ANL/EES-TM-255

BREEDER: A Microcomputer Program for Financial Analysis of a Large-Scale Prototype Breeder Reactor

R. F. Giese

RETURN TO REFERENCE FILE
TECHNICAL PUBLICATIONS
DEPARTMENT

ANL
EES

ARGONNE NATIONAL LABORATORY
Energy and Environmental Systems Division

Operated by
THE UNIVERSITY OF CHICAGO for U. S. DEPARTMENT OF ENERGY
under Contract W-31-109-Eng-38

Argonne National Laboratory, with facilities in the states of Illinois and Idaho, is owned by the United States government, and operated by The University of Chicago under the provisions of a contract with the Department of Energy.

LEGAL NOTICE

This report was prepared by Argonne National Laboratory, operated by the University of Chicago on behalf of the U. S. Department of Energy (DOE), as an account of work sponsored by the Electric Power Research Institute, Inc. (EPRI). Neither EPRI, members of EPRI, the University of Chicago, DOE, the U. S. Government, nor any person acting on their behalf: (a) makes any warranty or representation, express or implied, with respect to the information contained in this report; or (b) assumes any liabilities with respect to the use of, or for damages resulting from the use of, any information disclosed in this report.

This informal report presents preliminary results of ongoing work or work that is more limited in scope and depth than that described in formal reports issued by the Energy and Environmental Systems Division.

Printed in the United States of America. Available from National Technical Information Service, U. S. Department of Commerce, 5285 Port Royal Road, Springfield, Virginia 22161.

ARGONNE NATIONAL LABORATORY
9700 South Cass Avenue
Argonne, Illinois 60439

ANL/EES-TM-255

BREEDER: A MICROCOMPUTER PROGRAM FOR FINANCIAL ANALYSIS OF A LARGE-SCALE PROTOTYPE BREEDER REACTOR

by

Robert F. Giese

Energy and Environmental Systems Division
Special Projects and Industrial Applications Group

April 1984

work sponsored by

ELECTRIC POWER RESEARCH INSTITUTE
Consolidated Management Office
Naperville, Illinois

CONTENTS

ABSTRACT 1

1 INTRODUCTION 1

1.1 Background 1
1.2 Purpose and Scope 2

2 OVERVIEW 3

2.1 Hardware Requirements 3
2.2 Program Structure and Summary 4
2.3 Notation and Nomenclature 4

3 PROGRAM INPUT 8

3.1 Variable Definitions 8
3.2 User Interface for Input 8

4 CONSTRUCTION-PERIOD ANALYSIS 14

4.1 Algorithms 14
4.2 Net Equity Option 19
4.3 User Interface for Reports 20

5 OPERATION-PERIOD ANALYSIS 20

5.1 Income Statement and Book Tax Calculation 21
5.2 Tax Depreciation Worksheet 24
5.3 Regulatory Tax Calculation 25
5.4 Sources and Uses of Funds 26
5.5 Retained Earnings Statement 29
5.6 Partnership Cashflow Analysis 29
5.7 Balance Sheet 29
5.8 After-Tax Cash Flows to Utility Investors 31
5.9 Busbar Cost Analysis 32
5.10 Limitations 33

6 REPORT WRITER 34

6.1 Report Descriptions 34
6.2 User Interface 35

REFERENCES 36

APPENDIX A: BREEDER Source Code 37

APPENDIX B: Cross Reference between Inputs or Outputs and Internal Variable Specifications 153

APPENDIX C: A Sample Interactive Session with BREEDER 159

CONTENTS (Cont'd)

APPENDIX D: Sample Runs 173

APPENDIX E: Documentation for the Net Equity Option 247

FIGURES

1 Flowchart of BREEDER 5

A.1 Suggested Overlay Structure for CP/M Microcomputers 37

TABLES

1 Summary of Routines Used in BREEDER 6

2 Definitions of Input Variables 9

3 Input and Output Variables used in Reports Other than the Income Statement 24

B.1 INPUT_ DATA 153

B.2 SUMS 155

B.3 OUTPUT_ DATA 156

BREEDER: A MICROCOMPUTER PROGRAM FOR FINANCIAL ANALYSIS OF A LARGE-SCALE PROTOTYPE BREEDER REACTOR

by

Robert F. Giese

ABSTRACT

This report describes a microcomputer-based, single-project financial analysis program: BREEDER. BREEDER is a "user-friendly" model designed to facilitate frequent and rapid analyses of the financial implications associated with alternative design and financing strategies for electric generating plants and large-scale prototype breeder (LSPB) reactors in particular. The model has proved to be a useful tool in establishing cost goals for LSPB reactors. The program is available on floppy disks for use on an IBM personal computer (or IBM "look-a-like") running under PC-DOS or a Kaypro II transportable computer running under CP/M (and many other CP/M machines). The report documents version 1.5 of BREEDER and contains a user's guide. The report also includes a general overview of BREEDER, a summary of hardware requirements, a definition of all required program inputs, a description of all algorithms used in performing the construction-period and operation-period analyses, and a summary of all available reports. The appendixes contain a complete source-code listing, a cross-reference table, a sample interactive session, several sample runs, and additional documentation of the net-equity program option.

1 INTRODUCTION

1.1. BACKGROUND

An essential ingredient of economic growth is the availability of adequate electricity at reasonable cost. Because of the productivity-enhancing advantage of electricity, which is making it the fuel of choice in a growing number of applications, the demand for electricity is likely to increase. As electric utilities seek to minimize both the cost of electricity to their customers and the financial risk to their stockholders, a variety of supply options will be required. The strategic advantages of a competitive balance between coal-fired and nuclear power have been recognized by the Electric Power Research Institute (EPRI) and a number of individual utility companies, whose stated policies are to avoid exclusive dependence on either energy form.[1]

In October, 1981, President Reagan announced a series of policy initiatives to revitalize the nuclear industry, including the development and demonstration of breeder reactor technology, which promises an almost unlimited future energy source. In response to this directive, the Department of Energy (DOE), EPRI, reactor manufacturers, nuclear plant architects and engineers, and the national laboratories developed a multiphase approach to development of a large-scale prototype breeder (LSPB) reactor.[2] It was envisioned that the electric utilities would play a key role in the design, construction, and operation of LSPB reactors. In response, the electric utilities, acting through EPRI, set up the Consolidated Management Office (CoMO) to provide coordination with DOE and leadership and direction in the development of the LSPB reactor.

One aspect of CoMO's effort is to evaluate alternative organizational and financial arrangements for the design, construction, and operation of near-commercial-scale, liquid-metal fast breeder reactor plants. In order to provide CoMO with an in-house capability for evaluation and comparison of alternative financial arrangements for the construction and operation of LSPB reactors, as well as the capability to reproduce several earlier financial analyses performed for CoMO,[3] Argonne National Laboratory has developed a single-project financial simulation model, BREEDER. BREEDER permits comparison of the financial performance of LSPB reactors with conventional coal-fired boilers and with light-water reactors on the basis of detailed annual financial statements and levelized busbar costs. The model has been used by CoMO as a tool in establishing economic goals for the LSPB program.[4]

1.2 PURPOSE AND SCOPE

The purpose of this report is to document the assumptions used in BREEDER and provide a basic guide to the use and operation of version 1.5 of BREEDER. A general overview of BREEDER together with hardware requirements for using BREEDER is presented in Sec. 2. Section 3 describes the program input required to run BREEDER, while Secs. 4 and 5 describe the algorithms used in performing the construction-period and operation-period analyses. Finally, a summary of the reports available is presented in Sec. 6.

Additional information is contained in several appendixes:

Appendix A - BREEDER source code
Appendix B - Cross-reference between inputs/outputs and internal variable specifications
Appendix C - Sample interactive BREEDER session
Appendix D - Sample runs
Appendix E - Documentation for the net equity option

2 OVERVIEW

BREEDER is a microcomputer-based, single-project financial analysis program designed to facilitate frequent and rapid analyses of the financial implications associated with alternative design and financing strategies for electric generating plants and LSPB reactors in particular. The model performs the following tasks:

- File management,
- Data input,
- Construction analysis,
- Cash flow analysis, and
- Report writing.

BREEDER incorporates several "user-friendly" features that facilitate operation and data entry, including:

- A menu-driven command structure,
- Input error-trapping routines,
- Direct cursor-control logic,
- Control C key to halt program execution,
- Escape key to terminate lengthy screen displays, and
- Automatic scroll hold for multiple screen outputs.

The remainder of this section discusses the hardware requirements and overall program structure, with a brief summary of each routine and supporting data bases.

2.1 HARDWARE REQUIREMENTS

BREEDER is available on floppy disks for two popular microcomputer systems, the Kaypro II and the IBM PC, using CP/M and PC-DOS operating systems, respectively.

Kaypro II. The Kaypro II is a self-contained, portable microcomputer incorporating a keyboard; a 9", 80-column by 24-line monochromatic screen; 64 kilobytes (K) of memory; and two 5-1/4", single-sided disk drives holding 191K each. In addition to the Kaypro II unit, an 80-column printer is needed if one wants to produce printouts of BREEDER results. For the Kaypro II, BREEDER resides on three diskettes, two containing the source code and another containing the executable code and supporting data

files. Using the program UNIFORM, standard with the Kaypro II system, these disks can be reformatted for use on many other popular micros running under CP/M, having 64K of memory, and attached to an 80-column by 24-line screen. Disks can also be created for 8" drives in a standard single-density format for use on CP/M machines having 8" drives.

IBM PC. The minimal configuration required to run BREEDER on an IBM PC is 128K of memory; one 5-1/4", single-sided disk drive holding 160K; and an 80-column by 25-line monochromatic monitor.* As with the Kaypro unit, an 80-column printer is needed if one wants to produce printouts of BREEDER results. BREEDER resides on three diskettes, two containing the source code and one containing the executable code and supporting data files.

2.2 PROGRAM STRUCTURE AND SUMMARY

In its current version, BREEDER consists of over 5000 statements, which translates to 90K of executable code and 11K of supporting data bases. In order to provide a code that is reliable, easy to read, and easy to maintain, BREEDER incorporates a top-down, structured programming style. The program consists of 16 routines written in PL/I and a set of direct-input, direct-output, screen-control routines (PLIDIO) written in assembler language.** This technique also facilitates the use of overlay structures, which permits BREEDER to operate on machines that have as little as 64K of memory. Further, each routine is divided into a number of subroutines corresponding to functional blocks.

Figure 1 illustrates the program flow for BREEDER, and Table 1 provides a brief description of each routine. A complete source listing for BREEDER is presented in Appendix A; Appendix B lists cross references between the internal variables used in the BREEDER source code and the variables contained in the external inputs and outputs -- INPUT_DATA, SUMS, and OUTPUT_DATA -- described in the following sections.

2.3 NOTATION AND NOMENCLATURE

The following notation and conventions are used throughout:

- Variable names correspond to labels appearing in input requests and output summaries.

- Input variables are printed in capital letters in equations and text.

---

*For text applications such as BREEDER a monochromatic monitor will provide higher resolution, although a color monitor can be used.

**The PL/I source code can be compiled using the Digital Research PL/I-80 (R1.3) compiler for CP/M machines and the Digital Research PL/I-86 (R1.0 DOS) compiler for the IBM PC.

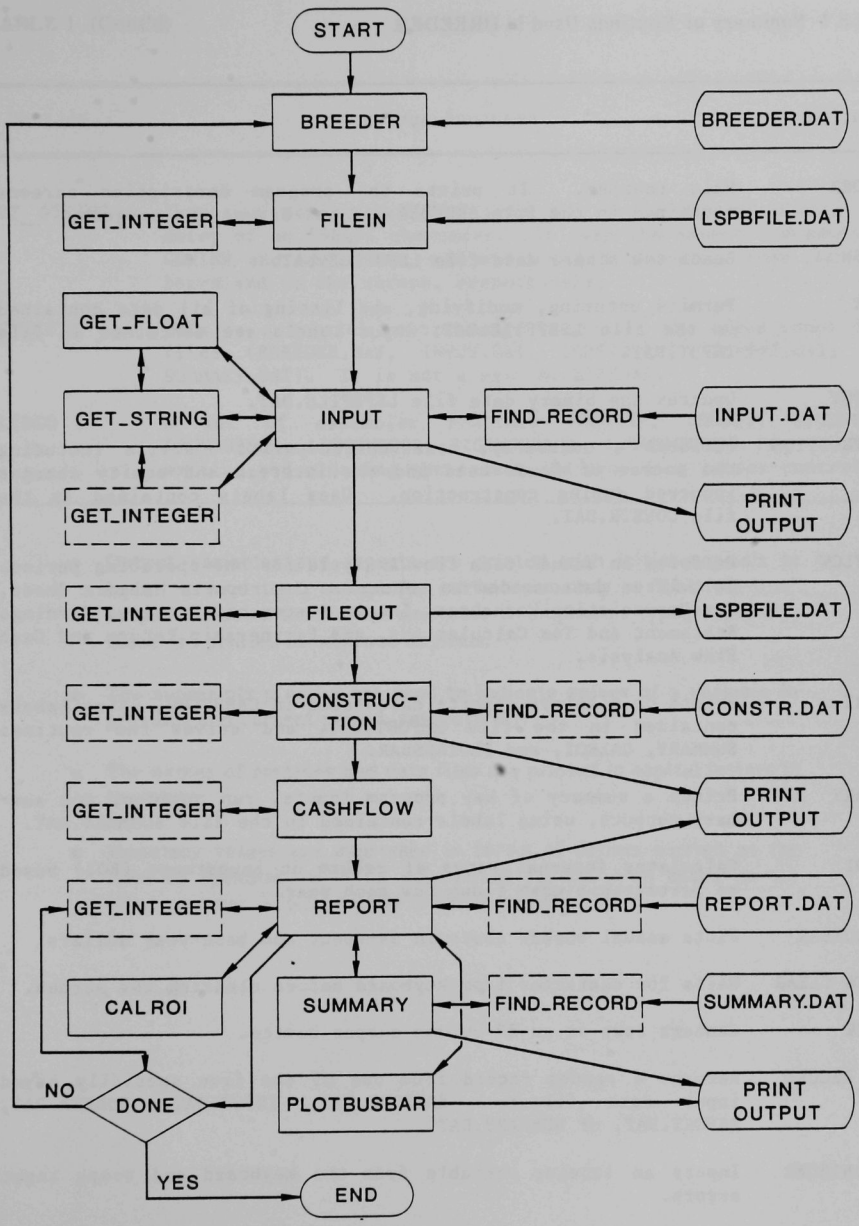


FIGURE 1 Flowchart of BREEDER

TABLE 1 Summary of Routines Used in BREEDER

Routine	Function
BREEDER	Main routine. It prints the program description screens contained in the file BREEDER.DAT and drives the program.
FILEIN	Reads the binary data file LSPBFILE.DAT.
INPUT	Permits entering, modifying, and listing of all data contained in the file LSBPFILE.DAT; input labels are contained in file INPUT.DAT.
FILEOUT	Updates the binary data file LSPBFILE.DAT.
CONSTRUCTION	Performs a quarterly, construction-period analysis including the source of funds used and the interest and equity charges incurred during construction. Uses labels contained in the file CONSTR.DAT.
CASHFLOW	Performs an annual cash flow analysis for the operating period. Calculates data needed to construct the reports Balance Sheet, Tax Depreciation Worksheet, Income Statement, Retained Earnings Statement and Tax Calculations, and Partnership Return and Cash Flow Analysis.
REPORT	Prints or displays reports calculated in CASHFLOW, using labels contained in the file REPORT.DAT, and drives the routines SUMMARY, CALROI, and PLOTBUSBAR.
SUMMARY	Prints a summary of key program inputs, run options, and summary outputs, using labels contained in the file SUMMARY.DAT.
CALROI	Calculates internal rates of return on investment (ROI) based on partnership cash flows for each year.
PLOTBUSBAR	Plots annual busbar costs in as-spent and base-year dollars.
SCREEN_CLEAR	Waits for character from keyboard before clearing the screen.
CENTER	Centers text on an 80-column output device.
FIND_RECORD	Returns a random record from one of the five specially keyed input data files: BREEDER.DAT, INPUT.DAT, CONSTR.DAT, REPORT.DAT, or SUMMARY.DAT.
GET_INTEGER	Inputs an integer variable from the keyboard and traps input errors.
GET_FLOAT	Inputs a float variable from the keyboard and traps input errors.

TABLE 1 (Cont'd)

Routine	Function
GET_STRING	Inputs a string variable from the keyboard, and checks for entry of an ESCAPE character. It uses the direct I/O routines CONINP and CONOUT to read and write a character from the keyboard and to the screen, respectively.
KEYWRITE	This is a special routine used to create the keyed input data files (BREEDER.DAT, INPUT.DAT, CONSTR.DAT, REPORT.DAT, and SUMMARY.DAT). It is not a part of BREEDER.
PLIDIO	A set of assembler routines (CONINP, CONOUT, UPCURSOR, DOWNCURSOR, LEFTCURSOR, RIGHTCURSOR, CLEARSCREEN) for reading, writing and clearing the screen and providing cursor control.

- Output variables and reports are printed with initial capitals in equations and text, except for certain summary or "botttom-line" output variables that it was desirable to emphasize typographically; these are printed in boldface capitals.

- The subscripts p and c are used to indicate values of a variable in previous and current period, respectively.

- The names of routines and data files are printed in capital letters in the text.

- Monetary values are expressed in terms of dollars current at the time of spending, with an adjustment for inflation based on the input variable GENERAL INFLATION (see Sec. 3).

3 PROGRAM INPUT

BREEDER requires the input of 397 variable values, including the multiple-value arrays required for several of the variables. This section defines each of the variables and describes the user interface for input of these variables.

3.1 VARIABLE DEFINITIONS

The input variables are grouped into eight categories (HEADER, YEARS, FACTORS, FLAGS, PLANT VARIABLES, RATES, PLANT ARRAYS, and CONSTRUCTION EXPENSES). Each variable in this section is listed in Table 2 exactly as it appears on the screen or in the hard copy summary, together with a definition and the allowed range of values. (Appendix B cross references these names to internal program variables contained in the structure INPUT_DATA.)

3.2 USER INTERFACE FOR INPUT

Three routines (FILEIN, INPUT, and FILEOUT; see Table 1) interface to the input variables. In each case a set of menu-driven command structures is used to elicit user responses. (See Appendix C for a sample interactive session.)

FILEIN is used either to read an existing binary data file (LSPBFILE.DAT) into the data structure INPUT_DATA or to create a new data file. In the case of the second option, all input variables are initially set to zero and the internal variable Analysis_Period is set to zero. This routine could easily be extended to include a library of input data files instead of the single file used in the current version.

INPUT is used to list and/or modify the input data variables. Two options for listing data are available: a soft-copy list on the screen or a hard-copy list on the printer. Titles, together with a variable indicating allowed range of each input variable, are contained in a specially keyed data file, INPUT.DAT. Variables can be modified individually, or a special option can be selected to modify all variables sequentially. This latter option is automatically selected if Analysis_Period has been set to 0, signifying creation of a new data file. A special routine is used for data modification; it traps input errors and checks that each input lies within the allowed variable range. Several options are available for entry of *array-type* variables (RATES, PLANT ARRAYS, and CONSTRUCTION EXPENSES). Modification of a single year, a range of years, or all years can be specified. If modification of more than a single year is requested, two data-entry modes are available: either the data can be entered by value for each year, or a first-year value and an annual escalation rate can be specified. In the latter case, values for subsequent years are automatically calculated by escalation of the first-year value.

FILEOUT provides an optional capability to update the binary data file LSPBFILE.DAT. As in the case of FILEIN, FILEOUT could easily be extended to operate on a library of input data files.

TABLE 2 Definitions of Input Variables

Variable	Definition	Range	Definition Number[a]
Header:	Program Title (up to 79 characters)	-	3-1
Years:			
Base Year	Starting year for rates arrays and reference year for dollar inputs in plant arrays and construction expenses	1982-1990	3-2
Construction Start	Starting year of construction period (1980.5 means start six months into 1980)	1982-2050	3-3
Construction End	End of construction period (must be an integral value: 1985 means construction ends at end of 1984)	1982-2065	3-4
Levelization Period	Number of years for busbar levelization	1-100	3-5
Debt Repayment Period	Number of years from start of plant operation to debt retirement	1-100	3-6
Factors (Decimal):			
Equity Fraction	Equity fraction of capitalization	0-1	3-7
Debt Fraction	Debt fraction of capitalization	0-1	3-8
Ad Valorem Tax	Revenue tax rate	0-1	3-9
Property Tax	Property tax rate	0-1	3-10
Income Tax Rate	Effective federal/state tax rate	0-1	3-11
ITC Base Rate	Investment tax credit rate	0-1	3-12

TABLE 2 (Cont'd)

Variable	Definition	Range	Definition Number[a]
Factors (Decimal): (cont'd)			
Target Capacity Factor	Nominal plant capacity factor; used in conjunction with trigger capacity factor for allocation of partner and DOE loans during operation period	0-1	3-13
Trigger Capacity Factor	Capacity factor below which cash flow deficits are covered by DOE loans	0-1	3-14
Target Return on Equity	Desired return on equity rate; used for determining cost of capital for levelizing busbar costs	0-1	3-15
Backfit-Replacement Rate	Backfit and replacement rate, as fraction of book starting asset base	0-1	3-16
Flags (0 or 1):		0 or 1	
95%(0)/Full(1) Dep. Flag	0: use 95% of full tax depreciation base 1: use full tax depreciation base	-	3-17
StLn(0)/ACRS(1) Dep. Flag	0: use straight-line tax depreciation 1: use ACRS (TEFRA)[b] tax depreciation	-	3-18
Flowthru(0)/Norm.(1) Flag	0: use flowthrough tax accounting 1: use normalized tax accounting	-	3-19
Uncons(0)/Cons(1)Tax Flag	0: single-project tax treatment 1: consolidated tax treatment	-	3-20
Quarter(0)/Annual(1) Flag	0: draw down short-term debt, common and preferred equity quarterly 1: capitalize short-term debt annually	-	3-21

TABLE 2 (Cont'd)

Variable	Definition	Range	Definition Number[a]
Flags (0 or 1): (cont'd)			
Compound(0)/Simple(1)Flag	0: use compound quarterly interest rate $[(1 + r)^{1/4} - 1]$ 1: use simple quarterly interest rate (r/4)	-	3-22
Gross(0)/Net(1)PropTxFlag	0: use initial book asset base for property tax 1: use net book asset base for property tax	-	3-23
Pre(0)/After(1)TxDiscFlag	0: use pretax cost-of-capital discount rate 1: use after-tax cost-of-capital discount rate	-	3-24
Net(0)/Fix(1) Equity Flag	0: calculate debt/equity ratios for zero net equity requirement 1: use input debt/equity ratios	-	3-25
Bot.Up(0)/Top.Dn.(1) Flag	0: perform bottom-up analysis [specify required return on equity (ROE) and calculate revenue requirements] 1: perform top-down analysis (specify busbar costs and calculate ROE)	-	3-26
Plant Variables:			
Size (MW)	Plant size in MW_e	0-2,000	3-27
Book Life (yr)	Plant book life	1-100	3-28
Tax Life (yr)	Plant tax life	1-100	3-29
Startup (million $)	Startup costs, $\$10^6$, in base-year dollars (assumed to be included in the last year's construction expenses)	0-2,000	3-30

TABLE 2 (Cont'd)

Variable	Definition	Range	Definition Number[a]
Plant Variables: (cont'd)			
Init.Work.Cap. (million $)	Initial working capital, $\$10^6$, in base-year dollars (assumed to be included in the last year's construction expenses)	0-2,000	3-31
Heat Rate (Btu/kWh)	Plant heat rate in Btu/kWh	8,000-1,2000	3-32
Rates (Decimal):			
General Inflation	30 yearly inflation values, starting with base year	0-1	3-33
Short-Term Debt	30 yearly interest rates on short-term debt, starting with base year	0-1	3-34
Long-Term Debt	30 yearly interest rates on long-term debt, starting with base year	0-1	3-35
Preferred Stock	30 yearly rates of return on preferred stock rate, starting with base year	0-1	3-36
Equity	50 yearly rates of return on common equity, starting with base year	0-1	3-37
Plant Arrays:			
Capacity Factor (Decimal)	30 yearly plant capacity factors, starting in first year of plant operation	0-1	3-38
Fuel Costs ($/MMBtu)	30 yearly fuel costs, $\$/10^6$ Btu, in mid-base-year dollars, starting in first year of plant operation	0-100	3-39

TABLE 2 (Cont'd)

Variable	Definition	Range	Definition Number[a]
Plant Arrays: (cont'd)			
O&M (mills/kWh)	30 yearly operation and maintenance costs, mills/kWh, in mid-base-year dollars, starting in first year of plant operation	0-100	3-40
Working Cap. (million $)	30 yearly working capital values, $\$10^6$, in mid-base-year dollars, starting in first year of plant operations	0-2,000	3-41
Federal Sub. (million $)	30 yearly federal subsidies (tax-free), $\$10^6$, current-year dollars, starting in first year of plant operation	0-2,000	3-42
Busbar Costs (cents/kWh)	30 yearly busbar costs in current-year dollars (used only in top-down analysis)	0-100	3-43
Construction Expenses (million $/year):			
Plant	Annual plant overnight[c] construction cost, $\$10^6$/yr, in base-year dollars starting in first year of plant construction (15 years maximum)	0-2,000	3-44

[a]See Appendix B for the internal variables corresponding to numbered definitions.

[b]ACRS = Accelerated capital recovery system, described in the Tax Equity and Fiscal Responsibility Act of 1982 (TEFRA).

[c]"Overnight" cost is the estimated cost with no allowance for future inflation, as if the plant could literally be built overnight.

4 CONSTRUCTION-PERIOD ANALYSIS

The construction-period analysis is performed by the routine CONSTRUCTION. This section describes the algorithms used, the effects of the various option flags (see Table 2), and the user interface to the report writer.

4.1 ALGORITHMS

The algorithms are grouped into five functional blocks (subroutines): CONSTRUCTION-EXPENSES, INTEREST, FINANCE, TAXES, AND FINALSUMS. The algorithms for each block are presented first in narrative and then in equation form below.

CONSTRUCTION-EXPENSES calculates the plant construction costs, which are prorated quarterly. Total plant construction costs include two components: overnight cost, which is the cost in base-year dollars, excluding inflation; and a component due to inflation over the period from the base year to the year of construction. Quarterly and cumulative (Cum.) values are calculated as:

$$\text{Total Construction} = \text{PLANT}/4 + [\text{PLANT}/4 \times (\text{Cum. Inflation}^* - 1)] \tag{4-1}$$

$$\text{Construction Book}_c = \text{Construction Book}_p + \text{Total Construction} \tag{4-2}$$

$$\text{Construction Base Year}_c = \text{Construction Base Year}_p + \text{PLANT} \tag{4-3}$$

INTEREST calculates financing charges on the basis of previous period balances. Interest charges on long-term debt are calculated and included in the tax accrual calculation at the end of each annual period, but are reported in the first quarter of the subsequent period (except for the last year of construction, which is discussed under FINALSUMS). Short-term debt, common equity, and preferred stock charges are calculated quarterly.

Two options are available for calculating quarterly interest rates:

- COMPOUND(0)/SIMPLE(1)FLAG = 0 (compound rate):

$$\text{Quarterly Rate} = (\text{annual rate} + 1)^{1/4} - 1 \tag{4-4}$$

- COMPOUND(0)/SIMPLE(1)FLAG = 1 (simple rate):

$$\text{Quarterly Rate} = \text{annual rate}/4 \tag{4-5}$$

---

*The cumulative inflation includes all inflation factors, 1 + GENERAL INFLATION (year), from the base to the end of the current quarter. Within a given year, inflation is assumed to grow geometrically, yielding a quarterly inflation factor of $(1 + \text{GENERAL INFLATION})^{1/4}$.

$$\text{Financing Charges} = \text{Cum. Short-Term Debt} \times \text{Short-Term Interest Rate} + \text{Common Equity} \times \text{Quarterly Equity} + \text{Preferred Stock Equity} \times \text{Preferred Rate} + \text{Total LT Debt} \times \text{LT Debt Rate} \times \text{Fact} \quad (4\text{-}6)$$

where the variables defined below are all as of the beginning of the previous period, and

Cum. Short-Term Debt = Cumulative short-term debt

Short-Term Interest Rate = Quarterly short-term interest rate

Common Equity = Cumulative common equity

Quarterly Equity = Quarterly equity charge rate

Preferred Stock Equity = Cumulative preferred stock

Preferred Rate = Imbedded preferred stock dividend rate

Total LT Debt = Cumulative long-term debt

LT Debt Rate = Imbedded annual long-term debt interest rate

Fact = 1 for the first quarter of each year;
0 for the last three quarters of each year

Also, cumulative allowance for funds used during construction (AFUDC) is calculated in as-spent (AFUDC Book) and base-year (AFUDC Base Year) dollars.

$$\text{AFUDC Book}_c = \text{AFUDC Book}_p + \text{Financing Charges}_c \quad (4\text{-}7)$$

$$\text{AFUDC Base Year}_c = (\text{AFUDC Base Year}_p + \text{Financing Charges}_c) \div \text{Cum. Inflation} \quad (4\text{-}8)$$

AFUDC Base Year expresses AFUDC book in base-year dollars, but does not factor out the inflation effects of using nominal interest rates to calculate financing charges.

FINANCE allocates total construction costs plus financing charges at the *beginning* of each quarter among short-term and long-term debt and common and preferred equity. Short-term debt is capitalized either quarterly or annually:

- QUARTER(0)/ANNUAL(1) FLAG = 0 (capitalize quarterly):

$$\text{Short-Term Debt} = \text{DEBT FRACTION} \times (\text{Total Construction} + \text{Financing Charges}) \quad (4\text{-}9)$$

$$\text{Common Equity} = \text{EQUITY FRACTION} \times (\text{Total Construction} + \text{Financing Charges}) \quad (4\text{-}10)$$

$$\text{Preferred Stock} = (1\text{-DEBT FRACTION} - \text{EQUITY FRACTION}) \times (\text{Total Construction} + \text{Financing Charges}) \quad (4\text{-}11)$$

- QUARTER(0)/ANNUAL(1) FLAG = 1 (capitalize annually):

$$\text{Short-Term Debt} = \text{Total Construction} + \text{Financing Charges} \tag{4-12}$$

$$\text{Common Equity} = 0 \tag{4-13}$$

$$\text{Preferred Stock} = 0 \tag{4-14}$$

At the end of each year all outstanding short-term debt is capitalized and included in the annual cash flow analysis, but is reported in the subsequent quarter. Two options are available:

- QUARTER(0)/ANNUAL(1) FLAG = 0 (capitalize quarterly):

$$\text{Long-Term Debt} = \text{Short-Term Debt} \tag{4-15}$$

- QUARTER(0)/ANNUAL(1) FLAG = 1 (capitalize annually):

$$\text{Long-Term Debt} = \text{Short-Term Debt} \times \text{DEBT FRACTION} \tag{4-16}$$

$$\text{Common Equity} = \text{Short-Term Debt} \times \text{EQUITY FRACTION} \tag{4-17}$$

$$\text{Preferred Stock} = \text{Short-Term Debt} \times (1 - \text{DEBT FRACTION} - \text{EQUITY FRACTION}) \tag{4-18}$$

At the end of each period, the long-term debt and preferred rates are updated. These rates represent imbedded financing rates. Cumulative long-term debt (i.e., the calculated variable Total LT Debt), preferred stock equity, and common equity are also calculated.

$$(\text{LT Debt Rate})_c = [(\text{LT Debt Rate}_p \times \text{Total LT Debt}_p) + (\text{Long-Term Debt}_c \times \text{LONG-TERM DEBT}_c)] \div (\text{Total LT Debt}_p + \text{Long-Term Debt}_c) \tag{4-19}$$

where

LT Debt Rate is the imbedded interest rate for long-term debt,

Total LT Debt is the cumulative long-term debt issued,

Long-Term Debt is the long-term debt issued in this period, and

LONG-TERM DEBT is the long-term debt interest rate for this period.

$$\text{Total LT Debt}_c = \text{Total LT Debt}_p + \text{Long-Term Debt}_c \tag{4-20}$$

$$\text{Preferred Rate}_c = [(\text{Preferred Rate}_p \times \text{Preferred Stock Equity}_p) + (\text{Preferred Stock}_c \times \text{PREFERRED STOCK}_c)] \div (\text{Preferred Stock Equity}_p + \text{Preferred Stock}_c) \tag{4-21}$$

where

Preferred Rate is the imbedded dividend rate for preferred stock,

Preferred Stock Equity is the cumulative preferred stock issued,

Preferred Stock is the preferred stock issued in this period, and

PREFERRED STOCK is the preferred stock dividend rate for this period.

$$\text{Preferred Stock Equity}_c = \text{Preferred Stock Equity}_p + \text{Preferred Stock}_c \tag{4-22}$$

$$\text{End Common Equity}_c = \text{End Common Equity}_p + \text{Common Equity}_c \tag{4-23}$$

where

End Common Equity is the cumulative common stock issued, and

Common Equity is the common stock issued for this period.

TAXES calculates investment tax credits (ITC) and deductions for interest during construction (IDC) annually. For purposes of this analysis, the fiscal year is assumed to correspond to the calendar year. Since interest charges are reported in the quarter *after* they are accrued, deductions for interest during construction are based upon quarters 2, 3, and 4 of the current tax year plus the first quarter of the subsequent tax year.*

$$\begin{aligned}\text{Interest Credit} = \sum_{p=c-4}^{c-1} & [(\text{Cum. Short-Term Debt}_p \\ & \times \text{Short-Term Interest Rate}_p) \\ & + (\text{Total LT Debt}_p \times \text{LT Debt Rate}_p \\ & \times \text{Fact})] \times \text{INCOME TAX RATE}\end{aligned} \tag{4-24}$$

where c indicates the current period (i.e., the summation runs over the previous four quarters) and

Cum. Short-Term Debt = Cumulative short-term debt

Short-Term Interest Rate = Quarterly short-term interest rate

Total LT Debt = Cumulative long-term debt

LT Debt Rate = Imbedded annual long-term debt interest rate

Fact = 1 for first quarter of each year; 0 for the last three quarters of each year.

$$\text{Tax Accrual}_c = \text{Tax Accrual}_p + \text{Interest Credit}_c \tag{4-25}$$

---

*TEFRA amended Section 189 of the Internal Revenue Code to require corporations, including regulated utilities, to capitalize interest and taxes on "real property" during its construction period. The definition of real property has yet to be litigated, and many utilities currently are not treating power plants as real property. BREEDER reflects this practice.

Investment tax credits (ITC) are calculated in accordance with the Tax Equity and Fiscal Responsibility Act of 1982 (TEFRA). Two ITC treatments are permitted under TEFRA. BREEDER permits the use of either:

- 95%(0)/FULL(1) DEP. FLAG = 0

  ITC Rate = ITC BASE RATE × Total Construction (4-26)

  (ITC BASE RATE = 10% under TEFRA)

- 95%(0)/FULL(1) DEP. FLAG = 1

  ITC Rate = (ITC BASE RATE-2%) × Total Construction (4-27)

If the 10% ITC rate is selected under TEFRA, then the tax depreciation base must be reduced to 95% of its nominal value (see Sec. 5). If the 8% option is elected, the full tax base may be used for tax depreciation. In either case, before-tax and after-tax cash flows are calculated as:

$$\text{Before Tax} = -\ \text{Common Equity}_c \tag{4-28}$$

$$\text{After Tax} = \text{Before Tax} + \text{Interest Credit} \tag{4-29}$$

The last year's construction costs include two additional expenses that are treated differently for tax purposes:

- STARTUP costs: One-time costs associated with plant startup

- INIT.WORK.CAP.: Initial working capital reserve.

STARTUP cost and INT.WORK.CAP. are input, although they are plant variables and are assumed to be included as part of the final year's construction costs (PLANT). Like PLANT, STARTUP and INIT.WORK.CAP. are prorated quarterly over the last construction period. The input INIT.WORK.CAP. is not recognized as part of the plant book value. Neither STARTUP nor INIT.WORK.CAP. is eligible for ITC, and neither is included in the tax depreciation base. However, TEFRA does permit straight-line write-off of startup costs over five years.

$$\text{Construction Book}_c = \text{Construction Book}_p - \text{INIT.WORK.CAP.} \tag{4-30}$$

$$\text{Total ITC}_c = (\text{Construction Book}_c - \text{STARTUP}) \times \text{ITC Rate} + \text{Total ITC}_p \tag{4-31}$$

FINALSUMS calculates expenses accrued during the final year of construction, which would be reported during the first quarter of the next year. All outstanding short-term debt and accrued financing charges are converted into long-term debt, preferred stock, and common equity, and the variables LT Debt Rate, Preferred Rate, Total LT Debt, Preferred Stock Equity, and End Common Equity are updated.

4.2 NET EQUITY OPTION

If this option is selected, by setting the NET(0)/FIX(1) EQUITY FLAG to 0, the program calculates equity drawdowns equal to the accrued tax benefits for each quarter. This option overrides the QUARTER(0)/ANNUAL(1) FLAG option, and the DEBT FRACTION and EQUITY FRACTION inputs are ignored.

Each quarter's financing consists of short-term debt and common equity only. No preferred stock is allowed, and the cumulative short-term debt is converted to long-term debt at the end of each year. The following two equations summarize the financing and tax-accrual relationships that are used to determine the amount of short-term debt and common equity required for each quarter.

$$\text{Total Construction} + \text{Financing Charges} = \text{Short-Term Debt} + \text{Common Equity} \tag{4-32}$$

$$\begin{aligned}\text{Common Equity} = \{&[(\text{Cum. Short-Term Debt} + \text{Short-Term Debt}) \\ &\times \text{Short-Term Interest Rate}] + (\text{Total LT Debt} \\ &\times \text{LT Debt Rate} \times \text{Fact})\} \times \text{INCOME TAX RATE} \\ &+ (\text{Total Construction} \times \text{ITC Rate})\end{aligned} \tag{4-33}$$

where

$$\text{Cum. Short-Term Debt} = \text{Cumulative short-term debt as of the beginning of quarter (after conversion to long-term debt, if required)} \tag{4-34}$$

$$\text{Short-Term Interest Rate} = \text{Quarterly short-term interest rate for quarter} \tag{4-35}$$

$$\text{Fact} = 0 \text{ in the first three quarters of each year; } 1 \text{ in the 4th quarter of each year} \tag{4-36}$$

These relationships are used to solve for the required financing:

$$\begin{aligned}\text{Short-Term Debt} = \{&[\text{Total Construction} \times (1 - \text{ITC Rate})] \\ &+ \text{Financing Charges} - [(\text{Cum. Short-Term Debt} \\ &\times \text{Short-Term Interest Rate}) \\ &+ (\text{Total LT Debt} \times \text{LT Debt Rate} \times \text{Fact})] \\ &\times \text{INCOME TAX RATE}\}/[(1 + \text{Short-Term} \\ &\text{Interest Rate}) \times \text{INCOME TAX RATE}]\end{aligned} \tag{4-37}$$

$$\text{Common Equity} = \text{Total Construction} + \text{Financing Charges} - \text{Short-Term Debt} \tag{4-38}$$

$$\text{Preferred Stock} = 0 \tag{4-39}$$

$$\text{Before Tax} = -\text{Common Equity} \tag{4-40}$$

$$\text{After Tax} = 0 \tag{4-41}$$

The final year accruals are financed entirely through long-term debt. A more complete discussion of this option is presented in Appendix E.

4.3 USER INTERFACE FOR REPORTS

Upon completion of the construction-period analysis, the user has the option of obtaining summary reports on either a quarterly or an annual basis or both. These reports are displayed on the screen and can be printed if a printer is available. The report header labels are contained in the specially keyed input file CONSTR.DAT; examples of these reports can be found in Appendixes C and D.

5 OPERATION-PERIOD ANALYSIS

This section discusses the algorithms and assumptions used to perform the cash flow analysis (routine CASHFLOW) during the plant operation phase. The variables for the following reports are calculated:

- Balance Sheet
- Income Statement
- Sources and Uses of Funds Statement
- Retained Earnings Statement
- Tax Depreciation Worksheet
- Book Tax Calculation
- Regulatory Tax Calculation
- Partnership Return and Cashflow Analysis
- After-Tax Cash Flows to Utility Investors
- Busbar Cost Analysis

The user is first requested to specify the number of years to be analyzed, not to exceed the value of the plant variable BOOK LIFE or 30 years, whichever is less. The analysis then proceeds by first calculating the *project* reports, starting with the Income Statement. Either a top-down or bottom-up analysis is allowed. The top-down analysis proceeds by using the input variable BUSBAR COSTS to calculate electric sales, and continues on to common earnings. The bottom-up analysis starts with the input variable EQUITY to calculate common earnings and works up through the Income Statement to calculate required electric sales. In either case, Pre-Tax Book Income is used to calculate federal income tax (FIT) on a book basis (see FIT from the Book Tax Calculation, Sec. 5.1).

Next the Regulatory Tax Calculation, using results from the Tax Depreciation Worksheet, is performed to calculate FIT Due, FIT Taken, and ITC Taken. FIT Taken and ITC Taken are then used to calculate Deferred ITC and Deferred FIT for the Source and Uses of Funds Statement. Partner loans or government loans are used to supplement sources of funds as required to ensure sufficient funds to cover the amounts specified for Preferred Stock Due (straightline retirement), Preferred Dividends, Working Capital Changes, and LTD Repayment (straightline retirement). Excess funds are used to retire outstanding partner loans first, and then the balance is disbursed as common dividends.

The output variables Common Dividends and Common Earnings are then used to calculate End Common Equity for the Retained Earnings Statement. Finally, values for the Balance Sheet report are calculated.

In addition to the above *project* reports, a special consolidated *partnership* analysis is performed. This analysis includes calculation of actual taxes paid (FIT Due) as well as an after-tax cash flow analysis used to calculate return on investment (ROI).

To generate the operational phase reports, a number of initial values must be established using results of the Construction Analysis (described in Sec. 4), as follows:

Repay Debt = Total LT Debt/DEBT REPAYMENT PERIOD (5-1)

Cum. Deferred FIT = 0 (5-2)

Deferred Reserve = 0 (5-3)

Partner Loans Outst. = 0 (5-4)

ITC Remaining = Total ITC (5-5)

Cum. Deferred ITC = Total ITC (5-6)

Cum. Working Capital = INIT.WORK.CAP. (5-7)

Prop. Tax Base = Construction Book_c + AFUDC Book_c (5-8)

Cum. to Partners = Total ITC + Tax Accrual - End Common Equity (5-9)

$$\text{Cum. Inflation} = \prod_{i=B}^{i=C} (1 + G)_i \qquad (5\text{-}10)$$

where

B = BASE YEAR

C = CONSTRUCTION END

G = GENERAL INFLATION, the array of input values, and

i = an index of successive years.

Total Plant Cost = Construction Book_c + AFUDC Book_c (5-11)

5.1 INCOME STATEMENT AND BOOK TAX CALCULATION

This section describes the calculation of the Income Statement and Book Tax Calculation. Two modes of analysis are permitted:

- Bottom-up: BOT.UP(0)/TOP.DN.(1) FLAG = 0. Required return on equity (EQUITY) is used to determine revenue requirements

- Top-down: BOT.UP(0)/TOP.DN.(1) FLAG = 1. BUSBAR COST is used to determine Electric Sales and resulting Common Earnings and Common Equity Rate

Before either of these modes can be applied, however, several intermediate values (e.g., depreciation, inflation, and fuel and operating expenses) must be calculated, as indicated in the following equations:

$$\text{Plant Book Dep.} = \text{Construction Book/BOOK LIFE} \tag{5-12}$$

$$\text{AFUDC Book Dep.} = \text{AFUDC Book/BOOK LIFE} \tag{5-13}$$

$$\text{Total Book Dep.} = \text{Plant Book Dep.} + \text{AFUDC Book Dep.} \tag{5-14}$$

$$\text{Amort ITC} = \text{Total ITC/BOOK LIFE} \tag{5-15}$$

$$\text{Mid-Inflation*} = \text{Cum. Inflation}_p \times [\text{GENERAL INFLATION}_{ic} \div \ln(1 + \text{GENERAL INFLATION}_{ic})] \tag{5-16}$$

$$\text{Cum. Inflation}_c = \text{Cum. Inflation}_p \times (1 + \text{GENERAL INFLATION}_{ic}) \tag{5-17}$$

$$\text{Federal Subsidy} = \text{FEDERAL SUB.} \tag{5-18}$$

$$\text{Fuel Expense} = \text{FUEL COST} \times 8.76 \times 10^{-6} \times \text{SIZE} \times \text{HEAT RATE} \times \text{Mid-Inflation} \tag{5-19}$$

$$\text{O\&M Expense} = \text{O\&M} \times 8.76 \times 10^{-3} \times \text{SIZE} \times \text{Mid-Inflation} \tag{5-20}$$

$$\text{Backfit \& Replacement} = \text{Total Plant Cost} \times \text{BACKFIT-REPLACEMENT RATE} \tag{5-21}$$

$$\text{Property Tax} = \text{Prop Tax Base}_p \times \text{PROPERTY TAX} \tag{5-22}$$

$$\text{LTD Interest} = \text{Total LT Debt}_p \times \text{LT Debt Rate} \tag{5-23}$$

$$\text{Pref. Dividends} = \text{Preferred Stock Equity}_p \times \text{Preferred Rate} \tag{5-24}$$

$$\text{Begin. Common Equity}_c{}^{+} = \textbf{END COMMON EQUITY}_p \tag{5-25}$$

The bottom-up analysis proceeds as shown below:

$$\text{Common Stock Earnings} = \text{Begin. Common Equity} \times \text{EQUITY} \tag{5-26}$$

$$\text{Pre-Tax Book Income**} = (\text{Common Stock Earnings} + \text{Preferred Dividends} - \text{Federal Subsidy}_c) \div (1 - \text{INCOME TAX RATE}) \tag{5-27}$$

---

*This factor results from the assumption that certain quantities such as O&M expenses are spent continuously throughout year i. This factor is approximately equal to 1 + GENERAL INFLATION$_i$/2.

+Calculated in Retained Earnings Statement.

**Calculated in the Book Tax Calculation.

Or the top-down analysis proceeds as below:

Electric Sales = BUSBAR COST × CAPACITY FACTOR × 8.76 × 10^6 × SIZE (5-28)

Ad Valorem Tax = Electric Sales × AD VALOREM TAX (5-29)

Total Oper. Expense = Fuel Expense + O&M Expense + Backfit & Replacement + Ad Valorem Tax (5-30)

Pre-Tax Book Income** = Electric Sales - Total Oper. Expense - Property Tax - Total Book Dep. - LTD Interest + Amort. ITC (5-31)

The next three equations are calculated for either mode.

Book FIT* = Pre-tax Book Income × INCOME TAX RATE (5-32)

Net FIT (Book Basis) = Book FIT - Amort. ITC (5-33)

Total Pre-Debt Expense = Total Oper. Expense + Total Book Dep. + Property Tax + Net FIT (Book Basis) (5-34)

The bottom-up calculation continues:

Electric Sales = (Total Pre-Debt Expense + LTD Interest + Common Stock Earnings + Pref. Dividends - Federal Subsidy)/(1 - AD VALOREM TAX) (5-35)

Ad Valorem Tax = Electric Sales × AD VALOREM TAX (5-36)

Total Oper. Expense = Fuel Expense + O&M Expense + Backfit & Replacement + Ad Valorem Tax (5-37)

BUSBAR COST = Electric Sales/(8.76 × 10^{-3} × SIZE × CAPACITY FACTOR) (5-38)

The next three equations are used for either mode:

TOTAL REVENUES = Electric Sales + Federal Subsidy (5-39)

Pre-Debt Net Income = **TOTAL REVENUES** - Total Pre-Debt Expense (5-40)

NET INCOME = Pre-Debt Net Income - LTD Interest (5-41)

Finally, common earnings are calculated for the top-down mode only.

Common Stock Earnings = **NET INCOME** - Pref. Dividends (5-42)

Table 3 shows the variables defined for the Income Statement that are subsequently used in reports other than the Income Statement.

---

*Calculated in the Book Tax Calculation.

TABLE 3 Input and Output Variables used in Reports Other than the Income Statement

Variable	Report
Federal Subsidy	Partnership Cash Flow Analysis
Total Oper. Expense	Book Tax Calculation, Regulatory Tax Calculation, Partnership Return
Total Book Dep.	Sources of Funds, Book Tax Calculation
Property Tax	Book Tax Calculation, Regulatory Tax Calculation, Partnership Return
LTD Interest	Book Tax Calculation, Regulatory Tax Calculation, Partnership Return
Preferred Dividends	Uses of Funds, Partnership Cash Flow Analysis
Common Stock Earnings	Retained Earnings Statement
Electric Sales	Book Tax Calculation, Regulatory Tax Calculation, Partnership Return
Net FIT (Book Basis)	Book Tax Calculation
Net Income	Sources of Funds

5.2 TAX DEPRECIATION WORKSHEET

The Tax Depreciation Worksheet calculates allowable tax depreciations for each year. Two schemes are permitted:

- Straight-line depreciation over TAX LIFE years; and
- Depreciation as specified by the accelerated capital recovery system (ACRS) described in TEFRA.

Under ACRS the allowable tax lives have been reduced from previous values. The allowable tax life for coal-fired generating plants is 15 years, whereas the allowable tax life for nuclear plants is 10 years. Only tax lives of 5, 10, or 15 years are permitted under this option. The annual depreciation factors defined by TEFRA are used for each category of property.

Under either system, STARTUP costs are depreciated over five years (straight-line). The tax base is calculated in one of two ways:

- 95%(0)/FULL(1) FLAG = 0

  Tax Base = 0.95 × (Construction Book - STARTUP)* (5-43)

- 95%(0)/FULL(1) FLAG = 1

  Tax Base = (Construction Book - STARTUP)* (5-44)

Startup Dep. = (0.20, 0.20, 0.20, 0.20, 0.20) × STARTUP (5-45)

5-Year Prop. Dep. = (0.15, 0.22, 0.21, 0.21, 0.21) × Tax Base (5-46)

10-Year Prop. Dep. = (0.08, 0.14, 0.12, 0.10, 0.10,
0.10, 0.09, 0.09, 0.09, 0.09) × Tax Base (5-47)

15-Year Prop. Dep. = (0.05, 0.10, 0.09, 0.08, 0.07,
0.07, 0.06, 0.06, 0.06, 0.06,
0.06, 0.06, 0.06, 0.06, 0.06) × Tax Base (5-48)

Straight-Line Dep. = Tax Base/TAX LIFE (for TAX LIFE years) (5-49)

TOTAL TAX DEPRECIATION = Startup Dep. + 5-Year Prop. Dep.
+ 10-Year Prop. Dep. + 15-Year
Prop. Dep. + Straight-Line Dep. (5-50)

The **TOTAL TAX DEPRECIATION** is used in the following reports:

- Regulatory Tax Calculation
- Partnership Return
- Partnership Cashflow Analysis

5.3 REGULATORY TAX CALCULATION

This section determines the net regulated federal income tax. Two tax treatments are allowed:

- Unconsolidated: UNCONSL(0)/CONS(1)TAX FLAG = 0 (Negative FIT is not allowed)

---

*Neither startup costs nor initial working capital reserve are included in the tax base; the initial working capital has already been excluded from Construction Book, in Eq. 4-30.

- Consolidated: UNCONSL(0)/CONS(1)TAX FLAG = 1 (Negative FIT is allowed at the project level and is assumed to offset other FIT due)

In the first case, deferred FIT and ITC accounts are maintained, as Cum. Deferred FIT and ITC Remaining, respectively. If FIT Due is negative, its absolute value is subtracted from the Cum. Deferred FIT account (See Balance Sheet, Sec. 5.7). Otherwise, first the Cum. Deferred FIT account and then the ITC Remaining account is used to reduce FIT Due to zero, but not to a negative amount.

In the second case, FIT and ITC credits are used as soon as they become available. In either case, the first step is to calculate pre-tax partnership income and FIT due.

Pre-Tax Part. Income = Electric Sales - Total Oper. Expense
- **TOTAL TAX DEPRECIATION** - Property Tax
- LTD Interest (5-51)

FIT Due = Pre-Tax Part. Income × INCOME TAX RATE (5-52)

- Unconsolidated:

FIT Taken = min (FIT Due, Cum. Deferred FIT) (5-53)

ITC Taken = min [(FIT Due - FIT Taken), ITC Remaining] (5-54)

$\text{ITC Remaining}_c = \text{ITC Remaining}_p - \text{ITC Taken}$ (5-55)

- Consolidated:

FIT Taken = 0 (5-56)

ITC Taken = ITC Remaining (5-57)

ITC Remaining = 0 (5-58)

In either case **NET REGULATED FIT** is given by:

NET REGULATED FIT = FIT Due - FIT Taken - ITC Taken (5-59)

Once Pre-Tax Part. Income and FIT Due have been calculated, as shown above, they are used in the report Partnership Return.

5.4 SOURCES AND USES OF FUNDS

In most cases, tax benefits due to timing differences are required to be normalized. These deferred taxes are treated as an additional source of funds. However, BREEDER also allows tax benefits to flow through to equity holders. (This option was included to reproduce a previous analysis[3] of a "first-of-a-kind" prototye breeder reactor for which government loans were assumed to supply 75% of the financing requirements.

The analysis did not allow for equity funds used during construction, and a 25-year, straight-line depreciation was elected under TEFRA to reduce negative cash flow late in the project. To offset the lack of allowance for equity funds used during construction, tax benefits due to depreciation were allowed to flow through to the equity holders. In contrast, many state utility commissions have adopted a policy of letting tax benefits flow through to the consumer instead of the equity holder. Neither of these flow-through treatments is allowed, however, if accelerated capital recovery is elected under TEFRA.)

- Normalized: FLOWTHRU(0)/NORM(1) FLAG = 1 (deferred taxes are treated as a source of funds)

- Flowthrough: FLOWTHRU(0)/Norm.(1) FLAG = 0 (deferred taxes are allowed to flow through to the equity holders)

In either case the need for partner and/or government loans, and the amount of common dividends to be paid, are determined. Either partner or government loans may be required, to prevent the variable Common Dividends from taking negative values. Partner loans are assumed to be interest-free loans from the partners. These loans are repaid before issuance of common dividends. Government loans are additional long-term debt issues that are due at the end of the DEBT REPAYMENT PERIOD. These loans incur the prevailing interest rate.

Deferred ITC = ITC Taken - Amort. ITC (5-60)

- Normalized:

X = Max [INCOME TAX RATE × (**TOTAL TAX DEPRECIATION** - Total Book Dep.), - Deferred Reserve] (5-61)

- Flowthrough:

X = 0 (5-62)

Deferred Reserve_c = Deferred Reserve_p + X (5-63)

Deferred FIT = X - FIT Taken (5-64)

TOTAL INTERNAL = **NET INCOME** + Total Book Dep. + Deferred ITC + Deferred FIT (5-65)

Working Capital Change = (WORKING CAP. × Mid-Inflation) - Cum. Working Capital (5-66)

Preferred Stock Due = Preferred Stock Equity/BOOK LIFE (5-67)

LTD Repayment = Repay Debt (5-68)

Loans Required = LTD Repayment + Pref. Dividends + Preferred Stock Due + Working Capital Change - **TOTAL INTERNAL** (5-69)

If Loans Required has a positive value, the determination of whether to issue interest-bearing government loans or interest-free partner loans is based on the current value of the plant-array variable CAPACITY FACTOR. When the two-year average CAPACITY FACTOR drops below the average of the TARGET CAPACITY FACTOR and the TRIGGER CAPACITY FACTOR, all required loans are financed by government loans; otherwise required loans are financed through cash advances from the partners. The basis for this algorithm is the assumption that if, in any two consecutive years, the plant exhibits a capacity factor less than the target capacity, this deficiency is due to an unplanned continuous outage possibly spanning two years. In this case, the algorithm checks whether or not the capacity factor during the outage period, prorated to one year, falls below the TRIGGER CAPACITY FACTOR.

If Loans Required > 0, then the amount of partner or government loans is calculated as follows:

$$\text{Partner Loan Repayment} = 0 \tag{5-70}$$

If the two-year average capacity factor ≤ (TARGET CAPACITY FACTOR + TRIGGER CAPACITY FACTOR)/2, then:

$$\text{Government Loans} = \text{Loans Required} \tag{5-71}$$

$$\text{Partner Loans} = 0 \tag{5-72}$$

$$\text{Repay Debt}_c = \text{Repay Debt}_p + \text{Loans Required} \div (\text{DEBT REPAYMENT PERIOD} - I) \tag{5-73}$$

where I is the number of years the plant has operated.

$$\text{LT Debt Rate}_c = [(\text{LT Debt Rate}_p \times \text{Total LT Debt}_p) + (\text{Loans Required}_c \times \text{LONG-TERM DEBT RATE})] \div (\text{Total LT Debt}_p + \text{Loans Required}_c) \tag{5-74}$$

$$\text{Total LT Debt}_c = \text{Total Debt}_p + \text{Loans Required}_c \tag{5-75}$$

If the two-year average capacity factor > (TARGET CAPACITY FACTOR + TRIGGER CAPACITY FACTOR)/2, then:

$$\text{Government Loans} = 0 \tag{5-76}$$

$$\text{Partner Loans} = \text{Loans Required} \tag{5-77}$$

If Loans Required < 0 then:

$$\text{Partner Loans} = 0 \tag{5-78}$$

$$\text{Government Loans} = 0 \tag{5-79}$$

$$\text{Partner Loan Repayment} = \min(-\text{Loans Required}, \text{Partner Loans Outst.}) \tag{5-80}$$

$$\text{TOTAL SOURCES} = \text{TOTAL INTERNAL} + \text{Partner Loans} + \text{Government Loans} \tag{5-81}$$

TOTAL USES = TOTAL SOURCES (5-82)

Common Dividends = **TOTAL SOURCES** - LTD Repayment
- Preferred Stock Due - Pref. Dividends
- Working Capital Change - Partner Loan Repayment (5-83)

Among the variables calculated in the Sources and Uses of Funds Statement, Partner Loans, Preferred Stock Due, Working Capital Change, and LTD Repayment are used in the Partnership Cashflow Analysis, and Common Dividends is used in the Retained Earnings Statement.

5.5 RETAINED EARNINGS STATEMENT

This section describes the calculation of equity remaining in the project and earned return on equity (ROE).

Total Common Equity = Begin. Common Equity + Common Stock Earnings (5-84)

END COMMON EQUITY = Total Common Equity - Common Dividends (5-85)

RETURN ON EQUITY = 100 × Common Stock Earnings
÷ Begin Common Equity (5-86)

END COMMON EQUITY is used in the Balance Sheet report, for the liability statement.

5.6 PARTNERSHIP CASHFLOW ANALYSIS

This section describes the calculation of after-tax cash flows to the partners. The partnership income is assumed to be consolidated at the partner's level for tax purposes. Furthermore, it is assumed that the partners can take full advantage of any tax benefits accruing from the partnership.

After-Tax Cash = Pre-Tax Part. Income - FIT Due (5-87)

Total Cash Avail. = After-Tax Cash + **TOTAL TAX DEPRECIATION**
+ Federal Subsidy (5-88)

NET TO PARTNERS = Total Cash Avail. - LTD Repayment
- Pref. Dividends - Preferred Stock Due
- Partner Loans - Working Capital Change (5-89)

CUMULATIVE TO PARTNERS$_c$ = **CUMULATIVE TO PARTNERS**$_p$
+ **NET TO PARTNERS**$_c$ (5-90)

5.7 BALANCE SHEET

This section describes the calculation of assets and liabilities.

Cum. Working Capital = WORKING CAP. × Mid-Inflation (5-91)

Gross Plant = Construction Book (5-92)

Accum. Plant $Dep._c$ = Accum. Plant $Dep._p$ + Plant Book Dep. (5-93)

Net Plant = Gross Plant - Accum. Plant Dep. (5-94)

Capitalized AFUDC = AFUDC Book (5-95)

Accum. AFUDC $Dep._c$ = Accum. AFUDC $Dep._p$ + AFUDC Book Dep. (5-96)

Net Capitalized AFUDC = Capitalized AFUDC - Accum. AFUDC Dep. (5-97)

TOTAL ASSETS = Cum. Working Capital + Net Plant + Net Capitalized AFUDC (5-98)

LTD Outstanding = min [(Total Debt - LTD Repayment), 0] (5-99)

Total LT $Debt_c$ = Total LT $Debt_p$ - LTD $Repayment_p$ (5-100)

Cum. Deferred ITC_c = Cum. Deferred ITC_p + Deferred ITC_c (5-101)

Cum. Deferred FIT_c = Cum. Deferred FIT_p + Deferred FIT_c (5-102)

- If years of operation < BOOK LIFE, then:

  LTD Due in Next Year = Repay Debt (5-103)

  P.S. Due in Next Year = Preferred Stock Due (5-104)

- If years of operation = BOOK LIFE, then:

  P.S. Due in Next Year = 0 (5-105)

  LTD Due in Next Year = 0 (5-106)

Preferred Stock $Equity_c$ = Preferred Stock Equity - Preferred Stock Due (5-107)

Pref. Stock Outstanding = min [(Preferred Stock Equity - Preferred Stock Due), 0] (5-108)

Partner Loans $Outst._c$ = Partner Loans $Outst._p$ + Partner Loans - Partner Loan Repayment (5-109)

Total Stock Equity = P.S. Due in Next Year + Pref. Stock Outstanding + **END COMMON EQUITY** + Partner Loans Outst. (5-110)

TOTAL LIABIL. & EQUITY = LTD Due in Next Year + LTD Outstanding + Cum. Deferred ITC + Cum. Deferred FIT + Total Stock Equity (5-111)

- If GROSS(0)/NET(1)PROPTXFLAG = 1, then:

  Prop. Tax Base = Net Plant + Net Capitalized AFUDC (5-112)

- If GROSS(0)/NET(1)PROPTXFLAG = 0, then:

  Prop. Tax Base simply remains as defined in Eq. 5-8.

5.8 AFTER-TAX CASH FLOWS TO UTILITY INVESTORS

This analysis calculates the actual after-tax cash flows realized by the partners. A separate analysis is performed for the construction period and the operation period.

During Construction Period:

Cash Inv. = Common Equity (5-113)

Tax Ben. = Annual Tax Accrual + ITC Rate (5-114)

Net A.T. Cash Flow = Tax Ben. - Cash Inv. (5-115)

During Operation Period:

Tax Due = FIT Due (5-116)

Cash Dist. = FIT due + **NET TO PARTNERS** (5-117)

Net A.T. Cash Flow = Cash Dist. - FIT Due (5-118)

In each case,

$$\text{Cum. A.T. Cashflow}_c = \text{Cum. A.T. Cashflow}_p + \text{Net A.T. Cashflow}_c \quad (5\text{-}119)$$

In addition, the net present value of the cash flows in base year dollars is calculated by first calculating the net after-tax cash flow in base-year dollars:

Net A.T.C.F. Real = Net A.T. Cashflow/Cum. Inflation (5-120)

$$\text{Cm.Dis.R.A.T.C.F.} = \text{Net.A.T.C.F.Real} \times \text{Discount Factor}^N \quad (5\text{-}121)$$

where

Cm.Dis.R.A.T.C.F. is the cumulative discounted real after-tax cash flow, in base-year dollars

N = Year - BASE YEAR + 1,

Discount Factor is as defined in Eq. 5-124, and

Cum. Inflation runs from the *beginning* of the base year to the *end* of the current year.

Finally, the realized internal rate of return on net equity investment (ROI) is calculated in terms of as-spent (inflated) dollars and base-year dollars. In each case an iterative procedure is used to calculate the discount rate that results in a net present value of zero for the net after-tax cash flows. A table of ROI covering the entire

operation period is calculated, where the ROI for a given year is based on all cash flows through that year.

5.9 BUSBAR COST ANALYSIS

This analysis includes a plot of annual busbar costs and a calculation of levelized busbar costs in as-spent and base-year dollars. Either a pretax or after-tax discount rate can be specified.

- PRE(0)AFTER(1)TXDISCFLAG = 0:

$$\begin{aligned}\text{Discount Rate} = {} & (\text{TARGET RETURN ON EQUITY} \times \text{EQUITY FRACTION}) \\ & + \text{Preferred Rate} \\ & \times (1 - \text{DEBT FRACTION} - \text{EQUITY FRACTION}) \\ & + (\text{LT Debt Rate} \times \text{DEBT FRACTION})\end{aligned} \tag{5-122}$$

- PRE(0)AFTER(1)TXDISCFLAG = 1:

$$\begin{aligned}\text{Discount Rate} = {} & (\text{TARGET RETURN ON EQUITY} \times \text{EQUITY FRACTION}) \\ & + \text{Preferred Rate} \\ & \times (1 - \text{DEBT FRACTION} - \text{EQUITY FRACTION}) \\ & + (1 - \text{INCOME TAX RATE}) \times \text{LT Debt Rate}\end{aligned} \tag{5-123}$$

In either case:

$$\text{Discount Factor} = 1/(1 + \text{Discount Rate}) \tag{5-124}$$

The levelized busbar cost (in as-spent dollars) is given by:

$$\text{Levelized Busbar Cost} = \sum_{i=1}^{i=L} [B_i \times d^i/(1 - d^L)] \tag{5-125}$$

where

L = LEVELIZATION PERIOD, years,

B = BUSBAR COST,*

d = Discount factor, and

i = an index that runs through the number of years in LEVELIZATION PERIOD.

---

*The BUSBAR COST can either be the input value, in the case of top-down analysis, or the value calculated by Eq. 5-37, in the case of a bottom-up analysis.

A levelized busbar cost in base-year dollars is also calculated. In this case, a real discount rate is defined by:

$$\text{Real Discount Rate} = (1 + \text{Discount Rate})/(1 + \text{GENERAL INFLATION}_e) \quad (5\text{-}126)$$

where the subscript e indicates that the value of GENERAL INFLATION used is the value for the input year CONSTRUCTION END.

Busbar cost in base-year dollars is given by:

$$\text{Busbar (base-year \$)} = \text{BUSBAR COST/Cum. inflation} \quad (5\text{-}127)$$

where Cum. Inflation is the cumulative inflation from the *beginning* of the base year to the *end* of the current year.

5.10 LIMITATIONS

The current version of BREEDER is based on the assumptions used in several earlier financial analyses performed for CoMO.[3] These assumptions include:

- Return is based on common equity;
- The allowance for funds used during construction is treated on the basis of the gross funds;
- Investment tax credits flow through ratably;
- The depreciation base for calculating deferred taxes equals the full book depreciation base.

Subsequent versions of BREEDER, currently under development, will include the following options:

- Return on rate base or on equity;
- The allowance for funds used during construction will be treated on either a gross or a net basis;
- Investment tax credits will either flow through ratably or yield a rate-base reduction;
- Interest during construction will be capitalized under provisions of Section 189 or Section 266 of the Internal Revenue Code.

6 REPORT WRITER

This section describes the reports provided by BREEDER and the user interface to the report writer. Four routines (REPORT, SUMARY, CALROI, and PLOTBUSBAR) are used to print the reports, calculate returns on investment, and plot busbar costs. Sample printouts are contained in Appendix D.

6.1 REPORT DESCRIPTIONS

Nine separate reports are available: a special summary report, six detailed annual financial reports covering the operational period, a summary of busbar costs, and a summary of the cash flow analysis for the full analysis period. A summary of each report is presented below:

- SUMMARY contains a summary of construction expenses (direct, startup, AFUDC, and working capital) and a breakdown of financing (total value, percentage, weighted cost as percent), and total IDC and FIT accruals. It also contains a summary of analysis options in force and analysis results, including ROI, levelized busbar costs, discount rates, and fixed charge rates.

- BALANCE SHEET provides annual summaries of plant assets and liabilities.

- INCOME STATEMENT presents annual summaries of revenue and expense breakdowns.

- SOURCES AND USES OF FUNDS STATEMENT summarizes the annual sources of funds (internal and external) and uses of funds (dividends and retirements).

- RETAINED EARNINGS STATEMENT AND TAX DEPRECIATION WORKSHEET illustrates changes in the common equity position and lists the accelerated tax depreciation schedule.

- BOOK TAX AND REGULATORY TAX CALCULATION summarizes the calculation of book taxes and deferred taxes.

- PARTNERSHIP RETURN AND CASHFLOW ANALYSIS shows the calculation of actual taxes paid (FIT Due) and annual and cumulative cash flows to the partners, assuming joint project ownership.

- CASH FLOW SUMMARY presents a summary of cash flows (investment and net income) and tax effects (benefits and taxes due) for the complete analysis period (construction plus operation). A summary of ROI by year is also included.

- BUSBAR PLOT includes a summary of annual busbar costs in graphical and tabular form and levelized busbar costs in as-spent and base-year dollars.

6.2 USER INTERFACE

The interface offers a menu format. The first menu allows the user to preview results on the screen or obtain a listing on the printer. Since most printers used with microcomputers support relatively slow data transfer rates (1200 baud or less), use of the screen option can provide a relatively fast (9600 baud) preview of the reports.

Next the user is presented with a menu of the nine reports available for listing. The user may select an individual report or "option 10," a special option to list all the reports sequentially. The ESCAPE key can be used to interrupt a long sequence of displays (in screen mode only); hitting the ESCAPE key in response to the "<STRIKE ANY KEY TO CONTINUE>" message returns the user to the report menu.

Due to limits imposed by an 80-column output format, only eight years of the analysis can be displayed at a time. For analysis periods longer than eight years, the report consists of a number of subreports, each eight years in length. Alternatively, "option 12" can be chosen; this option allows the user to select a subset of analysis years for display. Years can be entered in any order. Each year is checked against the analysis-period range. Entry of 0 terminates the selection procedure.

Upon completion of the report-writer routine, the user is requested to choose between performing another run or quitting. If the user chooses to make another run, the analysis begins again at the start of the input sequence (see FILEIN, Section 3.2).

REFERENCES

1. *Meeting Our Need for Electric Energy: The Role of Nuclear Power*, U.S. Department of Energy, Assistant Secretary for Nuclear Energy, Plans and Evaluation Division (in press, 1984).

2. Taylor, J.T., et al., *A Government/Industry Plan for Demonstration of Breeder Reactor Technology*, Palo Alto, Calif., EPRI draft report (Jan. 1982).

3. *A Program Providing Financing for a Cooperative Effort between the Public Utility Industry and the Federal Government in Designing, Constructing and Operating a First-of-a-Kind Commercial Scale Liquid Metal Fast Breeder Reactor*, Consolidated Management Office, EPRI, One Energy Center, Naperville, Ill., draft report (May 1983).

4. Snyder, C.R., and Hayman, R.B., *Economic Goals for the Large-Scale Prototype Breeder Program*, Consolidated Management Office, EPRI, One Energy Center, Naperville, Ill., draft report (April 1984).

APPENDIX A: BREEDER SOURCE CODE

This appendix contains a listing of the source code for BREEDER and a suggested overlay structure. The PL/I source code is identical for both the CP/M and IBM PC versions. However, since the IBM PC uses a memory-map technique for controlling the monitor, and Kaypro II and most other CP/M machines do not, a separate set of screen-control routines (PLIDIO), written in assembler language, has been included. The full set of PL/I source-code files and the assembler language file are contained on the two source-code diskettes. The Digital Research PL/I-80 and RMAC compilers can be used to compile the CP/M PL/I and assembler files, respectively. The Digital Research PL/I-86 (R1.0 DOS) and RASM-86 compilers can be used to compile the IBM PC PL/I and assembler files, respectively.

Figure A.1 illustrates a suggested overlay structure. The root module contains the main driver routine and the set of utility routines. The first overlay contains the main analysis routines. Finally, the second overlay includes three routines used in conjunction with report writing (SUMARY, CALROI, and PLOTBUSBAR).

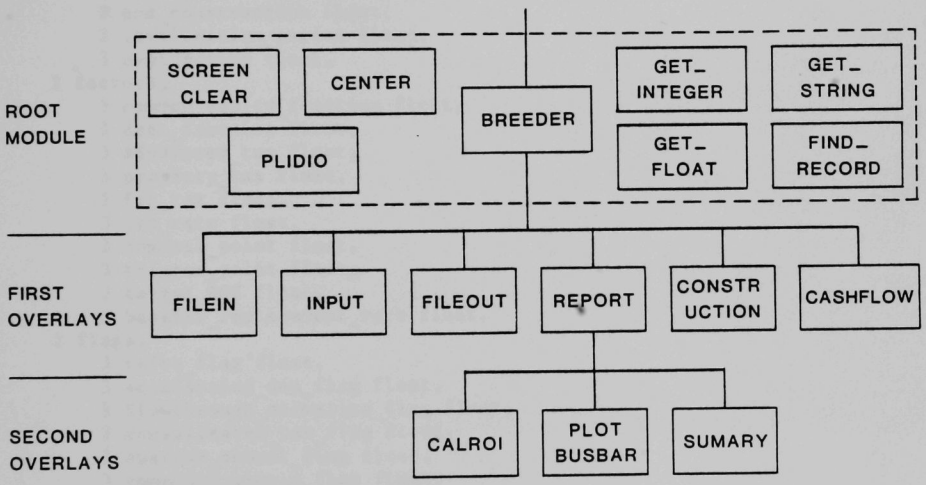


FIGURE A.1 Suggested Overlay Structure for CP/M Microcomputers

A.1 DECLARE FILES

```
/*****************************************************************************
* File I N P U T . D C L : input variable array structure.                   *
*****************************************************************************/
    %replace num_scalar by 5;
    %replace num_types by 8;
    %replace num_types_plus_1 by 9;
    %replace years_analysis by 30;
    %replace years_analysis_plus_1 by 31;
    %replace years_equity by 50;
    %replace years_construction by 15;
    %replace years_construction_plus_1 by 16;
    %replace quarters_construction_plus_1 by 61;
dcl 1 input_data static external,
    2 header char(79) var,
    2 years,
        3 base_year float,
        3 start_construction float,
        3 end_construction float,
        3 levelization_period float,
        3 debt_period float,
    2 factors,
        3 common_equity_fraction float,
        3 debt_fraction float,
        3 advalorem_tax float,
        3 property_tax float,
        3 fit_tax float,
        3 itc_rate float,
        3 nominal_point float,
        3 trigger_point float,
        3 target_ROE float,
        3 backfit_replacement_rate float,
    2 flags,
        3 tefra_flag float,
        3 accelerated_dep_flag float,
        3 flowthrough_normalize_flag float,
        3 consolidated_tax_flag float,
        3 quarter_annual_flag float,
        3 compound_simple_flag float,
        3 gross_net_flag float,
        3 pre_after_tax_flag float,
        3 net_equity_flag float,
        3 bottom_up_top_down_flag float,
    2 plant_variables,
        3 size float,
        3 book_life float,
        3 tax_life float,
        3 startup_cost float,
        3 working_capital float,
        3 heat_rate float,
    2 rates,
        3 inflation(years_analysis) float,
        3 ST debt(years analysis) float,
```

```
    3 LT_debt(years_analysis) float,
    3 preferred_stock(years_analysis) float,
    3 common_equity(years_equity) float,
2 plant_arrays,
    3 capacity_factor(years_analysis) float,
    3 fuel_cost(years_analysis) float,
    3 operation_maintanence(years_analysis) float,
    3 working_capital(years_analysis) float,
    3 Federal_subsidies(years_analysis) float,
    3 busbar_costs(years_analysis) float,
2 construction_costs,
    3 plant(years_construction) float,
    3 last entry;
```

```
/*****************************************************************************
* File C O N S T R . D C L:structure for variables calculated in CONSTRUCTION *
*****************************************************************************/
dcl 1 sums static external,
    2 plant_totals,
        3 construction float,
        3 AFUDC float,
        3 construction_real float,
        3 AFUDC_real float,
        3 common_equity float,
        3 LT_debt float,
        3 LT_debt_rate float,
        3 preferred_stock float,
        3 preferred_rate float,
        3 ITC_accrual float,
        3 tax_accrual float,
        3 book_dep_plant float,
        3 AFUDC_dep_plant float,
        3 amort_ITC_book float,
        3 book_depreciation float,
        3 other_dep float,
    2 cashflows(years_construction_plus_1),
        3 before_tax float,
        3 interest_credit float,
        3 ITC_credit float,
        3 after tax float;
```

```
/*****************************************************************************
* File O U T P U T . D C L : Output variable structure.                      *
*****************************************************************************/
dcl 1 output_data static external,
    2 assets,
        3 accum_dep(years_analysis) float,
        3 net_plant(years_analysis) float,
        3 accum_AFUDC_dep(years_analysis) float,
        3 net_capitalized_AFUDC(years_analysis) float,
        3 total_assets(years_analysis) float,
    2 liabilities,
        3 LT_debt_repayment(years_analysis_plus_1) float,
        3 LT_debt_outstanding(years_analysis) float,
        3 deferred_ITC(years_analysis) float,
        3 deferred_FIT(years_analysis) float,
        3 preferred_stock_outstanding(years_analysis) float,
        3 preferred_stock_repayment(years_analysis_plus_1) float,
        3 cum_partner_loans(years_analysis) float,
        3 total_equity(years_analysis) float,
    2 revenues,
        3 electric_sales(years_analysis) float,
        3 total_revenues(years_analysis) float,
    2 expenses,
        3 backfit_replacement(years_analysis) float,
        3 ad_valorem_tax(years_analysis) float,
        3 total_O_and_M(years_analysis) float,
        3 property_tax(years_analysis) float,
        3 total_pre_debt_expense(years_analysis) float,
        3 total_pre_debt_income(years_analysis) float,
        3 LT_debt_interest(years_analysis) float,
        3 preferred_stock_div(years_analysis) float,
        3 net_income(years_analysis) float,
        3 common_earnings(years_analysis) float,
    2 sources,
        3 amort_ITC(years_analysis) float,
        3 total_internal(years_analysis) float,
        3 partner_loans(years_analysis) float,
        3 DOE_loans(years_analysis) float,
        3 total_sources(years_analysis) float,
    2 uses,
        3 common_stock_div(years_analysis) float,
        3 delta_working_capital(years_analysis) float,
        3 partner_loan_repayment(years_analysis) float,
    2 retained_earnings,
        3 beginning_common_equity(years_analysis) float,
        3 total_common_equity(years_analysis) float,
        3 end_common_equity(years_analysis) float,
    2 book_tax,
        3 Pre_tax_income_book(years_analysis) float,
        3 FIT_book(years_analysis) float,
        3 net_FIT_book(years_analysis) float,
```

```
2 FERC_tax,
    3 startup_dep(years_analysis) float,
    3 five_year_property(years_analysis) float,
    3 ten_year_property(years_analysis) float,
    3 fifteen_year_property(years_analysis) float,
    3 total_tax_dep(years_analysis) float,
    3 pre_tax_income_FERC(years_analysis) float,
    3 FIT_tax_FERC(years_analysis) float,
    3 credit_ITC(years_analysis) float,
    3 deferred_FIT_taken(years_analysis) float,
    3 cum_deferred_FIT(years_analysis) float,
    3 net_FIT_FERC(years_analysis) float,
2 partnership_return,
    3 after_tax_cash(years_analysis) float,
    3 total_cash_avail(years_analysis) float,
    3 net_to_partners(years_analysis) float,
    3 cum_to_partners(years_analysis) float;
```

```
/*****************************************************************************
* File  D I O M O D . D C L : contains the entry DCLs for the direct I/O     *
* and cursor control routines in the assembler language program PLIDIO.     *
*  It also contains DCLs for the special control characters used in BREEDER. *
*****************************************************************************/

    dcl
        coninp entry returns(char(1)),
        conout entry(char(1)),
        clearscreen entry,
        upcursor entry(fixed),
        downcursor entry(fixed),
        leftcursor entry(fixed),
        rightcursor entry(fixed);

    dcl 1 controls static external,
        2 new_page char(1),
        2 bell char(1),
        2 escape char(1),
        2 carriage_return char(1),
        2 backspace char(1),
        2 line_feed char(1),
        2 analysis_period fixed;
```

A.2 PROGRAM

```
/*****************************************************************************
*  B R E E D E R :                                                          *
* This is the DRIVER routine for the breeder evaluation program.            *
* It prints out the various program description sceens using file           *
* BREEDER.DAT.  It calls the following routines:                            *
*     FILEIN - inputs data file                                             *
*     INPUT  - permits modification and listing of all input                *
*     FILEOUT- outputs data file                                            *
*     CONSTRUCTION - calculates construction expenditures                   *
*     CASHFL - calculates cashflow variables during operating period        *
*     REPORT - writes various financial summary reports                     *
*     GET_INTEGER - obtains integer input from consol                       *
*     SCREEN_CLEAR - wait for a character from terminal and clear screen.   *
*     CLEARSCREEN - clears the screen and positions cursor at upper left.   *
*                                                                           *
*     Original version  8/5/83                                              *
*     Version 1.5  4/15/84                                                  *
*        Change clearscreen to an external subroutine to permit             *
*        compatibility between CP/M and IBM/PC-DOS versions.                *
*****************************************************************************/
breeder:
    procedure options(main);

    %include 'input.dcl';
    %include 'constr.dcl';
    %include 'output.dcl';
    %include 'diomod.dcl';

    dcl (crt,printer,infile) file,
        (filein,fileout,input,construction,cashflow,report) entry,
        screen_clear entry(char(1),fixed),
        get_integer entry(fixed,fixed),
        continue char(1),
        buffer char(81) var,
        (i,escape_flag,irun) fixed;

/*****************************************************************************
* Initialize program BREEDER                                                *
*****************************************************************************/
    call initial;
    irun = 1;
        do while(irun = 1);
/*****************************************************************************
* Read input data file                                                      *
*****************************************************************************/
        call filein;
/*****************************************************************************
* Modify and/or list input data file                                        *
*****************************************************************************/
        call input;
/*****************************************************************************
```

```
                                                                                *
* Write output data file
*****************************************************************************/
        call fileout;
/*****************************************************************************
                                                                                *
* Perform plant construction analysis
*****************************************************************************/
        call construction;
/*****************************************************************************
                                                                                *
* Perform cashflow analysis for plant operating period
*****************************************************************************/
        call cashflow;
/*****************************************************************************
                                                                                *
* Write financial reports
*****************************************************************************/
        call report;
        irun = -1;
/*****************************************************************************
                                                                                *
* Print Menu and determine option.
*****************************************************************************/
            do while(irun<0!irun>1);
            call clearscreen;
            put file(crt) edit('Enter 0 - No More Runs',
            '1 - Do Another Run') (a,col(7));
            put file(crt) skip edit('? ') (a);
            call get_integer(escape_flag,irun);
            if irun<0!irun>1 then put file(crt) edit(bell) (a);
            end;
        end;
/*****************************************************************************
                                                                                *
* Terminate program BREEDER.
*****************************************************************************/
    call done;
    stop;

/*****************************************************************************
                                                                                *
*  I N I T I A L :
* This subroutine opens the CRT and PRINTER files and prints the first three  *
* CRT screens.                                                                 *
*****************************************************************************/
    initial:
        procedure;

        open file(crt) stream output title('$con');
        open file(printer) stream output title('$lst');
        open file(infile) stream input title('breeder.dat');
        call clearscreen;
        put file(crt) skip(6);
            do i = 1 to 11;
            read file(infile) into(buffer);
            put file(crt) edit(buffer) (a);
            end;
        call screen_clear(continue,6);
        if continue = escape then goto skip;
        put file(crt) skip(9);
```

```
            do i = 1 to 5;
            read file(infile) into(buffer);
            put file(crt) edit(buffer) (a);
            end;
        call screen_clear(continue,9);
        if continue = escape then goto skip;
        put file(crt) skip(5);
            do i = 1 to 12;
            read file(infile) into(buffer);
            put file(crt) edit(buffer) (a);
            end;
        call screen_clear(continue,6);
skip:   close file(infile);
        return;
        end initial;

/*****************************************************************************
*  D O N E :                                                                 *
* This subroutine prints a final CRT message and closes the CRT and          *
* PRINTER files.                                                             *
*****************************************************************************/
    done:
        procedure;

        close file(printer);
        put file(crt) skip edit('Normal End of Run',
        'Thank you for using BREEDER') (a,skip);
        put file(crt) skip(4);
        close file(crt);
        return;
        end done;

    end breeder;
```

```
/*****************************************************************************
*                                                                            *
*  F I L E I N :                                                             *
* This routine permits creation of a new data file or input the current     *
* LSPBFILE.DAT file.  For creation of a new file all input variables are    *
* zeroed and ANALYSIS_PERIOD is set to 0.  Upon successful input of the     *
* LSPBFILE.DAT file ANALYSIS_PERIOD is set to 1.                             *
*                                                                            *
*    Original Version  8/5/83                                                *
*    Version 1.5 4/15/84                                                     *
*        Change clearscreen to subroutine call to provide compatibility      *
*        between CP/M and IBM/PC-DOS versions.                               *
*****************************************************************************/
filein:
    procedure;

    %include 'input.dcl';
    %include 'diomod.dcl';

    dcl (crt,in) file,
        dummy(1) float based(address),
        (addlow,addhigh,address) pointer,
        get_integer entry(fixed,fixed) external,
        (i,idata,variable_number) fixed;

    idata = -1;
/*****************************************************************************
*                                                                            *
* Print Menu and determine option.                                           *
*****************************************************************************/
        do while(idata<1 ! idata>2);
        call clearscreen;
        put file(crt) edit('Enter 1 - To Create a New Data File',
        '2 - To Use an Existing File') (a,col(7));
        put file(crt) skip edit('? ') (a);
        call get_integer(i,idata);
        if idata<1 ! idata>2 then put file(crt) edit(bell) (a);
        end;
    if idata = 1
        then do;
/*****************************************************************************
* NEW INPUT option.  Set ANALYSIS_PERIOD to 0 and zero input array.          *
*****************************************************************************/
        analysis_period = 0;
        address = addr(input_data.years);
        header = '';
        addhigh = addr(input_data.last_entry);
        addlow = addr(input_data.years);
        variable_number = unspec(addhigh) - unspec(addlow);
        variable_number = divide(variable_number,4,15);
            do i=1 to variable_number;
            dummy(i) = 0;
            end;
        end;
```

```
        else do;
/*****************************************************************************
* Use file : LSPBFILE.DAT.  Open file IN; Read data; and Close file IN.      *
*****************************************************************************/
        analysis_period = 1;
        open file(in) input sequential env(b(1024)) title('LSPBFILE.DAT');
        read file(in) into(input_data);
        close file(in);
        end;
    return;
    end filein;
```

```
/*****************************************************************************
*  I N P U T :                                                               *
* This routine enables entering, modifing and listing of all input array     *
* data contained in the file LSPBFILE.DAT.  The routine protects against     *
* user input errors by trapping input of invalid characters.  The            *
* routine also performs data consistency and reasonalbeness checks on        *
* all user inputs and modifications.  Several program features permit        *
* easy entry and modification of input array variables:                      *
*   - Data values can be input for a single year, a range of years           *
*     or the entire array;                                                   *
*   - Multiyear input can be accomplished by entering each year's data       *
*     or specifing the first year's value plus an annual escalation rate.    *
* The routine makes use of eight subroutines:                                *
*   - FIND_RECORD which obtains data descriptor text from a specially        *
*     created keyed,variable length data file [INPUT.DAT].                   *
*   - GET_INTEGER returns an integer variable and read error code.           *
*   - SCREEN_CLEAR clears the screen and moves CURSOR to upper left.         *
*   - GET_FLOAT returns a floating point variable and read error code.       *
*   - GET_STRING returns a string variable and read error code.              *
*   - CLEARSCREEN clears the screen and positions CURSOR at upper left.      *
*   - RIGHTCURSOR moves the CURSOR to the right.                             *
*   - UPCURSOR moves the CURSOR up.                                          *
* Data input categories are:                                                 *
*   - Header (79 characters maximum),                                        *
*   - Year Variables (6),                                                    *
*   - Program Factors (10),                                                  *
*   - Program Flags (10),                                                    *
*   - Plant Variables (6),                                                   *
*   - Plant Arrays (5),                                                      *
*   - Rates Arrays (6),                                                      *
*   - Construction Cost Array.                                               *
*                                                                            *
*     Original Version  8/5/83                                               *
*     Version 1.1  9/15/83                                                   *
*        Split Program Factors into:                                         *
*            - Program Factors (8),                                          *
*            - Program Flags(10).                                            *
*       Add BOTTOM_UP_TOP_DOWN_FLAG                                          *
*       Add BUSBAR_COSTS to Plant Arrays                                     *
*       Reverse order of Plant Arrays and Rate Arrays                        *
*       Increase COMMON_EQUITY Rate Array to 50                              *
*       Insert FUEL_COST array after CAPACITY_FACTOR array in Plant Arrays   *
*    Version 1.2  10/10/83                                                   *
*       Add four variables to Program Factors:                               *
*            - TARGET_CAPACITY_FACTOR,                                       *
*            - TRIGGER_CAPACITY_FACTOR,                                      *
*            - TARGET_ROE,                                                   *
*            - BACKFIT_REPLACEMENT_RATE.                                     *
*       Change BACKFIT_REPLACEMENT array in Plant Arrays to FEDERAL_SUBSIDIES *
*    Version 1.4 11/10/83                                                    *
*       Change HIGH limits:                                                  *
*            - BASE YEAR : 2000                                              *
*            - CONSTRUCTION START : 2050                                     *
*            - CONSTRUCTION END : 2065                                       *
*            - START DEBT REPAYMENT : 2100                                   *
*            - Year Limits : 100                                             *
*       Limit Levelization Period to YEARS ANALYSIS ([illegible]
```

```
*    Version 1.5  4/15/84                                                    *
*       Change clearscreen to a subroutine call to provide compatability     *
*       between CP/M and IBM/PC:MS-DOS versions.                             *
*       Add two PROGRAM FLAGS:                                               *
*           - FLOWTHRU(0)/NORM.(1) FLAG                                      *
*           - UNCONS(0)/CONS(1)Tx. FLAG                                      *
*****************************************************************************/
input:
    procedure;

    %include 'input.dcl';
    %include 'diomod.dcl';

    dcl address(num_types_plus_1) pointer,
        find_record entry(fixed,fixed,char(79) var),
        screen_clear entry(char(1),fixed),
        get_float entry(fixed,float),
        get_integer entry(fixed,fixed),
        get_string entry(char(79) var),
        offset_number(num_types_plus_1) fixed(7),
        low(9) static float init((4)'1982','0','1','8000','0','0'),
        high(9) static float init('1990','2050','2065','2100','1','100',
            '12000','2000','100'),
        (i,j,k,l,imod,jmod,kmod,lmod,imenu,ilist,index_subtitle,fix_value,
        save_value,type_save,escape_flag,nlow,nhigh,first_year,last_year,
        array_size,year_base,year_last,number_entries,year) fixed,
        (temp,value,escalation) float,
        float_array(1) float based(arg_add),
        (crt,printer) file,
        continue char(1),
        (buffer,title,subtitle) char(79) var,
        arg_add pointer;

/*****************************************************************************
* Initialize input (INITIAL) and call NEW_INPUT if ANALYSIS_PERIOD          *
* is 0 (set in FILEIN).                                                     *
*****************************************************************************/
    call initial;
    if analysis_period=0 then call new_input;
    imenu = -1;
        do while(imenu = 0);
        imenu = -1;
/*****************************************************************************
*  Print main menu to LIST or MODIFY values                                 *
*****************************************************************************/
            do while(imenu < 0 ! imenu > 2);
            call clearscreen;
            put file(crt) skip(0) edit('Enter 0 - To Exit',
                '1 - To List File','2 - To Modify File') (a,col(7));
            put file(crt) skip edit('? ') (a);
            call get_integer(escape_flag,imenu);
            if (imenu < 0 ! imenu > 2) then put file(crt) edit(bell) (a);
            end;
        if imenu = 1 then call list; else if imenu = 2 then call modify;
        end;
```

```
/*****************************************************************************
*  Close INPUT.DAT file and exit INPUT                                       *
*****************************************************************************/
    call find_record(type_save,-1,buffer);
    return;

/*****************************************************************************
*  I N I T I A L I Z E :                                                     *
*  This routine initializes INPUT                                            *
*   -Variable labels are contained in the specially keyed data file: INPUT.DAT*
*   -ADDRESS points to major categories within the input array structure    *
*       The first category is always a HEADER (char(79) var)                *
*       Categories 2 - num_scalar are scalar float variables                *
*       The remaining categories (num_scalar+1-num_types) are arrays        *
*   -The number of variables in each category is calculated and stored      *
*    in the array OFFSET_NUMBER                                             *
*****************************************************************************/
    initial:
        procedure;

        buffer = 'input';
        call find_record(type_save,0,buffer);
        address(1) = addr(header);
        address(2) = addr(years);
        address(3) = addr(factors);
        address(4) = addr(flags);
        address(5) = addr(plant_variables);
        address(6) = addr(rates);
        address(7) = addr(plant_arrays);
        address(8) = addr(construction_costs);
        address(9) = addr(last_entry);
        offset_number(1) = 1;
        offset_number(2) = 2;
            do ilist = 3 to num_types+1;
            i = unspec(address(ilist)) - unspec(address(ilist - 1));
            if ilist <= num_scalar+1 then i = divide(i,4,15);
            else if ilist=num_types+1 then i=divide(i,years_construction*4,15);
            else i = divide(i,years_analysis*4,15);
            offset_number(ilist) = offset_number(ilist - 1) + i;
            end;
        return;
        end initial;

/*****************************************************************************
*  N E W _ I N P U T :                                                       *
*  New input file requested.  Sequentially modify all variables              *
*****************************************************************************/
    new_input:
        procedure;

        call clearscreen;
        put file(crt) skip(0) edit('NEW INPUT option selected:') (a);
        put file(crt) skip(1) edit(
  'All variables have been initialized to 0.  You will be requested to change',
  'each value  sequentially.  To TERMINATE execution, strike  ESCAPE  key now',
        'or when requested to change values.') (col(5`
```

```
        call screen_clear(continue,1);
        if continue = escape then call terminate;
            do imod = 1 to num_types;
            call find_record(type_save,imod,title);
                do jmod=offset_number(imod) to offset_number(imod+1)-1;
                call clearscreen;
                call find_record(type_save,num_types+jmod,subtitle);
                if imod <= num_scalar then call get_data(imod,jmod);
                    else do;
                    call clearscreen;
                    put file(crt) skip(0) edit(title) (a);
                    put file(crt) skip edit(subtitle) (col(5),a);
                    end;
                call mod_data(imod,jmod);
                if escape_flag = -1 then call terminate;
                end;
            end;
        return;
        end new_input;

/*****************************************************************************
*  L I S T :                                                                 *
*  This routine prints the menu for listing input values and is the          *
*  the driver for the NUM_TYPES of major input categories                    *
*****************************************************************************/
    list:
        procedure;

        ilist = -1;
            do while(ilist = 0);
            ilist = -1;
/*****************************************************************************
* Print LIST menu and obtain option value.                                   *
*****************************************************************************/
                do while(ilist < 0 ! ilist > 2);
                call clearscreen;
                put file(crt) skip(0) edit('Enter 0 - To Exit',
                    '1 - To Preview on Screen','2 - To List on Printer')
                    (a,col(7));
                put file(crt) skip edit('? ') (a);
                call get_integer(escape_flag,ilist);
                if ilist < 0 ! ilist > 2 then put file(crt) edit(bell) (a);
                end;
            if ilist = 1 then call clearscreen;
            else if ilist = 2 then put file(printer) skip edit(new_page) (a);
            else return;
/*****************************************************************************
* Loop over NUM_TYPES categories.                                            *
*****************************************************************************/
                do i = 1 to num_types;
                call find_record(type_save,i,title);
                if ilist=1 then call list_sub(i,crt);
                else call list_sub(i,printer);
                if ilist = 1 then if i >= num scalar
```

```
                then do;
                if continue = escape then call screen_clear(continue,1);
                else;
                if continue = escape then return;
                end;
            else if i= 1 then put file(crt) skip; else;
            else if i=6 then put file(printer) skip edit(new_page) (a);
            else put file(printer) skip;
            end;
        end;
    return;
    end list;

/*****************************************************************************
*  L I S T _ S U B :                                                         *
* This routine lists the major input category TITLES and is the driver      *
* for LIST_DATA.                                                             *
*    - I is the major category index.                                        *
*    - X is the output device (CRT or PRINTER).                              *
*****************************************************************************/
    list_sub:
        procedure(i,x);

        dcl x file,
            i fixed;

        if i > 2 ! (x = printer & i = 2) then put file(x) edit(title,':') (a);
        k = 1;
            do index_subtitle = offset_number(i) to (offset_number(i+1) - 1);
            call find_record(type_save,num_types+index_subtitle,subtitle);
            if i>num_scalar then put file(x) skip edit('     ') (a);
            if i>=2&i<=num_scalar then if divide(k,2,15)*2=k then put file(x)
            edit(' ') (col(45),a); else put file(x) skip edit('     ') (a);
            call list_data(i,index_subtitle,x);
            if continue = escape then return;
            k = k + 1;
            end;
        return;
        end list_sub;

/*****************************************************************************
*  L I S T _ D A T A :                                                       *
* This routine lists subtitles J and all values for major category I on     *
* device X.  It also calculates YEAR_BASE, YEAR_LAST, ARRAY_SIZE,            *
* and NUMBER_ENTRIES for each array.                                         *
*    - I is the major category index,                                        *
*    - J is the variable index, and                                          *
*    - X is the output file (CRT or PRINTER).                                *
*****************************************************************************/
    list_data:
        procedure(i,j,x);

        dcl x file,
            (i,j) fixed;
```

```
        arg_add = address(i);
        k = j - offset_number(i) +1;
        continue = ' ';
        if i>=2 & i<=num_scalar then put file(x) edit(subtitle,':') (a(25),a);
        else if i = 1 then put file(x) edit(subtitle,':') (a);
        if i=1
            then do;
            put file(x) edit(header) (x(2),a);
            end;
/*****************************************************************************
* This code formats each of the scalar categories.                           *
*****************************************************************************/
        else if i=2
            then do;
            put file(x) edit(float_array(k)) (f(8,2));
            end;
        else if i=3 then put file(x) edit(float_array(k))
            (f(8,4));
        else if i=4 then put file(x) edit(float_array(k)) (f(2,0));
        else if i=num_scalar  then put file(x) edit(float_array(k))
            (f(8,2));
            else do;
/*****************************************************************************
* Set range for array: YEAR_BASE,YEAR_LAST,ARRAY_SIZE,NUMVER_ENTERIES        *
*****************************************************************************/
            call set_range(i,j);
            nlow = (k-1)*array_size + 1;
            year_base = year_base - nlow;
            nhigh = nlow + number_entries - 1;
            if j=offset_number(7)!j=offset_number(7)+1!i=6
            then put file(x) skip edit((1+year_base,float_array(1) do 1=nlow
            to nhigh)) (col(12),4(f(4,0),f(7,4),x(3)),f(4,0),f(7,4));
            else put file(x) skip edit((1+year_base,float_array(1)
            do 1=nlow to nhigh)) (col(12),4(f(4,0),f(7,2),x(3)),f(4,0),f(7,2));
            if imenu=1&(i=6!i=7)&x=crt&k=3 then call screen_clear(continue,1);
            end;
        return;
        end list_data;

/*****************************************************************************
*  M O D I F Y :                                                             *
* This routine controls modification of variables and check for data         *
* consistency:                                                               *
*    - BASE_YEAE<=START_CONSTRUCTION<=END_CONSTRUCTION<=START_DEBT_REPAYMENT *
*    - COMMON_EQUITY_FRACTION + DEBT_FRACTION <= 1                           *
*    - ACRS TAX_LIFE = 5,10,or 15 years                                      *
*    - LEVEL_PERIOD <= YEARS_ANALYSIS.                                       *
*****************************************************************************/
    modify:
        procedure;

        ilist = -1;
```

```
        do while(ilist = 0);
        ilist = -1;
/*****************************************************************************
* Print menu and get option value to call MOD_ALL or MOD_ONE.                *
*****************************************************************************/
            do while(ilist < 0 ! ilist > 2);
            call clearscreen;
            put file(crt) skip(0) edit('Enter 0 - For No More Changes',
                '1 - To Modify All Entries',
                '2 - To Modify Selected Entries') (a,col(7));
            put file(crt) skip edit('? ') (a);
            call get_integer(escape_flag,ilist);
            if ilist < 0 ! ilist > 2 then put file(crt) edit(bell) (a);
            if ilist=1 then call mod_all; else if ilist=2 then call mod_one;
/*****************************************************************************
* Make data consistency checks.                                               *
*****************************************************************************/
            year_last = end_construction;
            fix_value = start_construction;
            fix_value = year_last - fix_value;
            if end_construction < start_construction !
            start_construction < base_year ! fix_value > years_construction
                then do;
                call clearscreen;
                put file(crt) skip(0) edit(bell,'<Warning>') (a,col(34));
                put file(crt) skip edit('Base Year<=Start of Construction',
                '<=End of Construction>Start of Construction+',
                years_construction) (2a,f(3,0));
                put file(crt) skip edit(base_year,start_construction,
                end_construction) (x(1),f(7,2),x(10),f(7,2),x(15),f(7,2));
                put file(crt) skip(2);
                call screen_clear(continue,1);
                ilist = -1;
                end;
            else if common_equity_fraction + debt_fraction > 1
                then do;
                call clearscreen;
                put file(crt) skip(0) edit(bell,'<Warning>') (a,col(20));
                put file(crt) skip edit('Common Equity Fraction Plus',
                ' Debt Fraction > 1.0') (a);
                put file(crt) skip edit(common_equity_fraction,
                debt_fraction) (col(9),f(6,4),col(32),f(6,4));
                call screen_clear(continue,1);
                ilist = -1;
                end;
            else if accelerated_dep_flag=1 & tax_life=5. & tax_life=10.
            & tax_life=15.
                then do;
                call clearscreen;
                put file(crt) skip(0) edit(bell,'<Warning>') (a,col(22));
                put file(crt) skip edit('Allowed Tax Lives:',
                ' 5,10 or 15 only.  Tax Life =',tax_life) (2a,f(6,2));
                call screen clear(continue,1);
```

```
                ilist = -1;
                end;
            else if levelization_period > years_analysis
                then do;
                call clearscreen;
                put file(crt) skip(0) edit(bell,'<Warning>') (a,col(22));
                put file(crt) skip edit('Levelization Period <=',
                years_analysis,' only.  Levelization Period =',
                levelization_period) (a,f(3,0));
                call screen_clear(continue,1);
                ilist = -1;
                end;
            else if year_last = end_construction
                then do;
                call clearscreen;
                put file(crt) skip(0) edit(bell,'<Warning>') (a,col(22));
                put file(crt) skip edit('End Construction = to integral ',
                'year.  End Construction =',end_construction) (2a,f(8,2));
                call screen_clear(continue,i);
                end;
            end;
        end;
    return;
    end modify;

/*****************************************************************************
*  M O D _ A L L :                                                          *
* This routine is the driver for sequential modification of all variables.  *
*****************************************************************************/
  mod_all:
    procedure;

        do imod = 1 to num_types;
        call find_record(type_save,imod,title);
            do jmod=offset_number(imod) to (offset_number(imod+1)-1);
            call find_record(type_save,num_types+jmod,subtitle);
            call mod_data(imod,jmod);
            if escape_flag < 0 then return;
            end;
        end;
    return;
    end mod_all;

/*****************************************************************************
*  M O D _ O N E :                                                          *
* This routine selects the variable or array for modification.              *
* First one of the NUM_TYPES major categories is selected and then          *
* the desired subcategory.                                                  *
*****************************************************************************/
  mod_one:
    procedure;

    imod = -1;
        do while(imod = 0);
        call clearscreen;
```

```
            do i = 1 to num_types;
            call find_record(type_save,i,title);
            put file(crt) edit(i,' - ',title) (col(1),f(2,0),a,a);
            end;
        imod = -1;
/*****************************************************************************
* Select major category (IMOD).                                              *
*****************************************************************************/
            do while(imod < 0 ! imod > num_types);
            put file(crt) skip(2) edit('Enter number corresponding'
                ,' to data to be changed (0 for no more changes)') (a);
            put file(crt) skip edit('? ') (a);
            call get_integer(escape_flag,imod);
            if imod<0 ! imod>num_types then put file(crt) edit(bell) (a);
            end;
        jmod = -1;
        number_entries = offset_number(imod+1) - offset_number(imod);
        call find_record(type_save,imod,title);
/*****************************************************************************
* Select subcategory  (JMOD).                                                *
*****************************************************************************/
            do while(jmod = 0 & imod > 1);
            year_base = offset_number(imod) - 1;
            call clearscreen;
            put file(crt) skip(0) edit(title,':') (a);
                do j = offset_number(imod) to (offset_number(imod+1)-1);
                call find_record(type_save,num_types+j,subtitle);
                put file(crt) skip edit(j-year_base,' - ',subtitle)
                    (col(5),f(2,0),a,a);
                end;
            jmod = -1;
                do while(jmod < 0 ! jmod > number_entries);
                put file(crt) skip(2) edit('Enter the number corresponding'
                ,' to the data to be changed (0 for no more changes)') (a);
                put file(crt) skip edit('? ') (a);
                call get_integer(escape_flag,jmod);
                if jmod < 0 ! jmod > number_entries then put file(crt)
                edit(bell) (a);
                end;
            if jmod =0
                then do;
                jmod = offset_number(imod) + jmod - 1;
                call find_record(type_save,num_types+jmod,subtitle);
/*****************************************************************************
* Modify data (MOD_DATA) for variable category IMOD and variable JMOD.      *
*****************************************************************************/
                call mod_data(imod,jmod);
                end;
            end;
        if imod=1 then call mod_data(imod,1);
        end;
    return;
    end mod_one;
```

```
/*****************************************************************************
*  M O D _ D A T A :                                                         *
* This routine list the current variable or array value(s).  Routine         *
* ONE_ENTRY is used to change scalars and MOD_ARRAY is used for arrays.      *
*   - IMOD is the major category index, and                                  *
*   - JMOD is the variable index.                                            *
*****************************************************************************/
    mod_data:
        procedure(imod,jmod);

        dcl (imod,jmod) fixed;

        kmod = -1;
            do while(kmod = 0);
            call clearscreen;
            put file(crt) skip(0) edit(title) (a);
            put file(crt) skip;
            call list_data(imod,jmod,crt);
            if imod <= num_scalar then call one_entry(imod,jmod,kmod);
            else call mod_array(imod,jmod,kmod);
            end;
        call clearscreen;
        return;
        end mod_data;

/*****************************************************************************
*  O N E _ E N T R Y :                                                       *
* This routine is the driver for modification of scalars.                    *
*   - IMOD is the major category index,                                      *
*   - JMOD is the variable index, and                                        *
*   - KMOD is a flag which is set to 0 upon successful data entry.           *
*****************************************************************************/
    one_entry:
        procedure(imod,jmod,kmod);

        dcl (imod,jmod,kmod) fixed;

        kmod = -1;
        put file(crt) skip(2) edit('Enter 0 - To Retain Current Value',
            '1 - To Modify Current Value') (a,col(7));
        put file(crt) skip edit('? ') (a);
        call get_integer(escape_flag,kmod);
        if kmod < 0 ! kmod > 1 then put file(crt) edit(bell) (a);
        if kmod = 1 then call get_data(imod,jmod);
        return;
        end one_entry;
```

```
/*****************************************************************************
*  M O D _ A R R A Y :                                                       *
* This routine is the driver for modification of arrays.  Two options       *
* are available:                                                            *
*    - Modify a range of years (GET_ARRAY)                                  *
*    - Modify a single year  (GET_YEAR).                                    *
* Routine inputs are:                                                       *
*    - IMOD is the major category index,                                    *
*    - JMOD is the variable index,                                          *
*    - KMOD is a flag set to 0 upon successful data entry.                  *
*****************************************************************************/
    mod_array:
        procedure(imod,jmod,kmod);

        dcl (imod,jmod,kmod) fixed;

        kmod = -1;
        put file(crt) skip(2) edit('Enter 0 - To Retain Current Values',
        '1 - To Modify Range of Years','2 - To Modify Selected Year')
        (a,col(7),a,col(7));
        put file(crt) skip edit('? ') (a);
        call get_integer(escape_flag,kmod);
        if kmod < 0 ! kmod > 2 then put file(crt) edit(bell) (a);
        if kmod = 1 then call get_array(imod,jmod);
        else if kmod = 2 then call get_year(imod,jmod);
        return;
        end mod_array;

/*****************************************************************************
*  G E T _ A R R A Y :                                                       *
* This routine specifies the range of years to be modified.  Two options    *
* are available:                                                            *
*    - Specify an explicit range of years (SET_YEARS)                       *
*    - Modify entire array (SET_RANGE).                                     *
* Routine inputs are:                                                       *
*    - IMOD is the major category index, and                                *
*    - JMOD is the variable index.                                          *
*****************************************************************************/
    get_array:
        procedure(imod,jmod);

        dcl (imod,jmod) fixed;
/*****************************************************************************
* Print menu, get value and call SET_YEARS or SET_RANGE.                    *
*****************************************************************************/
        lmod = -1;
            do while(lmod < 1 ! lmod > 2);
            put file(crt) skip edit('Enter 1 - To Specify Starting and',
            ' Ending Year','2 - To Process Entire Array')
            (a,a,col(7),a);
            put file(crt) skip edit('? ') (a);
            call get_integer(escape_flag,lmod);
            if escape_flag < 0 then return;
            if(lmod < 1 ! lmod > 2) then put file(crt) edit(bell) (a);
            end;
```

```
        call set_range(imod,jmod);
        if lmod = 1
            then do;
            call set_years(imod,jmod);
            if escape_flag = -1 then return;
            end;
        else year = year_base;
/*****************************************************************************
*  Print menu and get value how to enter values.                            *
*****************************************************************************/
        lmod = -1;
            do while(lmod < 1! lmod> 2);
            put file(crt) skip edit('Enter 1 - To Enter Array Values',
            '2 - To Specify ',year,' Value and Escalation Factor',' ? ')
            (a,col(7),a,f(5,0),a,skip,a);
            call get_integer(escape_flag,lmod);
            if escape_flag < 0 then return;
            if(lmod < 1 ! lmod > 2) then put file(crt) edit(bell) (a);
            end;
        call clearscreen;
        put file(crt) skip(0) edit(title) (a);
        put file(crt) skip;
        save_value = number_entries;
        call list_data(imod,jmod,crt);
        number_entries = save_value;
        if lmod = 2 then call fill_data(imod,jmod);
        else call get_data(imod,jmod);
        return;
        end get_array;

/*****************************************************************************
*  G E T _ Y E A R :                                                        *
* This routine gets the array year to be changed and changes value.         *
*   - IMOD is the major category index, and                                 *
*   - JMOD is the variable index.                                           *
*****************************************************************************/
    get_year:
        procedure(imod,jmod);

        dcl (imod,jmod) fixed;

        call set_range(imod,jmod);
        number_entries = 1;
        year = 0;
            do while(year < year_base ! year > year_last);
            put file(crt) skip edit('Enter year to be modified (',
                year_base,',',year_last,') ? ') (a,f(4,0));
            call get_integer(escape_flag,year);
            if escape_flag < 0 then return;
            if year<year_base!year>year_last then put file(crt) edit(bell) (a);
            else call get_data(imod,jmod);
            end;
        return;
        end get_year;
```

```
/*****************************************************************************
*  G E T _ D A T A :                                                         *
*  This routine gets the new value.                                          *
*   - IMOD is the major category index, and                                  *
*   - JMOD is the variable index.                                            *
*****************************************************************************/
    get_data:
        procedure(imod,jmod);

        dcl (imod,jmod) fixed;

        arg_add = address(imod);
        call set_base(imod,jmod,k);
        escape_flag = 1;
            do while(escape_flag > 0);
            put file(crt) skip(2) edit('Enter ',title,':') (a);
            if imod = 1 then put file(crt) skip edit(subtitle) (col(5),a);
            else put file(crt) skip;
            if imod <= num_scalar then put file(crt) edit('? ') (a);
            if imod = 1
                then do;
                call get_string(buffer);
                if buffer = escape
                    then do;
                    header = buffer;
                    escape_flag = 0;
                    end;
                else escape_flag = -1;
                end;
            else if imod <= num_scalar
                then do;
                escape_flag = type_save;
                call get_variable(escape_flag,float_array(k));
                end;
                else do;
                j = 1;
                put file(crt) skip;
                    do i = 1 to number_entries;
                    k = k + 1;
                    escape_flag = 1;
                        do while(escape_flag > 0);
                        escape_flag = type_save;
                        put file(crt) edit(year,': ? ') (f(4,0),a);
                        call get_variable(escape_flag,float_array(k));
                        if escape_flag < 0 then return;
                        end;
                    if j < 5
                        then do;
                        call rightcursor(j*15);
                        call upcursor(1);
                        j = j + 1;
                        end;
                        else do;
                        put file(crt) skip(0);
                        j = 1;
                        end;
```

```
                    year = year + 1;
                    end;
                year = year - 1;
                end;
            end;
        return;
        end get_data;

/*****************************************************************************
*  G E T _ V A R I A B L E :                                                 *
* This routine gets a floating point value and checks that is within a      *
* valid range.  TYPE contains the range specification index upon entry and  *
* the error code upon exit (1 or -1 for error).                             *
*   - TYPE is the range specification and returns error code,               *
*   - VALUE is the returned floating point value if TYPE = 0.               *
*****************************************************************************/
    get_variable:
        procedure(type,value);

        dcl (type_save,type) fixed,
            value float;

        type_save = type;
        call get_float(type,value);
        if type = 0 then return;
        type = type_save;
        if type = 10
            then do;
            nlow = low(5);
            nhigh = high(5);
            end;
            else do;
            nlow = low(type);
            nhigh = high(type);
            end;
        if type = 10
            then do;
            buffer = ',';
            if value >= nlow & value <= nhigh then type = 0;
            end;
            else do;
            buffer = ' or';
            if value = nlow ! value = nhigh then type = 0;
            end;
        if type = 0
            then do;
            put file(crt) skip edit(bell,'Value out of range: (',
            nlow,buffer,nhigh,').  Reenter') (a,a,f(7,2),a,f(5,0));
            put file(crt) skip;
            end;
        return;
        end get_variable;
```

```
/*****************************************************************************
*  S E T _ R A NG E :                                                        *
*  This routine sets YEAR_BASE,YEAR_LAST,NUMBER_ENTRIES,and ARRAY_SIZE.      *
*   - I is the major category index, and                                     *
*   - J is the variable index.                                               *
*****************************************************************************/
    set_range:
        procedure(i,j);

        dcl (i,j) fixed;

        if i = num_types
            then do;
            year_base = start_construction;
            year_last = end_construction;
            number_entries = year_last - year_base;
            year_last = year_last - 1;
            if number_entries > years_construction
            then number_entries = years_construction;
            array_size = years_construction;
            end;
        else if i = num_scalar + 1
            then do;
            year_base = base_year;
            if j=offset_number(7)-1 then number_entries=years_equity;
            else number_entries = years_analysis;
            year_last = year_base + number_entries - 1;
            array_size = years_analysis;
            end;
            else do;
            number_entries = years_analysis;
            array_size = years_analysis;
            year_base = end_construction;
            year_last = year_base + years_analysis - 1;
            end;
        return;
        end set_range;

/*****************************************************************************
*  S E T _ Y E A R S :                                                       *
* This routine sets the range of years to be modified.  YEAR_BASE is set     *
* to the first year and NUMBER_ENTRIES contains the total number of years.   *
*   - IMOD is the major category index, and                                  *
*   - JMOD is the variable index.                                            *
*****************************************************************************/
    set_years:
        procedure(imod,jmod);

        dcl (imod,jmod) fixed;

        first_year = 0;
        last year = 0;
```

```
            do while(first_year < year_base ! last_year > year_last);
            put file(crt) skip edit('Enter First Year (',year_base,',',
            year_last,') ? ') (a,f(5,0));
            call get_integer(escape_flag,first_year);
            if escape_flag < 0 then return;
            if(first_year >= year_base & first_year <= year_last)
                then do;
                put file(crt) edit('Enter Last Year  (',first_year,
                ',',year_last,') ? ') (a,f(5,0));
                call get_integer(escape_flag,last_year);
                if escape_flag < 0 then return;
                end;
            if(first_year<year_base!last_year<first_year!last_year>year_last)
                then do;
                first_year = 0;
                put file(crt) edit(bell) (a);
                end;
            end;
        year = first_year;
        number_entries = last_year - first_year + 1;
        return;
        end set_years;

/*****************************************************************************
*  F I L L _ D A T A :                                                       *
* This routine fills an array when the 'value + escalation' option is        *
* selected.  The routine automatically escalates the first year value (VALUE) *
* by the escalation rate (ESCALATION).                                       *
*   - IMOD is the major category index, and                                  *
*   - JMOD is the variable index.                                            *
*****************************************************************************/
    fill_data:
        procedure(imod,jmod);

        dcl (imod,jmod) fixed;

        lmod = 1;
            do while(lmod > 0);
            lmod=1;
                do while(lmod>0);
                lmod = type_save;
                put file(crt) skip(2) edit('Enter Value For ',year,
                 ':? ') (a,f(5,0));
                call get_variable(lmod,value);
                if lmod = -1 then return;
                end;
            lmod = 1;
                do while(lmod > 0);
                lmod = 6;
                put file(crt) edit('Enter Annual Escalation Value (i.e'
                ,'. 1.10): ? ') (a);
                call get_variable(lmod,escalation);
                if lmod = -1 then return;
                end;
```

```
            temp = value*(escalation**number_entries);
            if temp < low(type_save) ! temp > high(type_save) then lmod=1;
            if lmod=1 then put file(crt) edit(bell) (a);
            end;
        arg_add = address(imod);
        call set_base(imod,jmod,k);
            do i = 1 to number_entries;
            k = k + 1;
            float_array(k) = value;
            value = value*escalation;
            end;
        return;
        end fill_data;

/*****************************************************************************
*  S E T _ B A S E :                                                         *
*  This routine finds the correct array offset (K).                          *
*   - IMOD is the major category index,                                      *
*   - JMOD is the variable index, and                                        *
*   - K is the array offset.                                                 *
*****************************************************************************/
    set_base:
        procedure(imod,jmod,k);

        dcl (imod,jmod,k) fixed;

        k = jmod - offset_number(imod) + 1;
        if imod=num_scalar+1 then i=base_year; else i=end_construction;
        fix_value = start_construction;
        if imod=num_types then k=(k - 1)*years_construction+year-fix_value;
        else if imod>num_scalar then k=(k - 1)*years_analysis+year -i;
        return;
        end set_base;

/*****************************************************************************
*  T E R M I N A T E :                                                       *
*  This routine prints a closing message and terminates run.                 *
*****************************************************************************/
    terminate:
        procedure;

    close file(printer);
    call clearscreen;
    put file(crt)skip edit('User termination of BREEDER from NEW INPUT option')
    (a);
    close file(crt);
    call find_record(type_save,-1,buffer);
    stop;
    end terminate;

    return;
    end input;
```

```
/*****************************************************************************
*  F I L E O U T :                                                          *
* This routine asks whether the current data should be saved and saves      *
* it, if requested, as file LSPBFILE.DAT.                                   *
*                                                                           *
*     Original Version  8/5/83                                              *
*     Version 1.5  4/15/84                                                  *
*        Change clearscreen to subroutine call to provide compatibility     *
*        between CP/M and IBM/PC-DOS versions.                              *
*****************************************************************************/
fileout:
    procedure;

    %include 'input.dcl';
    %include 'diomod.dcl';

    dcl yes_no char(1),
        (crt,out) file;

    yes_no = 'e';
/*****************************************************************************
* Print Menu and determine option.                                          *
*****************************************************************************/
        do while(yes_no = 'e');
        call clearscreen;
        put file(crt) skip edit('Do you wish this data to',
        ' replace the current permanent data file (y/n)? ') (a);
        yes_no = coninp();
        yes_no = substr(yes_no,1,length(yes_no)-2);
        if yes_no= 'Y' then yes_no = 'y';
        else if yes_no = 'N' then yes_no='n';
        if yes_no='y'&yes_no='n'
            then do;
            put file(crt) edit(bell) (a);
            yes_no = 'e';
            end;
        end;
        if yes_no = 'y'
        then do;
/*****************************************************************************
* Write current data file INPUT.DAT to file LSPBFILE.DAT.                   *
*****************************************************************************/
        open file(out) output sequential env(b(1024)) title('LSPBFILE.DAT');
        write file(out) from(input_data);
        close file(out);
        end;
    return;
    end fileout;
```

```
/*****************************************************************************
*  C O N S T R U C T I O N :                                                 *
* This routine calculates construction period expenditures.  The following  *
* financing is included:                                                     *
*     Short-term debt,                                                       *
*     Long-term debt,                                                        *
*     Preferred stock,                                                       *
*     Common Equity.                                                         *
* All calculations are preformed quaterly. However, short-term debt can     *
* be converted to long-term debt or common/preferred equity either on       *
* a quarterly or annual basis (QUARTER_ANNUAL_FLAG).  Long-term debt        *
* interest charges are calculated annualy; short-term debt interest         *
* charges are calculated quarterly.  Either simple or compound interest     *
* charges are permitted (COMPOUND_SIMPLE_FLAG).  For accounting purposes    *
* interest and equity charges are accrued at the end of each quarter and    *
* displayed the following quarter.  A compound inflation factor is cal-     *
* ulated for each quarter.  OVERNIGHT capital costs are inflated to         *
* current-year $ quarterly.  ITC and FIT tax accruals are calculated        *
* annually.  Either a 10% or 8% ITC it permitted (TEFRA_FLAG).              *
*                                                                            *
*     Original Version  8/5/83                                               *
*     Version 1.1  9/15/83                                                   *
*        Increase RATES.EQUITY to YEARS_ANALYSIS years,                      *
*        Permit separate quarterly and/or annual printout summaries,         *
*        Correct several bugs:                                               *
*             - include QUARTER_ANNUAL_FLAG code for final_sums              *
*             - correct final_sums tax_accrual                               *
*             - correct before/after-tax calculation of annual/quarterly     *
*               summaries,                                                   *
*             - Initialize before-tax array to 0.                            *
*     Version 1.2  10/10/83                                                  *
*        Include STARTUP COSTS and WORKING CAPITAL,                          *
*        Ensure that TAXES is called for partial final year                  *
*     Version 1.3  10/20/83                                                  *
*        Update code for START_CONSTRUCTION = 1st quarter:                   *
*           - cum_inflation to starting quarter                              *
*           - quarter_construction prorated over remaining quarters          *
*        Update code for END_CONSTRUCTION = end of fourth quarter            *
*           - quarter_construction prorated over remaining quarters          *
*        Set cwip_fraction and AFUDC_fraction to 0                           *
*       For quarter_annual_flag = 0 option include annual_equity_drawdown    *
*       in before_tax when accrued not when reported.                        *
*       Include construction_real and AFUDC_real calculations.               *
*     Version 1.4  11/10/83                                                  *
*        Include option (NET_EQUITY_FLAG=0) to use variable % of equity      *
*        financing.  Equity financing in a given quarter equals ITC          *
*        credit plus interest tax accrual for the quarter.  No preferred     *
*        financing is allowed.  This option overrides the QUARTER_ANNUAL     *
*        _FLAG option (only quarterly financing is allowed).                 *
*     Version 1.5  4/15/84                                                   *
*        Replace "REAL" with "BASE YEAR" in final summary printout.          *
*        Delete cwip_fraction and AFUDC_fraction variables                   *
*        Make CLEARSCREEN a subroutine call to provide compatability between *
*        CP/M and the IBM/PC-DOS versions.                                   *
*****************************************************************************/
```

```
construction:
    procedure;

    %include 'input.dcl';
    %include 'constr.dcl';
    %include 'diomod.dcl';

    dcl (printer,crt) file,
        (i,j,k,l,istart,iend,idelta,number,num_sav,iyear,jyear,constr_year,
        quarter_annual_flag_save,annual_year,buf_length) fixed,
        (debt_LT,quarter_construction,quarter_STinterest,quarter_inflation,
        cum_inflation,cum_LTdebt,cum_equity,cum_preferred,cum_STdebt,
        totals(9),rate_itc,quarter_equity,quarter_preferred,denom,temp,
        last_year_factor) float,
        dummy(1) float based(address),
        address pointer,
        get_integer entry(fixed,fixed),
        find_record entry(fixed,fixed,char(79) var),
        screen_clear entry(char(1),fixed),
        center entry(file,fixed,fixed,char(79) var),
        buffer char(79) var,
        continue char(1);

    dcl 1 quarter(quarters_construction_plus_1),
        2 overnight_construction float,
        2 inflation_impact float,
        2 current_construction float,
        2 STdebt_interest float,
        2 LTdebt_interest float,
        2 preferred_charge float,
        2 equity_charge float,
        2 total_interest_charges float,
        2 total_construction_expense float,
        2 STdebt_drawdown float,
        2 LTdebt_drawdown float,
        2 preferred_drawdown float,
        2 equity_drawdown float,
        2 total_CWIP float,
        2 total_AFUDC float,
        2 cum_CWIP float,
        2 cum_AFUDC float,
        2 cum_construction_expense float;

    dcl 1 annual(years_construction_plus_1),
        2 annual_overnight_construction float,
        2 annual_inflation_impact float,
        2 annual_current_construction float,
        2 annual_STdebt_interest float,
        2 annual_LTdebt_interest float,
        2 annual_preferred_charge float,
        2 annual_equity_charge float,
        2 annual_total_interest_charges float,
        2 annual_total_construction_expen float,
        2 annual STdebt drawdown float,
```

```
        2 annual_LTdebt_drawdown float,
        2 annual_preferred_drawdown float,
        2 annual_equity_drawdown float,
        2 annual_CWIP float,
        2 annual_AFUDC float,
        2 annual_cum_CWIP float,
        2 annual_cum_AFUDC float,
        2 annual_cum_construction_expense float;

/*****************************************************************************
* Initialize variables for CONSTRUCTION period analysis.                     *
*****************************************************************************/
    call initial;
/*****************************************************************************
* Calculate variables for CONSTRUCTION period analysis.                      *
*****************************************************************************/
    call calculate;
/*****************************************************************************
* Print reports for CONSTRUCTION period analysis.                            *
*****************************************************************************/
    call report;
    quarter_annual_flag = quarter_annual_flag_save;
    return;
/*****************************************************************************
*  I N I T I A L :                                                           *
*  This routine initializes variables for CONSTRUCTION period analysis.      *
*   - Zero variables,                                                        *
*   - Calculate cumulative inflation factor; and                             *
*   - Calculate 1st quarter interest and equity rates.                       *
*****************************************************************************/
    initial:
        procedure;

        call clearscreen;
        put file(crt) skip(0) edit('Beginning CONSTRUCTION Analysis') (a);
        buffer = 'constr';
        call find_record(buf_length,0,buffer);
/*****************************************************************************
*  If TEFRA_FLAG = 0 then use TEFRA provision for reduced ITC rate and full  *
* depreciation basis; otherwise use full ITC rate and reduce depreciation base*
*****************************************************************************/
        if tefra_flag = 0 then rate_itc = itc_rate - 0.02;
        else rate_itc = itc_rate;
/*****************************************************************************
* Save QUARTER_ANNUAL_FLAG. Net equity option (NET_EQUITY_FLAG = 0) overrides *
* QUARTER_ANNUAL_FLAG (requires QUARTER_ANNUAL_FLAG = 0).                    *
*****************************************************************************/
        quarter_annual_flag_save = quarter_annual_flag;
        if net_equity_flag = 0 then quarter_annual_flag = 0;
        construction = 0;
        AFUDC = 0.;
        construction_real = 0.;
        AFUDC_real = 0.;
        LT debt rate = 0.;
```

```
        preferred_rate = 0.;
        cum_STdebt = 0.;
        cum_LTdebt = 0.;
        cum_equity = 0.;
        cum_preferred = 0.;
        ITC_accrual = 0.;
        tax_accrual = 0.;
        cum_inflation = 1.;
        last_year_factor = 1.;
        istart = start_construction;
        number = istart - base_year;
        constr_year = 0;
        iyear = 1;
        jyear = 1;
        annual_year = 1;
            do i = 1 to years_construction;
            interest_credit(i) = 0;
            end;
            do i = 1 to number;
            cum_inflation = cum_inflation*(1. + inflation(iyear));
            if iyear < years_analysis then iyear = iyear + 1;
            if jyear < years_analysis then jyear = jyear + 1;
            end;
        call set_rates;
        iend = end_construction;
        number = 4*(iend - istart);
        istart = (start_construction - istart)*4.;
            do i = 0 to istart;
            cum_inflation = cum_inflation*quarter_inflation;
            end;
/*****************************************************************************
*  Big loop to zero QUARTER array.                                          *
*****************************************************************************/
        istart = istart + 1;
        address = addr(quarter);
            do i = 1 to quarters_construction_plus_1*18;
            dummy(i) = 0.;
            end;
/*****************************************************************************
*  Big loop to zero ANNUAL array.                                           *
*****************************************************************************/
        address = addr(annual);
            do i = 1 to years_construction_plus_1*18;
            dummy(i) = 0.;
            end;
        return;
        end initial;

/*****************************************************************************
*  S E T _ R A T E S :                                                      *
*  This routine sets the quarter inflation,STinterest,equity and preferred  *
* rates for year iyear/jyear.                                               *
*****************************************************************************/
    set_rates:
        procedure;
```

```
        constr_year = constr_year + 1;
        quarter_inflation = (1. + inflation(iyear))**.25;
        quarter_construction = plant(constr_year)*.25;
        if iyear=1 then quarter_construction=quarter_construction*4/(5-istart);
        if compound_simple_flag = 0
            then do;
            quarter_equity = (1. + rates.common_equity(jyear))**.25-1.;
            quarter_preferred = (1. + rates.preferred_stock(iyear))**.25-1.;
            quarter_STinterest = (1. + ST_debt(iyear))**.25 -1.;
            end;
            else do;
            quarter_equity = rates.common_equity(jyear)*.25;
            quarter_preferred = rates.preferred_stock(iyear)*.25;
            quarter_STinterest = ST_debt(iyear)*.25;
            end;
        return;
        end set_rates;

/*****************************************************************************
*  C A L C U L A T E :                                                       *
*  This routine is the driver for the CONSTURCTION period analysis.          *
*****************************************************************************/
    calculate:
        procedure;

            do i = istart to number;
/*****************************************************************************
*  Calculate interest and equity charges based upon previous quarter values. *
*****************************************************************************/
            if i > istart then call interest;
/*****************************************************************************
*  Calculate direct-construction expenses.                                   *
*****************************************************************************/
            call construction_expenses;
/*****************************************************************************
*  Capitalize accrued construction expenses (direct + financing).            *
*****************************************************************************/
            call finance;
            end;
/*****************************************************************************
*  Capitalize final accruals and calculate totals.                           *
*****************************************************************************/
        call final_sums;
        return;
        end calculate;

/*****************************************************************************
*  I N T E R E S T :                                                         *
*  This routine calculates interest and equity charges based upon previous   *
* year's accruals.  Updates pointers at start of new year (NEW_YEAR).        *
*****************************************************************************/
    interest:
        procedure;
```

```
        STdebt_interest(i) = cum_STdebt*quarter_STinterest;
        interest_credit(annual_year) = interest_credit(annual_year) +
        STdebt_interest(i)*FIT_tax;
        equity_charge(i) = (cum_equity - equity_drawdown(i))*quarter_equity;
        preferred_charge(i) = preferred_charge(i-1)
        + preferred_drawdown(i-1)*quarter_preferred;
/*****************************************************************************
*  Update year values (NEW_YEAR) at start of each year.                      *
*****************************************************************************/
        if ((i - 1)/4)*4 + 1 = i then call new_year;
        else annual_STdebt_interest(annual_year) =
        annual_STdebt_interest(annual_year)
        +STdebt_interest(i);
        total_interest_charges(i) = total_interest_charges(i)+
        STdebt_interest(i) + equity_charge(i) + preferred_charge(i);
        total_AFUDC(i) = total_interest_charges(i);
        cum_AFUDC(i) = cum_AFUDC(i-1) + total_AFUDC(i);
        annual_total_interest_charges(annual_year) =
        annual_total_interest_charges(annual_year) + total_interest_charges(i);
        annual_AFUDC(annual_year) = annual_AFUDC(annual_year) + total_AFUDC(i);
        annual_cum_AFUDC(annual_year) = cum_AFUDC(i);
        annual_equity_charge(annual_year) = annual_equity_charge(annual_year)
        + equity_charge(i);
        annual_preferred_charge(annual_year) =
        annual_preferred_charge(annual_year) + preferred_charge(i);
        cum_inflation = cum_inflation*quarter_inflation;
        AFUDC_real = AFUDC_real + total_interest_charges(i)/cum_inflation;
        return;
        end interest;

/*****************************************************************************
*  N E W _ Y E A R :                                                         *
*  This routine updates year-end values and is called for the 1st quarter    *
* of each year.  For QUARTER_ANNUAL_FLAG = 1, ST debt is capitalized during  *
* the first quarter of each year.                                            *
*****************************************************************************/
    new_year:
        procedure;

            cum_STdebt = 0.;
            LTdebt_interest(i) = LTdebt_drawdown(i-4)*rates.LT_debt(iyear-1)
            + LTdebt_interest(i-4);
            interest_credit(annual_year) = interest_credit(annual_year) +
            LTdebt_interest(i)*FIT_tax;
            if net_equity_flag = 1 then after_tax(annual_year) =
            after_tax(annual_year) + interest_credit(annual_year);
            tax_accrual = tax_accrual + interest_credit(annual_year);
            total_interest_charges(i) = LTdebt_interest(i);
            annual_year = annual_year + 1;
            annual_STdebt_interest(annual_year) = STdebt_interest(i);
            annual_LTdebt_interest(annual_year) = LTdebt_interest(i);
            call set_rates;
        return;
        end new_year;
```

```
/*****************************************************************************
*  C O N S T R U C T I O N _ E X P E N S E S :                              *
*  This routine calculates direct construction costs including inflation.   *
*****************************************************************************/
    construction_expenses:
        procedure;

        overnight_construction(i) = quarter_construction;
        inflation_impact(i) = quarter_construction*(cum_inflation-1.);
        current_construction(i) = quarter_construction*cum_inflation;
        total_CWIP(i) = current_construction(i);
        if i=1 then cum_CWIP(i)=total_CWIP(i);
        else cum_CWIP(i)=cum_CWIP(i-1)+total_CWIP(i);
        construction_real = construction_real + quarter_construction;
        total_construction_expense(i) = total_interest_charges(i) +
        current_construction(i);
        cum_construction_expense(i) = total_construction_expense(i);
        if i>1 then cum_construction_expense(i) = cum_construction_expense(i) +
        cum_construction_expense(i-1);
        annual_overnight_construction(annual_year) = quarter_construction +
        annual_overnight_construction(annual_year);
        annual_inflation_impact(annual_year) = inflation_impact(i) +
        annual_inflation_impact(annual_year);
        annual_current_construction(annual_year) = current_construction(i)
        + annual_current_construction(annual_year);
        annual_CWIP(annual_year) = annual_CWIP(annual_year) + total_CWIP(i);
        annual_cum_construction_expense(annual_year) =
        cum_construction_expense(i);
        annual_cum_CWIP(annual_year) = cum_CWIP(i);
        annual_total_construction_expen(annual_year) =
        total_construction_expense(i) +
        annual_total_construction_expen(annual_year);
        return;
        end construction_expenses;

/*****************************************************************************
*  F I N A N C E :                                                          *
*  This routine capitalizes short-term debt.  Capitalization is performed   *
* quarterly or annually depending upon QUARTER_ANNUAL_FLAG.  Long-term      *
* debt is issued annually.  The routine also calls END_YEAR at the en of each *
* year.                                                                     *
*****************************************************************************/
    finance:
        procedure;

        denom = total_construction_expense(i);
        if quarter_annual_flag = 0
            then do;
/*****************************************************************************
*  ST debt capitalized quarterly (EQUITY and PREFERRED only); update values. *
*****************************************************************************/
            if net_equity_flag = 1
                then do;
/*****************************************************************************
*  Normal EQUITY financing option; update EQUITY and PREFERRED values.      *
*****************************************************************************/
```

```
                denom = denom*debt_fraction;
                equity_drawdown(i) = common_equity_fraction*
                total_construction_expense(i);
                preferred_drawdown(i)=(1.-debt_fraction-common_equity_fraction)
                *total_construction_expense(i);
                end;
                else do;
/*****************************************************************************
*  Special NET_EQUITY option.  Equity is set equal to quarterly tax (ITC/IDC) *
* benefits.  No PREFERRED financing allowed.                                  *
*****************************************************************************/
                denom = total_construction_expense(i) - FIT_tax*
                quarter_STinterest*cum_STdebt - current_construction(i)*
                rate_ITC*last_year_factor;
                if i > 3 then denom = denom - (LTdebt_drawdown(i-3)*
                rates.LT_debt(iyear) + LTdebt_interest(i-3))*FIT_tax;
                denom = denom/(1. + FIT_tax*quarter_STinterest);
                equity_drawdown(i) = total_construction_expense(i) - denom;
                preferred_drawdown(i) = 0;
                end;
            cum_equity = cum_equity + equity_drawdown(i);
            temp = cum_preferred + preferred_drawdown(i);
            if temp = 0 then preferred_rate = (cum_preferred*preferred_rate
            + preferred_drawdown(i)*rates.preferred_stock(iyear))/temp;
            cum_preferred = temp;
            end;
        cum_STdebt = cum_STdebt + denom;
        STdebt_drawdown(i) = STdebt_drawdown(i) + denom;
        annual_STdebt_drawdown(annual_year)=annual_STdebt_drawdown(annual_year)
        + STdebt_drawdown(i);
        annual_preferred_drawdown(annual_year) = preferred_drawdown(i)
        + annual_preferred_drawdown(annual_year);
        annual_equity_drawdown(annual_year) = equity_drawdown(i) +
        annual_equity_drawdown(annual_year);
        if (i/4)*4 = i then call end_year;
        return;
        end finance;

/*****************************************************************************
*  E N D _ Y E A R :                                                          *
*  This routine updates LTdebt at the end of each year.  It also calculates   *
* capitalization of debt for QUARTER_ANNUAL_FLAG = 1.                        *
*****************************************************************************/
    end_year:
        procedure;

            if iyear < years_analysis then iyear = iyear + 1;
            if jyear < years_analysis then jyear = jyear + 1;
/*****************************************************************************
* LT debt is issued annually independent of QUARTER_ANNUAL_FLAG specification.*
*****************************************************************************/
            if quarter_annual_flag = 0 then debt_LT = cum_STdebt;
            else debt_LT = cum_STdebt*debt_fraction;
            LTdebt drawdown(i+1) = debt LT;
```

```
            denom = cum_LTdebt + debt_LT;
            if denom = 0 then LT_debt_rate = (LT_debt_rate*cum_LTdebt
            + debt_LT*rates.LT_debt(iyear))/denom;
            cum_LTdebt = cum_LTdebt + debt_LT;
            annual_LTdebt_drawdown(annual_year+1) = debt_LT;
            STdebt_drawdown(i+1) = - cum_STdebt;
            if quarter_annual_flag = 1
                then do;
/*****************************************************************************
*  ST debt capitalized annually; update EQUITY and PREFERRED.                *
*****************************************************************************/
                equity_drawdown(i+1) = cum_STdebt*common_equity_fraction;
                before_tax(annual_year) = -equity_drawdown(i+1);
                cum_equity = cum_equity + equity_drawdown(i+1);
                preferred_drawdown(i+1) = (1. - debt_fraction -
                common_equity_fraction)*cum_STdebt;
                denom = cum_preferred + preferred_drawdown(i+1);
                if denom = 0 then preferred_rate = (cum_preferred*
                preferred_rate + preferred_drawdown(i+1)*
                rates.preferred_stock(iyear))/denom;
                cum_preferred = denom;
                end;
        call taxes;
        return;
        end end_year;

/*****************************************************************************
*  T A X E S :                                                               *
*  Taxes calculates tax effects at the end of each year.                     *
*****************************************************************************/
    taxes:
        procedure;

        ITC_credit(annual_year) = annual_current_construction(annual_year)
        *rate_ITC*last_year_factor;
        ITC_accrual = ITC_accrual + ITC_credit(annual_year);
        if quarter_annual_flag = 0 then before_tax(annual_year) =
        - annual_equity_drawdown(annual_year);
        if net_equity_flag = 0 then after_tax(annual_year) = 0;
            else do;
            after_tax(annual_year) = before_tax(annual_year) +
            ITC_credit(annual_year);
            end;
        if i = number
            then do;
            quarter_construction = annual_current_construction(annual_year)/
            annual_overnight_construction(annual_year);
            startup_cost = startup_cost*quarter_construction;
            plant_variables.working_capital = plant_variables.working_capital
            *quarter_construction;
            last_year_factor=1.-(startup_cost+plant_variables.working_capital)
            /quarter_construction;
            end;
        construction = construction + annual_current_construction(annual_year);
        AFUDC = AFUDC + annual_total_interest_charges(annual_year);
        return;
        end;
```

```
/*****************************************************************************
*  F I N A L _ S U M S :                                                    *
*  This routine calculates final accruals and saves construction totals.    *
*****************************************************************************/
    final_sums:
        procedure;

        i = number + 1;
        call interest;
        quarter_construction = 0.;
        call construction_expenses;
        denom = total_construction_expense(i);
        annual_STdebt_drawdown(annual_year) = STdebt_drawdown(i);
        if net_equity_flag = 1
/*****************************************************************************
*  Finance final year accruals among debt,common equity and preferred.      *
*****************************************************************************/
            then do;
            annual_equity_drawdown(annual_year)=denom*common_equity_fraction;
            annual_preferred_drawdown(annual_year) = denom*
            (1. - common_equity_fraction - debt_fraction);
            annual_LTdebt_drawdown(annual_year) = denom*debt_fraction;
            before_tax(annual_year) = - annual_equity_drawdown(annual_year);
            denom = cum_preferred + annual_preferred_drawdown(annual_year);
            if denom = 0 then preferred_rate = (cum_preferred*preferred_rate +
            annual_preferred_drawdown(annual_year)*
            rates.preferred_stock(iyear))/denom;
            cum_preferred = denom;
            annual_preferred_drawdown(annual_year) = preferred_drawdown(i) +
            annual_preferred_drawdown(annual_year);
            end;
/*****************************************************************************
*  NET EQUITY option.  All equity tax credits claimed in previous period.   *
*****************************************************************************/
            else annual_LTdebt_drawdown(annual_year) = denom;
        cum_equity = cum_equity + annual_equity_drawdown(annual_year);
/*****************************************************************************
*  Update LT debt and preferred imbedded rates.                              *
*****************************************************************************/
        denom = cum_LTdebt + annual_LTdebt_drawdown(annual_year);
        if denom = 0 then LT_debt_rate = (LT_debt_rate*cum_LTdebt +
        annual_LTdebt_drawdown(annual_year)*rates.LT_debt(iyear))/denom;
        cum_LTdebt = denom;
        annual_LTdebt_drawdown(annual_year)=annual_LTdebt_drawdown(annual_year)
        + LTdebt_drawdown(i);
        annual_equity_drawdown(annual_year) = equity_drawdown(i) +
        annual_equity_drawdown(annual_year);
        call taxes;
        before_tax(annual_year-1) =
        before_tax(annual_year-1) + before_tax(annual_year);
        after_tax(annual_year-1) = after_tax(annual_year-1) +
        after_tax(annual_year);
        interest_credit(annual_year-1) = interest_credit(annual_year-1) +
        interest_credit(annual_year);
```

```
        cum_STdebt = 0;
        plant_totals.common_equity = cum_equity;
        plant_totals.LT_debt = cum_LTdebt;
        plant_totals.preferred_stock = cum_preferred;
        return;
        end final_sums;

/*****************************************************************************
*  R E P O R T :                                                            *
*  Construction analysis is complete; display REPORT menu and get option.   *
*****************************************************************************/
    report:
        procedure;

        put file(crt) skip(2) edit('CONSTRUCTION analysis complete.') (a);
        put file(crt) skip(2) edit('You may now obtain a summary of the ',
        'the constuction profile.') (a);
        put file(crt) skip(2);
        istart = - 1;
            do while(istart = 0);
            istart = -1;
                do while(istart<0 ! istart>2);
                put file(crt) skip edit('Enter 0 - No Listing',
                '1 - Preview on Screen','2 - List on Printer') (a,col(7));
                put file(crt) skip edit('? ') (a);
                call get_integer(j,istart);
                if istart<0!istart>2 then put file(crt) edit(bell) (a);
                    else do;
                    iend = -1;
                        do while(iend = 0 & istart = 0);
                        iend = -1;
                            do while((iend<0 ! iend>3) & istart=0);
                            call clearscreen;
                            put file(crt) edit('Enter 0 - To Exit',
                            '1 - To print quarterly report',
                            '2 - To print annual report',
                            '3 - To print both reports') (a,col(7));
                            put file(crt) skip edit('? ') (a);
                            call get_integer(i,iend);
                            if iend<0!iend>3 then put file(crt) edit(bell) (a);
                            else if iend=0 then if istart=1
                            then call print_construction(crt);
                            else if istart = 2
                                then do;
                                call print_construction(printer);
                                put file(printer) skip edit('        ') (a);
                                end;
                            call clearscreen;
                            end;
                        end;
                    end;
                end;
            end;
        buffer = 'constr';
        call find_record(buf_length,-1,buffer);
        return;
        end report;
```

```
        call print_header(x,0,13);
        put file(x) skip edit(ITC_accrual,tax_accrual,construction,
        construction_real,AFUDC,AFUDC_real,preferred_rate,LT_debt_rate)
        (f(7,1),f(11,1),2f(10,1),f(9,1),f(10,1),2f(11,5));
        put file(x) skip edit(('_' do j=1 to 79)) (a);
        if x = crt then call screen_clear(continue,1);
        num_sav = num_sav*4;
        end;
    return;
    end print_construction;

/*****************************************************************************
*  P R I N T _ Q U A R T E R :                                              *
*  This routine prints or displays the quarterly summary table.             *
*    - x : output file (CRT or PRINTER).                                    *
*    - offset : offset in array QUARTER. (1 for EXPENSES, 10 for FINANCING). *
*    - num_rec : record number in file CONSTR.DAT of first line of header.  *
*****************************************************************************/
  print_quarter:
    procedure(x,offset,num_rec);

    dcl x file,
    (offset,num_rec) fixed;

    iyear = start_construction;
    idelta = 1;
        do i = 1 to num_sav;
        if x = crt & ((i-1)/16)*16 + 1 = i
            then do;
            if i > 1
                then do;
                call screen_clear(continue,1);
                if continue = escape then return;
                end;
            else call clearscreen;
            call print_header(x,1,num_rec);
            end;
        else if x = printer & ((i-1)/40)*40 + 1 = i
            then do;
            if i > 1 then put file(x) skip edit(('_' do j = 1 to 79)) (a);
            put file(x) skip edit(new_page) (a);
            call print_header(x,3,num_rec);
            end;
        if idelta = 1 then put file(x) skip edit(iyear) (f(4,0));
        address = addr(quarter(i));
        put file(x) skip edit(' Qrt.',idelta,(dummy(j) do j = offset
        to (offset + 8))) (a,f(2,0),9f(8,1));
        if idelta < 4 then idelta = idelta + 1;
            else do;
            idelta = 1;
            iyear = iyear + 1;
            end;
        end;
    if x=printer then put file(x) skip edit(('_' do j = 1 to 79)) (a);
    return;
    end print_quarter;
```

```
/*****************************************************************************
*  P R I N T _ A N N U A L :                                                 *
*  This routine prints or displays the annual summary table.                 *
*    - x : output file (CRT or PRINTER).                                     *
*    - offset : offset in array ANNUAL (1 for EXPENSES or 10 for FINANCING).  *
*    - num_rec : record number in file CONSTR.DAT for first line of header.   *
*****************************************************************************/
    print_annual:
        procedure(x,offset,num_rec);

        dcl x file,
            (offset,num_rec) fixed;

        iyear = start_construction;
        if x = crt then call print_header(x,0,num_rec);
        else if offset = 1 then call print_header(x,1,num_rec);
            else do;
            i = 17 - num_sav;
                do j = 1 to i;
                put file(x) skip;
                end;
            call print_header(x,0,num_rec);
            end;
            do i = 1 to num_sav;
            address = addr(annual(i));
            put file(x) skip edit(iyear,(dummy(j) do j = offset to
            offset + 8)) (f(5,0),x(1),9f(8,1));
            iyear = iyear + 1;
            end;
        address = addr(annual(num_sav+1));
        put file(x) skip edit('Accru:',(dummy(j) do j = offset to offset+8))
        (a,9f(8,1));
            do i = 1 to 9;
            totals(i) = 0.;
            end;
            do i = 1 to num_sav+1;
            k = 1;
            address = addr(annual(i));
                do j = offset to (offset + 8);
                if j<16 then totals(k) = totals(k) + dummy(j);
                else if i = num_sav+1 then totals(k) = dummy(j);
                k = k + 1;
                end;
            end;
        put file(x) skip edit('Total:',(totals(j) do j=1 to 9)) (a,9f(8,1));
        return;
        end print_annual;

/*****************************************************************************
*  P R I N T _ H E A D E R :                                                 *
*  This routine prints a header for each table.  The header labels are taken *
* from the keyed data file CONSTR.DAT.                                       *
*   - x : output file (CRT or PRINTER),                                      *
*   - num_skip : number of lines to be skipped prior to printing table,      *
*   - num_rec : record number in CONSTR.DAT of first hearder line.           *
*****************************************************************************/
```

```
/*****************************************************************************
*  P R I N T _ C O N S T R U C T I O N :                                     *
*  This routine prints or displays construction analysis summary tables.     *
*    - x : output file (CRT or PRINTER).                                     *
*****************************************************************************/
    print_construction:
        procedure(x);

        dcl x file;

        continue = ' ';
        num_sav = number;
        num_sav = ((num_sav+3)/4)*4;
        if iend = 2
            then do;
            call print_quarter(x,1,1);
            if continue = escape then return;
            if x = crt
                then do;
                call screen_clear(continue,1);
                if continue = escape then return;
                end;
            call print_quarter(x,10,4);
            if continue = escape then return;
            if x = crt
                then do;
                call screen_clear(continue,1);
                if continue = escape then return;
                end;
            end;
        if iend = 1
            then do;
            if x=crt then call clearscreen;
            else put file(x) skip edit(new_page) (a);
            num_sav = num_sav/4;
            call print_annual(x,1,7);
            put file(x) skip edit(('_' do j=1 to 79)) (a);
            if x = crt
                then do;
                call screen_clear(continue,1);
                if continue = escape then return;
                end;
            call print_annual(x,10,10);
            put file(x) skip edit(('_' do j=1 to 79)) (a);
            if x = crt
                then do;
                call screen_clear(continue,1);
                if continue = escape then return;
                end;
            if x = printer
                then do;
                i = 17 - num_sav;
                    do j = 1 to i;
                    put file(x) skip;
                    end;
                end;
```

```
print_header:
    procedure(x,num_skip,num_rec);

    dcl x file,
        (num_skip,num_rec) fixed;

    if x = printer & num_skip = 0
        then do;
        buf_length = length(header);
        call center(x,buf_length,num_skip,header);
        put file(x) skip(2);
        end;
        do l = num_rec to (num_rec + 2);
        call find_record(buf_length,l,buffer);
        call center(x,buf_length,l,buffer);
        if (x = printer ! num_skip = 0) & l=num_rec
        then put file(x) skip edit(('_' do j=1 to 79)) (a);
        end;
    if x = printer ! num_skip = 0 then
    put file(x) skip edit(('_' do k = 1 to 79)) (a);
    return;
    end print_header;

end construction;
```

```
/*****************************************************************************
*  C A S H F L O W :                                                         *
* This routine calculates cashflow variables during the operating phase.     *
* Three sets of books are maintained for all calculations:                   *
*     BOOK,                                                                  *
*     REGULATED,                                                             *
*     PARTNER.                                                               *
* Revenue requirements based upon required return-on-equity, operating       *
* costs,interest charges, book depreciation, property taxes, and book        *
* FIT are calculated.  All variables required for the following reports      *
* are calculated on an annual basis (maximum of YEARS_ANALYSIS years).       *
*     BALANCE SHEET                                                          *
*     INCOME STATEMENT                                                       *
*     SOURCES and USES of FUNDS STATEMENT                                    *
*     RETAINED EARNINGS STATEMENT                                            *
*     TAX DEPRECIATION WORKSHEET                                             *
*     BOOK TAX CALCULATION                                                   *
*     REGULATED TAX CALCULATION                                              *
*     PARTNERSHIP RETURN                                                     *
*     PARTNERSHIP CASHFLOW ANALYSIS                                          *
* Either STRAIGHTLINE or accelerated depreciation under TEFRA is allowed     *
* (ACCELERATED_DEP_FLAG).  For tax purposes only direct construction         *
* expenses are allowed and either a FULL or 95% basis (TEFRA_FLAG) is        *
* permitted.  Property tax is calculated either on the GROSS or NET          *
* asset basis (GROSS_NET_FLAG).  All fuel related expenses are expensed      *
* rather than capitalized.  For the regulated tax calculation, FIT is        *
* constrained to be non-negative (book or partnership FIT can be negative)   *
* for UNCONSOLIDATED returns (CONSOLIDATED_TAX_FLAG).  Deferred tax benefits *
* can either be flowed through to the stockholders (not allowed under TEFRA) *
* or normalized (FLOWTHROUGH_NORMALIZE_FLAG).                                *
*                                                                            *
*     Original Version  8/5/83                                               *
*     Version 1.1  9/15/83                                                   *
*          Eliminate debug print outs from version 1.0                       *
*     Version 1.2  10/10/83                                                  *
*          Add TOP-DOWN analysis capability (calculate return-on-equity      *
*          given BUSBAR costs).                                              *
*          Include code for calculating                                      *
*               - DOE loans                                                  *
*               - Partner loans                                              *
*               - Deferred FIT                                               *
*               - Preferred Stock                                            *
*               - Startup Costs                                              *
*               - Working Capital                                            *
*               - Federal Subsidies                                          *
*     Version 1.3  10/20/83                                                  *
*          Include deferred_FIT = - deferred_FIT_taken                       *
*          Add deferred_FIT to total_internal                                *
*          Include delta_working_capital in loans_required calculation       *
*          Change i < tax_life to i <= tax_life in ACRS tax depreciation     *
*          Change i < 5 to i <= 5 for startup_cost depreciation              *
*     Version 1.4  11/10/83                                                  *
*          Include Ad Valorem Tax                                            *
*          Limit cashflow analysis period to MIN(BOOK LIFE,30)               *
```

```
*     Version 1.5  4/15/84                                                    *
*          Include new program options:                                       *
*               - Flowthrough or Normalized treatments                        *
*               - Consolidated or Unconsolidated tax treatments               *
*          Previous program versions only permited flowthrough and            *
*          unconsolidated tax treatmenst.                                     *
*****************************************************************************/
cashflow:
    procedure;

    %include 'input.dcl';
    %include 'constr.dcl';
    %include 'output.dcl';
    %include 'diomod.dcl';

    dcl crt file,
        get_integer entry(fixed,fixed) external,
        (i,both,jyear,iyear,nyear) fixed,
        (cum_inflation,mid_inflation,kWh_factor,fuel_factor,repay_LT_debt,
        loans_required,O_and_M_factor,prop_tax_base,
        ITC_remaining,total_LT_debt,preferred_stock_equity,cum_wk_cap,
        AFUDC_net,total_plant_cost,cum_inflation_save,reduction,tax_base,
        factor,factor0,deferred_reserve) float;

    call initial;
    call calculate;
    call done;
    return;

/*****************************************************************************
*  I N I T I A L :                                                           *
*  This routine initializes the variables required for the CASHFLOW analysis. *
*****************************************************************************/
    initial:
        procedure;

        analysis_period = 0;
        nyear = end_construction;
        iyear = min(book_life,years_analysis);
/*****************************************************************************
* Print request for ANALYSIS PERIOD and get value<=(BOOK LIFE,YEARS_ANALYSIS).*
*****************************************************************************/
            do while(analysis_period < 1 ! analysis_period > iyear);
            put skip file(crt) edit('Analysis period starts in year: ',
            nyear,'Enter number of years to be analyzed (1,',iyear,'): ')
            (a,f(5,0),skip,a,f(2,0));
            call get_integer(i,analysis_period);
            if analysis_period < 1 ! analysis_period > iyear then put
            file(crt) edit(bell) (a);
            end;
        put file(crt) skip(2) edit('Beginning CASHFLOW analysis') (a);
        construction = construction - plant_variables.working_capital;
        prop_tax_base = construction + AFUDC;
        total_plant_cost = prop_tax_base;
```

```
        book_dep_plant = construction/book_life;
        accum_dep(1) = book_dep_plant;
        AFUDC_dep_plant = AFUDC;
        AFUDC_net = AFUDC_dep_plant;
        AFUDC_dep_plant = AFUDC_dep_plant/book_life;
        accum_AFUDC_dep(1) = AFUDC_dep_plant;
        deferred_reserve = 0.;
        cum_deferred_FIT(1) = 0.;
        cum_partner_loans(1) = 0.;
        book_depreciation = book_dep_plant + AFUDC_dep_plant;
        beginning_common_equity(1) = plant_totals.common_equity;
        LT_debt_repayment(1) = plant_totals.LT_debt/debt_period;
        total_LT_debt = plant_totals.LT_debt;
        amort_ITC_book = ITC_accrual/book_life;
        preferred_stock_equity = plant_totals.preferred_stock;
        cum_to_partners(1)=ITC_accrual+tax_accrual-beginning_common_equity(1);
        cum_wk_cap = plant_variables.working_capital;
        kWh_factor = 8.760e-02*size;
        fuel_factor = 8.76e-06*size*heat_rate;
        O_and_M_factor = 8.76e-03*size;
        tax_base = construction - startup_cost;
        if tefra_flag = 0 then tax_base = tax_base*0.95;
        if accelerated_dep_flag = 0 then other_dep = tax_base/tax_life;
        else other_dep = 0.;
/*****************************************************************************
*  Zero tax depreciation arrays.                                             *
*****************************************************************************/
            do i = 1 to analysis_period;
            preferred_stock_repayment(i)=plant_totals.preferred_stock/book_life;
            startup_dep(i) = 0.;
            five_year_property(i) = 0.;
            ten_year_property(i) = 0.;
            fifteen_year_property(i) = 0.;
            end;
        nyear = end_construction - base_year;
        iyear = 1;
/*****************************************************************************
*  Calculate cumulative inflation from BASE YEAR to start of analysis.       *
*****************************************************************************/
        cum_inflation = 1.;
            do i = 1 to nyear;
            cum_inflation = cum_inflation*(1. + inflation(iyear));
            if iyear < years_analysis then iyear = iyear + 1;
            end;
        cum_inflation_save = cum_inflation;
        mid_inflation = cum_inflation;
        ITC_remaining = ITC_accrual;
        jyear = nyear + 1;
        if jyear > years_equity then jyear = years_equity;
        i = book_life;
        plant_arrays.working_capital(i) = 0;
        return;
        end initial;
```

```
/*****************************************************************************
*  C A L C U L A T E :                                                       *
*  This routine is the driver for the operation-period analysis.             *
*****************************************************************************/
    calculate:
        procedure;

            do i = 1 to analysis_period;
/*****************************************************************************
*  Calculate debt repayment and preferred stock dividends and repayments.    *
*  Long-term debt and preferred stock are retired straight-line over BOOK    *
*  LIFE years.  LIABILITIES part of BALANCE SHEET.                           *
*****************************************************************************/
            call do_debt_repay;
/*****************************************************************************
*  Calculate plant assests.  ASSETS part of BALANCE SHEET.                   *
*****************************************************************************/
            call do_assets;
/*****************************************************************************
*  Calculate tax depreciation.  TAX DEPRECIATION WORKSHEET.                  *
*****************************************************************************/
            call do_tax_dep;
/*****************************************************************************
*  Calculate expenses including book tax (INCOME STATEMENT and BOOK TAX CAL. *
*****************************************************************************/
            call do_expenses;
/*****************************************************************************
*  Calculate taxes on tax-basis REGULATED TAX CALCULATION and SORUCES and    *
*  USE of FUNDS STATEMENT.                                                   *
*****************************************************************************/
            call do_FERC_tax;
/*****************************************************************************
*  Calculate equity posistion (RETAINED EARNINGS STATEMENT).                 *
*****************************************************************************/
            call do_equity;
/*****************************************************************************
*  Calculate PARTNERSHIP RETURN and PARTNERSHIP CASHFLOW ANALYSIS.           *
*****************************************************************************/
            call do_partnership_return;
            if iyear < years_analysis then iyear = iyear + 1;
            if jyear < years_equity then jyear = jyear + 1;
            end;
        return;
        end calculate;

/*****************************************************************************
*  D O _ D E B T _ R E P A Y :                                               *
*  This routine calculates debt repayment and preferred stock dividends and  *
* preferred stock retirement.  Retirements are straight-line based upod BOOK  *
* LIFE.  All normalization impacts are applied to equity.                    *
*****************************************************************************/
    do_debt_repay:
        procedure;
```

```
        if i > debt_period then LT_debt_repayment(i) = 0.;
        LT_debt_interest(i) = total_LT_debt*LT_debt_rate;
        total_LT_debt = total_LT_debt - LT_debt_repayment(i);
        if i < debt_period then LT_debt_outstanding(i) =
        total_LT_debt - LT_debt_repayment(i);
        preferred_stock_div(i) = preferred_rate*preferred_stock_equity;
        preferred_stock_equity = preferred_stock_equity -
        preferred_stock_repayment(i);
        if i < book_life then preferred_stock_outstanding(i) =
        preferred_stock_equity - preferred_stock_repayment(i);
        return;
        end do_debt_repay;

/*****************************************************************************
*  D O _ A S S E T S :                                                       *
*  This routine calculates plant ASSETS.  Deferred tax effects are shown     *
* under LIABILITIES>                                                         *
*****************************************************************************/
    do_assets:
        procedure;

        if i > 1
            then do;
            accum_dep(i) = accum_dep(i-1) + book_dep_plant;
            accum_AFUDC_dep(i) = accum_AFUDC_dep(i-1) + AFUDC_dep_plant;
        if gross_net_flag = 1 then prop_tax_base = total_assets(i-1);
            end;
        net_plant(i) = construction - accum_dep(i);
        net_capitalized_AFUDC(i) = AFUDC_net - accum_AFUDC_dep(i);
        if inflation(iyear) = 0 then mid_inflation = cum_inflation*
        inflation(iyear)/log(1.+inflation(iyear));
        cum_inflation = cum_inflation*(1. + inflation(iyear));
        plant_arrays.working_capital(i) = plant_arrays.working_capital(i)*
        mid_inflation;
        total_assets(i) = net_plant(i) + net_capitalized_AFUDC(i) +
        plant_arrays.working_capital(i);
        return;
        end do_assets;

/*****************************************************************************
*  D O _ E X P E N S E S :                                                   *
*  This routine calculates REVENUES and EXPENSES for the INCOME STATEMENT.   *
* FEDERAL SUBSIDY is treated as a tax-free source of funds.  EXPENSES include *
* operating expenses, taxes, interest, preferred dividends, and common       *
* earnings.  The entries for the BOOK TAX CALCULATION are also calculated to *
* obtain the correct BOOK TAX entry on the INCOME STATEMENT.  Two modes of   *
* calculation are available :                                                *
*    - TOP DOWN : start with REVENUES and calculate down to COMMON EARNINGS  *
*    - BOTTOM UP : start with required COMMON EARNINGS and calculate required *
*                  revenues.                                                 *
*****************************************************************************/
    do_expenses:
        procedure;
```

```
        fuel_cost(i)=fuel_cost(i)*fuel_factor*capacity_factor(i)*mid_inflation;
        operation_maintanence(i) = operation_maintanence(i)*O_and_M_factor*
        capacity_factor(i)*mid_inflation;
        backfit_replacement(i) = mid_inflation/cum_inflation_save*
        backfit_replacement_rate*total_plant_cost;
        total_O_and_M(i) = fuel_cost(i) + operation_maintanence(i) +
        backfit_replacement(i);
        expenses.property_tax(i) = prop_tax_base*factors.property_tax;
        if i > 1 then beginning_common_equity(i) = end_common_equity(i-1);
        if bottom_up_top_down_flag = 0
            then do;
/*************************************************************************
*  BOTTOM UP calculation.  Start with COMMON EARNINGS and calculate revenue   *
* requirements.                                                              *
*************************************************************************/
            common_earnings(i) = beginning_common_equity(i)*
            rates.common_equity(iyear);
            rates.common_equity(i) = rates.common_equity(iyear);
            pre_tax_income_book(i)=(common_earnings(i)+preferred_stock_div(i)
            - Federal_subsidies(i) + book_dep_plant + AFUDC_dep_plant -
            book_depreciation)/(1. - FIT_tax);
            end;
            else do;
/*************************************************************************
* TOP DOWN calculation.  Start with revenues and calculate COMMON EARNINGS.   *
*************************************************************************/
            electric_sales(i) = busbar_costs(i)*capacity_factor(i)*kWh_factor;
            pre_tax_income_book(i) = electric_sales(i)*(1. - advalorem_tax)
            - total_O_and_M(i) - expenses.property_tax(i) - book_depreciation
            -  LT_debt_interest(i) + amort_ITC_book;
            end;
        FIT_book(i) = pre_tax_income_book(i)*FIT_tax;
        net_FIT_book(i) = FIT_book(i) - amort_ITC_book;
        total_pre_debt_expense(i) = total_O_and_M(i) + book_dep_plant
        + AFUDC_dep_plant + expenses.property_tax(i) + net_FIT_book(i);
        if bottom_up_top_down_flag = 0 then
        electric_sales(i) = (total_pre_debt_expense(i) + LT_debt_interest(i)
        + common_earnings(i) + preferred_stock_div(i) - Federal_subsidies(i))/
        (1.-advalorem_tax);
        ad_valorem_tax(i) = electric_sales(i)*advalorem_tax;
        total_O_and_M(i) = total_O_and_M(i) + ad_valorem_tax(i);
        total_pre_debt_expense(i)=total_pre_debt_expense(i)+ad_valorem_tax(i);
        total_revenues(i) = electric_sales(i) + Federal_subsidies(i);
        total_pre_debt_income(i)=total_revenues(i)-total_pre_debt_expense(i);
        net_income(i) = total_pre_debt_income(i) - LT_debt_interest(i);
        if bottom_up_top_down_flag = 0 then if capacity_factor(i) = 0 then
        busbar_costs(i) = electric_sales(i)/(kWh_factor*capacity_factor(i));
        else busbar_costs(i) = 1.e10;
            else do;
            common_earnings(i) = net_income(i) - preferred_stock_div(i);
            if beginning_common_equity(i) > 0 then
            rates.common_equity(i) = common_earnings(i)/
            beginning_common_equity(i);
            else if common_earnings(i) > 0 then rates.common_equity(i)
            = 1.e10; else rates.common_equity(i) = -1.e10;
            end;
        rates.common_equity(i) = rates.common_equity(i)*100.;
        return;
        end do expenses;
```

```
/*****************************************************************************
*  D O _ T A X _ D E P :                                                    *
* This routine calculates applicable tax depreciations.  Two modes are      *
* available (ACCELERATED_DEP_FLAG) :                                        *
*    - STRAIGHT-LINE : straight-line over TAX LIFE years.                   *
*    - ACRS : accelerated depreciation (5,10 or 15 year property).          *
* In each case STARTUP COSTS are depreciated straight-line over 5 years.    *
*****************************************************************************/
    do_tax_dep:
        procedure;

        dcl 1 depreciation_schedule static,
            2 five_yr(5) float init(.15,.22,.21,.21,.21),
            2 ten_yr(10) float init(.08,.14,.12,.10,.10,.10,.09,.09,.09,.09),
            2 fifteen_yr(15) float init(.05,.10,.09,.08,.07,.07,.06,
                .06,.06,.06,.06,.06,.06,.06,.06);

        if accelerated_dep_flag = 1
            then do;
            if i<=tax_life then if tax_life = 5 then five_year_property(i)
            = tax_base*five_yr(i); else if tax_life = 10 then
            ten_year_property(i) = ten_yr(i)*tax_base; else if
            tax_life = 15 then fifteen_year_property(i) =
            fifteen_yr(i)*tax_base;
            total_tax_dep(i) = five_year_property(i) + ten_year_property(i)
            + fifteen_year_property(i);
            end;
            else do;
            total_tax_dep(i) = other_dep;
            if i = tax_life then other_dep = 0;
            end;
        if i <= 5 then startup_dep(i) = 0.2*startup_cost;
        total_tax_dep(i) = total_tax_dep(i) + startup_dep(i);
        return;
        end do_tax_dep;

/*****************************************************************************
*  D O _ F E R C _ T A X :                                                  *
*  This routine calculates deferred taxes and any required partner loans.   *
*  Two deferred tax modes are available (CONSOLIDATED_TAX_FLAG) :           *
*    - UNCONSOLIDATED : negative FIT DUE is not allowed (single project)    *
*    - CONSOLIDATED : negative taxes allowed (multi-project for tax purposes) *
* Deferred taxes can be NORMALIZED (required under TEFRA if using ACRS) or  *
* FLOWED THROUGH. (FLOWTHROUGH_NORMALIZE_FLAG).                             *
*****************************************************************************/
    do_FERC_tax:
        procedure;

        pre_tax_income_FERC(i) = electric_sales(i) - total_O_and_M(i) -
        LT_debt_interest(i) - total_tax_dep(i) - expenses.property_tax(i);
        FIT_tax_FERC(i) = pre_tax_income_FERC(i)*FIT_tax;
        if i>1 then cum deferred FIT(i)=cum deferred FIT(i-1)-deferred reserve;
```

```
        if consolidated_tax_flag = 0
            then do;
/*****************************************************************************
*  UNCONSOLIDATED : do not permit negative FIT (NET_FIT_FERC).               *
*****************************************************************************/
            deferred_FIT_taken(i) = min(FIT_tax_FERC(i),cum_deferred_FIT(i));
            cum_deferred_FIT(i) = cum_deferred_FIT(i) - deferred_FIT_taken(i);
            reduction = FIT_tax_FERC(i) - deferred_FIT_taken(i);
            credit_ITC(i) = min(reduction,ITC_remaining);
            end;
            else do;
/*****************************************************************************
*  CONSOLIDATED : take all available ITC credits.                            *
*****************************************************************************/
            deferred_FIT_taken(i) = 0.;
            credit_ITC(i) = ITC_remaining;
            end;
        net_FIT_FERC(i) = FIT_tax_FERC(i) - credit_ITC(i);
        ITC_remaining = ITC_remaining - credit_ITC(i);
        amort_ITC(i) = credit_ITC(i) - amort_ITC_book;
        deferred_ITC(i) = amort_ITC(i);
        if i > 1 then deferred_ITC(i) = deferred_ITC(i) + deferred_ITC(i-1);
        if flowthrough_normalize_flag = 1
            then do;
/*****************************************************************************
*  NORMALIZE deferred taxes.                                                 *
*****************************************************************************/
            deferred_FIT(i)=max(FIT_tax*(total_tax_dep(i)-book_depreciation),
            -deferred_reserve);
            deferred_reserve = deferred_reserve + deferred_FIT(i);
            cum_deferred_FIT(i) = cum_deferred_FIT(i) + deferred_reserve;
            deferred_FIT(i) = deferred_FIT(i) - deferred_FIT_taken(i);
            end;
/*****************************************************************************
*  FLOW THROUGH deferred taxes.                                              *
*****************************************************************************/
        else deferred_FIT(i) = - deferred_FIT_taken(i);
        total_internal(i) = net_income(i) + book_dep_plant +
        AFUDC_dep_plant + amort_ITC(i) + deferred_FIT(i);
        repay_LT_debt = LT_debt_repayment(i);
        delta_working_capital(i)=plant_arrays.working_capital(i)-cum_wk_cap;
        cum_wk_cap = plant_arrays.working_capital(i);
        loans_required = preferred_stock_repayment(i)+ preferred_stock_div(i) +
        repay_LT_debt+delta_working_capital(i)-total_internal(i);
        call do_loans;
        total_sources(i) = total_internal(i) + partner_loans(i) + DOE_loans(i);
        if DOE_loans(i) = 0 & i < analysis_period then LT_debt_repayment(i+1) =
        LT_debt_repayment(i);
        return;
        end do_FERC_tax;
```

```
/*****************************************************************************
*  D O _ L O A N S :                                                        *
*  This routine allocates loans among the government and equity holders.    *
*  If the two-year average plant capacity factor falls be low the average of *
* the TRIGGER_POINT and NOMINAL_POINT then required loans are covered by     *
* government loans.  Otherwise loans are covered by equity loans.            *
*****************************************************************************/
    do_loans:
        procedure;

        if loans_required>0 & beginning_common_equity(i)>=0
            then do;
            if trigger_point >= nominal_point then both = 1;
            else if i > 1 then if (capacity_factor(i-1) + capacity_factor(i)
            -2*trigger_point)/(nominal_point-trigger_point) <= 1 then both=1;
            else both = 0;
            else if i < analysis_period then if (capacity_factor(i+1) +
            capacity_factor(i) - 2*trigger_point)/(nominal_point -
            trigger_point) <= 1 then both = 1;
            if both = 1 & i < debt_period
                then do;
                DOE_loans(i) = loans_required;
                partner_loans(i) = 0;
                if i < analysis_period then LT_debt_repayment(i+1) =
                LT_debt_repayment(i) + DOE_loans(i)/(debt_period - i);
                LT_debt_rate = (LT_debt_rate*total_LT_debt + DOE_loans(i)*
                rates.LT_debt(iyear))/(total_LT_debt + DOE_loans(i));
                total_LT_debt = total_LT_debt + DOE_loans(i);
                LT_debt_outstanding(i)=total_LT_debt-LT_debt_repayment(i+1);
                end;
                else do;
                partner_loans(i) = loans_required;
                DOE_loans(i) = 0;
                if i > 1 then cum_partner_loans(i) = cum_partner_loans(i-1)
                + loans_required;
                else cum_partner_loans(i) = loans_required;
                end;
            partner_loan_repayment(i) = 0;
            end;
            else do;
            if i > 1 then cum_partner_loans(i) = cum_partner_loans(i-1);
            DOE_loans(i) = 0;
            partner_loans(i) = 0;
            if beginning_common_equity(i) >= 0 then partner_loan_repayment(i) =
            min(-loans_required,cum_partner_loans(i));
            else partner_loan_repayment(i) = 0.;
            cum_partner_loans(i) =
            cum_partner_loans(i)-partner_loan_repayment(i);
            end;
        return;
        end do_loans;
```

```
/*****************************************************************************
*  D O _ E Q U I T Y :                                                       *
*  This routine calculates the RETAINED EARNINGS STATEMENT.                  *
*****************************************************************************/
    do_equity:
        procedure;

        common_stock_div(i) = total_sources(i) - preferred_stock_repayment(i) -
        preferred_stock_div(i) - repay_LT_debt - partner_loan_repayment(i)
        - delta_working_capital(i);
        total_common_equity(i) = beginning_common_equity(i) +
        common_earnings(i);
        end_common_equity(i) = total_common_equity(i) -
        common_stock_div(i);
        total_equity(i) = end_common_equity(i) + preferred_stock_equity
        + cum_partner_loans(i);
        return;
        end do_equity;

/*****************************************************************************
*  D O _ P A R T N E R S H I P _ R E T U R N :                               *
*  This routine calculates the PARTNERSHIP RETURN and PARTNERSHIP CASHFLOW   *
* ANALYSIS.                                                                  *
*****************************************************************************/
    do_partnership_return:
        procedure;

        after_tax_cash(i) = pre_tax_income_FERC(i) - FIT_tax_FERC(i);
        total_cash_avail(i) = after_tax_cash(i) + total_tax_dep(i) +
        Federal_subsidies(i);
        net_to_partners(i)=total_cash_avail(i)-preferred_stock_repayment(i)-
        delta_working_capital(i)-preferred_stock_div(i)-LT_debt_repayment(i);
        if i > 1 then cum_to_partners(i) = cum_to_partners(i-1);
        cum_to_partners(i) = cum_to_partners(i) + net_to_partners(i);
        return;
        end do_partnership_return;

/*****************************************************************************
*  D O N E :                                                                 *
*  Print ANALYSIS COMPLETE message.                                          *
*****************************************************************************/
    done:
        procedure;

        put file(crt) skip(2) edit('CASHFLOW analysis complete') (a);
        return;
        end done;

    end cashflow;
```

```
/*****************************************************************************
*  R E P O R T :                                                            *
* This routine lists various summary reports.  Reports can be listed on     *
* either the printer or previewed on the consol.  The original version      *
* outputed the entire OUTPUT_DATA array.                                    *
*                                                                           *
*   Original Version  8/5/83                                                *
*   Version 1.1  9/15/83                                                    *
*     This version includes the following summary reports:                  *
*       - Run Summary                                                       *
*       - Balance Sheet                                                     *
*       - Income Statement                                                  *
*       - Sources and Uses of Funds Statement                               *
*       - Retained Earnings Statement/Tax Depreciation Worksheet            *
*       - Tax Calculation (Book/Regulation Basis)                           *
*       - Partnership Return & Cashflow Analysis                            *
*       - Cashflow Summary (includes ROI's)                                 *
*       - Busbar Cost Plot                                                  *
*     Reports can be listed individually or a special summary option        *
*     selected to list all reports.  Reports can be listed for all          *
*     years or selected years (up to 8 years can be displayed per page).    *
*     This routine calls eight subroutines:                                 *
*       FIND_RECORD - Reads the specially keyed,variable length text        *
*                     file REPORT.DAT containing report labels.             *
*       GET_INTEGER - Reads integer from consol for menu options.           *
*       SUMARY      - Prints special run summary.                           *
*       CALROI      - Calculates ROI's based upon after-tax cashflows.      *
*       CENTER      - Centers a line of output on screen or page.           *
*       SCREEN_CLEAR- Request character input and clears screen.            *
*       PLOTBUSBAR  - Plots busbar costs.                                   *
*       CLEARSCREEN - Clears the screen and positions the cursor at upper left*
*   Version 1.2  10/10/83                                                   *
*     Makes numberous revisions for including startup costs, working        *
*     capital, backfit/replacement costs, preferred stock, Federal          *
*     subsidies, and DOE/partner loans, including:                          *
*       - Balance Sheet:                                                    *
*           Delete TOTAL LT DEBT line                                       *
*           Delete TOTAL DEFERRED TAX line                                  *
*           Add WORKING CAPITAL line                                        *
*           Add PARTNER LOANS OUTSTANDING line                              *
*           Replace PREFERRED STOCK line with                               *
*           PREFERRED STOCK DUE and PREFERRED STOCK OUTSTANDING lines       *
*       - Income Statement:                                                 *
*           Add BACKFIT/REPLACEMENT line                                    *
*       - Sources and Uses of Funds Statement                               *
*           Add PREFERRED STOCK DUE line                                    *
*       - Tax Calculation                                                   *
*           Change TOTAL REVENUES to ELECTRIC SALES                         *
*       - Partnership Return & Cashflow Analysis                            *
*           Change TOTAL REVENUES to ELECTRIC SALES                         *
*           Add FEDERAL SUBSIDIES line                                      *
*           Add PREFERRED STOCK DUE line                                    *
*           Add CHANGE IN WORKING CAPITAL line                              *
```

```
*   Version 1.3  10/20/83                                                  *
*     Corrects two deferred tax addresses:                                 *
*       - Balance Sheet:                                                   *
*           Change deferred_FIT to cum_deferred_FIT                        *
*       - Sources and Uses of Funds:                                       *
*           Change cum_deferred_FIT to deferred_FIT                        *
*       - Change discount_factor = discount_rate**(number+1) in init_cash_sum *
*       - Change other_dep_year = total_tax_dep - startup_dep              *
*   Version 1.4  11/10/83                                                  *
*     Include AD VALOREM TAX line                                          *
*   Version 1.5  4/15/84                                                   *
*     Correct bug in calculation of DIS.R.A.T.C.F                          *
*     Increase print limit to f(7,1)                                       *
*     Change REAL labels to BASE YEAR labels                               *
*     Calculate Fixed Charge Rates                                         *
*     Change clearscreen to subroutine call to ensure compatability of     *
*     between CP/M and IBM/PC-DOS versions.                                *
*****************************************************************************/
report:
    procedure;
    %include 'input.dcl';
    %include 'constr.dcl';
    %include 'output.dcl';
    %include 'diomod.dcl';
    dcl (buff_length,escape_flag,st_analysis,i,j,imod,jmod,number_years,
        cashflow_init,iyear,jyear,construction_period,number,st_construction,
        line_number,index_number,address_number,index(7),years_array(30),
        start,year_end) fixed,
        (address(21,6),addd) pointer,
        (construction_year(30),AFUDC_year(30),book_depreciation_year(30),
        book_dep_plant_year(30),book_dep_AFUDC_year(30),total_book_dep_year(30)
        ,amort_ITC_book_year(30),roi_cur(30),roi_real(30),other_dep_year(30),
        atcf_cur(45),atcf_real(45),cum_before_tax,cum_const_tax,cum_btcf,
        cum_taxes_due,total_book_dep,discount_factor,discount_rate,cum_atcf(45)
        ,cum_discount_rate,cum_inflation,cum_inflation_save,level_busbar_cur,
        level_busbar_real,discount_rate_real,cum_kWh,fixed_charge_cur,
        cum_dis_atcf(45),cents_per_kWh_real(30),fixed_charge_real) float,
        print_array(30) float based(addd),
        buffer char(79) var, continue char(1),
        units char(29) var static init('(Millions of Current Year $)'),
        find_record entry(fixed,fixed,char(79) var),
        get_integer entry(fixed,fixed),
        calroi entry(fixed,fixed,fixed,(45) float,(30) float),
        plotbusbar entry(file,fixed,fixed,(30) float,(30) float),
        sumary entry(file,float,float,float,float,float,float,float,float),
        screen_clear entry(char(1),fixed),
        center entry(file,fixed,fixed,char(79) var),
        (crt,printer) file;

/*****************************************************************************
*  Initialize variables before writing reports.                             *
*****************************************************************************/
    call initial;
```

```
/*****************************************************************************
*  Print main menu (soft-copy display vs. hard-copy list) and get option.    *
*****************************************************************************/
    imod = -1;
        do while(imod = 0);
        put file(crt) skip edit('Enter 0 - No Listing',
        '1 - Preview on Screen','2 - List on Printer')
        (a,col(7));
        put file(crt) skip edit('? ') (a);
        call get_integer(escape_flag,imod);
        if (imod < 0 ! imod > 2) then put file(crt) skip edit(bell) (a);
        if imod=1 then call do_report(crt);
        else if imod=2
            then do;
            call do_report(printer);
            put file(printer) skip;
            end;
        if imod = 0 then call clearscreen;
        end;
/*****************************************************************************
*  Close REPORT.DAT file.                                                    *
*****************************************************************************/
    buffer = 'report';
    call find_record(buff_length,-1,buffer);
    return;

/*****************************************************************************
*  REPORT.INI contains the code for I N I T I A L.                           *
*****************************************************************************/
    %include 'report.ini';

/*****************************************************************************
*  L E V E L I Z E :                                                         *
*  This routine calculates levelized busbar costs for LEVELIZATION_PERIOD yrs.*
*    - RATE_DISCOUNT : discount rate for levelizing.                         *
*    - LEVEL_BUSBAR : levelized busbar cost.                                 *
*    - ADDD : points to array containing busbar costs.                       *
*****************************************************************************/
    levelize:
        procedure(rate_discount,level_busbar,addd);

        dcl (rate_discount,level_busbar,costs(30) based(addd)) float,
            addd pointer;

        discount_factor = 1.;
        level_busbar = 0.;
        j = levelization_period;
            do i = 1 to j;
            discount_factor = discount_factor*rate_discount;
            level_busbar = level_busbar + costs(i)*discount_factor;
            end;
        if rate_discount = 1 then level_busbar = level_busbar*
        (1. - rate_discount)/(rate_discount*(1. - rate_discount**j));
        else level_busbar = level_busbar/levelization_period;
        return;
        end levelize;
```

```
/*****************************************************************************
*  D O _ R E P O R T :                                                       *
*  This routine prints the report menu, gets the selection and calls the     *
* appropriate print routine.                                                 *
*    - X : output print file (CRT or PRINTER).                               *
*****************************************************************************/
    do_report:
        procedure(x);

        dcl x file;

        jmod = -1;
            do while(jmod = 0);
            call clearscreen;
            put file(crt) edit('Enter 0 - No More Reports',
            '1 - Summary','2 - Balance Sheet',
            '3 - Income Statement','4 - Sources and Uses of Funds Statement',
            '5 - Retained Earnings Statement/Tax Depreciation Worksheet',
            '6 - Tax Calculation (Book/Regulation Basis)',
            '7 - Partnership Return & Cashflow Analysis',
            '8 - Cashflow Summary','9 - Busbar Cost Plot','10 - All of Above',
            '11- Select Printout Years') (a,col(7));
            put file(crt) skip edit('? ') (a);
            call get_integer(escape_flag,jmod);
            if jmod < 0 ! jmod > 11 then put file(crt) skip edit(bell) (a);
            if jmod = 1 then call do_summary(x);
            if jmod = 10
                then do;
                jmod = 1;
                line_number = 0;
                if x=printer then put file(crt) skip edit('Printing Report: 1')
                (a);
                call do_summary(x);
                if x=printer then put file(crt) skip edit('Printing Report: 1')
                (a);
                    do while(jmod > 0 & jmod < 7);
                    if x = crt then line_number = 0;
                    jmod = jmod + 1;
                    if x=printer then put file(crt) edit(backspace,jmod)
                    (a,f(1,0));
                    call print_report(x,jmod);
                    end;
                if jmod = -1
                    then do;
                    jmod = 8;
                    if x = printer then put file(crt) edit(backspace,jmod)
                    (a,f(1,0));
                    call cash_sum(x);
                    end;
                if jmod = -1
                    then do;
                    if x = printer then put file(crt) skip
                    edit('Printing BUSBAR COST plot') (a);
                    call do_plot(x);
                    end;
                end;
```

```
            else if jmod > 1 & jmod < 8
                then do;
                line_number = 0;
                call print_report(x,jmod);
                end;
            else if jmod = 8 then call cash_sum(x);
            else if jmod = 9 then call do_plot(x);
            else if jmod = 11 then call select_years;
            end;
        return;
        end do_report;

/*****************************************************************************
*  D O _ S U M M A R Y :                                                     *
*  This routine prints the special SUMMARY report useing subroutine SUMARY.  *
*    - X : output print file (CRT or PRINTER).                               *
*****************************************************************************/
    do_summary:
        procedure(x);

        dcl x file;

        if cashflow_init = 0 then call init_cash_sum;
        buffer = 'report';
        call find_record(buff_length,-1,buffer);
        discount_factor = 100.0*(1./discount_rate - 1.);
        cum_discount_rate = 100.0*(1./discount_rate_real - 1.);
        call sumary(x,discount_factor,cum_discount_rate,level_busbar_cur,
        level_busbar_real,fixed_charge_cur,fixed_charge_real,roi_cur(30),
        roi_real(30));
        call find_record(buff_length,0,buffer);
        return;
        end do_summary;

/*****************************************************************************
*  P R I N T _ R E P O R T :                                                 *
*  This routine prints the JMODth report on output device X.  The routine   *
* first prints the report HEADER and then up to 8 years for each line of the *
* report.  Any value >= 99999.95 or <= -9999.95 is replaces by a string of  *
* stars to avoid a CONVERSION error.                                        *
*****************************************************************************/
    print_report:
        procedure(x,jmod);

        dcl x file,
            jmod fixed;

        iyear = 1;
/*****************************************************************************
*  Print NUMBER_YEARS of data, 8 years at a time.                            *
*****************************************************************************/
            do while(iyear <= number_years);
            jyear = iyear + 7;
            if jyear > number years then jyear = number years;
```

```
/*****************************************************************************
*  Print report HEADER.                                                      *
*****************************************************************************/
            call do_header(x,jmod,line_number);
            if jmod = -1 then return;
            address_number = 1;
            index_number = index(jmod-1) + 2;
                do while(index_number < index(jmod));
                addd = address(address_number,jmod-1);
                if unspec(addd) = 0 then put file(x) skip;
                else if unspec(addd) = 1
                    then do;
/*****************************************************************************
* Center and print header label.                                             *
*****************************************************************************/
                    call find_record(buff_length,index_number,buffer);
                    call center(x,buff_length,1,buffer);
                    end;
                    else do;
/*****************************************************************************
*  Print LABEL and upto 8 year's of values.                                  *
*****************************************************************************/
                    call find_record(buff_length,index_number,buffer);
                    put file(x) skip edit(buffer) (a(22));
                        do i = iyear to jyear;
                        if number_years=analysis_period then j=i;
                        else j=years_array(i);
                        if print_array(j) >= 99999.95 then
                        put file(x) edit('  *****') (a);
                        else if print_array(j) <= -9999.95 then
                        put file(x) edit(' -*****') (a); else
                        put file(x) edit(print_array(j)) (f(7,1));
                        end;
                    end;
                if unspec(addd) = 0 then index_number = index_number + 1;
                address_number = address_number + 1;
                line_number = line_number + 1;
                end;
            iyear = jyear + 1;
            end;
        if x = crt then call check_escape;
        return;
        end print_report;

/*****************************************************************************
*  C A S H _ S U M :                                                         *
*  This routine prints the CASHFLOW SUMMARY and ROIs on output device X.     *
*****************************************************************************/
    cash_sum:
        procedure(x);

        dcl x file,
            max_year fixed;
```

```
/*****************************************************************************
*  Call INIT_CASH_SUM to calculate derived CASHFLOW SUMMARY values (once only)*
*****************************************************************************/
        if cashflow_init = 0 then call init_cash_sum;
        line_number = 0;
/*****************************************************************************
*  Print HEADER.                                                            *
*****************************************************************************/
        call do_header(x,jmod,line_number);
/*****************************************************************************
*  Print report during construction period.                                 *
*****************************************************************************/
            do i = 1 to construction_period;
            put file(x) skip edit(i+start_construction-1,-before_tax(i),
            interest_credit(i)+ITC_credit(i),'--','--',after_tax(i),
            cum_atcf(i),cum_dis_atcf(i)) (f(4,0),f(10,3),f(11,3),x(6),a,x(8),
            a,f(14,3),2f(11,3));
            end;
/*****************************************************************************
*  Print summary for operation period.                                      *
*****************************************************************************/
        start = 0;
        max_year = construction_period + analysis_period;
        if x = crt then jyear = 15; else jyear = max_year;
        iyear = construction_period;
            do while(jyear<=max_year);
                do while(iyear<jyear);
                start = start + 1;
                iyear = iyear + 1;
                put file(x) skip edit(iyear+start_construction-1,'--','--',
                FIT_tax_FERC(start),net_to_partners(start)+FIT_tax_FERC(start),
                net_to_partners(start),cum_atcf(iyear),cum_dis_atcf(iyear))
                (f(4,0),x(5),a,x(9),a,f(14,3),f(10,3),3f(11,3));
                end;
            put file(x) skip edit(buffer) (a);
            if jyear = max_year then jyear = max_year + 1;
                else do;
                if x = crt
                    then do;
                    call check_escape;
                    if jmod = - 1 then return;
                    line_number = 0;
                    call do_header(x,jmod,line_number);
                    end;
                jyear = jyear + 15;
                if jyear > max_year then jyear = max_year;
                end;
            end;
/*****************************************************************************
*  Print report TOTALS.                                                     *
*****************************************************************************/
        put file(x) skip edit('TOT.',cum_before_tax,cum_const_tax,
        cum_taxes_due,cum_btcf,cum_atcf(iyear))
        (a,f(10,3),2f(11,3),f(10,3),f(11,3));
```

```
        put file(x) skip edit(buffer) (a);
        if x = crt
            then do;
            call check_escape;
            if jmod = -1 then return;
            end;
        else put file(x) skip edit(new_page) (a,skip(3));
/*****************************************************************************
*  Print ROIs in as-spent and base-year $.                                   *
*****************************************************************************/
        buffer = 'Summary of ROIs (Current $)';
        buff_length = length(buffer);
        call center(x,buff_length,3,buffer);
        call do_print(x,roi_cur);
        buffer = 'Summary of ROIs (Base-Year $)';
        buff_length = length(buffer);
        call center(x,buff_length,3,buffer);
        call do_print(x,roi_real);
        if x = crt then call check_escape;
        return;
        end cash_sum;

/*****************************************************************************
*  I N I T _ C A S H _ S U M :                                               *
*  This routine calculates NET A.T. CASH FLOW, CUM. A.T. CASH FLOW, and      *
* CM.DIS.R.A.T.C.F.  It also uses CAL_ROI to calculate annual ROIs in        *
* as-spent and base-year $.                                                  *
*****************************************************************************/
    init_cash_sum:
        procedure;

        dcl (sum_cash,dis_sum_cash) float;

        cashflow_init = 1;
        cum_before_tax = 0;
        cum_const_tax = 0;
        cum_btcf = 0;
        cum_taxes_due = 0;
        sum_cash = 0;
        dis_sum_cash = 0;
        discount_factor = discount_rate_real**number;
        cum_inflation = cum_inflation_save;
        if number < 30 then iyear = number; else number = 30;
/*****************************************************************************
*  Calculate values for construction period.                                 *
*****************************************************************************/
            do i = 1 to construction_period;
            discount_factor = discount_factor*discount_rate_real;
            cum_inflation = cum_inflation*(1. + inflation(iyear));
            if iyear < 30 then iyear = iyear + 1;
            cum_before_tax = cum_before_tax - before_tax(i);
            cum_const_tax = cum_const_tax + interest_credit(i) + ITC_credit(i);
            sum_cash = sum_cash + after_tax(i);
```

```
            atcf_cur(i) = after_tax(i);
            atcf_real(i) = after_tax(i)/cum_inflation;
            dis_sum_cash = dis_sum_cash + after_tax(i)*discount_factor/
            cum_inflation;
            cum_atcf(i) = sum_cash;
            cum_dis_atcf(i) = dis_sum_cash;
            end;
        jyear = construction_period;
/*****************************************************************************
*  Calculate values for operation period.                                   *
*****************************************************************************/
            do i = 1 to analysis_period;
            discount_factor = discount_factor*discount_rate_real;
            cum_inflation = cum_inflation *(1. + inflation(iyear));
            if iyear < 30 then iyear = iyear + 1;
            cum_taxes_due = cum_taxes_due + FIT_tax_FERC(i);
            cum_btcf = cum_btcf + net_to_partners(i) + FIT_tax_FERC(i);
            sum_cash = sum_cash + net_to_partners(i);
            dis_sum_cash = dis_sum_cash + net_to_partners(i)*discount_factor/
            cum_inflation;
            jyear = jyear + 1;
            cum_atcf(jyear) = sum_cash;
            cum_dis_atcf(jyear) = dis_sum_cash;
            atcf_cur(jyear) = net_to_partners(i);
            atcf_real(jyear) = net_to_partners(i)/cum_inflation;
            end;
/*****************************************************************************
*  Calculate annual ROIs in as-spent and base-year $.                       *
*****************************************************************************/
        put file(crt) skip(2) edit('Beginning ROI Analysis (Current $)') (a);
        call calroi(st_construction,jyear,analysis_period,atcf_cur,roi_cur);
        put file(crt) skip(2) edit('Beginning ROI Analysis (Base_Year $)') (a);
        call calroi(st_construction,jyear,analysis_period,atcf_real,roi_real);
        return;
        end init_cash_sum;

/*****************************************************************************
*  D O _ P L O T :                                                          *
*  This routine plots and prints busbar costs in as-spent and base-year $   *
* on output device X.                                                       *
*****************************************************************************/
    do_plot:
        procedure(x);

        dcl x file;

        if x = crt then call clearscreen;
        else put file(x) skip edit(new_page) (a);
        if x = crt then i = 0; else i = 1;
        buff_length = length(header);
        call center(x,buff_length,i,header);
/*****************************************************************************
*  Print header and busbar plot.                                            *
*****************************************************************************/
```

```
        put file(x) skip;
        call plotbusbar(x,st_analysis,analysis_period,busbar_costs,
        cents_per_kWh_real);
        if x = crt
            then do;
            i = 0;
            call clearscreen;
            end;
        else i = 2;
/*****************************************************************************
*  Print levelized and annual busbar costs in as-spent and base-year $.      *
*****************************************************************************/
        buffer = 'Summary of Levelized Busbar Costs';
        buff_length = length(buffer);
        call center(x,buff_length,i,buffer);
        put file(x) skip edit('(Levelized over ',levelization_period,
        ' years)') (col(28),a,f(2,0),a);
        discount_factor = 100.0*(1./discount_rate - 1.);
        cum_discount_rate = 100.0*(1./discount_rate_real - 1.);
        put file(x) skip(2)
        edit(level_busbar_cur,' (current cents/kWh,discount rate : ',
        discount_factor,' (%))',level_busbar_real,
        ' (base yr. cents/kWh,discount rate: ',cum_discount_rate,' (%))')
        (col(15),f(6,2),a,f(6,3),a);
        buffer = 'Summary of Busbar Costs (current cents/kWh)';
        buff_length = length(buffer);
        call center(x,buff_length,3,buffer);
        call do_print(x,busbar_costs);
        buffer = 'Summary of Busbar Costs (base yr. cents/kWh)';
        buff_length = length(buffer);
        call center(x,buff_length,3,buffer);
        call do_print(x,cents_per_kWh_real);
        if x = crt then call check_escape;
        end do_plot;

/*****************************************************************************
*  D O _ H E A D E R :                                                       *
*  This routine prints the HEADER for report JMOD on output device X and    *
* returns the number of lines printed in LINE_NUMBER.                       *
*****************************************************************************/
    do_header:
        procedure(x,jmod,line_number);

        dcl x file,
        (jmod,line_number,skip,nyear) fixed;

        if jmod = 5 ! jmod = 8 then j=1; else j = 0;
        i = index(jmod-1);
        skip = 0;
        if x = crt
            then do;
            if line_number = 0
                then do;
                call check_escape;
                if jmod = -1 then return;
                end;
```

```
    else call clearscreen;
    line_number = 1;
    if jmod > 3 & jmod = 6
        then do;
        buff_length = length(header);
        call center(x,buff_length,0,header);
        skip = 1;
        end;
    if j = 1 then skip = skip + 1;
    line_number = line_number + skip;
    buff_length = length(units);
    call find_record(buff_length,i,buffer);
    call center(x,buff_length,skip,buffer);
    if j = 1
        then do;
        line_number = line_number + 1;
        call center(x,buff_length,1,units);
        end;
    end;
    else do;
    if line_number > 30 ! line_number = 0
        then do;
        line_number = 0;
        put file(x) skip edit(new_page) (a);
        skip = 1;
        end;
    else skip = 5;
    buff_length = length(header);
    call center(x,buff_length,skip,header);
    line_number = line_number + skip;
    i = index(jmod-1);
    call find_record(buff_length,i,buffer);
    call center(x,buff_length,2,buffer);
    buff_length = length(units);
    call center(x,buff_length,1,units);
    put file(x) skip;
    line_number = line_number + 6;
    end;
if jmod = 8
    then do;
    put file(x) skip edit(' ') (col(21),a);
        do i = iyear to jyear;
        if number_years = analysis_period then nyear = i;
        else nyear = years_array(i);
        put file(x) edit(year_end + nyear - 1) (f(7,0));
        end;
    line_number = line_number + 1;
    end;
if x = printer then put file(x) skip;
i = index(jmod-1) + 1;
line_number = line_number + 1;
call find_record(buff_length,i,buffer);
call center(x,buff length,1,buffer);
```

```
        if jmod = 8
            then do;
            i = i + 1;
            call find_record(buff_length,i,buffer);
            call center(x,buff_length,1,buffer);
            i = i + 1;
            call find_record(buff_length,i,buffer);
            call center(x,buff_length,1,buffer);
            line_number = line_number + 2;
            end;
        return;
        end do_header;

/*****************************************************************************
*  D O _ P R I N T :                                                         *
*  This routine prints the values in ARRAY on output device X.  Out-of-range *
* values are trapped and printed as stars to prevent a conversion error.     *
*****************************************************************************/
    do_print:
        procedure(x,array);

        dcl x file,
        array(30) float;

            do i = 1 to analysis_period;
            if i=divide(i,5,15)*5+1 then put file(x) skip edit('') (col(3),a);
            else put file(x) edit('    ') (a);
            put file(x) edit(i+st_analysis-1) (f(4,0));
            if array(i) < -99.95 then put file(x) edit(' -*****') (a);
            else if array(i) > 1000 then put file(x) edit('  *****') (a);
            else put file(x) edit(array(i)) (f(7,2));
            end;
        return;
        end do_print;

/*****************************************************************************
*  S E L E C T _ Y E A R S :                                                 *
*  This routine allows output of either SELECTED or ALL analysis years.      *
*****************************************************************************/
    select_years:
        procedure;

        imod = -1;
/*****************************************************************************
*  Print menu and obtain option value.                                       *
*****************************************************************************/
            do while(imod = 0);
            call clearscreen;
            if number_years = analysis_period then put file(crt) skip(0)
            edit('ALL YEARS option selected') (a);
                else do;
                put file(crt)skip(0) edit('SELECTED YEARS option',number_years,
                ' years selected:') (a,skip,f(2,0));
                put file(crt) edit((i,' - ',years_array(i)+st_analysis-1
                do i = 1 to number_years)) (skip,5(f(2,0),a,f(4,0),x(6)));
                end;
```

```
        put file(crt) skip(2) edit('Enter 0 - To Exit',
        '1 - To Output All Years','2 - To Output Selected Years')
        (a,col(7));
        put file(crt) skip edit('? ') (a);
        call get_integer(escape_flag,imod);
        if imod < 0 ! imod > 2 then put file(crt) skip edit(bell) (a);
        if imod = 1
            then do;
/*****************************************************************************
* Output ALL years.                                                          *
*****************************************************************************/
            number_years = analysis_period;
            imod = 0;
            end;
        else if imod = 2
            then do;
/*****************************************************************************
*  Enter set of years to be output and check for valid range.                *
*****************************************************************************/
            iyear = 1;
            number_years = 0;
            call clearscreen;
            put file(crt) skip(0) edit('Enter years (8 years/page):') (a);
            put file(crt) skip;
                do while(iyear = 0 & number_years < analysis_period);
                put file(crt) skip(0) edit('Enter year ',number_years+1,
                ' (0 to exit) ? ') (a,f(2,0));
                call get_integer(escape_flag,iyear);
                if (iyear < st_analysis ! iyear > st_analysis +
                analysis_period -1) & iyear = 0 then put file(crt)
                edit(bell) (a);
                else if iyear = 0
                    then do;
                    number_years = number_years + 1;
                    years_array(number_years) = iyear - st_analysis + 1;
                    end;
                end;
            end;
        end;
    imod = 1;
    return;
    end select_years;

/*****************************************************************************
*  C H E C K _ E S C A P E :                                                 *
*  This routine clears the screen and checks for an ESCAPE character.  If an *
* ESCAPE character is input, JMOD is set to -1.                              *
*****************************************************************************/
    check_escape:
        procedure;

        call screen_clear(continue,1);
        if continue = escape then jmod = -1;
        return;
        end check_escape;
```

```
/*****************************************************************************
* File R E P O R T . I N I : contains the code for subroutine INITIAL of     *
* routine REPORT.                                                            *
*                                                                            *
*  I N I T I A L :                                                           *
*  This routine opens the file REPORT.DAT, initializes various variables and *
* arrays, and calculates levelized busbar costs.                             *
*****************************************************************************/
    initial:
        procedure;

        number_years = analysis_period;
        buffer = 'report';
        call find_record(buff_length,0,buffer);
        st_analysis = end_construction;
        cashflow_init = 0;
        total_book_dep = book_dep_plant + AFUDC_dep_plant;
/*****************************************************************************
*  Create print arrays for non-changing values.                              *
*****************************************************************************/
            do i = 1 to analysis_period;
            construction_year(i) = construction;
            AFUDC_year(i) = AFUDC_dep_plant*book_life;
            book_dep_plant_year(i) = book_dep_plant;
            book_dep_AFUDC_year(i) = AFUDC_dep_plant;
            total_book_dep_year(i) = total_book_dep;
            book_depreciation_year(i) = book_depreciation;
            amort_ITC_book_year(i) = amort_ITC_book;
            if accelerated_dep_flag = 1 then other_dep_year(i) = 0;
            else other_dep_year(i) = total_tax_dep(i)-startup_dep(i);
            end;
        preferred_stock_repayment(analysis_period+1) = 0;
        LT_debt_repayment(analysis_period+1) = 0;
/*****************************************************************************
*  Initialize BALANCE SHEET pointers.  Index gives offset of 1st label in    *
* file REPORT.DAT.                                                           *
*****************************************************************************/
        index(1) = 1;
        address(1,1) = addr(plant_arrays.working_capital);
        address(2,1) = addr(construction_year);
        address(3,1) = addr(accum_dep);
        address(4,1) = addr(net_plant);
        address(5,1) = addr(AFUDC_year);
        address(6,1) = addr(accum_AFUDC_dep);
        address(7,1) = addr(net_capitalized_AFUDC);
        address(8,1) = addr(total_assets);
/*****************************************************************************
*  An address of 0 is used to tell print routine to skip a line.             *
*****************************************************************************/
        unspec(address(9,1)) = 0;
/*****************************************************************************
*  An address of 1 is used to tell print routine to center and print header. *
*****************************************************************************/
        unspec(address(10,1)) = '0001'b4;
```

```
        address(11,1) = addr(LT_debt_repayment(2));
        address(12,1) = addr(LT_debt_outstanding);
        address(13,1) = addr(deferred_ITC);
        address(14,1) = addr(cum_deferred_FIT);
        address(15,1) = addr(preferred_stock_repayment(2));
        address(16,1) = addr(preferred_stock_outstanding);
        address(17,1) = addr(end_common_equity);
        address(18,1) = addr(cum_partner_loans);
        address(19,1) = addr(total_equity);
        address(20,1) = addr(total_assets);
/*****************************************************************************
*  Initialization of INCOME STATEMENT pointers                              *
*****************************************************************************/
        index(2) = 22;
        address(1,2) = addr(electric_sales);
        address(2,2) = addr(Federal_subsidies);
        address(3,2) = addr(total_revenues);
        unspec(address(4,2)) = '0001'b4;
        address(5,2) = addr(fuel_cost);
        address(6,2) = addr(operation_maintanence);
        address(7,2) = addr(backfit_replacement);
        address(8,2) = addr(ad_valorem_tax);
        address(9,2) = addr(total_O_and_M);
        address(10,2) = addr(book_dep_plant_year);
        address(11,2) = addr(book_dep_AFUDC_year);
        address(12,2) = addr(total_book_dep_year);
        address(13,2) = addr(expenses.property_tax);
        address(14,2) = addr(net_FIT_book);
        address(15,2) = addr(total_pre_debt_expense);
        address(16,2) = addr(total_pre_debt_income);
        address(17,2) = addr(LT_debt_interest);
        address(18,2) = addr(net_income);
        address(19,2) = addr(preferred_stock_div);
        address(20,2) = addr(common_earnings);
        address(21,2) = addr(busbar_costs);
/*****************************************************************************
*  Initialize SOURCES AND USES OF FUNDS STATEMENT                            *
*****************************************************************************/
        index(3) = 45;
        address(1,3) = addr(net_income);
        address(2,3) = addr(book_dep_plant_year);
        address(3,3) = addr(book_dep_AFUDC_year);
        address(4,3) = addr(total_book_dep_year);
        address(5,3) = addr(amort_ITC);
        address(6,3) = addr(deferred_FIT);
        address(7,3) = addr(total_internal);
        address(8,3) = addr(partner_loans);
        address(9,3) = addr(DOE_loans);
        address(10,3) = addr(total_sources);
        unspec(address(11,3)) = 0;
        unspec(address(12,3)) = '0001'b4;
        address(13,3) = addr(preferred_stock_repayment);
        address(14,3) = addr(preferred_stock_div);
        address(15,3) = addr(common_stock_div);
```

```
        address(16,3) = addr(delta_working_capital);
        address(17,3) = addr(LT_debt_repayment);
        address(18,3) = addr(partner_loan_repayment);
        address(19,3) = addr(total_sources);
/*****************************************************************************
*  Initialize RETAINED EARNINGS STATEMENT & TAX DEP. WORKSHEET               *
*****************************************************************************/
        index(4) = 65;
        unspec(address(1,4)) = 0;
        address(2,4) = addr(beginning_common_equity);
        address(3,4) = addr(common_earnings);
        address(4,4) = addr(total_common_equity);
        unspec(address(5,4)) = 0;
        address(6,4) = addr(common_stock_div);
        address(7,4) = addr(end_common_equity);
        address(8,4) = addr(rates.common_equity);
        unspec(address(9,4)) = 0;
        unspec(address(10,4)) = '0001'b4;
        unspec(address(11,4)) = 0;
        address(12,4) = addr(startup_dep);
        address(13,4) = addr(five_year_property);
        address(14,4) = addr(ten_year_property);
        address(15,4) = addr(fifteen_year_property);
        address(16,4) = addr(other_dep_year);
        address(17,4) = addr(total_tax_dep);
/*****************************************************************************
*  Initialize TAX CALCULATION (Book/Regulation Basis)                        *
*****************************************************************************/
        index(5) = 80;
        address(1,5) = addr(electric_sales);
        address(2,5) = addr(total_O_and_M);
        address(3,5) = addr(book_depreciation_year);
        address(4,5) = addr(expenses.property_tax);
        address(5,5) = addr(LT_debt_interest);
        address(6,5) = addr(pre_tax_income_book);
        address(7,5) = addr(FIT_book);
        address(8,5) = addr(amort_ITC_book_year);
        address(9,5) = addr(net_FIT_book);
        unspec(address(10,5)) = '0001'b4;
        address(11,5) = addr(electric_sales);
        address(12,5) = addr(total_O_and_M);
        address(13,5) = addr(total_tax_dep);
        address(14,5) = addr(expenses.property_tax);
        address(15,5) = addr(LT_debt_interest);
        address(16,5) = addr(pre_tax_income_FERC);
        address(17,5) = addr(FIT_tax_FERC);
        address(18,5) = addr(credit_ITC);
        address(19,5) = addr(deferred_FIT_taken);
        address(20,5) = addr(net_FIT_FERC);
/*****************************************************************************
*  Initialize PARTNERSHIP RETURN AND CASHFLOW ANALYSIS                       *
*****************************************************************************/
        index(6) = 102;
```

```
        address(1,6) = addr(electric_sales);
        address(2,6) = addr(total_O_and_M);
        address(3,6) = addr(total_tax_dep);
        address(4,6) = addr(expenses.property_tax);
        address(5,6) = addr(LT_debt_interest);
        address(6,6) = addr(pre_tax_income_FERC);
        address(7,6) = addr(FIT_tax_FERC);
        unspec(address(8,6)) = 0;
        unspec(address(9,6)) = '0001'b4;
        address(10,6) = addr(after_tax_cash);
        address(11,6) = addr(total_tax_dep);
        address(12,6) = addr(Federal_subsidies);
        address(13,6) = addr(total_cash_avail);
        address(14,6) = addr(LT_debt_repayment);
        address(15,6) = addr(preferred_stock_repayment);
        address(16,6) = addr(preferred_stock_div);
        address(17,6) = addr(partner_loans);
        address(18,6) = addr(delta_working_capital);
        address(19,6) = addr(net_to_partners);
        address(20,6) = addr(cum_to_partners);
        index(7)= 123;
/*****************************************************************************
*  Calculate as-spent and base-year discount rates and levelized busbar costs.*
*****************************************************************************/
        st_construction = start_construction;
        year_end = end_construction;
        construction_period = year_end - st_construction;
        cum_inflation = 1;
        iyear = 1;
        number = st_construction - base_year;
        if pre_after_tax_flag = 0 then discount_factor = 1;
        else discount_factor = 1. - fit_tax;
        iyear = year_end - base_year + 1;
        jyear = iyear;
        if jyear > 50 then jyear = 50;
        if iyear > 30 then iyear = 30;
        discount_rate = 1./(1. + LT_debt_rate*
        debt_fraction*discount_factor + preferred_rate*(1. - debt_fraction
        -common_equity_fraction)+target_ROE*common_equity_fraction);
        discount_rate_real = discount_rate*(1. + inflation(iyear));
        iyear = 1;
            do i = 1 to number;
            cum_inflation = cum_inflation*(1. + inflation(iyear));
            if iyear < 30 then iyear = iyear + 1;
            end;
        cum_inflation_save = cum_inflation;
            do i = 1 to construction_period;
            cum_inflation = cum_inflation*(1. + inflation(iyear));
            if iyear < 30 then iyear = iyear + 1;
            end;
        cum_kWh = 0.;
            do i = 1 to analysis_period;
            cum_inflation = cum_inflation*(1. + inflation(iyear));
            if iyear < 30 then iyear = iyear + 1;
```

```
    cents_per_kWh_real(i) = busbar_costs(i)/cum_inflation;
    if i<=levelization_period then cum_kWh=cum_kWh+capacity_factor(i);
    atcf_real(i) = total_O_and_M(i)/cum_inflation;
    atcf_cur(i) = electric_sales(i)/cum_inflation;
    end;
cum_kWh = cum_kWh*size*8.76e-02/levelization_period;
call levelize(discount_rate,level_busbar_cur,addr(electric_sales));
call levelize(discount_rate_real,level_busbar_real,addr(atcf_cur));
call levelize(discount_rate,fixed_charge_cur,addr(total_O_and_M));
call levelize(discount_rate_real,fixed_charge_real,addr(atcf_real));
fixed_charge_cur = (level_busbar_cur - fixed_charge_cur)/
(0.01*(construction + AFUDC + plant_variables.working_capital));
fixed_charge_real = (level_busbar_real - fixed_charge_real)/
(0.01*(construction_real + AFUDC_real));
level_busbar_cur = level_busbar_cur/cum_kWh;
level_busbar_real = level_busbar_real/cum_kWh;
call clearscreen;
put file(crt) skip edit('You may now list various CASHFLOW reports')(a);
return;
end initial;
```

```
/*****************************************************************************
*  S U M A R Y :                                                            *
* This routine prints a summary of program inputs, run options, and         *
* key output results.  The following inputs are passed:                     *
*    x          - file type (crt/printer)                                   *
*    rate_cur   - discount rate in current $                                *
*    rate_real  - discount rate in real $                                   *
*    busbar_cur - levelized busbar cost in current $                        *
*    busbar_real- levelized busbar cost in base year $                      *
*    roi_cur    - ROI in current $                                          *
*    roi_real   - ROI in base year $                                        *
*                                                                           *
*   Original version 1.1  9/15/83                                           *
*   Version 1.2  10/10/83                                                   *
*      Include STARTUP COSTS, WORKING CAPITAL, and TARGET ROI               *
*      Simplify print logic                                                 *
*      Deduct startup_cost from tax base (rate_tot)                         *
*   Version 1.3  10/20/83                                                   *
*      Include book_life in startup summary printout                        *
*  Version 1.4  11/10/83                                                    *
*     Calulate debt/equity ratios explicitly                                *
*     Replace GROSS_NET_AFUDC option with NET_EQUITY option                 *
*  Version 1.5  12/15/83                                                    *
*     Include summary for new options:                                      *
*       - Flow through vs. normalize                                        *
*       - Consolidated vs. unconsolidated                                   *
*     Include BASE YEAR value                                               *
*     Change clearscreen to a function call to provide compatability between *
*     CP/M and IBM/PC-DOS versions.                                         *
*****************************************************************************/
sumary:
    procedure(x,rate_cur,rate_real,busbar_cur,busbar_real,
    fixed_charge_cur,fixed_charge_real,roi_cur,roi_real);

    %include 'input.dcl';
    %include 'constr.dcl';
    %include 'diomod.dcl';

    dcl (x,crt,printer) file,
        (rate_tot,rate_cur,rate_real,busbar_cur,busbar_real,roi_cur,roi_real
        ,fixed_charge_cur,fixed_charge_real) float,
        buffer char(79) var, continue char(1),
        find_record entry(fixed,fixed,char(79) var),
        screen_clear entry(char(1),fixed),
        center entry(file,fixed,fixed,char(79) var),
        (skip,jmod,buff_length) fixed;

    if x = crt then call clearscreen;
    else put file(x) skip edit(new_page) (a);
    skip = 0;
    if x = printer
        then do;
        buff_length = length(header);
        call center(x,buff_length,2,header);
        skip = 2;
```

```
/*****************************************************************************
*  Open data file SUMMARY.DAT and list summary output.                      *
*****************************************************************************/
    buffer = 'summary';
    call find_record(buff_length,0,buffer);
    call find_record(buff_length,1,buffer);
    if x = crt then skip = 0;
    call center(x,buff_length,skip,buffer);
    if x = crt then skip = 1;
    call find_record(buff_length,2,buffer);
    if x = printer then call center(x,buff_length,1,buffer);
    call find_record(buff_length,3,buffer);
    buff_length = buff_length + 10;
    call center(x,buff_length,skip,buffer);
    put file(x) edit(start_construction,',',end_construction,')') (f(4,0),a);
    buffer = 'Base Year:';
    buff_length = length(buffer) + 5;
    call center(x,buff_length,1,buffer);
    put file(x) edit(base_year) (f(5,0));
    call find_record(buff_length,4,buffer);
    call center(x,buff_length,skip,buffer);
    call find_record(buff_length,5,buffer);
    call center(x,buff_length,1,buffer);
    call find_record(jmod,6,buffer);
    call center(x,buff_length,1,buffer);
    rate_tot = startup_cost/book_life;
    put file(x) edit(construction-startup_cost,book_life,book_dep_plant
    - rate_tot) (f(7,1),f(9,0),f(15,1));
    call find_record(jmod,7,buffer);
    call center(x,buff_length,1,buffer);
    put file(x) edit(startup_cost,book_life,rate_tot) (f(7,1),f(9,0),f(15,1));
    call find_record(jmod,8,buffer);
    call center(x,buff_length,1,buffer);
    put file(x) edit(AFUDC,book_life,AFUDC_dep_plant) (f(7,1),f(9,0),f(15,1));
    call find_record(jmod,9,buffer);
    call center(x,buff_length,1,buffer);
    put file(x) edit(plant_variables.working_capital,'--','--') (f(7,1),
    x(7),a,x(13),a);
    call find_record(buff_length,5,buffer);
    call center(x,buff_length,1,buffer);
    call find_record(jmod,10,buffer);
    call center(x,buff_length,1,buffer);
    put file(x) edit(construction+AFUDC+plant_variables.working_capital,
    '       --',book_dep_plant+AFUDC_dep_plant) (f(7,1),a,f(15,1));
    call find_record(buff_length,5,buffer);
    call center(x,buff_length,1,buffer);
    call find_record(buff_length,11,buffer);
    call center(x,buff_length,skip,buffer);
    call find_record(buff_length,12,buffer);
    call center(x,buff_length,1,buffer);
    call find_record(jmod,13,buffer);
    call center(x,buff_length,1,buffer);
    rate_tot = 100./(construction + AFUDC + plant_variables.working_capital);
    put file(x) edit(plant totals.common equity,plant totals.common equity*
```

```
rate_tot,100*target_ROE) (f(8,1),f(12,2),f(15,2));
call find_record(jmod,14,buffer);
call center(x,buff_length,1,buffer);
put file(x) edit(plant_totals.preferred_stock,
plant_totals.preferred_stock*rate_tot,100*preferred_rate)
(f(8,1),f(12,2),f(15,2));
call find_record(jmod,15,buffer);
call center(x,buff_length,1,buffer);
put file(x) edit(plant_totals.LT_debt,plant_totals.LT_debt*rate_tot,
100*LT_debt_rate) (f(8,1),f(12,2),f(15,2));
call find_record(buff_length,12,buffer);
call center(x,buff_length,1,buffer);
call find_record(jmod,16,buffer);
call center(x,buff_length,1,buffer);
if pre_after_tax_flag = 0 then rate_tot = rate_cur; else rate_tot =
rate_cur + 100.*FIT_tax*LT_debt_rate*debt_fraction;
put file(x) edit(construction+AFUDC+plant_variables.working_capital,
'100.00',rate_tot) (f(8,1),x(6),a,f(15,2));
call find_record(buff_length,12,buffer);
call center(x,buff_length,1,buffer);
call find_record(buff_length,17,buffer);
call center(x,buff_length,skip,buffer);
call find_record(jmod,18,buffer);
call center(x,buff_length,1,buffer);
put file(x) edit(ITC_accrual) (f(8,1));
call find_record(jmod,19,buffer);
call center(x,buff_length,1,buffer);
put file(x) edit(tax_accrual) (f(8,1));
if x = crt
    then do;
    call screen_clear(continue,1);
    if continue = escape
        then do;
        call find_record(buff_length,-1,buffer);
        return;
        end;
    end;
else put file(x) skip(5);
call find_record(buff_length,20,buffer);
call center(x,buff_length,0,buffer);
call find_record(buff_length,21,buffer);
call center(x,buff_length,skip,buffer);
call find_record(jmod,22,buffer);
call center(x,buff_length,1,buffer);
put file(x) edit(tax_life,' (years)') (f(3,0),a);
call option(x,23,accelerated_dep_flag);
call find_record(jmod,26,buffer);
call center(x,buff_length,1,buffer);
if tefra_flag = 1 then rate_tot = construction - startup_cost;
else rate_tot = 0.95*(construction - startup_cost);
put file(x) edit(rate_tot) (f(7,1));
call option(x,27,tefra_flag);
call print options;
```

```
/*****************************************************************************
*  Print summary of ROIs, busbar costs, discount rates, and fixed charge rates*
*****************************************************************************/
    call find_record(buff_length,30,buffer);
    call center(x,buff_length,skip,buffer);
    call find_record(jmod,31,buffer);
    call center(x,buff_length,1,buffer);
    call find_record(buff_length,32,buffer);
    call center(x,buff_length,1,buffer);
    call find_record(jmod,33,buffer);
    call center(x,buff_length,1,buffer);
    put file(x) edit(roi_cur,busbar_cur,rate_cur,fixed_charge_cur)
    (2f(13,2),f(17,2),f(18,2));
    call find_record(jmod,34,buffer);
    call center(x,buff_length,1,buffer);
    put file(x) edit(roi_real,busbar_real,rate_real,fixed_charge_real)
    (2f(13,2),f(17,2),f(18,2));
    if x = crt then call screen_clear(continue,1);
        else do;
        call find_record(buff_length,32,buffer);
        call center(x,buff_length,1,buffer);
        end;
/*****************************************************************************
*  Close data file SUMMARY.DAT.                                              *
*****************************************************************************/
    call find_record(buff_length,-1,buffer);
    return;

/*****************************************************************************
*  P R I N T _ O P T I O N S :                                               *
*  This routine prints a summary of program options in effect.               *
*****************************************************************************/
    print_options:
        procedure;

        call find_record(buff_length,35,buffer);
        call center(x,buff_length,skip,buffer);
        buff_length = 30;
        call option(x,36,compound_simple_flag);
        call option(x,39,quarter_annual_flag);
        call option(x,42,gross_net_flag);
        call option(x,45,net_equity_flag);
        call option(x,48,pre_after_tax_flag);
        call option(x,51,bottom_up_top_down_flag);
        call option(x,54,flowthrough_normalize_flag);
        call option(x,57,consolidated_tax_flag);
        return;
        end print_options;

/*****************************************************************************
*  O P T I O N :                                                             *
*  This routine prints the LABEL for flag FLAG and corresponding option      *
* LABEL on output device.  I points to the position of the FLAG LABEL in     *
* data file SUMMARY.DAT.  I+1 and I+2 records contain option LABEL for       *
* FLAG = 0 and FLAG = 1 respectively.                                        *
*****************************************************************************/
```

```
option:
    procedure(x,i,flag);

    dcl x file,
        i fixed,
        flag float;

    call find_record(jmod,i,buffer);
    call center(x,buff_length,1,buffer);
    if flag = 1 then jmod = i + 2; else jmod = i + 1;
    call find_record(jmod,jmod,buffer);
    put file(x) edit(buffer) (x(1),a);
    return;
    end;

end sumary;
```

```
/*****************************************************************************
*  C A L R O I :                                                             *
* This routine calculates ROIs based upon after-tax cashflow to the          *
* partners.  An ROI is calculated for each year of operation based           *
* upon cashflows to date.  Routine inputs include:                           *
*    Start_construction - Year construction starts                           *
*    Years_cash_flows   - Total number of cashflow years                     *
*    Years_rois         - Number of years for ROI loop                       *
*    Cash_flows(45)     - Cashflow enteries                                  *
*    Rois(30)           - Calculated ROIs                                    *
*                                                                            *
*   Original version 1.1  9/15/83                                            *
*   Version 1.2  10/10/83                                                    *
*      Include on-screen printout of current ROI loop index                  *
*      Modify error message printouts                                        *
*  Version 1.3  10/20/83                                                     *
*     Modify convergence algorithm.  If delta changes sign average return    *
*  Version 1.4  11/10/83                                                     *
*     Protect against all CASHFLOWS being the same sign.                     *
*****************************************************************************/
calroi:
    procedure(start_construction,years_cash_flows,years_rois,cash_flows,rois);

    %include 'diomod.dcl';

    dcl (max,min,cash_flows(45),derivs(45),rois(30),sum,delta,value,
        deriv,return,return_old,delta_last) float,
        (error_flag,nchange,ipos,stop,start,years_cash_flows,nconst,i,j,k,
        years_rois,start_construction) fixed,
        min_return float static init('.1'),
        max_return float static init('10.'),
        continue char(1),
        screen_clear entry(char(1),fixed),
        crt file;


    error_flag = 0;
/*****************************************************************************
* Initialize routine.  Calculate Net-Present_Value (NPV) derivatives, check  *
* for range of validity (i.e. single-valued), and select reference year.     *
*****************************************************************************/
    call do_initial;
/*****************************************************************************
* Calculate annual ROIs.                                                     *
*****************************************************************************/
    if ipos = 0 then call calculate;
    if error_flag = 0 then call screen_clear(continue,1);
    return;

/*****************************************************************************
*  D O _ I N I T I A L :                                                     *
*  This routine calculates partial sums (NPV for o discount rate), selects   *
* base year, and calculates NPV derivatives.                                 *
*****************************************************************************/
```

```
do_initial:
    procedure;

    nchange = 0;
    sum = 0;
    value = cash_flows(1);
    ipos = 0;
    start = 0;
    stop = 0;
/*****************************************************************************
* Calculate partial sums and partial-sum sign changes.  Solution guaranteed  *
* for one partial sum sign change only.  Convergence is improved by choosing *
* base year of NPV calculation corresponding to year of 1st positive partial *
* sum.                                                                       *
*****************************************************************************/
        do i = 1 to years_cash_flows;
        sum = sum + cash_flows(i);
        if sum*value < 0
            then do;
            value = sum;
            if nchange = 0 then start = i;
            if nchange = 1 then stop = i;
            nchange = nchange + 1;
            end;
        if cash_flows(i) > 0 & ipos = 0 then ipos = i;
        end;
    if nchange = 1 then stop = years_cash_flows + 1;
    else do;
    put file(crt) skip edit(bell,'Warning. ',nchange,
    ' Partial sum sign changes.') (a,a,f(2,0));
    put file(crt) skip edit('Analysis only valid for: (',start_construction
    +start-1,',',start_construction+stop-1,')') (a,f(4,0));
    error_flag = 1;
    if nchange = 0 then ipos = 0;
    end;
/*****************************************************************************
* Calculate derivative coefficients relative to discount rate and base year. *
*****************************************************************************/
    do i = 1 to years_cash_flows;
    derivs(i) = (ipos - i)*cash_flows(i);
    end;
    if ipos = 0
        then do;
            do i = 1 to years_rois;
            rois(i) = -1000.;
            end;
        end;
    return;
    end do_initial;

/*****************************************************************************
*  C A L C U L A T E :                                                       *
*  Calculate ROI for each year using a Newton-Ralphson method.               *
*****************************************************************************/
```

```
    calculate:
        procedure;

            nconst = years_cash_flows - years_rois;
            return = 1;
            put file(crt) skip(2);
            do j = 1 to years_rois;
            put file(crt) skip(0) edit('Calculating ROI for year: ',j)
            (a,f(2,0));
            delta = 1.e38;
            delta_last = delta;
            min = min_return;
            max = max_return;
            k = 0;
                do while(abs(delta) > abs(return*.0001));
                value =  0.;
                deriv = 0.;
/*****************************************************************************
* Calculate NPV and derivative for each year.                                *
*****************************************************************************/
                    do i = 1 to j + nconst;
                    value = (value + cash_flows(i))*return;
                    deriv = (deriv + derivs(i))*return;
                    end;
                deriv = deriv/return;
                if deriv= 0
                    then do;
                    delta = -value/deriv;
                    if delta*delta_last < 0 & delta_last = 1.e38
                        then do;
/*****************************************************************************
* Overshot solution.  Take a cut step to prevent oscillation.                *
*****************************************************************************/
                        if delta < 0 then max = return; else min = return;
                        if delta_last < 0 then return_old = min;
                        else return_old = max;
                        value = (return*abs(delta_last) + return_old*
                        abs(delta))/(abs(delta) + abs(delta_last));
                        end;
                        else value = return + delta;
                    if delta > 0 then if return = max_return then call error;
                        else if value > max_return then value = max_return;
                        else min = return;
                    else if return = min_return then call error;
                        else if value < min_return then value = min_return;
                        else max = return;
                    if value > max ! value < min then return = (max + min)*.5;
                    else return = value;
                    delta_last = delta;
                    end;
                    else k = 20;
                k = k + 1;
```

```
            if k>=20
                then do;
                put file(crt) skip edit(bell,'ROI Convergence Failure for '
                ,start_construction+nconst+j-1)
                (col(35),a,a,f(4,0));
                delta = 0.;
                return = 1;
                error_flag = 1;
                end;
            end;
        if k < 20 then rois(j) = (return - 1)*100;
        else if return < 1 then rois(j) = -1000; else rois(j) = 2000;
        end;
    return;
    end calculate;

/*****************************************************************************
*  E R R O R :                                                               *
*  This routine prints an error message for ROIs out of range.               *
*****************************************************************************/
    error:
        procedure;

        delta = 0;
        error_flag = 1;
        put file(crt) skip edit(bell,'Invalid ROI: <-90% or >900% for year ',
        start_construction + nconst + j - 1) (col(35),a,a,f(4,0));
        return;
        end error;

    end calroi;
```

```
/*****************************************************************************
*  P L O T B U S B A R :                                                    *
* This routine plots real and current $ busbar costs.  Routine inputs       *
* include:                                                                  *
*    X              - print file (crt/printer)                              *
*    Start_plot     - initial year for plot                                 *
*    Number_points  - number of points to be plotted                        *
*    Array1         - current $ busbar costs                                *
*    Array2         - base year $ busbar costs                              *
*                                                                           *
*   Original version 1.1  9/15/83                                           *
*****************************************************************************/
plotbusbar:
    procedure(x,start_plot,number_points,array1,array2);

    %include 'diomod.dcl';

    dcl (x,crt) file,
        (j,start_plot,j1,j2,i,k,number_points,number_plot,start_axis) fixed,
        (bin,value,max_value,delta,array1(30),array2(30)) float,
        screen_clear entry(char(1),fixed),
        buffer char(71),
        yaxis char(5) static init('YEARS'),
        (label,star,plus,blank) char(1);

/*****************************************************************************
* Initialize routine.                                                       *
*****************************************************************************/
    call initial;
    if x = crt & number_points > 15 then  number_plot = 15; else
    number_plot = number_points;
/*****************************************************************************
*  Do plot.                                                                 *
*****************************************************************************/
    call plot(1,number_plot);
    if x = crt & number_points > 15 & blank = escape
        then do;
/*****************************************************************************
*  Split the plot into two parts for screen display.                        *
*****************************************************************************/
        call clearscreen;
        number_plot = number_plot + 1;
        call header;
        call plot(number_plot,number_points);
        end;
    return;

/*****************************************************************************
*  I N I T I A L :                                                          *
*  This routine initializes the plot variables and finds the maximum plot   *
* value.  It also prints the plot HEADER.                                   *
*****************************************************************************/
    initial:
        procedure;
```

```
      max_value = 0.;
          do i = 1 to number_points;
          if array1(i) > max_value then max_value = array1(i);
          end;
      if max_value < 25 then max_value = 25;
      else if max_value < 50 then max_value = 50;
      else if max_value < 75 then max_value = 75;
      else if max_value < 100 then max_value = 100;
      delta = max_value/10.;
      bin = 70./max_value;
      buffer = '|';
      buffer = substr(buffer,1,70) || '|';
      star = '*';
      plus = 'X';
      blank = ' ';
      call header;
      return;
      end initial;

/*****************************************************************************
*  H E A D E R :                                                            *
* This routine prints the plot header and x-axis labels.                    *
*****************************************************************************/
    header:
        procedure;

        put file(x) skip edit('Annual Busbar Costs') (col(31),a);
        put file(x) skip edit((value do value= 0 to max_value by delta))
        (x(2),11(f(6,1),x(1)));
        put file(x) skip edit('|------',('+------' do i = 1 to 9),'|')
        (col(7),11a);
        return;
        end header;

/*****************************************************************************
*  P L O T :                                                                *
*  This routine the data for ARRAY values START to STOP.                    *
*****************************************************************************/
    plot:
        procedure(start,stop);

        dcl (start,stop) fixed;

        blank = ' ';
        start_axis = divide(stop - start + 1,2,15) - 3;
            do k = start to stop;
            j1 = array1(k)*bin + 0.5;
            if j1 > 0 & j1 < 70 then buffer = substr(buffer,1,j1) ||
            star || substr(buffer,j1+2);
            j2 = array2(k)*bin + 0.5;
            if j2 > 0 & j2 < 70 then buffer = substr(buffer,1,j2) ||
            plus || substr(buffer,j2+2);
            j = k - start axis;
```

```
        if j>0 & j<6 then label = substr(yaxis,j,1); else label = blank;
        put file(x) skip edit(label,start_plot+k-1,buffer) (a,f(5,0));
        if j1 > 0 & j1 < 70 then buffer = substr(buffer,1,j1) ||
        blank || substr(buffer,j1+2);
        if j2 > 0 & j2 < 70 then buffer = substr(buffer,1,j2) ||
        blank || substr(buffer,j2+2);
        end;
    put file(x) skip edit('|------',('+------' do i = 1 to 9),'|')
    (col(7),11a);
    put file(x) skip edit((value do value = 0 to max_value by delta))
    (x(2),11(f(6,1),x(1)));
    put file(x) skip edit('cents/kWh (X - Base Year,* - Current)')
    (col(25),a);
    if x = crt then call screen_clear(blank,1);
    return;
    end plot;

end plotbusbar;
```

```
/*****************************************************************************
*  G E T _ I N T E G E R :                                                   *
* This routine inputs a integer type variable from the console.  It traps    *
* all user input errors.  It uses subroutine GET_STRING using the direct read *
* /write routines CONINP and CONOUT.  A carriage return is equivalent to 0.  *
* Upon return the ESCAPE_FLAG signifies:                                     *
*     -1 input of ESCAPE character                                           *
*      0 normal integer input                                                *
*      1 user input error                                                    *
*                                                                            *
*     Original version  8/5/83                                               *
*     Version 1.5   4/15/84 changed to provide capatability with IBM/PC.     *
*****************************************************************************/
get_integer:
    procedure(escape_flag,value);

    %include 'diomod.dcl';

    dcl crt file,
        get_string entry(char(79) var),
        buffer char(79) var,
        (value,escape_flag) fixed;

/*****************************************************************************
*  This code traps CONVERSION errors and detects input of ESCAPE key.        *
*  If ESCAPE key input, ESCAPE_FLAG is set to -1 and VALUE = 0.  If          *
*  an input error is detected, ESCAPE_FLAG is set to 1 and VALUE = -1.       *
*****************************************************************************/
    on error(1)
        begin;
        if buffer = escape
            then do;
            escape_flag = -1;
            value = 0;
            end;
            else do;
            escape_flag = 1;
            value = -1;
            end;
        put file(crt) edit(bell) (a);
/*****************************************************************************
*  A transfer of control, not RETURN, is required.                           *
*****************************************************************************/
        go to retry;
        end;

    call get_string(buffer);
/*****************************************************************************
* An invalid integer string in BUFFER will result in a conversion error      *
* (ERROR(1)).                                                                *
*****************************************************************************/
    value = buffer;
    escape_flag = 0;
retry: return;
    end get_integer;
```

```
/*****************************************************************************
*  G E T _ F L O A T :                                                       *
* This routine inputs a float type variable from the console.  It traps      *
* all user input errors.  It uses subroutine GET_STRING using the direct read *
* /write routines CONINP and CONOUT.  A carriage return is equivalent to 0.  *
* Upon return the ESCAPE_FLAG signifies:                                     *
*     -1 input of ESCAPE character                                           *
*      0 normal integer input                                                *
*      1 user input error                                                    *
*                                                                            *
*     Original version  8/5/83                                               *
*     Version 1.5  4/15/84  Revised to make compatible with IBM/PC.          *
*****************************************************************************/
get_float:
    procedure(escape_flag,value);

    %include 'diomod.dcl';

    dcl crt file,
        get_string entry(char(79) var),
        buffer char(79) var,
        escape_flag fixed,
        value float;

/*****************************************************************************
*  This code traps CONVERSION errors and detects input of ESCAPE key.        *
*  If ESCAPE key input, ESCAPE_FLAG is set to -1 and VALUE = 0.  If          *
*  an input error is detected, ESCAPE_FLAG is set to 1 and VALUE = -1.       *
*****************************************************************************/
    on error(1)
        begin;
        if buffer = escape
            then do;
            escape_flag = -1;
            end;
            else do;
            escape_flag = 1;
            end;
        put file(crt) edit(bell) (a);
/*****************************************************************************
*  A transfer of control, not RETURN, is required.                           *
*****************************************************************************/
        go to retry;
        end;

    call get_string(buffer);
/*****************************************************************************
* An invalid float string in BUFFER will result in a conversion error        *
* (ERROR(1)).                                                                *
*****************************************************************************/
    value = buffer;
    escape_flag = 0;
retry: return;
    end get_float;
```

```
/*****************************************************************************
*   G E T _ S T R I N G :                                                    *
* This routine uses the direct read/write routines CONINP and CONOUT to read *
* and echo characters at the CRT.  Processing ends upon receipt of either a  *
* CARRIAGE RETURN or an ESCAPE character.  The string is returned in BUFFER. *
*   Version 1.5  4/15/84   This is the original version.                     *
*****************************************************************************/
get_string:
    procedure(buffer);

    %include 'diomod.dcl';

    dcl buffer char(79) var,one_char char(1);

    buffer = '';
    one_char = ' ';
        do while(one_char=carriage_return&one_char=escape);
        one_char = coninp();
        if one_char = backspace then if length(buffer) > 0
            then do;
/*****************************************************************************
* BACKSPACE detected:                                                        *
*   - Delete last character from input buffer.                               *
*   - backspace cursor.                                                      *
*   - print a blank character.                                               *
*   - backspace cursor.                                                      *
*****************************************************************************/
            buffer = substr(buffer,1,length(buffer)-1);
            call conout(backspace);
            call conout(' ');
            call conout(backspace);
            end;
        else;
        else if one_char = escape then buffer=escape;
        else if one_char = carriage_return
            then do;
/*****************************************************************************
* CARRIAGE RETURN detected.                                                  *
*   - Echo CARRIAGE RETURN.                                                  *
*   - Print LINE FEED.                                                       *
*****************************************************************************/
            call conout(carriage_return);
            call conout(line_feed);
            end;
            else do;
/*****************************************************************************
* Add character to input buffer and echo character to CRT.                  *
*****************************************************************************/
            buffer = buffer || one_char;
            call conout(one_char);
            end;
        end;
    return;
    end get string;
```

```
/*****************************************************************************
*  F I N D _ R E C O R D :                                                   *
* This routine finds record REC_NUMBER from a data file created by the       *
* program KEYWRITE.  The record in returned in PUT_BUFFER.  REC_TYPE         *
* the record type (used only in INPUT) or the record length for type 0       *
* records.  The routine is initialized by call with REC_NUMBER = 0 and       *
* PUT_BUFFER containing the file name (.DAT is always appended).  The        *
* routine in closed by calling with REC_NUMBER = -1.                         *
*                                                                            *
*     Original version 8/5/83                                                *
*****************************************************************************/
find_record:
    procedure(rec_type,rec_number,put_buffer);

    dcl put_buffer char(79) var,
        (rec_number,max_number,rec_type,rec_key,rec_start,rec_length) fixed,
        input file,
        input_buffer char(128),
        buff_add pointer;

    dcl 1 buff(128),
        2 key fixed(7),
        2 start fixed(7),
        2 length_rec fixed(7),
        2 type_rec fixed(7);

    dcl 1 record_buffer based(buff_add),
        2 string char(128);

    if rec_number = 0
        then do;
/*****************************************************************************
* Program Initialization (REC_NUMBER = 0):                                   *
*   - Open input file (PUT_BUFFER.DAT)                                       *
*   - Read in keys which follow text records (maximum of 128 records)        *
*   - There are 32 keys/key input record.                                    *
*   - Determine number of text records in data file (MAX_NUMBER)             *
*****************************************************************************/
        open file(input) direct env(f(128)) title(put_buffer || '.dat');
        rec_key = 0;
        key(1) = 1;
        buff_add = addr(buff(1));
            do while(key(1) = 0);
            read file(input) into(record_buffer) key(rec_key);
            rec_key = rec_key + 1;
            end;
        rec_number = 1;
        max_number = 1;
            do while(key(max_number) = -1);
            rec_number = rec_number + 1;
            max_number = max_number + 1;
```

```
            if rec_number > 32
                then do;
                rec_number = 1;
                buff_add = addr(buff(max_number));
                read file(input) into(record_buffer) key(rec_key);
                rec_key = rec_key + 1;
                end;
            end;
        max_number = max_number - 1;
        rec_number = 0;
        end;
    else if rec_number < 0 then close file(input);
/*****************************************************************************
* REC_NUMBER < 0.  Close input data file.                                    *
*****************************************************************************/
    else if rec_number > max_number
        then do;
/*****************************************************************************
* REC_NUMBER > MAX_NUMBER.  Print error message on CRT.                      *
*****************************************************************************/
        put_buffer = 'Attempt to read beyond end of file';
        rec_type = -1;
        end;
        else do;
/*****************************************************************************
* 0 < REC_NUMBER <= MAX_NUMBER.  Find text record REC_NUMBER, place          *
* in PUT_BUFFER, and return REC_TYPE.                                        *
*****************************************************************************/
        rec_key = key(rec_number);
        read file(input) into(input_buffer) key(rec_key);
        rec_start = start(rec_number);
        if rec_start < 0 then rec_start = 128;
        rec_length = length_rec(rec_number) - 2;
        put_buffer = substr(input_buffer,rec_start,rec_length);
        if rec_start + rec_length > 128
            then do;
            rec_key = rec_key + 1;
            read file(input) into(input_buffer) key(rec_key);
            put_buffer = put_buffer ||
            substr(input_buffer,1,rec_length+rec_start-129);
            end;
        if type_rec(rec_number)=0 then rec_type=rec_length;
        else rec_type=type_rec(rec_number);
        end;
    return;
    end find record;
```

```
/*****************************************************************************
*  S R E E N _ C L E A R :                                                   *
*  This routine skips NSKIP lines; prints the STRIKE ANY KEY message; waits  *
* for a character (CONTINUE) from the keyboard, and clears the screen.       *
*  Version 1.5  4/15/84  Original version.                                   *
*****************************************************************************/
screen_clear:
    procedure(continue,nskip);

    %include 'diomod.dcl';

    dcl crt file,
        nskip fixed,
        continue char(1);

    put file(crt) skip(nskip) edit('<STRIKE ANY KEY TO CONTINUE>') (a);
    continue = coninp();
    if rank(continue) = 0 then continue = coninp();
    call clearscreen;
    return;
    end screen clear;
```

```
/*****************************************************************************
*  C E N T E R :                                                            *
*  This routine skips SKIP lines,centers and prints the string LABEL on     *
* output file X. BUFF_LENGTH is assumed to contain length of LABEL.         *
*****************************************************************************/
    center:
        procedure(x,buff_length,skip,label);

    dcl x file,
        label char(79) var,
        (buff_length,skip,i,j) fixed;

        j = divide(79 - buff_length,2,15);
        put file(x) skip(skip) edit((' ' do i = 1 to j),label) (a);
        return;
        end center;
```

```
/*****************************************************************************
*  K E Y W R I T E :                                                         *
*  This routine prepares specially keyed, variable-length records for use by *
* BREEDER.  The program takes as input a file of variable-length text records *
* as produced by any standard test editor.  To this file, KEYWRITE appends a *
* set of up to 128 keys to permit access to any of the text records directly *
* by record number (i.e. random access rather than sequential access) using *
* subroutine FIND_RECORD.  Each key entry contains four values:              *
*     - The KEY associated with the start of the text record,                *
*     - The offset to start of text record,                                  *
*     - The length of the text record, and                                   *
*     - An optional TYPE value which can be used to characterize the record. *
* The files: BREEDER.DAT, INPUT.DAT, CONSTR.DAT, REPORT.DAT, and SUMMARY.DAT *
* have been prepared using KEYWRITE and are used in BREEDER.                 *
*****************************************************************************/
keywrite:
    procedure options(main);

    %include 'diomod.dcl';

    dcl (crt,printer,output) file,
        find_record entry(fixed,fixed,char(79) var),
        screen_clear entry(char(1),fixed),
        get_integer entry(fixed,fixed),
        (continue,end_of_file) char(1),
        filename char(8) var,
        buffer char(128),
        put_buffer char(79) var,
        sum_buffer char(254) var,
        buff_add pointer,
        (file_type,max_number,line_count,rec_index,rec_key,rec_start,number,
        rec_length,hard_copy) fixed;

    dcl 1 buff(128),
        2 key fixed(7),
        2 start fixed(7),
        2 length_rec fixed(7),
        2 type fixed(7);

    dcl 1 record_buffer based(buff_add),
        2 string char(128);

    on endfile(output)
        begin;
        end_of_file = 't';
        key(max_number) = -1;
        start(max_number) = -1;
        length_rec(max_number) = -1;
        type(max_number) = -1;
        end;
```

```
/*****************************************************************************
* Initialize routine                                                         *
*****************************************************************************/
    call initial;
/*****************************************************************************
* Routine to process file                                                    *
*****************************************************************************/
    call process;
/*****************************************************************************
* Write out file with key array appended                                     *
*****************************************************************************/
    call done;
/*****************************************************************************
* Provide summary of file on CRT or printer                                  *
*****************************************************************************/
    call summary;

/*****************************************************************************
*  I N I T I A L :                                                           *
*  This subroutine intializes KEYWRITE.  It requests the name of the input   *
* file and zeros the array for the keys (128 entries).                       *
*****************************************************************************/
    initial:
        procedure;

        open file(crt) output stream title('$con');
        call clearscreen;
        put file(crt) skip(0) edit('Enter file name: ') (a);
        get list(filename);
        open file(output)input keyed env(f(128),b(1024))title(filename||'.DAT');
        hard_copy = 1;
            do while(hard_copy = 0);
            call clearscreen;
            put file(crt) skip edit('Enter 0 for all TEXT file: ') (a);
            call get_integer(hard_copy,file_type);
            end;
        end_of_file = 'f';
        continue = 't';
        sum_buffer = '';
        put file(crt) skip edit('Beginning processing of file: ',filename,
        '.DAT') (a);
        put file(crt) skip;
            do max_number = 1 to 128;
            key(max_number) = 0;
            start(max_number) = 0;
            length_rec(max_number) = 0;
            type(max_number) = 0;
            end;
        rec_start = 0;
        call get_record;
        max_number = 1;
        rec_start = 0;
        return;
        end initial;
```

```
/*****************************************************************************
*  P R O C E S S :                                                          *
*  This subroutine processes the text file to either the end-of-file or dis-  *
* covery of a null first character of a record which indicates that the input *
* file is already keyed.  Each PHYSICAL record is read and the corresponding  *
* KEY is saved in REC_KEY.  Each PHYSICAL record contains 128 characters and  *
* may contain one or more TEXT records.  The end of a TEXT record is signified *
* by a CARRIAGE-RETURN character followed by a LINE-FEED character.  Each PHY- *
* SICAL record is searched for the next LINE-FEED character. The next TEXT rec-*
* ord starts following this character.  The corresponding offset is saved in  *
* START (the first text record starts at 0).  REC_KEY is saved in KEY and the  *
* record length of the preceeding TEXT record is saved in LENGTH. TEXT records *
* may span more than one PHYSICAL record.  For each TEXT record the user is re-*
* quested to enter an integer value that is associated with this TEXT record  *
* and is saved in TYPE.  A special program option may be selected to skip entry*
* of this variable.  PROCESS uses subroutine GET_RECORD to read each PHYSICAL  *
* record.                                                                     *
*****************************************************************************/
    process:
        procedure;

            do while(end_of_file = 'f');
            if rec_start < 128 then key(max_number) = rec_key;
            else key(max_number) = rec_key + 1;
            if rec_start < 128 then start(max_number) = rec_start + 1;
            else start(max_number) = rec_start - 127;
            rec_length = index(substr(sum_buffer,rec_index+1),line_feed);
            if rec_length = 0
                then do;
                call get_record;
                rec_length = index(substr(sum_buffer,rec_index+1),line_feed);
                end;
/*****************************************************************************
* Check for control-z (ascii(26)) indicating the end-of-string text          *
*****************************************************************************/
            else if substr(sum_buffer,rec_index+1,1)=ascii(26)
                then do;
                continue = 't';
                call get_record;
                end;
            if end_of_file = 'f'
                then do;
                hard_copy = 1;
                length_rec(max_number) = rec_length;
                if file_type = 0 then type(max_number) = 0;
                    else do while(hard_copy = 0);
                    put_buffer = substr(sum_buffer,rec_index+1,rec_length);
                    put file(crt) skip edit(put_buffer) (a);
                    put file(crt) edit('Enter variable type: ') (a);
                    call get_integer(hard_copy,number);
                    type(max_number) = number;
                    end;
                rec_start = rec_start + rec_length;
                rec_index = rec_index + rec_length;
                max_number = max_number + 1;
```

```
            if max_number > 128
                then do;
                put file(crt) skip edit('Maximum array size of 128 ',
                'exceeded.  Recompile with larger array.') (a);
                stop;
                end;
            end;
        end;
    return;
    end process;

/*****************************************************************************
*  G E T _ R E C O R D :                                                    *
*  This subroutine gets the next PHYSICAL record, stores the associated key in *
* REC_KEY and checks for the end of the set of TEXT records.               *
*****************************************************************************/
    get_record:
        procedure;

        read file(output) into(buffer) keyto(rec_key);
        if substr(buffer,1,1) = ascii(0) !  continue = 'f'
            then do;
                do while(end_of_file = 'f');
                read file(output) into(buffer) keyto(number);
                end;
            end;
        sum_buffer = substr(sum_buffer,rec_index+1) || buffer;
        rec_index = 0;
        rec_start = rec_start - 128;
        return;
        end get_record;

/*****************************************************************************
*  D O N E :                                                                *
*  This subroutine closes the text file and reopens it in the UPDATE mode,  *
* appends the special key data, and writes a message to the user.           *
*****************************************************************************/
    done:
        procedure;

        close file(output);
        open file(output) update direct env(f(128)) title(filename||'.DAT');
        max_number = max_number - 1;
        call clearscreen;
        put file(crt) skip edit(max_number,' records processed.',
        '  Now appending key map') (f(3,0),2a);
        rec_start = 1;
            do while(rec_start <= max_number);
            buff_add = addr(buff(rec_start));
            write file(output) from(record_buffer) keyfrom(rec_key);
            rec_key = rec_key + 1;
            rec_start = rec_start + 32;
            end;
```

```
        put file(crt)skip edit('Job successfully completed.','File ',filename,
        '.DAT contains',rec_key - 1,' records.') (a,skip,a,a,a,f(2,0));
        close file(output);
        call screen_clear(continue,3);
        return;
        end done;

/*****************************************************************************
*  S U M M A R Y :                                                          *
*  This subroutine provides an optional record of the keyed file.  A summary  *
* can be displayed on the CRT.  A hadr-copy option is also available.       *
*****************************************************************************/
    summary:
        procedure;

        hard_copy = -1;
            do while(hard_copy < 0 ! hard_copy > 2);
            call clearscreen;
            put file(crt) skip(0) edit('Enter 0 - To Quit',
            '1 - CRT Summmary Only',
            '2 - CRT/PRINTER Summary') (a,col(7));
            put file(crt) skip;
            call get_integer(file_type,hard_copy);
            if file_type = 0 then hard_copy = -1;
            end;
        if hard_copy = 0 then return;
        put file(crt) skip(2) edit('A list of the input records for file: ',
        filename,'.DAT follows.') (a);
        if hard_copy = 2
            then do;
            open file(printer) stream output title('$lst');
            put file(printer) skip edit(new_page,'A list of input records for '
            ,' file: ',filename,'.DAT follows.') (a);
            end;
        call find_record(max_number,0,filename);
        call clearscreen;
        line_count = 0;
        put file(crt) skip;
            do number = 1 to max_number;
            call find_record(rec_index,number,put_buffer);
            put file(crt) skip edit(number,put_buffer) (f(3,0),x(1),a);
            line_count = line_count + 1;
            if hard_copy = 2 then put file(printer) skip
            edit(number,put_buffer) (f(3,0),x(1),a);
            if line_count = 20
                then do;
                call screen_clear(continue,1);
                line_count = 0;
                end;
            end;
        if line_count = 0 then call screen_clear(continue,1);
        line_count = 0;
        put file(crt) edit('A summary of record keys follows') (col(9),a);
```

```
        if hard_copy = 2 then put file(printer) skip(4)
        edit('A summary of record keys follows') (col(9),a);
        if hard_copy = 2 then call header(printer);
            do number = 1 to max_number;
            call print_line(crt);
            if hard_copy = 2 then call print_line(printer);
            line_count = line_count + 1;
            if line_count = 20
                then do;
                call header(crt);
                end;
            end;
        call find_record(rec_index,-1,put_buffer);
        close file(crt);
        if hard_copy = 2 then put file(printer) skip;
        if hard_copy = 2 then close file(printer);
        return;
        end summary;

/*****************************************************************************
*  H E A D E R :                                                            *
*  This subroutine waits for an input character, clears the screen and resets  *
* the line counter LINE_COUNT.                                              *
*****************************************************************************/
    header:
        procedure(x);

        dcl x file;
        if x=crt
            then do;
            call screen_clear(continue,1);
            line_count = 0;
            end;
        put file(x) skip edit('RECORD KEY START LENGTH TYPE') (a);
        return;
        end header;

/*****************************************************************************
*  P R I N T _ L I N E :                                                    *
*  This subroutine prints the TEXT record number, associated KEY,START, and  *
* LENGTH followed by the first 51 characters of the TEXT record.            *
*****************************************************************************/
    print_line:
        procedure(x);

        dcl x file;

        call find_record(rec_index,number,put_buffer);
        put_buffer = substr(put_buffer,1,min(length(put_buffer),51));
        put file(x) skip edit(number,key(number),start(number),
        length_rec(number),rec_index,put_buffer)(f(4,0),f(5,0),3f(6,0),x(1),a);
        return;
        end print_line;

    end keywrite;
```

```
title    'Direct CP/M Calls From PL/I-80'
;
;*****************************************************************************
;*                                                                           *
;*    cp/m calls from pl/i for direct i/o                                    *
;*                                                                           *
;*****************************************************************************
public    coninp        ;direct console input (#6a)
public    conout        ;direct console output (#6b)
public    clearscreen   ;clear the screen.
public    upcursor      ;move cursor up n lines.
public    downcursor    ;move cursor down n lines.
public    leftcursor    ;move cursor left n columns.
public    rightcursor   ;move cursor right n columns.
extrn     ?bdos         ;bdos entry point
extrn     ?boot         ;system reboot entry point.
;
;*****************************************************************************
;*                                                                           *
;*    equates for interface to cp/m bdos and control character definitions   *
;*                                                                           *
;*****************************************************************************
stop   equ 3            ;control C character
diof   equ 6            ;direct i/o, version 2.0
bell   equ 7            ;bell
left   equ 8            ;moves cursor left.
bckspa equ 8            ;backspace character.
down   equ 10           ;moves cursor down.
linfd  equ 10           ;line feed character.
up     equ 11           ;moves cursor up.
right  equ 12           ;moves cursor right.
newpg  equ 12           ;printer top of form.
caretn equ 13           ;carriage return character.
clear  equ 26           ;control z character clears screen.
escape equ 27           ;escape character.
;
;    utility functions
;*****************************************************************************
;*                                                                           *
;*       general purpose routines used upon entry                            *
;*                                                                           *
;*****************************************************************************
;
getpl:                  ;get single byte parameter to register e
    mov e,m             ;low (addr)
    inx h
    mov d,m             ;high(addr)
    xchg                ;hl = .char
    mov e,m             ;to register e
    ret
```

```
;*****************************************************************************
;*  C U R S O R :  Moves cursor up,down,left,right.                          *
;*****************************************************************************
cursor:                 ;B register contains print character.
                        ;C register contains LIMIT value.
    call getpl          ;get number of shifts N in E register.
    mov a,e             ;N to A register.
    cpi  1              ;check for postive N.
    jp   positive       ;if positive continue.
    ret                 ;non-positive N. Quit.
positive:
    sub c               ;subtract LIMIT from N.
    jm  ok              ;N is < LIMIT.
    mov e,c             ;Set N to maximum value.
ok:
    mov d,e             ;N into D register.
    mov e,b             ;character to E register.
loop:
    push d              ;save D/E registers.
    call chrout         ;call BDOS.
    pop d               ;restore D/E registers.
    dcr d               ;decriment D register.
    jnz loop            ;repeat N times.
    ret                 ;return to caller.
;
;*****************************************************************************
;*  C O N I N P :                                                            *
;*****************************************************************************
coninp:                 ;perform console input, char returned in stack
    mvi e,0ffh          ;ffh to e register to signify read.
    mvi c,diof          ;function # to c register.
    call ?bdos          ;call to BDOS.
    cpi  0              ;check if null character returned.
    jz coninp           ;no character ready.  Try again.
    cpi stop            ;check if control C character returned.
    jz reboot           ;reboot system.
    pop h               ;recall return address
    push psw            ;save character
    inx sp              ;delete psw
    mvi a,l             ;length to a
    pchl                ;return to caller
reboot:                 ;system reboot.
    jmp ?boot
;
;*****************************************************************************
;*  C O N O U T :                                                            *
;*****************************************************************************
conout:                 ;direct console output
;    l->char(l)
    call    getpl       ;get parameter
chrout:                 ;common entry to print character.
    mvi c,diof          ;function # to c register.
    jmp ?bdos           ;return through BDOS.
;
```

```
;*****************************************************************************
;*  C L E A R S C R E E N :                                                  *
;*****************************************************************************
clearscreen:            ;clear the screen.
    mvi e,clear         ;CLEAR SCREEN character to E register.
    jmp chrout          ;print character and return through BDOS.
;
;*****************************************************************************
;*  U P C U R S O R :                                                        *
;*****************************************************************************
upcursor:               ;move cursor up n columns.
    mvi c,24            ;set limit value to 24.
    mvi b,up            ;move UP character to A register.
    jmp  cursor         ;jump to common cursor control routine.
;
;*****************************************************************************
;*  D O W N C U R S O R :                                                    *
;*****************************************************************************
downcursor:             ;move cursor down n columns.
    mvi c,24            ;set limit value to 24.
    mvi b,down          ;move DOWN character to A register.
    jmp  cursor         ;jump to common cursor control routine.
;
;*****************************************************************************
;*  L E F T C U R S O R :                                                    *
;*****************************************************************************
leftcursor:             ;move cursor left n columns.
    mvi c,79            ;set limit value to 79.
    mvi b,left          ;get LEFT character to A register.
    jmp  cursor
;
;*****************************************************************************
;*  R I G H T C U R S O R :                                                  *
;*****************************************************************************
rightcursor:            ;move cursor right n columns.
    mvi c,79            ;set limit value to 79.
    mvi b,right         ;move RIGHT character to A register.
    jmp  cursor
;
;*****************************************************************************
;*  C O N T R O L S : common block                                           *
;*****************************************************************************
common /controls/       ;initialize common block CONTROLS variables.
    db  newpg,bell,escape,caretn,bckspa,linfd,0,0
```

```
;*****************************************************************************
;*                                                                           *
;*                              P L I D I O                                  *
;*                                                                           *
;*  D i r e c t   R E A D / W R I T E   a n d   C U R S O R   C o n t r o l  *
;*                                                                           *
;*       F o r   I B M / P C   C a l l e d   F r o m   P L / I - 8 6         *
;*                                                                           *
;*     U s i n g   t h e   D O S   a n d   B I O S   R o u t i n e s         *
;*                                                                           *
;*****************************************************************************

DSEG
extrn ?begin:word          ;beginning of free list

CSEG
public coninp
public conout
public upcursor
public downcursor
public leftcursor
public rightcursor
public clearscreen

;*****************************************************************************
;*  D O S   a n d   B I O S   F u n c t i o n   N u m b e r   E q u a t e s  *
;*****************************************************************************

write_cursor equ 2
read_cursor equ 3
stop equ 3
diof equ 6
bell equ 7
bckspa equ 8
linfd equ 10
newpg equ 12
caretn equ 13
escape equ 27

;*****************************************************************************
;*                                                                           *
;*              G e n e r a l   P u r p o s e   R o u t i n e s              *
;*                                                                           *
;*****************************************************************************

?pcdos:                    ;the call to IBM DOS
    int 21h                ;DOS interrupt number
    ret

getpl:                     ;get single byte parameter to DL register
    mov bx,[bx]            ;BX = pointer to byte
    mov dl,[bx]            ;put byte in DL register
    ret
```

```
find_cursor:                ;saves registers, gets integer value, and
                            ;read cursor position.
    call savereg            ;saves AX,BX,CX,DX registers
    call getpl              ;get integer value (fixed(7))
    mov bl,dl               ;transfer to BL register
    cmp bl,0                ;compare with 0
    jge positive            ;jump if non-negative
    mov bl,0                ;force value to 0
positive:
    push bx                 ;save BX register (used for BIOS call)
    mov ah,read_cursor      ;BIOS function # to AH
    mov bh,0                ;set current page indicator for 1st page
    int 10h                 ;BIOS interrupt number for monitor
     pop bx                         ;restore BX register, (DH,DL) contain cursor
(row,col)
    ret

move_cursor:                ;position cursor at (row,column) in (DH,DL)
    mov bh,0                ;set current page indicator to 1st page
    mov ah,write_cursor     ;BIOS function # to AH
    int 10h                 ;BIOS interrupt number for monitor
    ret

savereg:                    ;save register AX,BX,CX,DX
    mov save_ax,ax
    mov save_bx,bx
    mov save_cx,cx
    mov save_dx,dx
    ret
save_ax rw 1
save_bx rw 1
save_cx rw 1
save_dx rw 1

allset:                     ;reposition cursor, restore registers, and return PL/I
    call move_cursor        ;routine to repostion cursor

done:                       ;restore AX,BX,CX,DX registers and return to PL/I
    mov ax,save_ax
    mov bx,save_bx
    mov cx,save_cx
    mov dx,save_dx
    ret

;*****************************************************************************
;*                                                                           *
;*                    The  IBM  DOS  Routines                                *
;*                                                                           *
;*****************************************************************************
```

```
;*****************************************************************************
;*  C O N I N P                                                              *
;*****************************************************************************
coninp:                     ;direct console input, char returned in stack (#6a)
    mov ah,diof             ;function # to AH
    mov dl,0ffh             ;set read flag in DL
    call ?pcdos             ;value returned in AL
    jz coninp               ;returns 0 if no character ready
    pop bx                  ;return address to BX
    cmp al,stop             ;check for control C character
    je reboot               ;reboot system
    mov ah,al               ;char to AH
    push ax                 ;character to the stack
    inc sp                  ;delete garbage byte
    mov al,1                ;set character length to 1
    jmp bx                  ;return to PL/I
reboot:                     ;system reboot. Location 40h in the base page has the
                            ;INT 20h needed to reboot. This routine sets up an
                            ;offset of 0 with that segment and jumps to it, ex-
                            ;cuting the INT 20 h.
    mov word ptr .3eh,0     ;set up an offset of zero
    jmpf dword ptr .3eh     ;jump

;*****************************************************************************
;*  C O N O U T                                                              *
;*****************************************************************************
conout:                     ;direct console output (#6b)
    call getpl              ;get character to DL
    mov ah,diof             ;function # to AH
    jmp ?pcdos              ;return through DOS

;*****************************************************************************
;*                                                                           *
;*                  T h e   I B M   B I O S   R o u t i n e s                *
;*                                                                           *
;*****************************************************************************

;*****************************************************************************
;*  C L E A R S C R E E N                                                    *
;*****************************************************************************
clearscreen:                ;this routine clears the screen
    call savereg            ;save registers
    push di                 ;save DI register
    push es                 ;save ES register
    mov ax,0                ;0 AX
    mov es,ax               ;set segment to absolute 0
    mov bx,410h             ;off-set to GRAPHICS/MONOCHROME switch value
    mov ax,0b800h           ;AX points to GRAPHICS memory
    mov di,es:[bx]          ;switch value to DI
    and di,30h              ;AND with 30h
    cmp di,30h              ;compare to 30h
    jne graphics            ;if DI = to 30h then GRAPHICS board
    mov ax,0b000h           ;AX points to MONCHROME memory
```

```
graphics:
    mov es,ax               ;set segment to MONITOR memory
    mov di,0                ;initialize destination value to 0
    mov al,' '              ;BLANK to AL
    mov ah,07h              ;set attribute to LIGHT on DARK
    mov cx,2000             ;set loop index to replace 2000 words (80x25) char
    cld                     ;causes positive increment
    rep stosw               ;"BLANK" monitor memory
    mov dx,0                ;set (DH,DL) for ROW 0 and COLUMN 0
    call move_cursor        ;reposition cursor
    pop es                  ;restore ES register
    pop di                  ;restore DI register
    jmp done                ;restore registers and return to PL/I

;*****************************************************************************
;*  U P C U R S O R                                                          *
;*****************************************************************************
upcursor:                   ;move CURSOR up I rows (I returned in BL)
    call find_cursor        ;current CURSOR position in (DH,DL) and I in BL
    sub dh,bl               ;subtract I from current row number (DH)
    cmp dh,0                ;do not allow negative row number
    jge plus_row
    mov dh,0                ;set row number to 0
plus_row:
    jmp allset              ;reposition CURSOR and return to PL/I

;*****************************************************************************
;*  D O W N C U R S O R                                                      *
;*****************************************************************************
downcursor:                 ;move CURSOR down I rows (I returned in BL)
    call find_cursor        ;current CURSOR position in (DH,DL) and I in BL
    add dh,bl               ;add I to current row number (DH)
    cmp dh,24               ;do not allow position beyond row 25
    jle ok_row
    mov dh,24               ;force to row 25
ok_row:
    jmp allset              ;reposition CURSOR and return to PL/I

;*****************************************************************************
;*  L E F T C U R S O R                                                      *
;*****************************************************************************
leftcursor:                 ;move CURSOR left I columns (I returned in BL)
    call find_cursor        ;current CURSOR position in (DH,DL) and I in BL
    sub dl,bl               ;subtract I from current column number (DL)
    cmp dl,0                ;do not allow position less than column 1
    jge plus_col
    mov dl,0                ;force to column 1
plus_col:
    jmp allset              ;reposition CURSOR and return to PL/I
```

```
;*****************************************************************************
;*  R I G H T C U R S O R                                                    *
;*****************************************************************************
rightcursor:                ;move CURSOR right I columns (I returned in BL)
    call find_cursor        ;current CURSOR position in (DH,DL) and I in BL
    add dl,bl               ;add I to current column number (DL)
    cmp dl,79               ;do not allow position greater than column 80
    jle ok_col
    mov dl,79               ;force to column 80
ok_col:
    jmp allset              ;reposition CURSOR and return to PL/I
controls dseg common        ;common block CONTROLS initialization
    db  newpg,bell,escape,caretn,bckspa,linfd,0,0
```

A.3 FILE BREEDER.DAT

Welcome to the Large-Scale-Prototype-Breeder(LSPB) Financial Analysis Model (BREEDER, Ver 1.5). BREEDER is a micro-computer-based financial model of the LSPB project. The Model performs the following tasks:

- File Management
- Data Input
- Construction Analysis
- Cashflow Analysis
- Report Writing

Each routine features a menu driven control logic with protection against most user input errors. ESCAPE returns the user to the current menu.
The results generated by this program are confidential and are intended for the use of Consolidated Management Office (COMO) staff in evaluating alternative financing strategies for the LSPB project. Release to the public of any information contained in this program or results produced by this program is prohibited withour prior approval by COMO staff.

BREEDER was developed:

by

Special Projects and Industrial Applications Group
Energy and Environmental Systems Division
Argonne National Laboratory

for

Consolidated Management Office
Electric Power Research Institute

A.4 FILE INPUT.DAT

Header
Years
Factors (Decimal)
Flags (0 or 1)
Plant Variables
Rates (Decimal)
Plant Arrays
Construction Expenses (million$/yr)

Base Year
Construction Start
Construction End
Levelization Period
Debt Repayment Period
Equity Fraction
Debt Fraction
Ad Valorem Tax
Property Tax
Income Tax Rate
ITC Base Rate
Target Capacity Factor
Trigger Capacity Factor
Target Return on Equity
Backfit-Replacement Rate
95%(0)/Full(1) Dep. Flag
StLn(0)/ACRS(1) Dep. Flag
Flowthru(0)/Norm.(1) Flag
Uncons(0)/Cons(1)Tax Flag
Quarter(0)/Annual(1) Flag
Compound(0)/Simple(1)Flag
Gross(0)/Net(1)PropTxFlag
Pre(0)/After(1)TxDiscFlag
Net(0)/Fix(1) Equity Flag
Bot.Up(0)/Top.Dn.(1) Flag
Size (MW)
Book Life (yrs)
Tax Life (yrs)
Startup (millions $)
Init.Work.Cap.(million $)
Heat Rate (Btu/kWh)
General Inflation
Short-Term Debt
Long-Term Debt
Preferred Stock
Equity
Capacity Factor (Decimal)
Fuel Cost ($/MMBtu)
O&M (mills/kWh)
Working Cap. (millions $)
Federal Sub. (millions $)
Busbar Costs (cents/kWh)
Plant

A.5 FILE CONSTR.DAT

```
Quarterly Construction Expenses (millions of current $)
        --Direct Construction--  -----Interest and Equity Charges-----   Total
        ON Cost  Infl.   Total  ST debt LT debt  Pref.  Common   Total   Cost
Quarterly Construction Financing (millions of current $)
        --------Source of Funds-------- ----Quarter---- ------Cumulative------
        ST debt Lt debt  Pref.  Common   CWIP    AFUDC   CWIP    AFUDC   Total
Annual Construction Expenses (millions of current $)
        --Direct Construction--  -----Interest and Equity Charges-----   Total
 Year   ON Cost  Infl.   Total  ST debt LT debt  Pref.  Common   Total   Cost
Annual Construction Financing (millions of current $)
        --------Source of Funds-------- ----Annual----- ------Cumulative------
 Year   ST debt Lt debt  Pref.  Common   CWIP    AFUDC   CWIP    AFUDC   Total
Construction-End Summary
Accruals (million $)  ------Direct------ ------AFUDC------- ---Weighted Rate---
Inv.Tx.Cr.  Tax Sav.  Current  Base Year Current  Base Year Preferred   LT debt
```

A.6 FILE REPORT.DAT

BALANCE SHEET
ASSETS
Cum. Working Capital
Gross Plant
Less Accum.Plant Dep.
Net Plant
Capitalized AFUDC
Less Accum.AFUDC Dep.
Net Capitalized AFUDC
TOTAL ASSETS
LIABILITIES
LTD Due in Next Year
LTD Outstanding
Cum. Deferred ITC
Cum. Deferred FIT
P.S. Due in Next Year
Pref.Stock Outstanding
End Common Equity
Partner Loans Outst.
Total Stock Equity
TOTAL LIABIL. & EQUITY
INCOME STATEMENT
REVENUES
Electric Sales
Federal Subsidy
TOTAL REVENUES
EXPENSES
Fuel Expense
O&M Expense
Backfit&Replacement
Ad Valorem Tax
Total Oper. Expense
Plant Book Dep.
AFUDC Book Dep.
Total Book Dep.
Property Tax
Net FIT (Book Basis)
Total Pre-Debt Expen.
Pre-Debt Net Income
LTD Interest
NET INCOME
Less Pref. Dividends
Common Stock Earnings
BUSBAR COST (cents/kWh)
SOURCES AND USES OF FUNDS STATEMENT
SOURCES OF FUNDS
Net Income
Plant Book Dep.
AFUDC Book Dep.
Total Book Dep.
Deferred ITC
Deferred FIT
TOTAL INTERNAL
Partner Loans

Government Loans
TOTAL SOURCES
USES OF FUNDS
Preferred Stock Due
Pref. Dividends
Common Dividends
Working Capital Change
LTD Repayment
Partner Loan Repayment
TOTAL USES
RETAINED EARNINGS STATEMENT & TAX DEP. WORKSHEET
RETAINED EARNINGS
Begin.Common Equity
Common Stock Earnings
Total Common Equity
 Less Common Dividends
END COMMON EQUITY
RETURN ON EQUITY (%)
TAX DEP. WORKSHEET
Startup Dep.
5-year Prop.Dep.
10-year Prop.Dep.
15-year Prop.Dep.
Straight-line Dep.
TOTAL TAX DEPRECIATION
TAX CALCULATION (Book/Regulation Basis)
BOOK TAX CALCULATION
Electric Sales
Total Oper. Expense
Total Book Dep.
Property Tax
LTD Interest
 Pre-tax Book Income
Book FIT
 Less amort.ITC
NET FIT (Book Basis)
REGULATORY TAX CALCULATION
Electric Sales
Total Oper. Expense
Total Tax Depreciation
Property Tax
LTD Interest
 Pre-tax Part. Income
FIT Due
 Less ITC Taken
 Less FIT Taken
NET REGULATED FIT
PARTNERSHIP RETURN AND CASHFLOW ANALYSIS
PARTNERSHIP RETURN
Electric Sales
Total Oper. Expense
Total Tax Depreciation
Property Tax
LTD Interest
 Pre-tax Part. Income
FIT Due

PARTNERSHIP CASHFLOW ANALYSIS
After-tax Cash
Total Tax Depreciation
Federal Subsidy
Total Cash Avail.
Less LTD Repayment
Less Pref. Dividends
Less Pref. Stock Due
Less Partner Loans
Less Work.Cap. Change
NET TO PARTNERS
CUMULATIVE TO PARTNERS
AFTER-TAX CASH FLOWS TO UTILITY INVESTORS

	During Construction		During Operation		Net A.T.	Cum.A.T.	Cm.Dis.R.
Year	Cash Inv.	Tax Ben.	Tax Due	Cash Dist.	Cash Flow	Cash Flow	A.T.C.F.

A.7 FILE SUMMAR.DAT

CONSTRUCTION SUMMARY
(All Amounts in millions of current year $)
Period: (

Expenses:	Total	Book Life	Book Dep.
Direct :			
Startup:			
AFUDC :			
Wk.Cap.:			
Total :			

Financing:	Total	% Financing	Charge Rate
Common :			
Preferred:			
LT.Debt :			
Total :			

Accruals:
ITC:
FIT:
OPERATING SUMMARY
Acceleration Tax Depreciation:
Tax Life :
Method :
Straight-line
ACRS (TEFRA version)
Basis :
TEFRA Opt.:
95 %
Full

Analysis Summary:	ROI (%)	Lev. Busbar (cents/kWh)	Discount Rate (%)	Fixed Charge Rate (%)
Current:				
Base Yr:				

Option Flags
IDC Compounding :
Compound
Simple
Financing Period :
Quarter
Annual
Property Tax Basis :
Gross
Net
Equity Finan. Basis:
Zero Net Equity
Fixed %
Discount Rate :
Pre-tax
After-tax
Analysis Type :
Bottom-up
Top-down
Accounting Basis :

Flow-thru
Normalized
Reg. Tax Treatment :
Unconsolidated
Consolidated

APPENDIX B: CROSS REFERENCE BETWEEN INPUTS OR OUTPUTS AND INTERNAL VARIABLE SPECIFICATIONS

This section provides a cross reference between the variable definitions found in Secs. 3, 4, and 5 and the internal variables used in the BREEDER source code. The cross-reference list includes all variables contained in the three external structures INPUT_DATA, SUMS, and OUTPUT_DATA* For each structure variable, the corresponding variable name and defining equation number from the text is given. Array dimensions are given in parentheses with the structure-variable name, where relevant.

---

*The structures are functionally equivalent to NAMED COMMON blocks in FORTRAN.

TABLE B.1 INPUT_DATA

Structure Variable (and Array Size)	Variable Name	Definition or Equation Number
Header	Header	3-1
Base_Year	Base Year	3-2
Start_Construction	Construction Start	3-3
End_Construction	Construction End	3-4
Levelization_Period	Levelization Period	3-5
Debt_Period	Debt Repayment Period	3-6
Common_Equity_Fraction	Equity Fraction	3-7
Debt_Fraction	Debt Fraction	3-8
Advalorem_Tax	Ad Valorem Tax	3-9
Property_Tax	Property Tax	3-10
FIT_Tax	Income Tax Rate	3-11
ITC_Tax	ITC Base Rate	3-12
Nominal_Point	Target Capacity Factor	3-13
Trigger_Point	Trigger Capacity Factor	3-14
Target_ROE	Target Return on Equity	3-15
Backfit_Replacement Rate	Backfit-Replacement Rate	3-16
TEFRA_Flag	95%(0)/Full(1)Dep. Flag	3-17
Accelerated_Dep_Flag	StLN(0)/ACRS(1)Dep. Flag	3-18
Flowthrough_Normalize_Flag	Flowthru(0)/Norm.(1) Flag	3-19
Consolidated_Tax_Flag	Uncons(0)Cons(1)Tax Flag	3-20

TABLE B.1 (Cont'd)

Structure Variable (and Array Size)	Variable Name	Definition or Equation Number
Quarter_Annual_Flag	Quarter(0)/Annual(1) Flag	3-21
Compound_Simple_Flag	Compound(0)/Simple(1)Flag	3-22
Gross_Net_Flag	Gross(0)/Net(1)PropTaxFlag	3-23
Pre_After_Tax_Flag	Pre(0)/After (1)TxDiscFlag	3-24
Net_Equity_Flag	Net(0)/Fix(1)Equity Flag	3-25
Bottom_Up_Top_Down_Flag	Bot.Up(0)/Top.Dn.(1) Flag	3-26
Size	Size (MW)	3-27
Book_Life	Book Life (yrs)	3-28
Tax_Life	Tax Life (yrs)	3-29
Startup_Cost	Startup (millions $)	3-30
Working_Capital	Init.Work.Cap.(million $)	3-31
Heat_Rate	Heat Rate (Btu/kWh)	3-32
Inflation (30)	General Inflation	3-33
ST_Debt (30)	Short-Term Debt	3-34
LT_Debt (30)	Long-Term Debt	3-35
Preferred_Stock (30)	Preferred Stock	3-36
Common_Equity (50)	Equity	3-37
Capacity_Factor (30)	Capacity Factor (Decimal)	3-38
Fuel_Cost (30)	Fuel Costs ($/MMBtu)	3-39
Operation_Maintenance (30)	O&M (mills/kWh)	3-40
Working_Capital (30)	Working Cap. (millions $)	3-41
Federal_Subsidies (30)	Federal Sub. (millions $)	3-42
Busbar_Costs (30)	Busbar Costs (cents/kWh)	3-43
Plant (15)	Plant	3-44
Last_Entry[a]	-	-

[a]Last_Entry is used to determine the length, in bytes, of the structure INPUT_DATA.

TABLE B.2 SUMS

Structure Variable (and Array Size)	Variable Name	Definition or Equation Number
Construction	Construction Book	4-2
AFUDC	AFUDC Book	4-7
Construction_Real	Construction Base Year	4-3
AFUDC_Real	AFUDC Base Year	4-8
Common_Equity	End Common Equity	4-23
LT_Debt	Total LT Debt	4-20
LT_Debt_Rate	LT Debt Rate	4-19
Preferred_Stock	Preferred Stock Equity	4-22
Preferred_Rate	Preferred Rate	4-21
ITC_Accrual	Total ITC	4-31
Tax_Accrual	Tax Accrual	4-25
Book_Dep_Plant	Plant Book Dep.	5-12
AFUDC_Dep_Plant	AFUDC Book Dep.	5-13
Amort_ITC_Book	Amort ITC	5-15
Book_Depreciation	Total Book Dep.	5-14
Other_Dep	Straight-Line Dep.	5-49
Before_Tax (16)	Pre-Tax Book Income	4-28,40
Interest_Credit (16)	LTD Interest	4-24
ITC_Credit (16)	ITC Rate	4-26,27
After_Tax (16)	After Tax	4-29,41

TABLE B.3 OUTPUT_DATA

Structure Variable (and Array Size)	Variable Name	Definition or Equation Number
Accum_Dep (30)	Accum. Plant Dep.	5-93
Net_Plant (30)	Net Plant	5-94
Accum_AFUDC_Dep (30)	Accum. AFUDC Dep.	5-96
Net_Capitalized_AFUDC (30)	Net Capitalized AFUDC	5-97
Total_Assets (30)	**TOTAL ASSETS**	5-98
LT_Debt_Repayment (31)	LTD Repayment	5-69
LT_Debt_Outstanding (30)	LTD Outstanding	5-99
Deferred_ITC (30)	Cum. Deferred ITC	5-6,101
Deferred_FIT (30)	Deferred FIT	5-64
Preferred_Stock_Outstanding (30)	Pref. Stock Outstanding	5-108
Preferred_Stock_Repayment (31)	Preferred Stock Due	5-68
Cum_Partner_Loans (30)	Partner Loans Outst.	5-4,109
Total_Equity (30)	Total Stock Equity	5-110
Electric_Sales (30)	Electric Sales	5-28,35
Total_Revenues (30)	**TOTAL REVENUES**	5-39
Backfit_Replacement (30)	Backfit & Replacement	5-21
Ad_Valorem_Tax (30)	Ad Valorem Tax	5-29,36
Total_O_and_M (30)	Total Oper. Expense	5-30,37
Property_Tax (30)	Property Tax	5-22
Total_Pre_Debt_Expense (30)	Total Pre-Debt Expense	5-34
Total_Pre_Debt_Income (30)	Pre-Debt Net Income	5-40
LT_Debt_Interest (30)	LTD Interest	5-23
Preferred_Stock_Div (30)	Pref. Dividends	5-24
Net_Income (30)	**NET INCOME**	5-41
Common_Earnings (30)	Common Stock Earnings	5-26, 42
Amort_ITC (30)	Deferred ITC	5-60
Total_Internal (30)	**TOTAL INTERNAL**	5-65
Partner_Loans (30)	Partner Loans	5-72,77,78
DOE_Loans (30)	Government Loans	5-71,76,79
Total_Sources (30)	**TOTAL SOURCES**	5-81
Common_Stock_Div (30)	Common Dividends	5-83
Delta_Working_Capital (30)	Working Capital Change	5-66
Partner_Loan_Repayment (30)	Partner Loan Repayment	5-70,80
Beginning_Common_Equity (30)	Begin. Common Equity	5-25
Total_Common_Equity (30)	Total Common Equity	5-84
End_Common_Equity (30)	**END COMMON EQUITY**	5-85
Pre_Tax_Income_Book (30)	Pre-Tax Book Income	5-27, 31
FIT_Book (30)	Book FIT	5-32
Net_FIT_Book (30)	**NET FIT** (Book Basis)	5-33

TABLE B.3 (Cont'd)

Structure Variable (and Array Size)	Variable Name	Definition or Equation Number
Startup_Dep (30)	Startup Dep.	5-45
Five_Year_Property (30)	5-year Prop. Dep	5-46
Ten_Year_Property (30)	10-Year Prop. Dep.	5-47
Fifteen_Year_Property (30)	15-Year Prop. Dep.	5-48
Total_Tax_Dep (30)	**TOTAL TAX DEPRECIATION**	5-50
Pre_Tax_Income_FERC (30)	Pre-Tax Part. Income	5-51
FIT_Tax_FERC (30)	FIT Due	5-52
Credit_ITC (30)	ITC Taken	5-54, 57
Deferred_FIT_Taken (30)	FIT Taken	5-53, 56
Cum_Deferred_FIT (30)	Cum. Deferred FIT	5-2,102
Net_FIT_FERC (30)	**NET REGULATED FIT**	5-59
After_Tax_Cash (30)	After-Tax Cash	5-87
Total_Cash_Avail (30)	Total Cash Avail.	5-88
Net_To_Partners (30)	**NET TO PARTNERS**	5-89
Cum_To_Partners (30)	**CUMULATIVE TO PARTNERS**	5-9,90

APPENDIX C: A SAMPLE INTERACTIVE SESSION WITH BREEDER

This appendix provides an annotated example of an interactive BREEDER session. Hard-copy printout of actual BREEDER screen displays, including menus and responses, is presented. To improve clarity, screen displays are set in a large sans-serif typeface (Like this) and all user responses are set in italic and underlined (*like this*); the author's explanatory comments are set in serif elite type (like this), in indented blocks.

This session illustrates modification of the input FUEL COST array for the period 1995 to 2000, screen preview of the annual construction summary report, selection of eight out of a possible 30 years for display by the report writer, screen preview of several reports, and use of the ESCAPE key to terminate excessive screen display.

The first three screen displays provide an introduction to the Financial Analysis Model. Successive displays are obtained by entering any valid character as requested. Examples of valid characters are a, 1, A, and carriage return. An example of an invalid character is a single shift key or control key since these do not generate any input to the computer. Two characters have special meaning: the ESCAPE key returns the user to the current menu and the CONTROL C key terminates program execution.

Welcome to the Large-Scale-Prototype-Breeder(LSPB) Financial Analysis Model (BREEDER, Ver 1.5). BREEDER is a micro-computer-based financial model of the LSPB project. The Model performs the following tasks:

- File Management
- Data Input
- Construction Analysis
- Cashflow Analysis
- Report Writing

Each routine features a menu driven control logic with protection against most user input errors. ESCAPE returns the user to the current menu.

<STRIKE ANY KEY TO CONTINUE> a

Neither the Electric Power Research Institute, Inc. (EPRI), the University of Chicago, the U.S. Department of Energy, the U.S. Government, nor any person acting on behalf:

- makes any warranty or representation, express or implied, with respect to this program or the information contained in this program; or

- assumes any liabilities with respect to the use of this program or any information contained in this program.

<STRIKE ANY KEY TO CONTINUE> 1

BREEDER was developed:

by

Special Projects and Industrial Applications Group
Energy and Environmental Systems Division
Argonne National Laboratory

for

Consolidated Management Office
Electric Power Research Institute

<STRIKE ANY KEY TO CONTINUE> A

At this point the user has an option to create an entirely new data file, which requires entry of all input variables, or to use the existing data file, LSPBFILE.DAT. For this session, the second option is selected.

```
Enter 1 - To Create a New Data File
      2 - To Use an Existing File
? 2
```

At this point the file can be listed or modified. The list option is selected.

```
Enter 0 - To Exit
      1 - To List File
      2 - To Modify File
? 1
```

The file can be previewed on the CRT screen or listed on the printer. Option 1 is selected.

```
Enter 0 - To Exit
      1 - To Preview on Screen
      2 - To List on Printer
? 1
```

The input file is shown in six screen displays. In this session, the ESCAPE key is used to halt the screen preview after the first display, returning the user to the list-or-modify menu.

```
  Kidder-Peabody Coal Case (1% fuel escal.,1250 MWe,1995 startup)  10/24/83
    Base Year                 : 1982.00       Construction Start       : 1989.00
    Construction End          : 1995.00       Levelization Period      :   20.00
    Debt Repayment Period     :   30.00
Factors (Decimal):
    Equity Fraction           :  0.3500       Debt Fraction            :  0.5000
    Ad Valorem Tax            :  0.0000       Property Tax             :  0.0200
    Income Tax Rate           :  0.5000       ITC Base Rate            :  0.1000
    Target Capacity Factor    :  0.6500       Trigger Capacity Factor  :  0.0000
    Target Return on Equity   :  0.1415       Backfit-Replacement Rate :  0.0000
Flags (0 or 1):
    95%(0)/Full(1) Dep. Flag : 0              StLn(0)/ACRS(1) Dep. Flag: 1
    Flowthru(0)/Norm.(1) Flag: 1              Uncons(0)/Cons(1)Tax Flag: 0
    Quarter(0)/Annual(1) Flag: 1              Compound(0)/Simple(1)Flag: 1
    Gross(0)/Net(1)PropTxFlag: 1              Pre(0)/After(1)TxDiscFlag: 0
    Net(0)/Fix(1) Equity Flag: 1              Bot.Up(0)/Top.Dn.(1) Flag: 0
Plant Variables:
    Size (MW)                 : 1250.00       Book Life (yrs)          :   30.00
    Tax Life (yrs)            :   15.00       Startup (million $)      :    0.00
    Init.Work.Cap.(million $):    0.00        Heat Rate (Btu/kWh)      :10150.00
<STRIKE ANY KEY TO CONTINUE> (ESCAPE)
```

The HEADER entry is now modified.

```
Enter 0 - To Exit
      1 - To List File
      2 - To Modify File
? 2

Enter 0 - For No More Changes
      1 - To Modify All Entries
      2 - To Modify Selected Entries
? 2

 1 - Header
 2 - Years
 3 - Factors (Decimal)
 4 - Flags (0 or 1)
 5 - Plant Variables
 6 - Rates (Decimal)
 7 - Plant Arrays
 8 - Construction Expenses (million$/yr)

Enter number corresponding to data to be changed (0 for no more changes)
? 1
```

The old header is printed on the screen and the user is given an option to retain or modify the header. For this example, the header is modified and the updated header is retained. The use of CARRIAGE RETURN as an alternative to entering 0 is also illustrated.

```
Header
  Kidder-Peabody Coal Case (1% fuel escal.,1250 MWe,1995 startup)  10/24/83
Enter 0 - To Retain Current Value
      1 - To Modify Current Value
? 1

Enter Header:
? This is a test

Header
  This is a test

Enter 0 - To Retain Current Value
      1 - To Modify Current Value
? (CARRIAGE RETURN)
```

Next the FUEL COST entry of the PLANT ARRAYS group is selected for modification.

```
1 - Header
2 - Years
3 - Factors (Decimal)
4 - Flags (0 or 1)
5 - Plant Variables
6 - Rates (Decimal)
7 - Plant Arrays
8 - Construction Expenses (million$/yr)

Enter number corresponding to data to be changed (0 for no more changes)

? 7


Plant Arrays:
     1 - Capacity Factor (Decimal)
     2 - Fuel Cost ($/MMBtu)
     3 - O&M (mills/kWh)
     4 - Working Cap. (million $)
     5 - Federal Sub. (million $)
     6 - Busbar Costs (cents/kWh)

Enter the number corresponding to the data to be changed (0 for no more changes)

? 2
```

The current array values are printed and the user has the option of modifying either a range of years or a single year. For this example, the years 1995 to 2000 are modified using the feature that escalates a starting value. This example also illustrates BREEDER's error recovery system. Instead of entering a first year of 1995, 1994 is typed. This value is out-of-range and BREEDER responds by sounding the CRT "bell" and retyping the request.

```
Plant Arrays
Fuel Cost ($/MMBtu):
          1995 2.2875   1996 2.3104   1997 2.3335   1998 2.3568   1999 2.3804
          2000 2.4042   2001 2.4282   2002 2.4525   2003 2.4770   2004 2.5018
          2005 2.5268   2006 2.5521   2007 2.5776   2008 2.6034   2009 2.6294
          2010 2.6557   2011 2.6823   2012 2.7091   2013 2.7362   2014 2.7635
          2015 2.7912   2016 2.8191   2017 2.8473   2018 2.8758   2019 2.9045
          2020 2.9336   2021 2.9629   2022 2.9925   2023 3.0224   2024 3.0527

Enter 0 - To Retain Current Values
      1 - To Modify Range of Years
      2 - To Modify Selected Year
? 1
```

```
Enter 1 - To Specify Starting and Ending Year
      2 - To Process Entire Array
? 1

Enter First Year ( 1995, 2024)

? 1994

(The bell sounds)

Enter First Year ( 1995, 2024)

? 1995

Enter Last Year  ( 1995, 2024)

? 2000

Enter 1 - To Enter Array Values
      2 - To Specify  1995 Value and Escalation Factor
? 2
```

The current fuel-cost values are first printed. Then values for the period 1995 to 2000 are entered using the value-and-escalation feature. A 1995 starting fuel value of $2.00/MMBtu is selected and escalated at 5% per year for the years 1996-2000.

```
Plant Arrays
Fuel Cost ($/MMBtu):
          1995 2.2875   1996 2.3104   1997 2.3335   1998 2.3568   1999 2.3804
          2000 2.4042   2001 2.4282   2002 2.4525   2003 2.4770   2004 2.5018
          2005 2.5268   2006 2.5521   2007 2.5776   2008 2.6034   2009 2.6294
          2010 2.6557   2011 2.6823   2012 2.7091   2013 2.7362   2014 2.7635
          2015 2.7912   2016 2.8191   2017 2.8473   2018 2.8758   2019 2.9045
          2020 2.9336   2021 2.9629   2022 2.9925   2023 3.0224   2024 3.0527

Enter Value For 1995
? 2

Enter Annual Escalation Value (i.e. 1.10)
? 1.05


Plant Arrays
Fuel Cost ($/MMBtu):
          1995 2.0000   1996 2.1000   1997 2.2050   1998 2.3152   1999 2.4310
          2000 2.5526   2001 2.4282   2002 2.4525   2003 2.4770   2004 2.5018
          2005 2.5268   2006 2.5521   2007 2.5776   2008 2.6034   2009 2.6294
          2010 2.6557   2011 2.6823   2012 2.7091   2013 2.7362   2014 2.7635
          2015 2.7912   2016 2.8191   2017 2.8473   2018 2.8758   2019 2.9045
          2020 2.9336   2021 2.9629   2022 2.9925   2023 3.0224   2024 3.0527
Enter 0 - To Retain Current Values
      1 - To Modify Range of Years
      2 - To Modify Selected Year
? 0
```

These changes are retained, and the input routine exits.

Plant Arrays:
1 - Capacity Factor (Decimal)
2 - Fuel Cost ($/MMBtu)
3 - O&M (mills/kWh)
4 - Working Cap. (million $)
5 - Federal Sub. (million $)
6 - Busbar Costs (cents/kWh)

Enter the number corresponding to the data to be changed (0 for no more changes)

? 0

1 - Header
2 - Years
3 - Factors (Decimal)
4 - Flags (0 or 1)
5 - Plant Variables
6 - Rates (Decimal)
7 - Plant Arrays
8 - Construction Expenses (million$/yr)

Enter number corresponding to data to be changed (0 for no more changes)

? 0

Enter 0 - For No More Changes
1 - To Modify All Entries
2 - To Modify Selected Entries

? 0

Enter 0 - To Exit
1 - To List File
2 - To Modify File

? 0

At this point the user can either replace the existing file LSPBFILE.DAT with current data or use the current data for this run only. The second option is selected by responding with "n" ("N" is also acceptable).

Do you wish this data to replace the current permanent data file (y/n)? n

Now the construction analysis is performed and the annual report is previewed on the screen.

Beginning CONSTRUCTION Analysis

CONSTRUCTION analysis complete.

You may now obtain a summary of the the constuction profile.

Enter 0 - No Listing
1 - Preview on Screen
2 - List on Printer
? 1

Enter 0 - To Exit
1 - To print quarterly report
2 - To print annual report
3 - To print both reports
? 2

Annual Construction Expenses (millions of current $)

Year	--Direct Construction-- ON Cost	Infl.	Total	-----Interest and Equity Charges----- ST debt	LT debt	Pref.	Common	Total	Total Cost
1989	49.0	25.0	74.0	3.0	0.0	0.0	0.0	3.0	77.1
1990	165.0	101.2	266.2	12.6	0.0	0.8	2.9	16.3	282.5
1991	327.0	236.6	563.6	29.8	3.8	4.2	14.4	52.3	615.8
1992	328.0	275.0	603.0	41.8	17.6	12.1	41.1	112.6	715.6
1993	232.0	222.4	454.4	42.2	47.9	22.2	75.4	187.7	642.1
1994	88.0	95.5	183.5	34.2	82.8	31.7	107.8	256.6	440.1
Accru:	0.0	0.0	0.0	10.6	114.0	8.5	28.9	162.0	162.0
Total:	1189.0	955.7	2144.7	174.3	266.1	79.5	270.6	790.5	2935.2

<STRIKE ANY KEY TO CONTINUE> a

Annual Construction Financing (millions of current $)

Year	--------Source of Funds-------- ST debt	Lt debt	Pref.	Common	----Annual----- CWIP	AFUDC	------Cumulative------ CWIP	AFUDC	Total
1989	77.1	0.0	0.0	0.0	74.0	3.0	74.0	3.0	77.1
1990	205.5	38.5	11.6	27.0	266.2	16.3	340.2	19.4	359.6
1991	333.3	141.3	42.4	98.9	563.6	52.3	903.8	71.6	975.4
1992	99.8	307.9	92.4	215.5	603.0	112.6	1506.8	184.2	1691.0
1993	-73.5	357.8	107.3	250.5	454.4	187.7	1961.3	371.9	2333.1
1994	-202.0	321.0	96.3	224.7	183.5	256.6	2144.7	628.5	2773.2
Accru:	-440.1	301.0	90.3	210.7	0.0	162.0	2144.7	790.5	2935.2
Total:	-0.0	1467.6	440.3	1027.3	2144.7	790.5	2144.7	790.5	2935.2

<STRIKE ANY KEY TO CONTINUE> a

Construction-End Summary

Accruals (million $)		------Direct------		------AFUDC-------		---Weighted Rate---	
Inv.Tx.Cr.	Tax Sav.	Current	Base Year	Current	Base Year	Preferred	LT debt
214.5	220.2	2144.7	1189.0	790.5	398.8	0.09713	0.09751

<STRIKE ANY KEY TO CONTINUE> a

No further construction summaries are requested. At this point the cash-flow analysis begins and the user is requested to enter the number of years of cash-flow analysis desired. In this case a 30-year period is selected.

```
Enter 0 - To Exit
      1 - To print quarterly report
      2 - To print annual report
      3 - To print both reports
? 0

Enter 0 - No Listing
      1 - Preview on Screen
      2 - List on Printer
? 0

Analysis period starts in year:  1995
Enter number of years to be analyzed (1,30)
30

Beginning CASHFLOW analysis

CASHFLOW analysis complete
```

At this point the analysis is complete and the user can preview the reports on the screen or list them on the printer. For this example the reports are previewed on the screen.

```
You may now list various CASHFLOW reports
Enter 0 - No Listing
      1 - Preview on Screen
      2 - List on Printer
? 1
```

Reports can be displayed either for the entire analysis period or for selected years. To limit the number of screen displays to one per report, eight years (1995-2002) are selected.

```
Enter 0 - No More Reports
      1 - Summary
      2 - Balance Sheet
      3 - Income Statement
      4 - Sources and Uses of Funds Statement
      5 - Retained Earnings Statement/Tax Depreciation Worksheet
      6 - Tax Calculation (Book/Regulation Basis)
      7 - Partnership Return & Cashflow Analysis
      8 - Cashflow Summary
      9 - Busbar Cost Plot
      10 - All of Above
      11- Select Printout Years
? 11

ALL YEARS option selected

Enter 0 - To Exit
      1 - To Output All Years
      2 - To Output Selected Years
? 2

Enter years (8 years/page):
Enter year  1 (0 to exit) ? 1995
Enter year  2 (0 to exit) ? 1996
Enter year  3 (0 to exit) ? 1997
Enter year  4 (0 to exit) ? 1998
Enter year  5 (0 to exit) ? 1999
Enter year  6 (0 to exit) ? 2000
Enter year  7 (0 to exit) ? 2001
Enter year  8 (0 to exit) ? 2002
Enter year  9 (0 to exit) ? 0

SELECTED YEARS option
 8 years selected:
 1 - 1995      2 - 1996      3 - 1997      4 - 1998      5 - 1999
 6 - 2000      7 - 2001      8 - 2002

Enter 0 - To Exit
      1 - To Output All Years
      2 - To Output Selected Years
? 0
```

Reports can either be displayed individually or in total. For this example, the all-reports option is selected. Display is halted after the BALANCE SHEET by entering the ESCAPE key, returning the user to the report menu. No more reports are selected for display.

```
Enter 0 - No More Reports
      1 - Summary
      2 - Balance Sheet
      3 - Income Statement
      4 - Sources and Uses of Funds Statement
      5 - Retained Earnings Statement/Tax Depreciation Worksheet
      6 - Tax Calculation (Book/Regulation Basis)
      7 - Partnership Return & Cashflow Analysis
      8 - Cashflow Summary
      9 - Busbar Cost Plot
      10 - All of Above
      11- Select Printout Years
? 10


Beginning ROI Analysis (Current $)

Calculating ROI for year: 30

Beginning ROI Analysis (Base-Year $)

Calculating ROI for year: 30
```

CONSTRUCTION SUMMARY
Period: (1989,1995)
Base Year: 1982

Expenses:	Total	Book Life	Book Dep.
Direct :	2144.7	30	71.5
Startup:	0.0	30	0.0
AFUDC :	790.5	30	26.4
Wk.Cap.:	0.0	--	--
Total :	2935.2	--	97.8

Financing:	Total	% Financing	Charge Rate
Common :	1027.3	35.00	14.15
Preferred:	440.3	15.00	9.71
LT.Debt :	1467.6	50.00	9.75
Total :	2935.2	100.00	11.29

Accruals:
ITC: 214.5
FIT: 220.2

```
<STRIKE ANY KEY TO CONTINUE> a
```

OPERATING SUMMARY

Acceleration Tax Depreciation:
Tax Life : 15 (years)
Method : ACRS (TEFRA version)
Basis : 2037.5
TEFRA Opt.: 95 %

Option Flags
IDC Compounding : Simple
Financing Period : Annual
Property Tax Basis : Net
Equity Finan. Basis: Fixed %
Discount Rate : Pre-tax
Analysis Type : Bottom-up
Accounting Basis : Normalized
Reg. Tax Treatment : Unconsolidated

Analysis Summary:	ROI (%)	Lev. Busbar (cents/kWh)	Discount Rate (%)	Fixed Charge Rate (%)
Current:	15.75	17.02	11.29	16.08
Base Yr:	8.82	4.91	4.79	8.58

<STRIKE ANY KEY TO CONTINUE> a

BALANCE SHEET

	1995	1996	1997	1998	1999	2000	2001	2002
ASSETS								
Cum. Working Capital	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Gross Plant	2144.7	2144.7	2144.7	2144.7	2144.7	2144.7	2144.7	2144.7
Less Accum.Plant Dep.	71.5	143.0	214.5	286.0	357.5	428.9	500.4	571.9
Net Plant	2073.2	2001.7	1930.2	1858.8	1787.3	1715.8	1644.3	1572.8
Capitalized AFUDC	790.5	790.5	790.5	790.5	790.5	790.5	790.5	790.5
Less Accum.AFUDC Dep.	26.4	52.7	79.1	105.4	131.8	158.1	184.5	210.8
Net Capitalized AFUDC	764.2	737.8	711.5	685.1	658.8	632.4	606.1	579.7
TOTAL ASSETS	2837.4	2739.5	2641.7	2543.9	2446.0	2348.2	2250.3	2152.5
LIABILITIES								
LTD Due in Next Year	48.9	48.9	48.9	48.9	48.9	48.9	48.9	48.9
LTD Outstanding	1369.8	1320.9	1271.9	1223.0	1174.1	1125.2	1076.2	1027.3
Cum. Deferred ITC	175.4	200.2	193.0	185.9	178.7	171.6	164.4	157.3
Cum. Deferred FIT	2.0	55.0	97.7	130.3	152.7	175.1	187.3	199.5
P.S. Due in Next Year	14.7	14.7	14.7	14.7	14.7	14.7	14.7	14.7
Pref.Stock Outstanding	410.9	396.3	381.6	366.9	352.2	337.6	322.9	308.2
End Common Equity	815.7	703.7	633.8	574.2	524.7	475.2	435.9	396.6
Partner Loans Outst.	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Total Stock Equity	1241.3	1114.6	1030.1	955.7	891.6	827.4	773.4	719.5
TOTAL LIABIL. & EQUITY	2837.4	2739.5	2641.7	2543.9	2446.0	2348.2	2250.3	2152.5

<STRIKE ANY KEY TO CONTINUE> *(ESCAPE)*

```
Enter 0 - No More Reports
      1 - Summary
      2 - Balance Sheet
      3 - Income Statement
      4 - Sources and Uses of Funds Statement
      5 - Retained Earnings Statement/Tax Depreciation Worksheet
      6 - Tax Calculation (Book/Regulation Basis)
      7 - Partnership Return & Cashflow Analysis
      8 - Cashflow Summary
      9 - Busbar Cost Plot
      10 - All of Above
      11- Select Printout Years
? 0

Enter 0 - No Listing
      1 - Preview on Screen
      2 - List on Printer
? 0
```

Finally, the user has the option of performing another run or terminating execution. The termination option is selected.

```
Enter 0 - No More Runs
      1 - Do Another Run
? 0

Normal End of Run
Thank you for using BREEDER
```

APPENDIX D: SAMPLE RUNS

D.1 FULL SAMPLE RUN USING COALKP.DAT

D.2 FULL SAMPLE RUN USING LMFBRKP.DAT

D.3 RUN ILLUSTRATING USE OF THE QUARTER (0)/ANNUAL(1) FLAG

D.4 RUN ILLUSTRATING USE OF THE NET (0)/FIX(1) EQUITY FLAG

D.5 RUN ILLUSTRATING USE OF THE COMPOUND (0)/SIMPLE(1) FLAG

D.6 RUN ILLUSTRATING USE OF THE BOT.UP(0)/TOP.DN.(1) FLAG

D.7 RUN ILLUSTRATING USE OF MISCELLANEOUS FLAGS

Along with the BREEDER program and supporting data files, the program disk contains two sample analysis files: COALKP.DAT and LMFBRKP.DAT. These files are derived from two historical analyses performed by Kidder-Peabody for EPRI CoMO. The sole purpose of including these files is to provide the user with a means of verifying the proper execution of the BREEDER program. The Electric Power Research Institute and Argonne National Laboratory do not necessarily endorse the values contained in these files or the results of these analyses.

The user can use either of these by invoking the appropriate system routine to copy one of the sample files into the BREEDER data file, LSPBFILE.DAT:

- For CP/M systems, PIP LSPBFILE.DAT = COALKP.DAT
- For PC-DOS systems, COPY COALKP.DAT LSPBFILE.DAT

In addition to complete program summaries for the two sample files, this section contains five runs illustrating the effects of various program options. In each case, the output formats have been modified slightly to conform to a manuscript line limitation of approximately 55 lines per page, as opposed to 66 lines on the printer. Otherwise, the results in this section faithfully reproduce the hard-copy computer results. Only five years of analysis results are presented (1st, 8th, 16th, 24th, and 30th), to conserve space.

D.1 FULL SAMPLE RUN USING COALKP.DAT

This run simulates the financial analysis of a 1250-MWe, coal-fired electricity-generation plant scheduled to start operation in 1995. The run employs a traditional financing distribution of 50% debt, 35% equity, and 15% preferred stock. The plant is built over six years and has an overnight construction cost of $950/kWe. The analysis uses ACRS, a 10% investment tax credit, and normalization as specified in TEFRA. All short-term debt is capitalized at the end of each year.

Kidder-Peabody Coal Case (1% fuel escal.,1250 MWe,1995 startup) 10/24/83

Years:

Base Year	: 1982.00	Construction Start	: 1989.00
Construction End	: 1995.00	Levelization Period	: 20.00
Debt Repayment Period	: 30.00		

Factors (Decimal):

Equity Fraction	: 0.3500	Debt Fraction	: 0.5000
Ad Valorem Tax	: 0.0000	Property Tax	: 0.0200
Income Tax Rate	: 0.5000	ITC Base Rate	: 0.1000
Target Capacity Factor	: 0.6500	Trigger Capacity Factor	: 0.0000
Target Return on Equity	: 0.1415	Backfit-Replacement Rate	: 0.0000

Flags (0 or 1):

95%(0)/Full(1) Dep. Flag	: 0	StLn(0)/ACRS(1) Dep. Flag	: 1
Flowthru(0)/Norm.(1) Flag	: 1	Uncons(0)/Cons(1)Tax Flag	: 0
Quarter(0)/Annual(1) Flag	: 1	Compound(0)/Simple(1)Flag	: 1
Gross(0)/Net(1)PropTxFlag	: 1	Pre(0)/After(1)TxDiscFlag	: 0
Net(0)/Fix(1) Equity Flag	: 1	Bot.Up(0)/Top.Dn.(1) Flag	: 0

Plant Variables:

Size (MW)	: 1250.00	Book Life (yrs)	: 30.00
Tax Life (yrs)	: 15.00	Startup (million $)	: 0.00
Init.Work.Cap.(million $)	: 0.00	Heat Rate (Btu/kWh)	:10150.00

Rates (Decimal):

General Inflation:

1982	0.0350	1983	0.0370	1984	0.0580	1985	0.0660	1986	0.0650
1987	0.0610	1988	0.0620	1989	0.0650	1990	0.0690	1991	0.0680
1992	0.0660	1993	0.0650	1994	0.0640	1995	0.0620	1996	0.0620
1997	0.0620	1998	0.0620	1999	0.0620	2000	0.0620	2001	0.0620
2002	0.0620	2003	0.0620	2004	0.0620	2005	0.0620	2006	0.0620
2007	0.0620	2008	0.0620	2009	0.0620	2010	0.0620	2011	0.0620

Short-Term Debt:

1982	0.1477	1983	0.1166	1984	0.1186	1985	0.1225	1986	0.1272
1987	0.1182	1988	0.1121	1989	0.1080	1990	0.1026	1991	0.1001
1992	0.0982	1993	0.0971	1994	0.0962	1995	0.0943	1996	0.0943
1997	0.0943	1998	0.0943	1999	0.0943	2000	0.0943	2001	0.0943
2002	0.0943	2003	0.0943	2004	0.0943	2005	0.0943	2006	0.0943
2007	0.0943	2008	0.0943	2009	0.0943	2010	0.0943	2011	0.0943

Long-Term Debt:

1982	0.1432	1983	0.1065	1984	0.1038	1985	0.1052	1986	0.1062
1987	0.1061	1988	0.1034	1989	0.0997	1990	0.0977	1991	0.0980
1992	0.0984	1993	0.0976	1994	0.0970	1995	0.0968	1996	0.0968
1997	0.0968	1998	0.0968	1999	0.0968	2000	0.0968	2001	0.0968
2002	0.0968	2003	0.0968	2004	0.0968	2005	0.0968	2006	0.0968
2007	0.0968	2008	0.0968	2009	0.0968	2010	0.0968	2011	0.0968

Preferred Stock:

1982	0.1442	1983	0.1064	1984	0.1036	1985	0.1051	1986	0.1061
1987	0.1060	1988	0.1032	1989	0.0995	1990	0.0974	1991	0.0977
1992	0.0981	1993	0.0973	1994	0.0963	1995	0.0965	1996	0.0965
1997	0.0965	1998	0.0965	1999	0.0965	2000	0.0965	2001	0.0965
2002	0.0965	2003	0.0965	2004	0.0965	2005	0.0965	2006	0.0965
2007	0.0965	2008	0.0965	2009	0.0965	2010	0.0965	2011	0.0965

Equity:

1982	0.1892	1983	0.1514	1984	0.1486	1985	0.1501	1986	0.1511
1987	0.1510	1988	0.1482	1989	0.1445	1990	0.1424	1991	0.1427
1992	0.1431	1993	0.1423	1994	0.1417	1995	0.1415	1996	0.1415
1997	0.1415	1998	0.1415	1999	0.1415	2000	0.1415	2001	0.1415
2002	0.1415	2003	0.1415	2004	0.1415	2005	0.1415	2006	0.1415
2007	0.1415	2008	0.1415	2009	0.1415	2010	0.1415	2011	0.1415
2012	0.1415	2013	0.1415	2014	0.1415	2015	0.1415	2016	0.1415
2017	0.1415	2018	0.1415	2019	0.1415	2020	0.1415	2021	0.1415
2022	0.1415	2023	0.1415	2024	0.1415	2025	0.1415	2026	0.1415
2027	0.1415	2028	0.1415	2029	0.1415	2030	0.1415	2031	0.1415

Plant Arrays:

Capacity Factor (Decimal):

1995	0.6500	1996	0.6500	1997	0.6500	1998	0.6500	1999	0.6500
2000	0.6500	2001	0.6500	2002	0.6500	2003	0.6500	2004	0.6500
2005	0.6500	2006	0.6500	2007	0.6500	2008	0.6500	2009	0.6500
2010	0.6500	2011	0.6500	2012	0.6500	2013	0.6500	2014	0.6500
2015	0.6500	2016	0.6500	2017	0.6500	2018	0.6500	2019	0.6500
2020	0.6500	2021	0.6500	2022	0.6500	2023	0.6500	2024	0.6500

Fuel Cost ($/MMBtu):

1995	2.2875	1996	2.3104	1997	2.3335	1998	2.3568	1999	2.3804
2000	2.4042	2001	2.4282	2002	2.4525	2003	2.4770	2004	2.5018
2005	2.5268	2006	2.5521	2007	2.5776	2008	2.6034	2009	2.6294
2010	2.6557	2011	2.6823	2012	2.7091	2013	2.7362	2014	2.7635
2015	2.7912	2016	2.8191	2017	2.8473	2018	2.8758	2019	2.9045
2020	2.9336	2021	2.9629	2022	2.9925	2023	3.0224	2024	3.0527

O&M (mills/kWh):

1995	6.10	1996	6.10	1997	6.10	1998	6.10	1999	6.10
2000	6.10	2001	6.10	2002	6.10	2003	6.10	2004	6.10
2005	6.10	2006	6.10	2007	6.10	2008	6.10	2009	6.10
2010	6.10	2011	6.10	2012	6.10	2013	6.10	2014	6.10
2015	6.10	2016	6.10	2017	6.10	2018	6.10	2019	6.10
2020	6.10	2021	6.10	2022	6.10	2023	6.10	2024	6.10

Working Cap. (million $):

1995	0.00	1996	0.00	1997	0.00	1998	0.00	1999	0.00
2000	0.00	2001	0.00	2002	0.00	2003	0.00	2004	0.00
2005	0.00	2006	0.00	2007	0.00	2008	0.00	2009	0.00
2010	0.00	2011	0.00	2012	0.00	2013	0.00	2014	0.00
2015	0.00	2016	0.00	2017	0.00	2018	0.00	2019	0.00
2020	0.00	2021	0.00	2022	0.00	2023	0.00	2024	0.00

Federal Sub. (million $):

1995	0.00	1996	0.00	1997	0.00	1998	0.00	1999	0.00
2000	0.00	2001	0.00	2002	0.00	2003	0.00	2004	0.00
2005	0.00	2006	0.00	2007	0.00	2008	0.00	2009	0.00
2010	0.00	2011	0.00	2012	0.00	2013	0.00	2014	0.00
2015	0.00	2016	0.00	2017	0.00	2018	0.00	2019	0.00
2020	0.00	2021	0.00	2022	0.00	2023	0.00	2024	0.00

Busbar Costs (cents/kWh):

1995	0.00	1996	0.00	1997	0.00	1998	0.00	1999	0.00
2000	0.00	2001	0.00	2002	0.00	2003	0.00	2004	0.00
2005	0.00	2006	0.00	2007	0.00	2008	0.00	2009	0.00
2010	0.00	2011	0.00	2012	0.00	2013	0.00	2014	0.00
2015	0.00	2016	0.00	2017	0.00	2018	0.00	2019	0.00
2020	0.00	2021	0.00	2022	0.00	2023	0.00	2024	0.00

Construction Expenses (million$/yr):

Plant:

1989	49.00	1990	165.00	1991	327.00	1992	328.00	1993	232.00
1994	88.00								

Kidder-Peabody Coal Case (1% fuel escal.,1250 MWe,1995 startup) 10/24/83

Quarterly Construction Expenses (millions of current $)

	--Direct Construction--			-----Interest and Equity Charges-----					Total
	ON Cost	Infl.	Total	ST debt	LT debt	Pref.	Common	Total	Cost
1989									
Qrt. 1	12.3	5.8	18.1	0.0	0.0	0.0	0.0	0.0	18.1
Qrt. 2	12.3	6.1	18.4	0.5	0.0	0.0	0.0	0.5	18.9
Qrt. 3	12.3	6.4	18.7	1.0	0.0	0.0	0.0	1.0	19.7
Qrt. 4	12.3	6.7	19.0	1.5	0.0	0.0	0.0	1.5	20.5
1990									
Qrt. 1	41.3	23.6	64.9	2.1	0.0	0.0	0.0	2.1	67.0
Qrt. 2	41.3	24.7	66.0	1.7	0.0	0.3	1.0	3.0	68.9
Qrt. 3	41.3	25.8	67.1	3.5	0.0	0.3	1.0	4.7	71.8
Qrt. 4	41.3	27.0	68.2	5.3	0.0	0.3	1.0	6.6	74.8
1991									
Qrt. 1	81.8	55.7	137.4	7.2	3.8	0.3	1.0	12.3	149.7
Qrt. 2	81.8	58.0	139.7	3.7	0.0	1.3	4.5	9.6	149.3
Qrt. 3	81.8	60.3	142.0	7.5	0.0	1.3	4.5	13.3	155.3
Qrt. 4	81.8	62.6	144.4	11.4	0.0	1.3	4.5	17.2	161.6
1992									
Qrt. 1	82.0	65.2	147.2	15.4	17.6	1.3	4.5	38.8	186.0
Qrt. 2	82.0	67.5	149.5	4.6	0.0	3.6	12.2	20.4	169.9
Qrt. 3	82.0	69.9	151.9	8.7	0.0	3.6	12.2	24.5	176.5
Qrt. 4	82.0	72.4	154.4	13.1	0.0	3.6	12.2	28.9	183.3
1993									
Qrt. 1	58.0	52.9	110.9	17.6	47.9	3.6	12.2	81.3	192.2
Qrt. 2	58.0	54.7	112.7	4.7	0.0	6.2	21.1	31.9	144.6
Qrt. 3	58.0	56.5	114.5	8.2	0.0	6.2	21.1	35.4	149.9
Qrt. 4	58.0	58.3	116.3	11.8	0.0	6.2	21.1	39.1	155.4
1994									
Qrt. 1	22.0	22.8	44.8	15.6	82.8	6.2	21.1	125.7	170.5
Qrt. 2	22.0	23.5	45.5	4.1	0.0	8.5	28.9	41.5	87.0
Qrt. 3	22.0	24.2	46.2	6.2	0.0	8.5	28.9	43.6	89.8
Qrt. 4	22.0	24.9	46.9	8.4	0.0	8.5	28.9	45.8	92.7

Kidder-Peabody Coal Case (1% fuel escal.,1250 MWe,1995 startup) 10/24/83

Quarterly Construction Financing (millions of current $)

	--------Source of Funds--------				----Quarter----		------Cumulative------		
	ST debt	Lt debt	Pref.	Common	CWIP	AFUDC	CWIP	AFUDC	Total
1989									
Qrt. 1	18.1	0.0	0.0	0.0	18.1	0.0	18.1	0.0	18.1
Qrt. 2	18.9	0.0	0.0	0.0	18.4	0.5	36.4	0.5	36.9
Qrt. 3	19.7	0.0	0.0	0.0	18.7	1.0	55.1	1.5	56.6
Qrt. 4	20.5	0.0	0.0	0.0	19.0	1.5	74.0	3.0	77.1
1990									
Qrt. 1	-10.1	38.5	11.6	27.0	64.9	2.1	138.9	5.1	144.0
Qrt. 2	68.9	0.0	0.0	0.0	66.0	3.0	204.9	8.1	213.0
Qrt. 3	71.8	0.0	0.0	0.0	67.1	4.7	272.0	12.8	284.8
Qrt. 4	74.8	0.0	0.0	0.0	68.2	6.6	340.2	19.4	359.6
1991									
Qrt. 1	-132.8	141.3	42.4	98.9	137.4	12.3	477.7	31.6	509.3
Qrt. 2	149.3	0.0	0.0	0.0	139.7	9.6	617.4	41.2	658.5
Qrt. 3	155.3	0.0	0.0	0.0	142.0	13.3	759.4	54.4	813.9
Qrt. 4	161.6	0.0	0.0	0.0	144.4	17.2	903.8	71.6	975.4
1992									
Qrt. 1	-429.9	307.9	92.4	215.5	147.2	38.8	1051.0	110.4	1161.4
Qrt. 2	169.9	0.0	0.0	0.0	149.5	20.4	1200.5	130.8	1331.3
Qrt. 3	176.5	0.0	0.0	0.0	151.9	24.5	1352.4	155.3	1507.8
Qrt. 4	183.3	0.0	0.0	0.0	154.4	28.9	1506.8	184.2	1691.0
1993									
Qrt. 1	-523.4	357.8	107.3	250.5	110.9	81.3	1617.8	265.5	1883.2
Qrt. 2	144.6	0.0	0.0	0.0	112.7	31.9	1730.5	297.4	2027.9
Qrt. 3	149.9	0.0	0.0	0.0	114.5	35.4	1845.0	332.8	2177.8
Qrt. 4	155.4	0.0	0.0	0.0	116.3	39.1	1961.3	371.9	2333.1
1994									
Qrt. 1	-471.6	321.0	96.3	224.7	44.8	125.7	2006.1	497.5	2503.6
Qrt. 2	87.0	0.0	0.0	0.0	45.5	41.5	2051.6	539.1	2590.6
Qrt. 3	89.8	0.0	0.0	0.0	46.2	43.6	2097.8	582.7	2680.5
Qrt. 4	92.7	0.0	0.0	0.0	46.9	45.8	2144.7	628.5	2773.2

Kidder-Peabody Coal Case (1% fuel escal.,1250 MWe,1995 startup) 10/24/83

Annual Construction Expenses (millions of current $)

Year	--Direct Construction-- ON Cost	Infl.	Total	-----Interest and Equity Charges----- ST debt	LT debt	Pref.	Common	Total	Total Cost
1989	49.0	25.0	74.0	3.0	0.0	0.0	0.0	3.0	77.1
1990	165.0	101.2	266.2	12.6	0.0	0.8	2.9	16.3	282.5
1991	327.0	236.6	563.6	29.8	3.8	4.2	14.4	52.3	615.8
1992	328.0	275.0	603.0	41.8	17.6	12.1	41.1	112.6	715.6
1993	232.0	222.4	454.4	42.2	47.9	22.2	75.4	187.7	642.1
1994	88.0	95.5	183.5	34.2	82.8	31.7	107.8	256.6	440.1
Accru:	0.0	0.0	0.0	10.6	114.0	8.5	28.9	162.0	162.0
Total:	1189.0	955.7	2144.7	174.3	266.1	79.5	270.6	790.5	2935.2

Annual Construction Financing (millions of current $)

Year	--------Source of Funds-------- ST debt	Lt debt	Pref.	Common	----Annual----- CWIP	AFUDC	------Cumulative------ CWIP	AFUDC	Total
1989	77.1	0.0	0.0	0.0	74.0	3.0	74.0	3.0	77.1
1990	205.5	38.5	11.6	27.0	266.2	16.3	340.2	19.4	359.6
1991	333.3	141.3	42.4	98.9	563.6	52.3	903.8	71.6	975.4
1992	99.8	307.9	92.4	215.5	603.0	112.6	1506.8	184.2	1691.0
1993	-73.5	357.8	107.3	250.5	454.4	187.7	1961.3	371.9	2333.1
1994	-202.0	321.0	96.3	224.7	183.5	256.6	2144.7	628.5	2773.2
Accru:	-440.1	301.0	90.3	210.7	0.0	162.0	2144.7	790.5	2935.2
Total:	-0.0	1467.6	440.3	1027.3	2144.7	790.5	2144.7	790.5	2935.2

Construction-End Summary

Accruals (million $) Inv.Tx.Cr.	Tax Sav.	------Direct------ Current	Base Year	------AFUDC------- Current	Base Year	---Weighted Rate--- Preferred	LT debt
214.5	220.2	2144.7	1189.0	790.5	398.8	0.09713	0.09751

Kidder-Peabody Coal Case (1% fuel escal.,1250 MWe,1995 startup) 10/24/83

CONSTRUCTION SUMMARY
(All Amounts in millions of current year $)

Period: (1989,1995)
Base Year: 1982

Expenses:	Total	Book Life	Book Dep.
Direct :	2144.7	30	71.5
Startup:	0.0	30	0.0
AFUDC :	790.5	30	26.4
Wk.Cap.:	0.0	--	--
Total :	2935.2	--	97.8

Financing:	Total	% Financing	Charge Rate
Common :	1027.3	35.00	14.15
Preferred:	440.3	15.00	9.71
LT.Debt :	1467.6	50.00	9.75
Total :	2935.2	100.00	11.29

Accruals:
ITC: 214.5
FIT: 220.2

OPERATING SUMMARY

Acceleration Tax Depreciation:
Tax Life : 15 (years)
Method : ACRS (TEFRA version)
Basis : 2037.5
TEFRA Opt.: 95 %

Option Flags
IDC Compounding : Simple
Financing Period : Annual
Property Tax Basis : Net
Equity Finan. Basis: Fixed %
Discount Rate : Pre-tax
Analysis Type : Bottom-up
Accounting Basis : Normalized
Reg. Tax Treatment : Unconsolidated

Analysis Summary:	ROI (%)	Lev. Busbar (cents/kWh)	Discount Rate (%)	Fixed Charge Rate (%)
Current:	15.75	17.14	11.29	16.08
Base Yr:	8.82	4.95	4.79	8.58

Kidder-Peabody Coal Case (1% fuel escal.,1250 MWe,1995 startup) 10/24/83

BALANCE SHEET
(Millions of Current Year $)

	1995	2002	2010	2018	2024
ASSETS					
Cum. Working Capital	0.0	0.0	0.0	0.0	0.0
Gross Plant	2144.7	2144.7	2144.7	2144.7	2144.7
Less Accum.Plant Dep.	71.5	571.9	1143.8	1715.8	2144.7
Net Plant	2073.2	1572.8	1000.9	428.9	0.0
Capitalized AFUDC	790.5	790.5	790.5	790.5	790.5
Less Accum.AFUDC Dep.	26.4	210.8	421.6	632.4	790.5
Net Capitalized AFUDC	764.2	579.7	368.9	158.1	0.0
TOTAL ASSETS	2837.4	2152.5	1369.8	587.0	0.0
LIABILITIES					
LTD Due in Next Year	48.9	48.9	48.9	48.9	0.0
LTD Outstanding	1369.8	1027.3	636.0	244.6	0.0
Cum. Deferred ITC	175.4	157.3	100.1	42.9	-0.0
Cum. Deferred FIT	2.0	199.5	236.0	0.0	0.0
P.S. Due in Next Year	14.7	14.7	14.7	14.7	0.0
Pref.Stock Outstanding	410.9	308.2	190.8	73.4	0.0
End Common Equity	815.7	396.6	138.7	162.6	-0.0
Partner Loans Outst.	0.0	0.0	4.6	0.0	0.0
Total Stock Equity	1241.3	719.5	348.8	250.6	-0.0
TOTAL LIABIL. & EQUITY	2837.4	2152.5	1369.8	587.0	0.0

Kidder-Peabody Coal Case (1% fuel escal.,1250 MWe,1995 startup) 10/24/83

INCOME STATEMENT
(Millions of Current Year $)

	1995	2002	2010	2018	2024
REVENUES					
Electric Sales	1127.6	1173.4	1544.2	2414.6	3429.6
Federal Subsidy	0.0	0.0	0.0	0.0	0.0
TOTAL REVENUES	1127.6	1173.4	1544.2	2414.6	3429.6
EXPENSES					
Fuel Expense	363.4	593.6	1040.1	1822.4	2775.3
O&M Expense	95.5	145.5	235.4	380.8	546.4
Backfit&Replacement	0.0	0.0	0.0	0.0	0.0
Ad Valorem Tax	0.0	0.0	0.0	0.0	0.0
Total Oper. Expense	458.9	739.1	1275.5	2203.2	3321.7
Plant Book Dep.	71.5	71.5	71.5	71.5	71.5
AFUDC Book Dep.	26.4	26.4	26.4	26.4	26.4
Total Book Dep.	97.8	97.8	97.8	97.8	97.8
Property Tax	58.7	45.0	29.4	13.7	2.0
Net FIT (Book Basis)	181.0	87.3	31.4	29.7	-1.9
Total Pre-Debt Expen.	796.4	969.2	1434.1	2344.4	3419.6
Pre-Debt Net Income	331.2	204.2	110.1	70.2	10.0
LTD Interest	143.1	109.7	71.6	33.4	4.8
NET INCOME	188.1	94.5	38.6	36.8	5.3
Less Pref. Dividends	42.8	32.8	21.4	10.0	1.4
Common Stock Earnings	145.4	61.7	17.2	26.8	3.9
BUSBAR COST (cents/kWh)	15.8	16.5	21.7	33.9	1.4

Kidder-Peabody Coal Case (1% fuel escal.,1250 MWe,1995 startup) 10/24/83

SOURCES AND USES OF FUNDS STATEMENT
(Millions of Current Year $)

	1995	2002	2010	2018	2024
SOURCES OF FUNDS					
Net Income	188.1	94.5	38.6	36.8	5.3
Plant Book Dep.	71.5	71.5	71.5	71.5	71.5
AFUDC Book Dep.	26.4	26.4	26.4	26.4	26.4
Total Book Dep.	97.8	97.8	97.8	97.8	97.8
Deferred ITC	175.4	-7.1	-7.1	-7.1	-7.1
Deferred FIT	2.0	12.2	-48.9	0.0	0.0
TOTAL INTERNAL	463.4	197.4	80.3	127.5	96.0
Partner Loans	0.0	0.0	4.6	0.0	0.0
Government Loans	0.0	0.0	0.0	0.0	0.0
TOTAL SOURCES	463.4	197.4	85.0	127.5	96.0
USES OF FUNDS					
Preferred Stock Due	14.7	14.7	14.7	14.7	14.7
Preferred Dividends	42.8	32.8	21.4	10.0	1.4
Common Dividends	357.0	101.0	0.0	53.9	30.9
Working Capital Change	0.0	0.0	0.0	0.0	0.0
LTD Repayment	48.9	48.9	48.9	48.9	48.9
Partner Loan Repayment	0.0	0.0	0.0	0.0	0.0
TOTAL USES	463.4	197.4	85.0	127.5	96.0

Kidder-Peabody Coal Case (1% fuel escal.,1250 MWe,1995 startup) 10/24/83

RETAINED EARNINGS STATEMENT & TAX DEP. WORKSHEET
(Millions of Current Year $)

	1995	2002	2010	2018	2024
RETAINED EARNINGS					
Begin.Common Equity	1027.3	435.9	121.5	189.7	27.1
Common Stock Earnings	145.4	61.7	17.2	26.8	3.8
Total Common Equity	1172.7	497.6	138.7	216.5	30.9
Less Common Dividends	357.0	101.0	0.0	53.9	30.9
END COMMON EQUITY	815.7	396.6	138.7	162.6	-0.0
RETURN ON EQUITY (%)	14.1	14.1	14.1	14.1	14.1
TAX DEP. WORKSHEET					
Startup Dep.	0.0	0.0	0.0	0.0	0.0
5-year Prop.Dep.	0.0	0.0	0.0	0.0	0.0
10-year Prop.Dep.	0.0	0.0	0.0	0.0	0.0
15-year Prop.Dep.	101.9	122.2	0.0	0.0	0.0
Straight-line Dep.	0.0	0.0	0.0	0.0	0.0
TOTAL TAX DEPRECIATION	101.9	122.2	0.0	0.0	0.0

Kidder-Peabody Coal Case (1% fuel escal.,1250 MWe,1995 startup) 10/24/83

TAX CALCULATION (Book/Regulation Basis)
(Millions of Current Year $)

	1995	2002	2010	2018	2024
BOOK TAX CALCULATION					
Electric Sales	1127.6	1173.4	1544.2	2414.6	3429.6
Total Oper. Expense	458.9	739.1	1275.5	2203.2	3321.7
Total Book Dep.	97.8	97.8	97.8	97.8	97.8
Property Tax	58.7	45.0	29.4	13.7	2.0
LTD Interest	143.1	109.7	71.6	33.4	4.8
Pre-tax Book Income	376.3	188.9	77.1	73.6	10.5
Book FIT	188.1	94.5	38.6	36.8	5.3
Less amort.ITC	7.1	7.1	7.1	7.1	7.1
NET FIT (Book Basis)	181.0	87.3	31.4	29.7	-1.9
REGULATORY TAX CALCULATION					
Electric Sales	1127.6	1173.4	1544.2	2414.6	3429.6
Total Oper. Expense	458.9	739.1	1275.5	2203.2	3321.7
Total Tax Depreciation	101.9	122.2	0.0	0.0	0.0
Property Tax	58.7	45.0	29.4	13.7	2.0
LTD Interest	143.1	109.7	71.6	33.4	4.8
Pre-tax Part. Income	365.1	157.4	167.8	164.3	101.2
FIT Due	182.5	78.7	83.9	82.2	50.6
Less ITC Taken	182.5	0.0	0.0	0.0	0.0
Less FIT Taken	0.0	0.0	0.0	0.0	0.0
NET REGULATED FIT	0.0	78.7	83.9	82.2	50.6

Kidder-Peabody Coal Case (1% fuel escal.,1250 MWe,1995 startup) 10/24/83

PARTNERSHIP RETURN AND CASHFLOW ANALYSIS
(Millions of Current Year $)

	1995	2002	2010	2018	2024
PARTNERSHIP RETURN					
Electric Sales	1127.6	1173.4	1544.2	2414.6	3429.6
Total Oper. Expense	458.9	739.1	1275.5	2203.2	3321.7
Total Tax Depreciation	101.9	122.2	0.0	0.0	0.0
Property Tax	58.7	45.0	29.4	13.7	2.0
LTD Interest	143.1	109.7	71.6	33.4	4.8
Pre-tax Part. Income	365.1	157.4	167.8	164.3	101.2
FIT Due	182.5	78.7	83.9	82.2	50.6
PARTNERSHIP CASHFLOW ANALYSIS					
After-tax Cash	182.5	78.7	83.9	82.2	50.6
Total Tax Depreciation	101.9	122.2	0.0	0.0	0.0
Federal Subsidy	0.0	0.0	0.0	0.0	0.0
Total Cash Avail.	284.4	200.9	83.9	82.2	50.6
Less LTD Repayment	48.9	48.9	48.9	48.9	48.9
Less Pref. Dividends	14.7	14.7	14.7	14.7	14.7
Less Pref. Stock Due	42.8	32.8	21.4	10.0	1.4
Less Partner Loans	0.0	0.0	4.6	0.0	0.0
Less Work.Cap. Change	0.0	0.0	0.0	0.0	0.0
NET TO PARTNERS	178.1	104.6	-1.1	8.6	[illegible]
CUMULATIVE TO PARTNERS	-414.6	586.6	1161.7	1237.2	[illegible]

Kidder-Peabody Coal Case (1% fuel escal.,1250 MWe,1995 startup) 10/24/83

AFTER-TAX CASH FLOWS TO UTILITY INVESTORS
(Millions of Current Year $)

Year	During Construction Cash Inv.	Tax Ben.	During Operation Tax Due	Cash Dist.	Net A.T. Cash Flow	Cum.A.T. Cash Flow	Cm.Dis.R. A.T.C.F.
1989	26.971	9.951	--	--	-17.019	-17.019	-7.589
1990	98.880	37.389	--	--	-61.491	-78.510	-32.157
1991	215.547	84.166	--	--	-131.381	-209.891	-79.017
1992	250.466	106.228	--	--	-144.238	-354.129	-124.987
1993	224.733	106.980	--	--	-117.753	-471.882	-158.583
1994	210.731	89.949	--	--	-120.782	-592.663	-189.462
1995	--	--	182.546	360.600	178.053	-414.610	-148.634
1996	--	--	100.232	299.274	199.042	-215.569	-107.622
1997	--	--	93.149	266.158	173.010	-42.559	-75.589
1998	--	--	92.025	244.962	152.937	110.378	-50.144
1999	--	--	92.343	226.648	134.305	244.683	-30.064
2000	--	--	83.915	211.217	127.303	371.985	-12.962
2001	--	--	85.674	195.787	110.113	482.098	0.331
2002	--	--	78.687	183.239	104.552	586.649	11.672
2003	--	--	71.701	170.691	98.991	685.640	21.322
2004	--	--	64.714	158.144	93.430	779.070	29.506
2005	--	--	57.728	145.597	87.869	866.939	36.422
2006	--	--	50.741	133.049	82.308	949.247	42.244
2007	--	--	43.754	120.501	76.747	1025.994	47.121
2008	--	--	36.768	107.954	71.186	1097.180	51.187
2009	--	--	29.781	95.407	65.625	1162.806	54.555
2010	--	--	83.920	82.859	-1.061	1161.745	54.506
2011	--	--	84.927	86.298	1.371	1163.116	54.563
2012	--	--	86.278	90.426	4.148	1167.264	54.717
2013	--	--	88.023	95.340	7.317	1174.581	54.962
2014	--	--	90.215	101.151	10.935	1185.517	55.291
2015	--	--	92.235	106.616	14.381	1199.898	55.679
2016	--	--	92.683	108.937	16.254	1216.152	56.074
2017	--	--	87.424	99.844	12.420	1228.572	56.345
2018	--	--	82.164	90.751	8.586	1237.158	56.513
2019	--	--	76.904	81.657	4.752	1241.911	56.597
2020	--	--	71.645	72.563	0.918	1242.829	56.612
2021	--	--	66.385	63.469	-2.916	1239.913	56.570
2022	--	--	61.126	54.376	-6.750	1233.164	56.484
2023	--	--	55.866	45.282	-10.584	1222.580	56.362
2024	--	--	50.606	36.189	-14.418	1208.162	56.213
TOT.	1027.328	434.664	2334.159	4134.986	1208.162		

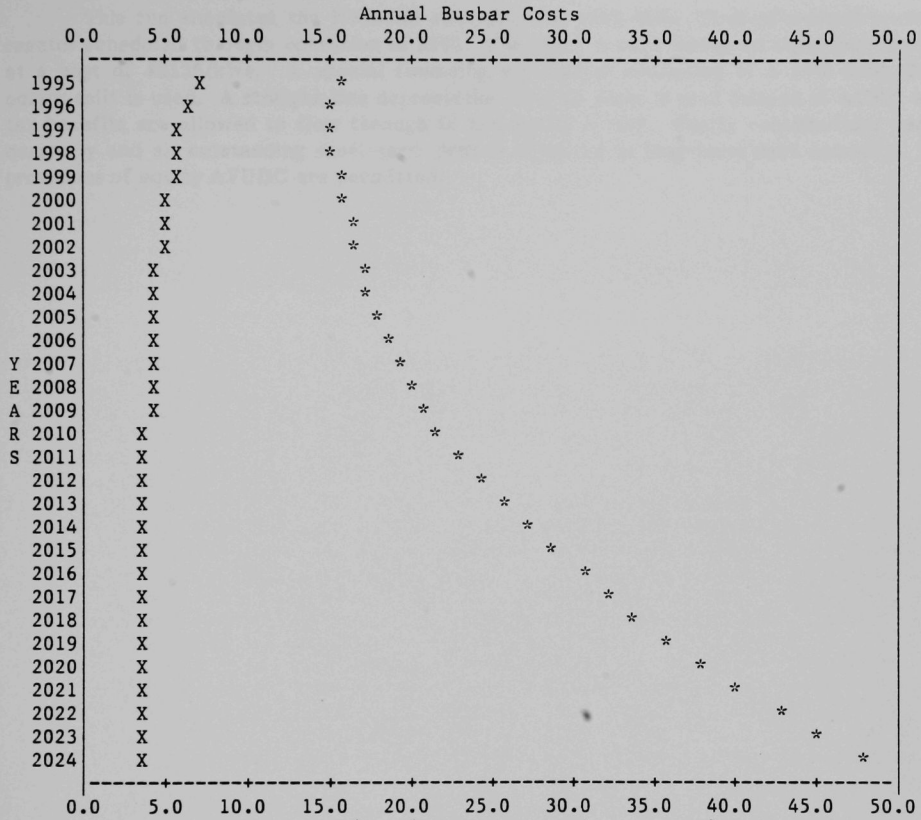
Summary of ROIs (Current $)

1995	-38.41	1996	-12.71	1997	-1.90	1998	4.00	1999	7.55
2000	9.97	2001	11.53	2002	12.68	2003	13.53	2004	14.17
2005	14.65	2006	15.02	2007	15.30	2008	15.52	2009	15.68
2010	15.68	2011	15.68	2012	15.69	2013	15.70	2014	15.71
2015	15.73	2016	15.74	2017	15.75	2018	15.75	2019	15.76
2020	15.76	2021	15.76	2022	15.75	2023	15.75	2024	15.75

Summary of ROIs (Base-Year $)

1995	-42.16	1996	-18.01	1997	-7.84	1998	-2.29	1999	1.06
2000	3.34	2001	4.82	2002	5.91	2003	6.71	2004	7.32
2005	7.78	2006	8.12	2007	8.39	2008	8.60	2009	8.75
2010	8.75	2011	8.75	2012	8.76	2013	8.77	2014	8.78
2015	8.80	2016	8.81	2017	8.82	2018	8.82	2019	8.83
2020	8.83	2021	8.83	2022	8.82	2023	8.82	2024	8.82

Kidder-Peabody Coal Case (1% fuel escal.,1250 MWe,1995 startup) 10/24/83



Summary of Levelized Busbar Costs
(Levelized over 20 years)

17.14 (current cents/kWh,discount rate : 11.285 (%))
4.95 (base yr. cents/kWh,discount rate: 4.788 (%))

Summary of Busbar Costs (current cents/kWh)

1995	15.84	1996	15.32	1997	15.23	1998	15.34	1999	15.52
2000	15.79	2001	16.09	2002	16.49	2003	16.93	2004	17.42
2005	17.97	2006	18.58	2007	19.25	2008	19.99	2009	20.80
2010	21.70	2011	22.90	2012	24.20	2013	25.60	2014	27.12
2015	28.75	2016	30.45	2017	32.12	2018	33.93	2019	35.87
2020	37.98	2021	40.25	2022	42.70	2023	45.34	2024	48.19

Summary of Busbar Costs (base yr. cents/kWh)

1995	6.99	1996	6.37	1997	5.96	1998	5.65	1999	5.39
2000	5.16	2001	4.95	2002	4.78	2003	4.62	2004	4.47
2005	4.35	2006	4.23	2007	4.13	2008	4.04	2009	3.96
2010	3.88	2011	3.86	2012	3.84	2013	3.83	2014	3.82
2015	3.81	2016	3.80	2017	3.77	2018	3.75	2019	3.74
2020	3.73	2021	3.72	2022	3.71	2023	3.71	2024	3.72

D.2 FULL SAMPLE RUN USING LMFBRKP.DAT

This run simulates the financial analysis of a 1250 MWe "first-of-a-kind" breeder reactor scheduled to begin operation in 1995. The plant is built during an eight-year period at a cost of $2135/kWe. A special financing alternative consisting of a 75% debt, 25% equity split is used. A straight-line depreciation over 25 years is used instead of ACRS, and tax benefits are allowed to flow through to the equity holder. Equity contributions occur quarterly and all outstanding short-term debt is coverted to long-term debt annually. No provisions of equity AFUDC are permitted.

Kidder-Peabody LMFBR Base Case 10/25/83

Years:

Base Year	: 1982.00	Construction Start	: 1987.00
Construction End	: 1995.00	Levelization Period	: 20.00
Debt Repayment Period	: 30.00		

Factors (Decimal):

Equity Fraction	: 0.2500	Debt Fraction	: 0.7500
Ad Valorem Tax	: 0.0000	Property Tax	: 0.0200
Income Tax Rate	: 0.5000	ITC Base Rate	: 0.1000
Target Capacity Factor	: 0.6500	Trigger Capacity Factor	: 0.0000
Target Return on Equity	: 0.1415	Backfit-Replacement Rate	: 0.0000

Flags (0 or 1):

95%(0)/Full(1) Dep. Flag	: 0	StLn(0)/ACRS(1) Dep. Flag	: 0
Flowthru(0)/Norm.(1) Flag	: 0	Uncons(0)/Cons(1)Tax Flag	: 0
Quarter(0)/Annual(1) Flag	: 0	Compound(0)/Simple(1)Flag	: 1
Gross(0)/Net(1)PropTxFlag	: 1	Pre(0)/After(1)TxDiscFlag	: 0
Net(0)/Fix(1) Equity Flag	: 1	Bot.Up(0)/Top.Dn.(1) Flag	: 0

Plant Variables:

Size (MW)	: 1250.00	Book Life (yrs)	: 30.00
Tax Life (yrs)	: 25.00	Startup (millions $)	: 0.00
Init.Work.Cap.(million $)	: 0.00	Heat Rate (Btu/kWh)	:10400.00

Rates (Decimal):

General Inflation:

1982	0.0350	1983	0.0370	1984	0.0580	1985	0.0660	1986	0.0650
1987	0.0610	1988	0.0620	1989	0.0650	1990	0.0690	1991	0.0680
1992	0.0660	1993	0.0650	1994	0.0640	1995	0.0620	1996	0.0620
1997	0.0620	1998	0.0620	1999	0.0620	2000	0.0620	2001	0.0620
2002	0.0620	2003	0.0620	2004	0.0620	2005	0.0620	2006	0.0620
2007	0.0620	2008	0.0620	2009	0.0620	2010	0.0620	2011	0.0620

Short-Term Debt:

1982	0.1177	1983	0.0866	1984	0.0886	1985	0.0925	1986	0.0972
1987	0.0882	1988	0.0821	1989	0.0780	1990	0.0790	1991	0.0780
1992	0.0760	1993	0.0750	1994	0.0740	1995	0.0720	1996	0.0720
1997	0.0720	1998	0.0720	1999	0.0720	2000	0.0720	2001	0.0720
2002	0.0720	2003	0.0720	2004	0.0720	2005	0.0720	2006	0.0720
2007	0.0720	2008	0.0720	2009	0.0720	2010	0.0720	2011	0.0720

Long-Term Debt:

1982	0.1388	1983	0.1063	1984	0.1041	1985	0.1055	1986	0.1065
1987	0.1064	1988	0.1038	1989	0.1003	1990	0.0983	1991	0.0987
1992	0.0989	1993	0.0982	1994	0.0976	1995	0.0975	1996	0.0975
1997	0.0975	1998	0.0975	1999	0.0975	2000	0.0975	2001	0.0975
2002	0.0975	2003	0.0975	2004	0.0975	2005	0.0975	2006	0.0975
2007	0.0975	2008	0.0975	2009	0.0975	2010	0.0975	2011	0.0975

Preferred Stock:

1982	0.1442	1983	0.1064	1984	0.1036	1985	0.1051	1986	0.1061
1987	0.1060	1988	0.1032	1989	0.0995	1990	0.0974	1991	0.0977
1992	0.0981	1993	0.0973	1994	0.0963	1995	0.0965	1996	0.0965
1997	0.0965	1998	0.0965	1999	0.0965	2000	0.0965	2001	0.0965
2002	0.0965	2003	0.0965	2004	0.0965	2005	0.0965	2006	0.0965
2007	0.0965	2008	0.0965	2009	0.0965	2010	0.0965	2011	0.0965

Equity:

1982	0.0000	1983	0.0000	1984	0.0000	1985	0.0000	1986	0.0000
1987	0.0000	1988	0.0000	1989	0.0000	1990	0.0000	1991	0.0000
1992	0.0000	1993	0.0000	1994	0.0000	1995	0.1415	1996	0.1415
1997	0.1415	1998	0.1415	1999	0.1415	2000	0.1415	2001	0.1415
2002	0.1415	2003	0.1415	2004	0.1415	2005	0.1415	2006	0.1415
2007	0.1415	2008	0.1415	2009	0.1415	2010	0.1415	2011	0.1415
2012	0.1415	2013	0.1415	2014	0.1415	2015	0.1415	2016	0.1415
2017	0.1415	2018	0.1415	2019	0.1415	2020	0.1415	2021	0.1415
2022	0.1415	2023	0.1415	2024	0.1415	2025	0.1415	2026	0.1415
2027	0.1415	2028	0.1415	2029	0.1415	2030	0.1415	2031	0.1415

Plant Arrays:

Capacity Factor (Decimal):

1995	0.6500	1996	0.6500	1997	0.6500	1998	0.6500	1999	0.6500
2000	0.6500	2001	0.6500	2002	0.6500	2003	0.6500	2004	0.6500
2005	0.6500	2006	0.6500	2007	0.6500	2008	0.6500	2009	0.6500
2010	0.6500	2011	0.6500	2012	0.6500	2013	0.6500	2014	0.6500
2015	0.6500	2016	0.6500	2017	0.6500	2018	0.6500	2019	0.6500
2020	0.6500	2021	0.6500	2022	0.6500	2023	0.6500	2024	0.6500

Fuel Cost ($/MMBtu):

1995	0.9615	1996	0.9615	1997	0.9615	1998	0.9615	1999	0.9615
2000	0.9615	2001	0.9615	2002	0.9615	2003	0.9615	2004	0.9615
2005	0.9615	2006	0.9615	2007	0.9615	2008	0.9615	2009	0.9615
2010	0.9615	2011	0.9615	2012	0.9615	2013	0.9615	2014	0.9615
2015	0.9615	2016	0.9615	2017	0.9615	2018	0.9615	2019	0.9615
2020	0.9615	2021	0.9615	2022	0.9615	2023	0.9615	2024	0.9615

O&M (mills/kWh):

1995	6.50	1996	6.50	1997	6.50	1998	6.50	1999	6.50
2000	6.50	2001	6.50	2002	6.50	2003	6.50	2004	6.50
2005	6.50	2006	6.50	2007	6.50	2008	6.50	2009	6.50
2010	6.50	2011	6.50	2012	6.50	2013	6.50	2014	6.50
2015	6.50	2016	6.50	2017	6.50	2018	6.50	2019	6.50
2020	6.50	2021	6.50	2022	6.50	2023	6.50	2024	6.50

Working Cap. (million $):

1995	0.00	1996	0.00	1997	0.00	1998	0.00	1999	0.00
2000	0.00	2001	0.00	2002	0.00	2003	0.00	2004	0.00
2005	0.00	2006	0.00	2007	0.00	2008	0.00	2009	0.00
2010	0.00	2011	0.00	2012	0.00	2013	0.00	2014	0.00
2015	0.00	2016	0.00	2017	0.00	2018	0.00	2019	0.00
2020	0.00	2021	0.00	2022	0.00	2023	0.00	2024	0.00

Federal Sub. (million $):

1995	0.00	1996	0.00	1997	0.00	1998	0.00	1999	0.00
2000	0.00	2001	0.00	2002	0.00	2003	0.00	2004	0.00
2005	0.00	2006	0.00	2007	0.00	2008	0.00	2009	0.00
2010	0.00	2011	0.00	2012	0.00	2013	0.00	2014	0.00
2015	0.00	2016	0.00	2017	0.00	2018	0.00	2019	0.00
2020	0.00	2021	0.00	2022	0.00	2023	0.00	2024	0.00

Busbar Costs (cents/kWh):

1995	0.00	1996	0.00	1997	0.00	1998	0.00	1999	0.00
2000	0.00	2001	0.00	2002	0.00	2003	0.00	2004	0.00
2005	0.00	2006	0.00	2007	0.00	2008	0.00	2009	0.00
2010	0.00	2011	0.00	2012	0.00	2013	0.00	2014	0.00
2015	0.00	2016	0.00	2017	0.00	2018	0.00	2019	0.00
2020	0.00	2021	0.00	2022	0.00	2023	0.00	2024	0.00

Construction Expenses (million$/yr):

Plant:

1987	482.00	1988	607.00	1989	669.00	1990	482.00	1991	263.00
1992	76.00	1993	46.00	1994	44.00				

Kidder-Peabody LMFBR Base Case 10/25/83

Quarterly Construction Expenses (millions of current $)

	--Direct Construction--			-----Interest and Equity Charges-----					Total
	ON Cost	Infl.	Total	ST debt	LT debt	Pref.	Common	Total	Cost
1987									
Qrt. 1	120.5	37.2	157.7	0.0	0.0	0.0	0.0	0.0	157.7
Qrt. 2	120.5	39.5	160.0	2.6	0.0	0.0	0.0	2.6	162.6
Qrt. 3	120.5	41.9	162.4	5.3	0.0	0.0	0.0	5.3	167.7
Qrt. 4	120.5	44.3	164.8	8.1	0.0	0.0	0.0	8.1	172.9
1988									
Qrt. 1	151.8	59.0	210.7	10.9	0.0	0.0	0.0	10.9	221.6
Qrt. 2	151.8	62.2	213.9	3.4	0.0	0.0	0.0	3.4	217.3
Qrt. 3	151.8	65.4	217.1	6.8	0.0	0.0	0.0	6.8	223.9
Qrt. 4	151.8	68.7	220.4	10.2	0.0	0.0	0.0	10.2	230.6
1989									
Qrt. 1	167.2	79.6	246.8	13.8	51.4	0.0	0.0	65.2	312.0
Qrt. 2	167.2	83.5	250.7	4.6	0.0	0.0	0.0	4.6	255.3
Qrt. 3	167.2	87.5	254.7	8.3	0.0	0.0	0.0	8.3	263.0
Qrt. 4	167.2	91.5	258.7	12.1	0.0	0.0	0.0	12.1	270.9
1990									
Qrt. 1	120.5	69.1	189.6	16.1	118.7	0.0	0.0	134.8	324.3
Qrt. 2	120.5	72.2	192.7	4.8	0.0	0.0	0.0	4.8	197.5
Qrt. 3	120.5	75.5	196.0	7.7	0.0	0.0	0.0	7.7	203.7
Qrt. 4	120.5	78.8	199.3	10.7	0.0	0.0	0.0	10.7	210.0
1991									
Qrt. 1	65.8	44.8	110.5	13.9	199.8	0.0	0.0	213.7	324.2
Qrt. 2	65.8	46.6	112.4	4.7	0.0	0.0	0.0	4.7	117.1
Qrt. 3	65.8	48.5	114.2	6.5	0.0	0.0	0.0	6.5	120.7
Qrt. 4	65.8	50.4	116.1	8.2	0.0	0.0	0.0	8.2	124.4
1992									
Qrt. 1	19.0	15.1	34.1	10.0	269.1	0.0	0.0	279.1	313.2
Qrt. 2	19.0	15.6	34.6	4.5	0.0	0.0	0.0	4.5	39.1
Qrt. 3	19.0	16.2	35.2	5.0	0.0	0.0	0.0	5.0	40.2
Qrt. 4	19.0	16.8	35.8	5.6	0.0	0.0	0.0	5.6	41.4
1993									
Qrt. 1	11.5	10.5	22.0	6.2	320.0	0.0	0.0	326.2	348.2
Qrt. 2	11.5	10.8	22.3	4.9	0.0	0.0	0.0	4.9	27.2
Qrt. 3	11.5	11.2	22.7	5.3	0.0	0.0	0.0	5.3	28.0
Qrt. 4	11.5	11.6	23.1	5.7	0.0	0.0	0.0	5.7	28.7
1994									
Qrt. 1	11.0	11.4	22.4	6.1	352.0	0.0	0.0	358.1	380.5
Qrt. 2	11.0	11.8	22.8	5.3	0.0	0.0	0.0	5.3	28.0
Qrt. 3	11.0	12.1	23.1	5.7	0.0	0.0	0.0	5.7	28.8
Qrt. 4	11.0	12.5	23.5	6.1	0.0	0.0	0.0	6.1	29.5

Kidder-Peabody LMFBR Base Case 10/25/83

Quarterly Construction Financing (millions of current $)

	--------Source of Funds--------				----Quarter----		------Cumulative------		
	ST debt	Lt debt	Pref.	Common	CWIP	AFUDC	CWIP	AFUDC	Total
1987									
Qrt. 1	118.2	0.0	0.0	39.4	157.7	0.0	157.7	0.0	157.7
Qrt. 2	122.0	0.0	0.0	40.7	160.0	2.6	317.7	2.6	320.3
Qrt. 3	125.8	0.0	0.0	41.9	162.4	5.3	480.1	7.9	488.0
Qrt. 4	129.7	0.0	0.0	43.2	164.8	8.1	644.9	16.0	660.9
1988									
Qrt. 1	-329.4	495.7	0.0	55.4	210.7	10.9	855.6	26.9	882.5
Qrt. 2	163.0	0.0	0.0	54.3	213.9	3.4	1069.5	30.3	1099.8
Qrt. 3	167.9	0.0	0.0	56.0	217.1	6.8	1286.7	37.1	1323.7
Qrt. 4	173.0	0.0	0.0	57.7	220.4	10.2	1507.1	47.3	1554.4
1989									
Qrt. 1	-436.1	670.1	0.0	78.0	246.8	65.2	1753.9	112.5	1866.4
Qrt. 2	191.5	0.0	0.0	63.8	250.7	4.6	2004.6	117.0	2121.7
Qrt. 3	197.2	0.0	0.0	65.7	254.7	8.3	2259.3	125.3	2384.7
Qrt. 4	203.2	0.0	0.0	67.7	258.7	12.1	2518.1	137.5	2655.5
1990									
Qrt. 1	-582.6	825.9	0.0	81.1	189.6	134.8	2707.6	272.2	2979.9
Qrt. 2	148.2	0.0	0.0	49.4	192.7	4.8	2900.4	277.1	3177.4
Qrt. 3	152.8	0.0	0.0	50.9	196.0	7.7	3096.3	284.8	3381.1
Qrt. 4	157.5	0.0	0.0	52.5	199.3	10.7	3295.6	295.5	3591.1
1991									
Qrt. 1	-458.5	701.7	0.0	81.1	110.5	213.7	3406.2	509.2	3915.4
Qrt. 2	87.8	0.0	0.0	29.3	112.4	4.7	3518.5	514.0	4032.5
Qrt. 3	90.5	0.0	0.0	30.2	114.2	6.5	3632.8	520.4	4153.2
Qrt. 4	93.3	0.0	0.0	31.1	116.1	8.2	3748.9	528.7	4277.5
1992									
Qrt. 1	-279.9	514.8	0.0	78.3	34.1	279.1	3783.0	807.8	4590.8
Qrt. 2	29.3	0.0	0.0	9.8	34.6	4.5	3817.6	812.3	4629.9
Qrt. 3	30.2	0.0	0.0	10.1	35.2	5.0	3852.9	817.3	4670.1
Qrt. 4	31.0	0.0	0.0	10.3	35.8	5.6	3888.6	822.9	4711.5
1993									
Qrt. 1	-64.3	325.5	0.0	87.0	22.0	326.2	3910.6	1149.1	5059.7
Qrt. 2	20.4	0.0	0.0	6.8	22.3	4.9	3933.0	1154.0	5086.9
Qrt. 3	21.0	0.0	0.0	7.0	22.7	5.3	3955.7	1159.3	5114.9
Qrt. 4	21.5	0.0	0.0	7.2	23.1	5.7	3978.7	1164.9	5143.6
1994									
Qrt. 1	-38.8	324.1	0.0	95.1	22.4	358.1	4001.1	1523.0	5524.1
Qrt. 2	21.0	0.0	0.0	7.0	22.8	5.3	4023.9	1528.3	5552.1
Qrt. 3	21.6	0.0	0.0	7.2	23.1	5.7	4047.0	1533.9	5580.9
Qrt. 4	22.2	0.0	0.0	7.4	23.5	6.1	4070.5	1540.0	5610.4

Kidder-Peabody LMFBR Base Case 10/25/83

Annual Construction Expenses (millions of current $)

Year	--Direct Construction-- ON Cost	Infl.	Total	-----Interest and Equity Charges----- ST debt	LT debt	Pref.	Common	Total	Total Cost
1987	482.0	162.9	644.9	16.0	0.0	0.0	0.0	16.0	660.9
1988	607.0	255.2	862.2	31.3	0.0	0.0	0.0	31.3	893.5
1989	669.0	342.0	1011.0	38.8	51.4	0.0	0.0	90.2	1101.2
1990	482.0	295.6	777.6	39.4	118.7	0.0	0.0	158.0	935.6
1991	263.0	190.3	453.3	33.3	199.8	0.0	0.0	233.1	686.4
1992	76.0	63.7	139.7	25.1	269.1	0.0	0.0	294.2	433.9
1993	46.0	44.1	90.1	22.0	320.0	0.0	0.0	342.1	432.2
1994	44.0	47.7	91.7	23.1	352.0	0.0	0.0	375.1	466.8
Accru:	0.0	0.0	0.0	6.5	383.6	0.0	0.0	390.1	390.1
Total:	2669.0	1401.5	4070.5	235.4	1694.7	0.0	0.0	1930.1	6000.5

Annual Construction Financing (millions of current $)

Year	--------Source of Funds-------- ST debt	Lt debt	Pref.	Common	----Annual----- CWIP	AFUDC	------Cumulative------ CWIP	AFUDC	Total
1987	495.7	0.0	0.0	165.2	644.9	16.0	644.9	16.0	660.9
1988	174.5	495.7	0.0	223.4	862.2	31.3	1507.1	47.3	1554.4
1989	155.8	670.1	0.0	275.3	1011.0	90.2	2518.1	137.5	2655.5
1990	-124.2	825.9	0.0	233.9	777.6	158.0	3295.6	295.5	3591.1
1991	-186.9	701.7	0.0	171.6	453.3	233.1	3748.9	528.7	4277.5
1992	-189.3	514.8	0.0	108.5	139.7	294.2	3888.6	822.9	4711.5
1993	-1.3	325.5	0.0	108.0	90.1	342.1	3978.7	1164.9	5143.6
1994	26.0	324.1	0.0	116.7	91.7	375.1	4070.5	1540.0	5610.4
Accru:	-350.1	642.7	0.0	97.5	0.0	390.1	4070.5	1930.1	6000.5
Total:	-0.0	4500.4	0.0	1500.1	4070.5	1930.1	4070.5	1930.1	6000.5

Construction-End Summary

Accruals (million $) Inv.Tx.Cr.	Tax Sav.	------Direct------ Current	Base Year	------AFUDC------- Current	Base Year	---Weighted Rate--- Preferred	LT debt
407.0	965.1	4070.5	2669.0	1930.1	1038.8	0.00000	0.09916

Kidder-Peabody LMFBR Base Case 10/25/83

CONSTRUCTION SUMMARY
(All Amounts in millions of current year $)

Period: (1987,1995)
Base Year: 1982

Expenses:	Total	Book Life	Book Dep.
Direct :	4070.5	30	135.7
Startup:	0.0	30	0.0
AFUDC :	1930.1	30	64.3
Wk.Cap.:	0.0	--	--
Total :	6000.5	--	200.0

Financing:	Total	% Financing	Charge Rate
Common :	1500.1	25.00	14.15
Preferred:	0.0	0.00	0.00
LT.Debt :	4500.4	75.00	9.92
Total :	6000.5	100.00	10.97

Accruals:
ITC: 407.0
FIT: 965.1

OPERATING SUMMARY
Acceleration Tax Depreciation:
Tax Life : 25 (years)
Method : Straight-line
Basis : 3866.9
TEFRA Opt.: 95 %

Option Flags
IDC Compounding : Simple
Financing Period : Quarter
Property Tax Basis : Net
Equity Finan. Basis: Fixed %
Discount Rate : Pre-tax
Analysis Type : Bottom-up
Accounting Basis : Flow-thru
Reg. Tax Treatment : Unconsolidated

Analysis Summary:	ROI (%)	Lev. Busbar (cents/kWh)	Discount Rate (%)	Fixed Charge Rate (%)
Current:	22.18	18.18	10.97	14.96
Base Yr:	14.77	5.22	4.50	6.95

Kidder-Peabody LMFBR Base Case 10/25/83

BALANCE SHEET
(Millions of Current Year $)

	1995	2002	2010	2018	2024
ASSETS					
Cum. Working Capital	0.0	0.0	0.0	0.0	0.0
Gross Plant	4070.5	4070.5	4070.5	4070.5	4070.5
Less Accum.Plant Dep.	135.7	1085.5	2170.9	3256.4	4070.5
Net Plant	3934.8	2985.0	1899.5	814.1	-0.0
Capitalized AFUDC	1930.1	1930.1	1930.1	1930.1	1930.1
Less Accum.AFUDC Dep.	64.3	514.7	1029.4	1544.1	1930.1
Net Capitalized AFUDC	1865.7	1415.4	900.7	386.0	0.0
TOTAL ASSETS	5800.5	4400.4	2800.3	1200.1	-0.0
LIABILITIES					
LTD Due in Next Year	150.0	150.0	150.0	150.0	0.0
LTD Outstanding	4200.4	3150.3	1950.2	750.1	0.0
Cum. Deferred ITC	214.6	298.5	190.0	81.4	0.0
Cum. Deferred FIT	0.0	0.0	0.0	0.0	0.0
P.S. Due in Next Year	0.0	0.0	0.0	0.0	0.0
Pref.Stock Outstanding	0.0	0.0	0.0	0.0	0.0
End Common Equity	1235.5	801.6	510.1	218.6	-0.0
Partner Loans Outst.	0.0	0.0	0.0	0.0	0.0
Total Stock Equity	1235.5	801.6	510.1	218.6	-0.0
TOTAL LIABIL. & EQUITY	5800.5	4400.4	2800.3	1200.1	-0.0

Kidder-Peabody LMFBR Base Case 10/25/83

INCOME STATEMENT
(Millions of Current Year $)

	1995	2002	2010	2018	2024
REVENUES					
Electric Sales	1435.5	1251.2	1260.9	1420.9	1693.5
Federal Subsidy	0.0	0.0	0.0	0.0	0.0
TOTAL REVENUES	1435.5	1251.2	1260.9	1420.9	1693.5
EXPENSES					
Fuel Expense	156.5	238.5	385.8	624.3	895.7
O&M Expense	101.7	155.0	250.8	405.8	582.2
Backfit&Replacement	0.0	0.0	0.0	0.0	0.0
Ad Valorem Tax	0.0	0.0	0.0	0.0	0.0
Total Oper. Expense	258.2	393.5	636.6	1030.1	1477.9
Plant Book Dep.	135.7	135.7	135.7	135.7	135.7
AFUDC Book Dep.	64.3	64.3	64.3	64.3	64.3
Total Book Dep.	200.0	200.0	200.0	200.0	200.0
Property Tax	120.0	92.0	60.0	28.0	4.0
Net FIT (Book Basis)	198.7	105.0	63.8	22.5	-8.4
Total Pre-Debt Expen.	777.0	790.5	960.4	1280.7	1673.5
Pre-Debt Net Income	658.5	460.7	300.5	140.2	20.0
LTD Interest	446.3	342.1	223.1	104.1	14.9
NET INCOME	212.3	118.6	77.3	36.1	5.2
Less Pref. Dividends	0.0	0.0	0.0	0.0	0.0
Common Stock Earnings	212.3	118.6	77.3	36.1	5.2
	2	17.6	17.7	20.0	23.8

Kidder-Peabody LMFBR Base Case 10/25/83

SOURCES AND USES OF FUNDS STATEMENT
(Millions of Current Year $)

	1995	2002	2010	2018	2024
		SOURCES OF FUNDS			
Net Income	212.3	118.6	77.3	36.1	5.2
Plant Book Dep.	135.7	135.7	135.7	135.7	135.7
AFUDC Book Dep.	64.3	64.3	64.3	64.3	64.3
Total Book Dep.	200.0	200.0	200.0	200.0	200.0
Deferred ITC	214.6	-13.6	-13.6	-13.6	-13.6
Deferred FIT	0.0	0.0	0.0	0.0	0.0
TOTAL INTERNAL	626.9	305.0	263.8	222.5	191.6
Partner Loans	0.0	0.0	0.0	0.0	0.0
Government Loans	0.0	0.0	0.0	0.0	0.0
TOTAL SOURCES	626.9	305.0	263.8	222.5	191.6
		USES OF FUNDS			
Preferred Stock Due	0.0	0.0	0.0	0.0	0.0
Preferred Dividends	0.0	0.0	0.0	0.0	0.0
Common Dividends	476.9	155.0	113.8	72.5	41.6
Working Capital Change	0.0	0.0	0.0	0.0	0.0
LTD Repayment	150.0	150.0	150.0	150.0	150.0
Partner Loan Repayment	0.0	0.0	0.0	0.0	0.0
TOTAL USES	626.9	305.0	263.8	222.5	191.6

Kidder-Peabody LMFBR Base Case 10/25/83

RETAINED EARNINGS STATEMENT & TAX DEP. WORKSHEET
(Millions of Current Year $)

	1995	2002	2010	2018	2024
		RETAINED EARNINGS			
Begin.Common Equity	1500.1	838.0	546.5	255.1	36.4
Common Stock Earnings	212.3	118.6	77.3	36.1	5.2
Total Common Equity	1712.4	956.6	623.9	291.1	41.6
Less Common Dividends	476.9	155.0	113.8	72.5	41.6
END COMMON EQUITY	1235.5	801.6	510.1	218.6	-0.0
RETURN ON EQUITY (%)	14.1	14.1	14.1	14.1	14.1
		TAX DEP. WORKSHEET			
Startup Dep.	0.0	0.0	0.0	0.0	0.0
5-year Prop.Dep.	0.0	0.0	0.0	0.0	0.0
10-year Prop.Dep.	0.0	0.0	0.0	0.0	0.0
15-year Prop.Dep.	0.0	0.0	0.0	0.0	0.0
Straight-line Dep.	154.7	154.7	154.7	154.7	0.0
TOTAL TAX DEPRECIATION	154.7	154.7	154.7	154.7	0.0

Kidder-Peabody LMFBR Base Case 10/25/83

TAX CALCULATION (Book/Regulation Basis)
(Millions of Current Year $)

	1995	2002	2010	2018	2024
BOOK TAX CALCULATION					
Electric Sales	1435.5	1251.2	1260.9	1420.9	1693.5
Total Oper. Expense	258.2	393.5	636.6	1030.1	1477.9
Total Book Dep.	200.0	200.0	200.0	200.0	200.0
Property Tax	120.0	92.0	60.0	28.0	4.0
LTD Interest	446.3	342.1	223.1	104.1	14.9
Pre-tax Book Income	424.5	237.2	154.7	72.2	10.3
Book FIT	212.3	118.6	77.3	36.1	5.2
Less amort.ITC	13.6	13.6	13.6	13.6	13.6
NET FIT (Book Basis)	198.7	105.0	63.8	22.5	-8.4
REGULATORY TAX CALCULATION					
Electric Sales	1435.5	1251.2	1260.9	1420.9	1693.5
Total Oper. Expense	258.2	393.5	636.6	1030.1	1477.9
Total Tax Depreciation	154.7	154.7	154.7	154.7	0.0
Property Tax	120.0	92.0	60.0	28.0	4.0
LTD Interest	446.3	342.1	223.1	104.1	14.9
Pre-tax Part. Income	456.3	268.9	186.4	104.0	196.8
FIT Due	228.2	134.5	93.2	52.0	98.4
Less ITC Taken	228.2	0.0	0.0	0.0	0.0
Less FIT Taken	0.0	0.0	0.0	0.0	0.0
NET REGULATED FIT	0.0	134.5	93.2	52.0	98.4

Kidder-Peabody LMFBR Base Case 10/25/83

PARTNERSHIP RETURN AND CASHFLOW ANALYSIS
(Millions of Current Year $)

	1995	2002	2010	2018	2024
PARTNERSHIP RETURN					
Electric Sales	1435.5	1251.2	1260.9	1420.9	1693.5
Total Oper. Expense	258.2	393.5	636.6	1030.1	1477.9
Total Tax Depreciation	154.7	154.7	154.7	154.7	0.0
Property Tax	120.0	92.0	60.0	28.0	4.0
LTD Interest	446.3	342.1	223.1	104.1	14.9
Pre-tax Part. Income	456.3	268.9	186.4	104.0	196.8
FIT Due	228.2	134.5	93.2	52.0	98.4
PARTNERSHIP CASHFLOW ANALYSIS					
After-tax Cash	228.2	134.5	93.2	52.0	98.4
Total Tax Depreciation	154.7	154.7	154.7	154.7	0.0
Federal Subsidy	0.0	0.0	0.0	0.0	0.0
Total Cash Avail.	382.8	289.1	247.9	206.7	98.4
Less LTD Repayment	150.0	150.0	150.0	150.0	150.0
Less Pref. Dividends	0.0	0.0	0.0	0.0	0.0
Less Pref. Stock Due	0.0	0.0	0.0	0.0	0.0
Less Partner Loans	0.0	0.0	0.0	0.0	0.0
Less Work.Cap. Change	0.0	0.0	0.0	0.0	0.0
NET TO PARTNERS	232.8	139.1	97.9	56.6	-51.6
	8	1212.3	2139.7	2737.2	2582.1

Kidder-Peabody LMFBR Base Case 10/25/83

AFTER-TAX CASH FLOWS TO UTILITY INVESTORS
(Millions of Current Year $)

Year	During Construction Cash Inv.	During Construction Tax Ben.	During Operation Tax Due	During Operation Cash Dist.	Net A.T. Cash Flow	Cum.A.T. Cash Flow	Cm.Dis.R. A.T.C.F.
1987	165.218	77.941	--	--	-87.276	-87.276	-48.824
1988	223.374	129.008	--	--	-94.366	-181.642	-96.439
1989	275.294	180.983	--	--	-94.311	-275.952	-139.320
1990	233.902	196.252	--	--	-37.651	-313.603	-154.702
1991	171.600	194.611	--	--	23.011	-290.592	-146.286
1992	108.487	184.617	--	--	76.129	-214.463	-121.337
1993	108.038	195.966	--	--	87.928	-126.534	-95.468
1994	214.222	212.728	--	--	-1.494	-128.028	-95.863
1995	--	--	228.164	460.983	232.819	104.791	-40.505
1996	--	--	190.722	386.099	195.378	300.169	1.356
1997	--	--	160.253	325.163	164.910	465.079	33.195
1998	--	--	155.097	314.852	159.755	624.833	60.988
1999	--	--	149.941	304.540	154.599	779.432	85.224
2000	--	--	144.785	294.228	149.443	928.875	106.336
2001	--	--	139.629	283.917	144.288	1073.163	124.703
2002	--	--	134.473	273.605	139.132	1212.295	140.662
2003	--	--	129.317	263.293	133.976	1346.271	154.511
2004	--	--	124.161	252.982	128.820	1475.091	166.509
2005	--	--	119.005	242.670	123.664	1598.755	176.888
2006	--	--	113.849	232.358	118.509	1717.264	185.851
2007	--	--	108.693	222.046	113.353	1830.617	193.576
2008	--	--	103.538	211.735	108.197	1938.814	200.221
2009	--	--	98.382	201.423	103.042	2041.856	205.923
2010	--	--	93.226	191.111	97.886	2139.741	210.804
2011	--	--	88.070	180.800	92.730	2232.471	214.971
2012	--	--	82.914	170.488	87.574	2320.046	218.517
2013	--	--	77.758	160.176	82.419	2402.464	221.524
2014	--	--	72.602	149.865	77.263	2479.727	224.064
2015	--	--	67.446	139.553	72.107	2551.834	226.200
2016	--	--	62.290	129.241	66.951	2618.785	227.988
2017	--	--	57.134	118.929	61.795	2680.580	229.474
2018	--	--	51.978	108.618	56.640	2737.220	230.702
2019	--	--	46.822	98.306	51.484	2788.704	231.708
2020	--	--	119.006	87.994	-31.012	2757.692	231.162
2021	--	--	113.850	77.683	-36.168	2721.524	230.588
2022	--	--	108.694	67.371	-41.323	2680.201	229.998
2023	--	--	103.538	57.059	-46.479	2633.722	229.399
2024	--	--	98.383	46.748	-51.635	2582.087	228.800
TOT.	1500.134	1372.106	3343.720	6053.830	2582.087		

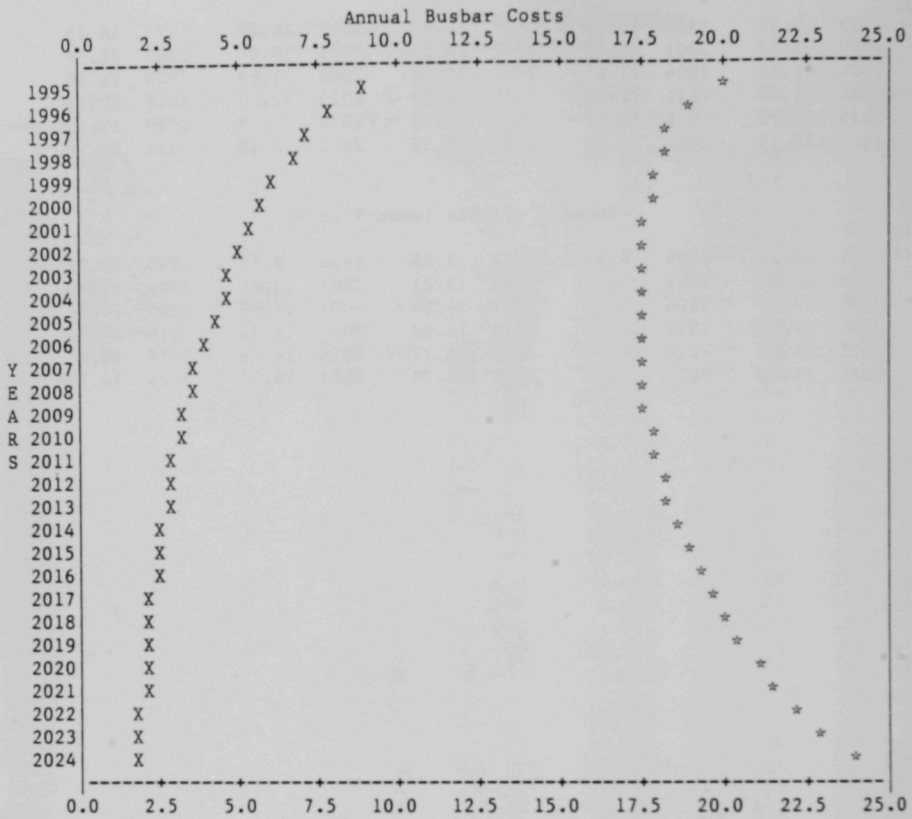
Summary of ROIs (Current $)

1995	5.35	1996	11.50	1997	14.69	1998	16.80	1999	18.25
2000	19.27	2001	20.01	2002	20.55	2003	20.96	2004	21.26
2005	21.48	2006	21.65	2007	21.79	2008	21.89	2009	21.96
2010	22.02	2011	22.06	2012	22.10	2013	22.13	2014	22.15
2015	22.16	2016	22.17	2017	22.18	2018	22.19	2019	22.19
2020	22.19	2021	22.19	2022	22.18	2023	22.18	2024	22.18

Summary of ROIs (Base-Year $)

1995	-1.15	1996	4.65	1997	7.66	1998	9.65	1999	11.03
2000	12.00	2001	12.70	2002	13.21	2003	13.60	2004	13.88
2005	14.10	2006	14.26	2007	14.39	2008	14.49	2009	14.56
2010	14.62	2011	14.66	2012	14.69	2013	14.72	2014	14.74
2015	14.75	2016	14.76	2017	14.77	2018	14.78	2019	14.78
2020	14.78	2021	14.78	2022	14.78	2023	14.77	2024	14.77

Kidder-Peabody LMFBR Base Case 10/25/83



Summary of Levelized Busbar Costs
(Levelized over 20 years)
18.18 (current cents/kWh,discount rate : 10.975 (%))
5.22 (base yr. cents/kWh,discount rate: 4.496 (%))

Summary of Busbar Costs (current cents/kWh)

1995	20.17	1996	19.08	1997	18.19	1998	18.04	1999	17.90
2000	17.77	2001	17.67	2002	17.58	2003	17.51	2004	17.47
2005	17.44	2006	17.44	2007	17.47	2008	17.52	2009	17.60
2010	17.72	2011	17.86	2012	18.04	2013	18.25	2014	18.51
2015	18.80	2016	19.14	2017	19.53	2018	19.96	2019	20.45
2020	20.99	2021	21.60	2022	22.26	2023	22.99	2024	23.79

Summary of Busbar Costs (base yr. cents/kWh)

1995	8.90	1996	7.93	1997	7.12	1998	6.65	1999	6.21
2000	5.81	2001	5.43	2002	5.09	2003	4.78	2004	4.49
2005	4.22	2006	3.97	2007	3.75	2008	3.54	2009	3.35
2010	3.17	2011	3.01	2012	2.86	2013	2.73	2014	2.60
2015	2.49	2016	2.39	2017	2.29	2018	2.21	2019	2.13
2020	2.06	2021	1.99	2022	1.94	2023	1.88	2024	1.83

D.3 RUN ILLUSTRATING USE OF THE QUARTER(0)/ANNUAL(1) FLAG

This run illustrates the use of the QUARTER(0)/ANNUAL(1) FLAG. Only the construction analysis, summary, and after-tax cashflow reports have been printed. All inputs are as in Sec. D.2 except that the QUARTER(0)/ANNUAL(1) FLAG has been set to 1. This results in slightly higher AFUDC charge, because all construction expenditures are financed from short-term debt, instead of only 75% as in the previous case.

Kidder-Peabody LMFBR (Quarter(0)/Annual(1) Flag = 1) 10/25/83

Quarterly Construction Expenses (millions of current $)

	--Direct Construction--			-----Interest and Equity Charges-----					Total
	ON Cost	Infl.	Total	ST debt	LT debt	Pref.	Common	Total	Cost
1987									
Qrt. 1	120.5	37.2	157.7	0.0	0.0	0.0	0.0	0.0	157.7
Qrt. 2	120.5	39.5	160.0	3.5	0.0	0.0	0.0	3.5	163.5
Qrt. 3	120.5	41.9	162.4	7.1	0.0	0.0	0.0	7.1	169.5
Qrt. 4	120.5	44.3	164.8	10.8	0.0	0.0	0.0	10.8	175.6
1988									
Qrt. 1	151.8	59.0	210.7	14.7	0.0	0.0	0.0	14.7	225.4
Qrt. 2	151.8	62.2	213.9	4.6	0.0	0.0	0.0	4.6	218.5
Qrt. 3	151.8	65.4	217.1	9.1	0.0	0.0	0.0	9.1	226.3
Qrt. 4	151.8	68.7	220.4	13.8	0.0	0.0	0.0	13.8	234.2
1989									
Qrt. 1	167.2	79.6	246.8	18.6	51.9	0.0	0.0	70.4	317.2
Qrt. 2	167.2	83.5	250.7	6.2	0.0	0.0	0.0	6.2	256.9
Qrt. 3	167.2	87.5	254.7	11.2	0.0	0.0	0.0	11.2	265.9
Qrt. 4	167.2	91.5	258.7	16.4	0.0	0.0	0.0	16.4	275.1
1990									
Qrt. 1	120.5	69.1	189.6	21.7	119.9	0.0	0.0	141.6	331.2
Qrt. 2	120.5	72.2	192.7	6.5	0.0	0.0	0.0	6.5	199.3
Qrt. 3	120.5	75.5	196.0	10.5	0.0	0.0	0.0	10.5	206.5
Qrt. 4	120.5	78.8	199.3	14.6	0.0	0.0	0.0	14.6	213.8
1991									
Qrt. 1	65.8	44.8	110.5	18.8	202.1	0.0	0.0	220.9	331.4
Qrt. 2	65.8	46.6	112.4	6.5	0.0	0.0	0.0	6.5	118.8
Qrt. 3	65.8	48.5	114.2	8.8	0.0	0.0	0.0	8.8	123.0
Qrt. 4	65.8	50.4	116.1	11.2	0.0	0.0	0.0	11.2	127.3
1992									
Qrt. 1	19.0	15.1	34.1	13.7	272.5	0.0	0.0	286.2	320.3
Qrt. 2	19.0	15.6	34.6	6.1	0.0	0.0	0.0	6.1	40.7
Qrt. 3	19.0	16.2	35.2	6.9	0.0	0.0	0.0	6.9	42.1
Qrt. 4	19.0	16.8	35.8	7.7	0.0	0.0	0.0	7.7	43.4
1993									
Qrt. 1	11.5	10.5	22.0	8.5	324.5	0.0	0.0	332.9	354.9
Qrt. 2	11.5	10.8	22.3	6.7	0.0	0.0	0.0	6.7	29.0
Qrt. 3	11.5	11.2	22.7	7.2	0.0	0.0	0.0	7.2	29.9
Qrt. 4	11.5	11.6	23.1	7.8	0.0	0.0	0.0	7.8	30.8
1994									
Qrt. 1	11.0	11.4	22.4	8.3	357.3	0.0	0.0	365.7	388.1
Qrt. 2	11.0	11.8	22.8	7.2	0.0	0.0	0.0	7.2	29.9
Qrt. 3	11.0	12.1	23.1	7.7	0.0	0.0	0.0	7.7	30.8
Qrt. 4	11.0	12.5	23.5	8.3	0.0	0.0	0.0	8.3	31.8

Kidder-Peabody LMFBR (Quarter(0)/Annual(1) Flag = 1) 10/25/83

Quarterly Construction Financing (millions of current $)

	--------Source of Funds--------				----Quarter----		------Cumulative------		
	ST debt	Lt debt	Pref.	Common	CWIP	AFUDC	CWIP	AFUDC	Total
1987									
Qrt. 1	157.7	0.0	0.0	0.0	157.7	0.0	157.7	0.0	157.7
Qrt. 2	163.5	0.0	0.0	0.0	160.0	3.5	317.7	3.5	321.2
Qrt. 3	169.5	0.0	0.0	0.0	162.4	7.1	480.1	10.6	490.6
Qrt. 4	175.6	0.0	0.0	0.0	164.8	10.8	644.9	21.4	666.3
1988									
Qrt. 1	-440.9	499.7	0.0	166.6	210.7	14.7	855.6	36.1	891.7
Qrt. 2	218.5	0.0	0.0	0.0	213.9	4.6	1069.5	40.7	1110.2
Qrt. 3	226.3	0.0	0.0	0.0	217.1	9.1	1286.7	49.8	1336.5
Qrt. 4	234.2	0.0	0.0	0.0	220.4	13.8	1507.1	63.6	1570.7
1989									
Qrt. 1	-587.1	678.3	0.0	226.1	246.8	70.4	1753.9	134.0	1887.9
Qrt. 2	256.9	0.0	0.0	0.0	250.7	6.2	2004.6	140.2	2144.8
Qrt. 3	265.9	0.0	0.0	0.0	254.7	11.2	2259.3	151.4	2410.7
Qrt. 4	275.1	0.0	0.0	0.0	258.7	16.4	2518.1	167.8	2685.8
1990									
Qrt. 1	-784.0	836.4	0.0	278.8	189.6	141.6	2707.6	309.4	3017.0
Qrt. 2	199.3	0.0	0.0	0.0	192.7	6.5	2900.4	315.9	3216.3
Qrt. 3	206.5	0.0	0.0	0.0	196.0	10.5	3096.3	326.4	3422.8
Qrt. 4	213.8	0.0	0.0	0.0	199.3	14.6	3295.6	341.0	3636.6
1991									
Qrt. 1	-619.3	713.1	0.0	237.7	110.5	220.9	3406.2	561.9	3968.0
Qrt. 2	118.8	0.0	0.0	0.0	112.4	6.5	3518.5	568.3	4086.9
Qrt. 3	123.0	0.0	0.0	0.0	114.2	8.8	3632.8	577.1	4209.9
Qrt. 4	127.3	0.0	0.0	0.0	116.1	11.2	3748.9	588.3	4337.2
1992									
Qrt. 1	-380.3	525.4	0.0	175.1	34.1	286.2	3783.0	874.5	4657.4
Qrt. 2	40.7	0.0	0.0	0.0	34.6	6.1	3817.6	880.5	4698.2
Qrt. 3	42.1	0.0	0.0	0.0	35.2	6.9	3852.9	887.4	4740.2
Qrt. 4	43.4	0.0	0.0	0.0	35.8	7.7	3888.6	895.1	4783.7
1993									
Qrt. 1	-91.5	334.9	0.0	111.6	22.0	332.9	3910.6	1228.0	5138.6
Qrt. 2	29.0	0.0	0.0	0.0	22.3	6.7	3933.0	1234.7	5167.6
Qrt. 3	29.9	0.0	0.0	0.0	22.7	7.2	3955.7	1241.9	5197.5
Qrt. 4	30.8	0.0	0.0	0.0	23.1	7.8	3978.7	1249.6	5228.3
1994									
Qrt. 1	-56.6	333.5	0.0	111.2	22.4	365.7	4001.1	1615.3	5616.4
Qrt. 2	29.9	0.0	0.0	0.0	22.8	7.2	4023.9	1622.5	5646.4
Qrt. 3	30.8	0.0	0.0	0.0	23.1	7.7	4047.0	1630.2	5677.2
Qrt. 4	31.8	0.0	0.0	0.0	23.5	8.3	4070.5	1638.5	5709.0

Kidder-Peabody LMFBR (Quarter(0)/Annual(1) Flag = 1) 10/25/83

Annual Construction Expenses (millions of current $)

	--Direct Construction--			-----Interest and Equity Charges-----					Total
Year	ON Cost	Infl.	Total	ST debt	LT debt	Pref.	Common	Total	Cost
1987	482.0	162.9	644.9	21.4	0.0	0.0	0.0	21.4	666.3
1988	607.0	255.2	862.2	42.2	0.0	0.0	0.0	42.2	904.4
1989	669.0	342.0	1011.0	52.3	51.9	0.0	0.0	104.2	1115.2
1990	482.0	295.6	777.6	53.3	119.9	0.0	0.0	173.2	950.8
1991	263.0	190.3	453.3	45.2	202.1	0.0	0.0	247.3	700.6
1992	76.0	63.7	139.7	34.3	272.5	0.0	0.0	306.8	446.5
1993	46.0	44.1	90.1	30.1	324.5	0.0	0.0	354.6	444.7
1994	44.0	47.7	91.7	31.6	357.3	0.0	0.0	388.9	480.6
Accru:	0.0	0.0	0.0	8.9	389.9	0.0	0.0	398.8	398.8
Total:	2669.0	1401.5	4070.5	319.2	1718.1	0.0	0.0	2037.3	6107.8

Annual Construction Financing (millions of current $)

	--------Source of Funds--------				----Annual-----		------Cumulative------		
Year	ST debt	Lt debt	Pref.	Common	CWIP	AFUDC	CWIP	AFUDC	Total
1987	666.3	0.0	0.0	0.0	644.9	21.4	644.9	21.4	666.3
1988	238.1	499.7	0.0	166.6	862.2	42.2	1507.1	63.6	1570.7
1989	210.8	678.3	0.0	226.1	1011.0	104.2	2518.1	167.8	2685.8
1990	-164.4	836.4	0.0	278.8	777.6	173.2	3295.6	341.0	3636.6
1991	-250.2	713.1	0.0	237.7	453.3	247.3	3748.9	588.3	4337.2
1992	-254.1	525.4	0.0	175.1	139.7	306.8	3888.6	895.1	4783.7
1993	-1.8	334.9	0.0	111.6	90.1	354.6	3978.7	1249.6	5228.3
1994	36.0	333.5	0.0	111.2	91.7	388.9	4070.5	1638.5	5709.0
Accru:	-480.6	659.6	0.0	219.9	0.0	398.8	4070.5	2037.3	6107.8
Total:	-0.0	4580.8	0.0	1526.9	4070.5	2037.3	4070.5	2037.3	6107.8

Construction-End Summary

Accruals (million $)		------Direct------		------AFUDC-------		---Weighted Rate---	
Inv.Tx.Cr.	Tax Sav.	Current	Base Year	Current	Base Year	Preferred	LT debt
407.0	1018.7	4070.5	2669.0	2037.3	1101.5	0.00000	0.09915

Kidder-Peabody LMFBR (Quarter(0)/Annual(1) Flag = 1) 10/25/83

CONSTRUCTION SUMMARY
(All Amounts in millions of current year $)

Period: (1987,1995)
Base Year: 1982

Expenses:	Total	Book Life	Book Dep.
Direct :	4070.5	30	135.7
Startup:	0.0	30	0.0
AFUDC :	2037.3	30	67.9
Wk.Cap.:	0.0	--	--
Total :	6107.8	--	203.6

Financing:	Total	% Financing	Charge Rate
Common :	1526.9	25.00	14.15
Preferred:	0.0	0.00	0.00
LT.Debt :	4580.8	75.00	9.92
Total :	6107.8	100.00	10.97

Accruals:
ITC: 407.0
FIT: 1018.7

OPERATING SUMMARY
Acceleration Tax Depreciation:
Tax Life : 25 (years)
Method : Straight-line
Basis : 3866.9
TEFRA Opt.: 95 %
Option Flags
IDC Compounding : Simple
Financing Period : Annual
Property Tax Basis : Net
Equity Finan. Basis: Fixed %
Discount Rate : Pre-tax
Analysis Type : Bottom-up
Accounting Basis : Flow-thru
Reg. Tax Treatment : Unconsolidated

Analysis Summary:	ROI (%)	Lev. Busbar (cents/kWh)	Discount Rate (%)	Fixed Charge Rate (%)
Current:	23.21	18.43	10.97	14.98
Base Yr:	15.74	5.29	4.50	6.97

Kidder-Peabody LMFBR (Quarter(0)/Annual(1) Flag = 1) 10/25/83

AFTER-TAX CASH FLOWS TO UTILITY INVESTORS
(Millions of Current Year $)

Year	During Construction Cash Inv.	Tax Ben.	During Operation Tax Due	Cash Dist.	Net A.T. Cash Flow	Cum.A.T. Cash Flow	Cm.Dis.R. A.T.C.F.
1987	166.568	82.524	--	--	-84.044	-84.044	-47.018
1988	226.094	135.183	--	--	-90.912	-174.956	-92.892
1989	278.791	188.803	--	--	-89.987	-264.943	-133.809
1990	237.695	203.992	--	--	-33.703	-298.646	-147.580
1991	175.149	201.622	--	--	26.473	-272.173	-137.897
1992	111.622	190.751	--	--	79.129	-193.044	-111.962
1993	111.165	202.664	--	--	91.498	-101.546	-85.041
1994	219.856	220.180	--	--	0.325	-101.221	-84.955
1995	--	--	233.745	469.463	235.718	134.498	-28.904
1996	--	--	195.386	392.747	197.361	331.859	13.386
1997	--	--	165.580	333.137	167.557	499.416	45.739
1998	--	--	160.298	322.573	162.275	661.690	73.974
1999	--	--	155.016	312.008	156.993	818.683	98.589
2000	--	--	149.733	301.444	151.711	970.393	120.023
2001	--	--	144.451	290.879	146.428	1116.822	138.665
2002	--	--	139.168	280.315	141.146	1257.968	154.857
2003	--	--	133.886	269.750	135.864	1393.832	168.903
2004	--	--	128.604	259.186	130.582	1524.414	181.067
2005	--	--	123.321	248.621	125.300	1649.713	191.585
2006	--	--	118.039	238.056	120.017	1769.731	200.663
2007	--	--	112.757	227.492	114.735	1884.466	208.484
2008	--	--	107.474	216.927	109.453	1993.919	215.206
2009	--	--	102.192	206.363	104.171	2098.090	220.972
2010	--	--	96.909	195.798	98.889	2196.979	225.904
2011	--	--	91.627	185.234	93.607	2290.585	230.111
2012	--	--	86.345	174.669	88.324	2378.910	233.688
2013	--	--	81.062	164.104	83.042	2461.952	236.718
2014	--	--	75.780	153.540	77.760	2539.712	239.276
2015	--	--	70.497	142.975	72.478	2612.190	241.423
2016	--	--	65.215	132.411	67.196	2679.385	243.218
2017	--	--	59.933	121.846	61.914	2741.299	244.707
2018	--	--	54.650	111.282	56.631	2797.930	245.935
2019	--	--	49.368	100.717	51.349	2849.280	246.938
2020	--	--	121.426	90.153	-31.273	2818.006	246.388
2021	--	--	116.143	79.588	-36.555	2781.451	245.808
2022	--	--	110.861	69.023	-41.838	2739.613	245.210
2023	--	--	105.579	58.459	-47.120	2692.494	244.603
2024	--	--	100.296	47.894	-52.402	2640.092	243.994
TOT.	1526.940	1425.719	3455.341	6196.650	2640.092		

Summary of ROIs (Current $)

1995	6.99	1996	12.95	1997	16.06	1998	18.11	1999	19.51
2000	20.49	2001	21.19	2002	21.71	2003	22.09	2004	22.37
2005	22.58	2006	22.74	2007	22.86	2008	22.95	2009	23.02
2010	23.07	2011	23.11	2012	23.14	2013	23.17	2014	23.18
2015	23.20	2016	23.21	2017	23.21	2018	23.22	2019	23.22
2020	23.22	2021	23.22	2022	23.22	2023	23.22	2024	23.21

Summary of ROIs (Base-Year $)

1995	0.39	1996	6.01	1997	8.94	1998	10.88	1999	12.20
2000	13.13	2001	13.81	2002	14.29	2003	14.66	2004	14.93
2005	15.13	2006	15.28	2007	15.40	2008	15.48	2009	15.55
2010	15.60	2011	15.64	2012	15.67	2013	15.69	2014	15.71
2015	15.72	2016	15.73	2017	15.74	2018	15.74	2019	15.75
2020	15.75	2021	15.74	2022	15.74	2023	15.74	2024	15.74

D.4 RUN ILLUSTRATING USE OF THE NET(0)/FIX(1) EQUITY FLAG

This run illustrates the use of the NET(0)/FIX(1) EQUITY FLAG and contains the construction summary, summary, and after-tax cashflow reports. All inputs are as in Sec. D.2 except that the NET(0)/FIX(1) EQUITY FLAG has been set to zero. This case has a higher AFUDC charge than for either of the cases in Secs. D.2 and D.3. Although the final split between debt and equity financing is almost the same as for case D.2, the timing of debt issues is different. In this example, the long-term debt is higher during the early years, resulting in higher AFUDC charges due to compounding. Since the net equity investment is constrained to be zero, the ROI is not calculable and is set to -1000.

Kidder-Peabody LMFBR (Net(0)/Fix(1) Equity Flag = 0) 10/25/83

Quarterly Construction Expenses (millions of current $)

	--Direct Construction--			-----Interest and Equity Charges-----					Total
	ON Cost	Infl.	Total	ST debt	LT debt	Pref.	Common	Total	Cost
1987									
Qrt. 1	120.5	37.2	157.7	0.0	0.0	0.0	0.0	0.0	157.7
Qrt. 2	120.5	39.5	160.0	3.1	0.0	0.0	0.0	3.1	163.1
Qrt. 3	120.5	41.9	162.4	6.3	0.0	0.0	0.0	6.3	168.7
Qrt. 4	120.5	44.3	164.8	9.5	0.0	0.0	0.0	9.5	174.3
1988									
Qrt. 1	151.8	59.0	210.7	12.9	0.0	0.0	0.0	12.9	223.6
Qrt. 2	151.8	62.2	213.9	4.1	0.0	0.0	0.0	4.1	218.0
Qrt. 3	151.8	65.4	217.1	8.1	0.0	0.0	0.0	8.1	225.2
Qrt. 4	151.8	68.7	220.4	12.1	0.0	0.0	0.0	12.1	232.6
1989									
Qrt. 1	167.2	79.6	246.8	15.7	60.6	0.0	0.0	76.2	323.0
Qrt. 2	167.2	83.5	250.7	5.8	0.0	0.0	0.0	5.8	256.5
Qrt. 3	167.2	87.5	254.7	10.2	0.0	0.0	0.0	10.2	264.9
Qrt. 4	167.2	91.5	258.7	14.7	0.0	0.0	0.0	14.7	273.4
1990									
Qrt. 1	120.5	69.1	189.6	18.0	137.1	0.0	0.0	155.1	344.6
Qrt. 2	120.5	72.2	192.7	6.4	0.0	0.0	0.0	6.4	199.1
Qrt. 3	120.5	75.5	196.0	9.8	0.0	0.0	0.0	9.8	205.8
Qrt. 4	120.5	78.8	199.3	13.4	0.0	0.0	0.0	13.4	212.7
1991									
Qrt. 1	65.8	44.8	110.5	14.8	227.9	0.0	0.0	242.7	353.2
Qrt. 2	65.8	46.6	112.4	6.6	0.0	0.0	0.0	6.6	119.0
Qrt. 3	65.8	48.5	114.2	8.6	0.0	0.0	0.0	8.6	122.9
Qrt. 4	65.8	50.4	116.1	10.7	0.0	0.0	0.0	10.7	126.8
1992									
Qrt. 1	19.0	15.1	34.1	9.9	301.8	0.0	0.0	311.7	345.8
Qrt. 2	19.0	15.6	34.6	6.4	0.0	0.0	0.0	6.4	41.1
Qrt. 3	19.0	16.2	35.2	7.1	0.0	0.0	0.0	7.1	42.3
Qrt. 4	19.0	16.8	35.8	7.8	0.0	0.0	0.0	7.8	43.5
1993									
Qrt. 1	11.5	10.5	22.0	5.1	352.0	0.0	0.0	357.1	379.1
Qrt. 2	11.5	10.8	22.3	7.0	0.0	0.0	0.0	7.0	29.3
Qrt. 3	11.5	11.2	22.7	7.4	0.0	0.0	0.0	7.4	30.1
Qrt. 4	11.5	11.6	23.1	7.9	0.0	0.0	0.0	7.9	30.9
1994									
Qrt. 1	11.0	11.4	22.4	4.8	378.4	0.0	0.0	383.3	405.7
Qrt. 2	11.0	11.8	22.8	7.4	0.0	0.0	0.0	7.4	30.1
Qrt. 3	11.0	12.1	23.1	7.8	0.0	0.0	0.0	7.8	30.9
Qrt. 4	11.0	12.5	23.5	8.3	0.0	0.0	0.0	8.3	31.8

Kidder-Peabody LMFBR (Net(0)/Fix(1) Equity Flag = 0) 10/25/83

Quarterly Construction Financing (millions of current $)

	--------Source of Funds--------				----Quarter----		------Cumulative------		
	ST debt	Lt debt	Pref.	Common	CWIP	AFUDC	CWIP	AFUDC	Total
1987									
Qrt. 1	140.3	0.0	0.0	17.3	157.7	0.0	157.7	0.0	157.7
Qrt. 2	144.0	0.0	0.0	19.1	160.0	3.1	317.7	3.1	320.8
Qrt. 3	147.7	0.0	0.0	21.0	162.4	6.3	480.1	9.4	489.4
Qrt. 4	151.4	0.0	0.0	22.9	164.8	9.5	644.9	18.9	663.8
1988									
Qrt. 1	-383.0	583.4	0.0	23.1	210.7	12.9	855.6	31.8	887.4
Qrt. 2	192.6	0.0	0.0	25.4	213.9	4.1	1069.5	35.9	1105.4
Qrt. 3	197.4	0.0	0.0	27.8	217.1	8.1	1286.7	43.9	1330.6
Qrt. 4	172.4	0.0	0.0	60.2	220.4	12.1	1507.1	56.1	1563.1
1989									
Qrt. 1	-467.4	762.9	0.0	27.6	246.8	76.2	1753.9	132.3	1886.2
Qrt. 2	226.3	0.0	0.0	30.2	250.7	5.8	2004.6	138.0	2142.6
Qrt. 3	232.1	0.0	0.0	32.8	254.7	10.2	2259.3	148.2	2407.5
Qrt. 4	170.0	0.0	0.0	103.4	258.7	14.7	2518.1	162.9	2681.0
1990									
Qrt. 1	-601.4	923.9	0.0	22.1	189.6	155.1	2707.6	318.0	3025.6
Qrt. 2	174.9	0.0	0.0	24.2	192.7	6.4	2900.4	324.4	3224.7
Qrt. 3	179.5	0.0	0.0	26.3	196.0	9.8	3096.3	334.2	3430.5
Qrt. 4	71.4	0.0	0.0	141.3	199.3	13.4	3295.6	347.6	3643.2
1991									
Qrt. 1	-409.5	748.3	0.0	14.4	110.5	242.7	3406.2	590.2	3996.4
Qrt. 2	103.4	0.0	0.0	15.5	112.4	6.6	3518.5	596.8	4115.4
Qrt. 3	106.1	0.0	0.0	16.8	114.2	8.6	3632.8	605.5	4238.2
Qrt. 4	-40.6	0.0	0.0	167.4	116.1	10.7	3748.9	616.2	4365.1
1992									
Qrt. 1	-168.6	507.8	0.0	6.6	34.1	311.7	3783.0	927.8	4710.8
Qrt. 2	34.1	0.0	0.0	7.0	34.6	6.4	3817.6	934.3	4751.9
Qrt. 3	34.9	0.0	0.0	7.4	35.2	7.1	3852.9	941.3	4794.2
Qrt. 4	-138.6	0.0	0.0	182.1	35.8	7.8	3888.6	949.1	4837.7
1993									
Qrt. 1	103.9	269.5	0.0	5.7	22.0	357.1	3910.6	1306.2	5216.8
Qrt. 2	23.4	0.0	0.0	6.0	22.3	7.0	3933.0	1313.2	5246.2
Qrt. 3	23.9	0.0	0.0	6.2	22.7	7.4	3955.7	1320.6	5276.3
Qrt. 4	-163.0	0.0	0.0	193.9	23.1	7.9	3978.7	1328.5	5307.2
1994									
Qrt. 1	142.0	257.7	0.0	5.9	22.4	383.3	4001.1	1711.8	5712.9
Qrt. 2	24.0	0.0	0.0	6.2	22.8	7.4	4023.9	1719.2	5743.1
Qrt. 3	24.5	0.0	0.0	6.5	23.1	7.8	4047.0	1727.0	5774.0
Qrt. 4	-174.9	0.0	0.0	206.7	23.5	8.3	4070.5	1735.3	5805.8

Kidder-Peabody LMFBR (Net(0)/Fix(1) Equity Flag = 0) 10/25/83

Annual Construction Expenses (millions of current $)

Year	--Direct Construction-- ON Cost	Infl.	Total	-----Interest and Equity Charges----- ST debt	LT debt	Pref.	Common	Total	Total Cost
1987	482.0	162.9	644.9	18.9	0.0	0.0	0.0	18.9	663.8
1988	607.0	255.2	862.2	37.2	0.0	0.0	0.0	37.2	899.4
1989	669.0	342.0	1011.0	46.3	60.6	0.0	0.0	106.9	1117.8
1990	482.0	295.6	777.6	47.6	137.1	0.0	0.0	184.7	962.2
1991	263.0	190.3	453.3	40.7	227.9	0.0	0.0	268.6	721.9
1992	76.0	63.7	139.7	31.2	301.8	0.0	0.0	332.9	472.7
1993	46.0	44.1	90.1	27.4	352.0	0.0	0.0	379.4	469.5
1994	44.0	47.7	91.7	28.4	378.4	0.0	0.0	406.8	498.5
Accru:	0.0	0.0	0.0	5.1	403.6	0.0	0.0	408.6	408.6
Total:	2669.0	1401.5	4070.5	282.7	1861.3	0.0	0.0	2144.0	6214.4

Annual Construction Financing (millions of current $)

Year	--------Source of Funds-------- ST debt	Lt debt	Pref.	Common	----Annual----- CWIP	AFUDC	------Cumulative------ CWIP	AFUDC	Total
1987	583.4	0.0	0.0	80.4	644.9	18.9	644.9	18.9	663.8
1988	179.5	583.4	0.0	136.5	862.2	37.2	1507.1	56.1	1563.1
1989	161.0	762.9	0.0	194.0	1011.0	106.9	2518.1	162.9	2681.0
1990	-175.5	923.9	0.0	213.9	777.6	184.7	3295.6	347.6	3643.2
1991	-240.6	748.3	0.0	214.1	453.3	268.6	3748.9	616.2	4365.1
1992	-238.3	507.8	0.0	203.2	139.7	332.9	3888.6	949.1	4837.7
1993	-11.8	269.5	0.0	211.8	90.1	379.4	3978.7	1328.5	5307.2
1994	15.6	257.7	0.0	225.3	91.7	406.8	4070.5	1735.3	5805.8
Accru:	-273.3	681.9	0.0	0.0	0.0	408.6	4070.5	2143.9	6214.4
Total:	-0.0	4735.4	0.0	1479.0	4070.5	2144.0	4070.5	2143.9	6214.4

Construction-End Summary

Accruals (million $) Inv.Tx.Cr.	Tax Sav.	------Direct------ Current	Base Year	------AFUDC------- Current	Base Year	---Weighted Rate--- Preferred	LT debt
407.0	1072.0	4070.5	2669.0	2144.0	1158.9	0.00000	0.09927

Kidder-Peabody LMFBR (Net(0)/Fix(1) Equity Flag = 0) 10/25/83

CONSTRUCTION SUMMARY
(All Amounts in millions of current year $)

Period: (1987,1995)
Base Year: 1982

Expenses:	Total	Book Life	Book Dep.
Direct :	4070.5	30	135.7
Startup:	0.0	30	0.0
AFUDC :	2144.0	30	71.5
Wk.Cap.:	0.0	--	--
Total :	6214.4	--	207.1

Financing:	Total	% Financing	Charge Rate
Common :	1479.0	23.80	14.15
Preferred:	0.0	0.00	0.00
LT.Debt :	4735.4	76.20	9.93
Total :	6214.4	100.00	10.98

Accruals:
ITC: 407.0
FIT: 1072.0

OPERATING SUMMARY

Acceleration Tax Depreciation:
Tax Life : 25 (years)
Method : Straight-line
Basis : 3866.9
TEFRA Opt.: 95 %

Option Flags

IDC Compounding	: Simple
Financing Period	: Quarter
Property Tax Basis	: Net
Equity Finan. Basis	: Zero Net Equity
Discount Rate	: Pre-tax
Analysis Type	: Bottom-up
Accounting Basis	: Flow-thru
Reg. Tax Treatment	: Unconsolidated

Analysis Summary:	ROI (%)	Lev. Busbar (cents/kWh)	Discount Rate (%)	Fixed Charge Rate (%)
Current:	-1000.00	18.53	10.98	14.84
Base Yr:	-1000.00	5.32	4.50	6.92

Kidder-Peabody LMFBR (Net(0)/Fix(1) Equity Flag = 0) 10/25/83

AFTER-TAX CASH FLOWS TO UTILITY INVESTORS
(Millions of Current Year $)

Year	During Construction Cash Inv.	Tax Ben.	During Operation Tax Due	Cash Dist.	Net A.T. Cash Flow	Cum.A.T. Cash Flow	Cm.Dis.R. A.T.C.F.
1987	80.367	80.367	--	--	0.000	0.000	0.000
1988	136.479	136.479	--	--	0.000	0.000	0.000
1989	193.962	193.962	--	--	0.000	0.000	0.000
1990	213.876	213.876	--	--	0.000	0.000	0.000
1991	214.121	214.120	--	--	0.000	0.000	0.000
1992	203.165	203.165	--	--	0.000	0.000	0.000
1993	211.812	211.811	--	--	0.000	0.000	0.000
1994	225.260	225.260	--	--	0.000	0.000	0.000
1995	--	--	228.744	454.310	225.567	225.567	53.580
1996	--	--	191.319	379.462	188.143	413.710	93.847
1997	--	--	161.032	318.889	157.857	571.566	124.290
1998	--	--	155.975	308.776	152.801	724.367	150.841
1999	--	--	150.919	298.664	147.745	872.112	173.973
2000	--	--	145.862	288.551	142.688	1014.800	194.103
2001	--	--	140.806	278.438	137.632	1152.433	211.598
2002	--	--	135.749	268.325	132.576	1285.008	226.782
2003	--	--	130.693	258.213	127.520	1412.528	239.943
2004	--	--	125.636	248.100	122.463	1534.991	251.330
2005	--	--	120.580	237.987	117.407	1652.399	261.167
2006	--	--	115.524	227.875	112.351	1764.749	269.649
2007	--	--	110.467	217.762	107.295	1872.044	276.948
2008	--	--	105.411	207.649	102.238	1974.283	283.214
2009	--	--	100.354	197.536	97.182	2071.465	288.581
2010	--	--	95.298	187.424	92.126	2163.590	293.166
2011	--	--	90.241	177.311	87.070	2250.660	297.070
2012	--	--	85.185	167.198	82.013	2332.673	300.383
2013	--	--	80.128	157.086	76.957	2409.630	303.185
2014	--	--	75.072	146.973	71.901	2481.531	305.543
2015	--	--	70.015	136.860	66.845	2548.376	307.519
2016	--	--	64.959	126.747	61.788	2610.164	309.164
2017	--	--	59.903	116.635	56.732	2666.896	310.525
2018	--	--	54.846	106.522	51.676	2718.572	311.643
2019	--	--	49.790	96.409	46.619	2765.191	312.551
2020	--	--	122.073	86.296	-35.777	2729.414	311.923
2021	--	--	117.017	76.184	-40.833	2688.581	311.277
2022	--	--	111.960	66.071	-45.889	2642.692	310.623
2023	--	--	106.904	55.958	-50.946	2591.746	309.969
2024	--	--	101.848	45.846	-56.002	2535.744	309.321
TOT.	1479.041	1479.041	3404.311	5940.050	2535.744		

Summary of ROIs (Current $)

1995 -*****	1996 -*****	1997 -*****	1998 -*****	1999 -*****
2000 -*****	2001 -*****	2002 -*****	2003 -*****	2004 -*****
2005 -*****	2006 -*****	2007 -*****	2008 -*****	2009 -*****
2010 -*****	2011 -*****	2012 -*****	2013 -*****	2014 -*****
2015 -*****	2016 -*****	2017 -*****	2018 -*****	2019 -*****
2020 -*****	2021 -*****	2022 -*****	2023 -*****	2024 -*****

Summary of ROIs (Base-Year $)

1995 -*****	1996 -*****	1997 -*****	1998 -*****	1999 -*****
2000 -*****	2001 -*****	2002 -*****	2003 -*****	2004 -*****
2005 -*****	2006 -*****	2007 -*****	2008 -*****	2009 -*****
2010 -*****	2011 -*****	2012 -*****	2013 -*****	2014 -*****
2015 -*****	2016 -*****	2017 -*****	2018 -*****	2019 -*****
2020 -*****	2021 -*****	2022 -*****	2023 -*****	2024 -*****

D.5 RUN ILLUSTRATING USE OF THE COMPOUND(0)/SIMPLE(1)FLAG

This run illustrates the use of the COMPOUND(0)/SIMPLE(1)FLAG and contains the construction summary, summary, and after-tax cash flow reports. All inputs are the same as in Sec. D.2 except that the COMPOUND(0)/SIMPLE(1)FLAG has been set to 0. This results in a slightly lower AFUDC charge, due to lower effective interest rates.

Kidder-Peabody LMFBR (Compound(0)/Simple(1) Flag = 0) 10/25/83

Quarterly Construction Expenses (millions of current $)

	--Direct Construction--			-----Interest and Equity Charges-----					Total
	ON Cost	Infl.	Total	ST debt	LT debt	Pref.	Common	Total	Cost
1987									
Qrt. 1	120.5	37.2	157.7	0.0	0.0	0.0	0.0	0.0	157.7
Qrt. 2	120.5	39.5	160.0	2.5	0.0	0.0	0.0	2.5	162.5
Qrt. 3	120.5	41.9	162.4	5.1	0.0	0.0	0.0	5.1	167.5
Qrt. 4	120.5	44.3	164.8	7.8	0.0	0.0	0.0	7.8	172.6
1988									
Qrt. 1	151.8	59.0	210.7	10.6	0.0	0.0	0.0	10.6	221.3
Qrt. 2	151.8	62.2	213.9	3.3	0.0	0.0	0.0	3.3	217.2
Qrt. 3	151.8	65.4	217.1	6.6	0.0	0.0	0.0	6.6	223.7
Qrt. 4	151.8	68.7	220.4	9.9	0.0	0.0	0.0	9.9	230.3
1989									
Qrt. 1	167.2	79.6	246.8	13.3	51.4	0.0	0.0	64.7	311.6
Qrt. 2	167.2	83.5	250.7	4.4	0.0	0.0	0.0	4.4	255.2
Qrt. 3	167.2	87.5	254.7	8.1	0.0	0.0	0.0	8.1	262.8
Qrt. 4	167.2	91.5	258.7	11.8	0.0	0.0	0.0	11.8	270.5
1990									
Qrt. 1	120.5	69.1	189.6	15.6	118.5	0.0	0.0	134.2	323.7
Qrt. 2	120.5	72.2	192.7	4.7	0.0	0.0	0.0	4.7	197.4
Qrt. 3	120.5	75.5	196.0	7.5	0.0	0.0	0.0	7.5	203.5
Qrt. 4	120.5	78.8	199.3	10.4	0.0	0.0	0.0	10.4	209.7
1991									
Qrt. 1	65.8	44.8	110.5	13.4	199.6	0.0	0.0	213.1	323.6
Qrt. 2	65.8	46.6	112.4	4.6	0.0	0.0	0.0	4.6	117.0
Qrt. 3	65.8	48.5	114.2	6.3	0.0	0.0	0.0	6.3	120.5
Qrt. 4	65.8	50.4	116.1	8.0	0.0	0.0	0.0	8.0	124.1
1992									
Qrt. 1	19.0	15.1	34.1	9.7	268.8	0.0	0.0	278.6	312.7
Qrt. 2	19.0	15.6	34.6	4.3	0.0	0.0	0.0	4.3	39.0
Qrt. 3	19.0	16.2	35.2	4.9	0.0	0.0	0.0	4.9	40.1
Qrt. 4	19.0	16.8	35.8	5.4	0.0	0.0	0.0	5.4	41.2
1993									
Qrt. 1	11.5	10.5	22.0	6.0	319.6	0.0	0.0	325.6	347.6
Qrt. 2	11.5	10.8	22.3	4.8	0.0	0.0	0.0	4.8	27.1
Qrt. 3	11.5	11.2	22.7	5.1	0.0	0.0	0.0	5.1	27.8
Qrt. 4	11.5	11.6	23.1	5.5	0.0	0.0	0.0	5.5	28.6
1994									
Qrt. 1	11.0	11.4	22.4	5.9	351.5	0.0	0.0	357.4	379.8
Qrt. 2	11.0	11.8	22.8	5.1	0.0	0.0	0.0	5.1	27.9
Qrt. 3	11.0	12.1	23.1	5.5	0.0	0.0	0.0	5.5	28.6
Qrt. 4	11.0	12.5	23.5	5.9	0.0	0.0	0.0	5.9	29.4

Kidder-Peabody LMFBR (Compound(0)/Simple(1) Flag = 0) 10/25/83

Quarterly Construction Financing (millions of current $)

	--------Source of Funds--------				----Quarter----		------Cumulative------		
	ST debt	Lt debt	Pref.	Common	CWIP	AFUDC	CWIP	AFUDC	Total
1987									
Qrt. 1	118.2	0.0	0.0	39.4	157.7	0.0	157.7	0.0	157.7
Qrt. 2	121.9	0.0	0.0	40.6	160.0	2.5	317.7	2.5	320.2
Qrt. 3	125.6	0.0	0.0	41.9	162.4	5.1	480.1	7.7	487.7
Qrt. 4	129.5	0.0	0.0	43.2	164.8	7.8	644.9	15.5	660.4
1988									
Qrt. 1	-329.3	495.3	0.0	55.3	210.7	10.6	855.6	26.0	881.6
Qrt. 2	162.9	0.0	0.0	54.3	213.9	3.3	1069.5	29.3	1098.9
Qrt. 3	167.8	0.0	0.0	55.9	217.1	6.6	1286.7	35.9	1322.6
Qrt. 4	172.7	0.0	0.0	57.6	220.4	9.9	1507.1	45.8	1552.9
1989									
Qrt. 1	-435.7	669.4	0.0	77.9	246.8	64.7	1753.9	110.5	1864.4
Qrt. 2	191.4	0.0	0.0	63.8	250.7	4.4	2004.6	115.0	2119.6
Qrt. 3	197.1	0.0	0.0	65.7	254.7	8.1	2259.3	123.0	2382.3
Qrt. 4	202.9	0.0	0.0	67.6	258.7	11.8	2518.1	134.8	2652.9
1990									
Qrt. 1	-582.2	825.0	0.0	80.9	189.6	134.2	2707.6	269.0	2976.6
Qrt. 2	148.1	0.0	0.0	49.4	192.7	4.7	2900.4	273.7	3174.0
Qrt. 3	152.6	0.0	0.0	50.9	196.0	7.5	3096.3	281.2	3377.5
Qrt. 4	157.3	0.0	0.0	52.4	199.3	10.4	3295.6	291.6	3587.2
1991									
Qrt. 1	-458.0	700.8	0.0	80.9	110.5	213.1	3406.2	504.7	3910.8
Qrt. 2	87.7	0.0	0.0	29.2	112.4	4.6	3518.5	509.3	4027.8
Qrt. 3	90.4	0.0	0.0	30.1	114.2	6.3	3632.8	515.6	4148.3
Qrt. 4	93.1	0.0	0.0	31.0	116.1	8.0	3748.9	523.5	4272.4
1992									
Qrt. 1	-279.4	513.9	0.0	78.2	34.1	278.6	3783.0	802.1	4585.1
Qrt. 2	29.2	0.0	0.0	9.7	34.6	4.3	3817.6	806.4	4624.1
Qrt. 3	30.1	0.0	0.0	10.0	35.2	4.9	3852.9	811.3	4664.1
Qrt. 4	30.9	0.0	0.0	10.3	35.8	5.4	3888.6	816.7	4705.3
1993									
Qrt. 1	-64.0	324.7	0.0	86.9	22.0	325.6	3910.6	1142.4	5053.0
Qrt. 2	20.3	0.0	0.0	6.8	22.3	4.8	3933.0	1147.1	5080.1
Qrt. 3	20.9	0.0	0.0	7.0	22.7	5.1	3955.7	1152.2	5107.9
Qrt. 4	21.4	0.0	0.0	7.1	23.1	5.5	3978.7	1157.7	5136.5
1994									
Qrt. 1	-38.5	323.3	0.0	95.0	22.4	357.4	4001.1	1515.2	5516.3
Qrt. 2	20.9	0.0	0.0	7.0	22.8	5.1	4023.9	1520.3	5544.2
Qrt. 3	21.5	0.0	0.0	7.2	23.1	5.5	4047.0	1525.8	5572.8
Qrt. 4	22.0	0.0	0.0	7.3	23.5	5.9	4070.5	1531.7	5602.1

Kidder-Peabody LMFBR (Compound(0)/Simple(1) Flag = 0) 10/25/83

Annual Construction Expenses (millions of current $)

	--Direct Construction--			-----Interest and Equity Charges-----					Total
Year	ON Cost	Infl.	Total	ST debt	LT debt	Pref.	Common	Total	Cost
1987	482.0	162.9	644.9	15.5	0.0	0.0	0.0	15.5	660.4
1988	607.0	255.2	862.2	30.3	0.0	0.0	0.0	30.3	892.5
1989	669.0	342.0	1011.0	37.6	51.4	0.0	0.0	89.0	1100.0
1990	482.0	295.6	777.6	38.2	118.5	0.0	0.0	156.8	934.3
1991	263.0	190.3	453.3	32.3	199.6	0.0	0.0	231.9	685.2
1992	76.0	63.7	139.7	24.4	268.8	0.0	0.0	293.2	432.9
1993	46.0	44.1	90.1	21.4	319.6	0.0	0.0	341.0	431.1
1994	44.0	47.7	91.7	22.4	351.5	0.0	0.0	373.9	465.7
Accru:	0.0	0.0	0.0	6.3	383.1	0.0	0.0	389.4	389.4
Total:	2669.0	1401.5	4070.5	228.4	1692.6	0.0	0.0	1921.1	5991.5

Annual Construction Financing (millions of current $)

	--------Source of Funds--------				----Annual-----		------Cumulative------		
Year	ST debt	Lt debt	Pref.	Common	CWIP	AFUDC	CWIP	AFUDC	Total
1987	495.3	0.0	0.0	165.1	644.9	15.5	644.9	15.5	660.4
1988	174.1	495.3	0.0	223.1	862.2	30.3	1507.1	45.8	1552.9
1989	155.6	669.4	0.0	275.0	1011.0	89.0	2518.1	134.8	2652.9
1990	-124.2	825.0	0.0	233.6	777.6	156.8	3295.6	291.6	3587.2
1991	-186.8	700.8	0.0	171.3	453.3	231.9	3748.9	523.5	4272.4
1992	-189.2	513.9	0.0	108.2	139.7	293.2	3888.6	816.7	4705.3
1993	-1.3	324.7	0.0	107.8	90.1	341.0	3978.7	1157.7	5136.5
1994	25.9	323.3	0.0	116.4	91.7	373.9	4070.5	1531.7	5602.1
Accru:	-349.3	641.3	0.0	97.3	0.0	389.4	4070.5	1921.1	5991.5
Total:	-0.0	4493.6	0.0	1497.9	4070.5	1921.1	4070.5	1921.1	5991.5

Construction-End Summary

Accruals (million $)		------Direct------		------AFUDC-------		---Weighted Rate---	
Inv.Tx.Cr.	Tax Sav.	Current	Base Year	Current	Base Year	Preferred	LT debt
407.0	960.5	4070.5	2669.0	1921.1	1033.5	0.00000	0.09916

Kidder-Peabody LMFBR (Compound(0)/Simple(1) Flag = 0) 10/25/83

CONSTRUCTION SUMMARY
(All Amounts in millions of current year $)

Period: (1987,1995)
Base Year: 1982

Expenses:	Total	Book Life	Book Dep.
Direct :	4070.5	30	135.7
Startup:	0.0	30	0.0
AFUDC :	1921.1	30	64.0
Wk.Cap.:	0.0	--	--
Total :	5991.5	--	199.7

Financing:	Total	% Financing	Charge Rate
Common :	1497.9	25.00	14.15
Preferred:	0.0	0.00	0.00
LT.Debt :	4493.6	75.00	9.92
Total :	5991.5	100.00	10.97

Accruals:
ITC: 407.0
FIT: 960.5

OPERATING SUMMARY
Acceleration Tax Depreciation:
Tax Life : 25 (years)
Method : Straight-line
Basis : 3866.9
TEFRA Opt.: 95 %

Option Flags
IDC Compounding : Compound
Financing Period : Quarter
Property Tax Basis : Net
Equity Finan. Basis: Fixed %
Discount Rate : Pre-tax
Analysis Type : Bottom-up
Accounting Basis : Flow-thru
Reg. Tax Treatment : Unconsolidated

Analysis Summary:	ROI (%)	Lev. Busbar (cents/kWh)	Discount Rate (%)	Fixed Charge Rate (%)
Current:	22.10	18.16	10.97	14.95
Base Yr:	14.69	5.22	4.50	6.95

Kidder-Peabody LMFBR (Compound(0)/Simple(1) Flag = 0) 10/25/83

AFTER-TAX CASH FLOWS TO UTILITY INVESTORS
(Millions of Current Year $)

Year	During Construction Cash Inv.	Tax Ben.	During Operation Tax Due	Cash Dist.	Net A.T. Cash Flow	Cum.A.T. Cash Flow	Cm.Dis.R. A.T.C.F.
1987	165.091	77.511	--	--	-87.579	-87.579	-48.994
1988	223.131	128.469	--	--	-94.662	-182.241	-96.758
1989	274.998	180.330	--	--	-94.668	-276.909	-139.801
1990	233.584	195.600	--	--	-37.984	-314.893	-155.320
1991	171.303	194.027	--	--	22.724	-292.169	-147.009
1992	108.229	184.113	--	--	75.884	-216.285	-122.140
1993	107.782	195.420	--	--	87.638	-128.648	-96.357
1994	213.762	212.126	--	--	-1.636	-130.284	-96.789
1995	--	--	227.695	460.269	232.575	102.291	-41.490
1996	--	--	190.329	385.540	195.211	297.502	0.335
1997	--	--	159.805	324.492	164.688	462.190	32.131
1998	--	--	154.659	314.202	159.543	621.732	59.887
1999	--	--	149.514	303.912	154.398	776.129	84.092
2000	--	--	144.369	293.621	149.252	925.382	105.176
2001	--	--	139.224	283.331	144.107	1069.489	123.520
2002	--	--	134.078	273.040	138.962	1208.452	139.460
2003	--	--	128.933	262.750	133.817	1342.269	153.292
2004	--	--	123.788	252.460	128.672	1470.941	165.276
2005	--	--	118.642	242.169	123.527	1594.468	175.644
2006	--	--	113.497	231.879	118.382	1712.850	184.597
2007	--	--	108.352	221.588	113.237	1826.086	192.314
2008	--	--	103.206	211.298	108.092	1934.178	198.952
2009	--	--	98.061	201.008	102.947	2037.125	204.649
2010	--	--	92.916	190.717	97.801	2134.926	209.525
2011	--	--	87.770	180.427	92.656	2227.582	213.689
2012	--	--	82.625	170.136	87.511	2315.093	217.232
2013	--	--	77.480	159.846	82.366	2397.460	220.237
2014	--	--	72.335	149.556	77.221	2474.681	222.776
2015	--	--	67.189	139.265	72.076	2546.757	224.911
2016	--	--	62.044	128.975	66.931	2613.687	226.698
2017	--	--	56.899	118.684	61.786	2675.473	228.184
2018	--	--	51.753	108.394	56.641	2732.114	229.412
2019	--	--	46.608	98.104	51.496	2783.609	230.418
2020	--	--	118.803	87.813	-30.990	2752.620	229.873
2021	--	--	113.657	77.523	-36.135	2716.485	229.300
2022	--	--	108.512	67.232	-41.280	2675.205	228.710
2023	--	--	103.367	56.942	-46.425	2628.780	228.112
2024	--	--	98.222	46.651	-51.570	2577.209	227.513
TOT.	1497.878	1367.595	3334.331	6041.820	2577.209		

Summary of ROIs (Current $)

1995	5.22	1996	11.38	1997	14.58	1998	16.69	1999	18.15
2000	19.17	2001	19.92	2002	20.46	2003	20.86	2004	21.16
2005	21.39	2006	21.56	2007	21.70	2008	21.80	2009	21.87
2010	21.93	2011	21.98	2012	22.01	2013	22.04	2014	22.06
2015	22.07	2016	22.09	2017	22.09	2018	22.10	2019	22.11
2020	22.10	2021	22.10	2022	22.10	2023	22.10	2024	22.10

Summary of ROIs (Base-Year $)

1995	-0.64	1996	4.53	1997	7.55	1998	9.55	1999	10.93
2000	11.90	2001	12.61	2002	13.12	2003	13.51	2004	13.80
2005	14.01	2006	14.18	2007	14.31	2008	14.40	2009	14.48
2010	14.53	2011	14.58	2012	14.61	2013	14.64	2014	14.66
2015	14.67	2016	14.68	2017	14.69	2018	14.70	2019	14.70
2020	14.70	2021	14.70	2022	14.70	2023	14.69	2024	14.69

D.6 RUN ILLUSTRATING USE OF THE BOT.UP(0)/TOP.DN.(1) FLAG

This run illustrates the use of the BOT.UP(0)/TOP.DN(1) FLAG and contains all but the construction summary reports. For this run, the annual busbar costs, as presented in Sec. D.2, were used as inputs to the BUSBAR cost array and the BOT.UP(0)/TOP.DN.(1) FLAG was set to 1. Otherwise all inputs are the same as in D.2. Since the busbar costs output from the D.2 run have been rounded to two decimal places, slight discrepancies between the results in D.2 and D.6 exist.

Kidder-Peabody LMFBR (Bot.Up(0)/Top.Dn.(1) Flag = 1) 10/25/83

CONSTRUCTION SUMMARY
(All Amounts in millions of current year $)

Period: (1987,1995)
Base Year: 1982

Expenses:	Total	Book Life	Book Dep.
Direct :	4070.5	30	135.7
Startup:	0.0	30	0.0
AFUDC :	1930.1	30	64.3
Wk.Cap.:	0.0	--	--
Total :	6000.5	--	200.0

Financing:	Total	% Financing	Charge Rate
Common :	1500.1	25.00	14.15
Preferred:	0.0	0.00	0.00
LT.Debt :	4500.4	75.00	9.92
Total :	6000.5	100.00	10.97

Accruals:
ITC: 407.0
FIT: 965.1

OPERATING SUMMARY
Acceleration Tax Depreciation:
Tax Life : 25 (years)
Method : Straight-line
Basis : 3866.9
TEFRA Opt.: 95 %

Option Flags
IDC Compounding : Simple
Financing Period : Quarter
Property Tax Basis : Net
Equity Finan. Basis: Fixed %
Discount Rate : Pre-tax
Analysis Type : Top-down
Accounting Basis : Flow-thru
Reg. Tax Treatment : Unconsolidated

Analysis Summary:	ROI (%)	Lev. Busbar (cents/kWh)	Discount Rate (%)	Fixed Charge Rate (%)
Current:	23.02	18.72	10.97	15.59
Base Yr:	15.56	5.38	4.50	7.25

Kidder-Peabody LMFBR (Bot.Up(0)/Top.Dn.(1) Flag = 1) 10/25/83

BALANCE SHEET
(Millions of Current Year $)

	1995	2002	2010	2018	2024
ASSETS					
Cum. Working Capital	0.0	0.0	0.0	0.0	0.0
Gross Plant	4070.5	4070.5	4070.5	4070.5	4070.5
Less Accum.Plant Dep.	135.7	1085.5	2170.9	3256.4	4070.5
Net Plant	3934.8	2985.0	1899.5	814.1	-0.0
Capitalized AFUDC	1930.1	1930.1	1930.1	1930.1	1930.1
Less Accum.AFUDC Dep.	64.3	514.7	1029.4	1544.1	1930.1
Net Capitalized AFUDC	1865.7	1415.4	900.7	386.0	0.0
TOTAL ASSETS	5800.5	4400.4	2800.3	1200.1	-0.0
LIABILITIES					
LTD Due in Next Year	150.0	150.0	150.0	150.0	0.0
LTD Outstanding	4200.4	3150.3	1950.2	750.1	0.0
Cum. Deferred ITC	216.1	298.5	190.0	81.4	0.0
Cum. Deferred FIT	0.0	0.0	0.0	0.0	0.0
P.S. Due in Next Year	0.0	0.0	0.0	0.0	0.0
Pref.Stock Outstanding	0.0	0.0	0.0	0.0	0.0
End Common Equity	1234.1	801.6	510.1	218.6	-0.0
Partner Loans Outst.	0.0	0.0	0.0	0.0	0.0
Total Stock Equity	1234.1	801.6	510.1	218.6	-0.0
TOTAL LIABIL. & EQUITY	5800.5	4400.4	2800.3	1200.1	-0.0

Kidder-Peabody LMFBR (Bot.Up(0)/Top.Dn.(1) Flag = 1) 10/25/83

INCOME STATEMENT
(Millions of Current Year $)

	1995	2002	2010	2018	2024
REVENUES					
Electric Sales	1438.4	1299.7	1335.2	1521.0	1812.8
Federal Subsidy	0.0	0.0	0.0	0.0	0.0
TOTAL REVENUES	1438.4	1299.7	1335.2	1521.0	1812.8
EXPENSES					
Fuel Expense	156.5	238.5	385.8	624.3	895.7
O&M Expense	101.7	155.0	250.8	405.8	582.2
Backfit&Replacement	0.0	0.0	0.0	0.0	0.0
Ad Valorem Tax	0.0	0.0	0.0	0.0	0.0
Total Oper. Expense	258.2	393.5	636.6	1030.1	1477.9
Plant Book Dep.	135.7	135.7	135.7	135.7	135.7
AFUDC Book Dep.	64.3	64.3	64.3	64.3	64.3
Total Book Dep.	200.0	200.0	200.0	200.0	200.0
Property Tax	120.0	92.0	60.0	28.0	4.0
Net FIT (Book Basis)	200.2	129.2	100.9	72.6	51.2
Total Pre-Debt Expen.	778.4	814.7	997.6	1330.7	1733.1
Pre-Debt Net Income	660.0	484.9	337.6	190.3	79.7
LTD Interest	446.3	342.1	223.1	104.1	14.9
NET INCOME	213.7	142.8	114.5	86.2	64.8
Less Pref. Dividends	0.0	0.0	0.0	0.0	0.0
Common Stock Earnings	213.7	142.8	114.5	86.2	64.8
BUSBAR COST (cents/kWh)	20.2	18.3	18.8	21.4	23.5

Kidder-Peabody LMFBR (Bot.Up(0)/Top.Dn.(1) Flag = 1) 10/25/83

SOURCES AND USES OF FUNDS STATEMENT
(Millions of Current Year $)

	1995	2002	2010	2018	2024
SOURCES OF FUNDS					
Net Income	213.7	142.8	114.5	86.2	64.8
Plant Book Dep.	135.7	135.7	135.7	135.7	135.7
AFUDC Book Dep.	64.3	64.3	64.3	64.3	64.3
Total Book Dep.	200.0	200.0	200.0	200.0	200.0
Deferred ITC	216.1	-13.6	-13.6	-13.6	-13.6
Deferred FIT	0.0	0.0	0.0	0.0	0.0
TOTAL INTERNAL	629.8	329.2	301.0	272.6	251.3
Partner Loans	0.0	0.0	0.0	0.0	0.0
Government Loans	0.0	0.0	0.0	0.0	0.0
TOTAL SOURCES	629.8	329.2	301.0	272.6	251.3
USES OF FUNDS					
Preferred Stock Due	0.0	0.0	0.0	0.0	0.0
Preferred Dividends	0.0	0.0	0.0	0.0	0.0
Common Dividends	479.8	179.2	150.9	122.6	101.3
Working Capital Change	0.0	0.0	0.0	0.0	0.0
LTD Repayment	150.0	150.0	150.0	150.0	150.0
Partner Loan Repayment	0.0	0.0	0.0	0.0	0.0
TOTAL USES	629.8	329.2	301.0	272.6	251.3

Kidder-Peabody LMFBR (Bot.Up(0)/Top.Dn.(1) Flag = 1) 10/25/83

RETAINED EARNINGS STATEMENT & TAX DEP. WORKSHEET
(Millions of Current Year $)

	1995	2002	2010	2018	2024
RETAINED EARNINGS					
Begin.Common Equity	1500.1	838.0	546.5	255.1	36.4
Common Stock Earnings	213.7	142.8	114.5	86.2	64.8
Total Common Equity	1713.9	980.8	661.0	341.2	101.3
Less Common Dividends	479.8	179.2	150.9	122.6	101.3
END COMMON EQUITY	1234.1	801.6	510.1	218.6	-0.0
RETURN ON EQUITY (%)	14.2	17.0	21.0	33.8	177.9
TAX DEP. WORKSHEET					
Startup Dep.	0.0	0.0	0.0	0.0	0.0
5-year Prop.Dep.	0.0	0.0	0.0	0.0	0.0
10-year Prop.Dep.	0.0	0.0	0.0	0.0	0.0
15-year Prop.Dep.	0.0	0.0	0.0	0.0	0.0
Straight-line Dep.	154.7	154.7	154.7	154.7	0.0
TOTAL TAX DEPRECIATION	154.7	154.7	154.7	154.7	0.0

Kidder-Peabody LMFBR (Bot.Up(0)/Top.Dn.(1) Flag = 1) 10/25/83

TAX CALCULATION (Book/Regulation Basis)
(Millions of Current Year $)

	1995	2002	2010	2018	2024
BOOK TAX CALCULATION					
Electric Sales	1438.4	1299.7	1335.2	1521.0	1812.8
Total Oper. Expense	258.2	393.5	636.6	1030.1	1477.9
Total Book Dep.	200.0	200.0	200.0	200.0	200.0
Property Tax	120.0	92.0	60.0	28.0	4.0
LTD Interest	446.3	342.1	223.1	104.1	14.9
Pre-tax Book Income	427.5	285.6	229.0	172.3	129.6
Book FIT	213.7	142.8	114.5	86.2	64.8
Less amort.ITC	13.6	13.6	13.6	13.6	13.6
NET FIT (Book Basis)	200.2	129.2	100.9	72.6	51.2
REGULATORY TAX CALCULATION					
Electric Sales	1438.4	1299.7	1335.2	1521.0	1812.8
Total Oper. Expense	258.2	393.5	636.6	1030.1	1477.9
Total Tax Depreciation	154.7	154.7	154.7	154.7	0.0
Property Tax	120.0	92.0	60.0	28.0	4.0
LTD Interest	446.3	342.1	223.1	104.1	14.9
Pre-tax Part. Income	459.2	317.4	260.8	204.1	316.1
FIT Due	229.6	158.7	130.4	102.0	158.0
Less ITC Taken	229.6	0.0	0.0	0.0	0.0
Less FIT Taken	0.0	0.0	0.0	0.0	0.0
NET REGULATED FIT	0.0	158.7	130.4	102.0	158.0

Kidder-Peabody LMFBR (Bot.Up(0)/Top.Dn.(1) Flag = 1) 10/25/83

PARTNERSHIP RETURN AND CASHFLOW ANALYSIS
(Millions of Current Year $)

	1995	2002	2010	2018	2024
PARTNERSHIP RETURN					
Electric Sales	1438.4	1299.7	1335.2	1521.0	1812.8
Total Oper. Expense	258.2	393.5	636.6	1030.1	1477.9
Total Tax Depreciation	154.7	154.7	154.7	154.7	0.0
Property Tax	120.0	92.0	60.0	28.0	4.0
LTD Interest	446.3	342.1	223.1	104.1	14.9
Pre-tax Part. Income	459.2	317.4	260.8	204.1	316.1
FIT Due	229.6	158.7	130.4	102.0	158.0
PARTNERSHIP CASHFLOW ANALYSIS					
After-tax Cash	229.6	158.7	130.4	102.0	158.0
Total Tax Depreciation	154.7	154.7	154.7	154.7	0.0
Federal Subsidy	0.0	0.0	0.0	0.0	0.0
Total Cash Avail.	384.3	313.4	285.1	256.7	158.0
Less LTD Repayment	150.0	150.0	150.0	150.0	150.0
Less Pref. Dividends	0.0	0.0	0.0	0.0	0.0
Less Pref. Stock Due	0.0	0.0	0.0	0.0	0.0
Less Partner Loans	0.0	0.0	0.0	0.0	0.0
Less Work.Cap. Change	0.0	0.0	0.0	0.0	0.0
NET TO PARTNERS	234.3	163.3	135.1	106.7	[illegible]
CUMULATIVE TO PARTNERS	106.3	1323.9	2503.2	3455.4	[illegible]

Kidder-Peabody LMFBR (Bot.Up(0)/Top.Dn.(1) Flag = 1) 10/25/83

AFTER-TAX CASH FLOWS TO UTILITY INVESTORS
(Millions of Current Year $)

Year	During Construction Cash Inv.	Tax Ben.	During Operation Tax Due	Cash Dist.	Net A.T. Cash Flow	Cum.A.T. Cash Flow	Cm.Dis.R. A.T.C.F.
1987	165.218	77.941	--	--	-87.276	-87.276	-48.824
1988	223.374	129.008	--	--	-94.366	-181.642	-96.439
1989	275.294	180.983	--	--	-94.311	-275.952	-139.320
1990	233.902	196.252	--	--	-37.651	-313.603	-154.702
1991	171.600	194.611	--	--	23.011	-290.592	-146.286
1992	108.487	184.617	--	--	76.129	-214.463	-121.337
1993	108.038	195.966	--	--	87.928	-126.534	-95.468
1994	214.222	212.728	--	--	-1.494	-128.028	-95.863
1995	--	--	229.627	463.909	234.282	106.254	-40.158
1996	--	--	179.813	364.282	184.469	290.723	-0.634
1997	--	--	176.478	357.614	181.135	471.858	34.338
1998	--	--	172.973	350.602	177.630	649.487	65.241
1999	--	--	169.263	343.184	173.920	823.408	92.506
2000	--	--	165.671	336.000	170.328	993.736	116.568
2001	--	--	162.159	328.976	166.816	1160.553	137.803
2002	--	--	158.689	322.035	163.346	1323.899	156.540
2003	--	--	155.218	315.093	159.875	1483.774	173.066
2004	--	--	151.702	308.062	156.360	1640.134	187.629
2005	--	--	148.095	300.849	152.753	1792.888	200.450
2006	--	--	144.348	293.353	149.006	1941.893	211.719
2007	--	--	141.117	286.893	145.775	2087.669	221.654
2008	--	--	137.281	279.220	141.939	2229.608	230.370
2009	--	--	133.847	272.352	138.505	2368.113	238.035
2010	--	--	130.395	265.449	135.054	2503.167	244.770
2011	--	--	126.859	258.377	131.518	2634.685	250.679
2012	--	--	123.167	250.992	127.826	2762.510	255.855
2013	--	--	119.599	243.856	124.258	2886.768	260.389
2014	--	--	116.074	236.807	120.733	3007.501	264.358
2015	--	--	112.507	229.673	117.166	3124.667	267.829
2016	--	--	109.163	222.985	113.822	3238.489	270.868
2017	--	--	105.590	215.839	110.249	3348.738	273.520
2018	--	--	102.040	208.740	106.700	3455.438	275.833
2019	--	--	98.406	201.471	103.065	3558.504	277.846
2020	--	--	172.267	194.513	22.246	3580.750	278.238
2021	--	--	168.820	187.620	18.800	3599.550	278.536
2022	--	--	165.276	180.532	15.256	3614.806	278.754
2023	--	--	161.495	172.971	11.476	3626.281	278.902
2024	--	--	158.043	166.067	8.024	3634.305	278.995
TOT.	1500.134	1372.106	4395.980	8158.310	3634.305		

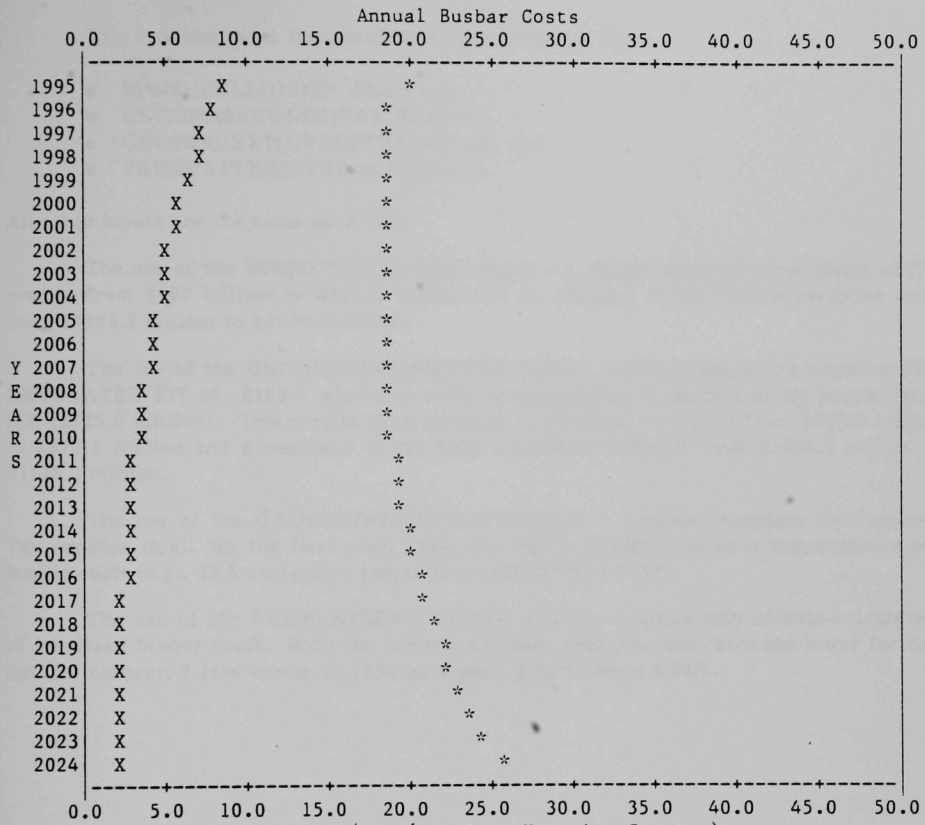
Summary of ROIs (Current $)

1995	5.42	1996	11.27	1997	14.79	1998	17.09	1999	18.66
2000	19.77	2001	20.57	2002	21.16	2003	21.60	2004	21.93
2005	22.18	2006	22.37	2007	22.52	2008	22.63	2009	22.72
2010	22.79	2011	22.85	2012	22.89	2013	22.92	2014	22.95
2015	22.97	2016	22.98	2017	22.99	2018	23.00	2019	23.01
2020	23.01	2021	23.01	2022	23.01	2023	23.01	2024	23.02

Summary of ROIs (Base-Year $)

1995	-0.55	1996	4.43	1997	7.75	1998	9.92	1999	11.41
2000	12.46	2001	13.22	2002	13.78	2003	14.20	2004	14.52
2005	14.76	2006	14.94	2007	15.08	2008	15.19	2009	15.28
2010	15.35	2011	15.40	2012	15.44	2013	15.47	2014	15.50
2015	15.52	2016	15.53	2017	15.54	2018	15.55	2019	15.56
2020	15.56	2021	15.56	2022	15.56	2023	15.56	2024	15.56

Kidder-Peabody LMFBR (Bot.Up(0)/Top.Dn.(1) Flag = 1) 10/25/83



Summary of Levelized Busbar Costs
(Levelized over 20 years)

18.72 (current cents/kWh,discount rate : 10.975 (%))
5.38 (base yr. cents/kWh,discount rate: 4.496 (%))

Summary of Busbar Costs (current cents/kWh)

1995	20.21	1996	18.77	1997	18.65	1998	18.54	1999	18.44
2000	18.36	2001	18.30	2002	18.26	2003	18.24	2004	18.24
2005	18.26	2006	18.30	2007	18.38	2008	18.47	2009	18.60
2010	18.76	2011	18.95	2012	19.17	2013	19.43	2014	19.73
2015	20.07	2016	20.46	2017	20.89	2018	21.37	2019	21.90
2020	22.49	2021	23.14	2022	23.85	2023	24.62	2024	25.47

Summary of Busbar Costs (base yr. cents/kWh)

1995	8.92	1996	7.80	1997	7.30	1998	6.83	1999	6.40
2000	6.00	2001	5.63	2002	5.29	2003	4.98	2004	4.68
2005	4.42	2006	4.17	2007	3.94	2008	3.73	2009	3.54
2010	3.36	2011	3.19	2012	3.04	2013	2.90	2014	2.78
2015	2.66	2016	2.55	2017	2.45	2018	2.36	2019	2.28
2020	2.21	2021	2.14	2022	2.07	2023	2.02	2024	1.96

D.7 RUN ILLUSTRATING USE OF MISCELLANEOUS FLAGS

This run illustrates the use of four other program flags:

- 95%(0)/FULL(1) DEP. FLAG = 1,
- UNCONS(0)/CONS(1)TAX FLAG = 1,
- GROSS(0)/NET(1)PROPTXFLAG = 0, and
- PRE(0)/AFTER(1)TXDISCFLAG = 1.

All other inputs are the same as in D.2.

The use of the 95%(0)/FULL(1) DEP. FLAG = 1 option results in a reduction of ITC credits from $407 million to $325.6 million and an increase in the tax depreciation basis from $3866.9 million to $4070.5 million.

The use of the UNCONS(0)/CONS(1)TAX FLAG = 1 option results in a negative NET REGULATED FIT of -$100.2 million in 1995, corresponding to capture of all accumulated ITC ($325.6 million). This results in an increase in Common Dividends from $476.9 million to $577.1 million and a decrease in the **END COMMON EQUITY** from $1235.5 million to $1135.3 million.

The use of the GROSS(0)/NET(1)PROPTXFLAG = 1 option increases the Property Tax expense in all but the first year, 1995, and results in higher revenue requirements and busbar costs (e.g., 25.5 cents/kWh versus 23.8 cents/kWh in 2024).

The use of the PRE(0)/AFTER(1)TXDISC FLAG = 1 option only affects calculation of levelized busbar costs. Both the current and base year discount rates are lower for this option: current, 7.26% versus 10.97%; base year, 0.99% versus 4.50%.

Kidder-Peabody LMFBR (Full;Cons;Gross;After) 10/25/83

CONSTRUCTION SUMMARY
(All Amounts in millions of current year $)

Period: (1987,1995)
Base Year: 1982

Expenses:	Total	Book Life	Book Dep.
Direct :	4070.5	30	135.7
Startup:	0.0	30	0.0
AFUDC :	1930.1	30	64.3
Wk.Cap.:	0.0	--	--
Total :	6000.5	--	200.0

Financing:	Total	% Financing	Charge Rate
Common :	1500.1	25.00	14.15
Preferred:	0.0	0.00	0.00
LT.Debt :	4500.4	75.00	9.92
Total :	6000.5	100.00	10.97

Accruals:
ITC: 325.6
FIT: 965.1

OPERATING SUMMARY
Acceleration Tax Depreciation:
Tax Life : 25 (years)
Method : Straight-line
Basis : 4070.5
TEFRA Opt.: Full

Option Flags
IDC Compounding : Simple
Financing Period : Quarter
Property Tax Basis : Gross
Equity Finan. Basis: Fixed %
Discount Rate : After-tax
Analysis Type : Bottom-up
Accounting Basis : Flow-thru
Reg. Tax Treatment : Consolidated

Analysis Summary:	ROI (%)	Lev. Busbar (cents/kWh)	Discount Rate (%)	Fixed Charge Rate (%)
Current:	20.01	18.70	7.26	15.13
Base Yr:	12.74	5.04	0.99	6.60

Kidder-Peabody LMFBR (Full;Cons;Gross;After) 10/25/83

BALANCE SHEET
(Millions of Current Year $)

	1995	2002	2010	2018	2024
ASSETS					
Cum. Working Capital	0.0	0.0	0.0	0.0	0.0
Gross Plant	4070.5	4070.5	4070.5	4070.5	4070.5
Less Accum.Plant Dep.	135.7	1085.5	2170.9	3256.4	4070.5
Net Plant	3934.8	2985.0	1899.5	814.1	-0.0
Capitalized AFUDC	1930.1	1930.1	1930.1	1930.1	1930.1
Less Accum.AFUDC Dep.	64.3	514.7	1029.4	1544.1	1930.1
Net Capitalized AFUDC	1865.7	1415.4	900.7	386.0	0.0
TOTAL ASSETS	5800.5	4400.4	2800.3	1200.1	-0.0
LIABILITIES					
LTD Due in Next Year	150.0	150.0	150.0	150.0	0.0
LTD Outstanding	4200.4	3150.3	1950.2	750.1	0.0
Cum. Deferred ITC	314.8	238.8	152.0	65.1	0.0
Cum. Deferred FIT	0.0	0.0	0.0	0.0	0.0
P.S. Due in Next Year	0.0	0.0	0.0	0.0	0.0
Pref.Stock Outstanding	0.0	0.0	0.0	0.0	0.0
End Common Equity	1135.3	861.3	548.1	234.9	-0.0
Partner Loans Outst.	0.0	0.0	0.0	0.0	0.0
Total Stock Equity	1135.3	861.3	548.1	234.9	-0.0
TOTAL LIABIL. & EQUITY	5800.5	4400.4	2800.3	1200.1	-0.0

Kidder-Peabody LMFBR (Full;Cons;Gross;After) 10/25/83

INCOME STATEMENT
(Millions of Current Year $)

	1995	2002	2010	2018	2024
REVENUES					
Electric Sales	1438.2	1299.6	1335.1	1521.0	1813.0
Federal Subsidy	0.0	0.0	0.0	0.0	0.0
TOTAL REVENUES	1438.2	1299.6	1335.1	1521.0	1813.0
EXPENSES					
Fuel Expense	156.5	238.5	385.8	624.3	895.7
O&M Expense	101.7	155.0	250.8	405.8	582.2
Backfit&Replacement	0.0	0.0	0.0	0.0	0.0
Ad Valorem Tax	0.0	0.0	0.0	0.0	0.0
Total Oper. Expense	258.2	393.5	636.6	1030.1	1477.9
Plant Book Dep.	135.7	135.7	135.7	135.7	135.7
AFUDC Book Dep.	64.3	64.3	64.3	64.3	64.3
Total Book Dep.	200.0	200.0	200.0	200.0	200.0
Property Tax	120.0	120.0	120.0	120.0	120.0
Net FIT (Book Basis)	201.4	116.6	72.2	27.9	-5.3
Total Pre-Debt Expen.	779.7	830.0	1028.9	1378.1	1792.6
Pre-Debt Net Income	658.5	469.6	306.2	142.9	20.4
LTD Interest	446.3	342.1	223.1	104.1	14.9
NET INCOME	212.3	127.4	83.1	38.8	5.5
Less Pref. Dividends	0.0	0.0	0.0	0.0	0.0
Common Stock Earnings	212.3	127.4	83.1	38.8	5.5
BUSBAR COST (cents/kWh)	20.2	18.3	18.8	21.4	25.5

Kidder-Peabody LMFBR (Full;Cons;Gross;After) 10/25/83

SOURCES AND USES OF FUNDS STATEMENT
(Millions of Current Year $)

	1995	2002	2010	2018	2024
SOURCES OF FUNDS					
Net Income	212.3	127.4	83.1	38.8	5.5
Plant Book Dep.	135.7	135.7	135.7	135.7	135.7
AFUDC Book Dep.	64.3	64.3	64.3	64.3	64.3
Total Book Dep.	200.0	200.0	200.0	200.0	200.0
Deferred ITC	314.8	-10.9	-10.9	-10.9	-10.9
Deferred FIT	0.0	0.0	0.0	0.0	0.0
TOTAL INTERNAL	727.1	316.6	272.3	227.9	194.7
Partner Loans	0.0	0.0	0.0	0.0	0.0
Government Loans	0.0	0.0	0.0	0.0	0.0
TOTAL SOURCES	727.1	316.6	272.3	227.9	194.7
USES OF FUNDS					
Preferred Stock Due	0.0	0.0	0.0	0.0	0.0
Preferred Dividends	0.0	0.0	0.0	0.0	0.0
Common Dividends	577.1	166.6	122.2	77.9	44.7
Working Capital Change	0.0	0.0	0.0	0.0	0.0
LTD Repayment	150.0	150.0	150.0	150.0	150.0
Partner Loan Repayment	0.0	0.0	0.0	0.0	0.0
TOTAL USES	727.1	316.6	272.3	227.9	194.7

Kidder-Peabody LMFBR (Full;Cons;Gross;After) 10/25/83

RETAINED EARNINGS STATEMENT & TAX DEP. WORKSHEET
(Millions of Current Year $)

	1995	2002	2010	2018	2024
RETAINED EARNINGS					
Begin.Common Equity	1500.1	900.4	587.2	274.0	39.1
Common Stock Earnings	212.3	127.4	83.1	38.8	5.5
Total Common Equity	1712.4	1027.9	670.3	312.8	44.7
Less Common Dividends	577.1	166.6	122.2	77.9	44.7
END COMMON EQUITY	1135.3	861.3	548.1	234.9	-0.0
RETURN ON EQUITY (%)	14.1	14.1	14.1	14.1	14.1
TAX DEP. WORKSHEET					
Startup Dep.	0.0	0.0	0.0	0.0	0.0
5-year Prop.Dep.	0.0	0.0	0.0	0.0	0.0
10-year Prop.Dep.	0.0	0.0	0.0	0.0	0.0
15-year Prop.Dep.	0.0	0.0	0.0	0.0	0.0
Straight-line Dep.	162.8	162.8	162.8	162.8	0.0
TOTAL TAX DEPRECIATION	162.8	162.8	162.8	162.8	0.0

Kidder-Peabody LMFBR (Full;Cons;Gross;After) 10/25/83

TAX CALCULATION (Book/Regulation Basis)
(Millions of Current Year $)

	1995	2002	2010	2018	2024
BOOK TAX CALCULATION					
Electric Sales	1438.2	1299.6	1335.1	1521.0	1813.0
Total Oper. Expense	258.2	393.5	636.6	1030.1	1477.9
Total Book Dep.	200.0	200.0	200.0	200.0	200.0
Property Tax	120.0	120.0	120.0	120.0	120.0
LTD Interest	446.3	342.1	223.1	104.1	14.9
Pre-tax Book Income	424.5	254.8	166.2	77.6	11.1
Book FIT	212.3	127.4	83.1	38.8	5.5
Less amort.ITC	10.9	10.9	10.9	10.9	10.9
NET FIT (Book Basis)	201.4	116.6	72.2	27.9	-5.3
REGULATORY TAX CALCULATION					
Electric Sales	1438.2	1299.6	1335.1	1521.0	1813.0
Total Oper. Expense	258.2	393.5	636.6	1030.1	1477.9
Total Tax Depreciation	162.8	162.8	162.8	162.8	0.0
Property Tax	120.0	120.0	120.0	120.0	120.0
LTD Interest	446.3	342.1	223.1	104.1	14.9
Pre-tax Part. Income	450.9	281.2	192.5	103.9	200.2
FIT Due	225.5	140.6	96.3	52.0	100.1
Less ITC Taken	325.6	0.0	0.0	0.0	0.0
Less FIT Taken	0.0	0.0	0.0	0.0	0.0
NET REGULATED FIT	-100.2	140.6	96.3	52.0	100.1

Kidder-Peabody LMFBR (Full;Cons;Gross;After) 10/25/83

PARTNERSHIP RETURN AND CASHFLOW ANALYSIS
(Millions of Current Year $)

	1995	2002	2010	2018	2024
PARTNERSHIP RETURN					
Electric Sales	1438.2	1299.6	1335.1	1521.0	1813.0
Total Oper. Expense	258.2	393.5	636.6	1030.1	1477.9
Total Tax Depreciation	162.8	162.8	162.8	162.8	0.0
Property Tax	120.0	120.0	120.0	120.0	120.0
LTD Interest	446.3	342.1	223.1	104.1	14.9
Pre-tax Part. Income	450.9	281.2	192.5	103.9	200.2
FIT Due	225.5	140.6	96.3	52.0	100.1
PARTNERSHIP CASHFLOW ANALYSIS					
After-tax Cash	225.4	140.6	96.3	52.0	100.1
Total Tax Depreciation	162.8	162.8	162.8	162.8	0.0
Federal Subsidy	0.0	0.0	0.0	0.0	0.0
Total Cash Avail.	388.3	303.4	259.1	214.8	100.1
Less LTD Repayment	150.0	150.0	150.0	150.0	150.0
Less Pref. Dividends	0.0	0.0	0.0	0.0	0.0
Less Pref. Stock Due	0.0	0.0	0.0	0.0	0.0
Less Partner Loans	0.0	0.0	0.0	0.0	0.0
Less Work.Cap. Change	0.0	0.0	0.0	0.0	0.0
NET TO PARTNERS	238.2	153.4	109.1	64.8	-49.9
CUMULATIVE TO PARTNERS	28.8	1218.9	2246.6	2919.7	[illegible]

Kidder-Peabody LMFBR (Full;Cons;Gross;After) 10/25/83

AFTER-TAX CASH FLOWS TO UTILITY INVESTORS
(Millions of Current Year $)

Year	During Construction Cash Inv.	Tax Ben.	During Operation Tax Due	Cash Dist.	Net A.T. Cash Flow	Cum.A.T. Cash Flow	Cm.Dis.R. A.T.C.F.
1987	165.218	65.044	--	--	-100.174	-100.174	-68.756
1988	223.374	111.764	--	--	-111.609	-211.784	-140.246
1989	275.294	160.764	--	--	-114.530	-326.313	-208.643
1990	233.902	180.700	--	--	-53.202	-379.515	-238.183
1991	171.600	185.545	--	--	13.945	-365.570	-231.011
1992	108.487	181.822	--	--	73.335	-292.235	-196.045
1993	108.038	194.164	--	--	86.126	-206.109	-157.903
1994	214.222	210.894	--	--	-3.328	-209.437	-159.273
1995	--	--	225.450	463.697	238.246	28.809	-67.984
1996	--	--	173.831	360.460	186.629	215.438	-1.311
1997	--	--	168.291	349.380	181.089	396.527	59.006
1998	--	--	162.751	338.301	175.549	572.076	113.522
1999	--	--	157.211	327.221	170.010	742.086	162.746
2000	--	--	151.671	316.141	164.470	906.556	207.145
2001	--	--	146.131	305.062	158.930	1065.486	247.146
2002	--	--	140.591	293.982	153.391	1218.877	283.140
2003	--	--	135.051	282.902	147.851	1366.727	315.488
2004	--	--	129.512	271.823	142.311	1509.039	344.517
2005	--	--	123.972	260.743	136.771	1645.810	370.529
2006	--	--	118.432	249.663	131.232	1777.042	393.799
2007	--	--	112.892	238.584	125.692	1902.734	414.578
2008	--	--	107.352	227.504	120.152	2022.886	433.098
2009	--	--	101.812	216.424	114.612	2137.498	449.569
2010	--	--	96.272	205.345	109.073	2246.571	464.184
2011	--	--	90.732	194.265	103.533	2350.104	477.118
2012	--	--	85.192	183.186	97.993	2448.098	488.531
2013	--	--	79.652	172.106	92.454	2540.552	498.571
2014	--	--	74.112	161.026	86.914	2627.466	507.370
2015	--	--	68.572	149.946	81.374	2708.840	515.052
2016	--	--	63.032	138.867	75.834	2784.674	521.726
2017	--	--	57.492	127.787	70.295	2854.969	527.494
2018	--	--	51.952	116.707	64.755	2919.724	532.448
2019	--	--	46.412	105.628	59.215	2978.939	536.672
2020	--	--	122.283	94.548	-27.735	2951.204	534.828
2021	--	--	116.743	83.468	-33.275	2917.929	532.764
2022	--	--	111.203	72.389	-38.815	2879.115	530.520
2023	--	--	105.663	61.309	-44.354	2834.761	528.130
2024	--	--	100.123	50.229	-49.894	2784.867	525.622
TOT.	1500.134	1290.697	3424.389	6418.690	2784.867		

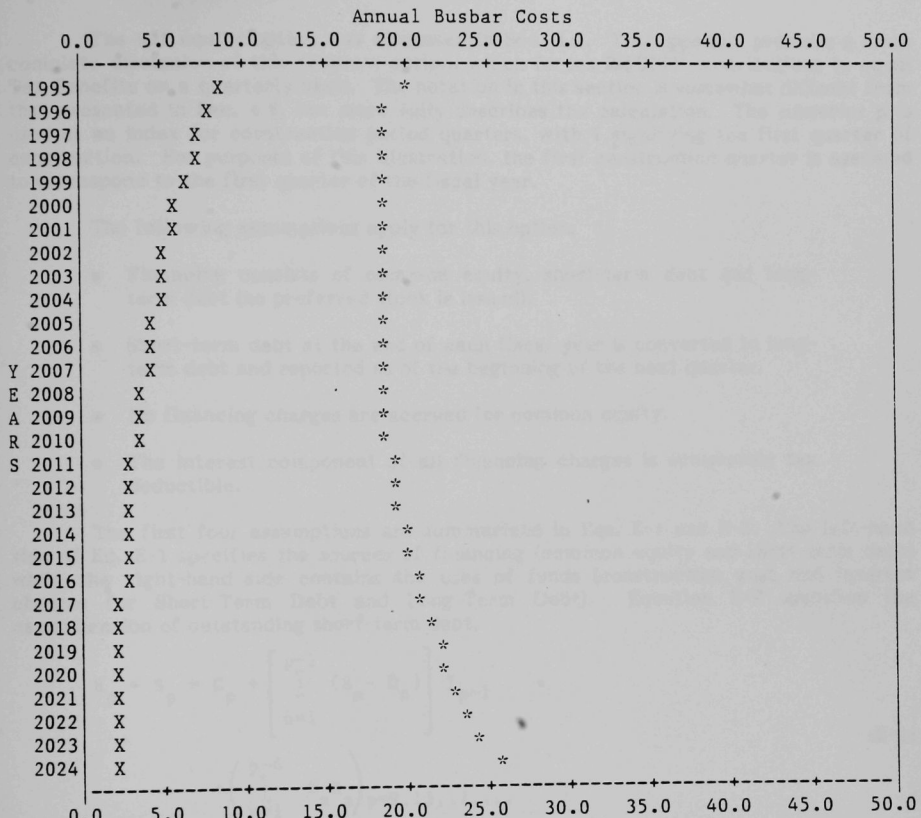
Summary of ROIs (Current $)

1995	1.32	1996	7.52	1997	11.24	1998	13.66	1999	15.33
2000	16.51	2001	17.37	2002	18.01	2003	18.48	2004	18.84
2005	19.11	2006	19.32	2007	19.49	2008	19.61	2009	19.71
2010	19.78	2011	19.84	2012	19.89	2013	19.92	2014	19.95
2015	19.97	2016	19.99	2017	20.00	2018	20.01	2019	20.02
2020	20.02	2021	20.01	2022	20.01	2023	20.01	2024	20.01

Summary of ROIs (Base-Year $)

1995	-4.93	1996	0.90	1997	4.41	1998	6.71	1999	8.29
2000	9.41	2001	10.22	2002	10.83	2003	11.28	2004	11.62
2005	11.88	2006	12.08	2007	12.24	2008	12.36	2009	12.45
2010	12.52	2011	12.58	2012	12.63	2013	12.66	2014	12.69
2015	12.71	2016	12.72	2017	12.74	2018	12.75	2019	12.75
2020	12.75	2021	12.75	2022	12.74	2023	12.74	2024	12.74

Kidder-Peabody LMFBR (Full;Cons;Gross;After) 10/25/83



Summary of Levelized Busbar Costs
(Levelized over 20 years)

18.70 (current cents/kWh,discount rate : 7.256 (%))
5.04 (base yr. cents/kWh,discount rate: 0.994 (%))

Summary of Busbar Costs (current cents/kWh)

1995	20.21	1996	18.77	1997	18.65	1998	18.54	1999	18.44
2000	18.36	2001	18.30	2002	18.26	2003	18.24	2004	18.24
2005	18.26	2006	18.30	2007	18.38	2008	18.47	2009	18.60
2010	18.76	2011	18.95	2012	19.17	2013	19.43	2014	19.73
2015	20.07	2016	20.46	2017	20.89	2018	21.37	2019	21.90
2020	22.49	2021	23.14	2022	23.85	2023	24.62	2024	25.47

Summary of Busbar Costs (base yr. cents/kWh)

1995	8.92	1996	7.80	1997	7.30	1998	6.83	1999	6.40
2000	6.00	2001	5.63	2002	5.29	2003	4.97	2004	4.68
2005	4.42	2006	4.17	2007	3.94	2008	3.73	2009	3.54
2010	3.36	2011	3.19	2012	3.04	2013	2.90	2014	2.78
2015	2.66	2016	2.55	2017	2.45	2018	2.36	2019	2.28
2020	2.21	2021	2.14	2022	2.07	2023	2.02	2024	1.96

APPENDIX E: DOCUMENTATION FOR THE NET EQUITY OPTION

The net equity option was discussed in Sec. 4.2. This appendix presents a more complete discussion of this program option, which forces EQUITY contributions to equal Tax Benefits on a quarterly basis. The notation in this section is somewhat diffrent from that presented in Sec. 4.2, but more fully describes the calculation. The subscript p is used as an index for construction period quarters, with 1 signifying the first quarter of construction. For purposes of this illustration, the first construction quarter is assumed to correspond to the first quarter of the fiscal year.

The following assumptions apply for this option:

- Financing consists of common equity, short-term debt and long-term debt (no preferred stock is issued).
- Short-term debt at the end of each fiscal year is converted to long-term debt and reported as of the beginning of the next quarter.
- No financing charges are accrued for common equity.
- The interest component of all financing charges is completely tax deductible.

The first four assumptions are summarized in Eqs. E-1 and E-2. The left-hand side of Eq. E-1 specifies the sources of financing (common equity and short-term debt) while the right-hand side contains the uses of funds (construction cost and interest charges for Short-Term Debt and Long-Term Debt). Equation E-2 specifies the capitalization of outstanding short-term debt.

$$E_p + S_p = C_p + \left[\sum_{n=1}^{p-1} (S_n - D_n) \right] I_{p-1} + \left(\sum_{n=1}^{p-4} D_n R_n \right)_{p=9,13,17,\ldots} \tag{E-1}$$

$$D_n = \sum_{n=p-4}^{p-1} S_n \quad \text{for } p = 5,9,13,\ldots \tag{E-2}$$

where:

E_p = Common equity provided during the present period

S_p = Short-term debt incurred in present period

C_p = Total construction cost during present period

S_n = Short-term debt incurred in period n

I_{p-1} = Short-term interest rate for the previous period

D_n = Long-term debt incurred in period n

R_n = Annual weighted cost of long-term debt in period n

Equation E-3 equates the common equity financing to the sum of investment tax credit and interest during construction tax savings, pro-rated quarterly.

$$E_p = r_p C_p + T \left[\left(\sum_{n=1}^{p-1} S_n - \sum_{n=1}^{p} D_n \right) X\ I_p + S_p I_p + \left(\sum_{n=1}^{p-3} D_n \right) \quad p=8,12,16,\ldots \right] \tag{E-3}$$

where:

r_p = Investment tax credit rate (Eq. 4-26, 4-27)

T = Effective income tax rate

I_p = Short-term interest rate for the present period

S_p = Short-term debt incurred in the present period

Elimination of E_p from Eqs. E-1 and E-3 yields the Short-Term Debt requirements, S_p, given in Eq. E-4:

$$S_p = \frac{1}{1 + TI_p} \left\{ C_p(1 - r_p) + I_{p-1} \left[\sum_{n=1}^{p-1} (S_n - D_n) \right] - TI_p \left(\sum_{n=1}^{p-1} S_n - \sum_{n=1}^{p} D_n \right) + \left[\sum_{n=1}^{p-4} D_n R_n \right] p=9,13,17,\ldots - T \left[\sum_{n=1}^{p-3} D_n R_n \right] p=8,12,16,\ldots \right\} \tag{E-4}$$

Finally, E_p can be solved for by substitution of E-4 into E-1 or E-3.

ARGONNE NATIONAL LAB WEST
3 4444 00012852 0

X

B