SAND96-8217
Unlimited Release
Printed May 1996

Distribution
Category UC-405

SURFACE CHEMKIN-III : A FORTRAN PACKAGE FOR ANALYZING HETEROGENEOUS CHEMICAL KINETICS AT A SOLID-SURFACE — GAS-PHASE INTERFACE

Michael E. Coltrin
Surface Processing Sciences Department
Sandia National Laboratories
Albuquerque, NM 87185

Robert J. Kee, Fran M. Rupley, and Ellen Meeks
Computational Mechanics Department
Sandia National Laboratories
Livermore, CA 94551

ABSTRACT

This document is the user's manual for the SURFACE CHEMKIN-III package. Together with CHEMKIN-III, this software facilitates the formation, solution, and interpretation of problems involving elementary heterogeneous and gas-phase chemical kinetics in the presence of a solid surface. The package consists of two major software components: an Interpreter and a Surface Subroutine Library. The Interpreter is a program that reads a symbolic description of a user-specified chemical reaction mechanism. One output from the Interpreter is a data file that forms a link to the Surface Subroutine Library, which is a collection of about seventy modular Fortran subroutines that may be called from a user's application code to return information on chemical production rates and thermodynamic properties. This version of SURFACE CHEMKIN-III includes many modifications to allow treatment of multi-fluid plasma systems, for example modeling the reactions of highly energetic ionic species with a surface. Optional rate expressions allow reaction rates to depend upon ion energy rather than a single thermodynamic temperature. In addition, subroutines treat temperature as an array, allowing an application code to define a different temperature for each species. This version of SURFACE CHEMKIN-III allows use of real (non-integer) stoichiometric coefficients; the reaction order with respect to species concentrations can also be specified independent of the reaction's stoichiometric coefficients. Several different reaction mechanisms can be specified in the Interpreter input file through the new construct of multiple materials.

ACKNOWLEDGMENTS

The development of SURFACE CHEMKIN-III was supported primarily by the U. S. Department of Energy, Office of Basic Energy Sciences, Division of Materials Science and Division of Chemical Sciences; it was also supported by the Advanced Research Projects Agency (ARPA) Materials Science Program. Work on the plasma modifications in SURFACE CHEMKIN-III was supported in part by a Sandia National Laboratories Cooperative Research and Development agreement (CRADA) with SEMATECH.

CONTENTS

ILLUSTRATIONS

TABLES

NOMENCLATURE

		Units
a_i	Pre-exponential factor in sticking coefficient expression	none
a_k	Activity of k^{th} bulk-phase species	none
a_{mk}	Coefficients to fits of thermodynamic data	depends on *m*
a_k^o	Standard state specific Helmholtz free energy for the k^{th} species	ergs / g
A	Area	cm^2
A_k^o	Standard state Helmholtz free energy for the k^{th} species	ergs / mole
A_i	Pre-exponential factor in the rate constant of the i^{th} reaction	depends on reaction
b_i	Temperature exponent in sticking-coefficient expression	none
c_i	Activation energy in sticking-coefficient expression	[cal / mole]*
c_{p_k}	Specific heat at constant pressure of the k^{th} species	ergs / (g K)
$C_{p_k}^o$	Standard state specific heat at constant pressure of the k^{th} species	ergs / (mole K)
D_{kj}	Ordinary multicomponent diffusion coefficients	cm^2 / sec
D_k^T	Thermal diffusion coefficient	g / (cm-sec)
E_i	Activation energy in the rate constant of the i^{th} reaction	[cal / mole]*
E_{ion}	Energy of a positive ionic species	[cal / mole]*
$E_{\text{ion},0}$	Energy threshold in ion-energy-dependent reactions	[cal / mole]*
$E_{\text{yield},0}$	Energy threshold in reaction yield expression	[cal / mole]*
f_i	Exponential constant in ion-energy-dependent reactions	none
F_{ki}	Forward reaction-order specified for the k^{th} species in i^{th} reaction	none
g_i	Exponential constant in ion-energy-dependent reactions	none
g_k^o	Standard state specific Gibbs free energy for the k^{th} species	ergs / mole
G	Bulk growth rate	cm / sec
G_k^o	Standard state Gibbs free energy for the k^{th} species	ergs / mole
h_k	Specific enthalpy of the k^{th} species	ergs / g
h_{yield}	Multiplicative factor in reaction yield expression	depends on reaction
H_f	Enthalpy of formation	ergs / mole
H_k^o	Standard state enthalpy of the k^{th} species	ergs / mole
H_k	Enthalpy of the k^{th} species	ergs / mole
i	Reaction index	

* By default, SURFACE CHEMKIN uses activation energies in calories instead of ergs.

I	Total number of reactions	
k	Species index	
k_{f_i}	Forward rate constant of the i^{th} reaction	depends on reaction
k_{r_i}	Reverse rate constant of the i^{th} reaction	depends on reaction
K	Total number of species	
K_b	Total number of bulk species	
$K_b^f(n)$	Index of the first bulk-phase species in phase n	
$K_b^l(n)$	Index of the last bulk-phase species in phase n	
K_g	Total number of gas-phase species	
K_g^f	Index of the first gas-phase species	
K_g^l	Index of the last gas-phase species	
K_s	Total number of surface species	
$K_s^f(n)$	Index of the first surface species in phase n	
$K_s^l(n)$	Index of the last surface species in phase n	
K_{phase}	Vector containing the number of species in each phase	
K_{c_i}	Equilibrium constant in concentration units for the i^{th} reaction	depends on reaction
K_{p_i}	Equilibrium constant in pressure units for the i^{th} reaction	none
M	Number of coefficients in polynomial fits to C_p^o/R	
m_k	Mass of the k^{th} species	g
n	Index for phases	
n	Surface-normal unit vector; points from the gas into the bulk layer	
N	Total number of phases	
N_A	Avogadro's number	
N_b	Total number of bulk phases	
N_b^f	Index of the first bulk phase	
N_b^l	Index of the last bulk phase	
N_g	Number of gas phases (always equals 1)	
N_s	Total number of surface site types (phases)	
N_s^f	Index of first surface phase	
N_s^l	Index of last surface phase	
P	Pressure	dynes/cm^2
P_{atm}	Pressure of one standard atmosphere	dynes/cm^2
q_i	Rate of progress of the i^{th}reaction	moles/(cm^{2}sec)
R	Universal gas constant	ergs/(mole K)
R_c	Universal gas constant, in same units as activation energy E_i	[cal/(mole K)]

R_{ki}	Reverse reaction-order specified for he k^{th} species in i^{th} reaction	none
$\dot{s}_k$	Production rate of the k^{th} species from surface reactions	moles / (cm^2 sec)
s_k^o	Standard state specific entropy of the k^{th} species	ergs / (g K)
S_k^o	Standard state entropy of the k^{th} species	ergs / (mole K)
t	Time	sec
T	Temperature	K
T_o	Ambient temperature	K
T_e	Electron temperature	K
t_i	Exponential constant in reaction yield expression	none
u	Convective velocity, Stefan flow velocity	cm / sec
u_k	Specific internal energy of the k^{th} species	ergs / g
u_i	Exponential constant in reaction yield expression	none
U_k	Internal energy of the k^{th} species	ergs / mole
U_k^o	Standard state internal energy of the k^{th} species	ergs / mole
V	Volume	cm^3
V_k	Diffusion velocity of the k^{th} species	cm / sec
W_{ion}	Molecular weight of the positive ionic species in a Bohm-type reaction	g / mole
W_k	Molecular weight of k^{th} species	g / mole
$\overline{W}$	Mean molecular weight of a mixture	g / mole
x	Height	cm
X	Array of species mole fractions	none
Z	Array of surface species site fractions	none
X_k	Mole fraction of the k^{th} species	none
$[X_k]$	Molar concentration of the k^{th} species	moles / cm^3
Y_k	Mass fraction of the k^{th} species	none
$Z_k(n)$	Site fraction of the k^{th} species on site n	none
α_i	Parameter in mechanism	
β_i	Temperature exponent in the rate constant of the i^{th} reaction	none
Γ_n^o	Standard-state density for surface phase n	moles / cm^2
Γ_n	Site density for surface phase n	moles / cm^2
$\dot{\Gamma}_n$	Production rate for surface phase n	moles / (cm^2 sec)
Γ_{tot}	Site density summed over all surface phases	moles / cm^2
γ_i	Sticking coefficient for the i^{th} surface reaction	none
λ	Thermal conductivity	erg / (cm K sec)
ρ	Mass density	g / cm^3

ρ_k	Mass density of the k^{th} bulk species	g/cm^3
υ_{ki}	Stoichiometric coefficient of species *k* in reaction *i*, $\upsilon_{ki} = \upsilon''_{ki} - \upsilon'_{ki}$	
υ'_{ki}	Stoichiometric coefficient of the k^{th} reactant species in the i^{th} reaction	
υ''_{ki}	Stoichiometric coefficient of the k^{th} product species in the i^{th} reaction	
$\dot{\omega}_k$	Production rate of the k^{th} species from gas-phase reactions	$\text{mole}/(\text{cm}^3\text{sec})$
σ	Stefan-Boltzmann constant	$\text{erg}/(\text{cm}^2\text{sec K}^4)$
σ_k	Number of sites occupied by the k^{th} species	
ε	Emissivity	none
ε_{ki}	Coverage parameter	[cal / mole K]
η_{ki}	Coverage parameter	none
μ_k	Chemical potential of the k^{th} species	none
μ_k^o	Chemical potential of the k^{th} species	none
μ_{ki}	Coverage parameter	none
Φ	Dependent variable in an application code	
ψ	Yield enhancement factor (ion-energy-yield reaction)	none
χ_k	Chemical symbol of the k^{th} species	

SURFACE CHEMKIN-III:* A FORTRAN PACKAGE FOR ANALYZING HETEROGENEOUS CHEMICAL KINETICS AT A SOLID-SURFACE – GAS-PHASE INTERFACE

I. INTRODUCTION

Heterogeneous reaction at the interface between a solid surface and adjacent gas is central to many chemical processes. Our development of the software package SURFACE CHEMKIN-III was motivated by our need to understand the complex surface chemistry in chemical vapor deposition systems involving silicon, silicon nitride, and gallium arsenide. However, we have developed the approach and implemented the software in a general setting. Thus, the software has found use in such diverse applications as chemical vapor deposition, chemical etching, combustion of solids, and catalytic processes, and a wide range of chemical systems. We believe that it provides a powerful capability to help model, understand, and optimize important industrial and research chemical processes.

The SURFACE CHEMKIN-III software is designed to work in conjunction with the CHEMKIN-III[1] software, which handles the chemical kinetics and thermodynamic properties in the gas phase. It may also be used in conjunction with the Transport Property Package,[2,3] which provides information about molecular diffusion. Thus, these three packages provide a foundation on which a user can build application software to analyze gas-phase and heterogeneous chemistry in flowing systems.

These packages should not be considered "programs" in the ordinary sense. That is, they are not designed to accept input, solve a particular problem, and report the answer. Instead, they are software tools intended to help a user work efficiently with large systems of chemical reactions and develop software representations of systems of equations that define a particular problem. It is up to the user to solve the problem and interpret the answer. A general discussion of this structured approach for simulating chemically reacting flow can be found in Kee and Miller.[4]

 This document describes SURFACE CHEMKIN-III. We expect that the software package will continue to evolve, and thus later versions may render portions of this document obsolete.

Structure and Use of SURFACE CHEMKIN-III

Using the SURFACE CHEMKIN-III package is analogous to using the CHEMKIN-III[1] package, and the SURFACE CHEMKIN-III package can only be used after the CHEMKIN-III Interpreter has been executed.† Therefore, it is necessary to be familiar with CHEMKIN-III before the SURFACE CHEMKIN-III package can be used effectively. The CHEMKIN-III interpreter introduces the chemical elements that are used in either the gas-phase reaction mechanism or the surface-reaction mechanism. Gas-phase species (which can appear in surface reactions) are also introduced with the CHEMKIN-III Interpreter. Thus, if a gas-phase species appears in the surface-reaction mechanism but not in the gas-phase mechanism, the user must identify this species in the CHEMKIN-III Interpreter.

Like CHEMKIN-III, the SURFACE CHEMKIN-III package is composed of two blocks of Fortran code:

-- the Surface Interpreter
-- the Surface Subroutine Library

To apply SURFACE CHEMKIN-III to a problem, the user must execute an application program that describes the particular set of governing equations. To aid in this programming effort, the application can call CHEMKIN-III and SURFACE CHEMKIN-III subroutines that define the terms in the equations relating to equation of state, chemical production rates, and thermodynamics, and then combine the results to define the problem. An expanding number of application programs using SURFACE CHEMKIN-III have been written. The subroutines in the Library may be called from Fortran or C.

After running the CHEMKIN-III Interpreter, the user runs the SURFACE CHEMKIN-III Interpreter, which first reads the user's symbolic description of the surface-reaction mechanism and then extracts from a Thermodynamic Database the appropriate thermodynamic information for the species involved.[7] CHEMKIN-III and the SURFACE CHEMKIN-III can share a common database. The database has essentially the same format as that used by the NASA complex chemical equilibrium code of Gordon and McBride.[8] The output of the SURFACE CHEMKIN-III Interpreter is the Surface Linking File, which contains all the pertinent information on the elements, species, and reactions in the surface reaction mechanism. Information on gas-phase species comes from the CHEMKIN-III Linking File, and thus is duplicated in the two linking files.

---

† Caution: SURFACE CHEMKIN III works only with the newer CHEMKIN-III, not the earlier CHEMKIN[5] or CHEMKIN-II[6] packages.

The Surface Linking File is read by an initialization routine in the Surface Subroutine Library that is called from the user's code. The purpose of the initialization is to create three data arrays (one integer, one floating point, and one character data type) for use internally by the other subroutines in the Surface Subroutine Library.

The Surface Subroutine Library has approximately seventy subroutines that return information on elements, species, reactions, thermodynamic properties, and chemical production rates. Generally, the input to these routines will be the state of gas and the surface—pressure, temperature, and species composition. The species composition is specified in terms of gas-phase mole fractions, surface site fractions, and bulk-phase activities; surface site densities are also input to complete the specification of the state of the surface.

Example

We illustrate the use of SURFACE CHEMKIN-III by a simple example involving deposition of silicon. The surface-reaction mechanism is shown in Fig. 1 as it appears for the input file to the Surface Interpreter. The first two lines identify a site type called "SILICON" that has a site density of 1.66×10^{-9} moles/cm^2. Only one species, SI(S), exists on this site type. The bulk material is identified as SI(B), and it has a mass density of 2.33 g/cm^3. This is a very simple example that has only one site type occupied by only one species and only one pure bulk material. In general, however, an input file could specify many different site types, each of which could be occupied by a variety of species. Furthermore, there could be several bulk-phase mixtures that could each be composed of several species. Examples of all these possibilities are given later in the manual.

The reaction mechanism itself is listed next. The symbol => in each reaction expression indicates that all the reactions are irreversible. The three numbers following each reaction expression are its Arrhenius rate parameters (pre-exponential factor, temperature exponent, and activation energy).

All of the reactions in the mechanism have the same form: a gas-phase species reacting on a silicon site. The reaction of silane at the surface is illustrated in Fig. 2. Each silicon-containing gas-phase species can react on an atomic surface site, SI(S), to deposit a silicon atom as SI(B) and release hydrogen back into the gas phase. We have included SI(S) as both a reactant and a product to indicate that a "site" must be available at which the gas-phase species can react. In the example, however, the surface silicon

SI(S) is distinguished from the bulk deposit SI(B) by virtue of its position as the top-most atom at the surface. Therefore, each time a SI(S) is consumed by a reaction the bulk layer becomes one atom thicker

```
SITE/SILICON/ SDEN/1.66E-09
       SI(S)
BULK   SI(B) /2.33/

REACTIONS

SIH4 + SI(S) =>SI(S) +  SI(B) + 2H2           1.05E17      0.5 40000
SI2H6 +2SI(S) =>2SI(S) + 2SI(B) + 3H2         4.55E26      0.5 40000
SIH2 + SI(S) =>SI(S) +  SI(B) +  H2           3.9933E11    0.5 0
SI2H2 +2SI(S) => 2SI(S) + 2SI(B) +  H2        1.7299E20    0.5 0
2SI2H3 +4SI(S) => 4SI(S) + 4SI(B) + 3H2       6.2219E37    0.5 0
H2SISIH2 +2SI(S) => 2SI(S) + 2SI(B) + 2H2     1.7007E20    0.5 0
2SI2H5 +4SI(S)=> 4SI(S) + 4SI(B) + 5H2        6.1186E37    0.5 0
2SIH3 +2SI(S) => 2SI(S) + 2SI(B) + 3H2        2.3659E20    0.5 0
2SIH +2SI(S) => 2SI(S) + 2SI(B) +  H2         2.4465E20    0.5 0
SI + SI(S) =>  SI(S) +  SI(B)                 4.1341E11    0.5 0
H3SISIH  +2SI(S) => 2SI(S) + 2SI(B) + 2H2     1.7007E20    0.5 0
SI2 +2SI(S) => 2SI(S) + 2SI(B)                1.7607E20    0.5 0
SI3 +3SI(S) => 3SI(S) + 3SI(B)                8.6586E28    0.5 0
END
```

Figure 1 Sample Reaction Mechanism as Read by the SURFACE CHEMKIN-III Interpreter

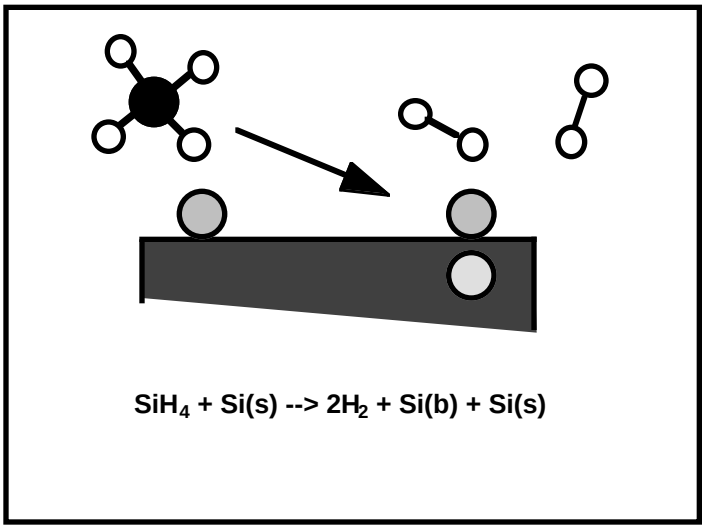


Figure 2 Illustration of Gas-Phase Silane Reacting at a Surface to Deposit a Silicon Atom and Release Two Hydrogen Molecules into the Gas Phase

and the silicon atom that just left the gas now forms the top-most surface layer, i.e., SI(S). For this mechanism, the SI(S) could have been just as well left out of the mechanism entirely. However, if other gas-phase species had been present (say phosphine carrying phosphorus as a dopant), these species could compete for the available silicon sites on the surface. Thus, by writing the reactions as we have, we have left open the possibility for other species to occupy surface sites and thus inhibit the deposition of silicon.

As an example of the full use of SURFACE CHEMKIN-III, assume that an application program needs to evaluate a boundary condition concerning the energy balance at a surface of an isothermal particle. The energy balance would take the following form (with the surface normal $\overline{n}$ pointing into the particle):

$$\overline{n}\cdot\left[-\lambda\nabla T+\sum_{k=1}^{K_g}\rho Y_k\left(\overline{V}_k+\overline{u}\right)h_k\right]=\sigma\varepsilon\left(T^4-T_0^4\right)+\sum_{k=K_s^f\left(N_s^f\right)}^{K_b^l\left(N_b^l\right)}\dot{s}_k W_k h_k \qquad (1)$$

The dependent variables in this expression are the temperature T, gas-phase mass fractions Y_k and convective velocity $\overline{u}$. The surface site fractions and the bulk-species activities are also dependent variables, but do not appear explicitly in the expression. The first term in this equation describes thermal conduction to the surface from the gas phase. The thermal conductivity λ would be evaluated by a call to the Transport Library,[3] and the temperature gradient could be evaluated by finite differences. The second term concerns the diffusive and convective flux of energy by gas-phase species at the surface. The mass density ρ and the gas-phase enthalpies h_k would be evaluated by calls to the CHEMKIN-III Library. The gas-phase species diffusion velocities V_k would be evaluated in terms of diffusion coefficients that are obtained from the Transport Package and finite difference approximations to the species gradients. The first term on the right-hand side concerns the thermal radiation to or from the surface.

We now concentrate on the final term, which concerns the energy generated or consumed from surface reaction. The summation is over all surface and bulk species, and the factors in the summation are the production rate of surface and bulk species by surface reaction, $\dot{s}_k$, the species molecular weights, W_k, and the enthalpies of the surface and bulk species, h_k. The Fortran representation of this term begins with Surface Library subroutine calls (the output variables are underlined to help distinguish them):

CALL SKINIT(LSIWK, LSRWK, LSCWK, LINKSK, LOUT, ISKWRK, RSKWRK, CSKWRK, IFLAG)
CALL SKWT(ISKWRK, RSKWRK, WT)
CALL SKHMS(T, ISKWRK, RSKWRK, HMS)
CALL SKRAT(P, T, ACT, SDEN, ISKWRK, RSKWRK, SDOT, SITDOT) }

The complete details for these calls are explained in later chapters of this document, the object here being to illustrate the relative simplicity of a SURFACE CHEMKIN-III application. Briefly, the first call is to the initialization subroutine SKINIT, which reads the Surface Linking File created by the Surface Interpreter and creates the three work arrays. LSIWK, LSRWK, and LSCWK are the dimensions provided by the user for the data arrays ISKWRK, RSKWRK, and CSKWRK. LINKSK is the logical file number of the Surface Linking File, LOUT is the logical file number for printed diagnostic and error messages, and IFLAG is an integer error flag. In the remaining calls, P and T are the pressure and temperature. The array ACT contains the gas-phase mole fractions, the surface site fractions, and the bulk species activities. The output variable arrays, HMS and WDOT, correspond to the factors in the summation, i.e., HMS = h_k, and SDOT = $\dot{s}_k$.

The Fortran representation of the summation in the last term, given by combining the results of the above subroutine calls, is simply

```
      SUM=0.0
      DO 100 K=FIRST_SURFACE_SPECIES, LAST_BULK_SPECIES
            SUM = SUM + SDOT(K)*WT(K)*HMS(K)
100   CONTINUE
```

The species indices FIRST_SURFACE_SPECIES and LAST_BULK_SPECIES are also available from a call to the Surface Library, which Chapters VII and VIII explain in detail.

Transportability

The SURFACE CHEMKIN-III package was developed on Unix workstations. However, we have not taken advantage of any special machine-dependent features. Written entirely in ANSI standard Fortran-77, the code has been transported to many computer systems. Since double-precision code is often required on small-word-length (i.e., 32-bit word) computers, we provide a utility to convert between single- and double-precision versions of the source code.[13]

Organization of this Report

Chapter II introduces the formalism developed to describe surface chemistry behavior. Unlike the case of gas-phase chemistry, where much software has been written to analyze mass-action kinetics and chemically reacting flow, elementary heterogeneous reactions are seldom treated with the generality provided in this package. For the treatment in SURFACE CHEMKIN-III we first had to define a systematic convention to translate heterogeneous reaction ideas into a form that was amenable to efficient computation.

In the spirit of CHEMKIN-III, Chapter III is a compendium of some important equations in heterogeneous chemical kinetics. Many of the equations are simply definitions; but, in any case, derivations are either sketchy or not given. Although some readers will find the equations quite familiar, we find it useful to have them stated concisely in one document. For most equations, the package contains a subroutine that, when given the variables on the right-hand side, returns the variable on the left. Below some of the equation numbers is stated (in brackets) the name of the subroutine that provides information about that equation.

Using CHEMKIN-III and SURFACE CHEMKIN-III (and possibly the Transport Package) requires the manipulation of many programs and files. Chapter IV explains the mechanics of using these software packages and describes the job-control logic for running a typical problem.

Chapter V explains the SURFACE CHEMKIN-III Interpreter and how to set up the required symbolic input to define a reaction mechanism. We have allowed the possibility of including multiple site types, multiple mixtures of bulk species, and multiple materials. Each site type and bulk mixture may contain several species. Therefore, the data structures needed to refer to the phases and the species can be complex. Chapter VI provides detailed information on the computational data structures that we use to refer to phases and species in each phase.

Chapters VII and VIII describe the Surface Subroutine Library, Chapter VII being composed of short descriptions for quick reference and Chapter VIII (an alphabetical listing) explaining the input and output in the call sequence. To demonstrate SURFACE CHEMKIN-III explicitly, Chapter IX goes through a sample problem in detail.

Appendix A defines the storage allocation of the three data arrays that are created from the Linking File. With this information, it is possible for a user to create new subroutines for the library to suit a specialized need that was not anticipated in the current version of the Library.

II. DEVELOPMENT OF SURFACE FORMULATION

In this chapter we discuss the mathematical formalism developed to describe surface kinetics for events such as adsorption, desorption, surface reactions, and deposition. This formalism is essentially a set of rules for keeping track of surface species concentrations, conservation of mass and surface sites, mass-action kinetics, and rates (such as deposition or etching rates).

For this discussion we define three types of species: gas-phase, surface, and bulk. The first is a species in the gas phase above the surface, which might be denoted in a reaction by "(g)". A surface species, perhaps denoted by "(s)", is defined to be the chemical species on the top-most layer of the solid, i.e., at the solid-gas interface.* Each surface species occupies one or more "sites;" the total number of sites is often assumed to be conserved. Any species in the solid below the surface layer is defined to be a "bulk" species and might be denoted by "(b)". In writing elementary reactions for a surface mechanism in a kinetic model, mass, elemental composition, and charge must all be conserved.

There can be more than one type of site on the surface. For example, one could specify that a surface consists of "ledge" sites and "plane" sites. The number of sites of each type might be a characteristic of the crystal face. In our formalism there can be any number of site types. One may define a species that only resides on a certain type of site. For example, the thermodynamic properties of a hydrogen atom on a ledge site might be different from a hydrogen on a plane site, and they could be specified as different species (even though their elemental composition is the same). The population of different species occupying a given type of site is specified by site fractions. The sum of the site fractions of the species on a given site is 1. (Thus an "open site" is considered as a distinct species.)

In the bulk there can be different types of bulk species. The simplest consists of a pure species. There can be any number of pure bulk species. It is also possible to specify a bulk mixture with components A and B. The composition of the bulk phase may be input by the user by specifying the activities of each of the bulk-phase components.

The activity of a bulk species is defined in terms of the following equation for the chemical potential:

---

* In actuality there is no constraint that the surface must be only one atom thick. However, defining a "surface" that is several monolayers thick may be conceptually much more difficult to deal with.

$$\mu_k(T,P,X) = \mu_k^o(T) + RT\ln\left(a_k(T,P,X)\right), \tag{2}$$

where μ_k^o is the standard state chemical potential of species *k* at temperature *T* and at the standard pressure, 1 atm. The vector X represents an array of the mole fractions of the species. Two conventions are normally used to complete the specification of the activity coefficient:

1. If the standard state is defined as being a pure bulk phase of *k* at temperature *T* and 1 atm, then a_k is further defined to approach X_k as X_k approaches 1 at 1 atm (Raoult's Law).
2. If the standard state is defined as the hypothetical state of species k in infinite dilution in bulk-phase species *j* at temperature *T* and 1 atm, then $a_k$is further defined to approach X_k as X_k approaches 0 at 1 atm (Henry's Law).

Both conventions for the standard state work with SURFACE CHEMKIN-III, as do any other definitions that conform to the formalism expressed by Eq. (2) for μ. $\mu_k^o(T)$ is specified through the entry for species *k* in the thermodynamics data file. The value of $a_k(T,P,X)$ is required as input to all SURFACE CHEMKIN-III subroutines that calculate bulk phase thermodynamic quantities and reaction rates. Therefore, if desired, users can construct their own subroutines to calculate $a_k(T,P,X)$, possibly incorporating models for non-ideality of the bulk phase, and can have the consequences properly incorporated into the surface kinetics mechanism. Although the activities of all components of an ideal solution must sum to 1, this condition is not enforced in SURFACE CHEMKIN-III. (It is, however, enforced in many of the application codes that employ SURFACE CHEMKIN-III.)

Since SURFACE CHEMKIN-III allows for a number of different types of species (gas species, any number of types of surface sites, species residing on surface sites, pure bulk species, bulk mixtures, and species present in a bulk mixture), it is necessary to be able to keep track of them. We use the notion of different physical "phases" to group the chemical species in a problem. Our nomenclature corresponds to that of Eriksson,[9] which has been extended to account for surface sites. The order in which we discuss the phases is the order in which SURFACE CHEMKIN-III groups them.

Phase number 1 is the gas phase. Information about species in the gas phase is passed to SURFACE CHEMKIN-III from the gas-phase CHEMKIN-III interpreter. The mole fractions of the gas-phase species correspond to species activities, mentioned below.

We consider every type of surface site to be a distinct "phase." If there are N_s types of sites specified, then phases 2 through $N_s + 1$ are these sites. The user can specify the names of chemical

species that exist only on a given site type. The site fractions of all the species that can exist on a given type of site (phase) sum to 1. The surface species site fractions also correspond to activities.

The next type of phase is a bulk mixture. If a given problem has N_b different types of bulk mixtures, then these are considered to be phases $N_s + 2$ through $N_s + N_b + 1$. The user specifies the names of the species that can exist in a given bulk mixture. The amounts of these species are determined indirectly by their activities, which the user supplies. A limiting case is a pure bulk species, which is treated as a bulk mixture with only one chemical species, whose activity is unity if the chemical potential does not depend on pressure.

We now consider in more detail how to write chemical reactions involving surface and bulk species. A chemical species on the top layer of the solid, i.e., a surface species, occupies a site. For example, an arsine molecule adsorbed on a surface could occupy a site, and might be denoted AsH_3(s). Another example might be a bare gallium atom, Ga(s), on top of a gallium arsenide crystal. What happens if another species, say a gas-phase AsH_3, lands on top of the Ga(s) (see Fig. 3)? In this case the gallium atom that was at the surface is covered up, so it is no longer a surface species. In our nomenclature it has become a bulk species. The adsorbed AsH_3 now occupies the top-most layer at this site, so it has become the surface species AsH_3(s). In our formalism, we would write the adsorption reaction in Fig. 3 as

$$AsH_3(g) + Ga(s) \rightarrow AsH_3(s) + Ga(b). \tag{3}$$

In this reaction, the number of sites included on the left-hand side of the reaction equals the number on the right-hand side; the reaction conserves sites.

Suppose that we had wanted to describe the reverse reaction, i.e., desorption of AsH_3 from the surface. We would then write the reaction as

$$AsH_3(s) + Ga(b) \rightarrow AsH_3(g) + Ga(s). \tag{4}$$

Here, Ga(b) is included as a reactant in order to achieve site and elemental balance. We denote the formalism described in reactions (3) and (4) as the Atomic Site Formalism.

An alternate way of posing the above example is to look at the situation on the left side of Fig. 3 not as having a surface gallium atom on a site, but to say that this is really an "open" site at which some event may take place (see Fig. 4). We would write the reaction of Fig. 4 as

$$O(s) + AsH_3(g) \rightarrow AsH_3(s), \quad (5)$$

where the symbol O(s) was used to denote an open site. Since O(s) contains no elements (it is empty), this reaction conserves both sites and elements. We denote the formalism described in reaction (5) as the Open Site Formalism.

The Atomic Site and Open Site Formalisms are equally valid ways of stating these surface reactions, and either is allowed by the SURFACE CHEMKIN-III code. Personal preference or, perhaps, the nature of a particular problem might dictate one over the other. Note that an "open" site must be considered as a species.

What are the thermochemical implications of reactions such as (3) and (5)? In the Atomic Site Formalism, the interpretation is straightforward. In reaction (3) we have converted $AsH_3(g)$ and Ga(s) into $AsH_3(s)$ and Ga(b). Thus, the change in a thermochemical property, e.g., ΔH_{rx}, is just the difference in the heats of formation of the products and the reactants. What about in the Open Site Formalism? What are the properties of O(s), the open site? Because these two formalisms describe an identical physical event, it is evident the properties of the open site must be related to those of Ga(b) and Ga(s). For example, the heat of formation of this open site is just

$$\Delta H_f(O(s)) = \Delta H_f(Ga(s)) - \Delta H_f(Ga(b)). \quad (6)$$

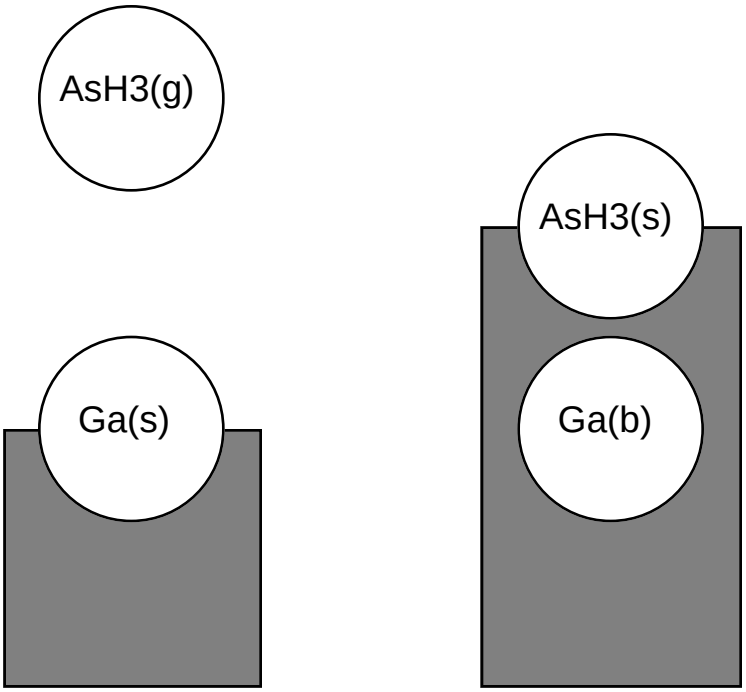


Figure 3 Illustration of an Adsorption Reaction using the Atomic Site Formalism

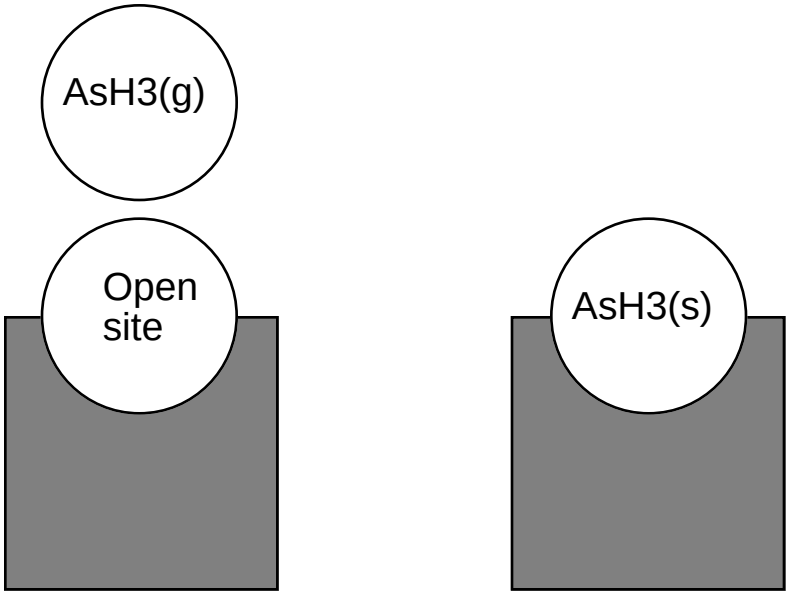


Figure 4 Illustration of an Adsorption Reaction using the Open Site Formalism

III. CHEMICAL RATE AND THERMODYNAMIC EXPRESSIONS

This chapter lists expressions and equations that are useful in formulating chemically reacting flow problems. For many expressions and equations the subroutine that evaluates it is named.

Species can exist in the gas phase, on surface sites, or in bulk mixtures. In some cases it is desirable to refer to information about species without regard to the phases, and in other cases it is desirable to determine information about species in one particular phase or group of phases. Therefore, before beginning to discuss our formalism in terms of mathematical expressions, we introduce a nomenclature that facilitates manipulating species information.

Information about a species (say a thermodynamic property) is presumed to be available in ordered arrays beginning with the first gas-phase species, continuing through the surface species, and ending with the last bulk species. In the expressions and equations below we presume that there are K species, and we use the index k to refer to a specific species. There are K_g gas-phase species, which, by convention, are always the first entries in the species arrays. The index of the first gas-phase species is K_g^f ($K_g^f = 1$ by our convention) and the last gas-phase species index is K_g^l $\left(K_g^l = K_g\right)$. Thus, the gas-phase species indices are $K_g^f \le k \le K_g^l$. In a similar way surface species indices are in the range $K_s^f \le k \le K_s^l$ and bulk species are in the range $K_b^f \le k \le K_b^l$. The surface species may be arranged on any number of sites, and the bulk species may exist in any number of bulk mixtures. Furthermore, situations can occur in which there are no surface species and / or no bulk species.

As discussed in Chapter II, the species are grouped in "phases." The first is the gas phase, whose index n=1. The next N_s phases (if they are present) are the surface sites, whose phase indices are bounded by $N_s^f \le n \le N_s^l$. The final N_b phases are the bulk mixtures, whose indices are bounded by $N_b^f \le n \le N_b^l$. In each phase n there are $K_{phase}(n)$ species, and those species have indices in the range $K_{phase}^f(n) \le k \le K_{phase}^l(n)$.

Concentration Units

In a later section we discuss mass-action kinetics, where the rate of progress of reactions depends on molar concentrations either in the gas phase or on surface sites and activities in the bulk phases. However, for the purposes of formulating and solving the conservation equations that describe physical situations, it is often more natural to use gas-phase mass fractions and surface site fractions as

dependent variables. Therefore, it is important to establish the rules for converting between the different ways to describe the composition of the gas and the surface.

For the gas-phase species the molar concentrations $[X_k]$ (in moles/cm^3) are written as

$$[X_k] = Y_k \rho / W_k, \qquad \left(k = K_g^f, ..., K_g^l\right) \tag{7}$$

where the Y_k are the mass fractions, ρ is the gas-phase mass density, and the W_k are the molecular weights.

On the surface sites we can describe the composition in terms of an array of surface species site fractions Z_k. This array is of length K_s. It is composed of N_s subunits of the site fractions of each of the species on a given site (phase) n. The site fractions on each site are normalized:

$$\sum_{k=K_s^f(n)}^{K_s^l(n)} Z_k(n) = 1. \qquad \left(n = N_s^f, ..., N_s^l\right) \tag{8}$$

The sum in Eq. (8) runs from the first species in phase *n* to the last species in phase *n*. The surface molar concentration of a species is then

$$[X_k] = Z_k(n)\Gamma_n / \sigma_k(n), \tag{9}$$

where Γ_n is the density of sites of phase *n* (in moles/cm^2) and $\sigma_k(n)$ is the number of sites that each species k occupies. For the sake of parallelism, we adopt the nomenclature for bulk species:

$$[X_k] = a_k. \qquad \left(k = K_b^f, ... K_b^l\right) \tag{10}$$

It is almost never a good approximation that bulk species form an ideal solution. Therefore, the concept of an activity (and the standard state to which it refers) must be introduced at the outset. In the limiting case of an ideal solution, the activity of a species is equal to its mole fraction.

SURFACE CHEMKIN-III takes the approach that the activity, a_k, of bulk species k is used in all chemical rate expressions. Moreover, the relationship between bulk mole fraction and the bulk activities is not explicitly evaluated by SURFACE CHEMKIN-III. Instead, it is up to the application code to specify the relationship between the two.

Surface Site Nonconservation

It is possible that a given surface reaction (or reactions) will not conserve the number of surface sites. In that case the density of sites Γ_n is not necessarily a constant. Therefore, one must take care in using an equality such as Eq. (9) when relating a site fraction and a surface molar concentration, that is, to ensure that the current (correct) value of $\Gamma_n(t)$ is used. It may be necessary to add equations to calculate the current value of the total site concentration of each surface phase. Because surface site nonconservation is an issue that can alter the basic governing equations of the system, we require that one acknowledge its use by adding a keyword on the REACTION line (discussed later). It is up to the user's application code to ensure that the current site concentrations are correct. Subroutines that return an array of species production rates also return an array of surface phase production rates, which would all be zero if sites are conserved in every elementary reaction.

Species Temperature Array

In many modeling applications, at each point in space there is a single (scalar) thermodynamic temperature. However, in models for multi-fluid plasma systems (for example) one might solve a separate energy equation for each gas-phase species or for groups of species. Subroutines in the CHEMKIN-III and SURFACE CHEMKIN-III Libraries consider temperature to be an array. The number of entries in the temperature array can be any number between 1 and the total number of gas-phase species. The example below illustrates how the temperature array may be defined and used in an application code.

```
      DIMENSION T(3), HML(KKTOT), KTFL(KKGAS), KION(KKGAS), ISKWRK(*), RSKWRK(*)
C      MAKE THE DEFAULT TEMPERATURE FOR ALL GAS-PHASE SPECIES TEMPERATURE NUMBER 1
      DO 100 K = 1, KKGAS
         KTFL(K) = 1
 100  CONTINUE
C     GET THE SPECIES NUMBER OF THE ELECTRON, THE NUMBER OF POSITIVE IONS
C     IN THE MECHANISM, AND THEIR SPECIES NUMBERS
      CALL SKKION (ISKWRK, KELECT, KKION, KION)
C     MAKE THE ELECTRON'S TEMPERATURE NUMBER 2
      IF (KELECT.NE.0) KTFL(KELECT) = 2
C     MAKE THE TEMPERATURE FOR ALL IONS NUMBER 3
      DO 200 K = 1, KKION
         KTFL(KION(K)) = 3
 200  CONTINUE
C     PUT THESE TEMPERATURE POINTERS INTO THE SURFACE CHEMKIN-III WORK SPACE
      CALL SKKTFL (ISKWRK, KTFL)
C     GET ARRAY OF SPECIES ENTHALPIES
      CALL SKHML (T, ISKWRK, RSKWRK, HML)
```

The array KTFL tells SURFACE CHEMKIN-III which entry in the temperature array to use for each gas-phase each species; the Library **always** uses temperature number 1 for surface or bulk-phase species. In this example, the default for (neutral) gas-phase species is to use temperature number 1 in

the temperature array. A separate energy equation may have been solved for the electron, and so for that species the example specifies that temperature number 2 in the temperature array is to be used. The energies of all of the ions may have been solved as a group by some other equation, and the example forces SURFACE CHEMKIN-III to use the third temperature in the temperature array for each ionic species. The fcall to SKKTFL tells the Library how to associate each species with the appropriate entry in the temperature array via the array KTFL.

The default in SURFACE CHEMKIN-III is a single thermodynamic temperature. If this is the case, an application code does not have to do anything with the KTFL array, and its entries are automatically set to 1; loop 100 in the above example is not strictly needed, but was included for clarity. It is up to the application code to decide whether to treat the temperature as an array or not. The call list for SKHML above, for example, looks just the same whether T is a scalar or an array. Thus, the form of the subroutine Library call lists (at least as far as temperature is concerned) is generally backwards compatible with previous versions of SURFACE CHEMKIN.

Standard-State Thermodynamic Properties

SURFACE CHEMKIN-III presumes that the standard-state thermodynamic properties for all species (regardless of phase) are given in terms of polynomial fits to the specific heats at constant pressure:

$$\frac{C_{p_k}^o}{R} = \sum_{m=1}^{M} a_{mk} T^{(m-1)}. \tag{11}$$

For the gas-phase species the superscript *o* refers to the standard state of an ideal gas at 1 atm. For perfect gases that we consider, however, the heat capacities are independent of pressure; the actual values equal the standard-state values.

For surface species the standard state of species k refers to the case of a chemical potential for a surface of pure species *k* (i.e., $Z_k \rightarrow 1$) with a fixed standard state site density Γ_n^o. Moreover, a perfect solution (i.e., non interacting) is assumed for the surface phase, which is independent of the system pressure. Under these assumptions the chemical potential for surface species k on surface site n may be written as

$$\mu_k(T,P,Z) = \mu_k^o(T) + RT \ln\left(\Gamma_n Z_k / \Gamma_n^o\right) \tag{12}$$

The standard state assumed by SURFACE CHEMKIN-III for bulk-phase species is discussed in the previous chapter.

Other thermodynamic properties are given in terms of integrals of the specific heats. First, the standard-state enthalpy is given by

$$H_k^o = \int_0^T C_{p_k}^o \, dT, \tag{13}$$

so that

$$\frac{H_k^o}{RT} = \sum_{m=1}^{M} \frac{a_{mk} T^{(m-1)}}{m} + \frac{a_{M+1,k}}{T}, \tag{14}$$

where the constant of integration $a_{M+1,k} R$ is the standard heat of formation at 0 K. Normally, however, this constant is evaluated from knowledge of the standard heat of formation at 298 K since the polynomial representations are usually not valid down to 0 K.

The standard-state entropy is written as

$$S_k^o = \int_0^T \frac{C_{p_k}^o}{T} dT \tag{15}$$

so that

$$\frac{S_k^o}{R} = a_{1k} \ln T + \sum_{m=2}^{M} \frac{a_{mk} T^{(m-1)}}{(m-1)} + a_{M+2,k}, \tag{16}$$

where the constant of integration $a_{M+2,k} R$ is evaluated from knowledge of the standard-state entropy at 298 K.

The above equations are stated for an arbitrary-order (Mth order) polynomial, but SURFACE CHEMKIN-III is designed to work with thermodynamic data in the form used in the NASA chemical

equilibrium code.[8] In this case, seven coefficients are needed for each of two temperature ranges.* These fits take the following form:

$$\frac{C_{p_k}^o}{R} = a_{1k} + a_{2k}T + a_{3k}T^2 + a_{4k}T^3 + a_{5k}T^4 \qquad (17)$$

[SKCPOR]

$$\frac{H_k^o}{RT} = a_{1k} + \frac{a_{2k}}{2}T + \frac{a_{3k}}{3}T^2 + \frac{a_{4k}}{4}T^3 + \frac{a_{5k}}{5}T^4 + \frac{a_{6k}}{T} \qquad (18)$$

[SKHORT]

$$\frac{S_k^o}{R} = a_{1k}\ln T + a_{2k}T + \frac{a_{3k}}{2}T^2 + \frac{a_{4k}}{3}T^3 + \frac{a_{5k}}{4}T^4 + a_{7k} \qquad (19)$$

[SKSOR]

Other thermodynamic properties are easily given in terms of C_p^o, H^o, S^o. The internal energy U is given as

$$U_k^o = H_k^o - RT, \qquad (20)$$

[SKUML]

the standard-state Gibbs free energy G^o is written as

$$G_k^o = H_k^o - TS_k^o, \qquad (21)$$

[SKGML]

and the standard-state Helmholtz free energy A^o is defined to be

$$A_k^o = U_k^o - TS_k^o. \qquad (22)$$

[SKAML]

For a perfect gas, the standard-state specific heats, enthalpies, and internal energies are also the actual values. Therefore, we drop the superscript *o* on those quantities.

* The SURFACE CHEMKIN-III Interpreter can be modified for additional temperature ranges, which would then require format changes to the thermodynamic data.

Often, specific thermodynamic properties are needed in mass units (per gram) rather than in molar units (per mole). The conversion is made by dividing the property in molar units by the molecular weight. The specific properties are thus given as

$$c_{p_k} = \frac{C_{p_k}}{W_k} \qquad (23)$$
[SKCPMS]

$$h_k = \frac{H_k}{W_k} \qquad (24)$$
[SKHMS]

$$s_k^o = \frac{S_k^o}{W_k} \qquad (25)$$
[SKSMS]

$$u_k = \frac{U_k}{W_k} \qquad (26)$$
[SKUMS]

$$g_k^o = \frac{G_k^o}{W_k} \qquad (27)$$
[SKGMS]

$$a_k^o = \frac{A_k^o}{W_k}. \qquad (28)$$
[SKAMS]

In addition to pure species properties, it is sometimes desirable to know mean properties for a mixture. The CHEMKIN-III[1] user's manual discusses this topic for gas-phase mixtures, and CHEMKIN-III provides subroutines to return mixture-average properties. At present, however, SURFACE CHEMKIN-III does not provide subroutines to return mixture-averaged properties for surface- or bulk-phase species. Thus, knowing the pure-species properties, the user must compute any averaged properties required in an application.

Chemical Reaction Rate Expressions

The *I* reversible (or irreversible) surface reactions involve K chemical species and can be represented in the general form

$$\sum_{k=1}^{K} \upsilon'_{ki} \chi_k \Leftrightarrow \sum_{k=1}^{K} \upsilon''_{ki} \chi_k \qquad (i = 1, \ldots, I) \tag{29}$$

The stoichiometric coefficients for elementary reactions υ_{ki} are integers* and χ_k is the chemical symbol for the kth species. Usually, an elementary reaction involves only three or four species; hence the υ_{ki} matrix is quite sparse for a large set of reactions.

The net production rate $\dot{s}_k$ (in moles/cm^2/sec) for each of the K species (regardless of phase) is the sum of the rate of production for all reactions involving the kth species:

$$\dot{s}_k = \sum_{i=1}^{I} \upsilon_{ki} q_i \qquad (k = 1, \ldots, K), \tag{30}$$

[SKRAT]

where

$$\upsilon_{ki} = \left(\upsilon''_{ki} - \upsilon'_{ki}\right). \tag{31}$$

[SKNU]

The rate-of-progress variable q_i for the *i*th reaction is given by the difference of the forward rates and the reverse rates:

$$q_i = k_{f_i} \prod_{k=1}^{K} [X_k]^{\upsilon'_{ki}} - k_{r_i} \prod_{k=1}^{K} [X_k]^{\upsilon''_{ki}}. \tag{32}$$

[SKROP]

It is not a requirement that the number of sites of type *n* balance in a given reaction. The production rate $\dot{\Gamma}_n$ (in moles/cm^2/sec) for each surface phase is

* Global reactions are sometimes stated with non-integer stoichiometric coefficients. This version of SURFACE CHEMKIN-III can accommodate non-integer stoichiometric coefficients.

$$\dot{\Gamma}_n = \sum_{i=1}^{I} \Delta\sigma(n,i) q_i, \qquad \left(n = N_s^f, ..., N_s^l\right) \tag{33}$$

where

$$\Delta\sigma(n,i) = \sum_{k=K_s^f(n)}^{K_s^l(n)} \upsilon_{ki} \sigma_k(n). \tag{34}$$

The term $\Delta\sigma(n,i)$ is the net change in number of surface sites of type *n* for surface reaction *i*. As discussed above, the form of the concentrations $\left[X_k\right]$ depends upon whether species k is in the gas phase, on the surface, or in the bulk. Furthermore, the units of the rate constants will depend on the reactants and products in a particular reaction.

The forward rate constants k_{f_i} for the *I* reactions are (by default) assumed to have the following Arrhenius temperature dependence:

$$k_{f_i} = A_i T^{\beta_i} \exp\left(\frac{-E_i}{R_c T}\right) \tag{35}$$

[SKABE, SKRAEX]

where the pre-exponential factor A_i, the temperature exponent β_i, and the activation energy E_i are specified.* These three parameters are required input to the SURFACE CHEMKIN-III package for each reaction. There are a number of ways in which the rate expression for a reaction can be altered, which are summarized as separate sections in this chapter.

For reversible reactions, the reverse rate constants k_{r_i} are related to the forward rate constants through the equilibrium constants as

$$k_{r_i} = \frac{k_{f_i}}{K_{c_i}} \tag{36}$$

---

* Two gas constants, *R* and R_C are used throughout this report and the SURFACE CHEMKIN-III code. R_C is used only in conjunction with the activation energy E_i and has compatible units. The reason for the duality is that many users would rather use different units (say calories/mole) for the activation energies even though other units (say cgs or SI) are used elsewhere.

(The user can over-ride the use of Eq. (36) by explicitly declaring Arrhenius coefficients for the reverse reaction in the Interpreter input via the auxiliary keyword REV, explained in Chapter V. An application code can call Library routine SKIREV to find out if reverse coefficients were input for a given reaction and their values.)

Although K_{c_i} is given in concentration units, the equilibrium constants are more easily determined from the thermodynamic properties in pressure units, K_{p_i} they are related by

$$K_{c_i} = \left(\frac{P_{\text{atm}}}{\text{RT}}\right)^{\sum_{k=1}^{K_g} \upsilon_{ki}} \prod_{n=N_s^f}^{N_s^l} \left(\Gamma_n^o\right)^{\sum_{k=K_s^f(n)}^{K_s^l(n)} \upsilon_{ki}} \prod_{k=K_s^f(n)}^{K_s^l(n)} \sigma_k^{-\upsilon_{ki}} K_{p_i} \qquad \text{(37)}$$

[SKEQ]

where P_{atm} denotes a pressure of 1 atm, and Γ_n^o is the standard-state surface site density of site type *n*. The sum in the first exponent runs only over the gas-phase species, and the sum in the second exponent runs only over surface species in surface phase *n*. The equilibrium constant K_{p_i} is obtained from the standard-state Gibbs free energy of reaction,

$$K_{p_i} = \exp\left(\frac{\Delta S_i^o}{R} - \frac{\Delta H_i^o}{RT}\right). \qquad \text{(38)}$$

The Δ refers to the change that occurs in passing completely from reactants to products in the *i*th reaction. More specifically,

$$\frac{\Delta S_i^o}{R} = \sum_{k=1}^{K} \upsilon_{ki} \frac{S_k^o}{R} \qquad \text{(39)}$$

$$\frac{\Delta H_i^o}{RT} = \sum_{k=1}^{K} \upsilon_{ki} \frac{H_k^o}{RT} \qquad \text{(40)}$$

Non-Integer Stoichiometric Coefficients

Previous versions of CHEMKIN and SURFACE CHEMKIN allowed only integer stoichiometric coefficients. This was based upon the reasonable assumption that kinetic mechanisms would deal with elementary chemical reactions, for which it makes little sense to talk about a fraction of a molecule participating as a product or reactant. However, in many real-world applications the elementary reactions are not known. Instead, the kinetics may only be summarized in terms of global expressions. In response to user requests, CHEMKIN-III and SURFACE CHEMKIN-III allow use of non-integer stoichiometric coefficients. Examples of reactions with such non-integer coefficients are:

$$\mathrm{H(s)} \leftrightarrow 0.5\mathrm{H}_2(g) \tag{41}$$

$$\mathrm{C_2H_6} \leftrightarrow 0.72\mathrm{C_2H_4} + 0.56\mathrm{H} + 0.28\mathrm{CH_4} + 0.28\mathrm{CH_2}. \tag{42}$$

The rate-of-progress of a reaction is, by default, still evaluated via Eq. (32), with the coefficients υ'_{ki} and υ''_{ki} definded as real numbers instead of integers. The CHEMKIN-III and SURFACE CHEMKIN-III Interpreters automatically allow real coefficients for reactions without requiring any special flags or keywords. An application code can call subroutine SKIRNU to find out which reactions were declared to the Interpreter with real coefficients, and get arrays of the coefficients.

Arbitrary Reaction Order

As just stated, by default the rate-of-progress of a reaction is evaluated by Eq. (32), which uses the concentration of each reactant or product species raised to the power of its stoichiometric coefficient. Thus, the rate-of-progress of a reaction that includes species A with a coefficient of 2 will be second-order with respect to the concentration of A. Equation (32) would always be valid when mass-action kinetics are obeyed, and the mechanism is written in terms of elementary reactions.

However, often in real-world applications the elementary kinetics are not known. In some cases, an experimental measurement finds that the rate of reaction is proportional to the concentration of a species raised to a some arbitrary power (different from its stoichiometric coefficient). CHEMKIN-III and SURFACE CHEMKIN-III allow the user to declare that the rate-of-progress of a reaction is proportional to the concentration of any species (regardless of whether that species even appears as a reactant or a product in the reaction) raised to any specified power. To modify the reaction order for the reaction in the forward or reverse direction, the user must declare the FORD or RORD auxiliary

keywords, respectively, in the Interpreter input file. (These keywords are discussed in Chapter V.) An application code can call subroutine SKIORD to find out which reactions were declared to the Interpreter with modified reaction orders, and get arrays of the species numbers and associated orders.

When the reaction-order-dependence of reaction i is changed via the FORD or RORD keywords, the rate-of-progress variable q_i for the reaction is evaluated by :

$$q_i = k_{f_i} \prod_{k=1}^{K} [X_k]^{F_{ki}} - k_{r_i} \prod_{k=1}^{K} [X_k]^{R_{ki}}, \tag{43}$$

where F_{ki} is the reaction order specified through the FORD keyword and R_{ki} is the reaction order specified through the RORD keyword for species k. The default for species participating in reaction i is the normal mass-action kinetics values:

$$F_{ki} = \upsilon'_{ki} \tag{44}$$

$$R_{ki} = \upsilon''_{ki} \tag{45}$$

if an order-change parameter is not given for species k.

The user is advised to exercise caution when specifying a change of reaction order. Such a change may produce unexpected and unphysical results in a kinetic simulation. The user should also consider the kinetics of the reverse reaction when changing reaction-orders for the forward reaction. For example, such a reaction may no longer satisfy microscopic reversibility. At equilibrium, elementary kinetics ensure that

$$k_{r_i} / k_{f_i} = \prod_{k=1}^{K} [X_k]^{\upsilon'_{ki}} / \prod_{k=1}^{K} [X_k]^{\upsilon''_{ki}} = \prod_{k=1}^{K} [X_k]^{\upsilon'_{ki} - \upsilon''_{ki}}. \tag{46}$$

A reaction for which one has specified a change in reaction order will not have the proper equilibrium behavior unless

$$F_{ki} - R_{ki} = \upsilon'_{ki} - \upsilon''_{ki}, \qquad (k = 1, \ldots, K). \tag{47}$$

The user specifying F_{ki} may also wish to adjust R_{ki} such that Eq. (47) is satisfied; SURFACE CHEMKIN-III does not do this automatically. Another alternative would be to simply specify that the reaction is irreversible, in which case the details of the reverse reaction become irrelevant.

Surface-Coverage Modification of Rate Expression

In some cases there are experimental data that indicate the Arrhenius expression for the rate constant, Eq. (35), is modified by the coverage (concentration) of some surface species. SURFACE CHEMKIN-III allows optional coverage parameters to be specified for species *k* and reaction *i*. through use of the auxiliary keyword COV, described later. In this case, the rate constant for the forward reaction is modified as

$$k_{f_i} = A_i T^{\beta_i} \exp\left(\frac{-E_i}{RT}\right) \prod_{k=K_s^f\left(N_s^f\right)}^{K_s^l\left(N_s^l\right)} 10^{\eta_{ki}\left[Z_k(n)\right]} \left[Z_k(n)\right]^{\mu_{ki}} \exp\left(\frac{-\varepsilon_{ki}\left[Z_k(n)\right]}{RT}\right), \tag{48}$$

where the three coverage parameters are η_{ki}, μ_{ki}, and ε_{ki} for species *k* and reaction *i*. The product in Eq. (38) runs over only those surface species that are specified as contributing to the coverage modification. Note that the surface site fractions appear in Eq. (48) rather than molar concentrations $\left[X_k\right]$ (moles/cm^2) for surface species. The term associated with μ_{ki} now makes it possible for the rate of progress of a reaction to be proportional to any arbitrary power of a surface species concentration. Also, using this modified expression for k_{f_i}, the net pre-exponential factor may be a function of coverage

$$\log_{10} A = \log_{10} A_i + \sum_{k=K_s^f\left(N_s^f\right)}^{K_s^l\left(N_s^l\right)} \eta_{ki}\left[Z_k(n)\right], \tag{49}$$

and the activation energy is a function of the coverage

$$E = E_i + \sum_{k=K_s^f\left(N_s^f\right)}^{K_s^l\left(N_s^l\right)} \varepsilon_{ki}\left[Z_k(n)\right]. \tag{50}$$

For reactions with optional coverage dependence, the rate or progress is calculated employing Eq. (32), with the forward rate coefficient from Eq. (48). The reverse rate constant is calculated via Eq. (36).

If the form of Eq. (48) is not flexible enough to describe a certain coverage behavior, one can repeat the same reaction several times with different values for the coverage parameters such that the sum of the rate constants approximates the desired form.

Ion-Energy Dependent Rate Expression

In many examples of materials processing, ions interact with surfaces to alter the morphology, sputter material, or enhance heterogeneous chemical reactions. Ions are often accelerated through a plasma sheath near grounded or electrically biased materials. In this way, the directed energy of ions encountering a surface may be significantly greater than the ion temperature in the plasma gas. SURFACE CHEMKIN-III therefore makes the provision for a reaction rate constant to depend upon the energy of a positive ionic reactant species, E_{ion}. The functional form allowed is as follows

$$k_i\left(E_{\text{ion}}\right) = k_i(\text{thermal}) \cdot \max\left[0, \left(E_{\text{ion}}^{f_i} - E_{\text{ion},0}^{f_i}\right)^{g_i}\right]. \tag{51}$$

The reaction rate depends upon a threshold energy, $E_{\text{ion},0}$, and the energy expressions can be raised to a specified power in two different ways through the use of the parameters f_i and g_i. Ion-energy dependent reactions are declared in the Interpreter input via the auxiliary keyword ENRGDEP. An application code can find out which reactions were declared as ion-energy-dependent reactions and get an array of the parameters by a call to SKIENR. Because the subroutines that evaluate rate constants in SURFACE CHEMKIN-III take temperature as an argument, and not species energy, subroutine SKRPAR must be called to input an array of ion energies, ENRGI before the rate constant routine is called. Use of the ENRGDEP keyword is only allowed for irreversible reactions.

Sticking Coefficients

For some simple surface reaction mechanisms we have found it convenient to specify the surface reaction rate constant in terms of a "sticking coefficient" (probability).* For example, one might have a measurement or intuition about the probability that a certain process takes place when a given collision occurs. For consistency in expressing each surface reaction in terms of a rate constant, we

---

* CAUTION: Because γ_i is defined as a probability, it must lie between 0 and 1 to make physical sense. Therefore, SURFACE CHEMKIN-III checks the value of γ_i, and an unphysical sticking coefficient greater than 1 is changed to the value 1. Some earlier versions of SURFACE CHEMKIN did not truncate the values at 1.

provide a conversion between this sticking coefficient form and the usual rate expression. We allow the sticking coefficient form only for the simple case of a surface reaction in which there is exactly one gas-phase reactant species, although there can be any number of surface species specified as reactants.

The sticking coefficients functional form is taken to be

$$\gamma_i = \min\left[1, a_i T^{b_i} e^{-c_i / R_c T}\right]. \tag{52}$$

In this case, a_i and b_i are unitless and c_i has units compatible with R_c. SURFACE CHEMKIN-III also allows for surface-coverage modification of a sticking coefficient, analogous to Eq. (48).

We give three successively complex examples of using sticking coefficients. First, to specify that SiH_2(g) reacts with probability γ_i upon each collision with the surface, one could write the reaction

$$SiH_2(g) \rightarrow Si(b) + H_2. \tag{53}$$

In this example, we have not explicitly included the surface in writing Eq. (53).

A somewhat more detailed way of using the sticking-coefficient specification would be to say that SiH_2(g) reacts with probability γ_i upon each collision with a bare surface silicon atom, Si(s):

$$SiH_2(g) + Si(s) \rightarrow Si(s) + Si(b) + H_2. \tag{54}$$

If the surface site fraction of Si(s) were unity, then a fraction γ_i of the collisions of SiH_2 with the surface would result in a reaction. However, for Si(s) coverages less than 1, the reaction rate decreases in proportion with the coverage of Si(s).

In a third (contrived) example, suppose there is a probability γ_i for a reaction to occur when SiH_2 collides with both a Si(s) and a C(s) reaction such as

$$SiH_2(g) + Si(s) + C(s) \rightarrow Si(b) + SiH(s) + CH(s) \tag{55}$$

The rate of this reaction would be proportional to both the coverage of Si(s) and C(s).

To convert rate constants given as sticking coefficients γ_i to the usual mass-action kinetic rate constants there is the relation*

$$k_{f_i} = \frac{\gamma_i}{\left(\Gamma_{tot}\right)^m}\sqrt{\frac{RT}{2\pi W_k}} \tag{56}$$

where R is the universal gas constant, W_k is the molecular weight of the gas-phase species, Γ_{tot} is the total surface site concentration summed over all surface phases (number of moles of surface sites per unit area), and m is the sum of all the stoichiometric coefficients of reactants that are surface species. The term involving Γ_{tot} raised to the m power is needed to convert from the unitless sticking coefficient form to units appropriate for a rate constant, and the term in the square root accounts for the gas/surface collision frequency. In the third example given above, Eq. (55), the value of m is 2, because there are two surface species appearing as reactants, i.e., Si(s) and C(s).

Implicit in the sticking coefficient description just presented is an assumption that the sticking coefficient is relatively small, i.e., much less than one. In this case the molecular motion in the vicinity of the solid surface is random and the collision frequency of gas-phase species with the surface is not affected by the surface itself. However, when the sticking coefficient is large, i.e., close to one, then the velocity distribution becomes skewed. Species whose random motion carries them close to the surface have a high probability of staying there, which causes a non-Maxwellian velocity distribution that, in turn, alters the net species flux near the surface. Motz and Wise[10] analyzed this situation and provided a correction factor that modified Eq. (56) as

$$k_{f_i} = \left(\frac{\gamma_i}{1-\gamma_i/2}\right)\frac{1}{\left(\Gamma_{tot}\right)^m}\sqrt{\frac{RT}{2\pi W_k}}. \tag{57}$$

Goodwin and Gavillet[11] have incorporated this effect in their analysis of chemical vapor deposition of diamond films.

---

* Versions of the software before SURFACE CHEMKIN-III always applied Eq. (57). Later versions allow optional use of Eq. (56) to relate the sticking coefficient to rate constants through use of the keyword MWOFF on the REACTION line (described later).

The rate of progress is calculated using Eq. (32), as usual. The sticking coefficient specification is only allowed for the forward reaction. If the reaction is written as reversible, the reverse reaction rate constant would be calculated from Eqs. (57) and (36) using microscopic reversibility.

Bohm Rate Expression for Ionic Reactions

The rate constant for a reaction involving a positive ion can be modified by applying a Bohm velocity correction, as follows

$$k_i(\text{Bohm}) = a_i T^{b_i} e^{-c_i / R_c T} \sqrt{\frac{kT_e}{W_{\text{ion}}}}. \tag{58}$$

In the expression in Eq. (58), the unitless pre-exponential, temperature exponent term, and activation energy correspond to the parameters in a sticking coefficient, explained above. However, the Bohm velocity expression (the term in the square root in Eq. (58)) is based on the electron temperature, instead of an equilibrium thermodynamic temperature.* The molecular weight in the last term is that of the positive ion. Bohm reactions can be declared through the Interpreter input via the auxiliary keyword BOHM. An application code can find out which reactions were declared as Bohm reactions by a call to SKIBHM. Use of the BOHM keyword is only allowed for irreversible reactions.

Ion-Enhanced Reaction Yield

In modeling plasma systems, one sometimes encounters reactions where the energy of the incident ion determines the number of surface species etched. Such surface reactions in SURFACE CHEMKIN-III can be modeled using a "yield enhancement" factor to account for the variable stoichiometry.

Imagine the case in which a positive ion, $I(g)$, hits a surface and knocks off a variable number (ψ) of surface species, $S(s)$. For each surface species $S(s)$ destroyed, the example reaction produces two gas-phase products, $P(g)$ and leaves behind some other surface species, $O(s)$; another gas species $Q(g)$, is produced by the reaction, but its stoichiometric coefficient is not dependent upon the number of surface species etched.

* Thus, the electron must be declared as a gas-phase species in the list of species names in the CHEMKIN-III interpreter input.

$$I(g) + \psi S(s) \rightarrow \psi 2P(g) + \psi O(s) + Q(g). \tag{59}$$

The coefficient ψ is essentially a variable stoichiometric coefficient which depends upon the energy of the positive ionic reactant.

A reaction written like Equation (59) is required to satisfy mass, charge, and elemental balance (as is every reaction in a SURFACE CHEMKIN-III mechanism). For this always to be the case, the "sub-reaction"

$$S(s) \rightarrow 2P(g) + O(s) \tag{60}$$

consisting of all of the species in the original reaction which are multiplied by the coefficient ψ must also satisfy mass, charge, and elemental balance. In addition, unless the NONCON keyword was declared on the REACTION line (described later), the sub-reaction must also conserve the number of surface sites.

An example of a reaction using the ion-enhanced yield option in the form accepted by the SURFACE CHEMKIN-III Interpreter is

```
E + CL+ + #SICL3(S) + #SI(B) + SICL(S)                    (61)
   => SICL2(S) + #SICL2 + #SICL(S).
```

The specieal character # identifies the energy-dependent multiplicative factor for the stoichiometric coefficient. Notice that the sub-reaction consisting of every species preceded by the # sign balances mass, elements, charge, and number of surface sites. The "yield" of this reaction (per incident `CL+` ion) depends upon the energy of the ion, Eq.(62) below.

We allow the following functional form for the yield enhancement

$$\psi\left(E_{\text{ion}}\right) = h_{\text{yield}} \max\left[0, \left(E_{\text{ion}}^{t_i} - E_{\text{yield},0}^{t_i}\right)^{u_i}\right]. \tag{62}$$

The ion-enhanced yield can depend upon a threshold energy, $E_{\text{yield},0}$, and the energy expressions can be raised to a specified power in two different ways through the use of the parameters t_i and u_i. Ion-enhanced-yield reactions can be declared through the Interpreter input via the auxiliary keyword YIELD. An application code can find out which reactions were declared as ion-enhanced-yield reactions

and get an array of the parameters via a call to SKIYLD. Because the subroutines that evaluate rate constants in SURFACE CHEMKIN-III take temperature as an argument, and not species energy, subroutine SKRPAR must be called to input an array of ion energies, ENRGI, before the rate constant routine is called. Use of the YIELD keyword is only allowed for irreversible reactions.

Manipulation of Chemical Rate Sensitivity Coefficients

We have found sensitivity analysis to be a powerful tool in helping interpret the results of computational simulations. Sensitivity analysis is used to determine quantitatively the dependence of a solution on certain parameters that appear in a model's definition. The "raw" first-order sensitivity coefficient matrices $S_{li} = \partial\Phi_l / \partial\alpha_i$ report the partial derivatives of the dependent variable vector Φ (e.g., temperature, mass fractions, surface composition) with respect to a parameter vector α_i (e.g., reaction rate constants). Since there is much mathematical literature on sensitivity analysis and various methods to compute the sensitivity coefficients from the solution, we do not discuss the computation of S_{li} here.

However, given the sensitivity matrix it is possible to manipulate it further to obtain the sensitivities of species production rates with respect to the dependent variables:

$$\frac{d\dot{s}_k}{d\alpha_i} = \frac{\partial\dot{s}_k}{\partial\alpha_i} + \sum_l \frac{\partial\dot{s}_k}{\partial[X_l]} \frac{\partial[X_l]}{\partial\Phi_l} \frac{\partial\Phi_l}{\partial\alpha_l}, \tag{63}$$

where the components of Φ are the mass fractions, site fractions, site fractions, and activities for gas-phase, surface, and bulk species, respectively. The term $\partial[X_l]/\partial\Phi_l$ converts from concentration units to the units of Φ:

$$\frac{\partial[X_l]}{\partial\Phi_l} = \begin{cases} \left(\frac{P}{RT}\right)\left\{\frac{\overline{W}}{W_l} - \frac{Y_l\overline{W}^2}{W_l^2}\right\} & K_g^f \le l \le L_g^l \\ \Gamma_n/\sigma_k(n), & K_s^f\left(N_s^f\right) \le l \le K_s^l\left(N_s^l\right) \\ 1, & K_b^f\left(N_b^f\right) \le l \le K_b^l\left(N_b^l\right) \end{cases} \tag{64}$$

We have included two subroutines in the Surface Library to facilitate calculation of these terms. The first gives the partial derivative of the production rate of species *k* with respect to the pre-exponential constant of the Arrhenius expression for surface reaction i^* :

$$\frac{\partial \dot{s}_k}{\partial \alpha_i} = \upsilon_{ki} q_i / \alpha_i . \tag{65}$$

[SKDRDA]

The production rate of species *k* due to reaction *i* is

$$\dot{s}_{ki} = \upsilon_{ki} q_i . \tag{66}$$

Therefore, the dependence of $\dot{s}_{ki}$ upon the concentration of some species *l* is

$$\begin{aligned} \frac{\partial \dot{s}_{ki}}{\partial [X_l]} = \upsilon_{ki} q_i^f \left[\frac{\upsilon'_{li}}{[X_l]} + \left(\frac{\sigma_k(n)}{\Gamma_n} \right) \left\{ \eta_{li} \ln(10) + \frac{\mu_{li}}{[X_l]} - \frac{\varepsilon_{li}}{R_c T} \right\} \right] \\ - \upsilon_{ki} q_i^r \left[\frac{\upsilon''_{li}}{[X_l]} + \left(\frac{\sigma_k(n)}{\Gamma_n} \right) \left\{ \eta_{li} \ln(10) + \frac{\mu_{li}}{[X_l]} - \frac{\varepsilon_{li}}{R_c T} \right\} \right]. \end{aligned} \tag{67}$$

The terms inside the curled braces will only be present if species *l* modifies the rate of reaction *i* through coverage parameters, as in Eq. (48). The partial of the production rate of species *k* due to all reactions with respect to the concentration of species *l* is then

$$\frac{\partial \dot{s}_k}{\partial [X_l]} = \sum_{i=1}^{I} \frac{\partial \dot{s}_{ki}}{\partial [X_l]} . \tag{68}$$

[SKDRDC]

These terms can all be combined to calculate the desired $\partial \dot{s}_k / \partial \alpha_i$ in the user's application code.

---

* Note that subroutine SKDRDA calculates the derivative with respect to the Arrhenius pre-exponential if the reaction was originally stated in standard Arrhenius form, or with respect to the sticking coefficient pre-exponential if a sticking coefficient was used.

Flux-Matching Conditions at a Gas-Surface Interface

Heterogeneous reactions at a gas-surface interface affect the mass and energy balance at the interface, and thus have an important influence on the boundary conditions in a chemically reacting flow simulation. The convective and diffusive mass fluxes of gas-phase species at the surface are balanced by the production (or depletion) rates of gas-phase species by surface reactions. This relationship is

$$\bar{n} \cdot \left[\rho Y_k \left(\bar{V}_k + \bar{u} \right) \right] = -\dot{s}_k W_k \qquad \left(k = 1, \ldots, K_g \right) \tag{69}$$

where **n** is the unit inward-pointing normal vector to the surface, and the gas-phase diffusion velocities are related to the species and temperature gradients by

$$\bar{V}_k = \frac{1}{X_k \overline{W}} \sum_{j \neq k}^{K_g} W_j D_{kj} \nabla X_j - \frac{D_k^T}{\rho Y_k} \frac{\nabla T}{T}. \qquad \left(k = 1, \ldots, K_g \right) \tag{70}$$

Here the X_k are the gas-phase mole fractions, the Y_k are the gas-phase mass fractions, $\overline{W}$ is the mean molecular weight, D_{kj} is the ordinary multicomponent diffusion coefficient matrix, and the D_k^T are the thermal diffusion coefficients. (Both types of diffusion coefficients can be evaluated by calls to the Transport Package[2,3].) In nonreacting fluid mechanics the fluid velocity normal to a solid wall is zero. However, if there are chemical reactions at the wall, then the velocity can be nonzero. This so-called Stefan flow velocity occurs when there is a net mass flux between the surface and the gas. The induced Stefan velocity is given by

$$\bar{n} \cdot \bar{u} = -\frac{1}{\rho} \sum_{k=1}^{K_g} \dot{s}_k W_k. \tag{71}$$

The expression for the Stefan velocity is easily obtained from the interfacial mass balance (Eq. 69) by summing over all K_g species and noting that the mass fractions must sum to one, i.e.,

$$\sum_{k=1}^{K_g} Y_k = 1, \tag{72}$$

and that the sum of the diffusion fluxes must be zero, i.e.,

$$\sum_{k=1}^{K_g} Y_k V_k = 0. \tag{73}$$

Exothermicity (or endothermicity) of surface reactions contribute to the energy balance at an interface. Diffusive and convective fluxes in the gas phase are balanced by thermal radiative and chemical heat release at the surface. This balance is stated as

$$\bar{\mathrm{n}} \cdot \left[-\lambda \nabla \mathrm{T}\big|_{\mathrm{gas}} + \sum_{k=1}^{K_g} \rho Y_k \left(\bar{V}_k + \bar{\mathrm{u}} \right) h_k \right] = \sigma \varepsilon \left(T^4 - T_o^4 \right) + \sum_{k=K_s^f\left(N_s^f\right)}^{K_b^l\left(N_b^l\right)} \dot{s}_k W_k h_k - \bar{\mathrm{n}} \cdot \lambda_{\mathrm{bulk}} \nabla \mathrm{T}\big|_{\mathrm{bulk}}. \tag{74}$$

The summation on the right-hand side runs over all surface and bulk species. It is interesting to note that by substituting Eq. (69) into the flux term on the left-hand side, the energy balance can be written in a more compact form as

$$-\bar{\mathrm{n}} \cdot \lambda \nabla \mathrm{T}\big|_{\mathrm{gas}} = \sigma \varepsilon \left(T^4 - T_o^4 \right) + \sum_{k=1}^{K} \dot{s}_k W_k h_k - \bar{\mathrm{n}} \cdot \lambda_{\mathrm{bulk}} \nabla \mathrm{T}\big|_{\mathrm{bulk}}. \tag{75}$$

Now the reaction-rate summation on the left-hand side runs over all species, including the gas-phase species.

The SURFACE CHEMKIN-III package allows the user to specify mass densities ρ_k for the bulk species. A possible use for the densities would be to convert surface reaction rate of production of a bulk species (in moles / cm^2 / sec) into a growth rate G (in cm / sec). The needed relationship is

$$G = \sum_{k=K_b^f\left(N_b^f\right)}^{K_b^l\left(N_b^l\right)} \frac{\dot{s}_k W_k}{\rho_k}. \tag{76}$$

IV. THE MECHANICS OF USING SURFACE CHEMKIN-III

SURFACE CHEMKIN-III is one component of a large body of software designed to facilitate the computational modeling of chemical kinetics in flowing systems. An application program (for example a Chemical Vapor Deposition analysis code) can draw on any of three major software packages:

- CHEMKIN-III, which handles gas-phase equation-of-state, thermodynamic properties, and chemical kinetics;
- the Transport Package, which handles gas-phase molecular transport properties; and
- SURFACE CHEMKIN-III, which handles surface thermodynamics and chemical kinetics.

Each package consists of a symbolic preprocessor called an Interpreter, a database of either thermodynamic or transport properties, and a library of subroutines that can be called from the application code.

The software is highly structured and modular, which provides great flexibility in applying it to a wide variety of problems. However, this modularity also compels the user to manipulate a number of programs and files. The flow of information from the first input to the CHEMKIN-III Interpreter to the inclusion of a library subroutine in an application program is shown in Fig. 5.

We presume that all problems involving surface chemistry also involve one or more gas-phase species above the surface. Therefore, the first step in any problem involving surface chemistry is to run the CHEMKIN-III Interpreter, which reads the user's description of the gas-phase reaction mechanism. The CHEMKIN-III Interpreter also draws on a Thermodynamic Database containing polynomial fits to individual species specific heats, enthalpies, and entropies. In addition to printed output, the CHEMKIN-III Interpreter creates a Linking File containing all possible information regarding the particular gas-phase reaction mechanism. The Linking File is read by an initialization subroutine in the Gas-Phase Subroutine Library that makes the information available to all the other subroutines in the library.

The next step is to execute the SURFACE CHEMKIN-III Interpreter, which reads the user's symbolic description of the surface-reaction mechanism. Required thermodynamic data can come from the same Thermodynamic Database used by CHEMKIN-III or from a separate Thermodynamic Database compiled for surface species. Both Interpreters provide the capability to add to or override the

data in the database by user input in the reaction description. The SURFACE CHEMKIN-III Interpreter extracts all needed information about gas-phase species from the CHEMKIN-III Linking File. (Thus the CHEMKIN-III Interpreter must be executed before the SURFACE CHEMKIN-III Interpreter.) Like the CHEMKIN-III Interpreter, the SURFACE CHEMKIN-III Interpreter also provides a printed output and a Linking File.* Again, the Surface Linking File is read by an initialization subroutine in the Surface Subroutine Library that makes the surface-reaction mechanism information available to all other subroutines in the Library.

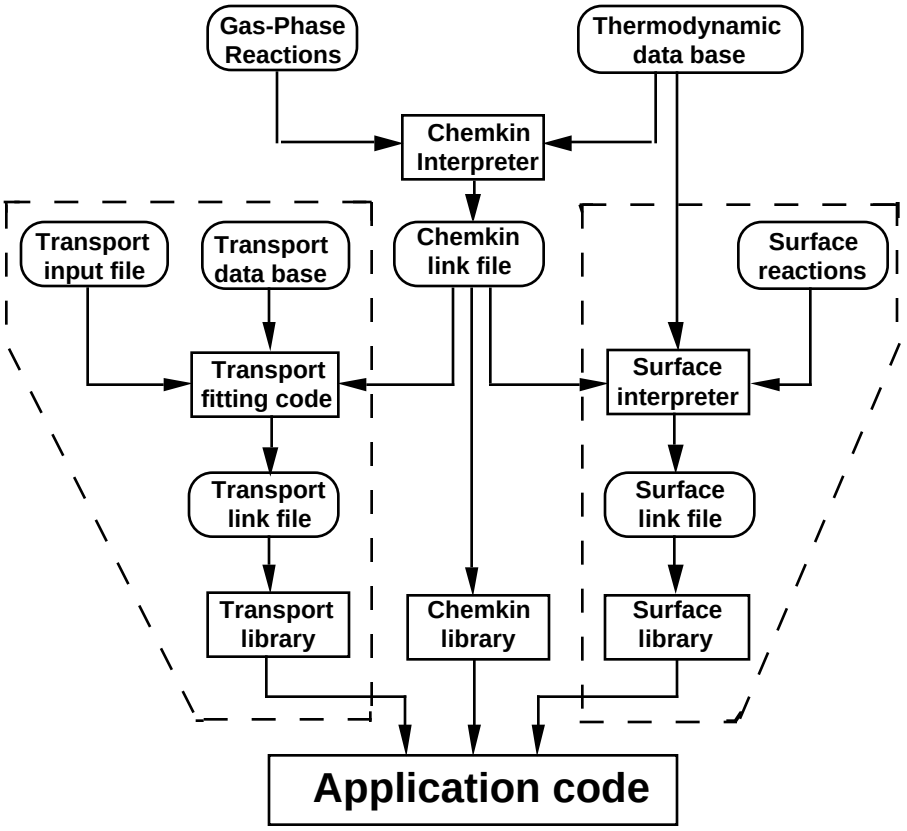


Figure 5. Relationships and Flow of Information between the CHEMKIN-III, Transport, and SURFACE CHEMKIN-III Packages, and a User's Application Program

* By default the linking files created by CHEMKIN-III and SURFACE CHEMKIN-III are binary files (named chem.bin and surf.bin, respectively); ASCII linking files (named chem.asc and surf.asc, respectively) are a new option, available by running the Change program[13] on each Interpreter source file.

A third software package, which handles gas-phase molecular transport, may or may not be needed in a particular application. If it is used, the Transport Property Fitting Code reads the CHEMKIN-III Linking File and identifies all the gas-phase species that are present in the gas-phase reaction mechanism. Then, drawing on a database of molecular parameters, it computes polynomial fits to the temperature-dependent pure-species viscosities, thermal conductivities, and binary diffusion coefficients. As with the other packages it provides a Linking File that is read by an initialization subroutine in the Transport Property Subroutine Library. Subroutines from this library may be called by the application code to return transport properties for individual species or for multicomponent gas mixtures.

It is clear from the foregoing description that using these software packages requires the interaction of many programs and the manipulation of many input and output files. Therefore, while the modular structure of the software provides a powerful capability to formulate a wide variety of applications, it also requires that users be reasonably familiar with their computers' operating systems. The functional equivalent of the sample Unix shell script shown in Fig. 6 is required on any computer system.

It may also be noted from the flow chart of Fig. 5 that the three software packages do not *solve* any problems---they simply make subroutines available to aid *formulation* of a problem. This structure provides maximum flexibility because the software does not need to be concerned with details of the limitless range of problems that a user may wish to pose and solve. At the same time, the burden is on the user to define the equations that describe his particular problem and to write an application code to solve these equations.

The schematic diagram in Fig. 7 shows how an applications code might interact with the low-level chemical kinetics and transport software packages. The boxes with the light borders indicate those blocks of software that are relegated to subroutine calls to one of the low-level packages, and the boxes with the heavy borders indicate those blocks of software that the user must write for a particular application. We show the problem-independent mathematical software as a box with a light border to indicate that many problems can solved with readily available, high-quality mathematical software. Certainly this is true for problems that can be formulated as systems of nonlinear algebraic equations or ordinary-differential-equation initial-value or boundary-value problems. However, for more complex problems involving systems of partial differential equations users may have to write their own mathematical software.

```
#!/bin/sh
# to execute:  sh sksamp.sh myrun &
sh 1> ${1}.log 2>&1 << ENDSH   # shell run output is "myrun.log"
set -x                         # echo all commands to stdout

#cd /scr/$LOGNAME              #go to user's scratch directory
#mkdir "${1}$$"                #make subdirectory /myrun##
#cd "${1}$$"                   #go to /myrun##

cat << EOF > makefile
FFLAGS = -g -mips4
LINK   = f77 -o
MACH = dmach.o

OBJS = sksample.o cklib.o sklib.o vode.o math.o
INPS = therm.dat chem.inp surf.inp sksamp.inp
OUTS = chem.bin chem.out surf.bin surf.out sksamp.out
EXES = chem.exe surf.exe sksamp.exe

chem.exe: ckinterp.o
          $(LINK) chem.exe ckinterp.o

surf.exe : skinterp.o cklib.o
           $(LINK) surf.exe skinterp.o cklib.o

sksamp.exe : $(OBJS)
           $(LINK) sksamp.exe $(OBJS)
EOF

touch makefile; make sksamp.exe
chem.exe < chem.inp > chem.out
surf.exe < surf.inp > surf.out
sksamp.exe < sksamp.inp > sksamp.out

ENDSH
```

Figure 6 Sample Unix command procedure, showing the steps required to run an application code using the CHEMKIN-III and SURFACE CHEMKIN-III packages

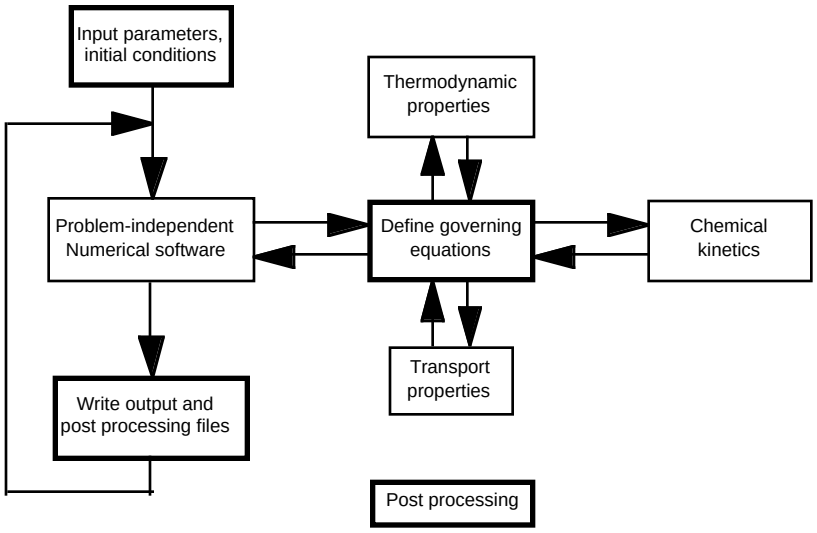


Figure 7 Schematic Representation of an "Ideal" Applications Program

V. USING THE SURFACE CHEMKIN-III INTERPRETER

The SURFACE CHEMKIN-III Interpreter is used to read a symbolic description of a surface-reaction mechanism (from the Fortran's standard input) and numerical information about a gas-phase mechanism (from the file chem.bin), and create a Linking File (surf.bin) of pertinent information about the surface mechanism and the species it involves. The information in the SURFACE CHEMKIN-III Linking File is subsequently accessed by various subroutines in the SURFACE CHEMKIN-III Library to provide information on thermodynamic properties and chemical production rates. Output from the interpreter is written to Fortran's standard output unit.

The Surface Interpreter input includes information on surface sites (phases), surface species, bulk phases, bulk species, thermodynamic data, and the reaction mechanism. Interpreter input information is given in 80-column format. An optional material name is read first, followed by site data, then by bulk data; optional thermodynamic data may follow, and optional reaction data are specified last. Such sets of information can be repeated for any number of different materials.* The thermodynamic data for the species may come from the interpreter input file and/or from a Thermodynamic Database (therm.dat). The syntax for the four types of input is described below.

With the exception of the thermodynamic data, all input is format free. For the thermodynamic data, we have chosen to use the same format as in the NASA Chemical Equilibrium code of Gordon and McBride[8] and CHEMKIN-III.[1]

Material Declaration

Entirely different surface reaction mechanisms (i.e., with different surface and bulk phases and species, and different surface reactions) can be specified in the same SURFACE CHEMKIN-III input file through the use of multiple materials. At the beginning of each separate portion of the input file corresponding to a given material, the user gives an input line with the keyword MATERIAL followed by an optional slash-delimited material name. If no name for the material is supplied, the interpreter substitutes a default name MATERIAL*n*, where *n* is the number of the material (e.g., MATERIAL2 for the second material given in an input file). An example of the usage of multiple materials is given at the end of Chapter VI of this manual.

---

* This is a new feature in SURFACE CHEMKIN-III.

Site Data

Surface-phase species exist on sites, and a site and its species must be identified on one or more lines of site data. The first line in a set of site data must start with the word SITE; an optional name may be associated with a site if it immediately follows SITE and is delimited by slashes(/). If no name for the site is supplied, the interpreter substitutes the default name SITE*n*, where *n* is the number of a site (e.g., SITE2 for the second site type listed). Following SITE and/or the site name, the word SDEN and a slash-delimited density (the standard-state site density for this site, in moles/cm^2) for the site is required. The species that can reside on the site type are declared by a list of species symbols (names) on the same line or on additional lines. The name of a site species must not duplicate the name of a gas-phase species or a bulk species, and must be unique among the species names listed *for that site*. However, the same site species name may appear in the list for *another* site. Thus, a user can refer to a surface species uniquely by specifying both the name of the site and the species name.

An optional slash-delimited site occupancy number may follow a species name, i.e., the number of individual sites that this species occupies. (For example, a large chemical species might cover two or more sites.) The default site occupancy for a surface species is 1. The sets of SITE data input can continue for as many site types as are needed.

Any set of up to sixteen upper- or lower-case characters can be used as a site name or species symbol.* In addition, each species must be composed of elements that have been identified in the CHEMKIN-III Interpreter and thus contained in the CHEMKIN-III Linking File (unit number LINKCK). One of the primary purposes of the site data is to define the order in which Fortran arrays of site species information are referenced in the Surface Subroutine Library.

Any line starting with or any portion of a line following an exclamation mark (!) is considered a comment and will be ignored. Blank lines are also ignored. Figure 8 shows sample site data. The rules for site data are summarized in Table I.

---

* Species symbols may not begin with a number, a plus sign (+), a pound sign (#), or an equality sign (=), have imbedded blanks, or include a slash (/). An ionic species may end with any number of plus or minus signs; an imbedded plus sign must be enclosed in parentheses.

```
SITE / PLANE /              SDEN/1.04E-9/                          !PLANAR SITE
       ASH(V)                                          ! FIRST SPECIES ON PLANE SITE
       ASH2(V)   ASH3(V)  H(S)  CH3(V)  AS(V)   AS2(V)/2/
       V                                                  ! EMPTY PLANAR SITE
          END                                        !AN END STATEMENT IS OPTIONAL
SITE / LEDGE /   SDEN/1.66E-10/                                 ! LEDGE SITE

       GACH(L)                                            !FIRST SPECIES ON LEDGE
       H(S)                                             !THIS IS A DUPLICATE NAME
       DMG(L)/2/                                 !THIS SPECIES OCCUPIES 2 LEDGE SITES
       L                                                    !EMPTY LEDGE SITE
SITE   SDEN/1.0E-10/ GA(S)                       !SITE WITH ONLY ONE SPECIES
                                                 !SITE NAME NOT INCLUDED
```

Figure 8 Sample Site Data

Table I. Summary of the Rules for Site Data

1. Site data must start with a line containing the word SITE, followed by an optional slash-delimited name (i.e., SITE / *name* /).
2. The standard-state site density is required as a slash-delimited number (in moles / cm^2) following the word SITE and / or the site name, and preceded by the word SDEN.
3. The site density is followed by one or more site species name declarations. Declaring a site with no site species is an error.
4. Site and species names are composed of up to sixteen upper- or lower-case character symbols. The names cannot begin with the characters +, =, #, or a number; an ionic species name may end with one or more + or - signs; an embedded plus sign must be enclosed in parentheses (+). Names cannot include a slash (/).
5. All species names within a given site should be unique; duplicate species names will be ignored and a warning issued. A species name may not duplicate a name of a gas-phase species, but may duplicate the name of a species on a different surface site.
6. A site name must not duplicate the name of any other phase (gas, surface site, or bulk phase).
7. Each surface species that subsequently appears in a surface reaction must have been declared in this section.
8. A site species name may appear anywhere on the line.
9. A site species may have a slash-delimited site occupancy (the number of sites that this species occupies on the surface) following the species name.
10. A species name declaration that begins on one line may not continue to the next line (i.e., do not break a species name into two lines).
11. There may be more than one set of SITE data.
12. All characters on a line following an exclamation mark are considered comments.
13. SITE data are not required.

Bulk Data

A set of bulk data may consist of one or more condensed-phase species. The first line in a set of bulk data must start with the word BULK and may be followed by an optional slash-delimited name for the bulk phase. If a name is not supplied for bulk phase *n*, then the name BULK*n* is supplied by the interpreter. Bulk species are declared by a list of unique species symbols (names) on the same line or on additional lines. An optional slash-delimited density (in g/cm^3) may follow a species name. If no density is supplied, the unphysical value of -1.0 is stored as a flag. The rules for bulk species symbols (names) are essentially the same as those for site species. Figure 9 shows sample bulk data. The rules for bulk data are summarized in Table II.

Table II. Summary of the Rules for Bulk Data

1. Bulk data must start with a line containing the word BULK, and may be followed by a slash-delimited name for the bulk phase (i.e., BULK/*name*/).
2. The BULK declaration and/or bulk name must be followed by one or more bulk species declarations. Declaring a bulk phase with no bulk species is an error.
3. Bulk and bulk species names are composed of up to sixteen upper- or lower-case character symbols. The names cannot begin with the +, =, #, or a number; an ionic species name may end with one or more + or - signs; an embedded plus sign must be enclosed in parentheses (+). Names cannot include a slash (/).
4. Bulk species names must be unique; duplicated species names will be ignored and a warning issued. A species name may not duplicate a name of a gas-phase or surface species, but may duplicate the name of a species in a different bulk phase.
5. A bulk phase name must not duplicate the name of any other phase (gas, surface site, or bulk phase).
6. Each bulk species that subsequently appears in a surface reaction must have been declared in this section.
7. A bulk species declaration may start anywhere on the line.
8. A bulk species name may be followed by an optional slash-delimited mass density (in g/cm^3).
9. A bulk species declaration that begins on one line may not continue to the next line (i.e., do not break species names into two lines).
10. There may be more than one set of BULK data.
11. All characters on a line following an exclamation mark are considered comments and are ignored.
12. BULK data are not required.

```
BULK / GA_RICH /  GA2AS(1)/3.0/  GA3AS(1)/3.0/     END
                                                !an END statement is optional
BULK / GA_RICH /
       GA2AS(1)/3.0/
       GA3AS(1)/3.0/
       GA2AS(1)/2.0/                  !THIS NAME IS A DUPLICATE AND WILL BE IGNORED
BULK AS(B)                                    !BULK PHASE WITH NO NAME SUPPLIED
                                     !ONLY ONE BULK SPECIES AND NO DENSITY SUPPLIED
END
```

Figure 9 Sample Bulk Data

Thermodynamic Data

Any chemical species that appears in a problem must have thermodynamic data associated with it. This data is used in evaluation of thermodynamic properties (entropy, enthalpy, heat capacity) and reverse reaction rate constants through the equilibrium constant. Often thermodynamic data for a species, for instance a surface species, is unknown. Such data can sometimes be calculated via theoretical techniques. However, the user can work around the need for actual thermodynamic data for all species with the following "trick." If every reaction in the mechanism is either irreversible, or if Arrhenius rate parameters are given explicitly for the reverse reaction, then the thermodynamic data for species are not actually used for anything related to the kinetics. In this case, the user can supply "dummy" thermodynamic data for the surface species to satisfy the Interpreter requirement.

Thermodynamic data may be extracted from a database (therm.dat) and/or read from the interpreter input file. If all the thermodynamic data are extracted from the database, then no thermodynamic data are required from the input file. However, if the user wishes to override information in the database or provide data on species not in the database, then SURFACE CHEMKIN-III Interpreter thermodynamic input is needed. In any case the format for the information is the same.

The format (see Table III) is a minor modification of that used by Gordon and McBride[8] for the Thermodynamic Database in the NASA Chemical Equilibrium code. Our modification allows for a different midpoint temperature for the fits to the properties of each chemical species. We also allow a species to be composed of a maximum of five elements, not four. However, the formatting is such that the CHEMKIN-III Interpreter can use the NASA database directly without any modification.

As Table III indicates, the pertinent information includes the species name, the elemental composition of the species, and the temperature ranges over which the polynomial fits to thermodynamic data are valid. The fits to C_p^o/R, H^o/RT, and S^o/R consist of seven coefficients for

each of two temperature ranges [see Eqs. (17-19)].* Further information about the fitting procedure and data for many species are in a report on the CHEMKIN Thermodynamic Database.[6]

When thermodynamic data input is required, it must immediately follow species data (SITE and BULK data). The first thermodynamic data line must start with the word THERMO (or THER). If all the thermodynamic data are input directly to the Interpreter, then the first line of this section must read THERMO ALL and the code will not expect a Thermodynamic Database from unit number LTHRM; for this option the next line must be line 2 of Table III. For either option, the subsequent thermodynamic data lines must be in the format of lines 3 - 6 of Table III. (For the THERMO option the default midpoint temperature is taken from the line 2 information already in the Thermodynamic Database.) As many species as needed can be included as THERMO input. Thermodynamic data for any species that were not declared in the SITE and BULK sections will be ignored.

Figure 10 shows some examples of thermodynamic property input. In these examples for OH, OH+, and OH-, it is seen from columns 25 - 34 that the elemental composition of each molecule is one O atom and one H atom. Columns 35 - 39 indicate that two of the species, OH+ and OH-, are ionic since they contain -1 and +1 electrons (E), respectively. The G in column 45 indicates that all three species are gaseous. (This phase information is ignored by SURFACE CHEMKIN-III.) The 1000.00 in columns 66 - 73 for OH+ indicates the common temperature (in K) between the high- and low-temperature fits. If columns 66 - 73 are left blank, as they are for OH+ and OH-, then the common temperature is that given in columns 21 - 30 of line 2 in Table III, which in this example is in the Thermodynamic Database.

The following cases summarize the possibilities for specifying thermodynamic data.

Case 1: All thermodynamic data from database only

1. Assign the database as file "therm.dat"
2. No THERMO data required as input.

Case 2: Thermodynamic data from database input:

1. Assign the database as file "therm.dat"
2. Include the following lines in the Interpreter input:
   THERMO
   Data in Table III format (lines 3 - 6 repeated) for species not in the database or to override species in database.
   END (optional)

---

* Additional temperature ranges and their fit coefficients may be accommodated by minor changes to the Interpreter and the Thermodynamic Database.

Case 3: All thermodynamic data from input

1. Do not need to supply the database.
2. Include the following lines in the Interpreter input:
   THERMO ALL
   Line 2 of Table III format.
   Data in Table III format (lines 3 - 6 repeated) for at least all species named in the species data.
   END (optional)

```
THERMO
OH                121286O   1H   1          G  0300.00   5000.00  1000.00      1
 2.88273048E+00 1.01397431E-03-2.27687707E-07 2.17468370E-11-5.12630534E-16    2
 3.88688794E+03 5.59571219E+00 3.63726592E+00 1.85091049E-04-1.67616463E-06    3
 2.38720266E-09-8.43144185E-13 3.60678174E+03 1.35886049E+00                   4
OH+               121286O   1H   1E  -1     G  0300.00   5000.00              1
 2.71905875E+00 1.50857132E-03-5.02936928E-07 8.26195154E-11-4.94745253E-15    2
 1.57634141E+05 6.23453617E+00 3.32697868E+00 1.34578592E-03-3.77716765E-06    3
 4.68774974E-09-1.78098215E-12 1.57402938E+05 2.74404216E+00                   4
OH-               121286O   1H   1E   1     G  0300.00   5000.00              1
 2.84620452E+00 1.04183471E-03-2.41685058E-07 2.48321502E-11-7.77560523E-16    2
-1.80728027E+04 4.42271185E+00 3.39003754E+00 7.92238105E-04-1.94342965E-06    3
 2.00176964E-09-5.70208702E-13-1.83049375E+04 1.24989224E+00                   4
END
```

Figure 10 Examples of Thermodynamic Data Input

Table III. Summary of the Rules for Thermo Data

Line Number	Contents	Format	Column
1	THERMO (or THERMO ALL[a])	Free	Any
2[b]	Temperature ranges for 2 sets of coefficients; lowest T, common T, highest T	3F10.0	1 to 30
3	Species name (must start in Column 1)	16A1	1 to 16
	Date (not used by the code)	6A1	19 to 24
	Atomic symbols and formula	4(2A1,I3)	25 to 44
	Phase of species (S, L, or G for solid, liquid, or gas, respectively; not used by the code)	A1	45
	Low temperature	E10.0	46 to 55
	High temperature	E10.0	56 to 65
	Common temperature (if needed) (blank for default)	E8.0	66 to 73
	Atomic symbol and formula (if needed; blank for default)	2A1, I3	74 to 78
	The integer 1	I1	80
4	Coefficients a_1 - a_5 in Eqs. (17-19), for upper temperature interval	5(E15.0)	1 to 75
	The integer 2	I1	80
5	Coefficients a_6, a_7 for upper temperature interval, and a_1, a_2 and a_3 for lower	5(E15.0)	1 to 75
	The integer 3	I1	80
6	Coefficients a_4, a_5, a_6, a_7 for lower temperature interval	4(E15.0)	1 to 60
	The integer 4	I1	80
. . .	Repeat lines 3 - 6 for each species.		
last	END (Optional, end of thermodynamic data)	Free	Any

[a]Use only when all the thermodynamic data are to be taken from Surface Interpreter input.

[b]Include line 2 in the Interpreter input *only* with THERMO ALL. Line 2 does not appear in the database itself.

Surface-Reaction Mechanism Description

The surface-reaction mechanism may consist of any number of chemical reactions involving the solid species named in the site and bulk data, as well as the gas-phase species declared in the gas-phase CHEMKIN-III Interpreter. A reaction may be reversible or irreversible.

Reaction data must start with the word REACTIONS (or REAC). On the same line the user may specify units of the Arrhenius rate coefficients to follow by including the words CAL/MOLE, KCAL/MOLE, JOULES/MOLE, KJOULES/MOLE, EVOLTS, or KELVINS to indicate the units for all rate parameters that have energy units, e.g., the activation energy E_i of Eq. (35), an ion energy such as E_{ion} or $E_{\text{ion},0}$ of Eq. (51), or $E_{\text{yield},0}$ of Eq. (62).* The special words MOLES or MOLECULES can appear on the REACTIONS line to indicate the units for A_i [see Eq. (35)]. If MOLECULES is specified, then the units for A_i are cm-molecules-sec-K. If units are not specified, A_i and E_i are assumed to be in cm-mole-sec-K and cal/mole, respectively. The lines following the REACTION line contain reaction descriptions together with their Arrhenius rate coefficients. The reaction description is composed of reaction data and perhaps optional auxiliary information data.

The Interpreter normally considers any reaction that does not conserve the number of surface sites in each surface phase to be in error. If the user wishes to include such reactions (which may be perfectly valid), the word NONCON must appear on the REACTION line.

If the user does not wish to include the Motz-Wise[10] correction in the conversion between a sticking coefficient and a rate constant, i.e., Eq. (57), the user has two choices. The keyword MWOFF can be included on the REACTION line to turn off this correction for all reactions that are specified as sticking-coefficient reactions, or MWOFF can be used as an auxiliary keyword following an individual reaction to turn off the Motz-Wise correction for that reaction only. Conversely, if the MWOFF parameter was given on the REACTION line, the user can include the MWON auxiliary keyword following an individual reaction, specifying that theMotz-Wise correction is to be used for that reaction only.

---

* Even if the default energy units are changed by giving one of these keywords, the temperature appearing in the Arrhenius expression of Eq. (35) , i.e., in T raised to the β power and in the denominator of the activation energy term, is still in Kelvins.

Surface Reaction Data

Each reaction "line" is divided into two fields, where a "line" may take up two or more physical lines if it is more than 80 characters long. A reaction data line is continued on the next line using the special character "&"; any information following the & symbol on the same line is ignored. The first field in the reaction line contains the symbolic description of the reaction, which the second contains the Arrhenius rate coefficients. Both fields are format free, and blank spaces are ignored. All characters on a line following an exclamation mark (!) are considered comments and are ignored. Blank lines are also ignored.

The reaction description, given in the first field, must be composed of the species symbols, coefficients, and delimiters as summarized below.

Symbols: Each species name in a reaction is described with a unique sequence of characters as they appear in the species data and the thermodynamic data. However, if a species name is not unique (because it is duplicated in another phase), the name must be modified by appending its slash-delimited phase name, i.e. as *name/phase/*.

Coefficients: A species symbol may be preceded by a positive integer or real* coefficient. This coefficient is interpreted as the number of moles of the particular species present as either a reactant or a product; e.g., 2OH is equivalent to OH + OH. (The "#" symbol is used to mark stoichiometric coefficients that are additionally multiplied by a "YIELD" coefficient. This is explained later.)

Delimiters:

- `+` A plus sign is the delimiter between all reactant species names and between all product species names.
- `=` An equality sign is the delimiter between the last reactant and the first product in a reversible reaction.
- `<=>` An equality sign enclosed by angle brackets can also be used as the delimiter between the last reactant and the first product in a reversible reaction.
- `=>` An equality sign with an angle bracket on the right is the delimiter between the last reactant and the first product in an irreversible reaction.

The second field of the reaction line is used to define the Arrhenius rate coefficients A_i, β_i, and E_i in that order, as given by Eq. (35). At least one blank space must separate the last species name in the reaction and first number. The three numbers must be separated by at least one blank space, be stated in either integer, floating point, or E format (e.g., 123 or 123.0 or 12.3E1), and have units associated with them (although the units do not appear on the input line). Unless modified by the REACTION line, the

* This is a new feature in SURFACE CHEMKIN-III.

default units for A_i are cgs (cm, sec, K, mole), the exact units depending on the order of the reaction. The factor β_i is dimensionless. The default units for the activation energies are cal/mole.

The second field of the reaction line may optionally be used to specify the coefficients a_i, b_i, and c_i of Eq. (52) for a sticking coefficient. In order for the second field to apply to sticking coefficient parameters, the next line of input must contain the auxiliary information word STICK.

Examples of some reaction data are shown in Fig. 11. Table IV summarizes the reaction data rules.

```
REACTIONS                      KCAL/MOLE                    NONCON

ASH3 + AS(P) <=> ASH3(P) + AS(D)                  4.0E11   0 25          ! Ref. 21
 ! ASH3 + AS(P) <=> ASH3(P) + AS(D)       4.0E11  0  0    ! same as previous reaction
ASH <=> AS(D) + H(S)    1.0   0  0
      STICK
GA(CH3)3(L) + GA2AS(A) <=> AS + GA(CH3)(L) + 2 GAME     & continued on next line
           1.0E13   0  4000.
```

Figure 11 Examples of Reaction Data

Table IV. Summary of the Rules for Surface Reaction Data

1. The first reaction line must start with the word REACTIONS (or REAC), and may be followed by units definition(s), the word MWOFF, or the word NONCON.
2. Valid unit declarations are EVOLTS, KELVINS, CAL/MOLE, KCAL/MOLE, JOULES/MOLE, KJOULES/MOLE, MOLES, and MOLECULES.
3. The word NONCON is required on the first reaction line if any of the reactions do not conserve the number of surface sites of a given type.
4. The word MWOFF can be used to turn off the Motz-Wise[10] correction of Eq. (57), for all sticking-coefficient reactions, or the word MWON can be used to specify that the Motz-Wise correction is to be used for all sticking-coefficient reactions (the default). Including MWOFF or MWON as an auxiliary keyword for an individual reaction (discussed later) will override the setting given on the REACTION line.
5. The reaction description can begin anywhere on this line. All blank spaces, except those separating the Arrhenius coefficients, are ignored.
6. Each reaction description must have =, <=>, or => between the last reactant and the first product.
7. Each species in a reaction is described with a unique sequence of characters (name) as they appear in the species data and the thermodynamic data. However, if a species name is not unique (because it is duplicated in another phase), the name must be modified by appending its slash-delimited phase name, i.e. as *name/phase/*.
8. Stoichiometric coefficients are represented by an integer or real number preceding a species name. The default is to assume a stoichiometric coeffient of 1. The "#" symbol preceding the stoichiometric coefficient denotes a coefficient which is additionally multiplied by a "yield" multiplier.
9. A reaction description may be contained on more than one line. If a line contains the symbol &, all information following the & symbol will be ignored and the next line will be considered a continuation of the first.
10. Three Arrhenius coefficients must appear in order (A_i, β_i, and E_i) on each Reaction line, separated from each other and from the reaction description by at least one blank space; no blanks are allowed within a number.
11. There cannot be more than six reactants or six products in a reaction.
12. To specify a sticking coefficient rather than a rate constant the three numbers after the reaction description have the meaning a_i, b_i, and c_i (see Eq. (52)) and the auxiliary information word STICK must appear on the next line of input. To use this option the reaction must have only one gas-phase species as a reactant and its stoichiometric coefficient must be 1.
13. All characters on a line following an exclamation mark are comments.

Surface Reaction Auxiliary Information Data

Auxiliary information data appears on one or more separate lines after the reaction data line is read, and serves to modify or give additional parameters needed to evaluate that reaction's rate expression. The format in an auxiliary information line is a character string keyword followed by a slash-delimited (/) field containing an appropriate number of parameters (either integer, floating point, E format, or character).

For a reversible reaction, auxiliary information data may follow the reaction to specify Arrhenius parameters for the reverse-rate expression. Here, the three Arrhenius parameters (A_i, β_i, and E_i) for the reverse rate must follow the auxiliary keyword REV. Using this option overrides the reverse rates that would normally be computed through the equilibrium constant, Eq. (36).

It sometimes happens that two or more reactions can involve the same set of reactants and products, but proceed through distinctly different reaction channels. In these cases it may be appropriate to state a reaction mechanism that has two or more reactions with identical reactants and products, but have different rate parameters. However, duplicate reactions are normally considered errors by the Interpreter; if the user requires duplication (e.g., the same reactants and products with different Arrhenius parameters), an auxiliary information statement containing the keyword DUP (with no parameters) must follow the reaction line of each duplicate reaction (including the first occurrence of the reaction that is duplicated.) For example, to specify different rate expressions for each of three identical reactions, there must be three occurrences of the keyword DUP, one following each of the reactions.

If the three coefficients given in the second field of the reaction line are to be interpreted as the parameters a_i, b_i, and c_i of Eq. (48) for a sticking coefficient, then the keyword STICK (with no parameters) must follow the reaction line as auxiliary information. There can be only one gas-phase reactant species in a sticking-coefficient reaction; moreover, its stoichiometric coefficient must be 1.

By default the Motz-Wise[10] correction of Eq. (57) will be applied for all reactions using sticking coefficients unless the MWOFF keyword is given on the REACTIONS line (discussed earlier), in which the new default will be to omit this correction term. Whichever of these choices has been set as the default can be overridden for an individual sticking-coefficient reaction by including the auxiliary keyword MWOFF or MWON following the reaction line.

To apply the Bohm velocity correction for a reaction involving a positive ionic species as in Eq. (58) the auxiliary keyword BOHM (with no parameters) must follow the reaction line. In this case the three coefficients given in the second field of the reaction line are interpreted as the parameters a_i, b_i, and c_i of Eq. (58). A BOHM reaction can have only one gas-phase reactant species; this gas-phase reactant must be a positive ion, and its stoichiometric coefficient must be 1. The electron must be declared as a gas-phase species in the gaa-phase reaction mechanism, i.e., in the CHEMKIN-III Interpreter input.

A reaction rate constant can be modified to include a dependence upon the energy of a positive ionic reactant species through use of the ENRGDEP auxiliary keyword, which is followed by the three parameters $E_{\text{ion},0}$, f_i, and g_i (slash-delimited) of Eq. (51). A reaction declared with ion-energy dependence must contain exactly one positive ionic reactant species.

The reaction order, i.e., the dependence of the reaction's rate-of-progress upon the concentration of chemical species, can be changed via the FORD or RORD auxiliary keywords for the forward or reverse reaction, respectively. Each occurrence of these keywords must be followed by the species name and the new reaction order, e.g., FORD/SI(S) 0.5/). This option overrides the values of υ'_{ki} and υ''_{ki} in Eq. (32) pertaining to the particular species named on the line. The reaction order for all other species maintain their default values of υ'_{ki} and υ''_{ki}. Multiple occurrences of the FORD and RORD construct may appear on the auxiliary line. A FORD or RORD keyword can be specified even for species that do not appear as a reactant or product in the reaction, although one might reasonably wonder how such a functional dependence could occur.

To modify the expression for the forward rate constant by optional coverage parameters (see Eq. (48)) one uses the auxiliary information keyword COV followed by (slash delimited) surface species name and the three parameters η_{ki}, μ_{ki}, and ε_{ki}. More than one set of COV data can appear for a given reaction, and these would be applied multiplicatively as in Eq. (48).

Ion-enhanced reaction yield can be applied to a reaction using the following two steps. First, each species name (or the species stoichiometric coefficient if one was given) which is subject to the ion-energy yield enhancement is preceded by the pound sign (#). The "sub-reaction" of species and coefficients demarked with the # sign must satisfy mass, elemental, charge and site balance (unless the NONCON flag was also given on the REACTION line). Second, following the reaction the auxiliary keyword YIELD must appear, followed by the four parameters h_{yield}, $E_{\text{yield},0}$, t_i, and u_i (slash-delimited) of Eq. (62). A reaction declared with ion-enhanced reaction yield must contain one (and only one) positive ionic reactant species.

The default units for specifying the reaction rate parameters (either the usual defaults set-up for SURFACE CHEMKIN-III, or the set of units requested by the user on the REACTION line) can be overridden for an individual reaction through use of the UNITS auxiliary keyword.* Following the reaction, one activates this option with UNITS/*string*/, where *string* is one of the unit specifiers EVOLTS, KELVINS, CAL/MOLE, KCAL/MOLE, JOULES/MOLE, or KJOULES/MOLE (to change the units for parameters that have energy units),† or MOLES or MOLECULES (to change the units of the pre-exponential A_i). The UNITS auxiliary keyword allows only one *string* parameter, but the user can repeat the UNITS/*string*/ option as many times as needed for a given reaction.

Any number of auxiliary information lines may follow a reaction line, in any order, and any number of keywords may appear on an auxiliary information line; however, an auxiliary keyword and its parameter(s) must appear on the same line.

Examples of auxiliary information are shown in Fig. 12. The above rules are summarized in Table V.

```
REACTIONS                                                    KCAL/MOLE
!      THE FOLLOWING ARE *CONTRIVED* EXAMPLES OF AUXILIARY KEYWORD USAGE
    SICL(S)  <=>  CL + SI(S)                                 1.0E-3  0.0    2.
                                                          REV/1.0E13  0.0   37./
    CL + SICL(S) <=>  CL2 + SI(S)                            0.1     1.1   20.
          DUPLICATE   STICK
          RORD /SI(S)  0/
    CL + SICL(S) <=>  CL2 + SI(S)                            1.4E11  0.0   15.
          DUPLICATE   COV/SICL(S)  -1.2  0.5  32./  FORD/CL+  1.0/
    CL*       =>  CL                                         1.0     0.0    0.
          STICK       MWOFF
    E + CL+ + SICL3(S) + SI(B) => SICL4 + SI(S)              0.50    0.0    0.
       BOHM
       ENRGDEP/1. 0.5 1.0/  UNITS/EVOLT/
    E + CL+ + #SICL3(S) + #SI(B) + SICL(S) &
          => SICL2(S) + #SICL2 + #SICL(S)              0.50  0.0  0.0
       BOHM
       YIELD/0.0712 1.21 0.5 1.0/  UNITS/EVOLT/
    E + CL2+ + SICL3(S) + SI(B) => SICL4 + SICL(S)     0.50  0.0  0.0
       FORD/ CL2+  2.43/
```

Figure 12 Examples of Auxiliary Information Definitions

---

* This is also a new feature in the gas-phase CHEMKIN-III Interpreter.

† Even if the default energy units are changed by giving one of these keywords, the temperature appearing in the Arrhenius expression of Eq. (35) , i.e., in T raised to the β power and in the denominator of the activation energy term, is still in Kelvins.

Table V. Summary of the Rules for Auxiliary Information Data

1. Auxiliary information lines may follow a reversible reaction to specify the reverse rate parameters explicitly; auxiliary information *must* follow any reactions that are duplicated.
2. Auxiliary keyword declarations may appear anywhere on the line, in any order.
3. Any number of auxiliary keywords may appear on a line, and more than one line may be used, but a keyword and its parameter(s) must appear on the same line.
4. Multiple keywords appearing on the same line must be separated by at least one blank space.
5. Any blank spaces between a keyword and the first slash are ignored and any blanks between the slashes and parameter(s) are also ignored. However, no blank spaces are allowed within a keyword or parameter.
7. The keyword REV followed by three slash-delimited Arrhenius coefficients may be used to specify the reverse rate parameters.
8. The keyword DUPLICATE (or DUP) must follow every occurrence of a duplicated reaction.
9. The keyword STICK indicates that the three coefficients on the reaction line are to be interpreted as the parameters a_i, b_i, and c_i in Eq. (52). There must be exactly one gas-phase reactant species; its stoichiometric coefficient must be 1.
10. The keyword COV is used to modify the forward rate constant by the expression in Eq. (48). The word COV is followed by a surface species name and the three coverage parameters η_{ki}, μ_{ki}, and ε_{ki}. The four entries after the word COV are slash-delimited.
11. The keyword BOHM indicates that the three coefficients on the reaction line are to be interpreted as the parameters a_i, b_i, and c_i in Eq. (58); the Bohm velocity correction is applied. There must be exactly one gas-phase reactant species and that species must be a positive ionic species; its stoichiometric coefficient must be 1. Only irreversible reactions are allowed with this option. The electron must be declared in the list of species names in the CHEMKIN-III Interpreter input.
12. The keyword ENRGDEP allows the rate constant to depend on ion energy according to Eq. (51). The keyword is followed by the three parameters $E_{\mathrm{ion},0}$, f_i, and g_i, which are slash-delimited. There must be exactly one positive ionic reactant species in the reaction. Only irreversible reactions are allowed with this option.
13. The keyword FORD or RORD can be used to change the reaction order (with respect to species concentration) of the forward or reverse reaction, respectively, for any species in the mechanism, regardless of whether the species appears as a reactant or a product in the reaction. The species name and the new reaction order (slash-delimited) follow the keyword.
14. To modify the stoichiometric coefficients in a sub-reaction using the ion-yield option, one must precede each species in the sub-reaction (or its stoichiometric coefficient) with the pound sign (#). Following the reaction line, declare the YIELD auxiliary keyword, then the four parameters h_{yield}, $E_{\mathrm{yield},0}$, t_i, and u_i of Eq. (62), between slashes. There must be exactly one positive ionic reactant species in the reaction. Only irreversible reactions are allowed with this option. The sub-reaction demarked with the # symbols must satisfy mass, elemental, charge and site balance (unless the NONCON option appeared on the REACTION line).

Table V. (cont.) Summary of the Rules for Auxiliary Information Data

15. The UNITS auxiliary keyword can be used to override the current default units for parameters with energy units or the pre-exponential for a given reaction. The usage is UNITS/*string*/, where *string* is one of the following: EVOLTS, KELVINS, CAL/MOLE, KCAL/MOLE, JOULES/MOLE, or KJOULES/MOLE (for parameters with energy units), or MOLES or MOLECULES (for pre-exponential).
16. The string MWOFF can be used to turn off the Motz-Wise[10] correction of Eq. (57) or the string MWON can be used to turn on this correction for a sticking coefficient reaction. Using the MWOFF or MWON auxiliary keyword overrides whatever default was setup on the REACTION line or the default supplied by SURFACE CHEMKIN-III (which is MWON).

Problems Having No Reactions

In some problems only information about the surface and bulk species is needed (e.g., chemical equilibrium computations). For these cases it is not necessary to include reaction data. The Interpreter will create the linking file surf.bin, but it will not contain any reaction information. Therefore, no subroutines in the SURFACE CHEMKIN-III Subroutine Library that deal with chemical reactions (e.g., chemical production rates) will be used (although doing so would not generate an error; the production rates of all species would be returned as zero).

Unit Conversion for Pre-exponential Factor

The SURFACE CHEMKIN-III Interpreter provides the option of specifying units of the Arrhenius parameters. The parameters are always stored internally in the same way, i.e., activation energies in Kelvins, dimensionless temperature exponents, and pre-exponential factors consistent with moles-cm-sec-K. The program converts the user input activation energies in an obvious way. However, it is worthwhile to state explicitly the conversion for the Arrhenius pre-exponential constant. In converting from "molecules" to "moles"

$$A_i(\text{moles}) = A_i(\text{molecules}) \times \left(N_A\right)^{n-1} \tag{77}$$

where N_A is Avogadro's number, and

$$n = \sum_{k=1}^{K_s^l(N_s^l)} \upsilon'_{ki} \tag{78}$$

Note that the υ'_{ki} are the stoichiometric coefficients for the gas-phase and surface reactant species (not product species or bulk-phase species).

Error Checks

The Interpreter checks each input line for proper syntax and writes self-explanatory diagnostic messages on the interpreter output file if errors are encountered. If an error occurs, the Interpreter continues to read and diagnose the input, but an error flag is written to the Linking file and Surface Library subroutine SKINIT will not initialize the work arrays. Therefore, the input must be error free before any of the SURFACE CHEMKIN-III subroutines can be called in an application code.

Possibilities for an error condition are as follows:

Site and Bulk Species Data

- A species symbol (name) duplicated within a phase is not considered a fatal error, but is eliminated from consideration and a warning diagnostic message is printed.
- No site density is found for a declared site, or the site density is negative.
- No thermodynamic data are found for a declared species.
- There are more species than the Interpreter is dimensioned for*
- A site or bulk phase name duplicates another phase name (gas surface site, or bulk phase name).
- A phase or species name contains an illegal character.
- Site occupancy number is negative.
- Bulk density for a bulk species is negative.

Thermodynamic Data

- Thermodynamic data are format sensitive and therefore provide possibilities for error if not formatted exactly as described by Table III.
- An element in the thermodynamic data for a declared species is not included in the CHEMKIN-III Interpreter input element data.
- With the THERMO ALL option, line 2 (of Table III) is not found.

Reaction Data

- A delimiter =>, <=>, or = between the reactants and the products is not found.
- Three Arrhenius parameters are not found.
- Reactants and / or products species names are not properly delineated by a plus sign (+).
- A species listed as a reactant or product is not declared in the species data.

---

* This limit may be changed by a simple modification of a parameter statement in the SURFACE CHEMKIN-III Interpreter

A reaction does not satisfy elemental balance.

The number of sites in a reaction does not balance and the word NONCON was not included on the first REACTIONS line.

The charge of the reaction does not balance.

A reaction is a duplicate not declared by the auxiliary data keyword DUP.

There are more reactions than the Interpreter is dimensioned for.

There are more than six reactants or six products in a reaction.

A species name is not unique and it is not followed by a slash-delimited phase name, i.e. as *name/phase/*.

Surface Reaction Auxiliary Data

An unknown or misspelled keyword occurs.

Parameters for a keyword are not enclosed in slashes.

There are the wrong number of parameters for a keyword.

REV is declared for an irreversible reaction.

Pre-exponential factor for a sticking coefficient is negative.

A species name is not unique and it is not followed by a slash-delimited phase name, i.e. as *name/phase/*.

For a sticking-coefficient reaction, there is more than one gas-phase species, or the stoichiometric coefficient for the gas-phase species is not 1.

More than one BOHM declaration appeared for a given reaction.

BOHM keyword is given for a reversible reaction.

A positive ionic species did not appear as a reactant or its stoichiometric coefficient was not 1 in a BOHM reaction.

More than 1 positive ionic species was a reactant in a BOHM reaction.

The electron species was not declared in the list of species in the CHEMKIN-III Interpreter input.

Invalid *string* given with the UNITS auxiliary keyword.

More than one ENRGDEP declaration appeared for a given reaction.

A positive ionic species did not appear as a reactant or its stoichiometric coefficient was not 1 in a ENRGDEP reaction.

Wrong number of ENRGDEP parameters given.

Invalid species name given for FORD or RORD keyword.

A reaction order value was not found with the FORD or RORD keyword.

RORD given for an irreversible reaction.

More than MAXORD changes in reaction order declared for a given reaction. (Should this error occur, the user can change a parameter statement in the SURFACE CHEMKIN-III Interpreter to increase MAXORD.)

No species coefficients were demarked with a # symbol for a YIELD reaction.

YIELD keyword given for a reversible reaction.

A positive ionic species did not appear as a reactant or its stoichiometric coefficient was not 1 in a YIELD reaction.

Wrong number of YIELD parameters given.

More than 1 positive ionic species was a reactant in a YIELD reaction.

VI. DATA STRUCTURES IN SURFACE CHEMKIN-III

Reaction mechanisms have to consider species that may be in the gas phase, on various surface sites, or in various bulk phases, and the number of species, surface sites, and bulk mixtures change from problem to problem. Furthermore, with the concept of multiple materials, all of these quantities as a group can be specified independently for as many different materials as desired. The user may want to refer to a species by an integer species index (such as in a Fortran DO loop) or refer to a species by a character string (such as in an input/output routine). Furthermore, in some circumstances it may be convenient to refer to a species in a long list of all species, and in other cases it may be convenient to refer only to those species in a particular phase. Thus it is important to have a data structure that is flexible enough to capture the required generality, yet sufficiently "friendly" to make it easy to use. In this chapter we use specific examples to illustrate the data structures and how to use them in an application program.

Consider a system involving Gallium Arsenide deposition* where an example gas-phase reaction mechanism is shown in Fig. 13 and a surface mechanism is shown in Fig. 14. The gas-phase mechanism includes fourteen species, even though only eleven of them appear directly in the gas-phase reaction mechanism. Arsine (ASH3) in our notation) is typically a reactant gas in a Gallium Arsenide deposition process. Monatomic arsenic (AS) and the arsenic dimer (AS2) have been included because the surface reaction mechanism includes desorption reactions, which introduce them into the gas. Therefore, because all gas-phase species are handled by CHEMKIN, these extra three species must be included as input to the CHEMKIN-III Interpreter. Of course, the gas-phase and surface-reaction mechanisms should not be considered entirely independently. If the surface mechanism generates a gas-phase species, then it is probably reasonable to ask if there are plausible gas-phase reactions that can consume it. For example, perhaps a gas-phase recombination reaction could be included to form an arsenic dimer.

The surface mechanism in the example has two surface sites, called "PLANE" and "LEDGE." There are five bulk "mixtures" (phases) although three of them are not actually mixtures, but pure compounds. The first three bulk mixtures have been assigned the names "GA-RICH," "GA-POOR," and "GA-STOICH," whereas the remaining two have not been given any specific names.

---

* The mechanisms have been contrived to illustrate a variety of points about the data structures and should not be taken as a source of data for Gallium Arsenide deposition processes.

```
ELEMENTS GA AS H C END
SPECIES
   ASH3 AS2 GAME3 GAME2 GAME GA CH3 CH4 H C2H6 C2H5 C2H4 H2
   AS
END
REACTIONS
2CH3(+M)<=>C2H6(+M)              0.903E+17 -1.18    654.
 LOW/ 3.18E41 -7.03 2762./
 TROE/ .6041 6927. 132./
 H2/2/
CH3+H(+M)<=>CH4(+M)              0.600E+17 -1.0      0.
 LOW/8.0E26 -3. 0.0/
 H2/2/
CH4+H<=>CH3+H2                   0.220E+05  3.0   8750. ! CLARK&DOVE
C2H6+CH3<=>C2H5+CH4              0.550E+00  4.0   8300.
C2H6+H<=>C2H5+H2                 0.540E+03  3.5   5210. ! CLARK&DOVE
H+C2H4(+M)<=>C2H5(+M)            0.221E+14  0.0   2066. ! MICHAEL,GLAN&TROE
 LOW/3.47E-12  2.76  2120./
  H2/2/
C2H5+H<=>CH3+CH3                 1.0E+14  0.0     0.
H+H+M<=>H2+M                     0.100E+19 -1.0   0.
H+H+H2<=>H2+H2                   0.920E+17 -0.6   0.
GAME3<=>GAME2+CH3                0.347E+16  0.0   59500. ! JACKO AND PRICE
GAME2<=>GAME+CH3                 0.871E+08  0.0   35410.
END
```

Figure 13 Sample Gas-Phase Reaction Mechanism

```
                                          ! IPHASE = 1 IS THE GAS, NPHASE(1)=KKGAS
SITE/PLANE/       SDEN/1.0E-09/           ! IPHASE = 2, NPHASE(2) = 3
   ASH3(P)   ASH(P)   AS(P)
SITE/LEDGE/       SDEN/1.0E-13/           ! IPHASE = 3, NPHASE(3) = 4
   GA(CH3)3(L)   GA(CH3)2(L)   GA(CH3)(L)   GA(L)
END
BULK/GA_RICH/                             ! IPHASE = 4, NPHASE(4) = 2
   GA2AS(1)/3.00/   GA3AS(1)/3.00/        ! THE DENSITY OF THE SPECIES IS OPTIONAL
BULK/GA_POOR/  GAAS2(2)/3.15/  GAAS3(2)/3.15/ GAAS4(2)/3.15/  END
                                          ! IPHASE = 5, NPHASE(5) = 3
BULK/GA_STOICH/                           ! IPHASE = 6, NPHASE(6) = 1
    GAAS(3)/5.24/                         ! A BULK PHASE CAN HAVE JUST ONE SPECIES
BULK GA(B)/2.5581/                        ! IPHASE = 7, NPHASE(7) = 1
BULK AS(B)/2.7489/                        ! IPHASE = 8, NPHASE(8) = 1
REACTIONS
   ASH3  + AS(P) <=> ASH3(P)        + AS(B)               4.0E11  0   0.
   GAME3 + GA(L) <=> GA(CH3)3(L)    + GA(B)               1.0E11  0   0.
   GA(CH3)3(L) + GA2AS(1) <=> AS    + GA(CH3)(L) + 2 GAME  1.0E13  0   40000.
                                                 REV  /1.0E11  0   25000./
   GA(CH3)3(L) + GA3AS(1) <=> AS    + GA(L) + 3GAME       1.0E13  0   40000.
            DUPLICATE
   GA(CH3)3(L) + GA3AS(1) <=> AS    + GA(L) + 3GAME       1.0E12  0.5 30000.
            DUPLICATE
   GA2AS(1) <=> GA(B) + GAAS(3)                           1.0E13  0   40000.
   GAAS2(2) <=> AS(B) + GAAS(3)                           1.0E13  0   40000.
   GAAS3(2) <=> AS2  + GAAS(3)                            1.0E13  0   40000.
   GAAS4(2) <=> AS(B) + AS2 + GAAS(3)                     1.0E13  0   40000.
END
```

Figure 14 Sample Surface-Reaction Mechanism

The data structure is illustrated in Fig. 15, where each column represents a "phase." The gas phase is *always* the first phase; the surface site phases are stored next, followed by the bulk phases. The species are stored sequentially beginning with the first gas-phase species and ending with the last bulk species. The gas-phase species are listed in the same order as they were presented to the CHEMKIN-III Interpreter. The species in the surface and bulk phases are in the same order as they appear in the Surface Interpreter input.

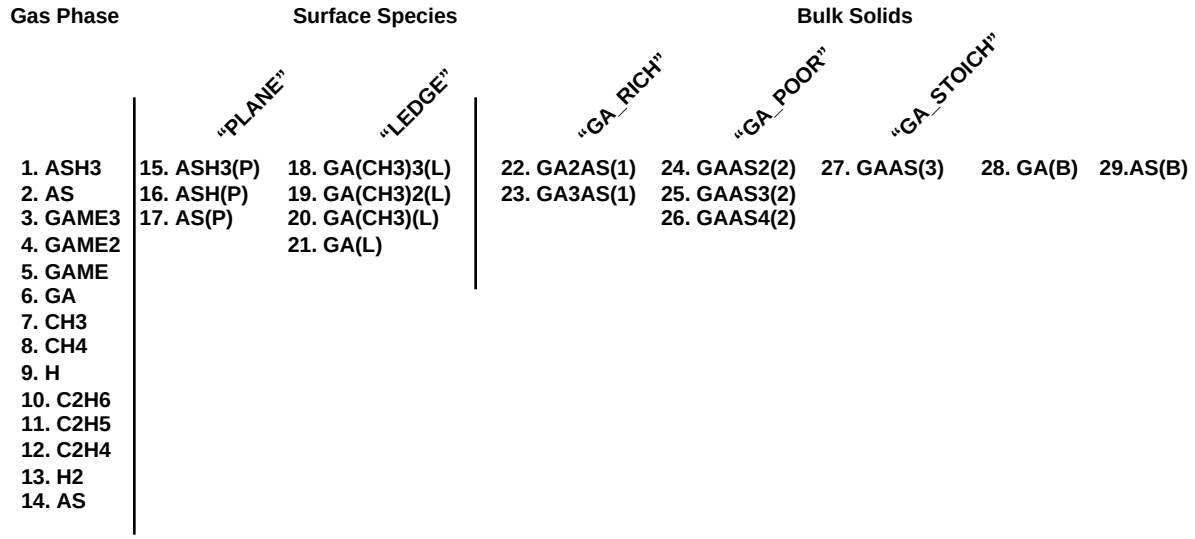


Figure 15 Schematic Diagram of the Phase and Species Data Structure

There are several indices and arrays of indices that are quite useful in referencing particular species and phases. They are returned from two subroutines, SKINDX and SKPKK, described in the following:

SUBROUTINE SKINDX (ISKWRK, NELEM, KKGAS, KKSUR, KKBULK, KKTOT, NNPHAS, NNSURF, NFSURF, NLSURF, NNBULK, NFBULK, NLBULK, IISUR)
SUBROUTINE SKPKK (ISKWRK, KKPHAS, KFIRST, KLAST)

The input to SKINDX and SKPKK is the integer work space ISKWRK. On output all the indices will have values that pertain to the particular reaction mechanisms. For the example we are considering here, the indices have the following values:

NELEM, Number of elements—Here, NELEM=4. As presented on the first line of input to the CHEMKIN-III interpreter, the four elements are GA, AS, H, C.

KKGAS, Number of gas-phase species—Here KKGAS=14

KKSUR, Number of surface species—There are seven surface species in this problem and they exist on two sites. Thus, KKSUR=7.

KKBULK, Total number of bulk species—There are eight bulk species that are arranged in five bulk phases. Thus, KKBULK=8.

KKTOT, Total number of species—KKTOT=KKGAS+KKSUR+KKBULK, so in this example KKTOT=29.

NNPHAS, Number of phases (gas + sites + bulk)—Here there are eight phases, NNPHAS=8. The first is the gas phase, the second two are surface phases, and the final five are bulk phases.

NNSURF, Number of surface phases—Here there are two surface phases, NNSURF=2.

NFSURF, Pointer to the first surface phase—The first surface phase is the second phase overall, i.e., NFSURF=2. (Since the gas phase is *always* the first phase, the first surface phase will always be the second phase overall. However, there can be problems in which there are no surface phases, only bulk phases. In that case NFSURF=0.)

NLSURF, Pointer to the last surface phase—Here the last surface phase is the third phase overall. Thus NLSURF=3. (If a problem should have no surface phases then NLSURF=0.)

NNBULK, Number of bulk phases—Here there are five bulk phase, i.e., NNBULK=5.

NFBULK, Pointer to the first bulk phase—Here the first bulk phase is the fourth phase overall, i.e., NFBULK=4. (In a problem that has no bulk phases NFBULK=0.)

NLBULK, Pointer to the last bulk phase—In this problem, NLBULK=8. (In a problem that has no bulk phases, NLBULK=0.)

IISUR, Number of surface reaction—Here there are nine surface reactions, so IISUR=9.

KKPHAS, Array of total number of species in each phase—In this example KKPHAS is an array of length 8, the total number of phases. The values of each element in the array indicate the number of species in the corresponding phase. Here, KKPHAS(1,...,8)=(14, 3, 4, 2, 3, 1, 1, 1,), which shows that the gas phase has 14 species, the first surface phase has 3, the second surface phase has 4, and so on.

KFIRST, Array of starting species numbers of the phases—In this example KFIRST is an array of length 8, the total number of phases. The values of each element in the array indicate the species number of the first species in the corresponding phase. Here, KFIRST(1,...,8) = (1, 15, 18, 22, 24, 27, 28, 29), which shows that in a complete list of species, the first gas-phase species is number 1, the first species in the first surface phase is number 15, the first species in the second surface phase is number 18, and so on.

KLAST, Ending species numbers for the phases—In this example KLAST is an array of length 8, the total number of phases. The values of each element in the array indicate the species number of the last species in the corresponding phase. Here, KLAST(1,...,8)=(14, 17, 21, 23, 26, 27, 28, 29), which shows

that in a complete list of species, the last gas-phase species number is 14, the last species in the first surface phase is number 17, the last species in the second surface phase is number 21, and so on.

Thermodynamic properties, molecular weights, and reaction rates, for example, have values for all species regardless of the phase in which they exist. In these cases the values are stored in long arrays that correspond to the data structure shown in Fig. 15. Other sets of variables or parameters do not exist for all species. For example, we may need the mass densities for the bulk species, but mass density makes no sense for surface species. Nevertheless, we maintain the same data structure. For example, the mass densities for all species can be retrieved by calling

SUBROUTINE SKDEN (P, T, ACT, SDEN, ISKWRK, RSKWRK, DEN).

Here, the input is the gas-phase pressure, the temperature, and the species activities (mole fractions for the gas phase, site fractions for the surface species, and activities for the bulk species) and the site densities of each site phase. The first segment of output array DEN contains the gas-phase species densities (in grams of species k per cm^3). The array elements corresponding to the surface species have no physical meaning and are set to the species molar densities (in moles / cm^2). The final elements of the DEN array contain the mass densities of the bulk species (in grams of species k per cm^3). If the user does not specify a mass density as input to the Surface Interpreter, then these are given a value of -1.

All the species can be identified by a character-string name, and the user can also optionally specify character-string names for the phases (and each "material"). The species names are those that were given as input to the Interpreters, and the application code can retrieve the names by calling the following subroutine:

SUBROUTINE SKSYMS (ISKWRK, CSKWRK, LOUT, KNAME, KERR)

The input is the integer and character working space ISKWRK and CSKWRK and LOUT (a file number on which to write diagnostic messages). The output is an array of character names KNAME and a logical error flag KERR. In the example we are considering here, the species names are shown in Fig. 15. As concrete illustrations,

KNAME(1) = "ASH3",
KNAME(KLAST(NLSURF)) = "GA(L)", or
KNAME(KFIRST(7)) = "GA(B)".

The phase names are optional and can be specified in the SURFACE CHEMKIN-III Interpreter input. The names can be retrieved from an application code by the following subroutine call:

SUBROUTINE SKSYMP (ISKWRK, CSKWRK, LOUT, PNAM, KERR)

Here the output consists of character-string names for the phases. The first phase is always the gas and its name is PNAM(1) = "GAS". In the present example, PNAME(2) = "PLANE", PNAME(NFBULK) = "GA-RICH", and so on. If the user does not specify phase names, default names are supplied. Thus in this example, PNAME(8) = "BULK5".

Mechanisms with Multiple Materials

In some reacting flow simulations there may be different types of solid surfaces at which very distinct heterogeneous chemistry takes place. For example, in a plasma etching reactor there might be a wafer composed of silicon and reactor side-walls composed of aluminum. Although the main purpose of the simulation might be to understand the etching chemistry at the wafer, reactions at the walls also serve to create and destroy species. Because chemical reaction mechanisms occurring at the surfaces can be quite different, SURFACE CHEMKIN-III allows the use of "multiple materials" in Interpreter input files. This essentially allows the user to specify several different and complete surface reaction mechanisms, one after another (separated by a MATERIAL line). An example of such an Interpreter input file is shown in Fig. 16.*

It should be pointed out that the numbering of surface and bulk species, phases, etc. proceeds independently for each material. This is best illustrated using the example mechanism in Fig. 16. In the spirit of the previous section, we list some of the relevant indices and counters for this input file.

For material 1: Material name=MATNAM(1)='WAFER,' NELEM(1)=4, KKGAS(1)=6, KKSUR(1)=4, KKBULK(1)=1, KKTOT(1)=11, NNPHAS(1)=3, NNSURF(1)=1, NFSURF(1)=2, NLSURF(1)=2, NNBULK(1)=1, NFBULK(1)=3, NLBULK(1)=3, IISUR(1)=4, KKPHAS(*,1)=(6,4,1), KFIRST(*,1)=(1,7,11), KLAST(*,1)=(6,10,11), KNAME(*,1)=(E,CL2+,CL+,SICL4,SICL2,CL,SI(S),SICL(S),SICL2(S),SICL3(S), SI(B)), PNAM(*,1)=(GAS,POLY,BULK1)

* The mechanism has been contrived to illustrate a variety of points about the data structures and should not be taken as a source of data for silicon or aluminum processes.

For material 2: Material name=MATNAM(2)='WALL,' NELEM(2)=4, KKGAS(2)=6, KKSUR(2)=2, KKBULK(2)=0, KKTOT(2)=8, NNPHAS(2)=2, NNSURF(2)=1, NFSURF(2)=2, NLSURF(2)=2, NNBULK(2)=0, NFBULK(2)=0, NLBULK(2)=0, IISUR(2)=2, KKPHAS(*,2)=(6,2), KFIRST(*,2)=(1,7), KLAST(*,2)=(6,8), KNAME(*,2)=(E,CL2+,CL+,SICL4,SICL2,CL,AL(S),ALCL(S)), PNAM(*,2)=(GAS, METAL)

A number of changes must be made in an application code to accommodate multiple materials. For example, instead of storing a simple integer KKSURF for the number of surface species, the code must store an integer array KKSURF(NMAT) dimensioned at least the number of materials specified in the input file. An array like KFIRST(N) becomes a two-dimensional array KFIRST(N,NMAT) with the first dimension large enough to hold the maximum number of phases in a given material, and the second dimension large enough to hold the number of materials input to the problem.

Sample logic that might appear in an application code to handle multiple materials is given in Fig. 17. Typically the application code would store the data space needed for all of the different materials in three long real, integer, and character arrays, with pointers to the starting locations of the work space particular to each material (e.g., the arrays IMRSK(N), IMISK(N), and IMCSK(N) point to the starting locations for material N in the example shown in Fig. 17). After such information is set up, calls to most SURFACE CHEMKIN-III subroutines look like before, but use off-sets into the work arrays for the material of interest. For example to get an array of the molecular weights of all species in the mechanism for material NMAT, the call would look like

CALL SKWT (I(IMISK(NMAT)), R(IMRSK(NMAT)), WT(1,NMAT)).

```
!------------------------------CHEMKIN-III INTERPRETER INPUT---------------------
ELEMENTS  SI CL E AL
SPECIES   E CL2+ CL+ SICL4 SICL2 CL

!--------------------------SURFACE CHEMKIN-III INTERPRETER INPUT-----------------
MATERIAL WAFER
SITE/POLY/ SDEN/2.25e-9/
 SI(S)  SICL(S) SICL2(S) SICL3(S)
END
BULK SI(B)/2.33/
REACTIONS  MWOFF
    CL + SI(S)    => SICL(S)                          1.0   0.0  0.0
       STICK
    E + CL2+ + 2SI(S) => 2SICL(S)                     0.4   0.0  0.0
       BOHM
    E + CL+ + SICL3(S) + SI(B) => SICL4 + SI(S)        0.50  0.0  0.0
       BOHM
       ENRGDEP/1. 0.5 1.0/  UNITS/EVOLT/
    E + CL+ + #SICL3(S) + #SI(B) + SICL(S) &
          => SICL2(S) + #SICL2 + #SICL(S)             0.50  0.0  0.0
       BOHM
       YIELD/0.0712 1.21 0.5 1.0/  UNITS/EVOLT/
!           /A   Eth[eV] a b /   for #=A(Ei^a-Eth^a)^b
END
MATERIAL WALL
SITE/METAL/ SDEN/2.25E-9/
 AL(S) ALCL(S)
END
REACTIONS  MWOFF
    CL+ + E       => CL                   0.6   0.0  0.0
       BOHM
    CL + AL(S)    => ALCL(S)              1.0   0.0  0.0
       STICK
END
```

Figure 16 Interpreter Input File using Multiple Materials

```
C  SET UP COUNTER AND LOOP OVER MULTIPLE MATERIALS IN SK LINK FILE
C
      NMAT = 0
      LENRSK = 0
      LENISK = 0
      LENCSK = 0
C
10    CONTINUE
C
      NMAT = NMAT + 1
      LASTRSK = LENRSK
      LASTISK = LENISK
      LASTCSK = LENCSK
      CALL SKLEN (LINKSK, LOUT, LENISK, LENRSK, LENCSK, IFLAG)
      IF (IFLAG .NE. 0) THEN
         WRITE (LOUT,9005) IFLAG
         STOP
      ENDIF
C
      IF (NMAT .EQ. 1) THEN
         IMRSK(1) = 1
         IMISK(1) = 1
         IMCSK(1) = 1
C  REAL WORK SPACE ARRAYS
         NSKW = 1
         NTOT = NSKW + LENRSK
C  INTEGER WORK SPACE ARRAYS
         ISKW = 1
         ITOT = ISKW + LENISK
C  CHARACTER WORK SPACE ARRAYS
         ICS  = 1
         ICTOT = ICS + LENCSK
C
C              INITIALIZE CHEMKIN AND SURFACE CHEMKIN
C
         IF (ITOT.LT.LENIWK .AND. NTOT.LT.LENRWK .AND.
     1      ICTOT.LT.LENCWK) THEN
            CALL CKINIT (LENICK, LENRCK, LENCCK, LINKCK, LOUT, I, R, C,
     1                   IFLAG)
            CALL CKINDX (I, R, MM, KKGAS, II, NFIT)
            CALL SKINIT (LENISK, LENRSK, LENCSK, LINKSK, LOUT,
     1                   I(ISKW), R(NSKW), C(ICS), IFLAG)
            CALL SKINDX (I(ISKW),MM, KKGAS, KKSURF(1), KKBULK(1),
     1                   KK(1), NPHASE(1), NNSURF(1), NFSURF(1),
     2                   NLSURF(1), NNBULK(1), NFBULK(1), NLBULK(1),
     3                   IISUR(1))
         ELSE
            WRITE (LOUT, *)'NOT ENOUGH WORK SPACE ALLOCATED TO START'
            RETURN
         ENDIF
         KKTOT = KK(1)
      ELSE
         IMRSK(NMAT) = IMRSK(NMAT-1) + LASTRSK
         IMISK(NMAT) = IMISK(NMAT-1) + LASTISK
         IMCSK(NMAT) = IMCSK(NMAT-1) + LASTCSK
         NTOT = NTOT + LENRSK
         ITOT = ITOT + LENISK
         ICTOT = ICTOT + LENCSK
```

Figure 17 Sample Program Statements to Handle Multiple Materials

```
C
C              INITIALIZE CHEMKIN AND SURFACE CHEMKIN
C
         IF (ITOT.LT.LENIWK .AND. NTOT.LT.LENRWK .AND.
     1      ICTOT.LT.LENCWK) THEN
            CALL SKINIT (LENISK, LENRSK, LENCSK, LINKSK, LOUT,
     1                   I(ISKW+IMISK(NMAT)-1), R(NSKW+IMRSK(NMAT)-1),
     2                   C(ICS+IMCSK(NMAT)-1), IFLAG)
            CALL SKINDX (I(ISKW+IMISK(NMAT)-1),MM, KKGAS, KKSURF(NMAT),
     1                   KKBULK(NMAT), KK(NMAT), NPHASE(NMAT),
     2                   NNSURF(NMAT), NFSURF(NMAT), NLSURF(NMAT),
     3                   NNBULK(NMAT), NFBULK(NMAT), NLBULK(NMAT),
     4                   IISUR(NMAT))
         ELSE
            WRITE (LOUT, *)
     1          'NOT ENOUGH WORK SPACE FOR SURFACE MATERIAL ', NMAT
            STOP
         ENDIF
         KKTOT = KKTOT + KKSURF(NMAT) + KKBULK(NMAT)
       ENDIF
C
C   CHECK END OF WORKSPACE FOR FLAG INDICATING ANOTHER MATERIAL
C
       MORE = I(ISKW + IMISK(NMAT) - 1 + LENISK - 1)
       IF (MORE .EQ. 1) GO TO 10
```

Figure 17 Sample Program Statements to Handle Multiple Materials (cont.)

VII. QUICK REFERENCE TO THE SURFACE SUBROUTINE LIBRARY

This chapter is arranged by topical area to provide a quick reference to each of the Surface Library Subroutines.* In addition to the subroutine call list itself, the purpose of the subroutine is briefly described.

Mnemonics

There are some rules of thumb for explaining the subroutine naming conventions. All subroutine names begin with the letters SK so that SURFACE CHEMKIN-III Subroutines are easily recognized and so that they are likely different from any user subroutine names. The four remaining letters generally identify the purpose of the subroutine.

Thermodynamic properties are referred to by CP (specific heat), H (enthalpy), S (entropy), U (internal energy), G (Gibbs free energy), and A (Helmholtz free energy). The thermodynamic property subroutines may be called to return properties in mass units, denoted by MS or S as the last letter(s), or in molar units, denoted by ML or L as the last letter(s).

The mnemonics for the variable names in the subroutine call lists are roughly the same as for the subroutine names. However, because six letters can be used (only four are available in the subroutine names because SK occupies two), the mnemonics can be more explicit.

1. Initialization

```
SUBROUTINE SKINDX (ISKWRK, NELM, KKGAS, KKSUR, KKBULK, KKTOT,
                   NNPHAS, NNSURF, NFSURF, NLSURF, NNBULK, NFBULK,
                   NLBULK, IISUR)
Returns a group of indices defining the size of the surface
reaction mechanism.

SUBROUTINE SKINIT (LENISK, LENRSK, LENCSK, LINSK, LOUT,
                   ISKWRK, RSKWRK, CSKWRK, IFLAG)
Reads the surface linkfile and creates internal work arrays ISKWRK,
RSKWRK, and CSKWRK. SKINIT must be called before any other SURFACE
CHEMKIN-III subroutine can be used, as the work arrays must be available
as their input.

SUBROUTINE SKLEN  (LINSK, LOUT, LENI, LENR, LENC, IFLAG)
Reads the first record of the linkfile to return the lengths
required for the integer, real, and character work arrays.
```

* In the following list, subroutines whose arguments lists have changed as of SURFACE CHEMKIN-III are underlined. Subroutine names appearing in bold font are new with this version of the Library.

2. Information about Elements

SUBROUTINE SKSYME (ISKWRK, CSKWRK, LOUT, ENAM, KERR)
Returns a character string array of element names.

3. Information about Species

SUBROUTINE SKATCZ (P, T, ACT, SDEN, ISKWRK, RSKWRK, CZ)
Returns the concentrations of the species, given the pressure, temperature and activities.

SUBROUTINE SKCHRG (ISKWRK, RSKWRK, KCHARG)
Returns an array containing electronic charges of the species.

SUBROUTINE SKCOV (ISKWRK, KOCC)
Returns an array of site occupancy numbers for the species.

SUBROUTINE **SKCZTA** (T, CZ, SDEN, ISKWRK, RSKWRK, ACT)
Returns the activities of the species, given the pressure, temperature and concentrations.

SUBROUTINE SKDEN (P, T, ACT, SDEN, ISKWRK, RSKWRK, DEN)
Returns a real array of species densities.

SUBROUTINE **SKKTFL** (ISKWRK, KTFL)
Allows the user to assign a location in the temperature array to use for the gas-phase species.

SUBROUTINE SKNCF (NELDIM, ISKWRK, NEL)
Returns the elemental composition of the species.

SUBROUTINE SKSYMS (ISKWRK, CSKWRK, LOUT, KNAM, KERR)
Returns a character array of species names.

SUBROUTINE SKWT (ISKWRK, RSKWRK, WT)
Returns the molecular weights of the species.

4. Information about Phases and Materials

SUBROUTINE SKPKK (ISKWRK, KKPHAS, KFIRST, KLAST)
Returns arrays of species pointers for the phases.

SUBROUTINE SKSDEN (ISKWRK, RSKWRK, SDEN0)
Returns a real array of standard-state phase densities as given on input to the interpreter.

SUBROUTINE **SKSYMM** (ISKWRK, CSKWRK, LOUT, MATNAM, KERR)
Returns the character string name of a material.

SUBROUTINE SKSYMP (ISKWRK, CSKWRK, LOUT, PNAM, KERR)
Returns a character string array of phase names.

5. Information about Surface Reactions

SUBROUTINE SKABE (ISKWRK, RSKWRK, RA, RB, RE, ISTFL)
Returns the Arrhenius coefficients or the sticking coefficients of the surface reactions, and integer flags to indicate the type of the coefficients.

SUBROUTINE **SKFLGS** (IR, ISKWRK, NRPP, IREV, ISTFL, ICOV, IMOTZ,
IEDP, IBHM, IORD, IYLD)

Returns several integer flags describing surface reaction IR.

SUBROUTINE **SKIBHM** (IR, ISKWRK, IBMFL)
Returns an integer flag to indicate whether reaction IR uses BOHM sticking coefficients.

SUBROUTINE SKICOV (IR, NDIM, ISKWRK, RSKWRK, NCOVI, KCOVI, CPARI)
Returns the coverage species index numbers and their coverage parameters for reaction IR.

SUBROUTINE **SKIENR** (IR, ISKWRK, SKWRK, IENRFL, IEION, PEDEP)
Returns an integer flag to indicate if reaction IR is ion-energy-dependent, and if so, formulation-specific parameters.

SUBROUTINE **SKINU** (IS, NDIM, ISKWRK, RSKWRK, NSPEC, KI, NU)
Returns the number of species in a surface reaction, and the species indices and stoichiometric coefficients.

SUBROUTINE **SKIORD** (IDIM, KDIM, ISKWRK, RSKWRK, NFORD, IFORD, FORD,
NRORD, IRORD, RORD)
Returns the number and indices of surface reactions with modified species orders, and the order values for the species in the surface mechanism.

SUBROUTINE **SKIREV** (IR, ISKWRK, RSKWRK, IREV, RAR, RBR, RER)
Returns an integer flag to indicate whether reaction IR has an explicitly assigned reverse rate constant. It also returns the reverse Arrhenius expression values for surface reaction IR, if it was explicitly assigned in the SURFACE CHEMKIN-III interpreter. If reverse Arrhenius values were not explicitly assigned, RAR, RBR and RER will be zero.

SUBROUTINE **SKIRNU** (IDIM, NDIM, ISKWRK, RSKWRK, NIRNU, IRNU, NSPEC,
KI, RNU)
Returns the number and indices of surface reactions with real stoichiometric coefficients, number of species in the reactions, and the species indices and coefficients;

SUBROUTINE SKISTK (IR, ISKWRK, ISTFL)
Returns an integer flag to indicate whether reaction IR uses sticking coefficients.

SUBROUTINE **SKIYLD** (IR, ISKWRK, RSKWRK, NYDIM, IYLD, IYION, KYLD, PYLD)
Returns an integer flag to indicate whether reaction IR has yield-modified species, the species index of its ion, yield-modify flags for its reactants and products, and parameters for the yield expression.

SUBROUTINE **SKKION** (ISKWRK, KELECT, KKION, KION)
Returns the species number of the electron, the number of positive ions in the gas phase, and an array of species number for each positive ion

SUBROUTINE SKNCON (ISKWRK, RSKWRK, NCON)
Returns the total number of surface reactions which do not conserve sites of the phases.

SUBROUTINE SKNU (IDIM, ISKWRK, RSKWRK, KSTOIC, NSTOIC)
Returns the stoichiometric coefficients of the species and the net change in phases for all of the surface reactions in a mechanism.

SUBROUTINE **SKNUF** (IDIM, ISKWRK, KSTOIF)
Returns the stoichiometric coefficients of the species
for all reactants in all surface reactions in a mechanism.
(note - reactants only! - they will all be negative)

SUBROUTINE SKRAEX (IR, ISKWRK, RSKWRK, RA)
Returns the Pre-exponential rate constant
(or sticking coefficient) of the IRth reaction, or changes its
value, depending on the sign of IR.

SUBROUTINE **SKRDEX** (IR, ISKWRK, RSKWRK, RD)
Returns the perturbation factor of the IRth reaction,
or changes its value, depending on the sign of IR.

SUBROUTINE **SKRPAR** (ISKWRK, RSKWRK, ENRGI)
Allows the user to input auxiliary reaction-rate parameters for
special types of reactions. The first parameter is the species (ion)
directed energy for ion-energy-dependent reactions.

SUBROUTINE SKSYMR (IR, LOUT, ISKWRK, RSKWRK, CSKWRK, LT, RNAM, KERR)
Returns the character string representation of reaction IR.

6. Gas Constants and Units

SUBROUTINE SKRP (ISKWRK, RSKWRK, RU, RUC, PATM)
Returns universal gas constants and the pressure of one standard
atmosphere.

7. Thermodynamic Properties (Non dimensional)

SUBROUTINE SKATHM (MDIM, NDIM1, NDIM2, ISKWRK, RSKWRK, NT, TMP, A)
Returns the polynomial coefficients of the fits for
thermodynamic properties of all of the species.

SUBROUTINE SKCPOR (T, ISKWRK, RSKWRK, CPOR)
Returns an array of the non dimensional specific heats at constant
pressure.

SUBROUTINE SKHORT (T, ISKWRK, RSKWRK, HORT)
Returns an array of the non dimensional enthalpies.

SUBROUTINE **SKMXTP** (ISKWRK, MXTP)
Returns the maximum number of temperatures used in
fitting the thermodynamic properties of the species.

SUBROUTINE **SKRHEX** (K, ISKWRK, RSKWRK, A6)
Returns an array of the sixth thermodynamic polynomial
coefficients for a species, or changes their value,
depending on the sign of K.

SUBROUTINE SKSOR (T, ISKWRK, RSKWRK, SOR)
Returns an array of the non dimensional entropies.

8. Thermodynamic Properties (Mass Units)

SUBROUTINE SKAMS (T, ISKWRK, RSKWRK, AMS)
Returns an array of the standard state Helmholtz free energies
in mass units.

```
SUBROUTINE SKCPMS (T, ISKWRK, RSKWRK, CPMS)
Returns an array of the specific heats at constant pressure
in mass units.

SUBROUTINE SKGMS  (T, ISKWRK, RSKWRK, GMS)
Returns an array of the standard state Gibbs free energies
in mass units.

SUBROUTINE SKHMS  (T, ISKWRK, RSKWRK, HMS)
Returns an array of the enthalpies in mass units.

SUBROUTINE SKSMS  (T, ISKWRK, RSKWRK, SMS)
Returns an array of the standard state entropies in mass units.

SUBROUTINE SKUMS  (T, ISKWRK, RSKWRK, UMS)
Returns an array of the internal energies in mass units.
```

9. Thermodynamic Properties (Molar Units)

```
SUBROUTINE SKAML  (T, ISKWRK, RSKWRK, AML)
Returns an array of the standard state Helmholtz free energies
in molar units.

SUBROUTINE SKCPML (T, ISKWRK, RSKWRK, CPML)
Returns an array of the specific heats at constant pressure
in molar units.

SUBROUTINE SKGML  (T, ISKWRK, RSKWRK, GML)
Returns an array of the standard state Gibbs free energies
in molar units.

SUBROUTINE SKHML  (T, ISKWRK, RSKWRK, HML)
Returns an array of the enthalpies in molar units.

SUBROUTINE SKSML  (T, ISKWRK, RSKWRK, SML)
Returns an array of the standard state entropies in molar units.

SUBROUTINE SKUML  (T, ISKWRK, RSKWRK, UML)
Returns an array of the internal energies in molar units.
```

10. Chemical Production Rates

```
SUBROUTINE SKCONT (KSPEC, ROP, ISKWRK, RSKWRK, CIK)
Returns the contributions of the surface reactions to the molar
production rate of species KSPEC.

SUBROUTINE SKDRDA (IR, P, T, ACT, SDEN, ISKWRK, RSKWRK, DKDAI)
Returns the partial of the rates of production of the species with
respect to the pre-exponential constant of surface reaction IR.

SUBROUTINE SKDRDC (KSPEC, P, T, ACT, SDEN, ISKWRK, RSKWRK, DKDC)
Returns the partial derivative of the production rates of the
species with respect to the concentration of species KSPEC.

SUBROUTINE SKDSDC (P, T, X, ACT, SDEN, ISKWRK, RSKWRK, DSDC, KKTOT,
                   SDOT, SITDOT)
Returns the partial derivative of the production rates of the
species with respect to the concentration of each species.
It also returns the matching production rates.
```

```
SUBROUTINE SKDSDX (P, T, X, ACT, SDEN, ISKWRK, RSKWRK, DSDX, KKTOT,
                   SDOT, SITDOT)
Returns the partial derivative of the production rates of the
species with respect to the activity for each species.
It also returns the matching production rates.

SUBROUTINE SKRAT  (P, T, ACT, SDEN, ISKWRK, RSKWRK, SDOT, SITDOT)
Returns production rates for the species and sites.

SUBROUTINE SKRATI (IR, ROP, ISKWRK, RSKWRK, SDOTI, SITDTI)
Returns rates of production of the species by surface reaction IR.
```

11. Equilibrium Constants and Rate-of-Progress Variables

```
SUBROUTINE SKEQ    (P, T, ACT, SDEN, ISKWRK, RSKWRK, EQKC)
Returns the equilibrium constants for the surface reactions given
pressure, temperature, species activities, and the site densities.

SUBROUTINE SKROP  (P, T, ACT, SDEN, ISKWRK, RSKWRK, ROP)
Returns rates of progress for the surface reactions.
```

12. Utilities

```
SUBROUTINE SKCOMP (ISTR, IRAY, NN, IND, NT)
Search for the occurrence of character string ISTR, in the NN
character strings of array IRAY;
IND is the first location in IRAY of ISTR if found, or 0 if not
found, and NT is the total number of times it occurs.
   Consider the following example,
      IRAY = {"BOOK","BLUE","BEAR","BOOK"}
      NN=4.
   If ISTR="BLUE" then IND=2 and NT=1;
   if ISTR="RED"  then IND=0 and NT=0; and
   if ISTR="BOOK",then IND=1 and NT=2.

SUBROUTINE SKPCMP (ISTR, IRAY, NN, SETS, NSETS, ISET, IND, NT)
This subroutine can do everything that the subroutine SKCOMP can do,
and additionally, has the capabilities of separating the elements of
IRAY into categories and then search IRAY by element and category.
The categories that each element of IRAY will be assigned to are
specified by the input character string vector SETS of vector length
NSETS. Elements of each category in IRAY must be grouped congruously.
The number of elements in each category within IRAY is specified by
the input integer vector ISET. To search for the existence of an
element within category ISTR may additionally be composed of two
substrings, ISTR="ELEMENT_NAME/CATEGORY_NAME/", where CATEGORY_NAME
is one of the categories specified in SETS. In this case, IND will
return the first position in IRAY where ELEMENT_NAME occurred within
the category CATEGORY_NAME. NT will return the total number of
times ELEMENT_NAME occurred within the category CATEGORY_NAME.
If ELEMENT_NAME is not found within the specified category, IND and
NT are returned with a value of zero. If no category is specified
within ISTR, IND and NT return with the same values as they would
from subroutine SKCOMP.
Consider the following example,
      IRAY = {"RED", "BLUE", "JADE", "RUBY", "TOPAZ", "JADE"}
      NN = 6
      SETS = {"COLORS", "STONES"},
      NSETS = 2
      ISET = {4, 2}.
This assumes that the elements of IRAY were grouped into two
```

```
sets, consisting of 4 and 2 elements, respectively, and the
following names
      "COLORS"  = {"RED", "BLUE", "JADE", "RUBY"}, and
      "STONES"  = {"TOPAZ", "JADE"}.
If ISTR="BLUE" then IND=2 and NT=1;
if ISTR="PINK" then IND=0 and NT=0; and
if ISTR="JADE",then IND=3 and NT=2.
If ISTR="BLUE/COLORS/" then IND=2 and NT=1;
if ISTR="BLUE/STONES/" then IND=0 and NT=0;
if ISTR="JADE/GEMS/"   then IND=0 and NT=0; and
if ISTR="JADE/STONES/",then IND=6 and NT=1.
```

SUBROUTINE SKPNT (LSAVE, LOUT, VERS, PREC, LENI, LENR, LENC, IERR)
Reads from a file information about a SURFACE CHEMKIN-III linkfile, pointers for the SURFACE CHEMKIN-III Library, and returns lengths of work arrays.

SUBROUTINE SKSNUM (LINE, NEXP, LOUT, KNAM, KKTOT, PNAM, NNPHAS, KKPHAS, KNUM, NT, NVAL, RVAL, KERR)

This subroutine is used to read a format-free input line of combined alphanumeric data. It can be used to parse an input character string, LINE, which may be composed of several blank-delimited substrings. This subroutine assumes that the first substring in LINE is the name of a species in the SURFACE CHEMKIN-III mechanism. If the species name is not unique within the Surface CHEMKIN-III mechanism, the phase of the species should be input immediately after the species name, delimited by slashes.
Upon return from the subroutine, KNUM returns the index position of the species within the SURFACE CHEMKIN-III linkfile. If the species name is not unique, KNUM returns the first position and NT returns the number of the times the species occurs within the linkfile. If the species name is not found, or there is a syntax error, on return, KNUM=0, NT=0, and KERR=.TRUE.
The substrings in LINE following the first are expected to represent numbers. They are converted into floating point values and stored in the output vector, RVAL(*). Upon input, NEXP is equal to the number of values expected to be found. If NEXP numbers are not found, KERR will be set to .TRUE. on return from the subroutine.

Example input:

LINE	= GA(S)/BULK1/ 1.2
NEXP	= 1, the number of values expected
LOUT	= 6, a logical unit number on which to write diagnostic messages
KNAM(*)	= Array of character species names
KKTOT	= Total number of species
PNAM(*)	= Array of character phase names
NNPHAS	= Total number of phases
KKPHAS(*)	= Index array of the number of species in the phases

Output:

KNUM	= The index number of the species which has the name "GA(S)" and resides in phase "BULK1"
NT	= 1, if there is only one species GA(S) in phase BULK1
NVAL	= 1, the number of values found in LINE following the species name
RVAL(1)	= 1.200E+00, the substring converted to a real number
KERR	= .FALSE.

SUBROUTINE SKSAVE (LOUT, LSAVE, ISKWRK, RSKWRK, CSKWRK)
Writes to a binary file information about a SURFACE CHEMKIN-III linkfile, pointers for the SURFACE CHEMKIN-III Library, and SURFACE CHEMKIN-III work arrays.

VIII. ALPHABETICAL LISTING OF THE SURFACE SUBROUTINE LIBRARY WITH DETAILED DESCRIPTIONS OF THE CALL LISTS

Each subroutine in the Surface Subroutine Library is described in this chapter, together with a detailed description of the variables in the call lists. For all arrays, information is given on the required dimensioning in the calling program. For all variables having units, the cgs units are stated.

```
SKABE     SKABE     SKABE     SKABE     SKABE     SKABE     SKABE
          ******************************************
                    *************************
                              ******

SUBROUTINE SKABE  (ISKWRK, RSKWRK, RA, RB, RE, ISTFL)
Returns the Arrhenius coefficients or the sticking coefficients
of the surface reactions, and integer flags to indicate the type
of the coefficients.

INPUT
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real workspace array;    dimension at least LENRSK.

OUTPUT
RA(*)     - Real array, pre-exponential constants for reactions;
            dimension at least IISUR, the total surface reaction
            count.
               cgs units, mole-cm-sec-K
RB(*)     - Real array, temperature dependence exponents for
            reactions;
            dimension at least IISUR, the total surface reaction
            count.
RE(*)     - Real array, activation energies for reactions;
            dimension at least IISUR, the total surface reaction
            count.
               cgs units, K
ISTFL(*)  - Integer array, sticking reaction information;
            dimension at least IISUR, the total surface reaction
            count.
            =1, a reaction uses sticking coefficients.
            =0, a rection does not.
```

```
SKAML     SKAML     SKAML     SKAML     SKAML     SKAML     SKAML
          ****************************************
                    *************************
                              ******

SUBROUTINE SKAML  (T, ISKWRK, RSKWRK, AML)
Returns an array of the standard state Helmholtz free energies
in molar units.

INPUT
T(*)      - Real array, temperature(s); dimension is determined by
            the application program to be the total number of
            species temperatures, nominally 1.
               cgs units, K
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real    workspace array; dimension at least LENRSK.

OUTPUT
AML(*)    - Real array, standard state Helmholtz free energies
            for species;
            dimension at least KKTOT, the total species count.
               cgs units, ergs/mole



SKAMS     SKAMS     SKAMS     SKAMS     SKAMS     SKAMS     SKAMS
          ****************************************
                    *************************
                              ******

SUBROUTINE SKAMS  (T, ISKWRK, RSKWRK, AMS)
Returns an array of the standard state Helmholtz free energies
in mas units.

INPUT
T(*)      - Real array, temperature(s); dimension is determined by
            the application program to be the total number of
            species temperatures, nominally 1.
               cgs units, K
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real    workspace array; dimension at least LENRSK.

OUTPUT
AMS(*)    - Real array, standard state Helmholtz free energies
            for species;
            dimension at least KKTOT, the total species count.
               cgs units, ergs/gm
```

```
SKATCZ    SKATCZ    SKATCZ    SKATCZ    SKATCZ    SKATCZ    SKATCZ
          ****************************************
                    *************************
                              ******

SUBROUTINE SKATCZ (P, T, ACT, SDEN, ISKWRK, RSKWRK, CZ)
Returns the concentrations of the species, given the pressure,
temperature and activities.

INPUT
P         - Real scalar, pressure.
               cgs units, dynes/cm**2
T(*)      - Real array, temperature(s); dimension is determined by
            the application program to be the total number of
            species temperatures, nominally 1.
               cgs units, K
ACT(*)    - Real array, activities of the species;
            dimension at least KKTOT, the total species count.
            The first KKGAS activities are mole fractions,
            the next KKSURF activities are site fractions
               (species density normalized by the site density;
                surface concentration in moles/cm**2 is
                ACT(K)*SITE_DENSITY / # sites per species), and
            the next KKBULK activities for bulk phase species
            should be from 0 to 1, and should sum to 1 for each
            phase.
SDEN(*)   - Real array, site densities for the site types;
            dimension at least NPHASE, the total phase count,
            but the subroutine only uses site phase entries,
            NFSURF <= N <= NLSURF.
               cgs units, moles/cm**2.
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real    workspace array; dimension at least LENRSK.

OUTPUT
CZ(*)     - Real array, gas-phase and surface species concentrations,
            and bulk species activities;
            dimension at least KKTOT, the total species count.
            The first KKGAS gas-phase concentrations are moles/cm**3,
            the next KKSURF site concentrations are moles/cm**2, and
            the final KKBULK entries are bulk species activities.
```

SKATHM SKATHM SKATHM SKATHM SKATHM SKATHM SKATHM

SUBROUTINE SKATHM (MDIM, NDIM1, NDIM2, ISKWRK, RSKWRK, NT, TMP, A)

Returns the polynomial coefficients of the fits for thermodynamic properties of all of the species.

INPUT

MDIM - Integer scalar, first dimension of an array of temperatures used in thermodynamic fits for species; MDIM must be at least MAXTP, the maximum number of temperatures used to fit the thermodynamics.

NDIM1 - Integer scalar, first dimension of A, the three-dimensional array of thermodynamic fit coefficients; NDIM1 must be at least NPCP2, the total number of coefficients for one temperature range.

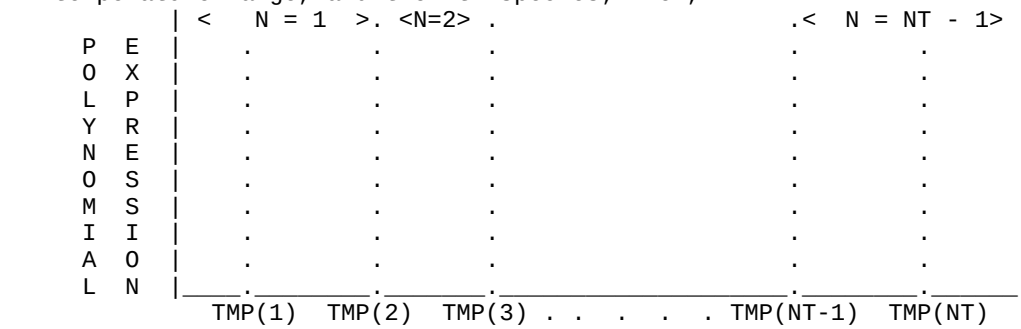
NDIM2 - Integer scalar, second dimension of A; NDIM2 must be at least MAXTP-1, the total number of temperature ranges.

ISKWRK(*) - Integer workspace array; dimension at least LENISK.

RSKWRK(*) - Real workspace array; dimension at least LENRSK.

OUTPUT

Where NT(K) is the number of temperatures used in fitting the thermodynamic properties of species K, TMP(N) is the Nth temperature, NT(K)-1 is the number of temperature ranges for which the polynomial coefficients are valid, then A (L, N, K) is the Lth polynomial coefficient, for the Nth temperature range, and the Kth species; i.e.,



NT(*) - Integer array, total number of temperatures used in fitting coefficients of thermodynamic properties for the species;
dimension at least KKTOT, the total species count.

TMP(*,*) - Real matrix, temperatures for dividing the thermodynamic fits for species; dimension at least MAXTP for the first, and at least KKTOT for the second, the total species count.
cgs units, K

A(*,*,*) - Real three-dimensioned array of fit coefficients to the thermodynamic data for species;
dimension exactly NPCP2 for the first, exactly MAXTP-1 for the second, and at least KKTOT for the third, the total species count.
The indicies in A(N,L,K) mean-
N = 1,NN represent polynomial coefficients in CP/R
CP/R(K)=A(1,L,K) + A(2,L,K)*T + A(3,L,K)*T**2 + ...
N = NN+1 is for the formation enthalpies, i.e.,
HO/R = A(NN+1,L,K)
N = NN+2 is for the formation entropies, i.e.,
SO/R = A(NN+2,L,K)

```
            L = 1 is for temperature <= TMP(2,K)
            L = 2 is for TMP(2,K) < temperature <= TMP(3)
              :
            L = (NTMP-1) is for TMP(NTMP-1) <= temperature;
            K  is  the  species index
```

```
SKCHRG    SKCHRG    SKCHRG    SKCHRG    SKCHRG    SKCHRG    SKCHRG
          ******************************************
                    *************************
                              ******
```

```
SUBROUTINE SKCHRG (ISKWRK, RSKWRK, KCHARG)
Returns an array containing electronic charges of the species.

INPUT
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real    workspace array; dimension at least LENRSK.

OUTPUT
KCHARG(*) - Integer array, electronic charges of the species;
            dimension at least KKTOT, the total species count.
            KCHARG(K)=-2 indicates that the species K has two excess
            electrons.
```

```
SKCOMP    SKCOMP    SKCOMP    SKCOMP    SKCOMP    SKCOMP    SKCOMP
          ******************************************
                    *************************
                              ******
```

```
SUBROUTINE SKCOMP (ISTR, IRAY, NN, IND, NT)
Search for the occurrence of character string ISTR, in the NN
character strings of array IRAY;
IND is the first location in IRAY of ISTR if found, or 0 if not
found, and NT is the total number of times it occurs.
   Consider the following example,
      IRAY = {"BOOK","BLUE","BEAR","BOOK"}
      NN=4.
   If ISTR="BLUE" then IND=2 and NT=1;
   if ISTR="RED"  then IND=0 and NT=0; and
   if ISTR="BOOK",then IND=1 and NT=2.

INPUT
ISTR      - Character string.
IRAY(*)   - Character string array.
NN        - Integer scalar, length of IRAY(*).

OUTPUT
IND       - Integer scalar, location in IRAY of the character string
            ISTR, or 0 if ISTR does not appear in IRAY.
NT        - Integer scalar, total number of times ISTR occurs
            in IRAY.
```

```
SKCONT    SKCONT    SKCONT    SKCONT    SKCONT    SKCONT    SKCONT
          ****************************************
                    *************************
                              ******

SUBROUTINE SKCONT (KSPEC, ROP, ISKWRK, RSKWRK, CIK)
Returns the contributions of the surface reactions to the molar
production rate of species KSPEC.

INPUT
KSPEC     - Integer scalar, species index.
ROP(*)    - Real array, rates of progress for the surface reactions;
            dimension at least IISUR, the total surface reaction
            count.
               cgs units, moles/(cm**2*sec)
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real    workspace array; dimension at least LENRSK.

OUTPUT
CIK(*)    - Real array, contributions of the surface reactions to the
            production rate of species KSPEC;
            dimension at least IISUR, the total surface reaction
            count.
               cgs units, mole/(cm**2*sec)



SKCOV     SKCOV     SKCOV     SKCOV     SKCOV     SKCOV     SKCOV
          ****************************************
                    *************************
                              ******

SUBROUTINE SKCOV  (ISKWRK, KOCC)
Returns an array of site occupancy numbers for the species.

INPUT
ISKWRK(*) - Integer workspace array; dimension at least LENISK.

OUTPUT
KOCC(*)   - Integer array, site occupancy numbers for the species;
            dimension at least KKTOT, the total species count.
```

```
SKCPML    SKCPML    SKCPML    SKCPML    SKCPML    SKCPML    SKCPML
          ****************************************
                    *************************
                              ******

SUBROUTINE SKCPML (T, ISKWRK, RSKWRK, CPML)
Returns an array of the specific heats at constant pressure
in molar units.

INPUT
T(*)      - Real array, temperature(s); dimension is determined by
            the application program to be the total number of
            species temperatures, nominally 1.
               cgs units, K
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real    workspace array; dimension at least LENRSK.

OUTPUT
CPML(*)   - Real array, specific heats at constant pressure for the
            species;
            dimension at least KKTOT, the total species count.
               cgs units, ergs/(mole*K)



SKCPMS    SKCPMS    SKCPMS    SKCPMS    SKCPMS    SKCPMS    SKCPMS
          ****************************************
                    *************************
                              ******

SUBROUTINE SKCPMS (T, ISKWRK, RSKWRK, CPMS)
Returns an array of the specific heats at constant pressure
in mass units.

INPUT
T(*)      - Real array, temperature(s); dimension is determined by
            the application program to be the total number of
            species temperatures, nominally 1.
               cgs units, K
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real    workspace array; dimension at least LENRSK.

 OUTPUT
 CPMS(*)  - Real array, specific heats at constant pressure for the
            species;
            dimension at least KKTOT, the total species count.
               cgs units, ergs/(gm*K)
```

```
SKCPOR    SKCPOR    SKCPOR    SKCPOR    SKCPOR    SKCPOR    SKCPOR
          ****************************************
                    *************************
                              ******

SUBROUTINE SKCPOR (T, ISKWRK, RSKWRK, CPOR)
Returns an array of the nondimensional specific heats at constant
pressure.

INPUT
T(*)      - Real array, temperature(s); dimension is determined by
            the application program to be the total number of
            species temperatures, nominally 1.
               cgs units, K
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real    workspace array; dimension at least LENRSK.

OUTPUT
CPOR(*)   - Real array, nondimensional specific heats at constant
            pressure for the species;
            dimension at least KKTOT, the total species count.



SKCZTA    SKCZTA    SKCZTA    SKCZTA    SKCZTA    SKCZTA    SKCZTA
          ****************************************
                    *************************
                              ******

SUBROUTINE SKCZTA (T, CZ, SDEN, ISKWRK, RSKWRK, ACT)
Returns the activities of the species, given the pressure,
temperature and concentrations.

INPUT
T(*)      - Real array, temperature(s); dimension is determined by
            the application program to be the total number of
            species temperatures, nominally 1.
               cgs units, K
SDEN(*)   - Real array, site densities for the site types;
            dimension at least NPHASE, the total phase count,
            but the subroutine only uses site phase entries,
            NFSURF <= N <= NLSURF.
               cgs units, moles/cm**2.
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real    workspace array; dimension at least LENRSK.
CZ(*)     - Real array, gas-phase and surface species concentrations,
            and bulk species activities;
            dimension at least KKTOT, the total species count.
            The first KKGAS gas-phase concentrations are moles/cm**3,
            the next KKSURF site concentrations are moles/cm**2, and
            the final KKBULK entries are bulk species activities.

OUTPUT
ACT(*)    - Real array, activities of the species;
            dimension at least KKTOT, the total species count.
            The first KKGAS activities are mole fractions,
            the next KKSURF activities are site fractions
               (species density normalized by the site density;
                surface concentration in moles/cm**2 is
                ACT(K)*SITE_DENSITY / # sites per species), and
            the next KKBULK activities for bulk phase species
            should be from 0 to 1, and should sum to 1 for each
            phase.
```

```
SKDEN     SKDEN     SKDEN     SKDEN     SKDEN     SKDEN     SKDEN
          ****************************************
                    *************************
                              ******

SUBROUTINE SKDEN  (P, T, ACT, SDEN, ISKWRK, RSKWRK, DEN)
Returns a real array of species densities.

INPUT
P         - Real scalar, pressure.
               cgs units, dynes/cm**2
T(*)      - Real array, temperature(s); dimension is determined by
            the application program to be the total number of
            species temperatures, nominally 1.
               cgs units, K
ACT(*)    - Real array, activities of the species;
            dimension at least KKTOT, the total species count.
            The first KKGAS activities are mole fractions,
            the next KKSURF activities are site fractions
               (species density normalized by the site density;
                surface concentration in moles/cm**2 is
                ACT(K)*SITE_DENSITY / # sites per species), and
            the next KKBULK activities for bulk phase species
            should be from 0 to 1, and should sum to 1 for each
            phase.
SDEN(*)   - Real array, site densities for the site types;
            dimension at least NPHASE, the total phase count,
            but the subroutine only uses site phase entries,
            NFSURF <= N <= NLSURF.
               cgs units, moles/cm**2.
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real    workspace array; dimension at least LENRSK.

OUTPUT
DEN(*)    - Real array, densities for the species;
            dimension at least KKTOT, the total species count.
               cgm units, gm/cm**3 for gas-phase species
                          bm/cm**2 for surface species
                          gm/cm**3 for bulk species
              NOTE:  mass densities are not required to be input to
                     the Interpreter for bulk-phase species.
                     If they are input, they are returned by this
                     subroutine.  If not, DEN = -1.0 for the bulk
                     species
```

```
SKDRDA    SKDRDA    SKDRDA    SKDRDA    SKDRDA    SKDRDA    SKDRDA
          ****************************************
                    *************************
                              ******

SUBROUTINE SKDRDA (IR, P, T, ACT, SDEN, ISKWRK, RSKWRK, DKDAI)
Returns the partial of the rates of production of the species with
respect to the pre-exponential constant of surface reaction IR.

INPUT
IR        - Integer scalar, surface reaction index.
P         - Real scalar, pressure.
               cgs units, dynes/cm**2
T(*)      - Real array, temperature(s); dimension is determined by
            the application program to be the total number of
            species temperatures, nominally 1.
               cgs units, K
ACT(*)    - Real array, activities of the species;
            dimension at least KKTOT, the total species count.
            The first KKGAS activities are mole fractions,
            the next KKSURF activities are site fractions
               (species density normalized by the site density;
                surface concentration in moles/cm**2 is
                ACT(K)*SITE_DENSITY / # sites per species), and
            the next KKBULK activities for bulk phase species
            should be from 0 to 1, and should sum to 1 for each
            phase.
SDEN(*)   - Real array, site densities for the site types;
            dimension at least NPHASE, the total phase count,
            but the subroutine only uses site phase entries,
            NFSURF <= N <= NLSURF.
               cgs units, moles/cm**2.
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real    workspace array; dimension at least LENRSK.

OUTPUT
DKDAI(*)  - Real array, partials of the partial of production rates
            of the species with respect to the pre-exponential
            constant for surface reaction IR;
            dimension at least KKTOT, the total species count.
               cgs units, moles/(cm**2*sec) / (units of A)
```

```
SKDRDC    SKDRDC    SKDRDC    SKDRDC    SKDRDC    SKDRDC    SKDRDC
          ****************************************
                    **************************
                              ******

SUBROUTINE SKDRDC (KSPEC, P, T, ACT, SDEN, ISKWRK, RSKWRK, DKDC)
Returns the partial derivative of the production rates of the
species with respect to the concentration of species KSPEC.

INPUT
KSPEC     - Integer scalar, species index
P         - Real scalar, pressure.
               cgs units, dynes/cm**2
T(*)      - Real array, temperature(s); dimension is determined by
            the application program to be the total number of
            species temperatures, nominally 1.
               cgs units, K
ACT(*)    - Real array, activities of the species;
            dimension at least KKTOT, the total species count.
            The first KKGAS activities are mole fractions,
            the next KKSURF activities are site fractions
               (species density normalized by the site density;
                surface concentration in moles/cm**2 is
                ACT(K)*SITE_DENSITY / # sites per species), and
            the next KKBULK activities for bulk phase species
            should be from 0 to 1, and should sum to 1 for each
            phase.
SDEN(*)   - Real array, site densities for the site types;
            dimension at least NPHASE, the total phase count,
            but the subroutine only uses site phase entries,
            NFSURF <= N <= NLSURF.
               cgs units, moles/cm**2.
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real    workspace array; dimension at least LENRSK.

OUTPUT
DKDC(*)   - Real array, partial of the production rates of the
            species with respect to the concentration of species
            KSPEC;
            dimension at least KKTOT, the total species count.
               cgs units, moles/(cm**2*sec) / (units of KSPEC)
```

```
SKDSDC    SKDSDC    SKDSDC    SKDSDC    SKDSDC    SKDSDC    SKDSDC
          ****************************************
                    *************************
                              ******

SUBROUTINE SKDSDC (P, T, X, ACT, SDEN, ISKWRK, RSKWRK, DSDC, KKTOT,
                   SDOT, SITDOT)
Returns the partial derivative of the production rates of the
species with respect to the concentration of each species.
It also returns the matching production rates.

INPUT
P         - Real scalar, pressure.
               cgs units, dynes/cm**2
T(*)      - Real array, temperature(s); dimension is determined by
            the application program to be the total number of
            species temperatures, nominally 1.
               cgs units, K
X(*)      - Real array, mole fraction (or its equivalent) of the
            species;
            dimension at least KKTOT, the total species count.
            The first KKGAS X are mole fractions,
            the next KKSURF X are site fractions
               (species density normalized by the site density;
                surface concentration in moles/cm**2 is
                X(K)*SITE_DENSITY / # sites per species),
            the next KKBULK X are bulk species mole fractions.
ACT(*)    - Real array, activities of the species;
            dimension at least KKTOT, the total species count.
            The first KKGAS activities are mole fractions,
            the next KKSURF activities are site fractions
               (species density normalized by the site density;
                surface concentration in moles/cm**2 is
                ACT(K)*SITE_DENSITY / # sites per species), and
            the next KKBULK activities for bulk phase species
            should be from 0 to 1, and should sum to 1 for each phase.
SDEN(*)   - Real array, site densities for the site types;
            dimension at least NPHASE, the total phase count,
            but the subroutine only uses site phase entries,
            NFSURF <= N <= NLSURF.
               cgs units, moles/cm**2.
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real    workspace array; dimension at least LENRSK.
KKTOT     - Integer scalar, total species count.

OUTPUT
DSDC(*,*) - Real matrix, the partial derivatives of the production
            rates of the species with respect to the concentration
            of species KSPEC;
            dimension at least KKTOT, the total species count, for
            both the first and second dimensions.
               cgs units, moles/(cm**2*sec) / (units of KSPEC)
SDOT(*)   - Real array, production rates of the species;
            dimension at least KKTOT, the total species count.
               cgs units, moles/(cm**2*sec)
            for 1,KKGAS, the production rates of gas-phase species,
            for KKGAS+1,KKGAS+KKSUR, the production rates of surface
            species,
            for KKGAS+KKSUR+1,KKTOT, the production rate of bulk
            species.
SITDOT(*) - Real array, production rates of the surface phases;
            dimension at least NPHASE, the total phase count, but
            subroutine only calculates entries for site phases.
               cgs units, moles/(cm**2*sec)
```

```
SKDSDX    SKDSDX    SKDSDX    SKDSDX    SKDSDX    SKDSDX    SKDSDX
          ****************************************
                    *************************
                              ******

SUBROUTINE SKDSDX (P, T, X, ACT, SDEN, ISKWRK, RSKWRK, DSDX, KKTOT,
                   SDOT, SITDOT)
Returns the partial derivative of the production rates of the
species with respect to the activity for each species.
It also returns the matching production rates.

INPUT
P         - Real scalar, pressure.
               cgs units, dynes/cm**2
T(*)      - Real array, temperature(s); dimension is determined by
            the application program to be the total number of
            species temperatures, nominally 1.
               cgs units, K
X(*)      - Real array, mole fraction (or its equivalent) of the
            species;
            dimension at least KKTOT, the total species count.
            The first KKGAS X are mole fractions,
            the next KKSURF X are site fractions
               (species density normalized by the site density;
                surface concentration in moles/cm**2 is
                X(K)*SITE_DENSITY / # sites per species),
            the next KKBULK X are bulk species mole fractions.
ACT(*)    - Real array, activities of the species;
            dimension at least KKTOT, the total species count.
            The first KKGAS activities are mole fractions,
            the next KKSURF activities are site fractions
               (species density normalized by the site density;
                surface concentration in moles/cm**2 is
                ACT(K)*SITE_DENSITY / # sites per species), and
            the next KKBULK activities for bulk phase species
            should be from 0 to 1, and should sum to 1 for each phase.
SDEN(*)   - Real array, site densities for the site types;
            dimension at least NPHASE, the total phase count,
            but the subroutine only uses site phase entries,
            NFSURF <= N <= NLSURF.
               cgs units, moles/cm**2.
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real    workspace array; dimension at least LENRSK.
KKTOT     - Integer scalar, total species count.

OUTPUT
DSDX(*,*) - Real matrix, partial derivatives of the production rates
            of the species with respect to the activity of species
            KSPEC;
            dimension at least KKTOT, the total species count, for
            both the first and second dimensions.
               cgs units, moles/(cm**2*sec) / (units of KSPEC)
SDOT(*)   - Real array, production rates of the species;
            dimension at least KKTOT, the total species count.
               cgs units, moles/(cm**2*sec)
            SDOT(K) is
            for 1,KKGAS, the production rate of gas-phase species,
            for KKGAS+1,KKGAS+KKSUR, the production rate of surface
            species,
            for KKGAS+KKSUR+1,KKTOT, the production rate of bulk species.
SITDOT(*) - Real array, production rates of the surface phases;
            dimension at least NPHASE, the total phase count, but
            subroutine only calculates entries for site phases.
               cgs units, moles/(cm**2*sec)
```

```
SKEQ      SKEQ      SKEQ      SKEQ      SKEQ      SKEQ      SKEQ
          ******************************************
                    *************************
                              ******
```

```
SUBROUTINE SKEQ    (P, T, ACT, SDEN, ISKWRK, RSKWRK, EQKC)
Returns the equilibrium constants for the surface reactions given
pressure, temperature, species activities, and the site densities.

INPUT
P         - Real scalar, pressure.
               cgs units, dynes/cm**2
T(*)      - Real array, temperature(s); dimension is determined by
            the application program to be the total number of
            species temperatures, nominally 1.
               cgs units, K
ACT(*)    - Real array, activities of the species;
            dimension at least KKTOT, the total species count.
            The first KKGAS activities are mole fractions,
            the next KKSURF activities are site fractions
               (species density normalized by the site density;
                surface concentration in moles/cm**2 is
                ACT(K)*SITE_DENSITY / # sites per species), and
            the next KKBULK activities for bulk phase species
            should be from 0 to 1, and should sum to 1 for each
            phase.
SDEN(*)   - Real array, site densities for the site types;
            dimension at least NPHASE, the total phase count,
            but the subroutine only uses site phase entries,
            NFSURF <= N <= NLSURF.
               cgs units, moles/cm**2.
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real    workspace array; dimension at least LENRSK.

OUTPUT
EQKC(*)   - Real array, equilibrium constants in concentration units
            for the reactions;
            dimension at least IISUR, the total surface reaction
            count.
               cgs units, depends on reaction (moles, cm)
```

```
SKFLGS    SKFLGS    SKFLGS    SKFLGS    SKFLGS    SKFLGS    SKFLGS
          ****************************************
                    *************************
                              ******

SUBROUTINE SKFLGS (IR, ISKWRK, NRPP, IREV, ISTFL, ICOV, IMOTZ,
                   IEDP, IBHM, IORD, IYLD)
Returns several integer flags describing surface reaction IR.

INPUT
IR        - Integer scalar, surface reaction index.
ISKWRK(*) - Integer workspace array; dimension at least LENISK.

OUTPUT
NRPP      - Integer scalar, number of species (reactants+products) for
            surface reaction IR, combined with reversibility flag.
            NRPP > 0, NRPP species, reversible surface reaction,
                < 0, ABS(NRPP) species, irreversible reaction.
IREV      - Integer scalar, flag for explicit reverse Arrhenius
            parameters.
            =1, reaction has explicit reverse Arrhenius parameters
            =0, no (may or may not be reversible, see NRPP).
ISTFL     - Integer scalar, flag for sticking coefficients;
            =1, reaction does not use sticking coefficients
            =0, no
IMOTZ     - Integer scalar, flag for Motz-Wise correction of
            sticking coefficients;
            =1, sticking reaction with Motz-Wise correction
            =0, no (may or may not be sticking reaction, see ISTFL)
ICOV      - Integer scalar, flag to indidicate that reaction has
            coverage dependence;
            =1, reaction has coverage dependence
            =0, no.
IEDP      - Integer scalar, flag for energy-dependence;
            =1, reaction is energy-dependent,
            =0, no.
IBHM      - Integer scalar, flag for Bohm correction;
            =1, Bohm reaction,
            =0, no
IORD      - Integer scalar, flag for species order change;
            =1, reaction has species order change,
            =0, no
IYLD      - Integer scalar, flag for yield-modification;
            =1, yield-modification in reaction;
            =0, no
```

```
SKGML     SKGML     SKGML     SKGML     SKGML     SKGML     SKGML
          ****************************************
                    *************************
                              ******

SUBROUTINE SKGML  (T, ISKWRK, RSKWRK, GML)
Returns an array of the standard state Gibbs free energies
in molar units.

INPUT
T(*)      - Real array, temperature(s); dimension is determined by
            the application program to be the total number of
            species temperatures, nominally 1.
               cgs units, K
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real    workspace array; dimension at least LENRSK.

OUTPUT
GML(*)    - Real array, standard state Gibbs free energies
            for the species;
            dimension KKTOT, the total species count.
               cgs units, ergs/mole



SKGMS     SKGMS     SKGMS     SKGMS     SKGMS     SKGMS     SKGMS
          ****************************************
                    *************************
                              ******

SUBROUTINE SKGMS  (T, ISKWRK, RSKWRK, GMS)
Returns an array of the standard state Gibbs free energies
in mass units.

INPUT
T(*)      - Real array, temperature(s); dimension is determined by
            the application program to be the total number of
            species temperatures, nominally 1.
               cgs units, K
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real    workspace array; dimension at least LENRSK.

OUTPUT
GMS(*)    - Real array, standard state Gibbs free energies
            for the species;
            dimension at least KKTOT, the total species count.
               cgs units, ergs/gm
```

SKHML SKHML SKHML SKHML SKHML SKHML SKHML

SUBROUTINE SKHML (T, ISKWRK, RSKWRK, HML)
Returns an array of the enthalpies in molar units.

INPUT
T(*) - Real array, temperature(s); dimension is determined by the application program to be the total number of species temperatures, nominally 1.
cgs units, K
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real workspace array; dimension at least LENRSK.

OUTPUT
HML(*) - Real array, enthalpies for the species; dimension at least KKTOT, the total species count.
cgs units, ergs/mole

SKHMS SKHMS SKHMS SKHMS SKHMS SKHMS SKHMS

SUBROUTINE SKHMS (T, ISKWRK, RSKWRK, HMS)
Returns an array of the enthalpies in mass units.

INPUT
T(*) - Real array, temperature(s); dimension is determined by the application program to be the total number of species temperatures, nominally 1.
cgs units, K
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real workspace array; dimension at least LENRSK.

OUTPUT
HMS(*) - Real array, enthalpies for the species; dimension at least KKTOT, the total species count.
cgs units, ergs/gm

SKHORT SKHORT SKHORT SKHORT SKHORT SKHORT SKHORT
****
****
******

SUBROUTINE SKHORT (T, ISKWRK, RSKWRK, HORT)
Returns an array of the nondimensional enthalpies.

INPUT
T(*) - Real array, temperature(s); dimension is determined by
  the application program to be the total number of
  species temperatures, nominally 1.
    cgs units, K
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real workspace array; dimension at least LENRSK.

OUTPUT
HORT(*) - Real array, nondimensional enthalpies for the species;
  dimension at least KKTOT, the total species count.

SKIBHM SKIBHM SKIBHM SKIBHM SKIBHM SKIBHM SKIBHM
****
****
******

SUBROUTINE SKIBHM (IR, ISKWRK, IBMFL)
Returns an integer flag to indicate whether reaction IR uses
BOHM sticking coefficients.

INPUT
IR - Integer scalar, surface reaction index.
ISKWRK(*) - Integer workspace array; dimension at least LENISK.

OUTPUT
IBMFL - Integer scalar, flag for Bohm reactions;
  0, reaction IR does not use BOHM sticking coefficients
  1, reaction IR does use BOHM sticking coefficients

```
SKICOV    SKICOV    SKICOV    SKICOV    SKICOV    SKICOV    SKICOV
          ****************************************
                    *************************
                              ******

SUBROUTINE SKICOV (IR, NDIM, ISKWRK, RSKWRK, NCOVI, KCOVI, CPARI)
Returns the coverage species index numbers and their coverage
parameters for reaction IR.

INPUT
IR        - Integer scalar, surface reaction index.
NDIM      - Integer scalar, first dimension of array CPAR, the
            coverage parameters; NDIM must be at least NSCOV,
            the total number of coverage parameters.
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real    workspace array; dimension at least LENRSK.

OUTPUT
NCOVI     - Integer scalar, total number of species that modify the
            rate of reaction IR through coverage dependence.
KCOVI(*)  - Integer array, species indices for the NCOVI species that
            modify the rate of a coverage dependence reaction;
            dimension at least KKTOT, the total species count.
CPARI(*,*)- Real matrix, coverage parameters for the coverage species
            of reaction IR;
            dimension at least NSCOV for the first, the number of
            coverage parameters required, and at least KKTOT for the
            second, the total species count.



SKIENR    SKIENR    SKIENR    SKIENR    SKIENR    SKIENR    SKIENR
          ****************************************
                    *************************
                              ******

SUBROUTINE SKIENR (IR, ISKWRK, SKWRK, IENRFL, IEION, PEDEP)
Returns an integer flag to indicate if reaction IR is ion-energy-
dependent, and if so, formulation-specific parameters.

INPUT
IR        - Integer scalar, reaction index;
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real    workspace array; dimension at least LENRSK.

OUTPUT
IENRFL    - Integer scalar,
            0, reaction IR does not have an ion-energy dependence
            1, reaction IR does have an ion-energy dependence
IEION     - Integer scalar, species index of the ion on which reaction
            is dependent
PEDEP(*)  - Real array, supplemental parameters for an
            ion-energy-dependent reaction rate formulation;
            dimension at least NEDPAR, the number of supplemental
            rate parameters required.
```

SKINDX SKINDX SKINDX SKINDX SKINDX SKINDX SKINDX
****
****
******

SUBROUTINE SKINDX (ISKWRK, NELM, KKGAS, KKSUR, KKBULK, KKTOT,
NNPHAS, NNSURF, NFSURF, NLSURF, NNBULK, NFBULK,
NLBULK, IISUR)
Returns a group of indices defining the size of the surface
reaction mechanism.

INPUT
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
OUTPU
NELM - Integer scalar, total element count.
KKGAS - Integer scalar, total gas-phase species count.
KKSUR - Integer scalar, total surface species count.
KKBULK - Integer scalar, total bulk species count.
KKTOT - Integer scalar, total species count (KKGAS+KKSUR+KKBULK).
NNPHAS - Integer scalar, total phase count (gas + sites + bulks).
NNSURF - Integer scalar, total surface phase count.
NFSURF - Integer scalar, phase index of the first surface phase.
NLSURF - Integer scalar, phase index of the last surface phase.
NNBULK - Integer scalar, total bulk phase count.
NFBULK - Integer scalar, phase index of the first bulk phase.
NLBULK - Integer scalar, phase index of the last bulk phase.
IISUR - Integer scalar, total surface reaction count.

SKINIT SKINIT SKINIT SKINIT SKINIT SKINIT SKINIT
****
****
******

SUBROUTINE SKINIT (LENISK, LENRSK, LENCSK, LINSK, LOUT,
ISKWRK, RSKWRK, CSKWRK, IFLAG)
Reads the surface linkfile and creates internal work arrays ISKWRK,
RSKWRK, and CSKWRK. SKINIT must be called before any other Surface
CHEMKIN-III subroutine can be used, as the work arrays must be available
as their input.

INPUT
LENISK - Integer scalar, length of the integer array ISKWRK.
LENRSK - Integer scalar, length of the real array RSKWRK.
LENCSK - Integer scalar, length of the character string array
CSKWRK.
LINSK - Integer scalar, linkfile input file unit number.
LOUT - Integer scalar, formatted output file unit number.

OUTPUT
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real workspace array; dimension at least LENRSK.
CSKWRK(*) - Character string workspace array; dimension at least
LENCSK.
IFLAG - Integer scalar to indicate successful reading of
linkfile; IFLAG>0 is an error type.

SKINU SKINU SKINU SKINU SKINU SKINU SKINU

SUBROUTINE SKINU (IS, NDIM, ISKWRK, RSKWRK, NSPEC, KI, NU)
Returns the number of species in a surface reaction, and the species indices and stoichiometric coefficients.

INPUT
IR - Integer scalar, index number of a surface reaction; IR must be greater than 0 and less than or equal to IISUR, the total surface reaction count.
NDIM - Integer scalar, dimension of the arrays KI and NU; NDIM must be at least MAXSPR, the total number of species allowed in a surface reaction.
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real workspace array; dimension at least LENRSK.

OUTPUT
NSPEC - Integer scalar, the number of species (reactants + products) in surface reaction IR.
KI(*) - Integer array, species indices for the species in surface reaction IR; dimension at least MAXSPR, the total number of species allowed in a surface reaction.
NU(*) - Integer array, stoichiometric coefficients of the species in surface reaction IR; dimension at least MAXSPR, the total number of species allowed in a surface reaction. NU is negative if the Nth species is a reactant; NU is positive if the Nth species is a product.

SKIORD SKIORD SKIORD SKIORD SKIORD SKIORD SKIORD

SUBROUTINE SKIORD (IDIM, KDIM, ISKWRK, RSKWRK, NFORD, IFORD, FORD, NRORD, IRORD, RORD)

Returns the number and indices of surface reactions with modified species orders, and the order values for the species in the surface mechanism.

INPUT

IDIM - Integer scalar, dimension of arrays IFORD and IRORD; IDIM must be at least NORD, the total number of surface reactions with modified species orders.

KDIM - Integer scalar, first dimension of the arrays FORD and RORD; KDIM must be at least NKK, the total species count.

ISKWRK(*) - Integer workspace array; dimension at least LENISK.

RSKWRK(*) - Real workspace array; dimension at least LENRSK.

OUTPUT

NFORD - Integer scalar, total number of surface reactions with modified forward species orders.

IFORD(*) - Integer array, indices of surface reactions with modified forward species orders; dimension at least NFORD.

FORD(*,*) - Real matrix, the modified forward species orders for the NFORD surface reactions; dimension at least KKTOT, the total species count, for the first, and at least NFORD for the second. FORD(K,N) is the forward order of species K for the Nth surface change-order reaction.

NRORD - Integer scalar, total number of surface reactions with modified reverse species orders.

IRORD(*) - Integer array, indices of surface reactions with modified reverse species orders; dimension at least NRORD.

RORD(*,*) - Real matrix, the modified reverse species orders for the NRORD surface reactions; dimension at least KKTOT for the first, the total species count, and at least NRORD for the second. RORD(K,N) is the reverse order of species K for the Nth surface change-order reaction.

SKIREV SKIREV SKIREV SKIREV SKIREV SKIREV SKIREV
****
****
******

SUBROUTINE SKIREV (IR, ISKWRK, RSKWRK, IREV, RAR, RBR, RER)
Returns an integer flag to indicate whether reaction IR has an
explicitly assigned reverse rate constant. It also returns the
reverse Arrhenius expression values for surface reaction IR,
if it was explicitly assigned in the SURFACE CHEMKIN-III interpreter.
If reverse Arrhenius values were not explicitly assigned,
RAR, RBR and RER will be zero.

INPUT
IR - Integer scalar, surface reaction index.
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real workspace array; dimension at least LENRSK.

OUTPUT
IREV - Integer scalar,
1, reaction IR has explicit reverse rate parameters
0, no.
RAR - Real scalar, explicit pre-exponential constants
for reaction IR.
cgs units, mole-cm-sec-K
RBR - Real scalar, explicit temperature dependence exponents
for reaction IR.
RER - Real scalar, explicit activation energy for reaction IR.
cgs units, Kelvins

```
SKIRNU    SKIRNU    SKIRNU    SKIRNU    SKIRNU    SKIRNU    SKIRNU
          ****************************************
                    *************************
                              ******

SUBROUTINE SKIRNU (IDIM, NDIM, ISKWRK, RSKWRK, NIRNU, IRNU, NSPEC,
                   KI, RNU)
Returns the number and indices of surface reactions with real
stoichiometric coefficients, number of species in the reactions,
and the species indices and coefficients;

INPUT
IDIM      - Integer scalar, dimension of the arrays IRNU and NSPEC,
            and the second dimension of matrices KI and RNU;
            IDIM must be at least NIRNU, the number of surface
            reactions with real stoichiometric coefficients.
NDIM      - Integer scalar, first dimension of matrices KI and RNU;
            NDIM must be at least MAXSPR, the maximum number of
            species allowed in a surface reaction.
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real    workspace array; dimension at least LENRSK.

OUTPUT
NIRNU     - Integer scalar, total number of surface reactions with
            real stoichiometric coefficients.
IRNU(*)   - Integer array, indices of surface reactions with real
            stoichiometric coefficients; dimension at least NIRNU.
NSPEC(*)  - Integer array, total number of species in a surface
            reaction;
            dimension at least NIRNU.
KI(*,*)   - Integer matrix, species indices for species in a surface
            reaction;
            dimension at least MAXSPR for the first, and at least
            NIRNU for the second.
            KI(M,N) is the species index of the Mth species in the
            Nth real coefficient surface reaction.
RNU(*,*)  - Real matrix, stoichiometric coefficients for species
            in the NIRNU reactions; dimension at least MAXSPR for
            the first, and at least NIRNU for the second.
            RNU(M,N) is the stoichiometric coefficient of the Mth
            species in the Nth real coefficient surface reaction, and
            RNU(M,*) < 0 if the Mth species is a reactant;
            RNU(M,*) > 0 if the Mth species is a product.



SKISTK    SKISTK    SKISTK    SKISTK    SKISTK    SKISTK    SKISTK
          ****************************************
                    *************************
                              ******

SUBROUTINE SKISTK (IR, ISKWRK, ISTFL)
Returns an integer flag to indicate whether reaction IR uses
sticking coefficients.

INPUT
IR        - Integer scalar, index of a surface reaction.
ISKWRK(*) - Integer workspace array; dimension at least LENISK.

OUTPUT
ISTFL     - Integer scalar,
            0, reaction IR does not use sticking coefficients
            1, reaction IR does use sticking coefficients
```

SKIYLD SKIYLD SKIYLD SKIYLD SKIYLD SKIYLD SKIYLD

SUBROUTINE SKIYLD (IR, ISKWRK, RSKWRK, NYDIM, IYLD, IYION, KYLD, PYLD)
Returns an integer flag to indicate whether reaction IR has yield-modified species, the species index of its ion, yield-modify flags for its reactants and products, and parameters for the yield expression.

INPUT
IR - Integer scalar, surface reaction index.
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real workspace array; dimension at least LENRSK.
NYDIM - Integer scalar, first dimension for YPAR, an array of yield-modify parameters;
NYDIM must be at least NYPAR, the number of parameters for the yield expression.

OUTPUT
IYLD - Integer scalar, flag for yield-modify reactions;
1, reaction IR uses yield-modification
0, no
IYION - Integer scalar, species index of the ion in a yield-modify reaction.
KYLD(*) - Integer array, yield flags for the species in a yield-modify reaction;
dimension at least MAXSPR, the maximum number of species allowed in a surface reaction.
1, species is yield-modified
0, no
PYLD(*) - Real array, parameters for the yield-expression in a yield-modify reaction;
dimension at least NYPAR, the number of parameters required.
If IYLD=1, and KYLD of the Nth species in the reaction is 1, the stoichiometric coefficient NU of the species is scaled by the results of the expression
PYLD(1) * [Ei**PYLD(3) - PYLD(2)**PYLD(3)] **PYLD(4)
where Ei is the ion energy of species IYION.

SKKION SKKION SKKION SKKION SKKION SKKION SKKION

SUBROUTINE SKKION (ISKWRK, KKELECT, KKION, KION)
Returns the species number of the electron, the number of positive ions in the gas phase, and an array of species number for each positive ion

INPUT
ISKWRK(*) - Integer workspace array; dimension at least LENISK.

OUTPUT
KELECT - Integer scalar, species index of the electron species.
KKION - Integer scalar, total gas-phase positive ion count.
KION - Integer array, species indices for the gas-phase positive ions;
dimension at least NKKGAS, the gas-phase species count.

SKKTFL SKKTFL SKKTFL SKKTFL SKKTFL SKKTFL SKKTFL
****
****
******

SUBROUTINE SKKTFL (ISKWRK, KTFL)
Allows the user to assign a location in the temperature array
to use for the gas-phase species.

INPUT
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
KTFL(*) - Integer array, ndices into the temperature(s) for
gas-phase species;
dimension at least KKGAS, the total gas-phase species
count.
Default value stored in ISKWRK is set to 1 in SKINIT.

SKLEN SKLEN SKLEN SKLEN SKLEN SKLEN SKLEN
****
****
******

SUBROUTINE SKLEN (LINSK, LOUT, LENI, LENR, LENC, IFLAG)
Reads the first record of the linkfile to return the lengths
required for the integer, real, and character work arrays.

INPUT
LINSK - Integer scalar, input unit number assigned to linkfile.
LOUT - Integer scalar, formatted output unit file number.

OUTPUT
LENI - Integer scalar, dimension required for integer work
array, ISKWRK.
LENR - Integer scalar, dimension required for real work
array, RSKWRK.
LENC - Integer scalar, dimension required for character work
array, CSKWRK.

SKMXTP SKMXTP SKMXTP SKMXTP SKMXTP SKMXTP SKMXTP
****
****
******

SUBROUTINE SKMXTP (ISKWRK, MXTP)
Returns the maximum number of temperatures used in
fitting the thermodynamic properties of the species.

INPUT
ISKWRK(*) - Integer workspace array; dimension at least LENISK.

OUTPUT
MXTP - Integer scalar, maximum number of temperatures used in
fitting the thermodynamic properties of the species.

SKNCF SKNCF SKNCF SKNCF SKNCF SKNCF SKNCF

SUBROUTINE SKNCF (NELDIM, ISKWRK, NEL)
Returns the elemental composition of the species.

INPUT
NELDIM - Integer scalar, first dimension of the matrix NEL;
must be at least NELEM, the total element count.
ISKWRK(*) - Integer workspace array; dimension at least LENISK.

OUTPUT
NEL(*,*) - Integer matrix, elemental compositions of the species;
dimension at least NELEM for the first, the total
element count, and at least KKTOT for the second, the
total species count.
NEL(M,K) is the quantity of element M in species K.

SKNCON SKNCON SKNCON SKNCON SKNCON SKNCON SKNCON

SUBROUTINE SKNCON (ISKWRK, RSKWRK, NCON)
Returns the total number of surface reactions which do not conserve
sites of the phases.

INPUT
ISKWRK(*) - Integer workspace array; dimension at least LENISK.

OUTPUT
NCON(*) - Integer array, count of surface reactions which do not
conserve sites in the phases;
dimension at least NPHASE, the total phase count.

```
SKNU      SKNU      SKNU      SKNU      SKNU      SKNU      SKNU
          ******************************************
                    *************************
                              ******
```

SUBROUTINE SKNU (IDIM, ISKWRK, RSKWRK, KSTOIC, NSTOIC)
Returns the stoichiometric coefficients of the species and the net change in phases for all of the surface reactions in a mechanism.

INPUT
IDIM - Integer scalar, first dimension of the array NSTOIC; must be at least IISUR, the total surface reaction count.
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real workspace array; dimension at least LENRSK.

OUTPUT
KSTOIC(*,*)-Integer matrix, stoichiometric coefficients for the species in the surface reactions; the first dimension must be at least IISUR, the total surface reaction count, and at least KKTOT for the second, the total species count.
NSTOIC(*,*)-Integer matrix, net change of the phases for the surface reactions; the first dimension must be at least IISUR, the total surface reaction count, and at least NPHASE for the second the total phase count.

```
SKNUF     SKNUF     SKNUF     SKNUF     SKNUF     SKNUF     SKNUF
          ******************************************
                    *************************
                              ******
```

SUBROUTINE SKNUF (IDIM, ISKWRK, KSTOIF)
Returns the stoichiometric coefficients of the species for all reactants in all surface reactions in a mechanism.
(note - reactants only! - they will all be negative)

INPUT
IDIM - Integer scalar, first dimension of the array NSTOIC; must be at least ISUR, the total surface reaction count.
ISKWRK(*) - Integer workspace array; dimension at least LENISK.

OUTPUT
KSTOIF(*,*)-Integer matrix, stoichiometric coefficients for the reactants in the surface reactions; dimension at least IISUR for the first, the total surface reaction ount, and at least KKTOT for the second, the total species count.

SKPCMP SKPCMP SKPCMP SKPCMP SKPCMP SKPCMP SKPCMP

SUBROUTINE SKPCMP (ISTR, IRAY, NN, SETS, NSETS, ISET, IND, NT)
This subroutine can do everything that the subroutine SKCOMP can do, and additionally, has the capabilities of separating the elements of IRAY into categories and then search IRAY by element and category. The categories that each element of IRAY will be assigned to are specified by the input character string vector SETS of vector length NSETS. Elements of each category in IRAY must be grouped congrously. The number of elements in each category within IRAY is specified by the input integer vector ISET. To search for the existence of an element within acategory ISTR may additionally be composed of two substrings, ISTR="ELEMENT_NAME/CATEGORY_NAME/", where CATEGORY_NAME is one of the categories specified in SETS. In this case, IND will return the first position in IRAY where ELEMENT_NAME occurred within the category CATEGORY_NAME. NT will return the total number of times ELEMENT_NAME occurred within the category CATEGORY_NAME. If ELEMENT_NAME is not found within the specified category, IND and NT are returned with a value of zero. If no category is specified within ISTR, IND and NT return with the same values as they would from subroutine SKCOMP.
Consider the following example,

```
IRAY = {"RED", "BLUE", "JADE", "RUBY", "TOPAZ", "JADE"}
NN = 6
SETS = {"COLORS", "STONES"},
NSETS = 2
ISET = {4, 2}.
```

This assumes that the elements of IRAY were grouped into two sets, consisting of 4 and 2 elements, respectively, and the following names

```
"COLORS"  = {"RED", "BLUE", "JADE", "RUBY"}, and
"STONES"  = {"TOPAZ", "JADE"}.
```

If ISTR="BLUE" then IND=2 and NT=1;
if ISTR="PINK" then IND=0 and NT=0; and
if ISTR="JADE",then IND=3 and NT=2.
If ISTR="BLUE/COLORS/" then IND=2 and NT=1;
if ISTR="BLUE/STONES/" then IND=0 and NT=0;
if ISTR="JADE/GEMS/" then IND=0 and NT=0; and
if ISTR="JADE/STONES/",then IND=6 and NT=1.

INPUT

ISTR - Character string, which may or may not end with a slash-delimited substring.

IRAY(*) - Character string array; dimension at least NN.

NN - Integer scalar, number of entries in IRAY(*).

SETS(*) - Character string array, cross-reference set to relate with elements of IRAY; dimension at least NSETS.

NSETS - Integer scalar, number of entries in SETS(*)

ISET(*) - Integer array, total number of entries in a subset of IRAY; dimension at least NSETS.

OUTPUT

IND - Integer scalar, index of ISTR in IRAY(*).
If ISTR is not in IRAY(*), IND = 0.
If the slash-delimited substring of ISTR is not in SETS(*), IND = 0.
If the slash-delimited substring of ISTR is in SETS(N), but the substring before the slash is not a member of the subset associated with SETS(N),

```
              IND = 0, whether or not the substring is in IRAY(*).
NT          - Integer scalar, total occurrence of ISTR in IRAY(*),
              or total number of times ISTR occurs in a subset
              of IRAY(*).



SKPKK     SKPKK     SKPKK     SKPKK     SKPKK     SKPKK     SKPKK
          ****************************************
                    *************************
                              ******

SUBROUTINE SKPKK  (ISKWRK, KKPHAS, KFIRST, KLAST)
Returns arrays of species pointers for the phases.

INPUT
ISKWRK(*) - Integer workspace array; dimension at least LENISK.

OUTPUT
KKPHAS(*) - Integer array, the total species counts for phases;
            dimension at least NPHASE, the total phase count.
KFIRST(*) - Integer array, species indices for the first species of
            the phases;
            dimension at least NPHASE, the total phase count.
KLAST(*)  - Integer array, species indices for the last species of
            the phases;
            dimension at least NPHASE, the total phase count.


SKPNT     SKPNT     SKPNT     SKPNT     SKPNT     SKPNT     SKPNT
          ****************************************
                    *************************
                              ******

SUBROUTINE SKPNT  (LSAVE, LOUT, VERS, PREC, LENI, LENR, LENC, IERR)
Reads from a file information about a SURFACE CHEMKIN-III linkfile,
pointers for the SURFACE CHEMKIN-III Library, and returns lengths
of work arrays.

INPUT
LSAVE     - Integer scalar, input unit for binary data file.
LOUT      - Integer scalar, formatted output file unit number.

OUTPUT
VERS      - Real scalar, version number of the SURFACE CHEMKIN-III
            linkfile.
PREC      - Character string, machine precision of the linkfile.
LENI      - Integer scalar, length required for integer work array.
LENR      - Integer scalar, length required for real work array.
LENC      - Integer scalar, length required for character work array.
KERR      - Logical, error flag.
```

```
SKRAEX    SKRAEX    SKRAEX    SKRAEX    SKRAEX    SKRAEX    SKRAEX
          ****************************************
                    *************************
                              ******

SUBROUTINE SKRAEX (IR, ISKWRK, RSKWRK, RA)
Returns the Pre-exponential rate constant
(or sticking coefficient) of the IRth reaction, or changes its
value, depending on the sign of IR.

INPUT
IR        - Integer scalar, reaction index;
            IR> 0 gets RA(I) from RSKWRK
            IR< 0 puts RA(I) into RSKWRK
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real    workspace array; dimension at least LENRSK.
If IR< 0:
RA        - Real scalar, pre-exponential or sticking coefficient for
            reaction IR.
               cgs units, mole-cm-sec-K for pre-exponential,
                          none for sticking coefficients

OUTPUT
If IR> 0:
RA        - Real scalar, pre-exponential or sticking coefficient for
            reaction IR.
               cgs units, mole-cm-sec-K for pre-exponential,
                          none for sticking coefficients
```

```
SKRAT     SKRAT     SKRAT     SKRAT     SKRAT     SKRAT     SKRAT
          ****************************************
                    *************************
                              ******

SUBROUTINE SKRAT  (P, T, ACT, SDEN, ISKWRK, RSKWRK, SDOT, SITDOT)
Returns production rates for the species and sites.

INPUT
P         - Real scalar, pressure.
               cgs units, dynes/cm**2
T(*)      - Real array, temperature(s); dimension is determined by
            the application program to be the total number of
            species temperatures, nominally 1.
               cgs units, K
ACT(*)    - Real array, activities of the species;
            dimension at least KKTOT, the total species count.
            The first KKGAS activities are mole fractions,
            the next KKSURF activities are site fractions
               (species density normalized by the site density;
                surface concentration in moles/cm**2 is
                ACT(K)*SITE_DENSITY / # sites per species), and
            the next KKBULK activities for bulk phase species
            should be from 0 to 1, and should sum to 1 for each
            phase.
SDEN(*)   - Real array, site densities for the site types;
            dimension at least NPHASE, the total phase count,
            but the subroutine only uses site phase entries,
            NFSURF <= N <= NLSURF.
               cgs units, moles/cm**2.
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real    workspace array; dimension at least LENRSK.

OUTPUT
SDOT(*)   - Real array, production rates of the species;
            dimension at least KKTOT, the total species count.
               cgs units, moles/(cm**2*sec)
            for 1,KKGAS, the production rate of gas-phase species,
            for KKGAS+1,KKGAS+KKSUR, the production rate of surface
            species,
            for KKGAS+KKSUR+1,KKTOT, the production rate of bulk
            species.
SITDOT(*) - Real array, production rates of the surface phases;
            dimension at least NPHASE, the total phase count, but
            subroutine only calculates entries for site phases.
               cgs units, moles/(cm**2*sec)
```

```
 SKRATI    SKRATI    SKRATI    SKRATI    SKRATI    SKRATI    SKRATI
           ****************************************
                     *************************
                               ******

 SUBROUTINE SKRATI (IR, ROP, ISKWRK, RSKWRK, SDOTI, SITDTI)
 Returns rates of production of the species by surface reaction IR.

 INPUT
 IR        - Integer scalar, reaction index;
 ROP(*)    - Real array, rates of progress for the surface reactions;
             dimension at least IISUR, the total surface reaction
             count.
                cgs units, moles/(cm**2*sec).
 ISKWRK(*) - Integer workspace array; dimension at least LENISK.
 RSKWRK(*) - Real    workspace array; dimension at least LENRSK.

 OUTPUT
 SDOTI(*)  - Real array, production rates of the species by reaction
             IR;
             dimension at least KKTOT, the total species count.
                cgs units, moles/(cm**2*sec)
             for 1,KKGAS, the production rate of gas-phase species,
             for KKGAS+1,KKGAS+KKSUR, the production rate of surface
             species,
             for KKGAS+KKSUR+1,KKTOT, the production rate of bulk
             species.
SITDTI(*)  - Real array, production rates of the surface phases due to
             reaction IR;
             dimension at least NPHASE, the total phase count, but
             subroutine calculates entries only for site phases.
                cgs units, moles/(cm**2*sec)



 SKRDEX    SKRDEX    SKRDEX    SKRDEX    SKRDEX    SKRDEX    SKRDEX
           ****************************************
                     *************************
                               ******

 SUBROUTINE SKRDEX (IR, ISKWRK, RSKWRK, RD)
 Returns the perturbation factor of the IRth reaction,
 or changes its value, depending on the sign of IR.

 INPUT
 IR        - Integer scalar, reaction index;
             IR> 0 gets RD(I) from RSKWRK
             IR< 0 puts RD(I) into RSKWRK
 ISKWRK(*) - Integer workspace array; dimension at least LENISK.
 RSKWRK(*) - Real    workspace array; dimension at least LENRSK.
 If IR< 0:
 RD        - Real scalar, perturbation factor for reaction IR.

 OUTPUT
 If IR> 0:
 RD        - Real scalar, perturbation factor for reaction IR.
```

SKRHEX SKRHEX SKRHEX SKRHEX SKRHEX SKRHEX SKRHEX

SUBROUTINE SKRHEX (K, ISKWRK, RSKWRK, A6)
Returns an array of the sixth thermodynamic polynomial
coefficients for a species, or changes their value,
depending on the sign of K.

INPUT
K - Integer scalar, species index;
K > 0 gets A6(*) from RSKWRK
K < 0 puts A6(*) into RSKWRK
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real workspace array; dimension at least LENRSK.
If K < 0:
A6(*) - Real array, the 6th thermodynamic polynomial
coefficients for species K, over the number of
temperature ranges used in fitting thermodynamic
properties;
dimension at least MAXTP-1, where MAXTP is the
maximum number of temperatures used in fitting the
thermodynamic properties of the species.

OUTPUT
If K > 0:
A6(*) - Real array, the 6th thermodynamic polynomial
coefficients for species K, over the number of
temperature ranges used in fitting thermodynamic
properties;
dimension at least MAXTP-1, where MAXTP is the
maximum number of temperatures used in fitting the
thermodynamic properties of the species.

```
SKROP     SKROP     SKROP     SKROP     SKROP     SKROP     SKROP
          ****************************************
                    *************************
                              ******

SUBROUTINE SKROP  (P, T, ACT, SDEN, ISKWRK, RSKWRK, ROP)
Returns rates of progress for the surface reactions.

INPUT
P         - Real scalar, pressure.
               cgs units, dynes/cm**2
T(*)      - Real array, temperature(s); dimension is determined by
            the application program to be the total number of
            species temperatures, nominally 1.
               cgs units, K
ACT(*)    - Real array, activities of the species;
            dimension at least KKTOT, the total species count.
            The first KKGAS activities are mole fractions,
            the next KKSURF activities are site fractions
               (species density normalized by the site density;
                surface concentration in moles/cm**2 is
                ACT(K)*SITE_DENSITY / # sites per species), and
            the next KKBULK activities for bulk phase species
            should be from 0 to 1, and should sum to 1 for each
            phase.
SDEN(*)   - Real array, site densities for the site types;
            dimension at least NPHASE, the total phase count,
            but the subroutine only uses site phase entries,
            NFSURF <= N <= NLSURF.
               cgs units, moles/cm**2.
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real    workspace array; dimension at least LENRSK.

OUTPUT
ROP(*)    - Real array, rates of progress for the surface reactions;
            dimension at least IISUR, the total surface reaction
            count.
               cgs units, moles/(cm**2*sec).




SKRP      SKRP      SKRP      SKRP      SKRP      SKRP      SKRP
          ****************************************
                    *************************
                              ******

SUBROUTINE SKRP   (ISKWRK, RSKWRK, RU, RUC, PATM)
Returns universal gas constants and the pressure of one standard
atmosphere.

INPUT
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real    workspace array; dimension at least LENRSK.

OUTPUT
RU        - Real scalar, universal gas constant.
               cgs units, 8.314510E7 ergs/(mole*K)
RUC       - Real scalar, universal gas constant used only in
            conjuction with activation energy.
               preferred units, RU / 4.184 cal/(mole*K)
PA        - Real scalar, pressure of one standard atmosphere.
               cgs units, 1.01325E6 dynes/cm**2
```

SKRPAR SKRPAR SKRPAR SKRPAR SKRPAR SKRPAR SKRPAR
****
****
******

SUBROUTINE SKRPAR (ISKWRK, RSKWRK, ENRGI)
Allows the user to input auxiliary reaction-rate parameters for special types of reactions. The first parameter is the species (ion) directed energy for ion-energy-dependent reactions.

INPUT
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real workspace array; dimension at least LENRSK.
ENRGI(*) - Real array, species ion energies used in the NIIEDP reactions;
dimension at least KKGAS, the total gas-phase species count.
Default value stored in RSKWRK is set to 0.0 in SKINIT.

SKSAVE SKSAVE SKSAVE SKSAVE SKSAVE SKSAVE SKSAVE
****
****
******

SUBROUTINE SKSAVE (LOUT, LSAVE, ISKWRK, RSKWRK, CSKWRK)
Writes to a binary file information about a SURFACE CHEMKIN-III linkfile, pointers for the SURFACE CHEMKIN-III Library, and SURFACE CHEMKIN-III work arrays.

INPUT
LOUT - Integer scalar, formatted output file unit number.
LSAVE - Integer scalar, unformatted output file unit number.
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real workspace array; dimension at least LENRSK.
CSKWRK(*) - Character string workspace array; dimension at least LENCSK.

SKSDEN SKSDEN SKSDEN SKSDEN SKSDEN SKSDEN SKSDEN
****
****
******

SUBROUTINE SKSDEN (ISKWRK, RSKWRK, SDEN0)
Returns a real array of standard-state phase densities as given on input to the interpreter.

INPUT
RSKWRK(*) - Real workspace array; dimension at least LENRSK.

OUTPUT
SDEN0(*) - Real array; standard-state densities for the site types, AS READ BY THE INTERPRETER;
dimension at least NPHASE, the total phase count, but the subroutine only uses site phase entries, NFSURF <= N <= NLSURF.
cgs units, moles/cm**2.

```
SKSMH     SKSMH     SKSMH     SKSMH     SKSMH     SKSMH     SKSMH
          ****************************************
                    *************************
                              ******
```

```
SUBROUTINE SKSMH  (T, ISKWRK, RSKWRK, SMH)
Returns the array of dimensionless entropies minus enthalpies
for the species.  It is normally not called directly by the user.

INPUT
T(*)      - Real array, temperature(s); dimension is determined by
            the application program to be the total number of
            species temperatures, nominally 1.
               cgs units, K
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real    workspace array; dimension at least LENRSK.

OUTPUT
SMH(*)    - Real array, dimensionless entropies minus enthalpies
            for the species;
            dimension at least KKTOT, the total species count.
            SMH(K) = S(K)/R - H(K)/RT.
```

```
SKSML     SKSML     SKSML     SKSML     SKSML     SKSML     SKSML
          ****************************************
                    *************************
                              ******
```

```
SUBROUTINE SKSML  (T, ISKWRK, RSKWRK, SML)
Returns an array of the standard state entropies in molar units.

INPUT
T(*)      - Real array, temperature(s); dimension is determined by
            the application program to be the total number of
            species temperatures, nominally 1.
               cgs units, K
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real    workspace array; dimension at least LENRSK.

OUTPUT
SML(*)    - Real array, standard state entropies for the species;
            dimension at least KKTOT, the total species count.
               cgs units, ergs/(mole*K)
```

SKSMS SKSMS SKSMS SKSMS SKSMS SKSMS SKSMS

SUBROUTINE SKSMS (T, ISKWRK, RSKWRK, SMS)
Returns an array of the standard state entropies in mass units.

INPUT
T(*) - Real array, temperature(s); dimension is determined by the application program to be the total number of species temperatures, nominally 1.
cgs units, K
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real workspace array; dimension at least LENRSK.

OUTPUT
SMS(*) - Real array, standard state entropies for the species; dimension at least KKTOT, the total species count.
cgs units, ergs/(gm*K)

```
SKSNUM    SKSNUM    SKSNUM    SKSNUM    SKSNUM    SKSNUM    SKSNUM
          ****************************************
                    *************************
                              ******

SUBROUTINE SKSNUM (LINE, NEXP, LOUT, KNAM, KKTOT, PNAM, NNPHAS,
                   KKPHAS, KNUM, NT, NVAL, RVAL, KERR)
This subroutine is used to read a format-free input line of
combined alphanumerical data.  It can be used to parse an input
character string, LINE, which may be composed of several blank-
delimited substrings.  This subroutine assumes that the first
substring in LINE is the name of a species in the SURFACE CHEMKIN-III
mechanism.  If the species name is not unique within the SURFACE
CHEMKIN-III mechanism, the phase of the species should be input
immediately after the species name, delimited by slashes.
Upon return from the subroutine, KNUM returns the index position
of the species within the SURFACE CHEMKIN-III linkfile.  If the
species name is not unique, KNUM returns the first position and
NT returns the number of the times the species occurs within the
linkfile.  If the species name is not found, or there is a
syntax error, on return, KNUM=0, NT=0, and KERR=.TRUE.
The substrings in LINE following the first are expected to
represent numbers.  They are converted into floating point
values and stored in the output vector, RVAL(*).  Upon input,
NEXP is equal to the number of values expected to be found.
If NEXP numbers are not found, KERR will be set to .TRUE. on
return from the subroutine.
   Example input:
           LINE     = GA(S)/BULK1/ 1.2
           NEXP     = 1, the number of values expected
           LOUT     = 6, a logical unit number on which to write
                      diagnostic messages
           KNAM(*)  = Array of character species names
           KKTOT    = Total number of species
           PNAM(*)  = Array of character phase names
           NNPHAS   = Total number of phases
           KKPHAS(*)= Index array of the number of species in the
                      phases
   Output:
           KNUM     = The index number of the species which
                      has the name "GA(S)" and resides in phase
                      "BULK1"
           NT       = 1, if there is only one species GA(S)
                      in phase BULK1
           NVAL     = 1, the number of values found in LINE
                      following the species name
           RVAL(1)  = 1.200E+00, the substring converted to a
                      real number
           KERR     = .FALSE.

INPUT
LINE      - Character string; length depends on calling routine.
NEXP      - Integer scalar, number of values to be found in LINE.
            If NEXP < 0, then IABS(NEXP) values are expected, but
            it is not an error condition if less values are found.
LOUT      - Integer scalar, formatted output file unit number.
KNAM(*)   - Character string array, species names;
            dimension at least KKTOT, the total species count.
KKTOT     - Integer scalar, the total species count.
PNAM(*)   - Character string array, phase names;
            dimension at least NNPHAS, the total phase count.
NNPHAS    - Integer scalar, the total phase count.
KKPHAS(*) - Integer array, total species counts for the phases;
            dimension at least NNPHAS, the total phase count.
```

```
OUTPUT
KNUM      - Integer scalar, species index if the species name appears
            in LINE.
NT        - Integer scalar, number of times the species name occurs
            in the linkfile.
NVAL      - Integer scalar, number of value character strings found
            in LINE.
RVAL(*)   - Real array, real values for their character strings in
            LINE;
            dimension at least NEXP, the number of values expected.
KERR      - Logical, syntax or dimensioning error flag;



SKSOR     SKSOR     SKSOR     SKSOR     SKSOR     SKSOR     SKSOR
          ****************************************
                    *************************
                              ******

SUBROUTINE SKSOR  (T, ISKWRK, RSKWRK, SOR)
Returns an array of the nondimensional entropies.

INPUT
T(*)      - Real array, temperature(s); dimension is determined by
            the application program to be the total number of
            species temperatures, nominally 1.
               cgs units, K
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real    workspace array; dimension at least LENRSK.

OUTPUT
SOR(*)    - Real array, nondimensional entropies for the species;
            dimension at least KKTOT, the total species count.




SKSYME    SKSYME    SKSYME    SKSYME    SKSYME    SKSYME    SKSYME
          ****************************************
                    *************************
                              ******

SUBROUTINE SKSYME (ISKWRK, CSKWRK, LOUT, ENAM, KERR)
Returns a character string array of element names.

INPUT
CSKWRK(*) - Character string workspace array; dimension at least
            LENCSK.
LOUT      - Integer scalar, formatted output file unit number.

OUTPUT
ENAM(*)   - Character string array, element names;
            dimension at least NELEM, the total element count.
KERR      - Logical, character length error flag.
```

```
SKSYMM    SKSYMM    SKSYMM    SKSYMM    SKSYMM    SKSYMM    SKSYMM
          ****************************************
                    *************************
                              ******

SUBROUTINE SKSYMM (ISKWRK, CSKWRK, LOUT, MATNAM, KERR)
Returns the character string name of a material.

INPUT
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
CSKWRK(*) - Character string workspace array; dimension at least
            LENCSK.
LOUT      - Integer scalar, formatted output file unit number.

OUTPUT
MATNAM    - Character string, material name.
KERR      - Logical, character length error flag.

SKSYMP    SKSYMP    SKSYMP    SKSYMP    SKSYMP    SKSYMP    SKSYMP
          ****************************************
                    *************************
                              ******

SUBROUTINE SKSYMP (ISKWRK, CSKWRK, LOUT, PNAM, KERR)
Returns a character string array of phase names.

INPUT
CSKWRK(*) - Character string workspace array; dimension at least
            LENCSK.
LOUT      - Integer scalar, formatted output file unit number.

OUTPUT
PNAM(*)   - Character string array, phase names;
            dimension at least NNPHAS, the total phase count.
KERR      - Logical, character length error flag.



SKSYMR    SKSYMR    SKSYMR    SKSYMR    SKSYMR    SKSYMR    SKSYMR
          ****************************************
                    *************************
                              ******

SUBROUTINE SKSYMR (IR, LOUT, ISKWRK, RSKWRK, CSKWRK, LT, RNAM,
                   KERR)
Returns the character string representation of reaction IR.

INPUT
IR        - Integer scalar, reaction index.
LOUT      - Integer scalar, formatted output file unit number.
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real    workspace array; dimension at least LENRSK.
CSKWRK(*) - Character string workspace array; dimension at least
            LENCSK.

OUTPUT
LT        - Integer scalar, nunber of non-blank characters in the
            reaction string.
RNAM      - Character string, representation of reaction.
KERR      - Logical, character length error flag.
```

```
SKSYMS    SKSYMS    SKSYMS    SKSYMS    SKSYMS    SKSYMS    SKSYMS
          ******************************************
                    *************************
                              ******
```

SUBROUTINE SKSYMS (ISKWRK, CSKWRK, LOUT, KNAM, KERR)
Returns a character array of species names.

INPUT
CSKWRK(*) - Character string workspace array; dimension at least
  LENCSK.
LOUT - Integer scalar, formatted output file unit number.

OUTPUT
KNAM(*) - Character string array, species names;
  dimension at least KKTOT, the total species count.
KERR - Logical, character length error flag.

```
SKUML     SKUML     SKUML     SKUML     SKUML     SKUML     SKUML
          ******************************************
                    *************************
                              ******
```

SUBROUTINE SKUML (T, ISKWRK, RSKWRK, UML)
Returns an array of the internal energies in molar units.

INPUT
T(*) - Real array, temperature(s); dimension is determined by
  the application program to be the total number of
  species temperatures, nominally 1.
     cgs units, K
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real   workspace array; dimension at least LENRSK.

OUTPUT
UML(*) - Real array, internal energies of the species;
  dimension at least KKTOT, the total species count.
     cgs units, ergs/mole

```
SKUMS     SKUMS     SKUMS     SKUMS     SKUMS     SKUMS     SKUMS
          ******************************************
                    *************************
                              ******
```

SUBROUTINE SKUMS (T, ISKWRK, RSKWRK, UMS)
Returns an array of the internal energies in mass units.

INPUT
T(*) - Real array, temperature(s); dimension is determined by
  the application program to be the total number of
  species temperatures, nominally 1.
     cgs units, K
ISKWRK(*) - Integer workspace array; dimension at least LENISK.
RSKWRK(*) - Real   workspace array; dimension at least LENRSK.

OUTPUT
UMS(*) - Real array, internal energies of the species;
  dimension at least KKTOT, the total species count.
     cgs units, ergs/gm

```
SKWT      SKWT      SKWT      SKWT      SKWT      SKWT      SKWT
          ******************************************
                    *************************
                              ******

SUBROUTINE SKWT   (ISKWRK, RSKWRK, WT)
Returns the molecular weights of the species.

INPUT
RSKWRK(*) - Real    workspace array; dimension at least LENRSK.

OUTPUT
WT(*)     - Real array, molecular masses for the species;
            dimension at least KKTOT, the total species count.
               cgs units, gm/mole
```

IX. SAMPLE PROBLEM

The problem that we have chosen for an example involves the time-dependent deposition of a solid film on the surfaces of a fixed-volume, fixed-temperature container that was initially filled with a gas mixture. As with any new application for the SURFACE CHEMKIN-III package, one of the first tasks is to derive a system of equations that describes the process to be simulated. Here, the first equation involves the conservation of mass in the container:

$$\frac{dm_k}{dt} = V\dot{\omega}_k W_k + A\dot{s}_k W_k, \qquad (k = 1,\ldots,K_g) \tag{79}$$

where t is time, m_k is the mass of gas-phase species k in the container, V is the container volume, $\dot{\omega}_k$ is the molar production (destruction) rate of gas-phase species by gas-phase chemical reactions, W_k are the species molecular weights, A is the container-wall surface area, and $\dot{s}_k$ is the molar production rate of gas-phase species by surface reactions. After introducing the gas-phase mass density $\rho = m/V$ (where m is the total gas-phase mass) and the gas-phase species mass fractions $Y_k = m_k/m$, some manipulation leads to the following equation:

$$\frac{dY_k}{dt} = -\frac{Y_k}{\rho}\frac{d\rho}{dt} + \frac{\dot{\omega}_k W_k}{\rho} + \frac{1}{\rho}\frac{A}{V}\dot{s}_k W_k. \qquad (k = 1,\ldots,K_g) \tag{80}$$

The total mass in the gas phase depends on the production (destruction) of gas-phase species by surface reaction, as stated by

$$\frac{dm}{dt} = \sum_{k=1}^{K_g} A\dot{s}_k W_k. \tag{81}$$

We rewrite this equation slightly to make ρ a dependent variable and use the area-to-volume A/V as a parameter:

$$\frac{d\rho}{dt} = \sum_{k=1}^{K_g} \frac{A}{V}\dot{s}_k W_k. \tag{82}$$

On the surface, the number of moles of species k is given by

$$N_k = Z_k(n)\Gamma_n A/\sigma_k(n), \tag{83}$$

where $Z_k(n)$ is the site fraction of species *k* on surface site n (fraction of sites occupied by species *k* in phase n), Γ_k is the density of sites in phase n (in moles/cm^2), and $\sigma_k(n)$ is the number of sites that species *k* occupies. The molar production (destruction) rate of surface species *k* by surface reaction is stated as

$$\frac{dN_k}{dt} = As_k \qquad (k = K_s^f, ..., K_s^l). \tag{84}$$

In terms of site fractions $Z_k(n)$, the equation governing the surface species is given as

$$\frac{dZ_k(n)}{dt} = \frac{\dot{s}_k \sigma_k(n)}{\Gamma_k} - \frac{Z_k(n)}{\Gamma_k}\frac{d\Gamma_k}{dt}, \tag{85}$$

where the last term can be dropped if the number of surface sites if fixed. When surface reactions create or destroy sites, then a conservation equation is included for the site densities of each phase n:

$$\frac{d\Gamma_n}{dt} = \dot{\Gamma}_n. \tag{86}$$

However, our sample mechanism conserves sites, so the time derivative in Eq. (86) is trivially zero.

The governing system of ordinary differential equations and accompanying initial conditions form an initial value problem. The equations will be solved using the code VODE.[12] We find this code to be highly reliable for the solution of a wide range of stiff initial-value problems.

The Fortran code for solution of the sample problem is given in Section 7 below. After initializing the gas-phase CHEMKIN-III Library and the SURFACE CHEMKIN-III Library, the code reads the initial nonzero moles from input. It then repeatedly calls subroutine VODE to obtain the solution at uniform print intervals. The governing equation formulation is found in SUBROUTINE FUN, which is called by VODE.

The sections below present a Unix shell script for the sample problem, CHEMKIN-III Interpreter input and output, SURFACE CHEMKIN-III Interpreter input and output, the input to the sample problem, Fortran code for the sample problem, and output for the sample program. The last section describes how to use VODE.

Discussion of Sample Problem

We illustrate the input and output of the various Interpreters and the example problem with an analysis of the deposition of Si_3N_4. The gas-phase reaction mechanism contains a detailed description of NH_3 decomposition (about which there is much published information), two reactions describing SiF_4 decomposition, and three cross-reactions. At the low pressures we consider, the gas-phase decomposition of reactants is slow. The surface reaction mechanism contains six steps describing the overall conversion of 3 SiF_4 and 4 NH_3 molecules to 3 Si(d) and 4 N(d) and 12 HF. (Note that the surface reaction mechanism is from a preliminary analysis at one temperature, and thus we have not supplied any activation energies. Also, the thermodynamic data in the mechanism is contrived and should not be used in other contexts. As such, this mechanism should only be considered as illustrative and not as a source of kinetic data on the Si_3N_4 system.)

The input to the sample problem gives the initial pressure as 2.63×10^{-3} atm (2 Torr) and temperature 1713 K. In this problem the temperature is fixed, but the pressure will increase as 12 moles of HF are produced for every 7 moles of reactant destroyed. The input gas-phase mole fractions represent a 6:1 ratio of NH_3 to SiF_4. Initial site fractions of the surface species came from a steady-state analysis of the system (not discussed here). The initial activities of the two bulk species are set to 1. The area to volume ratio is 6 (a cubic box).

The print-out from the sample problem shows the initial conditions followed by print-outs of the concentrations at subsequent states of the deposition. The reactants SiF_4 and NH_3 are seen to be depleted and the product HF forms. In this fixed-volume system the pressure rises (discussed above). There is a net decrease in the gas density as the heavy Si and N atoms are lost from the gas into the bulk.

1. Unix Shell Script for Running Sample Problem

```
#!/bin/sh
# to execute:  sh sksamp.sh myrun &

sh 1> ${1}.log 2>&1 << ENDSH   # shell run output is "myrun.log"

set -x                         # echo all commands to stdout

#cd /scr/$LOGNAME              #go to user's scratch directory
#mkdir "${1}$$"                #make subdirectory /myrun##
#cd "${1}$$"                   #go to /myrun##

cat << EOF > makefile

#*****compiler > cray
#FFLAGS =
#LINK   = segldr -f zeros -o
#MACH = cmach.o
#*****END compiler > cray

#*****compiler > sun
#FFLAGS = -static -O2
#LINK = f77 -o
#MACH = dmach.o
#*****END compiler > sun

#*****compiler > sgi R3000
#FFLAGS = -static -O2 -mips1
#LINK   = f77 -o
#MACH = dmach.o
#*****END compiler > sgi R3000

#*****compiler > sgi R4X00
FFLAGS = -g -mips4
LINK   = f77 -o
MACH = dmach.o
#*****END compiler > sgi R4X00

#*****compiler > ibm rsc6000
#FFLAGS = -static -O2
#LINK    = xlf -o
#MACH  = dmach.o
#*****END compiler > ibm rsc6000

#*****compiler > alpha
#FFLAGS = -static -O2
#LINK   = f77 -o
#MACH = amach.o
#*****END compiler > alpha

#*****compiler > HP
#FFLAGS = -K -O
#LINK = f77 -o
#MACH = dmach.o
#*****END compiler > HP


OBJS = sksample.o cklib.o sklib.o vode.o math.o
INPS = therm.dat chem.inp surf.inp sksamp.inp
OUTS = chem.bin chem.out surf.bin surf.out sksamp.out
EXES = chem.exe surf.exe sksamp.exe
```

```
chem.exe: ckinterp.o
          $(LINK) chem.exe ckinterp.o

surf.exe : skinterp.o cklib.o
           $(LINK) surf.exe skinterp.o cklib.o

sksamp.exe : $(OBJS)
           $(LINK) sksamp.exe $(OBJS)
EOF

touch makefile; make sksamp.exe
chem.exe < chem.inp > chem.out
surf.exe < surf.inp > surf.out
sksamp.exe < sksamp.inp > sksamp.out

ENDSH
```

2. Input to CHEMKIN-III Interpreter

```
ELEMENTS H N SI F
END
SPECIES
H2 H N2 N NH NH2 NNH N2H2 N2H3 N2H4
HF F SIF4 SIF3 SIHF3 SIF3NH2 NH3
END

REACTIONS
H+H+M=H2+M                    0.100E+19 -1.000      0.000 ! D-L
 H2/0.0/
H+H+H2=H2+H2                  0.920E+17 -0.600      0.000
NH+N=N2+H                     0.300E+14  0.000      0.000 ! JAM
NH+H=N+H2                     0.100E+15  0.000      0.000 ! NH3  CST
NH2+H=NH+H2                   0.692E+14  0.000   3650.000
NH3+H=NH2+H2                  0.636E+06  2.390  10171.000 ! MICHAEL
NNH=N2+H                      0.100E+05  0.000      0.000 ! JAM
NNH+H=N2+H2                   0.100E+15  0.000      0.000 ! JAM
NNH+NH2=N2+NH3                0.500E+14  0.000      0.000 ! JAM
NNH+NH=N2+NH2                 0.500E+14  0.000      0.000 ! JAM
NH2+NH=N2H2+H                 0.500E+14  0.000      0.000 ! NH3CST
NH+NH=N2+H+H                  0.254E+14  0.000      0.000 ! NH3 CST
NH2+N=N2+H+H                  0.720E+14  0.000      0.000 ! PG
N2H2+M=NNH+H+M                0.500E+17  0.000  50000.000 ! NH3 CST
    N2/2/ H2/2/
N2H2+H=NNH+H2                 0.500E+14  0.000   1000.000 ! NH3 CST
N2H2+NH=NNH+NH2               0.100E+14  0.000   1000.000 ! NH3 CST
N2H2+NH2=NH3+NNH              0.100E+14  0.000   1000.000 ! NH3 CST
NH2+NH2=N2H2+H2               0.500E+12  0.000      0.000 ! NH3  CST
NH3+M=NH2+H+M                 0.140E+17  0.000  90600.000 ! MSGK
N2H3+H=NH2+NH2                1.60E+12   0.0        0.0   ! MSGK
N2H3+M=N2H2+H+M               3.50E+16   0.0    46000.0   ! MSGK
N2H3+NH=NH2+N2H2              2.00E+13   0.0        0.0   ! MSGK
NH2+NH2+M=N2H4+M              3.00E+20  -1.0        0.0   ! MSGK
H+N2H4=H2+N2H3                1.30E+13   0.0     2500.0   ! MSGK
NH2+N2H4=NH3+N2H3             3.90E+12   0.0     1500.0   ! MSGK
NH+H+M=NH2+M                  2.00E+16  -0.5        0.0   ! MSGK
NH2+NH2=NH3+NH                5.00E+12   0.0    10000.0   ! MSGK
F+NH3=NH2+HF                  4.27E+11   0.5      800.0   ! KONDRATIEV
SIF4=SIF3+F                   3.00E+12   0.0   147170.0   ! PHO&MEC
H+SIF4=HF+SIF3                1.00E+13   0.0    50000.0   ! PHO&MEC
NH2+SIF4=SIF3NH2+F            1.00E+11   0.0    40950.0   ! GUESS
NH3+SIF3=SIF3NH2+H            1.00E+11   0.0     5000.0   ! GUESS
NH3+SIF3=SIHF3+NH2            1.00E+11   0.0    10000.0   ! PHO&MEC
END
```

3. Output from CHEMKIN-III Interpreter

```
CHEMKIN-III GAS-PHASE MECHANISM INTERPRETER:
DOUBLE PRECISION Vers. 5.1 April 24, 1996
Copyright 1995, Sandia Corporation.
The U.S. Government retains a limited license in this software.
```

ELEMENTS CONSIDERED	ATOMIC WEIGHT
1. H	1.00797
2. N	14.0067
3. SI	28.0860
4. F	18.9984

SPECIES CONSIDERED	PHASE	CHARGE	MOLECULAR WEIGHT	TEMPERATURE LOW	TEMPERATURE HIGH	ELEMENT COUNT H	N	SI	F
1. H2	G	0	2.01594	300	5000	2	0	0	0
2. H	G	0	1.00797	300	5000	1	0	0	0
3. N2	G	0	28.01340	300	5000	0	2	0	0
4. N	G	0	14.00670	300	5000	0	1	0	0
5. NH	G	0	15.01467	300	5000	1	1	0	0
6. NH2	G	0	16.02264	300	5000	2	1	0	0
7. NNH	G	0	29.02137	250	4000	1	2	0	0
8. N2H2	G	0	30.02934	300	5000	2	2	0	0
9. N2H3	G	0	31.03731	300	5000	3	2	0	0
10. N2H4	G	0	32.04528	300	5000	4	2	0	0
11. HF	G	0	20.00637	300	5000	1	0	0	1
12. F	G	0	18.99840	300	5000	0	0	0	1
13. SIF4	G	0	104.07960	300	2000	0	0	1	4
14. SIF3	G	0	85.08120	300	3000	0	0	1	3
15. SIHF3	G	0	86.08917	300	3000	1	0	1	3
16. SIF3NH2	G	0	101.10384	300	3000	2	1	1	3
17. NH3	G	0	17.03061	300	5000	3	1	0	0

REACTIONS CONSIDERED	(k = A T**b exp(-E/RT)) A	b	E
1. H+H+M=H2+M	1.00E+18	-1.0	.0
H2 Enhanced by .000E+00			
2. H+H+H2=H2+H2	9.20E+16	-.6	.0
3. NH+N=N2+H	3.00E+13	.0	.0
4. NH+H=N+H2	1.00E+14	.0	.0
5. NH2+H=NH+H2	6.92E+13	.0	3650.0
6. NH3+H=NH2+H2	6.36E+05	2.4	10171.0
7. NNH=N2+H	1.00E+04	.0	.0
8. NNH+H=N2+H2	1.00E+14	.0	.0
9. NNH+NH2=N2+NH3	5.00E+13	.0	.0
10. NNH+NH=N2+NH2	5.00E+13	.0	.0
11. NH2+NH=N2H2+H	5.00E+13	.0	.0
12. NH+NH=N2+H+H	2.54E+13	.0	.0
13. NH2+N=N2+H+H	7.20E+13	.0	.0

```
 14. N2H2+M=NNH+H+M                                  5.00E+16     .0    50000.0
     N2              Enhanced by    2.000E+00
     H2              Enhanced by    2.000E+00
 15. N2H2+H=NNH+H2                                   5.00E+13     .0     1000.0
 16. N2H2+NH=NNH+NH2                                 1.00E+13     .0     1000.0
 17. N2H2+NH2=NH3+NNH                                1.00E+13     .0     1000.0
 18. NH2+NH2=N2H2+H2                                 5.00E+11     .0        .0
 19. NH3+M=NH2+H+M                                   1.40E+16     .0    90600.0
 20. N2H3+H=NH2+NH2                                  1.60E+12     .0        .0
 21. N2H3+M=N2H2+H+M                                 3.50E+16     .0    46000.0
 22. N2H3+NH=NH2+N2H2                                2.00E+13     .0        .0
 23. NH2+NH2+M=N2H4+M                                3.00E+20   -1.0        .0
 24. H+N2H4=H2+N2H3                                  1.30E+13     .0     2500.0
 25. NH2+N2H4=NH3+N2H3                               3.90E+12     .0     1500.0
 26. NH+H+M=NH2+M                                    2.00E+16    -.5        .0
 27. NH2+NH2=NH3+NH                                  5.00E+12     .0    10000.0
 28. F+NH3=NH2+HF                                    4.27E+11     .5      800.0
 29. SIF4=SIF3+F                                     3.00E+12     .0   147170.0
 30. H+SIF4=HF+SIF3                                  1.00E+13     .0    50000.0
 31. NH2+SIF4=SIF3NH2+F                              1.00E+11     .0    40950.0
 32. NH3+SIF3=SIF3NH2+H                              1.00E+11     .0     5000.0
 33. NH3+SIF3=SIHF3+NH2                              1.00E+11     .0    10000.0

 NOTE:  A units mole-cm-sec-K, E units cal/mole


NO ERRORS FOUND ON INPUT,
BINARY Vers. 1.0 CHEMKIN linkfile chem.bin written.

WORKING SPACE REQUIREMENTS ARE
   INTEGER:     1101
   REAL:         771
   CHARACTER:     21
```

4. Input to SURFACE CHEMKIN-III Interpreter

```
SITE/SI3N4/    SDEN/4.1683e-9/
   NHSIF(S)/2/
   SIF3NH2(S)/2/
   SIF2NH(S)/2/
   NH2SIFNH(S)/2/
   NHSIFNHSIFNH(S)/4/
   NHNH2(S)/2/
END
BULK SI(D)/2.066/
BULK N(D) /1.374/
END
THERMO ALL
  300.      600.     1685.
NHSIF(S)          J 3/67N   1H   1SI  1F   1S   300.000  1685.000             1
 0.24753989E 01 0.88112187E-03-0.20939481E-06 0.42757187E-11 0.16006564E-13    2
-0.81255620E 03-0.12188747E 02 0.84197538E 00 0.83710416E-02-0.13077030E-04    3
 0.97593603E-08-0.27279380E-11-0.52486288E 03-0.45272678E 01                  4
NHNH2(S)          J 3/67N   2H   3SI  0F   0S   300.000  1685.000             1
 0.24753989E 01 0.88112187E-03-0.20939481E-06 0.42757187E-11 0.16006564E-13    2
-0.81255620E 03-0.12188747E 02 0.84197538E 00 0.83710416E-02-0.13077030E-04    3
 0.97593603E-08-0.27279380E-11-0.52486288E 03-0.45272678E 01                  4
SIF3NH2(S)        J 3/67N   1H   2SI  1F   3S   300.000  1685.000             1
 0.24753989E 01 0.88112187E-03-0.20939481E-06 0.42757187E-11 0.16006564E-13    2
-0.81255620E 03-0.12188747E 02 0.84197538E 00 0.83710416E-02-0.13077030E-04    3
 0.97593603E-08-0.27279380E-11-0.52486288E 03-0.45272678E 01                  4
SIF2NH(S)         J 3/67N   1H   1SI  1F   2S   300.000  1685.000             1
 0.24753989E 01 0.88112187E-03-0.20939481E-06 0.42757187E-11 0.16006564E-13    2
-0.81255620E 03-0.12188747E 02 0.84197538E 00 0.83710416E-02-0.13077030E-04    3
 0.97593603E-08-0.27279380E-11-0.52486288E 03-0.45272678E 01                  4
NH2SIFNH(S)       J 3/67N   2H   3SI  1F   1S   300.000  1685.000             1
 0.24753989E 01 0.88112187E-03-0.20939481E-06 0.42757187E-11 0.16006564E-13    2
-0.81255620E 03-0.12188747E 02 0.84197538E 00 0.83710416E-02-0.13077030E-04    3
 0.97593603E-08-0.27279380E-11-0.52486288E 03-0.45272678E 01                  4
NHSIFNHSIFNH(S)   J 3/67N   3H   3SI  2F   2S   300.000  1685.000             1
 0.24753989E 01 0.88112187E-03-0.20939481E-06 0.42757187E-11 0.16006564E-13    2
-0.81255620E 03-0.12188747E 02 0.84197538E 00 0.83710416E-02-0.13077030E-04    3
 0.97593603E-08-0.27279380E-11-0.52486288E 03-0.45272678E 01                  4
SI(D)             J 3/67SI  100  000  000  0S   300.000  1685.000             1
 0.24753989E 01 0.88112187E-03-0.20939481E-06 0.42757187E-11 0.16006564E-13    2
-0.81255620E 03-0.12188747E 02 0.84197538E 00 0.83710416E-02-0.13077030E-04    3
 0.97593603E-08-0.27279380E-11-0.52486288E 03-0.45272678E 01                  4
N(D)              J 3/67N   100  000  000  0S   300.000  1685.000             1
 0.24753989E 01 0.88112187E-03-0.20939481E-06 0.42757187E-11 0.16006564E-13    2
-0.81255620E 03-0.12188747E 02 0.84197538E 00 0.83710416E-02-0.13077030E-04    3
 0.97593603E-08-0.27279380E-11-0.52486288E 03-0.45272678E 01                  4
END

REACTIONS
 NH3             + NHSIF(S)  => NHNH2(S)   + SI(D) + HF    7.5620E08 0.5 0.0
 SIF4            + NHNH2(S)  => SIF3NH2(S) + N(D)  + HF    3.0967E08 0.5 0.0
 SIF3NH2(S)                  => SIF2NH(S)          + HF    1.0000E05 0.0 0.0
 NH3             + SIF2NH(S) => NH2SIFNH(S)        + HF    7.5620E08 0.5 0.0
 NH2SIFNH(S)     + SIF2NH(S) => NHSIFNHSIFNH(S)    + HF    1.0000E15 0.0 0.0
 NHSIFNHSIFNH(S) + SIF2NH(S) => 3.0NHSIF(S)  + N(D)  + HF    1.0000E15 0.0 0.0
END
```

5. Output from SURFACE CHEMKIN-III Interpreter

```
CHEMKIN-III SURFACE MECHANISM INTERPRETER:
DOUBLE PRECISION Vers. 5.1 April 24, 1996
Copyright 1995, Sandia Corporation.
The U.S. Government retains a limited license in this software.


CKLIB: CHEMKIN-III GAS-PHASE CHEMICAL KINETICS LIBRARY,
DOUBLE PRECISION Vers. 5.0 March 1, 1996
Copyright 1995, Sandia Corporation.
The U.S. Government retains a limited license in this software.
```

SPECIES CONSIDERED	MOLECULAR WEIGHT	Density	Nsites	ELEMENT COUNT H	N	SI	F
Gas phase species:							
1. H2	2.01594			2	0	0	0
2. H	1.00797			1	0	0	0
3. N2	28.01340			0	2	0	0
4. N	14.00670			0	1	0	0
5. NH	15.01467			1	1	0	0
6. NH2	16.02264			2	1	0	0
7. NNH	29.02137			1	2	0	0
8. N2H2	30.02934			2	2	0	0
9. N2H3	31.03731			3	2	0	0
10. N2H4	32.04528			4	2	0	0
11. HF	20.00637			1	0	0	1
12. F	18.99840			0	0	0	1
13. SIF4	104.07960			0	0	1	4
14. SIF3	85.08120			0	0	1	3
15. SIHF3	86.08917			1	0	1	3
16. SIF3NH2	101.10384			2	1	1	3
17. NH3	17.03061			3	1	0	0
SITE: SI3N4		.417E-08 moles/cm**2					
18. NHSIF(S)	62.09907		2	1	1	1	1
19. SIF3NH2(S)	101.10384		2	2	1	1	3
20. SIF2NH(S)	81.09747		2	1	1	1	2
21. NH2SIFNH(S)	78.12171		2	3	2	1	1
22. NHSIFNHSIFNH(S)	139.21281		4	3	3	2	2
23. NHNH2(S)	31.03731		2	3	2	0	0
BULK: BULK1							
24. SI(D)	28.08600	.207E+01 g/cm**3		0	0	1	0
BULK: BULK2							
25. N(D)	14.00670	.137E+01 g/cm**3		0	1	0	0

SURFACE REACTIONS CONSIDERED	(k = A T**b exp(-E/RT)) A	b	E
1. NH3+NHSIF(S)=>NHNH2(S)+SI(D)+HF	7.56E+08	.5	.0
2. SIF4+NHNH2(S)=>SIF3NH2(S)+N(D)+HF	3.10E+08	.5	.0
3. SIF3NH2(S)=>SIF2NH(S)+HF	1.00E+05	.0	.0
4. NH3+SIF2NH(S)=>NH2SIFNH(S)+HF	7.56E+08	.5	.0
5. NH2SIFNH(S)+SIF2NH(S)	1.00E+15	.0	.0

```
     =>NHSIFNHSIFNH(S)+HF
  6. NHSIFNHSIFNH(S)+SIF2NH(S)                    1.00E+15     .0         .0
     =>3.0NHSIF(S)+N(D)+HF

 NOTE:   A units moles, E units cal/mole
         Default Motz-Wise correction to sticking coefficients is turned ON.


NO ERRORS FOUND ON INPUT:
BINARY Version 1.0 surface linkfile surf.bin written.

WORKING SPACE REQUIREMENTS ARE
   INTEGER:      481
   REAL:         654
   CHARACTER:     34
```

6. Sample Problem Input

```
2.63e-3  1713
SIF4             0.14286
NH3              0.85714
NHSIF(S)         6.251E-2
NHNH2(S)         0.91587
SIF3NH2(S)       2.354E-4
SIF2NH(S)        2.0837E-2
NH2SIFNH(S)      1.806E-4
NHSIFNHSIFNH(S)  3.6127E-4
N(D)             1.0
SI(D)            1.0
END
6.0
5.0E-2   5.0E-3
```

```
      PROGRAM SKSAMP
C
C     Version 3.3:
C     CHANGES FOR VERSION 3.0 (3/15/94 F. Rupley)
C     1.  DOS/PC compatibility effort includes adding file names to
C         OPEN statements, removing unused variables in CALL lists,
C         unusued but possibly initialized variables.
C     CHANGES FOR VERSION 3.1 (7/14/94 F. Rupley)
C     1.  Add ISKWRK to some CALL SK lists.
C     CHANGES FOR VERSION 3.2 (1/19/95 F. Rupley)
C     1.  Add integer error flag to CKLEN,CKINIT,SKLEN,SKINIT
C         call lists.
C     CHANGES FOR VERSION 3.3 (2/27/95 F. Rupley)
C     1.  Change character index "(:" to "(1:"
C       Gas-phase and surface reaction in a constant volume isothermal
C       container with fixed surface area.
C
C*****precision > double
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER(I-N)
C*****END precision > double
C*****precision > single
C      IMPLICIT REAL (A-H,O-Z), INTEGER (I-N)
C*****END precision > single
C
      PARAMETER (LENIWK=4000, LENRWK=4000, LENCWK=200, LINCK=25,
     1           LINSK=26, NLMAX=55, LIN=5, LOUT=6, KMAX=50,
     2           MAXP=20, ITOL=1, IOPT=0, ITASK=1,
     3           RTOL=1.0E-6, ATOL=1.E-15)
C
      DIMENSION Z(KMAX), IWORK(LENIWK), RWORK(LENRWK)
C
      COMMON /RPAR/T, AVRAT, RU
      COMMON /IPAR/KKGAS, KKSURF, KKBULK, KKTOT, NFSURF, NLSURF,
     1             NFBULK, NLBULK, NISK, NIPKK, NIPKF, NIPKL,
     2             NRSK, NSDEN, NRCOV, NX, NWDOT, NWT, NACT,
     3             NSDOT, NSITDT
      CHARACTER*16 CWORK(LENCWK), KSYM(KMAX), PNAM(MAXP)
      DATA IWORK/LENIWK*0/, RWORK/LENRWK*0.0/, CWORK/LENCWK*' '/
      CHARACTER*80 LINE
C
      LOGICAL KERR, IERR
      EXTERNAL FUN
      DATA KERR/.FALSE./, KSYM/KMAX*' '/, PNAM/MAXP*' '/
C
C     Open the CHEMKIN and SURFACE LINK files
C
C*****gas linkfile > binary
      OPEN (LINCK,FORM='UNFORMATTED', STATUS='UNKNOWN',FILE='chem.bin')
C*****END gas linkfile > binary
C*****surface linkfile > binary
      OPEN (LINSK,FORM='UNFORMATTED', STATUS='UNKNOWN',FILE='surf.bin')
C*****END surface linkfile > binary
C*****gas linkfile > ascii
C      OPEN (LINCK, FORM='FORMATTED', STATUS='UNKNOWN', FILE='chem.asc')
C*****END gas linkfile > ascii
C*****surface linkfile > ascii
C      OPEN (LINSK, FORM='FORMATTED', STATUS='UNKNOWN', FILE='surf.asc')
C*****END surface linkfile > ascii
C
C     Find lengths necessary for arrays
C
```

```
      CALL CKLEN (LINCK, LOUT, LENI, LENR, LENC, IFLAG1)
      CALL SKLEN (LINSK, LOUT, LENIS, LENRS, LENCS, IFLAG2)
C
      IF (IFLAG1.GT.0 .OR. IFLAG2.GT.0) THEN
         WRITE (LOUT, *)
     1   'STOP...ERROR INITIALIZING LINKFILES...'
         STOP
      ENDIF
C
      LITOT = LENI + LENIS
      LRTOT = LENR + LENRS
      LCTOT = MAX(LENC, LENCS)
      IF (LITOT.GT.LENIWK .OR. LRTOT.GT.LENRWK .OR.LCTOT.GT.LENCWK) THEN
         IF (LITOT .GT. LENIWK) WRITE (LOUT, *)
     1   ' Error...LENIWK must be at least ', LITOT
         IF (LRTOT .GT. LENRWK) WRITE (LOUT, *)
     1   ' Error...LENRWK must be at least ', LRTOT
         IF (LCTOT .GT. LENCWK) WRITE (LOUT, *)
     1   ' Error...LENCWK must be at least ', LCTOT
         STOP
      ENDIF
C
C     Initialize CHEMKIN and SURFACE arrays
C
      CALL CKINIT (LENI, LENR, LENC, LINCK, LOUT, IWORK, RWORK,
     1             CWORK, IFLAG)
      IF (IFLAG .GT. 0) THEN
         WRITE (LOUT, *)
     1   'STOP...ERROR INITIALIZING CHEMKIN WORKSPACE...'
         STOP
      ENDIF
      CALL CKINDX (IWORK, RWORK, MM, KKGAS, II, NFIT)
C
      NISK = LENI + 1
      NRSK = LENR + 1
      CALL SKINIT (LENIS, LENRS, LENCS, LINSK, LOUT, IWORK(NISK),
     1             RWORK(NRSK), CWORK, IFLAG)
      IF (IFLAG .GT. 0) THEN
         WRITE (LOUT, *)
     1   'STOP...ERROR INITIALIZING SURFACE WORKSPACE...'
         STOP
      ENDIF
      CALL SKINDX (IWORK(NISK), NELEM, KKGAS, KKSUR, KKBULK, KKTOT,
     1             NNPHAS, NNSURF,
     2             NFSURF, NLSURF, NNBULK, NFBULK, NLBULK, IISUR)
C
      NIPKK = NISK  + LENIS
      NIPKF = NIPKK + NNPHAS
      NIPKL = NIPKF + NNPHAS
      NICOV = NIPKL + NNPHAS
      NEQ   = KKTOT + 1 + NNSURF
      NIODE = NICOV + KKTOT
      LIW   = 30 + NEQ
      ITOT  = NIODE + LIW - 1
C
      NSDEN = NRSK + LENRS
      NRCOV = NSDEN + NNPHAS
      NX    = NRCOV  + KKTOT
      NWDOT = NX     + KKTOT
      NWT   = NWDOT  + KKGAS
      NACT  = NWT    + KKTOT
      NSDOT = NACT   + KKTOT
      NSITDT= NSDOT  + KKTOT
      NRODE = NSITDT + NNPHAS
```

```
      LRW   = 22 + 9*NEQ + 2*NEQ**2
      NTOT  = NRODE + LRW - 1
C
      IF (LENIWK.LT.ITOT .OR. LENRWK.LT.NTOT .OR. KMAX.LT.KKTOT
     1   .OR. MAXP.LT.NNPHAS) THEN
          IF (LENIWK .LT. ITOT) WRITE (LOUT, *)
     1   ' Error...LENIWK too small...must be at least ', ITOT
          IF (LENRWK .LT. NTOT) WRITE (LOUT, *)
     1   ' Error...LENRWK too small...must be at least ', NTOT
          IF (KMAX .LT. KKTOT) WRITE (LOUT, *)
     1   ' Error...KMAX too small...must be at least ', KKTOT
          IF (MAXP .LT. NNPHAS) WRITE (LOUT, *)
     1   ' Error...MAXP too small...must be at least ', NNPHAS
          STOP
      ENDIF
C
      CALL SKPKK  (IWORK(NISK), IWORK(NIPKK), IWORK(NIPKF),
     1             IWORK(NIPKL))
      CALL SKSDEN (IWORK(NISK), RWORK(NRSK), RWORK(NSDEN))
      CALL SKCOV  (IWORK(NISK), IWORK(NICOV))
C
      DO 30 K = 1, KKTOT
C*****precision > double
         RWORK(NRCOV + K - 1) = DBLE(IWORK(NICOV + K - 1))
C*****END precision > double
C*****precision > single
C         RWORK(NRCOV + K - 1) = REAL(IWORK(NICOV + K - 1))
C*****END precision > single
          RWORK(NX + K - 1) = 0.0
   30 CONTINUE
C
      CALL SKSYMS (IWORK(NISK), CWORK, LOUT, KSYM, IERR)
      KERR = KERR.OR.IERR
      CALL SKSYMP (IWORK(NISK), CWORK, LOUT, PNAM, IERR)
      KERR = KERR.OR.IERR
      CALL SKWT   (IWORK(NISK), RWORK(NRSK), RWORK(NWT))
      CALL SKRP   (IWORK(NISK), RWORK(NRSK), RU, RUC, PATM)
      IF (KERR) THEN
         WRITE (LOUT, *)
     1   'STOP...ERROR INITIALIZING CONSTANTS...'
         STOP
      ENDIF
C
C     Pressure and temperature
C
      WRITE (LOUT, '(/A)')
     1               ' INPUT INITIAL PRESSURE(ATM) AND TEMPERATURE(K)'
      READ  (LIN,    *) PA, T
      WRITE (LOUT,7105) PA, T
      P = PA*PATM
C
C     Initial non-zero moles
C
   40 CONTINUE
      LINE = ' '
      WRITE (LOUT, '(/A)') ' INPUT INITIAL ACTIVITY OF NEXT SPECIES'
      READ  (LIN,  '(A)', END=45)   LINE
      WRITE (LOUT, '(1X,A)') LINE
      ILEN = INDEX (LINE, '!')
      IF (ILEN .EQ. 1) GO TO 40
C
      ILEN = ILEN - 1
      IF (ILEN .LE. 0) ILEN = LEN(LINE)
      IF (INDEX(LINE(1:ILEN), 'END') .EQ. 0) THEN
```

```
         IF (LINE(1:ILEN) .NE. ' ') THEN
            CALL SKSNUM (LINE(1:ILEN), 1, LOUT, KSYM, KKTOT, PNAM,
     1                   NNPHAS, IWORK(NIPKK), KNUM, NKF, NVAL,
     2                   VAL, IERR)
            IF (IERR) THEN
               WRITE (LOUT,*) ' Error reading moles...'
               KERR = .TRUE.
            ELSE
               RWORK(NX + KNUM - 1) = VAL
            ENDIF
         ENDIF
         GO TO 40
      ENDIF
C
   45 CONTINUE
      IF (KERR) THEN
         WRITE (LOUT, *) 'STOP...ERROR INITIALIZING USER INPUT...'
         STOP
      ENDIF
C
C     Surface area to volume ratio
C
      WRITE (LOUT, '(/A)') ' INPUT SURFACE AREA TO VOLUME RATIO'
      READ  (LIN,    *) AVRAT
      WRITE (LOUT,7105) AVRAT
C
C     Final time and print interval
C
      WRITE (LOUT, '(/A)') ' INPUT FINAL TIME AND DT'
      READ  (LIN,    *) T2, DT
      WRITE (LOUT,7105) T2, DT
C
C     Normalize the mole fractions for each phase
C
      DO 60 N = 1, NNPHAS
         XTOT = 0.0
         KFIRST = IWORK(NIPKF + N - 1)
         KLAST  = IWORK(NIPKL + N - 1)
         DO 50 K = KFIRST, KLAST
            XTOT = XTOT + RWORK(NX + K - 1)
   50    CONTINUE
         IF (XTOT .NE. 0.0) THEN
            DO 55 K = KFIRST, KLAST
               RWORK(NX + K - 1) = RWORK(NX + K - 1) / XTOT
   55       CONTINUE
         ELSE
            WRITE (LOUT, *)
     1      ' ERROR...NO SPECIES WERE INPUT FOR PHASE '//PNAM(N)
            KERR = .TRUE.
         ENDIF
   60 CONTINUE
      IF (KERR) THEN
         WRITE (LOUT, *) 'STOP...ERROR INITIALIZING SOLUTION...'
         STOP
      ENDIF
C
C     Initial conditions
C
      TT1  = 0.0
C     Initial gas-phase mass fractions
      CALL CKXTY (RWORK(NX), IWORK, RWORK, Z)
C     Initial surface site fractions
      KFIRST = IWORK(NIPKF + NFSURF - 1)
      KLAST  = IWORK(NIPKL + NLSURF - 1)
```

```
      DO 110 K = KFIRST, KLAST
         Z(K) = RWORK(NX + K - 1)
  110 CONTINUE
C     Initial bulk deposit amounts
      KFIRST = IWORK(NIPKF + NFBULK - 1)
      KLAST  = IWORK(NIPKL + NLBULK - 1)
      DO 120 K = KFIRST, KLAST
         Z(K) = 0.0
  120 CONTINUE
C     Initial gas-phase mass density
      CALL CKRHOY (P, T, Z, IWORK, RWORK, Z(KKTOT+1) )
C     Initial surface site densities
      DO 130 N = NFSURF, NLSURF
         Z(KKTOT+1+N-NFSURF+1) = RWORK(NSDEN + N - 1)
  130 CONTINUE
C
C     Integration control parameters for LSODE
C
      TT2   = TT1
      MF    = 22
      ISTATE= 1
C
C     Integration loop
C
  250 CONTINUE
C
C           Print the solution
C
      CALL CKPY  (Z(KKTOT+1), T, Z(1), IWORK, RWORK, P)
      WRITE (LOUT,*) ' '
      WRITE (LOUT,*) ' TIME = ', TT2
      WRITE (LOUT, 7100) P, T, Z(KKTOT+1)
      WRITE (LOUT, *) ' GAS-PHASE MOLE FRACTIONS'
      CALL CKYTX (Z, IWORK, RWORK, RWORK(NX))
      CALL PRT1  (KKGAS, KSYM, LOUT, RWORK(NX))
C
      DO 190 N = NFSURF, NLSURF
         WRITE (LOUT, *) ' SURFACE SITE FRACTIONS ON PHASE (SITE) ', N
         KKPHAS = IWORK(NIPKK + N - 1)
         KFIRST = IWORK(NIPKF + N - 1)
         CALL PRT1 (KKPHAS, KSYM(KFIRST), LOUT, Z(KFIRST))
C
         SUM = 0.0
         KFIRST = IWORK(NIPKF + N - 1)
         KLAST  = IWORK(NIPKL + N - 1)
         DO 185 K = KFIRST, KLAST
            SUM = SUM + Z(K)
  185    CONTINUE
         WRITE (LOUT,*)'  SUM OF SURFACE SITE FRACTIONS', SUM
         WRITE (LOUT,*)'  SURFACE SITE DENSITY  ', Z(KKTOT+1+N-NFSURF+1)
  190 CONTINUE
C
      DO 195 N = NFBULK, NLBULK
         WRITE (LOUT, *) ' BULK DEPOSITION (GM/CM**2) IN PHASE ', N
         KKPHAS = IWORK(NIPKK + N - 1)
         KFIRST = IWORK(NIPKF + N - 1)
         CALL PRT1 (KKPHAS, KSYM(KFIRST), LOUT, Z(KFIRST))
  195 CONTINUE
C
      IF (TT2 .GE. T2) THEN
         WRITE (LOUT, *) 'STOP...TIME LIMIT REACHED...'
         STOP
      ENDIF
      TT2 = MIN(TT2 + DT, T2)
```

```
C
C     Call the differential equation solver
C
  350 CONTINUE
C*****precision > single
C      CALL SVODE
C*****END precision > single
C*****precision > double
      CALL DVODE
C*****END precision > double
     *           (FUN, NEQ, Z, TT1, TT2, ITOL, RTOL, ATOL, ITASK,
     1            ISTATE, IOPT, RWORK(NRODE), LRW, IWORK(NIODE),
     2            LIW, JAC, MF, RWORK, IWORK)
C
      IF (ISTATE .LE. -1) THEN
         IF (ISTATE .EQ. -1) THEN
            ISTATE = 2
            GO TO 350
         ELSE
            WRITE (LOUT,*) 'ERROR, ISTATE=',ISTATE
            STOP
         ENDIF
      ENDIF
      GO TO 250
C
 7003 FORMAT (1H1)
 7100 FORMAT (1H , ' GAS-PHASE STATE', /,
     1   '  P = ', 1PE12.4, ' T = ', 1PE12.4, ' DENSITY = ', 1PE12.4)
 7105 FORMAT (12E11.3)
 7110 FORMAT (26X, 5(1X,A10))
 7115 FORMAT (22X, 10E11.3)
C
      END
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE FUN (NEQ, TIME, Z, ZP, RWORK, IWORK)
C
C*****precision > double
      IMPLICIT DOUBLE PRECISION(A-H,O-Z), INTEGER(I-N)
C*****END precision > double
C*****precision > single
C      IMPLICIT REAL (A-H,O-Z), INTEGER(I-N)
C*****END precision > single
C
      DIMENSION Z(NEQ), ZP(NEQ), RWORK(*), IWORK(*)
C
      COMMON /RPAR/T, AVRAT, RU
      COMMON /IPAR/KKGAS, KKSURF, KKBULK, KKTOT, NFSURF, NLSURF,
     1             NFBULK, NLBULK, NISK, NIPKK, NIPKF, NIPKL,
     2             NRSK, NSDEN, NRCOV, NX, NWDOT, NWT, NACT,
     3             NSDOT, NSITDT
C
C
C     Variables in Z are:  Z(K) = Y(K), K=1,KKGAS
C                          Z(K) = SURFACE SITE FRACTIONS,
C                                 K=KFIRST(NFSURF), KLAST(NLSURF)
C                          Z(K) = BULK SPECIES MASS,
C                                 K=KFIRST(NFBULK), KLAST(NLBULK)
C                          Z(K) = GAS-PHASE MASS DENSITY, K=KKTOT+1
C                          Z(K) = SURFACE SITE MOLAR DENSITIES,
C                                 K=KKTOT+2, KKTOT+1+NNSURF
C
C        Call CHEMKIN and SURFACE CHEMKIN subroutines
```

```
C
      CALL CKPY  (Z(KKTOT+1), T, Z(1), IWORK, RWORK, P)
      CALL CKWYP (P, T, Z(1), IWORK, RWORK, RWORK(NWDOT))
      CALL CKYTX (Z, IWORK, RWORK, RWORK(NACT))
C
      KFIRST = IWORK(NIPKF + NFSURF - 1)
      KLAST  = IWORK(NIPKL + NLSURF - 1)
      DO 100 K = KFIRST, KLAST
         RWORK(NACT + K - 1) = Z(K)
  100 CONTINUE
C
      KFIRST = IWORK(NIPKF + NFBULK - 1)
      KLAST  = IWORK(NIPKL + NLBULK - 1)
      DO 150 K = KFIRST, KLAST
         RWORK(NACT + K - 1) = RWORK(NX + K - 1)
150   CONTINUE
C
      DO 175 N = NFSURF, NLSURF
         RWORK(NSDEN + N - 1) = Z(KKTOT+1+N-NFSURF+1)
  175 CONTINUE
C
      CALL SKRAT (P, T, RWORK(NACT), RWORK(NSDEN), IWORK(NISK),
     1            RWORK(NRSK), RWORK(NSDOT), RWORK(NSITDT))
C
C        Form mass density equation
C
      SUM = 0.0
      DO 200 K = 1, KKGAS
         SUM = SUM + AVRAT * RWORK(NSDOT+K-1) * RWORK(NWT+K-1)
 200  CONTINUE
      ZP(KKTOT+1) = SUM
C
C        Form the gas-phase mass conservation equation
C
      DO 300 K = 1, KKGAS
         WDOT = RWORK(NWDOT + K - 1)
         WT   = RWORK(NWT   + K - 1)
         SDOT = RWORK(NSDOT + K - 1)
         ZP(K) = ( - Z(K) * ZP(KKTOT+1) + WDOT * WT
     1             + AVRAT * SDOT * WT ) / Z(KKTOT+1)
 300  CONTINUE
C
C        Form the surface mass equations
C
      DO 400 N = NFSURF, NLSURF
         SITDOT = RWORK(NSITDT + N - 1)
         SDEN0  = RWORK(NSDEN + N - 1)
         KFIRST = IWORK(NIPKF + N - 1)
         KLAST  = IWORK(NIPKL + N - 1)
         DO 400 K = KFIRST, KLAST
             SDOT = RWORK(NSDOT + K - 1)
             RCOV = RWORK(NRCOV + K - 1)
             ZP(K) = (SDOT*RCOV - Z(K) * SITDOT) / SDEN0
  400 CONTINUE
C
C        Form the bulk mass equations
C
      KFIRST = IWORK(NIPKF + NFBULK - 1)
      KLAST  = IWORK(NIPKL + NLBULK - 1)
      DO 500 K = KFIRST, KLAST
         ZP(K) = RWORK(NSDOT + K - 1)*RWORK(NWT + K - 1) * AVRAT
  500 CONTINUE
C
C        Form the surface site number-density equations
```

```
C
      DO 575 N = NFSURF, NLSURF
         ZP(KKTOT+1+N-NFSURF+1) = RWORK(NSITDT + N - 1)
  575 CONTINUE
C
      RETURN
      END
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE PRT1 (KK, KSYM, LOUT, X)
C
C*****precision > double
      IMPLICIT DOUBLE PRECISION (A-H, O-Z), INTEGER (I-N)
C*****END precision > double
C*****precision > single
C      IMPLICIT REAL (A-H,O-Z), INTEGER (I-N)
C*****END precision > single
C
      DIMENSION X(KK)
      CHARACTER*(*) KSYM(KK)
C
      DO 10 K = 1, KK, 3
         WRITE (LOUT, 6010) (KSYM(L), X(L), L=K, MIN(K+2, KK))
   10 CONTINUE
 6010 FORMAT (3X, 3(A12,'=', 1PE10.3, 4X))
C
      RETURN
      END
```

8. Output from Fortran Code

```
CKLIB: CHEMKIN-III GAS-PHASE CHEMICAL KINETICS LIBRARY,
DOUBLE PRECISION Vers. 5.0 March 1, 1996
Copyright 1995, Sandia Corporation.
The U.S. Government retains a limited license in this software.


SKLIB: CHEMKIN-III SURFACE KINETICS LIBRARY,
DOUBLE PRECISION Vers. 6.0 March 1, 1996
Copyright 1995, Sandia Corporation.
The U.S. Government retains a limited license in this software.


INPUT INITIAL PRESSURE(ATM) AND TEMPERATURE(K)
  .263E-02   .171E+04

INPUT INITIAL ACTIVITY OF NEXT SPECIES
SIF4            0.14286

INPUT INITIAL ACTIVITY OF NEXT SPECIES
NH3             0.85714

INPUT INITIAL ACTIVITY OF NEXT SPECIES
NHSIF(S)        6.251E-2

INPUT INITIAL ACTIVITY OF NEXT SPECIES
NHNH2(S)        0.91587

INPUT INITIAL ACTIVITY OF NEXT SPECIES
SIF3NH2(S)      2.354E-4

INPUT INITIAL ACTIVITY OF NEXT SPECIES
SIF2NH(S)       2.0837E-2

INPUT INITIAL ACTIVITY OF NEXT SPECIES
NH2SIFNH(S)     1.806E-4

INPUT INITIAL ACTIVITY OF NEXT SPECIES
NHSIFNHSIFNH(S)  3.6127E-4

INPUT INITIAL ACTIVITY OF NEXT SPECIES
N(D)            1.0

INPUT INITIAL ACTIVITY OF NEXT SPECIES
SI(D)           1.0

INPUT INITIAL ACTIVITY OF NEXT SPECIES
END

INPUT SURFACE AREA TO VOLUME RATIO
  .600E+01

INPUT FINAL TIME AND DT
  .500E-01   .500E-02

 TIME =  .0
 GAS-PHASE STATE
 P =   2.6648E+03 T =   1.7130E+03 DENSITY =   5.5132E-07
 GAS-PHASE MOLE FRACTIONS
```

```
 H2          =  .000E+00    H           =  .000E+00    N2          =  .000E+00
 N           =  .000E+00    NH          =  .000E+00    NH2         =  .000E+00
 NNH         =  .000E+00    N2H2        =  .000E+00    N2H3        =  .000E+00
 N2H4        =  .000E+00    HF          =  .000E+00    F           =  .000E+00
 SIF4        = 1.429E-01    SIF3        =  .000E+00    SIHF3       =  .000E+00
 SIF3NH2     =  .000E+00    NH3         = 8.571E-01
SURFACE SITE FRACTIONS ON PHASE (SITE)  2
 NHSIF(S)    = 6.251E-02    SIF3NH2(S)  = 2.354E-04    SIF2NH(S)   = 2.084E-02
 NH2SIFNH(S) = 1.806E-04    NHSIFNHSIFNH= 3.613E-04    NHNH2(S)    = 9.159E-01
 SUM OF SURFACE SITE FRACTIONS 1.0
 SURFACE SITE DENSITY   4.168300000000000E-09
BULK DEPOSITION (GM/CM**2) IN PHASE  3
 SI(D)       =  .000E+00
BULK DEPOSITION (GM/CM**2) IN PHASE  4
 N(D)        =  .000E+00

TIME =  5.000000000000000E-03
GAS-PHASE STATE
P =   3.0065E+03 T =   1.7130E+03 DENSITY =   4.8817E-07
GAS-PHASE MOLE FRACTIONS
 H2          = 2.704E-06    H           = 2.254E-08    N2          = 1.863E-10
 N           = 2.284E-14    NH          = 3.420E-10    NH2         = 5.428E-06
 NNH         = 1.422E-11    N2H2        = 3.396E-10    N2H3        = 1.230E-11
 N2H4        = 5.744E-13    HF          = 2.829E-01    F           = 9.958E-11
 SIF4        = 6.035E-02    SIF3        = 8.522E-11    SIHF3       = 2.630E-11
 SIF3NH2     = 1.277E-10    NH3         = 6.568E-01
SURFACE SITE FRACTIONS ON PHASE (SITE)  2
 NHSIF(S)    = 4.850E-02    SIF3NH2(S)  = 1.534E-04    SIF2NH(S)   = 1.291E-02
 NH2SIFNH(S) = 2.084E-04    NHSIFNHSIFNH= 4.171E-04    NHNH2(S)    = 9.378E-01
 SUM OF SURFACE SITE FRACTIONS 1.0
 SURFACE SITE DENSITY   4.168300000000000E-09
BULK DEPOSITION (GM/CM**2) IN PHASE  3
 SI(D)       = 4.700E-08
BULK DEPOSITION (GM/CM**2) IN PHASE  4
 N(D)        = 2.659E-08

TIME =  1.000000000000000E-02
GAS-PHASE STATE
P =   3.1752E+03 T =   1.7130E+03 DENSITY =   4.5844E-07
GAS-PHASE MOLE FRACTIONS
 H2          = 5.055E-06    H           = 2.264E-08    N2          = 2.256E-09
 N           = 2.490E-13    NH          = 1.806E-09    NH2         = 1.011E-05
 NNH         = 7.943E-11    N2H2        = 1.742E-09    N2H3        = 5.209E-11
 N2H4        = 2.231E-12    HF          = 4.074E-01    F           = 3.097E-10
 SIF4        = 2.667E-02    SIF3        = 5.378E-11    SIHF3       = 4.881E-11
 SIF3NH2     = 2.443E-10    NH3         = 5.659E-01
SURFACE SITE FRACTIONS ON PHASE (SITE)  2
 NHSIF(S)    = 2.927E-02    SIF3NH2(S)  = 7.350E-05    SIF2NH(S)   = 6.970E-03
 NH2SIFNH(S) = 1.896E-04    NHSIFNHSIFNH= 3.797E-04    NHNH2(S)    = 9.631E-01
 SUM OF SURFACE SITE FRACTIONS 1.0
 SURFACE SITE DENSITY   4.168300000000000E-09
BULK DEPOSITION (GM/CM**2) IN PHASE  3
 SI(D)       = 7.497E-08
BULK DEPOSITION (GM/CM**2) IN PHASE  4
 N(D)        = 3.963E-08

TIME =  1.500000000000000E-02
GAS-PHASE STATE
P =   3.2558E+03 T =   1.7130E+03 DENSITY =   4.4466E-07
GAS-PHASE MOLE FRACTIONS
 H2          = 7.295E-06    H           = 2.281E-08    N2          = 8.864E-09
 N           = 9.816E-13    NH          = 4.684E-09    NH2         = 1.453E-05
 NNH         = 1.978E-10    N2H2        = 4.188E-09    N2H3        = 1.147E-10
 N2H4        = 4.816E-12    HF          = 4.647E-01    F           = 5.495E-10
```

```
 SIF4        = 1.193E-02    SIF3        = 2.873E-11    SIHF3       = 6.054E-11
 SIF3NH2     = 3.091E-10    NH3         = 5.233E-01
SURFACE SITE FRACTIONS ON PHASE (SITE)  2
 NHSIF(S)    = 1.601E-02    SIF3NH2(S)  = 3.431E-05    SIF2NH(S)   = 3.499E-03
 NH2SIFNH(S) = 1.798E-04    NHSIFNHSIFNH= 3.600E-04    NHNH2(S)    = 9.799E-01
 SUM OF SURFACE SITE FRACTIONS 1.0
 SURFACE SITE DENSITY   4.168300000000000E-09
BULK DEPOSITION (GM/CM**2) IN PHASE  3
 SI(D)       = 8.990E-08
BULK DEPOSITION (GM/CM**2) IN PHASE  4
 N(D)        = 4.585E-08

TIME =  2.000000000000000E-02
GAS-PHASE STATE
P =   3.2934E+03 T =   1.7130E+03 DENSITY =   4.3840E-07
GAS-PHASE MOLE FRACTIONS
 H2          = 9.524E-06    H           = 2.306E-08    N2          = 2.265E-08
 N           = 2.505E-12    NH          = 9.016E-09    NH2         = 1.888E-05
 NNH         = 3.679E-10    N2H2        = 7.662E-09    N2H3        = 1.994E-10
 N2H4        = 8.294E-12    HF          = 4.914E-01    F           = 7.854E-10
 SIF4        = 5.356E-03    SIF3        = 1.435E-11    SIHF3       = 6.631E-11
 SIF3NH2     = 3.429E-10    NH3         = 5.032E-01
SURFACE SITE FRACTIONS ON PHASE (SITE)  2
 NHSIF(S)    = 8.165E-03    SIF3NH2(S)  = 1.573E-05    SIF2NH(S)   = 1.670E-03
 NH2SIFNH(S) = 1.749E-04    NHSIFNHSIFNH= 3.502E-04    NHNH2(S)    = 9.896E-01
 SUM OF SURFACE SITE FRACTIONS 1.0
 SURFACE SITE DENSITY   4.168300000000000E-09
BULK DEPOSITION (GM/CM**2) IN PHASE  3
 SI(D)       = 9.750E-08
BULK DEPOSITION (GM/CM**2) IN PHASE  4
 N(D)        = 4.874E-08

TIME =  2.500000000000000E-02
GAS-PHASE STATE
P =   3.3106E+03 T =   1.7130E+03 DENSITY =   4.3558E-07
GAS-PHASE MOLE FRACTIONS
 H2          = 1.178E-05    H           = 2.339E-08    N2          = 4.629E-08
 N           = 5.001E-12    NH          = 1.472E-08    NH2         = 2.322E-05
 NNH         = 5.909E-10    N2H2        = 1.220E-08    N2H3        = 3.066E-10
 N2H4        = 1.267E-11    HF          = 5.038E-01    F           = 1.009E-09
 SIF4        = 2.405E-03    SIF3        = 6.877E-12    SIHF3       = 6.908E-11
 SIF3NH2     = 3.599E-10    NH3         = 4.937E-01
SURFACE SITE FRACTIONS ON PHASE (SITE)  2
 NHSIF(S)    = 3.963E-03    SIF3NH2(S)  = 7.140E-06    SIF2NH(S)   = 7.736E-04
 NH2SIFNH(S) = 1.725E-04    NHSIFNHSIFNH= 3.455E-04    NHNH2(S)    = 9.947E-01
 SUM OF SURFACE SITE FRACTIONS 1.0
 SURFACE SITE DENSITY   4.168300000000000E-09
BULK DEPOSITION (GM/CM**2) IN PHASE  3
 SI(D)       = 1.012E-07
BULK DEPOSITION (GM/CM**2) IN PHASE  4
 N(D)        = 5.006E-08

TIME =  3.000000000000000E-02
GAS-PHASE STATE
P =   3.3184E+03 T =   1.7130E+03 DENSITY =   4.3431E-07
GAS-PHASE MOLE FRACTIONS
 H2          = 1.408E-05    H           = 2.380E-08    N2          = 8.252E-08
 N           = 8.522E-12    NH          = 2.167E-08    NH2         = 2.756E-05
 NNH         = 8.688E-10    N2H2        = 1.785E-08    N2H3        = 4.368E-10
 N2H4        = 1.794E-11    HF          = 5.095E-01    F           = 1.222E-09
 SIF4        = 1.080E-03    SIF3        = 3.211E-12    SIHF3       = 7.037E-11
 SIF3NH2     = 3.683E-10    NH3         = 4.894E-01
SURFACE SITE FRACTIONS ON PHASE (SITE)  2
 NHSIF(S)    = 1.861E-03    SIF3NH2(S)  = 3.222E-06    SIF2NH(S)   = 3.526E-04
```

```
 NH2SIFNH(S) = 1.714E-04    NHSIFNHSIFNH= 3.433E-04    NHNH2(S)    = 9.973E-01
 SUM OF SURFACE SITE FRACTIONS 1.0
 SURFACE SITE DENSITY   4.168300000000000E-09
BULK DEPOSITION (GM/CM**2) IN PHASE  3
 SI(D)       = 1.030E-07
BULK DEPOSITION (GM/CM**2) IN PHASE  4
 N(D)        = 5.066E-08

TIME =  3.500000000000000E-02
GAS-PHASE STATE
P =   3.3220E+03 T =   1.7130E+03 DENSITY =   4.3374E-07
GAS-PHASE MOLE FRACTIONS
 H2          = 1.643E-05    H           = 2.429E-08    N2          = 1.342E-07
 N           = 1.300E-11    NH          = 2.974E-08    NH2         = 3.190E-05
 NNH         = 1.203E-09    N2H2        = 2.465E-08    N2H3        = 5.901E-10
 N2H4        = 2.412E-11    HF          = 5.121E-01    F           = 1.428E-09
 SIF4        = 4.849E-04    SIF3        = 1.476E-12    SIHF3       = 7.096E-11
 SIF3NH2     = 3.723E-10    NH3         = 4.874E-01
SURFACE SITE FRACTIONS ON PHASE (SITE)  2
 NHSIF(S)    = 8.564E-04    SIF3NH2(S)  = 1.450E-06    SIF2NH(S)   = 1.594E-04
 NH2SIFNH(S) = 1.709E-04    NHSIFNHSIFNH= 3.422E-04    NHNH2(S)    = 9.985E-01
 SUM OF SURFACE SITE FRACTIONS 1.0
 SURFACE SITE DENSITY   4.168300000000000E-09
BULK DEPOSITION (GM/CM**2) IN PHASE  3
 SI(D)       = 1.038E-07
BULK DEPOSITION (GM/CM**2) IN PHASE  4
 N(D)        = 5.093E-08

TIME =  4.000000000000000E-02
GAS-PHASE STATE
P =   3.3236E+03 T =   1.7130E+03 DENSITY =   4.3349E-07
GAS-PHASE MOLE FRACTIONS
 H2          = 1.884E-05    H           = 2.486E-08    N2          = 2.041E-07
 N           = 1.831E-11    NH          = 3.881E-08    NH2         = 3.625E-05
 NNH         = 1.596E-09    N2H2        = 3.262E-08    N2H3        = 7.664E-10
 N2H4        = 3.121E-11    HF          = 5.132E-01    F           = 1.629E-09
 SIF4        = 2.177E-04    SIF3        = 6.732E-13    SIHF3       = 7.123E-11
 SIF3NH2     = 3.743E-10    NH3         = 4.865E-01
SURFACE SITE FRACTIONS ON PHASE (SITE)  2
 NHSIF(S)    = 3.895E-04    SIF3NH2(S)  = 6.515E-07    SIF2NH(S)   = 7.180E-05
 NH2SIFNH(S) = 1.707E-04    NHSIFNHSIFNH= 3.418E-04    NHNH2(S)    = 9.990E-01
 SUM OF SURFACE SITE FRACTIONS 1.0
 SURFACE SITE DENSITY   4.168300000000000E-09
BULK DEPOSITION (GM/CM**2) IN PHASE  3
 SI(D)       = 1.041E-07
BULK DEPOSITION (GM/CM**2) IN PHASE  4
 N(D)        = 5.105E-08

TIME =  4.500000000000000E-02
GAS-PHASE STATE
P =   3.3243E+03 T =   1.7130E+03 DENSITY =   4.3337E-07
GAS-PHASE MOLE FRACTIONS
 H2          = 2.131E-05    H           = 2.553E-08    N2          = 2.953E-07
 N           = 2.427E-11    NH          = 4.878E-08    NH2         = 4.060E-05
 NNH         = 2.047E-09    N2H2        = 4.176E-08    N2H3        = 9.658E-10
 N2H4        = 3.920E-11    HF          = 5.138E-01    F           = 1.828E-09
 SIF4        = 9.773E-05    SIF3        = 3.069E-13    SIHF3       = 7.135E-11
 SIF3NH2     = 3.752E-10    NH3         = 4.861E-01
SURFACE SITE FRACTIONS ON PHASE (SITE)  2
 NHSIF(S)    = 1.760E-04    SIF3NH2(S)  = 2.926E-07    SIF2NH(S)   = 3.228E-05
 NH2SIFNH(S) = 1.706E-04    NHSIFNHSIFNH= 3.416E-04    NHNH2(S)    = 9.993E-01
 SUM OF SURFACE SITE FRACTIONS 1.0
 SURFACE SITE DENSITY   4.168300000000000E-09
BULK DEPOSITION (GM/CM**2) IN PHASE  3
```

```
 SI(D)       = 1.043E-07
BULK DEPOSITION (GM/CM**2) IN PHASE  4
 N(D)        = 5.111E-08

TIME =  5.000000000000000E-02
GAS-PHASE STATE
P =   3.3246E+03 T =   1.7130E+03 DENSITY =   4.3332E-07
GAS-PHASE MOLE FRACTIONS
 H2          = 2.386E-05    H           = 2.628E-08    N2          = 4.107E-07
 N           = 3.071E-11    NH          = 5.957E-08    NH2         = 4.494E-05
 NNH         = 2.558E-09    N2H2        = 5.211E-08    N2H3        = 1.188E-09
 N2H4        = 4.810E-11    HF          = 5.140E-01    F           = 2.025E-09
 SIF4        = 4.387E-05    SIF3        = 1.412E-13    SIHF3       = 7.140E-11
 SIF3NH2     = 3.756E-10    NH3         = 4.859E-01
SURFACE SITE FRACTIONS ON PHASE (SITE)  2
 NHSIF(S)    = 7.928E-05    SIF3NH2(S)  = 1.314E-07    SIF2NH(S)   = 1.450E-05
 NH2SIFNH(S) = 1.705E-04    NHSIFNHSIFNH= 3.415E-04    NHNH2(S)    = 9.994E-01
 SUM OF SURFACE SITE FRACTIONS 1.0
 SURFACE SITE DENSITY   4.168300000000000E-09
BULK DEPOSITION (GM/CM**2) IN PHASE  3
 SI(D)       = 1.044E-07
BULK DEPOSITION (GM/CM**2) IN PHASE  4
 N(D)        = 5.113E-08

TIME =  5.000000000000000E-02
GAS-PHASE STATE
P =   3.3246E+03 T =   1.7130E+03 DENSITY =   4.3332E-07
GAS-PHASE MOLE FRACTIONS
 H2          = 2.386E-05    H           = 2.628E-08    N2          = 4.107E-07
 N           = 3.071E-11    NH          = 5.957E-08    NH2         = 4.494E-05
 NNH         = 2.558E-09    N2H2        = 5.211E-08    N2H3        = 1.188E-09
 N2H4        = 4.810E-11    HF          = 5.140E-01    F           = 2.025E-09
 SIF4        = 4.387E-05    SIF3        = 1.412E-13    SIHF3       = 7.140E-11
 SIF3NH2     = 3.756E-10    NH3         = 4.859E-01
SURFACE SITE FRACTIONS ON PHASE (SITE)  2
 NHSIF(S)    = 7.928E-05    SIF3NH2(S)  = 1.314E-07    SIF2NH(S)   = 1.450E-05
 NH2SIFNH(S) = 1.705E-04    NHSIFNHSIFNH= 3.415E-04    NHNH2(S)    = 9.994E-01
 SUM OF SURFACE SITE FRACTIONS 1.0
 SURFACE SITE DENSITY   4.168300000000000E-09
BULK DEPOSITION (GM/CM**2) IN PHASE  3
 SI(D)       = 1.044E-07
BULK DEPOSITION (GM/CM**2) IN PHASE  4
 N(D)        = 5.113E-08
STOP...TIME LIMIT REACHED...
```

9. VODE Summary

```
      SUBROUTINE DVODE (F, NEQ, Y, T, TOUT, ITOL, RTOL, ATOL, ITASK,
     1            ISTATE, IOPT, RWORK, LRW, IWORK, LIW, JAC, MF,
     2            RPAR, IPAR)
      EXTERNAL F, JAC
      DOUBLE PRECISION Y, T, TOUT, RTOL, ATOL, RWORK, RPAR
      INTEGER NEQ, ITOL, ITASK, ISTATE, IOPT, LRW, IWORK, LIW,
     1        MF, IPAR
      DIMENSION Y(*), RTOL(*), ATOL(*), RWORK(LRW), IWORK(LIW),
     1          RPAR(*), IPAR(*)
C-----------------------------------------------------------------------
C DVODE.. Variable-coefficient Ordinary Differential Equation solver,
C with fixed-leading coefficient implementation.
C This version is in double precision.
C
C DVODE solves the initial value problem for stiff or nonstiff
C systems of first order ODEs,
C     dy/dt = f(t,y) ,  or, in component form,
C     dy(i)/dt = f(i) = f(i,t,y(1),y(2),...,y(NEQ)) (i = 1,...,NEQ).
C DVODE is a package based on the EPISODE and EPISODEB packages, and
C on the ODEPACK user interface standard, with minor modifications.
C-----------------------------------------------------------------------
C Revision History (YYMMDD)
C   890615  Date Written
C   890922  Added interrupt/restart ability, minor changes throughout.
C   910228  Minor revisions in line format,  prologue, etc.
C   920227  Modifications by D. Pang:
C           (1) Applied subgennam to get generic intrinsic names.
C           (2) Changed intrinsic names to generic in comments.
C           (3) Added *DECK lines before each routine.
C   920721  Names of routines and labeled Common blocks changed, so as
C           to be unique in combined single/double precision code (ACH).
C   920722  Minor revisions to prologue (ACH).
C   920831  Conversion to double precision done (ACH).
C-----------------------------------------------------------------------
C References..
C
C 1. P. N. Brown, G. D. Byrne, and A. C. Hindmarsh, "VODE: A Variable
C    Coefficient ODE Solver," SIAM J. Sci. Stat. Comput., 10 (1989),
C    pp. 1038-1051.  Also, LLNL Report UCRL-98412, June 1988.
C 2. G. D. Byrne and A. C. Hindmarsh, "A Polyalgorithm for the
C    Numerical Solution of Ordinary Differential Equations,"
C    ACM Trans. Math. Software, 1 (1975), pp. 71-96.
C 3. A. C. Hindmarsh and G. D. Byrne, "EPISODE: An Effective Package
C    for the Integration of Systems of Ordinary Differential
C    Equations," LLNL Report UCID-30112, Rev. 1, April 1977.
C 4. G. D. Byrne and A. C. Hindmarsh, "EPISODEB: An Experimental
C    Package for the Integration of Systems of Ordinary Differential
C    Equations with Banded Jacobians," LLNL Report UCID-30132, April
C    1976.
C 5. A. C. Hindmarsh, "ODEPACK, a Systematized Collection of ODE
C    Solvers," in Scientific Computing, R. S. Stepleman et al., eds.,
C    North-Holland, Amsterdam, 1983, pp. 55-64.
C 6. K. R. Jackson and R. Sacks-Davis, "An Alternative Implementation
C    of Variable Step-Size Multistep Formulas for Stiff ODEs," ACM
C    Trans. Math. Software, 6 (1980), pp. 295-318.
C-----------------------------------------------------------------------
C Authors..
C
C               Peter N. Brown and Alan C. Hindmarsh
C               Computing and Mathematics Research Division, L-316
C               Lawrence Livermore National Laboratory
```

```
C               Livermore, CA 94550
C and
C               George D. Byrne
C               Exxon Research and Engineering Co.
C               Clinton Township
C               Route 22 East
C               Annandale, NJ 08801
C-----------------------------------------------------------------------
C Summary of usage.
C
C Communication between the user and the DVODE package, for normal
C situations, is summarized here.  This summary describes only a subset
C of the full set of options available.  See the full description for
C details, including optional communication, nonstandard options,
C and instructions for special situations.  See also the example
C problem (with program and output) following this summary.
C
C A. First provide a subroutine of the form..
C
C           SUBROUTINE F (NEQ, T, Y, YDOT, RPAR, IPAR)
C           DOUBLE PRECISION T, Y, YDOT, RPAR
C           DIMENSION Y(NEQ), YDOT(NEQ)
C
C which supplies the vector function f by loading YDOT(i) with f(i).
C
C B. Next determine (or guess) whether or not the problem is stiff.
C Stiffness occurs when the Jacobian matrix df/dy has an eigenvalue
C whose real part is negative and large in magnitude, compared to the
C reciprocal of the t span of interest.  If the problem is nonstiff,
C use a method flag MF = 10.  If it is stiff, there are four standard
C choices for MF (21, 22, 24, 25), and DVODE requires the Jacobian
C matrix in some form.  In these cases (MF .gt. 0), DVODE will use a
C saved copy of the Jacobian matrix.  If this is undesirable because of
C storage limitations, set MF to the corresponding negative value
C (-21, -22, -24, -25).  (See full description of MF below.)
C The Jacobian matrix is regarded either as full (MF = 21 or 22),
C or banded (MF = 24 or 25).  In the banded case, DVODE requires two
C half-bandwidth parameters ML and MU.  These are, respectively, the
C widths of the lower and upper parts of the band, excluding the main
C diagonal.  Thus the band consists of the locations (i,j) with
C i-ML .le. j .le. i+MU, and the full bandwidth is ML+MU+1.
C
C C. If the problem is stiff, you are encouraged to supply the Jacobian
C directly (MF = 21 or 24), but if this is not feasible, DVODE will
C compute it internally by difference quotients (MF = 22 or 25).
C If you are supplying the Jacobian, provide a subroutine of the form..
C
C           SUBROUTINE JAC (NEQ, T, Y, ML, MU, PD, NROWPD, RPAR, IPAR)
C           DOUBLE PRECISION T, Y, PD, RPAR
C           DIMENSION Y(NEQ), PD(NROWPD,NEQ)
C
C which supplies df/dy by loading PD as follows..
C     For a full Jacobian (MF = 21), load PD(i,j) with df(i)/dy(j),
C the partial derivative of f(i) with respect to y(j).  (Ignore the
C ML and MU arguments in this case.)
C     For a banded Jacobian (MF = 24), load PD(i-j+MU+1,j) with
C df(i)/dy(j), i.e. load the diagonal lines of df/dy into the rows of
C PD from the top down.
C     In either case, only nonzero elements need be loaded.
C
C D. Write a main program which calls subroutine DVODE once for
C each point at which answers are desired.  This should also provide
C for possible use of logical unit 6 for output of error messages
C by DVODE.  On the first call to DVODE, supply arguments as follows..
```

```
C F      = Name of subroutine for right-hand side vector f.
C          This name must be declared external in calling program.
C NEQ    = Number of first order ODE-s.
C Y      = Array of initial values, of length NEQ.
C T      = The initial value of the independent variable.
C TOUT   = First point where output is desired (.ne. T).
C ITOL   = 1 or 2 according as ATOL (below) is a scalar or array.
C RTOL   = Relative tolerance parameter (scalar).
C ATOL   = Absolute tolerance parameter (scalar or array).
C          The estimated local error in Y(i) will be controlled so as
C          to be roughly less (in magnitude) than
C             EWT(i) = RTOL*abs(Y(i)) + ATOL     if ITOL = 1, or
C             EWT(i) = RTOL*abs(Y(i)) + ATOL(i)  if ITOL = 2.
C          Thus the local error test passes if, in each component,
C          either the absolute error is less than ATOL (or ATOL(i)),
C          or the relative error is less than RTOL.
C          Use RTOL = 0.0 for pure absolute error control, and
C          use ATOL = 0.0 (or ATOL(i) = 0.0) for pure relative error
C          control.  Caution.. Actual (global) errors may exceed these
C          local tolerances, so choose them conservatively.
C ITASK  = 1 for normal computation of output values of Y at t = TOUT.
C ISTATE = Integer flag (input and output).  Set ISTATE = 1.
C IOPT   = 0 to indicate no optional input used.
C RWORK  = Real work array of length at least..
C             20 + 16*NEQ                    for MF = 10,
C             22 +  9*NEQ + 2*NEQ**2         for MF = 21 or 22,
C             22 + 11*NEQ + (3*ML + 2*MU)*NEQ  for MF = 24 or 25.
C LRW    = Declared length of RWORK (in user's DIMENSION statement).
C IWORK  = Integer work array of length at least..
C             30        for MF = 10,
C             30 + NEQ  for MF = 21, 22, 24, or 25.
C          If MF = 24 or 25, input in IWORK(1),IWORK(2) the lower
C          and upper half-bandwidths ML,MU.
C LIW    = Declared length of IWORK (in user's DIMENSION).
C JAC    = Name of subroutine for Jacobian matrix (MF = 21 or 24).
C          If used, this name must be declared external in calling
C          program.  If not used, pass a dummy name.
C MF     = Method flag.  Standard values are..
C          10 for nonstiff (Adams) method, no Jacobian used.
C          21 for stiff (BDF) method, user-supplied full Jacobian.
C          22 for stiff method, internally generated full Jacobian.
C          24 for stiff method, user-supplied banded Jacobian.
C          25 for stiff method, internally generated banded Jacobian.
C RPAR,IPAR = user-defined real and integer arrays passed to F and JAC.
C Note that the main program must declare arrays Y, RWORK, IWORK,
C and possibly ATOL, RPAR, and IPAR.
C
C E. The output from the first call (or any call) is..
C      Y = Array of computed values of y(t) vector.
C      T = Corresponding value of independent variable (normally TOUT).
C ISTATE = 2  if DVODE was successful, negative otherwise.
C          -1 means excess work done on this call. (Perhaps wrong MF.)
C          -2 means excess accuracy requested. (Tolerances too small.)
C          -3 means illegal input detected. (See printed message.)
C          -4 means repeated error test failures. (Check all input.)
C          -5 means repeated convergence failures. (Perhaps bad
C             Jacobian supplied or wrong choice of MF or tolerances.)
C          -6 means error weight became zero during problem. (Solution
C             component i vanished, and ATOL or ATOL(i) = 0.)
C
C F. To continue the integration after a successful return, simply
C reset TOUT and call DVODE again.  No other parameters need be reset.
C
```

REFERENCES

1. Kee, R. J., Rupley, F. M, and Miller, J. A., "CHEMKIN-III: A Fortran Chemical Kinetics Package for the Analysis of Multi-Fluid Gas-Phase Chemical Kinetics," Sandia National Laboratories Report, SAND96-8216 (1996).
2. Kee, R. J., Warnetz, J., and Miller, J. A., "A Fortran Computer Code Package for the Evaluation of Gas-Phase Viscosities, Conductivities, and Diffusion Coefficients," Sandia National Laboratories Report, SAND83-8209 (1983).
3. Kee, R. J., Dixon Lewis, G., Warnatz, J., Coltrin, M.E., and Miller, J. A., "A Fortran Computer Package for the Evaluation of Gas-Phase, Multicomponent Transport Properties," Sandia National Laboratories Report, SAND86-8246 (1986).
4. Kee, R. J. and Miller, J. A., "A Structured Approach to the Computational Modeling of Chemical Kinetics and Molecular Transport in Flowing Systems," *Springier Series in Chemical Physics* **47**, 196 (1986). (Also available as Sandia National Laboratories Report-8003.)
5. Kee, R. J., Miller, J. A., and Jefferson, T. J., "Chemkin: A General-Purpose Problem-Independent, Transportable, Fortran Chemical Kinetics Code Package," Sandia National Laboratories Report, SAND80-8003 (1980).
6. Kee, R. J., Rupley, F. M., and Miller, J. A., "Chemkin-II: A Fortran Chemical Kinetics Package for the Analysis of Gas-Phase Chemical Kinetics," Sandia National Laboratories Report, SAND89-8009 (1990).
7. Kee, R. J., Rupley, F. M., and Miller, J. A., "The Chemkin Thermodynamic Database," Sandia National Laboratories Report, SAND87-8215 (1987).
8. Gordon, S. and McBride, B. J., "Computer Program for Calculation of Complex Chemical Equilibrium Compositions, Rocket Performance, Incident and Reflected Shocks and Chapman-Jouguet Detonations," NASA SP-273 (1971).
9. Ericksson, G., *Acta Chem. Scand.* **25,** 2651 (1971).
10. Motz, H and Wise, H., *J. Chem. Phys.*, **31**, 1893 (1960).
11. Goodwin, D. G. and Gavillet, G. G., *J. Appl. Phys.* **68**, 6393 (1990).
12. Brown, P. N., Byrne, G. D., and Hindmarsh, A. C., "VODE, A Variable Coefficient ODE Solver," *SIAM J. Sci. Stat. Comput.*, **10,** 1038 (1989).
13. Grcar, J. F., "The Change Tool for Changing Programs and Scripts," Sandia National Laboratories Report, SAND92-8225 (1992).

APPENDIX A. STORAGE ALLOCATION FOR THE WORK ARRAYS

Work arrays ISKWRK, RSKWRK, and CSKWRK contain information about the elements, species and reaction in the mechanism; they also contain some work space needed for internal manipulations. A user wishing to modify a subroutine or to write new routines will probably want to use the work arrays directly. The pointers described below are starting addresses for information stored in the work arrays, and are found in the labeled common block COMMON /SKSTRT/, declared by the use of the include file skstrt.h.

```
   It should be noted that storage in work array ISKWRK serves
   several purposes, for instance,
   1. ISKWRK(N) may be a constant in regards to a surface described
      by the mechanism, and is subject to change if there are multiple
      surfaces,
   2. ISKWRK(N) may be a pointer, where L=ISKWRK(N) is used to locate
      locate information specific to a surface, either in
      ISKWRK, RSKWRK, or CSKWRK, then
   3. ISKWRK(L=ISKWRK(N)) may the first value of an integer array
      specific to a surface, RSKWRK(L=ISKWRK(N)) the first of a real
      array, and CSKWRK(L=ISKWRK(N)) the first of a character string array
   COMMON /SKSTRT/
   Integer constants
  1   MAXSPR, NELEM, NKKGAS, NSPAR, NSCOV, NEDPAR, NYPAR, MAXORD,
  2   MAXTP, NCP,    NCP1,  NCP2,   NCP2T,
   ISKWRK pointers to integer variables
  3   IiLENI, IiLENR, IiLENC, IiKSUR, IiKBLK, IiKTOT, IiNPHA, IiFSUR,
  4   IiLSUR, IiNSUR, IiFBLK, IiLBLK, IiNBLK, IiNIIS, IiNCOV, IiNREV,
  5   IiNSTK, IiNCON, IiNBHM, IiNRNU, IiNORD, IiNEDP, IiELEC, IiNYLD,
   ISKWRK pointers to integer arrays
  6   IiPKST, IiPKND, IiPTOT, IiKPHS, IiKCHG, IiKCMP, IiNSCV, IiKNT,
  7   IiNRPP, IiNREA, IiNUNK, IiNU,   IiNSUM, IiICOV, IiKCOV, IiIREV,
  8   IiISTK, IiMSTK, IiIBHM, IiKBHM, IiIRNU, IiIORD, IiKORD, IiIONS,
  9   IiKION, IiKTFL, IiNEDP, IiIEDP, IiKEDP, IiIYLD, IiYION, IiKYLD,
   ISKWRK pointers to real variables
  *   IrSKMN, IrPATM, IrRU,   IrRUC,
   ISKWRK pointers to real arrays
  1   IrSDEN, IrKTMP, IrKTHM, IrKDEN, IrAWT,  IrKWT,  IrPAR, IrKCOV,
  2   IrRPAR, IrEQ,   IrRNU,  IrNCF,  IrKORD, IrKFT,  IrKRT, IrKT1,
  3   IrKT2,  IrPT1,  IrIT1,  IrIT2,  IrIT3,  IrPEDP, IrENGI,
  4   IrPYLD, IrYNCF,
   ISKWRK pointers to character string arrays
  4   IcENAM, IcKNAM, IcMNAM, IcPNAM
   INTEGER CONSTANTS:
   MAXSPR, Maximum number of species in any surface reaction.
           Unless changed in the interpreter MAXSPR=12.
   NELEM,  Total count, elements in problem.
   NKKGAS, Total count, gas-phase species in problem.
   NSPAR,  Number of parameters  required in the rate expression
           for reactions;  in the current formulation NSPAR=3,
           however, a 4th parameter can be used for purposes of scaling.
   NSCOV,  Number of parameters required in the rate expression
           for a coverage reaction;  NSCOV=3.
   NEDPAR, Number of parameters required in the rate expression
           for ion-energy dependence reactions;  NEDPAR=3.
   NYPAR,  Number of parameters required in the yield-modified
           reactions; NYPAR=4.
   MAXORD, Maximum number of change-orders allowed in a reaction.
   MAXTP,  Maximum number of temperatures allowed in fits of
           thermodynamic properties for any species; MAXTP=3.
```

```
NCP,    Number of polynomial coefficients to fits of CP/R for
        a species; NCP=5.
NCP1,   NCP+1.
NCP2,   NCP+2.
NCP2T,  Total number of thermodynamic fit coefficients for
        species; NCP2T = (MAXTP-1)*NCP2 = 14.
ISKWRK POINTERS TO INTEGER VARIABLES:
IiLENI, ISKWRK(IiLENI) is the total length of ISKWRK required.
IiLENR, ISKWRK(IiLENR) is the total length of RSKWRK required.
IiLENC, ISKWRK(IiLENC) is the total length of CSKWRK required.
IiKSUR, ISKWRK(IiKSUR) is the total surface species count.
IiKBLK, ISKWRK(IiKTOT) is the total species count (gas+surface+bulk).
IiNPHA, ISKWRK(IiNPHA) is the total phase count.
IiFSUR, ISKWRK(IiFSUR) is the phase index of the first site.
IiLSUR, ISKWRK(IiLSUR) is the phase index of the last site.
IiNSUR, ISKWRK(IiNSUR) is the total surface phage count.
IiFBLK, ISKWRK(IiFBLK) is the phase index of the first bulk.
IiLBLK, ISKWRK(IiLBLK) is the phase index of the last bulk.
IiNBLK, ISKWRK(IiNBLK) is the total bulk phase count.
IiNIIS, ISKWRK(IiNIIS) is the total surface reaction count.
IiNCOV, ISKWRK(IiNCOV) is the total coverage reaction count.
IiNREV, ISKWRK(IiNREV) is the total count of surface reactions which
        use explicit reverse parameters.
IiNSTK, ISKWRK(IiNSTK) is the total count of sticking surface reactions
IiNCON, ISKWRK(IiNCON) is the total count of surface reactions which
        do not conserve sites.
IiNBHM, ISKWRK(IiNBHM) is the total count of Bohm surface reactions.
IiNRNU, ISKWRK(IiNRNU) is the total count of surface reactions with
        real stoichiometry coefficients.
IiNORD, ISKWRK(IiNORD) is the total count of surface reactions with
        changed-order species.
IiNEDP, ISKWRK(IiNEDP) is the total count of surface reactions with
        ion-energy dependence.
IiELEC, ISKWRK(IiELEC) is the location in a species array of the
        electron species.
IiNYLD, ISKWRK(IiNYLD) is the total count of surface reactions with
        yield-modified species.
ISKWRK POINTERS TO THE START OF ISKWRK ARRAY WORKSPACE:
IiPKST, ISKWRK(I = ISKWRK(IiPKST)) starts an array of species indices
        for the first species of the phases;
        ISKWRK(I + N - 1) is the index of the first species of
        phase N.
IiPKND, ISKWRK(I = ISKWRK(IiPKND)) starts an array of species indices
        for the last species of the phases;
        ISKWRK(I + N - 1) is the indes of the final species of
        phase N.
IiPTOT, ISKWRK(I = ISKWRK(IiPTOT)) starts an array of total counts
        of species in the phases;
        ISKWRK(I + N - 1) is the total species count of phase N.
IiKPHS, ISKWRK(I = ISKWRK(IiKPHS)) starts an array of physical
        phases for the species;
        ISKWRK(I + K - 1)  = -1, species K is a solid,
                           =  0, speciies K is a gas,
                           = +1, species K is a liquid.
IiKCHG, ISKWRK(I = ISKWRK(IiKCHG)) starts an array of electronic
        charges for the species;
        ISKWRK(I + K - 1) is the charge of species K,
        for example, a value of -2 indicates two excess electrons.
IiKCMP, ISKWRK(I = ISKWRK(IiKCMP)) starts a matrix of elemental
        composition for the species;
        ISKWRK(I + (K-1)*NELEM + M - 1) is the quantity of
        element M in species K.
IiNSCV, ISKWRK(I = ISKWRK(IiNSCV)) starts an array of site coverage
        for the species;
```

```
        ISKWRK(I + K - 1) is the site coverage of species K.
IiKNT,  ISKWRK(I = ISKWRK(IiKNT)) starts an array of the total
        number of temperatures dividing the ranges of thermodynamic
        fits of the species.
        ISKWRK(I + K - 1) is the number of dividing temperatures
        for thermodynamic fits for species K;
IiNRPP, ISKWRK(I = ISKWRK(IiNRPP)) starts an array of the total
        of participant species for the surface reactions,
        and indidates the reversibility of the reactions;
        ISKWRK(I + IS - 1) = +N, reaction IS has N participant
                                 species (reactants + products),
                                 and reaction IS is reversible,
                           = -N, N participant species in an
                                 irreversible reaction.
IiNREA, ISKWRK(I = ISKWRK(IiNREA)) starts an array of the total
        count of reactants only for the surface reactions.
        ISKWRK(I + N - 1) is the reactant count for the Nth
        surface reaction.
IiNUNK, ISKWRK(I = ISKWRK(IiNUNK)) starts a matrix of indices
        for the species in the surface reactions;
        ISKWRK(I + (N-1)*MAXSPR + L - 1) is the species index for
        Lth species in the Nth surface reaction.
IiNU,   ISKWRK(I = ISKWRK(IiNU)) starts a matrix of stoichiometric
        coefficients for the species in the surface reactions;
        ISKWRK(I + (N-1)*MAXSPR + L - 1) is the stoichiometric
        of the Lth species in the Nth surface reaction.
IiNSUM, ISKWRK(I = ISKWRK(IiNSUM)) starts an array containing sums
        of the stoichiometric coefficients of the (gas-phase only)
        species in the surface reactions;
        ISKWRK(I + N - 1) is the sum of gas-phase species
        stoichiometric coefficients for the Nth surface reaction.
IiICOV, ISKWRK(I = ISKWRK(IiCOV)) starts an array of reaction
        indices for those with coverage parameters;
        ISKWRK(I + N - 1) is the reaction index of the Nth reaction
        with coverage parameters.
IiKCOV, ISKWRK(I = ISKWRK(IiKCOV)) starts an array of species
        indices for the coverage-dependendent species in
        the surface reactions with coverage parameters;
        ISKWRK(I + N - 1) is the coverage-dependent species for the
        Nth reaction with coverage parameters.
IiIREV, ISKWRK(I = ISKWRK(IiIREV)) starts an array of reaction
        indices for those with explicit reverse Arrhenius parameters;
        ISKWRK(I + N - 1) is the reaction index of the Nth reaction
        with explicit reverse parameters.
IiISTK, ISKWRK(I = ISKWRK(IiISTK)) starts an array of reaction
        indices for those with sticking coefficients;
        ISKWRK(I + N - 1) is the reaction index of the Nth sticking
        coefficient reaction.
IiMSTK, ISKWRK(I = ISKWRK(IiMSTK)) starts an array of 0/1 flags
        for Motz-Wise rate correction for the surface reactions
        with sticking coefficients;
        ISKWRK(I + N - 1) = 1, use Motz-Wise rate correction for
        the Nth sticking reaction.
IiIBHM, ISKWRK(I = ISKWRK(IiIBHM)) starts an array of reaction
        indices for the Bohm surface reactions;
        ISKWRK(I + N - 1) is the reaction index of the Nth Bohm
        reaction.
IiKBHM, ISKWRK(I = ISKWRK(IiKBHM)) starts an array of species
        indices used in the Bohm formulation;
        ISKWRK(I + N - 1) is the index of the species used for the
        Nth Bohm reaction.
IiIRNU, ISKWRK(I = ISKWRK(IiIRNU)) starts an array of reaction
        indices for those with real stoichiometric coefficients;
        ISKWRK(I + N - 1) is the reaction index of the Nth real
```

```
        stoichiometry reaction.
IiIORD, ISKWRK(I = ISKWRK(IiIORD)) starts an array of reaction
        indices for those with changed-order species;
        ISKWRK(I + N - 1) is the reaction index of the
        Nth change-order reaction.
IiKORD, ISKWRK(I = ISKWRK(IiKORD)) starts a matrix of changed-
        species indices for the change-order reactions;
        ISKWRK(I + (N-1)*MAXORD + L - 1) is the index of the Lth
        changed-order species in the Nth change-order reaction.
IiIONS, ISKWRK(I = ISKWRK(IiIONS)) starts an array of species
        indices for the ionic species.
        ISKWRK(I + N - 1) is the species index of the Nth ionic
        species.
IiKTFL, ISKWRK(I = ISKWRK(IiKTFL)) starts an array of indices
        into a temperature array for the species;
        ISKWRK(I + K - 1) is the temperature index for species K.
IiIEDP, ISKWRK(I = ISKWRK(IiIEDP)) starts an array of reaction
        indices for the surface reactions with ion-energy dependence.
        ISKWRK(I + N - 1) is the species index of the Nth ion-
        -energy-dependent reaction.
IiKEDP, ISKWRK(I = ISKWRK(IiKEDP)) starts an array of species
        indices for the energy-dependent ions in the
        ion-energy-dependence reactions;
        ISKWRK(I + N - 1) is the species index of the ion for the Nth
        energy-dependent reaction.
IiIYLD, ISKWRK(I = ISKWRK(IiIYLD)) starts an array of reaction
        indices for those with modified yield;
        ISKWRK(I + N - 1) is the reaction index of the Nth yield-
        modified reaction.
IiYION, ISKWRK(I = ISKWRK(IiYION)) starts an array of species
        indices for the ion in a yield-modified reaction;
        ISKWRK(I + N - 1) is the species index of the ion for the
        Nth yield-modified reaction.
IiKYLD, ISKWRK(I = ISKWRK(IiKYLD)) starts a matrix of yield-
        modification flags for species in yield-modified reactions;
        ISKWRK(I + (N-1)*MAXSPR + L - 1)
        = 1, the Lth species of the Nth yield-modify reaction is to
             be modified,
        = 0, the species is not modified.
ISKWRK POINTERS TO RSKWRK REAL VARIABLES:
IrSKMN, RSKWRK(I = ISKWRK(IrSKMN)) is the minimum difference
        allowed for conservation of mass and site.
IrPATM, RSKWRK(I = ISKWRK(IrPATM)) is the pressure of one standard
        atmosphere (dynes/cm**2).
IrRU,   RSKWRK(I = ISKWRK(IrRU)) is the universal gas constant
        (ergs/mole-K).
IrRUC,  RSKWRK(I = ISKWRK(IrRUC)) is the universal gas constant
        (cal/mole-K).
ISKWRK POINTERS TO START OF RSKWRK ARRAY WORKSPACE:
IrSDEN, RSKWRK(I = ISKWRK(IrSDEN)) starts an array of phase
        densities;
        RSKWRK(I + N - 1) is the density of the Nth phase.
IrKTMP, RSKWRK(I = ISKWRK(IrKTMP)) starts a matrix of the
        dividing temperatures in the thermodynamic fits for
        the species;
        RSKWRK(I + (K-1)*MAXTP + N - 1) is the Nth temperature
        dividing the ranges of coefficients for species K.
IrKTHM, RSKWRK(I = ISKWRK(IrKTHM)) starts a three-dimensional
        array of coefficients for the fits to thermodynamic
        properties for the species;
        RSKWRK(I + (L-1)*NCP2 + (K-1)*NCP2T + N - 1) is the Nth
        polynomial coefficient A(N,L,K) for species K, in the Lth
        temperature range.
IrKDEN, RSKWRK(I = ISKWRK(IrKDEN)) starts an array of species
```

densities;
RSKWRK(I + K - 1) is the density of the species K
(gm/cm**3 for gas or bulk species, gm/cm**2 for surface
species).

IrAWT, RSKWRK(I = ISKWRK(IrAWT)) starts an array of atomic weights;
RSKWRK(I + M - 1) is the atomic weight of element M.

IrKWT, RSKWRK(I = ISKWRK(IrKWT)) starts an array of molecular
weights;
RSKWRK(I + K - 1) is the molecular weight of species K.

IrPAR, RSKWRK(I = ISKWRK(IrPAR)) starts a matrix of Arrhenius
parameters for the surface reactions;
RSKWRK(I + (N-1)*(NSPAR+1) + L - 1) is, if
L=1, the pre-exponential factor (mole-cm-sec-K)
L=2, the temperature exponent
L=3, the activation energy (K)
L=4 is used as a scalar, in sensitivity analysis,
for the Nth surface reaction.

IrCOV, RSKWRK(I = ISKWRK(IrKCOV)) starts a matrix of coverage
parameters for the coverage surface reactions;
RSKWRK(I + (N-1)*NSCOV + L - 1) is the Lth coverage
parameter for the Nth coverage reaction.

IrRPAR, RSKWRK(I = ISKWRK(IrRPAR)) starts a matrix of reverse
Arrhenius parameters for surface reactions which give them
explicitly;
RSKWRK(I + (N-1)*NSPAR + N - 1), for N=1,3, the
reverse parameters for the Nth reverse-parameter reaction,
and for N=4, a scaling factor.

IrEQ, RSKWRK(I = ISKWRK(IrEQ)) starts an array of scalars for
the surface reaction equilibrium constants;
RSKWRK(I + N - 1) is the scalar for the Nth surface reaction.

IrRNU, RSKWRK(I = ISKWRK(IrRNU)) starts a matrix of stoichiometric
coefficients for the surface reactions with real
coefficients;
RSKWRK (I + (N-1)*MAXSPR + L - 1) is the stoichiometric
coefficient for the Lth species in the Nth real-
stoichiometry reaction.

IrNCF, RSKWRK(I = ISKWRK(IrNCF)) starts a matrix of net site-changes
due to the surface reactions;
RSKWRK(I + (N-1)*NNPHAS + L - 1) is the net change in sites
for phase L due to the Nth surface reaction.

IrKORD, RSKWRK(I = ISKWRK(IrKORD)) starts a matrix of species
orders for the surface reactions with species change-orders;
RSKWRK(I + (N-1)*MAXORD + L - 1) is the order for the Lth
change-order species in the Nth change-order reaction.

IrKFT, RSKWRK(I = ISKWRK(IrKFT)) starts an array of the temperature-
dependent portion of forward reaction rates for the surface
reactions;
RSKWRK(I + N - 1) is the forward temperature-dependent rate
for the Nth surface reaction.

IrKRT, RSKWRK(I = ISKWRK(IrKRT)) starts an array of the temperature-
dependent portion of reverse reaction rates for the surface
reactions;
RSKWRK(I + N - 1) is the reverse temperature-dependent rate
for the Nth surface reaction.

IrKT1, RSKWRK(I = ISKWRK(IrKT1)) starts species scratch space.

IrKT2, RSKWRK(I = ISKWRK(IrKT2)) starts species scratch space.

IrPT1, RSKWRK(I = ISKWRK(IrPT1)) starts phase scratch space.

IrPt2, RSKWRK(I = ISKWRK(IrPT2)) starts phase scratch space.

IrIT1, RSKWRK(I = ISKWRK(IrIT1)) starts reaction scratch space.

IrIT2, RSKWRK(I = ISKWRK(IrIT2)) starts reaction scratch space.

IrIT3, RSKWRK(I = ISKWRK(IrIT3)) starts reaction scratch space.

IrPEDP, RSKWRK(I = ISKWRK(IrPEDP)) starts a matrix of parameters
for surface reactions with ion-energy dependence;
RSKWRK(I + (N-1)*NEDPAR + L - 1) is the Lth parameter for

```
        the Nth ion-energy-dependent reaction.
 IrENGI, RSKWRK(I = ISKWRK(IrENGI)) starts an array of ion energies
        for the gas-phase species only;
        RSKWRK(I + K - 1) is the ion energy of species K.
 IrPYLD, RSKWRK(I = ISKWRK(IrPYLD)) starts a matrix of yield
        parameters for the surface reactions with yield modification;
        RSKWRK(I + (N-1)*NYPAR + L - 1) is the Lth parameter for
        yield-modification reaction N.
 IrYNCF, RSKWRK(I = ISKWRK(IrYNCF)) starts a matrix of net site
        changes due to yield changes in the yield-modified reactions;
        RSKWRK(I + (N-1)*NNPHAS + L - 1) is the net change in sites
        for phase L due to the Nth yield-modify reaction.
 ISKWRK POINTERS TO CHARACTER WORKSPACE (for one surface):
 IcENAM, CSKWRK(I = ISKWRK(IcENAM)) starts an array of element names;
        CSKWRK(I + M - 1) is the name of element M.
 IcKNAM, CSKWRK(I = ISKWRK(IcKNAM)) starts an array of species names;
        CSKWRK(I + K - 1) is the name of species K.
 IcPNAM, CSKWRK(I = ISKWRK(IcPNAM)) starts an array of phase names;
        CSKWRK(I + N - 1) is the name of phase N.
 STORING DATA INTO THE ARRAYS is usually accomplished by a
 CALL SKINIT, which reads a linkfile generated by the surface
 mechanism interpreter;
 the linkfile consists of the following records:
 Linkfile information:
 1. FILVER, character*16, the linkfile format version
 2. PRVERS,  character*16, the interpreter program
 3. PREC,    character*16, the machine precision of the linkfile data
             (SINGLE, DOUBLE)
 4. KERR,    logical to indicate whether or not an error was found by
             the interpreter program
 Parameters and constants:
 5. LENISK, LENRSK, LENCSK, minimum lengths required to store linkfile
    data into the integer, real, and character workspace arrays
 6. MAXSPR, MAXTP, NCP, NSPAR, NSCOV,
    NEDPAR, NYPAR, MAXORD
 7. NELEM, NKKGAS, NKKSUR, NKKBLK, NKKTOT,
    NPHASE, NFSUR, NLSUR, NNSUR, NFBLK,
    NLBLK, NNBLK, NIISUR, NIICOV, NIIREV,
    NIISTK, NIICON, NIIBHM, NIIRNU, NIIORD,
    NIIEDP, NIIYLD, NKKION, KEL, MORE
 8. SKMIN
 Surface data:
 9. CSKWRK(ISKWRK(IcMNAM))                              material name
 Element data:
10. ( CSKWRK(ISKWRK(IcENAM) + M - 1), M = 1, NELEM)   names
11. ( RSKWRK(ISKWRK(IrAWT)  + M - 1), M = 1, NELEM)   weight
 Species data:
12. ( CSKWRK(ISKWRK(IcKNAM) + K - 1), K = 1, NKKTOT)  names
13. ( RSKWRK(ISKWRK(IrKWT)  + K - 1), K = 1, NKKTOT)  weight
14. ( (ISKWRK(ISKWRK(IiKCMP)+(K - 1)*NELEM + M - 1),
       M = 1, NELEM), K = 1, NKKTOT)                   composition
15. ( ISKWRK(ISKWRK(IiKCHG) + K - 1), K = 1, NKKTOT)  charge
16. ( ISKWRK(ISKWRK(IiKNT)  + K - 1), K = 1, NKKTOT)  #fit temp's
17. ( ISKWRK(ISKWRK(IiKPHS) + K - 1), K = 1, NKKTOT)  phase
18. ( ISKWRK(ISKWRK(IiNSCV) + K - 1), K = 1, NKKTOT)  coverage
19. ( RSKWRK(ISKWRK(IrKDEN) + K - 1), K = 1, NKKTOT)  density
20. ( (RSKWRK(ISKWRK(IrKTMP)+(K - 1)*MAXTP + L - 1),  fit temperatures
       L = 1, MAXTP), K = 1, NKKTOT)
21. ((( RSKWRK(ISKWRK(IrKTHM)+(L-1)*NCP2+(K-1)*NCP2T + N-1),
        N = 1, NCP2), L = 1, NTR), K = 1, NKKTOT)     thermo coeff'nts
 Ion data (if NKKION > 0):
22. NKKION
23. ( ISKWRK(ISKWRK(IiIONS) + K - 1), K = 1, NKKION)  species indices
 Phase data:
```

```
24. ( CSKWRK(ISKWRK(IcPNAM) + N - 1), N = 1, NPHASE)  names
25. ( ISKWRK(ISKWRK(IiPKST) + N - 1), N = 1, NPHASE)  starting species
26. ( ISKWRK(ISKWRK(IiPKND) + N - 1), N = 1, NPHASE)  ending species
27. ( ISKWRK(ISKWRK(IiPTOT) + N - 1), N = 1, NPHASE)  species count
28. ( RSKWRK(ISKWRK(IrSDEN) + N - 1), N = 1, NPHASE)  density
 Reaction data (if NIISUR > 0):
29. ( ISKWRK(ISKWRK(IiNRPP) + I - 1), I = 1, NIISUR)  species count
30. ( ISKWRK(ISKWRK(IiNREA) + I - 1), I = 1, NIISUR)  reactant count
31. ( (ISKWRK(ISKWRK(IiNU) + (I-1)*MAXSPR + N - 1),   stoichiometry
       ISKWRK(ISKWRK(IiNUNK)+(I-1)*MAXSPR + N - 1),   species indices
      N = 1, MAXSPR), I = 1, NIISUR)
32. ( ISKWRK(ISKWRK(IiNSUM) + I - 1), I = 1, NIISUR)  stoich. sum
33. ( (RSKWRK(ISKWRK(IrPAR)+(I-1)*(NSPAR+1)+N-1),
      N = 1, NSPAR), I = 1, NIISUR)                   Arrh. coeff'nts
34. ( RSKWRK(ISKWRK(IrEQ) + I - 1), I = 1, NIISUR)    equil. factor
35. ( (RSKWRK(ISKWRK(IrNCF) + (I-1)*NPHASE+N-1),      phase balance
      N = 1, NPHASE), I = 1, NIISUR)
 Coverage reaction data (if NIICOV > 0):
36. NIICOV, NSCOV
37. ( ISKWRK(ISKWRK(IiICOV) + N - 1), N = 1, NIICOV)  reaction indices
38. ( ISKWRK(ISKWRK(IiKCOV) + N - 1), N = 1, NIICOV)  species indices
39. ( (RSKWRK(ISKWRK(IrKCOV)+(N-1)*NSCOV+L-1),        parameters
      L = 1, NSCOV), N = 1, NIICOV)
 Reverse reaction data (if NIIREV > 0):
40. NIIREV
41. ( ISKWRK(ISKWRK(IiIREV) + N - 1), N = 1, NIIREV)  reaction indices
42. ( (RSKWRK(ISKWRK(IrPAR)+(N-1)*(NSPAR_1)+L-1),     rev. parameters
      L = 1, NSPAR), N = 1, NIIREV)
 Sticking reaction data (if NIISTK > 0):
43. NIISTK
44. ( ISKWRK(ISKWRK(IiISTK) + N - 1), N = 1, NIISTK)  reaction indices
45. ( ISKWRK(ISKWRK(IiMSTK) + N - 1), N = 1, NIISTK)  Motz-wise flag
 Bohm reaction data (if NIIBHM > 0):
46. NIIBHM
47. ( ISKWRK(ISKWRK(IiBHM) + N - 1),  N = 1, NIIBHM)  reaction indices
48. ( ISKWRK(ISKWRK(IiKBHM)+ N - 1),  N = 1, NIIBHM)
Real stoichiometry data (if NIIRNU > 0):
49. NIIRNU
50. ( ISKWRK(ISKWRK(IiRNU) + N - 1), N = 1, NIIRNU)   reaction indices
51. ( (RSKWRK(ISKWRK(IrRNU) + (N-1)*MAXSPR + L - 1),  real coeff'nts
      L = 1, MAXSPR), N = 1, NIIRNU)
Change-order data (if NIIORD > 0):
52. NIIORD, MAXORD
53. ( ISKWRK(ISKWRK(IiIORD)+ N - 1), N = 1, NIIORD)  reaction indices
54. ( (ISKWRK(ISKWRK(IiKORD)+(N-1)*MAXORD + L - 1),  species indices
      L = 1, MAXORD), N = 1, NIIORD)
55. ( (RSKWRK(ISKWRK(IrKORD)+(N-1)*MAXORD + L - 1),  order values
      L = 1, MAXORD), N = 1, NIIORD)
Temperature-dependent reaction data (if NIIEDP > 0):
56. NIIEDP, NEDPAR
57. ( ISKWRK(ISKWRK(IiIEDP)+ N - 1), N = 1, NIIEDP)  reaction indices
58. ( ISKWRK(ISKWRK(IiKEDP)+ N - 1), N = 1, NIIEDP)  species indices
59. ( (RSKWRK(ISKWRK(IrPEDP)+(L-1)*NEDPAR + L - 1),  parameters
      L = 1, NEDPAR), I = 1, NIIEDP)
Yield-modify data (if NIIYLD > 0):
60. NIIYLD, NYPAR
61. ( ISKWRK(ISKWRK(IiIYLD)+ N - 1), N = 1, NIIYLD)  reaction indices
62. ( ISKWRK(ISKWRK(IiYION)+ N - 1), N = 1, NIIYLD)  ion indices
63. ( (ISKWRK(ISKWRK(IiKYLD)+(N-1)*MAXSPR + L - 1),  yield flags
      L = 1, MAXSPR), N = 1, NIIYLD)
64. ( (RSKWRK(ISKWRK(IrPYLD)+(N-1)*NYPAR + L - 1),   yield parameters
      L = 1, NYPAR), N = 1, NIIYLD)
65. ( (RSKWRY(ISKWRY(IrYNCF)+(N-1)*NPHASE + L - 1),  phase balance
      L = 1, NPHASE), N = 1, NIIYLD)
```

INITIAL DISTRIBUTION

UNLIMITED RELEASE

0367 R. J. Buss, 1812
0601 J. Y. Tsao, 1126
0601 M. E. Bartram, 1126
0601 W. G. Breiland, 1126
0601 M. E. Coltrin, 1126 (20)
0601 J. R. Creighton, 1126
0601 P. Ho, 1126
0601 H. K. Moffat, 1126
0827 J. Johannes, 9114
0826 W. Hermina, 9111
0826 D. K. Gartling, 9111
0826 S. N. Kempka, 9111
0826 P. R. Schunk, 9111
0827 R. T. McGrath, 9114
0827 T. J. Bartel, 9114
0827 J. E. Brockmann, 9114
0827 R. B. Campbell, 9114
0827 S. J. Choi, 9114
0827 A. S. Geller, 9114
0827 M. L. Hudson, 9114
0827 J. Johannes, 9114
0827 D. J. Rader, 9114
0827 A. J. Russo, 9114
0827 R. Veerasingam, 9114
0827 C. C. Wong, 9114
0834 A. C. Ratzel, 9112
0834 M. R. Baer, 9112
0834 J. R. Torczynski, 9112
0841 P. J. Hommert, 9100
Attn: R. D. Skocypec,9102
J. H. Biffle, 9103
Elaine D. Gorham, 9104
S. E. Gianoulakis, 9113
W. H. Rutledge, 9115
C. W. Peterson, 9116
1078 C. W. Gwyn, 1302
1078 J. D. McBrayer, 1302
1079 A. D. Romig, 1300
Attn: R. S. Blewer, 1305
G. V. Herrera, 1308
P. Esherick, 1311
L. M. Cecchi, 1326

1109 K. D. Devine, 9224
1111 S. S. Dosanjh, 9221
1111 G. S. Heffelfinger, 9221
1111 S. A. Hutchinson, 9221
1111 J. N. Shadid, 9221
1111 A. G. Salinger, 9221
1423 G. H. Hays, 1128
Attn: G. A. Hebner, 1128
P. A. Miller, 1128
M. E. Riley, 1128
1427 P. L. Mattern, 1100
Attn: S. T. Picraux, 1112
J. Nelson, 1113
T. A. Michalske, 1114
G. A. Samara, 1152
E. B. Stechel, 1153
9001 T. Hunter, 8000
Attn: J. B. Wright, 2200
A. West, 8200
R. C. Wayne, 8400
P. N. Smith, 8500
L. A. Hiles, 8800
9042 C. M. Hartwig, 8345
9042 G. H. Evans, 8345
9042 J. F. Grcar, 8345
9042 S. K. Griffiths, 8345
9042 W. G. Houf, 8345
9042 R. J. Kee, 8303 (10)
9042 R. S. Larson, 8345
9042 A. E. Lutz, 8345
9042 E. Meeks, 8345 (10)
9042 R. H. Nilson, 8345
9042 F. M. Rupley, 8345 (50)
9042 J. W. Shon, 8345
9042 P. A. Spence, 8345
9042 A. Ting, 8345
9052 D. R. Hardesty, 8361
9052 M. D. Allendorf, 8361
9052 L. L. Baxter, 8361
9053 S. R. Vosen, 8362
9053 D. W. Hahn, 8366
9054 W. J. McLean
Attn: C. W. Robinson, 8301
W. Bauer, 8302
R. W. Carling, 8362
R. J. Gallagher, 8366

9055 F. P. Tully, 8353
9055 J. L. Durant, 8353
9055 J. A. Miller, 8353
9057 L. A. Rahn, 8351
9057 J. H. Chen, 8351
9057 T. Echekki, 8351
9057 H. N. Najm, 8351
9057 P. H. Paul, 8351
9141 S. Subramanian, 8800
9161 W. G. Wolfer, 8717
9162 D. A. Buchenauer, 8716
9214 C. F. Melius, 8117
9405 M. T. Dyer, 8700
Attn: M. W. Perra, 8711
M. I. Baskes, 8712
J. C. F. Wang, 8713
G. J. Thomas, 8715
K. L. Wilson, 8716
G. A. Benedetti, 8741
M. R. Birnbaum, 8742
P. E. Nielan, 8743
W. A. Kawahara, 8746
D. B. Nelson, 8783
9409 R. H. Stulen, 8250
9021 Technical Communications Department, 8815, for OSTI (10)
9021 Technical Communications Department, 8815/Technical Library, MS 0899, 4414
0899 Technical Library 4414 (4)
9018 Central Technical Files, 8950-2 (3)