DEVELOPMENT OF RECOMMENDATIONS AND GUIDELINES FOR PAVEMENT REHABILITATION DESIGN PROCEDURES FOR THE STATE OF IDAHO

PHASE 2: Development of a Mechanistic-Based Overlay Design System

Volume I
FLEXOLAY Program Documentation

Submitted to:

Idaho Transportation Department
P. O. Box 7129
Boise, Idaho 83707-1129

by

Fouad M. Bayomy
Principal Investigator

Walid M. Nassar
Graduate Assistant

and

Fawzi Al-Kandari
Graduate Student

University of Idaho
Department of Civil Engineering
Moscow, Idaho 83844-1022

ITD Project No. RP 121, Agreement #95-60
NCATT, U of I Contract No. FM-K372

Final Report - Volume I
June 1996

DEVELOPMENT OF RECOMMENDATIONS AND GUIDELINES FOR PAVEMENT REHABILITATION DESIGN PROCEDURES FOR THE STATE OF IDAHO

PHASE 2: Development of a Mechanistic-Based Overlay Design System

ITD Project Number RP 121, Agreement No. 95-60
NCATT, University of Idaho Contract No. FM-K372

Final Report
Volume I

FLEXOLAY Program Documentation

Submitted to:

Idaho Transportation Department
P. O. Box 7129
Boise, Idaho 83707-1129

by

Fouad M. Bayomy
Principal Investigator

Walid M. Nassar
Graduate Assistant

and

Fawzi Al-Kandari
Graduate Student

University of Idaho
Department of Civil Engineering
Moscow, Idaho 83844-1022

June 1996

Chapter 1

1. Introduction

Dear Bob:

This chapter is a brief description of the project and our approach to develop FLEXOLAY program.

Will be provided later

I was very pressed in time and we concentrated our efforts in the past week to modify the program according to your needs. I did not want to delay the delivery of the main document until I return from Auburn. Once I return I will provide this chapter.

Thanks

CHAPTER 2

2. Development of the Overlay Design System

Reviewing the existing mechanistic procedures, indicated that developing a mechanistic overlay design procedure requires the following steps :

1) Modeling the pavement structure.
2) Identifying the design inputs.
3) Evaluating the response of the pavement to loading.
4) Evaluating the allowable load that a pavement can carry based on specified failure criteria.
5) Performing a damage analysis.

2.1 Pavement Structure Model

The pavement is regarded as a multi-layered elastic system. The application of the multi-layer elastic theory in pavement analysis involves several assumptions[19] as follows :

a) Materials in the pavement layers are assumed to be elastic, homogenous and isotropic.
b) All layers, except the bottom one, are finite in depth and the bottom layer (subgrade) is assumed to be infinite in depth.
c) All layers are assumed to be infinite in extent in the lateral direction.
d) The applied loads are static and load imprints are assumed to be circular.
e) Pavement materials are characterized by a modulus of elasticity (also called dynamic modulus, if asphalt mixtures; or resilient modulus, if untreated granular or soil materials) and Poisson's ratio.

f) Full friction is assumed to have developed between layers at each interface.

These assumptions may be found unrealistic for actual pavement conditions. However, the existing mechanistic procedures showed that applying the multi-layered elastic theory gave acceptable results [1,2,12,13,15,16,18].

2.2 Design inputs

Design inputs can be divided into three categories : material properties, traffic and loading, and environmental factors. Each input category will be identified in the following sections

2.2.1 Material properties

Since the pavement has been regarded as a multi-layered elastic system, the elastic moduli and Poisson ratios must be specified.

The Poisson ratio is defined as the ratio of the lateral strain to the axial strain mesaured by laboratory testing. Because it has a relatively small effect on the pavement response, it is customary to assume reasonable value for design rather than to determine it from actual tests [20]. Table 2-1 shows typical Poisson ratios for paving materials.

The layer moduli values can be determined by performing resilient modulus tests on cores taken from the pavement. This operation disturbs and destroys the pavement components and may be costly. Non-destructive testing (NDT) represents an alternative technique to destructive testing. Deflection measurement by means of devices such as the Benkleman Beam, Dynaflect, Road Rater and Falling Weight Deflectometer (FWD) represent the most commonly used type of NDT. These devices have the ability to generate certain types of loading and measure the deflection response of the pavement. The FWD is becoming the most widely used device since it applies impulse loading which better simulates the actual traffic loading. Pavement layer moduli are predicted from deflection data using backcalculation techniques. Several programs have been developed that perform the backcalculation [21,22].

A study by Bayomy and Shah [4] was performed on selected backcalculation programs including MODULUS 4.0 and EVERCALC 3.3. They

Table 2-1 Poisson ratios for different materials(Ref. 8).

Material	Range	Typical
Hot mix asphalt	0.30 - 0.40	0.35
Portland cement concrete	0.15 - 0.20	0.15
Untreated granular materials	0.30 - 0.40	0.35
Cement-treated granular materials	0.10 - 0.20	0.15
Cement-treated fine grained soils	0.15 - 0.35	0.25
Lime-flyash mixtures	0.10 - 0.15	0.15
Loose sand or silt sand	0.20 - 0.40	0.30
Dense sand	0.30 - 0.45	0.35
Fine-grained soils	0.30 - 0.50	0.40
Saturated soft clay	0.40 - 0.50	0.45

investigated existing FWD deflection data at the Idaho Transportation Department for selected pavement sections represent different pavement layer combinations. From the analysis of the backcalculated moduli values determined by the selected programs, they developed some recommendations and guidelines to be followed when using these programs in order to minimize layer moduli prediction errors. This study adopts the Modulus 4.0 as a software for backcalculating layer moduli values of existing pavements.

It is well known that granular materials and subgrade soils are nonlinear with the resilient modulus varying with the level of stresses.

A simple but more popular relationship between the resilient modulus of granular materials and the state of stresses can be expressed as [23] :

$$M_r = K_1 \theta^{K_2} \tag{2-1}$$

in which K_1 and K_2 are experimentally derived constants and θ is the bulk stress, which is the sum of the principal stresses, i.e.,
$\theta = \sigma_1 + \sigma_2 + \sigma_3$.

For fine-grained soils, the stress dependent behavior can be described as [24] :

$$M_r = K\sigma_d^n \tag{2-2}$$

in which K and n are experimentally derived constants and σ_d is the devitoric stress, $\sigma_d = \sigma_1 - \sigma_3$.

Some design procedures [e.g. Asphalt Institute DAMA program) [8], consider the subgrade as linear elastic. This is a reasonable approximation because the variation of modulus due to change of subgrade stresses is usually quite small [8,20].

In the proposed design procedure, all layers are assumed to be linear elastic with constant elastic modulus, unless indicated otherwise.

In the nonlinear elastic case, these layers and the required parameters must be identified. An iterative procedure is used, in which the moduli of nonlinear layers are adjusted as the stress varies, while moduli of linear layers remain the same. During each iteration, a constant set of moduli is computed based on stresses obtained from the previous iteration. The process is repeated until the moduli converge to a specified

tolerance. The calculation of stresses will be discussed later in this chapter.

2.2.2 Traffic and Loading

Traffic is a major input parameter in the pavement design process. The consideration of traffic should include both loading magnitude and configuration, and number of load repetition.

The wheel configuration used in the design procedure is based on a single axle load supported by dual tires. The required loading magnitude is based on one tire on one side (e.g. for 18 kip ESAL, the input wheel load is 4.5 kip). To account for the dual tire action it is also required to specify the tire spacing from center to center. The tire pressure is used to determine the radius of the contact area between the tire and the pavement (80 psi is a typical value) as :

$$a = \sqrt{\frac{F}{\pi P}} \qquad (2\text{-}3)$$

Mixed traffic volume is often analyzed and converted into 18-kip equivalent single axle load (ESAL) repetitions. This study is based on the AASHTO method for converting mixed traffic to ESALs.

It is very important, however, that both number of ESALs already sustained by the pavement since its construction, and number of ESALs expected in its design life be estimated. The past ESALs figure is important for determining the amount of structural capacity left in the pavement, and the expected number of future ESALs is necessary for the overlay design.

2.2.3 Environment

The seasonal variation of material properties is critical factor for an overlay design procedure. Since the expected life of an overlay ranges from 4 to 20 years, the behavior of the pavement material throughout this life must be evaluated.

2.2.4 Subgrade Soils

The environment plays an important role in establishing subgrade resilient modulus. The modulus of soils decreases substantially when moisture content increases. Temperature cycling can alter the modulus. Frozen subgrades may exhibit an increase in their resilient modulus compared to the condition where the subgrade is considered normal. The thawing process substantially reduces the resilient modulus compared to the condition where the subgrade is considered normal.

A study conducted by Hardcastle [5] resulted in proposing a method for estimating seasonal values of resilient moduli of subgrade soils found in Idaho state . The study has identified the following :

a) Idaho pavement operation climate zones.
b) The various seasons in an average year, and most importantly, the time boundary of each season.
c) Adjustment coefficients to account for seasonal variations in subgrade resilient modulus .

The state of Idaho is divided into six pavement climate zones; This is based on six geographic areas, each having approximately equal climate parameters such as the annual air temperature and precipitation and equal climate indices, such as Thornthwaite Moisture Index and Freezing Index. The zone boundaries and their characteristics as presented by Hardcastle [5] are shown in Fig. 2-1. The Climate zones and their characteristics are used to determine the magnitude of the expected moisture changes for the various soil groups as a function of location, and to define the duration and onset dates of the possible operating periods. For each period there are corresponding subgrade conditions and resilient moduli values. In zone 1,2,4 and 5, which experience significant subgrade frost penetration, an average year in a pavement's life is divided into four periods :

- summer (normal) period.
- Freezing transition period.
- Winter (frozen) period
- spring-thaw recovery

In zone 3 and 6 which do not experience significant subgrade frost penetration, the average year is divided into three periods :

- summer (normal) period
- winter-spring (wet) period
- wet recovery period

The calculation of duration, and onset date of each operating period were performed on six representative locations, Driggs(Zone 1), Idaho Falls 46W(Zone 2), Twin Falls(Zone 3), Powell(Zone 4), McCall(Zone 5) and Moscow(Zone 6). The recommended results as presented by Hardcastle are shown in Table 2-2 .

Three adjustment coefficients relative to the summer resilient modulus, M_n, have been used in the proposed overlay design procedure. These coefficients are used for adjusting summer resilient modulus for freezing, thawing and wet conditions. Adjustments for freeze-thaw conditions are required in pavement operating climate zones 1, 2, 4 and 5 which experience significant subgrade frost penetration. Pavement operating climate zones 1 and 6 do not experience significant frost penetration; Thus, an adjustment is required to account for the temporary increase in subgrade water content during wet periods.

The resilient modulus of the subgrade, M_f, applicable during Frozen period is computed from the equation

$$M_f = M_n \times R_f \qquad (2\text{-}4)$$

where R_f is freezing adjustment factor determined from Fig. 3-2.

The resilient modulus of subgrade, M_t, applicable during thaw period is computed from equation

$$M_t = M_n \times R_t \qquad (2\text{-}5)$$

where R_t is thawing adjustment factor determined from Fig. 2-3.

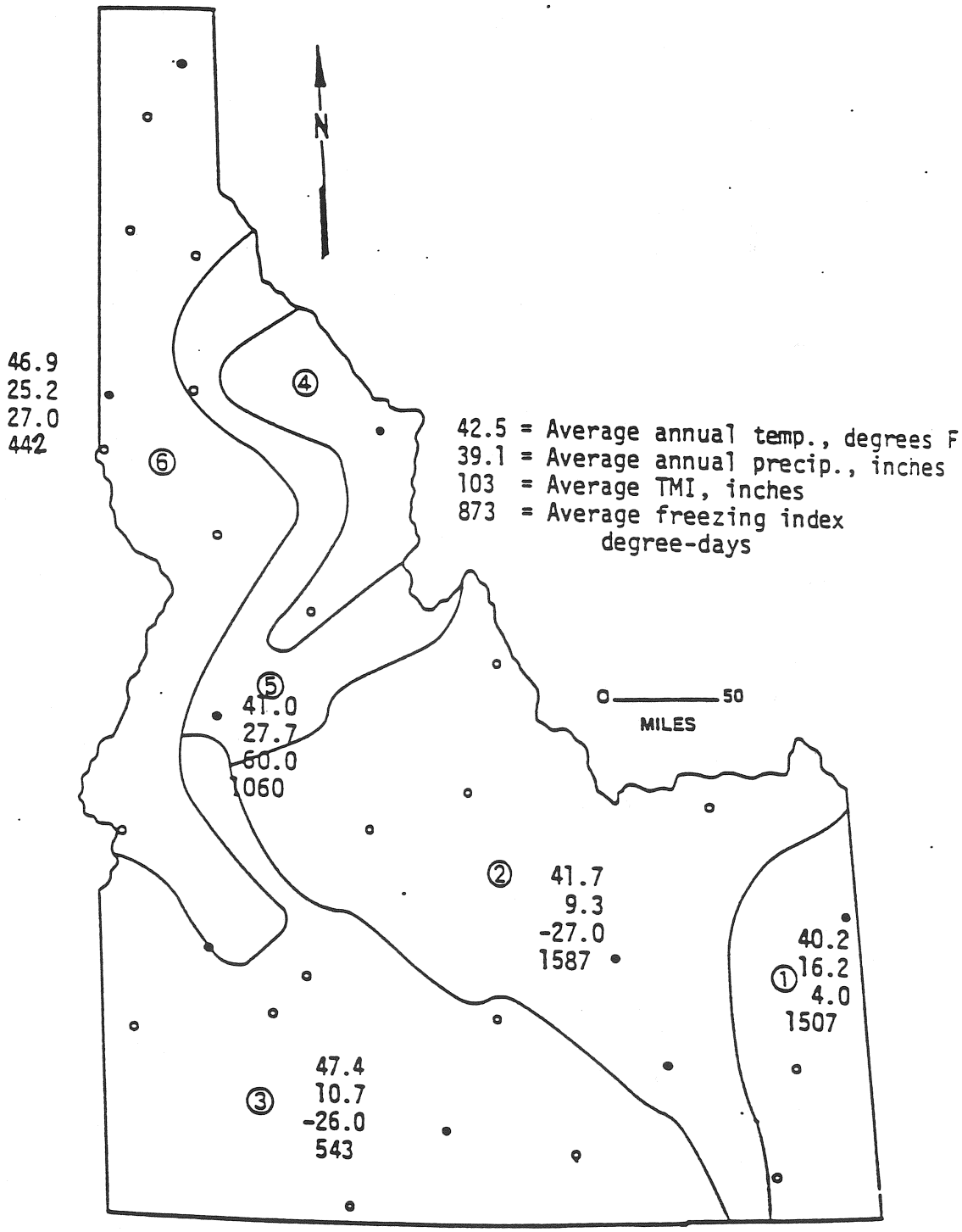


Figure 2-1 Idaho pavement climate zones (Ref. 5)

Table 2-2 Onset dates and duration of pavement operation periods

(Ref. 5)

Climate Zone	Representative Values				Freeeze Transition Perod, days	Winter Frozen or Spring Wet Period		Spring Thaw or Wet Recovery Period		Normal (Summer) Period, days
	Station	Depth Frost inch	Freez Index	Thaw Index		Onset Date	Duration days*	Onset Date	Duration days**	
1	Driggs	52	1507	395	15	Jan 10	120	May 10	38	192
2	Idaho Falls 46W	54	1587	415	9	Jan 3	126	May 9	36	194
3	Twin Falls	26	543	—	—	Feb 1	90	May 1	15	260
4	Powell	37	873	230	44	Feb 10	82	May 3	27	288
5	McCall	42	1062	279	24	Jan 30	110	May 16	24	245
6	Moscow	22	442	—	—	Feb 15	90	May 16	30	245

* Computed for Thow Index = 24 degrees-days

** Computed for Thow Index = 4.154 + 0.259 (AFI)

calculating the resilient modulus during wet periods for zone 3 and 6 requires the determination of the expected temporary increase in subgrade water content during this period form Fig. 2-4. Once the temporary increase in water content is determined, the temporary reduced resilient modulus for the spring-wet period is computed from equation

$$M_w = M_n \times R_w \tag{2-6}$$

where R_w is adjustment factor due temporary increase in water content determined from Fig. 2-5.

To sum up this section, Fig. 2-6 shows a graphical representation of the annual distribution of the resilient moduli values for the various pavement operating climate zones found in Idaho. It is decided to divide the analysis year into four separate seasons. Winter season represents the frozen period for climate zones 1, 2, 4 and 5 with subgrade resilient modulus equal to the average of M_n and M_f. For climate zones 3 and 6, winter-spring season represents the wet period with subgrade resilient modulus equal to M_w. Spring season represents the thaw recovery period for zones 1, 2, 4 and 5 with subgrade resilient modulus equal to the average of M_t and M_n. For climate zones 3 and 6, spring period represents the wet recovery period with subgrade resilient modulus equal to the average of M_w and M_n. Summer and fall-winter seasons together represent the normal period with subgrade resilient modulus equal to M_n.

2.2.5 Granular Bases and Subbases

Like subgrade soils, granular base and subbase resilient modulus can be affected by the environment. Base and subbase layers are expected to have higher resilient modulus than the subgrade beneath . From the inspection of Fig. 2-2 which shows an inverse relationship between normal resilient modulus M_n and the freezing factor, it is safe to assume that the effect of freezing on the base and subbase layers is negligible. This means that in pavement climate zones 1, 2, 4, and 5, winter (frozen) period resilient modulus for granular layers is the same as the summer (normal) resilient modulus.

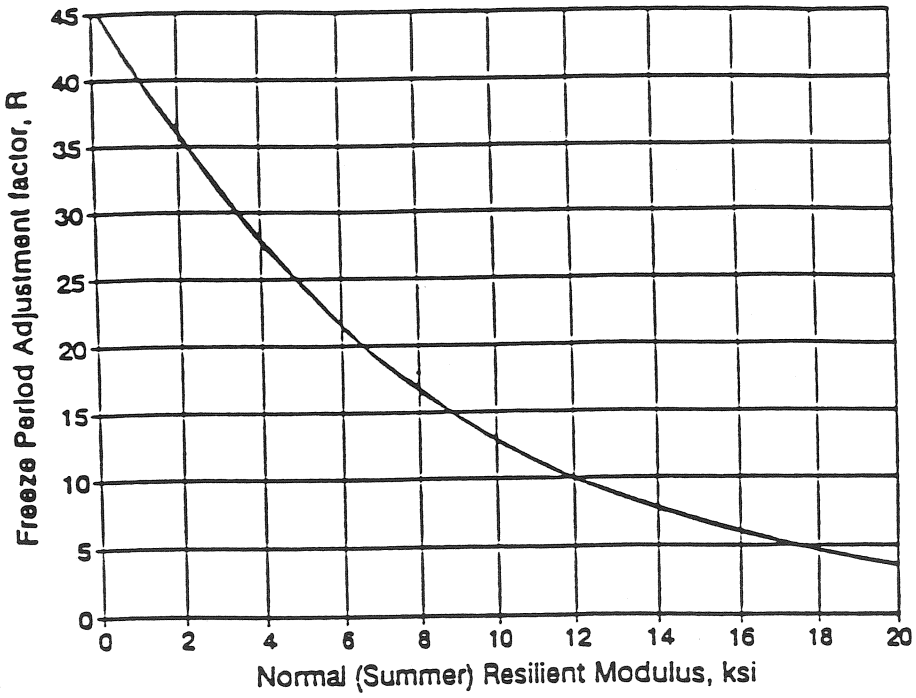


Figure 2-2 Adjustment factor for subgrade freezing (Ref. 5)

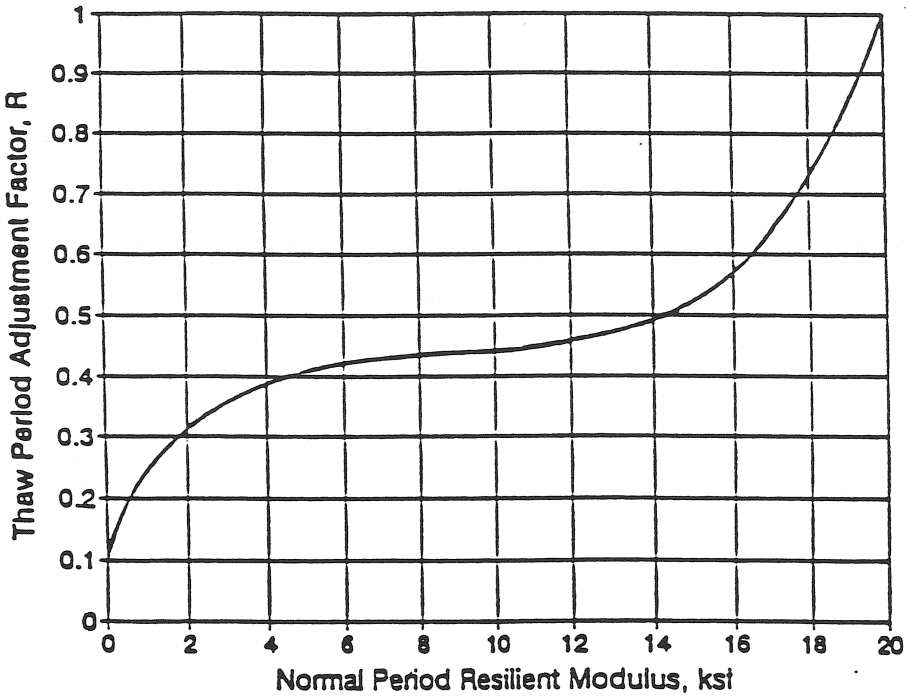


Figure 2-3 Adjustment factor for subgrade thawing (Ref. 5)

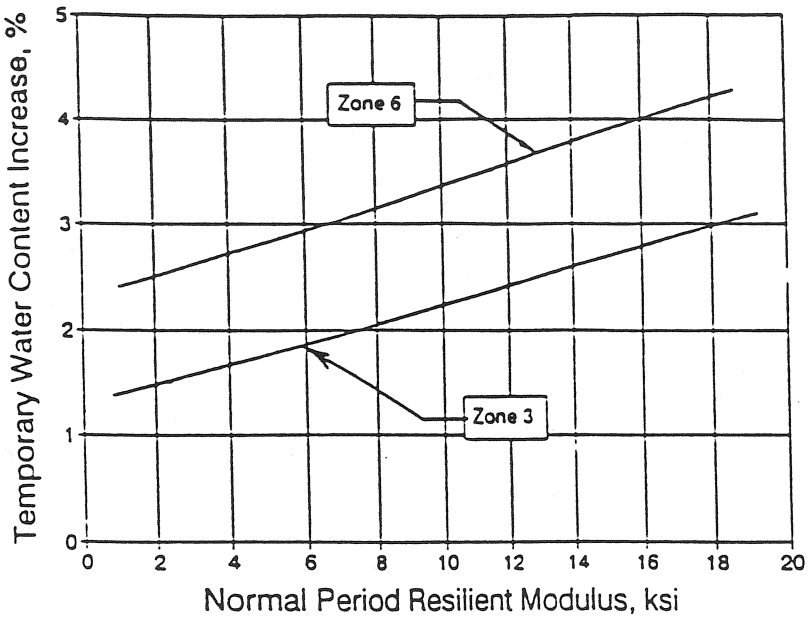


Figure 2-4 Winter-Spring temporary water content increases (Ref. 5).

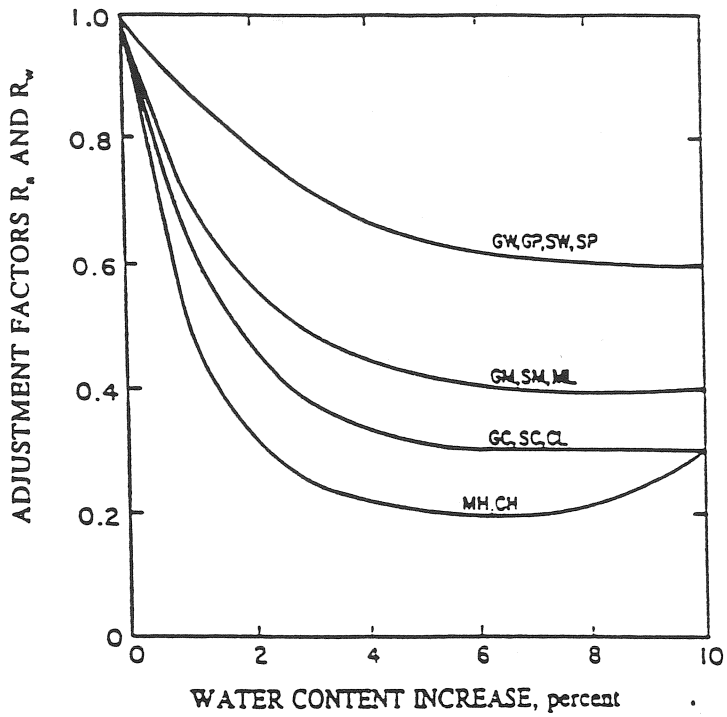
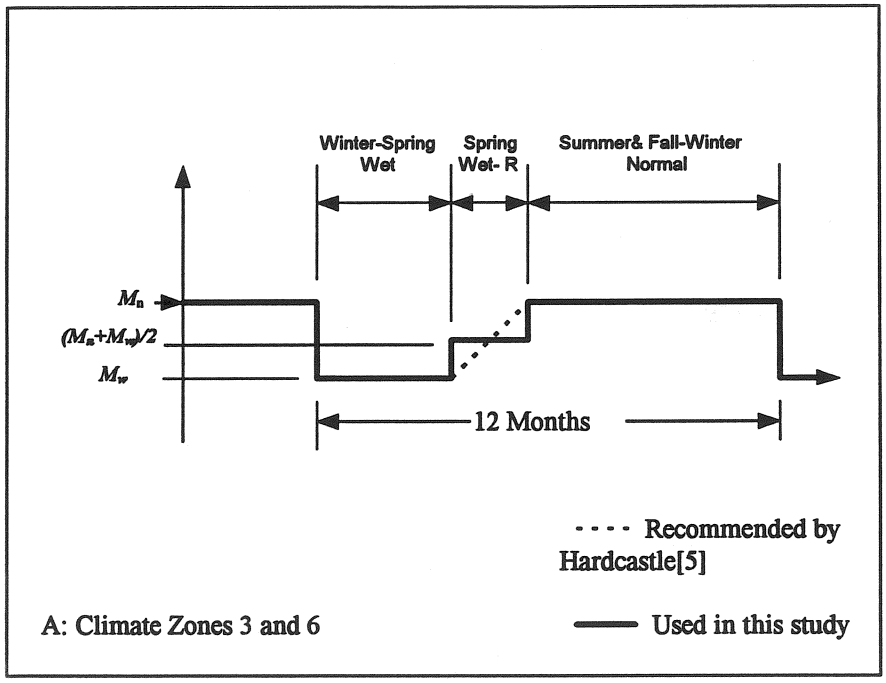


Figure 2-5 Adjustment factors for post-construction water content increases (Ref. 5).



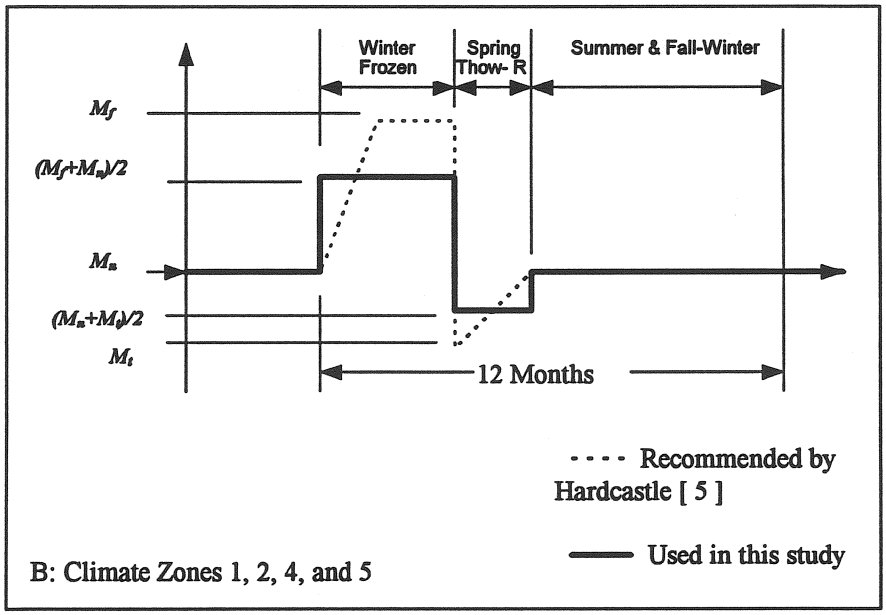


Figure 2-6 Graphical representation of subgrade modulus variations throughout the year for pavement climate zones in Idaho.

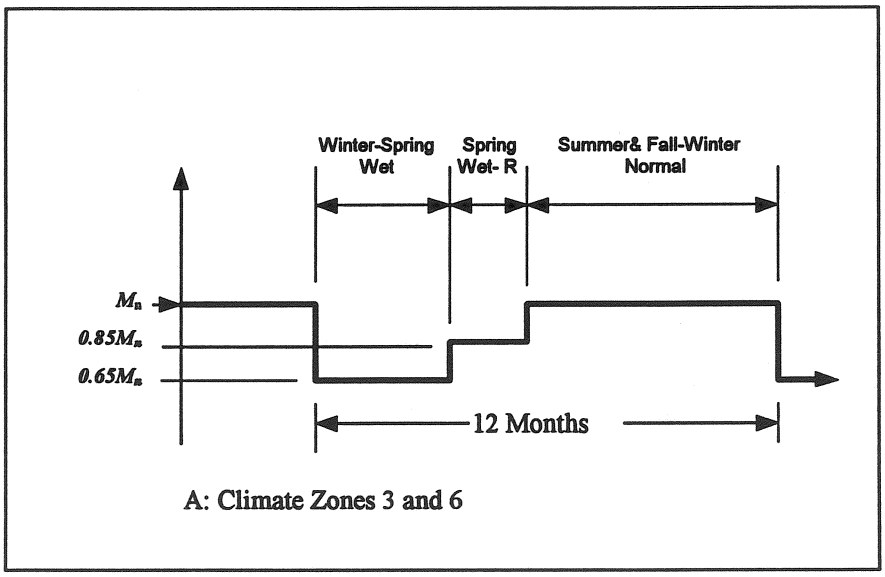
In a study conducted by Rutherford [25] involving seasonal pavement responses, he proposed reducing the Freeze-thaw period resilient modulus of granular soils between 25 to 50 percent of their normal values. Values in this range were used by some agencies [16,18]. In this study, an adjustment factor of 0.65 is applied to the normal period resilient modulus to determine the freeze-thaw period resilient modulus for the base and subbase layers. Assuming that Fig. 2-4 and Fig. 2-5 can be applied to granular bases and subbases, and knowing that the normal modulus of these layers may exceed 20 ksi, it is clear that a value of 0.65 may be a reasonable adjustment factor to account for water content increase during the wet period for zone 3 and 6. The adjustment factor for the wet-recovery period is taken as the average of the wet and the normal period resilient modulus. Fig. 2-7 shows a graphical representation of the annual distribution of the resilient moduli values for bases and subbases in Idaho operating pavement climate zones.

2.2.6 Asphalt Concrete Materials

Asphalt concrete is quite temperature sensitive, exhibiting modulus increase at lower temperatures and modulus decrease at higher temperatures. The determination of seasonal moduli values of asphalt concrete materials requires selection of a representative seasonal pavement temperature and then evaluating the asphalt concrete modulus at that temperature. A model developed by Witczak [26] incorporated in the Asphalt Institute's pavement design DAMA program [8] relates the mean pavement temperature, M_p and mean monthly air temperature, M_a as :

$$M_p = M_a \left(1 + \frac{1}{z+4}\right) - \frac{34}{z+4} + 6 \qquad (2\text{-}7)$$

in which z is the depth below surface in inches and it is usually calculated to the mid-depth of the asphalt layer.



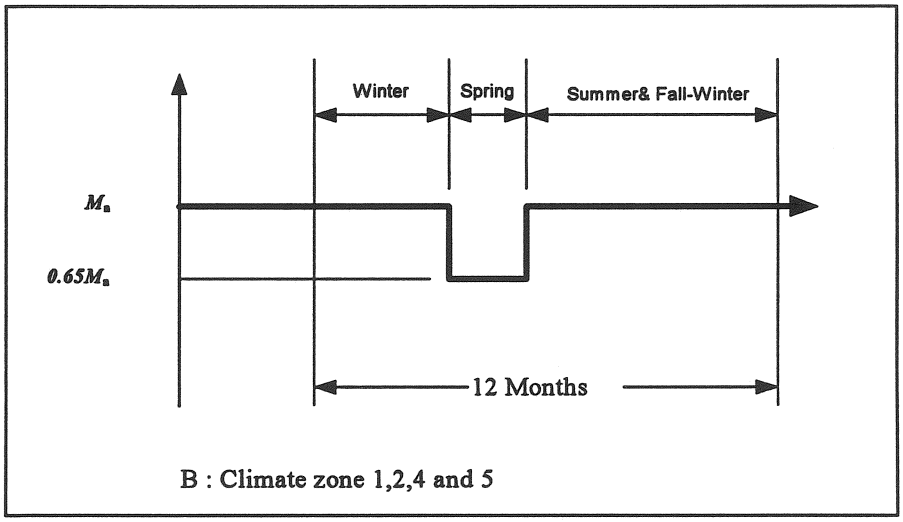


Figure 2-7 Graphical representation of seasonal granular base and subbase moduli values.

To determine the seasonal mean air temperature, maximum and minimum Daily air temperature data files for the representative locations of each climate zone were collected for the past 10 years form Idaho State Climatologist. These data files were loaded in a spread sheet software and analyzed. Table 2-3 shows a summary of the mean air temperatures determined for each season. Summer season duration was obtained by considering the hottest months in the year that follow the thaw period (spring) for zone 1,2,4 and 5 and the wet-recovery period (spring) for zone 3 and 6.

The modulus of asphalt concrete depends on its material characteristics and testing conditions (loading time and temperature). It can be estimated using SHRP's equation [27]:

$$\log_{10}[E_{ac}] = 0.553833 + 0.28829 * P_{200} * f^{-0.17033} - 0.03476 * V_a + 0.070377 * \eta_{70,10^6} + 0.000005 * [t_p^{(1.3+0.498251\log(f))} * P_{ac}^{0.5}] - 0.00189[t_p^{(1.3+0.49825*\log(f)} * P_{ac}^{0.5} * f^{-1.1}] + 0.931757 * f^{-0.02774} \quad (2\text{-}8)$$

where ;

E_{ac} = AC modulus, x10^5.

V_a = Percent air voids in mix.

f = Test frequency.

t_p = Mid depth AC layer temperature (°F).

P_{200} = Percent Aggregate weight passing #200 sieve.

$\eta_{70,106}$ = Asphalt viscosity at 70° F.

P_{ac} = Percent asphalt content by volume of mix.

The most sensitive variable in this equation is temperature. To avoid the use of this cumbersome equation, a graphical representation of this equation was prepared by Bayomy et al.[4] for an average conventional asphalt mixes as shown in Fig. 2-8.

Table 2-3 Summary of seasonal duration and mean air temperatures for Idaho pavement climate zones.

Climate Zone	Station	Winter Frozen or Spring Wet Period		Spring Thaw or Wet Recovery Period		Normal Summer		Normal Fall-Winter	
		Duration (months)	Temp. (° F)	Duration (months)	Temp. (° F)	Duration (months)	Temp. (° F)	Duration (months)	Temp. (° F)
1	Driggs	4	31	1.5	56	3.5	61	3	34
2	Idaho Falls 46 W	4	32	1.5	58	3.5	65	3	33
3	Twin Falls	3	44	1	58	4	66	4	36
4	Powell	3	33	1	53	4	62	4	36
5	McCall	4	33	1	56	4	60	3	35
6	Moscow	3	48	1	59	4	65	4	34

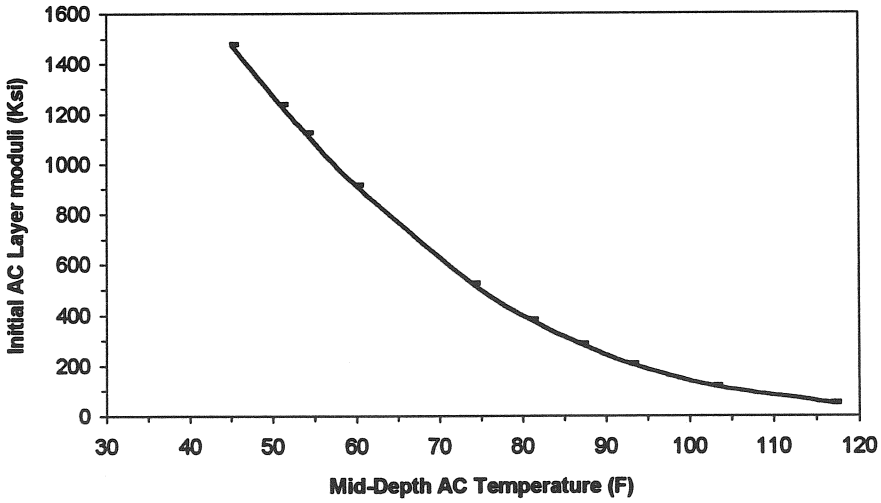


Figure 2-8 Graphical representation of SHRP's equation for the initial AC layer moduli (Ref. 4).

2.2.7 Cement Treated Bases

The modulus of cement treated bases approximately ranges from 1,000,000 to 3,000,000 psi[20]. However, in most cases this modulus will have decreased considerably towards the end of the service life of the pavement structure determined from evaluation measurements (i.e. backcalculated from FWD data). Therefore, it is generally recommended to assume that the cemented base has become unbound material [12]. In this study, the cement treated base is treated as unbound material and is always stress independent.

2.3 Pavement Response

Each material used in the pavement system has a specific tensile strength, compressive strength and shear strength. When the strength of a material is exceeded through repeated or single loading conditions, the material will fail. It is crucial to know precisely where stresses or strains will be at their maximum value and what that maximum value will be.

In flexible pavements the critical tensile strains are located at the bottom of the asphalt layers. The repetition of this strain as a result of the repeated traffic loading progressively damages the asphalt concrete until cracking begins. As the pavement is subjected to additional traffic, a fatigue crack propagates upward. The critical compressive strains are located at the top of the subgrade. Each load application produces some permanent deformation through consolidation of the subgrade material causing a distress called rutting.

The magnitude of stresses and strains induced in a pavement can be calculated using graphical solutions or computer programs, although any calculation for more than one or two layers becomes fairly complicated. Several computer programs have been developed (e.g. CHEVRON, BISAR, ELSYM5 and WES5) to provide solutions for the analysis of multi-layered elastic systems. This study adopted the CHEVRON program, which was developed by the Chevron research company [28] as a routine to handle the calculations of critical stresses and strains. A principal reason for its use is that the software is in the public domain.

The critical strains are calculated by applying the superposition principle for the dual tire action by calculating the strain at the required depth under the center of one tire and the strain at the required depth under a point located at a distance equal to the dual tire spacing.

The presence of stress dependent layers requires the calculation of the principle stresses, σ_1, σ_2 and σ_3 in order to estimate their resilient modulus from the relationships described earlier (e.g. Eqn. 2-1 for granular soils and Eqn. 2-2 for fine soils). It should be noted that the use of layered system for nonlinear analysis is an approximate approach. Thus, the following has been decided for non-linear layers:

- The principal stresses are calculated at a point on the axis of symmetry for a single tire.
- Because most granular materials cannot take any tension, the point at which the principal stresses are calculated for non-linear granular base and subbase is located in the upper quarter and the upper third of the layer. Because the point is in the upper part of the layer, the chance of negative θ is rare [20]. If θ turns out to be negative, a minimum modulus is assigned.
- Because the variation of modulus due the change of subgrade stresses is usually quite small [8], the depth at which the stresses are calculated for the stress dependent subgrade is assumed to be 1.5 the total thickness of the pavement.

2.4 Allowable Traffic and Failure Criteria

Research and field testing have shown that the performance of flexible pavements can be related to certain failure mechanisms. From a structural capacity point of view, flexible pavement may experience two kinds of failure: fatigue, which shows as excessive alligator cracking; and rutting, which shows as permanent deformations along the wheel path.

2.4.1 Fatigue Failure

Several investigations have shown that fatigue failure is best related to the horizontal tensile strain, ε_t, at the bottom of the asphalt layer [5,9,10,11]. Most of the transfer functions found in the literature have the following forms for fatigue failure :

$$N_f = f_1 \varepsilon_t^{-f_2} E_1^{-f_3} \tag{2-9}$$

where N_F is the allowable number of load repetitions to prevent fatigue cracking from reaching certain limit defined by the agency, ε_t is the tensile strain at the bottom of the asphalt layer, E_1 is the modulus of the asphalt layer, and f_1, f_2 and f_3 are coefficients which can be determined from fatigue tests. f_1 must shift from laboratory to field values by calibration. Shift factors may range from 5 to 700 [20].

Because the term E_1 has less effect on N_f than ε_t, some agencies neglect the E_1 term, hence Eqn. 2-8 is reduced to :

$$N_f = f_1 \varepsilon_t^{-f_2} \tag{2-10}$$

values for the exponents in some models used by various agencies are shown in Table 2-4, plotted in Fig. 2-9. The laboratory model developed by Monismith et al. [17] shown in Eqn. 2-11 has been used by various agencies as the base case fatigue model, which has been calibrated by means of shift factors to correlate with field conditions.

$$\log N_f = 14.82 - 3.291 \log\left(\frac{\varepsilon_t}{10^{-6}}\right) - 0.854 \log\left(\frac{E_1}{10^3}\right) \tag{2-11}$$

or

$$N_f = 0.00432(\varepsilon_t)^{-3.291}(E_1)^{-0.854} \tag{2-12}$$

Finn, et al. [28] modified the above equation by applying a shift factor of 18.4 to provide an indication of approximately 20 percent or

Table 2-4 Fatigue coefficients used by various agencies

NO.	Procedure	f_1	f_2	f_3	Ref.
1	ILLINOIS Department of Transportation	5.00E-06	3	0	14
2	Transport & Road Research Laboratory	1.66E-10	4.32	0	20
3	Belgian Road Research Center (BRRC)	4.92E-14	4.76	0	20
4	Austin Research Engineers (ARE)	9.73E-15	5.16	0	1
5	Federal Highway Administration	7.56E-12	4.68	0	13
6	ARIZONA Department of Transportation	9.33E-07	3.84	0	15
7	Asphalt Institute *	0.0796	3.291	0.854	8
8	SHELL Research *	0.0685	5.671	2.363	12
9	U. S. Army Corps of Engineers *	497.156	5	2.665	2

* plotted for asphalt temperature of 70°F (i.e. E1 ≅ 400 ksi)

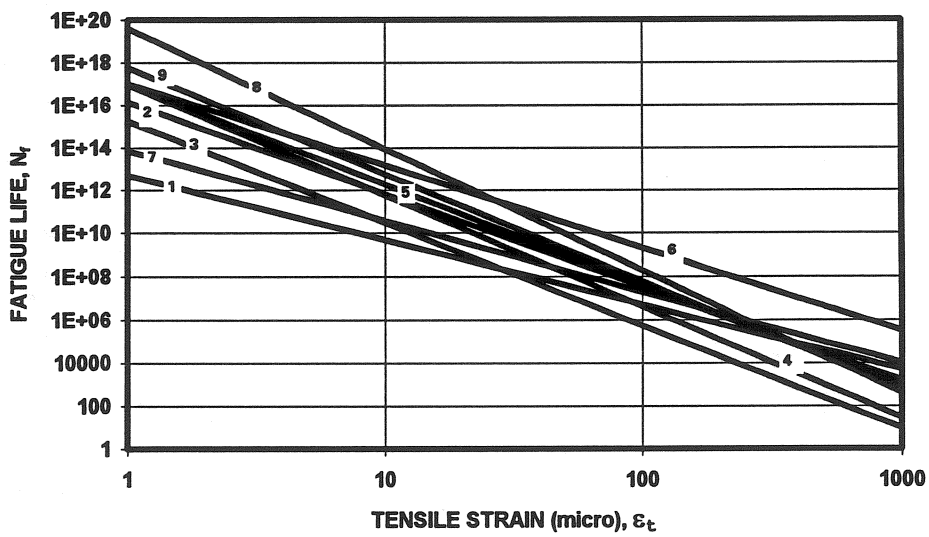


Figure 2-9 Fatigue models used by various agencies.

greater fatigue cracking (based on total area) in selected sections of the AASHO Road Test.

The Asphalt Institute adopted the Finn model and modified it to reflect the effect of both percent air void volume, V_a, and percent asphalt volume, V_b, [8]. The final form of the equation used by Asphalt institute is

$$N_f = 10^M (18.4)\,(0.00432)\;(\varepsilon_t)^{-3.291}(E_{ac})^{-0.854} \tag{2-13}$$

where

$$M = 4.84\left(\frac{V_b}{V_a + V_b} - 0.69\right) \tag{2-14}$$

In this study, the Asphalt Institute model (i.e. Eqn. 2-13), has been adopted as the fatigue criteria. The model was chosen because it has the following features :

1. Most of the models developed are usually accurate for the exact conditions for which they have been developed. However, the presence of asphalt mix properties (E, V_a, V_b) makes the Asphalt Institute applicable to other mixes than those used to develop the model with acceptable results.
2. Since most of the models were developed for new asphalt concrete, a concern of the applicability of the models for the existing asphalt layer will raise. The Asphalt Institute model has been used for stage construction, it has also been applied to the evaluation of existing pavements for designing overlays [8].

2.4.2 Rutting

Permanent deformation or rutting in flexible pavements has been modeled by various agencies as:

$$N_d = f_4 \varepsilon_c^{-f_5} \tag{2-15}$$

where N_d is the allowable number of load repetitions to limit rutting from reaching certain limit defined by the agency, ε_t is the compressive strain on the top of the subgrade and f_4 and f_5 are coefficients which can be determined from lab testing. Values of f_4 and f_5 used by several agencies are shown in Table 2-5, plotted in Fig. 2-10.

As can be seen from Table 2-5, the exponent f_5 falls within a narrow range but the coefficient f_4 varies a great deal. CHEVRON and Asphalt Institute models have almost the same exponents and give average life compared to the rest. It was decided to use the Asphalt Institute model to limit rutting, although without such a procedure, rutting in flexible pavements can best be addressed by improving the material selection and mixture design[8].

2.5 Damage Analysis

Since different loading conditions for each season will be encountered, some methods of combining the effect of different loading conditions must be considered. Miner's hypothesis is the one most often used, which allows an accumulation of damage from the various load conditions to be combined into one damage number[28].

For fatigue, the damage is a measure of the life consumed by all past traffic loading and can be determined as follows :

$$SDR_{past} = \sum_{i=1}^{4} \frac{N_{(past)_i}}{N_{(allow)_i}} \qquad (2\text{-}16)$$

where SDR_{past} is the sum of damage ratio or fraction of life consumed by past traffic, $N_{(past)_i}$ is the number of ESAL applications in season i, and $N_{(allow)_i}$ is the allowable number of ESAL applications for season i determined using Eqn. 2-13 with tensile strain on the underside of the existing pavement. The remaining life, RL, is calculated as :

$$RL = 1 - SDR_{past} \qquad (2\text{-}17)$$

The number of future ESAL applications that the existing pavement can sustain without overlay should satisfy the following :

Table 2-5 Rutting models used by various agencies

NO.	Procedure	f_4	f_5	Ref.
1	Transport & Road Research Laboratory	1.130×10^{-6}	3.57	20
2	Belgian Road Research Center (BRRC)	3.050×10^{-9}	4.35	20
3	U.S. Army Corps of Engineers	1.807×10^{-15}	6.527	2
4	Asphalt Institute	1.365×10^{-9}	4.477	8
5	SHELL Research	6.150×10^{-7}	4	12
6	CHEVRON	1.337×10^{-9}	4.484	16,18

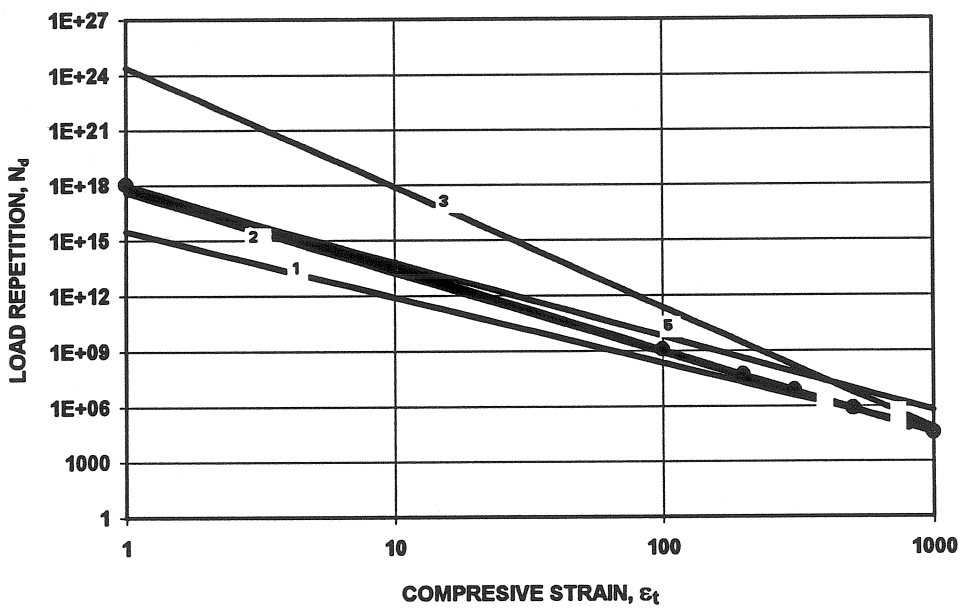


Figure 2-10 Rutting models used by various agencies.

$$\sum_{i=1}^{4} \frac{N_{(future)_i}}{N_{(allow)_i}} \leq RL \qquad (2\text{-}18)$$

For an intact pavement, the sum of damage ratio, SDR_p, should not exceed unity. There are number of possibilities depending on the condition of the existing asphalt layer:

CASE 1

When SDR_{past} exceeds unity (pavement life has been consumed), the existing asphalt layer is assumed to be completely cracked and an overlay is required. In this case, the existing pavement is assigned a reasonable low modulus value. A range from 20 to 70 ksi were used by various agencies for cracked asphalt layers [2,13,15]. In the proposed design procedure a value of 40 ksi at 77° F is used. The sum of damage ratio is calculated for the overlay from the future traffic (ESAL) and the allowable number of ESAL applications as determined from Eqn. 2-13 with tensile strain on the underside of the overlay, i.e.,

$$\text{Overlay } SDR_{future} = \sum_{i=1}^{4} \frac{N_{(future)_i}}{N_{(allow)_i}} \qquad (2\text{-}19)$$

Reaching the optimum overlay thickness requires incrementing the overlay thickness until SDR_{future} is equal to or less than unity.

CASE 2

When SDR_{past} does not exceed unity (existing asphalt layer has remaining life) but the future traffic exceeds the fraction of life remaining an overlay is required. The overlay thickness is determined such that SDR_{future} for the overlay does not exceed unity. For the existing AC layer the future traffic should satisfy the following :

$$\sum_{i=1}^{4} \frac{N_{(future)_i}}{N_{(allow)_i}} \leq RL \qquad (2\text{-}20)$$

where RL is the remaining life and N_{allow_i} is the allowable number of ESAL applications for season i determined using Eqn. 2-13 with tensile strain determined on the underside of the existing asphalt layer with the overlay above it.

The sum of damage ratio for the overlay is determined :

$$\text{Overlay } SDR_{future} = \sum_{i=1}^{4} \frac{N_{(future)_i}}{N_{(allow)_i}} \qquad (2\text{-}21)$$

where $N_{(allow)_i}$ is determined using Eqn. 2-13 with tensile strain determined on the underside of the overlay.

CASE 3

If future traffic does not exceed the existing pavement remaining life, the existing pavement is structurally adequate to sustain the future traffic without overlay.

Some agencies consider only the future traffic for overlay design [15,16] thus neglecting the effect of past traffic on the existing pavement. The assumption behind this procedure is that the pavement system can be modeled with a moduli derived from deflection data which represent the current conditions of the pavement. Other studies have shown that moduli values for a given load and at a given temperature tend to remain constant for a majority of the useful life of the pavement and then decreases only near the end of the pavement life [2]; Thus, neglecting the effect of past traffic may not be valid.

Rutting will normally be of concern at the surface of the overlay. It can be assumed that the existing rut if any, will be filled and that the development of rutting will only be a function of the future traffic to be applied on the "new" pavement structure with the overlay. As with fatigue, the sum of damage ratio can be determined as follows :

$$SDR_{(rutting)} = \sum_{i=1}^{4} \frac{N_{(future)_i}}{N_{(allow)_i}} \qquad (2\text{-}22)$$

where the allowable number of ESAL applications is determined using Eqn. 2-14 with the compressive strain determined on the top of the subgrade.

The final overlay thickness is the one that satisfies both fatigue and rutting requirements.

CHAPTER 3

3. PROGRAM DEVELOPMENT

Two programming languages were used to develop the overlay design program. The main routine, FLEXOLAY.EXE, was developed using VISUAL BASIC language to handle the input-output and file operations. The reason of using such language is to provide user friendly screens to facilitate inputting and editing data, viewing and printing results. The second routine, SYSAN.EXE, which include the multi-layer elastic program CHEVRON, was developed using FORTRAN language and compiled by POWER STATION FORTRAN which produces 32 BIT programs, hence, breaking the 640 KB barrier imposed by the DOS. SYSAN.EXE handles the intensive analysis and calculations, and works interactively with the former routine. The program also includes a third routine, DOSXMSF.EXE which comes with the POWER STATION FORTRAN and can be distributed royalty free to enable the 32 BIT programs produced by the compiler to run under DOS.

3.1 Program structure

The program structure is quite complicated to discuss in a comprehensive way; Therefore, the program steps are illustrated in a series of flow charts, which show briefly the links between the different blocks that describe the program. Fig.3-1 illustrates the main features and blocks of the proposed design program.

Block 1 : This block was developed using VISUAL BASIC to handle the input-output and file operations. The entire block represents FLEXOLAY.EXE. It consists of 2 BASIC modules and 9 forms. Fig.3-2 through Fig.3-7 illustrate the links between the components of the block. The input data has been divided into three categories :

- Pavement data
- Soil parameters data
- General data

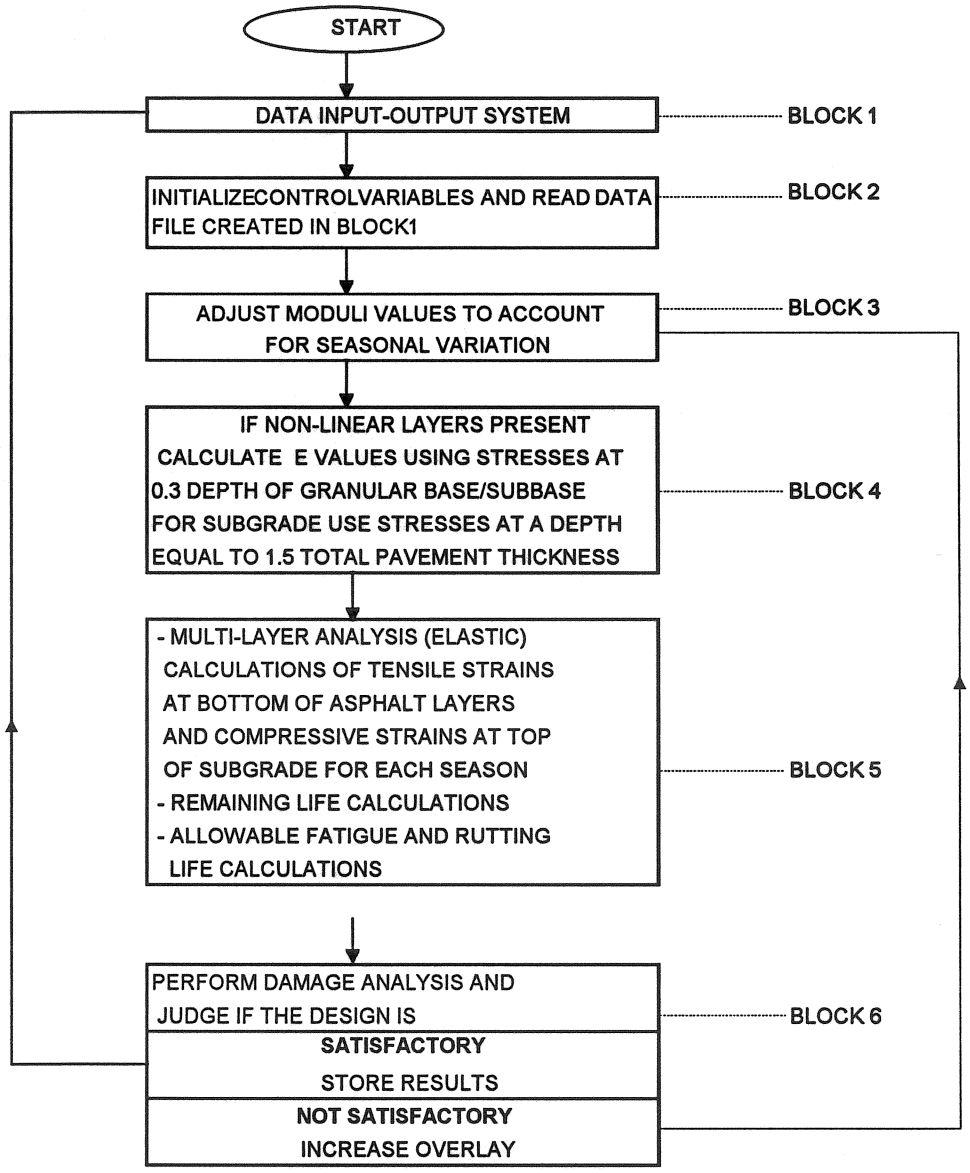


Figure 3-1 Main blocks flow chart of the proposed design procedure.

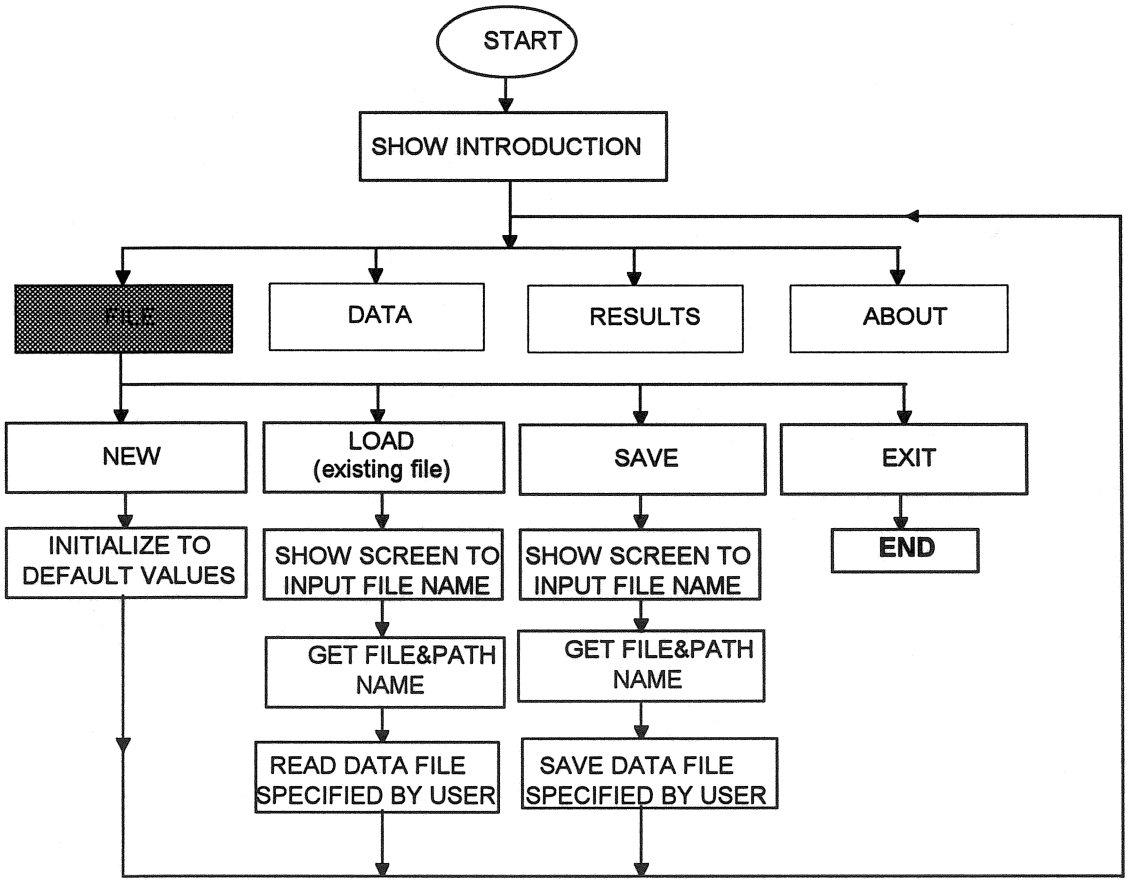


Figure 3-2 Block 1 -Input-Output data system (File operations flow chart).

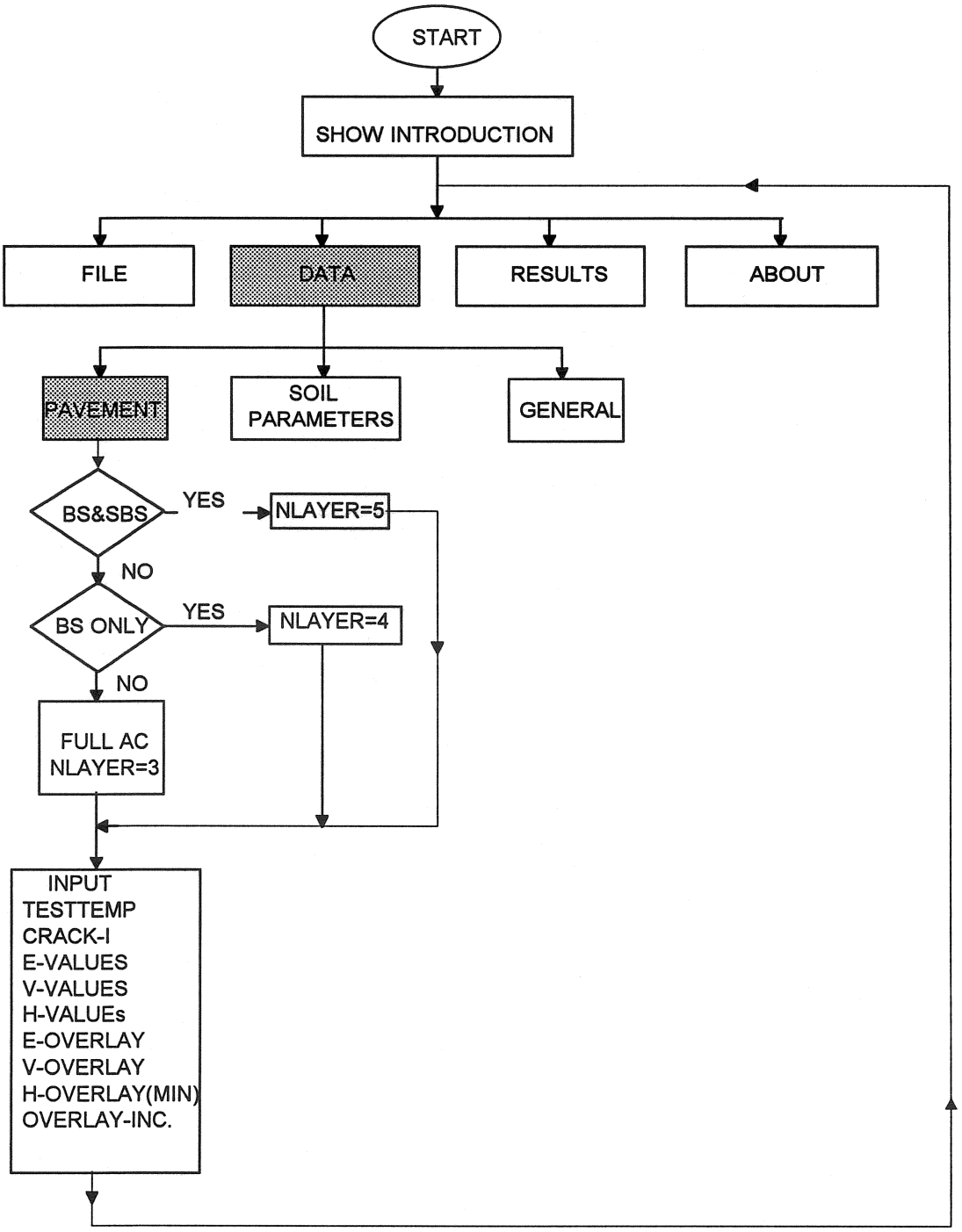


Figure 3-3 Block 1 (pavement data flow chart)

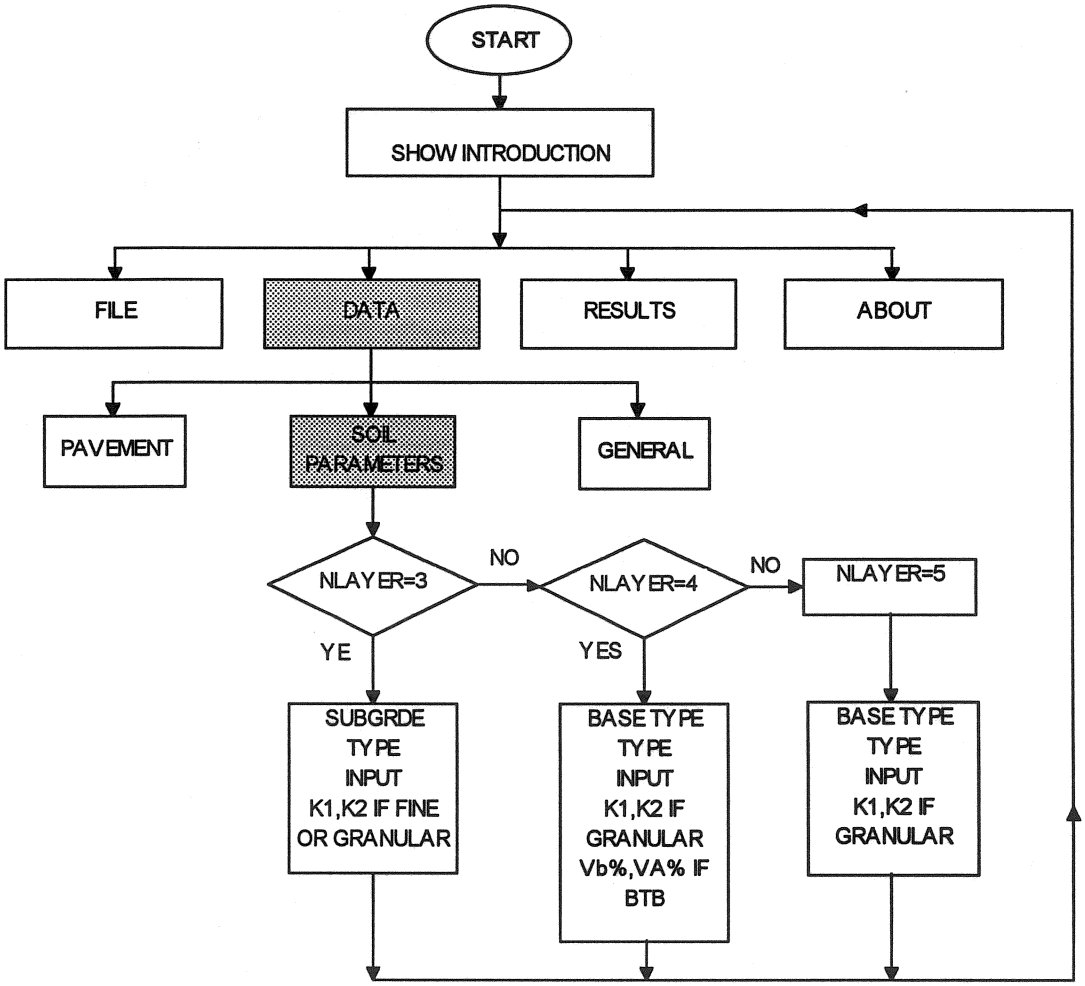


Figure 3-4 Block 1 (soil parameters data flow chart) .

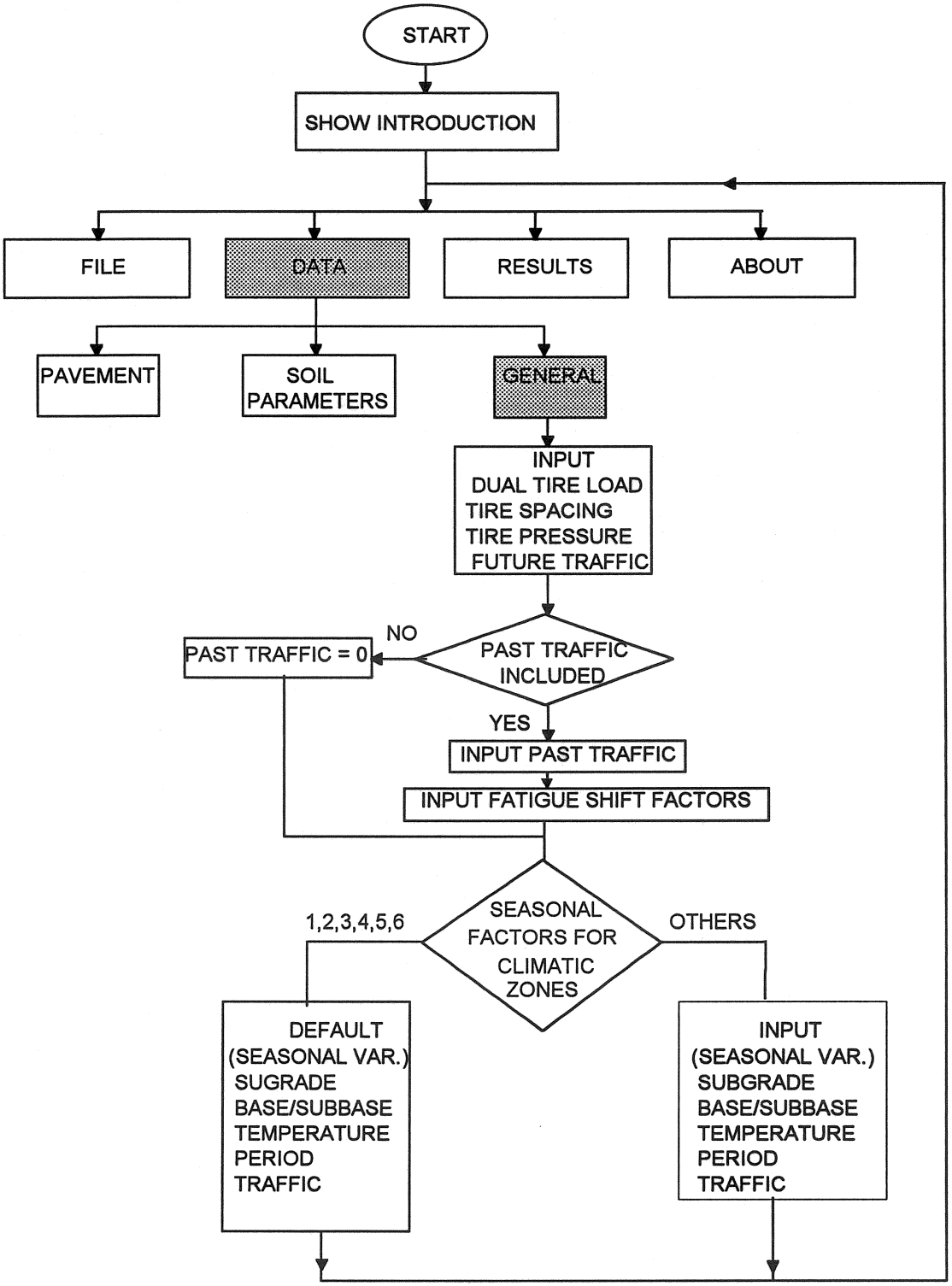


Figure 3-5 Block 1 (general data flow chart).

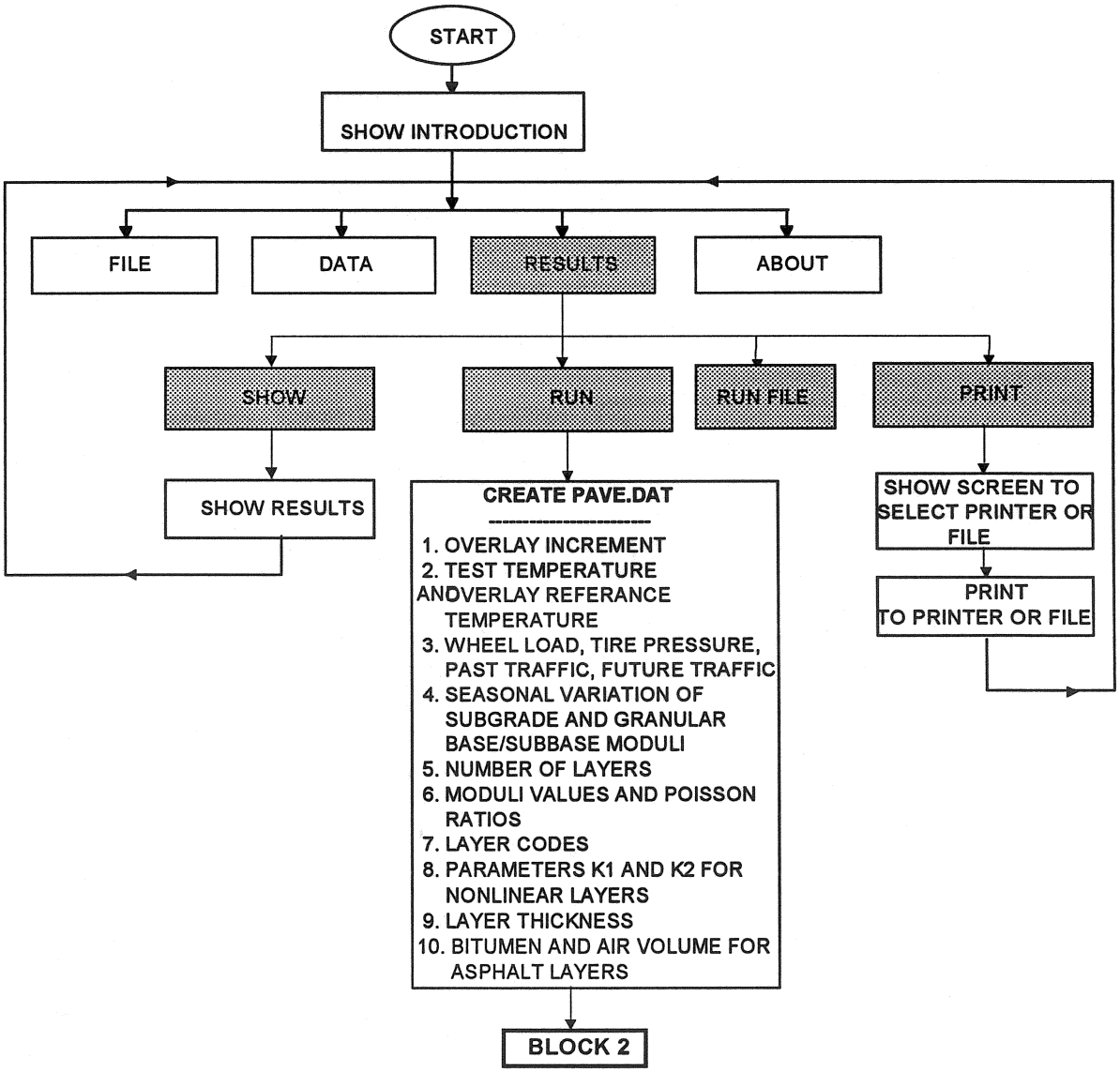


Figure 3-6 Block 1 : Results flow chart (for the RUN command).

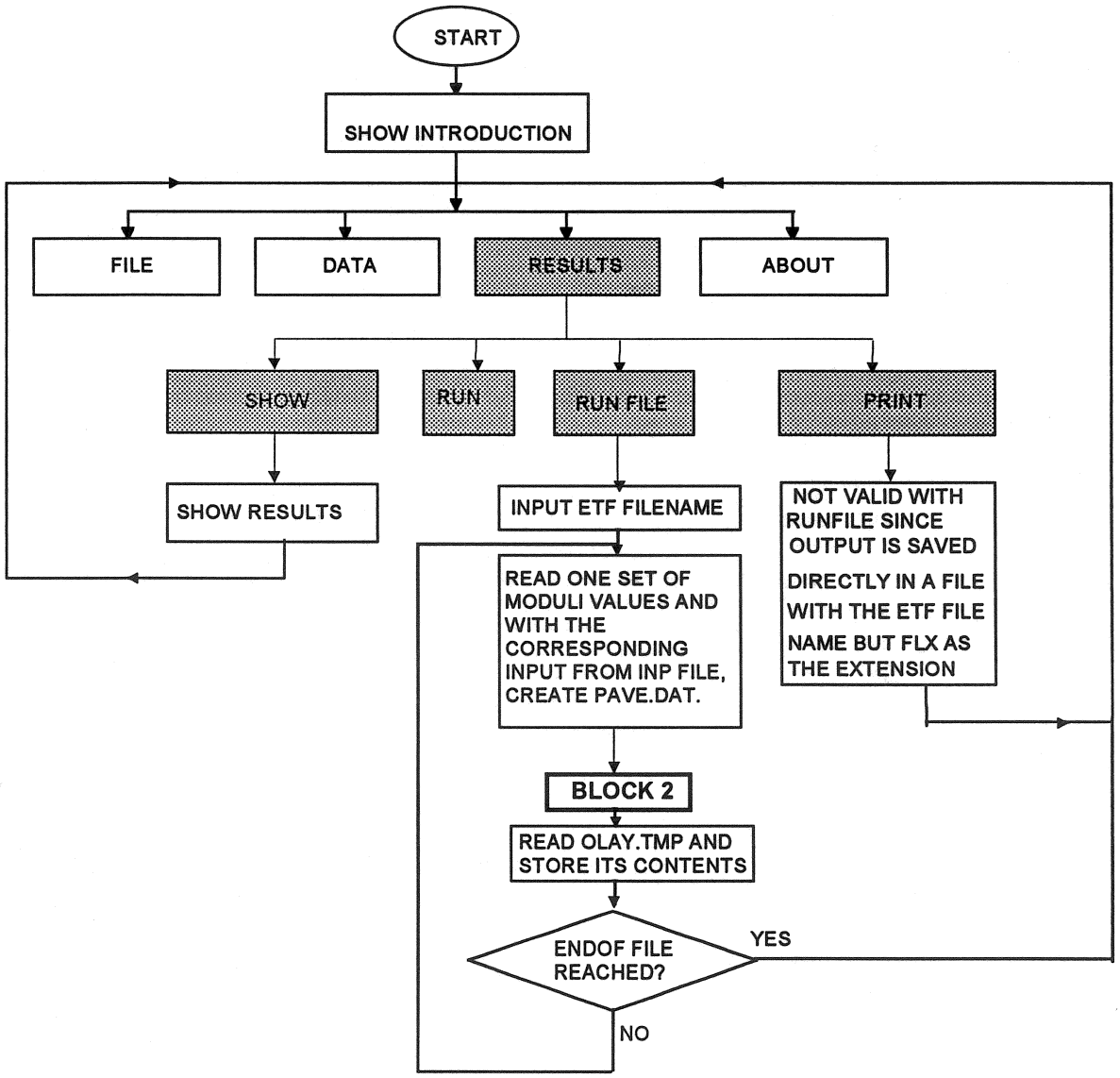


Figure 3-7 Block 1 Results flow chart (for the RUNFILE command).

Pavement data are handled by PAVE.FRM . The required input pavement data are :

1. Existing pavement option, i.e., includes base and subbase, base only or full depth asphalt.
2. Crack index of the existing pavement determined from the condition survey. According to Idaho Transportation department classification, the value should be from 0 for completely cracked pavement to 5 for a pavement with excellent condition.
3. Pavement temperature during FWD test in °F.
4. Moduli values of the existing layers in ksi as determined from backcalculation using FWD deflection data. The normal condition moduli values are required for the subgrade and the untreated base and subbase soils. If the FWD has been performed in a season where these layers are frozen, wet or experience thawing, the normal moduli values have to be determined from the backcalculated moduli values using Fig.2-2 through Fig.2-7 based on the pavement location (ZONE) and the season in which the FWD was performed. (Note : For non-linear layers, the modulus value to be entered is the initial minimum value, since the modulus of such layer is stress dependent and will be calculated based on the parameters K_1 and K_2).
5. Poisson's ratio of each layer. These values can be determined from laboratory testing. Because the Poisson ratio has a relatively small effect on pavement responses, it is customary to assume a reasonable value for use in the design, rather than to determine it from actual tests.
6. Thickness of the existing layers in inches, except the subgrade which is assumed as semi-infinite. Thickness of each layer can be accurately determined from cores taken from the existing pavement or can be approximated using the as built drawings.
7. Overlay modulus in ksi and the reference temperature in °F at which the modulus is determined. The overlay modulus may be determined from the lab using resilient modulus test or can be estimated from Fig.2-8.
8. Overlay minimum thickness and the increments allowed.

9. Initial percentage asphalt volume and air voids in the existing asphalt layer mix. These values may be difficult to obtain; Thus, values of 11% for the asphalt volume and 5% for air voids were set as default.
10. Percentage asphalt volume and air voids in the overlay asphalt mix. These were also set at a default value of 11% for the asphalt volume and 5% for air voids were set as default.

The soil parameters data are handled by MAT.FRM. The required input soil parameters are:

1. Type of subgrade , i.e., Linear, Fine or Granular. The subgrade is called linear when the resilient modulus is stress independent. The resilient modulus is assumed to follow Eq.2-2 for fine subgrade and Eq.2-1 if granular subgrade.
2. Subgrade Soil parameters K_1 and K_2 if subgrade is stress dependent, i.e., fine or granular. These parameters are determined from laboratory testing.
3. Type of base if included, i.e., linear, granular, cement treated or bitumen treated.
4. Base soil parameters, K_1 and K_2 if granular or percentage asphalt volume and air voids for bitumen treated base. These parameters are determined from laboratory testing.
5. Subbase type if included, i.e., linear or granular.
6. Subbase soil parameters, K_1 and K_2 if granular. These parameters are determined from laboratory testing.

The General data are handled by GENERAL.FRM. The required input general data are :

1. Wheel load in lb. (dual). If the traffic is expressed in terms of 18 kip ESAL, the dual tire load is entered as 4500 lb.
2. Dual tire spacing in inches (13.2 in is a typical value)
3. Tire pressure in psi . A value of 80 psi is recommended.
4. Estimated future traffic.
5. Past traffic if included.
6. Fatigue shift factor for the new AC.
7. Fatigue shift factor for the old AC.

8. Pavement operating climate ZONE, i.e., 1,2,3,4,5,6 or OTHERS.
9. Seasonal subgrade moduli adjustment factors, if the ZONE is chosen as OTHERS. If the user select a ZONE from 1 to 6, the adjustment factors are determined by the program from Fig.2-2 through Fig.2-6 based on the selected ZONE and the operating period (season). These figures were incorporated in the program by approximating the curves into a number of straight lines.
10. Seasonal base/subbase moduli adjustment factors, if the ZONE is chosen as OTHERS. If the user select a ZONE from 1 to 6, the adjustment factors are determined by the program from Fig.2-7.
11. Seasonal traffic variation. These are fractions required to adjust the traffic for each season. This requires traffic analysis and planing studies. Thus, for the time being, the seasonal traffic variations are entered as 1 for each season, which means that the traffic is evenly distributed throughout the year.
12. Seasonal temperatures, if the ZONE is chosen as OTHERS. If the user selects a ZONE from 1 to 6, the seasonal temperatures are determined by the program based on Table 2-2.
13. Seasonal periods, if the ZONE is chosen as OTHERS. If the user select a ZONE from 1 to 6, the seasonal periods are determined by the program based on Table 2-2.

BLOCK 2 : This block is the start of SYSAN.EXE FORTRAN program. It performs the initialization of the control variables and reads the data file, PAVE.DAT, which is created by FLEXOLAY.EXE as the user chooses the RUN command. Fig.3-8 illustrates the steps that take place in this block.

BLOCK 3 : This block performs the adjustment of moduli values of the various layers to account for seasonal variations. Fig.3-9 illustrates the steps that take place in this block. For each season, the resilient modulus of subgrade , untreated base and subbase soils is adjusted by multiplying the adjustment factor assigned to the season(general data - input No. 8) by the normal resilient modulus of the layer (pavement data- input No.4). If a nonlinear layer is present, the adjustment is performed by multiplying the adjustment factor by the parameter K_1 (soil parameter data input) based on what layer is considered as stress dependent.

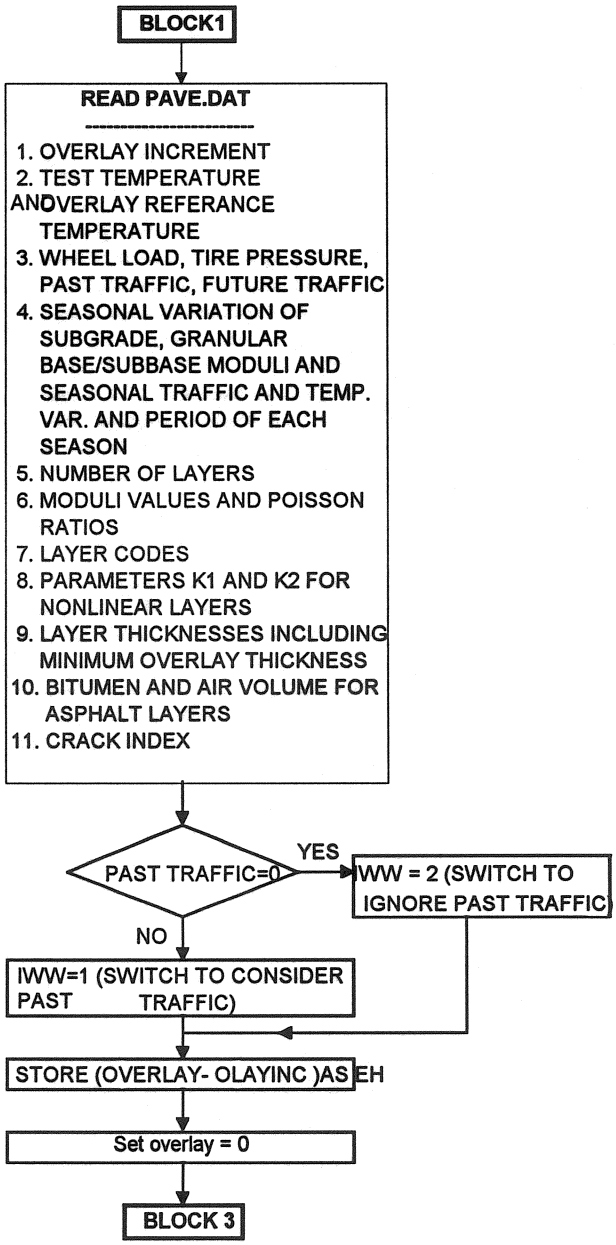


Figure 3-8 Block 2 flow chart (Initialization of control variables).

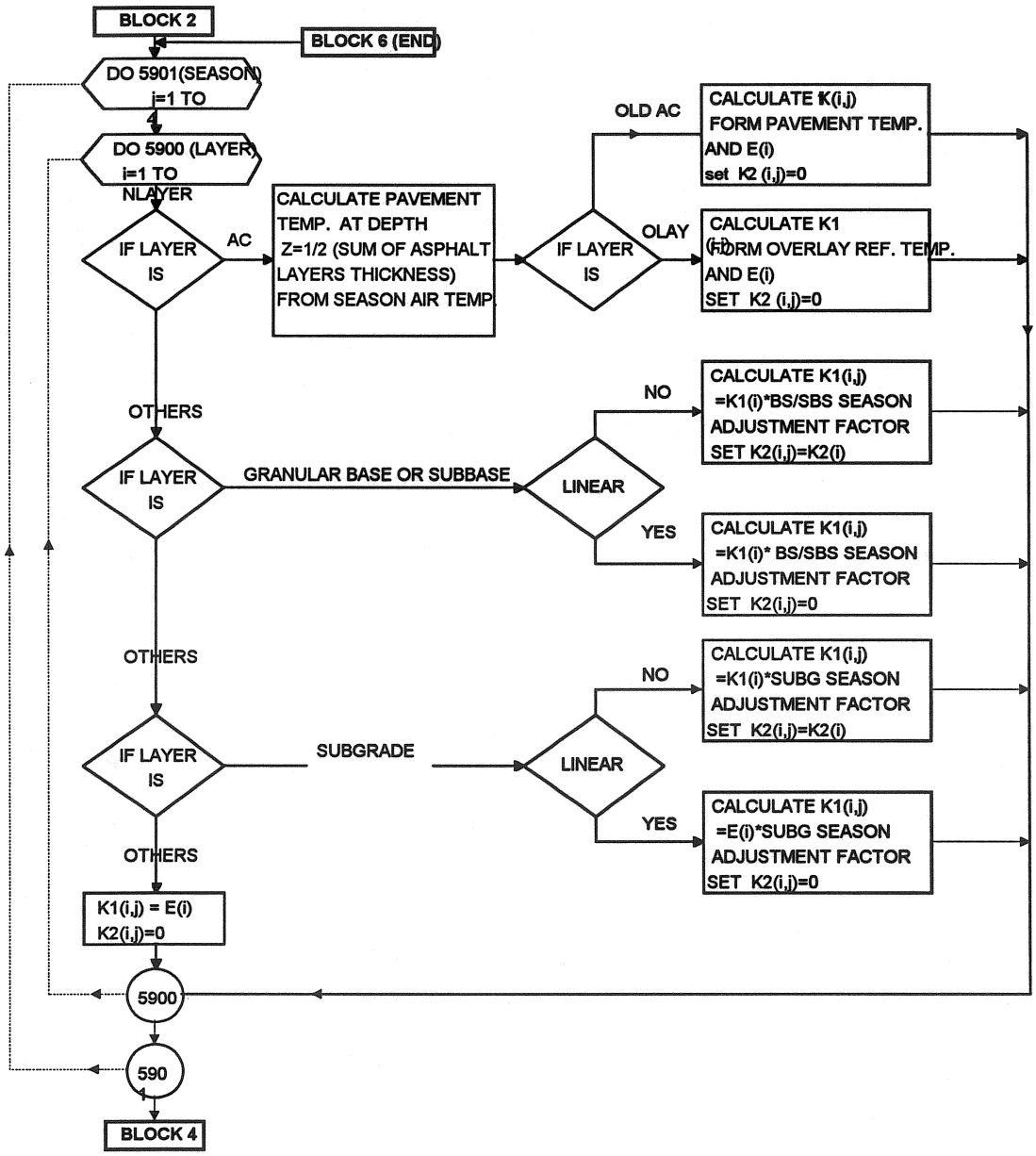


Figure 3-9 Block 3 flow chart (moduli adjustments).

The modulus of asphalt layers is adjusted based on the mean seasonal temperature. This is done as follows :

a) From the season temperature(general data - input no.9), the pavement representative temperature is determined at mid-depth of the asphalt layers using Eq.2-6.
b) The modulus of the asphalt layer (pavement data - input no.4) with the pavement temperature during FWD test (pavement data - input no.1) are plotted on Fig.2-8. If the layer is an overlay, the overlay modulus and the reference temperature (pavement data - input no.7) are plotted in the figure.
c) A curve is redrawn on Fig.2-8 so that it has the same slope as the original curve shown in the figure, but it runs through the point plotted in step b.
d) Using the new curve, the adjusted modulus can be determined from the representative seasonal pavement temperature determined in step a.

BLOCK 4 : This block performs the iterations to determine the modulus of non-linear layers. The program starts with the adjusted moduli values(From block 3). During each iteration, a constant set of moduli is computed based on the stresses obtained from the previous iteration using the multi-layer elastic routine and Eq.2-1 for granular nonlinear materials and Eq.2-2 for fine nonlinear materials. The process is repeated until the moduli converge to a 10% tolerance or the number of iterations reach 6. This process is repeated for each season. Degree of approximation is dependent on the number of iterations and tolerance. Fig.3-10 illustrates the steps that take place in this block. It is well known that most granular materials cannot take any tension. Unfortunately, when they are used as a base or subbase on a weaker subgrade, the horizontal stresses at the bottom of these materials are most likely in tension. To avoid this situation, the point to compute the modulus at is selected to be in upper quarter and upper third of the layer. Because the point is at the upper part of the layer, the chance of negative θ is rare. If θ turns out to be negative the adjusted initial E value is used.

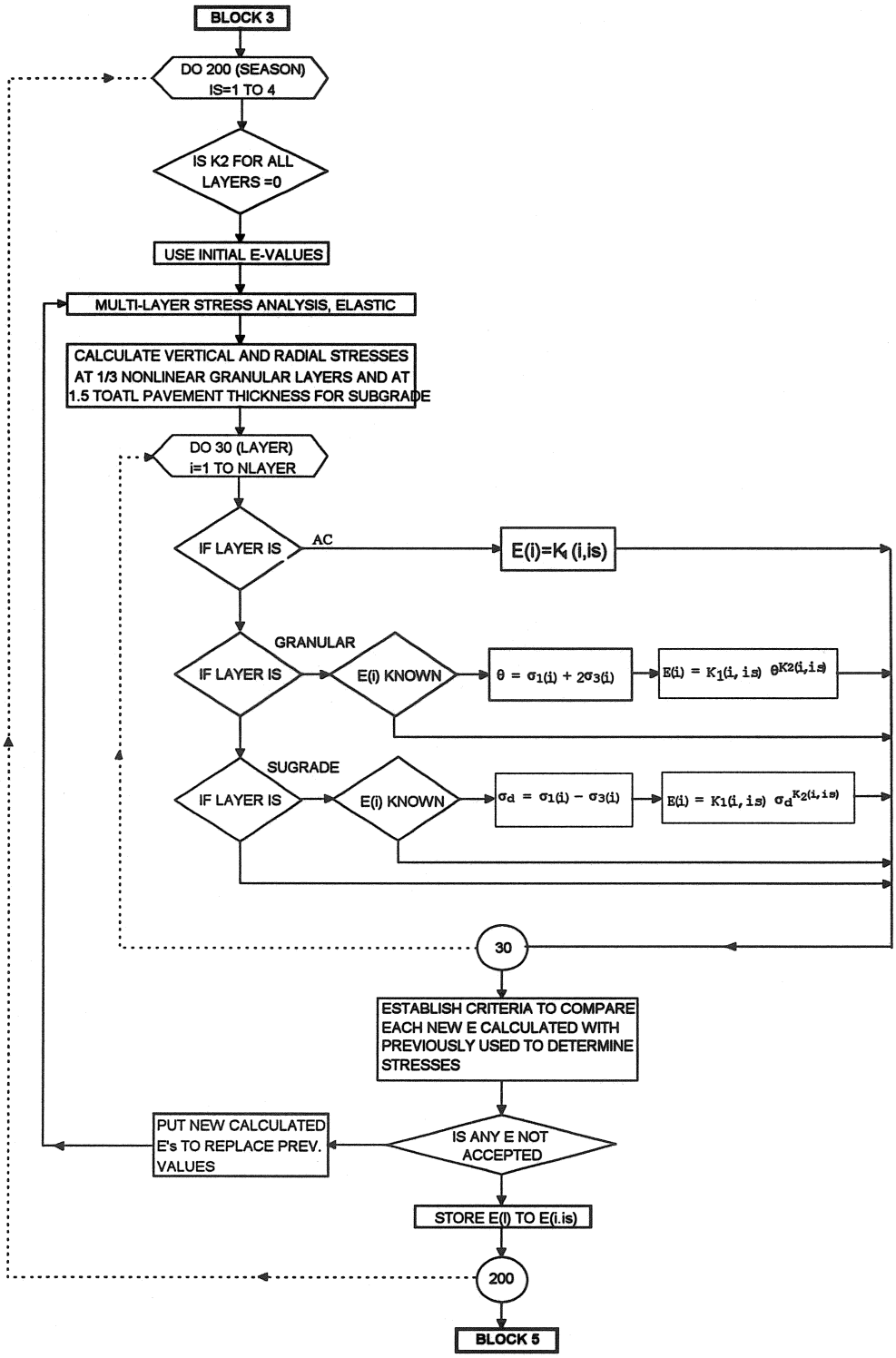


Figure 3-10 Block 4 flow chart (moduli values for non-linear layers).

BLOCK 5 : This block performs the calculations of the allowable fatigue and rutting life based on the tensile strain at the bottom of the asphalt layers (overlay or layer 1, existing asphalt layer or layer 2 and bitumen treated base or layer 3 if included) and the compressive strain at top of the subgrade (or lower layer). The calculations of the strains are performed for each season by the multi-layered elastic routine. The allowable life for fatigue and rutting for each season are calculated using the Asphalt Institute fatigue and rutting models discussed earlier. In block 5B, a special routine for thin overlays is presented. The function of this routine is to calculate the tensile strain at the bottom of thicker overlays and then extrapolated for the thin overlay. This is checked against tensile strain calculated for the thin overlay by the normal method and the larger of the two values is selected for the calculation of the allowable fatigue life for the overlay. This routine is an approximate way to reduce the errors that may result when using the multi-layer elastic routine with thin surface layers[18]. Fig.3-11 and Fig.3-12 illustrate the steps that take place in this block.

BLOCK 6 : this block represents the final step in the analysis. The sum of damage ratios due fatigue and rutting are checked as discussed in section 2-5. If the analysis prove that an overlay is required, the overlay is incremented and the iteration proceed to BLOCK 3, otherwise , the program stores the results in temporary files to be read when the user choose to view or print the results. Fig.3-13 illustrates the steps that take place in this block.

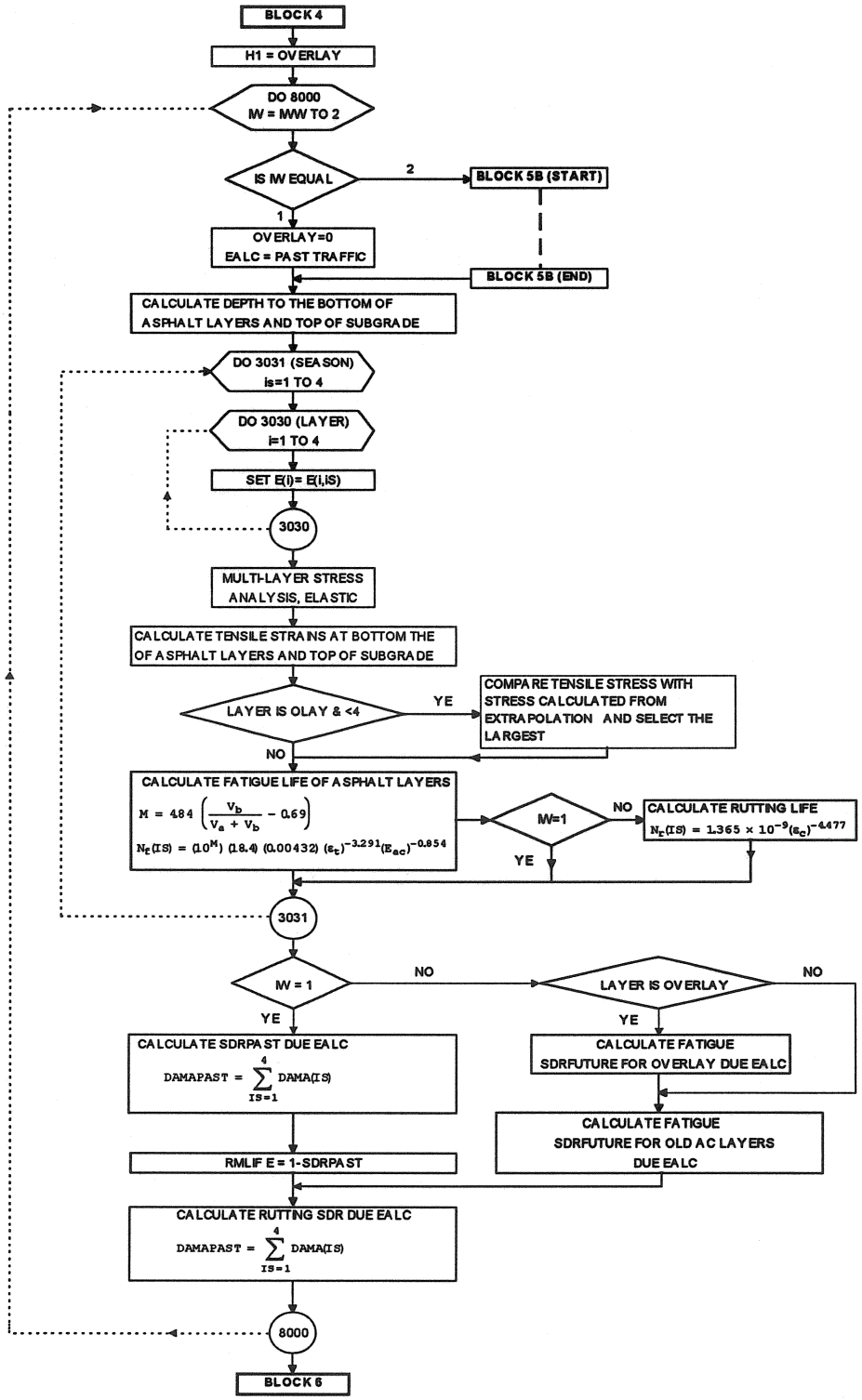


Figure 3-11 Block 5A flow chart (allowable fatigue and rutting life).

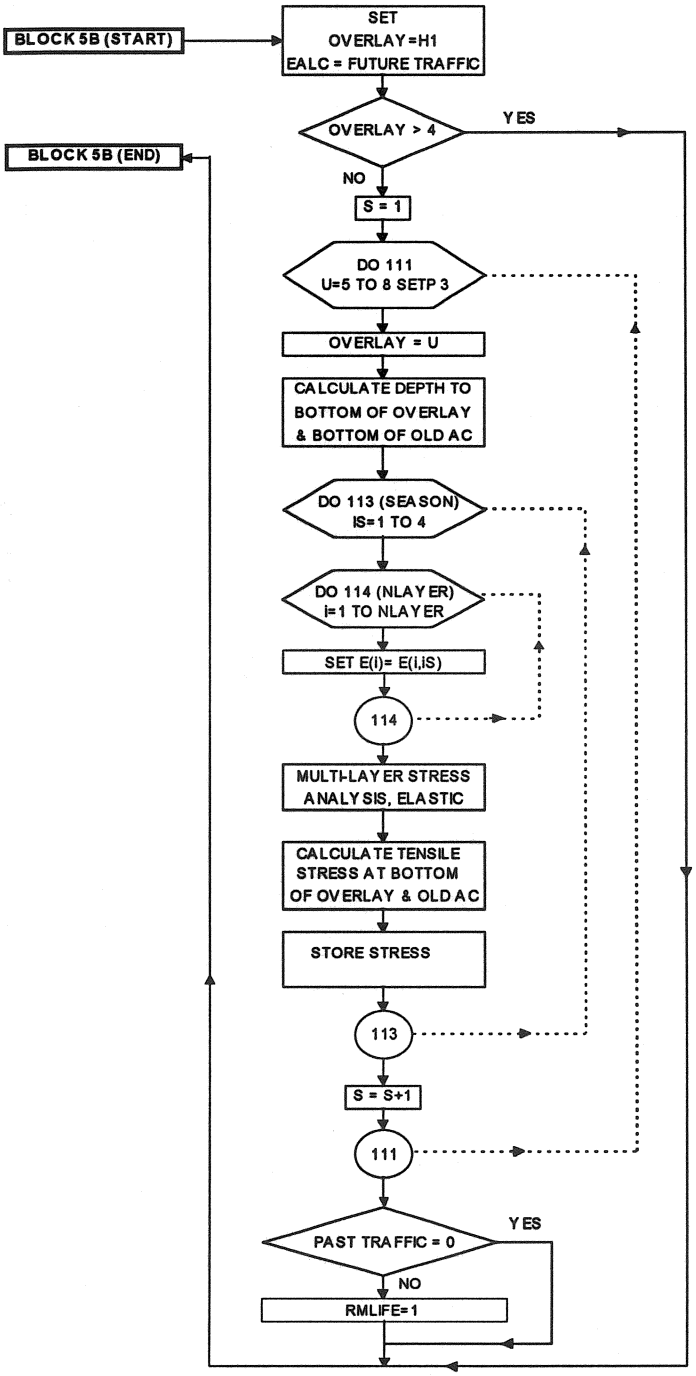


Figure 3-12 Block 5B (routine for thin layers).

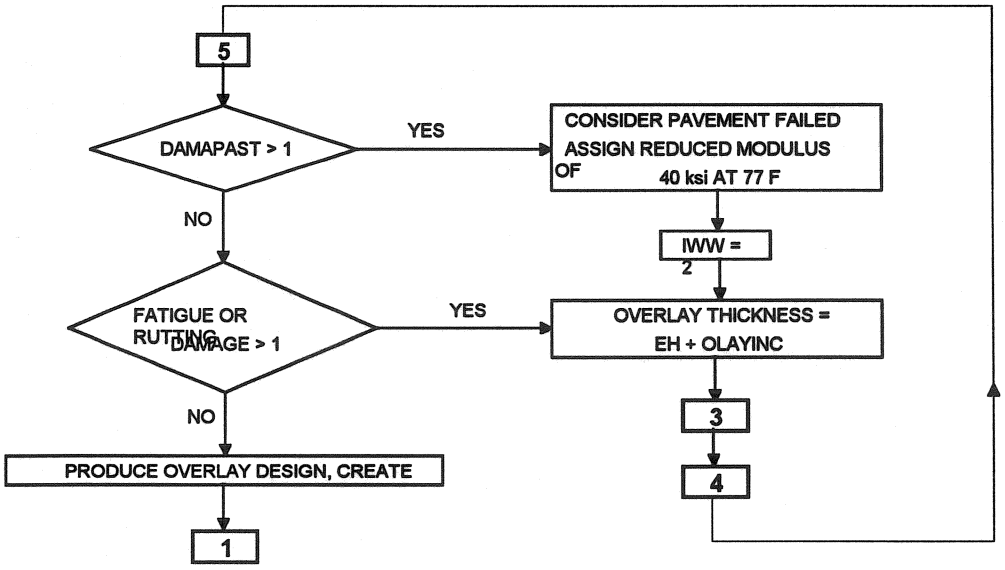


Figure 3-13 Block 6 flow chart (damage analysis).

REFRENCES

1) Finn, F., Monismith, C. ,"Asphalt Overlay Design Procedures", National Cooperative Highway Research Program (NCHRP) Report No. 116, Washington, DC, 1984.

2) Smith, R.E., Darter, M.I. and Laytton, R.L., "Mechanistic Overlay Design Procedure available to the Design Engineer", Transportation Research Board, National Research Council, Washington D.C., 1986.

3) Idaho Transportation Department Design Manual.

4) Bayomy, F.M. and Shah, Z.R., "Development of Recommendations and Guidelines for Pavement Rehabilitation Design Procedure for the State of Idaho; Phase 1 Software Evaluation and Data Analysis", ITD Project No. 92-35 Proj 112, Final Report, 1993.

5) Hardcastle, J. H. " Subgrade Resilient Modulus for Idaho Pavements", Final Report, Department of Civil Engineering, University of Idaho, Moscow, 1992.

6) AI, "Asphalt Overlays for Highway and Street Rehabilitation". Manual Series No. 17, Asphalt Institute, 1983.

7) AI, "Thickness Design-Asphalt Pavement for Highways And Streets", Manual Series No.1, Asphalt Institute, 1981a.

8) AI, "Research and Development of the Asphalt Institute Thickness Design Manual (MS-1), 9th ed., Research Report 82-2, Asphalt Institute.

9) AASHTTO, 'Guide for Design of Pavement Structures", American Association of State Highway and Transportation Officials", 1993.

10) Hall, T. K., Darter, M. I. And Elliott, R.P. " ROADHOG- A Flexible Pavement Overlay Design Procedure", Transportation Research Record 1374, Transportation Research Board, Washingto D.C., 1992.

11) California Department Of Transportation, "Asphalt Concrete Overlay Design Manual, 1979.

12) SHELL, "Shell Pavement Design Manual - Asphalt pavements and Overlays for Road Traffic", Shell International Petroleum, London, 1987.

13) Majidzadeh, K., Ilves, G. J. and May, R. W. "Overlay Design of Flexible Pavements: OAF Program", Transportation Research Record 930, Transportation Research Board, Washington D.C., pp.18-25.

14) M.R. Thomson, "Concepts for Developing a NDT-Based Design Procedure for Determining Asphalt Concrete Overlay Thickness", Transportation Research Record 930, Transportation Research Board, Washington D.C., pp. 12-18.

15) Mamlouk, S. M., Zaniewski, J. P., Houston, W. N. and Houston, S. L. " Overlay Design Method for Flexible Pavements in Arizona", Transportation Research Record 1286, Transportation Research Board, Washington D.C., pp. 112 - 122.

16) Pierce, L. M., Jackson, N. C. and Mahoney, J. P. " Development and Implementation of a Mechanistic, Empirically Based Overlay Design Procedure for Flexible Pavements", Transportation Research Record 1388, Transportation Research Board, Washington, D.C., pp.120 - 128.

17) Monsmith, C.L. and Epps, J.A.,"Asphalt Mixture Behavior in Repeated Flexure", Institute of Transportation and Traffic Engineering, University of California, Berkely, 1969.

18) Sebaaly, P. E., Siddharthan, R., Schoener, P.," Development of Nevada's Overlay Design Procedure", Transportation Research Record, Transportation Research Board, Washington D.C., 1994.

19) Burmister, D.M.,"Theory of Stress and Displacement in Layered System and Application to the Design of Airport Runways", HRB Proceedings, Washington D.C., 1943.

20) Huang, Yang H., "Pavement Analysis and Design", Printice Hall, Englewood Cliffs, New Jersey, 1993.

21) Lee, S.W.,"Backcalculation of Pavement Moduli By Use of Pavement Surface Deflections", Ph.D. Dissertation, Department of Civil Engineering, University of Washington, Seattle, 1988.

22) Rohde, G.T.,Scullion, T.,"Expansion and Validaton of the Modulus Backcalculation System", Texas Transportation Institute Report 1123-3, November, 1990.

23) Seed, H.B., Mity, F.G., Monosmith, C.L. and Chan, C.K."Factors Influencing the Resilient Deformation of Untreadted Aggregate Base In Two-Layer Pavements Subjected to Repeated Loading", Highway Transportation Record 190, Highway Research Board, Washington D.C., 1967, PP. 19-57.

24) Seed, H.B., Chan, C.K. and Lee, C.E. "Resilient Characteristics of Subgrade Soils and Their Relation to Fatigue Failures in Asphalt Pavements", Proceedings, International Conference on The structural Design of Asphalt Pavements, University of Michegan, 1962, pp.611-636.

25) Rutherford, M.S. "Pavement Response and Load Restrictions on Spring-Thaw Weakened Pavements", Transportation Research Record 1252, Transportation Research Board, Wasington D.C., 1989, pp. 1-11.

26) Witczak, M. W., "Design of Full Depth Asphalt Airfield Pavements", Proceedings, Vol.1, Third International Conference on the Structural Design of Asphalt Pavements, London, England, 1972, pp. 550-567.

27) Rada, G.R., Richter, C.A, Stephanos, P.J., "Layer Moduli From Deflection Measurments:Software Selection and Development of SHARP's Procedure for Flexible Pavements", Beltsvile, MD, 1991.

28) Finn, F., Sarsf, C.L., Kulkarni, R., Nair, K., Smith, W. And Abdullah, A., "Development of pavement structural subsystems", National Cooperative High Research program (NCHRP) No. 291, National Research Council, Washington D.C.

29) J. Michelos, "Analysis of Stress and Displacements in N-layered Elastic System Under a Load Uniformly Distributed on a Cicular Area", California Research Corporation, Richmond, California, 1963.

30) Miner, M. A.,"Comulative Damage in Fatigue", Transaction of the ASME, Vol 67, 1945, PP. A159-A164.

FLEXOLAY

Program Listing

FLEXOLAY.BAS

```
'********* 1- STATRTING MODULE (START.BAS)*************
'-----------------------------------------------
'$FORM Main
Main.SHOW

DECLARE SUB DrawAboutPicture ()

'$FORM frmCmnDlg
DIM SHARED E(5), V(5), HH(5), OPTIONS(3), KSUG(2), kbase(2), KSBASE(2), ZONES(6),
SUBGRD(4)
DIM SHARED WPSI(7), SUGV(4), BASEV(4), TRAFICV(4), TEMPV(4), periodv(4), BITAIR(2),
MT(5), CODE(5), OLDBAIR(2)
COMMON E(), V(), HH(), WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), crack,
TXTCHANG, ZONES(), SUBGRD()
COMMON OPTIONS(), KSUG(), kbase(), KSBASE(), BITAIR(), CODE(), OLDBAIR(), MT(),
NLAYER, OLAYINC, TESTTEMP, OLAYTEMP
COMMON DSCRPTION$, A, B, AA, BB, K, Y, Z, R, T, G, ZON, SHOWPRN, PASTCHCK, OLAYT,
DAMA1, DAMA2, DAMA3, DAMA4, DAMA22
CONST FALSE = 0
CONST TRUE = NOT FALSE


SUB About (AboutText AS STRING, ForeColor AS INTEGER, BackColor AS INTEGER, Flags AS
INTEGER)
   ON LOCAL ERROR GOTO AboutError

   frmCmnDlg.Caption = "ABOUT FLEXOLAY 1.2 PROGRAM"        ' Set form caption.

   ' Set dialog colors.
   frmCmnDlg.ForeColor = ForeColor
   frmCmnDlg.BackColor = BackColor
   frmCmnDlg.pctAbout.ForeColor = ForeColor
   frmCmnDlg.pctAbout.BackColor = BackColor
   frmCmnDlg.pctAboutPict.ForeColor = ForeColor
   frmCmnDlg.pctAboutPict.BackColor = BackColor
   frmCmnDlg.lblAboutText.ForeColor = ForeColor
   frmCmnDlg.lblAboutText.BackColor = BackColor
   frmCmnDlg.cmdAboutOK.BackColor = BackColor

   ' Determine if picture should be displayed.
   IF Flags = 1 THEN
      CALL DrawAboutPicture          ' Routine that draws picture.
      PictWidth% = frmCmnDlg.pctAboutPict.Width   ' Get Width and Height of picture for
      PictHeight% = frmCmnDlg.pctAboutPict.Height ' determining size of dialog.
   ELSE
      frmCmnDlg.pctAboutPict.visible = FALSE      ' Make picture visible.
      PictWidth% = 0
      PictHeight% = 0
   END IF
```

```
   ' Size and position label correctly for text display.
   frmCmnDlg.lblAboutText.Caption = AboutText
   frmCmnDlg.lblAboutText.MOVE frmCmnDlg.pctAboutPict.Left + PictWidth% + 3,
frmCmnDlg.lblAboutText.Top, frmCmnDlg.TEXTWIDTH(frmCmnDlg.lblAboutText.Caption),
frmCmnDlg.TEXTHEIGHT(frmCmnDlg.lblAboutText.Caption)
   LabelWidth% = frmCmnDlg.lblAboutText.Width      ' Get Width and Height of text for
   LabelHeight% = frmCmnDlg.lblAboutText.Height    ' determining size of dialog.

   ' Size and position About container.
   frmCmnDlg.pctAbout.BorderStyle = 0
   frmCmnDlg.pctAbout.visible = TRUE
   frmCmnDlg.pctAbout.Width = PictWidth% + LabelWidth% + 8
   IF LabelHeight% > PictHeight% THEN
      frmCmnDlg.pctAbout.Height = LabelHeight% + 6
   ELSE
      frmCmnDlg.pctAbout.Height = PictHeight% + 5
   END IF

   ' Center command button at the bottom of the dialog.
   frmCmnDlg.cmdAboutOK.MOVE (frmCmnDlg.pctAbout.ScaleWidth -
frmCmnDlg.cmdAboutOK.Width) \ 2, frmCmnDlg.pctAbout.ScaleHeight - 3
   frmCmnDlg.cmdAboutOK.Default = TRUE
   frmCmnDlg.cmdAboutOK.Cancel = TRUE

   ' Size and center dialog.
   frmCmnDlg.MOVE frmCmnDlg.Left, frmCmnDlg.Top, frmCmnDlg.pctAbout.Width + 2,
frmCmnDlg.pctAbout.Height + 2
   frmCmnDlg.MOVE (SCREEN.Width - frmCmnDlg.Width) \ 2, ((SCREEN.Height -
frmCmnDlg.Height) \ 2) - 2

   ' Display dialog modally.
   frmCmnDlg.SHOW 1

   ' Hide or unload dialog and return control to user's program.
   ' (Hide if user chose to preload form for performance.)
   IF LEFT$(frmCmnDlg.Tag, 1) = "H" THEN
      frmCmnDlg.pctAbout.visible = FALSE
      frmCmnDlg.HIDE
   ELSE
      UNLOAD frmCmnDlg
   END IF

   EXIT SUB

' Error handling routine.
AboutError:
   SELECT CASE ERR
   CASE 7:                          ' Out of memory.
       MSGBOX "Out of memory.  Can't load dialog.", 0, "About"
       EXIT SUB
   CASE ELSE
       RESUME NEXT
   END SELECT
END SUB
```

```
SUB CmnDlgClose ()
  UNLOAD frmCmnDlg          ' Unload form.
END SUB

SUB CmnDlgRegister (Success AS INTEGER)
  ' Set up error handling.
  ON LOCAL ERROR GOTO RegisterError

  LOAD frmCmnDlg            ' Load form.
  frmCmnDlg.Tag = "H"       ' Set flag for keeping form loaded after
                            ' each common dialog usage.

  Success = TRUE
  EXIT SUB

RegisterError:
  SELECT CASE ERR
  CASE 7:                           ' Out of memory.
     MSGBOX "Out of memory.  Can't load Common Dialogs.", 0, "Common Dialog"
     Success = FALSE
     EXIT SUB
  CASE ELSE
     MSGBOX ERROR$ + ".  Can't load Common Dialogs.", 0, "Common Dialog"
     Success = FALSE
     EXIT SUB
  END SELECT
END SUB

'
SUB DrawAboutPicture ()
  frmCmnDlg.pctAboutPict.Height = 8          ' Set picture height.
  frmCmnDlg.pctAboutPict.Width = 15          ' Set picture width.
  frmCmnDlg.pctAboutPict.BorderStyle = 1     ' Set border style.
  frmCmnDlg.pctAboutPict.visible = TRUE      ' Make picture visible.

  ' Display picture.
  frmCmnDlg.pctAboutPict.PRINT "  ITD-NCATT  "
  frmCmnDlg.pctAboutPict.PRINT "   PROJECT   "
  frmCmnDlg.pctAboutPict.PRINT "     on      "
  frmCmnDlg.pctAboutPict.PRINT "   Overlay   "
  frmCmnDlg.pctAboutPict.PRINT "Design System"
  frmCmnDlg.pctAboutPict.PRINT "   June 96   "
END SUB

SUB FileOpen (FileName AS STRING, PathName AS STRING, DefaultExt AS STRING, DialogTitle AS
STRING, ForeColor AS INTEGER, BackColor AS INTEGER, Flags AS INTEGER, Cancel AS
INTEGER)
  ' Set up error handling for option validation.
  ON LOCAL ERROR GOTO FileOpenError

  ' Set form caption.
  IF DialogTitle = "" THEN
    frmCmnDlg.Caption = "Open"
```

```
  ELSE
    frmCmnDlg.Caption = DialogTitle
  END IF

  ' Determine search pattern for file listbox.
  IF DefaultExt <> "" THEN
    frmCmnDlg.filOpenList.Pattern = DefaultExt
  ELSE
    frmCmnDlg.filOpenList.Pattern = "*.*"
  END IF

  ' Determine default path.
  IF PathName <> "" THEN
    ' Set drive and path for file-system controls.
    ' Set Directory listbox path.  If PathName is different
    ' than current path, PathChange event will be triggered
    ' which updates Drive listbox drive and File listbox path.
    frmCmnDlg.dirOpenList.Path = PathName
  END IF
  ' Display current path to the user.
  frmCmnDlg.lblOpenPath.Caption = frmCmnDlg.filOpenList.Path

  ' Determine default filename to display in edit field.
  IF FileName <> "" THEN
    frmCmnDlg.txtOpenFile.Text = UCASE$(FileName)
  ELSE
    frmCmnDlg.txtOpenFile.Text = frmCmnDlg.filOpenList.Pattern
  END IF

  ' Set default and cancel command buttons.
  frmCmnDlg.cmdOpenOK.Default = TRUE
  frmCmnDlg.cmdOpenCancel.Cancel = TRUE

  ' Size and position Open/Save container.
  frmCmnDlg.pctFileOpen.BorderStyle = 0
  frmCmnDlg.pctFileOpen.visible = TRUE

  ' Size and center dialog.
  frmCmnDlg.MOVE frmCmnDlg.Left, frmCmnDlg.Top, frmCmnDlg.pctFileOpen.Width + 2,
frmCmnDlg.pctFileOpen.Height + 2
  frmCmnDlg.MOVE (SCREEN.Width - frmCmnDlg.Width) \ 2, ((SCREEN.Height -
frmCmnDlg.Height) \ 2) - 2

  ' Set dialog colors.
  frmCmnDlg.ForeColor = ForeColor
  frmCmnDlg.BackColor = BackColor
  frmCmnDlg.pctFileOpen.ForeColor = ForeColor
  frmCmnDlg.pctFileOpen.BackColor = BackColor
  frmCmnDlg.lblOpenFile.ForeColor = ForeColor
  frmCmnDlg.lblOpenFile.BackColor = BackColor
  frmCmnDlg.txtOpenFile.ForeColor = ForeColor
  frmCmnDlg.txtOpenFile.BackColor = BackColor
  frmCmnDlg.lblOpenPath.ForeColor = ForeColor
  frmCmnDlg.lblOpenPath.BackColor = BackColor
```

```
  frmCmnDlg.filOpenList.ForeColor = ForeColor
  frmCmnDlg.filOpenList.BackColor = BackColor
  frmCmnDlg.drvOpenList.ForeColor = ForeColor
  frmCmnDlg.drvOpenList.BackColor = BackColor
  frmCmnDlg.dirOpenList.ForeColor = ForeColor
  frmCmnDlg.dirOpenList.BackColor = BackColor
  frmCmnDlg.cmdOpenOK.BackColor = BackColor
  frmCmnDlg.cmdOpenCancel.BackColor = BackColor

  ' Display dialog modally.
  frmCmnDlg.SHOW 1

  ' Determine if user canceled dialog.
  IF frmCmnDlg.cmdOpenCancel.Tag <> "FALSE" THEN
     Cancel = TRUE
  ' If not, return FileName and PathName.
  ELSE
     Cancel = FALSE
     FileName = frmCmnDlg.txtOpenFile.Text
     PathName = frmCmnDlg.filOpenList.Path
     frmCmnDlg.cmdOpenCancel.Tag = ""
  END IF

  ' Hide or unload dialog and return control to user's program.
  ' (Hide if user chose to preload form for performance.)
  IF LEFT$(frmCmnDlg.Tag, 1) = "H" THEN
     frmCmnDlg.pctFileOpen.visible = FALSE
     frmCmnDlg.HIDE
  ELSE
     UNLOAD frmCmnDlg
  END IF

  EXIT SUB

' Option error handling routine.
' Ignore errors here and let dialog's controls
' handle the errors.
FileOpenError:
  SELECT CASE ERR
  CASE 7:                          ' Out of memory.
      MSGBOX "Out of memory.  Can't load dialog.", 0, "FileOpen"
      Cancel = TRUE
      EXIT SUB
  CASE ELSE
      RESUME NEXT
  END SELECT
END SUB

SUB FilePrint (Copies AS INTEGER, ForeColor AS INTEGER, BackColor AS INTEGER, Cancel AS
INTEGER)
  ON LOCAL ERROR GOTO FilePrintError

  frmCmnDlg.Caption = "Print"        ' Set form caption.
```

```
  ' Determine default number of copies.
  IF Copies = 0 THEN
    frmCmnDlg.txtPrintCopies.Text = "1"
  ELSE
    frmCmnDlg.txtPrintCopies.Text = STR$(Copies)
  END IF

  ' Set default and cancel command buttons.
  frmCmnDlg.cmdPrintOK.Default = TRUE
  frmCmnDlg.cmdPrintCancel.Cancel = TRUE

  ' Size and position Print container.
  frmCmnDlg.pctFilePrint.BorderStyle = 0
  frmCmnDlg.pctFilePrint.visible = TRUE

  ' Size and center dialog.
  frmCmnDlg.MOVE frmCmnDlg.Left, frmCmnDlg.Top, frmCmnDlg.pctFilePrint.Width + 2,
frmCmnDlg.pctFilePrint.Height + 2
  frmCmnDlg.MOVE (SCREEN.Width - frmCmnDlg.Width) \ 2, ((SCREEN.Height -
frmCmnDlg.Height) \ 2) - 2

  ' Set dialog colors.
  frmCmnDlg.ForeColor = ForeColor
  frmCmnDlg.BackColor = BackColor
  frmCmnDlg.pctFilePrint.ForeColor = ForeColor
  frmCmnDlg.pctFilePrint.BackColor = BackColor
  frmCmnDlg.lblPrintCopies.ForeColor = ForeColor
  frmCmnDlg.lblPrintCopies.BackColor = BackColor
  frmCmnDlg.txtPrintCopies.ForeColor = ForeColor
  frmCmnDlg.txtPrintCopies.BackColor = BackColor
  frmCmnDlg.txtPrintFile.ForeColor = ForeColor
  frmCmnDlg.txtPrintFile.BackColor = BackColor
  frmCmnDlg.fraPrintTarget.ForeColor = ForeColor
  frmCmnDlg.fraPrintTarget.BackColor = BackColor
  FOR i% = 0 TO 3
    frmCmnDlg.optPrintTarget(i%).ForeColor = ForeColor
    frmCmnDlg.optPrintTarget(i%).BackColor = BackColor
  NEXT i%
  FOR i% = 0 TO 1
    frmCmnDlg.optPrintAppend(i%).ForeColor = ForeColor
    frmCmnDlg.optPrintAppend(i%).BackColor = BackColor
  NEXT i%
  frmCmnDlg.lblPrintAppend.ForeColor = ForeColor
  frmCmnDlg.lblPrintAppend.BackColor = BackColor
  frmCmnDlg.cmdPrintOK.BackColor = BackColor
  frmCmnDlg.cmdPrintCancel.BackColor = BackColor

  ' Display dialog modally.
  frmCmnDlg.SHOW 1

  ' Determine if user canceled dialog.
  IF frmCmnDlg.cmdPrintCancel.Tag <> "FALSE" THEN
    Cancel = TRUE
  ' If not, return number of copies to print.
```

```
  ELSE
    Cancel = FALSE
    IF VAL(frmCmnDlg.txtPrintCopies.Text) > 99 THEN
      Copies = 99
    ELSEIF VAL(frmCmnDlg.txtPrintCopies.Text) < 1 THEN
      Copies = 1
    ELSE
      Copies = VAL(frmCmnDlg.txtPrintCopies.Text)
    END IF
    frmCmnDlg.cmdPrintCancel.Tag = ""
  END IF

  ' Hide or unload dialog and return control to user's program.
  ' (Hide if user chose to preload form for performance.)
  IF LEFT$(frmCmnDlg.Tag, 1) = "H" THEN
    frmCmnDlg.pctFilePrint.visible = FALSE
    frmCmnDlg.HIDE
  ELSE
    UNLOAD frmCmnDlg
  END IF

  EXIT SUB

' Error handling routine.
FilePrintError:
  SELECT CASE ERR
  CASE 7:                          ' Out of memory.
      MSGBOX "Out of memory.  Can't load dialog.", 0, "FindPrint"
      Cancel = TRUE
      EXIT SUB
  CASE ELSE
      RESUME NEXT
  END SELECT
END SUB

SUB FileSave (FileName AS STRING, PathName AS STRING, DefaultExt AS STRING, DialogTitle AS
STRING, ForeColor AS INTEGER, BackColor AS INTEGER, Flags AS INTEGER, Cancel AS
INTEGER)
  ' Set up error handling for option validation.
  ON LOCAL ERROR GOTO FileSaveError

  ' Set form caption.
  IF DialogTitle = "" THEN
    frmCmnDlg.Caption = "Save As"
  ELSE
    frmCmnDlg.Caption = DialogTitle
  END IF
  frmCmnDlg.Tag = frmCmnDlg.Tag + "SAVE"          ' Set form tag for common unload procedure.

  ' Determine search pattern for file listbox.
  IF DefaultExt <> "" THEN
    frmCmnDlg.filOpenList.Pattern = DefaultExt
  ELSE
    frmCmnDlg.filOpenList.Pattern = "*.*"
```

```
END IF

' Determine default path.
IF PathName <> "" THEN
   ' If the path ends with a backslash, remove it.
   IF RIGHT$(PathName, 1) = "\" THEN
      PathName = LEFT$(PathName, LEN(PathName) - 1)
   END IF
   ' Set drive and path for file-system controls.

   ' Set File listbox path.  If PathName is different
   ' than current path, PathChange event will be triggered
   ' which updates Drive listbox drive and Directory listbox path.
   frmCmnDlg.filOpenList.Path = PathName
END IF
' Display current path to the user.
frmCmnDlg.lblOpenPath.Caption = frmCmnDlg.filOpenList.Path

' Determine default filename to display in edit field.
IF FileName <> "" THEN
   frmCmnDlg.txtOpenFile.Text = UCASE$(FileName)
ELSE
   frmCmnDlg.txtOpenFile.Text = frmCmnDlg.filOpenList.Pattern
END IF

' Set default and cancel command buttons.
frmCmnDlg.cmdOpenOK.Default = TRUE
frmCmnDlg.cmdOpenCancel.Cancel = TRUE

' Size and position Open/Save container.
frmCmnDlg.pctFileOpen.BorderStyle = 0
frmCmnDlg.pctFileOpen.visible = TRUE

' Size and center dialog.
frmCmnDlg.MOVE frmCmnDlg.Left, frmCmnDlg.Top, frmCmnDlg.pctFileOpen.Width + 2,
frmCmnDlg.pctFileOpen.Height + 2
frmCmnDlg.MOVE (SCREEN.Width - frmCmnDlg.Width) \ 2, ((SCREEN.Height -
frmCmnDlg.Height) \ 2) - 2

' Set dialog colors.
frmCmnDlg.ForeColor = ForeColor
frmCmnDlg.BackColor = BackColor
frmCmnDlg.pctFileOpen.ForeColor = ForeColor
frmCmnDlg.pctFileOpen.BackColor = BackColor
frmCmnDlg.lblOpenFile.ForeColor = ForeColor
frmCmnDlg.lblOpenFile.BackColor = BackColor
frmCmnDlg.txtOpenFile.ForeColor = ForeColor
frmCmnDlg.txtOpenFile.BackColor = BackColor
frmCmnDlg.lblOpenPath.ForeColor = ForeColor
frmCmnDlg.lblOpenPath.BackColor = BackColor
frmCmnDlg.filOpenList.ForeColor = ForeColor
frmCmnDlg.filOpenList.BackColor = BackColor
frmCmnDlg.drvOpenList.ForeColor = ForeColor
frmCmnDlg.drvOpenList.BackColor = BackColor
```

```
   frmCmnDlg.dirOpenList.ForeColor = ForeColor
   frmCmnDlg.dirOpenList.BackColor = BackColor
   frmCmnDlg.cmdOpenOK.BackColor = BackColor
   frmCmnDlg.cmdOpenCancel.BackColor = BackColor

   ' Display dialog modally.
   frmCmnDlg.SHOW 1

   ' Determine if user canceled dialog.
   IF frmCmnDlg.cmdOpenCancel.Tag <> "FALSE" THEN
      Cancel = TRUE
   ' If not, return FileName and PathName.
   ELSE
      Cancel = FALSE
      FileName = frmCmnDlg.txtOpenFile.Text
      PathName = frmCmnDlg.filOpenList.Path
      frmCmnDlg.cmdOpenCancel.Tag = ""
   END IF

   ' Hide or unload dialog and return control to user's program.
   ' (Hide if user chose to preload form for performance.)
   IF LEFT$(frmCmnDlg.Tag, 1) = "H" THEN
      frmCmnDlg.pctFileOpen.visible = FALSE
      frmCmnDlg.HIDE
      frmCmnDlg.Tag = "H"            ' Reset tag.
   ELSE
      UNLOAD frmCmnDlg
   END IF

   EXIT SUB

' Option error handling routine.
' Ignore errors here and let dialog's controls
' handle the errors.
FileSaveError:
   SELECT CASE ERR
   CASE 7:                           ' Out of memory.
       MSGBOX "Out of memory.  Can't load dialog.", 0, "FileSave"
       Cancel = TRUE
       EXIT SUB
   CASE ELSE
       RESUME NEXT
   END SELECT
END SUB


DECLARE SUB cmdPrintCancel_Click ()
' ------------------------------------------------------------------
' Visual Basic for MS-DOS Common Dialog Toolkit
'
' The Common Dialog Toolkit (CMNDLG.BAS and CMNDLGF.FRM)
' provides support for the following dialogs:
'     FileOpen
'     FileSave
```

```
'     FilePrint
'     FindText
'     ChangeText
'     ColorPalette
'     About
'
' Support for each dialog is provided via procedures with
' these same names that create the corresponding dialog
' and return user input to your program.  These procedures
' only provide the user interface and return user input.
' They do not actually carry out the corresponding actions
' such as opening the file.  Detailed descriptions of
' these procedures are contained in the comment headers
' above each.
'
' Special routines to preload (CmnDlgRegister) and unload
' (CmnDlgClose) the common dialog form for better
' performance (loaded forms display faster than unloaded
' forms) are also provided.  These routines are optional
' however as the common dialog form will automatically load
' and unload each time you invoke a common dialog.  Preloading
' the common dialog form will make common dialog access
' faster but will require more memory.
'
' All common dialogs are created from the same form (CMNDLGF.FRM).
' The necessary controls for each dialog are children of
' a container picture box for the dialog.  Thus the
' form (CMNDLGF.FRM) contains a picture box with
' appropriate controls for common dialog listed above.
' When a particular common dialog is created and displayed,
' the container picture box for that dialog is made visible
' (thus all controls on that picture box become visible)
' and the form is centered and sized to match the
' container picture box.
'
' To use these common dialogs in your programs, include
' CMNDLG.BAS and CMNDLGF.FRM in your program or use the
' supplied library (CMNDLG.LIB, CMNDLGA.LIB - AltMath version
' for Professional Edition only) and Quick library (CMNDLG.QLB)
' and call the appropriate procedure to invoke the dialog
' you need.
'
' Copyright (C) 1982-1992 Microsoft Corporation
'
' You have a royalty-free right to use, modify, reproduce
' and distribute the sample applications and toolkits provided with
' Visual Basic for MS-DOS (and/or any modified version)
' in any way you find useful, provided that you agree that
' Microsoft has no warranty, obligations or liability for
' any of the sample applications or toolkits.
' ---------------------------------------------------------------------
DIM SHARED E(5), V(5), HH(5), Options(3), KSUG(2), kbase(2), KSBASE(2), ZONES(6), SUBGRD(4)
DIM SHARED WPSI(7), SUGV(4), BASEV(4), TRAFICV(4), TEMPV(4), periodv(4), BITAIR(2),
MT(5), CODE(5), OLDBAIR(2)
```

```
COMMON E(), V(), HH(), WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), crack,
TXTCHANG, ZONES(), SUBGRD()
COMMON Options(), KSUG(), kbase(), KSBASE(), BITAIR(), CODE(), OLDBAIR(), MT(), NLAYER,
OLAYINC, TESTTEMP, OLAYTEMP
COMMON DSCRPTION$, A, B, AA, BB, K, Y, Z, R, T, G, ZON, SHOWPRN, PASTCHCK, OLAYT,
DAMA1, DAMA2, DAMA3, DAMA4, DAMA22

' Include file containing declarations for called procedures.
'$INCLUDE: 'CMNDLG.BI'

CONST FALSE = 0
CONST TRUE = NOT FALSE


SUB cmdAboutOK_Click ()
  Visible = FALSE
END SUB

SUB cmdOpenCancel_Click ()
  txtOpenFile.SETFOCUS
  Visible = FALSE
END SUB

SUB cmdOpenOK_CLick ()
  ' Set up error handling for directory/drive change errors.
  ON LOCAL ERROR GOTO OKError

  cmdOpenOK.SETFOCUS                ' Set focus to button, so focus can be reset to edit field if
needed.

  dirOpenList.Path = dirOpenList.List(dirOpenList.ListIndex)

  IF filOpenList.FileName <> txtOpenFile.TEXT THEN
    OldPattern$ = filOpenList.Pattern   ' Save old pattern.

    filOpenList.FileName = txtOpenFile.TEXT

    IF Visible = TRUE THEN
      IF (INSTR(txtOpenFile.TEXT, "*") + INSTR(txtOpenFile.TEXT, "?") < 1) THEN
        IF INSTR(Tag, "SAVE") THEN
          CALL filOpenList_DblClick
        ELSE
          MSGBOX "File not found", 0, Caption
          filOpenList.Pattern = OldPattern$   ' Restore old File listbox search pattern.
          txtOpenFile.SETFOCUS
        END IF
      ELSE
        txtOpenFile.TEXT = filOpenList.Pattern
        txtOpenFile.SETFOCUS
      END IF
    END IF
  ELSE
    CALL filOpenList_DblClick
  END IF
```

```
OKErrorReturn:
   EXIT SUB

' Drive/Path error handling routine.
OKError:
   MSGBOX ERROR$, 0, Caption          ' Display error message.
   txtOpenFile.SETFOCUS               ' Set focus to edit field so error can be fixed.
   RESUME OKErrorReturn               ' Exit procedure.
END SUB

' Click event procedure for Print dialog Cancel button.
' Makes dialog invisible to return control to FilePrint
' procedure (dialog was shown modally).
'
SUB cmdPrintCancel_Click ()
   Visible = FALSE
END SUB

SUB cmdPrintOK_Click ()
   ' Set up error handling for print to file errors.
   ON LOCAL ERROR GOTO PrintError

   ' Set print target
   IF optPrintTarget(0).value THEN
      PRINTER.PrintTarget = "LPT1:"      ' Use Basic LPT1 device (colon specifies this).
   ELSEIF optPrintTarget(1).value THEN
      PRINTER.PrintTarget = "LPT2:"      ' Use Basic LPT2 device (colon specifies this).
   ELSEIF optPrintTarget(2).value THEN
      PRINTER.PrintTarget = "LPT3"       ' No Basic LPT3 device, treat as a normal file open.
   ELSE
      ' Print target is a file.
   IF TXTPRINTFILE.TEXT = "" THEN
   MSGPR$ = "Invalid file name...Printing appended"
   MSGBOX MSGPR$, 0, "Print"
   CALL cmdPrintCancel_Click
'   frmCmnDlg.cmdPrintCancel.Tag = "true"
   GOTO 870
   END IF
      PRINTER.PrintTarget = TXTPRINTFILE.TEXT
      ' If user specified "Replace" instead of "Append"
      ' option, delete existing file.
      IF optprintappend(1).value THEN KILL TXTPRINTFILE.TEXT
   END IF
   IF optprintappend(0).value THEN
   prnfil$ = DIR$(TXTPRINTFILE.TEXT)
   IF prnfil$ <> "" THEN
   MSGPR$ = "File exist... Printing appended"
   MSGBOX MSGPR$, 0, "Print"
   CALL cmdPrintCancel_Click
'   frmCmnDlg.cmdPrintCancel.Tag = "true"
   END IF
   GOTO 870
   END IF
```

```
   cmdPrintCancel.Tag = "FALSE"
870 Visible = FALSE
   EXIT SUB

' Print to file error handling routine.
' Ignores File Not Found error that occurs when
' deleting a file that does not exist (when user
' chooses "Replace" option).
PrintError:
   RESUME NEXT
END SUB

SUB dirOpenList_Change ()
   ' Set up error handling for path errors.
   ON LOCAL ERROR GOTO DirChangeError

   ' Update file listbox path.
   filOpenList.Path = dirOpenList.Path

   ' Display new path to the user.
   lblOpenPath.Caption = dirOpenList.Path

   ' Update text box with search pattern.
   txtOpenFile.TEXT = filOpenList.Pattern

DirChangeErrorReturn:
   EXIT SUB

' Path change error handling routine.
DirChangeError:
   MSGBOX ERROR$, 0, Caption          ' Display error message.
   txtOpenFile.SETFOCUS               ' Set focus to edit field so error can be fixed.
   RESUME DirChangeErrorReturn        ' Exit procedure.
END SUB

SUB drvOpenList_Change ()
   ' Set up error handling for path errors.
   ON LOCAL ERROR GOTO DriveChangeError

   ' Update Dir listbox path.
   dirOpenList.Path = drvOpenList.Drive

DriveChangeErrorReturn:
   EXIT SUB

' Path change error handling routine.
DriveChangeError:
   MSGBOX ERROR$, 0, Caption            ' Display error message.
   drvOpenList.Drive = dirOpenList.Path ' Reset drive.
   RESUME DriveChangeErrorReturn        ' Exit procedure.
END SUB

SUB filOpenList_Click ()
   txtOpenFile.TEXT = filOpenList.FileName
```

```
END SUB

SUB filOpenList_DblClick ()
   txtOpenFile.SETFOCUS
   Visible = FALSE
   cmdOpenCancel.Tag = "FALSE"
END SUB

SUB filOpenList_PathChange ()
   ' Set up error handling for path errors.
   ON LOCAL ERROR GOTO FileChangeError

   ' Update drive and path.
   drvOpenList.Drive = filOpenList.Path
   dirOpenList.Path = filOpenList.Path

FileChangeErrorReturn:
   EXIT SUB

' Path change error handling routine.
FileChangeError:
   MSGBOX ERROR$, 0, Caption            ' Display error message.
   drvOpenList.Drive = dirOpenList.Path   ' Reset drive.
   filOpenList.Path = dirOpenList.Path    ' Reset path.
   RESUME FileChangeErrorReturn          ' Exit procedure.
END SUB

SUB filOpenList_PatternChange ()
   filOpenList.Pattern = UCASE$(filOpenList.Pattern)
END SUB

SUB Form_Click ()

END SUB

SUB optPrintTarget_Click (Index AS INTEGER)
   ' If file is chosen as print target, enable
   ' filename edit field and append/replace options.
   IF Index = 3 THEN
      TXTPRINTFILE.Enabled = TRUE
      pctPrintAppend.Enabled = TRUE
   ' If LPT1, LPT2, LPT3 is chosen as print target,
   ' disable filename edit field and append/replace options.
   ELSE
      TXTPRINTFILE.Enabled = FALSE
      pctPrintAppend.Enabled = FALSE
   END IF
END SUB

SUB txtOpenFile_GotFocus ()
   txtOpenFile.SelStart = 0
   txtOpenFile.SelLength = LEN(txtOpenFile.TEXT)
END SUB
```

```
SUB txtPrintCopies_GotFocus ()
  txtPrintCopies.SelStart = 0
  txtPrintCopies.SelLength = LEN(txtPrintCopies.TEXT)
END SUB


DECLARE SUB SUGCLASS_CLICK (INDEX AS INTEGER)
DECLARE SUB Form_Load ()
DIM SHARED E(5), V(5), HH(5), OPTIONS(3), KSUG(2), kbase(2), KSBASE(2), ZONES(6),
SUBGRD(4)
DIM SHARED WPSI(7), SUGV(4), BASEV(4), TRAFICV(4), TEMPV(4), periodv(4), BITAIR(2),
MT(5), CODE(5), OLDBAIR(2)
COMMON E(), V(), HH(), WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), crack,
TXTCHANG, ZONES(), SUBGRD()
COMMON OPTIONS(), KSUG(), kbase(), KSBASE(), BITAIR(), CODE(), OLDBAIR(), MT(),
NLAYER, OLAYINC, TESTTEMP, OLAYTEMP
COMMON DSCRPTION$, A, B, AA, BB, K, Y, Z, R, T, G, ZON, SHOWPRN, PASTCHCK, OLAYT,
DAMA1, DAMA2, DAMA3, DAMA4, DAMA22


SUB Check1_Click ()
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), crack, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA, BB, K, R, T, G,
OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2, DAMA3,
DAMA4, DAMA22
PASTCHCK = CHECK1.VALUE
IF CHECK1.VALUE = 0 THEN
LABEL12.VISIBLE = 0
TEXT1(4).VISIBLE = 0
WPSI(4) = 0
ELSE
LABEL12.VISIBLE = -1
TEXT1(4).VISIBLE = -1
END IF
END SUB

SUB Form_Load ()
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), BITAIR(), OLDBAIR(),
NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2
'WPSI(5) IS THE FATIGUE SHIFT FACTOR FOR NEW ASPHALT LAYER
'WPSI(6) IS THE FATIGUE SHIFT FACTOR FOR OLD ASPHALT LAYER
K = K + 1
IF K = 1 THEN
GOTO 15
END IF
CHECK1.VALUE = PASTCHCK
FOR I = 0 TO 6
AA = I
BB = I
IF WPSI(I) = 0 THEN
TEXT1(I).TEXT = ""
```

```
ELSE
TEXT1(I).TEXT = STR$(WPSI(I))
END IF
NEXT I
FOR I = 0 TO 3
A = I
B = I
IF SUGV(I) = 0 THEN
TEXT2(I).TEXT = ""
ELSE
TEXT2(I).TEXT = STR$(SUGV(I))
END IF
IF TRAFICV(I) = 0 THEN
text4(I).TEXT = ""
ELSE
text4(I).TEXT = STR$(TRAFICV(I))
END IF
IF TEMPV(I) = 0 THEN
text5(I).TEXT = ""
ELSE
text5(I).TEXT = STR$(TEMPV(I))
END IF
IF periodv(I) = 0 THEN
text6(I).TEXT = ""
ELSE
text6(I).TEXT = STR$(periodv(I))
END IF
NEXT I
FOR I = 0 TO 3
A = I
B = I
IF NLAYER = 3 THEN
text3(I).TEXT = ""
text3(I).enabled = 0
ELSEIF BASEV(I) = 0 THEN
text3(I).TEXT = ""
ELSE
text3(I).TEXT = STR$(BASEV(I))
END IF
NEXT I
IF ZONE(0).VALUE = -1 OR ZONE(1).VALUE = -1 OR ZONE(3).VALUE = -1 OR ZONE(4).VALUE
= -1 THEN
IF E(NLAYER - 1) > 0 THEN
SUGV(0) = (45 * (.8838 ^ E(NLAYER - 1)) + 1) / 2
IF SUGV(0) < 1 THEN SUGV(0) = 1
IF SUGV(0) >= 10 THEN TEXT2(0).TEXT = FORMAT$(SUGV(0), "00.0")
TEXT2(0).TEXT = FORMAT$(SUGV(0), "0.0")
END IF
IF E(NLAYER - 1) > 0 AND E(NLAYER - 1) <= 6 THEN SUGV(1) = (.002778 + .012549 *
E(NLAYER - 1)) ^ (1 / 3)

IF E(NLAYER - 1) > 6 AND E(NLAYER - 1) <= 20 THEN SUGV(1) = .43127 + 8.84E-09 *
(E(NLAYER - 1)) ^ 6
IF E(NLAYER - 1) > 20 THEN SUGV(1) = 1
```

```
IF SUGV(1) <> 1 THEN TEXT2(1).TEXT = FORMAT$(SUGV(1), "0.00")
END IF
IF ZONE(2).VALUE = -1 OR ZONE(5).VALUE = -1 THEN
FOR I = 0 TO 3
SUGCLASS(I).VALUE = SUBGRD(I)
NEXT I
END IF
  FOR I = 0 TO 6
  ZONE(I).VALUE = ZONES(I)
  NEXT I
15 A = 0
  B = 3
  AA = 0
  BB = 6
END SUB


SUB SUGCLASS_CLICK (INDEX AS INTEGER)
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), crack, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA, BB, K, R, T, G,
OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2, DAMA3,
DAMA4, DAMA22
FOR I = 0 TO 3
SUBGRD(I) = SUGCLASS(I).VALUE
NEXT I
IF ZONE(2).VALUE = -1 THEN
WN = 1.3125 + .09375 * E(NLAYER - 1)
ELSEIF ZONE(5).VALUE = -1 THEN
WN = 2.2857 + .1071 * E(NLAYER - 1)
ELSE
GOTO 550
END IF
IF SUGCLASS(0).VALUE = -1 THEN
IF WN >= 0 AND WN <= 2 THEN SUGV(0) = 1 - .1 * WN
IF WN > 2 AND WN <= 6 THEN SUGV(0) = .963 * (.91) ^ WN
IF WN > 6 THEN SUGV(0) = .66
END IF
IF SUGCLASS(1).VALUE = -1 THEN
IF WN >= 0 AND WN <= 1 THEN SUGV(0) = 1 - .3 * WN
IF WN > 1 AND WN <= 4 THEN SUGV(0) = .786 * (.869) ^ WN
IF WN > 4 THEN SUGV(0) = .41
END IF
IF SUGCLASS(2).VALUE = -1 THEN
IF WN >= 0 AND WN <= 1 THEN SUGV(0) = 1 - .4 * WN
IF WN > 1 AND WN <= 4 THEN SUGV(0) = .735 * (.798) ^ WN
IF WN > 4 THEN SUGV(0) = .3
END IF
IF SUGCLASS(3).VALUE = -1 THEN
IF WN >= 0 AND WN <= 1 THEN SUGV(0) = 1 - .5 * WN
IF WN > 1 AND WN <= 4 THEN SUGV(0) = .535 * (.7962) ^ WN
IF WN > 4 THEN SUGV(0) = .22
END IF
A = 0
```

```
B = 0
TEXT2(0).TEXT = FORMAT$(SUGV(0), "0.00")
A = 1
B = 1
TEXT2(1).TEXT = FORMAT$((SUGV(0) + 1) / 2, "0.00")
A = 0
B = 3
550 END SUB

SUB Text1_Change (INDEX AS INTEGER)
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), BITAIR(), OLDBAIR(),
NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2
FOR I = AA TO BB
WPSI(I) = VAL(TEXT1(I).TEXT)
NEXT I
END SUB

SUB Text2_Change (INDEX AS INTEGER)
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), BITAIR(), OLDBAIR(),
NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2
FOR I = A TO B
SUGV(I) = VAL(TEXT2(I).TEXT)

NEXT I

END SUB

SUB Text3_Change (INDEX AS INTEGER)
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), BITAIR(), OLDBAIR(),
NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2
FOR I = A TO B
BASEV(I) = VAL(text3(I).TEXT)

NEXT I

END SUB

SUB Text4_Change (INDEX AS INTEGER)
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), BITAIR(), OLDBAIR(),
NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2
FOR I = A TO B
TRAFICV(I) = VAL(text4(I).TEXT)

NEXT I

END SUB
```

```
SUB Text5_Change (INDEX AS INTEGER)
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), BITAIR(), OLDBAIR(),
NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2
FOR I = A TO B
TEMPV(I) = VAL(text5(I).TEXT)

NEXT I

END SUB

SUB Text6_Change (INDEX AS INTEGER)
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), BITAIR(), OLDBAIR(),
NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2
FOR I = A TO B
periodv(I) = VAL(text6(I).TEXT)

NEXT I

END SUB

SUB ZONE_Click (INDEX AS INTEGER)
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), crack, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2
TXTCHANG = -1
IF ZONE(0).VALUE = -1 OR ZONE(1).VALUE = -1 OR ZONE(3).VALUE = -1 OR ZONE(4).VALUE
= -1 THEN
WIN(0).CAPTION = " FROZ"
WIN(1).CAPTION = " THOW"
WIN(2).CAPTION = " NORM"
WIN(3).CAPTION = " NORM"
END IF
IF ZONE(2).VALUE = -1 OR ZONE(5).VALUE = -1 THEN
WIN(0).CAPTION = " WET"
WIN(1).CAPTION = "WET-R"
WIN(2).CAPTION = " NORM"
WIN(3).CAPTION = " NORM"
END IF

IF ZONE(0).VALUE = -1 THEN
SUGFRAM.VISIBLE = 0
ZON = 1
TEMPV(0) = 31
TEMPV(1) = 55
TEMPV(2) = 62
TEMPV(3) = 34
SUGV(0) = 1
SUGV(1) = 1
```

```
SUGV(2) = 1
SUGV(3) = 1
BASEV(0) = 1
BASEV(1) = .65
BASEV(2) = 1
BASEV(3) = 1
periodv(0) = 4
periodv(1) = 1.5
periodv(2) = 3.5
periodv(3) = 3
END IF
IF ZONE(1).VALUE = -1 THEN
SUGFRAM.VISIBLE = 0
ZON = 2
TEMPV(0) = 32
TEMPV(1) = 58
TEMPV(2) = 65
TEMPV(3) = 33
SUGV(0) = 1
SUGV(1) = 1
SUGV(2) = 1
SUGV(3) = 1
BASEV(0) = 1
BASEV(1) = .65
BASEV(2) = 1
BASEV(3) = 1
periodv(0) = 4
periodv(1) = 1.5
periodv(2) = 3.5
periodv(3) = 3
END IF
IF ZONE(2).VALUE = -1 THEN
FOR I = 0 TO 3
SUGCLASS(I).VALUE = SUBGRD(I)
NEXT I
SUGFRAM.VISIBLE = -1
ZON = 3
TEMPV(0) = 44
TEMPV(1) = 58
TEMPV(2) = 66
TEMPV(3) = 36
SUGV(0) = 1
SUGV(1) = 1
SUGV(2) = 1
SUGV(3) = 1
BASEV(0) = .65
BASEV(1) = .85
BASEV(2) = 1
BASEV(3) = 1
periodv(0) = 3
periodv(1) = 1
periodv(2) = 4
periodv(3) = 4
CALL SUGCLASS_CLICK(-1)
```

```
END IF
IF ZONE(3).VALUE = -1 THEN
SUGFRAM.VISIBLE = 0
ZON = 4
TEMPV(0) = 33
TEMPV(1) = 53
TEMPV(2) = 62
TEMPV(3) = 36
SUGV(0) = 1
SUGV(1) = 1
SUGV(2) = 1
SUGV(3) = 1
BASEV(0) = 1
BASEV(1) = .65
BASEV(2) = 1
BASEV(3) = 1
periodv(0) = 3
periodv(1) = 1
periodv(2) = 4
periodv(3) = 4
END IF
IF ZONE(4).VALUE = -1 THEN
SUGFRAM.VISIBLE = 0
ZON = 5
TEMPV(0) = 33
TEMPV(1) = 56
TEMPV(2) = 60
TEMPV(3) = 35
SUGV(0) = 1
SUGV(1) = 1
SUGV(2) = 1
SUGV(3) = 1
BASEV(0) = 1
BASEV(1) = .65
BASEV(2) = 1
BASEV(3) = 1
periodv(0) = 4
periodv(1) = 1
periodv(2) = 4
periodv(3) = 3
END IF
IF ZONE(5).VALUE = -1 THEN
FOR I = 0 TO 3
SUGCLASS(I).VALUE = SUBGRD(I)
NEXT I
SUGFRAM.VISIBLE = -1
ZON = 6
TEMPV(0) = 48
TEMPV(1) = 59
TEMPV(2) = 65
TEMPV(3) = 34
SUGV(0) = 1
SUGV(1) = 1
SUGV(2) = 1
```

```
SUGV(3) = 1
BASEV(0) = .65
BASEV(1) = .85
BASEV(2) = 1
BASEV(3) = 1
periodv(0) = 3
periodv(1) = 1
periodv(2) = 4
periodv(3) = 4
CALL SUGCLASS_CLICK(-1)
END IF
IF ZONE(6).VALUE = -1 THEN
SUGFRAM.VISIBLE = 0
WIN(0).CAPTION = ""
WIN(1).CAPTION = ""
WIN(2).CAPTION = ""
WIN(3).CAPTION = ""
ZON = 0
FOR I = 0 TO 3
TEXT2(I).enabled = -1
text6(I).enabled = -1
IF NLAYER = 3 THEN GOTO 321
text3(I).enabled = -1
321 text5(I).enabled = -1
NEXT I
ELSE
FOR I = 0 TO 3
TEXT2(I).enabled = 0
text3(I).enabled = 0
text5(I).enabled = 0
text6(I).enabled = 0
NEXT I
END IF
K = 1
FOR I = 0 TO 6
ZONES(I) = ZONE(I).VALUE
NEXT I
CALL Form_Load
END SUB

DECLARE SUB INPUTERROR (MSG$)
DECLARE SUB mnurunmsg_click ()
'$FORM FRMResult
'$FORM RUNMSG
'$FORM FRMGEN
DECLARE SUB FileOpen (FILENAME AS STRING, PATHNAME AS STRING, DefaultExt AS
STRING, DialogTitle AS STRING, FORECOLOR AS INTEGER, BACKCOLOR AS INTEGER, Flags
AS INTEGER, Cancel AS INTEGER)
'$FORM FRMPAVE
DIM SHARED E(5), V(5), HH(5), OPTIONS(3), KSUG(2), kbase(2), KSBASE(2), ZONES(6),
SUBGRD(4)
DIM SHARED WPSI(7), SUGV(4), BASEV(4), TRAFICV(4), TEMPV(4), periodv(4), BITAIR(2),
MT(5), CODE(5), OLDBAIR(2)
```

```
COMMON E(), V(), HH(), WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), CRACK,
TXTCHANG, ZONES(), SUBGRD()
COMMON OPTIONS(), KSUG(), kbase(), KSBASE(), BITAIR(), CODE(), OLDBAIR(), MT(),
NLAYER, OLAYINC, TESTTEMP, OLAYTEMP
COMMON DSCRPTION$, A, B, AA, BB, K, Y, Z, R, T, G, ZON, SHOWPRN, PASTCHCK, OLAYT,
DAMA1, DAMA2, DAMA3, DAMA4, DAMA22
COMMON SHARED LT
COMMON INPFILE$
'DECLARE SUB INPUTERROR (MSG$)


SUB AUTO_CLICK ()
SHARED LT
LT = 1
FOR I = 1 TO 4
FRMPAVE.TEXT1(I).ENABLED = 0
FRMPAVE.TEXT1(I).TEXT = " N/A"
NEXT I
FRMPAVE.TEXT5.ENABLED = 0
FRMPAVE.TEXT5.TEXT = " N/A"
END SUB

SUB Command1_Click ()
UNLOAD FRMINPUT
LOAD FRMPAVE
LOAD FRMGEN
FRMPAVE.SHOW
IF LT = 1 THEN
FOR I = 1 TO NLAYER - 1
FRMPAVE.TEXT1(I).ENABLED = 0
FRMPAVE.TEXT1(I).TEXT = " N/A"
NEXT I
END IF
END SUB

SUB Form_Click ()

END SUB

SUB MANUALLY_CLICK ()
SHARED LT
LT = 0
FOR I = 1 TO 4
FRMPAVE.TEXT1(I).ENABLED = -1
FRMPAVE.TEXT1(I).TEXT = ""
NEXT I
FRMPAVE.TEXT5.ENABLED = -1
FRMPAVE.TEXT5.TEXT = ""
END SUB


DECLARE SUB INPUTERROR1 (MSG$)
'$FORM FRMOUTPUT
'$FORM FRMINPUT
```

```
'$FORM MAIN
DECLARE SUB mnurunmsg_click ()
'$FORM RUNMSG
DECLARE SUB MNUPAGESHOW_CLICK ()
DECLARE SUB PAGESHOW ()
DECLARE SUB FilePrint (COPIES AS INTEGER, FORECOLOR AS INTEGER, BACKCOLOR AS
INTEGER, Cancel AS INTEGER)
DECLARE SUB MNUNEW_CLICK ()
DECLARE SUB MNUFILESAVEFILE_CLICK ()
DECLARE SUB INPUTERROR (MSG$)
DECLARE SUB MNUFILEEXIT_CLICK ()
DECLARE SUB FileSave (FILENAME AS STRING, PATHNAME AS STRING, DefaultExt AS
STRING, DialogTitle AS STRING, FORECOLOR AS INTEGER, BACKCOLOR AS INTEGER, Flags
AS INTEGER, Cancel AS INTEGER)
DECLARE SUB FileOpen (FILENAME AS STRING, PATHNAME AS STRING, DefaultExt AS
STRING, DialogTitle AS STRING, FORECOLOR AS INTEGER, BACKCOLOR AS INTEGER, Flags
AS INTEGER, Cancel AS INTEGER)
DECLARE SUB About (AboutText AS STRING, FORECOLOR AS INTEGER, BACKCOLOR AS
INTEGER, Flags AS INTEGER)
'$FORM frmcmndlg
'$FORM FRMPAVE
'$FORM FRMGEN
'$FORM frmmat
'$FORM FRMResult
'$FORM frmpage
'$INCLUDE: 'CONSTANT.BI'
DIM SHARED E(5), V(5), HH(5), OPTIONS(3), KSUG(2), kbase(2), KSBASE(2), ZONES(6),
SUBGRD(4)
DIM SHARED WPSI(7), SUGV(4), BASEV(4), TRAFICV(4), TEMPV(4), periodv(4), BITAIR(2),
MT(5), CODE(5), OLDBAIR(2)
COMMON E(), V(), HH(), WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), CRACK,
TXTCHANG, ZONES(), SUBGRD()
COMMON OPTIONS(), KSUG(), kbase(), KSBASE(), BITAIR(), CODE(), OLDBAIR(), MT(),
NLAYER, OLAYINC, TESTTEMP, OLAYTEMP
COMMON DSCRPTION$, A, B, AA, BB, K, Y, Z, R, T, G, ZON, SHOWPRN, PASTCHCK, OLAYT,
DAMA1, DAMA2, DAMA3, DAMA4, DAMA22
COMMON SHARED LT
COMMON INPFILE$
COMMON NETF, HEADER$

SUB Form_Click ()

END SUB

SUB Form_Load ()
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), CRACK, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA, BB, K, R, T, G,
OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2, DAMA3,
DAMA4, DAMA22
SHARED LT

SCREEN.ControlPanel(2) = 14
SCREEN.ControlPanel(1) = 8
```

```
SCREEN.ControlPanel(17) = 14
SCREEN.ControlPanel(16) = 9
SCREEN.ControlPanel(10) = 15
SCREEN.ControlPanel(9) = 3
SCREEN.ControlPanel(0) = 12
SCREEN.ControlPanel(3) = 0
SCREEN.ControlPanel(15) = -1
SCREEN.ControlPanel(DISABLED_ITEM_FORECOLOR) = 4
FILE$ = "UNTITLED"
SHOWPRN = 0
OPTIONS(0) = -1
NLAYER = 5
PASTCHCK = 1
CALL MNUPAGESHOW_CLICK
100 END SUB

SUB INPUTERROR (MSG$)
SHARED RESPONSE%
RESPONSE% = MSGBOX(MSG$, 1, "INPUT ERROR")
END SUB

SUB INPUTERROR1 (MSG$)
SHARED RESPONSE%
RESPONSE% = MSGBOX(MSG$, 0, "INPUT ERROR")
END SUB

SUB mnuabout_Click ()
UNLOAD frmpage
About "Principal Investigator: Fouad Bayomy", 5, 7, 1
END SUB

SUB mnudatagen_Click ()
UNLOAD FRMPAVE
FRMGEN.SHOW
END SUB

SUB MNUDATAMAT_Click ()
frmmat.SHOW
END SUB

SUB mnudatapave_Click ()
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), CRACK, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2

UNLOAD FRMGEN
FRMPAVE.SHOW
END SUB

SUB MNUDATA_Click ()
UNLOAD frmpage
END SUB
```

```
SUB MNUFILEEXIT_CLICK ()
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), CRACK, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK,
savxit
SHARED E(), DSCRPTION$, V(), HH(), TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA, BB, K, R, T, G,
OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2, DAMA3,
DAMA4, DAMA22
IF TXTCHANG = 0 THEN GOTO 213
savxit = 0
19 MSG$ = " SAVE [.INP] FILE BEFORE EXIT "
RESPONCE% = MSGBOX(MSG$, 3, "EXIT MESSAGE")
IF RESPONCE% = 2 THEN GOTO 31
IF RESPONCE% = 6 THEN
CALL MNUFILESAVEFILE_CLICK
IF savxit = -2 THEN GOTO 19
END IF
ON LOCAL ERROR GOTO 213
KILL "LIFE.OUT"
KILL "RESULTS.OUT"
KILL "RMLIFE.TMP"
KILL "OLAY.TMP"
213 END
31 END SUB

SUB mnufileopen_Click ()
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), CRACK, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK,
savxit
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2,
DAMA3, DAMA4, DAMA22
SHARED CANCL
SHARED INPFILE$
SHARED LT
'LT = 2
CALL MNUNEW_CLICK
UNLOAD FRMINPUT
IF CANCL = -1 THEN GOTO 8
FileOpen FILENAME$, PATHNAME$, "*.INP", "OPEN", 0, 7, 0, Cancel%
IF Cancel% = -1 THEN GOTO 8
OLDFILE$ = FILE$
OLDPATH$ = PATH$
IF RIGHT$(PATHNAME$, 1) = "\" THEN
PATHNAME$ = LEFT$(PATHNAME$, LEN(PATHNAME$) - 1)
END IF
FILE$ = FILENAME$
PATH$ = PATHNAME$
CAPTION = "MAIN : " + PATH$ + "\" + FILE$
OPEN PATHNAME$ + "\" + FILENAME$ FOR INPUT AS #3
UNLOAD FRMGEN
UNLOAD frmmat
UNLOAD FRMPAVE
ON LOCAL ERROR GOTO ERRORHANDLER
'INPUT #3, PASSWORD$
```

```
'IF PASSWORD$ <> "PASSWORD" THEN
'MSGBOX "Can't open the file", 0, "OPEN ERROR"
'CLOSE
'CAPTION = "MAIN : "
'TXTCHANG = 0
'CALL MNUNEW_CLICK
'GOTO 120
'END IF
INPFILE$ = FILENAME$
INPUT #3, OLAYINC
INPUT #3, TESTTEMP, OLAYTEMP
FOR I = 0 TO 6
INPUT #3, WPSI(I)
NEXT I
FOR I = 0 TO 3
INPUT #3, SUGV(I)
NEXT I
FOR I = 0 TO 3
INPUT #3, BASEV(I)
NEXT I
FOR I = 0 TO 3
INPUT #3, TRAFICV(I)
NEXT I
FOR I = 0 TO 3
INPUT #3, TEMPV(I)
NEXT I
FOR I = 0 TO 3
INPUT #3, periodv(I)
NEXT I
INPUT #3, NLAYER
FOR I = 0 TO NLAYER - 1
INPUT #3, E(I), V(I)
NEXT I
IF LT = 0 THEN
FOR I = 1 TO NLAYER - 1
FRMPAVE.TEXT1(I).ENABLED = -1
FRMPAVE.TEXT5.ENABLED = -1
NEXT I
END IF
IF NLAYER = 3 THEN
INPUT #3, CODE(2)
END IF
IF NLAYER = 4 THEN
INPUT #3, CODE(2)
INPUT #3, CODE(3)
END IF
IF NLAYER = 5 THEN
INPUT #3, CODE(2)
INPUT #3, CODE(3)
INPUT #3, CODE(4)
END IF
IF NLAYER = 3 THEN
FOR I = 0 TO 1
INPUT #3, KSUG(I)
```

```
NEXT I
END IF
IF NLAYER = 4 THEN
FOR I = 0 TO 1
INPUT #3, kbase(I)
NEXT I
FOR I = 0 TO 1
INPUT #3, KSUG(I)
NEXT I
END IF
IF NLAYER = 5 THEN
FOR I = 0 TO 1
INPUT #3, kbase(I)
NEXT I
FOR I = 0 TO 1
INPUT #3, KSBASE(I)
NEXT I
FOR I = 0 TO 1
INPUT #3, KSUG(I)
NEXT I
END IF
FOR I = 0 TO NLAYER - 2
INPUT #3, HH(I)
NEXT I
INPUT #3, OLDBAIR(0), OLDBAIR(1)
OLDBAIR(0) = 11
OLDBAIR(1) = 5
INPUT #3, BITAIR(0), BITAIR(1)
BITAIR(0) = 11
BITAIR(1) = 5
INPUT #3, CRACK
INPUT #3, Z, K, G
FOR I = 0 TO 2
INPUT #3, OPTIONS(I)
NEXT I
FOR I = 0 TO 6
INPUT #3, ZONES(I)
IF ZONES(I) = -1 THEN
ZON = I + 1
IF I = 6 THEN ZON = 0
END IF
NEXT I
FOR I = 0 TO 3
INPUT #3, SUBGRD(I)
NEXT I
INPUT #3, PASTCHCK
INPUT #3, DSCRPTION$
LOAD FRMPAVE
FRMPAVE.SHOW
8 CLOSE
TXTCHANG = -1
savxit = -1
CANCL = 0
120 EXIT SUB
```

```
ERRORHANDLER:
CLOSE
CAPTION = "MAIN : "
MSG$ = "Can't open the file"
MSGBOX MSG$
TXTCHANG = 0
CALL MNUNEW_CLICK
RESUME 1010
1010 END SUB


SUB MNUFILESAVEFILE_CLICK ()
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), CRACK, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK,
savxit
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2,
DAMA3, DAMA4, DAMA22
440 FileSave FILENAME$, PATHNAME$, FILE$, "save", 0, 7, 0, Cancel%
IF Cancel% = -1 THEN
savxit = -2
GOTO 88
END IF
IF RIGHT$(PATHNAME$, 1) = "\" THEN
PATHNAME$ = LEFT$(PATHNAME$, (LEN(PATHNAME$) - 1))
END IF
SAVFILE$ = DIR$(FILENAME$)
IF SAVFILE$ = FILENAME$ THEN
SVMSG$ = "File already exist ...Replace"
RESPONSE% = MSGBOX(SVMSG$, 4, "SAVE")
IF RESPONSE% = 7 THEN GOTO 440
END IF
FILE$ = FILENAME$
PATH$ = PATHNAME$
CAPTION = "MAIN : " + PATH$ + "\" + FILE$
OPEN PATHNAME$ + "\" + FILENAME$ FOR OUTPUT AS #9
'PRINT #9, "PASSWORD"
PRINT #9, OLAYINC
PRINT #9, TESTTEMP; OLAYTEMP
FOR I = 0 TO 6
PRINT #9, WPSI(I);
NEXT I
PRINT #9,
FOR I = 0 TO 3
PRINT #9, SUGV(I);
NEXT I
PRINT #9,
FOR I = 0 TO 3
PRINT #9, BASEV(I);
NEXT I
PRINT #9,
FOR I = 0 TO 3
PRINT #9, TRAFICV(I);
```

```
NEXT I
PRINT #9,
FOR I = 0 TO 3
PRINT #9, TEMPV(I);
NEXT I
PRINT #9,
FOR I = 0 TO 3
PRINT #9, periodv(I);
NEXT I
PRINT #9,
PRINT #9, NLAYER
FOR I = 0 TO NLAYER - 1
PRINT #9, E(I); V(I)
NEXT I
IF NLAYER = 3 THEN
PRINT #9, CODE(2)
END IF
IF NLAYER = 4 THEN
PRINT #9, CODE(2)
PRINT #9, CODE(3)
END IF
IF NLAYER = 5 THEN
PRINT #9, CODE(2)
PRINT #9, CODE(3)
PRINT #9, CODE(4)
END IF
IF NLAYER = 3 THEN
FOR I = 0 TO 1
PRINT #9, KSUG(I);
NEXT I
PRINT #9,
END IF
IF NLAYER = 4 THEN
FOR I = 0 TO 1
PRINT #9, kbase(I);
NEXT I
PRINT #9,
FOR I = 0 TO 1
PRINT #9, KSUG(I);
NEXT I
PRINT #9,
END IF
IF NLAYER = 5 THEN
FOR I = 0 TO 1
PRINT #9, kbase(I);
NEXT I
PRINT #9,
FOR I = 0 TO 1
PRINT #9, KSBASE(I);
NEXT I
PRINT #9,
FOR I = 0 TO 1
PRINT #9, KSUG(I);
NEXT I
```

```
PRINT #9,
END IF
FOR I = 0 TO NLAYER - 2
PRINT #9, HH(I)
NEXT I
PRINT #9, OLDBAIR(0), OLDBAIR(1)
PRINT #9, BITAIR(0), BITAIR(1)
PRINT #9, CRACK
PRINT #9, Z, K, G
FOR I = 0 TO 2
PRINT #9, OPTIONS(I)
NEXT I
FOR I = 0 TO 6
PRINT #9, ZONES(I)
NEXT I
FOR I = 0 TO 3
PRINT #9, SUBGRD(I)
NEXT I
PRINT #9, PASTCHCK
PRINT #9, DSCRPTION$
PRINT #9, FILE$
savxit = -1
TXTCHANG = 0
88 CLOSE
END SUB

SUB mnufile_Click ()
UNLOAD frmpage
END SUB

SUB MNUNEW_CLICK ()
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), CRACK, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2,
DAMA3, DAMA4, DAMA22
SHARED RTVALUE
SHARED CANCL
SHARED LT
UNLOAD FRMGEN
UNLOAD frmmat
UNLOAD FRMPAVE
IF TXTCHANG = 0 THEN GOTO 787
savxit = 0
119 MSG$ = " Save current [.INP] file "
RESPONCE% = MSGBOX(MSG$, 3, "NEW FILE MESSAGE")
IF RESPONCE% = 2 THEN
CANCL = -1
GOTO 331
END IF
IF RESPONCE% = 6 THEN
CALL MNUFILESAVEFILE_CLICK
IF savxit = -2 THEN GOTO 119
END IF
```

```
787 Z = 0
K = 0
G = 0
FILE$ = "UNTITLED"
CAPTION = "MAIN : " + FILE$
OPTIONS(0) = -1
OPTIONS(1) = 0
OPTIONS(2) = 0
BITAIR(0) = 11
BITAIR(1) = 5
OLAYINC = 0
TESTTEMP = 0
OLAYTEMP = 0
WPSI(0) = 4500
WPSI(1) = 13.5
WPSI(2) = 80
WPSI(3) = 0
WPSI(4) = 0
WPSI(5) = 18.4
WPSI(6) = 15
FOR I = 0 TO 4
SUGV(I) = 0
BASEV(I) = 0
TRAFICV(I) = 0
TEMPV(I) = 0
periodv(I) = 0
E(I) = 0
V(I) = 0
HH(I) = 0
CODE(I) = 0
ZONES(I) = 0
NEXT I
ZONES(5) = 0
ZONES(6) = -1
CRACK = 0
DSCRPTION$ = ""
SHOWPRN = 0
TXTCHANG = 0
NLAYER = 5
331 LOAD FRMINPUT
FRMINPUT.SHOW
OLDBAIR(0) = 11
OLDBAIR(1) = 5
END SUB

SUB MNUPAGESHOW_CLICK ()
frmpage.SHOW
END SUB

SUB MNURESPRN_Click ()
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), CRACK, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
```

```
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2,
DAMA3, DAMA4, DAMA22
DIM MODULUS(1 TO 4, 1 TO 5)
DIM STRAIN(1 TO 4)
DIM ZZ(1 TO 4)
DIM LIFE(1 TO 4)
ON LOCAL ERROR GOTO printerr
FilePrint COPIES%, 1, 7, Cancel%
IF Cancel% = -1 THEN GOTO 112
FOR P = 1 TO COPIES%
PRINTER.PRINT
PRINTER.PRINT FORMAT$(NOW, "    ddddd  ttttt")
PRINTER.PRINT
PRINTER.PRINT "    PAGE 1/2"
PRINTER.PRINT
PRINTER.PRINT "    DESCRIPTION : "; DSCRPTION$
PRINTER.PRINT
PRINTER.PRINT "    1. SUMMARY OF INPUT DATA"
PRINTER.PRINT "    ----------------------"
PRINTER.PRINT
PRINTER.PRINT "    1.1  TRAFFIC DATA"
PRINTER.PRINT "    ----------------"
FOR I = 0 TO 6
SELECT CASE I
CASE 0
PRINTER.PRINT "    DESIGN DUAL TIRE LOAD              ="; WPSI(I)
CASE 1
PRINTER.PRINT "    DESIGN DUAL TIRE SPACING           ="; WPSI(I)
CASE 2
PRINTER.PRINT "    TIRE PRESSURE (psi)                ="; WPSI(I)
CASE 3
PRINTER.PRINT "    DESIGN FUTURE TRAFFIC (ESALs)      ="; WPSI(I)
CASE 4
PRINTER.PRINT "    ESTIMATED PAST TRAFFIC (ESALs)     ="; WPSI(I)
CASE 5
PRINTER.PRINT "    FATIGUE SHIFT FACTOR FOR NEW ASPHALT  ="; WPSI(I)
CASE 6
PRINTER.PRINT "    FATIGUE SHIFT FACTOR FOR OLD ASPHALT  ="; WPSI(I)
END SELECT
NEXT I
PRINTER.PRINT
PRINTER.PRINT "    1.2  SEASONAL VARIATION DATA "
PRINTER.PRINT "    --------------------------- "
PRINTER.PRINT "                          WINTER  SPRING  SUMMER  FALL"
PRINTER.PRINT "    SUBGRADE VARIATION  "; "  "; FORMAT$(SUGV(0), "0.00"); "    ";
FORMAT$(SUGV(1), "0.00"); "    "; FORMAT$(SUGV(2), "0.00"); "    "; FORMAT$(SUGV(3), "0.00")
PRINTER.PRINT "    BASE/SBASE VARIATION "; "  "; FORMAT$(BASEV(0), "0.00"); "    ";
FORMAT$(BASEV(1), "0.00"); "    "; FORMAT$(BASEV(2), "0.00"); "    "; FORMAT$(BASEV(3),
"0.00")
PRINTER.PRINT "    TRAFFIC VARIATION   "; "  "; FORMAT$(TRAFICV(0), "0.00"); "    ";
FORMAT$(TRAFICV(1), "0.00"); "    "; FORMAT$(TRAFICV(2), "0.00"); "    ";
FORMAT$(TRAFICV(3), "0.00")
```

```
PRINTER.PRINT "    TEMPERATURE VARIATION"; "   "; FORMAT$(TEMPV(0), "0.00"); "   ";
FORMAT$(TEMPV(1), "0.00"); "   "; FORMAT$(TEMPV(2), "0.00"); "   "; FORMAT$(TEMPV(3),
"0.00")
PRINTER.PRINT "    PERIOD (MONTHS)     "; "   "; FORMAT$(periodv(0), "0.00"); "    ";
FORMAT$(periodv(1), "0.00"); "    "; FORMAT$(periodv(2), "0.00"); "    "; FORMAT$(periodv(3),
"0.00")
PRINTER.PRINT
PRINTER.PRINT "    1.3  PAVEMENT DATA"
PRINTER.PRINT "    -----------------"
PRINTER.PRINT "    CRACK INDEX =              "; CRACK
PRINTER.PRINT "    CLIMATIC ZONE :            "; ZON
PRINTER.PRINT "    TEMPERATURE AT FWD TEST     = "; TESTTEMP
PRINTER.PRINT "    OLD AC BITUMEN VOLUME (Vb) % = "; OLDBAIR(0)
PRINTER.PRINT "    OLD AC AIR VOLUME (Va) %    = "; OLDBAIR(1)
PRINTER.PRINT
PRINTER.PRINT "                    MODULUS   POISSON   THICKNESS"
PRINTER.PRINT "                     (ksi)     RATIO      (in.)"
PRINTER.PRINT "    OLD AC LAYER "; "      "; FORMAT$(E(1), "0000.00"); "     "; FORMAT$(V(1),
"0.00"); "     "; FORMAT$(HH(1), "00.00")
IF NLAYER > 3 THEN
PRINTER.PRINT "    BASE LAYER   "; "      "; FORMAT$(E(2), "0000.00"); "     "; FORMAT$(V(2),
"0.00"); "     "; FORMAT$(HH(2), "00.00")
END IF
IF NLAYER > 4 THEN
PRINTER.PRINT "    SUBBASE LAYER"; "      "; FORMAT$(E(3), "0000.00"); "     ";
FORMAT$(V(3), "0.00"); "     "; FORMAT$(HH(3), "00.00")
END IF
PRINTER.PRINT "    SUBGRADE     "; "      "; FORMAT$(E(NLAYER - 1), "0000.00"); "     ";
FORMAT$(V(NLAYER - 1), "0.00"); "     "; "SEMI-INFINITE"
PRINTER.PRINT
IF CODE(NLAYER - 1) = 0 THEN
PRINTER.PRINT "    SUBGRADE TYPE : LINEAR"
ELSEIF CODE(NLAYER - 1) = 2 THEN
PRINTER.PRINT "    SUBGRADE TYPE : GRANULAR"
PRINTER.PRINT "    K1(ksi) ="; KSUG(0); "   "; " K2 = "; KSUG(1)
ELSEIF CODE(NLAYER - 1) = 3 THEN
PRINTER.PRINT "    SUBGRADE TYPE : FINE"
PRINTER.PRINT "    K1(ksi) ="; KSUG(0); "   "; " K2 = "; KSUG(1)
END IF
PRINTER.PRINT
IF NLAYER > 3 THEN
IF CODE(2) = 0 THEN
PRINTER.PRINT "    BASE TYPE : LINEAR"
ELSEIF CODE(2) = 2 THEN
PRINTER.PRINT "    BASE TYPE : GRANULAR"
PRINTER.PRINT "    K1(ksi) ="; kbase(0); "   "; " K2 = "; kbase(1)
ELSEIF CODE(2) = 4 THEN
PRINTER.PRINT "    BASE TYPE : CEMENT TREATED"
ELSEIF CODE(2) = 5 THEN
PRINTER.PRINT "    BASE TYPE : BITUMEN TREATED"
PRINTER.PRINT "    BITUMEN VOLUME (Vb) % ="; kbase(0); "   "; "AIR VOLUME (Va) % = ";
kbase(1)
END IF
END IF
```

```
IF NLAYER > 4 THEN
IF CODE(3) = 0 THEN
PRINTER.PRINT "    SUB-BASE TYPE : LINEAR"
ELSEIF CODE(3) = 2 THEN
PRINTER.PRINT "    SUB-BASE TYPE : GRANULAR"
PRINTER.PRINT "    K1(ksi) ="; KSBASE(0); "   "; " K2 = "; KSBASE(1)
END IF
END IF
PRINTER.PRINT
PRINTER.PRINT "    OVERLAY MODULUS(ksi)     ="; E(0); " AT TEMPERATURE (F) =";
OLAYTEMP
PRINTER.PRINT "    POISSON RATIO       ="; V(0)
PRINTER.PRINT "    MINIMUM THICKNESS    ="; HH(0)
PRINTER.PRINT "    BITUMEN VOLUME (Vb) % ="; BITAIR(0); "   "; "AIR VOLUME (Va) % = ";
BITAIR(1)
PRINTER.PRINT
PRINTER.PRINT
PRINTER.NEWPAGE
PRINTER.PRINT
PRINTER.PRINT
PRINTER.PRINT
PRINTER.PRINT "    PAGE 2/2"
PRINTER.PRINT
PRINTER.PRINT
PRINTER.PRINT
PRINTER.PRINT "    2. SUMMARY OF RESULTS "
PRINTER.PRINT "    -------------------- "
PRINTER.PRINT
PRINTER.PRINT "    2.1  EVALUATION OF LAYER MDDULI VALUES FOR EACH SEASON"
PRINTER.PRINT
IF NLAYER = 3 THEN
PRINTER.PRINT "    SEASON   OVERLAY   OLD AC   SUBGRADE"
ELSEIF NLAYER = 4 THEN
PRINTER.PRINT "    SEASON   OVERLAY   OLD AC   BASE     SUBGRDE"
ELSE
PRINTER.PRINT "    SEASON   OVERLAY   OLD AC   BASE     SUBBASE   SUBGRADE"
END IF
OPEN "RESULTS.OUT" FOR INPUT AS #5
FOR I = 1 TO 4
PRINTER.PRINT "      "; I; "    ";
FOR J = 1 TO NLAYER
INPUT #5, MODULUS(I, J)
PRINTER.PRINT FORMAT$(MODULUS(I, J) / 1000, "0000.00   ");
NEXT J
PRINTER.PRINT
NEXT I
CLOSE #5
PRINTER.PRINT
PRINTER.PRINT "    2.2  OVERLAY THICKNESS AND DAMAGE ANALYSIS"
PRINTER.PRINT
IF OLAYT < .5 THEN
PRINTER.PRINT "    AN OVERLAY MAY NOT BE REQUIRED"
PRINTER.PRINT "    FATIGUE DAMAGE DUE PAST TRAFFIC = "; FORMAT$(DAMA22,
"00.0000")
```

```
PRINTER.PRINT "    FATIGUE DAMAGE ON OLD AC       = "; FORMAT$(DAMA2, "00.0000")
PRINTER.PRINT "    RUTTUING DAMAGE                = "; FORMAT$(DAMA4, "00.0000")
GOTO 107
END IF
PRINTER.PRINT "    FINAL OVERLAY THICKNESS        = "; FORMAT$(OLAYT, "00.00")
PRINTER.PRINT "    FATIGUE DAMAGE DUE PAST TRAFFIC = "; FORMAT$(DAMA22,
"00.0000")
PRINTER.PRINT "    FATIGUE DAMAGE ON OVERLAY      = "; FORMAT$(DAMA1, "00.0000")
IF DAMA22 > 1 THEN
PRINTER.PRINT "    RUTTUNIG DAMAGE                = "; FORMAT$(DAMA4, "00.0000")
GOTO 107
END IF
PRINTER.PRINT "    FATIGUE DAMAGE ON OLD AC       = "; FORMAT$(DAMA2, "00.0000")
IF ((NLAYER = 4 OR NLAYER = 5) AND CODE(2) = 5) THEN PRINTER.PRINT "    FATIGUE
DAMAGE ON BTB          = "; FORMAT$(DAMA3, "00.0000")
PRINTER.PRINT "    RUTTUNIG DAMAGE                = "; FORMAT$(DAMA4, "00.0000")
PRINTER.PRINT
107 OPEN "LIFE.OUT" FOR INPUT AS #7
PRINTER.PRINT
IT = 3
IF DAMA22 > 1 OR OLAYT < .5 THEN
IT = 2
GOTO 109
END IF
IF NLAYER = 4 AND CODE(NLAYER - 2) = 5 THEN IT = 4
IF NLAYER = 5 AND CODE(NLAYER - 3) = 5 THEN IT = 4
109 FOR I = 1 TO 4
PRINTER.PRINT "    SEASON"; I
FOR AZ = 1 TO IT
INPUT #7, ZZ(AZ), STRAIN(AZ), LIFE(AZ)
SELECT CASE AZ
CASE 1
IF OLAYT < .5 THEN
PRINTER.PRINT "    OLD AC  (B)"; " "; FORMAT$(ZZ(AZ), "00.00"); "  "; "RAD.  STR. MICRO ";
FORMAT$(STRAIN(AZ), "0000.00"); "   "; " Nf "; LIFE(AZ)
ELSE
PRINTER.PRINT "    OVERLAY (B)"; " "; FORMAT$(ZZ(AZ), "00.00"); "  "; "RAD.  STR. MICRO ";
FORMAT$(STRAIN(AZ), "0000.00"); "   "; " Nf "; LIFE(AZ)
END IF
CASE 2
IF IT = 2 THEN
PRINTER.PRINT "    SUBGRADE(T)"; " "; FORMAT$(ZZ(AZ), "00.00"); "  "; "COMP. STR. MICRO
"; FORMAT$(STRAIN(AZ), "0000.00"); "  "; " Nd "; LIFE(AZ)
GOTO 110
END IF
PRINTER.PRINT "    OLD AC  (B)"; " "; FORMAT$(ZZ(AZ), "00.00"); "  "; "RAD.  STR. MICRO ";
FORMAT$(STRAIN(AZ), "0000.00"); "   "; " Nf "; LIFE(AZ)
CASE 3
IF IT = 3 THEN
PRINTER.PRINT "    SUBGRADE(T)"; " "; FORMAT$(ZZ(AZ), "00.00"); "  "; "COMP. STR. MICRO
"; FORMAT$(STRAIN(AZ), "0000.00"); "  "; " Nd "; LIFE(AZ)
GOTO 110
END IF
```

```
PRINTER.PRINT "    BTB    (B)"; " "; FORMAT$(ZZ(AZ), "00.00"); "  "; "RAD.  STR. MICRO ";
FORMAT$(STRAIN(AZ), "0000.00"); "   "; " Nf "; LIFE(AZ)
CASE 4
PRINTER.PRINT "    SUBGRADE(B)"; " "; FORMAT$(ZZ(AZ), "00.00"); "  "; "COMP. STR. MICRO
"; FORMAT$(STRAIN(AZ), "0000.00"); "  "; " Nd "; LIFE(AZ)
END SELECT
110 NEXT AZ
PRINTER.PRINT
111 NEXT I
CLOSE
NEXT P
112 EXIT SUB
printerr:
MSGBOX ("Printer error..please check printer"), 0, "Print"
RESUME 112
END SUB

SUB MNURESRUN_Click ()
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), CRACK, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2,
DAMA3, DAMA4, DAMA22
SHARED RESPONSE%
SHARED LT
SHARED INPFILE$
LT = 0
UNLOAD FRMResult
OPEN "PAVE.DAT" FOR OUTPUT AS #1
GALAT = 0
IF OLAYINC <= 0 THEN
GALAT = 1 + GALAT
MSG$ = "Ivalid overlay thickness increment"
CALL INPUTERROR(MSG$)
IF RESPONSE% = 2 THEN GOTO 68
END IF
PRINT #1, OLAYINC
PRINT #1, TESTTEMP; OLAYTEMP
FOR I = 0 TO 6
IF WPSI(I) <= 0 THEN
IF I = 1 OR I = 4 THEN
IF WPSI(I) = 0 THEN GOTO 125
END IF
GALAT = GALAT + 1
SELECT CASE I
CASE 0
MSG$ = "INVALID DUAL TIRE LOAD"
CASE 1
MSG$ = "INVALID TIRE SPACING"
CASE 2
MSG$ = "INVALID TIRE PRESSURE "
CASE 3
MSG$ = "INVALID FUTURE ESAL"
CASE 4
```

```
MSG$ = "INVALID PAST ESAL"
CASE 5
MSG$ = "INVALID SHIFT FACTOR"
END SELECT
CALL INPUTERROR(MSG$)
IF RESPONSE% = 2 THEN GOTO 68
END IF
125 PRINT #1, WPSI(I);
NEXT I
PRINT #1,
FOR I = 0 TO 3
IF SUGV(I) <= 0 THEN
GALAT = GALAT + 1
SELECT CASE I
CASE 0
MSG$ = " INVAILD SPRING SUBGRADE ADJUSTMENT COEFFICIENT "
CASE 1
MSG$ = " INVAILD SUMMER SUBGRADE ADJUSTMENT COEFFICIENT "
CASE 2
MSG$ = " INVAILD FALL SUBGRADE ADJUSTMENT COEFFICIENT "
CASE 3
MSG$ = " INVAILD WINTER SUBGRADE ADJUSTMENT COEFFICIENT "
END SELECT
CALL INPUTERROR(MSG$)
IF RESPONSE% = 2 THEN GOTO 68
END IF
PRINT #1, SUGV(I);
NEXT I
PRINT #1,
FOR I = 0 TO 3
IF NLAYER = 3 THEN GOTO 89
IF BASEV(I) <= 0 THEN
GALAT = GALAT + 1
SELECT CASE I
CASE 0
MSG$ = " INVAILD SPRING BASE ADJUSTMENT COEFFICIENT "
CASE 1
MSG$ = " INVAILD SUMMER BASE ADJUSTMENT COEFFICIENT "
CASE 2
MSG$ = " INVAILD FALL BASE ADJUSTMENT COEFFICIENT "
CASE 3
MSG$ = " INVAILD WINTER BASE ADJUSTMENT COEFFICIENT "
END SELECT
CALL INPUTERROR(MSG$)
IF RESPONSE% = 2 THEN GOTO 68
END IF
89 PRINT #1, BASEV(I);
NEXT I
PRINT #1,
FOR I = 0 TO 3
IF TRAFICV(I) <= 0 THEN
GALAT = GALAT + 1
SELECT CASE I
CASE 0
```

```
MSG$ = " INVAILD SPRING TRAFFIC VARIATION "
CASE 1
MSG$ = " INVAILD SUMMER TRAFFIC VARIATION "
CASE 2
MSG$ = " INVAILD AUTUMN TRAFFIC VARIATION "
CASE 3
MSG$ = " INVAILD WINTER TRAFFIC VARIATION "
END SELECT
CALL INPUTERROR(MSG$)
IF RESPONSE% = 2 THEN GOTO 68
END IF
PRINT #1, TRAFICV(I);
NEXT I
PRINT #1,
FOR I = 0 TO 3
PRINT #1, TEMPV(I);
NEXT I
PRINT #1,
FOR I = 0 TO 3
IF periodv(I) < 0 THEN
GALAT = GALAT + 1
SELECT CASE I
CASE 0
MSG$ = " INVAILD SPRING PERIOD "
CASE 1
MSG$ = " INVAILD SUMMER PERIOD "
CASE 2
MSG$ = " INVAILD AUTUMN AUTUMN "
CASE 3
MSG$ = " INVAILD WINTER PERIOD "
END SELECT
CALL INPUTERROR(MSG$)
IF RESPONSE% = 2 THEN GOTO 68
END IF
PRINT #1, periodv(I);
NEXT I
PERIOD = periodv(0) + periodv(1) + periodv(2) + periodv(3)
IF PERIOD = 0 THEN
MSG$ = "INVALID TOTAL NUMBER OF PERIODS"
CALL INPUTERROR(MSG$)
END IF
PRINT #1,
PRINT #1, NLAYER
FOR I = 0 TO NLAYER - 1
IF E(I) <= 0 THEN
GALAT = GALAT + 1
SELECT CASE I
CASE 0
MSG$ = "INVALID OVERLAY MODULUS"
CASE 1
MSG$ = "INVALID OLD ASPHALT MODULUS"
CASE 2
IF NLAYER = 3 THEN
MSG$ = "INVALID SUBGRADE MODULUS"
```

```
ELSE
MSG$ = "INVALID BASE MODULUS"
END IF
CASE 3
IF NLAYER = 4 THEN
MSG$ = "INVALID SUBGRAD MODULUS"
ELSE
MSG$ = "INVALID SUBBASE MODULUS"
END IF
CASE 4
MSG$ = "INVALID SUBGRDAE MODULUS"
END SELECT
CALL INPUTERROR(MSG$)
IF RESPONSE% = 2 THEN GOTO 68
IF V(I) <= 0 OR V(I) > .8 THEN
GALAT = GALAT + 1
SELECT CASE I
CASE 0
MSG$ = "INVALID OVERLAY POISSON RATIO"
CASE 1
MSG$ = "INVALID OLD ASPHALT POISSON RATIO"
CASE 2
IF NLAYER = 3 THEN
MSG$ = "INVALID SUBGRADE POISSON RATIO"
ELSE
MSG$ = "INVALID BASE POISSON RATIO"
END IF
CASE 3
IF NLAYER = 4 THEN
MSG$ = "INVALID SUBGRAD POISSON RATIO"
ELSE
MSG$ = "INVALID SUBBASE POISSON RATIO"
END IF
CASE 4
MSG$ = "INVALID SUBGRDAE POISSON RATIO"
END SELECT
CALL INPUTERROR(MSG$)
END IF
END IF
PRINT #1, E(I); V(I)
NEXT I
IF NLAYER = 3 THEN
PRINT #1, CODE(2)
END IF
IF NLAYER = 4 THEN
PRINT #1, CODE(2)
PRINT #1, CODE(3)
END IF
IF NLAYER = 5 THEN
PRINT #1, CODE(2)
PRINT #1, CODE(3)
PRINT #1, CODE(4)
END IF
IF NLAYER = 3 THEN
```

```
FOR I = 0 TO 1
PRINT #1, KSUG(I);
NEXT I
PRINT #1,
END IF
IF NLAYER = 4 THEN
FOR I = 0 TO 1
PRINT #1, kbase(I);
NEXT I
PRINT #1,
FOR I = 0 TO 1
PRINT #1, KSUG(I);
NEXT I
PRINT #1,
END IF
IF NLAYER = 5 THEN
FOR I = 0 TO 1
PRINT #1, kbase(I);
NEXT I
PRINT #1,
FOR I = 0 TO 1
PRINT #1, KSBASE(I);
NEXT I
PRINT #1,
FOR I = 0 TO 1
PRINT #1, KSUG(I);
NEXT I
PRINT #1,
END IF
FOR I = 0 TO NLAYER - 2
IF HH(I) <= 0 THEN
GALAT = GALAT + 1
SELECT CASE I
CASE 0
MSG$ = "INVALID OVERLAY THICKNESS"
CASE 1
MSG$ = "INVALID OLD ASPHALT THICKNESS"
CASE 2
IF NLAYER > 3 THEN
MSG$ = "INVALID BASE THICKNESS "
END IF
CASE 3
IF NLAYER = 5 THEN
MSG$ = "INVALID SUBBASE THICKNESS"
END IF
END SELECT
CALL INPUTERROR(MSG$)
IF RESPONSE% = 2 THEN GOTO 68
END IF
PRINT #1, HH(I)
NEXT I
IF OLDBAIR(0) <= 3 OR OLDBAIR(0) > 20 THEN
GALAT = GALAT + 1
MSG$ = "INVALID OLD AC BITUMEN VOLUME (Vb %)"
```

```
CALL INPUTERROR(MSG$)
IF RESPONSE% = 2 THEN GOTO 68
END IF
IF OLDBAIR(1) <= 3 OR OLDBAIR(1) > 5 THEN
GALAT = GALAT + 1
MSG$ = "INVALID OLD AC AIR VOLUME (Va %)"
CALL INPUTERROR(MSG$)
IF RESPONSE% = 2 THEN GOTO 68
END IF
PRINT #1, OLDBAIR(0), OLDBAIR(1)
IF BITAIR(0) <= 3 OR BITAIR(0) > 20 THEN
GALAT = GALAT + 1
MSG$ = "INVALID OVERLAY BITUMEN CONTENT(Vb %)"
CALL INPUTERROR(MSG$)
IF RESPONSE% = 2 THEN GOTO 68
END IF
IF BITAIR(1) <= 3 OR BITAIR(1) > 5 THEN
GALAT = GALAT + 1
MSG$ = "INVALID OVERLAY AIR CONTENT (Va %)"
CALL INPUTERROR(MSG$)
IF RESPONSE% = 2 THEN GOTO 68
END IF
PRINT #1, BITAIR(0), BITAIR(1)
IF CRACK > 5 OR CRACK < 0 THEN
GALAT = GALAT + 1
MSG$ = "INVALID CRACK INDEX)"
CALL INPUTERROR(MSG$)
IF RESPONSE% = 2 THEN GOTO 68
END IF
PRINT #1, CRACK
PRINT #1, ZON
PRINT #1, Z, K, G
FOR I = 0 TO 2
PRINT #1, OPTIONS(I)
NEXT I
FOR I = 0 TO 6
PRINT #1, ZONES(I)
NEXT I
PRINT #1, DSCRPTION$
PRINT #1, FILE$
68 CLOSE
IF GALAT > 0 THEN
KILL "PAVE.DAT"
GOTO 67
END IF
RUNMSG.TOP = 7
RUNMSG.HEIGHT = 9
RUNMSG.TEXT1.VISIBLE = 0
RUNMSG.LABEL3.VISIBLE = 0
RUNMSG.TEXT2.VISIBLE = 0
RUNMSG.LABEL4.VISIBLE = 0
CALL mnurunmsg_click
OLAYTMP$ = "OLAY.TMP"
FILETMP$ = DIR$(OLAYTMP$)
```

```
IF FILETMP$ = OLAYTMP$ THEN KILL OLAYTMP$
SHELL ("SYSAN.exe")
UNLOAD RUNMSG
ON LOCAL ERROR GOTO RUNERR
OPEN "OLAY.TMP" FOR INPUT AS #8
INPUT #8, OLAYT, DAMA1, DAMA2, DAMA3, DAMA4, DAMA22
SHOWPRN = -1
FOR I = 1 TO NLAYER - 1
FRMPAVE.TEXT1(I).ENABLED = -1
NEXT I
FRMPAVE.TEXT5.ENABLED = -1
FRMPAVE.SHOW
67 EXIT SUB
RUNERR:
ERRORNUM% = ERR
ERRORLINE% = ERL
MSG$ = "RUNTIME ERROR....CHECK INPUT DATA"
MSGBOX MSG$, 0, "RUN ERROR"
FILELOG% = FREEFILE
LOGEXT$ = "LOG"
LOGFILE$ = MID$(INPFILE$, 1, L - 3) + LOGEXT$
OPEN LOGFILE$ FOR OUTPUT AS #FILELOG%
PRINT #FILELOG%, ERRORNUM%
PRINT #FILELOG%, ERRORLINE%
CLOSE #FILELOG%
RESUME 67
END SUB

SUB MNURESSHOW_Click ()
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), CRACK, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2,
DAMA3, DAMA4, DAMA22
SHARED LT
IF LT = 1 THEN
FRMOUTPUT.SHOW
ELSE
UNLOAD FRMGEN
UNLOAD frmmat
UNLOAD FRMPAVE
FRMResult.SHOW
END IF
END SUB

SUB MNURES_Click ()
UNLOAD frmpage
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), CRACK, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2,
DAMA3, DAMA4, DAMA22
IF SHOWPRN = 0 THEN
MNURESSHOW.ENABLED = -1
```

```
MNURESPRN.ENABLED = -1
ELSE
MNURESSHOW.ENABLED = -1
MNURESPRN.ENABLED = -1
END IF
END SUB

SUB mnurunfile_Click ()
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), CRACK, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK,
savxit
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2,
DAMA3, DAMA4, DAMA22
SHARED CANCL
SHARED RESPONSE%
SHARED INPFILE$
SHARED LT
SHARED NETF, HEADER$
LT = 1
IF FRMINPUT.MANUALLY.VALUE = -1 THEN
GOTO 20
END IF
 FILEINP% = FREEFILE
 FileOpen FILENAME$, PATHNAME$, "*.ETF", "OPEN", 0, 7, 0, Cancel%
IF Cancel% = -1 THEN GOTO 76
OLDFILE$ = FILE$
OLDPATH$ = PATH$
IF RIGHT$(PATHNAME$, 1) = "\" THEN
PATHNAME$ = LEFT$(PATHNAME$, LEN(PATHNAME$) - 1)
END IF
FILE$ = FILENAME$
PATH$ = PATHNAME$
CAPTION = "MAIN : " + PATH$ + "\" + FILE$
OPEN PATHNAME$ + "\" + FILENAME$ FOR INPUT AS #FILEINP%
ETFFILE$ = FILENAME$
INPUT #FILEINP%, NETF, HEADER$
I = 0
DO WHILE NOT EOF(FILEINP%)
  I = I + 1
  INPUT #FILEINP%, JUNK
LOOP
SIZE = I / (NLAYER + 1)
SIZE1 = SIZE + .5
IF SIZE1 <> INT(SIZE) + .5 THEN
  MSG$ = "PLEASE CHECK THE .ETF FILE FORMAT"
  CALL INPUTERROR1(MSG$)
  IF RESPONSE% = 1 THEN GOTO 76
END IF
IF (NETF + 1) <> NLAYER THEN
  MSG$ = "PLEASE CHECK THE .ETF FILE FORMAT"
  CALL INPUTERROR1(MSG$)
  IF RESPONSE% = 1 THEN GOTO 76
END IF
```

```
CLOSE #FILEINP%
DIM RFE(NLAYER - 1, SIZE), STATION(SIZE)
DIM TEMP(SIZE)
UNLOAD FRMResult
OPEN PATHNAME$ + "\" + FILENAME$ FOR INPUT AS #FILEINP%
INPUT #FILEINP%, NETF, HEADER$
FOR J = 1 TO SIZE
INPUT #FILEINP%, TEMP(J), STATION(J)
IF NLAYER = 5 THEN
INPUT #FILEINP%, RFE(1, J), RFE(2, J), RFE(3, J), RFE(4, J)
END IF
IF NLAYER = 4 THEN
INPUT #FILEINP%, RFE(1, J), RFE(2, J), RFE(3, J)
END IF
IF NLAYER = 3 THEN
INPUT #FILEINP%, RFE(1, J), RFE(2, J)
END IF
NEXT J
CLOSE #FILEINP%
FILEPAVE% = FREEFILE
FOR J = 1 TO SIZE
OPEN "PAVE.DAT" FOR OUTPUT AS #FILEPAVE%
GALAT = 0
IF OLAYINC <= 0 THEN
  GALAT = 1 + GALAT
  MSG$ = "Ivalid overlay thickness increment"
  CALL INPUTERROR(MSG$)
  IF RESPONSE% = 2 THEN GOTO 86
END IF
PRINT #FILEPAVE%, OLAYINC
PRINT #FILEPAVE%, TEMP(J); OLAYTEMP
FOR I = 0 TO 6
  IF WPSI(I) <= 0 THEN
  IF I = 1 OR I = 4 THEN
  IF WPSI(I) = 0 THEN GOTO 152
END IF
GALAT = GALAT + 1
SELECT CASE I
  CASE 0
  MSG$ = "INVALID DUAL TIRE LOAD"
  CASE 1
  MSG$ = "INVALID TIRE SPACING"
  CASE 2
  MSG$ = "INVALID TIRE PRESSURE "
  CASE 3
  MSG$ = "INVALID FUTURE ESAL"
  CASE 4
  MSG$ = "INVALID PAST ESAL"
  CASE 5
  MSG$ = "INVALID SHIFT FACTOR"

END SELECT
CALL INPUTERROR(MSG$)
IF RESPONSE% = 2 THEN GOTO 86
```

```
END IF
152 PRINT #FILEPAVE%, WPSI(I);
NEXT I
PRINT #FILEPAVE%,
FOR I = 0 TO 3
  IF SUGV(I) <= 0 THEN
  GALAT = GALAT + 1
  SELECT CASE I
    CASE 0
    MSG$ = " INVAILD SPRING SUBGRADE ADJUSTMENT COEFFICIENT "
    CASE 1
    MSG$ = " INVAILD SUMMER SUBGRADE ADJUSTMENT COEFFICIENT "
    CASE 2
    MSG$ = " INVAILD FALL SUBGRADE ADJUSTMENT COEFFICIENT "
    CASE 3
    MSG$ = " INVAILD WINTER SUBGRADE ADJUSTMENT COEFFICIENT "
  END SELECT
  CALL INPUTERROR(MSG$)
  IF RESPONSE% = 2 THEN GOTO 86
  END IF
  PRINT #FILEPAVE%, SUGV(I);
NEXT I
PRINT #FILEPAVE%,
FOR I = 0 TO 3
  IF NLAYER = 3 THEN GOTO 98
  IF BASEV(I) <= 0 THEN
  GALAT = GALAT + 1
  SELECT CASE I
    CASE 0
    MSG$ = " INVAILD SPRING BASE ADJUSTMENT COEFFICIENT "
    CASE 1
    MSG$ = " INVAILD SUMMER BASE ADJUSTMENT COEFFICIENT "
    CASE 2
    MSG$ = " INVAILD FALL BASE ADJUSTMENT COEFFICIENT "
    CASE 3
    MSG$ = " INVAILD WINTER BASE ADJUSTMENT COEFFICIENT "
  END SELECT
  CALL INPUTERROR(MSG$)
  IF RESPONSE% = 2 THEN GOTO 86
  END IF
98 PRINT #FILEPAVE%, BASEV(I);
NEXT I
PRINT #FILEPAVE%,
FOR I = 0 TO 3
  IF TRAFICV(I) <= 0 THEN
  GALAT = GALAT + 1
  SELECT CASE I
    CASE 0
    MSG$ = " INVAILD SPRING TRAFFIC VARIATION "
    CASE 1
    MSG$ = " INVAILD SUMMER TRAFFIC VARIATION "
    CASE 2
    MSG$ = " INVAILD AUTUMN TRAFFIC VARIATION "
    CASE 3
```

```
    MSG$ = " INVAILD WINTER TRAFFIC VARIATION "
  END SELECT
  CALL INPUTERROR(MSG$)
  IF RESPONSE% = 2 THEN GOTO 86
  END IF
  PRINT #FILEPAVE%, TRAFICV(I);
NEXT I
PRINT #FILEPAVE%,
FOR I = 0 TO 3
  PRINT #FILEPAVE%, TEMPV(I);
NEXT I
PRINT #FILEPAVE%,
FOR I = 0 TO 3
  IF periodv(I) < 0 THEN
  GALAT = GALAT + 1
  SELECT CASE I
    CASE 0
    MSG$ = " INVAILD SPRING PERIOD "
    CASE 1
    MSG$ = " INVAILD SUMMER PERIOD "
    CASE 2
    MSG$ = " INVAILD AUTUMN AUTUMN "
    CASE 3
    MSG$ = " INVAILD WINTER PERIOD "
  END SELECT
  CALL INPUTERROR(MSG$)
  IF RESPONSE% = 2 THEN GOTO 86
  END IF
  PRINT #FILEPAVE%, periodv(I);
NEXT I
PERIOD = periodv(0) + periodv(1) + periodv(2) + periodv(3)
IF PERIOD = 0 THEN
  MSG$ = "INVALID TOTAL NUMBER OF PERIODS"
  CALL INPUTERROR(MSG$)
END IF
PRINT #FILEPAVE%,
PRINT #FILEPAVE%, NLAYER
PRINT #FILEPAVE%, E(0); V(0)
FOR I = 1 TO NLAYER - 1
  IF V(I) <= 0 OR V(I) > .8 THEN
    GALAT = GALAT + 1
    SELECT CASE I
      CASE 1
      MSG$ = "INVALID OLD ASPHALT POISSON RATIO"
      CASE 2
      IF NLAYER = 3 THEN
        MSG$ = "INVALID SUBGRADE POISSON RATIO"
        ELSE
        MSG$ = "INVALID BASE POISSON RATIO"
      END IF
      CASE 3
      IF NLAYER = 4 THEN
        MSG$ = "INVALID SUBGRAD POISSON RATIO"
        ELSE
```

```
      MSG$ = "INVALID SUBBASE POISSON RATIO"
      END IF
     CASE 4
     MSG$ = "INVALID SUBGRDAE POISSON RATIO"
    END SELECT
    CALL INPUTERROR(MSG$)
    END IF
    PRINT #FILEPAVE%, RFE(I, J); V(I)
NEXT I
IF NLAYER = 3 THEN
  PRINT #FILEPAVE%, CODE(2)
END IF
IF NLAYER = 4 THEN
  PRINT #FILEPAVE%, CODE(2)
  PRINT #FILEPAVE%, CODE(3)
END IF
IF NLAYER = 5 THEN
  PRINT #FILEPAVE%, CODE(2)
  PRINT #FILEPAVE%, CODE(3)
  PRINT #FILEPAVE%, CODE(4)
END IF
IF NLAYER = 3 THEN
  FOR I = 0 TO 1
    PRINT #FILEPAVE%, KSUG(I);
  NEXT I
  PRINT #FILEPAVE%,
END IF
IF NLAYER = 4 THEN
  FOR I = 0 TO 1
    PRINT #FILEPAVE%, kbase(I);
  NEXT I
  PRINT #FILEPAVE%,
  FOR I = 0 TO 1
    PRINT #FILEPAVE%, KSUG(I);
  NEXT I
  PRINT #FILEPAVE%,
END IF
IF NLAYER = 5 THEN
  FOR I = 0 TO 1
    PRINT #FILEPAVE%, kbase(I);
  NEXT I
  PRINT #FILEPAVE%,
  FOR I = 0 TO 1
    PRINT #FILEPAVE%, KSBASE(I);
  NEXT I
  PRINT #FILEPAVE%,
  FOR I = 0 TO 1
  PRINT #FILEPAVE%, KSUG(I);
  NEXT I
  PRINT #FILEPAVE%,
END IF
FOR I = 0 TO NLAYER - 2
  IF HH(I) <= 0 THEN
    GALAT = GALAT + 1
```

```
    SELECT CASE I
      CASE 0
      MSG$ = "INVALID OVERLAY THICKNESS"
      CASE 1
      MSG$ = "INVALID OLD ASPHALT THICKNESS"
      CASE 2
      IF NLAYER > 3 THEN
        MSG$ = "INVALID BASE THICKNESS "
      END IF
      CASE 3
      IF NLAYER = 5 THEN
        MSG$ = "INVALID SUBBASE THICKNESS"
      END IF
    END SELECT
    CALL INPUTERROR(MSG$)
    IF RESPONSE% = 2 THEN GOTO 86
    END IF
    PRINT #FILEPAVE%, HH(I)
NEXT I
PRINT #FILEPAVE%, OLDBAIR(0), OLDBAIR(1)
PRINT #FILEPAVE%, BITAIR(0), BITAIR(1)
IF CRACK > 5 OR CRACK < 0 THEN
  GALAT = GALAT + 1
  MSG$ = "INVALID CRACK INDEX)"
  CALL INPUTERROR(MSG$)
  IF RESPONSE% = 2 THEN GOTO 86
END IF
PRINT #FILEPAVE%, CRACK
PRINT #FILEPAVE%, ZON
PRINT #FILEPAVE%, Z, K, G
FOR I = 0 TO 2
  PRINT #FILEPAVE%, OPTIONS(I)
NEXT I
FOR I = 0 TO 6
  PRINT #FILEPAVE%, ZONES(I)
NEXT I
PRINT #FILEPAVE%, DSCRPTION$
PRINT #FILEPAVE%, FILE$
86 CLOSE
IF GALAT > 0 THEN
KILL "PAVE.DAT"
GOTO 76
END IF
RUNMSG.TEXT1.TEXT = STR$(J)
RUNMSG.TEXT2.TEXT = STR$(SIZE)
CALL mnurunmsg_click
SHELL ("SYSAN.exe")
UNLOAD RUNMSG
ON LOCAL ERROR GOTO RUNFILERR
IF J = 1 THEN
DIM OLTH(SIZE), DAM1(SIZE), DAM2(SIZE), DAM3(SIZE), DAM4(SIZE), DAM22(SIZE)
END IF
FILEOLAY% = FREEFILE
OPEN "OLAY.TMP" FOR INPUT AS #FILEOLAY%
```

```
INPUT #FILEOLAY%, OLTH(J), DAM1(J), DAM2(J), DAM3(J), DAM4(J), DAM22(J)
CLOSE #FILEOLAY%
CLOSE #FILEPAVE%
NEXT J
FILEOUT% = FREEFILE
L = LEN(ETFFILE$)
EXT$ = "FLX"
TFILE$ = MID$(ETFFILE$, 1, L - 3) + EXT$
OPEN TFILE$ FOR OUTPUT AS #FILEOUT%
PRINT #FILEOUT%,
PRINT #FILEOUT%,
PRINT #FILEOUT%, "    INPUT FILE : "; INPFILE$
PRINT #FILEOUT%,
PRINT #FILEOUT%, "    DESCRIPTION : "; DSCRPTION$
PRINT #FILEOUT%,
PRINT #FILEOUT%, "    1. SUMMARY OF INPUT DATA "
PRINT #FILEOUT%, "    ------------------------ "
PRINT #FILEOUT%,
PRINT #FILEOUT%, "    1.1  TRAFFIC DATA"
PRINT #FILEOUT%, "    ----------------"
FOR I = 0 TO 6
SELECT CASE I
CASE 0
PRINT #FILEOUT%, "    DESIGN DUAL TIRE LOAD              ="; WPSI(I)
CASE 1
PRINT #FILEOUT%, "    DESIGN DUAL TIRE SPACING           ="; WPSI(I)
CASE 2
PRINT #FILEOUT%, "    TIRE PRESSURE (psi)                ="; WPSI(I)
CASE 3
PRINT #FILEOUT%, "    DESIGN FUTURE TRAFFIC (ESALs)      ="; WPSI(I)
CASE 4
PRINT #FILEOUT%, "    ESTIMATED PAST TRAFFIC (ESALs)     ="; WPSI(I)
CASE 5
PRINT #FILEOUT%, "    FATIGUE SHIFT FACTOR FOR NEW ASPHALT  ="; WPSI(I)
CASE 6
PRINT #FILEOUT%, "    FATIGUE SHIFT FACTOR FOR OLD ASPHALT  ="; WPSI(I)
END SELECT
NEXT I
PRINT #FILEOUT%,
PRINT #FILEOUT%, "    1.2  SEASONAL VARIATION DATA "
PRINT #FILEOUT%, "    --------------------------- "
PRINT #FILEOUT%, "                          WINTER  SPRING  SUMMER  FALL"
PRINT #FILEOUT%, "    SUBGRADE VARIATION   "; "   "; FORMAT$(SUGV(0), "0.00"); "    ";
FORMAT$(SUGV(1), "0.00"); "    "; FORMAT$(SUGV(2), "0.00"); "    "; FORMAT$(SUGV(3), "0.00")
PRINT #FILEOUT%, "    BASE/SBASE VARIATION "; "   "; FORMAT$(BASEV(0), "0.00"); "    ";
FORMAT$(BASEV(1), "0.00"); "    "; FORMAT$(BASEV(2), "0.00"); "    "; FORMAT$(BASEV(3),
"0.00")
PRINT #FILEOUT%, "    TRAFFIC VARIATION    "; "   "; FORMAT$(TRAFICV(0), "0.00"); "    ";
FORMAT$(TRAFICV(1), "0.00"); "    "; FORMAT$(TRAFICV(2), "0.00"); "    ";
FORMAT$(TRAFICV(3), "0.00")
PRINT #FILEOUT%, "    TEMPERATURE VARIATION"; "   "; FORMAT$(TEMPV(0), "0.00"); "   ";
FORMAT$(TEMPV(1), "0.00"); "   "; FORMAT$(TEMPV(2), "0.00"); "   "; FORMAT$(TEMPV(3),
"0.00")
```

```
PRINT #FILEOUT%, "    PERIOD (MONTHS)     "; "   "; FORMAT$(periodv(0), "0.00"); "    ";
FORMAT$(periodv(1), "0.00"); "    "; FORMAT$(periodv(2), "0.00"); "    "; FORMAT$(periodv(3),
"0.00")
PRINT #FILEOUT%,
PRINT #FILEOUT%, "    1.3  PAVEMENT DATA"
PRINT #FILEOUT%, "    -----------------"
PRINT #FILEOUT%, "    CRACK INDEX =              "; CRACK
PRINT #FILEOUT%, "    CLIMATIC ZONE :            "; ZON
PRINT #FILEOUT%, "    TEMPERATURE AT FWD TEST     = "; TESTTEMP
PRINT #FILEOUT%, "    OLD AC BITUMEN VOLUME (Vb) % = "; OLDBAIR(0)
PRINT #FILEOUT%, "    OLD AC AIR VOLUME (Va) %    = "; OLDBAIR(1)
PRINT #FILEOUT%,
PRINT #FILEOUT%, "                    POISSON   THICKNESS"
PRINT #FILEOUT%, "                    RATIO     (in.)"
PRINT #FILEOUT%, "    OLD AC LAYER "; "      "; FORMAT$(V(1), "0.00"); "      ";
FORMAT$(HH(1), "00.00")
IF NLAYER > 3 THEN
PRINT #FILEOUT%, "    BASE LAYER   "; "      "; FORMAT$(V(2), "0.00"); "      ";
FORMAT$(HH(2), "00.00")
END IF
IF NLAYER > 4 THEN
PRINT #FILEOUT%, "    SUBBASE LAYER"; "      "; FORMAT$(V(3), "0.00"); "      ";
FORMAT$(HH(3), "00.00")
END IF
PRINT #FILEOUT%, "    SUBGRADE     "; "      "; FORMAT$(V(NLAYER - 1), "0.00"); "      ";
"SEMI-INFINITE"
PRINT #FILEOUT%,
IF CODE(NLAYER - 1) = 0 THEN
PRINT #FILEOUT%, "    SUBGRADE TYPE : LINEAR"
ELSEIF CODE(NLAYER - 1) = 2 THEN
PRINT #FILEOUT%, "    SUBGRADE TYPE : GRANULAR"
PRINT #FILEOUT%, "    K1(ksi) ="; KSUG(0); "   "; " K2 = "; KSUG(1)
ELSEIF CODE(NLAYER - 1) = 3 THEN
PRINT #FILEOUT%, "    SUBGRADE TYPE : FINE"
PRINT #FILEOUT%, "    K1(ksi) ="; KSUG(0); "   "; " K2 = "; KSUG(1)
END IF
PRINT #FILEOUT%,
IF NLAYER > 3 THEN
IF CODE(2) = 0 THEN
PRINT #FILEOUT%, "    BASE TYPE : LINEAR"
ELSEIF CODE(2) = 2 THEN
PRINT #FILEOUT%, "    BASE TYPE : GRANULAR"
PRINT #FILEOUT%, "    K1(ksi) ="; kbase(0); "   "; " K2 = "; kbase(1)
ELSEIF CODE(2) = 4 THEN
PRINT #FILEOUT%, "    BASE TYPE : CEMENT TREATED"
ELSEIF CODE(2) = 5 THEN
PRINT #FILEOUT%, "    BASE TYPE : BITUMEN TREATED"
PRINT #FILEOUT%, "    BITUMEN VOLUME (Vb) % ="; kbase(0); "   "; "AIR VOLUME (Va) % = ";
kbase(1)
END IF
END IF
IF NLAYER > 4 THEN
IF CODE(3) = 0 THEN
PRINT #FILEOUT%, "    SUB-BASE TYPE : LINEAR"
```

```
ELSEIF CODE(3) = 2 THEN
PRINT #FILEOUT%, "    SUB-BASE TYPE : GRANULAR"
PRINT #FILEOUT%, "    K1(ksi) ="; KSBASE(0); "   "; " K2 = "; KSBASE(1)
END IF
END IF
PRINT #FILEOUT%,
PRINT #FILEOUT%, "    OVERLAY MODULUS(ksi)  ="; E(0); " AT TEMPERATURE (F) ="; 
OLAYTEMP
PRINT #FILEOUT%, "    POISSON RATIO         ="; V(0)
PRINT #FILEOUT%, "    MINIMUM THICKNESS     ="; HH(0)
PRINT #FILEOUT%, "    BITUMEN VOLUME (Vb) % ="; BITAIR(0); "   "; "AIR VOLUME (Va) % = 
"; BITAIR(1)
PRINT #FILEOUT%,
PRINT #FILEOUT%,
PRINT #FILEOUT%, "    ETF FILE : "; ETFFILE$
PRINT #FILEOUT%, "               "; HEADER$
PRINT #FILEOUT%,
IF NLAYER = 5 THEN
PRINT #FILEOUT%, " CASE", "MILE POST", "TEMPERATURE", "  E1  ", "  E2  ", "  E3  ", "  E4  ", 
"OVERLAY", "DAMA1", "DAMA2", "DAMA3", "DAMA4", "DAMA22"
PRINT #FILEOUT%, "----", "--------", "----------", "-----", "-----", "-----", "-----", "------", "----", "---
--", "----", "----", "-----"
FOR J = 1 TO SIZE
PRINT #FILEOUT%, J, STATION(J), TEMP(J), RFE(1, J), RFE(2, J), RFE(3, J), RFE(4, J), OLTH(J), 
DAM1(J), DAM2(J), DAM3(J), DAM4(J), DAM22(J)
NEXT J
END IF
IF NLAYER = 4 THEN
PRINT #FILEOUT%, " CASE", "MILE POST", "TEMPERATURE", "  E1  ", "  E2  ", "  E3  ", 
"OVERLAY", "DAMA1", "DAMA2", "DAMA3", "DAMA4", "DAMA22"
PRINT #FILEOUT%, "----", "--------", "----------", "-----", "-----", "-----", "------", "----", "----", "---
-", "----", "-----"
FOR J = 1 TO SIZE
PRINT #FILEOUT%, J, STATION(J), TEMP(J), RFE(1, J), RFE(2, J), RFE(3, J), OLTH(J), DAM1(J), 
DAM2(J), DAM3(J), DAM4(J), DAM22(J)
NEXT J
END IF
IF NLAYER = 3 THEN
PRINT #FILEOUT%, " CASE", "MILE POST", "TEMPERATURE", "  E1  ", "  E2  ", "OVERLAY", 
"DAMA1", "DAMA2", "DAMA3", "DAMA4", "DAMA22"
PRINT #FILEOUT%, "----", "--------", "----------", "-----", "-----", "------", "----", "----", "----", "----
", "-----"
FOR J = 1 TO SIZE
PRINT #FILEOUT%, J, STATION(J), TEMP(J), RFE(1, J), RFE(2, J), OLTH(J), DAM1(J), DAM2(J), 
DAM3(J), DAM4(J), DAM22(J)
NEXT J
END IF
PRINT #FILEOUT%,
PRINT #FILEOUT%,
PRINT #FILEOUT%, "  DAMA1  = FATIGUE DAMAGE ON OVERLAY"
PRINT #FILEOUT%, "  DAMA2  = FATIGUE DAMAGE ON OLD AC"
PRINT #FILEOUT%, "  DAMA3  = FATIGUE DAMAGE ON BTB"
PRINT #FILEOUT%, "  DAMA4  = RUTTING DAMAGE"
PRINT #FILEOUT%, "  DAMA22 = FATIGUE DAMAGE DUE TO PAST TRAFFIC"
```

```
CLOSE #FILEOUT%
FOR I = 1 TO NLAYER - 1
FRMPAVE.TEXT1(I).ENABLED = 0
FRMPAVE.TEXT1(I).TEXT = " N/A"
NEXT I
FRMPAVE.TEXT5.ENABLED = 0
FRMPAVE.TEXT5.TEXT = " N/A"
FRMPAVE.SHOW
76 EXIT SUB
RUNFILERR:
ERRORNUM% = ERR
ERRORLINE% = ERL
MSG$ = "RUNTIME ERROR....CHECK INPUT DATA"
MSGBOX MSG$, 0, "RUN ERROR"
FILELOG% = FREEFILE
OPEN "FLEXOLAY.LOG" FOR OUTPUT AS #FILELOG%
PRINT #FILELOG%, ERRORNUM%
PRINT #FILELOG%, ERRORLINE%
CLOSE #FILELOG%
RESUME 76
UNLOAD FRMINPUT
LOAD FRMPAVE
LOAD FRMGEN
FRMPAVE.SHOW
20 UNLOAD FRMINPUT
END SUB

SUB mnurunmsg_click ()
RUNMSG.SHOW
END SUB


DIM SHARED E(5), V(5), HH(5), OPTIONS(3), KSUG(2), kbase(2), KSBASE(2), ZONES(6),
SUBGRD(4)
DIM SHARED WPSI(7), SUGV(4), BASEV(4), TRAFICV(4), TEMPV(4), periodv(4), BITAIR(2),
MT(5), CODE(5), OLDBAIR(2)
COMMON E(), V(), HH(), WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), crack,
TXTCHANG, ZONES(), SUBGRD()
COMMON OPTIONS(), KSUG(), kbase(), KSBASE(), BITAIR(), CODE(), OLDBAIR(), MT(),
NLAYER, OLAYINC, TESTTEMP, OLAYTEMP
COMMON DSCRPTION$, A, B, AA, BB, K, Y, Z, R, T, G, ZON, SHOWPRN, PASTCHCK, OLAYT,
DAMA1, DAMA2, DAMA3, DAMA4, DAMA22


SUB Form_Load ()
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), crack, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2
G = G + 1
IF G = 1 THEN GOTO 85
IF CODE(NLAYER - 1) = 2 THEN
OPTION1(0).VALUE = 0
OPTION1(1).VALUE = -1
```

```
OPTION1(2).VALUE = 0
GOTO 80
ELSEIF CODE(NLAYER - 1) = 3 THEN
OPTION1(0).VALUE = -1
OPTION1(1).VALUE = 0
OPTION1(2).VALUE = 0
GOTO 80
ELSE
OPTION1(0).VALUE = 0
OPTION1(1).VALUE = 0
OPTION1(2).VALUE = -1
GOTO 80
END IF
80 IF OPTIONS(1) = -1 THEN
FRAME6.VISIBLE = 0
FRAME5.VISIBLE = -1
FRAME4.VISIBLE = -1
FOR I = 0 TO 1
R = I
T = I
TEXT1(I).TEXT = STR$(KSUG(I))
TEXT2(I).TEXT = STR$(kbase(I))
KSBASE(I) = 0
NEXT I
IF CODE(NLAYER - 2) = 2 THEN
OPTION2(0).VALUE = -1
OPTION2(1).VALUE = 0
OPTION2(2).VALUE = 0
OPTION2(3).VALUE = 0
GOTO 85
END IF
IF CODE(NLAYER - 2) = 0 THEN
OPTION2(0).VALUE = 0
OPTION2(1).VALUE = -1
OPTION2(2).VALUE = 0
OPTION2(3).VALUE = 0
GOTO 85
END IF
IF CODE(NLAYER - 2) = 4 THEN
OPTION2(0).VALUE = 0
OPTION2(1).VALUE = 0
OPTION2(2).VALUE = -1
OPTION2(3).VALUE = 0
GOTO 85
END IF
IF CODE(NLAYER - 2) = 5 THEN
OPTION2(0).VALUE = 0
OPTION2(1).VALUE = 0
OPTION2(2).VALUE = 0
OPTION2(3).VALUE = -1
GOTO 85
END IF
GOTO 85
END IF
```

```
IF OPTIONS(2) = -1 THEN
FRAME4.VISIBLE = -1
FRAME5.VISIBLE = 0
FRAME6.VISIBLE = 0
FOR I = 0 TO 1
R = I
T = I
TEXT1(I).TEXT = STR$(KSUG(I))
NEXT I
GOTO 85
END IF
IF OPTIONS(0) = -1 THEN
FRAME6.VISIBLE = -1
FRAME5.VISIBLE = -1
FRAME4.VISIBLE = -1
IF G = 1 THEN GOTO 85
FOR I = 0 TO 1
R = I
T = I
TEXT1(I).TEXT = STR$(KSUG(I))
TEXT2(I).TEXT = STR$(kbase(I))
TEXT3(I).TEXT = STR$(KSBASE(I))
NEXT I
IF CODE(NLAYER - 3) = 2 THEN
OPTION2(0).VALUE = -1
OPTION2(1).VALUE = 0
OPTION2(2).VALUE = 0
OPTION2(3).VALUE = 0
GOTO 82
END IF
IF CODE(NLAYER - 3) = 0 THEN
OPTION2(0).VALUE = 0
OPTION2(1).VALUE = -1
OPTION2(2).VALUE = 0
OPTION2(3).VALUE = 0
GOTO 82
END IF
IF CODE(NLAYER - 3) = 4 THEN
OPTION2(0).VALUE = 0
OPTION2(1).VALUE = 0
OPTION2(2).VALUE = -1
OPTION2(3).VALUE = 0
GOTO 82
END IF
IF CODE(NLAYER - 3) = 5 THEN
OPTION2(0).VALUE = 0
OPTION2(1).VALUE = 0
OPTION2(2).VALUE = 0
OPTION2(3).VALUE = -1
GOTO 82
END IF
82 IF CODE(NLAYER - 2) = 2 THEN
OPTION3(0).VALUE = -1
OPTION3(1).VALUE = 0
```

```
GOTO 85
END IF
IF CODE(NLAYER - 2) = 0 THEN
OPTION3(0).VALUE = 0
OPTION3(1).VALUE = -1
GOTO 85
END IF
END IF
85 R = 0
T = 1
END SUB

SUB Option1_Click (Index AS INTEGER)
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), crack, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2
IF OPTION1(0).VALUE = -1 THEN
TEXT1(0).VISIBLE = -1
TEXT1(1).VISIBLE = -1
LABEL1(0).CAPTION = "K1-F (ksi) "
LABEL2(0).CAPTION = "K2-F      "
CODE(NLAYER - 1) = 3
END IF
IF OPTION1(1).VALUE = -1 THEN
TEXT1(0).VISIBLE = -1
TEXT1(1).VISIBLE = -1
LABEL1(0).CAPTION = "K1-G (ksi) "
LABEL2(0).CAPTION = "K2-G      "
CODE(NLAYER - 1) = 2
END IF
IF OPTION1(2).VALUE = -1 THEN
TEXT1(0).VISIBLE = 0
TEXT1(1).VISIBLE = 0
TEXT1(0).TEXT = ""
TEXT1(1).TEXT = ""
LABEL1(0).CAPTION = " "
LABEL2(0).CAPTION = " "
CODE(NLAYER - 1) = 0
END IF
TXTCHANG = -1
END SUB

SUB Option2_Click (Index AS INTEGER)
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), crack, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2
IF OPTION2(0).VALUE = -1 THEN
TEXT2(0).VISIBLE = -1
TEXT2(1).VISIBLE = -1
LABEL1(1).VISIBLE = -1
LABEL2(1).VISIBLE = -1
LABEL1(1).CAPTION = "K1 (ksi)"
```

```
LABEL2(1).CAPTION = "K2"
IF NLAYER = 4 THEN
CODE(NLAYER - 2) = 2
END IF
IF NLAYER = 5 THEN
CODE(NLAYER - 3) = 2
END IF
GOTO 51
END IF
IF OPTION2(1).VALUE = -1 THEN
TEXT2(0).VISIBLE = 0
TEXT2(1).VISIBLE = 0
LABEL1(1).VISIBLE = 0
LABEL2(1).VISIBLE = 0
kbase(0) = 0
kbase(1) = 0
TEXT2(0).TEXT = ""
TEXT2(1).TEXT = ""
IF NLAYER = 4 THEN
CODE(NLAYER - 2) = 0
END IF
IF NLAYER = 5 THEN
CODE(NLAYER - 3) = 0
END IF
GOTO 51
END IF
IF OPTION2(2).VALUE = -1 THEN
kbase(0) = 0
kbase(1) = 0
TEXT2(0).TEXT = ""
TEXT2(1).TEXT = ""
TEXT2(0).VISIBLE = 0
TEXT2(1).VISIBLE = 0
LABEL1(1).VISIBLE = 0
LABEL2(1).VISIBLE = 0
IF NLAYER = 4 THEN
CODE(NLAYER - 2) = 4
END IF
IF NLAYER = 5 THEN
CODE(NLAYER - 3) = 4
END IF
GOTO 51
END IF
IF OPTION2(3).VALUE = -1 THEN
TEXT2(0).VISIBLE = -1
TEXT2(1).VISIBLE = -1
LABEL1(1).VISIBLE = -1
LABEL2(1).VISIBLE = -1
LABEL1(1).CAPTION = " Vb %"
LABEL2(1).CAPTION = " Va %"
IF NLAYER = 4 THEN
CODE(NLAYER - 2) = 5
END IF
IF NLAYER = 5 THEN
```

```
CODE(NLAYER - 3) = 5
END IF
GOTO 51
END IF
TXTCHANG = -1
51 END SUB

SUB Option3_Click (Index AS INTEGER)
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), crack, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2
IF OPTIONS(0) = 0 THEN GOTO 61
IF OPTION3(0).VALUE = -1 THEN
TEXT3(0).VISIBLE = -1
TEXT3(1).VISIBLE = -1
LABEL1(2).VISIBLE = -1
LABEL2(2).VISIBLE = -1
LABEL1(2).CAPTION = "K1 (ksi)"
LABEL2(2).CAPTION = "K2"
CODE(NLAYER - 2) = 2
END IF
IF OPTION3(1).VALUE = -1 THEN
TEXT3(0).VISIBLE = 0
TEXT3(1).VISIBLE = 0
LABEL1(2).VISIBLE = 0
LABEL2(2).VISIBLE = 0
CODE(NLAYER - 2) = 0
KSBASE(0) = 0
KSBASE(1) = 0
TEXT3(0).TEXT = ""
TEXT3(1).TEXT = ""
END IF
TXTCHANG = -1
61 END SUB

SUB Text1_Change (Index AS INTEGER)
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), crack, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2
FOR I = R TO T
KSUG(I) = VAL(TEXT1(I).TEXT)
NEXT I
TXTCHANG = -1
END SUB

SUB Text2_Change (Index AS INTEGER)
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), crack, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2
FOR I = R TO T
kbase(I) = VAL(TEXT2(I).TEXT)
```

```
NEXT I
TXTCHANG = -1
END SUB

SUB Text3_Change (Index AS INTEGER)
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), crack, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2
FOR I = R TO T
KSBASE(I) = VAL(TEXT3(I).TEXT)
NEXT I
TXTCHANG = -1
END SUB



SUB Form_KeyPress (keyascii AS INTEGER)
IF keyascii = 13 THEN UNLOAD frmpage
END SUB

SUB Form_MouseMove (Button AS INTEGER, Shift AS INTEGER, X AS SINGLE, Y AS SINGLE)
UNLOAD frmpage
END SUB



SUB Command1_Click ()
FRMOUTPUT.HIDE
END SUB


'$FORM frmmat
'$FORM FRMGEN
DIM SHARED E(5), V(5), HH(5), OPTIONS(3), KSUG(2), kbase(2), KSBASE(2), ZONES(6),
SUBGRD(4)
DIM SHARED WPSI(7), SUGV(4), BASEV(4), TRAFICV(4), TEMPV(4), periodv(4), BITAIR(2),
MT(5), CODE(5), OLDBAIR(2)
COMMON E(), V(), HH(), WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), crack,
TXTCHANG, ZONES(), SUBGRD()
COMMON OPTIONS(), KSUG(), kbase(), KSBASE(), BITAIR(), CODE(), OLDBAIR(), MT(),
NLAYER, OLAYINC, TESTTEMP, OLAYTEMP
COMMON DSCRPTION$, A, B, AA, BB, K, Y, Z, R, T, G, ZON, SHOWPRN, PASTCHCK, OLAYT,
DAMA1, DAMA2


SUB DESCRIPTION_Change ()
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), crack, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2
DSCRPTION$ = DESCRIPTION.TEXT
END SUB
```

```
SUB Form_Load ()
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), crack, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2
FOR I = 0 TO 2
OPTION1(I).VALUE = OPTIONS(I)
NEXT I
Z = Z + 1
G = 1
K = 1
IF Z = 1 THEN
GOTO 15
END IF
FOR I = 0 TO 4
X = I
Y = I
TEXT1(I).TEXT = STR$(E(I))
IF E(I) = 0 THEN TEXT1(I).TEXT = ""
TEXT2(I).TEXT = STR$(V(I))
IF V(I) = 0 THEN TEXT2(I).TEXT = ""
NEXT I
FOR I = 0 TO 3
TEXT3(I).TEXT = STR$(HH(I))
IF HH(I) = 0 THEN TEXT3(I).TEXT = ""
NEXT I
TEXT4.TEXT = STR$(OLAYINC)
IF OLAYINC = 0 THEN TEXT4.TEXT = ""
TEXT5.TEXT = STR$(TESTTEMP)
IF TESTTEMP = 0 THEN TEXT5.TEXT = ""
TEXT6.TEXT = STR$(OLAYTEMP)
IF OLAYTEMP = 0 THEN TEXT6.TEXT = ""
'TEXT7.TEXT = STR$(BITAIR(0))
'IF BITAIR(0) = 0 THEN TEXT7.TEXT = ""
'TEXT8.TEXT = STR$(BITAIR(1))
'IF BITAIR(1) = 0 THEN TEXT8.TEXT = ""
TEXT9.TEXT = STR$(crack)
IF crack = 0 THEN TEXT9.TEXT = ""
'OLDACBIT.TEXT = STR$(OLDBAIR(0))
'IF OLDBAIR(0) = 0 THEN OLDACBIT.TEXT = ""
'OLDACAIR.TEXT = STR$(OLDBAIR(1))
'IF OLDBAIR(1) = 0 THEN OLDACAIR.TEXT = ""
DESCRIPTION.TEXT = DSCRPTION$
15 X = 0
  Y = 4
  END SUB

SUB Label1_Change (Index AS INTEGER)

END SUB

SUB Label6_Change ()

END SUB
```

```
SUB OLDACAIR_Change ()
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), crack, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2
'OLDBAIR(1) = VAL(OLDACAIR.TEXT)
END SUB

SUB OLDACBIT_Change ()
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), crack, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2
'OLDBAIR(0) = VAL(OLDACBIT.TEXT)
END SUB

SUB Option1_Click (Index AS INTEGER)
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), crack, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2
UNLOAD FRMGEN
UNLOAD frmmat
FOR I = 0 TO 2
OPTIONS(I) = OPTION1(I).VALUE
NEXT I
IF OPTION1(1).VALUE = -1 THEN
sugl.cAption = "  SUBGRADE"
sbasel.cAption = "  SUBBASE LAYER"
TEXT1(3).visible = -1
TEXT2(3).visible = -1
TEXT2(3).visible = -1
TEXT3(2).visible = -1
sbasel.visible = -1
sugl.visible = 0
basel.cAption = "  BASE LAYER"
sbasel.cAption = "  SUBGRADE"
TEXT1(4).visible = 0
TEXT2(4).visible = 0
TEXT2(4).visible = 0
TEXT3(3).visible = 0
NLAYER = 4
ELSEIF OPTION1(2).VALUE = -1 THEN
sugl.visible = 0
sbasel.visible = 0
basel.visible = -1
basel.cAption = "  SUBGRADE"
TEXT1(3).visible = 0
TEXT2(3).visible = 0
TEXT2(3).visible = 0
TEXT1(4).visible = 0
TEXT2(4).visible = 0
TEXT2(4).visible = 0
```

```
TEXT3(3).visible = 0
TEXT3(2).visible = 0
IF CODE(2) >= 4 THEN CODE(2) = 0
NLAYER = 3
ELSE

sugl.visible = -1
basel.visible = -1
sbasel.visible = -1
sugl.cAption = "  SUBGRADE"
sbasel.cAption = "  SUBBASE LAYER"
basel.cAption = "  BASE LAYER"
TEXT1(3).visible = 1
TEXT2(3).visible = 1
TEXT2(3).visible = 1
TEXT1(4).visible = 1
TEXT2(4).visible = 1
TEXT2(4).visible = 1
TEXT3(3).visible = 1
TEXT3(2).visible = 1
NLAYER = 5
END IF
TXTCHANG = -1
END SUB

SUB Text1_Change (Index AS INTEGER)
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), crack, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2
UNLOAD FRMGEN
FOR I = X TO Y
E(I) = VAL(TEXT1(I).TEXT)
NEXT I
TXTCHANG = -1
END SUB

SUB Text2_Change (Index AS INTEGER)
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), crack, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2
FOR I = X TO Y
V(I) = VAL(TEXT2(I).TEXT)
NEXT I
TXTCHANG = -1
END SUB

SUB Text3_Change (Index AS INTEGER)
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), crack, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2
FOR I = X TO Y - 1
```

```
HH(I) = VAL(TEXT3(I).TEXT)
NEXT I
TXTCHANG = -1
END SUB

SUB Text4_Change ()
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), crack, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2
OLAYINC = VAL(TEXT4.TEXT)
TXTCHANG = -1
END SUB

SUB Text5_Change ()
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), crack, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2
TESTTEMP = VAL(TEXT5.TEXT)
TXTCHANG = -1
END SUB

SUB Text6_Change ()
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), crack, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2
OLAYTEMP = VAL(TEXT6.TEXT)
TXTCHANG = -1
END SUB

SUB Text7_Change ()
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), crack, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2
'BITAIR(0) = VAL(TEXT7.TEXT)
TXTCHANG = -1
END SUB

SUB Text8_Change ()
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), crack, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2
'BITAIR(1) = VAL(TEXT8.TEXT)
TXTCHANG = -1
END SUB

SUB Text9_Change ()
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), crack, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
```

```
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2
crack = VAL(TEXT9.TEXT)
TXTCHANG = -1
END SUB


DECLARE SUB Picture1_Click ()
DIM SHARED E(5), V(5), HH(5), OPTIONS(3), KSUG(2), kbase(2), KSBASE(2), ZONES(6),
SUBGRD(4)
DIM SHARED WPSI(7), SUGV(4), BASEV(4), TRAFICV(4), TEMPV(4), periodv(4), BITAIR(2),
MT(5), CODE(5), OLDBAIR(2)
COMMON E(), V(), HH(), WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), crack,
TXTCHANG, ZONES(), SUBGRD()
COMMON OPTIONS(), KSUG(), kbase(), KSBASE(), BITAIR(), CODE(), OLDBAIR(), MT(),
NLAYER, OLAYINC, TESTTEMP, OLAYTEMP
COMMON DSCRPTION$, A, B, AA, BB, K, Y, Z, R, T, G, ZON, SHOWPRN, PASTCHCK, OLAYT,
DAMA1, DAMA2, DAMA3, DAMA4, DAMA22


SUB Form_Load ()
SHARED WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), crack, TXTCHANG,
ZONES(), SUBGRD(), BITAIR(), OLDBAIR(), NLAYER, FILE$, ZON, SHOWPRN, PASTCHCK
SHARED E(), DSCRPTION$, V(), HH(), OLAYINC, TESTTEMP, OLAYTEMP, X, Y, Z, A, B, AA,
BB, K, R, T, G, OPTIONS(), KSUG(), kbase(), KSBASE(), MT(), CODE(), OLAYT, DAMA1, DAMA2,
DAMA3, DAMA4, DAMA22
LABEL1(5).VISIBLE = 0
LABEL2(5).VISIBLE = 0
LABEL2(0).CAPTION = STR$(OLAYT)
LABEL2(1).CAPTION = STR$(DAMA1)
LABEL2(2).CAPTION = STR$(DAMA2)
LABEL2(3).CAPTION = STR$(DAMA4)
LABEL2(4).CAPTION = STR$(DAMA22)
IF (NLAYER = 4 OR NLAYER = 5) AND (CODE(2) = 5) THEN
LABEL2(5).VISIBLE = -1
LABEL2(5).CAPTION = STR$(DAMA3)
LABEL1(5).VISIBLE = -1
END IF
END SUB


DIM SHARED E(5), V(5), HH(5), OPTIONS(3), KSUG(2), kbase(2), KSBASE(2), ZONES(6),
SUBGRD(4)
DIM SHARED WPSI(7), SUGV(4), BASEV(4), TRAFICV(4), TEMPV(4), periodv(4), BITAIR(2),
MT(5), CODE(5), OLDBAIR(2)
COMMON E(), V(), HH(), WPSI(), SUGV(), BASEV(), TRAFICV(), TEMPV(), periodv(), crack,
TXTCHANG, ZONES(), SUBGRD()
COMMON OPTIONS(), KSUG(), kbase(), KSBASE(), BITAIR(), CODE(), OLDBAIR(), MT(),
NLAYER, OLAYINC, TESTTEMP, OLAYTEMP
COMMON DSCRPTION$, A, B, AA, BB, K, Y, Z, R, T, G, ZON, SHOWPRN, PASTCHCK, OLAYT,
DAMA1, DAMA2, DAMA3, DAMA4, DAMA22
```

SYSAN.EXE

```
CC  MECHANISTIC APPROACH TO FLEXIBLE  OVERLAY DESIGN
CC  ----------------------------------------------
CC
CC
CC  BLOCK 2 ( LOADING DATA FILE )
CC  ******************************
CC
CC   INCLUDING GRAFHICS LIBRARIES
CC
    INCLUDE   'FGRAPH.FI'
    INCLUDE  'FGRAPH.FD'
CC
CC   COMMON BLOCK DATA & VARIABLE DECLERATION
CC
    COMMON/RMCOY/ AZ(250),  A(250,5), B(250,5), C(250,5),
   1   D(250,5), AJ(250),  RJ1(250), RJ0(250),BZ(250),
   2   X(5,4,4), SF,      PR,      PA,      P,
   3   T1P,      T1M,     T1,      T2,      T3,
   4   EP,       T4,      T5,      T6,      T2P,
   5   T2M,      BJ1,     BJ0,     SZ1,     SZ2,RDTR0(4),RDZR0(4),
   6   SG1,      SG2,     FM(2,2),  PM(4,4,4),RR(2),IR,IRT,
   7   V(5),     HH(4),   H(4),    TEST(11),
   8   SC(4),    ZZ(5),   E(5),    R,       Z,
   9   AR,       NS,      N,       L,       ITN,
   A   RSZ,      RSR,     ROM,     RMU,CSZ,WGTSUB,
   B   CST,      CSR,     CTR,     COM,     CMU,
   C   PSI,      NLINE,   NOUTP,   NTEST,I,RMLIF2,RMLIF3,
   D      ITN4,    K,          LC,  JT,  TZZ,   FAT258(4),    FAT278(2,4),
   E      ZF,        PH,      PH2,   VK2,      VKP2,IV,IZMID,IZEND, FAT58(4),
   F      VK4,       VKP4,    VKK8,        RDT,      RDS,FIFE4(4),FAT2(4), SPACE,
   G      SIGSTR,ITN6,PP,FFF,FIFE1(4),FIFE2(4),EEE,WGT,CHECK,IZ,IS,IW,
   H   FAT1(4),FRUT(4),FIFE3(4),BIT(3),AIR(3),FR,FAT78(2,4),KODE(5),
   I   NSESON,MT(5),SESON(4),EAL,EALP,SPRB(5),WINB(5),OLAY,FAT3(4),
   K      ECONST(5),IWW,DAMA23,ECONS1(5,4),ECONS2(5,4),FALLB(5),SUMB(5),
   L      DAMA1,DAMA2,DAMA3,DAMA4,DAMAPAST,OLAYINC,FINCHK,COUNTER,ITER,
   M      FATNSF,FATOSF
C     DIMENSION IH(5),IT(5),IM(5),IE(5)
     DOUBLE PRECISION BZ,SZ1,SZ2,SG1,SG2,AZ,P,AJ,A,B,C,D,X,EP,T1P,T1M,
   1T1,T2,T3,T4,T5,T6,T2P,T2M,BJ1,BJ0,PR,PA,SF,PP,RJ0,RJ1
    DOUBLE PRECISION PM,FM
C     DIMENSION ZD(5),EZ(5),ESTR(4,5),STV(5),STR(5)

C     INTEGER DN,NSS,I
     INTEGER    I
    INTEGER*2  FNAME
    DIMENSION E1(5),E2(5)
 1000   CONTINUE
CC
CC   READING PAVE.DAT FILE & INTIALIZTION OF VARIABLES
```

```
CC
          DAMA23=0
          IWW=1
          FNAME=3
          FINCHK=0
          COUNTER=0
   OPEN(FNAME,FILE="PAVE.DAT",STATUS='OLD',ERR=4160)
   READ(FNAME,*,ERR=4160)OLAYINC
   READ(FNAME,*,ERR=4160)PAVETEMP,OLAYTST
   READ(FNAME,*,ERR=4160)WGTSUB,SPACE,PSI,EAL,EALP,FATNSF,FATOSF
   DO 4161 I=1,5
   READ(FNAME,*,ERR=4160)WINB(I),SPRB(I),SUMB(I),FALLB(I)
4161 CONTINUE
   READ(FNAME,*,ERR=4160)NS
   DO 4162 I=1,NS
   READ(FNAME,*,ERR=4160)ECONST(I),V(I)
4162 CONTINUE
   DO 4171 I=3,NS
   READ(FNAME,*,ERR=4160)KODE(I)
4171 CONTINUE
   DO 4172 I=3, NS
   READ(FNAME,*,ERR=4160)E1(I),E2(I)
4172 CONTINUE
   BIT(3)=0
   AIR(3)=0
   KODE(1)=1
   KODE(2)=1
   MT(1)="AC"
   MT(2)="AC"
   SELECT CASE(NS)
   CASE(3)
   MT(3)="SG"
   CASE(4)
   MT(3)="BS"
   IF (KODE(3).EQ.4) KODE(3)=2
   MT(4)="SG"
   CASE(5)
   MT(3)="BS"
   IF (KODE(3).EQ.4) KODE(3)=2
   MT(4)="SBS"
   MT(5)="SG"
   END SELECT
CC
CC   TREAT BITUMEN TREATED BASE AS ASPHALT LAYER
   IF (((NS.EQ.4).OR.(NS.EQ.5)).AND.(KODE(3).EQ.5)) THEN
   BIT(3)=E1(3)
   AIR(3)=E2(3)
   E1(3)=0
   E2(3)=0
   MT(3)="AC"
   ENDIF
CC
   DO 4163 I=1,NS-1
   READ(FNAME,*,ERR=4160)HH(I)
```

```
4163 CONTINUE
     DO 4175 I=1, NS
     IF( E1(I).EQ.0) THEN
     E1(I)=ECONST(I)
     END IF
4175 CONTINUE
     READ(FNAME,*,ERR=4160)BIT(2),AIR(2)
     READ(FNAME,*,ERR=4160)BIT(1),AIR(1)
     read(fname,*,err=4160)crack
     CLOSE(FNAME)
     IF (EALP.EQ.0) THEN
     IWW=2
     GOTO 4890
     ENDIF
     EH=HH(1)-OLAYINC
     HH(1)=0.02
     GOTO 4890
4160 WRITE(*,*)'FILE WAS NOT FOUND'
     READ(*,*)
     GOTO 40
CC
CC  END OF BLOCK 2
CC-----------------------------------------------------------------
CC
CC  BLOCK 2 ( LAYER MODULI ADJUSTMENTS )
CC  ************************************
CC
     IWW=1
4890  NSESON=4
     SESON(1)='WINTER'
     SESON(2)='SPRING'
     SESON(3)='SUMMER'
   7 SESON(4)='FALL'
     DO 5901 J=1,NSESON
     DO 5900 I=1,NS
CC    CALCULATE DEPTH OF THE REPRESENTATIVE PAVEMENT TEMPERATURE
     Z=(HH(1)+HH(2))
CC    INCLUD BITUMEN TREATED BASE IN CALCULATIONS IF PRESENT
     IF (((NS.EQ.4).OR.(NS.EQ.5)).AND.(KODE(3).EQ.5)) THEN
     Z=Z+HH(3)
CC
     ENDIF
     IF (MT(I).EQ."AC") THEN
     SELECT CASE(J)
     CASE(1)
     MA=WINB(4)
     CASE(2)
     MA=SPRB(4)
     CASE(3)
     MA=SUMB(4)
     CASE(4)
     MA=FALLB(4)
     END SELECT
     Z=Z/2
```

```
CC
CC   PAVEMENT TEMPERATURE CALCULATIONS
     MPP=(MA*(1+1/(Z+4)))-(34/(Z+4))+6
     IF(I.EQ.1) THEN
CC   ADJUSTMENT FOR OVERLAY
     FCEPT=(ECONST(I)**0.35)+0.12692*OLAYTST
     ELSE
CC   ADJUSTMENT FOR OLD AC
     FCEPT=(ECONST(I)**0.35)+0.12692*PAVETEMP
     ENDIF
     ECONS1(I,J)=((FCEPT-0.12692*MPP)**(1/0.35))*1000
     ECONS2(I,J)=0
     ENDIF
     IF ((KODE(I).EQ.4).OR.(KODE(I).EQ.1)) THEN
     ECONS1(I,J)=ECONST(I)*1000
     ECONS2(I,J)=0
     ENDIF
CC   ADJUSTMENT FOR BASE/SUBBASE AND SUBGRADE
CC   WINTER
     IF((J.EQ.1).AND.(MT(I).EQ.'BS')) THEN
     ECONS1(I,J)=WINB(2)*E1(I)*1000
     IF ((ECONS1(I,J)/1000).GT.E1(I)) ECONS2(I,J)=0
     ELSE IF ((J.EQ.4).AND.(MT(I).EQ.'SG')) THEN
     ECONS1(I,J)=WINB(1)*E1(I)*1000
     ELSE IF ((J.EQ.4).AND.(MT(I).EQ.'SBS')) THEN
     ECONS1(I,J)=WINB(2)*E1(I)*1000
     ENDIF
CC   SPRING
     IF((J.EQ.2).AND.(MT(I).EQ.'BS')) THEN
     ECONS1(I,J)=SPRB(2)*E1(I)*1000
     IF ((ECONS1(I,J)/1000).GT.E1(I)) ECONS2(I,J)=0
     ELSE IF ((J.EQ.1).AND.(MT(I).EQ.'SG')) THEN
     ECONS1(I,J)=SPRB(1)*E1(I)*1000
     ELSE IF ((J.EQ.1).AND.(MT(I).EQ.'SBS')) THEN
     ECONS1(I,J)=SPRB(2)*E1(4)*1000
     ENDIF
CC   SUMMER
     IF((J.EQ.3).AND.(MT(I).EQ.'BS')) THEN
     ECONS1(I,J)=SUMB(2)*E1(I)*1000
     IF ((ECONS1(I,J)/1000).GT.E1(I)) ECONS2(I,J)=0
     ELSE IF ((J.EQ.2).AND.(MT(I).EQ.'SG')) THEN
     ECONS1(I,J)=SUMB(1)*E1(I)*1000
     ELSE IF ((J.EQ.2).AND.(MT(I).EQ.'SBS')) THEN
     ECONS1(I,J)=SUMB(2)*E1(I)*1000
     ENDIF
CC   FALL
     IF((J.EQ.4).AND.(MT(I).EQ.'BS')) THEN
     ECONS1(I,J)=FALLB(2)*E1(I)*1000
     IF ((ECONS1(I,J)/1000).GT.E1(I)) ECONS2(I,J)=0
     ELSE IF ((J.EQ.3).AND.(MT(I).EQ.'SG')) THEN
     ECONS1(I,J)=FALLB(1)*E1(I)*1000
     ELSE IF ((J.EQ.3).AND.(MT(I).EQ.'SBS')) THEN
     ECONS1(I,J)=FALLB(2)*E1(I)*1000
     ENDIF
```

```
      ECONS2(I,J)=E2(I)
 5900 CONTINUE
 5901 CONTINUE
CC
CC  END OF BLOCK 2
CC-----------------------------------------------------------------
CC
CC    CALLING FATRUT WHICH INCLUDE BLOCKS 3,4 AND 5
CC    AND CHEVRON PROGRAM TO PERFORM ELASTIC MULTI-LAYER ANALYSIS AND
CC    FATIGUE AND RUTTING DAMAGE ANALYSIS.
CC
      CALL FATRUT
CC-----------------------------------------------------------------
CC
CC  BLOCK 6 (CHECKING FATIGUE AND RUTTING DAMAGE)
CC  ----------------------------------------
CC
CC    CHECK FOR REMAINIG IF > 1 PAVEMENT IS FAILED AND COSIDER IT
CC    AS BASE WITH MODULUS OF 70 ksi
CC
      IF (DAMAPAST.GT.1) THEN
      MT(2)="BSA"
      KODE(2)=4
      ECONST(2)=70
      IF (MT(3).EQ."AC") THEN
      MT(3)="BSA"
      KODE(3)=4
      ECONST(3)=70
      ENDIF
      IWW=2
      DAMA23=DAMAPAST
      DAMAPAST=0
      HH(1)=EH+OLAYINC
      GOTO 4890
      ENDIF
CC
CC    CHECH FATUGE AND RUTTING DAMAGE IF >1 THEN INCREASE OVERLAY AND
CC    GO TO BLOCK 3
CC
      IF (DAMAPAST.GT.0) DAMA23=DAMAPAST
      IF((DAMA1 .GT. 1).OR.(DAMA2 .GT.1).OR.(DAMA3.GT.1).or.
     1 (dama4.gt.1)) THEN
      IF (HH(1).EQ.0.02)THEN
       HH(1)=EH
       GOTO 9800
      ENDIF
      IF (DAMA23.GE.1) THEN
      IF ((DAMA1.GT.1.5).AND.(OLAYINC.LE.0.5)) HH(1)=HH(1)+0.5
      IF ((DAMA1.GT.2).AND.(OLAYINC.LE.0.5))  HH(1)=HH(1)+0.5
      IF ((DAMA1.GT.3).AND.(OLAYINC.LE.0.5)) HH(1)=HH(1)+.5
      ENDIF
      IF (DAMA23.LT.1) THEN
      IF ((DAMA2.GT.1.5).AND.(OLAYINC.LE.0.5)) HH(1)=HH(1)+0.5
      IF ((DAMA2.GT.2).AND.(OLAYINC.LE.0.5))  HH(1)=HH(1)+0.5
```

```
      IF ((DAMA2.GT.3).AND.(OLAYINC.LE.0.5)) HH(1)=HH(1)+.5
      ENDIF

 9800 HH(1)=HH(1)+OLAYINC
      IWW=2
      GOTO 4890
      ENDIF
      IF (FINCHK.EQ.1) THEN
      HH(1)=OLAY
      IWW=2
      GOTO 4890
      ENDIF
      IF (OLAY.EQ.0.02) THEN
      HH(1)=EH+OLAYINC
      FINCHK=0
      IWW=2
      GOTO 4890
      ENDIF
CC
CC    PRODUCE OVERLAY IF FATIGUE AND RUTTING DAMAGE <1 AND GOTO BLOCK 1
CC
   39 OPEN(1,FILE="OLAY.TMP")
      IF (DAMA23.GT.99) DAMA23=99
      WRITE(1,'(F5.2,2X,F8.5,2X,F8.5,2X,F8.5,2X,F8.5,2X,F8.5)') OLAY,
     1DAMA1,DAMA2,DAMA3,DAMA4,DAMA23
      CLOSE(1)
      HSUG=0
      OPEN(7,FILE="LIFE.OUT")
      IF (OLAY.LE.0.05) ZZ(2)=ZZ(2)-0.02
      DO 45 IS=1,4
      IF (OLAY.GT.0.05) THEN
      WRITE(7,*) ZZ(1),FAT1(IS),FIFE1(IS)
      ENDIF
      IF (DAMA23.LT.1) WRITE(7,*) ZZ(2),FAT2(IS),FIFE2(IS)
      IF ((DAMA23.LT.1).AND.(KODE(3).EQ.5))WRITE(7,*) ZZ(3)
     1,FAT3(IS),FIFE3(IS)
      WRITE(7,*) ZZ(IZ),FRUT(IS),FIFE4(IS)
   45 CONTINUE
      CLOSE(7)
   40 END
CC
CC
CC          END OF THE SYSTEM ANALYSIS, RETURN TO (BLOCK 1) THE INPUT-
CC                OUTPUT DATA SYSTEM (VISUAL BASIC FLEXOLAY)
CC
CC*********************************************************************
CC*********************************************************************
CC
CC
CC
CC
CC
CC    ELASTIC MULTI-LAYER, FATIFUE DAMAGE AND RUT DAMAGE ANALYSIS
CC
```

```
      SUBROUTINE FATRUT
      INCLUDE 'FGRAPH.FD'
CC*************************************************************
CC
CC BLOCK 4 (DETERMINATION OF NON-LINEAR LAYERES MODULI VALUES
CC ************************************************************
CC
CC COMMON BLOCK DATA AND VARIABLE DECLERATIONS

      COMMON/RMCOY/ AZ(250), A(250,5), B(250,5), C(250,5),
     1 D(250,5), AJ(250), RJ1(250), RJ0(250),BZ(250),
     2 X(5,4,4), SF,    PR,    PA,    P,
     3 T1P,    T1M,    T1,    T2,    T3,
     4 EP,    T4,    T5,    T6,    T2P,
     5 T2M,    BJ1,    BJ0,    SZ1,    SZ2,RDTR0(4),RDZR0(4),
     6 SG1,    SG2,    FM(2,2), PM(4,4,4),RR(2),IR,IRT,
     7 V(5),    HH(4),    H(4),    TEST(11),
     8 SC(4),    ZZ(5),    E(5),    R,    Z,
     9 AR,    NS,    N,    L,    ITN,
     A RSZ,    RSR,    ROM,    RMU,CSZ,WGTSUB,
     B CST,    CSR,    CTR,    COM,    CMU,
     C PSI,    NLINE,    NOUTP,    NTEST,I,RMLIF2,RMLIF3,
     D    ITN4,    K,    LC, JT,    TZZ,    FAT258(4),    FAT278(2,4),
     E    ZF,    PH,    PH2,    VK2,    VKP2,IV,IZMID,IZEND, FAT58(4),
     F    VK4,    VKP4,    VKK8,    RDT,    RDS,FIFE4(4),FAT2(4), SPACE,
     G    SIGSTR,ITN6,PP,FFF,FIFE1(4),FIFE2(4),EEE,WGT,CHECK,IZ,IS,IW,
     H FAT1(4),FRUT(4),FIFE3(4),BIT(3),AIR(3),FR,FAT78(2,4),KODE(5),
     I NSESON,MT(5),SESON(4),EAL,EALP,SPRB(5),WINB(5),OLAY,FAT3(4),
     K    ECONST(5),IWW,DAMA23,ECONS1(5,4),ECONS2(5,4),FALLB(5),SUMB(5),
     L    DAMA1,DAMA2,DAMA3,DAMA4,DAMAPAST,OLAYINC,FINCHK,COUNTER,ITER,
     M    FATNSF,FATOSF
C     DIMENSION IH(5),IT(5),IM(5)
      DOUBLE PRECISION BZ,SZ1,SZ2,SG1,SG2,AZ,P,AJ,A,B,C,D,X,EP,T1P,T1M,
     1T1,T2,T3,T4,T5,T6,T2P,T2M,BJ1,BJ0,PR,PA,SF,PP,RJ0,RJ1
      DOUBLE PRECISION PM,FM
      DIMENSION ZD(5), EZ(5), ESTR(4,5), STV(5), STR(5)
      INTEGER NSS,I,DUMMY2,DUMMY4
      RECORD / rccoord / curpos
      DAMAPAST=0
CC

      OPEN (6, FILE ='RESULTS.OUT')
  500 CONTINUE
  300 CONTINUE
      NSS=NS-1
CC
CC   CALCULATE DEPTH TO EACH LAYER FOR STRESS CALCULATIONS
CC
      ZD(1) = HH(1)/2
      ZD(2)=HH(1)+HH(2)/2
      IF (NS.EQ.3) ZD(3)=(HH(1)+HH(2))*1.5
      IF (NS.EQ.4) THEN
      ZD(3)=HH(1)+HH(2)+HH(3)/3
      ZD(4)=(HH(1)+HH(2)+HH(3))*1.5
```

```
      ENDIF
      IF (NS.EQ.5) THEN
      ZD(3)=HH(1)+HH(2)+HH(3)/3
      ZD(4)=HH(1)+HH(2)+HH(3)+HH(4)/3
      ZD(5)=(HH(1)+HH(2)+HH(3)+HH(4))*1.5
      ENDIF
      DO 10 I=1,NS
   10 E(I)=ECONST(I)
CC
CC    IF ALL LAYERS ARE LINEAR SKIP THIS BLOCK AND GOTO BLOCK 5A

      DO 200 IS=1,NSESON
      IF ((ECONS2(3,IS).EQ.0).AND.(ECONS2(4,IS).EQ.0).AND.(ECONS2(5,IS)
     1.EQ.0)) THEN
      EZ(1)=ECONS1(1,IS)
      EZ(2)=ECONS1(2,IS)
      EZ(3)=ECONS1(3,IS)
      EZ(4)=ECONS1(4,IS)
      EZ(5)=ECONS1(5,IS)
      GOTO 19
      ENDIF
CC
CC    DETERMINE NUMBER ON NON-LINEAR LAYERS PRESENT
      IF (NS.EQ.3) THEN
      IZMID=3
      IZEND=3
      ENDIF
      IF ((NS.EQ.4).AND.(ECONS2(3,IS).NE.0).AND.(ECONS2(4,IS).EQ.0))
     1 THEN
      IZMID=3
      IZEND=3
      ELSEIF ((NS.EQ.4).AND.(ECONS2(3,IS).EQ.0).AND.(ECONS2(4,IS).NE.0
     1)) THEN
      IZMID=4
      IZEND=4
      ELSEIF ((NS.EQ.4).AND.(ECONS2(3,IS).NE.0).AND.(ECONS2(4,IS).NE.0
     1)) THEN
      IZMID=3
      IZEND=4

      ELSEIF ((NS.EQ.5).AND.(ECONS2(3,IS).NE.0).AND.(ECONS2(4,IS).EQ.0)
     1 .AND.(ECONS2(5,IS).EQ.0)) THEN
      IZMID=3
      IZEND=3
      ELSEIF ((NS.EQ.5).AND.(ECONS2(3,IS).EQ.0).AND.(ECONS2(4,IS).NE.0
     1).AND.(ECONS2(5,IS).EQ.0)) THEN
      IZMID=4
      IZEND=4
      ELSEIF ((NS.EQ.5).AND.(ECONS2(3,IS).EQ.0).AND.(ECONS2(4,IS).EQ.0
     1).AND.(ECONS2(5,IS).NE.0)) THEN
      IZMID=5
      IZEND=5
      ELSEIF ((NS.EQ.5).AND.(ECONS2(3,IS).NE.0).AND.(ECONS2(4,IS).NE.0
     1).AND.(ECONS2(5,IS).EQ.0)) THEN
```

```
      IZMID=3
      IZEND=4
      ELSEIF ((NS.EQ.5).AND.(ECONS2(3,IS).EQ.0).AND.(ECONS2(4,IS).NE.0
     1).AND.(ECONS2(5,IS).NE.0)) THEN
      IZMID=4
      IZEND=5
      ELSE
      IZMID=3
      IZEND=5
      ENDIF
      INTO=0
CC
CC    DETERMINE STRESSES AT CALCULATED DEPTHS CALLING ELASTI MULTI-
CC    LAYER ANALYSIS SUBROUTINE STRESS
CC
      IF (FINCHK.EQ.1) GOTO 100
      DUMMY2=SETTEXTCOLOR(14)
      DUMMY4=SETBKCOLOR(6)
      CALL settextposition(16 , 25, curpos )
      CALL outtext('    Evaluating seasonal moduli    ')
  100 CONTINUE
      DO 20 IV=IZMID,IZEND
      IZ=1
      ZZ(1)=ZD(IV)
      ITER=1
      IR=1
      RR(1)=0
      WGT=WGTSUB*2
      CALL STRESS
      STV(IV)=CSZ
      STR(IV)=CSR
      STV(IV)=-STV(IV)
      STR(IV)=-STR(IV)
   20 CONTINUE
CC
CC    CALCULATE MODULI VALUES OF NON-LINEAR LAYERS AT THE CALCULATED
CC    STRESS LEVEL
CC
      DO 30 I=1,NS
      IF ((KODE(I).LE.1).OR.(KODE(I).GT.3)) GOTO 11
      GOTO 12
   11 EZ(I) =ECONS1(I,IS)
      GOTO 30
   12 IF (KODE(I) .EQ. 2) GOTO 13
      GOTO 14
   13 IF (STR(I).LT.0) STR(I)=0
      THETAB = STV(I)+2*STR(I)
      IF (THETAB) 22,22,21
   22 THETAB = STV(I)
   21 EZ(I)=ECONS1(I,IS)*THETAB**ECONS2(I,IS)
      GOTO 30
   14 IF (KODE(I) .EQ. 3) GOTO 15
      GOTO 16
   15 IF (STR(I).LT.0) STR(I)=0
```

```
      THETAS=STV(I)+2*STR(I)
      IF (THETAS) 31,31,32
   31 THETAS = STV(I)
   32 EZ(I) = ECONS1(I,IS)*THETAS**ECONS2(I,IS)
      GOTO 30
   16 IF (KODE(I) .EQ. 4) GOTO 17
      GOTO 30
   17 DEVSTR = STV(I)-STR(I)
      IF(DEVSTR) 23,23,24
   23 EZ(I) = E(I)
      GOTO 30
   24 EZ(I)=ECONS1(I,IS)*DEVSTR**ECONS2(I,IS)
   30 CONTINUE
CC
CC    COMPARE CALCULATED MODULI WITH PREVIOUS MODULI VALUE, IF
CC    COVERGENCE CRITERIA SATISFIED PRODUCE LAYER MODULUS
CC    ELSE CONTINUE THE ITTERATIONS
      DO 40 I=IZMID,IZEND
      EDIF=E(I)-EZ(I)
      EDIF=ABS(EDIF)
      ELIMIT=0.1*E(I)
      IF(EDIF .GT. ELIMIT) GOTO 18
   40 E(I) = EZ(I)
      INTO = 6
      GOTO 51
   18 INTO = INTO + 1
      DO 50 I=1,NS
   50 E(I)=EZ(I)
   51   DO 65 I=1,NS
      IF(E(I) .LE. 0.0) GOTO 200
   65 CONTINUE
      IF(INTO .GT.5) GOTO 19
      GOTO 100
   19 DO 60 I=1,NS
   60 ESTR(IS,I)=EZ(I)
  200 CONTINUE
       DO 61 IS=1,NSESON
   61   WRITE(6,2010) (ESTR(IS,I),I=1,NS)
 2009 FORMAT(1H ,//,T50,'RESULTS OF MODULI EVALUATION FOR ALL SEASONS'
     1,//,T10,'SEASON NO.',5X,5(A8,5X),/)
 2010 FORMAT(1H ,7X,5(E10.3,3X))
CC
CC  END OF BLOCK 4
CC-----------------------------------------------------------------
CC
CC  BLOCK 5A & 5B (FATIGUE AND RUTTING DAMAGE ANALYSIS)
CC  ---------------------------------------------
CC

 2040  CONTINUE
      DUMMY2=SETTEXTCOLOR(14)
      DUMMY4=SETBKCOLOR(6)
       CALL settextposition(16 , 25, curpos )
       IF (HH(1).EQ.0.02) THEN
```

```
      CALL OUTTEXT('      Calculating remaining life    ')
      ELSE
      CALL outtext(' Overlay thickness in use = ')
      WRITE(*,'(F5.2\)')HH(1)
      ENDIF
      ITER=0
      H1=HH(1)
CC    PRODUCE TWO CASES, FIRST CASE WHEN CALCULATING REMAINING LIFE
CC    USING PAST TRAFFIC AND SECOND CASE WHEN PERFORMING DAMAGE
CC    ANALYSIS USING FUTURE TRAFFIC
CC
      DO 8000 IW=IWW,2
      SELECT CASE (IW)
      CASE(1)
      HH(1)=.02
      EALC=EALP

CC BLOCK 5 ( USING FUTURE TRAFFIC AND SPECIAL ROUTINE FOR STRAIN
CC          CALCULATIONS AT TNE BOTTOM OF THIN OVERLAYS )
CC
      CASE(2)
      HH(1)=H1
      EALC=EAL
CC
CC    SPECIAL ROUTINE FOR STRAIN CALCULATIONS AT BOTTOM OF THIN OVERLAY
CC
      IF (HH(1).GT.4) GOTO 2032
      S=1
      DO 111 U=5,8,3
      HH(1)=U
      IZ=2
      ZZ(1)=HH(1)-0.01
      ZZ(2)=ZZ(1)+HH(2)
      DO 113 IS=1,NSESON
      DO 114 I =1,NS
      E(I)=ESTR(IS,I)
  114 CONTINUE
      CHECK=0
      IR=2
      IRT=0
      RR(1)=0
      RR(2)=SPACE
      WGT=WGTSUB
      CALL STRESS
      FAT78(S,IS)= FAT58(IS)
      FAT278(S,IS)=FAT258(IS)
  113 CONTINUE
      S=S+1
  111 CONTINUE
CC
CC END OF BLOCK 5B
CC-----------------------------------------------------------------
CC    CALCULATE DEPTH TO BOTTOM OF ASPAHLT LAYERS FOR FATIGUE LIFE
CC    AND TOP OF SUBGRADE FOR RUTTING LIFE
```

```
      HH(1)=H1
      END SELECT
CC
CC    INCLUDE BTB IN FATIGUE CALCULATIONS
 2032 IF (((NS.EQ.4).OR.(NS.EQ.5)).AND.(KODE(3).EQ.5)) THEN
     IZ=4
     ZZ(1)=HH(1)- 0.01
     ZZ(2)=ZZ(1)+HH(2)
     ZZ(3)=HH(1)+HH(2)+HH(3)+0.01
     ELSE
     IZ=3
     ZZ(1)=HH(1)-0.01
     ZZ(2)=ZZ(1)+HH(2)
     ENDIF
     ZZ(IZ)=0
     DO 3001 I=1,NS-1
 3001 ZZ(IZ)=ZZ(IZ)+HH(I)
     ZZ(IZ)=ZZ(IZ)+0.02

     DO 3031 IS=1,NSESON
     DO 3030 I =1,NS
     E(I)=ESTR(IS,I)
 3030 CONTINUE
      CHECK=1
CC
CC    CALL ELASTIC MULTI-LAYER ANALYSIS SUBROUTINE TO CALCULATE FATIGUE
CC    LIFE FOR ASPHALT LAYERS FORM TENSILE STRAIN AT BOTTOM OF EACH LAYER
CC    AND RUTTING LIFE FROM COMPRESSIVE STRAIN AT TOP OF SUBGRADE.
CC
     IR=2
     RR(1)=0
     RR(2)=SPACE
      WGT=WGTSUB
      CALL STRESS
 3031 CONTINUE
CC
CC    CALCULATE SUM OF FATIGUE DAMAGE RATIO FOR FOR ALL SEASONS FOR
CC    OVERLAY
CC
      DAMA1=(SPRB(3)*EALC/(FIFE1(2)*12/SPRB(5)))+
     1(SUMB(3)*EALC/(FIFE1(3)*12/SUMB(5)))+
     1(FALLB(3)*EALC/(FIFE1(4)*12/FALLB(5)))+
     1(WINB(3)*EALC/(FIFE1(1)*12/WINB(5)))
      IF (DAMA23.GE.1) THEN
      DAMA2=0
      DAMA3=0
      GOTO 3070
      ENDIF
CC
CC    IF CASE 1 CALCULATE FATIGUE REMAINING LIFE OF OLD ASPHALT
CC    FOR EACH SEASON AND SUM OF FATIGUE DAMAGE RATIO DUE PAST TRAFFIC

      IF (IW.EQ.1) THEN
      DAMA2S1=(WINB(3)*EALC/(FIFE2(1)*12/WINB(5)))
```

```
      DAMA2S2=(SPRB(3)*EALC/(FIFE2(2)*12/SPRB(5)))
      DAMA2S3=(SUMB(3)*EALC/(FIFE2(3)*12/SUMB(5)))
      DAMA2S4=(FALLB(3)*EALC/(FIFE2(4)*12/FALLB(5)))
      DAMA2=DAMA2S1+DAMA2S2+DAMA2S3+DAMA2S4
      RMLIF2=1-DAMA2
      IF ((DAMA2.LT.1).AND.(DAMA2.GT.0.79)) RMLIF2=0.2
CC
CC
      ELSE
CC
CC    IF PAST TRAFFIC IS NOT CONSIDERED SET REMAINING LIFE
CC    EQUAL 1(i.e. NO LIFE CONSUMED)
      IF (EALP.EQ.0) THEN
      RMLIF2=1
      GOTO 6766
      ENDIF
CC
CC    THE SUM OF FATIGUE DAMAGE RATIO FOR ALL SEASONS FOR OLD
CC    ASPHALT ADJUSTED BY REMAINING LIFE
CC
6766  DAMA2=(SPRB(3)*EALC/(FIFE2(2)*RMLIF2*12/SPRB(5)))+
     1(SUMB(3)*EALC/(FIFE2(3)*RMLIF2*12/SUMB(5)))+
     1(FALLB(3)*EALC/(FIFE2(4)*RMLIF2*12/FALLB(5)))+
     1(WINB(3)*EALC/(FIFE2(1)*RMLIF2*12/WINB(5)))
      ENDIF
CC
CC    IF BTB PRESENT TREAT AS SECOND OLD ASPHALT AND CALCULATE
CC    REMAINING LIFE
CC
      IF (((NS.EQ.4).OR.(NS.EQ.5)).AND.(KODE(3).EQ.5)) THEN
      IF (IW.EQ.1) THEN
      DAMA3S1=(WINB(3)*EALC/(FIFE2(1)*12/WINB(5)))
      DAMA3S2=(SPRB(3)*EALC/(FIFE2(2)*12/SPRB(5)))
      DAMA3S3=(SUMB(3)*EALC/(FIFE2(3)*12/SUMB(5)))
      DAMA3S4=(FALLB(3)*EALC/(FIFE2(4)*12/FALLB(5)))
      DAMA3=DAMA2S1+DAMA2S2+DAMA2S3+DAMA2S4
      RMLIF3=1-DAMA3
      IF ((DAMA3.LT.1).AND.(DAMA3.GT.0.79)) RMLIF3=0.2
CC
CC
      ELSE
CC
CC    IF PAST TRAFFIC IS NOT CONSIDERED SET REMAINING LIFE
CC    EQUAL 1(i.e. NO LIFE CONSUMED)
CC
      IF (EALP.EQ.0) THEN
      RMLIF3=1
      GOTO 6767
      ENDIF
CC
CC    CALCULATE THE SUM OF FATIGUE DAMAGE RATIO FOR ALL SEASONS FOR
CC    OLD ASPHALT ADJUSTED BY REMAINING LIFE
CC
6767  DAMA3=(SPRB(3)*EALC/(FIFE3(2)*RMLIF3*12/SPRB(5)))+
```

```
     1(SUMB(3)*EALC/(FIFE3(3)*RMLIF3*12/SUMB(5)))+
     1(FALLB(3)*EALC/(FIFE3(4)*RMLIF3*12/FALLB(5)))+
     1(WINB(3)*EALC/(FIFE3(1)*RMLIF3*12/WINB(5)))
      ENDIF
      ELSE
      DAMA3=0
      ENDIF
CC
CC    SELECT THE LARGEST FATIGUE DAMAGE DUE PAST TRAFFIC BETWEEN OLD
CC    AC AND BTB

      IF (IW.EQ.1) THEN
      DAMAPAST=DAMA2
      IF (DAMA3.GT.DAMA2) DAMAPAST=DAMA3
      ENDIF
CC
CC    CALCULATE SUM OF RUTTING DAMAGE RATIO FOR ALL SEASONS
CC
 3070 DAMA4=(SPRB(3)*EALC/(FIFE4(2)*12/SPRB(5)))+
     1(SUMB(3)*EALC/(FIFE4(3)*12/SUMB(5)))+
     1(FALLB(3)*EALC/(FIFE4(4)*12/FALLB(5)))+
     1(WINB(3)*EALC/(FIFE4(1)*12/WINB(5)))
 8000 CONTINUE
      CLOSE(6)
CC
CC    PROCEDURE TO SPEED UP THE ITTERATIONS
      IF (HH(1).EQ.0.02) THEN
      GOTO 6000
      ENDIF
      OLAY=HH(1)
      counter=counter+1
      IF((DAMA1 .GT. 1).OR.(DAMA2 .GT.1).OR.(DAMA3.GT.1).or.
     1 (dama4.gt.1)) THEN
      IF (DAMA23.GE.1) THEN
      IF ((DAMA1.GT.1.5).AND.(OLAYINC.LE.0.5)) HH(1)=HH(1)+0.5
      IF ((DAMA1.GT.2).AND.(OLAYINC.LE.0.5))  HH(1)=HH(1)+0.5
      IF ((DAMA1.GT.3).AND.(OLAYINC.LE.0.5)) HH(1)=HH(1)+.5
      ENDIF
      IF (DAMA23.LT.1) THEN
      IF ((DAMA2.GT.1.5).AND.(OLAYINC.LE.0.5)) HH(1)=HH(1)+0.5
      IF ((DAMA2.GT.2).AND.(OLAYINC.LE.0.5))  HH(1)=HH(1)+0.5
      IF ((DAMA2.GT.3).AND.(OLAYINC.LE.0.5)) HH(1)=HH(1)+.5
      ENDIF

      HH(1)=HH(1)+OLAYINC
      IWW=2
      GOTO 2040
      ENDIF
      FINCHK=1+FINCHK
      IF (COUNTER.EQ.1) FINCHK=0
C     close(7)
 6000 RETURN
      END
CC
```

```
CC  END OF BLOCK 5
CC-------------------------------------------------------------
CC
CC
CC
CC
CC**************** ELASTIC MULTI-LAYER ANALYSIS ************************
CC
CC   ******************************************************
      SUBROUTINE STRESS
CC   ******************************************************
      COMMON/RMCOY/ AZ(250),  A(250,5), B(250,5), C(250,5),
     1  D(250,5), AJ(250),  RJ1(250), RJ0(250),BZ(250),
     2  X(5,4,4), SF,     PR,     PA,     P,
     3  T1P,     T1M,     T1,     T2,     T3,
     4  EP,      T4,      T5,     T6,     T2P,
     5  T2M,     BJ1,     BJ0,    SZ1,    SZ2,RDTR0(4),RDZR0(4),
     6  SG1,     SG2,     FM(2,2),  PM(4,4,4),RR(2),IR,IRT,
     7  V(5),    HH(4),   H(4),    TEST(11),
     8  SC(4),   ZZ(5),   E(5),    R,      Z,
     9  AR,      NS,      N,      L,      ITN,
     A  RSZ,     RSR,     ROM,    RMU,CSZ,WGTSUB,
     B  CST,     CSR,     CTR,    COM,    CMU,
     C  PSI,     NLINE,   NOUTP,  NTEST,I,RMLIF2,RMLIF3,
     D     ITN4,    K,          LC,  JT,  TZZ,  FAT258(4),  FAT278(2,4),
     E     ZF,      PH,      PH2,  VK2,     VKP2,IV,IZMID,IZEND, FAT58(4),
     F     VK4,     VKP4,    VKK8,     RDT,     RDS,FIFE4(4),FAT2(4), SPACE,
     G     SIGSTR,ITN6,PP,FFF,FIFE1(4),FIFE2(4),EEE,WGT,CHECK,IZ,IS,IW,
     H  FAT1(4),FRUT(4),FIFE3(4),BIT(3),AIR(3),FR,FAT78(2,4),KODE(5),
     I  NSESON,MT(5),SESON(4),EAL,EALP,SPRB(5),WINB(5),OLAY,FAT3(4),
     K     ECONST(5),IWW,DAMA23,ECONS1(5,4),ECONS2(5,4),FALLB(5),SUMB(5),
     L     DAMA1,DAMA2,DAMA3,DAMA4,DAMAPAST,OLAYINC,FINCHK,COUNTER,ITER,
     M     FATNSF,FATOSF
      DIMENSION IH(5),IT(5),IM(5),IE(5)
      DOUBLE PRECISION BZ,SZ1,SZ2,SG1,SG2,AZ,P,AJ,A,B,C,D,X,EP,T1P,T1M,
     1T1,T2,T3,T4,T5,T6,T2P,T2M,BJ1,BJ0,PR,PA,Y,SF,PP,RJ0,RJ1
      DOUBLE PRECISION PM,FM

      FFF=1
C       ** COMPUTE ZEROS OF J1(X) AND J0(X). SET UP GAUSS CONSTANTS **
      K=ITN-13
      DO 2 I=7,K,2
      T = I/2
      TD = 4.0*T - 1.0
      L=I+14
    2 BZ(L) = 3.1415927*(T - 0.25+0.050661/TD
     1        -0.053041/TD**3 + 0.262051/TD**5)
      DO 3 I=8,ITN,2
      T = (I-2)/2
      TD = 4.0*T + 1.0
      L=I+14
    3 BZ(L) = 3.1415927*(T + 0.25-0.151982/TD
     1        + 0.015399/TD**3-0.245270/TD**5)
      N = NS - 1
```

```
      DO 372 I=1,NS
      IE(I)=E(I)
      IH(I)= IE(I)-IE(I)/1000*1000+1000
      IM(I)=IE(I)/1000000
      IT(I)=IE(I)/1000-IM(I)*1000+1000
  372 CONTINUE
      AR = SQRT (WGT/(3.14159*PSI))
      NLINE = 17+NS
      NPAGE = 1
C  ** ADJUST LAYER DEPTHS **
   24 H(1)=HH(1)
      DO 25 I=2,N
   25 H(I)=H(I-1)+HH(I)
      IRT=0
C  ** START ON A NEW R **
C 100 IRT=IRT+1
  100 FFF=1
      IRT=IRT+1
      IF (IRT-IR) 105,105,1010
  105 R=RR(IRT)
      DO 31 I =1,IZ
      DO 31 J=1,N
      TZ  = ABS (H(J) - ZZ(I))
      IF(TZ - .0001) 32,32,31
   32 ZZ(I) = -H(J)
   31 CONTINUE
      NLINE = NLINE+1
C  ** CALCULATE THE PARTITION **
      CALL PART
C  ** CALCULATE THE COEFFICIENTS **
      DO 125 I=1,ITN6
      P=AZ(I)
  107 CONTINUE
      CALL COE5(I)
  109 IF (R) 115,115,110
  110 PR = P*R
C     CALCULATING STRESSES, STRAINS, AND DEFLECTIONS
C191MN  ****** MAIN ROUTINE - N-LAYER ELASTIC SYSTEM ******
      CALL BESSEL (0,PR,Y)
      RJ0(I) = Y
      CALL BESSEL (1,PR,Y)
      RJ1(I) = Y
  115 PA=P*AR
      CALL BESSEL (1,PA,Y)
      AJ(I)=Y
  125 CONTINUE
  195 IZT=0
C  ** START ON A NEW Z **
  200 IZT=IZT+1
      IF(IZT-IZ) 205,205,100
  205 Z=ABS (ZZ(IZT))
      IF ( NLINE - 54 ) 207,206,206
  206 NPAGE = NPAGE + 1
      NLINE = 8
```

```
 207 CONTINUE
C  ** FIND THE LAYER CONTAINING Z **
    TZZ = 0.0
    DO 210 J1=1,N
    J=NS-J1
    IF(Z-H(J)) 210,215,215
 210 CONTINUE
    L = 1
    GO TO 34
 215 L=J+1
    IF (ZZ(IZT)) 33,34,34
  33 L = J
    TZZ = 1.0
  34 CONTINUE
    CALL CALCIN
    IF (TZZ) 36,36,35
  35 ZZ(IZT) = -ZZ(IZT)
    IZT = IZT-1
  36 CONTINUE
    GO TO 200
1010 CONTINUE
    RETURN
    END
    BLOCK DATA
C
    COMMON/RMCOY/ AZ(250),  A(250,5), B(250,5), C(250,5),
   1  D(250,5), AJ(250),  RJ1(250), RJ0(250),BZ(250),
   2  X(5,4,4), SF,     PR,     PA,     P,
   3  T1P,     T1M,     T1,     T2,     T3,
   4  EP,     T4,     T5,     T6,     T2P,
   5  T2M,     BJ1,     BJ0,     SZ1,     SZ2,RDTR0(4),RDZR0(4),
   6  SG1,     SG2,     FM(2,2),  PM(4,4,4),RR(2),IR,IRT,
   7  V(5),    HH(4),   H(4),    TEST(11),
   8  SC(4),   ZZ(5),   E(5),    R,      Z,
   9  AR,      NS,      N,      L,      ITN,
   A  RSZ,     RSR,     ROM,     RMU,CSZ,WGTSUB,
   B  CST,     CSR,     CTR,     COM,     CMU,
   C  PSI,     NLINE,   NOUTP,   NTEST,I,RMLIF2,RMLIF3,
   D     ITN4,    K,          LC,  JT,   TZZ,   FAT258(4),   FAT278(2,4),
   E     ZF,      PH,      PH2,   VK2,      VKP2,IV,IZMID,IZEND, FAT58(4),
   F     VK4,     VKP4,    VKK8,        RDT,      RDS,FIFE4(4),FAT2(4), SPACE,
   G     SIGSTR,ITN6,PP,FFF,FIFE1(4),FIFE2(4),EEE,WGT,CHECK,IZ,IS,IW,
   H  FAT1(4),FRUT(4),FIFE3(4),BIT(3),AIR(3),FR,FAT78(2,4),KODE(5),
   I  NSESON,MT(5),SESON(4),EAL,EALP,SPRB(5),WINB(5),OLAY,FAT3(4),
   K     ECONST(5),IWW,DAMA23,ECONS1(5,4),ECONS2(5,4),FALLB(5),SUMB(5),
   L     DAMA1,DAMA2,DAMA3,DAMA4,DAMAPAST,OLAYINC,FINCHK,COUNTER,ITER,
   M     FATNSF,FATOSF
C   DIMENSION IH(5),IT(5),IM(5)
    DOUBLE PRECISION BZ,SZ1,SZ2,SG1,SG2,AZ,P,AJ,A,B,C,D,X,EP,T1P,T1M,
   1T1,T2,T3,T4,T5,T6,T2P,T2M,BJ1,BJ0,PR,PA,SF,PP,RJ0,RJ1
    DOUBLE PRECISION PM,FM

    EQUIVALENCE(BZ(1),ZB(1))
    REAL*8 ZB(20)
```

```
      DATA ZB /0.0,0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1.0,1.2,1.4,
     11.6,1.8,2.0,2.4048255577,3.38171,5.52007811,7.01559/
C     EQUIVALENCE(BZ(1),ZB(1))
      DATA ITN/40/,ITN6/240/
C     EQUIVALENCE(BZ(1),ZB(1))
      END
      SUBROUTINE BESSEL(NI,XI,Y)
C       ******SUBROUTINE BESSEL - N-LAYER ELASTIC SYSTEM ******
C
      REAL*8 PZ(6)/1.0D0,-1.125D-4,2.8710938D-7,-2.3449658D-9,
     A3.9806841D-11,-1.1536133D-12/,  QZ(6)/-5.0D-3,4.6875D-6,
     B-2.3255859D-8, 2.8307087D-10, -6.3912096D-12,  2.3124704D-12/,
     CP1(6)/ 1.0D0,  1.875D-4,      -3.6914063D-7,   2.7713232D-9,
     D-4.5114421D-11,1.2750463D-12/, Q1(6)/1.5D-2,  -6.5625D-6,
     E 2.8423828D-8,-3.2662024D-10,  7.1431166D-12, -2.5327056D-13/,
     F PI/3.1415927/,D(20)
      DOUBLE PRECISION N,X,Y,X2,FAC,C,T1,T2,T4,T5,T6,T7,P,Q,XI
C
C
    9 N = NI
      X = XI
      IF (X-7.0) 10,10,160
C
   10 X2=X/2.0
      FAC=-X2*X2
      IF (N) 11,11,14
   11 C=1.0
      Y=C
      DO 13 I=1,34
      T=I
      C=FAC*C/(T*T)
      CC=C
      TEST=ABS(CC)-10.0**(-8)
      IF (TEST) 17,17,12
   12 Y=Y+C
   13 CONTINUE
   14 C=X2
      Y=C
      DO 16 I=1,34
      T=I
      C=FAC*C/(T*(T+1.0))
      CC=C
      TEST=ABS(CC)-10.0**(-8)
      IF (TEST) 17,17,15
   15 Y=Y+C
   16 CONTINUE
   17 RETURN
  160 IF (N) 161,161,164
C
C
  161 DO 162 I=1,6
      D(I) = PZ(I)
      D(I+10) = QZ(I)
  162 CONTINUE
```

```
      GO TO 163
C
  164 DO 165 I=1,6
      D(I) = P1(I)
      D(I+10) = Q1(I)
  165 CONTINUE
  163 CONTINUE
      T1 = 25.0/X
      T2=T1*T1
      P = D(6)*T2+D(5)
      DO 170 I=1,4
      J = 5-I
      P = P*T2+D(J)
  170 CONTINUE
      Q = D(16)*T2+D(15)
      DO 171 I=1,4
      J = 5-I
      Q = Q*T2+D(J+10)
  171 CONTINUE
      Q = Q*T1
C
      T4 =DSQRT (X*PI)
      T6=DSIN(X)
      T7=DCOS(X)
C
      IF (N) 180,180,185
C
  180 T5 = ((P-Q)*T6 + (P+Q)*T7)/T4
      GO TO 99
  185 T5 = ((P+Q)*T6 - (P-Q)*T7)/T4
   99 Y = T5
      RETURN
      END
      SUBROUTINE PART
C       ******SUBROUTINE PART - N-LAYER ELASTIC SYSTEM ******
C
      COMMON/RMCOY/ AZ(250),  A(250,5), B(250,5), C(250,5),
     1  D(250,5), AJ(250),  RJ1(250), RJ0(250),BZ(250),
     2  X(5,4,4), SF,     PR,     PA,     P,
     3  T1P,     T1M,     T1,     T2,     T3,
     4  EP,     T4,     T5,     T6,     T2P,
     5  T2M,     BJ1,     BJ0,     SZ1,     SZ2,RDTR0(4),RDZR0(4),
     6  SG1,     SG2,     FM(2,2),  PM(4,4,4),RR(2),IR,IRT,
     7  V(5),    HH(4),   H(4),    TEST(11),
     8  SC(4),   ZZ(5),   E(5),     R,      Z,
     9  AR,      NS,      N,      L,      ITN,
     A  RSZ,     RSR,     ROM,     RMU,CSZ,WGTSUB,
     B  CST,     CSR,     CTR,     COM,     CMU,
     C  PSI,     NLINE,   NOUTP,   NTEST,I,RMLIF2,RMLIF3,
     D     ITN4,    K,          LC,  JT,   TZZ,   FAT258(4),   FAT278(2,4),
     E     ZF,         PH,      PH2,  VK2,      VKP2,IV,IZMID,IZEND, FAT58(4),
     F     VK4,      VKP4,    VKK8,        RDT,      RDS,FIFE4(4),FAT2(4), SPACE,
     G     SIGSTR,ITN6,PP,FFF,FIFE1(4),FIFE2(4),EEE,WGT,CHECK,IZ,IS,IW,
     H  FAT1(4),FRUT(4),FIFE3(4),BIT(3),AIR(3),FR,FAT78(2,4),KODE(5),
```

```
     I   NSESON,MT(5),SESON(4),EAL,EALP,SPRB(5),WINB(5),OLAY,FAT3(4),
     K       ECONST(5),IWW,DAMA23,ECONS1(5,4),ECONS2(5,4),FALLB(5),SUMB(5),
     L       DAMA1,DAMA2,DAMA3,DAMA4,DAMAPAST,OLAYINC,FINCHK,COUNTER,ITER,
     M       FATNSF,FATOSF
C     DIMENSION IH(5),IT(5),IM(5),IE(5)
      DOUBLE PRECISION BZ,SZ1,SZ2,SG1,SG2,AZ,P,AJ,A,B,C,D,X,EP,T1P,T1M,
     1T1,T2,T3,T4,T5,T6,T2P,T2M,BJ1,BJ0,PR,PA,SF,PP,RJ0,RJ1
      DOUBLE PRECISION PM,FM

      REAL*8 G1/0.9324695142/,G2/0.6612093865/,G3/0.2386191861/
    4 ZF = AR
      NTEST = 2
      IF (R) 8,8,9
    9 CONTINUE
      NTEST = AR/R + .0001
      IF (NTEST) 6,6,5
    6 CONTINUE
      NTEST = R/AR + .0001
      ZF = R
    5 CONTINUE
      NTEST = NTEST + 1
      IF (NTEST-10) 8,8,7
    7 CONTINUE
      NTEST = 10
    8 CONTINUE
C       ** COMPUTE POINTS FOR LEGENDRE-GAUSS INTEGRATION **
   15 K = 1
      ZF = 2.0*ZF
      SZ2 = 0.0
      DO 28 I=1,ITN
      SZ1 = SZ2
      SZ2 = BZ(I+1)/ZF
      SF = SZ2 - SZ1
      PP = SZ2 + SZ1
      SG1=SF*G1
      SG2=SF*G2
      SG3=SF*G3
      AZ(K)=PP-SG1
      AZ(K+1)=PP-SG2
      AZ(K+2)=PP-SG3
      AZ(K+3)=PP+SG3
      AZ(K+4)=PP+SG2
      AZ(K+5)=PP+SG1
      K=K+6
   28 CONTINUE
   40 RETURN
      END
      SUBROUTINE CALCIN
C     ******SUBROUTINE CALCIN - N-LAYER ELASTIC SYSTEM ******
C
      COMMON/RMCOY/ AZ(250),  A(250,5), B(250,5), C(250,5),
     1  D(250,5), AJ(250),  RJ1(250), RJ0(250),BZ(250),
     2  X(5,4,4), SF,     PR,     PA,     P,
     3  T1P,     T1M,     T1,     T2,     T3,
```

```
     4  EP,      T4,      T5,      T6,      T2P,
     5  T2M,     BJ1,     BJ0,     SZ1,     SZ2,RDTR0(4),RDZR0(4),
     6  SG1,     SG2,     FM(2,2), PM(4,4,4),RR(2),IR,IRT,
     7  V(5),    HH(4),   H(4),    TEST(11),
     8  SC(4),   ZZ(5),   E(5),    R,       Z,
     9  AR,      NS,      N,       L,       ITN,
     A  RSZ,     RSR,     ROM,     RMU,CSZ,WGTSUB,
     B  CST,     CSR,     CTR,     COM,     CMU,
     C  PSI,     NLINE,   NOUTP,   NTEST,I,RMLIF2,RMLIF3,
     D      ITN4,    K,          LC, JT,  TZZ,  FAT258(4),  FAT278(2,4),
     E      ZF,      PH,      PH2,  VK2,      VKP2,IV,IZMID,IZEND, FAT58(4),
     F      VK4,     VKP4,    VKK8,        RDT,      RDS,FIFE4(4),FAT2(4), SPACE,
     G      SIGSTR,ITN6,PP,FFF,FIFE1(4),FIFE2(4),EEE,WGT,CHECK,IZ,IS,IW,
     H   FAT1(4),FRUT(4),FIFE3(4),BIT(3),AIR(3),FR,FAT78(2,4),KODE(5),
     I   NSESON,MT(5),SESON(4),EAL,EALP,SPRB(5),WINB(5),OLAY,FAT3(4),
     K      ECONST(5),IWW,DAMA23,ECONS1(5,4),ECONS2(5,4),FALLB(5),SUMB(5),
     L      DAMA1,DAMA2,DAMA3,DAMA4,DAMAPAST,OLAYINC,FINCHK,COUNTER,ITER,
     M      FATNSF,FATOSF
      DOUBLE PRECISION BZ,SZ1,SZ2,SG1,SG2,AZ,P,AJ,A,B,C,D,X,EP,T1P,T1M,
     1T1,T2,T3,T4,T5,T6,T2P,T2M,BJ1,BJ0,PR,PA,SF,PP,RJ0,RJ1
      DOUBLE PRECISION PM,FM

      REAL*8 W(6)/0.1713244924,0.360761573,2*0.4679139346,0.360761573,
     10.1713244924/
    2 VL=2.0*V(L)
      EL=(1.0+V(L))/E(L)
      VL1=1.0-VL
      CSZ=0.0
      CST=0.0
      CSR=0.0
      CTR=0.0
      COM=0.0
      CMU=0.0
      NTS1 = NTEST + 1
      ITS = 1
      JT = 0
      ARP = AR
      IF (NOUTP) 4,4,5
    4 ARP = ARP*PSI
    5 CONTINUE
   10 DO 40 I=1,ITN
C     INITIALIZE THE SUB-INTEGRALS
      RSZ=0.0
      RST=0.0
      RSR=0.0
      RTR=0.0
      ROM=0.0
      RMU=0.0
C     COMPUTE THE SUB-INTEGRALS
      K=6*(I-1)
      DO 30 J=1,6
      J1 = K + J
      P=AZ(J1)
      EP=DEXP(P*Z)
```

```
      T1=B(J1,L)*EP
      T2=D(J1,L)/EP
      T1P=T1+T2
      T1M=T1-T2
      T1=(A(J1,L)+B(J1,L)*Z)*EP
      T2=(C(J1,L)+D(J1,L)*Z)/EP
      T2P=P*(T1+T2)
      T2M=P*(T1-T2)
      WA=AJ(J1)*W(J)
      IF (R) 20,20,15
   15 BJ1=RJ1(J1)*P
      BJ0=RJ0(J1)*P
      RSZ=RSZ+WA*P*BJ0*(VL1*T1P-T2M)
      ROM=ROM+WA*EL*BJ0*(2.0*VL1*T1M-T2P)
      RTR=RTR+WA*P*BJ1*(VL*T1M+T2P)
      RMU=RMU+WA*EL*BJ1*(T1P+T2M)
      RSR=RSR+WA*(P*BJ0*((1.0+VL)*T1P+T2M)-BJ1*(T1P+T2M)/R)
      RST=RST+WA*(VL*P*BJ0*T1P+BJ1*(T1P+T2M)/R)
      GO TO 30
C     SPECIAL ROUTINE FOR R = ZERO
   20 PP=P*P
      RSZ=RSZ+WA*PP*(VL1*T1P-T2M)
      ROM=ROM+WA*EL*P*(2.0*VL1*T1M-T2P)
      RST=RST+WA*PP*((VL+0.5)*T1P+0.5*T2M)
      RSR=RST
   30 CONTINUE
C
      SF=(AZ(K+4)-AZ(K+3))/0.4772383722
      CSZ=CSZ+RSZ*SF
      CST=CST+RST*SF
      CSR=CSR+RSR*SF
      CTR=CTR+RTR*SF
      COM=COM+ROM*SF
      CMU=CMU+RMU*SF
      RSZ = 2.0*RSZ*AR*SF
      TESTH=ABS(RSZ)-0.00005
      IF (ITS-NTS1 ) 31,32,32
   31 CONTINUE
      TEST(ITS) = TESTH
      ITS = ITS+1
      GO TO 40
   32 CONTINUE
      TEST(NTS1) = TESTH
      DO 33 J = 1,NTEST
      IF (TESTH-TEST(J)) 35,36,36
   35 CONTINUE
      TESTH = TEST(J)
   36 CONTINUE
      TEST(J) = TEST(J+1)
   33 CONTINUE
      IF (TESTH) 50,50,40
   40 CONTINUE
      JT = 1
   50 CSZ=CSZ*ARP
```

```
      CST=CST*ARP
      CTR=CTR*ARP
      CSR=CSR*ARP
      COM=COM*ARP
      CMU=CMU*ARP
      BSTS = CSZ+CST+CSR
      IF(ITER.EQ.1) GOTO 99
      RDZ=(1000000./E(L))*(CSZ-V(L)*(CSR+CST))
      RDT=(1000000./E(L))*(CST-V(L)*(CSZ+CSR))
      IF (IRT.EQ.1) THEN
      RDTR0(FFF)=RDT
      RDZR0(FFF)=RDZ
      FFF=FFF+1
      GOTO 99
      ENDIF
      IF (FFF.EQ.1) FAT58(IS)=RDT+RDTR0(FFF)
      IF (FFF.EQ.2) FAT258(IS)=RDT+RDTR0(FFF)
      IF (TZZ) 72,72,71
   71 Z = -Z
   72 CONTINUE
      IF(CHECK .NE. 1) THEN
      FFF=FFF+1
      GOTO 99
      ENDIF
      RDZ=(1000000./E(L))*(CSZ-V(L)*(CSR+CST))
C     RDT=(1000000./E(L))*(CST-V(L)*(CSZ+CSR))
      RDS=(1000000./E(L))*(CSR-V(L)*(CSZ+CST))
      SST=(2000000./E(L))*(1.0+V(L))*CTR
CC
CC    FATIGUE AND RUTTING LIFE CALCULATIONS
CC
      IF ((KODE(3).EQ.5).AND.(FFF .EQ. 3)) GOTO 555
      IF (FFF .EQ. IZ) GOTO 556
      IF (FFF .EQ. 2) GOTO 554
      IF (FFF .EQ. 1) GOTO 553
      FFF=FFF+1
CC
CC    FATIGUE LIFE FOR OVERLAY
CC
  553 RDT=RDT+RDTR0(FFF)
      FR=(10)**(4.84*(BIT(2)/(BIT(2)+AIR(2))-0.69))
      IF (IW.EQ.1) GOTO 600
      RDT78=((8-HH(1))*(FAT78(1,IS)-FAT78(2,IS)))/3+FAT78(2,IS)
      IF ((HH(1).LE.4).AND.(RDT.LT.RDT78))THEN
      RDT = RDT78
      ENDIF
  600 FSF=FATNSF
      IF (HH(1).LT.4) FSF=FATNSF
      FIFE1(IS)=0.00432*FSF*FR*(ABS(RDT*1e-6))**(-3.291)*(E(1))**
     1(-0.854)
      FFF=FFF+1
      FAT1(IS)=RDT
      IF ((IW.EQ.1).OR.(HH(1).LT.0.1)) GOTO 99
      GOTO 99
```

```
CC
CC   FATIGUE LIFE FOR OLD AC
CC
 554 RDT=RDT+RDTR0(FFF)
     FR=(10)**(4.84*(BIT(1)/(BIT(1)+AIR(1))-0.69))
     FSF=FATNSF
     IF ((HH(1)+HH(2)).LT.4) THEN
     FSF=FATOSF
     RDT278=((8-(HH(1)+HH(2)))*(FAT278(1,IS)-FAT278(2,IS)))/3+
    1FAT278(2,IS)
     IF (RDT.LT.RDT278)THEN
     RDT = RDT278
     ENDIF
     ENDIF
     FIFE2(IS)=0.00432*FSF*FR*(ABS(RDT*1e-6))**(-3.291)*(E(2))**
    1(-0.854)
     FFF=FFF+1
     FAT2(IS)=RDT
     IF ((IW.EQ.1).OR.(DAMA23.GT.1)) GOTO 99
     GOTO 99
CC
CC   FATIGUE LIFE FOR OLD BTB
CC
 555 RDT=RDT+RDTR0(FFF)
     FR=(10)**(4.84*(BIT(3)/(BIT(3)+AIR(3))-0.69))
     FIFE3(IS)=0.00432*18.4*FR*(ABS(RDT*1e-6))**(-3.291)*(E(3))**
    1(-0.854)
     FFF=FFF+1
     FAT3(IS)=RDT
     IF ((IW.EQ.1).OR.(DAMA23.GT.1)) GOTO 99
     GOTO 99
CC
CC   RUTTING LIFE
CC
c 556 FIFE4(IS) = (ABS(RDZ)**(-4.4843))*1.077*1e18
 556 RDZ=RDZ+RDZR0(FFF)
     FIFE4(IS) = (ABS(RDZ*1E-6)**(-4.477))*1.365*1e-9
     FFF=FFF+1
     FRUT(IS)=RDZ
  99 RETURN
     END
     SUBROUTINE COE5 (KIN)
C     USED FOR 5 OR FEWER LAYERS
C
     COMMON/RMCOY/ AZ(250),  A(250,5), B(250,5), C(250,5),
    1   D(250,5), AJ(250),  RJ1(250), RJ0(250),BZ(250),
    2   X(5,4,4), SF,      PR,      PA,      P,
    3   T1P,      T1M,     T1,      T2,      T3,
    4   EP,       T4,      T5,      T6,      T2P,
    5   T2M,      BJ1,     BJ0,     SZ1,     SZ2,RDTR0(4),RDZR0(4),
    6   SG1,      SG2,     FM(2,2),  PM(4,4,4),RR(2),IR,IRT,
    7   V(5),     HH(4),   H(4),    TEST(11),
    8   SC(4),    ZZ(5),   E(5),    R,       Z,
    9   AR,       NS,      N,       L,       ITN,
```

```
     A  RSZ,     RSR,     ROM,     RMU,CSZ,WGTSUB,
     B  CST,     CSR,     CTR,     COM,     CMU,
     C  PSI,     NLINE,   NOUTP,   NTEST,I,RMLIF2,RMLIF3,
     D     ITN4,    K,          LC,  JT,  TZZ,  FAT258(4),  FAT278(2,4),
     E     ZF,        PH,      PH2,  VK2,      VKP2,IV,IZMID,IZEND, FAT58(4),
     F     VK4,      VKP4,    VKK8,        RDT,      RDS,FIFE4(4),FAT2(4), SPACE,
     G     SIGSTR,ITN6,PP,FFF,FIFE1(4),FIFE2(4),EEE,WGT,CHECK,IZ,IS,IW,
     H  FAT1(4),FRUT(4),FIFE3(4),BIT(3),AIR(3),FR,FAT78(2,4),KODE(5),
     I  NSESON,MT(5),SESON(4),EAL,EALP,SPRB(5),WINB(5),OLAY,FAT3(4),
     K     ECONST(5),IWW,DAMA23,ECONS1(5,4),ECONS2(5,4),FALLB(5),SUMB(5),
     L     DAMA1,DAMA2,DAMA3,DAMA4,DAMAPAST,OLAYINC,FINCHK,COUNTER,ITER,
     M     FATNSF,FATOSF
C    DIMENSION IH(5),IT(5),IM(5),IE(5)
     DOUBLE PRECISION BZ,SZ1,SZ2,SG1,SG2,AZ,P,AJ,A,B,C,D,X,EP,T1P,T1M,
     1T1,T2,T3,T4,T5,T6,T2P,T2M,BJ1,BJ0,PR,PA,SF,PP,RJ0,RJ1
     DOUBLE PRECISION PM,FM,DFAC

C
     REAL*8 SV1(4,2),CV1(2,1),SV2(4,4),CV2(2,2),SV3(4,8),CV3(2,4),
     1     SV4(4,16),CV4(2,8),T(8)
     INTEGER*4  NT(14)
     LC = KIN
CS-MX        SET UP MATRIX X =DI*MI*KI*K*M*D
C    COMPUTE THE MATRICES X(K)
     DO 10 K=1,N
     T1=E(K)*(1.0+V(K+1))/(E(K+1)*(1.0+V(K)))
     T1M=T1-1.0
     PH=P*H(K)
     PH2=PH*2.0
     VK2=2.0*V(K)
     VKP2=2.0*V(K+1)
     VK4=2.0*VK2
     VKP4=2.0*VKP2
     VKK8=8.0*V(K)*V(K+1)
C
     X(K,1,1)=VK4-3.0-T1
     X(K,2,1)=0.0
     X(K,3,1)=T1M*(PH2-VK4+1.0)
     X(K,4,1)=-2.0*T1M*P
C
     T3=PH2*(VK2-1.0)
     T4=VKK8+1.0-3.0*VKP2
     T5=PH2*(VKP2-1.0)
     T6=VKK8+1.0-3.0*VK2
C
     X(K,1,2)=(T3+T4-T1*(T5+T6))/P
     X(K,2,2)=T1*(VKP4-3.0)-1.0
     X(K,4,2)=T1M*(1.0-PH2-VKP4)
C
     X(K,3,4)=(T3-T4-T1*(T5-T6))/P
C
     T3=PH2*PH-VKK8+1.0
     T4=PH2*(VK2-VKP2)
C
```

```
      X(K,1,4)=(T3+T4+VKP2-T1*(T3+T4+VK2))/P
      X(K,3,2)=   (-T3+T4-VKP2+T1*(T3-T4+VK2))/P
C
      X(K,1,3)=T1M*(1.0-PH2-VK4)
      X(K,2,3)=2.0*T1M*P
      X(K,3,3)=VK4-3.0-T1
      X(K,4,3)=0.0
C
      X(K,2,4)=T1M*(PH2-VKP4+1.0)
      X(K,4,4)=T1*(VKP4-3.0)-1.0
C     K = K
   10 CONTINUE
C     COMPUTE THE PRODUCT MATRICES PM
      SC(N)=4.0*(V(N)-1.0)
      IF (N-2) 13,11,11
   11 DO 12 K1=2,N
      M=NS-K1
      SC(M)=SC(M+1)*4.0*(V(M)-1.0)
   12 CONTINUE
   13 CONTINUE
C
      K = N
      DO 15 M=1,4
      DO 14 J=1,2
   14 SV1(M,J) = X(K,M,J+2)
   15 CONTINUE
      CV1(1,1) = -2.0*P*H(K)
      CV1(2,1) = 0.0
      K = K-1
      IF(K) 50,50,20
C
   20 CONTINUE
      DO 22 J=1,2
      J1 = J+J
      T(1) = SV1(1,J)
      T(2) = SV1(2,J)
      T(3) = SV1(3,J)
      T(4) = SV1(4,J)
      DO 21 M=1,4
      SV2(M,J1-1) = X(K,M,1)*T(1)+X(K,M,2)*T(2)
   21 SV2(M,J1) = X(K,M,3)*T(3)+X(K,M,4)*T(4)
   22 CONTINUE
      T(1) = CV1(1,1)
      T(2) = -2.0*P*H(K)
      CV2(1,1) = T(1)
      CV2(1,2) = T(2)
      CV2(2,1) = T(1)-T(2)
      CV2(2,2) = 0.0
      K = K-1
      IF (K) 50,50,30
C
   30 CONTINUE
      DO 34 J=1,4
      J1 = J
```

```
      IF (J1-2) 32,32,31
   31 J1 = J1+2
   32 CONTINUE
      T(1) = SV2(1,J)
      T(2) = SV2(2,J)
      T(3) = SV2(3,J)
      T(4) = SV2(4,J)
      DO 33 M=1,4
      SV3(M,J1) = X(K,M,1)*T(1)+X(K,M,2)*T(2)
   33 SV3(M,J1+2) = X(K,M,3)*T(3)+X(K,M,4)*T(4)
   34 CONTINUE
      T(1) = -2.0*P*H(K)
      DO 35 J=1,2
      CV3(1,J) = CV2(1,J)
      CV3(2,J) = CV2(1,J)-T(1)
      CV3(1,J+2) = CV2(2,J)+T(1)
      CV3(2,J+2) = CV2(2,J)
   35 CONTINUE
      K = K-1
      IF (K) 50,50,40
C
   40 CONTINUE
      DO 42 J=1,4
      T(1) = SV3(1,J)
      T(2) = SV3(2,J)
      T(3) = SV3(3,J)
      T(4) = SV3(4,J)
      T(5) = SV3(1,J+4)
      T(6) = SV3(2,J+4)
      T(7) = SV3(3,J+4)
      T(8) = SV3(4,J+4)
      DO 41 M=1,4
      SV4(M,J) = X(K,M,1)*T(1)+X(K,M,2)*T(2)
      SV4(M,J+4) = X(K,M,3)*T(3)+X(K,M,4)*T(4)
      SV4(M,J+8) = X(K,M,1)*T(5)+X(K,M,2)*T(6)
   41 SV4(M,J+12) = X(K,M,3)*T(7)+X(K,M,4)*T(8)
   42 CONTINUE
      T(1) = -2.0*P*H(K)
      DO 43 J=1,4
      CV4(1,J) = CV3(1,J)
      CV4(2,J) = CV3(1,J)-T(1)
      CV4(1,J+4) = CV3(2,J)+T(1)
      CV4(2,J+4) = CV3(2,J)
   43 CONTINUE
C
   50 CONTINUE
      NT(1) = 1
      DO 51 K=2,N
   51 NT(K) = NT(K-1)+NT(K-1)
      DO 80 K=1,N
      K1 = NS-K
      DO 52 M=1,4
      PM(K1,M,1) = 0.0
      PM(K1,M,2) = 0.0
```

```
 52 CONTINUE
    I1 = NT(K)
    DO 80 M=1,I1
    I2 = M+I1
    GO TO (61,62,63,64),K
 61 CONTINUE
    T(3) = CV1(1,M)
    T(4) = CV1(2,M)
    GO TO 65
 62 CONTINUE
    T(3) = CV2(1,M)
    T(4) = CV2(2,M)
    GO TO 65
 63 CONTINUE
    T(3) = CV3(1,M)
    T(4) = CV3(2,M)
    GO TO 65
 64 CONTINUE
    T(3) = CV4(1,M)
    T(4) = CV4(2,M)
 65 CONTINUE
    T(1) = 0.0
    T(2) = 0.0
    IF(T(3)+180.0)67,66,66
 66 T(1)=DEXP(T(3))
 67 IF(T(4)+180.0)69,68,68
 68 T(2)=DEXP(T(4))
 69 CONTINUE
    DO 80 J=1,2
    GO TO (71,72,73,74),K
 71 CONTINUE
    T(3) = SV1(J,M)
    T(4) = SV1(J,I2)
    T(5) = SV1(J+2,M)
    T(6) = SV1(J+2,I2)
    GO TO 75
 72 T(3) = SV2(J,M)
    T(4) = SV2(J,I2)
    T(5) = SV2(J+2,M)
    T(6) = SV2(J+2,I2)
    GO TO 75
 73 T(3) = SV3(J,M)
    T(4) = SV3(J,I2)
    T(5) = SV3(J+2,M)
    T(6) = SV3(J+2,I2)
    GO TO 75
 74 T(3) = SV4(J,M)
    T(4) = SV4(J,I2)
    T(5) = SV4(J+2,M)
    T(6) = SV4(J+2,I2)
 75 CONTINUE
C
    PM(K1,J,1) = PM(K1,J,1)+T(1)*T(3)
    PM(K1,J,2) = PM(K1,J,2)+T(1)*T(4)
```

```
      PM(K1,J+2,1) = PM(K1,J+2,1)+T(2)*T(5)
      PM(K1,J+2,2) = PM(K1,J+2,2)+T(2)*T(6)
   80 CONTINUE
C     SOLVE FOR C(NS) AND D(NS)
      V2=2.0*V(1)
      V21=V2-1.0
      DO 90 J=1,2
      FM(1,J)=P*PM(1,1,J)+V2*PM(1,2,J)+P*PM(1,3,J)-V2*PM(1,4,J)
   90 FM(2,J)=P*PM(1,1,J)+V21*PM(1,2,J)-P*PM(1,3,J)+V21*PM(1,4,J)
      DFAC=SC(1)/((FM(1,1)*FM(2,2)-FM(2,1)*FM(1,2))*P*P)
      A(LC,NS) = 0.0
      B(LC,NS) = 0.0
      C(LC,NS) = -FM(1,2)*DFAC
      D(LC,NS) =  FM(1,1)*DFAC
C     BACKSOLVE FOR THE OTHER A,B,C,D
      DO 91 K1=1,N
      A(LC,K1)=(PM(K1,1,1)*C(LC,NS)+PM(K1,1,2)*D(LC,NS))/SC(K1)
      B(LC,K1)=(PM(K1,2,1)*C(LC,NS)+PM(K1,2,2)*D(LC,NS))/SC(K1)
      C(LC,K1)=(PM(K1,3,1)*C(LC,NS)+PM(K1,3,2)*D(LC,NS))/SC(K1)
   91 D(LC,K1)=(PM(K1,4,1)*C(LC,NS)+PM(K1,4,2)*D(LC,NS))/SC(K1)
      RETURN
      END
```

DEVELOPMENT OF RECOMMENDATIONS AND GUIDELINES FOR PAVEMENT REHABILITATION DESIGN PROCEDURES FOR THE STATE OF IDAHO

PHASE 2: Development of a Mechanistic-Based Overlay Design System

Volume II
FLEXOLAY Program User Manual

Submitted to:

Idaho Transportation Department
P. O. Box 7129
Boise, Idaho 83707-1129

by

Fouad M. Bayomy
Principal Investigator
Associate Professor

Walid M. Nassar
Graduate Assistant

and

Fawzi Al-Kandari
Graduate Student

Department of civil Engineering
University of Idaho
Department of Civil Engineering
Moscow, Idaho 83844-1022

ITD Project No. RP 121, Agreement #95-60
NCATT, U of I Contract No. FM-K372

Final Report - Volume II
June 1996

DEVELOPMENT OF RECOMMENDATIONS AND GUIDELINES FOR PAVEMENT REHABILITATION DESIGN PROCEDURES FOR THE STATE OF IDAHO

PHASE 2: Development of a Mechanistic-Based Overlay Design System

ITD Project Number RP 121, Agreement No. 95-60
NCATT, University of Idaho Contract Number FM-K372

Volume II
FLEXOLAY Program User Manual

Submitted to:

Idaho Transportation Department
P. O. Box 7129
Boise, Idaho 83707-1129

by

Fouad M. Bayomy
Principal Investigator
Associate Professor

Walid M. Nassar
Graduate Assistant

and

Fawzi Al-Kandari
Graduate Student

Department of civil Engineering
University of Idaho
Department of Civil Engineering
Moscow, Idaho 83844-1022

June 1996

Installation and Operation of

FLEXOLAY 1.2

An Overlay Design Program for Flexible Pavements

1. System Requirements

As with any program, there are some specific hardware that are needed to run the overlay design program. The following is the minimum recommended hardware :

- MS-DOS operating system version 3.0 or later.
- Personal computer using 8088 or higher microprocessor.
- Machine with 1 MB of memory or greater.
- 1 MB free hard-disk space.
- Math co-processor.
- VGA, EGA, Hercules, or other compatible color monitor.
- Mouse or compatible pointing device (recommended).

2. Program Installation

To Install FLEXOLAY in the hard disk, follow the following steps:

- Create a directory in the hard disk and name it C:\FLEXOLAY. At the DOS prompt, type: C:\>md FLEXOLAY and enter.
- Place the distribution diskette of the FLEXOLAY program in the floppy diskette drive (e.g. Drive A).
- Copy all files from drive A to the created directory. From A:> Prompt type A:>COPY *.* C:\FLEXOLAY and enter.

- Now the C:\FLEXOLAY sub-directory contains all files on the FLEXOLAY distribution diskette. Make sure that the following three files exist in the sub-directory:
  FLEXOLAY.EXE, SYSAN.EXE, and DOSXMSF.EXE

3. Starting The Program

After the installation of the program is completed, type FLEXOLAY at the directory which contains the copied program files,(i.e.,C:\FLEXOLAY> FLEXOLAY). This will start the program and display the introduction screen. Move the mouse or press ENTER to clear the introduction. The screen will show a menu bar and a title bar as shown in Fig. 1.

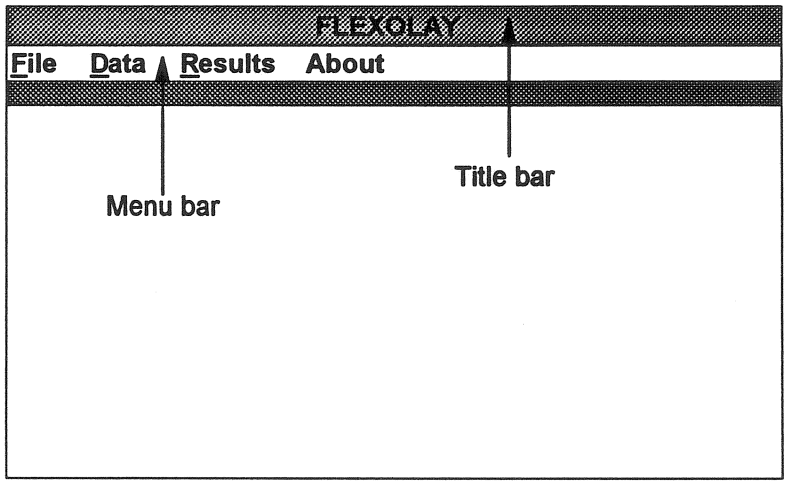


Figure 1 Overlay program menu bar.

After this step, the program is ready to take the commands. The menu bar contains File command menu, Data command menu , Results command menu and the About command. Each command menu contains related commands as shown in Fig. 2 through Fig. 4. There are two ways to select a command with the mouse :

- clicking on the menu name and dragging the mouse down while keeping the mouse button pressed. When the command of interest is highlighted, releasing the button will run the command.

- Clicking once on the menu name. The menu will remain open. Moving the pointer to the command of interest and clicking the mouse will run the command.

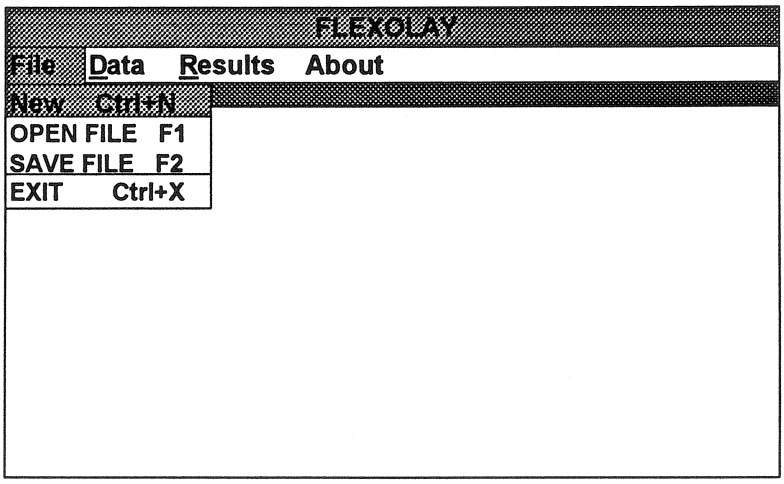


Figure 2 File command menu.

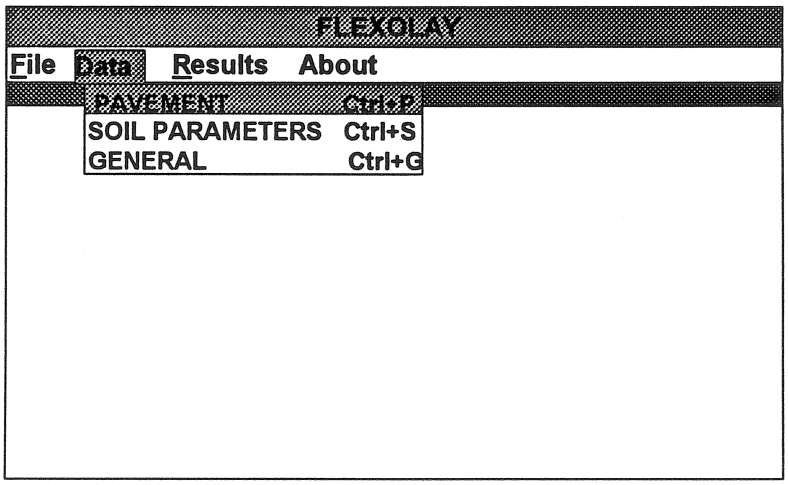


Figure 3 Data command menu

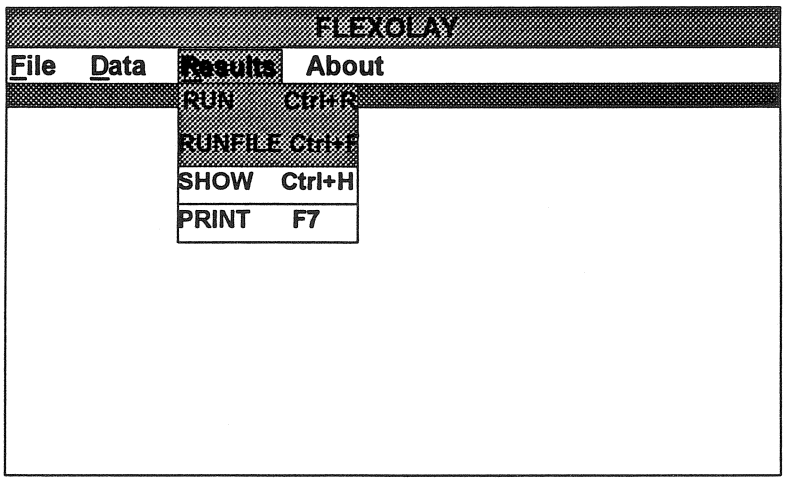


Figure 4 Results command menu.

Access to commands by the keyboard can be done by pressing Alt key. This will activate the menu bar. The command menu of interest, then, can be highlighted using the left and right arrow keys. Pressing Enter key will open the command menu and the command can be selected by using the up and down arrow keys. Once the command is selected, pressing Enter key will run the command. There are also short cut keys to activate a command menu and they are as follows :

- Alt+F to activate the file command menu.
- Alt+D to activate the data command menu.
- Alt+R to activate the results command menu.

4. Program Menu Bar Commands

Before describing the different commands under the different bar commands it is helpful to visualize the entire program operation as a hole. This can be can be done by understanding Figure. 5

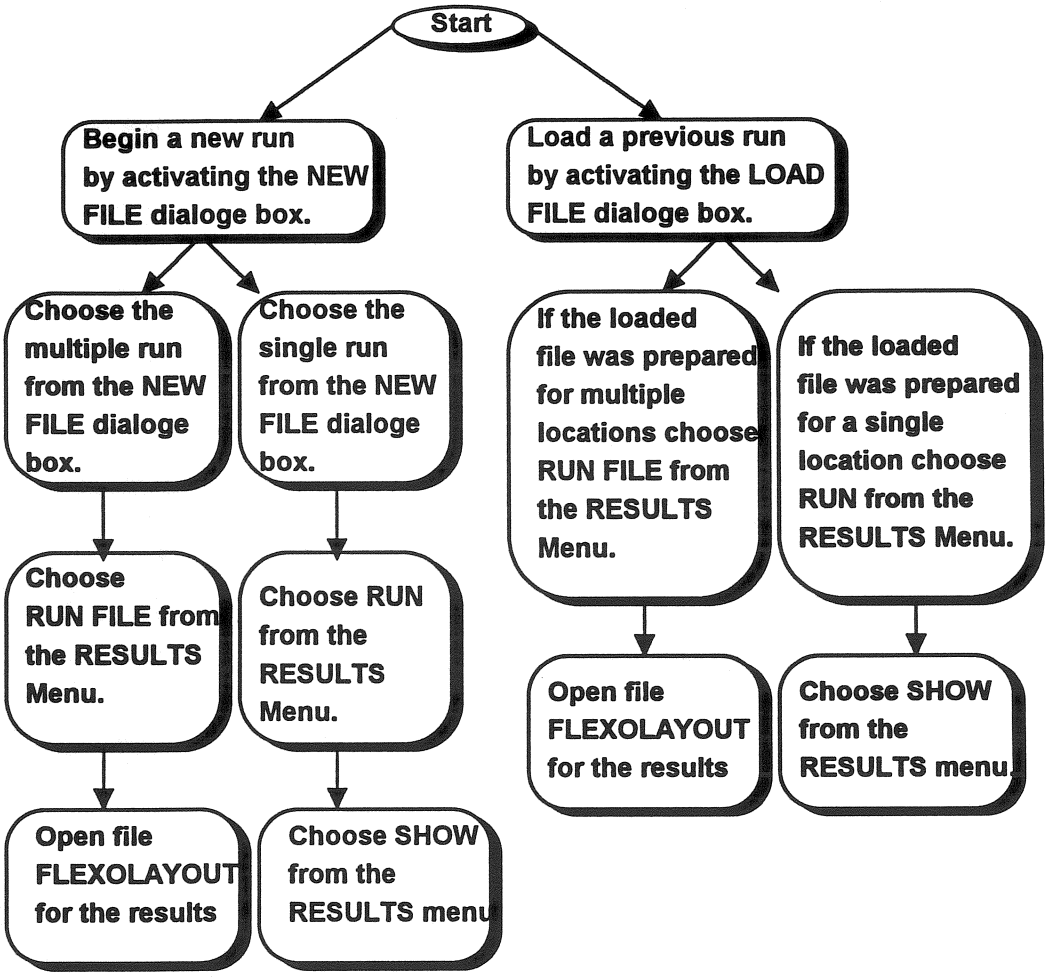


Figure 5. Overview of the program

The option of multiple locations facilitates running the program for different sets of moduli values as will be discussed later.

4.1 File Commands:

The file commands are :

- NEW (to clear the memory and open a new file - short cut key Ctrl+N).
- LOAD FILE (to load an existing file - short cut key F1).
- SAVE FILE (to save current file to disk - short cut key F2).
- Exit (to terminate the program - short cut key Ctrl+X).

The **new file** command opens a dialog box as shown in Fig. 6. The dialog box lets the user choose between two options:

- Running a single location.
- Running multiple locations.

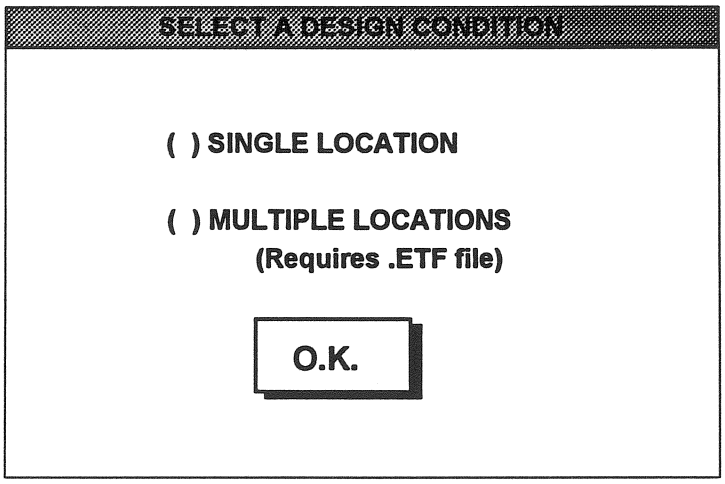


Figure 6. NEW file dialogue box

The **load file** command opens a dialog box as shown in Fig. 7 . The dialog box contains the following :

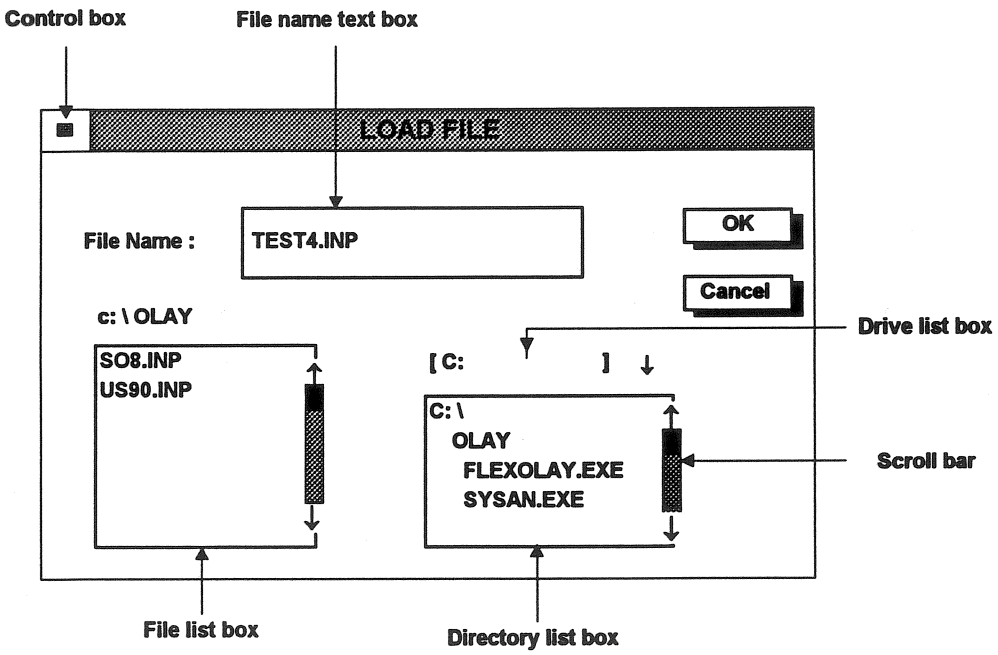


Figure 7 Load file dialog box.

- Drive list box : is the list of drives available (drive A, B, C etc..). The drives are listed when the down arrow attached is clicked. The drive can be selected by moving the mouse to the required drive name and clicking.
- Directory list box : is the list of the directories in the selected drive. The directory which contains the file to be loaded is selected by moving the mouse arrow to the required directory and then pressing the mouse button. The scroll bars attached to the box are used to scroll through the directory list by clicking on the up arrow for up movement or the down arrow for down movement.
- File list box : is the list of the files that are available in the current directory. The file to be loaded can be selected from this list box by moving the mouse pointer to the required file and then clicking. The scroll bars attached to the box are used to scroll through the file list by clicking on the up arrow for up movement or the down arrow for down movement.
- File name text box : the name of the file to be loaded is entered in this box.

Loading a file can be done by :

- directly typing the path (includes drive, directory and sub-directory) and the file name in the file name text box and then pressing the OK button. When typing the file name do not forget the extension, in this case "filename.INP".
- Selecting the drive, the directory and sub-directory, and the file name using the drive list box, the directory list box, and the file list box respectively and then pressing the OK button.

The selection between the options can be done by moving the mouse pointer to the required place (drive, directory, file, OK, Cancel, scroll bars, etc..) and pressing the mouse button. The keyboard also can be used by pressing the Tab key to move between the options (i.e. to move from the file text box to the file list etc..) and the Up and Down keys to scroll within the options.

The **save file** command opens a dialog box as shown in Fig. 8 to enter the path and the name of the file to be saved. Here also do not forget to type the extension of the file, again "filename.INP". The operations in the dialog box are the same as the ones for the load file dialog box.

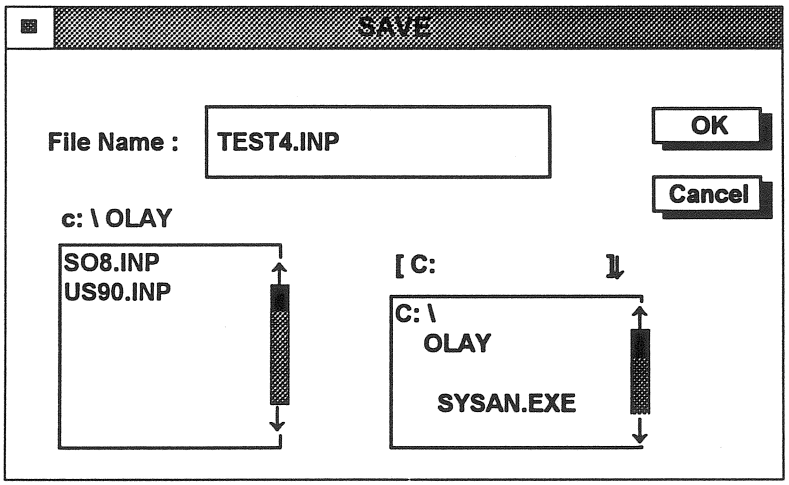


Figure 8 Save file dialog box.

When the **open file** or **save file** dialog boxes are activated only files with INP extension are listed but the program can load and run any file with any extension as long as it contains the correct format of the input data.

4.2 Data Commands:

The Data commands menu contains three sub forms: **Pavement**, **Soil Parameters** and **General**. Each screen has different control parameters which depend on the input data choices.

The **pavement data** input screen is shown in Fig. 9. Listed below are the definitions of terms used in the screen.

- Crack index : The user needs to enter a value between 0 and 5 for the crack index measured in accordance to the State of Idaho procedures. This index, however, is not used in any calculation. Thus any value will not affect the overlay design thickness.
- PAVE. TEMP. (F) : is the pavement temperature during FWD test in F.
- BS AND SBS option : the existing pavement section includes base and subbase.
- BS ONLY option : the existing pavement section includes base only.
- FULL AC option : the existing pavement section is full depth asphalt.
- E (ksi) : is the modulus of the layer in ksi.

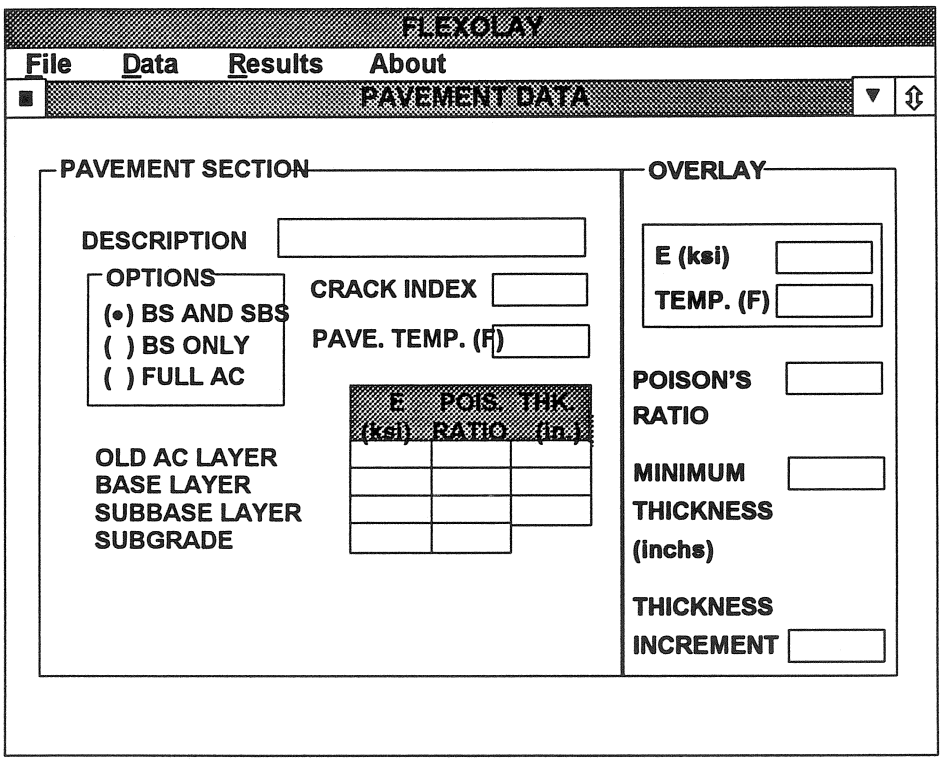


Figure 9 Pavement data screen with base and subbase included.

- POIS. RATIO : is the Poisson's ratio of the layer
- THICK. (in) : is the thickness of the layer in inches
- TEMP. (F): is the reference temperature at which the overlay modulus value was determined.
- MIN. THK.(in) : is the minimum overlay thickness required in inches.
- THK. ICR.(in) : is the overlay thickness increment in inches.

The screen changes as the selected option changes (i.e., changing the option between BS AND SBS ,BS ONLY and FULL AC). The base layer and the subbase layer disappear when FULL AC is selected. The subbase layer disappears when BS ONLY is selected. The data can be entered after moving the mouse pointer to the data boxes provided and pressing the mouse button. Also, the data can be entered after pressing the Tab key to move between the data boxes and the arrow keys to move between the options.

The **soil parameters** screen depends on the option selected in the pavement screen. The shape of the full screen is shown in Fig. 10. When the pavement

section includes base only, the subbase options will disappear and when the pavement section is full depth asphalt, the base and the subbase options will disappear. Listed below is the options used :

- FINE option : means that the subgrade is fine grained and stress dependent. When this options is selected the parameters K1 and K2 will appear, and it is required to enter the values in the data boxes provided.
- GRANULAR option : means that the layer is granular and stress dependent. When this options is selected, the parameters K1 and K2 will appear and it is required to enter the values in the data boxes provided.

- GRAN.[LINEAR] option : means that the layer is granular and considered as stress independent. When this option is selected, the parameters K1 and K2 will disappear.
- LINEAR option : means that the subgrade is considered as stress independent. When this option is selected, the parameters K1 and K2 will disappear.
- CEMENT T. B. option: means cement treated base. When this option is selected, the parameters K1 and K2 will disappear.
- BITUMEN T. B. option : means bitumen treated base. When this option is selected, the parameters Vb and Va will appear. Vb is the percentage bitumen volume and Va is the percentage air volume in the treated base. It is required to enter the values in the data boxes provided.

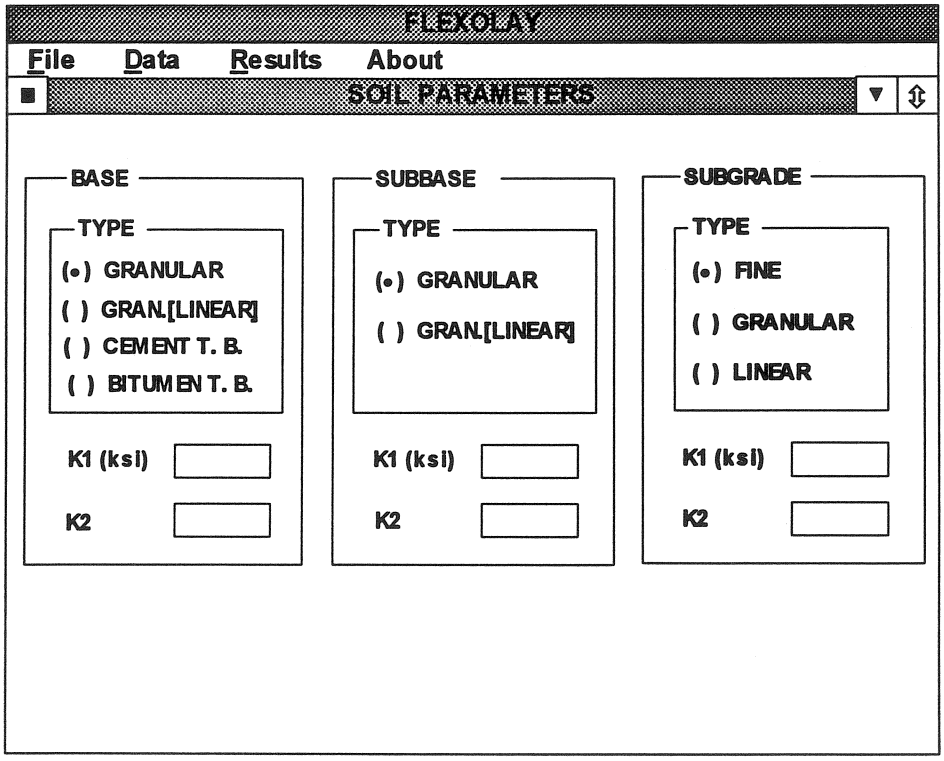


Figure 10 Soil parameters data full screen.

The options can be selected by moving the mouse pointer to required option and then pressing the mouse button. Also, the keyboard can be used by pressing the Tab key to select between the options and the up and down arrow key to select within the option.

The **general data** full screen is shown in Fig. 11 and it includes:

- DUAL TIRE LOAD in lb.
- DUAL TIRE SPACING in inches.
- TIRE PRESSURE in psi.
- ESTIMATED FUTURE : is the estimated future traffic repetitions in terms of equivalent single axle load for the design period.
- PAST ESAL : is past traffic repetitions in terms of equivalent single axle load.
- INCLD. PAST TEAFFIC : check box (when empty the past traffic label and the corresponding data box will disappear)

- FATIGUE SHIFT FACTOR FOR NEW AC : a default value of 18.4 is taken for this shift factor and can be changed here by the user.
- FATIGUE SHIFT FACTOR FOR OLD AC : a default value of 15 is taken for this shift factor and can be changed here by the user.

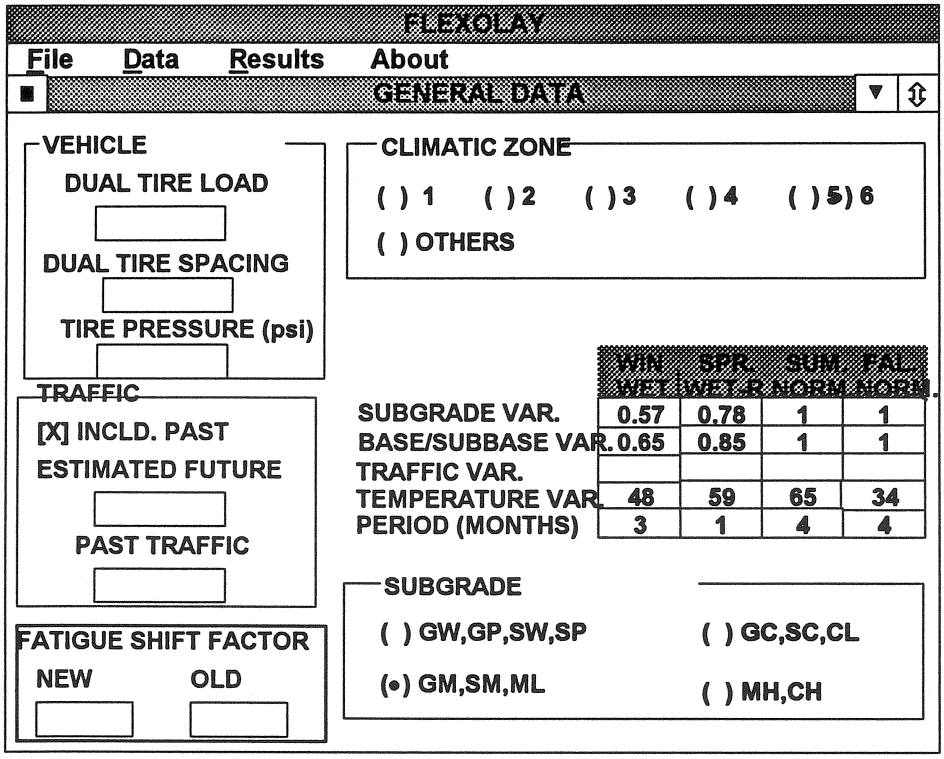


Figure 11 General data full screen.

- CLIMATIC ZONES option: is the pavement operating climate zone where the pavement section is located.
- SUBGRADE VAR. : is the seasonal subgrade modulus adjustment factor.
- BASE/SUBBASE VAR. : is the seasonal base\subbase modulus adjustment factor. (Will be disabled for full depth asphalt pavement section).
- TRAFFIC VAR. : is the seasonal variations in the traffic.
- TEMPERATURE VAR. : is the seasonal mean air temperature.
- PERIOD (MONTHS) : is the period of each season in months.

- SUBGRADE CLASSIFICATION options : is the subgrade classification according to the Unified Soil Classification system. These options only show when ZONE 3 or ZONE 6 is selected.

For ZONE 1 to 6 the SUBGRADE VAR., BASE/SUBBASE VAR., TEMPERATURE VAR. and PERIOD values are loaded automatically by the program and the user can not change these values. In order to change the values, the ZONE has to be selected as OTHERS. The SUBGRADE VAR. values depend on the subgrade modulus value entered in the pavement data screen and will change by changing the modulus.

The data are entered after moving the mouse pointer to the point of interest and pressing the mouse button. The keyboard can be used by pressing the Tab key to move the cursor to the point of interest.

4.3 Result Commands:

The result commands are :

- RUN to run the program for overlay calculations at a single location, i.e. running using one set of moduli values consisting of one modulus for each layer (short cut key Ctrl+R).
- RUN FILE here the user can run multiple locations without the need for re-entering the input values more than once, see section 4.4.
- SHOW (to show the results on the screen-short cut key Ctrl+H).
- PRINT (to print the results-short cut key F7).

After the completion of data entry, selecting the **RUN command** will start the overlay calculations and a running message will appear. The message informs the user about the operations that are taking place and the overlay thickness in use. After the completion of the running process, a summary of the results and the required overlay thickness can be seen on the screen by selecting the **show command** if the performed run was for single location. Detailed results can be printed by selecting the **PRINT command**. A dialog box as shown in Fig. 10 will be displayed when the PRINT command is selected. This dialog box contains the following :

- Printer port options (LPT1, LPT2, LPT3) : These options are selected when the results are required to be printed by a printer. The printer port to be selected is the port where the printer is connected (usually LPT1).
- File option : this option is selected when the results are required to be printed to a file.
- File name text box : When the file option is selected this box will be enabled, and it is required to enter the path and the name of the file to print the results to.

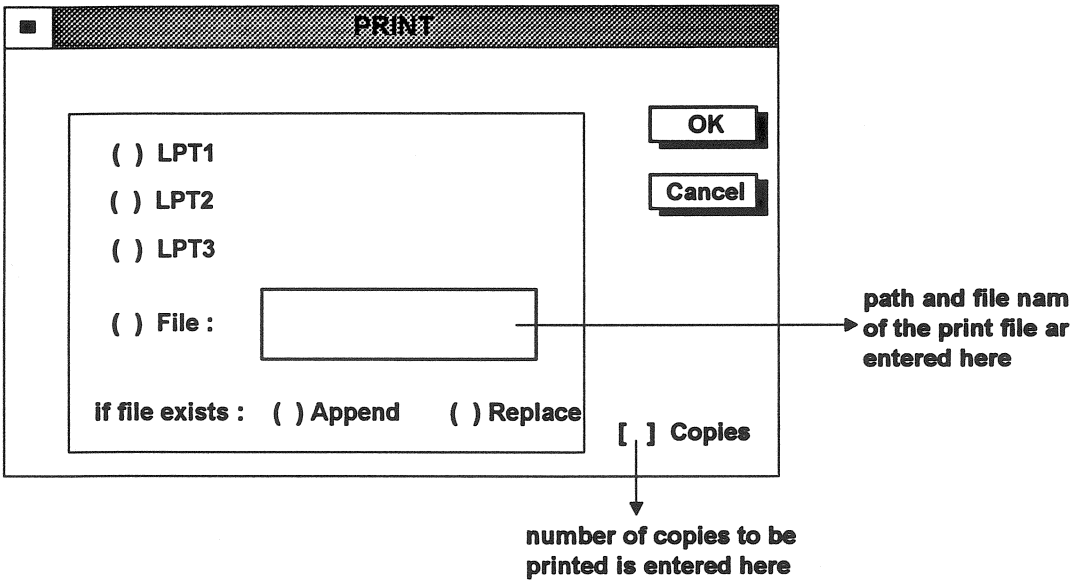


Figure 12 Print dialog box.

- Append option : this option is selected to stop the printing when a file has the same name as the name entered in the file name text box, and thus, preventing the program from replacing that file. This option only available when the file option is selected.
- Replace option : this option is selected to replace an existing file that has the same name as the one entered in the file name text box. This option is only available when the file option is selected.
- Copies : is the required number of copies to be printed.
  OK button : pressed to print the results.
- Cancel button : pressed to cancel the print command.

If the run was for multiple locations clicking the **show command** shows a message to tell the user that the output for the this run will be stored in file "filname.FLX".

4.4 RUN FILE Command:

Clicking this RESULTS Menu command calls for a file with extension .ETF which should be pre-prepared before activating this command. The ETF file is a DOS ASCII file which includes the FWD backcalculation results. It should be noted that the ETF file must be prepared separately in a format as given below:
The first row should include two inputs. In the first column the number of layers for this design excluding the new overlay. In the second column a header for this file, to make it easy for the user to identify the file.
After this, each row represents a data set for a testing location (mile post. A single data set is provided in the following columns:
Column 1: pavement testing temperature
Column 2: mile post
Columns 3-6: Modului values for the existing pavement layers (in ksi).
It is to be noted that if the exiting pavement is a full depth (i.e., only one layer over the subgrade), the user needs to enter only two E values (surface and subgrade)in column 3 and 4. If two layers exist on top of the subgrade, three E values (Surface, base and subgrade) will be needed and so on.
Make sure to create the EFT file under any DOS editor, otherwise you will get a runtime error stating: "Check ETF file format".
An example of a ETF file is provided in the APPENDIX B.

5. Files created by Flexolay

First: If you are running a single location design, you can view the results with the Show command under Results in the main menu. If you want to print the output straight to a printer or to a file, choose Print from the Results menu. One of the files created by FLEXOLAY is this output file which you specify as noted previously.
Second: In order to run a multiple location design, you will choose the name of the ETF file for the input of the temperature and the Moduli

values. A file with the same name but with extension .FLX will be created by FLEXOLAY for the output of this multiple location design.
Third: If you get a runtime error, FLEXOLAY will create a file with the same name as the ETF file but with extension "LOG". You will find two numbers in this file, the first number indicates the type of error you have done according to the code in appendix A. If the second number is anything but 86 reinstall FLEXOLAY.

6. Example

Two example files are provided on the distribution diskette. Example 1 for a single location (one mile post) and example 2 for a pavement section with multiple locations (multiple mile posts). Input files are with extension (*.INP). For design with multiple locations, the output is always stored in a file with extension FLX as mentioned above. The listing for an example of the output file (FLX file) is attached in Appendix B.

An example of the suggested procedures to get accurate and satisfactory results is shown in Appendix C. The example shows how to analyze the data obtained from the backcalculation program MODULUS and decide upon reliable values to use for FLEXOLAY, for either single or multiple sections.

7. Questions and Technical Assistance

Please call or send a FAX to:

Fouad Bayomy (208) 885-6784 FAX (208) 885-6608

Appendix A

List of Errors and Possible Solutions

ERROR NUMBER	*POSSIBLE SOLUTION(S)*
7	• If you are running the program from the A drive, copy the program into a sperate directory in the C drive and try running it from there. • Divid your ETF file into two or more parts and try running each one alone. • Check system configurations.
14	• You have specified a long file name.
25, 57, 68	• Check the drive you are trying to read from.
52, 64	• Bad file name.
53	• File not found.
61	• Disk full.
70, 71	• Make sure the floppy disk is not write protected.
75, 76	• Invalid path for the file in use.
342	• Your ETF file is too large.

Appendix B

Example of an ETF File and the Corresponding Output FLX File

4 Example of ETF file for with 4 moduli values (excluding overlay)

101	454.500	281.	92.4	97.8	3.3
102	454.600	857.	252.1	87.7	2.9
103	454.700	336.	190.6	116.2	3.9
104	454.800	382.	154.5	95.2	3.2
106	454.900	427.	186.1	86.3	2.9
105	455.000	360.	144.2	83.7	2.8
106	455.000	618.	164.2	77.4	2.6
107	455.100	756.	239.5	70.6	2.5
108	455.200	765.	254.9	59.2	2.5
108	455.300	755.	223.2	76.7	2.6
109	455.400	811.	267.2	70.7	2.7
109	455.500	284.	98.0	125.2	4.2
110	455.600	249.	42.3	96.0	6.1
111	455.712	200.	143.2	77.0	6.3

INPUT FILE : EXAPPB.INP

DESCRIPTION : Example: to be run with ETF in Appendix B

1. SUMMARY OF INPUT DATA

1.1 TRAFFIC DATA

DESIGN DUAL TIRE LOAD = 4500
DESIGN DUAL TIRE SPACING = 13.5
TIRE PRESSURE (psi) = 80
DESIGN FUTURE TRAFFIC (ESALs) = 4000000
ESTIMATED PAST TRAFFIC (ESALs) = 0
FATIGUE SHIFT FACTOR FOR NEW ASPHALT = 18.4
FATIGUE SHIFT FACTOR FOR OLD ASPHALT = 8

1.2 SEASONAL VARIATION DATA

	WINTER	SPRING	SUMMER	FALL
SUBGRADE VARIATION	0.44	0.72	1.00	1.00
BASE/SBASE VARIATION	0.65	0.85	1.00	1.00
TRAFFIC VARIATION	1.00	1.00	1.00	1.00
TEMPERATURE VARIATION	48.00	59.00	65.00	34.00
PERIOD (MONTHS)	3.00	1.00	4.00	4.00

1.3 PAVEMENT DATA

CRACK INDEX = 2.5
CLIMATIC ZONE : 6
TEMPERATURE AT FWD TEST = 0
OLD AC BITUMEN VOLUME (Vb) % = 11
OLD AC AIR VOLUME (Va) % = 5

	POISSON RATIO	THICKNESS (in.)
OLD AC LAYER	0.20	03.50
BASE LAYER	0.35	05.50
SUBBASE LAYER	0.40	08.80
SUBGRADE	0.45	SEMI-INFINITE

SUBGRADE TYPE : LINEAR

BASE TYPE : LINEAR
SUB-BASE TYPE : LINEAR

OVERLAY MODULUS(ksi) = 350 AT TEMPERATURE (F) = 77
POISSON RATIO = .2
MINIMUM THICKNESS = 2
BITUMEN VOLUME (Vb) % = 11 AIR VOLUME (Va) % = 5

ETF FILE : EXAPPB.ETF
Example of ETF file for with 4 moduli values (excluding overlay)

CASE	MILE POST	TEMPERATURE	E1	E2	E3	E4	OVERLAY	DAMA1	DAMA2	DAMA3	DAMA4	DAMA22
1	454.5	101	281	92.4	97.8	3.3	3	.00156	.20522	0	.9893	0
2	454.6	102	857	252.1	87.7	2.9	2	.00523	.04812	0	.84948	0
3	454.7	103	336	190.6	116.2	3.9	2	.00001	.03526	0	.59307	0
4	454.8	104	382	154.5	95.2	3.2	2.5	.00001	.08011	0	.93573	0
5	454.9	106	427	186.1	86.3	2.9	3	.00005	.05104	0	.85059	0
6	455	105	360	144.2	83.7	2.8	3.5	.00002	.08082	0	.908	0
7	455	106	618	164.2	77.4	2.6	3.5	.00063	.08458	0	.9373	0
8	455.1	107	756	239.5	70.6	2.5	3.5	.00191	.04323	0	.81145	0
9	455.2	108	765	254.9	59.2	2.5	3.5	.00219	.04297	0	.97203	0
10	455.3	108	755	223.2	76.7	2.6	3	.00247	.05368	0	.90534	0
11	455.4	109	811	267.2	70.7	2.7	3	.00314	.03815	0	.8239	0
12	455.5	109	284	98	125.2	4.2	2	.00119	.21253	0	.72722	0

13	455.6	110	249	42.3	96	6.1	3	.01494	.87363	0	.77361	0
14	455.712	111	200	143.2	77	6.3	2	.01307	.07643	0	.88567	0

DAMA1 = FATIGUE DAMAGE ON OVERLAY
DAMA2 = FATIGUE DAMAGE ON OLD AC
DAMA3 = FATIGUE DAMAGE ON BTB
DAMA4 = RUTTING DAMAGE
DAMA22 = FATIGUE DAMAGE DUE TO PAST TRAFFIC

Appendix C

Example of Suggested Method of Analysis

APPENDIX C

Example of Suggested Method of Analysis

A suggested method for analyzing the FWD data to get the moduli values of the different layers and use them to create an .INP file for FLEXOLAY is presented here. FWD tests made on US-30, from milepost 447.00 to 455.00, were used. MODULUS backcalculation program was used to assist in obtaining E values. The following procedures were made twice, once for the FWD data made before an overlay was placed and the other time for the FWD data after the overlay was constructed.

Concept:

To determine the required overlay thickness, the backcalculated E values from FWD data using Modulus program were analyzed to establish representative E values for each pavement section. Then these E values were used in FLEXOLAY program for single design case to design an overlay for each section.

- Step 1: A run of MODULUS was made for the hole part of the road under study, and the variations of the moduli values were plotted. Also the variation of the last deflection, D7, was plotted. This was done for FWD data before the construction of the overlay (pre-construction).
- Step 2: By analyzing the variation of E values and D7, the road was divided into three sections. The first section was identified from mile-post 447.00 to 449.50. The second was identified from milepost 450.50 to 454.00. Finally the third section was from mile-post 454.50 456.00.
- Step 3: MODULUS program was used again separately for each section. Several runs were made with different seed values until the most satisfactory values were obtained. This step is done to refine the data file by eliminating data points with outliers. This step involves engineering judgment and is dependent on the pavement designer experience.
- Step 4: The 87.5 percentile values of the moduli were calculated to be used later as input in the FLEXOLAY program.
- Step 5: The E values obtained from step 4 along with all other necessary inputs were used to create the input file for FLEXOLAY (.INP). The thicknesses of the required overlay were obtained for all three sections.
- Step 6: The above 5 steps were repeated for FWD data made after the construction of the overlay (post-construction).

Results of the above analysis revealed the following overlay thicknesses:

Section	Overlay Thickness (in) (Pre-construction)	Overlay Thickness (in) (Post-construction)
1	6.5	5
2	5	3
3	6	5

As noticed the average overlay thickness for the original pavement section (pre-construction) is 5.8″. The procedure indicated 4.33″ still needed for the assumed traffic and design life. It is to be noted that for this particular pavement section, the existing overlay is 3.75″ which is about 2″ smaller than the needed one at the time of construction. Flexolay, however, predicted a need of additional 4.33″ on to be added to new overlay. This makes a difference of almost 50%. It is believed that this difference is attributed to the shift factors used and inherited variability in backcalculated E values from FWD results.

If the variation of the overlay with the milepost is required prepare .ETF files (as was shown in Appendix B) for FLEXOLAY for each section, pre-construction and post-construction. Run FLEXOLAY and compare the overlays of pre-construction and post-construction at the different mileposts.

The next few pages are the outputs of MODULUS that were used in FLEXOLAY for the pre-construction case only.

ALLPRE

TTI MODULUS ANALYSIS SYSTEM (SUMMARY REPORT) (Version 4.2)

District: 5
Temp. : 105
Highway/Road: US030

		Thickness(in)	MODULI RANGE(psi) Minimum	Maximum	Poisson Ratio Values
Pavement	H1:	5.2	200,000	2,000,000	0.35
Base:	H2:	3.6	40,000	800,000	0.35
Subbase:	H3:	10.8	10,000	150,000	0.45
Subgrade	H4:	59.9	10,000		0.45

Station	(lbs)	Load R1	Measured R2	Deflectio R3	(mils): R4	R5	R6	R7	(E1)	Calculated Moduli (E2)	values (E3)	(ksi): (E4)	ERR/Sens	Absolute Dpth to Bedrock
447.3	11,881	25.33	20.62	16.77	11.74	8.22	4.2	2.07	810	209	17.9	2.7	5.52	76.57
447.4	11,817	11.22	8.15	6.14	4.05	2.94	2	1.56	869	110.2	150	5.9	2.2	95.77
447.5	11,877	12.89	10.08	8.29	6.1	4.61	3.25	2.25	1134	389.5	92.5	3.9	3.14	99.07
447.6	11,714	13.74	10.64	8.54	6.02	4.53	3.11	2.08	734	238.9	109.2	3.6	1.53	96.12
447.7	11,936	11.61	8.52	6.65	4.65	3.37	2.44	1.89	698	203.8	149.4	5	2.46	117.92
447.8	11,694	17.33	13.46	10.42	7.18	5.49	3.54	2.13	427	293.8	82.1	3.3	1.73	87.63
447.9	11,786	15.9	12.2	9.77	7.11	5.57	3.85	2.36	848	211.6	90.6	3	2.45	86.57
448	8,385	15.96	12.67	9.72	6.72	5.05	3.14	1.74	386	261.3	53.8	2.6	1.91	84.71
448	11,658	21.17	16.97	13.3	9.5	7.24	4.6	2.56	520	244.9	57.4	2.4	1.8	84.1
448.1	11,722	19.53	14.86	11.41	8.18	6.48	4.5	2.55	813	90.1	73.4	2.7	3.59	82.73
448.201	11,738	15.22	12.46	10.37	7.63	5.87	3.94	2.49	918	318.7	72.2	3.2	4.84	90.68
448.3	11,658	16.22	13.37	11.37	8.49	6.78	4.8	3.07	868	276	64.3	2.9	8.32	93.71
448.404	11,563	26.22	20.33	16.18	11.75	8.98	5.66	2.94	298	191.4	57.4	1.9	2.13	83.43
448.5	11,714	16.52	12.26	9.53	6.89	5.43	3.86	2.28	683	152.3	94.2	3.1	3.64	83.16
448.6	11,408	19.04	14.58	11.78	8.91	6.96	4.42	2.24	732	252	62.7	2.5	2.6	77.56
448.705	11,611	18.57	13.52	11.11	8.28	6.36	4.35	2.67	635	192.9	79.2	2.6	2.9	89.16
448.8	11,591	20.63	16.43	13.26	9.9	7.78	5.56	3.39	628	231.5	58.7	2.3	4.09	89.47
448.9	11,611	17.8	13.65	11.07	8.31	6.63	4.74	2.87	788	249.6	74.4	2.6	4.04	86.39
449	8,306	13.68	11.31	9.26	6.8	5.19	3.54	2.02	739	246.2	58.2	2.5	3.59	85.43
449	11,631	18.24	15.09	12.52	9.5	7.35	5.09	2.86	775	257.1	60	2.6	6.59	82.67
449.1	11,579	18.84	14.81	12.07	9.16	7.33	5.15	3.15	772	260	58.9	2.6	5.59	87.84
449.2	11,269	21.15	16.64	13.4	9.93	7.28	5.27	3.37	629	225.6	53.9	2.3	2.1	95.64
449.303	11,360	17.3	13.61	10.97	7.79	5.99	4.2	2.47	629	231.9	80.2	2.7	2.03	87.05
449.401	11,265	16.06	13.35	11.18	8.04	5.59	3.65	1.88	883	294.3	62.8	2.9	3.04	80.66

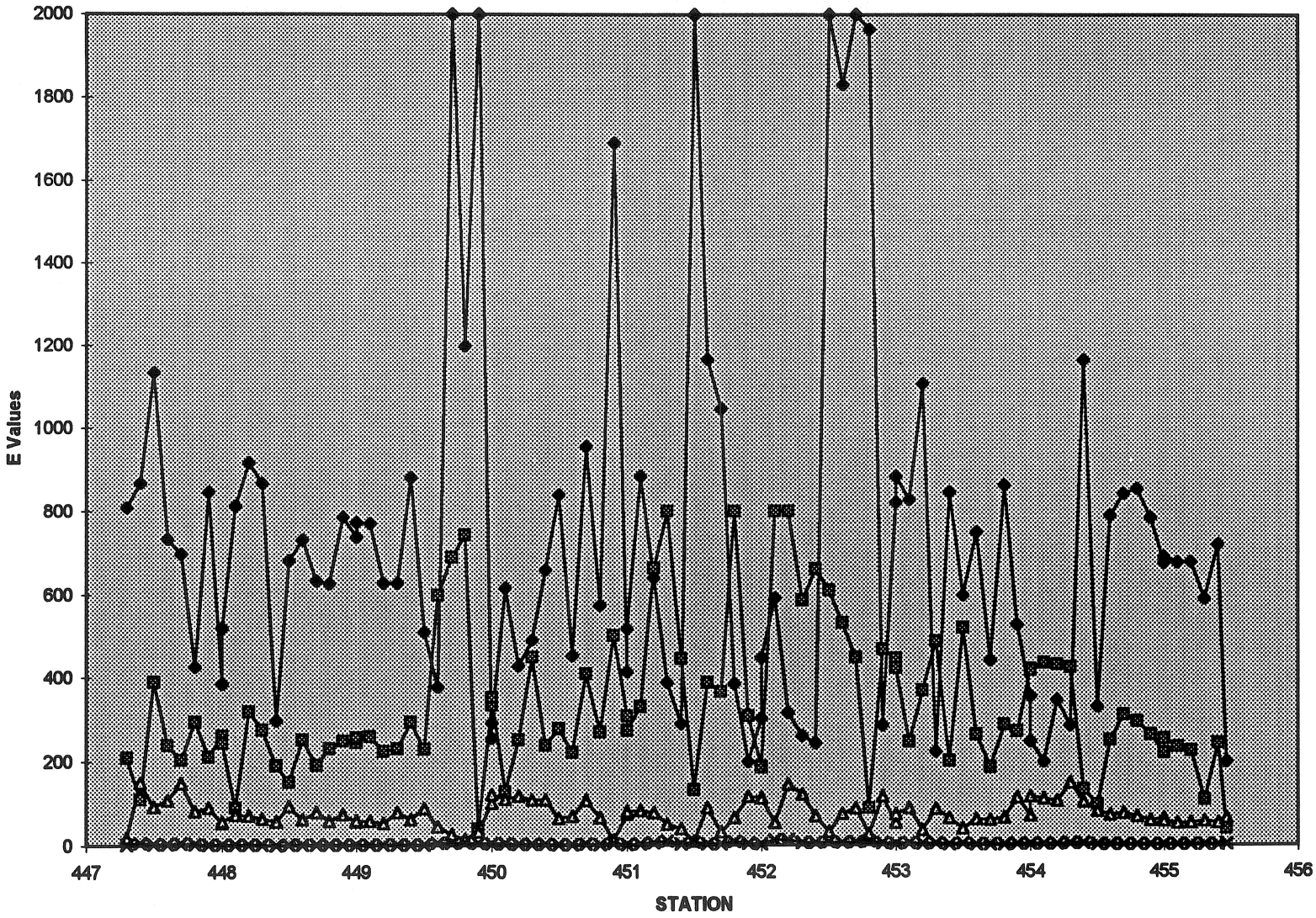
ALLPRE

449.5	11,293	16.61	12.55	9.91	7.29	5.49	3.76	2.15	511	230.1	89.4	3	1.79	81.65
449.6	11,269	14.83	10.89	8.03	4.97	3.34	1.67	0.84	378	600.2	45.3	6	2.98	70.37
449.704	11,364	9.94	8.06	6.58	4.74	3.31	1.7	0.82	2000	690.5	27.6	6.9	5.82	66.56
449.8	11,364	12	9.17	7.22	4.77	2.62	1.48	0.77	1200	743.9	15.2	7.4	3.41	69.26
449.9	11,583	12.66	10.31	7.64	5.04	3.66	2.36	1.43	2000	40	33.5	7	14.15	83.97
450	8,174	10.83	7.59	5.67	3.83	2.72	1.59	0.78	256	335.3	102.2	4.9	1.75	75.01
450	11,464	13.84	9.83	7.52	5.24	3.76	2.28	1.09	293	353.1	123.6	4.8	1.63	73.02
450.1	11,353	12.66	9.09	6.63	4.41	3.11	1.96	1.22	618	129.9	111.8	5.7	1.35	82.78
450.2	11,293	13.96	10.22	8.09	5.69	4.21	2.8	1.44	429	251.9	118.4	3.9	1.56	76.51
450.301	11,547	12.9	9.81	7.75	5.44	3.97	2.46	1.39	492	450.6	109.9	4.5	1.97	79.44
450.4	11,480	13.84	10.47	8.18	5.96	4.57	3.14	1.81	660	239.5	109.6	3.7	2.1	79.44
450.5	11,452	16.87	13.47	11.01	8.5	6.86	4.87	2.92	841	278.3	65.7	2.8	6.98	83.84
450.6	11,337	20.17	15.42	12.43	9.32	7.05	4.44	2.42	454	222.2	71.1	2.4	2.42	81.7
450.7	11,349	11.73	9.13	7.43	5.46	4.14	2.61	1.63	958	408.8	108.3	4.1	2.44	82.15
450.801	11,337	18.32	14.43	11.57	8.55	6.26	3.95	2.34	576	271	66.7	2.7	2.05	85.96
450.9	11,464	13.44	10.8	8.85	6.27	4.43	2.07	0.48	1692	502.8	12.6	5.6	6.37	66.66
451	8,214	12.85	10.02	7.8	5.49	4.13	2.56	1.22	414	308.6	74.4	3.1	2	76.56
451	11,444	17.32	13.78	10.69	7.88	6.26	3.83	2.04	519	274.8	82.6	2.8	2.35	77.87
451.1	11,448	14.5	11.46	9.32	6.91	5.08	3.33	2	886	330.6	83.2	3.3	1.78	83.88
451.201	11,460	11.05	8.43	6.22	4.31	2.85	1.66	0.87	641	665.1	79	6.7	0.79	72.59
451.301	11,547	11.6	7.87	5.66	3.48	2.12	0.85	0.26	388	800	50.9	9.9	5.06	61.22
451.403	11,400	19.25	13.99	10.52	6.73	4.52	2.28	1.12	290	445.5	40.5	4.5	3.16	72.98
451.5	11,488	13.96	11.03	8.88	6.26	4.15	2.17	0.94	2000	133.5	10	7.2	17.88	71.03
451.601	11,436	11.42	9.22	8.03	6.41	5.22	3.39	1.81	1167	388.9	90.3	3.9	10.18	72.97
451.7	11,388	17.31	13.26	10.65	7.6	5.53	3	0.97	1048	365.9	32.7	3.7	3.89	70.48
451.8	11,444	11.13	7.69	5.66	3.44	2.18	1.03	0.34	386	800	67.1	9.2	3.29	65.46
451.902	11,412	14.55	9.46	7.13	4.74	3.22	1.81	0.93	200	308.9	118.3	5.8	2.06	74.36
452	8,099	11.24	8.01	6.1	4.25	3.18	1.99	1.16	304	188.4	115	3.9	1.69	82.69
452	11,432	14.85	10.68	8.22	5.98	4.57	2.89	1.72	449	187.4	114.3	3.8	1.98	81.57
452.1	11,571	9.1	6.22	4.51	2.57	1.39	0.68	0.37	593	800	54.6	13.5	2.42	65.2
452.2	11,400	8.31	5.27	3.86	2.34	1.49	0.72	0.36	316	800	145.9	13.1	3.21	62.7
452.3	11,388	12.78	8.85	6.89	4.5	3.18	1.85	1.05	261	587.9	122.6	5.9	2.41	77.28
452.401	11,384	14.31	9.68	7.21	4.7	3.11	1.39	0.64	245	662.4	70.9	6.6	4.91	65.08
452.5	11,444	9.33	7.18	5.75	3.99	2.76	1.4	0.63	2000	610.8	33	8.1	4.31	65.65
452.6	11,468	9.61	7.44	6.09	4.43	3.25	1.89	0.89	1833	531.8	76	6.1	3.43	69.37
452.7	11,547	8.42	6.45	5.19	3.6	2.55	1.38	0.74	2000	448.2	88.7	7.7	4.13	67.27
452.8	11,416	12.09	9.64	6.41	3.67	2.23	1.08	0.52	1965	89.1	27.9	9.1	2.23	66.36

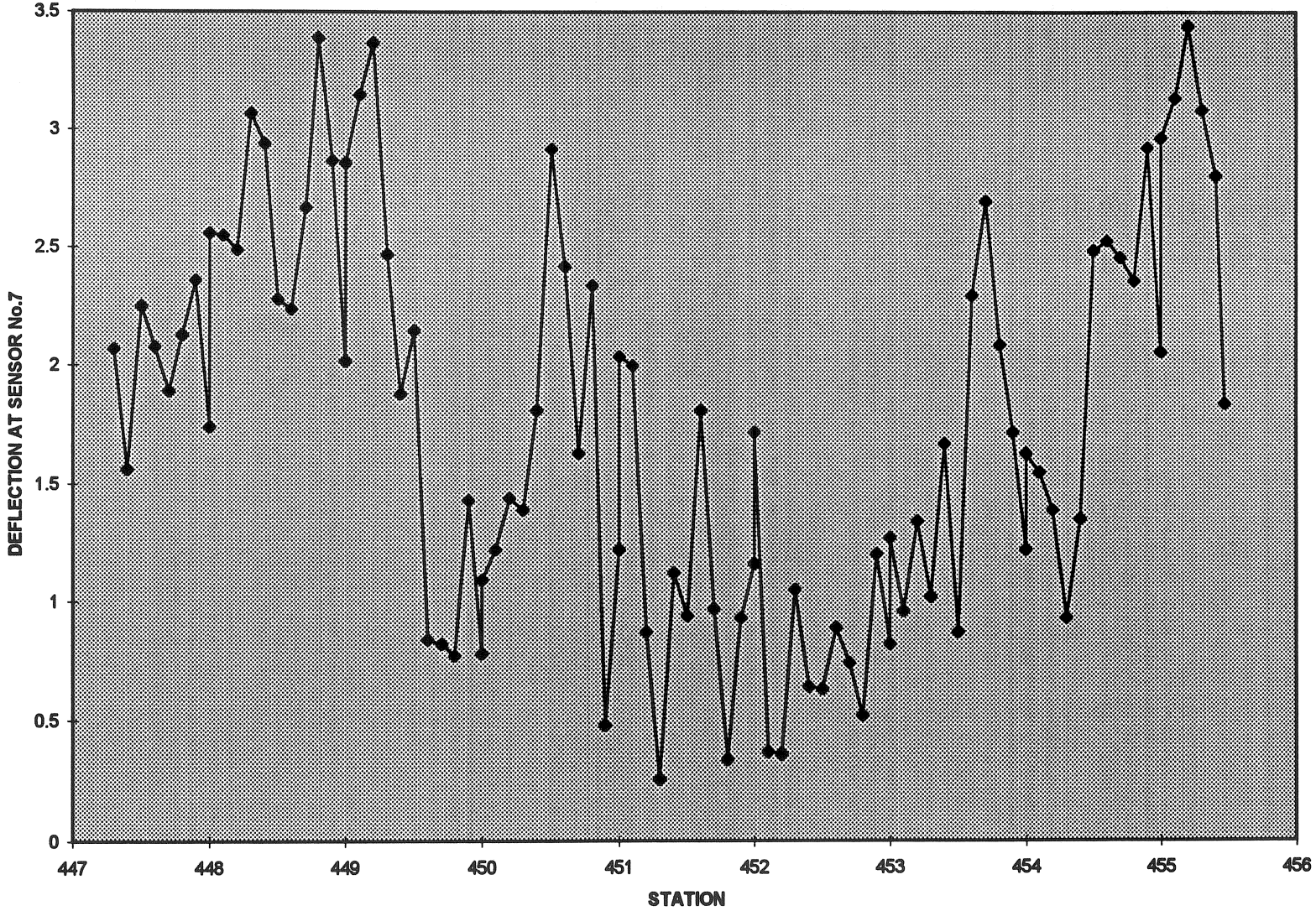
ALLPRE

452.9	11,368	13.93	9.95	7.81	5.42	3.91	2.3	1.2	287	468.4	117.2	4.7	2.36	76.96
453	7,960	9.99	7.65	6.14	4.24	2.95	1.72	0.82	822	445.4	53.6	4.5	2.6	75.82
453	11,309	13.15	10.13	8.24	5.9	4.24	2.55	1.27	884	423	74.5	4.2	2.87	75.26
453.1	11,396	12.35	9.47	7.04	4.87	3.62	2.09	0.96	828	247.7	88.9	5.2	2.33	70.7
453.2	11,273	15.37	12.62	10.52	7.66	5.14	3.09	1.34	1108	369.3	36.9	3.7	4.15	74.36
453.3	11,293	15.87	11.17	8.38	5.57	3.87	2.19	1.02	224	487.9	87	4.9	1.96	74.97
453.4	11,253	17.22	13.69	11.14	8.16	6.07	3.67	1.67	846	201.7	65.7	2.8	3.33	75.4
453.5	11,372	14.85	11.29	8.76	5.63	3.65	2.1	0.87	599	520.3	40.8	5.2	1.64	74.15
453.6	11,233	17.71	14.14	11.78	8.78	6.22	4.15	2.3	749	264.1	63.2	2.6	2.34	83.12
453.702	11,186	22.55	17.06	13.87	10.54	7.91	5.05	2.7	442	186.4	61.9	2.1	2.61	82.1
453.804	11,452	15.43	12.99	11.08	8.57	6.7	4.21	2.09	863	287.5	66.8	2.9	7.57	75.82
453.901	11,269	12.93	9.76	7.57	5.25	3.89	2.57	1.72	528	272.7	114.9	4.3	1.39	92.12
454	8,107	11.57	8.52	6.69	4.5	3.15	1.84	1.22	356	421.5	71.4	4.2	2.37	80.68
454	11,392	15.08	10.76	8.54	5.97	4.27	2.59	1.63	248	417.1	118.2	4.2	2.11	82.05
454.1	11,337	15.9	10.96	8.68	5.75	4.13	2.52	1.55	200	435.8	112.4	4.4	2.17	85.58
454.2	11,396	14.19	10.43	8.27	5.76	4.19	2.48	1.39	346	432.1	107.6	4.3	2.59	79.03
454.3	11,392	12.28	8.15	6.46	4.64	3.37	1.89	0.93	286	425.6	150	5.6	3.46	71.16
454.402	11,293	11.71	8.99	6.75	4.69	3.42	2.13	1.35	1165	134.1	105.1	5.1	1.63	80.15
454.502	10,955	20.83	13.95	10.57	7.88	6.09	4.12	2.49	331	98.1	82.7	2.8	4.81	85.74
454.6	11,237	16.93	13.11	10.83	8.23	6.41	4.3	2.53	790	250.6	72.6	2.6	2.84	84.26
454.701	11,186	14.34	12.02	10.12	7.22	5.56	3.83	2.46	842	311.8	77.6	3.1	4.92	95.21
454.8	11,412	16.26	12.65	10.32	7.73	5.98	4.01	2.36	854	295.6	69.9	3	2.87	83.7
454.9	11,368	18.23	14.1	11.77	8.97	7.08	4.89	2.93	785	264.5	59.9	2.6	5.96	86.96
455	7,948	14.19	10.49	8.68	6.62	5.26	3.59	2.06	676	221.5	65.3	2.3	3.96	84.89
455	11,253	18.75	14.43	12.17	9.45	7.59	5.2	2.97	691	255.1	58.5	2.6	7.85	83.58
455.1	11,257	20.53	15.47	12.94	9.89	7.88	5.47	3.14	677	235.8	53.5	2.4	6.12	85.83
455.2	11,217	20.49	15.48	13.19	10.2	8.22	5.8	3.44	679	226.3	55.3	2.3	7.12	88.5
455.3	11,110	23.13	18.19	13.81	10.3	8.18	5.55	3.09	591	109.7	59	2	2.44	81.49
455.4	11,281	19.69	15.36	12.9	9.77	7.65	4.94	2.81	721	244	54.4	2.4	5.3	85.53
455.467	11,102	25.96	15.92	10.67	6.73	4.75	3.06	1.84	200	40	66.2	3.7	4.35	89.29
Mean:		15.4	11.71	9.3	6.66	4.95	3.15	1.77	737	330.8	76	4.3	3.65	79.51
Std.	Dev:	3.9	3.11	2.63	2.13	1.79	1.33	0.84	450	178	31.2	2.2	2.61	9.01
Var	Coeff(%):	25.3	26.59	28.29	32.02	36.1	42.25	47.38	61	53.8	41.1	52.1	71.51	11.33

PRECONSTUCTION MODULI DISTRIBUTION



DISTRIBUTION OF D7



SEC1PRE

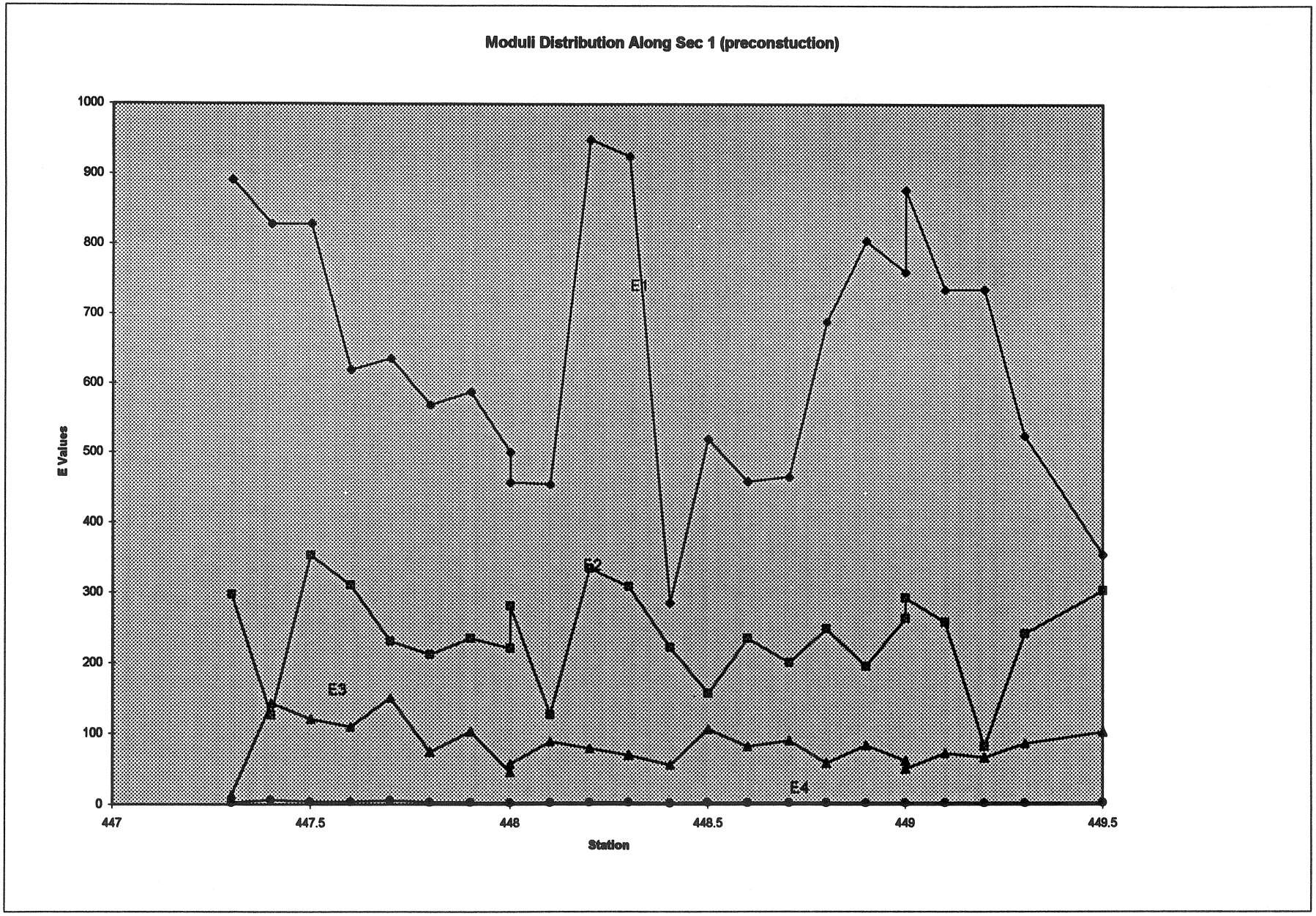
TTI MODULU ANALYSI SYSTEM (SUMMAR REPORT) (Version 4.2)

District: 5
Temp. : 85
Highway/Road: US030

	Thickness(in)	MODULI Minimum	RANGE(psi) Maximum	Poisson Ratio Values
Pavement:	H1: 5.2	200,000	2,000,000	0.35
Base:	H2: 3.6	40,000	800,000	0.35
Subbase:	H3: 10.8	10,000	150,000	0.45
Subgrade:	H4: 67	10,000		0.45

Station	Load (lbs)	R1	R2	R3	R4	R5	R6	R7	Calculated SURF(E1)	Moduli BASE(E2)	values SUBB(E3)	(ksi): SUBG(E4)	ERR/Sens	Absolute Dpth to Bedrock
447.3	11,881	25.33	20.62	16.77	11.74	8.22	4.2	2.07	892	297.4	13.5	3	4.9	76.57
447.4	11,817	11.22	8.15	6.14	4.05	2.94	2	1.56	828	125.9	142.8	6.8	2.15	95.77
447.5	11,877	12.89	10.08	8.29	6.1	4.61	3.25	2.25	829	353.3	120.6	4	1.3	99.07
447.6	11,714	13.74	10.64	8.54	6.02	4.53	3.11	2.08	619	311	109.7	4.2	1.45	96.12
447.7	11,936	11.61	8.52	6.65	4.65	3.37	2.44	1.89	636	231.1	150	5.8	2.34	117.92
447.8	11,694	17.33	13.46	10.42	7.18	5.49	3.54	2.13	569	212.2	74.1	3.7	1.84	87.63
447.9	11,786	15.9	12.2	9.77	7.11	5.57	3.85	2.36	588	235.6	102.6	3.4	1.98	86.57
448	8,385	15.96	12.67	9.72	6.72	5.05	3.14	1.74	501	221.2	45.9	3	1.83	84.71
448	11,658	21.17	16.97	13.3	9.5	7.24	4.6	2.56	458	281.6	57.5	2.8	1.85	84.1
448.1	11,722	19.53	14.86	11.41	8.18	6.48	4.5	2.55	455	126.2	88.7	3	2.06	82.73
448.201	11,738	15.22	12.46	10.37	7.63	5.87	3.94	2.49	950	334.5	79.3	3.3	2.04	90.68
448.3	11,658	16.22	13.37	11.37	8.49	6.78	4.8	3.07	926	308.7	69.9	3.1	5.94	93.71
448.404	11,563	26.22	20.33	16.18	11.75	8.98	5.66	2.94	285	222.3	55.9	2.2	2.14	83.43
448.5	11,714	16.52	12.26	9.53	6.89	5.43	3.86	2.28	520	156.9	106.1	3.5	2.74	83.16
448.6	11,408	19.04	14.58	11.78	8.91	6.96	4.42	2.24	459	235.2	81.8	2.7	2.43	77.56
448.705	11,611	18.57	13.52	11.11	8.28	6.36	4.35	2.67	466	201.1	89.6	3	2.36	89.16
448.8	11,591	20.63	16.43	13.26	9.9	7.78	5.56	3.39	689	249.1	59.1	2.5	2.78	89.47
448.9	11,611	17.8	13.65	11.07	8.31	6.63	4.74	2.87	806	195.5	83.5	2.8	2.84	86.39
449	8,306	13.68	11.31	9.26	6.8	5.19	3.54	2.02	761	263.8	62.5	2.6	1.58	85.43
449	11,631	18.24	15.09	12.52	9.5	7.35	5.09	2.86	879	292.9	50.7	2.9	5.63	82.67
449.1	11,579	18.84	14.81	12.07	9.16	7.33	5.15	3.15	736	258.2	72.4	2.6	2.73	87.84
449.2	11,269	21.15	16.64	13.4	9.93	7.28	5.27	3.37	736	81.6	66.5	2.5	2.06	95.64
449.303	11,360	17.3	13.61	10.97	7.79	5.99	4.2	2.47	525	242.3	86.1	3	1.63	87.05
449.5	11,293	16.61	12.55	9.91	7.29	5.49	3.76	2.15	355	303.8	102.1	3.4	1.24	81.65
						The 87.5% value of E's =			457.625	153.0625	55.25	2.5875		
Mean:		17.47	13.69	11	8	6.1	4.1	2.44	663	244.5	80.3	3.3	2.54	86.61
Std.	Dev:	3.67	3.02	2.47	1.86	1.43	0.92	0.5	208	71.9	31.5	1	1.24	7.23
Var	Coeff(%):	21.02	22.04	22.49	23.3	23.41	22.46	20.43	31	29.4	39.2	30.5	48.92	8.35

SEC1PRE

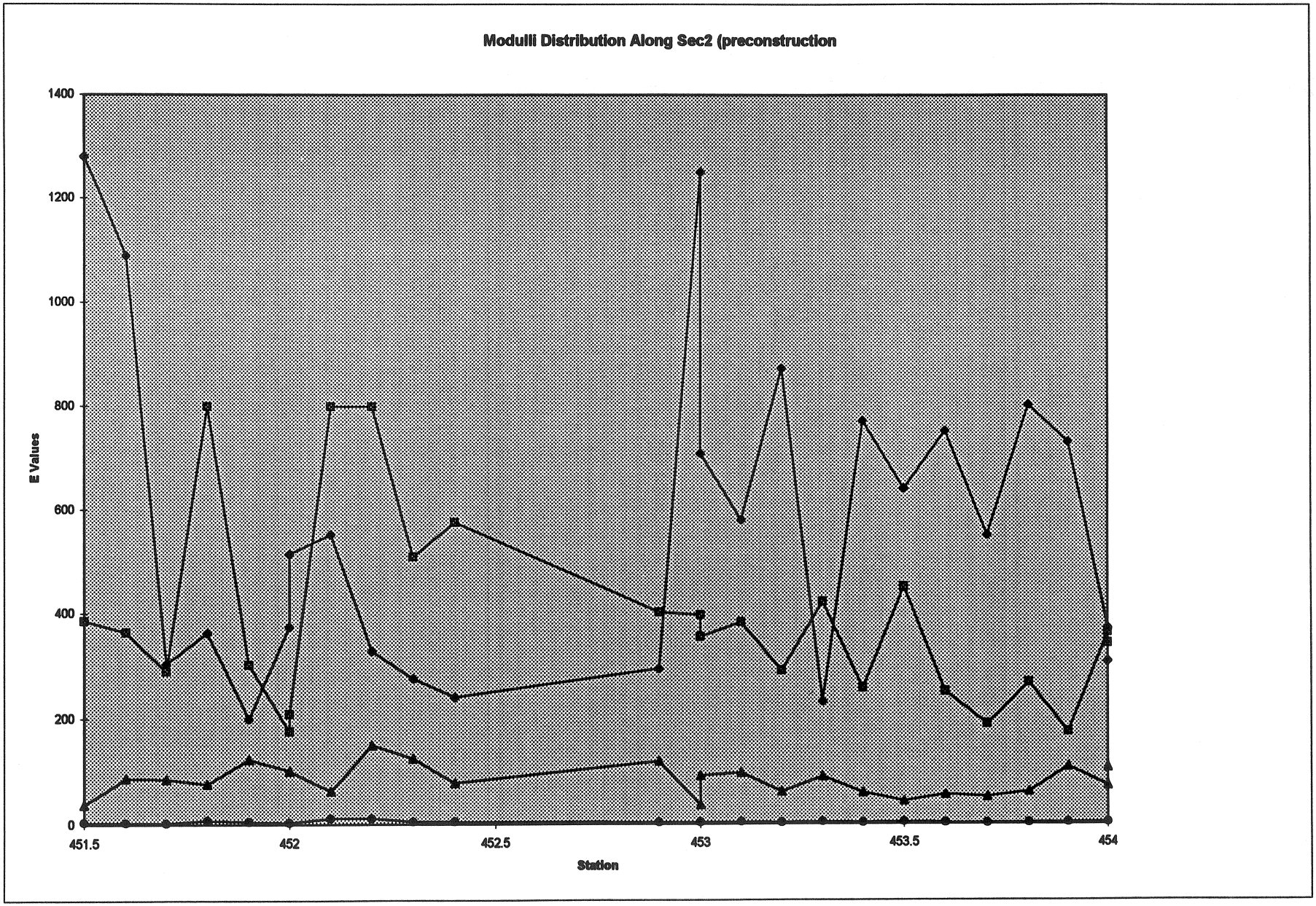

Moduli Distribution Along Sec 1 (preconstuction)
E Values
Station
E1
E2
E3
E4
0
100
200
300
400
500
600
700
800
900
1000
447
447.5
448
448.5
449
449.5

SEC2PRE

TTI MODULU ANALYSI SYSTEM (SUMMAR REPORT) (Version 4.2)

District:	5								MODULI	RANGE(psi)		Poisson Ratio		
Temp. :	95				Thickness(in)				Minimum	Maximum		Values		
Highway/Road:		US030		Pavement:	H1:		5.2		200,000	2,000,000		0.35		
				Base:	H2:		3.6		40,000	800,000		0.35		
				Subbase:	H3:		10.8		10,000	150,000		0.45		
				Subgrade:			53.4		10,000	H4:		0.45		Absolute Dpth to Bedrock
Station	Load (lbs)	Measured R1	Deflection R2	(mils): R3	R4	R5	R6	R7		Calculated SURF(E1)	Moduli BASE(E2)	values SUBB(E3)	(ksi): SUBG(E4)	ERR/Sens
451.5	11,488	13.96	11.03	8.88	6.26	4.15	2.17	0.94		1281	385.5	37.9	4.3	4.58	71.03
451.601	11,436	11.42	9.22	8.03	6.41	5.22	3.39	1.81		1090	363.2	88.8	3.6	12.58	72.97
451.7	11,388	17.31	13.26	10.65	7.6	5.53	3	0.97		304	289.3	86.8	2.9	5.03	70.48
451.8	11,444	11.13	7.69	5.66	3.44	2.18	1.03	0.34		363	800	77.5	8	3.03	65.46
451.902	11,412	14.55	9.46	7.13	4.74	3.22	1.81	0.93		200	302	123.5	5	1.93	74.36
452	8,099	11.24	8.01	6.1	4.25	3.18	1.99	1.16		374	176.8	103.4	3.4	1.98	82.69
452	11,432	14.85	10.68	8.22	5.98	4.57	2.89	1.72		515	208.6	101.9	3.4	2.83	81.57
452.1	11,571	9.1	6.22	4.51	2.57	1.39	0.68	0.37		553	800	64	11.7	2.73	65.2
452.2	11,400	8.31	5.27	3.86	2.34	1.49	0.72	0.36		328	800	150	11.4	2.86	62.7
452.3	11,388	12.78	8.85	6.89	4.5	3.18	1.85	1.05		276	508.9	125.4	5.1	2.35	77.28
452.401	11,384	14.31	9.68	7.21	4.7	3.11	1.39	0.64		241	575.7	79.7	5.8	4.64	65.08
452.9	11,368	13.93	9.95	7.81	5.42	3.91	2.3	1.2		295	402.2	120.6	4	2.32	76.96
453	7,960	9.99	7.65	6.14	4.24	2.95	1.72	0.82		1253	397.1	37.1	4.2	3.26	75.82
453	11,309	13.15	10.13	8.24	5.9	4.24	2.55	1.27		709	355	92.9	3.5	2.95	75.26
453.1	11,396	12.35	9.47	7.04	4.87	3.62	2.09	0.96		581	383.9	98.7	4.5	2.14	70.7
453.2	11,273	15.37	12.62	10.52	7.66	5.14	3.09	1.34		875	291.6	62.3	2.9	4.58	74.36
453.3	11,293	15.87	11.17	8.38	5.57	3.87	2.19	1.02		232	422	91.3	4.2	1.88	74.97
453.4	11,253	17.22	13.69	11.14	8.16	6.07	3.67	1.67		773	257.7	60.5	2.6	3.18	75.4
453.5	11,372	14.85	11.29	8.76	5.63	3.65	2.1	0.87		641	451.6	44.5	4.5	1.57	74.15
453.6	11,233	17.71	14.14	11.78	8.78	6.22	4.15	2.3		753	251	56.8	2.5	5.01	83.12
453.702	11,186	22.55	17.06	13.87	10.54	7.91	5.05	2.7		552	189.5	52.5	1.9	2.57	82.1
453.804	11,452	15.43	12.99	11.08	8.57	6.7	4.21	2.09		805	268.2	62.3	2.7	10.1	75.82
453.901	11,269	12.93	9.76	7.57	5.25	3.89	2.57	1.72		733	175.7	110.6	3.7	1.58	92.12
454	8,107	11.57	8.52	6.69	4.5	3.15	1.84	1.22		373	364.4	74.5	3.6	2.29	80.68
454	11,392	15.08	10.76	8.54	5.97	4.27	2.59	1.63		309	343.8	108.4	3.6	2.12	82.05
							The 87.5% value		=	276	208.6	52.5	2.7		
Mean:		13.32	9.98	7.87	5.5	3.92	2.3	1.17		727	402.6	82.7	4.8	3.62	73.04
Std.	Dev:	3.21	2.64	2.31	1.93	1.53	1.06	0.59		497	197.6	31.1	2.4	2.41	6.58
Var	Coeff(%):	24.08	26.51	29.35	35	39.14	46.02	50.3		68	49.1	37.6	50.3	66.67	9.01

SEC2PRE


Modulli Distribution Along Sec2 (preconstruction
1400
1200
1000
800
600
400
200
0
E Values
451.5
452
452.5
453
453.5
454
Station

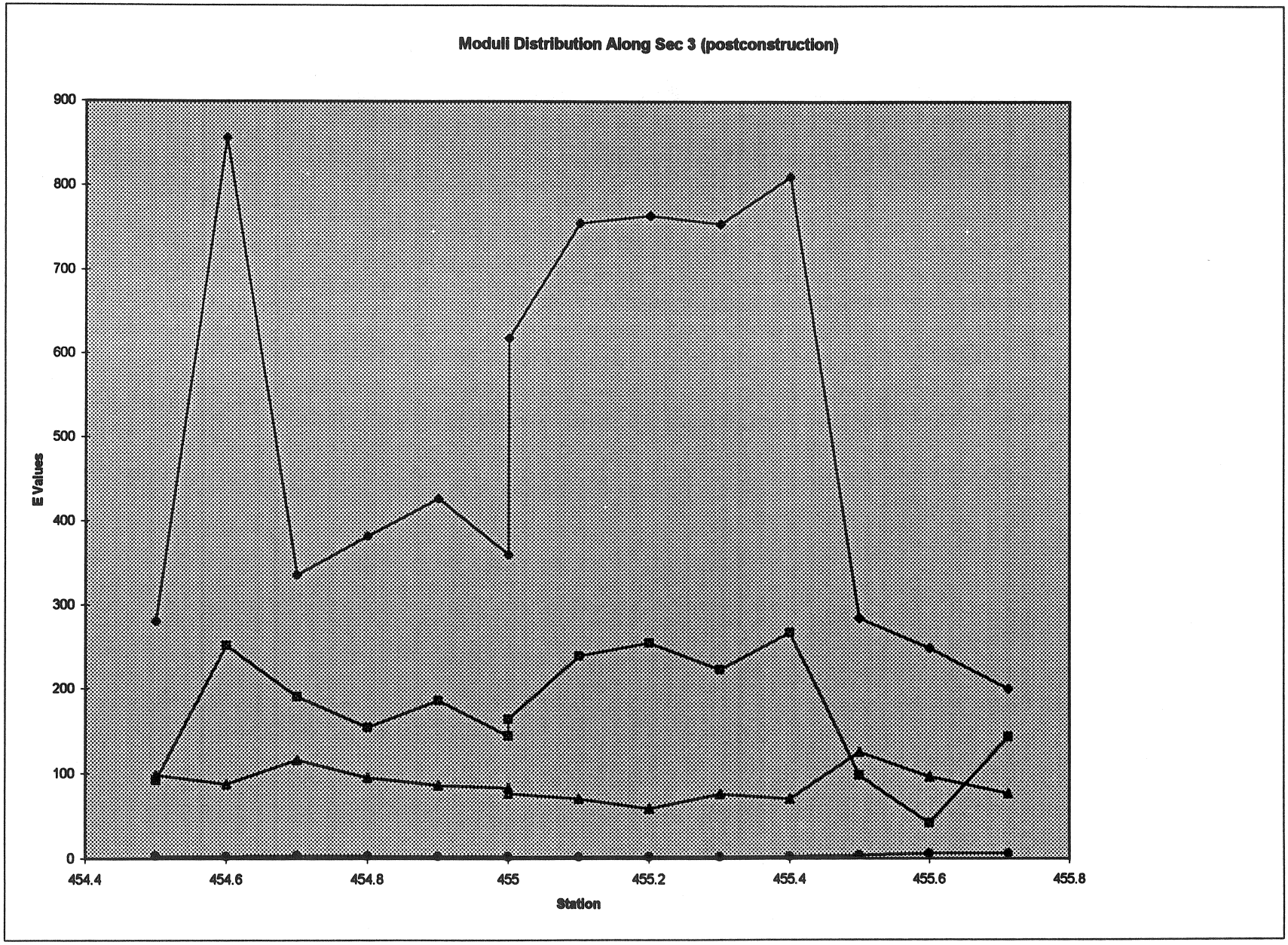
TTI MODULU ANALYSI SYSTEM (SUMMAR REPORT) (Version 4.2)

District: 5
Temp. : 107
Highway/Road: US030

		Thickness(in)	MODULI RANGE(psi) Minimum	Maximum	Poisson Ratio Values
Pavement:	H1:	5.2	200,000	2,000,000	0.35
Base:	H2:	3.6	40,000	800,000	0.35
Subbase:	H3:	10.8	10,000	150,000	0.45
Subgrade:	H4:	61.6	10,000		0.45

Station	Load (lbs)	Measured Deflection (mils): R1	R2	R3	R4	R5	R6	R7	Calculated Moduli values (ksi): SURF(E1)	BASE(E2)	SUBB(E3)	SUBG(E4)	ERR/Sens	Absolute Dpth to Bedrock
454.5	11,901	21.48	13.67	10.51	7.53	5.84	3.84	2.38	281	92.4	97.8	3.3	4.59	90.11
454.6	11,952	16.89	12.19	10.15	7.8	6.09	4.33	2.56	857	252.1	87.7	2.9	4.17	82.78
454.7	12,135	16.75	11.78	9.15	6.62	4.85	3.19	2.07	336	190.6	116.2	3.9	1.39	90.84
454.8	12,139	19.07	13.76	10.83	7.59	5.73	4.16	2.14	382	154.5	95.2	3.2	3.07	79.51
454.9	12,068	19.54	14.35	11.24	8.46	6.43	4.28	2.34	427	186.1	86.3	2.9	2.15	78.92
455	9,303	16.37	11.71	9.26	6.65	5.1	3.52	2.05	360	144.2	83.7	2.8	2.96	86.52
455	12,592	20.93	15.7	12.6	9.28	7.13	5.07	2.91	618	164.2	77.4	2.6	2.59	84.31
455.1	11,917	19.36	14.83	11.87	9.22	7.27	5.14	2.97	756	239.5	70.6	2.5	3.87	81.14
455.2	11,929	19.1	14.97	12.78	9.92	7.86	5.89	2.8	765	254.9	59.2	2.5	7.96	76.25
455.3	11,901	19.06	14.22	11.91	7.96	7.1	5.26	3.44	755	223.2	76.7	2.6	7.03	300
455.4	12,087	18.45	14.9	11.74	8.29	6.74	4.8	2.95	811	267.2	70.7	2.7	3.07	300
455.5	11,678	17.68	11.24	8.2	5.91	4.63	3.11	1.88	284	98	125.2	4.2	5.49	82.38
455.6	11,543	19.15	11.54	7.72	4.67	3.11	1.78	0.96	249	42.3	96	6.1	2.14	77.11
455.712	10,165	15.57	9.72	6.87	4.23	2.89	1.41	0.62	200	143.2	77	6.3	3.56	69.2
								The 87.5% values =	269	95.9	70.6625	2.5625		
Mean:		18.53	13.18	10.35	7.44	5.77	3.98	2.29	506	175.2	87.1	3.5	3.86	81.16
Std.	Dev:	1.7	1.79	1.86	1.68	1.51	1.29	0.77	240	68.2	18	1.3	1.88	23.94
Var	Coeff(%):	9.16	13.6	17.99	22.53	26.19	32.5	33.82	48	39	20.6	36.8	48.7	29.5

Sec3pre


Moduli Distribution Along Sec 3 (postconstruction)
E Values
900
800
700
600
500
400
300
200
100
0
454.4
454.6
454.8
455
455.2
455.4
455.6
455.8
Station