Transient Behavior of a Polymer Fill Material with Time Dependent Viscoelastic Properties

John Thompson
ANSYS, Inc.
Steven Groothuis
Micron Technology, Inc.
Hong Tang
SUNY, Buffalo
Paul Koeneman
Independent Consultant

Abstract

In an effort to simplify the implementation of viscoelastic material models into ANSYS, an ANSYS User-Programmable Feature (**UsrFictive**) has been modified to allow a more practical definition of viscoelastic material parameters. Standard ANSYS input of material data may be based upon temperature. Currently, ANSYS material data may not be a function of any other parameter. Many polymers used in industry are initially liquid and must be "cured" in a manufacturing process. This fact means that the properties will change over time. This application demonstrates one method of implementing a curing process simulation for a viscoelastic polymer.

Introduction

In the semiconductor packaging industry, many polymers used in the assembly of packages are viscoelastic in nature. A material is said to be viscoelastic, if the material has an elastic (recoverable) part as well as a viscous (non-recoverable) part. Upon application of a load, the elastic deformation is instantaneous while the viscous part occurs over time.

The viscoelastic model incorporated in ANSYS usually depicts the deformation behavior of glass or glassy materials and may simulate cooling and heating sequences of such material (**Reference 1**). These materials at high temperatures turn into viscous fluids and at low temperatures behave as solids. Further, the material is restricted to be thermorheologically simple (TRS). The concept of TRS materials implies the material response to a load at a high temperature over a short duration is identical to that at a lower temperature but over a longer duration. The material model is available with the viscoelastic elements **VISCO88** and **VISCO89**.

Analysis

Polymer Cure Process Simulation

The purpose of this study was to determine the mechanical behavior of the underfill polymers during cure. In order to capture the viscoelastic-related shrinkage and stresses that develop during a typical curing process, a finite element model was built (see Figures 1 and 2). This model incorporates (1) a silicon semiconductor chip, (2) a FR-4 laminated substrate, (3) solder joints contacting the chip and substrate, and (4) a polymeric underfill for protecting and mechanically coupling the chip, substrate, and solder joints together. This type of packaging is commonly called a flip chip package, since the silicon chip is flipped upside-down in order to make contacts with the substrate's electrical interconnections.

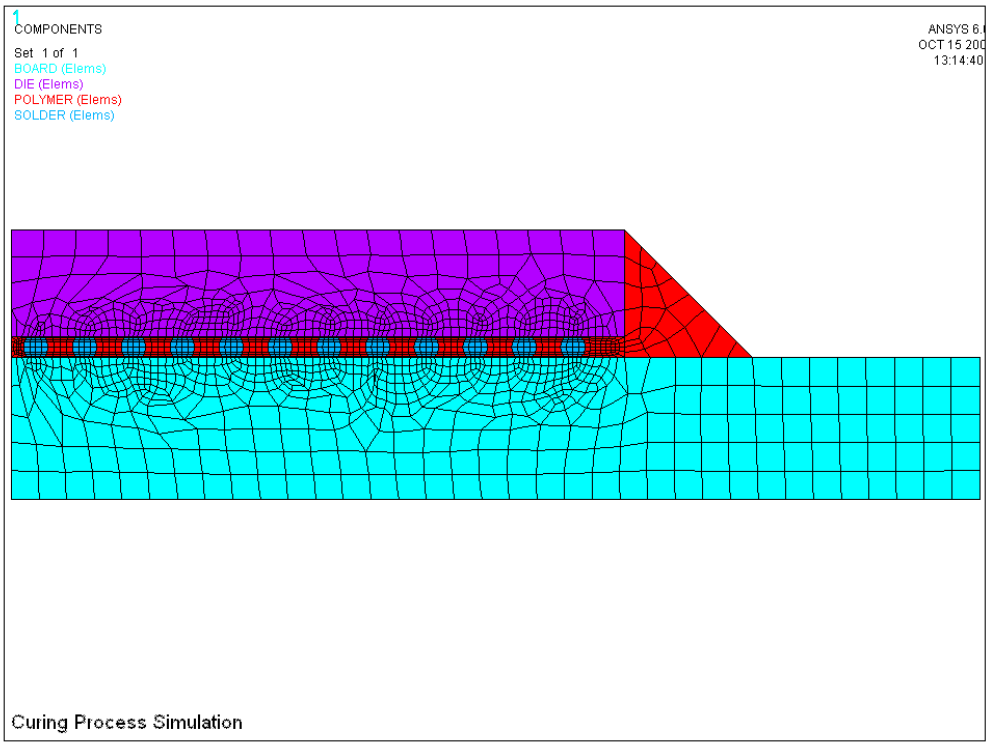

Figure 1 - Polymer Curing Finite Element Model

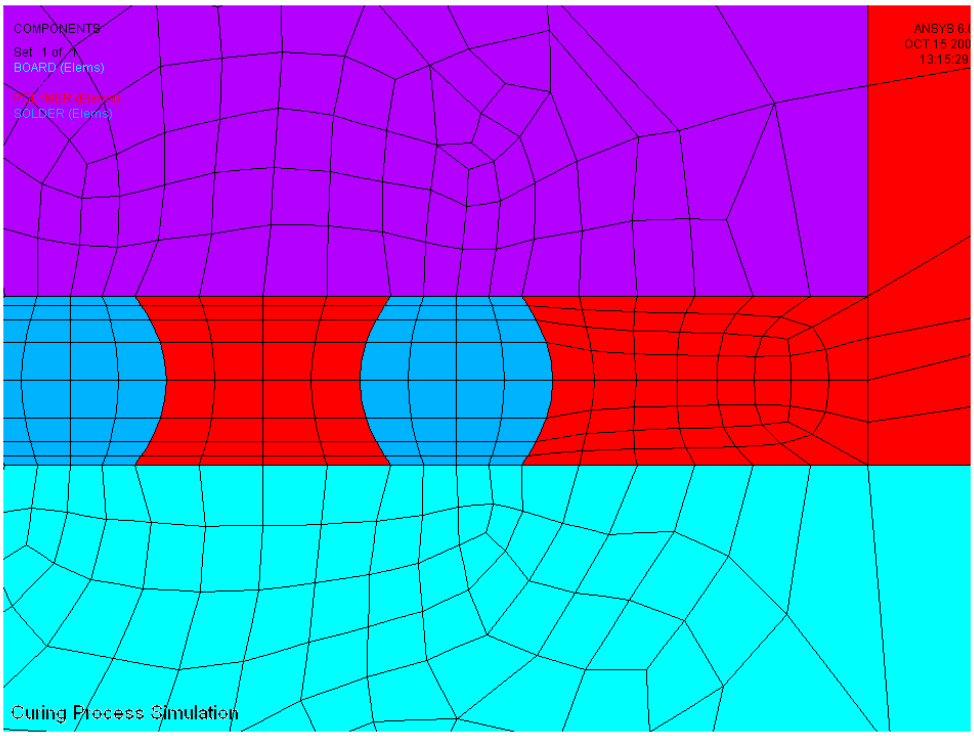

Figure 2 - Polymer Curing Finite Element Model (Zoom region)

The assembly is assumed to be stress-free at (313K). A complete geometric description of the finite element model can be found in the Appendix (**curing.inp**). In addition, the underfill's viscoelastic material properties can be found in the Appendix (**matcure.dat**).

The model provides a method of calculating the residual stress distribution in the underfill and the plastic strain in the solder during cure and cool down. Changes in modulus, relaxation times, and glass transition temperatures can be changed to find an optimal set of material properties. This type of analysis is similar to the work found in **Reference 2**. In developing the simulation procedure for polymer curing, two sets of approaches were constructed to solve the viscoelastic stress simulation.

Standard ANSYS input of material data may be based upon temperature. As currently implemented, the material data may not be a function of any other parameter. Many polymers used in industry are initially liquid and must be "cured" during the manufacturing process. This process means that the properties will change over time. This paper demonstrates one method of implementing a curing process into a viscoelastic polymer used in semiconductor packaging.

Conventional Approach

The underfill polymer is modeled with a VISCO88 element. This element characterizes the time-dependent material properties by a distribution of time constants and an Arrhenius activation energy driven temperature dependence (**Reference 3**). A pseudo time (ζ), based on the time-temperature superposition principle, is used. The equation representing the effective shear modulus (G) is depicted in following equation

$$G(\zeta) = \sum_{i=1}^{I} C_i \left[G(0) - G(\infty)\right] \exp\left(\frac{-\zeta}{\lambda_i}\right) + G(\infty).$$

where

G =

Shear modulus at specific times; G(0) for immediately after the applied load, G(t) for any arbitrary time, and G(∞) for a long time after the applied load.

λ_i =

Relaxation times of the Maxwell viscoelastic elements.

C_i =

Weights associated with the relaxation times.

The stresses are related to the strain at any time by the convolution integral

$$\sigma(t) = \int_0^t G(\zeta - \zeta') \frac{d\varepsilon(t')}{dt'} dt'.$$

where

σ =

Polymeric viscoelastic stress

ε =

Polymeric viscoelastic strain

The integration of time zero allows the representation of the Boltzmann superposition principle. The Boltzmann principle states that the creep at any time is a function of the entire loading history and that each

load step makes an independent contribution to the final deformation. For the VISCO88 element, the fictive temperature (represents the temperature hysteresis) was chosen to be the glass transition temperature (T_g) of the underfill material.

Since the completeness of a polymeric material cure is judged by its mechanical integrity, the mechanical degree of cure (DOC) is calculated by

$$p_m(t) = \frac{G'(t)}{G'_{\max} - G'_{\min}}.$$

where

$p_m(t)$ =

Mechanical degree of cure (DOC) as a function of time

G'_{max} =

Maximum storage modulus achieved at the end of cure

G'min =

Minimum storage modulus of the uncured material

For underfill materials used in semiconductor packaging processes, the viscosity in the uncured liquid state is small, so the minimum modulus is effectively zero.

For this model, cure shrinkage is included into the thermal expansion by calculating an effective coefficient of thermal expansion (α_{eff}). The strains caused by the thermal expansion (ε_T) and chemically-induced cure shrinkage (ε_C) are modeled respectively as

$$\varepsilon_T(t) = \alpha(T(t) - T_0)$$

and

$$\varepsilon_C(t) = bp_m(t)$$

where

α =

Coefficient of thermal expansion (underfill)

T(t) =

Temperature as a function of time

T_0 =

Reference temperature (underfill)

b =

Total chemical strain at the end of cure (negative for shrinkage)

A typical value for volumetric cure shrinkage in unfilled polymers is about 5%. When the effect of the filler particles is accounted for, the resulting linear chemical strain (b) is –0.005 for a typical commercial underfill material. The total strain (ε_{total}) is the sum of the thermal strain and the chemical strain. After some algebraic manipulation, the expression for the total strain is

$$\varepsilon_{total}(t) = \left(\alpha + \frac{bp_m(t)}{T(t) - T_0} \right) (T(t) - T_0).$$

This equation shows that the effective CTE is a function of time and is expressed as

$$\alpha_{eff}(t) = \alpha + \frac{bp_m(t)}{T(t) - T_0}.$$

With the fact that the degree of cure (DOC) and the temperature are known at every time step, effective coefficient of thermal expansion (α_{eff}) can be determined at every time step. As a result, the viscoelastic model requires the material properties to change between load steps. A series of solution restarts were used to account for viscoelastic changes in the material properties for the underfill in a semiconductor package. The appropriate material properties (indicating the degree of cure) are read into the database during the appropriate load step (relating time and temperature).

The resultant equivalent stresses in the overall model and in the viscoelastic underfill material are depicted in Figure 3 and Figure 4, respectively. The maximum equivalent stress in the underfill is 247 MPa.

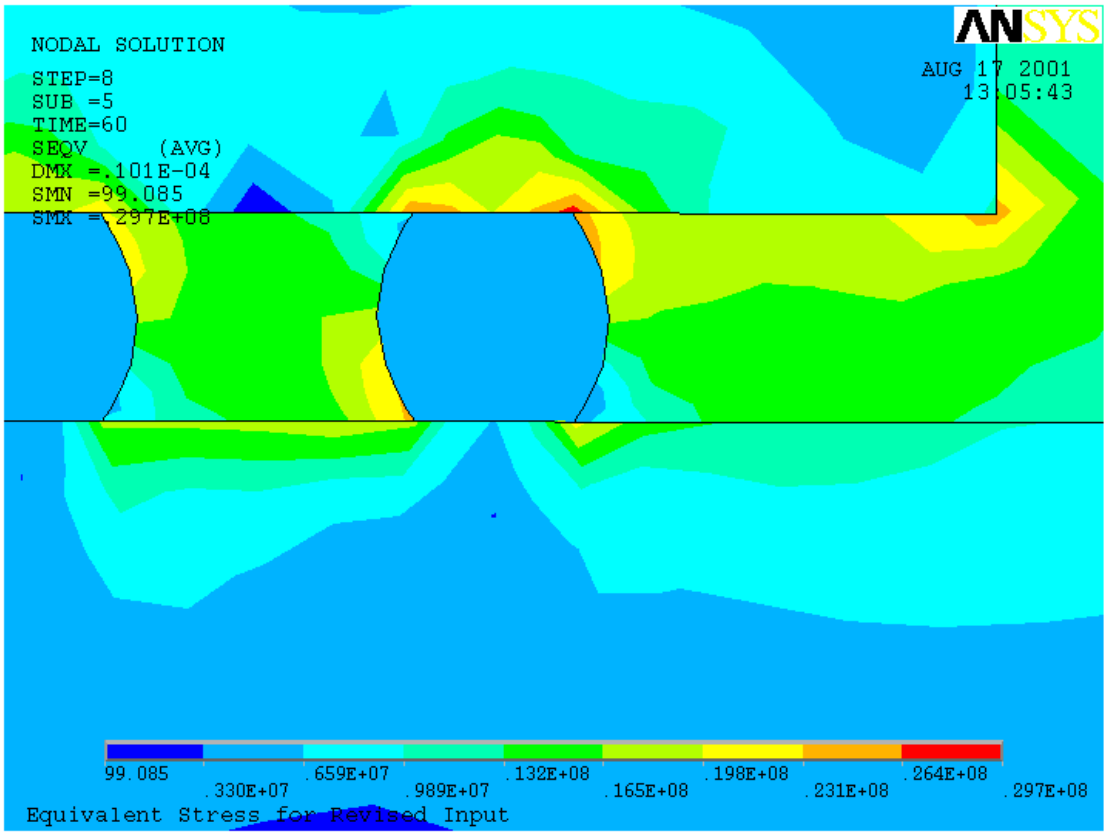

Figure 3 - Overall Equivalent Stress (curing.inp)

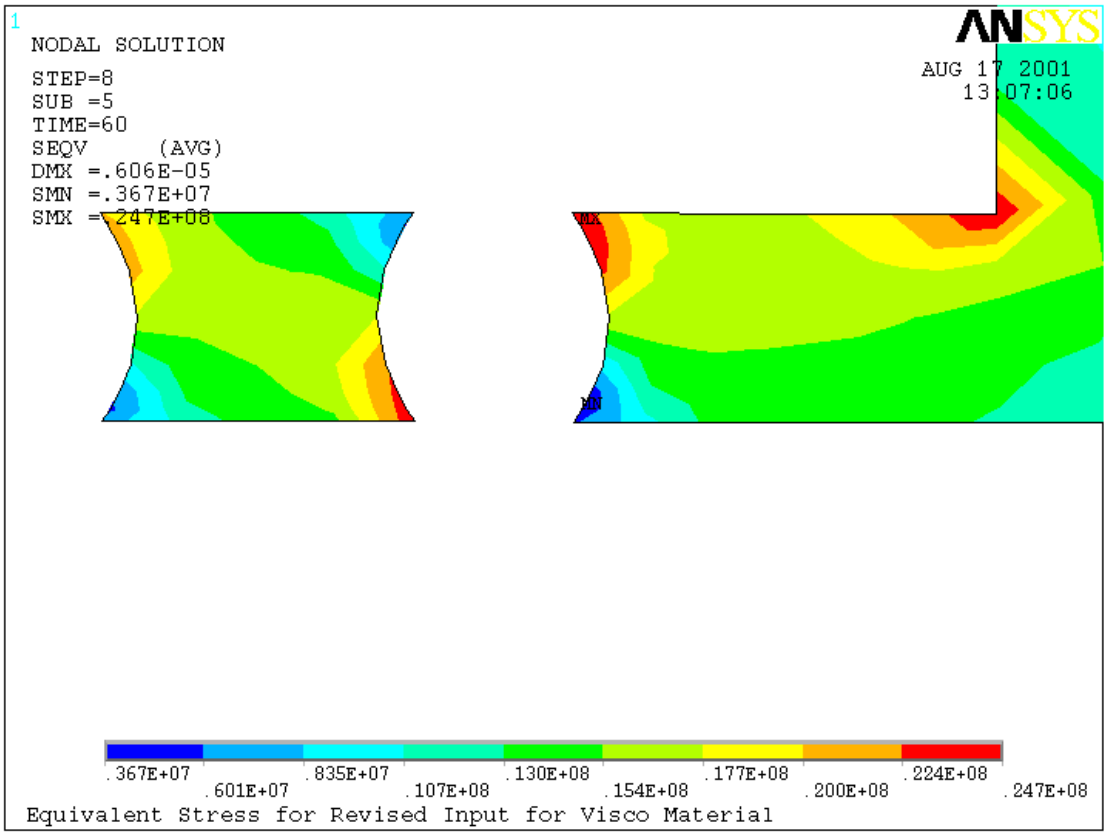

Figure 4 - Equivalent Stress in Viscoelastic Material (using curing.inp)

A more effective approach is through the use of the ANSYS feature called User-Defined Programmable Function (UPF).

User-Defined Programmable Function Approach

In using a combination of the conventional approach and a UPF approach provides a more user-friendly definition of a polymer curing simulation. The **UsrFictive** subroutine in ANSYS is particularly well suited for this type of application (for elements **VISCO88** or **VISCO89**), since the entire viscoelastic material table is passed when the subroutine is called using TBDATA,5,11 in the TB,EVISC command set. Subsequently, the data that enters the subroutine may be replaced with the "appropriate" data based upon time and returned to the calling subroutine where the viscoelastic calculations are performed. Sample Fortran coding for this application is shown in the modified file **USRFICTIVE.F** found in the Appendix. The subroutine has been modified in the following manner:

1. As originally coded, **TIME** is not included in the subroutine either as an argument or in one of the **COMMON** blocks. But in ANSYS, time is included in the **STEPCM.INC** include file and can be made available to the subroutine in this manner.
2. Two arrays are defined to hold the material data for interpolation. Array **CURTIM** is used for the curing time and array **VJMT** is used for the viscoelastic data. Array **VJMT** is made row major to facilitate the interpolation of the data. As coded here, up to 10 times may be input but this may be changed by the user.
3. The material data is made available to the subroutine as an ANSYS parameter. This will permit the user to change the data table without relinking the ANSYS code.

4. The interpolation is done using ANSYS subroutine **INTRP**. This is a standard ANSYS subroutine and the file header is included that identifies the meaning of the arguments.

Using a direct call to UsrFictive subroutine, the resultant equivalent stresses in the overall model and in the underfill material (Figures 5 and 6, respectively) are nearly identical to the conventional results shown in Figures 3 and 4, respectively.

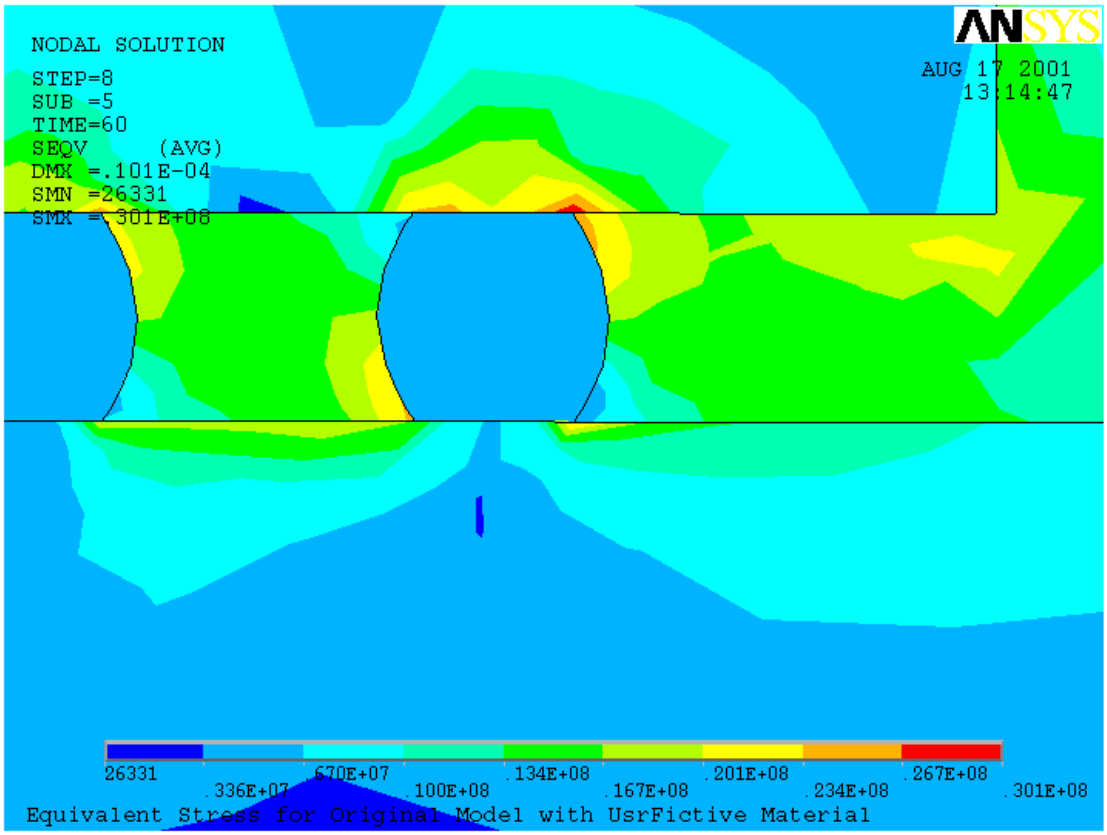

Figure 5 - Overall Equivalent Stress (curing.inp with call to UsrFictive)

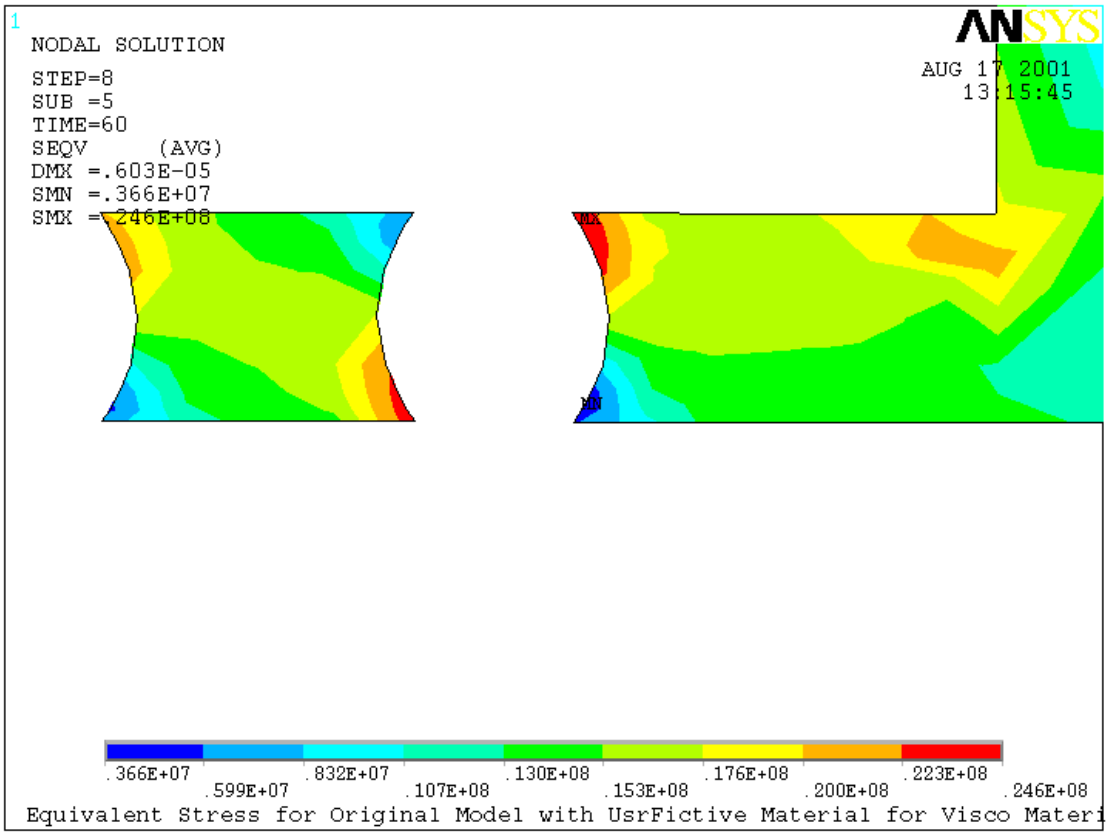

Figure 6 - Equivalent Stress in Viscoelastic Material (using curing.inp with call to UsrFictive)

Discussion

Although viscoelastic material properties require some amount of thermomechanical and chemical testing, the value is realized in more accurate stress simulations. Excellent correlation has been established between the two approaches used in this study. Further use of viscoelastic behavior during polymer curing is being investigated. As this particular finite element model demonstrated, other material models (e.g., viscoplasticity in solders) can be included for more complex, realistic simulations. Stress simulations can be added together to investigate not only the cure process, but also subsequent reliability test conditions like thermal cycling (i.e., solder joint fatigue).

During all simulation runs, the stresses relax to a stable level before the beginning of cool down. Comparison of other underfill materials and material properties have been performed with similar success.

Viscoelastic Material Model Enhancements

The change of material properties with time could be done with a simple shift in temperature, taking advantage of the time-temperature superposition principle. A shift function could be defined that would permit the material properties to change over time by changing the temperature. The down side of this is that the material would return to its apparent liquid state, if the temperature is returned to its initial value. An alternative is to specify the material properties as a simple lookup table whose entry value is based upon time.

Conclusion

In summary, supplying the viscoelasticity material properties via UsrFictive can allow for property definition as a function of another independent variable. In this case, the independent variable was degree of cure (DOC) that was a direct function of time and temperature. Since viscoelasticity is a time dependent phenomenon, the coupling time, temperature, and material property changes is essential.

References

1) O.C. Zienkiewicz, M. Watson, and I.P. King, "A Numerical Method of Visco-Elastic Stress Analysis", International Journal of Mechanical Science, Vol. 10, . 807--827 (1968).
2) P.B. Koeneman, "Viscoelastic stress analysis and fatigue life prediction of a flip-chip-on-board electronic package", Thesis (Ph.D.), University of Texas at Austin, Austin, TX 78712, USA; 1999. 113pp.
3) J.D. Ferry, "Viscoelastic Properties of Polymers," John Wiley & Sons, 2nd edition, 1970.

Appendix

UsrFictive.F (modified)

ANSYS user subroutine that requires relinking the ANSYS program. See the Guide to ANSYS User Programmable Features for details.

```
*deck,UsrFictive          parallel        user                    pck/jmt    001
c sid 5.1 copy of UsrFictive.F last changed by hjm on 98/08/18 14:04:15      002
      subroutine UsrFictive (veinpt,tref,toffst,tem,ftl, ftc)                003
c *** primary function:   allow users to write their own                     004
c                         fictive temperature relationship                  005
c                       this logic is accessed with c5 = 11 on the tb,evisc table
c *** secondary function:    demonstrate the use of a user-written          007
c                            fictive temperature relationship               008
c                                                                           009
c *** notice- this routine contains ansys,inc. confidential information *** 010
c                                                                           011
c         *** ansys(r) copyright(c) 1971,78,82,83,85,87,89,92,94-97         012
c         *** ansys, inc.                                                   013
c                                                                           014
c  input arguments:                                                         015
c     variable (type,size,intent)    description                           016
c                                                                           017
c     veinpt   (dp,ar(95),in)      - table from tb,evisc                    018
c     tref     (dp,sc,in)          - reference temperature                  019
c     toffst   (dp,sc,in)          - temperature offset from absolute zero  020
c     tem      (dp,sc,in)          - temperature at the end of this substep 021
c     ftl      (dp,sc,in)          - previous fictive temperature           022
c                                                                           023
c  output arguments:                                                        024
c     variable (type,size,intent)    description                           025
c     ftc      (dp,sc,in)          - fictive temperature                    026
c                                                                           027
#include "impcom.inc"                                                       028
#include "stepcm.inc"
#include "ansysdef.inc"
                                                                            029
c                                                                           030
      EXTERNAL PAREVL, wrinqr
c  double precision variables                                               031
      double precision  veinpt(95),tref,toffst,tem,ftl, ftc,                032
     x xh,omxh                                                              033
      DOUBLE PRECISION VJMT, CURTIM, CNU(10), PAREVL
      DOUBLE PRECISION XEND, XVAL
      INTEGER II, JJ, KEY, I, KALLONE, KERR, NTIMES
```

```
      integer iott, wrinqr
      CHARACTER*128 NUMNAM, CURNAM, VISNAM

      COMMON /USRVISCO/ VJMT(10,95), CURTIM(10)

      DATA        KALLONE /2/
      DATA        NTIMES  /10/

CCCC  DATA FOR UNCURED UNDERFILL A
CCCC  DATA ARE ENTER COLUMN MAJOR TO FACILITATE CODING
CCCC
c                                                                              034
                                                                               035
c        *** demonstration computation of the fictive temperature (ftc)        036

CCCC
CCCC    THIS ASSUMES THAT THE USER WILL INPUT THE FOLLOWING DATA VIA TB,EVISC
CCCC    EVISC(1) -> EVISC(3) FOR SHIFT FUNCTION VALUES
CCCC    EVISC(5) = 11 TO ACTIVATE THE USER FICTIVE SUBROUTINE

      iott = wrinqr(2)
      NUMNAM = 'NCURET'
      CURNAM = 'JCURE'
      VISNAM = 'JVISCO'

c     write (iott,9000) kallone
c9000 format ('  Entered User Fictive with kallone =',i3)
      IF (KALLONE .GE. 1) THEN
         CNU(1) = 1.0D0
         CNU(2) = 1.0D0
         CNU(3) = 1.0D0
         NTIMES = PAREVL (NUMNAM,0,CNU(1),2,KERR)
         DO 10 II = 1,NTIMES
            CNU(1) = II
            CNU(2) = 1.0D0
            CNU(3) = 1.0D0
            CURTIM(II) = PAREVL (CURNAM,1,CNU(1),2,KERR)
            DO 9 JJ = 1,95
               CNU(1) = JJ
               CNU(2) = II
               VJMT(II,JJ) = PAREVL (VISNAM,2,CNU(1),2,kerr)
    9       CONTINUE
   10    CONTINUE
         KALLONE = -1
c         write (iott,9002) ntimes
c9002    format (/,'   Number of curing temperatures is',i3)
c         write (iott,9003) ntimes, (curtim(ii),ii=1,6)
c9003    format (i5,6e14.6)
c        do 11 jj = 1,95
c           write (iott,9005) jj, vjmt(1,jj),vjmt(2,jj),vjmt(3,jj),
c    x                            vjmt(4,jj),vjmt(5,jj),vjmt(6,jj)
c9005       format (i5,6e14.6)
c  11    continue
      ENDIF


      XVAL = TIMVAL

      IF (XVAL .LE. CURTIM(1)) THEN
         DO 80 II = 6,95
            VEINPT(II) = VJMT(1,II)
   80    CONTINUE

      ELSEIF (XVAL .GE. CURTIM(NTIMES)) THEN
         DO 90 II = 6,95
            VEINPT(II) = VJMT(NTIMES,II)
   90    CONTINUE

      ELSE
         DO 100 II = 6,95
            DO 99 JJ = 1,NTIMES
```

```
              CNU(JJ) = VJMT(JJ,II)
   99       CONTINUE
c        write (iott,9005) ii, (cnu(jj),jj=1,6)
         CALL INTRP (0,0,0,XVAL,CURTIM(1),CNU(1),VEINPT(II),6,KEY)
  100    CONTINUE

      ENDIF

c     WRITE (iott,9001) XVAL, (VEINPT(I),I=46,50)
c9001 FORMAT (/,6e14.6)
C
C    THE FOLLOWING IS THE COMMENT SECTION OF INTRP
C
c     subroutine intrp (klog,kppx,kstpz,xval,ax,ay,yval,nmax,kyoff)
c *** primary function: **** subroutine for single interpolation ****
c                       (if double interpolation is needed, see intrpt)
c
c *** Notice - This file contains ANSYS Confidential information ***
c
c     typ=int,dp,log,chr,dcp   siz=sc,ar(n),func    intent=in,out,inout
c
c  input arguments:
c  variable (typ,siz,intent)      description
c     klog      (int,sc,in)       - interpolation type
c                                   = 0 - use linear interpolation
c                                   = 1 - use log-log interpolation
c                                     -- note:  there is no option yet for
c                                               lin-log or log-lin
c     kppx      (int,sc,in)       - X value end of table signal
c                                   = 0 - a repeated x-value will signal the end
c                                         of the table
c                                   = 1 - a repeated x-value will not signal the
c                                         end of the table
c                                         (only known use = c evaluation)
c     kstpz     (int,sc,in)       - Y value end of table signal
c                                   = 0 - a yval of zero will not signal the end
c                                         of the table (e.g. stress fitting)
c                                   = 1 - a yval of zero will signal the end of
c                                         the table (in general, material
c                                         properties (exception: alpx))
c
c                          NOTE: the end of the table will be signaled thru
c                                either of the above conditions, or more
c                                commonly, that nmax values have been processed,
c                                or that the present x table entry is less than
c                                the previous one (ax(i) .lt. ax(i-1)).
c                                evaluations done after the end of the table are
c                                evaluated as if they were at the end of the
c                                table. similarly, evaluations done before the
c                                beginning of the table are done as if they were
c                                done at the beginning of the table.
c
c     xval      (dp,sc,in)        - value of x with which to go into the table
c     ax        (dp,ar(*),in)     - table of x values, in ascending order
c     ay        (dp,ar(*),in)     - table of y values
c     nmax      (int,sc,in)       - maximum table size allowed
c
c  output arguments:
c     yval      (dp,sc,out)       - value of y which comes back from the table
c     kyoff     (int,sc,out)      - xval status flag
c                                   = 0 - xval in x range
c                                   = 1 - xval less than minimum x
c                                   = 2 - xval greater than maximum x
c

C      xh = veinpt(1)*veinpt(2)                                                037
C      omxh = (1.0d0 - veinpt(2))*veinpt(1)                                    038
C      ftc = veinpt(1)/(tref + toffst) - xh/(tem + toffst) -                   039
C     x         omxh/(ftl + toffst)                                           040

      ftc = 0.0d0
```

```
      return                                                            041
      end                                                               042
```

curing.inp

The input file that includes the restarts to change the viscoelastic material

```
/BATCH
/CLEAR,START
!
!***********************************
! ANSYS RELEASE  5.7.1 version
! - Simulation of Viscoelastic Cure
! - Viscoelasticty polymer (underfill)
! - Anand Viscoplasticity (solder)
!***********************************
!
/filname,CURE
/TITLE,Curing Process Simulation
/PREP7
!
!************* ELEMENT TYPES*********
!
et,1,plane183,,,2
et,2,visco108,,,2,,2,1
et,3,visco88,,,2
!
!******** MATERIAL PROPERTIES *******
!
! Temperature in K
! Moduli & stresses in Pascals
! CTE in 1/K
! Time in minutes
!
! READ-IN "DEGREE OF CURE" VISCO DATA
!
/inp,matcure,dat
!
toffst=0
tref=313
!
!  SILICON MATERIAL PROPERTIES
!
MP,EX,1,106.9E9
MP,ALPX,1,2.6E-6
MP,NUXY,1,0.25
MP,DENS,1,1
!
! SOLDER MATERIAL PROPERTIES
!
MP,NUXY,2,0.40
MP,ALPX,2,25.0E-6
TB,ANAND,2
TBDATA,1,12.41E6,9400,1.44E10,1.5,0.303,1.379E9
TBDATA,7,1.379E7,0.07,1.3
MPTEMP,1,200,450
MPDATA,EX,2,1,4.55E10,0.758E10
```

```
MPTEMP
MP,DENS,2,1
!
! FR-4 BOARD MATERIAL PROPERTIES
!
MP,EX,3,18.6E9
MP,ALPX,3,17.0E-6
MP,NUXY,3,0.20
MP,DENS,3,1
!
! UNDERFILL PROPERTIES (NOT CURED, 5%)
!
TB,EVISC,10
TBDATA,1,1.7264E4
TBDATA,2,0.5
TBDATA,3,1
!
!  NEEDED TO UNCOMMENT NEXT LINE TO
!  USE UsrFictive SUBROUTINE
!
!TBDATA,5,11
TBDATA,6,1.0
TBDATA,26,78.0E-6
TBDATA,31,27.0E-6
TBDATA,36,273
TBDATA,46,0.2875e9
TBDATA,47,0.01e9
TBDATA,48,0.54E9
TBDATA,49,0.54E9
TBDATA,50,4
TBDATA,51,0.2870
TBDATA,52,0.3362
TBDATA,53,0.1623
TBDATA,54,0.2145
TBDATA,61,1E-14
TBDATA,62,1E-14
TBDATA,63,1E-14
TBDATA,64,1E-14
MP,DENS,10,1
!
!****** GEOMETRY MODEL GENERATION ***
!
nyl=4
nxl=1
tsub=1.0e-3
sbh=150e-6
tdie=0.75e-3
dw=4.323e-3
sw=dw+2.5*tsub
sbw=58.0e-6
smw=86.0e-6
scc=dw-367e-6
pitch=344.0e-6
!
*dim,ylayer,array,nyl
*dim,xlayer,array,nxl
ylayer(1)=0.0
```

```
ylayer(2)=tsub
ylayer(3)=tsub+sbh
ylayer(4)=tsub+sbh+tdie
xlayer(1)=0.0
!
*do,j,1,nxl
 *do,i,1,nyl
  k,,xlayer(j),ylayer(i)
 *enddo
*enddo
k,5,sw,ylayer(1)
k,6,sw,ylayer(2)
k,7,dw,ylayer(3)
k,8,dw,ylayer(4)
!
!  SUBSTRATE AREAS
!
a,1,5,6,2
!
!  SILICON AREAS
!
a,3,7,8,4
!
!  SOLDER JOINT AREA
!
k,101,scc-sbw,ylayer(2)
k,102,scc+sbw,ylayer(2)
k,103,scc+smw,(ylayer(2)+ylayer(3))/2
k,104,scc+sbw,ylayer(3)
k,105,scc-sbw,ylayer(3)
k,106,scc-smw,(ylayer(2)+ylayer(3))/2
l,101,102
spline,102,103,104
l,105,104
spline,105,106,101
al,9,10,11,12,13,14
agen,12,3,,,-1.*pitch
!
!  UNDERFILL AREA
!
l,10,101
l,12,105
al,81,14,13,82,17,16
agen,11,15,,,-1.*pitch
l,2,3
l,2,69
l,3,73
al,143,144,80,79,145
k,,dw,ylayer(2)
l,141,102
l,7,141
l,7,104
al,146,147,148,11,10
k,,dw+sbh+tdie,ylayer(2)
l,141,8
l,141,142
l,142,8
```

```
al,149,150,151
asel,all
aglue,all
!
!******** MESHING ******************
!  SOLDER AREA
!
*do,i,0,11
  lccat,10+i*6,11+i*6
  lccat,13+i*6,14+i*6
*enddo
asel,s,area,,3,14
aatt,2,,2
*do,i,0,11
  lesize,9+i*6,,,4
  lesize,12+i*6,,,4
*enddo
*do,i,0,11
lesize,10+i*6,,,4,3.5
lesize,11+i*6,,,4,0.286
lesize,13+i*6,,,4,3.5
lesize,14+i*6,,,4,0.286
*enddo
amesh,all
!
!  UNDERFILL AREA
!
asel,s,area,,15
aatt,10,,3
lesize,81,(pitch-2*sbw)/4
lesize,82,(pitch-2*sbw)/4
amesh,all
asel,s,area,,30,39
aatt,10,,3
*do,i,152,171
lesize,i,(pitch-2*sbw)/4
*enddo
amesh,all
asel,s,area,,26
aatt,10,,3
lesize,144,(pitch-2*sbw)/5
lesize,145,(pitch-2*sbw)/5
lesize,143,sbh/4
amesh,all
asel,s,area,,27
aatt,10,,3
lesize,146,(pitch-2*sbw)/4
lesize,148,(pitch-2*sbw)/4
lesize,147,sbh/4
amesh,all
!
!  FR-4 SUBSTRATE
!
asel,s,area,,41
aatt,3,,1
esize,200e-6
amesh,all
```

```
!
!  SILICON CHIP
!
asel,s,area,,40
aatt,1,,1
esize,200e-6
amesh,all
!
!  UNDERFILL FILLET
!
asel,s,area,,29
aatt,10,,3
esize,100e-6
amesh,all
!
!******** APPLY BCS *****************
!
nsel,s,loc,x,0
dsym,symm,x,0
nsel,r,loc,y,0
d,all,uy,0
nsel,all
!
!******** GROUP ELEMENTS ************
!
esel,s,mat,,2
cm,solder,elem
esel,s,mat,,3
cm,board,elem
esel,s,mat,,1
cm,die,elem
esel,s,mat,,10
cm,polymer,elem
esel,all
!
!******** SOLVE *********************
!
/solu
outres,all,last
antype,static,new
nlgeom,on
solcon,on,off
autots,off
ncnv,0
neqit,30
tref,313
toffst,0
!
!******* PARAMETERS ***************
!
CTEr=78.e-6
CTEg=27.e-6
curetemp=438
t10=20.70
t30=21.30
t50=23.00
t70=25.00
```

```
t100=34.50
thold=30-(34.50-25.00)
shrink=-0.005
startemp=313
ramprate=5
reftemp=313
tcool=5
temp10=ramprate*t10+startemp
temp30=ramprate*t30+startemp
temp50=ramprate*t50+startemp
temp70=ramprate*t70+startemp
temp100=curetemp
!
deltim,0.1
tunif,startemp
time,1e-5
solve
!
!******** CURING PROCESS ***************
!  RAMP TO THE TIME OF 10% CURING DEGREE
!  5% CURING DEGREE PROPERTY USED
!
kbc,0
tunif,temp10
time,t10
save
solve
!
! RAMP TO THE TIME OF 30% CURING DEGREE
! 20% CURING DEGREE PROPERTY USED
!
tunif,temp30
time,t30
save
solve
!
! RAMP TO THE TIME OF 50% CURING DEGREE
! 40% CURING DEGREE PROPERTY USED
!
tunif,temp50
time,t50
save
solve
!
! RAMP TO THE TIME OF 70% CURING DEGREE
! 60% CURING DEGREE PROPERTY USED
!
tunif,temp70
time,t70
save
solve
!
! RAMP TO THE TIME OF 100% CURING DEGREE
! 80% CURING DEGREE PROPERTY USED
!
tunif,temp100
time,t100
```

```
save
solve
!
! HOLD TIME
! 100% CURING DEGREE PROPERTY USED
!
kbc,1
tunif,curetemp
time,t100+thold
save
solve
!
! COOL DOWN TO START TEMPERATURE
!
kbc,0
tunif,startemp
time,t100+thold+tcool
solve
save
finish
!
!******** POST-PROCESSING ***************
!
/post1
allsel
plesol,nl,plwk,0,1
finish
!
/exit
```

matcure.dat

A text file that contains the ANSYS array parameters for the time dependent viscoelastic material.

```
!!!! Parameter data for time dependent viscoelastic properties
*dim,jvisco,array,95,10
*dim,jcure,array,10

!Underfill properties (not cured, 5%) -- #10

!******** Curing process ***************
!** ramp to the time of 10% curing degree
!** 5% curing degree property used
jcure(1)=0.10
jvisco(1,1)=1.7264E4,0.5,1,0,11,1.0
jvisco(6,1)=1.0
jvisco(26,1)=78.0E-6
jvisco(31,1)=27.0E-6
jvisco(36,1)=273
jvisco(46,1)=0.2875e9,0.01e9,0.54E9,0.54E9,4
jvisco(51,1)=0.2870,0.3362,0.1623,0.2145
jvisco(61,1)=1E-14,1E-14,1E-14,1E-14

!******* Parameters ***************
CTEr=78.e-6
CTEg=27.e-6
```

```
curetemp=438
t10=20.70
t30=21.30
t50=23.00
t70=25.00
t100=34.50
shrink=-0.005
startemp=313
ramprate=5
reftemp=313
tcool=5
temp10=ramprate*t10+startemp
temp30=ramprate*t30+startemp
temp50=ramprate*t50+startemp
temp70=ramprate*t70+startemp
temp100=curetemp

!** ramp to the time of 30% curing degree
!** 20% curing degree property used
jcure(2)=20.7
CTEeffR=CTEr+0.20*shrink/(temp30-reftemp)
CTEeffG=CTEg+0.20*shrink/(temp30-reftemp)
jvisco(1,2)=1.7264E4,0.5,1,0,11,1.0
jvisco(26,2)=CTEeffR
jvisco(31,2)=CTEeffg
jvisco(36,2)=301
jvisco(46,2)=1.15E9,0.05E9,0.54E9,0.54E9,4
jvisco(51,2)=0.2870,0.3362,0.1623,0.2145
jvisco(61,2)=1.03E1,0.224E4,0.634E-3,1.04E-10

!** ramp to the time of 50% curing degree
!** 40% curing degree property used
jcure(3)=21.3
CTEeffR=CTEr+0.40*shrink/(temp50-reftemp)
CTEeffG=CTEg+0.40*shrink/(temp50-reftemp)
jvisco(1,3)=1.7264E4,0.5,1,0,11,1.0
jvisco(26,3)=CTEeffR
jvisco(31,3)=CTEeffg
jvisco(36,3)=329
jvisco(46,3)=2.3E9,0.10E9,0.54E9,0.54E9,4
jvisco(51,3)=0.2870,0.3362,0.1623,0.2145
jvisco(61,3)=2.06E1,0.448E4,1.27E-3,2.08E-10

!** ramp to the time of 70% curing degree
!** 60% curing degree property used
jcure(4)=23.0
CTEeffR=CTEr+0.60*shrink/(temp70-reftemp)
CTEeffG=CTEg+0.60*shrink/(temp70-reftemp)
jvisco(1,4)=1.7264E4,0.5,1,0,11,1.0
jvisco(26,4)=CTEeffR
jvisco(31,4)=CTEeffg
jvisco(36,4)=357
jvisco(46,4)=3.45E9,0.15E9,0.54E9,0.54E9,4
jvisco(51,4)=0.2870,0.3362,0.1623,0.2145
jvisco(61,4)=3.08E1,0.672E4,1.90E-3,3.13E-10

!** ramp to the time of 100% curing degree
```

```
!** 80% curing degree property used
Jcure(5)=25.0
CTEeffR=CTEr+0.80*shrink/(temp100-reftemp)
CTEeffG=CTEg+0.80*shrink/(temp100-reftemp)
jvisco(1,5)=1.7264E4,0.5,1,0,11,1.0
jvisco(26,5)=CTEeffR
Jvisco(31,5)=CTEeffg
jvisco(36,5)=385
jvisco(46,5)=4.60E9,0.20E9,0.54E9,0.54E9,4
jvisco(51,5)=0.2870,0.3362,0.1623,0.2145
jvisco(61,5)=4.11E1,0.896E4,2.54E-3,4.17E-10
Jcure(6)=34.5

!** Hold time
!** 100% curing degree property used
CTEeffR=CTEr+1.00*shrink/(curetemp-reftemp)
CTEeffG=CTEg+1.00*shrink/(curetemp-reftemp)
jvisco(1,6)=1.7264E4,0.5,1,0,11,1.0
jvisco(26,6)=CTEeffR
jvisco(31,6)=CTEeffg
jvisco(36,6)=413
jvisco(46,6)=5.75E9,0.25E9,0.54E9,0.54E9,4
jvisco(51,6)=0.2870,0.3362,0.1623,0.2145
jvisco(61,6)=5.14E1,1.12E4,3.17E-3,5.21E-10
ncuret=6
```