonations", the input data programs and the output files of the three cases.

C.1 Tables of the "Combustion Program"

This section includes the tables to which Chapter refers.

C.1.1 Table 1: Order and Format of the Input Data Cards, Reactants Cards

Table 1 includes two tables.

C.1.1.1 Table 1a: Order and Format of the Input Data Cards

Card order	Contents	Format	Card column
1	THERMO	3A4	1 to 6
2	Temperature ranges for 2 sets of coefficients: lowest T, common T, and highest T	3F10.3	1 to 30
3	Species name	3A4	1 to 12
	Date	2A3	19 to 24
	Atomic symbols and formula	4(A2, F3.0)	25 to 44
	Phase of species (S, L, or G for solid, liquid, or gas, respectively)	A1	45
	Temperature range	2F10.3	46 to 65
	Integer 1	I15	80
4	Coefficients a_i(i = 1 to 5) in equations (90) to (92) (for upper temperature interval)	5(E15.8)	1 to 75
	Integer 2	I5	80
5	Coefficients in equations (90) to (92) (a_6, a_7 for upper temperature interval, and a_1, a_2, and a_3 for lower)	5(E15.8)	1 to 75
	Integer 3	I5	80
6	Coefficients in equations (90) to (92) (a_4, a_5, a_6, a_7 for lower temperature interval)	4(E15.8)	1 to 60
	Integer 4	I20	80
(a)	Repeat cards numbered 1 to 4 in cc 80 for each species		
(Final card)	END (Indicates end of thermodynamic data)	3A4	1 to 3

[a] Gaseous species and condensed species with only one condensed phase can be in any order. However, the sets for two or more condensed phases of the same species must be adjacent. If there are more than two condensed phases of a species, their sets must be either in increasing or decreasing order according to their temperature intervals.

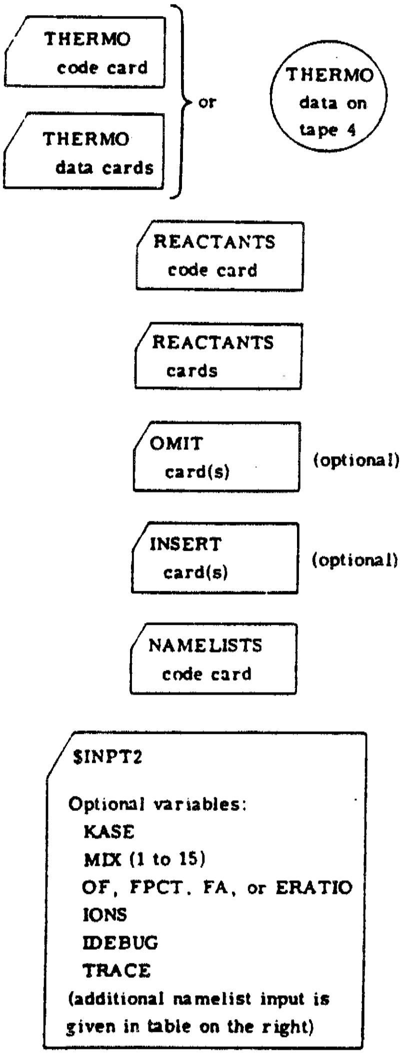
C.1.1.2 Table 1b: Reactants Cards

Order	Contents	Format	Card columns
First	REACTANTS	3A4	1 to 9
Any	One card for each reactant species (maximum 15). Each card contains:		
	(1) Atomic symbols and formula numbers (maximum 5 sets)[a]	5(A2, F7.5)	1 to 45
	(2) Relative weight[b] or number of moles	F7.5	46 to 52
	(3) Blank if (2) is relative weight or M if (2) is number of moles	A1	53
	(4) Enthalpy or internal energy[a], cal mole	F9.5	54 to 62
	(5) State: S, L, or G for solid, liquid or gas, respectively	A1	63
	(6) Temperature associated with enthalpy in (4)	F7.0	64 to 70
	(6a) J if (4) is in units of kJ/kg-mole and blank if (4) is in units of cal/g-mole	A1	71
	(7) F if fuel or O if oxidant	A1	72
	(8) Density in g/cm^3 (optional)	F8.5	73 to 80
Last	Blank		

[a] Program will calculate the enthalpy or internal energy (4) for species in the THERMO data at the temperature (6) if zeros are punched in card columns 37 and 38. (See section Reactant enthalpy for additional information.)

[b] Relative weight of fuel in total fuels or oxidant in total oxidants. All reactants must be given either all in relative weights or all in number of moles.

C.1.2 Table 2: Program Input



Problem	Namelist	Variables	
		Required	Optional

Assigned temperature and pressure (TP)	INPT2	TP = .TRUE.
		T(1 to 26)
		P(1 to 26)
			NSQM, PSIA, or MMHG

Assigned enthalpy and pressure (HP)	INPT2	HP = .TRUE.
		P(1 to 26)
			NSQM, PSIA, or MMHG

Assigned entropy and pressure (SP)	INPT2	SP = .TRUE.
		S0(1)
		P(1 to 26)
			NSQM, PSIA, or MMHG

Assigned temperature and volume or density (TV)	INPT2	TV = .TRUE.
		T(1 to 26)
		V(1 to 26) or
		RHO(1 to 26)	

Assigned internal energy and volume or density (UV)	INPT2	UV = .TRUE.
		V(1 to 26) or
		RHO(1 to 26)	

Assigned entropy and volume or density (SV)	INPT2	SV = .TRUE.
		V(1 to 26) or
		RHO(1 to 26)
		S0(1)	

Detonation (DETN)	INPT2	DETN = .TRUE. P(1 to 26) [initial gas]	NSQM, PSIA, or MMHG T(1 to 26)[initial gas]

Shock (SHOCK)	INPT2	SHOCK = .TRUE.
		P(1 to 13)
		T(1 to 13) [initial gas]
			NSQM, PSIA, or MMHG

	SHKINP	U1(1 to 13) or MACH1 (1 to 13)	INCDEQ = .FALSE. or INCDFZ = .FALSE.

Rocket (RKT)	INPT2	RKT = .TRUE.
		P(1 to 26)(chamber pressures)
			T(1 to 26)(chamber) NSQM, PSIA, or MMHG

	RKTINP		EQL = .FALSE. or FROZ = .FALSE.
			PCP(1 to 22)
			SUPAR(1 to 13)
			SUBAR(1 to 13)

C.1.3 Table 3: Variables in Inpt2 Namelist

Variable	Dimension	Type	Common label	Value before read	Definition and comments
KASE	1	I	INDX	0	Optional assigned number associated with case
P	26	R	POINTS	0	Assigned pressures; chamber pressures for rocket problems; values in atm unless PSIA, NSQM or MMHG = T (see below)
NSQM	1	L	-------	False	[a]Values in P array are in N/m^2
PSIA	1	L	-------	False	[a]Values in P array are in psia units
MMHG	1	L	-------	False	[a]Values in P array are in mm Hg units
V	26	R	POINTS	0	Volume, cm^3/g
RHO	26	R	POINTS[b](P)	0	Density, g/cm^3
T	26	R	POINTS	0	Assigned temperature, K
MIX	15	R	MISC[b](OXF)	0	Values of equivalence ratios if ERATIO = T; oxidant to fuel weight ratio if OF = T; percent fuel by weight if FPCT = T; and fuel to air weight ratio if FA = T
ERATIO	1	L	MISC	False	Equivalence ratios are given in MIX[a]
OF	1	L	MISC	False	Oxidant to fuel weight ratios are given in MIX[a]
FPCT	1	L	MISC	False	Percent fuel by weight are given in MIX[a]
FA	1	L	-----	False	Fuel to air weight ratios are given in MIX[a]
TRACE	1	R	MISC	0 (5.E-9 for SHOCK problem)	Option to print mole fractions ≥ TRACE in special E-format
IONS	1	L	INDX	False	Consider ionic species[a]
IDEBUG	1	I	INDX	0	Print intermediate output for all points indexed ≥ integer value
TP	1	L	INDX	False	Assigned temperature and pressure problem[a]
HP	1	L	INDX	False	Assigned enthalpy and pressure problem[a]
SP	1	L	INDX	False	Assigned entropy (S0) and pressure problem[a]
S0	1	R	MISC	0	Assigned entropy, cal/(g)(K)
TV	1	L	INDX	False	Assigned temperature and volume (or density) problem[a]
UV	1	L	INDX	False	Assigned internal energy and volume (or density) problem[a]
SV	1	L	INDX	False	Assigned entropy (S0) and volume (or density) problem[a]
RKT	1	L	----------	False	Rocket problem[a]
DETN	1	L	----------	False	Detonation problem[a]
SHOCK	1	L	INDX	False	Shock problem[a]
SIUNIT	1	L	----------	False	If true, the output tables will be in SI units

[a] If variable is set to be true.

[b] Equivalenced to variable given in parentheses.

C.2 Input Data Programs, Output files

This section contains the input data programs and the output files obtained by the "combustion program" for the three cases.

C.2.1 Isochoric–Isothermic Case

The isochoric–isothermic case is shown.

C.2.1.1 Input Data Program

```
thermo                                                                  *
reactants                                                               *
h 2.                                          100.    3512.0    g800.0    f
br 2.                                         100.    11834.    g800.0    o
                                                                        *
omit           br2(1)                                                   *
namelists                                                               *
&inpt2 tv t nsqm t v 1.0E-3 of t mix 79.904 trace 1.0E-12
       temper 800 kase 50 eta 1.0
&end
stop?
```

C.2.1.2 Output File

```
           THERMOCHEM INPUTS:

The Date is  05/10/1990
The Time is  14:15:02

thermo                                                                  *
reactants                                                               *
h     2.00         .00          .00          .00          .00
   100.000       3512.00   g     800.00      f      .0000
br    2.00         .00          .00          .00          .00
   100.000      11834.00   g     800.00      o      .0000
omit           br2(1)                                                   *
namelists                                                               *
$INPT2
   kase    =      50
   idebug  =       0
   ifuel   =       3

$END
```

```
 This version was created on Aug 10 1984
 Modified for use on IBM / MSDOS PC  Oct, 1988


Species being considered in this system:

     j 6/74  br                j12/61  br2               j 3/77  h
     j 9/65  hbr               j 3/77  h2


 Oxy to Fuel Ratio =  79.904000

                  Effective Fuel     Effective Oxidant       Mixture

 Internal Energy    Hpp(2)               Hpp(1)               Hsub0
 (kg-mol)(deg K)/kg .8767E+03          .3726E+02            .4764E+02
 kg-atoms/kg        b0p(i,2)            b0p(i,1)              b0(i)

      h             .9921E+00          .0000E+00            .1226E-01
      br            .0000E+00          .1251E-01            .1236E-01

 Point     h         br        itn       t


  1     -6.795   -8.147   13.000  800.000


    Thermodynamic Equilibrium Properties
          at assigned Temperature and Volume

 Case No.   50

                    Reactant Properties

Chemical Formula     (See Note)
         Wt Fraction  Energy    State    Temp    Density
                      cal/mol            deg k    g/cc

fuel       h    2.000
             1.000     3512.000     g     800.00      .000
oxidant    br   2.000
             1.000    11834.000     g     800.00      .000

   O/F =79.904    Percent Fuel =    1.236      Equivalence Ratio =    .992
   Phi =   .992   Reactant Density =     .000
  Thermodynamic Properties:

  p, atm            *********
  T, deg K              800.0
  rho, g/cc         8.1228 3
  h, cal/g              -62.4
  s, cal/(g)(k)         .2877

  M, mol wt            81.228
  (dlV/dlP)T         -1.00000
  (dlV/dlT)P           1.0000
  cp, cal/(g)(k)        .0915
  Gamma (s)            1.3651
  Son Vel,m/sec         334.3


  Mole Fractions

  br                7.3788-9
  br2               3.9424-3
  hbr               9.9605-1
  h2                4.4517-6


     Additional products which were considered
 but whose mole fractions were less than  1.00000E-12
 for all assigned conditions:

  h

   Note: Weight fraction of fuel in total fuel
      and of oxidant in total oxidants.

  stop

      Thermochem Outputs Complete.
```

C.2.2 Isobaric–Isothermic Case

The isobaric–isothermic case is shown.

C.2.2.1 Input Data Program

```
thermo                                                                    *
reactants                                                                 *
h 2.                                            100.   3512.0   g800.0   f
br 2.                                           100.   11834.   g800.0   o
                                                                          *
omit          br2(1)                                                      *
namelists                                                                 *
&inpt2 tp t nsqm t pres 1.0E+5 of t mix 79.904 trace 1.0E-12
      temper 800 kase 50 eta 1.0
&end
stop
```

C.2.2.2 Output File

```
          THERMOCHEM INPUTS:

The Date is  05/10/1990
The Time is  14:08:03

thermo                                                                    *
reactants                                                                 *
h     2.00          .00          .00          .00          .00
   100.000      3512.00  g     800.00         f         .0000
br    2.00          .00          .00          .00          .00
   100.000     11834.00  g     800.00         o         .0000
omit          br2(1)                                                      *
namelists                                                                 *
$INPT2
   kase    =      50
   idebug  =       0
   ifuel   =       3


$END

This version was created on Aug 10 1984
Modified for use on IBM / MSDOS PC  Oct, 1988


Species being considered in this system:

    j 6/74  br              j12/61  br2             j 3/77  h
    j 9/65  hbr             j 3/77  h2


Oxy to Fuel Ratio =  79.904000
                   Effective Fuel     Effective Oxidant      Mixture

Enthalpy              Hpp(2)             Hpp(1)              Hsub0
(kg-mol)(deg K)/kg  .8767E+03          .3726E+02           .4764E+02
kg-atoms/kg           b0p(i,2)           b0p(i,1)             b0(i)

    h               .9921E+00          .0000E+00           .1226E-01
    br              .0000E+00          .1251E-01           .1236E-01
```

```
 Point     h         br        itn       t


  1    -14.649  -16.003   14.000  800.000


    Thermodynamic Equilibrium Properties
          at assigned Temperature and Pressure

 Case No.   50

                    Reactant Properties

 Chemical Formula      (See Note)
          Wt Fraction   Energy   State    Temp    Density
                        cal/mol           deg k    g/cc

 fuel       h    2.000
              1.000      3512.000     g     800.00      .000
 oxidant    br   2.000
              1.000     11834.000     g     800.00      .000

    O/F =79.904    Percent Fuel =    1.236      Equivalence Ratio =    .992
    Phi =    .992  Reactant Density =    .000
 Thermodynamic Properties:

 p, atm               .9869
 T, deg K             800.0
 rho, g/cc          1.2212-3
 h, cal/g             -62.4
 s, cal/(g)(k)        .6720

 M, mol wt           81.227
 (dlV/dlP)T        -1.00000
 (dlV/dlT)P         1.0001
 cp, cal/(g)(k)       .0916
 Gamma (s)          1.3647
 Son Vel,m/sec       334.3


 Mole Fractions

 br                 1.9007-5
 br2                3.9328-3
 hbr                9.9604-1
 h2                 4.4624-6


     Additional products which were considered
but whose mole fractions were less than  1.00000E-12
for all assigned conditions:

 h

  Note: Weight fraction of fuel in total fuel
     and of oxidant in total oxidants.

 stop

     Thermochem Outputs Complete.
```

C.2.3 Isentropic–Isobaric Case

The isentropic-isobaric case is shown.

C.2.3.1 Input Data Program

```
thermo                                                              *
reactants                                                           *
h 2.                                          100.   3512.0  g800.0  f
br 2.                                         100.   11834.  g800.0  o
                                                                    *
omit           br2(l)                                               *
namelists                                                           *
&inpt2 sp t nsqm t pres 1.0E5 of t mix 79.904 trace 1.0E-12
       kase 50 eta 1.0
&end
stop
```

C.2.3.2 Output File

```
           THERMOCHEM INPUTS:

 The Date is  06/05/1990
 The Time is  18:22:48

 thermo                                                             *
 reactants                                                          *
 h     2.00            .00          .00         .00          .00
    100.000        3512.00  g      800.00       f       .0000
 br    2.00            .00          .00         .00          .00
    100.000       11834.00  g      800.00       o       .0000
 omit            br2(l)                                             *
 namelists                                                          *
 $INPT2
    kase    =       50
    idebug  =        0
    ifuel   =        3


 $END

 This version was created on Aug 10 1984
 Modified for use on IBM / MSDOS PC  Oct, 1988


Species being considered in this system:

     j 6/74  br                j12/61  br2               j 3/77  h
     j 9/65  hbr               j 3/77  h2


 Oxy to Fuel Ratio =  79.904000

                      Effective Fuel     Effective Oxidant      Mixture

 Enthalpy                 Hpp(2)              Hpp(1)              Hsub0
 (kg-mol)(deg K)/kg   .8767E+03           .3726E+02           .4764E+02
 kg-atoms/kg             b0p(i,2)            b0p(i,1)             b0(i)

     h                 .9921E+00           .0000E+00           .1226E-01
     br                .0000E+00           .1251E-01           .1236E-01
 Point     h         br        itn      t


   1  ********* 3308.743   13.000     .375
0the temperature=   .3752E+00 is out of range for point    1


     Thermodynamic Equilibrium Properties
           at assigned Entropy and Pressure

 Case No.   50

                    Reactant Properties

 Chemical Formula      (See Note)
          Wt Fraction   Energy    State    Temp    Density
                        cal/mol            deg k    g/cc

 fuel       h    2.000
              1.000      3512.000     g     800.00      .000
 oxidant    br   2.000
              1.000     11834.000     g     800.00      .000

    O/F =79.904    Percent Fuel =     1.236      Equivalence Ratio =    .992
    Phi =    .992   Reactant Density =     .000


 Thermodynamic Properties:

 p, atm              .9869
 T, deg K               .4
 rho, g/cc        2.6039 0
 h, cal/g           -132.3
 s, cal/(g)(k)       .0000

 M, mol wt          81.228
 (dlV/dlP)T       -1.00000
 (dlV/dlT)P         1.0000
 cp, cal/(g)(k)      .0882
 Gamma (s)          1.3837
 Son Vel,m/sec         7.3


 Mole Fractions

 br2              3.9379-3
 hbr              9.9606-1


     Additional products which were considered
but whose mole fractions were less than  1.00000E-12
for all assigned conditions:

 br            h            h2

  Note: Weight fraction of fuel in total fuel
     and of oxidant in total oxidants.

 stop
```

Appendix D

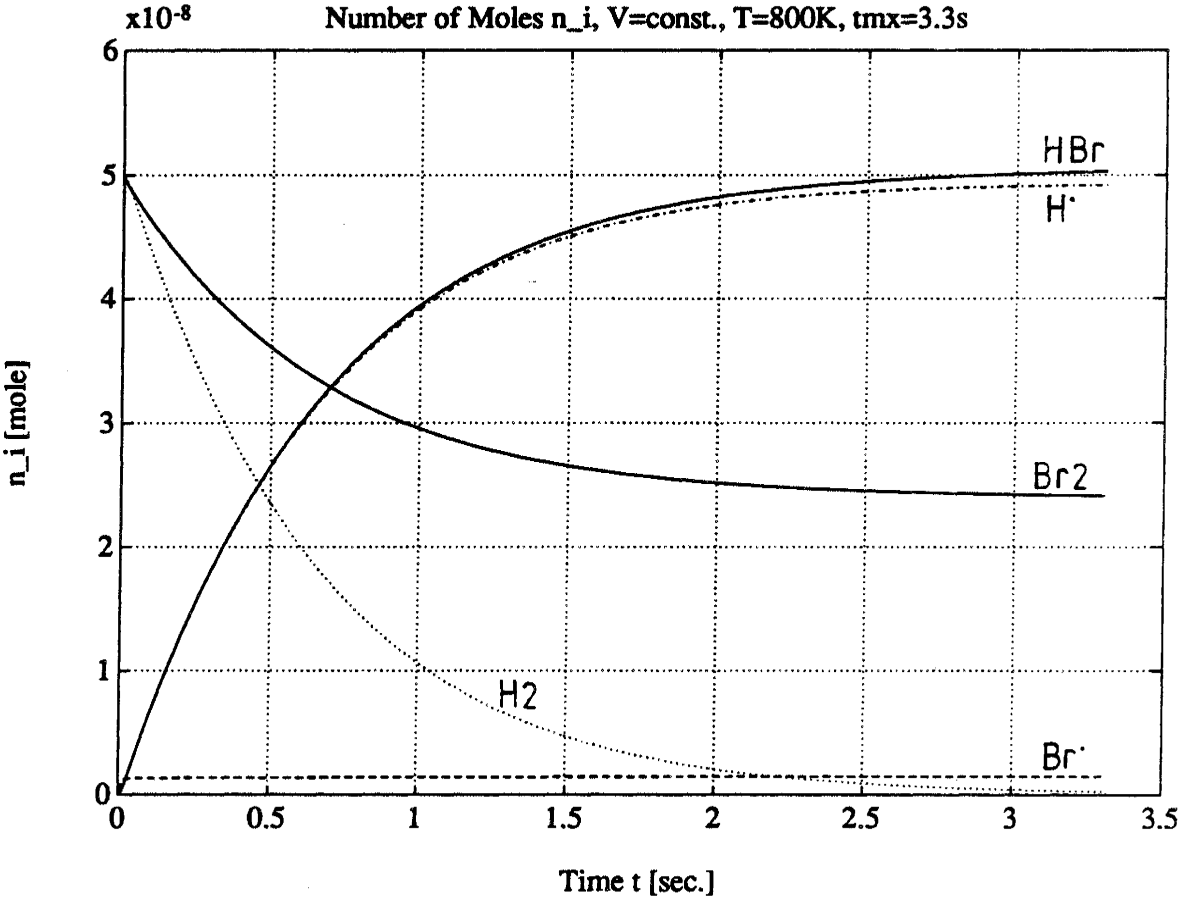
The Graphs of the Three Cases

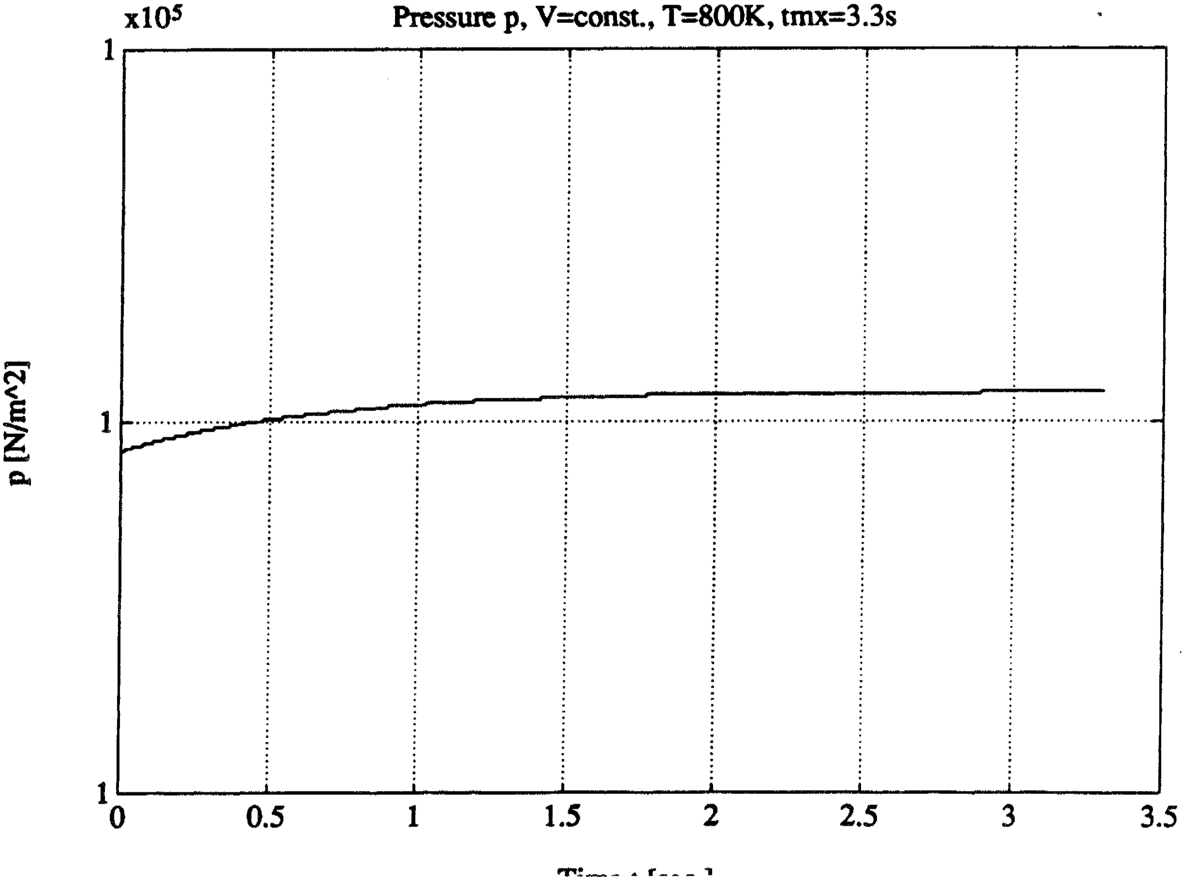
This appendix contains the graphs of the isochoric–isothermic, isobaric–isothermic, and isentropic–isobaric cases of the hydrogen–bromine reaction. The graphs show the change over time of the number of moles, the Gibbs' free energy, the reaction enthalpy, and the mole fractions. Also the graphs of total pressure, volume, and temperature are demonstrated if their values change over time. For the isochoric–isothermic case, the graphs are given for a simulation time both of $3.3 sec.$ and of $5000.0 sec.$.

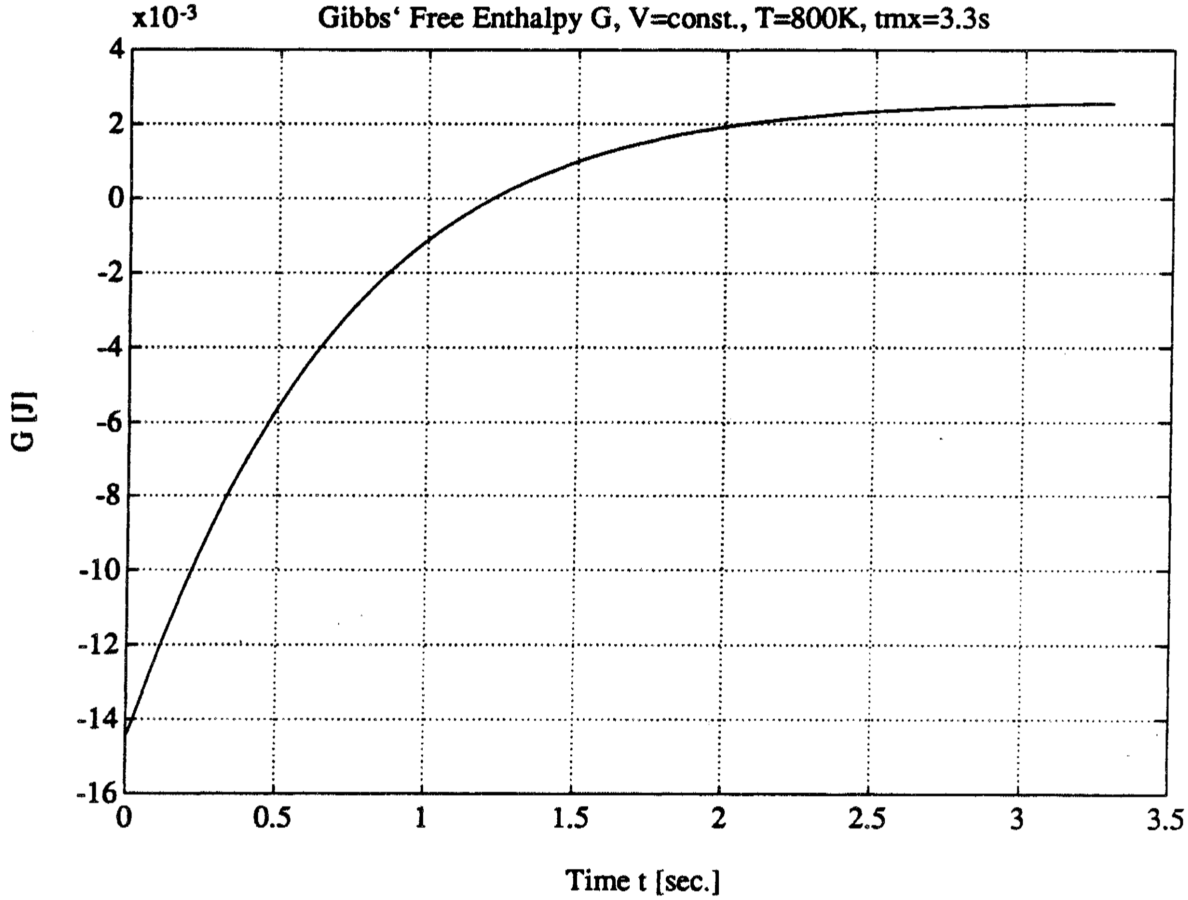
D.1 Isochoric–Isothermic Case

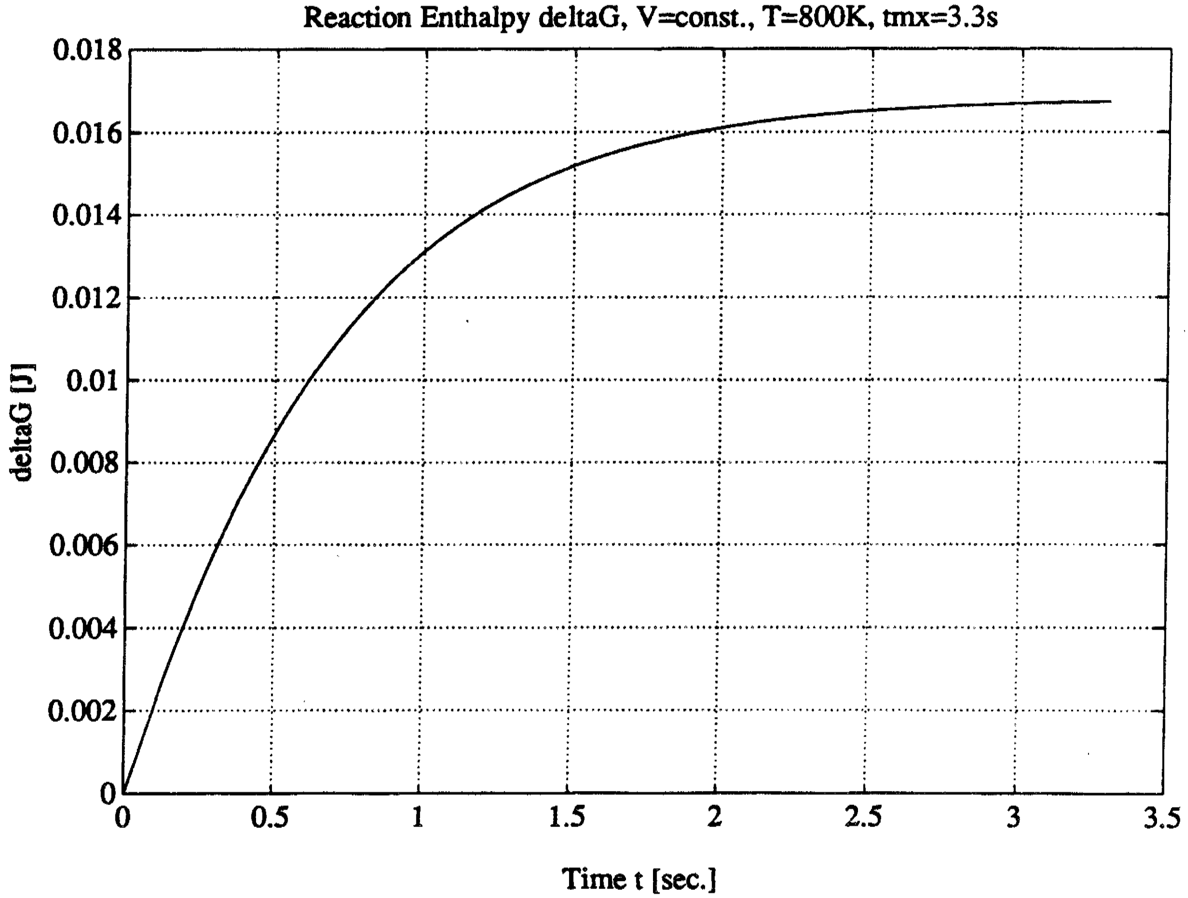
The graphs of isochoric–isothermic case are shown for both $tmx = 3.3sec$ and $tmx = 5000.0sec$.

D.1.1 Results for a Simulation Time of 3.3*sec.*

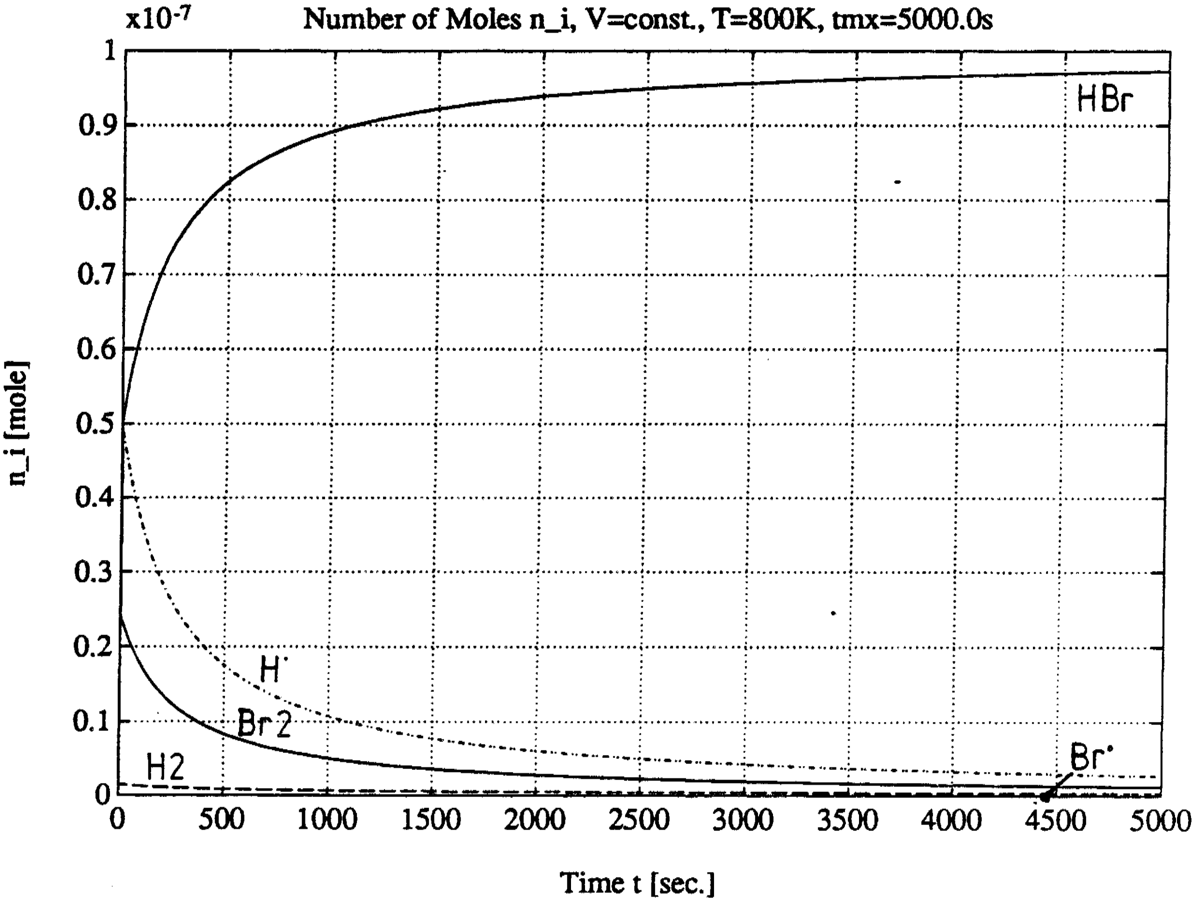


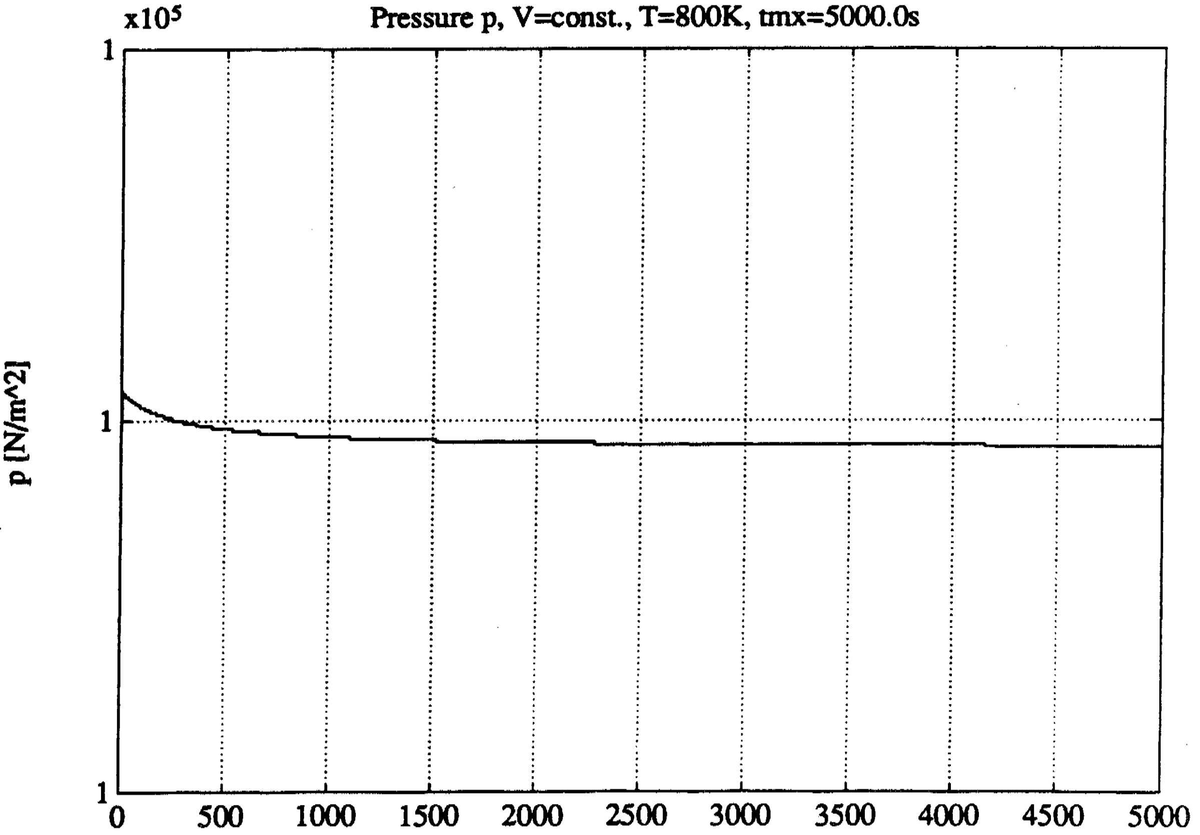


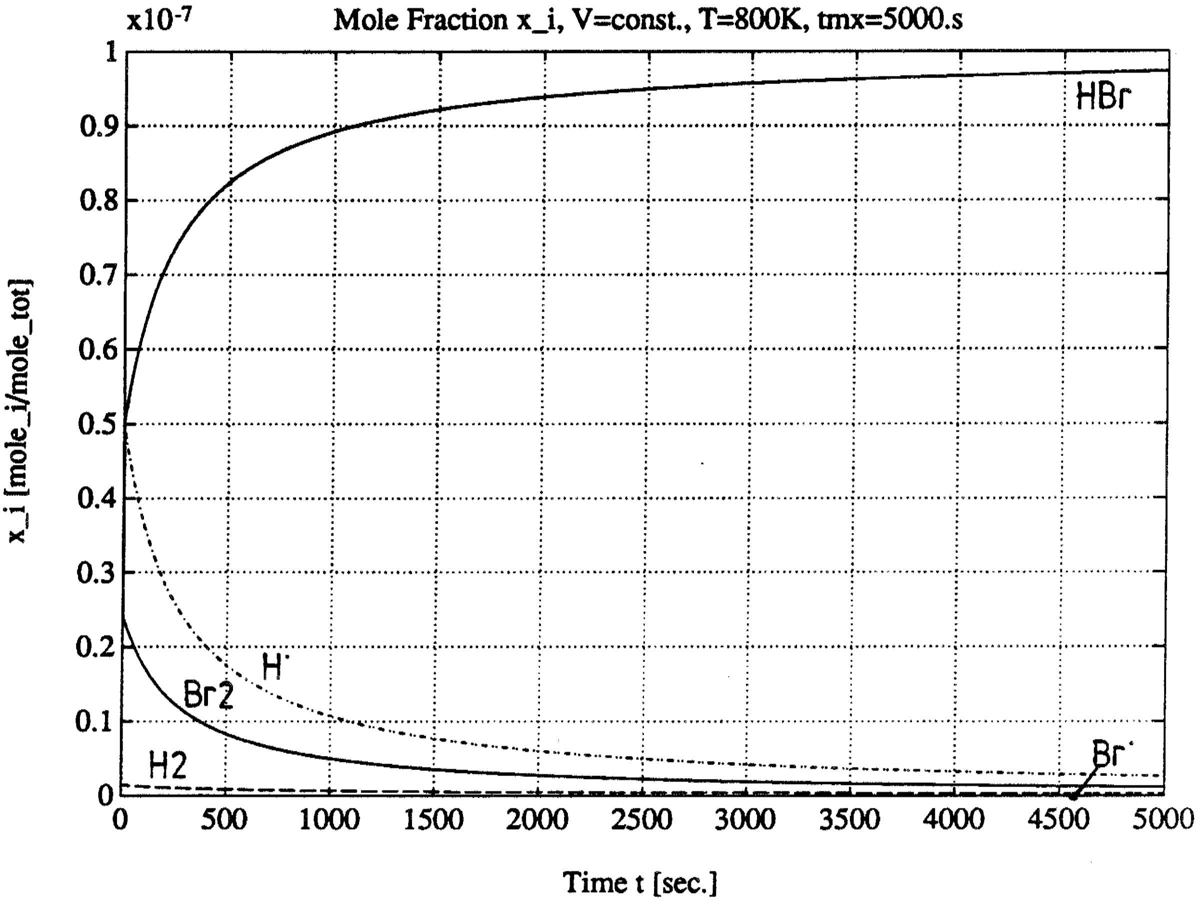

x10-3
Gibbs' Free Enthalpy G, V=const., T=800K, tmx=3.3s
4
2
0
-2
-4
-6
-8
-10
-12
-14
-16
G [J]
0
0.5
1
1.5
2
2.5
3
3.5
Time t [sec.]

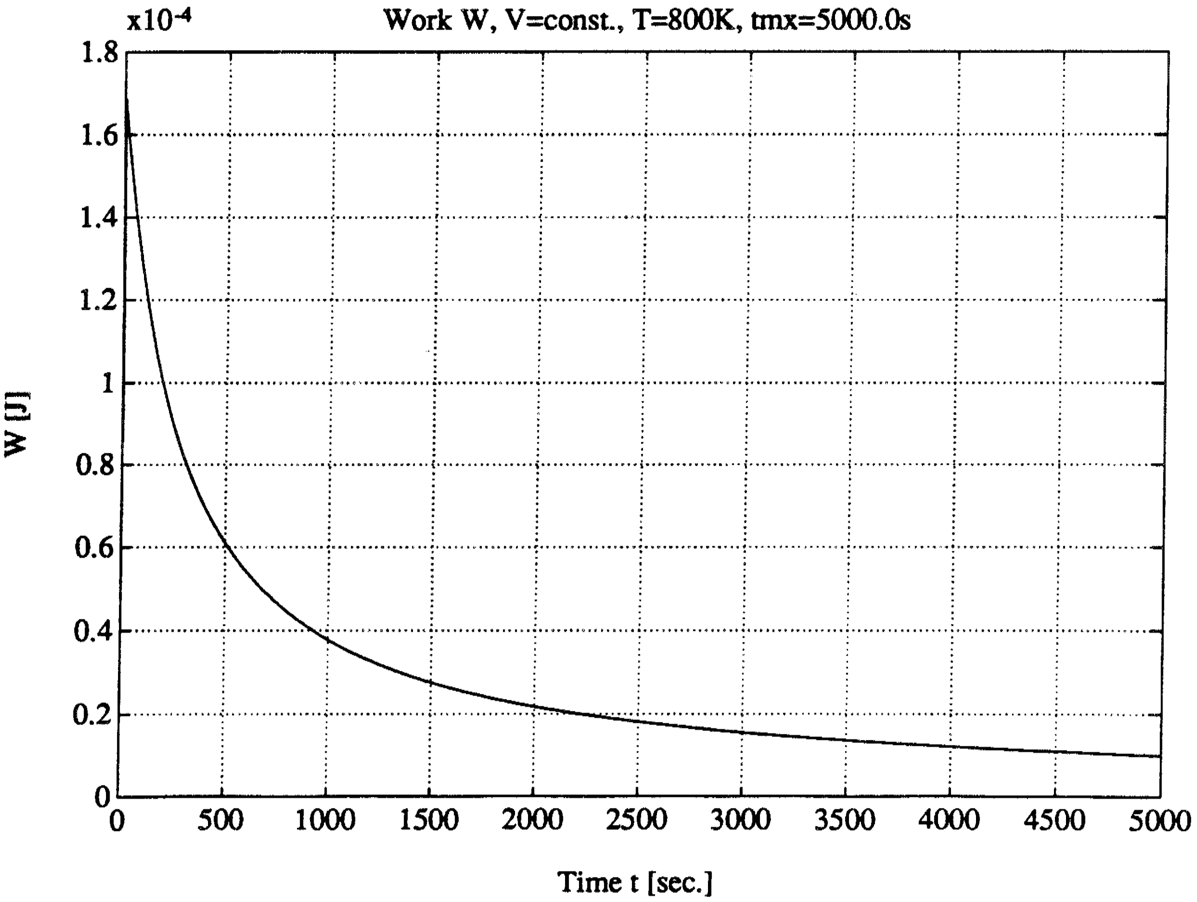

Reaction Enthalpy deltaG, V=const., T=800K, tmx=3.3s
0.018
0.016
0.014
0.012
0.01
0.008
0.006
0.004
0.002
0
deltaG [J]
0
0.5
1
1.5
2
2.5
3
3.5
Time t [sec.]

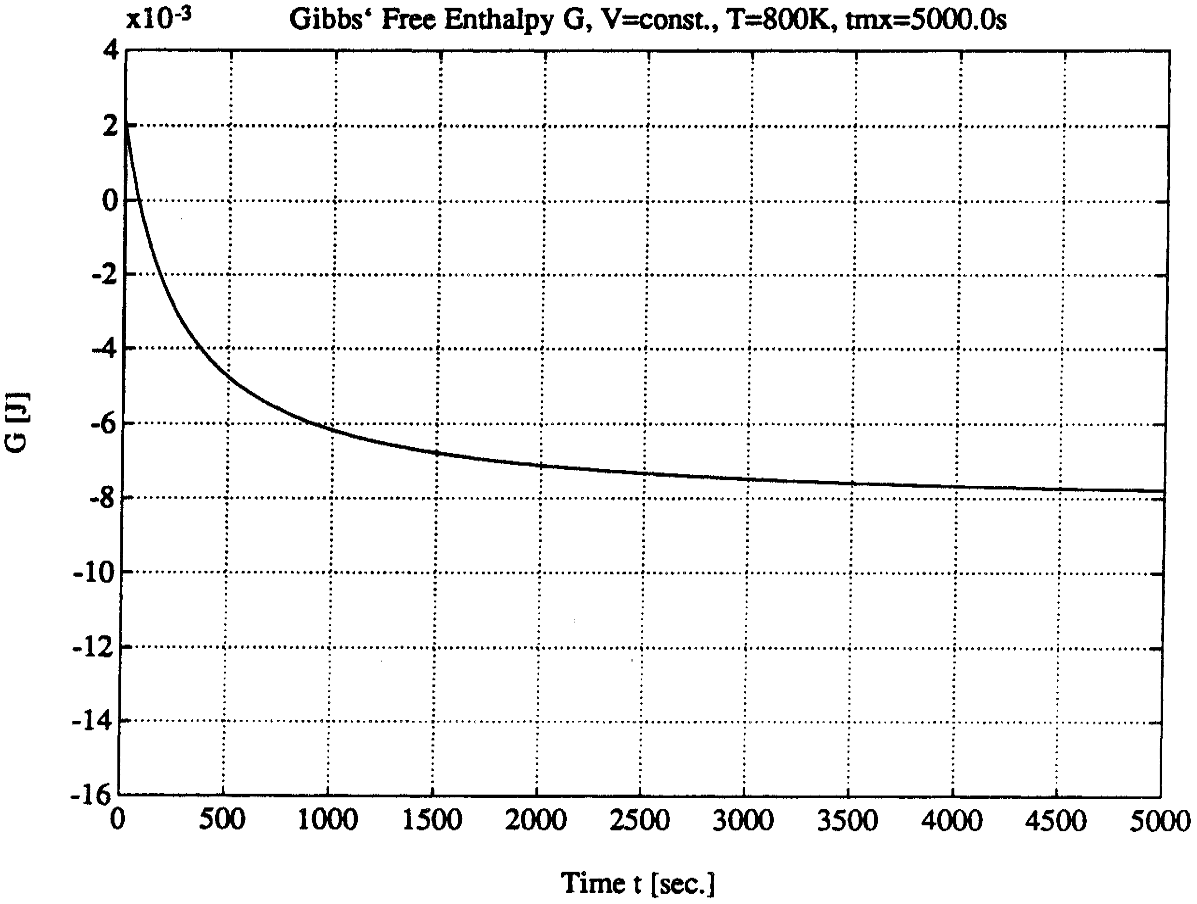
D.1.2 Results for a Simulation Time of 5000.0*sec.*

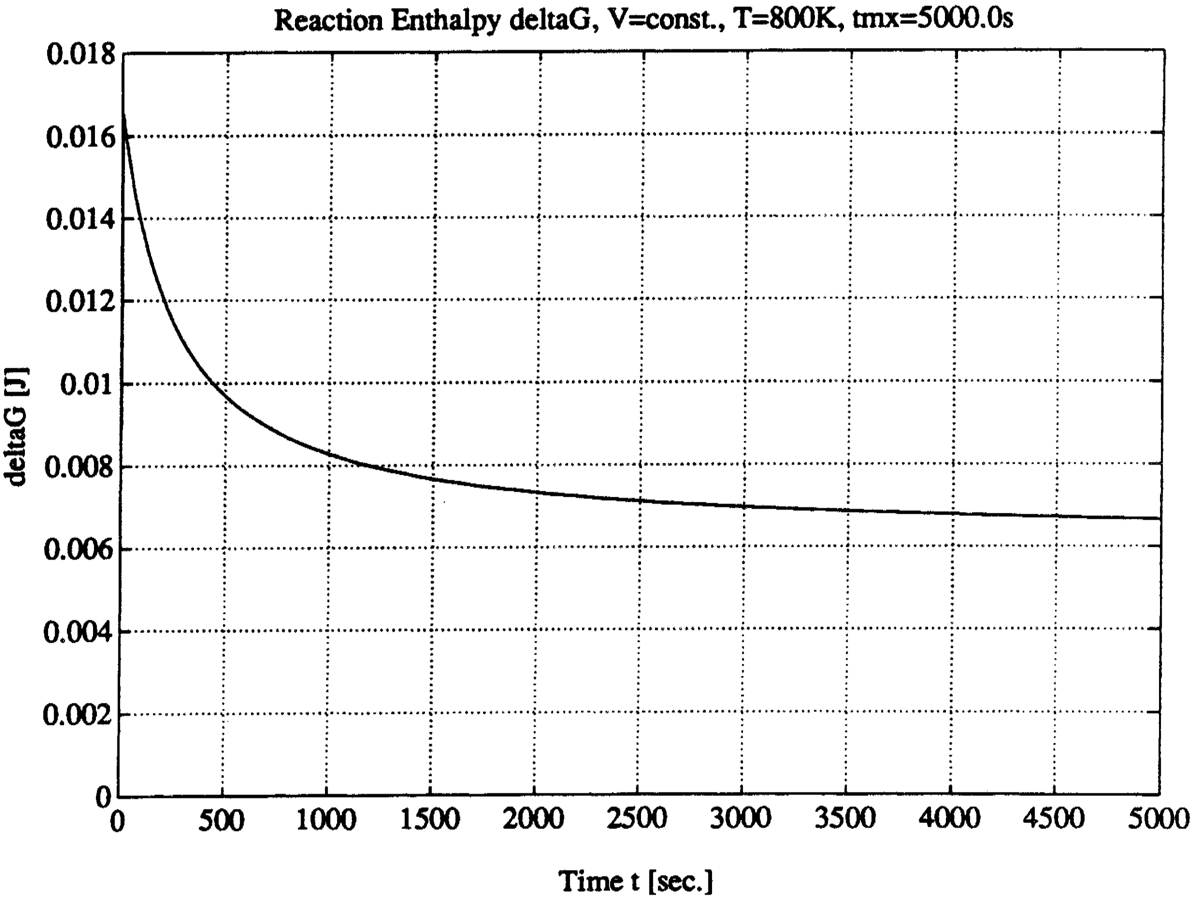





x10-7
Mole Fraction x_i, V=const., T=800K, tmx=5000.s
x_i [mole_i/mole_tot]
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
HBr
H·
Br2
H2
Br·
0
500
1000
1500
2000
2500
3000
3500
4000
4500
5000
Time t [sec.]


x10-4
Work W, V=const., T=800K, tmx=5000.0s
W [J]
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
0
500
1000
1500
2000
2500
3000
3500
4000
4500
5000
Time t [sec.]

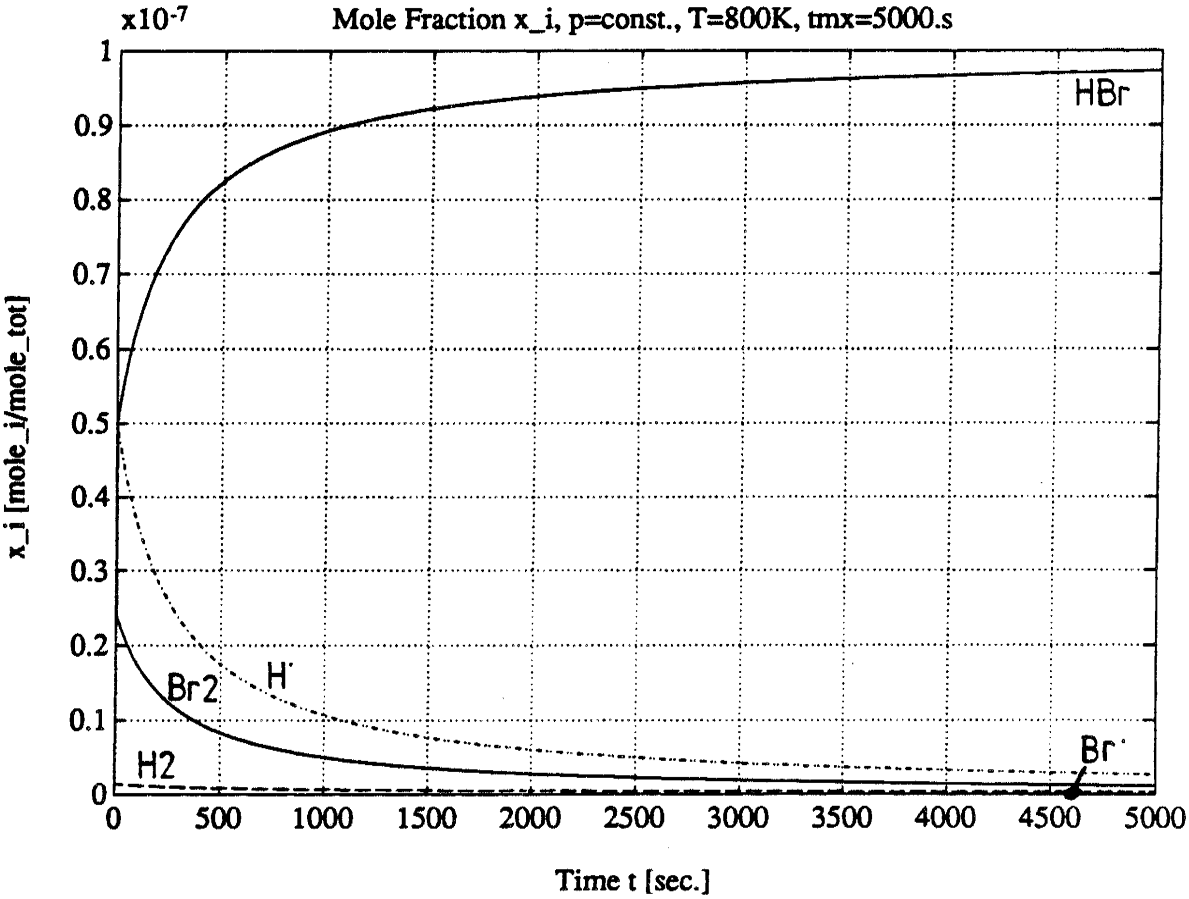

x10-3
Gibbs' Free Enthalpy G, V=const., T=800K, tmx=5000.0s
G [J]
4
2
0
-2
-4
-6
-8
-10
-12
-14
-16
0
500
1000
1500
2000
2500
3000
3500
4000
4500
5000
Time t [sec.]

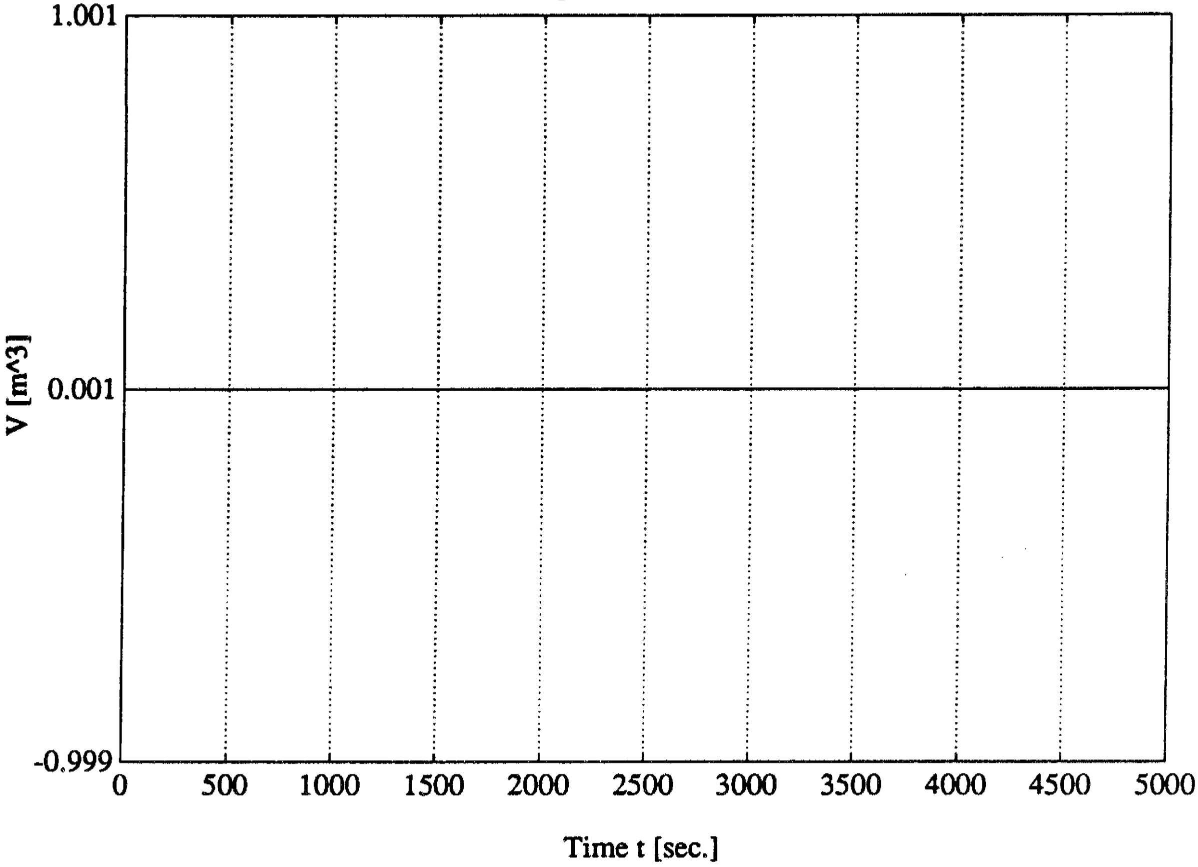

Reaction Enthalpy deltaG, V=const., T=800K, tmx=5000.0s
deltaG [J]
0.018
0.016
0.014
0.012
0.01
0.008
0.006
0.004
0.002
0
0
500
1000
1500
2000
2500
3000
3500
4000
4500
5000
Time t [sec.]

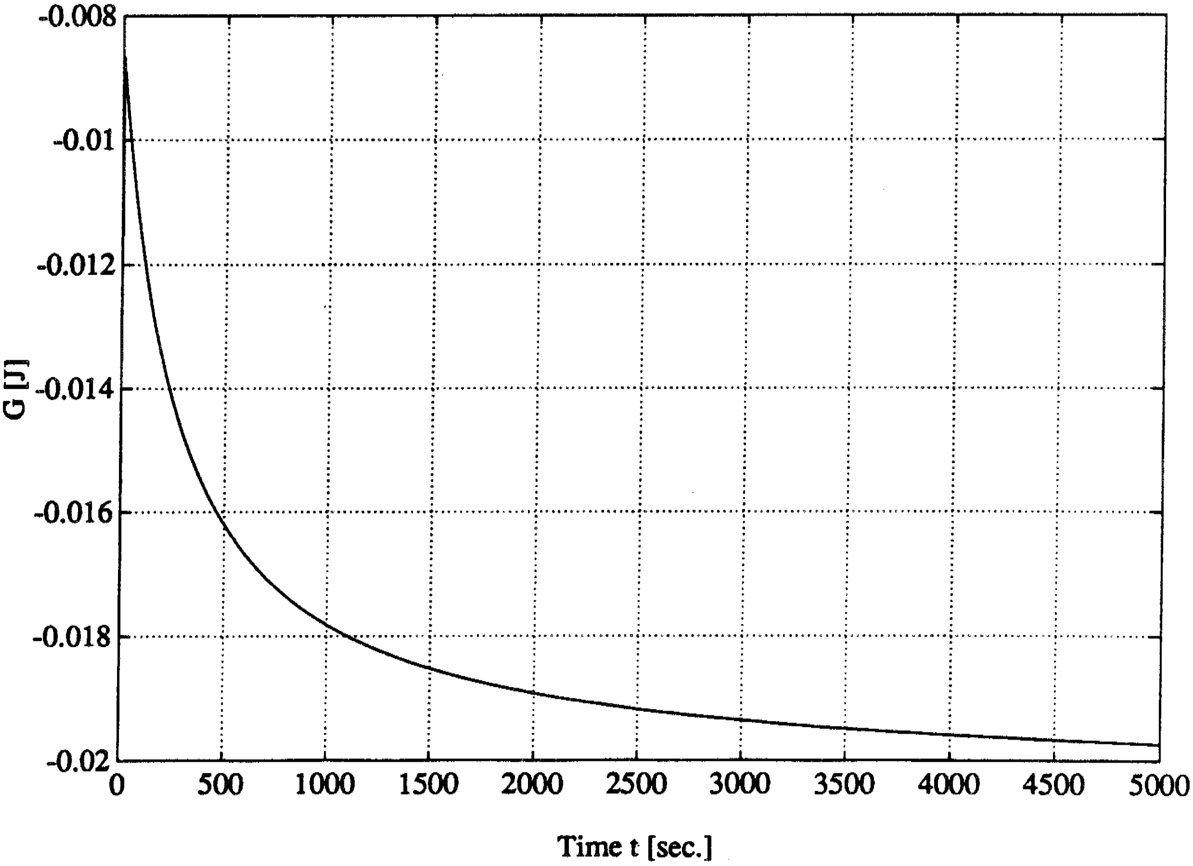
D.2 Isobaric–Isothermic Case

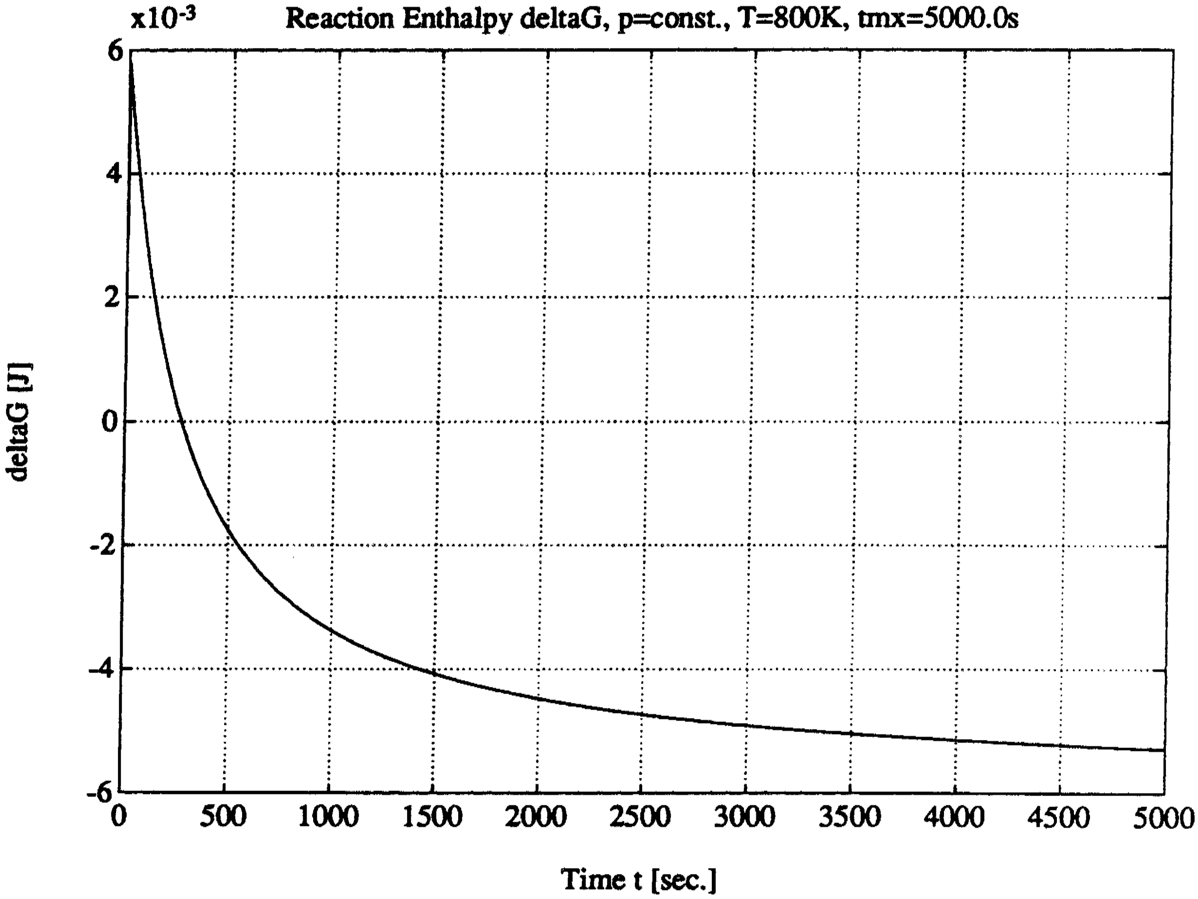
The graphs of isobaric–isothermic case are shown for $tmx = 5000.0sec.$

D.2.1 Results for a Simulation Time of $5000.0sec.$




Total Volume V, p=const., T=800K, tmx=5000.0s

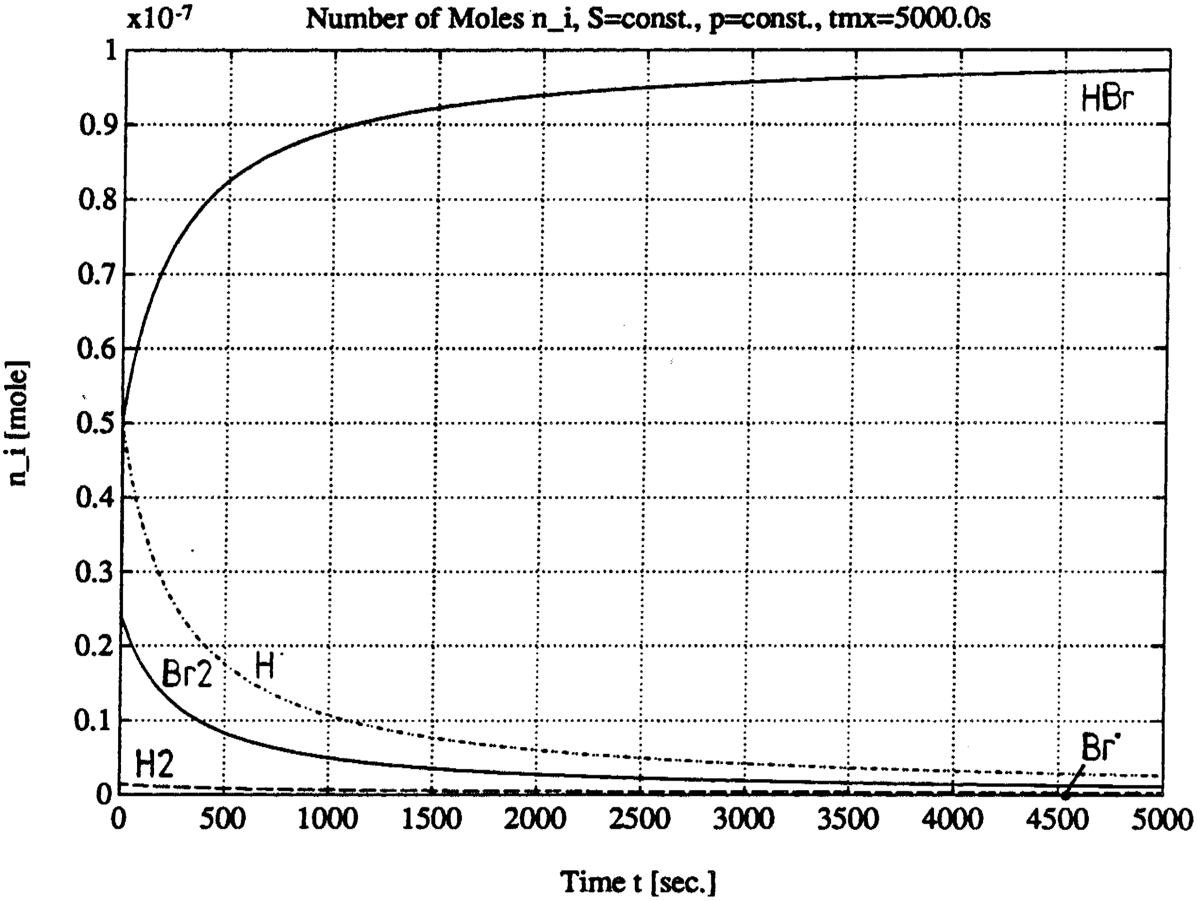

Gibbs' Free Enthalpy G, p=const., T=800K, tmx=5000.0s

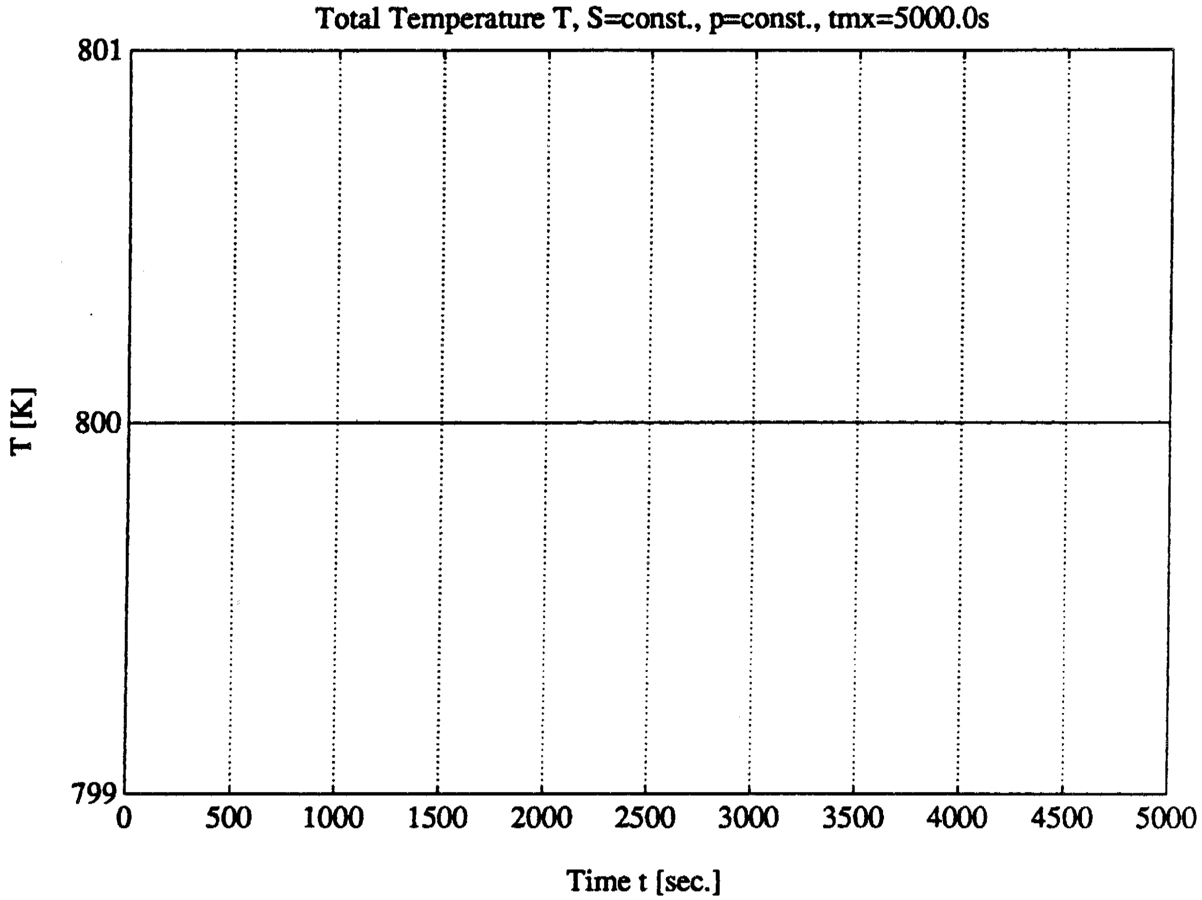

x10-3
Reaction Enthalpy deltaG, p=const., T=800K, tmx=5000.0s
deltaG [J]
6
4
2
0
-2
-4
-6
0
500
1000
1500
2000
2500
3000
3500
4000
4500
5000
Time t [sec.]

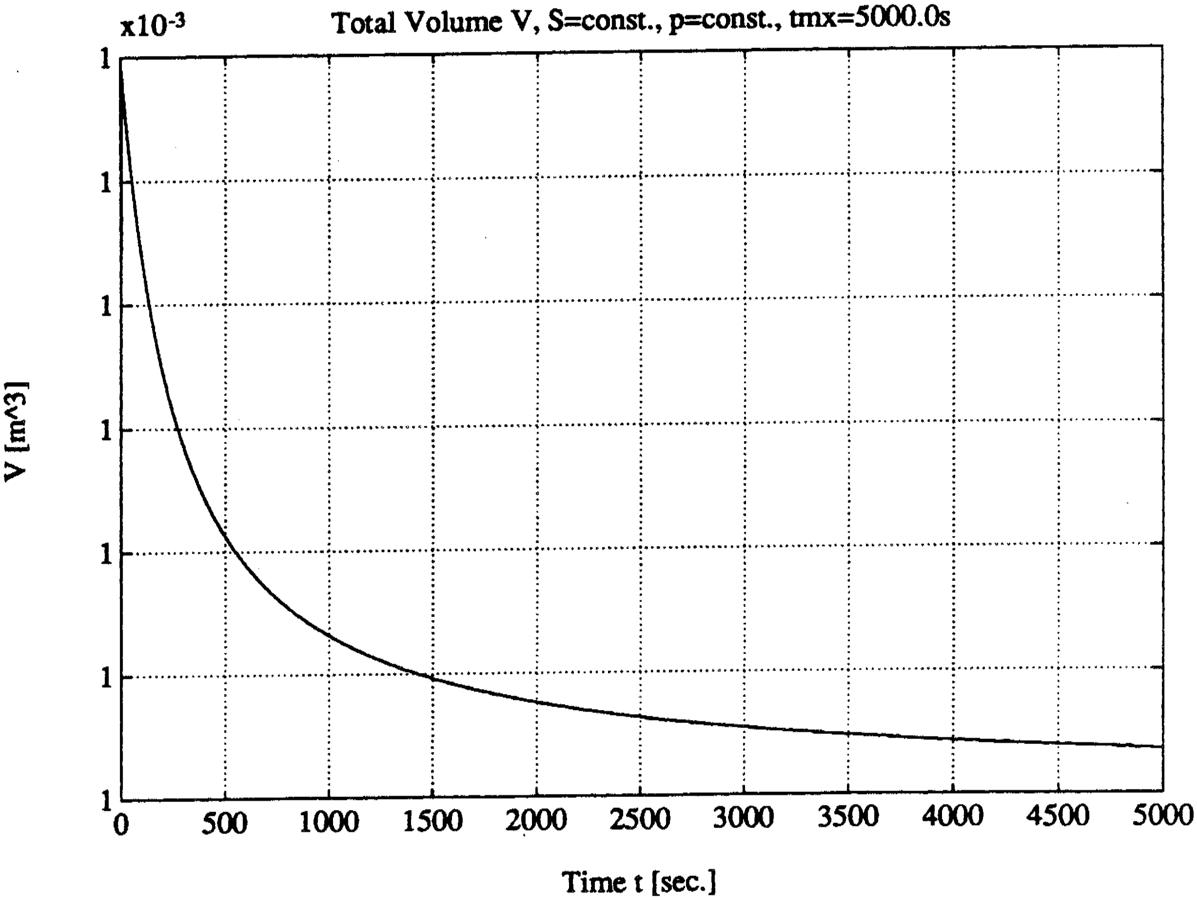
D.3 Isentropic–Isobaric Case

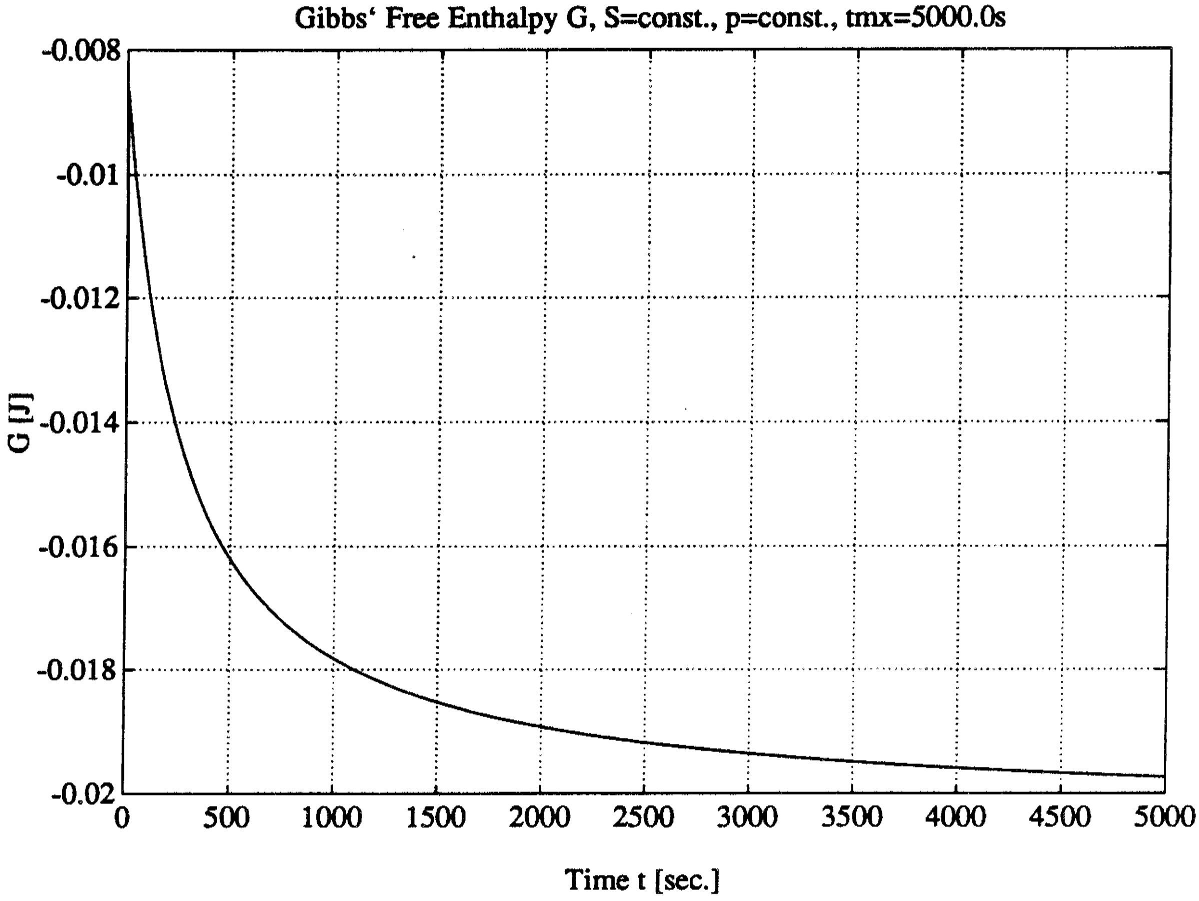
The graphs of isentropic–isobaric case are shown for $tmx = 5000.0 sec.$

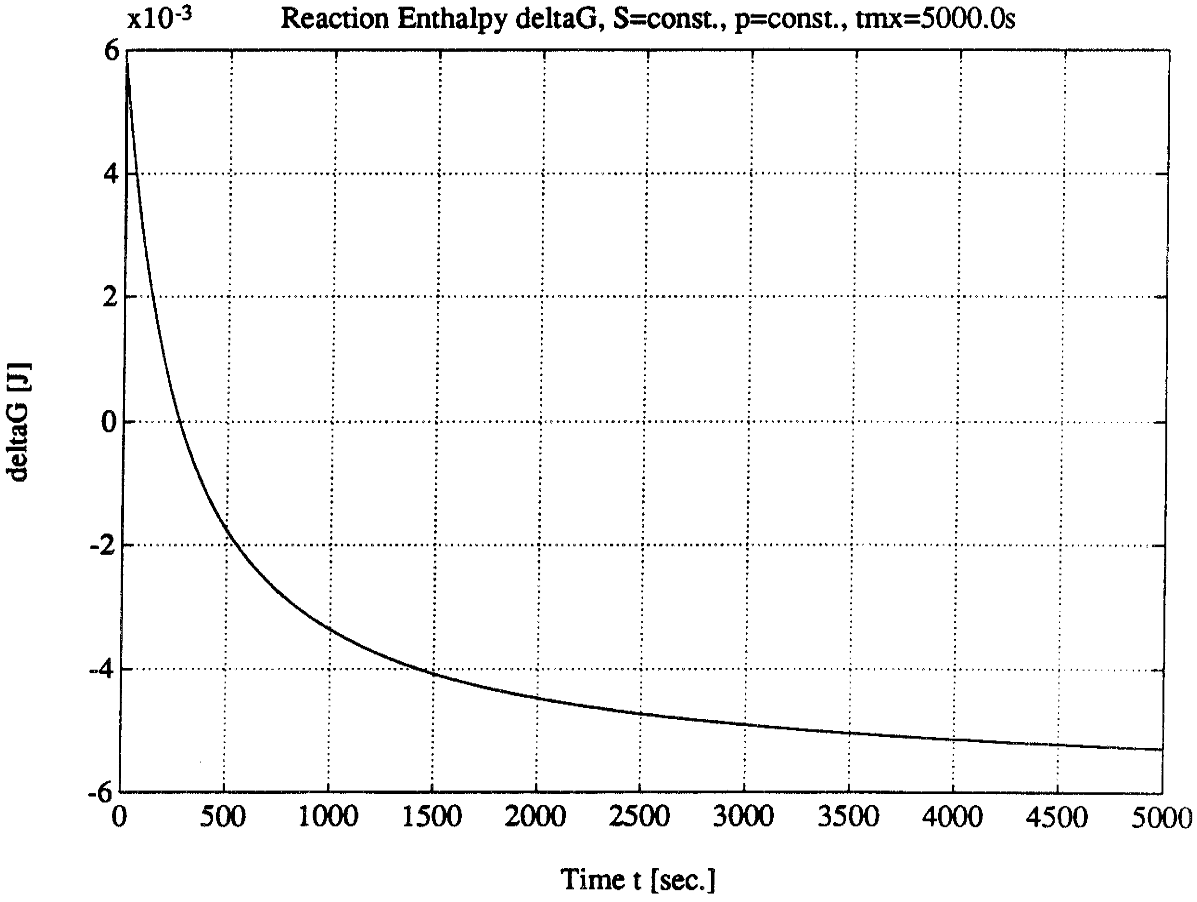
D.3.1 Results for a Simulation Time of $5000.0 sec.$




Total Temperature T, S=const., p=const., tmx=5000.0s
T [K]
801
800
799
0
500
1000
1500
2000
2500
3000
3500
4000
4500
5000
Time t [sec.]


x10-3
Total Volume V, S=const., p=const., tmx=5000.0s
V [m^3]
1
1
1
1
1
1
1
0
500
1000
1500
2000
2500
3000
3500
4000
4500
5000
Time t [sec.]


Gibbs' Free Enthalpy G, S=const., p=const., tmx=5000.0s


Reaction Enthalpy deltaG, S=const., p=const., tmx=5000.0s

Appendix E

The ACSL Programs of the Three Cases

This appendix contains the ACSL programs of the isochoric–isothermic, isobaric–isothermic, and isentropic–isobaric cases of the hydrogen–bromine reaction which were needed to obtain the graphs shown in Appendix E.

E.1 Isochoric–Isothermic Case

```
PROGRAM H2-Br2 Reaction [H9.1]
"Isochoric--isothermic"

  INITIAL

    CONSTANT   Temp1 = 800, V=1.0E-3
    CONSTANT   Br20 = 0.5E-7, H20 = 0.5E-7
    CONSTANT   R = 8.314, m = 10.0, S0 =  441.51
    CONSTANT   tmx = 5000.0, eps = 1.0E-15

    CONSTANT   p0 = 1.0E5, W0 = 0.0

    ERRTAG     mich
    CINTERVAL  cint = 1.0
    MINTERVAL  mint = 1.0E-14
    ALGORITHM  ialg = 2
    NSTEPS     nstp = 2001
    XERROR  nBr2 = 1.0E-14, nBr = 1.0E-13,  nH2 = 1.0E-15
    XERROR  nHBr = 1.0E-11, nH = 1.0E-13
    XERROR  mueBr2 = 1.0E4, mueBr = 1.0E4, mueH2 = 1.0E4
    XERROR  mueH = 1.0E4, mueHBr = 1.0E4
    XERROR  S = 1.0E-14, p = 1.0E2
  TABLE KBr2Br,1,19   ...
          /298.15,  300.0,  400.0,  500.0,  600.0,  700.0,   ...
           800.0,   900.0, 1000.0, 1100.0, 1200.0, 1300.0,   ...
          1400.0,  1500.0, 1600.0, 1700.0, 1800.0, 1900.0,   ...
          2000.0,                                            ...

          3.1623E-29, 7.7446E-29, 1.9543E-20, 2.2182E-15,    ...
          5.2844E-12, 1.3867E-9,  9.0782E-8,  2.3768E-6,     ...
          3.2509E-5,  2.7861E-4,  1.6788E-3,  7.6913E-3,     ...
          2.8510E-2,  8.8716E-2,  2.4044E-1,  5.8344E-1,     ...
          1.7947   ,  2.6061   ,  4.9431 /


  ak1 = 1.39E8 * Temp1**.5 * (189243/(R*Temp1))**1.97
  k1 = ak1     * EXP( -189243/(R*Temp1) )
  k2 = k1/KBr2Br( Temp1 )
  k3 = 10**11.43 * EXP( - 82400/(R*Temp1) )
  k5 = k3/( 10**(-0.54) * EXP( 67400/(R*Temp1)) )
  k4 = k5 / m



  mue0B2 = -204493.0
  mue0H2 = -53772.8
  mue0HB = -129023.0
  mue0Br =  106772.0
  mue0H  =  226828.0

  G0 = mue0B2*Br20 + mue0H2*H20

END $ "of INITIAL"

DYNAMIC

  DERIVATIVE

    xiek1 = k1*nBr2
    xiek2 = k2*nBr*nBr/V
    xiek3 = k3*nH2*nBr/V
    xiek4 = k4*nHBr*nH/V
    xiek5 = k5*nH*nBr2/V

    xieBr2 = -xiek1 + xiek2                    - xiek5
    xieBr  = 2*xiek1 - 2*xiek2 - xiek3 + xiek4 + xiek5
    xieH2  =                   - xiek3 + xiek4
    xieH   =                     xiek3 - xiek4 - xiek5
    xieHBr =                     xiek3 - xiek4 + xiek5

    nBr2 = INTEG( xieBr2, Br20 )
    nBr  = INTEG( xieBr , 0.0  )
    nH2  = INTEG( xieH2 , H20  )
    nHBr = INTEG( xieHBr, 0.0  )
    nH   = INTEG( xieH  , 0.0  )

    Sdotk1 = - muek1*xiek1/Temp1
    Sdotk2 = - muek2*xiek2/Temp1
    Sdotk3 = - muek3*xiek3/Temp1
    Sdotk4 = - muek4*xiek4/Temp1
    Sdotk5 = - muek5*xiek5/Temp1
    mudBr2 = xieBr2*R*Temp1/( nBr2 + eps )
    mudBr  = xieBr*R*Temp1/( nBr + eps )
    mudH2  = xieH2*R*Temp1/( nH2 + eps )
    mudH   = xieH*R*Temp1/( nH + eps )
    mudHBr = xieHBr*R*Temp1/( nHBr + eps )

    mueBr2 = INTEG( mudBr2, mue0B2 )
    mueBr  = INTEG( mudBr, mue0Br )
    mueH2  = INTEG( mudH2, mue0H2 )
    mueH   = INTEG( mudH, mue0H )
    mueHBr = INTEG( mudHBr, mue0HB )

    muek1 = -mueBr2 + 2*mueBr
    muek2 = mueBr2 - 2*mueBr
    muek3 = -mueH2 + mueHBr + mueH - mueBr
    muek4 = -mueH - mueHBr + mueBr + mueH2
    muek5 = mueBr + mueHBr - mueBr2 - mueH

    pdoBr2 = mudBr2*nBr2/V
    pdoBr  = mudBr*nBr/V
    pdoH2  = mudH2*nH2/V
    pdoH   = mudH*nH/V
    pdoHBr = mudHBr*nHBr/V

    Sdot = Sdotk1 + Sdotk2 + Sdotk3 + Sdotk4 + Sdotk5
    pdot = pdoBr2 + pdoBr + pdoH2 + pdoH + pdoHBr

    S = INTEG( Sdot, S0 )
    p = INTEG( pdot, p0 )
    W = INTEG( pdot*V, W0)

END $ "of DERIVATIVE"

  aBr = 2*nBr2 + nBr + nHBr
  aH  = 2*nH2 + nH + nHBr

  ntot = nBr2 + nBr + nH2 + nH + nHBr
  xBr2 = nBr2/ntot
  xBr  = nBr/ntot
  xH2  = nH2/ntot
  xH   = nH/ntot
  xHBr = nHBr/ntot
```

```
    G      = mueBr2*nBr2 + mueBr*nBr + mueH2*nH2 + mueH*nH ...
             + mueHBr*nHBr
    U      = Templ*S - p*V + G
    deltaG = G - G0

    GmW = G - W
    GpW = G + W

    Err = p*V - ntot*R*Templ

    TERMT( t.GE.tmx )

  END $ "of DYNAMIC"

END $ "of PROGRAM"
```

E.2 Isobaric–Isothermic Case

```
PROGRAM H2-Br2 Reaction [H9.1]
"Isobaric--isothermic"

  INITIAL

    CONSTANT   Temp1 = 800, p=1E5
    CONSTANT   Br20 = 0.5E-7, H20 = 0.5E-7
    CONSTANT   R = 8.314, m = 10.0, S0 = 441.57
    CONSTANT   tmx = 5000.0
    CONSTANT   Phik3 = 0.0, Phik4 = 0.0, Phik5 = 0.0

    CONSTANT   V0 = 1.0E-3

    ERRTAG     mich
    CINTERVAL  cint = 1.0                 ( Integration length
    MINTERVAL  mint = 1.0E-9
    ALGORITHM  ialg = 2
    NSTEPS     nstp = 2001
    XERROR  nBr2 = 1.0E-14, nBr = 1.0E-13,  nH2 = 1.0E-15
    XERROR  nHBr = 1.0E-11, nH = 1.0E-13
    XERROR  V = 1.0E-15, S = 1.0E-15

    TABLE KBr2Br,1,19    ...
            /298.15,  300.0,  400.0,  500.0,  600.0,  700.0,   ...
             800.0,   900.0, 1000.0, 1100.0, 1200.0, 1300.0,   ...
            1400.0,  1500.0, 1600.0, 1700.0, 1800.0, 1900.0,   ...
            2000.0,                                            ...

            3.1623E-29, 7.7446E-29, 1.9543E-20, 2.2182E-15,    ...
            5.2844E-12, 1.3867E-9,  9.0782E-8,  2.3768E-6,     ...
            3.2509E-5,  2.7861E-4,  1.6788E-3,  7.6913E-3,     ...
            2.8510E-2,  8.8716E-2,  2.4044E-1,  5.8344E-1,     ...
            1.7947    , 2.6061    ,  4.9431 /

   ak1 = 1.39E8 * Temp1**.5 * (189243/(R*Temp1))**1.97
   k1 = ak1     * EXP( -189243/(R*Temp1) )
   k2 = k1/KBr2Br( Temp1 )
   k3 = 10**11.43 * EXP( - 82400/(R*Temp1) )
   k5 = k3/( 10**(-0.54) * EXP( 67400/(R*Temp1)) )
   k4 = k5 / m

   mue0B2 = -204493.0
   mue0H2 = -53772.8
   mue0HB = -129023.0
   mue0Br =  106772.0
   mue0H  = -226828.0
   muek1 = -mue0B2 + 2*mue0Br
   muek2 = mue0B2 - 2*mue0Br
   muek3 = -mue0H2 + mue0HB + mue0H - mue0Br
   muek4 = -mue0H - mue0HB + mue0Br + mue0H2
   muek5 = mue0Br + mue0HB - mue0B2 - mue0H


   G0 = mue0B2*Br20 + mue0H2*H20

 END $ "of INITIAL"

 DYNAMIC

   DERIVATIVE


     xiek1 = k1*nBr2
     xiek2 = k2*nBr*nBr/V
     xiek3 = k3*nH2*nBr/V
     xiek4 = k4*nHBr*nH/V
     xiek5 = k5*nH*nBr2/V

     xieBr2 = -xiek1 + xiek2                    - xiek5
     xieBr  = 2*xiek1 - 2*xiek2 - xiek3 + xiek4 + xiek5
     xieH2  =                   - xiek3 + xiek4
     xieH   =                     xiek3 - xiek4 - xiek5
     xieHBr =                     xiek3 - xiek4 + xiek5

     nBr2 = INTEG( xieBr2, Br20 )
     nBr  = INTEG( xieBr , 0.0  )
     nH2  = INTEG( xieH2 , H20  )
     nHBr = INTEG( xieHBr, 0.0  )
     nH   = INTEG( xieH  , 0.0  )

     Phik1 = xiek1*R*Temp1/p
     Phik2 = -xiek2*R*Temp1/p

     Sdotk1 = ( p*Phik1 - muek1*xiek1 )/Temp1
     Sdotk2 = ( p*Phik2 - muek2*xiek2 )/Temp1
     Sdotk3 = ( p*Phik3 - muek3*xiek3 )/Temp1
     Sdotk4 = ( p*Phik4 - muek4*xiek4 )/Temp1
     Sdotk5 = ( p*Phik5 - muek5*xiek5 )/Temp1

     Phi = Phik1 + Phik2 + Phik3 + Phik4 + Phik5
     Sdot = Sdotk1 + Sdotk2 + Sdotk3 + Sdotk4 + Sdotk5

     V = INTEG( Phi, V0 )
     S = INTEG( Sdot, S0 )

END $ "of DERIVATIVE"
```

```
    aBr = 2*nBr2 + nBr + nHBr
    aH  = 2*nH2 + nH + nHBr

    ntot = nBr2 + nBr + nH2 + nH + nHBr
    xBr2 = nBr2/ntot
    xBr  = nBr/ntot
    xH2  = nH2/ntot
    xH   = nH/ntot
    xHBr = nHBr/ntot

    G      = mue0B2*nBr2 + mue0Br*nBr + mue0H2*nH2 + mue0H*nH ...
               + mue0HB*nHBr
    U      = Temp1*S - p*V + G
    deltaG = G - G0

    Err = p*V - ntot*R*Temp1

    TERMT( t.GE.tmx )

  END $ "of DYNAMIC"

END $ "of PROGRAM"
```

E.3 Isentropic–Isobaric Case

```
PROGRAM H2-Br2 Reaction [H9.1]
"Isentropic--isobaric"

  INITIAL

    CONSTANT   Temp0 = 298.15, p = 1.0E5, S=0.0
    CONSTANT   Br20 = 0.5E-7, H20 = 0.5E-7
    CONSTANT   R = 8.314, m = 10.0
    CONSTANT   tmx = 3.3, eps = 1.0E-15

    CONSTANT  V0=1.0E-3, Temp1 = 800, W0= 0.0

    ERRTAG     mich
    CINTERVAL  cint = 0.005
    MINTERVAL  mint = 1.0E-14
    ALGORITHM  ialg = 2
    NSTEPS     nstp = 2001
    XERROR  nBr2 = 1.0E-14, nBr = 1.0E-13,  nH2 = 1.0E-15
    XERROR  nHBr = 1.0E-11, nH = 1.0E-13
    XERROR  mueBr2 = 1.0E4, mueBr = 1.0E4, mueH2 = 1.0E4
    XERROR  mueH = 1.0E4, mueHBr = 1.0E4
    XERROR  Temp = 1.0E-1, V = 1.0E-5
    XERROR  W = 1.0E2

    TABLE SiBr2t,1,19     ...
            /298.15,  300.0,  400.0,  500.0,  600.0,  700.0,    ...
             800.0,   900.0, 1000.0, 1100.0, 1200.0, 1300.0,    ...
            1400.0,  1500.0, 1600.0, 1700.0, 1800.0, 1900.0,    ...
            2000.0,                                             ...

             245.510, 245.731, 256.220, 264.463, 271.246, 277.007, ...
             282.014, 286.444, 290.413, 294.014, 297.300, 300.336, ...
             303.145, 305.766, 308.224, 310.535, 312.716, 314.780, ...
             316.744 /

    TABLE HiBr2t,1,19     ...
            /298.15,  300.0,  400.0,  500.0,  600.0,  700.0,    ...
             800.0,   900.0, 1000.0, 1100.0, 1200.0, 1300.0,    ...
            1400.0,  1500.0, 1600.0, 1700.0, 1800.0, 1900.0,    ...
            2000.0,                                             ...

             30927.892, 30994.880, 34641.583, 38334.341, 42056.406, ...
             45795.218, 49546.591, 53306.338, 57074.458, 60846.764, ...
             64631.632, 68420.686, 72213.926, 76011.354, 79817.155, ...
             83627.143, 87441.318, 91263.866, 95090.602 /

    TABLE SiBrt,1,19      ...
            /298.15,  300.0,  400.0,  500.0,  600.0,  700.0,    ...
             800.0,   900.0, 1000.0, 1100.0, 1200.0, 1300.0,    ...
            1400.0,  1500.0, 1600.0, 1700.0, 1800.0, 1900.0,    ...
            2000.0,                                             ...
```

```
          175.017, 175.146, 181.117, 185.752, 189.553, 192.785, ...
          195.599, 198.098, 200.347, 202.394, 204.278, 206.020, ...
          207.644, 209.168, 210.600, 211.957, 213.242, 214.469, ...
          215.637 /

 TABLE HiBrt,1,19      ...
         /298.15,  300.0,  400.0,  500.0,  600.0,  700.0,   ...
          800.0,   900.0, 1000.0, 1100.0, 1200.0, 1300.0,   ...
         1400.0,  1500.0, 1600.0, 1700.0, 1800.0, 1900.0,   ...
         2000.0,                                            ...

          111913.164, 111950.845, 114027.498, 116104.151,  ...
          118189.177, 120286.764, 122392.724, 124515.432,  ...
          126650.700, 128798.528, 130963.104, 133140.240,  ...
          135334.123, 137540.567, 139759.571, 141995.322,  ...
          144247.820, 146512.879, 148794.685 /

 TABLE KBr2Br,1,19     ...
         /298.15,  300.0,  400.0,  500.0,  600.0,  700.0,   ...
          800.0,   900.0, 1000.0, 1100.0, 1200.0, 1300.0,   ...
         1400.0,  1500.0, 1600.0, 1700.0, 1800.0, 1900.0,   ...
         2000.0,                                            ...

         3.1623E-29, 7.7446E-29, 1.9543E-20, 2.2182E-15,    ...
         5.2844E-12, 1.3867E-9,  9.0782E-8,  2.3768E-6,     ...
         3.2509E-5,  2.7861E-4,  1.6788E-3,  7.6913E-3,     ...
         2.8510E-2,  8.8716E-2,  2.4044E-1,  5.8344E-1,     ...
         1.7947    ,  2.6061   ,  4.9431 /

 TABLE SiH2t,1,19      ...
         /298.15,  300.0,  400.0,  500.0,  600.0,  700.0,   ...
          800.0,   900.0, 1000.0, 1100.0, 1200.0, 1300.0,   ...
         1400.0,  1500.0, 1600.0, 1700.0, 1800.0, 1900.0,   ...
         2000.0,                                            ...

         130.670, 130.850, 139.148, 145.625, 150.959, 155.510, ...
         159.496, 163.046, 166.254, 169.189, 171.893, 174.405, ...
         176.758, 178.969, 181.058, 183.030, 184.931, 186.731, ...
         188.460 /

 TABLE HiH2t,1,19      ...
         /298.15,  300.0,  400.0,  500.0,  600.0,  700.0,   ...
          800.0,   900.0, 1000.0, 1100.0, 1200.0, 1300.0,   ...
         1400.0,  1500.0, 1600.0, 1700.0, 1800.0, 1900.0,   ...
         2000.0,                                            ...

         0.0,        54.428,     2939.134,   5840.586,   8767.159, ...
         11723.04,   14704.052,  17718.538,  20766.528,  23843.826, ...
         26950.432,  30094.718,  33264.126,  36471.215,  39711.798, ...
         42981.689,  46285.074,  49617.767,  52988.141 /

 TABLE SiHt,1,19       ...
         /298.15,  300.0,  400.0,  500.0,  600.0,  700.0,   ...
          800.0,   900.0, 1000.0, 1100.0, 1200.0, 1300.0,   ...
         1400.0,  1500.0, 1600.0, 1700.0, 1800.0, 1900.0,   ...
         2000.0,                                            ...
```

```
        114.681, 114.810, 120.793, 125.4365, 129.226, 132.433, ...
        135.2085, 137.662, 139.852, 141.836, 143.645, 145.307, ...
        146.852, 148.284, 149.628, 150.888, 152.077, 153.203,  ...
        154.271 /

    TABLE HiHt,1,19     ...
        /298.15,  300.0,  400.0,  500.0,  600.0,  700.0,   ...
         800.0,   900.0, 1000.0, 1100.0, 1200.0, 1300.0,   ...
        1400.0,  1500.0, 1600.0, 1700.0, 1800.0, 1900.0,   ...
        2000.0,                                            ...

        218149.027, 218186.208, 220267.548, 222348.388,    ...
        224429.227, 226505.880, 228586.720, 230667.559,    ...
        232748.399, 234829.238, 236905.891, 238986.731,    ...
        241067.570, 243148.410, 245229.250, 247305.902,    ...
        249386.742, 251467.582, 253548.421 /

    TABLE SiHBrt,1,15     ...
        /298.15,  300.0,  400.0,  500.0,  600.0,  700.0,   ...
         800.0,   900.0, 1000.0, 1100.0, 1200.0, 1300.0,   ...
        1400.0,  1500.0, 1600.0,                           ...

        198.718, 198.898, 207.276, 213.824, 219.250, 223.910, ...
        228.013, 231.702, 235.055, 238.145, 241.017, 243.701, ...
        246.234, 248.629, 250.9065 /

    TABLE HiHBrt,1,15     ...
        /298.15,  300.0,  400.0,  500.0,  600.0,  700.0,   ...
         800.0,   900.0, 1000.0, 1100.0, 1200.0, 1300.0,   ...
        1400.0,  1500.0, 1600.0,                           ...

        -36404.226, -36349.798, -33435.785, -30500.838, ...
        -27528.210, -24505.340, -21428.042, -18296.316, ...
        -15110.161, -11869.578,  -8570.380,  -5212.566, ...
        -1800.324, 1670.533, 5200.006 /


HiBr2 =  RSW((Templ.GT.2000.0.OR.Templ.LT.298.15),HiBr2t(298.15)     ...
    + 37.3839*( Templ - Temp0 ) + 0.2324E-3*( Templ*Templ - Temp0* ...
    Temp0 ) + 1.2937E5*( 1/Templ - 1/Temp0 ) , HiBr2t( Templ ) )
HiH2 =  RSW((Templ.GT.2000.0.OR.Templ.LT.298.15),HiH2t(298.15)       ...
    + 27.2979*( Templ - Temp0 ) + 1.63285E-3*( Templ*Templ - Temp0* ...
    Temp0 ) - 0.5024E5*( 1/Templ - 1/Temp0 ) , HiH2t( Templ ) )
HiHBr =  RSW((Templ.GT.1600.0.OR.Templ.LT.298.15),HiHBrt(298.15)     ...
    + 26.1675*( Templ - Temp0 ) + 2.9208E-3*( Templ*Templ - Temp0* ...
    Temp0 ) - 1.0886E5*( 1/Templ - 1/Temp0 ) , HiHBrt( Templ ) )
HiBr =  RSW((Templ.GT.2000.0.OR.Templ.LT.298.15),HiBrt(298.15)       ...
    + 19.8873*( Templ - Temp0 ) + 0.74525E-3*( Templ*Templ - Temp0* ...
    Temp0 ) - 0.4229E5*( 1/Templ - 1/Temp0 ) , HiBrt( Templ ) )
HiH =  RSW((Templ.GT.2000.0.OR.Templ.LT.298.15),HiHt(298.15)     ...
    + 20.8000*( Templ - Temp0) , HiHt( Templ ) )
```

```
SiBr2 =  RSW((Temp1.GT.2000.OR.Temp1.LT.298.15),SiBr2t(298.15)    ...
    + 3.3839*ALOG( Temp1/Temp0 ) + 2*0.2324E-3*( Temp1 - Temp0) ...
    + 1.2937E5/2*( 1/Temp1**2 - 1/Temp0**2 ) , SiBr2t( Temp1 ) )
SiH2 =  RSW((Temp1.GT.2000.OR.Temp1.LT.298.15),SiH2t(298.15)    ...
    + 27.2979*ALOG( Temp1/Temp0 ) + 2*1.63285E-3*( Temp1 - Temp0) ...
    - 0.5024E5/2*( 1/Temp1**2 - 1/Temp0**2 ) , SiH2t( Temp1 ) )
SiHBr =  RSW((Temp1.GT.1600.OR.Temp1.LT.298.15),SiHBrt(298.15)    ...
    + 26.1675*ALOG( Temp1/Temp0 ) + 2*2.92085E-3*( Temp1 - Temp0) ...
    - 1.0886E5/2*( 1/Temp1**2 - 1/Temp0**2 ) , SiHBrt( Temp1 ) )
SiBr =  RSW((Temp1.GT.2000.OR.Temp1.LT.298.15),SiBrt(298.15)    ...
    + 19.8873*ALOG( Temp1/Temp0 ) + 2*0.74525E-3*( Temp1 - Temp0) ...
    - 0.4229E5/2*( 1/Temp1**2 - 1/Temp0**2 ) , SiBrt( Temp1 ) )
SiH =  RSW((Temp1.GT.2000.OR.Temp1.LT.298.15),SiHt(298.15)    ...
    + 20.8000*ALOG( Temp1/Temp0 ) , SiHt( Temp1 ) )


    mue0B2 = HiBr2 - Temp1*SiBr2
    mue0H2  = HiH2  - Temp1*SiH2
    mue0HB = HiHBr - Temp1*SiHBr
    mue0Br  = HiBr  - Temp1*SiBr
    mue0H   = HiH   - Temp1*SiH

    G0 = mue0B2*Br20 + mue0H2*H20

   END $ "of INITIAL"

   DYNAMIC

    DERIVATIVE

         ak1 = 1.39E8 * Temp**.5 * (189243/(R*Temp))**1.97
         k1 = ak1        * EXP( -189243/(R*Temp) )
         k2 = k1/KBr2Br( Temp )
         k3 = 10**11.43 * EXP( - 82400/(R*Temp) )
         k5 = k3/( 10**(-0.54) * EXP( 67400/(R*Temp)) )
         k4 = k5 / m

         xiek1 = k1*nBr2
         xiek2 = k2*nBr*nBr/V
         xiek3 = k3*nH2*nBr/V
         xiek4 = k4*nHBr*nH/V
         xiek5 = k5*nH*nBr2/V

         xieBr2 = -xiek1 + xiek2                    - xiek5
         xieBr  = 2*xiek1 - 2*xiek2 - xiek3 + xiek4 + xiek5
         xieH2  =                   - xiek3 + xiek4
         xieH   =                     xiek3 - xiek4 - xiek5
         xieHBr =                     xiek3 - xiek4 + xiek5

         nBr2 = INTEG( xieBr2, Br20 )
         nBr  = INTEG( xieBr , 0.0  )
         nH2  = INTEG( xieH2 , H20  )
         nHBr = INTEG( xieHBr, 0.0  )
         nH   = INTEG( xieH  , 0.0  )
```

```
ntot = nBr2 +  nBr +  nH2 +  nH +  nHBr
xitot = xieBr2 +  xieBr +  xieH2 +  xieH +  xieHBr

Phik1 =  muek1*xiek1/p
Phik2 =  muek2*xiek2/p
Phik3 =  muek3*xiek3/p
Phik4 =  muek4*xiek4/p
Phik5 =  muek5*xiek5/p

TBr2 = Temp*nBr2/ntot
TBr  = Temp*nBr/ntot
TH2  = Temp*nH2/ntot
TH   = Temp*nH/ntot
THBr = Temp*nHBr/ntot

mudBr2 = (xitot*R*TBr2*S - mueBr2*xieBr2*S)/ ...
      (( nBr2 + eps )*R*ntot)
mudBr  = (xitot*R*TBr*S - mueBr*xieBr*S)/ ...
      (( nBr + eps )*R*ntot)
mudH2  = (xitot*R*TH2*S - mueH2*xieH2*S)/ ...
      (( nH2 + eps )*R*ntot)
mudH   = (xitot*R*TH*S - mueH*xieH*S)/...
      (( nH + eps )*R*ntot)
mudHBr = (xitot*R*THBr*S - mueHBr*xieHBr*S)/...
      (( nHBr + eps )*R*ntot)


mueBr2 = INTEG( mudBr2, mue0B2 )
mueBr  = INTEG( mudBr, mue0Br )
mueH2  = INTEG( mudH2, mue0H2 )
mueH   = INTEG( mudH, mue0H )
mueHBr = INTEG( mudHBr, mue0HB )

muek1 = -mueBr2 + 2*mueBr
muek2 = mueBr2 - 2*mueBr
muek3 = -mueH2 + mueHBr + mueH - mueBr
muek4 = -mueH - mueHBr + mueBr + mueH2
muek5 = mueBr + mueHBr - mueBr2 - mueH

TdoBr2 = -mudBr2*nBr2/(S + eps)
TdoBr  = -mudBr*nBr/(S + eps)
TdoH2  = -mudH2*nH2/(S + eps)
TdoH   = -mudH*nH/(S + eps)
TdoHBr = -mudHBr*nHBr/(S + eps)

Phi    = Phik1 + Phik2 + Phik3 + Phik4 + Phik5
```

```
          Tdot = TdoBr2 + TdoBr + TdoH2 + TdoH + TdoHBr

          V    = INTEG( Phi, V0 )
          Temp = INTEG( Tdot, Temp1 )
          W    = INTEG( Phi*p, W0)

  END $ "of DERIVATIVE"

    aBr = 2*nBr2 + nBr + nHBr
    aH  = 2*nH2 + nH + nHBr

    xBr2 = nBr2/ntot
    xBr  = nBr/ntot
    xH2  = nH2/ntot
    xH   = nH/ntot
    xHBr = nHBr/ntot

    G = mueBr2*nBr2 + mueBr*nBr + mueH2*nH2 + mueH*nH + mueHBr*nHBr
    U = Temp*S - p*V + G
    deltaG = G - G0

    Err = p*V - ntot*R*Temp

    TERMT( t.GE.tmx )

  END $ "of DYNAMIC"

END $ "of PROGRAM"
```

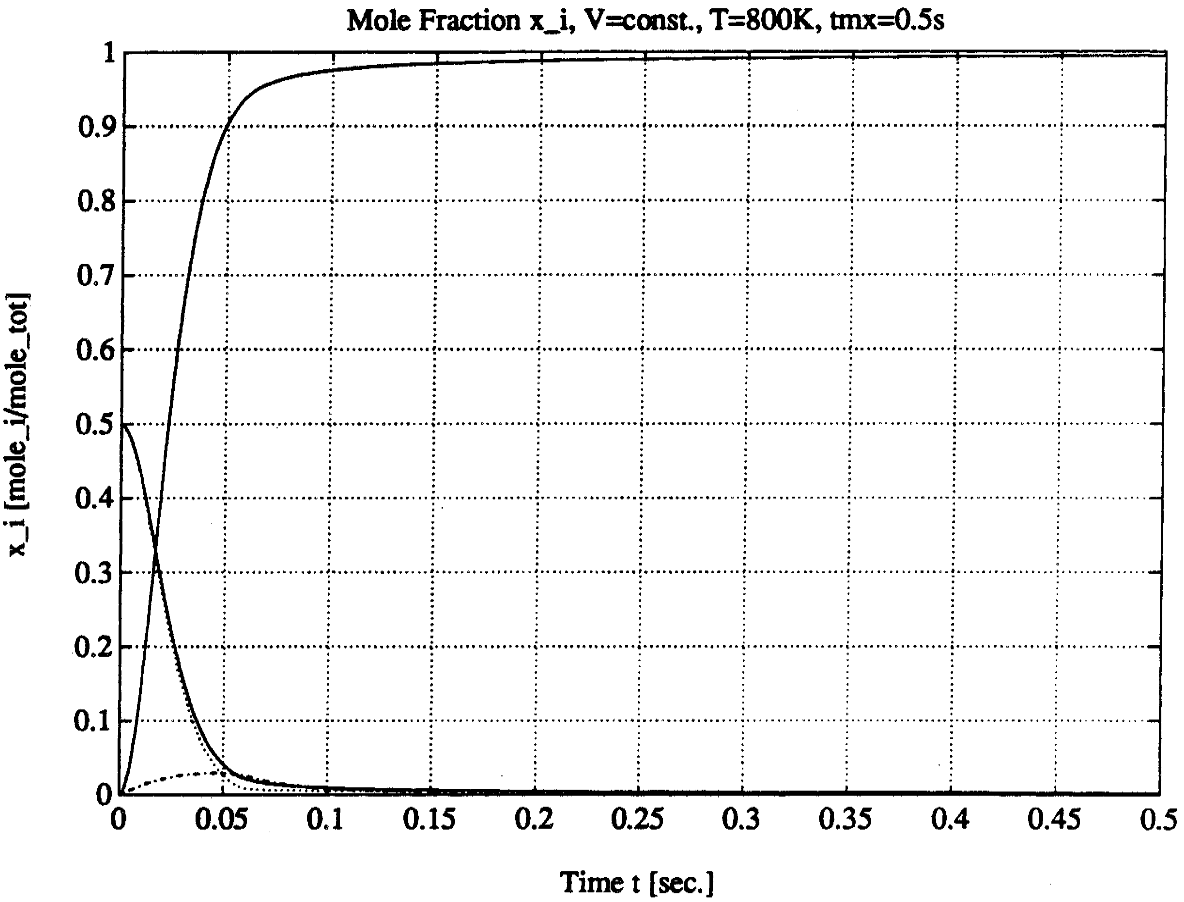
Appendix F

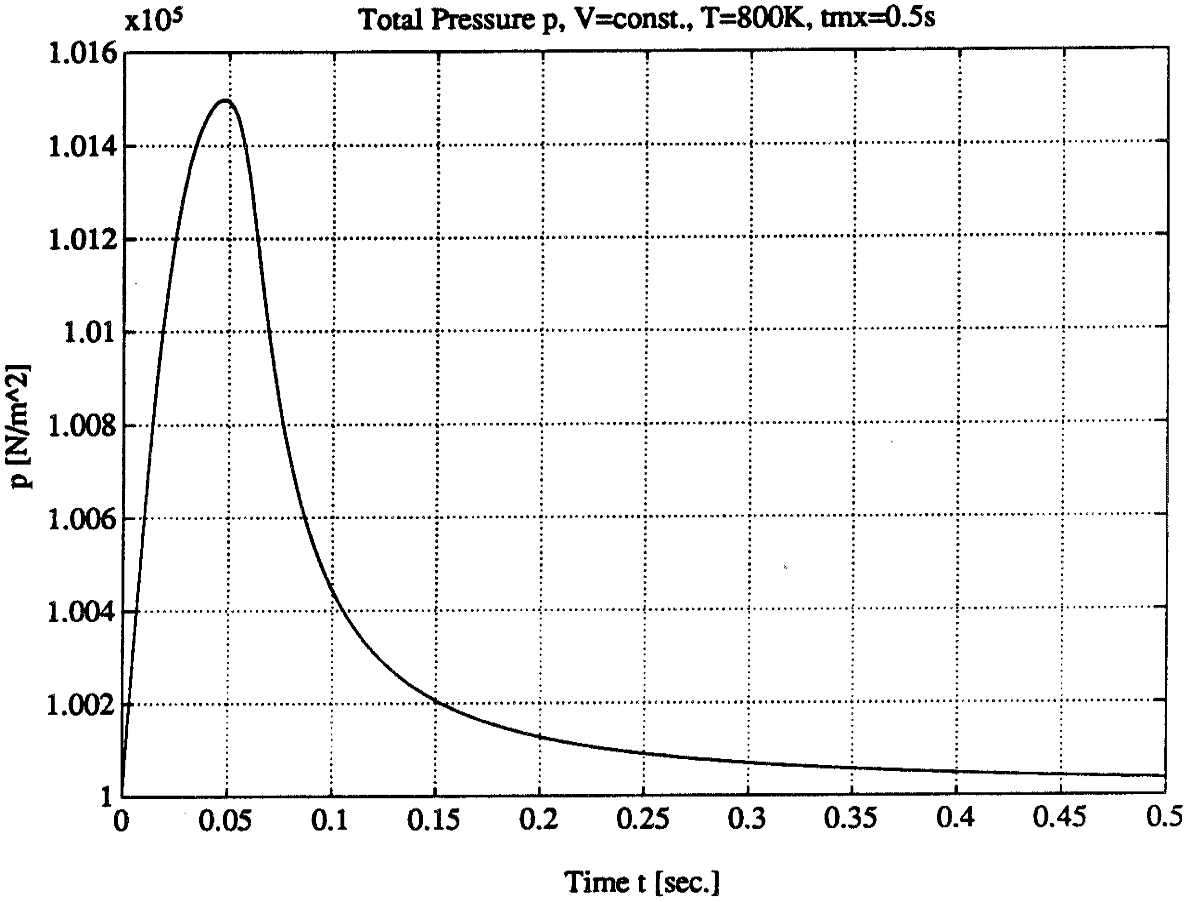
The Graphs from Different Simulation Runs

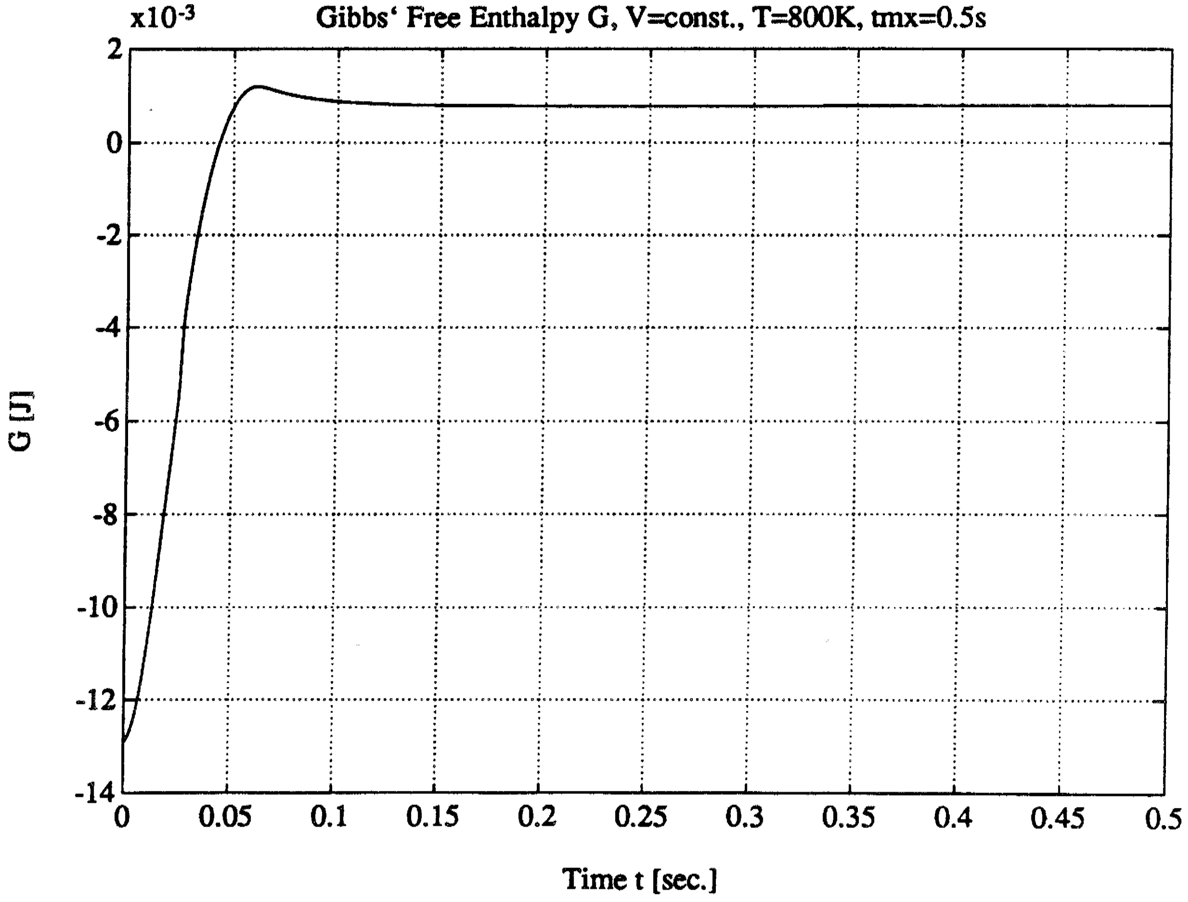
This appendix contains the graphs of the isochoric–isothermic and isobaric–isothermic cases of the hydrogen–bromine reaction. The graphs show the change over time of the number of moles, the Gibbs' free energy, the reaction enthalpy, and the mole fractions. Also the graphs of total pressure and volume are demonstrated if their values change over time. For both cases, the graphs are given for a simulation time of $0.5 sec.$.

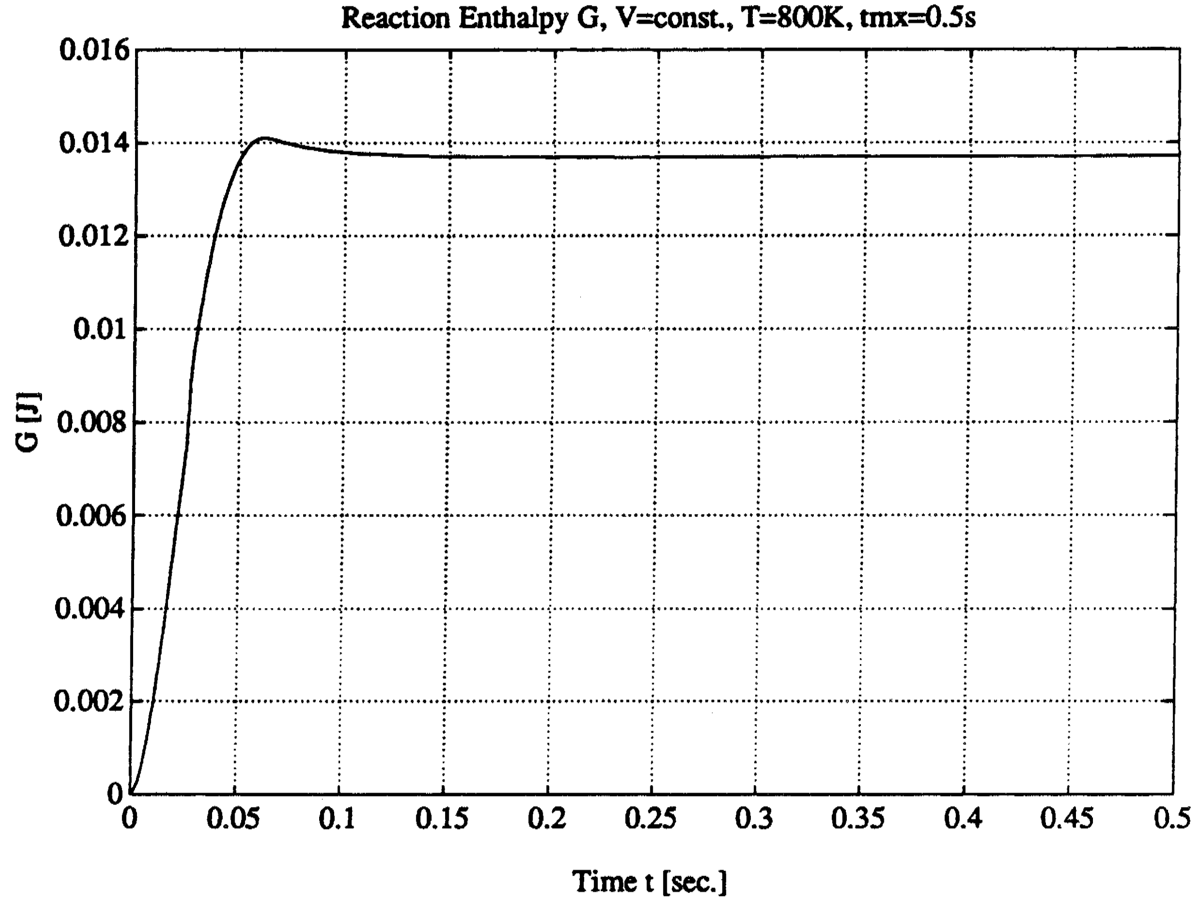
F.1 Isochoric–Isothermic Case

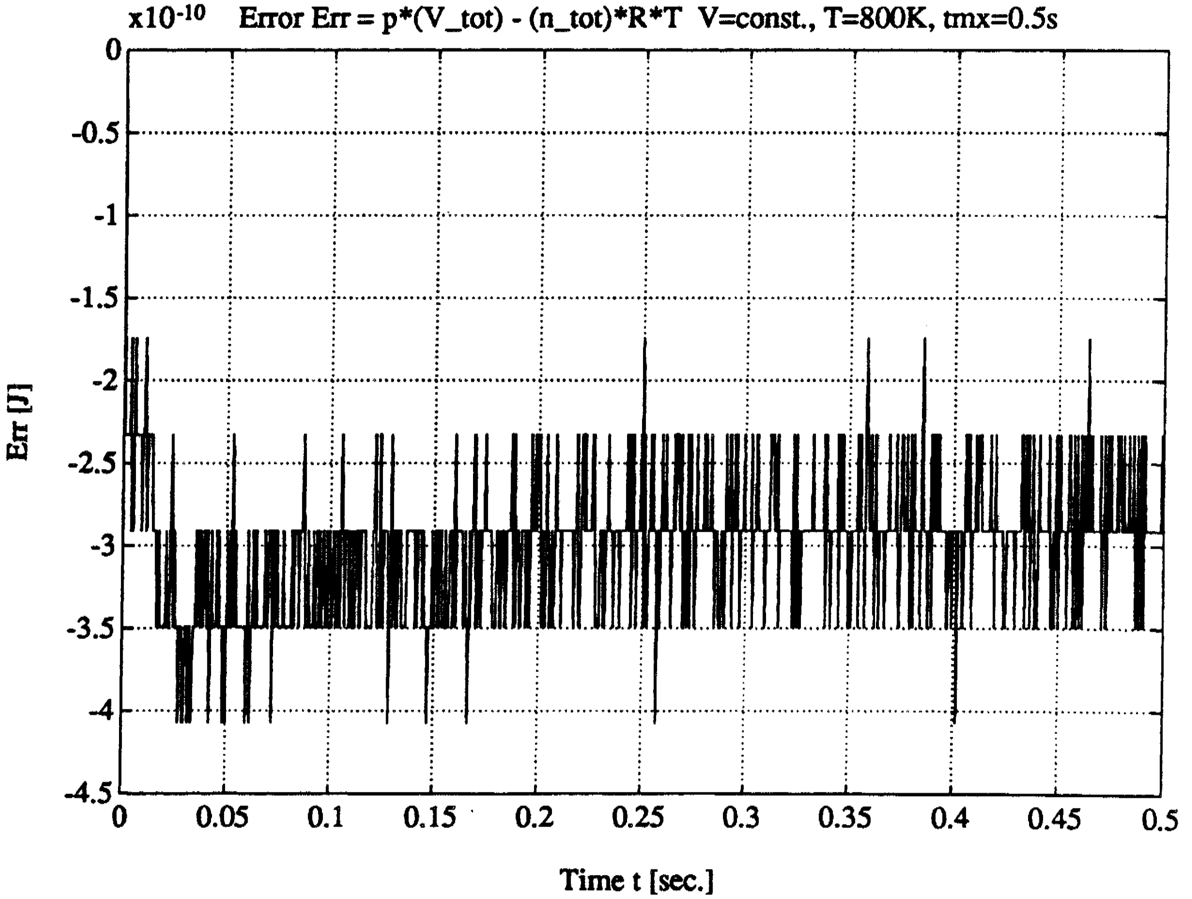
The constant value of the total volume V is given by the equation of state $V_{tot} = (n_{tot} \cdot R \cdot T)/p$. The numerical value turns out to $V_{tot} = (1.0E - 7 \cdot 8.314 \cdot 800)/1.0E5 = 6.6512E - 9m^3$.




x10^5
Total Pressure p, V=const., T=800K, tmx=0.5s
1.016
1.014
1.012
1.01
1.008
1.006
1.004
1.002
1
p [N/m^2]
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5
Time t [sec.]

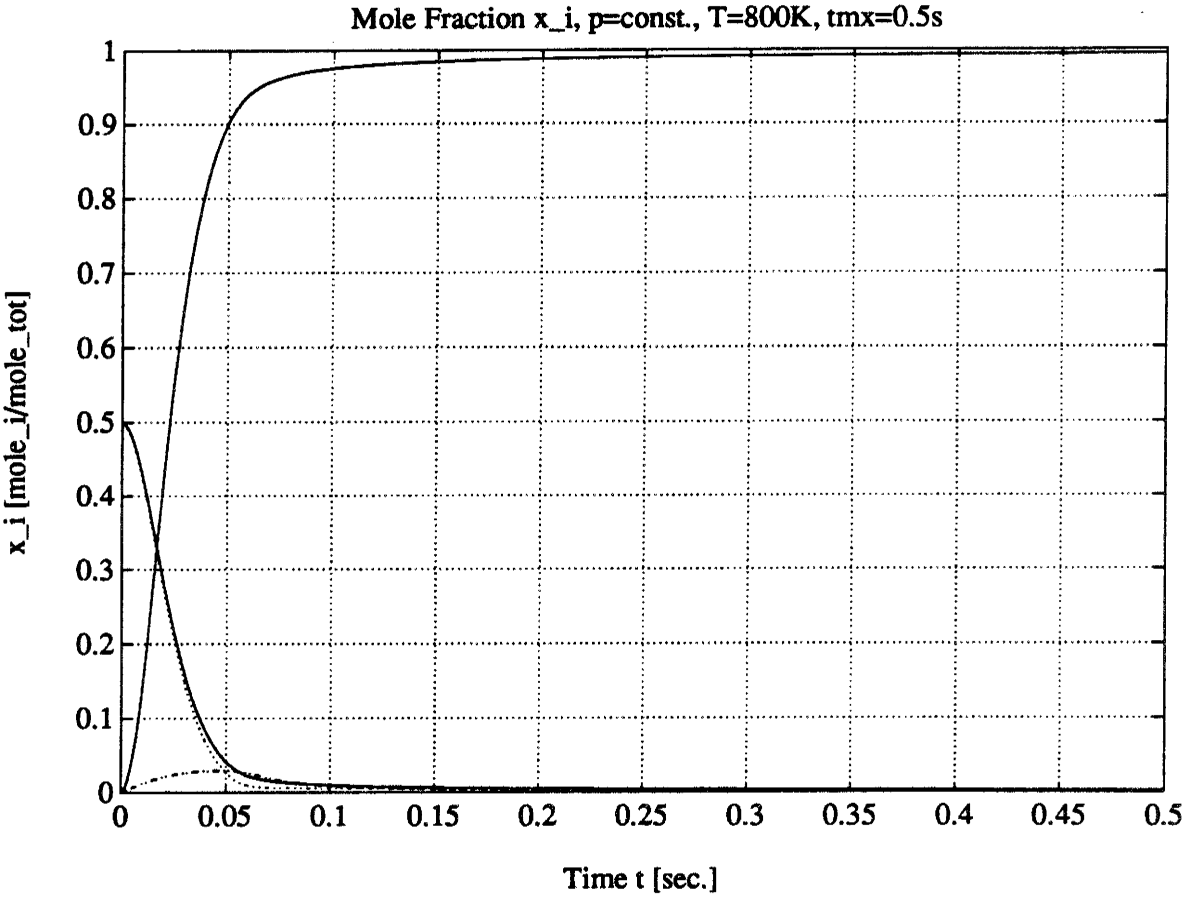

x10^-3
Gibbs' Free Enthalpy G, V=const., T=800K, tmx=0.5s
2
0
-2
-4
-6
-8
-10
-12
-14
G [J]
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5
Time t [sec.]

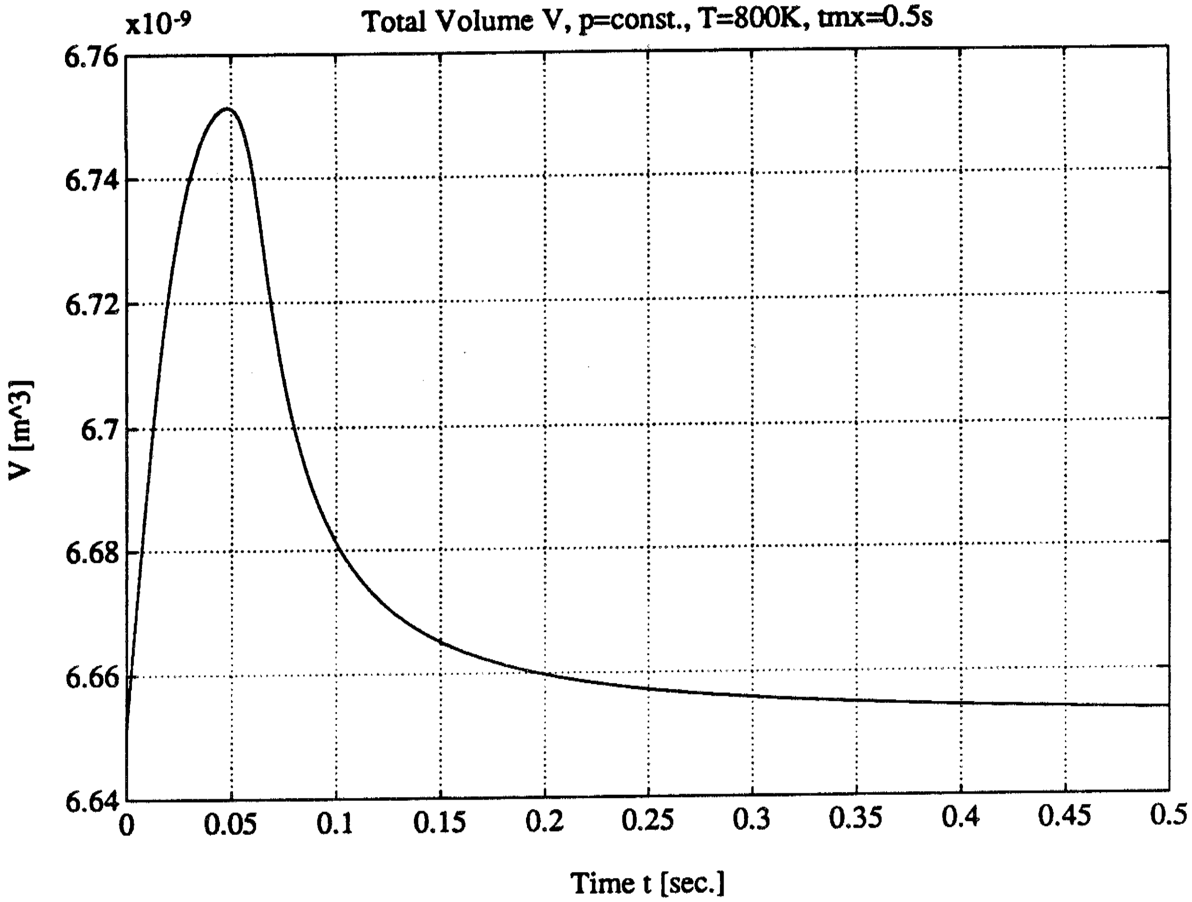

Reaction Enthalpy G, V=const., T=800K, tmx=0.5s
G [J]
0.016
0.014
0.012
0.01
0.008
0.006
0.004
0.002
0
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5
Time t [sec.]

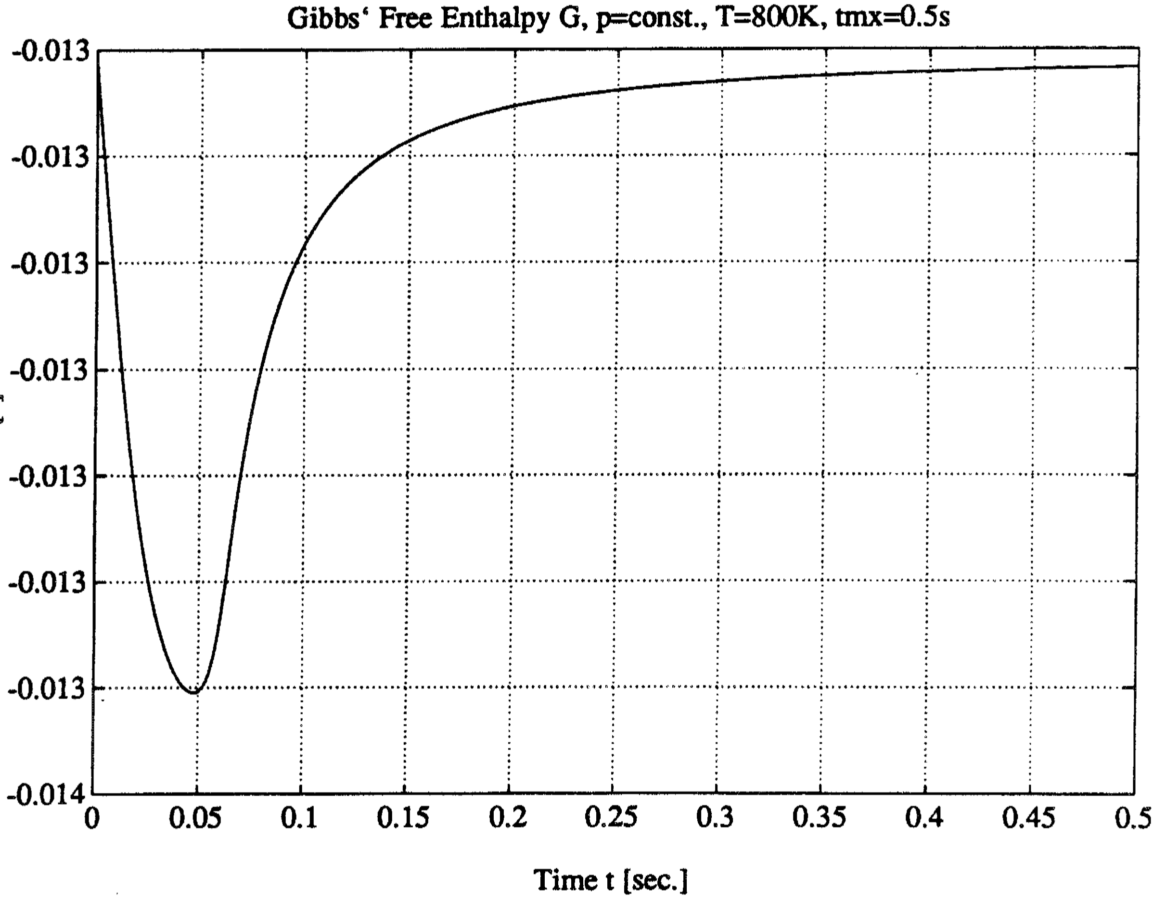

x10-10 Error Err = p*(V_tot) - (n_tot)*R*T V=const., T=800K, tmx=0.5s
Err [J]
0
-0.5
-1
-1.5
-2
-2.5
-3
-3.5
-4
-4.5
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5
Time t [sec.]

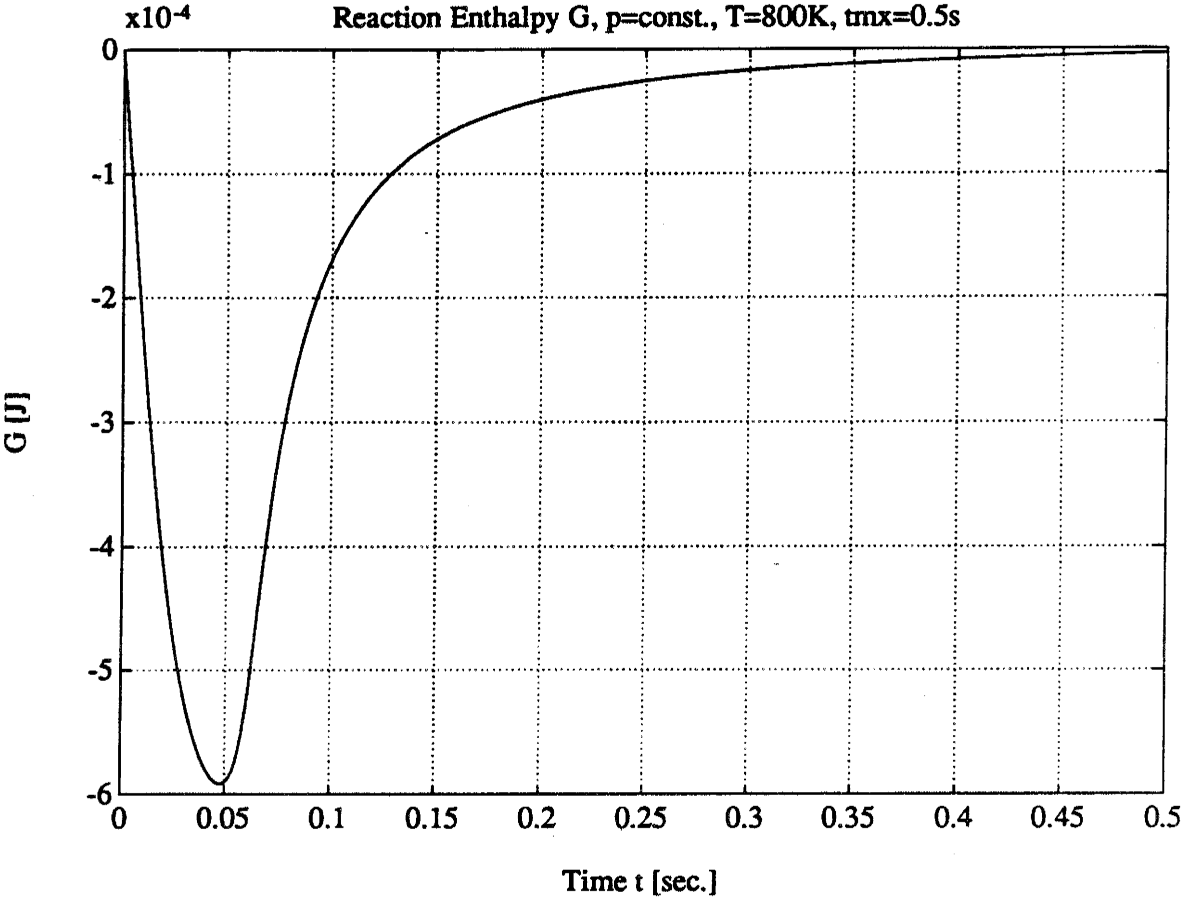
F.2 Isobaric–Isothermic Case

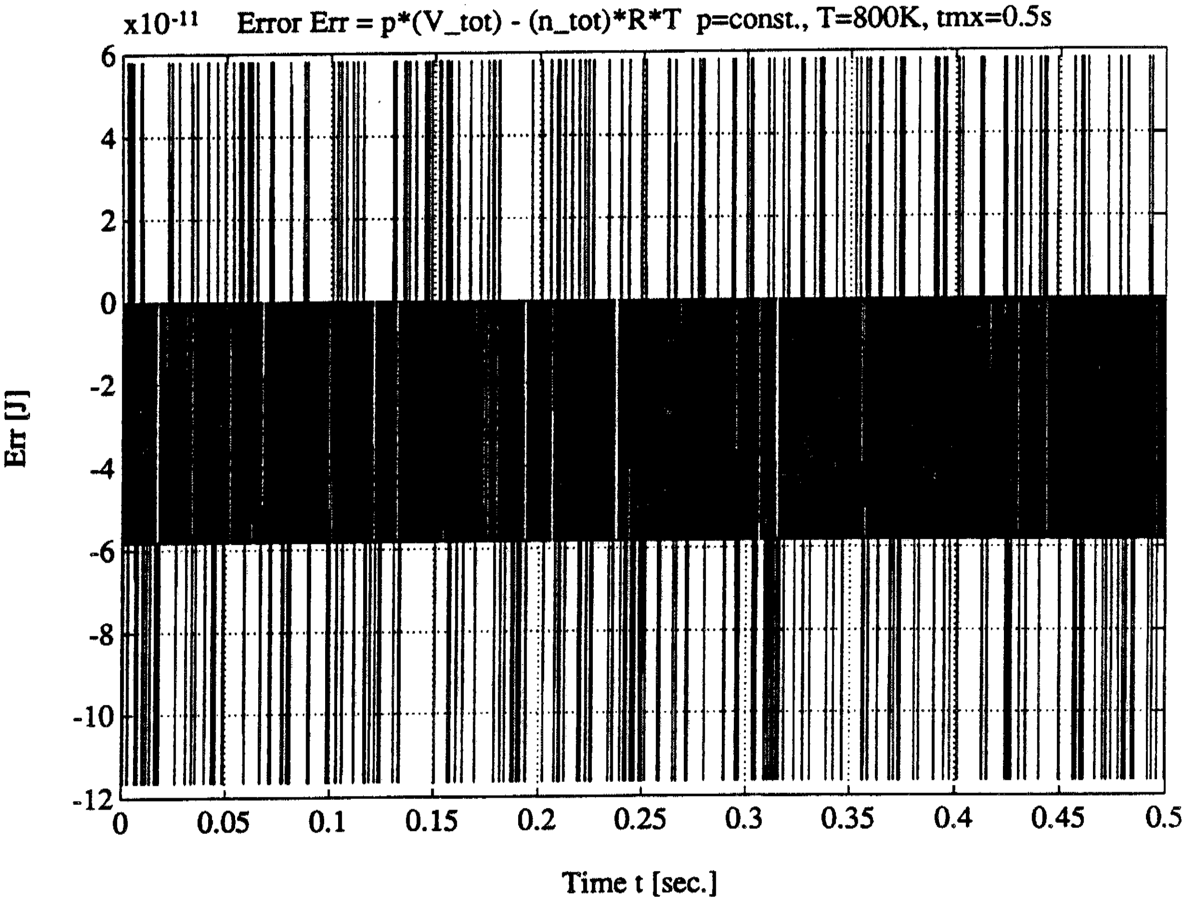
This time, the total volume V varies. The initial condition of V has the same value as above ($V_{tot} = 6.6512E - 9m^3$).




Total Volume V, p=const., T=800K, tmx=0.5s
x10^{-9}
6.76
6.74
6.72
6.7
6.68
6.66
6.64
V [m^3]
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5
Time t [sec.]


Gibbs' Free Enthalpy G, p=const., T=800K, tmx=0.5s
-0.013
-0.013
-0.013
-0.013
-0.013
-0.013
-0.013
-0.014
G [J]
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5
Time t [sec.]


x10^-4
Reaction Enthalpy G, p=const., T=800K, tmx=0.5s
0
-1
-2
-3
-4
-5
-6
G [J]
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5
Time t [sec.]


x10^-11
Error Err = p*(V_tot) - (n_tot)*R*T p=const., T=800K, tmx=0.5s
6
4
2
0
-2
-4
-6
-8
-10
-12
Err [J]
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5
Time t [sec.]